KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT ECONOMIE EN
BEDRIJFSWETENSCHAPPEN

Profit driven data mining in massive customer networks: new insights and algorithms

Proefschrift voorgedragen tot
het behalen van de graad van
Doctor in de Toegepaste
Economische Wetenschappen

door

Wouter VERBEKE

Nummer 379 2012

Committee

Chairman	Prof. dr. Jan Vanthienen	K.U.Leuven
Promotor	Prof. dr. Bart Baesens	K.U.Leuven
	Prof. dr. ir. David Martens	U.Antwerpen
	Prof. dr. Dirk Van den Poel	U.Gent
	Prof. dr. Foster Provost	New York University
	Prof. dr. Sunil Gupta	Harvard University

Acknowledgments

It is the glory of God to hide things,
But the glory of kings to investigate them.

Proverbs, 25:2

First and foremost I would like to thank my promotor, prof. dr. Bart Baesens, for the exceptional opportunities he has offered me, for the intelligent and efficient guidance, for the wise and caring advice, for the warm and welcome support, and for much more. Bart, it has been an unbelievable pleasure and adventure to work for and with you.

Prof. dr. David Martens has been a major source of inspiration throughout the realization of this doctoral project. I would like to thank him for the persistent interest in my work, for his passion and engagement, for the many in-depth and bright remarks, and for the many laughs we had.

I am also most grateful to Prof. dr. Jan Vanthienen, prof. dr. Dirk Van den Poel, prof. dr. Foster Provost, and prof. dr. Sunil Gupta for evaluating and assessing this dissertation, and for formulating many useful suggestions and constructive remarks. Many colleagues have helped and supported me during the past years, I would like to sincerely thank all of them. Two of them have become true *compagnons de route*: Pieter Hens and Thomas Verbraken. We have spent many hours discussing the world and beyond, relieving acute doctoral distress.

I deeply cherish and appreciate the everlasting love and support of my parents, brothers and sister, and my parents, brothers and sisters in law. Last but not least, I want to thank a strong, utmost beautiful, elegant, ravishing and graceful young lady: the girl who has become my wife, Luit.

Contents

Samenvatting

Door meten tot weten

Heike Kamerlingh Onnes (1853 - 1926) was een Nederlands experimenteel natuurkundige en laureaat van de nobelprijs voor de natuurkunde. Zijn slagzin was *door meten tot weten*. De wetenschappelijke bedrijfsvoering heeft deze rationele benaderings- en deductiewijze geadopteerd en vertaald naar een bedrijfscontext, met als doel het optimaliseren van bedrijfsprocessen op basis van kwantitatieve informatie. De ontwikkeling van de moderne informatie- en communicatietechnologieën heeft echter geleid tot een overvloed aan beschikbare gegevens en informatie, die de synthese van nuttige kennis uit deze gegevens sterk bemoeilijkt. Met andere woorden, meten alleen volstaat niet om tot weten te komen. Bijgevolg is een behoefte aan nieuwe analyse, planning, en synthesemethoden ontstaan om de overvloed aan beschikbare informatie te transformeren in waardevolle kennis en accurate beslissingen.

Deze doctoraatsverhandeling ontwikkelt zulke methoden, met name data mining technieken voor classificatie, en past deze toe om individueel klantverloop in de telecommunicatie sector te voorspellen. Data mining betreft het geautomatiseerd distilleren van bruikbare patronen en nuttige kennis uit gestructureerde databanken. Classificatie betreft het gebruik van deze patronen en kennis om de waarde van een discrete doelvariabele te voorspellen, bijvoorbeeld of een klant al dan niet zal verlopen. Data mining methoden omvatten een breed gamma aan heuristieken en modeleringstechnieken afkomstig uit wetenschapsdomeinen zoals statistiek, artificiële intelligentie, machine learning, algorithmic computing, en vele andere. Klantverloop is een groeiend probleem waar telecom operatoren, maar ook internet providers, televisie- en energiedistributeurs, banken, verzekeraars, en

vele andere bedrijven in groeiende mate mee geconfronteerd worden als een gevolg van de toenemende concurrentie, en dat de winstgevendheid van de bedrijfsactiviteiten ernstig bedreigt. Telecom operatoren rapporteren een jaarlijks klantverloop dat kan oplopen tot twintig procent van het totale klantenbestand, en in sommige deelsegmenten zelfs tot veertig procent en hoger. De totale kost van klantverloop in de telecom sector in Noord Amerika en Europa samen word ruwweg geschat op vier miljard euro per jaar.

Om het klantenbestand en voornamelijk de bedrijfsomzet op peil te houden, en omdat het aantrekken van nieuwe klanten duurder blijkt dan het weerhouden van bestaande klanten, besteden bedrijven meer en meer aandacht aan het bestrijden van klantverloop. Hiertoe worden retentie campagnes opgezet, die klanten in ruil voor hun trouw beloont. Concreet wordt aan klanten een incentive met een geldelijke waarde aangeboden om *trouw* te blijven en hun klantrelatie met het bedrijf verder te zetten, en dus om te vermijden dat ze verlopen. Om de efficiëntie van deze retentie campagnes te verhogen worden predictie modellen ingezet die voorspellen welke klanten de grootste waarschijnlijkheid vertonen om te verlopen. Zodoende kan de kost van deze campagnes gereduceerd worden door enkel aan klanten die met grote waarschijnlijkheid zullen verlopen een incentive aan te bieden.

Martens (2008a) introduceerde het begrip *aanvaardbare* classificatiemodellen, met aanvaardbaar gedefinieerd door middel van drie vereisten die opgelegd worden en betrekking hebben op:

- de precisie of de voorspellende kracht van het model,
- de begrijpelijkheid van het model, m.a.w. zijn de motieven van een model om een voorspelling te maken interpreteerbaar,
- de correctheid van het model met betrekking tot domeinkennis, m.a.w. zijn de motieven van een model om een voorspelling te maken intuïtief correct).

De bijdragen van deze doctoraatsverhandeling worden in de volgende secties verduidelijkt aan de hand van deze drie vereisten.

Begrijpelijke en intuïtieve predictie modellen

Regel-gebaseerde classificatietechnieken resulteren in regel sets, en leiden aldus tot begrijpbare modellen. In het eerste deel van deze thesis wordt de wenselijkheid en de afdwingbaarheid van begrijpelijkheid en intuïtieve correctheid van een klantverloop predictiemodel onderzocht. Daartoe worden in een eerste studie enkele recent ontwikkelde regel inductie technieken toegepast om klantverloop te voorspellen, gebruik makend van een publiek beschikbare data set.

In een tweede studie wordt een nieuwe techniek ontwikkeld, RULEM genaamd, die toelaat de intuïtieve correctheid van regel gebaseerde classificatiemodellen af te dwingen, zodat ze niet in tegenspraak zijn met de aanwezige expert- of domeinkennis. Domeinkennis wordt steevast uitgedrukt als een monotoon verband tussen de verklarende attributen in het model en de voorspelde doelvariabele. Bijvoorbeeld, hoe duurder het piektarief dat is vastgelegd in het abonnement van een klant, hoe groter de kans dat hij zal verlopen naar een andere, goedkopere operator. Bijgevolg kan van een predictief model verwacht worden dat het een positief monotoon verband incorporeert tussen de waarde van het verklarende attribuut *piektarief* en de voorspelde *probabiliteit te verlopen*. Want indien het model voorspellingen maakt in tegenspraak met deze kennis, zal het niet aanvaard worden door de gebruiker. Het incorporeren of opleggen van monotone verbanden laat toe om tot intuïtief correcte modellen te komen, en vindt toepassingen in vele domeinen ook buiten het voorspellen van klantverloop.

De ontwikkelingen en toepassingen voorgesteld in dit eerste deel hebben geleid tot de volgende wetenschappelijke artikels:

Verbeke, W., Martens, D., Mues, C., Baesens, B., 2011e. Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Systems with Applications 38 (3), 2354–2364

Verbeke, W., Martens, D., Baesens, B., 2011c. Rulem: Rule learning with monotonicity constraints for ordinal classification. IEEE Transactions on data and knowledge engineering, under review

Preciezere predictie modellen

Een tweede deel van deze verhandeling betreft de precisie van predictie modellen. Een eerste studie vergelijkt de voorspellende kracht van een uitgebreide waaier aan classificatietechnieken om klantverloop te voorspellen. Hiertoe zijn twaalf data sets verzameld van telecom operatoren over de gehele wereld. Ook de voorspellende waarde van verschillende types van informatie wordt onderzocht in deze studie, en een nieuwe performantie maatstaf geïntroduceerd die toelaat om klantverloop predictie modellen te evalueren vanuit een bedrijfsstandpunt, als de maximale opbrengst die op basis van het model met een retentiecampagne gegenereerd kan worden.

Een tweede studie behandelt de exploratie van sociale netwerk informatie voor het voorspellen van klantverloop. Telecom operatoren beschikken over belgegevens van hun klanten, wat toelaat om het sociale netwerk van deze klanten benaderend na te bootsen. De studie ontwikkelt een set van technieken die het mogelijk maken om van een *belnetwerk* te leren en voorspellingen te maken op basis van de informatie die impliciet aanwezig is in een netwerk. De voorgestelde technieken zijn ontworpen om met gigantische belnetwerken bestaande uit miljoenen klanten om te gaan, door de computationele complexiteit te reduceren en gebruik te maken van geavanceerde rekentechnieken. Daarnaast wordt ook de tijdsdimensie van klantverloop geïncorporeerd binnen bestaande technieken, en wordt een complementaire methode voorgesteld die toelaat om hogere orde effecten in een netwerk in rekening te brengen. De studie toont aan dat deze nieuwe methoden de precisie van de bestaande generatie van modellen kan verhogen. Gebruik makend van de nieuw ontwikkelde maatstaf wordt aangetoond dat de aldus gegenereerde winsten substantieel kunnen vergroot worden.

De ontwikkelingen en toepassingen voorgesteld in het tweede deel van deze verhandeling hebben geleid tot de volgende wetenschappelijke artikels:

Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B., 2011a. New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. European Journal of Operational Research, doi 10.1016/j.ejor.2011.09.031

Verbeke, W., Martens, D., Baesens, B., 2011d. Social network analysis for customer churn prediction. Management Science, under review

Dedicated to Luit,
The girl who makes the world turn.

Chapter 1

Introduction

Where is the wisdom we have lost in knowledge?
Where is the knowledge we have lost in information?

T.S. Eliot (1888 - 1965), The Rock (1934)

1.1 The numbers tell the tale

Friedrich Wilhelm Heinrich Alexander Freiherr von Humboldt (1769 - 1859) was a German naturalist and scientific explorer who traveled the planet to *measure the world*. He strongly believed that by measuring the physical characteristics of natural phenomena we would come to understand their nature and workings, and eventually grow insight in the basic rules that govern the natural world. Basically, he believed that the numbers tell the tale.

The ideas and principles of von Humboldt and kindred spirits lived on and influenced science as well as society. The quantitative and empirical research method has been generally adopted by the scientific community, and in today's complex economical society companies rely on quantitative information to make daily decisions in a rational and optimal manner. However, because of the intricate and multifaceted nature of managerial problems and the vast amounts of data that are available to companies and organi-

zations, the numbers alone do no longer suffice to gain insight and to make sound decisions. In other words, the numbers hide the tale. The knowledge we seek is lost in the available information. Therefore, tools and methods are needed to disclose this knowledge and to make effective and efficient decisions based on the available information.

Management science, also referred to as decision science or operations research, is an interdisciplinary branch of applied mathematics devoted to optimal decision planning on quantitative grounds. It uses various scientific research-based principles, strategies, and analytical methods including mathematical modeling, statistics, and numerical algorithms to improve an organization's ability to enact rational and meaningful management decisions by arriving at optimal or near optimal solutions to complex decision problems. In short, management sciences help businesses and organizations to achieve their goals using scientific methods (Beer, 1968). The scope of this dissertation encompasses one specific family of such scientific methods within the field of management science, i.e., Knowledge Discovery in Databases (KDD). Data mining can be found at the heart of KDD and management science, and involves extracting interesting patterns from data (Baesens et al., 2009).

1.2 Data mining and knowledge discovery

With the advent and rise of Information and Communication Technology (ICT), and more specifically the rapid evolution in data collection and storage technology, companies and organizations are able to store vast amounts of information on a broad range of production processes and customer transactions at low cost. Knowledge Discovery in Databases entails the extraction of valuable knowledge from raw data. Data mining, which is an integral part of knowledge discovery in databases, is the process of automatically discovering useful information in large data repositories (Tan et al., 2006). Data mining techniques comprise a range of heuristics and modeling techniques coming from fields such as statistics, artificial intelligence, machine learning, and algorithmic computing.

The two main data mining tasks are *prediction* and *description*. The prediction task concerns the estimation of an unknown value of a target variable or a class attribute pertaining to an instance, based on the known data attributes of this instance. In the case of a discrete target variable, the

prediction task is called *classification*, and in case of a continuous target variable this task is called *regression*. Descriptive data mining on the other hand mainly regards deriving human interpretable patterns that describe the data, and includes tasks such as *clustering* and *association analysis*. The main topic of this dissertation concerns classification.

Three main requirements have been identified (Martens, 2008a) that are applicable to classification models, which together constitute a *holistic* view on data mining. These requirements concern:

1. the predictive power,
2. the comprehensibility,
3. and the justifiability.

Predictive power, also referred to as discrimination power or classification performance, concerns the ability of a model to make accurate or truthful predictions towards the future. Since the predictions made by a classification model are effectively implemented in business settings for decision making, the correctness of the predicted labels usually has a direct impact on the efficiency and the efficacy of the decisions made by a company. As such, the predictive power of a classification model is of major importance and therefore has received much attention in the scientific literature.

However, a successful implementation of a data mining model as a decision making tool in a business context usually heavily depends on the comprehensibility and the justifiability of the model. Comprehensibility or interpretability is required when the model needs to be human-interpretable, in order to allow the user to understand the grounds upon which a model classifies an instance. A comprehensible model is also called a white-box model, in contrast to black-box models which are not or hardly interpretable.

A justifiable model is a model that is intuitively correct and in line with domain knowledge. The justifiability of a comprehensible model can be checked by a human user. However, this may not be a straightforward task in case of very large models, or in case of very complex models that include a large number of predictive attributes. Therefore, it may be preferable to automate this process, in order to avoid human error as well as to improve efficiency. A black-box model on the other hand cannot be interpreted by a human user and requires additional processing in order to check whether

it is in line with domain knowledge. Therefore, whenever justifiability is demanded, typically comprehensibility will be required as well, although not necessarily.

Classification models that meet these requirements are called *acceptable* for implementation (Martens and Provost, 2011). From a theoretical perspective, this dissertation aims to develop data mining techniques that allow to induce acceptable classification models. The holistic view on data mining serves as a general framework that interlinks the chapters of this thesis. Each chapter involves a stand-alone research project, and as such can be read separately. The contributions of each project can be related straightforward to the three requirements that are formulated in this section. From an application perspective, this dissertation examines the use of data mining techniques in a business context, and more specifically in the field of customer churn prediction in the telecommunication (telco) sector. As such, new insights are generated, as well as new challenges brought to light.

1.3 Customer churn prediction in the telecommunication sector

The telecommunication industry is a highly technological sector which has developed tremendously over the past two decades as a result of the emergence and commercial success of both mobile telecommunication and the internet. During the last decade, the number of mobile phone users has increased dramatically. At the end of 2010 the number of mobile phone users worldwide exceeded four billion[1], which is over 60% of the world population. Telco operators have gained millions of new customers over the past years and built enormous customer bases. However, wireless telecommunication markets are getting saturated, particularly in the developed countries, and mobile phone penetration rates are stagnating. Many Western countries already have mobile phone penetration rates above 100%, meaning there are more subscriptions than inhabitants. Moreover, a strong competition exist between operators in order to attract new customers, and telco operators have reported annual churn rates of up to 40% of the customer base.

Therefore, customer retention receives a growing amount of attention

[1] www.eito.com

from telco operators as a means to reduce customer churn and to control the related costs, and thus in order to safeguard profitability. It has been shown in the literature that customer retention is profitable to a company because: (1) acquiring new clients costs five to six times more than retaining existing customers (Bhattacharya, 1998; Rasmusson, 1999; Colgate et al., 1996; Athanassopoulos, 2000); (2) long-term customers generate higher profits, tend to be less sensitive to competitive marketing activities, become less costly to serve, and may provide new referrals through positive word-of-mouth, while dissatisfied customers might spread negative word-of mouth (Mizerski, 1982; Stum and Thiry, 1991; Reichheld, 1996; Zeithaml et al., 1996; Colgate et al., 1996; Paulin et al., 1998; Ganesh et al., 2000); (3) losing customers leads to opportunity costs because of reduced sales (Rust and Zahorik, 1993). A small improvement in customer retention can therefore lead to a significant increase in profit (Lariviere and Van den Poel, 2005).

Most wireless telco providers already operate a customer churn prediction (CCP) model that indicates the customers with the highest propensity to attrite. This allows an efficient customer management, and a better allocation of the limited marketing resources for customer retention campaigns. Customer churn prediction models are typically, but not exclusively, applied in contractual settings, such as the postpaid segment in the wireless telco industry. In a contractual setting usually more information is at hand than in a non-contractual setting, such as the prepaid segment which consists mostly of anonymous customers. Various types of information can be used to predict customer attrition, such as for instance socio-demographic data (e.g., sex, age, or zip code) and call behavior statistics (e.g., the number of international calls, billing information, or the number of calls to the customer helpdesk). Alternatively, social network information extracted from call detail record (CDR) data can be explored to predict churn (Dasgupta et al., 2008), which is especially interesting if no other information is available.

A number of challenges are presented when applying data mining techniques to predict customer churn:

1. A first challenge concerns the massive size of the customer base of telco operators, which typically contain millions of customers. In order to process the arising data cascades, computationally efficient algorithms are required both in terms of implementation and design. Although

computing power is available in large quantities and at low cost, the size of the data may be prohibitive for application of computationally inefficient algorithms, as will be illustrated in Chapter 5 of this dissertation.

2. A second challenge is posed by the uneven class distribution of churners and non-churners. Typically, retention campaigns are executed at least on a monthly basis, and the fraction of the customers that churns during this period is much smaller than the fraction of customers that does not churn. This results in a heavily skewed class distribution, which may cause data mining techniques to experience difficulties in learning powerful classification models.

3. Finally, a third challenge relates to the time dimension which is essential to the process of customer churn. Data mining techniques therefore have to explicitly incorporate the temporal aspect when learning patterns from the data.

These three challenges require specific adjustments to data mining techniques in order to be applicable in a customer churn prediction setting, and are handled in this dissertation as outlined in the next section.

1.4 Outline and contributions

This section sets forth the main contributions of each chapter, which can be linked to: (1) the three main requirements that are applicable to classification models, as introduced in Section 1.2; (2) the three challenges that arise when applying data mining for customer churn prediction, as discussed in Section 1.3. As such, this dissertation contributes both from a theoretical and an application point of view.

1.4.1 Chapter 2

Chapter 2 takes a closer look at the merits of rule-based learning techniques for customer churn prediction.

- The chapter provides an extended overview of the literature on the use of data mining in CCP modeling. It is shown that only limited attention has been paid to the comprehensibility and the intuitiveness or justifiability aspect of churn prediction models.

- Two recently proposed data mining techniques, AntMiner+ and ALBA, are applied to churn prediction modeling, and benchmarked to traditional rule induction techniques such as C4.5 and RIPPER. AntMiner+ and ALBA induce comprehensible classification rule-sets.

- AntMiner+ is a high performing data mining technique based on the principles of Ant Colony Optimization, and allows to incorporate domain knowledge by imposing monotonicity constraints on the final rule-set. The need for justifiability will be examined by assessing rule sets induced by AntMiner+ and Ripper, a state-of-the-art rule induction technique.

- ALBA combines the strong predictive power of a non-linear support vector machine (SVM) with the comprehensibility of the rule-set format. Its use in a customer churn prediction setting as a means to boost the performance of rule induction techniques is examined.

Chapter 2 has been published in:

Verbeke, W., Martens, D., Mues, C., Baesens, B., 2011e. Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Systems with Applications 38 (3), 2354–2364

1.4.2 Chapter 3

Chapter 3 presents a novel algorithm, i.e., RULEM, which allows to incorporate domain knowledge in rule-based ordinal classification models by enforcing monotone relations between predictor variables and the class variable.

- An extensive literature overview is provided on monotone classification approaches.

- An innovative approach is developed which guarantees monotonicity during a postprocessing step, i.e., after the induction of the classification model, and which can therefore be applied in combination with any rule- or tree-based classification technique.

- The algorithm checks whether a rule set or decision tree violates the imposed monotonicity constraints, and existing violations are resolved by inducing a set of additional rules which enforce monotone classification.
- The algorithm is able to handle non-monotonic noise, and can be applied to both partially and totally monotone problems with an ordinal target variable.
- Based on the RULEM concept, two novel justifiability measures are introduced which allow to calculate the extent to which a classification model is in line with domain knowledge expressed in the form of monotonicity constraints.
- An extensive benchmarking experiment is conducted to evaluate the performance of the RULEM algorithm on an extensive set of ordinal classification problems.

Chapter 3 has been submitted for publication in:

Verbeke, W., Martens, D., Baesens, B., 2011c. Rulem: Rule learning with monotonicity constraints for ordinal classification. IEEE Transactions on data and knowledge engineering, under review

1.4.3 Chapter 4

Chapter 4 conducts an extensive benchmarking experiment to assess the state-of-the-art in customer churn prediction. A full factorial experimental design is applied to test the impact of three factors on the predictive power of CCP models:

- A wide range of classification techniques are evaluated from different families of techniques, including decision trees, rule induction techniques, statistical classifiers, ensemble methods, and neural networks.
- Given the skewed class distribution, which is typical for customer churn data sets, the training data sets in the experiments have been oversampled to test the impact of oversampling on the classification performance.

- A wrapper approach for feature selection has been applied to reduce the number of variables included in the classification models, in order to test the impact of feature selection on the predictive power.

In order to allow rigorous statistical testing of the results, all techniques and procedures will be applied to eleven real-life data sets obtained from international telco operators. The framework described by Demšar (2006) will be applied to correctly compare and assess the performance of the induced models in order to draw valid conclusions about the impact of each factor.

Furthermore, Chapter 4 develops a novel, profit centric performance measure by calculating the maximum profit that can be generated by including the optimal fraction of customers with the highest predicted probabilities to attrite in a retention campaign. The novel measure is applied to select the optimal model and fraction of customers to include from a business perspective, and the resulting profits are compared to the results using statistically based performance measures. Finally, an extensive literature review is provided which discusses the existing business oriented approaches to evaluate CCP models that have been proposed in the literature.

Chapter 4 has been published in:

Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B., 2011a. New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. European Journal of Operational Research, doi 10.1016/j.ejor.2011.09.031

1.4.4 Chapter 5

The fifth chapter contributes from a theoretical perspective by proposing a range of new and adapted relational learning algorithms for customer churn prediction using social network effects, designed to handle large scale networks, a time dependent class label, and a skewed class distribution. Furthermore, an innovative approach to incorporate non-Markovian network effects within relational classifiers is presented, and a novel parallel modeling setup is introduced to combine a relational and non-relational classification model.

From an application perspective, a new profit driven evaluation methodology is applied to assess the results of two real life case studies on large scale telco data sets, containing both networked (call detail record data) and non-networked (customer related) information about millions of subscribers.

Chapter 5 has been submitted for publication in:

Verbeke, W., Martens, D., Baesens, B., 2011d. Social network analysis for customer churn prediction. Management Science, under review

Chapter 2

Building comprehensible customer churn prediction models with advanced rule induction techniques

A good decision is based on knowledge and not on numbers.

Plato (424 BC - 348 BC)

Abstract

Customer churn prediction models aim to detect customers with a high propensity to attrite. Both the predictive power, the comprehensibility, and the justifiability are key aspects of these models. An accurate model permits to correctly target future churners in customer retention campaigns, while a comprehensible and intuitive rule set allows to identify the main drivers for customers to churn and to develop an effective retention strategy in accordance with domain knowledge. This chapter provides an extended overview of the literature on the use of data mining for customer churn pre-

diction modeling[1,2]. It is shown that only limited attention has been paid to the comprehensibility and the intuitiveness of churn prediction models. Therefore, in this chapter two novel data mining techniques are applied to predict churn and benchmarked to traditional rule induction techniques such as C4.5 and Ripper. Both AntMiner+ and ALBA are shown to induce accurate as well as comprehensible classification rule sets. AntMiner+ is a high performing data mining technique based on the principles of Ant Colony Optimization that allows to incorporate domain knowledge by imposing monotonicity constraints on the final rule set. ALBA on the other hand combines the strong predictive power of a non-linear support vector machine model with the comprehensibility of the rule set format. The results of the benchmarking experiments show that ALBA improves learning of classification techniques, yielding comprehensible models with improved classification performance. AntMiner+ results in accurate, comprehensible, but most importantly justifiable models, unlike the other modeling techniques included in this chapter.

2.1 Introduction

In recent decades we have witnessed an explosion of data. Valuable knowledge is contained within this information, but it is hidden in the vast collection of raw data. Data mining entails the overall process of extracting knowledge from this data. Data mining techniques have been successfully applied in many different domains. Well-known examples are cancer detection in the biomedical sector (Li et al., 2004; Delen et al., 2005), market basket analysis in the retail sector (Berry and Linoff, 2004), and credit scoring in the financial sector (Baesens et al., 2003b; Ahn and Kim, 2009). This chapter however focuses on the use of data mining to predict customer churn.

CCP models aim to detect customers with a high propensity to attrite. An accurate segmentation of the customer base into future churners and

[1]Verbeke, W., Baesens, B., Martens, D., De Backer, M., Haesen, R., 2009a. Including domain knowledge in customer churn prediction using antminer+. In: Perner, P. (Ed.), Workshop Proceedings DMM 2009. Advances in Data Mining in Marketing. IbaI Publishing, pp. 10–21

[2]Verbeke, W., Martens, D., Mues, C., Baesens, B., 2011e. Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Systems with Applications 38 (3), 2354–2364

non-churners allows a company to target the customers that are the most likely to attrite in a retention marketing campaign in order to prevent them from effectively churning. Given the limited marketing resources this allows to improve the efficiency of these campaigns. In summary, customer retention is profitable to a company because (1) attracting new clients costs five to six times more than customer retention (Bhattacharya, 1998; Rasmusson, 1999; Colgate et al., 1996; Athanassopoulos, 2000); (2) long-term customers generate higher profits, tend to be less sensitive to competitive marketing activities, become less costly to serve, and may provide new referrals through positive word-of-mouth, while dissatisfied customers might spread negative word-of mouth (Mizerski, 1982; Stum and Thiry, 1991; Reichheld, 1996; Zeithaml et al., 1996; Colgate et al., 1996; Paulin et al., 1998; Ganesh et al., 2000); (3) losing customers leads to opportunity costs because of reduced sales (Rust and Zahorik, 1993). A small improvement in customer retention hence can lead to a significant increase in profit (Gupta et al., 2004; Lariviere and Van den Poel, 2005). Therefore both accurate and comprehensible churn prediction models are needed, in order to identify respectively the customers that are about to churn, and if possible, the main drivers to churn. As will be discussed in Section 2.2, many data mining techniques have already been tested on their ability to predict churn. Much less attention has been paid however to the comprehensibility and the justifiability of the developed models. Note that churn prediction is just one of the possible applications of data mining in a marketing context. Other examples include frequent item set mining (Agrawal and Srikant, 1994), sales forecasting (Thomassey and Happiette, 2007), and customer lifetime value prediction (Gupta et al., 2006; Glady et al., 2009).

In this chapter we introduce the application of two novel data mining techniques for customer churn prediction. The first technique, AntMiner+, uses Ant Colony Optimization (ACO) to infer rules from data, and explicitly seeks to induce accurate, comprehensible, and intuitive classification rule sets (Martens et al., 2007b). So far AntMiner+ has been successfully applied to credit scoring (Martens et al., 2006), software mining (Vandecruys et al., 2008), audit mining (Martens et al., 2008), and business/ICT alignment prediction (Cumps et al., 2009). An advantage of AntMiner+ is the possibility to incorporate domain knowledge (Martens et al., 2006), ensuring intuitive decision support models.

The second technique is an Active Learning Based Approach (ALBA)

for support vector machine rule extraction (Martens et al., 2009). ALBA manipulates a data set by changing the class labels of data instances by the SVM predicted labels, and by generating additional data instances close to the class boundaries. Applying simple rule induction techniques such as C4.5 or Ripper on the manipulated data set results in improved learning, and thus in a more accurate, but still comprehensible, rule set.

The remainder of this chapter is structured as follows. First, in Section 2.2, the domain of CCP modeling is introduced by means of a broad literature study. Then, in Section 2.3, the workings of AntMiner+ and ALBA are briefly explained. In Section 2.4 both techniques are applied to predict customer churn, and the setup and results of a series of experiments are discussed. The final section concludes the chapter.

2.2 Customer churn prediction modeling

Customer relationship management, and customer churn prediction in particular, have received a growing attention during the last decade. Table 2.1 provides an overview of the literature on the use of data mining techniques for CCP modeling. The table summarizes the applied modeling techniques, the characteristics of the assessed data sets, and the experimental setup. The characteristics of the data set comprise the sector, the number of customers and features, and whether the data set is public (1) or private (2). The experimental setup summarizes the applied evaluation metrics, the validation method, and whether or not sampling and feature selection are applied.

In this chapter we argue that both accurate and comprehensible churn prediction models are needed, in order to identify respectively the customers that are about to churn, and their reasons to do so. As can be seen from Table 2.1, a myriad of modeling techniques has been tested in a search for the most accurate modeling technique: logistic regression, decision trees, neural networks, support vector machines, random forests, regression forests, and many others. The comprehensibility of churn prediction models on the other hand has received much less attention in the literature (Lima et al., 2009), although several studies have focused on the analysis of churn drivers (Buckinx and Van den Poel, 2005; Kumar and Ravi, 2008), illustrating the need to gain insight in the causes of churn and confirming the need for comprehensible models.

Furthermore, also the justifiability of a model should be considered in the

Authors	**Title &** *Journal*	**Year**	**What?**	**Techniques**	**Data set**	**Exp. setup**
Eiben et al. (1998)	Genetic modeling of customer retention - *Lecture Notes in Computer Science*	1998	Comparison of modeling techniques, application on real life data set	Logistic regression, genetic programming, rough data analysis, CHAID	Financial services - 14.394 cust. - 213 feat. - (2)	PCC, Lift chart, CoC - no sampling - OMEGA software - hold-out
Madden et al. (1999)	Subscriber churn in the Australian ISP market - *Information Economics and Policy*	1999	Development of a churn prediction model and analysis of churn drivers	Binomial probit model	Internet service provider - 592 cust. - 19 feat. - (2)	PCC, goodness-of-fit - no sampling - no feat. selection - likelihood ratio test
Datta et al. (2000)	Automated cellular modeling and prediction on a large scale - *Artificial Intelligence Review*	2000	Description and application of automated cellular churn prediction modeling system	Neural network	Wireless telco - 500.000 cust. - 200 feat. - (2)	Lift, payoff - undersampling - forward selection with decision tree, genetic alg. - hold-out
Mozer et al. (2000)	Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry - *IEEE Transactions on Neural Networks*	2000	Prediction of churn probability, retention incentive optimization, and profit gains estimation	Logistic regression, (boosted) decision tree, (boosted) neural network	Wireless telco - 46.744 cust. - 134 feat. - (2)	Lift chart - no sampling - no feat. selection - 10 fold cross validation
Wei and Chiu (2002)	Turning telecommunications call details to churn prediction: a data mining approach - *Expert Systems with Applications*	2002	Application of the C4.5 algorithm to create a churn prediction model, using a limited number of features	Decision tree (C4.5)	Wireless telco - 114.000 cust. - over 12 feat. - (2)	Lift, lift chart, miss rate, false rate, detection error trade-off curve - oversampling - intuition - 3x10 fold cross validation

Authors	**Title &** *Journal*	**Year**	**What?**	**Techniques**	**Data set**	**Exp. setup**
Au et al. (2003)	A novel evolutionary data mining algorithm with applications to churn prediction - *IEEE Transactions on Evolutionary Computation*	2003	Application of a novel data mining technique to predict churn probabilities	Decision tree (C4.5), neural network, data mining by evolutionary learning	Wireless telco - 100.000 cust. - 251 feat. - (2)	Top 5% lift, lift chart - undersampling - intuition - 10 fold cross validation
Hwang et al. (2004)	An LTV model and customer segmentation based on customer value: a case-study on the wireless telecommunications industry - *Expert Systems with Applications*	2004	Churn prediction model as part of a customer lifetime value model	Logistic regression, decision tree, neural network	Wireless telco - 16.384 cust. - 200 feat. - (2)	Error rate, lift chart - no sampling - R^2 method - hold-out
Buckinx and Van den Poel (2005)	Customer base analysis: partial defection of behaviorally-loyal clients in a non-contractual FMCG retail setting - *European Journal of Operational Research*	2005	Comparison of techniques for partial defection prediction, focus on profitable customers in a non-contractual setting	Logistic regression, neural network, random forests	Grocery retail - 158.884 cust. - 61 feat. - (2)	PCC, AUC - no sampling - no feat. selection - hold-out
Lariviere and Van den Poel (2005)	Predicting customer retention and profitability by using random forests and regression forests techniques - *Expert Systems with Applications*	2005	Investigation of explanatory variables and modeling methods for customer churn prediction	Logistic regression, linear regression, random forests, regression forests	Financial services - 100.000 cust. - 30 feat. - (2)	AUC - no sampling - no feat. selection - hold-out, non-parametric test of De Long et al.

Authors	**Title &** *Journal*	**Year**	**What?**	**Techniques**	**Data set**	**Exp. setup**
Hung et al. (2006)	Applying data mining to telecom churn management - *Expert Systems with Applications*	2006	Comparative study and application of churn prediction modeling methods	Decision tree, neural network (on clustered segments)	Wireless telco - 160.000 cust. - over 40 feat. - (2)	Hit ratio, top-decile lift - oversampling - intuition, EDA - hold-out, t-test
Lemmens and Croux (2006)	Bagging and boosting classification trees to predict churn - *Journal of Marketing Research*	2006	Application of bagging and boosting techniques to improve predictive power of churn prediction models	Logistic regression, bagged & boosted decision trees	Wireless telco - 100.000 cust. - 171 feat. - (1)	Error rate, top-decile lift, Gini - proportional and oversampling - principal components analysis - hold-out
Neslin et al. (2006)	Defection detection: measuring and understanding the predictive accuracy of customer churn models - *Journal of Marketing Research*	2006	Analysis of the results of a churn prediction modeling tournament with focus on method and shelve life	Logistic regression, decision tree, neural network, discriminant analysis, Bayes	Wireless telco - 100.000 cust. - 171 feat. - (1)	Top-decile lift, Gini coefficient -
Burez and Van den Poel (2007)	CRM at a pay-TV company: using analytical models to reduce customer attrition by targeted marketing for subscription services - *Expert Systems with Applications*	2007	Development of churn prediction model, tested in real-life retention campaign	Logistic regression (with Markov chains), random forests	Pay-TV - 143.198 cust. - 81 feat. - (2)	PCC, cumulative lift, AUC - no sampling - no feat. selection - hold-out
Coussement and Van den Poel (2008)	Churn prediction in subscription services: an application of SVMs while comparing two parameter-selec. techniques - *Expert Systems with Applications*	2008	Application of support vector machine in churn prediction in a newspaper subscription environment	Logistic regression, support vector machine, random forests	Newspaper subscription - 90.000 cust. - 82 feat. - (2)	PCC, AUC, top-decile lift - undersampling - no feat. selection - 10 fold cross validation, De Long

Authors	**Title &** *Journal*	**Year**	**What?**	**Techniques**	**Data set**	**Exp. setup**
Kumar and Ravi (2008)	Predicting credit card customer churn in banks using data mining - *International Journal of Data Analysis Techniques and Strategies*	2008	Extensive study to compare the results of different sampling and modeling techniques to predict credit card customer churn	Logistic regression, decision tree, neural network, svm, random forest, rbf network, ensembles	Credit card - 14.814 cust. - 22 feat. - (1)	PCC, specificity, sensitivity, AUC - under- & oversampling (combined) & SMOTE - CART - hold-out, 10 fold cross validation
Burez and Van den Poel (2009)	Handling class imbalance in customer churn prediction - *Expert Systems with Applications*	2009	Study on sampling methods, evaluation metrics and methods, and modeling techniques	Logistic regression, gradient boosting, (weighted) random forests	Banks, telco, newspaper, pay-TV, supermarket - 32.371 to 143.198 cust. - 21 to 81 feat. - (2)	Error rate, AUC, lift - undersampling, CUBE - 5x2 fold cross validation, t-test
Lima et al. (2009)	Domain knowledge integration in data mining using decision tables: case studies in churn prediction - *Journal of the Operations Research Society*	2008	Incorporation of domain knowledge in churn prediction models	Logistic regression, decision tree	Wireless telco (2) - 5.000 and 100.000 cust. - 21 and 171 feat. - (1)	PCC, specificity, sensitivity, AUC - oversampling - Cramer's V-statistic, t-tests - hold-out, De Long

Table 2.1: Overview of literature on customer churn prediction modeling. The information on the data set comprises the sector, the number of customers and features, and whether the data set is public (1) or private (2). The experimental setup information summarizes the applied evaluation metrics, whether sampling and feature selection are applied, and the validation method.

evaluation of a churn prediction model. In a data mining context, a model is justifiable when it is in line with existing domain knowledge. For a model to be justifiable, it needs to be validated by a domain expert. This in turn means that the model should be comprehensible (Martens et al., 2006). A modeling technique that allows to take into account domain knowledge and yields a model that behaves intuitively correct is of much greater use than a technique that produces counter-intuitive results Martens et al. (2011). The most frequently encountered and researched aspect of knowledge fusion, i.e., incorporating the knowledge representing the experience of an expert into a data mining approach, is the monotonicity constraint. A positive (negative) monotonicity constraint demands that an increase (decrease) in a certain input cannot lead to a decrease in the output. For instance, in a customer churn prediction setting, an increase in the amount a customer is charged can be expected to yield an increased probability to churn. Including domain knowledge in churn prediction modeling is to our knowledge thus far only discussed in Lima et al. (2009), and is one of the main contributions of this chapter.

Customer churn prediction has been extensively researched, yet no general consensus exists on the performance of predictive modeling techniques in a CCP setting. For instance, both Mozer et al. (2000) and Hwang et al. (2004) apply logistic regression and neural networks to predict churn. However, the first study finds neural networks to perform best and the second logistic regression. Therefore, in Chapter 4 a broad benchmarking experiment is set up which allows to compare the performance of a variety of classification techniques. Although most studies summarized in Table 2.1 use private data sets, evaluating data mining techniques on publicly available data sets has many advantages (Vandecruys et al., 2008): (1) The creation of benchmarks is facilitated which makes it possible to compare and rank existing and new data mining techniques. (2) The impact of the characteristics of a data set on the performance of a data mining technique is the same for all techniques. Comparing the results and rankings of techniques applied on a variety of data sets on the other hand allows to evaluate and study the effect of data characteristics on the performance of a technique. (3) Using publicly available data sets provides insight in the impact of each step of the followed methodology. Data preprocessing steps like input variable selection and sampling have a significant impact on the final result, possibly even to a larger extent than the choice of modeling technique. To summarize, using

publicly available data sets improves the general comparability of results, techniques, and methodologies.

A final point of critique concerns the use of a single split up of the data set in a training and a test set to validate the results of a model. It should be clear that the average result on multiple split ups provides a more reliable measure of performance than a single shot result. Furthermore, to draw valid conclusions about differences in performance of techniques, results should be tested whether they differ significantly or not. A common heuristic to test the significance of performance differences is for instance the Student's paired t test (Dietterich, 1998), which will be applied in this chapter.

2.3 Advanced rule induction techniques

As CCP models should be both accurate and comprehensible, we will focus on the use of rule-based classification techniques. More specifically, we will induce rule sets from a churn data set using AntMiner+ and ALBA. This section explains the workings of AntMiner+ and ALBA.

2.3.1 AntMiner+: classification based on Ant Colony Optimization

AntMiner+ is a classification technique that employs artificial ants to induce rules[3]. Previous benchmarking studies reveal that the models generated by AntMiner+ meet both the accuracy and comprehensibility requirements (Martens et al., 2007b), and are also intuitively correct (Martens et al., 2006). In this section the main workings of this technique are explained, starting with a short introduction to the basic concept behind the workings of AntMiner+: Ant Colony Optimization.

Ant Colony Optimization

Ant Colony Optimization is a metaheuristic inspired on the foraging behavior of real ant colonies (Dorigo and Stützle, 2004). A biological ant by itself is a simple insect with limited capabilities, and is guided by straightforward decision rules. However, these simple rules are sufficient for the overall ant colony to find short paths from the nest to the food source. ACO employs

[3] www.antminerplus.com

artificial ants that cooperate in a similar manner as their biological counterparts, in order to find good solutions for discrete optimization problems. The first ACO algorithm developed was Ant System (Dorigo et al., 1996), where ants iteratively construct solutions and add pheromone to the paths corresponding to these solutions. Path selection is a stochastic procedure based on a history-dependent pheromone value and a problem-dependent heuristic value. The pheromone value indicates the number of ants that recently chose the trail, while the heuristic value is a problem dependent quality measure. When ants reach a decision point, they are more likely to choose a trail with a higher pheromone level and heuristic value. ACO has been applied to a wide variety of problems, such as the vehicle routing problem (Bullnheimer et al., 1999; Wade and Salhi, 2004; Porta Garcia et al., 2009), scheduling (Colorni et al., 1994; Blum, 2005), and routing in packet-switched networks (Di Caro and Dorigo, 1998). Recently, ACO has also been applied in the field of data mining, addressing both the clustering (see, e.g., the work by Abraham and Ramos (2003); Handl et al. (2006); Boryczka (2009)) and classification task (Parpinelli et al., 2001; Liu et al., 2003; Martens et al., 2007b), which is the topic of interest in this chapter.

AntMiner+ Algorithm

ACO can be applied to induce comprehensible and accurate rule-based classification models from data, as done by the AntMiner+ classification technique. This technique implements the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System (Stützle and Hoos, 2000) for classification. An environment is defined for the ants to walk through, in a way that each path corresponds to a classification rule. As such, the path chosen by each ant corresponds to a predictive rule. The principles of ACO guide the ants towards good predictive rules, as shown in the benchmarking study in Martens et al. (2007b). The outline of the workings of the AntMiner+ algorithm is provided as Algorithm 1. For more details on the workings of this technique, one may refer to Martens et al. (2007b).

Advantages of AntMiner+ are not only the strong predictive power and the comprehensibility of the generated models, but also the possibility to demand intuitive predictive models, which is crucial whenever comprehensibility is required. For example, when a classification rule is induced to predict whether or not a customer will churn, the rule "**if** $Charge > 50$ **then** class = no churner", is counterintuitive, as we would expect that the more

Algorithm 1 Pseudo-code of AntMiner+ algorithm

```
construct graph
while not early stopping or minimum percentage data covered do
    initialize heuristics, pheromones and probabilities of edges
    while not converged do
        create ants
        let ants run from source to sink
        evaporate pheromone on edges
        prune rule of best ant
        update path of best ant
        adjust pheromone levels if outside boundaries
        kill ants
        update probabilities of edges
    end while
    extract rule corresponding to converged path
    flag data points covered by the extracted rule
end while
evaluate performance on test set
```

a customer is charged, the more likely he is to churn, making the expected sign for this example "<". On the other hand, the rule "**if** $Charge > 50$ **then** class = churner" is intuitively correct. By imposing constraints on the inequality signs in the rules a domain expert is allowed to incorporate his knowledge and experience, resulting in a justifiable classification model.

2.3.2 ALBA: Active Learning Based Approach for SVM rule extraction

The support vector machine (Vapnik, 1995) is currently one of the state-of-the-art classification techniques. Benchmarking studies reveal that in general, the SVM performs best among current classification techniques (Baesens et al., 2003b), due to its ability to capture non-linearities. However, its strength is also its main weakness, as the generated non-linear models are typically regarded as incomprehensible black-box models. The opaqueness of SVM models can be remedied through the use of rule extraction techniques, which induce rules that mimic the black-box SVM model as closely as possible. By extracting rules some insight is provided into the logics of

Algorithm 2 Pseudo-code of ALBA algorithm

preprocess data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$
split data in training data $\mathcal{D}_{tr}$, and test data $\mathcal{D}_{te}$ in a 2/3, 1/3 ratio
tune SVM parameters with gridsearch on $\mathcal{D}_{tr}$
train SVM on $\mathcal{D}_{tr} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N_{tr}}$, providing an oracle SVM mapping a data input to a class label
change the class labels of the training data to the SVM predicted class
% *Calculate the average distance* $distance_k$ *of training data to support vectors, in each dimension* k
for k=1 to n **do**
  $distance_k = 0$
  for all support vectors $\mathbf{sv_j}$ **do**
    for all training data instance $\mathbf{d}$ in $\mathcal{D}_{tr}$ **do**
      $distance_k = distance_k + |d_k - sv_{j,k}|$
    end for
  end for
  $distance_k = \frac{distance_k}{\#\mathbf{sv} \times N_{tr}}$
end for
% *Create 1000 extra data instances*
for i=1 to 1000 **do**
  randomly choose one of the support vectors $\mathbf{sv_j}$
  % *Randomly generate an extra data instance* $\mathbf{x_i}$ *close to* $\mathbf{sv_j}$
  for k=1 to n **do**
    $x_{i,k} = sv(j,k) + \left[(rand - 0.5) \times \frac{distance_k}{2}\right]$ with $rand$ a random number in $[0, 1]$
  end for
  provide a class label y_i using the trained SVM as oracle: $y_i = SVM(\mathbf{x_i})$
end for
run rule induction algorithm on the data set containing both the training data $\mathcal{D}_{tr}$, and newly created data instances $\{(\mathbf{x_i}, y_i)\}_{i=1:1000}$
evaluate performance in terms of accuracy, fidelity and number of rules, on $\mathcal{D}_{te}$

the SVM model (Martens et al., 2007a).

ALBA is a rule extraction algorithm that uses specific concepts of the

SVM, being the support vectors, in combination with traditional rule induction techniques such as C4.5 and Ripper (Martens et al., 2009). Active learning entails the control of the learning algorithm over the input data on which it learns. More specifically, active learning focuses on the problem areas (Cohn et al., 1994), which for rule extraction are those areas in the input space where the noise is the strongest. These regions are found near the SVM decision boundary, which marks the transition of one class to another. First, we can change the labels of the data instances by the SVM predicted labels. In this manner the induced rules will mimic the SVM model and all noise is omitted from the data, removing any apparent conflicts in the data. Second, to incorporate the active learning approach additional data instances are generated close to the decision boundary. For this explicit use is made of the support vectors, which are typically close to the decision boundary. The support vectors are thus used as proxies for the decision boundary by generating additional data instances close to the support vectors. Since the distribution of the support vectors will follow the data distribution, more support vectors will be found in dense input areas, and less in more sparse ones. This implicit incorporation of the existing data distribution in the extra data generation step, eliminates the necessity to explicitly take into account density measures. The Active Learning Based Approach is described formally in Algorithm 2. A full discussion on ALBA can be found in Martens et al. (2009).

After improving and expanding the input data, simple rule induction techniques such as C4.5 and Ripper are applied to induce rule sets. C4.5 is a popular decision tree builder (Quinlan, 1993) where each leaf assigns a class label to observations. Each of these leaves can be represented by a rule and therefore C4.5 builds comprehensible classifiers. Ripper is a rule induction technique, generating a list of ordered rules (Cohen, 1995; Witten and Frank, 2000; Tan et al., 2006). The name Ripper is an acronym for Repeated Incremental Pruning to Produce Error Reduction. A detailed overview of this technique can be found in Cohen (1995) and Witten and Frank (2000).

2.4 Customer churn prediction with AntMiner+ and ALBA

2.4.1 Data set

AntMiner+ and ALBA are applied on a publicly available data set downloaded from the KDD library[4]. The data set is obtained from a wireless telco operator, and consists of 5000 observations. For each observation 21 features are available, with no missing values. 14.3% of the customers are indicated to churn in the coming three months. For a full description of the data set, one may refer to Larose (2005).

2.4.2 Data preprocessing

Data preprocessing was conducted in the form of discretization, input selection, and oversampling.

Discretization

All continuous variables are discretized following Fayyad and Irani (1993). Discretization and other data preprocessing procedures are performed using the open source data mining workbench Weka[5] (Witten and Frank, 2000).

Input selection and monotonicity constraints

To make sure that only relevant variables are included in the data set, and to decrease the computational burden, an input selection procedure is performed using a chi-squared based filter (Thomas et al., 2002; Martens et al., 2006). First, the observed frequencies of all possible combinations of values for class and variable are measured. Based on this, the theoretical frequencies, assuming complete independence between the variable and the class, are calculated. The hypothesis of equal odds provides a chi-squared test statistic; higher values allow one to more confidently reject the zero hypothesis of equal odds; hence, these values allow one to rank the variables according to predictive power. In this manner, the set of features was reduced from twenty to eleven as listed in Table 4.3. Also included in Table

[4] www.datalab.uci.edu/data/mldb-sgi/data/

[5] www.cs.waikato.ac.nz/ml/weka

Feature	Constraint	What?
Day_Mins		Daytime usage (minutes/month)
Day_Charge	+	Charge for daytime usage ($/month)
CustServ_Calls	-	Number of calls to customer service
Intl_Plan		International plan subscriber (0 = no)
Vmail_Plan		Voicemail plan subscriber (0 = no)
Vmail_Message		Number of voice mail messages
Intl_Charge	+	Charge for international calls ($/month)
Intl_Mins		International usage (minutes/month)
Eve_Mins		Evening usage (minutes/month)
Eve_Charge	+	Charge for evening usage ($/month)
Intl_Calls		Number of international calls

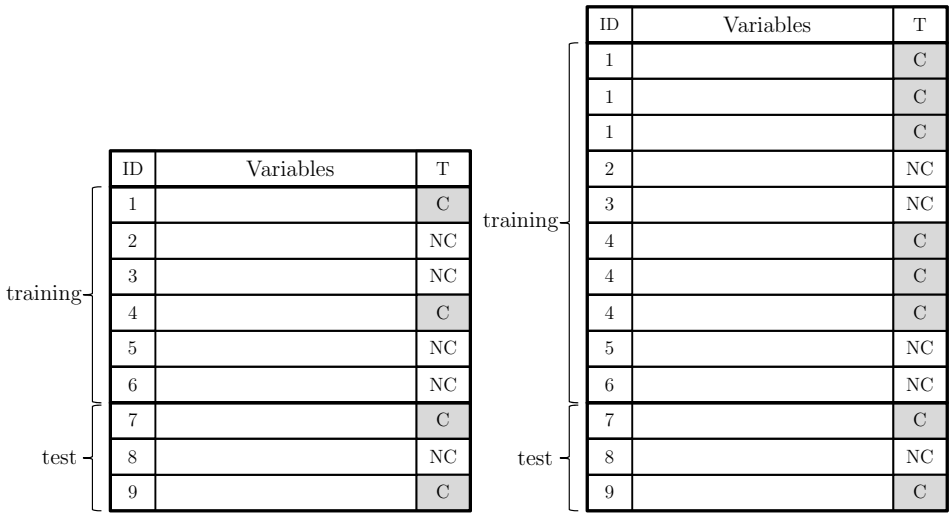
Table 2.2: Top eleven ranked features with chi-squared based filter and intuitive sign relations with churn

4.3 are the signs of the expected relations between the explanatory variables and the class variable, expressing domain knowledge. For instance, the positive relation sign between *churn* and *day_charge* indicates that according to domain knowledge a customer is more likely to churn when he is charged more. As can be seen from the table, only for a few features an explicit relation is presumed, in accordance with Lima et al. (2009). The resulting AntMiner+ rule set will be enforced to comply with domain knowledge by imposing monotonicity constraints.

Oversampling

The class variable of the data set is heavily skewed: the number of churners (14.3%) is much smaller than the number of non-churners (85.7%). This causes the classification modeling techniques to experience difficulties in learning which customers are about to churn. Since predicting future churners is a principle objective of the model, oversampling is used to improve learning (Provost et al., 1998). Figure 2.1 illustrates the principle of oversampling. Observations of the minority class in the training set are copied and added to the training set. Only the training data is oversampled and the test set is not, in order to provide an unbiased indication of the performance of the model towards future predictions.

Depending on the number of times churning class observations are re-



(a) Original data set (b) Oversampled data set

Figure 2.1: Illustration of the principle of oversampling. A small data set with target variable T and nine observations (left panel) is split into a training set of six observations and a test set of three observations. Training instances classified as churners (T = C) are repeated twice in the oversampled data set (right panel).

peated in the training set, the resulting accuracy (percentage correctly classified instances (PCC) as churners or non-churners), sensitivity (SENS, percentage of churners that is correctly predicted) and specificity (SPEC, percentage of non-churners that is correctly classified) varies. Table 2.3 illustrates this problem, showing exploratory single shot results for AntMiner+ on the studied data set for different degrees of oversampling. The results for the other reviewed classification techniques are similar. For the original data set (0 oversampling) we observe that AntMiner+ reaches a reasonable accuracy and a high specificity, but a fairly low sensitivity. This reflects the difficulties encountered in learning. A higher degree of oversampling results in a higher sensitivity, but implies a declining specificity. Because the share of churners in the data set increases with oversampling, the importance of sensitivity relative to specificity increases in the calculation of accuracy. This means that, as the degree of oversampling increases, the accuracy decreases, unless learning and thus classification improves. However, since the

# Oversampling	PCC	#R	SENS	SPEC
0	87.66	1	16.34	99.14
1	90.33	12	75.34	92.95
2	90.33	12	75.78	92.87
3	**90.33**	**12**	**76.68**	**92.71**
4	89.73	12	72.20	92.80
5	90.07	17	84.98	90.91

Table 2.3: Out-of-sample performance gain for AntMiner+ using oversampling

cost of not detecting a churner is likely to be higher than the cost of targeting a non-churner in a retention campaign, this is a trade-off that one is willing to make. A reasonable trade-off between sensitivity on the one hand and specificity and accuracy on the other hand, is reached at three times oversampling. At this oversampling rate, the distribution of churners versus non-churners is almost even, as there are about as much churners (57.2%) as non-churners in the data set (42.8%). This is reflected in the results, which show good performances both in terms of specificity and sensitivity. At five times oversampling an even higher sensitivity is reached. This happens however at the cost of a decrease in the comprehensibility, since the number of rules (#R) increases from 12 to 17. Finally, also note that the bigger the data set, the higher the computational requirements. This can become an issue for techniques such as SVM, which scales non-linearly with the number of observations. Therefore we decided to repeat each observation corresponding to a churner three times in the final training data sets.

2.4.3 Experimental setup

To evaluate the results of AntMiner+ (with and without including domain knowledge) and ALBA (with rule induction using both C4.5 and Ripper), a benchmarking study is performed that includes commonly used state-of-the-art classification techniques such as C4.5, Ripper, SVM, logistic regression, and simplistic majority vote. Because of the importance of comprehensibility in a churn prediction setting, both the results of C4.5 with standard pruning (*confidence factor* = 0.25) and with extensive pruning (*confidence factor* = 0.001) are included. Extensive pruning (indicated with XP in the

results table) leads to fewer and smaller rules, and thus to a more comprehensible rule set. Also the results of ALBA combined with Ripper are reported with standard and extensive pruning (minimum total weight of instances in a rule of respectively 2.0 and 12.0). While C4.5 induces an unordered rule set, AntMiner+ and Ripper induce ordered rule sets. C4.5, Ripper, logistic regression, and majority vote are all evaluated using the open-source Weka workbench (Witten and Frank, 2000).

