
# On Utilising Change over Time in Data Mining

## DISSERTATION

zur Erlangung des akademischen Grades

**Doktoringenieur (Dr.-Ing.)**

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von Diplom-Informatiker Mirko Böttcher

geboren am 04. Mai 1975 in Magdeburg

Gutachter:

Prof. Dr. Rudolf Kruse
Prof. Dr. Detlef Nauck
Prof. Trevor Martin

Magdeburg, den 13. November 2013

# Abstract

In many businesses data is continuously gathered over long periods of time and therefore reflects changes in the domain from which it has been derived. The last decade has seen a remarkable shift in the perception of the value of these changes. The initial focus of data mining research on maintaining the quality of models and patterns in spite of change is now superseded by an interest in gaining knowledge about change. This thesis closes the gap between the two research streams. It advocates the use of knowledge about change in order to improve the quality of models and patterns. Two different methods are detailed which prove that such a direction is viable. The first method targets the shortcoming of frequent item set discovery, namely the generation of vast numbers of item sets. It proposes a novel condensed representation of item sets which considers change in the data, yet is consistent with condensed representations that ignore change. The second method targets the shortcoming of decision tree induction, namely the degradation in the classification accuracy of a once learned decision tree as time proceeds. It proposes to anticipate decision trees by utilising models of the change in the measures that control its induction process. Through theory and experiments it is proven that both methods truly enhance their change-unaware counterparts and provide a blueprint for utilising change in data mining.

# Kurzfassung

Viele Unternehmen sammeln große Mengen an Daten. Diese werden zumeist kontinuierlich über lange Zeiträume erfasst und reflektieren zeitliche Veränderungen der Domäne, aus der sie stammen. Im Laufe des letzten Jahrzehnts konnte innerhalb der Data Mining Forschung ein bemerkenswerter Wandel in der Wahrnehmung des Wertes dieser Veränderungen beobachtet werden. Das ursprüngliche Ziel, die Qualität von Modellen und Mustern trotz Veränderungen zu erhalten, wurde verdrängt durch das Ziel der Gewinnung von Wissen über Veränderungen. Diese Dissertationsschrift schließt eine Lücke zwischen den diesen Zielen zugeordneten Forschungsrichtungen. Sie verfolgt den Ansatz, Wissen über Veränderungen zu nutzen, um die Qualität von Modellen und Mustern zu verbessern. Zwei Methoden werden ausführlich erläutert, um zu zeigen, dass dieser Ansatz machbar ist. Die erste Methode zielt auf die Schwäche der Entdeckung von häufigen Itemmengen (frequent item sets), eine riesige Anzahl an Mustern zu generieren. Sie nutzt eine neuartige, kondensierte Repräsentation (condensed representation) von Itemmengen, die Veränderungen in den Daten berücksichtigt. Zugleich ist sie konsistent zu existierenden Ansätzen, die Veränderungen ignorieren. Die zweite Methode zielt auf die Schwäche der Induktion von Entscheidungsbäumen, dass einmal gelernte Entscheidungsbäume im Laufe der Zeit in ihrer Klassifikationsgenauigkeit abnehmen. Die Methode antizipiert Entscheidungsbäume indem sie Veränderungsmodelle der Maße nutzt, die den Induktionsprozess der Bäume kontrollieren. Theoretisch und experimentell wird gezeigt, dass beide Methoden wirkliche Verbesserungen erzielen und gleichzeitig allgemeine Ansatzpunkte bieten, wie zeitliche Veränderungen in Daten im Data Mining genutzt werden können.

# Acknowledgments

Writing a PhD thesis is a solitary endeavour, and yet without the support, inspiration and advise of a number of people, I could not have written it.

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Dr. Rudolf Kruse for his continuous support and his unlimited patience and trust in me finishing this work. In many respects, he paved the way for my thesis. Not only did he raise my interest in data mining and soft computing when I was an undergraduate student, but he also provided the initial contact to British Telecom, and agreed to supervise my thesis when I asked him two years after I finished my studies.

I am deeply grateful to Detlef Nauck for giving me the freedom to start and pursue my own research in the course of the IDEAL project while I was working for British Telecom's Intelligent Systems Research Centre. Many of the ideas promoted in this thesis originate from this research. I would also like to thank him for being one of the first who (maybe without being aware of it) pointed me to the idea of doing a PhD and for always being interested in my research and its progress.

Many thanks also go to my third reviewer, Prof. Trevor Martin, for travelling to Magdeburg to personally attend my PhD defence which thus spared myself a video conference.

I owe a great deal to Martin Spott whose advise and critical passion substantially helped me to shape and challenge my ideas. If it were not for his thorough proof-reading and his insightful comments, parts of this thesis would be quite difficult to understand. I would also like to thank him for keeping in touch over so many years and for his continuous feedback to my work.

Many other people have followed this thesis and helped me to elaborate my ideas. In particular, I would like to give thanks and a special acknowledgement to Alex Healing for having read parts of this thesis and for his precious comments and language advise. I would also like to thank Myra Spiliopoulou and Frank Höppner. Our discussions and controversies during the preparation of our SIGKDD Explorations article and our subsequent tutorial at the ECML PKDD were a great source of inspiration to me. I am deeply thankful to Daniela Müller for providing me access to many research articles for which I would otherwise only have a reference. I would also like to thank her for showing a lot of patience and understanding when I sacrificed some of our quality time to work on this thesis.

Last, but surely not least, I am deeply thankful to Katja, my girlfriend, for always encouraging me to go on finishing this thesis and for enduring even the most stressful periods of work with admirable patience.

“Sie sagen: Gib mir das Aktuellste! Das Problem mit dem Aktuellsten ist, dass es oft unwahr ist. Und es hat keine Bedeutung. Die Frage ist nicht: Wie verläuft die schrittweise Entwicklung? Die Frage ist: Was ist die Hintergrundgeschichte? Was ist versteckt? Was wissen wir nicht? Wir brauchen eben manchmal Wochen und Monate, um herauszufinden, was wirklich passiert.”

Bob Woodward (Süddeutsche Zeitung, 28./29. November 2009)

# Contents

# Chapter 1

# Introduction

> *"...humans are built to detect real-world structure by detecting changes along physical dimensions [...] and representing these changes as relations [...] along subjective dimensions. Because change can only occur over time, it makes sense that time somehow be incorporated into a definition of structure."*
>
> Mari Riess Jones[1]

In the early 1980s Jennifer F. Freyd, at this time a PhD student in psychology at Stanford University, conducted an experiment in which people viewed frozen-action photographs, like one of a person jumping from a wall, and their memory for these scenes was tested. To her surprise she found that very often the people identified photographs as being the same when they were actually showing the scene at a later point in time. It appeared as if the people subconsciously animated the photographs to continue the action in mind. They enriched a static stimulus by information about *change*. Eventually, the experiments by J. F. Freyd led to the hypothesis that the representation of *change* plays an important role in human cognition independent of whether the perceptions are dynamic or static (Freyd, 1983).

In everyday life many examples can be found how we successfully recognise, analyse and utilise change which helps us predict what will happen and, consequently, how we should make *decisions*. When investing in stocks we do not simply look at the current price but also how it developed other the previous months. When we inspect an unknown object we often gain a better understanding by changing the perspective from which we view it, contrasting with previous views. When we drive a car we see many objects in our field of vision but we focus only on those which change, for example visually, like warning signs, or spatially, like pedestrians and cars.

In business life the situation is not much different: decisions have to be made and, as a prerequisite, the processes and the entities involved need to be understood. In contrast to everyday life though, the desired knowledge is often not readily available but instead hidden within the masses of data generated throughout the years during daily operations. As an answer to this problem the field of *data mining* has emerged which is concerned with discovering structure in data, or as Fayyad et al. (1996b) wrote "is the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data".

Because change is such a crucial source of information for understanding and decision making— cognitively, and in everyday and business life—it is instructive to ask how this is acknowledged by research in data mining. First of all, the data mining community is aware that domains and thus the data collected from it change over time. Considerable research has been conducted into maintaining models and patterns over time, into the

[1] Jones, M. R. (1976). Time, our lost dimension: toward a new theory of perception, attention, and memory. *Psychological Review*, 83(5):323–355.

detection of novel or emerging patterns and into the description and analysis of change. This led to the data mining fields *concept drift detection*, *contrast mining* and *change mining*. On the other hand, the vast majority of data mining approaches either ignore change altogether or regard it as a perturbing factor.

Both *perspectives on change* are extreme cases when compared to how humans process change. Whilst change is sometimes intentionally focused on and sometimes completely ignored, it is much more often unknowingly used as a further source of information helping us to focus our attention, to sharpen our perception and to substantiate our anticipation of the future. Transferred to data mining this leads to the question whether data mining approaches can be improved if change as a further source of information is incorporated and how this incorporation can be accomplished. Surprisingly these questions have not received much attention from the data mining community yet.

The question this thesis aims to answer is, if we actively take *change as a process* into account, if we analyse and model it, what advantages does this have on the results of data mining? Because change can only occur over time this means that the *time axis* should somehow be incorporated into data mining approaches. In particular, this thesis looks at data mining approaches from association analysis and classification, demonstrating for each how time and thus change can be incorporated into the mining process, which current shortcomings can thereby be improved and draws general principles for utilising change.

## 1.1 Fundamentals about Change

The ancient Greek philosopher Heraclitus once said: "The only thing that is constant is change". In fact, it is much easier to think about a domain which changes than to think about one which does not change at all. Taking the retail business as an example of a changing domain, customers are influenced in their perception of products and services by marketing campaigns, through the advice of family or friends, by reviews read in the internet, by the launch of competing or alternative products and their technological level, reliability and perceived trendiness. As a response to the evolving attitude of customers the retail business has to change too: business plans have to be revised, product development work refocused and marketing campaigns launched. These actions, in turn, have effect on the customers – a vicious cycle.

Customers, markets and products are typical entities analysed in data mining. Driven by the need of businesses to influence customer satisfaction, and to respond to market requirements and sales potentials, the value of the discovered knowledge is commonly assessed by its actionability (Silberschatz and Tuzhilin, 1996) and usefulness (Fayyad et al., 1996b). Thereby, the action is mostly targeted towards the analysed entities themselves in the expectation that they will change in a way that will benefit the businesses' objectives. For example, the insight that the sales figures of a product are too low in an important group of customers may lead to the release of a new, specifically tailored commercial. This commercial may indeed be successful and increase the sales figures in this segment, but it may also have a negative impact on the perception of the business in other customer groups. In either case, the results of data mining change the next time the domain is being analysed. One of the factors which triggered this change may have been the data mining itself.

At this point it should be clear that change is ubiquitous in many domains. If a domain does not change by itself, it certainly does by the application of knowledge as the result of data mining. This truly is a paradoxical situation. On the one hand, data mining may trigger change. On the other hand, data mining approaches very often assume that the domain under consideration is stable over time.

### 1.1.1 The Process of Change

So far, we have used the word 'change' based on its intuitive meaning. However, there are two possible notions of change which are employed in real life and consequently also in data mining research. Both have their own relevance and both are accounted for differently: *change* may denote *the process of change*, such as the evolution of a customer segment and its responsiveness to a marketing strategy that must be aligned again and again to keep it profitable. Or it may denote *the outcome of a process of change*, such as a sales collapse for a particular product in a certain customer segment. Clearly, the notion of change as a process provides the basis for the notion of change as an outcome.

These notions of change are associated with two intuitive questions: "How is the world changing?" and "When did/will a significant change occur?" From a data mining point of view the answer to the first question refers to the evolution of data and describes the nature of change either retrospectively or prospectively. The second question on the point in time of change can be the first step in linking significant changes with events that may have affected the domain under consideration. The first question asks for a description which can be provided having only little or perhaps no prior domain knowledge. In contrast, the second question is a typical search question and a substantially larger amount of prior domain knowledge is necessary to answer it. For example, it has to be known what kind of change can be expected and what renders a change significant. The difficulty is to determine significance. In fact, if no background knowledge is available it is difficult to verify whether or not a change at a certain point in time also is significant.

### 1.1.2 The Time Axis

Generally, change may be observed along different dimensions, but is often linked with the temporal or spatial dimension, or combinations of both. This thesis only deals with changes along the temporal dimension. Because such changes occur over time, it is clear that any kind of analysis which exploits change must somehow incorporate the time axis. Fortunately, data sets which contain time as an attribute are common. In finance, retail and medical care businesses, to name a few, it is often imperative to enable accountability or traceability of data creation and changes. For example, banks are legally obliged to maintain a complete trace of any customer transaction for a very long period of time. This obligation also covers details such as keeping the date and time and origin of the transaction. In retail applications often the complete history of customer interactions is archived: details about purchased goods and service usage are needed for billing and auditing purposes, and inquiries and complaints are kept for future reference and to improve customer satisfaction. In medical applications documenting symptoms and diagnoses over time helps in later examinations but may also guard against wrongful malpractice lawsuits.

These needs for storing time information are mirrored in the invention and meanwhile widespread use of data warehouse systems which were designed to capture the evolving histories of organisations. As Kimball (1996) pointed out: "The time dimension is the one dimension virtually guaranteed to be present in every data warehouse, because virtually every data warehouse is a time series." Even if organisations do not employ data warehouse technology the time attribute can be expected to be present in almost every database schema, not only because it is required but also because it is often the simplest one to gather. Even if no time attribute is present, modern database technology allows one to obtain the commit time of any row in a certain table. Assuming that most data is written to a database close to when it is collected this allows at least the temporal sequence of records to be reconstructed.

Although the time axis is omnipresent in real-world data sets, it is surprising that common benchmark data sets with a time attribute are virtually non-existant. At the time of this thesis being written the well-known UCI machine learning repository[2] contains 33 data sets with categorical attributes and of significant size (i.e. more than 1000 records), none of which contain a time attribute. Moreover, many data sets, such as survey or web access data, would have potentially been able to incorporate data from multiple years, months, or days, but in fact they only contain those from a single period. For this reason, many researchers employ artificially generated data such as a simple block-world data set which according to Tsymbal (2004) is the most popular one and was first used by Schlimmer and Granger (1986).

## 1.2 Perspectives on Change

In light of change being present everywhere and the fact it embodies cues for shaping the future and judging the present, it may come as a surprise that it is also perceived as a disturbance. These two perspectives on change not only relate to the area of data mining research, but have also existed also in other areas for some time, as the following historic example illustrates.

### 1.2.1 Vincenzo Viviani and Léon Foucault

Vincenzo Viviani (1622-1703) was an Italian nobleman, physician and mathematician. In 1639 Galileo Galilei impressed by Viviani's exceptional intelligence took him into his home as student, collaborator and companion to work with him on physics and geometry. After Galilei died in January 1642 Viviani stayed in Florence and continued his scientific work first as a collaborator of Evangelista Toricelli, Galilei's successor as the Medici's Court Mathematician, and later as a lecturer at the Accademia del Designo in Florence and Mathematician to the Medici Court.

His reports and notes are kept in a museum in Florence and they indicate that he must have undertaken substantial experimentation involving pendulums, using them, for example, as a timing device—an approach which had been invented by his mentor Galilei. One of these reports is quite remarkable from today's perspective. It consists of three notes which have been republished by Hagen and Rom (1930) and of which I will only quote the first two. In the first note Viviani states: "We observed that a unifilar

[2] http://archive.ics.uci.edu/ml (Retrieved on May, 9th, 2013)

attached pendulum deviates from a vertical plane always in the same direction[...]". The second note provides a hint at Viviani's reaction: "Because the ordinary unifilar pendulum due to its freedom of movement (whatever its reason may be) slowly deviates from its initial run [...] it was thought about suspending the bob on two strings to keep the oscillations on the same trajectory."

Léon Foucault (1819-1868) was a French experimental physician who first studied medicine which he abandoned to turn himself towards physics. In fact, he never studied physics but acquired his knowledge by self-teaching himself while working as a scientific writer to make a living. Given this background it may be not surprising that he carried out most of his experiments in the cellar of the house in which he lived with his mother reaching achievements like proving that light travels more slowly through water than through air, a counter-proof to Newton's corpuscle theory of light.

While Foucault's name is also linked to the construction of the first gyroscope and many other inventions, it is one experiment which laid the foundation of his fame and it also started in the cellar of his home. On January 6, 1851 he cleverly suspended a 2-meter long pendulum at the ceiling such that it could rotate freely. Making it swing he observed a sideward drift of the oscillation plane – the same drift Viviani had observed about 200 years earlier. In contrast to Viviani, however, he analysed this change and interpreted its roots thus showing that the oscillation plane remains fixed with respect to the stars while the earth rotates underneath it. For the first time it was proved that the earth rotates using a clever, but simple experiment.

Viviani and Foucault made the same observation but looked at it from two different perspectives. Viviani regarded the changing oscillation plane as a perturbation and attributed it to an imperfection of the experiment having the potential to falsify its desired outcome. There was no apparent reason for him to investigate further, except maybe curiosity. Rather than trying to explain or to take into account what he observed he provided a countermeasure by suspending the pendulum on two strings. Foucault, in contrast, regarded the change as an important source of information. He was primarily interested in observing and analysing the changing oscillation plane rather than the oscillation as such. In fact, the experiment he conducted was tailored to this specific purpose[3].

As the story of Viviani's and Foucault's observation illustrates it can be difficult to decide between either regarding change as perturbation or as information. Often it is deliberately decided whether to ignore it or to incorporate it depending on the current objectives and expectations. On the one hand, it is convenient to regard change as a perturbation if the analysis of change is not obviously related to the task at hand, or if the effort to analyse the change is disproportionate to the anticipated knowledge gain. On the other hand, it seems to be easier to regard change as a source of information if an initial hypothesis about the underlying causes of change exists and needs to be verified, or if it can be assumed that analysing change leads to actionable insight into the domain.

[3]What 'put him on path' to the pendulum experiment was twanging a round steel rod in the chuck of a lathe. He rotated the chuck and although the rod turned with it the plane of vibration of the rod stayed the same (Tobin, 2003). This effect must have surprised him as much as the drifting oscillation plane must have surprised Viviani.

### 1.2.2 The Two Perspectives in Data Mining

Both perspectives on change can also be found in data mining. On the one hand most 'classical' data mining tasks like classification, clustering or association analysis are built on the assumption that the domain from which data was derived is stable over time. If this assumption is violated because the domain changes nonetheless, change is regarded as a perturbing factor. On the other hand, many approaches have been published which put change at the centre of the analysis by dealing with its detection and description.

#### Ignoring Change

Relatively early after the term data mining had been coined it became clear that the different objectives for a person who is analysing data can be categorised based on the corresponding type of task to be carried out. Many of these *data mining tasks* can be called 'classical' because they already played an important role in fields like statistics and machine learning well before the term 'data mining' emerged but still expose many research challenges today. Among these 'classical' tasks are in particular those named in the list below. The list does not claim to be complete because the tasks vary among different authors depending on the year of publication, the chosen level of granularity and the author's background (see Fayyad et al., 1996a; Hand et al., 2001; Borgelt and Kruse, 2002). For each task an article is cited which reviews the work done in this respective area.

- **Classification and Regression**: The aim in this task is to predict the value of one attribute—the class—based on the known values of other attributes. In case of classification the attribute being predicted is nominal, in case of regression it is quantitative. (Kotsiantis et al., 2006)
- **Segmentation and Clustering**: The aim of this task is to find a description of the data by identifying a finite set of categories or clusters whereby the clusters can be, for example, mutually exclusive, overlapping or hierarchical. (Jain et al., 1999)
- **Association and Pattern Mining**: The aim here is to identify small structures which describe regions of the data but not the complete data set as models do. Often these small structures are described by associations, i.e. co-occurences of attribute values or objects in general. (Hipp et al., 2000; Ceglar and Roddick, 2006)

When much of the seminal work for each task was published, like *ID3* for classification by Quinlan (1986), *k-means* for clustering by McQueen (1967) or *apriori* for association mining by Agrawal et al. (1993), the available computing resources only allowed for processing data sets which were quite limited in size compared with what is possible today. Meaningful and accurate data analysis, however, requires large sample sizes. For this reason, the data used typically only represented a snapshot of a domain taken at a specific time point rather than over a longer time period. Change within a domain, if captured at all, was hence not regarded as an issue and thus ignored. From this lack of practical necessity in combination with methodical convenience arose the assumption that domains are stable over time (see Schlimmer and Granger, 1986).

This historically grown perspective on change is still predominant today even though over the decades, with the emergence of cheap memory, fast processors and sophisticated database technology, it became a reality to collect massive amounts of data over a long period of time. Nevertheless, it was observed that changes within a domain manifest themselves in the data and in the mid 1980s, if not earlier, it was realised that this has a perturbing influence on the results of the task at hand: patterns become invalid over time and models do not correspond to the real-world concept of interest anymore (see Schlimmer and Granger, 1986; Kubat, 1989). In other words, if change is somewhat evident in the data such that it can no longer be ignored it is instead seen as a perturbation.

### Focussing on Change

To draw on the example of Viviani, after being confronted with a perturbation which he could not gainfully relate to his task at hand he employed a countermeasure by suspending the pendulum on two strings. Likewise the machine learning and data mining community searched for countermeasures to the problems induced by changing domains. In contrast to the example of Viviani, however, the causes of change almost always cannot be controlled. Therefore, research efforts focused on the detection of significant changes in order to enforce the relearning and adaption of models and patterns, respectively.

Initially the methods developed for change detection were meant to be embedded into the learning algorithms as a contribution to areas like incremental (online) learning such that they "can detect context changes without being explicitly informed about them" (Widmer and Kubat, 1996). Nevertheless, in parallel to the increasing use of data mining as a business tool consensus was growing that knowing how a domain evolves is equally as important as producing highly accurate models (see, e.g., Liu et al., 2000; Kifer et al., 2004). Because methods for change detection already were available as part of approaches for dealing with concept drift, they provided a starting point to satisfy this upcoming knowledge need, for example by pinpointing the time when a substantial change has happened. Eventually, this led to a second perspective of data mining on change in which change is not ignored anymore but, instead, focused on as the primary analysis object. Nowadays, developing methods for change detection and analysis is seen as one of the primary research issues when dealing with evolving data (Gaber et al., 2005).

In recent years three research directions have emerged which dealing with the detection and description of change. They are listed below. Further references, review articles or, if none have been published yet, seminal works of each direction are provided.

- **Concept Drift Detection**: Under the assumption that a domain is at least temporarily stable, the objective of concept drift detection is to identify a time point such that preceding and succeeding data are sampled from a different distribution. (Schlimmer and Granger, 1986; Widmer and Kubat, 1996)
- **Contrast Mining**: The objective of contrast mining is to quantify and describe the difference between two data sets, which may have been collected subsequently, in terms of the models they induce or the patterns they contain. (Novak et al., 2009; Böttcher, 2011; Dong and Bailey, 2012)

- **Change Mining**: The objective of change mining is the discovery, modelling, monitoring and interpretation of changes in the patterns and models that describe an evolving domain. It encompasses methods for change analysis based on processing models and patterns instead of data. (Böttcher, 2011)

### 1.2.3 A Third Perspective

Both aforementioned perspectives on change are diametrically opposed: one ignores changes altogether while the other focuses on it entirely by discovering knowledge about change. Although change is ubiquitous and although the time dimension is present in many real-world data sets, change is solely utilised *explicitly* if it needs to be understood. This thesis looks at a third perspective where change can also change can also be utilised *implicitly* to the benefit of those 'classical' data mining methods which ignore change, for example to improve on the quality of the knowledge they extract.

Implicitly utilising change means that change within a data set is seen as another 'feature' of the data which carries information, similar to the data set's attribute values. It also means that the focal point of data mining algorithms is not describing and understanding change, like in the second of the mentioned perspectives, but rather on exploiting it as an additional source of information. Because change can only occur over time this implies that the time axis must somehow be incorporated into data mining algorithms whilst their results do not necessarily refer to it.

The fundamental idea to regard change as a feature of the data which is embodied in the time attribute is not new. In a seminal article on learning in changing domains Kubat (1989) wrote:

> What if the patterns are dynamic in the sense that some hidden attributes are present that make the validity of the knowledge base be only temporary? There are two ways to solve this. Either we try to uncover the hidden attributes and include them in the set of attributes, or, if this is not possible, an algorithm is necessary for permanent (periodic) learning and updating the knowledge base.

Kubat (1989) also noted that "in practice, a typical hidden attribute is the 'time' which can induce the changes in other potential attributes". Considering that more than twenty years ago, apart from pure time series, time was only rarely present in data sets it is not surprising that he suggested a solution for the latter case. But what surprises, from today's perspective, is that over the years with the increasing availability of time referenced data the first way he suggested has not spurred much general interest; it would most likely have led to the proposed third perspective of data mining on change. Instead the work by Kubat (1989) sparked considerable research in window-based and incremental learning from which many methods for concept drift detection originate.

## 1.3 Objective

This thesis advocates the above introduced third perspective on change with the objective to prove that it can be realised and is beneficial in practice. This leads to the following research question. Given are a data mining task, a corresponding data mining

method for solving it, and a time-stamped data set. Can this method be extended such that it considers the changes hidden within the data and through this improve on a selected, qualitative shortcoming of the original method's results?

The objective of this thesis is met, if the extended and thus change-aware method satisfies the following two requirements:

**Improvement:** Theory or experimentation show a measurable improvement in a qualitative shortcoming.

**Consistency:** The original method should be a special case of the change-aware method. Their results are consistent, i.e. they have a similar structure and do not contradict each other. Furthermore, existing advantages of the original method and its results are kept and not sacrificed for the desired improvement.

This thesis does not aim to provide a general proof that every data mining method can be extended with and enhanced by change utilisation capabilities. Rather the objective is to show by the example of two chosen data mining methods that such an extension is fruitful

## 1.4 Outline

The remaining part of this thesis is structured in four chapters. The following *Chapter 2* provides a comprehensive survey of methods for change analysis ranging from the first contributions to this subject proposed in the mid 1980s to the latest research results addressing many of the current research challenges. The chapter focusses on the second of the aforementioned data mining's perspectives on change. It puts particular emphasis on change analysis for classification and association mining, the two areas of data mining that will be dealt with in the following two chapters.

The subsequent two chapters move forward from traditional change analysis by applying some of its fundamental ideas to long-standing weaknesses of 'classical' data mining method and put forward algorithmic enhancements. The enhancements utilise information about change and thus are examples for the proposed novel perspective on change in data mining.

Based on this view *Chapter 3* targets frequent item set learners' weakness of producing a huge number of patterns by proposing temporally closed item sets, a novel approach for a condensed representation of item sets which is based on removing temporal redundancies. A theoretical analysis of this representation shows that it is a subset of closed item sets, which are probably the most well-known condensed representation of item sets if change over time is not considered.

*Chapter 4* provides an answer to how the classification accuracy of decision trees learned in changing domains can be improved by presenting an algorithm which anticipates future decision trees based on a model of change. In particular, this algorithm is based on analysing the changes of the decisions made during model learning.

The concluding *Chapter 5* summarises the results of this thesis and points out possible future work.

Experimental results and proofs can be found in two appendices at the end of this thesis.

Chapter 2

# Analysing Change

Change is ubiquitous. Due to the implied dynamics in the data's underlying structure, data mining is faced with old challenges but also with new opportunities: on the challenge side change is seen as perturbing the assumed stability of a domain. Incremental mining methods are required to keep the discovered knowledge efficiently up-to-date and to overcome the need for periodically starting a completely new data mining session. Significant research has been conducted into this area and a plethora of algorithms have been proposed that discount or even forget older examples, and adjust what has been induced from them. Their discussion is outside the scope of this thesis; the interested reader is referred to, for example, Klinkenberg (2004).

On the opportunity side, the change in the data's underlying structure can lead to interesting, valuable and novel insights into a domain, in particular its dynamics. Mainly over the past decade, many researchers have become interested in this task which opened up an alternative perspective on change that centres around using it *explicitely*, as a primary subject to data mining. The vast majority of approaches for analysing change can be assigned to either the areas *contrast mining* or *change mining* which differ in the aspects of change they target and in the questions they are able to answer.

Although this thesis focuses on *implicitly* utilising change, it shares with the aforementioned perspective the salient commonality to have to respond to the conceptual and practical challenges that arise from dealing with temporal data. Indeed, the task of analysing change is more difficult than it appears. For this reason, this chapter provides an overview of recent work on change analysis with a focus on frequent patterns and decision trees in order to gain a deeper understanding of the problems that have been identified and the fundamental techniques that are employed. Since it is inevitable for this discussion to have an, at least basic, understanding of frequent patterns and decision trees, both somewhat classical areas of data mining will be briefly and informally introduced in the course of this chapter. Large parts of this chapter have first been published in Böttcher (2011) and are reproduced here.

## 2.1 Approaches for Analysing Change

Threats and opportunities often manifest themselves by changes in an organisation's environment. Both of them require the systems exhibit an ability not only to detect but also to understand changes to be able to respond with corrective or exploitative action. To decide whether something needs to be done, what to do, and eventually doing it is one core managerial task. It consists of three phases: first, a predecision stage to identify actual and potential threats and opportunities; second, the decision making itself, and third, a postdecision stage to conduct periodical checks for deviations from the assumptions and knowledge on which the decision was based (Ackoff, 1981).

Even though the pre- and postdecision stage are directed at analysing and understanding changes within a domain, they demand different types of knowledge about it.

Starting with the postdecision stage, to check whether the assumptions and knowledge on which a decision rests are still valid entails to *contrast* the knowledge which was available at the time of decision making with the present one. In more general terms, such an approach compares the present state of a domain with a baseline which corresponds to its state at an earlier, distinguished time. As change is ubiquitous it is almost certain that a considerable number of deviations will be detected, though slight deviations only rarely provide enough reason for a corporation to abandon decisions. Owing to the costs, time and resources necessary to decide, revise and implement plans, corporations would intervene only if deviations are so large that the original decision, seen from the present, would be rendered wrong. By and large, the knowledge demanded at the postdecision stage is characterised by pinpointing what has changed and to which extent.

For the predecision stage, in contrast to the postdecision stage, almost always no baseline as a reference of comparison is available owing to the fact that it looks for unexpected, emerging changes in often only little understood domains. It is true that the approach of contrasting two time points is theoretically employable here too by using some arbitrary time point as the baseline. It is, nonetheless, also true that in practise its results will often lack meaning and utility.

The decision whether to buy stocks illustrates this shortfall. Assume it is July 17, 2009 and we are interested in buying automotive stocks which in view of the just finished automotive sales crisis are likely to be sold at a low price. To decide upon whether to buy we analyse the change of the stock price of a car manufacturer using the contrasting approach thereby arbitrarily choosing a time point one year earlier for the baseline, say July 17, 2008. We find that the stock price despite the sales crisis stayed almost level being 29.46 Euro on the first and 29.26 Euro on the second date. So, is this a good buy?

Although in this example the choice of decisions was predefined (buy, not buy) as opposed to real world scenarios where a multitude of alternatives are the norm, most of them being the result of time-intensive brainstorming and refinement, it demonstrates that the knowledge produced by sole comparison of two time points is only of limited use prior to decision making. What it lacks is a sense of *rate* and *anticipation*, this means, whether the observed change progresses slow or fast-paced, and how it extends into the future.

Anticipations of future trends are often developed by looking at the current point and past trends, and projecting them into the future (Ackoff, 1981). Because one shift up or down does not make a trend, an obvious solution for identifying changes to rest decisions on could thus be to contrast the present with a sequence of historic time points as opposed to merely a single one. This eventually yields a history of changes which can be analysed further. Likewise one could also contrast each time point with its predecessor, or the oldest one with any newer, and so forth. There are many alternatives of which each yields a sequence of changes but the general idea remains the same: to analyse change along a sequence of (many) time points. Having a *history* of how a domain looked like in the past, it can serve as the fundamental for the identification and assessment of patterns of change, the more simpler ones of which are trends.

Reconsider the example about buying automotive stocks. Figure 2.1 shows the devel-

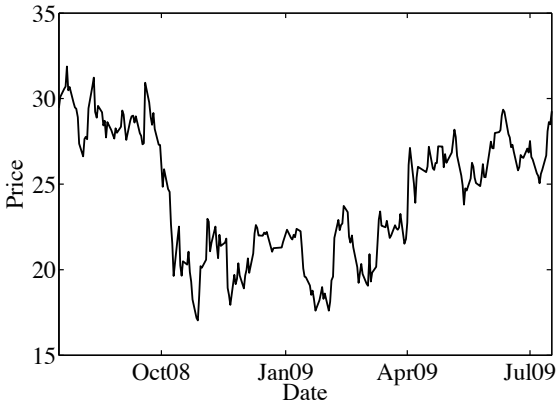



Figure 2.1: Stock market example: price history from July 17, 2008 to July 17, 2009

opment of the price between the two dates previously used to illustrate the contrasting approach, which means the price between July 17, 2008 and July 17, 2009. As one can clearly see, there was first a downward trend that turned into an upward trend showing a moderate rate of change. Given this information one would anticipate the continuation of the trend into the near future. Overall it can be said that the information obtained by looking at many time points rather than solely two reduces the uncertainty of making a buy decision.

## 2.2 Contrast and Change Mining

Many of the changes corporations are interested in, but rarely aware of, are captured in the masses of data collected during daily operations. Changes hidden in this data may mirror, for instance, external influences such as customer and market trends, but also internal influences such as supply shortages or shifts in product quality.

While data mining has traditionally concentrated on the analysis of a static world, in which data instances are collected, stored and analysed to derive models that describe the present, there is growing consensus that revealing how a domain changes is equally as important as producing highly accurate models (Liu et al., 2000; Kifer et al., 2004). Nowadays, developing methods for analysing and understanding change is seen as one of the primary research issues when dealing with evolving data (Gaber et al., 2005).

Led by practical needs, the two main subfields of data mining for analysing change—contrast mining and change mining—that have emerged reflect the two kinds of knowledge need laid out in the previous section.

**Contrast Mining** quantifies and describes the difference between *two* data sets in terms of the models they induce or the patterns they contain. It produces knowledge that describes *what* has changed as well as the *extent* of it.(Dong and Li, 1999; Ganti et al., 2002; Novak et al., 2009)

Instead of comparing two data sets directly, contrast mining approaches first learn patterns or models from the data sets which are then subsequently compared with each other. This approach has the advantage that the complexity and size of the data is

reduced while most of the information contained in the original raw data is being preserved. Nevertheless, this also means that the way how change is described and what kind of change is discovered strongly depends on which types of patterns and models are being used. Although this would theoretically yield a wide spectrum of contrast mining approaches, research has so far focused on employing frequent patterns, decision trees and clusters.

To illustrate the utility of contrast mining in real-world applications consider a retail business. For a retail manager, it is paramount to detect whether the present customer structure has significantly changed since the current marketing and sales plans were decided. If he learns that, for instance, the largest customer group used to be single twentysomethings but are now married middle agers he could revise the original plans to match the characteristics of the older customer group. Similarly, the retail manager could use contrast mining to check whether the revision was successful or had undesired side-effects by comparing data from before and after it.

While customer analytics is the area which is targeted by the vast majority of reported contrast mining applications (Song et al., 2001; Chen et al., 2005; Kim et al., 2005; Tsai and Shieh, 2009) it is not the only area it has been successfully applied to. For instance, Cormode and Muthukrishnan (2005) employed contrast mining in the field of computer network monitoring to find addresses and flows that differ significantly in traffic level compared to a reference time period.

Conventional data mining methods observe one data set and learn models or discover patterns upon it. Contrast mining methods take two data sets and compare them based on the models and patterns contained in them. They emphasise on describing *what* has changed, in terms of differences, but are unable to answer *how* a domain changes, in terms of trends. The need to also achieve the latter eventually led to the emergence of the field of *change mining*. A contrast mining it is a subarea of *higher order mining* (see Roddick et al., 2008), which encompasses methods for the discovery of knowledge by processing models (instead of data).

**Change Mining** aims at modelling, monitoring and interpreting changes in the patterns and models that describe an evolving domain over more than two, *often many*, temporally ordered data sets. (Liu et al., 2000; Böttcher et al., 2008a) It produces knowledge that enables *anticipation* of the future.

Similar to contrast mining almost every type of model and pattern can be subject to change mining. Most approaches, however, have focussed on utilising frequent patterns and clusters. Because there seems to be a trend toward building systems to store, query and analyse those 'historical' models which have been produced over time (Liu and Tuzhilin, 2008) it can be expected that the scope of change mining approaches will be further broadened in the future.

Change mining is particularly helpful in domains in which it is crucial to detect emerging trends as early as possible and which require to make decisions in anticipation rather than in reaction. One typical examples of such an application area is customer and business process analysis. Consider a business which operates in dynamic markets with customers who, driven by innovations and competing products, have highly changing demands and attitudes. In such a business it is vitally important to understand what their customers expect from them in the future. For example, in the utility industry

it is essential to understand the factors which drive customer to switch to a different provider well before these factors effect a large fraction of them. Here, change mining enables an analyst to identify emerging trends for dissatisfaction and thus to decide upon countermeasures before they become severe on a medium or long term scale (Böttcher et al., 2006a, 2009).

At last, it should be noted that the related topic of detecting change, as opposed to analysing it, has a long-standing history, in statistics and also in machine learning. In statistics a plethora of methods has been proposed for the identification of so-called *change points* (Chen and Gupta, 2000; Brodsky and Darkhovsky, 2010). In machine learning the contributions to change detection are directed toward the problem of concept drift, this means to determine whether a target concept to be learned has changed in order to quickly adapt the learner (Schlimmer and Granger, 1986; Widmer and Kubat, 1996). The difference to contrast and change mining, however, is that these methods are often meant to be embedded into the learning algorithms as a contribution to fields like incremental (online) learning such that they "can detect context changes without being explicitly informed about them" (Widmer and Kubat, 1996).

## 2.3 Change Analysis for Frequent Patterns

The problem of finding frequent patterns can be stated as follows: given a collection of sets of objects, discover the most frequent co-occurences among objects within the same set (Agrawal et al., 1993). The frequency of a set of co-occuring objects is referred to as *support* and is defined as the relative number of sets within the collection which contain it.

Frequent pattern mining emerged from market basket analysis and even though it is nowadays being applied to a much wider range of problems the used terminology still significantly borrows from this initial application domain. In market basket analysis the sets correspond to sales *transactions* and the objects correspond to purchased *items*. Consequently, co-occuring objects are called *item set*, and they are *frequent* if the support is greater than a user-specified threshold (minimum support). A typical result looks like "A basket contains the items milk, bread and butter with support 0.6". Such a result can then be used to create special offers, to arrange products in a market or for customer-tailored advertisement, assuming that the sale of one item will influence the sale of others.

Historically, frequent item sets have for a long time been seen as a first step in finding *association rules*. These are rules that imply the presence of some items based the presence of other items within the same transactions with significant certainty. An example of an association rule might be: "Among those customers who buy bread and butter, 0.9 will also buy milk." The attached relative frequency is called the *confidence*, and it measures the fraction of transactions that contain the consequent item set among the transactions that contain the antecedent item set. Nonetheless, today it is widely agreed that frequent item sets can provide insightful knowledge on their own such that the step of producing association rule can be seen as optional and is in practise often omitted.

The core ideas of *frequent item set* mining have been adapted to other structures which

coined the umbrella term *frequent patterns* that additionally embraces, for instance, *frequent sequences* (Agrawal and Srikant, 1995), for co-occurences within sets of ordered objects, and *frequent subgraphs* (Inokuchi et al., 2000; Kuramochi and Karypis, 2001), for co-occurences within sets of interrelated or interdependent objects.

Even though being the oldest, frequent item sets are still also the most popular type of frequent patterns. One reason is their versatility. Frequent item set mining can be applied to any nominal data by encoding every (attribute, attribute value) combination as an item. An item set then describes a conjunction of attributes and their values, and its support represents the relative number of instances which satisfy this conjunction. Stemming from the market basket analysis roots, the instances of a data set on which frequent item set mining is carried out are also called transactions. Generalising further, frequent item set mining can be applied to any data transformable into Boolean or 0/1 format, this means each attribute models the presence respectively absence of a real-world property. In this way, micro-array experiments and documents can be considered transactions whose items are gene expression properties respectively word occurrences.

Frequent item sets have received a high popularity as a facilitator for analysing change owing to several advantages they offer. First and foremost, they provide a nearly exhaustive overview of the patterns contained within a data set that is only limited by the chosen minimum support threshold. In this way, a rather detailed description of a data sets structure is obtained. Second, item sets are interpretable. To illustrate the interpretability consider survey data where each record corresponds to an individual. Taking the intuitive notion of a subgroup as a set individuals who share properties, an item set is a descriptor of such properties. The transactions, in turn, that contain this item set constitute the subgroup. From this point of view, frequent item set mining produces all subgroups contained in data which exceed a certain size. In more general terms it can be said that an item set represents a specific *subspace* of the data respectively the domain. Third, measures such as support, but also confidence, that were initially developed to guide the discovery process, are meaningful and practically valuable measures for these subspaces. Support measures their relative size and confidence the fraction of transactions in a subspace who have a certain property.

### 2.3.1 Contrast Mining

Consider two data sets that were collected during different time periods. If for all item sets their support would be similar across the data sets it would be reasonable to state that the data does not differ to a significant degree. In reverse, if the support of an item set differs considerably it is reasonable to state that the data set differs in the part of the attribute space described by it. The support difference can either be measured absolutely by subtracting the values, or relatively by dividing them. Nonetheless, using an absolute difference can be problematic sometimes: Because support is a relative measure, the interpretation of an absolute change varies with its value. For example, an increase by 0.05 would not be much for a support of 0.8 but for a support of 0.05 it means a doubling of the transactions covered by the corresponding item set and therefore a significant change. The larger the difference the more interesting an item set gets. In particular when a relative difference is employed it is obvious that item sets with low support have a greater chance to exhibit a strong difference than item sets with high support.

Considering that the number of item sets is exponential in the number of items it is self-evident that finding those item sets with maximal difference may require a huge computational effort. Wang et al. (2005) investigated the computational complexity of finding those item sets which are only present in one of the two data sets and have maximum support difference. They proved that the problem of finding those item sets is MAX SNP-hard. This implies that polynomial time approximation algorithms do not exist for the problem unless P=NP. Because this is a special case of the idea laid out above the result provides an indication of the computational complexity of contrast mining based on frequent patterns in general.

Dong and Li (1999, 2005) were among the first who put forward the idea of a contrast mining approach based on frequent patterns. They introduced the concept of emerging patterns as "item sets whose supports increase significantly from one data set to another". Having a (time-)ordered pair of data sets an item set is emerging if the relative difference, called *growth rate*, between its supports in the second and the first data set exceeds a user-defined threshold.

Dong and Li argue that the truly interesting emerging item sets are those with low to medium support. However, enumerating all item sets starting with the high support ones, like it is typically done in frequent item set mining, is unfeasible in this setting due to the potentially exponential number of item sets that need to be enumerated before realistic candidates for emerging item sets, i.e. those with low to medium support, are found.

To tackle this complexity challenge Dong and Li (1999, 2005) proposed to make a trade-off between the explicitness of the emerging item sets' representation and the time needed to produce them. In particular, they decided to derive the emerging patterns but not their actual growth rate and to represent the set of emerging item sets by two sets: one constituting its lower bound, the other constituting its upper bound. An item set thus is emergent if one of its subset is contained in the lower bound and one of its supersets is contained in the upper bound. The pair of both sets is called a *border description* of the set of emerging item sets. Dong and Li showed that a border description of emerging item sets always exists.

The emerging patterns paradigm has been further elaborated by different researchers. For example, Chu et al. (2009) presented a method which embeds emerging item sets into the data stream paradigm. Furthermore, several contributions have been dealing with the problem to reduce the usually large number of emerging patterns that are discovered either by incorporating additional measures of interests (Zhang et al., 2000; Fan and Ramamohanarao, 2003), by using condensed representations (Soulet et al., 2004), or by presenting only those fundamental changes which cannot be explained by others (Liu et al., 2001a).

Meanwhile a research area on its own are so-called *jumping emerging patterns* which are emerging patterns whose support increases sharply from zero in one data set to non-zero in the other (Li et al., 2001). Although they are a special type of emerging patterns they are related to much older concepts, in particular to *discriminant rules* and to *rough set reducts*. A discriminant rule is an assertion which discriminates one class of objects from other classes. (Han and Fu, 1996) A rough set reduct is a set of attributes providing complete knowledge about one particular classication hypothesis within a given

universum of objects. A jumping emerging pattern is a set of attribute-value pairs which is specific to objects belonging to one time period. Considering that a time period could also be modelled as a class, it is evident that all three concepts although emerged from different fields of research serve almost the same purpose. Indeed, basic mathematical relations between rough set reducts and jumping emerging patterns have been reported by Terlecki and Walczak (2007). No comparable analysis exists yet for discriminant rules.

