Generic Pattern Trees for Exhaustive Exceptional Model Mining

Florian Lemmerich[1], Martin Becker[1], and Martin Atzmueller[2]

[1] Artificial Intelligence and Applied Computer Science Group,
University of Würzburg, D-97074 Würzburg, Germany
{lemmerich, becker}@informatik.uni-wuerzburg.de
[2] Knowledge & Data Engineering Group,
University of Kassel, 34121 Kassel, Germany
atzmueller@cs.uni-kassel.de

Abstract. Exceptional model mining has been proposed as a variant of subgroup discovery especially focusing on complex target concepts. Currently, efficient mining algorithms are limited to heuristic (non exhaustive) methods. In this paper, we propose a novel approach for fast exhaustive exceptional model mining: We introduce the concept of valuation bases as an intermediate condensed data representation, and present the general GP-growth algorithm based on FP-growth. Furthermore, we discuss the scope of the proposed approach by drawing an analogy to data stream mining and provide examples for several different model classes. Runtime experiments show improvements of more than an order of magnitude in comparison to a naive exhaustive depth-first search.

Keywords: exceptional model mining, subgroup discovery

1 Introduction

Subgroup discovery [9,17] has been established as a general and broadly applicable technique for descriptive data mining: It aims at identifying descriptions of subsets of a dataset that show an interesting behavior with respect to a certain *target property of interest*. In this context, the concept of *exceptional model mining* has been introduced [6,13], which especially focuses on complex target properties: It tries to identify interesting patterns with respect to a local model derived from *a set* of attributes. The interestingness can be defined, e.g., by a significant deviation from a model that is derived from the total population or the respective complement set of instances within the population. While there exist heuristic algorithms [11] for exceptional model mining, the efficient exhaustive computation of exceptional models is still an open issue.

In this paper, we present the novel *GP-growth* algorithm that can be used for mining patterns with exceptional target models *exhaustively*. We extend the well-known FP-tree [7] data structure 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 an efficient computation of the target model parameters.

The contribution of this paper is threefold: First, we present the concept of valuation bases allowing us to derive a new algorithm capable of performing efficient exhaustive search for many different classes of exceptional models. Second, we characterize the scope of the presented approach and discuss its instantiations for model classes presented in literature. Third, we perform an evaluation of the presented approach using publicly available UCI data [14]. Furthermore, we present a scalability study on a real world dataset.

The remainder of the paper is structured as follows: Section 2 discusses related work. Section 3 introduces the formal background of exceptional model mining and briefly reviews the FP-growth algorithm. Next, Section 4 presents the novel GP-growth algorithm. Instantiations for different model classes are discussed in Section 5. After that, we demonstrate the effectiveness and validity of our approach using publicly available data in Section 6. Finally, the paper concludes with a summary and outlook on future work in Section 7.

2 Related Work

For a recent overview on algorithms for subgroup discovery [9,17], including heuristic and exhaustive approaches, we refer to [8]. Kralj et al. also discuss subgroup discovery in relation to other common approaches for supervised descriptive rule discovery [15]. Exceptional model mining has been proposed in [6,13]. [11] presents an heuristic algorithm for identifying descriptions of exceptional subgroups. Umek et al. describe subgroup discovery in a similar setting [16]. Again, their proposed algorithm does not perform an exhaustive search.

To the best of the authors' knowledge, no non-trivial exhaustive algorithm for the task of exceptional model mining has been published so far. To this end, we propose the GP-growth algorithm based on FP-growth and FP-trees. Originally developed for association rule mining [7], FP-trees are a widely-used data representation. It has been adapted to subgroup discovery with nominal [3] and numeric target concepts [1]. The transfer to exceptional model mining is not trivial but – as shown in this paper – possible for many model classes.

3 Background

In the following, we first provide a brief summary of exceptional model mining [6,13]. After that, we review the FP-growth algorithm.

