Improving the k-NN method: Rough Set in edit training set

Yailé Caballero[(1)], Rafael Bello[(2)], Delia Alvarez[(1)], Maria M. Garcia[(2)]
Yaimara Pizano[(3)]
[(1)]Department of Computer Science, Universidad of Camagüey, Cuba.
{yaile, dalvarez}@inf.reduc.edu.cu
[(2)]Department of Computer Science, Universidad Central de Las Villas, Cuba.
{rbellop, mmgarcia}@uclv.edu.cu
[(3)]Department of Computer Science, Universidad of Ciego de Avila, Cuba.

Abstract. Rough Set Theory (RST) is a technique for data analysis. In this study, we use RST to improve the performance of k-NN method. The RST is used to edit and reduce the training set. We propose two methods to edit training sets, which are based on the lower and upper approximations. Experimental results show a satisfactory performance of k-NN method using these techniques.

1 Introduction

A major goal of Machine learning is the classification of previously unseen examples. Beginning with a set of examples, the system learns how to predict the class of each one based on its features. Instance-based learning (IBL) is a machine learning method that classifies new examples by comparing them to those already seen and are in memory. This memory is a Training Set (TS) of preclassified examples, where each example (also called object, instance or case) is described by a vector of features or attribute values. A new problem is solved by finding the nearest stored example taking into account some similarity functions; the problem is then classified according to the class of its nearest neighbor. Nearest neighbor methods regained popularity after Kibler and Aha showed that the simplest of the nearest neighbor models could produce excellent results for a variety of domains. A series of improvements was introduced in the IB1 to IB5 [1]. IBL method is often faced with the problem of deciding how many exemplars to store, and what portion of the instance space it should cover.

An extension to the basic IBL paradigm consists in using the K nearest neighbors instead of just the nearest one; the class assigned is that of the majority of those K neighbors, taking into account the distance (or similarity) between the problem and each nearest neighbor. Instance-base learners are lazy in the sense that

they perform little work when learning from the TS, but extend more effort classifying new problems.

The aspects that have the most interest in the k-NN method are the reduction of the classification error and the reduction of the computational cost. The k-NN method is very sensitive to the presence of incorrectly labelled examples or objects close to the decision's boundary; incorrect instances are liable to create a region around them where new examples will also be misclassified, also they can be very sensitive to irrelevant attributes; therefore IBL methods can improve their behavior by means of an accurate selection of attributes describing the instances [1,2,3,4].

On the other hand, the search for the nearest neighbor can be a very costly task, above all, in high dimension spaces. Two elements determine the computational cost of the k-NN method: the amount of features and the amount of objects. A major problem of instance-based learners is that classification time increases as more examples are added to training set (TS).

Alternative solutions to these problems have been to: (i) reduce the TS, (ii) improve the search method of the nearest neighbor, y (iii) achieve a selection process of the features or learning the relative importance of features. Rough Set Theory (RST) provides efficient tools for dealing with these alternative solutions.

2. Rough Sets Theory

Rough Sets theory was proposed by Z. Pawlak in 1982 [5]. The rough set philosophy is founded on the assumption that some information is associated with every object of the universe of discourse [6, 7]. A training set can be represented by a table where each row represents objects and each column represents an attribute. This table is called Information System; more formally, it is a pair S= (U, A), where U is a non-empty finite set of objects called the Universe and A is a non-empty finite set of attributes. A Decision System is any information system of the form SD=(U, $A \cup \{d\}$), where $d \notin A$ is the decision attribute. Classical definitions of lower and upper approximations were originally introduced with reference to an indiscernible relation which assumed to be an equivalence relation.

Let $B \subseteq A$ and $X \subseteq U$. B defines an equivalence relation and X is a concept. X can be approximated using only the information contained in B by constructing the B-lower and B-upper approximations of X, denoted by B*X and B*X respectively, where $B^*X = \{ x : [x]B \subseteq X \}$ and $B^*X = \{ x : [x]B \cap X \neq \phi \}$, and [x]B denotes the class of x according to B-indiscernible relation. The objects in B*X are sure members of X, while the objects in B*X are possible members of X.

