# *STAT 508 Final*

# *Peer-To-Peer Lending*

---

## INTRODUCTION

Fundamentally, all business transactions boil down to the risk-reward payout, from the cost of your morning coffee to the layout of your favorite store, to making an investment. The goal is to minimize risk while maximizing the payout. Simple enough. But deciding this boundary can be quite challenging when dealing with the human element: 'how can we be sure the human factor in this equation will behave the way we anticipate?' While estimating human response is generally out of the scope of our capabilities, many institutions have made noble attempts at best calculating the risk-reward boundary. In the field of Individual Lending, this concept is critical. A financial institution can lend a sum of money to an individual, under the constraints that the recipient will eventually pay back the loan in full with additional capital as interest (payout). But how can we be sure an individual will follow these rules and guarantee the lender's reward? Our analysis attempts to pinpoint certain predictive demographic information about loan recipients to cultivate a model to predict whether or not a loan recipient will pay back their loan or default (charge-off).

## DATA

Our data set for this analysis is a subset of the LendingClub dataset from Kaggle (https://www.kaggle.com/wordsforthewise/lending-club). LendingClub is a US peer-to-peer lending company, headquartered in San Francisco, California. The platform was the first peer-to-peer lender to register its offerings as securities with the Securities and Exchange Commission (SEC), and to offer loan trading on a secondary market. LendingClub is the world's largest peer-to-peer lending platform. The dataset contains 27 columns and 396,030 observations collected from 2008 to 2016. In the table below are the dataset's original columns and their descriptions:

| Column | Description |
|---|---|
| loan_amnt | The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value |
| term | The number of payments on the loan. Values are in months and can be either 36 or 60 |
| int_rate | Interest Rate on the loan |
| installment | The monthly payment owed by the borrower if the loan originates |
| grade | LC assigned loan grade |

| | |
|---|---|
| sub_grade | LC assigned loan subgrade |
| emp_title | The job title supplied by the Borrower when applying for the loan |
| emp_length | Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years |
| home_ownership | The home ownership status provided by the borrower during registration or obtained from the credit report. Possible values are: RENT, OWN, MORTGAGE, ANY, OTHER, NONE |
| annual_inc | The self-reported annual income provided by the borrower during registration |
| verification_status | Indicates if income was verified by LC, not verified, or if the income source was verified |
| issue_d | The month which the loan was funded |
| loan_status | The current status of the loan |
| purpose | A category provided by the borrower for the loan request |
| title | The loan title provided by the borrower |
| zip_code | The first 3 numbers of the zip code provided by the borrower in the loan application |
| addr_state | The state provided by the borrower in the loan application |
| dti | A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower's self-reported monthly income |
| earliest_cr_line | The month the borrower's earliest reported credit line was opened |
| open_acc | The number of open credit lines in the borrower's credit file |
| pub_rec | Number of derogatory public records |
| revol_bal | Total credit revolving balance |
| revol_util | Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit |
| total_acc | The total number of credit lines currently in the borrower's credit file |
| initial_list_status | The initial listing status of the loan. Possible values are – W, F |
| application_type | Indicates whether the loan is an individual application or a joint application with two co-borrowers |
| mort_acc | Number of mortgage accounts |
| pub_rec_bankruptcies | Number of public record bankruptcies |
| address | Borrower's listed address at the time of the loan application |

Further, below are select distributions from the categorical data columns.

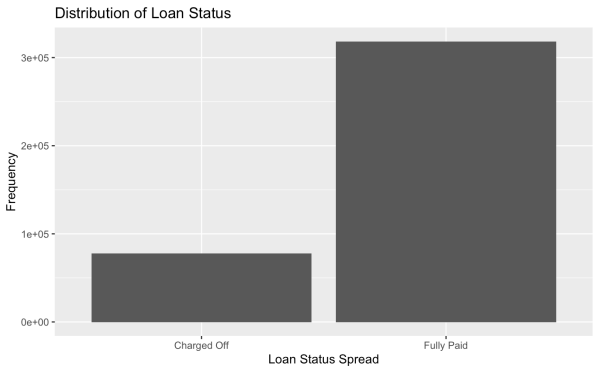


Figure 1

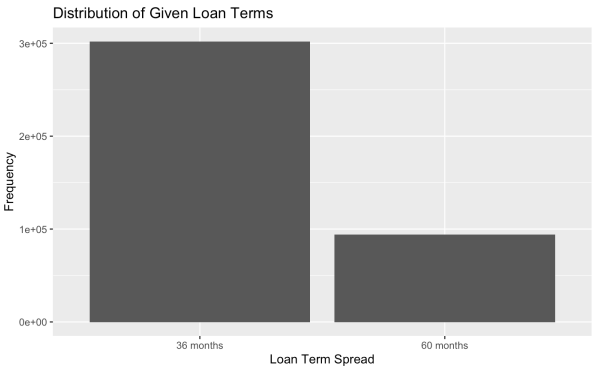


Figure 2

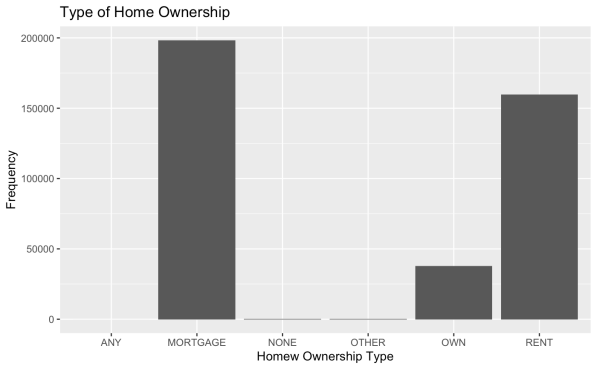


Figure 3

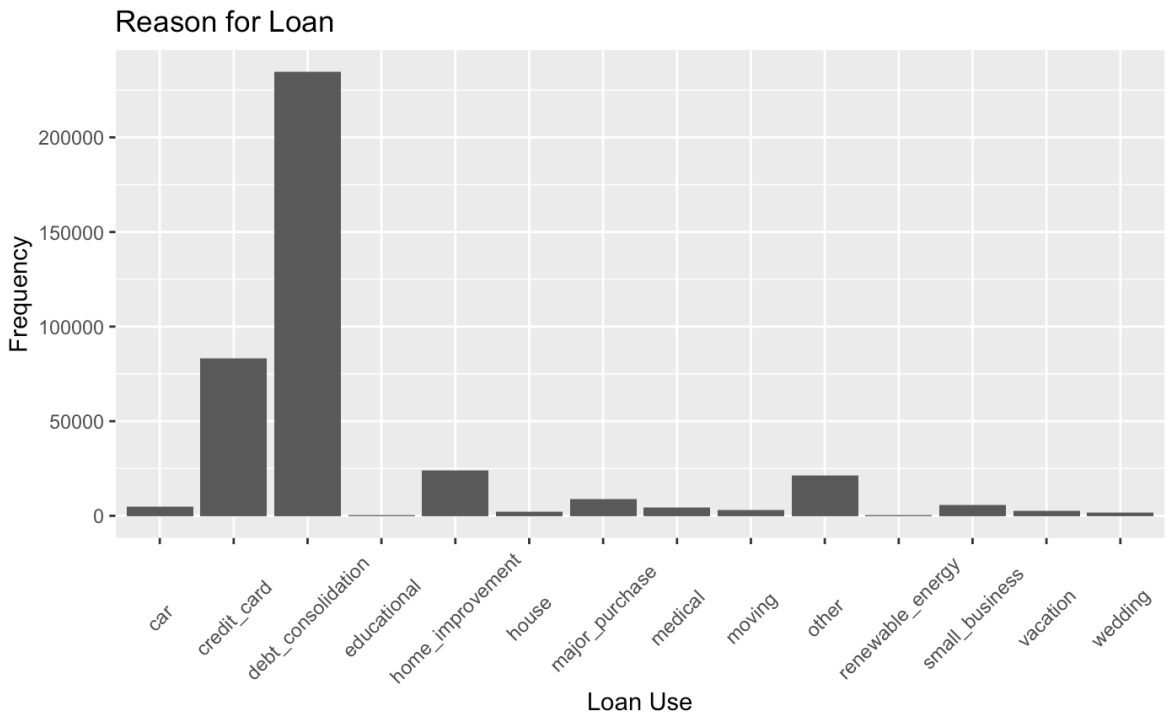


Figure 4

From the Figure 1 and Figure 2, we can make the following observations: the dataset contains more “Fully Paid” loans than “Charged Off” and a majority of the loans given were in the shorter, 36-month/ 3-year time frame than the 60-month/ 5-year time frame. Figure 3 gives us an impression of the participants’ home ownership status. From the plot, we can see most of the participants pay a mortgage, second-most pay rent for their home, while third-most own their home. Finally, Figure 4 displays the use listed in the original loan application. We can see an overwhelming majority of the loans provided were for the purpose of debt consolidation.

Below are included select distributions for the numerical data columns.

