CREDIT VALUATION MODELS

ECO 412
Fundamentals of Big Data
Gary Vartanian 804195815
Jiahui Xu 705055197
Kang Niu 405054744
Yanli Liu 605058747
Yuxuan Zhao 004880206

Background

In banking, loans are the primary source of income. Banks make money based off the spread they have from borrowing and then lending out the borrowed money to credible customers, businesses, and people. Because of this, the emphasis is on finding credit worthy and trust worthy customers to provide lending to. A bank would want to loan only to members that are safe and most importantly, prevent loans being given out to customers that would default. This is why FICO scores and means of tracking and rate credit are so important. With the rise of data science and machine learning community, banks recently challenged members to come up with a better model that can detect default.

For a machine learning task this can be categorized as a classification problem. Find the probability that someone would default or provide payment. The team tackled the problem by using statistical learning methods: Logistic Regression, Probit Regression, Stepwise Logistic (forward, backward, and both), and Regularization of Logistic Regression using Lasso and Ridge penalty. In addition to that, the team also conducted a few machine learning methods such as Decision Trees and Random Forest. The main reason for these techniques is that all provide some sort of white-box (ez interpretability) to allow for insight, variable importance, and variable weight to them. Thereby, giving more details to bank. In the end, the model that did the best was Random Forest due to its non-parametric approach and ensembling design. Furthermore, most, if not all, regressions seem to indicate that Payment History is the primary predictor whether someone would default in future or not.

Source of Data

The data was posted on Kaggle as part of the “Give me Some Credit” Competition.
Kaggle Link: https://www.kaggle.com/c/GiveMeSomeCredit

The main objective is to improve on the state of the art in credit scoring by predicting the probability that somebody will experience financial distress in the next two years.
Banks play a crucial role in market economies. They decide who can get finance and on what terms and can make or break investment decisions. For markets and society to function, individuals and companies need access to credit. Credit scoring algorithms, which make a guess at the probability of default, are the method banks use to determine whether or not a loan should be granted. This competition requires participants to improve on the state of the art in credit scoring, by predicting the probability that somebody will experience financial distress in the next two years.
The goal of the competition is to build a model that borrowers can use to help make the best financial decisions. Historical data are provided on 250,000 borrowers and the prize pool is $5,000 ($3,000 for first, $1,500 for second and $500 for third).

Data Variable Description

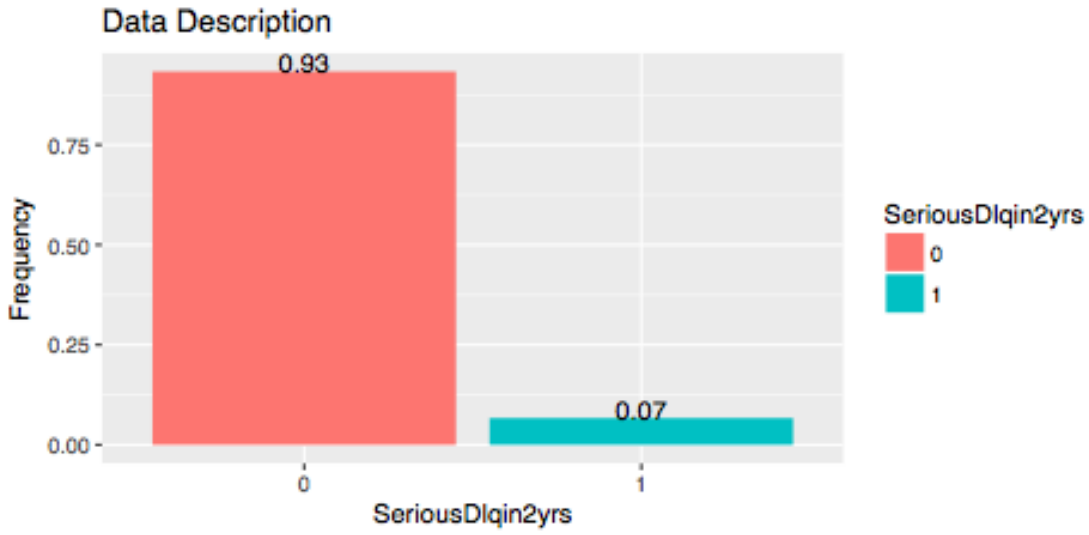
SeriousDlqin2yrs	Person experienced 90 days past due delinquency or worse
RevolvingUtilizationOfUnsecuredLines	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits
age	Age of borrower in years
NumberOfTime30-59DaysPastDueNotWorse	Number of times borrower has been 30-59 days past due but no worse in the last 2 years.
DebtRatio	Monthly debt payments, alimony,living costs divided by monthy gross income
MonthlyIncome	Monthly income
NumberOfOpenCreditLinesAndLoans	Number of Open loans (installment like car loan or mortgage) and Lines of credit (e.g. credit cards)
NumberOfTimes90DaysLate	Number of times borrower has been 90 days or more past due.
NumberRealEstateLoansOrLines	Number of mortgage and real estate loans including home equity lines of credit
NumberOfTime60-89DaysPastDueNotWorse	Number of times borrower has been 60-89 days past due but no worse in the last 2 years.
NumberOfDependents	Number of dependents in family excluding themselves (spouse, children etc.)

Split of Train and Test

An equal balance of 2500 of 0's and 1's were taken split from the total dataset and treated as a testing data. Everything else was put in training. The reason being would be at a 50% split for the training and testing, it would be much easier to distinguish the added power of the model and highlight any bias that any of the models might have.

Dealing with Imbalance

Introduction of Data Imbalance



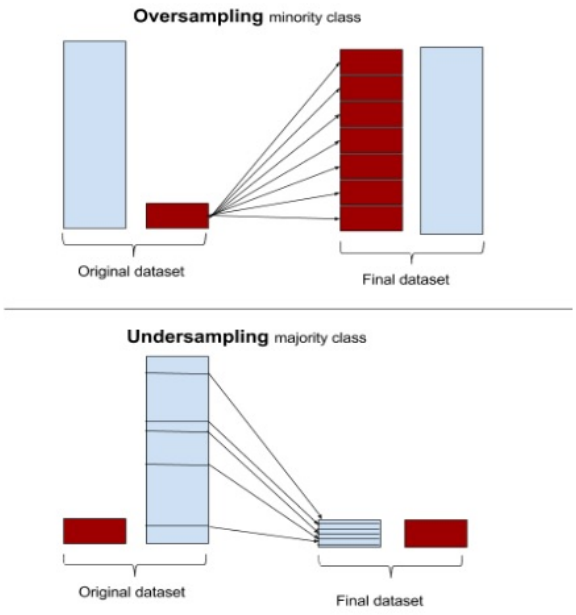
As you can see from the image above, that default happens rather little in our dataset. It only occurs 7% of the 250,000 data points we have (17,500). The data is highly skewed to those that do not default, because traditionally banks try not to loan out to people that would not pay back. Because of this, the team had a fear that without cleaning the data, any model would be biased towards 0 and inadequately capture the number of 1's well. Remember, it would cost more to the bank to lose the money then not loan it out. For comparison, all models were run using this data. It did well on the training, but could not hold well in the testing data. Furthermore, it did little to improve upon the parameters that the bank would follow. Therefore to mediate the situation, there were 5 different balancing techniques that we applied.