The parameters of AntMiner+ that need to be set are the total number of ants and the evaporation factor ρ, which are respectively initialized to 1000 and 0.85, as suggested by Martens et al. (2006). The number of extra data instances generated by ALBA is set to 2000 in order to obtain a good balance between predictive performance and computational burden (Martens et al., 2009). For the SVM a radial basis function (RBF) Kernel is selected as it is shown to achieve a good overall performance (Baesens et al., 2003b; Martens et al., 2009). The regularization parameter C, and bandwidth parameter σ are set using a grid search mechanism (Suykens et al., 2002).

The reported measures are the average out-of-sample performances on ten random 70/30 split ups of the data set in training and test sets. Early stopping is applied since the data set is relatively large. Therefore one third of the training data sets is set apart for validation. Four series of experiments were performed. In the first series the performance of each classification technique is tested on the original data set. Then, in the second series ALBA is applied as explained in subsection 2.3.2, using the support vector machines trained on the original data. In the third series the original data are oversampled as explained in subsection 2.4.2. Finally, in the fourth series the ALBA data set of the second series is oversampled.

2.4.4 Results and discussion

The results of the churn prediction experiments are summarized in Table 3.6, with the best performances in terms of average percentage correctly classified, specificity, and sensitivity underlined. Also included in the table are the number of induced rules and the standard deviation (STDV) of the accuracy. As discussed in section 2.2, a one-sided Student's paired t test is applied to test the performance differences. Performances that are not significantly different at the 5% level from the top performance with respect to a one-tailed paired t-test are tabulated in bold face. Statistically

significant underperformances at the 1% level are emphasized in italics, and performances significantly different at the 5% level but not a the 1% level are reported in normal script. Since the observations of the randomizations are not independent, we remark that this standard t-test is used as a common heuristic to test the performance differences (Dietterich, 1998).

Series	Technique	PCC	SENS	SPEC	#R	STDV
Original	**AntMiner+**	*90,85*	*37,09*	**99,71**	7,7	0,54
	AntMiner+ DK	*90,73*	*36,26*	**99,69**	8,0	0,44
	C4.5	**93,59**	*64,93*	**98,34**	21,1	0,49
	C4.5 XP	*92,94*	*56,88*	**98,90**	10,7	0,52
	Ripper	*92,92*	*62,31*	**97,99**	5,8	0,83
	Ripper XP	*92,83*	*60,71*	**98,15**	5,5	0,78
	SVM	*92,51*	*78,29*	*94,85*	-	0,52
	Logit	*86,87*	*29,18*	*96,44*	-	0,55
	Majority rule	*85,79*	*0,00*	**100,00**	-	0,56
ALBA	**AntMiner+**	*92,83*	*53,95*	**99,27**	13,9	1,24
	AntMiner+ DK	*91,29*	*41,32*	**99,57**	15,0	0,44
	C4.5	**93,79**	*65,53*	**98,49**	73,3	0,43
	C4.5 XP	**93,70**	*65,24*	**98,44**	29,7	0,47
	Ripper	**93,85**	*65,95*	**98,46**	20,5	0,39
	Ripper XP	**93,87**	*65,66*	**98,54**	10,5	0,38
Oversampled	**AntMiner+**	*93,15*	*65,76*	**97,72**	14,8	0,51
	AntMiner+ DK	*92,62*	67,98	**96,72**	16,3	0,81
	C4.5	*91,66*	**80,82**	**93,45**	23,6	0,21
	C4.5 XP	*90,94*	**82,29**	**92,31**	13,6	0,61
	Ripper	*91,73*	**81,49**	**93,41**	7,7	0,57
	Ripper XP	*91,89*	**81,28**	**93,64**	7,5	0,36
	Logit	*88,31*	*73,42*	**89,62**	-	3,41
ALBA	**AntMiner+**	*92,45*	*74,42*	**95,44**	13,8	0,70
Oversampled	**AntMiner+ DK**	*91,51*	*63,12*	**96,20**	13,8	1,57
	C4.5	*92,41*	77,42	**94,89**	84,5	0,33
	C4.5 XP	*92,32*	77,90	**94,71**	40,8	0,33
	Ripper	*92,39*	77,37	**94,88**	25,9	0,32
	Ripper XP	*92,41*	77,46	**94,88**	14,9	0,30

Table 2.4: Average out-of-sample results of the churn prediction experiments

Predictive power

As can be seen from the results table, the highest accuracy is reached using the combination of ALBA and Ripper with extra pruning. C4.5 applied on the original data set, and ALBA combined with C4.5 with standard and increased pruning and Ripper with standard pruning do not perform significantly worse. Other techniques follow closely however, and except for logistic regression and majority vote all results lie in the interval between 90% and 94%.

Accuracy alone is not an adequate performance measure to evaluate the experimental results though, as it implicitly assumes a relatively balanced class distribution among the observations and equal misclassification costs (Baesens et al., 2003b). The skewed distribution of the data set, which is typical for churn prediction, was already mentioned. But also the assumption of equal misclassification costs cannot be sustained. Typically, a customer relationship manager who applies data mining techniques for customer churn prediction will mainly be interested in the correct detection of future churners. Even to the extent that it is preferred to include a certain number of customers that will not churn in the nearby future in a retention campaign. Indeed, the costs of including a number of non-churning customers do not weigh up to the costs a company incurs due to churn, at least to a certain extent.

As the costs associated with the incorrect classification of churners are clearly higher than the costs associated with the incorrect classification of a non-churner, it seems fair to us to assume unequal misclassification costs. Consequently, a high sensitivity is of more importance to a company than a high specificity. Of course this does not mean that specificity can be neglected. A classification technique that classifies all customers as churners might well result in including all churners in a retention campaign, but the retention marketing costs will be unjustifiably high. A trade-off has to be made in order to obtain a high sensitivity combined with a reasonable specificity. This allows the company to efficiently allocate its retention marketing budget, by focusing on the customers that are classified to have the highest propensity to attrite.

The highest sensitivity in the churn prediction experiments is reached with C4.5 XP on the oversampled data set. C4.5, Ripper, and Ripper XP do not perform significantly worse at the 5 percent level, while the result of AntMiner+ DK does not differ significantly at the 1% level. The high-

est specificity on the other hand is reached with AntMiner+ applied on the original data set. However, only the SVM and the logit model perform significantly worse. Therefore, and because we are mainly interested in detecting future churners, we will no further assess the specificity in the evaluation of the results.

Oversampling the data set appears to improve significantly the sensitivity of all data mining techniques. ALBA and ALBA Oversampled do as well, but only to a limited extent. This is a consequence of training the support vector machine on the original data set. As can be seen from the table, the sensitivity of the SVM is remarkably high compared to the results of the other techniques applied on the original data set, which illustrates the power of the non-linear SVM. However, this result imposes an upper bound to the results that can be achieved using ALBA. Even if C4.5, Ripper, or any other technique classify the ALBA training data set 100% correct, the resulting sensitivity of the model can not be higher than 78.29%. As can be seen from Table 3.6, the sensitivity of C4.5 (XP) and Ripper (XP) applied on the oversampled ALBA data set indeed lies around 78%. A possible solution which could lead to better performances exists in an adjustment of ALBA in order to take into account the class distribution of the data set. The ALBA algorithm should strive towards class balance when adding data points that lie near the non-linear class boundaries. Or, if misclassification costs are unequal, even a distribution in favor of the minority class could be created. ALBA does not lead to the overall best results in this experimental setup, since oversampling leads to even higher sensitivities than the support vector machine trained on the original data set achieves. ALBA however does lead to the highest accuracies and remains an interesting technique to improve the performance of rule induction techniques.

Comprehensibility

High accuracy, sensitivity, and specificity are not the only important aspects in evaluating a churn prediction model. As stressed in the literature review, also the reasons for customers to churn are valuable information for a company. Such knowledge allows to develop a more effective retention strategy by focusing on the probable causes of churn. Therefore, comprehensibility of the classification model is an important requirement in churn prediction modeling. There is not really much to comprehend about a majority vote model. The only principle behind this technique is "majority

wins". Therefore, majority vote adds almost no value. Logistic regression performs reasonably well as to comprehensibility, but its model structure is arguably more opaque than a rule-based representation. C4.5, Ripper, and AntMiner+ on the other hand induce comprehensible rules from a data set. The comprehensibility of the resulting model decreases however as the number of rules increases. As can be seen from Table 3.6, AntMiner+ and Ripper clearly induce much less rules than C4.5, even with increased pruning. The issue faced by C4.5 is its greedy character, since every split made in the decision tree is irreversibly present in all leaves underneath. Hence AntMiner+ and Ripper, which on average result in a comparable number of rules, are the most comprehensible classification techniques tested in the experiments. This confirms previous results (Martens et al., 2006; Vandecruys et al., 2008). Finally, comprehensibility is also important to check the justifiability of a model.

Justifiability

The rule sets in Table 2.5 are induced by respectively AntMiner+ DK (with inclusion of domain knowledge) and Ripper. Figure 2.2 shows the decision tables corresponding to these rule sets derived using the PROLOGA software (Vanthienen et al., 1998b). A decision table is a more intuitive and user-friendly tabular representation of a rule set, and consists of four quadrants which are separated by horizontal and vertical double-lines. The upper left quadrant contains the condition subjects, which represent the attributes in the rules of the rule set. The action subjects in the lower left column describe the possible outcomes of the rules, which are in this case churn or non-churn. Every column in the upper-right quadrant of the decision table comprises a classification rule, leading for a certain combination of condition states to the classification outcome in the lower-right quadrant marked with 'x'. A dash symbol ('-') in an entry column indicates that the value of the attribute is irrelevant within the context of that column.

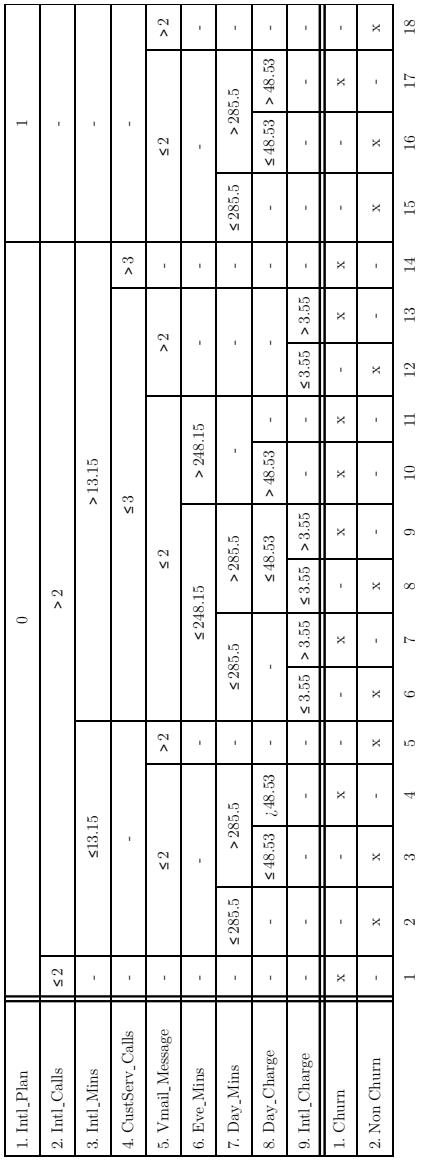
It can be seen from the upper panel of Table 2.5 and the corresponding decision table in Figure 2.2 that all rules induced with AntMiner+ DK comply with the monotonicity constraints in Table 4.3. This is of great value for the practical use of the model since domain knowledge and prediction model are aligned and give complementary results. The rule set in the lower panel of Table 2.5 induced by Ripper on the other hand contains rules that do not comply with the constraint on the variable *Intl_Charge*. According

AntMiner+ rule set
if $Intl\_Plan = 0$ **and** $Intl\_Calls \leq 2$
if $Day\_Mins > 285.5$ **and** $Day\_Charge > 48.53$ **and** $Vmail\_Message \leq 2$
if $Intl\_Plan = 0$ **and** $Intl\_Charge > 3.55$ **and** $Intl\_Mins > 13.15$
if $Intl\_Plan = 0$ **and** $Vmail\_Message \leq 2$ **and** $Intl\_Mins > 13.15$ **and** $Eve\_Mins > 248.15$
if $CustServ\_Calls > 3$ **and** $Intl\_Plan = 0$ **and** $Intl\_Mins > 13.15$
then class = churn
else class = non churn
Ripper rule set
if $Day\_Mins > 248.65$ **and** $VMail\_Plan = 0$
if $CustServ\_Calls > 3$ **and** $Day\_Mins \leq 168.05$
if $Intl\_Plan = 0$ **and** $Intl\_Calls \leq 2$
if $Day\_Mins > 221.85$ **and** $VMail\_Plan = 0$ **and** $Eve\_Mins > 248.15$
if $Intl\_Plan = 0$ **and** $Intl\_Charge > 3.55$
if $CustServ\_Calls > 3$ **and** $Day\_Mins \leq 221.85$ **and** $Eve\_Mins \leq 248.15$ **and** $Intl\_Charge \leq 3.55$
if $VMail\_Plan = 0$ **and** $Day\_Mins > 221.85$ **and** $CustServ\_Calls > 3$
then class = churn
else class = non churn

Table 2.5: AntMiner+ (upper panel) and Ripper (lower panel) rule sets for three times oversampling respectively with and without monotonicity constraints

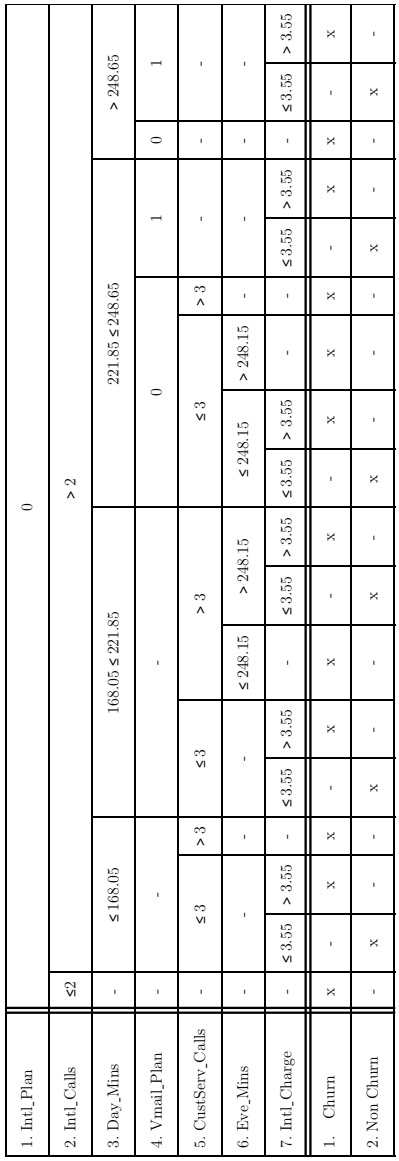
to domain knowledge the more a client is charged, the more probably he will churn. The first two rules in the Ripper rule set violate this principle. Therefore this rule set is not intuitive and will probably even be discarded by the user. Since *Intl_Charge* is included in the last row of the corresponding decision table in Figure 2.2, one can easily see that the opposite relation is modeled by columns 22 and 23 which partly represent the first two rules induced by Ripper. For instance, suppose two identical customers A and B with the same values for each feature, except for *Intl_Charge* which is equal to zero and ten for respectively customers A and B. The values of attributes *Intl_Plan*, *CustServ_Calls*, *Day_Mins*, and *Eve_Mins* are respectively equal to one, one, two hundred, and two hundred. Then the rule set or decision table will classify customer A as a churner and customer B as a non-churner, which is not positively monotone with respect to *Intl_Charge*.

It can be easily seen from the decision table that the AntMiner+ DK rule set on the other hand results in a positive monotone relation between the class variable and *Intl_Charge*, in accordance with domain knowledge.



		0															1								
1. Intl_Plan		0															1								
2. Intl_Calls	≤ 2	> 2															-								
3. Intl_Mins	-	≤13.15				> 13.15											-								
4. CustServ_Calls	-	-				≤ 3									> 3		-								
5. Vmail_Message	-	≤ 2		> 2		≤ 2				> 2				-		≤ 2		> 2							
6. Eve_Mins	-	-		-		≤ 248.15			> 248.15			-		-		-		-							
7. Day_Mins	-	≤ 285.5	> 285.5	-		≤ 285.5		> 285.5		-		-		-		≤ 285.5	> 285.5								
8. Day_Charge	-	-	≤ 48.53	¿ 48.53	-	-		≤ 48.53		> 48.53	-	-		-		-	≤ 48.53	> 48.53							
9. Intl_Charge	-	-	-	-	-	≤ 3.55	> 3.55	≤ 3.55	> 3.55	-	-	≤ 3.55	> 3.55	-		-	-	-							
1. Churn	x	-	-	x	-	-	x	-	x	x	x	-	x	x	x	-	-	x							
2. Non Churn	-	x	x	-	x	x	-	x	-	-	-	x	-	-	-	x	x	-							
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18						

(a)



		0																							
1. Intl_Plan		0																							
2. Intl_Calls	≤2	> 2																							
3. Day_Mins	-	≤ 168.05			168.05 ≤ 221.85						221.85 ≤ 248.65						> 248.65								
4. Vmail_Plan	-	-			-					0					1			0		1					
5. CustServ_Calls	-	≤ 3		> 3	≤ 3			> 3			≤ 3			> 3	-			-		-					
6. Eve_Mins	-	-		-	-	≤ 248.15		> 248.15		≤ 248.15		> 248.15		-	-			-		-					
7. Intl_Charge	-	≤ 3.55	> 3.55	-	≤ 3.55	> 3.55	-	≤ 3.55	> 3.55	≤ 3.55	> 3.55	-		-	≤ 3.55	> 3.55	-	≤ 3.55	> 3.55	-	≤ 3.55	> 3.55				
1. Churn	x	-	x	x	-	x	x	-	x	-	x	x	x	x	-	x	x	-	x	x	-	x			
2. Non Churn	-	x	-	-	x	-	-	x	-	x	-	-	-	-	x	-	-	x	-	-	x	-			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18						

(b)

1. Intl_Plan	1											
2. Intl_Calls	-											
3. Day_Mins	≤ 168.05		168.05 ≤ 221.85				221.85 ≤ 248.65				> 248.65	
4. Vmail_Plan	-		-				0			1	0	1
5. CustServ_Calls	≤ 3	> 3	≤ 3	> 3			≤ 3		> 3	-	-	-
6. Eve_Mins	-	-	-	≤ 248.15		> 248.15	≤ 248.15	> 248.15		-	-	-
7. Intl_Charge	-	-	-	≤ 3.55	> 3.55	-	-	-	-	-	-	-
1. Churn	-	x	-	x	-	-	-	x	x	-	x	-
2. Non Churn	x	-	x	-	x	x	x	-	-	x	-	x
	19	20	21	22	23	24	25	26	27	28	29	30

Figure 2.2: Decision table (a) corresponding to the AntMiner+ rule set in the lower panel of Table 2.5, and (b) corresponding to the Ripper rule set in the upper panel of Table 2.5

Although this small example might seem rather irrelevant, it illustrates perfectly that a data mining modeling technique should at least provide the possibility to impose constraints on a model, in order to include common domain knowledge and to improve the reliability of the model. This will enhance the comprehensibility of models and allow users to understand the workings of the model and the modeled relations between the variables inside the model.

To sum up, the results of the experiments show that ALBA improves learning by classification techniques and increases the accuracy, sensitivity, and specificity of the resulting models. The results are limited however by the performance of the support vector machine which depends on the data set. AntMiner+ on the other hand results in accurate, comprehensible, but most important of all justifiable models. Since decision makers are reluctant to use unintuitive models, regardless of their accuracy, they will probably discard models that do not correspond with domain knowledge (Martens et al., 2006, 2007b).

2.5 Conclusions and future research

As discussed in the literature review, churn prediction models should be both accurate and comprehensible in order to improve the efficiency of

retention marketing campaigns. This chapter presents the application of AntMiner+ and ALBA on a publicly available CCP data set. Both techniques explicitly seek to induce accurate as well as comprehensible rule sets. The results are benchmarked to C4.5, Ripper, SVM, and logistic regression. It is shown that ALBA, combined with Ripper or C4.5, results in the highest accuracy, while sensitivity is the highest for C4.5 and Ripper applied on an oversampled data set. AntMiner+ yields less sensitive rule sets, but allows to include domain knowledge and results in comprehensible rule sets which are much smaller than the rule sets induced with C4.5. Ripper also results in small and comprehensible rule sets, but may lead to an unintuitive model that violates domain knowledge. The comprehensibility of a churn prediction model is important since it facilitates the interpretation and the practical use of a model for marketing purposes. Comprehensibility also allows to check the concordance of a model with domain knowledge, which is of great importance since the intuitiveness of a model determines whether or not a model will be accepted by the end-users, and as such whether the model will effectively serve its purpose. AntMiner+ allows to include domain knowledge by imposing monotonicity constraints, leading to intuitive correct models that are still comprehensible and accurate, as proven by the results of the experiments.

Chapter 3

RULEM: rule learning with monotonicity constraints for ordinal classification

What men want is not knowledge, but certainty.

Bertrand Russel (1872 - 1970)

Abstract

In many real world applications classification models are required to be in line with domain knowledge and to respect monotone relations between predictor variables and the target class in order to be acceptable for implementation. This chapter presents a novel algorithm, RULEM, to induce monotone ordinal rule based classification models[1]. The proposed approach can be applied in combination with any rule- or tree-based classification technique, since monotonicity is guaranteed during a postprocessing step. The algorithm checks whether a rule set or decision tree violates the imposed

[1]Verbeke, W., Martens, D., Baesens, B., 2011c. Rulem: Rule learning with monotonicity constraints for ordinal classification. IEEE Transactions on data and knowledge engineering, under review

monotonicity constraints, and existing violations are resolved by inducing a set of additional rules which enforce monotone classification. The algorithm is able to handle non-monotonic noise, and can be applied to both partially and totally monotone problems with an ordinal target variable. Using the RULEM concept, two novel justifiability measures are introduced which allow to calculate the extent to which a classification model is in line with domain knowledge expressed in the form of monotonicity constraints. An extensive benchmarking experiment indicates that RULEM preserves the predictive power of the rule induction technique while guaranteeing monotone classification, at the cost of a small increase in the size of the rule set.

3.1 Introduction

Classification algorithms are a family of data mining techniques that are used to predict group or class membership of data instances (Hastie et al., 2001; Tan et al., 2006). In ordinal classification, the values of the target class possess a *natural* ordering, e.g., from small to large or from good to bad. A typical example of an ordinal classification problem is the estimation of bond ratings based on financial information (Van Gestel et al., 2007). Ratings represent the risk involved in a financial product, for instance a corporate or government bond. Financially healthy organizations with good perspectives are rated higher by rating agencies than organizations in financial distress. The target class, i.e., the rating, hence incorporates an ordering from good to bad. Other examples are the classification of customers in segments according to future spending, or any other classification problem with a continuous target variable that can be segmented into categories, such as for instance age, value, etc. Also binary classification problems can be of ordinal nature, such as good versus bad loan applications in credit scoring (Baesens et al., 2003b) and false versus non-false in customer churn prediction (Verbeke et al., 2011e).

Many powerful classification algorithms have been developed that are able to classify instances with high precision. The workings of most of these classification algorithms are based on modeling repeated patterns or correlations which are present in the data. However, it may occur that observations which are very evident to classify by a human domain expert, do not appear frequently enough in the data in order to be appropriately modeled

by a data mining algorithm. Hence, the intervention and interpretation of the induced model by a domain expert still remains crucial in many applications. A data mining approach that takes into account the knowledge representing the experience of domain experts is therefore much preferred and of great focus in current data mining research (Martens et al., 2011). Domain knowledge in a classification setting is typically expressed as a relation between an attribute and the target class. For instance, the rating of a bond is expected to be positively related with the solvency of a company. Such a relation can be translated in a mathematical constraint that is imposed on the resulting classification model (Daniels and Velikova, 2010), as will be discussed in Section 3.2.

Monotonicity is the most encountered domain constraint in the literature to be incorporated within a classification model. A positive (negative) monotonicity constraint demands that an increase in a certain input cannot lead to a decrease (increase) in the output. Monotonicity constraints exist in almost any domain. For instance in medicine, predictive classification models are used to predict the recurrence of breast cancer based on the characteristics of a patient (Delen et al., 2005). Domain knowledge states that an increase in tumor size leads to higher probability of recurrence. Therefore, a predictive model that classifies two patients with a small and a large tumor, assuming all other characteristics to be exactly the same, as respectively having a high and a low probability of recurrence, violates the expected monotone relation between probability of recurrence and tumor size.

In this chapter a novel technique is presented to induce monotone classification models, and more specifically monotone rule or tree based classifiers, which incorporate and respect monotonicity constraints. The proposed technique is called RULEM, which is an acronym for RUle LEarning with Monotonicity constraints. RULEM enforces monotonicity during a postprocessing step, i.e., after the classifier is induced. Therefore RULEM can be applied in combination with any rule or decision tree induction technique, which is a major asset. The RULEM algorithm exists of two modules. The first module checks whether an existing rule set or decision tree violates the imposed monotonicity constraints. If so, additional rules are induced by the second module to resolve the violations and to guarantee monotone classification.

The advantages of the novel technique are manyfold. Since monotonic-

ity is enforced during a postprocessing step, there is no interference with the inner workings of the classification technique that RULEM is combined with. The *optimal* model induced by a classification technique is adjusted by the postprocessing module to the smallest possible extent, in order to preserve the predictive power. Furthermore, the imposed constraints can be set freely by a domain expert. The constrained variables can be selected individually and both positive and negative constraints can be imposed directly without need to preprocess the data. Moreover, the number of class labels is unrestricted, and the data set does not have to be monotone and may contain non-monotonic noise, i.e., non-monotone data pairs. The resulting classification model remains comprehensible since a minimal number of additional rules is induced, and most importantly, guarantees monotone relations between the constrained attributes and the class variable. Finally, based on the RULEM technique two novel justifiability measures are formulated which both provide an intuitive and sensible indication of the extent to which a rule or tree based classifier is in line with domain knowledge.

The remainder of this chapter is structured as follows. The next section provides a general framework to the problem of ordinal monotone classification and introduces rule based classifiers. Then, Section 3.3 reviews the existing literature on monotone ordinal classification. Sections 3.4 and 3.5 describe the two modules that constitute RULEM to detect and resolve violations of monotonicity constraints by rule sets. Next, Section 3.6 introduces the two novel justifiability measures that follow straightforward from the RULEM algorithm. Finally, Section 3.7 discusses the results of an experiment to assess the performance of the presented approach. The last section concludes the chapter and sets out some interesting issues for future research.

3.2 Monotone ordinal classification

The first part of this section provides a general framework to the problem of ordinal monotone classification (Daniels and Velikova, 2010; Lievens and De Baets, 2010). The second part discusses rule based classifiers.

3.2.1 Problem description

Let $\mathcal{X} = \prod_{i=1}^{k} \mathcal{X}_i$ be an input space represented by k attributes, features, or variables. A particular observation or instance $\mathbf{x} \in \mathcal{X}$ is defined by the vector $\mathbf{x} = (x_1, x_2, \ldots, x_k)$, where $x_i \in \mathcal{X}_i$ and $i = 1$ to k. Furthermore, a totally ordered set of labels $\mathcal{L} = \{\ell_l\}$ is defined, with $l = 1$ to h and $\ell_l < \ell_{l+1}$. A function f is defined which maps to each attribute vector $\mathbf{x}$ a label $\ell \in \mathcal{L}$, i.e., $f : \mathcal{X} \to \mathcal{L}$. In classification problems, the objective is to find an approximation $\hat{f}$ of f as close as possible according to a certain distance measure, based on the information that is contained in a data set $D = (\mathbf{x}^j, \ell^j)$ with $j = 1$ to m and m the number of observations. In the literature, a range of classification techniques have been proposed to induce classification models $\hat{f}$. For an overview, one may refer to, e.g., Hastie et al. (2001); Tan et al. (2006).

The main assumption in this chapter is that f exhibits monotonicity properties with respect to the input variables, and therefore $\hat{f}$ should obey these properties as well in a strict fashion. Two types of problems and models can be distinguished, based on the set of input variables that are monotonically related to the target variable. Finding the model $\hat{f}$ is a *totally* monotone prediction problem if this set contains all the variables in the input space. The problem is a *partially* monotone prediction problem if only a subset of the variables in the input space is monotonically related to the target variable. Total positive monotonicity of $\hat{f}$ on $\mathbf{x}$ is defined on all independent variables by:

$$\mathbf{x}^1 \geq \mathbf{x}^2 \;\; \Rightarrow \;\; \hat{f}(\mathbf{x}^1) \geq \hat{f}(\mathbf{x}^2), \tag{3.1}$$

where $\mathbf{x}^1 \geq \mathbf{x}^2$ is a partial ordering on $\mathcal{X}$ defined by $x_i^1 \geq x_i^2$ for $i = 1$ to k. Total negative monotonicity is similarly defined on all independent variables by:

$$\mathbf{x}^1 \geq \mathbf{x}^2 \;\; \Rightarrow \;\; \hat{f}(\mathbf{x}^1) \leq \hat{f}(\mathbf{x}^2), \tag{3.2}$$

Partial positive monotonicity of the classifier $\hat{f}$ is defined by:

$$\mathbf{x}_{nm}^1 = \mathbf{x}_{nm}^2 \;\text{ and }\; \mathbf{x}_m^1 \geq \mathbf{x}_m^2 \;\; \Rightarrow \;\; \hat{f}(\mathbf{x}^1) \geq \hat{f}(\mathbf{x}^2), \tag{3.3}$$

and partial negative monotonicity[2] is defined by:

$$\mathbf{x}_{nm}^1 = \mathbf{x}_{nm}^2 \;\text{ and }\; \mathbf{x}_m^1 \geq \mathbf{x}_m^2 \;\; \Rightarrow \;\; \hat{f}(\mathbf{x}^1) \leq \hat{f}(\mathbf{x}^2), \tag{3.4}$$

[2]In the remainder of this chapter we will mainly discuss positive constraints, although the presented methodology is applicable to both positive and negative monotonicity constraints.

with the subscripts nm and m referring to respectively the independent variables $\mathcal{X}$ that are non-monotonically and monotonically related to the dependent variable, and which together constitute the instance vector $\mathbf{x} = (\mathbf{x}_{nm}, \mathbf{x}_m)$. Therefore, $\mathcal{X}_{nm}$ denotes the set of *non-monotone variables*, and the complementary set $\mathcal{X}_m$ refers to the *monotone variables*.

In case of positive monotonicity constraints, a pair of instances $(\mathbf{x^1}, \mathbf{x^2})$ of the data set D is called *comparable* if $\mathbf{x}^1 \geq \mathbf{x}^2$. Furthermore, this pair is also a *monotone pair* if the relationship defined by Equation 3.1 holds. Note that although the relation f might be totally monotone, not all the pairs in the data set are necessarily monotone, since non-monotonic noise can exist in the data set D. Based on the concepts of comparable and monotone pairs, a test to measure the degree of monotonicity (DgrMon) in a data set D has been introduced by Daniels and Velikova (2010) which is defined as follows:

$$\text{DgrMon}(D) = \frac{\#\text{Monotone pairs}(D)}{\#\text{Comparable pairs}(D)} \tag{3.5}$$

The degree of monotonicity varies by definition between zero and one. A value close to one indicates that the label has an increasing monotone relationship with the independent variables. A value close to zero on the other hand indicates that the response variable is decreasing with an increase in the independent variables. A value close to 0.5 indicates either that there are no monotone relationships between the class variable and the attributes, for instance when the labels are randomly distributed, or either that there are an even amount of negative and positive monotone relations. To check the monotone effect of a variable, the degree of monotonicity can be compared for the original data and for the data with the variable removed. Equation 3.5 will be used to analyze the degree of monotonicity of the data sets that are included in the experiments in Section 3.7, and to check whether domain knowledge, i.e., the imposed monotonicity constraints, is concordant with the empirical data.

3.2.2 Rule-based classifiers

Rule-based classifiers are a type of classification model consisting of a number of *if-then* rules. Table 3.1 provides a very simple example of a rule set, which will be used throughout the chapter for illustrative purposes. The rule set assigns a rating to a company (in an entirely fictional manner) based on its *profits* and *solvency*, which is defined as the degree to which

Rule set $\mathcal{R}$									
Rule ID	**Rule antecedents**							**Rule consequent**	
r_1	*if*	*profits* ≥ 1	$\wedge$	*solvency* ≥ 3				*then*	*rating* $= A$
r_2	*if*	*profits* < 2	$\wedge$	*solvency* ≥ 1	$\wedge$	*solvency* < 2		*then*	*rating* $= B$
r_3	*else*							*then*	*rating* $= C$

Table 3.1: A simple example rule set, representing the classification of a company into three possible rating classes A, B, and C based on the values of two attributes, i.e., *profits* and *solvency*.

the current assets of a company exceed the current liabilities. The rules of this classification model are represented in a disjunctive normal form, $\mathcal{R} = (r_1 \vee r_2 \vee \ldots \vee r_n)$, with $\mathcal{R}$ the *rule set* containing n *classification rules* or *disjuncts* r_e with $e = 1$ to n. A classification rule can be expressed as

$$r_e : p_e\left(\mathbf{x}\right) \rightarrow \ell_e, \tag{3.6}$$

with

$$p_e\left(\mathbf{x}\right) = (x_1 \;\; op \;\; v_{1,e}) \wedge (x_2 \;\; op \;\; v_{2,e}) \wedge \ldots \wedge (x_k \;\; op \;\; v_{k,e}), \tag{3.7}$$

or

$$p_e\left(\mathbf{x}\right) = \bigwedge_{i=1}^{k} s_{i,e} \;\text{ and }\; s_{i,e} = (x_i \;\; op \;\; v_{i,e}), \tag{3.8}$$

where $(x_i, v_{i,e})$ is an attribute-value pair and *op* is a logical operator chosen from the set $\{=, \neq, <, >, \leq, \geq\}$. The *if*-part of a rule, $p_e\left(\mathbf{x}\right)$, is called the *precondition* or the *rule antecedent*, and contains a conjunction of *attribute tests* or *conjuncts* $s_{i,e}$. The *then*-part of each rule is called the *rule consequent*, and assigns a class label $\ell \in \mathcal{L}$ to the instances that match the precondition.

A conjunct in fact may consist of a double test on an attribute, involving two values $v_{i,e}^1$ and $v_{i,e}^2$. A double test allows to select an interval of attribute values, such as the range of the *solvency* attribute between one and two that is selected in the second rule of the rule set in Table 3.1. In order to select and assign a class label to multiple, non-adjacent intervals of a single attribute, multiple rules in the normal form are required.

An instance $\mathbf{x}$ can match multiple rules if the rule set is *ordered*. However, a class label will be assigned by the highest ranked rule that covers an

instance, i.e., the rule with the highest priority. Both Ripper (Cohen, 1995) and AntMiner+ (Martens et al., 2007b), two state-of-the-art rule induction techniques that will be applied in the experiments in Section 3.7, result in ordered rule sets. Mutually exclusive rule sets on the other hand consist of non-overlapping rules, and an instance will trigger at most one precondition. The conversion of a decision tree into rules yields a rule set that is mutually exclusive. The RULEM approach presented in this chapter is able to handle both ordered and mutually exclusive rule sets.

The final rule in an ordered rule set is typically the *default rule*, which ensures that each combination of attribute values, i.e., the entire attribute space, is covered by the set. The default rule has an empty antecedent, which is equivalent to a conjunction of attribute tests covering the entire attribute space, and is triggered when all other rules have failed to classify an instance. The class assigned by the default rule is called the default class, which is typically the majority class of the instances in the training data that are not covered by the induced rule set (Tan et al., 2006).

Rules can be represented as hypercubes in a k dimensional space, with k the number of attributes in the data set. Each test in a conjunct of a rule defines the bounds of the hypercube in a particular dimension. For instance, the simple rule set of Table 3.1 is represented by squares in the two-dimensional attribute space as depicted in Figure 3.1. The part of the attribute space that is not covered by the squares defined by the first two rules in the rule set, is assigned a class label by the default rule. Rules may not explicitly set bounds for each dimension. For instance, the first rule in the example rule set does not set an upper bound on the value of the *solvency* attribute. Therefore, in Figure 3.1 the corresponding hypercube stretches towards the maximum possible value of the *solvency* attribute, which is set to four in this example. In general, when a rule does not contain a test for an attribute, the hypercube is unbounded in the corresponding dimension and covers the entire range of possible values for the corresponding attribute. In case a conjunct consists of a single test, the hypercube is partially unbounded, and in case the conjunct consists of two tests, the hypercube is fully bounded in the corresponding dimension. The default rule does not specify any conjunct and therefore covers the entire attribute space, minus the part of the attribute space that is covered by higher ranked rules. The representation of rules as hypercubes will be used to explain the workings of the RULEM technique, and allows to visualize how violations

of monotonicity by a rule set can be detected and resolved.

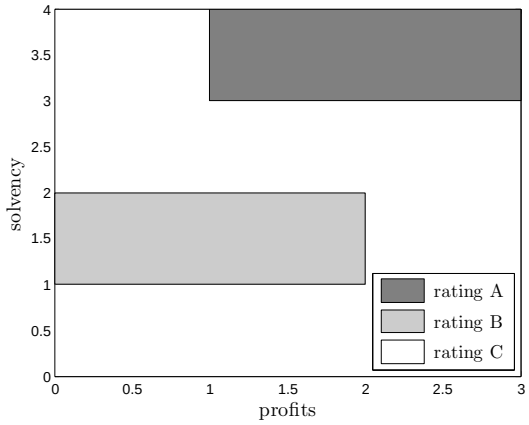


Figure 3.1: Graphical representation in the two dimensional attribute space of the rule set in Table 3.1.

3.3 Prior work

Since the introduction of the *Ordinal Learning Model* (OLM) by Sterling and Pao (1989), only a relatively small number of approaches have been developed that deal with the important practical issue of inducing ordinal monotone classification models. Table 3.2 provides a dense summary of the literature on ordinal monotone classification. Author, title, journal, and year of publication are listed in the first three columns of the table. The fourth and fifth column describe the main workings of the technique and the type of model that is induced. The sixth column indicates the stage in the data mining process where monotonicity is incorporated, i.e., in a preprocessing step (Pre), during the actual modeling phase (M), or in a postprocessing step (Post). Finally, the last three columns indicate whether the methodology is able to deal with total (T) and/or partial (P) monotonicity, whether the induced model guarantees monotone classification, yes (Y) or no (N), and whether the method can deal with non-monotonic noise in the data set. As can be seen from Table 3.2, the existing approaches differ in a number of aspects:

- The most common types of classification techniques that have been adjusted to incorporate monotonicity are decision trees, neural networks, and rule-based classifiers.
- The stage in the data mining process where monotonicity is incorporated differs for the different approaches: in the preprocessing stage, during the actual data mining step, or in a postprocessing step.
- The induced models do not always guarantee monotone classification, and sometimes only result in *more* monotone models. In case of the AntMiner+ technique (Martens et al., 2006), the user has the choice of imposing soft or hard monotonicity constraints. Soft constraints do not guarantee the final model to be monotone, but hint or direct the learning algorithm towards a monotone classifier, to a degree depending on the user's preferences. A fully monotone model can be enforced by imposing hard constraints.
- Some techniques are able to handle partially monotone problems without having to remove non-monotone variables or without having to enforce a monotone relationship for non-monotonic variables, i.e., without having to treat the problem as a totally monotone problem while it is not.
- Most approaches are able to deal with non-monotone noise in the data set, but some are not and require a preprocessing step to make the data monotone.

Except for the AntMiner+ classifier introduced by Martens et al. (2007b), all techniques included in Table 3.2 assume positive monotone relations. Therefore, in case of a negative relation between an attribute and the target variable, an extra preprocessing step is required in order to transform the attribute and its relation with the class variable into a positive constraint.

A final remark related to obtaining monotone classification models concerns the difference between *expressing* what is wanted, which is the task of a domain expert, and *enforcing* these constraints, which is the focus of our research. Sometimes, data mining reveals interesting but unexpected patterns (Silberschatz and Tuzhilin, 1995; Martens, 2008b), which might remain undetected when imposing constraints on the resulting classifier.

It should be stressed that in practice the justifiability and comprehensibility of a model will almost always be more important to the users of

Authors	Title & *Journal*	Year	What?	Type	When?	T/P	Guar.	Noise
Sterling and Pao (1989)	Learning and classification of monotonic ordinal concepts, *Computational Intelligence*	1989	The ordinal learning model (OLM) learns ordinal concepts by eliminating non-monotonic pairwise inconsistencies	Rule set	M	T	Y	Y
Ben-David (1995)	Monotonicity maintenance in information-theoretic machine learning algorithms, *Machine Learning*	1995	Monotonicity is incorporated in classification trees by combining both a standard impurity measure and a non-monotonicity measure in the splitting criterion	Decision tree	M	T	N	Y
Sill (1998)	Monotonic networks, *Advances in Neural Information Processing Systems*	1998	The output of an artificial neural network with two hidden layers is guaranteed to be monotonic by imposing signs on the weights from the input to the first hidden layer	Artificial neural network	M	T	Y	Y
Daniels and Kamp (1999)	Application of MLP Networks to Bond Rating and House Pricing, *Neural Computing and Applications*	1999	A constraint is added to a feed forward neural network that enforces all the weights among the processing elements to be non-negative, using non-decreasing transfer functions in all the nodes	Artificial neural network	M	T	Y	Y
Feelders (2000)	Prior Knowledge in economic applications of data mining, *Lecture Notes in Computer Science*	2000	In a simple generate and test approach, many different decision trees are generated, and the most monotonic one is selected	Decision tree	Post	T	N	Y
Dembczynsky et al. (2001)	Learning rule ensembles for ordinal classification with monotonicity constraints, *Fundamenta Informatica*	2001	After monotonizing the data using a Stochastic Dominance-based Rough Set Approach, a monotone rule ensemble is generated using a forwards stagewise additive modeling framework	Rule set	Pre/M	T	Y	N

Authors	Title & *Journal*	Year	What?	Type	When?	T/P	Guar.	Noise
Potharst and Feelders (2002)	Classification trees for problems with monotonicity constraints, *SIGKDD Explorations*	2002	Monotonic binary trees are built by adding corner elements to a node with proper ordinal labels, by artificially adding dummy examples to the data	Decision tree	M	T/P	Y	N
Feelders and Pardoel (2003)	Pruning for monotone classification trees, *Lecture Notes in Computer Science*	2003	Monotone classification is achieved by pruning classification trees; the method prunes the parent of the non-monotone leaf that provides the largest reduction in number of non-monotonic leaf pairs	Decision tree	Post	T	N	Y
Altendorf et al. (2005)	Learning from sparse data by eploiting monotonicity constraints, *Proceedings of the 21st conference on uncertainty in AI, Edinburgh, Scotland*	2005	Monotone Bayesian networks are built by imposing inequality constraints on the network parameters; monotonicity constraints on the parameter estimation problem are handled by imposing penalties in the likelihood function	Bayesian Network	M	T/P	Y	Y
Lang (2005)	Monotonic multilayer perceptron networks as universal approximators, *Lecture Notes in Computer Science*	2005	Constraints are imposed on the signs of the weights of a multilayer perceptron network in order to guarantee monotone classification	Artificial neural network	M	T/P	Y	Y
Daniels and Velikova (2006)	Derivation of monotone decision models from noisy data, *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*	2006	All non-monotone data pairs are removed, by iteratively changing the class of the data instance for which the increase in correctly labeled instances is maximal	-	Pre	T	N	N

Authors	**Title & *Journal***	**Year**	**What?**	**Type**	**When?**	**T/P**	**Guar.**	**Noise**
Martens et al. (2006)	Ant-based Approach to the Knowledge Fusion Problem, *Lecture Notes in Computer Science*	2006	Monotone rule sets are induced by restricting the search space of possible rules by imposing inequality signs; the approach is also able to impose soft monotonicity constraints	Rule set (binary target class)	M	T/P	Y	Y
Van Gestel et al. (2007)	Forecasting and analyzing insurance companies' ratings, *International Journal of Forecasting*	2007	Variables with opposite signs are removed from a linear regression model, the remaining parameters are re-estimated until the model is monotone	Linear regression	Post	T/P	Y	Y
Barile and Feelders (2008)	Nonparametric monotone classification with MOCA, *Eighth IEEE International Conference on Data Mining ICDM08*	2008	A monotone classifier is built to minimize L1 loss on the training data, and extended to the complete input space using a straightforward interpolation scheme that preserves monotonicity	Statistical mapping function	M	T	Y	Y
Duivesteijn and Feelders (2008)	Nearest Neighbor classification with monotonicity constraints, *Lecture Notes in Artificial Intelligence*	2008	Adjusted version of the nearest neighbor classifier that guarantees monotone classification, to be applied on monotone (relabeled) data	Nearest neighbor	M	T/P	Y	N
Lievens and De Baets (2010)	Supervised ranking in the Weka environment, *Information Sciences*	2010	The ordinal stochastic dominance learner (OSDL) solves a supervised ranking problem based on the concept of ordinal stochastic dominance	Statistical mapping function	M	T	Y	Y
Daniels and Velikova (2010)	Monotone and partially monotone neural networks, *IEEE Transactions on neural networks*	2010	The results of Sill (1998) are extended to the case of partially monotone problems	Artificial neural network	M	T/P	Y	Y

Table 3.2: Overview of the literature on ordinal classification with monotonicity constraints.

the model than the predictive power(Askira-Gelman, 1998). Users will be reluctant or even refuse to use a model that is contradicting domain knowledge, as reported by case studies in domains such as credit scoring (Van Gestel et al., 2007), medical diagnosis (Korn et al., 1998), audit mining (predicting the going concern opinion as issued by the auditor) (Martens et al., 2008), business/ICT alignment prediction (Cumps et al., 2009), and software fault prediction (Vandecruys et al., 2008). Therefore, a successful implementation of a model heavily depends on the comprehensibility and justifiability of a classification model. However, whereas the comprehensibility of a model merely depends on the choice of modeling technique and can thus quite easily be controlled, the justifiability is depending on the outcome of the modeling process, which is much harder to control.

3.4 Detecting violations of constraints

3.4.1 Rule-based classifiers and monotone classification

A totally or partially positive monotone rule-based classification model is defined as a rule set that complies with respectively Equations 3.1 or 3.3 over the entire input space $\mathcal{X} = \prod_{i=1}^{k} \mathcal{X}_i$. This means that Equations 3.1 or 3.3 apply to each possible pair of comparable attribute vectors $(\mathbf{x}^1, \mathbf{x}^2) \in \mathcal{X}$, with $\mathbf{x}^1 \geq \mathbf{x}^2$ in case of total monotonicity or $\mathbf{x}^1_{nm} = \mathbf{x}^2_{nm}$ and $\mathbf{x}^1_m \geq \mathbf{x}^2_m$ in case of partial monotonicity. To our knowledge, the only approach that is described in the literature to check whether a rule set is monotone over the entire attribute space $\mathcal{X}$ was introduced in Martens et al. (2011). However, this approach requires the conversion of a rule set in a decision table (see, e.g., Vanthienen et al. (1998b)) to calculate justifiability, which prohibits automatization and incorporation within a general algorithm to induce monotone rule sets.

An alternative approach exists in applying Equations 3.1 and 3.3 of total and partial monotonicity in a straightforward manner, and to exhaustively check the classification of all comparable attribute vector pairs in the attribute space. Remark that this is not equivalent to checking all the comparable attribute vector pairs *in a data set*, which is done for instance to calculate the degree of monotonicity as defined by Equation 3.5. In most settings the attribute space $\mathcal{X}$ is very large and checking all comparable attribute vectors infeasible. Therefore, a novel approach is introduced in this chapter which starts from the rules to check whether a rule set complies

with the imposed monotonicity constraints.

The first step of this approach consists in partitioning the k-dimensional attribute space $\mathcal{X}$ into a grid $\mathcal{G}$, which consists of elementary cells $\mathbf{g}$ with *homogeneous labeling*, meaning that all attribute vectors $\mathbf{x} \in \mathbf{g}$ yield the same class label. Cells are bounded by the values of the attribute tests in the preconditions of the rules, and by the minimum and maximum attribute values, i.e., the bounds of the attribute space $\mathcal{X}$. All the attribute vectors in an elementary cell trigger the same rule in the rule set and are assigned the same label, since by definition no rule in the rule set makes a further differentiation between the attribute values within an elementary cell.

The grid $\mathcal{G}$ of elementary cells is formally defined as follows:

$$\mathcal{G} = \prod_{i=1}^{k} \mathcal{G}_i, \tag{3.9}$$

with

$$\begin{aligned} \mathcal{G}_i &= \bigcup_{j=1}^{n_i} g_{i,j} \\ &= \bigcup_{j=1}^{n_i} [v^s_{i,j}, v^s_{i,j+1}), \end{aligned} \tag{3.10}$$

and

$$v^s_{i,1} = min(\mathcal{X}_i), \tag{3.11}$$

$$v^s_{i,n_i+1} = max(\mathcal{X}_i) + \varepsilon. \tag{3.12}$$

The elementary value ε is added to the maximum attribute value in Equation 3.12 to account for the open upper bound of the elementary intervals in Equation 3.10. Each dimension $\mathcal{X}_i$ of the attribute space $\mathcal{X}$ is partitioned into elementary intervals $g_{i,j}$, with $i = 1$ to k and $j = 1$ to n_i. The bounds of the elementary intervals are defined by the ascending set of attribute values $v^s_{i,j}$, which consists of (1) the minimum and maximum values $v^s_{i,1}$ and v^s_{i,n_i+1} of each attribute, defined in Equations 3.11 and 3.12 respectively as the lower and upper bound of each dimension; (2) the unique attribute values $v_{i,e}$ in the preconditions p_e of the rules r_e in the rule set $\mathcal{R}$, with $v_{i,e} \neq v^s_{i,1}$ and $v_{i,e} \neq v^s_{i,n_i+1}$. Each cell $\mathbf{g} \in \mathcal{G}$ consists of a unique combination of elementary intervals, i.e., $\mathbf{g} = (g_1, g_2, \ldots, g_k)$, with $g_i \in \{g_{i,1}, g_{i,2}, \ldots g_{i,n_i}\}$,

and n_i the number of intervals constituting attribute dimension $\mathcal{X}_i$. The number of elementary cells in the grid $\mathcal{G}$ equals $\prod_{i=1}^{k}(n_i)$.

RULEM adopts a convention which restricts the set of logical operators in a rule set to $\{=, \neq, <, \geq\}$. The $\leq$ and $>$ operator are converted respectively into the $<$ and $\geq$ operator by replacing attribute tests $x_i \leq v_{i,e}$ by $x_i < v_{i,e} + \varepsilon$, and attribute tests $x_i > v_{i,e}$ by $x_i \geq v_{i,e} + \varepsilon$, with ε a value smaller than the minimum difference between any two values of an attribute in the data set. This conversion increases the number of elementary intervals n_i of attribute dimension $\mathcal{X}_i$, and consequently the total number of cells in the grid, but allows to partition a rule set in a grid $\mathcal{G}$ with elementary cells that are strictly complementary, i.e., $g_{i,j} \cap g_{i',j'} = \varnothing$ with $i \neq i'$ or $j \neq j'$. Excluding the operators $\leq$ and $>$ impedes cells to overlap in the boundary values. The resulting intervals $g_{i,j}$ are all left-closed and right-open, as indicated in Equation 3.10.

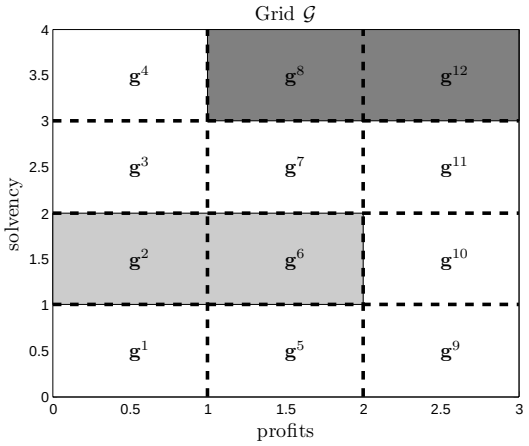


Figure 3.2: The elementary grid of the example rule set.

Figure 3.2 shows the partitioning into a grid of the simple example rule set of Table 3.1, and labels each cell in the grid. As can be seen, each cell in the grid is homogeneously labeled.

3.4.2 Detecting violations of monotonicity constraints

In a second step of the RULEM approach to detect violations of the monotonicity constraints, the *Conflict* or *C-score* is calculated for each cell

Algorithm 3 RULEM pseudo-code to calculate the C-score of a rule set $\mathcal{R}$

construct grid $\mathcal{G} = \prod_{i=1}^{k} \mathcal{G}_i$, with i the attribute dimension
$\mathcal{G}_i = \bigcup_{j=1}^{n_i} g_{i,j}$, with $g_{i,j}$ the elementary intervals
for all cells $\mathbf{g}^u \in \mathcal{G}$, with $u = 1$ to n, $n = \prod_{i=1}^{k} n_i$, and $\mathbf{g}^u = (g_1^u, g_2^u, \ldots, g_k^u)$ **do**
  set C-score($\mathbf{g}^u$) = 0
  for all constrained dimensions $\mathcal{X}_m \in \mathcal{X}$ **do**
    for all cells $\mathbf{g}^{u,j} \in \mathcal{G}$, with $j = 1$ to n_m, $g_{i \neq m}^{u,j} = g_i^u$, and $g_m^{u,j} = g_{m,j}$ **do**
      if positive constraint **then**
        if $g_m^{u,j} < g_m^u$ AND $\ell(\mathbf{g}^{u,j}) > \ell(\mathbf{g}^{u,j})$ **then**
          violation of constraint
          increase C-score($\mathbf{g}^u$)
        else if $g_m^{u,j} > g_m^u$ AND $\ell(\mathbf{g}^{u,j}) < \ell(\mathbf{g}^{u,j})$ **then**
          violation of constraint
          increase C-score($\mathbf{g}^u$)
        end if
      else if negative constraint **then**
        if $g_m^{u,j} < g_m^u$ AND $\ell(\mathbf{g}^{u,j}) < \ell(\mathbf{g}^{u,j})$ **then**
          violation of constraint
          increase C-score($\mathbf{g}^u$)
        else if $g_m^{u,j} > g_m^u$ AND $\ell(\mathbf{g}^{u,j}) > \ell(\mathbf{g}^{u,j})$ **then**
          violation of constraint
          increase C-score($\mathbf{g}^u$)
        end if
      end if
    end for
  end for
end for
C-score($\mathcal{R}$) = $\sum_{u=1}^{n}$ C-score($\mathbf{g}^u$)

in the grid. The C-score of a cell indicates the extent to which the label of a cell violates the imposed constraints. It is important to acknowledge that the label of a cell on itself does not cause any violation, but that in combination with the labels of other cells a violation may exist. The C-score of a cell in the grid is calculated as the number of other cells it is conflicting

with, with respect to the monotonicity constraints. The sum of all C-scores of the cells yields the total C-score of the rule set; if the total C-score is equal to zero, the rule set respects the imposed monotonicity constraints. If the total C-score is different from zero, the rule set violates the imposed monotonicity constraints. A formal algorithm describing the calculation of the C-score of each cell in the grid $\mathcal{G}$ by the RULEM algorithm is provided by Algorithm 3.