In the same year in which the idea of emerging pattern mining was first presented to a wider audience the very similar concept of *contrast sets* was independently proposed by Bay and Pazzani (1999, 2001), surprisingly also at the same conference. Bay and Pazzani (1999, 2001) proposed to describe the difference between two data sets by *contrast sets* which they defined as "conjunctions of attributes and values that differ meaningfully in their distribution across groups". It goes without saying that this representation is equivalent to an item set when applied to attribute-valued data such that the problem of finding contrast sets is transformed into the problem of searching all item sets which significantly differ in their support across data sets.

To differ meaningfully an item set's support difference must exceed a user defined threshold. Different from emerging item sets the absolute difference between support values is employed such that the aforementioned complexity issues are not so predominant. In addition, contrast sets require the support values in the data sets to be statistically significantly different by means of a chi-square test. To discover contrast sets Bay and Pazzani (1999, 2001) proposed the STUCCO algorithm which performs a breadth-first search in the item set lattice. It starts with testing the smallest item sets, then tests all next-larger ones, and so on. To overcome complexity problems the algorithm prunes the search space by not visiting an item set's supersets if it is determinable that they will not meet the conditions for contrast sets or if their support values are too small for a valid chi-square test.

Contrast sets seem to have received considerably less attention in research than emerging patterns. Hilderman and Peckham (2005) presented the CIGAR (Contrasting Grouped Association Rules) algorithm which employs different statistical tests to STUCCO. Wong and Tseng (2005) showed how negations can be incorporated into the contrast set description. It was demonstrated by Webb et al. (2003) that contrast set mining can be formulated as a classification task which produces rules that predict the data set an instance belongs to. This idea was rediscovered by Hido et al. (2008) who generalised it to work with arbitrary classifiers.

The concepts of contrast set and emerging patterns were not only concurrently developed, also their informal definitions appear astonishingly similar. That the problems of finding emerging patterns and contrast sets are also formally equivalent was shown by Novak et al. (2009). They also proved that both can be formulated as subgroup discovery problems (Wrobel, 1997).

### 2.3.2 Change Mining

The underlying idea of almost every change mining approach based on frequent patterns is to detect interesting changes by analysing their support, confidence or any other measure of interest (see Geng and Hamilton, 2006) along the time axis. The starting

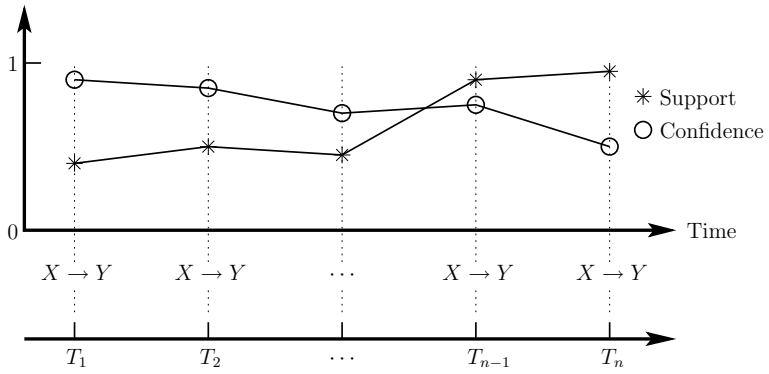



Figure 2.2: Pattern histories of support and confidence.

point of such an approach is depicted in Figure 2.2: a time-stamped data set is partitioned into intervals along the time axis. Frequent pattern discovery is then applied to each of these subsets. This yields sequences—often also called *pattern histories*—of the chosen measure for each pattern, which can be analysed further. Of particular interest are regularities in the histories which are called *change patterns*. They allow to make statements about the future development of a pattern and thus provide a basis for anticipative decision making.

At its core, change mining based on frequent patterns is a time series analysis problem. In spite of time series analysis having a long-standing tradition in statistics many of the 'classical' approaches, such as those described in the well-known text books by Chatfield (1996, 2001), cannot directly be applied to pattern histories. The main reason is that they rely on the availability of, or strong assumption on an underlying model. To fit a model a sufficiently large number of values for parameter estimation is necessary, but rarely available in change mining. Moreover, a fitted model may not always adequately describe the underlying process. For example, a regression line can be fitted to any pairs of values – independent of how the dependent value actually relates to the other. Therefore a model needs to be validated, but this typically involves human intervention. Considering the vast number of patterns typically discovered it is at hand that this in unfeasible. Pattern change mining and time series analysis, nevertheless, have in common that both aim at providing a descriptive (qualitative) statement about an evolving domain.

Research on how to analyse pattern histories for interesting changes has so far yielded into three different directions that differ considerably in terms of the employed techniques: template matching, statistical testing and heuristics. For example, consider Figure 2.3 which shows two histories. The history of item set $Z$ exhibits a very slight downward trend which can only be ascertained by means of a statistical test. Even though its support has significantly higher values than those of the item set $X$, the latter shows many apparent characteristic features which might be of interest to an expert: a trend turning point and a declining and inclining trend left, respectively right from it. These features could be analysed by means of a statistic, but also by heuristics to detect the turning point, or by template matching if one is interested in "steep declines followed by a moderate incline".

The first change mining approach based on frequent patterns was proposed by Agrawal

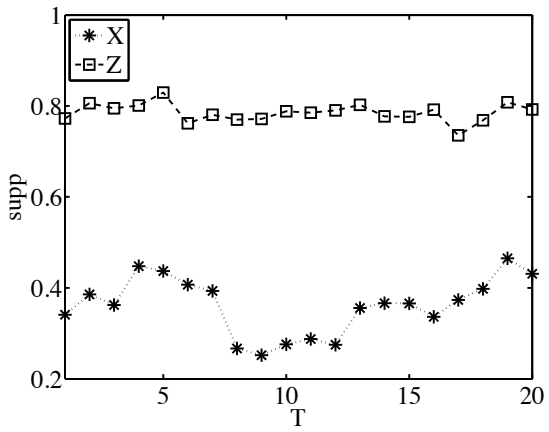



Figure 2.3: The support histories of two item sets $X$ and $Z$ taken from real-world data

and Psaila (1995). They considered a query language for the shape of the support and confidence histories of association rules. The user has to specify several shapes, which he regards interesting, and the histories are then matched against them. Several authors have further elaborated on this initial approach, for example by introducing fuzzy descriptions (Au and Chan, 2005; Steinbrecher and Kruse, 2008). It should also be noted that there are several similar methods in the field of shape retrieval from time series, such as the ones by Goldin and Kanellakis (1995) or Rafiei and Mendelzon (1997).

The work described by Liu et al. (2001b) is motivated by the observation that most users of association rules feel uncomfortable with the bare presentation of rule confidence and support. They are concerned about a rule's validity. Thus additional information about how a rule changes, if it exhibits trend, stability or other systematics, need to be provided to assist a user in judging the rule's value. The authors therefore argue that a rule's interestingness should be assessed by the characteristics of its behaviour over time. Given a measure like support or confidence, rules are assigned by means of a $\chi^2$-test (for stability) and a runs-test (for non-random behaviour) to one of three classes, if possible: semistable rules, stable rules and trend rules. Subsequently, rules are ranked within each class according to an interestingness measure. The idea to use the history of a pattern as a key to examine its interestingness has also received attention by different researchers. Böttcher et al. (2006b) proposed a framework which employs statistical tests different to those used by Liu et al. (2001b) and additionally accounts for temporal redundancies. Russ et al. (2007) considered using histories for establishing a relevance feedback process for association rules.

The framework *PAM* (pattern monitor), which is proposed in the dissertation of Baron (2004), aims to monitor rule histories in order to detect interesting short and long term changes for instance in the context of email traffic monitoring (Baron et al., 2003) and web usage analysis (Baron and Spiliopoulou, 2003). In contrast to the other approaches discussed in this section, the monitor treats a history not as a static, but as a dynamic object which is extended permanently after every new data mining session. For this reason the author makes the early detection of long term changes a key issue. The monitor's architecture basically consists of two layers. The first layer discovers, stores and maintains association rules and their histories. Each association rule is stored using

the *Generic Rule Model*, which is a record containing the discovered rule, its statistics for a particular period, a timestamp to indicate the period and a unique identifier for each rule. The second layer builds upon the Generic Rule Model and provides several heuristics to detect interesting short- and long term changes. A short term change refers to the change between two consecutive periods, whereas long term change means that the values of a rule's measure do not immediately return to their former level.

The approach by Chakrabarti et al. (1998) is different from the other methods discussed in this section, as it does not rely on a particular user-specified time partitioning. Instead, the method determines the best time partitioning for each item set individually driven by the assumption that the correlation between its constituting items should be maximally homogeneous within, but inhomogeneous between partitions. This is done by application of the *minimum description length principle* (Rissanen, 1978). For this purpose a binary coding scheme for time partitions is defined. It assigns a code length that declines with increasing homogeneity. The best partition is then chosen such that the sum of all partition code lengths is minimised. An elegant aspect of this approach is that the code length also reflects volatility — a potential measure of interestingness: Large code lengths indicate higher volatility of the correlation among the items and thus a potentially more interesting item set. The disadvantage of the method, however, is that it derives one partitioning per item set. Beyond the scalability restrictions thus implied, a juxtaposition of the evolution of different rules (e.g. overlapping ones) is more complicated.

## 2.4 Change Analysis for Decision Trees

Possibly the most commonly encountered classifier in data mining are decision trees which were introduced by Breiman et al. (1984) and Quinlan (1986). As the name already indicates a decision tree is an acyclic graph having a tree-structure. Each inner node of the tree is labelled with an attribute which is also called split attribute. For each attribute value of a split attribute an edge to a child node exists and is labeled with the attribute value. Each leaf node has a probabilities of class membership assigned to it. Given a new, unclassified instance the decision tree is interpreted starting from the root. In each inner node the instance is tested for the attribute stored within the node. According to the result of the test the corresponding edge is followed to a child node. When a leaf node is reached the most probable class assigned to it is taken as the class for the instance. In other words, each path from the root to a leaf describes a rule with the attribute and attribute values on the path forming the rule's antecedent and the majority class in the leaf the consequent.

Figure 2.4 illustrates the induction of decision trees from data as well as one of the big challenges linked with their comparison based on a simple 'block world' example. The upper part shows the distribution of training instances over the attribute space at two different time periods. Each instance belongs to one of two classes, squares and circles, each described by two attributes $A$ and $B$ with domains $\{a_1, a_2\}$ and $\{b_1, b_2\}$, respectively. Consider that decision trees with only one split attribute are being learned at the end of each period. In period $T_1$, shown in the left part, the attribute $A$ separates the classes much better than $B$ and it therefore would have been chosen as the split attribute. The split divides the attribute space into two partitions, as shown by the

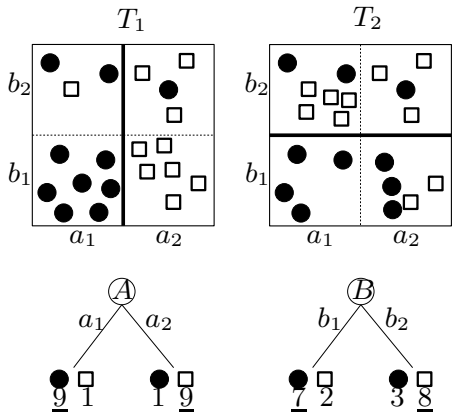



Figure 2.4: Two decision trees, each one learned from data of a different time period

thick solid line. Instances which fall into the partition defined by the attribute value $a_1$ will be classified as circles because they constitute with 9 to 1 instances the majority class, whereas instances which fall into the partition defined by $a_2$ will be classified as squares. Nonetheless, the distribution of samples shifts over time. Classifying the instances in time period $T_2$ with the decision tree learned in $T_1$ leads to a classification error of 8. Indeed, in period $T_2$ attribute $B$ would have been the superior split attribute, leading to a completely different decision tree compared to the one from $T_1$.

### 2.4.1 Contrast Mining

Due to trees being a fundamental and frequently encountered structure in computer science, the comparison of them is a rather well-researched area. For example, the *tree edit distance* problem is to compute a minimum edit script which transforms one tree into the other based on a set of edit operations, such as node insertion, deletion and relabeling. A related problem is that of *tree alignment* which aims to make two trees as similar as possible by inserting (empty) nodes in both trees such that they are structural isomorphic with a minimum label mismatch. Unfortunately, both problems are MAX SNP-hard for unordered labeled trees (see Bille, 2005). This means there exists no polynomial time approximation scheme unless P=NP. Nevertheless, for the special case that the degree of both trees is bounded the tree alignment problem can be solved in polynomial time (Jiang et al., 1995).

It is justifiable to say that structurally comparing decision trees may not be feasible in practise. First of all, although decision trees are unordered trees, they are not labeled because an attribute can be assigned to multiple nodes. For this reason, even the few algorithms for solving constrained versions of the tree edit distance and tree alignment problem cannot be applied. Even worse, decision trees are well-known for their instability (Breiman, 1996) as the previous example illustrated. Even minor changes in the data, for example introduced by some noisy instances, can lead to drastically different trees which produce very similar classifications. In contrast, drastically different classifications can sometimes be accounted to only minor changes in the tree.

For these reasons, the majority of the few publications on contrast mining using decision trees are based on comparing the classification behaviour rather than performing

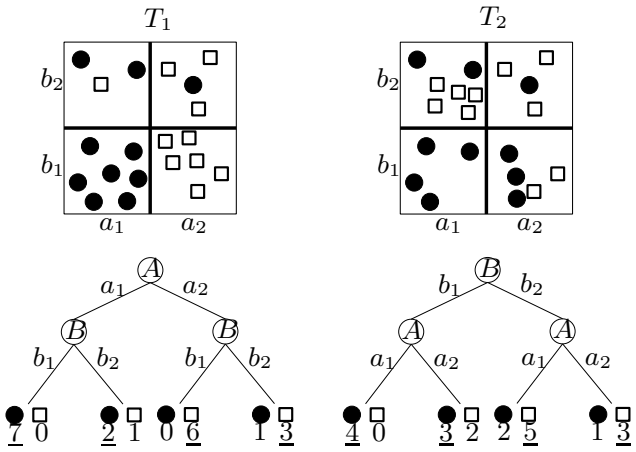



Figure 2.5: The 'new decision tree' method for comparing decision trees

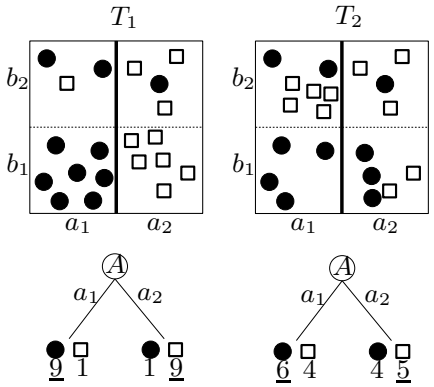



Figure 2.6: The 'same attribute and same cut' method for comparing decision trees

a disproportionally much more difficult structural tree comparison. They utilise that a decision tree describes a successive refinement of the attribute space such that eventually every path from the root to a leaf uniquely determines a partition of it. Each partition contains the probabilities of class membership of the instances which fall into it, whereby each instance can be assigned to one and only one partition. In principle, two main strategies have been proposed to compare decision trees based on the partitions they describe. They can be termed 'new decision tree' and 'same attribute and same cut' (Liu et al., 2000).

In the 'new decision tree' method, illustrated in Figure 2.5 using the previous example, a decision tree is induced independently for each data set. The attribute space partitions defined by the decision trees are then made identical by overlaying them. This means, finer partitions are constructed that constitute a common subset of the partitions defined by both trees. In each of these finer partitions the instances of both data sets which fall into it are then compared with each other, for example, with respect to their class membership probabilities. The observed differences can then be aggregated to yield an overall similarity score for the two decision trees. This method was first proposed by Ganti et al. (1999, 2002) who also showed that a *greatest common refinement* of two decision tree partitions always exists. Wang et al. (2003b) proposed a different approach

to comparing two decision trees which they call *correspondence tracing*. Rather than calculating a common refinement which they claim does not match the thinking of human experts they compare regions by means of their correspondence on a set of instances. Thereby, a region corresponds to another if the former contains a significant amount of the instances which are also contained in the latter.

The 'same attribute and same cut' method, depicted in Figure 2.6, starts with inducing a decision tree for the first data set. It then builds a decision tree from the second data set by using the same attributes and, in the case of numeric attributes, the same split points, as in the first tree in each step of the induction process. If a branch of the new tree needs to go beyond the corresponding branch in the old tree, the normal decision tree induction process is performed. This way, the obtained partitions are already comparable such that they do not need to be intersected as in the 'new decision tree' method. This method has first been considered by Liu et al. (2000) and further extended by Kim et al. (2005) who introduced heuristics to detect emerging, unexpected, added and removed rules.

Considering that decision trees describe a successive refinement of the attribute space they can also be seen as defining a hierarchy. Provided that class membership probabilities are additionally assigned to each inner node of the tree, the root node would describe the class probabilities for the whole data set, the nodes in the first level of the tree the class probabilities for the next-finer partition, and so on. Similar hierarchies can be found in OLAP (Online Analytical Processing) tools where a target variable, such as the revenue, is aggregated at various level of detail, for instance, first on country level, then on state level and then on town level. On the one hand, the difference to decision trees, however, is that the partitioning attributes in OLAP hierarchies are user-defined and invariant over time. On the other hand, the 'same attribute and same cut' method produces two decision trees with identical structure such that both approaches share some similarities. Taking this into account change explanation methods for OLAP hierarchies which have been, for example, proposed by Sarawagi (1999) and Agarwal et al. (2007) could be, but have not been yet, adapted for explaining change in decision trees.

## 2.5 Concept Drift Detection

While the analysis of change is a relatively young research topic, the question of detecting *whether* something has changed or is changing, without asking for the 'what' and 'how', can be traced back to the very early days of machine learning, where it is often associated with the term *concept drift detection*. In the following a brief introduction to the latter field is provided thereby emphasis is being put on theoretical results but also on seminal contributions.

Concept learning is an area of machine learning which deals with inferring the general definition of some concept given examples labeled as members or non-members of it, but is often also used synonymous for any task which learns a model from examples. While it is folklore that many concept definitions change over time, for example the definition of 'good weather' may now be different than a hundred years ago, this was not perceived as a problem for learning until the emergence of the first incremental, or on-line, algorithms in the mid 1980s. It was realised that many concepts depend on some hidden contexts which are not (and often cannot be) given explicitly as predictive attributes (Kubat, 1989; Widmer and Kubat, 1996). For example, the concept of 'good weather' is likely to

depend on hidden contexts relating to the global climate and the geographical location. Changes in the hidden context may introduce changes in the concept to learn. If this change is gradual it is generally referred to as *concept drift* (Widmer and Kubat, 1996). If it is abrupt the term *concept shift* is sometimes used. Because concept shift can be seen as an extreme case of concept drift many authors distinguish only between *gradual concept drift* and *abrupt concept drift*.

Mainly for the purpose of theoretical analysis concept drift is often quantified by its *rate* which measures how frequent a concept changes, its *extent* which measures how different two succeeding concepts are, and its *speed* which measures how gradual the concept changes. If the goal is to keep track of the current concept it seems to be apparent that it should become more difficult for a learning algorithm with increasing extent and rate; indeed, results from computational learning theory show that bounds exists for the maximum extent (Helmbold and Long, 1994) and rate (Kuh et al., 1990) tolerable by a learning algorithm. Based on these theoretical considerations concepts thus should be easier to track the less radical they change and the longer they are stable. Unfortunately, similar theoretical results do not exist for the speed of concept drift but experiments by several authors indicate that concept drift detection becomes a greater challenge the slower change happens (Widmer and Kubat, 1996; Lanquillon and Renz, 1999; Stanley, 2003). The slower the concept drift, the more it is difficult to distinguish the first signs of concept drift from noise, for instance mislabeled instances. Normally a compromise needs to be found between concept drift detection methods which are sensible to noise but detect drift early with a high likelihood of raising false alarms, on the one hand, and on the other hand, methods which are robust against noise but which will defer the detection decision until sufficient evidence for concept drift is available.

One consequence of concept drift is that a learning algorithm can only trust the most recent instances which enforces regular updates of the learned model. Most proposed strategies for coping with concept drift hence use variations of the sliding window idea: a window is maintained that keeps the most recent instances, and from which older instances are dropped according to some set of rules. The contents of the window is then used to detect the point in time when a concept starts drifting and, obviously, to have data to rebuild or revise the model after drift has been detected in order to maintain its quality. Considering that historically the latter was the primary motivation for developing concept drift detection methods it is not surprising that most of them are based on monitoring one or more (quality) measures. For instance, Schlimmer and Granger (1986) employ a Bayesian measure which models how well a characterisation matches a concept. Widmer and Kubat (1996) use a model's predictive performance measured over a fixed number of past classifications, and Lanquillon and Renz (1999) propose a combination of two measures; one based on the average confidence of a correct prediction on new instances and the second one describes the fraction of instances for which the confidence is below a given threshold.

Although these methods are based on heuristics they work reasonable well in practise. Due to their simplicity they also provide the advantage of being efficiently applicable which is in particular important in data stream contexts. On the downside they lack generality because they are often tightly related to a specific type of model and also they do not provide guarantees on the detection reliability. Kifer et al. (2004) were the first who addressed these shortcomings. They reformulated the problem of concept

drift detection as identifying a time point such that preceding and succeeding data are sampled from different distributions and proposed a non-parametric statistical test which allows to compare the data in the current window to the data in a reference window. Notably, Kifer et al. (2004) proved that their approach provides guarantees regarding its sensitivity while allowing to control the robustness against raising false alarms. Although much more sophisticated than the aforementioned heuristics their approach still is suitable for data stream mining and is thus seen as one of the seminal works in this area.

## 2.6 General Principles and Methodology

In general, the spectrum of approaches to analyse change is vast and only a tiny fraction of what may be achievable has been researched so far. For each type of model respectively pattern used in conventional data mining there is a manifold of ways to analyse change. Furthermore, the above discussion of contrast and change mining shows that these are two independent research streams which have been created and evolve with separate aims and objectives. Even though there may be the potential for unifying them by either generalising contrast mining approaches to many periods, or reducing change mining approaches to only two periods, this far this challenge seems to have not been addressed by the research community. Nevertheless, examination of these two research streams by juxtaposing and generalising the respective approaches reveals common concepts that can be shaped into general principles for dealing with models and patterns across time. These principles have first been published in Böttcher et al. (2008a) and are reproduced in a revised form in the following sections. It should be noted that the following discussion generalises beyond change analysis for frequent patterns and decision trees to embrace, for instance, cluster analysis. For this reason the reader is referred to the overview by Böttcher (2011) for a broader discussion on change analysis methods, that also includes cluster analysis.

### 2.6.1 Choosing the Time Periods

Time periods need to be chosen to obtain the data sets from which eventually those models respectively pattern sets are learned that are subject to change analysis. From the view point of change analysis this choice is as vitally important as it is difficult. Several aspects have to be weighed against each other.

On the one hand, a long period leads to a larger data set and thus enhances the reliability of the learned model or pattern set. However, long periods imply a coarse-grain partitioning of the time axis, in which interesting short-duration changes might be overseen and rather dramatic changes may get blurred due to ‘averaging out’ highly dynamic episodes. Also, they are less easy to locate on the time axis.

On the other hand, short periods force a more frequent re-learning of the model or pattern set from a—compared to long periods—smaller data set, implying that the model may become less robust and patterns less substantial. This makes the discovery of changes more difficult because the distinction between interesting model components, resp. patterns, and incidental noise gets more challenging (Höppner and Böttcher, 2007). At last,

it should also be kept in mind that frequent re-learning may introduce computational overhead especially when starting at each period from scratch.

More pragmatic approaches are often feasible though. For example, many applications are inherently designed around regular time intervals like days, weeks, months et cetera. For instance, customer surveys in marketing are often conducted regularly, so time periods for change analysis should be chosen in accordance with them. Additionally, the maximum number of periods is often restricted, for example, to match corporate policies that require performance reports to be generated quarterly or annually. Another option is to determine the length of the periods by fixing the size of the data sets. If all data sets have the same cardinality the risk of introducing sporadic changes due to a varying quality of models respectively patterns is reduced.

### 2.6.2 Specifying the Objects of Change

The granularity at which change should be analysed needs to be decided. Many models are a *composition* of smaller *model components*, each of which is interpretable and meaningful on its own. For instance, a decision tree is an aggregate of many classification rules, or a Bayesian network consists of probability distributions and relations thereof. For other models such as neural networks or support vector machines, a decomposition does not make sense because the individual parts would lack meaning. In contrast to the aforementioned, those models can only be observed as *monoliths*. Pattern sets, for example sets of item sets or clusters, are always decomposable, as may be expected. Nonetheless, for reasons of brevity the same terminology as for models will be used in the following to distinguish between the parts and the whole.

Both, the monolith and the individual components, are the *objects of change*. The decision for one or the other determines whether change is analysed on a coarse respectively fine level of detail. Which of the two alternatives to choose depends on factors such as the type of model for which change is analysed, the desired expressiveness of the insights about change but also the acceptable balance between the results' interpretability and the required effort to produce them.

*Monolithic approaches* recommend themselves when monitoring the performance of a model over time is at least as important as understanding the changes within the underlying domain. The model or pattern set is virtually seen as a 'black box'. For this reason, monolithic approaches are predominantly found in areas related to concept drift detection where a description of change is not of greater concern. In fact, monolithic approaches are independent of the employed model, apart from the naturally domain-driven differentiation between, for instance, predictors and classifiers.

*Compositional approaches* are advantageous to give more detailed answers to the question of what aspects of a domain are changing and how. Here, a set of structural components that jointly form a (global) model or a pattern set is examined. Corresponding components of models from different time periods are analysed for changes. Because in many cases decomposable models, such as decision trees, item set collections or cluster sets, describe a partitioning of the data space, the compositional approach enables a localisation of change. But since the type of partitioning and also the individual characteristics of model components may differ considerably for different kinds of models, compositional

approaches require an adaption to them. This, in turn, influences the type of changes that can be detected upon the selected objects of change.

In particular, change analysis for pattern sets and contrast mining for decision trees is dominated by compositional approaches. Notably, all methods that have been discussed in Section 2.3 regarding contrast and change mining for frequent patterns, and in Section 2.4 for decision trees qualify as such.

For fruitful change mining, the compositional approach seems preferable over the monolithic one. It requires that the data mining algorithm delivers interpretable models, whose components are meaningful in their own right. Yet, it is also more demanding. It requires an analyst to have an awareness on which change types are detectable by which models, and how well these change types fit the real-world problem to be solved. Depending on the real-world problem, the granularity of decomposition, and the type of the data mining problem, the *types of change* considered may be arranged in a *change taxonomy* (see Ganti et al., 1999, 2002; Spiliopoulou et al., 2006). Further, the compositional approach requires a higher computational effort, because for each model component—in the case of frequent patterns this may be thousands or even hundreds of thousands—a separate analysis task is carried out, on top of the effort to produce the model itself.

### 2.6.3 Establishing Correspondence across Time

After the time axis is suitably partitioned and the objects of change are defined, but before change eventually is analysed, corresponding models respectively components need to be (re-)identified across time.

The monolithic approach treats models as a black box such that independent of how the models actually look like, correspondence is always given. Monolithic approaches therefore often make use of measures that quantify and assess the model's behaviour or performance at time points that mark the end of periods. Change analysis is then conducted by contrasting these measures or regarding them as a time series.

The employed measures can either summarise structural properties or assess the behaviour of a model towards a test set. Generally applicable in this context are validity measures on homogeneity, separation or stability that are traditionally used for the evaluation of descriptive models (Vazirgiannis et al., 2003). For instance, Rissland and Friedman (1995) propose the *structural-instability metric* for decision trees: Two decision trees are juxtaposed by pairwise comparing their tree nodes and counting (and weighting) (mis)matches. Predictive models devise measures of their predictive power such as accuracy, variance and F-measure. For instance, the *conceptual equivalence metric* proposed by Yang et al. (2005) captures the equivalence between two models by comparing their classification verdict on each instance of a given test set.

For the compositional approach the problem of establishing correspondence is much more difficult to solve because components may not always be identical across time, yet describe the same real world entity. Important examples include classification rules are getting more specific or more general, or clusters are drifting, splitting, merging or simply (dis-)appear. On the opposite side, other model components may not change in their structure but only in associated measures. For instance, an item set can be seen

as time-invariant, i.e. it is formally present at any point in time and no item set can change into one having fewer, more, or other items. This is because the support—a measure—determines the item set's degree of validity, with zero support indicating that the item set is not present at all.

The distinction between a *structural component* and a *measure component* which jointly form a model component is also applicable to other models, for example decision trees or sets of clusters (see Ganti et al., 1999, 2002; Bartolini et al., 2004, 2009). The structural component often describes a region in data space whereas the measure component comprises a measure—or possibly a set of measures—for it. For instance, a structural component of a decision tree is a single classification rule's antecedent and the region in data space it encodes, whereas a measure component is the class label distribution in this region.

This distinction enables for establishing correspondence across time by first matching the structural components and then *quantifying* change based on the resulting series of measure components. This leads to statements on whether, for instance, a model component is *moving*, *shrinking*, *expanding* or—to pick up on the above decision tree example—its class label distribution changes. Moreover, the mere attempt to match on itself already allows for a *qualitative* description of change. If it is not successful, the model component qualifies for being an *emerging component* or a *vanishing component*. If it is successful on the long run, the model component may be a *stable component*.

Two strategies offer themselves for matching structural components. A *data-driven strategy* aims to identify those structural components among all possible pairings that have the most significant overlap in the subset of data they describe respectively are applicable to. This overlap can be determined in several ways. For instance, it can be the union of the regions in the data space described by the components (e.g. Ganti et al., 1999, 2002). It can also be determined based on the set of instances covered by one component which are also covered by the other (e.g. Spiliopoulou et al., 2006; Höppner and Böttcher, 2007). A *representation-driven* strategy aims to identify those structural components among all possible pairings that are significantly similar in their symbolic representation (see Bartolini et al., 2004, 2009). The similarity is accessed, for instance, by counting structural differences (e.g. Rissland and Friedman, 1995) or by transforming the representation such that common similarity measures are applicable (e.g. Russ et al., 2007). In a way, the representation-driven strategy focuses on the syntax of a structural component's representation rather than its semantic as the data-driven strategy does. Further, the representation-driven strategy is less frequently encountered in related publications as opposed to the data-driven.

Turning back to the problem of analysing a series of measure components, it needs to be observed that many of the measures employed for the monolithic approaches are aggregates of measurements on model components, for example accuracy and variance. Further, contemporary research shows that model components can, on a local scale, be model themselves (see Hand, 2002; Rüping, 2006). It therefore comes with no surprise that the same toolset employed in the monolithic approach can also be employed here, now to model components rather than the complete *global model*. Conversely, models may be build from (local) patterns (see Fürnkranz and Knobbe, 2010) such that ideas from compositional approaches are transferrable to monolithic ones.

The fundamental difference however is that in the compositional approach each component may be interrelated or interdependent with others. This interplay between components can change and should therefore be addressed by compositional approaches. It can neither be detected by looking at each component in isolation nor by monolithic approaches. In part, approaches that follow the above mentioned data-driven strategy for matching model components are applicable here too. Instead of assessing the overlap across time periods it is assessed for model components within the same single time period. A special form of overlap is containment which often manifests itself by one model component being a specialisation of another. For instance, one item set may be a proper superset of another one. Each instance for which the more specific model component is applicable is also covered by the more general one. The related data spaces thus contain respectively enclose each other. The well-known confidence measure for item sets is an example of how containment can be assessed.

## 2.7 Conclusion

It is crucial in many domains to discover, understand and assess ongoing changes early and to be able to automatically (re-)act in time. As a response two research fields have emerged in data mining which are addressing this knowledge need and that can be distinguished by the type of knowledge they produce. The field of contrast mining deals with detecting all significant differences between two time periods, or generally, two data sets. The field of change mining is concerned with discovering how a domain changes with an emphasis on long term monitoring and change pattern detection.

With the increasing number of change aspects current methods are dealing with also the number of research challenges increased. The biggest and also oldest challenge is how to distinguish noise from true changes. This challenge not only concerns the quality of the data but also the quality of the models and patterns to which change detection methods can be applied. Another challenge is the computational complexity of comparing structures, like trees or large pattern sets. Sometimes not even polynomial time approximation schemes exist such that a crucial research aim is to find a good balance between the desired knowledge about change and the feasibility to discover it in reasonable time. Last but not least, in many domains it is not only essential to detect change, but also to know which changes will occur in the future. Although from a practical point of view highly desirable the aspect of predicting change, however, has so far received only little attention in research. Still, this last challenge is somewhat easier to tackle than the first two and it can be expected to be solved in the near future.

Chapter 3

# Utilising Change for Item Sets

One approach to understand change within a domain is to analyse how patterns evolve. As the previous chapter illustrated a plethora of approaches have been proposed that target different aspects of change—almost all have in common that they require a temporally ordered sequence of data sets, each one corresponding to a time period. However, focusing on frequent item sets as one of the most prominent pattern types, a core problem still is unresolved. The number of item sets found in a data set often is often so huge, that an expert can hardly, if at all, inspect them in full.

This chapter proposes temporally closed item sets as a novel solution to this problem. Temporally closed item sets make use of the temporal dimension and thus utilise change. Still, they are compatible to both change mining approaches and existing reduction approaches for sets of item sets. Hence, temporally closed item sets are one example, how the utilisation of change is able to help improve on shortcomings of existing data mining approaches.

This chapter is organised as follows. Section 3.1 sheds light on the item set quantity problem and is followed by the problem statement in Section 3.2. Section 3.3 provides an introduction into frequent item set discovery and introduces the terminology and notion used throughout the reminder of this chapter. Existing approaches, so-called condensed representations, for reducing the number of item sets are discussed in Section 3.4. Section 3.5 describes representations of time in data sets. It is discussed how these representations are reflected in the results of condensed representation approaches if they are unaware of the representations' temporal meaning. Section 3.6 introduces the concept of temporal redundancy on which a novel time-aware condensed representation, defined in Section 3.7, rests.

## 3.1 Motivation

The ability to discover all patterns is an item set learner's strength and likewise its weakness. Usually the number of discovered item sets can be immense, easily in the thousands or even tens of thousands. This is particularly evident when dense data sets are analysed. According to Bayardo et al. (2000) dense data sets have any or all of the following properties: many frequently occurring items, strong correlations between several items, and many items in each transaction.

While market-basket data is mostly sparse, in many other domains dense data is being produced, for instance in the social sciences (census data) and in the telecommunication business. Because dense data is much more common than sparse data, item set quantity constitutes a significant practical problem: experts are forced to sift through the mass of discovered item sets to find the few that are truly important for them—a likely long and tedious task.

The *item set quantity problem* fuels a need for methods that reduce the number of item sets without reducing the knowledge they represent. *Condensed representations* address this need. They facilitate *redundancies* in the item sets. Redundancy may arise if an expert, with application of his background knowledge, is able to infer from the occurrence and properties of some item sets the occurrence and properties of others. Essentially, observed and inferred item sets have the same meaning to the expert.

The long duration needed by an expert to analyse a collection of item sets is in strong contrast to the relatively short duration needed for producing the collection from a data set. Also, data tends to be collected regularly. This led to the need for storing item sets. In fact, collections of item sets stemming from multiple time periods often are stored one after the other. The reasons are that experts want to have access to former item sets, for instance to justify past decisions, but also the prospect of conducting change analysis.

However, condensed representations to reduce the number of item sets only take into account a single data set. Similarly, if a temporal sequence of sets of item sets is available, condensed representations would consider each set individually. Condensed representations thus do not account for any temporal information and consequently ignore change and redundancies imposed by it. This ignores the opportunity for a significant further reduction in the number of item sets which would result from employing more, or broader forms of redundancy that rest upon consideration of the temporal dimension, and the change it embodies.

## 3.2 Problem Statement

The previous discussion leads to the following problem statement: Given a sequence of data sets each corresponding to a time period produce a condensed representation of the item sets contained therein that takes into account the temporal dimension. Thereby, it is assumed that the time periods are disjoint and that no more than one data set exists for each time period.

Any good solution constitutes an improvement over established condensed representations. To be eligible, the solution thus needs to be consistent to an established condensed representation—not only practical, based on experimental results, but also theoretical, based on conceptual similarities and equivalencies. Indeed, the question is whether to develop a condensed representation ‘from scratch’ or use the strength of an existing condensed representation as a leverage for the desired solution. Following the latter alternative makes more sense than following the first, to enforce consistency and to avoid developing ‘yet another condensed representation’. Moreover, to leverage on the strength of an established condensed representation, also means to rethink its weaknesses and to overcome them, at least in part.

Based on the aforesaid, the following requirements are identified:

- **Consistency:** The solution is a (temporal) extension or generalisation of an existing and established condensed representation, which hereafter will be referred to as the *reference representation*. Especially, it will produce no result that contradicts the reference representation.

- **Reduction:** Applying the solution to a sequence of data sets yields a smaller representation than the one obtained from producing the reference representation from the union of all data sets.
- **Robustness:** The effectiveness of the solution does not strongly depend on the level of noise in data.

As mentioned earlier, one reason sequences of item set collections are stored chronology preserving is the option to carry out change analysis. It is clear that an expert must have the very same option on the condensed representation to be developed, without having to accept essential losses in the knowledge about change that is discoverable. This leads to one further requirement:

- **Meaningfulness**: The type of redundancy that decides whether an item set is contained in the condensed representation should be meaningful and designed such that no potentially interesting change is lost.

Overall, on the way to find a solution for the above problem, several questions need to be answered. First and foremost, which of the available condensed representations is suitable as reference representation? What is a meaningful and suitable definition of redundancy under consideration of the temporal dimension, and how can it be robustly tested? What is the size of the resulting condensed representation? In this order, the following sections answer these questions.

## 3.3 Terminology and Notation

This section extends the commonly used terms for item set mining—motivated and informally introduced in Section 2.3—with a formal notation that will be used throughout the remaining chapter.

Frequent item set discovery is applied to a set $D$ of *transactions* $I \in D$. Every transaction $I$ is a subset of a set of literals $L$. These literals are called *items* and a subset $X \subseteq L$ with $|X| = k$ a *k-item set*, or short *item set*. It is said that a transaction $I$ *supports* an item set $X$ if $X \subseteq I$.

The significance of an item set $X$ is measured by its *s*upport defined as the fraction of transactions which contain $X$, and is also depicted in Figure 3.1:

$$\text{supp}(X) := \frac{|\{I \in D \mid X \subseteq I\}|}{|D|} \tag{3.1}$$

The nominator in Equation 3.1, which expresses the number of transactions that contain $X$, is called *absolute support* and is denoted by $\text{supp}_a(X)$.

If $X \subset Y$ holds for two item sets $X$ and $Y$ it is said that $X$ is *more general* than $Y$ because $X$ puts less restrictions on the underlying transaction set. The other way around, $Y$ is termed *more specific* than $X$. Furthermore, we define $XY := X \cup Y$ for simplicity. Obviously, the more specific an item set gets, the less it is supported by transactions and thus the smaller its support, i.e. for $Y \supset X$ it is $\text{supp}(X) \geq \text{supp}(Y)$. This is commonly referred to as the *downward-closure property* of item sets.

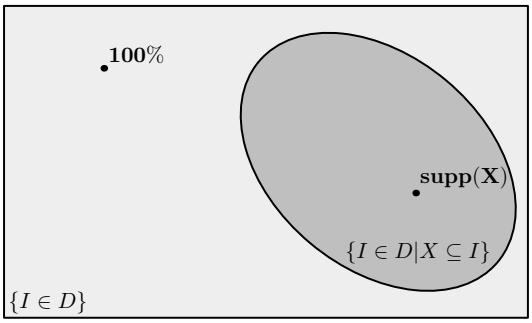



Figure 3.1: $\mathrm{supp}(X)$ is the relation between the cardinality of the set of transactions which contain $X$ relative to the size of the transaction set. Both sets are marked by a dot.

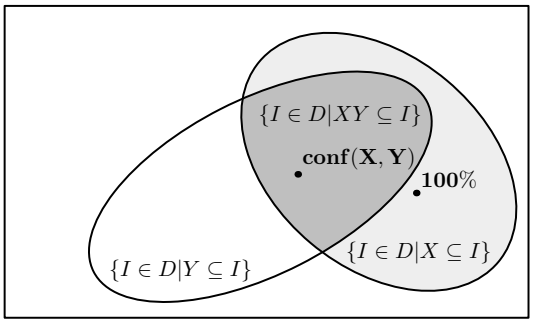



Figure 3.2: $\mathrm{conf}(X, Y)$ is the relation between the cardinality of the set of transactions which contain $XY$ relative to the set of transactions with $X$. Both sets are marked by a dot.

The *confidence* measure is used to assess the degree of overlap between two item sets $X$ and $Y$ in terms of transactions that support both. It measures the ratio of transactions that contain $Y$ additionally to $X$ with regard to the number of transactions that contain $X$. It is defined as follows, and also depicted in Figure 3.2:

$$\mathrm{conf}(X, Y) := \frac{|\{I \in D \mid X \cup Y \subseteq I\}|}{|\{I \in D \mid X \subseteq I\}|} \tag{3.2}$$

When mining item sets a vast number of candidate sets must be considered. The discoverable number of item sets grows exponentially with $|L|$. Practically, it is never required nor wanted to mine such a large number of item sets. The search space can be reduced by employment of a higher threshold $\sigma$ for support. An item set is *frequent* if its support is greater than or equal to $\sigma$. The set of *frequent item sets* is defined as

$$\mathit{Freq}(D, \sigma) := \{X \ : \ \mathrm{supp}(X) \geq \sigma\} \tag{3.3}$$

**Example 1.** Consider Table 3.1 which shows an example data set $D$ consisting of 14 transactions that are subsets of the set of items $L = \{a, c, d, t, w\}$. Each transaction is uniquely distinguishable by its transaction identifier (TID) shown in the first column. Choosing a minimum threshold for absolute support $\sigma_a = 4$ (i.e. to be frequent an item set has to be contained in at least four transactions), the 27 frequent item sets shown in Table 3.2 will be found.

Now, assume $D$ being a data set where each transaction is associated with a certain point in time. Further, let $[t_0,\ t_n]$ be the minimum time span that covers all its transactions. The interval $[t_0,\ t_n]$ is divided into $n > 1$ periods $T_i := [t_{i-1}, t_i]$, such that the corresponding subsets $D_i \subset D$ are characterised by the following three properties. First, each has a size $|D_i| \gg 1$. Second, the data sets $D_i$ do not share transactions, i.e. $D_i \cap D_j = \varnothing,\ i \neq j$ and, third, they represent a complete partitioning of $D$, i.e. $\cup_{i=1}^{n} D_i = D$. Based on their chronological order, the $D_i$ form a sequence of data sets. The set of all time periods will be denoted by $\hat{T} := \{T_1, \ldots, T_n\}$.

The same terminology and notation that has previously been discussed in the context of $D$ is also applicable to each individual $D_i$. The significant difference however is that the

| TID | Items | | | | |
|---|---|---|---|---|---|
| | a | c | d | t | w |
| 11 | 1 | 1 | | 1 | 1 |
| 12 | 1 | 1 | 1 | 1 | 1 |
| 13 | 1 | 1 | | 1 | 1 |
| 14 | | 1 | 1 | 1 | |
| 15 | 1 | 1 | | 1 | 1 |
| 16 | 1 | 1 | 1 | 1 | |
| 17 | 1 | 1 | 1 | 1 | 1 |
| 18 | | 1 | 1 | | 1 |
| 21 | 1 | 1 | | 1 | 1 |
| 22 | | 1 | | 1 | |
| 23 | 1 | 1 | | 1 | 1 |
| 24 | 1 | 1 | 1 | | 1 |
| 25 | 1 | 1 | 1 | 1 | 1 |
| 26 | | 1 | | 1 | |

Table 3.1: Example data set $D$

| $\text{supp}_a$ | Frequent Item Sets |
|---|---|
| 14 | c |
| 12 | t, ct |
| 10 | a, w, cw, ac |
| 9 | aw, at, acw, act |
| 8 | tw, atw, ctw, actw |
| 7 | d, cd |
| 5 | dw, ad, dt, cdw, acd, cdt |
| 4 | adt, adw, acdw, acdt |

Table 3.2: Frequent item sets contained in $D$ which satisfy $\sigma_a = 4$

measures of support and confidence are now related to a specific time period $T_i$. This will be indicated by using the notation $Freq_i(D, \sigma)$ for the set of frequent item sets, $\text{supp}_i(X)$ for support and $\text{conf}_i(X)$ for confidence. An item set $X$ which has been discovered in all periods is therefore described, for instance, by $n$ support values. Imposed by the order of time the values form sequences $(\text{supp}_1(X), \ldots \text{supp}_n(X))$ which are called *support histories*. *Confidence histories* are defined likewise.