Undersampling Majority:

As the name implies, this method is to help minimize the data imbalance that occurs by undersampling the majority class. Meaning take a small sample of the majority class in this case 0 and put it with the minority data . This is to help create a more balanced data set. Its pro is that it balanced it and reduced run time and storage time (less data). However, its shortcoming are that it leaves data behind and the random sample might not be representative.

Oversampling Minority

This is the oppositve of undersampling as it oversamples the minority class (1) in this case. The pro is that there is no information loss, but provides a case of overfitting to the minority class.



Hybrid (Oversample and Undersample)

This is a combination of the two techniques used above: oversampling minority class and undersampling majority class. It brings with it the benefit of both.

Random Oversampling (ROSE)

The Random Oversampling Example produces balance using smoothed bootstrapped method. The random samples are generated form a kernel estimate of the conditional density. Essentially, you are trying to find the distribution that your data is coming from and resample it.

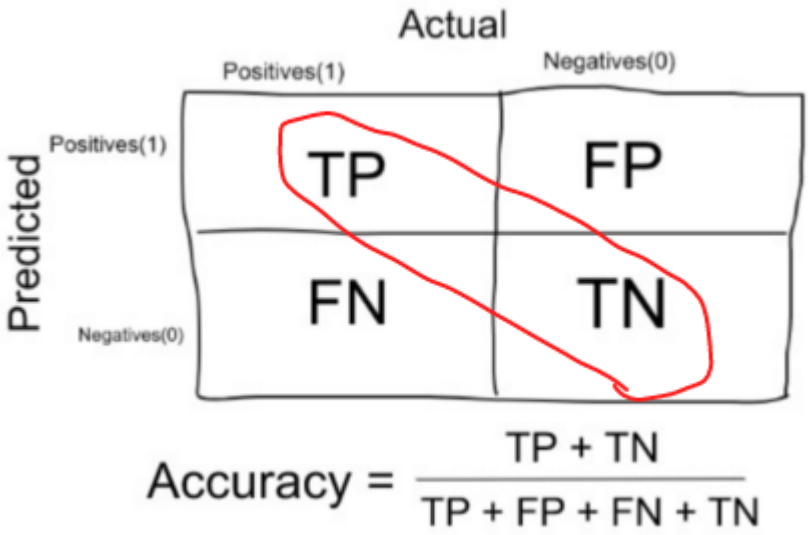
Synthetic Minority Over Sampling Technique (SMOTE)

Here you are creating our own data set by connecting lines between your existing data and then creating new synthetic data between those lines. The general idea is that anything within that vicinity is within that region. You then use k-nn and average the other variables for that synthetic point using the nearest neighbor.

Metrics Used

ACCURACY

Accuracy in classification problems is the number of correct predictions made by the model over all kinds predictions made.



In the Numerator, are our correct predictions (True positives and True Negatives)(Marked as red in the fig above) and in the denominator, are the kind of all predictions made by the algorithm(Right as well as wrong ones).

When to use Accuracy:
Accuracy is a good measure when the target variable classes in the data are nearly balanced.
Ex:60% classes in our fruits images data are apple and 40% are oranges.
A model which predicts whether a new image is Apple or an Orange, 97% of times correctly is a very good measure in this example.

When NOT to use Accuracy:
Accuracy should NEVER be used as a measure when the target variable classes in the data are a majority of one class.

Ex: In our example with 100 people, only 7 people default. Let's say our model is very bad and predicts every case as No Default. In doing so, it has classified those 93 non-defaulters correctly and 7 defaulters as non-defaulters. Now even though the model is terrible at predicting default, The accuracy of such a bad model is also 93%

OUR CASE: BEST NOT TO USE ACCURACY

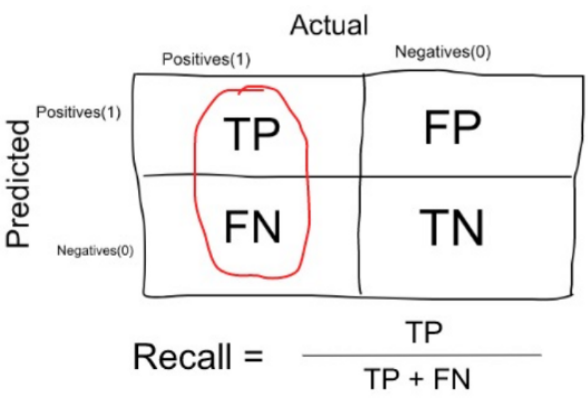
From the cases and description provided above, we know that accuracy is a misleading statistic because it doesn't properly capture the misclassification and tackle the huge data imbalance provided. For example, if we just chose zero for all of them we would be **93%** correct all the time, but as a result would face the maximum lost of providing funding to that **7%** that defaulted. Therefore, accuracy would not be a good use for skewed data and something that has a higher cost function for FALSE NEGATIVE.

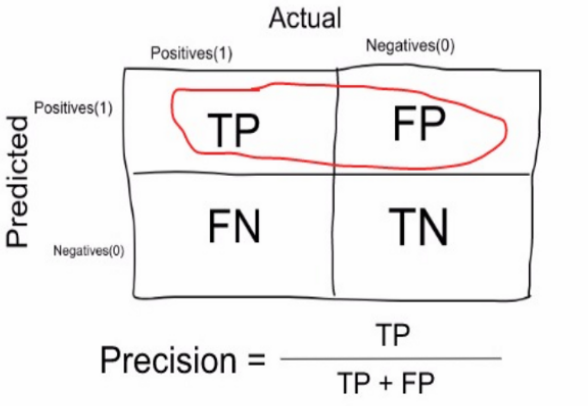
Precision:

Precision is a measure that tells us what proportion of people that we diagnosed as having defaulting, actually defaulted. The predicted positives (People predicted as defaulting are TP and FP) and the people actually defaulting are TP.
*Ex: In our banking example with 100 people, only 7 people default. Let's say our model is very bad and predicts every case as **Defaulter**. Since we are predicting everyone as defaulting, our denominator(True positives and False Positives) is 100 and the numerator, person defaulting and the model predicting his case as cancer is 7. So in this example, we can say that **Precision** of such model is 7%.*

Recall or Sensitivity:

Recall is a measure that tells us what proportion of clients that actually defaulted was predicted by the algorithm as defaulted. The actual positives (People defaulting are TP and FN) and the people diagnosed by the model having defaulting are TP. (Note: FN is included because the Person defaulted even though the model predicted otherwise).
Ex: In our cancer example with 100 people, 7 people actually defaulted. . Let's say that the model predicts every case defaults.
*So our denominator(True positives and False Negatives) is 7 and the numerator, person defaulting and the model predicting his case as default is also 7. So in this example, we can say that the **Recall** of such model is 100%. And Precision of such a model(As we saw above) is 7%*





When to use Precision and When to use Recall?:

It is clear that recall gives us information about a classifier's performance with respect to false negatives (how many did we miss), while precision gives us information about its performance with respect to false positives(how many did we caught).

Precision is about being precise. So even if we managed to capture only one cancer case, and we captured it correctly, then we are 100% precise.

Recall is not so much about capturing cases correctly but more about capturing all cases that have "cancer" with the answer as "cancer". So if we simply always say every case as "cancer", we have 100% recall.