Rough set model has several advantages to data analysis. It is based on the original data only and does not need any external information; no assumptions about data are necessary; it is suitable for analyzing both quantitative and qualitative features, and the results of rough set model are easy to understand [8].

Different authors have given their opinion about using RST in data analysis [9, 10, 11, 12, 13, 14, 15].

Two methods for editing training set based on the lower approximation and upper approximation are presented in epigraph 3.2. Experimental results show a satisfactory performance of k-NN using these techniques.

3. Editing training set by using rough set concepts

3.1 About editing training sets

The selection of examples from a domain to include in a training set is a present problem in all of the computational models for learning from examples. This selection process can be carried out either by Edition or Reduction.

The Reduction techniques pursue as objective the elimination of patterns or prototypes for decreasing the size of the learning matrix. It is about decreasing the computational work and, at times, it is disposed or ready to pay with a little less precision of the system, but with more computational efficiency.

The Editing techniques are applied to eliminate the prototypes that induce an incorrect classification, even though it is certain that they produce elimination of prototypes, their fundamental objective is to obtain a training sample of better quality to have a better precision with the system.

Various techniques of reducing the training sets have been reported. Many of them appear in [16, 17, 18, 19] with the purpose of reducing the training sets based on the nearest neighbor theory. Six new methods called DROP 1-5 and DEL are reported in [19] which can be used to reduce the number of instances in the training sets.

Hart, in 1968, made one of the first attempts to reduce the size of the training set with his Condensed Nearest Neighbor Rule (CNN). The main goal of these algorithms is the reduction of the size of the stored set of training instances while trying to maintain (or even improve) generalization accuracy. This algorithm is especially sensitive to noise, because noisy instances will usually be misclassified by their neighbors, and thus will be retained [19].

Editing algorithms from the training sample are described in [20], which are focused on the detection and elimination of noisy or atypical patterns in order to improve the classification's exactitude. Some of these are ENN (Wilson, 1972), All k-NN (Tomek, 1976) and Generalized Editing Algorithm (Koplowitz and Brown, 1978). Another editing method is Multiedit Algorithm (Devijver and Kittler, 1980) [21].

Wilson in 1972 developed the Edited Nearest Neighbor (ENN). This technique consists in applying the k-NN ($k > 1$) classifier to estimate the class label of every prototype in the training set and discard those instances whose class label does not agree with the class associated to the majority of the k neighbors. The benefits – improvements of the generalization accuracy- of Wilson's algorithm have been supported by theoretical and empirical evaluations [20]. The Repeated ENN (RENN) applies the ENN algorithm repeatedly until all instances remaining have a majority

of their neighbors with the same class, which continues to widen the gap between classes and smoothes the decision boundary.

Tomek in 1976 extended the ENN with his All k-NN method of editing. In his experiments, RENN produced higher accuracy than ENN, and the All k-NN method resulted in even higher accuracy yet. As with ENN, this method can leave internal points intact, thus limiting the amount of reduction that it can accomplish. These algorithms serve more as noise filters than serious reduction algorithms.

Koplowitz and Brown in 1978 obtained the Generalized Editing Algorithm. This is another modification of the Wilson's algorithm. Koplowitz and Brown were concerned with the possibility of too many prototypes being removed from the training set because of Wilson's editing procedure. This approach consists in removing some suspicious prototypes and changing the class labels of some other instances. Accordingly, it can be regarded as a technique for modifying the structure of the training sample (through re-labeling of some training instances) and not only for eliminating atypical instances.

In 1980 the Multiedit algorithm by Devijver and Kittler emerged. In each iteration of this algorithm, a random partition of the learning sample in N subsets is made. Then the objects from each subset are classified with the following subset applying the NN rule (the nearest neighbor rule). All the objects that were classified incorrectly from the learning sample in the previous step are eliminated and all the remaining objects are combined to constitute a new learning sample TS. If in the last I iterations no object has been eliminated from the learning sample, then end with the final learning sample TS. On the contrary, return to the initial step.

We have studied the performance of these algorithms when we use k-NN methods. The results are shown in table 1.

Aha in [1] presented some Instance-based Learning algorithms that use sample models, each concept is represented by sample set, each sample could be an abstraction of the concept or an individual instance of the concept.

