Ranking Interesting Subgroups

Stefan Rueping STEFAN.RUEPING@IAIS.FRAUNHOFER.DE
Fraunhofer IAIS, Schloss Birlinghoven, 53754 St. Augustin, Germany

Abstract

Subgroup discovery is the task of identifying the top k patterns in a database with most significant deviation in the distribution of a target attribute Y. Subgroup discovery is a popular approach for identifying interesting patterns in data, because it combines statistical significance with an understandable representation of patterns as a logical formula. However, it is often a problem that some subgroups, even if they are statistically highly significant, are not interesting to the user. We present an approach based on the work on ranking Support Vector Machines that ranks subgroups with respect to the user's concept of interestingness, and finds more interesting subgroups. This approach can significantly increase the quality of the subgroups.

1. Introduction

Subgroup discovery (Klösgen, 1996) is a well-known learning task that is especially suited for finding patterns in data that are not only predictive, but also understandable to the user. Subgroup discovery can be seen as an instance of the task of local pattern detection, which is defined as the search for local accumulations of data points with unexpected high density with respect to some background model (Hand, 2002). In subgroup discovery usually a high number of subgroups are returned. This can be a problem, because, as we will argue later in this paper, the interestingness of a subgroup to a user is not directly dependent on its statistical significance. Hence, the user may have to go through many statistically significant, but ultimately uninteresting subgroups before discovering some interesting ones. This is partly a result of the fact that the statistically most significant subgroups tend to be very frequently occurring, such that they are very likely already known to a user with some insight into the application domain at hand.

In this paper, we define interesting subgroups as subgroups which are not only valid, but also unknown to the user, in accordance with (Lavrac et al., 2004). The problem with interestingness is that novelty with respect to the user's knowledge is hard to model, as it is practically impossible to list up all the user's prior knowledge before starting the pattern search.

We assume that it is very easy for the user to look at an individual pattern, and give information about whether or not this pattern is interesting to him, or at least more interesting to him than other patterns. Hence, the user will be presented a ranked list of patterns, and be asked to inspect the top patterns from this list and mark the ones that he finds interesting.

We will present an algorithm that will target the search towards more interesting patterns from this feedback. In particular, our approach is aiming for a fully automatic optimization of the interestingness of subgroups, such that no data mining expert is required to adjust the algorithms parameters and data representation to give the desired results (having a data miner standing by is of course the gold standard for tailoring machine learning approaches to the user's requirements).

The remainder of the paper is structured as follows: in the following section, we formally introduce the task of subgroup discovery. Section 3 introduces our new approach, which will be evaluated in Section 4. Section 5 presents some related work and Section 6 concludes.

2. Subgroup Discovery

The basic idea of subgroup discovery is to investigate all subsets of a database which can be described by a propositional logic formula, calculate a measure of statistical deviation of a variable of interest in this subset from the overall data set, and find those subsets with the statistical most significant deviation.

Let $A_1, \ldots, A_d$ be a set of attributes, each with a finite domain $dom(A_i) = \{v_{i,1}, \ldots, v_{i,k_i}\}$ and define

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

$X = dom(A_1) \times \ldots \times dom(A_d)$. Also, let $Y = \{0, 1\}$. For a fixed data set $D = ((x_i, y_i))_{i=1}^n \subseteq X \times Y$ we define $p_0 = \frac{1}{n}\sum_i y_i$ and for any subset $D' \subseteq D$ let $g(D') = |D'|/n$ and $p(D') = \frac{1}{|D'|}\sum_{(x_i,y_i)\in D'} y_i$. $g(D')$ is called the generality of D' and $p(D')$ the expectation of Y given D'. Also, for a propositional formula S over $A_1, \ldots, A_d$ let $ext(S)$ be the extension of S in D, i.e. the set of all $(x, y) \in D$ for which $S(x)$ is true. We define $g(S) = g(ext(S))$ and $p(S) = p(ext(S))$. S is called a subgroup descriptor and $ext(S)$ the corresponding subgroup (however, in the following we will also call S a subgroup if no confusion is possible).

