Predicting probability of default for consumer loans with Machine Learning

By Tiantian Wu

Introduction and Problem Description

I would like to explore a key problem in the industry, which is the prediction of default for loans. The traditional banks are being challenged by Fintech companies to attract new loans. It is interesting to learn how the loans originated by banks behave differently than loans originated by a Fintech company through digital platform. In this study, I used the dataset from a bank, which is used in a kaggle data science competition. The dataset representing a Fintech company is from Lending Club, one of the very popular peer-to-peer marketplace connecting borrowers and investors. The main business questions are as follows:

- Which type of loan (i.e., bank personal loans vs. Fintech personal loans) has better credit quality;
- How credit quality can be better predicted, that is, which variables have more predicting power;
- Do traditional bank and Fintech Company have different underwriting standards and loan performance tracking mechanism, such as borrower creditworthiness requirement, income, debt-to-income ratio?

In terms of methodology, Loan defaults modeling is a categorical classification problem - a borrower may either default at some time during the loan term, or prepay before loan term ends, or complete the payment until the end of loan term. Traditionally, logistic regression and score card are used to model the default problem. More machine learning methods, such as decision tree, SVM, and Neural Networks, are being attempted by modelers to address the sample problem. In this study, I explored the performance and pros/cons of some of those machine learning methods.

1 Dataset 1 – Lending Club Loan Performance Data

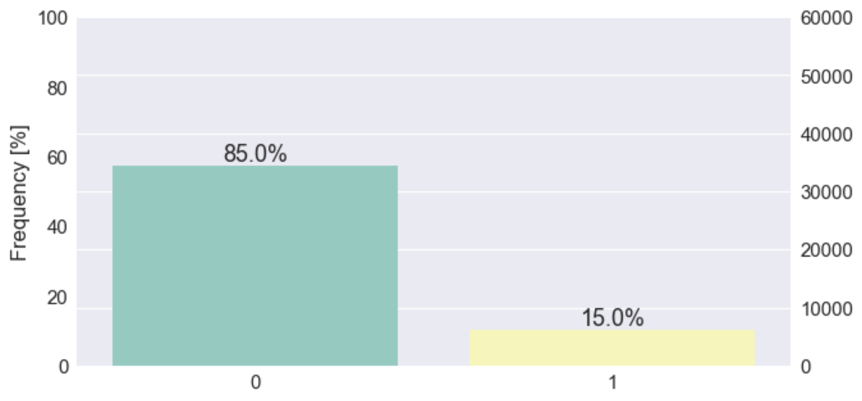
1.1 Data Description, Preprocessing and Feature Engineering

The data was downloadable from the web: https://www.lendingclub.com/info/download-data.action . The data include over 40K records and ~150 features. The label or predicting target is the loan default status. The following data preprocessing was used to prepare the data for analysis:

- imputing NaN with 0 or averages depending on situation,
- transforming nominal features to numeric discreet variables,

- removing obvious outliers,
- scaling the features

By looking at the cleaned data, we can see the default rate below. That is, 15% loans defaulted for the time period represented by the data. Compared to the analysis for bank loan data, we can tell Lending Club loans suffered much higher default rate.

