Module 6: Final Project: Loan Data Analysis

ALY6015 Introduction to Analytics
Northeastern University

GROUP 3: Manas Babbar
Snehal Pawar
Sudhamshu Vidyananda

Professor: Richard He
Date: 21st May 2022

INTRODUCTION

Dataset description

The dataset we have chosen for our final project belongs to Finance Sector where our all three group members interest involves. We have chosen our Dataset (Bank Loan Prediction) from Kaggle. This dataset is having 19 unique features and total of 80847 rows of data (without cleaning). The main variables that we have used throughout the analysis are annual_income, loan_status, cutomer_id, years_of_credit_history, credit_score, number_of_open_accounts etc. Credit score cards are a common risk control method in the financial industry. It uses personal information and data submitted by credit card applicants to predict the probability of future defaults and credit card borrowings. The bank can decide whether to issue a credit card to the applicant. Credit scores can objectively quantify the magnitude of risk.

Credit score cards are based on historical data. Once encountering large economic fluctuations. Past models may lose their original predictive power. Logistic model is a common method for credit scoring. Because Logistic is suitable for binary classification tasks and can calculate the coefficients of each feature. To facilitate understanding and operation, the score card will multiply the logistic regression coefficient by a certain value (such as 100) and round it.

At present, with the development of machine learning algorithms more predictive methods such as Boosting, Random Forest, and Support Vector Machines have been introduced into credit card scoring. However, these methods often do not have good transparency. It may be difficult to provide customers and regulators with a reason for rejection or acceptance.

This dataset can be useful to bank officials to predict if the customer is a good customer, or a bad customer based on their credit scores. They can also predict if loan amount can be approved or not and if the customer is eligible for loan amount based on his annual income/monthly debt etc.

Data Preparation

As part of data preparation, we have omitted the null values and normalized the dataset.

Data normalization involves scaling the attribute values to make them lie numerically in the same interval/scale, and thus have the same importance.

An efficient way of Normalizing values is through the Min-Max Scaling method. With Min-Max Scaling, we scale the data values between a range of 0 to 1 only. Due to this, the effect of outliers on the data values suppresses to a certain extent. Moreover, it helps us have a smaller value of the standard deviation of the data scale.

We have used ‘caret‘ library to pre-process and scale the data. The preProcess() function enables us to scale the value to a range of 0 to 1 using method = c('range') as an argument.

The predict() method applies the actions of the preProcess() function on the entire data frame as shown below.

```{r}
#Normalising the Data set
process= preProcess(credit_train,method = c("range"))
credit_train= predict(process,credit_train)
datatable(head(credit_train))
```

Purpose	Monthly Debt	Years of Credit History	Months since last delinquent	Number of Open Accounts	Number of Credit Problems	Current Credit Balance	Maximum Open Credit	Bankruptcies
Home Improvements	0.0432544710680762	0.274193548387097	NA	0.106382978723404	0.142857142857143	0.0307373378035984	0.000658910150159025	1
Debt Consolidation	0.276178678820659	0.352822580645161	8	0.723404255319149	0	0.0309779131369488	0.0013451603931241	0
Debt Consolidation	0.242208332545341	0.227822580645161	29	0.361702127659574	0.142857142857143	0.0401402503007192	0.00118595478908684	0
Debt Consolidation	0.0725110476514679	0.169354838709677	NA	0.170212765957447	0	0.0345276789598956	0.0006118128401598	0
Debt Consolidation	0.171199198139077	0.0504032258064516	NA	0.297872340425532	0	0.0341412228393008	0.000675397687042062	0

Omission of null values will help in cleaning up unwanted data from the dataset.

```
# Remove null  & NA values(normalization)
inputdataset <- na.omit(inputdataset)
inputdataset
```

Description: df [38,010 × 19]

Current_loan_amount <int>	Term <chr>	credit_score <int>	annual_income <int>
99999999	Short Term	741	2231892
217646	Short Term	730	1184194
548746	Short Term	678	2559110
99999999	Short Term	728	714628
99999999	Short Term	740	776188
234124	Short Term	727	693234
666204	Long Term	723	1821967
317108	Long Term	687	1133274
465410	Long Term	688	1722654
99999999	Short Term	746	1749748

1-10 of 38,010 rows | 5-8 of 19 columns Previous 1 2 3 4 5 6 ... 100 Next

Exploratory Data Analysis

Here, using the describe function of psych library, we have retrieved the descriptive statistics of the dataset.

