Local Cost Sensitive Learning for Handling Imbalanced Data Sets

M. G. Karagiannopoulos*, D. S. Anyfantis*, S. B. Kotsiantis* and P. E. Pintelas*
*Educational Development Laboratory,
Department of Mathematics, University of Patras, Hellas

***Abstract*—Many real-world data sets exhibit skewed class distributions in which almost all cases are allotted to a class and far fewer cases to a smaller, usually more interesting class. A classifier induced from an imbalanced data set has, typically, a low error rate for the majority class and an unacceptable error rate for the minority class. This paper firstly provides a systematic study on the various methodologies that have tried to handle this problem. Finally, it presents an experimental study of these methodologies with a proposed local cost sensitive technique and it concludes that such a framework can be a more effective solution to the problem. Our method seems to allow improved identification of difficult small classes in predictive analysis, while keeping the classification ability of the other classes in an acceptable level.**

Keywords: *supervised machine learning, imbalanced data sets, local learning.*

I. INTRODUCTION

Typically classifiers are expected to be able to generalize over unseen instances of any class with equal accuracy. In many applications classifiers are faced with imbalanced data sets, which can cause the classifier to be biased towards one class. This bias is the result of one class being heavily under represented in the training data compared to the other classes. It can be attributed to the way in which classifiers are designed. Inductive classifiers are typically designed to minimize errors over the training instances. Learning algorithms, because of the fact that the cost of performing well on the over-represented class outweighs the cost of poor performance on the smaller class, can ignore classes containing few instances. Moreover, the difficulty to distinguish between the rare cases (i.e., true exceptions) and noise is also responsible for poor performance on the minority class (Kotsiantis et al., 2006).

For a number of application domains, a huge disproportion in the number of cases belonging to each class is common. For instance, in detection of fraud in telephone calls (Fawcett and Provost, 1997) and credit card transactions (Chan et al., 1999), the number of legitimate transactions is much higher than the number of fraudulent transactions. Moreover, in direct marketing (Ling and Li, 1998), it is common to have a small response rate (about 1%) for most marketing campaigns. Other examples of domains with intrinsic imbalance can be found in the literature such as rare medical diagnoses (Witten and Frank, 2000) and oil spills in satellite images (Kubat et al., 1998). Thus, learning with skewed class distributions is an important issue in supervised learning.

The machine learning community has mainly addressed the issue of class imbalance in two ways. One is to assign distinct costs to training instances (Domingos, 1999). The other is to re-sample the original dataset, either by oversampling the minority class and/or under-sampling the majority class (Kubat & Matwin, 1997; Japkowicz & Stephen, 2002). Although many methods for coping with imbalanced data sets have been proposed, still remain open questions. One open question is whether simply changing the distribution skew can improve predictive performance systematically. Another question is whether we can tailor learning algorithms to this special learning environment so that the accuracy for the extreme class values can be improved.

To handle the problem, we developed a local cost sensitive technique. The effectiveness of our approach is evaluated over eight imbalanced datasets using the C4.5 (Quinlan, 1993) and Naive Bayes (Domingos & Pazzani, 1997) as classifiers and the geometric mean of accuracies as performance measure (Kubat et al., 1998).

In the following section we briefly describe the used machine learning techniques and we explain the reasons for their poor performance in imbalance data sets. Section 3 reviews the attempts for handling imbalanced data sets, while section 4 presents the details of our approach. Section 5 presents experimental results comparing our approach to other approaches. Finally, section 6 discusses the results and suggests directions for future work.

II. LEARNING TECHNIQUES AND ALGORITHMS

A small imbalance in the class distribution is not serious, but when some classes are heavily under-represented, many machine-learning methods are likely to run into problems. Cases belonging to small classes are lost among the more frequent cases during learning, and, consequently, classifiers such as decision trees and Bayesian networks are unable to classify correctly new unseen cases from the minority classes. In the following subsections we briefly describe decision trees and Bayesian networks and we refer to the reasons for their poor performance in minority class of imbalance data sets.

A. Decision trees

Murthy (1998) provides a recent overview of existing work in decision trees and a taste of their usefulness to the newcomers in the field of machine learning. Decision

trees are trees that classify instances by sorting them based on feature values. Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can take. Instances are classified starting at the root node and sorting them based on their feature values.

