2008

An Empirical Study of the Effects of Principal Component Analysis on Symbolic Classifiers

Huimin Zhao
University of Wisconsin - Milwaukee, hzhao@uwm.edu

Atish P. Sinha
University of Wisconsin - Milwaukee, sinha@uwm.edu

Sudha Ram
University of Arizona, ram@eller.arizona.edu

An Empirical Study of the Effects of Principal Component Analysis on Symbolic Classifiers

Huimin Zhao
University of Wisconsin-Milwaukee
hzhao@uwm.edu

Atish P. Sinha
University of Wisconsin-Milwaukee
sinha@uwm.edu

Sudha Ram
University of Arizona
ram@eller.arizona.edu

ABSTRACT

Classification is a frequently encountered data mining problem. While symbolic classifiers have high comprehensibility, their language bias may hamper their classification performance. Incorporating new features constructed based on the original features may relax such language bias and lead to performance improvement. Among others, principal component analysis (PCA) has been proposed as a possible method for enhancing the performance of decision trees. However, since PCA is an unsupervised method, the principal components may not represent the ideal projection directions for optimizing the classification performance. Thus, we expect PCA to have varying effects; it may improve classification performance if the projections enhance class differences, but may degrade performance otherwise. We also posit that the effects of PCA are similar on symbolic classifiers, including decision rules, decision trees, and decision tables. In this paper, we empirically evaluate the effects of PCA on symbolic classifiers and discuss the findings.

Keywords

Data mining, classification, principal component analysis, symbolic classifier, decision rule, decision tree, decision table.

INTRODUCTION

Classification is a type of supervised learning problem where a discrete dependent variable (referred to as class) needs to be predicted based on several independent variables (referred to as features). A classification algorithm induces a hypothetical classification model (referred to as classifier) based on a training dataset consisting of previously solved problem cases, whose class memberships are already known; the learned classifier can then be used to predict the class memberships of future unsolved problem cases. Symbolic classifiers learned by machine learning techniques, such as decision rules, decision tables, and decision trees, consist of logical combinations of tests involving individual decision variables (features). A major advantage of such symbolic classifiers is that they closely resemble human reasoning and can be easily understood by humans (Weiss and Kulikowski, 1991).

While the simplicity in the representation of symbolic classifiers is advantageous for human comprehension, it incurs constraints (referred to as language bias (Witten and Frank, 2005)) that potentially limit the ability of the classifiers to fit the training data. Only some of the features are utilized by the classifiers and the rest are simply ignored. In addition, the selected features are tested one at a time and not simultaneously. For example, the usual decision trees that test a single feature at each intermediate node are called univariate, orthogonal trees; the decision boundaries formed by such decision trees are restricted to be orthogonal to the testing features' axes in the feature space. Relaxing the language bias by allowing multivariate tests increases a classifier's ability to fit the training data and potentially improves classification performance. One way of achieving this, referred to as cascade generalization (Gama and Brazdil, 2000; Zhao and Ram, 2004), is to cascade other methods that construct new features based on the original features with basic symbolic classifier learners. Decision trees learned using this generalization method are called multivariate trees (Brodley and Utgoff, 1995) and oblique trees (Murthy, Kasif and Salzberg, 1994), because the intermediate nodes may involve multivariate tests and the decision boundaries may be oblique in the feature space. It has been shown that such multivariate, oblique decision trees are often more accurate than univariate, orthogonal trees (Brodley and Utgoff, 1995; Gama and Brazdil, 2000; Zhao and Ram, 2004).

Principal component analysis (PCA) extracts a sequence of orthogonal linear transformations (referred to as principal components) of the original features (Jolliffe, 2002). Among others, PCA has been proposed as a possible method for learning multivariate, oblique decision trees; it has been shown that adding principal components as new additional features into the training dataset can potentially reduce classification error rate (Popelínský and Brazdil, 2000a, 2000b; Popelínský 2001). At the same time, since each of the principal components accounts for decreasing amount of the variations in the original dataset and the first few components usually capture most of the information in the data, PCA has also been frequently used as a feature selection and dimensionality reduction technique prior to classifier learning (Fodor, 2002; Hall and Holmes, 2003). In such cases, the original features are simply replaced by the first few principal components.