As we see in the below screenshot, we derive the mean, median, kurtosis, skew, min, max, range values of the dataset.

Loan.Status variable is negatively skewed with a value of -1.43.
Negative excess values of kurtosis indicate that a distribution is flat and has thin tails.
Loan.ID and **Customer.ID** have the least negative kurtosis value.

Monthly.Debt	Years.of.Credit.History	Months.since.last.delinquent	Number.of.Open.Accounts	Number.of.Credit.Problems	Current.Credit.Balance	Maximum.Open.Credit
0	3.6	0	0	0	0	0
38.501	0.022	0.102	0.016	0.002	1189.557	26514.393
2.214	1.072	0.434	1.179	4.823	14.154	132.635
22.191	1.74	-0.746	3.042	48.009	697.449	20393.412
9969.714	5.93	25.204	4.448	0	170509.378	342333.823
12174.993	7.015	21.998	5.01	0.483	376170.935	8384503.472
16220.3	16.9	32	10	0	209817	467874
17090.614	17.529	33.632	10.663	0.05	239638.114	526283.538
18472.412	18.199	34.901	11.129	0.168	294637.382	760798.382
11	12	13	14	15	16	17
435843.28	70.5	176	76	15	32878968	1539737892
435843.28	66.9	176	76	15	32878968	1539737892
100000	100000	46859	100000	100000	100000	99998

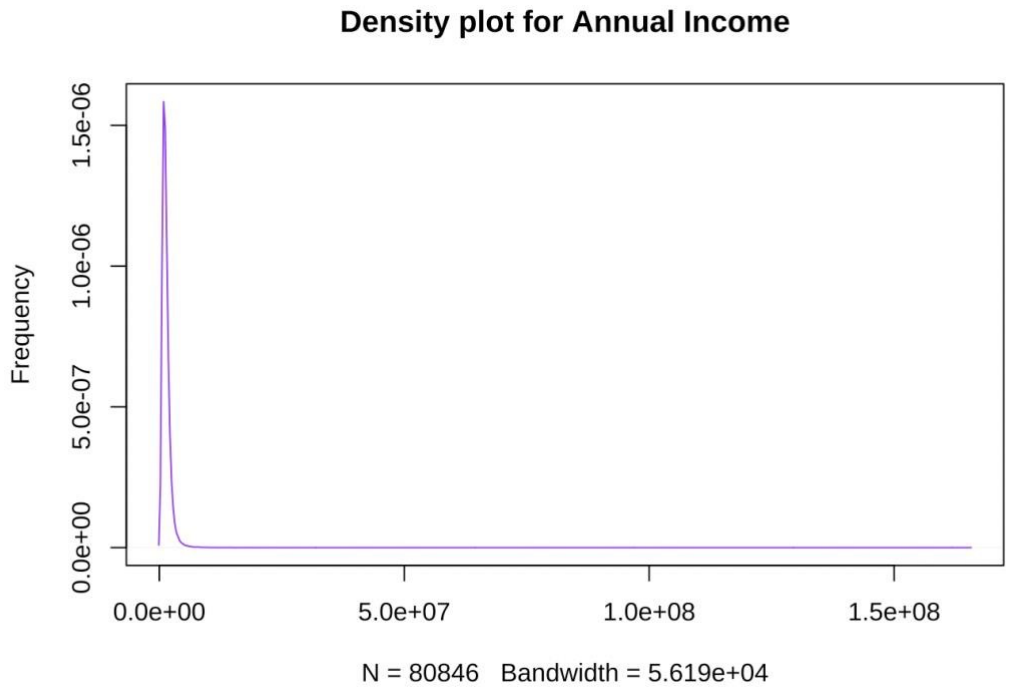
Maximum.Open.Credit	Bankruptcies	Tax.Liens
0	0	0
26514.393	0.001	0.001
132.635	3.506	15.5
20393.412	18.526	402.038
342333.823	0	0
8384503.472	0.351	0.258
467874	0	0
526283.538	0.013	0
760798.382	0.118	0.029
17	18	19
1539737892	7	15
1539737892	7	15
99998	99796	99990

Show 50 entries Search:

	Loan.ID*	Customer.ID*	Loan.Status*	Current.Loan.Amount	Term*	Credit.Score	Annual.Income	Years.in.current.job*	Home.Ownership*	Purpose*
min	1	1	1	10802	1	585	76627	1	1	1
se	75.119	75.089	0.001	100509.651	0.001	5.189	3803.129	0.01	0.003	0.007
skew	-0.002	0	-1.43	2.416	-1.097	3.863	46.887	0.714	0.017	2.122
kurtosis	-1.202	-1.202	0.561	3.837	-0.387	12.971	6623.594	-0.582	-1.539	4.436
mad	30573.436	30569.729	0	218926.646	0	25.204	564430.268	2.965	1.483	0
sd	23815.831	23806.15	0.436	31783942.546	0.463	1475.404	1081360.196	3.11	0.974	2.235
median	40826.5	40804.5	3	312246	3	724	1174162	5	4	5
trimmed	40808.764	40806.862	2.837	2191409.026	2.773	722.465	1249929.045	5.826	3.914	5.249
mean	40806.174	40807.707	2.765	11760447.389	2.713	1076.456	1378276.56	6.092	3.919	5.775
vars	1	2	3	4	5	6	7	8	9	10
max	82000	82000	3	99999999	3	7510	165557393	13	5	17
range	81999	81999	2	99989197	2	6925	165480766	12	4	16
n	100514	100514	100514	100000	100514	80846	80846	100514	100514	100514

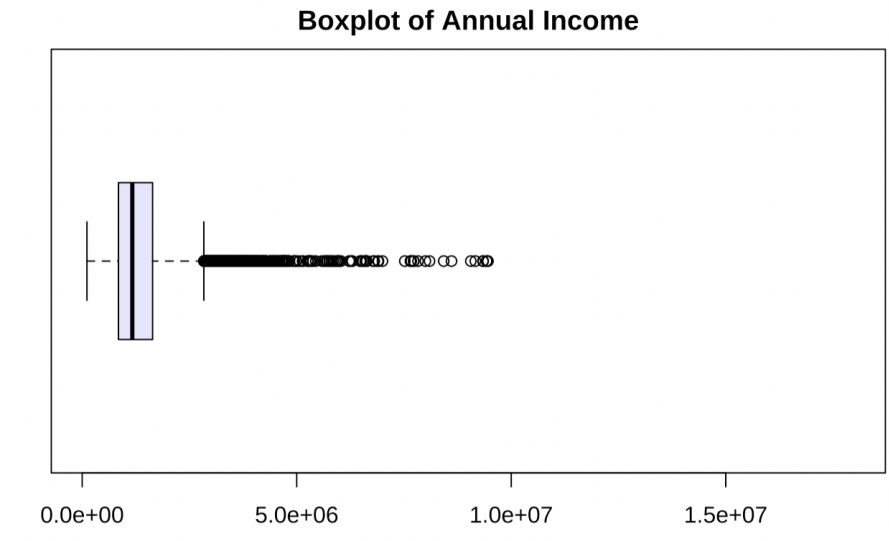
Showing 1 to 13 of 13 entries Previous 1 Next

Density plot for Annual Income



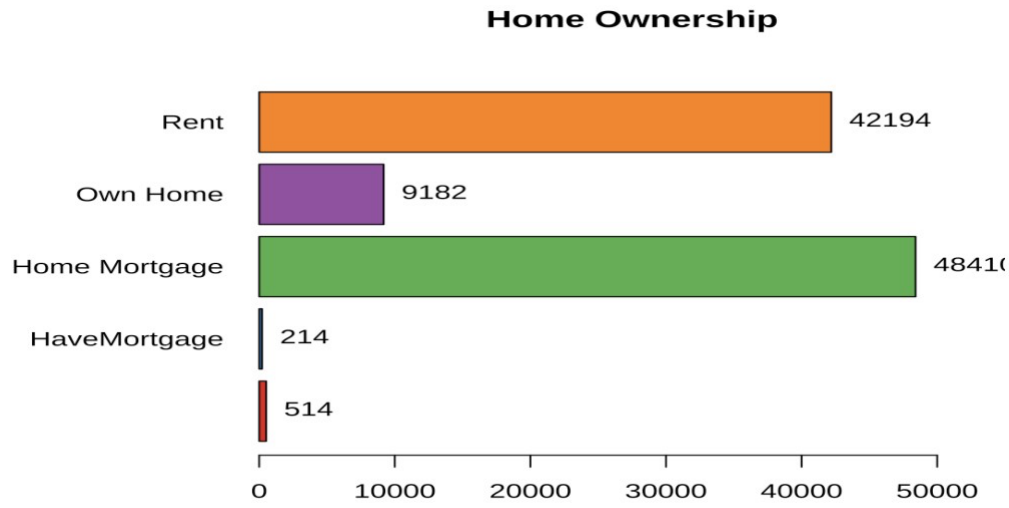
Density plot is used to show the probability density function of the variable of interest. Here, we are determining the density plot for Annual Income variable. As seen from the graph, the plot is touching it peak in the range $1110000 to $ 1200000.

