2008 Eighth IEEE International Conference on Data Mining

Predicting Future Decision Trees from Evolving Data

Mirko Böttcher
University of Magdeburg
Faculty of Computer Science
39106 Magdeburg, Germany
miboettc@iws.cs.uni-magdeburg.de

Martin Spott
BT Group
Intelligent Systems Research Centre
Ipswich, IP5 3RE, United Kingdom
martin.spott@bt.com

Rudolf Kruse
University of Magdeburg
Faculty of Computer Science
39106 Magdeburg, Germany
kruse@iws.cs.uni-magdeburg.de

Abstract

Recognizing and analyzing change is an important human virtue because it enables us to anticipate future scenarios and thus allows us to act pro-actively. One approach to understand change within a domain is to analyze how models and patterns evolve. Knowing how a model changes over time is suggesting to ask: Can we use this knowledge to learn a model in anticipation, such that it better reflects the near-future characteristics of an evolving domain? In this paper we provide an answer to this question by presenting an algorithm which predicts future decision trees based on a model of change. In particular, this algorithm encompasses a novel approach to change mining which is based on analyzing the changes of the decisions made during model learning. The proposed approach can also be applied to other types of classifiers and thus provides a basis for future research. We present our first experimental results which show that anticipated decision trees have the potential to outperform trees learned on the most recent data.

1. Introduction

In many application fields almost every data collected is time stamped, or, as Kimball [14] noted: "The time dimension is the one dimension virtually guaranteed to be present in every data warehouse, because virtually every data warehouse is a time series". Due to its temporal nature such data not only captures influences, like management decisions or the start of marketing campaigns, but also reflects the changes of the underlying domain. Often, change can mean a risk (like a shrinking subgroup of target customers) or an opportunity (like an evolving market niche). In either case, it is in many domains not only imperative to detect change in order to survive or to win but inevitable for successful decision making. In fact, recognizing, analyzing, and acting upon change is a virtue which is somewhat natural to us humans and a necessity in order to cope with everyday life, from driving a car to stock investments. For this reason it may seem surprising that in the area of data mining change has been regarded more a burden than a fortune. While research in overcoming typical problems imposed by evolving domains, like decreasing classifier performance, has been prospering, studies in methods how to effectively utilize change are still rather rare.

Recently, there has been an increasing research interest in methods which aim at analyzing the changes within a domain by describing and modelling how the results of data mining—models and patterns—change over time. *Change Mining* has been coined as an umbrella term for this relatively novel research area. So far, research on change mining has solely focused on patterns such as association rules and clusters where it has been successfully used to solve a variety of problems, for instance interestingness assessment [16] and the detection of tiny clusters in noisy domains [10]. Nevertheless, the application of change mining to classification models still is a rather unexplored field.

Similar to how it is used for patterns, change mining can be applied to models with the goal of describing change, for example by analyzing how the relevancy or sensitivity of attributes changes. However, for models it is more insightful to ask: If we do know how a model, for instance a decision tree, changes over time, can we use this knowledge to learn it in anticipation, such that it better reflects the near-future

1550-4786/08 $25.00 © 2008 IEEE
DOI 10.1109/ICDM.2008.90

characteristics of an evolving domain? Predicting a future model provides at least two advantages: Firstly, the predicted model provides insight into a domain's future characteristics. Secondly, it can be expected that a predicted, future model will perform better than one learned on past or even the most recent data, particularly in cases where the underlying domain evolves at a fast pace.

Our goal in this paper is twofold: first of all, we want to show the potential of change mining for classifiers by presenting an algorithm that uses a model of change to learn decision trees in anticipation. More precisely, it models how the change within a domain affects those measures that control the process of decision tree induction. It then predicts the future values of the measures and induces a decision tree from the prediction. We start the discussion with related work in Section 2, a brief Section 3 on notation and a short introduction of decision trees in Section 4. Our approach for predicting decision trees is presented in Section 5, followed by experimental results in Section 6 and a discussion of the computational complexity in Section 7.

Secondly, we want to provide a more theoretic framework for this kind of predictive change mining by generalizing the approach taken for decision trees. In Section 8 we are discussing typical difficulties connected to change mining of classifiers in general and their prediction in particular. Thus motivated we propose a framework for what we call *process-centric* change mining in Section 9. It circumvents the discussed difficulties and also provides a basis for future research due to its genericness.

2. Related Work

To our knowledge, the area of change mining for classifiers in general and the prediction of future decision trees based on change information in particular has not been dealt with in the literature.

In the broader context of pro-actively retrieving a future model the RePro system [22] is the only approach known to us. However, the RePro system does not predict a completely novel future model but searches for the best match out of a repository of past models. It strongly assumes that models repeat in time following a predictable repetition pattern. This in turn is only likely to occur if the change triggering events repeat in time too—in the same order and each time with the same impact on the domain. This assumption, furthermore, has to hold for a rather long duration because unless data is collected very frequently and the domain changes often it will take a considerable amount of time to derive the huge number of models necessary to reliably learn transition patterns.

As already noted in the Introduction analyzing the change of patterns and models has been studied in the field of change mining. While conventional data mining takes one dataset and produces models or patterns upon it, change mining goes one step further in that it analyzes how models and patterns evolve over time. So far, research on change mining has primarily focused on describing *how* association rules and clusters are changing (cf. [1, 16, 4, 20]). Change mining approaches for patterns typically first derive a sequence of patterns which are then related and compared for changes. In Section 8 we will show that the application of such an approach to classifiers has some serious practical problems.