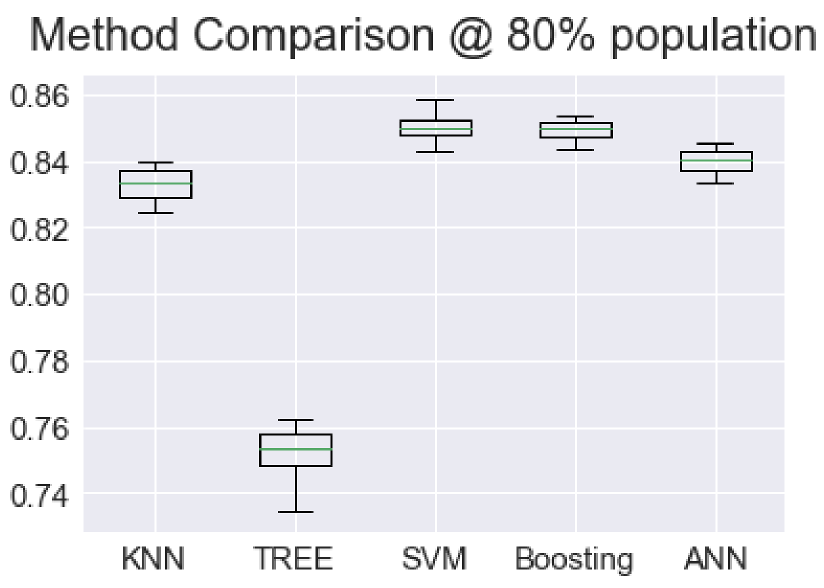


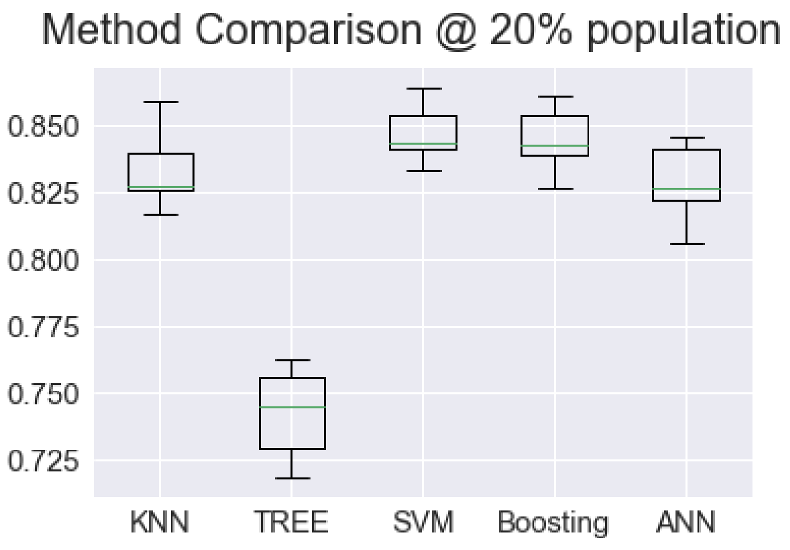
1.2 Algorithms Cross Validation

The following machine learning methods were used with Scikit-learn pyhon libraries:

1. k-nearest neighbors algorithm (KNN)
2. Classification Tree (TREE)
3. Support Vector Machines (SVM)
4. Adaboost Classifier (Boosting)
5. Artificial Neural Network (ANN)

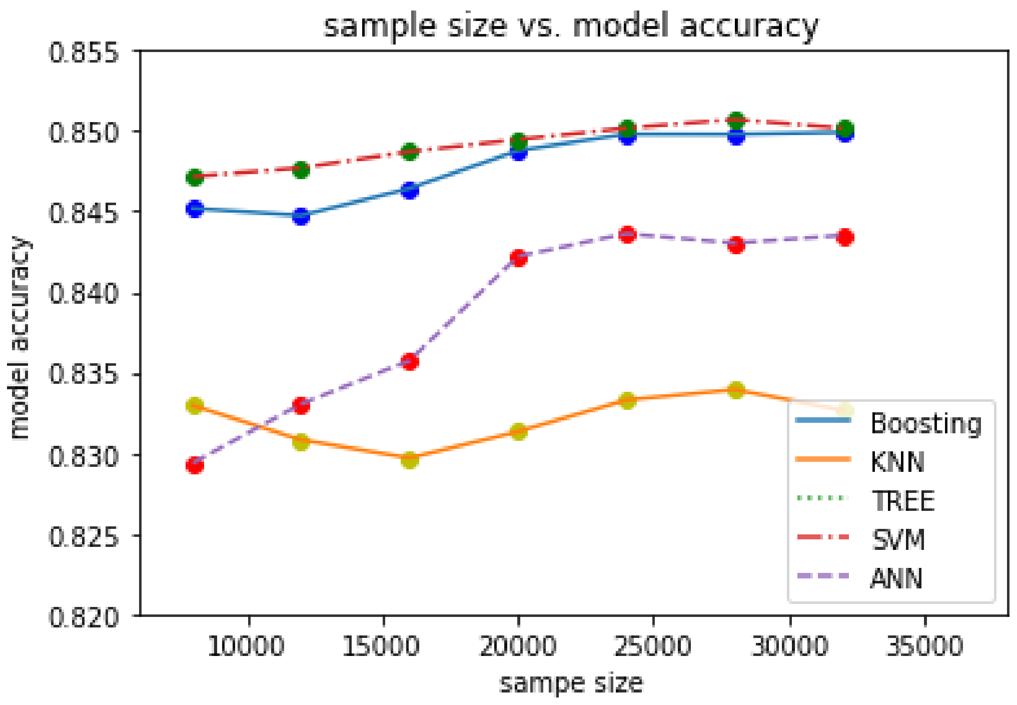
I used 80% of total dataset as training data, and the remaining 20% as test data. For the training data, I used 10 folds to explore model performance. The figure below shows the performance of each method. The boxplot shows both the mean and variance of model accuracy ratio. In general, the SVM method demonstrated the highest average model accuracy and relatively low variance. To test model performance with different training data size, I also performed the same analysis by using 20% of data as training data. The second boxplot shows the same pattern, i.e., SVM performed the best in terms of model accuracy ratio.