Boxplot of Annual Income



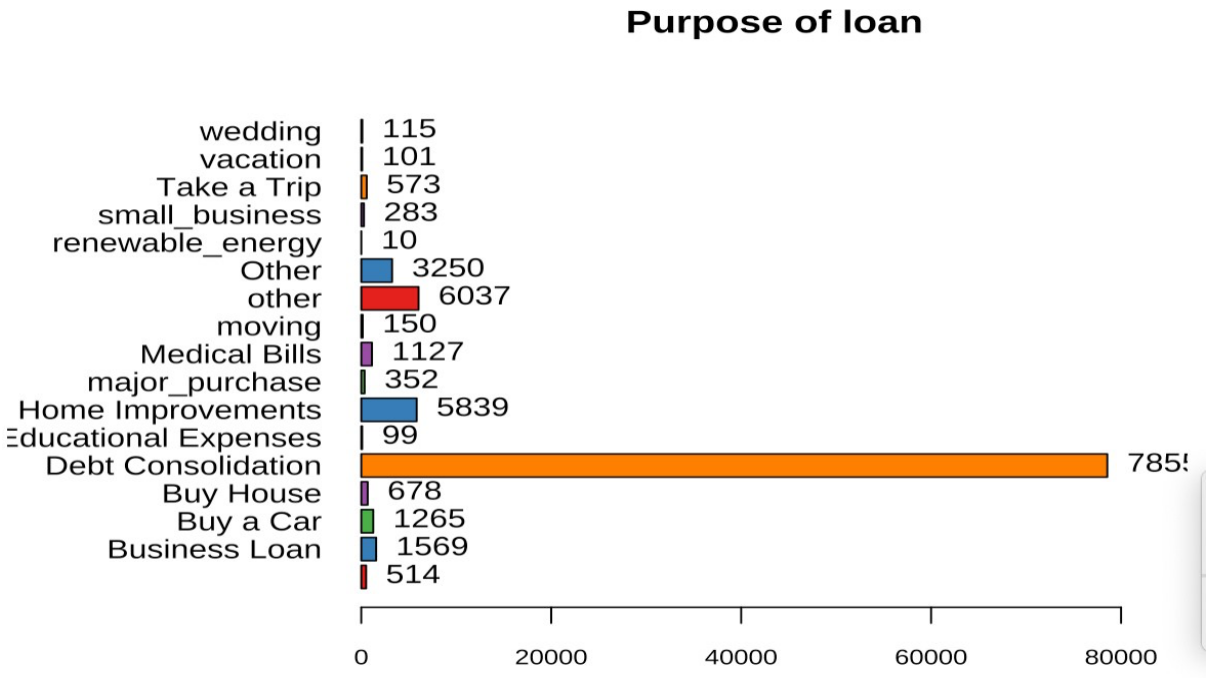
The boxplot() function takes in any number of numeric vectors, drawing a boxplot for each vector. The median value from the plot is 1174162. We see there are plenty of outliers at the higher extreme even though we have removed the outliers and median is more dispersed even after data normalization.

Bar Plot for Home Ownership



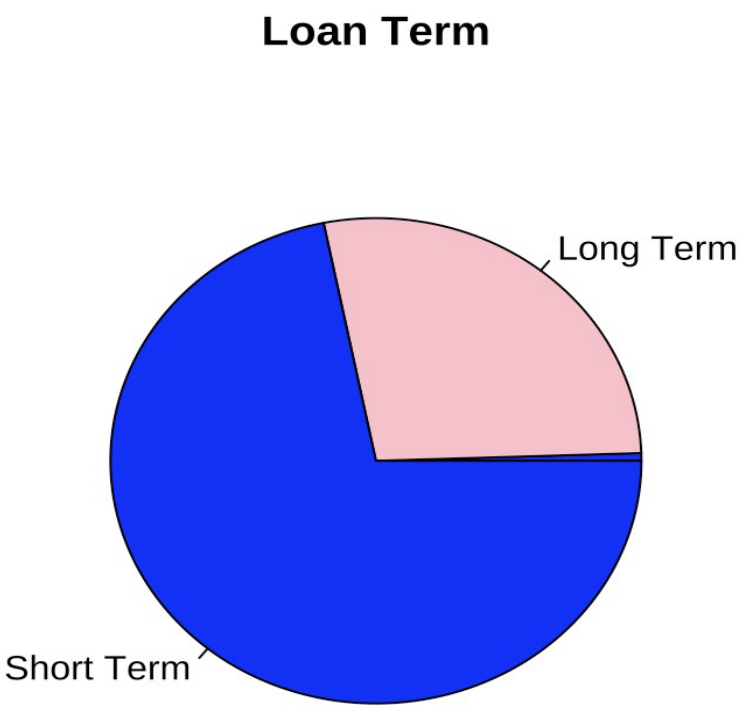
Here, number of loan taken is highest for home mortgage followed by Rent. HaveMortage is having the least number of loan taken from the bank.