The feature that best divides the training data would be the root node of the tree. The same process is then repeated on each partition of the divided data, creating sub trees until the training data are divided into subsets of the same class. At each level in the partitioning process a statistical property known as *information gain* is usually used to determine which feature best divides the training instances (Quinlan, 1993).

A decision tree, or any learned hypothesis *h*, is said to overfit training data if there exists another hypothesis *h′* that has a larger error than *h* when tested on the training data, but a smaller error than *h* when tested on the entire data set. There are two common approaches that decision tree induction algorithms can use to avoid overfitting training data:

1. Stop the training algorithm before it reaches a point in which it perfectly fits the training data,
2. Prune the induced decision tree.

For the scope of our study the most well-known decision tree algorithm - C4.5 (Quinlan, 1993) – was used. One of the latest researches that compare decision trees and other learning algorithms is made by (Tjen-Sien Lim et al. 2000) and shows that the mean error rates of most algorithms are sufficiently similar and that their differences are statistically insignificant. But, unlike error rates, there are huge differences between the training times of the algorithms. C4.5 has one of the best combinations of error rate and speed.

B. Naive Bayes

Probabilistic classifiers and, in particular, the naive Bayes classifier, are among the most popular classifiers in the machine learning community and they are used increasingly in many applications. Naive Bayes (NB) classifier is the simplest form of Bayesian network (Jensen, 1996) since it captures the assumption that every feature (a1, a2 ... ai) is independent of the rest of the features, given the state of the class feature V.

The formula used by the Naive Bayes classifier is:

$$v_{\max} = \max_{v_j \in V} P(v_j) \prod_i P(a_i \mid v_j). \qquad (1)$$

where V is the target output of the classifier and P(ai|vj) and P(vi) can be calculated based on their frequency in the training data.

For numerical features, one can model the component marginal distributions in a wide variety of ways. The simplest would be to adopt some parametric form e.g. marginal Gaussian estimators. The assumption of independence is clearly almost always wrong. However, Friedman (1997) explains why simple naive Bayes method remains competitive, even though it provides very poor estimates of the true underlying probabilities. Good probability estimates are not necessary for good classification; similarly, low classification error does not imply that the corresponding class probabilities are being estimated (even remotely) accurately. In addition, Domingos & Pazzani (1997) performed a large-scale comparison of Naive Bayes classifier with state-of-the-art algorithms for decision tree induction and instance-based learning on standard benchmark datasets, and found it sometimes to be superior to each of the other learning schemes even on datasets with substantial feature dependencies.

The extreme skewness in class distribution is problematic for Naïve Bayes. The prior probability of the majority class overshadows the differences in the attribute conditional probability terms.

III. Review Of Existing Techniques For Handling Imbalanced Data Sets

A classifier induced from an imbalanced data set has, typically, a low error rate for the majority class and an unacceptable error rate for the minority class. The problem arises when the misclassification cost for the minority class is much higher than the misclassification cost for the majority class. In this situation, it is important to accurately classify the minority class in order to reduce the overall cost.

A simple method that can be used to imbalanced data sets is to reweigh training instances according to the total cost assigned to each class (Domingos, 1998). The idea is to change the class distributions in the training set towards the most costly class. Japkowicz (2000) discussed the effect of imbalance in a dataset. She mainly evaluated two strategies: under-sampling and resampling. Two resampling methods were considered. Random resampling consisted of resampling the smaller class at random until it consisted of as many samples as the majority class and "focused resampling" consisted of resampling only those minority instances that occurred on the boundary between the minority and majority classes. Random under-sampling was also considered, which involved under-sampling the majority class samples at random until their numbers matched the number of minority class samples; focused under-sampling involved under-sampling the majority class samples lying further away. She noted that both the sampling approaches were effective, and she also observed that using the sophisticated sampling techniques did not give any clear advantage in the domain considered.