However, we should be aware that PCA is an unsupervised method and does not take the dependent variable into consideration. Consequently, the resulting principal components may not represent the ideal projection directions in the feature space for optimizing the classification performance on the dependent variable; they may not enhance and may even hide class differences (Hand, Mannila and Smyth, 2001). For example, applications of PCA in face recognition have shown that PCA tends to retain unwanted variations due to lighting and facial expression, as well as variations due to change in face identity, and thus the resulting principal components may not be optimal for the discrimination purpose (Belhumeur, Hespanha and Kriegman, 1997). Therefore, we expect PCA to have varying effects on learned classifiers; it may improve classification performance for some problems if the projections enhance class differences, but may also degrade performance for some other problems if the projections hide class differences. We also posit that the effects of PCA are similar on symbolic classifiers, including decision rules, decision trees, and decision tables. Thus, findings on decision trees (e.g., Popelínský and Brazdil, 2000a, 2000b; Popelínský, 2001) can be generalized to other symbolic classifiers.

In this paper, we empirically evaluate the effects of PCA on decision rules, decision trees, and decision tables using ten classification problems available in the machine learning repository maintained by the University of California, Irvine (UCI) (Blake and Merz, 1998). We compare the classification accuracy of classifiers built using: 1) the original features, 2) the extracted principal components, and 3) both the original features and the principal components. We discuss the implications of the results and provide possible explanations.

BACKGROUND

A classification problem is described by a feature space, $X = X_1 \times X_2 \times \cdots \times X_m$, and a discrete class space, $Y = \{1,2,\ldots, p\}$. A classification algorithm is used to learn a classifier, which is a mapping $f : X \to Y$, by evaluating a set (referred to as training dataset) of solved cases, each of which is a pair $< x, y >$, where $x =< x_1, x_2,..., x_m >\in X$ and $y \in Y$. The classifier f can then be used to predict the value of the class of a new case, given its values of the features. The performance of the classifier can be measured by accuracy (or inversely, error rate), defined as the probability of making a correct prediction when given a new case, Prob[$f(x) = y$].

Machine learning techniques learn symbolic classifiers, such as decision rules, decision trees, and decision tables. A set of decision rules is typically a disjunction of conjunctive conditions, each of which involves a single feature x_i ($i = 1,2,\ldots, m$). A decision rule learner usually follows a covering approach and repeatedly identifies rules that cover some of the training cases in a particular class until all of the training cases have been covered (Witten and Frank, 2005). One of the efficient and effective decision rule learners is Ripper (Cohen, 1995), which enhances an earlier algorithm named IREP (Furnkranz and Widmer, 1994).

Decision trees make the classification decision via a sequence of small tests, each of which involves a single feature x_i ($i = 1,2,\ldots, m$). A decision tree learner follows a divide-and-conquer approach and recursively partitions the training sample based on selected features. The result is a tree-structured decision model; the intermediate tree nodes are tests and the leaves are associated with the predicted classes. The testing features at the intermediate tree nodes are selected based on some heuristic goodness measures. For example, the C4.5 algorithm (Quinlan, 1993) uses information gain and gain ratio as the selection criteria. A decision tree can be translated into a set of unambiguous decision rules, although such rules are usually far more complex than those directly produced by decision rule learners, such as Ripper.

A decision table learner selects several most discriminating features based on the training dataset to form a lookup table, which is then used to classify new cases. Different subsets of features are evaluated using a performance estimation method, such as cross-validation (Kohavi, 1995a), and the best-performing subset is kept in the final decision table. Efficient heuristic algorithms (e.g., Kohavi, 1995b) exist for finding approximately optimal subsets of features.

Prior to classifier learning, several preprocessing steps, such as feature transformation, feature selection, and clustering, may be taken to potentially improve classification performance and efficiency (Hand, Mannila and Smyth 2001; Witten and Frank, 2005). New, potentially useful features may be defined by transforming the existing features. A subset of features may be selected according to some goodness measures to reduce the dimensionality of the feature space. Clustering may be used to identify previously unknown clusters (with cases within each cluster being similar to other cases in the same cluster but different from cases of other clusters) and define new features based on the memberships of the cases in the clusters.

