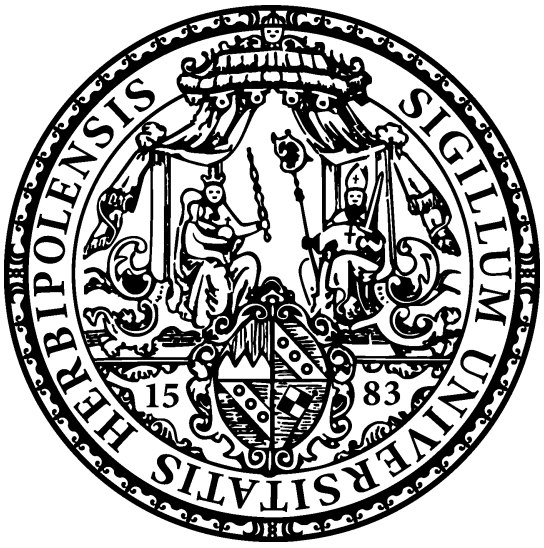
Novel Techniques for Efficient and Effective Subgroup Discovery

vorgelegt von

Florian Lemmerich



Dissertation zur Erlangung des naturwissenschaftlichen Doktorgrades
der Bayerischen Julius-Maximilians-Universität Würzburg

Abstract

Large volumes of data are collected today in many domains. Often, there is so much data available, that it is difficult to identify the relevant pieces of information. Knowledge discovery seeks to obtain novel, interesting and useful information from large datasets. One key technique for that purpose is subgroup discovery. It aims at identifying descriptions for subsets of the data, which have an interesting distribution with respect to a predefined target concept. This work improves the efficiency and effectiveness of subgroup discovery in different directions.

For efficient exhaustive subgroup discovery, algorithmic improvements are proposed for three important variations of the standard setting: First, novel optimistic estimate bounds are derived for subgroup discovery with numeric target concepts. These allow for skipping the evaluation of large parts of the search space without influencing the results. Additionally, necessary adaptations to data structures for this setting are discussed. Second, for exceptional model mining, that is, subgroup discovery with a model over multiple attributes as target concept, a generic extension of the well-known FP-tree data structure is introduced. The modified data structure stores intermediate condensed data representations, which depend on the chosen model class, in the nodes of the trees. This allows the application for many popular model classes. Third, subgroup discovery with generalization-aware measures is investigated. These interestingness measures compare the target share or mean value in the subgroup with the respective maximum value in all its generalizations. For this setting, a novel method for deriving optimistic estimates is proposed. In contrast to previous approaches, the novel measures are not exclusively based on the anti-monotonicity of instance coverage, but also takes the difference of coverage between the subgroup and its generalizations into account. In all three areas, the advances lead to runtime improvements of more than an order of magnitude.

The second part of the contributions focuses on the *effectiveness* of subgroup discovery. These improvements aim to identify more interesting subgroups in practical applications. For that purpose, the concept of expectation-driven subgroup discovery is introduced as a new family of interestingness measures. It computes the score of a subgroup based on the difference between the actual target share and the target share that could be expected given the statistics for the separate influence factors that are combined to describe the subgroup. In doing so, previously undetected interesting subgroups are discovered, while other, partially redundant findings are suppressed.

Furthermore, this work also approaches practical issues of subgroup discovery: In that direction, the *VIKAMINE 2* tool is presented, which extends its predecessor with a rebuild user interface, novel algorithms for automatic discovery, new interactive mining techniques, as well novel options for result presentation and introspection. Finally, some real-world applications are described that utilized the presented techniques. These include the identification of influence factors on the success and satisfaction of university students and the description of locations using tagging data of geo-referenced images.

Acknowledgements

This work was supported by many people: First of all, I want to thank my supervisor Frank Puppe for all the open-minded discussions, his always open door, and for providing a great and enjoyable research environment. In addition, I would like to offer my special thanks to Martin Atzmueller and Andreas Hotho for their guidance, expertise and feedback.

Many thanks to my colleagues and friends at our working group, who helped me with countless technical, organizational and scientific issues: Alexander Hörnlein, Beate Navarro Bullock, Daniel Zoller, Georg Dietrich, Georg Fette, Joachim Baumeister, Jochen Reutelshöfer, Lena Schwemmlein, Marianus Ifland, Martin Becker, Martin Toepfer, Peter Klügl, Petra Braun, Philip-Daniel Beck, Reinhard Hatko, and Thomas Niebler.

I also want to thank my reviewers Arno Knobbe from the Universiteit Leiden, and Nada Lavrač from the Jožef Stefan Institute Ljubljana, who accompanied my scientific journey from my first international workshop to the final weeks of this thesis.

I dedicate this work to my family. They provided me the anchor for my life and never-ending encouragement on my way. Most of all, this is for my wife Eva-Maria, who went with me through ups and downs. We've made it!

Würzburg, December 2013 / May 2014

Contents

1 Introduction

1.1 Motivation

"It's a very sad thing that nowadays there is so little useless information."[1] As this quote of Oscar Wilde shows, information was a valuable commodity already more than one century ago. Almost each piece of data seemed to be potentially beneficial already in these times. In the last decades, the significance of information in our society only increased further, and it increased rapidly. In fact, it increased so much that some call the current period of human history the *information age*. Massive amounts of information are collected and stored every day in the most diverse domains. However, this abundance of information also has its downsides: *"What information consumes is rather obvious: it consumes the attention of its recipients. Hence, a wealth of information creates a poverty of attention and a need to allocate that attention efficiently among the overabundance of information sources that might consume it."*[2] This expresses a common concern: There is so much raw information available that it gets difficult to extract relevant and interesting *knowledge*. To support this course of action with current computer technology, the research field of *knowledge discovery in databases* has been established in order "to assist humans in extracting useful information (knowledge) from the rapidly growing volumes of digital data" [80].

For knowledge discovery in databases, a diverse set of methods has been proposed in the last decades. One of the key techniques is *subgroup discovery*. Subgroup discovery aims at finding descriptions of subsets of the data instances that show an interesting distribution with respect to a predefined target concept. It is a *descriptive* and *supervised* technique. Descriptive means that results are directly interpretable by human experts. Supervised expresses that the user specifies a certain property of interest, which is the focus of the knowledge discovery process.

Consider as an illustrating example a dataset from the medical domain: The data instances are given by a large set of patients with a certain disease. For each patient a variety of information is provided, e.g., age, gender, measurements about the current medical condition, and previous diseases. Furthermore it is known for each patient if the treatment with a certain drug was successful or not. This information is of specific interest and is thus used as the target concept for subgroup discovery. Assume that the treatment was overall successful in 30% of the cases. Then, one can speculate that there are some groups of patients, for which the treatment is especially well suited. For these patients, the treatment should have a higher success rate. For example, one could

[1] Wilde, Oscar: A Few Maxims For The Instruction Of The Over-Educated, Saturday Review, 1894.

[2] Simon, Herbert A., Designing Organizations for an Information-Rich World, in: Greenberger, Martin, Computers, Communication, and the Public Interest, The Johns Hopkins Press, 1971.

hypothesize that the treatment is in particular successful for young, female patients with high blood pressure. For each conjunction of selection expressions (e.g., gender is female AND age is young AND blood pressure is high), one hypothesis about the effect of the treatment in this group of people can be formed. Unfortunately, given a large set of describing properties, there is a huge set of possible conjunctive combinations. Therefore, it is clearly intractable to report the statistics for all possible hypotheses to a human expert. Instead, an automatic subgroup discovery algorithm can be used to select subgroups, which are supposedly interesting for humans, e.g., because the success rate of the treatment is significantly increased or decreased. Thus, one relevant finding identified by subgroup discovery could be stated as *"While the success rate of the treatment was only at 30 % of all cases, for the 30 young, female patients with high blood pressure, the success rate was at 75%."* Such findings are usually identified by performing a search through the set of candidate hypotheses and score each of them with a value that rates its supposed interestingness. In addition to this purely automatic approach, the involvement of human experts is favorable in order to guide the search in an iterative and interactive approach. This example used a binary property as the target concept (the treatment is either successful or not), but variations of subgroup discovery also employ numeric properties or even complex models over multiple attributes as target concepts.

Subgroup discovery is an established technique that has been successfully applied in numerous practical applications. Nonetheless, it still provides diverse challenges, which will be approached in this work.

1.2 Goal

The main challenges in the research on subgroup discovery, can be distinguished into two different categories:

The first type is concerned with the *efficiency of discovery algorithms*. In order to identify interesting patterns, a large number of candidates has to be evaluated. That is, the number of subgroup evaluations is high-order polynomial or even exponential (depending on the applied constraints) with respect to the number of selection expressions allowed to describe the subgroup. Therefore, fast algorithms are key in order to handle large-scale applications. However, the speed of subgroup discovery algorithms is not only an issue for large datasets. Since subgroup discovery involves an iterative and interactive process, automatic discovery is often executed repeatedly with slightly modified settings by a domain expert with limited time for the analysis. In such a scenario, the reduction of runtime from a few minutes to a few seconds can substantially improve the user experience. The efficiency of algorithms is well explored in the standard subgroup discovery setting with binary target concepts and simple interestingness measures. However, for variations of this task fast algorithms are yet to be developed, e.g., for subgroup discovery with a numeric target concept, for exceptional model mining (subgroup discovery with multiple target attributes) and for generalization-aware interestingness measures. This work aims at providing novel techniques that enable efficient subgroup discovery also in these more complex settings. In these directions, in particular *exhaustive mining*

techniques are explored, which can guarantee that the optimal results are returned with respect to the employed selection criteria.

The second type of challenges relates to the *effectiveness of subgroup discovery*. This describes that the discovered patterns are close to what is actually interesting or useful in real-world applications. One problem in particular is that results from automatic algorithms are often redundant, that is, they can be expected with respect to other findings, especially to more general subgroup descriptions. Since often only the top (e.g., the top 20) subgroups are presented to human experts, other, potentially interesting subgroups are not displayed in favor of these redundant findings. In that direction, this work seeks to find novel selection criteria to improve the quality of subgroup discovery results. Finally, the introduced techniques should also be available for interactive real world-applications.

1.3 Contributions

The first part of the contribution focuses on the efficiency of automatic discovery algorithms. That is, the same subgroups as in previous approaches are discovered, but in substantially less time:

In that direction, algorithms for fast subgroup discovery from literature are reviewed in detail. Instead of describing the algorithms one-by-one, improvements are categorized in three dimensions, that is, the strategies for candidate enumeration, the utilized data structures and the applied pruning strategies, which allow skipping parts of the evaluation without affecting the results. Algorithmic components for all three dimensions are discussed in detail. This allows describing actual algorithms concisely in terms of the three dimensions. The summary of related work is not exclusively focused on subgroup discovery algorithms, but also references contributions from related fields, even if these differ in the applied terminology or specific goal.

The first set of novel techniques is concerned with *subgroup discovery with a numeric target concept*. For this task, optimistic estimate bounds are developed for a large variety of interestingness measures. Applying these optimistic estimate bounds allows to prune large parts of the search space while maintaining the optimality of results. For some of the bounds closed form expressions, which are based on a few key statistics, can be determined. This is required for algorithms that use FP-tree-based data structures. Other, ordering-based optimistic estimates can be derived by checking multiple subsets of the subgroup's instances. While the computation of these bounds requires a full pass over all instance for each subgroup, it results in tighter bounds. Additionally, the adaptation of two different data structures to the numeric target setting is presented, that is, FP-trees and bitset-based data representations. Two novel algorithms implement these improvements: The *SD-Map** algorithm utilizes adapted FP-trees as data structures and optimistic estimates in closed form. The *NumBSD* algorithm exploits bitset-based data structures and ordering-based optimistic estimate bounds. Experimental evaluations show that both outperform previous approaches by an order of magnitude.

Another contribution focuses on the extension of the FP-tree data structure to the area of *exceptional model mining*. This extension of classical subgroup discovery uses a model over multiple attributes instead of a single attribute as target concept. Since different model classes have heterogeneous computational requirements, a generic data structure is necessary. For that purpose, the concept of valuation bases is introduced as an intermediate condensed data representation. While the structure of the FP-tree is maintained, the information that is stored in each tree node is replaced by a valuation basis depending on the model classes. This approach is suited for many, but not all model classes. A concise characterization of applicable model classes is provided by drawing an analogy to parallel data stream mining. Furthermore, examples of valuation bases are presented for many model classes. The implementation of the novel data structure in a new algorithm called GP-growth leads to massive speed-ups in comparison to a naive exhaustive depth-first search.

Generalization-aware interestingness measures improve traditional selection criteria for subgroup discovery by comparing the statistics of a subgroup not only to the statistics of the overall population, but also to the statistics of all its generalizations. In doing so, redundant findings are avoided and more interesting subgroups are discovered. For improved efficiency of subgroup discovery with these measures, a novel method of deriving optimistic estimates bounds is introduced. Unlike previous approaches, the novel optimistic estimates are not only based on the anti-monotonicity of instance coverage, but also take into account the difference of the coverage between a subgroup and its generalizations. Incorporating the new optimistic estimates in state-of-the-art algorithms improves the runtime requirements by more than an order of magnitude in many cases.

The second part of the contributions focuses on the *effectiveness* of subgroup discovery. These improvements focus on identifying more interesting subgroups in practical applications. In that direction, the concept of *expectation-driven subgroup discovery* is introduced. This variant employs a novel family of interestingness measures, which aims at more interesting subgroup discovery results. It assumes that the statistics for describing basic influence factors imply expectations on the statistics of complex patterns with conjunctive descriptions. The interestingness of a subgroup is then determined by the difference between the expected and the actual share of the target concept in the subgroup. A formal computation of expectation values is difficult, as expectations are inherently subjective. For this task, it is shown how established modeling techniques, such as the leaky-noisy-or model, can be transferred from the research of bayesian network construction. Several experiments including two real-world case studies demonstrate that the novel techniques detect qualitatively different patterns in comparison to previously proposed interestingness measures. New, previously undetected interesting subgroups are discovered, while other, partially redundant findings are suppressed.

The above mentioned core contributions are concerned with more theoretical, but generally applicable improvements. Further contributions focus on practical issues of subgroup discovery: Since subgroup discovery is not a purely automatic task, but an iterative and interactive process, proper tool support is required. For that purpose, the tool *VIKAMINE 2* is presented. *VIKAMINE 2* was developed in the context of this work, significantly improving and extending its predecessor *VIKAMINE* in different di-

rections. As core component, it provides a broad collection of state-of-the-art algorithms and interestingness measures for binary as well as numeric target concepts. Automatic and interactive mining options are accessible in a new, appealing graphical user interface, which enables easy extensibility by using the Eclipse RCP-framework. It also features novel techniques for interactive mining in large datasets, the effective presentation of results and the introspection of subgroups.

Finally, some real-world applications are presented, which did benefit from the proposed novel techniques. A long-term project in the educational domain applied subgroup discovery in order to identify influence factors, which affect the success and the satisfaction of university students. Another case study showed how techniques from subgroup discovery can be adapted for the analysis of geo-referenced tagging data. In this direction, meta-data from the Flickr platform could be utilized to obtain descriptions for specific geographic locations. Further application examples included pattern mining for improved information extraction with conditional random fields, an industrial application that was concerned with the fault rate of products, and a challenge dataset regarding gene data analysis.

1.4 Structure of the Work

The remainder of this work is structured in nine chapters, which are to a high degree self-contained and intelligible in themselves:

Chapters 2 and 3 review previous research on subgroup discovery. Chapter 2 discusses the fundamentals of subgroup discovery, such as possible selection criteria, as well as general issues, e.g., process models or the statistical significance of discoveries. Chapter 3 focuses on algorithmic approaches from literature. In that direction, it discusses enumeration strategies, data structures, and pruning strategies in detail and provides an overview on previously presented algorithms.

Chapters 4 to 7 present the main theoretical contributions of this work: Chapter 4 introduces novel techniques for subgroup discovery with numeric target concepts, i.e., new optimistic estimate bounds for advanced pruning of the search space and the adaption of data structures to the numeric target setting. Chapter 5 focuses on exceptional model mining, that is, subgroup discovery with a model over multiple attributes as target concept. In this direction, it is shown how the well-known FP-tree data structure can be adapted for this setting, and for which model classes this can be accomplished. Chapter 6 presents difference-based estimates as a novel, generic scheme for optimistic estimate pruning, which can be applied for subgroup discovery with generalization-aware interestingness measures. In contrast to these efficiency improvements, Chapter 7 is concerned with the effectiveness of subgroup discovery. For that purpose, a novel class of interestingness measures, that is, expectation-driven interestingness measures, is introduced.

Chapter 8 and Chapter 9 discuss practical issues of subgroup discovery: Chapter 8 presents the interactive subgroup discovery tool *VIKAMINE 2*, while Chapter 9 reports on some successful real-world applications of subgroup discovery.

Finally, Chapter 10 concludes the work with a summary of results and an outlook on future work.

2 The Subgroup Discovery Problem

This chapter provides a general overview on subgroup discovery. First, it gives informal and formal definitions of the subgroup discovery task and discusses its different components in detail: the dataset, the search space, the target concept, and selection criteria, i.e., interestingness measures. Next, we review some important general issues in the research on subgroup discovery: the incorporation of background knowledge, process models for interactive mining, the complexity of the subgroup discovery task, and the question of the statistical significance of discoveries. Afterwards, the relationship between subgroup discovery and other data mining tasks is discussed. The chapter finishes with a summary of notations used in this work.

2.1 Definition of Subgroup Discovery

Kralj Novak et al. give the following definition of subgroup discovery:

Definition 1 "Given a dataset of individuals and a property of those individuals that we are interested in, find dataset subgroups that are statistically 'most interesting', for example, are as large as possible and have the most unusual (distributional) characteristics with respect to the property of interest." [158] □

Although other definitions of subgroup discovery in literature do slightly differ in details, there is a general agreement on the overall nature of the task, see for example [140, 260, 117].

In this context, subgroups are induced by descriptions of properties of the individuals, which a user can directly understand. The definition requires a "statistically interesting" distribution of the property of interest. In contrast to other definitions, see for example [243], this is not necessarily associated with a deviation in comparison to the distribution of the property of interest in the overall dataset. Although the definition implies that subgroups are selected based on statistical characteristics, it does not postulate the statistical significance of discovered subgroups. However, many interestingness measures are nonetheless derived from popular statistical significance tests, see Section 2.5.

Next, the subgroup discovery task is defined formally and the formal notations of this work are introduced: A *subgroup discovery task* is specified by a quadruple $(\mathcal{D}, \Sigma, T, Q)$. $\mathcal{D}$ is a dataset. Σ defines the *search space* of candidate subgroup descriptions in this dataset. The *target concept* T specifies the property of interest for this discovery task. Finally, Q defines a set of selection criteria that depends on the target concept. Selection criteria are constraints on the subgroups and/or an interestingness measure that scores candidate subgroups, e.g., by rating the distributions of the target concept and the

number of covered instances. An additional parameter k can specify how many subgroup patterns are returned. The result of the subgroup discovery task is either the set of all subgroups in the search space that satisfy all provided constraints or the best k subgroups in the search space according to the chosen interestingness measure.

We discuss the dataset $\mathcal{D}$, the search space Σ, the target concept T, and selection criteria Q in the next sections in detail.

2.2 Dataset

A dataset $\mathcal{D} = (\mathcal{I}, \mathcal{A})$ is formally defined as an ordered pair of a set of *instances* (also called individuals, cases or data records) $\mathcal{I} = c_1, c_2, \ldots, c_y$ and a set of attributes $\mathcal{A} = A_1, A_2, \ldots, A_z$. Each attribute $A_m : \mathcal{I} \rightarrow dom(A_m)$ is a function that indicates a characteristic of an instance by mapping it to a value in its range. Consequently, $A_m(c)$ denotes the value of the attribute A_m for the instance c. We distinguish between *numeric* attributes, which map instances to real numbers, and *nominal* attributes that map instances to finite sets of values: $\mathcal{A}^{num} = \{A_m \in \mathcal{A} \,|\, dom(A_m) = \mathbb{R}\}$, $\mathcal{A}^{nom} = \{A_m \in \mathcal{A} \,|\, |dom(A_m)| < \infty\}$. For some nominal attributes, an additional ordering of its values is provided. In this case, the attributes are called *ordinal attributes*.

As an example, in a dataset from the medical domain the instances $\mathcal{I}$ are given by a set of patients. Properties of the patient such as age, gender or the measured blood pressure are indicated by respective attributes. The attribute *age* with $dom(age) = [0, 140]$ is numeric, the attribute *gender* with $dom(gender) = \{male, female\}$ is nominal and the attribute *blood_pressure* with $dom(blood\_pressure) = \{low, ok, high, very\ high\}$ is ordinal.

The above definition of subgroup discovery is not necessarily limited to data in the form of a single table. However, this work uses the so-called *single-table-assumption* [261]: The complete data is contained in one single table with instances of the dataset as lines and the characteristics (attributes) of instances as columns. For many practical applications, this is a strongly simplifying assumption. Therefore, a variety of subgroup discovery algorithms has been proposed that are specialized for the *multi-relational* setting with multiple data tables, see for example [260, 169]. Nevertheless, a data source that consists of multiple tables can always be transformed in a single table by *propositionalization* and *aggregation* [149, 159, 161], although this can result in very large data tables and/or loss of information.

2.3 Search Space

The search space Σ consists of a large set of *subgroup descriptions* (also called patterns). Each subgroup description is composed of selection expressions that are called *selectors* (also named *conditions* or *basic patterns*). A selector $sel : \mathcal{I} \rightarrow \{true, false\}$ is a boolean function that describes a set of instances based on their attribute values. It is directly interpretable by humans. As an example, the selector *gender=male* is true for all instances, for which the attribute *gender* takes the value *male*.

Selectors are combined into subgroup descriptions by boolean formulas using a propositional description language. The set of instances, for which this formula evaluates to true, is called the *subgroup cover*.

The by far most common setting considers only conjunctive combinations of selectors since such subgroup descriptions are easier to comprehend and to interpret by human domain experts. Additionally, the restriction to such descriptions controls the combinatorial explosion of the number of patterns, which can be generated from a set of selectors. Furthermore, the loss of expressiveness is limited since one can reinterpret a *set* of discovered subgroup descriptions as a disjunction of the individual subgroup descriptions. For these reasons, most research as well as this work is concerned exclusively on conjunctive subgroup descriptions.

In the case of strictly conjunctive subgroup descriptions, a pattern $P = sel_1 \wedge sel_2 \wedge \ldots \wedge sel_d$ covers all instances, for which all selectors sel_i evaluate to *true*. It is also written equivalently as a set of selectors: $P = \{sel_1, sel_2, \ldots, sel_d\}$. In particular, the pattern $P_\emptyset = \emptyset$, which is given by the empty conjunction, describes all instances in the dataset. We write $P(c)$, $c \in \mathcal{I}$ for the boolean value that indicates if an instance i is covered by the pattern P. $sg(P) = \{c \in \mathcal{I} | P(c) = true\}$ denotes the set of instances, which are covered by P. $i_P = |sg(P)|$ is the count of these instances.

The *complement* $\neg P$ of a subgroup pattern P covers all instances, which are not covered by P. A subgroup P_{gen} is called a *generalization* of another subgroup P_{spec} iff $P_{gen} \subset P_{spec}$. P_{spec} is then a *specialization* of P_{gen}. Trivially, a generalization covers all instances, which are covered by its specializations:

$$P_{gen} \subset P_{spec} \Rightarrow sg(P_{gen}) \supseteq sg(P_{spec})$$

In principle, the individual selectors, which form subgroup descriptions, can be constructed over more than one attribute. As an example, a selector $sel_{A_1 > A_2} = true \Leftrightarrow A_1(i) > A_2(i)$ could indicate that in this instance the numeric attribute A_1 has a higher value than A_2. Employing such selectors in a meaningful way relies on background knowledge of the respective attribute, i.e., it must be known that both attributes work on the same scale and appear in a related context. Such knowledge can be incorporated in a pre-processing step, e.g., by defining a boolean attribute, which is true, iff the value of A_1 is higher than the value of A_2. We focus in the following on the more common case, in which each selector is extracted from a single attribute.

2.3.1 Nominal Attributes

Typical selectors for nominal attributes check for value-identity: $sel_{A_j = v_k}(c) = true \Leftrightarrow A_j(c) = v_k$. An example for this is the above mentioned selector $sel_{gender=male}$. For each value in the domain of each nominal attribute one selector is built. These selectors are *exclusive*: No interesting subgroup description contains more than one selector for each attribute since no instances are covered otherwise. For example, the subgroup description *gender=male* $\wedge$ *gender=female* cannot contain any instances since for each instance the attribute *gender* either takes the value *male* or the value *female*. Using

selectors based on value-identity is the most simple and most widespread method to obtain selectors for subgroup discovery.

An alternative for nominal attributes is to use negations of values as selectors instead of using the values directly: $sel_{A_j \neq v_k}(c) = true \Leftrightarrow A_j(c) \neq v_k$. As before, one selector is build for each value in the domain of each attribute. Although these approaches look very similar at first sight, the expressiveness of subgroup descriptions using negated selectors is higher: For each subgroup description that uses only value-identity based (non-negated) selectors, there is an equivalent conjunctive subgroup description that uses only negated selectors. This is not the case the other way round. Consider for example an attribute *color* with the four possible values $dom(color) = \{red, green, blue, yellow\}$. Then, the subgroup description $(color = red)$ describes the same instances as the description $(color \neq green \wedge color \neq blue \wedge color \neq yellow)$. On the other hand, there is no equivalent conjunctive subgroup description for $(color \neq red)$ that uses only value-identity selectors. In contrast to value-identity selectors, combining multiple negated selectors for a single attribute can lead to interesting subgroups. This is, because these subgroup descriptions are semantically equivalent to internal *disjunctions* of values for this attribute, although formally only conjunctions are used, see [22]. In the previous example, the subgroup description $color \neq green \wedge color \neq blue$ is equivalent to the subgroup description $color = (red \vee yellow)$ or the subgroup description $color \neq green$ is equivalent to the pattern $color = (green \vee blue \vee yellow)$.

2.3.2 Numeric Attributes

Determining useful selectors for numeric attributes is more challenging than in the nominal case. Although value-identity selectors can also be employed for numeric attributes, this leads only to reasonable results, if the attribute has a limited range and in particular does not exploit a continuous scale. For example, a selector like *numberOfChildren = 1* is perfectly fine. On the other hand, a selector such as *creditAmount = 3256.15* is hard to interpret and will most probably only cover very few instances. Therefore, it should not be applied to mining tasks.

Instead, selectors for numeric attributes are usually specified by intervals over the domain of the numeric attribute: $sel_{A_j \in]lb,ub[}(c) = true \Leftrightarrow lb < A_j(c) \wedge A_j(c) < ub$. The attribute values lb, ub that appear in the selectors of a numeric attribute are called *cut-points*. Open, half-open or closed intervals can be used. For example, potential selectors for a numeric attribute *age* can be all instances with an age of at least 20 and at most 40 ($sel_{age \in [20,40]}$), all instances with an age of more than 40 and at most 60 ($sel_{age \in]40,60]}$) or all instances with an age greater than 60 ($sel_{age \in]60,\infty[}$).

In subgroup discovery, there are two approaches to identify the best set of intervals. First, using *online-discretization* the best intervals are determined inside the mining algorithms that performs the automatic search for the best subgroups. For that purpose, specialized methods have been designed that allow for identifying suited intervals efficiently, see Section 3.6.2 for details.

Second, the intervals used as selectors are determined before the search in an *offline-discretization* step. Then, any subgroup discovery algorithm can be used later on. This

approach is taken in most scenarios in literature. For the task of offline-discretization, a wide variety of methods has been proposed, which are distinguished in *knowledge-based* and *automatic* discretization methods. For knowledge-based discretization, the cut-points are determined manually by domain experts of the application domain. This yields several advantages: first, the bounds can be interpreted well by the experts. Additionally, they are inline with the bounds established in the domain and thus allow for comparison with previous research results in the application domain. Furthermore, since fewer parameters are determined by automatic search the multi-comparison adjustments necessary for statistical significance tests are considerably lower, see Section 2.9. However, the manual elicitation of bounds requires the cooperation with domain experts that provide the necessary knowledge. In addition, this task is often time-consuming and potentially error-prone, especially if the dataset contains large amounts of numeric attributes or if collaborating domain experts have little experience with the applied tools and techniques. This problem is known as the *knowledge acquisition bottleneck* in the research on expert systems, see [114, 113, 90]

For this reason automatic methods are often preferred to knowledge-based discretization. For a numeric attribute that takes v different values in a dataset, there are $(v-1)$ potential cut-points between the attribute values that can be used for discretization. The task of selecting a beneficial and meaningful subset of these cut-points is a well studied task in data mining beyond the research of subgroup discovery, see [157] and [94] for recent overviews. Discretization methods for a numeric attribute A, which are especially suited for subgroup discovery, include:

- *Equal-width discretization:* For this technique the minimum and the maximum values of A are determined. This range is then divided into k intervals of equal size, where k is a user specified parameter. Note, that the difference between two neighboring cut-points using this method is constant at $\frac{\max(A)-\min(A)}{k}$. However, the number of instances in the dataset, for which the respective attribute value is in a certain interval, differs and can also be empty in extreme cases. Another problem is that this discretization method is heavily influenced by outliers of the attribute, which ideally should be removed beforehand.

- *Equal-frequency discretization:* For a user specified parameter k, this method determines cut-points, such that each interval between two neighboring cut-points covers $\frac{|\mathcal{I}|}{k}$ instances. The width of the intervals differs. This technique is less influenced by outliers. An advantage of equal-frequency as well as equal-width discretization is that they are easy to explain and to justify to domain experts.

- *3-4-5 rule:* The 3-4-5 rule discretizes an attribute into easy-to-read and therefore "natural" segments [110], which is in line with the end-user oriented task of subgroup discovery. In a top-down approach, the range of the attribute A is split into three, four, or five sub-intervals depending on the difference in the most significant digit in the attribute range. For example, if A takes a minimum value of 21000 and a maximum value of 60000, then the application of the 3-4-5 rule will generate the four sub-intervals $[21000, 30000]$, $]30000, 40000]$, $]40000, 50000]$ and

]50000, 60000]. Dependent on the number of desired cut-points the operation is then applied recursively on the generated sub-intervals.

The previously described discretization methods are *unsupervised* since the discretization is performed independent of the chosen target concept. In contrast, the next techniques take the (boolean) target concept into account and are therefore categorized as *supervised* discretization methods.

- *Entropy-based discretization:* Entropy-based discretization [78] uses a top-down approach: It starts with the range of the attribute A as a single interval. Then, cut-points are added one-by-one, splitting this interval into more and more smaller ones. New cut-points are selected in a greedy approach, such that the *entropy* of the interval with respect to the target concept is maximized. This aims at cut-points that separate instances with positive and negative target concepts. In the original approach, cut-points are added until a stop-criterion based on the minimum description length principle [216] is met. However, the approach is easy to adapt such that it stops after a user specified number of cut-points has been determined.

- *Chi-Merge discretization:* Chi-merge discretization [136, 183] uses a bottom-up strategy. At the beginning, each attribute value of A forms its own interval. Then, recursively two adjacent intervals are merged until a stopping criterion, e.g., a predefined number of intervals, is met. The two intervals that are joined are selected according to a χ^2 test with respect to the target attribute: The two intervals with the smallest χ^2 value are merged, since this indicates a similar distribution of the target concept, see also [110].

Each discretization method results in a set of cut-points C. These can be transformed into selectors in two different ways. Either one selector is built for every interval given by a pair of adjacent cut-points. This leads to $|C| + 1$ selectors. For example, assume that the cut-points determined by a discretization method were $C_{example} = \{10, 20, 30\}$. Then, the intervals for the selectors in the search space are $]-\infty, 10],]10, 20],]20, 30]$ and $[30, \infty[$. In this case, the selectors are exclusive: conjunctions of two or more of these selector have not to be considered in subgroup discovery since no instance is covered by more than one of these selectors.

As an alternative, for each cut-point two selectors can be created: One that covers a half-open interval from $-\infty$ to the cut-point and one from the cut-point to $+\infty$. This leads to $2 \cdot |C|$ selectors. In doing so, each interval bounded by any two cut-points is described by a conjunction of these selectors. Continuing the above example, the selectors created using the second approach for the set of cut-points $C_{example}$ are $]-\infty, 10[,]-\infty, 20[,]-\infty, 30[, [10, \infty[, [20, \infty[$ and $[30, \infty[$. A subgroup description for the interval $[10, 30]$ results from the conjunction of the selectors corresponding to the intervals $]-\infty, 30]$ and $[10, \infty[$.

2.4 Target Concept

The target concept T defines the property of interest for a subgroup discovery task. The task substantially differs depending on the type of the target concept, that is, for binary targets, numeric targets or complex targets, such as in exceptional model mining.

2.4.1 Binary Target Concepts

The most common setting for subgroup discovery uses a binary (boolean) target concept. In this case, the property of interest is specified by a pattern P_T. For any subgroup description P we call the instances, which are also covered by P_T, *positive instances* (denoted as $tp(P)$). The other instances of P are described as negative instances (denoted as $fp(P)$). For shorter notation, the number of positive (negative) instances covered by the subgroup description P is also written as $p_P = |tp(P)|$ ($n_P = |fp(P)|$). The distribution of the target concept for a subgroup description is then completely described by the *target share* $\tau_P = \frac{p_P}{p_P+n_P} = \frac{p_P}{i_P}$, that is, the share of positive instances within the subgroup. The overall task is then to identify subgroup descriptions, in which the target share is either unexpectedly high or unexpectedly low. In most applications, a binary target concept is specified by a single selector, e.g., *class = good*, but this is not a necessary requirement.

2.4.2 Numeric Target Concepts

In many applications of subgroup discovery the property of interest is numeric. In these scenarios it is specified by a numeric attribute $A \in \mathcal{A}^{num}$. In this case, the *target value* $T(i)$ is for each instance i of the dataset given by the value of this numeric attribute.

In general, the case of numeric target attributes can be transformed back to the binary case by applying one of the discretization techniques described above. For example, using the target variable *age* with $dom(age) = [0, 140]$ the group "older people" could be defined using the interval $[80, 140]$. A significant subgroup could then be formulated as follows: *"While in the general dataset only 6% of the people are older than 80, in the subgroup described by* xy *it is 12%"*. As discussed above, such thresholds are often difficult to determine. Additionally, information on the distribution of the target attribute is lost. This can produce misleading results. Using the boolean target concept of our example, a subgroup, which contains many people aged between 70 and 80, will not be regarded as a subgroup of "older people" – perhaps in contrast to the expectations of the user. Furthermore, using this discretization there is not necessarily a difference between a subgroup, in which the majority of people is around 60 years old and a subgroup in which the majority is around 20 years old. Thus, crucial information is possibly hidden.

Therefore, utilizing the complete distribution of the numeric target attribute is advantageous. The distribution of a numeric target attribute in a subgroup is more difficult to describe than the distribution of a binary target pattern: It is given by a multi-set of real values instead of just the numbers of positive and negative instances. Thus, most of the time the target distribution for a subgroup description P is compared with respect

to one or more distributional properties, e.g., the mean value μ_P, the median med_P or the variance σ_P^2 of the numeric target attribute. For example, an interesting subgroup based on the mean values can be formulated as: *"While in the general dataset the mean age is* 56 *years, in the subgroup described by* xy *it is* 62*"*.

For a detailed discussion of interestingness measures in the case of numeric target attributes, we refer to Section 2.5.3.3.

2.4.3 Complex Target Concepts

Beside the standard binary and numeric target concepts, more complex target concepts have been proposed for subgroup discovery in the last decade: *Multi-class subgroup discovery* [3, 2] uses a nominal attribute with more than two values as target concept. A subgroup is considered as interesting if the share of one or more values of the target attribute differs significantly from the expectation.

A recently developed variant of subgroup discovery is *exceptional model mining* [170]. In contrast to traditional subgroup discovery, the property of interest is not specified by a single attribute, but by a set of *model attributes*. In exceptional model mining, a *model* is built for each subgroup. It consists of a *model class*, which is fixed for a specific mining task, and *model parameters*, which depend on the values of the model attributes in the respective subgroup. The goal of exceptional model mining is to identify subgroup descriptions, for which the model parameters differ significantly from the parameters of the model built from the entire dataset.

A simple example of a model class is the correlation model, which requires two numeric model attributes: It has only a single parameter, that is, the Pearson correlation coefficient between these attributes. The task is then, to identify subgroups, in which the correlation between two numeric attributes is especially strong. An exemplary finding could be formulated as: *"While in the overall dataset the correlation between income and age is only small with a correlation coefficient of* 0.1*, there is a stronger correlation between these two attributes in the subdataset described by* xy*. For these instances, the correlation coefficient is* 0.4*."* Other model classes and according interestingness measures are presented in Section 2.5.3.4.

Exceptional model mining includes traditional subgroup discovery with binary or numeric targets as a special case, if the target shares or the mean values are considered a very simple model over a single model attribute.

2.5 Selection Criteria

The task of subgroup discovery is to select specifically those subgroups from the huge number of possible candidates in the search space, which are probably interesting to the user. For this task, a large amount of interestingness measures has been proposed in literature in the last decades. Since selection criteria directly determine the results of subgroup discovery, it is crucial for successful applications to choose the selection criteria carefully.

This section provides an overview on selection criteria for subgroup discovery. It starts by presenting general criteria for interestingness. Then, two different approaches to the application of selection criteria are compared: constraints and interestingness measures. Next, some important exemplary interestingness measures for different types of target concepts are discussed. In particular for numeric target concepts, a comprehensive overview on interestingness measures is provided. Afterwards, two more specific issues for determining interesting subgroups are investigated: The incorporation of generalizations into the assessment of subgroups and the problem of redundant discoveries.

2.5.1 Criteria for Interestingness

Although a huge amount of selection criteria have been proposed in literature, most of them are based on a few underlying intuitions and goals.

According to Fayyad et al., knowledge discovery in databases should in general lead to the identification of patterns that are *valid*, *novel*, *potentially useful* and *understandable* [79]. Validity describes that results can be transferred to new data with "some degree of certainty". Novelty means that the findings differ from previous or expected results. This is possibly, but not necessarily also considering background knowledge. Usefulness expresses that discoveries should ideally have an implication in the application domain, e.g., by suggesting a change in the course of action such as the treatment of a patient. Finally, understandability describes that discovered patterns should be directly interpretable by humans and lead to a better understanding of the underlying data in the best case.

Similarly, Major and Mangano presented four criteria to select patterns: *performance*, *simplicity*, *novelty* and (statistical) *significance* [188]. Here, performance of a pattern is determined by the generality (coverage) and the unusualness (increase of the target share) of the rule.

In a more precise approach, Geng and Hamilton break down interestingness into nine different criteria, which partially correlate with each other [97]:

1. *Conciseness:* A pattern is concise if it contains only few selectors. The result overall should only contain a limited amount of patterns.
2. *Generality/Coverage:* A pattern should cover a large part of the data.
3. *Reliability:* A pattern (in form of a rule) should have a high accuracy (confidence).
4. *Peculiarity:* A pattern is possibly more interesting if it covers outliers in the data.
5. *Diversity:* A set of patterns is often more interesting if the patterns differ from each other.
6. *Novelty:* An interesting pattern is not covered by the users' background knowledge and cannot be derived from other patterns.
7. *Surprisingness:* A pattern is interesting if it contradicts background knowledge or previously discovered findings.

8. *Utility.* A pattern is more interesting if it contributes to a users goal with respect to *utility functions* that the user defined in addition to the raw data, e.g., by weighting attributes.

9. *Actionability/Applicability:* A pattern should influence future decisions in the application domain.

These criteria are categorized by Geng and Hamilton into *objective measures*, *subjective measures* and *semantics-based measures*. Objective criteria can be computed only using the raw data, e.g., by using statistical or information theoretic measures. Subjective measures like novelty and surprisingness involve information on the users prior knowledge, see [137, 227, 228]. Therefore, these measures are more difficult to determine in a purely automatic process. Instead, an interactive and iterative approach is preferred, see Section 2.7. Semantics-based measures such as utility and actionability incorporate the meanings of the subgroup descriptions in the application domain. They are considered to be a sub-type of subjective measures by some authors [227, 265]. Since in many application scenarios no background information in addition to the raw data is available, purely objective measures are by far the most often utilized ones.

2.5.2 Applying Selection Criteria: Interestingness Measures versus Constraints

Criteria to select a set of interesting patterns from the search space can be applied in different ways. The two main approaches are *interestingness measures* and *constraints*.

Interestingness measures assign a real number to each subgroup pattern. That number reflects its supposed interestingness to the user. Additionally, the user specifies the maximum number of subgroups k that the result set should contain. An automatic subgroup discovery algorithm then returns the k subgroup patterns, which have the highest score with respect to the chosen interestingness measures. This is called *top-k* subgroup discovery. Other names for interestingness measures are *quality functions* (e.g., in [141]) or *evaluation functions* (e.g., in [140, 260]). In the related field of classification rules, the term *heuristics* is also used frequently [87]. An overview on interestingness measures for subgroup discovery is provided in the next section.

As an alternative to scoring with interestingness measures, a user can define a set of filter criteria (*constraints*) for the mining task, see for example [166, 200]. Then, all subgroups that satisfy all of these constraints are returned to the user. Popular constraints for a pattern P include (see for example [254]):

- *Description constraints* specify a maximum number of selectors a subgroup description contains: $|P| \leq \theta_{dc}$.
- *Coverage constraints* define a minimum number of instances which must be covered by each subgroup in the result: $i_P \geq \theta_{cc}$. For binary target concepts, also a minimum number of positive instance is used instead: $p_P \geq \theta_{cc^+}$.
- *Deviation constraints* set a minimum target share or a minimum deviation from the target share in the overall dataset for all resulting instances: $\tau_P \geq \theta_{dc}$ or $|\tau_P - \tau_\emptyset| \geq \theta_{dcc}$.

- *Significance constraints* filter out patterns that do not show a statistical significant deviation of the target concept: $\rho(P) \leq \theta_{sc}$, where $\rho(P)$ is the p-value according to a statistical test, see also [251].

All constraint thresholds θ_x are user specified constants. Other constraints test the productivity and redundancy of subgroups, see Sections 2.5.4 and 2.5.5 for detailed descriptions.

Constraints and interestingness measures are closely related to each other: Each constraint $C(P)$ on a single subgroup can be incorporated in an interestingness measure $q(P)$ by adapting the measure as following:

$$q'(P) = \begin{cases} q(P), if\ C(P) = true \\ 0, else \end{cases}$$

On the other hand, the value of any interestingness measure can be used as a constraint that is satisfied, iff the interestingness score is above a specified threshold c:

$$C(P) = true \Leftrightarrow q(P) > c$$

An advantage of top-k subgroup discovery is that the utilized interestingness measure implies a ranking of the resulting subgroups. This allows the user to inspect the supposedly most interesting subgroups first. On the other hand, constraints are often easier to explain to domain experts without mathematical expertise. The use of interestingness measures allows, but also requires, to specify the number of subgroup descriptions in the result set. In contrast, the number of result subgroups can freely vary for constraint-based approaches.

Most of the time, a mixture of both selection criteria is applied: The search space is filtered by a few basic constraints, e.g., in order to limit the number of selectors in a subgroup description. The remaining subgroups are scored according to an interestingness measure. The subgroups with the best scores that satisfy all constraints are returned to the user in the order implied by the interestingness measure. This approach has been described as *filtered-top-k association discovery* [254]. The remainder of this work focuses on this common setting.

An alternative technique that incorporates multiple interestingness measures $q_1, \ldots, q_m$ was proposed by Soulet et al. [229]. It uses the notion of dominance: A pattern P_a *dominates* a pattern P_b, iff the score of P_a for each interestingness measure q_j is not less than for P_b ($q_j(P_a) \geq q_j(P_b)$) and is truly greater for at least one interestingness measure q_{j^*} ($q_{j^*}(P_a) > q_{j^*}(P_b)$). A search task then returns all subgroups, which are not dominated by any other subgroup pattern.

2.5.3 Interestingness Measures

Diverse interestingness measures have been proposed for different types of target concepts. In the next section, measures for binary, numeric and complex target concepts are discussed.

2.5.3.1 Order Equivalence

Interestingness measures imply an ordering of the subgroups in the search space. Two interestingness measure $q_1(P)$ and $q_2(P)$, which imply the identical order for any pair of subgroups of a dataset, are called *order equivalent*, denoted as $q_1(P) \sim q_2(P)$. Obviously, order equivalent interestingness measure lead to identical results in an automatic top-k-search. For example, an order equivalent measure is generated by multiplying an interestingness measure with a factor that is constant for all subgroups in the search space, e.g., the number of instances in the dataset.

Many interestingness measures have been proposed as measures for (association or classification) rules $P \rightarrow X$ with arbitrary left- and right-hand sides. In contrast to this, the right-hand side of a rule for subgroup discovery is fixed, i.e., it is always given by the target concept. Therefore, some measures are order equivalent in the context of subgroup discovery, though they are not for arbitrary rules. For example, consider the two popular measures *added value* $q_{av}(P) = P(X|A) - P(X)$ and *lift* $q_{lift} = \frac{P(X|A)}{P(X)}$, where $P(X)$ is the probability of the right-hand side of the rule in the overall dataset and $P(X|A)$ the conditioned probability of the X, if the left-hand side of the rule applies. Since the probability for the right-hand rule side $P(X)$ differs for arbitrary rules, both measures are not order equivalent in the general case. However, for subgroup discovery the right-hand side of the rule and therefore the probability $P(X)$ is equal for all subgroups: $P(X) = \tau_\emptyset$ is the target share in the overall dataset. Thus, both measures differ from $P(X|A) = \tau_A$ only by a constant factor or summand respectively. Therefore, both measures are order equivalent with τ_A and thus order equivalent with each other.

2.5.3.2 Interestingness Measures for Binary Target Concepts

Interestingness measures for subgroups with binary targets are most often discussed in literature since the same measures are also applied for association rule mining.

The distribution of the binary target concept Interestingness measures use the statistical distribution of the target concept to score a subgroup pattern P. In case of a binary target concept, this distribution can be displayed in a contingency table, see Table 2.1. It shows the number of positive instances, negative instances and the overall number of instances for a pattern P, its complement $\neg P$ and the overall dataset. The table is fully determined by only four values: For example, the number of positives covered by the subgroup p_P, the number of negatives covered by the subgroup n_P, the number of positives in the overall dataset $p_\emptyset$ and the number of negatives in the overall dataset $n_\emptyset$ allow to compute the remaining entries of the contingency table.

The share of the target concept τ_P in the subgroup and in the overall dataset $\tau_\emptyset$ can be derived from the table entries: $\tau_P = \frac{p_P}{i_P}, \tau_\emptyset = \frac{p_\emptyset}{i_\emptyset}$. The majority of interestingness measures for subgroup discovery with binary target concepts are based exclusively on the entries of a contingency table and derived statistics.

Table 2.1: A contingency table describes the distribution of a binary target concept T for a subgroup pattern P, its complement $\neg P$, and the overall dataset $\mathcal{I}$.

	T	$\neg T$	$Total$
P	p_P	n_P	i_P
$\neg P$	$p_{\neg P}$	$n_{\neg P}$	$i_{\neg P}$
$dataset$	$p_\emptyset$	$n_\emptyset$	$i_\emptyset$

Axioms for binary interestingness measures Piatetsky-Shapiro and Frawley investigated objective interestingness measures that identify influence factors, which cause an increase of the target share. For this task, they postulated three axioms that all interestingness measures $q(P)$ in the binary setting should satisfy [208]. Major and Mangano added a fourth axiom to this set [188], see also [140, 14]:

1. $q(P) = 0$, if $\tau_P = \tau_\emptyset$.
2. $q(P)$ monotonically increases in τ_P for a fixed i_P.
3. $q(P)$ monotonically decreases in i_P if $\tau_P = \frac{c}{i_P}$ with a fixed constant c.
4. $q(P)$ monotonically increases in i_P for a fixed $\tau_P > \tau_\emptyset$.

The first axiom denotes that no pattern is interesting if it has the same target share as the overall dataset. The second axiom describes that given two subgroups with the same coverage, the one with a higher target share is more interesting. According to the third axiom, a subgroup with a certain number of positives is less interesting, if the subgroup covers more instances overall, i.e., if it covers more negative instances. The fourth axioms expresses that a subgroup with an increased target share in comparison to the overall dataset is more interesting, if it covers more instances. Although these axioms seem intuitive, they are not always satisfied by interestingness measures, see [97]. Other properties include the effects of permutations in the contingency table [234] and the *null invariance*: Null variance postulates that interestingness measures should be independent from instances, which are neither covered by the subgroup nor by the target concept [234, 263, 201]. This, however, contradicts a probabilistic interpretation of the data.

Example interestingness measures For binary target concepts, an abundant amount of interestingness measures have been proposed that score the statistics contained in a contingency table. For example, Geng and Hamilton list no less than 38 measures in their respective survey [97]. Therefore, we restrict ourselves to the most wide-spread measures in the context of subgroup discovery here and refer to the existing seminal works in literature [234, 190, 97] for comprehensive overviews.

For subgroup discovery, Kloesgen [140] proposed a now wide-spread family of interestingness measures q^a_{Kl}. It trades off the generality (size) of the subgroup i_P versus the difference between the target shares in the subgroup and the overall dataset $\tau_P - \tau_\emptyset$:

$$q^a_{Kl^*}(P) = \left(\frac{i_P}{i_\emptyset}\right)^a \cdot (\tau_P - \tau_\emptyset), \text{ which is order equivalent to}$$
$$q^a_{Kl}(P) = {i_P}^a \cdot (\tau_P - \tau_\emptyset)$$

The parameter a is usually chosen in the interval $[0, 1]$. Low parameter values for a prefer subgroups with a high deviation in the target share, even if only few instances are covered. In contrast, high values for the parameter a result in large subgroups with a possibly limited deviation in the target share. Choosing the right parameter a can be integrated in an iterative and interactive process, see Section 2.7: After an automatic subgroup discovery task is performed, results are inspected by human experts. If the subgroups cover too few instances, then the parameter a is increased before another automatic discovery is performed. If the deviation of the subgroups is considered too small, then a is decreased instead.

This family of interestingness measures includes several popular interestingness measures for specific values of a: Setting $a = 1$ results in the weighted relative accuracy [167]. For $a = 0.5$, it leads to simplified version of the binomial test (see below) and for $a = 0$ it is order equivalent to the added value and the lift measure, see Section 2.5.3.1. The approach of introducing a parameter to weight between generality and deviation of the target concept has also been followed by other authors [91, 133], see also [141]. However, the above formalization is the most wide-spread for subgroup discovery.

In their above form, the Kloesgen measures focus on subgroups that show an increase of the target share. To obtain a symmetric variant that also discovers patterns with a decrease in the target share, one can replace the difference in the measure with the absolute difference:

$$q^a_{sym}(P) = {i_P}^a \cdot |\tau_P - \tau_\emptyset|$$

Another popular interestingness measure is the (full) binomial test interestingness measure. It is given by [141]:

$$q_{bt}(P) = \frac{1}{\sqrt{\tau_\emptyset \cdot (1 - \tau_\emptyset)}} \cdot (\tau_P - \tau_\emptyset) \cdot \sqrt{i_P} \cdot \sqrt{\frac{i_\emptyset}{i_\emptyset - i_P}}$$

The first factor is a normalization factor that is constant for all subgroups in a dataset. It is required for consistency with the statistical binomial test. The next two factors are equivalent to $q^{0.5}_{Kl}$ and score the deviation of the target concept in the subgroup and the subgroup size (coverage). The last factor modifies the generality term to generate symmetric values for a subgroup and its complement [141].

As a further example, the *chi-square measure* relates to the statistical significance according to the χ^2 statistical test. It is computed as [140]:

$$q^a_{\chi^2}(P) = \frac{i_P}{i_\emptyset - i_P} \cdot (\tau_P - \tau_\emptyset)^2$$

Visualizing interestingness measures In a fixed dataset, many interestingness measures (including the above measures) can be considered as functions $\hat{q}(i_P, \tau_P) \to \mathbb{R}$ that depend only on the size of the subgroup i_P and the target share in the subgroup τ_P. This can be plotted in a 3-dimensional space using the points $(i_P, \tau_P, \hat{q}(i_P, \tau_P))$. To visualize such a function in a 2-dimensional picture, one can draw the *isolines* (also called isometrics or contour lines), i.e., lines that connect points with identical values for the function $\hat{q}$, in a diagram with the axes i_P and τ_P, see [140]. Intervals between two contour lines can be colored with increasingly intense colors to indicate the order and the level of the function values. This allows to quickly inspect the behavior of an interestingness measure for diverse subgroup characteristics. Visualizations of previously presented interestingness measures using this method are shown in Figure 2.1.

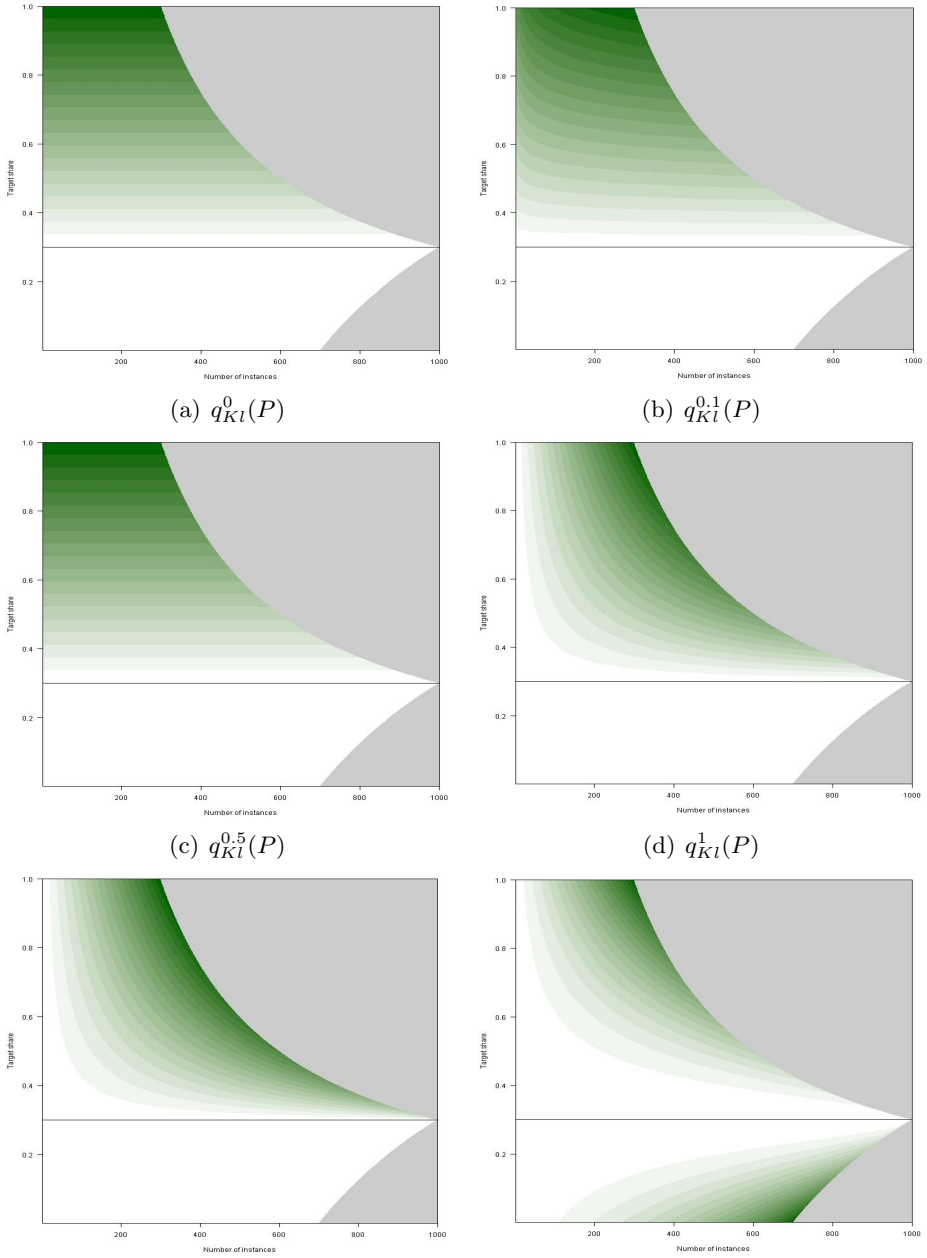
It displays the values of the interestingness measure in the (i_p, τ_P) space for an example dataset with $i_\emptyset = 1000$ instances and an overall target share of $\tau_\emptyset = 0.3$. Each point in the plot reflects the interestingness of a subgroup with the respective statistics. Darker shades of green indicate higher values of the interestingness measure. Color borders are isolines of the measure. For example, the isolines in the first plot are parallel to the x-axis since the interestingness measure $q^0_{Kl}(P)$ is independent from the number of instances covered by the subgroup. The second plot is for the measure $q^{0.1}_{Kl}(P)$. It shows that the size of subgroup influences the interestingness score only a little: The isolines, which connect subgroups with the same score, are almost, but not completely parallel to the axis for the target share. Grey areas refer to impossible subgroup statistics for the example dataset: As each subgroup in a dataset with 1000 instances and a target share of $\tau_\emptyset = 0.3$ contains at most 300 positive and at most 700 negative instances, e.g., a target share of 0.9 is impossible for a subgroup with 900 instances.

Since each discovered subgroup can be mapped to a certain point in the (i_P, τ_P) space, a variant of this diagram can also visualize the distributional characteristics of a set of discovered subgroups by adding pointers at the coordinate of the respective subgroup statistics to the plot.

Variations of this approach visualize the measures similarly, but utilize transformations of the x- and y-axis: The *coverage space* (PN-space) [85, 125] uses the number of positive and negative instance covered by the subgroup as axes. *Receiver operating characteristics (ROC plots)* normalize these to the intervals $[0, 1]$ [85, 77].

2.5.3.3 Interestingness Measures for Numeric Target Concepts

Although interestingness measure for numeric target concepts have been less intensively studied than interestingness measures for the binary case, several different approaches have been proposed for this task. However, overviews on that line of research are limited: Kloesgen included a listing of interestingness measures for numeric target concepts in



(a) $q^{0}_{Kl}(P)$ (b) $q^{0.1}_{Kl}(P)$

(c) $q^{0.5}_{Kl}(P)$ (d) $q^{1}_{Kl}(P)$

(e) $q_{bt}(P)$ (f) $q^{a}_{\chi^2}(P)$

Figure 2.1: Visualizations of exemplary interestingness measure in the (i_p, τ_P) space for $i_\emptyset = 1000$ instances and an overall target share of $\tau_\emptyset = 0.3$. Each point in the plot reflects the interestingness of a subgroup with the respective statistics. Darker shades of green indicate higher values of the interestingness measure, color borders are isolines of the measure. Grey areas refer to impossible subgroup statistics for the example dataset. For example, the isolines in the first plot are parallel to the x-axis since the interestingness measure $q^{0}_{Kl}(P)$ is independent from the number of instances covered by the subgroup.

his general overviews on subgroup discovery [140, 141]. A more recent discussion of several such measures was provided by Pieters et al. [210]. Substantially extending these works, the next section presents a concise and comprehensive survey on interestingness measures for numeric target concepts.

In contrast to the binary case, the distribution of the target concept in a subgroup and the overall dataset is more complex for numeric targets: The target values form multi-sets of real numbers. Many interestingness measures extract certain data characteristics, e.g., the mean or the median value, from these sets and compare the respective values obtained in the subgroup and in the overall dataset. We categorize interestingness measure for numeric target concepts with respect to the used data characteristics:

1. *Mean-based interestingness measures:* A simple approach to score subgroups in this setting is to compare the mean value in the subgroup μ_P with the mean value in the overall dataset $\mu_\emptyset$. A pattern is considered as interesting if the mean of the target values is (significantly) higher within the subgroup.

   In this direction, several interestingness measures have been proposed:

   a) *Generic mean evaluation functions:* A generic formalization for a variety of such measures can be constructed by adapting the interestingness measures q_a^{Kl} for binary targets presented above [140]: The target shares $\tau_P, \tau_\emptyset$ of subgroups and the general dataset are replaced by the respective mean values of the target variable in the subgroup μ_P and in the overall dataset $\mu_\emptyset$. This results in:

   $$q_{mean}^a(P) = i_P{}^a \cdot (\mu_P - \mu_\emptyset), a \in [0, 1]$$

   This includes the nominal version q_{Kl}^a as a special case: The numeric target value is set to $T(i) = 1$ if the boolean target concept for the instance c is *true*, and $T(i) = 0$ otherwise. Then, the mean values in the formula are equal to the respective target shares and the formula is identical to the Kloesgen measure q_{Kl}^a.

   This formalization is also reasonably easy to understand and adapt by (non-data mining) domain experts. As in the binary case, the specification of the parameter a can be determined in an iterative process: If the resulting subgroups tend to be too small, then a is increased. If the deviation of the mean target values in the resulting subgroups is too small, then a is decreased.

   This generic family of functions is either equal or order equivalent to several other interestingness measures proposed in literature, such as the average function, the mean test, the impact or the z-score:

   b) *Average function:* One of the most simple methods to score subgroups is to use the average value of the target variable $q_{avg}(P) = \mu_P$. This is order equivalent to $q_{mean}^0 = \mu_P - \mu_\emptyset$ since $\mu_\emptyset$ is constant for all subgroups in a dataset. This interestingness measure is very easy to understand. Since it

does not take the subgroup size into account, an additional constraint on the coverage of the subgroup is mandatory.

c) *Mean test:* The mean test function $q_{mt}(P) = q^{0.5}_{mean}(P) = \sqrt{i_P} \cdot (\mu_P - \mu_\emptyset)$ has been proposed by Kloesgen [140] and was used for example in [100].

d) *Impact:* The impact $q_{imp}(P) = q^1_{mean}(P) = i_P \cdot (\mu_P - \mu_\emptyset)$ is used by Webb [249] to favor larger subgroups.

e) *z-score:* The z-score interestingness measure [210] is defined as $q_z(P) = \frac{\sqrt{i_P} \cdot (\mu_P - \mu_\emptyset)}{\sigma_\emptyset}$, where $\sigma_\emptyset$ is the standard deviation in the overall dataset. The z-score is used in classical statistics to standardize variables. Since $\sigma_\emptyset$ is constant among all subgroups, this interestingness measure is order equivalent to the mean test $q^{0.5}_{mean}$.

f) *Generic symmetric mean evaluation functions:* The goal of the above interestingness measures is to identify subgroups that show increased values of the target concept. To find subgroups with decreased target values, one can just multiply all target values in the dataset by -1 in a preprocessing step. However, in many applications it is desired to discover subgroups with deviations in both directions at the same time. A simple method to achieve symmetry is to use the absolute value of the target share difference instead of the difference itself: $q^a_{abs}(P) = {i_P}^a \cdot |\mu_P - \mu_\emptyset|$. As an alternative, one could also use the squared difference instead: $q^a_{sq}(P) = {i_P}^a \cdot (\mu_P - \mu_\emptyset)^2$. These measures are order equivalent to $q^{\frac{a}{2}}_{abs}$.

g) *Variance reduction:* Another symmetric measure, which has been introduced in the context of regression tree learning, is the variance reduction [45], see also [140]. It explicitly takes the size of a subgroups complement into account:

$$q_{vr}(P) = \frac{i_P}{i_\emptyset - i_P} \cdot (\mu_P - \mu_\emptyset)^2$$

2. *Variance-based measures:* In descriptive statistics, the mean value denotes the first central moment of a distribution. The second central moment is the variance. It measures the *spread* of the distribution. Aumann and Lindell proposed to directly mine for patterns with a difference in the variance of the target concept [28].

   a) *Generic variance:* A simple method to trade-off between the change in variance and the subgroup size is to modify the Kloesgen measure by replacing the target shares $\tau_P, \tau_\emptyset$ with standard deviations $\sigma_P, \sigma_\emptyset$:

   $$q^a_{sd}(P) = {i_P}^a \cdot (\sigma_P - \sigma_\emptyset), a \in [0, 1],$$

   As before, this allows controlling the coverage of the results using the size parameter a. These measures do not directly correspond to a statistical significance test. To test the statistical significance of the deviation of variance, Aumann and Lindell propose to use an F-Test [28]. However, this test should

be applied carefully due to its strong sensitivity to the non-normality of the distribution [44].

Other measures combine the variance-based with mean-based components:

b) *t-score:* The t-score $q_t(P) = \frac{\sqrt{i_P} \cdot (\mu_P - \mu_\emptyset)}{\sigma_P}$ [210, 141] incorporates the mean μ_P and the standard deviation σ_P of the target values in a subgroup P. It reflects the significance of the deviation of target values in a subgroup using a Student's t-test. However, a direct statistical interpretation of the t-score should be avoided if the target concept is not normally distributed and the subgroup size is small, e.g., $i_P < 30$.

3. *Median-based measures:* Statistics based on the mean target value of subgroups are known to be sensitive to outliers. For example, a single instance with an extremely high, possibly erroneous target value shifts the mean of the entire subgroup significantly. A common method to avoid this issue is to use the median instead of the mean.

   a) *Generic median-based measure:* A generic family of median-based interestingness measures can again be derived by a small adaptation of q^a_{mean}: $q^a_{med}(P) = {i_P}^a \cdot (med_P - med_\emptyset)$, where med_P is the median of target values in the subgroup and $med_\emptyset$ the median in the total population. In general, there is no direct interpretation of these measures with respect to a statistical significance test. The measure can be modified to capture not increases of the median, but deviations in both directions by using the absolutes of the difference instead of the difference: $q^a_{sym/med}(P) = {i_P}^a \cdot |med_P - med_\emptyset|$

   b) *Median χ^2 Test:* As proposed in [210], the significance of a chi square test, which uses the median of the target attribute in the total population as a discretization cut-point, can be applied as an interestingness measure. From a computational point of view, this is accomplished by performing a discretization step during the pre-processing. Thus, in this work this measure will not be further considered in the context of numeric target concepts.

4. *(Full) Distribution-based measure (Kolmogorov-Smirnov measure):* A measurement that takes the complete distribution of the numeric target attribute in the subgroup and its complement into account was proposed for the discovery of *distribution rules* [184, 130]. These aim to identifying patterns, for which the subgroup and its complement are most significantly differently distributed according to a Kolmogorov-Smirnov significance test. The measure is order equivalent to the test statistic of this test:

$$q_{ks}(P) = \sqrt{\frac{i_P \cdot i_{\neg P}}{i_\emptyset}} \Delta_{(P,\neg P)},$$

where $\Delta_{(P,\neg P)}$ is the supremum of the differences in the empirical distribution function induced by the subgroup P and its complement $\neg P$. The empirical distri-

bution function is a function that computes for each value v in the target attributes domain the fraction of instances in the sample (pattern) with a target value smaller or equal to v. This measure can capture increases as well as decreases of the target values.

5. *Rank-based measures:* A variety of statistical tests for the deviation of numeric variables use the ranks of the target attribute over the target values themselves. That is, the instance with the highest target value is mapped to rank one, the instance with the second highest target value is mapped to rank two, and so on. This reduces the sensitivity to outliers compared to mean-based tests. Additionally, rank-based methods can also be applied to ordinal attributes.

   a) *Mann-Whitney measure:* This measure is based upon the statistical Mann-Whitney (also Wilcoxon-Mann-Whitney) rank sum test. Kloesgen proposed to transfer this test to interestingness measures in subgroup discovery [140]. A more detailed description of this measure has also been provided by Pieters et al. [210]. It compares the difference of the mean of ranks in the subgroup with the overall mean of ranks and computes its significance using a z-statistic. It is defined as:

$$q_{mw}(P) = i_P \cdot \frac{\frac{\mathcal{R}_P}{i_P} - \frac{i_\emptyset+1}{2}}{\sqrt{\frac{i_P i_{\neg P}(i_\emptyset+1)}{12}}} \sim \sqrt{\frac{i_P}{i_{\neg P}}} \cdot \left(\frac{\mathcal{R}_P}{i_P} - \frac{i_\emptyset+1}{2}\right) := q_{mw'}(P),$$

   where $\mathcal{R}_P$ is the sum of ranks within the subgroup P. Since the measure uses an approximation of the normal distribution (in the denominator in the formula), a direct statistical interpretation should be avoided for subgroups with only few covered instances.

   b) *AUC measure:* This quality function proposed in [210] computes the area under the ROC curve [77]. It can be computed by

$$q_{auc}(P) = \frac{\mathcal{R}_{\neg P} - \frac{i_{\neg P}\cdot(i_{\neg P}+1)}{2}}{i_P \cdot i_{\neg P}},$$

   with $\mathcal{R}_{\neg P}$ as the sum of ranks in the complement of the subgroup P. Since the measure in this form is normalized with respect to the subgroup size, it should be used only in connection with a constraint on the number of covered instances.

The probably most applied of these measures are the generic mean functions since they are adaptable in iterative approaches and easy to explain to humans with little mathematical expertise. However, choosing the right interestingness measure in the case of numeric target concepts is even more challenging than in the binary case, as more parameters have to be considered. That encourages an iterative and interactive process in cooperation with domain experts, as discussed in Section 2.7. Efficient discovery of

subgroups using these interestingness measures is one of the main contributions of this work, see Chapter 4.

2.5.3.4 Interestingness Measures for Complex Target Concepts

The next section outlines interestingness measures for complex target concepts, that is, multi-class target concepts and exceptional model mining targets.

Interestingness measures for multi-class target concepts For multi-class target concepts, a variety of interestingness measures has been proposed in literature. A common method to derive multi-class interestingness measures is, to adapt measures for the binary target setting. Binary targets split the instances of the dataset into two parts: those with a true and those with a false target concept. Since an increase of the occurrence of one class implies a decrease of occurrence of the other, a deviation of the distribution is fully described by the share of one these classes, i.e., the share of instances with a true target concept. For multi-class targets, the overall dataset is split into n parts instead, therefore a deviation in the occurrence of the target concept cannot be captured by the probability of just one class.

Thus, the difference $\tau_P - \tau_\emptyset$ of the target concept shares between the pattern P and the overall dataset is replaced by the sum of squared differences over all classes in the target concept: $\sum_{r \in dom(T)} (\tau_P^r - \tau_\emptyset^r)^2$. Here, τ_P^r denotes the probability that the target concept takes the value r. For example, a multi-class counterpart for the weighted relative accuracy measure $q_{Kl}^1(P) = i_p \cdot (\tau_P - \tau_\emptyset)$ is given by $q_{multi}^1(P) = i_p \cdot \sum_i (\tau_P^i - \tau_\emptyset^i)^2$, see [140]. Abudawood and Flach discuss different variations of this multi-class interestingness measure in [3].

Other measures for this task share are similarly composed. For example, the multi-class variant of the chi-square measure is given by:

$$q_{\chi^2-mc}^a(P) = \frac{i_P}{i_\emptyset - i_P} \cdot \sum_{r \in dom(T)} \frac{(\tau_P^r - \tau_\emptyset^r)^2}{\tau_\emptyset^r} ([140], \text{see also } [196]).$$

A listing of interestingness measures for this setting has been compiled in [140].

Interestingness measures for exceptional model mining Defining interestingness measures for exceptional model mining strongly depends on the chosen model class. Several different model classes have been proposed for exceptional model mining in literature, and for each of these model classes different interestingness measures can be used. Due to this large number of possibilities, this work only demonstrates construction principles of such interestingness measures using a few selected examples.

In the initial work of this area, Leman et al. presented a variety of model classes and discussed appropriate interestingness measures [170]. These model classes include:

- the correlation model. It uses the well-known correlation coefficient to measure the interdependency between the two numeric model attributes.

- the linear regression model. It computes the slope of the linear regression line for two numeric model attributes.

- different classification models. For these, one of the model attributes is chosen as a class attribute. Then, classification models of a predefined type (e.g., a logistic regression classifier or a decision table classifier) are built from the other model attributes, which predict the value of the class attribute.

As in classic subgroup discovery, some interestingness measure trade-off the generality of the subgroup versus the exceptionality of the model derived from the subgroup. For example, the entropy measure q_{entr} for the correlation model is given by [170]:

$$q_{entr}(P) = ent(P) \cdot |cor(P) - cor(\neg P)|.$$

Here, the generality of the subgroup is described by the entropy of the split of the overall dataset in the subgroup and its complement: $ent(P) = -\frac{i_P}{i_\emptyset} \cdot \log \frac{i_P}{i_\emptyset} - \frac{i_{\neg P}}{i_\emptyset} \cdot \log \frac{i_{\neg P}}{i_\emptyset}$. The exceptionality of the model is computed as the difference between the Pearson correlation coefficient of the two model attributes in the subgroup $cor(P)$ and this statistic in the subgroup's complement $cor(\neg P)$. Other proposed measures focus on the statistical significance of the deviation, which leads to more complex formalizations. For more details, we refer to the original work [170].

Grosskreutz et al. use a simple and intuitive interestingness measure for a set of numeric model attributes [102]: $q_{dst} = \sqrt{\frac{i_P}{i_\emptyset}} \cdot ||\mathbf{m}_P - \mathbf{m}_\emptyset||_1$. $\mathbf{m}_P$ is the vector of mean values of the model attributes for the pattern P, $\mathbf{m}_\emptyset$ for the overall dataset respectively. $|| \cdot ||_1$ denotes the well-known L_1 norm for vectors.

Duivesteijn et al. propose bayesian networks over the model attributes as another model class for exceptional model mining. As an interestingness measure in this setting, they propose $q_{bn} = ed(P) \cdot \sqrt{ent(P)}$. $ed(P)$ describes the edit distance between the network structure derived from the subgroup and the network structure derived from the overall dataset. As before, the entropy $ent(P)$ of the split induced by the subgroup is used to measure the generality [73, 71].

Beside these model classes, specialized interestingness measures have been proposed for specific applications: For example, for the task of descriptive community mining, which can be regarded as some kind of exceptional model mining, Atzmueller and Mitzlaff use the interestingness measures *conductance* and *modularity* [19]. These are transfered from the research area of social network analysis, see [178, 199]. In another scenario in the medical domain described by Mueller et al., a specialized *prediction quality* is used as interestingness measure. This reflects how good a screening test can predict the actual diagnosis. [198]

Overall, exceptional model mining is applied with a variety of different model classes, and for each model class many different interestingness measures can be defined. However, a common form of interestingness measure is: $g(P) \cdot \delta(P, \mathcal{I})$. Here, $g(P)$ expresses the generality of the subgroup P and $\delta_M(P, \mathcal{I})$ measures the distance between a model built in the subgroup P and the model built from the full dataset $\mathcal{I}$ (or alternatively,

the subgroup's complement). This formalization covers many of the above approaches including q_{entr}, q_{bn}, or q_{dst}.

