Scientific Journal of Riga Technical University
Computer Science. Information Technology and Management Science
2010
Volume 44

Using Data Structure Properties in Decision Tree Classifier Design

Inese Polaka, *Riga Technical University*, Arkady Borisov, *Riga Technical University*

***Abstract* –This paper studies the techniques of performance enhancement for decision tree classifiers (DTC) that are based on data structure analysis. To improve the performance of DTC, two methods are used – class decomposition that uses the structure of class density and taxonomy based DTC design that uses interactions between attribute values. The paper shows experimental exploration of the methods, their strengths and imperfections and also outlines the directions for further research.**

***Keywords* –attribute value taxonomy, class decomposition, classification, decision tree classifiers, classifier performance enhancement**

I. Introduction

The classification task is very popular in various domains – medicine, finance, biology, engineering etc. and there are many diverse classification methods that are applied to the data. One of the most popular approaches to classification is using decision tree classifiers to discriminate between classes. Decision tree classifiers are fairly easy to construct and they are very easy to understand even for experts that have little knowledge in data mining and about classification methods. One does not need to know how decision trees were constructed to successfully use them for classification of new instances.

Decision tree classifiers are quite an old approach (first algorithms CART and C4.5 were first used in the 1980s by Breiman et al.[1] and Quinlan[2]) and are modified and improved ever since their first appearance. Classification trees are an unstable method meaning that they produce different classification models if training data change even slightly. To avoid this imperfection, classifier ensembles are widely used. Decision tree classifiers are combined together resulting in decision forests and they are also combined with other classification methods (like neural networks, Naïve bayes classifiers etc.) resulting in hybrid decision trees.

Another way to improve the performance of decision tree classifiers is to use various data preprocessing routines and data analysis tools to gain meta-information about data sets that can be used in decision tree design. Various features of data sets can be used to modify the process of decision tree design; this area is not widely studied and there is almost no information about the most important features of a data set that can help in choosing the best classification algorithm and building a classifier that best fits the data.

If the data is viewed as a combination of input and output parameters, these two parts can both be used in classifier design. The input parameters hold various values of several attributes. The nature of these attributes and their values can be used to construct more compact and accurate classifiers. The composition of output data or classes can give more information about requirements towards classifiers and help the scientists to build classifiers that fit the complex nature of classes. Therefore this article examines two methods of decision tree classifier accuracy enhancement. Taxonomy based decision tree design uses interactions between values of attributes to build compact and accurate classifiers whereas class decomposition splits complex class structures into compact and dense areas that can be viewed as subclasses and separated more easily using classical classification algorithms.

The paper is organized as follows. An overview of related work by other researchers is presented in Section 2. A description of the used methods is given in Section 3. The results of the conducted experiments are presented Section 4. Finally, some concluding remarks are made.

II. Related Work

When classical algorithms do not give the expected results, scientists have tried to use the features of the data structure to adapt classifiers to data. In this section two directions of those researches are inspected – the use of data structure features to build local classifiers and the use of ontologies to build more efficient classifiers.

A. Data structure analysis for classifier design

The transition from global data structure exploration to local analysis was first introduced by Fulton et al. [2]. The authors described local design of decision trees, exploring the objects that are nearest to the object that is being classified. This method is similar to *k*Nearest Neighbor method and uses it to find the nearest objects (the authors also propose other methods to find the nearest objects). Then a classifier is built using the found objects as a training set. The proposed method introduces interactive classification but for each new sample there has to be found a set of objects that will classify the instance. Therefore this method is very resource costly. It is also very difficult to find the nearest neighborhood that would represent the specifics of the domain.

Brazdil et al. [4] proposed a meta-learning system that examines the data set and chooses the most suitable classifier by comparing the meta-information of the data set with the information about other data sets that is stored in a database. Then a classification method is chosen based on the performance of classifiers using data sets that are in the data base. It is very difficult to properly describe a data set and therefore such system cannot be built using methods and knowledge that is available at the moment. More data

structure analysis methods need to be constructed to fully understand the nature of the data and the behavior of the classification methods.