PCA is frequently used as a preprocessing step prior to other analyses. It derives a sequence of orthogonal linear combinations of the original features, $x'_j = \sum_{i=1}^{m} w_{ij} x_i$, referred to as principal components, which are eigenvectors of the covariance matrix of the original features. The components are selected such that the first component has maximum sample variance and each subsequent component has maximum sample variance subject to being uncorrelated with the previous components. The first few components usually account for most of the variance in the data, so that other components can be discarded without incurring a considerable loss of information. We chose to empirically study the varying effects of PCA across classification problems, as we had observed the lack of such an informative and much needed study in the literature.

EMPIRICAL STUDY

We have evaluated the effects of PCA on symbolic classifiers built for ten classification problems (with numeric features) available in the UCI machine learning repository (Blake and Merz, 1998). For each problem, we extracted principal components using one of two commonly adopted criteria: the eigenvalues of the selected components are over one or the cumulative variations explained by the selected components are about 75%. Table 1 summarizes the characteristics of these problems. The two criteria resulted in identical or similar results, except for the Wave dataset, where the components with eigenvalues over 1 capture only about 56.9% of the variations in the original data. Henceforth we will only report the results of the eigenvalues-over-one criterion. For each problem, we constructed three datasets: one with the original features, one with the principal components, and the other combining both the original features and the principal components. We will henceforth refer to these datasets as Original, Components, and Both.

Name	Description	Classes	Instances	Features	Components	
					A	B
Bala	Balance Scale	3	625	4	3	3
Glas	Glass Identification	7	214	9	4	4
Imag	Image Segmentation	7	2,310	19	4	4
Ioso	Ionosphere	2	351	34	8	10
Iris	Iris Plant	3	150	4	1	1
Pima	Pima Indians Diabetes	2	768	8	3	4
Sona	Sonar	2	208	60	13	11
Vehi	Statlog Project: Vehicle Silhouettes	4	946	18	4	3
Wave	Waveform: 5000	3	5,000	40	12	21
Wisc	Wisconsin Breast Cancer	2	699	10	1	2

Table 1. Characteristics of Ten UCI Classification Problems
(A: Eigenvalue > 1; B: Cumulative variations explained about 75%.)

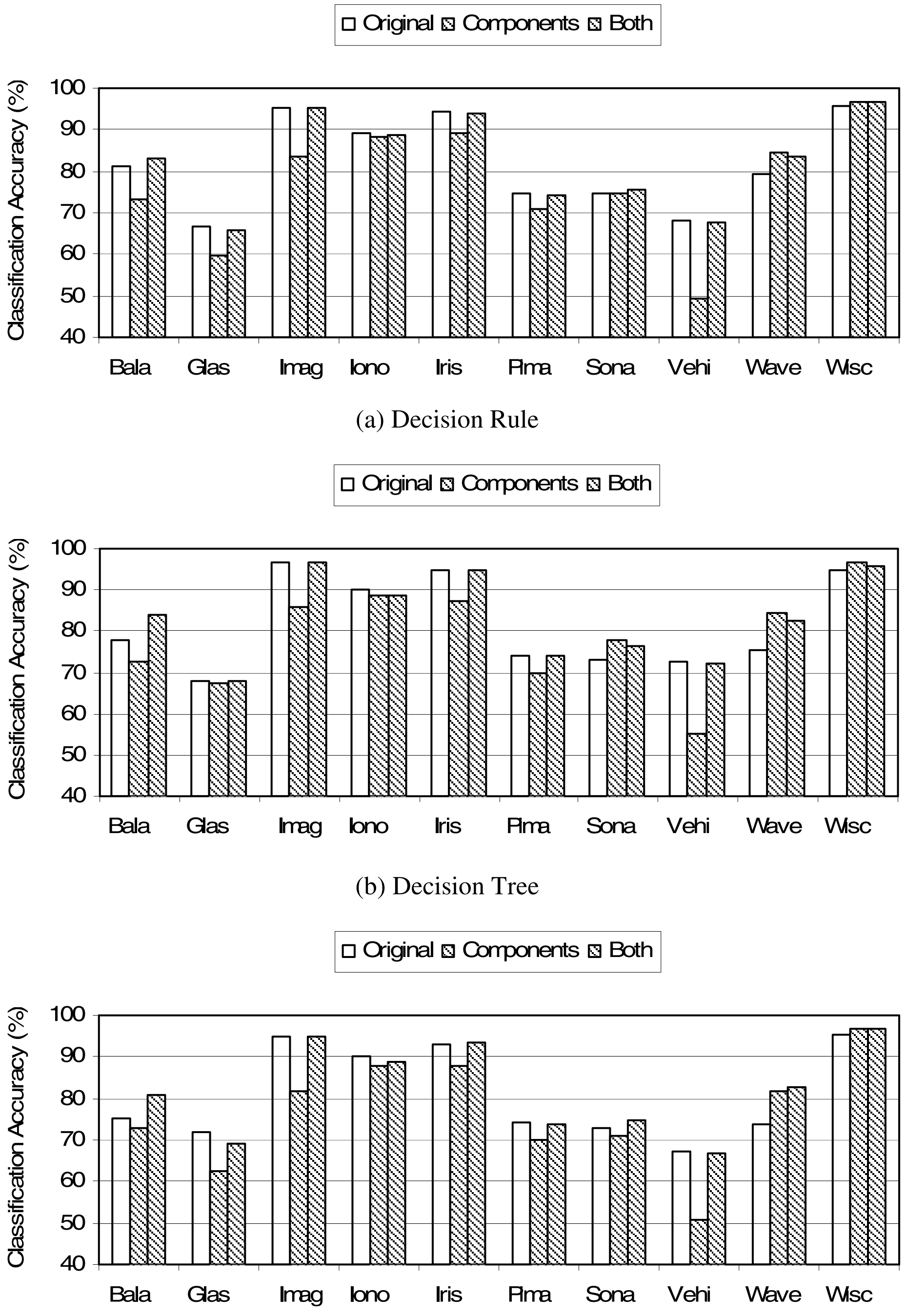
We used the Weka machine learning toolkit (Witten and Frank, 2005) for the evaluation purpose. We selected the following representative symbolic classifier learners: the Ripper decision rule learner (Cohen, 1995) (named JRip in Weka), the C4.5 decision tree learner (Quinlan, 1993) (named J4.8 in Weka), and a decision table learner (Kohavi, 1995b). We will henceforth refer to these algorithms as Rule, Tree, and Table. We retained Weka's default values for the parameters of these algorithms.

