LENDING CLUB LOAN DATA ANALYSIS – GROUP 4

DATA MINING – STAT 642 – GROUP 4

MENGYUAN (MEGAN) LIN – PARIKA GUPTA - HANG (JESSIE) LE – VANDANA AGRAWAL – QINTIAN (AARON) QI

ABSTRACT

THE PURPOSE OF THIS ANALYSIS IS TO APPLY DIFFERENT DATA MINING METHODS TO FIND PATTERNS IN THE FINANCIAL DATA SET AND PREDICT RISK DEFAUL RATE.

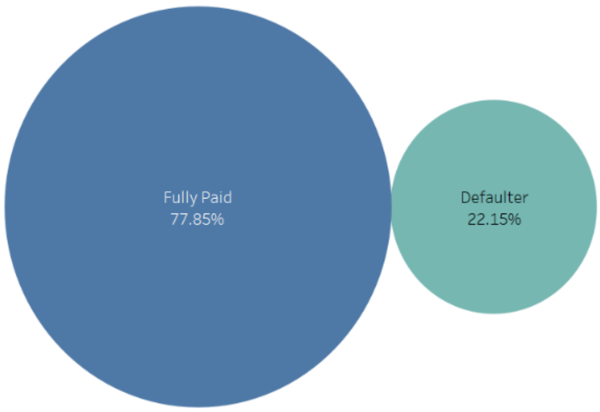
1. Introduction:

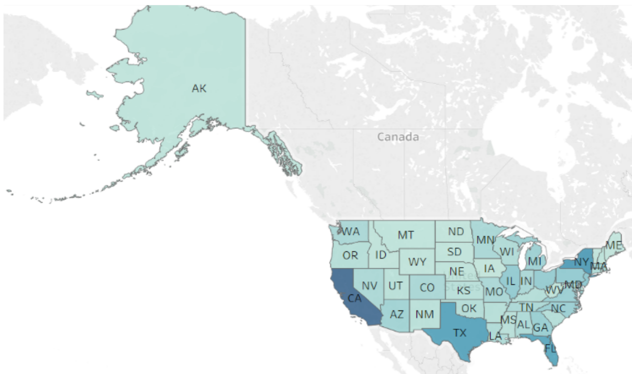
In this project, we want to apply several statistical analyses to the Lending Club loan dataset in order to answer different business questions. We will explore various relationships between loan amount and status with certain variables. From the marketing perspective, we use association rules analysis to identify certain customer groups to advertise different loan products to. And as part of the risk management prospects, we cluster customers into multiple groups with different risk profile so that the company could better understand its customers' behaviors in the future. And lastly, we compared different classification model, including linear discriminant and random forest, for a more accurate prediction.

2. Data Observation

The lending club loan data contains loan data issued through 2007 to 2015. It also contains the loan status for the customer i.e. Current, Charged Off, Late Payments, Fully Paid, Etc. The dataset has 145 variables and 2.26 million records. Other features that the dataset includes are credit scores, address including zip codes, states, interest rate, loan amount, annual income, etc. The dataset also describes what is the purpose of the loan, i.e. for debt consolidation, credit card, home improvement, vacation, education or wedding. There were also missing values in the dataset, which were deleted after the exploratory analysis to perform clustering, association and predictive modeling.

3. Exploratory Data Analysis



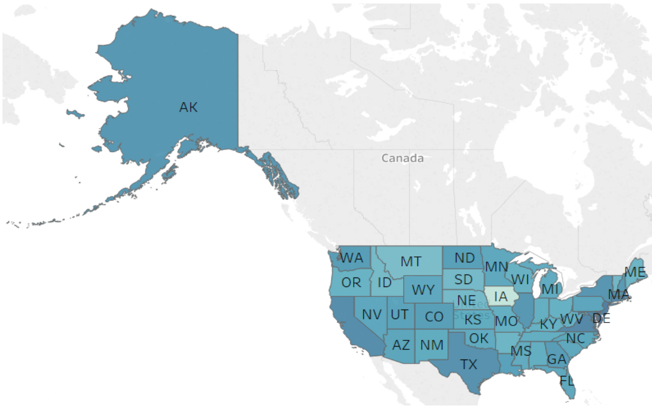


Graph 1: *Loan Status Comparison*

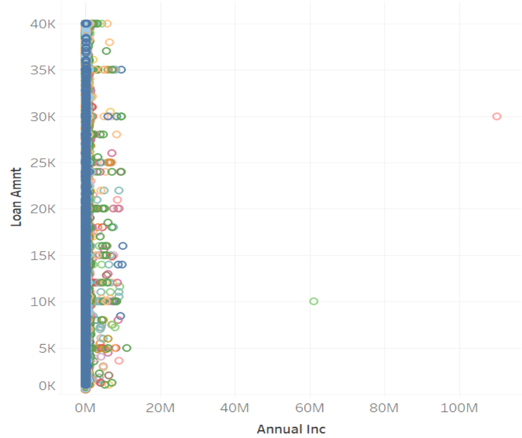
Graph 2*: Defaulters w.r.t States*

Graph 1 shows the comparison between the customers who are defaulters and who are not (who fully paid the loan). 77.85% customers fully paid the loan and 22.15% customers are defaulters.

Graph 2 is the demographic analysis showing number of the default customers in every state. We found out that California has the highest number of defaulters and Iowa had the lowest number of defaulters. After analyzing the insight, we found that out dataset had a greater number of records from California, thus more defaulters.



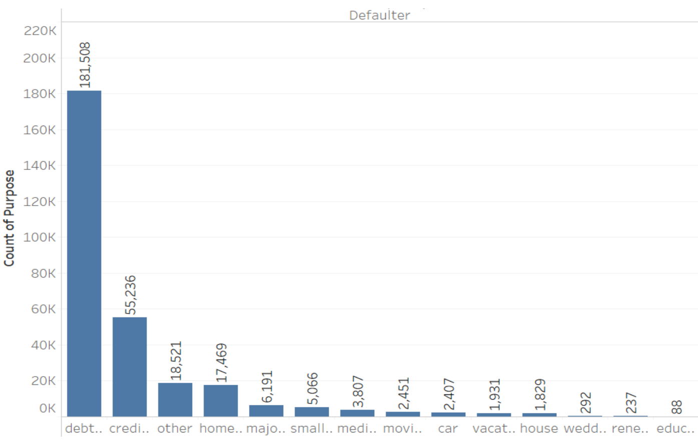
Graph 3: *Average Annual Income w.r.t State*



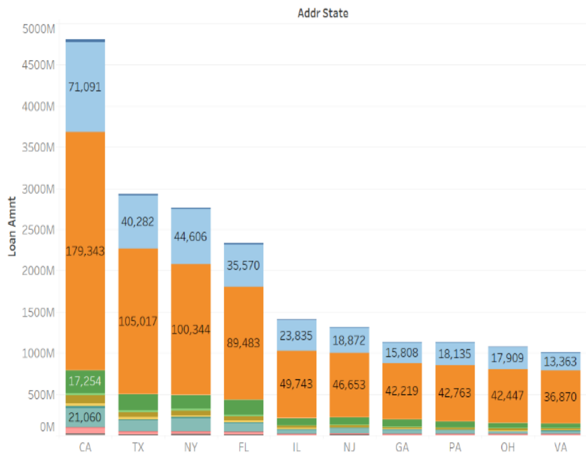
Graph 4*: Relationship between Loan Amount and Annual Income*

Graph 3 is the second demographic analysis which shows the average annual income per state. According to our dataset, DC, New Jersey, Connecticut and Maryland are some of the states which have the highest annual income.

Graph 4 shows that there is no discernible relationship between loan amount and annual income. But we found out some outliers in the graph such as, a customer whose annual income is 110 million and requests a loan amount of 30 thousand.



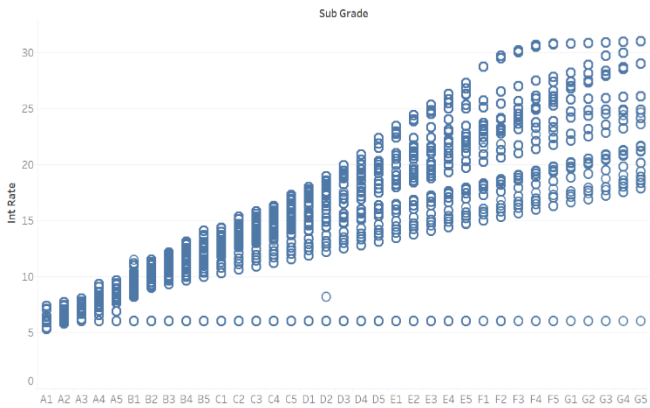
Graph 5: *Defaulters Purpose of Taking loan*



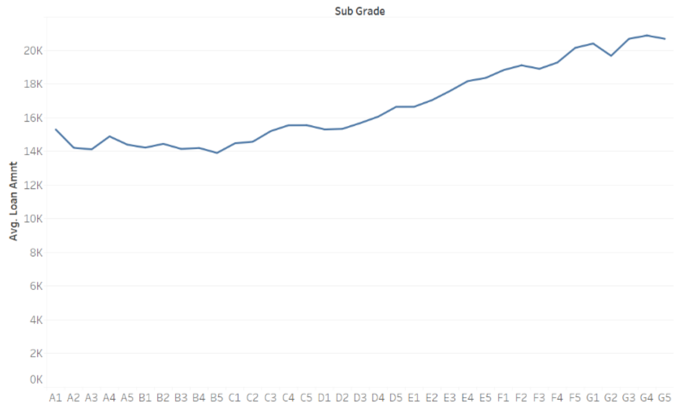
Graph 6: *Top 10 States having highest loan amount for all purposes*

Graph 5 shows that mostly the defaulters took the loan for debt consolidation and credit card purpose. More than 180 thousand defaulters took loan for debt consolidations and 55 thousand defaulters took loan for credit card purpose.

Graph 6: we can see that all the top 10 states have the highest loan amount for debt consolidation (orange) and second highest for credit card (Light Blue) purpose. Again, it is seen that California has the highest loan amount, which is because the dataset had a greater number of records for California.



Graph 7: *Relationship between Sub-grade and Interest Rate*



Graph 8: *Relationship between Sub-grade and Average loan amount*

Graph 7 shows that Sub grade is used to profile the customer. Low sub grade (A1) means less risky customers and high sub-grade (G5) implies very risk customer. As the sub grade moves from A1 to G5, the interest increases.

Graph 8: As we saw in previous graph that the sub-grade moves from A1 to G5, the interest rate increases. When we analyzed sub grade with average loan amount, we saw same trend. It is implied that there is a relationship between sub-grade, interest rate and loan amount.

4. Association Analysis

Association Rules Analysis

Association rules mining is a popular way to discover patterns in data. From this loan dataset, we would like to understand what kind of customer, in terms of their income and employment lengths, would go for what type of loan products. Therefore, from a marketing perspective, we would be able to advertise certain loan products to certain group of customers as the result.

Data Preparation

The entire dataset contains over 2 million observations and 145 variables. Out of the 145 variables, we selected 4 variables to run the association rules on: “ID”, “Employment length”, “Income”, and “Purpose”. To switch the numerical data type under “Income” to characters, we divided the income amount into 4 categories based on the U.S. 2018 tax bracket dividing lines. (The amount for each category were the tax income on Head of Household, as this filing status indicate the highest individual taxable income.) The categories division is shown in *Figure 1*.

Tax Rate	Income	Category
10~12%	<51,800	Low
22~24%	~157,500	Moderate
32~35%	~500,000	High
37%~	>500,000	Rich

Figure 1: *Categories division*

As the data is well prepared, we now have 11 different lengths of employment, 4 income categories, and 13 types of loan purpose.