## 3.4 Condensed Representations

Informally, the idea of *condensed representations*, is to compute a subset of the set of frequent item sets which might be as concise as possible while allowing to restore from it the whole collection of frequent item sets together with some of their characteristic properties, like their support, without looking at the data. The resulting set of item sets thus is smaller than the original set but still provides the same information to a user.

Condensed representations are assessable by two desirable properties.

- **Lossless**: A condensed representation is *lossless* if it allows, first, to determine for each item set whether it is frequent and, second, to restore for each frequent item set particular interestingness heuristics. Losslessness can mean an *exact* restoration or an *approximate* restoration depending on whether noise should be accounted for.
- **Concise**: A condensed representation is *concise* if it is significantly smaller than the original, unreduced collection of frequent item sets (Bonchi and Lucchese, 2004). A way of measuring conciseness therefore is counting item set.

While it was Mannila and Toivonen (1996) who first proposed the idea of condensed representations within a theoretic framework called *$\varepsilon$-adequate representations*, it was Zaki and Ogihara (1998) and Pasquier et al. (1998) who put this idea into practice

which independently led them to a condensed representation called *closed item sets*. Many other useful condensed representations have been proposed since then, like *δ-free item sets* (Boulicaut et al., 2000, 2003), *disjunction free sets* (Bykowski and Rigotti, 2001) and *non-derivable item sets* (Calders and Goethals, 2002, 2007). In the following the basic principles of condensed representations will be explained on the example of closed, $\delta$-free and disjunction free item sets. The interested reader is referred to Calders et al. (2005) for a survey on all four of the above representations.

### 3.4.1 Closed Item Sets

A closed item set is the largest item set common to a set of transactions, this means no larger item set can be found that is supported by exactly the same set of transactions. From this it follows that for a closed item set no larger item set exists which has the same support. Vice versa, if one item set is a subset of another and both have the same support also the transactions which contain them must be the same. This leads to the following definition:

**Definition 3.1** (Closed Item Set)**.** *An item set $X$ is a closed item set if and only if there exists no proper superset $Y \supset X$ such that* $\text{supp}(X) = \text{supp}(Y)$*. Equivalently, an item set $X$ is called non-closed iff there exists a proper superset $Y$ such that* $\text{supp}(X) = \text{supp}(Y)$*.*

The set of all closed item sets with respect to a data set $D$ will be denoted as $\mathit{Closed}(D)$ and the set of all frequent closed item sets with respect to a support threshold $\sigma$ denoted $\mathit{FreqClosed}(D, \sigma)$.

The largest superset $Y$ of $X$ having the same support as $X$ is called the *closure* of $X$ and a closed item set is equal to its closure. It can be shown that for any item set a closure exists and that this closure is unique (Pasquier et al., 1999; Zaki, 2004). The support of any (non-closed) item set can therefore be determined by the support of its smallest closed superset. Similarly, to decide whether an item set is frequent it suffices to find a frequent closed item set which is a superset. For these reasons the set of (frequent) closed item sets is a condensed representation of the set of (frequent) item sets.

Several algorithms have been proposed to efficiently discover the set of closed item sets from a given data set, for example: Close (Pasquier et al., 1999), Closet (Pei et al., 2001) and its extensions like Closet+ (Wang et al., 2003a), and CHARM (Zaki and Hsiao, 2005).

**Example 1 (continued).** The frequent item sets shown in Table 3.2 are represented by the frequent closed item sets listed in Table 3.3. It should be noticed that the number of frequent closed item sets is with 13, compared to 27, significantly smaller than the number of frequent item sets. The item set *atw* is non-closed because it has the same support as *actw*—one of its supersets, whereas *actw* is closed because all of its supersets have a lower support. The item set *actw* thus is the closure of *atw*.

Another perspective on closed item sets can be derived from Definition 3.1 by considering association rules and the definition of their confidence: an item set $X$ is non-closed with a closure $Y \supset X$ iff a rule $r : X \Rightarrow Y \setminus X$ exists with $\text{conf}(r) = 1$. It is almost certain, on the one hand, that in real-world data such *strict* rules, i.e. rules that hold with absolute certainty, are likely uncommon, particularly when the data sets are large and noisy. On

| $\text{supp}_a$ | Freq. Closed Item Sets |
|---|---|
| 14 | c |
| 12 | ct |
| 10 | cw, ac |
| 9 | acw, act |
| 8 | actw |
| 7 | cd |
| 5 | cdw, acd, cdt |
| 4 | acdw, acdt |

Table 3.3: Closed item sets contained in $D$ which satisfy $\sigma_a = 4$

| $\text{supp}_a$ | Freq. 0-Free Item Sets |
|---|---|
| 14 | c |
| 12 | t |
| 10 | a, w |
| 9 | aw, at |
| 8 | tw |
| 7 | d |
| 5 | dw, ad, dt |
| 4 | adt, adw |

Table 3.4: 0-free item sets contained in $D$ which satisfy $\sigma_a = 4$

the other hand, if rules are strict despite noise they presumably represent basic, already known facts about the data's underlying domain. Ideally, this means that given the item sets $X$ and $Y$ and knowing that $X \Rightarrow Y \setminus X$ is always true, experts can easily conclude that both item sets must have the same support. This way, $X$ and $Y$ can be termed *redundant* because they provide the same knowledge from the experts' point of view: Knowing the support of either of them and then learning about the other does not yield any new insights.

### 3.4.2 $\delta$-Free Item Sets

Two possibilities can be thought of when assuming an ideal world in that an expert knows any basic facts expressible by strict association rules. Both, in light of the previous discussion, can be interpreted as a form of redundancy reduction based on what are (implicitly) assumed to be well-known facts: Either only those item sets are kept to which no strict rule can be applied anymore, or only those are kept which cannot be produced by applying strict rules to smaller item sets. The first possibility yields closed item sets whereas the second leads to an alternative condensed representation known as *free item sets*.

Boulicaut et al. (2000, 2003) proposed $\delta$-free item sets, or short *free item sets*, based on the notion of a *$\delta$-strong rule*. Informally, a $\delta$-strong rule is an association rule $X \Rightarrow^{\delta} a$, where $X \subseteq L, a \in L \setminus X$, which is 'almost strict', this means it is violated in no more than $\delta$ transactions whereby $\delta$ is supposed to be small.

**Definition 3.2** ($\delta$- Free Item Set)**.** *An item set $Y$ is a $\delta$-free item set if and only if there is no valid $\delta$-strong rule $X \Rightarrow^{\delta} a$ such that $X \subset Y, a \in Y$, $a \notin X$.*

The set of all $\delta$-free item sets with respect to a data set $D$ will be denoted as $\mathit{Free}^{\delta}(D)$ and the set of all frequent $\delta$-free item sets with respect to a support threshold $\sigma$ denoted $\mathit{FreqFree}^{\delta}(D, \sigma)$.

**Example 1 (continued).** The frequent item sets shown in Table 3.2 contain the frequent 0-free item sets listed in Table 3.4. For instance, the item set $actw$ is non-0-free because the 0-strong-rule $atw \Rightarrow^{0} c$ holds for the data set, but also $atw$ is non-0-free because $tw \Rightarrow^{0} a$ holds too. In contrast, $tw$ is 0-free because neither $t \Rightarrow^{0} w$ nor $w \Rightarrow^{0} t$ are valid 0-strong-rules.

Unlike a non-closed item set which can be related to one and only one smallest closed superset, a non-$\delta$-free item $X$ set can have multiple $\delta$-free item sets that are a subset of $X$ and whose support could serve as an approximation. The best approximation can be obtained by choosing among all $\delta$-free item sets included in $X$ the one that has minimum support (Boulicaut et al., 2003).

In case of $\delta = 0$ the support can be exactly determined. In fact, 0-free item sets correspond to the notion of *key patterns* and *generators* developed independently by Bastide et al. (2000), respectively Pasquier et al. (1999), and which they used for closed item set generation. These works also establish a link between closed and 0-free item sets: any frequent closed item set is the closure of at least one 0-free item set.

Even though the frequent $\delta$-free item sets are sufficient to approximate (or determine) the support of any other frequent (non-$\delta$-free) item set they are not sufficient to decide whether an item set is frequent, or not. This entails that without knowledge about whether an item set's support is greater than the threshold it is impossible to approximate the item set's support.

**Example 1 (continued).** Consider the item sets *actw* and *dtw* which both are not in the set of frequent 0-free item sets shown in Table 3.4. Both have no frequent free superset but a frequent free subset: *tw* for *actw* with $\text{supp}_a(tw) = 8$, and *dw* for *dtw* with $\text{supp}_a(tw) = 5$. This would imply that *actw* and *dtw* both are frequent, with absolute supports $\text{supp}_a(actw) = 8$ and $\text{supp}_a(dtw) = 5$, respectively. In fact, for *actw* this is true, whereas *dtw* in truth is infrequent with $\text{supp}_a(dtw) = 3$ as can be seen from Table 3.1.

Intuitively, the set of frequent $\delta$-free sets must be supplemented with the set of the smallest non-frequent $\delta$-free sets. This can be expressed using the concept of a *negative border* (Mannila and Toivonen, 1997).

**Definition 3.3** (Negative Border)**.** *The negative border of a set of item sets $J$, denoted $Bd^-(J)$, is the set $\{X \mid X \subseteq L \wedge X \notin J \wedge (\forall Y \subset X : Y \in J)\}$.*

Formally, the set of frequent $\delta$-free sets $FreqFree^\delta$ thus needs to be complemented by the set $Bd^-(FreqFree^\delta) \cap Free^\delta$. Now, given any item set $Y$, if there exists $X \subseteq Y$ such that $X \in Bd^-(FreqFree^\delta) \cap Free^\delta$, then $Y$ must be infrequent. In the other case, the support of $Y$ can be approximated, respectively exactly determined for $\delta = 0$, as described above.

### 3.4.3 Disjunction Free Item Sets

Starting from the concept of 0-free item sets different alternative condensed representations can be derived by allowing for more complex rules than those of form $X \Rightarrow^0 a$. One possibility that has been put forward is to use *disjunctive rules* of the form $X \Rightarrow a \vee b$ where $X \subset L$ and $a, b \in L \setminus X$ (Bykowski and Rigotti, 2001).

**Definition 3.4** (Disjunction-Free Item Set)**.** *An item set $Y$ is a disjunction-free item set if and only if there is no valid disjunction rule $X \Rightarrow a \vee b$ such that $X \subset Y$, $a, b \in Y \setminus X$.*

In the following, the set of all disjunction-free item sets with respect to a data set $D$ will be denoted as $DisFree(D)$ and the set of all frequent disjunction-free item sets with respect to a support threshold $\sigma$ denoted $FreqDisFree(D, \sigma)$.

| $\text{supp}_a$ | Freq. Disj.-Free Item Sets |
|---|---|
| 14 | c |
| 12 | t |
| 10 | a, w |
| 9 | aw, at |
| 8 | tw |
| 7 | d |
| 5 | dw, ad, dt |
| 4 | |

Table 3.5: Disjunction-free item sets contained in $D$ which satisfy $\sigma_a = 4$

Further, it should be noticed that $\mathit{FreqDisFree}(D, \sigma)$ must be a subset of the frequent 0-free item sets $\mathit{FreqFree}^0(D, \sigma)$ because valid rules of the form $X \Rightarrow^0 a$ on which the concept of 0-freeness is based can be rewritten as disjunctive rules $X \Rightarrow^0 a \vee a$.

**Example 1 (continued).** The frequent item sets shown in Table 3.2 contain the frequent disjunction-free item sets listed in Table 3.5. For instance, the item sets $atw$ and $adw$ are non-disjunction-free because the disjunction rules $a \Rightarrow t \vee w$, resp. $a \Rightarrow d \vee w$, are valid for the data set $D$ (see Table 3.1).

Similarly to $\delta$-free sets, disjunction free item sets need to be augmented by the set $Bd^-(\mathit{FreqDisFree}) \cap \mathit{DisFree}$ to determine whether a given non-disjunction-free item set is frequent. In addition, the set of all valid disjunction rules need to be stored, because without them it is impossible to efficiently determine the support of a frequent non-disjunction-free item set, (Calders et al., 2005).

As a more space-efficient but also more time-demanding alternative Bykowski and Rigotti (2001) proposed to (recursively) reconstruct all valid disjunctive rules from $\mathit{FreqDisFree}$ and $Bd^-(\mathit{FreqDisFree})$ by searching for four item sets $X$, $Xa$, $Xb$ and $Xab$ such that $\text{supp}(X) = \text{supp}(Xa) + \text{supp}(Xb) - \text{supp}(Xab)$. If and only if the latter equation is satisfied the disjunctive rule $X \Rightarrow a \vee b$ is valid.

Lastly, it should be mentioned that disjunction-free item sets can be further generalised towards rules of the form $X \Rightarrow a_1 \vee \ldots \vee a_i \vee \ldots \vee a_n$ as has been pointed out by Bykowski and Rigotti (2001) and put into practise by Kryszkiewicz and Gajek (2002).

### 3.4.4 General Principles

Producing a condensed representation means to partition the collection of (frequent) item sets such that within each partition item sets have a particular property in common, but across partitions no two share the same. For 0-free and closed item sets this property is that the item sets are supported by the identical set of transactions. This implies that within a partition the support values are all equal. In general terms, it makes sense that the chosen property is linked to interestingness heuristics for item sets, and albeit the discussed approaches restrict the latter to support also other heuristics such as lift are possible (see Soulet and Crémilleux, 2008).

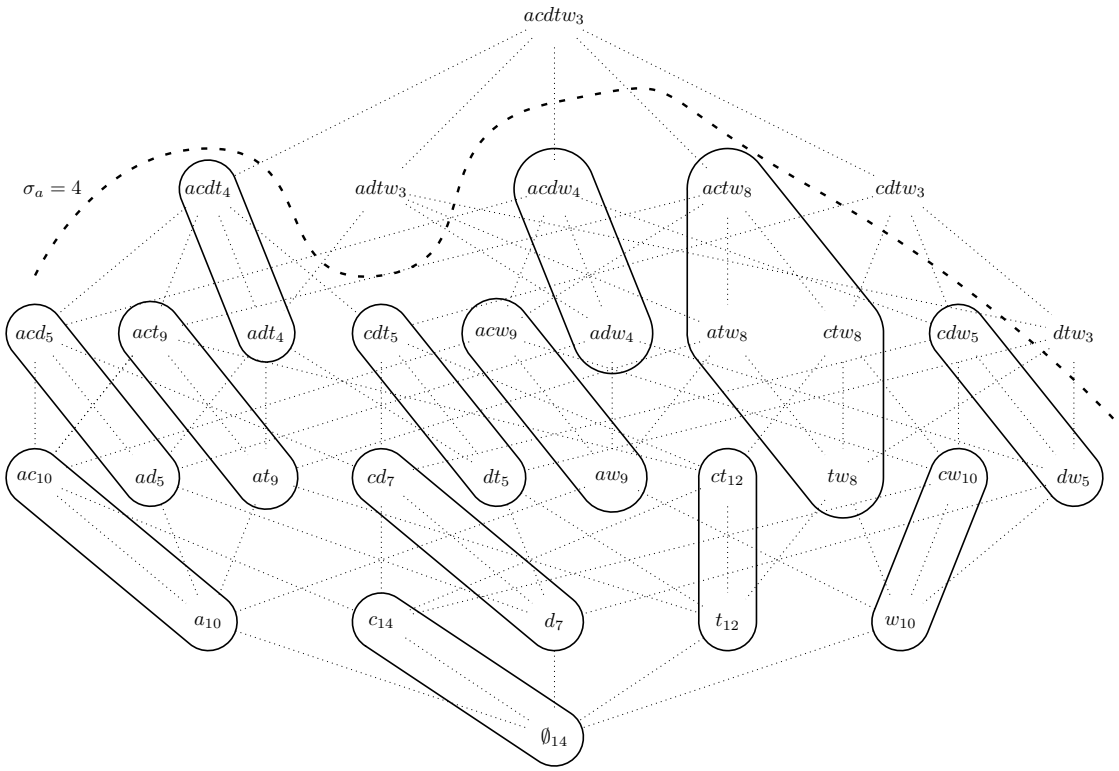



Figure 3.3: Hasse diagram for the item sets from Example 1 with equivalence classes of item sets which are supported by the identical set of transactions.

In other words, item sets of the same partition are equivalent to each other; the partitions themselves are *equivalence classes*. The item sets are equivalent because they are indiscernible with respect to the property that formed the partitions. They are also *redundant* because they represent the same information. A major step in reducing the number of item sets and eventually obtaining a condensed representation therefore is to choose a set of equivalence class representatives for each partition. Given an item set, the ideal choice of representatives allows to identify its corresponding partition, while, at the same time, being minimal in size. It makes sense to choose either the smallest (greatest lower bound), the largest item sets (least upper bound), or both as equivalence class representation, rather than arbitrary elements.

**Example 1 (continued).** Figure 3.3 shows the equivalence classes with respect to support of the frequent item sets displayed in Table 3.2. Each item set's support is shown as subscript. Comparison of the smallest and largest item sets in each equivalence class with Table 3.4 and Table 3.3 reveals that these are the frequent free, respectively frequent closed item sets.

Having only equivalence class representatives to describe the condensed representation's item sets is sometimes insufficient. As, for example, the cases of 0-free and disjunction free item sets show: having only the representative is not enough to determine whether a certain item set is frequent or its actual support. For this reason, these condensed representations introduced two *supplemental structures*. In large, this decomposes condensed representations into three building blocks: the *subset of the original collection*

| Representation | Rule Set Elements | Border Set |
|---|---|---|
| *FreqClosed* | $X \Rightarrow \mathbf{X} \cup \{\mathbf{a}\} \setminus X$ | |
| $FreqFree^0$ | $\mathbf{X} \Rightarrow X \cup \{a\} \setminus X$ | $Bd^-(FreqFree^0) \cap Free^0$ |
| $FreqFree^\delta$ | $\mathbf{X} \Rightarrow^\delta X \cup \{a\} \setminus X$ | $Bd^-(FreqFree^\delta) \cap Free^\delta$ |
| *FreqDisFree* | $\mathbf{X} \Rightarrow (X \cup \{a\} \setminus X) \vee (X \cup \{b\} \setminus X)$ | $Bd^-(FreqDisFree) \cap DisFree$ |

Table 3.6: Condensed representations and their building blocks.

*of item sets*, a *set of rules* and *a (negative or positive) border set*. The subset of the original collection, in an intuitive sense, is what can be termed the actual condensed representation; it is those building block a user is primarily interested in. The set of rules is used to restore the not included item sets and their interestingness heuristics. Note, that this set of rules may be stated implicitly, as in the case of closed and free item sets, or explicitly, as in the case of disjunction free item sets. With exception of $\sigma$-strong rules these are strict rules; they represent strong knowledge about a domain. The border set is necessary to be able to determine whether an item set is frequent. Knowing only this border set is sufficient to enumerate any frequent item set. It is thus conceptually similar to maximal frequent item sets (see Bayardo, Jr., 1998). As with the rule set, the border set is either given implicitly or explicitly.

Table 3.6 provides an overview on the building blocks of each discussed representation. It is sorted in order of publication, and as can be seen closed item sets are the root from which the discussed condensed representations have been derived. Free sets stem from the idea of representing the closed item set equivalence classes using their smallest elements rather than their largest. As a side effect this led to the introduction of border descriptions, because without those borders infrequent item sets would be falsely identified as belonging to an equivalence class of frequent item sets. Disjunction free sets, in turn, have been derived from 0-free sets by allowing a particular type of Boolean expression on the right hand side of the rules. Their introduction had added an explicitely represented rule set to the, still necessary, border description. Bold letters within the rules indicate those item sets that are kept as a candidate for the condensed representation.

From an expert's point of view the item sets of a condensed representation are the result of removing redundancies from the original set of item sets. The previously mentioned rule sets support this interpretation. By intuition, an item set is deemed redundant if its presence and properties are inferable from knowledge about the presence and properties of other item sets by application of the expert's domain knowledge. This way the inferred item sets are excludable from presentation without loss of meaning. Rule sets are a common representation of domain knowledge. Nonetheless, in addition, recalling his domain knowledge needs to be effortless for an expert, which implies that it must be widely known and obvious. The above mentioned rules hold within a given data set without exception; despite noise and an often poor data quality. Thus, these rules are so strong that there is a good chance that an expert already is aware of them.

### 3.4.5 Assessment

The afore-outlined condensed representations have received a varying level of attention in the scientific community, measured by means of published enhancements, extensions,

algorithms and applications. In this regard, closed item sets seem to have gained the highest maturity among the proposed representations, followed by 0-free item sets. Disjunction free sets only play a minor role, and so are many other condensed representations that have been proposed but are not further elaborated in this thesis.

Independent of their maturity, all condensed representations have their advantages and disadvantages, such that there is often no clear answer on which one is 'the best'. As explained at the beginning of Section 3.4, the objective of condensed representations is to describe the set of frequent item sets as *concise* and *lossless* as possible. Both requirements are conflicting such that in the ideal case a choice would have to be made between sacrificing the ability to precisely reconstruct the set of frequent item sets for the sake of a smaller representation, or strictly maintaining losslessness.

The following sections discuss closed, free and disjunction free item sets in the light of these requirements with the aim to identify one condensed representation that will serve as the reference for condensed representation to be developed in this chapter. It turns out that closed item sets are this reference representation.

#### Lossless

Closed, 0-free and disjunction free item sets are *strictly lossless* condensed representations. They enable to exactly restore for each given item set its particular interestingness heuristics, in these cases support. The restored collection of item sets and their support values is identical to the original collection that yielded the condensed representation. The disadvantages of an exact restoration are at least twofold. First, amongst useful information also noise and other unwanted artefacts in the data are being preserved. Second, the condensed representations are susceptible to a low data quality. One erroneous transaction can turn, for instance, an actually non-0-free item set into a 0-free one. As a result, the number of item sets in the condensed representation increases.

Since the chance that a data set contains erroneous transactions increases with its size, the problem becomes more apparent in large-scale, real-world data rather than in the small-scale, often synthetic data used in scientific publications. In fact, in many real-world data sets the number of closed, disjunction free respectively 0-free item sets will be almost the same as the one of frequent item sets—no reduction is achieved at all, the condensed representations miss the conciseness requirement further down. In Appendix A one real-world data set will be presented in which all frequent item sets are closed.

Strict losslessness results from the equivalence class view on condensed representations laid out in Section 3.4.4. To take closed and 0-free item sets as an example: Two item sets belong to the same equivalence class if and only if they are supported by the *identical* set of transactions, which implies their supports have to be equal. The use of the equality relation enforces the possibility of the support values' exact reproduction. From a theoretical standpoint, this approach has the appeal to provide for an algebraic interpretation of condensed representations, and indeed closed item sets are historically rooted in the theory of lattices and Galois connections (Pasquier et al., 1999).

From a practical standpoint, the idea is to sacrifice algebraic beauty for effectiveness on real data, and strict losslessness for the sake of conciseness. This leads to approaches that aim not to preserve noise in the condensed representation, and thus are *almost*

*lossless*. The concept of $\delta$-free item sets (Boulicaut et al., 2003), for $\delta > 0$, and a similar extension of closed item sets to *$\delta$-tolerant closed item sets* (Cheng et al., 2006) are a step into this direction. Both have two drawbacks. First, they require a parameter—$\delta$ in both cases—which describes how much noise is expected. Specifying such a *noise tolerance level* demands a degree of expert knowledge about the domain and the data collection process which is not always available. Second, in the case of $\delta$-free item sets the tolerance level is an absolute, support-independent number of erroneous transactions. Ideally, it should be a function of support such that for item sets with lower (absolute) support values less erroneous transactions are acceptable. The latter, nonetheless, holds for $\delta$-tolerant closed item sets were $\delta$ is a factor specifying the fraction of support that is tolerated to be attributable to erroneous transactions.

### Concise

Conciseness is understood as that "the size of the [condensed] representation is significantly smaller than the original set" (Bonchi and Lucchese, 2004). The emphasise is put on 'representation' and 'size'—despite this, it is left open what is measured and how. Section 3.4.4 identified three building blocks which in varying combinations belong to a condensed representation and are explicitly represented therein: the subset of the original collection of item sets, a set of rules and a border set. The question is, which blocks contribute towards measuring size and thus towards assessing conciseness? The problem becomes apparent when contrasting 0-free and disjunction free item sets. To determine whether an item set is kept within the condensed representation the approach of disjunction free item sets uses a generalised form of the criterion employed for 0-free item sets. As one result the set of disjunction-free item sets is smaller or equal than the set of 0-free item sets. At the same time more additional information need to be maintained: In addition to the negative border now also the set of disjunction rules is required to avoid their costly recalculation. Already the negative border by itself can be very large and sometimes even exceed the size of the corresponding collection of frequent item sets (Liu et al., 2008). If the size of each building block counts towards conciseness then it seems that decreasing the size of item sets by introducing more general or more complex criterions not always leads to a more concise condensed representation. In this case, closed item sets are favourable to 0-free and disjunctive item sets.

Conversely, condensed representations solve a presentation and not a storage problem. The vast number of frequent item sets entails a need for presenting only a meaningful subset without suppressing significant ones. The retrieval of support values or assertions about their significance for unpresented item sets is a later step, only carried out on demand. From this perspective, building blocks such as rule sets or borders will not be presented but only stored and queried. Having access to massive amounts of cheap storage nowadays is not a problem anymore. For this reason, they should not contribute towards the assessment of conciseness. In this regard, as opposed to the above, disjunction free item sets are favourable to 0-free item sets.

Another issue in assessing conciseness concerns the size of the item sets. For illustration, contrast the number of items contained in 0-free item sets with those contained in closed item sets. Both condensed representations describe the same equivalence classes of item sets; the ones formed by item sets appearing in the identical set of transactions (Bastide et al., 2000). 0-free item sets are the smallest elements and closed item sets the largest

elements of each class. 0-free item sets thus are smaller than closed item sets and may appear more concise.

However, as representatives for each equivalence class more than one free item set may be necessary, whereas always exactly one closed item set is sufficient (Bastide et al., 2000). Moreover, it is not guaranteed that 0-free item sets from the same equivalence classes are disjoint. Multiple 0-free sets may share items. Overall more items are needed to represent an equivalence class compared to taking the closed item set. This means, seen on the level of single items the smaller size of 0-free item sets provides no advantage for obtaining a concise representation.

Some authors have argued, the smaller size of 0-free item sets is more suitable classification purposes because they may generalise better (match unseen instances) (see Liu et al., 2008; Zimmermann, 2009). However, the equality relation used within the criterions for 0-free and closed item sets imposes such a strong condition on the data that whenever a 0-free item set matches an unseen instance, its closure (i.e. the corresponding closed item set) will tend to match too. On the other hand, free item sets are the worse choice for descriptive purposes, because it is hidden that multiple free item sets may describe the same set of transactions. This, however, can only be revealed by computing and comparing the corresponding closed item set. In fact, closed item sets are the better choice for descriptive purposes.

## 3.5 Condensed Representations and Time

A general shortcoming of all condensed representations is that they were not developed with the temporal dimension in mind. Even closed item sets, as the one with the highest maturity level and probably also the most widely used approach, does not incorporate time—although, as laid out in Chapter 1, almost every data set is time-stamped. As a result condensed representations do not account for redundancies imposed by the temporal dimension which would not only may have the potential to make condensed representations more concise but also have the appeal to seemingly integrate with the item set-based change mining approaches reviewed in Chapter 2. This section looks into two straightforward, simple ways of incorporating time into closed item sets and analyses their properties. Later on, they will serve as a baseline against which a proposed, more sophisticated approach for dealing with temporal redundancies is compared.

### 3.5.1 Time Periods as Independent Data Sets

Given a temporal sequence of data sets $D_1, \ldots, D_n$, closed item sets only take each data set independently into account. In fact, they were developed to be applied only to single data sets. For this reason, the first straightforward approach for incorporating time that will be discussed is to generalise the definition of closed item sets from a single data set, which corresponds to one time period, to a sequence of many by producing closed item sets independently for each $D_i$ and subsequently combining the results (see Böttcher et al., 2009).

On the one hand, from such a generalisation one would expect it to be straightforward in the sense that it resembles the original definition of closed item sets as well as possible. On the other hand, the generalisation should be sufficient for the purpose of change

mining: an item set should only be regarded as *non-closed over a sequence of time periods* if no important change information is lost that a user may be interested in.

As an example how change information can be lost consider an item set $X$ for which in one half of the periods there exists an item set $Y \supset X$ with $\text{supp}_i(X) = \text{supp}_i(Y)$ and in the other half an item set $Z \supset X$ with $\text{supp}_i(X) = \text{supp}_i(Z)$; only it is neither $Y \subset Z$ nor $Z \subset Y$. Should $X$ be regarded as closed or as non-closed over a sequence of time periods? If an item set with varying *reason for non-closedness* would be regarded as non-closed over a sequence of time periods and thus not presented to a user this change information is lost. Consequently, it is reasonable to regard such item sets as closed in the context of a generalisation of closed item sets towards sequences of time periods, even though it is non-closed in each time period.

Generally, two requirements have to be met by a temporal generalisation of closed item sets. In the first place, an item set $X$ should be non-closed over a sequence of time periods only if it is non-closed in all periods. Secondly, following the above discussion the underlying reason for non-closedness should be an invariant property.

A direct temporal generalisation of non-closed item sets which meets the requirements above is the following: An item set is *non-closed over a sequence of time periods* $T_i, i = 1, \ldots, n$ iff there exists an item set $Y \supset X$ such that for all periods $\text{supp}_i(X) = \text{supp}_i(Y)$ $i = 1, \ldots, n$. Equivalently, item sets which are *closed over a sequence of time periods* are defined as follows:

**Definition 3.5** (Closed over a Sequence)**.** *Let $D$ be a time-stamped data set and $\{T_1, \ldots, T_n\}$ a corresponding partition of the time axis into periods. An item set $X$ is closed over the sequence of time periods $\hat{T} := (T_1, \ldots, T_n)$ iff there exists no item set $Y \supset X$ such that for all $T_i$ : $\text{supp}_i(X) = \text{supp}_i(Y)$.*

In the following, the set of all item sets that are closed over a sequence is denoted by $\mathit{SeqClosed}(D, \hat{T})$.

The above definition means that $\mathit{SeqClosed}(D, \hat{T})$ contains all item sets which are closed in at least one period. Additionally, it also contains those item sets which are non-closed in each individual time period but for which their reason for non-closedness differs across periods. It should be noted, however, that the latter is a rare constellation as it has already been reported by authors in the field of incremental item set mining (see Chi et al., 2006).

An item set $X$ that is closed over a sequence of time periods $(T_1, \ldots, T_n)$ is *frequent* with respect to a support threshold $\sigma$ if it is frequent in all periods, this means if $\text{supp}_i(X) \geq \sigma$, $i = 1, \ldots, n$. The set of frequent item sets that are closed over a sequence will be represented by $\mathit{FreqSeqClosed}(D, \hat{T}, \sigma)$.

**Example 1 (continued).** Consider the data set $D$ shown in Table 3.1 as gathered in two different time periods: all transactions having a transaction identifier lower than 20 belong to period $T_1$ and all others to period $T_2$. Table 3.7 shows the frequent item sets that are closed over the sequence of time periods $\hat{T} = (T_1, T_2)$ contained in $D$ based on a support threshold of $\sigma_a = 2$. Note, that some item sets like *ct* are closed in $T_1$ and $T_2$, whereas others are only closed in one period, like *ac* which is non-closed in $T_1$.

In a time-stamped data set any time period uniquely determines a subset of it. The fol-

| $\text{supp}_a$ | | |
|---|---|---|
| $T_1$ | $T_2$ | *FreqSeqClosed* |
| 8 | 6 | c |
| 7 | 5 | ct |
| 6 | 4 | ac, cw |
| 6 | 3 | act |
| 5 | 4 | acw |
| 5 | 3 | actw |
| 5 | 2 | cd |
| 3 | 2 | cdw, acd |
| 2 | 2 | acdw |

Table 3.7: Item sets contained in $D$ which are closed over the sequence $(T_1, T_2)$ and are frequent based on $\sigma_a = 2$

lowing theorem demonstrates that this subset relation also transfers to the corresponding sets of closed item sets.

**Theorem 3.1.** *Let $D$ be a data set and $D' \subseteq D$, then it is $Closed(D') \subseteq Closed(D)$.*

The following theorem establishes a link between closed item sets and those item sets which are closed over a sequence.

**Theorem 3.2.** *Given a time-stamped data set $D$ and a corresponding partition of the time axis into periods $\hat{T} := \{T_1, \ldots, T_n\}$, the set of closed item sets contained in $D$ equals the set of item sets that are closed over the sequence $\hat{T}$, in other words $SeqClosed(D, \hat{T}) = Closed(D)$.*

Theorem 3.2 expresses that incorporating time into a condensed representation by producing closed item sets individually for each time period and then combining the produced results leads to the same set of closed item sets that is found in the original un-split data. Indeed, the result is the same as if time would not be considered at all. The representation of item sets that are closed over a sequence thus misses the initially set goal of obtaining a more concise condensed representation by incorporating time.

Still, it should be noticed that Theorem 3.2 has at least two practical implications. First of all, to develop a more concise condensed representation by incorporating time it is desirable to have an established condensed representation with which it can be compared, both theoretically and experimentally. Although closed item sets are probably the most widely used condensed representation they cannot directly be used for this purpose because they were not developed with the temporal dimension in mind. Definition 3.5 together with Theorem 3.2 put closed item sets into a temporal context and thus provide a basis for comparisons.

Secondly, Theorem 3.2 indicates that the problem of producing closed item sets for a given (not necessarily time stamped) data set can be solved by splitting it into smaller sub-problems that correspond to smaller data sets and subsequently merging the obtained solutions. This, in turn, matches modern paradigms for parallel processing of large data sets like *MapReduce* (Dean and Ghemawat, 2008). This idea, however, will not be further investigated in this thesis.

### 3.5.2 Time Periods as Items

A second straightforward approach for incorporating time is to encode the time period from which a transaction originates as an extra item. To each transaction such an item is added and a condensed representation obtained from the resulting, now temporally augmented transaction set. Formally, assume $L$ being a set of items and $D = D_1 \cup \ldots D_n$ a data set with $D_i$ being the subset of transactions $I \in D_i$, $I \subseteq L$ belonging to time period $T_i$.

**Definition 3.6** (Temporally Augmented Transaction Set)**.** *Let $t_i \notin L$ be an (artificial) item that uniquely represents $T_i$, and $L^T := L \cup \{t_1, \ldots, t_n\}$. Construct a data set $D_i^T$ from $D_i$ such that for all $I \in D_i$ there exists a $I \cup \{t_i\} \in D_i^T$, and for all $I \in D_i^T$ there exists a $I \setminus \{t_i\} \in D_i$. Then, $D^T = D_1^T \cup \ldots D_n^T$ is the temporally augmented transaction set to $D_i$.*

The following theorems establish a link between the closed item sets obtained from $D$ and $D^T$ (resp. $D_i$ and $D_i^T$). Thereby $Yt_i$ denotes the item set that results from adding $t_i$ to $Y$.

**Theorem 3.3.** *Let $X$ be an item set and $Y \in Closed(D_i)$ its closure in time period $T_i$. If $\text{supp}_i(X) > 0$ then $Yt_i \in Closed(D^T)$ and $\text{supp}_i(X) = \text{supp}^T(Yt_i) / \text{supp}^T(t_i)$ .*

Considering an item set $X \in Closed(D)$ this theorem states that for each time period $T_i$ in which $X$ exists a closed item set $Yt_i$, $X \subseteq Y$ will be produced from $D^T$. This means, the size of $Closed(D^T)$ will be larger than $Closed(D)$; in the worst case by at least a factor equal to the number of time periods. The next theorem shows that the intersection of $Closed(D)$ and $Closed(D^T)$, if non-empty, consists of those elements of $Closed(D)$ which are present in at least two periods.

**Theorem 3.4.** *If $X \in Closed(D)$ and there exist $i \neq j$ such that $\text{supp}_i(X) > 0$ and $\text{supp}_j(X) > 0$ then it follows $X \in Closed(D^T)$.*

It is not unexpected that adding further items, which represent time periods, does not reduce the collection of item sets. The obtained theorems nonetheless are significant because they devise a way to obtain the support histories of item sets. Support histories capture the temporal development of item sets and are thus the foundation of change mining approaches, as laid out in Section 2.3. Section 3.3 introduced their formal definition and notation.

In brief, support histories are obtained as follows. Having a time stamped data set $D$ and a partition of time, construct the data set $D^T$ and produce its closed item sets $Closed(D^T)$. According to Theorem 3.4, each item set $X \in Closed(D)$ which is present in all time periods is element of $Closed(D^T)$, and according to Theorem 3.3 also is $Xt_i$. Utilising $\text{supp}_i(X) = \text{supp}(Xt_i) / \text{supp}(t_i)$ yields the history of each closed item set. From the histories of all closed item sets the histories of the remaining item sets are retrievable (see Theorem 3.2).

The approach pointed out in the literature for obtaining item set histories involves the discovery of item sets—*not* closed item sets—individually for each time period (see, e.g., Baron and Spiliopoulou, 2001; Liu et al., 2001b). After each discovery run, each produced item set is compared to those discovered in previous periods and, in case of a match, its history extended. To facilitate an efficient matchmaking and to address a need

for long-term storage of item sets, change mining systems sometimes employ database technology (Baron and Spiliopoulou, 2001; Böttcher et al., 2006a).

The approach that encodes periods as items is by a factor of $|L|$ more time efficient than the latter 'matchmaking' approach in obtaining item sets' histories from a time-stamped data set. The following estimates for both approaches' time complexity are based on the assumption that the number of items $|L|$ is equal in all periods. Further, both estimates use the theoretical result that enumerating all frequent item sets for an arbitrary support threshold has a time complexity which is exponential in the number of items. In fact, the problem is known to be NP-hard (Gunopulos et al., 2003; Yang, 2004).

The matchmaking approach enumerates for each of the $n$ time periods the item sets discovered therein. In the worst case, for each time period this takes $O(2^{|L|})$, and thus overall $O(n \cdot 2^{|L|})$. For the first period this effort includes the construction of a prefix tree—the prevalently used data structure to represent item sets (Cheng et al., 2008, p. 4). Item sets of each subsequent periods are matched against the item sets stored in the prefix tree along with their so-far obtained support histories. The effort to find an item set within a prefix tree is bounded by the length of the longest possible item set which is $|L|$. Since this lookup needs to be done for each item set within each period, the time complexity for the matchmaking approach therefore is $O(n \cdot |L| \cdot 2^{|L|})$.

The approach that encodes time periods as items introduces $n$ more items. However, due to the restriction that periods are disjoint these items have the property that no more than one of them is contained within a frequent item set produced from $D^T$. This means, for each frequent item set already enumerated from $D$, a maximum of $n$ additional item sets will be enumerated when considering $D^T$. The approach's time complexity therefore is $O(n \cdot 2^{|L|})$.

It is noteworthy that the additional effort needed for the matchmaking approach solely depends on the chosen data structure for searching item sets. If the search would be accomplishable in $O(1)$, for instance using a hash table and a perfect hash function, then the time complexity is the same for both approaches. Nonetheless, the space complexity for the hash table-based approach will be worse because hash keys need to be stored for each item set. In addition to this gain in time-efficiency, the approach of encoding time periods as items offers a range of practical benefits, which will be discussed in Section 3.7.2.

## 3.6 Temporal Redundancy

The approaches discussed in the previous section devise two possibilities of incorporating time into closed item sets; yet they do not lead to any further reduction in the number of item sets in comparison to the condensed representations discussed in Section 3.4. The reason is they exploit no type of redundancy that is characteristic for a temporal context and whose presence only is determinable when chronological information about the data set's records is available.

There may exist a plethora of suitable types of *temporal redundancy* that differ in their occurrence, their meaningfulness but also their 'backward compatibility' to non-temporal types of redundancy as they are used in classical condensed representations, for instance closed or free item sets. The ultimate choice is characterised by the capability of being

the basis of a condensed representation. As such, it must contribute towards fulfilling the requirements thereon that are identified in the Problem Statement in Section 3.2. This section discusses one type of temporal redundancy which is present in many data sets. It puts special emphasis on demonstrating that this type is meaningful to experts and on showing how it serves as a main building block for a condensed representation that accounts for the temporal dimension.

### 3.6.1 Invariance and Uninterestingness

When elaborating on change mining in Chapter 2 it was pinpointed that properties of a domain which change are inherently interesting because they serve as an indicator for a required intervening action, for instance, to rectify a problem or to exploit an opportunity (Chakrabarti et al., 1998). In contrast, *invariant properties* are of less interest because they are almost always known and generally not judged as hinting at a serious problem (Berger, 2005).

**Example 1 (continued).** Consider the item sets in Table 3.7 (p. 46) and their support values in period $T_1$ and $T_2$. A closer look reveals two invariant properties of the same type. The ratio between the support of item set $ac$ and its superset $acd$ does not change, it is 2/3 in each period. The same applies to $cw$ and $cdw$, here the ratio also is 2/3. In other words, the fraction of transactions that contain $d$ additionally to $ac$, respectively $d$ additionally to $cw$ is invariant. This can be expressed in terms of uncertain rules $cw \Rightarrow d$ and $ac \Rightarrow d$ whose uncertainty is quantified by their confidence. The uncertainty factor is constant over time and thus represents an invariant domain property. For $i = 1$ and $i = 2$ it is $\mathrm{conf}_i(cw \Rightarrow d) = 2/3$ and $\mathrm{conf}_i(ac \Rightarrow d) = 2/3$.

In the example, the shape of the support history of $cdw$ and the one of $cw$, apart from scaling, are equal. It is $\mathrm{supp}_i(cwd) = 2/3\,\mathrm{supp}_i(cw)$. The same holds for $acd$ and $ac$. Knowing the above invariant property an expert is able to infer the shape of one item set's history from that of the other. The relevance of a particular item set is primarily dictated by its qualitative change over time and not by its actual support value (Agrawal and Psaila, 1995; Chakrabarti et al., 1998). Taking the aforesaid into account an expert yields from one item set, like $cwd$, all the necessary information to decide whether the other item set, like $cw$, is relevant. The reason is that invariant properties of a domain are known; for both item sets it thus can be stated that they are *temporally redundant* with respect to each other.

**Example 2.** Consider a data set which was collected from information about road accidents with casualties and that includes details about the accident such as data, time, location, weather conditions and causality severity.[1] The data is split into 10 smaller data sets corresponding to the years 2000 to 2009 and item sets are discovered for each. The item set $X$ represents accidents that happened at bad lighting conditions at a T, Y or staggered junction, while the item set $XY$ represents the subset of those with two involved cars.[2] Figure 3.4 shows the support history of both item sets. As can be seen, the change of $X$ is qualitatively the same apart from noise as the one of

[1] Further details on this data are found in Appendix A.

[2] Taking the attributes and attribute values from the original data, the item set $X$ consists of the items $A1.16 = 3$ and $A1.21 = 2$ and the item set $Y$ of $A1.5 = 2$.

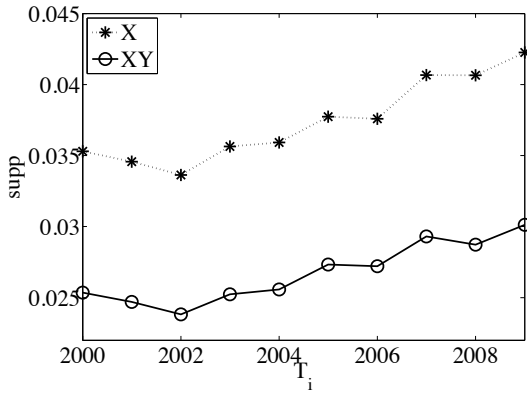



Figure 3.4: Histories of the item sets $XY$ and $X$ from Example 2

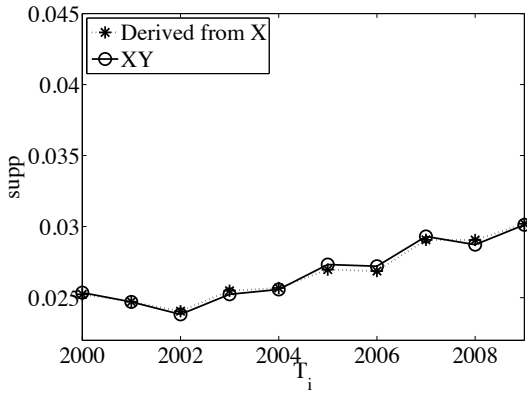



Figure 3.5: Approximated history of $XY$ using the history of $X$

$XY$. Both histories only differ by a scaling factor which is an invariant property of the domain.

