Less is more: Active Learning with Support Vector Machines

Panagiota Nikolaidou

9 mars 2006

1 Introduction

Supervised learning methods find applications in many important real life activities such as routing of electronic mail, character and voice recognition. However, such methods need a number of labeled examples, which can be obtained costly, usually requiring the presence of domain experts. Active learning is an approach which tries to solve this problem by using a subset of the training data that is the most informative and by this way it achieves better performance of the classifier with less labeled data. In the article, active learning is applied to the Support Vector Machines (SVM) method. An "optimal" approach of active learning, based on the expected error is described (*greedy optimal strategy*). Because this method is computationally impractical, the authors propose an other approach which uses *selective sampling* (*a simple heuristic*).

2 Support Vector Machines

The classifier used in the article is SVM. Given a set of labeled data $D = \{(x_1, y_1), ..., (x_m, y_m)\}$, where $x_i \in R^N$ and $y_i \in \{-1, +1\}$, SVM defines an optimal hyperplane, as the unique hyperplane that separates positive and negative examples, for which the margin is maximized. In the case where the data are not separable, a *soft SVM* is used. Soft SVM allow to separate the data with a minimal number of errors [1]. The examples that are incorrectly classified or are within the margin of the hyperplane, are called *bound examples*.

3 A Greedy Optimal Strategy

The authors propose an active learning algorithm which is based on probabilities that are assigned to points in the space that are classified by the SVM. The formula that they use is the following [2] :

$$P(y = 1|x) = \frac{1}{1 + exp(-f(x))}$$

where $f(x)$ is the output of the SVM. They also use the expected error defined as the sum of the error on each training example, weighted by the distribution of test examples, which reflects how much each training example represents the test set. The algorithm to select each new example is : For each unlabeled example x, calculate $P(y = 1|x)$ and $P(y = -1|x)$. Add $(x, 1)$ to the training set, retrain, and calculate the new expected error $E_{(x,1)}$. Remove $(x, 1)$, add $(x, -1)$ to the training set, retrain, and calculate $E_{(x,-1)}$. Estimate expected error as $E_x = P(y = 1|x) * E_{(x,1)} + P(y = -1|x) * E_{(x,-1)}$. Choose the unlabeled example x, which has the minimum E_x.

This active algorithm is optimal, however it is impractical because it requires for each example selection, to calculate for each example two quadratic programming problems (finding the hyperplane is a quadratic programming problem). For this reason, the authors propose a simplier and less expensive method based on a simple heuristic.

4 A Simple Heuristic

The active algorithm uses the simple heuristic that the unlabeled example that will be chosen next, is the one which is nearest to the dividing hyperplane. This unlabeled example is easy to find by calculating for all the unlabeled examples the distance between them and the hyperplane (dot product computation) and by selecting the one that has the minimum distance. This heuristic tries to reduce the uncertainty area which is situated near the dividing hyperplane.

One alternative appoach is used in the case of high dimensional domains, where the number of the training examples is greater than the number of the dimensions. In this case, the unlabeled examples that will be chosen next, are those that are situated in dimensions perpendicular to those defined by the current training examples.

5 Experiments

The experiments were done in two domains : binary classification of four newsgroup pairs from the *20 Newsgroups* data set and topic classification on a subset of five topics from *Reuters*. The number of examples that the active algorithm uses in every iteration is set, for the experiments, to $b = 8$. Generally, there must be a trade-off, against the cost of resolving a new QP problem (more examples per iteration, less QP problems) and the cost of labelling an example.

The experiments have showed that active learning performs better than by random selecting the training data.

They also determine when the active learning algorithm should stop. The stopping criterion is when the margin has been exhausted (when there are no other training examples within the margin).

Using the active learner algorithm, the performance increases up to a peak and after, it starts to decrease (and approximates the level achieved by the random learner after adding all data). The reason that this happens is that until the margin has been exhausted (until peak) the performance increases and the model remains consistent. But when the margin contains no available training data, then examples that make the model inconsistent may be added (since we can use a soft SVM) and the performance decreases.

6 Conclusions

The active learning algorithm, described above, not only reduces the number of the training examples but it also obtains a considerable reduce in time since there is no need to calculate for each example selection the SVM for each unlabeled example (greedy optimal strategy). It determines a stopping criterion to obtain the peak performance with the less possible training examples.

7 My opinion

The article is clear and the experimentation has been done on many examples, and shows impressive results, in particular with the Reuters data which shows that active learning is really efficient : only 10 percent of the test set is used to produce better results !

The algorithm they present is clear enough to be implemented, only the choice of b is not well detailed. Maybe they should have given bounds for b : for example, Figure 2 shows that if we want to perform active learning with only 100 examples, we can't use $b = 64$. So it seems more careful to use a $b < \frac{\text{training set target size}}{5}$.

The stopping criteria could also have been mentioned on the figures. The authors say that when there are non more examples in the margin, the algorithm should stop because the accuracy could decrease, but if we look on Figure 2 with $b = 4$, we can notice that the accuracy seems to decrease very soon, for only 75% of accuracy.

Références

[1] Corinna Cortes, Vladimir Vapnik : *Support-Vector Networks*

[2] John C. Platt : *Probabilistic Outputs for Suport Vector Machines and Comparisons to Regularized Likelihood Methods*