Cascaded Subgroups Discovery with an Application to Regression

Henrik Grosskreutz

Fraunhofer IAIS, Schloss Birlinghoven, 53754 St. Augustin, Germany
henrik.grosskreutz@iais.fraunhofer.de

Abstract. Subgroup discovery is a task from the area of Knowledge Discovery in Databases (KDD) that aims at finding interesting subgroups of a population. One problem with subgroup discovery algorithms is that many of them return a very high number of subgroups, including many redundant ones. In this paper, we present an approach to iteratively build up a set of subgroups for a numerical target attribute. The result is a additive representation of the patterns in the dataset, which can also be used as a regression model. The iterative scheme presented is similar to Transformation-Based Regression (TBR), an algorithm from the area of rule-based regression. While this is work in progress, first experiments show that the resulting sets of subgroups have a predictive accuracy that is similar to that of models generated by TBR, while the models are much more compact and arguably easier to interpret.

1 Introduction

Subgroup discovery [Klö96,Wro97] is a task from the area of Knowledge Discovery in Databases (KDD) that aims at finding interesting subgroups of a population. Unlike in classification or regression approaches, the goal of subgroup discovery is not primarily to come up with a predictive function, but instead to come up with a representation that is easily interpretable by a user and provides a compact description of the most interesting patterns in the data. Subgroup Discovery is a general approach that has shown to be useful in a variety of application scenarios (like medical consultation systems [ABP06], spatial analysis [KM02] and marketing campaign planning [LCGF04]).

One problem with subgroup discovery algorithms is that many of them return a very high number of subgroups. This is particularly true for algorithms that exhaustively search the space of subgroup descriptions and return *all* subgroup descriptions that are statistically significant. As a result, the user is often overwhelmed by a huge list of possibly redundant or uninteresting subgroups. In this paper, we consider the problem to construct a representative set of subgroups, that is we tackle what Klösgen called the "more ambitious [goal to construct] a best global system of subgroups" ([KZ02], Chapter 5.2). More precisely, we aim at building sets of subgroups that make up a good representation for *numerical* target attributes (like prices, costs or salaries). The case of numerical

subgroup discovery has gained little attention, unlike the case of binominal target attributes, for which some approaches have been proposed [LFKT02,Sch04].

Our approach to build up sets of subgroups is based on techniques from the domain of rule-based regression. In particular, we use an iterative scheme inspired by the symbolic regression technique Transformation Based Regression (TBR) [BKN+02] to incrementally build up a model of the data based on a set of subgroups. In some sense, subgroup discovery and rule-based regression complement one another: In the area of subgroup discovery, several techniques have been developed to efficiently calculated compact representations of local patterns, while rule-based regression techniques focus more on global models and predictive accuracy. Besides, subgroup discovery and rule-based regression techniques have in common that they emphasize representational issues [WI95].

Overall, our approach works as follows: Similar to TBR, our algorithm iteratively constructs a model, which essentially consists of a set of subgroup descriptions associated with an *offset*. The offset capture how being in that subgroup affects the target value. That is, the prediction for an example is calculated by taking the base prediction and *adding* the offsets of all subgroup descriptions matching that example. Our algorithm start with a simple (constant) model whose base prediction is simply the average target value. Thereafter, the iteration starts: First, the numeric label is predicted on the base of the model calculated to far. Thereafter, the algorithm searches for subgroups where the target values in the training set differs from the *prediction*. The description of the most significant subgroup is added to the model and the offsets are adjusted. Thereafter, the next iteration starts until no significant subgroup can be found.

The use of subgroups in this iteration scheme is appealing because subgroups represent subsets where the predictions significantly deviate from the actual values, and at the same time involve a large number of examples. Thus, the use of subgroup discovery techniques avoids selecting unrepresentative outliers. The output of our algorithm is a set of subgroup descriptions (a subgroup description is simply a set of conditions on attributes, like for example "cylinders=8" in a car dataset) annotated with an offset or impact on the output (like "+8000 €"). We believe that the resulting overall representation is very intuitive, as it is similar to many common mechanisms for price calculation.

As mentioned earlier, this is work in progress. We present first results obtained by applying our algorithm to four (slightly modified) datasets from UCI Repository [AN07]. The experiments show that our simple additive model, based on a set of subgroups, is much more compact than the representation obtained using TBR while it has a similar predictive accuracy. We also consider descriptive features of the resulting set of subgroups, like overlap and covering degree, as a means to compare the quality of the sets of subgroups.

The remainder of this paper is structured as follows: In Section 2 we formally introduce the task of subgroup discovery and thereafter illustrate it using an example from a medical domain in Section 3. Thereafter, we present our approach in Section 4. In Section 5 we present some experimental results. We review related work in Section 6, before we conclude in Section 7.

