TECHNOLOGICAL AND ECONOMIC DEVELOPMENT OF ECONOMY

2010
16(4): 765–781

CLUSTERING-BASED DECISION TREE CLASSIFIER CONSTRUCTION

Inese Polaka[1], Arkady Borisov[2]

Riga Technical University, 1 Kalku Street, LV-1658 Riga, Latvia
E-mails: [1]Inese.Polaka@rtu.lv; [2]Arkadijs.Borisovs@cs.rtu.lv

Received 24 March 2010; accepted 20 October 2010

Abstract. This article studies data structure investigation possibilities using cluster analysis. Density structures within classes are explored to implement class decomposition in order to enhance performance of decision tree classifiers. Classes are decomposed using cluster analysis and cluster merge evaluation using decision tree classifiers. Then impact of class decomposition is shown on C4.5 and CART classifiers. The main focus is on experiments carried out with real-valued data sets. The experiments are described in a step-by-step manner to illustrate the patterns discovered which affect previously proposed patterns in class decomposition methodology.

Keywords: classification, class decomposition, cluster analysis, decision trees, data mining.

Reference to this paper should be made as follows: Polaka, I.; Borisov, A. 2010. Clustering-based decision tree classifier construction, *Technological and Economic Development of Economy* 16(4): 765–781.

1. Introduction

The task described in this article is a classification task; it deals with objects allocation to predefined classes using a classification model. Classifiers can be either linear (e.g., Naïve Bayes classifier) or non-linear (e.g., decision trees). A classification model is considered linear if it divides the whole attribute space into classes using hyperplanes (each class is separated by a hyperplane). This also causes a problem if classes cannot be separated using hyperplanes (see Fig. 1 for an example) which causes increased error and complexity of a classification model. In these cases more sophisticated and non-linear models are used but they are more resource-consuming and therefore it is important to broaden the possibilities of using linear classifiers by improving their efficiency on non-linear cases. Decision trees divide attribute space iteratively using hyperplanes that are orthogonal to one of axis executing approximation of non-linear dividing lines. Taking into account the similarity between decision tree construction and linear methods we can assume that methods that can improve performance of linear classifiers can

ISSN 2029-4913 print/ISSN 2029-4921 online doi: 10.3846/tede.2010.47
http://www.tede.vgtu.lt

also be applied to decision tree construction. This work discusses decision tree algorithms C4.5 and CART as well as cluster analysis for class decomposition in order to improve the performance of the classifiers.

Decision trees were proposed by J. Ross Quinlan in (Quinlan 1986) describing algorithm ID3 that was used as a basis for other decision tree classifiers that were created changing evaluation functions and construction parameters. Algorithm C4.5 was proposed in (Quinlan 1993) and CART algorithm was presented in (Breiman *et al.* 1984). Both algorithms divide attribute space in a similar manner but they differ in tree structure, split criteria and pruning method. To improve these decision tree classifiers, we research the initial data structure of a training data set in order to gain information for constructions of more efficient decision trees. It is done by class decomposition (Vilalta *et al.* 2003) i.e., dividing initial classes into smaller areas according to some similarity.

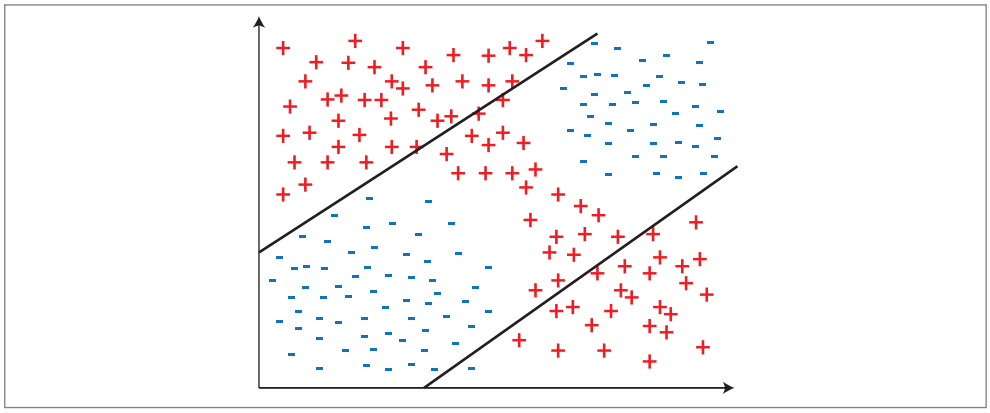

Fig. 1. Groups of objects cannot be separated using straight lines (hyperplanes in a two-dimensional space) that represent linear classifier

To explore the structure of a data set, we use hierarchical agglomerative classification that gives information about separate areas of high density of records that are represented by clusters. The Ward method is preferred in these experiments because it produces compact clusters that is of significance in class decomposition method. The distance between clusters shows how well these groups can be divided. Two groups that have a large distance between them can be separated using hyperplanes created by decision tree classifiers and easily separable clusters have a smaller error.

Information about data structure – the character of clusters – is used in class decomposition by replacing the initial classes by cluster labels that can be mapped to the original class after classification.

But a larger amount of classes often means an increased mistake. For example, the point labeled 'x' in Figure 2 can be assigned label A1, A2 (class A) or B2 (class B) which would be wrong. If classes A1 and A2 are merged, it becomes obvious that this point belongs to class A.