Bar plot to check the purpose type of loan



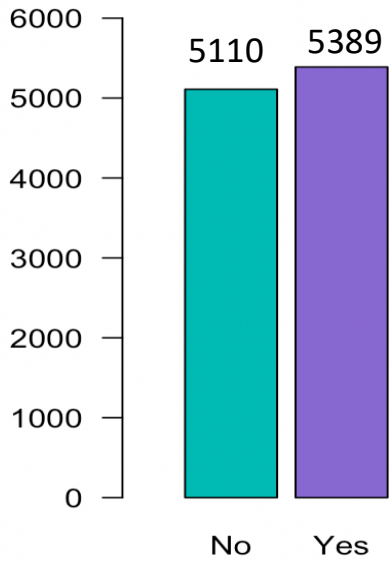
As per the graph, we see Debt Consolidation is having the highest count, followed by other category. Renewable_energy is having the lowest count of 10. So we can infer that, loan is taken highest for debt consolidation.

Pie Chart for Loan Term



Here, we see the percentage distribution for short term and long-term loans. Clearly Short-term loans is exceeding long terms loans and people prefer the loan to be for short term because of the interest rate tagged to it.

Bar graph to show the loan approval status



Problem Statements

1. Build a method or a learning model to predict if an applicant is 'good' or 'bad' client.
2. Build a model to predict the value of loan credit the applicant is eligible for.
3. Build a model to predict if the person requested for the loan amount should be approved or not.

These questions are useful to a bank official to predict decisions regarding loan sanctions to a customer.

METHODS

The Methods that we'll be using in our project to Analyze the data and make a model which is appropriate for our research question.

- Logistic Regression
- Multiple regression model
- Ridge Regression
- Lasso Regression

LOGISTIC REGRESSION

Research Question: Model to predict if the person requested for the loan amount should be approved or not.

So in order to answer the above question we used Logistic Regression as it is used when the dependent variable(target) is categorical which was in our case `Loan Approval`. As Loan Approval was a binomial variable in terms of "Yes" or "No" we used binary logistic Regression.To understand how logistic regression can be seen as GLM, we can elaborate this approach as follows:

Logistic regression measures the relationship between the dependent variable and one or more independent variables(features) by estimating probabilities using the underlying **logit function.** In statistics, the logit function or the log-odds is the logarithm of the odds. Given a probability p, the corresponding odds are calculated as p / (1 – p). For The logit function is the logarithm of the odds: logit(x) = log (x / (1 – x)). The **Odds** describes the ratio of success to ratio of failure. The **Odds ratio** is the ratio of odds and is calculated as the ratio of odds for each group. The inverse of the logit function is the **sigmoid function**. The formula for the sigmoid function is $\sigma(x) = 1/ (1 + exp(-x))$. The sigmoid function maps probabilities to the range [0, 1] – and this makes logistic regression as a classifier. Thus, many models have data generating processes that can be linearized by considering the inverse

E.g., $Y = e^{a+BX} \rightarrow log(Y) = a + BX$

The Logitistic Function

$$p = \frac{e^{(a+BX)}}{1 + e^{(a+BX)}}$$

$$logit(p) = a + BX$$

The logit and the sigmoid functions are useful in analysis because their gradients are simple to calculate. Many optimization and machine learning techniques make use of gradients (for example in neural networks). The biggest drawback of the sigmoid function for many analytics practitioners is the so-called “vanishing gradient” problem.

MULTIPLE REGRESSION

Research question : learning model to predict if an applicant is 'good' or 'bad' client.