So basically if we want to focus more on minimising False Negatives, we would want our Recall to be as close to 100% as possible without precision being too bad and if we want to focus on minimising False positives, then our focus should be get Precision as close to 100%

In our case, we want to MAXIMIZE RECALL, but want a balance of the two. Step in F1

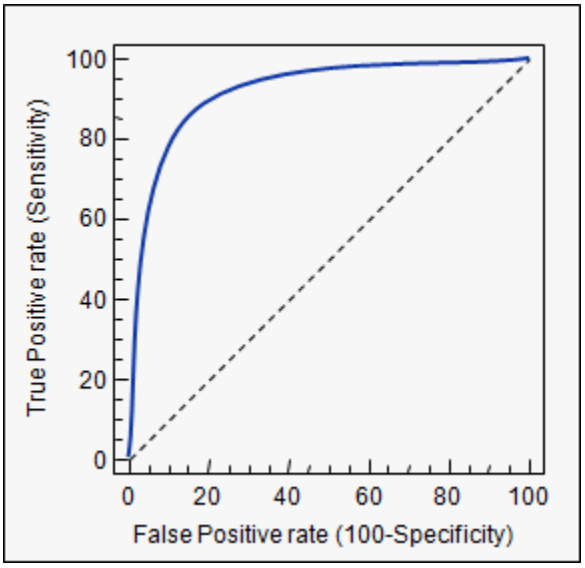
F1 Score (Combo of Both)

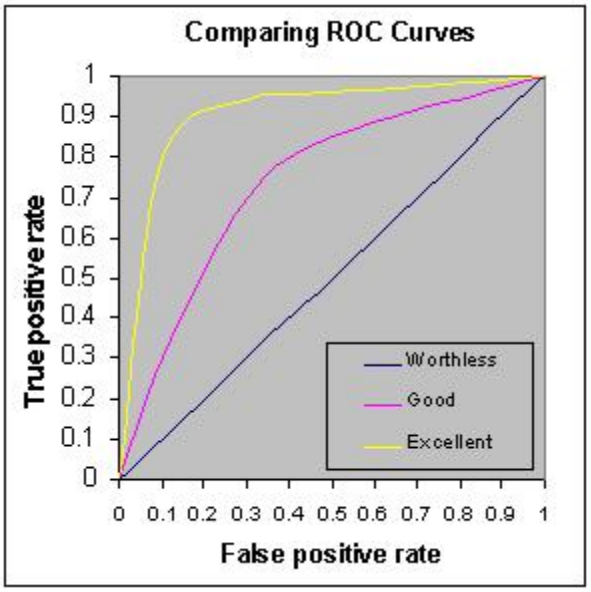
F1 score combines both Precision(P) and Recall(R) into a neat formula. Because it takes both factors into account. We will be using this as a golden metric.
Ideally, we want to maximize this score:
2 X (Precision X Recall) / (Precision + Recall)

ROC Curve

In a Receiver Operating Characteristic (ROC) curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (100-Specificity) for different cut-off points. Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner (100% sensitivity, 100% specificity). Therefore the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test (Zweig & Campbell, 1993).

This helps us visually see which models would be better. This is the second golden critera we used to help rate models.





Models

Probit regression

	Probit (F1 score)	Probit (auc)
Training_data	0.670	0.507
both_sample_train_data	0.672	0.513
over_sample_train_data	0.704	0.620
rose_sample_train_data	0.601	0.611
smote_sample_train_data	0.686	0.548
under_sample_train_data	0.680	0.607

In the first step, we fit these datasets to a probit model. After checking the F1 score and accuracy rate in the above table, we chose the model built with the over_sample_train_data. According to the previous parts, the bigger F1 score and accuracy rate, the better. And the F1 score and accuracy rate in over_sample_train_data are 0.704 and 0.620 separately, which are the largest one. Thus, we choose this sample and get the following results.

	Dependent variable:
	SeriousDlqin2yrs
RevolvingUtilizationOfUnsecuredLines	0.0001
	(0.0002)
age	−0.030***
	(0.002)
NumberOfTime30.59DaysPastDueNotWorse	0.642***
	(0.028)
DebtRatio	−0.0001***
	(0.00002)
MonthlyIncome	−0.00003***
	(0.00000)
NumberOfOpenCreditLinesAndLoans	0.006
	(0.004)
NumberOfTimes90DaysLate	0.666***
	(0.040)
NumberRealEstateLoansOrLines	0.110***
	(0.020)
NumberOfTime60.89DaysPastDueNotWorse	0.305***
	(0.047)
NumberOfDependents	0.036**
	(0.018)
Constant	1.330***
	(0.082)
Observations	12,542
Log Likelihood	−6,695.000
Akaike Inf. Crit.	13,412.000
Note:	*p<0.1; **p<0.05; ***p<0.01

This table above reports the estimation results by implementing probit model. We can see that with an increase in age, Debt Ratio, Monthly Income, the probability of default is lower. With an increase in Number of Real Estate Loans or Lines, Number of Times borrower has been 60-89 days past the due date but no worse in last 90 days, the probability of default is higher. Checking the accuracy rate, we can see that the accuracy rate is around 55%, which is not so good. So we fit these datasets to another model. Logistic model.

Logistic Regression

	Logistic (F1 score)	Logistic (auc)
Training_data	0.736	0.713
both_sample_train_data	0.716	0.699
over_sample_train_data	0.740	0.671
rose_sample_train_data	0.604	0.614
smote_sample_train_data	0.746	0.670
under_sample_train_data	0.672	0.604

Again, based on the F1 scores and accuracy rate, we chose the model with the largest F1 score and accuracy rate, which is the smote_sample_train_data.

	Dependent variable:
	SeriousDlqin2yrs
RevolvingUtilizationOfUnsecuredLines	−0.0001*
	(0.0001)
age	−0.029***
	(0.001)
NumberOfTime30.59DaysPastDueNotWorse	0.619***
	(0.013)
DebtRatio	−0.0001***
	(0.00001)
MonthlyIncome	−0.0001***
	(0.00000)
NumberOfOpenCreditLinesAndLoans	0.003
	(0.002)
NumberOfTimes90DaysLate	0.597***
	(0.018)
NumberRealEstateLoansOrLines	0.088***
	(0.010)
NumberOfTime60.89DaysPastDueNotWorse	−0.002
	(0.022)
NumberOfDependents	0.061***
	(0.009)
Constant	0.836***
	(0.040)
Observations	52,682
Log Likelihood	−29,031.000
Akaike Inf. Crit.	58,083.000
Note:	*p<0.1; **p<0.05; ***p<0.01

The table above reports the estimation results by implementing logistic model and we can get a similar result with probit model. With an increase in age, Debt Ratio, Monthly Income, the probability of default is lower. With an increase in Number of Real Estate Loans or Lines the probability of default is higher. But the variable Number of Times borrows has been 60-89 days past the due date but no worse in last 90 days is statistically insignificant in this logistic model. In addition, we can see from the accuracy rate that logistic model performs a little better than probit model. But they are still not desirable. So we fit other machine learning models in the following parts.

