Analysing Customer Churn in Insurance Data – A Case Study

Katharina Morik and Hanna Köpcke

Univ. Dortmund, Computer Science Department, LS VIII
morik@ls8.informatik.uni-dortmund.de
http://www-ai.cs.uni-dortmund.de

Abstract. Designing a new application of knowledge discovery is a very tedious task. The success is determined to a great extent by an adequate example representation. The transformation of given data to the example representation is a matter of feature generation and selection. The search for an appropriate approach is difficult. In particular, if time data are involved, there exist a large variety of how to handle them. Reports on successful cases can provide case designers with a guideline for the design of new, similar cases. In this paper we present a complete knowledge discovery process applied to insurance data. We use the TF/IDF representation from information retrieval for compiling time-related features of the data set. Experimental reasults show that these new features lead to superior results in terms of accuracy, precision and recall. A heuristic is given which calculates how much the feature space is enlarged or shrinked by the transformation to TF/IDF.

Keywords: preprocessing for KDD, insurance data analysis, customer relationship management, time-stamped data

1 Introduction

Insurance companies collect very large data sets, storing very many attributes for each policy they underwrite. The statistical analysis of their data has a long tradition ranging from the analysis of catastrophe insurance [1] and property and casuality insurance [2] to health insurance [3, 4]. The statistical methods are regularly used in order to model the credibility of customers, the amount of a single payment claimed, and the number of payments within a time span [5]. More recently, insurance companies also ask for data analysis in the context of their customer relationship management [6]. Direct marketing for cross- and up-selling is one of the current goals for data analysis. A special case is churn prediction, i.e. the prediction of policy termination before the end date. If those groups of customers or policies can be detected where the risk of churn is high, particular marketing actions can be pursued in order to keep the customers. Given this goal, it is not sufficient to model the distribution of churn or its overall likelihood, but policies or customers at risk should be actually identified. Then, the insurance salesmen can contact them. Along with the recent trend

J.-F. Boulicaut et al. (Eds.): PKDD 2004, LNAI 3202, pp. 325–336, 2004.

of customer relationship management, insurance companies move beyond statistical analysis. Knowledge discovery in insurance databases now builds upon datawarehousing projects in insurance companies (see, e.g., [7]).

In this paper, we want to present a knowledge discovery case whose solution was hard to find. We were provided with time-stamped data from the Swiss Life insurance company. There are many ways to handle time-related data, e.g., [8–13]. It is hard to select the appropriate approach [14]. The first question is, whether we actually need to handle the time information. There are cases, where the particular dates do not offer any additional information for the data analysis. A snapshot of the current state is sufficient for the prediction of the next state. In our case, ignoring the history of contracts did not succeed. Hence, we needed to take into account the time information. Since time was not given by equidistant events as in time series but by time-stamped changes to a contract, a promising approach is to learn event sequences. Another approach which has shown advantages in some knowledge discovery cases is the compilation of the time information into features. There are several compilation methods (e.g. windowing, single value decomposition [15]). We used the TF/IDF representation [16] from Information Retrieval to compile the time information into features. By the presentation of this case study we hope to offer a guideline for similar cases. Insurance is an important sector and, hence, many similar cases should exist. Moreover, we have turned the analysis into an efficient heuristic which applies to the raw data and estimates the size of the feature space after the transformation into TF/IDF attributes. The paper is organized as follows. First, we describe the insurance application in Section 2. Second, in Section 3 we describe the first experiments focusing on handling the history of contracts. Third, we describe the successful case in Section 4.1 and explain the effect of TF/IDF features to the size of the data set. We conclude by a proposal to gather successful cases of knowledge discovery at an abstract level and discuss related work in Section 5.

2 The Insurance Application

In the course of enhanced customer relationship management, the Swiss Life insurance company investigated opportunities for direct marketing [17]. A more difficult task was to predict churn in terms of a customer buying back his life insurance. Internal studies at the insurance company found that for some attributes the likelihood of churn differed significantly from the overall likelihood. However, these shifts of probabilities cannot be used for classification. Hence, we worked on knowledge discovery for the classification into early termination or continuation of policies. The discovered knowledge then selects customers at risk for further marketing actions and provides a basis to calculate the financial deposits needed in order to re-buy policies.