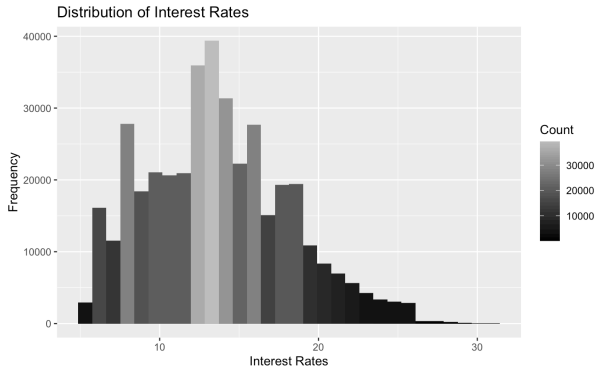


Figure 5

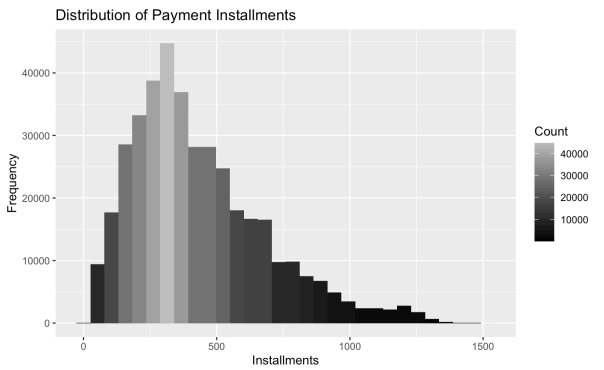


Figure 6

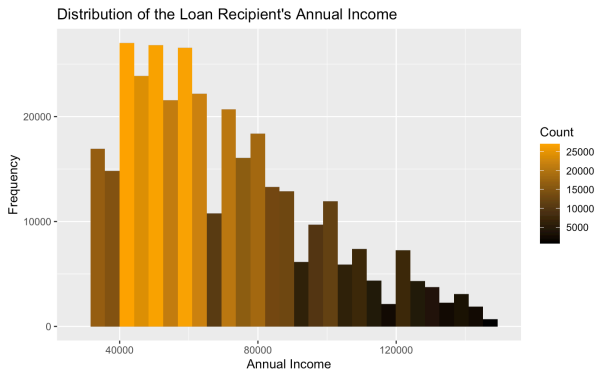


Figure 7

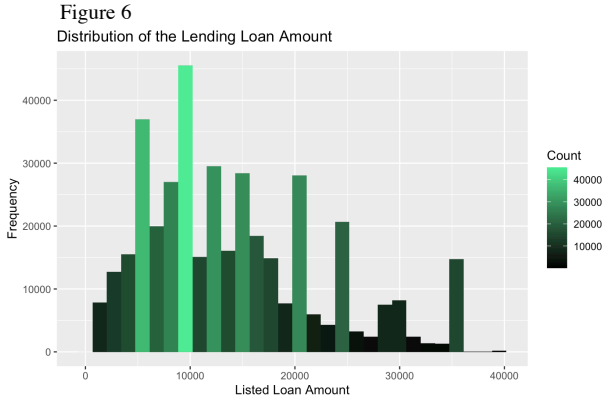


Figure 8

Figure 5 shows the distribution of interest rates given for the loans. There doesn’t appear to be a clear distribution pattern, and the average interest rate is 13.64%. Figure 6 shows the distribution of lendee payment installments before the loan was paid off. Here we can see a distinct Chi-Squared distribution with an average of 431.85 payment installments. In Figures 7 and 8, we have highligted distributions for some key demographics. Figure 7 displays the distribution of lendee’s annual income. Here we can see a vague skewed distribtion with an average value of $74,203. Note that for this plot, only 90% of the data is displayed due to some extreme outliers. The plotted data represents values from the 5$^{th}$ quantile through the 95$^{th}$ quantile. Lastly, Figure 8 shows the distribution of loan amounts given. There is not a clear distrbution here and the average loan amount is $14,114.

To get a better understanding of the relationships between variables in the dataset, see below the correlation plot:

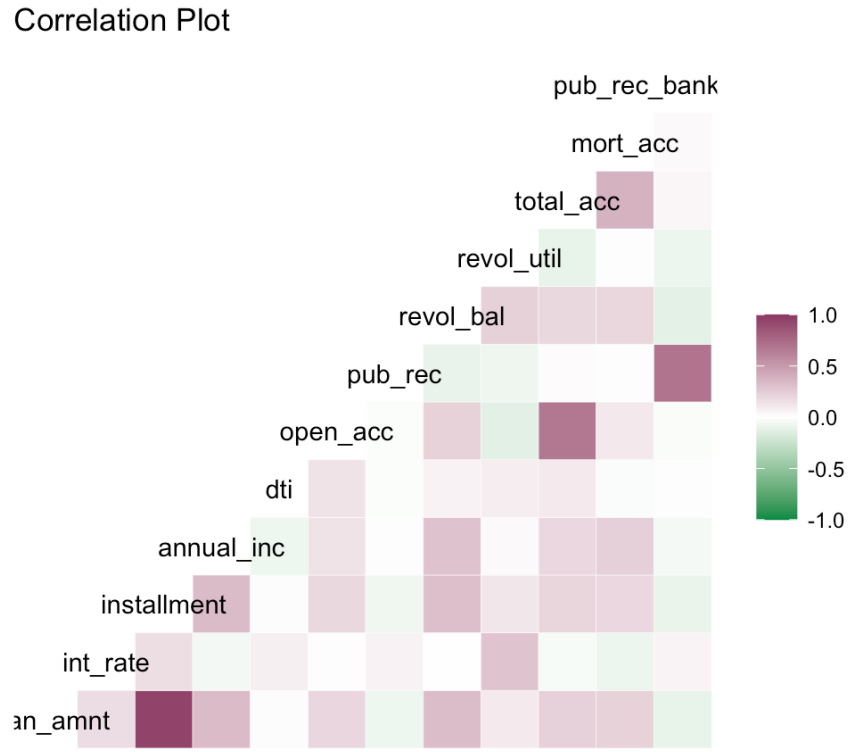


Figure 9

The color convention shows that dark pink elements are highly positively correlated while dark green elements are highly negatively corelated. The only relationships that significantly stand out show that loan amount and interest rate are highly positively correlated, open account and total account are somewhat positively correlated, and public record and public record bank are somewhat positively correlated. While the latter two relations are expected, the first relationship is an important observation. Larger loan amounts will be charged a higher interest rate.

# ANALYSIS

With a broad understanding of our dataset's characteristics, we can now begin the process of developing a statistical model to predict whether or not a loan recipient will pay back their loan. We will approach this goal using the following classification methods: Logistic Regression, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbors, Support Vector Machines, and Classification Trees.

# Logistic Regression Model

Logistic modeling is the process of modeling the probability of a discrete outcome given the predictor variable. In other words, logistic regression models the relationship between P(Y=1|X) and X using the logistic function:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

The process uses maximum likelihood to estimate the predictors in the logistic regression model. Logistic regression operates under the following assumptions and conditions:

1. The response variable must be binary
2. The observations must be independent of each other
3. There should be little or no multicollinearity among the predictors
4. The model assumes linearity of predictors and log odds
5. The sample size must be large

Using logistic regression, we want to determine the appropriate predictors from the dataset to predict the loan status for each borrower, that is, if a borrower is able to pay back the loan. To start our analysis, we first use logistic regression to test the following hypothesis:

$H_0$: loan_amnt = term = int_rate = installment = grade = sub_grade = emp_length = home_ownership = annual_inc = verification_status = purpose = dti = open_acc = pub_rec = revol_bal = revol_util = total_acc = initial_list_status = application_type = mort_acc = pub_rec_bankruptcies = 0

$H_A$: loan_amnt = term = int_rate = installment = grade = sub_grade = emp_length = home_ownership = annual_inc = verification_status = purpose = dti = open_acc = pub_rec = revol_bal = revol_util = total_acc = initial_list_status = application_type = mort_acc = pub_rec_bankruptcies ≠ 0

In layman's terms, our null hypothesis is that none of the listed predictors are statistically significant for predicting loan_status, while the alternative is that not all predictors are insignificant. Notice here that we excluded emp_title, issue_d, title, earliest_cr_line, and address. These values were initially considered too varied due to the nature of the content to be of significance. Using the glm() function from the stats package in R, we construct our first model to test the hypothesis, using the family = "binomial" parameter and data from our cleaned dataset called numdata.

Numdata varies from the initial dataset due to the following modifications: NA values in mort_acc and pub_rec_bankruptcies were assigned to zero; emp_title, title, and address were removed from the dataset; NA values were omitted from emp_length; and loan_status was converted from "Fully Paid" and "Charged Off" to 1 and 0, respectively (this column is called loan_status_num). These modifications were made through a transitionary dataset called finaldata. Further, additional modifications were made for the purpose of analyzing our first model: term and emp_length were converted to a numeric value; grade was converted from values A-G to values 1-7, respectively; and home_ownership was converted from values OWN, MORTGAGE, RENT, ANY, OTHER, and NONE to values 1 – 6, respectively. These

modifications were done to discover whether numerical values versus categorical values impacted the results after model comparison between the datasets.

From our numdata model, we use the summary() function to obtain a large table with predictor estimates, standard errors, z values, and p values. Comparing the p-values to our test significance of 0.05, we find that the predictor initial_list_status is not significant. So, our first model's results lead us to reject the null hypothesis. Before moving to the next model, we stop to make another adjustment to the predictors listed. The logistic regression model requires there to be little or no multicollinearity among the predictors. Given that the predictors grade and sub_grade are directly correlated by design, we choose to omit the sub_grade predictor in further analysis. For our first test model, all grade values were shown to be statistically significant while some of the sub_grade values were shown to be insignificant predictors. The removal of sub_grade will improve our model integrity for further analysis.

Repeating the modeling process, we move forward with our model now excluding the predictors initial_list_status and sub_grade and fit this to the same numdata dataset in addition to the numdata1 dataset. Numdata1 is a similar subset of our original dataset but does not have the numeric conversion modifications made (term and emp_length were converted to a numeric value; grade was converted from values A-G to values 1-7, respectively; and home_ownership was converted from values OWN, MORTGAGE, RENT, ANY, OTHER, and NONE to values 1 – 6, respectively). Apart from the transition of loan_status to a numeric column, the data is largely untouched from the finaldata dataset. Both models show that using this restricted predictor space yields home_ownership is no longer considered a statistically significant predictor. This model is repeated with both datasets and yields the same result. So, we can conclude that we will reject our initial null hypothesis for a significance level of 0.05 and conclude that there is a statistically significant relationship between loan_status, loan_amnt, term, int_rate, installment, grade, emp_length, annual_inc, verification_status, purpose, dti, open_acc, pub_rec, revol_bal, revol_util, total_acc, application_type, mort_acc, and pub_rec_bankruptcies. The model is of the form:

**Loan_status_num** = 3.550141 + 0.00001567655**loan_amnt** - 0.5645631**term60 months** - 0.01059888**int_rate** - 0.0008067713**installment** - 0.5507270**gradeB** - 0.9856544**gradeC** - 1.254043**gradeD** - 1.446836**gradeE** -1.530325**gradeF** - 1.643154**gradeG** + 0.1355364**emp_length10+ years** + 0.09283701**emp_length2 years** + 0.08728216**emp_length3 years** + 0.09.930398**emp_length4 years** + 0.09974195**emp_length5 years** + 0.1231149**emp_length6 years** + 0.1081749**emp_length7 years** + 0.07666886**emp_length8 years** + 0.06980994**emp_length9 years** + 0.000002562455**annual_inc** - 0.1569661**verification_statusSource Verified** – 0.04.240239**verification_statusVerified** - 0.07001807**purposecredit_card** - 0.1181851**purposedebt_consolidation** – 0.4274448**purposeeducational** - 0.1318816**purposehome_improvement** - 0.02808063**purposehouse** - 0.1452204**purposemajor_purchase** - 0.2249776**purposemedical** - 0.2351200**purposemoving** - 0.1414391**purposeother** - 0.3540098**purposerenewable_energy** - 0.5855912**purposesmall_business** - 0.1306199**purposevacation** + 0.3668408**purposewedding** - 0.02595605**dti** - 0.02154169**open_acc** - 0.09467273**pub_rec** + 0.000003498357**revol_bal** - 0.002733755**revol_util** + 0.009362366**total_acc** - 0.3553505**application_typeINDIVIDUAL** +