Vilalta [5] proposed a method of data exploration using *k*-means clustering to identify areas of high density within classes. The clusters found during the process of clustering are treated as separate classes. This enables the use of less complex classifiers (the author used linear classifiers) but the number of needed classifiers increases. The author also used Naïve Bayes classifier to test the proposed method and achieved better results.

B. Ontology based classifier design

The use of ontologies is very popular in data mining, especially in the field of biomedicine and text mining. But this approach is relatively new to classical classification tasks.

Taylor et al. [6] proposed the use of ontology in classifier design, using knowledge acquisition from data bases, combining relation data bases and ontologies to acquire new possibilities in query construction. The authors introduce the ParkaDB tool to manage ontologies and use them in rule construction and classification.

Zhang et al. [7] propose a new decision tree classification method that uses attribute value taxonomies. The proposed method ODT uses data in various abstraction levels and builds classification trees. To choose a split criterion this method searches the attribute space in various abstraction levels (enabled by attribute value taxonomies of each attribute) and chooses the best value based on Information gain. This research only leaves the question about the construction of taxonomies open.

Kang et al. [8] proposed the method of automatic taxonomy construction called AVT-Learner. This method is clustering based because it uses distances between attribute values and combines the closest values. The authors experimented with various distance measures and found Jensen-Shannon divergence to be the most successful. The authors used this proposed method with Naïve Bayes classifier and achieved better results than without the use of AVT-Learner.

III. METHODOLOGY

In this research two decision tree classifier efficiency enchancement methods that use information about data structure in building classifiers were used – the class decomposition that uses class density structures and the ontology based decision tree design that uses information about interactions between attributes.

A. Class Decomposition

Class decomposition enables the use of data structure features in the process of decision tree building. This method uses clustering approach to acquire meta-information like high-density areas within each class about the structure of input data. This information can later be used in building decision tree classifiers that are more accurate because the classifiers are adapted to the data. A benefit of this method is that it does not need any prior information about the data – it extracts all the needed information.

The first step of class decomposition is splitting the data according to classes. Given a data set $T=\{(x,y)\}$, where $x = \{x_1, x_2, \ldots, x_n\}$ is an n-dimensional vector that describes an object of the data set and $y = \{ y_1,y_2, \ldots, y_m\}$ is the set of classes, the data set T is split into subsets $T_1, T_2, \ldots, T_k$ so that all records of one subset belong to the same class $T_i = \{(x, y_i)\}$ and $i = 1 \ldots m$, and all records of the initial data set belong to exactly one subset.

A hierarchical agglomerative clustering algorithm is applied to each subset T_i. A hierarchical algorithm is chosen because it does not need any prior information about clusters (number of clusters, centroids etc.) and it can reveal the underlying patterns in the data. This method also reveals the number of clusters that are furthest from each other (maximum distance between two mergers in the dendrogram) that will be easier to separate using decision tree classifiers and the clusters are compact (Ward's distance is used to choose clusters that will be merged in each step of clustering). After clustering algorithm is applied, a number of clusters is chosen that represents the clusters that are furthest apart from each other. Each record x is then labeled according to their belonging to the clusters resulting in modified T_i subsets T_i'.

In the next step the optimal number of clusters and their structure is determined using various combinations of clusters chosen in the first step and decision tree classifiers (algorithms C4.5 and CART) to assess the separation ability. The best combination is the combination whose subclasses can be separated with the least error.

The records of the subset with the best cluster combination T_i * are each given a label y_j' according to the cluster combinations;as a result, a subset T_i' turns into subset T_i'' = $\{(x, y_j')\}$. The label is chosen in a way that lets identify the primary class from the new label that is treated as the new class.

Then $T'' = \bigcup_{i=1}^{k} T_i''$ of the subsets are joined into one set and the classification algorithm (C4.5 and CART) is applied to this set allowing to evaluate class decompositions. To evaluate the accuracy of the classifiers the new class labels are converted to original classes.