2.1 The Data

For the knowledge discovery task of churn prediction we received an anonymised excerpt of the data-warehouse of Swiss Life. The database excerpt consists of

12 tables with 15 relations between them. The tables contain information about 217,586 policies and 163,745 customers. The contracts belong to five kinds of insurances: life insurance, pension insurance, health insurance, incapacitation insurance, and funds bounded insurance. Every policy consists of average of 2 components. The table of policies has 23 columns and 1,469,978 rows. The table of components has 31 columns and 2,194,825 rows. Three additional tables store details of the components. The policies and components tables are linked indirectly by an additional table. If all records referring to the same policy and component (but at a different status at different times) are counted as one, there are 533,175 components described in the database. Concerning our prediction task, we can characterize the data in the following way:

Skewed data: Churn is only observed in 7.7% of the policies. Hence, we have the problem of skewed data [18].

High-dimensional data: Overall there are 118 attributes. If for the nominal attributes, their values would be transformed into additional Boolean attributes, we would have 2,181,401 attributes. In such a high-dimensional space visual data inspection or regular statistical methods fail. Even the MYSVM which is well suited for high-dimensional data cannot handle this number of attributes.

Sparse data: If the attribute values would be transformed into Boolean attributes, many of the attribute values would be zero.

Homogeneous accross classes: Those attribute values occurring frequently do so in the churn class as well as in the regular class.

All these characteristics contribute to the hardness of the knowledge discovery task. It becomes clear that the primary challenge lies in the mapping of the raw data into a feature space which allows an algorithm to learn. The feature space should be smaller than the original space, but should still offer the distinctions between the two classes, early termination of the contract and continuation. Finding the appropriate representation becomes even harder because of the time-stamps in the database. Each policy and each component may be changed throughout the period of a policy. For every change of a policy or a component, there is an entry in the policy table with the new values of the features, a unique mutation number, a code representing the reason of change, and the date of change. This means, that several rows of the policy table describe the history of the same policy. Each policy is on average changed 6 times, each component on average 4 times. Figure 1 shows the columns of the policy table which represent the time-stamps. The attribute VVID is the key identifying a contract. The attribute VVAENDNR is the unique number for a change. The attribute VVAENDART represents the reason of change. The attribute VVAENDDAT gives the date of change. As mentioned above, there are three alternative approaches to handling time-stamped data. The first choice that is successful quite often, is to ignore the time information. In our case, this means to select for each contract the row with the latest date (Section 3.1). The second choice is to explicitly model the sequential structures. In our case, this means that for each attribute of a contract, the begin date and the end date of a particular attribute

	VVID	VVAENDNR	VVWIVON	VVWIBIS	VVAENDDAT	VVAENDART	...
history of a contract	16423	1	1946	1998	1946	1000	
	16423	2	1998	1998	1998	27	
	16423	3	1998	1998	1998	4	
	16423	4	1998	1998	1998	54	
	16423	5	1998	1998	1998	4	
	16423	6	1998	9999	1998	61	
history of another contract	5016	1	1997	1999	1997	33	
	5016	2	1999	2001	1999	33	
	5016	3	2001	2001	2001	33	
	5016	4	2001	2001	2001	33	
	5016	5	2001	2002	2001	81	
	5016	6	2002	9999	2002	94	
	...						

Fig. 1. Extract of the policy table

value form a time interval (Section 3.2). The third choice is to compile the time information into the representation. Here, we counted for each attribute how often its value changed within one contract (Section 4).

3 First Experiments

3.1 Predicting Churn Without Time Information

Feature selection from the given database attributes is hardly obtained without reasoning about the application domain. Our first hypothesis was, that data about the customers could indicate the probability of contract termination. In this case, the changes of the contract components can be ignored. Therefore, we applied decision tree learning and MySVM to customer data, sampling equally many customers who continued their policy and those who re-bought it[1]. 10 attributes from the customer tables were selected and a Boolean attribute *churn* was generated from the raw data. The resulting set of eleven attributes was transformed into the input formats of the learning algorithms. Decision tree learning achieved a precision of 57% and a recall of 80%. MySVM obtained a precision of 11% and a recall of 57% with its best parameter setting (radial kernel) [21]. Trying association rule learning with the conclusion fixed to churn, did deliver correlated attributes. However, these were the same correlations that could be found for all customers [21]. The description of customers in the database does not entail the relevant information for predicting early contract termination. Changes in a customer's situation, e.g., buying a house, marriage, or child birth

[1] For decision tree and association rule learning we used WEKA [19], for support vector machine we used MySVM [20].

is not stored. These events can only indirectly be observed by changes of the policy or its components.