For analyzing this research question, we used Multiple regression too solve our problem statement as we wanted to find the relation between the credit score and the other variables to know how to predict that weather the client is good or bad.

Regression models are used to describe relationships between variables by fitting a line to the observed data. Regression allows you to estimate how a dependent variables changes as the independent variable(s) change.

Multiple linear regression is used to estimate the relationship between **two or more independent variables** and **one dependent variable**. You can use multiple linear regression when you want to know:

1. How strong the relationship is between two or more independent variables and one dependent variable (e.g. how rainfall, temperature, and amount of fertilizer added affect crop growth).
2. The value of the dependent variable at a certain value of the independent variables (e.g. the expected yield of a crop at certain levels of rainfall, temperature, and fertilizer addition).

In multiple linear regression, it is possible that some of the independent variables are actually correlated with one another, so it is important to check these before developing the regression model. If two independent variables are too highly correlated (r2 > ~0.6), then only one of them should be used in the regression model.

The line of best fit through the data points is a straight line, rather than a curve or some sort of grouping factor.

Multiple linear regression formula

The formula for a multiple linear regression is:

$$y = \beta_0 + \beta_1 X_1 + ... + \beta_n X_n + \varepsilon$$

- **y** = the predicted value of the dependent variable
- $\mathbf{B_0}$ = the y-intercept (value of y when all other parameters are set to 0)
- $\mathbf{B_1X_1}$= the regression coefficient (B_1) of the first independent variable ($\mathbf{X_1}$) (a.k.a. the effect that increasing the value of the independent variable has on the predicted **y** value)
- **…** = do the same for however many independent variables you are testing
- $\mathbf{B_nX_n}$ = the regression coefficient of the last independent variable
- **e** = model error (a.k.a. how much variation there is in our estimate of **y**)

REGULARISED METHODS RIDGE AND LASSO

Research Question: Model for predicting `Maximum Loan Credit` amount for which our customer is eligible for.

Too build above model we used tried too used more regular models as from our previous tried models we were able to understand that our models were very much underfitting and where these models help. As these models use all the categorical and continuous variables into considerations for prediction.

Bias vs Variance Trade off

For better knowledge too understand this model Let us consider that we have a very accurate model, this model has a low error in predictions and it's not from the target (which is represented by bull's eye). This model has low bias and variance. Now, if the predictions are scattered here and there then that is the symbol of high variance, also if the predictions are far from the target, then that is the symbol of high bias. Sometimes we need to choose between low variance and low bias. There is an approach that prefers some bias over high variance, this approach is called Regularization.

Ridge Regression:

In Ridge regression, we add a penalty term which is equal to the square of the coefficient. The *L2* term is equal to the square of the magnitude of the coefficients. We also add a Coefficient to control that penalty term. In this case if lambda is zero then the equation is the basic OLS else if then it will add a constraint to the coefficient. As we increase the value of this constraint causes the value of the coefficient to tend towards zero. This leads to tradeoff of higher bias (dependencies on certain coefficients tend to be 0 and on certain coefficients tend to be very large, making the model less flexible) for lower variance.

Limitation: Ridge regression decreases the complexity of a model but does not reduce the number of variables since it never leads to a coefficient been zero rather only minimizes it. Hence, this model is not good for feature reduction.

Lasso Regression:

Lasso regression stands for Least Absolute Shrinkage and Selection Operator. It adds penalty term to the cost function. This term is the absolute sum of the coefficients. As the value of

coefficients increases from *0* this term penalizes, cause model, to decrease the value of coefficients to reduce loss. The difference between ridge and lasso regression is that it tends to make coefficients to absolute zero as compared to Ridge which never sets the value of coefficient to absolute zero.

Limitation: Lasso sometimes struggles with some types of data. If the number of predictors *(p)* is greater than the number of observations *(n)*, Lasso will pick at most n predictors as non-zero, even if all predictors are relevant (or may be used in the test set). If there are two or more highly collinear variables, then LASSO regression selects one of them randomly which is not good for the interpretation of data

ANALYSIS

LOGISTIC REGRESSION

Research Question 3: Build a model to predict if the person requested for the loan amount should be approved or not.

Generally, when our data follows normal distribution, we use the probability density functions to analyze the data. But what if our data is not following the normal distribution curve. Then we must use the generalized linear models.