B. Ontology Based Classifier Design

To use the dependencies between attributes in classifier design, they have to be represented in a comprehensible manner. In this case taxonomy (a hierarchical ontology that uses IS-A links) is used to represent the connections. There are many ways to construct taxonomy. In this research manual and AVT-Learner produced taxonomies are used.

The manual taxonomy is constructed using the information that is available about the domain or that is obtained using statistical data analysis.

To construct the taxonomies automatically the AVT-Learner method (first proposed by Hang et al. in [8]) is used. This method implements the following process for each attribute:

1. For each value of the attribute a frequency with which class C_j is observed is calculated, then a probability $p(C_k|v_i^j)$ for each class is obtained where v_i^j is the value of the attribute;
2. Class probability distributions $P=\{p(C_1|v_i^j), p(C_2|v_i^j), ..., p(C_k|v_i^j)\}$ are calculated for each attribute value;
3. Based on probability distributions found in the second step the distances between each two attributes are calculated. The distance measure used in calculations is Jensen-Shannon divergence that has been proven to be effective in such situations [13].
4. A pair of attributes v_i^x and v_i^y with the least distance between them is found.
5. The closest values are combined into value v_i^{xy} and the class probability distribution $P(C|v_i^{xy})$ is calcualted for this value.
6. The value v_i^{xy} is added to the taxonomy as predecessor of values v_i^x and v_i^y.
7. The previous steps are repeated until all values of the attribute are combined into one value that is the root node of the taxonomy.

To use these taxonomies in decision tree design,the ODT algorithm [9] is used. Given a data set S with an attribute set $A=\{A_1, A_2, ..., A_n\}$ and a set of classes $C=\{ C_1, C_2, ..., C_m \}$, for each attribute there is a taxonomy in the taxonomy set $T=\{T_1, T_2, ..., T_n\}$. Let the successors of a node c be $\psi(c)$. Each leaf node of the taxonomy splits the data set into subsets. To each subset belong k pointers to k concepts in k taxonomies. $P=\{ p_1, p_2, ..., p_k \}$ represents the vector of pointers where each p_i points to a concept in taxonomy T_i. A signal $\Phi(P)=true$ is used if pointers of the vector P point to leaf nodes.

The algorithm chooses an attribute to split the data set which will cause the maximum decrease in entropy. The algorithm consists of the following steps:

1. If all records in data set S belong to the same class, it returns the class label in a tree that consists of one leaf;
2. Otherwise the best attribute and concept for splitting is determined using *Gain* measure;
3. The splitting criteria are successors of the chosen concept;
4. The data set S is split using the chosen criteria;
5. The algorithm performs the previous steps to construct the sub-trees.

The cut in the taxonomy is performed in a way that for each leaf node l of the taxonomy the following conditions are true: l belongs to the cut line or is a successor of a node that belong to the cut line and each two nodes that belong to the cut line are not predecessors or successors to each other.

C. Combination of both methods

This research also focuses on the combination of class decomposition and ontology based decision tree classifier design. First class decomposition is performed until labels of new classes are assigned to each record. Then taxonomies are built and the classification using these taxonomies is applied to the data set. Then the original class labels are assigned and the resulting accuracy is evaluated.

IV. Results

Experiments were conducted using each of the proposed methods alone and using several different data sets. Then the combination of methods was applied to three data sets. The results were compared to the performance of algorithms CART and C4.5 without the use of any performance enhancement methods.

A. Class decomposition

For experiments using class decomposition method *UCI Machine Learning Depository*[10]data sets were used: Iris data set, Wisconsin Breast Cancer (Diagnostic) data set and Parkinsons data set. These data sets were chosen because they had few classes (to decrease the number of classes that was derived using class decomposition for illustrational purposes) and enough records to support each of the new subclasses.

Without the use of class decomposition the error of decision tree classifiers using Iris data set was around 5.33%. But the use of class decomposition increased the performance to an error of 4%.

For Wisconsin Breast Cancer data set both classifiers showed different results – C4.5 algorithm had 4.92% error but CART algorithm showed 7.56% error. This once again shows that although both decision tree classification algorithms are similar, they may have a significant performance difference and no one decision tree classification algorithm is superior in all data sets.