In a second experiment we focused on the components table. From the 31 database attributes of the components table, 7 attributes which are just foreign keys, or code numbers, or the start and end date of a component state were ignored. 24 attributes of the components table were combined with a year (1960 - 2002), stating when this attribute was changed. Some of the 1008 combinations did not occur. The resulting table of 990 columns plus the target attribute *churn* shows in each row the complete set of changes of a component and whether the corresponding policy was re-bought, or not. Learning association rules with the conclusion fixed to churn clearly showed the peak of changes at 1998 where the Swiss law changed and many customers re-bought their contracts. Other changes were just the procedure of contract termination, such as finishing the component and the payment. Using MYSVM , 44% precision and 87% recall were achieved using a linear kernel. These results show either that the data do not entail relevant information for churn prediction, or the representation prepared for learning was not well chosen. The first experiments were sufficient, however, to select only attributes that describe policies and no longer search within the customer data for reasons of early policy termination.

3.2 Predicting Churn on the Basis of Time Intervals

Taking into account the time aspect of the contract changes was considered an opportunity to overcome the rather disappointing results from previous experiments. There, time was just part of an attribute name. Now, we represented time explicitly. The time stamps of changes were used to create time intervals during which a particular version of the policy was valid. Relations between the time intervals were formulated using Allen's temporal relations [22]. Following the approach of Höppner [23] who applies the APRIORI algorithm to one windowed time series [24], a modification to sets of time series has been implemented [25]. For each kind of an insurance, association rules about time intervals and their relations were learned according to two versions of time. The first version handles the actual dates, finding (again) that according to a change of Swiss law many customers bought back their contracts around 1998. The second version normalises the time according to the start of the contract such that time intervals of changes refer to the contract's duration. By filtering out the policies around 1998 we intended to prevent the analysis from misleading effects of the law change. Typical patterns of policy changes were found. For example, one rule states the following: *If a component is signed and sometimes after this the bonus is adjusted, then it is very likely that directly after the adjustment a profit payment is prolonged.* The prediction of churn was tried on the basis of both, component data and a combination of component and policy data, applying biased sampling such that churn and continuation became equally distributed. The rules learned from the set of histories leading to churn and the rules learned from the set of continued policy/component histories did not differ. The same interval relations were valid for both sets. Hence, the features representing a sequence of

changes did not deliver a characterisation of churn. We are again left with the question whether there is nothing within the data that could be discovered, or whether we have just represented the data wrong for our task.

4 Using TF/IDF Features

In the first experiments we have taken into account the customer attributes, the policy attributes, the component attributes and the time intervals for the states of components and/or policy attributes. However, each change of a component was handled as a singleton event. Either there was a column for an attribute changing in a particular year, or there was a time interval for a state of an attribute. Given the yearly time granularity of the database, a component attribute can only change once in a year. Similarly, the time intervals were unique for one policy. Hence, we had the representation boiled down to Boolean attributes. Frequencies were only counted for all policy histories during the data mining step. Then, the values of a column were summed up, or the frequency of a relation between time intervals in all policies was determined. Whether the same component was changed several times within the same policy was not captured by the representation. Counting the frequencies of changes within one policy could offer the relevant information. It would be plausible that very frequent changes of a policy are an effect of the customer not being satisfied with the contract. If we transform the chosen excerpt from the raw data (about policies) into a frequency representation, we possibly condense the data space in an appropriate way. However, we must exclude the frequencies of those changes that are common to all contracts, e.g. because of a change of law. A measure from information retrieval formulates exactly this: term frequency and inverse document frequency (TF/IDF) [16]. *Term frequency* here describes how often a particular attribute a_i of c_j, the contract or one of its components, has been changed within a policy.

$$tf(a_i, c_j) = \parallel \{x \in time\,points \mid a_i\,of\,c_j\,changed\} \parallel \tag{1}$$

Document frequency here corresponds to the number of policies in which a_i has been changed. The set of all policies is written C.

$$df(a_i) = \parallel \{c_j \in C \mid a_i\,of\,c_j\,changed\} \parallel \tag{2}$$

Hence the adaptation of the TF/IDF measure to policy data becomes for each policy c_j:

$$tfidf(a_i) = tf(a_i, c_j) log \frac{\parallel C \parallel}{df(a_i)} \tag{3}$$

This representation still shrinks the data set, but not as much as does the Boolean representation.