To illustrate the concept of the C-score let us return to the simple rule set of Table 3.1 and the corresponding representations of Figures 3.1 and 3.2. Assume that a positive constraint is imposed on the *solvency* attribute. According to this constraint, cell $\mathbf{g}^1$ is not in conflict with cells $\mathbf{g}^2$, $\mathbf{g}^3$, or $\mathbf{g}^4$ since for an increasing value of the *solvency* attribute, neither of these cells assigns a label that is smaller than the label assigned by cell $\mathbf{g}^1$. Therefore cell $\mathbf{g}^1$ is assigned a C-score equal to 0. Cell $\mathbf{g}^2$ on the other hand is not conflicting with cell $\mathbf{g}^1$ (conflicts are commutative, meaning that if cell $\mathbf{g}^1$ is (not) in conflict with cell $\mathbf{g}^2$, then cell $\mathbf{g}^2$ is (not) in conflict with cell $\mathbf{g}^1$), but it is in conflict with cells $\mathbf{g}^3$ and $\mathbf{g}^4$; labels assigned by these cells are smaller than the label assigned by cell $\mathbf{g}^2$, although instances that are situated in cells $\mathbf{g}^3$ and $\mathbf{g}^4$ may have an identical value for *profits* and a larger value for *solvency*, which is in conflict with the imposed constraint. Hence, cell $\mathbf{g}^2$ is assigned a C-score equal to 2. When imposing a second, positive constraint on the *profits* attribute, the C-score of the cell $\mathbf{g}^1$ remains zero, since no conflict exists with cells $\mathbf{g}^5$ and $\mathbf{g}^6$. The C-score of cell $\mathbf{g}^2$ on the other hand increases to three, since another conflict arises with cell $\mathbf{g}^{10}$. For each cell in the grid the C-score can be calculated in a similar way, and the resulting values are presented in Figure 3.3.

3.5 Resolving violations of constraints

The previous section explained the workings of the RULEM algorithm to detect violations of monotonicity constraints by a rule set. In a second step, the RULEM algorithm resolves these violations by adding complementary rules to the rule set, as will be explained in Section 3.5.1. Section 3.5.2 will describe how the RULEM algorithm induces these complementary rules, in order to guarantee monotone classification with a minimum impact on the predictive power and the size of the rule set.

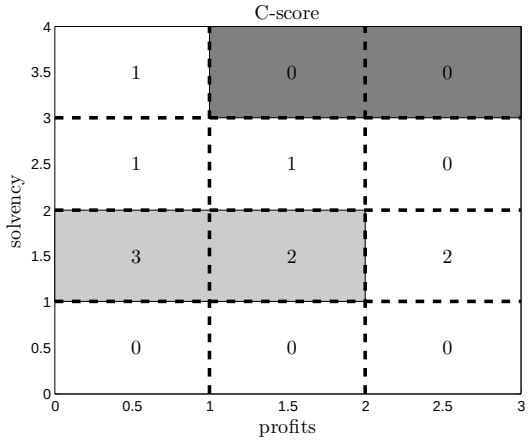


Figure 3.3: The C-scores of the elementary cells of the example rule set.

3.5.1 Adding rules to resolve monotonicity violations

Violations of monotonicity constraints by a rule set can be resolved by adding complementary rules. For instance, the example rule set of Figure 3.1 can be *monotonized* by adding rules that assign rating A to cells $\mathbf{g}^3$, $\mathbf{g}^4$, $\mathbf{g}^7$, and $\mathbf{g}^{11}$, and rating B to cell $\mathbf{g}^{10}$, as shown in Figure 3.4(a). By adding these rules, the C-scores of all cells in the grid become zero, which indicates that the resulting rule set respects the positive monotonicity constraints on both attributes. Since cells g^3, g^7, and g^{11} are identically labeled, neighboring, and constituting a rectangular volume in the attribute space (i.e., a hypercube), the related additional rules to relabel these cells can be merged into a single rule, and in total only three additional rules are required to monotonize the rule set, as provided by Table 3.3[3]. The additional complementary rules have priority over the original rules in the rule set and are therefore assigned the highest order in the rule set, since the additional rules have to overrule the existing labels. Consequently, the order of the original rules is decreased. The ordering among the additional rules does not matter, since by definition they cover non-intersecting subspaces of the attribute space.

In many cases multiple solutions exist to resolve violations of monotonic-

[3]In fact, RULEM will even further merge the additional rules with the original rules, yielding a rule set of in total only two rules.

ity by adding complementary rules. For instance, an alternative solution to monotonize the rule set of Table 3.1 would be to assign class label A to cell $\mathbf{g}^3$, and class label B to cells $\mathbf{g}^4$, $\mathbf{g}^7$, $\mathbf{g}^10$, and $\mathbf{g}^{11}$, as shown in Figure 3.4(b). Figure 3.4(c) proposes a third possible solution to resolve the violations, i.e., by *overruling* the entire second rule of the rule set and changing its class label to C. Many more solutions could be thought of, which illustrates the need for a formal strategy to induce complementary rules. The next section will elaborate such a solution strategy, with the aim to induce the *optimal* set of additional rules. Optimal will be defined in terms of both the predictive power, which is preferably high, and the number of rules, which is preferably small in order to maintain comprehensibility.

The RULEM approach to resolve violations of monotonicity constraints by adding complementary rules can also be applied to ensure monotone classification by decision trees. A decision tree can easily be converted into a rule set, which can then be checked for violations of monotonicity by applying the RULEM algorithm. Violations can be resolved by adding rules, and the resulting rule set can be converted back into a decision tree or table (Vanthienen et al., 1998a,b). A decision tree induction technique is included in the experimental part of the chapter to illustrate this approach.

An alternative to the proposed approach of adding rules in a postprocessing step would be to alter the inner workings of existing rule induction techniques, or to develop a novel rule-based classification technique, in order to incorporate monotonicity constraints during the induction of a rule set. However, whereas in case of a binary target variable this in-

Rule set $\mathcal{R}$										
Rule ID	**Rule antecedents**							**Rule consequent**		
$r_{add,1}$	*if*	*profits* < 1	$\wedge$	*solvency* ≥ 3				*then*	*rating* $= A$
$r_{add,2}$	*if*			*solvency* ≥ 2	$\wedge$	*solvency* < 3		*then*	*rating* $= A$
$r_{add,3}$	*if*	*profits* ≥ 2	$\wedge$	*solvency* ≥ 1	$\wedge$	*solvency* < 2		*then*	*rating* $= B$
r_1	*if*	*profits* ≥ 1	$\wedge$	*solvency* ≥ 3				*then*	*rating* $= A$
r_2	*if*	*profits* < 2	$\wedge$	*solvency* ≥ 1	$\wedge$	*solvency* < 2		*then*	*rating* $= B$
r_3	*else*							*then*	*rating* $= C$

Table 3.3: The rule set resulting from adding complementary rules to the rule set of Table 3.1 to resolve the violations of the positive monotonicity constraint imposed on the attributes *profits* and *solvency*, according to the solution of Figure 3.4(a).

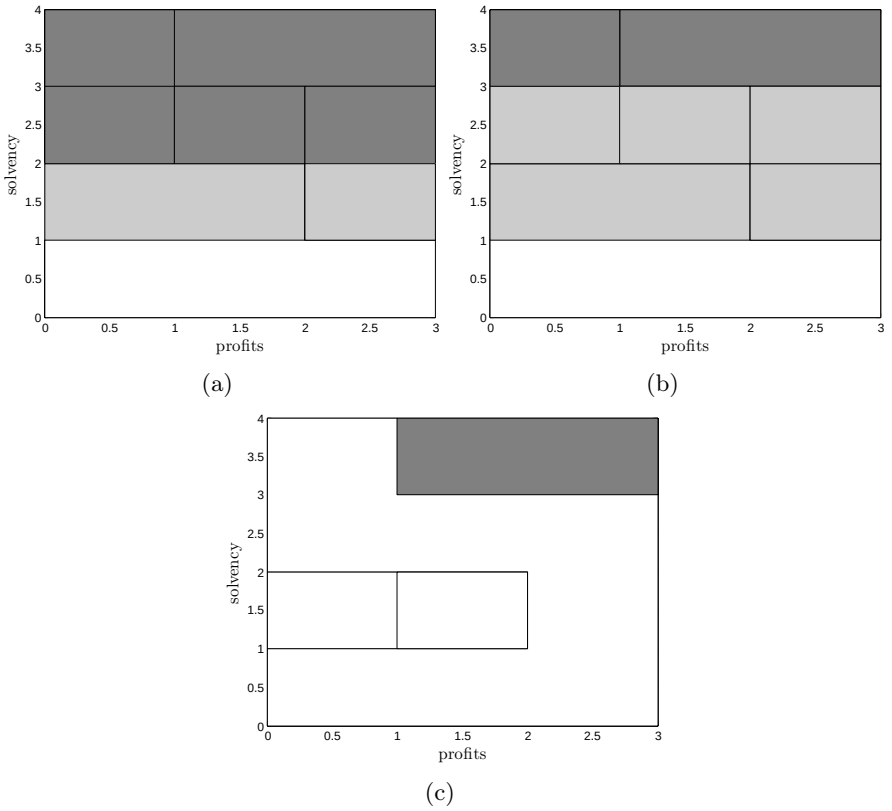


Figure 3.4: Adding complementary rules to resolve violations of monotonicity.

volves a rather simple adjustment to sequential covering algorithms such as AntMiner+ (Martens et al., 2006) or Ripper (see Section 3.7 and Appendix A), in case of multiclass classification problems this would likely lead to poor classification performance. Rule induction techniques typically apply a sequential covering approach, inducing one rule after another using a greedy stepwise procedure. To produce a monotone rule set in a direct manner, two alternative approaches could be conceived. The first approach would be to alter the way rules are induced in order to only produce rules that respect the imposed monotonicity constraints by reducing the search space. The second possible approach is to discard rules that are not compliant with the imposed monotonicity constraints immediately after their

induction, before a next rule is induced. Both approaches would lead in the end to a monotone rule set, since all the rules in the set are monotone by themselves. However, not all rules in a rule set have to be monotone for the rule set as a whole to be monotone, as proven by the rule set of Table 3.3. Therefore when enforcing monotone classification during the induction of a rule set, the search space for rules to be included in the rule set is heavily restricted, i.e., the solution space of possible (monotone) classifiers $\hat{f}$ is strongly reduced. Furthermore, since the attribute space is initially unlabeled, it might be impossible to check whether a rule is monotone before the entire rule set is induced. For instance, when the second rule in the example rule set of Table 3.1 is induced, the third rule, i.e., the default rule, has not been defined yet, and no label is assigned to cells $\mathbf{g}^1$, $\mathbf{g}^3$, and $\mathbf{g}^4$. Hence, at the moment rule r_2 is induced it cannot be judged whether it is monotone or not. Therefore, it seems appropriate to opt for an approach that enforces monotonicity during a postprocessing phase, in order to induce a monotone classifier with good predictive power.

3.5.2 Solution strategy

The basic idea behind the solution strategy to resolve violations is that the impact of the additional rules on the workings of the original rule set should be as small as possible, in order to restrict possible negative effects on the predictive performance, and to limit the number of additional rules. As illustrated by Figure 3.4, multiple solutions may exist to resolve violations of monotonicity by adding rules. RULEM aims to induce the *optimal* solution; the set of additional rules that is induced to resolve the violations and to guarantee monotonicity should preserve or even improve predictive power and consist of a small number of rules. Algorithm 4 provides the pseudocode of the solution strategy.

The RULEM algorithm resolves violations and guarantees monotonicity by adjusting the labels of cells in the grid in a stepwise procedure. In each step, RULEM first calculates the C-score of each cell in the elementary grid resulting from the current total rule set, which consists of the original rules and the complementary rules that have been generated in previous iterations of the procedure. Subsequently, the cells are ranked in order of decreasing C-score. Then, the procedure iteratively selects the cell with the highest C-score that has not been selected before during this step of the procedure, until *adjusting* the label of a selected cell yields a reduced total

Algorithm 4 Pseudo-code of the RULEM algorithm to resolve violations of monotonicity constraints

while C-score of rule set $\mathcal{R} > 0$ **do**
  sort cells $\mathbf{g} \in \mathcal{G}$ in descending order of C-score, yielding the set $\mathbf{g}^{s,j}$, with $j = 1$ to n
  set $j = 0$
  while no decrease in C-score of rule set $\mathcal{R}$ **do**
    $j = j + 1$
    select cell $\mathbf{g}^{s,j}$
    for all labels $\ell_l \in \mathcal{L}$, with $l = 1$ to h **do**
      add rule $r_{\text{add}} : \mathbf{g^{s,j}} \rightarrow \ell_l$ to rule set $\mathcal{R}$
      calculate C-score$(\mathcal{R})^l$
      remove rule r_{add} from $\mathcal{R}$
    end for
    select l, with $\arg\min_l(\text{C-score}(\mathcal{R})^l)$
    if C-score$(\mathcal{R})^l <$ C-score$(\mathcal{R})$ **then**
      add rule $r_{\text{add}} : \mathbf{g^{s,j}} \rightarrow \ell_l$ to rule set $\mathcal{R}$
    end if
  end while
end while
merge additional rules
remove redundant rules

C-score. The label of a cell is adjusted by assigning the label that results in the smallest total C-score. When a cell is selected, a rule is added to the rule set that adjusts the labeling of the selected cell, and the procedure continues to the next step by recalculating the C-scores of the total rule set, including the newly generated rule. As long as the total C-score of the rule set is larger than zero, the procedure continues to adjust labels of cells by adding rules. Once the total C-score of the complemented rule set equals zero, the added rules are merged to reduce the number of complementary rules. Finally, redundant rules, such as original rules that are fully overruled by additional rules, are removed from the rule set.

3.5.3 Refining the solution strategy

The solution strategy of the RULEM algorithm, as presented in the previous section, can be further refined in order to yield improved classification performance of the final total rule set, consisting of the original rules induced by any rule or decision tree induction technique and the additional complementary rules induced by RULEM. The definition of the C-score of a cell, which in fact controls the induction of additional rules, can be extended to take into account the predictive power of the cells it is conflicting with, leading to the definition of the I-score. The I-score is calculated similar to the C-score, but instead of counting the number of cells a cell is conflicting with, it counts the number of correctly classified instances within the cells it is conflicting with. Hence, the I-score is calculated following Algorithm 3, increasing the score of a cell by adding the number of correctly classified instances of a conflicting cell. By implementing the I-score, the RULEM algorithm will avoid to induce rules that overrule existing rules with high accuracy, thus preserving the predictive accuracy. The C-score on the other hand guides the algorithm towards a small number of additional rules, by selecting cells that conflict with many other cells.

As discussed above, besides the justifiability, both the predictive power and the comprehensibility of a rule set are two important requirements for a successful implementation of a classification model. By implementing the CI-score, which is defined as a weighted sum of the normalized C-score and the normalized I-score with a weight that can be set by the user, the RULEM algorithm aims to induce a small set of additional rules that preserves the predictive power of a rule set. The C-scores and I-scores of the cells are normalized to lie within a range between zero and one by dividing each score by the largest prevalent score. Adjusting the weights of the C- and the I-score in the formula of the CI-score allows to indicate a preference towards a smaller rule set by increasing the weight of the C score, possibly at the cost of losing predictive accuracy, or towards a rule set with good predictive accuracy by increasing the weight of the I-score, possibly at the cost of generating more additional rules. By default, the weights of both scores are set equal to 0.5.

The total CI-score of a rule set provides to some extent an indication of the justifiability of a rule set and the number of additional rules that will be required to resolve the violations. A high CI-score indicates that the rule set is strongly violating the imposed constraints, and that many

additional rules are likely needed to resolve the violations. This in turn might result in poor predictive power. Therefore, in the experiments in Section 3.7, a maximum value for the CI-score is implemented. If a rule set yields a total CI-score larger than the maximum value, no additional rules will be generated and the rule set is deemed to be unsuitable to generate an acceptable final classification model.

When a rule set yields a high CI-score, the imposed constraints appear to be contradicted by the empirical evidence in the data, and the imposed constraints may need to be reconsidered by the user. Hence, by calculating the CI-score, the RULEM algorithm provides the user an indication of the reliability of the imposed constraints. However, the relation between the CI-score of a rule set on the one hand, and its justifiability and the required number of additional rules on the other hand is not explicitly defined. Although unlikely, it may well be for instance that a very high CI-score can be resolved by adding a single rule to the rule set. Hence, the CI-score provides merely an indication of the justifiability of a rule set. In the next section, two formal measures based on the RULEM algorithm to calculate the justifiability of a rule set will be introduced.

3.6 Measuring justifiability

In this section two novel justifiability measures are defined based on the C-scores of the elementary cells in the grid as calculated by the RULEM algorithm. These measures indicate the extent to which a rule set or decision tree is in concordance with the imposed monotonicity constraints. The second section introduces a feature of the RULEM algorithm which allows to set a minimum justifiability that a rule set or decision tree needs to attain. This parameter allows to restrict the number of additional rules induced by the RULEM algorithm in case a trade-off exists between justifiability and comprehensibility.

3.6.1 Two novel justifiability measures

The idea of measuring the justifiability of a rule or tree based classification model, i.e., the extent to which a rule set or decision tree is intuitively correct and respects domain knowledge in the form of hard monotonicity constraints, was pioneered by Martens et al. (2011). A metric was introduced,

which requires the conversion of rule sets or decision trees into decision tables to find violations of monotonicity. The number of existing violations are then weighted to yield a measure between one and zero. Decision tables are a tabular representation used to describe and analyze decision situations (Vanthienen et al., 1998a,b).

The C-scores of the elementary cells in the grid as calculated by the RULEM algorithm allow to formulate two novel justifiability metrics, the rule set monotonicity Space (RULEMS) and Fraction (RULEMF) measure. These measures can be calculated in an automated manner and have a straightforward intuitive interpretation.

The RULEMS measure is calculated as one minus the fraction of the attribute space that is occupied by the elementary cells in the grid with a C-score that is different from zero. The intuition behind this measure is that instances in these cells are not classified in line with the imposed monotonicity constraints. The lower and upper bound of a continuous numeric attribute are defined by Equations 3.11 and 3.12, which allows to calculate the fraction of the related dimension that is occupied by a rule. The fraction of a dimension related to a categorical variable that is covered by a rule is defined as the fraction of the categorical values that are covered by the rule. For instance, imagine an attribute space that consists of a single, categorical attribute with ten possible values. A rule that assigns a certain class label to instances with a value of this attribute equal to one of these ten possible values covers $1/10 = 10\%$ of the attribute space.

Let us return to the simple example rule set of Table 3.1 represented in Figure 3.1. The total size of the attribute space equals $4 \times 3 = 12$. The size of the attribute space covered by cells with a C-score different from zero, as indicated by Figure 3.3, equals $3 \times 1 + 2 \times 1 + 1 \times 1 = 6$. The RULEMS justifiability measure therefore equals $1 - 6/12 = 0.5$, which means that the rule set of Table 3.1 is 50% in line with the imposed constraint. The following equation provides a formal definition of the RULEMS measure:

$$\text{RULEMS} = 1 - \frac{\sum_{\mathcal{G}} \mathcal{V}(\mathbf{g}^{C \neq 0})}{\sum_{\mathcal{G}} \mathcal{V}(\mathbf{g})}, \tag{3.13}$$

with $\mathbf{g}^{C \neq 0}$ the elementary cells in the grid $\mathcal{G}$ with C-score different from zero, and $\mathcal{V}(\mathbf{g})$ the volume in the attribute space occupied by an elementary cell.

The RULEMS measure has a very intuitive interpretation, i.e., the frac-

tion of the attribute space that is in concordance with domain knowledge expressed in the form of monotonicity constraints. It suffers however from one major drawback, which is the fact that large parts of the attribute space might be empty or sparsely populated with data points or observations, and other parts more densely. Therefore, from an intuitive point of view it might be more interesting to know how many of the observations in the data set are classified monotonically by a rule set. This is the basic idea behind the RULEMF measure, which indicates the fraction of the data instances that are classified in accordance with the imposed monotonicity constraints. The RULEMF measure hence equals one minus the fraction of instances that are covered by the elementary cells with a C-score larger than zero:

$$\text{RULEMF} = 1 - \sum_{\mathcal{G}} Cov(\mathbf{g}^{C \neq 0}), \tag{3.14}$$

with $Cov(\mathbf{g}^{C \neq 0})$ the coverage of the elementary cells $\mathbf{g}^{C \neq 0}$ in the grid $\mathcal{G}$ with a C-score different from zero. Both the RULEMS and RULEMF measure are independent of the selected set of additional rules, and only depend on the original rule and data set.

3.6.2 Setting a minimum justifiability

In practical applications a trade-off may exist between justifiability on the one hand side, and comprehensibility and predictive power on the other hand. Monotonizing a rule set or decision tree using RULEM, resulting in a 100% justifiable classifier, might come at the cost of a large number of additional rules and/or a large decrease in predictive power. Typically, the more rules or branches and leaves, the less comprehensible a rule set or decision tree is considered to be (Martens et al., 2011). This might be undesirable when the model needs to be comprehensible, for instance to understand why exactly a rating is assigned by a classification model. In case the number of rules should be as small as possible to guarantee comprehensibility, or when the predictive power should be maintained at a certain level, RULEM can be tuned in order to result in a smaller number of additional rules or to yield a required minimum level of predictive power, at the cost of a reduction in justifiability.

The RULEMS and RULEMF measures introduced in the previous section can be used as parameters within RULEM to fine-tune the trade-off

between justifiability, and comprehensibility or predictive power. When the user deems the decrease in predictive power or the number of additional rules added by RULEM to a rule set or a decision tree to be too large, a smaller value of the RULEMS or RULEMF justifiability measure that needs to be attained by the resulting monotone classifier can be specified. For instance, instead of demanding that a rule set is entirely monotone and yields a RULEMS or RULEMF measure equal to 1, the user could set the required RULEMS or RULEMF measure equal to a lower value, e.g., 0.95.

The RULEM algorithm will then induce in a first step additional rules following Algorithm 4 to yield a fully monotone classifier. In a second step a number of additional rules will be removed again from the rule set. Therefore, the rules are ranked according to their volume in case a RULEMS value is set, and according to their coverage in case a RULEMF value is set. Subsequently, rules are added to the rule set in order from large to small volume or coverage, until the requested justifiability is attained by the resulting rule set. A dense description of this procedure in pseudo-code is provided as Algorithm 5.

Equivalently, RULEM allows to specify a maximum number of additional rules or a minimum value for the predictive power that needs to be guaranteed by the final rule set. A similar procedure is followed as described by Algorithm 5. First, additional rules are generated to yield a fully monotone classifier. In case a maximum number of additional rules is specified, the induced additional rules are ranked by coverage or volume, depending on the preferred justifiability measure, respectively RULEMF or RULEMS. This will ensure the resulting rule set to yield maximal justifiability for the specified number of additional rules. In the while loop of the algorithm additional rules are added to the resulting rule set until the maximum number of rules is reached. In case a minimum predictive power is required, the additional rules are ranked according to both their classification accuracy and the justifiability of the resulting rule set, in order to attain maximal justifiability for the specified predictive power.

3.7 Experiments

A benchmarking experiment is set up in order to assess the performance of the RULEM algorithm to induce monotone classifiers, both in terms of classification accuracy, induced number of rules, and average number of

Algorithm 5 Pseudo-code of the RULEM algorithm to induce a rule set with a minimum justifiability parameter

original rule set $\mathcal{R}_{or} = \bigvee_{e=1}^{n_{or}} r_{e,or}$
apply RULEM to induce additional rules
$\mathcal{R}_{add} = \bigvee_{e=1}^{n_{add}} r_{e,add}$, with $r_{e,add} : p_{e,add}(\mathbf{x}) \rightarrow \ell_{e,add}$
switch: set minimal justifiability value
case: RULEMS $= v_s$
  rank additional rules: $\mathcal{R}^r_{add} = \bigvee_{e=1}^{n_{add}} r^r_{e,add}$,
  with $\mathcal{V}(p^r_{e,add}) \geq \mathcal{V}(p^r_{e+1,add})$
  set $\mathcal{R}_{tot} = \mathcal{R}_{or}$
  set $i = 1$
  while RULEMS$(\mathcal{R}_{tot}) < v_s$ **do**
    add rule $r^r_{i,add}$ to rule set $\mathcal{R}_{tot}$
    RULEMS$(\mathcal{R}_{tot}) =$ RULEMS$(\mathcal{R}_{tot}) + \mathcal{V}(p^r_{e,add})$
    set $i = i + 1$
  end while
case: RULEMF $= v_f$
  rank the additional rules $\mathcal{R}^r_{add} = \bigvee_{e=1}^{n_{add}} r^r_{e,add}$,
  with $cov(p^r_{e,add}) \geq cov(p^r_{e+1,add})$
  set $\mathcal{R}_{tot} = \mathcal{R}_{or}$
  set $i = 1$
  while RULEMF$(\mathcal{R}_{tot}) < v_f$ **do**
    add rule $r^r_{i,add}$ to rule set $\mathcal{R}_{tot}$
    RULEMF$(\mathcal{R}_{tot}) =$ RULEMF$(\mathcal{R}_{tot}) + cov(p^r_{e,add})$
    set $i = i + 1$
  end while
end switch

terms per rule. Section 3.7.1 describes the data sets and the imposed constraints, and Section 3.7.2 the experimental setup. Finally, in Section 3.7.3, the results of the experiment are discussed.

3.7.1 Data sets

Fourteen publicly available data sets with both monotone (positive and negative) and non-monotone attributes and an ordinal target variable have been collected. The main characteristics of these data sets are summarized

Name	**# Att.**	**# Obs.**	**# Class**	**Source**	**# P.C.**	**# N.C.**
Auto	7	398	2	UCI	2	1
Balance	4	625	3	UCI	2	2
Breast Cancer	9	286	2	UCI	3	0
Car	6	1728	3	UCI	6	0
Churn	19	5000	2	UCI	0	1
Contraceptive	8	1473	3	UCI	2	0
Era	4	1000	5	MLD	4	0
Esl	4	488	5	MLD	4	0
German Credit	24	1000	2	UCI	3	1
Haberman	3	306	2	UCI	2	1
Housing	13	506	5	UCI	0	1
Lev	4	1000	5	MLD	4	0
Pima	8	768	2	UCI	4	0
Swd	10	1000	4	MLD	10	0

Table 3.4: The characteristics of the 14 ordinal data sets included in the benchmarking study: ID, name, number of attributes, observations, and class labels, and the source of the data sets, which is either the UCI Machine Learning Repository (archive.ics.uci.edu/ml), or the MLD Machine Learning Data Set Repository (www.mldata.org). The last two columns indicate the number of positive (# P.C.) and negative constraints (# N.C.) that are imposed in the experiments.

in Table 3.4.

The monotonicity constraints that are imposed on the attributes in the *Auto*, *Breast Cancer Ljubljana*, *German Credit Scoring*, *Haberman*, and *Pima* data sets have been copied from Martens et al. (2006). The *Auto* data set concerns the prediction of car fuel consumption in gallons per mile (less or more than 28), based on eight car properties. Domain knowledge states that larger weight and displacement will lead to higher fuel consumption, where more recent models are presumed to be more fuel-efficient. In the *Breast Cancer Ljubljana* data set the recurrence of breast cancer needs to be predicted, where it is expected that an increase in tumor size, number of nodes involved, and the degree of malignancy will lead to higher probability of recurrence. The target variable to predict in the *Pima* data set is whether a person shows signs of diabetes. Increasing age, number of pregnancies, body mass index, and pedigree risk would suggest a higher chance

of being diabetic. In the *German Credit Scoring* data set, the classification problem consists of predicting clients as either good or bad (defaulted). Expert knowledge suggests bad customers will have less amount on checking and savings accounts, and have had more problems with their credit history. The *Haberman* data set concerns the prediction of the survival status of a patient that has undergone breast cancer surgery. Medical knowledge suggests a lower survival rate for patients that are older, have more detected positive axillary nodes, and whose operation was less recent.

The *ERA*, *ESL*, *LEV*, and *SVD* data sets have been described in Ben-David et al. (2009). All attributes in these data sets are positively related to the target variables, and total monotonicity of the classification model is demanded. The *Balance*, *Car*, *Churn*, *Contraceptive*, and *Housing* data sets have been gathered from the UCI Machine Learning Repository. The instances in the *Balance* data set are classified as having the balance scale tip to the left, tip to the right, or being balanced. The correct way to find the class is the greater of right-distance times right-weight and left-distance times left-weight. If they are equal, an instance is classified as balanced. Hence, a negative constraint is imposed on the left-distance and the left-weight, and a positive constraint on the right-distance and right-weight, with the ordering of the class variable left, balanced, and right. The target variable to predict in the *Car* evaluation database is the acceptability of a car. A higher buying and maintenance price is expected to have a negative impact on acceptability, while the number of doors and persons the car can carry, the size of the luggage boot, and the estimated safety of the car are expected to have a positive impact. The *Churn* data set concerns the prediction of customers that are about to churn. Typically, customers that have called the helpdesk are expected to have a lower probability to churn. The problem in the *Contraceptive* data set is to predict the current contraceptive method choice (no use, short-term methods, or long-term methods) of a woman based on her demographic and socio-economic characteristics. A positive constraint is imposed on the number of children a women has given birth, and on an attribute that indicates the standard of living (low to high). Finally, the class variable in the *Housing* data set is the median value of owner occupied homes in suburbs of Boston, which is expected to be smaller in suburbs where a large fraction of the population has a lower status. A full description of the data sets can be obtained from the respective source repositories as indicated in Table 3.4.

Removed variable	Comp. pairs	DgrMon
Auto		
- (original data)	388	0.7242
cylinders	389	0.7249
displacement	467	0.7709
horsepower	499	0.7756
weight	416	0.7404
acceleration	1332	0.7905
modelyear	3891	0.7931
origin	620	**0.6532**
Balance		
- (original data)	23401	1.0000
leftWeight	39549	**0.9747**
leftDistance	39496	**0.9752**
rightWeight	40297	**0.9711**
rightDistance	39673	**0.9737**
Breast Cancer Ljubljana		
- (original data)	1499	0.8499
age	3184	0.8725
menopause	2370	0.8532
tumor-size	2545	**0.7953**
inv-nodes	1781	**0.8265**
node-caps	1925	0.8618
deg-malig	2166	**0.8366**
breast1	2090	0.8598
breast-quad	2500	0.8716
irradiat	1854	0.8581
Car		
- (original data)	115741	0.9997
buying	182182	**0.9842**
maint	182970	**0.9867**
doors	182074	**0.9976**
persons	174954	**0.9586**
lugBoot	171607	**0.9929**
safety	171822	**0.9494**
Churn		
- (original data)	987	0.9139
Account Length	1967	0.9222
Area Code2 AC415	1326	0.9155
Area Code2 AC408	1488	0.9153
Intl Plan	1081	**0.8955**
Vmail Plan	987	0.9139
Vmail Message	1020	**0.9127**
Day Mins	987	0.9139
Day Calls	1910	0.9283
Day Charge	987	0.9139
Eve Mins	987	0.9139
Eve Calls	1734	0.9269

Removed variable	Comp. pairs	DgrMon
Churn (ctd.)		
Eve Charge	987	0.9139
Night Mins	988	0.9140
Night Calls	1924	**0.9111**
Night Charge	987	0.9139
Intl Mins	987	0.9139
Intl Calls	1849	0.9275
Intl Charge	987	0.9139
CustServ Calls	1613	0.9392
Contraceptive		
- (original data)	122062	0.7090
wifeAge	169636	0.7501
wifeEduc	153649	**0.6993**
husbEduc	135202	0.7136
nrChild	158936	**0.6729**
wifeRelig	152933	0.7096
wifeWork	145802	**0.7089**
standLivInd	149208	**0.7077**
mediaExp	131654	0.7131
Era		
- (original data)	70464	0.8310
In01	99511	**0.7720**
In02	124007	**0.7264**
In03	118216	**0.8122**
In04	107654	**0.8203**
Esl		
- (original data)	46164	0.9784
In01	51480	**0.9654**
In02	51077	**0.9629**
In03	51499	**0.9512**
In04	54826	**0.9322**
German Credit Scoring		
- (original data)	479	0.8622
In01	821	**0.8197**
In02	1162	**0.7823**
In03	571	0.8722
In04	1619	0.8863
In05	786	**0.8550**
In06	569	0.8664
In07	585	0.8718
In08	598	**0.8512**
In09	770	**0.8597**
In10	785	**0.8484**
In11	558	0.8728
In12	517	0.8627
In13	548	0.8741
In14	596	0.8758

Removed variable	Comparable pairs	DgrMon
German Credit Scoring (ctd.)		
In15	501	0.8683
In16	528	**0.8523**
In17	557	0.8797
In18	505	0.8653
In19	530	0.8698
In20	611	**0.8560**
In21	678	0.8820
In22	489	**0.8548**
In23	569	0.8717
In24	691	**0.8611**
Haberman		
- (original data)	14100	0.8735
age	14100	0.8735
year	27128	0.8826
nodes	23808	**0.8073**
Housing		
- (original data)	144	0.9306
CRIM	273	0.9377
ZN	148	**0.9257**
INDUS	192	**0.8854**
CHAS	146	0.9315
NOX	206	**0.9223**
RM	211	**0.8815**
AGE	291	**0.8969**
DIS	471	**0.8004**
RAD	150	0.9333
TAX	144	0.9306
PTRATIO	159	0.9371
B	273	**0.9048**
LSTAT	242	**0.8430**

Removed variable	Comp. pairs	DgrMon
Lev		
- (original data)	60874	0.9532
In01	106471	**0.9023**
In02	102700	**0.8284**
In03	102976	**0.9331**
In04	104705	**0.9325**
Pima		
- (original data)	11028	0.9763
pregnant	15282	**0.9704**
glucose	14537	**0.9548**
bloodpressure	14439	0.9768
skinthickness	13443	0.9763
serum	12776	0.9766
BMI	13327	**0.9589**
pedigree	17398	**0.9697**
age	12862	**0.9678**
Swd		
- (original data)	34379	0.9322
In01	38082	**0.9314**
In02	40757	**0.9267**
In03	36942	**0.9258**
In04	43945	**0.9308**
In05	42530	**0.9233**
In06	39715	0.9323
In07	40685	0.9325
In08	42753	0.9372
In09	38257	**0.9302**
In10	41168	0.9232

Table 3.5: Degree of monotonicity of the data sets in the experiments. A positive constraint is imposed on variables in bold, and a negative constraint is imposed on variables in bold and italic. Degrees of monotonicity in bold are smaller than the degree of monotonicity of the original data set. Degree and variable are underlined when conform, i.e., when a variable is constrained and the degree is smaller than the degree of the original data set.

To check the monotone effect of a variable, the degree of monotonicity, as introduced in Daniels and Velikova (2010) and discussed in Section 3.2, is compared in Table 3.5 for the original data and for the data with each

variable removed in turn. This allows to analyze whether domain knowledge as expressed by the imposed monotonicity constraints is in concordance with the data. To allow a straight comparison, negative constraints have been converted in positive constraints by multiplying a variable with minus one. A positive constraint is imposed on variables in bold, and a negative constraint is imposed on variables in bold and italic. Degrees of monotonicity in bold are smaller than the degree of monotonicity of the original data set. Degree and variable are underlined when conform, i.e., when a variable is constrained and the degree is smaller than the degree of the original data set. As can be seen, most constraints are confirmed by the DgrMon measure, except for the constraints in the *Auto* and *Churn* data set, and the constraints in the *German Credit Scoring* data set on variables In03 and In04, in the *Haberman* data set on variables age and year, and in the *SWD* data set on the variables In06, In07, In08, and In10.

3.7.2 Experimental setup

Four classification techniques will be applied on the selected data sets; Ripper (Cohen, 1995) and AntMiner+ (Martens et al., 2007b), two state-of-the-art rule induction techniques, C4.5 (Quinlan, 1993), a decision tree induction technique, and OLM (Sterling and Pao, 1989), which has been discussed in Table 3.2. The classifiers induced with Ripper, AntMiner+, and C4.5 will be postprocessed using RULEM to check whether the imposed constraints are respected, and if not, to resolve violations by inducing additional rules. OLM yields (totally) monotone classification models and has been included for comparison purposes. Both Ripper, C4.5, and OLM have been implemented in the Weka[4] environment (Witten and Frank, 2000). AntMiner+ on the other hand has been implemented in Matlab and can be downloaded freely from the web[5].

When the target variable is binary, AntMiner+ allows to impose monotonicity constraints directly during the induction of classification rules (Martens et al., 2006). This version of AntMiner+, denoted AntMiner+ DK (Domain Knowledge), will be applied in the benchmarking experiments on the data sets with a binary target variable. A similar feature has been developed and implemented for the Ripper algorithm, allowing to impose monotonicity constraints and to induce monotone rule sets for binary classification in

[4] www.cs.waikato.ac.nz/ml/weka
[5] www.antminerplus.com

a direct manner during the actual data mining process. This extended version of the Ripper algorithm will be denoted Ripper DK, and is described in Appendix A.

Each classification technique is applied to five random split ups of the data sets in 2/3 training and 1/3 test data. The discrimination power of the classification models will be reported as the percentage correctly classified instances, since the number of values of the target attribute varies over the data sets in a range from two to five. When the output of a classifier is a score or probability, the PCC measure depends on a threshold value(s). However, the applied classification techniques result in a model that assigns an explicit class label to each instance, and not a score or probability. Furthermore, PCC can be applied regardless of the number of values of the class attribute. Therefore, in this experimental setting the PCC serves well to compare the discrimination power of the original classification models resulting from the applied techniques and the RULEM postprocessed classifiers.

As mentioned in the introduction, the existing literature on predictive modeling or classification mainly focuses on the predictive power of classification models. However, it must be stressed that in a practical setting the comprehensibility and justifiability of classification models are often the determinant factors deciding upon the effective implementation and use of a model. Therefore, the number of rules in rule sets in the experiments will be reported as a measure of the comprehensibility of the induced models, as well as the average number of attribute tests in each rule. The justifiability of the original classification models on the other hand will be reported in terms of both the RULEMS and RULEMF measures, which were introduced and discussed in the previous section.

A procedure described in Demšar (2006) is followed to statistically test the results of the benchmarking experiments and contrast the accuracy, the number of induced rules, and the average number of terms per rule of the different techniques. To compare two techniques, the Wilcoxon signed-ranks test (Wilcoxon, 1945) is applied, which ranks the differences in performances for each data set, ignoring the signs, and compares the ranks for the positive and the negative differences. When comparing multiple techniques, in a first step the non-parametric Friedman test (Friedman, 1940) is performed to check whether differences in performance are due to chance. If the null hypothesis that no significant differences exist is rejected by the Friedman test, then the post-hoc Nemenyi test (Nemenyi, 1963) is performed to com-

pare the individual classifiers. These tests will be discussed into more detail in Section 4.5.3

3.7.3 Results

Table 3.6 summarizes the average results of the experiments over the five random holdout splits for the original and the postprocessed rule sets with RULEM. The top panel reports the mean accuracy, and the middle and bottom panel indicate the average size of the induced rule sets in terms of mean number of rules and mean number of conjuncts per rule in the resulting rule sets. Table 3.7 provides the mean RULEMS and RULEMF measures of the induced rule sets.

A maximum value of the CI-score equal to 50 was set in the experiments for RULEM to be executed and to induce additional rules to resolve violations of the imposed monotonicity constraints. When the initial CI-score is high, a large number of additional rules will likely be required to resolve the violations, and the resulting rule set becomes unacceptable for implementation. The maximum CI-score has been set to provide an unbiased indication of the performance of the RULEM technique, both in terms of predictive power and required number of additional rules that are induced to make a rule set monotone. It needs to be stressed that setting a minimum CI-score does not imply that RULEM is not able to induce a solution, but the quality of the induced solution is likely to be low. By setting a minimum CI-score we believe to provide a fair indication of the applicability of RULEM. In a real life setting it is the user which has to decide about the acceptability of a classification model, as well as on which constraints to impose.

The results in Table 3.6 that are marked with an asterisk are the average over less than five holdout splits. In case of RULEM, a dash indicates that for all five holdout splits an initial CI-score was obtained that was larger than 50. In case of Ripper DK and AntMiner+ DK, a dash indicates that the target variable in the data set is not binary, and consequently the classifiers could not be executed. Table 3.8 reports the number of times RULEM was not executed because the initial total CI-score was larger than 50.

The reason why the average number of induced rules by AntMiner+ with RULEM is smaller than without RULEM for the *Churn* data set is because the result of AntMiner+ with RULEM is averaged only over two holdout splits, of which one holdout split yields a number of rules that is small compared to the overall average.

	Ripper			**AntMiner+**			**C4.5**		**OLM**
	OR	RULEM	DK	OR	RULEM	DK	OR	RULEM	
				PCC (%)					
Auto	**89.33**	81.04	86.81	**89.32**	84.02*	84.66	**90.07**	82.47*	67.56
Balance	81.32	81.32	–	77.13	**77.89**	–	80.57	**81.23**	82.92
Breast Cancer	71.13	71.13	71.13	70.00	70.63	**72.50**	**71.13**	70.93	68.87
Car	94.80	**94.86***	–	93.92	–	–	96.67	**97.04**	93.91
Churn	**93.88**	93.68*	91.44	87.98	87.26*	**89.24**	92.76	–	84.76
Contraceptive	51.14	**51.98**	–	**41.75**	40.94*	–	52.61	–	44.07
Era	**45.06**	44.53	–	**39.34**	38.62*	–	**45.18**	44.71	33.53
Esl	70.84	70.84	–	**48.34**	44.17*	–	70.00	70.00	53.25
German Credit	**73.65**	71.65	72.18	69.82	68.86*	**70.24**	70.94	–	71.12
Haberman	**75.00**	**75.00**	73.46	71.37	**72.75**	**72.75**	72.69	72.69	74.23
Housing	71.74	71.74	–	**60.00**	50.30*	–	70.93	–	20.47
Lev	59.53	**60.41**	–	**47.31**	44.91	–	60.76	**60.88**	43.24
Pima	**74.94**	**74.94**	74.41	71.02	–	**71.33**	73.95	**74.71***	69.50
Swd	**56.24**	56.18*	–	44.79	**45.81***	–	**55.47**	55.29*	41.35
				# Rules					
Auto	4.4	13.6	3.2	4.0	11.5*	4.4	10.4	30.3*	56.8
Balance	9.0	9.0	–	10.2	20.6	–	33.0	34.8	74.2
Breast Cancer	2.0	2.0	2.0	3.2	10.6	4.2	8.8	14.0	52.6
Car	17.4	39.3*	–	22.8	–	–	44.8	61.2	39.8
Churn	8.0	20.8*	5.6	8.0	4.7*	6.0	30.6	–	1088.0
Contraceptive	4.0	8.4	–	6.4	9.0*	–	94.8	–	58.8
Era	4.4	6.0	–	6.6	12.0*	–	10.6	18.4	13.8
Esl	10.4	16.8	–	13.2	40.0*	–	22.0	33.6	13.2
German Credit	4.4	17.4	2.8	3.6	10.8*	4.2	37.2	–	196.6
Haberman	2.0	2.0	2.0	4.0	12.2	4.4	3.8	4.0	5.0
Housing	7.8	7.8	–	9.2	25.0*	–	29.8	–	294.8
Lev	9.6	26.6	–	8.0	19.8	–	24.4	46.2	31.6
Pima	3.4	3.4	3.2	8.0	–	7.6	16.2	36.3*	110.4
Swd	8.4	17.7*	–	7.6	33.0*	–	17.2	60.5*	47.2
				# Tests/Rule					
Auto	2.1	3.3	1.5	2.6	4.5*	2.0	4.6	3.3*	7.0
Balance	2.8	2.8	–	3.3	3.6	–	3.6	3.3	4.0
Breast Cancer	1.8	1.8	1.8	3.5	4.9	3.8	4.0	3.4	9.0
Car	4.7	5.2*	–	5.0	–	–	6.0	3.9	6.0
Churn	2.6	5.7*	2.6	3.6	3.7*	3.2	10.7	–	19.0
Contraceptive	3.1	3.6	–	4.8	5.6*	–	5.5	–	8.0
Era	2.9	3.1	–	3.6	3.9*	–	2.7	2.8	4.0
Esl	1.7	1.9	–	3.3	3.9*	–	3.1	3.2	4.0
German Credit	2.9	4.0	2.9	3.5	3.5*	5.0	8.1	–	19.0
Haberman	1.8	1.8	1.6	2.8	2.9	2.8	3.0	1.3	3.0
Housing	1.9	1.9	–	6.0	11.6*	–	6.2	–	13.0
Lev	2.9	3.4	–	3.5	3.8	–	3.2	3.5	4.0
Pima	2.0	2.0	2.0	4.0	–	3.7	6.6	4.1*	8.0
Swd	2.9	4.7*	–	5.5	7.3*	–	9.2	5.6*	10.0

Table 3.6: Average results of the experiments over five hold out splits.

	Ripper		**AntMiner+**		**C4.5**	
	RULEMF	RULEMS	RULEMF	RULEMS	RULEMF	RULEMS
Auto	34.44	40.10	91.66	65.64	25.57	32.66
Balance	100.00	100.00	99.37	97.25	99.53	98.98
Breast Cancer	100.00	100.00	100.00	91.75	81.92	97.69
Car	100.00	97.87	–	–	96.85	97.40
Churn	92.48	98.34	100.00	100.00	–	–
Contraceptive	84.56	90.45	100.00	99.32	–	–
Era	99.69	97.64	100.00	77.10	84.84	84.94
Esl	94.06	96.23	100.00	67.42	87.66	94.26
German Credit	94.33	65.76	74.98	50.15	–	–
Haberman	100.00	100.00	72.51	72.55	95.52	99.80
Housing	100.00	100.00	100.00	96.25	–	–
Lev	97.73	93.78	100.00	68.88	81.17	84.06
Pima	100.00	100.00	–	–	97.71	94.62
Swd	100.00	97.77	100.00	82.80	81.25	82.10

Table 3.7: Justifiability measures of the original rule sets without monotonicity constraints induced by Ripper, AntMiner+, and C4.5 classifiers.

Name	**Ripper**	**AntMiner+**	**C4.5**
Auto	0	1	2
Balance	0	0	0
Breast Cancer	0	0	0
Car	1	**5**	0
Churn	1	2	**5**
Contraceptive	0	3	**5**
Era	0	1	0
Esl	0	4	0
German Credit	0	1	**5**
Haberman	0	0	0
Housing	0	4	**5**
Lev	0	0	0
Pima	0	**5**	2
Swd	2	4	3

Table 3.8: The number of hold out splits resulting in a CI-score of the rule set or decision tree above the threshold value. These runs are not included in calculating the values in Table 3.6.

As can be seen from Table 3.6, in general, the performance of the RULEM postprocessed rule sets is either slightly worse or slightly better than the performance of the original rule set. RULEM yields the best classifier in 11 cases, and the original classifier in 17 cases. In eight cases the monotonicity is respected by the original rule set for all data points in the test set, resulting in equal performance. The Wilcoxon signed-ranks test to compare the predictive power of RULEM vs. the original rule set was not able to reject the null hypothesis that RULEM does not affect the predictive power of a rule set in a statistically significant manner ($p \cong 0.14$). When comparing the RULEM postprocessed rule sets to the rule sets induced using Ripper DK and AntMiner+ DK on the data sets with a binary target variable, the same conclusion holds; the predictive power in terms of PCC of the RULEM postprocessed rule sets is found not to be significantly different from the predictive power of the rule sets obtained using the DK versions ($p \cong 0.23$).

On the other hand, application of the Friedman test and the post hoc Nemenyi test indicates that the combination of Ripper and RULEM outperforms the OLM classifier at the 95% significance level, while the performance of both AntMiner+ and C4.5 in combination with RULEM does not significantly differ from either Ripper in combination with RULEM and OLM. The size of the induced rule sets, both in terms of number of rules ($p < 0.01$) and average number of attribute tests per rule ($p = 0.075$) appears to be significantly larger when applying the RULEM technique. Since RULEM induces additional rules to resolve monotonicity constraints, this is not surprising either. Compared to the OLM classifier, RULEM induces significantly smaller rule sets than the OLM classifier ($p < 0.01$). In general, the comprehensibility of the RULEM postprocessed rule sets appears to be maintained, given the overall small total number of rules. Converting a C4.5 decision tree into a rule set generally appears to yield a quite large number of rules. Consequently, RULEM also requires a relatively large number of rules to resolve violations and enforce monotonicity.

Table 3.7 reports the justifiability of the induced rule sets both in terms of the RULEMS and RULEMF measure, as defined in the previous section. For each data set in the experiment either Ripper or AntMiner+, and in most cases even both, results in a rule set that does not respect the imposed monotonicity constraints. C4.5 yields a non-monotone classification model for each data set. This result indicates that in general rule and tree induction

techniques do not yield monotone classification models, and that there is a need for techniques which are able to enforce monotone relations. Remark in Table 3.7 that the RULEMF measure may be equal to one while the RULEMS measure is not. When no instances in the data set are situated in the part of the attribute space where classification does not respect the imposed constraints, the RULEMF measure will be equal to one, while the RULEMS measure clearly will not.

The results of the experiments indicate that the RULEM technique outperforms the OLM technique, both in terms of predictive power, number of rules, and average number of tests per rule. RULEM does not outperform the DK versions of the classifiers, however, the DK versions can only be applied with a binary target variable, and not every rule or tree induction technique has a DK version in place, allowing to impose monotonicity constraints in a direct manner. This exactly illustrates the main strength of the RULEM approach, which lies in the fact that RULEM can be combined with any rule or tree induction technique, and does not require adjustments to the workings of these techniques. As shown by the experiments, RULEM preserves the predictive power of the original classification techniques. As such, the resulting predictive power mainly depends on the performance of the base rule induction technique RULEM is combined with, which opens opportunities to further improve predictive power, for instance by applying active learning strategies (Martens et al., 2009).

When the impact of RULEM on a classification model is large, both in terms of predictive accuracy and number of rules, the constraints that are imposed on the classification model might need to be reconsidered, since they appear not to be in line with the data. The domain knowledge or intuitive relations that result in the imposed constraints always has to be questioned before effectively imposing constraints on the resulting model.

Figure 3.5 plots the number of additional rules (# AR) and difference in percentage correctly classified between the RULEM postprocessed and the original rule set (Δ PCC), as a function of the justifiability of a rule set measured using the RULEMS and RULEMF measure. The figures also plot simple linear regression models, fitting the number of additional rules and the difference in accuracy as a function of the justifiability. The inclination of these linear regression models indicates that the number of additional rules decreases as a function of the justifiability ($R^2 = 0.19$ and $R^2 = 0.33$ for the linear regression models fitting the number of additional rules as a

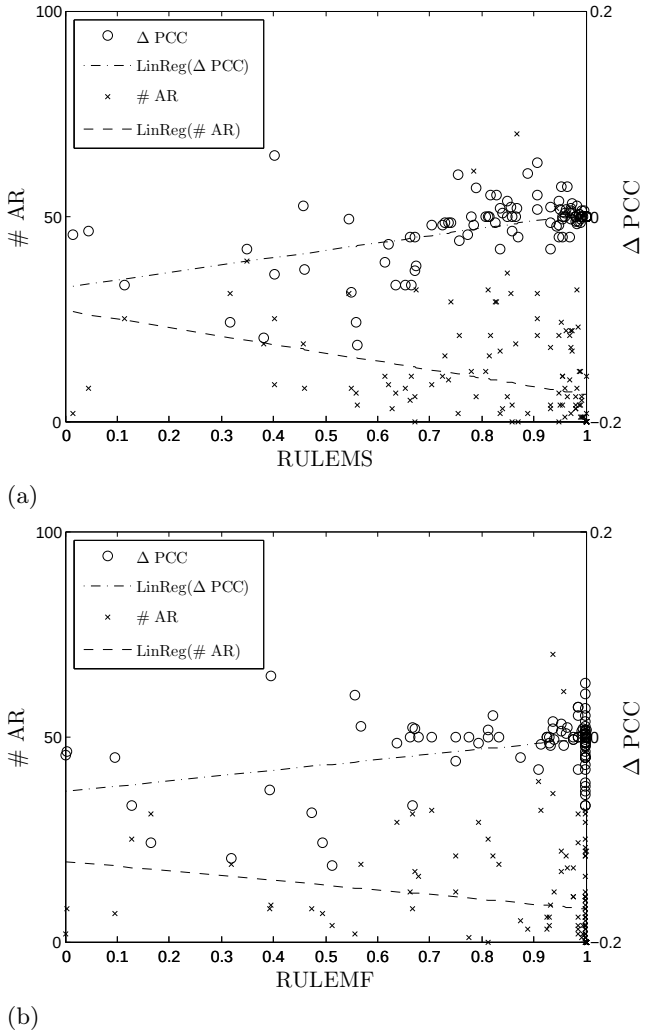


Figure 3.5: Number of additional rules (# AR) and difference in percentage correctly classified between the RULEM postprocessed and the original rule set (Δ PCC), as a function of the justifiability of a rule set measured using the RULEMS (upper panel) and RULEMF measure (lower panel), and simple linear regression models fitting # AR and Δ PCC as a function of RULEMS and RULEMF.

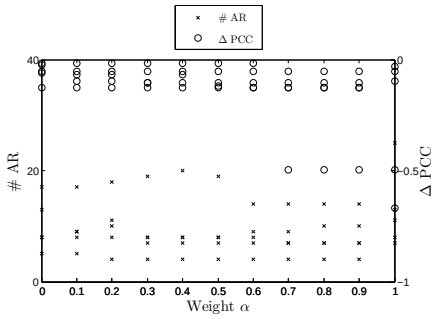


Figure 3.6: The number of additional rules (# AR) and the difference in accuracy (Δ PCC) as a function of the weight parameter α in the CI-score ($CI = \alpha C + (1 - \alpha)I$), for data set Auto, rule induction technique Ripper, and α ranging between zero and one.

function of respectively RULEMS and RULEMF), and that the difference in accuracy between the original model and the RULEM postprocessed rule set decreases as well ($R^2 = 0.41$ and $R^2 = 0.54$ for the linear regression models fitting ΔPCC as a function of respectively RULEMS and RULEMF). The plots in Figure 3.5 indicate that the stronger a rule set violates the imposed monotonicity constraints, the lower the predictive power of the RULEM postprocessed rule set will be, and the more additional rules will be required to resolve the violations.

Figure 3.6 plots the number of additional rules and the difference in accuracy between the RULEM postprocessed rule set and the original rule set as a function of the weight parameter α in the CI-score, with $CI = \alpha C + (1 - \alpha)I$, for each of the five holdout splits of the *Auto* data set, and with the original rule sets induced by Ripper. The weight α was varied between zero and one. The figure indicates that for a smaller value of the weight RULEM in general results in a more accurate rule set and a larger number of additional rules, while for a larger value of α less additional rules are induced at the cost of a decrease in predictive power compared to the original rule set. As explained in Section 3.5.2, the I-score was designed to preserve the predictive power of the original rule set, while the C-score aims at inducing a small number of additional rules. The results of Figure 3.6 confirm that by varying the weight in the CI-score, RULEM effectively allows to a certain extent to express a preference towards a smaller rule set

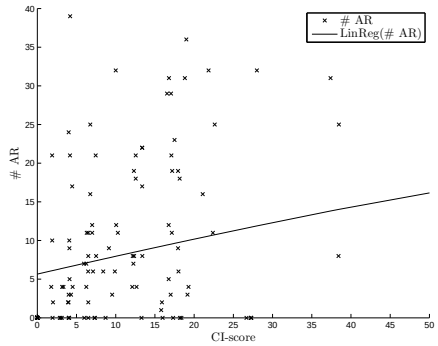


Figure 3.7: The number of additional rules (# AR) as a function of the CI-score, and a third order polynomial fitting the data points.

or towards a rule set with better predictive power. In the benchmarking experiment, the weight was set to 0.5, which incorporates a trade-off between both comprehensibility and accuracy.

Finally, Figure 3.7 plots the number of additional rules induced by RULEM as a function of the initial CI-score of the original rule sets. The figure also plots a third order[6] polynomial fitting the number of additional rules as a function of the initial CI-score, with $R^2 = 0.32$. The inclination of the polynomial clearly indicates that the number of additional rules is a positive function of the initial CI-score, and therefore justifies the maximum threshold value of the CI-score that was set in the experiments.

3.8 Conclusions and future research

Many real world applications require classification models to be in line with domain knowledge and to satisfy monotone relations between predictor variables and the target class, in order to be acceptable for implementation. However, existing techniques to induce monotone classification models either yield poor classification performance, or are only able to handle a binary class variable. Therefore, this chapter presents the novel RULEM algorithm to induce monotone ordinal rule based classification models. A main asset of the proposed approach is its complementarity, which allows the RULEM

[6] A first and second order polynomial yield the same conclusion, but result in a value of $R^2 < 0.20$.

approach to be combined with any rule- or tree-based classification technique, since monotonicity is guaranteed during a postprocessing step. The RULEM algorithm checks whether a rule set or decision tree violates the imposed monotonicity constraints, and existing violations are resolved by inducing a set of additional rules which enforce monotone classification. The algorithm is able to handle non-monotonic noise, and can be applied to both partially and totally monotone problems with an ordinal target variable.