2 Subgroup Discovery

In this section, we will formally define the problem of subgroup discovery. Let $DB = \{R_1, ..., R_N\}$ be a *database* or *dataset*, consisting of N rows, each built up of $l + 1$ values. We distinguish one attribute c, called the *target attribute*, from the l ordinary attributes $\{a_1, a_2, ...a_l\}$. In this paper we only consider datasets where the target attribute is a *numeric* target attribute; for simplicity, we will assume that the domain of the target attribute is the reals. For the other attributes $\{a_1, a_2, ...a_l\}$, we assume finite domains $D(a_i) = \{v_{1,i}, ..., v_{m_i,i}\}$, that is we consider nominal non-target attributes.

A *subgroup description* sd is a set of *terms* $\{t_1, ..., t_k\}$ where every term t_i is a constraint on an attribute, i.e. t_i has the form $(a_i = v_i)$, $v_i \in D(a_i)$. The *length* of the subgroup description is the number of terms it is built of. Given a database DB and a subgroup description sd, the *subgroup extension* of sd on DB is the set of rows $R_j \in DB$ that satisfy all terms $t_i \in sd$. We call the number of rows in the extension of a subgroup the *size* of the subgroup.

A *quality function* q is a mapping from $DB \times sd$ to the reals. Intuitively, a quality function expresses how "interesting" a subgroup is. Several quality functions have been proposed, for the numerical, the binary and the multi-class case [Klö96,SW00]. As already mentioned, in this paper, we only consider *numeric* target attributes and we will only consider the "mean test" quality function

$$\sqrt{n}(m - m_0)$$

where m and m_0 denote the mean in the subgroup and in the overall population, respectively, while n denote the size of the subgroup. This quality function is based on the statistical mean test [Klö96] and thus has a neat formal foundation. Now the problem of *subgroup discovery* is defined as follows: Given a database DB, the quality function $q(DB, sd) := \sqrt{n}(m - m_0)$, and a number k, determine the k subgroup descriptions sd with maximum quality. Or, put more formally: return a set of k subgroup descriptions G such that

$$\forall sd : \neg sd \in G \rightarrow q(DB, sd) \leq q* \tag{1}$$

where $q* = \min_{sd \in G} q(DB, sd)$. Please note that the set G may not be uniquely determined (if there are many subgroups with the same quality). In this case, any set of subgroup descriptions that satisfies the above condition 1 is considered a valid result.

3 Motivation and Example

The work presented in this paper was largely motivated by the goal to detect patterns in health care service data, which was done in the context of the European Union Project iWebCare (`http://iwebcare.iisa-innov.com/`), which aims at building a platform for the facilitation of Fraud Detection in health care e-government services. In this context, we considered datasets with prescriptions

issued by doctors. Besides the cost, the prescriptions are described by a set of attributes, like the doctor's specialty, the region of the doctor's practice, the kind of disease etc. Unfortunately, no information is available as to whether a prescription is fraudulent, which prevented the use of a supervised, classification-based approaches in the style of [OFR06,BH02]. Instead, we aimed at detecting patterns in cost of prescriptions, focusing on effects that seem suspicious and may be a hint on fraud.[1]

Subgroup discovery seemed a promising approach to detect such patterns. However, one problem is that the sought-after patterns are likely to be hidden behind dozens or hundreds of other patterns - for example, it is a well-known fact that medical branches or specialties that rely more heavily on technical equipment cause higher costs; the region has an effect on the cost; and many more. Please note that every combination of these subgroups also builds a - statistically - interesting subgroup, like radiologists working in urban regions. One way to overcome these difficulties might be the adaptation of one of the techniques proposed for binominal target attributes [LFKT02,Sch04], but instead we opted for a representation that expresses the effects represented by individual subgroups in a cumulative respectively additive way.

3.1 A Simple Motivating Example

To illustrate our approach, we use a simple example with hypothetical data, inspired from a medical domain. The data is shown in Table 1. The rows represent medical prescriptions. As target attribute, we consider the cost of the prescription. Beside this target attribute, the prescriptions contain the doctor's specialty and the information whether the doctor's practice is in an urban or a rural environment.

Cost	Specialty	Region
50	Surgery	Urban
20	Internal Med	Urban
20	Internal Med	Urban
10	Internal Med	Rural
40	Surgery	Rural
40	Surgery	Rural

Table 1. Prescription example

In this example, the three subgroup descriptions with positive quality are

- $\{Specialty = Surgery\}$ (with mean 43.33 and size 3),
- $\{Specialty = Surgery, Region = Urban\}$ (with mean 50 and size 1), and

[1] The question whether a statistically unusual prescription is actually fraudulent is clearly beyond the scope of this paper.