3. Notation

Throughout the paper we will make use of the following notation. We assume that a dataset $\mathcal{S}$ of sample cases is described by a set of nominal input attributes $\mathcal{A} := \{A^{(1)}, \ldots, A^{(m)}\}$ and a class attribute C. We assume that the domain of an attribute A has n_A values, i.e. $\mathrm{dom}(A) = \{a_1, \ldots, a_{n_A}\}$, and that the domain of attribute C has n_C values, i.e. $\mathrm{dom}(C) = \{c_1, \ldots, c_{n_C}\}$.

Since we are interested in how data changes over time, let $\mathcal{S}$ be a time-stamped data set and $[t_0,\ t_r]$ the minimum time span that covers all its samples. The interval $[t_0,\ t_r]$ is divided into $r > 1$ non-overlapping periods $[t_{i-1}, t_i[$, such that the corresponding subsets $\mathcal{S}^i \subset \mathcal{S}$ each have a size $|\mathcal{S}^i| \gg 1$. Without loss of generality, let $\hat{T} := \{1, \ldots, r, (r+1), \ldots\}$ be the set of all past ($i \leq r$) and future ($i > r$) period indices.

4. Decision Trees

Almost all algorithms for decision tree induction (cf. [18] and [6]) grow the tree top-down using a greedy strategy. The induction process is controlled by two different types of decisions: Firstly, starting at the root node an attribute A is selected that yields the highest score regarding an attribute evaluation measure I. The dataset is then split into n_A subsets each corresponding to one attribute value $a \in \mathrm{dom}(A)$ and a child node for each of them is created. Secondly, if all its cases have the same class label or a stop-criterion is reached, a subset is not split further and hence no children are created. The current node then becomes a leaf and is assigned the majority class $c \in dom(C)$ of its associated subset.

An attribute evaluation measure $I(C, A)$ rates the value of an attribute A for predicting the class attribute C. The most well-known measures are probably information gain [19] and information gain ratio [18]. The information gain $I_{gain}(C, A)$ measures the information gained, on average, about the class attribute C when the value of the attribute A becomes known. A disadvantage of the information gain is its bias towards attributes with many values. To

overcome this problem the information gain ratio $I_{gr}(C, A)$ was proposed which penalises many-valued attributes by dividing the information gain $I_{gain}(C, A)$ by the entropy of the attribute itself [19, 18].

5. Predicting Decision Trees

As an example for our approach to change mining we present the $PreDeT$ algorithm that anticipates future decision trees. It models how the measures which control the decisions during the tree induction process change over time, predicts their future values and derives a decision tree from the prediction.

5.1. Basic Idea

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

Figure 2(a) shows an example obtained from the same real world dataset which we also use for our experimental evaluation in Section 6. The information gain history of the attribute $A^{(1)}$ is stable apart from noise whereas the information gain history of $A^{(2)}$ shows an upward trend. Furthermore, it can be seen that for the vast majority of time periods $T = 1, \ldots, 15$ attribute $A^{(1)}$ has more predictive power and would therefore be chosen as the split attribute. However, due to the observed upward trend in the information gain of $A^{(2)}$ both histories will intersect and $A^{(2)}$ will become the split attribute in the near future.

Figure 2(b) shows the two histories from Figure 2(a) each modeled by a quadratic regression polynomial. In period 16 the – at the time of modeling unknown – information gain values of both attributes are marked. As it can be seen, the predictions made by the regression models anticipate the change in the ranking of candidate split attributes which happens between period 15 and 16.

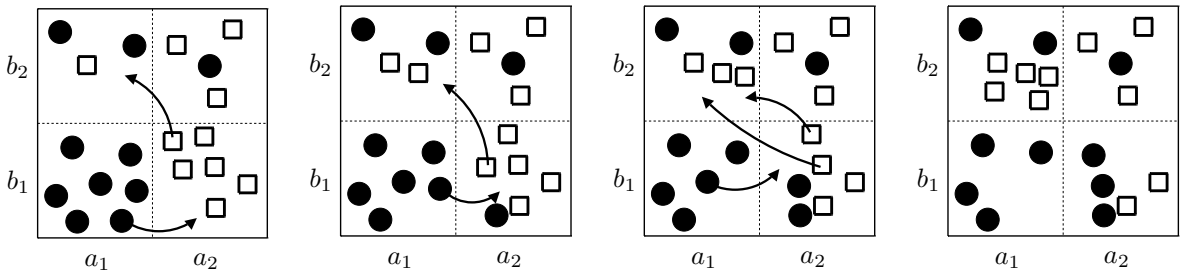
In summary, the basic idea of $PreDeT$ is to learn models which describe evaluation measure histories and class label distribution histories in each step of the decision tree induction. The models are then used to predict the value of the respective quantity for the next, future time period. Subsequently, the predictions are used to decide whether to grow a subtree and which class label to assign to a leaf node. As we already pointed out in Section 4 these two decisions are the main building blocks of the vast majority of decision tree learners. Because our algorithm leverages predictions for both it is finally capable to predict how a decision tree may look like in the future. In the presence of a changing domain this means that we should be able to provide classifiers with a higher accuracy than those which are solely reflecting the characteristics of historic data.

5.2. Predicting Attribute Evaluation Measures

Assume that we have a sequence of time-dependent data sets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ each described by the same attributes $A^{(i)}, i = 1, \ldots, m$ having the same domains in each time period. Quantities crucial for decision tree induction like attribute selection measure and the distribution of class labels are now related to a specific data set $\mathcal{S}^i$ and thus to a certain time period T_i. Therefore they form sequences of values which we will denote by $\mathcal{I} := (I(\mathcal{S}^1, A), \ldots, I(\mathcal{S}^r, A))$ for attribute evaluation measures and $\mathcal{P} := (P^1, \ldots, P^r)$ for the sequence of class label distributions. Thereby $P^k := (p^k_{1\cdot}, \ldots, p^k_{n_C\cdot})$ is the distribution of class labels and $p^k_{i\cdot}$ is the relative frequency of class attribute value i in time period k. We will refer to these sequences as attribute evaluation measure history and class label distribution history, respectively.

