CS123A TERM PROJECT: CLASSIFYING YELP DATA

Linyu Dong, William Tarimo, Christopher Parkin, Emmanuel Awa, Dimokritos Stamatakis

Brandeis University

ABSTRACT

This paper describes our process of building a classifier for ratings data as assigned in spring semester of Machine Learning (CS123A), taught by Pengyu Hong at Brandeis University. The data set can be characterized by a large number of sparsely populated attributes (n=291) corresponding to ratings of 1 through 5 from the online ratings system, Yelp. To attack this problem we tried a wide variety of algorithms and compared their results, concluding that the Logistic Regression and SMO algorithms provided the most robust classifications, though neither were able to break the 50% accuray barrier. We also discuss a failed attempt at employing a 2-step approach to classification that we believe may yield better results with further experimentation.

1. METHODS

The Weka suite of Machine Learning software written in Java was used for the majority of our testing and experimentation. For handling files and interfacing with Weka we also employed the RWeka package in R statistical programming environment as well as Ruby for file manipulation and data organization tasks.

1.1. Feature Selection

The modified Yelp dataset that was provided contains 291 features. With little knowledge of the features themselves, we first wanted to explore fatures selection to establish a set of features that were critical to proper rating classification. Weka was used to evaluate the individual predictive ability of each feature, along with amount of redundancy between features, using the CfsSubsetEval algorithm[1]. This analysis was done by randomly sampling small sets of data from the training set, running feature selection, and capturing the unique set of all features delivered by the algorithm. However, after building models with various classifiers, both with and without the feature selections, we found classification to be most accurate when including all features. As a result, all features were included for training and testing of our models.

1.2. Downsampling

There is a fairly clear imbalance in the data, where classes 4 and 5 account for the majority of samples (63%), while classes 1, 2 and 3 combine for the rest. Class 4 outnumbers class 1 by over 4:1. Based on previous work we suspected that some algorithms might be affected by the imbalance, but no good consensus for handling this problem seems evident in the literature. To account for the imbalance we chose to take a naive approach and downsample the more prevalent classes to result in a roughly even distribution. However, we again found that downsampling had adverse effects on our classifiers and we therefore chose to omit it[2].

1.3. Models Evaluation and Data Sets

From preliminary experiments on the training set and suggestions from related research works, we wished to compare multiple potential classifiers as reported in section 2. To facilitate training and testing of the models, we first divided our data into a 70%/30% split; 70% for building our models (training) and 30% for performance evaluation (testing). Candidate classifiers were trained on a small sampling of the training data (15%) in order to reduce the amount of time required to build and evaluate the models. The best performers from that round were then trained on the full training set to select a winner. Sample sizes of the training set, 15% training subset, and testing set were 207,947, 31,192, and 89,119, respectively. All models were trained using a minimum of 7-fold cross-validation in Weka.

2. CLASSIFICATION

The best type of classifier for this data was not immediately clear due to the relatively high number of sparsely populated features as well as the observed class imbalance. We therefore chose to focus our attention on the following set of classifiers to determine if a) there was a top performer in the group and b) if we could learn a little more about our data set in the process.

2.1. Random Forest (RF)

The Random Forest classification algorithm works by constructing multiple decision trees over the course of the train-

ing phase. Each tree outputs a class label for the dependent variable (rating), resulting in a set of predicted class labels. The algorithm then returns the class label that appears most often in the set, also known as the statistical mode. This algorithm incorporates bagging and random features selection while constructing the collection of decision trees with controlled variations.

2.2. Support Vector Machine (SMO)

Support vector machines are supervised learning models that construct a set of hyperplanes in a high dimensional space. These hyperplanes created by SVM are then used in association with other learning algorithms to analyze data and recognize patterns. SVM takes a set of input data and predicts which class the output belongs. Due to the sparsity of the dataset, we had the impression that SVM would achieve a strong class separation by employing its functional margin.

2.3. AdaBoost.M1

Due to the large number of attributes and the obvious class imbalance in the training set, we were forced to consider the addition of boosting to the base classifiers we used. Adding boosting has been shown to remove/reduce classification biases and improve classification accuracy by combining weak classifiers to form a more accurate classifier. The most suggested boosting algorithm, AdaBoostM1, was used together with other chosen base classifiers.