Some important model classes are introduced more formally in Chapter 5 of this work, which presents a novel generic data structure for the efficient mining of exceptional models.

2.5.4 Generalization-Awareness

The interestingness measures presented so far are only based on the statistics, e.g., the number of positives and negatives, in the subgroup itself and the overall dataset. These *traditional measures* are independent from the subgroup description. This can lead to uninteresting discoveries:

Assume that the target share in the total population is at $\tau_\emptyset = 30\%$. The subgroup with the description $age > 50$ has an increased target share of $\tau_{age>50} = 50\%$. When any other selector, which does not influence the target concept, is added to this description, then the target share of the resulting subgroup should not change. For example, the pattern $gender = male \wedge age > 50$ is also expected to have a target share around 50%, given that *gender* does not influence the occurrence of the target concept. However, even if the target share for the subgroup $gender = male \wedge age > 50$ decreases slightly in comparison to its generalization $age > 50$ to $\tau_{age>50} = 49.8\%$, this is still considerably higher than the target share in the overall dataset. Therefore this pattern might get a high score by a traditional interestingness measure, although its target share is probably not interesting at all given the information on its generalization.

To avoid that such subgroups are included in the result set, Bayardo et al. introduced a *minimum improvement constraint* [35]: This filter removes a subgroup from the result if it has a lower target share than any of its generalizations. Such subgroups have also been described as *unproductive* [250, 253]. A similar approach has been proposed for mean-based rules for numeric properties of interest: Here a rule is "undesired", if its mean value does not deviate significantly from the mean value of its generalizations [28].

This strict filtering approach has a significant downside: Continuing our example above, consider another subgroup $age > 50 \wedge height > 180$ that has a target share of $\tau_{age>50 \wedge height>180} = 50.2\%$. Although the influence of *height* on the target concept is only marginal and could be explained by noise in the data, this subgroup is not removed by a minimum improvement filter, since target share is slightly increased in comparison to $age > 50$. Since the relation to its generalization is only considered by an additional filter criterion, $age > 50 \wedge height > 180$ is possibly also high ranked in the result set in the ordering implied by a traditional interestingness measure.

To avoid this, recent approaches incorporate the target shares of generalizations directly into an interestingness measure. A simple method to accomplish this is to replace the comparison to the target share in the total population with a comparison to the maximum target share in any generalization of the assessed subgroup pattern. In this

direction, Batal and Hausknecht adapted the mean test interestingness measure $q_{Kl}^{0.5}$ to the form:

$$r_{Kl}^{0.5}(P) = \sqrt{i_P} \cdot (\tau_P - \max_{H \subset P} \tau_H), \text{ [29]}$$

where $\max_{H \subset P} \tau_H$ denotes the maximum target share in all generalizations of P. A similar approach was used by Grosskreutz et al. for a numeric target concept. In a case study on election analysis [102], they applied the following interestingness measure:

$$r_{mean}^{0.5}(P) = \sqrt{i_P} \cdot (\mu_P - \max_{H \subset P} \mu_H).$$

Here, $\max_{H \subset P} \mu_H$ is the maximum of mean values of the target concept in in all generalizations of P.

These adaptations can be applied to interestingness measure q_{Kl}^a or q_{mean}^a with arbitrary parameters a. Thus we obtain the family of *relative interestingness measures* r_{Kl}^a and r_{mean}^a:

$$r_{Kl}^a(P) = {i_P}^a \cdot (\tau_P - \max_{H \subset P} \tau_H), a \in [0, 1]$$
$$r_{mean}^a(P) = {i_P}^a \cdot (\mu_P - \max_{H \subset P} \mu_H), a \in [0, 1]$$

Efficient algorithms for subgroup discovery for such interestingness measures are essential for practical applications. As a contribution of this work, we analyze in Chapter 6 how specific properties of such measures can be exploited for advanced pruning in fast algorithms. As another contribution, Chapter 7 presents a novel approach that models knowledge about generalizations into bayesian network fragments in order to construct enhanced interestingness measures and thus to obtain more interesting results. One advantage of generalization-aware interestingness measure is also that some redundant results are avoided. For that purpose, also other approaches have been proposed.

2.5.5 Avoiding Redundancy

The above criteria select subgroups independent of other subgroups in the result set. In many cases, this causes the result set to contain similar, strongly overlapping subgroups, that is, subgroups that cover (almost) the same instances. Since those subgroups are potentially caused by a single phenomenon, presenting multiple overlapping subgroups may provide only little additional information to the user. As the number of subgroups in the result set is limited in a top-k-approach, other more interesting subgroups are suppressed and not presented to the user. In order to avoid such redundancies in the results of subgroup discovery tasks, different solutions have been proposed. These are discussed in the following sections.

2.5.5.1 Covering Approaches

The *sequential covering* method [192, 203] discovers subgroup patterns one-by-one. In each iteration, the best subgroup for the current dataset is identified and added to the result set. Afterwards, all instances, which are covered by this subgroup are removed from the overall dataset. Thus, the selection of further subgroups is only based on the remaining instances. Sequential covering has been applied with great success in the related field of classification rule learning [203, 89, 64], but in subgroup discovery often only the first few discovered patterns are considered as interesting. This is, because the following patterns are induced by only a small and possibly biased subset of instances [134, 167].

To enhance the covering approach for subgroup discovery, *weighted covering* was proposed [91, 167]. In this approach, subgroup patterns are also discovered iteratively. However, in contrast to sequential covering the instances covered by the currently best subgroup are not removed from the dataset. Instead, the mining algorithm stores for each instance the number of subgroups in the result set, which cover it. For each such subgroup the weight of an instance is reduced: When counting positive and negative instances for a subgroup or the total dataset in order to evaluate subgroup, then for each instance the count is not incremented by one, but by the weight of this instance. In doing so, positive instances with higher weights are more likely to be covered by subgroups in the next iteration than instances with a lower weight. Weighted covering is used either directly in the subgroup discovery algorithm [167, 134], or as a post-processing step to select a subset of previously discovered subgroups [91], see also [14].

2.5.5.2 Filtering Irrelevant Subgroups

Another method to avoid redundant output for subgroup discovery with binary target concepts was proposed by Gamberger and Lavrač [91, 166]. It performs a check for *relevance* by a pairwise comparison of subgroups. A subgroup P_{rel} is considered as *more relevant* than another subgroup P_{irrel}, if all positive instances covered by P_{irrel} are also covered by P_{rel} and all negative instances of P_{rel} are also covered by P_{irrel} [166]:

$$tp(P_{irrel}) \subseteq tp(P_{rel}) \wedge fp(P_{rel}) \subseteq fp(P_{irrel})$$

This definitions considers the actual sets of instances, not the instance counts. A subgroup is *irrelevant* if there exists any subgroup, which is more relevant. It is called *relevant* if no such subgroup exists. According to Gamberger and Lavrač [91], irrelevant subgroups should be filtered from the results. For most interestingness measures, e.g., all Kloesgen measures q^a_{Kl} or the binomial test measure, a more relevant subgroup also has a higher interestingness score. However, without additional filtering it is common that the result set includes irrelevant subgroups as well as the respective more relevant subgroups.

The notion of irrelevant subgroups is closely related to the concept of *closed patterns* in frequent itemset mining. In particular, it has been shown that relevant subgroups are always closed on the positive instances [95, 96, 104].

While the concept of irrelevance provides good results in many scenarios, it is very strict in the sense that in only applies to complete coverage. If only a single instance contradicts the irrelevancy conditions (e.g., due to noise in the data), then the subgroup will not be filtered. To address this problem, we proposed in a previous work the concept of *ϵ-relevancy* [171]. This generalization defines a subgroup P_{rel} as more irrelevant than a subgroup P_{rel}, if there exist sets of instances E_{tp}, E_{fp}, such that:

$$(tp(P_{irrel}) \subseteq tp(P_{rel}) \cup E_{tp}) \wedge (fp(P_{rel}) \subseteq fp(P_{irrel}) \cup E_{tp}) \wedge (|E_{tp}| + |E_{fp}| \leq \epsilon).$$

That is, there are only ϵ exceptions to the standard conditions of relevancy. Consequently, this definition includes the standard definition of irrelevant rules as a special case for $\epsilon = 0$. Using parameters $\epsilon > 0$ filters more subgroups, which are overlapping with other subgroups in the result set. This leads to more diverse subgroups in the result. This concept has later been extended and studied in-depth under the name of Δ-dominance [105].

2.5.5.3 Subgroup Set Mining

Another way to obtain a diverse set of subgroups is that not single subgroup patterns are scored and selected on their own, but *sets of subgroups*. The one pattern set with the highest score is then returned to the user. Scores for pattern sets combine components that measure the aggregated individual interestingness of the contained subgroups with components that measure the *diversity* of the subgroup patterns [271]. Measures for diversity of patterns assess the differences in the subgroup description and the subgroup coverage. In the case of exceptional model mining also the differences of models can be evaluated [243]. Measures for the redundancy of coverage can be transferred from similar methods in unsupervised settings, see [150, 151, 213]. For example, the *joint entropy* $H(\mathcal{P})$ of a set of subgroups $\mathcal{P} = \{P_1, \ldots, P_k\}$ is defined as [150, 243]:

$$H(\mathcal{P}) = - \sum_{B \in \{true,false\}^k} p(P_1 = b_1, \ldots, P_k = b_k) \cdot \log_2 p(P_1 = b_1, \ldots, P_k = b_k).$$

In this definition $B = (b_1, \ldots, b_k)$ is a tuple of boolean values and the sum iterates over all possible tuples. Each boolean value in the tuple corresponds to one subgroup in the subgroups set, for which the diversity is determined. $p(P_1 = b_1, \ldots, P_k = b_k)$ denotes the fraction of instances in the dataset, which are covered by exactly those patterns, for which the corresponding boolean value in B is true.

Unfortunately, the search space for mining sets of subgroups is substantially larger than it is for mining individual subgroups, which leads to severe computational issues. As a consequence, selecting high scoring, diverse pattern sets has mostly been applied as a post-processing step on a large result set of a traditional mining task, e.g., in [151]. Recently, an approach by van Leeuwen and Ukkonen focuses on the efficient exact mining of the best subgroup sets [244] in a novel approach that exploits entropy-based pruning of the search space. Another method that directly incorporates the subgroup set selection into beam-search was proposed by van Leeuwen and Knobbe [242, 243].

2.5.5.4 Clustering of Results

For a more comprehensive overview on subgroup discovery results, it was proposed to cluster the results with respect to their coverage [23]. In doing so, sets of subgroups, which cover similar instances, i.e., have a large overlap, are presented together to the user. The similarity of two subgroup patterns P_1, P_2 is for example measured by the *Jaccard coefficient* of the covered instances, that is, the fraction between the intersection and the union of these instances:

$$sim(P_1, P_2) = \frac{|sg(P_1) \cap sg(P_2)|}{|sg(P_1) \cup sg(P_2)|}$$

Based on this similarity measure, a hierarchical clustering (see for example [226]) is applied. Thus, potentially redundant subgroups, which have a high overlap, are summarized in one cluster. This allows experts to analyze similar subgroups simultaneously.

2.6 Background Knowledge

The integration of background knowledge (prior knowledge) is accepted as an important component for successful data mining in real world applications. On one hand this knowledge describes the application domain, on the other hand it captures the current beliefs and preferences of the user. Domain knowledge is either objective or widely accepted, such as normality ranges for numerical measurements in medical domains. It can be formalized in domain *ontologies*, e.g., in RDF or OWL formats. For many applications, these are also publicly available in the *semantic web* [10, 168]. In contrast to domain knowledge, user preferences are mostly subjective. Liu and Hsu distinguish between *direct user preferences*, which specify what class of results the user wants to see, and *indirect user preferences*, which describe previous knowledge of the user in the domain [180].

Background knowledge covers diverse types of information. This includes, but is not limited to (see also [21]):

- Taxonomies of attributes and taxonomies of attribute values. Both can be represented in a tree-like structure [11, 142].
- Default values and normality/abnormality for attributes.
- Similarity of attribute values.
- Ordering information for ordinal attributes.
- Discretization bounds for numeric attributes, see Section 2.3.2.
- Derived attributes, that is, additional attributes that are constructed from the original attributes of the dataset.

- Importance of attributes. This can be described by a partition of the attributes into "priority groups" [21] or by assigning weights to the attributes. Attributes, which are irrelevant for the current task, are excluded from the search.
- Already known correlations between attributes, especially correlation with the target attribute. This is often acquired and stored in the form of known (subgroup) patterns, but may use also more complex representations such as Bayesian networks, cf. [126, 127].
- Confounding variables, see for example [207, 191]. These are influence factors, which correlate with the target variable and another subgroup and therefore can lead to the (false) impression that there is a causal connection between this subgroup and the target concept. Knowledge of confounding variables can be incorporated into subgroup discovery, e.g., by applying a *stratified analysis* [26].

The acquisition of background knowledge is often difficult since it can involve manual elicitation of relevant information. This is a time-consuming process, as it suffers from the *knowledge acquisition bottleneck* [114, 113, 90], a well known problem in the research area of expert systems. For this task, a tailored environment for acquiring the knowledge is essential, see also Chapter 8. To remedy this problem, for some specific knowledge types, e.g., the similarity of attribute values or abnormality information, (semi-)automatic methods for learning background knowledge have been proposed [31].

In subgroup discovery, background knowledge is exploited in different stages of the mining process: In the data preparation phase, background knowledge helps to generate the search space appropriately. For example, discretization bounds can be derived from normality/abnormality information of attribute values, selectors that correspond to default values of attributes can be removed or similar attribute values can be summarized in a single selector, see [25]. During the search algorithm, interestingness measures can be modified with respect to the importance of attributes and known attribute correlations, e.g., by increasing the score of subgroups that contain important attributes. Additionally, algorithms exploit background knowledge to speed up the search, e.g., by preventing uninteresting attribute combinations to be explored. In post-processing, subgroups can be filtered with respect to background knowledge in order to avoid the rediscovery of already known patterns. Another approach uses background knowledge of a domain ontology to provide explanations for discovered subgroups with strictly defined semantics [245].

2.7 The Interactive Subgroup Discovery Process

Since knowledge discovery in databases is a non-trivial task, several process models have been presented to help practitioners. The most wide-spread (see [1]) methodology is *CRISP-DM* (Cross Industry Standard Process for Data Mining) [223]. The process employs the following 6 major phases: 1. business understanding, 2. data understanding, 3. data preparation, 4. modeling, 5. evaluation, and 6. deployment. Key data mining algorithms are only applied in the central modeling phase. Applying this model requires

moving back and forth between the phases in order to adapt according to obtained insights. The process model is visualized in Figure 2.2.

Figure 2.2: The CRISP-DM process model [223] (colored figure from [129]).

The interestingness of subgroups depends partially on subjective criteria and background knowledge that is potentially not fully formalized. Therefore, subgroup discovery in particular should not be regarded as an isolated, automatic mining task, but as one step in an interactive process, which is heavily influenced by the end-user. In that direction, Gamberger and Lavrač proposed a process model for *expert guided subgroup discovery* [91]. It consists of the following 8 steps: "1. problem understanding, 2. data understanding and preparation, 3. subgroup detection, 4. subgroup subset selection, 5. statistical characterization of subgroups, 6. subgroup visualization, 7. subgroup interpretation, and 8. subgroup evaluation." [91]. The process is *iterative* since some steps might be performed multiple times for optimizing the solution. It is also *interactive*, as the human expert (end-user) is closely incorporated in the process. The methodology is in line with the *active mining* approach [197, 237, 92] that focuses on close involvement of human experts in data mining.

Atzmueller et al. emphasize the knowledge-intensive nature of subgroup discovery that includes the incorporation of background knowledge and the analysis of subgroups. In that direction, they developed a process model for knowledge-intensive active subgroup mining that combines interactive and automatic components [25, 14] in an iterative approach. It involves three main steps: *discovery*, *inspection* and *refinement*. The steps are iterated in a spiral model: By inspection and analysis of subgroup discovery results additional background knowledge is obtained, which is used for improved subgroup discovery in the next iteration.

These interactive approaches require suitable presentation techniques as well as advanced tool support. Contributions of this work in that direction are presented in Chapter 8.

2.8 Complexity of the Subgroup Discovery Task

The computational complexity of a subgroup discovery task results from the huge size of the search space, which is caused by the *pattern explosion* problem. This describes that even for a relatively small number of selectors a huge number of conjunctive patterns can be generated. In general, each subset of the set of selectors forms a subgroup description, which is to be considered for mining. Therefore, the size of the search space $|\Sigma|$ for a set of selectors $\mathcal{S}$ is given by:

$$|\Sigma| = |\{P|P \in 2^{\mathcal{S}}\}| = 2^{|\mathcal{S}|}$$

In many scenarios, each attribute appears only in a single selector of each subgroup description. This is especially the case if only value-identity checks are used as selectors, see Section 2.3.1. For example, if a subgroup description contains the selector *color = red*, then this description should not also contain another selector *color = blue*, since the described set of instances will always be empty. Formally, the selectors are partitioned by their attributes into disjunct sets, which are mutually exclusive in subgroup descriptions. Thus, a significantly smaller number of candidate subgroups has to be evaluated: Consider $|\mathcal{A}|$ attributes with m selectors each. If these m selectors are mutually exclusive, i.e., each subgroup description contains at most one of these m selectors for each attribute, then the number subgroup descriptions, which have to be evaluated, decreases from of $2^{m \cdot |\mathcal{A}|}$ to (see [14]):

$$\begin{aligned} |\Sigma_*| &= |\{P \in \Sigma \,|\, \text{P contains at most one selector per attribute}\}| \\ &= \sum_{i=0}^{|\mathcal{A}|} m^i \binom{|\mathcal{A}|}{i} = (1+m)^{|\mathcal{A}|} \end{aligned}$$

This is substantially smaller than in the general case since the exponent is only the number of attributes $|\mathcal{A}|$ and not the number of selectors in the search space $|\mathcal{S}| = |\mathcal{A}| \cdot m$. Consider for example the case $m = 3$. That means that for each of the $|\mathcal{A}|$ attributes there are 3 selectors. Then, with the assumption that these 3 selectors are mutually exclusive the search space consists of $4^{|\mathcal{A}|}$ candidate pattern. Without this assumption, $2^{3 \cdot |\mathcal{A}|}$ are to be considered. Thus, the assumption, that only one selector for each attribute can be part of a subgroup description, reduces the size of the search space by a factor of $\frac{2^{3 \cdot |\mathcal{A}|}}{4^{|\mathcal{A}|}} = 2^{|\mathcal{A}|}$, see Figure 2.3.

This reduction exploits the exclusivity of the selectors for one attribute and thus can not be applied in all settings, i.e., not when using negated selectors, see Section 2.3.1.

As discussed before, one of the most wide-spread constraints limits the maximum number of selectors contained in a subgroup description. This reduces the size of the search space by orders of magnitude: Let d be the maximum number of selectors in a

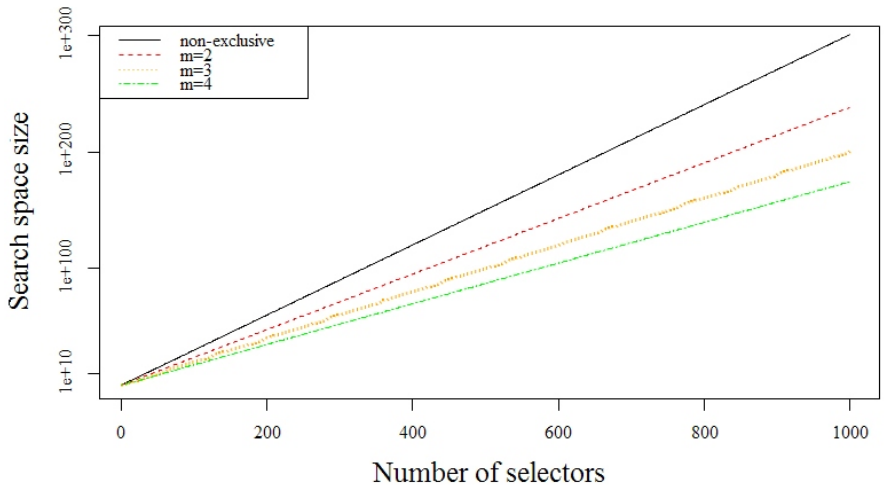


Figure 2.3: Comparison of the (unpruned) search space size for non-exclusive selectors and for disjunct sets of $m = 2, 3, 4$ mutually exclusive selectors per attribute.

subgroup description. Then, the size of the respective search space $|\Sigma^{1..d}|$ is given by:

$$\begin{aligned} |\Sigma^{1..d}| &= |\{P \in \Sigma \,|\, |P| \leq d\}| \\ &= \sum_{i=0}^{d} \binom{|\mathcal{S}|}{i}, \end{aligned}$$

which is in $O(|\mathcal{S}|^d)$ for small d ($d \ll |\mathcal{S}|$). Thus, the use of a description length constraint reduces the search task from a exponential to a polynomial problem.

These considerations describe the maximum number of subgroup descriptions that are contained theoretically in the search space. Practical algorithm compute optimum solutions to subgroup discovery tasks with the evaluation of far less subgroups by utilizing pruning of the search space, see Section 3.4.

2.9 Statistical Significance of Results

In some applications, subgroup discovery is considered as an exploratory method, which is used to generate hypotheses that can be confirmed in subsequent studies, cf. [261]. However, in other areas the need of statistically valid findings is emphasized. For this reason, many interestingness measures are based on tests from confirmatory statistics, e.g., the chi-square or the t-score interestingness measures.

However, the statistical significance of subgroup discovery results is influenced by the so-called *multiple comparison problem* [122, 119, 37, 128]: Since a very large amount of candidate patterns is assessed, some subgroups will pass a significance test with traditional confidence levels by pure chance. Consider for example a search space for sub-

group discovery of 10^6 candidate patterns and a statistical test with significance level of $\alpha = 0.05$, i.e, there is a 5% risk that a false finding is considered as valid. Then one must expect that up to $10^6 \cdot 0.05 = 5 \cdot 10^4$ subgroups pass the statistical test even if no subgroup in the search space is correlated with the target concept. Such findings are also called *false discoveries*. In order to control the risk of false discoveries, a variety of different methods have been proposed:

The *Bonferroni correction* (also Bonferroni adjustment) divides the required confidence level α by the number of tested hypotheses [74]. In the case of subgroup discovery, the number of hypotheses is equal to the number of subgroup patterns considered in the search space Σ. By doing so, the subgroup P is only considered as statistically significant, if $\rho(P) \leq \frac{\alpha}{|\Sigma|}$, where $\rho(P)$ is the p-value for the subgroup P according to the applied statistical test and α is the required overall significance level. The Bonferroni adjustment is considered a very conservative method: A subgroup has a very low probability of incorrectly passing the test, but the test might filter out many valid discoveries.

The *sequentially rejective test of Holm* (also called Holm-Bonferroni adjustment) [122] is a more powerful variation of this technique. It orders the discovered subgroups according to the p-value of the significance test and then applies iteratively decrementing correction divisors. However, the improvement is only moderate in the subgroup discovery setting with huge search spaces and comparatively few significantly deviating subgroups. It is also difficult to integrate in a search algorithms since the ordering of subgroup patterns according to their significance is not available during the search [250].

For subgroup discovery (respectively contrast set mining, see Section 2.10), specialized adaptations have been proposed, which have been described as *layered critical values* [33, 252]. These utilize that the basis for the Bonferroni correction, the Bonferroni inequality, does not require that each of the multiple statistical tests, which are applied for each subgroup in the search space, uses the same significance level. It only postulates that the sum of the individual significance levels is smaller than the overall required significance level α. Advanced approaches consequently do not divide the allowed error α equally among all candidate subgroups as in the classical Bonferroni adjustment. Instead, different significance thresholds are applied for different subgroups, depending on the complexity of their subgroup description. Bay and Pazzani propose the following cutoff α-values for the individual statistical tests [33]:

$$\alpha_l = \min\left(\frac{\alpha}{2^l \cdot |\Sigma^l|}, \alpha_{l-1}\right),$$

where α_l is the significance threshold for a subgroup described by l selectors, and Σ^l is the overall number of candidate subgroups in the search space with a description of l selectors. Since there are substantially less subgroups with short subgroup descriptions, the factor $|\Sigma^l|$ is smaller and the requirements for those subgroup patterns are less strict than for subgroups with longer descriptions. A variation of this approach with an improved scheme to distribute the error among the subgroups was proposed in [252].

A simple and robust alternative method to achieve statistically significant results

is *holdout evaluation*. This approach resembles the cross-fold methodology applied in machine learning: the dataset is divided into two subsets, an *exploratory* subset and a *holdout set* [250]. The exploratory data is used to perform traditional subgroup discovery in order to identify promising candidate subgroups. These findings are then confirmed by statistical tests on the data of the holdout set. If more than one candidate pattern is verified in the holdout data, still a Holm-Bonferroni adjustment has to be applied for the tests. However, since the tests are only applied on few subgroups (e.g., the top k) instead of the complete search space, the correction factor is by orders of magnitudes smaller. An empirical comparison of the holdout evaluation with the Bonferroni adjustment is included in [250].

Another method to control false discoveries was proposed by Duivesteijn and Knobbe [72]: Several baseline subsets $I_1, \ldots, I_m$ of instances are sampled from the overall instance set $\mathcal{I}$. For these subsets the values of the target concept are shuffled using a simplified variation of swap randomization [98, 98]. Then, for each of the subsets I_j the best subgroup P_j^* according to the chosen interestingness measure $q(P)$ is identified. According to the central limit theorem, the qualities $q(P_j^*)$ follow approximately a normal distribution for a sufficiently large number of random subsets. For this *distribution of false discoveries* the mean value and the standard deviation is determined. Using these parameters, for each result subgroup, which has been found by subgroup discovery in the original dataset, a p-value can be computed. It indicates, with which probability the pattern is generated from the same model as the false discoveries. This can validate the statistical significance of the discoveries.

2.10 Subgroup Discovery and Other Data Mining Tasks

The next section discusses the relations between subgroup discovery and other data mining tasks. It starts by summarizing techniques, which are closely related to subgroup discovery. These have been summarized with the term *supervised descriptive rule induction*. Then, differences and common grounds of subgroup discovery and the most popular data mining methods are outlined: classification, association rule mining and clustering.

Subgroup discovery is a *supervised* and *descriptive* data mining task. Supervised means that the task is focused on a specific property of interest. Descriptive expresses that it aims at unveiling dependencies with the target concept in a form that is directly understandable for humans and does not emphasize predictions of future events. The discussed data mining tasks can (simplified) be distinguished by these characteristics. Subgroup discovery is a supervised, descriptive technique. Classification is also supervised, but *predictive* instead of descriptive, that is, it focuses on correct predictions with limited attention to model interpretability. Association rule mining aims at the discovery of descriptive patterns, but is unsupervised. Clustering is an unsupervised mining method that does not necessarily result in descriptive, interpretable models. The author of this work is aware that this characterization is an oversimplification, and there are a multitude of approaches to cross the outlined borders.

2.10.1 Other Techniques for Supervised Descriptive Pattern Mining

Aside from subgroup discovery, also other pattern mining techniques have been proposed, which are also supervised and descriptive. These techniques differ in terminology and set different emphases on the task, but pursue a similar overall goal, cf. [158]. Nonetheless, they were explored in their own lines of research.

Contrast set mining [32, 33, 255] aims at finding "all contrast sets whose support differs meaningfully across groups" [33]. In this context, contrast sets are conjunctions of attribute-value pairs, and groups are predefined disjunct sets of instances. While (binary) subgroup discovery identifies descriptions of instances, for which the probability of the target concept is unusual, contrast set mining is concerned with finding descriptions of instances, which occur with different probabilities in the groups, see [88].

Mining emerging patterns [70] is another closely related task. Emerging patterns have been defined as "itemsets whose supports increase significantly from one dataset to another" [70]. Thus, they "can capture merging trends in time-stamped databases, or useful contrasts between data classes." [70]. In the terminology of subgroup discovery, the part of the data with a *true* binary target concept is considered as one dataset for emerging pattern mining, the rest of the instances as the other dataset. Subgroup descriptions with a high target share are then equivalent to the descriptions, which occur more often in the first dataset.

Kralj Novak et al. have recently shown that the tasks of subgroup discovery, contrast set mining and mining emerging patterns are essentially equivalent. They propose to summarize these fields in a framework called *supervised descriptive rule induction* or *supervised descriptive rule discovery* [158]. Consider for example the task of identifying influence factors for a successful surgery: In subgroup discovery a target concept $surgery\_result = successful$ is chosen. For contrast set mining, the dataset is divided into two groups: $surgery\_result = successful$ and $surgery\_result = \neg successful$. In emerging pattern mining, the data is also divided into two "datasets" characterized by the same conditions. An interesting pattern for all three tasks could be given by a conjunction of attribute-value pairs such as $gender = male \wedge blood\_pressure = high$. Depending on the task, such a pattern is called subgroup description, contrast set or itemset.

Exception rules are "typically represented as deviational patterns to common sense rules of high generality and accuracy" [233]. It is distinguished between *directed* exception rule mining, in which common sense rules are provided to the mining algorithm, e.g., by background knowledge, and *undirected* exception rule mining, which discovers common sense rules as well as deviations from these rules [231]. Exception rule mining has been regarded as a part of the larger field of *exception discovery* that also includes e.g., outlier detection [232]. A summary for the research on exception rule mining has been provided in [66].

Less frequently used names for problem statements, which are closely related to subgroup discovery, are *bump hunting* [82] and *change mining* [181, 247], see also [158, 88].

2.10.2 Classification

Like subgroup discovery, classification is a *supervised* data mining task, i.e., it is especially concerned with a specific property of interest. However, it is a predictive task, i.e., it focuses exclusively on forecasting the class (target concept) of instances. Thus, using large or complex models, which are difficult to comprehend for humans, is viable in classification, but not for descriptive tasks such as subgroup discovery.

Regarding classification algorithms, the approach that is most closely related to subgroup discovery is rule-based classification [192, 64], see [86] for a recent overview. These methods try to identify a set of rules, which are in the best case "complete and consistent" [87]. That means that all instances should be covered by at least one rule and each rule should make a correct prediction of the target concept. In this context, each subgroup description P implies a rule $P \rightarrow T$ that has the subgroup description as the condition of the rule and the target concept (class) T as its consequent. Despite the obvious similarities between classification rule learning and subgroup discovery the different overall goal leads to different considerations for the selection of patterns: For example, if the target share of a binary target concept is at 38% in the overall dataset and at 39% in a specific subgroup, then this increase is hard to exploit in predictive algorithms. This pattern may nonetheless be regarded as interesting in subgroup discovery since it may hint at previously unknown influence factors, cf. [261]. As another example, using a large amount of low-support patterns can increase the predictive performance of rule-based classification algorithms. In contrast, reporting a large amount of such results is discouraged for subgroup discovery because subgroup patterns have to be inspected one-by-one by human experts.

Although classification pursues a different overall goal than subgroup discovery in utilizing discovered patterns, the algorithmic challenges for efficient mining of classification rules are similar. This is especially true for classification algorithms that utilize association rule mining for classification such as *CBA* (classification based on associations) [182] or *CorClass* (correlated association rule mining for classification) [273]. Therefore, relevant methods for mining supervised patterns, which have been proposed in the context of classification, are included in the overview on algorithms in Chapter 3.

In another line of research, a supervised pattern mining step is performed for feature construction in the pre-processing of classification algorithms. The discovered patterns, which have been described as *discriminative patterns* [59, 60] or *minimum predictive patterns* [29, 30], are then used as additional features in an arbitrary – possibly black-box – classification algorithm. This approach has achieved considerable improvements of the classification accuracy in some settings [59, 30].

The LeGO-approach [148] can be considered as a mixture between these approaches. In this framework, a set of local interesting patterns is discovered first. In a second step, called *pattern set discovery*, a subset of these patterns with little redundancy is selected. In a third step, the resulting patterns are then combined in a global model, which can be used for classification, but also for other data mining tasks.

2.10.3 Association Rule Mining

Association Rule Mining [4] is a popular descriptive and *unsupervised* data mining method. It aims at discovering interesting relations between any attributes in the dataset. The relations are expressed as rules. The key technique for discovering associations is the discovery of *frequent itemsets*, that is, the identification of selector sets (itemsets) that occur often together in the dataset.

Subgroup discovery focuses on dependencies of a certain property of interest. Therefore, it can be considered as a special case of association rule mining, where the right-hand side of the association rule is restricted to the target concept of the subgroup discovery task. On the other hand, frequent itemset mining – the key step of association rule mining – can be performed by generic subgroup discovery algorithms. For that purpose a very simple interestingness measure is applied, which scores only the number of instances covered by a subgroup (support).

The close relationship between supervised pattern mining (such as subgroup discovery) and frequent itemset mining is reflected in the mining algorithms for these tasks. A very simple mining algorithm discovers supervised patterns in a two-step process: First discover all frequent patterns in the search space, then filter out patterns that show no interesting distribution of the target concept, see for example [59]. Of course, this trivial approach does not match the runtime performance of specialized algorithms. More advanced algorithms for subgroup discovery are adaptations of association rule algorithms, see Chapter 3.

The interestingness of association rules has been determined by minimum thresholds for *support* and *confidence* of the rule. However, in the last two decades a large amount of other, more sophisticated criteria of interestingness have been proposed. Many of these measures are also utilized for the task of subgroup discovery, see Section 2.5 for a detailed discussion.

While subgroup discovery uses datasets with diverse types of attributes, association rule mining is originally concerned with itemset data, that is, datasets with only binary attributes. Although a majority of research on association rules focuses on this setting, extensions have been proposed that handle also other data types, such as numeric data [28], sequential data [5], and graph data [121]. A recent overview on frequent pattern mining is provided by Han et al. [109].

2.10.4 Clustering

Clustering is the task of partitioning the instances of the dataset into groups, such that instances within one group are similar and instances in different groups are dissimilar. This differs from subgroup discovery in two essential properties: First, it is unsupervised and not focused on a specific property of interest. Second, the discovered model (the set of clusters) is not necessarily interpretable by humans. However, in a subfield of clustering called *conceptual clustering* instances are "grouped into clusters, which have a 'good' description in some language" [211]. For this task, Zimmermann and de Raedt proposed a general algorithm that could be categorized as a variant of exceptional model

mining algorithm [274]. In a divisive approach, existing clusters are iteratively divided into sub-clusters. To identify new sub-clusters a pattern mining step is performed, which is based on a *category utility* interestingness measure over all attributes. That is, in the notation of exceptional model mining all attributes are used as model attributes as well as search attributes.

Another subfield of clustering, which is somewhat related to subgroup discovery, is *subspace clustering*: It aims at identifying clusters of instances in arbitrary subspaces that are induced by subsets of attributes. As in subgroup discovery, a search space is explored that is implied by the set of attributes. However, in contrast to subgroup discovery, not the set of assessed instances is altered for each candidate in the search space, but the scoring function. For a review of subspace clustering methods, we refer to [204].

Beside this approach, research on subgroup discovery and clustering are only loosely connected.

2.11 Overview of Notations

For the convenience of the reader some notations, which are used in different chapters of this work, are summarized in Table 2.2: Some conventions are used to simplify remembering these: For general components of a dataset, like the overall set of instances, calligraphic fonts ($\mathcal{D}, \mathcal{I}, \mathcal{A}$) are used. Sets of instances, e.g., all instances covered by a subgroup, are described by two latin letters ($sg(P), tp(P), fp(P)$). Counts of such sets are denoted by a single letter with subscripts ($i_P, i_\emptyset, p_P, \ldots$). For aggregated information on the target concept in a set of instances greek letters with subscripts are utilized.

2.12 Summary

This chapter provided the background for the novel techniques presented in this work by reviewing previous research on subgroup discovery. It introduced formal and informal definitions of this task and discussed fundamental options, e.g., for the search space or for the target concept. Additionally, criteria to select interesting patterns, i.e., interestingness measures and constraints, were presented. In particular, interestingness measures for subgroup discovery with numeric target concepts and generalization-aware interestingness measures were reviewed in detail as a pre-condition for the novel algorithms presented in Chapters 4 and 6 and the new interestingness measures introduced in Chapter 7. Furthermore, several general issues of subgroup discovery were described. This included the integration of background knowledge, process models for iterative and interactive subgroup mining, the statistical significance of subgroup discovery results, and the relation between subgroup discovery and other data mining tasks. Finally, the notation used in this work was summarized. A discussion of efficient algorithms for subgroup discovery will be provided in the next chapter.

Table 2.2: Summary of some notations used in this work.

Notation	Explanation
$\mathcal{D}$	The dataset
$\mathcal{I}$	The set of all instances in $\mathcal{D}$
$\mathcal{A}$	The set of all attributes in $\mathcal{D}$
$\mathcal{S}$	The set of all (base) selectors to that can be combined in subgroup descriptions
Σ	The search space (set of all candidate subgroup descriptions) of a task
$\Sigma^{1..d}$	The search constraint to descriptions with at most d selectors
P	A subgroup (description)
$P(i)$	A boolean that indicates if instance $i \in \mathcal{I}$ is covered by P
$T(i)$	The value of the target concept in instance i.
$sg(P)$	All instances covered by P (the subgroup of P)
$tp(P)$	All positive instances covered by P
$fp(P)$	All negative instances covered by P
i_P	The number of instances in $sg(P)$
$i_\emptyset$	The number of instances in the overall dataset
p_P	The number of instances covered by P that have a positive target concept
$p_\emptyset$	The number of instances in the overall dataset with a positive target concept
n_P	The number of instances covered by P that have a negative target concept
$n_\emptyset$	The number of instances in the overall dataset with a negative target concept
τ_P	The target share in the subgroup described by P
$\tau_\emptyset$	The target share in the overall dataset
μ_P	The mean value of the numeric target attribute in the subgroup described by P
$\mu_\emptyset$	The mean value of the numeric target attribute in the overall dataset

3 Subgroup Discovery Algorithms

The previous chapter discussed various criteria on *which* subgroup patterns could be considered as interesting. This chapter will review methods from literature, *how* these patterns can be retrieved from a dataset *efficiently* in terms of runtime and memory requirements.

A thorough overview on the field of efficient subgroup discovery is difficult since the task of mining such conjunctive patterns has been studied under varying terminology. It has been described as *supervised pattern mining*, *contrast set mining*, *supervised descriptive rule induction*, *classification rule mining*, *correlated pattern mining*, *discriminative pattern mining*, *mining association rules with fixed consequents* and several others, see also Section 2.11. Although these methods differ with respect to the overall goal and to criteria by which patterns are selected, the mining algorithms is mostly equivalent. In particular, the descriptive or predictive nature of the task has little influence on the actual mining techniques. The differences between the tasks lie mainly in the applied interestingness measures, which can be exchanged in most algorithms with little effort. The following chapter provides an overview on algorithms for subgroup discovery that also takes algorithms of these related subfields of data mining and machine learning into account.

It can be observed that algorithmic advances in these fields often built upon existing algorithms and improve only certain aspects. Therefore, this chapter will first introduce a categorization scheme by decomposing algorithms in their constituents: the applied enumeration strategy, the used data structures and the employed pruning strategies. Then, previous solutions from literature for these components will be discussed. Afterwards, existing algorithm from literature are reviewed. Instead of describing the algorithms one-by-one in detail, most of the algorithms can concisely be described in terms of the introduced categories by referencing the respective algorithmic components. Additionally, algorithms for subgroup discovery in specific settings are summarized. This includes mining for relevant subgroups, subgroup discovery with describing selectors over numeric attributes, sampling approaches and distributed subgroup mining.

An overview on subgroup discovery algorithms has previously been provided by Herrera et al. in [117]. Their work, however, focuses on algorithms using the terminology subgroup discovery, ignoring methods from closely related fields. Another overview was given by Bringmann et. al [47], see also [271]. In their work, the authors emphasize the relatedness between tasks using different terminologies and also recognize the orthogonality between the applied data structure and the resulting patterns. However, they focus on the question, which patterns should be selected with respect to a classification goal in an iterative selection process, and do not discuss algorithmic efficiency in detail. In contrast to these summaries, the overview presented next in this work provides a detailed

discussion of the approaches with a focus on the runtime and memory performances of the different algorithmic components. This chapter focuses on classical subgroup discovery with a binary target concept and a traditional interestingness measure. Further method for subgroup discovery with numeric target concepts, exceptional model mining and mining with generalization-aware interestingness measures are reviewed as related work for the novel contributions in Chapter 4, Chapter 5, and Chapter 6.

3.1 Algorithmic Components

In a typical subgroup discovery task very large numbers of subgroup descriptions are to be considered, cf. Section 2.8. To perform the evaluation of these candidates efficiently, numerous different algorithms have been proposed in the last decades to match runtime and memory requirements in applications. In literature it is often claimed that "a new algorithm" for a supervised pattern mining task is introduced. However, innovations can often be attributed to either a new interestingness measure plugged into an existing algorithm or to one of the three following algorithmic components, leaving the other two unchanged or unspecified:

1. The *enumeration strategy* defines, in which order subgroup patterns are evaluated. In the case of heuristic strategies this also dictates, which patterns are evaluated, respectively ignored.

2. The *data structure* determines how the data is stored to allow for the efficient evaluation of candidates.

3. The *pruning strategy* identifies candidates, which are not required to be evaluated given previously acquired information. This may overlap with search strategies in the case of non-exhaustive approaches.

The enumeration strategy and the data structure is often independent of the applied interestingness measure. Thus, if an algorithm uses a specific interestingness measure and proposes a novel enumeration strategy, this algorithm can be transfered to other interestingness measures with minimal effort.

3.2 Enumeration Strategies

Most popular algorithms handle subgroup discovery as a search problem in the space of candidate patterns. This space forms a (mathematical) lattice with the empty subgroup description as supremum and the description that contains all selectors as infimum, see e.g., [266, 196]. In the vast majority of subgroup discovery algorithms, the search usually starts at the empty subgroup description and explores the search space by following the specialization links. Alternative search strategies, which traverse the search space "top-down", that is, from the longest and to the shortest descriptions, as well as a hybrid approach are outlined in [266] in the context of association mining. An alternative

approach to subgroup as search was proposed in [67] and [200]: Here pattern mining is described as a constraint satisfaction problem, which is solved by specialized constraint solvers. While these provide great flexibility in terms of additional constraints and additional pruning bounds, they are still outperformed by specialized implementations based on traditional mining algorithms, cf. [200] The remainder of the chapter will focus on the subgroup discovery as a "bottom-up" search problem since this formulation is most common.

Bottom-up enumeration strategies utilize different refinement operators to generate specializations of a pattern: A *full refinement operator* generates patterns by adding any of the non-used selectors $sel^{new} \in \mathcal{S}$ to the current pattern P.

$$FullRef(P) = \{P \wedge sel^{new} | sel^{new} \in \mathcal{S} \wedge sel^{new} \notin P\}$$

As a result, a candidate pattern is potentially generated multiple times by different generalizations. A full refinement operator is for example used in beam-search. To avoid multiple candidate checks, an *optimal refinement operator* can be used to generate specializations, as it is done for example in standard depth-first-search or best-first-search. An optimum refinement operator creates only a subset of direct specializations for a pattern by utilizing an ordering $\succ$ of the selectors, e.g., a lexical ordering. The refinement operator considers only the refinements, in which the one selector, which is added to the pattern P, occurs after all selectors contained in P according to the ordering $\succ$, see e.g., [272]:

$$OptRef(P) = \{P \wedge sel^{new} | (sel^{new} \in \mathcal{S}) \wedge (\forall sel^{old} \in P : sel^{new} \succ sel^{old})\}$$

When the search reaches an unexplored candidate pattern, then this pattern is evaluated. For the evaluation of a subgroup pattern, the score of the pattern is computed according to the chosen interestingness measure. This is done by computing statistics of the subgroup using the utilized data structure. Then, the score is compared with the interestingness threshold of the result set. If the interestingness value is high enough, then the pattern is added to the result set, potentially replacing previously found subgroups in a top-k-approach. For shorter notation, these actions will be summarized in the following just as the *evaluation of a subgroup pattern.*

In general, one can distinguish between two different main categories of search strategies: *exhaustive* search strategies aim at finding a guaranteed optimal solution to the subgroup discovery problem by traversing through the complete search space (and applying safe pruning strategies). In contrast, *heuristic* strategies try to identify as good as possible, but not necessarily optimal patterns, in limited time.

3.2.1 Depth-first-search and Variants

The following section describes the basic depth-first-search approach to subgroup discovery and discusses some variations.

Basic depth-first-search for subgroup discovery: One of the most simple as well as most common strategies for graph traversal is *depth-first-search*. Using this strategy for subgroup discovery, the search follows the specialization links within a branch of the search tree as deep as possible. Only after reaching the full subgroup description or a candidate pattern that triggers cutting the search space by pruning, backtracking is applied. An example of this enumeration order is shown in Figure 3.1 (a). The advantages of this approach are convenient implementation and very low memory requirements, which are in $O(d)$ for a maximum number of describing selectors d. For an optimal refinement operator, it can be avoided to store a todo list of the candidate patterns explicitly. Instead, candidates can be derived from the selectors already visited in this branch, see Algorithm 1.

Algorithm 1 Subgroup discovery by depth-first-search

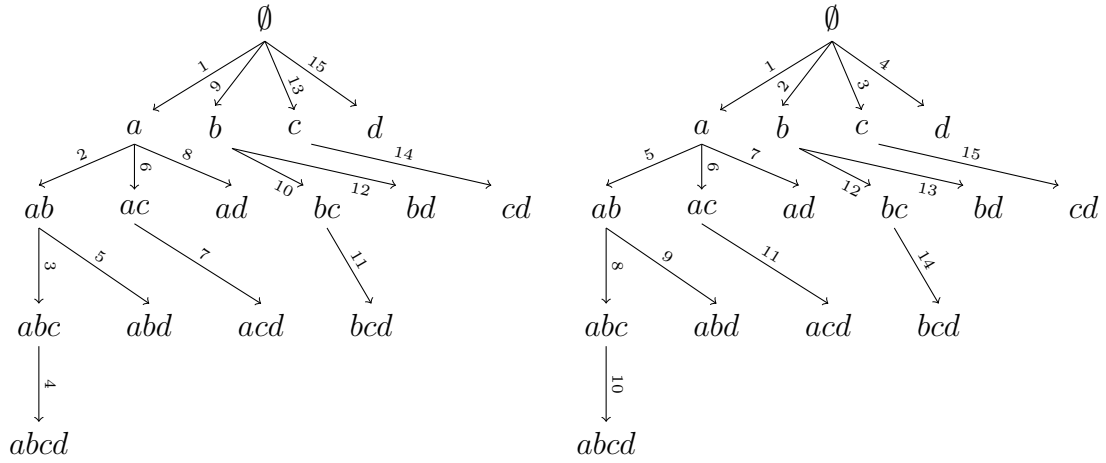
```
function DFS-SD(maxDepth)
    DFS − SD(∅, allSelectors, maxDepth)
function DFS-SD(prefix, remainingSelectors, maxDepth)
    EVALUATE(prefix)
    if (prefix.size < maxDepth) ∧ (¬pruning(prefix)) then
        for all sel in remainingSelectors do
            DFS − SD(prefix ∪ sel, remainingSelectors \ sel, maxDepth)
```

Iterative deepening In an interactive setting, it is often desired to obtain preliminary results early and deliver full results later on. For that purpose, the *"iterative deepening"* (see [154]) variant can be favorable: A complete search task using depth-first-search is performed several times with an additional maximum search depth constraint $d = 1, 2 \ldots, maxDepth$ that is increased in each iteration, see Algorithm 2. In doing so, one can achieve optimal results for short descriptions fast for immediate presentation. Since the search space for $d + 1$ is substantially larger than the one for d, this does not necessarily increase the runtimes by much. In addition, cached results from previous iterations can also be used in later iterations for improved pruning of the search space using optimistic estimates. Iterative deepening has been used for efficient mining of relevant patterns [104], cf. Section 3.6.1. It has also been used in the context of (graph) pattern mining [48].

Algorithm 2 Iterative deepening subgroup discovery

```
function ITERATIVE DEEPENING SD(maxDepth)
    for i = 1 → maxDepth do
        DFS-SD(maxDepth)
```

Forward checking A variation of standard depth-first-search is implemented by the addition of *forward checking*. That is, whenever the enumeration process reaches a



(a) Standard depth-first-search

(b) Depth-first-search with forward checking

Figure 3.1: Enumeration order for the selector set {a, b, c, d} for standard depth-first-search (a) and depth-first-search with forward checking (b). Small numbers on the specializations edges indicate the order, in which candidates are evaluated.

candidate, immediately *all* its direct successors are evaluated. Then, however, the search continues with the first of these successors as in standard depth-first-search. An example for the resulting enumeration order is shown in Figure 3.1 (b). Descendants of the other children are only processed after the enumeration of all previous branches of the search tree has been completed. This adaptation yields two main advantages: First, it often enables additional pruning options since for future patterns more generalizations have been already visited. Second, it allows for reordering of search tree branches according to the computed statistics, such that more promising patterns are explored first. While this modification slightly increases the memory requirements of depth-first-search from $O(d)$ to $O(d \cdot |S|)$, this approach is still much less demanding in that regard in comparison to other exhaustive enumeration strategies, e.g., levelwise approaches.

Algorithm 3 Depth-first-search with forward checking

```
function DFS-FC(prefix, remainingSelectors, maxDepth)
    nextSelectors ← ∅
    for all sel in remainingSelectors do
        EVALUATE(prefix ∪ sel)
        if ¬prune(prefix ∪ sel) then
            nextSelectors ← nextSelectors ∪ sel
    if prefix.size < maxDepth then
        for all sel in nextSelectors do
            DFS-FC(prefix ∪ sel, nextSelectors \ sel, maxDepth)
```

Re-ordering: In a top-k-approach, it is favorable to discover the best subgroups early in the process since raised quality thresholds in the result set allow for optimized pruning. To achieve this, the ordering of the nodes in the search space can be dynamically adapted in depth-first-search according to the interestingness score or optimistic estimates of the evaluated subgroup patterns. Two different reordering strategies can be applied:

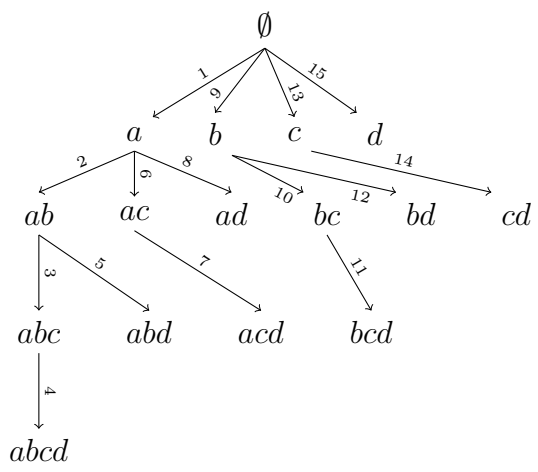
- *Full reordering:* First, one can change the order, in which the selectors are used to generate specializations in the classical depth-first-search algorithm. This also changes the contents of different branches within the search tree. However, the longest branch of the search tree is still visited first. For example, in a search space with selectors a, b, c and d, the order $(a \rightarrow b \rightarrow c \rightarrow d)$ implies that the pattern $(a \wedge b \wedge c \wedge d)$ is contained in the first branch of the search tree as a descendant from a and is visited after a and before b. Using this variant of reordering, changing this order to $(b \rightarrow a \rightarrow c \rightarrow d)$, $(a \wedge b \wedge c \wedge d)$ is still part of the first branch of the search tree, but now as a descendant from b, being visited after b, but before a.

- *Branch reordering:* Second, the contents of the branches can be determined and fixed by an initial ordering. Reordering then only decides in which order these branches are evaluated. As an example, assume again that $(a \rightarrow b \rightarrow c \rightarrow d)$ is the initial ordering. Then, regardless of any later reordering $a \wedge b \wedge c \wedge d$ is always contained in the branch of the search tree that begins with a. However, if this branch is evaluated before or after the branch starting with b can depend on later reordering.

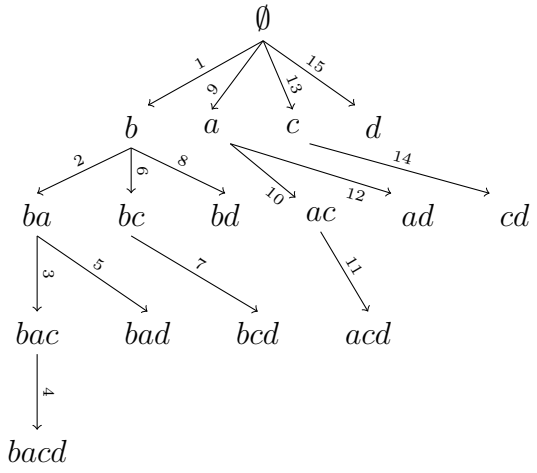
Of course, both variants of reordering can be applied at any level of the search to adapt to additional information obtained in the search. An example for both variants of reordering is given in Figure 3.2.

Reverse depth-first-search: As an extreme case for branch reordering, one can completely reverse the standard ordering for depth-first-search, such that the shortest unconsidered branches are always evaluated next, see Figure 3.2 (d) for an example. This *reverse depth-first-search* ensures that for each subgroup all generalizations are visited beforehand. For this reason, this enumeration order has been proposed in the context of frequent itemset mining [156], in particular non-derivable itemset mining, a generalization-aware variation of this task [52, 53, 54].

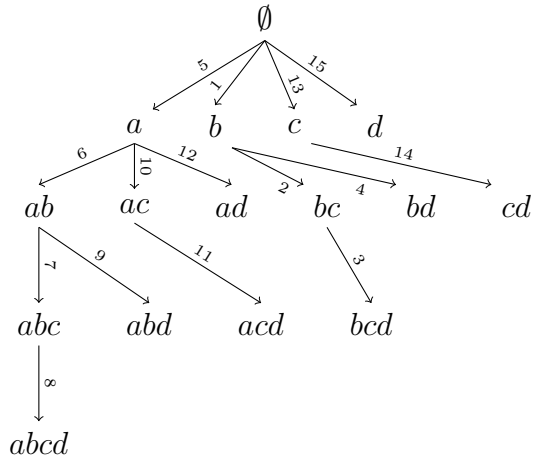
In the field of subgroup discovery, this ordering has to the author's knowledge not yet been used in mining algorithms. It could especially be suited for search tasks with generalization-aware interestingness measures, which require statistics of generalizations for the evaluation of a subgroup, since each subgroup is evaluated after all its generalizations. This "generalizations first" property is also shared by the family of level-wise search strategies, which will be discussed next.



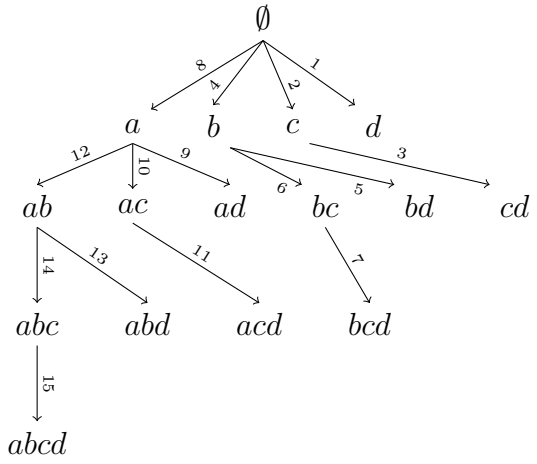
(a) Standard depth-first-search



(b) Full reordering depth-first-search



(c) Branch reordering depth-first-search



(d) Reverse depth-first-search

Figure 3.2: Different orderings for traversing the selector set {`a`,`b`,`c`,`d`}. Small numbers on the specializations edges indicate the order, in which candidates are visited when using (a) standard depth-first-search using lexical ordering, (b) full reordering switching `a` and `b` at the first level, (c) branch reordering switching `a` and `b` at the first level, and (d) reverse depth-first-search.

Algorithm 4 Reverse depth-first-search

```
function REVERSE-DFS(prefix, remainingSelectors, maxDepth)
    for i = remainingSelectors.count → 0 do
        i ← i − 1
        nextSel ← remainingSelectors.get(i)
        prefix.add(nextSel)
        EVALUATE(prefix)
        if (prefix.size < maxDepth) ∧ (¬pruning(prefix)) then
            newRemainingSels ← ∅
            for j = i + 1 → remainingSelectors.size() do
                j ← j + 1
                newRemainSels ← newRemainSels ∪ remainingSelectors.get(j)
            REVERSE-DFS(prefix,newRemainSels, maxDepth)
        prefix.remove(nextSel)
```

3.2.2 Levelwise Approaches

In contrast to the depth-first-strategies described before, levelwise strategies strictly evaluate in order of the number of describing selectors: For example, each candidate pattern with two describing selectors is evaluated after each candidate described by only a single selector, but before any candidate pattern described by three selectors.

Breadth-first-search A traditional alternative to depth-first-search is *breadth-first-search*. Using this strategy, not yet evaluated candidates are stored in a FIFO todo list. The search starts with a list that only contains the empty pattern. Iteratively, the first pattern is removed from the list and evaluated. Then, all specializations of this patterns, which currently are not contained in the todo list, are added at the end of this list. In doing so, the search space is traversed in an order that explores shorter patterns strictly before longer patterns. While this approach guarantees that every candidate subgroup is visited exactly once, at one point during the search all patterns of length d are contained in the todo list. Therefore, the memory requirements for this list is in $O(|\mathcal{S}|^d)$, which is too large for many practical applications.

Apriori As an improvement, the well-known *apriori* strategy, which origins in frequent item set mining [4, 7] and has later been transferred to classification rule learning and subgroup discovery [131, 135], achieves the same enumeration order without storing an explicit todo list (although the list of candidates can be considered as kind of a replacement for that). Instead, candidate patterns for the next level can be generated one by one from the shorter patterns of the previous level. The search starts by evaluating all patterns of level 1, that is, all patterns described by a single selector. Whenever all patterns of a level are evaluated, the set of patterns is filtered using available pruning criteria, see Section 3.4. Candidate patterns for the level $d+1$ are then generated from this filtered list L of patterns of level d as follows: For each pair of patterns in L, which

share the first $d-1$ describing selectors, a new candidate with $d+1$ selectors is created by joining the selector sets of both patterns. For the novel pattern, it is then checked if also all other generalizations are contained in the list L. Only if this is the case, the new pattern of level $d+1$ needs to be evaluated. The search continues by evaluating and generating candidates for the next level iteratively until the maximum search depth is reached or no more candidates are generated. Note, that this approach does not require to store candidate patterns for the last level of search in a list since the patterns can be generated and evaluated ad hoc from the patterns of the previous level. Therefore, the worst case memory requirements are in $O(|\mathcal{S}|^{d-1})$, by a factor of d smaller than in the naive breadth-first-search. However, optimized pruning of candidate patterns often results in memory usage that is by orders of magnitudes lower.

The main advantage of apriori in comparison to the depth-first-search approach is that it provides substantially better pruning options: For each pattern *all* generalizations are checked for pruning possibilities before the evaluation of the pattern, while depth-first-search only checks one direct generalization. This is accomplished, however, at the cost of increased memory requirements ($O(|\mathcal{S}|^{d-1})$ vs. $O(d)$).

Algorithm 5 Apriori for subgroup discovery. The evaluations in the for-loop can be accomplished in a single run over the database, updating the counts for each candidate in parallel. Furthermore, the candidate sets C_k and C'_k are not necessarily built explicitly, especially on the last level of search. Instead, it is sufficient to be able to iterate over this set. Note, that the algorithm provided here is not identical with the Apriori-SD presented in [135] since it does omit the wrapper for weighted covering.

function SD-APRIORI(maxDepth)
  $L_1 \leftarrow allSelectors$
  $k \leftarrow 2$
  while $(L_{k-1} \neq \emptyset)$ **do**
    $L_k \leftarrow \emptyset$
    $C_k = \{X \cup Y | X, Y \in L_{k-1}, |X \cap Y| = k-2\}$ // "generate step"
    $C'_k = \{X \in C_k | L_{k-1}$ *contains* k *specializations of* $X\}$ //"prune step"
    for all c in C'_k **do**
      EVALUATE(c)
      if $(\neg pruning(c)) \wedge (k < maxDepth)$ **then**
        $L_k \leftarrow L_k \cup c$
    $k \leftarrow k+1$

3.2.3 Best-first-search

Another popular search strategy that is used for the exploration of graphs is *best-first-search*, see for example [205]. A well known example in this category of search strategies is the A^* algorithm for pathfinding. In the best-first approach, all candidates that have been evaluated by the algorithm, but have not yet been expanded are stored in a todo list, which is sorted by a quality criterion. For supervised pattern mining, an

upper bound of the interestingness of all respective specializations is used as a sorting criterion [248, 274]. In each step of the algorithm the first element in the list, that is, the most promising node, is evaluated and expanded by adding its specializations to the todo list using an optimal refinement operator. The insertion in the todo list can be skipped if the upper bound for the pattern is lower than the interestingness value that is currently required by the result set. Additionally, whenever the quality threshold in the result set is raised, all candidates with an optimistic estimate below the quality threshold can be removed from the todo list. That can be efficiently accomplished since the list is ordered by that criterion. Optimistic estimates can be exploited very efficiently by using best-first-search: Each candidate is evaluated after all other patterns with a higher optimistic estimate. This increases the likelihood that the pattern is not required to be evaluated, because it is pruned beforehand. In a worst case scenario, all subgroup descriptions of the last level of search are contained in the todo list at some point during the search. Therefore, the memory usage can be in $O(|S|^d)$. As for Apriori, however, the memory requirements in practical applications is heavily influenced by the quality of the applied pruning criteria.

As Apriori, this search strategy also guarantees that all generalizations of a pattern are always explored before the pattern itself if the upper bound for the interestingness of a generalization is higher than that of its specialization, which is the case in the traditional subgroup discovery setting.

Algorithm 6 Subgroup discovery by best-first-search, see also [274]

```
function BEST-FIRST-SD(prefix, remainingSelectors, maxDepth)
    TODO ← {∅} // todo list is initialized with the empty pattern
    while not TODO.isEmpty() do
        p = TODO.popBest()
        if (prefix.size < maxDepth) then
            Candidates = p.specializations //using an optimum refinement operator
            for all c in Candidates do
                EVALUATE(c)
                if (¬pruning(c)) then
                    TODO ← TODO ∪ c
            pruneAccordingToNewThreshold(TODO)
```

3.2.4 Beam-search

In contrast to these exhaustive methods, also heuristic methods are often employed for subgroup discovery. For these methods, the enumeration order does not guarantee to traverse all patterns in the search space. Instead, it tries to find as good patterns as possible in shorter time by evaluating only promising candidates.

The most popular approach to heuristic supervised pattern mining is *beam-search*, see e.g. [63]. It follows the intuition that patterns are more likely to have interesting statistics if also their generalization are interesting. The specification of the algorithm

includes a parameter w, the *beam width*. At any point in the search, the currently best w hypotheses are stored in a list (the "beam"). The search starts by evaluating all patterns of size 1 and then follows an iterative approach. In each iteration, the algorithm evaluates all specializations of all patterns within the beam, possibly replacing candidate in the beam with higher scoring subgroups. This is repeated until no more improvements in the beam are achieved.

For the parameter w often the desired size for the final result set is used, but this is not mandatory. Beam-search usually utilizes a full refinement operator [63]. Therefore it is quite possible that the same pattern is evaluated multiple times.

Algorithm 7 Subgroup discovery by beam-search, see also [63, 14].

```
function BEST-FIRST-SD(prefix, allSelectors, maxDepth)
    BEAM ← {∅} // Beam is initialized with the empty pattern
    while BEAM changed in last iteration do
        for all p in BEAM do
            if (prefix.size < maxDepth) then
                for all sel in allSelectors do
                    candidate ← p ∪ sel
                    EVALUATE(candidate)
                    if (candidate.quality >(worstQuality in BEAM)) then
                        replace worst pattern in BEAM with candidate
```

3.2.5 Genetic Algorithms

Another heuristic search strategy for the subgroup discovery task is given by genetic algorithms, see for example [99, 194]. Genetic algorithms are "computer programs that mimic the processes of biological evolution in order to solve problems [...]" [194]. In genetic algorithms, a set of candidate solutions represented by their inherent properties are iteratively evolved towards the better solutions of an optimization problem by altering (*"mutation"*), re-combining (*"crossover"*), and choosing (*"selection"*) candidates for the next iteration with respect to an evaluation function (*"fitness"*).

Berlanga and del Jesus [39] proposed to apply this methodology to the task of subgroup discovery. In their approach, each candidate is represented by a bitstring that defines the subgroup description, that is, each position in the bitstring represents a possible selector. Bits in the bitstring, which are set to *true*, are interpreted as conjunctions if the attribute of the selector is different, and as internal disjunctions if the attribute is identical for two or more selectors. Subgroup descriptions for the next population are generated by applying a two-point crossover operator for recombination and a biased uniform operator for mutation, which aims at partially eliminating variables from the subgroup description to achieve more general rules. The fitness of individuals is determined by the number of other individuals dominated with respect to the multiple applied interestingness measures.

3.3 Data Structures

The next section describes different methods how the data can be stored in order to allow for efficient evaluation candidate of subgroup patterns.

3.3.1 Basic Data Storage

The most basic setup uses data directly from a file-system or a database, which is possibly only accessible via network. Since access times for these storage devices are by orders of magnitude slower than for in-memory operations, it is always to be preferred to load the complete data in the main memory if possible, i.e., if the available main memory is large enough for the dataset.

After pre-processing the dataset is stored in a simple tabular form, in our setting a single table in *horizontal layout*. That is, similar to relational databases each instance is stored in one line and each attribute is stored in one column of a table. Cell values represent the value of the instance of this line for the attribute in the respective column. Whenever a candidate patterns is evaluated, all conditioning selectors are required to be checked against the original data. Depending on the exact storage system and the complexity of the applied selection expressions those checks are potentially time-consuming. Therefore, more advanced approaches apply these checks only once and cache the results in optimized data structures, which can then be used for a more efficient mining process. These will be discussed next.

3.3.2 Vertical Data Structures

In a typical application, the data is stored in a so-called horizontal layout. That is, for each case/transaction as key the respective attribute values are stored. For efficient evaluation of subgroup patterns, *vertical data representations* can be used instead. These have been studied in depth in the field of frequent item set mining [266] and have later been transferred to supervised settings. In these representations, all cases for a pattern are stored in a specialized structure, i.e., the subgroup description is used as a key. This can be seen as a 90° shift from the typical horizontal data representation, see Figure 3.3

TID-lists TID-lists [266] are basic vertical data representations. In these, each case/-transaction is referenced by a unique (integer) identifier. For each selector in the search space as well as for the binary target concept, an ordered list is created from the original data representation. These contain identifiers of all cases, which are covered by the respective selector, see Figure 3.3 (b) for an example. The mining of the conjunctive search space can be accomplished by using only these lists: To compute a TID-list, which corresponds to a conjunction of selectors, one has to determine the list entries that are shared by all lists for the single selectors. Since the lists are ordered, this can be achieved in a single pass over all lists. Of course, TID-lists for conjunctive patterns can be cached and reused to speed up the computation of further specializations. For traditional subgroup discovery, the overall instance count and the number of positive

instances are required to compute the interestingness of a pattern. The instance count of a pattern is given by the number of identifiers stored in the TID-list. To obtain the count of positive instances, the shared members of the respective TID-list and the list for the target concept are counted.

The memory as well as the runtime requirements are strongly dependent on the size of the TID-lists. Therefore, TID-lists are efficient for sparse datasets, where each selector covers only a few instances in the database, but are less favorable for dense datasets.

As an improvement for TID-lists, Zaki and Gouda proposed *DiffSets* [267]. In this data structure, lists for a conjunction of d selectors are not materialized explicitly. Instead, only the differences to a direct generalization are stored, which can reduce the required memory.

Bitsets An alternative vertical data structure is given by bitsets (also called bitmaps, bitstrings or bitvectors), which have been used for example in the Explora system [139, 140]. In these data representations, the instances that correspond to a subgroup pattern are stored in words of single bits. Each word contains as many bits as the dataset contains cases. The *i-th* bit in each word belongs to the *i-th* instance in the dataset. The bit is set to 1 if this instance is covered by the respective subgroup description, and is set to 0 otherwise. Similarly to TID-lists, one such bitset is generated for each basic selector and one additional bitset reflects the occurrence of the target concept. These initial bitsets for the selectors can be generated in a single pass over the dataset.

To create bitsets, which correspond to conjunctive patterns, a logical AND operation is performed on the bitsets of the contained selectors. For example, to compute the bitsets which corresponds to the pattern $A \wedge B$, a bitwise AND is executed on the bitset corresponding to A and the bitset corresponding to B, see Figure 3.3 (c) for an example. This is especially efficient since most third-generation programming languages achieve high performances for logical operations on bitsets. The size of a subgroup can then be derived by determining the cardinality of the bitset, that is, the number of bits set to 1. The number of positive instances can be computed similarly: A logical AND is performed on the bitset of the subgroup and the bitset corresponding to the target concept. The number of bits with value 1 are counted. For the efficient counting of bits in a bitset, which are set to *true*, specialized algorithms and even supporting hardware implementations have been developed, see for example [76].

In contrast to TID-lists, the memory requirements for the bitset representation of a single pattern is independent of the dataset density: For each selector and for the target concept a bitsets of length $i_\emptyset$ is required, where $i_\emptyset$ is the number of instances in the dataset. Thus, the overall memory requirements for the data representation using bitsets are $i_\emptyset \cdot (|\mathcal{S}|+1)$. Additionally, the runtime for performing a single AND operation between data structures does not depend on the data sparseness, unlike to the TID-list approach.

Sorted representations For the use of vertical data representations in the context of subgroup discovery, we have proposed a minor improvement in [177]: Instead of storing

all instances for a subgroup pattern in a single bitset and then extracting the positive instances with an additional AND-operation, one can initially sort the instances with respect to the target concept into positives and negatives. Then, for each selector two bitsets are created separately, one for the positives and one for the negatives. Representations for conjunctive patterns are then generated analogously by performing a logical AND on both representations, e.g., bitsets. This approach requires not only one less AND operation in the evaluation of each subgroup, but can further speedup the computation: In a first step the data structure is generated for the positive instances only. Then, pruning options can be checked based on this information, for example by exploiting a constraint on the number of positive instances, or by computing optimistic estimates. Thus, the computation of negative instances can potentially be skipped completely. This is especially advantageous in unbalanced data, i.e., if only a small part of the data has a positive target concept.

3.3.3 FP-tree Structures and Derivates

FP-trees have been introduced into frequent itemset mining for the well-known FP-Growth algorithm [111] and have later been transferred to the field of subgroup discovery with binary target concepts [22]. As a major contribution of this work, it will be shown later in this work how these data structures can be extended for subgroup discovery with numeric properties of interest and to exceptional model mining, see Chapter 4 and Chapter 5.

Mining with FP-trees employs a divide-and-conquer strategy using extended prefix tree structures. Like vertical data structures they provide a condensed but with respect to the task complete representation of the dataset. However, the instances, which are covered by a pattern, are not directly stored. Instead, aggregated information on the statistics that are necessary for the evaluation of patterns is saved. For association rule mining, this is the number of instances covered by a pattern, for classical subgroup discovery the count of positive and negative instances covered by the pattern. Information can be aggregated and conditioned on certain selectors to derive statistics for conjunctive patterns as necessary.

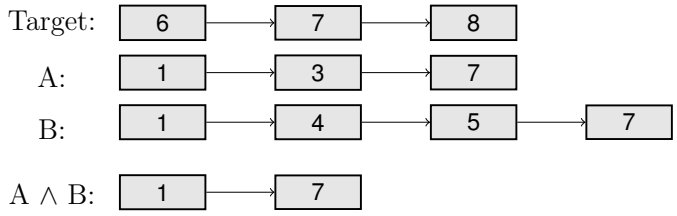
This section first outlines FP-trees in its original frequent item set mining setting, then adaptations for subgroup discovery are discussed.

Structure and construction of FP-trees A FP-tree consists of tree nodes, which are connected by two different types of links. Each tree node (except the root node) corresponds to a selector and stores (in the traditional frequent item set mining setting) a frequency count. Directed edges between the tree nodes induce a parent-child structure. Paths from the root node to any other node can be interpreted as a conjunction of corresponding selectors. Additionally, links between nodes that refer to the same selector are maintained in a second, auxiliary link structure. A *header table* aggregates for each selector the information contained in the respective list.

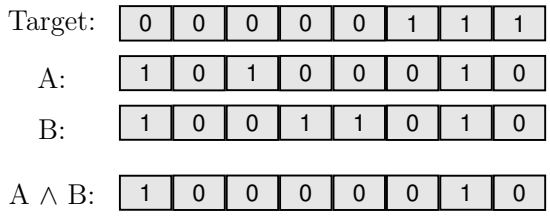
The construction of FP-trees consists of two steps: In the first step, all simple patterns in the search space are evaluated, filtered using available pruning mechanisms and sorted

id	*A*	*B*	*Target*
1	*t*	*t*	*f*
2	*f*	*f*	*f*
3	*t*	*f*	*f*
4	*f*	*t*	*f*
5	*f*	*t*	*f*
6	*f*	*f*	*t*
7	*t*	*t*	*t*
8	*f*	*f*	*t*

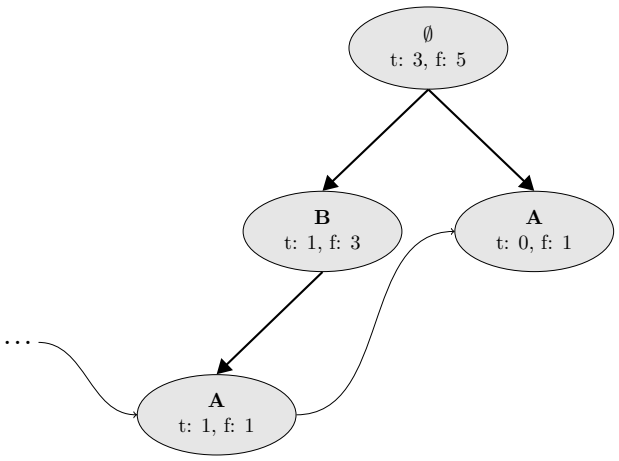
(a) Horizontal data representation



(b) TID-lists



(c) Bitsets



(d) FP-tree

Figure 3.3: An examplary dataset in (a) horizontal layout, (b) represented as TID-list, (c) as bitsets, and (d) in a FP-tree. For TID-list and Bitset representation, the data structure for the conjunction $A \wedge B$ is additionally displayed as a computation example.

in descending order according to their frequency counts. This order of the selectors increases the chance of shared prefixes between data instances, leading to much more compact representations compared to a lexical ordering. While this ordering step is not necessarily required, it is therefore highly recommended in literature, see [112].