Brighton and Mellish [22] introduced the batch edition method Iterative Case Filtering (ICF), this edition method is based on the reachable and coverage sets, which are based on the neighborhood and the set of associates of an object O. In [23] three edition methods were introduced: Depuration, k-NCN and iterative k-NCN. In [24] the NNEE (Neural Network Ensemble Editing) method was proposed. In [25] two edition schemes in order to reduce the runtimes of BSE without a significant reduction in the classification accuracy was proposed. In [26] a new method for selecting prototypes with Mixed Incomplete Data (MID) object description, based on an extension of the Nearest Neighbor rule was introduced.

3.2 Two methods for editing training set based on rough sets

There are two important concepts in Rough Sets Theory: Lower and Upper Approximation of decision systems. Lower approximation groups objects that certainly belong to its class, this guarantee that object inside lower approximation have no noise.

We have studied the application of rough sets for the edition of training sets. We propose two methods for editing training sets by using upper and lower

approximations. First, we use the lower approximations of classes to create the edited training set.

The basic idea of employing rough sets for editing training sets is the following: in the training set we put the examples of the initial decision system that belong to the lower approximation of each class, that is, given an application's domain with m classes and the equivalence relation B, then,

$$TS = B^*(D1) \cap B^*(D2) \cap ... \cap B^*(Dm)$$

This is equivalent to saying that the training set will be the positive region of the decision system. In this manner, objects that are incorrectly labeled or very near to the decision's boundary can be eliminated from the training set which affect the quality of the inference/deduction. Studies on multiediting presented in [27] show that isolated objects included in other regions or near to the decision's boundary are frequently eliminated.

Edit1RS Algorithm:

Step1. Construct the set B, $B \subseteq A$. Preferably, B is a reduct from the decision system.

Step2. Form the sets $Xi \subseteq U$, such that all the elements of the universe (U) that have value di in the decision's attribute are in Xi.

Step3. For each set Xi, calculate its lower approximation $B_*(Xi)$.

Step4. Construct the edited training set as the union of all the sets $B_*(Xi)$.

In the second case, we use lower approximations and boundary region of classes to create the edited training set.

In the Edit1RS method only the elements which to the lower approximations are taken into account. Also, it is important to also take into consideration those elements that are in the boundary (BNB). The Generalized Editing Algorithm consists of removing some suspicious prototypes and changing the class labels of some other instances. Accordingly, it can be regarded as a technique for modifying the structure of the training sample (through re-labeling of some training instances) and not only for eliminating atypical instances [20]. The second algorithm is proposed taking into account these ideas.

Edit2RS Algorithm:

Step1. Construct the set B, $B \subseteq A$. Preferably, B is a reduct from the decision system.

Step2. Form the sets $Xi \subseteq U$, such that all the elements of the universe (U) that have value di in the decision's attribute are in Xi.

Step3. $S = \emptyset$

Step4. For each set Xi do:

Calculate their lower approximation ($B_*(Xi)$) and upper approximation ($B^*(Xi)$).

$S = S \bigcup B_*(Xi)$.

$Ti = B^*(Xi) - B_*(Xi)$.

Step5. Calculate the union of the sets Ti. $T = \bigcup Ti$ is obtained.

Step6. Apply the Generalized Editing method to each element in T and the result is the set T'.

Step7. S = S $\cup$ T'. The edited training set is obtained as the resultant set in S.

The computational complexity of our algorithms don't surpass $O(ln^2)$, near to the ideal value of $O(n^2)$, while in the rest of the algorithms (epigraph 2.1) it is of $O(n^3)$.

The Edit1RS and Edit2RS algorithms based on the rough set theory are characterized in the following way:

Representation: Retain a subset of the original instances. In the case of the Edit2RS algorithm, this can change the class of some instances.

Direction of the search of the instances: The construction of the subset S from the training set E is achieved in batch form. In addition, the selection is achieved on a global vision of the training set not separated by decision classes.

Type of point of the space to retain: The Edit1RS algorithm retains the instances situated in the centre or interior of the classes. The Edit2RS algorithm retains these instances and others included in the boundary regions of the classes.