In this example the invariant property is the ratio of accidents that involve two cars relative to all accidents that happen on junctions under poor visibility conditions. This property did not change over a period of 10 years, such that with high probability it is well-known to experts and of no interest, as claimed at the beginning of this section. But also for non-experts this invariant is of no surprise. Presumably many accidents on crossings under poor visibility are due to the inattentiveness of the driver. A driver will be inattentive in general, and not only towards specific other road users. There is little to no reason to assume that car drivers would develop a preference to overlook another car, but not pedestrians or road sign poles, for instance.

Among the many existing types of temporal redundancy the one laid out before has the appeal that a condensed representation build thereupon aligns well with the type of redundancy underlying closed and 0-free item sets. As mentioned earlier it is based on uncertain rules $X \Rightarrow Y$ whose uncertainty is measured through $\mathrm{conf}_i(X, Y)$ and is constant over time. In comparison, closed and 0-free item sets are based on rules $X \Rightarrow Y$ which are certain; this means it is $\mathrm{conf}(X, Y) = 1$. It appears as if the former can be interpreted as a generalisation of the latter. This aspect is elaborated in Section 3.7.1.

The discussed type of temporal redundancy is very common in practice. Many of the changes observed in item set histories are simply the effect of a combination of changes in other item set histories with invariant domain properties (see Liu et al., 2001a). This leads to the situation depicted in Figure 3.4: Histories of item sets have qualitatively the same shape, thus provide the same information to an expert and have the same relevance. Having a collection of item sets that are linked through generalisation and specialisation relationships and have histories with equal shape, only one representative needs to be presented. From the perspective of condensed representations this opens the opportunity to achieve a higher degree of conciseness. From the perspective of change mining this eradicates the otherwise tedious and time eating task of identifying the fundamental changes that take place within a domain.

### 3.6.2 Definition and Probabilistic Interpretation

Figure 3.4 reveals that the type of temporal redundancy motivated in the previous section manifests itself by a (simple) functional dependency between the history of $X$ and $XY$. Other functional dependencies exist, but may be less prevalent or too complex to be practically meaningful. Böttcher et al. (2005) provide an analysis of functional dependencies other than the aforementioned one. Since these dependencies are functions of time, item sets will be called *temporally dependent*, if their histories are linked in such a way.

**Definition 3.7** (Temporally Dependent)**.** *Let $X$, $X_1, X_2 \dots X_p$ be item sets with $X \supset X_i$ for all $i$ and $p > 0$. Let* supp *the support,* $\text{supp}|_X(T) := \text{supp}(X,T)$ *and* $\text{supp}|_{X_i}(T) := \text{supp}(X_i,T)$ *its functions over time and $\mathcal{M} := \{g : \hat{T} \longrightarrow [0,1]\}$ be the set of functions from time into the unity interval. The item set $X$ is temporally dependent iff a function $f : \mathcal{M}^p \longrightarrow \mathcal{M}$ exists such that for all $T \in \{T_1, \dots, T_n\}$*

$$\text{supp}|_X(T) = f(supp|_{X_1}, supp|_{X_2}, \dots, supp|_{X_p})(T)$$

The main idea behind the above definition is that an item set is temporally dependent, if its history can be constructed as a mapping of the histories of (a subset) of its supersets. To compute the value $\text{supp}(X,T)$ the values $\text{supp}(X_i,T)$ are thereby considered. The definition above does not allow for a pointwise definition of $f$ on just the $T \in \{T_1, \dots, T_n\}$, but instead states a general relationship between the support values independent from the point in time. It can therefore also be used to predict the value of $\text{supp}(X)$ given future values of the $\text{supp}(X_i)$.

Based on this definition the type of temporal redundancy discussed in the previous section can be stated $\text{supp}|_X = f(\text{supp}|_{X_1}) = \varepsilon\ \text{supp}|_{X_1}$, i.e. the support history of an item set $X$ can be obtained by multiplying the support history of a subset $X_1$ with a constant $\varepsilon$. If this kind of relationship is present between the history of an item set and the history of one of its subsets it is said that the first is *temporally derivable* from the latter. Formally:

**Definition 3.8** (Temporally Derivable)**.** *Let $X, Z \neq \emptyset$, $Z \supset X$ be item sets and* $(\text{supp}_1(X), \dots, \text{supp}_n(X))$*, respectively* $(\text{supp}_1(Z), \dots, \text{supp}_n(Z))$ *their support histories. The item set $Z$ is temporally derivable with regard to $X$, denoted $X \hookrightarrow Z$, iff there exists a constant $\varepsilon$, $0 < \varepsilon \leq 1$ such that* $\text{supp}_i(Z) = \varepsilon\,\text{supp}_i(X),\ i = 1, \dots, n$.

Explained intuitively, the definition states that an item set is temporally derivable if it has the same shape as the history of a subset apart from a scaling factor $\epsilon$. To emphasise the scaling factor $\epsilon$ sometimes the notation $X \overset{\epsilon}{\hookrightarrow} Y$ will be used. Assuming $XY = Z$ the criterion $\text{supp}_i(XY) = \epsilon\,\text{supp}_i(X), i = 1, \dots, n$ used within the definition can be rewritten as $\epsilon = \text{supp}_i(XY)/\,\text{supp}_i(X) = \text{conf}(X,Y)$. This means, the fraction of transactions containing $Y$ additionally to $X$ constantly grows in the same proportion as $X$. In other words, the confidence (represented by the scaling factor $\epsilon$) of the rule $X \Rightarrow Y$ does not change over time thus is an invariant property of the domain.

Figures 3.4 and 3.5 show an example of a temporally derivable item set taken from the road accident data also used for other experiments (see Appendix A). The meaning of the referred item set is detailed in Example 2 (p. 49). Figure 3.4 shows the support

histories of the less specific item set at the top and the more specific item set below, both over ten years. The shape of the two histories is obviously very similar and it turns out that the history of the more specific item set $XY$ can approximately be determined using the more general one $X$ by applying a scaling factor. Vice versa, the history of the less specific item set can be determined from the more specific in the same way. As shown in Figure 3.5 the reconstruction is not exact. The reason for this is noise.

Due to noise two histories, which for an expert would appear being temporally derivable from each other, are not being detected as such when strictly adhering to Definition 3.8 and utilising a test for equality. This is a serious practical limitation because far fewer temporally derivable item sets will be discovered within data than are actually contained. For this reason, a noise-tolerant approach for testing for temporal derivability is desirable and will eventually be described in the following section. One step into this direction is to look at temporal derivability from a probabilistic and information theoretic perspective.

Suppose a transaction $I$ being the result of a random process. Then, for each transaction it can be measured whether it includes a given item set $X$ and the result of this measurement can be described by a random variable $\mathcal{X}$ with $\mathrm{dom}(\mathcal{X}) = \{1, 0\}$. Thereby, $\mathcal{X} = 1$ indicates that inclusion holds, i.e. $X \subseteq I$. Further, a different measurement that can be applied to the random process regards the time period during which a transaction was generated. The result can be described by a random variable $\mathcal{T}$ with domain $\mathrm{dom}(\mathcal{T}) := \hat{T} = \{T_1, \ldots, T_n\}$ assuming that the random process' start and end time are known and that a complete partitioning of this time span into a finite number of periods already exists.

In practise the probability distributions $P(\mathcal{X})$ and $P(\mathcal{T})$ will almost never be exactly known. They can, however, be approximated. The support of an item set measures the relative frequency with which the item set is contained in a sample of transactions. From a probabilistic point of view a relative frequency provides an estimate for the probability that a certain event occurs and according to the law of large numbers this estimate becomes more accurate the larger the sample size gets. Because the support $\mathrm{supp}(X)$ measures the relative frequency of the event $\mathcal{X} = 1$ it is therefore $P(\mathcal{X} = 1) \approx \mathrm{supp}(X)$. From the two-valued domain of $\mathcal{X}$ it follows $P(\mathcal{X} = 0) = 1 - P(\mathcal{X} = 1) \approx 1 - \mathrm{supp}(X)$. As concerns $P(\mathcal{T})$, its values $P(\mathcal{T} = T_i)$ are approximated by the relation between the number of transactions that are from time period $T_i$ and the overall number of transactions, that is $P(\mathcal{T} = T_i) \approx |D_i|/|D|$.

Having the support history $(\mathrm{supp}_1(X), \ldots \mathrm{supp}_n(X))$ of an item set $X$ and following a similar line of argumentation as above it thus provides an estimate for the conditional probabilities $P(\mathcal{X} = 1 \,|\, \mathcal{T} = T_1), \ldots, P(\mathcal{X} = 1 \,|\, \mathcal{T} = T_n)$. Utilising that $P(\mathcal{X} = 0 \,|\, \mathcal{T} = T_i) = 1 - P(\mathcal{X} = 1 \,|\, \mathcal{T} = T_i)$ the conditional probability distribution $P(\mathcal{X} \,|\, \mathcal{T})$ follows immediately. This means, a support history can be equivalently represented as a conditional probability distribution, which again allows for a probabilistic reformulation of temporal derivability.

**Theorem 3.5.** *The item set $XY$ is temporally derivable from the item set $X$ if and only if $\forall\, T_i \in \hat{T} \,:\, P(\mathcal{Y} \,|\, \mathcal{X} = 1,\ \mathcal{T} = T_i) = P(\mathcal{Y} \,|\, \mathcal{X} = 1)$.*

This means, if it is known that a transaction contains $X$ and then learning about the time period from which it stems does not change the knowledge on whether the transaction also contains the item set $Y$. But if any knowledge about time does not tell anything

new about a certain item set occurrence, then the respective part of the domain which this item set models should be deemed stable.

**Example 1 (continued).** In the example on page 49 the item set $acd$ was identified temporally derivable from the item set $ac$. Let $X := ac$ and $Y := d$, for each $T_i$ the probability that a transaction which contains $ac$ also contains $d$ can be estimated based on the support values given in Table 3.7 (p. 46) $P(\mathcal{Y} = 1 \,|\mathcal{X} = 1,\ \mathcal{T} = T_1) \approx \operatorname{supp}_1(acd)/\operatorname{supp}_1(ac) = 0.5$, and $P(\mathcal{Y} = 1 \,|\mathcal{X} = 1,\ \mathcal{T} = T_2) \approx \operatorname{supp}_2(acd)/\operatorname{supp}_2(ac) = 0.5$. According to Theorem 3.5 these probabilities should be equal to $P(\mathcal{Y} = 1 \,|\mathcal{X} = 1)$. Indeed, it is $P(\mathcal{Y} = 1 \,|\mathcal{X} = 1) \approx \operatorname{supp}(acd)/\operatorname{supp}(ac) = 0.5$ (cp. Table 3.2, p. 35).

In the extreme case that for any item set and for any number of time periods $\mathcal{Y}$ is probabilistically independent of $\mathcal{T}$, the 'classical' assumption in data mining that a domain is stable would be met. A (colloquially expressed) definition of 'stable' therefore could be: stability is in place when adding time to the analysis does not alter any prior knowledge.[3] On the other extreme, no temporally derivable item set is detected. This will only occur if change is omnipresent and no stable parts of the domain exist; every item set occurrence then depends on time, regardless whether, or not, the probability is conditioned on the occurrence of another item set.

Both extremes will almost never occur. It can be expected that some aspects of a domain will change (and thus are probabilistically dependent on time) while others remain stable (and thus are probabilistically independent of time). Each item set describes knowledge about such an aspect of a domain. The stronger an item set depends on time the more effective it gets to utilise change—implicitly represented through a time axis—for the enhancement of knowledge, as it is advocated in this thesis. Conversely, if knowledge is probabilistically independent of time then its augmentation with information about change is meaningless and ineffective. The concept of temporal derivability thus provides a way of separating those parts of knowledge for which change is worth to be utilised from those for which it is not.

### 3.6.3 Information Theoretic Assessment

Theorem 3.5 only provides a probabilistic criterion on whether two item sets are temporally derivable based on the independence of an item set's occurrence and time. It does neither establish a way of testing this criterion nor does it yield a statement on the strength of the relation in case of dependence. Two questions are of interest: Is the occurrence of the item set $Y$ independent of time? And, second, how much information does time contain about the occurrence of $Y$? Thereby it may be already known that a transaction contains an (not necessarily non-empty) item set $X$. Both questions are related, an answer of zero for the second should imply a positive answer for the first question, and vice versa. Thus, both should be answered using the same tool set.

Information theory provides such a tool set; an introduction to its basic concepts provides Appendix C.1. Information theory introduces the concept of entropy $H(\mathcal{A})$ which is the average number of bits needed to describe a random variable $\mathcal{A}$. It is thus a measure of its information content. Having two random variables $\mathcal{A}$ and $\mathcal{B}$ the concept of *mutual*

[3] As the experiments in Appendix A show this is not the case by far. This, in turn, supports the assumption that change is ubiquitous and that the stability assumption is often wrong.

*information* $I(\mathcal{A}, \mathcal{B})$ describes how much information contains one about the other. Mutual information also is a measure of the inefficiency of assuming that the distribution is $P(\mathcal{A} = a)P(\mathcal{B} = b)$ when the true distribution is $P(\mathcal{A} = a, \mathcal{B} = b)$. The more the assumption of independence is violated the more bits are necessary to describe $\mathcal{A}$ when having knowledge about $\mathcal{B}$. When wrongfully assuming independence one would need on average $I(\mathcal{A}, \mathcal{B})$ extra bits (or $H(\mathcal{A} \mid \mathcal{B}) + I(\mathcal{A}, \mathcal{B})$ overall). However, if they are independent the following theorem can be stated which is, for instance, proven in Cover and Thomas (2006, p. 28f; Theorem 2.6.3 and Corollary).

**Theorem 3.6.** *For any two random variables, $\mathcal{A}$ and $\mathcal{B}$, it is $I(\mathcal{A}, \mathcal{B}) = 0$ if and only if $\mathcal{A}$ and $\mathcal{B}$ are probabilistically independent.*

Returning to the initial problem of determining how much information time $\mathcal{T}$ contains about an item set occurrence $\mathcal{Y}$, the mutual information between $\mathcal{Y}$ and $\mathcal{T}$ is denoted

$$I_X(\mathcal{Y}, \mathcal{T}) = \sum_{t \in \mathrm{dom}(\mathcal{T})} \sum_{y \in \mathrm{dom}(\mathcal{Y})} P(\mathcal{Y} = y, \mathcal{T} = t \mid \mathcal{X} = 1) \cdot \log \frac{P(\mathcal{Y} = y, \mathcal{T} = t \mid \mathcal{X} = 1)}{P(\mathcal{Y} = y \mid \mathcal{X} = 1)P(\mathcal{T} = t \mid \mathcal{X} = 1)} \quad (3.4)$$

The subscript $X$ emphasise that the probabilities are conditioned on $\mathcal{X} = 1$, that is on having prior knowledge that a transaction already contains the item set $X$ (cp. Theorem 3.5).

In this context the utilisation of mutual information is appropriate and meaningful. As has been pointed out in Section 3.6.2 temporal derivability is associated with the concept of stability, and manifests itself in the probabilistic independence of item set occurrence $\mathcal{Y}$ and time $\mathcal{T}$ (see Theorem 3.5). Taking the aforesaid into account, $I_X(\mathcal{Y}, \mathcal{T})$ thus fulfils three purposes. First and foremost, it measures the information that time yields about a particular item set occurrence. The higher its value the easier it gets to guess for a transaction with known time period whether it contains this item set. Secondly, it quantifies in bits the inefficiency of wrongfully assuming stability when describing $\mathcal{Y}$ (over time). The more the stability assumption is violated, the more extra bits need to be spend. Thirdly, it serves as another criterion for testing temporal derivability, as follows immediately from combining Theorem 3.5 and Theorem 3.6.

**Corollary 3.1.** *The item set $XY$ is temporally derivable from the item set $X$ if and only if $I_X(\mathcal{Y}, \mathcal{T}) = 0$.*

The meaningfulness of mutual information in combination with Corollary 3.7 renders it a good starting point for testing the condition in Theorem 3.5 and thus determining whether temporal derivability holds for given item sets $X$ and $XY$. Yet, there is a practical pitfall here. The mutual information $I_X(\mathcal{Y}, \mathcal{T})$ depends on the joint probability distribution $P(\mathcal{Y}, \mathcal{T} \mid \mathcal{X} = 1)$ which almost always will be unknown. Nevertheless, as discussed in Section 3.6.2 the probabilities can be estimated by support.

For notational convenience, consider the required (absolute) support histories to be

| | $T_1$ | ... | $T_j$ | ... | $T_n$ | $\sum$ |
|---|---|---|---|---|---|---|
| # supp. | $o_{11}$ | ... | $o_{1j}$ | ... | $o_{1n}$ | $o_{1\cdot}$ |
| # not supp. | $o_{21}$ | ... | $o_{2j}$ | ... | $o_{2n}$ | $o_{2\cdot}$ |
| $\sum$ | $o_{\cdot 1}$ | ... | $o_{\cdot 1}$ | ... | $o_{\cdot n}$ | $o_{\cdot\cdot}$ |

Table 3.8: $2 \times n$ contingency table for support histories

organised in a $2 \times n$ contingency table as shown in Table 3.8 using the abbreviations:

$$o_{1j} := |D_j| \ \mathrm{supp}_j(XY) \qquad o_{2j} := |D_j| \ (\mathrm{supp}_j(X) - \mathrm{supp}_j(XY))$$

$$o_{1\cdot} := \sum_{j=1}^{n} o_{1j} = |D| \ \mathrm{supp}(XY) \qquad o_{2\cdot} := \sum_{j=1}^{n} o_{2j} = |D| \ (\mathrm{supp}(X) - \mathrm{supp}(XY))$$

$$o_{\cdot j} := o_{1j} + o_{2j} = |D| \ \mathrm{supp}_j(X) \qquad o_{\cdot\cdot} := \sum_{j=1}^{n} o_{\cdot j} = \sum_{i=1}^{2} o_{i\cdot} = |D| \ \mathrm{supp}(X)$$

The first row contains for each data set $D_j$ the absolute number of transactions which support $X$ and $Y$. The second row contains for each $D_j$ the absolute number of transactions which do support $X$ but do *not* support $Y$. These numbers are called *observed frequencies* $o_{ij}$, $i = 1, 2$ ; $j = 1, \ldots, n$. Obviously, it is

$$o_{1j}/o_{\cdot\cdot} \approx P(\mathcal{Y} = 1, \mathcal{T} = T_j \,|\, \mathcal{X} = 1) \qquad o_{2j}/o_{\cdot\cdot} \approx P(\mathcal{Y} = 0, \mathcal{T} = T_j \,|\, \mathcal{X} = 1)$$
$$o_{1\cdot}/o_{\cdot\cdot} \approx P(\mathcal{Y} = 1 \,|\, \mathcal{X} = 1) \qquad o_{2\cdot}/o_{\cdot\cdot} \approx P(\mathcal{Y} = 0 \,|\, \mathcal{X} = 1)$$
$$o_{\cdot j}/o_{\cdot\cdot} \approx P(\mathcal{T} = T_j \,|\, \mathcal{X} = 1)$$

Substitution in (3.4) followed by some basic transformations yields

$$\hat{I}_X(\mathcal{Y}, \mathcal{T}) := \frac{1}{o_{\cdot\cdot}} \sum_{i=1}^{2} \sum_{j=1}^{n} o_{ij} \log \frac{o_{\cdot\cdot} o_{ij}}{o_{i\cdot} o_{\cdot j}} \tag{3.5}$$

Note that for any given data set and random variables such an estimate can be obtained by substituting in the definition of mutual information the probabilities with their relative frequency counts (see Appendix C.1). In either way the resulting mutual information $\hat{I}$ will only be an estimate of the exact mutual information $I$, and it is clear that due to noise and sampling effects $\hat{I}$ can only be zero in exceptional cases. Because $\hat{I}$ is based on the outcomes of a random process (represented by a data set $D$), it is a random variable itself. Kullback (1968, p. 155ff) obtained and analysed its probability distribution, albeit Wilks (1935, 1938) achieved an equivalent result while researching the probability distribution of likelihood ratios from which the following statement can be easily derived.

**Theorem 3.7.** *Given a data set $D$, if the hypothesis that $\mathcal{A}$ and $\mathcal{B}$ are probabilistically independent is true, then the statistic $G^2 := 2|D| \log(e) \ \hat{I}(\mathcal{A}, \mathcal{B})$ approximates to a $\chi^2$-distribution with $(r-1)(s-1)$ degrees of freedom, where $r = |\mathrm{dom}(\mathcal{A})|$ and $s = |\mathrm{dom}(\mathcal{B})|$.*

Combining Theorem 3.5 and Theorem 3.7 allows to statistically test the following hypothesis $H_0$ against its alternative $H_1$:

$H_0$ : $XY$ is temporally derivable from $X$
$H_1$ : $XY$ is **not** temporally derivable from $X$

| | Time Period | | | | |
|---|---|---|---|---|---|
| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ |
| $\lvert D_j \rvert$ | 233729 | 229014 | 221751 | 214030 | 207410 |
| $\text{supp}_j(X)$ | 0.035293 | 0.034574 | 0.033627 | 0.035644 | 0.035919 |
| $\text{supp}_j(XY)$ | 0.025345 | 0.024692 | 0.023824 | 0.025239 | 0.025577 |
| | $T_6$ | $T_7$ | $T_8$ | $T_9$ | $T_{10}$ |
| $\lvert D_j \rvert$ | 198735 | 189161 | 182115 | 170591 | 163554 |
| $\text{supp}_j(X)$ | 0.037738 | 0.037587 | 0.040672 | 0.040646 | 0.042255 |
| $\text{supp}_j(XY)$ | 0.027322 | 0.027204 | 0.029311 | 0.028723 | 0.030130 |

Table 3.9: Data set sizes and support histories for Example 2

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ | $T_{10}$ | $\sum$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $o_{1j}$ | 5924 | 5655 | 5283 | 5402 | 5305 | 5430 | 5146 | 5338 | 4900 | 4928 | 53311 |
| $o_{2j}$ | 2325 | 2263 | 2174 | 2227 | 2145 | 2070 | 1964 | 2069 | 2034 | 1983 | 21254 |
| $\sum$ | 8249 | 7918 | 7457 | 7629 | 7450 | 7500 | 7110 | 7407 | 6934 | 6911 | 74565 |

Table 3.10: $2 \times 10$ contingency table for Example 2

The test statistic is defined by (3.5). If the null hypothesis holds it follows a $\chi^2$ distribution with $(2-1)(n-1)$ degrees of freedom. The null hypothesis $H_0$ is rejected at significance level $\alpha$, if the test statistic exceeds the threshold value $\chi^2_{1-\alpha}$ of the $\chi^2$ distribution with $(n-1)$ degrees of freedom.

The testing procedure is illustrated in the following example.

**Example 2 (continued).** Table 3.9 shows the actual values of the two histories that have previously been depicted in Figure 3.4 on Page 50 and Table 3.10 the resulting contingency table. Applying (3.5) leads to the test statistic $\hat{I} = 13.4$. If the null hypothesis $H_0$, that $XY$ is temporally derivable from $X$, holds the test statistic $\hat{I}$ is $\chi^2$-distributed with $(2-1)(10-1) = 9$ degrees of freedom. At significance level $\alpha = 0.05$ it is $\chi^2_{0.95} = 16.91 > 13.4 = \hat{I}$. Thus the null hypothesis cannot be rejected. By statistical means $XY$ is temporally derivable from $X$.

To carry out a statistical test the proposition which has to be shown is generally formulated as the alternative $H_1$, whereas the null hypothesis $H_0$ is merely formulated to be rejected. This allows to control the probability that the test rejects $H_0$, although it is in fact correct, by the significance level. The method above connects the decision for temporal derivability to the non-rejection of the null hypothesis. This means, the significance level does *not* control the probability that an item set is wrongly identified as temporally derivable, but the probability that an item set is identified by the test as being non-temporally derivable, although in fact it is.

The above choice of the null hypothesis reflects the 'conservative' position of domain stability. The test thus is optimal in settings where this position has to be disproved. This is, for instance, the case when all those item sets are of interest that have a unique history in the sense that there exists no more general, or likewise no more specific item set, whose history has the same shape. Clearly, such are non-temporally derivable item sets which are represented by the alternative hypothesis.

If instead the focus is on identifying stable aspects of a domain which is equivalent to

discovering temporally derivable item sets, the above choice of the hypothesis is suboptimal. The probability that an item set is wrongly identified as stable might possibly increase by lowering the significance level. However, the theoretical foundations of statistical tests require an equality relation in the null hypothesis, such that there seems to be no way to circumvent the problem described above. The implication, however, basically is that the detection of temporally derivable item sets is optimistically biased. An alternative to generally circumvent hypothesis testing would be the introduction of a user-defined threshold for the mutual information. Nonetheless, its reasonable definition may be a tedious and complicated task for a user.

## 3.7 Temporally Closed Item Sets

If an item set $X$ is temporally derivable from $Y$ then both item sets have histories with qualitatively the same shape. Under the assumption that the relevance of an item set is primarily determined by the qualitative changes represented in its history both item sets, $X$ and $Y$, have the same interestingness. For example, in Figure 3.4 on page 50 both histories show the same characteristic features that attract a user's attention: an upward trend coupled with what appears to be a seasonal variation. Hence, if one item set and its history are known, learning about the other item set and its history does not lead to any new insights. One item set of the two is superfluous and does not need to be presented to an expert. Removing superfluous item sets obviously leads to a smaller collection of item sets without losing any knowledge. Further elaborating on this idea leads to a condensed representation which builds upon the concept of temporal derivability. It accounts for redundancy imposed by the temporal dimension and eventually is smaller than condensed representations that do not account for it.

### 3.7.1 Definition and Properties

If two item sets are in relation by temporal derivability they have, apart from scaling, the same shape and are linked by set inclusion (i.e. one is a superset of the other). Temporal derivability thus partitions a collection of item sets into equivalence classes. Within a class item set histories have qualitatively the same shape and all are directly or indirectly connected by set inclusions. The latter means that if $X$ and $Y$ belong to the same equivalence class then, in the direct case, it is $X \hookrightarrow Y$. Otherwise there exists a $Z$ within the same class such that either $X \hookrightarrow Z$ and $Y \hookrightarrow Z$, or $Z \hookrightarrow X$ and $Z \hookrightarrow Y$, which is the indirect case.

**Example 1 (continued).** Figure 3.6 shows the Hasse diagram of this example's item sets (see Table 3.2, p. 35) together with their support histories. The support threshold is $\sigma_a = 2$. Equivalence class membership is indicated by related item sets being circumscribed. Temporal derivability partitions the collection of frequent item sets into nine equivalence classes, each one having a size between two and four elements. Note that albeit *acd* and *cdw* have the same shape they are not within the same equivalence class, because there exists no item set $Z$ such that either $acd \hookrightarrow Z$ and $cdw \hookrightarrow Z$, or $Z \hookrightarrow acd$ and $Z \hookrightarrow cdw$.

Taking the aforesaid into account, that an item set's interestingness is primarily dictated by its temporal development over time and not by its actual support values, the item sets

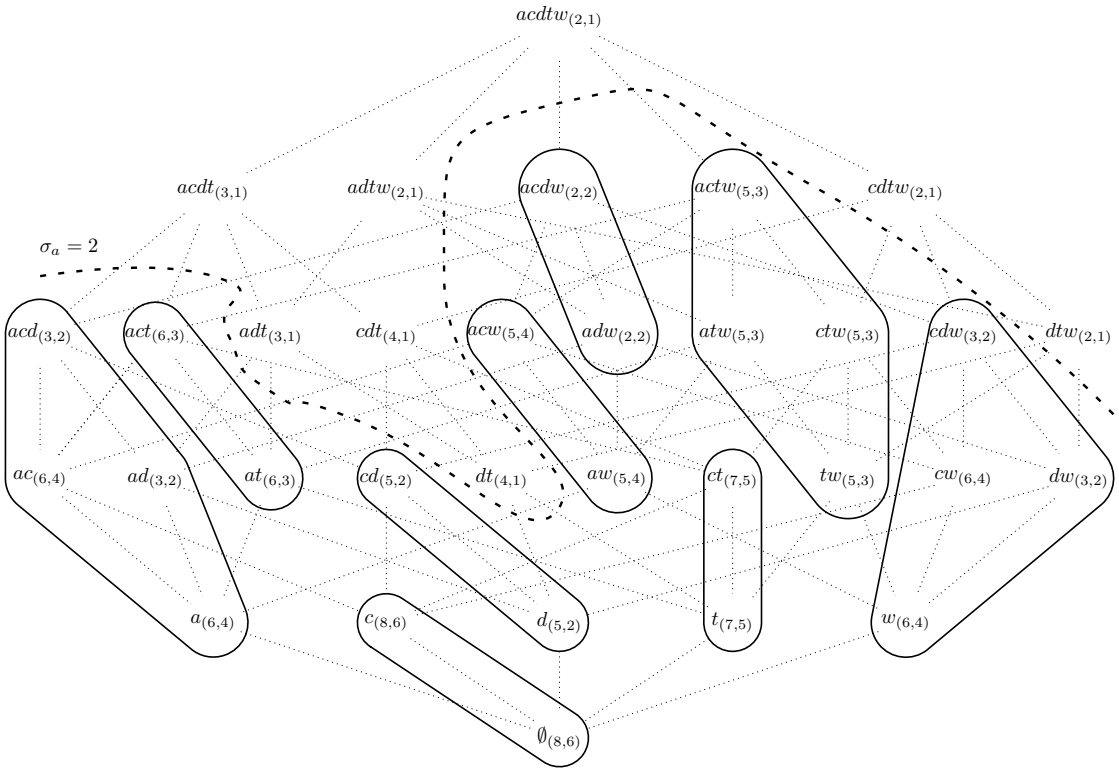



Figure 3.6: Hasse diagram for the item sets from Example 1 with equivalence classes imposed by temporal derivability

within an equivalence class are indiscernible with respect to their interestingness. Their essential information is captured within an equivalence class representative. For the sake of consistency to closed item sets, as the chosen reference condensed representation, the largest item sets of each class are taken. Leaning on the definition of closed item sets, *temporally closed item sets* are defined as follows:

**Definition 3.9** (Temporally Closed)**.** *An item set $X$ is temporally closed iff there exists no item set $Y \supset X$ such that $X \hookrightarrow Y$.*

The set of temporally closed item sets for a given data set $D$ and a partition of the time axis $\hat{T} := \{T_1, \ldots, T_n\}$ will be denoted $\mathit{TClosed}(D, \hat{T})$.

Temporally closed item sets are a subset of the item sets that are closed over a sequence, as the following theorem states:

**Theorem 3.8.** *Given a time-stamped data set $D$ and a corresponding partition of the time axis into periods $\hat{T} := (T_1, \ldots, T_n)$, the set of temporally closed item sets contained in $D$ is a subset of the set of item sets that are closed over the sequence $\hat{T}$, in other words $\mathit{TClosed}(D, \hat{T}) \subseteq \mathit{SeqClosed}(D, \hat{T})$.*

Combining Theorem 3.2 and Theorem 3.8 leads to the following corollary.

**Corollary 3.2.** *Given a time-stamped data set $D$ and a corresponding partition of the time axis into periods $\hat{T} := (T_1, \ldots, T_n)$, $\mathit{TClosed}(D, \hat{T}) \subseteq \mathit{Closed}(D)$.*

The last example demonstrated that *TClosed* indeed can be a proper subset of *Closed*.

| $\text{supp}_a$ | | |
|---|---|---|
| $T_1$ | $T_2$ | *FreqTClosed* |
| 8 | 6 | c |
| 7 | 5 | ct |
| 6 | 3 | act |
| 5 | 4 | acw |
| 5 | 3 | actw |
| 5 | 2 | cd |
| 3 | 2 | cdw, acd |
| 2 | 2 | acdw |

Table 3.11: Item sets contained in $D$ which are temporally closed and frequent based on $\sigma_a = 2$

This means, every temporally closed item set is also closed on the overall (not temporally split) data set but not every item set which is closed is also a temporally closed one. In fact, as the experimental results in Appendix A show, the set of temporally closed item sets is significantly smaller than the set of closed ones, which again can be significantly smaller than the set of frequent item sets.

Corollary 3.2 is of high practical relevance for three reasons. First, reconsidering the problem statement in Section 3.2, Corollary 3.2 ensures the required consistency of temporally closed item sets to an established condensed representation. It allows an interpretation of temporally closed item sets as a time-aware generalisation of closed item sets. Thus, temporally closed item sets are not just 'yet another' condensed representation.

Second, also as required in the problem statement the corollary shows that a further reduction in the number of item sets is possible when the temporal dimension of a data set is considered, compared to its neglectance.

Even though the set of temporally closed item sets describes a subset of the set of closed item sets, its size theoretically can be exponential in the number of items. For this reason, only the *frequent temporally closed item sets* are produced, analogous to frequent item set mining but also the other condensed representation discussed earlier. Nonetheless, because the definition of temporally closed item sets is based on a sequence of temporally ordered data sets rather than merely a single one the notion of what makes an item set frequent needs to be extended.

**Definition 3.10** (Frequent Temporally Closed)**.** *A temporally closed item set $X$ is frequent with regard to a minimum support threshold $\sigma$ iff $X$ is temporally closed and for its support history* $(\text{supp}_1(X), \ldots, \text{supp}_n(X))$ *it is* $\text{supp}_i(X) \geq \sigma,\ i = 1, \ldots, n$.

The set of frequent temporally closed item sets for a given data set $D$ with the partition of the time axis $\hat{T} := \{T_1, \ldots, T_n\}$ and support threshold $\sigma$ will be denoted $\mathit{FreqTClosed}(D, \hat{T}, \sigma)$.

**Example 1 (continued).** Table 3.11 shows temporally closed item sets contained within the example data set based on a support threshold of $\sigma_a = 2$. As expected from Corollary 3.2 the displayed item sets are a subset of the closed ones enlisted in Table 3.3 (p. 37).

### 3.7.2 Discovery

The third reason is, Corollary 3.2 devises a way to search a data set for temporally closed item sets. Rather than testing each individual frequent item set it is sufficient to consider only item sets that are closed in $D$, which means that in comparison far fewer tests are necessary. Since algorithms for identifying closed item sets have reached a high level of maturity, this offers the opportunity to yield an algorithm for temporally closed item sets through building upon prior work, as opposed to a development from scratch.

#### The CHARM Algorithm

The *CHARM* algorithm (Zaki and Hsiao, 2002, 2005) for frequent closed item set discovery serves as the foundation of the later proposed method for discovering temporally closed item sets. The choice of CHARM in favour of other algorithms such as *Close* (Pasquier et al., 1999), *CLOSET* (Pei et al., 2000) and *CLOSET+* (Wang et al., 2003a) lies in its superior performance (see Zaki and Hsiao, 2005, p. 473ff) while being similar in its general approach (see Ceglar and Roddick, 2006, p. 26).

Central to CHARM is the concept of *prefix-based equivalence classes* for item sets. Suppose an (arbitrary) order is defined on the items $x \in L$ and let $x_1, \ldots, x_n$ be the resulting ordered sequence of items. For each item set $X \subseteq L$ the function $p(X, k)$ returns $X$'s prefix of length $k$ based on the afore-defined order. This defines an equivalence relation $\theta_k$ on item sets as follows: $X \equiv_{\theta_k} Y \Leftrightarrow p(X, k) = p(Y, k)$. Thus, two item sets are in the same equivalence class if they share a common prefix of length $k$. Obviously, from $X \equiv_{\theta_{k+1}} Y$ follows $X \equiv_{\theta_k} Y$ such that a class based on prefix $P$ is a composition of the equivalence classes with prefix $Py$, whereby $y \in L$ and $x < y$ for all $x \in P$. In the same way, these classes are again sub-divisible and so on, until no further items are available to extend the prefix.

This yields a tree structure of equivalence classes, whereby the equivalence class with prefix length 0—the empty prefix—constitutes the root node. This class is identical to the power set of $L$, that is, it contains all item sets. Each (inner) node represents one equivalence class and its children are those equivalence classes that result from extending the node's prefix by one more item in accordance with the item order. Because each prefix uniquely corresponds to an item set (by neglecting the item order) and vice versa each item set yields a prefix (by sorting its items). Traversing the tree enumerates not only all equivalence classes but also all item sets.

Traversing the sub-tree rooted at a node with prefix $P$ returns all elements of the respective equivalence class. Hence, each node only has to store the information which items lead to its children. In short form, an equivalence class therefore is denoted $[P] = \{x_{i_1}, \ldots, x_{i_r}\}$, where $P$ is the prefix and parent node, and each $x_{i_j}$ a single item that represents the node (equivalence class) with prefix $Px_{i_j}$. To facilitate fast support calculation and other operations on the transaction set, CHARM augments each $x \in [P]$ with the set of those transactions which contain the item set that the prefix $Px$ represents. The set is denoted $t(Px)$, or short $t(x)$ if the prefix is clear. For notational simplicity when referring to a node in the tree, the notation of $[P]$ sometimes is extended to $[P] = \{x_{i_1} \times t(x_{i_1}), \ldots, x_{i_r} \times t(x_{i_r})\}$.

The tree structure breaks the problem of finding the frequent (closed) item sets into

```
CHARM(D, σ)
1  [∅] ← {x_i × t(x_i) : x_i ∈ L ∧ supp(x_i) ≥ σ}
2  D ← ∅
3  CHARM-EXTEND([∅], D)
4  return D

CHARM-EXTEND([P], D)
1  for each x_i × t(x_i) in [P]
2  do P_i ← P ∪ x_i and [Px_i] ← ∅
3     for each x_j × t(x_j) in [P] with j > i
4     do CHARM-PROPERTY(x_i, x_j, t(x_i) ∩ t(x_j), P_i, [Px_i], [P])
5     SUBSUMPTION-CHECK(P_i, D)
6     CHARM-EXTEND([Px_i], D)
7     delete [Px_i]

CHARM-PROPERTY(x_i, x_j, t_ij, P_i, [Px_i], [P])
 1  if supp(P_i ∪ x_j) ≥ σ
 2     then if t(x_i) = t(x_j)
 3             then Remove x_j from [P]
 4                  P_i ← P_i ∪ x_j
 5             else if t(x_i) ⊂ t(x_j)
 6                     then P_i ← P_i ∪ x_j
 7             else if t(x_i) ⊃ t(x_j)
 8                     then Remove x_j from [P]
 9                          Add x_j × t_ij to [Px_i]
10             else if t(x_i) ≠ t(x_j)
11                     then Add x_j × t_ij to [Px_i]

SUBSUMPTION-CHECK(P, D)
1  h(P) ← Σ_{T∈t(P)} T
2  for each Y in HASHTABLE(h(P))
3  do
4     if P ⊆ Y ∧ supp(Y) = supp(P)
5        then return
6  D ← D ∪ P
```

Figure 3.7: The CHARM-Algorithm for discovering frequent closed item sets from a data set $D$ using a minimum support threshold $\sigma$. (Zaki and Hsiao, 2005)

independent sub-problems and makes it recursively describable: The search results from the equivalence class with prefix $P$ are the aggregate of the results obtained from searching its child equivalence classes in the tree. Clearly, no sub-tree of a node with infrequent prefix needs to be visited. Zaki and Hsiao (2005, p. 465f) provide the following properties of pairs $x \times t(x)$ which allow to skip equivalence classes during traversal and thus leverage a faster closed item set discovery.

**Theorem 3.9.** *Let $x \times t(x)$ and $y \times t(y)$ be any two members of a prefix equivalence class $[P]$ and $x < y$. Further let $c(Px)$ denote the function that maps an item set $Px$ to its closure (i.e. the corresponding closed item set). The following four properties hold:*

1. *If $t(Px) = t(Py)$, then $c(Px) = c(Py) = c(Pxy)$.*
2. *If $t(Px) \subset t(Py)$, then $c(Px) \neq c(Py)$, but $c(Px) = c(Pxy)$.*

3. *If* $t(Px) \supset t(Py)$*, then* $c(Px) \neq c(Py)$*, but* $c(Py) = c(Pxy)$.

4. *If* $t(Px) \not\subseteq t(Py)$ *and* $t(Px) \not\supseteq t(Py)$*, then* $c(Px) \neq c(Py) \neq c(Pxy)$.

Property 1 implies that node $Px$ is skippable and that the traversal can continue with node $Pxy$. Node $Py$ can be removed from further consideration as its closure is the same as the one of $Pxy$. Property 2 also implies to skip over $Px$, yet not to remove $Py$ from further consideration because its closure is different from those of $Px$. Property 3 is symmetric to Property 2, nevertheless has different consequences. It implies that $Py$ can be removed from further consideration, as its closure and the one of $Pxy$ are the same, and $Pxy$ will already be visited while traversing the sub-tree rooted at $Px$. Property 4 states that both $Px$ and $Py$ have different closures, hence no skips are possible.

Figure 3.7 shows CHARM in pseudo-code notation. It consists of four parts: CHARM, CHARM-EXTEND, CHARM-PROPERTY and SUBSUMPTION-CHECK. CHARM in Line 1 initialises the equivalence class with prefix length 0—the root of the tree—to the frequent single items. Each item is paired with the set of transactions that support it. In Line 3 CHARM then passes the root node to CHARM-EXTEND. This routine in Line 6 recursively traverses depth first through the tree routed at node $[P]$ provided in the routine's argument. CHARM-EXTEND is also responsible for constructing the child nodes $[Px_i]$ of $[P]$ (Line 2). For each item $x_i \times t(x_i)$ in $[P]$ (Line 1), it combines it with any other pair $x_j \times t(x_j)$, that in the employed order comes after the former (Line 3). For just enumerating all nodes in the tree (and thus all item sets) it would be sufficient to append all resulting pairs $x_j \times t(x_i) \cap t(x_j)$ to $[Px_i]$ and to subsequently call CHARM-EXTEND with the latter.

This, however, is inefficient since the properties provided in Theorem 3.9 allow for skipping nodes. The properties are tested in CHARM-PROPERTY which is called in Line 4 of CHARM-EXTEND. The test result may trigger changes to the node $[Px_i]$: by adding the element $x_j$ to its prefix, in case property 1 or 2 hold, or by appending the pair $x_j \times t(x_i) \cap t(x_j)$ to it, in case property 3 or 4 hold. If property 3 is true, CHARM-PROPERTY additionally modifies the node $[P]$ by removing the element $x_j \times t(x_j)$, thus that the sub-tree rooted at $[Px_j]$ will not be traversed anymore. After producing the node $[Px_i]$ it is guaranteed that no node visited in the future will render $[Px_i]$ non-closed. It is thus closed unless a previously found closed item set (i.e. one, that does not share a common prefix with $[Px_i]$) is its closure. SUBSUMPTION-CHECK tests whether this is the case. If no such closed item sets exists, $Px_i$ is inserted in the set of closed item sets (Line 6).

### The CHARM-T Algorithm

*CHARM-T* extends CHARM such that temporally closed item sets are discovered. Those later described extensions serve two purposes: to obtain the support history of each frequent closed item set and to speed up the discovery of temporally closed item sets by skipping or avoiding unpromising sub-trees.