4.1 Preprocessing

The first experiments have shown that the tables describing the customers can be ignored. We focused on the policy changes. We selected all attributes which

describe the state of the policy, ways, number, and amount of payments, the type of the insurance, unemployment, and disablement insurance. We ignored attributes which link to other tables or cannot be changed (such as, e.g. the currency). As a result, 13 attributes plus the identifier from the original attributes of the policy table were selected. One of them is the reason entered for a change of a policy. There are 121 different reasons. We transformed these attribute values into Boolean attributes. Thus we obtained 134 features describing changes of a policy. The TF/IDF values for the 13 original attributes we calculated from the history of each contract. We ordered all rows concerning the same policy by its time-stamps and compared in each column the successive values in order to detect changes. The term frequency of an attribute is simply the number of its value changes. For the 121 newly created features we counted how often they occurred within the changes. It was now easy to calculate the document frequencies for each attribute as the number of policies with a term frequency greater than 0.

4.2 Results

Since churn is only observed in 7.7% of the contracts MYSVM learned from a very large sample. We performed a 10-fold cross-validation on 10,000 examples[2]. The test criteria were accuracy, precision, recall, and the F-measure. If we denote positives which are correctly classified as positive by A, positives which are classified as negative by C, negatives that are correctly classified as negative as D, and negatives that are classified as positive by B, we write the definitions:

$$Accuracy = \frac{A+D}{A+B+C+D} \quad Precision = \frac{A}{A+B} \quad Recall = \frac{A}{A+C} \tag{4}$$

In order to balance precision and recall, we used the F-measure:

$$F_\beta = \frac{(\beta^2+1)Prec(h)Rec(h)}{\beta^2 Prec(h) + Rec(h)} \tag{5}$$

where β indicates the relative weight between precision and recall. We have set $\beta = 1$, weighting precision and recall equally. This led to an average precision of 94.9%, an accuracy of 99.4%, and a recall of 98.2% on the test sets. We wondered, whether these excellent results were caused by the chosen feature space, or due to the advantages of MYSVM . Therefore, we have performed 10-fold crossvalidation also using the algorithms Apriori, J48, and Naive Bayes. Table 1 shows the results. All algorithms were trained on the original attributes and then on the TF/IDF attributes. J4.8 clearly seleted the 'reason of change' attributes by its decision trees, in both representations. For MYSVM , the advantage of the TF/IDF feature space compared with the original attributes is striking. However, for all other algorithms, the TF/IDF feature space is also superior to the original one. The representation of input features contributes to the success of learning

[2] We also tried biased sampling but this delivered no enhanced results.

Table 1. Results comparing different learning algorithms and feature spaces

	Apriori		J4.8	
	TF/IDF attr	Original attr	TF/IDF attr	Original attr
Accuracy	93.48%	94.3%	99.88%	97.82%
Precision	56.07%	84.97%	98.64%	96.53%
Recall	72.8%	18.39%	99.8%	70.08%
F-Measure	63.35%	30.24%	99.22%	81.21%
	mySVM		Naive Bayes	
	TF/IDF attr	Original attr	TF/IDF attr	Original attr
Accuracy	99.71%	26.65%	88.62%	87.44%
Precision	97.06%	8.73%	38.55%	32.08%
Recall	98.86%	100%	78.92%	77.72%
F-Measure	97.95%	16.06%	51.8%	45.41%

at least as much as the algorithm. How can we describe more formally, whether the transformation into TF/IDF attributes expands or shrinks the data set? Is this transformation very expensive in terms of the resulting size of the data set? The Euclidian length of the vectors can be used as a measure of compression. A vector $\boldsymbol{x}$ of n attributes has the Euclidian length R