Definition 1: Given a database D and some $a > 0$, the quality q of a subgroup S is defined as $q(S) = g(S)^a(p(S) - p_0)$

Note that for $a = 0.5$, which is usually used here, this quality is order-equivalent to a statistical Binomial test. The task of subgroup discovery can now be formally defined as follows:

Definition 2: Given a database D and an integer k, the task of subgroup discovery is to find among all possible subgroups the k subgroups with largest quality values q.

Fast algorithms for subgroup discover exist (Kavsek et al., 2003; Grosskreutz et al., 2008).

3. Finding Interesting Subgroups

Subgroup discovery has become popular because it effectively combines a statistical measure of deviation with a propositional logic hypothesis space. Finding the top k subgroups effectively directs the user's attention on subgroups with the highest statistical unusualness. However, the following example shows that the interestingness of a subgroup to the user is not the same as the statistical significance of the subgroup:

In the data set in Table 1, using ice cream sales as the target attribute Y, the subgroup `weather = good` $\rightarrow$ `sales = high` has a generality of $g = 4/8$ and a probability of $p = 1$, and hence a quality of $(4/8)^{0.5}(1 - 5/8) = 0.265$. The subgroup `advertised = yes` $\rightarrow$ `sales = high` instead has only a quality of 0.187. However, for a sales analyst the first subgroup is obvious, while the second one about the reaction of sales to his marketing campaign is quite interesting and useful. This property is called *actionability* in (Lavrac et al., 2004). The ice cream example exemplifies how in order to find interesting subgroups, the user's background knowledge has to be taken into account. Unfortunately, encoding all the user's knowledge about the domain at hand much too time consuming and practically infeasible.

Table 1. Exemplary Data Set

Weather	Advertised	Ice Cream Sales
good	yes	high
good	no	high
good	no	high
good	no	high
bad	no	low
bad	yes	high
bad	no	low
bad	no	low

In the following we will present an approach that iteratively finds interesting subgroups, while taking only the user's feedback to a prior list of subgroups into account. The rationale is to find a level of detail that is specific enough for the user to give concrete feedback about a single real-world concept (instead of asking him about his general knowledge about the domain), and general enough to allow an efficient generalization of the information about interestingness (compared to asking about specific examples). Our hypothesis is that subgroups do provide the desired level of detail.

Our approach is based on the assumption that the interestingness of a subgroup to the user deviates from its statistical properties as expressed by $q(S)$ for one of the following three reasons:

The user is interested in subgroups with a certain **complexity**. While it seems obvious to prefer less complex subgroups, which are easier to understand, it also has been reported in (Bratko, 1996) that in the presence of prior knowledge experts actually prefer longer rules, saying that shorter ones "do not tell enough", regardless of their statistical quality. Hence, the choice of the right complexity level is non-trivial.

The user is more interested in certain **attributes** than others. For example, it often happens that two variables in a data set are highly correlated such that both may be adequate to explain a variable of interest. From a given data set, it may be impossible to distinguish between the effects of these two attributes, while for the user it can be very easy to see which one is the right explanatory attribute.

The user is interested in a certain **subset of the database**. Often, it can be quickly identified that many of the subgroups found in a database are not new to the domain expert, and hence it is interesting to target the search to an area where the expert can find novel, useful information.

3.1. Re-ranking Subgroups

Assume we have found a list of subgroups $\mathcal{S} = s_1, s_2, \ldots$ of subgroups, ordered by quality such that $q(S_1) \geq q(S_2) \geq \ldots > 0$. We present this list to the user and ask him to label the subgroups that are most interesting to him. Following the findings in the field of optimizing search engines from clickthrough data (Joachims, 2002; Radlinski et al., 2008), we do not expect the user to give a detailed assessment of the interestingness of each subgroup in the form of a numerical value, neither do we expect him to label all of the subgroups. However, we assume that he inspects some subgroups in the order that they are given, and among this subgroups marks all subgroups that are interesting to him. From the users input we can then infer that whenever the user labeled subgroup i as interesting and subgroup j not, where $j < i$, then the quality of subgroup i should not be less than that of subgroup j. Formally, we write $S_i > S_j$, or $i > j$ for short, when the user prefers S_i over S_j, such that for the true quality q^* it should hold that