According to Definition 3.9 temporally closed item sets are based on the notion of temporal derivability, for which Section 3.6.3 provides a statistical test. Temporal derivability and the test rely on the support histories of item sets being available, that is the item sets' support within each time period. CHARM, however, only operates on a single

```
CHARM-T(D^T, σ)
1  [∅] ← {x_i × t(x_i) : x_i ∉ T̂ ∧ supp(x_i) ≥ σ} ∪{ x_i × t(x_i) : x_i ∈ T̂}
2  FreqTClosed ← ∅
3  CHARM-T-EXTEND([∅], FreqTClosed)
4  return FreqTClosed

CHARM-T-EXEND([P], FreqTClosed)
 1  for each x_i × t(x_i) in [P] with x_i ∉ T̂
 2  do P_i ← P ∪ x_i and [Px_i] ← ∅
 3     for each x_j × t(x_j) in [P] with j > i
 4     do if x_j ∉ T̂
 5           then CHARM-PROPERTY(x_i, x_j, t(x_i) ∩ t(x_j), [Px_i], [P])
 6        if x_j ∈ T̂
 7           then if supp(P_i ∪ x_j)/ supp(x_j) > σ
 8                   then Add x_j × (t(x_i) ∩ t(x_j)) to [Px_i]
 9                   else Jump to outer loop and continue with next x_i
10     if TEMPORAL-DERIVABILITY-TEST([P], [Px_i])
11       then FreqTClosed ← FreqTClosed \P
12     SUBSUMPTION-CHECK([Px_i], FreqTClosed)
13     CHARM-T-EXTEND([Px_i], FreqTClosed)

CHARM-PROPERTY(x_i, x_j, t_ij, [Px_i], [P])
 1  if supp(P_i ∪ x_j) ≥ σ
 2    then if t(x_i) = t(x_j)
 3            then Remove x_j from [P]
 4                 P_i ← P_i ∪ x_j
 5            else if t(x_i) ⊂ t(x_j)
 6                    then P_i ← P_i ∪ x_j
 7            else if t(x_i) ⊃ t(x_j)
 8                    then Remove x_j from [P]
 9                         Add x_j × t_ij to [Px_i]
10            else if t(x_i) ≠ t(x_j)
11                    then Add x_j × t_ij to [Px_i]

SUBSUMPTION-CHECK(P, FreqTClosed)
1  for each P' in FreqTClosed with P' ⊃ P
2  do
3     if TEMPORAL-DERIVABILITY-TEST([P], [P'])
4       then return
5  FreqTClosed ← FreqTClosed ∪{P}
```

Figure 3.8: The CHARM-T-Algorithm for discovering frequent temporally closed item sets from a data set $D^T$ using a minimum support threshold $\sigma$.

data set, not on multiple ones that are temporally ordered. Section 3.5.2 (p. 47ff) offers a solution to this conflicting situation: Time periods are encoded as individual items and each transaction is extended accordingly prior to invoking CHARM. Following the notion introduced earlier, the data set $D$ then turns into a data set $D^T$.

CHARM enumerates the closed item sets contained in a data set $D$. Because according to Corollary 3.2 the temporally closed item sets are a subset of the closed item sets, each closed item set is a candidate temporally closed item set. This entails that when

replacing $D$ with $D^T$ as input to CHARM, these candidates must still be produced. Because only frequent temporally closed item sets are of interest, the candidate closed item sets can be restricted to those that have non-zero support in each time period. Under this assumption, Theorem 3.3 and Theorem 3.4 apply. Theorem 3.3 guarantees that for any item set which is closed in $D$, its entire support history can be constructed from the result produced by CHARM on $D^T$. Theorem 3.4 ensures that the result produced on $D^T$ also contains any item set which is closed in $D$ and present in each time period. Consequently, the result from CHARM on $D^T$ contains the (frequent) closed item sets from $D$ plus their support histories—all information necessary for finding the temporally closed item sets is available.

The test for temporal derivability provided in Section 3.6.3 may be conducted as a post-processing step after applying CHARM to $D^T$. However, this would require to enumerate each closed item set at least twice—once during the CHARM run, and once in the post-processing step. A more efficient way is to incorporate the test into CHARM. The result is the CHARM-T algorithm shown in Figure 3.8.

CHARM-T extends and amends CHARM in several ways. As discussed above, CHARM-T receives as input the data set $D^T$ which differs to $D$ in that it encodes time periods as additional items. Since these items have a special meaning, it is necessary to distinguish them from the others. For this the (short-handed) predicate $x \in \hat{T}$ is used. The concept of predicate equivalence classes and the traversal through the item sets at their core are the same as in CHARM. However, items $x \in \hat{T}$ are treated differently in order to reduce the number of visited prefix equivalence class predicates (i.e. nodes). First and foremost, when constructing the child equivalence classes $[Px_i]$ of $[P]$ only items $x_i \notin \hat{T}$ are considered in CHARM-EXTEND (Line 1). The reason is that the equivalence class prefixes later result in closed item sets, but there is no interest in closed item sets which contain an item that marks a time period. In fact, the interest solely lies in obtaining a support history. For this, it is sufficient that for all $x_i \in \hat{T}$ the pair $x_i \times t(x_i)$ is contained in $[P]$ (Line 8). In Line 10 all the information necessary is available to conduct the test for temporal derivability. In case of a positive outcome the item set $P$ is not temporally closed and hence is removed from *FreqTClosed*.

The sub-tree rooted at $[Px_i]$ is not traversed if $Px_i$ is infrequent in at least one period. To avoid unnecessary computations, this is already tested before adding the item $x_i$ to the prefix equivalence class (Line 1 of CHARM-PROPERTY, Line 1 of CHARM-T). The tested condition $\text{supp}(P_i \cup x_j) \geq \sigma$, however, provides only a necessary criterion. That is, if it is false it implies that a time period exists in which $Px_i$ is infrequent; but if it is true it cannot be concluded that $Px_i$ is frequent in all periods. Thus, for those $Px_i$ which 'slip through' a second test using a sufficient criterion is carried out later, when the pairs $x_j \times t(x_j),\ x_j \in \hat{T}$ are added to $[Px_i]$ (Line 7 of CHARM-T-EXTEND). In Line 12 of CHARM-T-EXTEND the prefix equivalence class $[Px_i]$ and thus also the history of the item set $Px_i$ are complete. Before starting the traversal of $[Px_i]$'s corresponding sub-tree in Line 13, it is tested whether $Px_i$ is temporally non-closed with regard to one of its earlier-visited supersets. This requires a data structure for *subset containment* queries, to efficiently retrieve for $Px_i$ all supersets stored in *FreqTClosed*. Charikar et al. (2002) published such a data structure. If no earlier-visited superset is found which renders $Px_i$ temporally non-closed, the item set $Px_i$ is provisionally added to the set of temporally closed item sets (Line 6 of SUBSUMPTION-CHECK). It may be

removed again if it temporally closed with regard to its later-visited supersets (Line 11 of CHARM-T-EXTEND).

### 3.7.3 Supplemental Structures

Knowing all elements in *FreqTClosed* and their support histories allows for identifying those frequent item sets that have a unique history and for determining whether a non-temporally closed item set is frequent. In practise, nonetheless, two more questions sometimes are of interest: First and foremost: What is the shape of the support history of a non-temporally closed item set? And second: What is the actual support value of a non-temporally closed item set? The latter is equivalent to asking for the values of the item sets' support history. While the first question reflects the change analysis perspective on an item set, that its interestingness is primarily dictated by its histories' shape, not its concrete values; the second question takes on a more traditional view on item set interestingness.

*FreqTClosed* on its own is not sufficient to answer either of these questions. A supplemental structure is necessary. Section 3.4.4 analysed that all the condensed representations presented in Section 3.4 require one or more of such structures, with closed item sets being the only exception. In particular, Section 3.4.4 identified two types of supplemental structures: rule sets and border sets. The former being used to determine the support, the latter being used to determine the frequentness of those item sets which are not part of the condensed representation. Throughout the remainder of this section it should be remembered from the discussion in Section 3.4.4 that condensed representations solve a presentation and not a storage problem. For this reason, the presence of a supplemental structure only is of minor importance.

To answer the question for actual support values, *FreqTClosed* must be supplemented with the collection of all valid rules of the form $Y \Rightarrow Z$, where $Y \cup Z \in$ *FreqTClosed*, $Y \in$ *FreqClosed* and $Y \hookrightarrow Y \cup Z$. Using those rules the frequent closed item sets can be reconstructed, and from them the support histories of all other frequent item sets inferred. The combination of several of this Section's findings provides the justification: The frequent closed item sets are produced by CHARM-T as the only candidates that are to be tested for temporal closedness (see Section 3.7.2, p. 62f). A closed item set is non-temporally closed only if another closed item set can be temporally derived from it (see Section 3.7.1, p. 57f). Being temporally derivable, in turn, is conceptually equivalent to the occurrence of rules whose confidence does not change over time (see Section 3.6.2, p. 51f); such as the above ones. Indeed, and overall, the above suggested supplemental structure already is implicitly generated during a CHARM-T run and thus only needs to be unfolded.

This approach has two practical benefits: First of all, the size of the supplemental structure is bounded by the number of closed item sets minus the number of temporally closed item sets. This is a huge improvement over other condensed representations' supplemental structure whose size may become larger than those of the set of frequent item sets (see p. 43). The second benefit is the consequence of item set discovery being sometimes seen merely as a prior step to producing association rules. Analogous to how support histories are used for item sets to judge their interestingness, confidence histories can be defined for association rules to serve the same purpose. Following the arguments

provided in Section 3.6.1, rules with invariant confidence are regarded uninteresting. Exactly these rules are represented by the above defined structure, and thus they may serve as a backlist for rules not to be displayed to a user. This avoids an explicit test for confidence histories' invariance.

To answer the question for the shape of item sets' history, *FreqTClosed* must be supplemented with the collection of all valid rules of the form $Y \Rightarrow Z$, where $Y \cup Z \in$ *FreqTClosed*, $Y \in$ *FreqClosed*, $Y \hookrightarrow Y \cup Z$ and there exists no $X \subset Y$, $X \in$ *FreqClosed* with $X \hookrightarrow Y$. This definition is the same as for the rule set before with an extra constraining condition added. The resulting rule set thus is a subset of those necessary to produce the actual support values. Intuitively, these rules describe the lower bounds of each equivalence class that results from temporally derivability (see Section 3.7.1, p. 57ff). In combination with *FreqTClosed* these lower bounds allow to state for any item set whether it belongs to a certain equivalence class, or not. Since all histories within an equivalence class have the same shape, the above question can therefore be answered. It should, however, be noted that the reconstruction of the history shape does not account for noise, hence it is only an approximation.

## 3.8 Data-centric Change Utilisation

As an example how change can be utilised for item set mining, this chapter describes a change-based condensed representation which lessens the shortfall of frequent item set mining to produce a vast number of item sets more effectively than conventional, change-ignoring condensed representation. For this example, specific theorems and algorithms have been detailed. Yet, for change utilisation to be carried over to other data mining tasks, general principles and a more theoretic framework are needed.

Taking a bird's eye perspective on the content of this chapter, three core principles can be identified. First, the data is enriched by additional information which, in its symbolic representation, encodes temporal aspects of the data. Second, the resulting temporally augmented data set is processed by an algorithm which is unaware of the temporal semantics of some of the data set's symbols. Third, the resulting pattern set contains extractable higher level temporal information (compared to the one in the input data) whose characteristic is that filtering on it yields a pattern set which is in some aspects superior to the one that would be retrieved from the original, not enriched data.

It is clear that the second and third principle only are applicable and effective if the symbolic representation and the temporal information it encodes are cleverly chosen with these principles in mind. This puts strong emphasise on the data rather than the algorithm it is processed by and for this reason this approach to change utilisation is named *data-centric*.

More formally, let $\mathcal{A}$ be a set of symbols and $\mathbb{D}[A]$ be the space of all data sets that are based on $\mathcal{A}$'s elements. Let $\mathcal{A}^T$ be a symbol set that results from extending $\mathcal{A}$ with additional symbols $t_i$ which encode temporal information, e.g. $\mathcal{A}^T := \mathcal{A} \cup \{t_1, \ldots, t_n\}$, $t_i \notin \mathcal{A}$, and $\mathbb{D}[A^T]$ the space of all data sets based on it. Further, let $\mathbb{P}[\mathcal{A}]$ be the space of all pattern sets that are based on $\mathcal{A}$.

The data-centric approach to change utilisation is diagrammed in Figure 3.9. Its goal is the same as for any data mining approach, to produce a pattern set $\mathcal{P}$ from a data set $D$

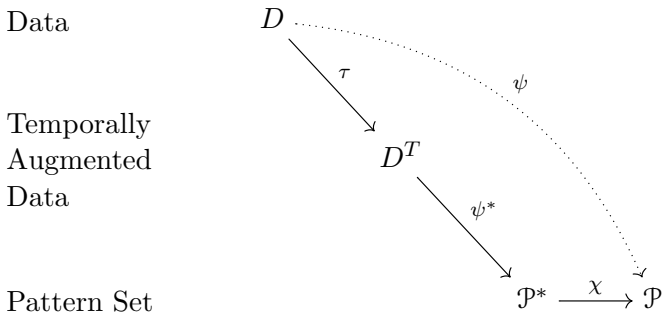



Figure 3.9: Data-centric change utilisation uses a temporally-augmented data set $D^T$

by some algorithmic procedure, which formally is described as a function $\psi : \mathbb{D}[A] \longrightarrow \mathbb{P}[A]$. Central to data-centric change utilisation is the function $\tau : \mathbb{D}[A] \longrightarrow \mathbb{D}[A^T]$ with $\tau(D) = D^T$ which maps a data set $D$ to a temporally enriched one $D^T$. To $D^T$ a function $\psi^* : \mathbb{D}[A^T] \longrightarrow \mathbb{P}[\mathcal{A}^T]$ with $\psi^*(D^T) = \mathcal{P}^*$ is applied which extracts patterns from it, yet is unaware of the temporal semantics that some of $D^T$'s symbols are carrying. The algorithm which realises $\psi^*$ thus can be an existing data mining algorithm. In particular, it must not even have been originally designed with intending change utilisation, like the closed item set miner $CHARM$ that was employed in this chapter. $\mathcal{P}^*$ contains three kinds of patterns: those stemming from the original data set, higher level aggregates of the temporal symbols, and mixtures of the both. As the last step, a filter function $\chi : \mathbb{P}[\mathcal{A}^T] \longrightarrow \mathbb{P}[\mathcal{A}]$ with $\chi(\mathcal{P}^*) = \mathcal{P}$ is applied which filters the first kind of patterns based on criterions defined on the latter two kinds.

In practise, as seen in Section 3.7.2, there may be no necessity for a strict separation of concerns between $\psi$ and $\chi$. For reasons of computational efficiency, the latter might be integrated into the former to filter patterns 'on the fly', as they are generated. Furthermore, analysis of the function $\chi$, might allow for $\psi$ integrating optimisations to avoid the needless patterns being produced.

Data-centric change utilisation does not only apply to frequent patterns or item sets, in particular. Indeed, it is applicable to other approaches for pattern discovery as well. For instance, Höppner and Böttcher (2007) published the PAMALOC algorithm which "enables the detection of local patterns in noisy data sets more reliable compared to the case when the temporal information is ignored" by making use of the observation "that noise does not reproduce its incidental structure over time but even small patterns do". The authors thus employed change utilisation for enhancing the reliability of small pattern detection, specifically clustering. In light of the previous discussion, their approach briefly reads as follows: The function $\tau$ adds to the data set a further attribute which uniquely assigns a data slice number to each record. A slice can be thought of as describing data from a certain small time period. These slices are then aggregated to sub-data sets covering larger time-periods which overlap by one slice. A records membership to sub-data sets is described by adding appropriate attributes to the data. The function $\psi^*$ basically is a standard OPTICS cluster algorithms which produces a clustering for each sub-data set. The function $\chi$ then filters out all incidental clusters by determining whether they do not repeat over time.

## 3.9 Conclusion

Chapter 1 assigned data mining areas to perspectives, depending on whether they ignore change or describe it, and proposed a third perspective that centres around its *implicit* utilisation as a leverage for tackling weaknesses of today's data mining approaches. This Chapter focused on the weakness of frequent item set mining to produce result sets of a size that is manageable only with great difficulty. It proposed and described a solution—temporally closed item sets—which reduces the number of produced item sets through incorporation of the temporal dimension. Because change can only occur over time, the solution does not ignore change; knowledge about change is not produced either. For this reason it neither belongs to the first nor to the second traditional perspective on change.

Indeed, temporally closed item sets belong to the in Chapter 1 proposed third perspective on change for three reasons. First of all, they address a problem for which also conventional data mining solutions exist that do not incorporate change at all, specifically condensed representations. In particular, they regard change as another 'feature' of the data, encoded by items that represent time periods. Second, temporally closed item sets do not ignore this previous work. They take advantage of closed item sets and at the same time enhance them. Third, temporally closed item sets provide a benefit for item set-based change mining because they reduce the number of observed changes dramatically—as the experiments in Appendix A (p. 97ff) demonstrate—and help to identify the core changes within a domain.

The concept of temporally closed item sets and the CHARM-T algorithm for their discovery have several characteristic technical features which are described next, and put in relationship to the requirements defined in the problem statement in Section 3.2.

- **Consistency:** The concept of temporally closed item sets is derived from the concept of closed item sets, and CHARM-T is an extension of CHARM. Thus, temporally closed item sets are comparable to closed item sets, which in this way becomes a reference representation. Temporally closed item sets and closed item sets do not contradict each other, thus are consistent, because the first is a subset of the latter, as Corollary 3.2 states.
- **Reduction:** The set of temporally closed item sets is produced from a sequence of data sets, as opposed to closed item sets which are produced from the data sets' union. That means, both representations are produced from the same data, yet there are fewer temporally closed item sets due to the aforementioned set inclusion. Appendix A.1 documents experiments on two real-world data sets, which show that the set of temporally closed item sets is significantly smaller than the set of closed item sets. On one data set, closed item set discovery reduces the number of item sets by 20%, whereas temporally closed item set discovery reduces it by 54%. On the other data set, closed item set mining leads to no reduction in the number of item sets, whereas with temporally closed item sets a reduction by 55% is obtained.
- **Robustness:** Section 3.4.5 discussed why closed item sets are not suitable in the presence of (even slightly) noisy data. Even in massive data sets one erroneous transaction is able to turn an actually non-closed item sets into a closed one. The reason that has been identified is the approaches for producing closed item

sets, such as CHARM, rely on a test for strict support equality between item sets. CHARM-T does not have this deficit. Whether an item sets is temporally closed, or not, is determined by means of a statistical test, which is presented in Section 3.6.3.

- **Meaningfulness**: Section 3.6.1 identified that properties of a domain are interesting when they change. Conversely, invariant properties are uninteresting. Item sets represent such a domain property. The concept of temporal derivability which underlies temporally closed item sets (see Definitions 3.8 and 3.9) describes the invariant relationships in a domain, specifically between item sets. The relationships are representable by rules with constant confidence. Both, the utilisation of invariance and the rule representation when describing redundancy contribute towards the meaningfulness of temporally closed item sets

Chapter 4

# Utilising Change for Classifiers

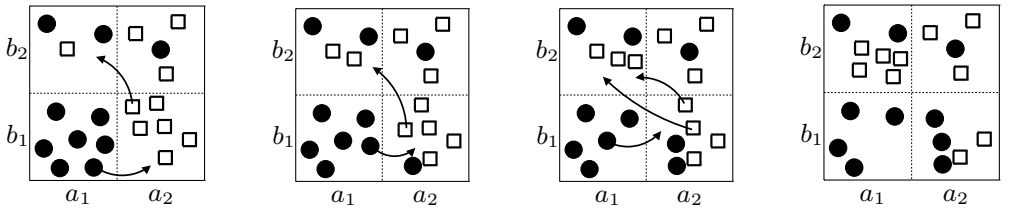
In a *classification* problem the goal is to assign class labels to test cases that are described by a set of attributes. In data mining this problem is approached inductively, by learning a *classifier* from a training data set of already classified cases. A classifier hence is a mapping between attributes and classes. The most popular type of classifiers are decision trees (Wu et al., 2008). Their popularity stems from their ability to deliver good classification results while still being comprehensible. The induction of a decision tree from a data set is a mature and well-researched topic. Aiming at improvements in comprehensibility and classification accuracy many algorithms have been proposed which emphasise on different aspects of the induction process. However, albeit all this research, decision trees share the following shortcoming with all other classifiers: in the presence of a changing domain a decision tree is out of date as soon as it has been induced. Applied to unseen data, the classification accuracy can therefore drop considerably. Shortening the interval between tree renewals is the most common method to limit the drop in performance, but that does not address the conceptual problem.

On the other hand, algorithms for decision tree induction have been developed based on the assumption that the underlying domain is stable. Thus they do not use the information contained in the training data in its entirety. Knowing that domains change over time is suggesting to ask: Can the temporal dimension and the change it represents be used to overcome the aforementioned problems, or at least to lessen them? The goal of this chapter is to show the potential of utilising change for classifiers by presenting an algorithm that uses a temporal model to learn decision trees in anticipation.

This chapter is structured as follows. Section 4.1 analyses the problems linked to decision tree induction, which then results in a problem statement in Section 4.2. Section 4.3 introduces the terminology and notations used in this chapter. Section 4.4 then outlines the induction algorithm for decision trees, followed by a discussion of typical approaches from the literature for coping with the given problem. Section 4.7 then continues with a detailed description of the proposed change utilisation approach. The final Section 4.8 then concludes this chapter.

## 4.1 Motivation

*Comprehensibility* is one of the major reasons why decision trees are chosen as classifiers. This is particularly important in applications where decisions have to be traceable and verifiable by human experts. Indeed, if insights on how the classifier operates are irrelevant, why not choose a different one such as a neural network or a support vector machine? Decision trees are predominantly comprehensible because they yield a set of classification rules. It would also be desirable to have the smallest tree that describes a certain problem. However, optimal learning of decision trees is NP-hard (Hyafil and Rivest, 1976). Instead a local search for a local optimum in the space of all possible




| (a) Period 1 | (b) Period 2 | (c) Period 3 | (d) Period 4 |
|---|---|---|---|
| $I_{gain}(A) = 0.531$ | $I_{gain}(A) = 0.278$ | $I_{gain}(A) = 0.118$ | $I_{gain}(A) = 0.007$ |
| $I_{gain}(B) = 0.007$ | $I_{gain}(B) = 0.030$ | $I_{gain}(B) = 0.066$ | $I_{gain}(B) = 0.191$ |

Figure 4.1: Illustration of how change within a domain can lead to trends in information gain.

decision trees is conducted; decision tree algorithms employ a greedy search strategy (see Section 4.4).

*Concept drift* is a major reason for a degrading classification *accuracy*. Chapter 1 showed that virtually any domain is subject to changes, and that the data it generates changes too. Over time, the concept, represented by a once learned decision tree (i.e. the concept of what constitutes the classes), may match reality less and less, and eventually fail to represent unclassified cases. To maintain a high classification accuracy regular relearning or revision of the decision tree is advocated in the literature. Still, as concept drift is continuous, training data is always, at least a bit, outdated. In consequence the classification accuracy will never be optimal. From the perspective of a comprehensive classifier the knowledge represented by the tree is (in a strict sense) only valid for the past but it yields no insights about the present or future. Even worse, caused by their inherent instability (see Breiman, 1996) the newly produced decision tree—even for a very slight concept drift—may completely differ from the previous one.

Section 2.4 detailed, that analysing change based on tree comparisons is not straightforward and constitutes a difficult problem which arises from its computational intractability. This, in combination with the aforementioned instability makes it impossible to anticipate decision trees directly based on a sequence of past ones. Moreover, the exploitation of temporal information in the data has not produced much research in the context of decision trees.

These problems raise the following question in relation to the topic of this thesis: How can knowledge about change be utilised during decision tree induction such that the produced decision tree better reflects the present or near-future characteristics of an evolving domain? The following two examples underline this idea.

**Example 3.** This example picks up on the example given in Section 2.4 and the corresponding Figure 2.4 on page 22. Figure 2.4 shows two decision trees learned at two different points in time. The aim of each is to classify instances as circles or squares. Or, from a different angle, it is to describe the concepts 'circle' and 'square' with the help of the attributes $A$ and $B$. As can be seen, the concepts at both time points $T_1$ and $T_2$ are completely different. For instance, at $T_1$ a circle is primarily describable as having the attribute value $a_1$, whereas in $T_2$ the description shifted to $b_1$. This change did not happen suddenly. Figure 4.1 shows the changes over four periods; period $T_4$ here corresponds to what has previously been $T_2$. Over time, the change appears to

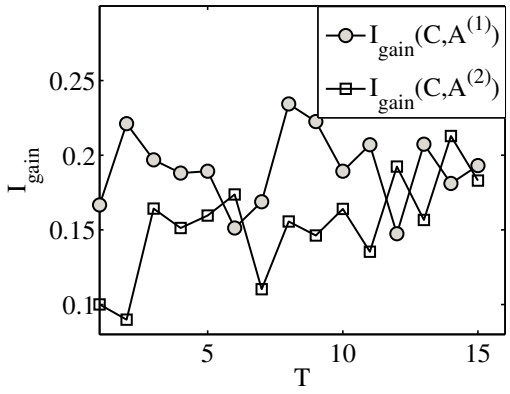



Figure 4.2: Histories of information gain values for two different attributes. The history of $A^{(1)}$ is apart from noise stable. The history of $A^{(2)}$ shows an upward trend.

be gradual; $B$ gains discriminative power, $A$ loses it and $B$ eventually outperforms $A$ somewhere between $T_3$ and $T_4$. The question is, whether one can already say in $T_3$ that $B$ may be the better choice in the future.

**Example 4.** Figure 4.2 has been obtained from a real world data set which is described in Böttcher et al. (2008b). It shows the discriminative power of the attributes $A^{(1)}$ and $A^{(2)}$ over 15 time periods. Focussing only on period 6 (and forgetting about the others), it appears that $A^{(2)}$ is superior to $A^{(1)}$. In fact, looking at all periods it becomes clear that this superiority was only due to noise. The question is, whether the noise can be identified as such and be removed by taking into account the functional relationship between time and discriminative power.

## 4.2 Problem Statement

The previous discussion leads to the following problem statement: Given is a sequence of data sets. Each data set corresponds to a time period. It is assumed that the time periods are disjoint and no more than one data set exists for each. The problem is to find a method which induces a decision tree from the sequence of data sets, thereby utilising the changes hidden within the data. The so produced decision tree should have a higher classification accuracy than a decision tree learned solely from the most recent data set by a change-unaware algorithm. As was elaborated, decision trees have the shortcoming of suffering from a degrading classification accuracy when learned in changing domains (as any classifier). Compared to other classifiers, however, decision trees have a good comprehensibility and usability. It is thus reasonable that a solution to the problem should improve on this shortcoming, but at the same time maintain comprehensibility and usability. The solution should be assessed by the following aspects:

- **Accuracy:** The classification accuracy is at least as good as those of trees which are solely learned from the most recent data set available. Ideally, they outperform the latter in the presence of a changing domain and noise.
- **Consistency:** The solution produces decision trees which look like as if they had been induced by a change-unaware decision tree learner. This means, they have the same general structure and do not exhibit any change-related properties. Consistency thus entails comprehensibility.
- **Usability:** The solution should depend only on a few, understandable parameters.

Since any solution to the above problem is entirely novel, some simplifying assumptions and decisions are made. First of all, the approach only deals with the mere induction process, pruning approaches are not discussed and are left for future research. Furthermore, the focus is on two-class, rather than multi-class classification. Because a multi-class problem is transformable into a set of two-class problems, this does not constitute a strong restriction. Lorena et al. (2008) review such transformation strategies.

## 4.3 Terminology and Notation

Throughout the chapter the following notation is used. A data set $D$ of sample cases is described by a set of nominal input attributes $\mathcal{A} := \{A^{(1)}, \ldots, A^{(m)}\}$ and a class attribute $C$. The domain of an attribute $A$ has $n_A$ values, i.e. $\mathrm{dom}(A) = \{a_1, \ldots, a_{n_A}\}$, and the domain of attribute $C$ has $n_C$ values, i.e. $\mathrm{dom}(C) = \{c_1, \ldots, c_{n_C}\}$. For stating evaluation measures it is necessary to refer to the number of sample cases in the data set having certain properties. This leads to the following notation:

| | |
|---|---|
| $p_{i\cdot}$ | relative frequency of class $c_i$ |
| $p_{\cdot j}$ | relative frequency of attribute $a_j$ |
| $p_{ij}$ | relative frequency of the combination of the values $c_i$ and $a_j$ |
| $p_{i\|j}$ | relative frequency of the value $c_i$ conditioned on $a_j$, i.e. $p_{i\|j} = \frac{p_{ij}}{p_{\cdot j}}$ |

Since the focus lies in the temporal dimension, let $D$ be a time-stamped data set and $[t_0,\ t_r]$ the minimum time span that covers all its samples. The interval $[t_0,\ t_r]$ is divided into $r > 1$ non-overlapping periods $[t_{i-1}, t_i[$, such that the corresponding subsets $D_i \subset D$ each have a size $|D_i| \gg 1$. Further, let $\hat{T} := \{1, \ldots, r, (r+1), \ldots\}$ be the set of all past $(i \leq r)$ and future $(i > r)$ period indices.

## 4.4 Learning Decision Trees

Algorithms for decision tree induction, such as CART (Breiman et al., 1984) or C4.5 (Quinlan, 1993), grow the tree recursively starting from the top using a greedy strategy. Figure 4.3 shows their basic structure (c.f. Borgelt and Kruse, 1998). The induction process is controlled by two different types of decisions: Firstly, starting at the root node an attribute $A$ is selected that yields the highest score regarding an attribute evaluation measure $I$ (Lines 2–6). The data set is then split into $n_A$ subsets each corresponding to one attribute value $a \in \mathrm{dom}(A)$ and a child node for each of them is created (Lines 10–12). Secondly, if all its cases have the same class label or a stop-criterion is reached, a subset is not split further and hence no children are created (Line 8). The current node then becomes a leaf and is assigned the majority class $c \in dom(C)$ of its associated subset (Line 9).

An attribute evaluation measure $I(C, A)$ rates the value of an attribute $A$ for predicting the class attribute $C$. The most well-known measures are information gain (Quinlan, 1986) and information gain ratio (Quinlan, 1993). The information gain $I_{gain}(C, A)$ measures the information gained, on average, about the class attribute $C$ when the value of the attribute $A$ becomes known. Based on the Shannon entropy $H = -\sum_{i=1}^{n} p_{i\cdot} \log_2 p_{i\cdot}$

```
GrowTree(D)
 1  I_best ← WORTHLESS
 2  for all untested attributes A
 3  do I ← I(D, A)
 4     if I > I_best
 5        then I_best ← I
 6             A_best ← A
 7  if I_best = WORTHLESS
 8     then create leaf node v
 9          assign c = argmax_{c_i}(p_1., ..., p_i., ..., p_{n_c}.) to v
10     else assign test on A_best to v
11          for all a ∈ dom(A_best)
12          do v.child[a] ←
13                GrowTree(D|_{A_best=a})
14  return v
```

Figure 4.3: Basic structure of algorithms for decision tree induction (Borgelt and Kruse, 1998)

as a measure for the information content (see Appendix C.1 for some background knowledge), the information gain is defined as

$$\begin{aligned} I_{gain}(C, A) &= H_C - H_{C|A} \\ &= -\sum_{i=1}^{n_C} p_{i\cdot} \log_2 p_{i\cdot} - \sum_{j=1}^{n_A} p_{\cdot j} \sum_{i=1}^{n_C} p_{i|j} \log_2 p_{i|j} \end{aligned} \tag{4.1}$$

A disadvantage of information gain is its bias towards attributes with many values. To overcome this problem the information gain ratio $I_{gr}(C, A)$ was proposed which penalises many-valued attributes by dividing the information gain $I_{gain}(C, A)$ by the entropy $H_A$ of the attribute itself.

$$I_{gr}(C, A) = I_{gain}(C, A)/H_A \tag{4.2}$$

Many other attribute evaluation measures have been published. Still, no single best measure exists and in practise often several are tried in order to find the one leading to be best decision tree (Borgelt and Kruse, 1998).

## 4.5 Handling Concept Drift

Almost since the early days of machine learning it is known that changing domains lead to a degradation in classifier performance. The classifier, seen as representing a model for a real concept, does not match the reality anymore. This is why research introduced the terms 'concept drift' and 'concept shift'. Section 2.5 (p. 24) already provided an introduction into these topics together with links to some classical readings, in the context of the question: How can be determined when a domain has changed considerably? The following section sheds light on contributions regarding another aspect: How can a degrading classifier performance be prevented? In particular, two types of approaches are discussed. The first type focuses on the most recent data, the second type focuses

on the reuse of past classifiers. For both types their core ideas are revealed and their drawbacks outlined.

### 4.5.1 Windowing and Forgetting

If a domain changes over time, it is sensible to assume that the more recent the training data is, the better the learned classifier may reflect the present and hopefully the near future. Thus, it comes with no surprise that the two basic techniques employed for learning in the face of concept drift are *moving temporal windows* (Widmer and Kubat, 1996; Hulten et al., 2001; Klinkenberg, 2004) and *age dependent weighting* (Klinkenberg, 2004). The method of moving temporal windows learns the decision tree from samples that were gathered within a certain, recent time window. For instance, in Widmer and Kubat (1996) a framework is described which heuristically and dynamically adapts the window size during the learning process. A moving temporal window approach to learn decision trees – the CVFDT algorithm – which scales up to very-large databases was proposed in Hulten et al. (2001). It maintains class counts in each tree node and when new data arrives decides whether or not a subtree needs to be re-learned.

For window-based approaches the choice of an appropriate window size is a crucial and difficult problem. In general, a compromise has to be found between small windows, which are required for a fast adaptation to concept drift, and large windows, which are required for a good generalisation. Kuh et al. (1990) and Helmbold and Long (1994) theoretically determined upper bounds on the speed of concept drift which are acceptable to learn a concept with a fixed minimum accuracy. Hence a window with a certain minimal fixed size allows to learn concepts for which the speed of drift does not exceed a certain limit.

Age dependent weighting simulates a data-ageing process by crediting more recent samples higher than old ones during learning. A different interpretation is that the learning process gradually forgets about older samples. In Klinkenberg (2004) methods for age dependent weighting are shown and compared with temporal window approaches. The weights are chosen such that learning emphasises for slowly changing domains on a suitable large set of samples and fast changing domains on only the most recent samples. In a way, age-dependent weighting is a 'fuzzified' version of the window-based approach. The age expresses a sample having a degree of membership to the training set, whilst in a window-based approach a sample is either contained, or not. This, however, leads to the same problem as for window-based approaches: a poor performance if a domain is more dynamic than initially expected.

### 4.5.2 Model Repositories

A different type of approaches is based on reusing past models. It rests on the assumption that in the long run a domain may changes the same way twice, if not more. This means, it assumes that classifiers repeat in the course of time.

In the broader context of pro-actively retrieving a future model the RePro system by Yang et al. (2005) constitutes seminal work. Its core building block is a repository of past classification models which is treated as a Markov chain. In RePro the Markov chain describes transition patterns between classification models. Using the Markov

chain RePro calculates the probability of each alternative model following the present model. The Markov Chain is triggered when the error rate within a small window of the most recent data drops below a threshold. If the chain predicts a model with a low probability the system defaults to retrieve that model from the history which has the highest classification accuracy on the same small window.

### 4.5.3 Assessment

The two discussed strategies have received varying levels of attention in the research community. While in particular windowing approaches already emerged in the 1980s, the usage of model repositories to retrieve recurring models is a rather new approach. Likewise, the former sparked considerable research in enhancements and extensions, whereas the latter seems to have received no greater attention.

Nevertheless, both strategies have their advantages and disadvantages. As discussed earlier, their primary aim is to increase the classification accuracy in the presence of concept drift. At the same time it is, however, desired to maintain comprehensibility and usability. In the following these three requirements are used to assess windowing and model repository approaches.

#### Consistency

All discussed approaches produce a single decision tree. This means, there is no difference to the result from standard decision tree induction algorithms. In this regard, there is no loss, but also no gain in consistency and comprehensibility.

However, the model repository approach does not predict a completely novel future model but searches for the best match out of a repository of past models. A past model is retrieved by conceptual equivalence to a former model. In the RePro system this equivalence is defined by comparison of the classification verdict of both models on a set of test instances. The meaning of the rules which the decision tree contains is completely ignored. This may lead to the situation that a tree is considered equivalent albeit its corresponding rules are not, based on the judgement of an expert.

The model repository approach further strongly assumes that models repeat in time following a predictable repetition pattern. This, in turn, is only likely to occur if the change triggering events repeat in time too—in the same order and each time with the same impact on the domain. This assumption has to hold for a rather long duration because unless data is collected very frequently and the domain changes often it will take a considerable amount of time to derive the huge number of models necessary to reliably learn transition patterns.

These characteristics not only fail to match a user's intuition and thus make the approach less comprehensible, they also render model repository approaches unsuitable for domains whose change is driven by a multitude of unknown, volatile and, in particular, one-off events. One-off events are predominant in many domains; ideally, an approach for dealing with dynamic domains should not have the above outlined restrictions.

#### Usability

Window-based approaches require a window size to be specified, or, if the window size is adaptable, the corresponding algorithm's parameters need to be chosen. Moreover, for a user it is difficult to decide whether a small or large window is preferable, in particular because the user is in the conflicting situation to choose between fast adaption to concept drift, and hence a small window size, or a statistically reliable model, and hence a large window size. Similar arguments hold for automatically adapted window sizes. Here the user needs to decide how fast it should react to a changing domain.

For the model repository approach a user must specify a metric for the conceptual equivalence of two models. Because computing the structural similarity on decision trees, like the tree edit distance, is MAX SNP-hard (see Bille, 2005), defining a meaningful but still computational feasible metric is a challenge.

#### Accuracy

The previously cited theoretical results by Kuh et al. (1990) and Helmbold and Long (1994) imply that window-based approaches – independent from whether they use a fixed or adaptive window size – perform well in domains with slow concept drift but may result in models with a low accuracy in very dynamic domains. Moreover, these approaches devise a global sample weighting scheme. For this reason they are unable to account for concept drift having considerably different speeds amongst subspaces of a domain. This means, a domain may change in certain aspects more drastically than in others. This inflexibility may have a negative impact on classification accuracy.

## 4.6 Process-centric Change Utilisation

Two strategies are conceivable to utilise change in the scope of decision tree induction; either the utilisation takes place as a post-processing step and combines the decision trees from each individual time period, or the utilisation takes place during the process of decision tree induction and exploits the time stamps contained in the data. The first strategy turns out to be practically unfeasible for the same reasons that also prevented the emergence of change mining methods for trees: tree comparison has a very high computational complexity and, moreover, decision trees learned by the typical top-down greedy approach are inherently instable. Section 2.4.1 provides more details on those two problems.

As an alternative to the intractable direct model comparison this section follows the second of the above strategies and proposes a tractable, generic approach that is based on a decomposition of the model induction process. Consider the induction process of a decision tree as an example. It can be described as a sequence of decisions comprising which attribute to take for the next split, when to stop growing the tree and which class label to assign to a leaf node. Decisions are driven by an attribute selection measure $I$ like the information gain and the class label distribution $P$. Following the algorithm of tree induction, the sequence of decisions then uniquely determines the model. In other words, if all possible values – the image – of the information gain and the class label distribution are known, a direct computation of the model is possible without going

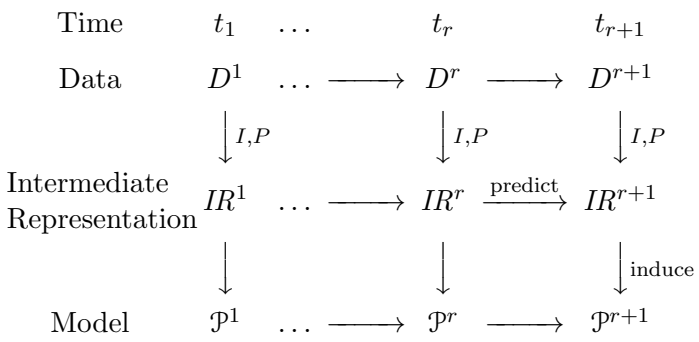



Figure 4.4: Process-centric change mining analyzes the sequences of intermediate representations $IR^i$

back to the data. In this respect, the image of $I$ and $P$ together form an *intermediate representation* of a decision tree that is sufficient for tree induction. An example for such an intermediate representation is the well known concept of *sufficient statistics* in probability theory. For instance, statistics can be sufficient to uniquely determine a probability distribution being the equivalent of our model.

More formally, when deciding on the attribute for the next split, $I$ is evaluated on the data subset $D' \subseteq D$ of the current branch of the tree for all attributes $A$ in the set of attributes $\mathcal{A}$. Then, the attribute is picked that maximises (or minimises) $I$. In other words, if the image $\mathcal{I} := I(\mathcal{A}, 2^D)$ of all possible combinations of attributes and subsets of $D$ is known, all the required information is available to pick the best attribute for a split at any stage of growing the tree. Adding the image $\mathcal{P} := P(2^D)$ of the class label distribution $P$ forms the intermediate representation $IR = (\mathcal{I}, \mathcal{P})$.

In order to predict a future model it is therefore sufficient to look at how the intermediate representation changes over time, predict their future values and then follow the usual model induction process to derive the future model. Having a sequence of data sets $(D^1, \ldots, D^r)$ a sequence of intermediate representations $(IR^1, \ldots, IR^r)$ can be generated. Almost always, the values in $IR^i$ will change over time, so do the decisions based on them and, finally, so does the resulting model. This means, by analysing how the values of $IR^i$ and thus the corresponding decisions change over time in each step of the induction process information about how the model will change can be retrieved. Figure 4.4 illustrates the proposed decomposition with predicted intermediate representation $IR^{r+1}$ and induced future model $\mathcal{P}^{r+1}$.

In fact, as will be seen in Section 4.7.4 in case of decision trees there is no necessity to compute and store the complete intermediate representation which can be computationally expensive. Instead, the algorithm generates the required parts of the $IR^i$ on the fly, when they are needed.

It is obvious that such an approach needs to be embedded into the learning process itself rather than being a subsequent analysis step as the idea of direct model comparison. Because this approach to change utilisation is tightly coupled with the learning process it is named *process-centric*.

The approach is useful only, if the intermediate representation is sufficient to derive the model, i.e. if the decomposition is well defined, and if the intermediate representation is of such a form that it can easily be predicted. As can be seen from the generic

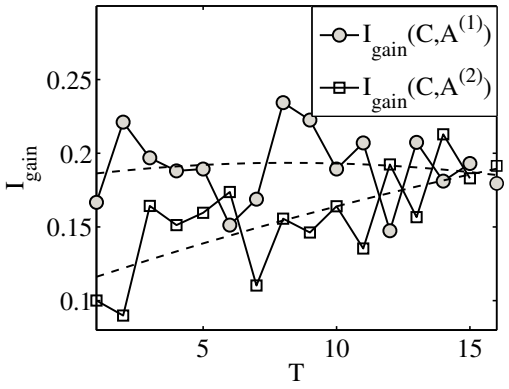



Figure 4.5: The histories from Figure 4.2 modelled by quadratic polynomials and displayed as dashed lines. In period 16 the values to be predicted are shown.

structure of algorithms for decision tree induction in Figure 4.3 both requirements are met for decision trees. The second requirement shows that decision trees are indeed well suited as an example. Since the measures $I$ and $P$ form real-valued time series for every attribute-data subset combination, we can apply standard prediction methods to determine $IR^{r+1}$.

## 4.7 Predicting Decision Trees

The approach proposed in this section employes process-centric change utilisation. The later presented $PreDeT$[1] algorithm models how the measures which control the decisions during the tree induction process change over time, predicts their values and derives a decision tree from the prediction. Predicting future values, it is capable of anticipating future decision trees which may not have occurred before. Predicting present values it makes the induction of decision trees more robust against noise. Unlike the aforementioned methods it does not rely on a repository of past trees and thus has the capability to deal with change initiated by one-off events.

**Example 3 (continued).** Figure 4.1 (p. 72) illustrates the change in a data set and the resulting change in information gain. It shows the distribution of samples over the attribute space at four consecutive time periods. Each sample belongs to one of two classes, squares or bullets, each described by two attributes $A$ and $B$ with domains $\{a_1, a_2\}$ and $\{b_1, b_2\}$, respectively. Assume a decision tree is being learned at the end of each period which predicts the samples in the next period. In period 1, shown in Figure 4.1(a), the information gain of $A$ is much higher than that of $B$ and it therefore would have been chosen as the split attribute. However, the distribution of samples shifts over time which is indicated by arrows in Figure 4.1(a) to Figure 4.1(c). In period 3 the information gain of $A$ is still higher than the one of $B$ and therefore $A$ would be the split attribute. This would lead to a classification error of 8 using the samples from period 4 for testing. However, in period 4 attribute $B$ would have been the superior split attribute. The choice solely based on the samples from period 3 was sub-optimal. Looking at how the information gain developed between periods 1 and 3 reveals a downward trend for $A$ and an upward trend for $B$. Using an appropriate model for both time series it would have been possible to anticipate the change in the split attribute and to choose $B$. This choice leads to a much smaller classification error of 5.