– $\{Specialty = Surgery, Region = Rural\}$ (with mean 40 and size 2)

and the corresponding subgroups consist of the rows that fulfill these conditions (that is rows 1, 5 and 6; resp. row 1; resp. rows 5 and 6).

While these two subgroups are surely interesting, they arguable do not optimally capture the structure of the data. In fact, a close inspection of the data shows that the price can be explained by two effects: Surgery is more costly than internal medicine, and being in an urban regions also results in higher costs, compared to rural regions. These effects could be described by the following set of subgroups, where the effects are *additive*, meaning that a prescription dealing with surgery and issued in an urban region is affected by both effects:

– $\{Specialty = Surgery\}$ (with offset 30 and size 3)
– $\{Region = Urban\}$ (with offset 10 and size 3)

The above set of subgroups express that the specialty surgery affects the price by augmenting it on average by 30, and that similarly the fact that the region is urban increases the price by 10.[2] As already mentioned, the model is additive, meaning that a prescription dealing with surgery and made in an urban region is affected by both effects.

The algorithm we will present in Section 4 constructs models that are based on this simple and easily interpretable representation.

4 The Algorithm

Before we present our algorithm, we will now formally define the structure of our models. A model consists of

– A base prediction value *base* (a Real value);
– A set of subgroup descriptions annotated with an effect on the outcome. That is, a set of pairs $(sd_i, \mathit{offset}_i)$, where sd_i is a subgroup description as described in Section 2 and offset_i is a Real.

Our models have a simple additive semantics, meaning that if multiple subgroup descriptions match an example, the offsets of all matching subgroup descriptions are added. We will now make this precise.

Given a model $(base, (sd_i, \mathit{offset}_i)^n)$, the prediction for an example is determined as follows:

$$\mathit{prediction}(e) = base + \sum_{i|matches(sd_i,e)} \mathit{offset}_i \tag{2}$$

Here, $matches(sd, e)$ expresses that the example e satisfies all terms t in subgroup description sd.

[2] The base price for prescriptions that are neither in urban regions nor deal with surgery is 10.

Now that we have described our model representations, we present the overall algorithm. It is listed in the figure Algorithm 1. The algorithm proceeds as follows: First, a new model *model* is initialized which simply predicts the average value. Thereafter, the algorithm enters a loop. In this loop, a temporary copy of the dataset is created and the new target values are computed as the difference between the target value in the original dataset and the prediction (computed by ***calcPrediction*** as specified in Equation 2). Next, the most interesting subgroup *sg* is computed (using a standard subgroup discovery algorithm). This subgroup represents the part of the population where the deviation between the prediction and the actual target value is most significant. If this subgroup is not statistically significant (as determined by the function ***isSignificant*** discussed below), the current *model* is returned as result and the computation ends. Else, *sg* is added to the *model* and the parameters of *model* are optimized (by the function ***optimizeParameters*** discussed below). Finally, the next iteration starts.

Algorithm 1 Additive Subgroup TBR

```
Input: Database DB with numerical target
Output: A list of subgroups
avg := average target value in DB
model:= new model with offset avg and no subgroups
while true do
   currentDB := copy of DB
   for all examples e in currentDB do
      e.target := e.target - calcPrediction(model, e)
   end for
   sg:= Search for most significant subgroup in currentDB
   if not (isSignificant(sg, currentDB)) or sg ∈ model then
      return result
   end if
   add sg to model
   call optimizeParameters(model, DB)
end while
```

We will now discuss two functions called by the algorithm.

- The function ***isSignificant*** determines if a subgroup represents a significant pattern in the database. More precisely, it tests the null hypothesis that "the mean of the target values in subgroup and in the rest of the dataset do not differ". This is done by a t-test (for two independent samples with different sizes and variances). The details of the t-test can be found in any textbook on statistics. In our experiments, we used both significance threshold $\alpha = 0.95$ and $\alpha = 0.99$.
  We remark that the t-test as stopping criterion fits particularly well with the mean test quality function, which is itself based on this test.
- The function ***optimizeParameters*** optimizes the parameters *base* and $offset_i$ to best fit the dataset. This is essentially a linear regression task. Please note

that the optimization of the parameters of the model is important to ensure that the average of the prediction is (roughly) equal to the actual average of the target attribute in the original dataset. Without this step, the addition of new subgroups to the overall model, using the mean test quality function, would cause the prediction to monotonically increase, which would prevent the model to converge.
More precisely, the calculation of the optimal parameters is done as follows: First, we compute every set of examples that is matches by the same set of subgroups. That is, we compute every maximal set of examples E_i such that

$$\forall e_1, e_2 \in E_i : [\forall sd \in model : matches(sd, e_1) \equiv matches(sd, e_2)]$$