Volume of the reduction: The volume of the reduction depends on the amount of inconsistencies in the information system; there will always be a reduction in the training set, except if the information system is consistent.

Increment of speed: On decreasing the amount of examples the velocity of the next processing is increased.

Precision of the generalization: In the majority of cases, one of the algorithms or both of them increased significantly the efficiency of the k-NN method.

Tolerance to noise: The rough set theory offers a pattern oriented to model the uncertainty given by inconsistencies, for which it is effective in the presence of noise. In fact, the lower approximation eliminates the cases with noise.

Learning speed: The computational complexity of finding the lower approximation is O(ln2), according to [28] and [29], near to the ideal value of O(n2), and less than that of the calculation of the coverage (O(n3)), so l (amount of attributes considered in the equivalence relation) is in the majority of the cases significantly smaller than n (amount of examples).

Incremental growth: This is an incremental method so for each new instance that appears it is enough to determine if it belongs to some lower approximation of some class so that it can be added or not to the training set.

4 Experimental results

We have study the computational behavior of the algorithms when they are employed in the k-NN method. We have used decision systems constructed from the data bases that were found in: http://www.ics.uci.edu/~mlearn/MLRepository.html

The results that are shown in Table 1 and Table 2 allow you to compare the classification accuracy reached by the k-NN method using the original data bases, and the edited ones using various methods from the edition of training sets algorithms proposed by other authors and the two which are presented in section 3. We have considered two alternatives: (i) B uses all features, (ii) B is a reduct. A

reduct is a minimal set of attributes from A that preserves the partitioning of universe (and hence the ability to perform classifications) [6].

Table 1. Classification accuracy using classic methods

Name of data base (CaseCount, FeatureCount)	**Original data bases**	**ENN**	**All k-NN**	**Generalized Editing**	**MultiEdit**
Ballons-a (20,4)	60.00	60.00	100	80.00	100
Iris (150,4)	94.66	100	100	98.65	100
Hayes-Roth (133,4)	23.48	70.37	66.66	22.10	100
Bupa (345,6)	67.82	88.74	88.11	84.98	100
E-Coli (336,7)	86.60	98.28	98.06	97.70	100
Heart (270,13)	82.22	97.30	98.40	96.10	100
Pima (768,8)	73.04	94.32	96.63	90.43	100
Breast- Cancer (683,9)	96.77	99.24	99.69	99.26	100
Yeast (1484,8)	59.02	90.02	93.76	90.00	99.73
Dermatology (358,34)	97.48	98.56	100	99.14	100
Lung-Cancer (27,56)	48.14	50.00	75.00	55.55	0.00
Average	71.93	86.08	92.39	83.08	90.88

Table 2. Classification accuracy using methods based on RST

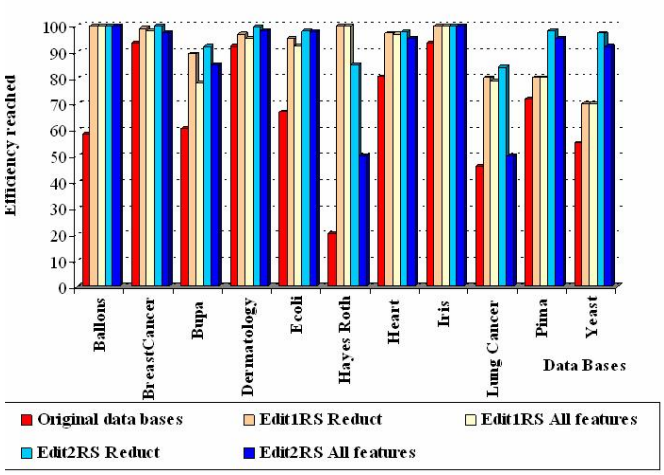
Name of data base (CaseCount, FeatureCount)	**Edit1RS**		**Edit2RS**	
	B= Reduct	**B=All features**	**B= Reduct**	**B=All features**
Ballons-a (20,4)	100	100	100	80.00
Iris (150,4)	100	98.93	98.65	100
Hayes-Roth (133,4)	85.29	100	84.61	30.27
Bupa (345,6)	76.47	76.47	82.15	82.15
E-Coli (336,7)	91.89	61.53	95.70	96.49
Heart (270,13)	95.41	89.54	93.65	92.36
Pima (768,8)	80.00	80.00	90.36	90.36
Breast- Cancer (683,9)	98.01	98.27	97.65	97.35
Yeast (1484,8)	70.00	71.42	91.01	91.74
Dermatology (358,34)	93.46	98.78	95.01	98.26
Lung-Cancer (27,56)	77.77	48.14	72.22	48.14
Average	88.03	83.92	91.00	82.47
	90.00		91.56	