[1] Acronym for '**Pre**dicting **De**cision **T**rees'

**Example 4 (continued).** Figure 4.2 (p. 73) shows an example obtained from a real-world data set. The information gain history of the attribute $A^{(1)}$ is stable apart from noise whereas the information gain history of $A^{(2)}$ shows an upward trend. Furthermore, it can be seen that for the vast majority of time periods $T = 1, \ldots, 15$ attribute $A^{(1)}$ has more predictive power and would therefore be chosen as the split attribute. However, due to the observed upward trend in the information gain of $A^{(2)}$ both histories will intersect and $A^{(2)}$ will become the split attribute in the near future. Figure 4.5 shows the two histories from Figure 4.2 each modeled by a quadratic regression polynomial. In period 16 the – at the time of modeling unknown – information gain values of both attributes are marked. As it can be seen, the predictions made by the regression models anticipate the change in the ranking of candidate split attributes which happens between period 15 and 16.

In summary, the basic idea of the subsequently presented *PreDeT* algorithm is to learn models which describe the relationship between time and attribute evaluation measure, respectively class label distribution in each step of the decision tree induction. The models are then used to predict the value of the particular quantity for the next, future time period. Subsequently, the predictions are used to decide whether to grow a subtree and which class label to assign to a leaf node. As Section 4.3 already pointed out these two decisions are the main building blocks of the vast majority of decision tree learners. With the assumption that the predictions reflect the changes in the underlying domain, it is sensible to conjecture that *PreDeT* is capable to predict how a decision tree may look like in the future. In the presence of a changing domain this means that the produced classifiers should exhibit a higher accuracy than those which are solely reflecting the characteristics of historic data.

### 4.7.1 Models and Methods for Prediction

The *PreDeT* algorithm faces an inference problem involving data $\{(t_i, y_i) \mid i = 1, \ldots, r,\ t_i \in \mathbb{R},\ y_i \in \mathbb{R}\}$ where the $t_i$ are the inputs representing time, and the $y_i$ are the targets representing either attribute evaluation measure values or the relative frequencies of a class label. The algorithm requires *predictions* $y_*$ for new inputs $t_*$. Because in any practical application the true underlying relationship $\varphi' : \mathbb{R} \longrightarrow \mathbb{R}$ with $\varphi'(t) = y$ is an *unknown function*, such predictions are only feasible on basis of a *model* $\varphi : \mathbb{R} \longrightarrow \mathbb{R}$ with $\varphi(t) = y$ which reasonably approximates $\varphi'$.

Detailed knowledge about a domain would enable an informed choice regarding the *model structure*, that is the *family of functions* $\varphi'$ belongs to. One possibility of defining such a family is by using a parameterised function, for instance $\varphi(t) = a_2 t^2 + a_1 t + a_0$. The inference task then is to determine values for the parameters $a_i$ such that the resulting function $\varphi$ best fits the given data. In practise such detailed domain knowledge rarely is available and consequently the informed choice of a parameterised function is not practicable. In the particular case of the *PreDeT* algorithm the reason is the vast amount of models that are required during decision tree induction. In each step of the induction process a model and a prediction is required for each of the considered attributes. Further models are required for the class labels in each leaf node. Because each model may be significantly different from any other, the use of the same parameterised function as the basis for all models is not sensible.

The huge number of required models imposes the following four requirements on the choice of an approximation and prediction method. First, the method should be able to consider a large set of different families of functions. Second, it should be applicable even if not much data is available, i.e. if $r$ is low. Third, it should work without sophisticated parameters and the manual tuning and model validation effort they entail. Fourth, it should be sufficient for predicting the present or the very near future.

At its core, the history $\{(t_i, y_i) \mid i = 1, \ldots, r\}$ is a time series. However, the above requirements render typical methods for time series forecasting unsuitable, such as ARIMA. The reasons are the manual effort that is required for their application and their reliance on the availability of long time series. These issues have already been discussed in Section 2.3.2 (p. 19).

Two different methods which meet the requirements above, are linear regression with basis functions in combination with a model selection measure, and Gaussian process regression. Appendix C.2 describes these methods in detail. Both methods consider functions

$$\varphi(t) = \sum_{i=0}^{q} a_i \zeta_i(t) \tag{4.3}$$

with $a_i$ being the parameters and independent basis functions $\zeta_i(t)$. Any choice of $q$ in combination with any selection for the $\zeta_i$ yields a different family of functions. Linear regression with a model selection measure requires an explicit choice of the $\zeta_i$ and also a limit for their number using an upper bound for $q$. Gaussian process regression considers an unlimited number of $\zeta_i$ and thus no bounds for $q$ are required. The $\zeta_i$ are defined implicitly using a kernel function which encodes basic knowledge about the unknown function $\varphi'$ such as stationarity or degree of smoothness.

The main difference between linear regression with a model selection measure and Gaussian processes concerns how the model $\varphi$ is selected. A model selection measure, such as the Akaike criterion, aims to balance the complexity of a model against goodness of fit. Its idea is to use a limited set of families of functions, obtain a model candidate from each and select the final model by scoring the candidates' goodness of fit against their complexity. Figure 4.6 shows three functions fitted to the same data. The families of functions are linear, quadratic and cubic polynomials

$$\zeta_i(t) = t^i \tag{4.4}$$

for $q = 1, 2, 3$. A choice of $q = 2, 3$ already may impose the risk of overfitting the data, which may lead to predictions of poor quality, in particular because polynomials get highly oscillatory with increasing degree. A model $q = 1$ has a lower complexity but may lead to $\varphi$ only coarsely reflecting $\varphi'$.

Gaussian process regression accounts for functions appearing equally adequate in light of the available data without enforcing the strict selection of one. Its idea is to define a distribution over all possible functions, apply Bayes' rule given some data to infer a posterior distribution, and obtain a predictive distribution for an input by averaging over all individual functions' predictions weighted by their corresponding probability. Figure 4.7 shows two functions fitted to the same data as in Figure 4.6 but only this time randomly selecting

$$\zeta_i(t) = e^{-\frac{1}{2}(t-c_i)^2} \tag{4.5}$$

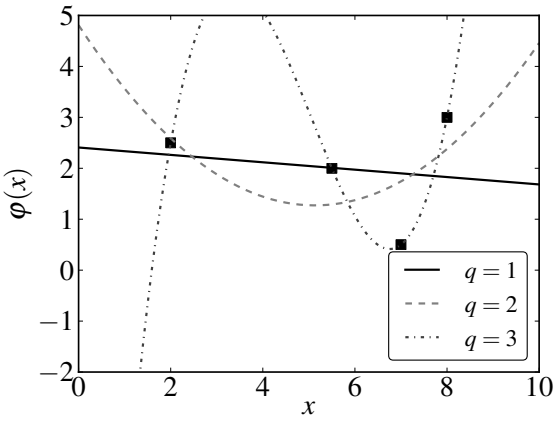



Figure 4.6: Regression using polynomials of different degree.

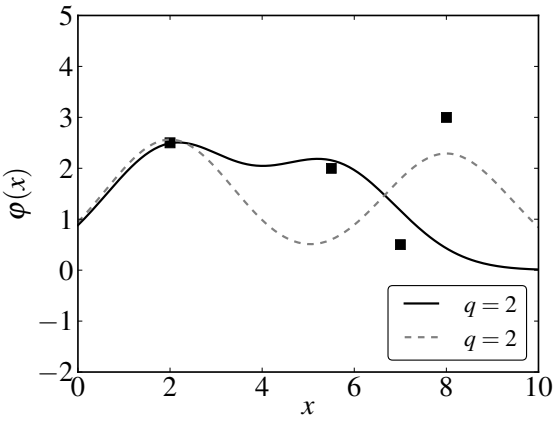



Figure 4.7: Regression using radial basis functions with randomly chosen centres.

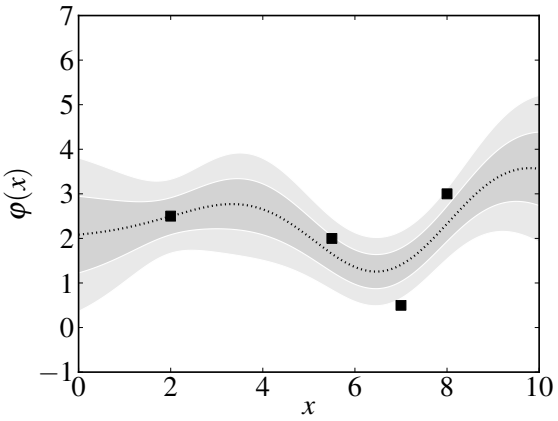



Figure 4.8: The dotted line shows the Gaussian processes' mean, which corresponds to the predicted value for each $x$ for the posterior.

for $q = 2$ (i.e. radial basis function networks with two nodes). Both functions reasonably fit and have the same complexity. Nevertheless, they are so different that only one or even none of them may reflect the unknown function $\varphi'$. Gaussian process regression, instead of arbitrarily selecting one model, takes both models (and an infinite number of other ones) into account by assigning to each model a posterior probability. Figure 4.8 shows the Gaussian process that results from the example data by considering an infinite number of radial basis functions.

On the one hand, Gaussian process regression is more flexible than linear regression with a model selection measure as it only requires basic knowledge about the function $\varphi'$ such as stationarity or degree of smoothness and considers a much larger number of families of function. On the other hand, it is much more computational demanding. As the experiments in Appendix A.2 show, good results can be obtained for both methods.

### 4.7.2 Predicting Attribute Evaluation Measures

Let $(D_1, \ldots, D_r)$ be a sequence of time-dependent data sets each described by the same attributes $A^{(i)}, i = 1, \ldots, m$ having the same domains in each time period. Quantities crucial for decision tree induction like the attribute selection measure (and also the distribution of class labels) are now related to a specific data set $D_i$ and thus to a certain time period $t_i$. For this reason they form sequences of values which are denoted by $\{(t_1, I^1(C, A)), \ldots, (t_r, I^r(C, A))\}$, or short $\mathfrak{I} := (I^1(C, A), \ldots, I^r(C, A))$. These sequences are referred to as *attribute evaluation measure histories*.

Assume that the relationship between $t \in \hat{T} \subset \mathbb{R}$ and the attribute evaluation measure $I \in \mathbb{R}$ at a particular node in the decision tree is expressible by a function $\varphi' : \mathbb{R} \longrightarrow \mathbb{R}$. By employing the methods described in Section 4.7.1 a model $\varphi(t)$ for an attribute evaluation measure that approximates $\varphi'(t)$ is obtained by inference from the data $\mathfrak{I}$. Due to $\hat{T} \subset \mathbb{R}$, $\varphi(t)$ is a function $\varphi : \mathbb{R} \longrightarrow \mathbb{R}$. To emphasise that it is based on a history $\mathfrak{I}$ of attribute evaluation measures it will also be denoted by $\varphi[\mathfrak{I}]$.

Let $t_*$ be the time for which a prediction has to be produced. In the *PreDeT* algorithm, during decision tree learning at each inner node a model $\varphi(t)$ is employed to obtain a prediction $\varphi[\mathfrak{I}](t_*)$ for the attribute evaluation measure's value at time $t_*$. This prediction is then used to decide for the split attribute at the particular node.

### 4.7.3 Predicting the Class Label Distribution

As in Section 4.7.2, let $(D_1, \ldots, D_r)$ be a sequence of time-dependent data sets each described by the same attributes $A^{(i)}, i = 1, \ldots, m$ having the same domains in each time period. Like for attribute evaluation measures, the class label distributions are now related to a specific data set $D_i$. Through their time dependence they form sequences, also called *class label distribution histories*, $\{(t_1, P^1), \ldots, (t_r, P^r)\}$, or short $\mathcal{P} := (P^1, \ldots, P^r)$. Therein, $P^k := (p^k_{1.}, \ldots, p^k_{n_C.})$ is the distribution of class labels and $p^k_{i.}$ is the relative frequency of class attribute value $i$ in time period $k$.

A model $\psi$ for histories of class label distributions is a function $\psi : \mathbb{R} \longrightarrow [0, 1]^{n_C}$, due to $\hat{T} \subset \mathbb{R}$. It is learned from the history of class label distributions $\mathcal{P}$. The dependency of $\psi$ on $\mathcal{P}$ is denoted by $\psi[\mathcal{P}]$. Within the *PreDeT* algorithm a model $\psi$ is used in each leaf node to predict the class label distribution at time point $t_{r+1}$.

The model $\psi$ is a vector of functions $\psi_i : \mathbb{R} \longrightarrow [0, 1]$ each of which models a dependency between the time period and the relative frequency (estimated probability) of a class label. Because the relative frequencies must sum up to one $\sum_{i=1}^{n_C} \psi_i(t) = 1$ must hold, i.e.

$$\psi(t) = \begin{pmatrix} \psi_1(t) \\ \psi_2(t) \\ \vdots \\ \psi_{n_C}(t) \end{pmatrix} = \begin{pmatrix} \psi_1(t) \\ \psi_2(t) \\ \vdots \\ 1 - \sum_{i=1}^{n_C - 1} \psi_i(t) \end{pmatrix} \tag{4.6}$$

Because values $\psi_i(t)$ are relative frequencies additional constraints have to be imposed on the choice of the function $\psi_i$. Let $t_*$ be the time for which a predicted is to be

obtained, the following should always hold.

$$\forall t \in \{t_1, \ldots, t_r, t_*\} \forall i \in \{1, \ldots, n_C\} \; : \; 0 \leq \psi_i(t) \leq 1 \tag{4.7}$$

$$\forall t \in \{t_1, \ldots, t_r, t_*\} \; : \; \sum_{i=1}^{n_C} \psi_i(t) = 1 \tag{4.8}$$

In practise, these constraints may turn out to be too strict. For illustration, in the case $p_i^k = p_i^{k+1} = p_i^{k+2} = 1$ and $p_i^j \ll 1$ for $j \notin \{k, k+1, k+2\}$ it may be difficult to find a continuous, low-complexity and non-volatile model for $\psi_i$. As consequence and under consideration that the only aim is to predict values for the period $t_*$ the following weaker constraints are given preference over the above:

$$0 \leq \psi_i(t_*) \leq 1 \tag{4.9}$$

$$\sum_{i=1}^{n_C} \psi_i(t_*) = 1 \tag{4.10}$$

Applying either constraint (4.8) or (4.10), renders the sub-models $\psi_i$ mutually dependent. Learning a model for the class label distribution thus requires to learn each sub-model $\psi_i$ taking into account the others. For instance, for determining the coefficients $\mathbf{a} := (a_0, \ldots, a_q)^T$ of $\psi_i(t) = \sum_{j=0}^{q} a_j \zeta_j(t)$, this turns the linear least squares problem (see Appendix C.2.1), into a constrained linear least-squares problem. There exist several methods from the field of optimisation for solving this sort of problem. They will not be discussed here in greater detail. For further reading see, for example, Gill et al. (1989). It still is an open research question how Gaussian processes can be learned under constraints such as the ones above.

If focussed on two-class problems only, learning a model for histories of class label distributions turns out to be more tractable. A look into (4.6) and (4.8), respectively (4.10), reveals that for $n_C = 2$ it is $\psi_2 = 1 - \psi_1$. This implies that if $\psi_1$ meets constraint (4.7), respectively (4.9), $\psi_2$ meets it too. Thus, only one model, e.g. $\psi_1$, needs to be learned. The methods discussed in Section 4.7.1 are applied to model $\psi_1$, respectively $\psi_2$. In the *PreDeT* algorithm the predicted class distribution then determines the class label assignment at a leaf node.

### 4.7.4 Putting the Parts Together

Having explained the main building blocks of how to predict future decision trees in the previous two sections, this one explains how they can be used in combination with a decision tree learner to anticipate future decision trees. This will eventually lead to the *PreDeT* algorithm.

Figure 4.9 shows the *PreDeT* algorithm. Similar to the vast majority of decision tree learners it consists of two consecutive stages. In the first stage (Lines 1–8) the split attribute for the current node is searched. In the second stage (Lines 9–18) it is decided whether the current node is a leaf (Line 9) or inner node (Line 15). Respectively, either a class label is assigned to the leaf node based on the majority class in this node, or the data sets are split according to the split attribute and the *PreDeT* algorithm continues

```
PreDeT((D^1, ..., D^r), t_*)
 1  I_best ← WORTHLESS
 2  for all untested attributes A
 3  do ℑ ← (I(D^1, A), ..., I(D^r, A))
 4     learn prediction model φ[ℑ]
 5     Ĩ ← φ[ℑ](t_*)
 6     if Ĩ > I_best
 7        then I_best ← Ĩ
 8             A_best ← A
 9  if I_best = WORTHLESS
10    then create leaf node v
11         P^k ← (p^k_{1·}, ..., p^k_{n_C·}),  k = 1, ..., r
12         learn prediction model ψ[(P^1, ..., P^r)]
13         (p̃*_{1·}, ..., p̃*_{n_c·}) ← ψ[(P^1, ..., P^r)](t_*)
14         assign c = argmax_{c_i}(p̃*_{1·}, ..., p̃*_{n_c·}) to v
15    else assign test on A_best to v
16         for all a ∈ dom(A_best)
17         do v.child[a] ←
18                PreDeT((D^1|_{A_best=a}, ..., D^r|_{A_best=a}), t_*)
19  return v
```

Figure 4.9: Outline of the $PreDeT$ algorithm

recursively (Line 17). It should be clear that the basic ideas laid out in Section 4.7.2 and Section 4.7.3 can be used in connection with any decision tree learner that uses attribute evaluation measures to determine splits.

In contrast to other decision tree learners $PreDeT$ takes as input a sequence of data sets $(D^1, \ldots, D^r)$ representing time periods $t_1, \ldots, t_r$ and the time $t_*$ for which a decision tree is to be predicted. It uses these data sets to predict the value of the attribute evaluation measure in the time period $t_*$ using a learned model $\varphi$ (Lines 4–5). The class label distribution within each data set is used to predict the class label distribution in time period $t_*$ using a learned model $\psi$ (Lines 11–13). Note that every decision about the structure of the tree – the choice of the split attribute in inner and of the class label in leaf nodes – is solely based on estimated future values of the used metrics. For this reason the tree learned by $PreDeT$ is a prediction of the decision tree in period $t_*$.

## 4.8 Conclusion

This chapter provided a first research step into the field of utilising change for classifiers. Taking the example of decision trees it presented the $PreDeT$ algorithm which predicts decision trees for a given time period by utilising models which describe the functional relationship between time and attribute evaluation measures and class label distributions, respectively. $PreDeT$ utilises the change embodied within the temporal dimension without explicitly producing knowledge about it. In doing so, it improves existing decision trees in terms of classification accuracy and noise robustness. For these reasons, the advocated approach belongs to the in Chapter 1 proposed novel third perspective on change. Appendix A (p. 101ff) is laying out the obtained experimental results; they are

promising as they show that the approach is able to learn decision trees with a higher classification accuracy than approaches which only use the most recent data available. Notably, part of the experiments only used a simple polynomial regression function to achieve those results.

The *PreDeT* algorithm has several distinctive properties which will be enumerated next and that are linked to the requirements stipulated in the problem statement in Section 4.2.

- **Consistency:** The trees are the same as the ones produced by established methods. This is a consequence of the temporal dimension having only influence on the decision made during decision tree induction; but not on the general structure of the tree itself. Indeed, the algorithmic skeleton of *PreDeT* is identical to the one used in many other decision tree learning algorithms. If *PreDeT* would be used with models which always return the most recent element of the attribute evaluation measure and class label distribution history, the *PreDeT* would resemble a typical windowing approach. Overall, *PreDeT* thus is consistent with change-unware decision tree learners and windowing approaches.
- **Usability:** To use *PreDeT* several parameters need to be fixed. First and foremost, it requires decisions on the length of the time periods covering the individual data sets and the actual point in time to be predicted. Nonetheless, these decisions are driven by the domain (e.g. intervals of data collection) and the problem to be analysed (e.g. time points of specific interest). Often, the corresponding parameters are not free of choice. Another parameter is the prediction model to be used for attribute evaluation measures and class labels. As discussed in Section 4.7.1, if having only vague domain knowledge, Gaussian process regression is the preferred choice, because it is non-parametric and is very flexible. Experiments on different data sets indicate that good predictions are obtained by using a squared exponential kernel. In sum, no additional parameters are required compared to established decision tree learners, thus *PreDeT* has the same level of usability.
- **Accuracy:** *PreDeT* learns at each node and leaf of the decision tree a model of the change in the data. Because each node represents a certain subspace of the data, *PreDeT* is capable of dealing with change that in its pace is heterogeneously distributed across the data, i.e. some regions of the data change faster than others. Because windowing approaches define only one window for the whole tree, they do not have this capability of a fine grained-adaption. Overall, this added flexibility leads to a higher accuracy of the learned decision trees. The experimental results in Appendix A.2 support this claim. They show that the improvement in classification accuracy depends on the chosen attribute evaluation measure and on the number of periods. For each of the data sets at least one combination of these parameters was found which leads to a statistically significant improvement. For the other settings no statistically loss in classification accuracy was observed.

At last, it must be noted that the described approach of process-centric change utilisation does not only apply to decision tree learners. It is suitable for other algorithms, for instance in the area of Bayesian networks. An example is the K2 algorithm. In general, every algorithm that builds upon a greedy strategy can be described as such a sequence of measure dependent decisions. More generally, the approach works wherever the mapping

of data onto models can be decomposed into more or less complex sequences of mappings with intermediate representations. The basic mechanism remains the same, as long as an intermediate representations is available that can easily be predicted and that fully determines the model.

Chapter 5

# Summary

> *"'Who controls the past,' ran the Party slogan, 'controls the future: who controls the present controls the past.' And yet the past, though of its nature alterable, never had been altered. Whatever was true now was true from everlasting to everlasting. It was quite simple."*
>
> George Orwell, 1984, Chapter 3

This thesis started with three observations concerning the current state-of-the-art in data mining. First, change is ubiquitous in any domain and is thus reflected in data sets collected over time. Collecting data over time is the norm in many applications, rather than the exception. Second, data mining handles changes following either one of two perspectives: ignoring change and analysing change. The first perspective is attributed to 'classical' data mining tasks, the latter leads to areas such as concept drift detection, contrast and change mining. Third, some data mining approaches are known for having long-standing shortcomings, for which no satisfying solution approaches exist yet.

Extending from these observations this thesis advocated a novel, third perspective on change in data mining, which fits into the middle of the previously mentioned ones: *utilising change*. This perspective treats change as yet another, but special feature of data, that helps to assist in the learning of patterns and models with the potential of lessening the shortcomings of the applied learning method. From a bird's eye view this novel perspective is justified by the fact that anything past, though not of primary interest, has a strong influence on how the present is and the future will be. Hence, when learning models or patterns in the present, to be applied in the future, the present state of a domain is equally as important as the past and the changes that led to the present.

The viability of this third perspective on change was demonstrated by introducing two novel methods: the discovery of temporally closed item sets as a change-aware condensed representation of item sets and the *PreDeT* algorithm which learns decision trees by anticipation. As detailed in the following, both methods meet the requirements that were set at the beginning of this thesis. Overall, the results testify that change utilisation is a promising new direction in data mining research.

## 5.1 Contributions

Because this thesis focused on both item sets and decision trees in the context of change utilisation, summary and main contributions are separately presented for each of these areas.

### Change Utilisation for Item Sets

As an example for change utilisation in the context of frequent item set discovery, temporally closed item sets were proposed. They target the shortcoming of frequent item set discovery to produce vast numbers of item sets. The improvement through change-utilisation therefore was measured by the number of produced item sets and the method compared to the change-unaware condensed representation of closed item sets.

As concerns the requirements from this thesis' objective in Section 1.3 the following can be stated:

**Improvement:** It was theoretically proven that temporally closed item sets are a subset of the closed item sets. In particular, this implies that their number is never larger than those of the closed item sets. Experiments confirmed that the number of the former can be significantly smaller than those of the latter. Even in cases where closed item set discovery fails, temporally closed item set discovery still leads to a reduction.

**Consistency:** Closed item set discovery is a special case of temporal item set discovery. In particular, temporally closed item sets extend the notion of redundancy used by closed item sets towards the temporal dimension. For these reasons, both approaches are comparable and their results are not contradictory.

The main contributions in Chapter 3 are the following:

- The concept of temporally closed item sets as a condensed representation of item sets which accounts for temporal redundancies in the data.
- A proof that temporally closed item sets are a subset of the closed item sets. The concept thus bridges the gap between 'classical' data mining and change mining approaches.
- A method that allows for faster creation of item set histories compared to the 'matchmaking' approach suggested in other publications.
- An algorithm CHARM-T for mining the set of temporally closed item sets from a transaction data set.
- The general framework of 'data-centric change utilisation' which focuses on enhancing data with specifically encoded temporal information.

### Change Utilisation for Decision Trees

As an example for change utilisation in the context of decision trees, the *PreDeT* algorithm which anticipates decision trees was proposed. It targets the shortcoming of decision tree induction that in a changing domain decision trees are already outdated as soon as they are produced and then degrade in their performance as time proceeds. The improvement through change utilisation therefore was measured by the classification accuracy and the algorithm compared to a standard decision tree learner.

As concerns the requirements from this thesis' objective in Section 1.3 the following can be stated:

**Improvement:** Experiments show that the decision trees produced by *PreDeT* may have a higher classification accuracy than trees produced by a change-unaware algorithm which operates on the most recent data. The achievable gain in classification accuracy depends on the choice of the attribute evaluation measure and the number of considered time periods. Nonetheless, independent of this configuration the experiments did not show a significant drop in classification accuracy. This means, in terms of classification accuracy *PreDeT* is as least as good as the change-unaware decision tree learner.

**Consistency:** The trees produced by *PreDeT* have the same structure as the trees of any other decision tree learner: the inner nodes represent decisions on attribute values and the leaf nodes represent class labels. This is because *PreDeT* utilises change only during the process of decision tree induction, for the prediction of attribute evaluation measures and class label distributions. Due this structural consistency, advantages of decision trees, such as their comprehensibility, are kept.

The main contributions in Chapter 4 are the following:

- The *PreDeT* algorithm which predicts decision trees for future time periods by modelling how attribute evaluation measure and class label distribution evolve over time.
- To the author's knowledge the first combination of Gaussian process models with decision tree and also one of the first approaches for accessing the change of decision trees over multiple time periods.
- The general framework of 'process-centric change utilisation' which focuses on the analysis of changes in the decisions made during model induction.

Some more contributions concern the data mining community and those people who are interested in conducting research in change mining or change utilisation:

- Chapter 2 is to the author's knowledge the first survey covering the current state-of-the-art of methods for change analysis in data mining, with an emphasise on change mining.
- Appendix A contains the reference to and the description of two publicly available, real-world and real-size data sets which have been collected over multiple years. In particular change mining research so far only uses non-disclosable, artificial or small sized data sets, due to a lack of appropriate public data.

## 5.2 Future Directions

Research on change utilisation still is in its early stages and consequently there are many open questions which still need to be addressed and answered. The following two lists identify several areas that the author believes merit future work which would significantly enhance the capabilities of the methods this thesis proposes. The last paragraph addresses the general prospects of change utilisation with regard to other data mining tasks.

### Change Utilisation for Item Sets

- Temporally closed item sets require a rule set as their supplemental structure. Section 3.6.2 showed that these rules can be interpreted as conditional probabilities which are constant over time. The supplemental structure may be further reducible by considering probabilistic conditional independence between items in combination with probabilistic inference based on, for example, the graphoid properties known from the theory of graphical models (see Borgelt and Kruse, 2002).
- The use of mutual information in combination with the information theoretic interpretation of temporal derivability (see Section 3.6.2) may be extendable to a general approach for measuring the conciseness of condensed representations by description length rather than by mere item set counting.
- Because sometimes association rules are generated from frequent item sets it would be interesting to investigate how the concept of temporal closedness carries over to them with the aim of developing a notion of 'temporally non-redundant' association rules.

### Change Utilisation for Decision Trees

- To decide upon the best split attribute *PreDeT* models the history of each attribute evaluation measure separately, thus makes separate predictions and then uses these predictions for a decision. It would be interesting to see whether a single model could be created which takes as input the attribute evaluation measure histories of all attributes and then directly delivers a decision for the split attribute. Since less models are learned, more data would be available for the single model. This could have the potential of enhancing the reliability of the method.
- A by-product of Gaussian process models is a Gaussian probability distribution for the (predicted) attribute selection measure at a certain point in time. The confidence intervals derived from this distribution serve as an indicator for the certainty that in the future an attribute indeed separates classes as strong as expected by the actual attribute selection measure. This could be utilised to develop a pruning strategy for the trees produced by *PreDeT*. As an idea, a subtree is pruned, if aggregated over all its nodes, the confidence in the respective evaluation measures is lower than the confidence in those of the root node.
- Decision tree learners are known to be highly susceptible to data perturbation. The tree undergoes radical changes in response to minor changes in the training data, rather than being stable. This constitutes a practical problem, because it contradicts human intuition to have entirely different sets of classification rules from statistically equivalent data sets. Finding a method to construct stable trees is seen as a pressing research issue in data mining (Wu et al., 2008). One step in this direction could result from considering that the prediction models used within *PreDeT* in a sense 'average out' the influences of noise and thus may lead to trees which are less susceptible to it.

Last but not least, while this thesis focused on classification and frequent pattern discovery, change utilisation is not limited to these. Investigating the application of change

utilisation to other data mining tasks, or particular data mining methods they embrace, therefore offers a virtually unlimited number of starting points for future research efforts. One data mining task to start with is cluster analysis, because particularly in higher dimensional data sets it still has many shortcomings, such as a lack of methods to differentiate between 'good' and 'bad' clusterings, or between noise and real structure. Evidence that this direction also delivers what it may promise was provided in a seminal work by Höppner and Böttcher (2007) who employed change utilisation to cater for the latter of these shortcomings.

# Appendix A

# Experimental Results

This chapter provides a detailed documentation of the experiments carried out with CHARM-T and *PreDeT*.

## A.1 Utilising Change for Item Sets

The CHARM-T-Algorithm addresses the item set quantity problem by reducing the number of produced item sets through facilitation of temporal redundancies. It builds upon the notion of temporally closed item sets which is derived from the well-known concept of closed item sets. CHARM-T itself extends the closed item set miner CHARM. Temporally closed item sets provable are a subset of closed item sets. Applying CHARM-T to a sequence of data sets yields a smaller number of item sets than the application of a closed item miner to their union. In this regard CHARM-T outperforms CHARM to a significant extent, and the following sections present the corresponding experimental evidence.

### A.1.1 Description of Data Sets

Two different data set are employed to evaluation CHARM-T, both being large real-life data sets collected by public authorities. The data sets and the pre-processing steps applied to them are detailed in the remainder of this section.

#### IPUMS Data

The IPUMS project[1] is dedicated to collecting, harmonising and freely distributing census data. The subsequently described experiments use an extract from IPUMS which is restricted to data from the US states New Jersey, New York and Pennsylvania and the census years 2001–2006. From the available attributes a subset of 14 was chosen (plus the census year). The attributes describe the interviewee (e.g. age, race, sex), the housing conditions (e.g. number of bedrooms, year of built), and the person's profession (e.g. weekly travel time to work, avg. hours worked per week, net income).

Numeric attributes were converted into nominal ones. The domain size of the attributes varies between 2 (e.g. for sex) and 9 (for age). Splitting the data set year-wise results in a sequence of six data sets each containing on average 226,082 records. Table A.1 lists the attributes used in the experiments and provides details on the pre-preprocessing that has been applied to numerical ones.

[1] http://usa.ipums.org/usa

| Data Set | Attributes |
|---|---|
| IPUMS | rooms (0, 4, 5, 6, 7, 8, 9); builtyr (1940, 1950, 1960, 1970, 1980, 1990); homestr; bedrooms (0, 1, 2, 3, 4, 5); fuelheat; vehicles; age (0, 10, 20, 30, 40, 50, 60, 70, 80); sex; marstat; rracwht; school; hourswrk (0, 10, 20, 30, 40, 50); income2 (0, 10000, 20000, 40000, 70000); travtime (0, 1, 11, 21, 31) |
| Road Accident | A1.2 (Police Force Code); A3 (Accident Severity); A1.5 (Number of Vehicles); A1.6 (Number of Casualties); ACCMTH (Month); A7 (Day of week); A8H (Hour of Accident); A1.12 (1st Road Class); A1.14 (Road Type); A1.15 (Speed Limit); A1.16 (Junction Detail); A1.17 (Junction Control); A1.18 (2nd Road Class); A1.20A (Pedestrian Crossing - Human Control); A1.20B (Pedestrian Crossing - Physical Facilities); A1.21 (Light Conditions); A1.22 (Weather Conditions); A1.23 (Road Surface Conditions); A1.24 (Special Conditions at Site); A1.25 (Carriage Way Hazards) |

Table A.1: Attributes of the data sets. For the IPUMS data the (lower-inclusive) interval borders used for discretisation are given in brackets. For the road accident data the attributes are listed by their original name, followed by their description in brackets.

#### Road Accident Data

The Road Accident Data is available from the UK data archive[2]. It contains details for all road accidents in the UK that led to human death or personal injury, notified to the police within 30 days of occurrence and involving one or more vehicles. The data is annually published for the previous year's accidents. For each year the data consists of three parts distributed in individual files, providing details concerning the accident itself (e.g. road and weather conditions, number of vehicles and casualties), the casualties (e.g. sex, age) and the involved vehicles (e.g. type and damaged parts), respectively.

The data for the experiments is based on the accident part of the data and covers the publication years 2000–2009. Table A.1 details the considered subset of attributes. It constitutes of those attributes that over the years only had minor changes in the meaning of their values and thus required only a minimum on harmonisation effort. The documentation which accompanies the data details what has changed with respect to the previous year. The attributes all have a symbolic (sometimes ordered) value range, such that no discretisation is necessary. Overall, this results in a sequence of ten data sets having an average size of 202,008 records.

### A.1.2 Experimental Setup

The question to be answered is how much the set of temporally closed item sets is smaller than the set of closed item sets and how both compare to the set of frequent item sets.

[2]Department for Transport. Road Accident Statistics Branch, *Road Accident Data, 2000–2009* [computer files]. Colchester, Essex: UK Data Archive [distributor]. GN: 33267, http://discover.ukdataservice.ac.uk/series/?sn=2000045

To produce temporally closed item sets, CHARM-T was implemented in C++ on basis of the original CHARM source code kindly provided by Zaki. This implementation of CHARM was also used to produce the closed item sets. The frequent item sets are produced using the *Eclat* algorithm (Zaki, 2000), because CHARM originates from it and its implementation still is contained within the previously mentioned CHARM software package.

The three software tools need as input a transaction set. For each element in the IPUMS and Road Accident sequence of data sets a transaction set was constructed by encoding every (attribute, attribute value) combination as an item represented by a unique number. This results in two sequences of transaction sets, each stemming from one of the two data sources. CHARM and Eclat receive as input the union of the transaction sets, whereas CHARM-T receives the temporally augmented transaction set, constructed according to Definition 3.6 (p. 47). That is, the union of the transaction sets in the sequence enriched by artificial items representing each transaction's time.

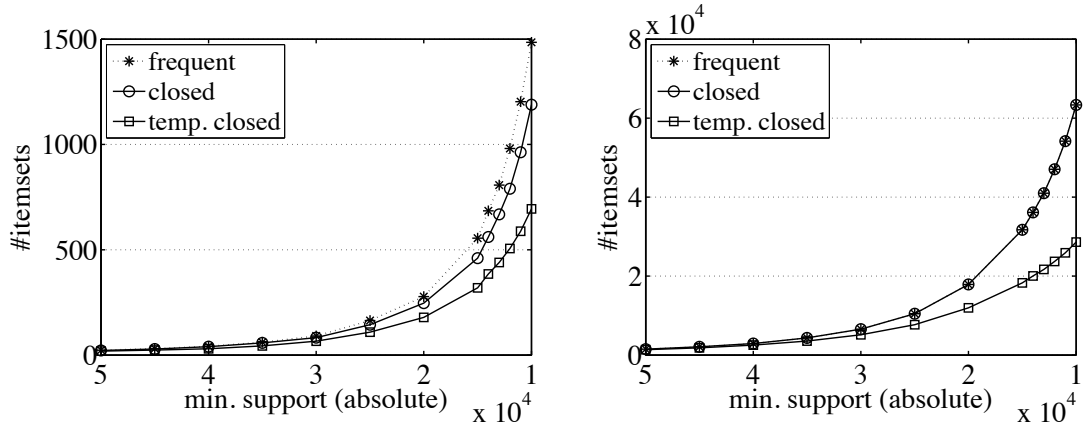
Each algorithm (CHARM-T, CHARM, Eclat) was run using different (absolute) support thresholds. For CHARM-T these thresholds are $\sigma_a = 10000, 11000, \ldots, 15000, 20000, \ldots, 100000$ representing the minimum number of transactions allowed in a time period. Because CHARM and Eclat are unaware of time periods, for their runs these thresholds must be scaled up to be relative to the whole data set, and for this reason are multiplied by the number of time periods in the respective data set. It should be noted that this approach is heuristic based on the assumption that each time period contains a similar number of transactions and that there are no drastic support differences across time periods for an item set.

For each run of CHARM-T, CHARM and Eclat the number of returned item sets is counted. For CHARM-T two additional measurements are taken: the factor $\epsilon$ (see Definition 3.8), which maps its history to the smallest temporally closed item set derivable from it, and the test statistic $G^2$ (see Theorem 3.7), which describes the similarity of both histories.

### A.1.3 Experimental Results

Figure A.1(a) shows the experimental results for the IPUMS data and Figure A.1(b) those for the Road Accident data. Each figure shows the number of returned temporally closed, closed and frequent item sets for different absolute support thresholds. As can be seen, the approach of temporally closed item sets leads to a significant reduction in the number of item sets compared to the other item set representations. Picking the runs with an absolute support threshold of $\sigma_a = 10000$, while mining only for closed item sets reduces the IPUMS result set's initial size to roughly 80% and those of the Road Accident result set not at all (i.e. 0%), the temporally closed item set approach reduces it to 46% and 45%, respectively. This means, for the IPUMS data the set of temporally closed item sets is by a factor of 1.7 smaller than the set of strictly closed item sets. For the Road Accident data this factor is with 2.2 even better.

Figure A.3 shows how the factor $\epsilon$ is distributed which maps the history of a non-temporally closed item set to the smallest temporally closed item set derivable from it. The figure only shows non-temporally closed but not closed item sets, i.e. it leaves out those with $\epsilon = 1$. For both data sets the range of $\epsilon$ is spread over its complete value range




(a) IPUMS (b) Road Accident

Figure A.1: Number of discovered frequent, closed and temporally closed itemsets

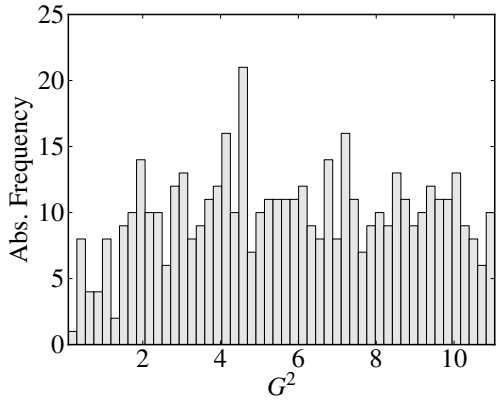
of $(0, 1)$. Both histograms exhibit a pronounced peak at $\epsilon \approx 1$ which indicates the rough number of item sets that could have been non-closed, but are not, due to some noisy transactions, for instance. If the strict equality comparison in the definition of closed item sets would be relaxed to account for noise, these item sets would be non-closed, thus not appear in the result set and eventually leading to the respective data having a considerable smaller set of closed item sets produced from it than it has now. CHARM-T, in contrast, discards each item set the histogram represents and thus reduces the number of closed item sets even further.

For the Road Accident data, the condensed representation approach of producing closed item sets fails. There are no non-closed item sets and consequently the sets of closed and frequent item sets are identical. This illustrates and practically underlines the in Section 3.4.5 (p. 42) issued criticism on condensed representations which rely on strict equality comparisons: One erroneous transaction can turn, for instance, an actually non-closed item set into a closed one. Indeed, the bar on the far right in Figure A.3(b) shows that the number of such 'almost closed' item sets takes on an extremely large value.

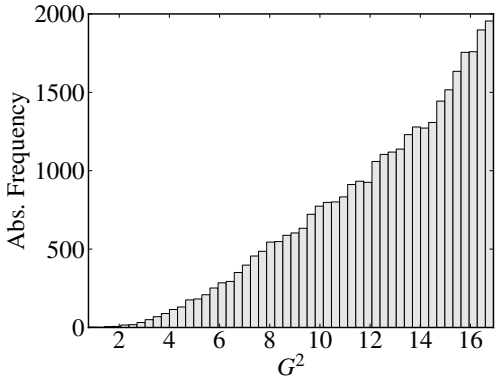
Figure A.2 shows the distribution of the $G^2$ metric which expresses the similarity between the histories of a non-temporally closed item sets and the smallest temporally closed item set derivable from it. The smaller $G^2$ the more similar the histories are. Contrasting Figure A.2(a) and Figure A.2(b) shows that for the IPUMS data $G^2$ is rather evenly distributed while for the Road accident data the number of non-temporally closed item sets in each histogram bin increases monotonically with decreasing dissimilarity. This difference is explained by the Road accident data being made of more time periods that cover a longer time range than the IPUMS data. For the Road accident data it is thus unlikelier to have properties that have stayed strictly invariable over the full time range; but only such properties entail very similar histories (see Section 3.6.1 and Section 3.6.2).

Note, that $G^2$ and $\epsilon$ both measure different characteristics of the same relation between a non-temporally closed item set and a temporally closed one; the first one measures the similarity and the second one the 'distance' between the two corresponding histories. Intuitively, one would assume that both are correlated: the farther one history is away

from the other by means of support, the likelier they are dissimilar. However, looking for both data sets at the scatter plot of $\varepsilon$ against $G^2$, displayed in Figure A.4, no such correlation is apparent. This means, first of all, non-temporally closed item sets are more than just a 'noise-tolerant' version of non-closed item sets and, second, both data sets contain a considerable number rules with confidences significantly lower than 1 and these confidences are over time (almost) constant. (see Section 3.6.2).