GLM generally consists of three components. The systematic component, the link function, and the random component. The random component defines the response variable's probability distribution, such as the normal distribution in a classical regression model or the binomial distribution in a binary logistic regression model. There is no separate error term in the model; this is the sole random component.

```
   {r message=FALSE, warning=FALSE}
#Split the data into a train and test set - refer to the Feature_Selection_R.pdf
document for information on how to split a dataset.
set.seed(222)
credit_trainIndex <- createDataPartition(credit_train$`LoanApproval`, p = 0.6, list =
FALSE)
SampleTrain= credit_train[credit_trainIndex,]
SampleTest= credit_train[-credit_trainIndex,]
```

```
#Using GLM
model= glm(as_factor(`LoanApproval`)~.,data = SampleTrain,family = binomial(link =
"logit"))
summary(model)
```

```
##
## Call:
## glm(formula = as_factor(LoanApproval) ~ ., family = binomial(link = "logit"),
##     data = SampleTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5108  -1.1703   0.9036   1.1717   1.5080
##
## Coefficients: (1 not defined because of singularities)
##                                      Estimate Std. Error z value
## (Intercept)                          1.293e+01  8.287e+00   1.560
## Applicant_ID...1                    -2.636e-06  1.618e-06  -1.629
## Applicant_ID...2M                   -7.481e-02  6.854e-02  -1.091
## Applicant_ID...3                    -1.795e-02  5.702e-02  -0.315
## Owned_Email                          0.055147 .
## Job_TitleCleaning staff              0.047545 *
## Job_TitleCooking staff               0.127477
## Job_TitleCore staff                  0.037523 *
## Job_TitleDrivers                     0.061623 .
## Job_TitleHigh skill tech staff       0.040216 *
## Job_TitleHR staff                    0.780282
## Job_TitleIT staff                    0.513419
## Job_TitleLaborers                    0.008179 **
## Job_TitleLow-skill Laborers          0.555918
## Job_TitleManagers                    0.001921 **
## Job_TitleMedicine staff              0.193563
## Job_TitlePrivate service staff       0.024293 *
## Job_TitleRealty agents               0.965865
## Job_TitleSales staff                 0.020235 *
## Job_TitleSecretaries                 0.677061
## Job_TitleSecurity staff              0.212581
## Job_TitleWaiters/barmen staff        0.098914 .
## Total_Family_Members                 0.438600
## Applicant_Age                        0.000825 ***
## Years_of_Working                     0.007222 **
## Total_Bad_Debt                       0.734688
## Total_Good_Debt                      0.070130 .
```

As you can see in this picture, we have first used a very general approach to find out which are the more significant variables in the dataset.
glm(formula = as_factor(LoanApproval) ~ ., family = binomial(link = "logit"), data = SampleTrain) was the general formula. Upon using this we found out that Job_TitleLaborers, Applicant_Age , Years_of_Working and Job_TitleManagers are some of the significant variables.

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8733.5  on 6299  degrees of freedom
## Residual deviance: 8666.0  on 6242  degrees of freedom
## AIC: 8782
##
## Number of Fisher Scoring iterations: 11
```

We then used the following more focused formula.
glm(formula = as_factor(LoanApproval) ~ SampleTrain$Applicant_Age + SampleTrain$Years_of_Working + SampleTrain$Total_Good_Debt, family = binomial(link = "logit"), data = SampleTrain)
This one gave us a more in-depth analysis of the data.

```r
#Model 2
model2=glm(as_factor(`LoanApproval`)~SampleTrain$Applicant_Age+SampleTrain$Years_of_Working+SampleTrain$Total_Good_Debt,data = SampleTrain,family = binomial(link = "logit"))
summary(model2)
#Display model coef
exp(coef(model2))
```