$$S_i > S_j \quad \Rightarrow \quad q^*(S_i) \geq q^*(S_j). \tag{1}$$

Hence, from the user's input we collect a set $\mathcal{U} = (u_1, \ldots, u_n)$ of preference assessments $u_k = (i, j)$. These u_k will be the examples from which we learn the true quality values $q^*(S_i)$.

Our approach consists of three steps:

1. First, we construct a numerical representation $r(S)$ of subgroups that describes their statistical properties, the attributes they contain, and their relation to other, special subgroups.

2. We assume a multiplicative model for the user-defined interestingness q^* of subgroups, which has the important properties that (1) it contains the standard subgroup quality measure q as a special case for zero weights on the new attributes, and (2) by taking the logarithm it allows to convert the subgroup ranking problem into variant of the ranking SVM with a linear target function.

3. Using the weights from the SVM optimization procedure, we can construct a new subgroup discovery task that is more aligned with the user's expectations. This allows to construct an iterative optimization process.

Note that while Step 2 is a post-processing step on the previously discovered subgroups, Step 3 actually constructs new subgroups that are potentially more interesting. This is important, because as the number of possible subgroups scales exponentially with the dimension of the data set, it is practically infeasible to enumerate all potentially interesting subgroups and apply only a single post-processing step. The algorithm's pseudocode is given in Algorithm 1

Algorithm 1 Iterative Ranking Algorithm

```
Input: Data X, labels Y, subgroup parameter a0
a = a0; S = {}
while S not good enough do
  S = subgroupSearch(X,Y,a)
  R = getUserRanking(S)
  Z = convert2rankingExamples(R,S)
  (M,a) = optimizeRanking(Z)
  Y = model2labels(M,Y)
end while
Output: Subgroups S
```

3.2. Representation of Subgroups

3.2.1. Representing Subgroup Attributes

For each attribute in the original dataset we construct an attribute describing the subgroups. For the i-th attribute we set

$$r^*_{att,i}(S) = \begin{cases} 1 & \text{iff } S \text{ contains attribute } i \\ 0 & else \end{cases}$$

where a subgroup contains an attribute, if this attribute is contained in the rule defining the subgroup.

Following (Morik & Köpcke, 2004) it is advisable to use TF/IDF-like features that give higher attribute values to examples that appear less often. Hence, for an example set of n subgroups $S_1, \ldots, S_n$, we define the actual features r_{att} of each subgroup S as

$$r_{att,i}(S) \quad = \quad \frac{r^*_{att,i}(S)}{1 + \sum_{j=1}^{n} r^*_{att,i}(S_j)}$$

With this definition, features that appear in only few subgroups will receive a higher weight than features that appear in many subgroups.

3.2.2. Representing Subgroup Relations

From the set of preference pairs given by the user we can generate a list $\mathcal{S}_{rel}$ of especially relevant or irrelevant subgroups, which we define as the subgroups which most frequently over-ranked other subgroups, or were most frequently over-ranked, respectively. We will now define a set of features

$$r_{rel}(S) = r_{rel,T_1}(S), \ldots, r_{rel,T_k}(S)$$

that describes the similarity of a subgroup S to the set of subgroup $\mathcal{S}_{rel} = \{T_1, \ldots, T_k\}$, for some appropriate definition of similarity. The idea is that when a subgroup is known to be irrelevant, similar subgroups are likely to be irrelevant too. Likewise, if a subgroup is relevant, similar subgroups are likely to be relevant.

