Positive-only Semi-supervised Classification

Hidayath Ansari, Chaitanya Gokhale
University of Wisconsin-Madison

Abstract

We discuss in this report, the task of binary classification on a test set, given a training set consisting of large number of unlabeled examples and a handful of examples belonging to one class. The task is part of the *UCSD Data Mining Contest 2008.*

We are given 20 dimensional data from a real-world scientific experiment. The training set consists of 68560 examples, of which only 60 are labeled positive and the rest are unlabeled. We are given a test set of 11427 unlabeled examples. The goal is to classify this set into two classes, achieving the highest F1 score. The F1 score is given by

$$F1 = \frac{2.P.R}{P+R}$$

where P is Precision and R is Recall.

In order to gauge our performance continually, we are given a quiz data set consisting of 11427 examples, all unlabeled, for which we can submit label predictions once per day and obtain the F1 score.

Initial Approaches

First we estimated the fraction of positive examples present in the data. This information is useful irrespective of the method we use for classification. We can rewrite the equation for F1 score as follows, using the definitions for Precision and Recall.

$$P = \frac{TP}{TP+FP} \quad R = \frac{TP}{TP+FN}$$
$$\frac{2}{F1} = \quad 2 + \frac{FP+FN}{TP}$$

If we label all the examples as positives, then FN = 0, FP = total - TP. The F1 value can thus be used to to find TP. We found that $\frac{1}{8}^{th}$ of the data belongs to the positive class.

Next, we tried to use some intuitive methods. As we discuss below, some of them were just not worth trying while others performed miserably.

Clustering: We thought of clustering the given data to look for any observable interesting patterns. However, at this scale (68000 points) we didn't have the resources for an in-memory clustering algorithm. So we randomly sampled fractions consisting of a tenth of the unlabeled data and clustered it along with the given positive data, using a variety of algorithms (hierarchical, spectral, k-means, affinity propagation) (Zelnik-Manor and Perona 2004; Ng, Jordan, and Weiss 2001). However, we couldn't find any consistent meaningful clustering of the daya.

SVD and visualization: Considering the possibility that some of the dimensions in the data may be adding noise rather than assisting in classification, we reduced the dimensions of the data, by applying singular valu decomposition (SVD) to the data. We then tried visualizing the data to see if the positives fell neatly along some direction. We took 3 dimensions at a time from the first 10 significant dimensions but no discernible patterns could be seen. We also ran SVD on the positive points only and projected the entire set along the resulting axes.

Bayesian Classifier: The survey on Semi-supervised learning, (Zhu 2005), mentions Bayesian learning as a promising method for this type of task. Unfortunately, that didn't seem to apply to our problem.

We estimated the feature-wise probability distribution for the data assuming independent features and got the probability function for the complete data vector. Thus, we computed $P(x)$ and $P(x|+)$, where x is any example. We can compute $P(-|x)$, as below:

$$P(-|x) = \frac{P(x)-P(+).P(x|+)}{1-P(+)}$$

However, the number of positive examples is so small (60) in our case, that the estimate obtained for the probability is highly unreliable. As a result, when computing $P(-|x)$ we even ended up getting highly negative probability values for some of the examples!

Revised Plans of Attack

Sampling negatives: In order to use any conventional clustering or classification algorithm we need a set of examples that we are reliably sure are negative. To do this we tried several techniques of sampling negative examples based on their distance from positive examples and from each other. The aim was to get negative examples that were

representative of the entire dataset and not just from one or a few limited corners.

Using SVMs: After refining a set of negative examples by iterations of intersections with other sets of known high negative density, we trained linear and non-linear SVM classifiers using the given positives and guess negatives (Joachims 1999). We then used this classifier on the entire training set and partitioned it. Using the confidence values given by the SVM we picked the most confident positive and negative points, and re-trained another classifier based on those points. We made sure to include all initial positive and negative examples in each training set. We ran this for 3,5 and 10 iterations. The best result was for 3 iterations but it was only a marginal improvement over our other F1 scores so far.

- Non-linear kernel: It gave marginally improved results over 5 iterations.
- Transductive SVM: In a transductive approach, an SVM frames optimization problems based on given labeled data and uses that to come up with labels for the rest. It doesn't try to generalize and induct a wider boundary or classifier. However, this gave worse results than the other types of SVMs used. We believe this is due to sheer sparsity of known positive examples.

Nearest Neighbour The next major technique tried were variants of nearest-neighbor. Since our positive dataset was so sparse and spread out we needed a denser set of positives to run any kind of induction algorithm. Since the data was not easily linearly classifiable, we assume there must be a non-linear structure to it. While spectral clustering is meant for such problems, it couldn't be applied due to the time and space complexity required. A simpler way of detecting manifold structure in data is by looking at the nearest neighbor of points and growing following nearest neighbor links that way. We tried 3 major ways of growing, followed by several hybrids.