1.010164**application_typeJOINT** + 0.05008985**mort_acc** + 0.07336196**pub_rec_bankruptcies**

Given that we have the logistic regression model form, we now move forward to set up our testing and training dataset models using these predictors. Given that our cleaned dataset (numdata1) has 351,604 observations, we set up our training set to be a randomly selected half of numdata1 while the testing dataset is composed of the other half. After fitting our model to the training subset of numdata1, we use the predict() function to produce a vector of predictions P(Y=1|X). Given that we used the convention loan_status Fully Paid = 1 and Charged Off = 0, we reassign these probabilities where values over 0.5 yield Fully Paid and values below 0.5 yield Charged Off. Then comparing our predictions to the true values in the dataset, we obtain the following confusion matrix, or error matrix, to observe the model's performance:

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 1707 | 1358 |
| Fully Paid | 32238 | 140796 |

As we can see, values along the main diagonal are the number of cases for each class (Charged Off, Fully Paid) where the model predicted the same value as the true value given. Using this knowledge, our training model made correct predictions 81.05% of the time and made incorrect predictions 18.95% of the time. Finally, we fit our logistic regression model to the test dataset to evaluate its performance on "new" data. This outcome is crucial to our central goal. Using again glm() of family binomial and our testing subset, we create the test model and calculate its predictions. Included below are marginal effects plots for select predictors using our final test logistic regression model:

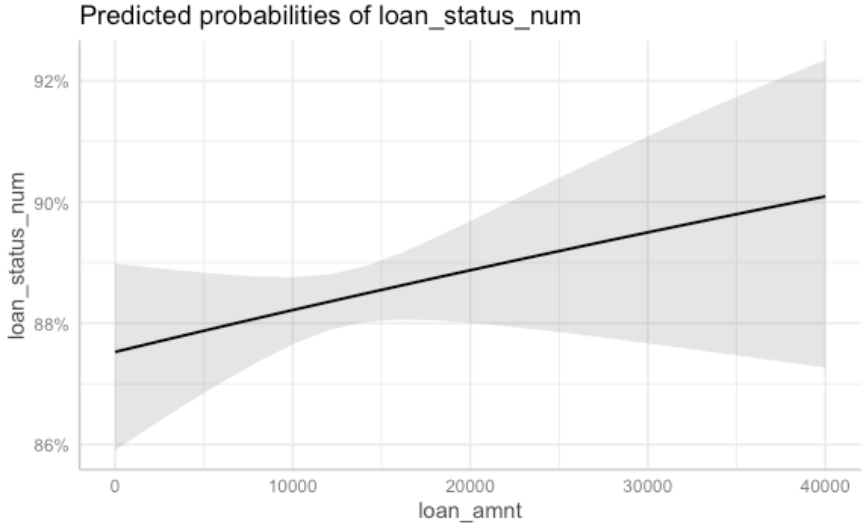


Figure 10

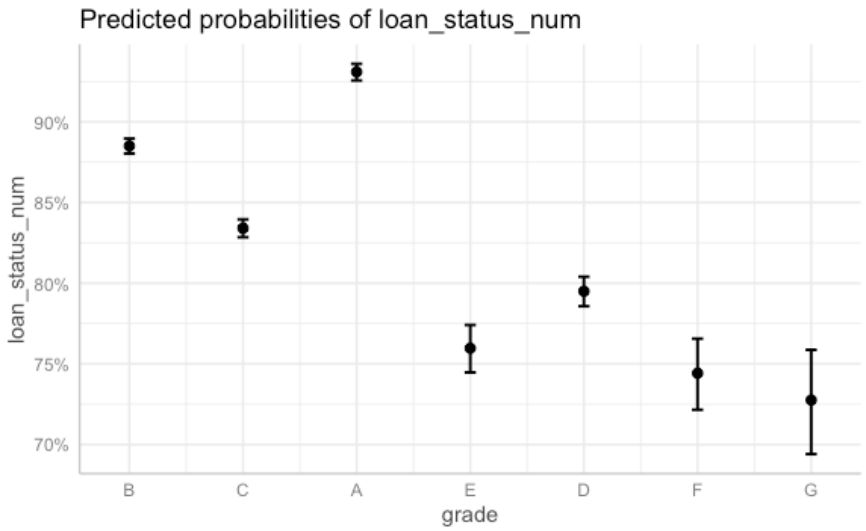

Figure 11

Below is the resulting confusion matrix:

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 1647 | 1348 |
| Fully Paid | 32125 | 140607 |

From the table, we can see that our developed logistic regression model for the response loan_status makes correct new predictions 80.92% of the time and incorrect new predictions 19.08% of the time. Using prediction accuracy as our basis or comparison, this model is fairly "good" since a significant majority of the predictions are correctly made using the predictors we initially identified. But can we make a better model?

## Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a classification method in which the class space (with k possible unique values) is separated into k classes using the discriminant function:

$$\hat{\delta}_k(x) = x \cdot \frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + \log(\hat{\pi}_k)$$

LDA uses the Bayesian rule to classify observations x to a class if it has the highest likelihood among all k classes. Since the discriminant function is a linear function of x, the decision boundary between any pair of classes is also a linear function of x. Linear discriminant analysis operates under the following assumptions and conditions:

1. Each feature is normally distributed (Gaussian)
2. Each class has identical covariance matrices

Using linear discriminant analysis, we want to develop our model to predict the loan_status response and compare this with our logistic regression model to see if this method yields

predictive improvements. We start by utilizing the same final predictors that were used in the logistic regression model and using our established training subset to create the LDA model using the lda() function from the package GGally. The training LDA model yields the following confusion matrix:

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 3662 | 3644 |
| Fully Paid | 30009 | 138487 |

Again, aggregating the correct predictions from each class (Fully Paid and Charged Off), we see the LDA training model predicts the loan status correctly 80.86% of the time and incorrectly 19.14% of the time.

Repeating the modeling process again now for our test set, we obtain our test model and calculate its predictions using the predict() function. Comparing the predicted values, to the true values, we obtain the following confusion matrix:

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 3478 | 3559 |
| Fully Paid | 30279 | 138486 |

The testing model produces accurate predictions 80.75% of the time and inaccurate predictions 19.25% of the time. We can see the model predictions plotted against the linear discriminant and the loan_amnt and grade respectively below:

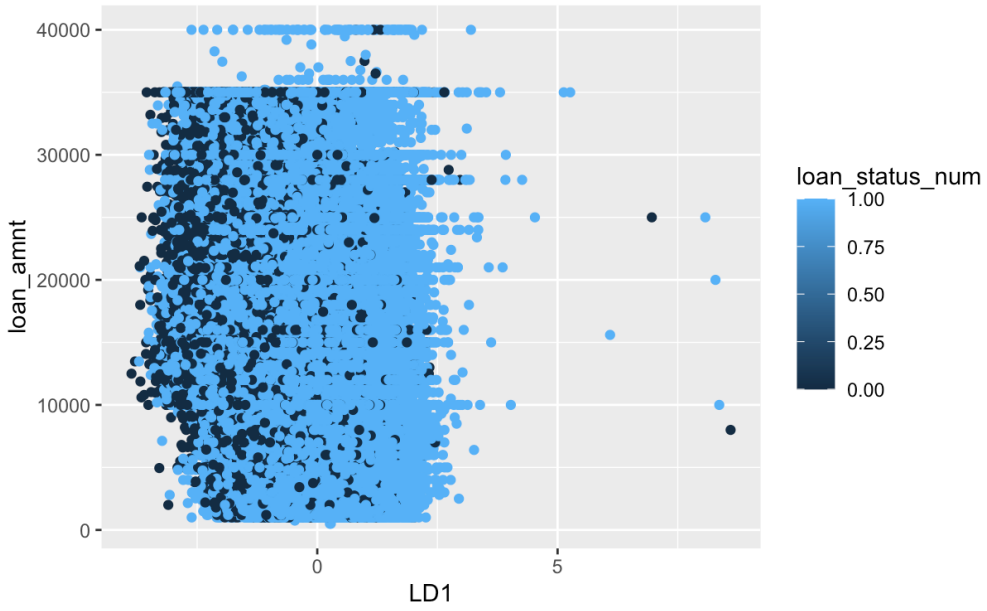


Figure 12

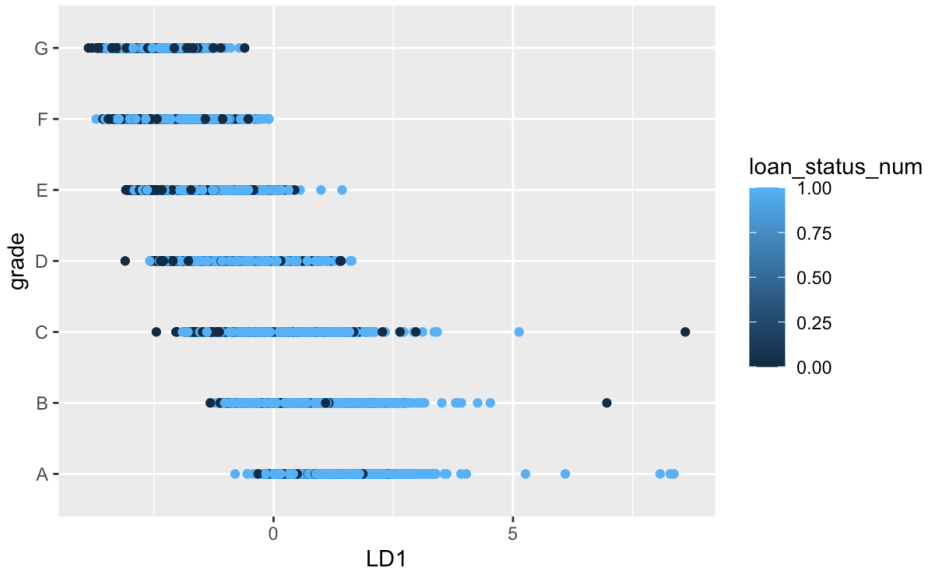


Figure 13

While this model's predictive power is similar to the value found for our logistic regression model, we will continue to explore other classification model methods to find the best predictive model.