TABLE I
Classifiers' Partition Efficiency for Cluster Combinations

No.	Cluster combinations	Error (%)	
		C4.5	CART
1.	C_1 C_2 C_3 C_4	14.1509	13.2075
2.	$\{C_1, C_2\}$ C_3 C_4	7.0755	11.3208
3.	$\{C_1, C_3\}$ C_2 C_4	9.434	11.3208
4.	$\{C_1, C_4\}$ C_2 C_3	14.6226	15.0943
5.	C_1 $\{C_2, C_3\}$ C_4	10.8491	11.7925
6.	C_1 $\{C_2, C_4\}$ C_3	9.9057	12.2642
7.	C_1 C_2 $\{C_3, C_4\}$	14.1509	11.7925
8.	$\{C_1, C_2, C_3\}$ C_4	4.2453	5.1887
9.	C_1 $\{C_2, C_3, C_4\}$	5.1887	7.0755
10.	$\{C_1, C_3, C_4\}$ C_2	8.9623	10.8491
11.	$\{C_1, C_2, C_4\}$ C_3	9.9057	8.0189
12.	$\{C_1, C_2\}$ $\{C_3, C_4\}$	11.3208	12.2642
13.	$\{C_1,$ $\{C_2,$	11.7925	10.3774

	C_3 }	C_4 }		
14.	{C_1, C_4 }	{C_2, C_3 }	11.7925	11.3208

Each class in the Wisconsin Breast Cancer data set was split into four clusters because dendrograms showed that this number of clusters was optimal – the next merge was between clusters with the largest distance in the dendrogram.The last merge was not taken into account because splitting a class into two subclasses gives no reason for cluster combinations and gives little information about the class structure. If the structure of the class is best described by these two clusters, the cluster combination that represents these two clusters will reveal this structure. All combinations of the clusters were tested using classification algorithms. The result for class Malicious is shown in Table I.

TABLE II

CLASSIFICATION ERRORS FOR DECOMPOSED CLASSES

Combination					Error (%)	
Class Benign		Class Malicious			C4.5	CART
Without 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	M_123	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 Vilalta [5] states that the best combinations can be revealed within each class by classification, two of the best combinations for each class were chosen for full classification to test the hypothesis. Then all combinations of single class decompositions were used for full classification using algorithms C4.5 and CART. The classification results are shown in Table II. The classification accuracy decreased from 4.92% to 5.62% for algorithm C4.5 when class decomposition was applied. This can be explained by specific class structure that makes it harder to distinguish subclasses for this particular algorithm. The performance of CART algorithm increased from 7.56% to 5.80% when class decomposition was applied.

The last row in the table shows results for class decomposition combination of Malicious class that did not show the best results in the within class evaluation and the cardinality is not the highest possible (previously proposed heuristic for choosing the best combination) but the result in the full classification is the best for CART algorithm. This means that subclasses of different classes overlap although subclasses within one class did not overlap and were easily partitioned by the same classification algorithms.Within class evaluation of cluster combinations needs more research and new heuristics. The best evaluation using available heuristics is the full classification that needs more calculations and resources.

TABLE III

CLASSIFICATION ERRORS FOR DECOMPOSED CLASSES

Combination				Error (%)	
Class 0		Class 1		C4.5	CART
Without 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

The results using Parkinsons data set are shown in Table III. Without the use of class decomposition the best result is using C4.5 algorithm with 14.36%. But the use of class decomposition improved the performance of CART algorithm more and the best result was 10.77% error that outperformed the best result of C4.5 algorithm (the best result using C4.5 algorithm was 13.33%).

These experiments show that class decomposition can significantly improve the performance of decision tree classifiers but it needs more research to improve the heuristics of cluster combination choice.

B. Ontology based classifiers

To successfully use ontologies in decision tree classifier design it is important to use specific data sets that have attributes with many values that can be merged to build attribute value taxonomies. For this purpose Miles per Gallon data set (mostly continuous variables that can be categorized) and Iris data set (also continuous attributes) from the *UCI Machine Learning Repository* were used. Also the following data sets from *The Data and Story Library* were used: Acorn data set, Flea Beetle data set and Reading test data set.

