Final Project Report **CS290I**

June 12, 2014 Yalun Qin

Task Definition

The goal of my project is to compare the performances of different classifiers on a multi-class data set. I chose the data set Vehicle from Statlog mainly because it is not linearly separable and the performance of SVM classifier is not too good ($< 90\%$). It is really meaningless to use a data set on which using SVM alone can achieve a high classification accuracy.

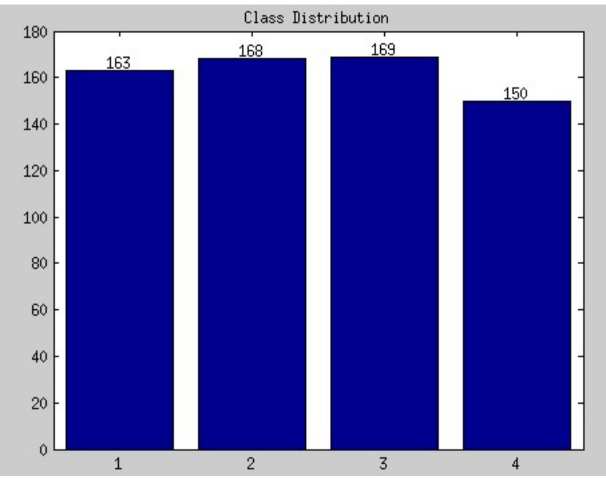


Figure 1: Class Distribution

Moreover, as can be seen from Figure 1, the data set is almost balanced. I don't encounter unbalanced data set very often, and usually I can always convert unbalanced data set to balanced one by collecting more data. Also note that this data set has 18 features for each data point, which complicates the classification and analysis of the data.

There are 650 instances in the training data set and 49×4 instances in the testing data set, both of which are scaled to [-1, 1].

Random forest classification and results

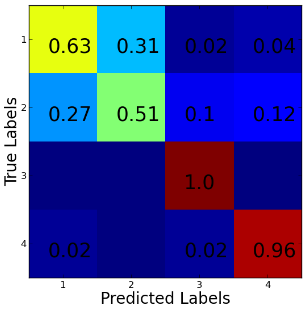


Figure 2: Confusion matrix of random forest classifier

To maximize the performance of RF classifier, it's necessary to find the best parameters, which basically are number of trees, the number of features to consider when looking for the best split, etc. Though it seems that there are so many parameters to tune, most of them only have very few values, and thus using grid search based on 10-fold cross validation is enough to find the best values.

After training on the training data set using the best parameters found, I tested on the test data set, and got an accuracy of about 76%. Note that this number may vary over different tests, but in a small range of about ± 3%. The confusion matrix is shown in Figure 2. As you can see, it doesn't perform well in differentiating between class 1 and class 2, although it does a relatively good job in classifying third and fourth class.

One good thing about random forest classifier is that it can give feature importances, as is shown in Figure 3. Usually it is a good figure to show which features are more importance and thus useful for feature selection. However as you can see in this figure, for my data set there are very big deviation of importance score for each class. Also I ran RF several times and got totally different orders of feature importances. This may probably because none of the feature can differentiate classes much better than others and thus I think feature selection may not be able to improve classification accuracy.