Let k be the number of these sets of examples. Now the optimal parameters can be computed by solving the following system of linear equations

$$n_1[base + \sum_{j|sd_j \in\ model\ \wedge\ matches(sd_j, E_1)} offset_j] = s_1$$

$$...$$

$$n_k[base + \sum_{j|sd_j \in\ model\ \wedge\ matches(sd_j, E_k)} offset_j] = s_k$$

where n_i is the number of examples in the i'th set E_i, s_i the sum of the target values of all examples in E_i in the original dataset, and $matches(sg_j, E_i)$ a shortcut for $\forall e \in E_i : matches(sg_j, e)$.

4.1 An Example and Some Considerations

To illustrate the algorithm, we will describe its computation on the simple example from Table 1.

The initial base prediction is 30. In the first iteration, the subgroup discovery searches for subgroups that have a target value with mean above 30. The most significant subgroup is *Specialty=Surgery*, which has a size of 3 and a mean of $m = 13.33$ (versus $m_0 = 0$, the mean of the difference of the original target value and the prediction). This subgroup is added to the model and its parameters are optimized, yielding $base = 16.66$ and $offset_1 = 26.66$. In the next iteration, the algorithm uses the new model to predict the target value (unlike in the first iteration, where the prediction was always 30, now there are two possible values for the prediction). The next search for subgroups yields the subgroup *Region=Urban*. This subgroup is added to the model and the parameters are optimized to $base = 10$, $offset_1 = 30$ and $offset_2 = 10$. In the third iteration no significant subgroups are detected and the algorithm ends. The resulting model looks as follows:

$base = 10$
$sd_1 = \{Specialty{=}Surgery\},\ offset_1 = 30$
$sd_2 = \{Region{=}Urban\},\ offset_2 = 10$

The model consists of two subgroups, a total of two conditions and three real values. The two subgroup have an overlap of 1 out of 3 rows (namely the row with Specialty=Surgery and Region=Urban), that is they have an average overlap of 33%. Overall, the additive model represents a compact characterization of the patterns in the example dataset.

However, there is a caveat. While in this example our algorithm effectively computes the optimal decomposition into two additive subgroups, this is not guaranteed. Indeed, even in this simple setting it is straightforward to construct examples where the algorithm gets stuck in a local optimum and does not come up with an optimal set of subgroups. As already mentioned in the introduction this is work in progress and Algorithm 1 is just a first attempt to construct good sets of additive subgroups.

Beside this limitation, there is another issue that deserves closer attention. The models generated by the our algorithm can include subgroups whose quality (as measured by the mean test quality function) is not positive on the original dataset. Table 2 shows an example: here, three medical specialties are considered. One results in a low costs, the second one in medium costs and the last one in a high costs.

Cost	Specialty
10	Internal Med
20	Neurology
30	Surgery

Table 2. Prescription example

When applied to this dataset, our algorithm returns the following model:[3]

$$base = 10$$
$$sd_1 = \{Specialty{=}Surgery\},\ offset_1 = 20$$
$$sd_2 = \{Specialty{=}Neurology\},\ offset_2 = 10$$

This model includes one subgroup description, *Specialty=Neurology*, whose quality on the original dataset is 0 (The subgroup was added to the model because Algorithm 1 considered the quality of the subgroups on the *difference* between the target values in the original dataset and the intermediate prediction). Even though according to the standard definition this subgroup is thus not interesting, in the above model it is an important part of the description of the patterns in the dataset.

[3] We remark that in this example the significance level has to be decreased, because the dataset consists of just three examples.

5 Evaluation

To evaluate the quality of the subgroup sets respectively of the subgroup-based model computed by our algorithm, we considered both predictive and descriptive evaluation measures.

We ran our experiments on four (modified) datasets from UCI Repository [AN07]: the auto dataset; the housing dataset; the servo dataset; and finally the solar flare 2 dataset[4] . As our algorithm cannot deal with non-target attributes that are numerical, for the auto dataset we omitted all numerical (non-target) attributes. For the other three datasets, we discretized all numerical (non-target) attributes into four equally sized bins. We also removed all examples involving missing values. Table 3 summarizes the characteristics of the four datasets.