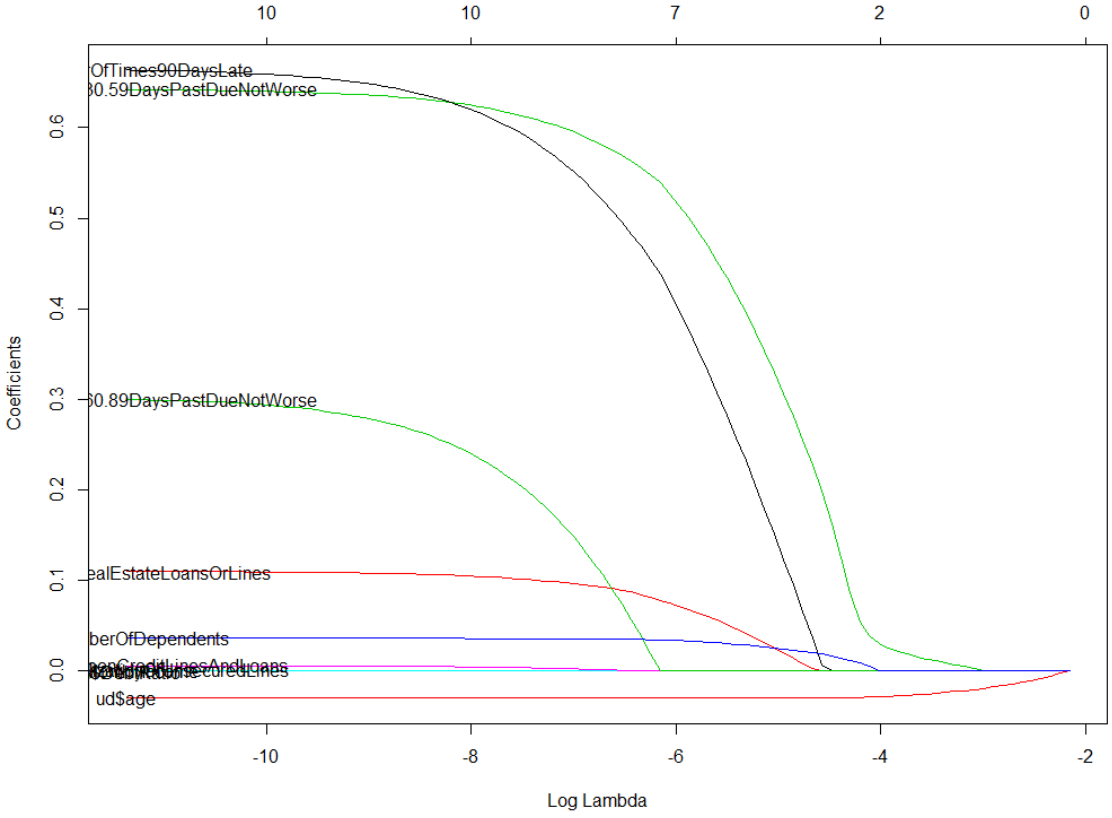
LASSO

In our project, we want to know which variables are the most important and which are not when predicting the defult rate. LASSO provides an answer to this question which is also highly interpretable. We still have six datasets for this part, and these dataset are generated by different re-sampling method talked earlier.

We regressed all the 11 variables on defult status, which is a binominal variable. 0 stands for non-defult, 1 stands for defult. We did cross validation for each dataset, and choose the best lambda for each dataset accordingly.

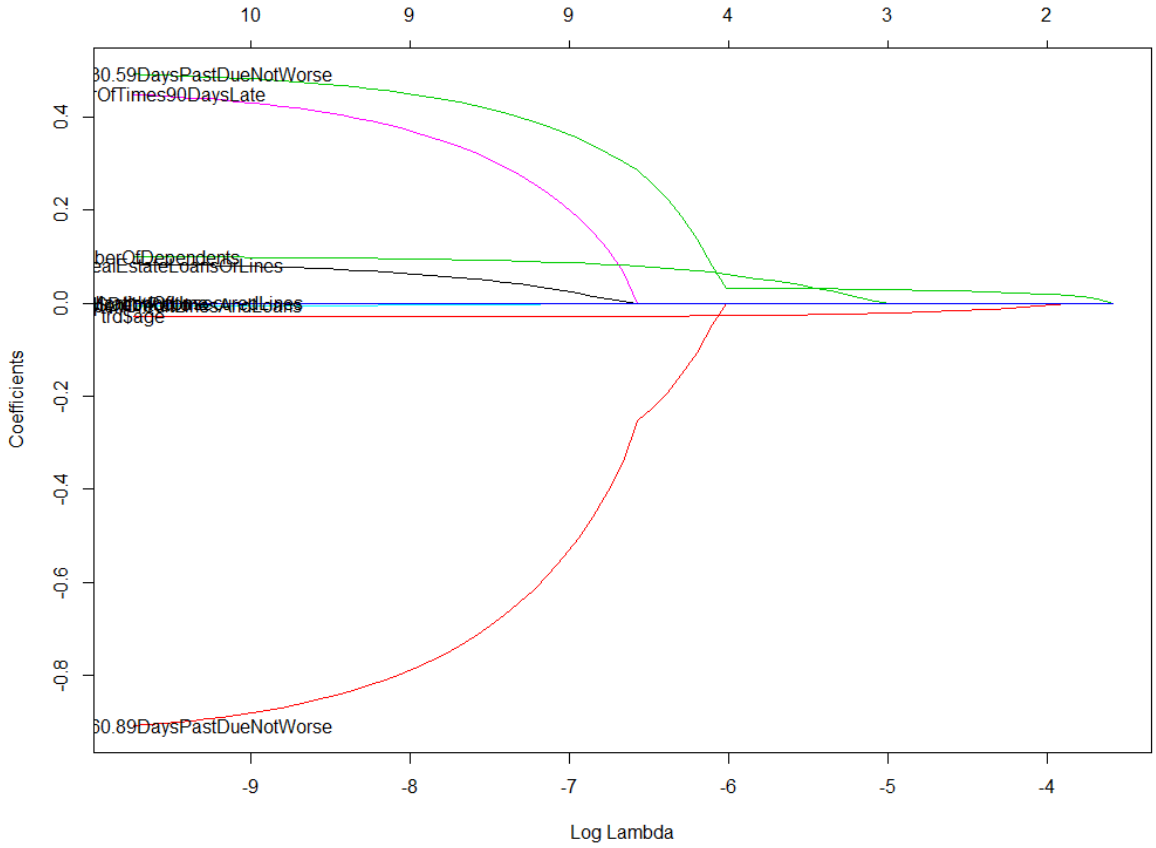
And with all these six regressions for six datasets, we run the result on the test dataset, which is a new dataset that we can used to check for accuracy. We check model's predicting ability based on two things, the result from the confusion matrix, and the result from the ROC curve. After comparing all the results, we found under sample dataset is the one with the biggest area under ROC curve, and the highest F1 value.

Thus we fit our final model with this dataset,and the LASSO regression plot of this dataset is shown below:



The main result we found is that variables including the number of open credit lines and loans, revolving utilization of unsecured lines, age, number of dependents they are all less important variables when predicting the defult rate. Variables including the number of times of 30 to 59 days past due payment, 90 days late and 69 to 89 days late payment converge really slow, and they are thus more important variable. If there are customers with same number of loans come

to apply for loans, the financial institution should lend to the one with lower number of late payments.
Such results hold for almost every dataset we used except for the training dataset. The graph for the training dataset is shown below:



Where the 69 to 89 days late payment performance in the training data is counterintuitive. As shown in the graph below, this variable's coefficient is negative. We will get a counterintuitive interpretation for this result. A person with higher late payment times will have lower rate of defult. We believe the reason for this result is because of the imbalance of the dataset. The training dataset contains over 90% of data is assigned as non-defult. Thus whatever the value for 69 to 89 days late payment is, the result is more likely to be non-defult. Also other variables' coefficients are relatively small, this is also proves our guess for the imbalance dataset driven wrong prediction.