We define

$$r_{rel,T}(S) = \frac{|S \cap T|}{|S||T|}$$

which can be seen as a measure of dependency between subgroups S and T. The reasoning behind this definition is that when the user has defined a subgroup T as "special" (either especially relevant or irrelevant), all examples covered by T are assumed to be special, and hence all $S \subseteq T$ should be marked as special as well ($r_{rel,T}(S) > 0$). However, the larger T, the more likely it is that there exists local patterns insider T, such that it is harder to infer about a subset S from the properties of T. This is covered by the fact that for $S \subseteq T$ we have $r_{rel,T}(S) = \frac{1}{|T|}$. Likewise, for every superset of T we can assume that it is special, but less so the larger S is, and hence in this case $r_{rel,T}(S) = \frac{1}{|S|}$.

Other possible definitions of set similarity, such as the Jaccard coefficient, have been tested in a preliminary study, but none outperformed $r_{rel,T}(S)$. The number of subgroups to use can be defined by the user, in the experiments 2 subgroups per iteration were used.

Definition 3: For a subgroup S, we define the representation $r(S)$ of S as

$$r(S) \quad = \quad (\log g, \log(p - p_0), r_{att}(S), r_{rel}(S)) \quad (2)$$

where $g = g(S)$ and $p = p(S)$[1].

Note that we do not include trivial subgroups with $q(S) \leq 0$ in the analysis, such that $g(S) \neq 0$ and $p(S) > p_0$, so the logarithms are well-defined.

3.3. Ranking Optimization Problem

For each user assessment of relevance $u_k = (i, j)$ we define the ranking example $x_k := r(S_i) - r(S_j)$. Our goal is to compute a new ranking score q^*, such that

$$\forall_{u_k=(i,j)} : q^*(S_i) \geq q^*(S_j), \quad (3)$$

i.e. the examples are ranked according to the users definition. We will solve this problem by an approach similar to ranking SVMs (Herbrich et al., 2000).

[1]Throughout the paper, for a scalar a and a vector v we write (a, v) as shorthand for the vector $(a, v_1, \ldots, v_{dim(v)})$

We assume that q^* has the form

$$q^*(S) = g(S)^a (p(S) - p_0) \prod_{i=3}^{d} e^{w_{i-2} r(S)_i}$$

where $d = dim(r(S)) = dim(w) + 2$ and the index of $r(S)$ goes from 3 such that it matches with Definition 3 (note that the first two components of $r(S)$ are listed explicitely). The motivation is that on the one hand q^* includes the usual definition of subgroup quality from Definition 1 as a special case (for $w_i = 0$), such that by adjusting the parameters according to the user's input we can find a quality function that better resembles the user's perception of quality, but at the same time is as close as possible to the original one. The form of q^* also allows a linearization of the learning problem by taking the logarithm:

$$\begin{aligned} \log q^*(S) &= \log \left(g(S)^a (p(S) - p_0) \prod_{i=3}^{d} e^{w_{i-2} r(S)_i} \right) \\ &= a * \log(g(S)) + \log(p(S) - p_0) \\ &\quad + \sum_{i=3}^{d} w_{i-2} r(S)_i \\ &= (a, 1, w) r(S) \end{aligned}$$

where $(a, 1, w)$ is shorthand for the vector $(a, 1, w_1, w_2, \ldots)$. It can be easily seen, that the task defined in Equation (3) is transformed into finding an appropriate linear function $(a, 1, w)$. Additionally, we can optimize the influence of the features in w and the subgroup parameter a at the same time.

In order for the weight vector $(a, 1, w)$ to generalize well, we introduce a complexity term. In addition, we relax the problem to allow some misclassifications by introducing slack variables ξ_i, such that we solve the following SVM-like problem instead:

$$\begin{aligned} \frac{1}{2}||(a - a_0, 1, w)||^2 + C \sum_i \xi_i \quad &\rightarrow \quad \min \\ \text{subject to } q^*(S_{i,1}) \quad &\geq \quad q^*(S_{i,2}) - \xi_i \\ \xi_i \quad &\geq \quad 0 \end{aligned}$$

which is equivalent to

$$\frac{1}{2}(a - a_0)^2 + \frac{1}{2}||w||^2 + C \sum_i \xi_i \quad \rightarrow \quad \min \quad (4)$$

