Tuning classification for prime implicants based learner

Ludmil Dakovski, Zekie Shevked

Abstract: *This paper considers an algorithm for learning from examples from the viewpoint of validation and improving classification accuracy. The examined algorithm represents the sets of positive and negative training instances as logical function in sum-of-minterms form and finds this functions prime implicants. After choosing suitable validation scheme the goal is to investigate in improving classification accuracy by determining influencing factors (parameters) and ways to set their optimal values (process more popular as tuning)*.

Key words: *Parameter tuning, validation, Learning from examples, Classification.*

INTRODUCTION

The algorithms for learning from examples use a set of training instances with a specified class to construct a model of the domain of interest. It is important to be known how predictive the resulting classifier is. Usually error on training data is not a good indicator of performance on future data because new data will probably not be exactly the same as the training data. Fitting the training data too precisely may lead to poor performance in new data. That is why avoiding this problem named overfitting is one of significant requirements to a learning algorithm because it impacts in general classification accuracy of the learned model. Performance of a classifier is evaluated via applying the model on test data. This way classification accuracy is determined as number of correctly classified instances divided by the total number of instances in the test set. Another popular measure is confusion matrix. The testing procedure itself might be done by different techniques. The most widespread ones are hold out, cross validation and leave-one-out method. In hold out approach certain amount of data is reserved for testing and the reminder is used for training as usually one third of the available data is for testing and the rest is for training. Essential to note is that test data is not used in any way to create the classifier. The process has two stages:

- Build the model on the training set;
- Test resulting models performance on the test set.

Some learning algorithms result in a classifier with one or more parameters whose values impact performance. For this type of algorithms the next task to be done after building the model is tuning parameters. Note that test data can not be used in any way for parameter tuning. Proper procedure works with three instead of two sets: train, validate and test. Validation data is used to optimize parameters.

Cross validation differs than hold out in data set split. In k-fold cross validation all of the examples are partitioned into k subsamples. Of the k subsamples is retained for testing the model, and the remaining k-1 subsamples are used as training data. The cross validation process is then repeated k times, with of the k subsamples used exactly once as the validation data. The k results from the folds then can be averaged (or otherwise combined) to produce a single estimation [6].

The third approach, leave-one-out method, involves using a single observation from the original sample as the validation data and the remaining observations as the training data. This is repeated such that each observation in the sample is used once as the validation data. This is the same as k-fold cross validation where k is equal to the number of observations in the original sample [6].

1. PRIME IMPLICANTS BASED LEARNER – TUNING PARAMETER AND VALIDATION

In this paper we continue investigation the learning algorithm introduced in [3]. Shortly it works as follows: training examples are used to build two logical functions in

sum-of-minterms form – one that corresponds to positive examples and other that correspond to negative examples. The main goal of the algorithm is to find minimal representation of target classification functions approximation that corresponds to all positive examples and probably some of unknown cases but does not correspond to any negative sample. After applying the minimization algorithm described in [3] a list of prime implicants is got. This list is used for classification as if a new unclassified instance is corresponded by implicants list, it belongs to the learned concept, i.e. it is classified positively otherwise it is classified negatively [3]. Significant feature of the presented algorithm is that the results depend strongly and entirely on the training data itself. That is why in order to avoid overfitting a parameter is brined in. This parameter is used on classification stage and represents the minimal number of prime implicants, which the currently tested instance must correspond, to be classified as positive. This restriction might be needed when positive and negative examples distribution leads to prime implicants list that correspond to a huge part of unknown area besides available positive instances. The goal of using this parameter is to avoid such situations that will bring along to unjust positive classification for a lot of unknown cases. This way when a class is determined for a test instance the requirement to it is to correspond to at least n prime implicants, where n is the value of the parameter. The meaning of this restriction to the tested instance consists of the following: as many prime implicants is corresponded by as increasing possibility more positive training examples to confirm its classification. Of course, this does not mean that the best value for n is its maximal value all prime implicants count, i.e. the tested instance is classified positively only when corresponds to all positive training examples (conjunctive hypothesis, as in version space approach of Mitchell [5]). It is shown that conjunctive hypothesizes has limited expressive power [4]. That is why since practical implementations of version space approach requiring a restricted concept language, it may be unable to induce consistent concept [1]. It is common opinion that conjunction does not fully represent possible concepts and both conjunction and disjunction should be used as it is in the proposed new algorithm where every implicant represents conjunction and the whole set represents disjunction. From described above follows the necessity of introducing such a parameter for classification.