Based on the RULEM algorithm, two novel justifiability measures are introduced. The RULEMS and RULEMF measures allow to calculate the extent to which a classification model is in line with domain knowledge expressed in the form of monotonicity constraints. Both measures provide an intuitive indication of the justifiability of a rule set, and can be calculated in a fully automated manner.

An extensive benchmarking experiment has been set up to test the impact of the RULEM approach on the predictive power and the comprehensibility of the resulting rule set. The results of the experiments indicate that the proposed approach preserves the predictive power of the original rule induction techniques while guaranteeing monotone classification, at the cost of a small increase in the size of the rule set. Hence, the RULEM algorithm is shown to yield accurate, comprehensible, and justifiable rule based classification models. The predictive power of the final rule set therefore depends on the selected rule induction technique that RULEM is combined with.

An important topic for future research is the further development and refinement of the RULEM algorithm, and more precisely the improvement of the heuristic approach to merge the induced additional rules. This will allow to further reduce the size of the final rule set. Moreover, further analysis and experiments are needed to examine the exact nature of the relation between the C(I)-score, the justifiability, and the induced number of additional rules. Finally, an interesting and challenging topic for future research will be the development of a justifiability measure for non-rule based classification models.

Chapter 4

New insights into churn prediction in the telecommunication sector: a profit driven data mining approach

Information is not knowledge.

Albert Einstein (1879 - 1955)

Abstract

Customer churn prediction models aim to indicate the customers with the highest propensity to attrite, allowing to improve the efficiency of customer retention campaigns, and to reduce the costs associated with churn. Although cost reduction is the prime objective of such customer retention campaigns, customer churn prediction models are typically evaluated using statistically based performance measures, which may lead to suboptimal

model selection from a profit point of view. Therefore, in the first part of this chapter, a novel, profit centric performance measure is developed, by calculating the maximum profit that can be generated by including the optimal fraction of customers with the highest predicted probabilities to attrite in a retention campaign[1]. The novel measure selects the optimal model and fraction of customers to include and yields a significant increase in profits compared to statistical measures.

In the second part of this chapter an extensive benchmarking experiment is conducted, evaluating a wide range of classification techniques on eleven real-life data sets from telco operators worldwide using both the profit centric and statistically based performance measures. The experimental results show that a small number of variables suffices to predict churn with high accuracy, and that oversampling generally does not improve the performance significantly. Finally, a large group of classifiers is found to yield comparable performance.

4.1 Introduction

In this chapter, we examine the performance of various state-of-the-art data mining classification algorithms, by applying them on eleven real-life churn prediction data sets from wireless telco operators around the world. Techniques that are implemented comprise rule based classifiers (Ripper, PART), decision tree approaches (C4.5, CART, Alternating Decision Trees), neural networks (Multilayer Perceptron, Radial Basis Function Network), nearest neighbor (kNN), ensemble methods (Random Forests, Logistic Model Tree, Bagging, Boosting), and classic statistical methods (logistic regression, Naive Bayes, Bayesian Networks). Furthermore, the power and usefulness of the support vector machine and the least squares support vector machine (LSSVM) classifiers have not yet been thoroughly investigated in the context of customer churn prediction, and are therefore applied using both linear and radial basis function kernels. Finally, also data preprocessing techniques such as variable selection and oversampling may have a significant impact on the final performance of the model, and will therefore be tested in the benchmarking experiments.

[1] Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B., 2011a. New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. European Journal of Operational Research, doi 10.1016/j.ejor.2011.09.031

In the literature, the performance of a classification model is usually evaluated in terms of the *area under the receiver operating curve* (AUC), which basically represents the behavior of a classifier without regard to class distribution or misclassification costs. However, since only a small fraction of the customers can be included in a retention campaign, a CCP model is typically evaluated using top decile lift instead of AUC, which only takes into account the performance of the model regarding the top 10% of customers with the highest predicted probabilities to attrite. However, as indicated in Section 4.3 and demonstrated by the results of the benchmarking experiment in Section 4.6, from a profit centric point of view using the top decile lift (or the lift at any other cut-off fraction for that matter) results in a suboptimal model selection. Therefore, a novel profit centric performance measure is introduced, i.e., the maximum profit criterion, which calculates the profit generated by a model when including the optimal fraction of top-ranked customers in a retention campaign. The results of the benchmarking study will be evaluated using both *statistical* performance measures, such as AUC and top decile lift, as well as the newly developed *profit centric* performance measure, which allows to compare both approaches and demonstrate the merits of the newly proposed criterion. Finally, all the experimental results will be rigorously tested using the appropriate test statistics, following a procedure described by Demšar (2006).

The main contributions of this chapter lie in the development of a novel, profit centric approach to (1) evaluate and (2) deploy a CCP model, by (a) calculating the maximum profit that can be generated using the predictions of the model and by (b) including the optimal fraction of customers in a retention campaign. The results of an extensive benchmarking experiment show that both optimizing the included fraction of customers and applying the maximum profit criterion to select a classification model yield significant cost savings. Finally, a number of key recommendations are formulated based on the experimental results regarding the technical and managerial side of the CCP modeling process.

The remainder of this chapter is structured as follows. The next section provides a brief introduction to CCP modeling. Then, in Section 4.3 the maximum profit criterion to evaluate CCP models is developed, based on a formula to calculate the profits generated by a retention campaign introduced by Neslin et al. (2006). Next, Section 4.4 defines the experimental design of the benchmarking experiment, and provides an overview

of the state-of-the-art classification techniques that are included in the experiment. Also input selection and oversampling for churn prediction are discussed. Section 4.5 describes the procedure to test the results of the experiments in a statistically sound and appropriate way. Also a brief review is provided of the most commonly applied *statistical* (as opposed to profit centric) performance measures. Section 4.6 then presents the empirical findings of the experiments, and compares the results of the maximum profit and statistical performance measures. Finally, the last section concludes the chapter with a number of managerial and technical recommendations regarding CCP modeling, and identifies some interesting issues for future research.

4.2 Customer churn prediction modeling

Customer churn prediction is a management science problem for which typically a data mining approach is adopted. Data mining entails the overall process of extracting knowledge from data. Based on historical data a model can be trained to classify customers as future churners or non-churners. Numerous classification techniques have been adopted for churn prediction, including traditional statistical methods such as logistic regression (Lemmens and Croux, 2006; Neslin et al., 2006; Burez and Van den Poel, 2009), non-parametric statistical models like for instance k-nearest neighbor (Datta et al., 2000), decision trees (Wei and Chiu, 2002; Lima et al., 2009), and neural networks (Au et al., 2003; Hung et al., 2006). Often conflicts arise when comparing the conclusions of some of these studies. For instance, Mozer et al. (2000) found that neural networks performed significantly better than logistic regression for predicting customer attrition, whereas Hwang et al. (2004) reported that the latter outperforms the former. Furthermore, most of these studies only evaluate a limited number of classification techniques on a single churn prediction data set. Therefore the issue of which classification technique to use for churn prediction remains an open research issue, in which the benchmarking experiment described in this chapter aims to provide further insights. For an extensive literature review on CCP modeling one may refer to Verbeke et al. (2011e).

Figure 4.1 represents the process of developing a CCP model. The first step in this process consists of gathering relevant data and selecting candidate explanatory variables. The resulting data set is then cleaned and

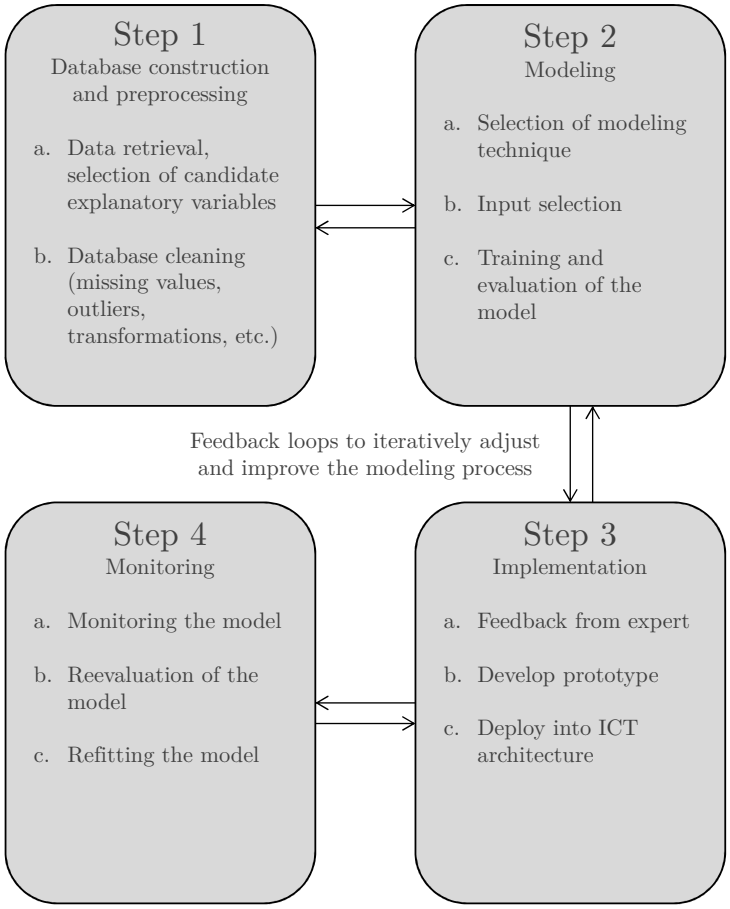


Figure 4.1: Data mining process of building a customer churn prediction model.

preprocessed. The second step encompasses the actual building of a model. A modeling technique is selected based on the requirements of the model and the type of data. Input selection is often applied to reduce the number of variables in order to get a consistent, unbiased, and relevant set of explanatory variables. Depending on the number of observations, which can be small in case of new products, a model is trained by cross validation or by splitting the data set in a separate training and test set. The resulting model is then evaluated, typically by comparing the true values of the target

variable with the predicted values, but also, if possible, by interpreting the selected variables and the modeled relation with the target variable. A variety of performance measures to evaluate a classification model have been proposed in the literature, as will be discussed in Sections 4.3 and 4.5. In a third step the model is assessed by a business expert to check whether the model is intuitively correct and in line with business knowledge. A prototype of the model is then developed, and deployed in the ICT architecture. The final step, once a model is implemented that performs satisfactory, consists of regularly reviewing the model in order to assess whether it still performs well. Surely in a highly technological and volatile environment as the telco sector, a continuous evaluation on newly gathered data is of crucial importance. At the end of each phase the results are evaluated, and if not satisfactory one returns to a previous step in order to adjust the process.

4.3 The maximum profit criterion

This section provides an extensive overview of the existing approaches to evaluate a CCP model from a profit centric point of view. These approaches form a business oriented alternative to the current standard practice of adopting a statistically based performance measure. We prove that the existing approaches are essentially identical, and select the profit formula proposed by Neslin et al. (2006) to calculate the profit generated by a retention campaign as a function of the fraction of customers with the highest probability to attrite that are included in the campaign. Optimizing this fraction, i.e., determining the optimal threshold probability to attrite for a customer to be included in the retention campaign, results in the maximum profit that can be generated by using the output of a CCP model. The maximum profit can then be used as a profit centric performance measure to evaluate and compare the performance of different CCP models.

4.3.1 Business oriented evaluation approaches

In the literature only a limited number of studies have developed a profit centric approach to evaluate the performance of prediction models in a customer relationship management setting, and more specifically the performance of CCP models. This section provides a summary and a short review of these approaches, and proofs that they are in fact equivalent.

Profitability of a churn management campaign

Figure 4.2 schematically represents the dynamical process of customer churn and retention within a customer base. New customers flow into the customer base by subscribing to a service of an operator, and existing customers flow out of the customer base by churning. When setting up a churn management campaign, a fraction of the customer base is identified correctly by the implemented CCP model as would-be churners, and offered an incentive to stay. A fraction γ of these customers accepts the offer and is retained, but the remaining fraction $(1-\gamma)$ is not and effectively churns. On the other hand, a fraction of the customer base is incorrectly classified as would-be churners, and also offered an incentive to stay. All of these customers are assumed to accept the offer and none of them will churn. Finally, a fraction of the would-be churners in the customer base is not identified as such, and thus are not offered an incentive to stay. Hence, all of these customers will effectively churn, and together with the correctly identified would-be churners that are not retained constitute the outflow of the customer base.

Given this dynamical process of customer churn and retention, the profit of a single churn management campaign can be expressed as (Neslin et al., 2006):

$$\Pi = N\alpha\left[\beta\gamma(CLV - c - \delta) + \beta(1-\gamma)(-c) + (1-\beta)(-c-\delta)\right] - A \quad (4.1)$$

with Π the profit generated by a single customer retention campaign, N the number of customers in the customer base, α the fraction of the customer base that is targeted in the retention campaign and offered an incentive to stay, β the fraction true would-be churners of the customers included in the retention campaign, δ the cost of the incentive to the firm when a customer accepts the offer and stays, γ the fraction of the targeted would-be churners who decide to remain because of the incentive (i.e., the success rate of the incentive), c the cost of contacting a customer to offer him or her the incentive, CLV the customer lifetime value (i.e., the net present value to the firm if the customer is retained), and A the fixed administrative costs of running the churn management program.

The factor $N\alpha$ in Formula 4.1 reflects that the costs and profits of a retention campaign are solely related to the customers that are included in the campaign, except for the fixed administrative cost A which reduces the overall profitability of a retention campaign. The term $\beta\gamma(CLV - c - \delta)$ represents the profits generated by the campaign, i.e., the reduction in lost

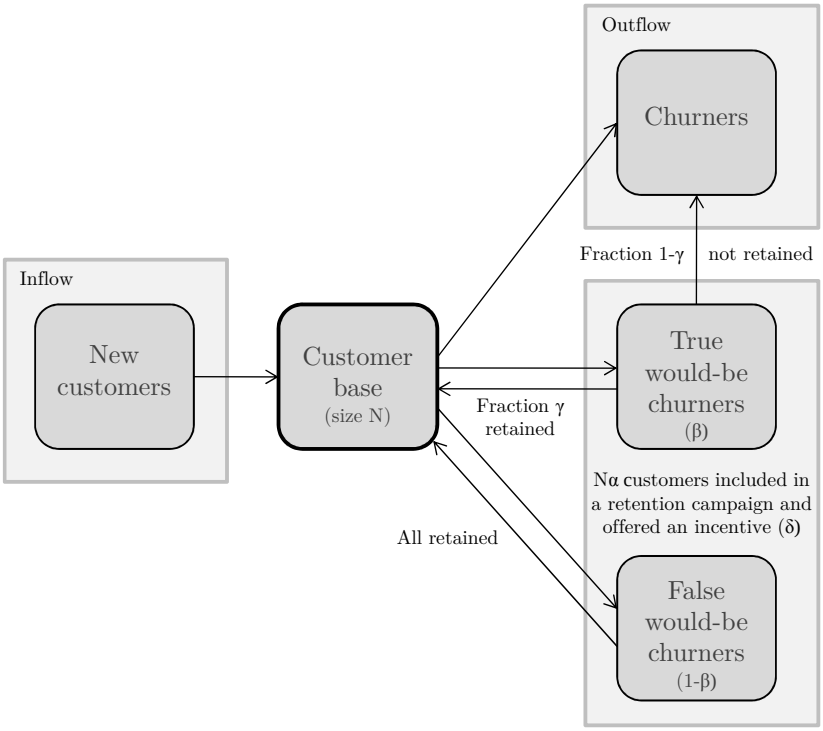


Figure 4.2: Schematic representation of customer churn and retention dynamics within a customer base.

incomes reduced with the costs of the campaign $(CLV - c - \delta)$ due to retaining a fraction γ of the would-be churners of the fraction correctly identified would-be churners β that are included in the campaign. The costs of the campaign are reflected by the term $\beta(1-\gamma)(-c)$, i.e., the cost of including correctly identified would-be churners that are not retained, and by the term $(1-\beta)(-c-\delta)$, which represents the cost due to including non-churners in the campaign, which are all logically expected to take advantage of the advantageous incentive offered to them in the retention campaign.

The term β reflects the ability of the predictive model to identify would-be churners, and can be expressed as:

$$\beta = \lambda\beta_0 \tag{4.2}$$

with β_0 the fraction of all the operator's customers that will churn, and

λ the lift, i.e., how much more the fraction of customers included in the retention campaign is likely to churn than all the operator's customers. The lift indicates the predictive power of a classifier, and is a function of the included fraction of customers α with the highest probabilities to attrite, as indicated by the model. Lift can be calculated as the percentage of churners within the fraction α of customers, divided by β_0. Thus, $\lambda = 1$ means that the model provides essentially no predictive power because the targeted customers are no more likely to churn than the population as a whole. Substituting Equation 4.2 in Equation 4.1, and rearranging the terms, results in:

$$\Pi = N\alpha\{[\gamma CLV + \delta(1-\gamma)]\,\beta_0\lambda - \delta - c\} - A \tag{4.3}$$

The incremental gain in profit from a unit increase in predictive accuracy λ equals the slope of this equation:

$$GAIN = N\alpha\{[\gamma CLV + \delta(1-\gamma)]\,\beta_0\} \tag{4.4}$$

From this formula can be seen that the gain in profit from improved accuracy increases when:

- the size of the campaign is larger ($N\alpha$)
- the potential recaptured CLV is higher
- the campaign's success rate (γ) is higher
- the incentive cost (δ) is higher (because more incentive money will be wasted if accuracy is poor)
- the base churn rate (β_0) is higher

According to (Neslin et al., 2006), the direct link between lift and profitability in this equation demonstrates the relevance of using (top decile) lift as a performance criterion in predictive modeling. Lift is also indicated to be the most commonly used prediction criterion in predictive modeling by Neslin et al. (2006), which is confirmed by an extensive literature study on CCP modeling provided in Chapter 3. However, as will be shown in the next section, using lift as a performance measure may lead to suboptimal model selection and a loss in profit, since the lift of a model is a function of the fraction α of customers that is included in the retention campaign. The

approach proposed by Neslin et al. (2006), has also been applied in Burez and Van den Poel (2007) to calculate the profit contribution of a retention campaign in a pay-TV setting.

The optimal decision-making policy

Mozer et al. (2000) is to our knowledge the first to propose the idea of using a maximum profit performance measure to evaluate and compare CCP models, although the idea is not elaborated nor is such a measure effectively applied. The selection of the optimal threshold churn probability for a customer to be included in a retention campaign, that maximizes the expected cost savings to the carrier, is referred to as the *optimal decision-making policy*. Cost savings are indicated to be dependent on:

- the discriminative ability of the CCP model
- the cost to the carrier of providing the incentive, C_i
- the time horizon over which the incentive has an effect on the subscriber's behavior
- the reduction in probability that the subscriber will leave within the time horizon as a result of the incentive, P_i
- the lost revenue cost that results when a subscriber churns, C_l

Subsequently, a time horizon of six months is adopted, and the average subscriber bill over this time horizon is used as the lost revenue due to churn, i.e., the cost of churn. The performance of the CCP model is characterized by four statistics:

- $N(pL, aL)$: number of subscribers who are predicted to leave (churn) and who actually leave, barring intervention
- $N(pS, aL)$: number of subscribers who are predicted to stay (non-churn) and who actually leave, barring intervention
- $N(pL, aS)$: number of subscribers who are predicted to leave and who actually stay
- $N(pS, aS)$: number of subscribers who are predicted to stay and who actually stay

Then the cost to an operator of performing no intervention equals:

$$\text{net(no intervention)} = [N(pL, aL) + N(pS, aL)]C_l \tag{4.5}$$

The cost of providing an incentive to all subscribers whom are predicted to churn equals:

$$\begin{aligned}\text{net(incentive)} =&[N(pL, aL) + N(pL, aS)]C_i + \\ &[P_i N(pL, aL) + N(pS, aL)]C_l\end{aligned} \tag{4.6}$$

Finally, the cost savings to the operator as a result of offering incentives based on churn prediction are expressed as:

$$\text{Total savings} = \text{net(no intervention)} - \text{net(incentive)} \tag{4.7}$$

Elaborating this equation results in:

$$\begin{aligned}\text{Total savings} =&[N(pL, aL) + N(pS, aL)]C_l - [N(pL, aL) + N(pL, aS)]C_i \\ &- [P_i N(pL, aL) + N(pS, aL)]C_l\end{aligned} \tag{4.8}$$

$$\text{Total savings} = N(pL, aL)(C_l - C_i - P_i C_l) + N(pL, aS)C_i \tag{4.9}$$

Equation 4.7 relates to Equation 4.3 introduced by Neslin et al. (2006) in the following way. Total savings is equivalent to profit Π. $N(pL, aL)$ is the number of correctly predicted churners included in the campaign, and using the notation introduced by Neslin et al. (2006), equal to $N\alpha\lambda\beta_0$. Furthermore, $N(pL, aS)$ refers to the number of non-churners included in the campaign, i.e., $N\alpha - N\alpha\lambda\beta_0$. The cost to the carrier of providing the incentive C_i corresponds to the cost c of contacting a customer to offer an incentive to stay. This cost is different from the cost of the actual incentive (i.e., δ), which should only be taken into account for the customers that effectively respond to the offer, i.e., the contacted would-be churners that decide to stay, and the contacted non-churners that take profit from the offer. Mozer et al. (2000) however does not make this difference in types of cost, and in Equation 4.6 the cost C_i is accounted for all the customers that are contacted. Therefore C_i corresponds to c, and not to δ. Finally, the lost revenue C_l equals CLV, and the reduced (and not *the reduction in* as denoted in Mozer et al. (2000)) probability to churn P_i corresponds to $(1 -$

γ). Then Equation 4.9 can be reformulated using the notation introduced in Section 4.3.1:

$$\Pi = N\alpha\lambda\beta_0(CLV - c - (1-\gamma)CLV) - N\alpha c + N\alpha\lambda\beta_0 c \tag{4.10}$$

$$\Pi = N\alpha(\gamma CLV\lambda\beta_0 - c) \tag{4.11}$$

which is identical to Equation 4.3 with $\delta = 0$ and $A = 0$. Hence, δ and A appear not to be taken into account in the expression proposed by Mozer et al. (2000). On the other hand, Mozer et al. (2000) mentions an indirect cost of customer churn that is not explicitly taken into account in Equation 4.3 (but in fact, neither in Equation 4.7), i.e., the cost of acquiring a new customer to replace a lost customer. This cost can however be taken implicitly into account in Equation 4.3 by augmenting the customer lifetime value CLV of an existing customer with the cost needed to replace a customer.

Value based training versus post-processing

Masand and Piatetsky-Shapiro (1996) and Piatetsky-Shapiro and Masand (1999) present a methodology for initial cost/benefit analysis of a direct marketing campaign (e.g., to reduce customer churn) based on a data mining model, and a formula for estimating the entire lift curve and the expected profits. The following key parameters of a targeted marketing campaign are identified:

- N = the total number of customers
- T = the fraction of target customers who have the desired behavior
- B = benefit of an accepted offer A by a customer correctly identified as a target
- C = cost of making an offer A to a customer, whether a target or not.

The profit of a campaign depends on the fraction P of all customers included in a campaign, with $0 < P < 1$. If a campaign targets all customers (i.e., $P = 1$), then the total profit equals:

$$\text{profit}(P = 1) = NTB - NC = N(TB - C) \tag{4.12}$$

Whether the profit is positive or not depends on the benefit/cost ratio, i.e., whether $TB/C > 1$. The promise of a data mining approach is that it can find a subset P_2 of size NP_2 of the population where the fraction of targets T_2 is substantially higher than in the overall population, so that making an offer to P_2 is more profitable than making an offer to all. The target frequency of the subset T_2 over the target frequency in the entire population is the lift of the data mining model, defined as $\text{lift}(P_2) = T_2/T$. Then the profit of a campaign including the fraction of the population indicated by the data mining approach can be calculated as:

$$\text{profit}(P_2, T_2) = NP_2(T\text{Lift}(P_2)B - C) \tag{4.13}$$

which is profitable if $\text{lift}(P_2) > C/BT$, defined as the campaign profitability condition.

Equation 4.13 can be reformulated using the notation of Neslin et al. (2006), with $\text{profit}(P_2, T_2)$ equal to Π, $N = N$, $P_2 = \alpha$, the fraction of target customers with the *desired* behavior T equal to the fraction of churners β, or expressed as a function of the lift curve $\lambda\beta_0$, and B equal to γCLV and not CLV, since only a fraction of the included customers with the *desired* behavior, i.e., the fraction of correctly identified churners that respond to the offer, will generate a benefit. Similar to Mozer et al. (2000), only the cost of including a customer in a campaign is taken into account by Piatetsky-Shapiro and Masand (1999), and not the cost of the accepted offer δ itself, and thus $C = c$. Using this notation, and after rearranging the terms, Equation 4.13 becomes:

$$\Pi = N\alpha(\gamma CLV\beta_0\lambda - c) \tag{4.14}$$

This is identical to Equation 4.11, and thus the profit formula derived in Masand and Piatetsky-Shapiro (1996) is equivalent to the formulas derived in Mozer et al. (2000) and Neslin et al. (2006). The formula derived in Neslin et al. (2006) however adds to the formulas derived in Masand and Piatetsky-Shapiro (1996) and Mozer et al. (2000) a further specification of the costs, i.e., the costs of a retention campaign are split in the cost of making an offer to a customer c and the cost of the offer itself δ, which is only to be taken into account if the customer effectively accepts the offer (whether it be a would-be churner or a non-churner). Furthermore, Neslin et al. (2006) also explicitly specifies a fixed administrative cost of running a customer retention campaign, A. This is a constant term however, which

cancels out when using the profit formula to compare the results of different algorithms and their respective lift curves, and therefore it is of limited practical importance.

A customer lifetime value approach

Xevelonakis (2004) develops a framework for building effective and profitable customer retention campaigns based on customer behavior, customer profitability, and customer risk. The approach starts from a general formula to calculate the Net Present Value (NPV) of a customer, which is derived as a function of the churn rate p_t, the discount offer d_t, and the customer's constant monthly contribution margin (profit) w. p_t and d_t are time dependent, w is not. Future cash flows are discounted at a monthly discount rate r, and summed over a time period typically in the range of 12 to 36 months.

$$\text{NPV} = \sum_t \frac{w \cdot (\Pi_t(1-p_t) - d_t}{(1+r)^t} \tag{4.15}$$

The churn rate p_t is assumed to decrease during a customer retention campaign since customers receive an incentive to stay. Equation 4.16 expresses the churn rate as a function of time t, and two parameters a and b which depend on market dynamics:

$$p_t = \frac{1}{a \cdot (t+b)} \tag{4.16}$$

The NPV for the case in which no campaign is carried out, with a constant churn rate and a considered time period from $t = 1$ to $t = T2$, equals:

$$\text{NPV}_{\text{nocampaign}} = \sum_{t=1}^{t=T2} \frac{w \cdot (1-p_{churn})^t}{(1+r)^t} \tag{4.17}$$

On the other hand, when a campaign is carried out, the NPV can be calculated as follows, making use of the above equations:

$$\begin{aligned} \text{NPV}_{\text{campaign}} = & \sum_{t=1}^{t=T1} \frac{w \cdot \Pi_{n=1}^{t}(1-\frac{1}{a\cdot(n+b)}) - d}{(1+r)^t} \\ & + \sum_{t=T1+1}^{t=T2} \frac{w \cdot \Pi_{n=1}^{T1}(1-\frac{1}{a\cdot(n+b)}) \cdot (1-p_{churn})^{(t-T1)}}{(1+r)^t} \end{aligned} \tag{4.18}$$

with $T1$ the last month of the campaign's duration, and $T2$ the last considered month after the campaign. Finally, the additional profit of a campaign ΔNPV, is the difference between the NPV when a campaign is carried out and the NPV if no campaign is carried out:

$$\Delta\text{NPV} = \text{NPV}_{\text{campaign}} - \text{NPV}_{\text{nocampaign}} \tag{4.19}$$

The proposed approach in Xevelonakis (2004) defines the profit of a campaign as the difference between carrying out a campaign or doing nothing, which is similar to the approach of Mozer et al. (2000) (the formula of Neslin et al. (2006) on the other hand calculates the profit as the difference between the costs and incomes generated by a campaign, which is essentially equivalent as shown in Section 4.3.1). The approach proposed by Xevelonakis (2004) is on the other hand substantially different from the above discussed approaches in that it takes the time aspect of customer churn and customer retention campaigns into account. It is more a dynamic model of future cash flows than a static calculation of the profit generated by a customer retention campaign. The resulting expression does not take directly into account the effectiveness of the CCP model, and therefore it does not allow to compare different modeling approaches, nor does it offer the possibility to maximize the profit resulting from a retention campaign by optimizing the included fraction of customers. Furthermore, a number of assumptions are made regarding the evolution in time of the churn rate, the probability to churn, and the probability to be retained, which are not supported by any quantitative evidence. Therefore, this approach is deemed not suitable for the purpose of developing a maximum profit criterion, although it might offer a number of interesting features and insights, which complement these of the discussed approaches and of the criterion that will be introduced in the next section.

4.3.2 The maximum profit criterion

As shown by Equation 4.3, lift is directly related to profit, and therefore many studies on customer churn prediction use lift as a performance measure to evaluate CCP models. Since comparing entire lift curves is impractical and moreover rather meaningless, typically the lift at $\alpha = 5\%$ or $\alpha = 10\%$ is reported.

A first issue regarding the use of the lift criterion is illustrated by Figure 4.3. The lift curves of two different CCP models A and B intersect, resulting in a different model selection using top 5% lift and top 10% lift. In the case of Figure 4.3, if a model is selected based on top 10% lift but the effectively included fraction of customers in the retention campaign equals 5%, then the generated profit will be suboptimal due to a suboptimal choice of CCP model, which results directly from using an inappropriate performance criterion. Therefore, if lift is used to assess and compare the outcomes of different CCP models, then the lift at the fraction that will effectively be included in the retention campaign should be used.

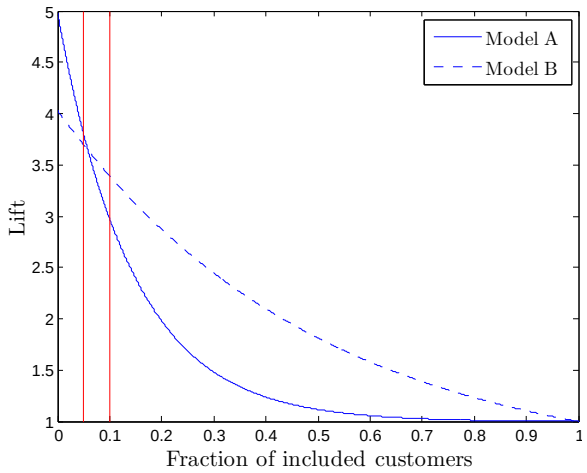


Figure 4.3: Lift Curves.

Furthermore, as illustrated by Figures 4.4(a) and 4.4(b), the profit will also be suboptimal if the fraction of customers that is included in the retention campaign is not the optimal fraction. Figures 4.4(a) and 4.4(b) represent on the Y-axis the profit per customer associated with including the fraction α on the X-axis of the customers ranked according to their probability to attrite in a retention campaign. The profit curves in Figures 4.4(a) and 4.4(b) are calculated using Equation 4.3 for models A and B and the lift curves shown in Figure 4.3. The values of the other parameters are equal to the values provided in Neslin et al. (2006). When using top decile lift or profit, model B would be selected, resulting in a suboptimal profit per customer. When using top 5% lift or profit, model A would be

selected, but when including the top 5% of the customers with the highest propensities to attrite, still a suboptimal profit is generated. And thus it is clear that in a practical setting the optimal fraction of customers should be included in a retention campaign to maximize profit, and that the profit or lift for the optimal fraction should be used to assess and compare the performance of CCP models.

Since the ultimate goal of a company by setting up a customer retention campaign is to minimize the costs associated with customer churn, it is straightforward to evaluate CCP models by using the maximum profit they can generate as a performance measure. In the remainder of this paper we will refer to this performance measure as the maximum profit (MP) criterion, which is formally defined as:

$$\text{MP} = \max_{\alpha}(\Pi) \tag{4.20}$$

In order to calculate the maximum profit measure a pragmatic approach is adopted, making two assumptions; (1) the retention rate γ is independent of the included fraction of customers α, and (2) the average CLV is independent of the included fraction of customers α. These assumptions allow to use a constant value for both γ and CLV in Equation 4.3, and given the lift curve of the classification model which represents the relation between the lift and α, the maximum of Equation 4.3 over α can be calculated in a straightforward manner.

In a realistic customer churn prediction setting, a retention campaign will only be profitable when including a rather small top-fraction of customers, with high predicted probabilities and with a relatively large subfraction of true would-be churners (i.e., with high lift). Hence, the optimal fraction α to maximize the returns of a retention campaign will lie within a rather small interval, with the lower bound equal to 0%. The assumptions that are made therefore relax to independency of both γ and CLV of α within this limited interval of α. In other words, the churners within the top-fraction of customers that are detected by classification models are assumed to be randomly distributed over this interval with respect to their CLV and probability to be retained. Since classification models induced by different techniques yield different probability scores to churn, and consequently result in a different distribution of the detected churners within the top ranked customers, this seems to be a reasonable assumption, which has not been contradicted to our knowledge by any empirical study in the

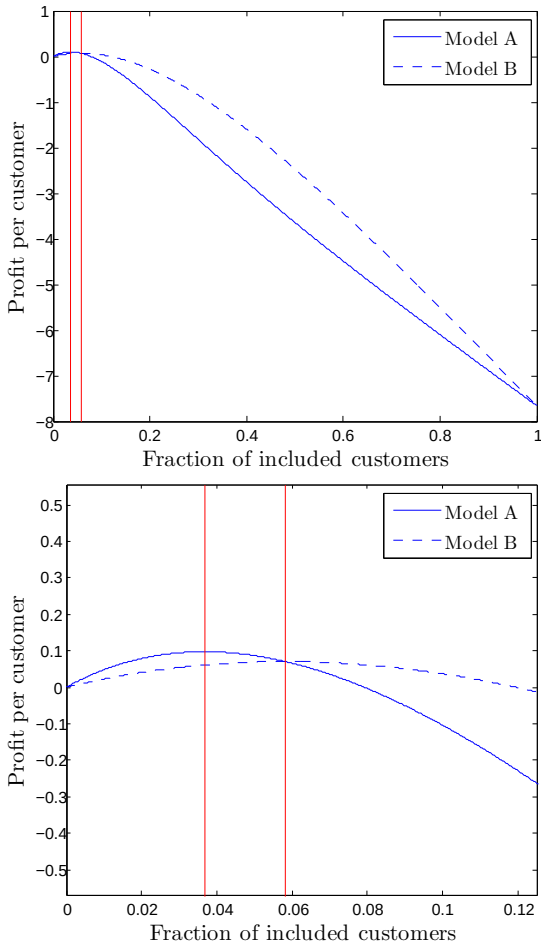


Figure 4.4: Profit function with vertical lines indicating the maximum profit per customer, and detail of top decile results.

literature.

The independency assumptions discussed above are strongly related to uplift modeling for churn prediction, i.e., adding a second stage model to a CCP model to identify the customers with the highest predicted probabilities to be retained, or with the highest expected return on investment, within the group of customers with the highest predicted probabilities to churn (as indicated by the first stage CCP model). Uplift modeling, the

existence of a correlation between the probability to be retained and the predicted probability to churn, and between the tendency to churn and the CLV, are marked as prime topics for future research, given the limited amount of studies that are available on this subject and the major relevance to customer relationship management and the development of retention strategies.

To calculate the maximum profit, in the remainder of this paper the values of the parameters γ, CLV, δ, and c in Equation 4.3 are set equal to respectively 0.30, 200, 10, and 1. The average values of the retention rate and the CLV are estimated by telco operators with high accuracy: huge budgets are spent on customer retention campaigns, which are typically conducted on a monthly basis and target millions of customers. Consequently, telco operators need to analyze the costs and benefits of such campaigns into detail, requiring accurate estimates of the average CLV of retained customers and average retention rates. Of course, the values may well differ for different segments (prepaid vs. postpaid, private vs. professional, etc.), and for different providers. The average values of the retention rate and the CLV that are used in the study are based on values found in the scientific literature (Neslin et al., 2006; Burez and Van den Poel, 2007), and based on information provided by data mining specialists of several telco providers. However, these values do not necessarily apply to each setting and may need to be adjusted when applying the MP measure. Moreover, it needs to be stressed that the contribution of this paper lies in the development and the application of the maximum profit criterion, rather than in the reported values of the maximum profits resulting from these parameter values.

Finally, it should be stressed that optimizing the fraction of customers to include in a retention campaign in order to maximize the profit generated by a customer retention campaign is an innovative, and highly valuable, key-insight for practitioners in the field of customer churn prediction. As will be shown in the next sections, optimizing the included fraction of customers, as well as selecting the optimal CCP model by using the maximum profit criterion, can lead to significant gains in profit for a company.

4.4 Experimental design

The experimental design of the benchmarking study consists of a full factorial experimental setup, in order to assess the effects of three different factors

on the performance of a CCP model. The first factor concerns the *classification technique*, and has 21 possible levels, i.e., one per technique that is evaluated. The main workings of the applied techniques will be briefly explained in Section 4.4.1. The second factor, *oversampling*, has two possible levels. Level zero means that the original data set is used, while level one indicates that oversampling is applied to improve the learning process, as will be explained in Section 4.4.2. Finally, the third factor represents *input selection*, and also has two levels. Level zero means no input selection is applied, and level one means that a generic input selection scheme is applied which will be presented in Section 4.4.3. This results in a $21 \times 2 \times 2$ experimental setup. The aim of the benchmarking study is to contrast the different levels and combinations of levels of these three factors, in order to draw general conclusions about the effects of classification technique, oversampling, and input selection on the performance of a CCP model. A full factorial experimental design allows to statistically test the effect of each factor separately, as well as the effects of interactions between the factors.

4.4.1 Classification techniques

Table 4.1 provides an overview of the classification techniques that are included in the benchmarking experiments (Lessmann et al., 2008). Previous studies reporting on the performance of a technique in a CCP modeling setting are referred to in the last column of the table. The included techniques are selected based on previous applications in churn prediction and expectations of good predictive power. An extensive overview of classification techniques can be found in Tan et al. (2006) or Hastie et al. (2001).

When appropriate, default values for the hyperparameters of the various techniques are used, based on previous empirical studies and evaluations reported in the literature. If unknown, a parameter optimization procedure is performed which calculates the performance of a model trained on $2/3$ of the training data and evaluated on the remaining validation set, for a range of parameter values. The values resulting in the best performing model are selected, and the final model is trained on the full training set using the selected parameter values. E.g., this procedure is performed for neural networks in order to determine the optimal number of hidden neurons, and for SVMs and LSSVMs to tune the kernel and regularization parameters. The benchmarking experiments are performed using implementations of the classification techniques in Weka, Matlab, SAS, and R.

Classification technique		Previous studies in churn prediction
Decision tree approaches		
A decision tree is grown in a recursive way by partitioning the training records into successively purer subsets. A minimum number of observations needs to fall into each subset, otherwise the tree is pruned. The metric to measure the pureness or homogeneity of the groups differs for the different techniques. C4.5 uses an entropy measure, while CART uses the Gini criterion. ADT is a boosted decision tree (see Ensemble methods) which distinguishes between alternating splitter and prediction nodes. A prediction is computed as the sum over all prediction nodes an instance visits while traversing the tree.		
Alternating Decision Tree (Freund and Trigg, 1999)	(ADT)	
C4.5 Decision Tree (Quinlan, 1993)	(C4.5)	Mozer et al. (2000), Wei and Chiu (2002), Au et al. (2003), Hwang et al. (2004), Hung et al. (2006), (Neslin et al., 2006), Kumar and Ravi (2008)
Classification and Regression Tree (Breiman et al., 1984)	(CART)	
Ensemble methods		
Ensemble methods use multiple base-classifiers resulting in better predictive performance than any of the constituent models, which are built independently and participate in a voting procedure to obtain a final class prediction. Random forest incorporates CART as base learner, Logistic Model Tree utilizes Logit, and both bagging and boosting use decision trees. Each base learner is derived from a limited number of attributes. These are selected at random within the RF procedure, whereby the user has to predefine the number. LMT considers only univariate regression models, i.e., uses one attribute per iteration, which is selected automatically. Bagging repeatedly samples with replacement from a data set according to a uniform probability distribution, and trains the base classifiers on the resulting data samples. Boosting adaptively changes the distribution of the training examples so that the base classifiers, will focus on examples that are hard to classify.		
Bagging	(Bag)	Lemmens and Croux (2006)
Boosting	(Boost)	Lemmens and Croux (2006)

Classification technique		Previous studies in churn prediction
Ensemble methods (ctd.)		
Logistic Model Tree (Landwehr et al., 2005)	(LMT)	
Random Forest	(RF)	Buckinx and Van den Poel (2005), Lariviere and Van den Poel (2005), Burez and Van den Poel (2007), Burez and Van den Poel (2009), Coussement and Van den Poel (2008), Kumar and Ravi (2008)
Nearest neighbors		
Nearest neighbor methods classify an instance based on the k-most similar or nearest instances. kNN methods measure the analogy or similarity between instances using the Euclidean distance. Following (Baesens et al., 2003b), both $k = 10$ *and* $k = 100$ *are included in the experiments.*		
k-Nearest Neighbors $k = 10$	(kNN10)	(Datta et al., 2000)
k-Nearest Neighbors $k = 100$	(kNN100)	
Neural networks		
Neural networks mathematically mimic the functioning of biological neural networks such as the human brain. They consist of a network of neurons, interconnected by functions and weights which need to be estimated by fitting the network to the training data. By applying the trained network on a customer's attributes, an approximation of its posterior probability of being a churner is obtained.		
Multilayer Perceptron (Bishop, 1996)	(NN)	Datta et al. (2000), Mozer et al. (2000), Au et al. (2003), Hwang et al. (2004) Buckinx and Van den Poel (2005), Hung et al. (2006), Neslin et al. (2006), Kumar and Ravi (2008)
Radial Basis Function Network	(RBFN)	Kumar and Ravi (2008)

Classification technique		Previous studies in churn prediction
Rule induction techniques		
Rule induction techniques result in a comprehensible set of if-then rules to predict the minority class, while the majority class is assigned by default. RIPPER is currently one of the dominant schemes for rule-learning, operating in two stages. First an initial rule set is induced, which is refined in a second optimization stage to filter contradictory rules. PART on the other hand infers rules by repeatedly generating partial decision trees, combining rule learning from decision trees with the separate-and-conquer rule-learning technique.		
PART (Frank and Witten, 1998)	(PART)	
RIPPER (Cohen, 1995)	(RIP)	Verbeke et al. (2011e)
Statistical classifiers		
Statistical classifiers model probabilistic relationships between the attribute set and the class variable. Posterior probabilities are estimated directly in logistic regression. Naive Bayes estimates the class-conditional probability by assuming that attributes are conditionally independent, given the class label, so that class-conditional probabilities can be estimated individually per attribute. Bayesian Networks allow a more flexible approach and extend Naive Bayes by explicitly specifying which pair of attributes is conditionally independent.		
Bayesian networks	(BN)	Neslin et al. (2006)
Logistic regression	(Logit)	Eiben et al. (1998), Mozer et al. (2000), Hwang et al. (2004), Buckinx and Van den Poel (2005), Lariviere and Van den Poel (2005), Lemmens and Croux (2006), Neslin et al. (2006), Burez and Van den Poel (2007), Burez and Van den Poel (2009), Coussement and Van den Poel (2008), Kumar and Ravi (2008)
Naive Bayes	(NB)	Neslin et al. (2006)

Classification technique		Previous studies in churn prediction
SVM based techniques		
SVM based classifiers construct a hyperplane or set of hyperplanes in a high-dimensional space to optimally discriminate between churners and non-churners, by maximizing the margin between two hyperplanes separating both classes. A kernel function enables more complex decision boundaries by means of an implicit, nonlinear transformation of attribute values. This kernel function is polynomial for the VP classifier, whereas SVM and LSSVM consider both a radial basis and a linear kernel function.		
LSSVM with linear kernel (Suykens and Vandewalle, 1999)	(linLSSVM)	
LSSVM with radial basis function kernel	(rbfLSSVM)	
SVM with linear kernel (Vapnik, 1995)	(linSVM)	Coussement and Van den Poel (2008), Kumar and Ravi (2008)
SVM with radial basis function kernel	(rbfSVM)	Coussement and Van den Poel (2008), Kumar and Ravi (2008)
Voted Perceptron (Freund and Schapire, 1999)	(VP)	

Table 4.1: Summary of the classification techniques that are evaluated in the benchmarking study.

4.4.2 Oversampling

The class variable of a typical churn prediction data set is heavily skewed, i.e., the number of churners is much smaller than the number of non-churners. This often causes classification techniques to experience difficulties in learning which customers are about to churn, resulting in bad performance. Since predicting future churners is the main objective of the model, sampling schemes can be used to change the class distribution and improve learning (Provost et al., 1998). Figure 2.1 in Chapter 2 illustrates the principle of oversampling. As discussed in Section 2.4.2, observations of the minority class in the training set are simply copied and added to the training set, thus changing its distribution. Oversampling does in fact not add

any new information to a data set, but only makes parts of the available information more explicit. Note that the class distribution of the test set is not altered, because a trained classifier is always evaluated on a pseudo-realistic data sample, in order to provide a correct indication of the future performance.

Alternatively, the class distribution of the training set can also be altered by removing observations from the majority class, which is called undersampling. However, undersampling reduces the available amount of information and therefore we have applied oversampling. Table 4.3 in Section 4.6.1 summarizes the number of observations in each data set included in the benchmarking study, and the class distribution of the original and oversampled data set. The degree of sampling affects the performance of the resulting classifier. Classification techniques typically perform best when the class distribution is approximately even, and therefore the data sets are oversampled to approximate an even class distribution.

4.4.3 Input selection

The third factor that possibly impacts the performance of a CCP model is the attribute selection procedure. In practice usually a limited number of highly predictive variables is preferred to be included in a classification model, in order to improve the comprehensibility, even at the cost of a decreased discrimination power (Piramuthu, 2004; Martens et al., 2007b). Therefore a procedure can be applied in order to select the most predictive attributes and to eliminate redundant attributes. In this chapter, a generic variable input selection procedure is applied which iteratively reduces the number of variables included in the model, i.e., a wrapper approach (Tan et al., 2006). Previous to applying the generic input selection procedure a number of redundant variables are already filtered from the data set using the Fisher score. This filter is applied since the computational requirements to apply the wrapper approach scales exponentially with the number of variables that is present in the data set. The Fisher score does not require discretization of the variables, and is defined as follows:

$$Fisher\ score = \frac{|\overline{x}_C - \overline{x}_{NC}|}{\sqrt{s_C^2 + s_{NC}^2}} \tag{4.21}$$

with $\overline{x}_C$ and $\overline{x}_{NC}$ the mean value, and s^2_C and s^2_{NC} the variance of a variable for respectively churners and non-churners. Typically, the 20 variables with the highest Fisher scores, indicating good predictive power, are selected. As will be shown in the results section, a subset of 20 variables suffices to achieve optimal performance.

Algorithm 6 Pseudo-code of input selection procedure

```
choose initial number of attributes k to start input selection procedure
split data in training data D_tr, and test data D_te in a 2/3, 1/3 ratio
calculate Fisher score of attributes in D_tr
select k attributes with highest Fisher scores and continue with this reduced data set D^k_tr
for i = k to 1 do
    for j = 1 to i do
        train model excluding attribute j from D^i_tr
        calculate performance P^i_j of model j
    end for
    remove attribute A_m from D^i_tr with P^i_m = max_j(P^i_j) resulting in D^{i-1}_tr
    performance in step i of input selection procedure P^i = P^i_m
end for
plot (i, P^i) with i = 1, ..., k
select cut-off value i with optimal trade-off between performance and number of variables
```

The input selection procedure starts from this reduced data set. In each step as many models are trained as there are variables left. Each of these models includes all variables except for one. The variable that is not included in the model with the best performance is removed from the data set, and a next iteration is started with one variable less in the data set. Hence the procedure starts with 20 models that are calculated, with each model including only nineteen variables. The variable excluded in the model with the best performance is then effectively removed from the data set, thus leaving a data set with only nineteen variables. This procedure is repeated, and eighteen models are estimated on the reduced data set. Again the variable excluded in the best performing model is removed from the data set. The procedure continues until no variables are left. A formal description of this procedure can be found in Algorithm 6. Figure 4.5 illustrates the input selection procedure by plotting the performance of the sequentially

best classifiers with a decreasing number of attributes. The number of attributes is shown on the X-axis, and the Y-axis represents the performance measured in terms of AUC, as will be explained in Section 4.5.2.

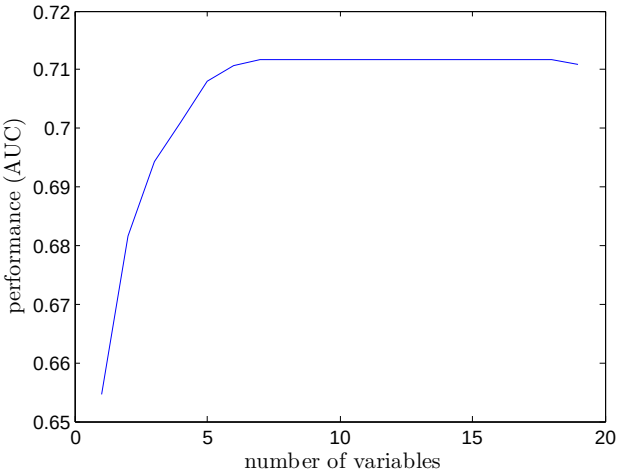


Figure 4.5: Example of the evolution of the performance during the input selection procedure for a decreasing number of variables (technique ADT applied on data set KDD without oversampling, cfr. infra). The X-axis represents the number of variables included in the model, while the Y-axis represents the performance of the model measured in terms of AUC.

As can be seen in Figure 4.5, removing a variable typically does not substantially impact the performance of a classifier when the number of variables remains large. The performance of the classifier drops however when the number of attributes left in the data set becomes too small. The model with the number of variables at the elbow point is generally considered to incorporate an optimal trade-off between minimizing the number of variables and maximizing the discriminatory power. The performance at the elbow point is the result of the input selection procedure that is reported in Section 4.6.

4.5 Research methodology

A robust experimental setup and the use of appropriate test statistics and performance measures are crucial in order to draw valid conclusions. Sec-

tion 4.5.1 describes the methodology that is followed in preprocessing the raw data sets, and Section 4.5.2 provides a non-exhaustive review of statistical (as opposed to profit centric) measures to assess the performance of classification models. This allows to correctly interpret the reported performance results in Section 4.6. Finally, Section 4.5.3 describes the statistical tests that will be applied to check the significance of differences in performance.

4.5.1 Data preprocessing

The general data mining process of developing a CCP model described in Section 4.2 is followed to apply the selected classification techniques on the collection of data sets. A first important step in this process concerns the preprocessing of the raw data. Missing values are handled depending on the percentage of missing values of an attribute. If less than 5% is missing, then the instances containing the missing value are removed from the data set. Since missing values from different attributes often seem to occur for the same instances, i.e., usually for the same customers multiple data fields are missing, the overall number of removed instances remained small. If more than 5% of the values of an attribute are missing then imputation procedures were applied. In case of categorical variables with many categories, coarse classification using hierarchical agglomerative clustering with the Euclidean distance is applied to reduce the number of categories to four (Tan et al., 2006). Finally all categorical variables are turned into binary variables using dummy encoding. No further preprocessing steps or transformations have been applied on the data.

4.5.2 Statistical performance measures

Percentage correctly classified

The percentage correctly classified observations measures the proportion of correctly classified cases on a sample of data. Although straightforward, the PCC may not be the most appropriate performance criterion in a number of cases, because it tacitly assumes equal misclassification costs for false positive and false negative predictions. This assumption can be problematic, since for most real-life problems, one type of classification error may be much more expensive than the other. For instance in a customer churn prediction setting the costs associated with not detecting a churner will

likely be greater than the costs of incorrectly classifying a non-churner as a churner (i.e., the costs associated with losing a customer tend to be greater than the costs of including a non-churner in a retention campaign). A second implicit assumption when using the PCC as evaluation criterion is that the class distribution (class priors) among examples is presumed constant over time, and relatively balanced (Provost et al., 1998). As mentioned in Section 4.4.2, the class distribution of a churn data set is typically skewed. Thus, using the PCC alone proves to be inadequate, since class distributions and misclassification costs are rarely uniform, and certainly not in the case of customer churn prediction. However, taking into account class distributions and misclassification costs proves to be quite hard, since in practice they can rarely be specified precisely, and are often subject to change (Fawcett and Provost, 1997).

Sensitivity, specificity, and the ROC curve

Class-wise decomposition of the classification of cases yields a confusion matrix as specified in Table 4.2. If TP, FP, FN, and TN represent the number of *true positives*, *false positives*, *false negatives*, and *true negatives*, then the *sensitivity* or *true positive rate* measures the proportion of positive examples which are predicted to be positive (TP/(TP + FN)) (e.g., the percentage of churners that is correctly classified), whereas the *specificity* or the *true negative rate* measures the proportion of negative examples which are predicted to be negative (TN/(TN + FP)) (e.g., the percentage of non-churners that are correctly classified).

		Actual	
		+	−
Predicted	+	True Positive (TP)	False Positive (FP)
	−	False Negative (FN)	True Negative (TN)

Table 4.2: The confusion matrix for binary classification.

Using the notation of Table 4.2, we may now formulate the overall accuracy as PCC = (TP + TN)/(TP + FP + TN + FN). Note that sensitivity, specificity, and PCC vary together as the threshold on a classifier's continuous output is varied between its extremes. The *receiver operating characteristic curve* (ROC) is a 2-dimensional graphical illustration of the sensitivity on the Y-axis versus (1-specificity) on the X-axis for various val-

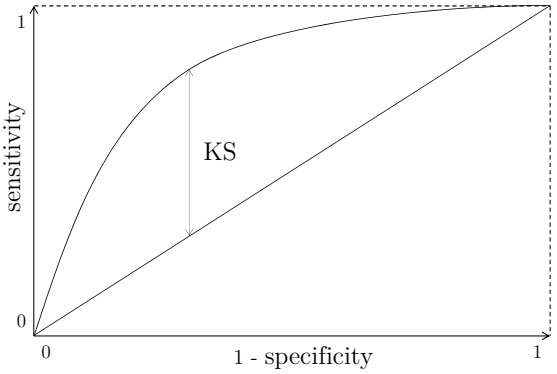


Figure 4.6: Example of ROC curve with Kolmogorov-Smirnov statistic indicated.

ues of the classification threshold. It basically illustrates the behavior of a classifier without regard to class distribution or misclassification cost, so it effectively decouples classification performance from these factors (Egan, 1975; Swets and Pickett, 1982). An example of a ROC curve is shown in Figure 4.6.

Area under the ROC curve

In order to compare ROC curves of different classifiers, one often calculates the *area under the receiver operating characteristic curve* (AUROC or AUC). Assume a classifier produces a score $s = s(x)$, function of the attribute values x, with corresponding probability density function of these scores for class k instances $f_k(s)$ and cumulative distribution function $F_k(s)$, with only two classes $k = 0, 1$. Then the AUC is defined as (Krzanowski and Hand, 2009):

$$AUC = \int_{-\infty}^{\infty} F_0(s) f_1(s) ds \tag{4.22}$$

The AUC provides a simple figure-of-merit for the performance of the constructed classifier. An intuitive interpretation of the AUC is that it provides an estimate of the probability that a randomly chosen instance of class 1 is correctly rated or ranked higher than a randomly selected instance of class 0 (e.g., the probability that a churner is assigned a higher probability to churn than a non-churner). Note that since the area under the diagonal

corresponding to a pure random classification model is equal to 0.5, a good classifier should yield an AUC much larger than 0.5.