Ontologies were built for all attributes that had more than four values and they were used for decision tree classifier design using algorithm C4.5. Ontologies were built using available information about the data sets (also statistical information) and AVT-Learner algorithm.

TABLE IV

CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING MILES PER GALLON DATA SET

Parameter	Values		
	Without AVT	Manual AVT	AVT-Learner
Percentage of incorrectly classified instances	25.77 %	26.53 %	23.98 %
Number of nodes in the tree	64	62	32
Number of leaves in the tree	48	38	19

The performance of C4.5 classifier using Miles per Gallon data set without the use of ontologies was 25.77% and the classification tree was rather large for the data set. The use of ontologies improved the performance of the classifiers and the use of ontologies built by AVT-Learner method also decreased the number of nodes in the classification tree while improving the performance. The results are shown in Table IV.

TABLE V

CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING IRIS DATA SET

Parameter	Values

	Without AVT	Manual AVT	AVT-Learner
Percentage of incorrectly classified instances	5.33 %	3.33 %	5.33 %
Number of nodes in the tree	19	11	5
Number of leaves in the tree	17	8	3

The performance of the algorithm using Iris data set without the use of ontologies was 5.33% and it did not improve when AVT-Learner ontologies were used. But the use of manually built ontologies improved the performance by 2%. The Iris data set is widely known and the meta-information about the attributes is easy to deduce therefore the manually built taxonomies outperformed AVT-Learner taxonomies that did not capture the character of the attribute value interactions, however the use of AVT-Learner ontologies reduced the tree size significantly and therefore improved the efficiency of the classifier.The results are shown in Table V.

TABLE VI
CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING ACORN DATA SET

Parameter	Values		
	Without AVT	Manual AVT	AVT-Learner
Percentage of incorrectly classified instances	28.21 %	23.08 %	20.51 %
Number of nodes in the tree	1	7	5
Number of leaves in the tree	1	5	4

The classification error of the classifier using Acorn data set was 28.21%. The classifier consisted of one leaf node and classified all instances into the dominant class. When the taxonomies were used to build the classifier it classified instances into both classes. The best performance was shown by the classifier that used AVT-Learner taxonomies. The performance increased by almost 8% and the tree was compact – it consisted of five nodes, four of which were leafs. This means that one attribute was enough to reach this classification accuracy. Maybe using more attributes would increase the performance but the data set is small and larger trees are not efficient and are often overfitted. The results are shown in Table VI.

TABLE VII
CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING FLEA BEETLE DATA SET

Parameter	Values		
	Without AVT	Manual AVT	AVT-Learner
Percentage of incorrectly classified instances	48.65 %	6.76 %	1.35 %
Number of nodes in the tree	10	7	6
Number of leaves in the tree	9	5	4

When classification was performed without using taxonomies on Flea Beetle data set, the classification error was very large – 48.65%. The use of manual taxonomies increased the performance by 40% and reduced the size of the classification tree. The use of AVT-Learner taxonomies in decision tree design improved the performance even more and the classification error was only 1.35% and the tree size was also reduced.

TABLE VIII
CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING READING TEST DATA SET

Parameter	Values		
	Without AVT	Manual AVT	AVT-Learner
Percentage of incorrectly classified instances	62.12 %	40.91 %	31.82 %
Number of nodes in the tree	27	22	13
Number of leaves in the tree	26	15	7

The great improvement of the classifier efficiency can be explained by the specific character of the data set. The classifier built without the use of taxonomies classified all instances using only one attribute and the projections of the classes on this attribute axis overlapped a lot. The number of instances in this data set is small and the tree was not grown further. The use of taxonomies enabled the use of attribute values with higher abstraction level and therefore less values and each leaf held more instances and the tree could be built further resulting in better classification performance because the classes in two-dimensional space overlap less than their projections on one axis. The results are shown in Table VII.