In the second step, all instances $c \in \mathcal{I}$ of the dataset are inserted one by one into the tree structure as follows [111, 22]: First, all selectors in the search space, which apply in the instance c, are identified and added to an ordered List L according to the previously computed frequency order. Then, the *insert* (L, N) procedure is called with the complete list L, and the root node of the tree structure as arguments. This method checks if there exists a child node that corresponds to the first selector in L. If such a node exists, then the count stored in this node is incremented. Otherwise, such a node is created with an initial size of one. The new created node is inserted in the tree structure as child of the last visited node and is also added to the list of nodes corresponding to the respective selector in the auxiliary link structure. In any case, the selector is removed from the list L. Unless L is empty, the *insert* procedure is then called recursively with the last modified node N and the now shorter list L. Pseudocode of the *insert* procedure is shown in Algorithm 8.

After the insertion of instances is completed, the information in the header table is actualized. For that purpose, the counts contained in all nodes in the auxiliary list for this selector are summed up.

Algorithm 8 Insertion of an instance in a FP-tree

```
function FPTree_insert(L,N)
    nextSelector ← L.pop()
    nextNode ← N.getChild(nextSelector)
    if nextNode.isNull() then
        nextNode ← N.createChild(nextSelector, 1)
    else
        nextNode.count++;
    if not L.isEmpty() then FPTree_insert (nextNode, L)
```

Conditioning in the FP-tree The header table of the initial FP-tree contains the instance counts for all basic selectors (items). To determine the counts of conjunctive descriptions (itemsets), so-called *conditional trees* are used. A conditional tree conditioned on a selector sel_C can be derived from a FP-tree as follows: First, all prefix-paths from the root of the tree to the tree leaves that contain sel_C are identified using the auxiliary link structure. For each node of these paths the contained count is limited to the count in the lowest (leaf) node. This set of paths describes a new, conditioned dataset that contains only those instances, which are covered by sel_C and only those selectors, which occur before sel_C in the initial ordering. Using these paths, a new smaller FP-tree is constructed. Of course, conditioning an already conditioned tree on another selector

sel_{C_2}, is equivalent to conditioning on a conjunctive pattern. Thus, the search space can be explored in a recursive manner.

Due to memory restrictions FP-trees are usually applied in combination with a variant of depth-first-search. Also, reordering of the search space is limited to branch reordering: Since each conditional tree contains only selectors that precede the conditioning selectors in the initial ordering, full reordering can not be applied.

Selectors, which do not appear with the conditioning selector in any instance, are not contained in the constructed conditional tree. This substantially decreases the size of the tree and the header table. Since these combinations of selectors, which cover no instances anyway, are not contained in the FP-tree, they are also not explicitly evaluated by the algorithm. Thus, FP-trees can be considered to have an inherent pruning mechanism regarding not occurring combinations of selectors.

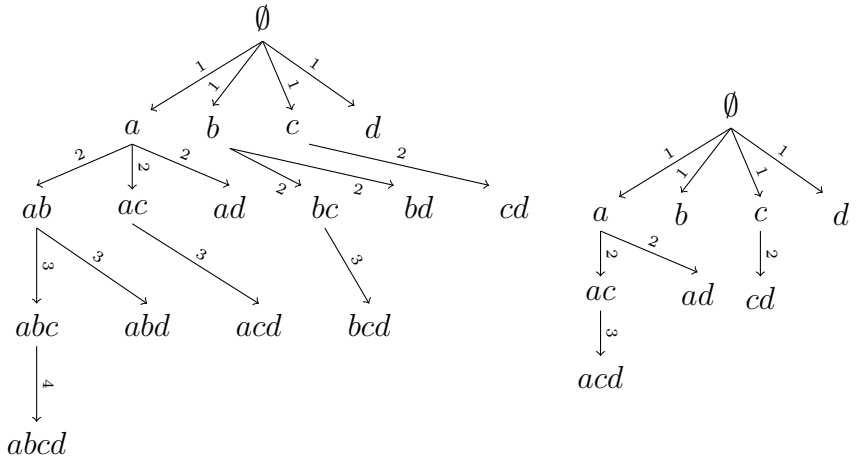
In addition to the techniques presented here, FP-trees exploits specific properties of the tree structure for speedups. Most notably, *single-prefix-path* allows evaluating patterns in parallel, if they share their only prefix path in the tree structure. For more detailed descriptions of FP-trees, we refer to [111] and [112].

Adaptation to subgroup discovery A naive approach of utilizing FP-trees for subgroup discovery is to use them to identify frequent patterns, and evaluate these candidate patterns according to the interestingness measure in an independent second phase of the algorithm, cf. [27]. However, the FP-tree can also easily be exploited for direct mining in the classical subgroup discovery setting: The number of positives $p_\emptyset$ and negatives $n_\emptyset$ in the complete dataset (and thus the target share $\tau_\emptyset$) can be computed in the initial pass over the dataset. To evaluate a subgroup pattern with respect to a traditional interestingness measure, the number of positive and negative instances are required for the respective subgroup. This can be accomplished in a FP-tree by storing not a single value for the instance count in each node of the tree and the header table, but the instance counts of positives and negatives separately. Aggregation of nodes can be performed analogously, adding these two counts up independently from each other. For more details on the adaptation of FP-trees to subgroup discovery with binary targets, we refer to [22].

Overall, FP-trees have been established as a very efficient data structure, especially for large or sparse datasets.

3.4 Pruning Strategies

Algorithms for subgroup discovery follow a branch-and-bound [164, 65] philosophy as it was introduced in the context of enumeration strategies. The branch step performs the division of the overall subgroup discovery task into several subtasks, e.g., finding all interesting descriptions that include a specific selector. These subtasks can be solved to a large extent independent from each other. The bounds step improves the performance of the algorithm by implementing an early detection that the subtasks will not (or unlikely) contribute to the final result. It is distinguished between *safe bounds*, which only prune



(a) Unpruned search space (b) Pruning at node b

Figure 3.4: The search space of a levelwise search without pruning on the left-hand side and with pruning at the single pattern b on the right-hand side.

parts of the search space that are guaranteed to not influence the final results, and *heuristic bounds*, for which this is only unlikely. Thus, exploiting safe bounds never changes the overall result, in contrast to heuristic bounds.

As an example for the importance of such pruning possibilities, consider Figure 3.4: Here, the full search tree for a small pattern mining problem is depicted on the left-hand side. Now assume that in a levelwise approach the pattern b was evaluated and it could be shown that none of its specializations are interesting. As shown on the right-hand side of the figure, this cuts the number of patterns to be evaluated on the remaining levels in half.

As it has been described in literature, implementing advanced pruning bounds can reduce the number of patterns to be evaluated and thus improve the runtime of algorithms by orders of magnitude. This can already be achieved by utilizing only safe bounds, which will be the focus of this work.

3.4.1 Anti-monotone Constraints

If the subgroup discovery task employs *anti-monotone* constraints, these can be easily exploited for pruning the search space. A constraint $C(P)$ is called anti-monotone if and only if for each pattern P it holds that

$$(C(P) = \mathit{false}) \wedge (S \supset P) \Rightarrow C(S) = \mathit{false},$$

that is, if a constraint is not satisfied in a pattern P, then it is also not fulfilled in any of its specializations S. Since search strategies generate new candidates from one (as in depth-first-search) or more (as in apriori) generalizations, these anti-monotone

constraints can be easily exploited by not generating candidates for patterns, which dissatisfy an anti-monotone constraint. Popular anti-monotone constraints include:

- *Maximum number of selectors in a subgroup description:* By definition each specialization has more selectors than its generalizations, thus the constraint is anti-monotone.

- *Minimum subgroup size:* Since with each additional describing selector the size of the covered instances decreases, this is anti-monotone.

- *Minimum number of positives:* This is analogous to the constraint before, but limited to the positive instances.

In contrast to anti-monotone constraints, monotone constraints, e.g., a maximum number of negative examples, are usually not exploited in subgroup discovery algorithms. This is due to the applied general-to-special enumeration order and due to lower performance gains, since in a depth-limited search most patterns do have significantly more specializations than generalizations.

3.4.2 Optimistic Estimate Pruning

For subgroup discovery tasks, which are not based on constraints, but on interestingness measures, other pruning options are available: Unfortunately, most measures are not anti-monotone, i.e., strictly decreasing for specializations, themselves. However, for many measures *optimistic estimate* bounds can be derived and exploited instead. Given a subgroup description P and an interestingness measure q an optimistic estimate $oe_q(P)$ is a function such that for each specialization $S \supset P$ the interestingness score is not greater than the value of the optimistic estimate function for the pattern P:

$$\forall S \supset P : q(S) \leq oe_q(P)$$

If the search task aims at finding only subgroup patterns above a certain interestingness threshold (e.g., a significance value), then the specializations of all subgroups with an optimistic estimate below this threshold can be skipped in the search: The optimistic estimate property already guarantees that the interestingness value of specializations will not be sufficient for the specialization to be included in the result set. Optimistic estimates can also be utilized for search tasks that do not employ a fixed quality threshold, but a top-k approach. In this case, the interestingness of the pattern with the k-th best quality found so far, that is, the subgroup with the worst score in the current result set, can be used as a dynamic minimum quality threshold. Therefore, it depends on the already evaluated pattern if a pattern can be pruned by using optimistic estimate bounds. As a consequence, the exact enumeration strategy and reordering of the search space can be crucial.

3.4.3 Optimistic Estimates for Common Binary Interestingness Measures

In contrast to the search strategy or the data structure, optimistic estimates depend strongly on the applied interestingness measure. Therefore, optimistic estimates bounds must in general be derived for each measure individually.

A simple method to compute these upper bounds is to bound the terms of the interestingness measures separately, exploiting the anti-monotonicity of instance counts. That is, the number of instances for a specialization is never greater than in its generalization. Consider for example the interestingness measure weighted relative accuracy, which is given by $q^1_{bin}(P) = i_P \cdot (\tau_P - \tau_\emptyset)$. Here, i_P is the instance count for the pattern P, τ_P is the target share in P, and $\tau_\emptyset$ is the target share in the overall population. Since the instance counts of specializations never exceed the instance count of P, and the highest possible target share is 1, the following optimistic estimate can be derived for this interestingness measure, see [260]:

$$oe_{q^1_{bin}}(P) = i_P \cdot (1 - \tau_\emptyset) = i_P \cdot (1 - \tau_\emptyset)$$

This estimate can be significantly improved using another generic approach: For an important class of interestingness measures optimistic estimates can be constructed in a straight forward way, that is, for *convex measures*. An interestingness measure is called convex if its value in a fixed dataset is computed by a convex two-dimensional function $q(i_P, p_P)$ over the overall number of instances i_P and number of positive instances p_P of the evaluated subgroup P. A two-dimensional function f is called convex if and only if $\forall x_1, x_2 \in \mathbb{R}^2, \lambda \in [0,1] : f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) \cdot (1-\lambda) f(x_2)$. This can be visualized such that the function value of each linear combination of two points x_1, x_2 is always on or below a straight line between the function values of these parameters. By these definitions, some of the most important interestingness measures have been shown to be convex, such as the chi-square measure, the information gain [196], the weighted relative accuracy and the category utility [274].

For convex interestingness measures $q(P) = q(i_P, p_P)$ it can be shown that an upper bound for the score of any specialization is given by $\max(q(p_P, p_P), q(i_p - p_P, 0))$. This describes that for any arbitrary pattern the best specialization is given by a refinement that covers either all positive or all negative instances. Using this approach, an improved optimistic estimate for the weighted relative accuracy q^1 can be derived [108, 274]:

$$oe_{q^1}(P) = \max(p_P \cdot (1 - \tau_\emptyset), (i_p - p_P) \cdot (0 - \tau_\emptyset) = p_P \cdot (1 - \tau_0)$$

In an interesting addition to this concept, bounds for (zero-diagonal) convex interestingness measures can be further improved, if *unavoidable* instances are identified, that is, instances that will be covered by all refinements of a subgroup, cf. [200].

As a minor contribution of this work, the best refinement of a subgroup can also be characterized for any function, for which the well known axioms postulated in [140], see Section 2.5.3.2, apply:

Theorem 1 *Assume that P is a subgroup with a binary target and a target share higher than the target share in the overall population ($\tau_P > \tau_\emptyset$), and q is any interestingness measure for which the well known axioms postulated in [140] (see Section 2.5.3.2) apply. Then, for any subset of the covered instances $r \subseteq sg(P)$ it holds that:*

$$q(r) \leq q(s^*), \text{ where } s^* = tp(P) = \{c \in sg(P) | T(c) = true\}$$

PROOF According to the third axiom ("$q(P)$ monotonically decreases in i_P if $\tau_p = \frac{k}{i_P}$ with a fixed constant k") it holds, that $q(P) \leq q(s^*)$. Here, the number of positive instances p_P in P is used as the constant k. This expresses that the interestingness of any subgroup P with positive quality is always lower or equal compared to the interestingness of the instance subset, which only contains the positive examples of P.

For arbitrary $r \subseteq sg(P)$, two cases are now to be considered: If $s^* \subseteq r$, then $q(r) \leq q(s^*)$ as shown above, since r contains the same positive instances as s^*, but more (or equal) negative instances. Otherwise it holds that $s^* \not\subseteq r$, and there exists a subset $r' = tp(r)$, which contains only the positive examples of r. Then, as above the third axiom implies that $q(r') \geq q(r)$. On the other hand $r' \subseteq s^*$ is true, as r' only covers positives examples, which are also covered by P. Therefore, the fourth axiom of [140] ("$q(P)$ monotonically increases in i_P for a fixed $\tau_P > \tau_\emptyset$.") can be applied: As $\tau_{r'} = \tau_{s^*} = 1$ it follows that $q(r') \leq q(s^*)$. Thus, it holds that $q(r) \leq q(r') \leq q(s^*)$, proving the theorem. ■

For interestingness measures following these axioms, the highest score for any subset of the subgroup's instances is always achieved by the subset that contains exactly all positive instances. As for convex functions, this can be used to determine an optimistic estimate in a simple way, that is, by computing the interestingness score for this subset.

Despite these results, the computation of optimistic estimates for more complex (non-classic) interestingness measure remains an open issue. In that direction, novel results regarding interestingness measures for numeric target concepts and generalization-aware interestingness measures will be presented in Chapter 4 and Chapter 6.

3.5 Algorithms

The following section provides an overview on algorithms, which have been proposed for supervised pattern mining in literature. This includes algorithms for descriptive tasks (such as subgroup discovery) as well as predictive tasks (such as rule learning algorithms for classification), since the tasks differ very little from an algorithmic point of view, as discussed before. The introduced categorization allows for describing the algorithms very succinct by referencing the respective enumeration strategy, data structure and pruning strategy. The used interestingness measures for the method will only be discussed if pruning strategies, which are specific for certain interestingness measures, are employed. This is, because one easily exchange the measure otherwise.

Subgroup discovery environments, such as *SubgroupMiner* [143, 144], *Orange*[1] or *Cortana*[2], which incorporate a variety of algorithms, will not be discussed here, since they only implement existing algorithms that are described independent of the system. We do include, however, the pioneering system *Explora* due to its historical importance in the field and since the implemented algorithms were innovative at their time. The overview here is also limited to algorithm for standard subgroup discovery with a binary target and a traditional interestingness measure. Further algorithms for numeric target concepts, exceptional model mining and subgroup discovery with generalization-aware interestingness measures are discussed as related work for the novel contributions in Chapter 4, Chapter 5, and Chapter 6. Additionally, previous algorithm for more specific scenarios are described separately in Section 3.6.

3.5.1 Explora

The *Explora* system [140] was one of the first software environments for subgroup discovery. It employed a variety of interestingness measures ("evaluation functions"). Supported enumeration orders included exhaustive as well as heuristic approaches, e.g., (heuristic) best-first-search and depth-first-search. Additionally, filters were utilized to focus the search on interesting parts of the search space. For the efficient computation of conjunctions of patterns, Explora did also exploit bitset-based data structures. Explora could be used to discover patterns with binary or numeric target concepts.

3.5.2 MIDOS

Wrobel [260] proposed the *MIDOS* algorithm for multi-relational subgroup discovery. The algorithm could be applied using a variety of basic search strategies such as depth-first-search, breadth-first-search or heuristic best-first-search. For performance improvements, it employed minimal support pruning as well as optimistic estimate pruning, which exploited an upper bound for the weighted relative accuracy interestingness measure. Specialized data structures are to the author's knowledge not described for this system. However, the system makes use of sample strategies for applicability in large scale domains.

3.5.3 The CN2 Family

CN2 [63, 62] is a classical rule learning algorithm for classification. It utilizes a beam-search method to identify the "best" rule according to a rule heuristic. The selection of a single rule is embedded in a sequential covering approach [193]. The algorithm *CN2-SD* [165] extends this approach by adding a weighted covering scheme, see Section 2.5.5.1. Since the focus of this line of research is more on the expressiveness of the rule set than on scalability, there are no especially efficient data structures specified for

[1] http://orange.biolab.si
[2] http://datamining.liacs.nl/cortana.html

these algorithms. Additionally, no additional pruning criteria for efficient rule identification have been described. The CN2-SD algorithm has also been extended by variants for subgroup discovery with multi-class targets [3] (called *CN2-MSD*) and relational subgroup discovery [169, 270] (called RSD).

3.5.4 OPUS

Webb introduced *OPUS* as a general algorithm for unordered search that follows a branch-and-bound philosophy [248]. It has also been used for supervised rule learning. OPUS is proposed in two different variations: While $OPUS^S$ focuses on satisfying search, $OPUS^O$ is used for optimization search, that is, to find the best node in the search space according to an evaluation function. It describes the use of "optimistic pruning" in combination with a best-first strategy to explore the search space.

No specialized data structure is described. In the context of classification rule learning, Webb also introduced optimistic estimates for specific "preference functions" for OPUS, i.e., the maximum consistent preference function and the Laplace preference function [256]. The algorithm is currently the center-piece of the commercial MAGNUM OPUS software for association discovery. In the original publication, a depth-first-search with reordering according to the optimistic bound is recognized as an alternative enumeration order for settings, in which memory is critical.

3.5.5 SD

The algorithm *SD* [92] also uses a beam-search strategy for enumeration. Regarding pruning, it utilizes simple support pruning. Additionally, irrelevant subgroups are never added to the beam. No specialized data structures are specified. The description of the algorithm emphasizes the interactive nature of the addressed task. Therefore, the algorithm is embedded in an environment featuring different interestingness measures and visualizations.

3.5.6 STUCCO

For mining contrast sets, Bay and Pazzani proposed the *STUCCO* algorithm [32, 33], which searches for interesting patterns in a breadth-first strategy. For pruning, support thresholds for different groups are used as well as an optimistic estimate bound for the χ^2 measure. Additionally, specializations that have the same support as generalizations are pruned, as they are considered as not interesting.

3.5.7 Apriori-C and Apriori-SD

Jovanovski and Lavrač adapted the Apriori algorithm for itemset mining to obtain classification rules [131]. In addition to standard support pruning, also another pruning criterion is used. It filters specializations, for which a generalization already is considered as interesting based on support and confidence thresholds. However, neither strict optimistic estimates nor specialized data structures are described. The classification

centered algorithm *Apriori-C* has later been transferred to subgroup discovery in an algorithm called *Apriori-SD* [167]. The changes mainly focus on a different interestingness measure and the introduction of a new covering scheme, i.e., weighted covering, to avoid redundancy.

3.5.8 Apriori SMP

Morishita and Sese [196] introduced the *AprioriSMP* algorithm for association rule mining with a fixed conclusion. They combine the levelwise apriori enumeration order with "statistical metric pruning", which is equivalent to optimistic estimates in the terminology of this work. For their algorithm, they provide upper bounds for the interestingness measures chi-square, entropy gain, gini-index, and the correlation coefficient measuring the correlation between the description and the target. For AprioriSMP, no specialized data structure has been proposed.

3.5.9 Harmony

The *Harmony* algorithm for mining classification rules [246] performs a depth-first-search with different reordering options, e.g., reordering depending on confidence, entropy or a correlation measure. It also supports pruning based on optimistic estimates by combining the support threshold with the confidence function, which is used as interestingness measure.

3.5.10 CorClass and CG

The *CorClass* algorithm [273] was proposed by Zimmermann and de Raedt for the mining of class association rules, that is, association rules with the fixed class attribute of the dataset as a consequence. The algorithm emphasizes the usefulness of optimistic estimate pruning. To maximize the benefits of the bounds, they propose a best-first enumeration strategy and utilize general upper bounds for convex interestingness measures. No specialized data structures are mentioned.

The algorithm was refined into the *CG*-algorithm [274] with similar computational properties regarding pruning- and search strategy. In this approach, the supervised pattern mining task is not only regarded as a stand-alone task as in subgroup discovery, but also as a core step for other areas such as classification or (conceptual) clustering.

3.5.11 Corrmine

The *Corrmine* algorithm for correlated itemset mining [200] exploits additional bounds (*4-bounds*) by propagating itemsets, which are unavoidable, in addition to traditional optimistic estimates. This is embedded in an algorithm based on *ECLAT* [269] that employs a depth-first-search and vertical data representations.

3.5.12 SDIGA and MESDIF

Del Jesus et al. propose the genetic algorithm *SDIGA* for a subgroup discovery task in the context of a case study in the domain of marketing [68]. They utilize a fuzzy discretization of numeric variables as well as (internal) disjunctions in rule description, which is claimed to still provide an "explanatory and understandable form that can be used by the expert" [68]. The fitness of individuals is assigned by measuring the number of individuals in a population, which are dominated by a pattern with respect to the three applied objectives (interestingness measures), confidence, support and "original support". Another closely related algorithm presented in the same domain is *MESDIF* [39].

3.5.13 Algorithm of Cerf

Cerf et al. propose an algorithm for mining *class association rules* [57]. Their algorithm employs a breadth-first enumeration strategy. Regarding pruning, only basic pruning based on the support of the pattern is used. Additionally, specializations of patterns, which are considered as interesting, are not explored. The search is repeated for each value of the target attribute, carrying over already discovered interesting patterns. No specialized data structure is described.

3.5.14 CCCS

The *CCCS*-algorithm tries to find rules for associative classification in imbalanced data efficiently [12]. The search is performed in a depth-first manner. Interestingly, the search space is build explicitly by adding transactions (instances) one-by-one into an enumeration tree. This resembles a feature of FP-trees, in which also non-occurring patterns are left out from the start. However, in this approach a pass over the dataset is required for each candidate. Additionally, no pruning schemes are exploited to enhance the performance.

3.5.15 CMAR

For improved classification based on associations, the CMAR algorithm [179] was introduced. For the efficient mining of rules, it employs a variation of FP-trees as data structure in a depth-first-search. No optimistic estimate pruning is described.

3.5.16 SD-Map

The *SD-Map* algorithm [22] transferred the FP-tree data structure to the subgroup discovery setting. The algorithm employs a strict depth-first-search enumeration strategy. For pruning, only constraints for the subgroup size and the number of positive instances are utilized, but no optimistic estimate pruning.

3.5.17 DpSubgroup

The algorithm *DpSubgroup* [100, 107] improves this approach by adding support for tight optimistic estimates to the FP-tree approach. Furthermore, the enumeration strategy is slightly adapted to a depth-first-search with forward checking and branch reordering in order to maximize the benefit of the bounds.

3.5.18 DDPMine

Cheng et al. also proposed a mining algorithm for supervised patterns, which is based on FP-trees. The respective algorithm is called *DDPMine* [60]. It uses a standard depth-first-search in combination with optimistic estimate pruning by using an upper bound for the single interestingness measure information gain.

3.5.19 Overview of Algorithms

The presented algorithms for the standard setting are summarized in tabular form, see Table 3.1. For each algorithm, the year of publication, the enumeration order, the applied data structure, and the research subfield are provided.

3.6 Algorithms for Special Cases

The following section discusses algorithms for specialized variations of subgroup discovery. In particular, algorithms for mining relevant (non-redundant) subgroups, algorithms for the online-discretization of numeric describing variables and adaptations for the mining of very large datasets are reviewed.

3.6.1 Algorithms for Relevant Subgroups

To avoid redundancy in the result of a subgroup discovery task, the notion of *relevant subgroups* [95, 96] has been defined. Mining only relevant subgroups avoids multiple subgroups with a similar coverage in the result set. In particular, a subgroup P is removed from the result if another subgroup covers all positive instances of P, but only a subset of its negative instances, see Section 2.5.5.2 for a more detailed description.

From a computational point of view, this relevancy criterion requires additional checks. However, it can also be exploited to reduce the search space and thus to speed up the search. In that direction, several approaches have been proposed:

Garriga et al. observed [95] that relevant subgroups are always *closed* (as defined in frequent itemset mining) with respect to the covered positive instances. As a consequence, the efficient *LCM*-algorithm [240] for closed itemset mining can be utilized for subgroup discovery in a two step approach: First, all patterns that are closed with respect to the positive instances are identified by LCM. This algorithm reduces the search space significantly by partitioning the set of candidate subgroups into equivalence classes with respect to their coverage in the dataset. In the second step, these candidates are evaluated according to the applied interestingness measure.

Table 3.1: An overview of supervised pattern mining algorithms, categorized with respect to enumeration strategy, pruning strategy, data structure, and original research field.

Algorithm	Year	Enum. strategy	Pruning	Data structure	Research field
Apriori-C	2001	levelwise	support+special	-	classification rules
Apriori-SD	2004	levelwise	support+special	-	subgroup discovery
Apriori-SMP	2000	levelwise	opt. est.	-	ass. rules w/ fixed concl.
CCCS	2006	DFS	-	-	associative classification
Cerf	2008	breadth-first	support	-	associative classification
CMAR	2001	DFS	-	FP-trees	associative classification
CN2	1989	Beam	-	-	classification rules
CN2-SD	2002	Beam	-	-	subgroup discovery
CorrClass	2009	Best-first-opt	opt. est.	-	several
Corrmine	2009	Depth-first	opt. est. + 4-bounds	depth-first	correlated itemset min.
DDPMine	2008	DFS	opt. est.	FP-trees	discriminative patterns
DpSubgroup	2008	DFS+FC	opt. est.	FP-trees	subgroup discovery
Explora	1996	DFS,BFS,Best-First	redundancy	bitsets	subgroup discovery
HARMONY	2005	DFS+Reorder	opt. est.	-	classification rules
MIDOS	1997	DFS,Best-First	opt. Est	-	multi-relational
OPUS	1995	Best-First-Opt	opt. est.	-	general search
SD	2003	Beam	support+relevance	-	subgroup discovery
SDIGA	2007	Genetic	-	-	subgroup discovery
SD-Map	2006	DFS	support	FP-trees	subgroup discovery
STUCCO	1999	BFS,(DFS)	support	-	contrast set mining

An extension of this approach by Boley and Grosskreutz focuses on the identification of minimum representatives for each equivalence class [42]. The resulting algorithm *imr* is reported to match the performance of the standard algorithm *DpSubgroup* (cf. Section 3.5.17) in datasets, in which the compression through equivalence classes is high.

In previous work, the author of this work performed a comparison of data structures for the mining of irrelevant subgroups [171, 177]. It turned out that vertical data structures are better suited for this task than FP-tree-based structures, since they allow for performing efficient relevance checks directly in the search algorithm. The proposed algorithm *BSD* combines depth-first-search, a bitset-based data representation, and optimistic estimate pruning with (optional) efficient relevancy checks. However, the algorithm can not theoretically guarantee to obtain the best relevant subgroups, although they are almost always found in practice. This problem is caused by the fact that promising subgroups, which are first added to the top-k result set, are shown to be irrelevant later in the search process and thus have to be removed from the result. Grosskreutz and Paurat solved this problem by employing an iterative deepening depth-first-search as enumeration strategy [104].

Recently, Grosskreutz introduced another algorithm, which is able to enumerate *all* relevant subgroups in polynomial time with respect to the result size [101]. It utilizes a more precise characterization of candidate subgroup than the closed-on-the positives property. This is exploited by a new relevance check based on the direct generalizations of a subgroup in order to apply more effective pruning in the search algorithm. The algorithm uses breadth-first-search as enumeration strategy.

3.6.2 Numeric Selectors

As discussed in Section 2.3.2, numeric attributes in the search space can be incorporated in standard mining algorithms through discretization in a pre-processing step. Since this involves loss of information, online-discretization, i.e., identifying the best intervals for numeric attributes within the subgroup mining algorithm, has been postulated. However, this can increase the search space of subgroup discovery substantially. In order to handle this problem efficiently, specialized subgroup discovery algorithms have been proposed for this setting:

The *MergeSD* algorithm [106] of Grosskreutz and Rueping is designed for exhaustive search. It exploits that for the Kloesgen measures $q^a_{Kl}(P) = {i_P}^a \cdot (\tau_P - \tau_\emptyset)$ an optimistic estimate for a subgroup description $P \wedge A \in]t_l, t_r[$ can be calculated based on statistical information regarding the patterns $P \wedge A \in]t_l, t'[$ and $P \wedge A \in]t', t_r[$ for any numeric attribute A and $t_l \leq t' \leq t_r$. Information on the derived bounds is stored in a specialized data structure called *boundTable*. The pruning scheme is integrated in a depth-first-search with look-ahead.

Mampaey et al. analyzed the refinement step for greedy algorithms such as beam-search (cf. Section 3.2.4) with respect to online discretization of numeric describing attributes [189]. They propose a method that allows for finding the best interval of a numeric attribute that is added to the current subgroup description in linear time of the number potential cutpoints, in contrast to the quadratic time required by the

trivial approach. This is accomplished by focusing on the subgroups, which are on the *convex hull* of candidate subgroups in the ROC space. These can be determined efficiently bottom-up from the convex hulls of the basic half-open intervals using *Minkowski differences*.

In the field of association rule mining, related approaches have been discussed: Fukuda et al. investigated numeric attributes in the rule condition of *optimized association rules* [84]. These aim at identifying intervals for numeric attributes, which imply a maximum confidence for the rule. For efficient mining, the authors exploit *convex hull trees* in order to compute the best interval for one attribute with respect to the rule confidence in linear time. This problem setting has been extended in [214, 46] to include disjunctions of intervals.

3.6.3 Large Scale Mining Adaptations

Although sophisticated mining algorithms improve the scalability of subgroup discovery by orders of magnitude, the proposed methods may still be intractable for very large datasets. Besides optimizing existing algorithms and decreasing the complexity of the search, e.g., by selecting a subset of relevant attributes or by setting a limit for the maximum number of selectors in a subgroup description, there are two different possibilities to handle specifically very large datasets: *sampling* and *parallelization*.

3.6.3.1 Sampling

Sampling describes the (random) selection of a subset of the current dataset. Characteristics of the complete dataset can be estimated from the characteristics of the sample. These can be determined more efficiently due to the smaller size of the sample.

A first line of research focuses on deriving *guarantees* about the quality of results derived from a specific, randomly drawn sample. This can be modified to determine the size of the sample set, which is required to guarantee that the computed quality for a pattern exceeds a given threshold. In that direction, Schaeffer and Wrobel proposed the *GSS* algorithm (Generic Sequential Sampling) [218, 220]. It iteratively draws new instances from the database, until it can be guaranteed with a certain confidence that the current k best hypotheses are the overall best hypotheses in the dataset. This technique was explored for a variety of interestingness measures. The original algorithm requires all hypotheses (subgroups) to be stored in memory. An improved version of the algorithm, *LCM-GSS*, only requires fixed memory, but this comes at the cost of larger sample sizes [219].

These algorithms aim at achieving samples of the dataset, which approximate the distributional properties of the overall dataset as close as possible. Other approaches aim at the selection of representative instances instead of random subsets. In that direction, several strategies have been proposed in the related field of association rule mining, e.g., the FAST algorithm [58] or the ϵ-approximation algorithm [49]. Regarding subgroup discovery, Cano et al. investigated different sample strategies, e.g., by employing a nearest neighbor prediction approach [56].

A few other approaches use the sampling procedure to alter the distributions in order to obtain subsets of instances, which are particularly interesting or useful. In that direction, Scholz presents a method to incorporate background knowledge into subgroup discovery [222, 221]. It utilizes directed sampling to prevent the rediscovery of prior knowledge. This is done by re-weighting instance for the sampling according to previously known patterns. In another study by Cano et al., a stratified selection algorithm is explored, which increases the presence of the minority class in imbalanced data, that is, a dataset, in which one value of the target concept occurs much more frequently than the other [55].

Traditional sampling methods select subsets of instances in the dataset. Recently, some novel approaches have been proposed that focus on sampling *subsets of patterns* instead. In that direction, Boley and Grosskreutz proposed a method, which uses a *Markov chain Monte Carlo* method (MCMC) to sample local patterns such as association rules or subgroups [41]. To overcome the scalability issues of MCMC an alternative two-step approach has been proposed [43]: In a first step a few data instances are drawn, then local patterns are sampled based on the drawn instances.

3.6.3.2 Parallelization

A second possibility to handle very large datasets is to assign subtasks to different computation units, which can operate at the same time. Two kinds of parallelization techniques can be distinguished, see for example [162]: *Multicore computing* describes that several processing units of one machine are involved in the computation. In this scenario, the same data is available for all cores in the *shared memory*. In contrast, *distributed computing* describes data processing on different machines, which are only connected by a network and use separate memory stores.

Parallel subgroup mining with shared memory is relatively simple to implement and to adapt by many discovery algorithms. The branches of the search tree can be assigned to different computation units, which explore their respective part of the search tree independent of each other. The only required communication is concerned with the propagation of the minimum interestingness score of the result set for optimistic estimate pruning. Additionally, computation units can reassign parts of their assigned task to idle units if these completed their task earlier. This provides for simple load-balancing. This kind of parallelization has been implemented successfully in subgroup discovery systems, e.g., in the MIDOS system [260] or for the algorithm BSD [177].

Subgroup mining in a distributed scenario is much more complicated: Communication costs, e.g., for propagating the requirements for optimistic estimate pruning, have to be strongly considered in this case. Wurst and Scholz investigated a setting, in which the data is distributed among several nodes (computation units) [264]. For this setting, they propose an algorithm, in which each subgroup pattern is assigned to a node. This node decides if pruning can be applied for this pattern. If not, then *count polling* is initiated, that is, the counting of positives and negatives in all nodes. After results are collected, it is determined if specializations of the current rule are potentially interesting. In this case, specializations are generated and assigned for evaluation to other computational nodes in

the network. Pruned subgroups and changes in the result set are always communicated to all nodes, leading to communication costs that increase linearly with number of evaluated subgroups and computation nodes. Another distributed subgroup mining algorithm that especially emphasizes the data privacy for each node is proposed by Grosskreutz et al. in [103].

In the related field of association rule discovery, several approaches have been presented for distributed mining, see also [217]. Different approaches cover the distribution of the data to different nodes, the distribution of candidates to different nodes, or a combination of both [6, 61, 268, 224]. Several methods focus on distributed computations for the FP-tree data structure [51, 123]. Adapting these methods seems to be a promising direction for future subgroup discovery research.

3.7 Summary

This chapter provided an overview on efficient supervised pattern mining algorithms from literature. This overview included not only algorithms that use the terminology of subgroups discovery, but also algorithms from closely related fields. It presented methods for different algorithmic components in detail, that is, for the enumeration strategy, for the data representation, and for the pruning strategy. Algorithms could then concisely be described in terms of these components. Beside the standard supervised pattern mining setting, also some important variations have been outlined, such as the discovery of relevant subgroups and subgroup discovery with numeric describing attributes. Regarding large scale applications, methods for sampling and parallelization were summarized.

4 Algorithms for Numeric Target Concepts[1]

Practical applications often involve numeric attributes with a continuous domain as the target property of interest. Although it is possible to transform this problem setting to the standard nominal one by using discretization techniques in a pre-processing step [78, 157], this yields a possibly crucial loss of information. Therefore, over the years a variety of interestingness measures has been proposed, which are directly based on the distribution of a numeric target attribute. These have been summarized in Section 2.5.3.3.

For example, a potentially interesting subgroup using target discretization could be expressed as *"While in the general population only 6% of the people are older than 80, in the subgroup described by* xy *it is 12%"*. In contrast, using the distribution of the numeric targets concept directly with a mean-based interestingness measure, a subgroup is described like *"While in the general population the mean age is* 56 *years, in the subgroup described by* xy *it is* 62*"*. Until now, the numeric target setting is less intensively studied than the traditional case with binary targets. Therefore, this chapter investigates this scenario in detail and proposes several improvements for the efficient exhaustive mining of subgroups with numeric target settings.

As discussed in Chapter 3, algorithms are characterized by the employed *enumeration strategy*, by the used *data structure*, and by the applied *pruning strategy*, which allows to omit parts of the search space safely. Since the search space does not change, if a numeric target concept is used instead of a binary one, enumeration strategies can be directly transferred from the traditional setting with binary targets. Instead, this work focuses on optimistic estimate bounds for efficient pruning of the search space and on the adaptation of data structures specialized for this problem setting. In more detail, the contributions of this chapter are the following:

1. For a large variety of interestingness measures, novel optimistic estimate bounds are introduced. These are used for pruning the search space: If it can be proven that no subset of a subgroup's instances induces a sufficiently high interestingness score to be included in the result set of subgroups, then no specialization of this subgroup is included in the result. These specialization can be skipped in the search process without influencing the optimality of the result. We distinguish between bounds with closed-form representations and bounds based on ordering, which allow the computation of optimistic estimate by checking few candidate subsets of instances.

[1] A minor part of this chapter, i.e., the introduction of the *SD-Map** algorithm, has previously been published as [15]: Martin Atzmueller and Florian Lemmerich. Fast Subgroup Discovery for Continuous Target Concepts. In *Proceedings of the 18th International Symposium on Foundations of Intelligent Systems (ISMIS)*, 2009.

2. It is shown how the most important data representations, i.e., FP-trees and bitset data structures can be adapted to the problem setting of numeric target concepts. It is shown that the application of different data structures allows for the application of different optimistic estimate bounds.

3. Two novel algorithms called *SD-Map** and *NumBSD* implement these approaches and allow for efficient mining in practical applications. Both algorithms apply optimistic estimate pruning, but utilize different core data structures and thus possibly different pruning bounds. Both outperform simple approaches by orders of magnitudes.

4. For the proposed bounds and algorithms an extensive experimental evaluation on publicly available datasets is provided. These show the benefits of the novel approaches and aim also to identify recommendations for choosing the best mining algorithm for specific tasks.

The rest of the chapter is structured as follows: It starts by summarizing previous approaches to subgroup discovery with numeric target concepts in Section 4.1. After that, novel optimistic estimate bounds for these measures are introduced in Section 4.2. Then, modifications of data representations for the numeric setting are discussed in Section 4.3. Next, Section 4.4 presents two practical algorithms for efficient subgroup discovery with numeric targets. Section 4.5 provides an evaluation of the presented approaches. Computational properties of the different interestingness measures are summarized in Section 4.6, before we conclude in Section 4.7.

4.1 Related Work

Generally, subgroup discovery with numeric target concepts can be performed by applying discretization on the target concept. In doing so, the task is transformed into a standard subgroup discovery task with a binary target concept. A discretization method, which was specifically designed for numeric targets in subgroup discovery, is *TargetCluster* [195]. It uses a scoring of clustering solutions to find appropriate intervals for the target concept. Nonetheless, discretizing the target concept leads to a loss of information, see also Section 2.5.3.3.

Subgroup discovery with numeric target concepts without discretization has been introduced in the pioneering system Explora [140]. It applied a variety of enumeration strategies for subgroup discovery. Regarding numeric attributes, it was able to identify "mean patterns", that is, subgroups with a significantly deviating mean value in the numeric target attribute in comparison to the total population. An exemplary case study for this system using mean patterns is provided in [138]. For Explora, no optimistic estimates or data structures, which are specific for the mining of subgroups with numeric target concepts, are described.

Later, Aumann and Lindell [27, 28] discussed this problem setting in the context of association rules, described as *quantitative association rules*. For the discovery of desired rules, they apply a three-stage process: First, all frequent patterns in the search space

are mined by a standard frequent itemset mining algorithm, i.e., Apriori. In a second step, an interestingness value of these patterns is computed based on the deviation of either the mean or the variance of the target. Sub-rules, which are contained in more general rules are then filtered out in a third step. As a result of this multi-stage process, pruning is only based on the support of the patterns. In contrast to this method, the approach described in this work focuses on the direct mining of interesting patterns in one step. This allows for the application of advanced pruning techniques that can substantially improve the mining performance.

Webb [249] utilizes a best-first approach to efficiently find association rules with a specific numeric variable in the rule head. In this approach, optimistic estimate pruning is applied for the utilized interestingness measure. The presented upper bounds are however limited to a single interestingness measure ("impact rules", which is equivalent to q^1_{mean} in the terminology of this paper). In contrast, this chapter presents upper bounds for a wide variety of interestingness measures. In addition, adaptations to data structures are discussed, which allow for the efficient computation of interestingness measures and their upper bounds.

Jorge et al. present an algorithm for the discovery of distribution rules called CAREN-DR [130]. These can be considered as subgroups for a numeric target concept that use a Kolmogorov-Smirnov-based interestingness measure. The algorithm follows a depth-first enumeration order. It utilizes (unordered) bitsets to speedup the evaluation of patterns. Pruning is limited to support pruning and does not take advantage of optimistic estimates.

Pieters et al. discuss different interestingness measures [210, 209] for subgroup discovery with numeric targets. However, they focus on the performance of different measures and do not investigate techniques for efficient exhaustive mining.

4.2 Optimistic Estimates

Optimistic estimates are upper bounds for the interestingness score of all specializations of a subgroup. They are used to speedup the search process for subgroup discovery: If the optimistic estimate of a subgroup is lower than the interestingness value required in the result set, then the specializations of this subgroup can be excluded from the search, see also Section 3.4.2. As research for binary target concepts has shown, this can reduce the number of candidate subgroups in discovery algorithms by orders of magnitudes. Nonetheless, for subgroup discovery with numeric target concepts, optimistic estimate bounds have only received limited attention in literature so far.

This section thoroughly analyzes optimistic estimate pruning for subgroup discovery with numeric target concepts for a large variety of interestingness measures. It shows first why the problem of deriving optimistic estimates is more difficult for numeric targets than in the traditional binary target setting. Then, optimistic estimates in closed-form expressions are derived for the discussed interestingness measures. These upper bounds can also be determined by algorithms that use a FP-tree like data structures. Next, a property of interestingness measures is introduced that allows intuitive derivation

Table 4.1: Derivation of the best subset of example subgroup P. The subset with the best interestingness score for a measure is printed in bold. It can be seen that the subset differs for different interestingness measures.

Target values of a subset $r \subseteq sg(P)$	$q^1_{mean}(r)$	$q^{0.5}_{mean}(r)$	$q^0_{mean}(r)$
$\{100\}$	$1 \cdot (100 - 50) = 50$	$\sqrt{1} \cdot (100 - 50) = 50$	$\mathbf{100 - 50 = 50}$
$\{100, 75\}$	$2 \cdot (87.5 - 50) = 75$	$\sqrt{\mathbf{2}} \cdot (\mathbf{87.5 - 50}) \approx \mathbf{53}$	$87.5 - 50 = 37.5$
$\{100, 75, 53\}$	$\mathbf{3} \cdot (\mathbf{76 - 50}) = \mathbf{78}$	$\sqrt{3} \cdot (76 - 50) \approx 45$	$76 - 50 = 26$
$\{100, 75, 53, 12\}$	$4 \cdot (60 - 50) = 40$	$\sqrt{4} \cdot (60 - 50) = 20$	$60 - 50 = 10$

of upper bounds. In addition to these bounds, a novel approach provides a series of increasingly tight upper bounds, which can be computed by using only a part of the instances covered by a subgroup.

4.2.1 Differences to the Binary Setting

An optimistic estimate bound of a subgroup is given by the subset of instances, which implies the highest score according to the used interestingness measure. In the traditional setting with a binary target concept, it is easy to determine this subset, as long as the interestingness measures follow the common axioms of Piatetsky-Shapiro and Major & Mangano, see 2.5.3.2: The best subset is the set of all instances that have a positive target concept, see Theorem 1. In contrast to that, the continuous case is more challenging. Here, the best refinement of a subgroup depends on the applied interestingness measure, as demonstrated in the following example:

Example 1 Assume a dataset with an average target value of $\mu_\emptyset = 50$. We determine the best refinement for a subgroup P according to mean-based interestingness measures $q^a_{mean}(P) = {i_P}^a \cdot (\mu_P - \mu_\emptyset)$, with $a = 0, a = 0.5$, and $a = 1$. In this example P covers 4 instances: $sg(P) = \{c_1, c_2, c_3, c_4\}$. The target values for these instances are $T(c_1) = 100$, $T(c_2) = 75$, $T(c_3) = 53$ and $T(c_4) = 12$. Then, for the interestingness measure *impact* q^1_{mean} the subset with the best interestingness score contains the three cases with values 100, 75, and 53, see Table 4.1. On the other hand, for the *mean test* interestingness measure $q^{0.5}_{mean}$, it contains two cases with the values 100 and 75. For the *average* interestingness measure q^0_{mean}, the best subset contains only the instance with value 100. □

Since a simple and effective way to derive optimistic estimate bounds is to compute or estimate the interestingness value of the best refinement, the task of deriving upper bounds for all specializations is significantly more challenging for a numeric target concept. Another reason is that in the binary case the complete distribution of the target concept in a subgroup is precisely specified by just two numbers (the numbers of positive and negative examples), but it is more complex to describe in the numeric case.

4.2.2 Optimistic Estimates with Closed Form Expressions

Next, novel optimistic estimates are derived for the interestingness measures that have been discussed in Section 2.5.3.3. The estimates presented below have a closed-form expression that uses a limited amount of statistics derived from the subgroup. In particular, the bounds for a single subgroup are computable in a distributed single-pass algorithm. Thus, these statistics are also computable in efficient FP-tree data structures, as will be shown in Chapter 5, which analyzes applicability issues of this data structure in general.

4.2.2.1 Mean-based Interestingness Measures

Theorem 2 *As described in [249], an optimistic estimate for the impact interestingness measure* $q^1_{mean}(P) = i_P \cdot (\mu_P - \mu_\emptyset)$ *is for any subgroup* P *given by:*

$$oe^1_{mean}(P) = \sum_{c \in P : T(c) > \mu_\emptyset} (T(c) - \mu_\emptyset).$$

This estimate is tight.

Proof We reformulate this interestingness measure:

$$\begin{aligned} q^1_{mean}(P) &= i_P \cdot (\mu_P - \mu_\emptyset) \\ &= i_P \cdot (\frac{\sum_{c \in sg(P)} T(c)}{i_P} - \frac{i_P \cdot \mu_\emptyset}{i_P}) \\ &= \sum_{c \in sg(P)} T(c) - i_P \cdot \mu_\emptyset \\ &= \sum_{c \in sg(P)} (T(c) - \mu_\emptyset) \end{aligned}$$

For all subsets $r \subseteq sg(P)$, this sum reaches its maximum for the instance set r^*, which contains all cases with larger target values than the average of the population. For this set of instances, the summand in the above formula contains all positive summands, but no negatives ones. The interestingness score for r^* is given by $q_1(r^*) = oe^1_{mean}(P)$ using the above formula. As no other subset of $sg(P)$ leads to a higher interestingness, $oe^1_{mean}(P)$ is an upper bound for the interestingness score of all specializations of P and an optimistic estimate. Since for any given subgroup the estimate is reached by one of its subsets, the estimate is tight. ∎

Interestingly, the tight optimistic estimate for the binary case presented in [108], i.e., $p_P \cdot (1 - \tau_\emptyset)$, can be seen as special case of this formula, considering $T(c) = 1$ for true target values and $T(c) = 0$ for false target values:

$$oe^1_{mean}(P) = \sum_{c \in sg(P), T(c) > \mu_\emptyset} (T(c) - \mu_\emptyset) = \sum_{c \in sg(P), T(c)=1} (1 - \tau_\emptyset) = p_P \cdot (1 - \tau_\emptyset).$$

This optimistic estimate bound can easily be extended to the other generic mean-based interestingness measures q^a_{mean}:

Theorem 3 *$oe^1_{mean}(P)$ is an optimistic estimate for any generic mean-based interestingness measure $q^a_{mean}(P) = {i_P}^a \cdot (\mu_P - \mu_\emptyset)$ with arbitrary $a \in [0,1]$.*

PROOF For any refinement $r \subseteq sg(P)$ with $q^a_{mean}(r) \geq 0$ and any $a \in [0,1]$, it holds:

$$\begin{aligned} q^a_{mean}(r) &= |r|^a(\mu_r - \mu_\emptyset) \\ &\leq |r|^1(\mu_r - \mu_\emptyset) \\ &= q^1_{mean}(r) \leq oe^1_{mean}(P) \end{aligned}$$ ■

Example 2 Revisiting Example 1, we consider the subgroup P_1 that has four instances with target values $T(c_1) = 100$, $T(c_2) = 75$, $T(c_3) = 53$ and $T(c_4) = 12$ in a dataset with an overall target mean of 50. Then, the optimistic estimate oe^1_{mean} sums over all instances with a target value greater than 50, that is c_1, c_2 and c_3: $oe^1_{mean}(P_1) = (100-50)+(75-50)+(53-50) = 78$. The value of this optimistic estimate is independent from the parameter a of the chosen interestingness measure. □

Theorem 4 *An alternative optimistic estimate bound for $q^a_{mean}(P) = {i_P}^a \cdot (\mu_P - \mu_\emptyset)$ with arbitrary $a \in [0,1]$ is given by:*

$$\overline{oe}^a_{mean}(P) = {\widetilde{p}_P}^{\ a} \cdot (T^{max}_P - \mu_\emptyset),$$

where $\widetilde{p}_P = |\{c \in sg(P) | T(c) > \mu_\emptyset\}|$ is the number of instances in the subgroup with a target value higher than the population mean of the target and T^{max}_P is the maximum target value in the subgroup.

PROOF It is proven first that no instance with a target value lower than $\mu_\emptyset$ is part of the best refinement: Consider any subset $r \subseteq sg(P)$. Let $r^+ = \{i \in r | T(c) > \mu_\emptyset\}$ the set of all instances in r, which have a target value higher than the mean of the population and $r^- = \{i \in r | T(c) \leq \mu_\emptyset\}$ the complement of this set, so $r = r^+ \cup r^-$. Then, the interestingness score according to any q^a_{mean} is always equal or higher, if all instances of r^- are removed from the subgroup: So it is needed to show that:

$$\begin{aligned} q^a_{mean}(r^+) &\geq q^a_{mean}(r) \\ |r^+|^a(\mu_{r^+} - \mu_\emptyset) &\geq |r|^a(\mu_r - \mu_\emptyset) \end{aligned}$$

Similar to the proof of Theorem 2 this can be transformed as follows:

$$\begin{aligned} |r^+|^a \frac{\sum_{i \in r^+}(T(c) - \mu_\emptyset)}{|r^+|} &\geq |r|^a \frac{\sum_{i \in r}(T(c) - \mu_\emptyset)}{|r|} \\ |r^+|^a \frac{\sum_{i \in r^+}(T(c) - \mu_\emptyset)}{|r^+|} &\geq (|r^+| + |r^-|)^a \frac{\sum_{i \in r^+}(T(c) - \mu_\emptyset) + \sum_{i \in r^-}(T(c) - \mu_\emptyset)}{|r^+| + |r^-|} \end{aligned}$$

For shorter notation, we define $S^+ := \sum_{i \in r^+}(T(c) - \mu_\emptyset)$ and $S^- := \sum_{i \in r^-}(T(c) - \mu_\emptyset)$. Due to the construction of r^+ and r^- it holds that $S^+ \geq 0 \geq S^-$. Therefore:

$$\begin{aligned}
|r^+|^a \frac{S^+}{|r^+|} &\geq (|r^+| + |r^-|)^a \frac{S^+ + S^-}{|r^+| + |r^-|} \\
(|r^+| + |r^-|)|r^+|^a S^+ &\geq |r^+|(|r^+| + |r^-|)^a (S^+ + S^-) \\
(|r^+| + |r^-|)|r^+|^a S^+ &\geq |r^+|(|r^+| + |r^-|)^a S^+ + |r^+|(|r^+| + |r^-|)^a S^-
\end{aligned}$$

Since $S^- \leq 0$, it holds that $((|r^+| + |r^-|)^a S^-) \leq 0$. Thus, the above inequality is satisfied if

$$\begin{aligned}
(|r^+| + |r^-|)|r^+|^a S^+ &\geq |r^+|(|r^+| + |r^-|)^a S^+ \\
(|r^+| + |r^-|)^{1-a} S^+ &\geq |r^+|^{1-a} S^+ \\
(|r^+| + |r^-|)^{1-a} &\geq |r^+|^{1-a} \\
|r^+| + |r^-| &\geq |r^+|
\end{aligned}$$

This is always true. Therefore, the interestingness value of an instanceset r according to any q^a_{mean} never decreases, if all instances are removed that have a target value less than the mean target value in the overall population. Consequently, there is always a best refinement of a subgroup that does not contain any instance with target value equal or lower than the population mean. The largest possible number of instances of such a refinement is given by $\widetilde{p}_P$. Trivially, the mean value of this refinement never exceeds the largest value of the original subgroup. Thus $\widetilde{p}_P \cdot (T^{max}_P - \mu_\emptyset)$ is an optimistic estimate for P. ■

Example 3 Continuing the above example again, P_1 has four instances with target values $T(c_1) = 100$, $T(c_2) = 75$, $T(c_3) = 53$ and $T(c_4) = 12$ in a dataset with an overall target mean of $\mu_\emptyset = 50$. Then, the optimistic estimate $\overline{oe}^a_{mean}(P_1)$ is computed as follows: There are three instances with a target value greater than the population mean target value, c_1, c_2 and c_3, thus $\widetilde{p}_{P_1} = 3$. The maximum target value in the subgroup is $T^{max}_{P_1} = 100$. Therefore, $\overline{oe}^a_{mean}(P) = \widetilde{p}_P{}^a \cdot (T^{max}_P - \mu_\emptyset) = 3^a \cdot (100 - 50) = 3^a \cdot 50$. Depending on the generality parameter a of the applied interestingness measures this results in $\overline{oe}^1_{mean}(P_1) = 150$ for $a = 1$, in $\overline{oe}^{0.5}_{mean}(P) \approx 86.6$ for $a = 0.5$ or $\overline{oe}^1_{mean}(P) = 50$ for $a = 0$.

Comparing these bounds with the bound $oe^1_{mean}(P_1) = 78$, which was derived in Example 2, it is evident that no bound is superior in every case: For a high value of a such as $a = 1$, $oe^1_{mean}(P)$ is tighter than $\overline{oe}^a_{mean}(P)$, for a low value of a such as $a = 0$ $\overline{oe}^a_{mean}(P)$ is tighter. □

As demonstrated in the examples, the optimistic estimates oe^1_{mean} and $\overline{oe}^a_{mean}(P)$ are both not tight for arbitrary parameters a of the interestingness measure: For the

subgroup P_1 the best refinement r^* using the mean test measure $q^{0.5}_{mean}$ contains the first two instances and has the interestingness score $\sqrt{2} \cdot 37.5 \approx 53$. The upper bound computed by this equation is $oe^1_{mean}(P) = (100 - 50) + (75 - 50) + (53 - 50) = 78$. The alternative bound gives an estimate of $\overline{oe}^{0.5}_{mean}(P) = \sqrt{4}(100 - 50) = 100$.

In general, the bound oe^1_{mean} can be very weak for smaller a. In the worst case, the optimistic estimate exceeds the interestingness value of the best refinement by factor ${i_P}^{1-a}$. It is more precise for high values of a, that is, for interestingness measures, which favor large subgroups, even if they only have a limited deviation of the target mean value. However, the estimate is still useful also for lower settings of a if the value distribution of the target variable is widely spread. In contrast, the bound $\overline{oe}^a_{mean}$ provides good estimates, if the target values are not too dispersed, but is less tight, if the target values are spread out. In particular, it is also useful for pruning small subgroups in settings with a small value for the parameter a. For $a = 0$, the estimate is tight since in this case a best refinement contains only the single instance with the highest target value.

Most importantly, both estimates $oe^1_{mean}(P)$ and $\overline{oe}^a_{mean}(P)$ are not exclusive, but can easily be combined: One can compute the values for both bounds and use the tighter one, i.e., the one with the smaller value, to apply optimistic estimate pruning.

Symmetric Mean-based Measures It is easy to extend the above presented optimistic estimates to symmetric variants:

Theorem 5 *An optimistic estimate for the generic symmetric mean evaluation functions $q^a_{abs}(P) = {i_P}^a \cdot |\mu_P - \mu_\emptyset|$ is given by:*

$$oe^1_{abs}(P) = \max \left(\sum_{c \in sg(P), T(c) < \mu_\emptyset} (\mu_\emptyset - T(c)), \sum_{c \in sg(P), T(c) > \mu_\emptyset} (T(c) - \mu_\emptyset) \right).$$

This bound is tight for q^1_{abs}. Another bound is:

$$\overline{oe}^a_{abs}(P) = \max \left({\widetilde{p}_P}^a \cdot (T^{max}_P - \mu_\emptyset), {\widetilde{n}_P}^a \cdot (\mu_\emptyset - T^{min}_P) \right),$$

where T^{max}_P, T^{min}_P are the maximum and minimum target values in the subgroup P. $\widetilde{p}_P = |\{c \in sg(P) | T(c) > \mu_\emptyset\}|$, $\widetilde{n}_P = |\{c \in sg(P) | T(c) < \mu_\emptyset\}|$ are the numbers of instances in the subgroup with a target value greater (respectively smaller) than the population mean of target values.

Proof This follows straightforward from the last two theorems since for the interestingness value of any subset $r \subseteq sg(P)$ it holds that $q^a_{abs}(r) = |r|^a |\mu_r - \mu_\emptyset| = max\,(|r|^a(\mu_r - \mu_\emptyset), -(|r|^a(\mu_r - \mu_\emptyset))$. So in essence, one can just compute an upper bound for $q^a_{mean}(r)$ and $-q^a_{mean}(r)$ separately and use the maximum of both bounds as a bound for q^a_{abs}.

Example 4 As in the previous examples, P_1 has four instances with target values $T(c_1) = 100$, $T(c_2) = 75$, $T(c_3) = 53$ and $T(c_4) = 0$ in a dataset with an overall target mean of 50. Then, the optimistic estimates for symmetric mean-based measures are computed as follows: $oe^1_{abs}(P_1) = \max((50-0),(100-50)+(75-50)+(53-50)) = \max(50,78) = 78$. The second bound depends on the parameter a of the interestingness measure: $\overline{oe}^a_{abs}(P_1) = \max(1^a \cdot (50-12), 3^a \cdot (100-50)) = 3^a \cdot 50$. □

Theorem 6 *For the variance reduction* $q_{vr}(P) = \frac{i_P}{i_\emptyset - i_P} \cdot (\mu_P - \mu_\emptyset)^2$, *two optimistic estimates are given by:*

$$oe_{vr}(P) = \max\left(\frac{i_P}{i_\emptyset - i_P} \cdot (T_P^{max} - \mu_\emptyset)^2, \frac{i_P}{i_\emptyset - i_P} \cdot (T_P^{min} - \mu_\emptyset)^2\right),$$

$$\overline{oe}_{vr}(P) = \max\left(\frac{\left(\sum_{c \in sg(P), T(c) > \mu_\emptyset} (T(c) - \mu_\emptyset)\right)^2}{i_\emptyset - 1}, \frac{\left(\sum_{c \in sg(P), T(c) < \mu_\emptyset} (T(c) - \mu_\emptyset)\right)^2}{i_\emptyset - 1}\right),$$

where T_P^{max} *(*T_P^{min}*) is the maximum (minimum) target value in the subgroup* P.

Proof The proof of the first estimate is straight forward: The first factor is strictly increasing with i_P and thus reaches its maximum for all specializations of P at i_P, since all specializations cover at most as much instances as P. The maximum difference in the second factor occurs if the mean value in the specialization is either maximal or minimal. Trivially, the maximum or minimum for each specialization is in the interval $[T_P^{max}, T_P^{min}]$.

Regarding the second estimate, it holds for any subset $r \subseteq sg(P)$ with positive interestingness score that:

$$\begin{aligned} q_{vr}(r) &= \frac{|r|}{i_\emptyset - |r|}(\mu_r - \mu_\emptyset)^2 \\ &= \frac{|r|^2}{(i_\emptyset - |r|)|r|}(\mu_r - \mu_\emptyset)^2 \\ &= \frac{|r|^2}{(i_\emptyset - |r|)|r|}\left(\frac{\sum_{i \in r}(T(c) - \mu_\emptyset)}{|r|}\right)^2 \\ &= \frac{1}{(i_\emptyset - |r|)|r|}\left(\sum_{i \in r}(T(c) - \mu_\emptyset)\right)^2 \end{aligned}$$

$(i_\emptyset - |r|)|r|$ gets minimized for $|r| = 1$. The squared sum is maximized if the sum is either maximized or minimized. That is accomplished by either including only positive or only negative summands, that is, only instances with a target value higher than the

population mean target value or with a lower target value respectively. This leads to the presented optimistic estimate. ■

Example 5 As in the previous examples, P_1 has four instances with target values $T(c_1) = 100$, $T(c_2) = 75$, $T(c_3) = 53$ and $T(c_4) = 12$ in a dataset with an overall target mean of $\mu_\emptyset = 50$. Additionally, it is assumed that the overall population consists of 10 instances. Then, the upper bounds according to the above theorem are then given by: $oe_{vr}(P) = \max\left(\frac{4}{10-4} \cdot (100-50)^2, \frac{4}{10-4} \cdot (12-50)^2\right) = \frac{4}{6} \cdot 50^2 = 1666.\overline{6}$ and $\overline{oe}_{vr}(P_1) = \max\left(\frac{1}{9} \cdot ((100-50)^2 + (75-50)^2 + (53-50)^2), \frac{1}{9}(12-50)^2\right) = \frac{1}{9} \cdot (50^2 + 25^2 + 3^2) \approx 348.2$. In this case, the second bound is substantially tighter (lower) and is therefore used for pruning. □

4.2.2.2 Median-based Measures

For median-based interestingness measures, the practical use of a direct estimate, which can be computed in a parallel single pass algorithm, is doubtful, since the median itself cannot be computed in such a way. Nonetheless, a very simple, but loose estimate can be specified:

Theorem 7 *For the generic median-based measure $q^a_{med}(P) = {i_P}^a \cdot (med_P - med_\emptyset)$ an optimistic estimate is given by:*

$$oe^a_{med}(P) = {i_P}^a \cdot (T^{max}_P - med_\emptyset)$$

Proof The proof of this theorem is straight forward: The maximum median in any refinement cannot exceed the maximum occurring value in the subgroup, and the size of a refinement cannot exceed the size of the subgroup. ■

In contrast to the generic mean-based functions, the best refinement for median-based function can contain values with target values lower than the population mean as demonstrated in the following example:

Example 6 Consider a subgroup P_2 with target values $\{3, 2, 2, 0, -1, -1\}$ in a dataset with an overall median target value of 1. Then, for the interestingness measure q^1_{med} the best subset of instances contains the first 5 instances, as shown in Table 4.2. The optimistic estimate according to Theorem 7 is $6^1 \cdot (3-1) = 12$.

4.2.2.3 (Full) Distribution-based Measures

Theorem 8 *An optimistic estimate for the Kolmogorov-Smirnov interestingness measure $q_{ks}(P) = \sqrt{\frac{i_P \cdot i_{\neg P}}{i_\emptyset}}\Delta_{(P,\neg P)}$ for any subgroup P with $i_P < \frac{i_\emptyset}{2}$ is given by*

$$oe_{ks}(P) = \sqrt{\frac{i_P(i_\emptyset - i_P)}{i_\emptyset}}.$$

Table 4.2: Determine the best subset of covered instances example subgroup P_2. The best subset is printed in bold.

Target values for subset of $sg(P_2)$	q^1_{med}
$\{3\}$	$1 \cdot (3-1) = 1$
$\{3,2\}$	$2 \cdot (2.5-1) = 1$
$\{3,2,2\}$	$3 \cdot (2-1) = 3$
$\{3,2,2,0\}$	$4 \cdot (2-1) = 4$
$\{\mathbf{3},\mathbf{2},\mathbf{2},\mathbf{0},\mathbf{-1}\}$	$\mathbf{5} \cdot (\mathbf{2}-\mathbf{1}) = \mathbf{5}$
$\{3,2,2,0,-1,-1\}$	$6 \cdot (1-1) = 0$

□

PROOF The interestingness value of q_{KS} is given by $\sqrt{\frac{i_P \cdot i_{\neg P}}{i_\emptyset}} \cdot \Delta_{(P,\neg P)}$. The test statistic $\Delta_{(P,\neg P)}$ is computed as the supremum of differences in the empirical distribution functions of P and its complement. Since the range of the empirical distribution function is $[0,1]$ the supremum of the difference $\Delta_{(P,\neg P)} \leq 1$. For a fixed population, the left term $\sqrt{\frac{i_P \cdot i_{\neg P}}{i_\emptyset}}$ is only dependent on the number of instances covered by the subgroup. We determine the maximum of this term for any refinement r of $sg(P)$: If $i_P \leq \frac{i_\emptyset}{2}$ the term is monotone. In particular, $|r|(i_\emptyset - |r|) < i_P(i_\emptyset - i_P)$. Otherwise a maximum is reached at $i_P = \frac{i_\emptyset}{2}$. However, this is an overall bound for the interestingness measure and is therefore not useful for pruning. ■

Given a minimum interestingness value required by the result set, the optimistic estimate derived from this theorem implies a minimum number of instances, which must at least be covered by any subgroup that has a sufficiently high interestingness score. In contrast to the other introduced optimistic estimates, this bound does not take the distribution of the target variable in the subgroup into account. Therefore, it is to be expected that this bound is less tight than other optimistic estimates.

Example 7 Consider again the subgroup P_1 that covers 4 instances of the 10 instances in the overall dataset. The optimistic estimate for the Kolmogorov-Smirnov interestingness measure is then given by: $\sqrt{\frac{4 \cdot (10-4)}{10}} = \sqrt{2.4}$. The target values of the instances covered by the subgroup do not influence the optimistic estimate. □

4.2.2.4 Rank-based Measures

Theorem 9 *Two optimistic estimate bounds for the Mann-Whitney interestingness measure* $q_{mw'}(P) = \sqrt{\frac{i_P}{i_{\neg P}}} \cdot (\frac{\mathcal{R}_P}{i_P} - \frac{i_\emptyset+1}{2})$ *are given by:*

$$oe^1_{mw'}(P) = \sum_{c \in sg(P), \rho(c) > \frac{i_\emptyset+1}{2}} \left(\rho(c) - \frac{i_\emptyset + 1}{2} \right)$$

$$\overline{oe}_{mw'}(P) = \sqrt{i_P^+}(\rho_P^{max} - \frac{i_\emptyset + 1}{2}),$$

where $\rho(c)$ *is the rank of instance* c *in order of the target values,* ρ_P^{max} *is the maximum rank in the subgroup* P *and* $i_P^+ = |\{c \in sg(P) \,|\, \rho(c) > \frac{i_\emptyset+1}{2}\}|$ *is the number of instances in the subgroup with a rank higher than the populations rank mean.*

Proof It holds for all refinements r with a positive interestingness value, that

$$q_{mw'}(P) = \sqrt{\frac{i_P}{i_{\neg P}}}(\frac{\mathcal{R}}{i_P} - \frac{i_\emptyset + 1}{2}) \leq \sqrt{i_P}(\frac{\mathcal{R}}{i_P} - \frac{i_\emptyset + 1}{2})$$

Since $\frac{\mathcal{R}}{i_P}$ is the mean of the ranks within the subgroup and $\frac{i_\emptyset+1}{2}$ is the mean of the ranks in the overall population, the right part of this equation is equal to the mean test function $q^{0.5}_{mean}$ if the target values are given by the ranks. Thus, we can transfer the upper bounds from Theorem 3. However, these bounds are substantially less tight for this interestingness measure due to the initial estimation. ■

Example 8 Consider again the subgroup P_1 in a dataset of 10 instances and assume that the 4 instances covered by the subgroup have the ranks $\rho(c_1) = 2$, $\rho(c_2) = 3$, $\rho(c_3) = 6$ and $\rho(c_4) = 8$. That is, the instance c_1 has the second highest target value in the dataset, c_2 has the third highest target value, c_3 has the sixth highest target value and c_4 has the eighth highest target value. Then, the optimistic estimates according to the above theorem are given by:

$$oe^1_{mw'}(P_1) = \sum_{c \in sg(P), \rho(c) > \frac{i_\emptyset+1}{2}} \left(\rho(c) - \frac{i_\emptyset + 1}{2} \right) = (6 - \frac{11}{2}) + (8 - \frac{11}{2}) = 3$$

$$\overline{oe}_{mw'}(P_1) = \sqrt{i_P^+}(\rho_P^{max} - \frac{i_\emptyset + 1}{2}) = \sqrt{2} \cdot (8 - \frac{11}{2}) = \sqrt{2} \cdot 2.5)$$

The first bound is tighter in this example and is therefore used as the overall bound for optimistic estimate pruning. In doing so, specializations of the subgroup P_1 have not to be considered in the subgroup discovery algorithm, if the result set requires a minimum interestingness score of more than 3. □

Theorem 10 *If there are no ties in the ranks, then an optimistic estimate for the area-under-the-curve interestingness measure* $q_{auc}(P) = \frac{\mathcal{R}_{\neg P} - \frac{i_{\neg P} \cdot (i_{\neg P}+1)}{2}}{i_P \cdot i_{\neg P}}$ *is given by:*

$$oe_{auc}(P) = \frac{i_\emptyset - \rho_P^{min}}{i_\emptyset - 1},$$

where ρ_P^{min} *is the minimum rank for an instance in* P*.*

PROOF Due to the construction of the area-under-the-curve the best subset of $sg(P)$ contains only one instance, which is the one with the lowest rank ρ_{min}. The interestingness of this refinement S is $q_{auc}(S) = \frac{\mathcal{R}_{\neg S} - \frac{i_{\neg S}(i_{\neg S}+1)}{2}}{i_S i_{\neg S}} = \frac{\frac{(i_\emptyset+1)i_\emptyset}{2} - \rho_P^{min} - \frac{i_\emptyset(i_\emptyset-1)}{2}}{1 \cdot (i_\emptyset - 1)} = \frac{i_\emptyset - \rho_P^{min}}{i_\emptyset - 1}$ ■

Example 9 Consider again the subgroup P_1 that covers instances with following ranks with respect to the target concept: $\rho(c_1) = 2$, $\rho(c_2) = 3$, $\rho(c_3) = 6$ and $\rho(c_4) = 8$ in a dataset that consists of 10 instances. Then, the above theorem provides the optimistic estimate: $oe_{auc}(P) = \frac{10-2}{9} = \frac{8}{9}$. □

4.2.3 Ordering-based Bounds

In order to apply pruning of the search tree more frequently in a search algorithm, a tight optimistic estimate is desired. However, the estimates presented above only provide loose estimates in many cases. As an alternative, one can check all subsets of the subgroups instances for the one with the best interestingness score in order to obtain a tighter upper bound. Unfortunately, this is not directly feasible, as there is an exponential number of possible refinements for each subgroup. In the following, two useful properties of interestingness measures are defined that simplify the identification of the best refinement and thus the induction of optimistic estimate bounds.

4.2.3.1 One-pass Estimates by Ordering

Definition 2 An interestingness measure q is *one-pass estimable by ordering*, if it holds for any subgroup P and any refinement $r \subseteq sg(P)$, that:

$$q(r) \leq \max\left(q(s_1^{desc}), \ldots, q(s_{i_P}^{desc})\right),$$

where s_j^{desc} is the set of instances that consists of the j instances of $sg(P)$ with the highest target values. □

This property expresses that for these measures the interestingness, which is implied by a set of instances, always increases, if one of the instances is exchanged with another instance that has a greater target value. It reduces the number of candidates for the best refinement of a subgroup from 2^{i_P} to i_P.

This motivates the following approach for subgroup discovery with numeric target concepts using such interestingness measures: In a preprocessing step, the instances in

the database are sorted in descending order with respect to their target values. Whenever a subgroup is evaluated, instances are added one by one to the subgroup, starting with the highest value. After each addition, the interestingness of the instance set is evaluated. The maximum of these interestingness values is used as a tight optimistic estimate. In doing so, only a single pass over each subgroup is required. In most cases, the gain through additional pruning options by far exceeds the additional effort for computing the bounds, as shown in the evaluation in Section 4.6.