$$\text{subject to } (a, 1, w) x_i \quad \geq \quad -\xi_i \quad (5)$$

$$\xi_i \quad \geq \quad 0 \quad (6)$$

This optimization problem differs in two aspects from standard SVM ranking. First, in the target function in line (4), we use the term $(a - a_0)^2$, where a_0 is a

given parameter, instead of a^2, which might be expected. This is because the least complex hypothesis is not given at $a = 0$, as $g^0(p-p_0) = p-p_0$ is not a particularly sensible subgroup quality measure. Instead, as the given subgroups have been found by a subgroup search with some parameter a_0, and the least invasive choice is not to change this value.

Second, in the standard ranking algorithm, the right side of the correctness constraint in Equation (5) reads $\ldots \geq 1 - \xi_i$, where the 1 is necessary to introduce a margin between the ranked examples. In our approach this effect is introduced by setting the second component of the weight vector to 1. This prevents the ranking scores from shrinking arbitrarily close together, and instead scales the ranking values to be as close to the default quality $g^a(p - p_0)$ as possible.

It is straight-forward to derive and solve the dual formulation of the optimization problem. Another straight-forward extension of the algorithms would be the use of kernels. It would be particularly interesting to use a combination of a polynomial kernel for the attributes $r^*_{att,i}$ and a linear kernel for the other attributes. As a polynomial kernel of degree g represents also products of up to g attributes, this would allow to base the decision of interestingness of subgroups also the appearance of on combinations of subgroup features, and not only on single features independently.

3.4. Iterative Algorithm

So far, our algorithm is only a post-processing approach that re-ranks the results of standard subgroup discovery, and given a suitable representation of subgroups also other ranking learners could be applied. The drawback of a post-processing approach is that new subgroups that were not part of the original top k groups can not be discovered. In order to identify more interesting subgroups, a larger k has to be chosen, with a negative impact on runtime.

To avoid this problem, we now turn our approach into an iterative search for additional interesting subgroups. First, it is straight-forward to use the new parameter value a instead of the old a_0. Second, once we know that a certain subgroup S is estimated to be more or less relevant than defined by its quality measure q, we can use this knowledge to modify the labels y of the examples in this subgroup, such that its quality is raised or lowered directly, without the need to apply a re-ranking. The advantage is that with the new labels also the relevance of other subgroups that at least partly cover the corresponding examples is modified, allowing to focus the search on this region of the input space. This approach is related to (Kavsek et al., 2003; Scholz, 2005), where subgroups are removed from the data by sampling from an appropriately biased distribution.

In order to follow this approach, we replace the binary target values y with real values $y \in [0, 1]$, giving y a probabilistic interpretation. Note that the definitions of subgroup quality work with a real-valued y as well.

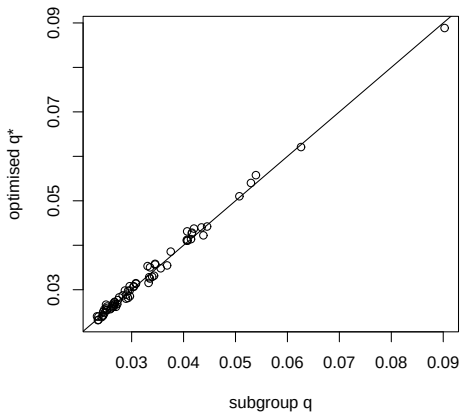

Figure 1. Exemplary plot of original subgroup q against optimized q^*

Assume that the ranking algorithm predicts the optimal quality value q^* of the subgroup S to be

$$q^*(S) \quad = \quad q(S)e^v \tag{7}$$

where the weight v is the element of the weight vector w found in the ranking algorithm that corresponds to the feature $r_{rel,S}$. We would like to modify the labels y of the examples in S such that $q'(S) = q^*(S)$ where q' is the standard quality given by the new labels y'.

