Classifiers based on two-layered learning

Jan G. Bazan

Institute of Mathematics, University of Rzeszów
Rejtana 16A, 35-959 Rzeszów, Poland
bazan@univ.rzeszow.pl

Abstract. In this paper we present an exemplary classifier (classification algorithm) based on two-layered learning. In the first layer of learning a collection of classifiers is induced from a part of original training data set. In the second layer classifiers are induced using patterns extracted from already constructed classifiers on the basis of their performance on the remaining part of training data. We report results of experiments performed on the following data sets, well known from literature: diabetes, heart disease, australian credit (see [5]) and lymphography (see [4]). We compare the standard rough set method used to induce classifiers (see [1] for more details), based on minimal consistent decision rules (see [6]), with the classifier based on two-layered learning.

1 Introduction

A *classifier* (*classification algorithm*) is an algorithm that permits us to repeatedly make a forecast on the basis of accumulated knowledge in new situations. Classifiers are induced from training data and next they are used to classify new, unseen cases. Each new object is assigned to a class belonging to a predefined set of classes on the basis of observed values of suitably chosen attributes (features).

There have been proposed many approaches for classifier construction like classical and modern statistical techniques, neural networks, decision trees, decision rules and inductive logic programming (see, e.g., [3, 5] for more details).

One of the most popular method for inducing of classifiers is based on learning rules from examples. The standard rough set methods based on calculation of all (or some) reducts make it possible to compute, for a given data, the descriptions of concepts by means of minimal consistent decision rules (see, e.g., [6], [2]).

In the majority of rough set applications the computation of decision rules is done only at some initial stage of inductive learning. Next, the decision rules are used to built a classifier that can be applied to any tested object.

Another approach can be based on a two-layered learning strategy. In the first layer of learning a collection of classifiers is induced from a part of original training data set. Whereas in the second layer classifiers are induced using patterns extracted from already constructed classifiers on the basis of their performance on the remaining part of training data (called validation table).

The aim of the paper is to compare the performance of classifiers based on calculation of all minimal consistent decision rules with methods based on

two-layered learning presented in Section 2. We report experiments supporting our hypothesis that classifiers induced using the two-layered strategy show better performance on unseen objects than the traditional classifiers (see Section 3), especially if the misclassification cost is very high (e.g., some medical data, market data etc.). For comparison we use several data sets, in particular: lymphography (see, e.g., [4]) and StatLog's data (see [5]).

2 Classifiers Based on Two-layered Learning

Many decision rule generation methods have been developed using rough set theory (see, e.g., [1, 4, 8]). We assume that the reader is familiar with basic notions of the rough set theory. One of the most interesting approaches to the decision rule generation is related to *minimal consistent decision rules* (i.e. decision rule with minimal number of descriptors in premise) (see [6, 1]). The classifier based on all minimal consistent decision rules we will be denoted by Algorithm 1.

The standard rough set methods based on calculation of all minimal consistent decision rules are not always relevant for inducing classifiers. This happens, e.g., when the number of examples supporting the decision rule is relatively small. Then, instead of minimal consistent decision rules, we use approximate decision rules to eliminate this drawback. Different methods are now widely used to generate approximate decision rules (see [1] for more details). In our method for computing of approximate rules, we begin with generation of minimal consistent decision rules from a given decision table. Next, we compute approximate rules from already calculated decision rules. Our method is based on the notion of consistency of decision rule. The original optimal rule is reduced (by descriptor dropping) to an approximate rule with confidence exceeding a fixed threshold (see, e.g., [1] for more details).

In the majority of rough set applications the computation of decision rules is done only at some initial stage of inductive learning. Therefore, if we use algorithm based on approximate rules, the threshold of confidence for approximate rules computation has to be chosen at the moment of rules computation and next we can use approximate rules computed by this threshold for any tested object. The choice of threshold can be optimized for a given data set in the following way. The original training table is divided into a *training table* and *validation table*. Approximate decision rules are computed for the training table for any selected confidence threshold from a fixed set of thresholds. An optimization of the confidence threshold is performed using the validation table. The set of rules with the lowest classification error rate for validation table is chosen.

However, for the vast majority of data sets, we cannot find one, optimal confidence threshold that is relevant for all approximate rules (in classifying of any tested object). For example, we can have several thresholds of confidence obtained for a given data. Some of them can be optimal for some part of data set, the others can be optimal for some other parts. In the classical method, (mentioned above) a threshold relevant for the largest part of data set is chosen.

Another approach can be based on classifier construction using all approximate rule sets computed for a family of confidence thresholds. Such classifiers are used to classify new objects using a special decision table. Such table is constructed on the basis of classification results of objects from the validation table using decision rule sets computed for the training table. Decision rules computed for this special table can resolve conflicts between decision rule sets computed for training table. Therefore this special table will be called a *conflict table*. The calculation of rules for the training table is called a *learning on the first layer*. Whereas, the calculation of rules for the conflict table is called a *learning on the second layer*.