A model φ for attribute evaluation measures is a function $\varphi : \hat{T} \longrightarrow \mathbb{R}$. In general, it will be determined based on a history $\mathcal{I} := (I(\mathcal{S}^1, A), \ldots, I(\mathcal{S}^r, A))$ of attribute evaluation measures which will be denoted by $\varphi[\mathcal{I}]$. A model φ is then used in each inner node to obtain a prediction $\varphi[\mathcal{I}](r + 1)$ for attribute evaluation measure's value in the next time period T_{r+1}.

As the set of potential candidate models we chose the set of polynomials $\varphi(T) = \sum_{i=0}^{q} a_i T^i$ fitted to $\mathcal{I}$ using least squares regression. Linear regression in contrast to other possible model classes, like neural networks [11], offers the advantage that no large sample sizes (long histories) are re-



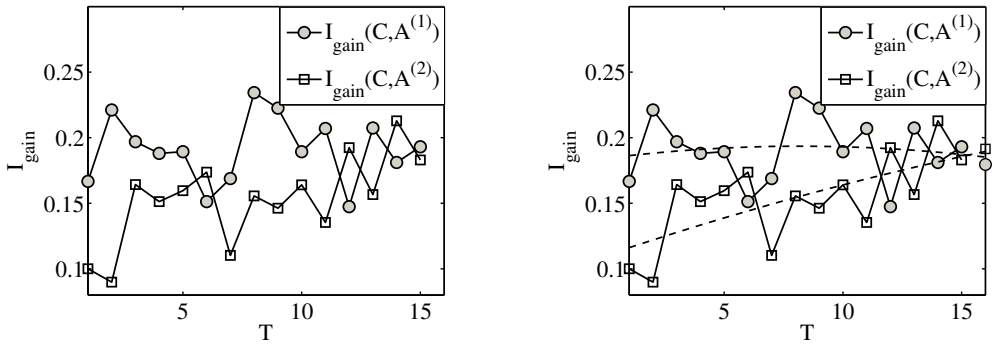
(a) Period 1: $I_{gain}(A) = 0.531$, $I_{gain}(B) = 0.007$

(b) Period 2: $I_{gain}(A) = 0.278$, $I_{gain}(B) = 0.030$

(c) Period 3: $I_{gain}(A) = 0.118$, $I_{gain}(B) = 0.066$

(d) Period 4: $I_{gain}(A) = 0.007$, $I_{gain}(B) = 0.191$

Figure 1. Illustration of how change within a domain can lead to trends in information gain.



(a) The history of $A^{(1)}$ is apart from noise stable. The history of $A^{(2)}$ shows an upward trend.

(b) Both histories modeled by quadratic polynomials shown as dotted lines. In period 16 the values to be predicted are shown.

Figure 2. Histories of information gain values for two different attributes

quired and that the underlying algorithms are fast. The first aspect is helpful when the domain changes rather fast. The latter aspect is important because models for a large number of histories need to be learned. The advantage of polynomial linear regression is, specifically, that it offers a simple way to obtain a set of candidate models by varying the degree q of the polynomial.

Having a set of fitted regression polynomials the best polynomial needs to be selected. In this case 'best' means that polynomial which provides the best trade-off between goodness of fit and complexity and is, for this reason, less prone to overfit the data. This can be measured using the Akaike information criterion (AIC) [2]. Let r be the number of observations, i.e. the length of the history, $q + 1$ the number of parameters of the polynomial and RSS the residual sum of squares of the fitted regression polynomial. Then AIC is defined as:

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

Commonly, the number of time periods for which data is available can be rather small. For example, the data we use for our experiments in Section 6 consists of 25 data sets obtained weekly. The original Akaike information criterion, however, should only be applied to data sets with large sample sizes [7], i.e. if $r/(q + 1) > 40$. To overcome this limitation a number of corrections of the Akaike criterion for small sample sizes have been developed. In our $PreDeT$ algorithm we use the following known as AIC_C [13]:

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

For large sample sizes r AIC_C converges to AIC, therefore it can be always used regardless of sample size [7].

5.3. Predicting the Majority Class in Leafs

A model ψ for histories of class label distributions is a function $\psi : \hat{T} \longrightarrow [0,1]^{n_C}$. It is learned from the history of class label distributions $\mathcal{P} := (P^1, \ldots, P^r)$. The dependency of ψ on $\mathcal{P}$ will be denoted by $\psi[\mathcal{P}]$. Within our $PreDeT$ algorithm a model ψ is used in each leaf node to predict the class label distribution at time point T_{r+1}.

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

$$\psi(\hat{T}) = \begin{pmatrix} \psi_1(T) \\ \psi_2(T) \\ \vdots \\ \psi_{n_C}(T) \end{pmatrix} = \begin{pmatrix} \psi_1(T) \\ \psi_2(T) \\ \vdots \\ 1 - \sum_{i=1}^{n_C - 1} \psi_i(T) \end{pmatrix} \tag{3}$$

To model each ψ_i we also use polynomials of degree q, i.e. $\psi_i = \sum_{j=0}^{q} a_j T^j$. The degree of the polynomials is, similar to Section 5.2, determined using the Akaike information criterion.