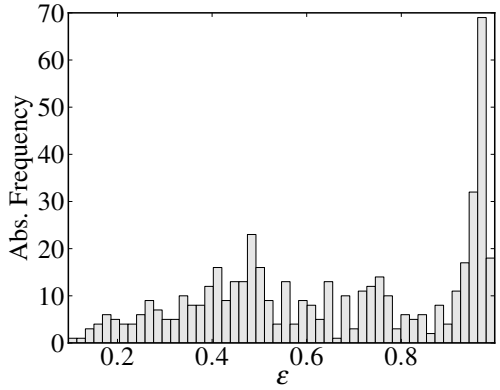



(a) IPUMS ($\sigma_a = 10000$)

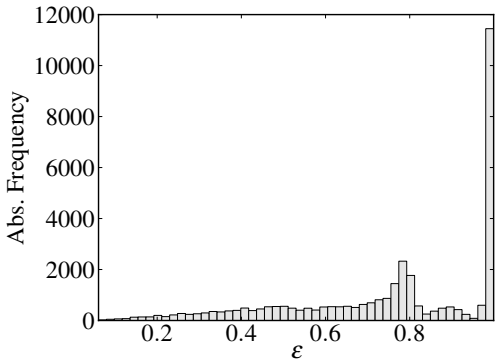



(b) Road Accident ($\sigma_a = 10000$)

Figure A.2: Histogram of the similarity $G^2$ between the histories of non-temporally and corresponding temporally closed item sets

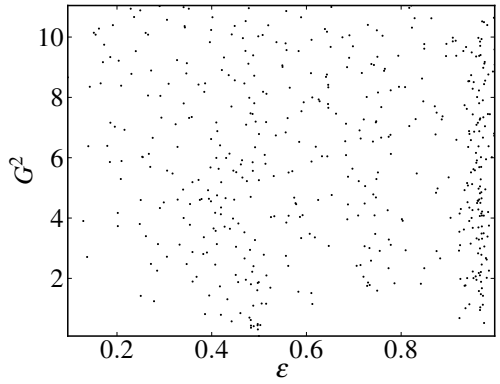



(a) IPUMS ($\sigma_a = 10000$)

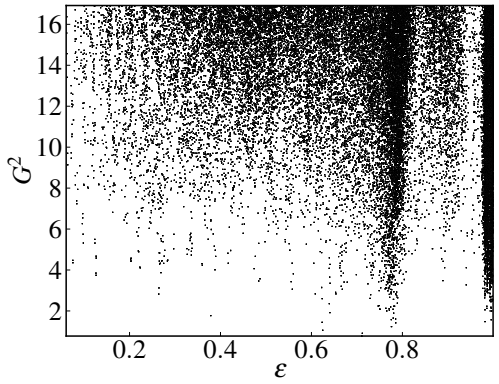



(b) Road Accident ($\sigma_a = 10000$)

Figure A.3: Histogram of the distance $\varepsilon$ between the histories of non-temporally and the corresponding temporally closed item sets




(a) IPUMS ($\sigma_a = 10000$)




(b) Road Accident ($\sigma_a = 10000$)

Figure A.4: Scatter plot of distance $\varepsilon$ against similarity $G^2$ between the histories of non-temporally and the corresponding temporally closed item sets

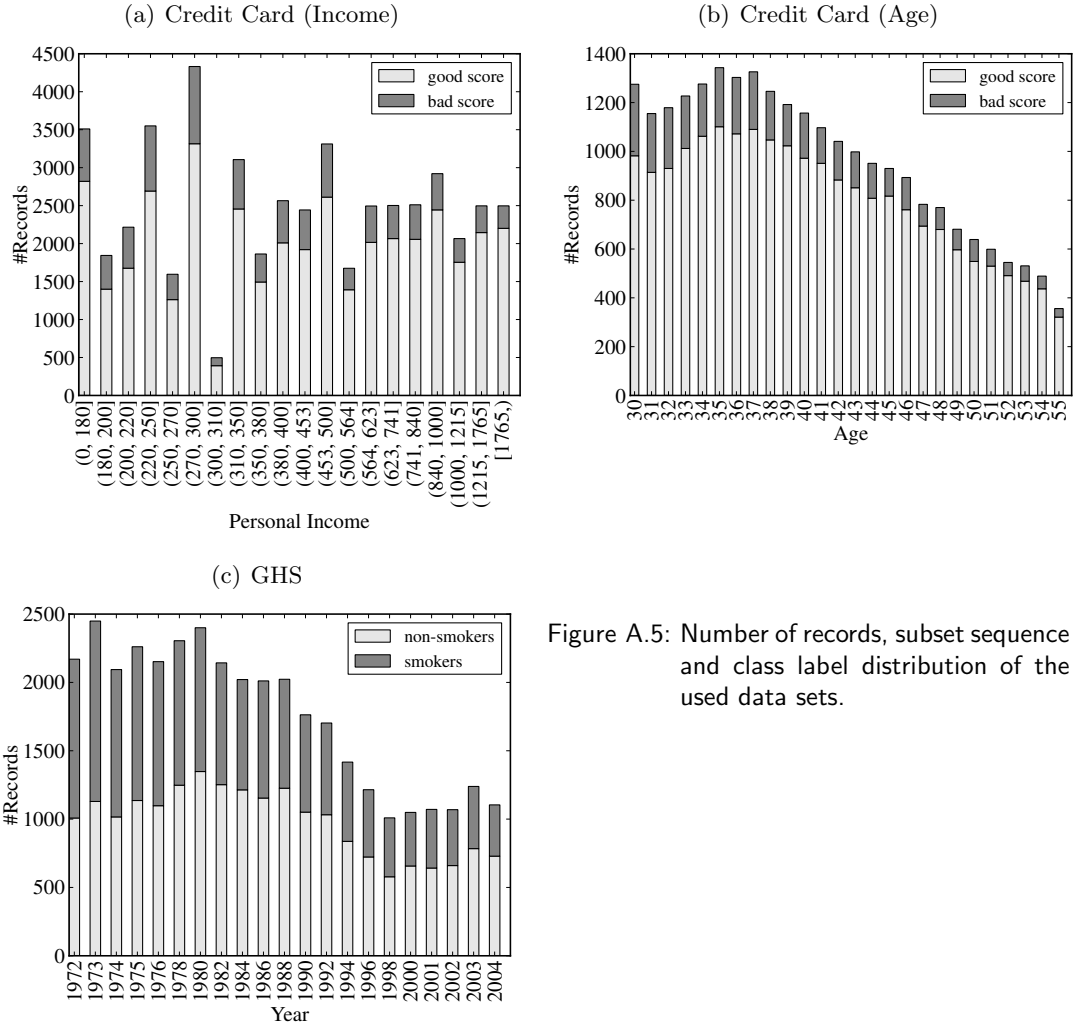



Figure A.5: Number of records, subset sequence and class label distribution of the used data sets.

## A.2 Utilising Change for Classifiers

The aim of the *PreDeT* algorithm is to improve on the classification accuracy of decision trees by utilising the temporal dimension and the change it represents. This section presents the results of experiments which show that decision trees produced by *PreDeT* statistically significant outperform those induced by a 'conventional' decision tree learner which ignores the temporal dimension.

### A.2.1 Description of Data Sets

Three different data sets are employed to evaluate the *PreDeT* algorithm. All of them being real-life data sets that have been collected by corporations and public authorities. The data sets, their associated classification task and pre-processing steps applied to them are laid out in the following.

| Data Set | Attributes |
|---|---|
| Credit Card (Income/ Age) | *age*, area_code_residencial_phone, flag_card_insurance_option, flag_fathers_name, flag_mothers_name, flag_other_card, flag_residence_state=working_state, flag_residence_town=working_town, flag_residencial_address=postal_address, flag_residencial_phone, marital_status, months_in_residence, months_in_the_job, payment_day, *personal_net_income*, quant_additional_cards_in_the_application, residence_type, sex, **target_label_bad=1** |
| GHS | degree, page, **pcigsmk1**, pcob1, pdvilo3a, pdvmardf, phrpseg3, pn0to4, pn5to15, pnumveh, psex, ptenure, *year* |

Table A.2: Attributes used for the experiments. A bold font indicates the target attribute and an italic font indicates the attribute used for splitting the data.

### CRM Data

This representative data set is from the domain of Customer Relationship Management (CRM). It contains answers of customers to a survey conducted by a telecommunications company over a period of 25 weeks. Each sample is described by 13 nominal attributes with a domain size between 2 and 9. One attributes describes the week in which the survey data was collected.

The classification task is to predict whether a customer is satisfied or dissatisfied with the services he uses based on the remaining 12 attributes, i.e. the data set has two classes to predict.

In order to have an equal class distribution the original data set has been balanced by removing samples of satisfied customers. It was then split into 25 subsets $D^i$, each corresponding to a time period of one week. The subsets contain between 243 and 399 samples. Unfortunately, disclosure of further details regarding the data set's attributes and class label distribution is not possible for reasons of data protection.

### Credit Card Data

The 2009 PAKDD Data Mining Competition[3] published this data set. A bank collected the data from people to which it granted a credit card. Its attributes detail the customer's demographic and economic background, such as their age and income. Table A.2 shows the attributes that are used for the experiments.

The classification task (the same the competition focused on) is to determine whether a customer will default in paying his/her credit card bill. Each record in the data set is labelled according whether the customers is deemed being a 'good' or a 'bad' one. The

[3] http://sede.neurotech.com.br/PAKDD2009/

published solutions to the competitions indicate that finding a good classifier is tough, even sophisticated models only attain a apparently low performance.

The credit card data set lacks a real time attribute, maybe the data has not been gathered over time. To nevertheless create a sequence of data sets, the mandatory input to *PreDeT*, two alternative split attributes are available: age and income. This way, the credit card data set yields two distinct subset sequences for *PreDeT*, in the following referred to as *credit card (income)* and *credit card (age)*. Their size and class label distributions are shown in Figure A.5(a), respectively Figure A.5(b). The class label distribution is in all subsets of each subset sequence highly imbalanced, because regardless of income and age defaulting customers are rare. This make it necessary to grow decision trees so deep that only a few records fall into the leaf nodes. As a consequence the history of the class label distributions in these nodes will expose dramatic changes even if only one record's class label would be changed. Adequately modelling these histories thus is hard, if not impossible. For this reason, the experiments with *credit card (income)* and *credit card (age)* do not utilise predictions of class label distributions but take the majority class of the most recent subset.

#### GHS Data

The General Household Survey (GHS) time series data set is available from the UK data archive[4]. It is a continuous, annual survey of people living in private households in the UK which comprises information regarding household, family, employment, education and health, and many more topics. The data set covers the years 1972 to 2004 and includes only those attributes that have not seen major changes in their corresponding survey questions over the years. Table A.2 shows the attributes that are used for the experiments.

The classification task is to determine whether a young adult (that is, a person in the age of 18 to 24) currently smokes, or never has smoked before.

Not all available attributes where used, but focused on those which are available for most of the years and that appear decisive concerning the classification task. Records with missing values in these attributes were removed and the data set split year-wise in 21 subsets. Their size and class label distribution is shown in Figure A.5(c). Some years are missing, even though they are contained in the raw data. The reason is that questions related to smoking behaviour have not been asked in these years.

### A.2.2 Experimental Setup

The goal of the experiments is to demonstrate on the example of decision trees that utilisation of changes yields a gain in classification accuracy. For this reason, the trees produced by *PreDeT* are compared to trees produced by a benchmark decision tree learner that ignores the temporal dimension.

[4]Office for National Statistics. Social and Vital Statistics Division, *General Household Survey: Time Series Dataset, 1972-2004* [computer file]. Colchester, Essex: UK Data Archive [distributor], July 2007. SN: 5664, http://dx.doi.org/10.5255/UKDA-SN-5664-1

In order to make a fair and realistic comparison three arguments need to be considered.

First, when learning decision trees in the presence of concept drift it is common practise to use only the most recent data available because their characteristics are very likely to be best reflecting those of (unknown) near future data. It has been demonstrated by several authors that such a *temporal moving window approach* almost always outperforms trees which have been learned from all of the available data (cf. Section 4.5)

Second, from an abstract perspective $PreDeT$ (implicitly) learns a sequence of $r$ decision trees each corresponding to a data set $D^j, j = 1, \ldots, r$ and then anticipates the tree in the future period $r + 1$ using a prediction model (cf. Sections 4.7.4 and 4.6). As with any prediction model, the obtained prediction cannot have a better quality than its inputs. The quality of a decision tree, in particular its generalisation ability, does strongly depend on the size of the data set used for training. From this it follows that the trees anticipated by PreDeT do have a similar quality to trees that would have been learned directly on a data set with a size similar to those of each $D^j, j = 1, \ldots, r$.

Third, a fair comparison requires that $PreDeT$ and the benchmark decision tree learner differ only in that one utilises time and the other one not. Apart from that they should be structurally similar. Otherwise, there would be the risk whether observed changes in accuracy are either due to tweaks in the benchmark learner or indeed attributable to the utilisation of change.

For these reasons, $PreDeT$ is compared with the decision tree learner which results from setting in the algorithm in Figure 4.9 the models $\varphi$ and $\psi$ for the attribute evaluation measure, resp. class label distribution, to $\varphi[(I(D^1, A), \ldots, I(D^r, A))](t_*) := I(D^{r+1})$ and $\psi[(P^1, \ldots, P^r)](t_*) := P^r$. With this configuration the tree induction process uses only the most recent data set of each sequence; the algorithm resembles a conventional decision tree learner, like the one in Figure 4.3. The accuracy of the resulting decision trees are used as the benchmark for the trees produced by $PreDeT$.

The previous section described three raw data sets and how from them four temporally ordered sequences of (sub-)data sets $(D^1, \ldots, D^s)$ are created. They have lengths $s = 25$ for the CRM, $s = 21$ for the GHS, $s = 20$ for the Credit Card (Income) and for $s = 26$ the Credit Card (Age) data. Each *experiment* uses a sub-sequence of $r$ consecutive data sets $(D^i, \ldots, D^{i+r-1})$ within the available $s$ ones. For each $i$, $i = 0, \ldots, s-r$ the $PreDeT$ algorithm learns a decision tree and obtains classifications for the samples in the data set $D^{i+r}$ that chronologically follows the sub-sequence. The data set $D^{i+r}$ thus serves as the test data set.

Consider, for instance, the GHS data. Section A.2.1 outlines how this data yields a sequence of $s = 21$ sub-data sets. Choosing an experimental set-up $r = 5$ gives 16 sub-sequences. For each sub-sequence $PreDeT$ and the benchmark decision tree learner induce a decision tree each. Overall, this amounts to 16 classification accuracy comparisons. This experimental setup is the same for information gain ratio and the information gain. For $r = 10$ and $r = 15$ only the number of classification accuracy comparisons would be less (11, respectively 6), but the general approach stays the same.

$PreDeT$ primarily depends on two factors: the attribute evaluation measure and the number $r$ of past periods taken into account (i.e. the length of the sub-sequence). Therefore, the experiments are carried out using the information gain ratio $I_{gr}$ and the inform-

ation gain $I_{gain}$. For the length $r = 5, 10, 15$ is used. Furthermore, for the CRM data $PreDeT$ internally uses regression polynomials in combination with the Akaike criterion to produce prediction models, whereas all other data sets use Gaussian process regression. As the kernel for the Gaussian process models the sum of a squared exponential kernel with a noise kernel was chosen, as it yielded the best results compared to, for example, neural network and Matern kernels. The reason that regression polynomials are only used for the CRM data set has historic reasons. For early versions of $PreDeT$ only this data set was available. Later, Gaussian process regression was integrated into PreDeT, but at this point the CRM data set was not accessible anymore due to its confidentiality. The other data sets were also tested using polynomial regression with the Akaike criterion, but the resulting trees had a similar classification accuracy to the trees induced by the benchmark learner.

### A.2.3 Experimental Results

Figure A.6 reports the experimental results in several charts organised in rows and columns. Each row refers to one data set, such as GHS, Credit Card (Age) or Credit Card (Income). Each column corresponds to an attribute evaluation measure; information gain on the left, and information gain ratio on the right. Each chart displays a *box plot* for each $r = 5, 10, 15$. The ordinate axis shows the difference in classification accuracy between $PreDeT$ and the benchmark decision tree learner. A difference greater than zero expresses that $PreDeT$ performs better than the benchmark decision tree learner. In turn, a difference lower than zero expresses that the benchmark learner outperforms $PreDeT$. The underlying classification accuracies are measured in percent; hundred percent thus mean that no misclassifications occur.

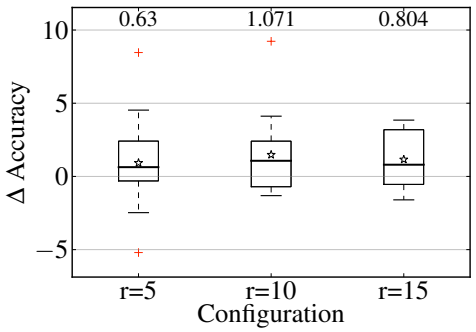
The *box plots* are simple to understand: The central bold line is the median of the accuracy differences. Its numerical value is also shown on top of the chart above each box plot. In addition, the mean is shown as a star-like symbol. The horizontal line above and below the median represent the upper and lower quartile. This means, 25 percent, accuracy differences fall above, respectively below these quartile lines. The so-called whiskers, which extend from the quartile lines, indicate the overall range of the accuracy differences. The cross symbols above and below the whiskers represent outliers.

If the claim is true that utilisation of the temporal dimension yields a gain in classification accuracy, then the median of the accuracy differences must be significantly greater than zero in the majority of experiments. In those cases $PreDeT$ outperforms the decision tree learner on more than fifty percent of the sub-sequences. As the Figure A.6 shows, this is the case in 21 experiments, in two experiments the median is lower (Figure A.6(c) for $r = 15$ and Figure A.6(d) for $r = 10$), and in one there is a tie (Figure A.6(h) for $r = 15$).

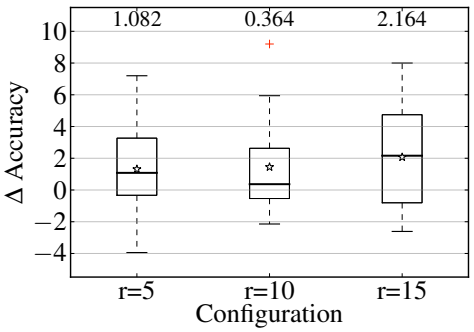
To assess the statistical signicance of the observed gains in classification accuracy, a (one-tailed) Wilcoxon's signed ranks test (Wilcoxon, 1945) is carried out for the results of each experiment in order to gain a p-value for the following hypothesis:

$H_0$ : The difference $\Delta$ = Accuracy(PreDeT) - Accuracy(DTree)
has a median value lower equal zero.

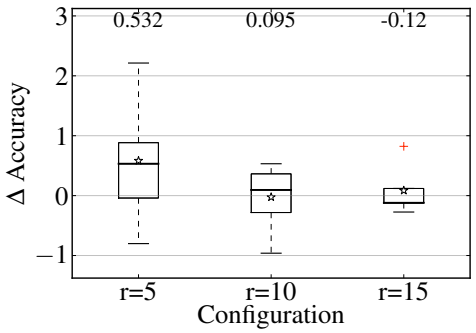
$H_1$ : The median value of the difference $\Delta$ is greater than zero.



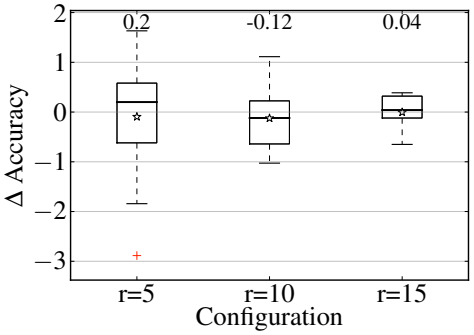

(a) CRM/ Information Gain



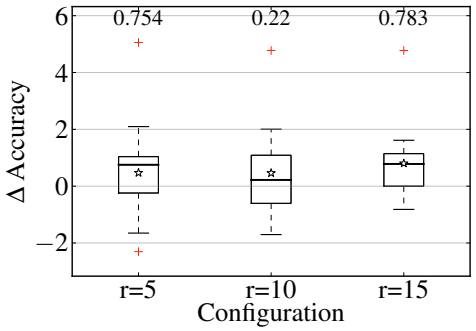

(b) CRM/ Information Gain Ratio



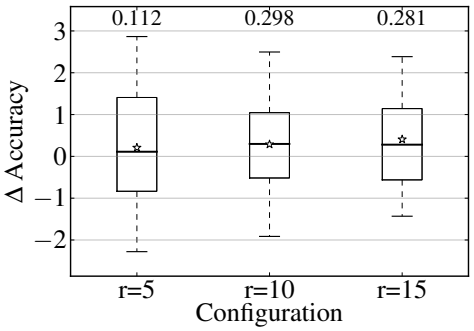

(c) Credit (Income)/ Information Gain



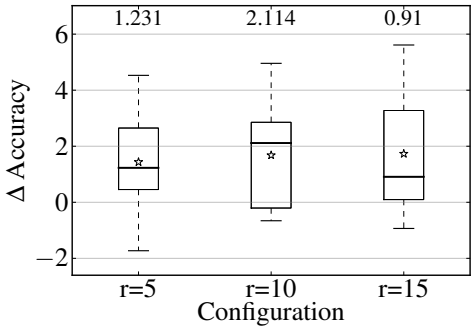

(d) Credit (Income)/ Information Gain Ratio



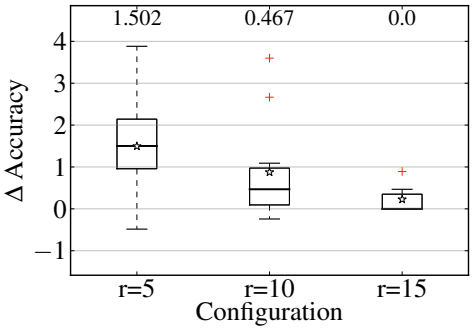

(e) Credit (Age)/ Information Gain




(f) Credit (Age)/ Information Gain Ratio




(g) GHS/ Information Gain




(h) GHS/ Information Gain Ratio

Figure A.6: Box plots of the gain (or loss) in classification accuracy when using $PreDeT$ compared to a Decision Tree learner

(a) Information Gain

| | $r = 5$ | | $r = 10$ | | $r = 15$ | |
|---|---|---|---|---|---|---|
| | p-value | W (n) | p-value | W (n) | p-value | W (n) |
| CRM | 0.0909 | 61 (19) | **0.0365** | 28 (15) | 0.0820 | 10 (9) |
| Credit Card (Income) | **0.0177** | 23 (15) | 0.4551 | 21 (9) | 0.5934 | 9 (5) |
| Credit Card (Age) | 0.1519 | 85 (21) | 0.1796 | 43 (15) | **0.0273** | 6 (9) |
| GHS (Year) | **0.0006** | 18 (16) | 0.0210 | 10 (11) | 0.0781 | 3 (6) |

(b) Information Gain Ratio

| | $r = 5$ | | $r = 10$ | | $r = 15$ | |
|---|---|---|---|---|---|---|
| | p-value | W (n) | p-value | W (n) | p-value | W (n) |
| CRM | **0.0407** | 45 (18) | 0.0520 | 26 (14) | 0.0654 | 12 (10) |
| Credit Card (Income) | **0.5000** | 45 (13) | 0.6875 | 33 (10) | **0.500** | 7 (5) |
| Credit Card (Age) | 0.2810 | 98 (21) | **0.2019** | 51 (16) | 0.2065 | 23 (11) |
| GHS | **0.0002** | 3 (15) | 0.0068 | 6 (11) | – | 0 (2) |

Table A.3: Statistical significance of the results from Figure A.6 measured by a Wilcoxon-Signed Rank Test.

According to Demsar (2006) the Wilcoxon's signed ranks test is the recommended test for comparing two models. Table A.3 shows the resulting test statistics and their associated p-value for each experiment.

Regarding all experiments carried out using the information gain, for each data set at least one setting of $r$ can be found on which the null hypothesis is rejected with a statistical significance of 0.05. For the GHS data set even all p-values are below a significance level of 0.1, and still for the Credit Card (Age) data all p-values are moderately significant with a level of 0.2. Only the Credit Card (Income) data falls short of the other results. Even though it has a p-value of 0.0177 for $r = 5$, the p-values for $r = 10$ and $r = 15$ are large. Nonetheless, they indicate that *PreDeT* does not lead to worse results than the benchmark decision tree learner. The main reason is that income, in strong contrast to age, is only very weakly correlated with the class attribute. During decision tree learning this may lead to attribute evaluation measure and class label distribution histories which are unsteady and thus very challenging to predict.

Regarding the information gain ratio, the p-values for the GHS and CRM data are below a significance level of 0.01, respectively 0.1, leading to a clear rejection of the above null hypothesis. However, for Credit Card (Income) and Credit Card (Age) the p-values are worse than the ones for information gain; even the best one does exceed a significance level of 0.2. But still, the trees produced by *PreDeT* do not perform worse than the benchmark ones. One explanation for these results could be that the information gain ratio is more sensitive to noise, in the sense that minor changes in the data lead to drastic changes in information gain ratio and hence in histories of it, eventually making prediction less reliable. This claim is left unproved in this thesis and should the subject of future research.

In contrast to the CRM and GHS data, the credit card data does not contain a proper

time attribute, but instead uses age, respectively income, to 'simulate' it. Herein lies the reason why *PreDeT* performs worse on the credit card data sets compared to the other two. Consider the Credit Card (Age) data and the concept of a 'defaulting customer' which stems from the classification task. This concept does not always change smoothly and steadily over the years, but sometimes abruptly. The reason are life events which primarily occur at a certain age and that may dramatically change the possibility of a default. Such events are, for example, the first well-paid job after university, the purchase of a home and retirement. But there exist many more. As a result of these events, the attribute evaluation measure and class label histories for these data sets show many abrupt changes. These changes are difficult to model and almost impossible to predict, which eventually leads to *PreDeT*'s poor performance on the data. In general, this is a limitation of *PreDeT*. It is more suited for domains which exhibit steady and smooth change, and less suited for ones which suddenly change to a large extent.

Overall, the experiments indicate that utilising the temporal dimension during decision tree induction often leads to trees with a significantly higher accuracy than trees that do not utilise it. The experiments further show that even in the case of no improvement, the trees learned by *PreDeT* do not perform worse than conventional ones. The degree of improvement depends on the attribute evaluation measure and the choice of $r$. Generally, the exploitation of this and other dependencies between parameters leaves space for further optimisations of the *PreDeT* algorithm and should be a subject of future research.

## Appendix B

# Proofs

## B.1 Proof of Theorem 3.1

**Theorem 3.1.** *Let $D$ be a data set and $D' \subseteq D$, then it is $Closed(D') \subseteq Closed(D)$.*

*Proof.* It is $Closed(D') \subseteq Closed(D)$ if and only if $X \in Closed(D') \Rightarrow X \in Closed(D)$. In the following the contraposition $X \notin Closed(D) \Rightarrow X \notin Closed(D')$ is proven.

Suppose $X \notin Closed(D)$. By Definition 3.1 this is the case if

$$\exists Y : \ Y \supset X \ : \ \text{supp}(X) = \text{supp}(Y)$$

Using the definition of support and multiplication with $|D|$ yields

$$\exists Y : \ Y \supset X \ : \ |\{I \in D | X \subseteq I\}| = |\{I \in D | Y \subseteq I\}|$$

Using the notion $t(X, D) := \{I \in D | X \subseteq I\}$ to describe the subset of transactions in $D$ that contain $X$ this can be rewritten as

$$\exists Y : \ Y \supset X \ : \ t(X, D) = t(Y, D)$$

Independent of whether this proposition is satisfied, or not, it is always true that

$$\forall Y : \ Y \supset X \ : \ t(X, D') \supseteq t(Y, D')$$

because for $I \in t(Y, D')$ it follows from $X \subset Y$ that $X \subset I$ and then $I \in t(X, D')$.

Hence, it remains to be shown that whenever

$$\exists Y : \ Y \supset X \ : \ t(X, D) \subseteq t(Y, D) \tag{B.1}$$

it follows

$$t(X, D') \subseteq t(Y, D')$$

Assume that (B.1) is true and that $I \in t(X, D')$. Because $D' \subset D$ it is $I \in t(X, D)$. According to (B.1) there exists an item set $Y \supset X$ such that $I \in t(Y, D)$. Note that $Y$ does not depend on $D'$ but only on the choice of $X$ and thus could be the same for any subset of $D$. It is $D = D' \cup D \setminus D'$ and thus $t(Y, D) = t(Y, D') \cup t(Y, D \setminus D')$. Since it is $I \in t(Y, D)$ and also $I \in D'$ (due to $I \in t(X, D')$) it follows $I \in t(Y, D')$. □

## B.2 Proof of Theorem 3.2

**Theorem 3.2.** *Given a time-stamped data set $D$ and a corresponding partition of the time axis into periods $\hat{T} := \{T_1, \ldots, T_n\}$, the set of closed item sets contained in $D$ equals the set of item sets that are closed over the sequence $\hat{T}$, in other words $SeqClosed(D, \hat{T}) = Closed(D)$.*

*Proof.* We prove the above set equality by showing that each set is a subset of the other, i.e. $Closed(D) = SeqClosed(D, \hat{T})$ if and only if $Closed(D) \subseteq SeqClosed(D, \hat{T})$ and $SeqClosed(D, \hat{T}) \subseteq Closed(D)$.

1. It is $Closed(D) \subseteq SeqClosed(D, \hat{T})$ if and only if $X \in Closed(D) \Rightarrow X \in SeqClosed(D, \hat{T})$. This is equivalent to $X \notin SeqClosed(D, \hat{T}) \Rightarrow X \notin Closed(D)$.

   Suppose $X \notin SeqClosed(D, \hat{T})$. By Definition 3.5 this is the case if

   $$\exists Y : \ Y \supset X \ : \ \forall T_i : \ T_i \in \hat{T} \ : \ \mathrm{supp}_i(X) = \mathrm{supp}_i(Y)$$

   Using the definition of support yields

   $$\exists Y : \ Y \supset X \ : \ \forall T_i : \ T_i \in \hat{T} \ : \ \frac{|\{I \in D_i | X \subseteq I\}|}{|D_i|} = \frac{|\{I \in D_i | Y \subseteq I\}|}{|D_i|}$$

   By multiplying both sides with $D_i$ and summation over $T_i$ this implies

   $$\exists Y : \ Y \supset X \ : \ \sum_{i=1}^{n} |\{I \in D_i | X \subseteq I\}| = \sum_{i=1}^{n} |\{I \in D_i | Y \subseteq I\}|$$

   Because it is $D_i \cap D_j = \varnothing, \ i \neq j$ and $\cup_{i=1}^{n} D_i = D$ (see Section 3.3) this is equivalent to

   $$\exists Y : \ Y \supset X \ : \ |\{I \in D | X \subseteq I\}| = |\{I \in D | Y \subseteq I\}|$$

   Division by $|D|$ and consideration of the definitions of support and closed item sets (Definition 3.1) then yields $X \notin Closed(D)$.

2. Like in the first part of the proof, we prove the contraposition $X \notin Closed(D) \Rightarrow X \notin SeqClosed(D, \hat{T})$ .

   Using the same notation as in the proof of Theorem 3.1 this can be rewritten:

   $$\exists Y : \ Y \supset X \ : \ t(X, D) = t(Y, D)$$

   implies

   $$\exists Y : \ Y \supset X \ : \ \forall T_i : \ T_i \in \hat{T} \ : \ t(X, D_i) = t(Y, D_i)$$

   Theorem 3.1 already states that if $X$ is non-closed in $D$ it is also non-closed in any subset and thus also in $D_i$. For this reason, it only remains to be shown that the superset $Y$ of $X$, which renders $X$ non-closed, is the same for any $D_i$.

   This is proven by contraposition. Assume that there is an item set $Xx_l \in D_l$ such that $t(X, D_l) = t(Xx_l, D_l)$ and an item set $Xx_k \in D_l$ such that $t(X, D_k) = t(Xx_k, D_k)$ with $x_l \neq x_k$ and $X \cap \{x_k, x_l\} = \varnothing$. Further, assume that there is no item $x \in L \setminus X$ such that $t(X, D_l) = t(Xx, D_l)$ and $t(X, D_k) = t(Xx, D_k)$. In

words, $X$ is non-closed in time periods $T_k$ and $T_l$ but there is no common superset of $X$ which is responsible for their non-closedness. Based on these assumptions, there must be transactions $I_l \in t(X, D_l)$ with $x_k \notin I_l$ and $I_k \in t(X, D_k)$ with $x_l \notin I_k$. From $D_l \subset D$ and $D_k \subset D$ follows $I_l \in t(X, D)$ and $I_k \in t(X, D)$. Consequently, it is $I_l \notin t(Xx_k, D)$ and $I_k \notin t(Xx_l, D)$. This implies $t(X, D) \supset t(Xx_k, D)$ and $t(X, D) \supset t(Xx_l, D)$. From this it follows that $X$ must be closed in $D$.

□

## B.3 Proof of Theorem 3.3

**Theorem 3.3.** *Let $X$ be an item set and $Y \in Closed(D_i)$ its closure in time period $T_i$. If $\text{supp}_i(X) > 0$ then $Yt_i \in Closed(D^T)$ and $\text{supp}_i(X) = \text{supp}^T(Yt_i)/\text{supp}^T(t_i)$ .*

The proof of the theorem needs the following two lemmata:

**Lemma B.1.** *Let $X \subseteq L$, and let $\text{supp}_i^T$ denote an item set's support in regard to $D_i^T$. The following two properties hold:*

1. $\text{supp}_i(X) = \text{supp}_i^T(Xt)$
2. $\text{supp}(X) = \text{supp}^T(X)$

*Proof.* Definition 3.6 of $D_i^T$ states that each transaction $I \in D_i$ bijectively maps to a transaction $I \cup \{t_i\} \in D_i^T$ giving an exact pairing of the transactions in both data sets. Note that in the construction of $D_i^T$, respectively $D^T$, neither are transactions added or removed, nor are items deleted from them. This implies

$$|D_i| = |D_i^T| \tag{B.2}$$

and also that item sets which not contain $t_i$ have the same support in both data sets:

$$\text{supp}_i(X) = \text{supp}_i^T(X) \tag{B.3}$$

1. By Definition 3.6 each transaction in $D_i^T$ contains $t_i$. From this it follows that

$$\{I \in D_i^T : \ X \subseteq I\} \subseteq \{I \in D_i^T : \ t_i \in I\}$$

This in combination with Theorem 1 in Zaki and Hsiao (2005) implies that $X$ and $Xt_i$ have the same closure in $D_i^T$. Using the Definition 3.1 of closed item sets yields

$$\text{supp}_i^T(X) = \text{supp}_i^T(Xt_i) \tag{B.4}$$

Assuming that $t_i \notin X$ and combining (B.3) with (B.4) implies

$$\text{supp}_i(X) = \text{supp}_i^T(Xt)$$

2. Due to $D = D_1 \cup \ldots \cup D_n$ and $D_i \cap D_j = \emptyset$, $i \neq j$ it is

$$\text{supp}(X) = \frac{1}{|D|} \sum_{i_1}^{n} |D_i| \, \text{supp}_i(X) = \frac{1}{\sum_{i=1}^{n} |D_i|} \sum_{i_1}^{n} |D_i| \, \text{supp}_i(X)$$

Substituting from (B.2) and (B.3) yields

$$\text{supp}(X) = \frac{1}{\sum_{i=1}^{n} |D_i^T|} \sum_{i_1}^{n} |D_i^T| \, \text{supp}_i^T(X)$$

which is equivalent to

$$\text{supp}(X) = \text{supp}^T(X)$$

□

**Lemma B.2.** *It is $X \in Closed(D_i)$ if and only if $Xt_i \in Closed(D_i^T)$.*

*Proof.* The lemma is logically equivalent to the statement $X \notin Closed(D_i)$ if and only if $Xt_i \notin Closed(D_i^T)$ which is prove in the following. $Xt_i \notin Closed(D_i^T)$ is defined as there exists an item set $Y, X \cap Y = \emptyset$ such that $\text{supp}_i^T(Xt_i) = \text{supp}_i^T(XYt_i)$ (see Definition 3.1). Applying Lemma B.1 this is the case if and only if $\text{supp}_i(X) = \text{supp}_i(XY)$. By Definition 3.1 it is therefore $X \notin Closed(D_i)$. □

*Proof.* (of Theorem 3.3) From Lemma B.2 it is known that $Y \in Closed(D_i)$ implies $Yt_i \in Closed(D_i^T)$. Recalling that $D^T = D_1^T \cup \ldots D_n^T$ and applying Theorem 3.1 leads to $Closed(D_i^T) \subseteq Closed(D^T)$ and thus $Yt_i \in Closed(D^T)$. This proves the first part of the theorem.

Next, $\text{supp}_i(X) = \text{supp}^T(Yt_i)/\,\text{supp}^T(t_i)$ is shown:

$$\begin{aligned}
\text{supp}^T(Yt_i) &= \frac{|\{I \in D^T \ : \ Yt_i \subseteq I\}|}{|D^T|} \\
&= \frac{|\{I \in D_i^T \ : \ Yt_i \subseteq I\}|}{|D^T|} \qquad \text{(Each } T_i \text{ has its unique } t_i\text{)} \\
&= \frac{\text{supp}_i^T(Yt_i)}{|D^T|} \, |D_i^T| \\
&= \text{supp}_i^T(Yt_i) \ \text{supp}^T(t_i)
\end{aligned}$$

According to Lemma B.1 it is $\text{supp}_i^T(Yt_i) = \text{supp}_i(Y)$ and hence $\text{supp}_i(Y) = \text{supp}^T(Yt_i)/\,\text{supp}^T(t_i)$.

□

## B.4 Proof of Theorem 3.4

**Theorem 3.4.** *If $X \in Closed(D)$ and there exist $i \neq j$ such that $\text{supp}_i(X) > 0$ and $\text{supp}_j(X) > 0$ then it follows $X \in Closed(D^T)$.*

*Proof.* In the following, let $X \subseteq L$, i.e. $X$ does not contain any item $t \in L^T \setminus L$. The theorem assumes that $X$ is present in at least two time periods, i.e. $\text{supp}_i(X) > 0$ and $\text{supp}_j(X) > 0$, $i \neq j$.

The statement

$$X \in Closed(D) \Longrightarrow X \in Closed(D^T)$$

is logically equivalent to the statement

$$X \notin \mathit{Closed}(D^T) \Longrightarrow X \notin \mathit{Closed}(D)$$

which is subsequently proven.

Suppose $X \notin \mathit{Closed}(D^T)$. By Definition 3.1 this is the case if

$$\exists Z: \; Z \supset X \; : \; \mathrm{supp}^T(Z) = \mathrm{supp}^T(X) \tag{B.5}$$

According to Lemma B.3 (to be found below this proof) $Z \subseteq L$ must hold, because otherwise this would contradict the above assumption of $X$ being present in at least two time periods. Applying now Lemma B.1 to (B.5) yields

$$\exists Z: \; Z \supset X \; : \; \mathrm{supp}(Z) = \mathrm{supp}(X)$$

which entails

$$X \notin \mathit{Closed}(D)$$

□

**Lemma B.3.** *Assume $X \subset L$ and in $D^T$ there exists an item set $Z \supset X$ with $\mathrm{supp}^T(Z) = \mathrm{supp}^T(X) \neq 0$. If $\mathrm{supp}_i(X) > 0$ and $\mathrm{supp}_j(X) > 0$, $i \neq j$, then $Z \subseteq L$.*

*Proof.* This proof is by contraposition. Assume it is $t' \in Z$ such that $Z := Yt'$ with $t' \in L^T \setminus L$. From $Z = Yt' \supseteq Xt' \supset X$ and $\mathrm{supp}^T(Z) = \mathrm{supp}^T(Yt') = \mathrm{supp}^T(X)$ it follows $\mathrm{supp}^T(Xt') = \mathrm{supp}^T(X)$ (for a proof see Lemma 3.5 in Pei et al. (2000)).

Each transaction belongs to exactly one time period and time periods do not overlap. Hence, the $D_i^T$ are disjoint and it is

$$\begin{aligned}\mathrm{supp}^T(X) &= \sum_{t \in L^T \setminus L} \mathrm{supp}^T(Xt) \\ &= \sum_{t \in L^T \setminus (L \cup \{t'\})} \mathrm{supp}^T(Xt) + \mathrm{supp}^T(X)\end{aligned}$$

Which is equivalent to

$$\sum_{t \in L^T \setminus (L \cup \{t'\})} \mathrm{supp}^T(Xt) = 0$$

The support measure always is greater equal zero. If therefore the initial assumption $t' \in Z$ holds the following must hold too

$$\forall t \in L^T \setminus (L \cup \{t'\}) \; : \; \mathrm{supp}^T(Xt) = 0 \tag{B.6}$$

The latter statement leads to the negation of the lemma's assumption $\mathrm{supp}_i(X) > 0$ and $\mathrm{supp}_j(X) > 0$, $i \neq j$ as shown next. Let $k \in \{i, j\}$. Using Lemma B.1 it is

$$\mathrm{supp}_k^T(Xt_k) = \mathrm{supp}_k(X) > 0 \tag{B.7}$$

Because of $\mathrm{supp}_k(X) > 0$, $D_k$ is non-empty and thus $\mathrm{supp}^T(t_k) > 0$. Further it is

$$\mathrm{supp}_k(X) = \frac{\mathrm{supp}^T(Xt_k)}{\mathrm{supp}^T(t_k)}$$

Substitution into (B.7) followed by some basic transformations thus results in

$$\text{supp}^T(Xt_k) = \text{supp}_k^T(Xt_k)\,\text{supp}^T(t_k) > 0$$

□

## B.5 Proof of Theorem 3.5

**Theorem 3.5.** *The item set $XY$ is temporally derivable from the item set $X$ if and only if $\forall T_i \in \hat{T} : P(\mathcal{Y} \,|\mathcal{X} = 1,\ \mathcal{T} = T_i) = P(\mathcal{Y} \,|\mathcal{X} = 1)$.*

*Proof.* It needs to be shown that

$$X \hookrightarrow XY \quad \Leftrightarrow \quad P(\mathcal{Y} \,|\mathcal{X} = 1,\ \mathcal{T}) = P(\mathcal{Y} \,|\mathcal{X} = 1)$$

By Definition 3.8 it is $X \hookrightarrow XY$ if and only if

$$\exists \varepsilon \in [0,1] :\ \forall T_i \in \hat{T}\ :\ \text{supp}_i(XY) = \varepsilon\,\text{supp}_i(X)$$

Using the probabilistic interpretation of support this is equivalent to

$$\exists \varepsilon \in [0,1] :\ \forall T_i \in \hat{T} :\ P(\mathcal{Y} = 1,\ \mathcal{X} = 1 \mid \mathcal{T} = T_i) = \varepsilon P(\mathcal{X} = 1 \mid \mathcal{T} = T_i)$$

and, as a first step, it will be shown that this is equivalent to

$$\forall T_i \in \hat{T} :\ P(\mathcal{Y} = 1 \,|\mathcal{X} = 1, \mathcal{T} = T_i) = P(\mathcal{Y} = 1 \,|\mathcal{X} = 1)$$

.

Next, each direction of the latter equivalence relation is proven individually.

$(\Rightarrow)\ \exists \varepsilon \in [0,1] :\ \forall T_i \in \hat{T} :\ P(\mathcal{Y} = 1, \mathcal{X} = 1 \mid \mathcal{T} = T_i) = \varepsilon P(\mathcal{X} = 1 \mid \mathcal{T} = T_i)$

$$\begin{aligned}
\Leftrightarrow \quad & \exists \varepsilon \in [0,1] :\ \forall T_i \in \hat{T} : && \frac{P(\mathcal{Y} = 1, \mathcal{X} = 1, \mathcal{T} = T_i)}{P(\mathcal{T} = T_i)} \\
& && = \varepsilon \frac{P(\mathcal{X} = 1, \mathcal{T} = T_i)}{P(\mathcal{T} = T_i)} \\
\Leftrightarrow \quad & \exists \varepsilon \in [0,1] :\ \forall T_i \in \hat{T} : && P(\mathcal{Y} = 1, \mathcal{X} = 1, \mathcal{T} = T_i) \\
& && = \varepsilon P(\mathcal{X} = 1, \mathcal{T} = T_i) \\
\Rightarrow \quad & \exists \varepsilon \in [0,1] : && \sum_{T_i \in \text{dom}(\mathcal{T})} P(\mathcal{Y} = 1, \mathcal{X} = 1, \mathcal{T} = T_i) \\
& && = \varepsilon \sum_{T_i \in \text{dom}(\mathcal{T})} P(\mathcal{X} = 1, \mathcal{T} = T_i) \\
\Leftrightarrow \quad & \exists \varepsilon \in [0,1] : && P(\mathcal{Y} = 1, \mathcal{X} = 1) = \varepsilon P(\mathcal{X} = 1)
\end{aligned}$$

From this it follows that $\varepsilon = P(\mathcal{Y} = 1 \mid \mathcal{X} = 1)$ which substituted into the above equation and after division by $P(\mathcal{X} = 1 \mid \mathcal{T} = T_i)$ yields the assertion.

$$
\begin{aligned}
(\Leftarrow)\ \forall T_i \in \hat{T}:\ & P(\mathcal{Y}=1\,|\mathcal{X}=1,\,\mathcal{T}=T_i) = P(\mathcal{Y}=1\,|\mathcal{X}=1) \\
\Leftrightarrow\quad \forall T_i \in \hat{T}:\ & \frac{P(\mathcal{Y}=1,\mathcal{X}=1 \mid \mathcal{T}=T_i)}{P(\mathcal{X}=1 \mid \mathcal{T}=T_i)} \\
& \qquad = \frac{P(\mathcal{Y}=1,\mathcal{X}=1)}{P(\mathcal{X}=1)} \\
\Leftrightarrow\quad \forall T_i \in \hat{T}:\ & P(\mathcal{Y}=1,\mathcal{X}=1 \mid \mathcal{T}=T_i) \\
& \qquad = \frac{P(\mathcal{Y}=1,\mathcal{X}=1)}{P(\mathcal{X}=1)} P(\mathcal{X}=1 \mid \mathcal{T}=T_i) \\
\Rightarrow\quad \exists \varepsilon \in [0,1]:\ \forall T_i \in \hat{T}:\ & P(\mathcal{Y}=1,\mathcal{X}=1 \mid \mathcal{T}=T_i) \\
& \qquad = \varepsilon P(\mathcal{X}=1 \mid \mathcal{T}=T_i)
\end{aligned}
$$

Utilising the binary nature of $\mathcal{Y}$ and the universal quantification over $T_i$ the proven equivalence relation can be reformulated such that the theorem results. □

## B.6 Proof of Theorem 3.8

**Theorem 3.8.** *Given a time-stamped data set $D$ and a corresponding partition of the time axis into periods $\hat{T} := (T_1, \ldots, T_n)$, the set of temporally closed item sets contained in $D$ is a subset of the set of item sets that are closed over the sequence $\hat{T}$, in other words $TClosed(D, \hat{T}) \subseteq SeqClosed(D, \hat{T})$.*

To proof this theorem the definition of an item set which is closed over a sequence of time periods needs to be linked to the notion of temporal derivability. Comparison of Definition 3.5 with Definition 3.8 suggests to express the link between the two concepts as follows:

**Lemma B.4.** *An item set $X$ is closed over the sequence of time periods $\{T_1, \ldots, T_n\}$ iff there exists no item set $Y \supset X$ such that $X \overset{1}{\hookrightarrow} Y$.*

*Proof.* Follows directly from the definition of a temporally derivable item set (see Definition 3.8, page 51). □

Using this lemma the above theorem is provable.