The accuracy of the classifier was very low using Reading Test data set – the error was 62.12% because the classes overlap a lot and are hard to separate. The use of taxonomies in decision tree design improved the efficiency of the classifier – the classification error was reduced by 30% and the tree size was reduced by 50% leading to more accurate and faster classification of new instances. The results are shown in Table VIII.

The use of taxonomies in decision tree design process improved the accuracy of decision tree classifiers significantly. It also reduced the size of the classifiers making them more resource-efficient while classifying new instances.

C. Combination of both methods

Although both methods that were examined previously use information about data structure to design more accurate classifiers that suit data better, they have different approaches in studying the structure of data sets. Therefore the use of both methods simultaneously could improve the efficiency of decision tree classifiers even more.

For working with both methods the data has to suit both methods (especially the use of attribute value taxonomies because they require attributes with many values that would be enough to build a taxonomy) and the following data sets were chosen for experiments: Iris data set, Acorn data set and Flea Beetle data set.The combination of both methods was tested using C4.5 algorithm.

TABLE IX
CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING TAXONOMIES AND CLASS DECOMPOSITION FOR IRIS DATA SET

	Error %	# Misclassified instances	Tree size	
			Nodes	Leaves
Without AVT or CD	5.33%	8	19	17

AVT	3.33%	5	11	8
CD	4%	6	46	41
AVT and CD	4%	6	23	16

The performance of decision tree classifiers using Iris data set was improved by both methods when only one method was applied. The use of attribute value taxonomies in decision tree design decreased the classification error from 5.33% to 3.33% (the results are shown in Table IX) whereas the use of class decomposition resulted in 4% classification error. When both methods were applied to this data set the classification error was 4% (the error was the same when only class decomposition was applied) but the size of the classification tree decreased significantly – from 46 nodes to 23 nodes.

TABLE X

CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING TAXONOMIES AND CLASS DECOMPOSITION FOR ACORN DATA SET

	Error %	# Misclassified instances	Tree size	
			Nodes	Leaves
Without AVT or CD	28.21 %	11	1	1
AVT	20.51 %	8	5	4
CD	23.08 %	9	13	12
AVT andCD	23.08 %	9	17	12

The performance of decision tree classifiers using Acorn data set was very similar. The use of taxonomies showed the best improvement in accuracy – the classification error decreased by almost 8%. The use of class decomposition improved the accuracy by 5%. Also the use of both methods simultaneously showed the same classification error as class decomposition alone although the classifier was different and the tree was slightly larger (17 nodes instead of 13). The results are shown in Table X.

TABLE XI

CLASSIFICATION ERRORS AND CLASSIFICATION TREE SIZES USING TAXONOMIES AND CLASS DECOMPOSITION FOR FLEA BEETLE DATA SET

	Error %	# Misclassified instances	Tree size	
			Nodes	Leaves
Without AVT or CD	48.65 %	36	10	9
AVT	1.35 %	1	6	4
CD	32.43 %	24	10	9
AVT andCD	1.35 %	1	9	6

The C4.5 algorithm had the most increase in accuracy when taxonomy was used in decision tree design – the classification error dropped by 47%. The use of class decomposition also improved the accuracy but the increase was not that sharp – the classification error was 16% lower. The use of both methods showed the same classification accuracy as the use of taxonomies but the tree was slightly larger.The results are shown in Table XI.

This shows that although both methods proved to increase the efficiency of decision tree classifiers, the use of both methods simultaneously does not show even better results although the results are comparable to those obtained using only one of the methods. This means that both methods interfere in the process of classifier design and prevent further improvement of the performance.

V. CONCLUSION

The methods examined in this paper deal with two imperfections of decision tree classifiers – the static attribute values that impact the performance of classifiers and the tree sizes, and the approach of classification – decision trees divide the attribute space into hyperplanes that belong to the same class, therefore classes that are not compact and overlay other classes are hard to identify. Attribute-value taxonomies were used to get the most information from attributes while keeping the tree compact. Class decomposition was used to split classes withcomplex structure into areas of high density and improve the classifier's ability to separate the classes.