Dataset	number of attributes	number of examples
autos	10	199
housing	13	506
servo	4	167
solar flare	10	1066

Table 3. Datasets

We won't go into the details of our implementation except to remark that it made use of FP-Trees [HPY00] and (relatively simple) optimistic estimates [Wro97] to speedup the computation of the subgroups (see also the paragraph on fast subgroup discovery in the related work section). More details can be found in [GRSW08]. Using this implementation, for every dataset in Table 3 we could calculation a subgroup-based model in less than 30 seconds on an Intel Core 2 Duo E8400 with 3 GB of RAM under Windows XP.

5.1 Descriptive Evaluation Measures

Size of the Models Table 4 shows the size of the models obtained using our algorithm for values of 0.95 and 0.99 for α ("SG-0.95" and "SG-0.99"), as well as using TBR[5]. The table shows both the number of subgroups and the number of terms (i.e. conditions of the form "Attribute = Value") for the models generated by our algorithm; For TBR, the table shows the number of rules and the number of conditions.

[4] In the solar flare dataset, there are 3 candidate target attributes: the number of common flares, of moderate flares and of severe flares. In our experiments we simply used the sum of these flares as target attribute.

[5] We used a faithful reimplementation of TBR in Java, following the description given in [BKN+02]. As minor modification, we limited the maximal number of rules in a model to 200.

Dataset	SG-0.95	SG-0.99	TBR
	# Sg/ #Terms	# Sg/ #Terms	# Rules/ #Terms
autos	29 / 60	18 / 35	127 / 1016
housing	64 / 220	37 / 123	200 / 2019
servo	100 / 297	25 / 72	189 / 573
solar flare	43 / 168	26 / 93	189 / 1585

Table 4. Number of subgroups/number of terms in the models

The figures show that the models learned using our algorithm based on subgroups discovery are tremendously more compact than those using TBR. Intuitively, this is not really surprising, because in subgroup discovery the size of the subgroup affects the quality of a subgroup, which is not the case in the greedy rule creation algorithm of TBR.

Dataset	Algorithm	# SG p.e.	# overl. SG	overl. prop.	o. with largest o. SG
auto	Alg.1	1.06	3.45	0.71	0.57
	top-29	8.75	28	1	0.99
	closed-29	6.98	28	1	0.99
housing	Alg.1	1.98	5.97	0.91	0.63
	top-64	21.9	63	1	0.99
	closed-64	19.8	63	0.99	0.99
servo	Alg.1	2.51	5.24	0.89	0.79
	top-100	4.32	24.8	0.99	0.96
	closed-100	4.32	24.8	0.99	0.96
solar flare	Alg.1	0.73	3.44	0.83	0.74
	top-43	7.71	42	1	1
	closed-43	6.54	40.9	0.99	0.99

Table 5. Overlapping of the subgroups for $\alpha = 0.95$

Overlapping Next, we consider the amount of overlapping in the set of subgroups computed by our algorithm. The Tables 5 and 6 show, for $\alpha = 0.95$ respectively $\alpha = 0.99$, i) the average number of subgroups that match an example, ii) the average number of subgroups a subgroup intersects with, iii) the average proportion of the examples in a subgroups that are also matched by another subgroup and iv) the average overlap with the largest subgroup overlapping with a subgroup. For comparison, the table also shows the figures if instead of the subgroups computed by our algorithm one would simply use the most significant subgroups ("top-X") respectively the most significant closed subgroups

("closed-X")[6] Here, X corresponds to the number of subgroups contained in the model generated by our algorithm. For example, for the auto dataset and $\alpha = 0.95$ our algorithm returns a set of 29 subgroups and hence we listed the figures for the top 29 (closed) subgroups.

Dataset	Algorithm	# SG p.e.	# overl. SG	overl. prop.	o. with largest o. SG
auto	Alg.1	0.89	3.67	0.71	0.55
	top-18	6.1	17	1	0.99
	closed-18	5.5	17	1	0.99
housing	Alg.1	1.49	4.87	0.81	0.56
	top-37	13.5	36	1	0.99
	closed-37	12.5	36	0.99	0.98
servo	Alg.1	1.18	4.56	0.79	0.75
	top-25	2.53	15.4	0.96	0.88
	closed-25	2.53	15.4	0.96	0.88
solar flare	Alg.1	0.62	3.08	0.85	0.74
	top-26	4.99	25	1	1
	closed-26	4.67	25	0.99	0.98

Table 6. Overlapping of the subgroups for $\alpha = 0.99$

The figures clearly show that the overlap in the sets returned by our algorithm is smaller than in exhaustive or closed subgroup discovery. The figures also show that in the subgroup-based models generated by our algorithm, on average less than two or three subgroups match an example, meaning that on average the prediction can be calculated by about two summations.