Kubat and Matwin (1997) also selectively under-sampled the majority class while keeping the original population of the minority class with satisfied results. Batista et al. (2000) used a more sophisticated under-sampling technique in order to minimize the amount of potentially useful data. The majority class instances are classified as "safe", "borderline" and "noise" instances. Borderline and noisy cases are detected using Tomek links, and are removed from the data set. Only safe majority class instances and all minority class instances are used for training the learning system. A Tomek link can be defined as follows: given two instances x and y

belonging to different classes, and be d(x, y) the distance between x and y. A (x, y) pair is called a Tomek link if there is not a case z, such that d(x, z) < d(x, y) or d(y, z) < d(y, x).

Both, under-sampling and over-sampling, have known drawbacks. Undersampling can throw away potentially useful data, and over-sampling can increase the likelihood of occurring overfitting, since most of over-sampling methods make exact copies of the minority class instances. In this way, a symbolic classifier, for instance, might construct rules that are apparently accurate, but actually, cover one replicated instance.

Another approach is that of Ling and Li (1998). They combined over-sampling of the minority class with under-sampling of the majority class. However, the over-sampling and under-sampling combination did not provide significant improvement. Chawla et al. (2002) propose an over-sampling approach in which the minority class is over-sampled by creating ``synthetic'' instances rather than by over-sampling with replacement with better results.

Changing the class distribution is not the only way to improve classifier performance when learning from imbalanced data sets. A different approach to incorporating costs in decision-making is to define fixed and unequal misclassification costs between classes. Cost model takes the form of a cost matrix, where the cost of classifying a sample from a true class j to class i corresponds to the matrix entry λ_{ij}. This matrix is usually expressed in terms of average misclassification costs for the problem. The diagonal elements are usually set to zero, meaning correct classification has no cost. We define conditional risk for making a decision α_i as:

$$R(a_i \mid x) = \sum_i \lambda_{ij} P(v_j \mid x) . \qquad (2)$$

The equation states that the risk of choosing class i is defined by fixed misclassification costs and the uncertainty of our knowledge about the true class of x expressed by the posterior probabilities. The goal in cost-sensitive classification is to minimize the cost of misclassification, which can be realized by choosing the class (vj) with the minimum conditional risk.

An alternative to balancing the classes is to develop a learning algorithm that is intrinsically insensitive to class distribution in the training set. An example of this kind of algorithm is the SHRINK algorithm (Kubat et al., 1998) that finds only rules that best summarize the positive instances (of the small class), but makes use of the information from the negative instances. MetaCost (Domingos, 1999) is another recently proposed method for making a classifier cost-sensitive. The procedure begins to learn an internal cost-sensitive model by applying a cost-sensitive procedure, which employs a base learning algorithm. Then, MetaCost procedure estimates class probabilities using bagging and then re-labels the training instances with their minimum expected cost classes, and finally relearns a model using the modified training set.

Furthermore, Schapire et al. (1998) gave different weights for false positives and false negatives to apply AdaBoost than bagging in text-filtering. AdaBoost uses a base classifier to induce multiple individual classifiers in sequential trials, and a weight is assigned to each training instance. At the end of each trial, the vector of weights is adjusted to reflect the importance of each training instance for the next induction trial. This adjustment effectively increases the weights of misclassified instances and decreases the weights of the correctly classified instances. Fan et al. (1999) also proposed a similar technique. Their intuition was that in addition to assigning high initial weights to costly instances, the weight-updating rule should also take cost into account and increase the weights of costly misclassification more but decrease the weights of costly correct classification less.

IV. Proposed Technique

The proposed technique is based on the previous referred cost-sensitive technique; however, we do not apply this technique globally but locally. If all training instances are taken into account when classifying a new test case, the classifier works as a global method, while when the nearest training instances are taken into account, the classifier works as a local method, since only data local to the area around the testing instance contribute to the classification.

Generally, local methods have significant advantages when the probability measure defined on the space of symbolic features for each class is very complex, but can still be described by a collection of less complex local approximations (Atkeson et al., 1997). The proposed algorithm builds the cost sensitive classifier for each point to be estimated, taking into account only a subset of the training points. This subset is chosen on the basis of the preferable distance metric between the testing point and the training point in the input space. In other words, the proposed technique consists of four steps :

1. Determine a suitable distance metric.
2. Find the k-nearest neighbours using the selected distance metric.
3. Build cost sensitive learning model using as training instances these k instances.
4. The estimates given by local classifier are the final class.