Stepwise Selection Regression

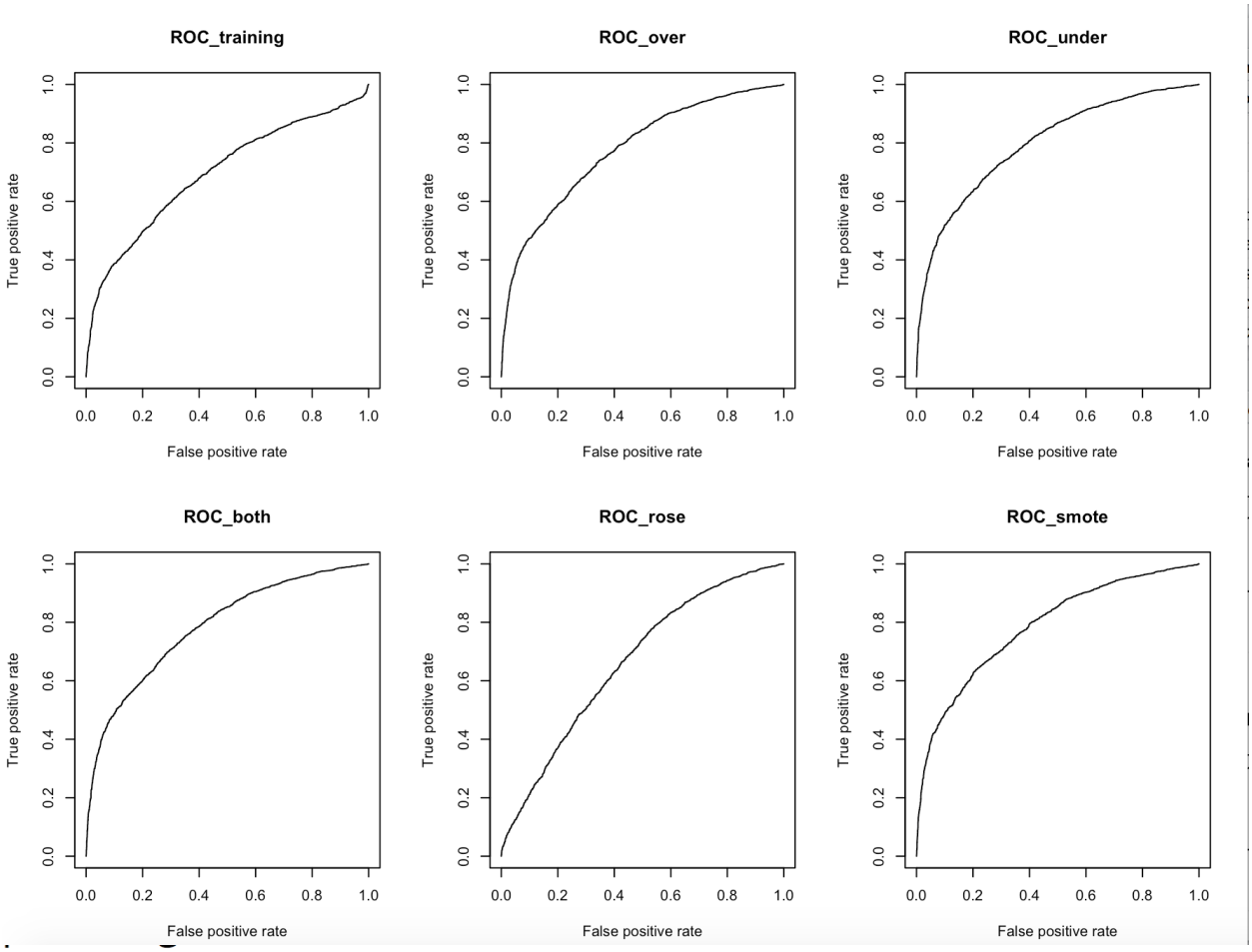
Stepwise selection is a method of fitting regression models by choosing different predictive variables. In each step, a variable is considered for addition to or subtraction from the set of predictive variables based on some prespecified criterion. This method including both forward and backward selection.

Forward selection is often used to provide an initial screening of the candidate variables when a large group of variables exists. For example, suppose you have fifty to one hundred variables to

choose from, way outside the realm of the all- possible regressions procedure. A reasonable approach would be to use this forward selection procedure to obtain the best ten to fifteen variables and then apply the all-possible algorithm to the variables in this subset. This procedure is also a good choice when multicollinearity is a problem.The forward selection method is simple to define. You begin with no candidate variables in the model. Select the variable that has the highest R-Squared. At each step, select the candidate variable that increases R-Squared the most. Stop adding variables when none of the remaining variables are significant. Note that once a variable enters the model, it cannot be deleted.

Backward selection model is less popular because it begins with a model in which all candidate variables have been included. However, because it works its way down instead of up, you are always retaining a large value of R-Squared. The problem is that the models selected by this procedure may include variables that are not really necessary. The user sets the significance level at which variables can enter the model. The backward selection model starts with all candidate variables in the model. At each step, the variable that is the least significant is removed. This process continues until no nonsignificant variables remain. The user sets the significance level at which variables can be removed from the model.

Because we take both ways into consideration, we should have 6 candidates, including models based on training dataset, over sample dataset, under sample dataset, both sample dataset, rose sample dataset and smote sample dataset, in total to compare. In order to find the best model, we calculated the accuracy, precision, recall, f1 and also AUC which is the area under Roc curve.



Accuracy Rates

	Stepwise (F1 score)	Stepwise (auc)
Training_data	0.6721	0.6955
both_sample_train_data	0.7166	0.7843
over_sample_train_data	0.6708	0.7766
rose_sample_train_data	0.6038	0.6630
smote_sample_train_data	0.7431	0.7857
under_sample_train_data	0.6708	0.7819