*Proof.* By Definition 3.9 it is

$$X \in TClosed(D, \hat{T}) \Longleftrightarrow \nexists Y \supset X : \exists \epsilon \in (0,1] : X \overset{\epsilon}{\hookrightarrow} Y$$

from this it follows

$$\nexists Y \supset X : X \overset{1}{\hookrightarrow} Y$$

Application of Lemma B.4 then yields

$$X \in SeqClosed(D, \hat{T})$$

From $X \in TClosed(D, \hat{T}) \Rightarrow X \in SeqClosed(D, \hat{T})$ it follows that $TClosed(D, \hat{T}) \subseteq SeqClosed(D, \hat{T})$. □

## Appendix C

# Background

The following topics may be considered typical textbook knowledge and for this reason they have been moved to the end of this thesis and into the appendix. They provide background knowledge on some of the concepts and methods employed in this thesis and are intended to help the less knowledgeable reader.

## C.1 Background on Entropy and Mutual Information

Central to information theory is the concept of the *entropy of a random variable* as a measure of its information content (Shannon, 1948).

**Definition C.1** (Shannon Entropy)**.** *Let $\mathcal{A}$ be a discrete random variable with domain* $\mathrm{dom}(\mathcal{A})$ *and $P$ its probability distribution. Then*

$$H(\mathcal{A}) = -\sum_{a \in \mathrm{dom}(\mathcal{A})} P(\mathcal{A} = a) \log P(\mathcal{A} = a)$$

*is called the (Shannon) entropy of $\mathcal{A}$ w.r.t. $P$.*

Entropy reflects the uncertainty associated with a random variable. Consider a random variable that describes an experiment with two different outcomes, like the flip of a coin. If one outcome has probability one, and the other never occurs then the entropy is zero. There is no uncertainty about the value of the random variable and thus the information it contains is zero. On the contrary, if it is a fair coin and both outcomes have equal probability, the entropy is one. This is the situation of maximum uncertainty, hence the random variable contains the most information.

An alternative interpretation of entropy is that of a lower bound on the average description length of a random variable. In detail, it measures the average number of bits needed at least to describe its values. Note that the unit 'bit' is only used in conjunction with a logarithm of base 2 in Definition C.1. Unless otherwise stated all logarithms in this chapter will be to this base. Intuitively, an efficient code should assign long descriptions to less probable values and short descriptions to highly probable and thus frequent values. In this way, having a sequence of values information compression can be achieved. For example, in Morse code the most frequent letter in the English language, that is the 'e', is assigned the shortest description, a single 'dot'; whereas infrequent letters, like the 'j' have the longest description ("dot, dash, dash, dash"). Morse code, although being reasonably efficient, still has an average description length above the theoretical limit given by the entropy of the English language (Cover and Thomas, 2006, p. 104-5).

The uncertainty about a random variable and hence its average description length can be reduced by knowledge of another random variable. The resulting entropy thus is conditional on the values of the other. As such it is defined as the expected value of

the entropies of the conditional distributions averaged over the conditioning variable (Shannon, 1948).

**Definition C.2** (Conditional Entropy)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be discrete random variables with domains* $\mathrm{dom}(\mathcal{A})$ *and* $\mathrm{dom}(\mathcal{B})$*, and $P$ their (joint) probability distribution. Then*

$$\begin{aligned} H(\mathcal{A} \,|\, \mathcal{B}) = & \sum_{b \in \mathrm{dom}(\mathcal{B})} P(\mathcal{B} = b)\; H(\mathcal{A} | \mathcal{B} = b) \\ = & - \sum_{b \in \mathrm{dom}(\mathcal{B})} P(\mathcal{B} = b) \sum_{a \in \mathrm{dom}(\mathcal{A})} P(\mathcal{A} = a | \mathcal{B} = b)\; \log P(\mathcal{A} = a | \mathcal{B} = b) \end{aligned}$$

*is called the conditional entropy of $\mathcal{A}$ w.r.t. $\mathcal{B}$.*

Taking into consideration that the (unconditioned) entropy as in Definition C.1 expresses the information content prior to obtaining any other knowledge, the difference between both quantities appears suitable as a measure on how much information a random variable contains about another one. Indeed, in information theory this measure is known as *mutual information* (Shannon and Weaver, 1949).

**Definition C.3** (Mutual Information)**.** *Let $\mathcal{A}$ and $\mathcal{B}$ be discrete random variables with domains* $\mathrm{dom}(\mathcal{A})$ *and* $\mathrm{dom}(\mathcal{B})$*, and $P$ their joint probability distribution. Then*

$$I(\mathcal{A}\,,\, \mathcal{B}) = \quad H(\mathcal{A}) - H(\mathcal{A} \,|\, \mathcal{B}) \tag{C.1}$$

$$= \sum_{a \in \mathrm{dom}(\mathcal{A})} \sum_{b \in \mathrm{dom}(\mathcal{B})} P(\mathcal{A} = a,\, \mathcal{B} = b)\; \log \frac{P(\mathcal{A} = a,\, \mathcal{B} = b)}{P(\mathcal{A} = a) P(\mathcal{B} = b)} \tag{C.2}$$

*is called the mutual information of $\mathcal{A}$ and $\mathcal{B}$.*

Note that the argument of the logarithm is the ratio of the joint probability distribution and the product of the marginal distributions. Thus, the mutual information is zero if and only if $\mathcal{A}$ and $\mathcal{B}$ are probabilistically independent since this means $P(\mathcal{A} = a,\, \mathcal{B} = b) = P(\mathcal{A} = a) P(\mathcal{B} = b)$. In turn, mutual information grows with the strength of the dependence between $\mathcal{A}$ and $\mathcal{B}$. In a sense, the assumption of $\mathcal{A}$ and $\mathcal{B}$ being independent serves as a baseline against which $P(\mathcal{A} = a,\, \mathcal{B} = b)$ is assessed.

## C.2 Background on Regression Methods

Given is an inference problem involving data $\{(t_i, y_i) \mid i = 1, \ldots, r,\; t_i \in \mathbb{R},\; y_i \in \mathbb{R}\}$ where the $t_i$ are the inputs, and the $y_i$ are the targets. The task is to obtain *predictions* $y_*$ for new inputs $t_*$. Because the true underlying relationship $\varphi' : \mathbb{R} \longrightarrow \mathbb{R}$ with $\varphi'(t) = y$ is an *unknown function*, such predictions are only feasible on basis of a *model* $\varphi : \mathbb{R} \longrightarrow \mathbb{R}$ with $\varphi(t) = y$ which reasonably *approximates* $\varphi'$. In the following two methods for approximation and subsequent prediction are detailed. The first is linear regression in combination with a model selection heuristic, the second are Gaussian processes.

### C.2.1 Linear Regression with Basis Functions

This method requires a set of family of functions being specified in advance, by selecting in (4.3) different types of functions $\zeta_i$ and values for $q$. Two popular types of functions

are shown in (4.4) and (4.5) in the previous section. Commonly, the $\zeta_i$ are fixed to a certain type of function and only $q$ is varied. Utilising sample data, for each family parameter values are calculated that yield a 'best fit' of $\varphi$. Least square regression accomplishes this.

The objective of *least square regression* is to determine the parameters $\mathbf{a}_{min} := (a_0, \ldots, a_q)^T$ in (4.3) such that

$$\mathbf{a}_{min} = \underset{\mathbf{a}}{\operatorname{argmin}} \frac{1}{2} \|C\mathbf{a} - \mathbf{p}\|_2^2 \tag{C.3}$$

$$\text{with } C := \begin{pmatrix} \zeta_0(t_1) & \cdots & \zeta_q(t_1) \\ \vdots & \ddots & \vdots \\ \zeta_0(t_r) & \cdots & \zeta_q(t_r) \end{pmatrix} \text{ and } \mathbf{p} := \begin{pmatrix} y_1 \\ \vdots \\ y_r \end{pmatrix}$$

Having a fitted function for each selected family the 'best' one needs to be selected. Several heuristics have been developed for this purpose which serve a variety of definitions of 'best'. They all are agglomerated under the term *model selection heuristics*.

One is the Akaike information criterion (AIC) (Akaike, 1974). For it, 'best' means that function which provides the best trade-off between goodness of fit and complexity and is, for this reason, assumed to be less prone to overfit the data. Notably, Böttcher et al. (2008b) used linear regression with polynomials in combination with $AIC$ in the context of $PreDeT$. Let $r$ be the number of observations, e.g. the length of the history, $q+1$ the number of parameters and $RSS$ the residual sum of squares of a fitted function. Then, AIC is defined as:

$$AIC = 2(q+1) + r \ln \frac{RSS}{r} \tag{C.4}$$

In particular when dealing with histories the number of time periods for which data is available is sometimes small. The original Akaike information criterion, however, should only be applied to data sets with large sample sizes (Burnham and Anderson, 2004), i.e. if $r/(q+1) > 40$. To overcome this limitation a number of corrections of the Akaike criterion for small sample sizes have been developed, such as the following known as $AIC_C$ (Hurvich and Tsai, 1989):

$$AIC_C = AIC + \frac{2(q+1)(q+2)}{r-q-2} \tag{C.5}$$

For large sample sizes $r$ $AIC_C$ converges to $AIC$, therefore it can always be used regardless of sample size (Burnham and Anderson, 2004).

### C.2.2 Gaussian Process Regression

Gaussian process regression learns a model from sample data by an approach that is not dependent on strong prior assumptions regarding $\varphi'$, such as assumptions concerning the specific family of functions. The remainder of this section provides an illustrative introduction into some of its core concepts. It should be stressed that Gaussian process regression involves many more aspects than the ones sketched here; the discussion of those would by far go beyond the scope of this thesis. These aspects, among many

others, concern noise handling and hyper-parameter learning for kernel functions. For a detailed discussion of those the reader is referred to the book by Rasmussen and Williams (2005).

Loosely speaking, a *Gaussian process* is a probability distribution over functions. It generalises the well-known (multivariate) Gaussian probability distribution. This means, while the latter is defined on a finite number of random variables, a Gaussian process defines a distribution over an infinite number of random variables. Drawing samples from a Gaussian process yields functions; not single values or vectors as for univariate respectively multivariate Gaussian distributions. Analogue to a Gaussian probability distribution, a Gaussian process is defined by a mean and a covariance, only that both are not expressed by a value but are functions themselves.

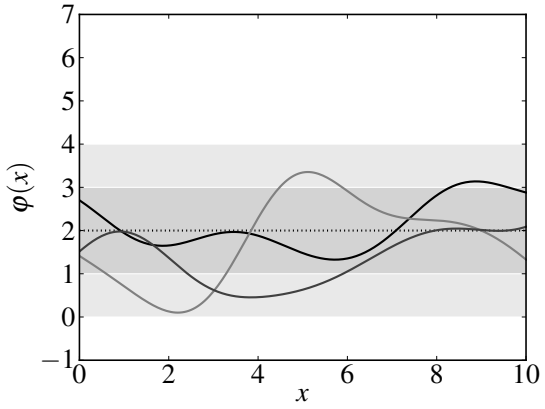
Given sample data (e.g., a history of attribute evaluation measures), Gaussian process regression is carried out using Bayesian inference. It assumes that the observed function values differ from the unknown real ones by additive Gaussian noise. The aim of Bayesian inference is to yield a *posterior distribution* over the function space by combining a *prior distribution* (which encodes the initial believes about the particular problem) with the sample data (which represents information regarding the unknown function $\varphi'$).

Figure C.1(i) shows three functions drawn from a Gaussian process with a mean of zero. The mean is shown as a black dotted line. The dark shaded area covers once and the light shaded area twice the standard deviation at each input $x$. Figure C.1(j) shows three functions drawn from the posterior Gaussian process which results from combining the prior in Figure C.1(i) with the sample data shown as black squares and assuming no noise. Apparently, the uncertainty is reduced, close to the samples but also between them.

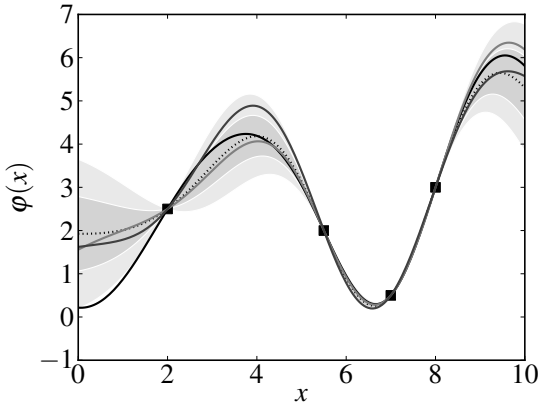
Applied to the search for a model $\varphi : \mathbb{R} \longrightarrow \mathbb{R}$ with $\varphi(t) = y$, a Gaussian process defines for each $t \in \mathbb{R}$ in input space a (univariate) Gaussian distribution over $\varphi(t) = y \in \mathbb{R}$ in output space. This means, $\varphi(t)$ is seen as a random variable which is indexed by $t$. A characteristic property of Gaussian processes is that for any finite subset of indices $\{t_1, \ldots, t_m\} \subset \mathbb{R}$ the corresponding random variables follow a multivariate Gaussian distribution. Thus, multivariate Gaussian distributions are a special case of Gaussian processes.

The mean of the Gaussian process at $t_*$ is used as prediction for the value $\varphi'(t_*)$. Figure C.1(k) and Figure C.1(j) show the mean of Gaussian processes which account, respectively not account, for noise. Because a Gaussian process sees $\varphi(t_*)$ as a normally distributed random variable, the standard deviation is an indicator for the goodness of the prediction. For example, in Figure C.1(j) a prediction at $t_* = 7.5$ has a standard deviation of almost zero, whereas a prediction at $t_* = 3.8$ has a much greater standard deviation; this prediction thus is connected with a significantly higher uncertainty. In Figure C.1(k) the input value $(7,\ 1.5)$ is more than twice the standard deviation away from the mean, indicating that this could be an outlier. Note that such assertions are not possible with standard linear least squares regression discussed in the preceding section.

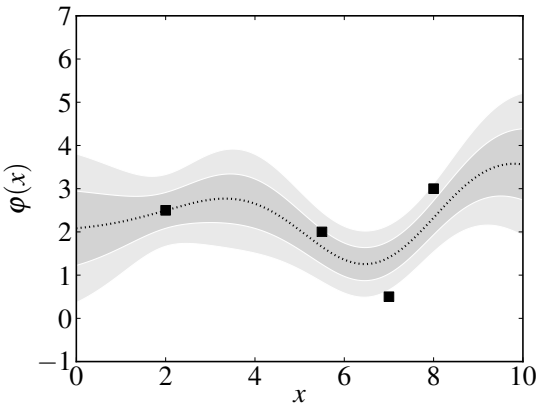
The covariance function, $cov : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$ of the Gaussian process defines a prior over functions based on properties, such as the above stated. Precisely, $cov(\varphi(t), \varphi(t'))$ specifies the covariance between the output of the function $\varphi$ at a pair of indices $(t, t')$. It measures how much the random variables $\varphi(t)$ and $\varphi(t')$ influence each other. For




(i) Samples from the prior




(j) Samples from the posterior




(k) Mean of the posterior with noise.

Figure C.1: Samples from Gaussian processes before and after considering sample data and noise. The dotted line shows the Gaussian processes' mean, which corresponds to the predicted value for each $x$ for the posterior.

Gaussian process regression $cov(\varphi(t), \varphi(t'))$ is represented by a so-called *kernel function* $k(t, t') = cov(\varphi(t), \varphi(t'))$. Kernels for Gaussian process regression are discussed in Rasmussen and Williams (2005), Chapter 4. One example is the *squared exponential kernel*

$$k_{SE}(t, t') = e^{-\frac{(t-t')^2}{2l^2}}$$

The parameter $l$ is called *length scale*. Informally, it specifies at which distance function values (or samples) roughly become uncorrelated along the input dimension. As Figure C.2 shows, for this kernel the covariance is almost unity if the inputs are very close and decreases slowly with increasing distance between them, eventually converging to zero. Consequently, a Gaussian process with squared exponential kernel favours very smooth functions over those that change suddenly. The functions in Figure C.1(i) are drawn from a Gaussian process with squared exponential kernel.

In practical applications of Gaussian process regression it is often assumed that the mean function of the prior is zero everywhere without loss of generality, because the data can be always be shifted accordingly. For this reason, the main modelling task for Gaussian process regression is the suitable choice of the *kernel function* which encodes

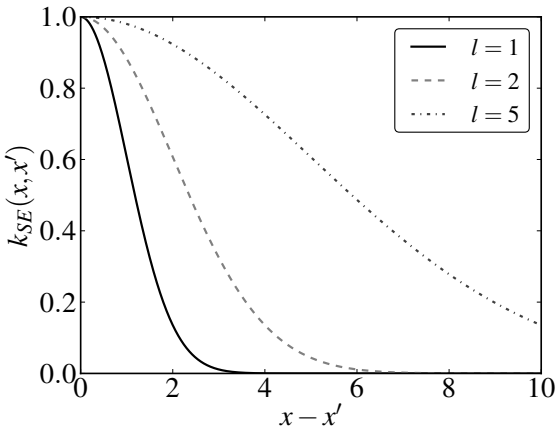



Figure C.2: Squared exponential kernel with different length scales $l$

prior knowledge concerning $\varphi'$, specifically properties like derivability, smoothness or periodicity. Because it is intuitively easier and also requires less knowledge to directly judge such properties of a domain, rather than judging them indirectly via selecting the basis functions $\zeta$, obtaining a suitable model becomes likelier.

At last, Gaussian process regression has some correspondence with linear regression with basis functions. Williams (1999) proved that general Gaussian process regression is equivalent to generalised Bayesian linear regression, that is Bayesian linear regression (see, e.g., Gelman et al., 2003) with an *infinite* number of basis functions $\zeta_i$. Each choice of kernel in the Gaussian process framework is linked to a particular type of basis function in the linear regression framework $\varphi(x) = \sum_{i=0}^{\infty} a_i \zeta_i(x)$. For many kernels the corresponding family of basis functions is not analytically derivable and, in turn, not every family of basis functions yields a valid kernel function. Conducting such transformation, however, is only of theoretical interest.

For example, the squared exponential kernel is linked to radial basis functions as in (4.5) with $c$ denoting their centre (Rasmussen and Williams, 2005, p. 84). Considering models with a finite number of these basis functions leads to *radial basis function networks* (RBF) established in the field of Neural Networks (see ). Radial basis functions also have been discussed in the previous section as one example of basis function for linear least squares regression. Outside the scope of Gaussian process regression, the quality of a RBF model strongly depends on putting the centres at the right place prior to learning its parameters, because at the end only one fitted function will be created. In comparison, Gaussian process regression with squared exponential kernel implicitly consider an infinite number of basis functions centred everywhere in $\mathbb{R}$. Instead of producing one fitted function the fit is described by a posterior probability over all functions. Gaussian process regression thus is much more flexible and less restrictive than linear regression with basis functions. Comparing Figure 4.7 with Figure C.1(k), underlines its superiority.

# Bibliography


Ackoff, R. L. (1981). *Creating the corporate future : plan or be planned for*. Wiley, New York.

Agarwal, D., Barman, D., Gunopulos, D., Young, N. E., Korn, F., and Srivastava, D. (2007). Efficient and effective explanation of change in hierarchical summaries. In *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 6–15, New York, NY, USA. ACM.

Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington D.C. ACM.

Agrawal, R. and Psaila, G. (1995). Active data mining. In Fayyad, Usama, M. and Uthurusamy, R., editors, *Proceedings of the 1st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3–8, Montreal, Quebec, Canada. AAAI Press, Menlo Park, CA, USA.

Agrawal, R. and Srikant, R. (1995). Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, ICDE '95, pages 3–14, Washington, DC, USA. IEEE Computer Society.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723.

Au, W.-H. and Chan, K. (2005). Mining changes in association rules: a fuzzy approach. *Fuzzy Sets and Systems*, 149(1):87–104.

Baron, S. (2004). *Temporale Aspekte entdeckten Wissens – Ein Bezugssystem für die Evolution von Mustern*. PhD thesis, Humboldt University Berlin.

Baron, S. and Spiliopoulou, M. (2001). Monitoring change in mining results. In *Proceedings of the 3rd International Conference on Data Warehousing and Knowledge Discovery*, Munich, Germany.

Baron, S. and Spiliopoulou, M. (2003). Monitoring the evolution of web usage patterns. In *1st European Web Mining Forum, Workshop at ECML/PKDD 2003*.

Baron, S., Spiliopoulou, M., and Günther, O. (2003). Efficient monitoring of patterns in data mining environments. In *Proceedings of the 7th East-European Conference on Advances in Databases and Information Systems (ADBIS'03)*, volume 2798 of *Lecture Notes in Computer Science*, pages 253–265, Berlin, Heidelberg, New York. Springer.

Bartolini, I., Ciaccia, P., Ntoutsi, I., Patella, M., and Theodoridis, Y. (2004). A unified and flexible framework for comparing simple and complex patterns. In *PKDD '04: Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 3202 of *Lecture Notes In Computer Science*, pages 496–499, New York, NY, USA. Springer-Verlag New York, Inc.

Bartolini, I., Ciaccia, P., Ntoutsi, I., Patella, M., and Theodoridis, Y. (2009). The PANDA framework for comparing patterns. *Data and Knowledge Engineering*, 68(2):244–260.

Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., and Lakhai, L. (2000). Mining frequent patterns with counting inference. *SIGKDD Explorations Newsletter*, 2(2):66–75.

Bay, S. D. and Pazzani, M. J. (1999). Detecting change in categorical data: mining contrast sets. In *KDD '99: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 302–306, New York, NY, USA. ACM.

Bay, S. D. and Pazzani, M. J. (2001). Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery*, 5(3):213–246.

Bayardo, R., Agrawal, R., and Gunopulos, D. (2000). Constraint-based rule mining in large, dense databases. *Data Mining and Knowledge Discovery*, 4(3):217–240.

Bayardo, Jr., R. J. (1998). Efficiently mining long patterns from databases. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 85–93, New York, NY, USA. ACM.

Berger, C. R. (2005). Slippery slopes to apprehension: Rationality and graphical depictions of increasingly threatening trends. *Communication Research*, 32(1):3–28.

Bille, P. (2005). A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1-3):217–239.

Bonchi, F. and Lucchese, C. (2004). On closed constrained frequent pattern mining. In *Proceedings of the 4th IEEE International Conference on Data Mining*, ICDM '04, pages 35–42, Washington, DC, USA. IEEE Computer Society.

Borgelt, C. and Kruse, R. (1998). Attributauswahlmasse für die Induktion von Entscheidungsbäumen: Ein Überblick. In Nakhaeizadeh, G., editor, *Data Mining: Theoretische Aspekte und Anwendungen*, pages 77–98. Physica-Verlag, Heidelberg, Germany.

Borgelt, C. and Kruse, R. (2002). *Graphical Models*. John Wiley & Sons.

Böttcher, M. (2011). Contrast and change mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):215–230.

Böttcher, M., Nauck, D., Borgelt, C., and Kruse, R. (2006a). A framework for discovering interesting business changes from data. *BT Technology Journal*, 24(3):219–228.

Böttcher, M., Nauck, D., Ruta, D., and Spott, M. (2006b). Towards a framework for change detection in datasets. In Bramer, M., Coenen, F., and Tuson, A., editors, *Research and Development in Intelligent Systems XXIII*, Proceedings of AI-2006, the 26th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, pages 115–128. BCS SGAI, Springer London.

Böttcher, M., Spiliopoulou, M., and Höppner, F. (2008a). On exploiting the power of time in data mining. *SIGKDD Explorations Newsletter*, 10(2):3–11.

Böttcher, M., Spott, M., and Kruse, R. (2008b). Predicting future decision trees from evolving data. In *ICDM '08: Proceedings of the 8th IEEE International Conference on Data Mining*, pages 33–42. IEEE Computer Society.

Böttcher, M., Spott, M., and Kruse, R. (2009). A condensed representation of itemsets for analyzing their evolution over time. In *ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, volume 5781 of *Lecture Notes In Artificial Intelligence*, pages 163–178. Springer-Verlag.

Böttcher, M., Spott, M., and Nauck, D. (2005). Detecting temporally redundant association rules. In *Proceedings of the Fourth International Conference on Machine Learning and Applications*, pages 397–403, Washington, DC, USA. IEEE Computer Society.

Böttcher, M., Spott, M., Nauck, D., and Kruse, R. (2009). Mining changing customer segments in dynamic markets. *Expert Systems with Applications*, 36(1):155–164.

Boulicaut, J.-F., Bykowski, A., and Rigotti, C. (2000). Approximation of frequency queris by means of free-sets. In *Proceedings of the 4th European Conference on Principles and Practice of Data Mining and Knowledge Discovery*, number 1910 in Lecture Notes in Computer Science, pages 75–85, London, UK. Springer-Verlag.

Boulicaut, J.-F., Bykowski, A., and Rigotti, C. (2003). Free-sets: A condensed representation of boolean data for the approximation of frequency queries. *Data Mining and Knowledge Discovery*, 7(1):5–22.

Breiman, L. (1996). The heuristics of instability in model selection. *Annals of Statistics*, 24:2350–2383.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth, Belmont.

Brodsky, B. and Darkhovsky, B. (2010). *Nonparametric Methods in Change-Point Problems*. Kluwer Academic Publishers, Dordrecht, Boston, London.

Burnham, K. P. and Anderson, D. R. (2004). Multimodel inference: understanding AIC and BIC in model selection. *Sociological Methods & Research*, 33:261–304.

Bykowski, A. and Rigotti, C. (2001). A condensed representation to find frequent patterns. In *PODS '01: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 267–273, New York, NY, USA. ACM.

Calders, T. and Goethals, B. (2002). Mining all non-derivable frequent itemsets. In *PKDD '02: Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 74–85, London, UK. Springer-Verlag.

Calders, T. and Goethals, B. (2007). Non-derivable itemset mining. *Data Mining and Knowledge Discovery*, 14(1):171–206.

Calders, T., Rigotti, C., and Boulicaut, J.-F. (2005). A survey on condensed representations for frequent sets. In *Constraint Based Mining and Inductive Databases*, Lecture Notes in Artificial Intelligence, pages 64–80. Springer-Verlag.

Ceglar, A. and Roddick, J. F. (2006). Association mining. *ACM Computing Surveys*, 38(2):5.

Chakrabarti, S., Sarawagi, S., and Dom, B. (1998). Mining surprising patterns using temporal description length. In *VLDB '98: Proceedings of the 24th International Conference on Very Large Databases*, pages 606–617. Morgan Kaufmann Publishers Inc.

Charikar, M., Indyk, P., and Panigrahy, R. (2002). New algorithms for subset query, partial match, orthogonal range searching, and related problems. In *ICALP '02: Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, volume 2380 of *Lecture Notes in Computer Science*, pages 451–462. Springer-Verlag.

Chatfield, C. (1996). *The Analysis of Time Series – An Introduction.* Chapman and Hall/CRC, Boca Raton, London, New York.

Chatfield, C. (2001). *Time-Series Forecasting.* Chapman and Hall/CRC, Boca Raton, London, New York.

Chen, J. and Gupta, A. (2000). *Parametric Statistical Change Point Analysis.* Birkhäuser Boston.

Chen, M.-C., Chiu, A.-L., and Chang, H.-H. (2005). Mining changes in customer behavior in retail marketing. *Expert Systems with Applications*, 28(4):773–781.

Cheng, J., Ke, Y., and Ng, W. (2006). $\delta$-tolerance closed frequent itemsets. In *ICDM '06: Proceedings of the 6th IEEE International Conference on Data Mining*, pages 139–148, Washington, DC, USA. IEEE Computer Society.

Cheng, J., Ke, Y., and Ng, W. (2008). A survey on algorithms for mining frequent itemsets over data streams. *Knowledge and Information Systems*, 16:1–27.

Chi, Y., Wang, H., Yu, Philip, S., and Muntz, Richard, R. (2006). Catch the moment: maintaining closed frequent itemsets over a data stream sliding window. *Knowledge and Information Systems*, 10(3):265–294.

Chu, C.-J., Tseng, V. S., and Liang, T. (2009). Efficient mining of temporal emerging itemsets from data streams. *Expert Systems with Applications*, 36(1):885–893.

Cormode, G. and Muthukrishnan, S. (2005). What's new: finding significant differences in network data streams. *IEEE/ACM Transactions on Networking*, 13(6):1219–1232.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory, 2nd Edition.* Wiley-Interscience.

Dean, J. and Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51:107–113.

Demsar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

Dong, G. and Bailey, J., editors (2012). *Contrast Data Mining: Concepts, Algorithms, and Applications.* CRC Press.

Dong, G. and Li, J. (1999). Efficient mining of emerging patterns: discovering trends and differences. In *KDD '99: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 43–52, New York, NY, USA. ACM.

Dong, G. and Li, J. (2005). Mining border descriptions of emerging patterns from dataset pairs. *Knowledge and Information Systems*, 8(2):178–202.

Fan, H. and Ramamohanarao, K. (2003). Efficiently mining interesting emerging patterns. In *Advances in Web-Age Information Management*, volume 2762 of *Lecture Notes in Computer Science*, pages 189–201, Berlin, Heidelberg, New York. Springer.

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. (1996a). From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37–54.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R. (1996b). *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press.

Freyd, J. J. (1983). The mental representation of movement when static stimuli are viewed. *Perception & Psychophysics*, 33(6):575–581.

Fürnkranz, J. and Knobbe, A. J. (2010). Guest editorial: Global modeling using local patterns. *Data Mining and Knowledge Discovery*, 21:1–8.

Gaber, M. M., Zaslavsky, A., and Krishnaswamy, S. (2005). Mining data streams: a review. *SIGMOD Record*, 34(2):18–26.

Ganti, V., Gehrke, J., and Ramakrishnan, R. (1999). A framework for measuring changes in data characteristics. In *PODS '99: Proceedings of the 18th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 126–137, New York, NY, USA. ACM.

Ganti, V., Gehrke, J., Ramakrishnan, R., and Loh, W.-Y. (2002). A framework for measuring differences in data characteristics. *Journal of Computer and System Sciences*, 64(3):542–578.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2003). *Bayesian Data Analysis, Second Edition (Chapman & Hall/CRC Texts in Statistical Science)*. Chapman and Hall/CRC, 2nd edition.

Geng, L. and Hamilton, H. J. (2006). Interestingness measures for data mining: A survey. *ACM Computing Surveys*, 38(3):9.

Gill, P. E., Murray, W., and Wright, M. H. (1989). *Practical Optimization*. Academic Press, London.

Goldin, D. Q. and Kanellakis, P. C. (1995). On similarity queries for time-series data: Constraint specification and implementation. In *Proceedings of the 1st International Conference on Principles and Practice of Constraint Programming*, volume 976 of *Lecture Notes in Computer Science*, pages 137–153. Springer.

Gunopulos, D., Khardon, R., Mannila, H., Saluja, S., Toivonen, H., and Sharma, R. S. (2003). Discovering all most specific sentences. *ACM Transactions on Database Systems*, 28:140–174.

Hagen, J. G. and Rom, S. J. (1930). Die zwei unabhängigen Beweise der Erddrehung beim Foucaultschen Pendelversuch. *Die Naturwissenschaften*, 18(38):805–807.

Han, J. and Fu, Y. (1996). Exploration of the power of attribute-oriented induction in data mining. In U.M. Fayyad, G. Piatetsky-Shapiro, P. S. and Uthurusamy, R.,

editors, *Advances in Knowledge Discovery and Data Mining*, pages 83–115, Menlo Park, California. AAAI/MIT Press.

Hand, D. (2002). Pattern detection and discovery. In Hand, D., Adams, N., and Bolton, R., editors, *Pattern Detection and Discovery*, volume 2447 of *Lecture Notes in Computer Science*, pages 161–173. Springer Berlin / Heidelberg.

Hand, D. J., Mannila, H., and Smyth, P. (2001). *Principles of Data Mining*. The MIT Press.

Helmbold, D. P. and Long, P. M. (1994). Tracking drifting concepts by minimizing disagreements. *Machine Learning*, 14(1):27–45.

Hido, S., Idé, T., Kashima, H., Kubo, H., and Matsuzawa, H. (2008). Unsupervised change analysis using supervised learning. In *Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2008)*, volume 5012 of *Lecture Notes in Computer Science*, pages 148–159, Berlin, Heidelberg, New York. Springer.

Hilderman, R. J. and Peckham, T. (2005). A statistically sound alternative approach to mining contrast sets. In *Proceedings of the 4th Australasian Data Mining Conference (AusDM-05)*, pages 157–172.

Hipp, J., Guentzer, U., and Nakhaeizadeh, G. (2000). Algorithms for association rule mining - a general survey and comparison. *SIGKDD Explorations Newsletter*, 2(1):58–64.

Höppner, F. and Böttcher, M. (2007). Matching partitions over time to reliably capture local clusters in noisy domains. In *PKDD '07: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 4702 of *Lecture Notes in Computer Science*, pages 479–486. Springer-Verlag.

Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 97–106, New York, NY, USA. ACM Press.

Hurvich, C. M. and Tsai, C. L. (1989). Regression and time series model selection in small samples. *Biometrika*, 76:297–307.

Hyafil, L. and Rivest, R. L. (1976). Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, 5(1):15–17.

Inokuchi, A., Washio, T., and Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '00, pages 13–23, London, UK. Springer-Verlag.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323.

Jiang, T., Wang, L., and Zhang, K. (1995). Alignment of trees - an alternative to tree edit. *Theoretical Computer Science*, 143(1):137–148.

Kifer, D., Ben-David, S., and Gehrke, J. (2004). Detecting change in data streams.

In *VLDB '04: Proceedings of the 13th International Conference on Very Large Data Bases*, pages 180–191.

Kim, J. K., Song, H. S., Kim, T. S., and Kim, H. K. (2005). Detecting the change of customer behavior based on decision tree analysis. *Expert Systems*, 22(4):193–205.

Kimball, R. (1996). *Data Warehouse Toolkit: Practical Techniques for Building High Dimensional Data Warehouses*. John Wiley & Sons.

Klinkenberg, R. (2004). Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300.

Kotsiantis, S. B., Zaharakis, I. D., and Pintelas, P. E. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Reviews*, 26(3):159–190.

Kryszkiewicz, M. and Gajek, M. (2002). Concise representation of frequent patterns based on generalized disjunction-free generators. In *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD '02, pages 159–171, London, UK. Springer-Verlag.

Kubat, M. (1989). Floating approximation in time-varying knowledge bases. *Pattern Recognition Letters*, 10(4):223–227.

Kuh, A., Petsche, T., and Rivest, R. L. (1990). Learning time-varying concepts. In *NIPS-3: Proceedings of the 1990 Conference on Advances in Neural Information Processing Systems 3*, pages 183–189, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kullback, S. (1968). *Information Theory and Statistics*. Dover Publications, New York, 1997 edition.

Kuramochi, M. and Karypis, G. (2001). Frequent subgraph discovery. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, pages 313–320, Washington, DC, USA. IEEE Computer Society.

Lanquillon, C. and Renz, I. (1999). Adaptive information filtering: detecting changes in text streams. In *CIKM '99: Proceedings of the 8th International Conference on Information and Knowledge Management*, pages 538–544, New York, NY, USA. ACM.

Li, J., Dong, G., and Ramamohanarao, K. (2001). Making use of the most expressive jumping emerging patterns for classification. *Knowledge and Information Systems*, 3(2):1–29.

Liu, B., Hsu, W., Han, H.-S., and Xia, Y. (2000). Mining changes for real-life applications. In *Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery*, pages 337–346, London, UK. Springer.

Liu, B., Hsu, W., and Ma, Y. (2001a). Discovering the set of fundamental rule changes. In *KDD '01: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 335–340, New York, NY, USA. ACM.

Liu, B., Ma, Y., and Lee, R. (2001b). Analyzing the interestingness of association rules from the temporal dimension. In *Proceedings of the IEEE International Conference on Data Mining*, pages 377–384. IEEE Computer Society.

Liu, B. and Tuzhilin, A. (2008). Managing large collections of data mining models. *Communications of the ACM*, 51(2):85–89.

Liu, G., Li, J., and Wong, L. (2008). A new concise representation of frequent itemsets using generators and a positive border. *Knowledge and Information Systems*, 17:35–56.

Lorena, A. C., Carvalho, A. C., and Gama, J. a. M. (2008). A review on the combination of binary classifiers in multiclass problems. *Artificial Intelligence Reviews*, 30:19–37.

Mannila, H. and Toivonen, H. (1996). Multiple uses of frequent sets and condensed representations (extended abstract). In *Proceedings of the 2nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 189–194. AAAI Press.

Mannila, H. and Toivonen, H. (1997). Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1:241–258.

McQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

Novak, P. K., Lavrač, N., and Webb, G. I. (2009). Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. *The Journal of Machine Learning Research*, 10:377–403.

Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. (1999). Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1):25–46.

Pasquier, N., Bastide, Y., Taouil, R., Taouil, R., and Lakhal, L. (1998). Pruning closed itemset lattices for association rules. In *In Actes Bases de Donnèes Avancèes BDA'98, Hammamet, Tunisie*, pages 177–196.

Pei, J., Han, J., and Lakshmanan, L. V. (2001). Mining frequent itemsets with convertible constraints. In *Proceedings of the 17th International Conference on Data Engineering*, pages 433–442.

Pei, J., Han, J., and Mao, R. (2000). CLOSET: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 21–30.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Rafiei, D. and Mendelzon, A. (1997). Similarity-based queries for time series data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 13–25, New York. ACM Press.

Rasmussen, C. E. and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning*. MIT Press.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, 14:465–471.

Rissland, E. L. and Friedman, M. T. (1995). Detecting change in legal concepts. In *ICAIL '95: Proceedings of the 5th International Conference on Artificial Intelligence and Law*, pages 127–136, New York, NY, USA. ACM.

Roddick, J. F., Spiliopoulou, M., Lister, D., and Ceglar, A. (2008). Higher order mining. *SIGKDD Explorations Newsletter*, 10(1):5–17.

Rüping, S. (2006). *Learning Interpretable Models*. PhD thesis, Universität Dortmund.

Russ, G., Böttcher, M., Nauck, D., and Kruse, R. (2007). Relevance feedback for association rules by leveraging concepts from information retrieval. In Bramer, M., Coenen, F., and Petridis, M., editors, *Research and Development in Intelligent Systems XXIV*, Proceedings of AI-2007, the 27th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, pages 253–266. BCS SGAI, Springer London.

Sarawagi, S. (1999). Explaining differences in multidimensional aggregates. In *VLDB '99: Proceedings of the 25th International Conference on Very Large Data Bases*, pages 42–53, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Schlimmer, J. C. and Granger, R. H. (1986). Incremental learning from noisy data. *Machine Learning*, 1(3):317–354.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27:379–423.

Shannon, C. E. and Weaver, W. (1949). *The Mathematical Theory of Communication*. University of Illinois Press.

Silberschatz, A. and Tuzhilin, A. (1996). What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970–974.

Song, H. S., Kim, J. K., and Kim, S. H. (2001). Mining the change of customer behavior in an internet shopping mall. *Expert Systems with Applications*, 21(3):157–168.

Soulet, A. and Crémilleux, B. (2008). Adequate condensed representations of patterns. *Data Mining and Knowledge Discovery*, 17(1):94–110.

Soulet, A., Crémilleux, B., and Rioult, F. (2004). Condensed representation of emerging patterns. In *Advances in Knowledge Discovery and Data Mining, 8th Pacific-Asia Conference, PAKDD 2004*, volume 3056 of *Lecture Notes in Computer Science*, pages 127–132, Berlin, Heidelberg, New York. Springer.

Spiliopoulou, M., Ntoutsi, I., Theodoridis, Y., and Schult, R. (2006). MONIC: modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 706–711, New York, NY, USA. ACM.

Stanley, K. (2003). Learning concept drift with a committee of decision trees. Technical Report UT-AI-TR-03-302, Department of Computer Science, University of Texas at Austin, USA.

Steinbrecher, M. and Kruse, R. (2008). Identifying temporal trajectories of association

rules with fuzzy descriptions. In *Proc. Conf. North American Fuzzy Information Processing Society (NAFIPS 2008)*, pages 1–6.

Terlecki, P. and Walczak, K. (2007). On the relation between rough set reducts and jumping emerging patterns. *Information Sciences*, 177(1):74 – 83.

Tobin, W. (2003). *The Life and Science of Léon Foucault: The Man who Proved the Earth Rotates*. Cambridge University Press.

Tsai, C.-Y. and Shieh, Y.-C. (2009). A change detection method for sequential patterns. *Decision Support Systems*, 46(2):501–511.

Tsymbal, A. (2004). The problem of concept drift: definitions and related work. Technical Report TCD-CS-2004-15, Trinity College Dublin, Computer Science Department.

Vazirgiannis, M., Halkidi, M., and Gunopoulos, D. (2003). *Uncertainty Handling and Quality Assessment in Data Mining*. Springer.

Wang, J., Han, J., and Pei, J. (2003a). Closet+: Searching for the best strategies for mining frequent closed itemsets. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 236–245, New York, NY, USA. ACM.

Wang, K., Zhou, S., Fu, A. W.-C., and Yu, J. X. (2003b). Mining changes of classification by correspondence tracing. In *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM-03)*, pages 95–106, Philadelphia, PA. SIAM.

Wang, L., Zhao, H., Dong, G., and Li, J. (2005). On the complexity of finding emerging patterns. *Theoretical Computer Science*, 335(1):15–27.

Webb, G. I., Butler, S., and Newlands, D. (2003). On detecting differences between groups. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 256–265, New York, NY, USA. ACM.

Widmer, G. and Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Wilks, S. S. (1935). The likelihood test of independence in contingency tables. *The Annals of Mathematical Statistics*, 6(4):190–196.

Wilks, S. S. (1938). The large-sample distribution of the likelihood ratio for testing composite hypotheses. *The Annals of Mathematical Statistics*, 9(1):60–62.

Williams, C. K. I. (1999). Prediction with gaussian processes: from linear regression to linear prediction and beyond. In Jordan, M. I., editor, *Learning in graphical models*, pages 599–621. MIT Press, Cambridge, MA, USA.

Wong, T.-T. and Tseng, K.-L. (2005). Mining negative contrast sets from data with discrete attributes. *Expert Systems with Applications*, 29(2):401–407.

Wrobel, S. (1997). An algorithm for multi-relational discovery of subgroups. In *PKDD '97: Proceedings of the 1st European Symposium on Principles of Data Mining and*

*Knowledge Discovery*, volume 1263 of *Lecture Notes in Computer Science*, pages 78–87, London, UK. Springer-Verlag.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z.-H., Steinbach, M., Hand, D. J., and Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14:1–37.

Yang, G. (2004). The complexity of mining maximal frequent itemsets and maximal frequent patterns. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 344–353, New York, NY, USA. ACM.

Yang, Y., Wu, X., and Zhu, X. (2005). Combining proactive and reactive predictions for data streams. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 710–715, New York, NY, USA. ACM Press.

Zaki, M. J. (2000). Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3):372–390.

Zaki, M. J. (2004). Mining non-redundant association rules. *Data Mining and Knowledge Discovery*, 9(3):223–248.

Zaki, M. J. and Hsiao, C.-J. (2002). CHARM: An efficient algorithm for closed itemset mining. In *Proceedings of the 2nd SIAM International Conference on Data Mining*, Arlington, VA. SIAM.

Zaki, M. J. and Hsiao, C.-J. (2005). Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):462–478.

Zaki, M. J. and Ogihara, M. (1998). Theoretical foundations of association rules. In *In 3rd ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 1–8.

Zhang, X., Dong, G., and Kotagiri, R. (2000). Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets. In *KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 310–314, New York, NY, USA. ACM.

Zimmermann, A. (2009). *Mining Sets of Patterns*. PhD thesis, Katholieke Universiteit Leuven.