The proposed technique has some free parameters such as the distance metric. In our experiments, we used the most well known -Euclidean similarity function- as distance metric. The proposed algorithm also requires choosing the value of k. There are several ways to do this. A first, simple solution is to fix k a priori before the beginning of the learning process. However, the best k for a specific dataset is obviously not the best one for another dataset (Frank et al, 2003). A second, more time-consuming solution is therefore to determine this best k automatically through the minimization of a cost criterion. The idea is to apply a model selection process

upon which the different hypothesis that can be built. One way to do that is to evaluate the geometric mean of the accuracies on a test set and thus keep as k the value for which the error is the least.

A key feature of our method is that it does not require any modification of the underlying learning algorithm; it is applicable as long as the classifier produces class probability estimates. In the following section, we empirically evaluate the performance of our approach with the other well known techniques using a decision tree and a Bayesian model as base learners.

V. EXPERIMENTS

In Table I, there is a brief description of the data sets that we used for our experiments. Except for the "eap" data set, all were drawn from the UC Irvine Repository (Blake, Keogh & Merz, 1998). Eap data is from Hellenic Open University and was used in order to determine whether a student is about to drop-out or not (Kotsiantis et al., 2003). The data sets from UC Irvine Repository are from domains of: image recognition (ionosphere), medical diagnosis (breast-cancer, diabetes, haberman, hepatitis, sick) and commodity trading (credit-g).

The performance of machine learning algorithms is typically evaluated using predictive accuracy. However, this is not appropriate when the data is imbalanced. A simple but effective strategy for classification would be to simply assign the majority class to all unknown instances.

A classifier's performance of two class problems can be separately calculated for its performance over the positive instances (denoted as α+) and over the negative instances (denoted as α-). The true positive rate (α+) or sensitivity is the fraction of positive instances predicted correctly by the model. Similarly, the true negative rate (α-) or specificity is the fraction of negative instances predicted correctly by the classifier.

Kubat et al. (1998) propose the geometric mean of the accuracies: $g = \sqrt{a^+ \times a^-}$ for imbalanced data sets. The basic idea behind this measure is to maximize the accuracy on both classes. Moreover, ROC curves (Receiving Operator Characteristic) provide a visual representation of the trade off between true positives (α+) and false positives (α-). These are plots of the percentage of correctly classified positive instances α+ with respect to the percentage of incorrectly classified negative instances α- (Provost and Fawcett, 2001). If the model is perfect, then its area under the ROC curve would equal to 1. If the model corresponds to random guessing, then its area under ROC curve would be equal to 0.5. Anything less than 0.5 would be worse than random guessing.

TABLE I.
DESCRIPTION OF THE DATA SETS

Data sets	Instances	Categorical features	Numerical features	Instances of minority class	Classes
breast-cancer	286	9	0	85	2
credit-g	1000	13	7	300	2
Diabetes	768	0	8	268	2
Haberman	306	0	3	81	2
Hepatitis	155	13	6	32	2
Ionosphere	351	34	0	126	2
Eap	344	11	0	122	2
Sick	3772	22	7	231	2

The most popular method for plotting a ROC curve is threshold variation (Witten and Frank, 2000): given a set of test instances and a classifier, the numeric output for each test instance is computed, and the instances are ordered according to the corresponding numeric prediction. Then, for each instance, a (1-α+, α+) point is obtained, that is, considering that instances before it are classified as positive and instances after it are classified as negative. Subsequent (1-α+, α+) points are linked. The method for plotting a ROC curve is closely related to a method for making algorithms cost-sensitive, that we call Threshold method (Witten and Frank, 2000). This method uses a threshold so as to maximize the given performance measure in the curve.

The problem of determining which proportion of positive/negative examples is the best for learning is an open problem of learning from imbalanced data sets. In order to make the experiment more realistic, parameters of the cost models were not optimized for each data set, the relationship between false negative and false positive costs was chosen to be the inverse of the assumed prior to compensate for the imbalanced priors.

Classification ability of the learning methods in our experiments was measured with geometric mean of the accuracies. In the following Tables, win (v) indicates that the specific method along with the learning algorithm performed statistically better than the single classifier according to t-test with $p<0.05$. Loss (*) indicates that the specific method along with the learning algorithm performed statistically worse than the single classifier according to t-test with $p<0.05$. In all the other cases, there is no significant statistical difference between the results.