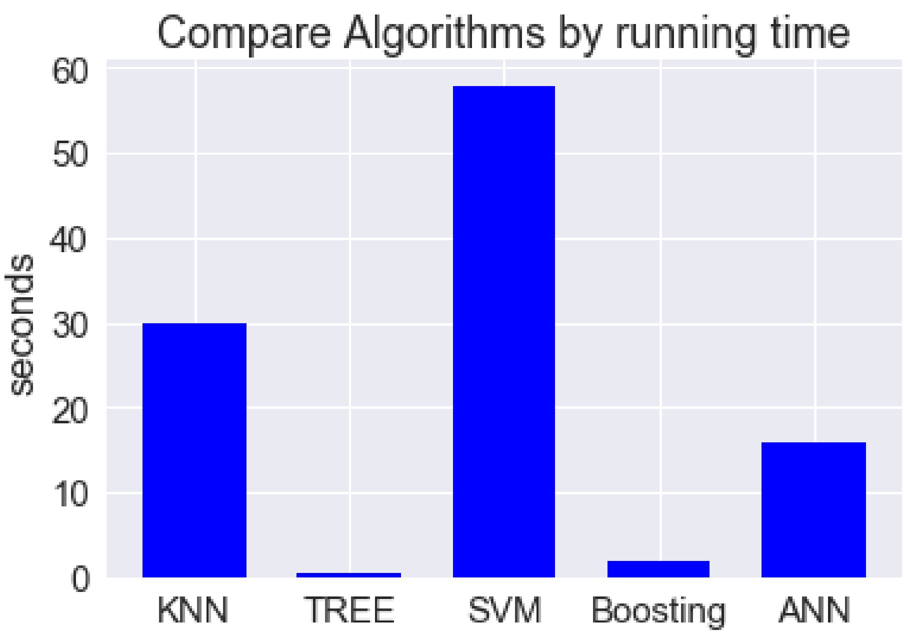
1.3 Algorithms Performance by Sample Size

To further evaluate model performance by sample size, I plotted model accuracy ratio vs. sample size across all classification methods. As we can see, the SVM method consistently showed best performance, especially for smaller sample size. However, it seems that the performance of Boosting and SVM gradually converged when data sample size was sufficiently large.



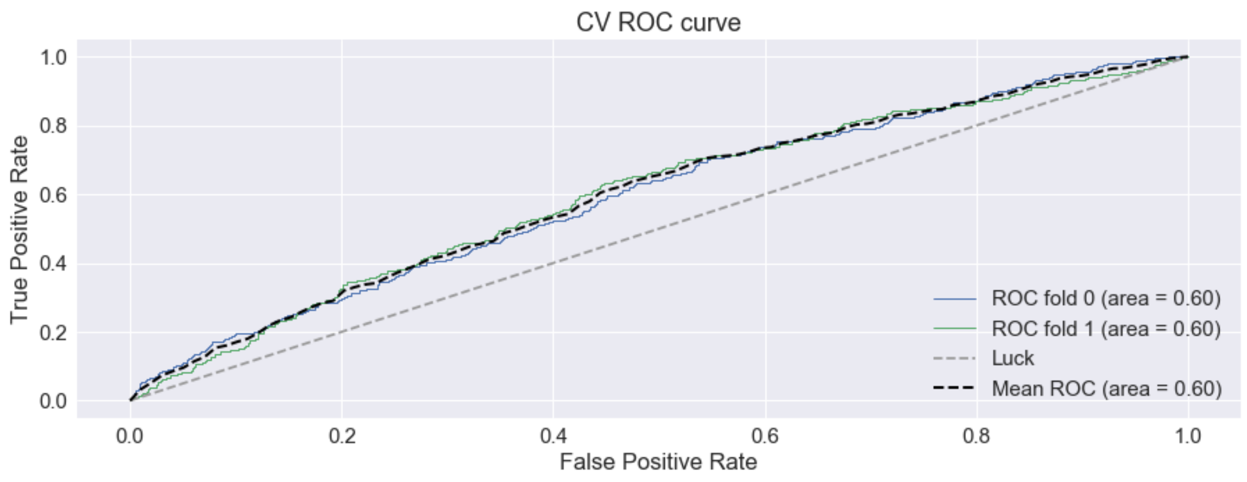
1.4 Model Complexity and Final Model Selection