Because values $\psi_i(T)$ are relative frequencies additional constraints have to be imposed on the choice of the function ψ_i. In particular, the following should always hold.

$$\forall T \in \{0, \ldots, r+1\} \forall i \in \{1, \ldots, n_C\} : 0 \leq \psi_i(T) \leq 1 \tag{4}$$

In our experience this constraint can be too strict. For example, in the case $p_i^k = p_i^{k+1} = 1$ and $p_i^j \neq 1$ for $j \neq k$ and $j \neq k+1$ it is rather difficult to find a continuous, low-complexity model class for ψ_i. For this reason and because we only aim to predict values for the period $r+1$ we use the weaker constraint

$$0 \leq \psi_i(r+1) \leq 1 \tag{5}$$

Applying this constraint the model ψ_i cannot be derived using standard regression analysis anymore. Instead, we obtain the coefficients $\mathbf{a} := (a_0, \ldots, a_q)^T$ of the polynomial $\psi_i = \sum_{j=0}^{q} a_j T^j$ by solving the constrained linear least-squares problem

$$\mathbf{a} = \underset{\mathbf{a}}{\operatorname{argmin}} \frac{1}{2} \|C\mathbf{a} - \mathbf{p}\|_2^2 \tag{6}$$

$$\text{with } C := \begin{pmatrix} 1^0 & \cdots & 1^q \\ \vdots & \ddots & \vdots \\ r^0 & \cdots & r^q \end{pmatrix} \text{ and } \mathbf{p} := \begin{pmatrix} p_i^1 \\ \vdots \\ p_i^r \end{pmatrix}$$

There exist several methods from the field of optimisation for solving constrained linear least-squares problems. They will not be discussed here in greater detail. For further reading see [9].

5.4. Putting the Parts Together

Having explained the main building blocks of how to predict future decision trees in the previous two sections we will now go ahead and explain how they can be used in combination with a decision tree learner to anticipate future decision trees. This will finally lead us to the $PreDeT$ algorithm.

Figure 3 shows the $PreDeT$ algorithm. Similar to the vast majority of decision tree learners it consists of two consecutive stages. In the first stage (lines 1–8) the split attribute for the current node is searched. In the second stage (lines 9–18) it is decided whether the current node is a leaf (line 9) or inner node (line 15). Respectively, either a class label is assigned to the leaf node based on the majority class in this node, or the data sets are split according to the split attribute and the $PreDeT$ algorithm continues recursively (line 17). It should be clear that the basic ideas laid out in Section 5.2 and Section 5.3 can be used in connection with any decision tree learner that uses attribute evaluation measures to determine splits.

```
PREDET((S^1, ..., S^r))
 1  I_best ← WORTHLESS
 2  for all untested attributes A
 3  do I ← (I(S^1, A), ..., I(S^r, A))
 4     learn prediction model φ[I]
 5     Ĩ ← φ[I](r + 1)
 6     if Ĩ > I_best
 7        then I_best ← Ĩ
 8             A_best ← A
 9  if I_best = WORTHLESS
10    then create leaf node v
11         P^k ← (p^k_1., ..., p^k_{n_C·}), k = 1, ..., r
12         learn prediction model ψ[(P^1, ..., P^r)]
13         (p̃^{r+1}_1., ..., p̃^{r+1}_{n_c·}) ← ψ[(P^1, ..., P^r)](r + 1)
14         assign c = argmax_{c_i}(p̃^{r+1}_1., ..., p̃^{r+1}_{n_c·}) to v
15    else assign test on A_best to v
16         for all a ∈ dom(A_best)
17         do v.child[a] ←
18                PREDET((S^1|_{A_best=a}, ..., S^r|_{A_best=a}))
19  return v
```

Figure 3. Outline of the $PreDeT$ algorithm

In contrast to other decision tree learners $PreDeT$ takes as input a sequence of data sets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ representing time periods $1, \ldots, r$. It uses these data sets to estimate the value of the attribute evaluation measure in the next time period $r+1$ using a learned model φ (lines 4–5). The class label distribution within each data set is used to predict the likely class label distribution in time period $r+1$ using a learned model ψ (lines 11–13). Note that every decision

about the structure of the tree – the choice of the split attribute in inner and of the class label in leaf nodes – is solely based on estimated future values of the used metrics. For this reason the tree learned by $PreDeT$ can be seen as a prediction of the decision tree in period $r + 1$.

6. Experimental Evaluation

The $PreDeT$ algorithm does primarily depend on two parameters: first of all, the length r of the sequence of data sets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$, and secondly, the attribute evaluation measure I. In our experiments we evaluated how these factors influence the accuracy of the anticipated decision trees and how this accuracy compares to the one of conventionally induced decision trees.

For our experiments we chose a representative real-life dataset from the domain of Customer Relationship Management (CRM). The dataset contains answers of customers to a survey conducted by a telecommunications company over a period of 25 weeks. Each sample is described by 13 nominal attributes with a domain size between 2 and 9. Goal of the classification task is to predict whether a customer will be satisfied or dissatisfied with a certain service using the remaining 12 attributes, i.e. the data set has two classes to predict. In order to have an equal class distribution we did balance the original data set by removing samples of satisfied customers.

We split the original data set into 25 subsets $\mathcal{S}^i$, each corresponding to a time period of one week. The subsets contain between 243 and 399 samples. For each experiment we chose a sequence of r consecutive data sets $(\mathcal{S}^i, \ldots, \mathcal{S}^{i+r-1})$ within the available 25 ones. For each $i,\ i = 0, \ldots, 25 - r$ we then learned a decision tree using the $PreDeT$ algorithm and obtained classifications for the samples in the data set $\mathcal{S}^{i+r}$ that chronologically follows the sequence. For instance, for $r = 5$ we have 20 sequences, learn 20 decision trees and thus obtain the classification accuracy for 20 data sets.