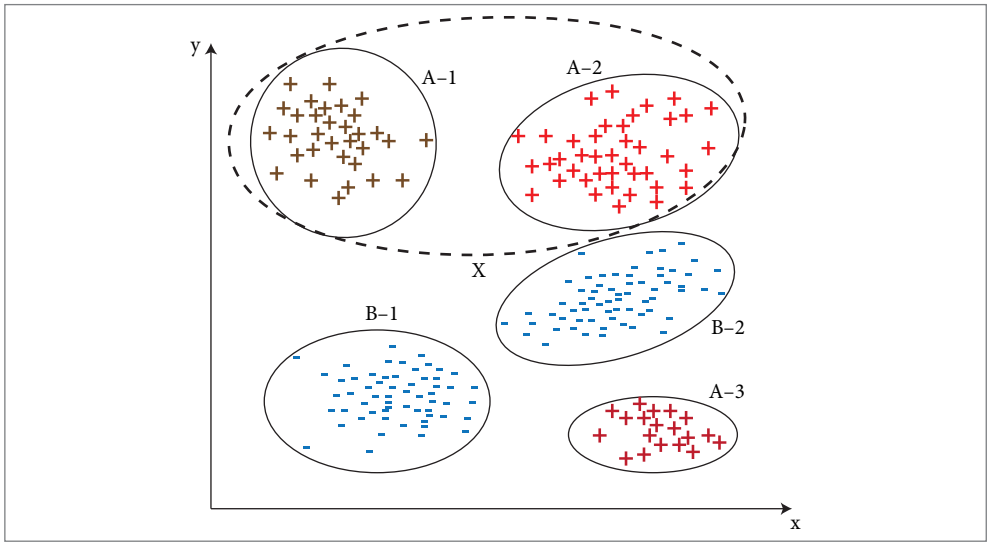

Fig. 2. In the case of a complex class structure combinations of clusters should be considered because a change in class structure changes the classification outcome

Due to these differences, the impact of different merges should be researched but it is a very resource-consuming process because the number of possible combinations grows exponentially with each extra cluster. And clusters that are easily separable within one class are not necessarily the best choice for class decomposition because the performance differs depending on the character of overlapping areas. Vilalta *et al.* (2003) use a heuristic method that is based on assumption that combinations with higher cardinality merges perform better in class decomposition; but this also depends on the structure of classes.

2. Problem statement

While working with decision trees and algorithm C4.5 in particular, we came across the problem of significantly varying performance on various data sets. In the decision tree studies the C4.5 algorithm was applied to data sets with different data types and structures and got apparently different results even when data and tasks seemed similar. To find the cause of these differences, more should be learned about the behavior of decision trees and its dependency on class distribution in attribute space structure. Then performance of the algorithm could be improved by designing decision trees using meta-data acquired while investigating the attribute space.

Investigation of data structure in this case will include clustering of each class of data to obtain information about the structure of areas with high data density for finding an optimal number of clusters within each class. These clusters will then be merged to achieve better division of attribute space into classes by decision tree classifiers. This step is time- and resource consuming; therefore we need to find a trade-off between the number of clusters and achieved improvement in classifier performance.

3. Related work

The task is to analyze and describe the structure of input data sets in order to improve efficiency of classifier training.

The idea of transition from a global research of the input data structures to local research of data is outlined in (Fulton *et al.* 1996). Authors describe the so-called local approach to decision tree design. According to this approach, first a subset of the available data is formed by selecting examples that are in the locality of a given test example that is being classified. Then a classifier is built using the obtained subset. The input data set is fragmented to some extent by this medium.

The previously described idea is similar to that behind the k-nearest neighbour method. Taking into consideration the fact that every test example requires re-defining of the subset of relevant examples used for classification this allows for a kind of interactive approach to classifier building.

While implementing this method, a question of an appropriate similarity measure arises because it has to describe the specific character of the problem domain as well as the unique character of the new test example. Implementation of the algorithm is associated with a large amount of computations although it may be allowed in some cases when accuracy of classification is crucial.

Jiang *et al.* (2007) proposed a statistical approach of data structure analysis. The authors used supervised density estimation techniques that not only consider the location of a data point but also a variable of interest associated with the data point to model the distribution of the data points. They measure density as the product of an influence function with the variable of interest and use it for clustering using Supervised Clustering Using Density Estimation algorithm but it can also be used in classification algorithms that rely on decision boundaries extracted from the supervised density functions.

Brazdil *et al.* (2003) presented a meta-learning method to support the selection of classification algorithm based on specific characteristics of the structure of a data set. Data structure is explored using the k-nearest neighbour algorithm that compares the similarity of a given data set with the characteristics of standard data sets that have been previously studied. The features of previously studied data sets are related to a ranged amount of classification algorithms. The choice of a certain classification algorithm is executed taking into consideration various parameters.

There is also an approach that is based on analysis of geometrical features of the training set using shadows of fuzzy sets (Ozols and Borisov 2001). This approach allows implementing the evaluation of the dependencies among attributes and constructing new efficient attributes that incorporate combinations of initial attributes based on these dependencies. In this adjustment the connection between the structure of a training set and the parameters of a classifier is explicit.

Vilalta (2006) proposed an approach oriented towards preliminary identification of local high density areas in a training set using cluster analysis. The clusters discovered in cluster analysis are viewed as separate classes that are classified using a simple linear classifier. Furthermore, the problem of variance of examples within a single class is overcome; however

another problem appears that is the increase in linear discriminants that ensures the growth of complexity (number of clusters) of the training set.

A similar clustering approach was also used in Michalopoulos *et al.* (1999) and Looney (2002). Michalopoulos *et al.* used a combination of fuzzy *c*-means clustering and top down induction of decision trees to produce accurate classification rules. They used clustering to learn the structure of data and used this obtained information in classification. Looney used an algorithm based on fuzzy *c*-means algorithm to study the structure of data. He modified the algorithm to form fewer yet more uniform clusters than *k*-means algorithm using prototypes that are placed in dense areas and merging similar clusters.

4. Proposed approach

In (Vilalta 2006) k-means algorithm is proposed for cluster analysis but this algorithm is inconsistent in its performance because of the random character of initially chosen centroids that sets in if the disposition of clusters is previously unknown. This choice has a great effect on the result of clustering and quality of its performance. In our approach we propose hierarchical agglomerative clustering for data structure analysis and algorithms C4.5 and CART as tools for decision tree classifier design.

In this work, we present evaluation of effectiveness of a global classifier, i.e. a classifier that classifies data without further analysis of the initial data set, and a classifier that implements classification based on research of the geometrical features of the initial attribute space.

This paper is based on experimental research of classification tasks since, in authors' opinion, experimental research that has been carried out previously is insufficient to succeed in defining relationships between qualities of a training set and effectiveness of classifiers. Only additional well-planned experiments can show the diversity of the links between characteristics of the original data set and the effectiveness of classifiers. For the present it has been impossible to find this kind of dependency in theory therefore we are to put trust in experiments.