Gini coefficient and Kolmogorov-Smirnov statistic

A measure that is closely related to the AUC is the *Gini coefficient* (Thomas et al., 2002), which is equal to twice the area between the ROC curve and the diagonal, i.e., $Gini = 2{*}\text{AUC}-1$. The Gini coefficient varies between 0 (i.e., the ROC curve lies on the diagonal and the model does not perform better than a random classification model) and 1 (i.e., maximum ROC curve and perfect classification).

Another performance measure related to the ROC curve is the *Kolmogorov-Smirnov* (KS) statistic. The KS statistic gives the maximum distance between the ROC curve and the diagonal at a specific cut-off value. Again, a value of the KS performance measure equal to one means a perfect classification, and KS equal to zero means no better classification than a random classifier. The KS measure is indicated in Figure 4.6.

4.5.3 Statistical tests

A procedure described in Demšar (2006) is followed to statistically test the results of the benchmarking experiments and contrast the levels of the factors. In a first step of this procedure the non-parametric Friedman test (Friedman, 1940) is performed to check whether differences in performance are due to chance. The Friedman statistic is defined as:

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right] \tag{4.23}$$

with R_j the average rank of algorithm $j = 1, 2, \ldots, k$ over N data sets. Under the null hypothesis that no significant differences exist, the Friedman statistic is distributed according to χ_F^2 with $k-1$ degrees of freedom, at least when N and k are big enough (i.e., $N > 10$ and $k > 5$), which is the case in this chapter when comparing different techniques ($N = 11$ and $k = 21$). When comparing the levels of the factors oversampling and input selection, k equals two and exact critical values need to be used to calculate the statistic.

If the null hypothesis is rejected by the Friedman test we proceed by performing the post-hoc Nemenyi (Nemenyi, 1963) test to compare all classifiers to each other. Two classifiers yield significantly different results if their average ranks differ by at least the critical difference equal to:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{4.24}$$

with critical values q_α based on the Studentized range statistic divided by $\sqrt{2}$. To compare all classifiers with the best performing classifier the Bonferroni-Dunn test (Dunn, 1961) is applied, which is similar to post-hoc Nemenyi but adjusts the confidence level in order to control the family-wise error for making $k-1$ instead of $k(k-1)/2$ comparisons.

The previous tests are applied to compare the results over multiple data sets, and to draw general conclusions about the impact of a factor on the performance of a model. To compare the performance, measured in AUC, of two classifiers on a single data set, the test of DeLong, DeLong, and Clarke-Pearson is used. After complex mathematical calculus, following a non-parametric approach whereby the covariance matrix is estimated using the theory on generalized U-statistics, DeLong et al. (1988) derive the following test statistic:

$$(\hat{\Theta} - \Theta) c^T [cSc^T]^{-1} c (\hat{\Theta} - \Theta)^T \tag{4.25}$$

which has a chi-square distribution with degrees of freedom equal to the rank of cSc^T with $\hat{\Theta}$ the vector of the AUC estimates, S the estimated covariance matrix, and c a vector of coefficients such that $c\Theta^T$ represents the desired contrast.

4.6 Empirical results

4.6.1 Data sets

Eleven data sets were obtained from wireless telco operators around the world. Table 4.3 summarizes the main characteristics of these data sets, some of which have been used in previous studies referred to in the last column of the table. As can be seen from the table, the smallest data set

[2] www.fuqua.duke.edu/centers/ccrm/datasets/download.html
[3] www.sgi.com/tech/mlc/db
[4] www.kddcup-orange.com

ID	Source	# Obs.	# Att.	C.R. orig.	C.R. samp.	Reference
O1	Operator	47,761	53	3.69	50.01	Mozer et al. (2000)
O2	Operator	11,317	21	1.56	47.44	Hur and Kim (2008)
O3	Operator	2,904	15	3.20	55,52	Hur and Kim (2008)
O4	Operator	2,969	48	4.41	45.99	Hur and Kim (2008)
O5	Operator	2,180	15	3.21	55.97	Hur and Kim (2008)
O6	Operator	338,874	727	1.80	50	
D1	Duke[2]	93,893	197	1.78	49.75	Neslin et al. (2006) Lemmens and Croux (2006) Lima et al. (2009)
D2	Duke[2]	38,924	77	1.99	49.81	
D3	Duke[2]	7,788	19	3.30	56.49	
UCI	UCI[3]	5,000	23	14.14	50.28	Lima et al. (2009) Verbeke et al. (2011e)
KDD	KDD Cup[4]	46,933	242	6.98	50.56	

Table 4.3: Summary of data set characteristics: ID, source, number of observations, number of attributes, original and sampled churn rates (C.R.), and references to previous studies using the data set.

contains $2,180$ observations, and the largest up to $338,874$ observations. This allows to split each data set at random into $2/3$ training set and $1/3$ test set. The training set is used to learn a model, which is then evaluated on the test set to obtain an unbiased indication of the performance of the model. For computational reasons, occasionally a subsample of the original data set was used to train a model.

The data sets also differ substantially regarding the number of attributes, in a range from 15 up to 727. However, more attributes do not guarantee a better classification model. The final performance of a classifier mainly depends on the explanatory power of the attributes, and not on the number of attributes available to train a model. For instance, the number of times a customer called the helpdesk will most probably be a better predictor of churn behavior than the zip code. A large number of attributes heavily increases the computational requirements. Therefore the number of variables in data sets O6, D1, and KDD is reduced to a number of 50 using the Fisher score, in order to remove redundant data before applying the classification techniques without input selection. As explained in Section 4.4.3, before applying the wrapper input selection procedure the number of variables in all data sets is reduced to a maximum of 20.

Table 4.3 also indicates the class distribution, which is for all data sets heavily skewed. The percentage of churners typically lies within a range of 1% to 10% of the entire customer base, depending on the length of the period in which churn is measured.

The definition of churn also slightly differs over the data sets, depending on the operator providing the data set. Most of the data however is collected over a period of three to six months, with a churn flag indicating whether a customer churned in the month after the month following the period when the data was collected. The one month lag of the data on the churn flag gives the marketing department time to setup campaigns aimed at retaining the customers that are predicted to churn.

4.6.2 Results and discussion

In the first part of this section the results of the benchmarking experiment are evaluated to assess the impact of input selection, oversampling, and classification technique on the performance of a CCP model. The performance is measured and evaluated using both statistical performance measures, i.e., top decile lift and AUC, and the maximum profit criterion. The use of these performance measures is analyzed and discussed in the second part of this section.

Tables 4.4, 4.5, and 4.6 report the results of the benchmarking study in terms of respectively maximum profit, top decile lift, and AUC, both with and without oversampling and input selection applied. In all three tables the Bonferroni-Dunn test is used to evaluate the average ranks (AR) resulting from the respective performance measures. The best performance is in bold and underlined, and results that are not significantly different from the top performance at the 95% confidence level are tabulated in bold. Statistically significant differences in performance at the 99% level are emphasized in italics, and significantly different results at the 95% level but not at the 99% level are reported in normal script. In Table 4.6, as explained in Section 4.5.3, the results of the test of DeLong, DeLong, and Clarke-Pearson to compare the performance in AUC on each data set separately are reported following the same notational convention.

Data set	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD	AR	AA	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD	AR	AA
Without input selection																										
NN	*0.10*	*0.05*	*0.34*	*1.38*	*0.00*	*0.03*	*0.00*	*0.00*	*0.22*	*5.36*	*0.26*	**6.00**	*0.70*	*0.07*	*0.00*	*0.33*	*1.40*	*0.14*	*0.00*	*0.00*	*0.00*	*0.00*	*0.09*	*3.97*	*0.33*	**8.59**	*0.57*
linSVM	*0.00*	*0.00*	*0.06*	*1.41*	*0.05*	*0.00*	*0.00*	*0.00*	*0.00*	*3.05*	*0.09*	*14.14*	*0.42*	*0.00*	*0.00*	*0.45*	*1.31*	*0.05*	*0.01*	*0.00*	*0.00*	*0.00*	*0.06*	*3.72*	*0.09*	**10.55**	*0.52*
rbfSVM	*0.03*	*0.00*	*0.28*	*1.37*	*0.11*	*0.00*	*0.00*	*0.00*	*0.00*	*4.92*	*0.09*	**10.50**	*0.62*	*0.00*	*0.00*	*0.00*	*1.16*	*0.05*	*0.00*	*0.00*	*0.01*	*0.00*	*5.16*	*0.09*	**12.41**	*0.59*
linLSSVM	*0.04*	*0.00*	*0.48*	*1.42*	*0.15*	*0.00*	*0.00*	*0.00*	*0.16*	*3.80*	*0.21*	**9.64**	*0.57*	*0.04*	*0.00*	*0.53*	*1.32*	*0.14*	*0.01*	*0.00*	*0.00*	*0.11*	*3.78*	*0.23*	**7.73**	*0.56*
rbfLSSVM	*0.08*	*0.00*	*0.46*	*1.49*	*0.14*	*0.00*	*0.00*	*0.00*	*0.11*	*4.96*	*0.07*	**9.59**	*0.66*	*0.04*	*0.00*	*0.00*	*1.05*	*0.14*	*0.00*	*0.00*	*0.00*	*0.00*	*4.81*	*0.09*	**11.36**	*0.56*
RIPPER	*0.02*	*0.00*	*0.16*	*0.48*	*1.57*	*0.01*	*0.00*	*0.00*	*0.16*	*5.24*	*0.00*	**10.86**	*0.70*	*0.00*	*0.00*	*0.04*	*0.26*	*1.35*	*0.00*	*0.00*	*0.00*	*0.04*	*4.45*	*0.01*	**12.55**	*0.56*
PART	*0.00*	*0.00*	*0.17*	*1.14*	*1.28*	*0.03*	*0.00*	*0.01*	*0.18*	*4.97*	*0.10*	**7.64**	*0.72*	*0.00*	*0.00*	*0.13*	*0.77*	*1.37*	*0.00*	*0.00*	*0.00*	*0.05*	*5.15*	*0.10*	**11.18**	*0.69*
C4.5	*0.06*	*0.00*	*0.00*	*0.72*	*1.30*	*0.00*	*0.00*	*0.00*	*0.21*	*5.09*	*0.14*	**10.59**	*0.68*	*0.00*	*0.02*	*0.15*	*0.70*	*1.36*	*0.00*	*0.00*	*0.00*	*0.04*	*4.99*	*0.07*	**11.23**	*0.67*
CART	*0.00*	*0.00*	*0.00*	*1.04*	*1.53*	*0.00*	*0.00*	*0.00*	*0.00*	*5.41*	*0.08*	*12.36*	*0.73*	*0.00*	*0.00*	*0.15*	*1.29*	*1.60*	*0.00*	*0.00*	*0.00*	*0.00*	*5.42*	*0.01*	**11.09**	*0.77*
ADT	*0.12*	*0.01*	*0.08*	*0.98*	*1.57*	*0.00*	*0.00*	*0.00*	*0.21*	*5.21*	*0.30*	**6.41**	*0.77*	*0.12*	*0.01*	*0.07*	*1.39*	*1.61*	*0.00*	*0.00*	*0.00*	*0.08*	*5.21*	*0.30*	**5.59**	*0.80*
RF	*0.06*	*0.00*	*0.36*	*1.03*	*1.30*	*0.00*	*0.00*	*0.00*	*0.18*	*5.76*	*0.14*	**7.27**	*0.80*	*0.08*	*0.01*	*0.22*	*1.38*	*1.32*	*0.00*	*0.00*	*0.00*	*0.22*	*5.80*	*0.14*	**5.77**	*0.83*
LMT	*0.00*	*0.00*	*0.00*	*1.09*	*1.36*	*0.00*	*0.00*	*0.00*	*0.00*	*5.49*	*0.24*	**11.18**	*0.74*	*0.00*	*0.00*	*0.19*	*0.60*	*1.32*	*0.00*	*0.00*	*0.00*	*0.06*	*5.19*	*0.04*	**10.50**	*0.67*
Bag	*0.09*	*0.02*	*0.41*	*1.13*	*1.65*	*0.04*	*0.00*	*0.00*	*0.19*	*5.63*	*0.29*	**4.18**	*0.86*	*0.02*	*0.00*	*0.32*	*1.22*	*1.54*	*0.00*	*0.02*	*0.12*	*5.36*	*0.19*	*0.01*	**12.55**	*0.80*
Boost	*0.03*	*0.00*	*0.67*	*1.06*	*1.45*	*0.00*	*0.00*	*0.00*	*0.18*	*4.35*	*0.13*	**9.64**	*0.72*	*0.03*	*0.00*	*0.20*	*1.15*	*1.48*	*0.00*	*0.00*	*0.00*	*0.19*	*4.10*	*0.14*	**8.18**	*0.71*
RBFN	*0.00*	*0.00*	*0.40*	*0.95*	*0.51*	*0.00*	*0.00*	*0.00*	*0.18*	*4.36*	*0.02*	*12.27*	*0.58*	*0.02*	*0.00*	*0.41*	*0.09*	*0.13*	*0.00*	*0.00*	*0.00*	*0.21*	*4.38*	*0.13*	**9.45**	*0.49*
VP	*0.00*	*0.00*	*0.00*	*0.00*	*0.08*	*0.00*	*0.00*	*0.00*	*0.00*	*0.03*	*0.00*	*16.18*	*0.01*	*0.00*	*0.00*	*0.00*	*0.08*	*0.08*	*0.00*	*0.00*	*0.00*	*0.00*	*0.94*	*0.00*	*16.14*	*0.10*
Logit	*0.08*	*0.00*	*0.40*	*1.38*	*0.14*	*0.03*	*0.00*	*0.00*	*0.19*	*4.00*	*0.24*	**8.50**	*0.59*	*0.08*	*0.00*	*0.48*	*1.25*	*0.15*	*0.01*	*0.00*	*0.00*	*0.11*	*3.97*	*0.22*	**7.27**	*0.57*
kNN10	*0.00*	*0.00*	*0.11*	*1.07*	*0.05*	*0.00*	*0.00*	*0.00*	*0.04*	*4.02*	*0.05*	*12.59*	*0.49*	*0.00*	*0.00*	*0.17*	*0.97*	*0.30*	*0.00*	*0.00*	*0.00*	*0.02*	*3.77*	*0.00*	**12.50**	*0.48*
kNN100	*0.00*	*0.00*	*0.23*	*0.98*	*0.06*	*0.01*	*0.00*	*0.00*	*0.10*	*4.28*	*0.04*	*13.00*	*0.52*	*0.00*	*0.00*	*0.16*	*1.32*	*0.05*	*0.00*	*0.00*	*0.00*	*0.06*	*3.99*	*0.00*	**11.00**	*0.51*
BN	*0.04*	*0.00*	*0.59*	*1.40*	*1.22*	*0.05*	*0.00*	*0.00*	*0.35*	*4.08*	*0.25*	**6.64**	*0.73*	*0.00*	*0.00*	*0.00*	*0.05*	*0.47*	*0.00*	*0.00*	*0.00*	*0.11*	*3.71*	*0.25*	**11.36**	*0.42*
NB	*0.00*	*0.00*	*0.42*	*1.07*	*0.11*	*0.00*	*0.00*	*0.00*	*0.21*	*4.57*	*0.11*	**10.82**	*0.59*	*0.00*	*0.00*	*0.45*	*1.02*	*0.14*	*0.00*	*0.00*	*0.00*	*0.21*	*4.57*	*0.11*	**9.77**	*0.59*
With input selection																										
NN	*0.05*	*0.01*	*0.49*	*1.22*	*1.45*	*0.01*	*0.00*	*0.03*	*0.09*	*3.66*	*0.21*	**7.68**	*0.66*	*0.08*	*0.00*	*0.25*	*1.11*	*1.53*	*0.02*	*0.00*	*0.00*	*0.15*	*5.86*	*0.28*	**7.59**	*0.84*
linSVM	*0.02*	*0.00*	*0.60*	*1.51*	*0.08*	*0.00*	*0.00*	*0.01*	*0.06*	*3.65*	*0.00*	**12.86**	*0.54*	*0.00*	*0.00*	*0.39*	*1.30*	*0.00*	*0.00*	*0.00*	*0.00*	*0.04*	*3.81*	*0.11*	14.91	*0.51*
rbfSVM	*0.05*	*0.00*	*0.69*	*0.94*	*0.98*	*0.00*	*0.00*	*0.00*	*0.00*	*5.06*	*0.09*	**13.32**	*0.71*	*0.00*	*0.00*	*0.00*	*1.33*	*1.39*	*0.00*	*0.00*	*0.00*	*0.05*	*5.61*	*0.11*	**12.50**	*0.77*
linLSSVM	*0.01*	*0.00*	*0.46*	*1.26*	*0.08*	*0.00*	*0.00*	*0.01*	*0.02*	*3.97*	*0.20*	**13.41**	*0.55*	*0.03*	*0.00*	*0.50*	*1.37*	*0.11*	*0.00*	*0.00*	*0.02*	*0.02*	*3.97*	*0.20*	**10.59**	*0.56*
rbfLSSVM	*0.05*	*0.00*	*0.45*	*1.41*	*1.05*	*0.00*	*0.00*	*0.01*	*0.22*	*5.45*	*0.16*	**9.82**	*0.80*	*0.06*	*0.01*	*0.00*	*1.15*	*1.39*	*0.00*	*0.07*	*0.00*	*0.20*	*5.37*	*0.16*	**9.05**	*0.76*
RIPPER	*0.03*	*0.00*	*0.14*	*0.85*	*1.65*	*0.00*	*0.00*	*0.02*	*0.26*	*5.64*	*0.06*	**9.55**	*0.79*	*0.00*	*0.00*	*0.05*	*0.53*	*1.52*	*0.02*	*0.00*	*0.01*	*0.05*	*5.19*	*0.01*	**11.45**	*0.67*
PART	*0.01*	*0.00*	*0.18*	*1.39*	*1.57*	*0.03*	*0.00*	*0.02*	*0.24*	*5.47*	*0.26*	**6.86**	*0.83*	*0.00*	*0.00*	*0.05*	*1.22*	*1.45*	*0.02*	*0.54*	*0.01*	*0.04*	*5.41*	*0.21*	**8.86**	*0.81*
C4.5	*0.04*	*0.00*	*0.46*	*1.14*	*1.45*	*0.03*	*0.00*	*0.02*	*0.27*	*5.33*	*0.21*	**6.36**	*0.81*	*0.00*	*0.00*	*0.36*	*1.07*	*1.53*	*0.02*	*0.46*	*0.01*	*0.00*	*5.17*	*0.20*	**10.23**	*0.80*
CART	*0.00*	*0.00*	*0.64*	*1.05*	*1.46*	*0.02*	*0.00*	*0.01*	*0.12*	*5.21*	*0.31*	**7.18**	*0.81*	*0.02*	*0.00*	*0.64*	*1.10*	*1.46*	*0.02*	*0.00*	*0.02*	*0.10*	*5.21*	*0.29*	**6.05**	*0.81*
ADT	*0.05*	*0.00*	*0.64*	*1.05*	*1.46*	*0.02*	*0.00*	*0.01*	*0.12*	*5.21*	*0.31*	**7.18**	*0.81*	*0.02*	*0.00*	*0.64*	*1.10*	*1.46*	*0.02*	*0.00*	*0.02*	*0.10*	*5.21*	*0.29*	**6.05**	*0.81*
RF	*0.01*	*0.01*	*0.38*	*1.14*	*1.45*	*0.03*	*0.66*	*0.02*	*0.16*	*5.73*	*0.13*	**7.14**	*0.88*	*0.01*	*0.00*	*0.53*	*0.95*	*1.35*	*0.02*	*0.67*	*0.02*	*0.15*	*5.69*	*0.13*	**7.73**	*0.86*
LMT	*0.03*	*0.00*	*0.25*	*1.50*	*1.65*	*0.03*	*0.00*	*0.02*	*0.05*	*5.40*	*0.30*	**6.77**	*0.84*	*0.00*	*0.00*	*0.46*	*0.96*	*1.44*	*0.03*	*0.36*	*0.00*	*0.02*	*5.32*	*0.17*	**11.14**	*0.79*
Bag	*0.07*	*0.01*	*0.45*	*1.13*	*1.65*	*0.02*	*0.14*	*0.00*	*0.08*	*5.63*	*0.34*	**5.91**	*0.86*	*0.02*	*0.01*	*0.30*	*0.91*	*1.45*	*0.03*	*0.67*	*0.00*	*0.15*	*5.46*	*0.13*	**8.09**	*0.83*
Boost	*0.02*	*0.00*	*0.70*	*1.11*	*1.45*	*0.00*	*0.00*	*0.00*	*0.04*	*4.66*	*0.16*	**12.77**	*0.74*	*0.03*	*0.00*	*0.70*	*1.11*	*1.45*	*0.00*	*0.00*	*0.00*	*0.05*	*4.10*	*0.18*	**10.27**	*0.69*
RBFN	*0.06*	*0.00*	*0.36*	*1.30*	*1.38*	*0.00*	*0.00*	*0.01*	*0.21*	*4.82*	*0.21*	**10.00**	*0.76*	*0.06*	*0.00*	*0.41*	*1.21*	*1.17*	*0.00*	*0.00*	*0.02*	*0.20*	*5.03*	*0.16*	**10.14**	*0.75*
VP	*0.01*	*0.00*	*0.06*	*0.63*	*0.15*	*0.00*	*0.00*	*0.00*	*0.00*	*0.12*	*0.00*	*15.73*	*0.09*	*0.00*	*0.00*	*0.40*	*0.01*	*0.08*	*0.00*	*0.00*	*0.02*	*0.03*	*1.98*	*0.00*	*15.68*	*0.23*
Logit	*0.07*	*0.00*	*0.49*	*1.39*	*0.03*	*0.01*	*0.00*	*0.02*	*0.05*	*4.00*	*0.18*	**9.86**	*0.57*	*0.09*	*0.00*	*0.49*	*1.36*	*0.03*	*0.01*	*0.00*	*0.02*	*0.03*	*4.04*	*0.17*	**10.00**	*0.56*
kNN10	*0.02*	*0.00*	*0.33*	*1.11*	*1.37*	*0.01*	*0.00*	*0.00*	*0.08*	*5.07*	*0.24*	**12.09**	*0.74*	*0.00*	*0.00*	*0.54*	*1.27*	*1.33*	*0.01*	*0.60*	*0.00*	*0.04*	*4.98*	*0.24*	**9.32**	*0.82*
kNN100	*0.02*	*0.00*	*0.33*	*1.11*	*1.27*	*0.00*	*0.00*	*0.00*	*0.08*	*5.29*	*0.19*	**13.32**	*0.75*	*0.02*	*0.00*	*0.40*	*1.02*	*1.45*	*0.01*	*0.03*	*0.00*	*0.13*	*5.36*	*0.23*	**9.59**	*0.79*
BN	*0.05*	*0.00*	*0.57*	*1.27*	*1.53*	*0.00*	*0.00*	*0.02*	*0.26*	*4.28*	*0.36*	**7.32**	*0.76*	*0.01*	*0.00*	*0.57*	*1.09*	*1.47*	*0.01*	*0.54*	*0.01*	*0.04*	*4.17*	*0.33*	**8.41**	*0.75*
NB	*0.03*	*0.00*	*0.44*	*1.27*	*0.81*	*0.00*	*0.00*	*0.02*	*0.23*	*4.56*	*0.19*	**10.86**	*0.68*	*0.03*	*0.00*	*0.44*	*1.27*	*0.81*	*0.00*	*0.00*	*0.02*	*0.23*	*4.58*	*0.19*	**9.64**	*0.69*
Without oversampling														**With oversampling**												

Table 4.4: Results of the benchmarking experiment evaluated using the MP criterion.

Data set	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD	AR	AA	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD	AR	AA	
NN	3.52	1.93	4.28	7.86	1.59	3.77	1.50	1.09	3.59	6.10	2.67	**6.77**	3.45	3.71	1.75	5.13	7.86	3.98	4.02	1.37	0.73	3.81	3.20	2.82	**8.14**	3.49	
linSVM	2.01	0.70	1.43	7.38	2.39	1.39	1.06	0.18	1.91	3.12	0.98	16.82	2.05	1.55	1.93	5.70	7.86	1.19	4.92	1.50	0.73	4.04	3.68	0.92	**10.27**	3.09	
rbfSVM	2.68	1.58	3.99	7.86	3.18	2.05	1.06	0.91	3.48	4.68	1.52	**11.68**	3.00	2.46	1.93	2.00	7.38	2.78	0.74	1.37	1.82	1.12	5.93	1.29	**11.45**	2.62	
linLSSVM	2.98	1.58	6.27	7.62	2.39	2.79	1.23	0.55	3.59	3.55	2.47	**10.18**	3.18	3.19	1.23	5.70	8.10	1.99	4.92	1.50	0.73	4.04	3.42	2.48	**8.05**	3.39	
rbfLSSVM	3.29	1.75	5.99	7.86	4.38	3.12	1.30	0.55	3.70	5.50	2.04	**8.64**	3.59	3.14	1.75	3.14	7.14	2.78	2.54	1.67	1.09	1.12	5.67	2.21	**10.18**	2.93	
RIPPER	1.04	0.88	3.14	3.49	8.35	0.90	1.33	0.55	1.79	6.36	0.86	13.91	2.61	1.85	1.23	3.42	2.86	7.96	1.48	1.33	0.91	2.02	4.20	1.41	14.55	2.61	
PART	2.58	1.23	4.28	6.74	7.16	4.67	1.60	0.91	3.81	6.02	2.25	**8.36**	3.75	1.77	1.75	3.42	4.29	7.16	1.39	1.26	1.09	2.47	5.28	1.95	13.05	2.89	
C4.5	2.61	0.88	2.00	4.88	7.16	0.74	1.33	0.55	2.02	6.17	1.48	13.82	2.71	1.62	1.93	3.71	4.76	7.16	1.97	1.02	0.91	2.47	5.32	1.92	12.91	2.98	
CART	0.84	0.88	2.00	6.05	8.35	0.74	1.33	0.55	1.12	6.23	1.35	14.27	2.68	2.11	1.40	4.28	6.90	8.35	1.64	1.43	0.73	2.80	5.84	1.87	**11.23**	3.40	
ADT	3.39	1.58	4.28	6.05	8.35	5.33	1.50	1.09	4.26	5.97	2.73	**6.14**	4.05	3.39	1.93	4.28	7.38	8.75	5.17	1.57	1.09	3.81	5.97	2.73	**4.50**	4.19	
RF	2.82	2.80	5.42	6.51	7.56	3.61	0.99	0.55	3.48	6.71	2.12	**8.82**	3.87	2.93	2.10	4.56	7.62	8.75	4.26	1.13	1.09	3.59	6.71	2.32	**6.45**	4.10	
LMT	0.84	0.88	2.00	6.05	7.16	0.74	1.33	0.55	1.12	6.54	2.65	13.23	2.71	1.94	1.93	4.56	5.48	8.75	3.85	1.26	0.55	2.02	6.10	1.99	**10.64**	3.49	
Bag	3.41	2.45	5.42	6.05	8.75	5.90	1.54	1.82	3.25	6.75	2.74	**4.05**	4.37	2.98	2.10	4.28	7.14	8.75	4.18	1.43	0.91	3.03	6.67	2.41	**6.41**	3.99	
Boost	2.87	2.28	5.99	6.74	9.15	2.87	1.67	1.27	3.93	4.42	2.43	**5.73**	3.96	3.05	2.10	6.27	5.95	9.15	2.87	1.67	1.27	3.93	3.55	2.50	**5.32**	3.85	
RBFN	2.60	1.58	5.13	6.74	5.17	3.53	1.09	1.27	4.15	4.59	1.60	**9.68**	3.40	2.31	1.93	5.42	2.62	3.18	3.20	1.06	1.45	4.38	4.46	2.46	**8.95**	2.95	
VP	0.84	0.88	2.00	1.16	0.80	0.74	1.33	0.55	1.12	0.95	1.06	17.77	1.04	2.04	0.88	1.71	1.43	1.19	2.79	1.33	0.73	1.12	1.43	1.16	17.27	1.44	
Logit	3.66	1.05	5.42	7.91	1.99	5.08	1.57	0.91	3.93	3.72	2.59	**7.09**	3.44	3.68	1.05	5.99	7.62	1.99	5.66	1.43	1.09	3.93	3.77	2.55	**7.09**	3.52	
kNN10	1.84	2.45	3.14	6.90	4.38	3.20	1.13	0.73	3.14	4.37	1.89	**11.86**	3.02	1.89	2.28	3.71	6.67	3.98	3.20	1.26	0.55	2.80	4.33	1.68	**12.36**	2.94	
kNN100	2.09	2.63	4.85	6.74	4.38	4.35	1.37	0.73	3.93	4.33	1.93	**9.36**	3.39	1.91	1.93	3.14	7.86	4.38	3.77	1.33	1.09	3.81	4.33	1.65	**10.18**	3.20	
BN	2.85	1.75	5.13	8.14	7.16	5.98	1.54	1.45	4.49	4.50	2.45	**4.95**	4.13	2.31	1.75	2.00	1.43	6.36	0.66	1.30	0.91	2.80	4.33	2.43	13.23	2.39	
NB	2.95	1.75	5.70	7.21	3.58	2.46	1.64	1.27	4.26	4.55	2.35	**6.86**	3.43	2.95	1.58	5.70	6.90	3.18	2.46	1.60	1.27	4.15	4.55	2.34	**7.77**	3.34	
NN	3.39	2.80	6.27	7.67	8.75	4.84	1.65	1.88	3.70	2.94	2.42	**6.82**	4.21	3.59	2.28	4.28	7.21	8.75	4.92	1.44	1.41	3.81	6.28	2.71	**7.82**	4.24	
linSVM	2.38	1.23	4.56	7.67	2.78	3.69	1.45	1.72	2.92	3.72	1.46	14.45	3.05	1.67	1.75	5.70	7.67	2.78	4.43	1.66	1.25	3.81	3.42	2.33	**13.27**	3.32	
rbfSVM	2.68	1.40	6.56	6.05	8.35	2.79	1.50	2.03	4.15	4.94	1.77	**11.09**	3.84	1.87	1.58	2.28	7.67	8.75	4.51	1.66	1.41	3.59	6.32	2.23	**12.05**	3.81	
linLSSVM	3.12	1.93	5.99	7.91	1.99	4.43	1.13	1.56	3.81	3.03	2.39	**10.64**	3.39	3.27	2.28	6.27	7.91	2.78	4.43	1.39	1.88	3.48	3.03	2.36	**10.36**	3.55	
rbfLSSVM	2.53	2.10	5.99	7.91	8.75	3.61	1.47	1.56	4.26	5.93	2.29	**9.41**	4.22	3.15	2.98	2.85	6.98	8.75	4.51	5.78	1.41	3.81	6.23	2.21	**9.59**	4.42	
RIPPER	1.01	0.88	3.14	4.88	8.75	0.74	1.04	1.41	2.24	6.54	1.00	16.45	2.88	1.75	1.93	2.85	4.88	8.75	4.35	7.22	2.03	3.03	4.46	1.78	13.82	3.91	
PART	2.98	1.58	4.28	8.37	8.75	4.35	1.65	2.19	4.04	6.58	2.75	**7.09**	4.32	2.24	2.28	2.85	6.51	8.75	4.76	7.59	1.88	2.13	6.15	2.53	**10.05**	4.33	
C4.5	2.31	0.88	5.13	6.74	8.75	3.20	1.04	1.41	2.36	6.49	1.80	14.45	3.65	2.24	1.23	4.85	6.74	8.75	4.02	7.49	2.34	2.13	5.37	2.50	**11.23**	4.33	
CART	0.84	0.88	2.00	6.28	9.15	3.20	1.04	1.56	1.12	6.58	2.59	13.82	3.20	2.09	1.93	5.42	6.51	8.75	4.84	7.44	2.19	3.14	5.11	2.00	**10.73**	4.49	
ADT	3.04	2.45	5.70	6.05	9.15	4.84	1.64	2.19	4.15	5.97	2.81	**5.82**	4.36	3.25	2.28	5.70	6.74	9.15	4.26	1.69	2.50	4.04	5.97	2.75	**6.77**	4.40	
RF	2.75	3.51	5.13	6.98	8.75	4.92	7.58	2.50	3.48	6.80	1.94	**6.59**	4.94	2.78	3.68	6.56	6.98	8.75	4.84	7.63	1.56	4.26	6.71	1.94	**5.77**	5.06	
LMT	2.80	0.88	3.42	7.91	9.15	4.92	1.04	1.72	3.70	6.02	2.82	**8.91**	4.03	1.99	2.80	6.27	6.74	8.75	4.92	7.19	2.81	2.92	6.02	2.29	**7.82**	4.79	
Bag	3.22	3.16	4.85	6.51	8.75	4.92	4.56	1.72	3.70	6.71	2.82	**6.18**	4.63	2.85	2.10	4.85	6.74	8.75	4.84	7.41	1.88	2.92	6.80	1.88	**9.77**	4.64	
Boost	2.66	2.10	6.27	6.98	9.15	2.87	1.54	1.56	3.48	4.03	2.54	**9.95**	3.93	3.07	2.63	6.27	6.98	9.15	2.95	1.58	1.09	3.25	3.55	2.59	**9.86**	3.92	
RBFN	3.34	2.63	5.42	7.91	8.75	4.43	1.55	2.34	4.38	5.37	2.52	**5.68**	4.42	3.20	2.45	5.99	6.98	8.35	4.43	1.54	1.88	4.60	5.11	2.50	**8.68**	4.28	
VP	0.89	0.88	2.28	4.19	1.19	0.82	1.04	1.25	1.12	1.00	1.04	19.23	1.43	2.09	0.88	5.42	0.93	3.18	2.87	1.33	1.56	2.80	1.73	1.10	17.73	2.17	
Logit	3.39	1.40	5.99	7.91	3.18	4.67	1.40	1.88	3.48	3.42	2.35	**9.86**	3.55	3.59	2.45	6.27	7.21	3.18	4.67	1.38	1.72	3.59	3.29	2.27	**10.36**	3.60	
kNN10	2.19	1.75	5.13	7.44	8.75	4.76	2.20	1.88	3.37	5.54	2.24	**10.14**	4.11	2.18	3.16	5.13	7.67	8.75	4.84	7.09	1.88	3.81	5.93	2.30	**7.77**	4.79	
kNN100	3.00	1.75	5.13	7.44	8.75	4.43	2.20	1.88	3.37	4.59	2.44	**9.45**	4.09	2.73	2.10	5.42	6.51	8.75	4.76	3.57	1.25	4.15	5.19	2.56	**10.09**	4.27	
BN	3.15	1.75	5.42	7.67	9.15	4.51	1.47	1.88	4.38	4.55	2.93	**6.77**	4.26	2.50	2.63	5.70	7.21	8.75	4.59	7.19	1.88	3.93	4.89	2.88	**7.23**	4.74	
NB	3.17	2.28	6.27	7.91	6.36	4.43	1.48	2.03	4.49	4.11	2.39	**7.18**	4.08	3.17	2.10	6.27	7.91	6.76	2.87	1.48	2.03	4.49	4.11	2.39	**9.23**	3.96	
	Without oversampling													**With oversampling**													

Table 4.5: Results of the benchmarking experiment evaluated using the top decile lift performance criterion.

Data set	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD	AR	AA	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD	AR	AA
	Without input selection													**With input selection**												
NN	*73.0*	53.6	*72.4*	*90.0*	*55.2*	*84.0*	*54.2*	53.8	*62.2*	**89.3**	*67.8*	**8.1**	**68.7**	**75.5**	**61.8**	*76.0*	**86.5**	*67.3*	*81.5*	55.1	**54.5**	*68.3*	*83.8*	**71.2**	**6.4**	**71.0**
linSVM	*59.8*	*44.8*	*56.6*	*87.7*	*61.0*	*75.8*	*52.5*	*43.3*	*63.1*	*79.9*	*50.5*	*15.2*	*61.4*	*55.2*	*51.6*	**87.0**	**88.3**	*62.1*	*86.4*	**56.7**	**53.6**	*75.8*	*83.4*	*48.4*	**10.2**	**68.0**
rbfSVM	*67.5*	**58.3**	*82.6*	*88.1*	*63.9*	*75.9*	*59.1*	48.7	*64.3*	*88.8*	*58.3*	10.3	68.1	*61.6*	**57.0**	*50.0*	**89.3**	*73.0*	*50.4*	*54.0*	**54.0**	*59.6*	**90.6**	*57.7*	**10.5**	**63.4**
linLSSVM	*69.0*	**57.8**	**89.5**	*90.3*	*64.5*	*77.0*	*54.1*	49.8	*70.5*	*83.7*	*66.6*	**8.5**	**70.3**	*69.0*	*53.4*	**86.8**	**89.8**	*62.2*	**86.9**	**57.5**	**51.4**	*75.6*	*83.6*	*66.6*	**7.2**	**71.2**
rbfLSSVM	*70.3*	**63.5**	**89.4**	*88.4*	*74.1*	*60.2*	*53.0*	*49.8*	*63.0*	**89.9**	*64.4*	**8.5**	**69.7**	*69.2*	**58.7**	78.3	**89.7**	*67.2*	*62.2*	55.4	**53.7**	*63.5*	**89.7**	*65.2*	**7.7**	**68.4**
RIPPER	*50.6*	*50.0*	*55.4*	*63.0*	**91.8**	*50.8*	*50.0*	*50.0*	*55.4*	*86.9*	*50.0*	*14.5*	*59.4*	*54.4*	*51.3*	*57.0*	*61.4*	**89.1**	*53.6*	*50.5*	50.4	*54.4*	*78.7*	*59.7*	*16.3*	*60.0*
PART	*58.2*	53.4	*63.3*	*74.0*	79.5	*78.0*	55.2	*46.9*	*68.2*	*78.4*	*58.7*	*13.2*	*64.9*	*54.0*	*54.2*	*59.7*	*69.3*	85.8	*53.3*	*49.9*	**52.5**	*57.3*	*85.6*	*55.4*	*15.0*	*61.6*
C4.5	*56.4*	*50.0*	*50.0*	*60.6*	79.6	*50.0*	*50.0*	*50.0*	*55.6*	*82.5*	*57.1*	*15.7*	*58.3*	*53.1*	*55.2*	*59.4*	*70.6*	85.7	*56.6*	*49.2*	50.0	*57.0*	*82.7*	*55.2*	*15.8*	*61.3*
CART	*50.0*	*50.0*	*50.0*	*76.3*	**91.5**	*50.0*	*50.0*	*50.0*	*50.0*	*86.6*	*60.2*	*14.7*	*60.4*	*56.2*	*52.4*	*64.3*	82.3	**91.8**	*54.0*	*50.9*	*49.6*	*58.8*	*86.1*	*55.3*	*14.0*	*63.8*
ADT	*72.0*	**64.3**	83.9	**94.5**	**89.2**	**59.8**	**59.5**	*72.5*	*88.5*	**70.9**	**3.6**	**76.1**		*72.0*	**65.3**	**82.2**	**87.6**	**93.9**	**88.7**	**59.1**	**59.2**	*70.8*	**88.5**	**70.9**	**2.6**	**76.2**
RF	*66.0*	**59.6**	*79.4*	*82.1*	88.4	*63.9*	*49.9*	*47.7*	*63.4*	**90.4**	*62.2*	10.7	68.5	*66.1*	56.3	*72.0*	**88.3**	**92.3**	*67.3*	*52.1*	**54.7**	*66.8*	**89.5**	*63.0*	**8.2**	**69.8**
LMT	*50.0*	*50.0*	*50.0*	*83.2*	**87.7**	*50.1*	*50.0*	*50.0*	*50.0*	**90.0**	*68.0*	*12.9*	*61.7*	*59.2*	**56.1**	74.0	*77.9*	**93.5**	*74.2*	*49.2*	*45.0*	*54.1*	**89.2**	*60.0*	*12.1*	*66.6*
Bag	**74.4**	**64.4**	**90.3**	*81.2*	**96.4**	**90.2**	**59.1**	**61.8**	*71.0*	**91.8**	**71.3**	**2.2**	**77.5**	*67.1*	56.5	*73.9*	**86.3**	**93.3**	*66.4*	*53.6*	*46.3*	*64.1*	**90.3**	*65.3*	**9.1**	**69.4**
Boost	*68.3*	**64.2**	**86.6**	*82.8*	**95.0**	*81.5*	**59.9**	**59.6**	*69.8*	*85.0*	*68.4*	**4.9**	**74.6**	*69.5*	**63.3**	**88.0**	**87.3**	**94.7**	*81.5*	**59.9**	**57.4**	*69.9*	*83.7*	*68.8*	**3.8**	**74.9**
RBFN	*65.5*	*52.9*	*77.8*	83.7	*75.0*	*80.3*	55.5	**53.1**	*76.6*	*84.6*	*61.2*	**9.3**	**69.7**	*68.6*	*54.3*	**82.7**	*73.2*	*72.2*	*82.2*	*53.3*	**50.7**	**78.3**	*86.0*	*64.2*	**9.2**	**69.6**
VP	*50.0*	*50.0*	*50.0*	*50.0*	*50.0*	*50.0*	*50.0*	*50.0*	*50.0*	*50.4*	*51.2*	*18.0*	*50.1*	*65.9*	*54.5*	*48.6*	*52.2*	*56.2*	*80.8*	*52.6*	**58.1**	*50.0*	*62.3*	*53.7*	*14.9*	*57.7*
Logit	**74.7**	53.9	86.7	**91.0**	*62.8*	87.3	**57.2**	53.3	75.6	*84.6*	*68.2*	**5.8**	**72.3**	**75.3**	*54.7*	**87.4**	**86.2**	*62.1*	*76.0*	*62.5*	*51.9*	*47.8*	*64.0*	*83.5*	*59.4*	*12.9*	*64.7*
kNN10	*58.9*	**59.7**	*66.1*	*81.7*	*74.7*	*62.5*	*51.8*	*48.0*	*64.3*	*83.6*	*59.7*	*12.7*	*64.6*	*58.8*	**59.7**	*66.7*	*81.2*	*76.0*	*62.5*	*51.9*	*47.8*	*64.0*	*83.5*	*59.4*	*12.9*	*64.7*
kNN100	*65.6*	**63.6**	*75.3*	**87.6**	*67.6*	*77.3*	**56.5**	*44.5*	*76.6*	*80.8*	*64.1*	**9.9**	**69.0**	*64.2*	**61.7**	*73.7*	**87.8**	*72.1*	*76.9*	*53.8*	46.5	**75.5**	*82.4*	*63.6*	**10.3**	**68.9**
BN	*69.9*	**54.4**	**88.8**	**89.8**	92.7	**89.6**	**58.3**	**54.4**	**80.8**	*86.5*	*64.9*	**4.5**	**75.5**	*69.9*	54.8	*68.2*	*61.0*	87.2	*75.2*	*53.8*	**52.6**	*70.9*	*86.5*	*65.7*	**9.1**	**67.8**
NB	*69.9*	55.9	84.1	*86.4*	*75.2*	*79.3*	55.6	**53.9**	**80.9**	*83.4*	*67.1*	**6.9**	**72.0**	*66.3*	*54.3*	**84.1**	**87.0**	*74.9*	*79.2*	55.6	**54.0**	**79.8**	*80.8*	*65.7*	**8.0**	**71.1**
NN	**72.8**	**64.5**	**90.3**	**91.0**	**50.4**	*87.3*	**59.4**	**60.6**	76.8	*82.5*	*66.5*	**7.6**	**72.9**	**72.8**	**64.8**	**88.7**	**88.0**	**94.8**	**87.6**	**59.1**	**60.6**	**77.0**	**91.7**	**71.2**	**4.4**	**77.8**
linSVM	*65.3*	**59.5**	*84.8*	**92.0**	*46.2*	*84.1*	**57.2**	**59.3**	*71.9*	*82.8*	*63.1*	*13.3*	*69.7*	*67.0*	**61.4**	**88.4**	**92.5**	*67.9*	85.8	**59.5**	**57.7**	76.6	*83.4*	*65.9*	**10.9**	**73.3**
rbfSVM	*67.3*	**62.6**	**89.1**	**83.9**	**45.8**	*83.1*	**59.8**	**59.1**	*67.8*	*89.5*	*62.8*	*12.1*	*70.1*	*60.8*	**64.4**	*82.8*	**90.5**	*95.6*	85.7	**55.0**	**58.7**	76.3	**90.8**	*68.2*	**9.7**	**75.3**
linLSSVM	*69.1*	**63.3**	89.3	**92.5**	*44.0*	*85.1*	*54.2*	**63.8**	*76.8*	*83.2*	*65.8*	**10.3**	**71.6**	*71.1*	**62.8**	**88.8**	**93.0**	*65.4*	*85.5*	**59.1**	**59.6**	77.0	*83.2*	*65.6*	**9.7**	**73.8**
rbfLSSVM	*66.1*	**64.5**	**90.3**	**93.3**	*55.5*	*84.4*	*54.9*	**63.6**	76.3	**90.9**	*68.1*	**7.6**	**73.4**	*70.6*	**70.1**	**87.0**	**88.6**	94.6	86.6	**54.8**	**63.1**	*71.2*	**90.8**	*65.5*	**8.6**	**76.6**
RIPPER	*50.8*	*50.0*	*55.3*	*71.0*	**93.8**	*50.0*	*50.0*	*50.8*	*58.1*	*87.5*	*50.8*	*17.3*	*60.7*	*64.1*	**63.4**	*76.2*	*80.1*	**93.8**	85.5	*51.3*	**55.1**	*74.0*	*85.4*	*66.8*	*15.0*	*72.3*
PART	*70.9*	**63.9**	82.4	**89.0**	**95.4**	86.6	**60.0**	**59.4**	**77.9**	*88.7*	*70.1*	**7.5**	**76.8**	*64.3*	60.6	*76.2*	88.1	**93.5**	**87.4**	*52.0*	**56.8**	*71.5*	*87.5*	*68.5*	12.6	73.3
C4.5	*57.0*	*50.0*	*70.6*	*80.7*	**94.8**	*82.3*	*50.0*	*50.8*	*58.9*	*88.2*	*63.4*	*15.8*	*67.9*	*62.5*	*59.3*	*65.6*	*80.5*	**93.9**	*86.0*	*52.0*	**55.6**	*62.6*	*84.3*	*68.5*	*15.9*	*70.1*
CART	*50.0*	*50.0*	*50.0*	*78.0*	*94.9*	*82.3*	*50.0*	**54.2**	*50.0*	*88.4*	*67.2*	*15.9*	*65.0*	*55.8*	60.0	*71.7*	88.1	**94.4**	*87.2*	*51.3*	54.9	*60.7*	*85.3*	*65.8*	*15.4*	*70.5*
ADT	71.9	**65.4**	**87.7**	*86.9*	**97.2**	*87.3*	**60.4**	**59.2**	**79.3**	*88.5*	**71.1**	**5.3**	**77.7**	70.6	**65.6**	**87.7**	88.3	**97.1**	86.3	**61.0**	**56.8**	**77.3**	*88.5*	**71.2**	**6.4**	**77.3**
RF	*64.6*	**63.3**	*78.6*	**89.6**	**94.9**	**87.7**	*54.7*	**58.7**	*69.1*	90.6	*65.6*	**10.5**	**74.3**	*65.6*	**63.7**	79.6	87.3	**94.6**	**87.0**	**54.8**	**58.8**	*71.2*	**91.4**	*65.7*	**11.2**	**74.5**
LMT	*69.9*	*50.2*	*52.4*	**90.8**	**95.5**	*87.3*	*50.0*	**54.7**	*73.7*	**90.7**	**71.8**	**9.4**	**71.5**	*63.0*	**65.6**	**88.7**	**87.8**	**95.2**	*87.3*	**57.1**	**62.8**	*73.7*	**90.3**	*66.6*	**7.5**	**76.2**
Bag	**73.6**	**69.7**	**89.6**	87.5	**97.0**	*87.3*	**60.5**	**60.4**	**78.5**	**91.8**	**71.2**	**2.7**	**78.8**	*67.5*	61.0	82.2	**89.6**	**94.1**	*87.2*	**55.8**	**61.5**	*67.3*	**90.6**	*65.4*	**10.8**	**74.7**
Boost	*67.8*	**64.4**	**88.0**	*84.9*	**95.1**	*81.1*	**59.9**	**54.7**	77.8	*85.9*	*68.8*	**10.5**	**75.3**	*70.0*	**64.8**	**88.0**	*84.9*	**95.1**	*81.9*	**60.0**	**55.7**	76.2	*83.7*	*69.5*	**10.0**	**75.4**
RBFN	**71.8**	**64.6**	**87.1**	**90.6**	93.2	86.1	**58.4**	**59.0**	**79.2**	*88.9*	*69.4*	**7.8**	**77.1**	71.7	*59.8*	**88.4**	**92.4**	91.2	85.7	**58.9**	**57.8**	**81.8**	*89.0*	*69.8*	**8.0**	**77.0**
VP	*50.3*	*50.0*	*51.4*	*66.3*	*52.0*	*50.4*	*50.0*	*50.0*	*50.0*	*50.8*	*53.0*	*18.9*	*52.2*	*66.6*	*55.2*	**86.9**	*61.5*	*67.9*	*80.9*	**57.5**	**57.1**	*67.8*	*68.3*	*53.4*	*16.6*	*65.7*
Logit	**72.8**	62.7	**88.7**	**93.1**	*68.9*	**86.9**	**59.2**	**60.4**	77.6	*83.7*	*66.5*	**8.3**	**74.6**	**73.1**	**62.4**	**89.5**	**92.2**	*68.3*	**86.7**	**59.6**	**60.3**	77.0	*83.8*	*66.3*	**7.7**	**74.5**
kNN0	*62.5*	**65.9**	**86.5**	88.8	**93.8**	**87.1**	**57.6**	**63.3**	**78.2**	*89.0*	*70.3*	**7.5**	**76.6**	*62.4*	**65.8**	**87.6**	**90.2**	**94.4**	**87.2**	**55.3**	**58.7**	**78.0**	*89.1*	*67.9*	**8.4**	**76.1**
kNN100	*57.8*	**65.9**	**86.5**	88.8	93.8	*86.1*	**57.6**	**63.3**	**78.2**	*89.0*	*71.1*	**8.0**	**76.2**	*68.1*	**65.1**	**87.1**	**89.0**	**94.5**	86.9	**56.3**	**61.5**	**78.4**	**90.7**	*70.0*	**6.5**	**77.1**
BN	71.6	58.4	**89.4**	90.3	**97.0**	**86.3**	**60.0**	**55.5**	**81.7**	*84.0*	**71.6**	**6.7**	**76.9**	*68.1*	**63.4**	**89.6**	**89.1**	**96.3**	86.1	**57.4**	**60.6**	**79.7**	*83.2*	**71.4**	**6.6**	**76.8**
NB	**71.8**	**64.7**	**88.6**	**92.1**	*90.8*	*86.1*	**58.7**	**59.5**	**81.8**	*86.9*	*68.8*	**7.1**	**77.3**	71.8	**64.1**	**88.6**	**92.0**	*90.9*	*81.1*	**58.7**	**59.5**	**81.8**	*86.9*	*68.8*	**7.9**	**76.7**
	Without oversampling													**With oversampling**												

Table 4.6: Results of the benchmarking experiment evaluated using the AUC performance criterion.

Input selection

The results of the experiments with and without input selection can be found respectively in the lower and upper panels of Tables 4.4, 4.5, and 4.6. Applying the Friedman test to compare the results for each measure yields a p-value around zero, both when including results with and/or without oversampling. This indicates that classifiers yield a significantly better predictive performance when applying input selection. At first sight this result might seem somewhat counterintuitive. However, it makes sense that it is easier to learn from a smaller data set with few, yet highly predictive, variables, than from an extensive set containing much redundant or noisy data. This result indicates that it is crucial to apply an input selection procedure in order to attain good predictive power. Moreover, a model containing less variables is advantageous as it will be more stable, since collinearity is reduced. More importantly from a practical point of view, a concise model is also easier to interpret, since the number of variables included in the model is minimized. Figure 4.7 plots the results of the input selection procedure on data set O1 for logistic regression, which is exemplary for most techniques and data sets. The Y-axis represents the performance measured in AUC, and the number of variables included in the model is plotted on the X-axis. From this figure, it can be seen that adding a variable improves the performance of the logit model dramatically when only few variables are included in the model (i.e., on the left side of the figure the curve has a steep, positive slope).

However, the positive effect of adding extra variables flattens at eight variables, then reaches a maximum at nine variables, and when including more than twelve variables the performance decreases. The optimal trade-off between the number of attributes that is included in the model and the predictive performance as required by a business expert lies at eight variables. Selecting less variables yields poor predictive power, while including more variables makes the model harder to interpret and adds only little predictive power. Figure 4.7 can also be interpreted starting from the right side, with many variables included in the data set. At first removing variables improves the performance since mostly redundant attributes will be removed. When too much information is filtered from the data however, the performance drops.

In Figure 4.8 a boxplot summarizes the number of variables of the different data sets that is used by the eight best performing techniques according

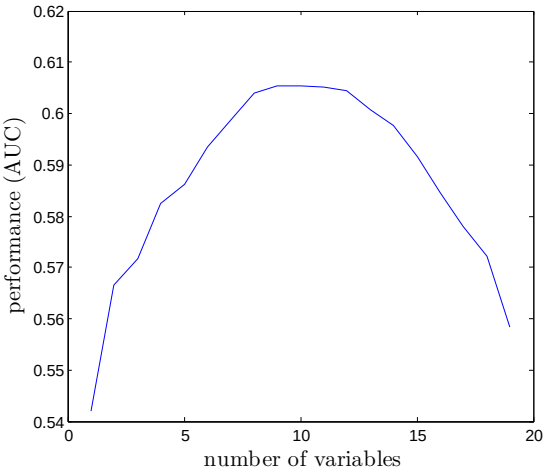


Figure 4.7: Performance evolution of the input selection process for logistic regression applied on data set D2.

to the MP criterion. On each box, the central mark indicates the median number of variables, the edges of the box represent the 25th and 75th percentiles, the whiskers extend to the most extreme numbers that are not considered to be outliers, and outliers finally are plotted by crosses. ADT, NB, and C4.5 appear to be very efficient algorithms, which are able to produce powerful models with only a very small number of attributes. The number of variables needed by RF, NN, PART, and LMT on the other hand seems to be heavily dependent on the data (i.e., the boxes and whiskers are spread over a wide range). On average, these eight techniques only need around 6 or 7 variables to yield optimal performance.

This means that a surprisingly small number of variables suffices to build an effective and powerful CCP model. Hence from an economical point of view, to build or improve the predictive power of such a model, it is operationally more efficient to focus on collecting and improving the quality of a small set of well chosen attributes, than to collect as much data as possible. Data quality is an important issue in any data mining context, and generally speaking the return on investment to improve data quality depends on the structure of the processes within an organization. For instance, data entry processes should be carefully designed to avoid outliers and/or missing values. Section 4.6.3 will further analyze the results

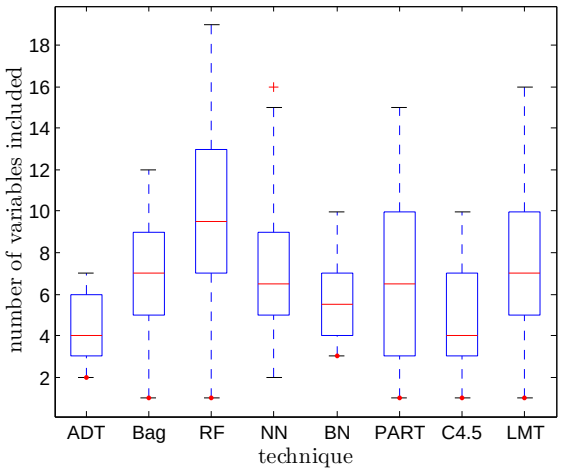


Figure 4.8: Boxplot of the number of variables used by the eight best performing techniques.

of the input selection experiments to indicate which type of information is required to predict customer churn.

