**Test and Analysis About Common Classifier for the Finance Data**

**Author(s)**

Shuting Chen Junchen Lu

**Introduction**

Motivation of the problem, brief discussion of related work on the problem, and summary of the project effort and findings.

Finance prediction is always a topic of interest. Though finance prediction is always something that machine learning can completely handle. In Quora.com, one of the researchers says for him, he had not correctly identified the difficulty of the whole trading problem. But it indeed helps the part of the decision making.

Related work of this problem involves . In finance field, it is a specially used in portforlio management, fraud detection, payment prediction, loan risk analysis, mortgage scoring, determining transaction manipulation, determining financial risk management, determining customer profile and foreign exchange market. Classification and clustering of customers for targeted marketing. And some common data mining techniques involve A: Bayes Classification, B: Decision Tree, C: Boosting, D: Bagging, E: Random forest Algorithm

The project focus the effort in the using common data mining approach in a real world finance data pulled from Yahoo Finance. See what the common classifier do with a real world data set and get a insight knowledge about how can a prediction system be built. We have worked out a sample implementation about using multiple classifiers with different parameters. After showing the accuracy using cross validation, precision, recall as well as the f1 score, we can get some analysis done by looking at the data and tell what classifier is probably better and which is less useful. Note that we have made it a classification problem instead of a regression problem like what we present in our presentation.

**Technical Approach**

Detailed description of the data sets used, methods used for data preprocessing and data mining tasks, and experimental procedure followed. Note: you don’t need to give detailed description for well-known methods (for example, those in the text-book). Save the space for describing project-specific technical details.

In order to make the program work well, we take the online tutorial about machine learning in python and get a lot of ideas on how to use the sklearn package as well as a fully preprocessed data with missing value replacement and max-min normalization. The data is preprocessed from the 2 years of html file from the s&p500 stock market about some major tech company. 35 features are chosen from the original data. The make this problem less complex, we have made it a classification model with the negative referring to the fact that the price will go down and positive referring to the fact that the price will go up. And we get 20 best features of the 35 using the SelectK method from the sklearn package. The input file is called finance\_stat.csv, there are a total number of 2974 entries in there.

For the classifiers, we have used SVM with Radial basis kernel, polynomial kernel, linear kernel as well as sigmoid kernel, decision tree classifier, random forest classifier, Naïve Bayes classifier and KNN classifier. And we get the accuracy, precision, recall as well as f1 score of all of the mentioned classifiers. Note that we intend to use the Neural Network classifier. But Neural Network is only supported in the next develop version of the software, which is hard to get hands on. But since Neural network is important, we will make it our future work. For KNN classifier, sqrt(size) / 2 is chosen to be the k value. Note that size there is the size of the entire data set, which is 2974 in this case. For polynomial svm, degree 3 is chosen to avoid overfit and meanwhile provide some variance. Fro random forest, 100 estimators (each estimator is one decision tree) are chosen to get the final result.

Cross validation is performed to get the final result of the labeled data. The validation we used here is random sampling cross validation. A total number of 25 test set and related validation set are generated. 0.05 are chosen to become the test size (since it gives the better result than other common value like 0.1 or 0.2). 25 iterations are used to process the 25 random samples. So you know the final value is the mean value of 25 validation data. Total run time is 95.58 seconds in my own machine, which is a Macbook Pro with intel core i5 2.7GHz, 8GB memory, 128GB SSD.

Finally, a bar chart is generated with every output value to show the result is a straight forward way.

**Results and Discussion**

Present experimental results, and analyze observations and trends in the results.

Finally, we have our results of all of the classifiers. We will begin the analysis about the results we are getting and provide some explanation about why we are getting this result.

*Output:*

*-------------------------------------------------begin of input---------------------------------------------------*

*The number of samples in the original dataset is 2974*

*Classifier is rbf accuracy:0.612483221477 precision:0.594810240187 recall:0.812810499986 f1:0.685894347545*

*Classifier is poly accuracy:0.578791946309 precision:0.557781057597 recall:0.941883315031 f1:0.699671607995*

*Classifier is linear accuracy:0.565100671141 precision:0.556410932096 recall:0.838233775636 f1:0.667435286387*

*Classifier is sigm accuracy:0.52322147651 precision:0.52322147651 recall:1.0 f1:0.686091936216*

*Classifier is DecTree accuracy:0.693959731544 precision:0.70787628414 recall:0.707967972884 f1:0.706496272405*

*Classifier is Ranforest accuracy:0.771140939597 precision:0.767820461536 recall:0.806773247179 f1:0.785767604533*

*Classifier is NaiBayes accuracy:0.557181208054 precision:0.552565620282 recall:0.809142548572 f1:0.655523937588*

*Classifier is KNN accuracy:0.613087248322 precision:0.613520902494 recall:0.703730721797 f1:0.65420035837*

*time used is 371.368865013 seconds*

*-----------------------------------------end of output----------------------------------------------*

Here is a straight forward reading of all of the output data is as follows. Note that the equal sign here (==) means almost not strictly equal.

In terms of accuracy, random forest > Decision tree > 0.7 > radial basis svm == KNN > 0.6 > rest.