In Table 2, one can see the comparisons of the proposed technique with other attempts that have tried to obtain the best performance of a given imbalance data set using Naive Bayes (NB) as base classifier. Four well-known algorithms were used for the comparison: Threshold method (Witten and Frank, 2000), Cost Sensitive method (Domingos, 1998), Adaboost cost sensitive method (Schapire et al., 1998) and Metacost algorithm (Domingos, 1999). We also present the accuracy of the simple Bayes algorithm as borderline. It must be mentioned that we used the free available source code for these methods by (Witten and Frank, 2000) for our experiments. In the Table 2 except for geometric mean we also present the true-positive rate, and true-negative rate. It must be mentioned that positive class for our experiments is the majority class. In the last row of the Table 2, the mean value of the geometric means is also calculated in all data sets.

In general, all the tested techniques give better results than the single Naive Bayes. The most remarkable improvement is from our technique, even though the

Threshold method gives, on average, the best accuracy in the minority class In general, all the tested techniques give better results than the single Naive Bayes. The most remarkable improvement is from our technique, even though the Threshold method gives, on average, the best accuracy in the minority class. The Metacost cannot improve the results of the NB as his author suspects. It must be mentioned that for Naïve Bayes classifier, modifying the decision boundary (Cost Sensitive method) is equivalent to reweighing training instances so as the relationship between false negative and false positive costs to be the inverse of the imbalanced priors. Moreover, Adaboost cost sensitive method cannot give better results than Cost Sensitive, even though it is a more time consuming technique.

TABLE II.
ACCURACY ON MAJORITY CLASS (A+), ACCURACY ON MINORITY CLASS (A-) AND GEOMETRIC MEAN (G) WITH NB AS BASE CLASSIFIER

Data sets		LCSNB	ThresNB	CostNB	AdabcosNB	MetacostNB	NB
breast-cancer	g	0.66	0.63	0.66	0.63	0.65	0.6*
	α^+	0.71	0.62 *	0.74	0.72	0.79 v	0.85v
	α^-	0.61	0.65 v	0.58	0.56 *	0.54 *	0.43*
credit-g	g	0.73	0.71	0.72	0.71	0.66 *	0.65*
	α^+	0.77	0.69 *	0.75	0.75	0.77	0.86v
	α^-	0.7	0.74 v	0.69	0.67	0.57 *	0.49*
diabetes	g	0.74	0.72	0.73	0.73	0.70 *	0.71
	α^+	0.75	0.65 *	0.78	0.77	0.75	0.84v
	α^-	0.73	0.8 v	0.68 *	0.69 *	0.66 *	0.6 *
haberman	g	0.58	0.59	0.56	0.56	0.57	0.44*
	α^+	0.69	0.64 *	0.89 v	0.88 v	0.87 v	0.94v
	α^-	0.49	0.55 v	0.35 *	0.36 *	0.38 *	0.21*
hepatitis	g	0.84	0.76 *	0.8 *	0.78 *	0.81 *	0.78*
	α^+	0.93	0.87 *	0.83 *	0.86 *	0.79 *	0.87*
	α^-	0.75	0.67 *	0.78	0.71 *	0.84 v	0.7*
ionosphere	g	0.9	0.88	0.82 *	0.91	0.77*	0.83
	α^+	0.89	0.93 v	0.78 *	0.93 v	0.68 *	0.8*
	α^-	0.92	0.81 *	0.87 *	0.9	0.88 *	0.86*
eap	g	0.86	0.83	0.85	0.83	0.85	0.84
	α^+	0.89	0.86	0.87	0.85 *	0.88	0.9
	α^-	0.83	0.81	0.83	0.82	0.83	0.78*
sick	g	0.86	0.76*	0.86	0.87	0.8*	0.86
	α^+	0.90	0.98 v	0.82 *	0.88	0.73 *	0.94v
	α^-	0.83	0.59 *	0.9 v	0.86	0.87 v	0.78*
MEAN	g	0.77	0.74	0.75	0.75	0.73	0.71

In Table 3, one can see the comparisons of the proposed technique with other attempts that have tried to obtain the best performance of a given imbalance data sets using C4.5 as base classifier. The same four well-known techniques for handling imbalanced data sets were also used for this comparison.