Oversampling

For each of the performance criteria, when applying the Friedman test to compare the results with and without oversampling on an aggregate level, no significant impact on performance is detected. Hence, no clear conclusion can be drawn about the impact of oversampling on the performance of a CCP model.

The test of DeLong, DeLong, and Clarke-Pearson is applied to compare the performance of each classification technique with and without oversampling, for the results with input selection measured in AUC, which can be found in the right and the left lower panels of Table 4.6. The resulting p-values are reported in Table 4.7, indicating the probability that a difference in performance is due to chance. The p-values smaller than 0.05 and 0.01 indicate that a difference in performance is significant with a confidence level of respectively 95% and 99%, and are tabulated in normal and italic script. Furthermore, when the effect of oversampling on the performance of a classification technique is found to be significant, but negative, then the reported p-value is underlined. Non-significant differences finally are tabu-

lated in bold. The image of Table 4.7 is rather diffuse, since there seem to be as many positive as negative significant effects on performance, and in many cases the results are not significantly different.

Data set	O1	O2	O3	O4	O5	O6	D1	D2	D3	UCI	KDD
NN	**0.955**	**0.952**	**0.407**	**0.274**	**0.300**	0.012	**0.139**	**0.704**	**0.945**	*0.000*	*0.000*
linSVM	*0.003*	**0.928**	*0.007*	**0.694**	**0.718**	0.036	**0.133**	**0.643**	*0.004*	**0.616**	*0.000*
rbfSVM	*0.000*	**0.705**	**0.152**	0.013	**0.444**	*0.000*	0.026	**0.882**	0.022	**0.051**	*0.000*
linLSSVM	*0.000*	**0.869**	**0.643**	**0.339**	**0.670**	**0.183**	*0.000*	**0.365**	**0.853**	**0.549**	**0.730**
rbfLSSVM	*0.000*	**0.116**	**0.130**	**0.053**	*0.000*	*0.000*	*0.000*	**0.924**	**0.068**	**0.396**	*0.002*
JRIP	*0.000*	*0.000*	*0.000*	0.019	**0.979**	*0.000*	*0.000*	**0.090**	*0.000*	**0.071**	*0.000*
PART	*0.000*	**0.367**	0.043	**0.661**	**0.283**	**0.071**	*0.000*	**0.580**	0.011	**0.293**	0.015
C4.5	*0.000*	0.013	**0.308**	**0.942**	**0.626**	*0.000*	*0.000*	0.021	**0.284**	*0.002*	*0.000*
CART	*0.000*	0.011	*0.000*	*0.002*	*0.000*	*0.000*	*0.000*	**0.849**	*0.000*	*0.006*	*0.000*
ADT	**0.137**	**0.937**	**1.000**	**0.196**	**0.444**	**0.051**	**0.919**	**0.373**	**0.220**	**1.000**	**0.617**
RF	**0.426**	**0.901**	**0.844**	**0.305**	**0.897**	0.023	**0.664**	**0.991**	**0.422**	**0.276**	**0.296**
LMT	*0.000*	*0.000*	*0.000*	**0.388**	**0.886**	*0.000*	*0.000*	**0.140**	**0.983**	**0.762**	*0.000*
Bag	*0.000*	**0.058**	0.011	**0.386**	**0.268**	**0.598**	*0.000*	**0.783**	*0.000*	**0.193**	*0.000*
Boost	0.019	**0.495**	**1.000**	**0.987**	**1.000**	**0.066**	**0.056**	**0.613**	**0.184**	0.013	**0.116**
RBFN	**0.839**	**0.169**	**0.338**	**0.214**	**0.156**	**0.295**	**0.052**	**0.773**	**0.090**	**0.851**	**0.374**
VP	*0.000*	0.044	*0.000*	**0.201**	*0.001*	*0.000*	*0.000*	**0.110**	*0.000*	*0.000*	**0.337**
Logit	**0.133**	**0.909**	**0.269**	**0.206**	**0.277**	**0.319**	**0.684**	**0.531**	**0.488**	**0.741**	**0.670**
IBK10	**0.957**	**0.985**	**0.701**	**0.552**	**0.821**	**0.319**	*0.000*	**0.232**	**0.891**	**0.760**	**0.499**
IBK100	**0.329**	**0.835**	**0.643**	**0.966**	*0.003*	**0.139**	*0.000*	**0.593**	**0.796**	0.020	*0.000*
BN	*0.000*	**0.234**	**0.850**	**0.714**	**0.591**	**0.371**	*0.000*	**0.179**	**0.223**	**0.551**	**0.489**
NB	**0.359**	**0.351**	**0.794**	**0.260**	**0.398**	*0.000*	**0.939**	**0.217**	**0.971**	**0.303**	**0.146**

Table 4.7: The resulting p-values of the DeLong, DeLong, and Clarke-Pearson test applied to compare the performances of the classification techniques with and without oversampling, with input selection, on each data set separately. Performances that are not significantly different at the 95% confidence level are tabulated in bold face. Significant differences at the 99% level are emphasized in italics, and differences at the 95% level but not at the 99% level are reported in normal script. If the performance without oversampling is significantly better than the result with oversampling, the p-value is underlined.

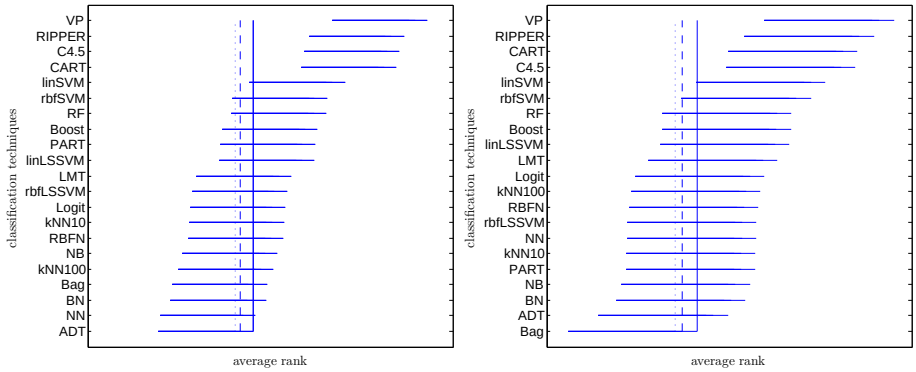
As illustrated by Table 4.7, the effect of oversampling strongly depends on the data set and the technique that is applied. For instance, oversampling improves the performance of Ripper on 8 out of the 11 data sets, while the results of ADT and RBFN are never found to be significantly impacted. These last techniques are apparently able to learn properly even with a very skewed class distribution. Furthermore, differences in performance on data sets $O4$ and $D2$ are almost never found significant, and in case of data set $D1$, oversampling yielded a positive effect in 6 cases, a negative

effect in 7 cases, and no significant effect in 8 cases, which illustrates the apparent randomness in the effect of oversampling on the predictive power. Therefore it is recommended to adopt an empirical approach when building a CCP model, and to consistently test whether oversampling provides better classification results or not.

Classification techniques

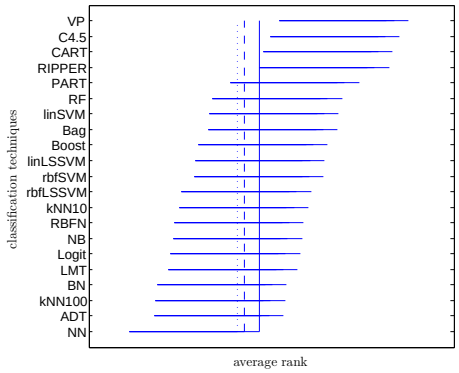
In the previous paragraphs input selection is found to be crucial in order to obtain good predictive power. Therefore only the aggregate results with input selection (i.e., the results in the lower two panels of Tables 4.4, 4.5, and 4.6) are included to compare the classification techniques using the Friedman and post-hoc Nemenyi tests. The Friedman test results in a p-value close to zero for each of the three performance measures, and both with and/or without oversampling, indicating significant differences in performance to exist among the applied techniques. We thus proceed by performing post-hoc Nemenyi to compare all classifiers, as explained in Section 4.5.1. The results are plotted in Figures 4.9(a) to 4.9(c), respectively for the maximum profit criterion, top decile lift, and AUC. The horizontal axes in these figures represent the average ranking of a technique on all data sets. The more a technique is situated to the left, the better its ranking. Techniques are represented by a line, the left end of this line depicts the actual average ranking, while the line itself represents the critical distance for a difference between two classifiers to be significant at the 99% confidence level. The dotted, dashed, and full vertical lines in the figure indicate the critical differences with the best performing technique at respectively the 90%, 95%, and 99% confidence level. A technique is significantly outperformed by the best technique if it is situated at the right side of the vertical line.

The specific differences between the rankings in Figures 4.9(b) to 4.9(c) will be discussed in detail in the next section. In general, we can conclude from Figures 4.9(a) to 4.9(c) that a large number of techniques do not perform significantly different. Although the reported results in Tables 4.4, 4.5, and 4.6 are widely varying, the majority of techniques yield classification performances which are on average quite competitive to each other. The conclusions of this section are comparable to previous benchmarking studies in credit scoring (Baesens et al., 2003b) and software defect prediction (Lessmann et al., 2008), which also reported a flat maximum effect and a limited



(a) Aggregate, with input selection.

(b) Without oversampling, with input selection.



(c) With oversampling, with input selection.

Figure 4.9: Ranking of classification techniques, the dotted vertical line indicates the 90% significance level, the dashed line the 95% level, and the full line the 99% level.

difference in predictive power between a broad range of classification techniques. Hence, the impact of the classification technique on the resulting performance is less important than generally assumed. Therefore, depending on the setting other aspects beside discriminatory power have to be taken into account when selecting a classification technique, such as for instance comprehensibility or operational efficiency (Martens et al., 2011). In many business settings, a comprehensible model will often be preferred over a better performing black box model, since an interpretable model al-

lows the marketing department to learn about customer churn drivers, and provides actionable information to set up retention initiatives, as will be discussed in Section 4.6.3. Furthermore, comprehensibility allows to check whether the classifier functions intuitively correct and in line with business knowledge. The most interpretable types of models are rule sets and decision trees, but also logistic regression and Bayesian techniques result in comprehensible classification models. Neural networks or SVMs on the other hand result in complex, non-linear models, which are very hard to interpret and therefore called black box models. However, a combination of comprehensibility and good predictive power can also be achieved indirectly, by adopting a hybrid approach such as for instance rule-extraction from neural networks (Baesens et al., 2003a). The operational efficiency concerns the ease of implementation, execution, and maintenance of a model, which are represented by steps three and four in the data mining modeling process discussed in Section 4.2. Rule sets and linear models are very fast in execution, and easy to implement and maintain. Nearest neighbor methods on the other hand do not result in a final model that can be implemented and executed straightforwardly, and have to calculate the k nearest neighbors each time a customer needs to be classified. Ensemble methods on the other hand involve a multitude of models that need to be executed, implemented, and maintained, and therefore typically score bad on this aspect.

Finally, Tables 4.4, 4.5, and 4.6 provide a benchmark to CCP modeling experts in the industry to compare the performance of their CCP models.

Statistical performance measures versus the maximum profit criterion

Figure 4.10 compares the rankings of the classification techniques in terms of maximum profit, top decile lift, and AUC. For each performance measure, the techniques that are not significantly different at the 95% confidence level according to the Bonferroni-Dunn test are grouped within grey boxes. As can be seen from the figure, the rankings vary substantially over the different performance measures, but show as well some resemblances. At the top, ADT is best using MP and AUC, and second best using top decile lift, which indicates that this technique has an overall good performance. RF on the other hand, having the best top decile lift and third in terms of MP, is only classified fifteenth using AUC. RF apparently performs well regarding the customers it assigns the highest propensities to attrite, resulting in the

best top decile lift. However, when taking into account the entire ranking of the customers the performance of RF declines sharply, as indicated by the AUC measure.

At the bottom of the rankings a number of techniques seems to perform bad for all three measures, such as for instance VP, rbfSVM, and linSVM. The bad performance of the latter two techniques is rather surprising, given the competitive performance results reported in the literature in other domains. However, this can be due to the fact that these classifiers had to be trained using smaller samples, possibly leading to poor discriminatory power. On the other hand, a remarkable difference in ranking exists regarding the rule induction techniques and decision tree approaches. Evaluating C4.5, RIPPER, CART, and PART using the maximum profit criterion yields average (RIPPER and CART) to good performance (C4.5 and PART). However, in terms of top decile lift or AUC, all except for PART are found to be significantly outperformed by the best performing technique. This can be explained by the fact that rule sets and decision trees do not provide a continuous output, and therefore their ROC curve has only as many points as there are rules or leaves, resulting in a discontinuous, piecewise monotone ROC curve and a fairly low AUC. Furthermore, these models only classify a fraction of the customers to be churners approximately equal to the fraction of churners in the data set, i.e., the base churn rate β_0. The base churn rate is in most data sets below 10% as indicated by Table 4.3, and therefore these techniques yield poor top decile lift. The optimal fraction of customers to include in a retention campaign however usually lies more near to the base churn rate. For instance, the average optimal fraction for technique PART on the data sets with input selection, both with and without oversampling, is equal to 3.38%, for C4.5 3.11%, for CART 3.45%, and for RIPPER 2.73%. The average over all techniques lies at 4%. Therefore the maximum profit criterion allows a more fair comparison regarding rule sets and decision trees.

Figure 4.11 shows the average profit per customer over the eleven data sets, to illustrate the impact on the resulting profits of selecting a CCP model using each of the three performance measures. Both for top decile lift and AUC the resulting profit depends on the fraction of customers that is included in the retention campaign, whereas the MP criterion automatically determines the optimal fraction of customers to include and results in the maximum profit. Therefore, the average profit per customer generated

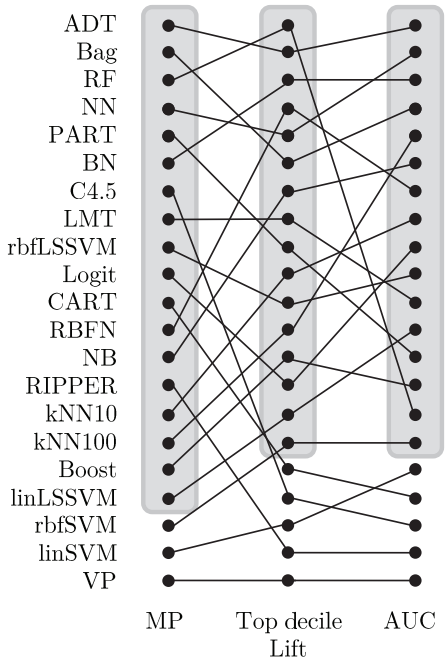


Figure 4.10: Comparison of the rankings of classification techniques resulting from the benchmarking experiment using the maximum profit criterion, top decile lift, and AUC. The techniques that are not significantly different at the 95% confidence level according to a post hoc Nemenyi test, are grouped in the grey boxes for each performance measure.

by using the MP criterion is shown as a constant function in Figure 4.11, represented by a dotted line and equal to 0.9978. The grey-most area between this function and the dash-dotted horizontal line below, indicating the maximum average profit using top decile lift, represents the difference in profit per customer resulting from suboptimal model selection using top decile lift. This difference equals $0.9978 - 0.5321 = 0.4677$ per customer. In this setting, for an operator with a customer base of one million customers, the difference in profit due to suboptimal CCP model selection yields half a million euros per retention campaign. On top of this, an additional difference in profit per customer will exist if a suboptimal fraction of customers is included in the retention campaign, indicated by the middle-grey area between the lower horizontal line and the top decile lift profit curve. For

instance, if a model is selected using top decile lift, and the top decile of customers is effectively included in the retention campaign, then the difference in profit per customer amounts to $0.4677 + 0.0352 = 0.5321$.

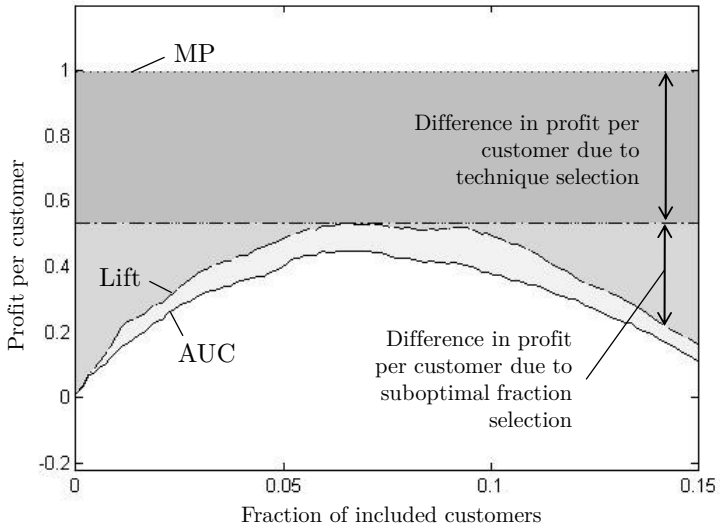


Figure 4.11: Average profit per customer using maximum profit (dotted line), lift (dashed line), and AUC (full line).

4.6.3 Customer churn drivers and managerial insights

Four types of attributes can be identified in the data sets[2]:

- *Socio-demographic data*: contains personal information about customers such as age, gender, or zip code.
- *Call behavior statistics* or *usage attributes*: are mostly summary statistics such as number of minutes called per month, or variables capturing the evolution or trends in call behavior such as increase or decrease

[2]The variables in data set KDD are anonymized and cannot be interpreted, and UCI is a synthetic data set which is only included in the study to allow comparison with previous and future studies. Therefore the input selection results of these data sets are not included in this analysis.

in number of minutes called. Usage attributes are purely related to the actual consumption.

- *Financial information*: contains billing and subscription information. Examples are average monthly revenue, revenue from international calls, type of plan chosen, and credit class of a customer.

- *Marketing related variables*: contain information about interactions between the operator and the customer, such as promotions that are offered to a customer, calls from the retention team to a customer, but also calls from the customer to the helpdesk. Marketing related variables are of particular interest to learn about how customers can be retained (Bolton et al., 2006).

As a result of the input selection procedure, we are able to identify which type of data is most important to predict churn. The variables selected by the best performing techniques during the input selection procedure are analyzed and binned into the four categories. This results in the pie charts shown in Figure 4.12, which represent the percentage of the selected variables belonging to each of the four types. Pie charts are shown for each data set separately, and on an aggregate level. Not all data sets include as many attributes of each type, resulting in some variance between the charts. However, as can be seen from the figure, most charts are similar to the aggregate chart. Therefore we can conclude that usage variables are the most solicited type of variable, and seem to be the best predictors of future churn. The other three types of data are used almost equally, each category representing roughly twenty percent of the selected variables. Hence, none of the four categories can be excluded from the data, and complete data sets containing information on each type will tend to yield better predictive performance.

The attributes present in the data sets used in this chapter are rather diverse, nevertheless a number of interesting findings can be reported regarding specific variables that are relevant to predict churn. As mentioned, marketing variables are of specific interest to the marketing department since they provide actionable information. Attributes related to the hand sets provided by the operator to the customer, such as the price and age of the current equipment, generally seem to be very relevant in order to predict churn. Also the number of contacts between operator and customer is typically a good predictor. Concerning socio-demographic variables, the

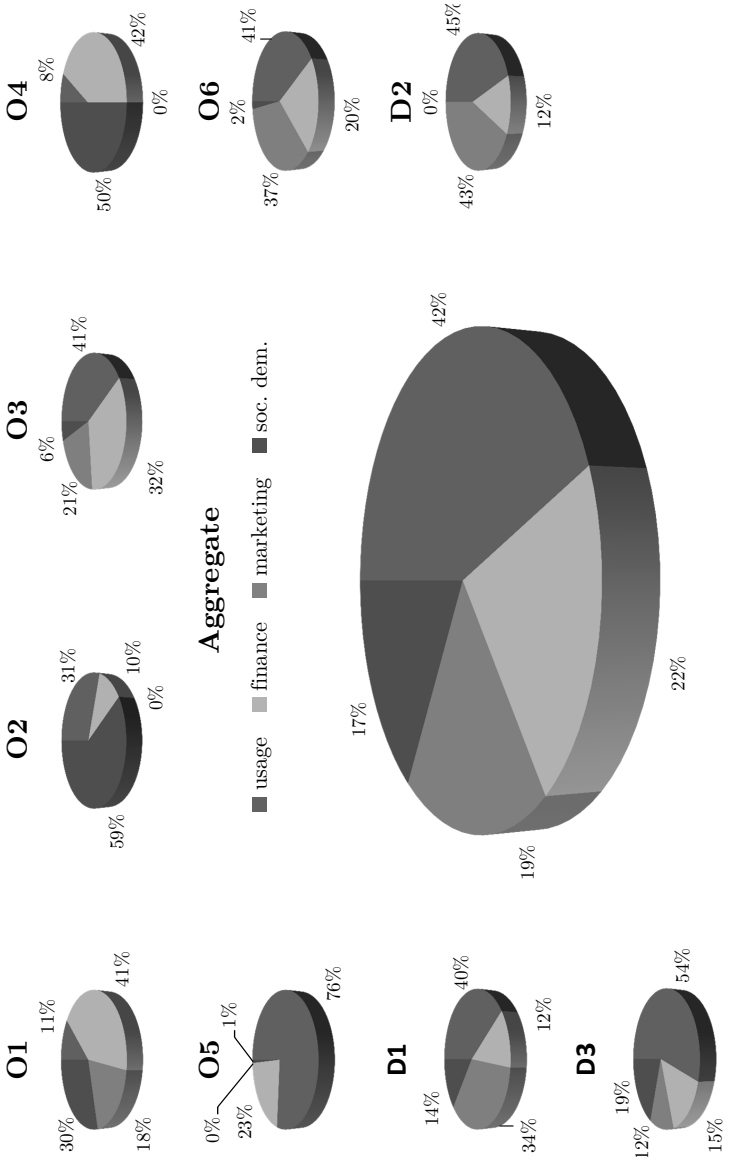


Figure 4.12: Pie charts of the type of variables selected by the best performing techniques.

age of a customer turns out to have good predictive power, but zip code or similar information on the other hand not at all, as might be expected. Examples of often selected financial variables are mean total monthly recurring charge, the type of plan, and the credit class of a customer. Usage statistics that are often solicited are the mean number of outbound voice calls (to a specific competitor), or even simply the total number of call detail records. Trend variables such as recent to early usage ratio, this month to last months usage ratio, or simply an indicator of a positive or negative evolution seem to be frequently selected as well.

A typical issue when selecting variables is whether an attribute is a predictor for future churn, or a symptom of occurring churn. For instance, as a result of the modeling process a usage attribute indicating a strong drop in total minutes called might show to be strongly correlated with churn. However, this drop is likely to occur post-factum, when the event of churn has already taken place but is not logged in the data yet. Also a sudden peak in usage just before churn can typically be observed in churn data sets, and might therefore be considered a good predictor. This peak however usually indicates that the customer already decided to change from operator, and is consuming all the remaining minutes he already paid for. Therefore, such attributes cannot be used to predict customer churn since they do not allow to give an early warning, preferably even before the customer is actively considering to attrite. To successfully retain customers an early detection is of crucial importance in order to allow the marketing department to act on the results of the model, before the customer has made a final decision. This problem can partially be solved by lagging the data sufficiently to the churn event indicator. The lead of the churn flag on the attributes in the data sets included in this chapter is at least one month. A lead of more than three months on the other hand is expected to result in weak predictive power. Finally, a model always has to be checked and interpreted by a business expert to validate the selection of predictive variables, which again illustrates the need for a comprehensible model.

4.7 Conclusions and future research

CCP models are typically evaluated using statistically based performance measures, such as for instance top decile lift or AUC. However, as shown in Sections 4.3 and 4.6 of this chapter, this can lead to suboptimal model

selection and a loss in profits. Therefore, in the first part of this chapter a novel, profit centric performance measure is developed. Optimizing the fraction of included customers with the highest predicted probabilities to attrite yields the maximum profit that can be generated by a retention campaign. Since reducing the cost of churn is the main objective of CCP models, this chapter advocates that the maximum profit should be used to evaluate CCP models.

In the second part of the chapter a large benchmarking experiment is conducted, including twenty-one state-of-the-art predictive algorithms which are applied on eleven data sets from telco operators worldwide, in order to analyze the impact of classification technique, oversampling, and input selection on the performance of a CCP model. The results of the experiments are tested rigorously using the appropriate test statistics, and evaluated using both the novel profit centric based measure and statistical performance measures, leading to the following conclusions.

Applying the maximum profit criterion and including the optimal fraction of customers in a retention campaign leads to substantially different outcomes. Furthermore, the results of the experiments provide strong indications that the use of the maximum profit criterion can have a profound impact on the generated profits by a retention campaign.

Secondly, the effect of oversampling on the performance of a CCP model strongly depends on the data set and the classification technique that is applied, and can be positive or negative. Therefore, we recommend to adopt an empirical approach, and as such to consistently test whether oversampling is beneficial.

Third, the choice of classification technique significantly impacts the predictive power of the resulting model. Alternating Decision Trees yielded the best overall performance in the experiments, although a large number of other techniques were not significantly outperformed. Hence, other properties of modeling techniques besides the predictive power have to be taken into account when choosing a classification technique, such as comprehensibility and operational efficiency. Rule induction techniques, decision tree approaches, and classical statistical techniques such as logistic regression and Naive Bayes or Bayesian Networks score well on all three aspects, and result in a powerful, yet comprehensible model that is easy to implement and operate. Therefore these techniques are recommended to be applied for CCP modeling. Comprehensibility or interpretability is an important aspect

of a classifier which allows the marketing department to extract valuable information from a model, in order to design effective retention campaigns and strategies. The comprehensibility of a model however also depends on the number of variables included in a model. Clearly a model including ten variables is easier to interpret than a model containing fifty variables or more.

This leads to a fourth conclusion, i.e., input selection is crucial to achieve good performance, and six to eight variables generally suffice to predict churn with high accuracy. Consequently, from an economical point of view it is more efficient to invest in data quality, than in gathering an extensive range of attributes capturing all the available information on a customer. Furthermore, the input selection procedure has shown that usage attributes are the most predictive kind of data. However, also socio-demographic data, financial information, and marketing related attributes are indispensable sources of information to predict customer churn. Moreover, marketing related attributes such as the hand set that is provided to a customer by the operator, are important sources of actionable information to design effective retention campaigns. Finally, this chapter also provides benchmarks to the industry to compare the performance of their CCP models.

As a topic for future research, a fifth type of information remains to be explored on its ability to predict churn, i.e., social network information. Call Detail Record data is usually present in abundance, and can be analyzed to extract a large graph, representing the social network between the customers of an operator. Initial results of a pilot study indicate that social network effects play an important role in customer churn (Dasgupta et al., 2008). A model that incorporates these effects as an extra source of information therefore promises to yield improved performance.

Chapter 5

Social network analysis for customer churn prediction

Everything touches everything.

Jorge Luis Borges (1899 - 1986)

Abstract

This chapter contributes from a theoretical perspective by proposing a range of new and adapted relational learning algorithms for customer churn prediction using social network effects, designed to handle large scale networks, a time dependent class label, and a skewed class distribution[1]. Furthermore, an innovative approach to incorporate non-Markovian network effects within relational classifiers is presented, and a novel parallel modeling setup is introduced to combine a relational and non-relational classification model. From an application perspective, a new profit driven evaluation methodology is applied to assess the results of two real life case studies on large scale telco data sets, containing both networked (call detail record data) and non-networked (customer related) information about millions of subscribers. A

[1]Verbeke, W., Martens, D., Baesens, B., 2011d. Social network analysis for customer churn prediction. Management Science, under review

significant impact of social network effects, including non-Markovian effects, on the performance of a customer churn prediction model is found, boosting the profits generated by a retention campaign.

5.1 Introduction

In recent years, vast amounts of networked data on a broad range of network processes and information flows between interlinked entities have become available, such as calls and text messages linking telephone accounts (Dasgupta et al., 2008) and money transfers connecting bank accounts (Martens and Provost, 2011). These massive, networked data logs potentially hide information that is highly valuable to companies and organizations, and as such open new perspectives for innovative business applications (Bonchi et al., 2011).

Networked data present both complications and opportunities for predictive data mining. The data are patently not independent and identically distributed, which introduces bias to learning and inference procedures (Jensen and Neville, 2002; Macskassy and Provost, 2007). Relational learning aims to exploit the information contained within the network structure of data instances, and to incorporate this information within a network classification or regression model (Džeroski and Lavrač, 2001; Getoor and Taskar, 2007). Relational classifiers (RC) learn directly from a graph or network, as opposed to non-relational classifiers (N-RC) which require an attribute-value representation of the data.

The central research question in this chapter consists of a *theoretical* and an *applied* component, concerning respectively the *use* and the *merits* of social network information for customer churn prediction in the telco industry. More specifically, this chapter develops a number of approaches to mine social network information for churn prediction, and examines the effect on the predictive power of the resulting models. Customer churn undermines the profitability of telco operators, facing annual churn rates up to 20% and higher. Therefore customer relationship management, and more specifically customer retention, receives a growing amount of attention from telco operators. Customer churn prediction (CCP) models aim to detect customers with a high propensity to attrite. This allows a company to improve the efficiency of retention campaigns which aim to prevent customers from churning, by directing personalized retention efforts (Murthi and Sarkar,

2003) to the customers that are effectively about to churn.

Figure 5.1 introduces a general framework for customer churn prediction modeling in the telco industry. The top panel of the figure describes the current state-of-the-art approaches. A telco operator has access to two main data sources[2]: (1) subscription records, i.e. the information a customer provides when subscribing to a service (e.g., postal address, name, etc.); (2) call detail records (CDRs), i.e. communication logs describing the identity (i.e., the phone number) and operator of interacting subscribers, and the exact type, time, and duration of each interaction. Typically, subscription records are only available for the contractual postpaid (as opposed to the non-contractual prepaid) customer segment. These records can be converted into local attributes, which summarize general information related to individual customers, and network attributes, which aggregate information related to the (social) network of a customer. Using historical data, non-relational classifiers are able to learn a classification model to predict future churn based on these attributes. A differentiation between the type of CCP model is made based on the types of attributes that are included in the model. Currently, a great focus exists in the industry on the use of network attributes in order to incorporate social network effects and to improve the predictive power of a CCP model (yielding models of types 2 and 3 in Figure 5.1). However, to our knowledge, the merits of incorporating social network attributes in a CCP model thus far have not been studied thoroughly in the literature. Therefore, in this chapter the impact of network attributes on the predictive power of a non-relational CCP model will be examined in more detail.

As an alternative to the currently applied non-relational approaches discussed above, this chapter develops an alternative approach which is depicted in the bottom panel of Figure 5.1. CDRs can be converted into a graph, with the nodes in the graph representing subscribers and the links between the nodes representing social ties between the subscribers. This graph is called the call graph (Nanavati et al., 2008) and represents the social network of the subscribers of the telco operator. From a theoretical perspective, this chapter contributes by developing relational classifiers that allow to incorporate social network effects within a CCP model and to handle the specific characteristics of a customer churn prediction setting.

[2]A third source concerns external data providers, which have not been included to keep the framework transparent.

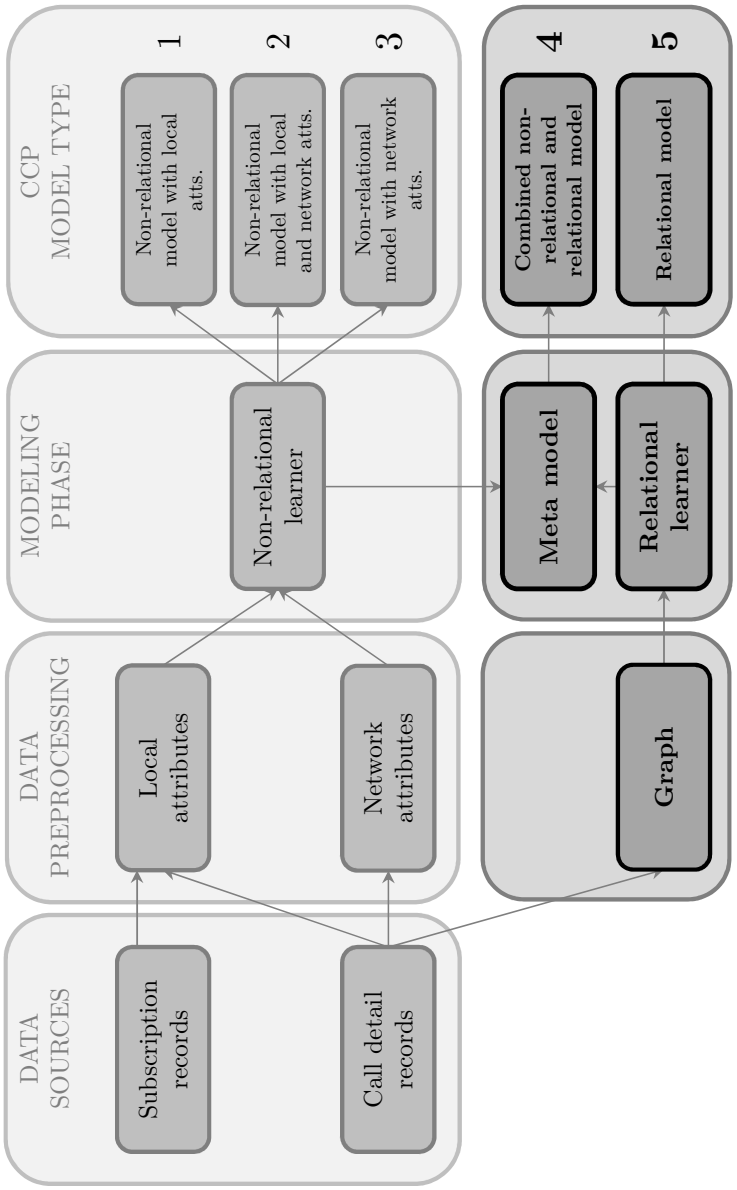


Figure 5.1: General framework for customer churn prediction in the telco industry, with the current modeling approaches depicted in the top panel and the approaches developed in this study in the bottom panel.

For instance, the class distribution of churners and non-churners is typically very skewed, since there are much less churners than non-churners. This may cause existing data mining techniques to experience difficulties in learning powerful models. In a relational learning context, typical methods to handle a skewed class distribution, such as sampling techniques, are not applicable. Therefore, the existence of non-Markovian or higher order social network effects and their use for customer churn prediction is examined. Relational learners typically restrict the impact of the network on a node to the first order neighborhood, i.e., the nodes in the network that are directly connected to a particular node (e.g., Macskassy and Provost (2007), Neville and Jensen (2007)). Existing relational learners are extended in order to incorporate higher order network effects by allowing them to take into account higher order network effects *as if* first order network effects. Furthermore, existing techniques are adjusted to properly handle the time dimension present in customer churn, as well as to deal with the massive size of the graph representing the social network of the customer base of a telco operator, which typically consists of millions of subscribers. Finally, a number of approaches to combine a non-relational and a relational CCP model are presented, aiming to reinforce the predictive power of current CCP modeling approaches by adding a relational model.

From an application perspective, this chapter contributes by evaluating and comparing the CCP modeling approaches summarized in Figure 5.1 in two extensive, real life case studies. This will allow to assess the impact of social network attributes on the performance of a non-relational CCP model, as well as to compare a non-relational, a relational, and a combined CCP model. The two case studies concern the prediction of churn in respectively a prepaid and a postpaid customer segment. To this end, two large-scale, real life data sets have been obtained containing networked (CDRs) and non-networked (usage statistics, sociodemographic, marketing related) information about millions of customers. The results of the experiments will be assessed using lift as well as the novel maximum profit measure, which allows to assess the performance of a CCP model from a profit point of view.

5.2 Social network information for customer churn prediction

5.2.1 Graph theoretical definitions and notations

Boccaletti et al. (2006) defines graph theory as the natural framework for the exact mathematical treatment of complex networks, and formally, a complex network can be represented as a graph. In this chapter the graph and the complex network coincide, since the analyzed networks are approximate mathematical representations of the social ties between the customers of a telco operator. A graph $\mathbf{G}$ consists of a set of vertices (or nodes, or points) $v \in \mathbf{V}$ that are connected by a set of edges (or links, or lines) $e \in \mathbf{E}$, and $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. The number of elements in the sets $\mathbf{V}$ and $\mathbf{E}$ are denoted respectively by n and k. The links in a graph can be *directed* or *undirected*. Directed links point from an origin to a destination node and incorporate a direction property, while undirected links do not. In an undirected graph maximum one link exists between two nodes v_i and v_j, whereas in a directed graph maximum two links exist between two customers v_i and v_j, with one link representing the interactions going from v_i towards v_j and the other representing the interactions from v_j towards v_i.

A graph can be represented by an adjacency matrix $\mathbf{A} = (a_{ij})$ or a weight matrix $\mathbf{W} = (w_{ij})$ of size $n \times n$. An entry a_{ij} or w_{ij} represents the edge between vertex v_i and vertex v_j. The value of a_{ij} in an adjacency matrix is equal to one if an edge exists between vertices v_i and v_j, and equal to zero when no connection exists. In principle, the weights $w_{ij} \in \mathbf{W}$ can take any value. Typically, a weight expresses a characteristic of a link, such as the strength of a social tie between two customers of a telco operator in a call graph. A value of w_{ij} equal to zero means that no relation exists between two vertices. When the weights represent distances, e.g. the length of links in traffic networks, the weight matrix is called the *distance matrix*. The values a_{ii} on the diagonal of an adjacency matrix depend on the convention that is adopted, and either equal once or twice the number of edges from vertex v_i to itself or so-called loops. In case of a weight matrix, the values w_{ii} on the diagonal depend on the property expressed by the weights. For instance, when the weights represent distances, the values on the diagonal will be equal to zero. In this chapter, the values of a_{ii} and w_{ii} will be set equal to zero by definition.

Definition 1. ***The order*** *o* ***network neighborhood*** $\mathcal{N}_i^o$ ***of node*** v_i *is defined as the subset of nodes of the set of all nodes in the network* $\mathbf{V}$ *that are directly or indirectly connected to node* v_i*, with the maximum number of links constituting the shortest path between the nodes in the neighborhood and the node* v_i *equal to the order* o*, and including node* v_i *itself.*

The first order neighborhood $\mathcal{N}_i^1$ or equivalent $\mathcal{N}_i$ of v_i consists of all the nodes in the network that are directly connected to v_i and node v_i itself. The second order neighborhood adds to the first order neighborhood the nodes that are directly connected to the nodes in the first order neighborhood, which are not already in the first order neighborhood, etc. The neighborhood $\mathcal{N}_i^0$ of order zero is a singleton with element node v_i. Section 5.5 will discuss some further properties of graphs. For an extensive overview on graph theory, one may refer to, e.g., Newman (2010).

5.2.2 Related work

A limited number of related prior studies have proposed approaches to use social network information in order to predict customer churn. Nanavati et al. (2008) analyzed the structure and evolution of a massive telecommunication or call graph for a single mobile operator for four different regions in India with different socio-demographic, urbanization, and cultural characteristics, and with the number of nodes for the regions ranging up to 1.25 million. A weight matrix is constructed which only contains information about intra-regional calls, i.e., intra network calls. The period of data gathering was also different for the four regions, ranging from a week to a month.

Dasgupta et al. (2008) presumably analyze the same network as Nanavati et al. (2008), and is to our knowledge the first of its kind in predicting customer churn using social ties between the subscribers of a telco operator. The chapter focuses on the prepaid segment of customers, for which CDR data are indicated to be the only available source of information. A diffusion based modeling technique to predict customer churn is developed, which will be discussed in detail in Section 5.3 and applied in the case studies reported in Section 5.5. As an interesting topic for future research, Dasgupta et al. (2008) indicate the possibility to apply relational classifiers to predict customer churn using CDR data, and specifically NetKit, as developed by Macskassy and Provost (2007) and implemented and applied

in this chapter. The results of the case study reported by Dasgupta et al. (2008) confirm the existence and relevance of social network information for customer churn prediction. However, one of the major differences with this chapter concerns the churn rate, which is much smaller in the data sets that are used in the case studies in this chapter. A skewed class distribution causes relational learners and classification techniques in general to experience difficulties in learning powerful classification models.

Recently, Richter et al. (2010) presented the group-first churn prediction approach to predict customer churn based on the analysis of social groups or communities derived from CDR data. The presented approach assigns a churn score to each subscriber based on the churn score of the social group as well as personal characteristics. The results of the study reconfirm the potential of improving the current generation of CCP models by adding information that captures the social interactions of a subscriber, and indicates that group structure and membership are determinants of churn behavior. This study opens interesting alternative modeling approaches to exploit the information contained within the network structure of the customers of a telco operator. However, given the essential differences with the relational learners that are applied in this chapter, the group-first approach has not been included in the experiments.

Numerous non-relational classification techniques have been adopted for churn prediction and customer targeting in general, including traditional statistical methods such as logistic regression (Lemmens and Croux, 2006; Neslin et al., 2006; Burez and Van den Poel, 2009) and Bayesian techniques, non-parametric statistical models like for instance k-nearest neighbor, decision trees, and neural networks (Baesens et al., 2003a). An extensive overview and discussion of the literature on customer churn prediction can be found in Verbeke et al. (2011e), and a large scale benchmarking study comparing the performance of a range of non-relational classification techniques for customer churn prediction can be found in Verbeke et al. (2011a).

5.2.3 Evaluating customer churn prediction models

CCP models are typically evaluated using lift. The lift curve plots the churn rate among the top fraction of customers with the highest predicted probabilities to churn on the x-axis, divided by the overall base churn rate in the entire customer base. Lift indicates how much better a model identifies

churners compared to randomly targeting a fraction of customers, and as such provides an intuitive measure of model performance. However, the lift curves of different models may intersect, and the highest lift is obtained by a different model depending on the top fraction that is selected. Therefore, using lift curves to compare the performance of CCP models may not provide a conclusive answer as to which model performs best. Moreover, the commonly used top-decile lift may lead to suboptimal model selection as well, since setting the value of the top fraction to ten percent is arbitrary.

Therefore, a performance measure to evaluate CCP models from a profit centric point of view will be applied that was recently introduced by Verbeke et al. (2011a). This performance measure builds on an expression introduced by Neslin et al. (2006) to calculate the profit associated with a customer retention campaign that targets subscribers based on the outcomes of a CCP model.

Definition 2. ***The total profit*** P_t *generated by a retention campaign equals:*

$$P_t = n\eta \left[(\gamma CLV + \delta(1-\gamma))\, \pi_0 \lambda - \delta - \phi \right] - A, \tag{5.1}$$

with η *the fraction of the customer base that is targeted,* CLV *the average customer lifetime value,* δ *the cost of the incentive,* ϕ *the cost of contacting the customer, and* A *the fixed administrative costs. The lift coefficient,* λ*, is the percentage of churners within the targeted fraction* η *of customers, divided by the base churn rate,* π_0*. Finally,* γ *is the probability of a targeted churner to be retained by the offered incentive. It is assumed that all parameters are positive, and that* $CLV > \delta$*.*

Equation 5.1 states that the profit resulting from a retention campaign to prevent customers from churning by offering an incentive to remain loyal, equals the saved value (CLV) associated with retained churners (fraction γ of the churners included in the campaign) minus the costs of the campaign. The costs of the campaign equal (1) the cost of the accepted incentives (δ) by the non-churners and the fraction of churners that are retained, (2) the cost of contacting the fraction of the customers included in the retention campaign (ϕ), and (3) a fixed administrative cost (A).

The total profit generated by a retention campaign hence depends on the fraction of included customers η, and the lift associated with this fraction, which is directly associated with the CCP model. Hence, the included fraction of customers needs to be optimized in order to maximize the generated profits by a retention campaign (Padmanabhan and Tuzhilin, 2003).

Definition 3. ***The maximum profit measure for customer churn*** *(MPC) is defined as the maximum profit that can be obtained by using the outcomes of a customer churn prediction model, and which can be used to evaluate the predictive power of a CCP model:*

$$MPC = \max_{\forall \eta} P_t(\eta; \gamma, CLV, \delta, \phi). \tag{5.2}$$

Customer retention campaigns aim to minimize the costs associated with customer churn, and therefore it is straightforward to evaluate and select a CCP model by using the maximum profit that can be generated as a performance measure. In this chapter, both the lift and the MPC will be applied to assess the performance of the generated models, providing complementary insights.

5.3 Classification in networked data

The basic premise for customer churn prediction using social network information is that customers interlinked with customers that have churned are more likely to churn themselves. Possible explanations for the existence of such *network effects* are the word-of-mouth effect, social leader influence, promotional offers from operators to acquire groups of friends, and reduced tariffs for intra-operator traffic. The principle that a contact between similar people occurs at a higher rate than among dissimilar people has been observed in many kinds of social networks and is called homophily (Blau, 1977; McPherson et al., 2001; Macskassy and Provost, 2007), assortativity (Newman, 2010), or relational autocorrelation (Jensen and Neville, 2002).

An indication of the existence of network effects on the churn behavior of telco subscribers is provided by Figure 5.2, which plots the network neighborhood of a particular customer in the CDR data set presented in Section 5.5 up to degree eight. A viral like spreading of churn can be observed in this part of the call graph, and a churn rate that is substantially higher than the base churn rate in the entire call graph. Many of such network effects can be observed, hinting towards a great potential of exploiting social network information for churn prediction.

The following subsections discuss the presented approaches in the framework of Figure 5.1 to incorporate social network information within a customer churn prediction model. The first two subsections examine the use of

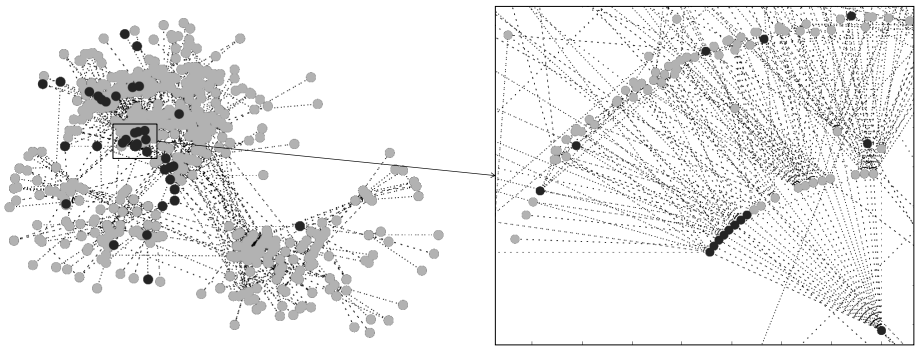


Figure 5.2: The neighborhood of order eight of a particular customer in the call graph, with churners represented by black dots, and non-churners represented by gray dots. A sequence of twelve subsequent churners can be found in this network neighborhood, indicating a viral-like propagation or spreading of churn throughout the call graph.

relational learning techniques, while the third subsection explores the use of network variables in a non-relational model. The fourth subsection finally presents a range of meta modeling approaches to combine a relational and a non-relational model.

5.3.1 Relational learning for customer churn prediction

Macskassy and Provost (2007) introduced a framework for learning in networked data. In this node-centric framework, a relational learner comprises a relational classifier and optionally a collective inference (CI) procedure. The original formulation of relational learners as described in Macskassy and Provost (2007) assumes that part of the class labels of the nodes in the network are known, and can be used to estimate the unknown class labels. In other words, the set of vertices $\mathbf{V}$ consists of a subset of vertices with a known label, $\mathbf{V}^K$, and a subset of vertices with an unknown label, $\mathbf{V}^U$, i.e. $\mathbf{V} = \{\mathbf{V}^K, \mathbf{V}^U\}$. The inference of the unknown labels based on the known labels is determined by the relational classifier. The order in which unknown class labels are inferred, as well as how the inferred labels influence each other, is coordinated by the CI procedure.

Relational classifiers

Table 5.1 provides a dense summary of the relational classifiers that are applied in the case study in Section 5.5. Throughout this chapter, c will be used to refer to a non-specified class value, and k is an index running over all possible class labels. All relational classifiers included in Table 5.1 have been implemented using sparse and parallel computation techniques in order to be applicable on massive networks consisting of millions of nodes[3].

In order to deal with the time dimension that is explicitly present in customer churn, both the CDRN and the NLB relational classifier have been reformulated. When predicting future customer churn in real life, none of the future labels, i.e. at time $t+1$, are known when training a classification model at time t. In other words, the set of vertices with an unknown label equals the set of all vertices, $\mathbf{V} = \mathbf{V}^U$, and the set of vertices with known labels is empty, $\mathbf{V}^K = \emptyset$. Therefore, the unknown labels at time $t+1$ need to be estimated based on the labels at time t, requiring an adjustment to the formulation of the CDRN and the NLB relational classifiers.

The class vector $CV(v_i)$ of a node v_i in the formulation of the CDRN classifier can be defined as the vector of summed linkage weights to the various classes at time t. The reference vector $RV(c)$ of a class c would then be the average of these class vectors for nodes known to be of class c at time $t+1$. However, none of the labels at time $t+1$ are known when training a classifier at time t. Therefore, the reference vectors have to be calculated using data from a previous time frame. For instance, by calculating the average of the class vectors at time $t-1$ for nodes known to be of class c at time t. The class vectors at time t can then be compared to these reference vectors in order to make predictions for time $t+1$.

Similar to the CDRN classifier, the NLB classifier uses the class vectors at time t as independent variables in order to predict the class labels at time $t+1$ by fitting a logistic regression model to the class vectors of nodes known to be of class c at time $t+1$. Since no class labels at time $t+1$ are known on beforehand, the model has to be trained on data from a previous time frame, for instance by using the class vectors at time $t-1$ to predict the class labels at time t.

[3]Nonetheless, the computational complexity of the Network-only Bayes Classifier, as included in NetKit (Macskassy and Provost, 2007) and based on a relational classifier introduced by Chakrabarti et al. (1998), appeared to be prohibitive for application on a very large network, specifically in combination with a CI procedure.

RC	Summary
CDRN	The *Class-Distribution Relational Neighbor classifier* (Rocchio, 1971; Perlich and Provost, 2003, 2006) learns a model based on the distribution of neighbor class labels. The class vector $CV(v_i)$ of a node v_i is defined as the vector of summed linkage weights to the various classes: $$CV(v_i)_k = \sum_{v_j \in \mathcal{N}_i} w_{ij} \cdot P(l_j = c_k | \mathcal{N}_j) \quad (5.3)$$ with l_j the non-specified label of node v_j. The reference vector $RV(c)$ of a class c is defined as the average of the class vectors for nodes known to be of class c: $$RV(c) = \frac{1}{|\mathbf{V}_c^K|} \sum_{v_i \in \mathbf{V}_c^K} CV(v_i) \quad (5.4)$$ with $\mathbf{V}_c^K = \{v_i | v_i \in \mathbf{V}^K, l_i = c\}$, and $\mathbf{V}^K \subset V$ the subset of vertices with a known class label. Then the probability for a customer to be a churner can be calculated as the normalized vector similarity between v_i's class vector and the churners' class reference vector: $$P(l_i = c | \mathcal{N}_i) = sim(CV(v_i), RV(c)) \quad (5.5)$$ where $sim(a, b)$ can be any vector similarity measure normalized to lie in the range $[0, 1]$. In the experimental section of this study cosine similarity will be applied.
NLB	The *Network-only Link Based classifier* (Lu and Getoor, 2003) applies logistic regression on feature vectors which are constructed for each node by aggregating the labels of neighboring nodes, e.g. existence (binary), the mode, and value counts. The count model is equivalent to the class vector $CV(v_i)$ defined in Equation 5.3, and has been shown to perform best by Lu and Getoor (2003). $$CV(v_i)_k = \frac{\sum_{v_j \in \mathcal{N}_i} w_{ij} \cdot P(l_j = c_k | \mathcal{N}_j)}{\sum_{v_j \in \mathcal{N}_i} w_{ij}} \quad (5.6)$$ $$P(l_i = c | \mathcal{N}_i) = \frac{1}{1 + e^{-\beta_0 - \beta \cdot CV(v_i)}} \quad (5.7)$$ with β_0 and β representing the parameters of the logistic regression model.

RC	Summary
SPA RC	The *Spreading Activation Relational Classifier* is based on the Spreading Activation technique (SPA) proposed by Dasgupta et al. (2008). The original SPA technique models the propagation of churn through a network as a diffusion process of *churn energy*. Incorporating SPA within the modular framework of Macskassy and Provost (2007) results in the SPA RC relational classifier, defined as follows: $$P(l_i = c\|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} d \cdot \frac{w_{ji}}{\sum_{s \in \mathcal{N}_j} w_{js}} \cdot P(l_j = c\|\mathcal{N}_j) \quad (5.8)$$ where Z is a normalizer to convert the energy levels in probability scores, and d a parameter that controls the diffusion process. This expression is similar to the WVRN classifier, but now the impact of a neighboring node v_j on node v_i does not depend on the relative weight w_{ij} within the neighborhood of node v_i, i.e. $\mathcal{N}_i$, but instead on the relative weight within the neighborhood of node v_j, i.e. $\mathcal{N}_j$.
WVRN	The *Weighted-Vote Relational Neighbor classifier* estimates the probability of a customer to churn as a function of the probabilities of its neighbors to churn: $$P(l_i = c\|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{ij} \cdot P(l_j = c\|\mathcal{N}_j) \quad (5.9)$$ with Z a normalizer for the probabilities to sum up to one.

Table 5.1: Summary of relational classifiers.

Finally, the original *SPreading Activation based approach* (SPA) as proposed by Dasgupta et al. (2008) and discussed in Section 5.2.2, is reformulated and split into a relational classifier (SPA RC, see Table 5.1) and a collective inference procedure (SPA CI, cfr. infra), in order to fit within the framework proposed by Macskassy and Provost (2007).

Collective inference procedures

Table 5.2 summarizes a range of existing and adjusted collective inference procedures that are applied in the experiments in Section 5.5. The computational complexity of both the original Gibbs Sampling and Iterative

Classification formulations appeared to be prohibitive for application on massive social networks consisting of more than a million nodes and links. More specifically, the iterative application of a relational classifier in step 2.(a) of both procedures, running over all the nodes in the network, increases their complexity dramatically. Therefore, an adjusted version of both Gibbs Sampling and Iterative Classification is proposed, which allows them to be applied on very large networks, by making inferences concurrently instead of iteratively. The Gibbs Sampling with Simultaneous Labeling (GSSL) and Iterative Classification with Simultaneous Labeling (ICSL) collective inference procedures schematically work as indicated in Table 5.2.

Similar to the above described collective inference procedures, the Spreading Activation based Collective Inference procedure (SPA CI) applies a relational learner in each iteration, using the result of iteration i as the input to the next iteration $i+1$. However, whereas Gibbs sampling, relaxation labeling, and iterative classification consist of a specified number of iterations, the SPA CI procedure ends when a stopping criterion is met, consisting of two conditions. The procedure ends when (1) the set of active nodes is not extended, and (2) the amount of energy that is spread, i.e., the overall change in the assigned labels, is smaller than a predefined amount E_t. When combining SPA CI with SPA RC, the stopping criterion will be met since the amount of energy that is passed is reduced by the SPA RC classifier in each iteration, by application of the diffusion coefficient $d \in (0, 1)$. However, convergence is not guaranteed when combining SPA CI with any of the other relational classifiers defined in Section 5.3. Therefore, similar to the RL procedure, a simulated annealing approach with a predetermined maximum number of iterations is applied when combining SPA CI with any of the other relational classifiers. Equation 5.12 is then replaced by Equation 5.11.