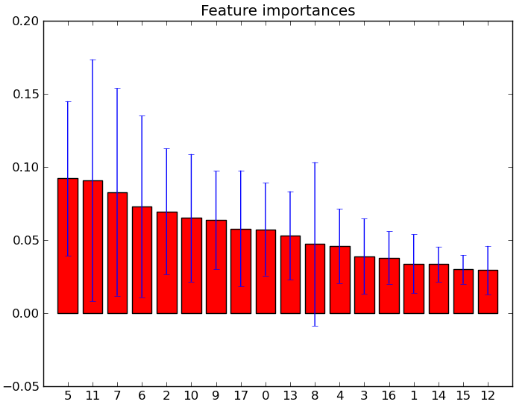
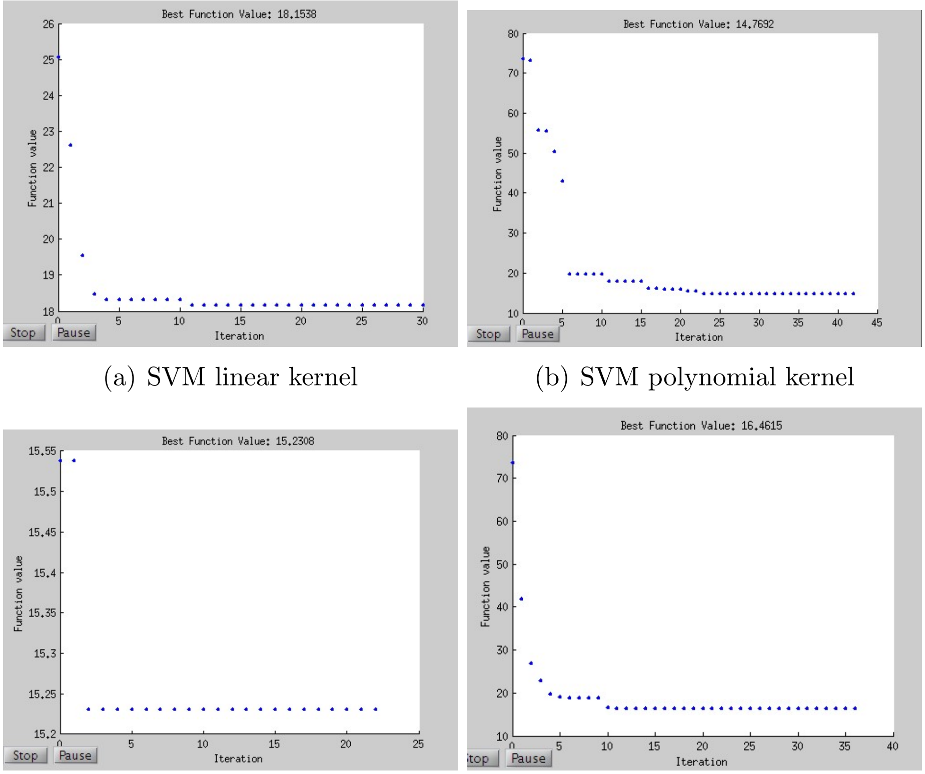


Figure 3: Feature importances. Red bars are mean values of importance scores and blue segments indicate the deviations of the importance scores

SVM classification and results

To find the best parameters, I used both grid search and pattern search[2]. Pattern search is much faster than grid search, and usually can converge to the global minimum. The objective function for PS is a function that outputs the error rate of 10-fold cross validation on training data using SVM classifier given its parameters.

Figure 4 shows the error rate of each iteration in the pattern search. I also used grid search to find best parameters for RBF kernel, which are very closed to the ones found by pattern search. After I got the best parameters, I used these parameters to train SVM classification models on the whole training data set and tested them on the test data set. The confusion matrices corresponding to the 4 SVM kernels are shown in Figure 5. Very similar to RF classifier, the SVM classifiers also perform better on class 3 and 4 than



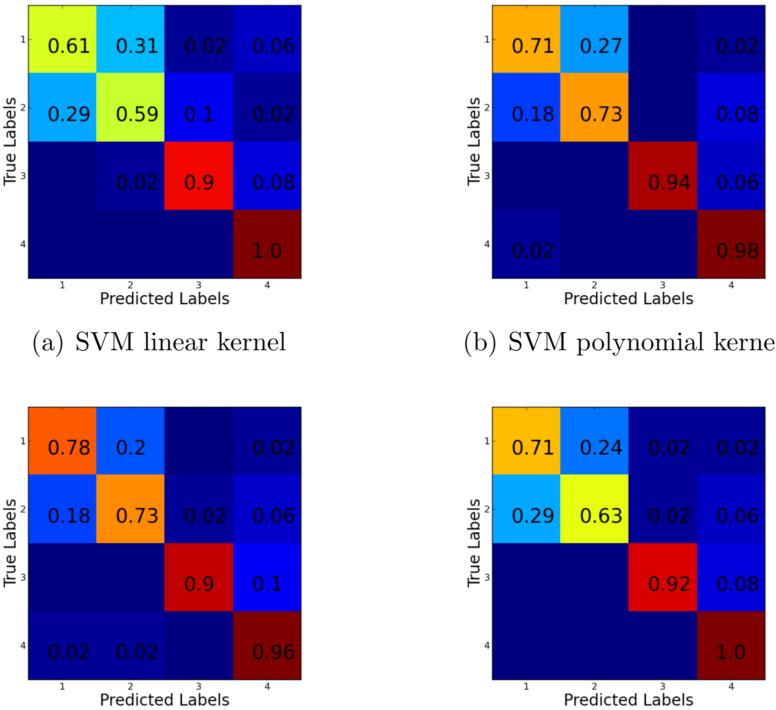
(a) SVM linear kernel

(b) SVM polynomial kernel

(c) SVM RBF kernel

(d) SVM sigmoid kernel

Figure 4: Pattern search results for 4 kernels of SVM classifier



(a) SVM linear kernel

(b) SVM polynomial kernel

(c) SVM RBF kernel

(d) SVM sigmoid kernel

Figure 5: Confusion matrices using 4 kernels of SVM classifier

class 1 and 2.