## Quadratic Discriminant Analysis

Similar to linear discriminant analysis, quadratic discriminant analysis (QDA) uses a discriminant function to separate the class space into k classes. However, for QDA, the discriminant function is quadratic:

$$\begin{aligned}\delta_k(x) &= -\frac{1}{2}(x-\mu_k)^T\boldsymbol{\Sigma}_k^{-1}(x-\mu_k) - \frac{1}{2}\log|\boldsymbol{\Sigma}_k| + \log\pi_k \\ &= -\frac{1}{2}x^T\boldsymbol{\Sigma}_k^{-1}x + x^T\boldsymbol{\Sigma}_k^{-1}\mu_k - \frac{1}{2}\mu_k^T\boldsymbol{\Sigma}_k^{-1}\mu_k - \frac{1}{2}\log|\boldsymbol{\Sigma}_k| + \log\pi_k\end{aligned}$$

This means that the decision boundary between any pair of classes is also a quadratic function of x. The classification rule states that the predicted class for each observation will be to the class for which the discriminant function is maximized. The advantage that QDA provides over LDA is that the process is considered to be more flexible and tends to fit the data better given that the decision boundary is non-linear and can be "bent" around the class region more appropriately in cases with overlapping class observations. Quadratic discriminant analysis operates under the following assumptions and conditions:

1. Observations of each class are drawn from a normal (Gaussian) distribution
2. Each class has its own covariance matrix

Again, we want to develop our model to predict the loan_status response and compare with our previous models to see if there are any improvements to prediction power by implementing the QDA method. We start by utilizing the same final predictors that were used in the logistic regression model and LDA model and use our same training subset to develop a QDA train

model. Using the qda() function from the package GGally, we create our test model and its respective predictions, and compare the results to the "true" value in the following confusion matrix:

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 12324 | 22072 |
| Fully Paid | 21347 | 120059 |

Following the matrix main diagonal, we can see that our test QDA model made accurate predictions 75.30% of the time and incorrect predictions 24.70% of the time. This is slightly less accurate than the LDA models we formed; however, we only need to consider the performance of the test model.

Using our test set once again, we create the QDA test model and its predictions for loan_status and we observe the results of the comparison in the confusion matrix below:

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 11073 | 20356 |
| Fully Paid | 22684 | 121689 |

Following the matrix main diagonal, we can see that our test QDA model made accurate predictions 75.52% of the time and incorrect predictions 24.48% of the time. Compared to our previous models (logistic regression and linear discriminant analysis), this model seems to fall short in terms of predictive power. Why is that? Our natural intuition would be that since QDA produces quadratic decision boundaries, it should naturally better fit and classify the data as the class regions would be more fine-tuned. This rationale, however, is missing a crucial point. While the number of parameters estimated in LDA increases linearly with the mean, the number of parameters estimated in QDA increases quadratically which leads to a worse performance when the dimension is large. Given that we lose predictive accuracy with each that needs estimating, it is easy to see how our QDA model would not perform as well as LDA or logistic, given our 18 predictors.

## K-Nearest Neighbors

K-Nearest Neighbors is a non-parametric type of classifier that can be used on non-linear data. No assumptions are made about the decision boundary. Essentially using the k-nearest neighbors works how it sounds. A test statistic, x, is observed and the k, a chosen integer, nearest data points, neighbors, to our selected data point are chosen and their class/values are recorded. The dominant or average value is then assigned to our test statistic. When the decision boundary is highly non-linear, given a large sample size with a small number of predictors, the k-nearest neighbors method dominates LDA and Logistic regression. K-nearest neighbors could work well in our situation as we have a large sample size, 351,604, and 18 predictors. However, if 18 is not a small enough number of predictors, QDA may work better than k-nearest neighbors. The

downside to the k-nearest neighbors method is that unlike other methods it does not tell us which predictors are important.

When using the k-nearest neighbors method choosing the correct number, k, of neighboring data points is very important. If k is very small the decision boundary is overly flexible and finds patterns in the data that don't correspond to the Bayes decision boundary. This leads to low bias but high variance. However, as k gets bigger the decision boundary becomes less flexible and closer to linear. This then leads to the opposite problem of the small k in that we now have lower variance but a higher bias. So, choosing the correct level of flexibility to balance this problem is extremely important. When it came to our case, we looked at several different values of k before choosing our best value for building a predictive model was k = 9, as shown by the graph here.

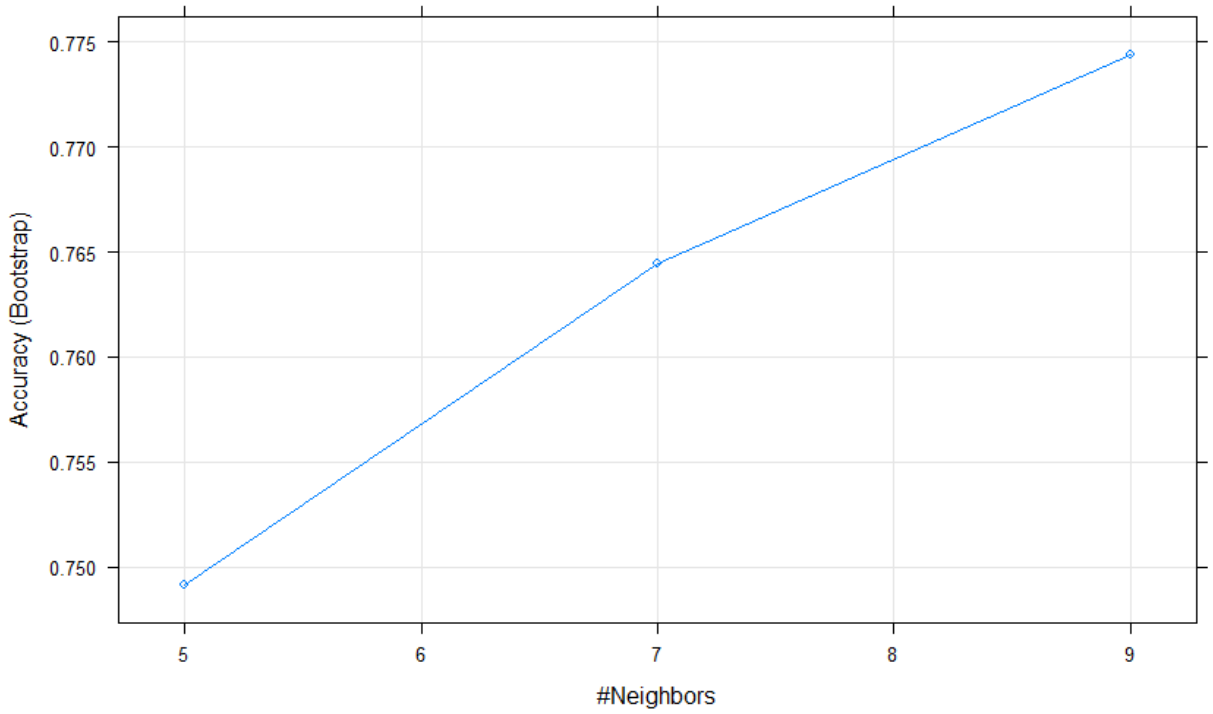


Figure 14

Using our k=9 with our 18 predictors we then used then tried to make predictions on whether or not the borrower was able to pay the loan back. These results were then compared to the actual results for each point in the test dataset and gave us the table shown here.

| | Truth | |
|---|---|---|
| Prediction | Charged Off | Fully Paid |
| Charged Off | 1486 | 3840 |
| Fully Paid | 32228 | 138248 |

This testing model here has an accuracy rate of 79.48% on getting the prediction made correct compared to the actual result. While this accuracy rate is good, it is not as good as the LDA or logistic regression approach. This leads us to believe that the data we are working with

is linear because, as explained above, k-nearest neighbors would dominate both LDA and logistic regression for highly non-linear data.

## Support Vector Machine

The objective of our support vector machine (SVM) is to draw a hydroplane that distinctly classifies our data points in the dataset. For our SVM we can only use numeric factors, therefore we are going to need to convert all of our categorical data into factors.

Data structure of dataset

```
'data.frame':   351604 obs. of  25 variables:
 $ loan_amnt           : num  10000 8000 15600 7200 24375 ...
 $ term                : chr  " 36 months" " 36 months" " 36 months" " 36 months" ...
 $ int_rate            : num  11.44 11.99 10.49 6.49 17.27 ...
 $ installment         : num  329 266 507 221 609 ...
 $ grade               : chr  "B" "B" "B" "A" ...
 $ sub_grade           : chr  "B4" "B5" "B3" "A2" ...
 $ emp_length          : chr  "10+ years" "4 years" "< 1 year" "6 years" ...
 $ home_ownership      : chr  "RENT" "MORTGAGE" "RENT" "RENT" ...
 $ annual_inc          : num  117000 65000 43057 54000 55000 ...
 $ verification_status : chr  "Not Verified" "Not Verified" "Source Verified" "Not Verified" ...
 $ issue_d             : chr  "Jan-2015" "Jan-2015" "Jan-2015" "Nov-2014" ...
 $ loan_status         : chr  "1" "1" "1" "1" ...
 $ purpose             : chr  "vacation" "debt_consolidation" "credit_card" "credit_card" ...
 $ dti                 : num  26.2 22.1 12.8 2.6 34 ...
 $ earliest_cr_line    : chr  "Jun-1990" "Jul-2004" "Aug-2007" "Sep-2006" ...
 $ open_acc            : num  16 17 13 6 13 8 8 11 13 13 ...
 $ pub_rec             : num  0 0 0 0 0 0 0 0 0 0 ...
 $ revol_bal           : num  36369 20131 11987 5472 24584 ...
 $ revol_util          : num  41.8 53.3 92.2 21.5 69.8 ...
 $ total_acc           : num  25 27 26 13 43 23 25 15 40 37 ...
 $ initial_list_status : chr  "w" "f" "f" "f" ...
 $ application_type    : chr  "INDIVIDUAL" "INDIVIDUAL" "INDIVIDUAL" "INDIVIDUAL" ...
 $ mort_acc            : num  0 3 0 0 1 4 3 0 3 1 ...
 $ pub_rec_bankruptcies: num  0 0 0 0 0 0 0 0 0 0 ...
 $ address             : chr  "0174 Michelle Gateway\nMendozaberg, OK 22690" "1076 Carney Fort Apt. 347\nLo
```

We then drop other redundant column such as subgrade, and address. We also drop installment and annual income since the column dti basically sums this up for us already.