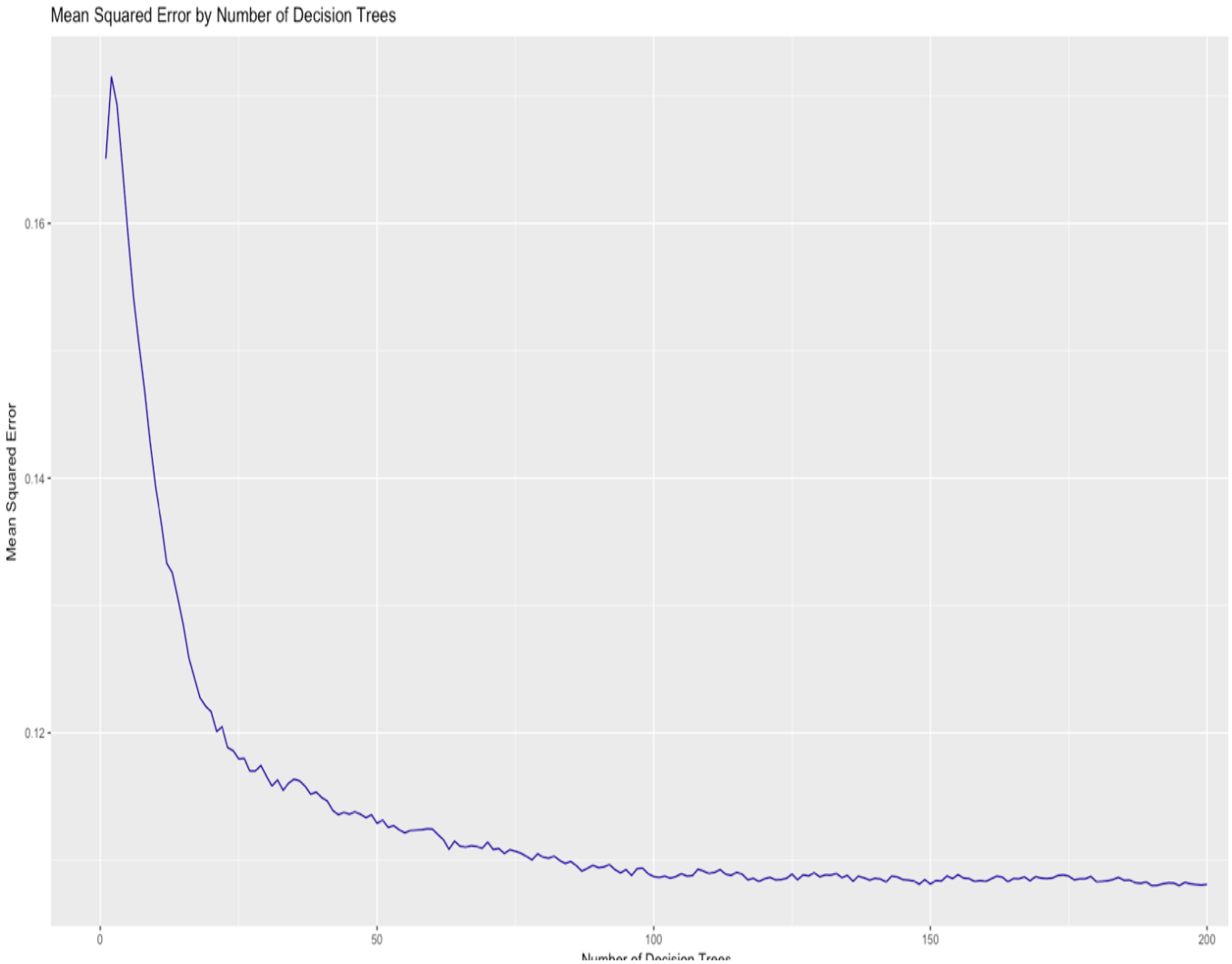
Based on the performance of these 6 models, we prefer higher F1 and auc, thus finally select the model using smote training data set, which includes 9 predictive variables and is as follows:

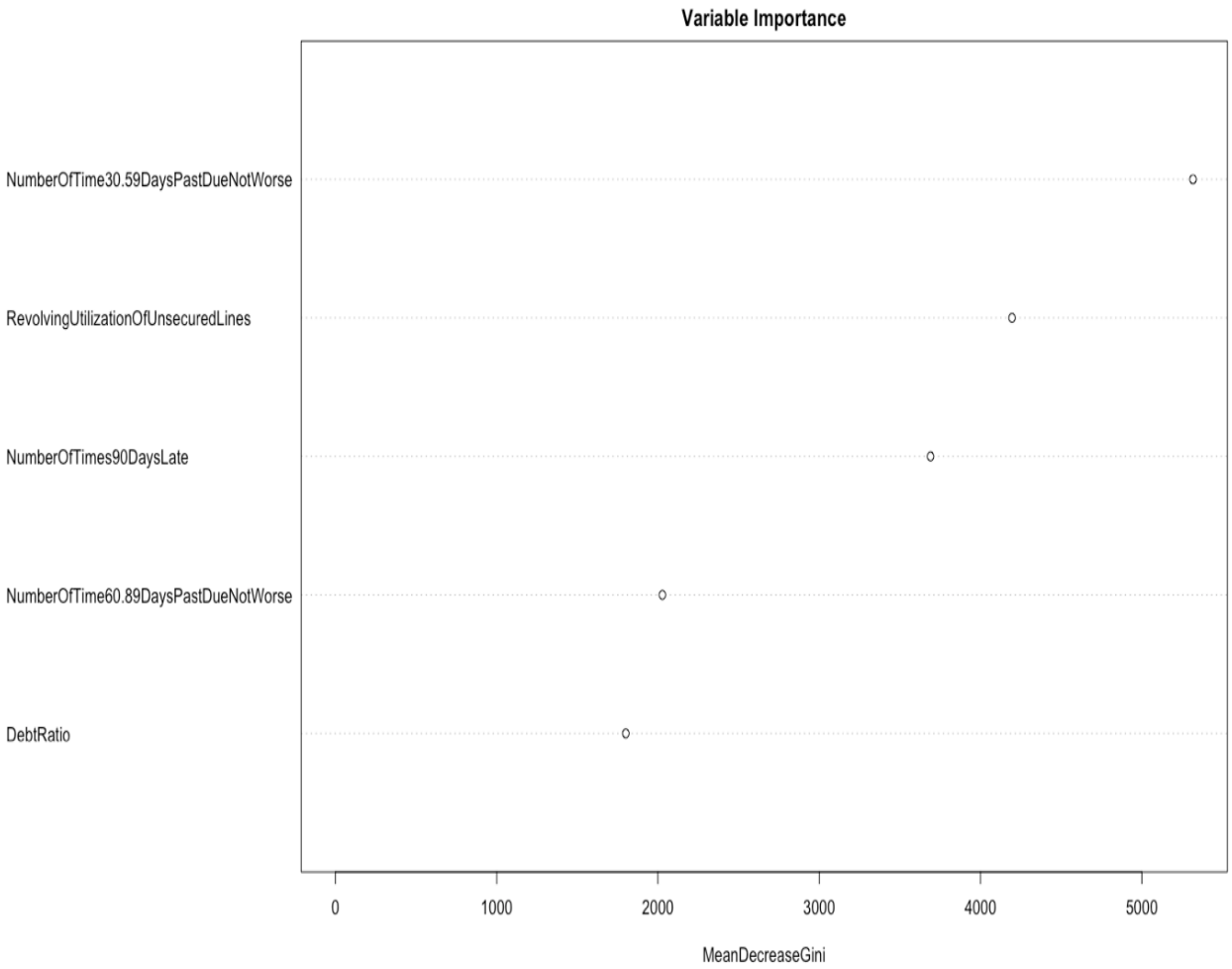
```
Coefficients:
                                        Estimate Std. Error z value Pr(>|z|)
(Intercept)                            9.091e-01  3.816e-02  23.822   <2e-16 ***
NumberOfTime30.59DaysPastDueNotWorse   6.222e-01  1.315e-02  47.317   <2e-16 ***
NumberOfTimes90DaysLate                6.014e-01  1.833e-02  32.813   <2e-16 ***
age                                   -3.020e-02  7.308e-04 -41.318   <2e-16 ***
MonthlyIncome                         -4.974e-05  2.702e-06 -18.411   <2e-16 ***
NumberRealEstateLoansOrLines           9.192e-02  9.783e-03   9.395   <2e-16 ***
DebtRatio                             -1.076e-04  1.028e-05 -10.475   <2e-16 ***
NumberOfOpenCreditLinesAndLoans        3.179e-03  2.191e-03   1.451   0.1469
NumberOfTime60.89DaysPastDueNotWorse   3.717e-03  2.200e-02   0.169   0.8658
RevolvingUtilizationOfUnsecuredLines -1.217e-04  7.090e-05  -1.716   0.0861 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Random Forest

The random forest model is particularly powerful for credit fraud detection because it leverages the considerable strength of decision trees including handling non-linear relationships, being robust to outliers and intuitive interpretability. Unlike decisions trees, the random forest model can be viewed as an ensemble of decision tree models. As a result, the random forest model is robust to overfitting and generally delivers more accurate result than the decision tree model.

Random Forests are an ensemble of k untrained Decision Trees (trees with only a root node) with M bootstrap samples (k and M do not have to be the same) trained using a variant of the random subspace method or feature bagging method. The graph below shows that as the number of decision tree increases, the mean squared error declines exponentially ultill the number of decision tree reaches 100. We choose tree size =100 for all training data.





This graph provides us with a sorted list of predictor importance. Larger “Gini” values indicate that the variable is more important. Strangely, it choose number of past due 30 to 59 days as the most important factor. Even though past due 60 to 89 days should provide strong evidence for default. Perhaps this is because the there are very few people that past due over 60 days. As a result there is not enough training data for the machine learning algorithm.

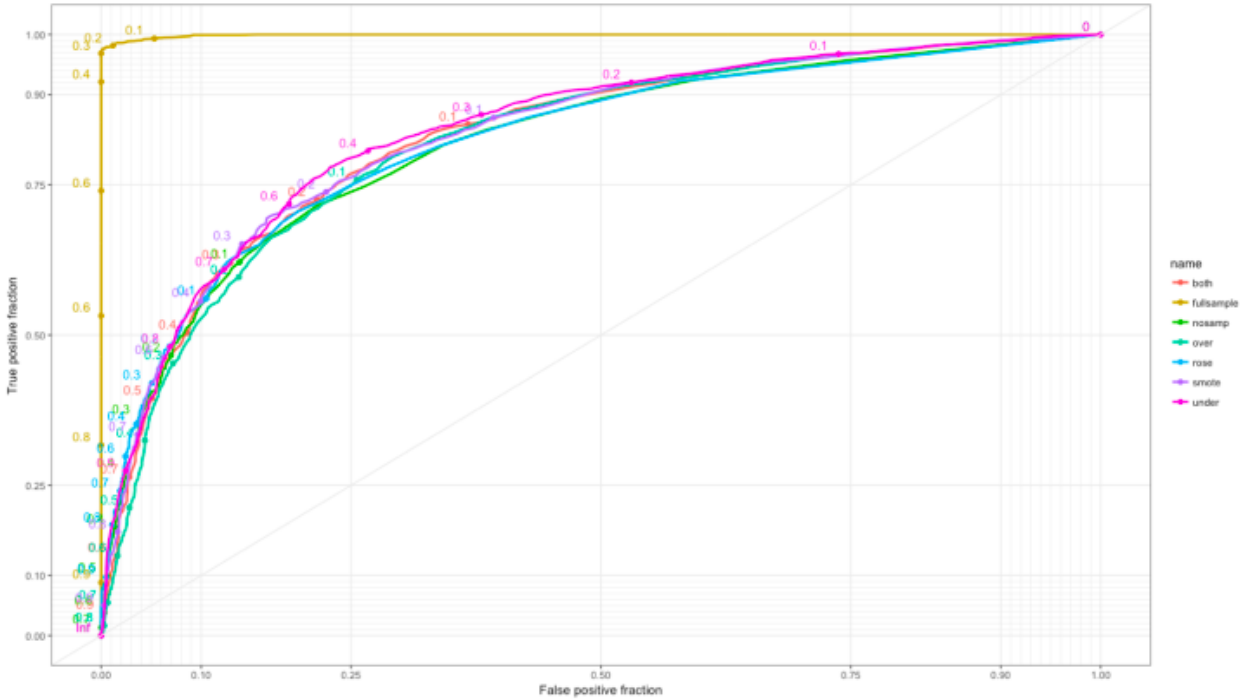
In the next section, we will examine the out of sample forecast results for the model. Since random forests calculate error on out-of-bag observations, technically we don’t need separate training and testing datasets. However, in order to be consistent with the research we have conducted with all other machine learning models, we make a distinction between training and testing data set. The out of sample forecasting result is shown on the table below.

Accuracy Rates

	F1 score	AUC

Training_data	0.736	0.813
both_sample_train_data	0.762	0.847
over_sample_train_data	0.710	0.839
rose_sample_train_data	0.712	0.827
smote_sample_train_data	0.773	0.849

The Synthetic Oversampling technique (SMOTE) has the highest F1 score and Area Under Curve.



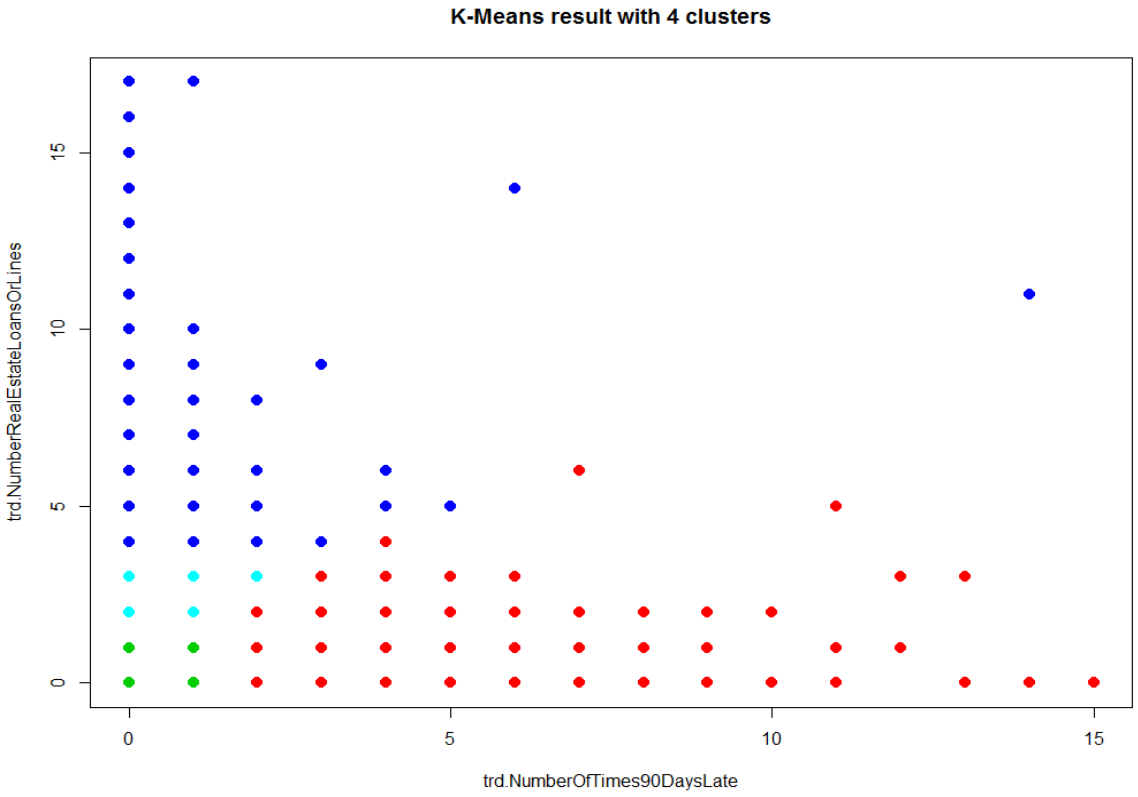
This is a graphical representation of the table above. Due to the bootstrapping nature of random forests, additional sampling technique on the training data does not significantly alter the results from random forest. The yellow line shows the sensitivity(true positive fraction) and specificity(false positive fraction) trade off for random forests. From the yellow line we can see that the random forest model fit perfectly into the training data. The in sample accuracy for predicting default is over 99%. Normally, a decision tree model with such a perfect fit would suffer from overfitting problem. In this case, the out of sample prediction accuracy is above 76%

regardless of the sampling technique we used for training data. Although it may seem redundant to bootstrap training data for random forests. The Synthetic Oversampling technique (SMOTE) out performs other sampling technique.

Market Segmentation

K mean clustering

K mean clustering is an unsupervised method to group data. Based on the decision tree's selection, we believe there are three variables that are most important for prediction, but we only choose two of them as K mean clustering's dimension is for the purpose of better interpretation. These two variables are the number of real estate loan lines, and the number of 90 days late payment but not worse. With the help of R, we determined the best number of group for our data is 4, and the clustered plot looks like below:

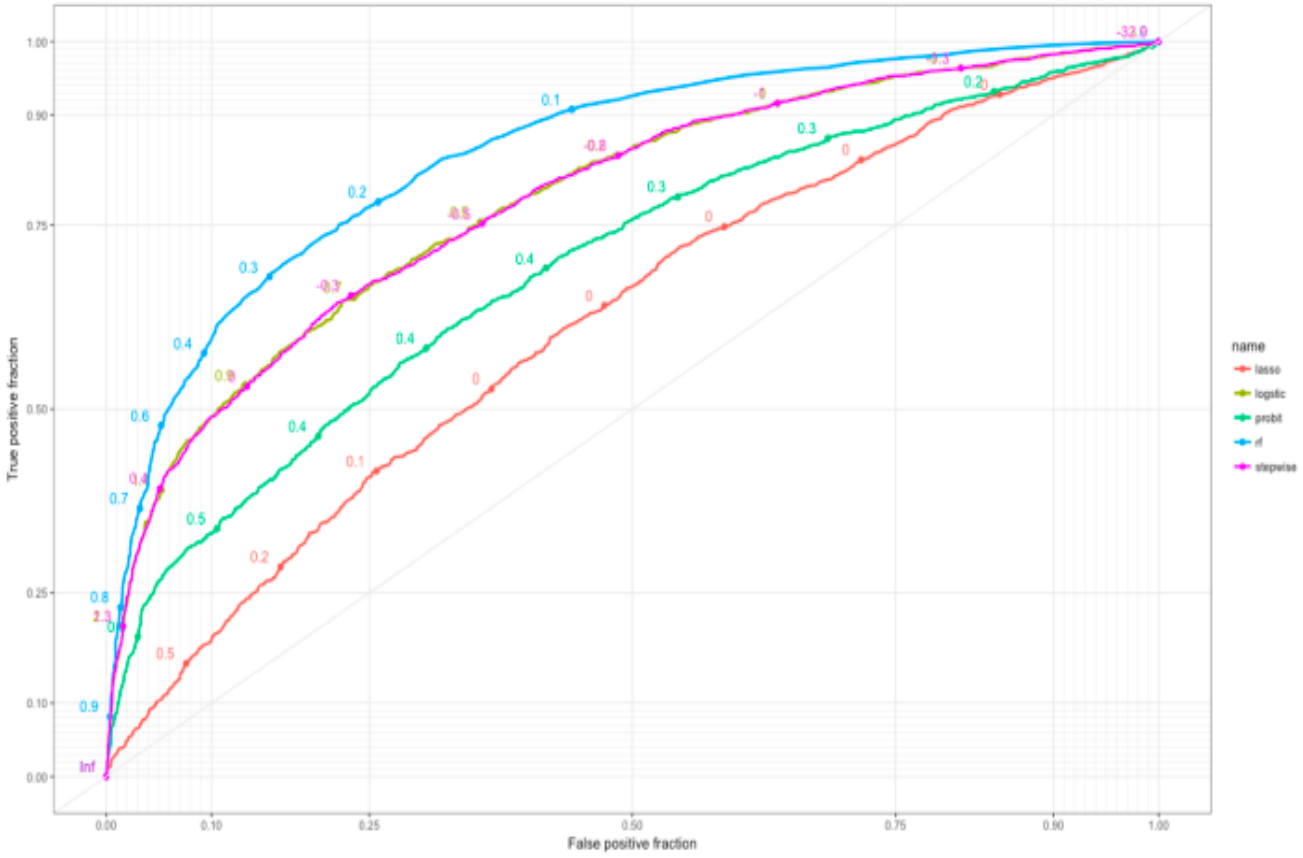


The red points area are customers with a lot of times of late payments but only have a few real estate loan lines. The blue area are customers with few late payment records, but with a lot of real estate loan lines. The green points and light blow points are customers with low late payment times and low number of real estate loans. Intuitively it is hard to compare which one is more risker between the blue group and red group. Thus we subset all the data points in different groups, and combine them with their defult status into a list, and calculated the defult rate for all the four groups. The results are shown below:

red	blue	green	Light blue
0.4806	0.0909	0.0408	0.044

It turns out that the red group is significantly risker than any other groups with a defult rate of 48%. As for the application of this result, when a new customer's data come in, including his or her number of 90 days late payment, and number of real estate loans, we can put it into the model we have and check which group he or she belongs to, if the new customers lies in the red group, we will have a good reason to reject offering loans to this customer.

Conclusion



This is a graphical representation of accuracy rates for different model. We depict sensitivity(true positive fraction) against specificity(false positive fraction). The greater the area under the curve the better the model. Models from the best to the worst are random forest, stepwise regression, logistic regression, probit and lasso.

We developed several machine learning models to preemptively detect the credit fraud, The best performing model is random forests. It tells us that the most important factor in predicting credit fraud is the history of default and the excess credit line. Both random forest and clustering algorithm suggest that people with more than 3 past overdues and small credit balance have the highest risk of default.

In order to avoid biased estimating from the unbalanced data. We implemented 6 different sampling technique. Among all bootstrapping method. The over-sampling method is rarey the best. However, it consistently performs moderately well when testing against different machine learning models. Due to its simplicity, the over-sampling method is a good start for resampling any unbalanced data. The under-sampling method is the most dangerous method. While the problem may not be pronounced for big training data. In our project, the under-sampling method removed over 97% of the data, resulting in a biased estimate for lasso and probit regressions. Overall, the synthetic oversampling method performs best in most of our models, we generally recommend using smote when dealing with unbalanced data.

Future Work

There are 3 ways that we think that we could have improved on the study. First being introduction of black box techniques to create a better model, toggle with the probability classification, or change the cost function.

Less Interpretable Models

All of the techniques we covered above have some sort of interpretability to them, however in machine learning there is a trade off between interpretability and model complexity. If we increased the complexity of the model using black box techniques such as: neural networks, deep learning, or xgboost we can get much better results but lose interpretability. It would be interesting to see how that would compare.

Furthermore, one can try an ensemble method combining different models to boost predicting power, but with that you would also lose interpretability. But it would be interesting to see how they would compare.

Changing Probability Threshold

Something that the team could play around with would be the probability classification. For classification, the threshold is at 50% but it would be interesting to see how our results would compare or what would be the optimal threshold to classify into a 1.

Changing Cost Function

Another thing for future work would be to adjust the cost function using Cost Sensitive Learning so that the Cost of False Negative would be greater than False Positive. We tried capturing this by optimizing using F1 score, Precision, and Recall, but it would be great if we can somehow add a penalizing cost based off the amount loaned or loss.