A prerequisite of this approach is that the optimized q^* follow a similar statistical distribution as the q, such that is indeed theoretically possible that a distribution of the y exists that generates q^* directly. In the investigated data sets it turned out that this is indeed the case, which is exemplified for one data set in Figure 1.

Having established the dependency from Equation (7) between q and q^*, we can deduct that:

$$\begin{aligned} & q'(S) = q^*(S) \\ \Leftrightarrow \quad & g'(S)(p'(S) - p_0) = g(S)(p(S) - p_0)e^v \\ \Leftrightarrow \quad & p'(S) - p_0 = (p(S) - p_0)e^v \\ \Leftrightarrow \quad & \frac{1}{|S|}\sum_S y' - p_0 = (p(S) - p_0)e^v \end{aligned}$$

$$\Leftrightarrow \quad \frac{1}{|S|}\sum_{S}(y+\delta_y) - p_0 = (p(S)-p_0)e^v$$
$$\Leftrightarrow \quad p(S) + \delta_y - p_0 = (p(S)-p_0)e^v$$
$$\Leftrightarrow \quad \delta_y = (p(S)-p_0)(e^v-1)$$

where we modify each y by adding an identical constant offset δ_y. These step is executed for each subgroup for which an feature in the weight vector exists. After that, y-values are clipped to lie in $[0,1]$. With the new labels y', the process of subgroup search, user feedback, and ranking algorithm is re-started.

4. Experiments

It is hard to record and evaluate user feedback for a large number of data sets, in particular if a significant amount of domain knowledge is required. Hence, in the experiments we show instead that our approach can, by using user relevance feedback alone, flexibly adapt to patterns that are connected to the concept which is expressed by the target attribute.

We used some simulation over data sets from the UCI Machine Learning Repository (Asuncion & Newman, 2007). In our simulation approach, we used the classification label as the label describing the true concept the user is interested in. This label was never given to the learning algorithm, but only used to judge the interestingness of the subgroups found by the algorithm and to simulate the user's feedback. To start off the algorithm we selected the attribute in the data set with the highest correlation to the true label as a "proxy label" $\hat{y}$ instead, i.e. as the target label for subgroup discovery. An example of a real-world setting which closely follows this simulation is fraud detection, where the true fraud label is often not available, but is known to be correlated to high amounts of money spent.

In this way, we guarantee that the users interest and the actual learning task are correlated (of course the approach does not work if the data and the user's interest are totally independent). In addition, it guarantees that we indeed try to find a meaningful real-world concept and not some artificially defined goal.

To give a realistic example, the truly interesting target may be some kind of insurance fraud, which is not explicitely available in the data, such that a proxy attribute, such as the amount of money claimed, is used for analysis. Experts know that these two attributes are correlated, however it is equally well known that the two concepts are not identical (there are perfectly legal reasons for claiming high amounts of money).

We selected 15 data sets in total. Trivial example sets with less than 300 examples were not considered (it should be noted, however, that the number of examples in the data set is not the number of examples given to the ranking learner, as the ranking learner works on representations of *subgroups*). The datasets were preprocessed in the following way: multi-class datasets were converted into binary classes by classifying the majority class versus the rest. Numerical attributes have been discretized. Table 2 summarizes the data sets used in our experiments.

Table 2. Data Sets

Name	Size	Dimension
Diabetes	768	8
Breast-w	699	9
Vote	435	16
Segment	1000	18
Vehicle	846	18
Heart-c	303	22
Primary-tumor	339	23
Hypothyroid	3772	31
Ionosphere	351	33
Credit-a	690	42
Credit-g	1000	59
Colic	368	60
Anneal	898	63
Soybean	683	83
Mushroom	8124	110

Two different error measure were used to measure different aspects of the performance of the approach:

Area under the curve AUC: as the quality of the results may vary depending on how many subgroups one investigates, we investigated the following performance curve of the algorithm: for each k between 1 and 100, the number of the true top k subgroups that appeared among the top k ranked subgroups was counted. The area under this curve, normalized to lie between 0 and 1, is a measure for the algorithm's ability to discover the truly most interesting groups.