Predictors we are using for svm

```
'data.frame':   351604 obs. of  19 variables:
 $ loan_amnt           : num  10000 8000 15600 7200 24375 ...
 $ term                : Factor w/ 2 levels " 36 months"," 60 months": 1 1 1 1 2 1 1 1
 $ int_rate            : num  11.44 11.99 10.49 6.49 17.27 ...
 $ grade               : Factor w/ 7 levels "A","B","C","D",..: 2 2 2 1 3 3 1 2 2 3 ..
 $ emp_length          : Factor w/ 10 levels "< 1 year","10+ years",..: 2 5 1 7 10 2 3
 $ home_ownership      : Factor w/ 6 levels "ANY","MORTGAGE",..: 6 2 6 6 2 2 2 6 6 2 .
 $ verification_status : Factor w/ 3 levels "Not Verified",..: 1 1 2 1 3 3 2 1 3 3 ...
 $ loan_status         : Factor w/ 2 levels "0","1": 2 2 2 2 1 2 2 2 2 2 ...
 $ purpose             : Factor w/ 14 levels "car","credit_card",..: 13 3 2 2 2 3 5 2
 $ dti                 : num  26.2 22.1 12.8 2.6 34 ...
 $ open_acc            : num  16 17 13 6 13 8 8 11 13 13 ...
 $ pub_rec             : num  0 0 0 0 0 0 0 0 0 0 ...
 $ revol_bal           : num  36369 20131 11987 5472 24584 ...
 $ revol_util          : num  41.8 53.3 92.2 21.5 69.8 ...
 $ total_acc           : num  25 27 26 13 43 23 25 15 40 37 ...
 $ initial_list_status : Factor w/ 2 levels "f","w": 2 1 1 1 1 1 1 1 2 1 ...
 $ application_type    : Factor w/ 3 levels "DIRECT_PAY","INDIVIDUAL",..: 2 2 2 2 2 2
 $ mort_acc            : num  0 3 0 0 1 4 3 0 3 1 ...
 $ pub_rec_bankruptcies: num  0 0 0 0 0 0 0 0 0 0 ...
> any(is.na(datasvm))
[1] FALSE
```

In the above image, we can view the structure of our new dataset and we see its all factors and numericals and there are no NAs. We end up using 19 of the predictors for our SVM.

SVMs don't perform well when we have large datasets and would take very long to run, so for the SVM, we are going to use the first 10000 observations.

First, we try a linear SVM, and use a cost of 100.

```
> svmlinear = svm(loan_status~., kernal = "linear", data=train, cost =100)
> summary(svmlinear)

Call:
svm(formula = loan_status ~ ., data = train, kernal = "linear", cost = 100)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  100

Number of Support Vectors:  3347

 ( 1264 2083 )


Number of Classes:  2

Levels:
 0 1
```


Figure 15

We end up with an accuracy of **79.4%** with a much higher accuracy in predicting fully paid as expected.
Next, we repeat this process with a radial SVM.

```
Call:
svm(formula = loan_status ~ ., data = train, kernal = "radial", gamma = 0.01, cost = 100)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  100

Number of Support Vectors:  3187

 ( 1276 1911 )


Number of Classes:  2

Levels:
 0 1
```

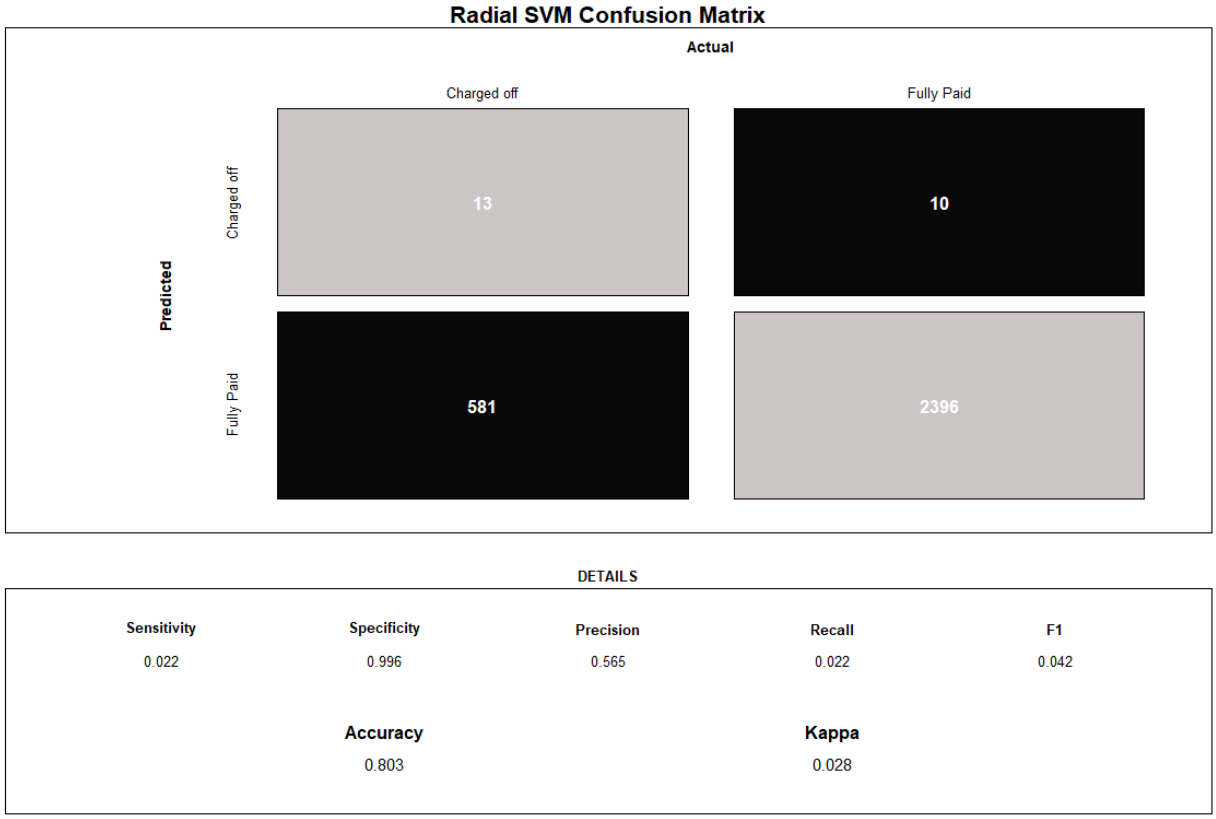


Figure 16

With the Radial SVM, we end up with an accuracy of **80.3%** which is moderately higher than our linear SVM. In both SVM cases we see a lot of Type I error which is expected with a dataset like this since banks don't predict many individuals to be charged off.

## Random Forest

For our random forest model, we returned the dataset back to its original size. We run the random forest with 100 trees and get the result below

```
Call:
 randomForest(formula = loan_status ~ ., data = train, ntree = 100,      mtry = 20)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 18

        OOB estimate of  error rate: 19.59%
Confusion matrix:
     0      1 class.error
0 3601  30053  0.89299935
1 4379 137769  0.03080592
```

```
> summary(rf.pred)
     0      1
  6535 169267
```

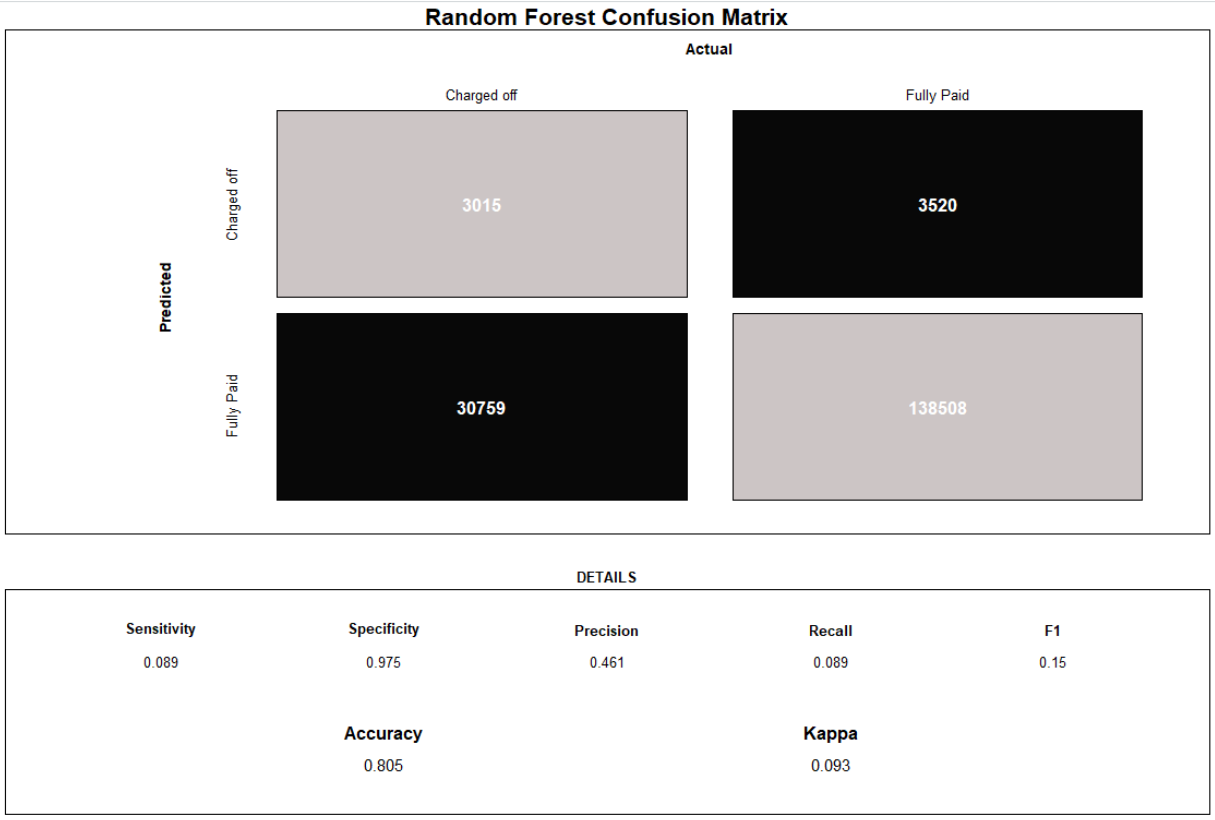


Figure 17

With our Random Forest model, we end up with an accuracy of **80.5%** which is also higher than our SVM.