5. Methods used

In the presented experiments we used cluster analysis and decision tree classifiers to explore structures of data. To test features found in data exploration, decision tree classifiers were used to classify data with and without class decomposition.

5.1. Decision trees

For classification we use decision trees (Quinlan 1993); they are constructed from data by dividing a training set into subsets until subsets can be assigned a class. Division is carried out by choosing an attribute and its value as threshold. Choice of the attribute can be carried out using different criteria. Criteria used in algorithm C4.5 is usually information gain or gain ratio. Information gain is the change in entropy of information if the state of information is changed. Let C be the class attribute with values $\{c_1, c_2, \ldots, c_n\}$ and A attribute with values

$\{a_1, a_2, \ldots, a_k\}$, $H(C)$ be the entropy of the attribute C, and $H(C|A)$ conditional entropy that shows entropy of C if state of attribute A is known; information gain is:

$$I(C, A) = H(C) - H(C|A). \quad (1)$$

The entropy of attribute C is:

$$H(C) = -\sum_{i=1}^{n} P(C = c_i)\log_2(P(C = c_i)), \quad (2)$$

where $P(C= c_n)$ is the relative frequency of class value c_n. And the conditional entropy is:

$$H(C|A) = -\sum_{j=1}^{k} P(A = a_j)H(C \mid A = a_j). \quad (3)$$

Information gain favors attribute with higher number of values. To avoid that, gain ratio can be used. This criterion penalize

$$\frac{I(C, A)}{H(A)}, \quad (4)$$

where the entropy of attribute A is calculated as follows:

$$H(A) = -\sum_{j=1}^{k} P(A = a_j)\log_2(P(A = a_j)). \quad (5)$$

Whereas CART usually uses Gini index as splitting criteria. Gini index is calculated as:

$$G(C) = 1 - \sum_{i=1}^{n} P(C = c_i). \quad (6)$$

CART and C4.5 have also other differences like pruning method, missing values handling and others (Kohavi, Quinlan 2002). Pruning examines and substitutes subtrees of the whole tree with a leaf or a branch of the subtree where necessary. Both methods use tree pruning to avoid overfitting because trees that execute full classification of the training set (every record is assigned to its correct class) tend to perform worse on unknown data but they differ in their approach to pruning. C4.5 uses reduced error pruning that analyzes if a subtree replacement with a leaf leads to less error. This technique requires a separate data set for pruning, which can be a drawback but it examines every subtree once and is much faster than other techniques (Quinlan 1987). CART uses minimal cost complexity pruning technique which assigns costs to subtrees based on the error from pruning and the size of the subtree (Kohavi, Quinlan 2002). This technique does not require a separate data set for pruning.

To improve error estimate that is directly affected by the availability of test data, both methods use cross-validation. Without cross-validation, error is usually estimated using holdout method which partitions data into two mutually exclusive sets – a training set for model building and a test set for error estimation via applying the model to unknown data (Kohavi 1995). In a k-fold cross-validation the initial data set is split into k subsets and the model is trained and tested k times. Each run is carried out using k-1 subsets as training data and one subset that is different for each run for training. Then the error is estimated as the average of errors of each run.

5.2. Cluster analysis

Clustering finds groups of similar data either by partitioning data into k subsets (partitioning methods) or creating a hierarchical decomposition of the data (hierarchical methods) or

building groups of data based on density, grid or models (Han, Kamber 2006). There are many similarity measures (Deza, E.; Deza, M. 2009) that can be used for that purpose, for example, the Euclidean distance that is used in our approach. The Euclidean distance between two points $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ in an n-dimensional space is calculated as follows:

$$d(a,b) = \sqrt{\sum_{i=1}^{n}(a_i^2 - b_i^2)}\,. \tag{7}$$

Hierarchical clustering creates a hierarchy of data records and subsets either by dividing the whole data set until a given size of subsets (divisive approach) or agglomerating records into subsets until all records belong to a given number of sets or one set (agglomerative approach). Division and merging is based on the distance between groups called linkage that can be calculated as the distance between closest or center points of groups as well as distance among all points of different groups (He 1999). In the presented experiments Ward's criterion (Ward 1963) that shows increase in variance if groups are merged is used for linkage.

6. Experiments

In these experiments we used three real-world data sets. With each data set we carried out the same procedure to illustrate its work with different data sets.

6.1. Data sets with real-valued attributes

The following data sets were used for experiments: 'Breast Cancer Wisconsin (Diagnostic)', 'Iris' and 'Parkinson's' from the University of California at Irvine Repository (Asuncion, Newman 2007).

6.2. Methodology

For all data sets the same steps for class decomposition are implemented and cross-validation is used for classification:

1. Let $x = \{x_1, x_2, \ldots, x_n\}$ be an n-dimensional vector describing an object in the training set. Let $\{ y_1, y_2, \ldots, y_k \}$ be the set of classes. The input set T is divided into subsets T_1, T_2, …, T_k where each $T_i = \{(x, y_i)\}$ holds all of the examples of the same class i from the initial set T. Hierarchical clustering is performed with each subset T_i.
2. Choosing the best number of clusters (by largest distance) and assigning the cluster label to each example x.
3. Merging clusters to find a combination that leads to the least classification error on clustered.
4. Evaluating combinations by assigning labels of merged clusters y_i' mapping the subset T_i to a set $T_i' = \{(x, y_i')\}$ and classifying complete training data set T. Every label y_i' contains information about existing class allowing to map the y_i' to the original class y_i.
5. Finding the best combination by analyzing the results of classification performed on the set $T' = \bigcup_{i=1}^{k} T_i'$.

6. Re-labeling classified examples to match the existing classes $\{y_1, y_2, \ldots, y_k\}$.

To implement clustering, we use Orange 2.0 and WEKA collection of machine-learning algorithms for classification.

6.3. Performance comparison on the Iris problem

The first data set discussed here is 'Iris' with five attributes – four real number attributes and one class attribute with three classes. This is a very popular data set with easily separable classes using two of the attributes (see Figs 3 and 4). Since the structure of the data is mainly known, it is challenging to see if performance of decision tree classifiers can be improved by using class decomposition and if it leads to using more than two attributes in the tree structure.