Association Rules Mining in R Studio

Due to the size of the dataset but a handful of variables selections, we wanted to include as many rules on loan purpose as possible. We decided the support as 0.01 and a confidence level of 0.01 which generated 225 association rules for us (The complete rules result could be found in **Appendix 2**). Next we downsized to the subgroup where we first eliminated the rules that have lower than 0.2 confidence and set the lift larger than 1. A lift ratio larger than 1 implies that the relationship between the antecedent and the consequent is more significant than would be expected if the two sets were independent. As a result, 106 rules were left after the minimum lift was identified as 1.

Association Rules Results

The logic of our association rules mining is to discover certain pattern or combinations of patterns that would result in certain purpose of the loan. Therefore, we only keep the 28 rules that lead to a particular loan purpose at the end. The final association rules were summed in Figure 2. As we could see, customers who go for debt consolidation loan are the people who have more than 5 years employment with low to moderate income. Customers in this group have been working for a while and have a stable income, they seek for loan to save money on interest, lower their monthly payments, and pay down previous debts faster.

For the people who choose to loan for their credit card refinancing, they have less than 5 years' employment and the income has ranged from low to high. This group of customers has just started working or hasn't settled for the job in a long run. It's easy for them to go over the credit line on one card and keep turning to another. As a result, they have to transfer the balance of several credit cards to another credit card and loan to save interest on monthly payments.

Another rule we have are for the people who applied loan for their home improvement. In this customer group, they have been working over 10 years. People, at this stage of life, owned their home or about to get out from the home mortgage are now seeking for loan for their home improvement projects.

Employment Length	Income Category	Purpose	Support	Confidence	Lift	Count
9		Debt Consolidation	0.02	0.581	1.03	46145
8		Debt Consolidation	0.023	0.576	1.02	52974
7		Debt Consolidation	0.024	0.574	1.02	53213
6		Debt Consolidation	0.026	0.567	1	58169
10+		Debt Consolidation	0.192	0.579	1.02	432996
	Moderate	Debt Consolidation	0.355	0.571	1.01	801686
9	Moderate	Debt Consolidation	0.014	0.585	1.03	30586
8	Moderate	Debt Consolidation	0.015	0.582	1.03	34600
7	Moderate	Debt Consolidation	0.015	0.578	1.02	34045
6	Moderate	Debt Consolidation	0.016	0.57	1	36571
5	Low	Debt Consolidation	0.012	0.569	1	26497
10+	Low	Debt Consolidation	0.042	0.578	1.02	94169
10+	Moderate	Debt Consolidation	0.14	0.58	1.03	314353
	High	Credit Card	0.013	0.236	1.03	28291
4		Credit Card	0.014	0.233	1.02	31862
5		Credit Card	0.014	0.23	1	32120
1		Credit Card	0.016	0.246	1.08	36493
3		Credit Card	0.019	0.239	1.05	43237
<1		Credit Card	0.021	0.251	1.096	47626
2		Credit Card	0.022	0.242	1.059	49310
	Low	Credit Card	0.075	0.23	1	169018
3	Moderate	Credit Card	0.01	0.24	1.04	25506
<1	Moderate	Credit Card	0.01	0.26	1.12	26497
2	Moderate	Credit Card	0.013	0.244	1.07	28698
10+		Home Improvement	0.026	0.078	1.18	58508
	Moderate	Home Improvement	0.045	0.072	1.08	101380
10+	Moderate	Home Improvement	0.019	0.08	1.21	43117
	Low	other	0.024	0.075	1.22	54997

Table 1: *Summary of association rules*

5. Clustering analysis:

Clustering purpose

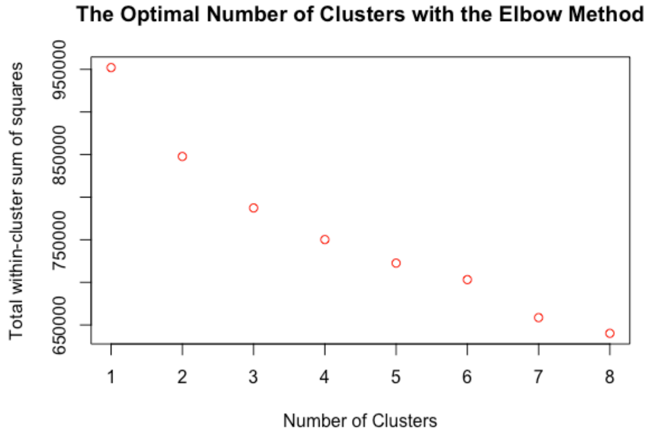
Clustering analysis is one of the most popular techniques used in financial data set. There are two main purposes of performing clustering analysis for this loan data set. The first purpose is to categorize customers into different groups to deliver customized marketing strategy in the future. The second purpose of clustering is to identify and develop profile of high-risk borrowers and analyze their behaviors.

Data Preparation

For better visualization and deeper understanding about each cluster, we randomly subset our data into smaller size: 100,000 observations and 23 attributes. There are 15 numeric attributes and 8 categorical attributes in our data set. Categorical attributes include personal information such as home ownership and customer's grade and loan data information such as application type, verify status, lending purposes, loan status, hardship and debt settlement. Numeric attributes include information such as annual income, employment length, number of open accounts, loan amount, term, interest rate, DTI, number of public records and average balance in current bank account. Then we performed data cleaning process to delete null values and high-correlated variables. The next step is transforming categorical variables into numeric data because K-means clustering can only deal with numeric data. For ordinal variables, different scales are used to present different levels of importance while for nominal variables, we use 0 and 1 to present yes/no status. Because data was captured in different scale, the final step is standardizing data to avoid miscalculating distance between data points. After cleaning, our data has 48895 observations and 23 attributes. Summary of data after cleaning and clustering visualization through PCA can be found at **Appendix 1**.

Clustering analysis in R

The method we will use to perform clustering analysis is K-means. This technique requires our understanding about data set to choose how many cluster centers to start with. We use Elbow method to choose number of cluster centers which can both minimize total distance between data points in the same cluster and spare distances among clusters.



Graph 9: *Number of cluster using Elbow Method*

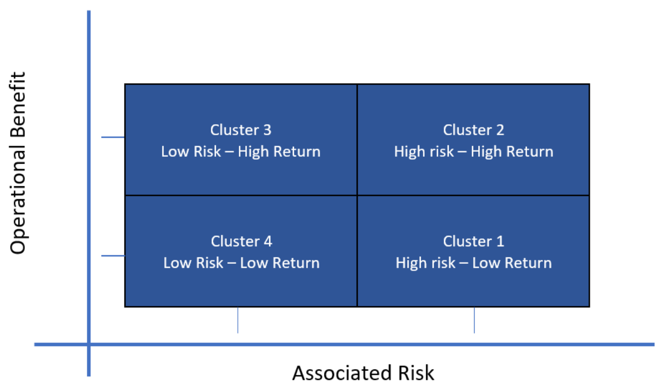
```
List of 9
 $ cluster     : int [1:48895] 4 3 2 3 4 3 4 4 3 4 ...
 $ centers     : num [1:4, 1:32] 0.581 0.678 0.402 -0.665 0.113 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4] "1" "2" "3" "4"
  .. ..$ : chr [1:32] "loan_amnt" "term" "int_rate" "installment" ...
 $ totss       : num 952056
 $ withinss    : num [1:4] 126708 165121 192945 265444
 $ tot.withinss: num 750218
 $ betweenss   : num 201839
 $ size        : int [1:4] 3840 11410 11612 22033
 $ iter        : int 5
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

Figure 2: *Summary of cluster analysis*

Graph 9 shows us that with 4 cluster centers, the total within-cluster sum of square starts decreasing slowly. Therefore, we choose to start with 4 cluster centers in our analysis.

Figure 2 shows summary of our cluster analysis using K-means algorithm. The total within-cluster sum of square is 750,218 with only 4 cluster centers used. The size of each cluster is 7.8%, 23.3%, 23.8% and 45% respectively.

Cluster Interpretation



Graph 10: *Cluster Interpretation*

By looking at cluster centers, we can see differences among group of customer assigned to different clusters. Graph 10 shows us the overall segmentation of customer based on clustering results in term of operational benefits and associated risk belong to that group of customers. There are two important groups of customers that are important for the company to look at:

- Cluster 2: This cluster includes customer who pose the highest risk level. In term of risk management, Lending club should take a closer look to understand their behaviors. Some characteristic from this group of customer: they have the highest loan amount, the shortest employment lengths, the lowest income and they are having public record of dealing with debt settlement companies to settle their previous debt. They also usually apply for hardship loan which is usually used in case borrowers are in pressure of money. However, customers in this group are also among the most profitable customers for the company because of high interest rate that they are paying.
- Cluster 3: This cluster includes the least risky customer but still the most profitable customers. Although they have high loan amount, they also have highest annual income, longest employment length and low DTI ratio which can be used to guarantee their capability to pay back loan. Moreover, customers in this group contribute to the profit of the company by using different products from companies.

Cluster Validation

There are 3 different ways our group is recommending for clustering validation in this analysis.

- *Using different number of cluster centers*: in our analysis, we choose to go with 4 cluster centers. However, to validate the goodness of clusters, we tried to apply analysis with 5 cluster centers in K-means algorithm. With 5 cluster centers, the total within-cluster sum of square is going down to 723,812 (approximately decrease by 3.5%). This represents the closer among data points in each cluster and hence, the better clustering in term of statistics.
- *Using k-prototypes clustering*: This clustering algorithm is developed by Z.Huang (1998) as an extension to K-means Algorithm for clustering large data sets with categorical variables. The major difference between K-prototype compared with K-means is the cost functioned built under K-prototypes algorithm to measure the similarities between categorical and numeric attributes. K-prototype clustering can be performed by clustMixType package in R.
- *Using distance-based clustering*: the concept of Gower distance is: for each variable type, a particular distance metric that works well for that type is used and scaled to fall between 0 and 1. Then, a linear combination using user-specified weights (most simply an average) is calculated to create the final distance matrix. Usually, Manhattan distance is used for quantitative attributes while Dice coefficient is used to calculate similarities between nominal attributes. Compared to K-means algorithm, distance-based clustering can deal with noise and outlier much better. However, it also takes more time and memory for computer to run.

6. Predictive Modeling

- **Linear Regression Classifier**

First, we used Naïve Bayes to calculate the proportion of default and full paid given all circumstances in order to select the important variables. By comparing the difference between the probability of default and paid off, we could determine which feature may exert an important influence in our analysis. For example, among all the people who are paid off ('0'), the proportion of individual leader is 99.7%. It seems reasonable that individual leader is an important factor because the priori probability of paid off is only 87%. However, among all the default people, the proportion of being an individual lender is 99.4%, which is almost the same with all the default people. So, we could get rid of the variable 'individual' in our analysis. We could also ignore the feature 'CA' because no matter people are default or paid off, the probability of living in CA under the two circumstances are both around 85%, which indicates no difference.

```
   Individual
y           0           1
  0 0.002398864 0.997601136
  1 0.005578409 0.994421591

   CA
y           0          1
  0 0.8425578 0.1574422
  1 0.8530603 0.1469397
```

```
   Not.Verified
y         0        1
  0 0.669750 0.330250
  1 0.766891 0.233109
```

Take the variable 'Not. Verified' as another example, the proportion of not verified is 33% and 23% respectively when people are paid off and default, which indicates a big difference. So, we can involve the feature 'Not Verified' in our analysis.

Then we use logistic regression to filter all the insignificant variables because there is maybe correlation between these important variables. We selected variables whose p-value is less than 0.05. In the end, we choose 40 variables out of 137 variables and these 40 variables will be put into our analysis.

Finally, it is time to make classification by using prediction models. We choose discriminant analysis as our first model. It is the result of 10-fold cross validation. As we can see, the standard deviation of our model is pretty small because the accuracies of the 10 folds are all around 91%. So, the model is very stable and we can trust the result of our model when making prediction in another dataset.