The two methods based on the Rough Set Theory show superior behavior to those results obtained without editing. The achieved results with Edit1RS and Edit2RS are similar to those achieved by the ENN, Generalized Editing method, Multiedit and All-KNN methods.

By comparing the two methods based on the Rough Set Theory the superior results obtained by the Edit2RS method are appreciated. By taking an average of the best results for each one of these methods (obtained with B = reduct or B = all the

attributes) 90.00 % was obtained, while 91.56 % was obtained for Edit2RS. Results inferior to 71 % with the Edit1RS were superior to 91 % when the Edit2RS method was applied. In addition, our methods are very simple methods and are very easy of implementing.

The following graph show the classification accuracy with original data bases and the new methods.



In order to verify efficiently the described results previously the statistical test of Crossed Validation was applied, for which each data set in 5 samples was divided, at every moment 4 of them were taken to train and the other to classify, so that each one of these sets was taken to classify in one of the 5 experiments of the same. A Student Test was applied to Cross Validation results and the p-value obtained was less than 0.05 for each one of the topics to demonstrate: i) The results of Edit2RS method were better in classification than Edit1RS one, and ii) Classification Accuracy percents for Edit1RS and Edit2RS methods were better by using a reduct than working with all the features. It's possible to state that there are significant differences between the results obtained by Edit1RS method and Edit2 RS one. The two methods based on the Rough Set Theory show superior behavior to those results obtained without editing. The achieved results with Edit1RS and Edit2RS are similar to those achieved by the ENN, Generalized Editing method, Multiedit and All-KNN methods.

5. Conclusions

The possibility of applying the elements of the Rough Set Theory for the analysis of data when the k-NN method is used was presented in this paper.

A study of the possibility of applying the elements of the Rough Set Theory in data analysis when the k-NN method is used was presented in this paper. Two methods for the edition of training sets are proposed. Experimental results show that using rough sets to construct training sets to improve the work of the k-NN method is

feasible. Our methods obtained similar results to the methods with high performance and these obtained the best result in some case. Therefore, we think these new methods can be taking into account for editing training sets in k-NN method. The results obtained with the Edit1RS and Edith2RS methods were higher in the majority of cases when B is a reduct. Our methods are very simple methods and are very easy of implementing.

References

[1] Aha, D.W. Case-based Learning Algorithms. Proceedings of the DARPA Case-based Reasoning Workshop. Morgan Kaufmann Publishers. 1991.

[2] Domingos, P. Unifying instance-based and rule-based induction. International Joint Conference on Artificial Intelligence. 1995.

[3] Lopez, R.M. and Armengol, E.. Machine learning from examples: Inductive and Lazy methods.

[4] Barandela, R. et al.. The nearest neighbor rule and the reduction of the training sample size. Proceedings 9th Symposium on Pattern Recognition and Image Analysis, 1, 103-108, Castellon, España, 2001

[5] Pawlak, Z.. Rough sets. International Journal of Information & Computer Sciences 11, 341-356, 1982.

[6] Komorowski, J. Pawlak, Z. et al.. Rough Sets: A tutorial. In Pal, S.K. and Skowron, A. (Eds) Rough Fuzzy Hybridization: A new trend in decision-making. Springer, pp. 3-98. 1999.

[7] Polkowski, L.. Rough sets: Mathematical foundations. Physica-Verlag, p. 574. Berlin, Germany. 2002.

[8] Tay, F.E. and Shen, L.. Economic and financial prediction using rough set model. European Journal of Operational Research 141, pp. 641-659. 2002.