3.1 Exceptional Model Mining

Exceptional model mining aims at identifying models (patterns) that are *exceptional* concerning a set of target attributes. For some basic definitions, a database $D = (I, A)$ is given by a set of data instances (cases) $i \in I$ and a

set of attributes A. The attributes consist of two (usually non-overlapping) sets of describing attributes A_D and model attributes A_M. We denote the value of attribute X in instance i as i_X. *Selectors* or *basic patterns* are boolean functions $I \rightarrow \{false, true\}$ defined by selection expressions on the set of describing attributes A_D. Typical selection expression are given by attribute-value pairs in the case of nominal attributes, or by intervals in the case of numeric attributes. The selector $age = [12; 18]$ is true, for example, iff the attribute *age* has a value between 12 and 18 for the respective instance. A *subgroup description* or (complex) *pattern* combines selectors into a boolean formula. For a typical conjunctive description language, a pattern $p = \{\text{sel}_1, \ldots, \text{sel}_k\}$ is defined by a set of selectors sel_j, which are interpreted as a conjunction, i.e. $p = \text{sel}_1 \wedge \ldots \wedge \text{sel}_k$. A *subgroup* corresponding to a pattern p contains all instances for which p evaluates to *true*.

A model consists of a specific *model class*, which is fixed for a specific mining task, and *model parameters* which depend on the values of the model attributes in the instances of the respective subgroup. The goal of exceptional model mining is then to identify descriptions of subgroups, for which the model parameters differ significantly from the parameters of the model built from the entire dataset. Formally this is accomplished by using an *exceptionality measure* q that maps a subgroup (pattern) to a real number corresponding to its quality (interestingness) based on its model parameters. As a simple example, consider the task of identifying subgroups in which the correlation between two numeric attributes is especially strong. This *correlation model class* has exactly one parameter, i.e., the correlation coefficient. A short overview on different model classes presented in literature is included in Section 5.

To accomplish the exceptional model mining task for a set of l selectors there are $O(2^l)$ subgroup descriptions, for which the model parameters need to be determined. For practical purposes it is often possible to limit the search space to patterns with a maximum number d of contained selectors ($|p| \leq d$) (since longer patterns are difficult to interpret by humans). However, identifying the best pattern is still challenging since the size of the search space is still $O(l^d)$. In this paper, we provide an efficient exhaustive algorithm for this task.

3.2 FP-growth

The FP-growth [7] algorithm has been introduced as an efficient approach for frequent pattern mining. It avoids scanning the whole dataset in order to evaluate each pattern by recursively building a special data structure, the so-called FP-tree. This extended prefix tree structure contains the relevant data in a compressed way. Each tree node contains a reference to a selector and a frequency count. Additionally, links between nodes referring to the same selector are maintained. The FP-tree is built by sorting the selectors of each data instance according to their descending frequency in the dataset. Then, each data instance is inserted into the FP-tree. The order of the selectors increases the chance of shared prefixes between data instances decreasing the size of the FP-tree. Most importantly, the resulting FP-tree contains the complete condensed frequency information for each data instance.

FP-growth starts with creating an FP-tree for the initial dataset. Patterns containing exactly one selector are evaluated by the frequencies collected during the first pass over the dataset. Then, the algorithm recursively extends those patterns by adding further selectors in a depth-first manner, building conditional trees conditioned on the current pattern prefix. Each node corresponds to a conditional data instance built from the selectors referred to by its parent nodes. In this way, FP-growth enables a compact and efficient mining of the condensed tree structure. Due to the limited space, we refer to [7] for more details.

4 The GP-growth Algorithm

In this section, we present our novel approach called *generic pattern growth (GP-growth)*. GP-growth is based on the FP-growth algorithm substituting frequencies by an intermediate, condensed data representation called a valuation basis. We first introduce the concept of valuation bases and define properties needed for their application to the GP-growth algorithm. After that, we provide a theorem concerning the existence and construction of efficient valuation bases for specific model classes.

4.1 The Concept of Valuation Bases

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

We define a valuation basis 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 definition is based on the model parameters, it can be derived from such a valuation basis.

A visualization of the overall approach is given in Figure 1: For each subgroup (set of data instances) a valuation basis can be derived using a function ϕ (*valuation projector*). The model parameters for the chosen model class can be 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.