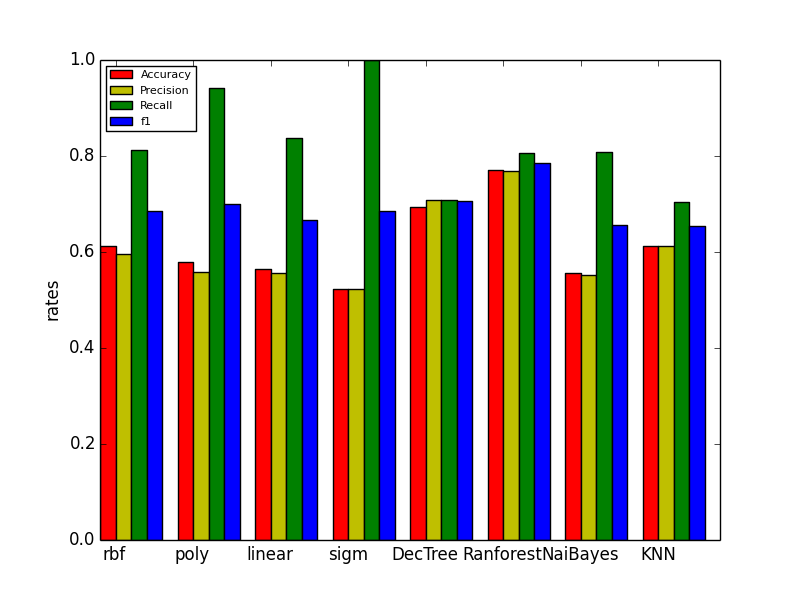
In terms of precision, sigmoid svm > polynomial svm > linear svm > rbf svm > Random forest == Naïve Bayes > Decision tree > kNN.

In terms of recall, random forest > Decision tree > rest(rest are almost equal)

In terms of f1, random forest > decision tree > rest(this part is almost equal) > kNN.

Overall, random forest provides the most accuracy (around 80 percent), most stability (the most balanced value in terms of precision, recall and f1 score). Decision tree also provides good accuracy along with balanced precision and recall. According to the sklearn document, the decision tree implementation uses CART (Classification and Regression Tree), which is something similar to C4.5 but can be used in classification algorithms as well as regression algorithms. The idea of CART algorithm is that it uses greedy approach to always get the one feature with the best information gain and make a split during each of the iteration. Decision tree has several advantages. First, little data preparation is needed. Second, able to handle numerical or categorical values. Since the data is from the real world, it’s very hard to do a perfect preprocessing. These particular advantages of decision tree will give it a performance boost among non-tree-based classifiers. Random forest is often referred to as a ensemble method. In a random forest, a split is chosen if it is the best split among a random set of features. So for each

*The chart is like this:*

**

single random tree, the bias will slightly increase compare to the non-random tree. But due to the averaging of the result, it will still outperform a decision tree.

Radial basis kernel, polynomial kernel, linear kernel and sigmoid kernel svm have slightly different accuracy, precision as well as recall and f1. But they share a common feature. That is, they all have a very high up recall (90%) while their precision remain lowest among all of the classifiers. Higher recall means high ratio of relevant data retrieved from all the relevant result while low precision means low ration of relevant data retrieved from all the retrieved result. Note that relevant data means stock whose price is going up and irrelevant data means the other way around. Retrieved data means data that is considered relevant by the classifier. Note that the kernel we are using here are all general purpose kernel. But for this data set, the general purpose kernels only performs around 60% accuracy, which is not ideal. The idea of SVM is to minimize the generalization error or cost function in the existing data set. So error in the testing dataset is very minimized. That also explains why recall rate is very high. But lower precision means it misclassify a lot of data that should not be relevant to the relevant category. So I think it is clearly overfitting since test error is much higher than generalization error. If one wants to improve the performance of SVM. According to one of the reference, deep knowledge of the finance data is required so one can write a fully customized kernel to the SVM and improve performance.

K nearest neighbor performs the second worst among all of the classifiers. The reason is because for this high dimension data (20 dimensions), this is especially bad for k-nearest neighbors If the number of dimensions is very high the nearest neighbors can be very far away. So performance rely heavily on curse of dimensionality (feature reduction). It’s hard for the kNN algorithm to deal with high dimensional data.

Naïve Bayes classifier’s performance is the worst among all. NB classifier makes assumption that features do not dependent on each other. In certain scenario like any text classification (eg. Spam email), it works well. But for this real world finance data, that is not the case. Features in fact depend heavily on each other. That’s the major reason why NB classifier has this kind of performance.

**Conclusions**

From this course project, we learned the performance of various types of technique on real world data. We also learned some new technique, like ensemble technique random forest as well as programming for the entire machine learning pipeline, from preprocessing, sampling, learning to evaluating with graphs.

For future works, first of all, more ML algorithm should be applied, like back-propagation neural networks, more ensemble techniques like abaBoost and bagging. Second, more validation technique will be applied. We only use the random resampling for this application. But techniques like k-folder cross validation or leave p out cross validation should also be great.

**Reference**

<https://www.researchgate.net/post/How_can_we_find_the_optimum_K_in_K-Nearest_Neighbor>

<http://stackoverflow.com/questions/11568897/value-of-k-in-k-nearest-neighbour-algorithm>

<https://www.quora.com/What-are-Kernels-in-Machine-Learning-and-SVM>

<http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

<https://en.wikipedia.org/wiki/Precision_and_recall#Precision>

<https://www.creighton.edu/fileadmin/user/HSL/docs/ref/Searching_-_Recall_Precision.pdf>

<http://scikit-learn.org/stable/modules/ensemble.html>

<https://www.quora.com/What-are-the-advantages-of-different-classification-algorithms>

<http://thesai.org/Downloads/Volume4No11/Paper_5-Performance_Comparison_between_Na%C3%AFve_Bayes.pdf>