[9] Greco, S. Et al. Rough sets theory for multicriteria decision analysis. European Journal of Operational Research 129, pp. 1-47, 2001.

[10] Pal, S.K. and Skowron, A. (Eds).. Rough Fuzzy Hybridization: a new trend in decision-making. Springer-Verlag, 1999.

[11] Kohavi, R. and Frasca, B. Useful feature subsets and Rough set Reducts. Proceedings of the Third International Workshop on Rough Sets and Soft Computing. 1994.

[12] Maudal, O. Preprocessing data for neural network based classifiers: Rough sets vs Principal Component Analysis. Project report, Dept. of Artificial Intelligence, University of Edinburgh. 1996.

[13] Koczkodaj, W.W. et al.. Myths about Rough Set Theory. Comm. of the ACM, vol. 41, no. 11, nov. 1998.

[14] Zhong, N. et al.. Using Rough sets with heuristics for feature selection. Journal of Intelligent Information Systems, 16, 199-214. 2001.

[15] Pal, S.K. et al. Web mining in Soft Computing framework: Relevance, State of the art and Future Directions. IEEE Transactions on Neural Networks, 2002.

[16] Gates, G. W. The Reduced Nearest Neighbor Rule. IEEE Transactions on Information Theory, IT-18-3, pp. 431-433. 1972.

[17] Ritter, G. L. Woodruff, H. B. Lowry, S. R. Isenhour, T. L. An Algorithm for a Selective Nearest Neighbor Decision Rule. IEEE Transactions on Information Theory, 21-6, November, pp. 665-669. 1975.

[18] Lowe, David G. Similarity Metric Learning for a Variable-Kernel Classifier. Neural Computation., 7-1, pp. 72-85. 1995.

[19] Wilson, Randall. Martinez, Tony R. Reduction Techniques for Exemplar-Based Learning Algorithms. Machine Learning. Computer Science Department, Brigham Young University. USA 1998.

[20] Barandela, Ricardo; Gasca, Eduardo, Alejo, Roberto. Correcting the Training Data. Published in "Pattern Recognition and String Matching", D. Chen and X. Cheng (eds.), Kluwer, 2002.

[21] Devijver, P. and Kittler, J. Pattern Recognition: A Statistical Approach, Prentice Hall, 1982.

[22] Brighton, H. and Mellish, C. Advances in Instance Selection for Instance-Based Learning Algorithms. Data Mining and Knowledge Discovery, 6, pp. 53-172, 2002.

[23] Sánchez, J. S., Barandela, R., Marqués, A. I., Alejo, R., Badenas, J. Analysis of new techniques to obtain quality training sets. Pattern Recognition Letters, 24-7, pp. 1015-1022, 2003.

[24] Jiang Y., Zhou, Z.-H. Editing training data for kNN classifiers with neural network ensemble. In: Advances in Neural Networks, LNCS 3173, pp. 356-361, Springer-Verlag, 2004.

[25] Olvera-López, José A., Carrasco-Ochoa, J. Ariel and Martínez-Trinidad, José Fco. Sequential Search for Decremental Edition. Proceedings of the 6th International Conference on Intelligent Data Engineering and Automated Learning, IDEAL 2005. Brisbane, Australia, vol 3578, pp. 280-285, LNCS Springer-Verlag, 2005.

[26] García, M. and Shulcloper, J. Selecting Prototypes in Mixed Incomplete Data. Lectures Notes in computer Science (LNCS 3773), pp. 450-460. Springer, Verlag, Berlin Heidelberg. New York. ISSN 0302-9743 ISBN 978-3-540-29850.

[27] Cortijo, J.B. Techniques of approximation II: Non parametric approximation. Thesis. Department of Computer Science and Artificial Intelligence, Universidad de Granada, Spain. October 2001.

[28] Bell, D. and Guan, J. Computational methods for rough classification and discovery. Journal of ASIS 49, 5, pp. 403-414. 1998.

[29] Deogun, J.S. et al. Exploiting upper approximations in the rough set methodology. In Proceedings of First International Conference on Knowledge Discovery and Data Mining, Fayyad, U. Y Uthurusamy, (Eds.), Canada, pp. 69-74. 1995.