Theorem 11 *The interestingness measures* $q^a_{mean}(P) = {i_P}^a \cdot (\mu_P - \mu_\emptyset)$ *are one-pass estimable by ordering.*

PROOF We consider any refinement $r \subseteq sg(P)$ and compare it to the instance set $r^* = s^{desc}_{|r|}$. This is the subset with the same number of covered instances as r, but the highest target values contained in $sg(P)$: Then, $|r^*| = |r|$ and $\mu_{r^*} \geq \mu_r$. It follows that according to a mean-based interestingness measure the refinement r^* is at least as interesting as r:

$$\begin{aligned} q^a_{mean}(r) &= |r|^a \cdot (\mu_r - \mu_\emptyset) \\ &\leq |r^*|^a \cdot (\mu_{r^*} - \mu_\emptyset) \\ &= q^a_{mean}(r^*) \end{aligned}$$

■

Example 10 In a dataset with a mean target value of 50, the subgroup P_3 covers the instances $sg(P) = \{c_2, c_3, c_4, c_5\}$. The target values for these instances are $T(c_2) = 75$, $T(c_3) = 53$, $T(c_4) = 12$, $T(c_5) = 100$. These instances are ordered according to their target value in order to generate the subsets s^{desc}_j: $s^{desc}_1 = \{c_5\}, s^{desc}_2 = \{c_5, c_2\}, s^{desc}_3 = \{c_5, c_2, c_3\}, s^{desc}_4 = \{c_5, c_2, c_3, c_4\}$. For each of these subsets the score of the interestingness measure is computed. In this example, the mean test measure $q^{0.5}_{mean}(P) = \sqrt{i_P} \cdot (\mu_P - \mu_\emptyset)$ is used. The resulting scores are:

$$\begin{aligned} q^{0.5}_{mean}(s^{desc}_1) &= \sqrt{1} \cdot (100 - 50) = 50 \\ q^{0.5}_{mean}(s^{desc}_2) &= \sqrt{2} \cdot (87.5 - 50) \approx 53 \\ q^{0.5}_{mean}(s^{desc}_3) &= \sqrt{3} \cdot (76 - 50) \approx 45 \\ q^{0.5}_{mean}(s^{desc}_3) &= \sqrt{4} \cdot (60 - 50) = 20 \end{aligned}$$

The maximum of these interestingness score, in this case $q^{0.5}_{mean}(s^{desc}_2) \approx 53$, then defines an optimistic estimate bound for the subgroup. This bound is tighter than the estimates with closed form expressions presented in Theorem 2 and Theorem 4:

$$\begin{aligned} oe^1_{mean}(P_3) &= (100 - 50) + (75 - 50) + (53 - 50) = 78 \\ \overline{oe}^a_{mean}(P_3) &= \sqrt{3} \cdot (100 - 50) \approx 85.6 \end{aligned}$$

□

Theorem 12 *The z-score interestingness measure* $q_z(P) = \frac{\sqrt{i_P} \cdot (\mu_P - \mu_\emptyset)}{\sigma_\emptyset}$ *is one-pass estimable by ordering.*

PROOF This follows directly from Theorem 11 since $q_z(P)$ is order equivalent with $q^{0.5}_{mean}$. ■

Theorem 13 *The median-based interestingness measures* $q^a_{med}(P) = {i_P}^a \cdot (med_P - med_\emptyset)$ *are one-pass estimable by ordering.*

PROOF Analogously to Theorem 11, replacing the mean by the median. ■

Theorem 14 *The rank-based interestingness measures* $q_{mw}(P)$ *and* $-q_{auc}(P)$ *are one-pass estimable by ordering.*

PROOF For any refinement r with $|r| = j$, the sum of ranks gets maximized (or minimized, depending on the ordering of ranking) for s^{desc}_j, therefore $q(s^{desc}_j) \geq q(r)$ for these interestingness measures. ■

4.2.3.2 Two-pass Estimates by Ordering

For symmetric interestingness measures, which indicate increases as well as decreases of the target concept, the most interesting refinement potentially covers only instances with low target values. Thus, symmetric measures are never *one-pass estimable*. However, a broadened definition of this property is applicable in this case:

Definition 3 An interestingness measure q is *two-pass estimable by ordering*, if it holds for any subgroup P and any refinement $r \subseteq sg(P)$ that:

$$q(r) \leq \max\left(q(s^{desc}_1), \ldots, q(s^{desc}_{i_P}),\ q(s^{asc}_1), \ldots, q(s^{asc}_{i_P})\right),$$

where s^{desc}_j is the set of instances that consists of the j instances of $sg(P)$ with the highest target values and s^{asc}_j is the set of j instances of $sg(P)$ that have the lowest target values. □

Trivially, any interestingness measure, which is one-pass estimable by ordering, is also two-pass estimable by ordering. For interestingness measures with this property, the best subset of instances is found by traversing the current subgroup twice — once in descending and once in ascending order of the target values. In both passes, instances are added one-by-one to the current set of instances. As before, after each addition the interestingness score of the instance set is computed. The overall optimistic estimate is then given by the maximum of all those scores. In doing so, $2 \cdot i_P$ subsets of the instances of the current subgroup are considered as candidates to find its best refinement.

Several symmetric interestingness measures, which identify increases as well as decreases of the target value, fulfill this property:

Theorem 15 *The symmetric mean based measures* $q^a_{abs}(P) = {i_p}^a \cdot |\mu_P - \mu_\emptyset|$ *are two-pass estimable by ordering.*

PROOF Consider any refinement $r \subseteq sg(P)$. Without loss of generality, let $j = |r|$ be the number of instances covered by r. If $\mu_r \geq \mu_\emptyset$ then $q_a(s_j^{desc}) \geq q_a(r)$ in analogy to the proof of Theorem 11. Otherwise, $\mu_r < \mu_\emptyset$ and we can conclude that $q_a(s_j^{asc}) \geq q_a(r)$ since $|s_j^{asc}| = |r|$ and $\mu_{s_i^{asc}} \leq \mu_r$ and therefore $|\mu_{s_j^{asc}} - \mu_\emptyset| \geq |\mu_r - \mu_\emptyset|$. ■

Example 11 As in the last example, let the subgroup P_3 cover 4 instances with target values $T(c_2) = 75$, $T(c_3) = 53$, $T(c_4) = 12$, $T(c_5) = 100$. As interestingness measure, any symmetric mean based measure is used. To compute the optimistic estimate for this interestingness measure, the subsets s_j^{desc} are formed as in the previous example. Additionally, the subsets s_j^{asc} are constructed using the ascending order of the target values: $s_1^{asc} = \{c_4\}, s_2^{asc} = \{c_4, c_3\}, s_3^{asc} = \{c_4, c_3, c_2\}, s_4^{asc} = \{c_4, c_3, c_2, c_5\}$ For each of these subset the score of the interestingness measure is computed and the maximum of these interestingness scores determines the optimistic estimate. □

Theorem 16 *The interestingness measure variance reduction* $q_{vr}(P) = \frac{i_P}{i_\emptyset - i_P} \cdot (\mu_P - \mu_\emptyset)^2$ *is two-pass estimable by ordering.*

PROOF For any refinement $r \subseteq sg(P)$ that covers $j = |r|$ instances, it holds that $(\mu_P - \mu_\emptyset)^2 \leq \max\left((\mu_{s_j^{desc}} - \mu_\emptyset)^2, (\mu_{s_j^{asc}} - \mu_\emptyset)^2\right)$. Thus, $q_{vr}(r) \leq \max\left(q_{vr}(s_j^{desc}), q_{vr}(s_j^{asc})\right)$. ■

4.2.3.3 Interestingness Measures not Estimable by Ordering

As shown before, many interestingness measures for numeric target concepts are one- or two pass estimable by ordering. Although these properties seem intuitive, they do not apply to all interestingness measures:

Theorem 17 *The generic variance interestingness measure* $q_{sd}^a(P) = {i_P}^a \cdot (\sigma_P - \sigma_\emptyset)$ *is* not *one-pass or two-pass estimable by order.*

PROOF Assume that the standard deviation in a dataset is overall $\sigma_\emptyset = 1$ and the subgroup P_4 covers 3 instance with target values $T(c_1) = 10$, $T(c_2) = 0$, and $T(c_3) = -10$. The subset r^* with highest standard deviation and therefore the highest score according to $q_{sd}^0(P)$ then consists of the two instances c_1 and c_3. The mean value for this subset is $\mu_{r^*} = 0$ and its variance is $sd_{r^*}^2 = \frac{(10-0)^2+(0-(-10))^2}{2-1} = 200$. The interestingness score for this subset is then $q_{sd}^0(r^*) = 2^0 \cdot 200 - 1 = 199$, which is greater than the interestingness score of all subsets s_j^{desc} or s_j^{asc}. ■

Theorem 18 *The t-score interestingness measure* $q_t(P) = \frac{\sqrt{i_P} \cdot (\mu_P - \mu_\emptyset)}{\sigma_P}$ *is* not *one-pass or two-pass estimable by order.*

PROOF Consider a dataset with $\mu_\emptyset = 0$ and a subgroup P within the dataset that contains four instances $i_1, \ldots, i_4$ with the target values $T(i_1) = 20$, $T(i_2) = 10$, $T(i_3) = 10 + \epsilon$, $T(i_4) = 0, \epsilon \ll 0.1$. Then the best refinement r^* contains the instances i_2 and i_3. The interestingness of r^* approaches infinity if ϵ approaches 0. Thus, $q_t(r^*) > max(q_t(s_i^{desc}), q_t(s_i^{asc}))$ for any i given a small ϵ. This contradicts the definition of one-pass and two-pass estimability by ordering. ■

The novel algorithm *NumBSD*, which is introduced later in this chapter, incorporates the presented ordering-based estimates in an efficient algorithm. However, these bounds can not be computed using arbitrary data structures, i.e., they can not be used in combination with FP-trees, cf. also Section 5.3.

4.2.4 Fast Bounds using Limited Information

This section presents a novel method to speed up the computation process by applying a sequence of less precise upper bounds, which are computed during the evaluation of a single subgroup. The bounds are determined by using only the refinements s_j^{desc} of P, that is, the j instances in P with the highest target values. The section focuses exclusively on the probably most popular family of interestingness measures in this setting, the generic mean-based interestingness measures q^a_{mean}.

The main idea is as following: As before, instances are incorporated one-by-one in descending order of the target values into the evaluation of a subgroup. After each instance, the interestingness for the set s_j^{desc} of the already incorporated j instances is computed. By definition, the maximum of all interestingness values $q(s_j^{desc})$ is an optimistic estimate for interestingness measures, which are one-pass estimable by ordering. We now consider a certain point in time during this pass over the dataset, at which the first n instances have already been processed. At this point, it is guaranteed that the target values for all following instances in the subgroup are not greater than the target value of the current case. This fact is used to determine an upper bound for all the interestingness values $q(s_j^{desc})$, which have not yet been computed: For all instances, which have not been visited yet, it is assumed that the target value of these instances is equal to the target value of the instance, which was added last. By assuming greater target values, the computed interestingness value according to any generic mean-based interestingness measure q^a_{mean} always increases, since the interestingness measure is one-pass estimable. Thus, we compute the maximum value that is obtained by adding any number of instances with the current target value. This forms an upper bound for the remaining interestingness values $q(s_j^{desc})$, $j > n$. If this weaker upper bound already indicates that none of the refinements of the current subgroup (nor the current subgroup itself) will be added to the result set, then we can skip the rest of the evaluation of this subgroup. Formally, this approach can be based on the following formal theorem:

Theorem 19 *For a subgroup P, let $s_n^{desc} \subseteq sg(P)$ be the n instances with the highest target values. Furthermore, let $\sigma = \sum_{c \in s_n^{desc}} T(c)$ be the sum of target values for s_n^{desc} and θ the lowest target value for the instances in s_n^{desc}, that is, the n-th highest target value in $sg(P)$. Then, an optimistic estimate for generic mean-based interestingness measures $q^a_{mean}(P) = {i_P}^a \cdot (\mu_P - \mu_\emptyset)(P)$ is given by:*

$$oe^a_{fast}(P) = \max\left(q^a_{mean}(s_1^{desc}), \ldots, q^a_{mean}(s_n^{desc}), oe^a_{remaining}(P)\right),$$

$$oe^a_{remaining}(P, n) = {i_P}^a \cdot \left(\frac{\sigma + (i_P - n) \cdot \theta}{i_P} - \mu_\emptyset\right)$$

PROOF The theorem is proven by showing that for each subset $r \subseteq sg(P)$ the interestingness score is not higher than the provided optimistic estimate: $q^a_{mean}(P) \leq oe^a_{fast}(P)$. Since $q^a_{mean}(P)$ is one-pass estimable, for each r the interestingness is lower than the interestingness of the subset, which has the same number of instances, but covers the instances with the highest target values: $q^a_{mean}(r) \leq q^a_{mean}(s^{desc}_{|r|})$. Thus, the theorem holds for all r with $|r| \leq n$.

For all refinements r with $|r| > n$, it remains to show that $q^a_{mean}(s^{desc}_{|r|}) \leq oe^a_{fast}(P)$. To do so, the interestingness of s^{desc}_j for any $j = n + x$ with $x > 0, x \leq x_{max}, x_{max} = i_P - n$ is estimated. Let c_k be the instance with the k-highest target value.

$$\begin{aligned} q^a_{mean}(s^{desc}_j)(P) &= j^a \cdot \left(\frac{\sum_{k=1}^{j} T(c_k)}{j} - \mu_\emptyset \right) \\ &= (n+x)^a \cdot \left(\frac{\sum_{k=1}^{n} T(c_k) + \sum_{k=n+1}^{x} T(c_k)}{n+x} - \mu_\emptyset \right) \\ &\leq (n+x)^a \cdot \left(\frac{\sigma + x \cdot \theta}{n+x} - \mu_\emptyset \right) := f^a(x) \end{aligned}$$

The inequality utilizes that the target values are ordered descending and it therefore holds for $k > n$ that $T(c_k) \leq T(c_n) = \theta$. The function $f^a(x)$ describes an upper bound for the interestingness of the instance set s^{desc}_j that consists of x instances more than the last evaluated instance set s^{desc}_n. Unfortunately, the size j of the instance set with the maximum interestingness and the respective x value is not known. However, an upper bound for the interestingness of $q(s^{desc}_j)(P)$ is given by the maximum value of $f^a(x)$. For this family of functions, it can be shown by computing the first and second derivative that the maximum value is reached either at $x = 0$ or at $x = x_{max} = i_P - n$. The formal proof is provided in the appendix of this chapter. This means that the maximum upper bound is reached, if either none or all remaining instances are added to the last evaluated instance with an assumed target value of θ. For $x = 0$, the value of the function f^a is equal to the interestingness score of the instance set s^{desc}_n: $f^a(0) = (n+0)^a \cdot \left(\frac{\sigma + 0 \cdot \theta}{n+0} - \mu_\emptyset \right) = n^a \cdot (\mu_{s^{desc}_n} - \mu_P) = q^a_{mean}(s^{desc}_n)$.

As a consequence it holds for all $r \subseteq sg(P)$ with $|r| = j > n$ that

$$\begin{aligned} q^a_{mean}(r) &\leq q(s^{desc}_j) \\ &\leq f^a(x) \\ &\leq \max\left(f^a(0), f^a(i_P - n)\right) \\ &= \max\left(q^a_{mean}(s^{desc}_n), {i_P}^a \cdot \left(\frac{\sigma + (i_P - n) \cdot \theta}{i_P} - \mu_\emptyset \right) \right) \\ &\leq oe^a_{fast}(P) \end{aligned}$$

This proves the theorem. ∎

This theorem provides an upper bound for the interestingness score of all specializations of a subgroup and also for the interestingness of the subgroup itself. It is based only on a part of the instances of a subgroup, that is, the ones with the highest target values. Thus, the above estimates can be checked during an iteration over the subgroup instances even before this iteration has finished. If after only a few instances the computed upper bound indicates that the subgroup itself and all its specializations do not have a sufficient interestingness for the result set, the evaluation of the subgroup can be stopped. In doing so the majority of the subgroup has not to be considered, thus speeding up the subgroup discovery process. To the author's knowledge, this is the first approach that uses optimistic estimates, which are based only on a part of a subgroup's instances.

Example 12 Again, the subgroup P_1 is considered, which covers 4 instances with target values $T(c_1) = 100$, $T(c_2) = 75$, $T(c_3) = 53$, $T(c_4) = 12$. The mean target value in the overall dataset is $\mu_\emptyset = 50$. Additionally, it is assumed that a score of at least 150 is currently required by the result set using the impact interestingness measure q^1_{mean}.

For the evaluation of P_1, instances are added one-by-one, starting with the instance c_1, since it has the highest target value. The interestingness value of a subgroup that covers only this single instance is $q(s_1^{desc}) = 1 \cdot (100 - 50) = 50$. To compute the optimistic estimate $oe^a_{fast}(P_1)$ after the first instance, additionally the value of $oe^a_{remaining}(P_1)$ is required. This is determined as $i_{P_1}{}^a \cdot \left(\frac{\sigma+(i_{P_1}-n)\cdot\theta}{i_{P_1}} - \mu_\emptyset\right) = 4^{\,1} \cdot \left(\frac{100+(4-1)\cdot 100}{4} - 50\right) = 200$. Since 200 exceeds the minimum required interestingness of 150, the evaluation of P_1 continues.

Next, the instance c_2 is added, as it has the second highest target value. The corresponding interestingness value is $q(s_2^{desc}) = 2 \cdot (87.5 - 50) = 75$. Additionally, the value of $oe^a_{remaining}(P_1)$ is updated: $oe^a_{remaining}(P_1) = 4^1 \cdot \left(\frac{175+(4-2)\cdot 75}{4} - 50\right) = 125$. It follows that $oe^a_{fast}(P_1) = \max\left(q(s_1^{desc}), q(s_2^{desc}), oe^a_{remaining}(P)\right) = \max(50, 75, 125) = 125$ is an optimistic estimate for the subgroup P: P itself and all of its specializations are guaranteed to not have interestingness scores higher than 125. As the minimum required interestingness value of the result set is 150, the evaluation of P can stop without considering the remaining instances c_3 and c_4. □

The formula for the above bound requires the number of instances, which are covered by a subgroup. This number might not yet be known during the evaluation iteration of the subgroup. However, a simple upper bound for the maximum number of instances in a subgroup can be estimated, e.g., by the known number of instances covered by a generalization of the subgroup.

4.3 Data Representations

In subgroup discovery, specialized data structures allow for the efficient computation of subgroup statistics. These are required to determine interestingness scores and optimistic estimates of subgroups. This section introduces adaptations of two data structures to the

setting of subgroup discovery with numeric target concepts, that is, FP-trees and vertical bitset-based data structures. It primarily focuses on the most important interestingness measures, that is, generic mean-based interestingness measures.

4.3.1 Adaptations of FP-trees

As described in detail in Section 3.3.3, the adapted FP-tree structure for subgroup discovery as proposed for the SD-Map algorithm [22] consists of nodes, which are connected by two link structures, tree links and auxiliary links. The modifications for numeric target concepts do not affect the link structures, but only exchanges the information that is stored in each node. For the binary case, an FP-tree node stores the number of instances and the number of positive instances for the respective instance set. In the case of a numeric target concept and a mean-based interestingness measure, the sum of target values and the instance count is stored instead. This enables the computation of the mean target value of an instance set and thus allows for determining the interestingness value. Using these adaptations for numeric target variables, the case of a binary target is included as a special case, if the value of the target is set to 1 for true target concepts and to 0 for false target concepts.

To efficiently compute the optimistic estimate presented above, additional information is required, which is also stored in the tree nodes: To compute the optimistic estimate oe^1_{mean} presented in Theorem 2 one additional field is used, which is initialized with 0. For each instance c corresponding to the node, the value $\max(0, T(c) - \mu_\emptyset)$ is added to this field. As the other stored values, this field is propagated recursively, when conditional trees are built. In doing so, the value stored in this field of each node reflects the sum $\sum_{c \in P: T(c) > \mu_\emptyset} (T(c) - \mu_\emptyset)$, that is, the exact value of the optimistic estimate. Thus, the optimistic estimate is directly available if pruning options are checked. Analogously, for the optimistic estimate $oe^a_{mean}(P) = \widetilde{p}_P{}^a \cdot (T_P^{max} - \mu_\emptyset)$, each node must keep track of the number of instances $\widetilde{p}$, which have a target value greater than the population mean target value, and the maximum target value corresponding to this node T^{max}. These are propagated accordingly and allow for the efficient computation of these bounds.

Adaptations for other interestingness measures For other interestingness measures, different kinds of information needs to be stored in the tree nodes:

To apply a variance-based interestingness measure such as the t-score measure $q_t(P)$, in addition to the sum of values the sum of squared values needs to be stored to determine the variance within the subgroup. This is discussed in detail in the context of *generalized FP-trees*, see Chapter 5. Unfortunately, it is difficult to determine optimistic estimates for this function in general, see Section 4.2.

To compute the symmetric generic mean-based measures, no additional information is required other than the sum of values and the frequency count of instances. In order to determine optimistic estimates, it is required to additionally store the sum of target values, which are below the population target mean, and the minimum target value. Similarly, for the variance reduction $q_{vr}(P)$ the instance count and the overall sum of target values is required to compute the interestingness itself. For the computation

of optimistic estimate bounds, the sum of target values higher, resp. lower than the population mean as well as the minimum and maximum target value is also required.

The generic median-based measures cannot be computed by applying an FP-tree-based data structure since more than one pass over the subgroup is required to compute the required median statistics, cf. Chapter 5. The same applies to the Kolmogorov-Smirnov interestingness measure q_{ks}.

To compute rank-based interestingness measures using a variation of FP-trees, the ranks of instances must be determined in a preprocessing step. Afterwards, ranks replace the original target values in the algorithm itself. In the FP-tree nodes, only the instance count and the sum of ranks is stored and aggregated.

4.3.2 Adaptation of Bitset-based Data Structures

To adapt bitset-based, vertical data structures to numeric target settings and to the introduced ordering-based bounds, two adaptations to the data structure in the binary setting are necessary. First, the instances of the total population are initially sorted in descending order with respect to the target variable. The ordering allows for easy computation of ordering-based optimistic estimate bounds. Analogously to the binary case, a bitset of length $i_\emptyset$ is constructed for each selector in the search space, such that the i-th bit in each bitset is set to true, iff the instance with the i-th highest target value is covered by the respective selector. For subgroups with conjunctive descriptions, a corresponding bitset is then computed by performing logical *AND* operations on the bitset of the respective selectors.

Second, the numeric target values are stored in an array in descending order. This replaces the additional bitset used for the target concept in the binary case. The array of target values and the bitsets for the selectors correspond to each other via the position of the instances, i.e., the target value of an instance, which is represented by the n-th bit of a bitset, is given at position n of the array of target values.

The computation of (for example) the mean value of a subgroup $sel_1 \wedge sel_2$, requires one iteration over all bits, which are set to true in the bitset that corresponds to this subgroup. For each bit, which is set to true, the respective target value of the array is added to a total sum and the count of instances is incremented. These statistics are then used to compute the mean value.

Technically, each bitset is divided into words (e.g., of 32 or 64 bits), on which logical boolean operations (such as *OR* and *AND*) can be applied very efficiently. For each selector, the respective bitset is stored in a map for quick access.

The runtime requirement to sort a dataset with $i_\emptyset$ instances according to the target values is in $O(i_\emptyset \cdot \log i_\emptyset)$. This is dominated by the search process in the subgroup discovery algorithm, unless the search space is very small in comparison to the number of instances in the dataset, i.e., $|\Sigma| < \log i_\emptyset$. The construction of the bitset is accomplished in one single pass through the database. The rest of the algorithm can then operate on the generated data structures only and does not need further access to the database. The memory consumption for $i_\emptyset$ instances in the database and $|\mathcal{S}|$ selectors is thus given by $i_\emptyset \cdot (|\mathcal{S}| + 64)$ bits, if the numeric target values are stored with 64 bit precision.

4.4 Algorithms for Subgroup Discovery with Numeric Targets

Next, two novel algorithms for efficient subgroup discovery are presented that utilize the theoretical results of this chapter, that is, novel optimistic estimate bounds and data structures. While both algorithms employ a similar enumeration strategy – depth-first-search with a one-level look-ahead – the algorithms differ fundamentally in the used data structures: The *SD-Map** algorithm, which is presented first, is based on adapted FP-trees. The algorithm *NumBSD* works on an adapted bitset-based data structure. Both algorithm exploit efficient strategies for optimistic estimate pruning based on the bounds derived in Section 4.2. Due to the employed data structures the *SD-Map** algorithm is limited to bounds in closed form expressions. In contrast, the algorithm *NumBSD* can also utilize ordering-based bounds, including the fast bounds presented in Section 4.2.4.

This section describes algorithm implementations for the generic mean-based interestingness measures. Solutions for other measures can be derived by slight modifications, as discussed in the previous sections.

4.4.1 The *SD-Map** Algorithm

The *SD-Map** algorithm is a novel algorithm for subgroup discovery with numeric target concepts. It improves its predecessor SD-Map [22] in several directions: While SD-Map focuses exclusively on binary targets, *SD-Map** modifies the employed FP-tree data structure in order to determine statistics of the numeric target concept as described in Section 4.3.1.

The statistics contained in the nodes of the FP-trees is not only used to compute the interestingness of subgroups, but also their optimistic estimates. In that direction, *SD-Map** allows the incorporation of pruning based on the bounds in closed form expressions, which have been presented in Section 4.2.2. Ordering-based bounds cannot be applied since the ordering information is not captured by FP-tree representations, see Section 5.3 for a formal proof.

Pruning is applied in two different forms within the algorithm: First, *selector pruning* is performed in the recursive step, when a conditional FP-tree is built. A (conditioned) branch is omitted if the optimistic estimate for the conditioning selector is below the threshold given by the k best subgroup qualities. Second, *header pruning* is used, when a (conditional) frequent pattern tree is constructed. Here, all the nodes with an optimistic estimate below the mentioned interestingness threshold can be omitted.

To maximize the efficiency of pruning, also the search strategy was slightly modified: Instead of the basic depth-first-search used in the SD-Map algorithm, *SD-Map** applies a depth-first strategy, similar to the *DpSubgroup* algorithm by Grosskreutz et al. [100]. Reordering of the search space is performed by sorting of the header nodes: During the iteration over the candidate selectors for the recursive call, the selectors are reordered according to their optimistic estimate value. In doing so, more promising selectors are evaluated first. In a top-k-approach, this helps to include high scoring subgroups early into the result set in order to provide higher thresholds for more efficient pruning.

4.4.2 The *NumBSD* Algorithm

Although FP-tree-based approaches have shown excellent performance particularly with large datasets, they are not applicable for all interestingness measures. Additionally, they can not make use of ordering-based pruning schemes and the construction of an initial FP-tree can require significant overhead, particularly if the search is limited to short search depths. Therefore, we present the exhaustive subgroup discovery algorithm *NumBSD* as an alternative: It uses an efficient vertical, bitset-based data structure as described in the previous section. As search strategy, *NumBSD* employs a depth-first-search approach with one level look-ahead, similar to the *SD-Map** algorithm. The algorithm applies efficient pruning strategies, including ordering-based bounds and fast bounds, see Sections 4.2.3 and 4.2.4.

The algorithm *NumBSD* and its sub-procedures are shown in Algorithm 9. It first initializes the vertical data structures and then calls the main recursive function *recurse*.

This function consists of two parts. In the first part (lines 2-9) all direct specializations, that is, all subgroups created by adding a single selector to the description of the current subgroup, are considered. For these specializations the corresponding bitsets, the interestingness value and the optimistic estimates are computed. This is achieved efficiently in a single run through the subgroup by the method *computeRefinement* described below. If the interestingness value of a specialization is sufficiently high, then it is added to the result set. This potentially replaces a subgroup with a lower interestingness score and increases the minimum required interestingness score of the result set. Only if the optimistic estimate for a specialization exceeds the minimum interestingness value of the result set, this refinement is also considered for the recursive search. In the second part of the function (lines 10-12), it calls itself recursively for these candidate subgroups.

For the performance of the algorithm, an efficient computation of the bitset, the interestingness score and the optimistic estimate of a refinement is essential. This is performed in the function *computeRefinement* of Algorithm 9. First, an upper bound for the maximum number of instances of a refinement is given by the number of instances for the current subgroup and for the additional selector. Each bitset technically consists of words of 32 (respectively 64) bits. The bitset representing the instances of the specialization *spec* is computed word by word by a logical *AND* between the bitset of the current subgroup and the bitset of the new selector. Then, for each bit in this word, which is set to true (each instance of the refinement), the count and sum of target values are adjusted. Based on these values the interestingness score of the current part of the refinement is computed. When considering the i-th bit of the refinement, the interestingness is equal to $q(s_i^{desc})$ for the refinement *spec* in the terminology of Theorem 11. Thus the maximum of the interestingness scores computed in this way determines a tight optimistic estimate for the subgroup under evaluation. Since $s_{i_P}^{desc} = sg(P)$, the last of these scores is equal to the interestingness of the overall subgroup.

As a further improvement, the fast optimistic estimates as proposed in Theorem 19 are checked after each word of the bitset. If neither this bound nor any of the already computed values $q(s_i^{desc})$ are sufficiently high for the result set, then the evaluation of

Algorithm 9 *NumBSD* algorithm

```
1:  function NumBSD(maxDepth)
2:      Sort(allInstances) // sort w.r.t. target values
3:      TargetVal ← array of target values
4:      for all sel in allSelectors do
5:          bitsets(sel) ← createBitset(sel)
6:      allTrue ← new bitset, all bits set to 1
7:      recurse(allTrue, ∅, allSelectors, maxDepth)

1:  function recurse(currentBitset, currentDescription, remainingSels, maxDepth)
2:      nextSelectors ← ∅
3:      for all sel in remainingSels do
4:          nextBitSet ← computeRefinement(currentSG,sel)
5:          if nextBitSet.estimate > result.minQuality then
6:              nextBitsets(sel) ← nextBitset
7:              nextSelectors ← nextSelectors ∪ sel
8:              if nextBitSet.quality > result.minQuality then
9:                  result.add (currentDescription ∪ sel)
10:     if prefix.size < maxDepth then
11:         for all sel in nextSelectors do
12:             recurse(nextBitsets(sel), prefix ∪ sel, nextSelectors \ sel, maxDepth)

1:  function computeRefinement(currentBitset, sel, minQualityThreshold)
2:      maxN ← Math.min(currentBitset, bitsets(sel).cardinality)
3:      n ← 0
4:      sum ← 0
5:      maxEstimate ← 0;
6:      refinement ← new bitset ()
7:      for all i = 0 to countWords (currentBitset) do
8:          refinement.word[i] ← currentBitset.word[i] AND bitsets(sel).word[i]
9:          for all each bit b in refinement.bitset.word[i], that is set to true do
10:             n ← n+1
11:             currentValue ← TargetVal [global position of b]
12:             sum ← sum + currentValue
13:             maxEstimate = max (maxEstimate, computeQuality (n, sum))
14:         sumEstimateAtEnd = sum + currentValue · (maxN - n);
15:         maxOEatEnd = computeQuality (maxN, sumEstimateAtEnd)
16:         if (maxEstimate < minQuality ∧ maxOEatEnd < minQuality) then
17:             refinement.optEstimate = max (maxEstimate, maxOEatEnd)
18:             return refinement % exploit fast pruning bounds
19:     return refinement
```

the current subgroup can stop. Since the subgroup itself and all of its generalization will not contribute to the result set, the exact values of the interestingness and the optimistic estimate are not of interest anymore. Thus, parts of this computation can be safely omitted using the fast bounds introduced previously. In the worst case, the method *computeRefinement* requires one complete pass through the instances of the current subgroup.

4.5 Evaluation

The benefits of the proposed improvements were evaluated in a wide range of experiments. The algorithms were implemented in the *VIKAMINE 2* environment, see Chapter 8. Runtime experiments were performed on a standard office PC with a 2.2 GHz CPU and 2 GB RAM. Experiments, which count the number of evaluated candidates, were executed on additional machines since results are independent from the hardware performance.

The experiments used publicly available datasets from the UCI [13] and KEEL [9] data repositories. For nominal attributes, attribute-value pairs were used as selectors. Numeric attributes were discretized into ten intervals by equal-frequency discretization. No overlapping intervals were generated.

The first part of the evaluation was concerned with the effects of the introduced optimistic estimate bounds. In this regard, also the influence of the result size was investigated. The next part of experiments focused on the proposed data structures: It compared the runtimes of a simple depth-first-search algorithm, the *NumBSD* algorithm and the *SD-Map** algorithm without the use of optimistic estimate pruning. Then, the full algorithms with advanced data structures and optimistic estimate pruning were evaluated in another series of experiments. Finally, the effects of the introduced fast bounds with limited information were investigated.

A focus of the experiments is on the generic mean-based interestingness measures since these are most popular in practical applications.

4.5.1 Effects of Optimistic Estimates

The first set of experiments evaluated the use of the introduced optimistic estimate bounds. In that direction, a subgroup discovery algorithm with depth-first-search, one level look-ahead, and no resorting was run for different interestingness measures and different search depths (maximum numbers of selectors in a description). For the optimistic estimate pruning, a top-1 approach was applied, that is, it was searched only for the one subgroup with the top score. Each run was executed three times in different variations: With no optimistic estimate pruning, with optimistic estimate pruning using the bounds in closed form, which were presented in Section 4.2.2, and with ordering-based optimistic estimate pruning, cf. Section 4.2.3. For each variation, the number of evaluated candidates in recursive calls, that is, after the initial evaluation of the basic selectors (not including these), was counted.

The respective results are summarized in Table 4.3, Table 4.4, and Table 4.5. Results for the mean-based interestingness measures, see Table 4.3 and Table 4.4, are discussed first.

It can be observed that the number of required evaluations is reduced massively by applying the presented optimistic estimate bounds. Even at low search depths, significant improvements are achieved, often several orders of magnitude. The difference increases even further for higher search depths. In particular for ordering-based pruning, in many datasets (almost) all further specializations can be pruned at depth two or three. In doing so, the required number of evaluations is not increased at all if the search is extended to higher depths. In contrast, the unpruned size of the search size increases substantially.

Regarding different interestingness measures, optimistic estimate pruning has generally less impact if the parameter a in the interestingness measures is lower (e.g., $a = 0.5$ and $a = 0.1$), that is, if deviations of the target concept are more important. This can be explained by the fact that even small subgroups can achieve high scores in this scenario and thus the anti-monotonicity of the subgroup size is more difficult to exploit in these cases. An exception is the extreme parameter $a = 0$, which is equivalent to the *average interestingness measure*: Since the best refinement of a subgroup is already determined by the single instance with the highest target value, all subgroups that do not cover one of the instances with high target values can immediately be pruned by applying the novel estimates of Theorem 4.

For the extreme settings $a = 1$ and $a = 0$, the bounds in closed form are tight, that is, they allow for the same amount of pruning as ordering-based bounds. For the intermediate settings $a = 0.5$ and $a = 0.1$, bounds in closed forms are considerably less precise. Therefore, often substantially more candidates must be evaluated in comparison to ordering-based bounds. However, the optimistic estimate bounds in closed form still reduce the number of required evaluations by orders of magnitude in comparison to the unpruned search space. Note that ordering-based bounds cannot be combined with all data structures and come at higher computational costs.

Results for additional interestingness measures, are displayed in Table 4.5. As before, the maximum search depth was limited to $d = 5$. Similar to the mean-based interestingness measures, applying optimistic estimate bounds can lead to a massive reduction of necessary subgroup evaluations. However, the amount of the decrease of course is heavily influenced by the utilized interestingness measure. For the symmetric mean-based measure and the variance reduction, the number of evaluated candidates is often decreased to less than 1000, if ordering-based optimistic estimates are applied. The optimistic estimates in closed form are less tight, but still reduce the number of required evaluations by more than an order of magnitude or more. Ordering-based bounds are also very effective for the other investigated interestingness measures, that is, the median-based measure $q_{med}^{0.5}(P)$, the Mann-Whitney measure $q_{mw}(P)$, and the area-under-the-curve $q_{auc}(P)$. Regarding optimistic estimates in closed form, even relatively simple to derive bounds can reduce the number of required candidate evaluations substantially, as indicated by the results for the Kolmogorov-Smirnov interestingness measure. The least effective bounds were by far the optimistic estimates for the Mann-Whitney, which only was able to prune about 40% of the candidates on average.

Table 4.3: Comparison of pruning schemes, i.e., no pruning (None), ordering-based bounds (Order.) and bounds in closed form (Closed). The table provides numbers of subgroups, which had to be evaluated in a depth-first-search with one level look-ahead and no resorting using the mean-based interestingness measure with the parametrization $a = 0.5$ (not counting the basic selectors). A maximum search depth $d = 2$, $d = 3$, or $d = 4$ was applied as indicated in the column header.

Dataset	$d = 2$			$d = 3$			$d = 4$		
	None	Cl. form	Order.	None	Cl. form	Order.	None	Cl. form	Order.
adults	7,408	2,754	228	181,781	17,289	733	1,756,627	48,852	1,363
ailerons	59,340	6,486	193	4,175,423	43,438	680	166,962,685	234,292	1,986
autos	20,178	770	51	416,326	2,665	70	2,862,509	5,164	65
breast-w	2,774	317	19	36,831	412	19	124,310	427	19
census-kdd*	87,725	4,254	995	6,544,276	39,199	5,072	73,374,193	262,179	19,012
communities*	708,645	237,077	84	246,954,908	357,336	91	$> 2 \cdot 10^9$	384,142	92
concrete_data	2,346	1,656	123	35,531	2,440	130	130,138	2,505	131
credit-a	3,515	2,267	259	66,035	8,679	552	540,727	18,280	805
credit-g	3,986	2,502	258	98,357	14,933	577	1,241,692	44,898	818
diabetes	2,277	2,015	96	39,485	4,663	101	206,121	5,113	102
elevators	11,175	8,128	100	453,250	20,250	184	11,435,341	38,657	296
flare	1,045	98	98	7,732	216	216	34,721	359	359
forestfires	6,241	853	316	121,187	860	172	574,004	1,007	172
glass	2,761	72	16	34,228	81	17	93,952	81	17
heart-c	2,000	1,754	353	33,451	6,386	712	233,043	11,429	930
house	10,585	10,585	442	461,635	126,016	480	13,337,630	170,889	482
housing	6,105	319	67	134,738	1,099	155	675,904	2,689	276
letter	9,971	5,048	9	390,239	10,173	9	7,770,059	10,226	9
mv	3,003	58	35	59,689	60	35	722,532	60	35
pole	5,778	2,141	657	181,280	12,866	3,785	3,813,384	50,315	18,816
sonar	175,527	10,905	12	25,518,877	15,115	12	277,955,231	15,467	12
spambase	21,527	18,530	6,848	1,296,913	314,807	48,526	46,468,924	4,324,121	641,296
ticdata	176,715	55,633	5,345	16,421,461	1,913,911	58,952	780,664,796	33,729,942	650,638
yeast	1,908	1,162	22	26,294	1,790	26	132,486	1,916	27

Table 4.4: Comparison of pruning schemes, i.e., no pruning (None), ordering-based bounds (Order.) and bounds in closed form (Closed). The table provides numbers of subgroups that had to be evaluated in a depth-first-search with one level look-ahead and no resorting (not counting the basic selectors). The search was restricted to a maximum of 5 selectors (only 4 for the two datasets marked with a "*"). For the applied mean-based interestingness measures, different parameters a were evaluated as indicated in the column headers. If no pruning is applied, the number of required candidate evaluations is independent from the applied interestingness measure.

Dataset		1.0		0.5		0.1		0.0	
	None	Closed	Order.	Closed	Order.	Closed	Order.	Closed	Order.
adults	8,503,218	253	253	87,800	1,872	410,321	32,873	1,252	1,252
ailerons	984,289,405	4,298	4,298	912,256	4,384	46,081,383	148,830	1,470	1,470
autos	12,316,190	17	17	8,347	66	52,764	1,884	134	134
breast-w	219,993	12	12	427	19	6,413	963	132	132
census-kdd*	73,374,193	1,815	1,815	262,179	19,012	3,576,514	145,315	1,554	1,554
communities*	$> 2 \cdot 10^9$	22	22	384,142	92	22,581,558	79,187	2,248	2,248
concrete_data	209,041	19	19	2,532	131	3,936	707	457	457
credit-a	2,231,118	277	277	26,358	940	17,105	867	777	777
credit-g	8,389,271	364	364	88,883	906	84,227	383	319	319
diabetes	350,466	17	17	5,113	102	2,008	316	342	342
elevators	121,983,859	62	62	54,677	388	214,793	2,389	1,133	1,133
flare	101,946	29	29	446	446	1,497	1,497	5	5
forestfires	1,209,242	30	30	1,090	172	875	164	161	161
glass	141,714	13	10	81	17	529	87	147	147
heart-c	823,995	224	224	14,086	975	5,942	516	357	357
house	173,768,450	21	21	182,583	482	465,228	13,424	769	769
housing	1,554,972	43	43	4,885	387	4,011	273	127	129
letter	69,157,431	8	8	10,226	9	183,379	3,409	988	990
mv	5,542,943	34	29	60	35	2,262	34	1,499	1,499
pole	47,553,142	48,077	48,077	144,353	72,851	650,349	146,175	270	270
sonar	1,737,064,885	12	12	15,521	12	156,647	1,253	1,208	1,208
spambase	1,045,755,337	2,741,042	2,741,042	44,663,301	6,726,031	11,381,117	2,923,091	878	878
ticdata	1,254,395,632	1,783,651	1,783,651	249,736,031	5,658,523	288,205,279	6,370,265	61	61
yeast	291,042	23	23	1,941	27	3,190	259	217	217

Table 4.5: Effects of the introduced optimistic estimate bounds for further interestingness measures, i.e., a symmetric mean-based measure ($q^{0.5}_{sym}$), the variance reduction (q_{vr}), a median-based measure ($q^{0.5}_{med}$), the Kolmogorov-Smirnov measure (q_{ks}), the Mann-Whitney (q_{mw}) measure, and the Area-under-the-Curve measure (q_{auc}). Depending on the interestingness measure, ordering-based bounds (Order.), bounds in closed form (Closed), or both were tested. The table provides numbers of subgroups that had to be evaluated in a depth-first-search with one level look-ahead and no resorting. The search was restricted to a max. of 5 selectors (only 4 for the datasets marked with a "*"). Bounds in closed form for the measure q_{auc} are not guaranteed to return optimal results in case of ties.

Dataset		$q^{0.5}_{sym}$	$q^{0.5}_{sym}$	$q^{0.5}_{vr}$	q_{vr}	q_{med}	q_{ks}	q_{mw}	q_{MW}	q_{auc} !	q_{auc}
	None	Closed	Order.	Closed	Order.	Order.	Closed	Closed	Order.	Closed	Order.
adults	8,503,218	114,884	313	174,961	195	16,962	78,546	6,086,727	17,900	8654	339
ailerons	984,289,405	11,580,481	49,312	16,419,078	13,905	13,974	2,263,350	949,909,065	28,175	12049	408
autos	12,316,190	24,917	69	29,252	66	4,089	68,856	4,994,035	4,769	1651	347
breast-w	219,993	8,985	14	9,801	17	4,094	15,541	180,065	4,418	995	896
census.*	73,374,193	1,488,608	34,599	1,846,126	5,227	19,281	86,246	63,134,807	15,306	106	856
comm.*	$> 2 \cdot 10^9$	4,184,249	94	13,070,933	110	27,150	6,501,639	$> 2 \cdot 10^9$	47,376	5788	638
concrete.	209,041	13,165	28	15,625	21	4,173	37,009	115,277	5,385	4438	412
credit-a	2,231,118	87,980	1,054	120,844	885	16,663	210,403	1,237,561	23,015	8812	276
credit-g	8,389,271	297,275	906	446,215	867	20,186	1,175,479	5,489,826	46,207	2760	310
diabetes	350,466	15,451	117	19,494	84	8,961	78,174	177,820	11,241	1330	486
elevators	121,983,859	311,094	388	886,706	290	10,173	1,042,651	78,737,713	30,138	3636	716
flare	101,946	3,062	446	7,322	380	2,796	55,649	21,208	2,544	270	1364
forestfires	1,209,242	2,492	171	2,492	168	1,254	180,998	570,104	9,454	365	637
glass	141,714	3,743	29	6,006	25	1,613	23,831	60,032	3,165	451	993
heart-c	823,995	33,453	658	41,367	654	8,070	107,226	475,716	10,594	2900	264
house	173,768,450	733,605	509	1,041,836	401	40,540	196,344	111,649,531	19,420	1977	735
housing	1,554,972	10,339	382	10,831	226	4,862	25,462	438,041	3,430	1543	380
letter	69,157,431	93,471	15	150,737	7	32,817	144,219	44,679,902	26,358	2225	613
mv	5,542,943	69,771	43	120,249	41	7,520	76,108	4,053,866	8,238	24385	492
pole	47,553,142	190,154	72,850	1,300,541	70,712	163,186	1,477,117	27,444,125	300,872	308	2664
sonar	1,737,064,885	185,796	5	241,948	5	11,107	1,885,904	657,387,130	75,905	3279	552
spambase	1,045,755,337	45,509,446	6,726,031	76,577,712	6,709,191	5,456,553	5,288,584	687,069,672	7,088,977	796	2556
ticdata	1,254,395,632	220,568,015	6,659,758	253,256,371	5,528,202	3,715,310	4,218,711	961,636,828	4,662,825	19873	504
yeast	291,042	17,578	28	21,942	28	2,868	54,281	168,478	4,197	3190	439

Overall, the reduction of required candidate evaluations was massive for almost all interestingness measures and datasets. This clearly shows the use of the introduced optimistic estimate bounds for diverse interestingness measures. The remainder of the evaluation will focus on the most popular measures, that is, differently parametrized mean-based interestingness measures.

4.5.2 Influence of the Result Set Size

Top-k subgroup discovery aims at finding the best k subgroups according to the chosen interestingness measure. Optimistic estimate pruning exploits that the specializations of the current subgroup can be guaranteed to receive a lower score than the best k subgroups found so far. Therefore, the thresholds that are utilized for pruning increase slower, if more subgroups are included in the result set. As a consequence, pruning can be applied less often and more candidate subgroups must be explored.

The influence of the result set size on the number of required subgroup evaluations was studied in another series of experiments. Table 4.6 shows the number of candidate evaluations, which were performed in a depth-first-search with one level look-ahead using the mean-based interestingness measure $q^{0.5}_{mean}(P)$ for different sizes k of the result set. The search depth was limited to five. For pruning, ordering-based bounds were applied.

It can be seen that for all sizes of the result set, the number of evaluated candidates is still by orders of magnitudes smaller than in a search without optimistic estimates. Nonetheless, the size of the result set has a noteworthy impact on the amount of pruning in many datasets: As expected, the number of evaluated candidates increases with the size of the result set. Fortunately, the (relative) increase is much more moderate for datasets, which require large numbers of subgroup evaluations even with optimistic estimate pruning, see the datasets *spambase* and *ticdata*. This can be explained by the fact that the large amount of evaluations is required, because the dataset contains many subgroups with similar scores according to the applied interestingness measure. In this case, however, the threshold for the result set, that is, the minimum interestingness score in the result set, is also less influenced by the size of the result set k. Overall, optimistic estimate pruning is clearly also very useful for larger result set sizes.

4.5.3 Influences of Data Structures

In the next series of experiments, the effects of different data structures were investigated. In that direction, the runtimes of the presented algorithms *without* applying optimistic estimate pruning were measured. This was performed for the *NumBSD* algorithm, which is based on a bitset-based representation, as well as for the *SD-Map** algorithm, which is based on FP-trees. For comparison, the task was also solved by a trivial depth-first-search without any specialized data structure (repeated checking of the selection expressions in memory). Since no optimistic estimate bounds are exploited, the runtime was (almost) independent from the applied interestingness measure and size of the result set k, see also Section 5.5.1. The experiments were performed with different maximum

Table 4.6: Comparison of candidate evaluations for different sizes of the result set k if ordering-based optimistic estimate pruning is applied. As interestingness measure, the mean test function $q^{0.5}_{mean}$ was used. The maximum search depth (maximum number of selectors in a description) was limited to $d = 5$.

Dataset	No Pruning	$k = 1$	$k = 10$	$k = 100$
adults	8,503,218	1,872	4,793	21,854
ailerons	984,289,405	4,384	5,440	18,935
autos	12,316,190	66	291	4,283
breast-w	219,993	19	178	3,751
census-kdd*	73,374,193	19,012	22,474	42,815
communities*	$> 2 \cdot 10^9$	92	1,266	32,845
concrete_data	209,041	131	866	4,294
credit-a	2,231,118	940	6,241	18,982
credit-g	8,389,271	906	8,703	24,604
diabetes	350,466	102	2,971	8,002
elevators	121,983,859	388	2,408	8,563
flare	101,946	446	1,210	3,062
forestfires	1,209,242	172	330	2,128
glass	141,714	17	360	4,691
heart-c	823,995	975	4,248	9,203
house	173,768,450	482	4,302	27,033
housing	1,554,972	387	688	81,706
letter	69,157,431	9	1,098	20,229
mv	5,542,943	35	2,072	9,629
pole	47,553,142	72,851	108,189	3,416,279
sonar	1,737,064,885	12	2,094	12,706
spambase	1,045,755,337	6,726,031	6,849,488	6,861,101
ticdata	1,254,395,632	5,658,523	5,710,867	6,215,858
yeast	291,042	27	660	3,001

Table 4.7: Comparison of data structures: The table shows the runtimes in seconds of simple depth-first-search (Simple), *NumBSD* (BSD), and *SD-Map** (SDM) *without optimistic estimate pruning* for different maximum search depths d. Here, results are shown for the interestingness measure $q^{0.5}_{mean}$, but results are very similar for all applicable interestingness measures.

Dataset	instances	$d=2$			$d=3$			$d=4$		$d=5$		$d=6$	
		Simple	BSD	SDM	Simple	BSD	SDM	BSD	SDM	BSD	SDM	BSD	SDM
adults	32561	98.0	1.5	4.3	3068.8	4.5	7.5	24.6	16.3	103.8	42.5	300.5	86.3
ailerons	13750	394.7	1.5	47.5	>4 h	38.4	268.5	1009.0	1748.6	>4 h	10554.0	>4 h	>4 h
autos	205	1.3	< 0.1	0.3	41.4	0.6	1.7	4.3	7.5	20.6	36.3	75.6	145.0
breast-w	699	0.5	< 0.1	0.1	7.5	0.1	0.1	0.2	0.2	0.3	0.4	0.5	0.5
census-kdd	199523	8465.7	47.2	223.4	>4 h	533.2	1039.1	10470.0	5436.0	>4 h	>4 h	>4 h	>4 h
communities	1994	332.7	3.3	482.3	>4 h	567.3	10505.1	>4 h	>4 h	>4 h	>4 h	>4 h	>4 h
concrete_data	1030	0.6	< 0.1	0.1	9.7	0.1	0.2	0.2	0.3	0.4	0.5	0.4	0.5
credit-a	690	0.8	< 0.1	0.2	16.4	0.2	0.7	1.0	2.2	4.1	6.6	10.8	14.8
credit-g	1000	1.4	< 0.1	0.7	36.2	0.3	3.0	3.1	11.6	18.8	40.8	77.5	128.8
diabetes	768	0.4	< 0.1	0.1	7.4	0.1	0.2	0.3	0.4	0.5	0.5	0.6	0.6
elevators	16599	49.2	0.5	9.7	2198.8	5.6	28.9	83.8	96.4	672.1	309.0	2627.7	887.0
flare	1066	0.4	< 0.1	0.1	3.2	< 0.1	0.1	0.1	0.2	0.3	0.4	0.7	0.8
forestfires	517	1.1	< 0.1	0.2	27.4	0.2	0.5	0.9	1.3	2.0	2.9	3.4	4.4
glass	214	0.2	< 0.1	< 0.1	2.5	0.1	0.1	0.1	0.2	0.2	0.3	0.3	0.3
heart-c	303	0.2	< 0.1	0.1	3.7	0.1	0.2	0.4	0.7	1.3	1.8	2.7	3.4
house	22784	70.1	0.8	16.9	3077.3	7.7	51.9	131.6	157.2	1149.7	464.3	3565.7	1060.1
housing	506	0.8	< 0.1	0.2	19.3	0.2	0.5	1.0	1.4	2.5	3.5	4.6	6.3
letter	20000	54.1	0.7	10.6	2157.8	5.3	31.5	64.0	92.5	436.6	275.2	1455.9	601.3
mv	40768	31.4	0.8	5.1	605.7	2.2	8.2	13.5	14.3	76.0	27.5	196.7	44.6
pole	14998	25.0	0.5	7.1	762.1	4.2	27.2	47.4	112.6	430.7	447.8	2894.4	1725.9
sonar	208	9.5	0.5	8.5	1480.1	28.4	100.9	353.1	780.2	2971.2	9084.8	>4 h	>4 h
spambase	4601	25.4	0.5	30.6	1650.4	11.9	287.0	266.8	2521.1	4626.0	>4 h	>4 h	>4 h
ticdata	5822	244.6	2.0	155.2	>4 h	91.0	2018.7	3414.8	>4 h	>4 h	>4 h	>4 h	>4 h
yeast	1484	0.8	< 0.1	0.1	11.3	0.1	0.2	0.3	0.4	0.5	0.7	0.7	0.7

search depths d. Table 4.7 displays representative results for the measure $q^{0.5}_{mean}$ and a result set size of $k = 1$.

The results show that both introduced data structures – the bitset-based structure as well as the FP-tree-based representation – substantially outperform the trivial approach. A direct comparison between the two approaches is more difficult: For lower search depths ($d = 2, 3, 4$), bitset-based structures enable faster runtimes than FP-trees. The differences reach an order of magnitude for some datasets, e.g., for the *communities* and *spambase* datasets. For higher search depths ($d = 5, 6$), the results are more ambiguous: For some datasets the bitsets perform better, for some they perform worse than FP-trees. In particular, for datasets with a high instance count, the FP-tree-based approach are able to finish the tasks fast. In the *census-kdd* dataset, which is the largest tested dataset (in terms of instances), FP-trees perform better than bitsets already at a search depth of 4. This can be explained by the fact that the FP-trees achieve a better compression of the data in datasets with a high instance count.

In summary, FP-trees are the data structure of choice if the dataset contains many instances, and if the maximum allowed number of selectors in a description is large. In contrast, bitsets are preferred if the search is restricted to low search depths, or if the instance count is comparatively low.

For some interestingness measure, it is not possible to derive optimistic estimate bounds, e.g., for generic variance-based measures or the t-score. Therefore, the runtimes of the algorithms without optimistic estimate pruning shown in Table 4.7 reflect the actual algorithm runtimes for these measures.

4.5.4 Runtimes of the Full Algorithms

Another series of experiments compared the runtimes of the full algorithms. Exemplary results of these evaluations are shown in Table 4.8 and Table 4.9. Experiments displayed in Table 4.8 utilized different interestingness measures and fixed search depth of $d = 5$. In contrast, experiments shown in Table 4.9 employed the fixed interestingness measure $q^{0.5}_{mean}$, but a variable search depth.

Results in Table 4.8 indicate that for a search depth of five, the application of optimistic estimate bounds leads to a substantial reduction of runtimes in comparison to the variations without optimistic estimate pruning in almost all cases, cf. Table 4.7. The biggest improvements can be observed for the interestingness measures q^{1}_{mean} and q^{0}_{mean}. This corresponds the respective reduction in necessary candidate evaluations, see Table 4.4. Although the applied pruning bounds are in theory tighter for the *NumBSD* algorithm, the speedups are often larger for the *SD-Map** algorithm. This has two reasons: First, pruning in *SD-Map** is exploited twice, as header pruning and as selector pruning when conditional trees are built. Since the size of the conditional trees is reduced by pruning, not only less candidate evaluations are required with optimistic estimates, but each candidate evaluation also takes less time to compute. Second, the computation of the ordering-based bounds in *NumBSD* is more costly in itself. In one single experiment with unfavorable pruning properties, the computational costs for determining the bounds in this algorithm outweighed the use of pruning, that is, for the

Table 4.8: Comparison of the full algorithms: The table shows the runtimes in seconds for *NumBSD* (BSD) and for *SD-Map** (SDM) with *all pruning options enabled* for different interestingness measures. The search was limited to a maximum search depth of $d = 5$. The first two columns show results of the algorithms without optimistic estimate pruning for comparison.

Dataset	No Pruning		q^1_{mean}		$q^{0.5}_{mean}$		$q^{0.1}_{mean}$		q^0_{mean}	
	BSD	SDM	BSD	SDM	BSD	SDM	BSD	SDM	BSD	SDM
adults	103.8	42.5	0.8	2.3	11.1	6.1	19.7	14.1	1.3	2.5
ailerons	>4 h	10554.0	6.0	1.7	45.3	17.7	95.5	3345.9	0.6	147.0
autos	20.6	36.3	< 0.1	< 0.1	< 0.1	0.1	< 0.1	0.3	< 0.1	< 0.1
breast-w	0.3	0.4	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
census-kdd	>4 h	>4 h	53.3	52.6	5184.3	1280.5	14399.2	5795.2	35.0	52.1
communities	>4 h	>4 h	0.1	0.9	0.1	4.3	2.0	108.0	0.3	0.9
concrete_data	0.4	0.5	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	0.1	< 0.1	< 0.1
credit-a	4.1	6.6	< 0.1	< 0.1	0.1	0.2	< 0.1	0.2	< 0.1	< 0.1
credit-g	18.8	40.8	< 0.1	0.1	0.2	0.9	0.1	0.9	< 0.1	0.1
diabetes	0.5	0.5	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
elevators	672.1	309.0	0.2	0.9	1.0	2.4	1.9	4.1	0.3	0.9
flare	0.3	0.4	< 0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	< 0.1	< 0.1
forestfires	2.0	2.9	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
glass	0.2	0.3	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
heart-c	1.3	1.8	< 0.1	< 0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	< 0.1
house	1149.7	464.3	0.2	1.5	0.8	6.1	1.5	7.3	0.6	1.3
housing	2.5	3.5	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
letter	436.6	275.2	0.1	1.3	0.2	3.9	0.7	10.5	0.5	1.5
mv	76.0	27.5	1.0	1.7	1.9	1.4	0.9	2.8	0.9	1.6
pole	430.7	447.8	26.8	1.6	263.0	38.6	186.8	81.0	0.3	2.5
sonar	2971.2	9084.8	< 0.1	< 0.1	< 0.1	0.1	< 0.1	0.4	< 0.1	< 0.1
spambase	4626.0	>4 h	692.9	609.5	7657.5	4269.1	3319.2	1900.3	0.2	151.2
ticdata	>4 h	>4 h	820.4	56.9	12686.5	>4 h	5588.2	>4 h	0.1	1.0
yeast	0.5	0.7	< 0.1	< 0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	< 0.1

Table 4.9: Comparison of the full algorithms: The table shows the runtimes in seconds of *NumBSD* (BSD), and *SD-Map** (SDM) with *all pruning options enabled* for different maximum search depths d. As interestingness measure, the mean test $q^{0.5}_{mean}$ was used.

Dataset	$d = 2$		$d = 3$		$d = 4$		$d = 5$		$d = 6$	
	BSD	SDM	BSD	SDM	BSD	SDM	BSD	SDM	BSD	SDM
adults	2.5	3.7	5.0	4.3	8.4	5.1	11.1	6.1	12.5	7.0
ailerons	3.1	4.6	7.9	5.8	20.9	9.1	45.3	17.7	78.4	41.3
autos	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	<0.1	< 0.1	0.1	< 0.1	0.1
breast-w	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
census-kdd	92.2	96.6	353.6	199.1	1490.7	462.8	5184.3	1280.5	> 4 h	3469.5
communities	0.2	4.0	0.2	4.0	0.2	4.1	0.1	4.3	0.2	4.4
concrete_data	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
credit-a	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.2	0.1	0.2
credit-g	0.1	0.3	0.1	0.5	0.2	0.7	0.2	0.9	0.2	1.0
diabetes	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
elevators	0.6	2.1	0.8	2.2	0.9	2.2	1.0	2.4	1.1	2.3
flare	< 0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1	< 0.1
forestfires	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
glass	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
heart-c	< 0.1	< 0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1
house	0.7	5.3	0.7	5.5	0.7	5.6	0.8	6.1	0.7	5.6
housing	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	0.1
letter	0.2	3.7	0.2	3.8	0.2	3.8	0.2	3.9	0.2	3.8
mv	1.7	1.4	1.7	1.2	1.7	1.2	1.9	1.4	1.7	1.3
pole	3.5	2.7	18.6	5.9	79.8	14.7	263.0	38.6	699.2	90.2
sonar	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1	< 0.1	0.1
spambase	6.0	21.2	87.6	131.2	865.4	781.6	7657.5	4269.1	> 4 h	> 4 h
ticdata	10.0	88.5	125.9	539.1	1429.2	2939.1	12686.5	> 4 h	> 4 h	> 4 h
yeast	< 0.1	0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	0.1	< 0.1	0.1

spambase dataset and the interestingness measure $q^{0.5}_{mean}$. For the interestingness measure q^{0}_{mean}, the runtimes of *SD-Map** are surprisingly high in the datasets *spambase* and *aileron*. This could be explained by the fact that for this interestingness measure and these datasets very specific subgroups have to be found early to exploit the optimal bounds. Additionally many ties occur in the sorting based on the optimistic estimates, which are differently solved in the different algorithms. Therefore, *SD-Map** explores more candidates than necessary in the best case. However, there is still is a substantial speedup in comparison to unpruned algorithm variants.

The runtimes of the both proposed algorithms differ significantly in several cases. Unfortunately, a recommendation for choosing between the two novel algorithms in a certain task remains difficult. As a tendency, for the scenarios with relevant runtimes (> 5 seconds), *SD-Map** is preferred for the interestingness measures that select subgroups with higher coverage (q^{1}_{mean} and $q^{0.5}_{mean}$) and *NumBSD* is preferred for $q^{0.1}_{mean}$ and q^{0}_{mean}, but there are several exceptions for this rule of thumb in the experiments.

Table 4.9 displays the algorithm runtimes for different search depth using the interestingness measure $q^{0.5}_{mean}$. A comparison to the unpruned algorithms, see Table 4.7, shows substantial runtime improvements in most cases. The improvements were in particular strong for larger search depths, i.e., $d = 5$ and $d = 6$, where most runtimes decreased by more than an order of magnitude. For several datasets, the runtime did not (or only marginally) increase with higher search depths, e.g., for the datasets *autos*, *communities*, or *elevators*. This is a sharp contrast to the variants, which do not employ optimistic estimate pruning and can be explained by the fact that already at search level two or three all further candidates can be pruned. For medium search depths $d = 3$ and $d = 4$, also substantial runtime improvements can be observed in most cases with relevant runtimes (> 5 seconds), but the performance gains are not as massive as for the high depth searches. For the minimum search depth $d = 2$, the gains for *SD-Map** were only moderate, while *NumBSD* took even more time than its variation without pruning. At this low search depth, the effects of the pruning seems to have less influence than the additional computational costs for computing the bounds. However, for this search depth the runtimes of *NumBSD* were very low in most cases anyway. For a few datasets the costs for computing the optimistic estimates exceeded the gains from utilizing the pruning bounds even for higher search depth, e.g., in the datasets *ticdata* and *spambase*. This was never the case for the *SD-Map** algorithm.

Comparing both novel algorithms with each other, *SD-Map** excels for higher search depths ($d \geq 4$), where it outperforms *NumBSD* for most experiments with relevant runtimes, that is, if tasks take more than five seconds to complete. In contrast, for the low search depths $d = 3$ and especially $d = 2$, *NumBSD* performs better. In these cases, the necessary overhead for the FP-trees in *SD-Map** seems to be too high to be worth it. These results are in general in line with the previous recommendations for the unpruned algorithm versions. However, since *SD-Map** does profit more from the optimistic estimate bounds than *NumBSD*, it also performs better at the medium search depths $d = 3$ and $d = 4$ in several cases. The runtimes and thus the preferences of the algorithms do not correlate as strong with the dataset size as in the unpruned variants, but also depend strongly on the pruning opportunities in the respective datasets. Unfortunately,

Table 4.10: Comparison of the full algorithms for the median interestingness measure $q_{med}^{0.5}$ for different search depth d: The table shows the runtimes in seconds for the *NumBSD* (BSD) with all pruning options enabled and for a variation that does not exploit the optimistic estimate pruning bounds (NoPr).

Dataset	$d = 2$		$d = 3$		$d = 4$		$d = 5$	
	NoPr	BSD	NoPr	BSD	NoPr	BSD	NoPr	BSD
adults	5.3	3.3	19.4	5.8	68.0	8.7	203.3	10.7
ailerons	12.9	2.5	174.1	7.0	2268.6	18.4	> 4 h	38.0
autos	0.1	< 0.1	0.9	< 0.1	5.6	< 0.1	25.0	< 0.1
breast-w	0.1	< 0.1	0.1	< 0.1	0.3	< 0.1	0.5	< 0.1
census-kdd	205.4	103.8	2416.5	268.4	> 4 h	666.3	> 4 h	1321.3
communities	13.5	0.4	806.9	0.4	> 4 h	0.4	> 4 h	0.4
concrete_data	0.1	< 0.1	0.2	< 0.1	0.3	< 0.1	0.5	< 0.1
credit-a	0.1	0.1	0.5	0.1	1.9	0.1	5.9	0.1
credit-g	0.3	0.1	1.5	0.2	7.7	0.2	32.6	0.2
diabetes	0.1	< 0.1	0.1	< 0.1	0.4	< 0.1	0.6	< 0.1
elevators	3.6	0.9	20.8	1.0	140.6	1.1	830.5	1.2
flare	0.1	< 0.1	0.3	< 0.1	0.9	< 0.1	1.8	0.1
forestfires	0.1	< 0.1	0.3	< 0.1	1.0	< 0.1	2.2	< 0.1
glass	< 0.1	< 0.1	0.1	< 0.1	0.2	< 0.1	0.2	< 0.1
heart-c	< 0.1	< 0.1	0.2	< 0.1	0.6	0.1	1.5	0.1
house	4.2	1.7	22.5	1.7	177.5	1.7	1248.7	1.7
housing	0.1	< 0.1	0.3	< 0.1	1.3	< 0.1	2.9	< 0.1
letter	3.7	0.8	18.7	0.8	106.0	0.8	538.2	0.8
mv	3.5	1.4	9.4	1.4	28.3	1.4	94.6	1.4
pole	6.0	3.4	46.2	17.5	295.3	73.5	1603.0	241.5
sonar	0.6	< 0.1	29.1	< 0.1	360.9	< 0.1	3200.4	< 0.1
spambase	8.1	6.3	144.1	73.3	2051.8	822.1	> 4 h	7369.1
ticdata	22.9	8.7	642.3	109.1	> 4 h	1162.8	> 4 h	9802.7
yeast	0.1	< 0.1	0.2	< 0.1	0.4	< 0.1	0.7	< 0.1

the respective properties are difficult to determine beforehand.

The experimental results for the mean-based interestingness measure showed substantial runtime reductions. Similar performance gains as for mean-based measures can also be achieved for other interestingness measures. As one more example, Table 4.10 displays results for the median-based interestingness measure $q_{med}^{0.5}$. It shows the runtimes of two variations of the *NumBSD* algorithm for this setting, which utilize the (ordering-based) optimistic estimate bounds or ignore them respectively. It can be observed that the runtimes gains are substantial if the optimistic estimates are employed, as it was suggested by the number of required candidate evaluations for this measure, see Table 4.5. Similar to mean-based interestingness measures, for many datasets the runtime does not increase further with higher search depths, since already all candidates could be pruned at lower depth. Note that *SD-Map** can not be used for this interestingness measure, as the median cannot be computed by the FP-tree data structure, cf. Section 5.3.

4.5.5 Effects of the Fast Pruning Bounds

Section 4.2.4 introduced a new category of optimistic estimates, which can already be applied if only a part of the current subgroup is analyzed. These are incorporated in the *NumBSD* algorithm and were also included in the previous experiments. To measure the effects of the novel bounds, the runtimes of the full *NumBSD* algorithm was compared with a variation that did not employ these bounds. The search employed a maximum search depth of $d = 5$ and differently parametrized mean-based interestingness measures. The results are displayed in Table 4.11. Datasets, which could be solved very fast (< 0.2 seconds) by both variants, are omitted.

The results indicate that the influence of the additional bounds, which can be computed early in the evaluation process, is somewhat limited. The runtimes are most improved for the interestingness measure q^1_{mean}: For this measure, the improvements for most datasets are between ten and forty percent. Pruning bounds for this measure seem to be more easily exploitable since this measure requires subgroups that cover many instances.

For other interestingness measure the benefits are less significant and do not exceed ten percent in many cases. However, only in a single setting (for the dataset *mv*), the computational of determining the additional bounds were higher than the saved efforts. Potentially, this kind of pruning requires additional optimization in the implementations to show its full benefits, e.g., by checking the additional bounds only at certain points in the evaluation.

Overall, it has to be reported that for now the novel kind of bounds has not the decisive effect that we hoped for. Instead, it is more of a minor addition in order to optimize the algorithm. In the future, this kind of pruning could be exploited with possibly stronger effects in distributed subgroup mining: If nodes are assigned to computational units according to their target values, and pruning bounds can already be applied at one unit, then the other units are not required to be involved, thus reducing the communication costs.

4.5.6 Evaluation Summary

The experiments showed clearly the effectiveness of the proposed improvements. The presented optimistic estimates were able to massively reduce the number of required candidate evaluations for almost all interestingness measures. As expected, ordering-based bounds had even stronger effects, but bounds in closed forms were good approximations most of the time. Although increasing the size of the result set reduces pruning possibilities, still the vast majority of the search space can be pruned in most cases. Regarding data structures, both novel data structures outperformed a simple approach by far. While for searches with high search depths and large datasets the FP-tree structure enabled faster completion of the tasks, a bitset-based structure is better suited for the other tasks. A comparison of the full algorithms showed that improvements on data structures and optimistic estimate bounds can be combined well. The incorporation of the bounds further reduced the runtimes by an order of magnitude. The *SD-Map** algo-

Table 4.11: Evaluation of the full *NumBSD* (BSD) algorithm with a variation that does *not* employ the fast pruning bounds that can already be exploited by evaluating a part of the subgroup (NoFP). The comparison was performed with a maximum search depth of $d = 5$ and different mean-based interestingness measures. Datasets, which could be solved very fast (< 0.2 seconds) by both variants, are omitted.

Dataset	q^1_{mean}		$q^{0.5}_{mean}$		$q^{0.1}_{mean}$		q^0_{mean}	
	NoFP	BSD	NoFP	BSD	NoFP	BSD	NoFP	BSD
adults	1.7	0.8	14.4	11.1	27.4	19.7	1.4	1.3
ailerons	6.5	6.0	47.1	45.3	110.9	95.5	0.6	0.6
census-kdd	89.6	53.3	5872.8	5184.3	>4 h	>4 h	35.6	35.0
communities	0.2	0.1	0.4	0.1	3.9	2.0	0.3	0.3
credit-g	0.1	< 0.1	0.4	0.2	0.1	0.1	< 0.1	< 0.1
elevators	0.3	0.2	1.3	1.0	3.2	1.9	0.3	0.3
house	0.5	0.2	1.2	0.8	3.2	1.5	0.6	0.6
letter	0.4	0.1	0.7	0.2	1.0	0.7	0.5	0.5
mv	0.7	1.0	1.3	1.9	1.0	0.9	0.8	0.9
pole	34.0	26.8	282.6	263.0	204.8	186.8	0.3	0.3
spambase	853.1	692.9	7989.2	7657.5	3555.2	3319.2	0.2	0.2
ticdata	1019.4	820.4	>4 h	12686.5	6440.6	5588.2	0.4	0.1

rithm did profit more from the additional pruning bounds since also the computational costs for single candidate evaluations are reduced. Unfortunately, a clear recommendation between the two novel algorithms remains difficult. As a tendency, *SD-Map** is to be preferred for more demanding tasks with higher search depths, while *NumBSD* performs better for low search depths. An additional examination of the introduced fast pruning bounds in the *NumBSD* algorithm showed that the effects of these additional bounds are only limited.

4.6 Overview of Computational Properties of Interestingness Measures

This chapter showed that efficiency optimizations for subgroup discovery with numeric target concepts does strongly depend on the applied interestingness measures. Some interestingness measures, such as the t-score, can be determined by *SD-Map**, taking full advantage of the more sophisticated, compressed FP-tree data structure. Other interestingness measures in turn, cannot be determined by *SD-Map** at all, such as for example median-based measures. Additionally, ordering-based optimistic estimate bounds could be derived for a large variety of interestingness measures, but for a few exceptions this was not possible, e.g., for the t-score.

Table 4.12 summarizes interestingness measures with respect to their computational properties. In particular, the table shows for each interestingness measure if there is an optimistic estimate in closed form presented in this work, which can be computed using

FP-trees, if it is estimable by ordering, and if the measure itself is computable by the *SD-Map** algorithm.

Table 4.12: A summary of interestingness measure with respect to properties regarding efficient computation.

Measure	Notation	Formula	Estimate in closed form	Estimable by ordering	Computable in *SD-Map**
Impact	$q^1_{mean}(P)$	$i_P(\mu_P - \mu_\emptyset)$	yes	one-pass	yes
Generic mean	$q^a_{mean}(P)$	${i_P}^a(\mu_P - \mu_\emptyset)$	yes	one-pass	yes
z-score	$q_z(P)$	$\sqrt{i_P}\frac{(\mu_P - \mu_\emptyset)}{\sigma_0}$	yes	one-pass	yes
Generic variance-based	$q^a_\sigma(P)$	${i_P}^a(\sigma_P - \sigma_\emptyset)$	no	no	yes
t-score	$q_t(P)$	$\sqrt{i_P}\frac{(\mu_P - \mu_\emptyset)}{\sigma_s}$	no	no	yes
Generic symmetric mean	$q^a_{sym}(P)$	${i_P}^a\lvert\mu_P - \mu_\emptyset\rvert$	yes	two-pass	yes
Variance reduction	$q_{vr}(P)$	$\frac{i_P}{(i_\emptyset - i_P)}(\mu_P - \mu_\emptyset)^2$	yes	two-pass	yes
Generic median measure	$q^a_{med}(P)$	${i_P}^a(med_s - med_0)$	(yes)[2]	one-pass	no
Kolmogorov-Smirnov	$q_{ks}(P)$	$\sqrt{\frac{i_P \cdot \lvert\bar{s}\rvert}{i_\emptyset}}\Delta_{P,\neg P}$	yes	no[3]	no
Mann-Whitney	$q_{mw}(P)$	$\sqrt{\frac{i_P}{\lvert\bar{s}\rvert}}(\frac{\mathcal{R}}{i_P} - \frac{i_\emptyset}{2})$	yes	one-pass	yes
Area-under-the-curve	$q_{auc}(P)$	$\frac{\bar{\mathcal{R}} - \frac{i_{\neg P} i_{\neg P} + 1}{2}}{i_P \lvert\bar{s}\rvert}$	yes	one-pass	yes

4.7 Summary

This chapter analyzed the task of efficient exhaustive subgroup discovery with numeric target concepts. For this task, improvements were introduced in several directions: Novel optimistic estimate bounds were proposed for a large variety of interestingness measures. While some of the bounds are expressed in closed forms based on few key statistics, others rely on a specific property of interestingness measures, which was introduced in this work. For the most popular class of measures in that area, generic mean-based measures, additional bounds were proposed that can be applied with only partial knowledge of the currently evaluated subgroup, allowing to prune a candidate pattern before its full evaluation. Additionally, it was shown how popular data structures for subgroup discovery can be adapted to the task of subgroup discovery with numeric target concepts.