As an example, consider a very basic model for a single model attribute X, for which the only model parameter is the mean value of 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

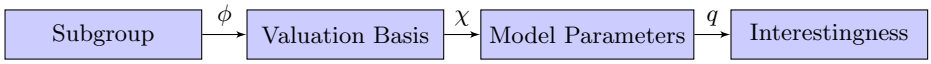

Fig. 1. The pipeline visualizing our approach: For each subgroup (set of data instances) s a valuation basis can be derived using a function ϕ (valuation projector). The model parameters for the chosen model class can be extracted from these valuation bases using another function χ (model extractor). Model parameters are then used to determine the interestingness of the respective pattern using an exceptionality measure q.

interestingness 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 we 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). 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 general applicable approach, the main advantages in terms of memory and runtime performance are usually lost. Therefore, we aim to construct valuation bases for a given model class which are as small as possible. We call a valuation basis a *condensed valuation basis*, if its memory requirement is sublinear with respect to the number of instances it represents. The examples of valuation bases that we present in this paper will all use constant memory with respect to the instance count.

We model the possibility of aggregating valuation bases corresponding to sets of data instances by introducing the notion of *valuation domains*:

Definition 1 (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 2.

Definition 2 (Valuation Projector). *Let 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*

$$\phi : I \to V$$

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

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

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

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

Sometimes patterns are evaluated by comparing their corresponding model attributes derived from the set of data instances they cover with the model attributes of their complementary set of data instances. In order to handle this case efficiently we need to assume a valuation domain with a subtraction operator $\ominus$: Let I be the set of all instances and let $I' \cup I''$ be an arbitrary partition of I. If the valuation basis of I' is subtracted from the valuation basis of I, then the result must be the valuation basis of I'': $\bar{\phi}(I) \ominus \bar{\phi}(I') = \bar{\phi}(I'')$. This can be utilized by computing the valuation basis corresponding to all instances in the dataset I in an initial pass over the dataset.

4.2 Algorithmic Adaptations

Essentially, the FP-growth algorithm (cf. Section 3.2 for a brief description) is generalized by substituting frequencies with the more general concept of valuation bases. We call the resulting algorithm *GP-growth*. The generalized tree structure storing 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 4.1.

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
- a subtraction operator $\ominus$ to support complement comparisons as mentioned in Section 4.1, if 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.*

Please note, that for the very simple valuation basis, that 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 [3]. Thus, the approach presented in this paper can be regarded as a true generalization of both, FP-growth [7] and SD-Map.

Like subgroup discovery, exceptional model mining is often applied in a top-k approach. That is, the goal is to find the best k patterns with respect to an interestingness measure. This can be accomplished by storing the current top k patterns in a separate result set and replacing its entries with higher quality patterns as required.

When using a top-k approach, substantial speed-ups can be achieved by using optimistic estimate pruning, see for example [17]. As an example, a quality function using a correlation coefficient model could exploit the fact, that the correlation coefficient never exceeds a value of 1. However, efficient boundaries need to be determined for each quality function, which may even vary for a single model class. In this paper, we focus on a generic data structure that allows for the efficient computation of model parameters. Therefore, we will not discuss possible optimistic estimate boundaries for model exceptionality measures in the context of this work.

4.3 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 done in [6].

Therefore, in the following we provide a characterization of model classes, for which GP-trees can be applied with strongly reduced memory requirements. We do this 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 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. We can then construct a parallel single-pass algorithm as follows: In each computation node N we loop through all assigned instances I_N, updating the respective valuation basis v_N. For each instance $i \in I_N$ we extract the valuation basis $\phi(i)$ and use it to update the current accumulated valuation basis: $v_N^{new} = v_N^{old} \oplus \phi(i)$. 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 can then be 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. We show, 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, we can use this vector as our valuation basis 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 corresponds to a valuation basis. v_j must be sufficient to extract the model parameters for the data D_j, since it could be that N_j is 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 we 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. Some important examples of this approach are shown in the next section.

5 Valuation Bases for Important Model Classes

In this section, we discuss the application of GP-trees to different model classes. Most of the presented model classes have been proposed in [13], to which we refer for a more detailed description of the models and exceptionality measures.

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, we utilize the following well known formula:

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

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:

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

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 i contains the constant 1 as the instance count, the value of X in i and the squared value of X in i. 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.

Correlation Model The (Pearson product-moment) correlation coefficient $\rho(X, Y)$ measure 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: $\rho(X, Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}$.

The covariance is defined as $Cov(X, Y) = \frac{\sum_{(x,y)}(x-\mu_X)(y-\mu_Y)}{N}$, where μ describes the mean value of the attribute and N the number of instances. We determine the measures $Cov(X, Y)$, σ_X and σ_Y independently from each other.