5.2 Predictive Measures

Regression Performance Despite the fact that optimizing prediction performance is not the primary goal of subgroup discovery ([LKFT04]), we measured the predictive performance of our algorithm. More precisely, we calculated the root mean squared error (RMSE) in a 10-fold cross-validation. We compared the results with those achieved by TBR and by a state-of-the-art SVM regression algorithm, namely the improved SMO Algorithm for SVM Regression [SKBM00][7]. We considered both a linear and a quadratic Kernel. The results are shown in Table 7:

[6] Closed subgroups are computed by calculating all subgroups and leaving only one subgroup description for every set of subgroup descriptions that match exactly the same set of examples. This means that in the closed top subgroups there is no pairs of different subgroup descriptions that match exactly the same set of examples.

[7] We used the implementation of improved SMO provided by RapidMiner (formerly YALE [MWK+06]), using the default parameters.

Dataset	SG-0.95	SG-0.99	TBR	SVM-Linear	SVM-Quadratic
autos	4258	4342	4130	2815	3629
housing	5212	4988	6539	4858	6256
servo	0.52	0.51	0.63	0.9	0.52
solar flare	1.06	1.03	1.05	0.95	1.03

Table 7. X-Validation Performance (RMSE)

Overall, the experiments show that the models learned by our algorithm perform roughly as good as TBR. The SVMs perform slightly better. On the auto dataset, the RMSE using our algorithm is slightly higher than that using TBR, while the SVMs perform significantly better. On the housing dataset, our algorithm is clearly better than TBR. The result using an SVM depends on the Kernel: Using a linear Kernel, the results are better, while using a quadratic Kernel they are inferior. On the servo dataset, our algorithm performs best, with the quadratic SVM performing similarly. TBR results in a slightly higher RMSE, while SVM using a linear Kernel is clearly inferior. Finally, on the solar flare dataset all approaches perform roughly similar.

Summarizing, the prediction performance is solid but not outstanding (as one would expect from an approach focusing on representation issues and not on prediction accuracy).

6 Related Work

Our iterative approach involves several calculations of subgroups, which can become quite time consuming. Thus, it depends on algorithms that quickly perform subgroup discovery. Fortunately, recently several approaches have been proposed to speed up this task: For one, [AP06] proposed the use of efficient data structures based on FP-Trees [HPY00] to enhance the performance. For another, the use of optimistic estimates [Wro97] has been shown to allow to prune large parts of the search space and thus significantly increase the overall performance [GRW08]. Finally, randomized approaches have been proposed that allow to search for subgroups by considering only a sample of the overall dataset, while guaranteeing precise bounds on confidence and quality of solutions [SW00].

Other approaches to discover interesting pattern in numeric attributes include the search for "impact rules" proposed by Webb [Web01], which builds on earlier work by Aumann and Lindell [AL99]. Impact rules are quite similar to numeric subgroups: they consist of an antecedent (a conjunction of conditions) and a consequent, which describes the impact on the target variable. As done in standard numeric subgroup discovery [Klö96], the interestingness of impact rules is measured in terms of their cover (size) and mean target value. Webb also presents interesting strategies for the fast calculation of impact rules, based on his OPUS framework. However, unlike our algorithm these approaches merely

collect the rules with the highest impact value, but do not consider their interconnection as we do (by taking into account the effect of subgroups found in earlier iterations, treating their effects as additive).

As already mentioned in the introduction, our algorithm is quite similar to Transformation-Based Regression (TBR) [BKN+02], a rule-based regression technique based on Transformation Based Learning [Bri95]. TBR iteratively builds up a prediction model by refining the model by means of transformation rules. More specifically, in the $i+1$-th iteration a transformation rules takes as input the prediction of the i-th iteration and performs a linear transformation to obtain the (better-fitting) prediction of iteration $i+1$. The rules thus have the following structure: "If [condition$_{t+1}$] then prediction$_{t+1}$ = a_{t+1} + b_{t+1} * prediction$_t$", where a_{t+1} and b_{t+1} are Reals. Thus, every rule includes an additive and a multiplicative component.

Other approaches to rule based regression use pseudo-classes, which essentially corresponds to a discretization of the numerical target [WI95]. Regression trees [BFOS84] build up tree-structured prediction models that are, similar to rule-based learners, relatively simple to interpret.