$$R = \sqrt{\sum_{i=1}^{n} x_i^2} \tag{6}$$

The data space with the original 13 attributes could be such that each attribute is changed m times giving us $\sqrt{13} \cdot m$ – the worst case. We found in our data that $m = 15$, i.e. no attribute was changed more than 15 times. If all attributes had that many changes, the feature space for the TF/IDF attributes would be $R = \sqrt{13} \cdot 15 = 54.08$. In this case, the feature space were not sparse and TF/IDF features would only be used if learning could not distinguish the classes otherwise. Note, that we investigate here the characterization of feature spaces, not the learnability. That is, we want to see, what the data transformation does to our raw data. We now estimate the average case for our data set. We assume a ranking r of attributes with respect to their frequency, e.g., $r = 1$ for the most frequent attribute. Experimental data suggests that Mandelbrot distributions [26]

$$tf_r = \frac{c}{(k+r)^{\phi}} \tag{7}$$

with parameters c, k and ϕ provide a good fit. The average Euclidian length using the distribution is:

$$R^2 = \sum_{r=1}^{d} \left(\frac{c}{(r+k)^2} \right)^2 \tag{8}$$

In our case $d = 4$ which means that only four distinct attributes have a value greater than zero. We bound $R^2 \leq 37$ according to the Mandelbrot distribution

and see that the average Euclidian length is significantly shorter than the worst case.

In order to ease the design process of new knowledge discovery cases, we want to transfer our analysis to new cases. In other words, we should know before the transformation whether the data space will be condensed, or not. A heuristic provides us with a fast estimate. We illustrate the heuristic using our case. We order the original table with time-stamps such that the states of the same contract are in succeeding rows. We consider each policy c_j a vector and calculate the frequency of changes for each of its n attributes $a_1 ... a_n$ in parallel "on the fly". We can then determine in one database scan the contract c_j with a maximum Euclidian length:

$$\hat{R} = \max_{c_j} \left(\sqrt{\sum_{i=1}^{n} tf(a_i, c_j)^2} \right) \tag{9}$$

If $\hat{R} \leq \sqrt{n}m$ where m is the maximum frequency, the transformation into TF/IDF features is worth a try, otherwise only strict learnability results could force us to perform the transformation. In our case $n = 13$ and $m = 15$ so that $\hat{R} = 22,91$ which is in fact less than $\sqrt{13} \cdot 15 = 54.08$.

5 Related Work and Conclusion

Time-related data include time series (i.e. equidistant measurements of one process), episodes made of events from one or several processes, and time intervals which are related (e.g., an interval overlaps, precedes, or covers another interval). Time-stamped data refer to a calendar with its regularities. They can easily be transformed into a collection of events, can most often be transformed into time intervals, and sometimes into time series. Time series are most often analysed with respect to a prediction task, but also trend and cycle recognition belong to the statistical standard (see for an overview [27, 28]). Following the interest in very large databases, indexing of time series according to similarity has come into focus [29, 30]. Clustering of time series is a related topic (cf. e.g., [31]) as is time series classification (cf. e.g., [32, 33]). The abstraction of time series into sequences of events or time intervals approximates the time series piecewise by functions (so do [12, 34, 35]). Event sequences are investigated in order to predict events or to determine correlations of events [15, 9, 11, 13, 36]. The approach of Frank Höppner abstracts time series to time intervals and uses the time relations of James Allen in order to learn episodes [23, 22]. Also inductive logic programming can be applied. Episodes are then written as a chain logic program, which expresses direct precedence by chaining unified variables and other time relations by additional predicates [37, 38]. Time-stamped data have been investigated in-depth in [10].

In this paper, we have presented a knowledge discovery case on time-stamped data, together with its design process. The design process can be viewed as the search for a data space which is small enough for learning but does not remove

the crucial information. For insurance data, the design process is particularly difficult, because the data are skewed, extremely high-dimensional, homogeneous accross classes, and time-stamped. The particular challenge was the handling of time-stamped data. Neither the snapshot approach nor the intervals approach were effective in our insurance application. The key idea to solving the discovery task was to generate TF/IDF features for changes of policies. The compilation of time features into TF/IDF has been analysed. Moreover, we have turned the analysis into an efficient heuristic which applies to the raw data and estimates the size of the feature space after the transformation into TF/IDF attributes. Our report on the insurance case might be used as a blueprint for similar cases. The heuristic and the generation of frequency features for time-stamped data with respect to the aspect of state change has been implemented in the MiningMart system [39][3].