5.3.2 Non-relational learning with network variables

An alternative approach to the relational classifiers and collective inference procedures discussed in the previous sections, exists in transforming the information that is contained within the social network structure into a set of *network variables* or attributes. These network variables can then be used by traditional, non-relational data mining techniques, yielding a CCP model of type two or three according to the framework of Figure 5.1. Methods that transform a relational representation of a learning problem into

CI	Summary
GS	*Gibbs Sampling* (Geman and Geman, 1984) schematically works as follows (Macskassy and Provost, 2007): 1. Generate a random ordering, O, of vertices in $\mathbf{V}^U$. 2. For elements $v_i \in O$ in order: (a) Apply the relational classifier model: $\hat{\mathbf{c}}_i \leftarrow \mathcal{M}_R(v_i)$. (b) Sample a value c_s from $\hat{\mathbf{c}}_i$, such that $P(c_s = c_k \vert \hat{\mathbf{c}}_i) = \hat{\mathbf{c}}_i(k)$. (c) Set $l_i \leftarrow c_s$. 3. Repeat prior step 200 times without keeping any statistics (burnin period). 4. Repeat again for 2000 iterations, counting the number of times each l_i is assigned a particular value $c \in \mathcal{L}$. Normalizing these counts forms the final class probability estimates.
IC	*Iterative Classification* (Lu and Getoor, 2003) is formulated by Macskassy and Provost (2007) as follows: 1. Generate a random ordering, O, of vertices in $\mathbf{V}^U$. 2. For elements $v_i \in O$ in order: (a) Apply the relational classifier model, $\hat{\mathbf{c}}_i{}^{(0)} \leftarrow \mathcal{M}_R$, using all non-null labels (entities which have not yet been classified are ignored). If all neighbor entities are null, then return null. (b) Classify v_i: $l_i = c_k$ and $k = \text{argmax}_j\,(\hat{\mathbf{c}}_i(j))$, where $\hat{\mathbf{c}}_i(j)$ is the j^{th} value in vector $\hat{\mathbf{c}}_i$. 3. Repeat for $T = 1000$ iterations, or until no entities receive a new class label. The estimates from the final iteration will be used as the final class probability estimates.

CI	Summary
RL	*Relaxation Labeling* (Chakrabarti et al., 1998) is defined by Macskassy and Provost (2007) as follows: 1. For elements $v_i \in \mathbf{V}^U$: Estimate l_i by applying the relational model: $$\hat{\mathbf{c}_i}^{(t+1)} \leftarrow \mathcal{M}_R(v_i^{(t)}) \quad (5.10)$$ where $\mathcal{M}_R(v_i^{(t)})$ denotes using the estimates $\hat{\mathbf{c}_j}^{(t)}$ for $v_j \in \mathbf{V}^U$, and t is the iteration count. This has the effect that all the iterations are done pseudo-simultaneously based on the state of the graph after iteration t. 2. Repeat for T iterations. $\hat{c}^T$ will comprise the final class probability estimates.
RL SA	*Relaxation Labeling with Simulated Annealing* applies simulated annealing for the resulting labels to converge, and substitutes Equation 5.10 by the next expression: $$\hat{\mathbf{c}_i}^{(t+1)} = \beta^{(t+1)} \cdot \mathcal{M}_R(v_i^{(t)}) + (1 - \beta^{(t+1)}) \cdot \hat{\mathbf{c}_i}^{(t)} \quad (5.11)$$ with $\beta^0 = k$, and $\beta^{(t+1)} = \beta^{(t)} \cdot \alpha$, k a constant between zero and one, and α a decay constant.
GS SL	*Gibbs Sampling with Simultaneous Labeling* replaces steps 1 and 2 in the above inference scheme by a single step: 2. (a) Apply the relational classifier model: $\hat{\mathbf{c}_i^{t+1}} \leftarrow \mathcal{M}_R(v_i^t)$, where $\mathcal{M}_R(v_i^{(t)})$ denotes using the estimates $\hat{\mathbf{c}_j}^{(t)}$ for $v_j \in \mathbf{V}^U$, and t is the iteration count. This has the effect that all the iterations are done pseudo-simultaneously based on the state of the graph after iteration t. (b) Sample a value c_s from $\hat{\mathbf{c}_i}$, such that $P(c_s = c_k \vert \hat{\mathbf{c}_i}) = \hat{\mathbf{c}_i}(k)$. (c) Set $l_i \leftarrow c_s$.

CI	Summary
IC SL	*Iterative classification with Simultaneous Labeling* replaces steps 1 and 2 in the original IC scheme by a single step: 2. (a) Apply the relational classifier model, $\hat{\mathbf{c}}_i^{(t+1)} \leftarrow \mathcal{M}_R(v_i^{(t)})$, using all non-null labels (entities which have not yet been classified are ignored), and where $\mathcal{M}_R(v_i^{(t)})$ denotes using the estimates $\hat{\mathbf{c}}_j^{(t)}$ for $v_j \in \mathbf{V}^U$, and t is the iteration count. This has the effect that all the iterations are done pseudo-simultaneously based on the state of the graph after iteration t. If all neighbor entities are null, then return null. (b) Classify v_i: $l_i = c_k$ and $k = \text{argmax}_j\,(\hat{\mathbf{c}}_i(j))$, where $\hat{\mathbf{c}}_i(j)$ is the j^{th} value in vector $\hat{\mathbf{c}}_i$.
SPA CI	The *Spreading Activation Collective Inference* procedure based on the SPA approach described by Dasgupta et al. (2008) can be defined as follows: 1. For $v_i \in \mathbf{V}^U$, initialize the prior: $\hat{\mathbf{c}}_i^{(0)} \leftarrow \mathcal{M}_L(v_i)$, where $\hat{\mathbf{c}}_i$ is defined as above in the Gibbs sampling algorithm. 2. For elements $v_i \in \mathbf{V}^U$: Estimate l_i by applying the relational model: $$\hat{\mathbf{c}}_i^{(t+1)} \leftarrow \mathcal{M}_R(v_i^{(t)}) \quad (5.12)$$ where $\mathcal{M}_R(v_i^{(t)})$ denotes using the estimates $\hat{\mathbf{c}}_j^{(t)}$ for $v_j \in \mathbf{V}^U$, and t is the iteration count. This has the effect that all the iterations are done pseudo-simultaneously based on the state of the graph after iteration t. 3. Repeat while (a) $\sum(\hat{\mathbf{c}}_i^{(t+1)} - \hat{\mathbf{c}}_i^{(t)}) > \Delta c_{min}$, with Δc_{min} the minimum overall difference in predicted class labels, (b) OR $\#(\hat{\mathbf{c}}_i^{(t+1)} > 0) > \#(\hat{\mathbf{c}}_i^{(t)} > 0)$, (c) AND $t < T_{max}$, with T_{max} the maximum number of iterations. The resulting $\hat{c}^{t_{end}}$ will comprise the final class probability estimates.

Table 5.2: Summary of original and adjusted collective inference procedures.

a propositional, feature-based or attribute-value representation are known as propositionalization or featurization approaches (Kramer et al., 2001). In a customer churn prediction setting a range of network variables can be defined, such as the aggregate number of connections between a subscriber and previously churned subscribers, the total time called to churners, etc. These network variables can be derived from the call graph, or directly from the call detail records as shown in Figure 5.1.

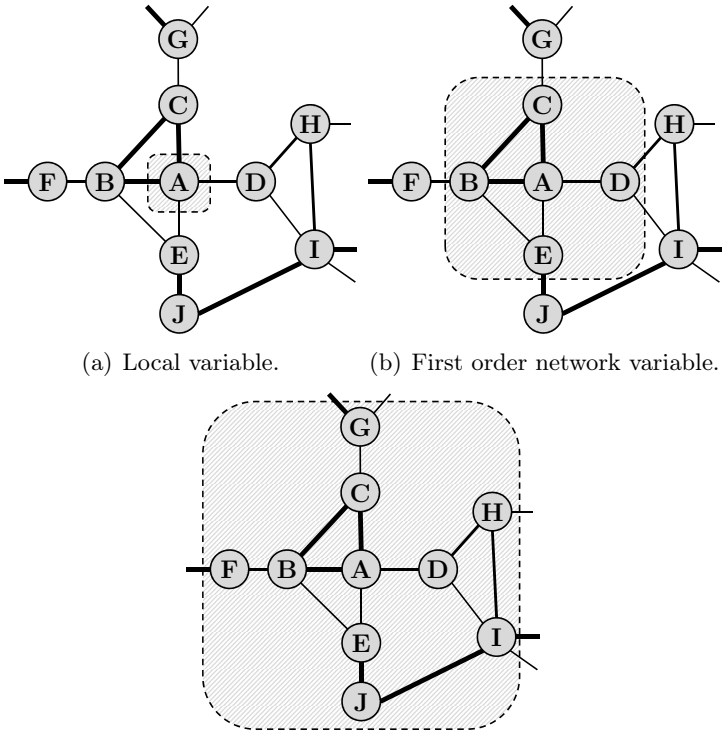
Typically, an important fraction of the explanatory variables in a CCP model are usage statistics (Verbeke et al., 2011e), which aggregate information contained within call detail records, such as the number of contacts (i.e., neighbors in the call graph) and the number of contacts that are subscribers of a competing operator. In order to make a clear distinction between a network variable and a non-network or *local* variable, a formal definition is introduced:

Definition 4. ***A network variable*** *related to instances or objects aggregates information that is contained within a graph or network structure and makes a differentiation in the destination of outgoing links or the origin of incoming links. A network variable of order i aggregates information related to an instance or object contained within its order i neighborhood.*

Definition 5. ***A local variable*** *represents information related to instances or objects that are treated as isolated entities with unspecified connections to the outside world. A local variable is a network variable of order* 0*.*

Figure 5.3 schematically represents the difference between a local variable and a network variable of order 1 and 2, which is in fact the part or range of the network that is *visible* and summarized by the variable. According to Definitions 4 and 5, a variable such as the number of contacts of a customer is a local variable, since it aggregates information from the call graph but does not differentiate between types of contacts. On the other hand, the number of contacts of a customer that are subscribers of a competing operator is a network variable, since a differentiation is made between the origin of incoming and the destination of outgoing links.

The advantage of a propositionalization approach is that it allows to use powerful, accustomed, non-relational modeling techniques. The disadvantage of this approach is that it does not fully take advantage of the possibilities offered by networked data, and that valuable information may be lost in the conversion of the network into attributes.



(a) Local variable.

(b) First order network variable.

(c) Second order network variable.

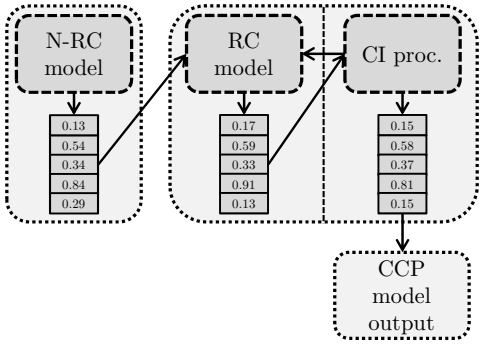
Figure 5.3: Schematic representation of the scope of a local (left panel) and a first (middle panel) and second (right panel) order network variable related to instance A.

5.3.3 Combining relational and non-relational classifiers

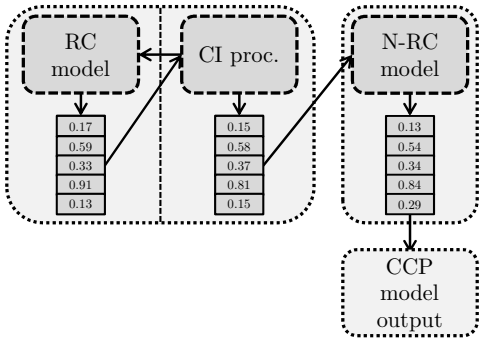
In this section four approaches are presented to combine a relational and non-relational classifier, constituting the base classifiers of an *ensemble* or combined CCP model of type 4 according to the framework of Figure 5.1. For an extensive overview on ensemble learning for classification, one may refer to, e.g., Hastie et al. (2001).

A first approach to combine a non-relational classifier with a relational classifier and optionally a collective inference procedure in a unified setup follows from the definitions of the RC and CI algorithms; the probabilities or scores resulting from a non-relational classification model can be used to

initialize the labels of the nodes in the network. The non-relational model is used in this setup to provide a first estimate of the class labels of the nodes in the network. Subsequently, the relational classifier and the collective inference procedure are added as a second model layer on top of the non-relational classifier, with the intention to refine and improve the results of the non-relational model by using the information that is incorporated within the networked data.



(a) Non-relational model as input to relational model.



(b) Relational model as input to non-relational model.

Figure 5.4: Schematic representation of two approaches to combine a collective inference procedure and a relational and non-relational classification model.

Conversely, the predicted probabilities by the relational model can be included in the non-relational model as an explanatory variable. In this approach, the non-relational model constitutes a second model layer on top

of the relational model. In fact, this approach can be regarded as an *automated* propositionalization of the network, leading to a single network attribute that is not explicitly defined but nonetheless aggregates information that is contained within the network. Both *cascading* approaches or *sequential* model setups are schematically represented in Figure 5.4.

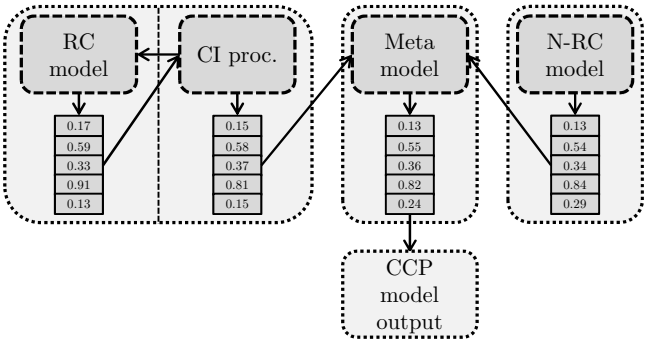
An alternative approach called *stacking* exists in combining the output scores of the relational and non-relational classifier by learning a model on top of these two models. The stacking approach uses the probabilities resulting from the non-relational classification model and the relational classification model, whether or not in combination with a CI procedure, as input variables. This approach is schematically represented in Figure 5.5(a). In principle, any non-relational classification technique could be applied to build a second model layer on top of the relational and non-relational classification models.

Finally, the relational and the non-relational model can also be applied in a parallel, non-integrated setup, i.e., by selecting customers indicated to have a high probability to churn either by the non-relational model or by the relational model (or by both models), as shown by Figure 5.5(b). This approach is called *voting*, and a customer is classified to be a churner when either the relational or the non-relational CCP model classifies a customer as a churner. Remark that when the base classifiers result in a probability estimate or a continuous output score, with a higher score incorporating a higher probability to churn, the cutoff value for a customer to be classified as a churner can be set independently for the base classifiers. In fact, as indicated in Section 5.2.3, these cutoff values need to be optimized in order to maximize the resulting lift or profit, involving a combinatorial optimization problem.

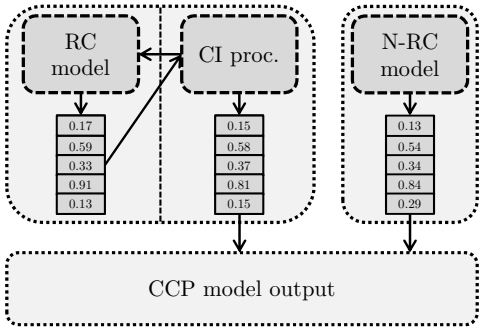
5.4 Modeling non-Markovian network effects

5.4.1 Non-Markovian network effects

The order n network neighborhood of a particular node was defined in Section 5.2. The concept of network neighborhood can be applied straightforward to a set of nodes as well, as the union of all the neighborhoods of the nodes in the set. In this section the *exclusive* order n network neighborhood of a particular node, which is applicable to a set of nodes as well, is defined as follows:



(a) Meta-model approach.



(b) Parallel model setup.

Figure 5.5: A meta-model approach (left panel) and a parallel, non integrated setup (right panel) to combine a non-relational model, a relational model, and a collective inference procedure.

Definition 6. ***The exclusive order*** *o* ***network neighborhood*** $N_i^{o,e}$ ***of node*** v_i *is defined as the subset of nodes of the set of all nodes in the network* $\mathbf{V}$ *that are connected to node* v_i *with the number of links of the shortest path connecting the nodes in the neighborhood and the node* v_i *exactly equal to the order* o*.*

Figure 5.6 provides an indication of the existence of higher order network effects in a customer churn setting, resulting from the analysis of a social network derived from call detail record data, which will be described into detail in Section 5.5. The figure plots the effect on the prior probability to churn in the exclusive order x neighborhood of the churners in the net-

work, as a function of the order x. The grey bars indicate the fraction of the customer base $\mathbf{V}$ that is included in the exclusive order x network neighborhood $N^x_{\mathbf{V}_c^{t-1}}$ of the set of customers $\mathbf{V}_c^{t-1}$ that churn in time frame $t-1$. The black line indicates the churn rate in time frame t in this neighborhood, and the dashed black line represents the base churn fraction in the entire customer base in time frame t. As can be observed from this plot, customers that are direct, first order neighbors of churners clearly have a much larger probability to churn than random customers. However, also customers that are second and even higher order neighbors of churners display an increased prior probability to churn. As the order of the neighborhood increases, the network effect on the probability to churn decreases, and as of order five the effect has entirely disappeared. Figure 5.6 provides an important indication of the existence of higher order effects on the churn behavior of customers. The order 1 effect in Figure 5.6 is exactly what has been called in Section 5.3 homophily, assortativity, or relational autocorrelation.

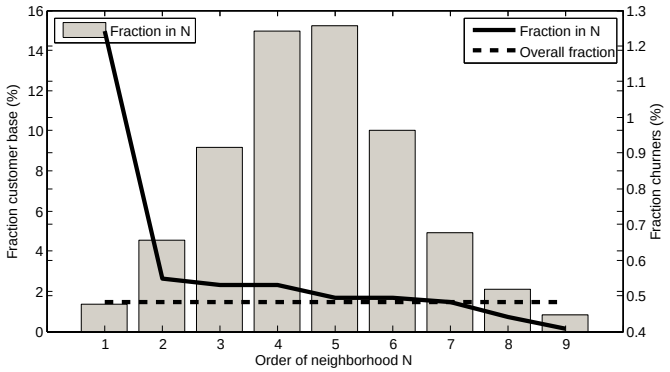


Figure 5.6: Effect on the prior probability to churn of the order x. The grey bars indicate the fraction of the customer base $\mathbf{V}$ that is included in the exclusive order x network neighborhood $N^x_{\mathbf{V}_c^{t-1}}$ of the set of customers that churn in time frame $t-1$, $\mathbf{V}_c^{t-1}$. The black line indicates the churn rate in time frame t in this neighborhood, and the dashed black line represents the base churn fraction in the entire customer base.

Notice that the set of customers in the order x neighborhood does not contain the customers in the order $x-1$ neighborhood, since the concept of the exclusive order x neighborhood is applied. Furthermore, in order to can-

cel out sequences of first order effects instead of *pure* higher order effects[4], the higher order neighbors of churners in lower order neighborhoods are not taken into account. This is important in order to isolate and clearly distinguish between the effects of the neighborhood order. However, whether higher order effects are in fact sequences of first order effects or not, it is relevant to take these effects into account. Figure 5.6 shows that for an increasing order of the exclusive neighborhood the amount of subscribers in the neighborhood increases as well, until order five. Although a smaller fraction of the subscribers in the exclusive neighborhood of order two churn, the absolute number of churners in this neighborhood is approximately equal to the absolute number of churners in the order one neighborhood, which reconfirms the importance of taking into account higher order network effects.

5.4.2 The weight product

When applying relational classifiers that restrict the impact of the network to the first order neighborhood in combination with collective inference procedures, the impact of a particular node partially propagates or spreads throughout the network and reaches nodes in higher order neighborhoods. This is the result of the iterative application of first order neighborhood relational classifiers, since subsequent first order effects constitute a higher order effect. However, as will be shown in the experimental results section, the application of collective inference procedures may deteriorate the performance of the resulting classification model. Nonetheless, as indicated in the above section, the impact of higher order neighborhood nodes appears relevant to be incorporated within a CCP model. Therefore, this section introduces a novel approach to *upgrade* the order of the weight matrix in order to include higher order nodes within the local neighborhood that is taken into account by the relational learners. Higher order nodes are transformed into first order neighborhood nodes, with appropriate values assigned to the respective weights. This allows relational classifiers to model non-Markovian

[4]For instance, assume that a churning subscriber in time frame $t-1$ causes a connected subscriber to churn in time frame t. Within time frame t, this churned neighbor causes one of his connections to churn, resulting in a *second* order effect, which is in fact a sequence of *two first order effects*. This also illustrates the impact of the time period frame. Higher order effects resulting from sequential first order effects can be expected to have a larger impact for longer time frames t.

network effects without the need to apply a collective inference procedure and without the need to adjust the inner workings of the relational learning techniques.

Definition 7. *The* ***standard matrix product*** *or the* plus-times *matrix product $C = A \cdot B = (c_{ij})$ for $n \times n$ matrices $A = (a_{ij})$ and $B = (b_{ij})$ is defined by:*

$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}. \tag{5.13}$$

Definition 8. *The* ***distance matrix product*** *or the* minimum-plus *matrix product $C = A \star B$ is defined similar to the plus-times product $C = A \cdot B$, but with the summation operator replaced by the minimum operator and the product operator replaced by the summation operator:*

$$\begin{aligned} c_{ij} &= a_{ij} \star b_{ij} \\ &= \min_{k}(a_{ik} + b_{kj}). \end{aligned} \tag{5.14}$$

The distance product $D^{\star 2} = D \star D^T$, with D a distance matrix and elements d_{ij} representing the first order distance between nodes v_i and v_j, results in a matrix $D^{\star 2}$ with elements $d_{ij}^{\star 2}$ equal to the shortest distance between nodes spanning *exactly* two edges. The matrix $min(D, D^{\star 2})$ yields the shortest distance between nodes spanning *maximum* two edges.

In order to upgrade the order of the weights matrix, we define a matrix operation that is equivalent to the distance product but applies to weight matrices, by assuming that weights are equivalent to the inverse of distance, i.e., the higher the value of a weight w_{ij}, the closer two nodes are related:

Definition 9. *The* ***weight matrix product*** *or the* maximum-of-times-divided-by-plus *matrix product $C = A \circledast B$, is defined as follows:*

$$\begin{aligned} c_{ij} &= a_{ij} \circledast b_{ij} \\ &= \max_{k}\left(\frac{a_{ik} \cdot b_{kj}}{a_{ik} + b_{kj}}\right). \end{aligned} \tag{5.15}$$

Theorem 1. *The weight matrix product $W^{\circledast 2} = W \circledast W^T$ defined by Definition 9 is the equivalent matrix operation for weight matrices of the distance matrix product for distance matrices $D^{\star 2} = D \star D^T$ defined by Definition 8,*

assuming weights representing a link in a graph are equivalent to the inverse of distances associated with links in a graph:

$$(w_{ij}) \circledast (w_{ij})^T = (w_{ij})^{-1} \star (w_{ij})^{-1,T}$$

or,

$$(d_{ij}) \star (d_{ij})^T = (d_{ij})^{-1} \circledast (d_{ij})^{-1,T} \tag{5.16}$$

Proof. Proof of Theorem 1

Assuming a weight is the equivalent of an inverted distance, a weights matrix $W = (w_{ij})$ can be transformed into an equivalent distance matrix $D = (d_{ij})$ by taking the inverse of each weight in the matrix, i.e., $(w_{ij}) = 1/(d_{ij})$. Subsequently, the distance matrix (d_{ij}) can be upgraded to the second order distance matrix D^2 by applying the distance product, $d_{ij}^{\star 2} = d_{ik} \star d_{kj}^T = \min_k(d_{ik} + d_{kj}^T)$. Finally, the resulting distance product matrix can be converted in a weights matrix again, $(w_{ij}^{\circledast 2}) = 1/(d_{ij}^{\star 2})$. This sequence of operations can be expressed as a single operation, i.e., the weight product as defined by Definition 9.

Let us first express $d_{ij}^{\star 2}$ in terms of the weight matrix as follows:

$$\begin{aligned} d_{ij}^{\star 2} &\equiv \min_k \left(d_{ik} + d_{kj}^T\right) \\ &= \min_k \left(\frac{1}{w_{ik}} + \frac{1}{w_{kj}^T}\right) \\ &= \min_k \left(\frac{w_{ik} + w_{kj}^T}{w_{ik} \cdot w_{kj}^T}\right). \end{aligned} \tag{5.17}$$

Next, the distance product matrix (d_{ij}^2) within the equation $(w_{ij}^2) = 1/(d_{ij}^2)$ can be substituted by Equation 5.17:

$$\begin{aligned} w_{ij}^{\circledast 2} &= \frac{1}{d_{ij}^{\star 2}} \\ &= \frac{1}{\min_k \left(\frac{w_{ik}+w_{kj}^T}{w_{ik} \cdot w_{kj}^T}\right)} \\ &= \max_k \left(\frac{w_{ik} \cdot w_{kj}^T}{w_{ik} + w_{kj}^T}\right) \\ &= w_{ij} \circledast w_{ij}^T, \end{aligned} \tag{5.18}$$

which formally proofs Theorem 1. □

The maximum of the weight matrix W and the weight product of the weight matrix $W^{\circledast 2}$, i.e., $max(W, W^{\circledast 2})$, yields an upgraded weight matrix that incorporates second order nodes *as if* first order nodes with appropriate weights assigned to the links representing second order connections. p subsequent applications of the weight product and selection of the maximum weight for each relation between two nodes in each step, yields the weight matrix of order p.

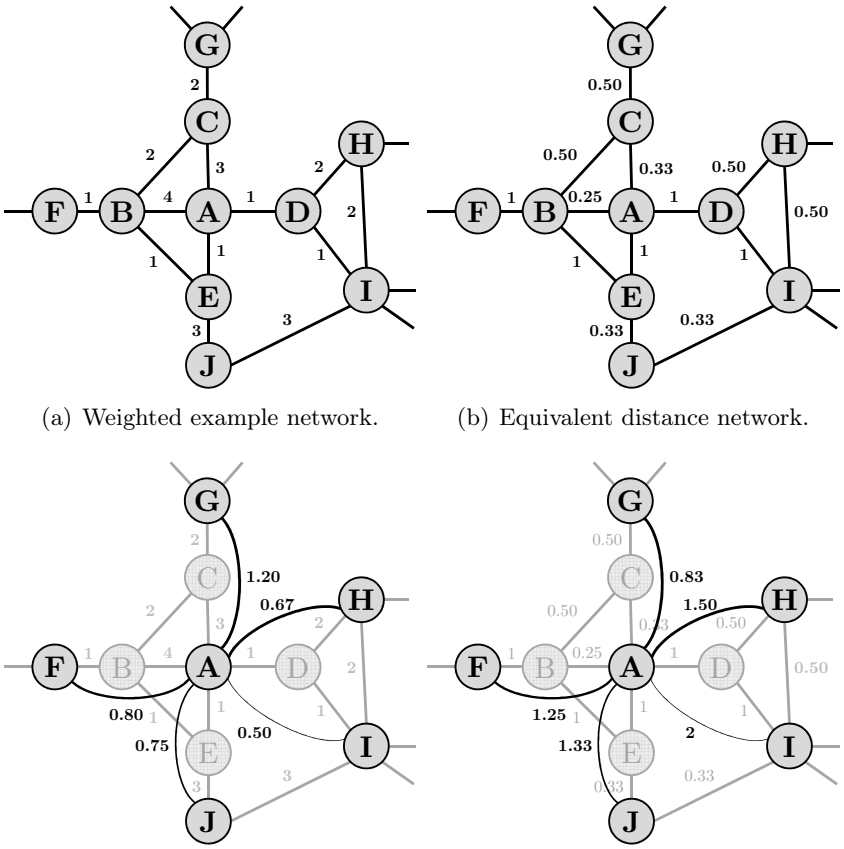
The equivalence between the weight and distance product is illustrated by Figure 5.7, which plots a weighted version of the simple example network used in the previous section with first and second order links to node A, and the equivalent distance networks.

5.5 Case studies

This section presents the application of the presented techniques in two real life case studies, concerning a prepaid and postpaid customer segment.

5.5.1 Data set and experimental setup

CDRs of voice to voice calls for both a prepaid and a postpaid customer segment were provided by an anonymous European telco operator. The time range of the CDR data covers a period of five months, which will be denoted M1 to M5. Churn labels, indicating the exact date when customers churned, were available for this period and one month prior and after. Based on previous studies and extensive discussions with telco experts, CDRs are converted into a network by defining nodes as subscribers of the operator, and edges as the total number of seconds of voice to voice calls between these subscribers. The resulting network is represented by an undirected weight matrix, which was preferred over a directed matrix to reduce computational complexity of the relational learning process. The number of subscribers in the customer base is $1,673,724$ for the prepaid segment and $1,226,286$ for the postpaid segment. The number of edges connecting customers in the social network derived from the CDR data equals $2,414,945$ for the prepaid segment and $3,706,384$ for the postpaid segment. The class distribution is very skewed, which is typically the case for customer churn data sets as discussed in Section 5.2, with on average only 0.52% and 0.57%



(a) Weighted example network.

(b) Equivalent distance network.

(c) Second order weights.

(d) Second order distances.

Figure 5.7: The weighted example network around node A and the equivalent distance network with first (upper panels) and second order (lower panels) weights (left panels) and distances (right panels). Remark that the width and length of links in these networks are not representative for the actual values of the edge weights or distances.

of the customers in respectively the prepaid and postpaid segment that churn each month. Furthermore, 119 and 58 attributes, both local and network variables, were provided for each customer in respectively the prepaid and postpaid customer base.

Three months of data, M2 to M4, have been selected to induce a call graph and training labels, indicating which subscribers churned during this

period. The call graph and the training labels are used by the relational classifiers to predict the test labels, i.e., to predict which subscribers will churn in month M5 immediately after the period M2 to M4. As indicated in Section 5.3, in order to correctly handle the time dimension, both the CDRN and NLB classifier require a previous time frame in order to calculate the class vectors. This previous time frame consists of the months M1 to M3, with test labels of month M4. Furthermore, the data attributes of month M2 and churn labels of month M3 were used to train a non-relational model. The induced model was then applied to predict churn in month M5 using the data attributes of month M4. Remark that the predictions that are made for month M5 result in a strict out-of-time evaluation of the performance of the induced models, which provides a correct indication of how a model would perform in a real life setting, since no data of month M5 have been used in training the models.

5.5.2 Results and discussion

A selection of non-relational classification techniques is evaluated with and without network variables to assess the impact of network variables on the predictive power of a CCP model. Next, the applicability of relational learning techniques for CCP modeling is assessed, including the impact of collective inference procedures and upgrading the network order, and compared to non-relational classifiers. Finally, ensembles of relational and non-relational classifiers are experimentally evaluated and compared to the stand-alone base classifiers.

Non-relational classification with network variables

Non-relational CCP models were built by applying five non-relational classification techniques on the provided data attributes, i.e., Alternating Decision Trees (ADT) (Freund and Trigg, 1999), Bagging (Bag) (Breiman, 1996), Random Forests (RF) (Breiman, 2001), Bayesian Networks (BN), and Logistic Regression (Logit). The first four of these techniques constitute the top ranked non-relational classifiers for churn prediction in the telco industry as found in a large benchmarking experiment reported in Verbeke et al. (2011a), by evaluating the experiments using either the maximum profit measure or top decile lift. Logistic regression on the other hand is considered to be a general industry standard, and was not found to be

	Prepaid					Postpaid				
	Logit	ADT	RF	BAG	BN	Logit	ADT	RF	BAG	BN
Lift at 0.5%										
Without NV	4.35	**4.78**	3.43	**5.74**	**4.81**	4.30	3.34	1.99	2.27	6.27
With NV	**4.94**	4.75	**3.47**	4.85	4.37	**5.84**	**5.56**	**2.88**	**4.52**	**6.83**
Lift at 1%										
Without NV	3.93	3.64	2.70	**4.28**	**4.16**	3.56	3.39	1.98	3.33	5.40
With NV	**3.99**	**3.69**	**2.74**	3.99	3.82	**5.32**	**5.17**	**2.50**	**3.58**	**5.84**
Lift at 5%										
Without NV	3.11	**2.82**	1.64	2.71	3.13	2.44	3.43	1.79	3.22	3.67
With NV	**3.14**	2.08	**1.67**	**3.09**	**3.19**	**3.82**	**3.71**	**1.82**	**3.69**	**3.75**
Lift at 10%										
Without NV	2.72	**2.51**	1.30	2.25	2.80	1.86	2.74	**1.63**	2.71	2.93
With NV	**2.74**	1.93	**1.32**	**2.78**	**2.82**	**3.06**	**3.41**	1.55	**3.19**	**2.97**

Table 5.3: Results of the non-relational classification techniques with and without network variables (NV) in terms of lift.

statistically significantly outperformed in the benchmarking experiments.

The data attributes provided by the telco operator contain both local and network variables as defined in Section 5.3.2. Examples of network variables are the number of contacts between a customer and previous churners, the number of contacts between a customer and customers of particular competing operators, etc. Non-relational CCP models were induced using the five selected classifiers on the top 25 ranked attributes, excluding network variables in the first series of experiments and including network variables in the second series. Identical to a procedure applied in Verbeke et al. (2011a), the attributes were ranked using a chi squared filter to retain the most relevant attributes and facilitate learning. Comparing the results of the first and the second series of experiments allows to assess the impact of network attributes on the predictive power of a non-relational CCP model.

Table 5.3 reports the results of the experiments in terms of top 0.5%, 1%, 5%, and 10% lift for the prepaid and postpaid case study. As can be seen from the table, including network variables clearly boosts the performance of the non-relational classification model in the postpaid case study. The models including network variables consistently outperformed the models without network variables. The results of the prepaid case study on the other hand are less conclusive about the impact of network variables, but

indicate a slight improvement in lift for larger top fractions (top 5% and 10%).

In the postpaid case study three different techniques yield the highest lift depending on the selected top fraction. This illustrates the shortcomings of using lift as a measure to evaluate classification models in a customer churn prediction setting, and motivates the application of the MPC measure as introduced in Section 5.2.3. When applying the MPC measure to evaluate the predictions of the non-relational classification models, network variables are found to boost the profit per customer, since fractions roughly between 3% and 10% are selected, for which higher lift is obtained when including network variables, as indicated by Table 5.3.

Furthermore, the results in Table 5.3 show that the models in the postpaid case study generally obtain better predictive power than the models in the prepaid case study. In a prepaid setting no subscription records are available and thus less information is available to predict churn, yielding lower lift figures. Moreover, in the postpaid case study a significant fraction of churn is explained by an attribute indicating the end-of-contract, which is not available in the prepaid case. The Logit model with network variables will be used in the next sections as the base non-relational model, because of its widespread use in the industry as well as in the research community as a benchmark model.

Relational learning for customer churn prediction

Table 5.4 reports the results of the relational classifiers presented in Section 5.3 for both the prepaid and the postpaid case study. Except for lift at small top fractions in the prepaid case, the relational classifiers generally yield weaker predictive power than the non-relational classifiers. This is not surprising given the fact that a much smaller amount of information is used to build these models. Only the amount of communication between the customers of the operator and a label indicating churn serves as input to the relational learners. Incorporating information in the relational learning process related to communication with customers of competing operators, as included in the form of network variables in the non-relational models discussed in the previous section, may be an interesting topic for future research to improve the performance of stand-alone relational models.

However, although less powerful, relational learners may be useful since they detect different segments or types of churners than non-relational clas-

		Prepaid					
	CI	**-**	**GS SL**	**RL**	**RL SA**	**IC SL**	**SPA CI**
	RC	Lift at 0.5%					
NO=1	**WVRN**	4.58	2.40	3.48	3.54	1.51	2.74
	CDRN	4.58	1.06	1.00	1.00	1.00	4.16
	NLB	4.58	1.00	1.00	1.00	1.00	1.00
	SPA RC	**4.91**	1.00	1.73	1.56	1.00	1.50
NO=2	**WVRN**	**4.65**	2.29	3.16	3.34	1.31	2.80
	CDRN	**4.65**	1.03	1.37	1.37	1.00	3.41
	NLB	**4.65**	1.33	1.37	1.37	1.00	2.65
	SPA RC	4.02	1.04	1.53	1.57	1.00	1.47
		Lift at 1%					
NO=1	**WVRN**	**4.24**	1.95	2.62	2.70	1.25	2.18
	CDRN	3.70	1.05	1.00	1.00	1.00	3.41
	NLB	**4.24**	1.00	1.00	1.00	1.00	1.00
	SPA RC	3.78	1.00	1.73	1.54	1.00	1.50
NO=2	**WVRN**	**3.91**	1.89	2.52	2.62	1.16	2.12
	CDRN	3.87	1.01	1.37	1.37	1.00	2.74
	NLB	**3.91**	1.33	1.37	1.37	1.00	2.46
	SPA RC	3.24	1.02	1.53	1.55	1.00	1.47
		Lift at 5%					
NO=1	**WVRN**	**1.94**	1.56	1.79	1.79	1.05	1.70
	CDRN	1.52	1.02	1.00	1.00	1.00	1.77
	NLB	1.71	1.00	1.00	1.00	1.00	1.00
	SPA RC	1.53	1.00	1.58	1.53	1.00	1.50
NO=2	**WVRN**	2.44	1.50	1.70	1.74	1.03	1.58
	CDRN	1.56	1.01	1.37	1.37	1.00	1.70
	NLB	**2.48**	1.09	1.37	1.37	1.00	2.08
	SPA RC	2.01	1.00	1.53	1.54	1.00	1.47
		Lift at 10%					
NO=1	**WVRN**	1.45	1.41	1.48	1.46	1.02	1.53
	CDRN	1.25	1.02	1.00	1.00	1.00	1.41
	NLB	1.34	1.00	1.00	1.00	1.00	1.00
	SPA RC	1.25	1.00	**1.56**	1.52	1.00	1.50
NO=2	**WVRN**	1.68	1.25	1.60	1.63	1.01	1.51
	CDRN	1.27	1.00	1.37	1.37	1.00	1.37
	NLB	**1.92**	1.04	1.37	1.37	1.00	1.81
	SPA RC	1.48	1.00	1.53	1.54	1.00	1.47

		Postpaid					
	CI	**-**	**GS SL**	**RL**	**RL SA**	**IC SL**	**SPA CI**
	RC	Lift at 0.5%					
NO=1	**WVRN**	2.79	1.31	1.81	2.00	1.43	1.50
	CDRN	2.79	1.21	1.07	1.06	1.00	2.65
	NLB	2.79	1.00	1.01	1.01	1.00	1.02
	SPA RC	**3.69**	1.00	1.19	1.17	1.00	1.09
NO=2	**WVRN**	**3.15**	1.25	1.37	1.46	1.15	1.29
	CDRN	**3.15**	1.12	1.06	1.06	1.00	2.22
	NLB	**3.15**	1.00	1.00	1.00	1.00	1.00
	SPA RC	3.12	1.00	1.23	1.20	1.00	1.08
		Lift at 1%					
NO=1	**WVRN**	2.76	1.23	1.59	1.68	1.21	1.33
	CDRN	2.76	1.11	1.07	1.06	1.00	2.39
	NLB	2.76	1.00	1.01	1.01	1.00	1.02
	SPA RC	**3.31**	1.00	1.17	1.15	1.00	1.09
NO=2	**WVRN**	**2.89**	1.24	1.31	1.38	1.08	1.25
	CDRN	**2.89**	1.06	1.06	1.06	1.00	1.96
	NLB	**2.89**	1.00	1.00	1.00	1.00	1.00
	SPA RC	2.72	1.00	1.19	1.18	1.00	1.08
		Lift at 5%					
NO=1	**WVRN**	**1.96**	1.12	1.23	1.25	1.04	1.18
	CDRN	1.60	1.02	1.07	1.06	1.00	1.61
	NLB	1.57	1.00	1.01	1.01	1.00	1.02
	SPA RC	1.81	1.00	1.16	1.13	1.00	1.09
NO=2	**WVRN**	1.78	1.15	1.21	1.22	1.01	1.18
	CDRN	1.62	1.01	1.06	1.06	1.00	1.54
	NLB	1.46	1.00	1.00	1.00	1.00	1.00
	SPA RC	**1.81**	1.00	1.16	1.16	1.00	1.08
		Lift at 10%					
NO=1	**WVRN**	**1.45**	1.09	1.16	1.19	1.02	1.12
	CDRN	1.28	1.01	1.07	1.06	1.00	1.30
	NLB	1.27	1.00	1.01	1.01	1.00	1.02
	SPA RC	1.38	1.00	1.15	1.13	1.00	1.09
NO=2	**WVRN**	1.55	1.13	1.17	1.18	1.01	1.15
	CDRN	1.31	1.01	1.06	1.06	1.00	1.30
	NLB	1.24	1.00	1.00	1.00	1.00	1.00
	SPA RC	**1.57**	1.00	1.16	1.15	1.00	1.08

Table 5.4: Results of the experiments for the two case studies (prepaid and postpaid), combining a relational classifier (RC) and a collective inference (CI) procedure for the network neighborhood order (NO) equal to one and two. The highest lift per segment is indicated in bold, and the overall highest lift per top fraction is underlined.

sifiers. Figure 5.8 plots the fraction of the churners detected by a non-relational model (Logit with network variables) that is not detected by the relational models (left panel), and vice versa (right panel), as a function of the selected top fraction of customers. The selected top fraction of customers on the x-axis is the same for both the non-relational and the relational models. For instance, when selecting the top 10% of customers with the highest predicted probabilities to churn as indicated by the relational models, then on average about 80% of these churners are not included in the top 10% selected by the non-relational model.

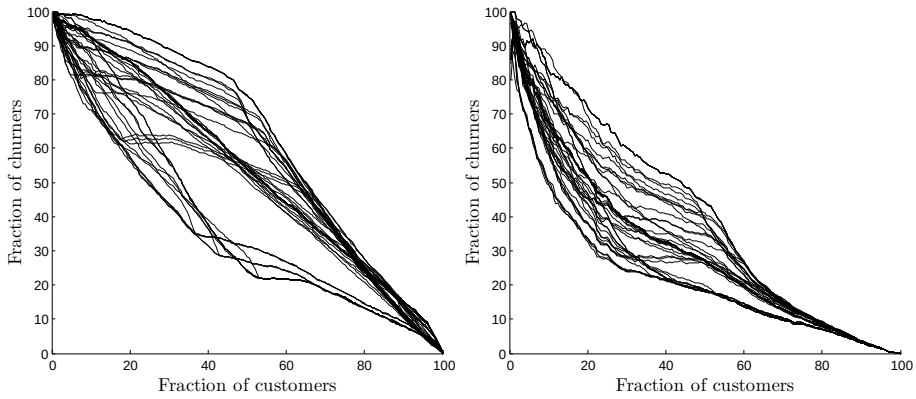


Figure 5.8: The fraction of the churners detected by the non-relational model (Logit) that is not detected by the relational classification models reported in Table 5.4 as a function of the selected fraction of customers with the highest predicted probability to churn (left panel), and vice versa (right panel), for the prepaid segment.

The reason why a relational and a non-relational model detect different segments or types of churners, may be explained by the fact that only a limited fraction of churn events is accounted for by social network effects. These *social churners* are the only churners that can possibly be detected by the relational learners, which also explains the weaker lift of these models for large top fractions (top 5% and 10%) as reported in Table 5.4. In case of the postpaid segment, the fraction of social churners is even smaller than in case of the prepaid segment, explaining the difference in lift figures between both segments. On the other hand, because the social churners only constitute a relatively small fraction within the already small fraction of

churners, non-relational classifiers are not able to detect this type of churn, since the related patterns are not sufficiently or not explicitly present in the attribute value representation of the data in order for a non-relational learner to incorporate such a pattern in the resulting classification model. Non-relational learners appear to incorporate other patterns in the model, which are more prevalent in the data and have better predictive power.

The complementarity of relational and non-relational classifiers with regards to their ability to detect different segments of churners, opens opportunities for combining a relational and a non-relational model to obtain a classification model with increased predictive power. The next section presents the results of the different approaches to combine a relational and a non-relational model as discussed in Section 5.3.3.

A second main finding of Table 5.4 concerns the impact of upgrading the network neighborhood order on the predictive power of the relational classification model. As can be seen from Table 5.4, for small top fractions (0.5 and 1%) the best predictive power was obtained for network neighborhood order equal to one, both in the prepaid and postpaid case study. However, for top 5% and 10% in prepaid and top 10% in postpaid, predictive power increased when upgrading the network neighborhood order to two[5].

Upgrading the network neighborhood order induces noise with regards to first order social network effects. This results in decreased predictive power when assessing lift at small top fractions. However, as explained above, only a very small fraction of the churners are social churners and are directly (i.e., within their neighborhood of order 1) connected to previous churners, leading to poor lift at top 5% and 10%. The weight product to upgrade the network neighborhood order, as developed in the previous section, was specifically designed to handle the skewed class distribution and to improve the overall classification performance, i.e. at large top fractions. The increased lift figures for top 5% and 10% as reported in Table 5.4 indicate that the proposed approach effectively functions, and allows to induce classification models that lead to a better overall classification of the customers at the cost of a small decrease in classification performance with regards to the customers with the highest predicted probabilities to churn.

In the postpaid segment only a minor increase in lift is obtained by upgrading the network neighborhood order. On the other hand, including net-

[5]Further upgrading the network neighborhood to order three and higher did not yield improved classification performance.

		Prepaid				Postpaid			
	Lift at	**0.5%**	**1%**	**5%**	**10%**	**0.5%**	**1%**	**5%**	**10%**
NO=1	**WVRN**	5.08	**4.74**	**3.47**	**2.90**	**5.84**	**5.48**	3.84	3.08
	CDRN	5.08	**4.74**	3.37	2.84	**5.84**	**5.48**	3.84	3.08
	NLB	5.08	**4.74**	3.37	2.89	**5.84**	**5.48**	3.84	3.08
	SPA RC	**5.49**	4.64	3.28	2.85	**5.84**	**5.48**	**3.92**	**3.11**
NO=2	**WVRN**	5.46	4.67	3.35	2.86	**5.84**	**5.48**	3.88	3.09
	CDRN	5.46	4.67	3.35	2.85	**5.84**	**5.48**	3.88	3.09
	NLB	5.46	4.67	3.35	2.86	**5.84**	**5.48**	3.88	3.09
	SPA RC	5.27	4.31	3.23	2.79	**5.84**	**5.48**	3.87	3.08

Table 5.5: Results of the experiments in terms of top 0.5, 1, 5, and 10% lift, for combining a non-relational logistic regression classification model with relational classifiers for a network neighborhood order (NO) equal to one and two, using a parallel model setup. The highest lift per segment is indicated in bold, and underlined if better than the lift of the stand-alone relational and non-relational model.

work variables in a non-relational network model consistently improved the predictive power, as shown by the results reported in Table 5.3 in the previous section. This indicates that a significant fraction of churn is explained by social network effects, but mainly competing operator traffic. This confirms that incorporating such information within a relational model, as indicated above, is a prime issue for future research.

A final finding from Table 5.4 concerns collective inference procedures, which clearly have a negative impact on the classification performance in a customer churn prediction setting. Therefore in the next section we will not consider CI procedures when combining relational and non-relational classifiers. The bad performance of collective inference procedures is due to the large amount of noise they introduce to the resulting predictions by spreading the impact of each node and smoothing the predictions over a wide part of the network. This in fact stems from their initial intent and use, since these procedures were initially designed for image restoration (Chakrabarti et al., 1998).

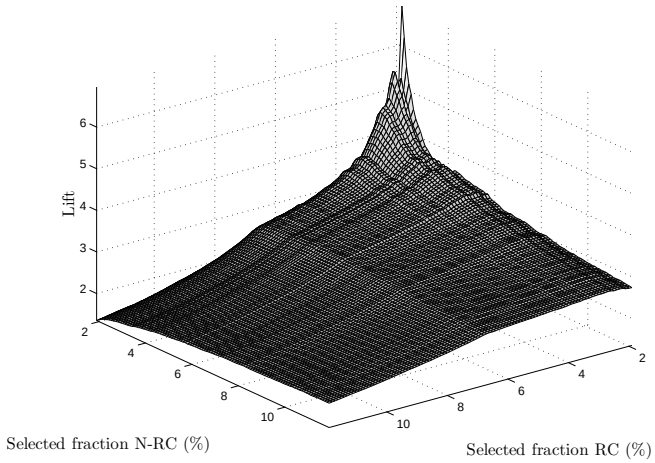
Combined relational and non-relational classification model

Both the cascading and stacking approaches as shown in Figures 5.4(a), 5.4(b), and 5.5(a) have been found unable to improve the predictive power of the stand-alone non-relational model. The main reason lies in the fact

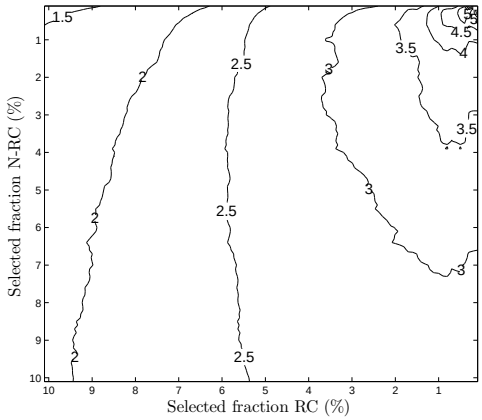
that a local and a network model detect different types of churners. The initialization, automated propositionalization, and stacking approaches favor customers with high *average* probabilities to churn (averaged over the relational and non-relational model). However, churners detected by the network model are assigned a fairly low probability to churn by the non-relational model, and vice versa. This results in a medium average probability to churn, and consequently relatively poor classification power of the integrated combination approaches. The parallel setup however selects customers indicated to have a high probability to churn either by the non-relational model or by the relational model, yielding improved predictive power compared to the stand-alone models. Table 5.5 reports the results of the parallel combination approach for different relational learners and the Logit non-relational model with network variables. The improvement in predictive power of the combined model in the postpaid case study is only faint, because the stand-alone relational learner is weaker compared to the prepaid case.

The parallel model approach yields a three dimensional lift curve, indicating the lift for each possible combination of selected top fractions of the relational and non-relational model. The three dimensional lift curve of the combined Logit-WVRN model is shown in Figure 5.9(a), and the related iso-lift curves are shown in Figure 5.9(b). A two dimensional lift curve, allowing a straight comparison to the lift curves of the constituting stand-alone models, can be derived from the three dimensional curve by selecting the maximum lift for each resulting total fraction selected by the parallel model, i.e., by solving the combinatorial optimization problem mentioned in the final paragraph of Section 5.3.3. The total selected fraction differs from the sum of the selected fractions of the base models because some overlap exists between these two fractions, as shown by Figure 5.10(a). Figure 5.10(b) plots the lift curves of the non-relational Logit model, the WVRN relational model, and the combined parallel model Logit-WVRN for the prepaid case study. From this figure can be seen that the combined model clearly improves the predictive power of the stand-alone relational model and the stand-alone non-relational model.

Although the gain in lift appears to be minor, it may well result in a significant increase in profit. Figure 5.11 plots the increase in MPC of the combined Logit-WVRN model for $NO = 1$ (top left panel) and for $NO = 2$ (top right panel), compared to the stand-alone Logit model with network



(a) Lift curve of parallel model.



(b) Contour plot of lift curve.

Figure 5.9: Three dimensional lift curve (top panel) with contour plot (bottom panel) of a relational model (WVRN) and a non-relational model with network variables (Logit).

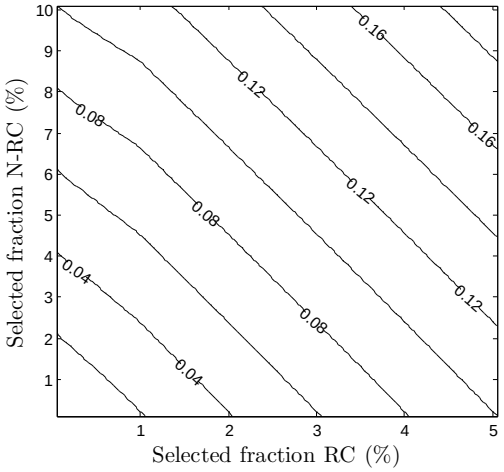
variables for the prepaid case. Depending on the values of the retention rate and the CLV, and with $\delta = 1$ and $c = 0.05$, the increase in MPC ranges between 0 and 18 cents per customer and per retention campaign, which may yield significant profit gains for customer bases typically consisting of millions of customers, and with retention campaigns typically executed each month of the year.

Furthermore, Figure 5.11(c) shows the difference in maximum profit between the parallel model with and without second order effects. It is found that including non-Markovian social network effects within relational classification for customer churn prediction generates additional profits because of improved classification power, which, depending on the value of the retention rate and the CLV, amount to 6 cents per customer in the customer base, per retention campaign. Finally, Figure 5.11(d) demonstrates to what extent these novel relational learning techniques improve the traditionally used CCP models, showing the percentage increase in MPC of the parallel model compared to the stand-alone Logit model. The profit gains are in the order of 20 to 30 percent, and even higher for small absolute values of the MPC, or equal to zero when the MPC equals zero (i.e., when a retention campaign would bear loss instead of profit).

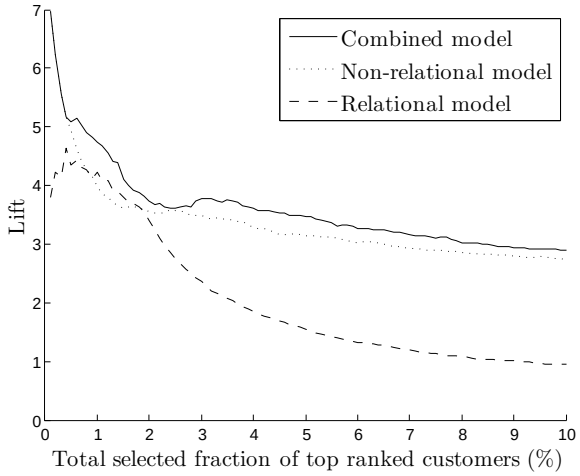
The optimal fraction of included customers in the retention campaign when maximizing the profit, ranges between 0 to 1% for very small values of the retention rate and CLV, and up to 15% for high retention rates and CLV. Hence, more customers should be included in a retention campaign when customers are retained more easily (i.e., when the retention rate is higher), and when it is more profitable to retain customers (i.e., when the average CLV is higher). For the same reason the absolute difference in MPC between the parallel model and the stand-alone non-relational Logit model increases for a higher retention rate and CLV.

5.6 Conclusions and future research

This chapter develops a range of new and adapted relational learning algorithms for customer churn prediction using social network effects, designed to handle the massive size of the call graph, the time dimension, and the skewed class distribution typically present in a customer churn prediction setting. Furthermore, an innovative approach to incorporate non-Markovian network effects within relational classifiers is presented, i.e., the

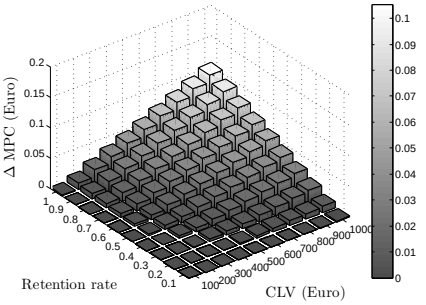


(a) Total selected fraction.

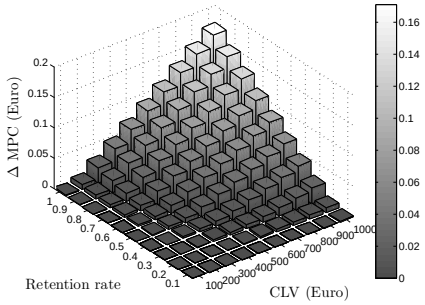


(b) Lift curves.

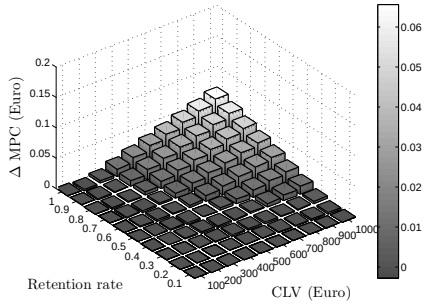
Figure 5.10: Total selected fraction of customers by the parallel model as a function of the selected fractions of the relational and non-relational model (top panel); the lift curves of a non-relational (Logit), a relational (WVRN), and a combined model (Logit-WVRN) for the prepaid case study (bottom panel).