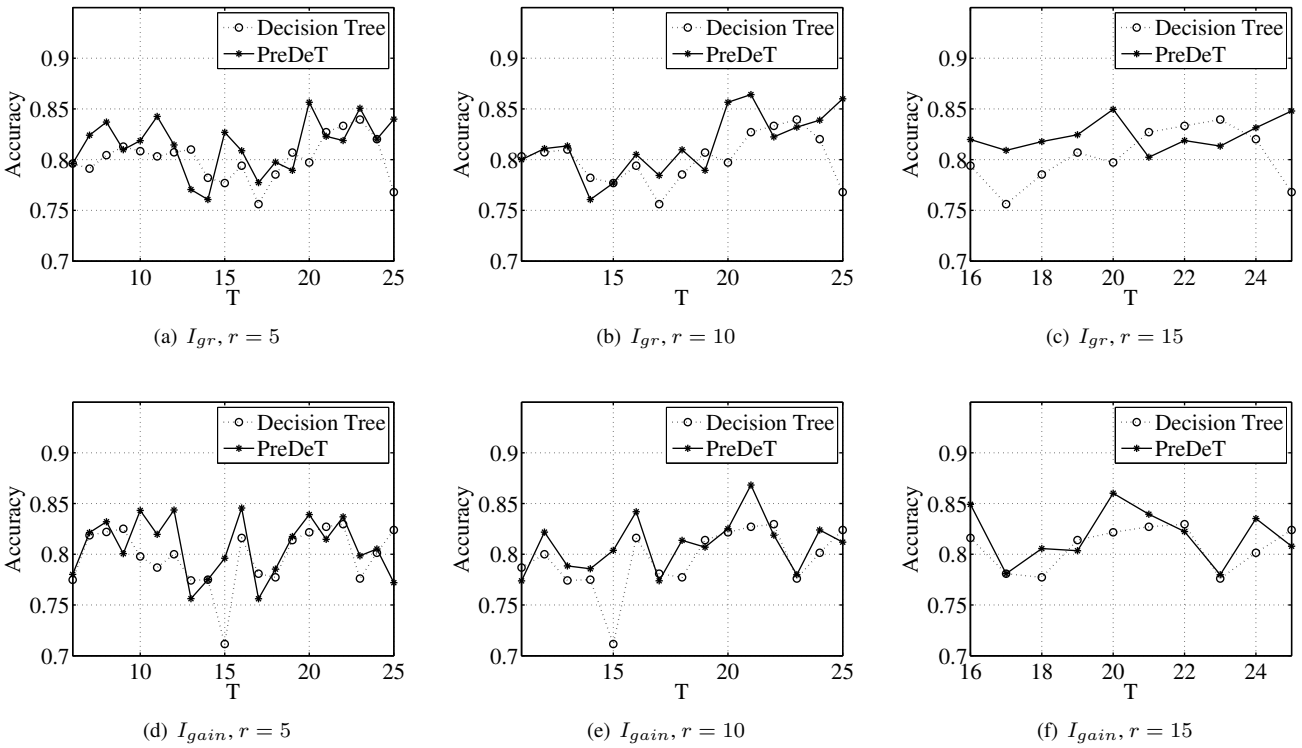
We did compare the accuracy of anticipated decision trees with those of conventionally induced ones[1]. In order to make a fair and realistic comparison two arguments need to be considered in the choice of the training data set for the induced decision trees. Firstly, when learning decision trees in the presence of concept drift it is common practise to use only the most recent data available because their characteristics are very likely to be best reflecting those of (unknown) near future data. It has been demonstrated by several authors that such a *temporal moving window approach* almost always outperforms trees which have been learned from all of the available data [21, 12, 15]. Secondly, from an abstract perspective $PreDeT$ (implicitly) learns a sequence of r decision trees each corresponding to a data set $\mathcal{S}^j, j = i, \ldots, i + r - 1$ and then anticipates the tree in the future period $i + r$ using a prediction model (cf. Section 9). As with any prediction model, the obtained prediction cannot have a better quality than its inputs. The quality of a decision tree, in particular its generalisation ability, does strongly depend on the size of the data set used for training. From this it follows that the trees anticipated by PreDeT do have a similar quality to trees that would have been learned directly on a data set with a size similar to those of each $\mathcal{S}^j, j = i, \ldots, i + r - 1$. For these two reasons, we chose to compare anticipated decision trees with decision trees induced from the most recent data set of each sequence.

Using the experimental setup above we carried out experiments using the information gain ratio I_{gr} and information gain I_{gain} and varied in each case the length of the sequence by using $r = 5, 10, 15$. Figure 4 shows the results of our experiments. For each combination of I and r the figure contains a chart whose abscissa axis shows the time period. The ordinate axis shows the classification accuracy of the anticipated decision tree (solid line) and the induced decision tree (dotted line) on each test data set.

For the information gain ratio I_{gr} we can see in Figures 4(a)–4(c) that for $r = 5$ the anticipated decision tree has a higher accuracy in 12 periods, equal accuracy in 2 periods, and a lower accuracy in 6 periods. For $r = 10$ the anticipated tree performs better in 9, equally in 1, and worse in 5 periods. For $r = 15$ it performs better in 7, and worse in 3 periods. This shows that anticipated decision trees outperform the induced ones on a considerably larger number of data sets. To show that the observed gain in accuracy is statistically significant we carried out a (one-tailed) Wilcoxon's signed ranks test which is the recommended test for comparing two models [8]. The test yields p-values of 0.033 for $r = 5$, of 0.0502 for $r = 10$, and of 0.0704 for $r = 15$. This means, for each r the null-hypothesis that the difference (accuracy(anticipated tree) – accuracy(induced tree)) has a median value lower than or equal to zero is rejected using a significance level of $\alpha = 0.05$ for $r = 5$, and of $\alpha = 0.1$ for $r = 10$ and $r = 15$, respectively.