# CONCLUSION

In our pursuit of a predictive model for loan_status, we used several classification methods: Logistic Regression, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbors, Classification Trees, and Support Vector Machines. Below is included the test model prediction accuracies per modeling method that we explored in the report:

| Model Method | Prediction Accuracy |
|---|---|
| Logistic Regression | 80.92% |
| Linear Discriminant Analysis | 80.75% |
| Quadratic Discriminant Analysis | 75.52% |
| K-Nearest Neighbors | 79.48% |
| Support Vector Machine | 80.30% |
| Random Forest | 80.50% |

Using our condition that the "best" model is the one with the highest prediction accuracy, we can see that the logistic regression model has the highest prediction accuracy, 80.92%. That is, to predict whether or not a loan recipient will pay back their loan or default for the LendingClub sub-dataset lending_club_loan_two, the best classification model is found using logistic regression for the predictors loan_amnt, term, int_rate, installment, grade, emp_length, annual_inc, verification_status, purpose, dti, open_acc, pub_rec, revol_bal, revol_util, total_acc, application_type, mort_acc, pub_rec_bankruptcies.

# APPENDIX

Group 13 Contributions:

***Arielle Thibeault*** – Introduction, Data, Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Conclusion, Appendix

***Brian Stella*** – K-Nearest Neighbors, Appendix

***Matthew Tartt*** – Support Vector Machines, Random Forest, Appendix

## Brain Stella Code:

```
datafinal = read.csv(file.choose())
sum(is.na(datafinal))
datafinal$pub_rec_bankruptcies[is.na(datafinal$pub_rec_bankruptcies)] <- 0
datafinal$mort_acc[is.na(datafinal$mort_acc)] <- 0
datafinal = datafinal[,-c(7,15)]
datafinal$emp_length[datafinal$emp_length != "< 1 year" &  datafinal$emp_length != "2 years"
&  datafinal$emp_length != "3 years" &  datafinal$emp_length != "4 years"
&  datafinal$emp_length != "5 years" &  datafinal$emp_length != "6 years"
```

```
&  datafinal$emp_length != "7 years" &  datafinal$emp_length != "8 years"
&  datafinal$emp_length != "9 years" &  datafinal$emp_length != "10+ years"] <- NA
datafinal = na.omit(datafinal)
datafinal[,12][datafinal[,12]=="Fully Paid"]<- 1
datafinal[,12][datafinal[,12]=="Charged Off"]<- 0
str(datafinal)
as.factor(datafinal$application_type)
datafinal$term <- ifelse(datafinal$term == "60 months", 1, 0)
datafinal$home_ownership <- ifelse(datafinal$home_ownership == "MORTGAGE", 1, 0)
datafinal$verification_status <- ifelse(datafinal$verification_status == "Source Verified", 1, 0)
datafinal$initial_list_status <- ifelse(datafinal$initial_list_status == "f", 1, 0)
datafinal$term <- ifelse(datafinal$term == "", 1, 0)
datafinal[,7][datafinal[,7]== "< 1 year"] <- 1
datafinal[,7][datafinal[,7]== "2 years"] <- 2
datafinal[,7][datafinal[,7]== "3 years"] <- 3
datafinal[,7][datafinal[,7]== "4 years"] <- 4
datafinal[,7][datafinal[,7]== "5 years"] <- 5
datafinal[,7][datafinal[,7]== "6 years"] <- 6
datafinal[,7][datafinal[,7]== "7 years"] <- 7
datafinal[,7][datafinal[,7]== "8 years"] <- 8
datafinal[,7][datafinal[,7]== "9 years"] <- 9
datafinal[,7][datafinal[,7]== "10+ years"] <- 10
as.numeric(datafinal$emp_length)
as.factor(datafinal$grade)
datafinal[,5][datafinal[,5]== "A"] <- 1
datafinal[,5][datafinal[,5]== "B"] <- 2
datafinal[,5][datafinal[,5]== "C"] <- 3
datafinal[,5][datafinal[,5]== "D"] <- 4
datafinal[,5][datafinal[,5]== "E"] <- 5
datafinal[,5][datafinal[,5]== "F"] <- 6
datafinal[,5][datafinal[,5]== "G"] <- 7
datafinal[,22][datafinal[,22]== "DIRECT_PAY"] <- 1
datafinal[,22][datafinal[,22]== "INDIVIDUAL"] <- 2
datafinal[,22][datafinal[,22]== "JOINT"] <- 3
str(datafinal)
#KNN
library(caret)
inTrain = createDataPartition(y=datafinal$loan_status, p = 1/2, list = FALSE)
Train=datafinal[inTrain,]
Test=datafinal[-inTrain,]
Direction2 = datafinal$loan_status[-inTrain]
library(class)
train.X <- Train[,-c(2,6,11:13,15,25)]
test.X <- Test[,-c(2,6,11:13,15,25)]
Direction2.train = Train[,12]
knn_train = train(train.X, Direction2.train, method = "knn", preProcess = c("center","scale"))
```

```
knn_train
plot(knn_train)
knn.pred <- knn(train.X,test.X,Direction2.train,k = 9)
table(knn.pred,Direction2)
sum(Train[,12]== 1)
sum(datafinal[,12] == 1)
```