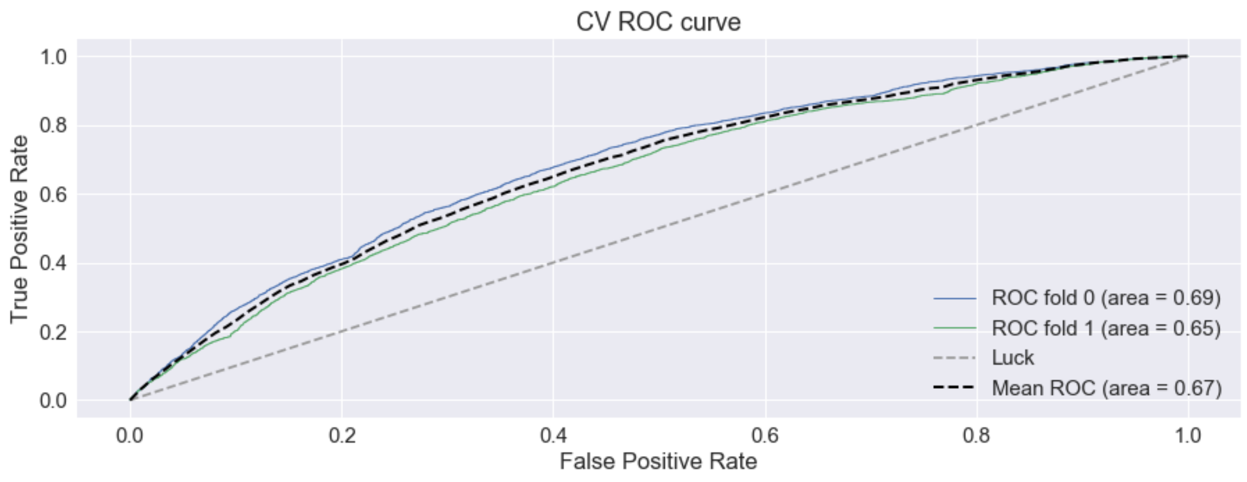
Based on the above analysis, the SVM was the selected as the model of the choice for this analysis. One challenge found was that the computational time for the SVM method seemed to be the worst. I further analyzed computational time across algorithms. Based on the chart below, SVM indeed had the worst computational efficiency. However, Boosting method provided the best computational efficiency while providing similar accuracy ration. Regardless, for this dataset, I chose SVM as the preferred method. Note that, if the dataset is too large, we may have to consider the Boosting method as the alternative given its advantage in computational efficiency.



1.5 Hyper Parameter Analysis and Testing Data Model Results

GridSearchCV in Scikit was used to explore hyper parameter tuning. From the previous analysis, model error variance is not a major concern for SVM. The tuning was focused on the choice of kernel, regularization parameter C, and gamma. Given the three choices of kernel, 'linear', 'poly', and 'rbf', the suitable kernel is 'rbf'. The first ROC chart below is for 'linear' kernel. Compared to the 'rbf' kernel, the 'linear' kernel performed poorly. Therefore, we used the default 'rbf' kernel. In addition, by setting up search grids including C and Gamma, the optimal grid is selected: {'C': 0.01, 'gamma': 0. 01}. The second ROC chart shows the performance of model with selected hyper parameters.





The results of final model using testing data are given below:

Accuracy Score

0.8508771929824561

Confusion Matrix

```
[[6884    0]
 [1207    3]]
```

Classifications Report

```
             precision    recall  f1-score   support
          0       0.85      1.00      0.92      6884
          1       1.00      0.00      0.00      1210
avg / total       0.87      0.85      0.78      8094
```

Therefore, we can conclude that the chosen model was suitable for the dataset.