Acknowledgment

We thank Jörg-Uwe Kietz and Regina Zücker for their information about the insurance practice and the anonymised database. For stimulating discussions on the support vector machine, mathematics, and representation languages we thank Stefan Rüping wholeheartedly.

References

1. Goldie, C., Klüppelberg, C.: Subexponential distributions. In Adler, R., Feldman, R., Taqqu, M., eds.: A practical guide to heavy tails: Statistical techniques for analysing heavy tails. Birkhauser (1997)
2. C.Apte, Pednault, E., Weiss, S.: Data mining with extended symbolic methods. In: Procs. Joint Statistical Meeting. (1998) IBM insurance mining.
3. Pairceir, R., McClean, S., Scotney, B.: Using hierarchies, aggregates, and statisticalk models to discover knowledge from distributed databases. In: Procs. AAAI WOrkshop on Learning Statistical Models from Relational Data, Morgan Kaufmann (2000) 52 – 58
4. Lang, S., Kragler, P., Haybach, G., Fahrmeir, L.: Bayesian space-time analysis of health insurance data. In Schwaiger, M., O.Opitz, eds.: Exploratory Data Analysis in Empirical Research. Springer (2002)
5. Klugmann, S., Panjer, H., Wilmot, G.: Loss Models – Fram Doata to Decisions. Wiley (1998)
6. Staudt, M., Kietz, J.U., Reimer, U.: A data mining support environment and its application to insurance data. In: Procs. KDD. (1998) insurance mining.
7. Kietz, J.U., Vaduva, A., Zücker, R.: MiningMart: Metadata-driven preprocessing. In: Proceedings of the ECML/PKDD Workshop on Database Support for KDD. (2001)

[3] For more details you might visit *www.mmart.cs.uni-dortmund.de*. The described insurance case will also be published in terms of its meta-data. Of course, the business data are strictly confidential.

8. Agrawal, R., Psaila, G., Wimmers, E.L., Zaït, M.: Querying shapes of histories. In: Proceedings of 21st International Conference on Very Large Data Bases, Morgan Kaufmann (1995) 502–514
9. Baron, S., Spiliopoulou, M.: Monitoring change in mining results. In: Proceedings of the 3rd International Conference on Data Warehou-sing and Knowledge Discovery, Springer (2001) 51–60
10. Bettini, C., Jajodia, S., Wang, S.: Time Granularities in Databases, Data Mining, and Temporal Reasoning. Springer (2000)
11. Blockeel, H., Fürnkranz, J., Prskawetz, A., Billari, F.: Detecting temporal change in event sequences: An application to demographic data. In De Raedt, L., Siebes, A., eds.: Proceedings of the 5th European Conference on the Principles of Data Mining and Knowledge Discovery. Volume 2168 of Lecture Notes in Computer Science., Springer (2001) 29–41
12. Das, G., Lin, K.I., Mannila, H., Renganathan, G., Smyth, P.: Rule Discovery from Time Series. In Agrawal, R., Stolorz, P.E., Piatetsky-Shapiro, G., eds.: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, AAAI Press (1998) 16 – 22
13. Mannila, H., Toivonen, H., Verkamo, A.: Discovering frequent episode in sequences. In: Procs. of the 1st Int. Conf. on Knowledge Discovery in Databases and Data Mining, AAAI Press (1995)
14. Morik, K.: The representation race - preprocessing for handling time phenomena. In de Mántaras, R.L., Plaza, E., eds.: Proceedings of the European Conference on Machine Learning 2000 (ECML 2000). Volume 1810 of Lecture Notes in Artificial Intelligence., Berlin, Heidelberg, New York, Springer Verlag Berlin (2000)
15. Domeniconi, C., shing Perng, C., Vilalta, R., Ma, S.: A classification approach for prediction of target events in temporal sequences. In Elomaa, T., Mannoila, H., Toivonen, H., eds.: Principles of Data Mining and Knowledge Discovery. Lecture Notes in Artificial Intelligence, Springer (2002)
16. Salton, G., Buckley, C.: Term weighting approaches in automatic text retrieval. Information Processing and Management **24** (1988) 513–523
17. Kietz, J.U., Vaduva, A., Zücker, R.: Mining Mart: Combining Case-Based-Reasoning and Multi-Strategy Learning into a Framework to reuse KDD-Application. In Michalki, R., Brazdil, P., eds.: Proceedings of the fifth International Workshop on Multistrategy Learning (MSL2000), Guimares, Portugal (2000)
18. Bi, Z., Faloutsos, C., Korn, F.: The DGX distribution for mining massive, skewed data. In: 7th International ACM SIGKDD Conference on Knowledge Discovery and Data Mining, ACM (2001)
19. Witten, I., Frank, E.: Data Mining – Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann (2000)
20. Rüping, S.: mySVM-Manual. Universität Dortmund, Lehrstuhl Informatik VIII. (2000) http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/.
21. Bauschulte, F., Beckmann, I., Haustein, S., Hueppe, C., El Jerroudi, Z., Koepcke, H., Look, P., Morik, K., Shulimovich, B., Unterstein, K., Wiese, D.: PG-402 Endbericht Wissensmanagement. Technical report, Fachbereich Informatik, Universität Dortmund (2002)
22. Allen, J.F.: Towards a general theory of action and time. Artificial Intelligence **23** (1984) 123–154
23. Höppner, F.: Discovery of Core Episodes from Sequences. In Hand, D.J., Adams, N.M., Bolton, R.J., eds.: Pattern Detection and Discovery. Volume 2447 of Lecture notes in computer science., London, UK, ESF Exploratory Workshop, Springer (2002) 1–12

24. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the ACM SIGMOD Conference on Management of Data, Washington, D. C. (1993) 207–216
25. Fisseler, J.: Anwendung eines Data Mining-Verfahrens auf Versicherungsdaten. Master's thesis, Fachbereich Informatik, Universität Dortmund (2003)
26. Mandelbrot, B.: A note on a class of skew distribution functions: Analysis and critique of a paper by H.A.Simon. Informationi and Control **2** (1959) 90 – 99
27. Box, G.E.P., Jenkins, G.M., Reinsel, G.C.: Time Series Analysis. Forecasting and Control. Third edn. Prentice Hall, Englewood Cliffs (1994)
28. Schlittgen, R., Streitberg, B.H.J.: Zeitreihenanalyse. 9. edn. Oldenburg (2001)
29. Keogh, E., Pazzani, M.: Scaling up dynamic time warping for datamining applications. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press (2000) 285–289
30. Agrawal, R., Faloutsos, C., Swami, A.: Efficient similarity search in sequence databases. In: Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms. Volume 730., Springer (1993) 69–84
31. Oates, T., Firoiu, L., Cohen, P.R.: Using dynamic time warping to bootstrap hmm-based clustering of time series. In: Sequence Learning ? Paradigms, Algorithms, and Applications. Volume 1828 of Lecture Notes in Computer Science. Springer Verlag (2001) 35?–52
32. Geurts, P.: Pattern extraction for time series classification. In: Pro-ceedings of the 5th European Conference on the Principles of Data Mining and Knowledge Discovery. Volume 2168 of Lecture Notes in Computer Science., Springer (2001) 115–127
33. Lausen, G., Savnik, I., Dougarjapov, A.: Msts: A system for mining sets of time series. In: Proceedings of the 4th European Conference on the Principles of Data Mining and Knowledge Discovery. Volume 1910 of Lecture Notes in Computer Science., Springer Verlag (2000) 289–298
34. Guralnik, V., Srivastava, J.: Event detection from time series data. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, San Diego, USA (1999) 33 – 42
35. Morik, K., Wessel, S.: Incremental signal to symbol processing. In Morik, K., Kaiser, M., Klingspor, V., eds.: Making Robots Smarter – Combining Sensing and Action through Robot Learning. Kluwer Academic Publ. (1999) 185 –198
36. Mannila, H., Toivonen, H., Verkamo, A.: Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery **1** (1997) 259–290
37. Klingspor, V., Morik, K.: Learning understandable concepts for robot navigation. In Morik, K., Klingspor, V., Kaiser, M., eds.: Making Robots Smarter – Combining Sensing and Action through Robot Learning. Kluwer (1999)
38. Rieger, A.D.: Program Optimization for Temporal Reasoning within a Logic Programming Framework. PhD thesis, Universität Dortmund, Dortmund, Germany (1998)
39. Morik, K., Scholz, M.: The MiningMart Approach to Knowledge Discovery in Databases. In Zhong, N., Liu, J., eds.: Intelligent Technologies for Information Analysis. Springer (2003) to appear.