Figures 3 and 4 show the difference between algorithms C4.5 and CART in practice. Although both algorithms use the same strategy for building the model – splitting the data into groups using the most appropriate attribute and its value, they use different techniques to choose attributes and threshold values for splitting nodes. Therefore they divide the attribute space into different sub-spaces belonging to classesThe structure of separate classes can be analyzed using hierarchical clustering. Since hierarchical clustering does not need any prior knowledge about the number of clusters, which is important for the primary precondition of the research – the task with unknown data, it can be used to derive such information from data with previously unknown structure. A positive feature of using hierarchical clustering is visualization of the cluster hierarchy in a dendrogram. It also makes the choice of number of clusters easier because distances between clusters are shown as links between clusters and the point of their merge. Besides, clusters with the longest distance between them are easier

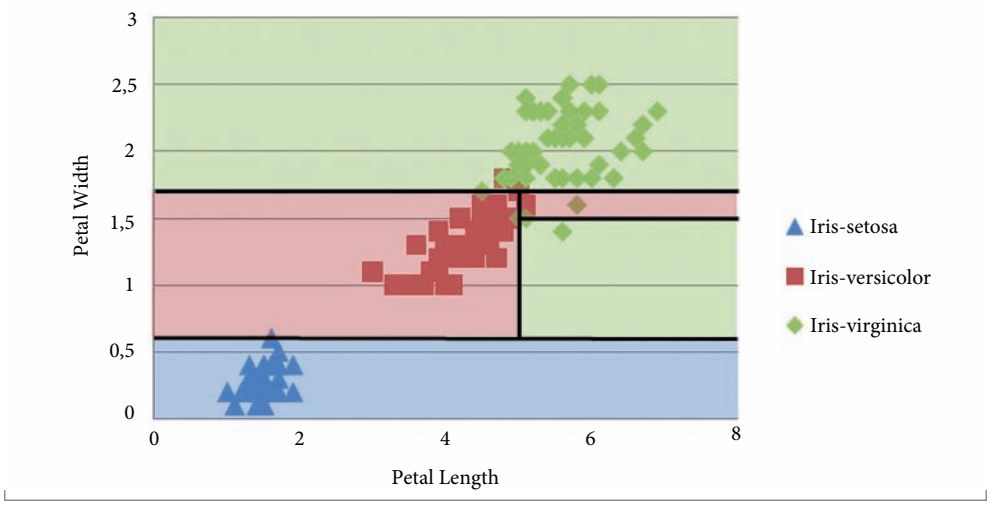

Fig. 3. Three classes of Iris data set classified using C4.5 – Iris-Setosa fully separated at the lower part and other two classes overlap a little causing error

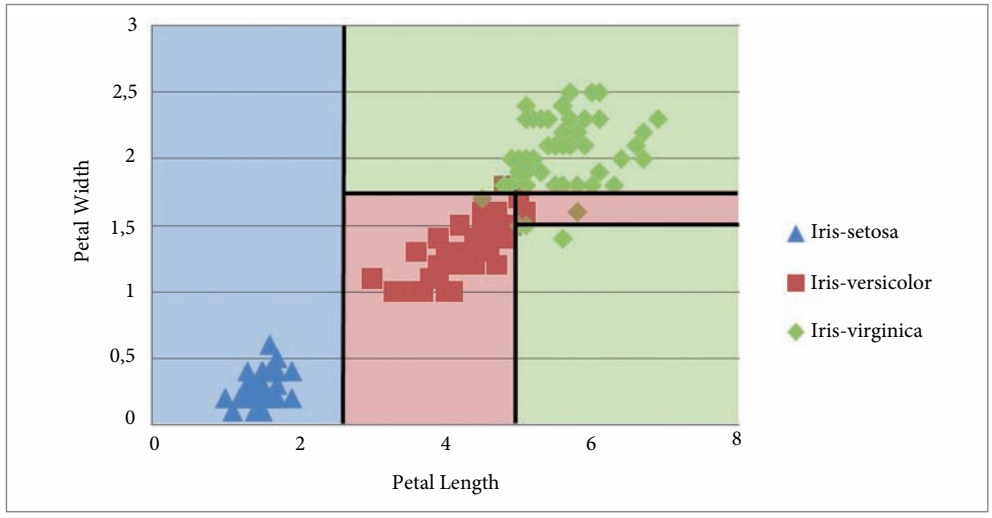

Fig. 4. Three classes of Iris data set classified using CART – Iris-Setosa fully separated at the left-hand part and other two classes overlap still causing error

to divide using hyperplanes that are orthogonal to attribute axes like it is done by most decision tree classifiers.

Optimal number of clusters for manual calculations is four (with 14 combinations of interest) because the number of combinations grows exponentially with each next cluster. To show the process of class decomposition, each class is divided into four clusters using hierarchical clustering. This allows showing the differences of merges with different cardinalities while still keeping the amount of needed calculations fairly low. Then these four clusters are merged into different combinations for each class (see Table 1).

The best results are not always for the highest cardinality merges but overall tendency confirms that higher cardinality gives better results – average error (percentage of incorrectly classified records) for single merges with cardinality of two is 14.67 for class Setosa (minimal error for this cardinality is 10), 13.33 for Versicolor (minimal 14) and 11.67 for Virginica (minimal 6), whereas single merges with cardinality of three have following average errors: 9.5 for class Setosa (minimal error for this cardinality is 2), 9 for Versicolor (minimal 6) and 6 for Virginica (minimal 0). Combinations with more than one merge with cardinality of two have a different error pattern than those with one merge of same cardinality – the errors are sometimes lower than with combinations with one merge of higher cardinality (e.g., class Virginica – minimal error is 0 with a combination with two merges of cardinality of four).

Using CART to evaluate separation of the merged clusters resulted in similar numbers with slight changes that could influence the choice of combinations for classification. The best results for class Setosa were exactly the same as with C4.5 classifier, results for class Versicolor were similar with the best combination being $\{C_1, C_2, C_3\},\{C_4\}$ and there were few differences for class Virginica where the best combination was $\{C_1, C_2, C_3\},\{C_4\}$.