The proposed methods improved the performance of decision tree classifiers in most cases, although there were some exceptions when performance decreased when class decomposition was applied. This can be explained by specific structure of classes in these data sets – the subclasses of different classes are closer and therefore harder to separate than the original classes. Also the shape of classes or subclasses has a great impact on performance of decision tree classifiers when class decomposition was applied.

The experiments show that there is not enough information about choosing the right algorithm and the influence of the proposed methods. In some experiments the method that showed superior results without the use of these efficiency enhancement methods was outperformed by other classification methods when class decomposition was applied.

The experiments also showed that the heuristics for choosing the best cluster combinations in class decomposition proposed by other authors (like choosing combinations with the highest cardinality) are not always correct. The best evaluation of cluster combinations should include analysis of overlapping with other classes that has a great impact on later classification using the chosen subclasses.

The use of ontologies in decision tree classifier design can improve classification accuracy and also produce more compact trees. There is no best approach to building a taxonomy that fits all data sets equally. In data sets where the underlying interactions between attribute values are well known the manually built taxonomies (taxonomies built by experts) are the best choice but if the knowledge about the data is not good enough the best results can be achieved using AVT-Learner taxonomies that explore the interactions in data using statistical approach.

The combination of both methods does not give more improvement to efficiency because both methods use information about data structure to improve the performance of classification algorithms and the meta-data that they use appear to be similar and interfere with the other method.

REFERENCES

[1] **L. Breiman**, **J. Friedman**, **R. Olshen**,**C. Stone**.*Classification and Regression Trees*. Belmont, CA: Wadsworth Int. Group, 1984.

[2] **J. R. Quinlan**. "Induction of Decision Trees." Machine learning, Vol. 1, Issue 1, pp. 81-106, 1986.
[3] **E. Simoudis**, **J. Han**, **U. Fayyad**, Eds.*Second International Conference on Knowledge Discovery and Data Mining*, August 2 - 6, 1996,Portland, Oregon, USA. Portland: AAAI Press, 1996.
[4] **P. B. Brazdil**,**C. Soares**,**J. P. Da Costa**. "Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results." Machine Learning, Vol. 50, pp. 251-277, 2003.
[5] **R. Vilalta**. "Identifying and Characterizing Class Clusters to Explain Learning Performance." Journal of Systems and Software, Vol. 80, Issue 1, pp. 63-73, 2006.
[6] **J. Peckham**, Ed. *SIGMOD Dataming and Knowledge Discovery Workshop*, May 11, 1997, Tucson, Arizona, USA. USA: ACM Press, 1997.
[7] **S. Koenig**,**R. C. Holte**, Eds. *Fifth InternationalSymposium on Abstraction, Reformulation and Approximation*, August 2-4, 2002, Kananaskis, Alberta, Canada. Berlin, Heidelberg: Springer-Verlag, 2002.
[8] **A. K .H. Tung**, **Q. Zhu**, **N. Ramakrishnan**, **O. R. Zaïane**, **Y. Shi**, **Chr. W. Clifton**,**X. Wu**,Eds. *Fourth IEEE International Conference on Data Mining*, November 1-4, 2004, Brighton, UK. Washington: IEEE Computer Society, 2004.
[9] **Zhang J.**, **Silvescu A.**, **Honavar V**. Ontology-Driven Induction of Decision Trees at Multiple Levels of Abstraction. / Zhang J., Silvescu A., Honavar V. // Proceedings of Symposium on Abstraction, Reformulation and Approximation 2002, Kananaskis, Alberta, Canada, August 2-4, 2002. – Berlin: Springer Berlin/Heidelberg, 2002. – p. 316-323.
[10] **A. Asuncion**,**D. J. Newman**. *UCI Machine Learning Repository* [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science, 2007.

Inese Polakais first year graduate student at Riga Technical University. She finished her Master studies at Riga Technical University majoring in Information Technology in 2010 taking a Mg. sc. ing. degree.
Her research interests include machine learning methods and classification tasks in bioinformatics, decision tree classifiers, classifier efficiency improvement methods, use of ontology in machine learning, ontology based classifier design, descriptive statistics,and exploratory data analysis.