Kendall's τ as a measure of correlation between the true quality of the found subgroups (as defined by the true label and the true complexity parameter a) and their estimated quality (as predicted by the ranking algorithm). This is a measure of the algorithms precision, i.e. the internal consistency of the top k subgroups with the user's ranking.

We consider the AUC to be the central measure of the algorithms performance, because it most closely matches the intent of presenting the user the most interesting subgroups. For better comparison, we do not report the absolute values of these performance mea-

Table 3. Results per Data Set

Name	ΔAUC	$\Delta\tau$
Diabetes	0.256	0.008
Breast-w	0.759	0.120
Vote	0.664	0.051
Segment	0.596	0.601
Vehicle	0.053	0.500
Heart-c	0.180	0.036
Primary-tumor	0.739	0.532
Hypothyroid	0.729	0.307
Ionosphere	0.227	0.708
Credit-a	0.050	0.241
Credit-g	0.019	0.285
Colic	1.94E-4	0.213
Anneal	0.030	0.329
Soybean	1.94E-4	0.040
Mushroom	0.542	0.320
Mean	0.323	0.286

sures, but the increase Δ achieved over the unmodified initial set of subgroups.

For each data set, the algorithm was iterated 5 times, and the maximum value of each of these performance measures is reported. The use of the maximum instead of the average or the performance at the final iteration is justified, because the user is required to inspect each iteration anyway, and is free to stop the process at any iteration once he has found an interesting solution.

We did not use cross-validation in the experiments, as the algorithm does not work on the level of single examples, but on the level of subgroups. While the example sets of different cross-validation runs may be considered as independent enough to increase the validity of the results, the subgroups extracted from these data sets are highly similar, the most significant pattern being of course discovered in any sample. Hence, a cross-validation would run essentially over the same data set in all folds, rendering it useless. We instead opted to evaluate the algorithm over a large set of independent data sets, and to evaluate the statistical significance of the results by a Wilcoxon signed rank test, as suggested in (Demsar, 2006).

The results of the experiments can be seen in Table 3. One can see that for every data set there is an increase in both the AUC and the correlation. Overall, the mean increase of the AUC is 0.323. A Wilcoxon signed rank test confirms the better performance of the ranking approach with a p-value of less than 0.0004. The mean increase of the correlation τ is 0.286 with a p-value of less than 0.0004. This demonstrates that our approach significantly outperforms standard subgroup detection when looking for interesting patterns.

Table 4. Validation of Techniques

Name	ΔAUC	$\Delta\tau$
default	0.323	0.286
r_{att} removed	0.271	0.239
r_{rel} removed	0.307	0.272

On some data sets, the increase in the AUC is particularly small. To investigate this case further, we compare the ΔAUC with the similarity between the user's true interest and the original subgroup task, as measured by the absolute correlation between the true label and the proxy label in our simulation. It turns out that the three data sets with the least improvement in the AUC (colic, credit-g and soybean) are exactly the data sets with the minimal correlation between the true and proxy label. The correlation between ΔAUC and the similarity of the tasks is 0.590, which backs up the intuition that the ranking problem gets harder for increasingly independent user interests.

To validate the effect of the different techniques that were introduced in Section 3, we performed additional experiments where in each experiment one of the techniques was disabled. Table 4 gives the average performance of each of this variants over all data sets. As one can see, leaving out any of the proposed techniques reduces the performance of the algorithm. Little performance is gained by using the dimension attributes. However, this may be an effect of the simulation, which is more targeted at focusing the search on some areas of the input space and does not contain any differentiation between different attributes. There may be a higher effect in practice, when users actually prefer some description of a subgroup over another.

5. Related Work

In (Atzmueller et al., 2005) an overview over techniques for generating interesting subgroups using background knowledge is given, e.g. defining filters on rules and on the subgroup search space. Similar approaches exist in the field of association rules for example (Srikant et al., 1997) where it is suggested to target association rule mining by the definition of constraints that describe the patterns the user is interested in. However, these approaches are limited by the fact that the constraints and filters have to be given explicitely, such that they can be integrated into the pattern search beforehand. This requires a lot of work on the users side, and hardly allows the algorithm to be run automatically without supervision of a data

mining expert. Hence, these approaches are complementary to the work presented here.