Adaboost classification and results

It is shown in the paper[5] that although AdaBoost was proposed to solve both the two-class and the multi-class problems (Freund & Schapire 1997), it sometimes can fail in the multi-class case. Therefore the authors proposed two new algorithms, SAMME and SAMME.R to improve the performance of Adaboost in multi-class case. I used these two algorithms on my data set, and the main parameters I need to tune is the number of weak classifiers and the complexity of the base classifiers. To find the best values of these two parameter, I just simply tried every values and pick the one that maximize the accuracy. For the first parameter, usually the larger the better, so I picked a value that is a little bigger than the found one. For the second parameter, I tried almost all the supported base classifiers and found random forest is the best one. From the results I found that SAMME.R always achieves higher accuracy than SAMME, and also converges faster. Therefore I decided to only use SAMME.R. After training on the training set and testing on the testing set, the average accuracy is about 79%, which is higher than the accuracy of random forest. Moreover, this accuracy turned out to be quite stable over multiple tests, which varies in a small range of ± 1%.

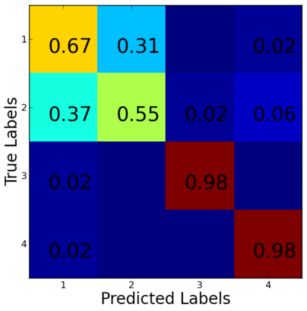


Figure 6: Confusion matrix of Adaboost classifier

Figure 6 shows the confusion matrix on the testing data. Although still poor on classifying the first two classes, but performs extremely well on the last two classes.

Comparison of the classification performances

As can be seen from Figure 7, SVM classifiers with polynomial and RBF kernel have a higher overall accuracy over other classifiers. Note that due to the randomness of random forest classifier and Adaboost classifier, I took the mean of five tests for each classifier. The blue segments represent the standard deviations. From this figure we can also see that the two ensemble classifiers generally perform almost no better than simple SVM linear classifier. If we only compare the two ensemble classifiers, we'll find that Adaboost truly boosts the performance of random forest classifier, but at the cost of time spent on both training and testing processes.

Figure 8 shows the performance of the classifiers for each class. As can be seen from this figure and previous confusion matrices, all of the classifiers perform much better on