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

Inese Poļaka, Arkādijs Borisovs. Datu struktūras īpašību izmantošana lēmumu koku klasifikatoru projektēšanā
Rakstā tiek apskatītas lēmumu koku klasifikatoru veiktspējas uzlabošanas metodes ar mērķi izveidot efektīvākus un kompaktākus klasifikācijas kokus, izmantojot algoritmus C4.5 un CART. Tiek apskatītas divas klasifikatoru efektivitātes uzlabošanas metodes – klašu dekompozīcija un uz ontoloģijas balstīta lēmumu koku būvēšana.
Klašu dekompozīcijā tiek izmantota meta-informācija par klases iekšējo struktūru, lai pielāgotu lēmumu kokus sarežģītajai klašu struktūrai. Metodes pamatā tiek veikta hierarhiskā klasterizācija katras klases iekšienē, lai noskaidrotu blīvuma apgabalus (klasterus, kas noteikti izmantojot hierarhisko aglomeratīvo klasterizāciju), un tad šie blīvuma apgabali tiek izmantoti kā apakšklases klasifikācijas uzdevumā, kura risināšanai tiek pielietoti algoritmi C4.5 un CART.
Uz ontoloģijas balstītā koku būvēšanā tiek izmantota informācija par atribūta vērtību savstarpējām sakarībām, kas tiek attēlota atribūtu vērtību taksonomijā (hierarhiska atribūtu vērtību struktūra, kas balstās uz atribūtu vērtību savstarpējām līdzībām). Taksonomiju veidošanai eksperimentu gaitā tika izmantotas manuālā un AVT-Learner metodes. Pēc taksonomiju izveidošanas uz to pamata tiek būvēti lēmumu koki, kas var izmantot atribūtu vērtības dažādos abstrakcijas līmeņos, būvējot precīzākus un kompaktākus kokus.
Tika veikti praktiski eksperimenti, izmantojot abas piedāvātās metodes un Internetā par brīvu pieejamās datu bāzes, lai izpētītu abu metožu darbības principus. Eksperimenti pierādīja, ka šādu metožu pielietošana, klasifikatoru veidošanā izmantojot informāciju par datu struktūru, ievērojami uzlabo klasifikatoru veiktspēju un tiek būvēti kompaktāki lēmumu koki. Abu metožu vienlaicīga izmantošana nedeva papildus ieguvumus klasifikācijas precizitātē bet lielākoties tika samazināti koku lielumi.

Инесе Поляка, Аркадий Борисов. Использование характеристик структуры данных для построения классификаторовдеревьев решений
Работа посвящена методам повышения эффективности классификаторов деревьев решений с целью создания более эффективных и компактных деревьев классификации с использованием алгоритмов C4.5 и CART. Рассмотрены два метода повышения эффективности классификаторов - декомпозиция классов и построение деревьев решений на основе онтологии.
В декомпозиции классов используется мета-информация о внутренней структуре классов, чтобы адаптировать деревья решений к сложной структуре классов. В основе метода лежит иерархическая кластеризация в пределах каждого класса для определения областей плотности (кластеров, определенных иерархической агломеративной кластеризацией); затем области плотности используются в качестве подклассов в задачеклассификации, для решения которой используются алгоритмы C4.5 и CART.
В построении деревьев решений на основе онтологии используется информация о взаимосвязи между значениями атрибута, которая отображается в таксономии значений атрибута (иерархическая структура значений атрибута на основе сходства значений). Для построения таксономийв ходе эксперимента были использованы методы мануального проектированияи AVT-Learner. После создания таксономий на их основе строятся деревья решений, которые могут использовать значения атрибутов на разных уровнях абстракции для построения более точных и компактных деревьев.
Для изучения работы методов была проведенасерия экспериментов с использованием предложенных методов и баз данных, свободно доступных в Интернете. Эксперименты показали, что использование методов, которые используют информацию о структуре данных при построении классификаторов, существенно повышает производительность классификаторов, и деревья решений получаются более компактными. Оба метода, использованные в совокупности, не дали дополнительногоулучшения точности классификации, но в большинстве случаев были снижены размеры дерева.