1. The best combinations (those with the smallest error) are chosen and the examples are re-labeled assigning new names that map the examples to the original classes (prefix,

e.g., set_) and also contain information about cluster merges (indexes of clusters). Then this modified data set is classified according to these new classes. Confusion matrix of classification with default data set is shown in

2. Table 2 (classes are as follows: a – Iris-setosa, b – Iris-versicolor, c – Iris-virginica).

Table 1. Cluster combination classification errors for each class using C4.5. Best result lines are shaded

Class Iris-Setosa

Clusters				Err%
C_1	C_2	C_3	C_4	16
$\{C_1, C_2\}$		C_3	C_4	10
$\{C_1, C_3\}$		C_2	C_4	24
$\{C_1, C_4\}$		C_2	C_3	18
C_1	$\{C_2, C_3\}$		C_4	10
C_1	$\{C_2, C_4\}$		C_3	16
C_1	C_2	$\{C_3, C_4\}$		10
$\{C_1, C_2, C_3\}$			C_4	6
C_1	$\{C_2, C_3, C_4\}$			2
$\{C_1, C_3, C_4\}$			C_2	18
$\{C_1, C_2, C_4\}$			C_3	12
$\{C_1, C_2\}$		$\{C_3, C_4\}$		4
$\{C_1, C_3\}$		$\{C_2, C_4\}$		22
$\{C_1, C_4\}$		$\{C_2, C_3\}$		6

Class Iris-Versicolor

Clusters				Err%
C_1	C_2	C_3	C_4	16
$\{C_1, C_2\}$		C_3	C_4	14
$\{C_1, C_3\}$		C_2	C_4	12
$\{C_1, C_4\}$		C_2	C_3	18
C_1	$\{C_2, C_3\}$		C_4	8
C_1	$\{C_2, C_4\}$		C_3	16
C_1	C_2	$\{C_3, C_4\}$		12
$\{C_1, C_2, C_3\}$			C_4	4
C_1	$\{C_2, C_3, C_4\}$			6
$\{C_1, C_3, C_4\}$			C_2	16
$\{C_1, C_2, C_4\}$			C_3	10
$\{C_1, C_2\}$		$\{C_3, C_4\}$		14
$\{C_1, C_3\}$		$\{C_2, C_4\}$		12
$\{C_1, C_4\}$		$\{C_2, C_3\}$		4

Class Iris-Virginica

Clusters				Err%
C_1	C_2	C_3	C_4	5
$\{C_1, C_2\}$		C_3	C_4	3
$\{C_1, C_3\}$		C_2	C_4	9
$\{C_1, C_4\}$		C_2	C_3	6
C_1	$\{C_2, C_3\}$		C_4	6
C_1	$\{C_2, C_4\}$		C_3	5
C_1	C_2	$\{C_3, C_4\}$		6
$\{C_1, C_2, C_3\}$			C_4	2
C_1	$\{C_2, C_3, C_4\}$			2
$\{C_1, C_3, C_4\}$			C_2	6
$\{C_1, C_2, C_4\}$			C_3	2
$\{C_1, C_2\}$		$\{C_3, C_4\}$		0
$\{C_1, C_3\}$		$\{C_2, C_4\}$		6
$\{C_1, C_4\}$		$\{C_2, C_3\}$		10

Table 2. Confusion matrix for default classes

		Classified as		
		a	**b**	**c**
Real class	**a**	49	**1**	0
	b	0	47	**3**
	c	0	**2**	48

First class (Setosa) is decomposed into two subclasses – set_1 (class Setosa cluster C_1; labeled 'a' in confusion matrix) and set_234 (b), class Versicolor into vers_14 (c) and vers_23 (d) and class Virginica into subclasses virg_12 (e) and virg_34 (f). The resulting confusion matrix is shown in Table 3.

Table 3. Confusion matrix for decomposed classes

		Classified as					
		a	**b**	**c**	**d**	**e**	**f**
Real class	**a**	16	0	0	0	0	0
	b	1	32	**1**	0	0	0
	c	0	0	13	2	0	**2**
	d	0	0	3	28	0	**2**
	e	0	0	0	0	19	0
	f	0	0	0	**1**	0	30

The misclassified examples outside original classes are concentrated in class Versicolor that are classified as Virginica. Therefore this result could be improved by choosing different cluster combination, e.g. [{C_1, C_2, C_3}, {C_4}] has the same result as [{ C_1, C_4}, { C_2, C_3}] using C4.5 and higher using CART. The results of using this combination are shown in confusion matrix (Table 4).

Table 4. Confusion table for decomposed classes. Different combination for class Versicolor

		Classified as					
		a	**b**	**c**	**d**	**e**	**f**
Real class	**a**	16	0	0	0	0	0
	b	1	32	0	1	0	0
	c	0	0	42	1	0	3
	d	0	0	1	3	0	0
	e	0	0	0	0	19	0
	f	0	0	1	0	0	30

Changing cluster combination for class Versicolor which had the most misclassified examples led to slight improvement even if both combinations seemed equal.

6.4. Performance comparison on the Breast Cancer (Diagnostic) problem

Another data set is 'Breast Cancer Wisconsin (Diagnostic)' with 32 attributes – 1 label attribute, 30 real number attributes and 1 class attribute with following classes: M – malignant and B – benign.

According to dendrograms (Figures 5 and 6) attained by hierarchical clustering, class B can be divided into four easily distinguishable clusters and class M – into three clusters. To explore the changes in error depending on the number of clusters, the same data (class M) was also divided into four clusters.