For the information gain I_{gain} we obtain a similar result shown in Figures 4(d)–4(f). For $r = 5$ the anticipated tree outperforms the induced tree in 14 periods, has an equal accuracy in 1 period and performs worse in 5 periods. For $r = 10$ the anticipated tree has a greater accuracy in 10 periods and and a lower one in 5 periods. For $r = 15$ the anticipated tree has a greater, equal, and lower accuracy than the induced tree in 6, 1 and 3 periods, respectively. To assess the statistical signicance of the observed gain in accuracy we once more used a (one-tailed) Wilcoxon signed ranks test. The test yields for $r = 5$, $r = 10$ and $r = 15$ p-values of 0.0856, 0.03515 and 0.082, respectively, implying that anticipated trees performs statistically better than conven-

[1] We used the decision tree implementation by Christian Borgelt that can be obtained from http://www.borgelt.net/dtree



(a) $I_{gr}, r = 5$ (b) $I_{gr}, r = 10$ (c) $I_{gr}, r = 15$

(d) $I_{gain}, r = 5$ (e) $I_{gain}, r = 10$ (f) $I_{gain}, r = 15$

Figure 4. Classification accuracy for two different attribute evaluation measures in several consecutive time periods T using different sequences of r time periods length. For comparison, the performance of a conventionally induced decision tree is shown.

tionally induced decision trees in these three settings.

In terms of statistical significance the attribute evaluation measures I_{gr} and I_{gain} yield the best classification outcomes for different settings of the sequence length r, i.e. $r = 5$ for I_{gr} and $r = 10$ for I_{gain}. This gives rise to the assumption that the attribute evaluation measure is one factor the optimal choice of r depends on. Generally, the exploitation of this and other dependencies between parameters leaves space for further optimisations of the $PreDeT$ algorithm and is part of our future research agenda.

7. Computational Complexity

As already pointed out in Section 5 $PreDeT$ follows the same greedy algorithm structure which is also employed by the vast majority of decision tree learners. Here, we will discuss the additional computational effort needed by PreDeT and compare it with the one of a conventional decision tree learner. In the following discussion we will assume that $PreDeT$ receives a sequence of (sub-)datasets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ as its input while the conventional decision tree learner receives the data set $\mathcal{S}^r$. Further we assume that each dataset is of size n and that the number of attributes in each dataset is m.

In each step of the tree-growing process of a conventional decision tree learner each candidate attribute is examined and the one with the highest attribute evaluation measure is selected as the splitting attribute. The most time-consuming part is the calculation of the attribute evaluation measure. The algorithm must pass through each instance in a subset $\mathcal{S}^r$, for each of which it iterates through each candidate attribute. Because the union of the subsets of each level of the tree is $\mathcal{S}^r$ the time complexity for each level thus is $O(m \cdot n)$. Because a tree can have no more than m levels the overall complexity of a conventional decision tree learning algorithm thus is $O(m^2 \cdot n)$.

Obviously, while the conventional decision tree learner has to calculate the attribute evaluation measure values and the class label distributions in leaf nodes for only one data set, $PreDeT$ needs to calculate it for r data sets. The same holds for splitting datasets: a conventional learner has to split one while $PreDeT$ has to split r. Additionally,

$PreDeT$ learns a regression polynomial for each sequence of attribute evaluation measure values (line 4 in Figure 3)). This step involves solving a system of linear equations of size $(q+1) \times (q+1)$ whereby q denotes the degree of the regression polynomial. A solution can be obtained in $O((q+1)^3)$. Further, a constraint linear least square problem needs to be solved by $PreDeT$ to decide upon the class label of a leaf node (line 12 in Figure 3). The problem can be formulated by a $r \times (q+1)$ matrix and a solution obtained by a quadratic programming approach in $O(q^4)$. The degree q of the employed polynomial functions is almost always significantly small, typically values of $q = 1$, $q = 2$ or $q = 3$ are employed. For this reason, the effort for these two computations is very small compared to the effort needed to calculate the attribute evaluation measure and to split the data set such that it does not significantly contribute to the overall runtime of the algorithm.

Therefore, the computational effort of $PreDeT$ is approximately r-times higher than those of a conventional decision tree algorithm that uses $\mathcal{S}^r$ as its input leading to a time complexity of $O(r \cdot m^2 \cdot n)$. Mostly, however, the length of the sequence r will be rather low, for example in our experiments we did obtain good results for $r = 5$ and $r = 10$, so that the additional effort is still manageable.

The memory space complexity of $PreDeT$ is the same as for a conventional decision tree learner apart from the fact that it operates on more data. In particular, $PreDeT$ does not store the time series of statistics needed to anticipate decision trees, they are calculated on-the-fly during the run of the algorithm (cf. lines 3 and 13 in Figure 3).

8. Predicting Future Models

Having introduced an approach to predicting decision trees in previous sections, we will now discuss possible alternatives in more detail and finally generalize the chosen approach in Section 9 with a framework for *process-centric change mining*. Formally, the problem of predicting a classification model can be described as follows. Using a data mining approach a model $\mathcal{M}^i$ is derived for each time period $[t_{i-1}, t_i[, i > 0$ based on the data $\mathcal{S}^i$. Hence, we have two sequences: a sequence of data sets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ and a sequence of data mining models $(\mathcal{M}^1, \ldots, \mathcal{M}^r)$.