## Matthew Tartt Code:

```
library(ggplot2)
library(ggthemes)
library(e1071)
library(dplyr)
library(caTools)
library(randomForest)
library(caret)

datafinal = read.csv(file.choose())
sum(is.na(datafinal))
datafinal$pub_rec_bankruptcies[is.na(datafinal$pub_rec_bankruptcies)] <- 0
datafinal$mort_acc[is.na(datafinal$mort_acc)] <- 0
datafinal = datafinal[,-c(7,15)]
datafinal$emp_length[datafinal$emp_length != "< 1 year" &  datafinal$emp_length != "2 years"
&  datafinal$emp_length != "3 years" &  datafinal$emp_length != "4 years" &  datafinal$emp_length !=
"5 years" &  datafinal$emp_length != "6 years" &  datafinal$emp_length != "7 years"
&  datafinal$emp_length != "8 years" &  datafinal$emp_length != "9 years" &  datafinal$emp_length !=
"10+ years"] <- NA
datafinal = na.omit(datafinal)
datafinal[,12][datafinal[,12]=="Fully Paid"]<- 1
datafinal[,12][datafinal[,12]=="Charged Off"]<- 0

##Feature engineering##
str(datafinal)

###Deleting data not related to SVM (subgrade, earliest creditline, address)
datasvm = datafinal[,-c(6,11,15,25)]

## show structure of dataset
str(datasvm)

### converting to factor for svm
datasvm$term = factor(datasvm$term)
datasvm$grade = factor(datasvm$grade)
datasvm$emp_length = factor(datasvm$emp_length)
datasvm$home_ownership = factor(datasvm$home_ownership)
datasvm$verification_status = factor(datasvm$verification_status)
datasvm$loan_status = factor(datasvm$loan_status)
datasvm$purpose = factor(datasvm$purpose)
```

```
datasvm$initial_list_status = factor(datasvm$initial_list_status)
datasvm$application_type = factor(datasvm$application_type)

####DTI == debt/payment, drop installment and annual income
datasvm1 = datasvm[,-c(4,8)]

####Feature engineering

### Check data structure again
str(datasvm1)
any(is.na(datasvm))

##selecting first 800 rows, svm bad with lasrge dataset
df=datasvm1[1:10000,]
split = sample(nrow(df),7000, replace = FALSE)

train=df[split,]
test=df[-split,]

##SVM
set.seed(69)
svmlinear = svm(loan_status~., kernal = "linear", data=train, cost =100)
summary(svmlinear)

svm1train = predict(svmlinear,train)
svm1test= predict(svmlinear,test)

table(svm1test, test$loan_status)

cm1= confusionMatrix(data=svm1test, reference = test$loan_status)
draw_confusion_matrix <- function(cm) {

  layout(matrix(c(1,1,2)))
  par(mar=c(2,2,2,2))
  plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')
  title('Linear SVM Confusion Matrix', cex.main=2)

  rect(150, 430, 240, 370, col='#CCC5C5')
  text(195, 435, 'Charged off', cex=1.2)
  rect(250, 430, 340, 370, col='#080808')
  text(295, 435, 'Fully Paid', cex=1.2)
  text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)
  text(245, 450, 'Actual', cex=1.3, font=2)
  rect(150, 305, 240, 365, col='#080808')
  rect(250, 305, 340, 365, col='#CCC5C5')
  text(140, 400, 'Charged off', cex=1.2, srt=90)
  text(140, 335, 'Fully Paid', cex=1.2, srt=90)
```

```
  res <- as.numeric(cm$table)
  text(195, 400, res[1], cex=1.6, font=2, col='white')
  text(195, 335, res[2], cex=1.6, font=2, col='white')
  text(295, 400, res[3], cex=1.6, font=2, col='white')
  text(295, 335, res[4], cex=1.6, font=2, col='white')

  plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', yaxt='n')
  text(10, 85, names(cm$byClass[1]), cex=1.2, font=2)
  text(10, 70, round(as.numeric(cm$byClass[1]), 3), cex=1.2)
  text(30, 85, names(cm$byClass[2]), cex=1.2, font=2)
  text(30, 70, round(as.numeric(cm$byClass[2]), 3), cex=1.2)
  text(50, 85, names(cm$byClass[5]), cex=1.2, font=2)
  text(50, 70, round(as.numeric(cm$byClass[5]), 3), cex=1.2)
  text(70, 85, names(cm$byClass[6]), cex=1.2, font=2)
  text(70, 70, round(as.numeric(cm$byClass[6]), 3), cex=1.2)
  text(90, 85, names(cm$byClass[7]), cex=1.2, font=2)
  text(90, 70, round(as.numeric(cm$byClass[7]), 3), cex=1.2)

  text(30, 35, names(cm$overall[1]), cex=1.5, font=2)
  text(30, 20, round(as.numeric(cm$overall[1]), 3), cex=1.4)
  text(70, 35, names(cm$overall[2]), cex=1.5, font=2)
  text(70, 20, round(as.numeric(cm$overall[2]), 3), cex=1.4)
}
draw_confusion_matrix(cm1)

####Radial kernal
set.seed(100)
svmradial = svm(loan_status~., kernal = "radial", data=train,gamma = .01, cost = 100)
summary(svmradial)

svmrtest= predict(svmradial,test)

table(svmrtest, test$loan_status)
cm2= confusionMatrix(data=svmrtest, reference = test$loan_status)
draw_confusion_matrix <- function(cm) {

  layout(matrix(c(1,1,2)))
  par(mar=c(2,2,2,2))
  plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')
  title('Radial SVM Confusion Matrix', cex.main=2)

  rect(150, 430, 240, 370, col='#CCC5C5')
  text(195, 435, 'Charged off', cex=1.2)
  rect(250, 430, 340, 370, col='#080808')
  text(295, 435, 'Fully Paid', cex=1.2)
  text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)
  text(245, 450, 'Actual', cex=1.3, font=2)
```

```
  rect(150, 305, 240, 365, col='#080808')
  rect(250, 305, 340, 365, col='#CCC5C5')
  text(140, 400, 'Charged off', cex=1.2, srt=90)
  text(140, 335, 'Fully Paid', cex=1.2, srt=90)

  res <- as.numeric(cm2$table)
  text(195, 400, res[1], cex=1.6, font=2, col='white')
  text(195, 335, res[2], cex=1.6, font=2, col='white')
  text(295, 400, res[3], cex=1.6, font=2, col='white')
  text(295, 335, res[4], cex=1.6, font=2, col='white')

  plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', yaxt='n')
  text(10, 85, names(cm$byClass[1]), cex=1.2, font=2)
  text(10, 70, round(as.numeric(cm$byClass[1]), 3), cex=1.2)
  text(30, 85, names(cm$byClass[2]), cex=1.2, font=2)
  text(30, 70, round(as.numeric(cm$byClass[2]), 3), cex=1.2)
  text(50, 85, names(cm$byClass[5]), cex=1.2, font=2)
  text(50, 70, round(as.numeric(cm$byClass[5]), 3), cex=1.2)
  text(70, 85, names(cm$byClass[6]), cex=1.2, font=2)
  text(70, 70, round(as.numeric(cm$byClass[6]), 3), cex=1.2)
  text(90, 85, names(cm$byClass[7]), cex=1.2, font=2)
  text(90, 70, round(as.numeric(cm$byClass[7]), 3), cex=1.2)

  text(30, 35, names(cm2$overall[1]), cex=1.5, font=2)
  text(30, 20, round(as.numeric(cm$overall[1]), 3), cex=1.4)
  text(70, 35, names(cm2$overall[2]), cex=1.5, font=2)
  text(70, 20, round(as.numeric(cm$overall[2]), 3), cex=1.4)
}
draw_confusion_matrix(cm2)
#### Random Forest#####

split2 = sample(nrow(datasvm1),175802, replace = FALSE)

train=datasvm1[split2,]
test=datasvm1[-split2,]

set.seed(300)
rf.model = randomForest(loan_status~., data = train, ntree = 100,
                        mtry = 20 )
rf.model

rf.pred = predict(rf.model, test)
summary(rf.pred)

table(rf.pred, test$loan_status)

cm3= confusionMatrix(data=rf.pred, reference = test$loan_status)
```

```
draw_confusion_matrix <- function(cm) {

 layout(matrix(c(1,1,2)))
 par(mar=c(2,2,2,2))
 plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')
 title('Random Forest Confusion Matrix', cex.main=2)

 rect(150, 430, 240, 370, col='#CCC5C5')
 text(195, 435, 'Charged off', cex=1.2)
 rect(250, 430, 340, 370, col='#080808')
 text(295, 435, 'Fully Paid', cex=1.2)
 text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)
 text(245, 450, 'Actual', cex=1.3, font=2)
 rect(150, 305, 240, 365, col='#080808')
 rect(250, 305, 340, 365, col='#CCC5C5')
 text(140, 400, 'Charged off', cex=1.2, srt=90)
 text(140, 335, 'Fully Paid', cex=1.2, srt=90)

 res <- as.numeric(cm3$table)
 text(195, 400, res[1], cex=1.6, font=2, col='white')
 text(195, 335, res[2], cex=1.6, font=2, col='white')
 text(295, 400, res[3], cex=1.6, font=2, col='white')
 text(295, 335, res[4], cex=1.6, font=2, col='white')

 plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', yaxt='n')
 text(10, 85, names(cm$byClass[1]), cex=1.2, font=2)
 text(10, 70, round(as.numeric(cm$byClass[1]), 3), cex=1.2)
 text(30, 85, names(cm$byClass[2]), cex=1.2, font=2)
 text(30, 70, round(as.numeric(cm$byClass[2]), 3), cex=1.2)
 text(50, 85, names(cm$byClass[5]), cex=1.2, font=2)
 text(50, 70, round(as.numeric(cm$byClass[5]), 3), cex=1.2)
 text(70, 85, names(cm$byClass[6]), cex=1.2, font=2)
 text(70, 70, round(as.numeric(cm$byClass[6]), 3), cex=1.2)
 text(90, 85, names(cm$byClass[7]), cex=1.2, font=2)
 text(90, 70, round(as.numeric(cm$byClass[7]), 3), cex=1.2)

 text(30, 35, names(cm2$overall[1]), cex=1.5, font=2)
 text(30, 20, round(as.numeric(cm$overall[1]), 3), cex=1.4)
 text(70, 35, names(cm2$overall[2]), cex=1.5, font=2)
 text(70, 20, round(as.numeric(cm$overall[2]), 3), cex=1.4)
}
draw_confusion_matrix(cm3)
```

## Arielle Thibeault Code:

```
---
title: "Peer-To-Peer Lending - Kaggle - STAT 508 Final Project"
```

Written by Arielle Thibeault
output: word_document
---
This code goes through the process of cleaning, examining, and performing analysis on our chosen dataset, lending club loan two. We will make a copy, called Lending, of the original data set to manipulate for the purposes of certain analysis, included below:
```{r}
install.packages("GGally")
library(GGally)
install.packages("ggeffects")
library(ggeffects)
install.packages("patchwork")
library(patchwork)
install.packages("car")
library(car)
install.packages("klaR")
library(klaR)
library(ggplot2)
library(MASS)
library(readr)
lending_club_loan_two <- read_csv("Desktop/STAT 508/Final
Materials/lending_club_loan_two.csv")
set.seed(17)
dim(lending_club_loan_two)
summary(lending_club_loan_two)
```
Important takeaways:
- 27 variables, 396030 observations
- ave loan amt = 14114
- ave int_rate = 13.64
- ave installment = 431.85
- ave annual_inc = 74203
---
In the next section, we begin our dataset cleaning process. Here we reassign NA values from pub_rec_bankruptcies and mort_acc, remove all NAs from emp_length, and also add a column for numeric mapping of loan_status for use in later analysis
```{r}
datafinal <- lending_club_loan_two
sum(is.na(datafinal))
datafinal$pub_rec_bankruptcies[is.na(datafinal$pub_rec_bankruptcies)] <- 0
datafinal$mort_acc[is.na(datafinal$mort_acc)] <- 0
datafinal <- datafinal[,-c(7,15,27)]
datafinal$emp_length[datafinal$emp_length != "< 1 year" &  datafinal$emp_length != "2 years"
&  datafinal$emp_length != "3 years" &  datafinal$emp_length != "4 years" &
datafinal$emp_length != "5 years" &  datafinal$emp_length != "6 years" &
```

```
datafinal$emp_length != "7 years" &  datafinal$emp_length != "8 years" &
datafinal$emp_length != "9 years" &  datafinal$emp_length != "10+ years"] <- NA
datafinal <- na.omit(datafinal)
datafinal$loan_status_num <- datafinal$loan_status
datafinal[,25][datafinal[,25] == "Fully Paid"] <- "1"
datafinal[,25][datafinal[,25] == "Charged Off"] <- "0"
datafinal$loan_status_num <- as.numeric(datafinal$loan_status_num)
```
Further in our data preparation, here we will reassign values for convenience for our analysis and create separated datasets for numeric and the original values. Here numdata includes a binary column for loan_status for use later.
```{r}
numdata <- datafinal[,-12]
numdata1 <- numdata #saving this for posterity
classdata <- datafinal[,-25]

#making term numeric
unique(numdata$term)
index1 <- which(numdata$term == "36 months")
index2 <- which(numdata$term == "60 months")
numdata$term[index1] <- 36
numdata$term[index2] <- 60
numdata$term <- as.numeric(numdata$term)

#making grade numeric, A-G from 1-7
unique(numdata$grade)
index3 <- which(numdata$grade == "G")
index4 <- which(numdata$grade == "F")
index5 <- which(numdata$grade == "E")
index6 <- which(numdata$grade == "D")
index7 <- which(numdata$grade == "C")
index8 <- which(numdata$grade == "B")
index9 <- which(numdata$grade == "A")
numdata$grade[index3] <- 7
numdata$grade[index4] <- 6
numdata$grade[index5] <- 5
numdata$grade[index6] <- 4
numdata$grade[index7] <- 3
numdata$grade[index8] <- 2
numdata$grade[index9] <- 1
numdata$grade <- as.numeric(numdata$grade)

#employment years
unique(numdata$emp_length)
index10 <- which(numdata$emp_length == "< 1 year")
index11 <- which(numdata$emp_length == "1 year")
```

```
index12 <- which(numdata$emp_length == "2 years")
index13 <- which(numdata$emp_length == "3 years")
index14 <- which(numdata$emp_length == "4 years")
index15 <- which(numdata$emp_length == "5 years")
index16 <- which(numdata$emp_length == "6 years")
index17 <- which(numdata$emp_length == "7 years")
index18 <- which(numdata$emp_length == "8 years")
index19 <- which(numdata$emp_length == "9 years")
index20 <- which(numdata$emp_length == "10+ years")
numdata$emp_length[index10] <- 0
numdata$emp_length[index11] <- 1
numdata$emp_length[index12] <- 2
numdata$emp_length[index13] <- 3
numdata$emp_length[index14] <- 4
numdata$emp_length[index15] <- 5
numdata$emp_length[index16] <- 6
numdata$emp_length[index17] <- 7
numdata$emp_length[index18] <- 8
numdata$emp_length[index19] <- 9
numdata$emp_length[index20] <- 10
numdata$emp_length <- as.numeric(numdata$emp_length)

#home ownership
unique(numdata$home_ownership)
index21 <- which(numdata$home_ownership =="OWN")
index22 <- which(numdata$home_ownership =="MORTGAGE")
index23 <- which(numdata$home_ownership =="RENT")
index24 <- which(numdata$home_ownership =="ANY")
index25 <- which(numdata$home_ownership =="OTHER")
index26 <- which(numdata$home_ownership =="NONE")
numdata$home_ownership[index21] <- 1
numdata$home_ownership[index22] <- 2
numdata$home_ownership[index23] <- 3
numdata$home_ownership[index24] <- 4
numdata$home_ownership[index25] <- 5
numdata$home_ownership[index26] <- 6
numdata$home_ownership <- as.numeric(numdata$home_ownership)
```
Below are some distribution visualizations
```{r}
qplot(lending_club_loan_two$loan_status, main = "Distribution of Loan Status", ylab =
"Frequency", xlab = "Loan Status Spread") + geom_bar()

qplot(lending_club_loan_two$term, main = "Distribution of Given Loan Terms", ylab =
"Frequency", xlab = "Loan Term Spread") + geom_bar()
```

```
qplot(lending_club_loan_two$home_ownership, main = "Type of Home Ownership", ylab = "Frequency", xlab = "Homew Ownership Type") + geom_bar()

qplot(lending_club_loan_two$purpose, main = "Reason for Loan", ylab = "Frequency", xlab = "Loan Use") + geom_bar() + theme(axis.text.x = element_text(angle = 45, vjust = 0.5))

qplot(lending_club_loan_two$int_rate, geom= "histogram", main = "Distribution of Interest Rates", ylab = "Frequency", xlab = "Interest Rates") +
  geom_histogram(aes(fill = ..count..)) +
    scale_fill_gradient("Count", low = "black", high = "gray")

qplot(lending_club_loan_two$installment, geom= "histogram", main = "Distribution of Payment Installments", ylab = "Frequency", xlab = "Installments") +
  geom_histogram(aes(fill = ..count..)) +
    scale_fill_gradient("Count", low = "black", high = "gray")

qplot(Lending$loan_amnt, geom= "histogram", main = "Distribution of the Lending Loan Amount", ylab = "Frequency", xlab = "Listed Loan Amount") +
  geom_histogram(aes(fill = ..count..)) +
  scale_fill_gradient("Count", low = "black", high = "seagreen2")

quantile(Lending$annual_inc, probs = c(0.05, 0.95))
qplot(Lending$annual_inc, geom= "histogram", main = "Distribution of the Loan Recipient's Annual Income", ylab = "Frequency", xlab = "Annual Income", xlim = c(28000,150000)) +
  geom_histogram(aes(fill = ..count..)) +
  scale_fill_gradient("Count", low = "black", high = "orange")

#ggplot(lending_club_loan_two, aes(x = grade, y = loan_status)) +
#  geom_col(position = "dodge", colour = 'red')
```
Correlation Plot
```{r}
ggcorr(lending_club_loan_two, low = "springgreen4", mid = "white", high = "hotpink4") +
ggtitle(label = "Correlation Plot")

ggcorr(numdata, low = "springgreen4", mid = "white", high = "hotpink4") + ggtitle(label = "Correlation Plot")
```
Logistic Regression for Classification
- Check the predictors
```{r}
Loan.fit <-
glm(loan_status_num~loan_amnt+term+int_rate+installment+grade+sub_grade+emp_length+home_ownership+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+initial_list_status+application_type+mort_acc+pub_rec_bankruptcies, data = numdata, family = "binomial")
```

```
summary(Loan.fit)
```
From the summary, it appears that only the variable initial_list_status is not significant for alpha = 0.05, as all other coefficient estimate p-values are much smaller than alpha. We can check our hypothesis by comparing to a Chi-Square Statistic with the same degrees of freedom
```{r}
1-pchisq(343716-311851, 351603-351536)
```
Additionally, to dissolve any issues of collinearity, we will remove the sub_grade predictor as several of the sub_grade values are not significant predictors of loan_status. (we can easily check that the removal does not significantly affect the prediction power of the model)