The next task is determining its optimal value. Usually for such purposes a validation set is used. That is why we have chosen a three-step hold out estimate. At the same time it requires minimal computations thus usually it is the more preferred one in practice. The whole available set of data is separated to three parts. One bigger for training - the learned model is build on it. As a result we get a list of prime implicants.

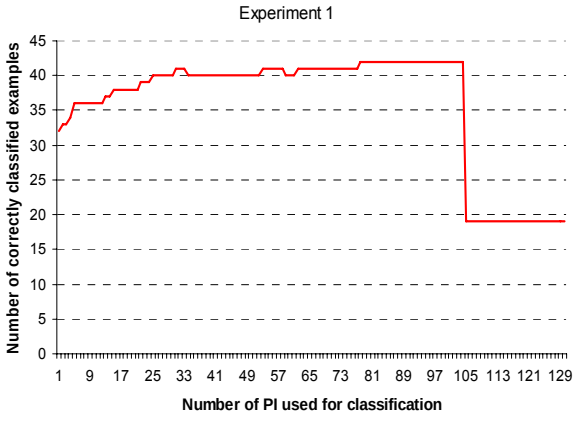
The second part of data is used for validation, tuning the optimal value of the parameter is done on it. Here a value is determined for the settled parameter. This value will be used in the next stage when classifying the test instances. After this stage we have the final look of the target classifier with specified set of prime implicants and the minimal necessary number of implicants required to classify an example as belonging to the class.

The remaining part of data represents the test set. Experiments are done on this set and classification accuracy is evaluated on it as well. This way test data is not used neither to create the classifier the classifier nor for parameter tuning.

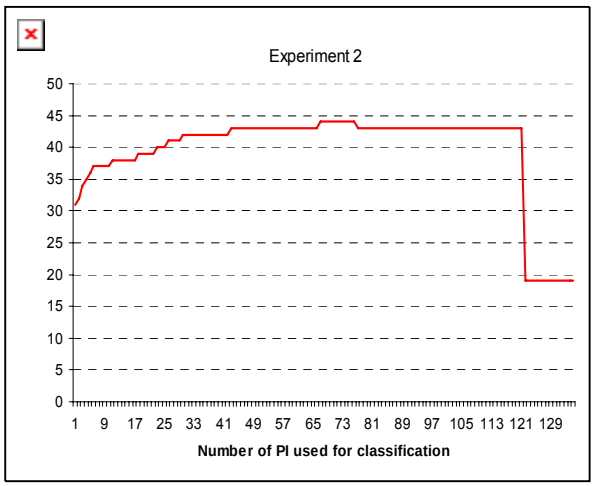
Another way to improve the process of building a classifier and its accuracy is to repeat hold out procedure. It can be made more reliable by repeating the hold out estimate with different subsamples. In all iterations, a certain proportion is randomly selected for training. Here another trick might be applied. Often the available dataset is small or unbalanced and then samples might not be representative, because we have only few or none instances of some classes. That is why when randomly separate data should make sure that each class is represented with approximately equal proportions in subsets.