- One at a time: Starting with the initial set of positives we found the nearest point to any of the positives and labeled it positive. Then we located the nearest point to the new set of positives, and so on. We grew this set to 300,500,1000 and 4000.
- Batch-wise: Starting with the initial set, we found the closest point to EACH of the given positives and labeled them positive. We continued growing like that for the new positive sets, doubling in size each time. Another variant was growing by a fixed size each time.
- Average distance: The criteria for a point in this variation was average distance to all 60 positive points.

The first two approaches gave high similar sets of points, and the batchwise one gave us the best results so far (F1 score 0.4). The average distance performed the worst, in fact worse than random guessing. Some more things we tried related to nearest neighbor:

- **Growing positive clusters:** On observing the data points added by the above nearest-neighbour approaches, we found that all these points are added as a result of being close to a subset of the given positives. If in reality we have more than one disjoint cluster of positives, then with the above approaches, we might be completely missing one of the clusters. Hence, we first clustered the given positive examples (using hierarchical clustering) to form 5 clusters (the number was arrived at based on the inter-cluster distance distribution), and then let these positive clusters grow, using the batch-wise approach discussed above. The performance was very much comparable to the batch-wise approach, but there wasn't any improvement over our current best.
- **k-NN:** The basic premise of this method was if a data point has p or more positive data points in its k-closest neighbours, then that data point can be labeled positive. We identified positives from the training data using p=2 and k=10 and added them to the given positives to form the 'seed' positives to be used for quiz set. Using the 'seed' positives, p=2 and k=20 we identified positive data points from the quiz set. Adding them to the seed, we continued the process for 4 iterations. This approach performed worse than most other NN approaches.

One-class SVM: One-class classifier (Manevitz and Yousef 2001; Sch et al. 1999) learns a function that maximally covers the examples of a given class. We used the `LibSVM` implementaion of one-class SVM. Starting with the given positive examples, we tried to learn and iteratively improve our one-class classifier for the positive examples. In each iteration, we would add the unlabeled examples from training set, identified as positives by the one-class SVM, to our training set for the next iteration. We continued for about 8 iterations. We used this final classifier to label the quiz set examples. The results obtained were worse than the batch-approach.

Results

Our F1 scores ranged from 0.18 to 0.4 over 15 submissions. Nearest-neighbor variants were the best scoring, giving F1 scores from 0.12 (average) to 0.4 (batch). The clustering approach gave 0.36, and k-NN 0.29. SVM methods were slightly worse with scores of 0.24 (linear), 0.26 (RBF), 0.23 (transductive) and 0.31 (one-class).

Conclusions and Future Work

We tried a few other methods as well, such as simple ranking and a more complicated version similar to PageRank.

The best F1 score we obtained so far was 0.4. We went from bring third initially to fourth for a long time, and on May 7 we slid down two further places.

However, through this project we learnt some hard realities about real-world data. We had a chance to experiment hands-on with implementations of most of the math and algorithms covered in the course: Bayes classifiers, entropy and information theory, SVMs, regression, clustering, accuracy measures, dimensionality reduction techniques to mention a few. In addition we had to improvise and improve some techniques heuristically.

Future Work

Most of the problems we faced was due to a lack of reliable negative examples. Towards the end we got a fairly good set. Also, we feel that each technique has not been utilized maximally, and an ensemble of tuned algorithms would go a long way in improving results. Also the possibility of training a distance metric based on density of points and not just Euclidean distance. Co-training (Zhou, Zhan, and Yang 2007) also appears to be another promising technique if we can prove that disjoint subsets of the 20 features are enough to independently learn a classifier. Artifical Neural Networks is alsa another thing to be tried.

References

Joachims, T. 1999. Making large-scale support vector machine learning practical. 169–184.

Manevitz, L. M., and Yousef, M. 2001. One-class SVMs for document classification. *Journal of Machine Learning Research* 2:139–154.

Ng, A. Y.; Jordan, M. I.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. In *NIPS*, 849–856.

Sch, B.; o Platt; Shawe-Taylor, J.; Smola, A.; and Williamson, R. 1999. Estimating the support of a high-dimensional distribution.

Zelnik-Manor, L., and Perona, P. 2004. Self-tuning spectral clustering. In *NIPS*.

Zhou, Z.-H.; Zhan, D.-C.; and Yang, Q. 2007. Semi-supervised learning with very few labeled training examples. In *AAAI*, 675–680.

Zhu, X. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.