[2]The generic measure itself cannot be computed by *SD-Map**.
[3]Not yet determined.

The proposed improvements are incorporated in two novel algorithms, that is, the *SD-Map** algorithm and the *NumBSD* algorithm. An extensive runtime evaluation shows, both algorithms achieve runtime improvements of more than an order of magnitude in comparison to a basic approach.

4.8 Appendix

Lemma 1 *Using the notations of Theorem 19, the function* $f^a(x)(n+x)^a \cdot \left(\frac{\sigma+x\cdot\theta}{n+x} - \mu_\emptyset\right)$ *has no local maxima inside its domain of definition:*

$$f^a(x) \leq max(f(0), f(x_{max})) \qquad \square$$

Proof We distinguish three cases by the parameter a of the applied generic mean interestingness measure:

First, for $a = 1$, it holds that

$$\begin{aligned} f^1(x) &= (n+x)^1 \cdot \left(\frac{\sigma + x\cdot\theta}{n+x} - \mu_\emptyset\right) \\ &= \sigma + \theta x - \mu_\emptyset n - \mu_\emptyset x \\ &= (\theta - \mu_\emptyset)\cdot x + \sigma - \mu_\emptyset n \end{aligned}$$

As this is a linear function in x, the function $f^1(x)$ is strictly increasing for $\theta > \mu_\emptyset$ and strictly decreasing otherwise. Thus, the theorem holds for $a = 1$.

Second, we consider the case $(a \neq 1) \wedge (\sigma = \theta n)$, that is, the first n instances all had the same target value. In this case, the function $f^a(x)$ is given by $f^a(x) = (n+x)^a(\theta - \mu_\emptyset)$. This is strictly monotone since $n > 0, x > 0$. Thus, again $f^(x)$ has no local maximum.

Third, the case $(a \neq 1) \wedge (\sigma \neq \theta n)$ is considered in detail: Since σ was computed as a sum of n values that are at least as large as θ it can be assumed that $\theta \cdot n < \sigma$. In the following, the maxima of $f^a(x)$ is determined by deriving this function twice.

$$\begin{aligned} f^{a\,\prime}(x) = \frac{d}{dx} f^a(x) &= (n+x)^a \cdot \left(\frac{\sigma + x\cdot\theta}{n+x} - \mu_\emptyset\right) \\ &= (n+x)^a \left(\frac{d}{dx}\left(\frac{\theta x + \sigma}{n+x} - \mu_\emptyset\right)\right) + \left(\frac{\theta x+\sigma}{n+x} - \mu_\emptyset\right)\cdot\left(\frac{d}{dx}(n+x)^a\right) \\ &= (n+x)^a \left(\frac{d}{dx}\left(\frac{\theta x + \sigma}{n+x}\right)\right) + \left(\frac{\theta x+\sigma}{n+x} - \mu_\emptyset\right)\cdot a(n+x)^{a-1} \\ &= (n+x)^a \left(\frac{\theta}{n+x} - \frac{\theta x+\sigma}{(n+x)^2}\right) + \left(\frac{\theta x+\sigma}{n+x} - \mu_\emptyset\right)\cdot a(n+x)^{a-1} \\ &= (n+x)^{a-2}\left((\theta(n+x) - (\theta x+\sigma)) + a\left(\theta x + \sigma - \mu_\emptyset(n+x)\right)\right) \\ &= (n+x)^{a-2}(\theta n - \sigma + a\theta x + a\sigma - a\mu_\emptyset n - a\mu_\emptyset x)) \\ &= (n+x)^{a-2}\left((x(a\theta - a\mu_\emptyset) + a\sigma - a\mu_\emptyset n + \theta n - \sigma\right) \end{aligned}$$

In line 2, the product rule is used. In line 3 the chain rule is applied, substituting (n+x). $\mu_\emptyset$ can be omitted, as it is constant with respect to x. In line 4 the quotient rule is used. Finally, in line 5 $(n+x)^{a-2}$ is factored out.

Since $x > 0, n > 0$ by definition, the first factor is obviously greater than zero for any valid x. For $a = 0$ or $\theta = \mu_\emptyset$, the second factor of this function is independent from x, so it has no root, thus $f(x)$ has no maxima except the definition boundaries in this case. Otherwise the root of this function and therefore the only candidate for a maximum of $f^a(x)$ is given at the point

$$x^* = \frac{-a\sigma + an\mu_\emptyset - \theta n + \sigma}{a(\theta - \mu_\emptyset)}.$$

In the following, it is shown that x^* can not be a maximum value in our setting: For that purpose, the second derivative of $f(x)$ is computed at the point x^*:

$$\begin{aligned}
f^{a\,\prime\prime}(x) &= \frac{d}{dx} f'(x) \\
&= (n+x)^{a-3}(a-2)(x(a\theta - a\mu_\emptyset) + a\sigma - an + \theta n - \sigma) + (a\theta - a\mu_\emptyset)(n+x)^{a-2} \\
&= (n+x)^{a-3}((a-2)(x(a\theta - a\mu_\emptyset) + a\sigma - an\mu_\emptyset + \theta n - \sigma) + (a\theta - a\mu_\emptyset)(n+x)) \\
&= (n+x)^{a-3}(a^2x\theta - a^2x\mu_\emptyset + a^2\sigma - a^2\mu_\emptyset n + a\theta n - a\sigma - 2xa\theta + 2ax\mu_\emptyset - 2a\sigma \\
&\quad + 2an\mu_\emptyset - 2\theta n + 2\sigma + a\theta n - an\mu_\emptyset + a\theta x - ax\mu_\emptyset) \\
&= (n+x)^{a-3}(a-1)(a\theta x + a\sigma - an\mu_\emptyset - ax\mu_\emptyset + 2\theta n - 2\sigma) \\
&= (n+x)^{a-3}(a-1)(x(a\theta - a\mu_\emptyset) + a\sigma - an\mu_\emptyset + 2\theta n - 2\sigma)
\end{aligned}$$

We now can determine the second derivative of f in x^*:

$$\begin{aligned}
f^{a\,\prime\prime}(x^*) &= (n+x^*)^{a-3}(a-1)(x^*(a\theta - a\mu_\emptyset) + a\sigma - an\mu_\emptyset + 2\theta n - 2\sigma) \\
&= (n+x^*)^{a-3}(a-1)(\frac{-a\sigma + an\mu_\emptyset - \theta n + \sigma}{a(\theta - \mu_\emptyset)}(a\theta - a\mu_\emptyset) + a\sigma - an\mu_\emptyset + 2\theta n - 2\sigma) \\
&= (n+x^*)^{a-3}(a-1)(-a\sigma + an\mu_\emptyset - \theta n + \sigma + a\sigma - an\mu_\emptyset + 2\theta n - 2\sigma) \\
&= (n+x^*)^{a-3}(a-1)(\theta n - \sigma) \\
&= (n+x^*)^{a-3}(a-1)(\theta n - \sigma)
\end{aligned}$$

Since $f^a(x)$ is defined only for positive x, the first factor is always positive. Since by premise $a < 1$ and $\theta n < \sigma$, the second derivative at point x^* is always positive. Thus, if x^* is an extreme value of $f(x)$, then it is a local minimum. Since it was shown above that $f(x)$ has no other candidates for extreme values besides x^*, this proves the lemma.■

5 Efficient Exhaustive Exceptional Model Mining through Generic Pattern Trees[1]

The next chapter approaches the task of efficient exhaustive exceptional model mining [170]. In this variation of subgroup discovery, not the deviation of a single target attribute determines the interestingness value of a subgroup, but the deviation in the model parameters of a model that is derived from a set of model attributes, see Sections 2.4.3 and Section 2.5.3.4. For this task, *generic pattern trees (GP-trees)* are introduced. These are novel data structures that enable fast discovery algorithms.

Since for exceptional model mining the search space is identical to traditional subgroup discovery, enumeration orders can be directly transfered from this setting. Regarding pruning possibilities, upper bounds can be determined for specific interestingness measures. As an example, an interestingness measure, which uses a correlation coefficient model, could exploit the fact that the correlation coefficient never exceeds a value of 1. However, there is a wide variety of model classes and for each of these model classes different interestingness measures are applied. Since optimistic estimates bounds have to be determined separately for each interestingness measure, it is difficult to find generic improvements, which are applicable in most settings.

Therefore, efficient and generic *data structures* for exhaustive mining of exceptional models are the focus of this chapter in order to achieve runtime improvements for a large number of models classes and interestingness measures. In particular, the well-known FP-tree [111] data structure is significantly extended by replacing the frequency information stored in each node of the tree by the more general concept of valuation bases. Valuation bases are dependent on a specific model class and allow for efficient computation of the target model parameters. We call these generalized data structures *generic pattern trees*. They are used to derive an efficient novel algorithm, which is flexible: While the main search algorithm and core data structure is the same for all model classes, small parts of the data structure, that is, the valuation bases, can be plugged-in to adapt the algorithm to new model classes.

The contribution of this chapter is fourfold: First, the concept of valuation bases is presented. It allows for deriving a new algorithm, which is capable of performing efficient exhaustive mining for many different classes of exceptional models. Second, the scope of the presented approach is characterized by a theorem that determines, if GP-trees can be used for a certain model class or not. Third, instantiations of GP-trees are discussed

[1] This chapter is based on previously published work [173]: Florian Lemmerich, Martin Becker, and Martin Atzmueller. Generic Pattern Trees for Exhaustive Exceptional Model Mining. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2012.

for several important model classes, which have been presented in literature. Fourth, the presented approach is evaluated using publicly available UCI data [13]. In that direction, also a scalability study on a real world dataset is performed.

The chapter is organized as follows: After a short outline of related work in Section 5.1, the concept of valuation bases as well as the novel algorithm is presented in Section 5.2. Then, the limitations of the presented approach with respect to the applicability to different model classes is characterized in Section 5.3. Afterwards, Section 5.4 discusses instantiations of the presented generic algorithm by showing how different model classes presented in literature fit in the proposed framework. The effectiveness of the approach is evaluated in Section 5.5. Finally, the main results of this chapter are summarized in Section 5.6.

5.1 Related Work

Since the concept of exceptional model mining has been introduced only recently, very few generic algorithms have been proposed for that task. In the original paper introducing this concept, Leman et. al. propose to use a heuristic beam search strategy for the mining process [170]. Additional constraints on the pattern complexity and the minimum support are used to speedup the search. Van Leeuwen and Knobbe extended this approach in the DSSD algorithm to achieve a more diverse and non-redundant set of result patterns [242, 243]. Another approach by van Leeuwen is the EMDM algorithm [241]. This method follows a two-step approach: In the first step (EM step) a candidate subgroup is refined to maximize its exceptionality. A second step (DM step) aims at minimizing the description complexity of the candidate pattern by utilizing overfitting preventing mechanisms of a (rule-based) classifier. These steps are iterated until no more changes occur. In contrast to these heuristic algorithms, the generic algorithm proposed in this work performs an exhaustive search over the search space and thus guarantees optimal results.

For multi-label subgroup discovery, Mueller et al. present a subgroup discovery algorithm in a case study on breast cancer diagnosis as an improvement of Apriori-SD [198]. In addition to the original algorithm, their development *SD4TS* also utilizes pruning on subgroup support and interestingness measure-based pruning on the multi-label interestingness measure they use.

Atzmueller and Mitzlaff proposed the COMODO algorithm for descriptive community mining [19], which could be considered as a specific exceptional mining task. While this algorithm also features prefix trees as a data structures, it is only suited for a specific model class in this particular problem setting. In contrast, the approach presented here provides a generic algorithm which can be used for a wide range of model classes and interestingness measures.

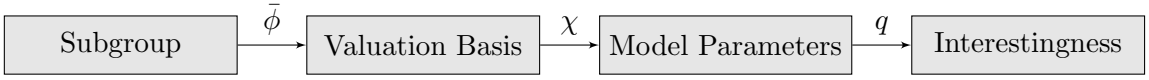


Figure 5.1: The pipeline visualizing the novel approach: For each subgroup (set of data instances) a valuation basis is derived using a function $\bar{\phi}$ (valuation projector). The model parameters for the chosen model class is extracted from these valuation bases using another function χ (model extractor). Model parameters are then used to determine the interestingness of the respective pattern using an exceptionality measure q.

5.2 GP-growth

In this section, a new approach on exhaustive exceptional model mining is presented. The novel method is called *generic pattern growth (GP-growth)*, cf. also [36]. GP-growth is based on the FP-growth algorithm substituting frequencies by an intermediate, condensed data representation called a valuation basis, which is introduced first. Then, the modifications of the traditional FP-growth algorithm that lead to the novel GP-growth algorithm are discussed.

5.2.1 The Concept of Valuation Bases

In the traditional FP-growth approach, frequencies are stored in the nodes of the tree, cf. Section 3.3.3. These nodes can then be aggregated to obtain frequencies for patterns (itemsets, subgroups) in the search space. The frequency is then used to rate the patterns determining those with high instance counts. In the proposed approach, this frequency count, which is stored in each node of the tree structure, is replaced by a more generic concept that we call a *valuation basis*.

A valuation basis is defined as a (condensed) representation of a set of data instances that is sufficient to extract the model parameters for a given model class. Consequently, the kind of information stored in a valuation basis is dependent on the model class. Since the interestingness of a subgroup in our setting is based on the model parameters only, it can be derived from such a valuation basis.

A visualization of the overall approach is given in Figure 5.1: For each subgroup (set of data instances) a valuation basis can be derived using a function $\bar{\phi}$ (*valuation projector*), e.g., by aggregating information on subsets of the subgroup. The model parameters for the chosen model class are extracted from these valuation bases using another function χ (*model extractor*). Model parameters are then used to determine the interestingness score of the respective pattern using an exceptionality measure q.

The data structure for the *SD-Map** algorithm from Chapter 4 can be seen as a simple example for this approach: Consider a model with a single model attribute X. The only model parameter is the mean value of X in all instances covered by the subgroup. Then, an appropriate valuation base can consist of the instance count and the sum of all values

of X of all instances of the subgroup. The instance count and the sum of values can be accumulated in an FP-tree like structure. Given the accumulated valuation basis for each pattern, the stored instance count and the value sum are used to compute the mean. The actual interestingness value of the pattern can then be determined using this mean value, e.g. as the deviation from the mean value in the total population.

Please note that one can construct a trivial type of valuation bases that defines a valuation basis as the exact same set of data instances it represents (restricted to the model attribute values). That is, in each tree node a complete list with references to all instances is stored. Obviously, this most general type of valuation bases trivially contains all relevant information associated with the original set of data instances. Therefore, model parameters for any model class on the original set of data instances can be derived from this type of valuation bases. However, while this trivial kind of valuation basis allows for a generally applicable approach, the main advantages in terms of memory and runtime performance are lost. Therefore, one aims to construct valuation bases for a given model class which are as small as possible. A valuation basis is called *condensed valuation basis* if its memory requirement is sublinear with respect to the number of instances it represents. The examples of valuation bases, which are presented in this work, use constant memory with respect to the instance count.

The possibility of aggregating valuation bases corresponding to sets of data instances is modeled by *valuation domains*:

Definition 4 (Valuation Domain, Valuation Basis) A valuation domain is an abelian semi-group $\mathbb{V} = (V, \oplus)$, where V is an arbitrary set and $\oplus$ is a binary operator on V, i.e. $\oplus : V \times V \to V$ and

- V is closed under $\oplus$, i.e. $a, b \in V \Rightarrow a \oplus b \in V$
- $\oplus$ is associative, i.e. $a, b, c \in V \Rightarrow a \oplus (b \oplus c) = (a \oplus b) \oplus c$
- $\oplus$ is commutative, i.e. $a, b \in V \Rightarrow a \oplus b = b \oplus a$

An element $v \in V$ is called a *valuation basis*. □

In order to derive valuation bases from data instances, we define the notion of a *valuation projector* ϕ as in Definition 5.

Definition 5 (Valuation Projector) Let $\mathcal{I}$ be a set of all data instances and let $\mathbb{V} = (V, \oplus)$ be a valuation domain. Then a valuation projector is defined as

$$\phi : \mathcal{I} \to V$$ □

Given the definition of valuation domains above, the *valuation projector* ϕ can be naturally extended to sets of data instances $S \in 2^{\mathcal{I}}$:

$$\begin{aligned} \bar{\phi} : 2^{\mathcal{I}} &\to V \\ S &\mapsto \bigoplus_{s \in S} \phi(s) \end{aligned}$$

As a result, for any disjunct pair of sets of data instances $S' \cap S'' = \emptyset$ it holds that:

$$\bar{\phi}(S' \cup S'') = \bar{\phi}(S') \oplus \bar{\phi}(S'')$$

Sometimes subgroups are evaluated by comparing the values of the model attributes derived in the subgroup and the complement of the subgroup. In order to handle this case efficiently, an additional subtraction operator $\ominus$ is required for the valuation domain: Let $\mathcal{I}$ be the set of all instances and let $I'' \cup I''$ be an arbitrary partition of $\mathcal{I}$. If the valuation basis of I' is subtracted from the valuation basis of $\mathcal{I}$, then the result must be the valuation basis of I'': $\bar{\phi}(I) \ominus \bar{\phi}(I') = \bar{\phi}(I'')$. Thus, the valuation basis for the complement can easily be derived by subtracting the subgroup's valuation basis from the overall valuation basis. The overall valuation basis can easily be computed in an initial pass over the dataset.

5.2.2 Algorithmic Adaptations

Essentially, the FP-growth algorithm (cf. Section 3.3.3 for a detailed description) is generalized by substituting frequencies with the more general concept of valuation bases. The resulting algorithm is called *GP-growth*. The generalized tree structure that stores valuation bases instead of frequencies is called a *GP-tree*.

Whereas the FP-growth algorithm adds up frequencies in its FP-trees, the GP-growth algorithm aggregates valuation bases in its GP-trees. Hence, GP-growth produces aggregated valuation bases instead of frequencies for each pattern. Note that a valuation basis can also contain a frequency as described in the mean value example in Section 5.2.1. An illustration of this approach is given in Figure 5.2.

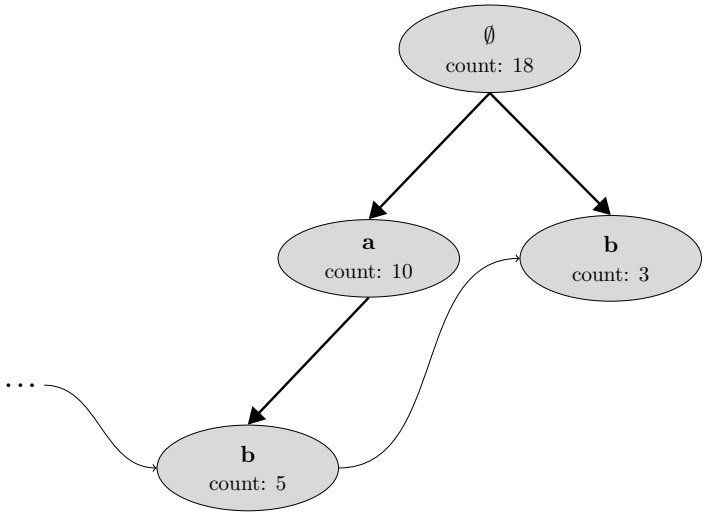
In order to aggregate valuation bases for each pattern, the algorithm requires

- a valuation domain $\mathbb{V} = (V, \oplus)$ to draw valuation bases from,
- a valuation projector ϕ to project single data instances onto valuation bases, and
- optionally a subtraction operator $\ominus$ to support complement comparisons as mentioned in Section 5.2.1 if the statistics for the complement are required by the utilized exceptionality measure.

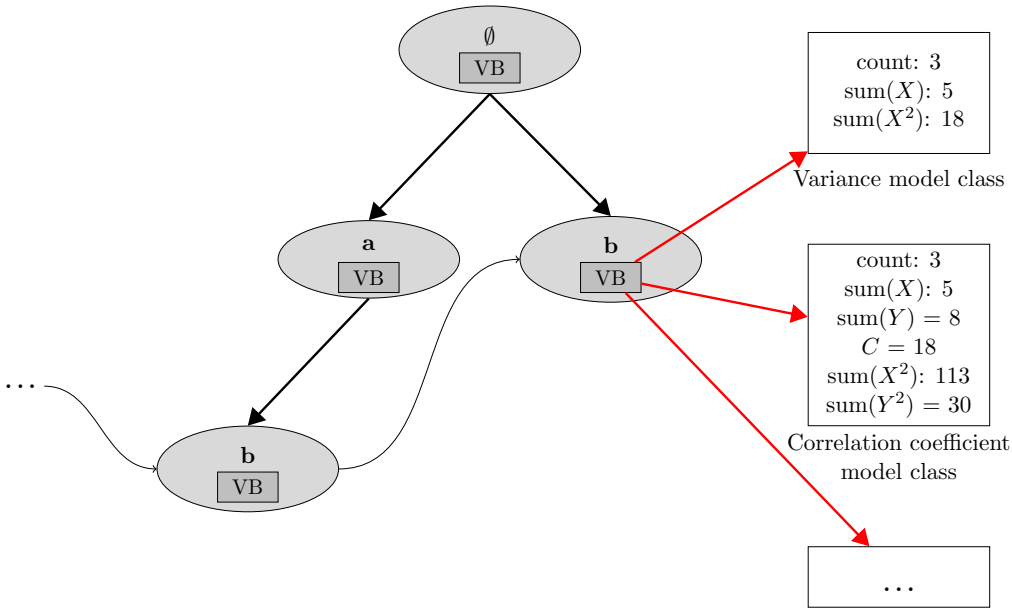
Patterns are then evaluated based on their valuation bases by applying

- a model extractor χ to map valuation bases $v \in V$ onto model parameters, and
- an exceptionality measure q based on these model parameters.

These adaptations of FP-growth allow for a generic implementation. That is, the code for the main algorithm is identical for all model classes. To apply it to a new model class, only the valuation domain with its aggregation operator $\oplus$, the corresponding valuation projector ϕ, and the model extractor χ must be implemented. We call the tuple $(V, \oplus, \phi, \chi)$ a *model configuration*.



(a) Exemplary tree structure of a traditional FP-tree



(b) Exemplary tree structure of a GP-tree, the novel data structure introduced in this work

Figure 5.2: An example for the generalization from FP-trees to GP-trees: The tree structure and the auxiliary links that connect nodes for same selector remains identical, but the information stored in each node differs. In traditional FP-trees only the count for the corresponding set of instances is saved. In contrast, in GP-trees the nodes store *valuation bases*, which contain more information. The kind of information that is saved in the valuation bases depends on the chosen model class of the mining task. Thus, the main data structure is the same for all model classes, but the valuation bases can be exchanged in order to adapt to novel model classes.

Please note, that for the very simple valuation basis, which only counts the instances, the resulting algorithm is identical to FP-growth. Furthermore, traditional subgroup discovery can be implemented in this generic algorithm by using valuation bases that count instances with a positive and a negative target concept separately, as done in the SD-Map algorithm [22]. Thus, the approach presented in this work can be regarded as a true generalization of both, FP-growth [111] and SD-Map. The *SD-Map** algorithm is limited to the setting with a single numeric target variable, but does additionally provide efficient, specialized pruning options. In the future, pruning could also be incorporated in the GP-growth algorithms, if the respective optimistic estimate bound are derived for the utilized interestingness measures. Unfortunately, these have not yet been researched for exceptional model mining.

5.3 Theorem on Condensed Valuation Bases

The approach of generalized FP-trees is especially efficient if it is possible to derive a small condensed valuation basis for a model class. If the constructed model itself is very complex, then it seems difficult to derive suitable condensed valuation bases that are sufficient to extract the model parameters. This includes, for example, computationally expensive models that involve the learning of a bayesian network as explored in [73].

In order to define the scope of the presented approach, the following theorem characterizes model classes, for which GP-trees can be applied with strongly reduced memory requirements. This is done by drawing a parallel to data stream mining:

Theorem 1 *There is a condensed valuation domain for a given model class if and only if the following conditions are met: (1) There is a* parallel single-pass algorithm *with sublinear memory requirements to compute the model parameters from a given set of instances, which are distributed randomly on one of the (parallel) computation nodes; (2) the only communication between the computation nodes in this algorithm takes place when combining results.* □

Proof $\Rightarrow$: First, assume there is a model configuration $(V, \oplus, \phi, \chi)$ that can be used to determine the model parameters of a subgroup. Then, a parallel single-pass algorithm can be constructed as following: In each computation node N one loops through all instances I_N assigned to this node, updating the respective valuation basis v_N. For each instance $c \in I_N$, the valuation basis $\phi(c)$ is extracted and used to update the current accumulated valuation basis: $v_N^{new} = v_N^{old} \oplus \phi(c)$. Thus, after each step the valuation basis v_N corresponds to all instance handled so far. After the loop through all instances of this computation node, the valuation basis v_N can be used to extract the model parameters for the set of instances I_N. Furthermore, the resulting valuation bases from different computation nodes can be combined by using the aggregation operator $\oplus$ again. This leads to a valuation basis that corresponds to all instances of the dataset. The model parameters are then extracted using the model extractor χ; this completes the parallel single-pass algorithm for computing these parameters.

$\Leftarrow$: Assume there is a parallel single-pass algorithm with the properties presented above. Then, there is a set of variables V_C that are used in the computation within each of the nodes, which is sublinear with respect to the number of contained instances. It is shown that this set of variables defines a model configuration $(V, \oplus, \phi, \chi)$. Since the algorithm is single-pass, the assignments for these variables are updated only once for each instance using the values of the model attributes for this instance. Now let v_i be the vector of values (variable assignments) of the variables V_C after the instance i is processed as the *first* instance in this computation node. Then, one can use this vector as the projector function $\phi(i) = v_i$. This is sufficient for a valuation basis; if there was only one instance in the dataset, then a correct algorithm would be able to extract the model parameters for the model built from the single instance using only the data v_i.

Next, assume that each computational node N_j has finished the computation of its partition of the data D_j, each resulting in variable assignments v_j, which correspond to a valuation basis. v_j must be sufficient to extract the model parameters for the data D_j since N_j could be the only computational node. The method used for this subtask can be regarded as a model extractor function χ. Now consider two valuation bases v_1 and v_2 that result from two computation nodes and are corresponding to data partitions D_1 and D_2. A correct parallel algorithm must come with an appropriate method to combine the results v_1 and v_2 into new variable assignments that is suited to extract model parameters for the data $D_1 \cup D_2$. This method can be used as a general aggregation function $\oplus$ for valuation bases. Thus, given a parallel single-pass algorithm with the properties presented above one can derive a model configuration $(V, \oplus, \phi, \chi)$.■

The proof is constructive. It describes a method to transfer parallel single-pass algorithms for specific model classes to valuation domains that can be used for efficient exceptional model mining with GP-trees. Some important examples of this approach are shown in the next section.

5.4 Valuation Bases for Important Model Classes

This section discusses the application of GP-trees to different model classes. Most of the presented model classes have been proposed in [170], to which it is referred for a more detailed description of the models and exceptionality measures.

5.4.1 Variance Model

The variance model identifies patterns, in which the variance of a single target variable X is especially high/low. Although this model features only a single model attribute, this task can not be accomplished by traditional subgroup discovery algorithms utilizing FP-trees, such as SD-Map.

For an efficient computation of the variance, the following well known formula is utilized:

$$Var(X) = E[X^2] - E[X]^2 = \frac{\sum x^2}{n} - (\frac{\sum x}{n})^2,$$

where $E[X]$ is the expected value for the variable X. For computing the variance of an attribute, only the total count, the sum of all values and the sum of all squared values are required. Formally, a model configuration $(V_\sigma, \oplus_\sigma, \phi_\sigma, \chi_\sigma)$ that is sufficient to compute the variance (or equivalently, the standard deviation) of a variable X can be defined as:

$$\begin{aligned} V_\sigma &= \mathbb{R}^3 \\ v \oplus_\sigma u &= v + u \\ \phi_\sigma(c) &= (1, X(c), X(c)^2)^T \\ \chi_\sigma(v) &= \frac{v_3}{v_1} - (\frac{v_2}{v_1})^2 \end{aligned}$$

Each valuation basis stores a vector of three real numbers. Aggregating valuation bases using the operator $\oplus$ is equivalent to adding vectors in euclidean space. The valuation basis extracted from a single data instance c contains the constant 1 as the instance count, the value of X in c and the squared value of X in c. To extract the model parameter $Var(X)$ from a valuation basis $v \in V_\sigma$, the computation χ_σ has to be performed using the three components of the vector stored in the valuation basis v.

Example 13 Assume that in the mining algorithm a new tree node is created for an instance i_1, where the single model attribute X takes the value $X(i_1) = 5$. Like any tree node in this settings, the new node stores three values, in this case 1, 5, and $5^2 = 25$. Now assume another node v in the tree structure corresponds to three instances with values $X(i_2) = 2$, $X(i_3) = 3$, and $X(i_4) = 10$. Thus, the values in v are 3, 15, and 113. In the mining algorithm tree nodes are aggregated to create new nodes, which then correspond to the union of instances of the single nodes. Assume that both nodes described above are aggregated. Then the values stored in the novel node are determined as: $(1, 5, 25)^T + (3, 15, 113)^T = (4, 20, 138)^T$. The variance for the four instances, which correspond to this new node can be computed as: $\frac{138}{4} - (\frac{20}{4})^2 = 34.5 - 25 = 9.5$. □

5.4.2 Correlation Model

The (Pearson product-moment) correlation coefficient $\rho(X, Y)$ is a very well known statistical measure that reflects the linear dependency between two numerical attributes X and Y. The correlation coefficient is defined as the fraction of the covariance and the product of the standard deviations of these two attributes: $\rho(X, Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}$.

We will first derive a model configuration for the covariance. This will then be utilized to derive a configuration for the correlation model. The covariance of a set of instances is defined as $Cov_S(X, Y) = \frac{\sum_{(x,y)}(x-\mu_X)(y-\mu_Y)}{N}$, where μ_X (μ_Y) describes the mean value of the attribute X (Y) in the instance set S and N the number of instances in S. In the valuation domain for GP-trees, which is derived here, the measures $Cov(X, Y)$, σ_X and σ_Y are determined independently from each other.

To efficiently compute the covariance of two variables X, Y a pairwise update formula was introduced in [38] for a parallel single-pass algorithm. It allows to compute $C_S(X,Y) = Cov_S(X,Y) \cdot |S| = \sum_{(x,y)\in S}(x-\mu_X)(y-\mu_Y)$ for a set of data instances $S = S_1 \cup S_2, S_1 \cap S_2 = \emptyset$ given statistical information of the partitioning sets S_1 and S_2:

$$C_S(X,Y) = C_{S_1}(X,Y) + C_{S_2}(X,Y) + \frac{i_{S_1} i_{S_2}}{i_{S_1} + i_{S_2}}(\mu_{X,2} - \mu_{X,1})(\mu_{Y,2} - \mu_{Y,1}),$$

where $i_{S_1} = |S_1|$ and $i_{S_2} = |S_2|$ denote the instance count in S_1 and S_2, and $\mu_{X,2}$, $\mu_{X,1}$, $\mu_{Y,2}$, $\mu_{Y,1}$ are the mean values of X and Y in the sets S_1 and S_2. According to Theorem 1, this update formula can be used to construct a model configuration for the correlation model.

To compute the covariance for each subgroup with this formula, one needs to keep track of the value of $C_S(X,Y)$ in the respective set of instances, the cardinality (number of instances) of the subgroup, and the mean values of the variables X and Y. The latter can be computed by the sum of the values for the respective attributes and the cardinality of the subgroup. Therefore, a model configuration for the covariance can be defined by using the above formula:

$$\begin{aligned}
V_{cov} &= \mathbb{R}^4 \\
v \oplus_{cov} u &= \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} \oplus \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} v_1 + u_1 \\ v_2 + u_2 \\ v_3 + u_3 \\ v_4 + u_4 + \frac{v_1 u_1}{v_1 + u_1}(\frac{v_2}{v_1} - \frac{u_2}{u_1})(\frac{v_3}{v_1} - \frac{u_3}{u_1}) \end{pmatrix} \\
\phi_{cov}(c) &= (1, X(c), Y(c), 0)^T \\
\chi_{cov}(v) &= \frac{v_4}{v_1}
\end{aligned}$$

In this formalization of a model configuration, the first component of a valuation basis reflects the size of the corresponding set, the second and third component store the sum of the values of X and Y and the fourth component keeps track of the measure C as defined above.

To compute the actual correlation coefficient this valuation basis is combined with the valuation basis used for the variance model in order to compute $Cov(X,Y)$, σ_X and σ_Y in a single model configuration:

$$V_{cor} = \mathbb{R}^6$$

$$v \oplus_{cor} u = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix} \oplus \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{pmatrix} = \begin{pmatrix} v_1 + u_1 \\ v_2 + u_2 \\ v_3 + u_3 \\ v_4 + u_4 + \frac{v_1 u_1}{v_1 + u_1}(\frac{v_2}{v_1} - \frac{u_2}{u_1})(\frac{v_3}{v_1} - \frac{u_3}{u_1}) \\ v_5 + u_5 \\ v_6 + u_6 \end{pmatrix}$$

$$\phi_{cor}(c) = (1, X(c), Y(c), 0, X(c)^2, Y(c)^2)^T$$

$$\chi_{cor}(v) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\frac{v_4}{v_1}}{\sqrt{\frac{v_5}{v_1} - (\frac{v_2}{v_1})^2}\sqrt{\frac{v_6}{v_1} - (\frac{v_3}{v_1})^2}} = \frac{v_1 v_4}{\sqrt{v_1 v_5 - v_2^2}\sqrt{v_1 v_6 - v_3^2}}$$

Example 14 Consider model attributes X and Y and the small dataset shown in Table 5.1: When mining for exceptional correlation models, each tree node in the generic

Table 5.1: A small example dataset with values for the model attributes X and Y.

instance	X	Y
i_1	5	3
i_2	2	1
i_3	3	2
i_4	10	5

pattern tree stores six values. When a new tree node u is created for i_1, it stores the values using the function $\phi_{cor}(c)$: 1, $X(i_1) = 5$, $Y(i_1) = 3$, 0, $X(i_1)^2 = 25$, and $Y(i_1)^2 = 9$. Another node v in the tree structure corresponds to three instances i_2, i_3, i_4. To compute the respective stored values is slightly more complicated, they turn out to be $(3, 15, 8, 18, 113, 30)^T$ in vector notation. Assume that both nodes described above are aggregated to create a new node, which then correspond to the union of instances of the respective single nodes, that is, i_1, i_2, i_3, and i_4. Then the six values stored in the novel node are determined by the aggregation operation $\oplus_{cor}$: $1 + 3 = 4$, $5 + 15 = 20$, $3 + 8 = 11$, $0 + 18 + \frac{1 \cdot 3}{1+3}(\frac{5}{1} - \frac{15}{3})(\frac{3}{1} - \frac{7}{3}) = 18$, $25 + 113 = 138$, and $9 + 30 = 39$. The correlation coefficient for these four instances is then computed based on the information stored in the node with the function χ_{cor}: $\frac{4 \cdot 18}{\sqrt{4 \cdot 138 - 20^2}\sqrt{4 \cdot 39 - 11^2}} \approx 0.987$. □

5.4.3 Linear Regression Model

The simple linear regression model is perhaps the most intuitive statistical model to show the dependency between two numeric variables X and Y. It is built by fitting a straight line in the two dimensional space by minimizing the squared residuals e_j of the model:

$$y_j = a + bx_j + e_j$$

As proposed in [170], the difference of the slope b of this line in a subgroup and the total population (or the complement of the subgroup within the population) can be used to identify interesting patterns. As known from statistics, the slope b can be computed by the covariance of both variables and the variance of X:

$$slope(X,Y) = \frac{Cov(X,Y)}{Var(X)}$$

Thus, a model configuration is given by combining the valuation domains for the variance and the covariance, similar to the correlation model:

$$\begin{aligned}
V_{slope} &= \mathbb{R}^5 \\
v \oplus_{slope} u &= \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{pmatrix} \oplus \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix} = \begin{pmatrix} v_1 + u_1 \\ v_2 + u_2 \\ v_3 + u_3 \\ v_4 + u_4 + \frac{v_1 u_1}{v_1 + u_1}(\frac{v_2}{v_1} - \frac{u_2}{u_1})(\frac{v_3}{v_1} - \frac{u_3}{u_1}) \\ v_5 + u_5 \end{pmatrix} \\
\phi_{slope}(c) &= (1, X(c), Y(c), 0, X(c)^2)^T \\
\chi_{slope}(v) &= \frac{Cov(X,Y)}{Var(X)} = \frac{\frac{v_4}{v_1}}{\frac{v_5}{v_1} - (\frac{v_2}{v_1})^2} = \frac{v_1 v_4}{v_1 v_5 - v_2^2}
\end{aligned}$$

Here, again the first four components represent the cardinality of the corresponding set of instances, the sum of values for X and Y and the measure C_S as defined above. The last component is additionally required to compute the variance as described previously for the variance model.

5.4.4 Logistic Regression Model

Next, the logistic regression model is considered. This model is used for the classification of a binary target attribute $Y \in A_M$ from a set of independent binary attributes $X_j \in A_M \setminus Y, j = 1, \ldots, |A_M| - 1$. The model is given by: $y = \frac{1}{1+e^{-z}}, z = b_0 + \sum_j b_j x_j$. An exceptional model mining goal could then be identify patterns in which the model parameters b_j differ significantly from the ones derived from the total population.

Unfortunately, to the author's knowledge no exact single-pass algorithm has been proposed yet for determining the parameters for logistic regression due to the non-linear

nature of parameter fitting. Since according to Theorem 1 the existence of such an algorithm is necessary for the existence of a sufficient condensed valuation basis, the efficient computation of parameters is not possible so far.

5.4.5 DTM-Classifier

The next discussed model is based on the DTM-classifier [152]: This method predicts a target attribute $Y \in A_M$ from a set of independent attributes $X_j \in A_M \setminus Y, j = 1, \ldots, |A_M| - 1$ by determining the probability of each target attribute value for each combination of values of the independent attributes. For value combinations, which did not occur in the training set, the probability distribution of the complete training set is used. Then, for a given new instance i the target value is predicted that has the highest probability conditioned on the respective combination of values of the X_j in i. If the specific combination of the values of X_j did not occur in the training data, then the instance is classified as the most frequent target value in the complete training set. In the context of exceptional model mining, amongst others, the *Hellinger distance* has been proposed as an exceptionality measure for this model class. It measures the difference between the distribution within a subgroup P and the distribution in its complement $\neg P$. It is computed as

$$\sum_{y,x_1,\ldots x_k} \left(\sqrt{\mathbf{P}_P(y|x_1,\ldots,x_k)} - \sqrt{\mathbf{P}_{\neg P}(y|x_1,\ldots,x_k)}\right)^2$$

For an efficient computation in FP-trees, the probabilities for all value combinations of $Y, X_1, \ldots, X_{|A_M-1|}$ are stored in the valuation basis.

In the following, all attributes are assumed to be binary for the sake of simpler notation. The generalization for non-binary attributes is straightforward. Furthermore, it is assumed that the combinations of values in the decision table are arranged in a predefined order, such that the positive values of Y are on odd positions and the corresponding negative value of Y for the same combination of independent attribute values is immediately following. Then, a slightly simplified model configuration is given by:

$$\begin{aligned}
V_{dtm} &= \mathbb{R}^{2^{|A_M|}} \\
v \oplus_{dtm} u &= v + u \\
\phi_{dtm}(c) = (v_j) &= \begin{cases} 1, & \text{if the } j\text{-th combination of values is true in } c \\ 0, & \text{else} \end{cases} \\
\chi^{(k)}_{dtm}(v) &= \frac{v_{2k-1}}{v_{2k-1} + v_{2k}}
\end{aligned}$$

In this model configuration, an index position is computed for each combination of values. The valuation basis then stores the frequency counts for these combination of values using these indices. Please note that in this case the valuation basis stored in each node of the GP-tree needs to store 2^m values, where m is the number of (binary)

model attributes. Therefore this model configuration is not tractable for large numbers of model attributes. However, this should not be the case in most practical applications since larger models are typically difficult to comprehend by human users.

5.4.6 Accuracy of Classifiers

One application of exceptional model mining could be, to identify subgroups, in which the classification model learned from the instances of a subgroup is especially accurate/inaccurate for these instances. Assuming that the decision of the classifier is based on all learning instances, the generalized FP-tree approach is *not* applicable for this task:

The computation of the accuracy of a classifier requires two passes through the database. The first pass is used to determine the model parameters for the classifier. Then, a second pass over the instances is required to classify the instances. According to Theorem 1, it is therefore not possible to construct a suitable condensed valuation basis.

Please note that as a consequence the presented approach allows to efficiently find patterns, in which the classification model built from the instances of this pattern differs strongly from the model built from the entire population, but it is not possible to identify patterns, in which this model performs exceptionally well or bad.

5.4.7 Bayesian Networks

Bayesian networks have been proposed as complex target models for exceptional model mining [73]. Since to the best of the author's knowledge there is currently no parallel single-pass algorithm for learning bayesian networks – which is a complex task on its own – a condensed valuation basis for this model class cannot be provided here. However, there is ongoing research in that area [50], which can possibly be exploited in future work.

5.5 Evaluation

This section presents runtime evaluations of the proposed approach using publicly available UCI-datasets [13] as well as a scalability study in a large real world dataset.

5.5.1 Runtime Evaluations on UCI data

The GP-tree approach was evaluated by performing runtime experiments using publicly available UCI datasets. The algorithms were implemented in the subgroup discovery environment *VIKAMINE 2*, see Chapter 8. The experiments were performed on a standard office PC with a 2.2 GHz CPU and 2 GB RAM. The search space consisted of the non-model attributes in the respective dataset. For nominal attributes, attribute-value pairs were used as selectors. Numeric attributes were discretized into five intervals by using equal-frequency discretization. Only intervals between two adjacent cutpoints were used as selectors.

Table 5.2: Runtime in seconds for the GP-growth algorithm using the *credit-g* dataset for different model classes and various search depth (maximum number of selectors in a single subgroup description).

Model Class	2	3	4	5
Frequent Itemsets	0.7	3.7	17.5	70.0
Subgroup Discovery	0.7	3.6	17.3	68.2
Variance	0.8	3.7	17.4	70.7
Correlation Coefficient	0.8	4.0	19.2	77.3
Linear Regression	0.8	4.0	19.1	78.2
DTM-classifier	1.6	8.4	44.0	187.4

Table 5.3: Runtime in seconds for the GP-growth algorithm using the *adults* dataset for different model classes and various search depth (maximum number of selectors in a single subgroup description).

Model Class	2	3	4	5
Frequent Itemsets	4.7	8.5	19.8	50.4
Subgroup Discovery	4.7	8.4	19.6	48.7
Variance	4.7	8.5	19.8	50.1
Linear Regression	4.8	8.9	21.8	55.8
Correlation Coefficient	4.8	8.9	21.7	55.7
DTM-classifier	15.1	36.7	90.2	213.7

Table 5.4: Runtime in seconds for the GP-growth algorithm using the *autos* dataset for different model classes and various search depth (maximum number of selectors in a single subgroup description).

Model Class	2	3	4	5
Frequent Itemsets	0.4	3.0	17.1	73.4
Subgroup Discovery	0.4	2.9	16.3	72.3
Variance	0.4	3.0	17.0	74.2
Linear Regression	0.4	3.2	18.1	84.3
Correlation Coefficient	0.4	3.2	18.0	81.1
DTM-classifier	0.7	5.2	31.8	150.7

Table 5.5: Runtime in seconds for different UCI-Datasets for the *linear regression* model class for various search depth (maximum number of selectors in a single subgroup description), comparing a simple depth-first-search (DFS) with the GP-growth algorithm (GPG).

Max depth	2		3		4		5	
Dataset	DFS	GPG	DFS	GPG	DFS	GPG	DFS	GPG
adults	218.3	4.8	5481.2	8.9	>2 h	21.8	>2 h	55.8
autos	1.4	0.4	42.7	3.2	632.8	18.1	5416.9	84.3
breast-w	0.1	0.0	0.9	0.0	3.0	0.1	6.6	0.1
census-kdd	>2 h	227.0	>2 h	1139.9	>2 h	7418.9	>2 h	>2 h
colic	1.4	0.8	33.5	2.3	428.5	10.9	3009.8	39.2
credit-a	1.2	0.2	16.3	0.8	129.4	2.8	654.5	8.2
credit-g	3.3	0.8	73.2	4.0	975.6	19.1	>2 h	78.6
diabetes	0.3	0.1	2.2	0.1	10.2	0.2	28.4	0.3
forestfires	0.8	0.1	11.8	0.5	93.3	1.6	378.6	3.6
glass	0.1	0.0	0.8	0.1	3.6	0.1	8.6	0.2
heart-h	0.2	0.1	2.1	0.1	10.9	0.3	34.1	0.5
hepatitis	0.2	0.1	3.6	0.6	35.4	2.8	219.5	10.2
hypothyroid	11.4	1.3	261.7	4.6	3791.7	21.8	>2 h	111.0
ionosphere	3.1	1.5	145.1	18.9	3625.0	162.9	>2 h	1083.2
labor	0.1	0.0	0.7	0.1	3.6	0.2	10.7	0.5
segment	6.6	0.9	156.7	3.5	2153.9	15.3	>2 h	58.2
spambase	19.3	4.3	627.7	20.9	>2 h	146.0	>2 h	1438.9
vehicle	2.4	0.5	57.4	2.5	784.3	12.1	5956.8	43.1
vowel	2.0	0.3	37.6	1.0	374.8	3.5	1808.0	8.5

In a first set of experiments the runtime of the GP-tree approach was compared for different model classes. For better comparability the two numeric attributes per dataset, which were used as model attributes for the correlation coefficient and the linear regression models, were also excluded from the search space for the frequent itemsets, subgroup discovery and variance model classes. In doing so, the search spaces for these tasks were identical. Exemplary results for the datasets *credit-g*, *adults* and *autos* are shown in Tables 5.2, 5.3 and 5.4. Further experiments on other datasets showed the same overall characteristics.

As can be seen in these tables, the algorithm runtimes for different model differ only little. The runtimes for the model classes linear regression and correlation coefficient, which require slightly larger valuation bases, are increased marginally in comparison to the simpler model class, e.g., the frequent itemset model. For the DTM-classifier, the results only differ in our implementation by a small constant factor (about 2-4), which can explained by the required more complex model configuration. The similarity of the runtimes is due to the fact that no pruning scheme is utilized. Consequently, the search

Table 5.6: Runtime in seconds for different UCI-Datasets for the *correlation coefficient* model class for various search depth (maximum number of selectors in a single subgroup description), comparing a simple depth-first-search (DFS) with the GP-growth algorithm (GPG).

Max depth	2		3		4		5	
Dataset	DFS	GPG	DFS	GPG	DFS	GPG	DFS	GPG
adults	173.0	4.8	4794.9	8.9	0.0	21.7	>2 h	55.7
autos	1.2	0.4	37.9	3.2	590.7	18.0	5125.4	81.1
breast-w	0.1	0.0	0.7	0.0	2.5	0.1	5.7	0.1
census-kdd	>2 h	226.4	>2 h	1144.4	>2 h	7409.6	>2 h	>2 h
colic	1.1	0.7	25.2	2.2	335.7	10.8	2419.0	38.0
credit-a	0.9	0.2	13.0	0.8	106.0	2.8	553.7	8.2
credit-g	2.5	0.8	58.1	4.0	814.3	19.2	6975.3	77.3
diabetes	0.2	0.0	1.7	0.1	8.8	0.2	25.2	0.3
forestfires	0.6	0.1	9.8	0.5	82.4	1.6	345.1	3.6
glass	0.1	0.0	0.7	0.1	3.2	0.1	7.9	0.2
heart-h	0.2	0.1	1.6	0.1	8.6	0.3	27.9	0.5
hepatitis	0.2	0.1	2.7	0.6	27.8	2.8	177.9	10.3
hypothyroid	8.4	1.3	200.3	4.6	3002.9	21.1	>2 h	108.5
ionosphere	2.5	1.6	121.3	20.0	3272.3	161.1	>2 h	1053.0
labor	0.1	0.0	0.6	0.1	2.8	0.2	8.3	0.5
segment	5.2	0.9	129.3	3.5	1878.3	15.4	>2 h	58.9
spambase	14.2	4.3	473.0	20.8	$< 2h$	148.0	>2 h	1459.4
vehicle	1.8	0.5	46.8	2.6	680.1	12.3	5324.8	43.2
vowel	1.6	0.3	32.2	1.0	343.4	3.5	1716.6	8.4

space is the same for all model classes and also the number of tree nodes and number of required aggregations of tree nodes is identical.

Next, an extensive runtime analysis of our approach was performed using 19 datasets from the UCI repository. For that purpose, the proposed GP-growth algorithm was compared to a simple depth-first-search without any specialized data structure. Due to the runtime similarity for different model classes, the presented results are limited to two exemplary model classes, that is, the slope of the linear regression and the correlation coefficient. The results are shown in Tables 5.5 and 5.6.

It can be observed that even at a search depth of 2 (searching only for subgroup descriptions that have a maximum of 2 selectors) the GP-growth algorithm outperforms the simple depth-first-search approach significantly. This difference increases for larger search depth. At a search depth of 5, GP-growth completes the task more than an order of magnitude faster for all datasets. These results clearly demonstrate the power of efficient data structures such as the GP-tree.

5.5.2 Scalability Study: Social Image Data

In the following, a short case study on real world data is presented that demonstrates the advantages of the presented approach in large scale applications. As a dataset, publicly available metadata of pictures uploaded to the Flickr[2]-platform was used. More specifically, the view counts as well as all tagging information were crawled of all pictures which are geo-referenced to a location in Germany and were uploaded in 2010. Details on this dataset are provided in Section 9.2. The dataset was limited to tags with more than 1000 occurrences. This lead to about 1200 tags that were used as describing attributes for about 1.1 million instances.

Since pictures viewed by more people are naturally also tagged by more people, there is a correlation to the number of tags assigned to a picture. To evaluate the scalability of the GP-growth approach, an exceptional model mining task was executed to identify combinations of tags (as subgroup descriptions), for which this correlation is especially strong. As a result, even for a search depth of 2, the simple DFS algorithm did not finish the task within two full days. In contrast, the same task performed by GP-growth finished in about 8 minutes.

The massive difference for this dataset can be explained by the sparseness of the tagging data, which especially favors the utilized tree structure. Furthermore, even for an increased search depth of 3, the task could be completed within 10 minutes. This small difference in comparison to a search depth of 2 is reasonable, as less combinations of three tags occur in dataset than combinations of two tags due to the sparseness of the dataset. Overall, the runtime improvements for the Flickr dataset are even larger than in the previously investigated smaller datasets, showing the scalability of our approach.

5.6 Summary

This chapter proposed a novel approach for fast exhaustive exceptional model mining: It introduced the concept of valuation bases as an intermediate condensed data representation and presented the generic GP-growth algorithm. This algorithm transfers the FP-tree data structure to the setting of exceptional model mining in order to allow for efficient exhaustive mining for a wide variety of model classes. The applicability of the proposed approach was discussed in detail: Model classes, for which the algorithm is applicable, can be identified by a theorem that draws an analogy to data stream mining. For several model classes the implementations of the generic approach have been presented. Runtime experiments show improvements of more than an order of magnitude of the novel approach in comparison to a naive exhaustive depth-first-search.

[2] www.flickr.com

6 Difference-based Estimates for Generalization-Aware Subgroup Discovery[1]

The next chapter introduces a novel approach to derive optimistic estimates for generalization-aware interestingness measures. In contrast to previous approaches the bounds are not only based on the anti-monotonicity of instances that are contained within the subgroup. Instead, also the number of instances that are covered by a pattern, but not by its generalizations are considered. This allows for significantly tighter optimistic estimate bounds and thus faster subgroup discovery when using generalization-aware measures.

For motivation, the problem setting is recapitulated in short, for more details see Section 2.5.4: The selection of subgroups is commonly based on an interestingness measure. These measures use statistics derived from the instances covered by a subgroup to determine its score. The best subgroups according to this score are then returned to the user. As an example, consider a dataset of patients and their medical data. Let the target concept be *surgery successful*, which is true for 30% of the patients. Then a subgroup like *gender=male* $\wedge$ *smoker=false* with a higher rate of successful surgeries, e.g. 50%, receives a high score and is likely to be included in the result.

Practical applications have shown that results for traditional interestingness measures often contain variants of the same pattern multiple times. To avoid this problem, several authors postulated that a subgroup should not only be evaluated with respect to its own statistics, but also with respect to the statistics of its generalizations, see for instance [34, 27, 29], see also Chapter 7. Considering the example above, the subgroup *gender=male* $\wedge$ *smoker=false* would be rated as less interesting if it can be explained by one of its generalizations alone, e.g., if the subgroup *smoker=false* already describes a set of patients with a 50% surgery success rate. While the practical use of such generalization-aware interestingness measures has been widely acknowledged, the efficient mining in this setting has received little attention.

A key technique to improve runtime performance of subgroup discovery in general is the application of optimistic estimates, that is, upper bounds for the interestingness of any specialization of the currently evaluated subgroup, cf. Section 3.4.2. Although research has shown that improving the tightness of the utilized bounds improves the runtime performance substantially [108], there has been no extensive research so far

[1] This chapter is based on previously published work [174]: Florian Lemmerich, Martin Becker, and Frank Puppe: Difference-Based Estimates for Generalization-Aware Subgroup Discovery. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD), Best Paper Award*, 2013.

concerning upper bounds for generalization aware interestingness measures beyond the trivial transfer of bounds for traditional measures.

In this chapter, a novel method to exploit specific properties of generalization-aware measures to derive additional optimistic estimate bounds is proposed. Unlike previous approaches, the bounds are not exclusively based on the instances that are contained in the currently examined subgroup, but on the instances that were excluded in comparison to generalizations of the current pattern. In doing so, tighter optimistic estimate bounds can be computed and the runtime of algorithms is significantly reduced. It is shown how this general concept can be applied to exemplary interestingness measures in different setting, i.e., for subgroup discovery with binary target concepts and with numeric target concepts using a mean-based interestingness measure. The bounds are incorporated in a novel *apriori*-based algorithm that allows efficient propagation of the required statistics. The effectiveness of the presented approach is evaluated in experiments on publicly available data. It can be observed that the novel optimistic estimates are especially effective in tasks that incorporate selectors, which cover a majority of the dataset.

The rest of this chapter is structured as follows: Related work is discussed in Section 6.1. Next, the new scheme to derive optimistic estimate bounds and its application to different interestingness measures is presented in Section 6.2. Afterwards, it is explained how the new optimistic estimate bounds can efficiently be exploited in an algorithm in Section 6.3. Section 6.4 presents experimental results, before main results of the chapter are summarized in Section 6.5.

6.1 Related Work

Pruning based on optimistic estimates [248, 260] is an essential technique for efficient subgroup discovery. As Grosskreutz et al. showed, the efficiency of the pruning is strongly influenced by the *tightness* of the bounds [108]. A more general method to derive optimistic estimates for a whole class of interestingness measures, that is, *convex* measures, was introduced in [196] and later extended in [274]. This chapter presents a different technique to determine optimistic estimates to another family of interestingness measures, i.e., generalization-aware measures. In contrast to previous approaches, the optimistic estimate of a subgroup is not exclusively based on the instances covered by this subgroup, but takes also other, more general subgroups into account.

The necessity to consider also generalizations of patterns in selection criteria has been recognized for example in [34, 27], see also Chapter 7. These early approaches used a *minimum improvement constraint*, which is applied only as a post-processing operation after the mining algorithm. Webb and Zhang presented an efficiency improvement in mining with this constraint in the context of association rules [257] by introducing a pruning condition based on the difference in covering. While the method of Webb and Zhang requires *full* coverage on all instances, the method presented in this work can also be applied with only partial coverage. In addition, our method is used to derive upper bounds for interestingness measures instead of exploiting constraints and is also applied in settings with numeric target concepts.

Recent approaches incorporate differences with respect to generalizations directly in the interestingness measure. This showed positive results in descriptive [102] as well as predictive settings [30] for both binary and numeric target concepts. However, these papers focus more on which patterns are to be selected and not on efficient mining through pruning. As an exception, Batal and Hausknecht utilized a pruning scheme in an Apriori-based algorithm that is based exclusively on the positives covered by a subgroup [29]. This algorithm is used for comparison in the evaluation section. Utilizing pruning in settings with numeric concepts of interest is more challenging than in the binary case. Optimistic estimates for standard (non-generalization-aware) interestingness measures are discussed in Chapter 4. To the author's knowledge, no pruning bounds for numeric generalization-aware measures have been proposed until now.

6.2 Estimates for Generalization-Aware Subgroup Mining

This section introduces a novel scheme to derive optimistic estimates for generalization-aware interestingness measures. These optimistic estimates help to improve the runtime performance of algorithms by pruning the search space.

This chapter concentrates on two families of interestingness measures:

$$r^a_{bin}(P) = {i_P}^a \cdot (\tau_P - \max_{H \subset P} \tau_H), a \in [0,1]$$
$$r^a_{num}(P) = {i_P}^a \cdot (\mu_P - \max_{H \subset P} \mu_H), a \in [0,1]$$

These trade-off the number of instance covered by a subgroup i_P with the increase of the target share (mean value) in the subgroup τ_P (μ_P) in comparison to the maximum target share (mean value) in all generalizations of P. Although other interestingness measures could also be adapted to take generalizations into account, this chapter focuses on the measures $r^a_{bin}(P)$ and $r^a_{num}(P)$, since they are the only ones, which have been described in previous literature and applied in practice. It is also not argued about advantages of these functions in comparison to traditional measures or other methods that avoid redundant output, such as closed patterns [96], see also Section 2.5.5. Instead, the efficient mining for these generalization-aware measures by introducing novel, difference-based optimistic estimates is investigated.

Next, optimistic estimates that have been previously presented for this task are generalized. This outlines the conventional approach to derive upper bounds. Then, the core idea of our new scheme to derive upper bounds is presented: difference-based optimistic estimates. Afterwards, it is demonstrated how this concept can be exploited by deriving estimates for interestingness measure in the binary and the numeric case using measures r^a_{bin} and r^a_{num}.

6.2.1 Optimistic Estimates Based on Covered Positive Instances

Traditionally, optimistic estimates for subgroup discovery are based only on the anti-monotonicity of instance coverage. That is, when adding an additional selector to a

subgroup description P, the resulting subgroup only covers a subset of the instances covered by P. To give an example for this traditional approach, the following theorem generalizes the optimistic estimate bounds for r^a_{bin} used in [29], which covers only the special case using the parameter $a = 0.5$.

Theorem 2 *Let p_P be the number of all positive instances covered by the currently evaluated pattern P and $\max_{H \subset P}(\tau_H)$ the maximum of the target shares for P and any of its generalizations. Then, optimistic estimate bounds $oe_{r^a_{bin}}$ for the family of interestingness measures r^a_{bin} are given by: $oe_{r^a_{bin}} = (p_P)^a \cdot (1 - \max_{H \subseteq P}\ (\tau_H))$.* □

Proof It is shown first that the interestingness score of any specialization S does not decrease if all negative instances are removed. Let n_s be the number of negatives in S. Then, it holds that

$$r^a_{bin}(S) = (p_S + n_S)^a \cdot (\frac{p_S}{p_S + n_S} - \max_{H \subset S} \tau_H) = \frac{p_S}{(p_S + n_S)^{1-a}} - (p_S + n_S)^a \cdot \max_{H \subset S} \tau_H.$$

This term is examined as a function of n_S, $n_S \geq 0$: The first summand decreases with increasing n_S since $1 - a \geq 0$. The second, negative summand increases with increasing n_S, as $\tau_H \geq 0$. Thus, the maximum is reached for $n_S = 0$. One can conclude that:

$$\begin{aligned} r^a(S) &= (p_S + n_S)^a \cdot (\frac{p_S}{p_S + n_S} - \max_{H \subset S} \tau_H) \\ &\leq (p_S)^a \cdot (\frac{p_S}{p_S} - \max_{H \subset S} \tau_H) \\ &\leq (p_P)^a \cdot (1 - \max_{H' \subseteq P} \tau_{H'}), \end{aligned}$$

as the number of positives in the specialization S is smaller than the number of positives in the more general pattern P, and the generalizations of S include all generalizations of P. ■

As it has been exemplified in [29], this bound can already achieve significant runtime improvements. Note that these bounds exploit only the anti-monotonicity of the covered positive instances. The next sections will demonstrate how additional information on the difference of negative instances between subgroups and their generalizations can be used to derive additional bounds.

Example 15 Assume that the subgroup $A = A_1 \wedge \ldots \wedge A_n$ covers 16 positive and 8 negative instances and the maximum target share in any generalization of A is 20%. When the subgroup discovery task uses the interestingness measure $r^{0.5}_{bin}$, then an optimistic estimate bound for A is given by $16^{0.5} \cdot (1 - 20\%) = 3.2$. That means that if the result set requires an interestingness score of more than 3.2, then no specializations of the subgroup A have to be considered, since these specialization have a maximum interestingness value of 3.2. Skipping such specializations can substantially improve the runtime performance of search algorithms. □

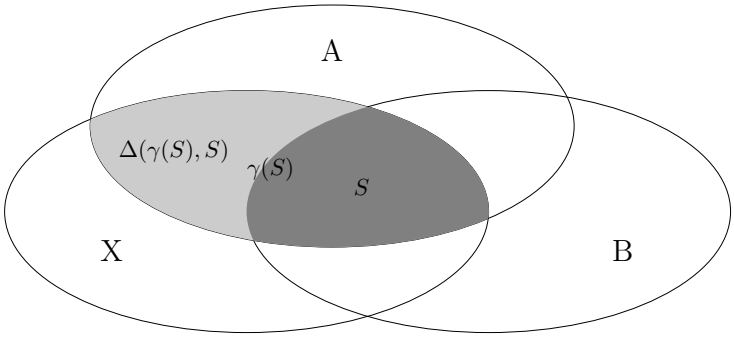


Figure 6.1: Illustration of Lemma 2: For each specialization $S = A \wedge B \wedge X$ (visualized in dark gray) of $A \wedge B$, there is a generalization $\gamma(S) = A \wedge X$ (visualized by the whole gray area), such that the difference $\Delta(\gamma(S), S)$ (visualized in light gray) is a subset of the difference $\Delta(A, B) = A \backslash B$.

6.2.2 Difference-based Pruning

Next, the core idea of the novel pruning scheme to derive optimistic estimates is provided. It incorporates the set of instances, by which two subgroups A and B differ. Formally, $\Delta(A, B) = sg(A) \setminus sg(B)$ will describe the instances, which are covered by A, but not by B. The novel approach utilizes that these difference sets are – in a certain way – anti-monotonic. More specifically, the following lemma will be exploited to derive optimistic estimates:

Lemma 2 *Let $P = A \wedge B$ be any subgroup description with $A = A_1 \wedge \ldots \wedge A_{|A|}$ and $B = B_1 \wedge \ldots \wedge B_{|B|}$ potentially being conjunctions themselves, and $|A| \geq 0, |B| \geq 1$. Then for any specialization $S \supset P$ there exists a generalization $\gamma(S) \subset S$, such that $\Delta(\gamma(S), S) \subseteq \Delta(A, B)$.* □

Proof Consider for any specialization $S = A \wedge B \wedge X$ (X being potentially a conjunction itself) the pattern $\gamma(S) = A \wedge X$. This is a real generalization of S, since $B \neq \emptyset$. Then it holds that:

$$\begin{aligned}
\Delta(\gamma(S), S) &= sg(A \wedge X) \setminus sg(A \wedge B \wedge X) \\
&= (sg(A) \cap sg(X)) \setminus (sg(A) \cap sg(B) \cap sg(X)) \\
&= sg(X) \cap (sg(A) \setminus (sg(A) \cap sg(B)) \\
&= sg(X) \cap (sg(A) \backslash sg(B)) \\
&= sg(X) \cap \Delta(A, B).
\end{aligned}$$

This is a subset of $\Delta(A, B)$. ■

The different sets used in the lemma are visualized in Figure 6.1. The subset property of this lemma implies directly that for each specialization S the generalization $\gamma(S)$

contains at most $i_S + i_{\Delta(A,B)}$ instances. Additionally, in the case of a binary target, one can estimate the number of negative instances in this generalization: $n_{\gamma(S)} \leq n_S + n_{\Delta(A,B)}$. Furthermore, in the case of a numeric target, the minimum target value of $\Delta(\gamma(S), S)$ is higher than the minimum target value in $\Delta(A, B)$. In mining algorithms, statistics for $\Delta(A, B)$ can be computed with almost no additional effort. For instance, n_A and $n_{A \wedge B}$ are both required anyway in order to evaluate the pattern $A \wedge B$ with r^a_{bin}. Then, $n_{\Delta(A,B)}$ is given by $n_{\Delta(A,B)} = n_A - n_{A \wedge B}$.

The property can be used to prune the search space, as demonstrated in the following example:

Example 16 Assume that the pattern A covers 20 positive and 10 negative instances and the evaluation of the pattern $A \wedge B$ shows that this pattern also covers 10 negative instances. That is, B covers all negative instances, which are covered by A, and therefore $n_{\Delta(A,B)} = 0$. Now consider any specialization S of this subgroup. According to the lemma, S has another generalization $\gamma(S)$ that contains the same number of negative instances as S since $n_{\gamma(S)} \leq n_S + n_{\Delta(A,B)} = n_S + 0$. As S (as a specialization of $\gamma(S)$) additionally has no more positive instances than S, the target share in S is equal or smaller than for its generalization $\gamma(S)$. Thus, the interestingness values of S according to any generalization-aware measure r^a_{bin} is ≤ 0. Since this is the case for any specialization of $A \wedge B$, specializations of $A \wedge B$ can be pruned from the search space without influencing the results. □

This is an extreme example: *all* negative instances of A are also covered by $A \wedge B$. Now assume that $A \wedge B$ had covered only 8 negative instance, thus $n_{\Delta(A,B)} = 10 - 8 = 2$. In this case, the lemma guarantees that S has a generalization $\gamma(S)$ with *at most 2* negative instances more than S. If S itself covers a decent amount of instances, then the target share in S cannot be much higher than in $\gamma(S)$. As a consequence, either S is small or there is only a small increase (or a decrease) in the target share comparing S and its generalization $\gamma(S)$. In both cases, the interestingness score of S according to r^a_{bin} is low.

Overall it can be concluded that if the difference of covered instances between A and $A \wedge B$ is small, then the interestingness score for all specializations is limited. The next sections formalize these considerations by deriving formal optimistic estimate bounds that can be used to prune the search space.

6.2.3 Difference-based Optimistic Estimates for Binary Targets

In the following, novel optimistic estimate bounds for generalization-aware measures $r^a_{bin} = i_P{}^a \cdot (\tau_P - \max_{H \subset P} \tau_H)$ with binary targets are inferred. These are based on the difference of the coverage of a subgroup in comparison to the coverage of its generalizations.

Theorem 3 *Consider the pattern P with p_P positive instances. $P' \subseteq P$ is either P itself or one of its generalizations and $P'' \subset P'$ a generalization of P'. Let $n_\Delta = n_{P''} - n_{P'}$*

be the difference in coverage of negative instances between these patterns. Then, an optimistic estimate of P for r^a_{bin} is given by:

$$oe_{r^a_{bin}}(P) = \begin{cases} \frac{p_P \cdot n_\Delta}{p_P + n_\Delta}, & \text{if } a = 1 \\ \frac{n_\Delta}{1 + n_\Delta}, & \text{if } a = 0 \\ \frac{\hat{p}^a \cdot n_\Delta}{\hat{p} + n_\Delta}, \text{ with } \hat{p} = \min(\frac{a \cdot n_\Delta}{1-a}, p_P), & \text{else} \end{cases}$$ □

Proof Let S be any specialization of P and $G = \gamma(S)$ the generalization with $\Delta(G, S) \subseteq \Delta(P', P'')$. Such a generalization exists according to the previous lemma, since S is also a specialization of P'. The number of negatives in G is equal to the number of negatives covered by S plus the number of negatives, which are covered by G, but not by S: $n_G = n_S + n_{\Delta(G,S)}$. By construction it holds that $n_{\Delta(G,S)} \leq n_\Delta$. Additionally, it can be assumed that $p_S > 0$, that is, S contains at least one positive instance since $r^a_{bin}(S) \leq 0$ otherwise.

The proof first derives an upper bound that depends on the number of positives in the specialization S, which is unknown at the time P is evaluated. Therefore, the maximum value of this function is determined in a second step. The interestingness score of S is given by:

$$\begin{aligned}
r^a_{bin}(S) &= (p_S + n_S)^a \cdot (\tau_S - \max_{H \subset S} \tau_H) && (6.1)\\
&\leq (p_S + n_S)^a \cdot (\tau_S - \tau_G) && (6.2)\\
&= (p_S + n_S)^a \cdot (\frac{p_S}{p_S + n_S} - \frac{p_G}{p_G + n_S + n_{\Delta(G,S)}}) && (6.3)\\
&\leq (p_S + n_S)^a \cdot (\frac{p_S}{p_S + n_S} - \frac{p_S}{p_S + n_S + n_{\Delta(G,S)}}) && (6.4)\\
&= (p_S + n_S)^a \cdot (\frac{p_S \cdot (p_S + n_S + n_{\Delta(G,S)}) - (p_S \cdot (p_S + n_S))}{(p_S + n_S)(p_S + n_S + n_{\Delta(G,S)})}) && (6.5)\\
&= \frac{p_S \cdot n_{\Delta(G,S)}}{(p_S + n_S)^{1-a}(p_S + n_S + n_{\Delta(G,S)})} && (6.6)\\
&\leq \frac{p_S \cdot n_{\Delta(G,S)}}{(p_S)^{1-a}(p_S + n_{\Delta(G,S)})} && (6.7)\\
&= \frac{p^a_S \cdot n_{\Delta(G,S)}}{(p_S + n_{\Delta(G,S)})} && (6.8)\\
&\leq \frac{p^a_S \cdot n_\Delta}{(p_S + n_\Delta)} := f^a(p_S) && (6.9)
\end{aligned}$$

The transformation to line 2 is possible, since $G \subset S$. In line 4, it is used that $p_S \leq p_G$, as the positives of S are a subset of the positive of its generalization G. In line 7, it is exploited that the denominator is strictly increasing with increasing n_S because $1 - a \in [0, 1]$. Therefore, the smallest denominator and thus the largest value for the overall term is achieved by setting $n_S = 0$. The term in line 8 is strictly increasing as a

function of $n_{\Delta(G,S)}$. Since $n_{\Delta(G,S)} \leq n_\Delta$, line 9 follows.

In the final line 9, the function $f^a(p_S)$ is defined, which provides an upper bound on the interestingness of P that depends on the number of positives within the specialization. Some examples of this function for different values of a are illustrated in Figure 6.2.

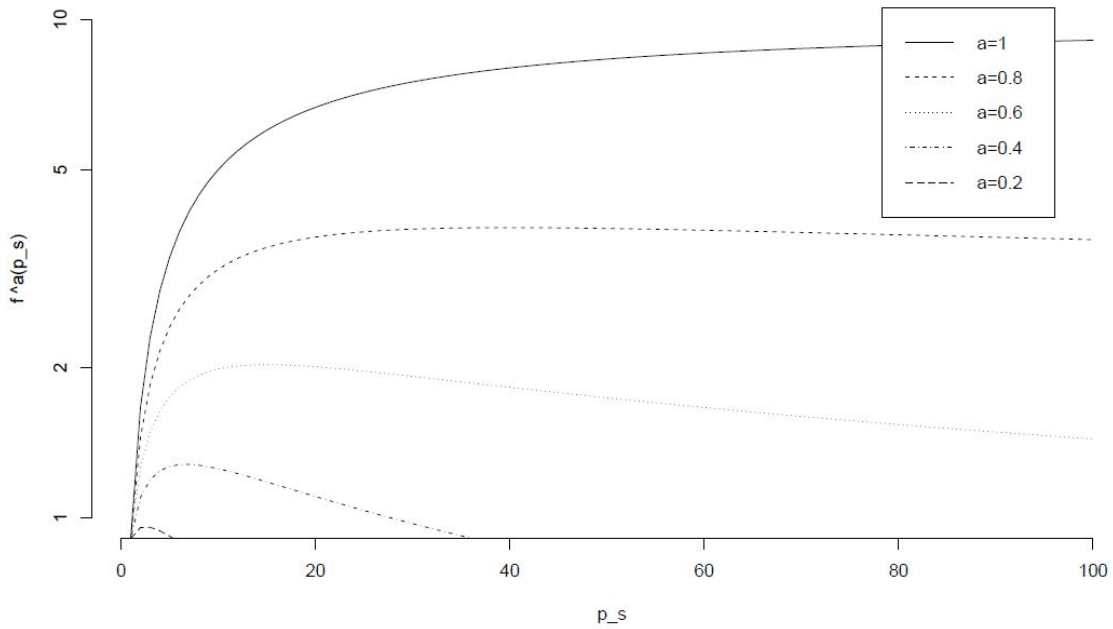


Figure 6.2: Some examples for the functions $f^a(p_S) = \frac{p_S^a \cdot n_\Delta}{(p_S + n_\Delta)}$ for $n_\Delta = 10$ and different values of a. For these functions the maximum values are determined formally in the proof in order to deduce upper bounds. As it is exemplified in the plots, the function is strictly increasing for $a = 1$ and has exactly one maximum value for $0 < a < 1$. For $a = 0$ (not shown in the plot) the function is strictly decreasing.

The number of positives in the specialization is not known, when the subgroup P is evaluated. Intuitively, for large number of positives in the specialization removing n_Δ negative instances will not change the target share in the subgroup much. Therefore, the interestingness of the generalization is limited. On the other hand, for small numbers of positive instances S is overall small and possibly not interesting for that reason. p_S is assumed to be at least 1, since S otherwise is not interesting anyway and at most p_P, as the number of positives for S is smaller than for its generalization P. Next, it is analyzed for which value of p_S the function $f^a(p_S)$ of line 9 reaches its maximum in the interval $[1, p_P]$. This depends on the parameter a of the interestingness measure:

1. For $a = 1$, it holds that $f^1(p_S) = \frac{p_S \cdot n_\Delta}{p_S + n_\Delta}$. This function is strictly increasing in p_S. That is, the more positive instances are contained in S, the higher is the derived upper bound. The maximum is reached at the highest value in the domain of definition: $\max(f^1(p_S)) = f^1(p_P) = \frac{p_P \cdot n_\Delta}{p_P + n_\Delta}$.