2. EXPERIMENTAL RESULTS

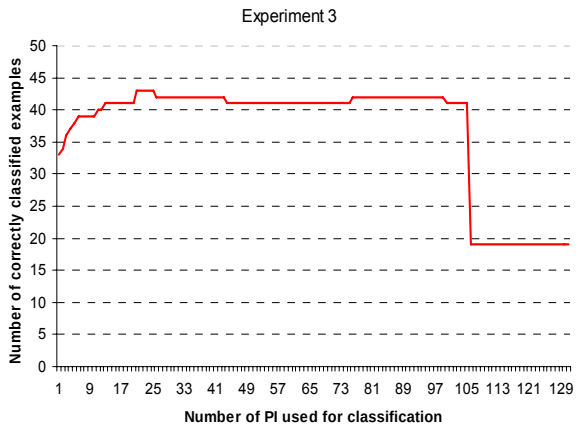
Experiments are performed on voting records drawn from the [2]. Each example is described by 16 attributes as their values are votes – yes, no or unknown (did not vote). Two classes to separate are democrat and republican. Learning is performed using the mentioned above prime implicants based algorithm and hold out method is chosen for validation. Experiments are repeated five times with different initial sample partitioning. On each experiment the whole set of 300 available examples is randomly separated in three parts – one for training consisting of 210 instances, second for validation and third for testing 45 each. Initial analysis is done on the training set and prime implicants are found. They are used in the next step tuning to classify examples in the second part of data with different values of tuned parameter. After that the optimal value is chosen, this is the one which leads to best classification accuracy on validation set. We already have built the model and settled the parameter, this represents its final look, which is gone, be used for classification. The last step is real testing of the got tuned classifier on remaining test set and evaluating its real performance. Results from experiments are presented on figure 1. Notice that without using this parameter, i.e. by default its value is 0, classification accuracy varies from 68.889 to 73.333. Increasing parameter value firstly increases classification accuracy as this goes on till values 20-30 where an interval of high classification accuracy starts and increasing parameter value has not notable impact on accuracy. After specific parameter value (different for all training set but approximately between 100 and 120) classification accuracy vastly decreases – required number of confirming prime implicants becomes to restrictive, i.e. used hypothesis gets near to conjunctive. It turns out that for this data set a fairly broad interval of appropriate parameter values exists and all of them ensure high classification accuracy. Maximal classification accuracy achieved is 44/45 = 0.978 for three of experimental sets and 43/45 = 0.956 for remaining two sets. The learning algorithm strongly depends on training data nature so it is very likely to get different graphics for new domains of interest.



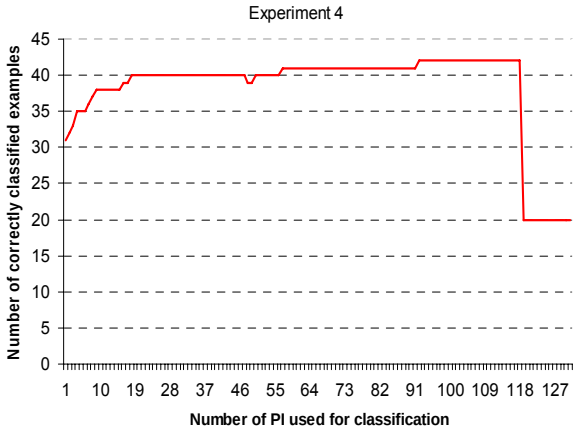
Found PI count: 130
Max number of correctly classified examples: 42
Optimal number of PI used for classification: 78
Correctly classified test instances: 43



Found PI count: 135
Max number of correctly classified examples: 44
Optimal number of PI used for classification: 67
Correctly classified test instances: 44



Found PI count: 131
Max number of correctly classified examples: 43
Optimal number of PI used for classification: 21
Correctly classified test instances: 44



Found PI count: 131
Max number of correctly classified examples: 42
Optimal number of PI used for classification: 92
Correctly classified test instances: 44

CONCLUSIONS AND FUTURE WORK

In this paper we continue studying prime implicants based learner. The goal is choosing suitable validation method. Reasons are given for preferring three step hold out estimate. Parameter impacting classification accuracy is determined and the proper procedure for tuning its value as well. These ideas are implemented in practice by realizing experiments on test data set. Future works persuades improving considered learning algorithm and continues investigating in suitable validation schemes for it.

REFERENCES

[1] Carpineto, C. Trading off consistency and efficiency in version-space induction. Proceedings of Ninth International Machine Learning Conference, Aberdeen, Scotland, pp. 43-48, 1992.

[2] Congressional Quarterly Almanac, 98th Congress, 2nd session 1984, Volume XL: Congressional Quarterly Inc. Washington, D.C., 1985.

[3] Dakovski, L., Z. Shevked. Alternative approach for learning from examples, *Proceedings of CompSysTech Conference*, Varna, Bulgaria, 2005

[4] Idemstam-Almquist. Demand networks: an alternative representation of version spaces. Master's thesis, Department of Computer Science and Systems Sciences, The Royal Institute of Technology and Stockholm University, Stockholm, Sweden, 1990.

[5] Mitchell, T.. *Version spaces: an approach to concept learning*. PhD thesis, Electrical Engineering Dept.,Stanford University, Stanford, CA, 1978.

[6] www.wikipedia.org.

ABOUT THE AUTHOR

Ludmil Dakovski, D.S., Department of Computer Systems and Technologies, Technical University - Sofia, Phone: +359 02 9652414, E-mail: seven-in@bulinfo.net.

Zekie Shevked, PhD student, Technical University – Sofia, E-mail: zekie_shevked@yahoo.com