To efficiently compute the covariance of two variables X, Y we utilize the following pairwise update formula that was introduced in [4] for an parallel single-pass algorithm. It allows us to compute $C_S(X,Y) = Cov(X,Y) \cdot |S| = \sum_{(x,y)\in S}(x-\mu_X)(y-\mu_Y)$ 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:

$$C_S(X,Y) = C_{S_1}(X,Y) + C_{S_2}(X,Y) + \frac{n_1 n_2}{n_1+n_2}(\mu_{X,2}-\mu_{X,1})(\mu_{Y,2}-\mu_{Y,1}),$$

where, $n_1 = |S_1|$ and $n_2 = |S_2|$ 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_2 and S_1.

For computing the covariance for each pattern using this formula, we need to keep track of the value of C in the respective set of instances, the cardinality of the subgroup, and the mean values of the variables X and Y. The latter can be computed by the sum of the respective values and the cardinality. Therefore, using the formula above we can define a model configuration for the covariance as follows:

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

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 we combine this valuation basis 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:

$$\begin{aligned}
V_{cor} &= \mathbb{R}^6 \\
v \oplus_{cor} u &= \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix} \oplus \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{pmatrix} = \begin{pmatrix} v_1+u_1 \\ v_2+u_2 \\ v_3+u_3 \\ v_4+u_4+\frac{v_1 u_1}{v_1+u_1}(\frac{v_2}{v_1}-\frac{u_2}{u_1})(\frac{v_3}{v_1}-\frac{u_3}{u_1}) \\ v_5+u_5 \\ v_6+u_6 \end{pmatrix} \\
\phi_{cor}(i) &= (1, i_X, i_Y, 0, {i_X}^2, {i_Y}^2)^T \\
\chi_{cor}(v) &= \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\frac{v_4}{v_1}}{\sqrt{\frac{v_5}{v_1}-(\frac{v_2}{v_1})^2}\sqrt{\frac{v_6}{v_1}-(\frac{v_3}{v_1})^2}} = \frac{v_1 v_4}{\sqrt{v_1 v_5 - v_2^2}\sqrt{v_1 v_6 - v_3^2}}
\end{aligned}$$

Simple 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 [13], 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, we define a model configuration by combining the valuation domains for the variance and the covariance, similar to the correlation model:

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

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 as defined above. Additionally, the last component is additionally required to compute the variance as described previously for the variance model.

Logistic Regression Model Next, we consider the logistic regression model. 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 best of the authors' knowledge until now no exact single-pass algorithm has been proposed 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 exact computation of parameters for the logistic regression model relies on the trivial valuation basis so far.

DTM-classifier Next, we discuss two models based on the DTM-classifier [10]: It 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 S and the distribution in its complement $\bar{S}$. It is computed as

$$\sum_{y, x_1, \ldots x_k} (\sqrt{P_S(y|x_1, \ldots, x_k)} - \sqrt{P_{\bar{S}}(y|x_1, \ldots, x_k)})^2.$$

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

In the following, we assume that all attributes are binary for the sake of simpler notation. The generalization for non-binary attributes is straightforward. Furthermore, we assume that value combinations are arranged in a predefined order, where 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:

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

In this model configuration each component of the valuation basis corresponds to a combination of values. Please note, that in this case the valuation basis stored in each node of the GP-tree needs to store 2^k values, where k 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.

Bayesian Networks Bayesian networks have been proposed as complex target models for exceptional model mining [6]. Since to the best of the authors' knowledge there is currently no parallel single-pass algorithm for learning bayesian

networks — which is a complex task on its own — we cannot provide an efficient valuation basis for this model class here, but refer to the trivial valuation basis. However, there is ongoing research in that area [5] which can possibly be exploited in future work.

6 Evaluation

In this section, we present runtime evaluations of the proposed approach using UCI-datasets [14] as well as a scalability study in a large real world dataset.

6.1 Runtime evaluations on UCI Data

We evaluated the presented approach by performing runtime experiments using a set of well known UCI datasets. The algorithms were implemented in the open source data mining environment VIKAMINE[3][2]. The experiments were performed on a standard office PC with a 2.2 GHz CPU and 2 GB RAM. As search space we used all non-model attributes in the respective dataset. Numeric attributes were discretized in five intervals using equal-frequency discretization.