2. In contrast for $a = 0$, $f^0(p_S) = \frac{n_\Delta}{p_S + n_\Delta}$ is strictly decreasing. Thus, the maximum value of f^0 is reached for $p_S = 1$, the minimum possible value of p_S: $\max(f^0(p_S)) = f^0(1) = \frac{n_\Delta}{1+n_\Delta}$.

3. For $0 < a < 1$, f^a reaches a maximum for a certain value p^* within the domain of definition. To determine that, the first derivative of f^a is calculated using the quotient rule.

$$\begin{aligned}\frac{d}{d\,p_S} f^a(p_S) &= n_\Delta \cdot \frac{d}{dp_S} \frac{p_S^a}{p_S + n_\Delta} \\ &= n_\Delta \cdot \frac{(n_\Delta + p_S) \cdot a \cdot p_S^{a-1} - p_S^a}{(n_\Delta + p_S)^2} \\ &= n_\Delta \cdot p_S^{a-1} \frac{a n_\Delta + a \cdot p_S - p_S}{(n_\Delta + p_S)^2} := (f^a)'\end{aligned}$$

   The only root of this derivative is at $p^* := \frac{a \cdot n_\Delta}{1-a}$. As can be easily shown, $(f^a)'(p_S)$ is greater than zero for p_S smaller than p^* and lower than zero for p_S greater than p^*. Therefore, p^* is the only maximum of $f^a(p_S)$. Thus, if $p_P > p^*$, then p^* is the maximum value of f^a, otherwise the maximum is reached at the highest value of the domain of definition: $\max(f^a(p_S)) = f^a(\hat{p}) = \frac{\hat{p}^a \cdot n_\Delta}{\hat{p} + n_\Delta}$, with $\hat{p} = \min(\frac{a \cdot n_\Delta}{1-a}, p_P)$.

   Overall, for any specialization S it holds that $r_{bin}^a(S) \leq f^a(p_S) \leq \max f^a(p_S) = oe_{r_a^{bin}}(P)$, with the function maxima as described above, therefore $oe_{r_a^{bin}}(P)$ as defined in the theorem is a correct optimistic estimate. ■

For any pair of two generalizations of P as well as for any pair of P and one of its generalization (if $P = P'$), this theorem provides an optimistic estimate of P. The optimistic estimate bound is dependent on the number of positives in the subgroup and the difference of negative instances between P' and P''. It is low if either there are only few positives in P or the difference of negative instances between the pair of generalizations is small (or a combination of both). Since the number of positives in P is independent of the chosen pair P', P'', the pair with the minimum difference of negative instances implies the tightest upper bound, which should be used to maximize the effects of pruning.

As a special case, the theorem includes that the interestingness score of any subgroup is ≤ 0 if n_Δ is 0. To the author's knowledge, it is the first measure that includes these differences in optimistic estimate bounds for subgroup discovery.

In some situations, this theorem provides significantly tighter bounds than the traditional approach of Theorem 2, as can be observed in the following example:

Example 17 As in the previous example, assume that the subgroup A covers 20 positive and 10 negative instances and the evaluation of the pattern $A \wedge B$ shows that this subgroup covers 16 positive and 8 negative instances. As interestingness measure the subgroup discovery task uses $r_{bin}^{0.5}(P)$. Then, Theorem 3 can be used to derive an optimistic estimate bound for the subgroup $A \wedge B$: The required statistic $\hat{p}$ is determined as

$\hat{p} = \min(\frac{a \cdot n_\Delta}{1-a}, p_{A \wedge B}) = \min(\frac{0.5 \cdot (10-8)}{1-0.5}, 16) = 2$. The optimistic estimate bound is then: $oe_{r^{0.5}_{bin}}(A \wedge B) = \frac{\hat{p}^a \cdot n_\Delta}{\hat{p}+n_\Delta} = \frac{2^{0.5} \cdot 2}{2+2} = \frac{\sqrt{2}}{2}$.

The traditional bound from Theorem 2 depends on the maximum target share in the generalizations of $A \wedge B$. If this value is for example at 20%, then the derived upper bound for the subgroup $A \wedge B$ takes the value 3.2 as computed in Example 15. This is significantly less tight than the novel bound from Theorem 3. This can be explained by the fact that only the novel bound exploits that the subgroup $A \wedge B$ and its generalization A differ only by a few negative instances. □

6.2.4 Difference-based Optimistic Estimates for Numeric Targets

A related approach can be used to obtain optimistic estimates for generalization-aware interestingness $r^a_{num} = {i_P}^a \cdot (\mu_P - \max_{H \subset P} \mu_H)$ in settings with numeric target concepts:

Theorem 4 *In a task with a numeric target concept, consider the pattern P with i_P instances and a maximum target value of $\max_P$. $P' \subseteq P$ is either P itself or one of its generalizations and $P'' \subset P'$ is a generalization of P'. Let $i_\Delta = |\Delta(P'', P')|$ be the number of instances contained in P'', but not in P' and $\min_\Delta$ the minimum target value contained in $\Delta(P'', P')$. Then, an optimistic estimate of P for the generalization aware interestingness measure r^a_{num} is given by:*

$$oe_{r^a_{num}}(P) = \max(0, oe'_{r^a_{num}}(P)),$$

$$oe_{r^a_{num}}(P)' = \begin{cases} \frac{i_\Delta \cdot i_P}{i_P + i_\Delta} \cdot (\max_P - \min_\Delta), & \text{if } a = 1 \\ \frac{i_\Delta}{1+i_\Delta} \cdot (\max_P - \min_\Delta), & \text{if } a = 0 \\ \frac{\hat{i}^a \cdot i_\Delta}{\hat{i}+i_\Delta} \cdot (\max_P - \min_\Delta), \text{ with } \hat{i} = \min(\frac{a \cdot i_\Delta}{1-a}, i_P), & \text{else} \end{cases}$$ □

Proof Consider any specialization $S \supset P$ and its generalization $G = \gamma(S)$ according to Lemma 2. Then one can estimate the interestingness of S:

$$r^a_{num}(S) = {i_S}^a \cdot (\mu_S - \max_{H \subset S} \mu_H) \tag{6.1}$$

$$\leq {i_S}^a \cdot (\mu_S - \mu_G) \tag{6.2}$$

$$= {i_S}^a \cdot \left(\frac{\sum_{c \in sg(S)} T(c)}{i_S} - \frac{\sum_{c \in sg(S)} T(c) + \sum_{c \in \Delta(G,S)} T(c)}{i_S + i_{\Delta(G,S)}} \right) \tag{6.3}$$

$$= {i_S}^{a-1} \cdot \left(\sum_{c \in sg(S)} T(c) - \frac{i_S \cdot \left(\sum_{c \in sg(S)} T(c) + \sum_{c \in \Delta(G,S)} T(c) \right)}{i_S + i_{\Delta(G,S)}} \right) \tag{6.4}$$

$$= {i_S}^{a-1} \cdot \left(\frac{i_{\Delta(G,S)} \sum_{c \in sg(S)} T(c) - i_S \sum_{c \in \Delta(G,S)} T(c)}{i_S + i_{\Delta(G,S)}} \right) \tag{6.5}$$

$$\leq {i_S}^{a-1} \cdot \left(\frac{i_{\Delta(G,S)} \cdot i_S \cdot \max_{c \in S} T(c) - i_S \cdot i_{\Delta(G,S)} \cdot \min_{c \in \Delta(G,S)} T(c)}{i_S + i_{\Delta(G,S)}} \right) \tag{6.6}$$

$$= \frac{i_{\Delta(G,S)} \cdot {i_S}^a}{i_S + i_{\Delta(G,S)}} \cdot \left(\max_{c \in S} T(c) - \min_{j \in \Delta(G,S)} T(c) \right) \tag{6.7}$$

$$\leq \frac{i_\Delta \cdot {i_S}^a}{i_S + i_\Delta} \cdot (max_P - min_\Delta) = f^a(i_S) \cdot (max_P - min_\Delta) \tag{6.8}$$

In line 2, it is used that G is a generalization of S. Then it is exploited that $sg(G) = sg(S) \cup \Delta(G,S), S \cap \Delta(G,S) = \emptyset$. Line 6 exploits that the sum of any set of values is bigger than the minimum appearing value times the size of the set, but smaller than the maximum appearing value times the size of the set. Line 8 utilizes that $i_{\Delta(G,S)} \leq i_\Delta$.

f^a is a function over the unknown number of all instances in the specialization, which can be any number in $[1, i_P]$. f^a is always positive. Therefore, if $(\max_P - \min_\Delta) \leq 0$, the optimistic estimate is given by 0. Else, the maxima of f^a, which have already been derived in the proof of Theorem 2, determine the bound: $f^a(i_S)$ is strictly increasing for $a = 1$, strictly decreasing for $a = 0$ and reaches a maximum at $\frac{a \cdot i_\Delta}{1-a}$ or at i_P otherwise. Thus: $r^a_{num}(S) \leq (f^a(i_S)) \cdot (\max_P - \min_\Delta) \leq \max(f^a(i_S) \cdot (\max_P - \min_\Delta)$. The bounds follow directly from inserting the respective maximum values. Since this holds for any specialization S of P, $oe_{r^a_{num}}(P)$ is a correct optimistic estimate for P. ■

Similar to the optimistic estimate in the binary case, the derived optimistic estimate is low, if either the number of instances covered by P is low, or if the difference between the numbers of instances covered by the generalizations P'' and P' is low (or a combination of both). However additionally, the bound also considers the range of the target variable in these patterns, that is, the maximum occurring target value in P and the minimum target value in the difference set of instances. As a result, the bound becomes zero if the minimum target value removed by adding a selector to a generalization of P was higher than the maximum remaining target value in P.

6.3 Algorithm

The presented optimistic estimates can in general be applied in combination with any search strategy. This chapter focuses on adapting an exhaustive algorithm, i.e., apriori [4, 135], see also Chapter 3 for a detailed description of the base algorithm. This approach is especially suited for the task of generalization-aware subgroup discovery, since its levelwise search strategy guarantees that specializations are always evaluated after their generalizations and the highest target share found in generalizations can efficiently be propagated from generalizations to specializations, see [29]. Therefore, and for better comparability with previous approaches, apriori was chosen as a basis for the novel algorithm presented here. Using the following adaptations the algorithm is not only capable of determining the proposed optimistic estimates. The algorithm also propagates the required information very efficiently. The algorithmic adapations are described for the binary case first.

Apriori performs a levelwise search, where new candidate patterns are generated from the last level of more general subgroups. In the adaptation of the algorithm, additional information is stored for each candidate. This includes the maximum target share in

generalizations of this subgroup, the minimum number of negatives covered by any generalization and the minimum number of negatives that were removed in generalizations of this subgroup. After the evaluation of a subgroup the number of positives, the number of negatives and the resulting target share are additionally saved in each candidate. The minimum number of negative instances in a generalization is required to compute the minimum number of instances, which are contained in the subgroup, but not in a generalization. The other statistics are directly required to compute either the interestingness score or the optimistic estimates of the subgroup.

Whenever a new candidate subgroup P is generated in apriori, it is checked if all its direct generalizations G are contained in the candidate set of the previous level. During this check, the statistics for the maximum target share in generalizations, the minimum number of negatives in a generalization and the minimum number of negatives that were removed in any generalization of this subgroup are computed by using the information stored in the generalizations and applying simple minimum/maximum functions. In doing so, the statistics required to compute the interestingness score of the subgroup and the optimistic estimates are propagated very efficiently from one level of subgroups to the next level of more specific subgroups.

In the evaluation phase (the counting phase in classical apriori), each candidate is scored with respect to the applied interestingness measure. This requires to determine the coverage of the subgroup. Combined with previously computed statistics about generalizations this is used to compute the interestingness according to the chosen generalization-aware measure. Subgroups with sufficient high scores are placed in the result set, potentially replacing others in a top-k approach. Afterwards the target share in generalizations and the minimum number of removed negative instances are updated by using the statistics of the current subgroup's coverage. After the evaluation of a subgroup all optimistic estimates, that is, traditional estimates (see Theorem 2) and difference-based estimates are computed from the information stored for a candidate. If any optimistic estimate is lower than the threshold given by the result set for a top-k subgroup, then the subgroup is removed from the list of current candidates. Thus, no specializations of this subgroup are explored in the next level of search.

The approach for numeric target concepts is very similar, except that minimum/maximum and mean target values as well as overall instance counts of the candidate subgroup are stored instead of counts of positives and negatives. When determining the pruning bounds, a subgroup is compared with all its direct generalizations. For each generalization, an optimistic estimate bound is computed based on the difference of instances between the generalization and the specialization and the stored minimum/maximum target values. The tightest bound can be applied for pruning.

6.4 Evaluation

This section demonstrates the effectiveness of the presented approach in an evaluation using well-known datasets from the UCI [13] repository[2]. As a baseline algorithm a variant of the MPR-algorithm presented in [29] is used as the current state-of-the-art algorithm for this setting. The algorithm was slightly modified in order to support top-k mining and to incorporate the bounds of Theorem 2 for any a. Since this algorithm follows the same search strategy as our novel algorithm, that is, apriori, this allows to determine the improvements that originate directly from the advanced pruning bounds presented in this work. For the experiments, the algorithms were implemented as a plugin in the open-source environment *VIKAMINE 2*, see Chapter 8. The implementation utilizes an efficient bitset-based data structure to determine the coverage of subgroups. Results below are shown for $k = 20$, a realistic number for practical applications, which was also used for example as beam size in [274]. Different choices of k lead to similar results. For the numeric attributes an equal-frequency discretization was used, using all half-open intervals from the cutpoints and the intervals between two adjacent cutpoints as selectors. The experiments were performed on an office PC with 2.8 Ghz and 6 GB RAM.

The first part of the evaluation compared the runtimes of the presented algorithm for binary target concepts using different generalization-aware interestingness measures r^a_{bin}, see Table 6.1 and Table 6.2.

The results in Table 6.1 show that utilizing difference-based pruning leads to significant runtime improvements in almost all tasks. The runtime improvements tend to be larger for higher parameters of a, since pruning can be applied more effectively if the interestingness measures favors larger subgroups. The improvements are explained by the fact that by exploiting the additional pruning options substantially less candidates have to be evaluated, see Table 6.2. This direct dependency between the number of evaluated candidates and runtime can also be observed for the subsequent experiments, for which the candidate numbers will not be listed separately.

Some datasets in particular showed massive runtime improvements, e.g., the datasets *ionosphere*, *spammer* or *segment*. For a more detailed analysis, these tasks were investigated more closely. It turned out that the search space for these datasets contains multiple selectors that cover a vast majority of the instances. Conjunctive combinations of subsets of these selectors still cover a large part of the dataset and especially of the positive instances. As traditional optimistic estimates are based on this number of covered positive instances, pruning cannot be applied on these combinations efficiently. In contrast, since the number of negative instances, by which those patterns differ from generalizations, is often very low in these cases, such combinations can be pruned often using the difference-based optimistic estimates presented in this work. This leads to the massive improvements. It can be concluded that our new pruning scheme is especially efficient if many selectors cover a majority of the dataset. In some cases, the algorithms

[2] Additionally, the dataset *spammer* is derived from the publicly available data from the ECML/PKDD discovery challenge 2008, see also Section 7.7.4.

Table 6.1: Runtime comparison (in s) of the base algorithm with traditional pruning based on the positives (std) and the novel algorithm with additional difference-based pruning (dbp) using different size parameters a for interestingness measures r^a_{bin}. The maximum number of describing selectors was limited to $d = 5$. "-" indicates that the algorithm did not finish due to lack of memory.

a	0.0		0.1		0.5		1.0	
Pruning	dpb	std	dpb	std	dpb	std	dpb	std
adults	0.9	0.9	16.5	40.8	1.0	6.8	0.6	1.3
audiology	< 0.1	0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
autos	0.4	-	0.3	30.6	0.1	16.0	< 0.1	1.7
census-kdd	15.4	15.2	1103.5	-	46.0	245.9	16.1	28.2
colic	0.1	< 0.1	0.6	1.1	0.2	1.6	0.1	0.5
credit-a	< 0.1	< 0.1	0.7	1.7	0.2	1.8	< 0.1	0.3
credit-g	0.2	0.1	14.3	26.4	3.8	25.7	0.4	4.2
diabetes	1.3	5.3	8.0	19.6	1.8	16.2	0.1	1.2
hepatitis	0.3	6.4	0.7	1.6	0.1	1.3	< 0.1	0.2
hypothyroid	< 0.1	0.6	0.2	2.2	0.2	1.7	< 0.1	0.6
ionosphere	0.1	0.1	148.4	-	0.3	-	0.1	102.2
spammer	1.5	1.9	277.8	-	12.1	412.8	0.5	41.4
segment	31.5	-	5.5	317.4	0.1	69.3	< 0.1	12.1

Table 6.2: Comparison of the number of evaluated candidates in the base algorithm with traditional pruning based on the positives (std) and in the novel algorithm with additional difference-based pruning (dbp). Each double column uses a different size parameter a for the interestingness measures r^a_{bin}. The maximum number of describing selectors was limited to $d = 5$. "-" indicates that the algorithm did not finish due to lack of memory.

a	0.0		0.1		0.5		1.0	
Pruning	dpb	std	dpb	std	dpb	std	dpb	std
adults	66,909	148,408	1,272,897	4,559,066	76,280	773,210	14,456	123,321
audiology	10,950	46,390	6,349	13,791	6,349	13,791	6,387	13,829
autos	209,553	-	47,422	6,387,287	31,320	4,410,259	16,783	626,939
census-kdd	82,215	164,430	22,400,736	-	815,878	6,140,536	88,276	461,130
colic	6,903	13,806	109,323	404,352	80,074	563,018	29,377	192,550
credit-a	4,186	8,372	146,824	595,565	95,036	683,878	10,986	118,741
credit-g	68,806	149,175	2,496,144	7,834,159	1,104,248	6,837,491	176,187	1,444,033
diabetes	449,096	2,183,798	1,527,242	5,755,861	560,236	4,643,343	26,482	423,433
hepatitis	115,483	2,105,085	168,461	642,124	56,970	533,107	8,573	74,797
hypothyroid	14,994	178,394	42,049	555,357	62,195	564,959	8,945	186,236
ionosphere	56,956	114,587	22,724,266	-	80,483	-	62,881	20,680,731
spammer	171,565	390,367	13,127,644	-	1,237,991	14,567,622	14,669	3,922,800
segment	6,591,619	-	924,140	26,802,236	34,946	11,920,849	19,514	3,045,612

did not finish due to out-of-memory errors despite the large amount of available memory. This occurs less often using the novel bounds, see for example the results for the vehicle dataset, since less candidates are generated and have to be stored in apriori.

In the second part of the evaluation, the interestingness measure $r_{bin}^{0.5}$, a generalization-aware variant of the binomial-test, was further analyzed by comparing the runtimes for different search depth (maximum number of selectors in a subgroup description), see Table 6.3. As before, almost all tasks finished earlier using the novel difference-based pruning. While the improvement is only moderate for low search depth, massive speedups can be observed for $d = 5$ and $d = 6$. For $d = 6$, several runs with only traditional pruning did not finish because of limited memory. This did not happen for the datasets in the experiments if the additional pruning was applied.

Table 6.3: Runtime comparison (in s) of the base algorithm with traditional pruning based on the positives (std) and the novel algorithm with additional difference-based pruning (dbp) using different maximum numbers d of describing selectors in a pattern. As interestingness measure the generalization-aware mean test $r_{bin}^{0.5}$ was used. "-" indicates that the algorithm did not finish due to lack of memory.

d	3		4		5		6	
Pruning	dpb	std	dpb	std	dpb	std	dpb	std
adults	0.7	0.8	0.9	1.6	1.0	6.8	1.3	26.7
autos	< 0.1	0.1	0.1	1.5	0.1	16.0	0.2	182.4
census-kdd	16.6	17.5	26.0	48.1	46.0	245.9	79.8	6392.5
colic	0.1	0.1	0.2	0.5	0.2	1.6	0.3	3.6
credit-a	< 0.1	0.1	0.1	0.4	0.2	1.8	0.3	5.5
credit-g	0.2	0.2	1.2	2.6	3.8	25.7	6.3	235.1
diabetes	0.1	0.1	0.6	1.6	1.8	16.2	3.6	91.9
hepatitis	< 0.1	< 0.1	0.1	0.3	0.1	1.3	0.2	4.4
hypothyroid	< 0.1	0.1	0.1	0.4	0.2	1.7	0.2	6.3
ionosphere	0.1	1.2	0.1	31.7	0.3	-	0.1	-
spammer	1.0	1.4	3.5	13.4	12.1	412.8	30.2	-
segment	0.1	0.2	0.1	3.9	0.1	69.3	0.1	-

In the last part of the evaluation, the improvements in a setting with numeric target concepts and interestingness measures $r_{num}^{a}(P)$ were examined. For subgroup discovery with numeric targets and generalization-aware interestingness measures, no specialized optimistic estimates have been proposed so far. To allow for a comparison nonetheless, the optimistic estimate bound $\bar{o}e_{num}^{1} = \sum_{c:T(c)>\mu_\emptyset} (T(c)-\mu_\emptyset)$ is used, which has been shown to be a correct optimistic estimate for the non-generalization-aware measure $q_{num}^{1}(P) = {i_P}^{a} \cdot (\mu_P - \mu_\emptyset)$ [249]. Since $r_{num}^{a}(P) \leq r_{num}^{1}(P) \leq q_{num}^{1}(P)$, this can also be used as a (non-tight) optimistic estimate for any generalization-aware interestingness measure r_{num}^{a}. Using the considerations of Chapter 4 more sophisticated bounds could be derived

to further decrease runtimes, but for better comparability the experiments focus on this established bound.

Results are shown in Table 6.4 and Table 6.5. Since the applied traditional bound is tight for $a = 1$, the runtimes in this case are already relatively low for the studied datasets with traditional pruning, leaving only little room for improvement. For lower values of a, significant runtime improvements can be observed, which reach a full order of magnitude (e.g., for the datasets concrete_data and housing). The average relative runtime improvement is on average highest for $a = 0.5$. This can be explained by the fact that for lower values of a even small subgroups can be considered as interesting. This makes it more difficult to exclude subgroups from search by pruning also when using the difference based bounds. As in the binary case, the advantages of improved pruning increase for higher search depths, cf. Table 6.5.

Table 6.4: Runtime comparison (in s) of the base algorithm with traditional pruning based on the instances covered by a subgroup only (std) and the novel algorithm with additional difference-based pruning (dbp) for *numeric target concepts*. Each double column corresponds to a different size parameter a for the interestingness measure r^a_{num}. The maximum number of describing selectors was limited to $d = 5$. "-" indicates that the algorithm did not finish due to lack of memory.

a	0.0		0.1		0.5		1.0	
Pruning	dpb	std	dpb	std	dpb	std	dpb	std
adults	17.0	71.0	19.9	73.7	11.4	49.9	3.9	11.6
autos	2.8	-	5.5	-	0.7	-	< 0.1	0.9
breast-w	0.5	2.1	0.9	2.2	0.1	1.3	< 0.1	0.1
concrete_data	8.5	42.1	11.7	47.7	2.1	29.6	0.1	0.6
credit-a	1.3	5.6	1.9	5.9	1.1	4.7	0.1	0.4
credit-g	3.4	21.8	5.7	23.1	4.1	16.2	0.2	0.5
diabetes	4.3	22.8	6.0	25.1	5.0	20.0	0.4	0.9
forestfires	1.9	7.5	2.8	7.9	2.2	6.8	2.4	4.9
glass	1.7	19.4	2.4	19.3	0.3	7.3	< 0.1	0.2
heart-c	2.7	7.1	3.6	7.7	1.8	6.1	0.1	0.3
housing	3.2	43.6	5.5	46.0	2.8	38.4	0.2	5.7
yeast	3.3	13.7	4.3	14.5	1.7	10.4	0.1	1.3

Overall, the evaluation results clearly indicate that the difference-based pruning that is introduced in this work substantially improves the runtime performance for generalization-aware subgroup discovery. The effects are especially beneficial, if the dataset contains selectors, which cover a majority of the dataset or if the maximum search depth of the discovery task is high.

Table 6.5: Runtime comparison (in s) of the base algorithm with traditional pruning based on the instances covered by a subgroup only(std) and the novel algorithm with additional difference-based pruning (dbp) for *numeric target concepts*. Each double column corresponds to a different maximum number d of describing selectors for a subgroup. As interestingness measures the generalization-aware mean test $r_{num}^{0.5}$ was used. "-" indicates that the algorithm did not finish due to lack of memory.

d	3		4		5		6	
Pruning	dpb	std	dpb	std	dpb	std	dpb	std
adults	1.9	2.2	5.5	11.0	11.4	49.9	17.4	263.4
autos	0.2	0.5	0.5	13.2	0.7	-	0.7	-
breast-w	< 0.1	< 0.1	0.1	0.3	0.1	1.3	0.1	6.0
concrete_data	0.3	0.3	1.0	2.9	2.1	29.6	3.3	-
credit-a	0.1	0.1	0.5	0.8	1.1	4.7	1.7	24.3
credit-g	0.3	0.2	1.3	2.1	4.1	16.2	7.4	99.1
diabetes	0.2	0.2	1.4	2.0	5.0	20.0	10.5	189.8
forestfires	0.2	0.2	0.9	1.4	2.2	6.8	5.3	59.6
glass	0.1	0.1	0.2	0.9	0.3	7.3	0.3	80.5
heart-c	0.1	0.1	0.5	0.8	1.8	6.1	3.6	43.4
housing	0.2	0.2	1.0	2.6	2.8	38.4	5.6	-
yeast	0.2	0.2	0.7	1.6	1.7	10.4	2.7	77.6

6.5 Summary

In this chapter, a new scheme of deriving optimistic estimates bounds for subgroup discovery with generalization-aware interestingness measures was proposed. In contrast to previous approaches on deriving optimistic estimate bounds, these are not only based on the anti-monotonicity of instances covered by a subgroup. Instead, they incorporate also the number of instances, which are covered by the subgroup, but not by its generalization. The optimistic estimates have been incorporated in an efficient algorithm that outperforms previous approaches, in many cases by more than an order of magnitude. The speed-up is especially high if the dataset contains selection expressions that cover a large part of the dataset.

7 Local Models for Expectation-Driven Subgroup Discovery[1]

This chapter introduces a novel family of interestingness measures, which enables the identification of previously undiscovered interesting subgroups and avoids uninteresting findings. This is accomplished by estimating the target share of a subgroup, which a data analyst would expect if statistics related to this subgroup are available. Deviations from these expectations are then considered as interesting. The novel interestingness measures can be formalized in a general framework, which also includes previously proposed measures as a special case.

7.1 Approach and Motivation

A key issue of subgroup discovery is to find appropriate interestingness measures, which select subgroups with an *interesting* target share. For that purpose, the share of the target concept in a subgroup is traditionally compared with the target share in the overall dataset. This chapter argues that this leads to subgroups that might be considered as uninteresting by users, as their target share is *expected* considering the effects of the influence factors that are combined in this subgroup. This problem is approached by adapting interestingness measures, such that subgroups are not scored isolated from each other. Instead, statistics of the influence factors that are combined in a subgroup are used to model the user's *expectation* of the target share in this subgroup considering the interaction of the influence factors with each other and the target concept. Then, the comparison between a subgroup's target share and the target share in the total population is replaced with a comparison to the formed expectation. In doing so, subgroups are selected, for which the actual target share deviates from the user's expectation.

As the expectation of a user is subjective, it is certainly not possible to capture it in one single mathematical function. Instead, this chapter introduces methods that find values, which are *plausible* and in line with human reasoning. Fortunately, a related task has been studied in depth in another research field before, that is, the research on graphical knowledge representations and in particular the elicitation of probability tables for the manual construction of *bayesian networks*. In this work, it is shown how techniques from this field can be transferred to subgroup discovery: Fragments of bayesian networks are generated for each candidate subgroup in order to capture the dependency between the influence factors and the target concept. This is used to determine plausible expectations

[1] The approach presented in this chapter has previously been published as [176]: Florian Lemmerich and Frank Puppe: Local Models for Expectation-Driven Subgroup Discovery. In *Proceedings of the 11th International Conference on Data Mining (ICDM)*, 2011.

for complex subgroups, that is, subgroups with conjunctive descriptions of more than one selector.

The contribution of this chapter is threefold: First, a formal generalization of traditional subgroup discovery is presented that allows for expectation-aware evaluation of subgroups. Second, several practical methods are presented to approximate human expectations for a subgroup's target share by constructing fragments of bayesian networks as local models. Third, the effectiveness of the presented techniques is evaluated in several experiments including two real-world case studies.

The approach of expectation-driven subgroup discovery is motivated in a two-part example:

Example 18 For an imaginary medical domain, consider the influence factors (subgroups) presented in Table 7.1.

Table 7.1: Motivational examples.

Influencing factors	Share of target concept
$\emptyset$ (*population*)	0.5
S	0.1
P	0.3
$S \wedge P$	0.5
$\emptyset$ (*population*)	0.5
A	0.7
B	0.7
C	0.5
$A \wedge B$	0.75
$A \wedge C$	0.75

The target concept describes the success of some treatment of a primary disease. The average success rate in the total population is 0.5. S is a secondary disease that lowers the chance of healing to 0.1. P is a pharmaceutical that is used only in difficult cases. As in these cases the treatment tends to be less successful, the share of the target concept given treatment with P is lower than in the overall population, it is at 0.3. Anyway, P helps in particular people that suffer also from the disease S, so in this case the success rate is at 0.5. This subgroup is probably interesting to domain experts, which are unaware of the interaction between S and P that leads to an unexpected high target share. In traditional subgroup discovery the subgroup $S \wedge P$, which describes patients suffering from S and also receive treatment P, is considered as uninteresting and is not discovered, because it does not have a higher share of the target concept in comparison to the overall dataset. In contrast to this, the novel approach presented in this chapter will formalize that the combination of S and P is expected to lead to a target share below 0.5, since both influence factors lead to a decrease of the target share individually. Therefore, the subgroup $S \wedge P$ is reported as interesting, as its target share is above expectation.

For the second part of the example, consider three other influence factors A, B and C, for which no interaction is previously known. The presence of either A or B increases the likelihood of a successful treatment from 0.5 to 0.7. The presence of C alone does not influence the target share. If A occurs together with B or together with C, then the target share is increased to 0.75. Since B leads to an increase of the target share in the total population, it is not surprising that it also leads to an increase for the patients, which are influenced by A. Thus, for $A \wedge B$ a target share, which is higher than the target share of A alone, could be expected. In contrast, C alone does not alter the target share of successful treatments. Given only this information, one would also expect no effect of C in the subpopulation of patients with influence factor A. But this is not the case in the example. As a consequence, the subgroup $A \wedge C$ is probably more interesting to users than the subgroup $A \wedge B$. This is not reflected in traditional interestingness measures for subgroup discovery, even if extensions for minimum improvements are used, cf. Section 2.5.4. For many applications, it is also not important that experts do not know the statistics for the single influence factors A, B or C beforehand: If a subgroup such as $A \wedge B$ is reported as interesting, then the data analysts will request and incorporate statistics for the single influence factors in the assessment of the discovered subgroups. □

The remainder of this chapter is structured as follows: Section 7.2 discusses relevant related work. Next, the general idea of the novel approach, *expectation-driven subgroup discovery*, is presented in Section 7.3. Section 7.4 introduces, how techniques from bayesian network research can be transfered to this setting. Afterwards, several possibilites to compute expectations in practice are presented in Section 7.5. In Section 7.6 computational issues are outlined. Then, Section 7.7 describes several experiments that show the effectiveness of our approach. A discussion of potential problems and possible improvements to the novel approach is provided in Section 7.8. Finally, the main results of this chapter are summarized in Section 7.9.

7.2 Related Work

Since selecting the most interesting patterns is a key issue for subgroup discovery, a large variety of interestingness measures has been proposed for this task. These are summarized in Section 2.5.3. Most of these measures focus exclusively on the statistics of the evaluated subgroup and do not incorportate statistics of other related subgroups.

Several improvements have been suggested in the past to avoid redundant results, which can be explained by other parts of the result, but most of them focus on the cover of the subgroups [91, 95, 104], see Section 2.5.5 for more details. However, in many applications it is preferred to report several subgroups as influencing factors on the target concept, even if they describe overlapping subpopulations, since domain experts might be unaware of this co-occurrence.

Considering adaptations based on the subgroup descriptions, it was proposed in [35] to apply a constraint on the minimum improvement for rule discovery. This constraint often leads to more useful results, but as demonstrated in the introduction it may suppress interesting subgroups and it may also still include predictable ones. This approach was

adapted in [29, 30] to include only subgroups with a statistically significant improvement over all generalizations, but this approach in general suffers from the same problems. Grosskreutz et al. presented a case study [102], in which they applied the minimum improvement function not as a constraint, but as a modified interestingness measure. This approach can be seen as a special case of the more general framework presented in this work. For more details on approaches in this direction, we refer to Section 2.5.4.

In the context of exception rule mining, Hussain et al. proposed another approach that scores the interestingness of a rule with respect to its generalizations, if these are already known *common sense rules* [124]. Their approach is difficult to comprehend for domain experts, and unlike our approach, it can not be generalized. Freitas proposes an adaption to interestingness measures that aims at including the *attribute surprisingness* [81]. In contrast to the novel approach of this work, this method is not suited to combine factors with opposing influences on the target concept and can also not be generalized.

For the related field of itemset mining, Tatti [235] uses entropy-based *flexible models* to determine itemsets that have a surprising support given its sub-itemsets. In the same area, Webb proposed an approach in which only those itemsets are considered as interesting that have higher support than expected under the assumption of independence in any partition of the itemsets [253]. Regarding this independence assumption as a very simple local model, the approach proposed in this work could be seen as a transfer of this general idea to the field of subgroup discovery. However, this transfer is non-trivial, since obviously each interesting subgroup implies an immediate dependency between the target concept and the variable describing a subgroup. Therefore, in this work it is proposed to utilize an *independence of the influences* on the target concept instead of an independence of the selectors/itemsets. Local models are used to determine the effects of these independent influences.

Estimating the probability distribution of a target variable given only some conditional probabilities and measuring the interaction of their influence are well studied problems in the field of bayesian networks. To the best of our knowledge, these estimations have not yet been used in interestingness measures for subgroup discovery. Bayesian networks have been proposed to store background knowledge and to find patterns deviating from this background knowledge by Jaroszewicz et al., but their approach includes no explicit mechanisms to avoid uninteresting combinations of influencing factors [127]. De Bie et al. proposed a general framework for pattern mining, in which prior information is encoded in a data model [40]. Then, maximally informative patterns with respect to this model are selected. While this work mainly follows a similar general idea, its focus on subgroup discovery and *local* data models allows for an easy application in practice. Recently, Kontonasios et al. summarized several approaches, which aim at mining "unexpected patterns" [153]. These techniques are concerned with integrating background knowledge in the discovery process. In contrast, the novel approach presented in this work does not rely on background knowledge, but only on the dataset itself.

7.3 A Generalized Approach on the Subgroup Discovery Task

This section presents a novel, generalized approach to construct interestingness measures for subgroup discovery. It is based on the expectations of users regarding the target shares for subgroups with complex descriptions, i.e., subgroup descriptions with more than one selector.

The approach is based on the following observations of the domain experts behavior during the analysis of subgroup discovery results:

1. Subgroups with conjunctive descriptions are usually evaluated with respect to related subgroups. In particular, the effects of the basic influence factors are investigated in order to assess the combination of influence factors that describes the subgroup. E.g., when a complex subgroup description $A \wedge B$ is presented to domain experts as a result of an automatic search, the experts additionally require to see statistics of its generalization A and its generalization B for the introspection of this subgroup. These are used to study the interaction with each other and with the target concept. As an alternative, it is inspected what effects a single influence factor has in absence of other influences. That is, for the result subgroup $A \wedge B$ also the subgroups $A \wedge \neg B$ and $\neg A \wedge B$ are considered.

2. Based on the knowledge of these statistics domain experts form an implicit or explicit impression, what the target share in a subgroup "should" be at most. This expectation is partially subjective, but follows general intuitions. It may also depend on prior knowledge.

3. The assessment of a subgroup then compares the actual target share of a subgroup in the data and the expected target share. The subgroup is considered as interesting if the formed expectation differs clearly from the actual target share.

To support this course of action in automated subgroup discovery, this work proposes to adapt the well-known Kloesgen measure $q^a_{Kl}(P) = {i_P}^a \cdot (\tau_P - \tau_\emptyset)$, that depends on the subgroup size i_P, the target share in the subgroup τ_P and the target share in the overall dataset $\tau_\emptyset$. It is adjusted into the new family of interestingness measures $q^a_{exp_\chi}$:

$$q^a_{exp_\chi}(P) = {i_P}^a \cdot (\tau_P - \chi(P)), a \in [0, 1],$$

where the expectation function $\chi(P)$ describes the expectation for the target share of the subgroup P that is formed by the inspection of related subgroups. The expectation value is the highest target share that can be explained by domain experts given this information. To increase the acceptance of the results, it can be helpful to use a function χ that is understandable for domain experts. Although additional background knowledge could be integrated in this framework, this work focuses on expectation functions $\chi(P)$ that take only statistical information contained in the dataset into account.

It is not assumed that the domain experts know all statistics, which are utilized to form the expectation, in mind a-priori. However, *if* a subgroup would be presented in the result set and then be analyzed by experts, then these statistics would be taken

into account to examine this subgroup. From this point of view, the novel approach can be considered as an anticipation of the subgroup analysis phase that follows the automatic subgroup discovery step: The automatic discovery integrates knowledge, which would come up in a more detailed introspection of a subgroup, directly into the interestingness measure and thus into the selection process of the automatic search. This is especially important, as for many application scenarios the domain experts' time is a major limiting factor and the analysis of subgroups can be tedious, especially for complex patterns with more than one selector. Additionally, the novel approach enables the identification of subgroups, which are interesting considering the additional information, but remain undetected using traditional subgroup discovery interestingness measures. This is exemplified in the case studies of Section 7.7. Although the general concept of expectation-driven subgroup discovery can also be extended to other types of attributes, this chapter is limited to a setting with binary target concepts only.

7.4 Expectations Through Bayesian Network Fragments

Since expectations are subjective, the expected target share $\chi(P)$ can be determined in many different ways. This work, however, focuses on models, which are inspired from the research on bayesian networks, cf. for example [206]. Bayesian networks are knowledge representations in form of probabilistic graphical models with directed and acyclic graphs. In these models, attributes (random variables) of a dataset are represented by graph nodes. Interdependencies between attributes are specified by directed edges between these nodes. Additionally each node has a corresponding conditional probability table, which assigns a probability to each attribute value for each combination of attribute values in the parent nodes.

Knowledge from domain experts can be utilized to construct bayesian networks. To make the knowledge acquisition process easier, different advanced techniques have been proposed: For example, when building a bayesian network, the complete probability distribution of an attribute must be specified in a conditional probability table. This is a difficult and tedious task for domain experts. In order to simplify the elicitation process, the entries of this table can be constructed with the input of only a few base probabilities. For instance, using such techniques the probability table of an attribute can be generated by providing only the probability distribution of this attribute for cases, in which exactly one of the binary parent attributes is set to true instead of specifying the distribution for all possible combinations of values in the parent attributes.

The problem of generating an expectation value for the target share of a subgroup P is transfered to the setting of bayesian networks as following: For each specialization $S \subset P$ a small local model is constructed by partitioning the subgroup description P into the two factors $A = S$ and $B = P \setminus S$. Thus, the description of P is equal to $A \wedge B$. Each local model uses the simple graph structure of Figure 9.2.

It contains three nodes for A, B, and the target concept T and two edges, which are directed from A and B to the target concept. The conditional probability table for the target concept T then contains the values $P(T|A,B)$, $P(T|A,\bar{B})$, $P(T|\bar{A},B)$ and

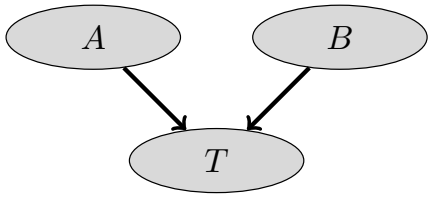


Figure 7.1: Basic structure of the bayesian network fragment used as a local model to determine plausible expectation values.

$P(T|\bar{A},\bar{B})$. The entry $P(T|A,B)$ is most important, as it is used as an estimation for the target share in the subgroup described by $A \wedge B$, that is, the subgroup P. It is assumed that the other values of this probability table are known to the experts, as they are a typical part of subgroup introspection. Based on these other values a plausible estimate for $P(T|A,B)$ is constructed, which is then compared with the actual target share for this subgroup.

Fortunately, the task of completing conditional probability tables that are plausible and in line with human expectations under these circumstances has already been studied in depth in the bayesian network community for the task of elicitation of probability tables by domain experts, see for example [259]. In this field several methods have been proposed, which utilize an assumption of *independence of the influences* on the target variable (in contrast to the independence of the variables themselves). Other techniques originate in the idea of measuring the *causal interaction* of two variables with respect to a target variable. One can then compute an estimation for $P(T|A,B)$ by assuming that this causal interaction is zero. The next section summarizes some of the suggested approaches to determine $P(T|A,B)$.

By using these techniques, one local expectation value is obtained for each model, that is, for each partition of the describing selectors. The expectation value for one model will be denoted as $\lambda(A,B)$. For subgroups with larger subgroup descriptions, there is more than one partition of this subgroup P in A and B. In this case the maximum of the expectations $\lambda(A, P \setminus A)$ is used as the result for the global expectation function:

$$\chi(P) = \max_{A \subset P} \lambda(A, P \setminus A).$$

Thus, the global expectation value is the highest value that can be explained by the interaction between a part of the influence factors of P and their complement (with respect to the subgroup description). As an additional lower bound for this expectation, the minimum of target shares of all generalizations is used. That means that the target share of a subgroup is never considered as unexpectedly high if it is lower than the target share of all its generalizations.

7.5 Expectations for Influence Combination

This section introduces different methods to compute an expectation for the target share in practice. It first shows how existing techniques can be modeled as a special case of the previously presented formalization of expectation-driven subgroup discovery. Then, different local models $\lambda(P)$ are presented, which are used to generate more plausible expectation estimations of the target shares.

7.5.1 Classic Subgroup Discovery

The most simple case for an expectation function is to use the target share in the overall dataset $\tau_\emptyset$ independent from the parent patterns A and B:

$$\lambda_{classicSGD}(A, B) = \tau_\emptyset$$

As a consequence, also the global expectation value is equal to the target share, since $\chi(P) = \max_{A \subset P} \lambda(A, P \setminus A) = \tau_\emptyset$. In doing so, the unexpected patterns correspond exactly to the results of classic subgroup discovery with Kloesgen interestingness measures q^a_{Kl}. Thus, classic subgroup discovery is a special case of expectation-driven subgroup discovery presented in this work.

7.5.2 Minimum Improvement

The minimum improvement constraint is often applied as an additional constraint in subgroup discovery, cf. [35]. It implies that a subgroup is only considered as interesting if its target share is higher than the target shares of all its generalizations. Of course, the amount of the minimum improvement can be used itself as an interestingness measure, as it was done for example in [102, 29], see also Section 2.5.4. This family of interestingness measures can be captured by the proposed framework using the expectation function

$$\lambda(A, B) = \max(\tau_A, \tau_B).$$

Thus, the global expectation is given by the maximum of all target shares τ_G in any generalization G of P: $\chi(P) = \max_{G \subset P} \tau_G$.

7.5.3 Leaky-Noisy-Or

The probably most popular model in bayesian network research to estimate the complete probability table of an attribute from few base probabilities is the *(leaky-)noisy-or model*, see for example [116] or [115]. It assumes the so called *independence of causal influence*. In contrast to traditional statistical independence, this does not imply that there is no interaction between the respective variables, but that the causal mechanisms by which they affect the target concept do not interact, cf. [69].

Using the noisy-or model for the graph structures of the local models, A and B are assumed to be the only two causes of T. They occur independently from each other and also influence T independent from each other. In the leaky-noisy-or model, an extension

of noisy-or, all potential other causes are modeled in an additional independent variable X ("leak parent"). Each of these three variables can cause an intermediate (hidden) variable to be true. If any of these intermediate variables is true, then also the variable for the target concept T is deterministically set to true. The value for $P(T|A,B)$ from the constructed table can then be considered as the estimation of the target share under the leaky-noisy-or assumption. As described for example in [202], the expectation estimation is computed as:

$$\lambda_{LN-Or}(P) = P(T|A,B) = 1 - \frac{\big(1 - P(T|A,\bar{B})\big) \cdot \big(1 - P(T|\bar{A},B)\big)}{1 - \big(P(T|\bar{A},\bar{B})\big)}.$$

7.5.4 Additive Synergy and Multiplicative Synergy

Additive synergy and multiplicative synergy are measures for the combination of qualitative influence, see for example [185, 186]. Both are used to describe qualitative behavior given two parent variables in a bayesian network. The *additive synergy* is defined by conditional probabilities as following:

$$Syn_{Add} = P(T|A,B) + P(T|\bar{A},\bar{B}) - P(T|\bar{A},B) - P(T|A,\bar{B})$$

If it is assumed that the influence of A and B is independent according to this measure, then the synergy is equal to zero: $Syn_{Add} = 0$. Thus, a (local) expectation function for $P(T|A,B)$ is derived by solving the equation for $P(T|A,B)$:

$$P(T|A,B) = P(T|\bar{A},B) + P(T|A,\bar{B}) - P(T|\bar{A},\bar{B}) := \lambda_{Add}$$

This follows the idea that if the presence of A alone increases the probability of T by e.g., 5% and the presence of B alone increases the probability by e.g., 3%, then the presence of A and B together should increase the probability of T by $5\% + 3\% = 8\%$.

Analogously, an expectation function can be constructed for the alternative measure *multiplicative synergy* by replacing the sum in the above formula by the product, resulting in:

$$P(T|A,B) = \frac{P(T|\bar{A},B) \cdot P(T|A,\bar{B})}{P(T|\bar{A},\bar{B})} := \lambda_{Mult}$$

Additive and multiplicative synergy are very easy to explain and using them to generate expectation estimates can lead to interesting results. However, there are also downsides of this approach, in particular if two strong positive or two strong negative factors are combined: In this case the computed value for the expectation deviates too much from the base target share. In extreme cases, values obtained by this formula even exceed the limits zero and one, cf. also Section 7.7.2.

7.5.5 Logit-Model

An alternative approach to measure *causal interaction* between influencing factors in bayesian networks is the *loglinear* or *logit* model, see [239]. A plausible estimation can be obtained by assuming, that this causal interaction between the influence factors is zero. The probability $P(T|A,B)$ can in this case be computed given the other entries of the conditional probability table.

In the logit model for bayesian networks, the conditional probability table for a binary node with two binary parents is specified by a set of equations: $\log \frac{P(T|\mathcal{AB})}{P(\neg T|\mathcal{AB})} = a + b \cdot z(\mathcal{A}) + c \cdot z(\mathcal{B})$, with $\mathcal{A} = \{A, \bar{A}\}$,$\mathcal{B} = \{B, \bar{B}\}$, and $z(X) = 1$ iff X is true. In particular, this describes the following four equations:

$$\log \frac{P(T|\bar{A},\bar{B})}{P(\neg T|\bar{A},\bar{B})} = a$$
$$\log \frac{P(T|A,\bar{B})}{P(\neg T|A,\bar{B})} = a + b$$
$$\log \frac{P(T|\bar{A},B)}{P(\neg T|\bar{A},B)} = a + c$$
$$\log \frac{P(T|A,B)}{P(\neg T|A,B)} = a + b + c$$

Since probabilities for the value combinations $(\bar{A},\bar{B})$, $(A,\bar{B})$ and $(\bar{A},B)$ are assumed to be known, the parameters a, b, c can be determined from the respective three equations. Assuming the so called *logit-non-interaction* the expectation value for the target share can then be computed as follows, see [239] for details:

$$\lambda_{Logit} = P(T|A,B) = \frac{e^{(a+b+c)}}{1 + e^{(a+b+c)}}$$

7.6 Computational Aspects of Mining Unexpected Patterns

The search space for expectation-driven subgroup discovery is the same as for classic subgroup discovery, so in principle any search strategies can be transferred directly. As for generalization-aware mining, the statistics of the single influence factors contained in the subgroup description are required in order to evaluate a subgroup. Therefore, depth-first-search, which is often applied for exhaustive subgroup discovery in the traditional setting, is not recommended. Instead, a levelwise search strategy such as apriori or breadth-first-search with extensive caching of subgroup statistics to avoid multiple computations is preferred. This guarantees that generalizations of a subgroup are always evaluated before the subgroup itself. The high memory requirements for these search strategies are less detrimental since the number of candidates stored in apriori-based algorithms is lower than the number of cached statistics.

Providing generally applicable pruning bounds for expectation-driven subgroup discovery is not an easy task since this depends on the exact computation of the expectation

values. Even for a fixed expectation function, e.g., the leaky-noisy-or model, it is difficult to estimate a maximum value for the difference between the expectation and the actual value in all specializations of a subgroup: Regardless of the target share of a subgroup, there might exist a specialization with a very low target value, which implies respective expectations. Then, further specializations of this specialization possibly have an unexpected high target share in comparison to this low value.

Nonetheless, for efficient mining the following trivial bound can be exploited: The actual target share of a subgroup never exceeds 1 and the minimum expectation is always at least 0. This can be used in combination with the anti-monotonicity of the subgroup size i_P to determine an optimistic estimate for any expectation-driven interestingness measure:

$$\forall S \subset P : q^a_{exp\chi}(S) = {i_S}^a \cdot (\tau_S - \chi(S)) \leq {i_S}^a \cdot (1 - 0) = {i_S}^a \leq {i_P}^a$$

Thus, ${i_P}^a$ is an optimistic estimate for the subgroup P, which is independent from the target share in the subgroup. If the required interestingness value of the result set is higher than ${i_P}^a$, then specializations of P have not to be considered by the search.

For our experiments, a fast bitset-based data representation was utilized in the mining algorithm to speed up the computations.

7.7 Evaluation

The effectiveness of the proposed approach was evaluated in several experiments and case studies, which are described in the following section. First, it is demonstrated that traditional interestingness measures are biased, i.e., they often result in a set of subgroups, which are specializations of only few basic influence factors. Then, the expectation values generated by the different proposed expectation functions $\chi(P)$ are assessed more systematically. Afterwards, the results of the novel approach are compared with results from traditional methods in two real-world applications regarding university dropouts and the discovery of spammers in a social bookmarking system.

7.7.1 Experiments with Public Data

Expectation-driven subgroup discovery was motivated by the deficits of traditional approaches in some scenarios. In particular, results often include many specializations of the same few high-impact influence factors. In the best case, novel approaches should be less effected by this redundancy problem.

This was assessed in a series of experiments on data from the public UCI data repository [13], which were performed separately for different expectation functions: In each dataset the top 5 basic selectors (single influence factors) were determined first. Afterwards the top 20 subgroups with conjunctive subgroup descriptions of two selectors were identified. For these 20 subgroups, it was counted how many were a specialization of one of the 5 top single influence factors. For this experiment, the size parameter a in the interestingness measures was set to $a = 0.5$. Results are shown in Table 7.2.

Table 7.2: The number of subgroups within the top 20, which are a specialization of one of the top 5 basic patterns.

Dataset	Classic SGD	Min. Impr.	SynAdd	SynMult	LN-Or	Logit
anneal	17	15	15	13	15	13
breast-cancer	17	10	8	6	8	8
breast-w	19	20	0	0	2	0
colic	14	10	3	2	3	3
credit-a	13	20	6	1	5	7
credit-g	16	15	7	5	8	6
diabetes	18	18	8	3	10	6
hepatitis	16	13	8	2	11	4
hypothyroid	3	14	0	0	0	0
labor	18	10	8	4	10	5
mushroom	16	14	4	3	4	3
primary-tumor	14	20	14	8	17	11
segment	20	20	18	17	19	17
soybean	20	20	20	17	20	17
spambase	19	20	11	1	20	16
vehicle	20	18	14	11	14	12
adults	18	20	18	0	20	3
splice	20	20	20	20	20	20
waveform-5000	20	20	18	9	19	17
median	18	18	8	4	11	7

It can be observed that for classic subgroup discovery as well as for the minimum improvement functions the majority of the results for a search depth of 2 are specializations of only few top influence factors. While this is not necessarily a bad thing for all applications, it can lead to the suppression of other potentially interesting subgroups. In contrast to these traditional measures, for the expectation functions *multiplicative synergy* and – to a lesser degree – for *additive synergy* and *logit* such specializations of the top subgroups are far less common. Finally, the *leaky-noisy-or* can be regarded as a compromise between these groups. While this does not automatically mean, interesting or useful subgroups are returned, it will be shown in subsequent experiments that leaky-noisy-or leads to good results in practice.

Table 7.3: Expectations according to different expectation function λ given the values of $P(T|\bar{A},\bar{B})$, $P(T|\bar{A},B)$ and $P(T|A,\bar{B})$. This table shows results for $P(T|\bar{A},\bar{B}) = 0.3$.

$P(T\|\bar{A},\bar{B})$	$P(T\|\bar{A},B)$	$P(T\|A\bar{B})$	λ_{Add}	λ_{Mult}	λ_{LN-Or}	λ_{Logit}
0.3	0.1	0.1	0.1	0.1	0.1	0.1
0.3	0.1	0.2	0.1	0.1	0.1	0.1
0.3	0.1	0.3	0.1	0.1	0.1	0.1
0.3	0.1	0.4	0.2	0.13	0.23	0.15
0.3	0.1	0.5	0.3	0.17	0.36	0.21
0.3	0.1	0.6	0.4	0.2	0.49	0.28
0.3	0.1	0.7	0.5	0.23	0.61	0.38
0.3	0.1	0.8	0.6	0.27	0.74	0.51
0.3	0.1	0.9	0.7	0.3	0.87	0.7
0.3	0.2	0.2	0.2	0.2	0.2	0.2
0.3	0.2	0.3	0.2	0.2	0.2	0.2
0.3	0.2	0.4	0.3	0.27	0.31	0.28
0.3	0.2	0.5	0.4	0.33	0.43	0.37
0.3	0.2	0.6	0.5	0.4	0.54	0.47
0.3	0.2	0.7	0.6	0.47	0.66	0.58
0.3	0.2	0.8	0.7	0.53	0.77	0.7
0.3	0.2	0.9	0.8	0.6	0.89	0.84
0.3	0.3	0.3	0.3	0.3	0.3	0.3
0.3	0.3	0.4	0.4	0.4	0.4	0.4
0.3	0.3	0.5	0.5	0.5	0.5	0.5
0.3	0.3	0.6	0.6	0.6	0.6	0.6
0.3	0.3	0.7	0.7	0.7	0.7	0.7
0.3	0.3	0.8	0.8	0.8	0.8	0.8
0.3	0.3	0.9	0.9	0.9	0.9	0.9
0.3	0.4	0.4	0.5	0.53	0.49	0.51
0.3	0.4	0.5	0.6	0.67	0.57	0.61
0.3	0.4	0.6	0.7	0.8	0.66	0.7
0.3	0.4	0.7	0.8	0.93	0.74	0.78
0.3	0.4	0.8	0.9	1.07	0.83	0.86
0.3	0.4	0.9	1	1.2	0.91	0.93
0.3	0.5	0.5	0.7	0.83	0.64	0.7
0.3	0.5	0.6	0.8	1	0.71	0.78
0.3	0.5	0.7	0.9	1.17	0.79	0.84
0.3	0.5	0.8	1	1.33	0.86	0.9
0.3	0.5	0.9	1.1	1.5	0.93	0.95
0.3	0.6	0.6	0.9	1.2	0.77	0.84
0.3	0.6	0.7	1	1.4	0.83	0.89
0.3	0.6	0.8	1.1	1.6	0.89	0.93
0.3	0.6	0.9	1.2	1.8	0.94	0.97
0.3	0.7	0.7	1.1	1.63	0.87	0.93
0.3	0.7	0.8	1.2	1.87	0.91	0.96
0.3	0.7	0.9	1.3	2.1	0.96	0.98
0.3	0.8	0.8	1.3	2.13	0.94	0.97
0.3	0.8	0.9	1.4	2.4	0.97	0.99
0.3	0.9	0.9	1.5	2.7	0.99	0.99

7.7.2 Values for Expectation Functions

In a second series of experiments, the expectation values, which are generated by the different introduced models, were computed for a wide range of settings. In particular, the additive synergy λ_{Add}, the multiplicative synergy λ_{Mult}, the leaky-noisy-or model λ_{LN-Or}, and the logit-model λ_{Logit} were investigated. Excerpts of the results for a base probability of $P(T|\bar{A},\bar{B}) = 0.3$ are shown in Table 7.3.

For example, the fourth line in Table 7.3 can be interpreted as following: Assume, there are two influence factors A and B. In this example, the share of the target concept is 0.3 in absence of both factors, it is at 0.1 if only A is present, and it is at 0.4 if only B is present. In this situation, the different models can be used to generate a plausible expectation for the target share in the subgroup $A \wedge B$. The additive synergy proposes an expectation value of 0.2, the multiplicative synergy a value of 0.13, the leaky-noisy a value of 0.23 and the logit model a value of 0.15.

All models have some properties in common: If both influence factors have a preventing influence on the target concept, then the generated expectation is on the lower bound, that is, the minimum target share of the single influence factors. Additionally, if one of the factors has no influence on the target share, e.g., $P(T|\bar{A},\bar{B}) = P(T|A,\bar{B})$, then all models expect a target share equal to the target share of the second influence factor. In most other situations, the expectation values differ. Additive synergy and multiplicative synergy create implausible high expectation values if two factors with a positive influence are combined. In some cases, the computed value even exceeds 1. The leaky-noisy-or and the logit models generate expectations, which are similar to each other. In comparison, leaky-noisy-or expects higher target shares if one factor with increasing and one with decreasing influence are combined. In contrast, when combining two factors with increasing effects on the target share, the logit model generates higher expectation values. Overall, these two models seem to generate "plausible" models in most cases. Since the leaky-noisy-or model matched our (subjective) expectations best, this model was used in the subsequent case studies in comparison to the traditional approaches. Computations for other probabilities $P(T|\bar{A},\bar{B})$, $P(T|\bar{A},B)$ and $P(T|A,\bar{B})$, which are not explicitly shown here, led to the same overall conclusions.

7.7.3 Case Study: Educational Domain

Next, a case study on the analysis of undergraduate students in a public university is described. Here, it is only focused on the effects of applying expectation-driven subgroup discovery, which was applied to identify influence factors on the dropout rate. More context and details on the overall project of this case study are provided in Section 9.1. This case study was performed in 2011 and was concerned with a dataset of 9300 students from certain degree programs[2]. The available data included matriculation information, final school exam grades and personal information, e.g., sex or birthday. Numeric attributes were manually discretized in pre-processing steps. The average dropout rate for all students in this dataset was 28.6% (71.4% non-dropouts).

[2]More details are omitted due to privacy reasons.

Table 7.4: Result subgroups in the university domain for the target concept *dropout* = *false*. The overall target share was **71.4**%. Statistics for the subgroup and the target shares if one or none of the contained influence factors is present, is displayed in subsequent columns. The ranking in the top 20 resulting subgroups for expectation-driven subgroup discovery using the leaky-noisy-or expectation function in comparison to the state-of-the-art approaches are shown on the right-hand side. An entry "-" means that the subgroup was not contained in the top 20 subgroups.

Subgroup	Selector A	Selector B	Size	$P(T\|A,B)$ = Target share	$P(T\|A)$	$P(T\|B)$	$P(T\|A,\bar{B})$	$P(T\|\bar{A},B)$	$P(T\|\bar{A},\bar{B})$	Classic SGD	Min. improvement	Leaky-noisy-or
S_1	grade=[2.0-2.5]	age=[20-21]	1140	80.1%	77.9%	74.1%	74.4%	72.1%	67.6%	1	6	-
S_2	previousDegreeProgr=f	grade=[1.5-2.0]	997	80%	72.1%	79.6%	70.9%	76.4%	68%	2	-	-
S_3	grade=[2.0-2.5]	previousDegreeProgr= f	1671	77.6%	77.9%	72.1%	79.8%	70.5%	67%	3	-	-
S_4	age=[20-21]	grade=[1.5-2.0]	772	80.4%	74.1%	79.6%	72.8%	77.9%	68%	4	-	-
S_5	grade=[2.5-3.0]	sex=female	1338	78.3%	75.2%	72.8%	72%	70.7%	69.2%	5	1	1
S_6	previousCourses= t	grade=[2.5-3.0]	82	85.4%	70.8%	75%	66.7%	75%	70.1%	13	2	6
S_7	grade=[0.66-1.5]	previousDegreeProgr=t	68	80.9%	61.2%	68.6%	58.8%	68.1%	73.1%	-	3	3
S_8	previousCourses= t	age=[20-21]	37	86.5%	70.8%	74.1%	69.1%	74.0%	68.7%	19	4	-
S_9	grade=[0.66-1.5]	age>30	13	92.3%	61.2%	60%	60.6%	58.0%	72.5%	16	5	8
S_{10}	grade>3.0	age=[26-30]	130	76.9%	67.5%	62.4%	66.8%	59.1%	73.8%	-	10	2
S_{11}	grade=[0.66-1.5]	sex=male	214	71.5%	61.2%	70%	55.8%	69.9%	74.4%	-	-	4
S_{12}	age=[22-25]	grade=[0.66-1.5]	112	75%	70.9%	61.2%	70.7%	58.2%	73.1%	-	15	5

Initially, the effects of the single influence factors with respect to the dropout rate were assessed. Then, subgroup discovery was applied to detect interesting combinations of these factors, in particular combinations with an increased share of non-dropouts. For this task classic subgroup discovery, generalization-aware subgroup discovery based on the minimum improvement and expectation-driven subgroup discovery using the leaky-noisy-or model was applied. The next paragraphs compare the results of these approaches by discussing the top five subgroups for each method. A size parameter of $a = 0.5$ was used for all interestingness measures in order to balance between target share deviations and instance coverage.

Top result subgroups that are described by a conjunction of two influence factors are displayed in Table 7.4: The labels in the first column are only for references in this work. The next columns provide the description of the respective subgroup followed by its size and target share. The following five columns give information about the target share in the generalizations of the subgroup, and the target share, if none or exactly one of the contained influence factors is present. The rightmost columns show the ranking of the respective subgroup for different expectation functions.

The results of the three approaches differ significantly. Minimum improvement and leaky-noisy-or have a higher overlap with each other than with classic subgroup discovery. In the top subgroup for classic subgroup discovery S_1, the influence factor $grade = [2.0 - 2.5]$ leads to a significant increase of the target share (non-dropouts) from 71.4% to 77.9%. Since also the second influence factor $age = [20 - 21]$ increases the target share (to 74.1%), it is not surprising that the combination of both patterns has an even higher rate of non-dropouts (80.1%). Therefore, this subgroup is not considered as interesting when using the leaky-noisy-or model. The subgroup S_1 is also included in the result set for the minimum improvement, as it shows an increase of the target share in comparison to all its generalizations. Whether this subgroup is considered as interesting or not depends on the personal preferences of the user.

In contrast, the result subgroups S_2, S_3 and S_4, which are returned only by classic subgroup discovery, are probably uninteresting for domain experts. They combine a very good influence factor with another selector that alters the target share only marginally. Therefore, they do not provide significant information and should (in the author's opinion) not be reported to decision makers.

The top subgroup according to the minimum improvement expectation function is subgroup S_5. This subgroup shows a high deviation of the target share in comparison to both generalizations. In absence of both influence factors of this subgroup, the target share is only 69.2% . The presence of only one influence factor increases the target share only moderately to 72.0%, respectively 70.7% Therefore, the target share of 78.3% for the subgroup S_5 is indeed unexpectedly high. Thus, this subgroup is also top ranked for the leaky-noisy-or function. The subgroups S_6 to S_9 are all relatively small and are therefore partially suppressed or ranked lower for the leaky-noisy-or function in favor of other (more) unexpected subgroups. An exception is the subgroup S_7: For this subgroup both influence factors alone have a negative impact on the target share, but the combination of both factors leads to a surprising increase of the target share. Thus it is also considered as interesting by the leaky-noisy-or model.

The subgroups S_{10}, S_{11} and S_{12} are ranked very low or are not discovered at all using classic subgroup discovery or the minimum improvement measure. Nonetheless, they were regarded as informative for the target application. For example, the subgroup S_{11} shows that although the sex has very little influence on the dropout rate in the total population, in the subpopulation of students with an exceptional good final school exam grade, male students dropout much less often. This can be interpreted as an *exception* from the general rule that very good students dropout more often. Similarly, the subgroup S_{10} describes that older students ($age = [26 - 30]$) with a bad grade ($grade > 3.0$) have a decreased dropout rate, although both of these factors alone lead to a clear increase.

For the official presentation of our analysis, the most interesting results were manually picked. These consisted of the top subgroups returned by the leaky-noisy-or function supplemented by some subgroups for minimum improvement. An automatic approach to combine these results is proposed in the discussion, cf. Section 7.8.

7.7.4 Case Study: Spammer Domain

In the second case study, it was tried to identify indicators for spammers in a social bookmarking system. The used dataset is derived from the publicly available data from the ECML/PKDD discovery challenge 2008[3]. It contains data for 31032 users. Each user is described by 25 attributes, including information on its profile (e.g., its username), its location (e.g, his IP-address) and activity-based features (e.g., the time difference between registration and the first post). Numeric attributes were discretized in a pre-processing step. A particular challenge for this dataset is the imbalance between the target classes, since the vast majority of 94.2% of the users are spammers. For more details on the dataset we refer to [160, 18].

Subgroup discovery in this domain aimed at unveiling groups of users that show an interestingly high share of spammers. As before, the experiments were performed for different expectation functions, i.e., classical subgroup discovery, minimum improvement and leaky-noisy-or. A size parameter of $a = 0.5$ was used in the interestingness measures. The top resulting subgroups that combine two influence factors are shown in Table 7.5.

The results for classical subgroup discovery and minimum improvement are overall similar. The top subgroups S_1 and S_2 for these approaches are "good" subgroup patterns since they have a large size and a high target share. However, in a more detailed analysis, the target shares in these subgroups are less surprising, as in both cases the single influence factors already have a significant effect on the spammer rate: In absence of both influence factors, there are only 78.3% (respectively 68.6%) spammers. This rate is increased to over 90% if one of the influence factors is present. Since the leaky-noisy-or predicts an even more increased target share for the combination of both factors in this case, it does not consider this subgroup as interesting. The same assessments also holds to a lesser degree for the subgroups $S_4 - S_7$.

[3] http://www.kde.cs.uni-kassel.de/bibsonomy/dumps/

Table 7.5: Result subgroups in the spammer domain for the target concept *spammer = true*. The overall target share was **94.2**%. Statistics for the subgroup and the target shares, if one or none of the contained influence factors is present, is displayed in subsequent columns. The ranking in the top 20 resulting subgroups for expectation-driven subgroup discovery using the leaky-noisy-or expectation function in comparison to the state-of-the-art approaches are shown on the right-hand side. An entry "-" means that the subgroup was not contained in the top 20 subgroups.

Subgroup	Selector A	Selector B	Size	$P(T\|A,B) =$ Target share	$P(T\|A)$	$P(T\|B)$	$P(T\|A,\bar{B})$	$P(T\|\bar{A},B)$	$P(T\|\bar{A},\bar{B})$	Classic SGD	Min. improvement	Leaky-noisy-or
S_1	tldcount=>15092	tascount=>33	11322	98,8%	96,6%	97%	94,8%	90,7%	78,3%	1	1	-
S_2	tldcount=>15092	date_diff=>1104	16191	97,8%	96,6%	96,3%	94,4%	91%	68,6%	2	2	-
S_3	date_diff=>1104	tascount=>33	11152	98,3%	96,3%	97%	94%	92,6%	88,6%	3	4	7
S_4	namedigit=>0	date_diff=>1104	8763	98,5%	97,8%	96,3%	95,8%	94,7%	87,1%	4	-	-
S_5	maillen=>17	tldcount=>15092	22026	96,9%	94,9%	96,6%	85,8%	94,4%	79,9%	5	-	-
S_6	maildigit=>0	tascount=>33	4672	99,2%	97,2%	97%	95,4%	96%	89,9%	-	3	-
S_7	namelen=>9	tascount=>33	6838	98,6%	96,7%	97%	94,9%	95,6%	89,2%	-	5	-
S_8	namedigit=0	tldcount=>15092	15173	95,6%	91,9%	96,6%	77,4%	98,3%	95,9%	-	-	1
S_9	namedigit=0	domaincount=2174-4473	4115	97,9%	91,9%	98,1%	90,2%	98,5%	97,7%	-	-	2
S_{10}	maildigit=0	date_diff=>1104	13871	95,6%	92,7%	96,3%	87,1%	97,9%	95,7%	-	-	3
S_{11}	realname2=>0	date_diff=>1104	12196	97,4%	95,4%	96,3%	88%	94,8%	90,7%	-	-	4
S_{12}	realnamelen=0	domaincount=>4473	3600	95,2%	91,5%	95,8%	88,9%	96%	94,8%	-	-	5

In contrast to these results, applying the leaky-noisy-or model leads to a completely different set of results: The target shares in the result subgroups are comparatively lower on an absolute scale. However, they can not be explained as easily. Consider for example the subgroup S_8, the top subgroup according to the leaky-noisy-or model. This subgroup describes users, who have no digits in their username (*namedigits=0*) and use an email address from a frequently used top-level domain (*tldcount>15092*). For users with a username without digits, the spammer rate is substantially decreased to 77.4%. In other words, the share of real users (non-spammers) is increased by about a factor of four. On the hand, the influence of the top-level domain on the spammer rate is moderate in the overall dataset. However, for users with a no-digit username, the information on the top-level domain is significantly more important: If both factors occur together, then the likelihood of a spammer is re-increased to 95.6%. The other top subgroups for the leaky-noisy-or expectation function show similar statistical properties.

Overall, it can be observed that the discovered subgroups differ significantly for the applied measures. Both types of subgroups can be of interest for the users, depending on the exact project goal. If the required type of subgroups is not clear beforehand, then a combination of results from novel and traditional approaches might be useful.

Expectation-driven subgroup discovery was also used to discover interesting subgroups with a larger description, i.e., subgroup described by a conjunction of three selectors. One of the top subgroups discovered in this setting is described by *(tasperpost=[0-1] ∧ tascount=[6-11] ∧ date_diff=[0-7])*. The statistics of this subgroup and its generalizations are displayed in Table 7.6.

Table 7.6: Statistics for an exemplary result subgroup of expectation-driven subgroup discovery that is described by a conjunction of three selectors.

Subgroup description	Size	Target share
tasperpost=[0-1]∧ tascount=[6-11]∧ date_diff=[0-7]	30	86.7%
tascount=[6-11]∧ date_diff=[0-7]	123	54.5%
tasperpost=[0-1]∧ date_diff=[0-7]	110	38.2%
tasperpost=[0-1]∧ tascount=[6-11]	188	81.9%
date_diff=[0-7]	899	53.5%
tascount=[6-11]	5103	93.2%
tasperpost=[0-1]	1588	79.5%
All instances	31034	94.1%

All single influence factors of this subgroup either have little influence or a strong negative influence on the target share. As could be expected, all combinations of two of these influence factors lead to a strong decrease of the target share as well. However, if all three of these influence factors are combined, then the target share is increased in comparison to all combinations of two of these factors. This potentially interesting subgroup is never discovered by the traditional methods since the target share of

(tasperpost='0-1' ∧ tascount='6-11' ∧ date_diff='0-7') is still lower than the target share in the overall dataset. Nonetheless, this example also shows that the explanation, *why* a certain subgroup is considered as interesting gets more complex and more difficult to comprehend for subgroups with longer descriptions. Therefore, the maximum number of selectors in a subgroup description in these case studies were limited to two or three, dependent on the specific task.

In summary, the application of expectation-driven subgroup discovery using a local model enabled in both case studies the identification of interesting subgroups with a target share that was higher than expected. Using traditional approaches, these subgroups often remained undetected.

7.8 Discussion and Possible Improvements

As shown in the previous sections, the mining for unexpected subgroups, e.g., by the leaky-noisy-or function finds qualitatively different subgroups than the traditional approaches on subgroup discovery: Result subgroups potentially have a target share that is *not* increased in comparison to all generalizations or the overall dataset, but are still considered as interesting, if the single influence factors that compose the pattern imply a stronger decrease. Such subgroups receive a negative score in subgroup mining with traditional measures $q_{Kl}(P)$ or minimum improvement-based interestingness measures $q_{MI}(P)$. Thus, they will never be discovered using traditional measures, even if the number of presented subgroups is increased, e.g., to the best 100 subgroups. While such subgroups are certainly interesting in some applications, this depends on the specific goal of the project and might not be the case in others.

If the required statistical properties of subgroups are uncertain, then a good solution can be, to combine the result subgroups for different interestingness measures. For that purpose, different approaches can be used. In the most simple case, the top subgroups for the different interestingness measures are compiled in a single list, e.g., by taking the best five subgroups of each approach. Since the minimum improvement can also be applied as an additional constraint, another solution could be, to use results for classic subgroup discovery with an additional filter for (significant) minimum improvement and supplement this list with additional results obtained from the leaky-noisy-or-model. In both cases, however, an overall ranking on the combined result subgroups is missing. Another combination approach generates such an overall ranking: The definition on interestingness measures with expectation functions allows for a comfortable method to combine results from different techniques directly in the search: For this purpose, a new (meta-)expectation function is created that combines expectation functions $\chi_1 \dots \chi_I$ with user defined weights $w_1 \dots w_I$ in a weighted sum:

$$\chi_{comb}(exp_i, w_i, sg) = \frac{\sum_{i=1}^{I} w_i \cdot \chi_i(sg)}{\sum_{i=1}^{I} w_i}$$

Mining subgroups using such a combined expectation function takes all involved expectation functions into account. Additionally, weighting the different functions allows the user to emphasize on the type of subgroups he is most interested in.