Dataset 2– Bank Loan Performance Data

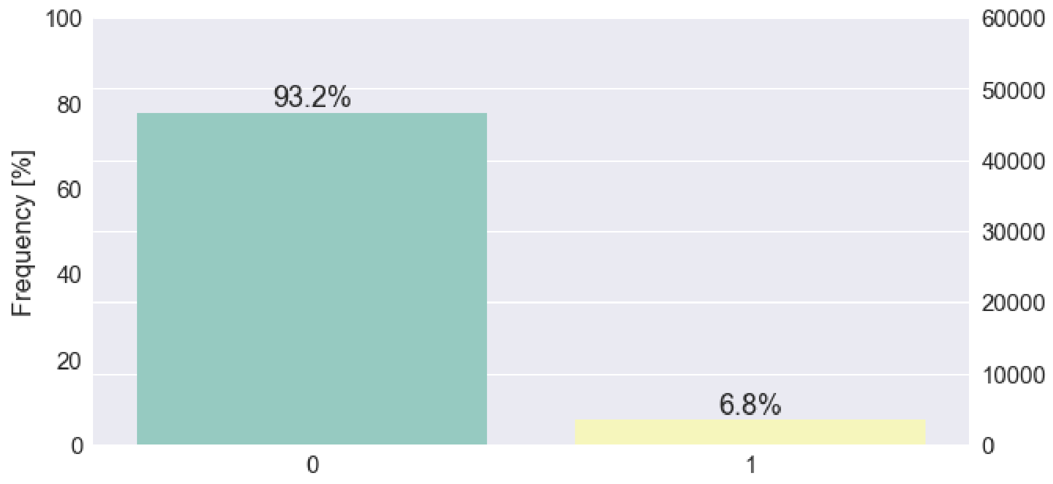
2.1 Data Description, Preprocessing and Feature Engineering

The bank loan data from kaggle include over 150K records and 11 features as show in table below. The label or predicting target is the loan default status. The following data preprocessing was used to prepare the data for analysis: removing obvious outliers, and scaling the features. Given the large amount of data, we dropped very small portion of rows containing NaN. Further analysis (in two boxplots) proved that this drop would not cause biased analysis because before and after cleaning default rate was very similar, 6.8% vs. 6.6%.

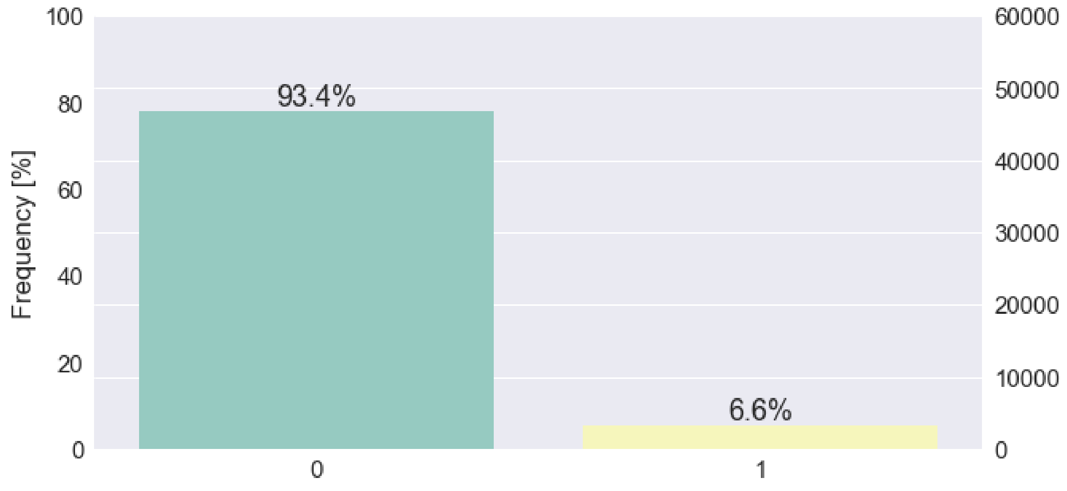
Columns	Type
SeriousDlqin2yrs	**Y/N**
RevolvingUtilizationOfUnsecuredLines	%
age	integer
NumberOfTime30-59DaysPastDueNotWorse	integer
DebtRatio	%
MonthlyIncome	numeric
NumberOfOpenCreditLinesAndLoans	integer
NumberOfTimes90DaysLate	integer
NumberRealEstateLoansOrLines	integer
NumberOfTime60-89DaysPastDueNotWorse	integer
NumberOfDependents	integer

By looking at the cleaned data, we can see the default rate below. That is, 6.6% loans defaulted for the time period represented by the data. Compared to the analysis for Lending Club loan data, we can tell bank loans are usually with higher credit quality because of much lower default rates, 6.6% vs. 15.0% (Lending Club).