The idea to iteratively search for subgroups, masking the effects of the subgroups already discovered in the subsequent iterations, was also the base for the Algorithm CN2-SD presented in [LFKT02]. To considers different parts of the instance space in each iteration, CN2-SD uses a weighted covering algorithm which assigns a smaller weight to examples already covered. The same weighting scheme has also been used in in other subgroup discovery systems like APRIORI-SD [KLJ03]. Another approach to take into account information or subgroups proposed by Scholz [Sch04] is to make use of sampling. Please note, however, that all these approaches consider the task of classification, while our algorithm (and TBR) consider the task of numeric regression.

The idea to incrementally build up a model by iteratively refining the prediction made in earlier stages is also underlying the technique of Boosting [FS99]. This idea is also related to the Cascade-Correlation Learning Architecture [FL90], which iteratively builds up a neural network layer by layer, where each layer builds on the previous, unmodified layers.

Of course, there is also a large body of work on regression respectively function learning that is not based on rules. In particular, regression approaches based on Support Vector Machines (SVM) [SKBM00] often achieve very good prediction performance. However, these approaches focus on predictive accuracy and not on obtaining compact, human-readable representations, and thus their models are often not easily interpretable. The same holds for neural networks like multilayer perceptrons. Some approaches exist that first build a neural network or SVM representation and thereafter extract rules based on the non-symbolic predictor constructed in the first stage [CS96,HBV06].

7 Summary and Discussion

In this paper, we have presented an approach to iteratively build up a set of subgroups for a numeric target attribute. Our approach brings together ideas from the area of subgroup discovery as well as from the field of rule-based regression. In particular, it uses an iterative scheme inspired by Transformation-Based Regression to incrementally build up its overall model. In every iteration, it performs a numeric subgroup discovery to obtain a description of that part of the dataset where the current prediction deviates the most from the actual target values. This description is added to the overall model, which consists of a base prediction together with a set of subgroup descriptions annotated with an (additive) effect on the prediction.

The models generated by our algorithm have an additive semantics, that is, the prediction for a given example is calculated by *adding* the offsets of all subgroup descriptions matching the example. The question how to calculate a prediction from a set of overlapping subgroups can also be answered in other ways: One possibility is to interpret the sequence of subgroups as an (ordered) decision list [Riv87] and to only consider the first matching subgroup description. Our additive interpretation, where the order of the subgroup descriptions is irrelevant, has the advantage that it is similar to familiar mechanisms for price calculation.

In the experimental section, we show that the resulting subgroup-based models have a predictive accuracy that is similar to that of TBR, while they are much more compact that the models generated by TBR. Furthermore, as the models have a simple additive semantics (unlike the linear transformations involved in TBR rules), they are arguable easier to interpret for a human user. Our experiments also show that the subgroups have less overlap than sets of subgroups computed simply by exhaustive search or exhaustive closed subgroup search.

As already mentioned, this paper presents work in progress and much remains to be done. In particular, we plan to compare the overlap of the subgroups generated by our algorithm with the overlap of subgroups generated using other techniques, like weighting or sampling [Sch04,LFKT02,KLJ03]. However, as these techniques have been proposed for binary target attributes, an adaptation to the numerical case will be necessary. We also plan to consider other subgroup quality functions than the mean-test quality function, for example quality functions that consider both positive and negative deviations. Finally, we plan to investigate whether it is possible to come up with better models by pruning the assembled set of subgroups. As already mentioned in Section 3.1, our current algorithm can get stuck in non-global optima even in relatively simple examples. We plan to address these issues in future work.

Acknowledgments: Part of the research presented in this paper has been funded by the European Union project "iWebCare".

References

[ABP06] Martin Atzmüller, Joachim Baumeister, and Frank Puppe. Introspective subgroup analysis for interactive knowledge refinement. In Geoff Sutcliffe and Randy Goebel, editors, *FLAIRS Conference*, pages 402–407. AAAI Press, 2006.

[AL99] Yonatan Aumann and Yehuda Lindell. A statistical theory for quantitative association rules. In *Knowledge Discovery and Data Mining*, pages 261–270, 1999.

[AN07] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

[AP06] Martin Atzmüller and Frank Puppe. SD-map - a fast algorithm for exhaustive subgroup discovery. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *PKDD*, volume 4213 of *Lecture Notes in Computer Science*, pages 6–17. Springer, 2006.

[BFOS84] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[BH02] R. Bolton and D. Hand. Statistical fraud detection: A review. *Statistical Science*, 17(3):235–255, 2002.

[BKN+02] Björn Bringmann, Stefan Kramer, Friedrich Neubarth, Hannes Pirker, and Gerhard Widmer. Transformation-based regression. In Claude Sammut and Achim G. Hoffmann, editors, *ICML*, pages 59–66. Morgan Kaufmann, 2002.

[Bri95] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

[CS96] Mark W. Craven and Jude W. Shavlik. Extracting tree-structured representations of trained networks. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 24–30. The MIT Press, 1996.