The structure of any classifier constructed on the first layer of learning allows us to identify the classifier with an algorithm for membership function derivation (see [2]). We can define it as a parameterised function μ_{CLASS_k} of the form:

IF $\max(w_{yes}, w_{no}) < \omega$ **THEN** $\mu_{CLASS_k}(u) = 0$

OTHERWISE

$$\mu_{CLASS_k}(u) = \begin{cases} 1 & \text{if } w_{yes} - w_{no} \geq \theta \\ \frac{\theta + (w_{yes} - w_{no})}{2\theta} & \text{if } |w_{yes} - w_{no}| < \theta \\ 0 & \text{if } w_{yes} - w_{no} \leq -\theta \end{cases}$$

where u is any tested object, $CLASS_k$ is the k-th decision class from the given decision table, real values w_{yes}, w_{no} called "for" and "against" weights to the object u are a normalised functions depending on properties of rules which have been computed for the training table (see [2], [1]) and ω, θ are parameters set by the user. These parameters allow us to search for new relevant features (attributes) based on attributes from the given decision table. We use these parameters to define new attributes on the second layer of complex classifier. Hence, condition attribute values of the conflict table are computed using function μ_{CLASS_k} for any decision class $CLASS_k$, where $1 \leq k \leq d$ (d is a number of decision classes) and for any θ from a fixed set (see [2]). Whereas, the decision attribute of the conflict table is the same as in the validation table.

We have mentioned before that in the vast majority of data sets it is not possible to find the optimal confidence threshold that is universally optimal (i.e., guaranteeing the high classification quality tested objects) for all approximate rules computed using such threshold. Hence, building the conflict table, we cannot classify objects from validation table using all approximate rule sets. Therefore for any set of approximate rules we construct a special table, called an *initial classifying table*. The structure of this table is the same as the structure of validation table, apart from the decision attribute that is replaced by another attribute computed using the result of classification test performed on the validation table by decision rules computed from the training table. Any object from the validation table is classified by rules computed from the training table and if the result of classification is correct, a value of decision attribute in the initial classifying table is equal 1. If the result of classification is incorrect or tested object isn't recognized, a value of decision attribute in the initial classifying table is equal 0. Hence, decision rules computed from the initial classifying table can

classify any tested object to the decision class 1 or 0. If the result of classification is equal 1, than original object from validation table (or more extended table) can be classified by rules generated from the original training table, values of function μ_{CLASS_k} (for any decision class) can be computed and inserted to the conflict table. Otherwise, the value MISSING (does not concern) is inserted to the conflict table instead of values of function μ_{CLASS_k}.

The experimental data sets usually consist of numerical attributes with large numbers of values. Therefore, if we want to calculate decision rules of the high quality, we have to use some discretization method. In this paper we use discretization methods based on the rough set techniques (see [7, 1] for details).

We present an algorithm based on the two-layered learning more formally.

Algorithm 2. *The classifier based on the two-layered learning*

Step 1. Split randomly a given table T into two subtables: the training table $T1$ and the validation table $T2$ (e.g., $T1$ is 50% of T).

Step 2. Generate discretization cuts for the table T and store them using a set variable C.

Step 3. Discretize the tables T, $T1$ and $T2$ using cuts from the set C (tables after discretization will by represented by TS, $TS1$ and $TS2$).

Step 4. For all selected confidence thresholds p perform the following operations:

a) calculate all rules for the table $TS1$ and store them using a variable $R(p)$,

b) shorten rules from set $R(p)$ to an approximate rules with threshold p

c) construct table $T2(p)$ in the following way:

(i) condition attributes of table $T2(p)$ are the same as in the table $T2$;

(ii) decision attribute of table $T2(p)$ is obtained by classifying table $TS2$ using rules from set $R(p)$:

- if object u from table $TS2$ is properly classified by rules from $R(p)$ then set value 1 as the decision value for u in the table $T2(p)$
- otherwise set value 0 as the decision value for u in the table $T2(p)$,

d) generate discretization cuts for table $T2(p)$ and store cuts to set $C2(p)$,

e) discretize table $T2(p)$ using cuts from set $C2(p)$ (table after discretization will by represented as $TS2(p)$),

f) calculate all rules for table $TS2(p)$ and store them in the set $RR(p)$.

Step 5. Create empty table TC.