```
> model$resample
    Accuracy     Kappa Resample
1  0.9093997 0.6068433   Fold01
2  0.9103068 0.6119990   Fold02
3  0.9135442 0.6275262   Fold03
4  0.9071175 0.5945381   Fold04
5  0.9084395 0.6000070   Fold05
6  0.9100849 0.6084526   Fold06
7  0.9074890 0.5963450   Fold07
8  0.9092453 0.6065734   Fold08
9  0.9110498 0.6145129   Fold09
10 0.9109920 0.6157370   Fold10
```

```
          Reference
Prediction      0      1
         0 155288  15965
         1   1036  16123

 Accuracy (average) : 0.9098
```

As we can see in the confusion matrix, the precision of our model is very high (over 99%) while the recall is pretty a little low (only 50.2%). We not only need a good precision and also need a good recall. So we also use other analytics models to make prediction.

- **<u>Random Forest Classifier</u>**

The classification result of Linear discriminate analysis is excellent in terms of accuracy and precision for identifying which customer going to default on their loan. We observed that lot of the

predictor variables does not have a normal distribution. Linear discriminate analysis requires variable to have normal distribution and not good for few categories of variable. It computes the addition of Multivariate distribution compute CI and it suffers multicollinearity.
In order to build more robust model, we choose Random forest to compare the result from Linear discriminate analysis. Random forest is supervised learning algorithm. It uses ensemble of decision tree and trained using bagging method. Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Instead of looking for the most important feature while splitting a node, it searches for the best feature among a random subset of features. It means random forest replaces the data/population used to construct the tree and the explanatory variables are bootstrapped so that partition is not done on the same important variable. This results in a wide diversity that generally results in a better model. For our analysis we have used Gini index to calculate the impurity.

Out of 196 features in the original data set with Random forest we can identify 40 most important feature which are the most significant when it comes to predicting the loan status of a customer. Please see the **Appendix 3** to see list of the features. The most important feature comes out to be Recoveries: post charge off gross recovery, meaning that amount of charge off person has paid as recovery amount. On further analysis on this component we found that even when a person pays charge off or it settled. FICO score is negatively affected by the fact that person has be charged off in past. Other important features are Remaining outstanding principal for total amount funded, flag indicator whether or not the borrower who has charge off is working with a debt settlement company, interest rate, DTI which is ratio of total monthly debt payment on total debt obligation over total monthly income of borrower (self-reported)

Criteria	Imbalanced Data	Balanced Data
10-fold Cross Validation score	0.94 0.94 0.93 0.93 0.94 0.93 0.94 0.94 0.94 0.94	0.86 0.87 0.86 0.86 0.87 0.86 0.87 0.87 0.86 0.86
Accuracy	0.94	0.86
Precision	0.99	0.95
Recall	0.72	0.77
F-Score	0.83	0.85

Table 2: *Comparison between Random Forest Model performed on Balanced Data and Imbalanced Data*

Comparison Between LDA and Random Forest

By comparing the classification matrix results of LDA and random forest we can see that Random forest is much better model in terms of Recall rate and F-score. For our analysis it was important to correctly predict if a customer defaults or not in order to build the customer risk profile. Also Recall rate is important from the business perspective for example if company wanted to target few high-risk customers, they should be sure and absolute correct when it comes to targeting.

	LDA	Random Forest
Accuracy	.90	.94
Precision	.99	.84
Recall	.54	.72
F-Score	.65	.76

Table 3: *Comparison between Random Forest Classifier and Linear Regression Classifier*

6. Result and discussion:

Conclusion of above result is that if we combine LDA and Random forest for prediction. From RF Model we can take out Recall and from LDA we can take out precision. By joining both models, we can better predict the probability of borrower will default or not and based on that we can build customer risk profile or create various risk management strategies and hence.
The recall of the random forest is much larger than discriminant analysis while the precision of discriminant analysis is much better than random forest. So, we decided to combine the two models together to make prediction.
First, we will ignore all the prediction result of "1" in discriminant analysis and only trust the prediction of "0". Likewise, we will ignore all the prediction of "0" in random forest and only trust the prediction of "1".
Second, we need to deal with the duplicated area. That means when the random forest model make prediction of "1" and at the same time the discriminant analysis model makes a prediction of "0", we have to determine which model should be trust given this certain circumstance.
Not only we should take consideration of recall, but also we need to guarantee a good precision. The accuracy of "1" in the discriminant model is 90.3% while the accuracy of "0" in the random forest model is 86.2%. If we want to get a high recall, we can predict all these cases as "0" which could give us a recall around 75%. If we want to get a good accuracy, we can predict all the cases as "1".

In conclusion, it all depends on the needs of the company.

7. Individual Contribution:

Individual Name	Individual Contribution
Parika Gupta	Data collection and Data cleaning for Exploratory Data Analysis, Data visualization, Paper Writing (Part 2 & 3), Presentation editing.
Mengyuan (Megan) Lin	Data Cleaning for Association Analysis, Modeling and Programming (Association), Paper Writing (Part 1 & 4)
Hang (Jessie) Le	Data Cleaning for Clustering analysis, Modeling and Programming (Clustering Analysis), Paper writing (Part 5), Report editing.
Vandana Agrawal	Data collection and Data cleaning (overall), Data transforming, Modeling (Random Forest Analysis), Model Comparison, Paper Writing (RF analysis)
Qintian (Aaron) Qi	Data Cleaning and Data Transforming, Modeling (LDA), Model Combination, Paper Writing (LDA & Part 6)

References:

1. https://www.investopedia.com/terms/c/chargeoff.asp
2. https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/

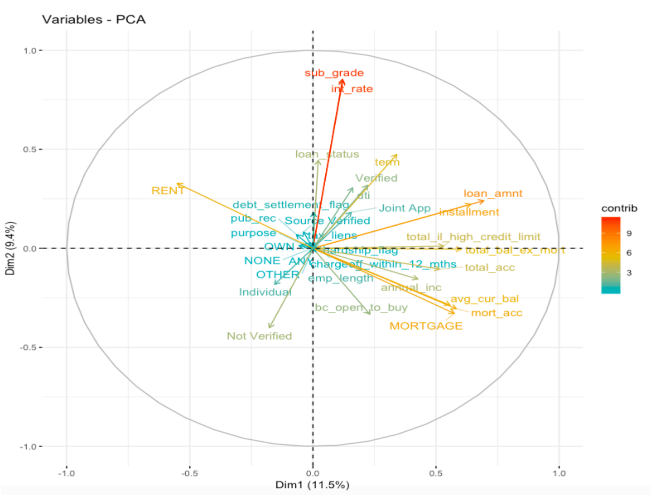
3. Z.Huang (1998): Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Variables, Data Mining and Knowledge Discovery 2, 283-304

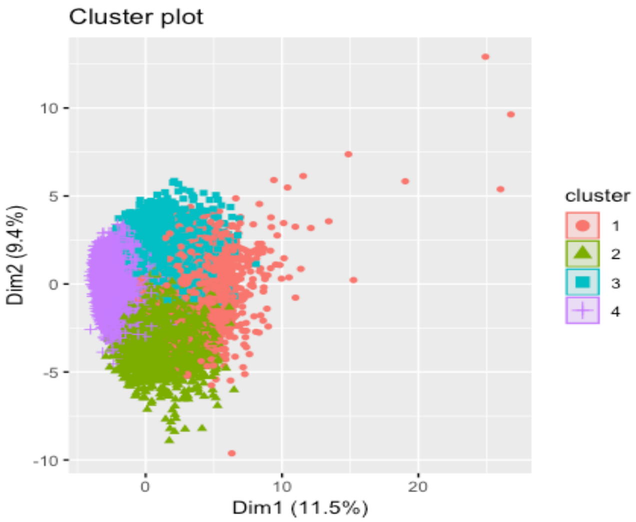
APPENDIX 1 – CLUSTERING DATA AND CLUSTERING VISUALIZATION THROUGH PCA

Clustering – Data Transforming:

```
Classes 'tbl_df', 'tbl' and 'data.frame':	48895 obs. of  32 variables:
 $ loan_amnt                 : num  4175 20000 13975 20000 2500 ...
 $ term                      : num  36 36 60 36 36 36 36 36 60 36 ...
 $ int_rate                  : num  13.35 8.18 25.49 10.64 14.09 ...
 $ installment               : num  141.4 628.4 414.2 651.4 85.6 ...
 $ sub_grade                 : num  12 6 24 7 10 5 7 20 7 9 ...
 $ emp_length                : num  1 1 4 10 10 10 7 10 1 7 ...
 $ annual_inc                : num  10000 110000 48440 140000 40000 ...
 $ loan_status               : num  1 0 0 0 0 0 0 0 0 0 ...
 $ purpose                   : num  1 2 2 1 7 2 2 5 1 2 ...
 $ dti                       : num  22.1 22.7 28.1 14 22.4 ...
 $ pub_rec                   : num  1 0 1 0 0 0 0 0 0 2 ...
 $ total_acc                 : num  16 45 24 22 36 22 10 43 12 14 ...
 $ avg_cur_bal               : num  2203 2727 16064 36213 10650 ...
 $ bc_open_to_buy            : num  1737 223 5488 8077 1550 ...
 $ chargeoff_within_12_mths  : num  0 0 0 0 0 0 0 0 0 0 ...
 $ mort_acc                  : num  0 2 1 4 1 1 0 0 1 0 ...
 $ tax_liens                 : num  0 0 0 0 0 0 0 0 0 2 ...
 $ total_bal_ex_mort         : num  17625 46361 37255 53855 32446 ...
 $ total_il_high_credit_limit: num  10021 10000 32467 28122 41577 ...
 $ hardship_flag             : num  0 0 0 0 0 0 0 0 0 0 ...
 $ debt_settlement_flag      : num  0 0 0 0 0 0 0 0 0 0 ...
 $ Individual                : num  1 1 1 1 1 1 1 1 0 1 ...
 $ Joint App                 : num  0 0 0 0 0 0 0 0 1 0 ...
 $ ANY                       : num  0 0 0 0 0 0 0 0 0 0 ...
 $ MORTGAGE                  : num  0 1 1 1 1 1 0 0 1 0 ...
 $ NONE                      : num  0 0 0 0 0 0 0 0 0 0 ...
 $ OTHER                     : num  0 0 0 0 0 0 0 0 0 0 ...
 $ OWN                       : num  0 0 0 0 0 0 0 0 0 0 ...
 $ RENT                      : num  1 0 0 0 0 0 1 1 0 1 ...
 $ Not Verified              : num  1 1 0 0 1 0 1 0 0 0 ...
 $ Source Verified           : num  0 0 1 0 0 0 0 0 1 0 ...
 $ Verified                  : num  0 0 0 1 0 1 0 1 0 1 ...
```

Clustering visualization using PCA:





APPENDIX 2 – ASSOCIATION RULES