In a first set of experiments we compared the runtime of the GP-tree approach for different model classes. Due to the limited space we just show the results for the exemplary *credit-g* dataset, see Table 1. In this dataset, we used the attributes *duration* and *credit_amount* as model attributes, which were discretized if necessary. We excluded these attributes from the search space for all model classes to increase the comparability of the results. Experiments on other datasets showed similar characteristics. As can be seen in the table, the

Model Class	Model Attr.	2	3	4	5
Frequent Pattern	-	0.8	3.5	17.1	70.0
Subgroup Discovery	duration	0.8	3.6	17.0	69.3
Variance	duration	0.8	3.6	17.0	72.6
Linear Regression	both	0.9	3.8	18.6	77.3
Correlation Coefficient	both	0.9	3.8	18.7	77.8
DTM classifier	both	1.8	7.4	32.0	118.2

Table 1. 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).

runtimes differ only marginally. For the DTM-classifier, the results only differ by a small constant factor, which can explained by the more complex model configuration. The similarity of the runtimes is due to the fact that no pruning scheme is utilized; therefore, the search space is the same for all model classes.

[3] www.vikamine.org

Next, we performed an extensive runtime analysis of our approach using 19 datasets from the UCI repository. For that purpose, we compared the proposed GP-growth algorithm to a simple depth first search without any specialized data structure. Due to the runtime similarity for different model classes and due to the limited space, we only show the results for one model class, that is, the slope of the linear regression. The results are shown in Table 2. 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 two orders of magnitude faster for all datasets. These results clearly demonstrate the power of efficient data structures such as the GP-tree.

max depth	2		3		4		5	
dataset	DFS	GPG	DFS	GPG	DFS	GPG	DFS	GPG
adults	230.9	6.2	8905.7	10.4	> 6h	25.1	> 6h	65.6
autos	1.6	0.5	72.4	3.8	2279.5	21.9	> 6h	104.0
breast-w	0.2	0.1	0.9	0.1	4.5	0.1	15.8	0.1
colic	1.3	0.7	32.9	2.6	639.8	12.9	9940.8	46.4
credit-a	1.2	0.2	24.1	0.9	339.6	3.2	3849.3	9.8
credit-g	2.9	0.9	68.7	3.8	1202.0	18.6	16593.8	77.3
diabetes	0.3	0.1	2.6	0.1	16.9	0.3	87.8	0.4
forestfires	0.8	0.2	16.1	0.6	235.8	2.0	2670.2	4.3
glass	0.1	0.0	1.3	0.1	10.6	0.2	66.2	0.3
heart-h	0.2	0.1	2.7	0.1	23.9	0.3	168.9	0.6
hepatitis	0.2	0.1	3.3	0.7	40.0	3.3	373.2	12.4
housing	0.2	0.0	1.4	0.1	7.7	0.2	32.8	0.4
hypothyroid	10.4	1.4	247.7	5.1	4405.4	24.3	> 6h	131.3
ionosphere	3.1	1.9	154.7	22.5	5581.3	184.3	> 6h	1136.7
labor	0.1	0.0	1.0	0.1	12.0	0.3	108.4	0.6
segment	6.5	1.0	175.3	3.9	3437.8	16.1	> 6h	59.1
spambase	18.7	6.0	558.6	28.4	12498.8	196.1	> 6h	1869.1
vehicle	2.5	0.6	66.0	3.2	1367.3	15.0	> 6h	53.1
vowel	2.1	0.3	45.3	1.3	754.4	4.3	10064.1	10.4

Table 2. 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 with the GP-tree.

6.2 Scalability Study: Social Image Data

In the following, we present a short case study on real world data that shows the advantages of the presented approach in large scale applications. As a dataset,

we used publicly available metadata of pictures uploaded to the Flickr[4]-platform. More specifically, we crawled the view counts as well as all tagging information for all pictures geo-referenced to a location in Germany uploaded in 2010. We limited the dataset to tags with more than 1000 occurrences leading to a dataset of about 1.1 million instances and about 1200 tags that we used as describing attributes. 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 we performed the task of identifying 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 is reasonable, as due to the sparseness of the dataset less combinations of three tags might occur in dataset than combinations of two tags. Overall, the runtime improvements for the Flickr dataset are even larger than in the previous datasets, showing the scalability of our approach.