We used classification accuracy as the performance measure and evaluated the effects of PCA on symbolic classifiers in terms of performance improvement or degradation. We used 10-fold cross-validation (Kohavi, 1995a) to estimate the accuracy of each learned classifier. Cross-validation randomly divides a training dataset into approximately equal-sized, stratified sub sets, called folds, and repeatedly uses each fold for performance testing while the other folds are used for training a classifier. The average of the testing performance measured over the runs is then used as an overall estimate of the performance of the classifier learned using the entire training dataset. Empirical evaluation has shown that 10-fold cross-validation usually results in reasonably accurate estimates (Kohavi, 1995a). In addition, we repeated 10-fold cross-validation 100 times for each combination of dataset and classification method to get more reliable estimates.

Table 2 summarizes the evaluation results. Figure 1 contrasts the performance of the classifiers learned using the Original, Components, or Both datasets. We can make several observations from the results. Adding principal components into the original features often performs better than using the components only (except for the Wave and Wisc problems). This implies that the extra training time incurred by adding the additional features may be justified when performance is a more important consideration than training efficiency.

Problem	Dataset	Rule		Tree		Table	
		Mean	StdDev	Mean	StdDev	Mean	StdDev
Bala	Original	81.11	1.11	77.93	0.87	75.10	0.91
	Components	73.48 ●	1.22	72.66 ●	1.06	72.62 ●	0.88
	Both	83.31 ○	1.05	84.13 ○	0.96	80.88 ○	0.87
Glas	Original	66.87	2.23	67.88	2.15	71.96	1.52
	Components	59.47 ●	2.22	67.21 ●	1.73	62.29 ●	1.30
	Both	65.88 ●	2.36	67.94	2.28	69.02 ●	2.19
Imag	Original	95.41	0.56	96.86	0.29	95.00	0.26
	Components	83.41 ●	0.61	85.96 ●	0.50	81.78 ●	0.43
	Both	95.11 ●	0.64	96.89	0.25	95.00	0.26
Iono	Original	89.21	0.95	89.94	1.15	90.03	0.96
	Components	88.06 ●	1.12	88.64 ●	1.11	87.67 ●	0.91
	Both	88.97	1.14	88.83 ●	1.18	88.67 ●	1.09
Iris	Original	94.49	1.38	94.91	0.78	93.00	0.87
	Components	88.99 ●	1.53	87.48 ●	1.23	87.73 ●	1.26
	Both	93.83 ●	1.69	94.93	0.79	93.27	0.92
Pima	Original	74.89	0.88	74.22	1.04	74.15	0.97
	Components	71.00 ●	1.01	69.68 ●	0.87	70.12 ●	0.64
	Both	74.31 ●	0.91	73.86 ●	1.03	73.67 ●	1.09
Sona	Original	74.89	2.35	73.04	2.22	73.03	2.56
	Components	74.90	2.29	77.64 ○	2.03	71.16 ●	1.38
	Both	75.75 ○	2.83	76.34 ○	2.17	74.90 ○	2.40
Vehi	Original	68.26	1.37	72.57	1.17	67.03	1.19
	Components	49.16 ●	1.54	55.29 ●	1.28	50.70 ●	0.83
	Both	67.49 ●	1.25	72.15 ●	1.09	66.91	1.12
Wave	Original	79.26	0.40	75.26	0.51	73.68	0.46
	Components	84.45 ○	0.29	84.26 ○	0.23	81.89 ○	0.20
	Both	83.53 ○	0.40	82.35 ○	0.42	82.52 ○	0.33
Wisc	Original	95.55	0.44	94.69	0.49	95.18	0.48
	Components	96.88 ○	0.20	96.68 ○	0.13	96.65 ○	0.17
	Both	96.63 ○	0.27	95.92 ○	0.32	96.62 ○	0.25

Table 2. Cross-validated Classification Accuracy (○ (●) indicates statistically significant improvement (degradation) compared with using original features, based on the *t* test, α=0.05.)



(a) Decision Rule

(b) Decision Tree

(c) Decision Table

Figure 1. Contrasts of Classification Accuracy across Different Schemes

The effects of PCA vary across classification problems. Adding principal components to the original features (Both dataset) or replacing the original features with the components (Components dataset) may improve or degrade classification accuracy, depending on the problem. The Components dataset often (for 80% of the problems on Table, 70% of the problems on Rule and Tree) results in lower accuracy compared with the Original dataset. This implies that when used as a dimensionality reduction technique, PCA often degrades the classification performance. Even the Both dataset sometimes (for 30% of the problems on Tree and Table, 50% of the problems on Rule) degrades classification performance. This implies that when used in cascade generalization for performance improvement, PCA is less promising than a supervised learning technique, such as logistic regression (Hosmer and Lemeshow, 2000), which has been shown to usually improve and seldom degrade classification performance (Brodley and Utgoff, 1995; Gama and Brazdil, 2000; Zhao and Ram, 2004). A possible reason is that principal components are derived without considering the class at all and therefore may not enhance and may even hide class differences, while the linear functions induced by logistic regression are explicitly sought to maximize class differences.

On the other hand, the Both dataset does sometimes (for 40% of the problems on all three classifiers) result in performance improvement. Even the Components dataset sometimes (for 20% of the problems on Rule and Table, 30% of the problems on Tree) improves classification accuracy. One possible reason is that the first few principal components contain most of the information of the data. Since symbolic classifiers only retain a few features, when they are built upon the original features, the information of the unselected features is not utilized. On the other hand, when the first few principal components are used, they capture more information than the individual original features and may improve the classification performance of symbolic classifiers if they do not hide class differences.

The effects of PCA are similar on rule, tree, and table. If adding principal components improves (or degrades) one symbolic classifier, it is likely to improve (or degrade) others too. The only statistically significant contradiction is on the Sona dataset, where using the Components dataset results in performance improvement on Tree but degradation on Table.

CONCLUSION AND FUTURE RESEARCH DIRECTIONS

Our empirical results show that PCA has varying effects on symbolic classifiers, depending on the classification problem. One should, therefore, be aware of the potential consequences (besides the often expected benefits) when applying PCA with symbolic classifiers. When used as a dimensionality reduction technique, PCA often degrades classification performance. The savings in training time should justify the potential performance loss. Even when used in addition to the original features, principal components may improve or degrade performance. For the purpose of performance improvement through cascade generalization, a supervised method like logistic regression has higher potential than an unsupervised method like PCA, although the latter is much more efficient.