The idea of re-ranking of results has been developed in the context of the optimization of search engines from clickthrough data (Joachims, 2002). However, in the case of search engines, the full set of documents relating to a query is available, while in our case it is practically infeasible to enumerate all subgroups.

6. Conclusions

Subgroup discovery is well suited for finding interesting patterns in data. In this paper, we have presented an approach based on learning rankings that effectively improves the interestingness of patterns from relevance feedback by the user. This allows not only to post-process existing lists of subgroups, but also to actively search for new subgroups that have not been found with standard subgroup quality measures. The approach improves the practical usability of the identified patterns without forcing the user to explicitely state all relevant knowledge beforehand.

A user study to compare the different approaches of standard subgroup discovery, subgroup discovery with manual optimization and our automatic optimization in terms of required effort and perceived gain in quality would be very interesting, but is outside the scope of this paper. It would also be interesting to compare the effectiveness of ranking examples with other possible types of user feedback, such as constraints or filters.

Acknowledgments: The author would like to acknowledge the financial support of the European Commission under the project RACWeB (No. 045101).

References

Asuncion, A., & Newman, D. (2007). UCI machine learning repository. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Atzmueller, M., Puppe, F., & Buscher, H.-P. (2005). Exploiting background knowledge for knowledge-intensive subgroup discovery. *Proc. 19th International Joint Conference on Artificial Intelligence (IJCAI-05)* (pp. 647–652).

Bratko, I. (1996). Machine learning: Between accuracy and interpretability. In G. Della Riccia, H.-J. Lenz and R. Kruse (Eds.), *Learning, networks and statistics*, vol. 382 of *CISM Int. Centre for Mechanical Sciences Courses and Lectures*, 163–177. Springer.

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

Grosskreutz, H., Rüping, S., & Wrobel, S. (2008). Tight optimistic estimates for fast subgroup discovery. *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (pp. 440–456). Springer.

Hand, D. (2002). Pattern detection and discovery. In D. Hand, N. Adams and R. Bolton (Eds.), *Pattern detection and discovery*. Springer.

Herbrich, R., Graepel, T., & Obermayer, K. (2000). Large margin rank boundaries for ordinal regression. In P. J. Bartlett, B. Schölkopf, D. Schuurmans and A. J. Smola (Eds.), *Advances in large margin classifiers*, 115–132. MIT Press.

Joachims, T. (2002). Optimizing search engines using clickthrough data. *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)* (pp. 133–142).

Kavsek, B., Lavrac, N., & Jovanoski, V. (2003). Apriori-sd: Adapting association rule learning to subgroup discovery. *Proc. 5th International Symposium on Intelligent Data Analysis* (pp. 230–241).

Klösgen, W. (1996). Explora: A Multipattern and Multistrategy Discovery Assistant. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy (Eds.), *Advances in knowledge discovery and data mining*, chapter 3, 249–272. AAAI / MIT Press.

Lavrac, N., Cestnik, B., Gamberger, D., & Flach, P. A. (2004). Decision support through subgroup discovery: Three case studies and the lessons learned. *Machine Learning*, *57*, 115–143.

Morik, K., & Köpcke, H. (2004). Analysing Customer Churn in Insurance Data - A Case Study. *Knowledge Discovery in Databases: PKDD 2004* (pp. 325–336). Springer.

Radlinski, F., Kurup, M., & Joachims, T. (2008). How does clickthrough data reflect retrieval quality? *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM)* (pp. 43–52).

Scholz, M. (2005). Sampling-Based Sequential Subgroup Mining. *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Databases (KDD'05)* (pp. 265–274).

Srikant, R., Vu, Q., & Agrawal, R. (1997). Mining association rules with item constraints. *Proceedings 3rd International Conference on Knowledge Discovery and Data Mining* (pp. 66–73). AAAI Press.