2.4. Logistic Regression

The logistic regression classifier employs regression analysis to predict the value of a categorical variable based on one or more input parameters. The relationship between the class variable and the input parameters is defined using parameter probabilities in a maximum likelihood calculation. The Weka implementation of Logistic Regression is a modification of the work[3] of le Cessie and van Houwelingen (1992), in which Ridge estimates were incorporated. Also note that this experiment required multinomial logistic regression due to having five possible distinct classes.

2.5. Bayesian Networks

Bayesian networks are directed acyclic graphs whose nodes represent random variables in the Bayesian sense: they may be observable quantities, latent variables, unknown parameters or hypotheses. Edges represent conditional dependencies, nodes which are not connected represent variables which are conditionally independent of each other. Each node is associate with a probability function which takes as input a particular set of values for the nodes parent variables and gives the probability of the variable represented by the node[4].

3. A TWO-STEP APPROACH

As mentioned there was a clear imbalance in class distributions in the data, which made us think to look at the classification from a different angle. It became apparent to us that there should be large differences between classifications of 1 and 5, but that differences between ratings of 4 and 5, or 1 and 2, may not be as evident. To focus on these differences we first segregated the data into three ratings: A = 4, 5, B = 3 and C = 1, 2. By training separate classifiers on A and C, we obtained methods for teasing out those subtle differences. This process also required transformation of the original dataset from a ratings system of 1-5 into the rating of A,B,C and therefore required use of a second classifier. This approach can therefore be thought of as occurring in two classification steps:

1. Classify the data into ratings A (4 or 5), B (3) or C (1 or 2);

2. Further classify data in group A as 4 or 5, and group C as 1 or 2.

It became apparent fairly early into this research that this approach was not likely not bear fruit. Achievable classifier accuracies remained relatively low (less than 70%) for both steps of the process, leading to a an even lower end result. Embarrassingly low. So embarrassingly low that we chose not to include it in this paper. However, we believe that with further research its possible that a stronger step 1 classification could have resulted in a much better end result.

4. EXPERIMENT AND RESULTS

Classifier	10-fold cross-validation correct classification % accuracy on training data	correct classification % accuracy on testing data
Naive Bayes	38.89	38.44
SMO	47.53	47.80
Linear LibSVM (7-fold cross-validation)	34.37	34.45
AdaBoostM1 with SMO	47.525	47.80
Random Forest	38.49	38.95
J48 (Decision Tree)	34.92	34.75
AdaBoostM1 with J48	38.65	39.05
Logistic	47.79	**48.21**
AdaBoostM1 with Logistic	47.79	**48.21**
Simple Logistic	**47.82**	47.95

Fig. 1. Classifier accuracies on models trained using the 15% subset of our training data, and evaluations on our testing data set.

Classifier	10-fold cross-validation correct classification % accuracy
J48	36.3108
Logistic	**48.906**
AdaBoostM1 with Logistic	**48.906**
Naive Bayes	43.1682
SMO	***48.56***
Random Forest	40.3466
Random Tree	33.4145

Fig. 2. 10-fold cross-validation accuracies from classifiers trained on the entire training dataset.

Training data	10-fold cross-validation correct classification % accuracy
Original unmodified training dataset	**48.906**
100,000 downsampled training dataset	47.0491
48-feature-selected training dataset	45.3185

Fig. 3. 10-fold cross-validation accuracies on Logistic classifier on downsampling, feature-selection, original training datasets.

5. DISCUSSION

As mentioned, the high dimensionality and size of the dataset led us to train the classifiers on a small subset of the training data. We feel that while this subset was not fully representative of the training data, it provided a sufficient number of samples to determine which classifiers were best suited for handling this type of data in a preliminary analysis. Table 1 shows experimental results from evaluation of multiple classifiers trained on the 15% subset of training data and evaluated using the testing data. Logistic Regression and SMO classifiers performed in a similar fashion at just over 47% accuracy. Other tree-based methods yielded relatively poor accuracies, indicating that the high dimensionality and complexity of this data was too much to overcome in building trees, even for self-pruning algorithms such as J48.