Given a a sequence of data sets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ we are interested in a data mining model $\mathcal{M}^{r+1}$ that represents the data $\mathcal{S}^{r+1}$ of the future time period $[t_r, t_{r+1}[$. Since $\mathcal{S}^{r+1}$ is unknown the model $\mathcal{M}^{r+1}$ cannot be derived in the usual way. To solve the problem two basic approaches are apparent:

1. to predict the data set $\mathcal{S}^{r+1}$ from the sequence $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ of previous data sets and then induce the model $\mathcal{M}^{r+1}$
2. to predict the model $\mathcal{M}^{r+1}$ directly from the sequence $(\mathcal{M}^1, \ldots, \mathcal{M}^r)$ of previous models

For the first approach, directly using the history of data sets for prediction is not promising, because the relationship between data points in different time periods is almost always unknown. Instead, one could try to derive models for data generation, predict a data generator for $[t_r, t_{r+1}[$, generate $\mathcal{S}^{r+1}$ and learn $\mathcal{M}^{r+1}$. This, in turn, is an instance of the second approach because it involves the prediction of a model based on a sequence of past models. One may object that there is a conceptual difference between data generating models and prediction models. Nevertheless, the difficulties linked to the prediction of a model based on a history of models are the same in both cases.

In fact the degree of difficulty depends on the type of model. Simple models based on parameterized probability distributions, for instance, can easily be predicted. However, models with a manageable number of parameters are only available for numeric data and even then usually too simple. Predicting complex models based on a history of past models, on the other hand, proves to be extremely problematic. Since a general discussion would be outside the scope of this paper, we will discuss the difficulties using decision trees as an example. Although they have a rather simple structure it turns out that the task of directly comparing them for changes is more than challenging. Three major difficulties can be identified:

- *Complexity*: To identify changes between two or more graph-based models structural matches between them must be identified and dissimilarities recorded. These, in turn, are instances of well-known problems from graph theory: the inexact graph matching, common subtree and tree editing distance problem. All of them are known to be NP-hard and also known to be difficult to approximate [3, 17].

- *Instability:* In particular decision trees are known to be instable [5]. This means, even small changes in the input training samples, e.g. due to noise, may cause dramatically large changes in the produced models. When comparing two models this means that even if the underlying domain remains stable some noise may cause a model to change fundamentally. For this reason, it is extremely difficult to distinguish true changes from noise-based changes.

- *Utility:* Imagine that a satisfying solution for the first two problems may exist. Then, a pair-wise comparison of a sequence of trees would likely yield a sequence of additions, deletions and changes that convert one tree into the other. The value of this sequence in terms of providing meaningful and actionable knowledge about a changing domain remains arguable, in particular in

combination with instability. The same holds for its utility as an input to a further (predictive) analysis step.

9. Process-centric Change Mining

In Section 8 we discussed that analyzing changes for decision trees and models in general is rather difficult, if the common change mining approach of direct model comparison is employed. As a solution we propose a generic approach that is based on a decomposition of the model induction process. Consider the induction process of a decision tree as an example. It can be described as a sequence of decisions comprising which attribute to take for the next split, when to stop growing the tree and which class label to assign to a leaf node. Decisions are driven by an attribute selection measure I like the information gain and the class label distribution P. Following the algorithm of tree induction, the sequence of decisions then uniquely determines the model. In other words, if we know all possible values – the image – of the information gain and the class label distribution, we can directly compute the model without going back to the data. In this respect, the image of I and P together form an *intermediate representation* of a decision tree that is sufficient for tree induction. An example for such an intermediate representation is the well known concept of *sufficient statistics* in probability theory. For instance, statistics can be sufficient to uniquely determine a probability distribution being the equivalent of our model.

More formally, when deciding on the attribute for the next split, we evaluate I on the data subset $\mathcal{S}' \subseteq \mathcal{S}$ of the current branch of the tree for all attributes A in the set of attributes $\mathcal{A}$. We then pick the attribute that maximizes (or minimizes) I. In other words, if we know the image $\mathcal{I} := I(\mathcal{A}, 2^{\mathcal{S}})$ of all possible combinations of attributes and subsets of $\mathcal{S}$, we have all the information required to pick the best attribute for a split at any stage of growing the tree. Adding the image $\mathcal{P} := P(2^{\mathcal{S}})$ of the class label distribution P forms the intermediate representation $IR = (\mathcal{I}, \mathcal{P})$.

In order to predict a future model we propose to look at how the intermediate representation changes over time, predict their future values and then follow the usual model induction process to derive the future model. Having a sequence of data sets $(\mathcal{S}^1, \ldots, \mathcal{S}^r)$ a sequence of intermediate representations $(IR^1, \ldots, IR^r)$ can be generated. Almost always, the values in IR^i will change over time, so do the decisions based on them and, finally, so does the resulting model. This means, by analyzing how the values of IR^i and thus the corresponding decisions change over time in each step of the induction process information about how the model will change can be retrieved. Figure 5 illustrates the proposed decomposition with predicted intermediate representation IR^{r+1} and induced future model $\mathcal{M}^{r+1}$.