Before cleaning



After cleaning (not biased)

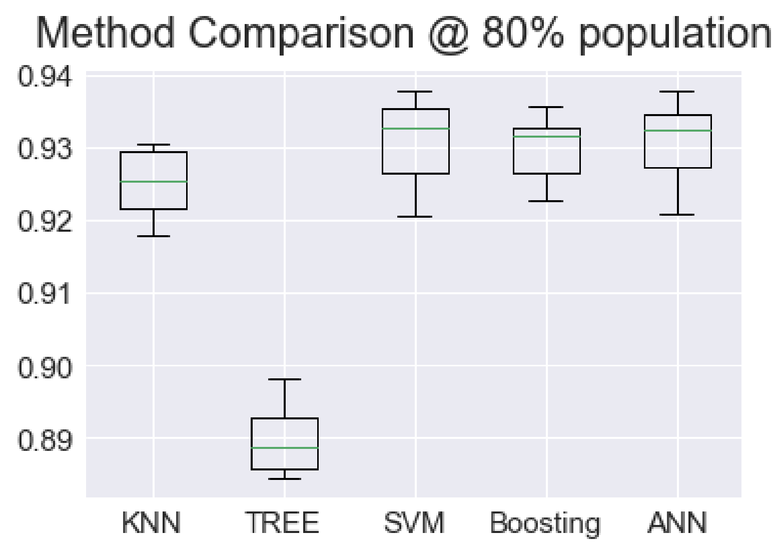


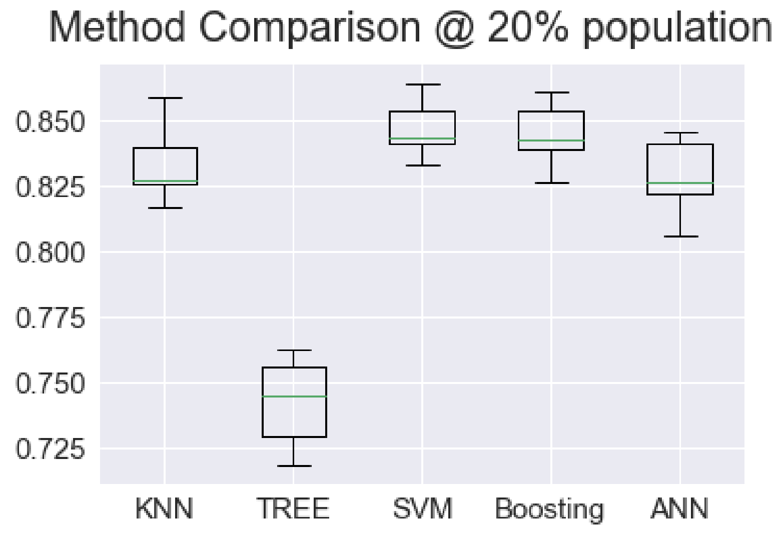
2.2 Algorithms Cross Validation

I performed the same cross validation as for Lending Club data. The following machine learning methods were used with Scikit-learn pyhon libraries: KNN, TREE, SVM, Boosting, and ANN. Similarly, I used 80% of total dataset as training data, and the remaining 20% as test data. For the training data, I used 10 folds to explore model performance. The figure below shows the performance of each method. The boxplot shows both the mean and variance of model accuracy ratio.

In general, the SVM method demonstrated the highest average model accuracy. However, Boosting and ANN methods have comparable model accuracy, but relatively low variance. To test model performance with different training data size, I also performed the same analysis by using 20% of data as training data. The second boxplot shows that SVM and Boosting performed similarly in terms of model accuracy ratio.

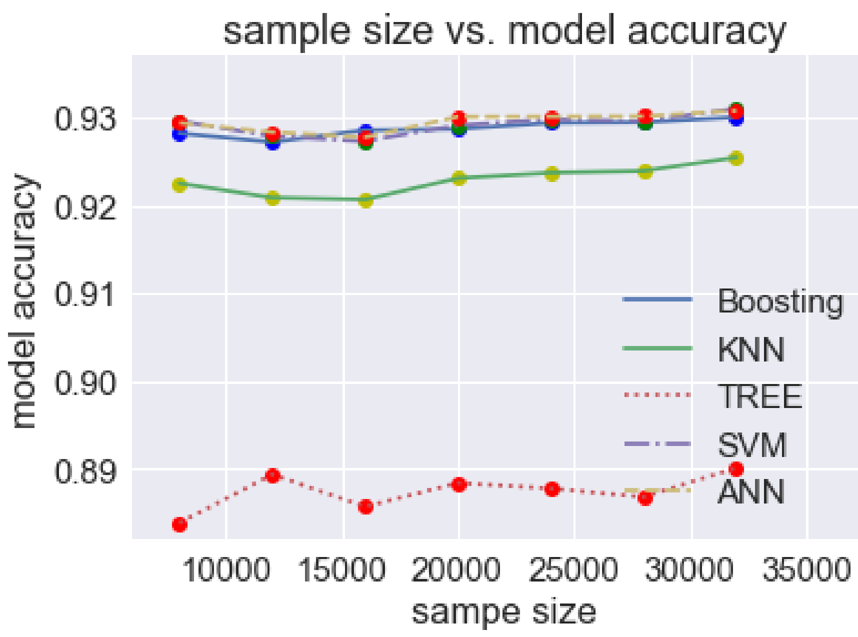
Overall, if we balance all the factors, such as accuracy ratio, variance and performance at smaller sample size, Boosting and SVM are top two choices.