Our expectation was also that libSVM would be competetive in our analysis, but we saw it fall short on the small training set where it yielded an accuracy of approximately 34%. This might be because SVM is basically a binary linear classifier, predicting the outcome as one of two classes, while our dataset had five classes with very distinct relationships. Therefore SVM is forced to do many rounds of binary classification in order to deduce the multi-class assignments, and this may have been made more difficult given this type of data and the imbalanced subset. Higher observed accuracies were observed when running SVM on the entire training set, but these observations cannot be verified now due to a loss of the generated model and time constraints in re-generating it.

Overall, the results in Table 1 demonstrate that Logistic regression performs the best compared to other classifiers, with SMO trailing closely behind. From these observations we focused our attention on Logistic Regression and SMO to determine which would perform better in predicting unknown outcomes after being trained on our complete set of known data (297,066 samples). Results of training these two classifiers using the entire training dataset, and the the 10-fold cross-validation accuracy results, are shown in Table 2. This provides us with another indication that logistic regression most accurately models training data, though again SMO is close behind. Further, additional experiments using feature selection, balanced data sets and parameter optimization Logistic Regression and SMO still did not yield more favorable outcomes. Based on these findings, we employed the default Logistic Regression classifier in Weka to predict outcome labels for our unknown data set of 33,005 samples.

Another unexpected result was found when comparing runs with and without feature selection, and with and without downsampling (Table 3). The common indication is that these common practices will lead to a more robust classifier and therefore do a better job of prediction, but we found the opposite to be true. The lesson learned from this exercise is that there's no golden rule when it comes to classification of data, especially when it's highly dimensional like what we saw here. There was also an added constraint in that we knew very little about the data itself. Given more information, it's possible that we may have been able to formulate better feature selection procedures and see a benefit rather than a deficit.

6. CONCLUSION

Technological advancements in global communications and social networking have generated large amounts of data across various fields, a perfect example of which can be found in the online retail and restaurant rating system known as Yelp. Aside from the obvious tasks of finding a great restaurant for Friday night, these data collections provide us with the opportunity to study how humans think, what they like and dislike, and how they interact; both in solitude and in social settings. However, this data is only useful in the research setting if we have a computational method for harnessing the knowledge and making sense of it in some meaningful way.

As an example, we were provided with a largely transformed dataset from Yelp, and have trained classifiers over this model to predict unknown ratings and learn patterns given only a set of user-supplied inputs. We attempted a number of widely accepted best-practices, such as feature selection, down-sampling, and combining algorithms in order to obtain the best approach.

The results of our experiments show strong evidence that Logistic Regression and Sequential Minimal Optimization (SMO) were the most accurate classification models for the provided data. After cross validation of approximately 10 folds, on the full dataset, we got an accuracy between 47 - 49% for both classifiers.

7. MEMBER CONTRIBUTIONS

This project was a truly collaborative effort, with an equal amount of contribution made by all. Chris experimented with down-sampling, feature selection and a two-step classification approach. Linyu carried out experiments on different sizes of training datasets to find an optimal size using various classifiers, and worked to format the final report using LaTeX. William researched methods such as SVM, Random Forests with and without Boosting, and Naive Bayes, and generated output tables for the report. Dimos spread large runs over various servers to test the effects of running classifiers with increasingly large training sets. Emmanuel developed scripts for running weka on terminals to generate data for many different classifiers. Each member did a great job of contributing to researching, writing, formatting and editing the final report.

8. REFERENCES

[1] M.A.Hall, "Correlation-based feature subset selection for machine learning," 1998.

[2] Cailing Dong Gongping Yang Guangtong Zhou Xinjian Guo, Yilong Yin, "On the class imbalance problem," .

[3] J.C. Le Cessie S., Van Houwelingen, "Ridge esitmators in logistic regression," *Applied Statistics*, vol. 41(1), pp. 191–201, 1992.

[4] David Heckerman, "A tutorial on learning with bayesian networks," *Technical Report MSR TR*, vol. 95-06.