7 Conclusions

In this paper, we have proposed a novel approach for fast exhaustive exceptional model mining: We have introduced the concept of valuation bases as an intermediate condensed data representation and presented the general GP-growth algorithm for efficient exhaustive exceptional model mining. We discussed the applicability of the proposed approach by drawing an analogy to data stream mining, and provided implementation examples for several model classes. Our runtime experiments show improvements of more than an order of magnitude in comparison to a naive exhaustive depth-first search.

For future work, we aim to analyze methods for considering the diversity of pattern discovery results, e.g., [12] in order to improve the result sets. Another interesting direction for future research is the adaptation of other more advanced data structures, such as bit vectors, for exceptional model mining.

Acknowledgment

This work has partially been supported by the VENUS research cluster at the interdisciplinary Research Center for Information System Design (ITeG) at Kassel University, and by the EU project EveryAware.

[4] www.flickr.com

References

1. Atzmueller, M., Lemmerich, F.: Fast Subgroup Discovery for Continuous Target Concepts. In: Proc. 18th International Symposium on Methodologies for Intelligent Systems (ISMIS 2009). LNCS (2009)
2. Atzmueller, M., Lemmerich, F.: VIKAMINE - A Rich-Client Environment for Pattern Mining and Subgroup Discovery. In: Proc. LWA 2011 (KDML Track) (2011)
3. Atzmueller, M., Puppe, F.: SD-Map – A Fast Algorithm for Exhaustive Subgroup Discovery. In: Proc. 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2006). pp. 6–17. No. 4213 in LNAI, Springer Verlag, Berlin (2006)
4. Bennett, J., Grout, R., Pébay, P., Roe, D., Thompson, D.: Numerically Stable, Single-Pass, Parallel Statistics Algorithms. In: IEEE International Conference on Cluster Computing and Workshops (CLUSTER'09). pp. 1–8. IEEE (2009)
5. Bromberg, F., Patterson, B., Yaramakala, E.: Mining bayesian networks from streamed data (2003)
6. Duivesteijn, W., Knobbe, A., Feelders, A., van Leeuwen, M.: Subgroup Discovery Meets Bayesian Networks–An Exceptional Model Mining Approach. In: 10th IEEE Intl Conference on Data Mining (ICDM). pp. 158–167. IEEE (2010)
7. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns Without Candidate Generation. In: Intl. Conf. on Management of Data. pp. 1–12. ACM Press (2000)
8. Herrera, F., Carmona, C., González, P., del Jesus, M.: An Overview on Subgroup Discovery: Foundations and Applications. Knowledge and Information Systems 29(3), 495–525 (2011)
9. Klösgen, W.: Explora: A Multipattern and Multistrategy Discovery Assistant. In: Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining, pp. 249–271. AAAI Press (1996)
10. Kohavi, R.: The Power of Decision Tables. In: Proceedings of the European Conference on Machine Learning. pp. 174–189. Springer Verlag (1995)
11. van Leeuwen, M.: Maximal Exceptions with Minimal Descriptions. Data Min. Knowl. Discov 21(2), 259–276 (2010)
12. van Leeuwen, M., Knobbe, A.: Non-Redundant Subgroup Discovery in Large and Complex Data. In: Proceedings of the ECML/PKDD 2011 (2011)
13. Leman, D., Feelders, A., Knobbe, A.: Exceptional Model Mining. Machine Learning and Knowledge Discovery in Databases pp. 1–16 (2008)
14. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI Repository of Machine Learning Databases, http://www.ics.uci.edu/mlearn/mlrepository.html (1998)
15. Petra Kralj Novak, Nada Lavrac, G.I.W.: Supervised Descriptive Rule Discovery: A Unifying Survey of Contrast Set, Emerging Pattern and Subgroup Mining. Journal of Machine Learning Research 10, 377–403 (2009)
16. Umek, L., Zupan, B.: Subgroup Discovery in Data Sets with Multi-Dimensional Responses. Intelligent Data Analysis 15(4), 533–549 (2011)
17. Wrobel, S.: An Algorithm for Multi-Relational Discovery of Subgroups. In: Proc. 1st European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD-97). pp. 78–87. Springer Verlag, Berlin (1997)