One should also be aware that PCA may reduce the comprehensibility of the learned symbolic classifiers. The principal components are harder to interpret than the original features, which usually have direct operational meanings. A large reason for the popularity of symbolic classifiers is that they resemble human reasoning and are easily understandable by humans. In situations where comprehensibility is weighted higher than accuracy, symbolic classifiers learned using just the original features may well be preferred to classifiers incorporating principal components, even if the latter are more accurate.

Our empirical study can be replicated in other settings. While we used classification problems with only numerical features, future studies may use problems with both numerical and nominal features. In that case, nominal features need to be transformed into dummy variables first. Future studies may also evaluate the effects of other dimensionality reduction methods, such as factor analysis, projection pursuit, and independent component analysis (Hand, Mannila and Smyth 2001; Kwak and Choi, 2003). As these methods are all unsupervised, we expect that, similar to PCA, they have varying effects on symbolic classifiers. While we have shown that PCA has varying effects on symbolic classifiers, future studies can further investigate when PCA improves or degrades classification performance. Another future research direction is to identify what types of data and what types of learning techniques would substantially benefit from PCA and other dimensionality reduction methods.

REFERENCES

1. Belhumeur, P. N., Hespanha, J. P. and Kriegman, D. J. (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 7, 711-720.
2. Blake, C.L. and Merz, C. J. (1998) UCI repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html.
3. Brodley, C. E. and Utgoff, P. E. (1995) Multivariate decision trees, *Machine Learning*, 19, 1, 45-77.
4. Cohen, W. W. (1995) Fast effective rule induction, in A. Prieditis and S. Russell (Eds.) *Proceedings of the Twelfth International Conference on Machine Learning*, Lake Tahoe, CA, Morgan Kaufmann, 115-123.
5. Fodor, I. K. (2002) A survey of dimension reduction techniques, LLNL Technical Report, UCRL-ID-148494.
6. Furnkranz, J. and Widmer, G. (1994) Incremental reduced error pruning, in *Proceedings of the Eleventh International Conference on Machine Learning*, 70-77.
7. Gama, J. and Brazdil, P. (2000) Cascade generalization, *Machine Learning*, 41, 3, 315-343.

8. Hall, M. A. and Holmes, G. (2003) Benchmarking attribute selection techniques for discrete class data mining, *IEEE Transactions on Knowledge and Data Engineering*, 15, 6, 1437-1447.
9. Hand, D., Mannila, H. and Smyth, P. (2001) Principals of data mining, MIT Press.
10. Hosmer, D. W. and Lemeshow, S. (2000) Applied logistic regression, second edition, John Wiley & Sons, Inc.
11. Jolliffe, I. T. (2002) Principal component analysis, 2nd Edition, Springer.
12. Kohavi, R. (1995a) A study of cross-validation and bootstrap for accuracy estimation and model selection, in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1137-1143.
13. Kohavi, R. (1995b) The power of decision tables, in N. Lavrac and S. Wrobel (Eds.) *Proceedings of the Eighth European Conference on Machine Learning*, Berlin, Germany, Springer-Verlag, 174-189.
14. Kwak, N. and Choi, C. (2003) Feature extraction based on ICA for binary classification problems, *IEEE Transactions on Knowledge and Data Engineering*, 15, 6, 1374-1388.
15. Murthy, S. K., Kasif, S. and Salzberg, S. (1994) A system for induction of oblique decision trees, *Journal of Artificial Intelligence Research*, 2, 1-32.
16. Popelínský, L. (2001) Combining the principal components method with different learning algorithms, in *Proceedings of the ECML/PKDD2001 IDDM Workshop*.
17. Popelínský, L. and Brazdil, P. (2000a) Combining the principal components method with decision tree learning, in *Proceedings of the 2000 Multistrategy Learning Workshop*, Guimares, Portugal.
18. Popelínský, L. and Brazdil, P. (2000b) The principal components method as a pre-processing stage for decision tree learning, in *Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases*, Lyon, France.
19. Quinlan, J. R. (1993) C4.5: Programs for machine learning. Morgan Kaufmann.
20. Weiss, S. M. and Kulikowski, C. A. (1991) Computer systems that learn - classification and prediction methods from statistics, neural nets, machine learning, and expert system, Morgan Kaufmann.
21. Witten, I. H. and Frank, E. (2005) Data mining: practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann.
22. Zhao, H. and Ram, S. (2004) Constrained cascade generalization of decision trees, *IEEE Transactions on Knowledge and Data Engineering*, 16, 6, 727-739.