Likewise with the previous experiment, our method has better performance than the other techniques. However, Metacost has really better performance with C4.5 than NB. It must also be mentioned that Threshold method gives worst performance than single C4.5. Adaboost cost sensitive method, as in the previous experiment, cannot give better results than reweighing method even though it uses more time for training.

VI. CONCLUSION

Several aspects may influence the performance achieved by a classifier created by a Machine Learning system. One of these aspects is related to the difference between the numbers of instances belonging to each class. When this difference is large, the learning system may have difficulties to learn the concept related to the minority class.

In this work, we survey some methods proposed by the Machine Learning community to solve the problem, we discuss some limitations of these methods and we propose a local cost sensitive technique as a more effective solution to problem. Our method allows improved identification of difficult small classes in predictive analysis, while keeping the classification ability of the other classes in an acceptable level.

For large datasets, the benefit of local cost sensitive models is somewhat offset by the cost of storing and querying the training dataset for each test set instance. For this reason, in a following project we will focus on the problem of reducing the size of the stored set of instances while trying to maintain or even improve generalization performance by avoiding noise and over-fitting. In (Brighton and Mellish, 2002), numerous instance selection methods that can be combined with the proposed technique can be found. In a future research project, we will also examine the efficiency of our approach in multi-class data sets.

VII. REFERENCES

[1] Aha, D. (1997). Lazy Learning. Dordrecht: Kluwer Academic Publishers.

[2] Atkeson, C. G., Moore, A.W., & Schaal, S.: Locally weighted learning. Artificial Intelli-gence Review 11 (1997) 11–73.

[3] Batista G., Carvalho A., Monard M. C. (2000), Applying One-sided Selection to Unbalanced Datasets. In O. Cairo, L. E. Sucar, and F. J. Cantu, editors, MICAI 2000, pages 315–325. Springer-Verlag.

TABLE III.
ACCURACY ON MAJORITY CLASS (A^+), ACCURACY ON MINORITY CLASS (A^-) AND GEOMETRIC MEAN (G) WITH C4.5 AS BASE CLASSIFIER

Data sets		LCSC4.5	ThresC4.5	CostC4.5	Adabcos C4.5	Metacost C4.5	C4.5
breast-cancer	g	0.6	0.45*	0.5 *	0.56 *	0.55 *	0.5 *
	α+	0.8	0.8	0.85 v	0.77	0.84 v	0.95 v
	α-	0.45	0.25 *	0.3 *	0.41 *	0.36 *	0.26 *
credit-g	g	0.65	0.64	0.61*	0.62	0.64	0.58 *
	α+	0.73	0.7	0.82 v	0.81 v	0.76	0.85 v
	α-	0.58	0.58	0.46 *	0.47 *	0.54 *	0.4 *
diabetes	g	0.71	0.7	0.72	0.67	0.73	0.7
	α+	0.76	0.69 *	0.78	0.79	0.78	0.82v
	α-	0.66	0.71 v	0.67	0.57 *	0.67	0.6*
haberman	g	0.62	0.56 *	0.58 *	0.57 *	0.62	0.52 *
	α+	0.65	0.61 *	0.66	0.76 v	0.76 v	0.85 v
	α-	0.59	0.51 *	0.51 *	0.43 *	0.52 *	0.32 *
hepatitis	g	0.73	0.62 *	0.64 *	0.7	0.68 *	0.58 *
	α+	0.80	0.78	0.86 v	0.9 v	0.83	0.9 v
	α-	0.66	0.49 *	0.48 *	0.55 *	0.56 *	0.37 *
ionosphere	g	0.89	0.88	0.88	0.9	0.9	0.88
	α+	0.94	0.95	0.94	0.94	0.98 v	0.94
	α-	0.84	0.81	0.82	0.86	0.82	0.82
eap	g	0.80	0.69 *	0.83 v	0.79	0.82	0.83 v
	α+	0.81	0.91 v	0.94 v	0.85 v	0.89 v	0.94 v
	α-	0.78	0.53 *	0.74 *	0.74 *	0.76	0.74
sick	g	0.96	0.92 *	0.96	0.95	0.96	0.93
	α+	0.99	0.99	0.99	1	0.98	0.99
	α-	0.94	0.85 *	0.92	0.9 *	0.95	0.87 *
MEAN	g	0.75	0.68	0.72	0.72	0.74	0.69

[4] Blake, C., Keogh, E. & Merz, C.J. (1998). UCI Repository of machine learning databases [http:// www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California.