```
> inspect (rulesW2)
    lhs                        rhs                        support confidence      lift   count
[1]   {}                    => {purpose=car}             0.01062208 0.01062208 1.0000000   24013
[2]   {}                    => {purpose=small_business}     0.01092111 0.01092111 1.0000000   24689
[3]   {}                    => {purpose=medical}          0.01215924 0.01215924 1.0000000   27488
[4]   {}                    => {purpose=major_purchase}     0.02231420 0.02231420 1.0000000   50445
[5]   {}                    => {emp_length=9 years}       0.03512015 0.03512015 1.0000000   79395
[6]   {}                    => {emp_length=8 years}       0.04065789 0.04065789 1.0000000   91914
[7]   {}                    => {emp_length=7 years}       0.04100337 0.04100337 1.0000000   92695
[8]   {}                    => {emp_length=6 years}       0.04539720 0.04539720 1.0000000  102628
[9]   {}                    => {inc_category=High}        0.05310200 0.05310200 1.0000000  120046
[10]  {}                    => {emp_length=4 years}       0.06042683 0.06042683 1.0000000  136605
[11]  {}                    => {purpose=other}           0.06168088 0.06168088 1.0000000  139440
[12]  {}                    => {emp_length=5 years}       0.06179501 0.06179501 1.0000000  139698
[13]  {}                    => {emp_length=n/a}          0.06498389 0.06498389 1.0000000  146907
[14]  {}                    => {emp_length=1 year}        0.06564564 0.06564564 1.0000000  148403
[15]  {}                    => {purpose=home_improvement}   0.06655422 0.06655422 1.0000000
150457
[16]  {}                    => {emp_length=3 years}       0.07995557 0.07995557 1.0000000  180753
[17]  {}                    => {emp_length=< 1 year}       0.08404065 0.08404065 1.0000000  189988
[18]  {}                    => {emp_length=2 years}       0.09009594 0.09009594 1.0000000  203677
[19]  {}                    => {purpose=credit_card}       0.22868064 0.22868064 1.0000000  516971
[20]  {}                    => {inc_category=Low}         0.32428955 0.32428955 1.0000000  733111
[21]  {}                    => {emp_length=10+ years}      0.33087786 0.33087786 1.0000000  748005
[22]  {}                    => {purpose=debt_consolidation} 0.56526522 0.56526522 1.0000000 1277877
[23]  {}                    => {inc_category=Moderate}     0.62108058 0.62108058 1.0000000 1404057
[24]  {purpose=major_purchase}     => {inc_category=Moderate}      0.01362783 0.61072455
0.9833258   30808
[25]  {inc_category=Moderate}     => {purpose=major_purchase}     0.01362783 0.02194213
0.9833258   30808
[26]  {emp_length=9 years}       => {inc_category=Low}         0.01004172 0.28592481 0.8816960
22701
[27]  {inc_category=Low}         => {emp_length=9 years}       0.01004172 0.03096530 0.8816960
22701
[28]  {emp_length=9 years}       => {purpose=debt_consolidation} 0.02041211 0.58120788
1.0282039   46145
[29]  {purpose=debt_consolidation} => {emp_length=9 years}       0.02041211 0.03611067
1.0282039   46145
[30]  {emp_length=9 years}       => {inc_category=Moderate}     0.02313254 0.65866868 1.0605205
52295
[31]  {inc_category=Moderate}     => {emp_length=9 years}       0.02313254 0.03724564 1.0605205
52295
[32]  {emp_length=8 years}       => {inc_category=Low}         0.01206900 0.29684270 0.9153632
27284
[33]  {inc_category=Low}         => {emp_length=8 years}       0.01206900 0.03721674 0.9153632
27284
```

[34] {emp_length=8 years} => {purpose=debt_consolidation} 0.02343290 0.57634310 1.0195977 52974
[35] {purpose=debt_consolidation} => {emp_length=8 years} 0.02343290 0.04145469 1.0195977 52974
[36] {emp_length=8 years} => {inc_category=Moderate} 0.02631523 0.64723546 1.0421119 59490
[37] {inc_category=Moderate} => {emp_length=8 years} 0.02631523 0.04237007 1.0421119 59490
[38] {emp_length=7 years} => {inc_category=Low} 0.01264759 0.30845245 0.9511637 28592
[39] {inc_category=Low} => {emp_length=7 years} 0.01264759 0.03900092 0.9511637 28592
[40] {emp_length=7 years} => {purpose=debt_consolidation} 0.02353862 0.57406548 1.0155684 53213
[41] {purpose=debt_consolidation} => {emp_length=7 years} 0.02353862 0.04164172 1.0155684 53213
[42] {emp_length=7 years} => {inc_category=Moderate} 0.02603434 0.63493177 1.0223017 58855
[43] {inc_category=Moderate} => {emp_length=7 years} 0.02603434 0.04191781 1.0223017 58855
[44] {emp_length=6 years} => {purpose=credit_card} 0.01031775 0.22727716 0.9938627 23325
[45] {purpose=credit_card} => {emp_length=6 years} 0.01031775 0.04511858 0.9938627 23325
[46] {emp_length=6 years} => {inc_category=Low} 0.01442361 0.31772031 0.9797427 32607
[47] {inc_category=Low} => {emp_length=6 years} 0.01442361 0.04447758 0.9797427 32607
[48] {emp_length=6 years} => {purpose=debt_consolidation} 0.02573089 0.56679464 1.0027057 58169
[49] {purpose=debt_consolidation} => {emp_length=6 years} 0.02573089 0.04552003 1.0027057 58169
[50] {emp_length=6 years} => {inc_category=Moderate} 0.02839426 0.62546284 1.0070559 64190
[51] {inc_category=Moderate} => {emp_length=6 years} 0.02839426 0.04571752 1.0070559 64190
[52] {inc_category=High} => {purpose=credit_card} 0.01251444 0.23566799 1.0305551 28291
[53] {purpose=credit_card} => {inc_category=High} 0.01251444 0.05472454 1.0305551 28291
[54] {inc_category=High} => {emp_length=10+ years} 0.02035903 0.38339470 1.1587197 46025
[55] {emp_length=10+ years} => {inc_category=High} 0.02035903 0.06153034 1.1587197 46025
[56] {inc_category=High} => {purpose=debt_consolidation} 0.02716719 0.51160389 0.9050687 61416
[57] {purpose=debt_consolidation} => {inc_category=High} 0.02716719 0.04806096 0.9050687 61416

[58] {emp_length=4 years} => {purpose=credit_card} 0.01409406 0.23324183 1.0199457 31862
[59] {purpose=credit_card} => {emp_length=4 years} 0.01409406 0.06163208 1.0199457 31862
[60] {emp_length=4 years} => {inc_category=Low} 0.02079385 0.34411625 1.0611389 47008
[61] {inc_category=Low} => {emp_length=4 years} 0.02079385 0.06412126 1.0611389 47008
[62] {emp_length=4 years} => {purpose=debt_consolidation} 0.03371039 0.55787123 0.9869194 76208
[63] {purpose=debt_consolidation} => {emp_length=4 years} 0.03371039 0.05963641 0.9869194 76208
[64] {emp_length=4 years} => {inc_category=Moderate} 0.03627689 0.60034406 0.9666122 82010
[65] {inc_category=Moderate} => {emp_length=4 years} 0.03627689 0.05840931 0.9666122 82010
[66] {purpose=other} => {inc_category=Low} 0.02432777 0.39441337 1.2162383 54997
[67] {inc_category=Low} => {purpose=other} 0.02432777 0.07501865 1.2162383 54997
[68] {purpose=other} => {emp_length=10+ years} 0.02007504 0.32546615 0.9836444 45383
[69] {emp_length=10+ years} => {purpose=other} 0.02007504 0.06067205 0.9836444 45383
[70] {purpose=other} => {inc_category=Moderate} 0.03442124 0.55805364 0.8985205 77815
[71] {inc_category=Moderate} => {purpose=other} 0.03442124 0.05542154 0.8985205 77815
[72] {emp_length=5 years} => {purpose=credit_card} 0.01420819 0.22992455 1.0054395 32120
[73] {purpose=credit_card} => {emp_length=5 years} 0.01420819 0.06213114 1.0054395 32120
[74] {emp_length=5 years} => {inc_category=Low} 0.02059834 0.33333333 1.0278880 46566
[75] {inc_category=Low} => {emp_length=5 years} 0.02059834 0.06351835 1.0278880 46566
[76] {emp_length=5 years} => {purpose=debt_consolidation} 0.03457297 0.55947830 0.9897625 78158
[77] {purpose=debt_consolidation} => {emp_length=5 years} 0.03457297 0.06116238 0.9897625 78158
[78] {emp_length=5 years} => {inc_category=Moderate} 0.03778175 0.61140460 0.9844207 85412
[79] {inc_category=Moderate} => {emp_length=5 years} 0.03778175 0.06083229 0.9844207 85412
[80] {emp_length=n/a} => {purpose=credit_card} 0.01458330 0.22441409 0.9813427 32968
[81] {purpose=credit_card} => {emp_length=n/a} 0.01458330 0.06377147 0.9813427 32968

[82] {emp_length=n/a} => {inc_category=Low} 0.04042168 0.62202618 1.9181197
91380
[83] {inc_category=Low} => {emp_length=n/a} 0.04042168 0.12464688 1.9181197
91380
[84] {emp_length=n/a} => {purpose=debt_consolidation} 0.03513784 0.54071624 0.9565709
79435
[85] {purpose=debt_consolidation} => {emp_length=n/a} 0.03513784 0.06216169 0.9565709
79435
[86] {emp_length=n/a} => {inc_category=Moderate} 0.02396991 0.36885921 0.5938991
54188
[87] {inc_category=Moderate} => {emp_length=n/a} 0.02396991 0.03859387 0.5938991
54188
[88] {emp_length=1 year} => {purpose=credit_card} 0.01614257 0.24590473 1.0753194
36493
[89] {purpose=credit_card} => {emp_length=1 year} 0.01614257 0.07059003 1.0753194
36493
[90] {emp_length=1 year} => {inc_category=Low} 0.02592597 0.39493811 1.2178564
58610
[91] {inc_category=Low} => {emp_length=1 year} 0.02592597 0.07994697 1.2178564
58610
[92] {emp_length=1 year} => {purpose=debt_consolidation} 0.03664315 0.55819626
0.9874944 82838
[93] {purpose=debt_consolidation} => {emp_length=1 year} 0.03664315 0.06482471
0.9874944 82838
[94] {emp_length=1 year} => {inc_category=Moderate} 0.03645737 0.55536613 0.8941934
82418
[95] {inc_category=Moderate} => {emp_length=1 year} 0.03645737 0.05869990 0.8941934
82418
[96] {purpose=home_improvement} => {inc_category=Low} 0.01533839 0.23046452
0.7106751 34675
[97] {inc_category=Low} => {purpose=home_improvement} 0.01533839 0.04729843
0.7106751 34675
[98] {purpose=home_improvement} => {emp_length=10+ years} 0.02588085 0.38886858
1.1752632 58508
[99] {emp_length=10+ years} => {purpose=home_improvement} 0.02588085 0.07821873
1.1752632 58508
[100] {purpose=home_improvement} => {inc_category=Moderate} 0.04484515 0.67381378
1.0849056 101380
[101] {inc_category=Moderate} => {purpose=home_improvement} 0.04484515 0.07220505
1.0849056 101380
[102] {emp_length=3 years} => {purpose=credit_card} 0.01912576 0.23920488 1.0460216
43237
[103] {purpose=credit_card} => {emp_length=3 years} 0.01912576 0.08363525 1.0460216
43237
[104] {emp_length=3 years} => {inc_category=Low} 0.02835843 0.35467738 1.0937059
64109
[105] {inc_category=Low} => {emp_length=3 years} 0.02835843 0.08744788 1.0937059
64109

[106] {emp_length=3 years}   => {purpose=debt_consolidation} 0.04447579 0.55625633
0.9840625 100545
[107] {purpose=debt_consolidation} => {emp_length=3 years}   0.04447579 0.07868128
0.9840625 100545
[108] {emp_length=3 years}   => {inc_category=Moderate}   0.04726214 0.59110499 0.9517364
106844
[109] {inc_category=Moderate}   => {emp_length=3 years}   0.04726214 0.07609663 0.9517364
106844
[110] {emp_length=< 1 year}   => {purpose=credit_card}   0.02106722 0.25067899 1.0961968
47626
[111] {purpose=credit_card}   => {emp_length=< 1 year}   0.02106722 0.09212509 1.0961968
47626
[112] {emp_length=< 1 year}   => {inc_category=Low}   0.03381478 0.40236225 1.2407500
76444
[113] {inc_category=Low}   => {emp_length=< 1 year}   0.03381478 0.10427343 1.2407500
76444
[114] {emp_length=< 1 year}   => {purpose=debt_consolidation} 0.04608859 0.54840832
0.9701788 104191
[115] {purpose=debt_consolidation} => {emp_length=< 1 year}   0.04608859 0.08153445
0.9701788 104191
[116] {emp_length=< 1 year}   => {inc_category=Moderate}   0.04564713 0.54315536 0.8745328
103193
[117] {inc_category=Moderate}   => {emp_length=< 1 year}   0.04564713 0.07349630 0.8745328
103193
[118] {emp_length=2 years}   => {purpose=credit_card}   0.02181214 0.24209901 1.0586773
49310
[119] {purpose=credit_card}   => {emp_length=2 years}   0.02181214 0.09538253 1.0586773
49310
[120] {emp_length=2 years}   => {inc_category=Low}   0.03313578 0.36778330 1.1341201
74909
[121] {inc_category=Low}   => {emp_length=2 years}   0.03313578 0.10217962 1.1341201
74909
[122] {emp_length=2 years}   => {purpose=debt_consolidation} 0.04998744 0.55482455
0.9815296 113005
[123] {purpose=debt_consolidation} => {emp_length=2 years}   0.04998744 0.08843183
0.9815296 113005