[FL90] S. E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 524–532, Denver 1989, 1990. Morgan Kaufmann, San Mateo.

[FS99] Y. Freund and R. Schapire. A short introduction to boosting. *J. Japan. Soc. for Artif. Intel.*, 14(5):771–780, 1999.

[GRSW08] Henrik Grosskreutz, Stefan Rüping, Nuhad Shaabani, and Stefan Wrobel. Optimistic estimate pruning strategies for fast exhaustive subgroup discovery. Technical report, Fraunhofer Institute IAIS, 2008. URL: http://publica.fraunhofer.de/eprints/urn:nbn:de:0011-n-723406.pdf.

[GRW08] Henrik Grosskreutz, Stefan Rüping, and Stefan Wrobel. Tight optimistic estimates for fast subgroup discovery. In *To appear at ECML/PKDD-2008*, 2008.

[HBV06] Johan Huysmans, Bart Baesens, and Jan Vanthienen. Iter: An algorithm for predictive regression rule extraction. In A. Min Tjoa and Juan Trujillo, editors, *DaWaK*, volume 4081 of *Lecture Notes in Computer Science*, pages 270–279. Springer, 2006.

[HPY00] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In Weidong Chen, Jeffrey F. Naughton, and Philip A. Bernstein, editors, *SIGMOD Conference*, pages 1–12. ACM, 2000.

[KLJ03] Branko Kavsek, Nada Lavrac, and Viktor Jovanoski. Apriori-sd: Adapting association rule learning to subgroup discovery. In Michael R. Berthold, Hans-Joachim Lenz, Elizabeth Bradley, Rudolf Kruse, and Christian Borgelt, editors, *IDA*, volume 2810 of *Lecture Notes in Computer Science*, pages 230–241. Springer, 2003.

[Klö96] W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*, pages 249–271. 1996.

[KM02] Willi Klösgen and Michael May. Spatial subgroup mining integrated in an object-relational spatial database. In *PKDD '02*, pages 275–286, London, UK, 2002. Springer-Verlag.

[KZ02] W. Klösgen and J. Zytkow, editors. *Handbook of Data Mining and Knowledge*. Oxford University Press, 2002.

[LCGF04] Nada Lavrac, Bojan Cestnik, Dragan Gamberger, and Peter A. Flach. Decision support through subgroup discovery: Three case studies and the lessons learned. *Machine Learning*, 57(1-2):115–143, 2004.

[LFKT02] Nada Lavrac, Peter A. Flach, Branko Kavsek, and Ljupco Todorovski. Adapting classification rule induction to subgroup discovery. In *ICDM*, pages 266–273. IEEE Computer Society, 2002.

[LKFT04] Nada Lavrac, Branko Kavsek, Peter Flach, and Ljupco Todorovski. Subgroup discovery with cn2-sd. *Journal of Machine Learning Research*, 5(Feb):153–188, February 2004.

[MWK+06] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale: Rapid prototyping for complex data mining tasks, 2006.

[OFR06] Pedro A. Ortega, Cristián J. Figueroa, and Gonzalo A. Ruz. A medical claim fraud/abuse detection system based on data mining: A case study in chile. In Sven F. Crone, Stefan Lessmann, and Robert Stahlbock, editors, *DMIN*, pages 224–231. CSREA Press, 2006.

[Riv87] Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.

[Sch04] Martin Scholz. Knowledge-based sampling for subgroup discovery. In Katharina Morik, Jean-François Boulicaut, and Arno Siebes, editors, *Local Pattern Detection*, volume 3539 of *Lecture Notes in Computer Science*, pages 171–189. Springer, 2004.

[SKBM00] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy. Improvements to the SMO algorithm for SVM regression. *IEEE-NN*, 11(5):1188, September 2000.

[SW00] T. Scheffer and S. Wrobel. A sequential sampling algorithm for a general class of utility criteria. pages 330–334, New York, NY, USA, 2000. ACM.

[Web01] Geoffrey I. Webb. Discovering associations with numeric variables. In *Knowledge Discovery and Data Mining*, pages 383–388, 2001.

[WI95] Sholom M. Weiss and Nitin Indurkhya. Rule-based machine learning methods for functional prediction. *Journal of Artificial Intelligence Research*, 3:383–403, 1995.

[Wro97] Stefan Wrobel. An algorithm for multi-relational discovery of subgroups. In Jan Komorowski and Jan Zytkow, editors, *Proc. First European Symposion on Principles of Data Mining and Knowledge Discovery (PKDD-97)*, pages 78–87. Springer, 1997.