The target share expectation that a user forms on a specific subgroup is heavily influenced by the provided information. For the models in this work, i.e., the additive and multiplicative synergies, the leaky-noisy-or model and the logit model, the knowledge on the target share in presence of exactly one of the two influence factors is assumed, e.g., $P(T|A, \bar{B})$. If instead other statistics are used for the subgroup introspection, e.g., the target share for a generalization $P(T|A)$, models could be adapted accordingly. For example, as a rough approximation the equality $P(T|A, \bar{B}) = P(T|A)$ could be used.

Furthermore, the proposed models, in particular the leaky-noisy-or, assume in theory that the influence factors that are combined in a subgroup are conditionally independent from each other. They do not take the similarity of coverage of two factors into account. Therefore, the computed expectation for the target share might be off if two factors are strongly dependent from each other: For example, assume that the influence factor X has a decreased target share in comparison to the overall dataset and the factor Y has the same target share as the overall dataset. Based on this information, one would expect that the target share in $X \wedge Y$ is also decreased, since Y has seemingly no effect on the target share. This is also reflected by the computed expectation value from the leaky-noisy-or model. However, if it is additionally known that all instances covered by Y are also covered by X, i.e., $sg(Y) \subset sg(X)$, then the user's expectations change. Since in this case the coverage of $X \wedge Y$ is identical to the coverage of Y, the target shares in these subgroups are also identical. In this situation, the target share of $X \wedge Y$ is higher than the expectation computed by leaky-noisy-or. As a consequence, the subgroup $X \wedge Y$ is a possible result if the leaky-noisy-or is applied to calculate expectation values. However, the subgroup is obviously not interesting considering the covering information. One possibility to avoid such findings is, to ignore subgroups, which show a very large overlap (e.g., more than 90%) with a generalization. This adaptation has already been applied in the case studies of Section 7.7.

Finally, the framework in its current form takes only the absolute differences between the actual and the expected target share into account. However, a fixed increase of the target share is probably more important for very low or very high probabilities. For example, if the expected target share is at 50% then a difference of 3% to the actual target share of 53% is probably not that important. In contrast, a difference between 95% and 98% target share is substantial. These considerations could be used to further improve the proposed approach in future extensions.

7.9 Summary

This chapter introduced a novel family of interestingness measures in order to improve subgroup discovery results. The novel measures incorporate the *expectations* of the subgroups' target shares. These are implied by the statistics for the influence factors that are combined to describe the subgroup. To compute plausible values for these expectations,

which are in line with human reasoning, techniques from research on bayesian networks could be transfered to the subgroup discovery setting. In particular, it was proposed to generate fragments of bayesian networks as local models. Based on these models several techniques were proposed for the calculation of expectation values. The advantages of the novel approach were demonstrated in several experiments, including two real-world case studies. In both domains the novel approach enabled the identification of interesting subgroups that could not be detected using state-of-the-art algorithms. Advantages as well as potential problems with the novel approach have been discussed. This provided also some pointers to future improvements. Furthermore, the extension of the proposed framework in the direction of numeric target attributes and exceptional model mining as well as the integration of background knowledge into the local models seem promising.

8 VIKAMINE 2: A Flexible Subgroup Discovery Tool

Subgroup Discovery is not an isolated automatic process, but an iterative and interactive task, see also Section 2.7. To support the user in this course of action, advanced tool support is required.

For this reason, already early subgroup discovery algorithms have been embedded into powerful tools that provide a graphical user interface as well as options to visualize the results. These pioneering systems include Explora [140], KEPLER [262] and SubgroupMiner [143, 144]. Nowadays, the most popular data mining tools cover a wide variety of data mining tasks, for example RapidMiner[1], Knime[2], Orange[3] or Keel[4]. The integration of different data mining approaches in a single tool provides many benefits. However, the task of subgroup discovery is in these tools often only supported on a basic level with few discovery algorithms and visualization options. Often, subgroup discovery is only available in these tools as a plugin.[5] In addition, these multi-purpose tools are arguably more difficult to start with for practitioners. An open-source tool specialized on subgroup discovery is *Cortana*[6]. It provides an easy-to-use graphical user interface, multiple mining algorithms and options for result visualization. From an algorithmic point of view, *Cortana* places emphasis on heuristic search algorithms and exceptional model mining.

VIKAMINE is an alternative, freely available data mining tool that is also specialized on the task of subgroup discovery. In the context of this work, *VIKAMINE* was substantially improved and extended to the novel *VIKAMINE 2*. The improvements include additional algorithms, interactive mining approaches, pre-processing techniques, methods for result presentation, and a novel user interface.

This chapter first provides a general overview of *VIKAMINE 2*. Then, some contributions to *VIKAMINE 2* that were developed in the context of this work are highlighted. Previously available functionality of *VIKAMINE* has been described in [20].

[1] www.rapidminer.com
[2] www.knime.org
[3] www.orange.biolab.si
[4] www.keel.es
[5] As an example, http://kt.ijs.si/petra_kralj/SubgroupDiscovery/ provides plugins for Orange and RapidMiner are available.
[6] www.datamining.liacs.nl/cortana.html

8.1 *VIKAMINE 2*: Overview[7]

The *VIKAMINE* project has been started in 2003 by Martin Atzmueller. The software is implemented in Java and is freely available as open-source, licensed under the Lesser General Public License (LGPL). *VIKAMINE 2* is available for download at `www.vikamine.org`. In collaboration with the original author, the tool was significantly enhanced in the context of this work. The current features of *VIKAMINE 2* include:

- **State-of-the-art algorithms for subgroup discovery:** *VIKAMINE 2* comes with a variety of established and state-of-the-art algorithms for automatic subgroup discovery, e.g., beam search [167], BSD [177] or SD-Map [22, 24]. The algorithmic improvements introduced in chapters 4, 5 and 6 have also been implemented for *VIKAMINE 2*, partially as plugins.

  For the algorithms, the user can choose from a wide selection of interestingness measures, e.g., weighted relative accuracy, the binomial measure, the χ^2 measure, the lift, the t-score, the variance reduction and many more, see Section 2.5.3 for a description of these measures. Results can be optimized by applying discretization or filtering techniques that utilize relevancy or significance criteria.

- **Visualizations and interactive mining:** Powerful visualizations are essential to achieve a quick understanding of the data and mined subgroups. Visualizations implemented in *VIKAMINE 2* include, for example, pie visualizations, box visualizations, specialization graphs, and visualizations of subgroup statistics in the ROC-space, see also Section 8.6. Furthermore, *VIKAMINE 2* provides methods for the exploratory mining of subgroups, e.g., the zoomtable [25] or the dynamic subgroup tree, see Section 8.5.

- **Import/Export:** Datasets can be imported from different file-formats, for example, from CSV files in different configurations or from the ARFF format of WEKA. Modified or filtered datasets can be exported in the same file formats. Connections to SQL databases are enabled by plugins. Results can then be exported as plain text, to XML, or directly for presentation in standard Office Tools, e.g., MS-Excel.

- **Background knowledge:** *VIKAMINE 2* supports various methods to integrate background knowledge into the mining process, e.g., on default values or discretization cutpoints. Background knowledge can be acquired using form-based or document-based approaches, see also Section 8.7.

- **Extensibility:** Using the Rich Client Platform (RCP) of Eclipse, *VIKAMINE 2* can easily be extended by specialized plugins. Extension points allow, for example,

[7] This section contains contents previously published in [16]: Martin Atzmueller and Florian Lemmerich: VIKAMINE – Open-Source Subgroup Discovery, Pattern Mining, and Analytics. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2012.

for quick integration of new interestingness measures, search algorithms, visualizations, types of background knowledge and specialized views on the data into the graphical user interface.

- **Modularity:** *VIKAMINE 2* utilizes a strict separation between kernel components, i.e., data representations and algorithms, and the graphical interface components. Thus, the algorithmic core functionalities of *VIKAMINE 2* can be easily integrated in other systems and applications, e.g., for the integration in production environments, or for evaluations of algorithms by researchers, see also the following section.

8.2 Architecture

Application scenarios for a subgroup discovery tool are diverse: It may be used for interactive subgroup discovery by domain experts, for the regular automatic generation of documents as reports, or as a library to determine result subgroups, which are used as input for other machine learning or data mining algorithms. Many of these applications do not require the heavy-weight components used for graphical user interfaces or the generation of office documents, e.g., spreadsheets. To reduce the overhead in such applications, *VIKAMINE 2* pursues a strict separation of components:

- Vikamine Kernel: The kernel is the core of the software. It provides functionality for loading, storing and handling the dataset, and for mining subgroup patterns. It contains a wide collection of efficient subgroup discovery algorithms and interestingness measures as well as methods for filtering results. Mining tasks can be described using Java code or using a declarative XML specification.

- Vikamine Office: This is a minor addition to the kernel functionality that allows exporting subgroup discovery results in MS-Excel/LibreOffice Calc compatible file formats and to generate overview reports.

- Vikamine RCP: This component provides a visual interactive user interface for automatic and interactive subgroup discovery. It is described in more detail in Section 8.3.

- Vikamine Plugins: Plugins extend *VIKAMINE 2* with additional functionality. These include novel algorithms, interfaces for interactive mining, reports, additional handling of background knowledge and visualization options. Thanks to the used RCP-environment GUI components are integrated seamlessly into the existing interface.

8.3 An Extensible User Interface

The original *VIKAMINE* featured a *Swing/AWT*-based user interface. In the context of this work, this was replaced by an interface based on the Eclipse rich client platform

(RCP). This framework provides an appealing, industry-proven robust and extensible interface environment. Information is organized in views and perspectives, which can be freely rearranged by the users.

A screenshot showing some fundamental views of *VIKAMINE 2* is depicted in Figure 8.1: In the *workbench explorer* on the left-hand side the source files for different projects can be organized, e.g., datasets, text files providing background knowledge or stored options from previous mining sessions. In the top middle, the currently selected subgroup and its statistics are displayed. At its right-hand side, the *subgroup workspace* shows the results of the last discovery algorithm run or a set of user-selected subgroups. Below, the *zoomtable* allows for interactive mining of subgroups. On the right-hand side, the *attribute navigator* organizes the attributes of the dataset. It provides a variety of options in the context menu, e.g., the discretization of attributes or modifying the currently selected subgroup. Additional, more specific views are in the background or can be added from the menu. Several commands, e.g., for running an automatic discovery task, are accessible from the toolbar at the top.

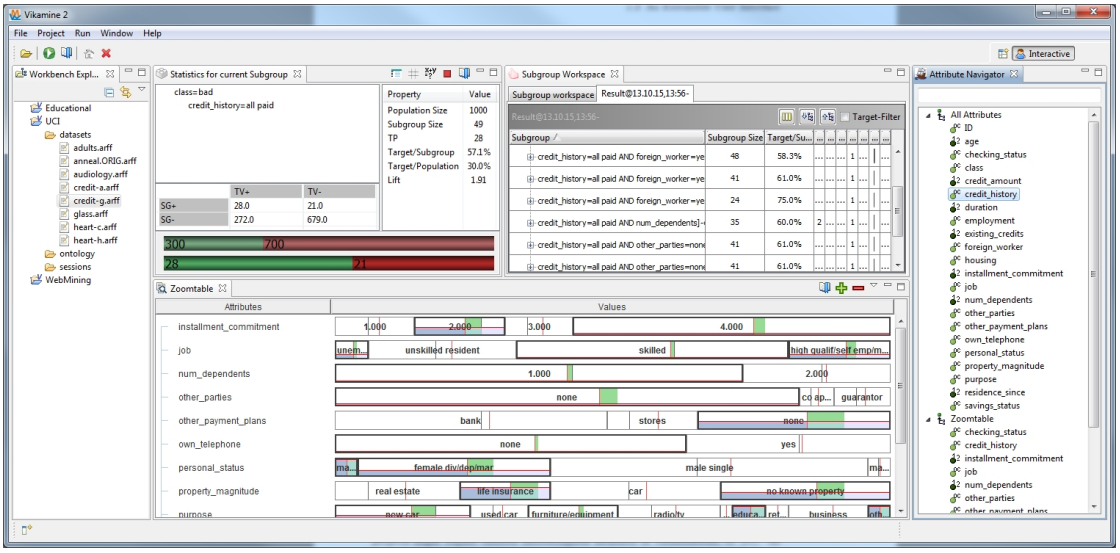


Figure 8.1: A screenshot with some fundamental views of *VIKAMINE 2*: On the left-hand side, the *workbench explorer* for project organization is shown. At the top, left of the middle statistics for the currently selected subgroup are displayed. Right of that, a set of result subgroups is displayed. At the bottom, the *zoomtable* view allows for interactive mining. On the right-hand side, the *attribute navigator* organizes the attributes of the dataset. Additional views can be opened, e.g., from the menu.

For data mining tools, *extensibility* is especially important since specific application projects often require tailored algorithms, interestingness measures or visualizations, cf. also [262]. As a key functionality, the RCP-based architecture of *VIKAMINE 2* allows for the simple development of extensions. These are created as (mostly) independent

plug-ins and are seamlessly integrated in the user interface without modifying the base source code. This is accomplished by implementing so called *extension points*. For example, *VIKAMINE 2* provides extension points for the quick integration of novel

- subgroup discovery algorithms,
- interestingness measures,
- visualization components,
- acquisition and handling of background knowledge, and
- additional views, e.g., for interactive exploration.

The plugin architecture is in particular important for a tool heavily used in research and education: It allows maintaining a small, but clean code base for fundamental components, but also enables integrated prototypes of experimental extensions.

8.4 Handling of Numeric Attributes

The original *VIKAMINE* had an emphasis on the nominal attributes in a dataset. *VIKAMINE 2* provides improved support for subgroup discovery with numeric attributes in different areas:

Numeric target concepts are now available for automatic and interactive subgroup discovery. For automatic discovery, a variety of algorithms are available, including the *SD-Map** and *NumBSD* algorithms presented in Chapter 4. To present subgroups with numeric targets appropriately, presentation methods have been adapted accordingly, e.g., the subgroup workspace or the visualization of patterns in PN-space (coverage space).

Numeric attributes can also be incorporated in the description of subgroups. The necessary discretization is performed either by manual elicitation using text-based or form-based knowledge acquisition, by automatic discretization, or by an interactive combination of both. Figure 8.2 shows the dialog that allows the interactive discretization of attributes with respect to the chosen target concept for subgroup discovery. On the top left-hand side an automatic discretization method can be selected, e.g., equal-width, equal-frequency, entropy-based, or chi-merge discretization, see Section 2.3.2. The table below allows manual refinement of the results of the automatic step. On the right-hand side, the distribution of the instances with respect to the current discretization is shown. The grey parts of the bars indicate the share of instances with a positive target concept. The name of the discretized attribute to be created is given at the bottom of the right-hand side.

8.5 A Scalable View for Interactive Mining

The interactive discovery of subgroups is a key application of *VIKAMINE*. However, previously proposed and implemented methods, e.g., the *zoomtable* or the *tuning table* [20], are severely limited in the number of involved attributes. To make interactive

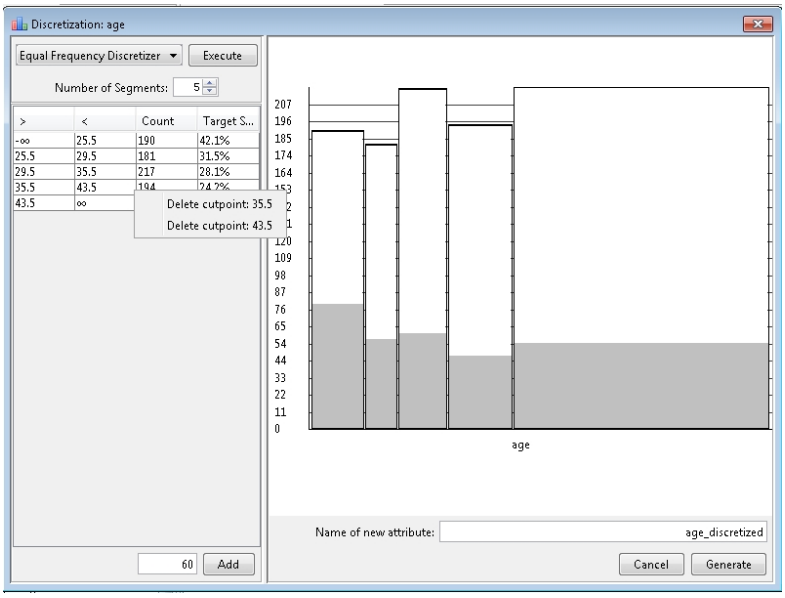
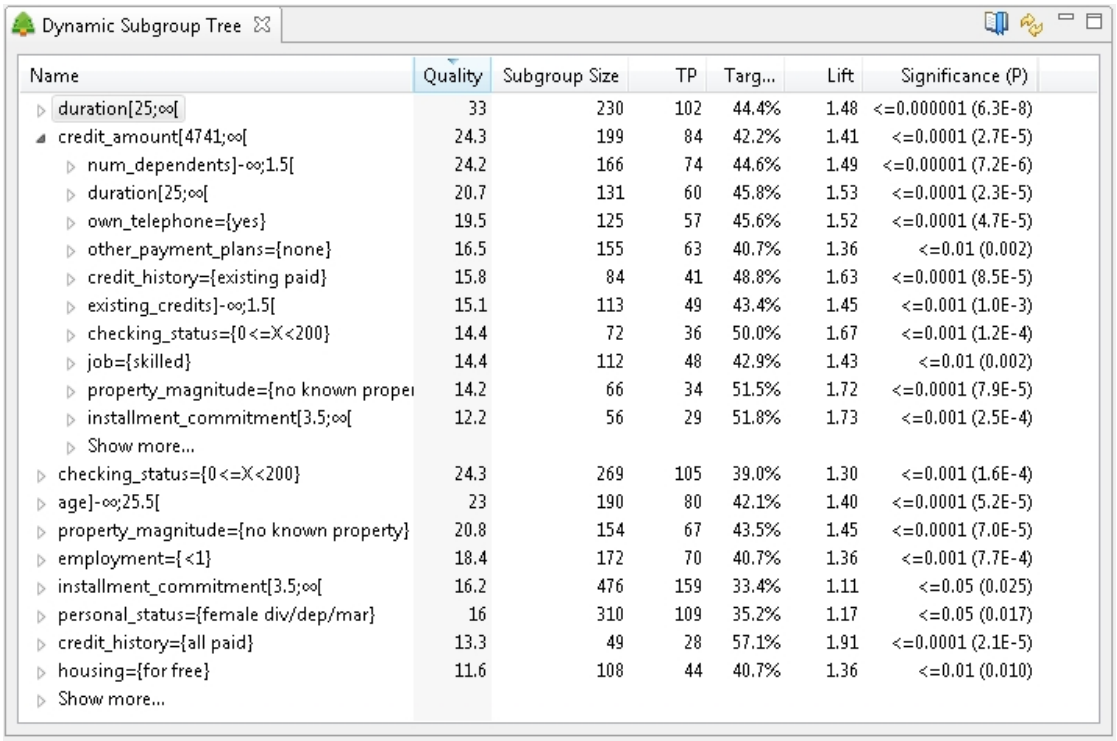


Figure 8.2: The discretization dialog in *VIKAMINE 2*: On the top left-hand side automatic discretization is performed. Below, the results can be manually refined. On the right-hand side the distribution of the instances with respect to the current discretization is shown. Grey parts in the bars indicate the share of instances with a positive target concept.

mining available for large datasets with many attributes, the *dynamic subgroup tree* has been developed as a combination of automatic and manual subgroup discovery. It is implemented as a *VIKAMINE 2* plugin.

In this view, each line of a table provides detailed statistical information on one subgroup. The initially shown subgroups are selected from the subgroups described by a single selector with respect to their score by a user chosen interestingness measure. Clicking on a subgroup dynamically expands the tree by computing and displaying the best direct specializations of this subgroup. More subgroups with the same generalization can be shown on demand. An example of a *dynamic subgroup tree* is displayed in Figure 8.3.

This view follows an intermediate approach between automatic and manual discovery: An automatic component identifies potentially interesting candidate subgroups at each level, while the user can guide the search by selecting for which candidates specializations are investigated, cf. also [75]. Computations for specializations are only computed on demand, i.e., if the user chooses to inspect specializations of the respective subgroup. In doing so, interactive discovery can also be performed in large datasets with hundreds of attributes. To improve scalability for large numbers of instances, interesting subgroups and their statistics are computed on samples of the dataset first. As soon as more precise information is available, the view is automatically updated.



Name	Quality	Subgroup Size	TP	Targ...	Lift	Significance (P)
duration[25;∞[	33	230	102	44.4%	1.48	<=0.000001 (6.3E-8)
credit_amount[4741;∞[	24.3	199	84	42.2%	1.41	<=0.0001 (2.7E-5)
num_dependents]-∞;1.5[	24.2	166	74	44.6%	1.49	<=0.00001 (7.2E-6)
duration[25;∞[	20.7	131	60	45.8%	1.53	<=0.0001 (2.3E-5)
own_telephone={yes}	19.5	125	57	45.6%	1.52	<=0.0001 (4.7E-5)
other_payment_plans={none}	16.5	155	63	40.7%	1.36	<=0.01 (0.002)
credit_history={existing paid}	15.8	84	41	48.8%	1.63	<=0.0001 (8.5E-5)
existing_credits]-∞;1.5[	15.1	113	49	43.4%	1.45	<=0.001 (1.0E-3)
checking_status={0<=X<200}	14.4	72	36	50.0%	1.67	<=0.001 (1.2E-4)
job={skilled}	14.4	112	48	42.9%	1.43	<=0.01 (0.002)
property_magnitude={no known prope	14.2	66	34	51.5%	1.72	<=0.0001 (7.9E-5)
installment_commitment[3.5;∞[	12.2	56	29	51.8%	1.73	<=0.001 (2.5E-4)
Show more...						
checking_status={0<=X<200}	24.3	269	105	39.0%	1.30	<=0.001 (1.6E-4)
age]-∞;25.5[	23	190	80	42.1%	1.40	<=0.0001 (5.2E-5)
property_magnitude={no known property}	20.8	154	67	43.5%	1.45	<=0.0001 (7.0E-5)
employment={<1}	18.4	172	70	40.7%	1.36	<=0.001 (7.7E-4)
installment_commitment[3.5;∞[	16.2	476	159	33.4%	1.11	<=0.05 (0.025)
personal_status={female div/dep/mar}	16	310	109	35.2%	1.17	<=0.05 (0.017)
credit_history={all paid}	13.3	49	28	57.1%	1.91	<=0.0001 (2.1E-5)
housing={for free}	11.6	108	44	40.7%	1.36	<=0.01 (0.010)
Show more...						

Figure 8.3: The *dynamic subgroup tree* plugin in *VIKAMINE 2*. Each line provides statistics of a subgroup. Clicking on a subgroup dynamically expands the tree by computing the best direct specializations according to a chosen interestingness measure. More specializations can be shown on demand by clicking on the "Show more..." node in the tree.

8.6 Result Presentation

Efficient communication of the result to the end-users is a crucial step in data mining. This is in particular important for descriptive techniques like subgroup discovery, cf. [91, 258]. The visualization of subgroups has always been a focus of the *VIKAMINE* system. Currently implemented visualizations include specialization-graphs, representations in ROC-space and PN-space, pie visualizations, stacked bar visualization, box visualizations, and clustering-based overlap visualizations, see Figure 8.4 for some examples. For a more detailed description of these implementations, we refer to [20, 14]. An overview on visualization techniques for subgroup discovery is also provided in [158], see also [93].

This section introduces two novel presentation methods for subgroup discovery results: The *pie circle* visualization and the *subgroup treetable*. These are in particular motivated by the intuition that subgroups with a conjunctive description should always be assessed with respect to their generalizations, see also Chapters 6 and 7.

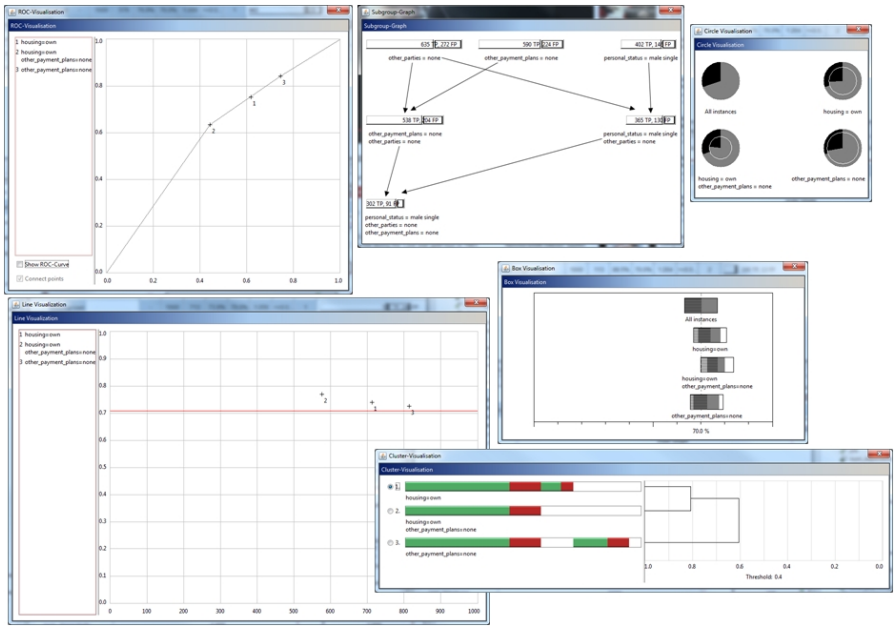


Figure 8.4: Exemplary visualizations in *VIKAMINE*: representations in ROC-space and PN-space, a specializations graph, pie visualizations, box visualizations, and clustering-based overlap visualizations.

8.6.1 Pie Circle Visualization

The pie circle visualization provides a quick overview on the most important influences on a target concept. It is constructed as a graph of nodes and edges. The nodes of the graph are laid out in a large circle in order to minimize overlaps between the edges. An example of this visualization illustrating the dropout rate in an educational domain can be seen in Figure 8.5.

A single influence factor (subgroup) is represented by a node that is rendered as a small pie chart, similar to [91]. The size of the subgroup is visualized by the node size: Larger nodes represent subgroups with a high instance count, smaller nodes represent subgroups that cover only few instances. The share of the target concept is displayed by the share of the filled part in the pie chart. For comparison, the target share for the overall dataset is indicated by a narrow black line in the pie chart. Additionally, the deviation of the target share in comparison to the overall dataset can be observed by the node color. That is, nodes are displayed in red color, if the target share of the respective subgroup is higher than in the overall dataset, in green color, if it is lower (or equal).

In automatic discovery algorithms, subgroups are scored by an interestingness measure. Such a measure can also be used to highlight the supposedly most interesting subgroups in the visualization: Nodes with a high score are displayed with full color, less intense/lighter colors are used for subgroups with low scores. In doing so, the initial view on the visualization is more focused on the most important subgroups, while less

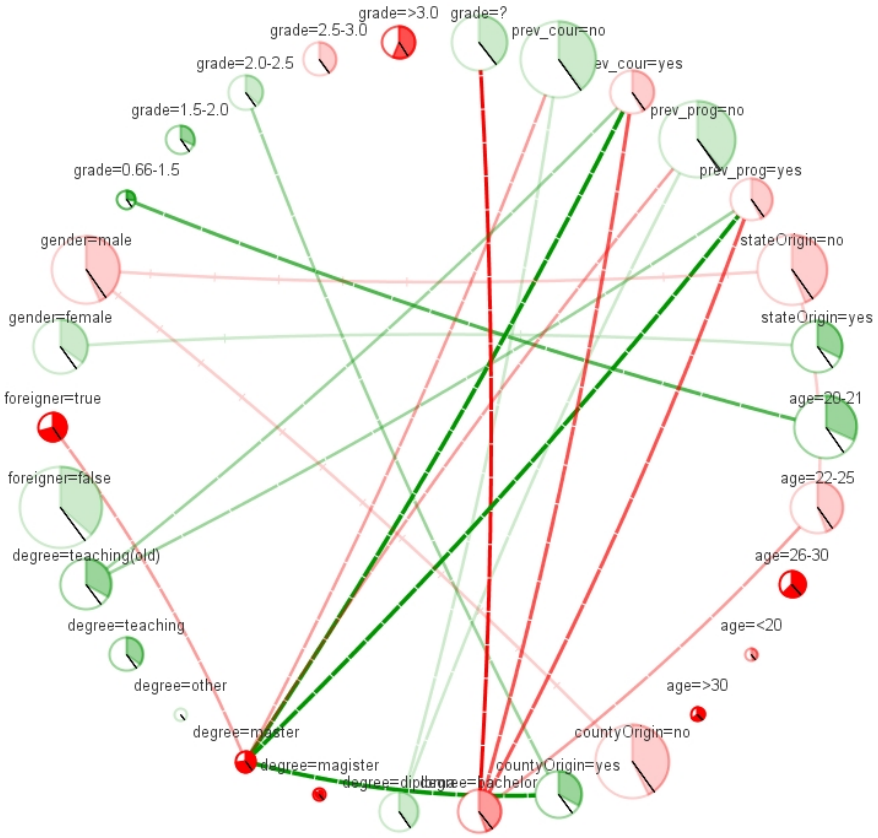


Figure 8.5: An example of a pie circle visualization in the education domain. As target concept *dropout = true* was used. A single influence factor is represented by a node. The size of the subgroup is visualized by the node size. The share of the target concept is displayed by the share of the filled part in the pie chart. Additionally, the deviation of the target share in comparison to the overall dataset can be observed by the node color. Edges provide information on the deviation of the target share for subgroups described by combinations of influence factors.

important ones are displayed unobtrusive.

For example, consider the node *foreigner=true* at the left of Figure 8.5. The color of this node is red since the chosen target concept – *dropout=true* – occurs more often in the subgroup described by this selector than in the total population. The filled part of this node indicates that for almost three quarters of this subgroup the target concept is true, while this is the case for only less than half in the total population as referenced by the black line. The used interestingness measure *relative added value*, that is, the minimum deviation of the target share with respect to generalizations of the subgroup, considers the node as important. Therefore, the intensity of the red color is high. However, the number of instances for this subgroup is quite low, which is indicated by the small size

of the node.

If a combination of influence factors shows an interesting deviation of the target share, this is indicated by an edge between these nodes: Red colors signal a target share that is *higher than expected* given the statistics of the incident nodes, green colors indicate a lower than expected target share. For possibilities to generate plausible expectation values for the target share of subgroups given the statistics of its generalizations, we refer to Chapter 7. As for the nodes, a high color intensity implies a high interestingness score using the chosen interestingness measure. Additionally, for important subgroups the edge width is slightly increased for emphasis.

For example, consider the edge between the nodes *foreigner=true* and *degree=master*. The red color of this edge implies that the subgroup *foreigner=true* $\wedge$ *degree=master* has a higher target share than expected given the target shares for each of the two more general subgroups. The applied interestingness measure indicates that this subgroup is of limited importance, the color intensity is low to medium.

The pie circle visualization is originally designed to present subgroups with at most two selectors. Nonetheless, by adding *nodes* for subgroups with larger descriptions, also more complex subgroups can be represented. In this visualization, a larger number of displayed subgroups requires a smaller node sizes to avoid overlap. However, as shown in the example up to 30 influence factors can be clearly presented, far more than by other visualizations that illustrate relations between subgroups, e.g., by a specialization graph. Exact statistics for the influence factors and their combinations are available via a tooltip. Of course, colors can be adapted for color blind users.

8.6.2 Subgroup Treetable

The *subgroup treetable* allows the interactive exploration of large result sets with standard office software, i.e., MS-Excel or LibreOffice Calc. In doing so, interactive mining can be performed by domain experts without additional tools or potentially unsecure online access to the data. For that purpose, data mining experts generate the interactive report with *VIKAMINE 2* and then send it to the domain experts. This report document contains a large set of candidate subgroups, e.g., by combining the results of different discovery algorithm runs. These subgroups and all their generalizations are arranged in a tree like structure in the document. Children in the trees are initially hidden by the "group and outline" functionality of the spreadsheets and are only shown on demand. This follows the idea of the *Information Seeking Mantra* [225]: First provide an overview, then zoom in and provide details on demand.

For each subgroup, different statistics are displayed, e.g., the number of covered instances, the target share in the subgroup, and the statistical significance of the deviation. Since subgroups with longer descriptions appear multiple times in the tree, the size of the tree grows fast. This can be avoided by filtering out repeated appearances of a subgroup, if the deviation of the target share is low in comparison to the subgroup of the parent node.

An example of a subgroup treetable in MS-Excel is provided in Figure 8.6. The first column shows the description of the subgroup. The next columns present the subgroup

	Description	Size	Target	Abs. significance	Rel. significance
3	age				
4	----age=20-21	1781	31.8%	<=0,000001	<=0,000001
5	-------- AND grade=0.66-1.5	95	13.7%	<=0,000001 (1,0E-7)	<=0,0001 (9,5E-5)
6	----age=22-25	1258	44.0%	<=0,001 (3,2E-4)	<=0,001 (3,2E-4)
7	-------- AND degree=Bachelor	272	52.9%	<=0,00001 (6,3E-6)	<=0,001 (8,4E-4)
8	-------- AND stateOrigin=no	878	48.2%	<=0,000001 (1,4E-8)	<=0,00001 (6,9E-6)
9	----age=26-30	323	62.2%	<=0,000001	<=0,000001
10	----age=<20	57	49.1%	<=0,2 (0,159)	<=0,2 (0,159)
11	----age=>30	86	62.8%	<=0,0001 (1,3E-5)	<=0,0001 (1,3E-5)
12	countyOrigin				
13	----countyOrigin=no	2518	42.9%	<=0,000001 (4,5E-8)	<=0,000001 (4,5E-8)
14	----countyOrigin=yes	987	32.8%	<=0,000001 (4,5E-8)	<=0,000001 (4,5E-8)
15	-------- AND degree=Master	8	00,0%	<=0,01 (0,008)	<=0,05 (0,018)
16	-------- AND grade=2.0-2.5	172	25.6%	<=0,0001 (7,1E-5)	<=0,05 (0,026)

Figure 8.6: A screenshot of a subgroup treetable presentation in MS-Excel. Each row presents statistics for one subgroup, e.g., the number of covered instances, the target share in the subgroup, the statistical significance of the target share deviation in comparison to the overall dataset and the statistical significance with respect to the generalization in the parent node. The signs – and + on the left can be used to hide and show branches of the tree.

size, the target share within the subgroup and the statistical significance with respect to the overall dataset using a chi-squared test. The rightmost column indicates the relative significance, that is, the significance using the parent subgroup in the tree as the total population for the statistical test. Colors indicate strong deviations in the target share with respect to the parent nodes. Branches of the tree can be shown or hidden in MS-Excel using the markings + and − on the left side.

The structure of the tree resembles the dynamic subgroup tree, see Section 8.5. However, in contrast to this view only pre-computed results are displayed. Since full subgroup discovery with higher search depths must be performed to generate the report, the creation process takes more time than the refinement steps in the dynamic subgroup tree. However, the pattern treetable report enables (limited) exploratory mining of subgroups for domain experts, who only use standard office tools.

8.7 A Framework for Textual Acquisition of Background Knowledge

Integrating background knowledge in the mining process is important for efficient mining, see also Section 2.6. It is often limited by the *knowledge acquisition bottleneck* [114, 113, 90], the problem of eliciting required knowledge from domain experts. In the previous versions of *VIKAMINE*, knowledge was acquired and used in separated dialogs that directly initiated actions. In research on knowledge acquisition, the paradigm of *document-centered* knowledge acquisition has been proposed as an alternative to this *form-based* approach: Users type their knowledge in a text document in an intuitive, easy-to-learn syntax. These are then processed by appropriate parsers to extract the

formal knowledge. This approach provides a variety of advantages, including low entry barriers, example-based authoring and versioning support [215].

In addition to the form-based approach, *VIKAMINE 2* also supports document-based knowledge acquisition using a predefined syntax: Source documents containing knowledge can be edited within the program using the integrated text editor. If documents in the respective folder of a mining project are created or edited, they are automatically parsed. The extracted, formalized knowledge is then saved in a central *reactive knowledge store* in the form of triples similar to a *RDF-Triple*, e.g., *<AttributeX; hasDefaultValue; none>*. The reactive knowledge store manages a set of triggers, which initiate actions if certain types of knowledge are inserted to the store. For example, a new attribute is created if a novel discretization scheme for a numeric attribute is provided. Additionally, interface views or mining algorithms can query the store for knowledge. The set of parsers to extract knowledge from the documents and the triggers for the knowledge can be easily expanded using extension points in order to adapt to project specific requirements. This course of action is also illustrated in Figure 8.7.

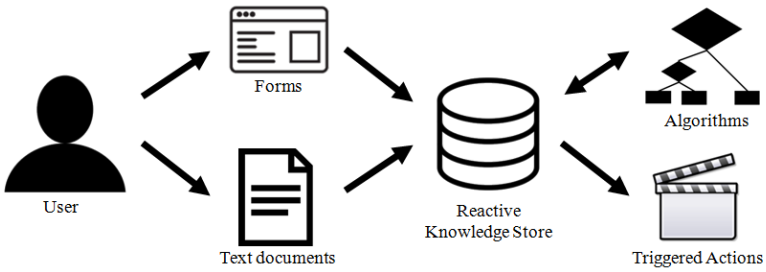


Figure 8.7: The main course of action knowledge handling in *VIKAMINE 2*: The user can insert knowledge using forms (dialogs) or using text documents written in an intuitive syntax. These documents are parsed and the extracted knowledge is stored in a central *reactive knowledge store*. This store manages a set of triggers. These are activated, if a specific type of knowledge is changed, and initiate respective actions. Additionally, the store can be queried from discovery algorithms or views for interactive mining.

The following simple examples of text fragments illustrate how some important types of knowledge can be formalized and used. However, the power of the approach lies not in these specific use cases, but in its easy extensibility with respect to novel knowledge types and actions, which allows adapting to the problem at hand quickly.

- **Default values:** Knowledge on default values can for example be used to ignore the respective attribute values in subgroup discovery algorithms.

  ```
  DEFAULT of previous_degree_program: false
  ```

- **Discretization:** This provides a natural segmentation of the domain of a numeric attribute. When knowledge of this type is inserted to the reactive store, then a trigger is activated to create a new discretized attribute.

```
DISCRETIZATION of starting_age: [18;20;25;30]
```

- **Abnormality information:** Similar to discretization information, entering this knowledge creates a new attribute with values *low, normal*, and *high*. This kind of background knowledge is in particular common in medical domains.

```
NORMAL of blood_pressure: 90-120
```

- **Value categorization:** This creates a new attribute that summarizes values of the original attribute. In the following example, the dataset contains an attribute *course_program* with the values *physics, chemistry, biology, history, law, literature* and some other values. The provided knowledge implies the creation of a new attribute *faculty* with three values: *other, science* and *humanities*, which are set in the instances according to the value of the attribute *course_program*.

```
CATEGORIZATION of course_program: faculty, DEFAULT other
- science
-- physics
-- chemistry
-- biology
- humanities
-- history
-- law
-- literature
```

- **Ordinality:** This type of knowledge provides an ordering for the values of an attribute. This can be used, for example, to generate selectors such as *highest_degree* $\geq$ *master*.

```
ORDINALITY of highest_degree: none < bachelor < master < phd
```

As shown by these examples, textual knowledge acquisition methods enable the intuitive elicitation of background knowledge, which can be used for more effective subgroup discovery.

8.8 Integrated Exploratory Data Analysis with EDAT

A pre-requisite for effective subgroup discovery is that users are familiar with the main characteristics of the analyzed data. For this task, *exploratory data analysis* [238, 118] has been developed in the field of statistics. This approach analyzes the distribution of a dataset by providing the user a large variety of visualization methods, e.g., pie charts, bar charts, histograms, mosaic plots, scatter plots and many more. A similar set of techniques is required for subgroup introspection, that is, the examination of potentially interesting subgroup patterns. Thus, techniques from exploratory data analysis do not

only allow obtaining an overview of the analyzed data. By visualizing distributions and dependencies in the subgroup as well as in the overall data, they also allow assessing and possibly explaining deviations of the target concept in the subgroup.

The *Exploratory Data Analysis Tool (EDAT)* was developed as a plugin to make methods for exploratory data analysis accessible in *VIKAMINE 2*, see also [187]. It can use exploratory techniques implemented in Java, but also functions as a bridge between the Java-based *VIKAMINE 2* environment and the statistical programming language *R*[8] [212]. For R a wide variety of statistical and graphical implementations are freely available, either in the core installation or in one of almost 5000 extension packages. However, R is primarily designed as a programming language for experienced users, who write code in a console. EDAT provides an intuitive graphical user interface for visualizations implemented in R to make these accessible to laymen. It offers a predefined, but extensible set of these visualizations directly integrated in *VIKAMINE 2*. Thus, datasets as well as subgroups can be easily inspected using one of many available visualization techniques implemented in *R*.

Figure 8.8 shows an exemplary screenshot of the EDAT plugin: In the top left view one of the available analysis methods is selected. Depending on the chosen evaluation, parameters can be specified below, e.g., which attributes are visualized. In the center of the screen, the result of the last command is displayed, in this example a density plot. Results can be exported as graphics or internally stored for later use in the history view, which is shown at the bottom of the screen. The view on the right-hand side provides additional information, e.g., the available attributes or the *R* code of the last generated evaluation. For users familiar with the *R* language, code can also directly be entered in a console (in this screenshot only in the background).

The available evaluation techniques presented in the user interface can easily be extended: Declarative XML-files specify the name, category and parameters of an evaluation as well as the (Java or R) code templates, which are to be executed to generate the result. Overall, the exploratory data analysis tool EDAT provides many visualization techniques to inspect the distribution of a dataset or a specific subgroup by integrating implementations written in R into the *VIKAMINE 2* user interface.

8.9 Summary

The subgroup discovery tool *VIKAMINE* was substantially improved and extended in the context of this work: *VIKAMINE 2* provides a broad collection of state-of-the-art algorithms and interestingness measures for mining with binary as well as numeric target concepts, including the novel algorithms presented in the previous chapters. A new graphical user interface based on the Eclipse RCP framework enables easy extensibility. In addition, new options for interactive mining and result presentation were developed, e.g., the dynamic subgroup tree, the pie circle visualization and the subgroup treetable. Furthermore, the plugin EDAT adds a large range of visualization options for inspecting

[8] www.r-project.org

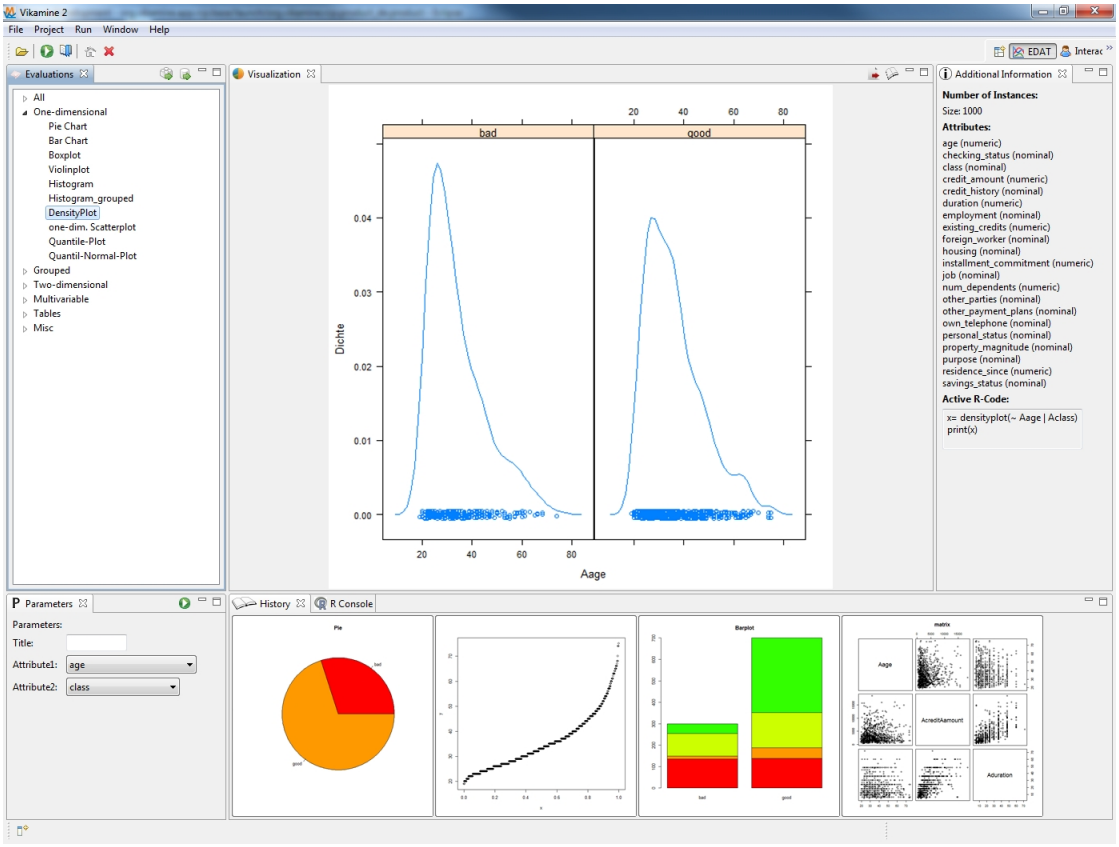


Figure 8.8: The *EDAT* perspective in *VIKAMINE 2*. In the top left view, the user chooses a visualization. Below parameters for this visualization can be selected. The central view shows the last executed query. In the bottom middle, the history of results is displayed. The view on the right-hand side provides additional information.

the distribution in the overall dataset as well as in specific subgroups by providing a bridge to the programming language R.

9 Applications: Case Studies in Subgroup Discovery

This chapter presents several case studies that were performed in the context of this work. They exemplify how subgroup discovery can successfully be applied in real-world scenarios using the developed methods and algorithms. First, an ongoing long term project regarding the analysis of educational data from the university domain is introduced. In this context, it is described, how subgroup discovery can reveal influence factors on the success and the satisfaction of students. Furthermore, an extended case study concerning the mining of social web data is provided, which aimed at describing locations using the tags that users provided for geo-referenced images. Additionally, the chapter outlines further applications of supervised pattern mining that were approached in the context of this work. These are concerned with the improvement of graphical models for information extraction, an evaluation of the case-based training system CaseTrain, the identification of influences that lead to faulty products in an industrial domain, and with the IPAT pattern mining challenge.

9.1 Students' Success and Satisfaction

Monitoring the curricula of university students provides great benefits for faculty staff: It allows for example to identify changes in enrollments, delayed graduations, difficult courses, and individual students in danger of dropout. By keeping these factors under surveillance, an early-alarm-system can be implemented in order to take appropriate actions in time or to evaluate the effects of applied changes.

The project *StudiPro* implements such a monitoring system for the University of Würzburg. It regularly generates a variety of compact reports, which present summarized information on different levels: For degree programs key numbers include the overall number of students, beginners, dropouts, graduations, female students, foreigners and several other statistics. On course level, the average grade, the rate of failed exams and the rate of exams passed too late in the curriculum are key statistics to identify critical teaching modules. Beside these key reports, further reports provide more detailed information on demand. Several of these tasks employ subgroup discovery as a data mining technique. These applications are described in the next sections in more detail.

Several concrete actions were at least partially motivated and/or monitored by this project: For example, individual students in danger of dropout have been invited to a discussion with a mentor. As another example, a previously joint course for several degree programs was split since it turned out to be too difficult for a part of the students.

The created statistics are currently also a required part of the yearly reports of the faculties to the heads of the university for quality control in the degree programs.

9.1.1 Dropout Analysis[1]

A first analysis focuses on influence factors, which lead to the dropout of students, that is, the abandonment of degree programs without graduation. Dropouts often occur early in the degree programs, that is, in the first two semesters of the studies. Therefore, indicators should not be based on exam results of the students, but on information, which is already known at the beginning of the study. For that purpose, datasets are generated that include information on the place, type and grade of the school final exam, previous courses and degree programs on this university, and personal information such as gender and age. As target concept, the overall success of the studies was used, that is, dropout or graduation for the degree program. Separate analyses in that direction are generated and reported regularly in an automatic way, but these are limited to simple subgroup descriptions without conjunctions, see Figure 9.1. In addition, more detailed analysis reports are created on demand. These also contain interesting combinations of subgroups and can cover a wide range of different degree programs for more general findings.

However, the starting point for any more elaborated subgroup discovery was always given by an overview on the basic influence factors, either in a separate report as depicted in Figure 9.1, or in the respective view in *VIKAMINE 2*. This was also the case in the subsequent applications, even if it is not explicitly mentioned. Exceptions are only case studies, which already have a very large number of single influence factors, such as in social media case study, see Section 9.2.

One exemplary, interesting finding was that the school final exam grade is strongly correlated with the success of the university degree programs. This correlation is in line with educational research on this topic, cf. [83]. In this direction, an interesting detail could be observed for several degree programs: In general the dropout ratio is lower, the better the school final exam grade was. However, this is not the case for students with very good school grades. These drop out slightly more often then their colleagues with good, but not very good grades. For example, in a certain degree program, which is also analyzed in Section 9.1.2, the overall dropout rate of the 708 analyzed students was 30.9%. The 124 Students with a school final exam grade between 2 and 2.5 (1 is best) dropped out in only 12.9% of cases. However, for the subgroup of 71 students with the best final school grades (better than 2) the decrease was not as strong since they dropped out only in 25.4% of the cases, see also Figure 9.1.

Additional results for the analysis of student dropouts, which also cover combinations of influence factors, have already been presented and discussed in depth in the context of expectation-driven subgroup discovery, see Section 7.7.3 of this work.

[1] Some contents of this section have been published as [175]: Florian Lemmerich, Marian Ifland, and Frank Puppe: Identifying Influence Factors on Students Success by Subgroup Discovery, In *Proceedings of the 4th International Conference on Educational Data Mining (EDM)*, 2011.

1	**Degree Program X**			
2		Students	Dropouts	Share
3	*Overall*	*708*	*219*	*30,9%*
4	School final exam grade			
5	----School final exam grade <= 2	71	18	25,4%
6	----School final exam grade =]2;2.5]	164	23	14,0%
7	----School final exam grade =]2.5;3]	249	74	29,7%
8	----School final exam grade > 3	143	57	39,9%
9	Age			
10	----Age <= 20	230	57	24,8%
11	----Age =]20;21]	214	50	23,4%
12	----Age =]21;23]	171	61	35,7%
13	----Age > 23	93	51	54,8%
14	Foreigner			
15	----Foreigner=True	57	28	49,1%
16	----Foreigner=False	651	191	29,3%
17	Parallel degree program			
18	----Parallel degree program=True	106	24	22,6%
19	----Parallel degree program=False	602	195	32,4%
20	Gender			
21	----Geschlecht=Male	405	134	33,1%
22	----Geschlecht=Female	303	85	28,1%
23	Previous courses			
24	----Previous courses=True	139	58	41,7%
25	----Previous courses=False	569	161	28,3%
26	Previous degree program			
27	----Previous degree program=True	138	58	42,0%
28	----Previous degree program=False	570	161	28,3%
29	State origin			
30	----State origin=True	335	80	23,9%
31	----State origin=False	373	139	37,3%
32	County origin			
33	----County origin=True	224	66	29,5%
34	----County origin=False	484	153	31,6%

Figure 9.1: Example of a basic report on student dropouts for one degree program in a spreadsheet (translated). Colored rows highlight influence factor with a strong deviation of the target concept *dropout*.

9.1.2 Indicators for Thesis Grades

A similar dataset as for the dropout analysis can also be used for other target concepts that measure student success. For that purpose, respective attributes are added to the dataset, e.g., the final grade of the degree, the number of semesters until graduation, the continuation of studies in a master degree after graduation, or the grade of the (bachelor or master) thesis. Since these statistics are measured at the end of studies, they can also be related to student performances within the degree program, e.g., the average grade or the number of ECTS credits acquired in certain time frames. If the analysis focuses on a single degree program, then additionally grades and semesters for specific required courses can be used. In the following, some results for an exemplary analysis are outlined, which was concerned with influence factors on the final thesis grade of a certain established degree program:

Table 9.1: Some exemplary results for influence factors on the thesis grade (1 is best grade).

Description	# Stud.	Avg. thesis grade
Overall	*489*	*2.18*
Grade course A > 3.5	109	2.52
Grade course B > 3.5	83	2.51
Avg. grade all required courses > 3.5	14	2.46
Avg. grade all required courses > 3.0	286	2.42
Grade course A > 3.5 AND Grade course B > 3.5	24	2.98
Grade course C < 1.5 AND Avg. grade non-req. < 1.5	21	1.56
Grade course C < 1.5	98	1.99
Avg. grade non-req.< 1.5	70	1.93

The average thesis grade in this degree program was a 2.18. Regarding single influence factors, subgroup discovery identified in particular two required courses, whose grades are indicators for weaker thesis grades. For students with a bad grade (> 3.5) at these modules, the average thesis grade was significantly worse than 2.18, that is, 2.52 and 2.51 respectively. Having a bad grade at these modules had an even stronger impact on the thesis grade than the average grade of all required courses, which was at about 2.4.

To find candidates for interesting combinations of influence factors, subgroup discovery with minimum improvement interestingness measures was applied. Since it was focused on subgroups with a large deviation of the target concept, the size parameter was set to a low value of $a = 0.1$. Interestingly, a search depth of three returned the same results as a search with a depth of two, indicating that no combinations of three describing selectors were considered as interesting by the chosen interestingness measure.

The most prominent finding was that if both of the previously identified critical modules were completed with a bad grade, then the average thesis grade was disproportionately worse, as it deteriorated to 2.98. A possible explanation for this phenomenon is that both courses are from quite diverse subfields of the degree program. If students have bad grades in different areas, then they will also have to choose a thesis topic from such a subfield. Another interesting finding was a subgroup that described student with a good grade in another, math-related course and also good average grades in the non-required courses. For these, the thesis grade was also very good on average, even though the results for these factors were less decisive on their own: If grades in both areas were good, then the average grade was 1.56. The respective value for the single influence factors were 1.99 and 1.93 respectively, see also Table 9.1.

In this scenario, the interactive nature of subgroup discovery was evident. While automatic subgroup discovery with minimum improvement measures generated interesting hypotheses, these were always compared manually to subgroups with related and "more general" descriptions. Here, however, the term "general" does not only refer to the strict

formal definition, but also to background knowledge: The average grade of all required courses is perceived as "more general" than the grade of a single module. For this task, the interactive GUI of *VIKAMINE 2* proved to be an essential support tool to compare subgroup statistics quickly. For the future, an extension of expectation-driven interestingness measures for numeric target concepts could be helpful to find further interesting subgroup patterns.

9.1.3 A Survey Analysis on Student Satisfaction

In another sub-project of StudiPro we conducted a large voluntary online survey, which was directed at all students of the University of Würzburg. Beside generating overview statistics for this survey, subgroup discovery was applied for a deeper analysis.

9.1.3.1 Dataset

The key question of the survey was how satisfied students are with their studies overall. Additional questions asked the students, which degree programs they study, how far they have progressed in their studies, how fit they feel for the requirements of their program, how well they feel informed about their situation, how much time they spend on studies and at additional jobs to earn their living, and which organizational issues affect their studies. Subjective impressions, e.g., about their satisfaction or current state of information, were given on a scale from 1 to 6, matching the German school grading system. Since one student can be enrolled in more than one degree program, one boolean attribute was used for each degree program. Additionally, degree programs were summarized by their field (e.g., *science, humanities, economics,...*) and the kind of degree (e.g., *bachelor, master, state examination, ...*). Thus, the overall dataset contained 119 attributes. To not decrease the response rate of the survey by privacy concerns, it was made completely anonymous. Therefore, it could not be linked to the students exam data. Overall, about 2800 student participated at the survey.

9.1.3.2 Influence Factors for the Overall Satisfaction

The main questions of this survey were concerned with the overall satisfaction of the students and which factors do influence this rating. For this task, subgroup discovery was applied with the attribute *satisfaction* as the numeric target concept. The mean target value using this attribute was at 2.61 (on a scale from 1-6, where lower numbers indicate higher satisfaction), showing an overall high satisfaction with the university experience. Numeric describing attributes were discretized manually to achieve more expressive values. Regarding the attributes for degree programs, only selectors were created, which refer to a *true* value of the attribute. That is, the search space contained no selectors that indicate that a student is *not* studying a certain subject.

As selection criteria, standard mean-based interestingness measures as well as generalization-aware measure using the minimum improvement were tested. Using a trivial search algorithm with a maximum search depth of 5, an exemplary task took up to one hour to compute. By utilizing a bitset-based data structure and the improved pruning

bounds presented in Chapters 4 and 6, the runtime could be substantially decreased: The same task took about 20 seconds using a bitset-based data structure without optimistic estimate pruning and only two seconds using the full *NumBSD* algorithm. Although subgroup discovery could also be performed with less sophisticated methods in this context, the improved algorithmic efficiency allowed to comfortably explore a large variety of different interestingness measures.

Result subgroups from the classic mean-based interestingness measures had higher absolute deviations of the target concept. However, they also had longer, more difficult to interpret descriptions and were often highly redundant to each other. For example, for a search depth of 2, 14 of the top 20 subgroups that indicate a decrease of the satisfaction were specializations of one single influence factor. In contrast, the results for generalization-aware measures offered a larger variety of subgroups that had shorter, simpler descriptions. Therefore, these results were overall more useful for the goal of identifying understandable influence factors. Since in this particular task higher deviations were considered as more important than large coverage, relatively low values for the size parameter a provided the (subjectively) best results. The exemplary results described next have all been discovered using a generalization-aware interestingness measure with the parameter $a = 0.3$. They are summarized in Table 9.2.

Table 9.2: Exemplary results for the survey analysis. These subgroups show an interesting deviation from the overall satisfaction level in the survey. The satisfaction was provided on a scale from 1 to 6, 1 being the rating for the highest satisfaction.

Description	# Students	Mean satisfaction
Overall	*2799*	*2.61*
Matching requirem. = [4-6]	258	3.46
Matching requirem. = [4-6] $\wedge$ Information = [4-6]	55	3.93
Overcrowded courses = true	887	2.89
Overlapping courses = true	1030	2.85
Degree program = Medicine	321	2.24
Degree program = Medicine $\wedge$ Semester= [1-2]	70	1.93

The overall satisfaction correlates strongly with how students feel up to their studies: Students, who do not feel to match the requirements (with a grade of 4 or worse on a scale from 1-6) have an overall satisfaction with their studies of 3.46, compared to the 2.61 in the total population. However, this is the case for less than 10% of the survey participants. The satisfaction decreased further if the students felt additionally not well informed about their situation. For the respective students the average satisfaction dropped to 3.93. Organizational issues also played a role for student satisfaction: Students, which were affected by overcrowded or overlapping courses, also were less satisfied with their studies overall, showing a rating of 2.89 and 2.85 respectively. As the high

number of concerned students indicates, these are perceived as wide spread problems at this university.

Regarding degree programs, some subjects had especially satisfied students: For example, students aiming at a medical degree had an average satisfaction rating of 2.24. In particular, students in the first semesters of this degree program were satisfied. They showed an average rating of 2.05.

Overall, subgroup discovery showed to be an effective tool for identifying influence factors in this setting. Subgroups with short descriptions were more useful for this task, as it was aimed at identifying general influences for student satisfaction. Formal statistical significance of the discovered subgroups in this setting was considered as not relevant in this context, as the voluntary participation in the survey already implies a bias. Although some of the findings could be expected beforehand, the comparison of the impact of different factors lead to interesting insights. Some of the previously assumed factors turned out to have little influence on satisfaction, e.g., the time required for the studies in semester breaks. While such findings could not be discovered using automatic discovery, they could easily be checked using the interactive *VIKAMINE 2* tool.

9.1.3.3 Gender Diversity in Student Satisfaction

Diversity, in particular gender diversity, has become an important topic in education in the last years. The overall survey results suggest that there is only a marginal difference in the overall student satisfaction between male and female students: Male students provided a mean satisfaction score of 2.56, while the mean score for females was 2.62. However, there may be certain conditions, which cause the satisfaction to differ between genders. To identify these conditions, we applied a variant of exceptional model mining. Classic subgroup discovery cannot be applied in this setting, since the target concept involves *two* attributes, the binary attribute *gender* and the numeric attribute *satisfaction rating*. As this problem setting is to the author's knowledge not covered by previously proposed model classes for exceptional model mining, a new model class had to be introduced. For this model class, all instances of the subgroup are partitioned by the binary target attribute, i.e., the gender attribute. For both partitions, the mean value of the numeric target attribute is computed. Then, the goal is to find subgroups, for which the difference of this mean value deviates in comparison to the respective difference in the overall dataset. This is weighted in an interestingness measure against the coverage i_P of the subgroup:

$$q^a_{meanDiff}(P) = {i_P}^a \cdot ((\mu_{F \wedge P} - \mu_{M \wedge P}) - (\mu_F - \mu_M))$$

Here, $\mu_{F \wedge P}$ is the mean value of the numeric target attribute for all females in the evaluated subgroup, $\mu_{M \wedge P}$ is the mean value of the numeric target attribute for all male students in the subgroup and μ_F, μ_M are the respective mean values for males and females in the total population. As an additional constraint, it was required that each partition (e.g., all males) of the subgroup covered at least 10 instances in order to limit the influence of single outlier instances.

Although this interestingness measure has not a well-founded statistical background, it was able to discover interesting subgroups in the application: Some degree programs showed significant differences between genders. For example, one specific subject in the area of natural sciences had a mean satisfaction rating of 2.5. The mean satisfaction for the 40 male survey participants of this subject was at 2.25, but that score was significantly lower for the 24 female participants (at 2.96). This can be seen as a hint that there might be diversity issues for this degree program. Similar, but not as strong tendencies could also be observed for some other degree programs for natural sciences. In contrast, for subjects from economic, medical and pedagogical fields female students were more satisfied with their study experience. Another interesting finding was that for the subgroup of students, who had to study more than 20 hours a week in the semester breaks and had more than 1 hour of journey to the university, the satisfaction rating differed by gender: The 10 male students were not disturbed by these conditions and gave a mean rating of 2.2, but the 27 females affected by these conditions provided much worse scores with a mean of 2.93.

In summary, the formulation as an exceptional model mining problem allowed to efficiently discover interesting subgroups of survey participants, for which the mean satisfaction rating differed between genders. Although a novel model class and interestingness measures had to be developed for this scenario, the required implementations could be incorporated in existing *VIKAMINE 2* algorithms with little effort. The implemented interestingness measures are now also available for other projects with similar problem settings, i.e., for finding subgroups, in which the difference of target mean values between two fixed groups deviates from the respective difference of mean values in the overall population.

9.1.4 Case Study Summary

This case study reported on the successful application of subgroup discovery in the university domain. In the application, simple results are generated regularly and automatically, while more sophisticated analyses are performed on demand. Three subtasks have been described in more detail: The mining for factors that influence the dropout of students, the identification of indicators for good or bad thesis grades and the analysis of a large survey on student satisfaction. The results showed the benefits of automatic discovery methods and of the interactive mining with *VIKAMINE 2*. For the survey analysis, also a new model class for exceptional model mining was introduced, which was able to identify subgroups of students, for which the average satisfaction differs between genders.

9.2 Mining Patterns in Geo-referenced Social Media Data[2]

In the last decade, *social media* have gained overwhelming popularity. These describe "Internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of User Generated Content" [132]. Analyzing the users' contributions in these systems does not only allow for describing or predicting the users behavior itself, but can also provide insights on the real world.

In an extended case study, we investigated exploratory subgroup discovery in social media using tagging information and geo-reference annotations. In this direction meta-data from the well-known photo sharing platform *Flickr*[3] was analyzed in order to obtain (sets of) tags, which are specifically used at a certain location, e.g., a landmark or a region of interest. This follows multiple purposes: It shows that techniques for supervised pattern mining can easily be adapted to analyze the tagging behavior of users, e.g., regarding the geographic distribution of the tagged resources. It also investigates how the users' tagging behavior forms emergent semantics, cf. [155], with respect to geographic locations. On the other hand, the obtained tags can be used to describe the target location. Such descriptions could for example be used in the future to automatically extract or rank tourist attractions in the target area, cf. also [8]: Since popular tourist attractions are photographed and tagged more often, the respective tags occur more frequently in the dataset, but in contrast to general terms (e.g., "people", "party") they are only used near a certain location.

Automatic mining techniques allowed to identify tags and tag combinations that are representative for one location on the map. Extensions to the *VIKAMINE 2* user interface enabled the visual exploration of tag usage, e.g., on a map.

9.2.1 Dataset

As the dataset for this case study, meta-data from the photo sharing platform Flickr was crawled. More specifically, all images that were uploaded in 2010 and contained a geo-reference (GPS-coordinate) located in Germany were considered. For these pictures the geo-references, the tags that users provided for this image, and the uploading user were extracted to generate a dataset. In this dataset each instance represents data of one image, each tag is a binary attribute, the username of the uploader is a multi-valued nominal attribute and the geo-coordinates are numeric attributes. To limit the size of the dataset, it was focused on tags with at least 100 appearances. As a result, the dataset contained about 11,000 binary attributes for more than 1.1 million instances. For the future, it is planned to crawl updated versions of this dataset to enable for example the analysis of the temporal development of tagging behavior.

[2] This section summarizes previously published work [172]: Florian Lemmerich and Martin Atzmueller: Describing Locations using Tags and Images: Explorative Pattern Mining in Social Media, In *Modeling and Mining Ubiquitous Social Media, Revised selected papers from the Workshop Modeling Social Media at the IEEE SocialCom*, 2012. Cf. also [17]: Martin Atzmueller and Florian Lemmerich: Exploratory Pattern Mining on Social Media using Geo-References and Social Tagging Information, *International Journal of Web Science*, 2(1-2):80-112, 2013.

[3] www.flickr.com

9.2.2 Automatic Techniques

To solve the problem of finding (sets of) tags, which are specific for a certain location on the map, subgroup discovery can be applied. The describing attributes are given by the utilized tag set. The construction of an appropriate target concept in this setting is slightly more complex. It is discussed in the following section. Afterwards, a method to avoid bias towards few very active users is described.

9.2.2.1 Target Concept Construction

To construct a target concept that indicates if certain (sets of) tags are used specifically for one location c, a distance function between the location of interest and each image location is determined. For that purpose, the position of c was specified by its latitude and longitude: $c = (lat_c, long_c)$. Similarly, for each image the value of the target concept is computed based on the geographic distance between c and the location $p = (lat_p, long_p)$ of the respective image.

Given latitudes and longitudes, the distance $d(p)$ on the earth surface of any point $p = (lat_p, long_p)$ to the specified point of interest $c = (lat_c, long_c)$ can be computed by:

$$d(p) = r_e \cdot \arccos(\sin(lat_p) \cdot \sin(lat_c) + \cos(lat_p) \cdot \cos(lat_c) \cdot \cos(long_c - long_p))$$

where r_e is the earth radius. Descriptions for specific locations are then given by sufficiently large subgroups, which minimize this distance. An example of an interesting subgroup could be described as: *"Pictures with this tag are on average 25 km from the specified point of interest, but the average distance for all pictures to the point of interest is 455 km"*.

Using this kind of target concept cannot find descriptions, which are specific to more than one location: For example, the tag "olympic" can be a regarded as specific for the Berlin olympic stadium. However, since there are also other olympic stadiums (e.g., in Munich) the average distance for the tag "olympic" to the berlin stadium is quite large. Therefore, we propose two alternative measures, which are more robust in that regard:

First, the neighborhood function $neighborhood(p)$ considers tags as interesting if they occur relatively more often in a radial surrounding of the specified location than in the total dataset.

$$neighbor(p) = \begin{cases} 0, & \text{if } d(p) < dist_{max} \\ 1, & \text{else,} \end{cases}$$

with $dist_{max}$ being a user chosen parameter. For example, the target concept for an interesting pattern in this case could be described as: *"While only 1% of all pictures are in the neighborhood of the specified point of interest, 33% for pictures with tag x are in this neighborhood."* The downside of this approach is, however, that it is strongly dependent on the chosen parameter d_{max}.

Second, the *fuzzy neighborhood* function reduces the strictness of this approach: Instead of a single distance d_{max}, a minimum distance d_{lmax} and a maximum distance

d_{umax} is defined for the neighborhood. Images are counted fully to the neighborhood if their distance is smaller than d_{lmax}, but only partially for distances between d_{lmax} and d_{umax}:

$$fuzzy(p) = \begin{cases} 0, & \text{if } d(p) < d_{lmax} \\ \frac{d(p)-d_{lmax}}{d_{umax}-d_{lmax}}, & \text{if } d(p) > d_{lmax} \text{ and } d(p) < d_{umax} \\ 1, & \text{otherwise} \end{cases}$$

In doing so, results are less sensible to slight variations of the chosen parameters since there is a smooth transition between instances within or outside the chosen neighborhood.

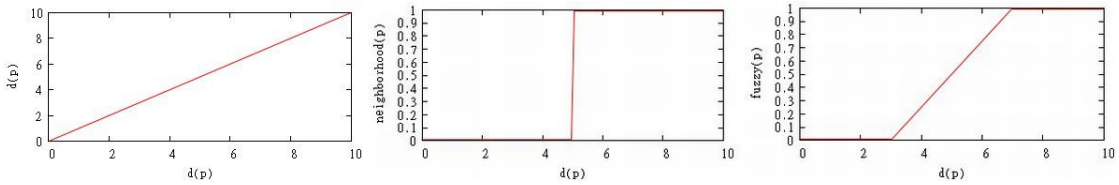


Figure 9.2: The three proposed distance functions $d(p)$, $neighbor(p)$ with a threshold of $dist_{max} = 5$ and $fuzzy(p)$ with thresholds $d_{lmax} = 3$ and $d_{umax} = 7$ as a function over $d(p)$. It can be observed that $d(p)$ is (obviously) linear, $neighbor(p)$ is a step function, and $fuzzy(p)$ combines both properties in different sections.

Figure 9.2 depicts the described options. For more details on the construction of the target concept, it is referred to [172]. Depending on the applied method to construct a target concept, the target is either a numeric or a binary target attribute. Therefore, traditional interestingness measures for subgroup discovery can be utilized to weight between the coverage of the subgroup and the deviation in the constructed target concept in comparison to the overall dataset. However, the target share (or the mean value respectively) is to be minimized in order to get descriptions with a low distance to the location of interest.

9.2.2.2 Avoiding User Bias: User–Resource Weighting

In social media, often few users make up for a large part of the contents. Therefore, the discovered subgroups are potentially highly biased towards the contributions of these users. As an extreme example, consider a single "power user", who shared hundreds of pictures of a specific event at one location and tags all photos of this event with several unusual tags. Without appropriate adaptations, these tags would then be identified as good descriptions for this location.

To approach this problem, a weight $w(i)$ is applied to each instance i. This weight is smaller if its owner contributed many images: When computing statistics of a subgroup, e.g., the overall count or the mean target value, the contributions of the instance is

weighted by $w(i)$. For our experiments, a weighting function of $w(i) = \frac{1}{\sqrt{u_i}}$ was used. Here, u_i is the number of images with the same owner as i.

Instance weighting is supported by SD-Map as well as other important subgroup discovery algorithms since it is also applied in other variants of the standard setting, e.g., weighted covering, see Section 2.5.5.1. Therefore, it can be incorporated efficiently into the automatic mining process.

9.2.3 Visualization

The *VIKAMINE 2* user interface allowed for the interactive examination of the candidates. For this case study, in particular visualizations generated with the EDAT plugin, provided interesting additional information for subgroup introspection. As an example, consider the histograms in Figure 9.3, which was produced for the application example in the Berlin area, see Section 9.2.4: It shows for the subgroup of all images, which have been tagged with *brandenburgertor*, the distribution of distances to the actual location of this landmark. It can be seen in the left histogram that the tag is very specific, since the vast majority of pictures with this tag is within a 5 km range of the location. The histogram on the right-hand side shows the distance distribution up to 1 km in detail. It can be observed that most pictures are taken at a distance of about 200 m to the sight.

In addition to these generic subgroup visualization methods, specific views were developed that allowed for the tailored display of geo-spatial data. For example, a dialog allows for the interactive creation of distance attributes by selecting a point on a dragable and zoomable map. The *tag map* view visualizes the spatial distribution of tags on a map by placing a marker for each picture of a selected subgroup. It also enables the comparison of different subgroups, as shown in the example in Figure 9.4. Using the plugin structure of *VIKAMINE 2*, cf. Section 8.3, these could seamlessly integrated into the existing user interface. However, due to scalability issues, it might be favorable to work on samples of the dataset in some cases.

9.2.4 Application Example: Berlin Area

The approach of describing location by using tags of *Flickr* images was evaluated for several different locations. The next section focuses on a single one, which was concerned with descriptions for the centre of Berlin, Germany. More precisely, the location of the Brandenburger Tor was chosen as the target location. As results tags such as *berlin*, *brandenburgertor* or *reichstag* were expected.

First, it was investigated, which candidate tags were returned by an automatic search using the different proposed target concept options. To describe locations, only the presence (not the absence) of a tag was used as a selector in the search space. For the size parameter of the interestingness measure, an intermediate value of $a = 0.5$ was chosen.