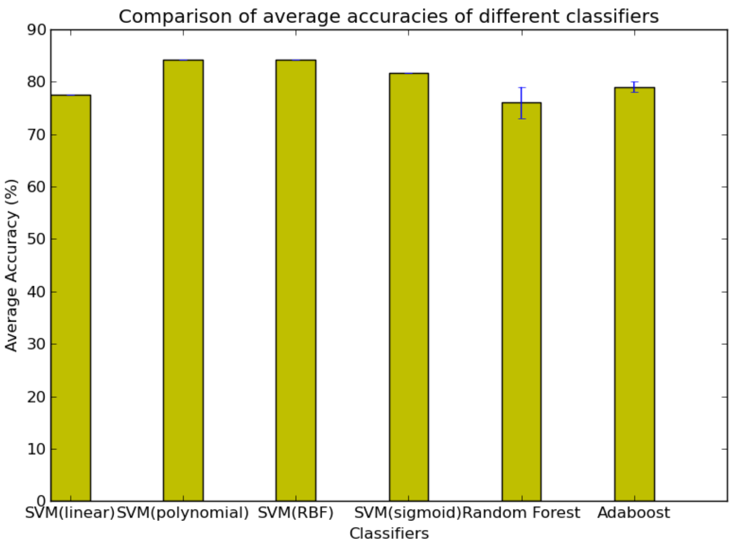


Figure 7: Average accuracy for each classifier

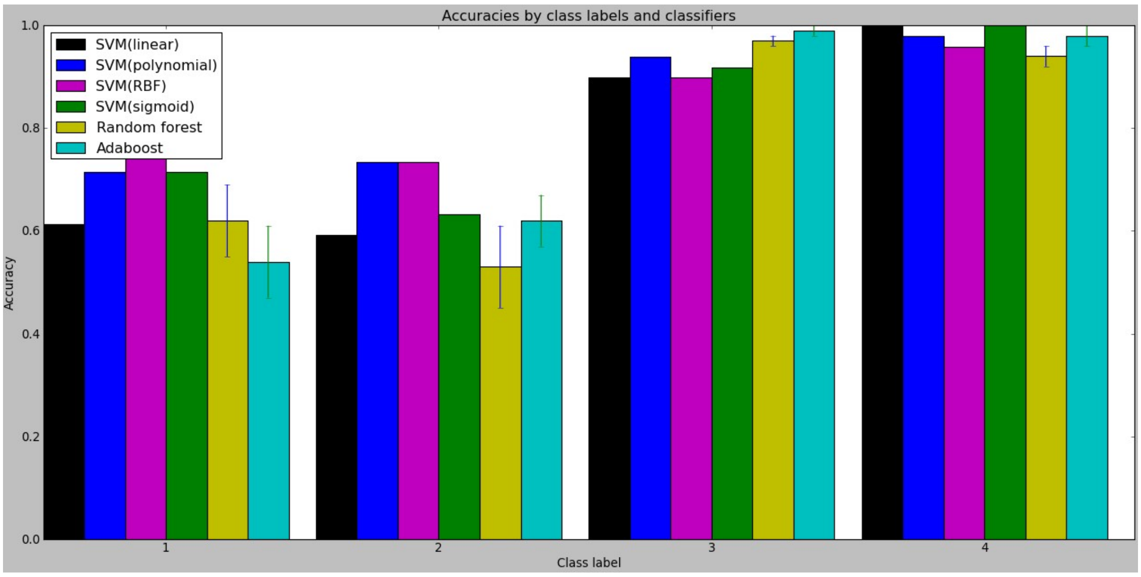


Figure 8: Accuracies by class labels and classifiers

classifying class 3 and 4 than class 1 and 2. The reason for this, I think, has nothing to do with the classifier itself. It is probably because the data sets contain much more noisy data for class 1 and 2. Also as can be seen from previous confusion matrices, these two classes are often misclassified as each other, which is probably because the features used in the data sets can not well differentiate these two classes. Because of these two reasons, class 1 and class 2 can not be well classified by any classifiers I used. However for these two classes, SVM classifiers, especially with polynomial kernel and RBF kernel, have achieved much higher accuracies than the other classifiers. But for another two classes, the ensemble classifiers, especially Adaboost, slightly win over SVM classifiers with RBF and polynomial kernels. Therefore we can conclude that at least for my data set, ensemble classifiers can achieve quite a good performance if the data are not too noisy or hard to differentiated due to bad features. But these two classifiers are quite sensitive to those failures, not as robust as the SVM classifiers. Also these two ensemble classifiers have much less parameters to tune, but require much longer time to train and predict the classes of new data.

Summary and possible improvements

There are still some places that need to be improved. First is about the size of training data set and testing data set. Considering the number of features, 650 training data and 196 testing data are not quite enough. More data should be collected in order to make more reliable conclusions. Second is that I actually didn't try all the base classifiers due to time constraint when using Adaboost. Maybe using some other base classifiers can improve the overall performance.

Since I only used one data set, I can not guarantee that the conclusions I got can be applied to other data sets. To get more general conclusions, much more data sets are needed for test, which requires much more time than I have. However the advantages and disadvantages about the classifiers I found are still quite useful and can be a good reference for me if I want to do classification using other multi-class data sets. Also through this project I got more familiar with some machine learning tools, especially the powerful python ML library Scikit-learn, which is very helpful for my future study.

Reference

[1] L.Breiman, Random Forests, Machine Learning, 45(1), 5-32, 2001.

[2] Audet, Charles and J. E. Dennis Jr. "Analysis of Generalized Pattern Searches." SIAM Journal on Optimization, Volume 13, Number 3, 2003, pp. 889903.

[3] Scikit-learn user guide 1.9. Ensemble methods: `http://scikit-learn.org/stable/modules/ensemble.html#id2`

[4] Scikit-learn user guide, Multi-class AdaBoosted Decision Trees: `http://scikit-learn.org/stable/auto_examples/ensemble/plot_adaboost_multiclass.html#example-ensemble-plot-adaboost-multiclass-py`

[5] J.Zhu, H. Zou, S. Rosset, T. Hastie, Multi-class AdaBoost, 2009.

[6] Slides of this course