Now we refit without initial_list_status and sub_grade
```{r}
Loan.fit <-
glm(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+home_ownership+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_type+mort_acc+pub_rec_bankruptcies, data = numdata, family = "binomial")
summary(Loan.fit)
```
Let's repeat this with the original data (not transformed) and see if we get any different results:
```{r}
Loan.fita <-
glm(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+home_ownership+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+initial_list_status+application_type+mort_acc+pub_rec_bankruptcies, data = numdata1, family = "binomial")
summary(Loan.fita)
```
From this output, we indeed see a different outcome: here, home_ownership and initial_list_status are not significant, while all others are significant up to 0.05 alpha level

Let's fit another model with these removed:
```{r}
Loan.fita <-
glm(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_type+mort_acc+pub_rec_bankruptcies, data = numdata1, family = "binomial")
summary(Loan.fita)
coef(Loan.fita)
```
Now let's check our hypothesis by comparing to a Chi-Square Statistic with the same degrees of freedom
```{r}
1-pchisq(343716-313093, 351603-351558)
```

Since the p-value is effectively zero and definitely less than 0.05, we reject the null hypothesis and conclude that there is a statistically significant relationship between loan_status_num, loan_amnt, term, int_rate, installment, grade, emp_length, annual_inc, verification_status, purpose, dti, open_acc, pub_rec, revol_bal, revol_util, total_acc, application_type, mort_acc, and pub_rec_bankruptcies.

Given that this is a mirror of the original dataset and our hypothesis passed, we will move forward with these predictors.

Let's fit this on a training dataset. Given the size of our set, we can allocate a large proportion to the training set. We choose 50%.

```{r}
n <- dim(numdata1)
train <- sample(1:n, n/2)
test <- (1:n)[-train]

Loan.fita.train <-
glm(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_type+mort_acc+pub_rec_bankruptcies, data = numdata1[train,], family = "binomial")
summary(Loan.fita.train)
coef(Loan.fita.train)
```

Below we compute the confusion matrix, or the error matrix, displaying the performance

```{r}
Loan.probs <- predict(Loan.fita.train, newdata = numdata1[train,], type = "response")
length(Loan.probs)
Loan.pred <- rep("Charged Off", 175802)
Loan.pred[Loan.probs > .5] = "Fully Paid"
table(predict = Loan.pred, truth = numdata1[train,]$loan_status_num)
```

From the table we see that our training model made correct predictions (1707+140773)/175802 = 0.81045721891 or 81.05% of the time. This is considerable.

Now we fit our model to the test set and evaluate performance as this is really the metric we care about for measuring predictive power.

```{r}
Loan.fita.test <-
glm(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_type+mort_acc+pub_rec_bankruptcies, data = numdata1[test,], family = "binomial")
summary(Loan.fita.test)
coef(Loan.fita.test)

Loan.probs <- predict(Loan.fita.test, newdata = numdata1[test,], type = "response")
length(Loan.probs)
```

```
Loan.pred <- rep("Charged Off", 175802)
Loan.pred[Loan.probs > .5] = "Fully Paid"
table(predict = Loan.pred, truth = numdata1[test,]$loan_status_num)

ggplot(numdata1[test,],aes(x=loan_status_num,y=loan_amnt,color=factor(grade)))+geom_point(
)+stat_smooth(method="lm",se=FALSE)

plot(ggpredict(Loan.fita.test,"loan_amnt"))
plot(ggpredict(Loan.fita.test,"grade"))
```
From the table we see that our test model made correct predictions (1647+140607)/175802 = 0.80917168177 or 80.92% of the time. This is approximately the same to that of our training set and is still fairly good.

Linear Discriminant Analysis
```{r}
Loan.lda.train <-
lda(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+ver
ification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_ty
pe+mort_acc+pub_rec_bankruptcies, data = numdata1[train,], family = binomial)
summary(Loan.lda.train)
Loan.lda.pred <- predict(Loan.lda.train, newdata = numdata1[train,])
table(Loan.lda.pred$class, numdata1[train,]$loan_status_num)
mean(Loan.lda.pred$class==numdata1[train,]$loan_status_num)
```
The LDA training model predicts the loan status correctly 0.8085744 or 80.86% of the time.
Let's now check the testing model:
```{r}
Loan.lda.test <-
lda(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+ver
ification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_ty
pe+mort_acc+pub_rec_bankruptcies, data = numdata1[test,], family = binomial)
summary(Loan.lda.test)
Loan.lda.test.pred <- predict(Loan.lda.test, newdata = numdata1[test,])
table(Loan.lda.test.pred$class, numdata1[test,]$loan_status_num)
mean(Loan.lda.test.pred$class==numdata1[test,]$loan_status_num)

lda_plot <- cbind(numdata1[test,], Loan.lda.test.pred$x)
ggplot(lda_plot, aes(LD1,loan_amnt)) + geom_point(aes(color = loan_status_num))
ggplot(lda_plot, aes(LD1,grade)) + geom_point(aes(color = loan_status_num))
```
The LDA test model predicts the loan status correctly 0.8075221 or 80.75% of the time.

Quadratic Discriminant Analysis
Training model first:

```{r}
Loan.qda.train <-
qda(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_type+mort_acc+pub_rec_bankruptcies, data = numdata1[train,])
summary(Loan.qda.train)
Loan.qda.train.pred <- predict(Loan.qda.train, numdata1[train,])$class
table(Loan.qda.train.pred, numdata1[train,]$loan_status_num)
mean(Loan.qda.train.pred==numdata1[train,]$loan_status_num)
```
The QDA train model predicts the loan status correctly 0.7530233 or 75.30% of the time.

Now onto the test QDA model:
```{r}
Loan.qda.test <-
qda(loan_status_num~loan_amnt+term+int_rate+installment+grade+emp_length+annual_inc+verification_status+purpose+dti+open_acc+pub_rec+revol_bal+revol_util+total_acc+application_type+mort_acc+pub_rec_bankruptcies, data = numdata1[test,])
summary(Loan.qda.test)
Loan.qda.test.pred <- predict(Loan.qda.test, numdata1[test,])$class
table(Loan.qda.test.pred, numdata1[test,]$loan_status_num)
mean(Loan.qda.test.pred==numdata1[test,]$loan_status_num)
```
The QDA test model predicts the loan status correctly 0.7551791 or 75.52% of the time.