For the standard mean distance function, the top results included several tags such as *Potsdam* or *Leipzig*, which are not specific for the location of interest, but for other cities in eastern Germany. This can be explained by the fact that these tags are quite

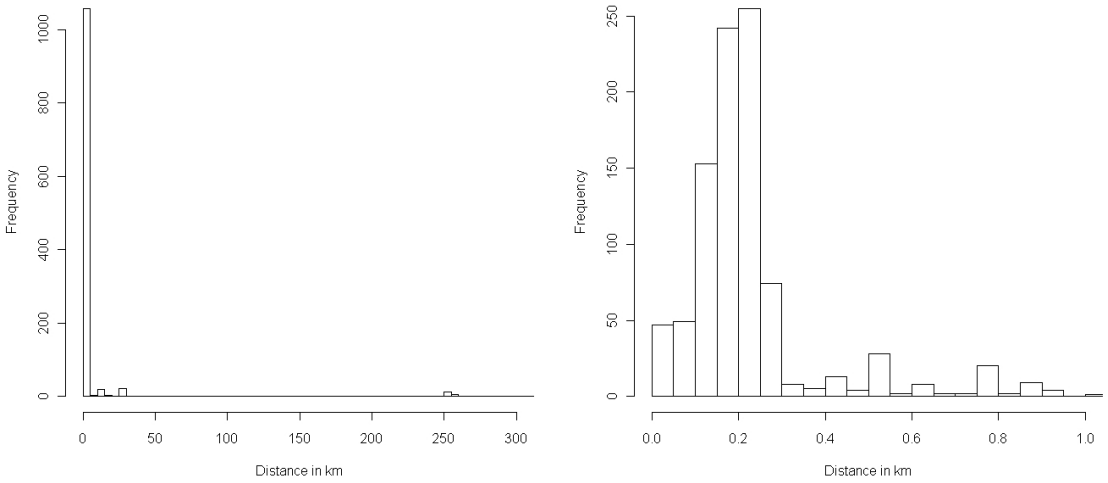


Figure 9.3: Histogram showing the distances of pictures with the tag "brandenburgertor" to the actual location. It can be seen in the left histogram that the tag is very specific, since the vast majority of pictures with this tag is within a 5 km range of the location. The histogram on the right side shows the distance distribution up to 1 km in detail. It can be observed that most pictures are taken at a distance of about 100 to 250 meters to the sight.

popular and the average distance for pictures with this tag is still lower than in the total population. Results for target concepts based on the neighborhood function are strongly dependent on the distance threshold parameter d_{max}: For a very small value of $d_{max} = 0.1$ km the results seem to be strongly influenced by noise since the number of pictures in that neighborhood is relatively small. For example it includes the tags *metro*, *gleis* (translated: "rail track") or *verkehrsmittel* (translated "means of transport"). While these tags should occur more often in urban areas, they are by no means the most representative tags for the area around the Brandenburger Tor. In contrast, for a parameter of $d_{max} = 5$ km top results included tags such as *tiergarten*, *kreuzberg* or *alexanderplatz*, which describe other areas in Berlin, but not specifically the area around the Brandenburger Tor. Finally, a fuzzified distance function with parameters $d_{lmax} = 1$ km and $d_{umax} = 5$ km as lower and upper bounds for the neighborhood threshold was applied. The results are displayed in Table 9.3. They indicate that this function forms a compromise between different parameter choices for the simple neighborhood functions. However, the results still are relatively general, as they describe the whole central part of Berlin. For more precise descriptions of a small target area, more data will be required.

Including user-based instance weighting as described above had a positive effect on the results: It functioned as a filtered of result tags, which have predominantly been used by only few very active users. For example, the tags *heinrichböllstiftung* and *karnevalderkulturen* were removed from the previously presented result set, as they were used by only few users. The discovered subgroups could be interactively and visually inspected with *VIKAMINE 2* using plugins that were especially developed for this application, see for

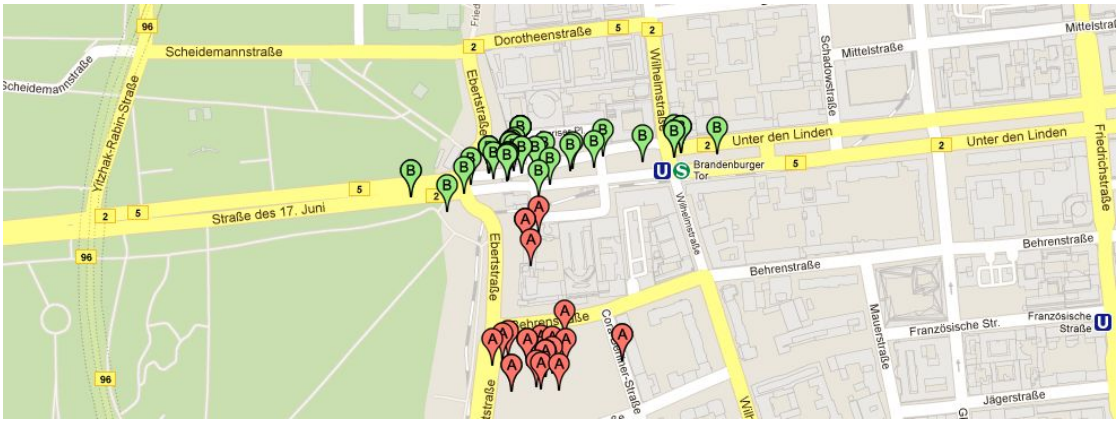


Figure 9.4: Example tag map visualization from the case study (zoomed in): Pictures with tag "holocaust" are marked with a red "A", while pictures for the tag "brandenburgertor" are marked with a green "B".

example Figure 9.3 and Figure 9.4.

For a more detailed description of the pre-processing steps and for further results (including combinations of tags), it is referred to the original publication [172], which also covers results for a second popular location in Hamburg.

9.2.5 Case Study Summary

This case study showed, how subgroup discovery can be utilized to extract knowledge about the world from user behavior in social media. In particular, descriptions for user-defined locations could be extracted from the tags that users provided for images on the Flickr image sharing platform. It was shown how an appropriate target concept could be constructed for this task. Additional methods for reducing bias towards heavy users and for inspecting the result subgroups were presented. Efficient implementations of subgroup discovery algorithm allowed for efficient automatic discovery in this large dataset. Tailored plugins for the user interface allowed for the intuitive visual inspection of the results in *VIKAMINE 2*.

Table 9.3: Brandenburger Tor: top patterns (description size 1) for the 'fuzzified' target concept distance function ranging from 1 km to 5 km.

Tag	Subgroup size	Mean target value
berlin	113977	0.46
reichstag	2604	0.05
potsdamerplatz	2017	0.05
mitte	3507	0.42
berlinmitte	3053	0.30
heinrichböllstiftung	1211	0.01
hauptstadt	2350	0.34
brandenburgertor	1136	0.10
alexanderplatz	1699	0.28
city	18246	0.76
tiergarten	2497	0.42
platz	2171	0.4
touristen	2815	0.47
nachbarn	3691	0.55
sonycenter	803	0.02

9.3 Pattern Mining For Improved Conditional Random Fields[4]

This work so far focused strongly on subgroup discovery as a knowledge discovery technique, that is, the mining of patterns which are aimed directly at domain experts. However, supervised pattern mining can also be exploited in black box machine learning techniques as subtasks within more sophisticated algorithms. Since the respective pattern mining subtasks are algorithmically very similar to subgroup discovery, the efficient techniques and implementations presented in this work can also be employed in this type of application.

In that direction, we used supervised pattern mining in a novel approach for information extraction in domains with context-specific consistencies. The general task was to extract structured information from a text segment, e.g., to extract the author and title fields from reference sections of scientific papers. State-of-the-art algorithms for this task are based on *conditional random fields* [163, 230]. These are undirected, graphical

[4]This section refers to the previously published works [145] and [146]: (1) Peter Kluegl, Martin Toepfer, Florian Lemmerich, Andreas Hotho, and Frank Puppe: Exploiting Structural Consistencies with Stacked Conditional Random Fields, *Mathematical Methodologies in Pattern Recognition and Machine Learning*, 30:111-125, 2013. (2) Peter Kluegl, Martin Toepfer, Florian Lemmerich, Andreas Hotho, and Frank Puppe: Collective information extraction with context-specific consistencies. *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2012. This section only outlines these approach from a subgroup discovery perspective.

probabilistic models, which are commonly applied in machine learning for predicting label sequences. The specific approach proposed by our working group exploits that instances are not all independent and identically distributed, but are generated in certain contexts that imply similar characteristics. For example, references originate from papers, which follow different bibliographic styles in different papers. However, they are usually coherently formatted in a single paper. This local consistency is exploited in two novel methods that both employ supervised pattern mining as subtasks:

In the first algorithm, two conditional random fields are combined in a stacked approach: Initially, a first conditional random field is trained as usual. Then, supervised pattern mining is applied on all output labels of a certain context to learn descriptions for its specific properties in a certain context. For that purpose, tabular training datasets are constructed, in which each token of the input sequence represents one instance. A subset of the input features of the conditional random fields is used for the describing attributes. For each possible transition between two labels, one target attribute is created. Then, for each target attribute one pattern mining task is performed to identify characteristic properties for this label transition. The pattern mining results are then used as additional features to train a second conditional random field that produces the final results. In the second approach, the features are similarly discovered, but are added as additional factors to extend the linear-chain conditional random fields. As a result, both approaches achieved an error reduction of about 30%. For more details on these approaches, it is referred to the original papers [147, 146], see also [145].

From a pattern mining point of view this was an interesting task, since efficient computation of the patterns is key: Although the datasets were only medium-sized – a typical dataset contained about 3000 instances and about 200 binary describing attributes – algorithmic efficiency was required, since many runs of the pattern mining task had to be performed in the inference step of the conditional random fields. For that purpose the bitset-based algorithm *BSD* was applied, cf. [177]. Additionally a new interestingness measure $q_{F1_{exp}}$ was requested for the pattern mining tasks:

$$q_{F1_{exp}}(P) = \frac{2 \cdot p_P}{i_P + p_\emptyset} \cdot \left(1 - \left(\frac{|i_P - E_y|}{\max(i_P, E_y)}\right)^2\right)$$

The left part of this measure is the traditional F_1-measure that describes how well the pattern reproduces the predicted transition. The right factor is a penalty term for the divergence of the amount of instances classified as boundaries in comparison to a given expectation E_y, which is derived from background knowledge. For example, in the domain of reference segmentation, it is expected that each reference contains exactly one title segment.

Using the flexible *VIKAMINE 2*-kernel architecture, supervised pattern mining could be easily integrated in the overall algorithm. The efficient implementations of *VIKAMINE 2* allowed to perform the numerous pattern mining tasks efficiently within the overall machine learning algorithm.

9.4 CaseTrain

CaseTrain is an online case-based training system for higher education [120], which is widely used at the University of Würzburg. On average, more than 800 training cases were solved by the students per day in the last years. To evaluate the benefits of the system for students, exceptional model mining was applied. Despite the high usage of the system, large scale statistical analysis of the usage was difficult since statistics from the training system cannot easily be linked to external student data, e.g., for privacy reasons. Therefore, the exceptional model mining focused on a single course with about two hundred users.

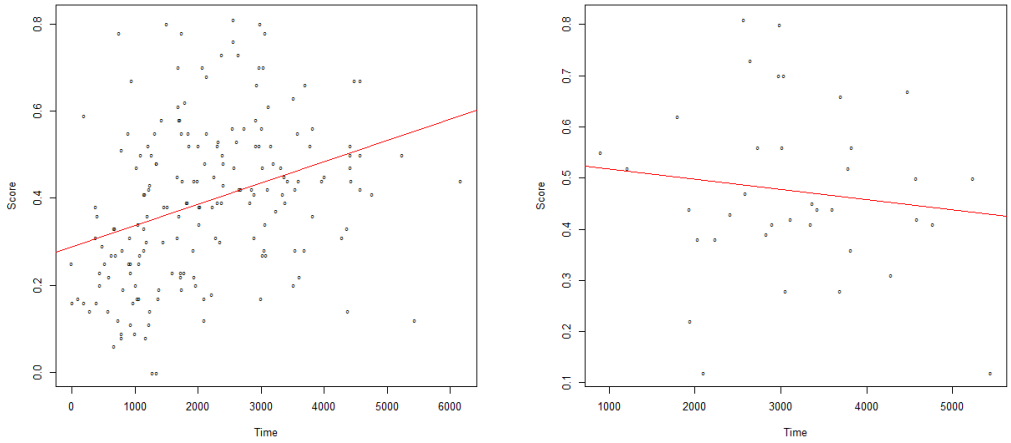
For each student, a variety of data was available, most importantly the time the students spent with the system and the score in the final exam of the course. Overall, there was a significant correlation between these two attributes with a Pearson coefficient of 0.336. That is, the more time the students spent with the system, the better the exam score for the course was. Even though this might be influenced by confounding factors, it indicates the usefulness of the system. The exceptional model mining now aimed at identifying influence factors, which alter this correlation between the time spent with the system and the final score. Therefore, the correlation between these attributes was used as the target model for the mining task. As interestingness measure, the function $q_{cor}(P) = {i_P}^a \cdot (\rho_P - \rho_\emptyset)$ was applied, where i_P is the number of instances in the subgroup and $\rho_P, \rho_\emptyset$ are the correlation coefficients between the target attributes in the subgroup and the total population respectively.

Two analyses were performed with different describing attributes. The first analysis was concerned with the question if certain usage patterns are indicators for a stronger or weaker correlation between the overall time spent and the final grade. In that direction, the describing attributes contained information, at which time (relative to the exam) the system was used, how many training cases were finished successfully or canceled, and what the average achieved score in the training cases was. Here, one result was that the subgroup of students, who use the system throughout the semester and not only directly before the exam, had a higher correlation between training time and score in the final exam (54 students with a correlation coefficient of 0.480).

In the second analysis, it was investigated if there are certain training cases in the system that are especially favorable or unfavorable. For that purpose, additional describing attributes indicated for each training case, how often a student worked with it. One interesting finding was concerned with a certain training case that was marked as a task from a previous exam. The 37 students, which executed this case at least three times, had a weak grade with respect to the time they worked with the system: The correlation coefficient between the two target attribute for these students decreased substantially to -0.127, see also Figure 9.5. However, also ignoring tasks from previous exams altogether seemed not to be a good strategy for students: For the 47 students, who did not work at least once with the respective task, the correlation coefficient also decreased to 0.017. Thus, it could be concluded that it is most beneficial for students to include this task into their exam preparation, but not to focus too much on it. Other

subgroups with larger conjunctive descriptions had larger deviations in the correlation coefficient, but were also more difficult to interpret.

Overall, exceptional model mining was able to extract interesting, interpretable findings. Even though the dataset was comparatively small and subgroup mining was limited to short subgroup descriptions, identifying these patterns would have been tedious without the proper tool support of *VIKAMINE 2*. The findings could also be visualized comfortably using the EDAT-Plugin, Section 8.8.



(a) Overall Population

(b) Subgroup

Figure 9.5: Comparison of correlation coefficients between the time spent with the *CaseTrain* system and the score in the final exam of the respective course. On the left-hand side the scatterplot with the linear regression line is drawn for the overall population, on the right-hand it is shown for the subgroup of students, who worked at least three times with a certain training case from previous exams.

9.5 Industrial Application

In another project, subgroup discovery was applied on a dataset from the manufacturing domain. Since the investigated production process is complex, the resulting products are sometimes of insufficient quality or in need of an additional repair step. To identify influence factors that increase or decrease the rate of these faulty parts, subgroup discovery was applied. Describing information included the time of production (time of the day and day of the week), environmental variables (e.g., temperature), the group of workers and used materials. Since the mining was performed interactively together with domain experts, tool support and efficient algorithms were required. Using *VIKAMINE 2* and the *SD-Map** algorithm, see Section 4.4.1, for this task enabled the close involvement of the domain experts in the discovery process. As one interesting result, the storage tanks

for certain chemicals played an important role in the production process. Unfortunately, this work cannot provide more details on this project due to privacy issues.

9.6 I-Pat Challenge

The I-Pat challenge was a local pattern mining competition organized for the workshop "Mining and exploiting interpretable local patterns" at the ECML/PKDD 2012. For the challenge, a dataset on *gene expression analysis* was provided. The goal of this research area is to describe sets of genes, which are associated with a certain disease. The dataset consisted of 6172 genes (instances), which were described by more than 9000 binary features from a gene ontology. The binary target label indicated if the gene was statistically associated with a certain protein for tumor suppression, cf. also [236]. A submission to the challenge consisted of a set of three subgroups that should contain interesting dependencies to this target concept. The subgroups were assessed and scored by domain experts with to respect novelty, usability and generality.

To evaluate the usefulness of some improvements presented in this work, the author participated in the challenge. As a strategic decision, it was decided to restrict subgroup descriptions to selection expressions, which indicate the presence of a gene ontology term (feature is set to true), and ignore the absence of features. For the selection of subgroups it was focused on generalization-aware measures based on the minimum improvement and expectation-driven subgroup discovery with the leaky-noisy-or model, cf. Chapter 7. The efficient algorithms implemented in *VIKAMINE 2* allowed to quickly obtain and inspect the results of different parameterizations. From the result subgroups, the final decision was made manually, picking subgroups that achieved good scores according to both, the minimum improvement and the leaky-noisy-or model. It was decided to only submit subgroups descriptions that combine at most two gene terms, since more complex patterns were assumed to be more complicated to comprehend.

This submission received the best overall average score from the domain experts, beating 6 other submissions of subgroup patterns to win the I-Pat 2012 challenge. Two of the three submitted subgroup received the maximum score. An interesting general feedback of domain experts was that many submitted subgroups were too general, contradicting a widely applied intuition for the selection of subgroups.

9.7 Discussion

Several case studies were predominantly performed for pragmatic problem solving in practical applications. However, also from a research point of view they offer benefits in three different directions:

First, the documented experiences provide guidance for future practitioners. In this direction, detailed descriptions of the respective mining scenarios and applied settings for automatic subgroup discovery steps, i.e., the exact interestingness measures, have been presented. As one overall result, generalization-aware or expectation-driven interestingness measures were considered in most cases as superior to traditional measures.

Second, the case studies were useful to evaluate the implemented techniques. Here, the flexible *VIKAMINE 2* software environment showed to be an effective tool for successful subgroup discovery: Required and helpful additions, such as novel interestingness measures or visualizations could be integrated quickly on demand. Although the sizes of the datasets were limited in most cases, the efficient implementations (including the novel algorithms presented in Chapters 4, 5, and 6) were often highly useful in reducing the waiting time of users since automatic discovery steps were often executed multiple times with slightly modified settings in an iterative approach. However, a large dataset that ultimately required efficient algorithms was only employed by a single case study, that is, by the mining of patterns in social media. (For the scalability of exceptional model mining in this dataset, see also Section 5.5.2.) Therefore, the performed case studies were not necessarily suited for an efficiency evaluation of the proposed algorithmic improvements.

Third, case studies can be used to get pointers for future research directions. In this context, it is notable that in all settings most of the discovered interesting subgroups had rather short descriptions, that is, they were described by a single selector or a conjunction of two selectors. This may partially be caused by the fact that in the investigated scenarios subgroup discovery was employed in a reporting-like scenario, in which the domain experts feedback could only be integrated in few, time-consuming process cycles. Nevertheless, this can be seen as a hint to move the research focus from more and more sophisticated algorithms that discover patterns with long descriptions efficiently towards efficient algorithms for the discovery of more interesting and less redundant patterns that potentially involve more complex interestingness measures. In this direction, the extension of expectation-driven interestingness measures to numeric target concepts and exceptional model mining seems beneficial for future applications. Another open issue in that regard is the selection of appropriate, non-redundant descriptions for numeric describing attributes.

9.8 Summary

This chapter reported on several real-world applications that successfully employed subgroup discovery in diverse domains: Two larger case studies were concerned with the investigation of factors that influence the success and satisfaction of university students and with the description of geographic locations by using tagging data from the social web. Further applications included the application of supervised pattern mining methods to improve conditional random fields for information extraction, the evaluation of the educational training system CaseTrain, a minor industrial application, and a submission to the IPAT pattern mining challenge. Finally, the benefits and limitations of the case studies were discussed.

10 Conclusions

This chapter first summarizes the main contributions of this work. Then, an outlook concludes the work with some pointers to promising future research directions.

10.1 Summary

Knowledge discovery is concerned with the extraction of novel, statistically valid, interpretable, and ultimately useful patterns from large datasets [79]. A key technique for that purpose is *subgroup discovery*. Subgroup discovery finds describable subsets in the data that show an interesting statistical distribution with respect to a certain predefined target concept. Descriptions are usually defined as conjunctions of simple selection expressions over the dataset attributes. Although this problem setting is well-researched in its basic form, there are still various issues that limit its use in practical applications. To approach these issues, contributions in several directions have been presented in this work:

Since subgroup discovery suffers from the curse of dimensionality, the *efficiency* of algorithms is important. Fast algorithms do not only make subgroup discovery tractable for very large datasets, they also improve the user experience in interactive settings since users can try out many variations of mining tasks with little waiting time. Algorithmic advances for efficient mining can be categorized in three dimensions: First, the enumeration strategy defines, in which order candidate subgroups are explored. Second, the data structure determines how the data is stored to allow for the efficient evaluation of candidates. Third, the pruning strategy allows skipping parts of the search space without affecting the optimality of results, e.g., by using optimistic estimate bounds. These are upper bounds for the interestingness of all specializations of a subgroup, which are known to be highly useful for safe pruning of the search space.

Chapter 3 provided a detailed description of these algorithmic components. The categorization of algorithms in these dimensions allowed for a concise description of existing algorithms. The provided summary of algorithms from literature was not limited to those using the terminology of subgroup discovery, but also covered approaches from closely related fields.

Most efficient algorithms for subgroup discovery concentrate on the basic setting with a binary target concept and an interestingness measure, which is only based on the statistics of the current subgroup. In this work, efficiency improvements for various variants of this standard setting were developed:

In Chapter 4, *subgroup discovery with a numeric target concept* was investigated. In that direction, novel optimistic estimate bounds have been introduced for a large variety

of interestingness measures. Some of the bounds are expressed in closed forms based on few key statistics and can be employed in combination with FP-tree-based data structures. The computation of additional bounds requires a full pass over all subgroup instances for each evaluation, but results in tighter bounds. Regarding data structures, it was shown how two different advanced data structures could be adapted for this setting, that is, bitset-based data representations and FP-trees. The proposed improvements were incorporated in two novel algorithms, that is, the *SD-Map** algorithm and the *NumBSD* algorithm. Both outperform previous approaches by more than an order of magnitude.

Exceptional model mining is an extension of classical subgroup discovery that features a model over multiple attributes instead of a single attribute as target concept. Chapter 5 proposed a novel approach for fast exhaustive exceptional model mining: It adapts the FP-tree data structure for exceptional model mining in a generic way. This is accomplished by introducing the concept of valuation bases as an intermediate condensed data representation. Many, but not all model classes are suited for that approach. The respective model classes were characterized by a theorem that draws an analogy to data stream mining. Concrete implementations of the generic approach for several model classes could be derived and lead to runtime improvements of more than an order of magnitude in comparison to a naive exhaustive depth-first-search.

The interestingness measures for subgroup selection are traditionally based exclusively on the statistics of the evaluated subgroup. Recently, *subgroup discovery with generalization-aware interestingness measures* has received increased attention in the research community. These measures compare the share of the target concept in the subgroup not only with the target share in the overall dataset, but also with the target shares of all generalizations of the current subgroup. In doing so, redundant findings are avoided and more interesting subgroups are discovered. In order to improve algorithmic efficiency in this setting, Chapter 6 introduced a new method of deriving optimistic estimates for subgroup discovery with these interestingness measures. In contrast to previous approaches to derive optimistic estimates, the novel bounds are not only based on the anti-monotonicity of instances covered by a subgroup, but incorporate also the difference in coverage between the subgroup and its generalizations. The incorporation of the new optimistic estimates in state-of-the-art algorithms improved the runtime requirements by more than an order of magnitude in many cases.

The previously mentioned contributions focus on the efficiency of algorithms. That is, the proposed automatic discovery algorithms produce the same results as existing algorithms, but require substantially less time. Other contributions improve the *effectiveness* of subgroup discovery. That means that the produced results are closer to what is wanted in actual applications.

In that direction, Chapter 7 presented the novel concept of *expectation-driven subgroup discovery*. It introduced a novel family of interestingness measures in order to achieve more interesting subgroup discovery results. For that purpose, the expectations on the target share of a subgroup with a binary target concept are estimated using the statistics for the separate influence factors, which are combined to describe the subgroup. The difference between the expected and the actual target shares then determines the

interestingness of a subgroup. Since expectations are inherently subjective, there is not a single correct function to compute them. However, plausible values, which are in line with human reasoning, can be achieved by transferring techniques from the research on bayesian networks to the subgroup discovery setting. The generation of bayesian network fragments as local models allowed to apply well known methods of this research area, e.g., the leaky-nosiy-or model or the logit model. The novel approach resulted in qualitatively different subgroups than traditional approaches, unveiling previously undiscovered relations. The usefulness of this approach could be demonstrated in several experiments, including two real-world case studies.

Beside these more theoretical core contributions, also practical issues were approached: Subgroup discovery is not an isolated automatic process, but an iterative and interactive task, which requires a flexible, interactive mining environment. For that purpose, the *VIKAMINE 2 tool* was developed in the context of this work, which significantly improves and extends its predecessor, see Chapter 8. It provides a broad collection of state-of-the-art algorithms and interestingness measures for mining with binary as well as numeric target concepts, including novel methods proposed in this work. A new, appealing graphical user interface based on the Eclipse RCP framework enables easy extensibility. Additionally, it features novel methods for the interactive mining in large datasets, e.g., the dynamic subgroup tree, and the effective presentation of results, e.g., the pie circle visualization and the subgroup treetable. The EDAT plugin provides additional options for inspecting the statistical properties in the overall dataset as well as in specific subgroups by utilizing a bridge to the statistical language R.

The benefits of the novel techniques contributed to efficient and effective subgroup discovery in several *real world applications*, see Chapter 9. In the educational domain, subgroup discovery was successfully applied to discover parameters that affect the success rate of university students and to find influence factors on the students' satisfaction rating in a large survey dataset. In the application area of social web mining, subgroup discovery was used to describe certain geographic locations by interpreting tags for geo-referenced images crawled from the Flickr platform. Further application examples included pattern mining for improved information extraction with conditional random fields, an industrial application regarding the fault rate of products, and a challenge dataset regarding gene data analysis.

10.2 Outlook

This work presented several novel techniques that improve the efficiency and effectiveness of subgroup discovery. These contributions also encourage the exploration of ideas in future research:

Regarding efficiency, the exploitation of optimistic estimate bounds for fast exceptional model mining could lead to substantial performance gains. Especially the exploration of difference-based optimistic estimates could be interesting in this area. Since interestingness measures and model classes in this area are heterogeneous, the identification of categorizing properties would allow for more generic, widely applicable bounds. In sub-

group discovery with binary targets, tight optimistic estimates are still to be developed for expectation-driven interestingness measures. Here, also intelligent caching strategies could be beneficial to reduce the memory requirements of algorithms. In order to avoid the discovery of redundant subgroups, the topic of subgroup set mining will quite possibly receive increased attention in future research. In that area, the exploration of efficient mining algorithms has started only recently. Additionally, the distribution of subgroup discovery tasks to a network of computational nodes provides for interesting challenges, in particular the propagation of optimistic estimate bounds with limited communication costs.

Regarding the effectiveness of subgroup discovery, improving the interestingness measures for automatic discovery is still a challenge. In that direction, the extension of expectation-driven subgroup discovery to settings with numeric or exceptional model mining targets seems promising. In order to explore, what results human users really perceive as interesting or useful, larger quantitative studies of interestingness measure are desirable. Another open issue is concerned with the redundancy of discovered subgroups. In particular, avoiding redundant findings for subgroup descriptions that involve numeric attributes still is an open issue. While several recent approaches aim at reducing the redundancy of results, e.g., weighted covering, subgroup set mining, or generalization-aware interestingness measures, a thorough comparison of these approaches is still missing in literature. However, dealing with redundant discoveries is quite possibly not only an issue of algorithmic tuning, but also of human-computer interaction and interface design. For that purpose, further adjustments of subgroup discovery tools will be required.

Subgroup discovery has been established as an important technique for knowledge discovery in order to extract useful findings from large amounts of raw data. However, practical applications of subgroup discovery still involve critical and unexplored challenges. This encourages further scientific research in this field. There is still much to discover.

Bibliography

[1] Poll: Data Mining Methodology, website: http://www.kdnuggets.com/polls/2007/data_mining_methodology.htm, 2007.

[2] Tarek Abudawood and Peter Flach. First-Order Multi-class Subgroup Discovery. In *Proceedings of the 5th Starting AI Researchers' Symposium at STAIRS*, 2010.

[3] Tarek Abudawood and Peter A. Flach. Evaluation Measures for Multi-class Subgroup Discovery. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2009.

[4] Rakesh Agrawal, Tomasz Imielienski, and Arun Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 1993.

[5] Rakesh Agrawal and Giuseppe Psaila. Active Data Mining. In *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining (KDD)*, 1995.

[6] Rakesh Agrawal and John C. Shafer. Parallel Mining of Association Rules. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):962–969, 1996.

[7] Rakesh Agrawal and Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, 1994.

[8] Shane Ahern, Mor Naaman, Rahul Nair, and Jeannie Yang. World Explorer: Visualizing Aggregate Data from Unstructured Text in Geo-Referenced Collections. In *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL)*, 2007.

[9] Jesus Alcala-Fdez, Alberto Fernandez, Julian Luengo, Joaquin Derrac, Salvador Garcia, Luciano Sanchez, and Francisco Herrera. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17(2–3):255–287, 2011.

[10] Grigoris Antoniou and Frank Van Harmelen. *A Semantic Web Primer*. MIT Press, 2004.

[11] John M. Aronis, Foster J. Provost, and Bruce G. Buchanan. Exploiting Background Knowledge in Automated Discovery. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, 1996.

[12] Bavani Arunasalam and Sanjay Chawla. CCCS: A Top-down Associative Classifier for Imbalanced Class Distribution. In *Proceedings of the 12th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2006.

[13] A. Asuncion and D. J. Newman. UCI Machine Learning Repository: http://www.ics.uci.edu/∼mlearn/MLRepository.html, 2007.

[14] Martin Atzmueller. *Knowledge-Intensive Subgroup Mining - Techniques for Automatic and Interactive Discovery*. PhD thesis, University of Würzburg, 2006.

[15] Martin Atzmueller and Florian Lemmerich. Fast Subgroup Discovery for Continuous Target Concepts. In *Proceedings of the 18th International Symposium on Foundations of Intelligent Systems (ISMIS)*, 2009.

[16] Martin Atzmueller and Florian Lemmerich. VIKAMINE - Open-Source Subgroup Discovery, Pattern Mining, and Analytics. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2012.

[17] Martin Atzmueller and Florian Lemmerich. Exploratory Pattern Mining on Social Media using Geo-References and Social Tagging Information. *International Journal of Web Science*, 2(1–2):80–112, 2013.

[18] Martin Atzmueller, Florian Lemmerich, Beate Krause, and Andreas Hotho. Towards Understanding Spammers - Discovering Local Patterns for Concept Description. In *From Local Patterns to Global Models, Workhop at the ECML/PKDD*, 2009.

[19] Martin Atzmueller and Folke Mitzlaff. Efficient Descriptive Community Mining. In *Proceedings of the 24th Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2011.

[20] Martin Atzmueller and Frank Puppe. Semi-Automatic Visual Subgroup Mining using VIKAMINE. *Journal of Universal Computer Science*, 11(11):1752–1765, 2005.

[21] Martin Atzmueller and Frank Puppe. A Methodological View on Knowledge-Intensive Subgroup Discovery. In *Proceedings of the 9th European Knowledge Acquisition Workshop (EKAW)*, 2006.

[22] Martin Atzmueller and Frank Puppe. SD-Map - A Fast Algorithm for Exhaustive Subgroup Discovery. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.

[23] Martin Atzmueller and Frank Puppe. Semi-Automatic Refinement and Assessment of Subgroup Patterns. In *Proceedings of the 21th International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2008.

[24] Martin Atzmueller and Frank Puppe. A Knowledge-Intensive Approach for Semi-Automatic Causal Subgroup Discovery. In Bettina Berendt et al., editors, *Knowledge Discovery Enhanced with Semantic and Social Information*, volume 220, pages 19–36. 2009.

[25] Martin Atzmueller, Frank Puppe, and Hans-Peter Buscher. Exploiting Background Knowledge for Knowledge-Intensive Subgroup Discovery. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

[26] Martin Atzmueller, Frank Puppe, and Hans-Peter Buscher. A Semi-Automatic Approach for Confounding-Aware Subgroup Discovery. *International Journal on Artificial Intelligence Tools*, 18(1):81–98, 2009.

[27] Yonatan Aumann and Yehuda Lindell. A Statistical Theory for Quantitative Association Rules. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1999.

[28] Yonatan Aumann and Yehuda Lindell. A Statistical Theory for Quantitative Association Rules. *Journal of Intelligent Information Systems*, 20(3):255–283, 2003.

[29] Iyad Batal and Milos Hauskrecht. A Concise Representation of Association Rules using Minimal Predictive Rules. In *Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2010.

[30] Iyad Batal and Milos Hauskrecht. Constructing Classication Features Using Minimal Predictive Patterns. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, 2010.

[31] Joachim Baumeister, Martin Atzmueller, and Frank Puppe. Inductive Learning for Case-Based Diagnosis with Multiple Faults. In *Proceedings of the 9th European Conference on Advances in Case-Based Reasoning*, 2002.

[32] Stephen D. Bay and Michael J. Pazzani. Detecting Change in Categorical Data: Mining Contrast Sets. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1999.

[33] Stephen D. Bay and Michael J. Pazzani. Detecting Group Differences: Mining Contrast Sets. *Data Mining and Knowledge Discovery*, 5(3):213–246, 2001.

[34] Roberto J. Bayardo. Efficiently Mining Long Patterns from Databases. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, 1998.

[35] Roberto J. Bayardo, Rakesh Agrawal, and Dimitrios Gunopulos. Constraint-Based Rule Mining in Large, Dense Databases. *Data Mining and Knowledge Discovery*, 4(2–3):217–240, 1999.

[36] Martin Becker. *Constraint Based Descriptive Pattern Mining.* Diploma Thesis, Supervised by the author of this work, University of Würzburg, 2011.

[37] Yoav Benjamini and Yosef Hochberg. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.

[38] Janine Bennett, Ray Grout, Philippe Pébay, Diana Roe, and David Thompson. Numerically Stable, Single-Pass, Parallel Statistics Algorithms. In *Proceedings of the IEEE International Conference on Cluster Computing and Workshops (CLUSTER)*, 2009.

[39] Francisco Berlanga, María José del Jesus, Pedro González, Francisco Herrera, and Mikel Mesonero. Multiobjective Evolutionary Induction of Subgroup Discovery Fuzzy Rules: A Case Study in Marketing. In *Proceedings of the 6th Industrial Conference on Data Mining (ICDM)*, 2006.

[40] Tijl De Bie, Kleanthis-Nikolaos Kontonasios, and Eirini Spyropoulou. A Framework for Mining Interesting Pattern Sets. In *Workshop on Useful Patterns, Workshop at the ACM SIGKDD*, 2010.

[41] Mario Boley, Thomas Gärtner, and Henrik Grosskreutz. Formal Concept Sampling for Counting and Threshold-Free Local Pattern Mining. In *Proceedings of the 10th SIAM International Conference on Data Mining (SDM)*, 2010.

[42] Mario Boley and Henrik Grosskreutz. Approximating the Number of Frequent Sets in Dense Data. *Knowledge and Information Systems*, 21(1):65–89, 2009.

[43] Mario Boley, Claudio Lucchese, Daniel Paurat, and Thomas Gärtner. Direct Local Pattern Sampling by Efficient Two-Step Random Procedures. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2011.

[44] George E. P. Box. Non-Normality and Tests on Variances. *Biometrika*, 40:318–335, 1953.

[45] Leo Breiman, Jerome H. Friedman, Charles J. Stone, and Richard A. Olshen. *Classification and Regression Trees.* Chapman & Hall, 1984.

[46] Sergey Brin, Rajeev Rastogi, and Kyuseok Shim. Mining Optimized Gain Rules for Numeric Attributes. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

[47] Björn Bringmann, Siegfried Nijssen, and Albrecht Zimmermann. Pattern-Based Classification: A Unifying Perspective, arXiv preprint arXiv:1111.6191, 2011.

[48] Björn Bringmann, Albrecht Zimmermann, Luc De Raedt, and Siegfried Nijssen. Don't Be Afraid of Simpler Patterns. In *Proceedings of the 10th European conference on Principle and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.

[49] Herve Broennimann, Bin Chen, Manoranjam Dash, Peter Haas, Yi Qiao, and Peter Scheuermann. Efficient Data-Reduction Methods for On-Line Association Rule Discovery. In *Data Mining: Next-Generation Challenges and Future Directions, Selected Papers from the NSF Workshop on Next-Generation Data Mining (NGDM)*, 2004.

[50] Facundo Bromberg, Brian Patterson, and Sandeep Yaramakala. Mining Bayesian Networks from Streamed Data. Technical report, Iowa State University, Ames, 2003.

[51] Gregory Buehrer, Srinivasan Parthasarathy, Shirish Tatikonda, Tahsin Kurc, and Joel Saltz. Toward Terabyte Pattern Mining: An Architecture-conscious Solution. In *Proceedings of the 12th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2007.

[52] Toon Calders and Bart Goethals. Mining All Non-Derivable Frequent Itemsets. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 2002.

[53] Toon Calders and Bart Goethals. Depth-First Non-Derivable ItemsetMining. In *Proceedings of the 5th SIAM International Conference on Data Mining (SDM)*, 2005.

[54] Toon Calders and Bart Goethals. Non-Derivable Itemset Mining. *Data Mining and Knowledge Discovery*, 14(1):171–206, 2007.

[55] José-Ramón Cano, Salvador García, and Francisco Herrera. Subgroup Discovery in Large Size Data Sets Preprocessed using Stratified Instance Selection for Increasing the Presence of Minority Classes. *Pattern Recognition Letters*, 29:2156–2164, 2008.

[56] José-Ramón Cano, Francisco Herrera, Manuel Lozano, and Salvador García. Making CN2-SD Subgroup Discovery Algorithm Scalable to Large Size Data Sets using Instance Selection. *Expert Systems with Applications*, 35(4):1949–1965, 2008.

[57] Loic Cerf, Dominique Gay, Nazha Selmaoui, and Jean-Francois Boulicaut. A Parameter-Free Associative Classification Method. In *Proceedings of the 10th International Conference on Data Warehousing and Knowledge Discovery (DaWaK)*, 2008.

[58] Bin Chen, Peter Haas, and Peter Scheuermann. A New Two-Phase Sampling Based Algorithm for Discovering Association Rules. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[59] Hong Cheng, Xifeng Yan, Jiawei Han, and Chih-Wei Hsu. Discriminative Frequent Pattern Analysis for Effective Classification. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE)*, 2007.

[60] Hong Cheng, Xifeng Yan, Jiawei Han, and Philip S. Yu. Direct Discriminative Pattern Mining for Effective Classification. In *Proceedings of the 24th International Conference on Data Engineering (ICDE)*, 2008.

[61] David W. Cheung, Jiawei Han, Vincent T. Ng, Ada W. Fu, and Yongjian Fu. A Fast Distributed Algorithm for Mining Association Rules. In *Proceedings of the 4th International Conference on Parallel and Distributed Information Systems*, 1996.

[62] Peter Clark and Robin Boswell. Rule Induction with CN2: Some Recent Improvements. In *Proceedings of the European Working Session on Machine Learning (EWSL)*, 1991.

[63] Peter Clark and Tim Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3(4):261–283, 1989.

[64] William W. Cohen. Fast Effective Rule Induction. In *Proceedings of the 11th International Conference on Machine Learning (ICML)*, 1995.

[65] R. J. Dakin. A Tree-search Algorithm for Mixed Integer Programming Problems. *The Computer Journal*, 8(3):250–255, 1965.

[66] Olena Daly and David Taniar. Exception Rules in Data Mining. In Mehdi Khosrow-Pour, editor, *Encyclopedia of Information Science and Technology (II)*, pages 1144–1148. 2005.

[67] Luc De Raedt, Tias Guns, and Siegfried Nijssen. Constraint Programming for Itemset Mining. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2008.

[68] María José del Jesus and Pedro González. Evolutionary Fuzzy Rule Induction Process for Subgroup Discovery: A Case Study in Marketing. *IEEE Transactions on Fuzzy Systems*, 15(4):578–592, 2007.

[69] Francisco J. Diez and Marek J. Druzdzel. Canonical Probabilistic Models for Knowledge Engineering. Technical report, Universidad Nacional de Educacion a Distancia Madrid, 2006.

[70] Guozhu Dong and Jinyan Li. Efficient Mining of Emerging Patterns: Discovering Trends and Differences. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1999.

[71] Wouter Duivesteijn, Ad Feelders, and Arno J. Knobbe. Different Slopes for Different Folks - Mining for Exceptional Regression Models with Cook's Distance. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2012.

[72] Wouter Duivesteijn and Arno J. Knobbe. Exploiting False Discoveries – Statistical Validation of Patterns and Quality Measures in Subgroup Discovery. In *Proceedings of the 11th International Conference on Data Mining (ICDM)*, 2011.

[73] Wouter Duivesteijn, Arno J. Knobbe, Ad Feelders, and Matthijs van Leeuwen. Subgroup Discovery Meets Bayesian Networks – An Exceptional Model Mining approach. In *Proceedings of the 10th International Conference on Data Mining (ICDM)*, 2010.

[74] Olive J. Dunn. Multiple Comparisons Among Means. *Journal of the American Statistical Association*, 56(293):52–64, 1961.

[75] Vladimir Dzyuba and Matthijs van Leeuwen. Interactive Discovery of Interesting Subgroup Sets. *Proceedings of the 12th International Symposium on Advances in Intelligent Data Analysis (IDA)*, 2013.

[76] Eyas El-Qawasmeh. Beating the Popcount. *International Journal of Information Technology*, 9(1):1–18, 2003.

[77] Tom Fawcett. An Introduction to ROC Analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[78] Usama M. Fayyad and Keki B. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, 1993.

[79] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Advances in Knowledge Discovery and Data Mining: An Overview. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–34. 1996.

[80] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From Data Mining to Knowledge Discovery in Databases. *AI magazine*, 17(3):37–54, 1996.

[81] Alex A. Freitas. On Rule Interestingness Measures. *Knowledge-Based Systems*, 12:309–315, 1999.

[82] Jerome H. Friedman and Nicholas I. Fisher. Bump Hunting in High-dimensional Data. *Statistics and Computing*, 9(2):123–143, 1999.

[83] Marlene Fries. Welchen Wert hat das Abitur für ein erfolgreiches Studium? *Beiträge zur Hochschulforschung*, 1:30–51, 2002.

[84] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Mining Optimized Association Rules for Numeric Attributes. In *Proceedings of the 15th ACM Symposium on Principles of Database Systems (PODS)*, 1996.

[85] Johannes Fürnkranz and Peter A. Flach. ROC 'n' rule Rule Learning - Towards a Better Understanding of Covering Algorithms. *Machine Learning*, 58(1):39–7, 2005.

[86] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of Rule Learning*. Springer, 2012.

[87] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. Rule Learning in a Nutshell. In Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač, editors, *Foundations of Rule Learning*, pages 19–55. 2012.

[88] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. Supervised Descriptive Rule Learning. In Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač, editors, *Foundations of Rule Learning*, pages 247–265. 2012.

[89] Johannes Fürnkranz and Georg Widmer. Incremental Reduced Error Pruning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 1994.

[90] Brian R. Gaines. Knowledge Acquisition: Past, Present and Future. *International Journal of Human-Computer Studies*, 71(2):135–156, 2013.

[91] Dragan Gamberger and Nada Lavrač. Expert-Guided Subgroup Discovery: Methodology and Application. *Journal of Artificial Intelligence Research*, 17:501–527, 2002.

[92] Dragan Gamberger, Nada Lavrač, and Goran Krstačić. Active Subgroup Mining: a Case Study in Coronary Heart Disease Risk Group Detection. *Artificial Intelligence in Medicine*, 28(1):27–57, 2003.

[93] Dragan Gamberger, Nada Lavrač, and Dietrich Wettschereck. Subgroup Visualization: A Method and Application in Population Screening. In *Proceedings of the 7th International Workshop on Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP-02)*, 2002.

[94] Salvador García, Julian Luengo, Jose A. Saez, Victoria Lopez, and Francisco Herrera. A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):734–750, 2013.

[95] Gemma C. Garriga, Petra Kralj Novak, and Nada Lavrač. Closed Sets for Labeled Data. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.

[96] Gemma C. Garriga, Petra Kralj Novak, and Nada Lavrač. Closed Sets for Labeled Data. *Journal of Machine Learning Research (extended version of the identically titled conference paper)*, 9:559–580, 2008.

[97] Liqiang Geng and Howard J. Hamilton. Interestingness Measures for Data Mining: A Survey. *ACM Computing Surveys*, 38(3): Article no. 9, 2006.

[98] Aristides Gionis, Heikki Mannila, Taneli Mielikäinen, and Panayiotis Tsaparas. Assessing Data Mining Results via Swap Randomization. *ACM Transactions on Knowledge Discovery from Data*, 1(3): Article number 14, 2007.

[99] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, 1989.

[100] Henrik Grosskreutz. Cascaded Subgroups Discovery with an Application to Regression. In *From Local Patterns to Global Models, Workshop at the ECML/PKDD*, 2008.

[101] Henrik Grosskreutz. Class Relevant Pattern Mining in Output-Polynomial Time. In *Proceedings of the 12th SIAM International Conference on Data Mining (SDM)*, 2012.

[102] Henrik Grosskreutz, Mario Boley, and Maike Krause-Traudes. Subgroup Discovery for Election Analysis: A Case Study in Descriptive Data Mining. In *Proceedings of the 13th International Conference on Discovery Science (DS)*, 2010.

[103] Henrik Grosskreutz, Benedikt Lemmen, and Stefan Rüping. Secure Distributed Subgroup Discovery in Horizontally Partitioned Data. *Transactions on Data Privacy*, 4(3):147–165, 2011.

[104] Henrik Grosskreutz and Daniel Paurat. Fast and Memory-Efficient Discovery of the Top-k Relevant Subgroups in a Reduced Candidate Space. In *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2011.

[105] Henrik Grosskreutz, Daniel Paurat, and Stefan Rüping. An Enhanced Relevance Criterion For More Concise Supervised Pattern Discovery. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2012.

[106] Henrik Grosskreutz and Stefan Rüping. On Subgroup Discovery in Numerical Domains. *Data Mining and Knowledge Discovery*, 19(2):210–226, 2009.

[107] Henrik Grosskreutz, Stefan Rüping, Nuhad Shaabani, and Stefan Wrobel. Optimistic Estimate Pruning Strategies for Fast Exhaustive Subgroup Discovery. Technical report, Fraunhofer IAIS, 2008.

[108] Henrik Grosskreutz, Stefan Rüping, and Stefan Wrobel. Tight Optimistic Estimates for Fast Subgroup Discovery. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2008.

[109] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. Frequent Pattern Mining: Current Status and Future Directions. *Data Mining and Knowledge Discovery*, 15(1):55–86, 2007.

[110] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2006.

[111] Jiawei Han, Jian Pei, and Yiwen Yin. Mining Frequent Patterns without Candidate Generation. *ACM SIGMOD Record*, 29(2):1–12, 2000.

[112] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004.

[113] Anna Hart. Knowledge Acquisition for Expert Systems. Technical report, School of Computing, Lancashire Polytechnic, Preston, 1986.

[114] Frederick Hayes-Roth, Donald Waterman, and Douglas Lenat. *Building Expert Systems*. Addison Wesley, 1984.

[115] David Heckerman and John S Breese. Causal Independence for Probability Assessment and Inference using Bayesian Networks. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 26(6):826–831, 1996.

[116] Max Henrion. Practical Issues in Constructing a Bayes' Belief Network. In *Proceedings of the 3rd Annual Conference on Uncertainty in Artificial Intelligence*, 1987.

[117] Francisco Herrera, Cristóbal José Carmona, Pedro González, and María José del Jesus. An Overview on Subgroup Discovery: Foundations and Applications. *Knowledge and Information Systems*, 29(3):495–525, 2010.

[118] David C. Hoaglin, Frederick Mosteller, and John Wilder Tukey. *Understanding Robust and Exploratory Data Analysis*. Wiley New York, 1983.

[119] Yosef Hochberg and Ajit C. Tamhane. *Multiple Comparison Procedures*. Wiley New York, 1987.

[120] Alexander Hoernlein, Marianus Ifland, Peter Kluegl, and Frank Puppe. Konzeption und Evaluation eines fallbasierten Trainingssystems im universitätsweiten Einsatz (CaseTrain). *GMS Medizinische Informatik Biometrie Epidemiologie 2009*, 5(1):Doc07, 2009.

[121] Lawrence B. Holder, Diane J. Cook, and Surnjani Djoko. Substructure Discovery in the SUBDUE System. In *Proceedings of the Workshop on Knowledge Discovery in Databases*, 1994.

[122] Sture Holm. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

[123] Jian Hu and Xiang Yang-Li. A Fast Parallel Association Rules Mining Algorithm Based on FP-Forest. In *Proceedings of the 5th International Symposium on Neural Networks (ISNN)*, 2008.

[124] Farhad Hussain, Huan Liu, Einoshin Suzuki, and Hongjun Lu. Exception Rule Mining with a Relative Interestingness Measure. In *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2000.

[125] Frederik Janssen and Johannes Fürnkranz. On the Quest for Optimal Rule Learning Heuristics. 78(3):343–379, 2010.

[126] Szymon Jaroszewicz and Tobias Scheffer. Fast Discovery of Unexpected Patterns in Data, Relative to a Bayesian Network. In *Proceeding of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2005.

[127] Szymon Jaroszewicz, Tobias Scheffer, and Dan A. Simovici. Scalable Pattern Mining with Bayesian Networks as Background Knowledge. *Data Mining and Knowledge Discovery*, 18(1):56–100, 2008.

[128] David D. Jensen. Knowledge Evaluation: Statistical Evaluations. In Willi Klösgen and Jan M. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery*, pages 475–489. 2002.

[129] Kenneth Jensen. Crisp-DM process. website: http://en.wikipedia.org/wiki/File:CRISP-DM_Process, 2012.

[130] Alipio M. Jorge, Paulo J. Azevedo, and Fernando Pereira. Distribution Rules with Numeric Attributes of Interest. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.

[131] Viktor Jovanoski and Nada Lavrač. Classification Rule Learning with APRIORI-C. In *Proceedings of the 10th Portuguese Conference on Progress in Artificial Intelligence, Knowledge Extraction, Multi-agent Systems, Logic Programming and Constraint Solving (EPIA)*, 2001.

[132] Andreas M. Kaplan and Michael Haenlein. Users of the World, Unite! The Challenges and Opportunities of Social Media. *Business Horizons*, 53(1):59–68, 2010.

[133] Kenneth A. Kaufman and Ryszard S. Michalski. Learning From Inconsistent and Noisy Data: The AQ18 Approach. In *Proceedings of the 11th Symposium ISMIS*, 1999.

[134] Branko Kavšek and Nada Lavrač. Analysis of Example Weighting in Subgroup Discovery by Comparison of Three Algorithms on a Real-life Data Set. In *Advances in Inductive Rule Learning, Workshop at the ECML/PKDD*, 2004.

[135] Branko Kavšek and Nada Lavrač. Apriori-SD: Adapting Association Rule Learning To Subgroup Discovery. *Applied Artificial Intelligence*, 20:543–583, 2006.

[136] Randy Kerber. Chimerge: Discretization of Numeric Attributes. In *Proceedings of the 10th National Conference on Artificial Intelligence*, 1992.

[137] Mika Klemettinen, Heikki Mannila, Pirjo Ronkainen, Hannu Toivonen, and A. Inkeri Verkamo. Finding Interesting Rules from Large Sets of Discovered Association Rules. In *Proceedings of the 3rd International Conference on Information and Knowledge Management*, 1994.

[138] Willi Klösgen. Exploration of Simulation Experiments by Discovery. Technical Report WS-04-03, 1994.

[139] Willi Klösgen. Efficient Discovery of Interesting Statements in Databases. *Journal of Intelligent Information Systems*, 4(1):53–69, 1995.

[140] Willi Klösgen. Explora: A Multipattern and Multistrategy Discovery Assistant. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 249–271. 1996.

[141] Willi Klösgen. Data Mining Tasks and Methods: Subgroup Discovery: Deviation Analysis. In Willi Klösgen and Jan M. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery*, pages 354–361. 2002.

[142] Willi Klösgen. Domain Knowledge to Support the Discovery Process: Taxonomies. In Willi Klösgen and Jan M. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery*, pages 451–456. 2002.

[143] Willi Klösgen and Michael May. Census Data Mining - An Application. In *Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2002.

[144] Willi Klösgen and Michael May. Spatial Subgroup Mining Integrated in an Object-Relational Spatial Database. In *Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2002.

[145] Peter Kluegl, Martin Toepfer, Florian Lemmerich, Andreas Hotho, and Frank Puppe. Collective Information Extraction with Context-specific Consistencies. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2012.

[146] Peter Kluegl, Martin Toepfer, Florian Lemmerich, Andreas Hotho, and Frank Puppe. Stacked Conditional Random Fields Exploiting Structural Consistencies. In *Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2012.

[147] Peter Kluegl, Martin Toepfer, Florian Lemmerich, Andreas Hotho, and Frank Puppe. Exploiting Structural Consistencies with Stacked Conditional Random Fields. *Mathematical Methodologies in Pattern Recognition and Machine Learning*, 30:111–125, 2013.

[148] Arno J. Knobbe, Bruno Crémilleux, Johannes Fürnkranz, and Martin Scholz. From Local Patterns to Global Models: The LeGo Approach to Data Mining. In *From Local Patterns to Global Models, Workshop at the ECML/PKDD*, 2008.

[149] Arno J. Knobbe, Marc de Haas, and Arno Siebes. Propositionalisation and Aggregates. In *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 2001.

[150] Arno J. Knobbe and Eric K. Y. Ho. Pattern Teams. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.

[151] Arno J. Knobbe and Eric K. Y. Ho. Pattern Teams (long version), available at http://www.kiminkii.com/publications.html. Technical report, 2006.

[152] Ron Kohavi. The Power of Decision Tables. In *Proceedings of the 8th European Conference on Machine Learning (ECML)*, 1995.

[153] Kleanthis-Nikolaos Kontonasios, Eirini Spyropoulou, and Tijl De Bie. Knowledge Discovery Interestingness Measures Based on Unexpectedness. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(5):386–399, 2012.

[154] Richard E. Korf. Depth-First Iterative-Deepening: An Optimal Admissible Tree Search. *Artificial Intelligence*, 27(1):97–109, 1985.

[155] Christian Körner, Dominik Benz, Andreas Hotho, Markus Strohmaier, and Gerd Stumme. Stop Thinking, Start Tagging: Tag Semantics Emerge from Collaborative Verbosity. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, 2010.

[156] Walter A. Kosters and Wim Pijls. APRIORI, A Depth First Implementation. In *Frequent Itemset Mining Implementations, Workshop at the CEUR-WS*, 2003.

[157] Sotiris Kotsiantis and Dimitris Kanellopoulos. Discretization Techniques: A Recent Survey. *GESTS International Transactions on Computer Science and Engineering*, 32(1):47–58, 2006.

[158] Petra Kralj Novak, Nada Lavrač, and Geoffrey I. Webb. Supervised Descriptive Rule Discovery: A Unifying Survey of Contrast Set, Emerging Pattern and Subgroup Mining. *Journal of Machine Learning Research*, 10:377–403, 2009.

[159] Stefan Kramer, Nada Lavrač, and Peter A. Flach. Propositionalization Approaches to Relational Data Mining. In Sašo Džeroski and Nada Lavrač, editors, *Relational Data Mining*, pages 262–286. 2001.

[160] Beate Krause, Christoph Schmitz, Andreas Hotho, and Gerd Stumme. The Anti-Social Tagger - Detecting Spam in Social Bookmarking Systems. In *4th International Workshop on Adversarial Information Retrieval on the Web, Workshop at the AICPS*, 2008.

[161] Mark-André Krogel. *On Propositionalization for Knowledge Discovery in Relational Databases*. PhD thesis, Otto-von-Guericke-Universitaet Magdeburg, 2005.

[162] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. *Introduction to Parallel Computing*. Benjamin/Cummings Publishing, 1994.

[163] John Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, 2001.

[164] Ailsa H. Land and Alison G. Doig. An Automatic Method of Solving Discrete Programming Problems. *Econometrica: Journal of the Econometric Society*, 28(3):497–520, 1960.

[165] Nada Lavrač, Peter A. Flach, Branko Kavšek, and Ljupco Todorovski. Adapting Classification Rule Induction to Subgroup Discovery. In *Proceedings of the 2002 International Conference on Data Mining (ICDM)*, 2002.

[166] Nada Lavrač and Dragan Gamberger. Relevancy in Constraint-Based Subgroup Discovery. In *Revised Selected Papers of the European Workshop on Inductive Databases and Constraint Based Mining*, 2006.

[167] Nada Lavrač, Branko Kavšek, Peter A. Flach, and Ljupco Todorovski. Subgroup Discovery with CN2-SD. *Journal of Machine Learning Research*, 5:153–188, 2004.

[168] Nada Lavrač, Anže Vavpetič, Larisa Soldatova, Igor Trajkovski, and Petra Kralj Novak. Using Ontologies in Semantic Data Mining with SEGS and g-SEGS. In *Proceedings of the 14th International Conference on Discovery Science (DS)*, 2011.

[169] Nada Lavrač, Filip Železný, and Peter A. Flach. RSD: Relational Subgroup Discovery through First-Order Feature Construction. In *Proceedings of the 12th International Conference on Inductive Logic Programming (ILP)*, 2003.

[170] Dennis Leman, Ad Feelders, and Arno J. Knobbe. Exceptional Model Mining. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2008.

[171] Florian Lemmerich and Martin Atzmueller. Incorporating Exceptions: Efficient Mining of ϵ-Relevant Subgroup Patterns. In *From Local Patterns to Global Models, Workshop at the ECML/PKDD*, 2009.

[172] Florian Lemmerich and Martin Atzmueller. Describing Locations using Tags and Images: Explorative Pattern Mining in Social Media. In *Revised selected papers from the Workshops on Modeling and Mining Ubiquitous Social Media*, 2012.

[173] Florian Lemmerich, Martin Becker, and Martin Atzmueller. Generic Pattern Trees for Exhaustive Exceptional Model Mining. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2012.

[174] Florian Lemmerich, Martin Becker, and Frank Puppe. Difference-Based Estimates for Generalization-Aware Subgroup Discovery. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD), Best Paper Award*, 2013.

[175] Florian Lemmerich, Marianus Ifland, and Frank Puppe. Identifying Influence Factors on Students Success by Subgroup Discovery. In *Proceedings of the 4th International Conference on Educational Data Mining (EDM)*, 2011.

[176] Florian Lemmerich and Frank Puppe. Local Models for Expectation-Driven Subgroup Discovery. In *Proceedings of the 11th International Conference on Data Mining (ICDM)*, 2011.

[177] Florian Lemmerich, Mathias Rohlfs, and Martin Atzmueller. Fast Discovery of Relevant Subgroup Patterns. In *Proceedings of the 23rd Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2010.

[178] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *Internet Mathematics*, 6(1):29–123, 2009.

[179] W Li, Jiawei Han, and Jian Pei. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. In *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM)*, 2001.

[180] Bing Liu and Wynne Hsu. Domain Knowledge to Support the Discovery Process: User Preferences. In Willi Klösgen and Jan M. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery*, pages 467–474. 2002.

[181] Bing Liu, Wynne Hsu, Heng-Siew Han, and Yiyuan Xia. Mining Changes for Real-Life Applications. In *Proceedings of the Second International Conference on Data Warehousing and Knowledge Discovery (DaWak)*, 2000.

[182] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating Classification and Association Rule Mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD)*, 1998.

[183] Huan Liu and Rudy Setiono. Chi2: Feature Selection and Discretization of Numeric Attributes. In *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 1995.

[184] Joel P. Lucas, Alipio M. Jorge, Fernando Pereira, Ana M. Pernas, and Amauri A. Machado. A Tool for Interactive Subgroup Discovery using Distribution Rules. In

Proceedings of the Artificial Intelligence 13th Portuguese Conference on Progress in Artificial Intelligence (EPIA), 2007.

[185] Peter J. F. Lucas. Bayesian Network Modelling by Qualitative Patterns. In *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI)*, 2002.

[186] Peter J. F. Lucas. Bayesian Network Modelling through Qualitative Patterns. *Artificial Intelligence*, 163(2):233–263, 2005.

[187] Manuel Lutz. *Intelligent Exploratory Data Analysis*. Diploma Thesis, Supervised by the author of this work, University of Würzburg, 2011.

[188] John A. Major and John J. Mangano. Selecting Among Rules Induced From A Hurricane Database. *Journal of Intelligent Information Systems*, 4(1):39–52, 1995.

[189] Michael Mampaey, Siegfried Nijssen, Ad Feelders, and Arno J. Knobbe. Efficient Algorithms for Finding Richer Subgroup Descriptions in Numeric and Nominal Data. In *Proceedings of the 12th International Conference on Data Mining (ICDM)*, 2012.

[190] Ken McGarry. A Survey of Interestingness Measures for Knowledge Discovery. *The Knowledge Engineering Review*, 20(1):39–61, 2005.

[191] Roseanne McNamee. Confounding and Confounders. *Occupational and Environmental Medicine*, 60(3):227–234, 2003.

[192] Ryszard S. Michalski. On the Quasi-Minimal Solution of the General Covering Problem. In *Proceedings of the 5th International Symposium on Information Processing (FCIP)*, 1969.

[193] Ryszard S. Michalski, Igor Mozetic, Jiarong Hong, and Nada Lavrač. The Multipurpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains. In *Proceedings of the 5th National Conference on Artificial Intelligence*, 1986.

[194] Melanie Mitchell. Genetic Algorithms: An Overview. *Complexity*, 1(1):31–39, 1995.

[195] Katherine Moreland and Klaus Truemper. Discretization of Target Attributes for Subgroup Discovery. In *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, 2009.

[196] Shinichi Morishita and Jun Sese. Traversing Itemset Lattices with Statistical Metric Pruning. In *Proceedings of the 19th ACM Symposium on Principles of Database Systems (PODS)*, 2000.

[197] Hiroshi Motoda, editor. *Active Mining: New Directions of Data Mining*. IOs press, 2002.

[198] Marianne Mueller, Romer Rosales, Harald Steck, Sriram Krishnan, Bharat Rao, and Stefan Kramer. Subgroup Discovery for Test Selection: A Novel Approach and Its Application to Breast Cancer Diagnosis. In *Proceedings of the 8th International Symposium on Intelligent Data Analysis (IDA)*, 2009.

[199] Mark E. J. Newman and Michelle Girvan. Finding and Evaluating Community Structure in Networks. *Physical review E*, 69(2):Document Nr. 026113, 2004.

[200] Siegfried Nijssen, Tias Guns, and Luc De Raedt. Correlated Itemset Mining in ROC Space: A Constraint Programming Approach. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (KDD)*, 2009.

[201] Edward R. Omiecinski. Alternative Interest Measures for Mining Associations in Databases. *IEEE Transactions on Knowledge and Data Engineering*, 15(1):57–69, 2003.

[202] Agnieska Onisko, Marek J. Druzdzel, and Hanna Wasyluk. Learning Bayesian Network Parameters from Small Data Sets: Application of Noisy-OR Gates. *International Journal of Approximate Reasoning*, 27, 2001.

[203] Giulia Pagallo and David Haussler. Boolean Feature Discovery in Empirical Learning. *Machine Learning*, 5(1):71–99, 1990.

[204] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace Clustering for High Dimensional Data: A Review. *ACM SIGKDD Explorations Newsletter*, 6(1):90–105, 2004.

[205] Judea Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison Wesley, 1984.

[206] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[207] Judea Pearl. Why There is No Statistical Test For Confounding, Why Many Think There Is, and Why They Are Almost Right. In Judea Pearl, editor, *Causality – Models, Reasoning, and Inference*, pages 182–185. 2000.

[208] Gregory Piatetsky-Shapiro and William Frawley. *Knowledge Discovery in Databases*. AAAI / MIT press, 1991.

[209] Barbara F. I. Pieters. Subgroup Discovery on Numeric and Ordinal Targets, with an Application to Biological Data Aggregation. Technical report, Universiteit Utrecht, 2010.

[210] Barbara F. I. Pieters, Arno J. Knobbe, and Sašo Džeroski. Subgroup Discovery in Ranked Data, with an Application to Gene Set Enrichment. In *Preference Learning, Workshop at the ECML/PKDD*, 2010.

[211] Leonard Pitt and Robert E. Reinke. Criteria for Polynomial Time (Conceptual) Clustering. *Machine Learning*, 2(4):371–396, 1988.

[212] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013.

[213] Luc De Raedt and Albrecht Zimmermann. Constraint-Based Pattern Set Mining. In *Proceedings of the 13th SIAM International Conference on Data Mining (SDM)*, 2007.

[214] Rajeev Rastogi and Kyuseok Shim. Mining Optimized Association Rules with Categorical and Numeric Attributes. *IEEE Transactions on Knowledge and Data Engineering*, 14(1):29–50, 2002.

[215] Jochen Reutelshoefer, Joachim Baumeister, and Frank Puppe. A Meta-Engineering Approach for Customized Document-centered Knowledge Acquisition. In *Modellierung*, 2012.

[216] Jorma Rissanen. Modeling by Shortest Data Description. *Automatica*, 14(5):465–471, 1978.

[217] Mathias Rohlfs. *Techniken zur effizienten und verteilten Subgruppenentdeckung in großen Datenmengen*. Diploma Thesis, Supervised by the author of this work, University of Würzburg, 2009.

[218] Tobias Scheffer and Stefan Wrobel. Incremental Maximization of Non-Instance-Averaging Utility Functions with Applications to Knowledge Discovery Problems. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, 2001.

[219] Tobias Scheffer and Stefan Wrobel. A Scalable Constant-Memory Sampling Algorithm for Pattern Discovery in Large Databases. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, 2002.

[220] Tobias Scheffer and Stefan Wrobel. Finding the Most Interesting Patterns in a Database Quickly by Using Sequential Sampling. *Journal of Machine Learning Research*, 3:833–862, 2003.

[221] Martin Scholz. Knowledge-Based Sampling for Subgroup Discovery. In *Revised Selected Papers from the International Seminar on Local Pattern Detection, Dagstuhl Castle*, 2005.

[222] Martin Scholz. Sampling-Based Sequential Subgroup Mining. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD)*, 2005.

[223] Colin Shearer. The CRISP-DM Model: The New Blueprint for Data Mining. *Journal of Data Warehousing*, 5:13–22, 2000.

[224] Takahiko Shintani and Masaru Kitsuregawa. Hash Based Parallel Algorithms for Mining Association Rules. In *Proceedings of the 4th International Conference on Parallel and Distributed Information Systems*, 1996.

[225] Ben Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages (VL)*, 1996.

[226] R. Sibson. SLINK: an Optimally Efficient Algorithm for the Single-link Cluster Method. *The Computer Journal*, 16(1):30–34, 1973.

[227] Avi Silberschatz and Alexander Tuzhilin. On Subjective Measures of Interestingness in Knowledge Discovery. In *Proceedings of the1st International Conference on Knowledge Discovery and Data Mining (KDD)*, 1995.

[228] Avi Silberschatz and Alexander Tuzhilin. What Makes Patterns Interesting in Knowledge Discovery Systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970–974, 1996.

[229] Arnaud Soulet, Chedy Raïssi, Marc Plantevit, and Bruno Crémilleux. Mining Dominant Patterns in the Sky. In *Proceedings of the 11th International Conference on Data Mining (ICDM)*, 2011.

[230] Charles Sutton and Andrew McCallum. An Introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012.

[231] Einoshin Suzuki. Undirected Discovery of Interesting Exception Rules. *International Journal of Pattern Recognition and Artificial Intelligence*, 16(8):1065–1086, 2002.

[232] Einoshin Suzuki. Data Mining Methods for Discovering Interesting Exceptions from an Unsupervised Table. *Journal of Universal Computer Science*, 12(6):627–653, 2006.

[233] Einoshin Suzuki and Jan M. Zytkow. Unified Algorithm for Undirected Discovery of Exception Rules. *International Journal of Intelligent Systems*, 20(7):673–691, 2005.

[234] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the Right Interestingness Measure for Association Patterns. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[235] Nikolaj Tatti. Maximum Entropy based Significance of Itemsets. *Knowledge and Information Systems (KAIS)*, 17(1):57–77, 2008.

[236] Georgia Tsiliki, Sophia Kossida, Natalja Friesen, Stefan Rueping, Manolis Tzagarakis, and Nikos Karacapilidis. Data Mining Based Collaborative Analysis of

Microarray Data. In *Proceedings of the 24th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2012.

[237] Shusaku Tsumoto, Takahira Yamaguchi, Masayuki Numao, and Hiroshi Motoda. Active mining project: Overview. In *Revised Selected papers from the 2nd Workshop on Active Mining*, 2005.

[238] John Wilder Tukey. *Exploratory Data Analysis*. Addison Wesley, 1977.

[239] Charles R. Twardy and Kevin B. Korb. Causal Interaction in Bayesian Networks. Technical report, 2002.

[240] Takeaki Uno, Tatsuya Asai, Yuzo Uchida, and Hiroki Arimura. An Efficient Algorithm for Enumerating Closed Patterns in Transaction Databases. In *Proceedings of the 7th International Conference on Discovery Science (DS)*, 2004.

[241] Matthijs van Leeuwen. Maximal Exceptions with Minimal Descriptions. *Data Mining and Knowledge Discovery*, 21(2):259–276, 2010.

[242] Matthijs van Leeuwen and Arno J. Knobbe. Non-redundant Subgroup Discovery in Large and Complex Data. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2011.

[243] Matthijs van Leeuwen and Arno J. Knobbe. Diverse Subgroup Set Discovery. *Data Mining and Knowledge Discovery*, 25:208–242, 2012.

[244] Matthijs van Leeuwen and Antti Ukkonen. Discovering Skylines of Subgroup Sets. *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2013.

[245] Anže Vavpetič, Vid Podpečan, Stijn Meganck, and Nada Lavrač. Explaining Subgroups through Ontologies. In *Proceedings of the 12th Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, 2012.

[246] Jianyong Wang and George Karypis. In *Proceedings of the 5th SIAM International Conference on Data Mining (SDM)*, 2005.

[247] Ke Wang, Senqiang Zhou, Chee A. Fu, and Jeffrey X. Yu. Mining Changes of Classification by Correspondence Tracing. In *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM)*, 2003.

[248] Geoffrey I. Webb. OPUS: An Efficient Admissible Algorithm for Unordered Search. *Journal of Artificial Intelligence Research*, 3(1):431–465, 1995.

[249] Geoffrey I. Webb. Discovering Associations with Numeric Variables. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2001.

[250] Geoffrey I. Webb. Discovering Significant Rules. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2006.

[251] Geoffrey I. Webb. Discovering Significant Patterns. *Machine Learning*, 68(1):1–33, 2007.

[252] Geoffrey I. Webb. Layered Critical Values: a Powerful Direct-Adjustment Approach to Discovering Significant Patterns. *Machine Learning*, 71(2–3):307–323, 2008.

[253] Geoffrey I. Webb. Self-Sufficient Itemsets: An Approach to Screening Potentially Interesting Associations Between Items. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(1):Article no. 3, 2010.

[254] Geoffrey I. Webb. Filtered-top-k Association Discovery. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):183–192, 2011.

[255] Geoffrey I. Webb, Shane Butler, and Douglas Newlands. On Detecting Differences between Groups. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2003.

[256] Geoffrey I. Webb and Kotagiri Ramamohanarao. Inclusive Pruning: A New Class of Pruning Rule for Unordered Search and its Application to Classification Learning. In *Proceedings of the 19th Australian Computer Science Conference (ACSC)*, 1996.

[257] Geoffrey I. Webb and Songmao Zhang. Removing Trivial Associations in Association Rule Discovery. In *Proceedings of the 1st International NAISO Congress on Autonomous Intelligent Systems (ICAIS)*, 2002.

[258] Dietrich Wettschereck, Alipio M. Jorge, and Steve Moyle. Visualization and Evaluation Support of Knowledge Discovery through the Predictive Model Markup Language. In *Proceedings of the 7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES)*, 2003.

[259] B. W. Wisse, Sicco P. van Gosliga, Nicole P. van Elst, and Ana I. Barros. Relieving the Elicitation Burden of Bayesian Belief Networks. In *Bayesian Modelling Applications, Workshop at UAI*, 2008.

[260] Stefan Wrobel. An Algorithm for Multi-relational Discovery of Subgroups. In *Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD)*, 1997.

[261] Stefan Wrobel. Inductive Logic Programming for Knowledge Discovery in Databases. In Sašo Džeroski and Nada Lavrač, editors, *Relational Data Mining*, pages 74–101. 2001.

[262] Stefan Wrobel, Dietrich Wettschereck, Edgar Sommer, and Werner Emde. Extensibility in Data Mining Systems. Technical report, Arbeitspapiere-GMD, 1996.

[263] Tianyi Wu, Yuguo Chen, and Jiawei Han. Association Mining in Large Databases: A Re-Examination of Its Measures. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2007.

[264] Michael Wurst and Martin Scholz. Distributed Subgroup Mining. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.

[265] Hong Yao and Howard J. Hamilton. Mining Itemset Utilities from Transaction Databases. *Data & Knowledge Engineering*, 59(3):603–626, 2006.

[266] Mohammed J. Zaki. Scalable Algorithms for Association Mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3):372–390, 2000.

[267] Mohammed J. Zaki and Karam Gouda. Fast Vertical Mining Using Diffsets. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2003.

[268] Mohammed J. Zaki, M. Ogihara, Srinivasan Parthasarathy, and W. Li. Parallel Data Mining for Association Rules on Shared-Memory Multi-Processors. In *Proceedings of the 1996 ACM/IEEE Conference on Supercomputing*, 1996.

[269] Mohammed J. Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New Algorithms for Fast Discovery of Association Rules. Technical report, University of Rochester, 1997.

[270] Filip Zelezny and Nada Lavrač. Propositionalization-based Relational Subgroup Discovery with RSD. *Machine Learning*, 62(1-2):33–63, 2006.

[271] Albrecht Zimmermann, Björn Bringmann, Siegfried Nijssen, Nikolaj Tatti, and Jillen Vreeken. {Mining, Sets, of, Patterns } Part III, Slides of a Tutorial given at ICDM 2011, website: http://usefulpatterns.org/msop/slides/part3.pdf, 2011.

[272] Albrecht Zimmermann and Luc De Raedt. Cluster-Grouping: From Subgroup Discovery to Clustering. In *Proceedings of the 15th European Conference on Machine Learning (ECML)*, 2004.

[273] Albrecht Zimmermann and Luc De Raedt. Corclass: Correlated Association Rule Mining for Classification. In *Proceedings of the 7th International Conference on Discovery Science (DS)*, 2004.

[274] Albrecht Zimmermann and Luc De Raedt. Cluster-Grouping: From Subgroup Discovery to Clustering. *Machine Learning*, 77(1):125–159, 2009.