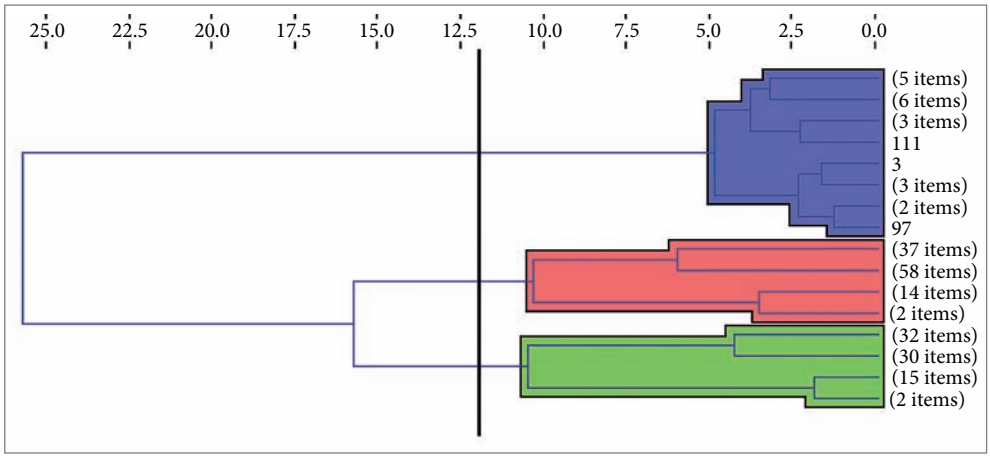

Fig. 5. Examples of class M are grouped into clusters. Optimal number of clusters – 3 (cut-off line at height 12)

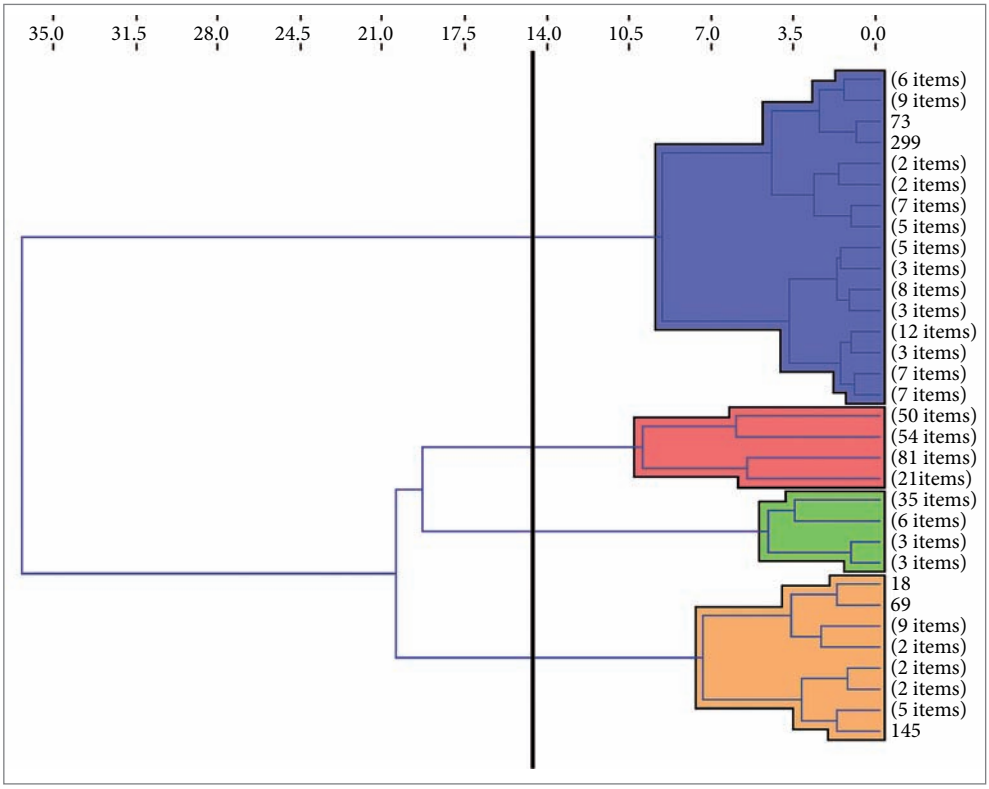

Fig. 6. Examples of class B are grouped into clusters. Optimal number of clusters – 4 (cut-off line at height 15)

These clusters are assigned new labels and then they are merged in different combinations. Exhaustive search was used to illustrate contradictions between a small error when dividing one class and error in the final classification. Cluster combinations and corresponding results of attribute space division (percentage of incorrectly classified records) for each class are given in Table 5. Four cluster and corresponding three cluster combinations are shown for class M.

In this case, the least classification errors fall under combinations with higher cardinality of merged clusters for both classifiers – C4.5 and CART. The best combinations using C4.5 and using CART also coincide. Results for CART are less scattered but have higher classification error. The combinations with least classification error (for class M there is one combination of three clusters and one of four clusters) are then chosen for the next step – class names are replaced with cluster labels (representing original class name and cluster combination, for example for class B combination [{C_1, C_2, C_3}, {C_4}] cluster labels B_123 and B_4 are used) and are then treated as separate classes in further classification.

Since there are two combinations chosen for each class, combinations of these options will be used to form output data set. The results of classification are shown in Table 6. The combinations are fitted to get the best result (see the last line of Table 6 for CART). From Table 5 and Table 6 one can see that best scores for combinations in classification that classifies subclasses of one class do not correspond to the best scores in final classification although the difference is not significant for classifier C4.5.

Table 5. Classification errors for each class using C4.5. Best combination lines are shadowed

Class B

Clusters				Err%
C_1	C_2	C_3	C_4	17.6471
{ C_1, C_2}		C_3	C_4	12.605
{ C_1, C_3}		C_2	C_4	18.2073
{ C_1, C_4}		C_2	C_3	19.3277
C_1	{ C_2, C_3}		C_4	12.605
C_1	{ C_2, C_4}		C_3	15.1261
C_1	C_2	{ C_3, C_4}		17.9272
{ C_1, C_2, C_3}			C_4	5.8824
C_1	{ C_2, C_3, C_4}			8.1232
{ C_1, C_3, C_4}			C_2	15.9664
{ C_1, C_2, C_4}			C_3	5.3221
{ C_1, C_2}		{ C_3, C_4}		10.084
{ C_1, C_3}		{ C_2, C_4}		14.2857
{ C_1, C_4}		{ C_2, C_3}		12.0448