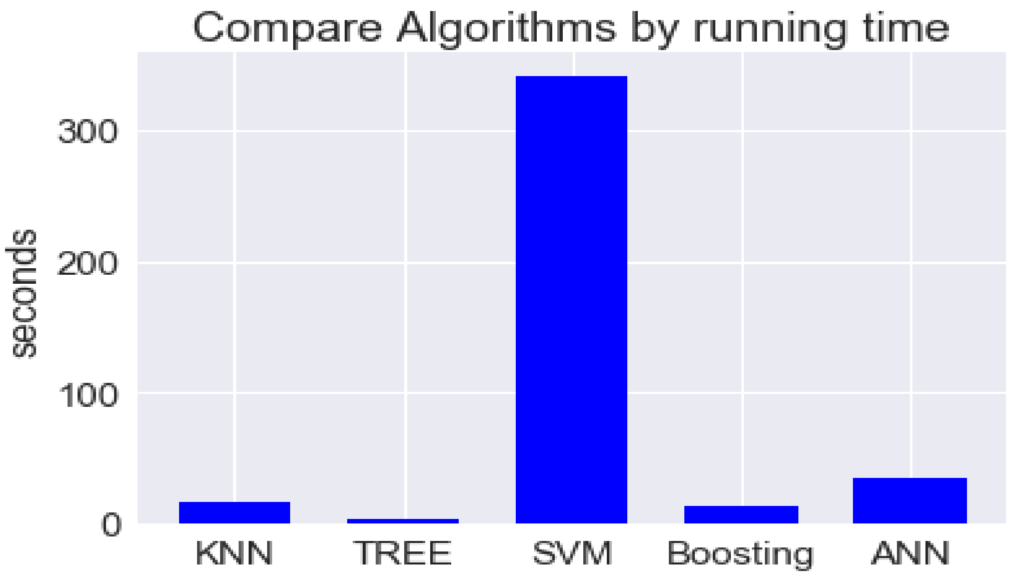
2.3 Algorithms Performance by Sample Size

To further evaluate model performance by sample size, I plotted model accuracy ratio vs. sample size across all classification methods. As we can see, when sample size is over 25,000, the performance for the best three methods, SVM, Boosting, and ANN, nearly converged.



2.4 Model Complexity and Final Model Selection

Based on the above analysis, the SVM and Boosting were two best models based on accuracy ratio. One challenge found was that the computation time for the SVM method was the worst. I further analyzed computational time across algorithms. Based on the chart below, SVM indeed had the worst computational efficiency. However, Boosting method provided the best computational efficiency while providing similar accuracy ratio. Therefore, given the significant advantage in computational time and the large amount of data size, I chose Boosting as the preferred method.



2.5 Hyper Parameter Analysis and Testing Data Model Results

GridSearchCV in Scikit was used to explore hyper parameter tuning. DecisionTreeClassfier was selected as base_estimator. The algorithm used was SAMME.R. The tuning was focused on the choice of n_estimators and learning_rate given the fact of there is a tradeoff between the two parameters.

By setting up search grids including n_estimators and learning_rate , the optimal grid is selected: {'learning_rate': 0.01, 'n_estimators': 300}. The first ROC chart below is for the initial state of the search, which shows a less optimal model choice. The second ROC chart shows the performance of model with selected hyper parameters.

The results of final model using testing data are given below:

Accuracy Score

```
0.9323
```

Confusion Matrix

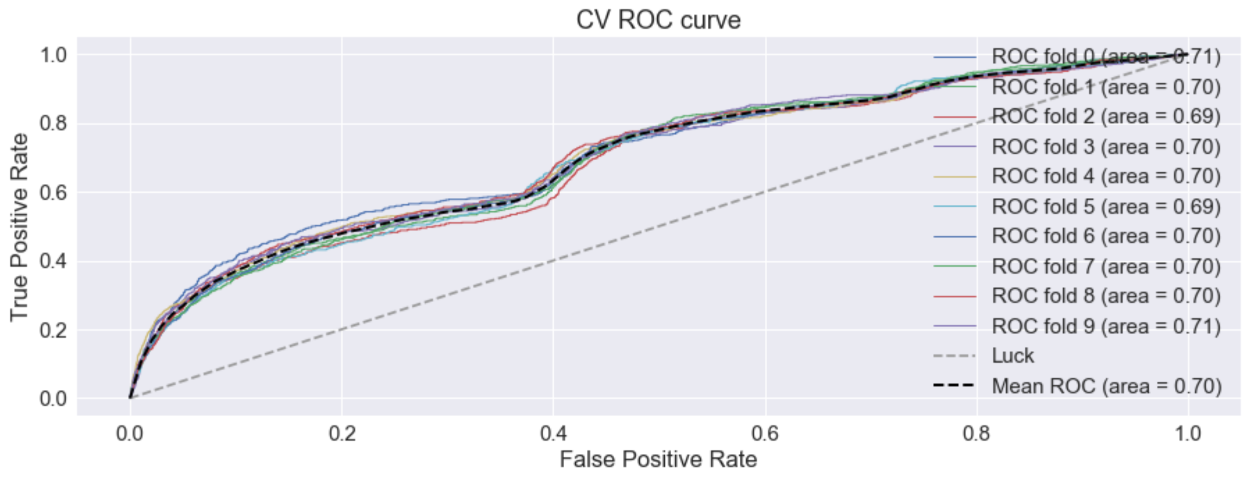
```
[[27541   383]
 [ 1648   428]]
```

Classifications Report

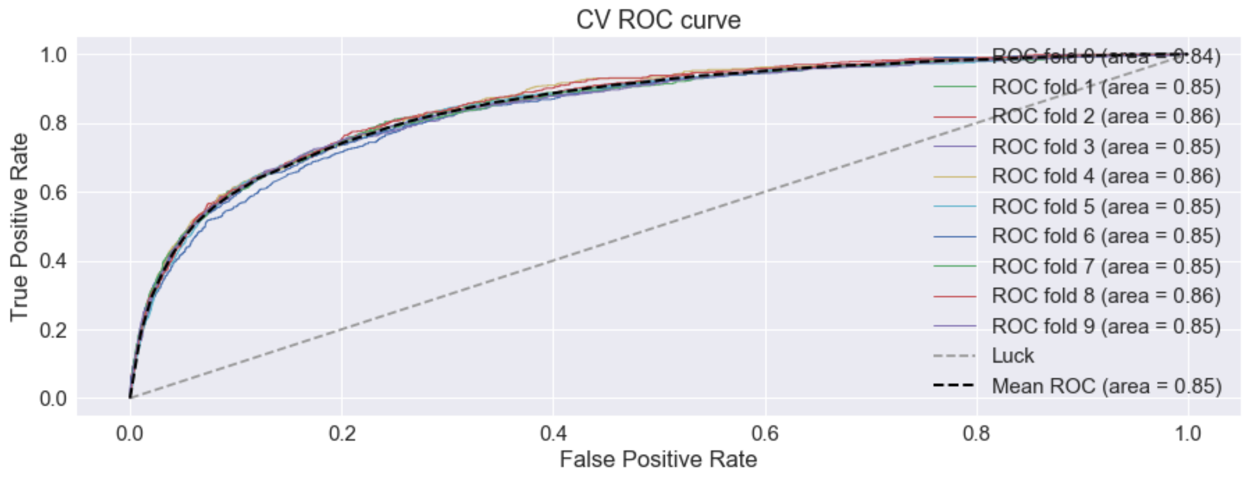
```
             precision    recall  f1-score   support
          0       0.94      0.99      0.96     27924
          1       0.53      0.21      0.30      2076
avg / total       0.91      0.93      0.92     30000
```

Therefore, we can conclude that the chosen model was suitable for the dataset.

Initial State of the Search



Optimal Search Results



Conclusion

This study analyzed two datasets: p-to-p loans and conventional bank loans. Two methods, SVM and Boosting, performed similarly in predicting loan defaults. The choice of final model depended on both accuracy ratio and computational time. For a larger amount of loans, Boosting method was chosen over SVM because of its computational efficiency.

It is worth noting that Lending Club loans contained a lot more features (~150) , but yielded lower accuracy ratio compared to Bank Loans (11 features) because of lower data quality and fewer more relevant data fields. The Lending Club loans also had much higher default rates. Both lower data quality and higher default rates indicated Lending Club loans had lower underwriting credit standards.

Notes:

1. To run the code, someone needs to change the working director path to load the data. Some intermediate results are outputted previously as objects and are loaded in the code to save running time.