Step 6. For all selected thresholds p do

a) For all selected θ insert attributes to the table TC computed in the following way:

(1) discretize object u using $C2(p)$,

(2) classify object u by rules from set $RR(p)$,

(3) if u is classified to the class with label 1 then

- discretize object u using C,
- classify object u by rules from set $R(p)$,
- compute values $\mu_{CLASS_1}(u)$, ..., $\mu_{CLASS_d}(u)$,
- insert values $\mu_{CLASS_1}(u)$, ..., $\mu_{CLASS_d}(u)$ to the table TC as value of attributes determined by $R(p)$ and θ;

otherwise insert value MISSING to the table TC instead of values $\mu_{CLASS_1}(u)$, ..., $\mu_{CLASS_d}(u)$

b) Copy the decision attribute from table $T2$ to table TC.

Step 7. Calculate all rules for table TC and store them using a set variable RC.

Table 1. Results of experiments with data sets

Data	Algorithm	Error rate	Coverage	Real error rate
Diabetes	A1	0.288	1.0	0.288
Diabetes	A2	0.241	0.747	0.435
Heart disease	A1	0.199	1.0	0.199
Heart disease	A2	0.161	0.868	0.272
Australian credit	A1	0.151	1.0	0.151
Australian credit	A2	0.141	0.962	0.174
Lymphography	A1	0.241	1.0	0.241
Lymphography	A2	0.218	0.835	0.348

Algorithm 2 can be used to classify any tested object u as follows. For a given object u we construct object uc analogously to the case of objects from the table TC (see Step 6 of Algorithm 2). Next, the object uc is classified using rules from RC, but if uc isn't recognized by rules from RC or it is classified to the boundary region or classification on u described by the decision attribute from the original data is inconsistent (table TC can be inconsistent) then the object uc is not classified.

Our hypothesis is that the performance of the classifier presented above is better than the classifier constructed using Algorithm 1. In the next section we will test this hypothesis using different data sets.

3 Experiments with Data

We present the results of experiments performed on the following four data sets, well known from literature: diabetes, heart disease, australian credit (see "StatLog" project - [5]) and lymphography (see, e.g., [4]).

The results of experiments were obtained by: 12, 9, 10 and 10-fold cross validation for the diabetes, heart disease, australian credit and lymphography data respectively.

Algorithms presented in this paper are implemented in object-oriented programming library: "RSES-lib 2.1", creating the computational kernel of the system "RSES 2.1" (see [8]).

We compare the results of Algorithm 2, presented in this paper, with those obtained by Algorithm 1 (see Section 2). Table 1 shows the results of the considered classification algorithms for the data sets mentioned before. In case of the cross-validation method we present the average (from all folds) values of error rate and coverage.

One can see that error rate of Algorithm 2 (on the covered region of objects) is lower than error rate obtained by Algorithm 1 for analyzed data sets. Therefore we conclude, that results of Algorithm 2 is better than results of Algorithm 1 for recognized objects by these algorithms.

In the algorithm presented above we use an algorithm for calculation of all reducts. Hence, the complexity of Algorithm 2 is very high. Therefore we plan to develop heuristics to obtain some knowledge about the reduct set instead calculation of all reducts (see [8]).

4 Summary

We have presented classifiers based on two-layered learning. The experiments show that the classification quality of such classifiers on covered regions of objects are better than results of the classifier based on the whole set of decision rules. In case the object misclassification cost is high (e.g., some medical data, market data etc.) one can prefer to have a classifier (Algorithm 2) returning answer *I don't know* if there is a high risk that the generated decision by classifier is false.

Acknowledgements

I wish to thank professor Andrzej Skowron for inspiration, stimulating discussions and great support while writing this paper.

The research has been supported by the grant 3T11C00226 from Ministry of Scientific Research and Information Technology of the Republic of Poland.

References

1. Bazan J.: A comparison of dynamic non-dynamic rough set methods for extracting laws from decision tables. In: [7], pp. 321–365.
2. Bazan, J., Nguyen, H.S., Skowron, A., Szczuka, M.: A view on rough set concept approximation. In Wang, G., Liu, Q., Yao, Y., Skowron, A., eds.: Proceedings of the Ninth International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC'2003), Chongqing, China. Volume 2639 of Lecture Notes in Artificial Intelligence, Heidelberg, Germany, Springer-Verlag (2003) 181-188.
3. Friedman, J.H. and Hastie, T., and Tibshirani, R.: The elements of statistical learning: Data mining, inference, and prediction. Springer-Verlag, Heidelberg (2001).
4. Grzymała-Busse J., A New Version of the Rule Induction System LERS Fundamenta Informaticae, Vol. 31(1), 1997, pp. 27–39
5. Michie, D., Spiegelhalter, D., J., Taylor, C., C.: Machine learning, neural and statistical classification. Ellis Horwood, New York (1994)
6. Pawlak Z., Rough sets: Theoretical aspects of reasoning about data, Kluwer Dordrecht, 1991.
7. Polkowski L., Skowron A. (eds.), Rough Sets in Knowledge Discovery vol. 1–2, Physica-Verlag, Heidelberg, 1998
8. The RSES Homepage – `logic.mimuw.edu.pl/~rses`