Class M

Clusters (4)				Clusters (3)			Err%
C_1	C_2	C_3	C_4	–			14.1509
{ C_1, C_2}		C_3	C_4	–			7.0755
{ C_1, C_3}		C_2	C_4	–			9.434
{ C_1, C_4}		C_2	C_3	–			14.6226
C_1	{ C_2, C_3}		C_4	–			10.8491
C_1	{ C_2, C_4}		C_3	–			9.9057
C_1	C_2	{ C_3, C_4}		C_1	C_2	C_3	14.1509
{ C_1, C_2, C_3}			C_4	–			4.2453
C_1	{ C_2, C_3, C_4}			C_1	{ C_2, C_3}		5.1887
{ C_1, C_3, C_4}			C_2	–			8.9623
{C_1, C_2, C_4}			C_3	–			9.9057
{ C_1, C_2}		{ C_3, C_4}		{ C_1, C_2}		C_3	11.3208
{ C_1, C_3}		{ C_2, C_4}		–			11.7925
{ C_1, C_4}		{ C_2, C_3}		–			11.7925

Table 6. Classification errors of decomposed classes transformed for existing classe

Combination					Error (%)	
Class B		Class M			C4.5	CART
No class decomposition					4.9209	7.5571
B_123	B_4	M_123	M_4		6.3269	7.7329
B_123	B_4	M_1	M_234		7.0299	7.3814
B_124	B_3	M123	M_4		5.9754	6.1511
B_124	B_3	M_1	M_234		5.6239	6.3269
B_123	B_4	M_1	M_2	M_34	6.8541	5.7996

Although dendrograms of both classes (Figures 6 and 5) show that there are easily separable clusters (three for class 'Malicious' and four for class 'Benign'), dividing data into subclasses does not lead to improved performance of C4.5 classifier and improves performance of CART only slightly. The best result for CART is not better than C4.5 without class decomposition. Apparently clusters of different classes overlap and are easier to distinguish for the approach used in CART. This leads to even worse results for C4.5 than initial data. A possible solution is choosing a larger number of clusters but this process also consumes much more time and resources and the obtained improvement might be too small for the invested work.

6.5. Performance comparison on the Parkinson's problem

The third data set is 'Parkinson's' with 24 attributes – a label attribute unique for each record, 22 real valued attributes and class attributes with two values: 0 – healthy, 1 – Parkinson's disease.

This dataset is also split into subsets with records belonging to the same classes. These subsets are clustered and cluster labels are assigned to records, allowing us to perform classification to explore structures of the classes. Classification results are shown in Table 7.

This case shows similar pattern as the previous data sets – the least classification errors fall under combinations with cardinality of three and combinations with two merges with cardinality of two. Although the best scores were achieved using C4.5 the minimal errors within both classes are achieved using CART: 0% error for combination [{C_1, C_2, C_3}, {C_4}] in class 1 and 0% error for combination [{C_1, C_2, C_4}, {C_3}]. The best results using CART are for the same combinations that had the best result with C4.5 but the second best results differ: [{C_1, C_2}, {C_3, C_4}] for class 1 but the best results for class 0 are for combinations [{C_1, C_2},{ C_3},{C_4}], [{C_1, C_4},{C_2, C_3}] and [{C_1, C_2}, {C_3, C_4}].

Classification of the data using CART and C4.5 without class decomposition favors CART as better classifier with 14,4% incorrectly classified records instead of C4.5 with 17,9% (see the second line of Table 8).

When classification is performed using data with decomposed classes, it is obvious that C4.5 performs better with decomposed classes than it does with original classes. C4.5 with

decomposed classes even outperforms CART when solving this problem. Classification error for C4.5 classifier is reduced up to 40%. Although classes decomposed according to combinations that scored best in explorative classification don't show reduced performance in C4.5 classification, they certainly can worsen performance of CART classifier (only one case of class decomposition shows a slightly better result).

Table 7. Cluster combination classification errors for each class using C4.5. Best combination lines are shadowed

Class 1

Clusters				Err%
C_1	C_2	C_3	C_4	19.7279
$\{C_1, C_2\}$		C_3	C_4	10.8844
$\{C_1, C_3\}$		C_2	C_4	12.2449
$\{C_1, C_4\}$		C_2	C_3	16.3265
C_1	$\{C_2, C_3\}$		C_4	12.9252
C_1	$\{C_2, C_4\}$		C_3	18.3673
C_1	C_2	$\{C_3, C_4\}$		9.5238
$\{C_1, C_2, C_3\}$			C_4	3.4014
C_1	$\{C_2, C_3, C_4\}$			14.2857
$\{C_1, C_3, C_4\}$			C_2	10.2041
$\{C_1, C_2, C_4\}$			C_3	12.9252
$\{C_1, C_2\}$		$\{C_3, C_4\}$		9.5238
$\{C_1, C_3\}$		$\{C_2, C_4\}$		6.8027
$\{C_1, C_4\}$		$\{C_2, C_3\}$		12.9252

Class 0

Clusters				Err%
C_1	C_2	C_3	C_4	14.5833
$\{C_1, C_2\}$		C_3	C_4	10.4167
$\{C_1, C_3\}$		C_2	C_4	16.6667
$\{C_1, C_4\}$		C_2	C_3	6.25
C_1	$\{C_2, C_3\}$		C_4	20.8333
C_1	$\{C_2, C_4\}$		C_3	8.3333
C_1	C_2	$\{C_3, C_4\}$		18.75
$\{C_1, C_2, C_3\}$			C_4	8.3333
C_1	$\{C_2, C_3, C_4\}$			8.3333
$\{C_1, C_3, C_4\}$			C_2	6.25
$\{C_1, C_2, C_4\}$			C_3	2.0833
$\{C_1, C_2\}$		$\{C_3, C_4\}$		8.3333
$\{C_1, C_3\}$		$\{C_2, C_4\}$		4.1667
$\{C_1, C_4\}$		$\{C_2, C_3\}$		20.8333

Table 8. Classification errors of decomposed classes transformed for existing classes

Class 0		Class 1		CART Error	C4.5 Error
No class decomposition				14.359	17.9487
0_124	0_3	1_123	1_4	15.3846	12.3077
0_124	0_3	1_13	1_24	18.9744	10.7692
0_13	0_24	1_123	1_4	13.3333	15.3846
0_13	0_24	1_13	1_24	24.1026	12.8205

7. Conclusions

It is hard to define a mathematical approach to attribute space analysis and description for classification model building because the efficiency of methods and their parameters depend on data structure. There are different approaches to class decomposition that lead to differ-

ent results and fit certain data structures. Hierarchical clustering is a suitable approach to high density area discovery within classes because of its elasticity choosing cluster defining parameters (similarity measures and linkage options) and it also does not need any prior information about the number of clusters and their positions (centroids). This allows a more objective class structure analysis.