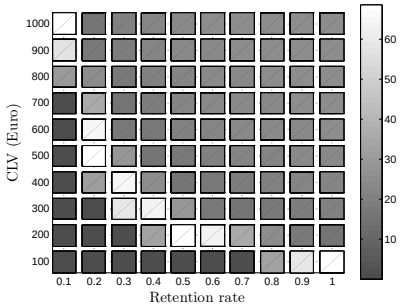
(a) Increase in MPC, 1st order WVRN.



(b) Increase in MPC, 2nd order WVRN.



(c) Difference in MPC increase 2nd-1st order.



(d) Percentage increase in MPC.

Figure 5.11: Comparison in terms of the MPC measure of a parallel model incorporating a traditional non-relational Logit model and a relational WVRN model, with 1st order network effects (top left panel) and with 1st and 2nd order effects (top right panel), compared to the stand-alone Logit model. The bottom right bar plot shows the percentage increase in MPC of the parallel model compared to the stand-alone Logit model.

weight product, which allows to upgrade the network neighborhood of the weight matrix. The weight product can be applied complementary to existing relational classifiers, and is shown to be the equivalent operation for weighted networks to the distance product for distance networks. Finally, a novel parallel model setup to combine a relational and non-relational classification model is introduced, which selects a top-fraction of the customers with the highest predicted probabilities to churn of both models.

The performance of the newly proposed techniques is experimentally tested, and a new profit driven evaluation methodology is applied to assess the results of a real life case study on a large scale telco data set, containing both networked (call detail record data) and non-networked (customer related) information about millions of subscribers. The experiments indicate the existence of a significant impact of social network effects on the churn behavior of telco subscribers. Interestingly, also non-Markovian social network effects are observed: the churn behavior of not only the friends, but also the friends of friends have an impact on the churn behavior of telco subscribers.

Relational and non-relational classifiers, which are built using respectively networked and non-networked information, are found to detect different groups or types of churners, hinting towards a great potential of social network analysis for customer churn prediction. A propositionalization approach to incorporate social network information within a non-relational model by including network variables yields the best performing integrated CCP model. However, the novel parallel modeling setup, resulting in a non-integrated model, outperforms all other approaches to combine a relational and non-relational model and generates significant gains in profit compared to integrated models, including the propositionalization approach. Hence, the main finding of this chapter with strong consequences for the practice of customer relationship management is that applying relational classifiers in combination with the current generation of CCP models can generate significant profit gains. This results from the fact that relational and non-relational CCP models detect different types of churners. Finally, including second order network effects by upgrading the network neighborhood using the weight product, is shown to improve classification power and to further increase the generated profits compared to a model that restricts the impact of the network to first order effects.

Chapter 6

Conclusions and future research

If you torture the data long enough, it will confess.

Ronald Harry Coase (1910)

In this dissertation a range of new algorithms have been developed and applied in real life case studies. As such, this doctoral thesis contributes both from a theoretical and an application point of view. The main findings and conclusions will be recapitulated in the first section of this final chapter. The innovative character of the presented approaches opens new perspectives towards future research, as will be discussed in Section 6.2.

6.1 Conclusions

The introductory chapter of this dissertation discussed three main requirements that are applicable to classification models, i.e., predictive power, comprehensibility, and justifiability. Classification models that meet these requirements are called *acceptable* for implementation. Moreover, the business driven research problem of customer churn prediction in the telecommunication sector was introduced. Customer churn prediction is characterized by the massive size of the customer base, the skewed class distribution

of churners vs. non-churners, and the time dimension. Together the acceptability requirements and the characteristics of customer churn prediction constitute a framework that links the various chapters of this dissertation and motivates the contributions.

In the second chapter an overview of the state-of-the-art in customer churn prediction is provided. An extensive literature review of the most important contributions in the scientific literature indicates that a wide range of data mining techniques have been tested on their ability to predict customer churn. However, the comprehensibility and justifiability aspects of CCP models have mainly been ignored. CCP models should be both accurate and comprehensible to improve the efficiency of customer retention campaigns, respectively in order to select the customers that are the most likely to churn (accuracy requirement), and to understand the reasons why a CCP model indicates these customers to have a high probability to attrite, and as such to uncover the main drivers for churn (comprehensibility requirement). This, on its turn, allows to develop effective retention offers. Moreover, as a general requirement, CCP models need to be justifiable in order to be acceptable for implementation.

Therefore, two advanced rule induction techniques, AntMiner+ and ALBA, have been applied to predict churn. Both techniques explicitly seek to induce accurate as well as comprehensible rule sets. The results are benchmarked to C4.5, Ripper, SVM, and logistic regression. It is shown that ALBA, combined with Ripper or C4.5, results in the highest accuracy, while sensitivity is the highest for C4.5 and Ripper applied on an oversampled data set. AntMiner+ yields a lower sensitivity, but allows to include domain knowledge and results in comprehensible rule sets which are much smaller than the rule sets induced by C4.5. Ripper also results in small and comprehensible rule sets, but may lead to an unintuitive model that violates domain knowledge. The comprehensibility of a churn prediction model is important since it facilitates the interpretation and the practical use of a model for marketing purposes. Comprehensibility also allows to check the concordance of a model with domain knowledge, which is of great importance since the intuitiveness of a model determines whether or not a model will be accepted by the end-users, and as such whether the model will effectively serve its purpose. AntMiner+ allows to include domain knowledge by imposing monotonicity constraints, leading to intuitive correct models that are still comprehensible and accurate, as indicated by the results of the

experiments.

The results of the second chapter illustrate that in order to be acceptable for implementation many real world applications require classification models to be in line with domain knowledge and to satisfy monotone relations between predictor variables and the target class. However, existing techniques to induce monotone classification models either yield poor classification performance, or are only able to handle a binary class variable (e.g., AntMiner+). Therefore, in the third chapter the novel RULEM algorithm to induce monotone ordinal rule based classification models is developed. A main asset of the proposed approach is its complementarity with existing rule-based classification techniques. Since monotonicity is guaranteed during a postprocessing step, the RULEM approach can be combined with any rule- or tree-based classification technique. In a first step, the RULEM algorithm checks whether a rule set or decision tree violates the imposed monotonicity constraints. Existing violations are resolved in a second step by inducing a set of additional rules which enforce monotone classification. The algorithm is able to handle non-monotonic noise, and can be applied to both partially and totally monotone problems with an ordinal target variable.

Based on the RULEM algorithm, two novel justifiability measures are introduced. The RULEMS and RULEMF measures allow to calculate the extent to which a classification model is in line with domain knowledge expressed in the form of monotonicity constraints. Both measures provide an intuitive indication of the justifiability of a rule set, and can be calculated in a fully automated manner.

An extensive benchmarking experiment has been set up to test the impact of the RULEM approach on the predictive power and the comprehensibility of the resulting rule set. The results of the experiments indicate that the proposed approach preserves the predictive power of the original rule induction techniques while guaranteeing monotone classification, at the cost of a small increase in the size of the rule set. Hence, the RULEM algorithm is shown to yield accurate, comprehensible, and justifiable rule based classification models. The predictive power of the final rule set therefore depends on the selected rule induction technique that RULEM is combined with.

CCP models are typically evaluated using statistically based performance measures, such as for instance top decile lift or AUC. However, as shown in Chapter 4, this may lead to suboptimal model selection, and conse-

quently to a loss in profits. Therefore, in the first part of Chapter 4, a novel, profit centric performance measure is developed. Optimizing the fraction of included customers with the highest predicted probabilities to attrite yields the maximum profit that can be generated by a retention campaign. Since reducing the costs or losses associated with customer churn is the main objective of a CCP model, this chapter advocates the use of the maximum profit (i.e., the maximum reduction in costs or losses) that can be generated by using the output of the model as a measure to evaluate CCP models.

In the second part of the fourth chapter a large benchmarking experiment is conducted, including twenty-one state-of-the-art predictive algorithms that are applied on eleven data sets from telco operators worldwide. The benchmarking experiment allows to rigorously analyze and assess the impact of classification technique, oversampling, and input selection on the performance of a CCP model. The results of the experiments are tested using the appropriate test statistics, and evaluated using both the novel profit centric based measure and statistical performance measures, leading to the following conclusions:

- Applying the maximum profit criterion and including the optimal fraction of customers in a retention campaign leads to substantially different outcomes. Furthermore, the results of the experiments provide strong indications that the use of the maximum profit criterion can have a profound impact on the generated profits by a retention campaign.

- Secondly, the effect of oversampling on the performance of a CCP model strongly depends on the data set and the classification technique that is applied, and can be positive or negative. Therefore, we recommend to adopt an empirical approach, and as such to consistently test whether oversampling is beneficial.

- Third, the choice of classification technique significantly impacts the predictive power of the resulting model. Alternating Decision Trees yielded the best overall performance in the experiments, although a large number of other techniques were not significantly outperformed. Hence, other properties of modeling techniques besides the predictive power have to be taken into account when selecting a classification technique, such as comprehensibility, justifiability, and operational efficiency. Rule induction techniques, decision tree approaches, and clas-

sical statistical techniques such as logistic regression and Naive Bayes or Bayesian Networks score well on all three aspects, and result in a powerful, yet comprehensible model that is easy to implement and operate. Therefore these techniques are recommended to be applied for CCP modeling. Comprehensibility or interpretability is an important aspect of a classifier which allows the marketing department to extract valuable information from a model, in order to design effective retention campaigns and strategies. The comprehensibility of a model however also depends on the number of variables included in a model. Clearly a model including ten variables is easier to interpret than a model containing fifty variables or more.

- This leads to a fourth conclusion, i.e., input selection is crucial to achieve good predictive power, and six to eight variables generally suffice to predict churn with high accuracy. Consequently, from an economical point of view it may be more efficient to invest in data quality, than in gathering an extensive range of attributes capturing all the available information on a customer. Furthermore, the input selection procedure has shown that usage attributes are the most predictive kind of data. However, also socio-demographic data, financial information, and marketing related attributes are indispensable sources of information to predict customer churn. Moreover, marketing related attributes such as the hand set that is provided to a customer by the operator, are important sources of actionable information to design effective retention campaigns.

- Finally, this chapter also provides benchmarks to the industry to compare the performance of their CCP models.

The fifth chapter develops a range of new and adapted relational learning algorithms for customer churn prediction using social network effects, designed to handle the massive size of the call graph, the time dimension, and the skewed class distribution typically present in a customer churn prediction setting. Furthermore, an innovative approach to incorporate non-Markovian network effects within relational classifiers is presented, i.e., the weight product, which allows to upgrade the network neighborhood of the weight matrix. The weight product can be applied complementary to existing relational classifiers, and is shown to be the equivalent operation for weighted networks to the distance product for distance networks. Finally, a

novel parallel model setup to combine a relational and non-relational classification model is introduced, which selects a top-fraction of the customers with the highest predicted probabilities to churn of both models.

The performance of the newly proposed techniques is experimentally tested, and the profit driven evaluation methodology presented in Chapter 4 is applied to assess the results of two real life case studies on large scale telco data sets, containing both networked (call detail record data) and non-networked (customer related) information about millions of subscribers. The experiments indicate the existence of a significant impact of social network effects on the churn behavior of telco subscribers. Interestingly, also non-Markovian social network effects are observed: the churn behavior of not only the friends, but also the friends of friends have an impact on the churn behavior of telco subscribers.

Relational and non-relational classifiers, which are built using respectively networked and non-networked information, are found to detect different groups or types of churners, hinting towards a great potential of social network analysis for customer churn prediction. A propositionalization approach to incorporate social network information within a non-relational model by including network variables yields the best performing stand-alone CCP model. However, the novel parallel modeling setup, resulting in a non-integrated model, outperforms all other approaches to combine a relational and non-relational model and generates significant gains in profit compared to integrated models, including the propositionalization approach. Hence, the main finding of this chapter with strong consequences for the practice of customer relationship management is that applying relational classifiers in combination with the current generation of CCP models can generate significant profit gains. This results from the fact that relational and non-relational CCP models detect different types of churners. Finally, including second order network effects by upgrading the network neighborhood using the weight product, is shown to improve classification power and to further increase the generated profits compared to a model that restricts the impact of the network to first order effects.

6.2 Future research

6.2.1 Profit based evaluation framework for classification models

A first main topic for future research concerns the elaboration of a holistic, profit based evaluation framework for classification models in a business context. The need to understand the relationships among customer metrics and profitability has never been more critical (Gupta, 2006). The maximum profit measure, introduced in Chapter 4 and developed for application in a customer churn prediction setting, can be considered to be a first step in this direction, and has been translated to a direct marketing setting by Martens and Provost (2011). Verbraken et al. (2011b) provides a theoretical underpinning to the MP measure, and inspired by the recently introduced H-measure (Hand, 2009) also develops the *Expected* Maximum Profit measure (EMP). The EMP allows to take into account uncertainty about the main parameter in the MP measure, i.e., the retention rate. The H-measure aims to be a coherent alternative measure for classification performance to the area under the ROC curve, by making explicit assumptions about the misclassification costs. As such, the H-measure allows to take into account uneven misclassification costs when assessing the performance of classification models. The MP and EMP measures on the other hand focus on the benefits associated with correctly classifying an instance. As such, both the MP and EMP measure acknowledge the importance of taking into account how the output of a classification model will be used as an input to subsequent business processes, when assessing and evaluating the performance of a classification model.

In a customer churn prediction setting, a further optimization of the resulting profits can be achieved by integrating CCP models, customer lifetime value models, and response models. The MP measure optimizes the fraction of customers included in a retention campaign, starting from the ranking of customers with the highest probability to churn provided by the CCP model. However, the selection procedure to include customers in a retention campaign could also take into account the response rate and the customer lifetime value of customers, in order to increase the profits generated by the retention campaign:

- Including the customers which are the most likely to accept the retention offer, and to be effectively retained by the campaign, can improve

the retention rate, which has a direct impact on the generated profits. The probability to be retained is likely to be a function of the value of the retention offer. The value of the retention offer therefore is a crucial parameter of the problem, and an integral part of the optimization exercise.

- Including the customers with the highest customer lifetime value in the retention campaign, even with lower probabilities, can increase the revenues that are retained by the retention campaign, and as such the profitability of the campaign.

By combining the predicted probability to churn, the estimated customer lifetime value, and the probability to respond to a retention offer, the profits can be further optimized as a function of the selected customers and the value of the retention offer, which can be set individually for each customer.

6.2.2 Classification models and justifiability

A second important topic for future research concerns the further development and refinement of the RULEM algorithm to guarantee ordinal monotone classification, and the related justifiability measures in Chapter 3:

- The heuristic approach to merge the induced additional rules needs to be optimized. This will allow to further reduce the size of the final rule set. This problem in fact boils down to finding the largest hypercubes with homogenous labeling (different from the default label) in the n-dimensional attribute space, constituted by the additional and original rules. Currently, the largest *string* of hypercubes in a single dimension is merged into a single rule.

- Further analysis and experiments are needed to examine the exact nature of the relation between the C(I)-score, the predictive power, the justifiability, and the induced number of additional rules. The developed heuristic approach requires additional theoretical underpinning, and improvements may be possible with regards to the proposed CI-score in order to further increase the classification accuracy of the resulting total rule set. The induced additional rules already guarantee monotone relations between the predictors and the target variable, but should be able to guarantee in the majority of cases improved predictive power as well.

- Finally, an interesting and challenging topic for future research will be the development of a justifiability measure for non-rule based classification models. The main difficulty with regards to the development of such a measure concerns the non-rectangular boundaries induced by these classifiers. A solution to this problem may exist in the induction of an approximate elementary grid, with the resolution of the elementary cells iteratively determined as a function of the variability of the labeling of the cells, in order to yield a grid with many cells around the boundaries in the attribute space, and few in homogeneously labeled regions. The approximate elementary grid can then be used as input to the RULEM procedure to calculate an approximate measure of the justifiability.

6.2.3 Social network analysis for classification

A third main topic for future research concerns the further exploration of social network information for customer churn prediction, as well as for other business applications. More specifically, further research is required to combine *network* and *local* information in a single model setup. As discussed and illustrated in Chapter 5, typically a separate network and local model are built and then combined. However, an integrated setup would offer more flexibility, and could possibly profit from exploiting both types of information simultaneously, yielding more powerful classification models. This may be achieved by developing a networked representation of local information, which can then be included straightforward within existing or newly developed network classification schemes.

Existing relational classification techniques find their roots in applications such as image restoration, which had a clear impact on the design. Straightforward application of these heuristics, for instance to predict customer churn, may not always be meaningful. Therefore, a new generation of relational classifiers is needed which are developed with social network analysis in mind, possibly taking advantage of, in line with, and complementary to graph theory and social network science (Wasserman and Faust, 1997; Kolaczyk, 2009; Newman, 2010). An interesting idea to be explored in this context is the application of centrality measures for relational classification.

Finally, although explicitly present in a customer churn prediction setting, current relational as well as non-relational classification techniques do not allow to take into account the time dimension in an effective manner,

although dynamical aspects presumably have a major impact on the processes that occur in such a setting. Incorporating the time aspect in a *natural* manner within these techniques, possibly inspired by time series analysis or survival analysis techniques, therefore may improve their predictive power and extend their applicability.

Appendix A

Ripper DK algorithm

Ripper has been introduced in Cohen (1995) as an extended version of the Incremental Reduced Error Pruning (IREP) rule learning algorithm proposed in Fürnkranz and Widmer (1994). The Ripper algorithm can be extended to incorporate monotone relations between attributes and a binary class variable by constraining the rule growing process as indicated in Algorithm 7, similar to Martens et al. (2006).

Assume a binary class variable with two values, i.e., zero and one. The Ripper algorithm induces rules to predict the minority class. If class one is the minority class, then the attributes on which a positive constraint is imposed should not be bounded from above by any attribute test in a rule. Therefore, the conjuncts related to a positively constrained attribute are only allowed to apply the *greater than or equal to* ($\geq$) or *greater than* ($>$) operator. Negative constraints on the other hand lead to attribute tests which are only allowed to apply the *less than or equal to* ($\leq$) or *less than* ($>$) operator. These restrictions are simply reversed when class zero is the minority class. The Ripper DK algorithm has been implemented in the Weka environment, and can be obtained freely from the authors upon request.

Algorithm 7 Pseudo-code of the extended Ripper algorithm to induce a monotone rule set for data set $\mathcal{D}$ with binary target variable

data set $\mathcal{D}$ with binary class variable $\ell \in \{0, 1\}$
$\mathcal{D} = (Pos,Neg)$, with Pos the set of instances of the minority class
rule set $\mathcal{R} = \varnothing$
while $Pos \neq \varnothing$ **do**
  % grow and prune a new rule
  split (Pos,Neg) into $(GrowPos,GrowNeg)$ and $(PrunePos,PruneNeg)$
  rule $r \leftarrow$ GrowRule($GrowPos,GrowNeg$)
  $r : p(\mathbf{x}) \rightarrow 1$
  with $p_e(\mathbf{x}) = \bigwedge_{i=1}^{k}(x_i \ op \ v_{i,e})$
  if minority class = 1 **then**
    if positive constraint on attribute $\mathcal{X}_i$ **then**
      $(x_i \ op \ v_{i,e})$ with $op \in \{>, \geq\}$
    else if negative constraint on attribute $\mathcal{X}_i$ **then**
      $(x_i \ op \ v_{i,e})$ with $op \in \{<, \leq\}$
    else
      $(x_i \ op \ v_{i,e})$ with $op \in \{>, \geq, =, <, \leq\}$
    end if
  else
    % minority class = 0
    if positive constraint on attribute $\mathcal{X}_i$ **then**
      $(x_i \ op \ v_{i,e})$ with $op \in \{<, \leq\}$
    else if negative constraint on attribute $\mathcal{X}_i$ **then**
      $(x_i \ op \ v_{i,e})$ with $op \in \{>, \geq\}$
    else
      $(x_i \ op \ v_{i,e})$ with $op \in \{>, \geq, =, <, \leq\}$
    end if
  end if
  rule $r \leftarrow$ PruneRule(rule r,$PrunePos,PruneNeg$)
  if the error rate of rule on $(PrunePos,PruneNeg)$ exceeds 50% **then**
    return rule set $\mathcal{R}$
  else
    add rule r to $\mathcal{R}$
    remove examples covered by rule r from (Pos,Neg)
  end if
end while
return rule set $\mathcal{R}$

List of Figures

List of Tables

List of Algorithms

Bibliography

Abraham, A., Ramos, V., 2003. Web usage mining using artificial ant colony clustering. In: Proceedings of the Congress on Evolutionary Computation, CEC '03. IEEE Press, pp. 1384–1391.

Agrawal, R., Srikant, R., 1994. Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases. VLDB '94. Morgan Kaufmann Publishers Inc., San Francisco, CA, U.S.A., pp. 487–499.

Ahn, H., Kim, K., 2009. Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach. Applied Soft Computing 9 (2), 599–607.

Altendorf, E., Restificar, E., Dietterich, T., 2005. Learning from sparse data by exploiting monotonicity constraints. In: Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, Edinburgh, U.K.

Askira-Gelman, I., 1998. Knowledge discovery: Comprehensibility of the results. In: Proceedings of the 31st Annual Hawaii International Conference on System Sciences, HICSS '98. Vol. 5. IEEE Computer Society, Washington, D.C., U.S.A., pp. 247–255.

Athanassopoulos, A., 2000. Customer satisfaction cues to support market segmentation and explain switching behavior. Journal of Business Research 47 (3), 191–207.

Au, W., Chan, K., Yao, X., 2003. A novel evolutionary data mining algorithm with applications to churn prediction. IEEE Transactions on Evolutionary Computation 7 (6), 532–545.

Baesens, B., Mues, C., Martens, D., Vanthienen, J., 2009. 50 years of data mining and or: upcoming trends and challenges. Journal of the Operational Research Society 60 (8), 16–23.

Baesens, B., Setiono, R., Mues, C., Vanthienen, J., 2003a. Using neural network rule extraction and decision tables for credit-risk evaluation. Management Science 49 (3), 312–329.

Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., Vanthienen, J., 2003b. Benchmarking state-of-the-art classification algorithms for credit scoring. Journal of the Operational Research Society 54 (6), 627–635.

Barile, N., Feelders, A., 2008. Nonparametric monotone classification with moca. In: Proceedings of the 8th IEEE International Conference on Data Mining, ICDM '08.

Beer, S., 1968. Management science: the business use of operations research. Science and technology series. Aldus, London, U.K.

Ben-David, A., 1995. Monotonicity maintenance in information-theoretic machine learning algorithms. Machine Learning 19 (1), 29–43.

Ben-David, A., Sterling, L., Tran, T., 2009. Adding monotonicity to learning algorithms may impair their accuracy. Expert Systems with Applications 36 (3), 6627–6634.

Berry, M., Linoff, G., 2004. Data Mining Techniques: For Marketing, Sales and Customer Relationship Management. John Wiley and Sons, New York, NY, U.S.A.

Berteloot, K., Verbeke, W., Castermans, G., Van Gestel, T., Martens, D., Baesens, B., 2011. Credit rating migration modeling using macroeconomic indices. International Journal of Forecasting, under review.

Bhattacharya, C., 1998. When customers are members: Customer retention in paid membership contexts. Journal of the Academy of Marketing Science 26 (1), 31–44.

Bishop, C., 1996. Neural networks for pattern recognition. Oxford University Press, Oxford, U.K.

Blau, P., 1977. Inequality and Heterogeneity: A Primitive Theory of Social Structure. NY Free Press, New York, NY, U.S.A.

Blum, C., 2005. Beam-ACO - hybridizing ant colony optimization with beam search: An application to open shop scheduling. Computers and Operations Research 32 (6), 1565–1591.

Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D., 2006. Complex networks: Structure and dynamics. Physics Reports 424, 175–308.

Bolton, R., Lemon, K., Bramlett, M., 2006. The effect of service experiences over time on a supplier's retention of business customers. Management Science 52 (12), 1811–1823.

Bonchi, F., Castillo, C., Gionis, A., Jaimes, A., 2011. Social network analysis and mining for business applications. ACM Transactions on Intelligent Systems and Technology 2 (3), 1–37.

Boryczka, U., 2009. Finding groups in data: Cluster analysis with ants. Applied Soft Computing 9 (1), 61–70.

Breiman, L., 1996. Bagging predictors. Machine Learning 24 (2), 123–140.

Breiman, L., 2001. Random forests. Machine Learning 45 (1), 5–32.

Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. Classification and Regression Trees. Chapman and Hall, New York, NY, U.S.A.

Buckinx, W., Van den Poel, D., 2005. Customer base analysis: Partial defection of behaviourally loyal clients in a non-contractual FMCG retail setting. European Journal Of Operational Research 164 (1), 252–268.

Bullnheimer, B., Hartl, R., Strauss, C., 1999. Applying the ant system to the vehicle routing problem. In: Voss, S., Martello, S., Osman, I., Roucairol, C. (Eds.), Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization.

Burez, J., Van den Poel, D., 2007. CRM at a pay-TV company: Using analytical models to reduce customer attrition by targeted marketing for subscription services. Expert Systems with Applications 32 (2), 277–288.

Burez, J., Van den Poel, D., 2009. Handling class imbalance in customer churn prediction. Expert Systems with Applications 36 (3), 4626–4636.

Chakrabarti, S., Dom, B., Indyk, P., 1998. Enhanced hypertext categorization using hyperlinks. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. pp. 307–319.

Cohen, W., 1995. Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning, ICML. pp. 115–123.

Cohn, D., Atlas, L., Ladner, R., 1994. Improving generalization with active learning. Machine Learning 15 (2), 201–221.

Colgate, M., Stewart, K., Kinsella, R., 1996. Customer defection: A study of the student market in Ireland. International Journal of Bank Marketing 14 (3), 23–29.

Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M., 1994. Ant system for job-shop scheduling. Journal of Operations Research, Statistics and Computer Science 34 (1), 39–53.

Coussement, K., Van den Poel, D., 2008. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. Expert Systems with Applications 34 (1), 313–327.

Cumps, B., Martens, D., De Backer, M., Viaene, S., Dedene, G., Haesen, R., Snoeck, M., Baesens, B., 2009. Inferring rules for business/ict alignment using ants. Information and Management 46 (2), 116–124.

Daniels, H., Kamp, B., August 1999. Application of MLP networks to bond rating and house pricing. In: Neural Computing and Applications. Vol. 8. pp. 226–234.

Daniels, H., Velikova, M., 2006. Derivation of monotone decision models from noisy data. IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews 36 (5), 705–710.

Daniels, H., Velikova, M., 2010. Monotone and partially monotone neural networks. IEEE Transactions on Neural Networks 21 (6), 906–917.

Dasgupta, K., Singh, R., Viswanathan, B., Chakraborty, D., Mukherjea, S., Nanavati, A., Joshi, A., 2008. Social ties and their relevance to churn in mobile telecom networks. In: Proceedings of the 11th international conference on Extending Database Technology: Advances in database technology, EDBT '08. pp. 697–711.

Datta, P., Masand, B., Mani, D., Li, B., 2000. Automated cellular modeling and prediction on a large scale. Artificial Intelligence Review 14, 485–502.

Dejaeger, K., Verbeke, W., Huysmans, J., Mues, C., Vanthienen, J., Baesens, B., 2010a. Rule based predictive models, decision table and tree: an empirical evaluation on comprehensibility. In: Proceedings of the EURO 2010 conference, Lisbon, Portugal.

Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2010b. De kosten van software-ontwikkeling voorspellen. Informatie 52 (9), 8–13.

Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2011a. Benchmarking classification algorithms for software effort prediction. IEEE Transactions on Software Engineering, published online, forthcoming.

Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2011b. Het voorspellen van software-ontwikkelkosten. Informatie, forthcoming.

Delen, D., Walker, G., Kadam, A., 2005. Predicting breast cancer survivability: a comparison of three data mining methods. Artificial Intelligence in Medicine 34 (2), 113–127.

DeLong, E., DeLong, D., Clarke-Pearson, D., 1988. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. Biometrics 44, 837–845.

Dembczynsky, K., Kotlowski, W., Slowinski, R., 2001. Learning rule ensembles for ordinal classification with monotonicity constraints. Fundamenta Informatica, 1001–1016.

Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30.

Di Caro, G., Dorigo, M., 1998. Antnet: Distributed stigmergetic control for communications networks. Journal of Artificial Intelligence Research 9, 317–365.

Dietterich, T., 1998. Approximate statistical test for comparing supervised classification learning algorithms. Neural Computation 10 (7), 1895–1923.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics 26 (1), 29–41.

Dorigo, M., Stützle, T., 2004. Ant Colony Optimization. MIT Press, Cambridge, MA, U.S.A.

Duivesteijn, W., Feelders, A., 2008. Nearest neighbour classification with monotonicity constraints. Lecture Notes in Artificial Intelligence 5211, 301–316.

Dunn, O., 1961. Multiple comparisons among means. Journal of the American Statistical Association 56, 52–64.

Džeroski, S., Lavrač, N., 2001. Relational Data Mining. Kluwer, Berlin, Germany.

Egan, J., 1975. Signal Detection Theory and ROC analysis. Series in Cognition and Perception. Academic Press, New York, NY, U.S.A.

Eiben, A., Koudijs, A., Slisser, F., 1998. Genetic modeling of customer retention. Lecture Notes in Computer Science 1391, 178–186.

Fawcett, T., Provost, F., 1997. Adaptive fraud detection. Data Mining and Knowledge Discovery 1-3, 291–316.

Fayyad, U., Irani, K., 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, IJCAI. Morgan Kaufmann Publishers Inc., Chambéry, France, pp. 1022–1029.

Feelders, A., 2000. Prior knowledge in economic applications of data mining. In: Proceedings of the 4th European conference on principles and practice of knowledge discovery in data bases. Vol. 1910 of Lecture Notes in Computer Science. Springer, New York, NY, U.S.A., pp. 395–400.

Feelders, A., Pardoel, M., 2003. Pruning for monotone classification trees. In: Lecture Notes in Computer Science. Vol. 2810. Springer, New York, NY, U.S.A., pp. 1–12.

Frank, E., Witten, I., 1998. Generating accurate rule sets without global optimization. In: Proceedings of the 15th International Conference on Machine Learning, ICML. pp. 144–151.

Freund, Y., Schapire, R., 1999. Large margin classification using the perceptron algorithm. Machine Learning 37 (3), 277–296.

Freund, Y., Trigg, L., 1999. The alternating decision tree learning algorithm. In: Proceedings of the 16th International Conference on Machine Learning, ICML. pp. 124–133.

Friedman, M., 1940. A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics 11, 86–92.

Fürnkranz, J., Widmer, G., 1994. Incremental reduced error pruning. In: Proceedings of the 11th International Conference on Machine Learning, ICML. pp. 70–77.

Ganesh, J., Arnold, M., Reynolds, K., 2000. Understanding the customer base of service providers: An examination of the differences between switchers and stayers. Journal of Marketing 64 (3), 65–87.

Geman, S., Geman, D., 1984. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. IEEE Transactions on Pattern Analysis and Machine Intelligence 6, 721–741.

Getoor, L., Taskar, B., 2007. Statistical Relational Learning. MIT Press, Cambridge, MA, U.S.A.

Glady, N., Baesens, B., Croux, C., 2009. A modified pareto/NBD approach for predicting customer lifetime value. Expert Systems with Applications 36 (2), 2062–2071.

Gupta, S., 2006. Customer metrics and their impact on financial performance. Marketing science 25 (6), 718–739.

Gupta, S., Hanssens, D., Hardie, B., Kahn, W., Kumar, V., Lin, N., Ravishanker, N., Sriram, S., 2006. Modeling customer lifetime value. Journal of Service Research 9 (2), 139–155.

Gupta, S., Lehmann, D., Stuart, J., 2004. Valuing customers. Journal of Marketing Research 41 (1), 7–18.

Hand, D., 2009. Measuring classifier performance: a coherent alternative to the area under the roc curve. Machine Learning 77 (1), 103–123.

Handl, J., Knowles, J., Dorigo, M., 2006. Ant-based clustering and topographic mapping. Artificial Life 12 (1), 35–61.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning, Data Mining, Inference, and Prediction. Springer, New York, NY, U.S.A.

Hung, S., Yen, D., Wang, H., 2006. Applying data mining to telecom churn management. Expert Systems with Applications 31 (3), 515–524.

Hur, J., Kim, J., 2008. A hybrid classification method using error pattern modeling. Expert Systems with Applications 34 (1), 231–241.

Hwang, H., Jung, T., Suh, E., 2004. An LTV model and customer segmentation based on customer value: a case study on the wireless telecommunication industry. Expert Systems with Applications 26 (2), 181–188.

Jensen, D., Neville, J., 2002. Linkage and autocorrelation cause feature selection bias in relational learning. In: Proceedings of the 19th International Conference on Machine Learning, ICML. pp. 259–266.

Kolaczyk, E., 2009. Statistical Analysis of Network Data, Methods and Models. Springer, New York, NY, U.S.A.

Korn, P., Sidiropoulos, N., Faloutsos, C., Siegel, E., Protopapas, Z., 1998. Fast and effective retrieval of medical tumor shapes. IEEE Transactions on Knowledge and Data Engineering 10 (6), 889–904.

Kramer, S., Lavrač, N., Flach, P., 2001. Relational data mining. Kluwer, Berlin, Germany, Ch. Propositionalization approaches to relational data mining, pp. 262–286.

Krzanowski, W., Hand, D., 2009. ROC curves for continuous data. Chapman and Hall, New York, NY, U.S.A.

Kumar, D., Ravi, V., 2008. Predicting credit card customer churn in banks using data mining. International Journal of Data Analysis Techniques and Strategies 1 (1), 4–28.

Landwehr, N., Hall, M., Eibe, F., 2005. Logistic model trees. Machine Learning 59 (1), 161–205.

Lang, B., 2005. Monotonic multi-layer perceptron networks as universal approximators. Lecture Notes in Computer Science 3697, 31–37.

Lariviere, B., Van den Poel, D., 2005. Predicting customer retention and profitability by using random forest and regression forest techniques. Expert Systems with Applications 29 (2), 472–484.

Larose, D., 2005. Discovering Knowledge in Data: An Introduction to Data Mining. Wiley, New Jersey, NJ, U.S.A.

Lemmens, A., Croux, C., 2006. Bagging and boosting classification trees to predict churn. Journal of Marketing Research 43 (2), 276–286.

Lessmann, S., Baesens, B., Mues, C., Pietsch, S., 2008. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. IEEE Transactions on Software Engineering 34 (4), 485–496.

Li, L., Tang, H., Wu, Z., Gong, J., Gruidl, M., Zou, J., Tockman, M., Clark, R., 2004. Data mining techniques for cancer detection using serum proteomic profiling. Artificial Intelligence in Medicine 32 (2), 71–83.

Lievens, S., De Baets, B., 2010. Supervised ranking in the weka environment. Information Sciences 180, 4763–4771.

Lima, E., Mues, C., Baesens, B., 2009. Domain knowledge integration in data mining using decision tables: case studies in churn prediction. Journal of the Operational Research Society 60 (8), 1096–1106.

Liu, B., Abbass, H., McKay, B., 2003. Classification rule discovery with ant colony optimization. In: Proceedings of the IAT conference. pp. 83–88.

Lu, Q., Getoor, L., 2003. Link-based classification. In: Proceedings of the 20th International Conference on Machine Learning, ICML. pp. 496–503.

Macskassy, S., Provost, F., 2007. Classification in networked data. Journal of Machine Learning Research 8, 935–983.

Madden, G., Savage, S., Coble-Neal, G., 1999. Subscriber churn in the australian isp market. Information Economics and Policy.

Martens, D., 2008a. Building acceptable classification models for financial engineering applications. Ph.D. thesis, K.U.Leuven, Leuven, Belgium.

Martens, D., 2008b. Building acceptable classification models for financial engineering applications. SIGKDD Explorations 10 (2), 3–30.

Martens, D., Baesens, B., Van Gestel, T., Vanthienen, J., 2007a. Comprehensible credit scoring models using rule extraction from support vector machines. European Journal of Operational Research 183 (3), 1466–1476.

Martens, D., Bruynseels, L., Baesens, B., Willekens, M., Vanthienen, J., 2008. Predicting going concern opinion with data mining. Decision Support Systems 45, 765–777.

Martens, D., De Backer, M., Haesen, R., Baesens, B., Mues, C., Vanthienen, J., 2006. Ant-based approach to the knowledge fusion problem. In: Proceedings of the 5th International Workshop on Ant Colony Optimization and Swarm Intelligence. Lecture Notes in Computer Science. Springer, New York, NY, U.S.A., pp. 85–96.

Martens, D., De Backer, M., Haesen, R., Snoeck, M., Vanthienen, J., Baesens, B., 2007b. Classification with ant colony optimization. IEEE Transactions on Evolutionary Computation 11 (5), 651–665.

Martens, D., Provost, F., 2011. Construction and inference of networked data in a bank setting. Working paper CeDER-11-05, Stern School of Business, New York University.

Martens, D., Van Gestel, T., Baesens, B., 2009. Decompositional rule extraction from support vector machines by active learning. IEEE Transactions on Knowledge and Data Engineering 21 (2), 178–191.

Martens, D., Vanthienen, J., Verbeke, W., Baesens, B., 2011. Performance of classification models from a user perspective. Decision Support Systems 51, 782–793.

Masand, B., Piatetsky-Shapiro, G., 1996. A comparison of approaches for maximizing business payoff of prediction models. In: Proceedings of the

2nd ACM SIGKDD international conference on knowledge discovery and data mining, KDD-96. pp. 195–201.

McPherson, M., Smith-Lovin, L., Cook, J., 2001. Birds of a feather: Homophily in social networks. Annual Review of Sociology 27, 415–444.

Mizerski, R., 1982. An attribution explanation of the disproportionate influence of unfavourable information. Journal of Consumer Research 9 (12), 301–310.

Mozer, M., Wolniewicz, R., Grimes, D., Johnson, E., Kaushansky, H., 2000. Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. IEEE Transactions on Neural Networks 11 (3), 690–696.

Murthi, B., Sarkar, S., 2003. The role of the management sciences in research on personalization. Management Science 49 (10), 1344–1362.

Nanavati, A., Singh, R., Chakraborty, D., Dasgupta, K., Mukherjea, S., Das, G., Gurumurthy, S., Joshi, A., 2008. Analyzing the structure and evolution of massive telecom graphs. IEEE Transactions on Knowledge and Data Engineering 20 (5), 703–718.

Nemenyi, P., 1963. Distribution-free multiple comparisons. Ph.D. thesis, Princeton University.

Neslin, S., Gupta, S., Kamakura, W., Lu, J., Mason, C., 2006. Detection defection: Measuring and understanding the predictive accuracy of customer churn models. Journal of Marketing Research 43 (2), 204–211.

Neville, J., Jensen, D., 2007. Relational dependency networks. Journal of Machine Learning Research 8, 653–692.

Newman, M., 2010. Networks: An Introduction. Oxford University Press, Oxford, U.K.

Padmanabhan, B., Tuzhilin, A., 2003. On the use of optimization for data mining: Theoretical interactions and ecrm opportunities. Management Science 49 (10), 1327–1343.

Parpinelli, R., Lopes, H., Freitas, A., 2001. An ant colony based system for data mining: Applications to medical data. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001. Morgan Kaufmann Publishers Inc., San Francisco, CA, U.S.A., pp. 791–797.

Paulin, M., Perrien, J., Ferguson, R., Salazar, A., Seruya, L., 1998. Relational norms and client retention: External effectiveness of commercial banking in Canada and Mexico. International Journal of Bank Marketing 16 (1), 24–31.

Perlich, C., Provost, F., 2003. Aggregation-based feature invention and relational concept classes. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD-03. pp. 167–176.

Perlich, C., Provost, F., 2006. Distribution-based aggregation for relational learning with identifier attributes. Machine Learning 62 (1/2), 65–105.

Piatetsky-Shapiro, G., Masand, B., 1999. Estimating campaign benefits and modeling lift. In: Proceedings of the 5th ACM SIGKDD international conference on knowledge discovery and data mining, KDD-99. pp. 185–193.

Piramuthu, S., 2004. Evaluating feature selection methods for learning in data mining applications. European Journal of Operational Research 156 (2), 483–494.

Porta Garcia, M., Montiel, O., Castillo, O., Sepúlveda, R., Melin, P., 2009. Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. Applied Soft Computing 9 (3), 1102–1110.

Potharst, R., Feelders, A., 2002. Classification trees for problems with monotonicity constraints. SIGKDD Explorations 4 (1), 1–10.

Provost, F., Fawcett, T., Kohavi, R., 1998. The case against accuracy estimation for comparing induction algorithms. In: Proceedings of the 15th International Conference on Machine Learning, ICML. Morgan Kaufmann Publishers Inc., San Francisco, CA, U.S.A., pp. 445–453.

Quinlan, J., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, U.S.A.

Rasmusson, E., 1999. Complaints can build relationships. Sales and Marketing Management 151 (9), 89–90.

Reichheld, F., 1996. Learning from customer defections. Harvard Business Review 74 (2), 56–69.

Richter, Y., Yom-Tov, E., Slonim, N., 2010. Predicting customer churn in mobile networks through the analysis of social groups. In: Proceedings of the 10th SIAM International Conference on Data Mining. pp. 732–741.

Rocchio, J., 1971. Relevance feedback in information retrieval. In: Salton, G. (Ed.), The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice Hall, pp. 313–323.

Rust, R., Zahorik, A., 1993. Customer satisfaction, customer retention, and market share. Journal of Retailing 69 (2), 193–215.

Setiono, R., Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2010a. Software effort prediction using regression rule extraction from neural networks. In: Proceedings of the 22nd International Conference on Tools with Artificial Intelligence, ICTAI 2010, Arras, France. pp. 45–52.

Setiono, R., Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2010b. Software effort prediction using regression rule extraction from neural networks. In: Proceedings of the OR52 Annual Conference, London, U.K.

Silberschatz, A., Tuzhilin, A., 1995. On subjective measures of interestingness in knowledge discovery. In: Proceedings of the 1st ACM SIGKDD conference on Knowledge Discovery and data mining, KDD-95. pp. 275–281.

Sill, J., 1998. Monotonic networks. In: Jordan, M., Kearns, M., Solla, S. (Eds.), Advances in Neural Information Processing Systems. Vol. 10. MIT Press, Cambridge, MA, U.S.A.

Sterling, A. B.-D. L., Pao, Y., 1989. Learning and classification of monotonic ordinal concepts. Computational Intelligence 5 (1), 45–49.

Stum, D., Thiry, A., 1991. Building customer loyalty. Training and Development Journal 45 (4), 34–36.

Stützle, T., Hoos, H., 2000. $\mathcal{MAX}$-$\mathcal{MIN}$ ant system. Future Generation Computer Systems 16 (8), 889–914.

Suykens, J., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J., 2002. Least Squares Support Vector Machines. World Scientific, Singapore.

Suykens, J., Vandewalle, J., 1999. Least squares support vector machine classifiers. Neural Processing Letters 9 (3), 293–300.

Swets, J., Pickett, R., 1982. Evaluation of Diagnostic Systems: Methods from Signal Detection Theory. Academic Press, New York, NY, U.S.A.

Tan, P., Steinbach, M., Kumar, V., 2006. Introduction to Data Mining. Addison Wesley, Boston, MA, U.S.A.

Thomas, L., Edelman, D., Crook, J. (Eds.), 2002. Credit Scoring and its Applications. SIAM.

Thomassey, S., Happiette, M., 2007. A neural clustering and classification system for sales forecasting of new apparel items. Applied Soft Computing 7 (4), 1177–1187.

Van Gestel, T., Martens, D., Baesens, B., Feremans, D., Huysmans, J., Vanthienen, J., 2007. Forecasting and analyzing insurance companies' ratings. International Journal of Forecasting 23 (3), 513–529.

Van Gool, J., Baesens, B., Sercu, P., Verbeke, W., 2009. An analysis of the applicability of credit scoring for microfinance. In: Proceedings of the Academic and Business Research Institute Conference, Orlando, FL, U.S.A.

Van Gool, J., Verbeke, W., Sercu, P., Baesens, B., 2010. Credit scoring for microfinance: Is it worth it? International Journal of Finance and Economics, published online, forthcoming.

Vandecruys, O., Martens, D., Baesens, B., Mues, C., De Backer, M., Haesen, R., 2008. Mining software repositories for comprehensible software fault prediction models. Journal of Systems and Software 81 (5), 823–839.

Vanthienen, J., Mues, C., Aerts, A., 1998a. An illustration of verification and validation in the modelling phase of KBS development. Data and Knowledge Engineering 27 (3), 337–352.

Vanthienen, J., Mues, C., Wets, G., Delaere, K., 1998b. A tool-supported approach to inter-tabular verification. Expert Systems with Applications 15 (3-4), 277–285.

Vapnik, V., 1995. The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, U.S.A.

Verbeke, W., 2007. Verkeersmanagement op basis van controle van subsystemen. In: Proceedings of the Colloquium Vervoersplanologisch Onderzoek, Antwerp, Belgium.

Verbeke, W., Baesens, B., 2009. Van credit crunch naar ict crash, of niet? Data News.

Verbeke, W., Baesens, B., Martens, D., De Backer, M., Haesen, R., 2009a. Including domain knowledge in customer churn prediction using antminer+. In: Perner, P. (Ed.), Workshop Proceedings DMM 2009. Advances in Data Mining in Marketing. IbaI Publishing, pp. 10–21.

Verbeke, W., Berteloot, K., Castermans, G., Martens, D., Van Gestel, T., Baesens, B., 2010a. Modeling credit rating migrations dependent on the business cycle. In: Proceedings of the EURO 2010 conference, Lisbon, Portugal.

Verbeke, W., Dejaeger, K., Baesens, B., 2010b. Comparing classification techniques to forecast customer churn. In: Proceedings of the OR52 Annual Conference, London, U.K.

Verbeke, W., Dejaeger, K., Martens, D., Baesens, B., 2010c. Customer churn prediction: does technique matter? In: Proceedings of the Joint Statistical Meeting, JSM2010, Vancouver, Canada.

Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B., 2011a. New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. European Journal of Operational Research, doi 10.1016/j.ejor.2011.09.031.

Verbeke, W., Dejaeger, K., Verbraken, T., Martens, D., Baesens, B., 2011b. Mining social networks for customer churn prediction. In: Proceedings of the Interdisciplinary Workshop on Information and Decisions in Social Networks, WIDSLIDS, Cambridge, MA, U.S.A.

Verbeke, W., Martens, D., Baesens, B., 2009b. Building comprehensible customer churn prediction models with advanced rule induction techniques. In: Dag van het Vlaams Wetenschappelijk Economisch Onderzoek, Hasselt, Belgium.

Verbeke, W., Martens, D., Baesens, B., 2011c. Rulem: Rule learning with monotonicity constraints for ordinal classification. IEEE Transactions on data and knowledge engineering, under review.

Verbeke, W., Martens, D., Baesens, B., 2011d. Social network analysis for customer churn prediction. Management Science, under review.

Verbeke, W., Martens, D., Mues, C., Baesens, B., 2011e. Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Systems with Applications 38 (3), 2354–2364.

Verbeke, W., Tampère, C., Van Paesschen, N., Immers, B., 2008. Modeling traffic operations on intersections using monte-carlo simulation techniques. In: Proceedings of the 87th Transportation Research Board Annual Meeting, Washington D.C., U.S.A.

Verbeke, W., Verbraken, T., Martens, D., Baesens, B., 2011f. Relational learning for customer churn prediction: the complementarity of networked and non-networked classifiers. In: Proceedings of the Conference on the Analysis of Mobile Phone Datasets and Networks, NETMOB, Cambridge, MA, U.S.A.

Verbraken, T., Goethals, F., Verbeke, W., Baesens, B., 2011a. Using social network classifiers for predicting ecommerce. In: The Tenth Workshop on E-Business, WEB2011, Shanghai, China.

Verbraken, T., Verbeke, W., Baesens, B., 2011b. Novel profit maximizing metrics for measuring classification performance of customer churn prediction models. IEEE Transactions on data and knowledge engineering, under review.

Verbraken, T., Verbeke, W., Baesens, B., 2011c. Profit optimizing customer churn prediction with bayesian network classifiers. Intelligent Data Analysis, forthcoming.

Viti, F., Verbeke, W., Tampère, C., 2008a. Sensor locations for reliable travel time prediction and dynamic management of traffic networks. Transportation Research Record 2049, 103–110.

Viti, F., Verbeke, W., Tampère, C., 2008b. Sensor locations for reliable travel time prediction and dynamic management of traffic networks. In: Proceedings of the 87th Transportation Research Board Annual Meeting, Washington D.C., U.S.A.

Wade, A., Salhi, S., 2004. An ant system algorithm for the mixed vehicle routing problem with backhauls. In: Metaheuristics: computer decision-making. Kluwer Academic Publishers, Norwell, MA, U.S.A., pp. 699–719.

Wasserman, S., Faust, K., 1997. Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge, MA, U.S.A.

Wei, C., Chiu, I., 2002. Turning telecommunications call details to churn prediction: a data mining approach. Expert Systems with Applications 23 (2), 103–112.

Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biometrics 1, 80–83.

Witten, I., Frank, E., 2000. Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann Publishers Inc., San Francisco, CA, U.S.A.

Xevelonakis, E., 2004. Developing retention strategies based on customer profitability in telecommunications: An empirical study. Database marketing and customer strategy management 12 (3), 226–242.

Zeithaml, V., Berry, L., Parasuraman, A., 1996. The behavioural consequences of service quality. Journal of Marketing 60 (2), 31–46.

Publication list

Articles in internationally reviewed scientific journals

- Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B., 2011a. New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. European Journal of Operational Research, doi 10.1016/j.ejor.2011.09.031

- Verbeke, W., Martens, D., Mues, C., Baesens, B., 2011e. Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Systems with Applications 38 (3), 2354–2364

- Verbraken, T., Verbeke, W., Baesens, B., 2011c. Profit optimizing customer churn prediction with bayesian network classifiers. Intelligent Data Analysis, forthcoming

- Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2011a. Benchmarking classification algorithms for software effort prediction. IEEE Transactions on Software Engineering, published online, forthcoming

- Van Gool, J., Verbeke, W., Sercu, P., Baesens, B., 2010. Credit scoring for microfinance: Is it worth it? International Journal of Finance and Economics, published online, forthcoming

- Martens, D., Vanthienen, J., Verbeke, W., Baesens, B., 2011. Performance of classification models from a user perspective. Decision Support Systems 51, 782–793

- Viti, F., Verbeke, W., Tampère, C., 2008a. Sensor locations for reliable travel time prediction and dynamic management of traffic networks. Transportation Research Record 2049, 103–110

Papers at international conferences and symposia, published in full in proceedings

- Setiono, R., Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2010a. Software effort prediction using regression rule extraction from neural networks. In: Proceedings of the 22nd International Conference on Tools with Artificial Intelligence, ICTAI 2010, Arras, France. pp. 45–52

- Verbeke, W., Baesens, B., Martens, D., De Backer, M., Haesen, R., 2009a. Including domain knowledge in customer churn prediction using antminer+. In: Perner, P. (Ed.), Workshop Proceedings DMM 2009. Advances in Data Mining in Marketing. IbaI Publishing, pp. 10–21

- Verbeke, W., Tampère, C., Van Paesschen, N., Immers, B., 2008. Modeling traffic operations on intersections using monte-carlo simulation techniques. In: Proceedings of the 87th Transportation Research Board Annual Meeting, Washington D.C., U.S.A

- Viti, F., Verbeke, W., Tampère, C., 2008b. Sensor locations for reliable travel time prediction and dynamic management of traffic networks. In: Proceedings of the 87th Transportation Research Board Annual Meeting, Washington D.C., U.S.A

Meeting abstracts, presented at international conferences and symposia, published or not published in proceedings or journals

- Verbraken, T., Goethals, F., Verbeke, W., Baesens, B., 2011a. Using social network classifiers for predicting ecommerce. In: The Tenth Workshop on E-Business, WEB2011, Shanghai, China

- Verbeke, W., Verbraken, T., Martens, D., Baesens, B., 2011f. Relational learning for customer churn prediction: the complementarity of networked and non-networked classifiers. In: Proceedings of the Conference on the Analysis of Mobile Phone Datasets and Networks, NETMOB, Cambridge, MA, U.S.A

- Verbeke, W., Dejaeger, K., Verbraken, T., Martens, D., Baesens, B., 2011b. Mining social networks for customer churn prediction. In: Proceedings of the Interdisciplinary Workshop on Information and Decisions in Social Networks, WIDSLIDS, Cambridge, MA, U.S.A

- Verbeke, W., Dejaeger, K., Baesens, B., 2010b. Comparing classification techniques to forecast customer churn. In: Proceedings of the OR52 Annual Conference, London, U.K

- Verbeke, W., Dejaeger, K., Martens, D., Baesens, B., 2010c. Customer churn prediction: does technique matter? In: Proceedings of the Joint Statistical Meeting, JSM2010, Vancouver, Canada

- Verbeke, W., Berteloot, K., Castermans, G., Martens, D., Van Gestel, T., Baesens, B., 2010a. Modeling credit rating migrations dependent on the business cycle. In: Proceedings of the EURO 2010 conference, Lisbon, Portugal

- Setiono, R., Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2010b. Software effort prediction using regression rule extraction from neural networks. In: Proceedings of the OR52 Annual Conference, London, U.K

- Dejaeger, K., Verbeke, W., Huysmans, J., Mues, C., Vanthienen, J., Baesens, B., 2010a. Rule based predictive models, decision table and tree: an empirical evaluation on comprehensibility. In: Proceedings of the EURO 2010 conference, Lisbon, Portugal

- Van Gool, J., Baesens, B., Sercu, P., Verbeke, W., 2009. An analysis of the applicability of credit scoring for microfinance. In: Proceedings of the Academic and Business Research Institute Conference, Orlando, FL, U.S.A

- Verbeke, W., Baesens, B., Martens, D., De Backer, M., Haesen, R., 2009a. Including domain knowledge in customer churn prediction using antminer+. In: Perner, P. (Ed.), Workshop Proceedings DMM 2009. Advances in Data Mining in Marketing. IbaI Publishing, pp. 10–21

Meeting abstracts, presented at local conferences and symposia, published or not published in proceedings or journals

- Verbeke, W., Martens, D., Baesens, B., 2009b. Building comprehensible customer churn prediction models with advanced rule induction techniques. In: Dag van het Vlaams Wetenschappelijk Economisch Onderzoek, Hasselt, Belgium

- Verbeke, W., 2007. Verkeersmanagement op basis van controle van subsystemen. In: Proceedings of the Colloquium Vervoersplanologisch Onderzoek, Antwerp, Belgium, **Winner of the student master thesis prize**

Other journal publications / miscellaneous

- Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2011b. Het voorspellen van software-ontwikkelkosten. Informatie, forthcoming

- Dejaeger, K., Verbeke, W., Martens, D., Baesens, B., 2010b. De kosten van software-ontwikkeling voorspellen. Informatie 52 (9), 8–13

- Verbeke, W., Baesens, B., 2009. Van credit crunch naar ict crash, of niet? Data News

Articles submitted for publication in internationally reviewed scientific journals

- Verbeke, W., Martens, D., Baesens, B., 2011d. Social network analysis for customer churn prediction. Management Science, under review

- Verbeke, W., Martens, D., Baesens, B., 2011c. Rulem: Rule learning with monotonicity constraints for ordinal classification. IEEE Transactions on data and knowledge engineering, under review

- Verbraken, T., Verbeke, W., Baesens, B., 2011b. Novel profit maximizing metrics for measuring classification performance of customer churn prediction models. IEEE Transactions on data and knowledge engineering, under review

- Berteloot, K., Verbeke, W., Castermans, G., Van Gestel, T., Martens, D., Baesens, B., 2011. Credit rating migration modeling using macroeconomic indices. International Journal of Forecasting, under review

Doctoral dissertations from the faculty of business and economics

A full list of the doctoral dissertations from the Faculty of Business and Economics can be found at:

www.kuleuven.ac.be/doctoraatsverdediging/archief.htm.