[124] {emp_length=2 years}   => {inc_category=Moderate}   0.05204479 0.57765973 0.9300882
117656
[125] {inc_category=Moderate}   => {emp_length=2 years}   0.05204479 0.08379717 0.9300882
117656
[126] {purpose=credit_card}   => {inc_category=Low}   0.07476463 0.32693904 1.0081701
169018
[127] {inc_category=Low}   => {purpose=credit_card}   0.07476463 0.23054899 1.0081701
169018
[128] {purpose=credit_card}   => {emp_length=10+ years}   0.07101264 0.31053193 0.9385092
160536
[129] {emp_length=10+ years}   => {purpose=credit_card}   0.07101264 0.21461889 0.9385092
160536

[130] {purpose=credit_card} => {inc_category=Moderate} 0.14110387 0.61703461 0.9934856
318989
[131] {inc_category=Moderate} => {purpose=credit_card} 0.14110387 0.22719092 0.9934856
318989
[132] {inc_category=Low} => {emp_length=10+ years} 0.07205879 0.22220510 0.6715623
162901
[133] {emp_length=10+ years} => {inc_category=Low} 0.07205879 0.21778063 0.6715623
162901
[134] {inc_category=Low} => {purpose=debt_consolidation} 0.18277828 0.56362679
0.9971015 413201
[135] {purpose=debt_consolidation} => {inc_category=Low} 0.18277828 0.32334959
0.9971015 413201
[136] {emp_length=10+ years} => {purpose=debt_consolidation} 0.19153454 0.57886779
1.0240640 432996
[137] {purpose=debt_consolidation} => {emp_length=10+ years} 0.19153454 0.33884012
1.0240640 432996
[138] {emp_length=10+ years} => {inc_category=Moderate} 0.23776424 0.71858611
1.1569934 537506
[139] {inc_category=Moderate} => {emp_length=10+ years} 0.23776424 0.38282349
1.1569934 537506
[140] {purpose=debt_consolidation} => {inc_category=Moderate} 0.35462350 0.62735772
1.0101068 801686
[141] {inc_category=Moderate} => {purpose=debt_consolidation} 0.35462350 0.57097824
1.0101068 801686
[142] {emp_length=9 years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.01352963 0.66282371
1.0672105 30586
[143] {inc_category=Moderate,
emp_length=9 years} => {purpose=debt_consolidation} 0.01352963 0.58487427 1.0346900
30586
[144] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=9 years} 0.01352963 0.03815209 1.0863306
30586
[145] {emp_length=8 years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.01530521 0.65315060
1.0516358 34600
[146] {inc_category=Moderate,
emp_length=8 years} => {purpose=debt_consolidation} 0.01530521 0.58161035 1.0289159
34600
[147] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=8 years} 0.01530521 0.04315904 1.0615169
34600
[148] {emp_length=7 years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.01505971 0.63978727
1.0301196 34045
[149] {inc_category=Moderate,
emp_length=7 years} => {purpose=debt_consolidation} 0.01505971 0.57845553 1.0233347
34045
[150] {inc_category=Moderate,

purpose=debt_consolidation} => {emp_length=7 years} 0.01505971 0.04246675 1.0356894
34045
[151] {emp_length=6 years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.01617708 0.62870257
1.0122721 36571
[152] {inc_category=Moderate,
emp_length=6 years} => {purpose=debt_consolidation} 0.01617708 0.56973049 1.0078994
36571
[153] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=6 years} 0.01617708 0.04561761 1.0048551
36571
[154] {inc_category=High,
emp_length=10+ years} => {purpose=debt_consolidation} 0.01050486 0.51598045
0.9128112 23748
[155] {inc_category=High,
purpose=debt_consolidation} => {emp_length=10+ years} 0.01050486 0.38667448
1.1686321 23748
[156] {emp_length=10+ years,
purpose=debt_consolidation} => {inc_category=High} 0.01050486 0.05484577 1.0328381
23748
[157] {inc_category=Low,
emp_length=4 years} => {purpose=debt_consolidation} 0.01163860 0.55971324 0.9901781
26311
[158] {emp_length=4 years,
purpose=debt_consolidation} => {inc_category=Low} 0.01163860 0.34525247 1.0646426
26311
[159] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=4 years} 0.01163860 0.06367603 1.0537709
26311
[160] {emp_length=4 years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.02036743 0.60418854
0.9728022 46044
[161] {inc_category=Moderate,
emp_length=4 years} => {purpose=debt_consolidation} 0.02036743 0.56144373 0.9932395
46044
[162] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=4 years} 0.02036743 0.05743396 0.9504711
46044
[163] {emp_length=10+ years,
purpose=other} => {inc_category=Moderate} 0.01355529 0.67523081 1.0871871
30644
[164] {inc_category=Moderate,
purpose=other} => {emp_length=10+ years} 0.01355529 0.39380582 1.1901849
30644
[165] {inc_category=Moderate,
emp_length=10+ years} => {purpose=other} 0.01355529 0.05701146 0.9242970
30644
[166] {inc_category=Low,

emp_length=5 years}   => {purpose=debt_consolidation} 0.01172087 0.56902032 1.0066431
26497
[167] {emp_length=5 years,
purpose=debt_consolidation} => {inc_category=Low}   0.01172087 0.33901840 1.0454188
26497
[168] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=5 years}   0.01172087 0.06412618 1.0377242
26497
[169] {emp_length=5 years,
purpose=debt_consolidation} => {inc_category=Moderate}   0.02115304 0.61183756
0.9851178   47820
[170] {inc_category=Moderate,
emp_length=5 years}   => {purpose=debt_consolidation} 0.02115304 0.55987449 0.9904634
47820
[171] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=5 years}   0.02115304 0.05964929 0.9652768
47820
[172] {inc_category=Low,
emp_length=n/a}   => {purpose=debt_consolidation} 0.02198156 0.54380608 0.9620370
49693
[173] {emp_length=n/a,
purpose=debt_consolidation} => {inc_category=Low}   0.02198156 0.62558066 1.9290806
49693
[174] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=n/a}   0.02198156 0.12026350 1.8506664
49693
[175] {emp_length=n/a,
purpose=debt_consolidation} => {inc_category=Moderate}   0.01287230 0.36633726
0.5898385   29100
[176] {inc_category=Moderate,
emp_length=n/a}   => {purpose=debt_consolidation} 0.01287230 0.53701927 0.9500306
29100
[177] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=n/a}   0.01287230 0.03629850 0.5585769
29100
[178] {inc_category=Low,
emp_length=1 year}   => {purpose=debt_consolidation} 0.01444617 0.55720867 0.9857473
32658
[179] {emp_length=1 year,
purpose=debt_consolidation} => {inc_category=Low}   0.01444617 0.39423936 1.2157017
32658
[180] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=1 year}   0.01444617 0.07903659 1.2039885
32658
[181] {emp_length=1 year,
purpose=debt_consolidation} => {inc_category=Moderate}   0.02047802 0.55884980
0.8998024   46294
[182] {inc_category=Moderate,

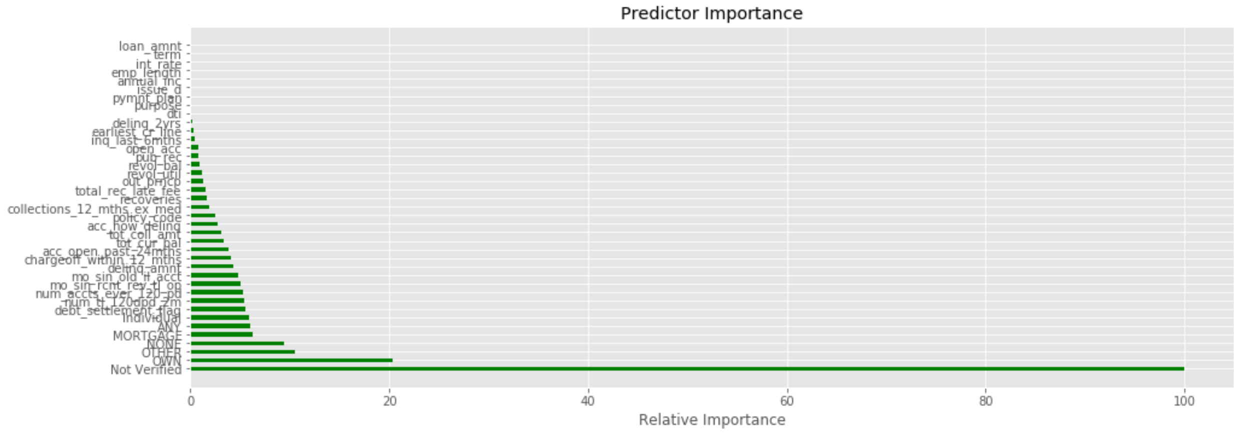
emp_length=1 year}   => {purpose=debt_consolidation} 0.02047802 0.56169769 0.9936887
46294
[183] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=1 year}   0.02047802 0.05774580 0.8796593
46294
[184] {emp_length=10+ years,
purpose=home_improvement}   => {inc_category=Moderate}   0.01907268 0.73694196
1.1865481   43117
[185] {inc_category=Moderate,
purpose=home_improvement}   => {emp_length=10+ years}   0.01907268 0.42530085
1.2853711   43117
[186] {inc_category=Moderate,
emp_length=10+ years}   => {purpose=home_improvement}   0.01907268 0.08021678
1.2052846   43117
[187] {emp_length=3 years,
purpose=credit_card}   => {inc_category=Moderate}   0.01128251 0.58991142 0.9498146
25506
[188] {inc_category=Moderate,
emp_length=3 years}   => {purpose=credit_card}   0.01128251 0.23872187 1.0439094
25506
[189] {inc_category=Moderate,
purpose=credit_card}   => {emp_length=3 years}   0.01128251 0.07995887 1.0000413
25506
[190] {inc_category=Low,
emp_length=3 years}   => {purpose=debt_consolidation} 0.01578693 0.55669251 0.9848342
35689
[191] {emp_length=3 years,
purpose=debt_consolidation} => {inc_category=Low}   0.01578693 0.35495549 1.0945635
35689
[192] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=3 years}   0.01578693 0.08637201 1.0802500
35689
[193] {emp_length=3 years,
purpose=debt_consolidation} => {inc_category=Moderate}   0.02651871 0.59625044
0.9600211   59950
[194] {inc_category=Moderate,
emp_length=3 years}   => {purpose=debt_consolidation} 0.02651871 0.56109842 0.9926286
59950
[195] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=3 years}   0.02651871 0.07477990 0.9352682
59950
[196] {emp_length=< 1 year,
purpose=credit_card}   => {inc_category=Moderate}   0.01172087 0.55635577 0.8957868
26497
[197] {inc_category=Moderate,
emp_length=< 1 year}   => {purpose=credit_card}   0.01172087 0.25677129 1.1228379
26497
[198] {inc_category=Moderate,

purpose=credit_card} => {emp_length=< 1 year} 0.01172087 0.08306556 0.9883974
26497
[199] {inc_category=Low,
emp_length=< 1 year} => {purpose=debt_consolidation} 0.01864405 0.55135786 0.9753967
42148
[200] {emp_length=< 1 year,
purpose=debt_consolidation} => {inc_category=Low} 0.01864405 0.40452630 1.2474232
42148
[201] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=< 1 year} 0.01864405 0.10200363 1.2137416
42148
[202] {emp_length=< 1 year,
purpose=debt_consolidation} => {inc_category=Moderate} 0.02510939 0.54480713
0.8771923 56764
[203] {inc_category=Moderate,
emp_length=< 1 year} => {purpose=debt_consolidation} 0.02510939 0.55007607 0.9731292
56764
[204] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=< 1 year} 0.02510939 0.07080578 0.8425182
56764
[205] {emp_length=2 years,
purpose=credit_card} => {inc_category=Moderate} 0.01269448 0.58199148 0.9370628
28698
[206] {inc_category=Moderate,
emp_length=2 years} => {purpose=credit_card} 0.01269448 0.24391446 1.0666162
28698
[207] {inc_category=Moderate,
purpose=credit_card} => {emp_length=2 years} 0.01269448 0.08996548 0.9985521
28698
[208] {inc_category=Low,
emp_length=2 years} => {purpose=debt_consolidation} 0.01846490 0.55724946 0.9858195
41743
[209] {emp_length=2 years,
purpose=debt_consolidation} => {inc_category=Low} 0.01846490 0.36939073 1.1390769
41743
[210] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=2 years} 0.01846490 0.10102347 1.1212878
41743
[211] {emp_length=2 years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.02899984 0.58014247
0.9340857 65559
[212] {inc_category=Moderate,
emp_length=2 years} => {purpose=debt_consolidation} 0.02899984 0.55720915 0.9857482
65559
[213] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=2 years} 0.02899984 0.08177641 0.9076592
65559
[214] {inc_category=Low,

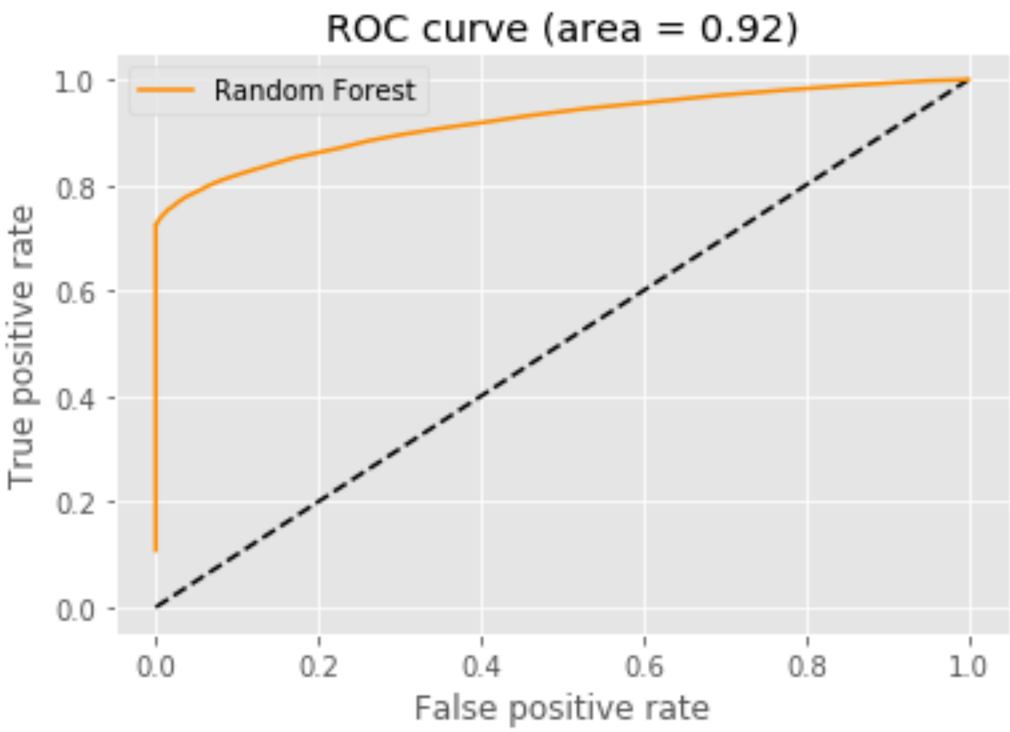
purpose=credit_card} => {emp_length=10+ years} 0.01566926 0.20958123 0.6334096
35423
[215] {emp_length=10+ years,
purpose=credit_card} => {inc_category=Low} 0.01566926 0.22065456 0.6804245
35423
[216] {inc_category=Low,
emp_length=10+ years} => {purpose=credit_card} 0.01566926 0.21745109 0.9508942
35423
[217] {emp_length=10+ years,
purpose=credit_card} => {inc_category=Moderate} 0.05050941 0.71127348 1.1452193
114185
[218] {inc_category=Moderate,
purpose=credit_card} => {emp_length=10+ years} 0.05050941 0.35795905 1.0818465
114185
[219] {inc_category=Moderate,
emp_length=10+ years} => {purpose=credit_card} 0.05050941 0.21243484 0.9289586
114185
[220] {inc_category=Low,
emp_length=10+ years} => {purpose=debt_consolidation} 0.04165539 0.57807503
1.0226616 94169
[221] {inc_category=Low,
purpose=debt_consolidation} => {emp_length=10+ years} 0.04165539 0.22790119
0.6887774 94169
[222] {emp_length=10+ years,
purpose=debt_consolidation} => {inc_category=Low} 0.04165539 0.21748238 0.6706426
94169
[223] {emp_length=10+ years,
purpose=debt_consolidation} => {inc_category=Moderate} 0.13905315 0.72599516
1.1689226 314353
[224] {inc_category=Moderate,
emp_length=10+ years} => {purpose=debt_consolidation} 0.13905315 0.58483626
1.0346228 314353
[225] {inc_category=Moderate,
purpose=debt_consolidation} => {emp_length=10+ years} 0.13905315 0.39211487
1.1850743 314353

APPENDIX 3 – IMPORTANT FEATURE

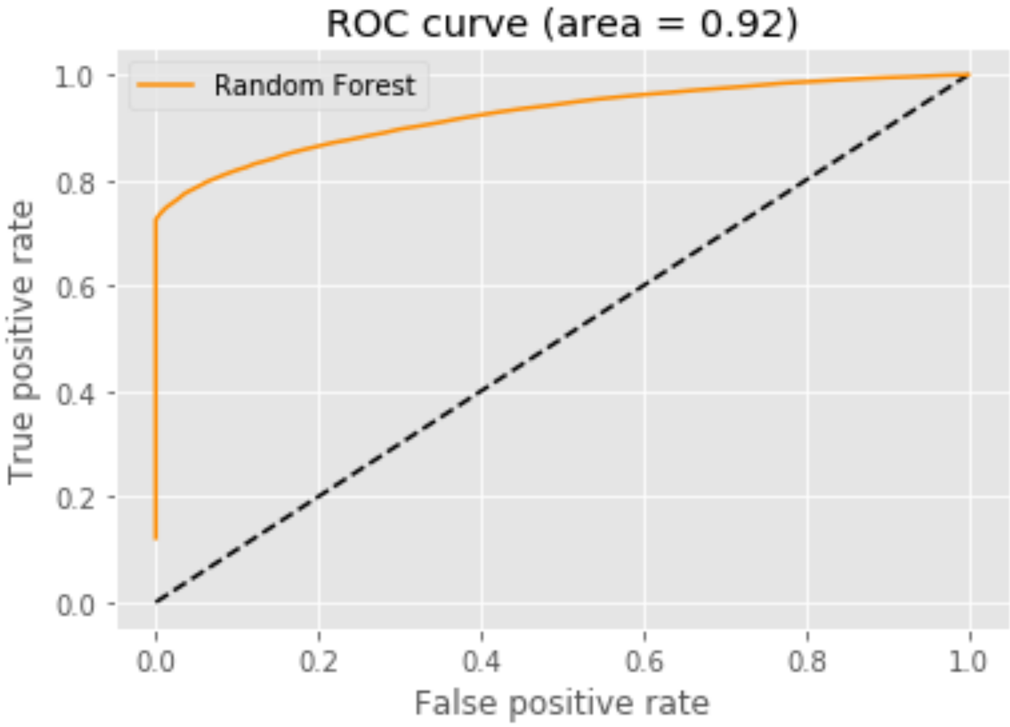
Feature Importance RF Model:



ROC Curve Imbalanced Data set RF Model



ROC Curve Balanced data set RF Model:



APPENDIX 4 – CODING

ASSOCIATION:

```
library('dplyr')
library('arules')
loan4=Loan_statusdata
loan4$inc_category <- cut(loan4$annual_inc, breaks = c(-Inf, 51800, 157500, 500000,Inf),
         labels = c("Low", "Moderate", "High","Rich"))
loan4 = select(loan4, "id","inc_category","emp_length","purpose")
dim(loan4)

summarise_all(loan4, n_distinct)

loanppW=as(loan4,'transactions')
loanppW
summary (loanppW)

if (!require("RColorBrewer")) {
 +  # install color package of R
  +  install.packages("RColorBrewer")
 +  #include library RColorBrewer
  +  library(RColorBrewer)
 + }
itemFrequencyPlot(loanppW,topN=20,type="absolute",col=brewer.pal(8,'Pastel2'), main="Absolute
Item Frequency Plot 4")
itemFrequencyPlot(loanppW, support=0.01)
itemFrequencyPlot(loanppW,topN=20,type="relative",col=brewer.pal(8,'Pastel2'), main="Relative
Item Frequency Plot 4")

rulesW2=apriori(loanppW, parameter=list(support=0.01, confidence=0.01))
inspect (rulesW2)
inspect(subset(rulesW2, lift > 1))
```

```
> rulesW2=apriori(loanppW, parameter=list(support=0.01, confidence=0.01))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target
       0.01    0.1    1 none FALSE            TRUE       5    0.01      1     10  rules
   ext
 FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 22606

set item appearances ...[0 item(s)] done [0.00s].
```

set transactions ...[30 item(s), 2260668 transaction(s)] done [0.99s].
sorting and recoding items ... [23 item(s)] done [0.08s].
creating transaction tree ... done [1.83s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [225 rule(s)] done [0.00s].
creating S4 object ... done [0.90s].

CLUSTERING:

```
library(readxl)
Data_clean_new_final <- read_excel("Desktop/Data_clean_new_final.xlsx")
data = Data_clean_new_final
str(data)
#Scale data but keep dummy variables
scaleContinuous = function(data) {
  binary = apply(data, 2, function(x) {all(x %in% 0:1)})
  data[!binary] = scale(data[!binary])
  return(data)
}

scaleContinuous(data)
new_data = as.data.frame(scaleContinuous(data))
head(new_data)
#perfoming clustering using K-means
set.seed(123)
cl=kmeans(new_data,4)
cl
str(cl) #check cluster information

#plot number of cluster to choose
tss<-rep(0,8)
for (k in 1:8){tss[k]=kmeans(new_data,k)$tot.withinss}
plot(1:8,tss, main = "The Optimal Number of Clusters with the Elbow Method",
     xlab="Number of Clusters",ylab="Total within-cluster sum of squares", col = "red")

set.seed(123)
fviz_nbclust(new_data, kmeans, method = "wss")

#visualize cluster
cluster <-as.data.frame(cl$centers)
library(cluster)
clusplot(data, cl$cluster, color=TRUE, shade=TRUE,
         labels=2, lines=0)
# Save the cluster number in the dataset as column 'Cluster'
data$Cluster <- as.factor(cl$cluster)
```

DATA CLEANING AND RANDOM FOREST ANALYSIS:

Created on Sun May 26 17:22:37 2019

@author: Vandana

```
"""

import pandas as pd
import numpy as np


df1 = pd.read_csv('loan.csv')

df=df1.sample(500000)

df.to_csv(r'loan_new.csv')



pd.set_option('display.max_rows', 150) ##display setting (row)
pd.set_option('display.max_columns', 150) ##display setting (column)
pd.set_option('display.width', 81)  ##display setting (width)


df=df.dropna(thresh=0.9*len(df), axis=1) ##drop columns: missing value >10%
df=df.drop(['title','emp_title',
'zip_code','grade','disbursement_method','initial_list_status','last_credit_pull_d','last_pymnt_d'],
axis=1) ##drop certain columns
df.shape
##states=['NY','FL','TX','CA']
##df =df[df.addr_state.isin(states)]

##sub_grade
df.loc[df.sub_grade == 'A1','sub_grade'] = 1
df.loc[df.sub_grade == 'A2','sub_grade'] = 2
df.loc[df.sub_grade == 'A3','sub_grade'] = 3
df.loc[df.sub_grade == 'A4','sub_grade'] = 4
df.loc[df.sub_grade == 'A5','sub_grade'] = 5
df.loc[df.sub_grade == 'B1','sub_grade'] = 6
df.loc[df.sub_grade == 'B2','sub_grade'] = 7
df.loc[df.sub_grade == 'B3','sub_grade'] = 8
df.loc[df.sub_grade == 'B4','sub_grade'] = 9
df.loc[df.sub_grade == 'B5','sub_grade'] = 10
df.loc[df.sub_grade == 'C1','sub_grade'] = 11
df.loc[df.sub_grade == 'C2','sub_grade'] = 12
df.loc[df.sub_grade == 'C3','sub_grade'] = 13
df.loc[df.sub_grade == 'C4','sub_grade'] = 14
df.loc[df.sub_grade == 'C5','sub_grade'] = 15
df.loc[df.sub_grade == 'D1','sub_grade'] = 16
df.loc[df.sub_grade == 'D2','sub_grade'] = 17
df.loc[df.sub_grade == 'D3','sub_grade'] = 18
df.loc[df.sub_grade == 'D4','sub_grade'] = 19
df.loc[df.sub_grade == 'D5','sub_grade'] = 20
df.loc[df.sub_grade == 'E1','sub_grade'] = 21
```

```
df.loc[df.sub_grade == 'E2','sub_grade'] = 22
df.loc[df.sub_grade == 'E3','sub_grade'] = 23
df.loc[df.sub_grade == 'E4','sub_grade'] = 24
df.loc[df.sub_grade == 'E5','sub_grade'] = 25
df.loc[df.sub_grade == 'F1','sub_grade'] = 26
df.loc[df.sub_grade == 'F2','sub_grade'] = 27
df.loc[df.sub_grade == 'F3','sub_grade'] = 28
df.loc[df.sub_grade == 'F4','sub_grade'] = 29
df.loc[df.sub_grade == 'F5','sub_grade'] = 30
df.loc[df.sub_grade == 'G1','sub_grade'] = 31
df.loc[df.sub_grade == 'G2','sub_grade'] = 32
df.loc[df.sub_grade == 'G3','sub_grade'] = 33
df.loc[df.sub_grade == 'G4','sub_grade'] = 34
df.loc[df.sub_grade == 'G5','sub_grade'] = 35

##term
df.term=df.term.apply(lambda x: x.strip('months'))


##Emp_length

df['emp_length'] = df['emp_length'].astype(str).str.replace('\D+', '')
df.emp_length=df.emp_length.apply(lambda x: x.strip('years'))
df.emp_length=df.emp_length.apply(lambda x: x.strip('<'))
df.loc[df.emp_length == '10+','emp_length']=10
df.emp_length = df.emp_length.replace('', np.nan, regex=True)

##issue_d
df.issue_d=df.issue_d.str.replace('\d+', '')
df.issue_d=df.issue_d.str.replace('-', '')
df.loc[df.issue_d =='Dec','issue_d']=12
df.loc[df.issue_d =='Nov','issue_d']=11
df.loc[df.issue_d =='Oct','issue_d']=10
df.loc[df.issue_d =='Sep','issue_d']=9
df.loc[df.issue_d =='Aug','issue_d']=8
df.loc[df.issue_d =='Jul','issue_d']=7
df.loc[df.issue_d =='Jun','issue_d']=6
df.loc[df.issue_d =='May','issue_d']=5
df.loc[df.issue_d =='Apr','issue_d']=4
df.loc[df.issue_d =='Mar','issue_d']=3
df.loc[df.issue_d =='Feb','issue_d']=2
df.loc[df.issue_d =='Jan','issue_d']=1

##loan_status
df.loan_status = df.loan_status.replace('Current', np.nan, regex=True)
df.loc[df.loan_status == 'Fully Paid','loan_status'] = 0
df.loc[df.loan_status == 'Does not meet the credit policy. Status:Fully Paid','loan_status'] = 0
df.loc[df.loan_status == 'Late (31-120 days)','loan_status'] = 1
df.loc[df.loan_status == 'In Grace Period','loan_status'] = 1
```

```
df.loc[df.loan_status == 'Charged Off','loan_status'] = 1
df.loc[df.loan_status == 'Does not meet the credit policy. Status:Charged Off','loan_status'] = 1
df.loc[df.loan_status == 'Late (16-30 days)','loan_status'] = 1
df.loc[df.loan_status == 'Default','loan_status'] = 1



#Purpose
df.loc[df.purpose =='credit_card','purpose']= 1
df.loc[df.purpose =='debt_consolidation','purpose']= 2
df.loc[df.purpose =='house','purpose']= 3
df.loc[df.purpose =='car','purpose']= 4
df.loc[df.purpose =='other','purpose']= 5
df.loc[df.purpose =='vacation','purpose']= 6
df.loc[df.purpose =='home_improvement','purpose']= 7
df.loc[df.purpose =='small_business','purpose']= 8
df.loc[df.purpose =='major_purchase','purpose']= 9
df.loc[df.purpose =='medical','purpose']= 10
df.loc[df.purpose =='renewable_energy','purpose']= 11
df.loc[df.purpose =='moving','purpose']= 12
df.loc[df.purpose =='wedding','purpose']= 13
df.loc[df.purpose =='educational','purpose']= 14

##earliest_cr_line

df['earliest_cr_line'] = df['earliest_cr_line'].astype(str).str.replace('\D+', '')

##hardship_flag
df.loc[df.hardship_flag =='N','hardship_flag']= 0
df.loc[df.hardship_flag == 'Y','hardship_flag']= 1

##debt_settlement_flag
df.loc[df.debt_settlement_flag =='N','debt_settlement_flag']= 0
df.loc[df.debt_settlement_flag =='Y','debt_settlement_flag']= 1

##categorical into dummy
d_home_ownership = pd.get_dummies(df['home_ownership'])
d_application_type = pd.get_dummies(df['application_type'])
d_verification_status = pd.get_dummies(df['verification_status'])
d_addr_state = pd.get_dummies(df['addr_state'])

##join multiple dummy variables
df = pd.concat([df, d_application_type,d_home_ownership,d_verification_status,d_addr_state], axis=1)

###drop categerious columns
df=df.drop(['home_ownership', 'application_type','verification_status','addr_state'], axis=1)

###pymnt_plan
df.loc[df.pymnt_plan =='n','pymnt_plan']= 0
```

```
df.loc[df.pymnt_plan =='y','pymnt_plan']= 1



df = df.dropna(axis=0)
df.isnull().sum()


df['sub_grade']=df['sub_grade'].astype(object).astype(int)
df['term']=df['term'].astype(object).astype(int)
df['emp_length']=df['emp_length'].astype(object).astype(int)
df['issue_d']=df['issue_d'].astype(object).astype(int)
df['loan_status']=df['loan_status'].astype(object).astype(int)
df['pymnt_plan']=df['pymnt_plan'].astype(object).astype(int)
df['purpose']=df['purpose'].astype(object).astype(int)
df['earliest_cr_line']=df['earliest_cr_line'].astype(object).astype(int)
df['hardship_flag']=df['hardship_flag'].astype(object).astype(int)
df['debt_settlement_flag']=df['debt_settlement_flag'].astype(object).astype(int)

df.to_csv(r'jee_data.csv')

@author: Vandana
"""

import numpy as np
import pandas as pd
from sklearn import cross_validation
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.metrics import roc_auc_score, roc_curve, auc, classification_report
from sklearn.metrics import f1_score, precision_score, recall_score
from sklearn.metrics import mean_squared_error, cohen_kappa_score, make_scorer
from sklearn.metrics import confusion_matrix, accuracy_score, average_precision_score
from sklearn.metrics import precision_recall_curve, SCORERS
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RandomizedSearchCV

from matplotlib import pyplot as plt
import matplotlib
matplotlib.style.use('ggplot')
from scipy.stats import randint as sp_randint
import seaborn as sns



df = pd.read_csv('Data_Imbalanced.csv')
```

```
corr_matrix = df.corr().abs()

upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.bool))
to_drop = [column for column in upper.columns if any(upper[column] > 0.50)]
to_drop.remove('loan_status')
df=df.drop(df[to_drop], axis=1)

x=df[['loan_amnt', 'term', 'int_rate', 'emp_length', 'annual_inc', 'issue_d',
      'pymnt_plan', 'purpose', 'dti', 'delinq_2yrs',
      'earliest_cr_line', 'inq_last_6mths', 'open_acc', 'pub_rec', 'revol_bal',
      'revol_util', 'out_prncp', 'total_rec_late_fee', 'recoveries',
      'collections_12_mths_ex_med', 'policy_code', 'acc_now_delinq',
      'tot_coll_amt', 'tot_cur_bal', 'acc_open_past_24mths',
      'chargeoff_within_12_mths', 'delinq_amnt', 'mo_sin_old_il_acct',
      'mo_sin_rcnt_rev_tl_op', 'num_accts_ever_120_pd', 'num_tl_120dpd_2m',
      'debt_settlement_flag', 'Individual', 'ANY', 'MORTGAGE', 'NONE', 'OTHER',
      'OWN', 'Not Verified']]
y=df['loan_status']

##Data Scaling
sc = StandardScaler()
x = sc.fit_transform(x)

## Spliting data into test and train
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)

## Model Building And prediction
clf=RandomForestClassifier(n_estimators=100)
clf.fit(x_train,y_train)
y_pred=clf.predict(x_test)
rf_probs = clf.predict_proba(x_test)[:, 1]