In fact, as we have seen in Section 5 in case of decision

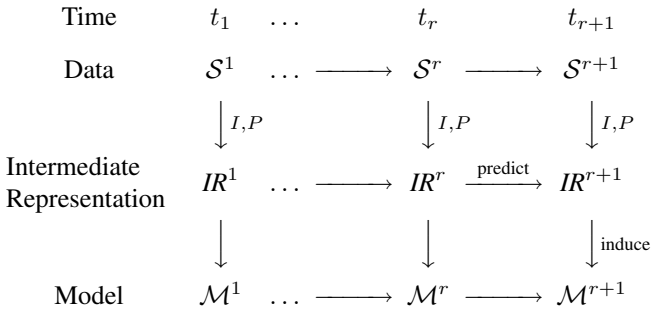


Figure 5. Process-centric change mining analyzes the sequences of intermediate representations IR^i

trees we do not necessarily need to compute and store the complete intermediate representation which can be computationally expensive. Instead we generate the required parts of the IR^i on the fly, when they are needed.

It is obvious that such an approach needs to be embedded into the learning process itself rather than being a subsequent analysis step as with the model-based approach. Because our approach to change mining is tightly coupled with the learning process we will call it *process-centric*.

The approach is useful only, if the intermediate representation is sufficient to derive the model, i.e. if the decomposition is well defined, and if the intermediate representation is of such a form that it can easily be predicted. As we have shown in Section 5 both requirements are met for decision trees. The second requirement shows that decision trees are indeed well suited as an example. Since the measures I and P form real-valued time series for every attribute-data subset combination, we can apply standard time series prediction to determine IR^{r+1}.

The described approach does not only apply to decision tree learners. It is suitable for other algorithms, for instance in the area of Bayesian networks. An example is the K2 algorithm. In general, every algorithm that builds upon a greedy strategy can be described as such a sequence of measure dependent decisions. More generally, the approach works wherever the mapping of data onto models can be decomposed into more or less complex sequences of mappings with intermediate representations. The basic mechanism remains the same, as long as we can find intermediate representations that can easily be predicted and that fully determine the model. A general proof that the diagram in Figure 5 commutes for a particular mining model like decision trees is difficult. Section 5.1 showed with an example at hand how a predicted intermediate representation reflects data of the following time period.

10. Conclusion

In this paper we provided a first research step into the field of change mining for classifiers. We presented the $PreDeT$ algorithm which predicts decision trees for future time periods by modelling how attribute evaluation measure and class label distribution evolve over time. First experimental results we obtained are very promising because they show that our approach is able to learn decision trees with a higher classification accuracy than approaches which only use the most recent data available. In particular, we only used a rather simple polynomial regression function for the prediction of the attribute evaluation measure in each step of $PreDeT$. It is likely that the results can be improved to a significant extent if more sophisticated prediction models are used.

We also showed that the approach of learning a sequence of models is unsuitable for both providing a basis for change mining and for future model prediction due to the complexity of direct model comparison. As an alternative we proposed process-centric change mining which is a novel approach to change mining that focuses on the analysis of changes in the decisions made during model induction.

We are currently working on the application of process-centric change mining to other types of models, in particular Bayesian networks. As part of this work we also look into proving that Figure 5 commutes under certain conditions. We are also working on several enhancements of the $PreDeT$ algorithm. In the first place, we investigate the advantages of using more sophisticated and more robust regression methods. A rather challenging research question to answer here is which type of model can best reflect the changes in the attribute selection measure. Secondly, as the algorithm stands a new decision tree has to be predicted every time a new batch of data arrives. For this reason it would be advantageous w.r.t. computational costs if $PreDeT$ supported incremental learning.

References

[1] R. Agrawal and G. Psaila. Active data mining. In M. Fayyad, Usama and U. Ramasamy, editors, *Proceedings of the 1st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'95)*, pages 3–8, Montreal, Quebec, Canada, 1995. AAAI Press, Menlo Park, CA, USA.

[2] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[3] P. Bille. A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1-3):217–239, 2005.

[4] M. Boettcher, D. Nauck, D. Ruta, and M. Spott. Towards a framework for change detection in datasets. In *Proceedings of the 26th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 115–128. Springer, 2006.

[5] L. Breiman. The heuristics of instability in model selection. *Annals of Statistics*, 24:2350–2383, 1996.

[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984.

[7] K. P. Burnham and D. R. Anderson. Multimodel inference: understanding AIC and BIC in model selection. *Sociological Methods & Research*, 33:261–304, 2004.

[8] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[9] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London, 1989.

[10] F. Hoeppner and M. Boettcher. Matching partitions over time to reliably capture local clusters in noisy domains. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledege Discovery in Databases (PKDD'07)*, pages 479–486. Springer, 2007.

[11] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[12] G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 97–106, New York, NY, USA, 2001. ACM Press.

[13] C. M. Hurvich and C. L. Tsai. Regression and time series model selection in small samples. *Biometrika*, 76:297–307, 1989.

[14] R. Kimball. *Data Warehouse Toolkit: Practical Techniques for Building High Dimensional Data Warehouses*. John Wiley & Sons, 1996.

[15] R. Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300, 2004.

[16] B. Liu, Y. Ma, and R. Lee. Analyzing the interestingness of association rules from the temporal dimension. In *Proceedings of the IEEE International Conference on Data Mining*, pages 377–384, San Jose, CA, 2001.

[17] L. Lovasz and M. Plummer. *Matching Theory*. Mathematics Studies. Elsevier Science, 1986.

[18] J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1992.

[19] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1996.

[20] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult. Monic – modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06)*, pages 706–711, Philadelphia, USA, Aug. 2006. ACM.

[21] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[22] Y. Yang, X. Wu, and X. Zhu. Combining proactive and reactive predictions for data streams. In *Proceeding of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'05)*, pages 710–715, New York, NY, USA, 2005. ACM Press.