[5] Brighton, H. Mellish, C., Advances in Instance Selection for Instance-Based Learning Algo-rithms, Data Mining and Knowledge Discovery, 6, 153–172, 2002.

[6] Chawla N., Bowyer K., Hall L., Kegelmeyer W. (2002), SMOTE: Synthetic Minority Over-sampling Technique, Journal of Artificial Intelligence Research 16, 321 - 357.

[7] Domingos P. (1998), How to get a free lunch: A simple cost model for machine learning applications. Proc. AAAI-98/ICML98, Workshop on the Methodology of Applying Machine Learning, pp1-7.

[8] Domingos P. & Pazzani M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. Machine Learning, 29, 103-130.

[9] Domingos, P. (1999). MetaCost: A General Method for Making Classifiers Cost-Sensitive. Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining, 155-164. ACM Press.

[10] Fan, W., Stolfo, S.J., Zhang, J. & Chan, P.K. (1999). AdaCost: Misclassification costsensitive boosting. Proceedings of the Sixteenth International Conference on Machine Learning, 97-105. San Francisco: Morgan Kaufmann.

[11] Fawcett T. and Provost F. (1997), Adaptive Fraud Detection. Data Mining and Knowledge Discovery, 1(3):291–316.

[12] Frank, E., Hall M. and Pfahringer, B., Locally weighted naive Bayes. Proc. of the 19th Conference on Uncertainty in Artificial Intelligence. Acapulco, Mexico. Morgan Kaufmann, 2003.

[13] Friedman J. H. (1997), On bias, variance, 0/1-loss and curse-of-dimensionality. Data Mining and Knowledge Discovery, 1: 55-77.

[14] Japkowicz N. (2000), The class imbalance problem: Significance and strategies. In Proceedings of the International Conference on Artificial Intelligence, Las Vegas.

[15] Japkowicz N. and Stephen, S. (2002), The Class Imbalance Problem: A Systematic Study Intelligent Data Analysis, Volume 6, Number 5.

[16] Kotsiantis S., Pierrakeas C., Pintelas P. (2003), Preventing student dropout in distance learning systems using machine learning techniques, AI Techniques In Web-Based Educational Systems at Seventh International Conference on Knowledge-Based Intelligent Information & Engineering Systems, 3-5 September 2003.

[17] Kotsiantis S., Kanellopoulos, D. Pintelas, P. (2006), Handling imbalanced datasets: A review, GESTS International Transactions on Computer Science and Engineering, Vol.30 (1), pp. 25-36.

[18] Kubat, M. and Matwin, S. (1997), 'Addressing the Curse of Imbalanced Data Sets: One Sided Sampling', in the Proceedings of the Fourteenth International Conference on Machine Learning, pp. 179-186.

[19] Kubat, M., Holte, R. and Matwin, S. (1998), 'Machine Learning for the Detection of Oil Spills in Radar Images', Machine Learning, 30:195-215.

[20] Ling, C., & Li, C. (1998). Data Mining for Direct Marketing Problems and Solutions. In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98) New York, NY. AAAI Press.

[21] Murthy (1998), Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey, Data Mining and Knowledge Discovery, 2, 345–389 (1998), Kluwer Academic Publishers.

[22] Provost, F. and Fawcett, T. (2001). Robust Classification for Imprecise Environments", Machine Learning, 42, 203–231.

[23] Quinlan J.R. (1993), C4.5: Programs for machine learning. Morgan Kaufmann, San Francisco.

[24] Schapire R., Singer Y. and Singhal A. (1998). Boosting and Rochhio applied to text filtering. In SIGIR'98.

[25] Tjen-Sien Lim, Wei-Yin Loh, Yu-Shan Shih (2000), A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms. Machine Learning, 40, 203–228, 2000, Kluwer Academic Publishers.

[26] Witten, I. and Frank E. (2000) Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, San Mateo, CA, 2000.