# Feature Imporatance

def Plot_predictor_importance(best_model, feature_columns):
   feature_importance = best_model.feature_importances_
   feature_importance = 100.0 * (feature_importance / feature_importance.max())
   sorted_idx = np.argsort(feature_importance)
   y_pos  = np.arange(sorted_idx.shape[0]) + .5
   fig, ax = plt.subplots()
   fig.set_size_inches(15.5, 5.5, forward=True)
   ax.barh(y_pos,
         feature_importance[sorted_idx],
         align='center',
         color='green',
         ecolor='black',
         height= .5)
```

```
    ax.set_yticks(y_pos)
    ax.set_yticklabels(feature_columns)
    ax.invert_yaxis()
    ax.set_xlabel('Relative Importance')
    ax.set_title('Predictor Importance')
    plt.show()


feature_columns=['loan_amnt', 'term', 'int_rate', 'emp_length', 'annual_inc', 'issue_d',
       'pymnt_plan', 'purpose', 'dti', 'delinq_2yrs',
       'earliest_cr_line', 'inq_last_6mths', 'open_acc', 'pub_rec', 'revol_bal',
       'revol_util', 'out_prncp', 'total_rec_late_fee', 'recoveries',
       'collections_12_mths_ex_med', 'policy_code', 'acc_now_delinq',
       'tot_coll_amt', 'tot_cur_bal', 'acc_open_past_24mths',
       'chargeoff_within_12_mths', 'delinq_amnt', 'mo_sin_old_il_acct',
       'mo_sin_rcnt_rev_tl_op', 'num_accts_ever_120_pd', 'num_tl_120dpd_2m',
       'debt_settlement_flag', 'Individual', 'ANY', 'MORTGAGE', 'NONE', 'OTHER',
       'OWN', 'Not Verified']


Plot_predictor_importance(clf,feature_columns)



## 10 fold Cross Validation
scores = cross_val_score(clf, x, y, cv=10)
print(" 10 fold Cross Validation score:",scores)
#Accuracy
print("Accuracy: Test Data:",metrics.accuracy_score(y_test, y_pred))
print("Precison: Test Data:",metrics.precision_score(y_test, y_pred))
print("Recall: Test Data:",recall_score(y_test, y_pred))
print("F-Score: Test Data:",metrics.f1_score(y_test, y_pred))




## Confusion Matrix
conf_mat = confusion_matrix(y_test, y_pred)
##Plot Confusion Matrix
plt.clf()
plt.imshow(conf_mat, interpolation='nearest', cmap=plt.cm.Wistia)
classNames = ['Negative','Positive']
plt.title('Customer Will Deafult Confusion Matrix - Test Data')
plt.ylabel('True label')
plt.xlabel('Predicted label')
tick_marks = np.arange(len(classNames))
plt.xticks(tick_marks, classNames, rotation=45)
plt.yticks(tick_marks, classNames)
s = [['TN','FP'], ['FN', 'TP']]
```

```
for i in range(2):
    for j in range(2):
        plt.text(j,i, str(s[i][j])+" = "+str(conf_mat[i][j]))
plt.show()

#ROC Curve
auc = roc_auc_score(y_test, rf_probs)
false_positive, true_positive, _ = roc_curve(y_test, rf_probs)
plt.figure()
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(false_positive, true_positive, color='darkorange', label='Random Forest')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve (area = %0.2f)' % auc)
plt.legend(loc='best')
plt.show()
```

DATA CLEANING AND LDA

```
@author: qiqintian



"""

import pandas as pd
import numpy as np

df = pd.read_csv('loan.csv')

pd.set_option('display.max_rows', 150) ##display setting (row)
pd.set_option('display.max_columns', 150) ##display setting (column)
pd.set_option('display.width', 81)  ##display setting (width)


df=df.dropna(thresh=0.9*len(df), axis=1) ##drop columns: missing value >10%
df.shape

df=df.drop(['title','emp_title',
'zip_code','grade','disbursement_method','initial_list_status','last_credit_pull_d','last_pymnt_d'],
axis=1) ##drop certain columns
df.shape

##sub_grade
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('A1','1'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('A2','2'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('A3','3'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('A4','4'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('A5','5'))
```

```
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('B1','6'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('B2','7'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('B3','8'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('B4','9'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('B5','10'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('C1','11'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('C2','12'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('C3','13'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('C4','14'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('C5','15'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('D1','16'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('D2','17'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('D3','18'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('D4','19'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('D5','20'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('E1','21'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('E2','22'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('E3','23'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('E4','24'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('E5','25'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('F1','26'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('F2','27'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('F3','28'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('F4','29'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('F5','30'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('G1','31'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('G2','32'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('G3','33'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('G4','34'))
df.sub_grade=df.sub_grade.apply(lambda x: x.replace('G5','35'))
##term
df.term=df.term.apply(lambda x: x.strip('months'))

##emp_length
##remove all non_digits
df['emp_length'] = df['emp_length'].astype(str).str.replace('\D+', '')


##issue_d
df.issue_d=df.issue_d.str.replace('\d+', '')
df.issue_d=df.issue_d.str.replace('-', '')
df.issue_d=df.issue_d.apply(lambda x: x.replace('Dec','12'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Nov','11'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Oct','10'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Sep','9'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Aug','8'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Jul','7'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Jun','6'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('May','5'))
```

```
df.issue_d=df.issue_d.apply(lambda x: x.replace('Apr','4'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Mar','3'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Feb','2'))
df.issue_d=df.issue_d.apply(lambda x: x.replace('Jan','1'))



##loan_status
df.loan_status = df.loan_status.replace('Current', 2, regex=True)
df.loan_status=df.loan_status.str.replace('Fully Paid','0')
df.loan_status=df.loan_status.str.replace('Charged Off','1')
df.loan_status=df.loan_status.str.replace('In Grace Period','1')
### change the values=1 if the condition matches 'Late (31-120 days)'
df.loan_status[df.loan_status == 'Late (31-120 days)'] = 1
df.loan_status[df.loan_status == 'Late (16-30 days)'] = 1


###pymnt_plan
df.pymnt_plan=df.pymnt_plan.str.replace('n','0')
df.pymnt_plan=df.pymnt_plan.str.replace('y','1')

##purpose
df.purpose=df.purpose.str.replace('credit_card','1')
df.purpose=df.purpose.str.replace('car','2')
df.purpose=df.purpose.str.replace('debt_consolidation','3')
df.purpose=df.purpose.str.replace('home_improvement','4')
df.purpose=df.purpose.str.replace('house','5')
df.purpose=df.purpose.str.replace('major_purchase','6')
df.purpose=df.purpose.str.replace('medical','7')
df.purpose=df.purpose.str.replace('moving','8')
df.purpose=df.purpose.str.replace('renewable_energy','9')
df.purpose=df.purpose.str.replace('small_business','10')
df.purpose=df.purpose.str.replace('vacation','11')
df.purpose=df.purpose.str.replace('other','12')
df.purpose=df.purpose.str.replace('wedding','13')
df.purpose=df.purpose.str.replace('14al','14')


##earliest_cr_line
df.earliest_cr_line
##replace non-digit with ''
df['earliest_cr_line'] = df['earliest_cr_line'].astype(str).str.replace('\D+', '')

##hardship_flag
df.hardship_flag=df.hardship_flag.str.replace('N','0')
df.hardship_flag=df.hardship_flag.str.replace('Y','1')

##debt_settlement_flag
df.debt_settlement_flag=df.debt_settlement_flag.str.replace('N','0')
```

```
df.debt_settlement_flag=df.debt_settlement_flag.str.replace('Y','1')

##categorical into dummy
home_ownership = pd.get_dummies(df['home_ownership'])
application_type = pd.get_dummies(df['application_type'])
addr_state = pd.get_dummies(df['addr_state'])
verification_status = pd.get_dummies(df['verification_status'])

##join multiple dummy variables
df = pd.concat([df, application_type,addr_state,home_ownership,verification_status], axis=1)

df.head()

###drop categerious columns
df=df.drop(['home_ownership','addr_state', 'application_type','verification_status'], axis=1) ##drop
certain columns



##### general insight
df.info()
df.describe()
df.shape

####check how many rows have missing data
df1=df.drop(['loan_status'], axis=1) ##drop certain columns
df1.shape[0] - df1.dropna().shape[0]

### drop rows with missing value
df1 = df1.dropna(axis=0)

##### count the nmuber of missing value in the predicted variable
df.loan_status.isna().sum()

### fill the certan column with value '2'
df['loan_status'].fillna(2, inplace=True)

###drop rows with missing value in the whole dataset
df = df.dropna(axis=0)

###count the total number of missing value: make sure the data has no missing value
df.isnull().sum()

#### replace 2 with N/A
df.loan_status = df.loan_status.replace(2, np.nan, regex=True)

##export the scv file
df.to_csv(r'/Users/qiqintian/Desktop/BA learning material/quarter3/Data Mining
/project/project5.csv')
```

```
Created on Sat May 25 13:24:38 2019

@author: qiqintian
"""

df = pd.read_csv('loan1.csv')

###select data without missing value in a certain column
df1=df[df.loan_status.notnull()]
###select data with missing value in a certain column
df2=df[df.loan_status.isnull()]


##ocnvert df into np
A=df1.get_values()

df1.head()
df.shape

import numpy as np
import matplotlib.pyplot as plt
from sklearn import cross_validation
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from matplotlib import colors
# %% Read data from csv file
A = np.loadtxt('project.csv', delimiter=',')

## get the column number given the column name
df.columns.get_loc("loan_status")

y = A[:, 8]
#Remove targets from input data
x = A[:, 8:]

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x = sc.fit_transform(x)

####drop collinear variables
from statsmodels.stats.outliers_influence import variance_inflation_factor

def calculate_vif_(x, thresh=20.0):
    variables = list(range(x.shape[1]))
    dropped = True
    while dropped:
        dropped = False
        vif = [variance_inflation_factor(x.iloc[:, variables].values, ix)
```

```
            for ix in range(x.iloc[:, variables].shape[1])]

        maxloc = vif.index(max(vif))
        if max(vif) > thresh:
            print('dropping \'' + x.iloc[:, variables].columns[maxloc] +
                  '\' at index: ' + str(maxloc))
            del variables[maxloc]
            dropped = True

    print('Remaining variables:')
    print(x.columns[variables])
    return x.iloc[:, variables]

x.shape


# Create correlation matrix
corr_matrix = df.corr().abs()

# Select upper triangle of correlation matrix
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.bool))

# Find index of feature columns with correlation greater than 0.95
to_drop = [column for column in upper.columns if any(upper[column] > 0.50)]

# Drop features
df=df.drop(df[to_drop], axis=1)

###discriminant analisis
import numpy as np
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
clf = LinearDiscriminantAnalysis()
clf.fit(x, y)
###predict
print(clf.predict([[-0.8, -1]]))


#### k fold cross validation
from sklearn.model_selection import cross_val_score
##model:clf cv:k
scores = cross_val_score(clf, x, y, cv=10)
scores


###using confusion matrix as scoring metric in cross validation
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
y_pred = cross_val_predict(clf, x, y, cv=10)
conf_mat = confusion_matrix(y, y_pred)
```

```
import statsmodels.api as sm
from sklearn.datasets import make_blobs
import statsmodels.formula.api as smf

x, y = make_blobs(n_samples=50, n_features=2, cluster_std=5.0,
        centers=[(0,0), (2,2)], shuffle=False, random_state=12)

logit_model = sm.Logit(y, sm.add_constant(x)).fit()
print(clf.summary())
```