The choice of classification model is also relevant because the best model for initial data is not always the best model for data with decomposed classes. This also depends on class structure and the character of overlapping areas.

There cannot be a universal approach to cluster combination selection without taking into account the final classification. Although the observed trend complies with previously proposed heuristics, the best results do not always comply with the trend.

This paper puts forward a cluster analysis based technique for analyzing classifier training set structure. The methodology of effective classifier design is implemented by synthesizing class descriptions that employ a combination of well separable clusters.

Information about class structure can be used to improve performance of the classifiers. Experiments show a significant improvement up to 40%.

In future research, more attention will be paid to studying cluster structure sensitivity that reflects real location of classes as well as to studying clustering quality.

References

Asuncion, A.; Newman, D. J. 2007. *UCI Machine Learning Repository.* Irvine, CA: University of California, School of Information and Computer Science. Available from Internet: <http://www.ics.uci.edu/~mlearn/MLRepository.html>.

Brazdil, P. B.; Soares, C.; Da Costa, J. P. 2003. Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results, *Machine Learning* 50: 251–277. doi:10.1023/A:1021713901879

Breiman, L.; Friedman, J.; Olshen, R.; Stone, C. 1984. *Classification and Regression Trees*. Wadsworth Int. Group.

Deza, E.; Deza, M. M. 2009. *Encyclopedia of Distances.* Springer-Verlag, Berlin, Heidelberg. doi:10.1007/978-3-642-00234-2

Fulton, T.; Kasif, S.; Salzberg, S.; Waltz, D. 1996. Local induction of decision trees: towards interactive data mining, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 14–19.

Han, J.; Kamber, M. 2006. *Data Mining: Concepts and Techniques.* 2nd ed. Morgan Kaufmann.

He, Q. 1999. A review of clustering algorithms as applied in IR, *UIUCLIS–1999/6+IRG*.

Jiang, D.; Eick, C. F.; Chen, C. 2007. On supervised density estimation techniques and their application to spatial data mining, in *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems, p. 65; Seattle, Washington, USA, November 7-9, 2007.* doi:10.1145/1341012.1341089

Kohavi, R. 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection, *IJCAI*: 1137–1145.

Kohavi, R.; Quinlan, J. R. 2002. Decision-tree discovery, in Will Klosgen and Jan M. Zytkow (Eds.). *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, 267–276.

Looney, C. G. 2002. Interactive clustering and merging with a new fuzzy expected value, *Pattern Recognition* 35(11): 2413–2423. doi:10.1016/S0031-3203(01)00213-8

Michalopoulos, M.; Dounias, G. D.; Thomaidis, N.; Tselentis, G. 1999. Decision making using fuzzy c-means and inductive machine learning for managing bank branches performance, in *Proceedings of the European Symposium on Intelligent Techniques, ESIT'99, 7 pages (Proc. on CD-Rom); Chania, Greece, June 3–4 1999.*

Ozols, J.; Borisov, A. 2001. Fuzzy classification based on pattern projections analysis, *Pattern Recognition* 34: 763–781. doi:10.1016/S0031-3203(00)00029-7

Quinlan, J. R. 1986. Induction of Decision Trees, *Machine Learning* 1(1): 81–106. doi:10.1007/BF00116251

Quinlan, J. R. 1987. Simplifying decision trees, *International Journal of Man-Machine Studies* 27(3): 221–248. doi:10.1016/S0020-7373(87)80053-6

Quinlan, J. R. 1993. C4.5: *Programs for Machine Learning*. Morgan Kaufmann Pub., San Mateo, CA.

Vilalta, R. 2006. Identifying and characterizing class clusters to explain learning performance, *Journal of Systems and Software* 80(1): 63–73.

Vilalta, R.; Achari, M. K.; Eick, C. F. 2003. Class decomposition via clustering: a new framework for low-variance classifiers, in *Third IEEE International Conference on Data Mining (ICDM'03)*, 673–676.

Ward, J. H. 1963. Hierarchical grouping to optimize an objective function, *J. Am. Statist. Assoc.* 58: 236–244. doi:10.2307/2282967

KLASTERIŲ SPRENDIMŲ MEDŽIU PAGRĮSTAS STATYBOS KLASIFIKATORIUS

I. Polaka, A. Borisov

Santrauka. Straipsnyje nagrinėjamos duomenų struktūros tyrimo galimybės naudojant klasterinę analizę. Tiriamos įvairaus tankio klasės ir siekiama tokio klasės išskaidymo laipsnio, kad padidėtų sprendimų medžio klasifikatorių veiksmingumas. Klasės suskaidomos klasterinės analizės ir klasterių sujungimo metodais, naudojant sprendimų medžio klasifikatorius. Tada klasės skilimo poveikis parodomas C4.5 ir CART klasifikatoriuose. Pagrindinis dėmesys skiriamas bandymams, atliktiems su realių duomenų rinkiniais. Eksperimentai aprašomi žingsnis po žingsnio, iliustruojant atrastus modelius, kurie veikia anksčiau siūlytus klasių modelius.

Reikšminiai žodžiai: klasifikacija, klasės skilimas, klasterinė analizė, sprendimų medžiai, duomenų generavimas.

Inese POLAKA received her Master's degree in Information Technology from Riga Technical University. Her research interests include data mining, classification methods, data preprocessing, ontology aware classification and other methods to increase effectivity of classification models.

Arkady BORISOV is Professor of Computer Science in the Faculty of Computer Science and Information Technology at Riga Technical University (Latvia). He holds a Doctor of Technical Sciences degree in Control in Technical Systems and the Dr.habil.sci.comp. degree. He has supervised a number of national research grants and participated in the European research project ECLIPS. His research interests include fuzzy sets, fuzzy logic and computational intelligence. He has 205 publications in the area.