```
##
## Call:
## glm(formula = as_factor(LoanApproval) ~ SampleTrain$Applicant_Age +
##     SampleTrain$Years_of_Working + SampleTrain$Total_Good_Debt,
##     family = binomial(link = "logit"), data = SampleTrain)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.339  -1.180   1.058   1.172   1.295
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   0.180016   0.115479   1.559  0.11903
## SampleTrain$Applicant_Age    -0.008041   0.002812  -2.859  0.00424 **
## SampleTrain$Years_of_Working  0.012892   0.004194   3.074  0.00211 **
## SampleTrain$Total_Good_Debt   0.002825   0.001690   1.671  0.09469 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8733.5  on 6299  degrees of freedom
## Residual deviance: 8717.4  on 6296  degrees of freedom
## AIC: 8725.4
##
## Number of Fisher Scoring iterations: 3
```

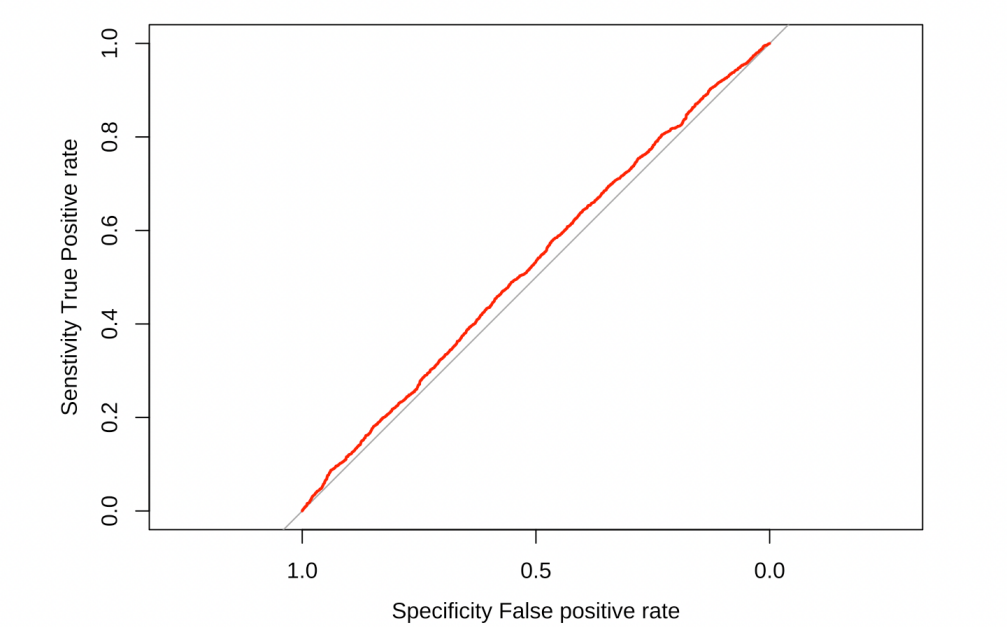
There was a very slight decrease in the AIC value of the first and the second model. The AIC of the first model was 8782 and that of the second one was 8725.4. Which proves that the second model is a better one.

```{r}
#MAKE PREDICTIONS Train set
PSampleTrain= predict(model2,newdata = SampleTrain,type = "response")
Pclasess= as.factor(ifelse(PSampleTrain>=0.5,"Yes","No"))

```

```{r}
#Confusion Matrix
confusionMatrix(Pclasess,as.factor(SampleTrain$LoanApproval),positive = "Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1374 1485
##        Yes 1790 1651
##
##                Accuracy : 0.4802
##                  95% CI : (0.4678, 0.4926)
##     No Information Rate : 0.5022
##     P-Value [Acc > NIR] : 0.9998
##
##                   Kappa : -0.0393
##
##  Mcnemar's Test P-Value : 1.084e-07
##
##             Sensitivity : 0.5265
##             Specificity : 0.4343
##          Pos Pred Value : 0.4798
##          Neg Pred Value : 0.4806
##              Prevalence : 0.4978
##          Detection Rate : 0.2621
##    Detection Prevalence : 0.5462
##       Balanced Accuracy : 0.4804
##
##        'Positive' Class : Yes
##
```

In the confusion matrix we found that the true positive value i.e., the sensitivity of the model is 0.5265 and the specificity which is also called the true negative value is 0.4343.




```
Roc$auc
```

```
## Area under the curve: 0.5268
```

A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The line shows the random classifier and anything above it is a better model. Since the red line is above the random classifier, we say it is a better model but not an accurate one. This is because the AUC value is 0.5268. Model with AUC 0.5 does no better than a model that performs random guessing. Anything between 0.7-0.8 is considered acceptable model but anything between 0.8-0.9 is considered excellent and above 0.9 is considered outstanding.

As mentioned above, since the data is bad even after data clean up and normalization, receiving an AUC value of 0.5 is no surprise.

MULTIPLE AND LINEAR REGRESSION MODEL

Research Question 1: Build a method or a learning model to predict if an applicant is 'good' or 'bad' client.

Below is the hypothesis.

Ho: If the credit score is low, applicant is bad

Ha: If the credit score is more, applicant is good

We have selected this model to predict the outcome of dependent variable based on the independent variables used.

Current_loan_amount is considered as the dependent variable and monthly_debt is taken as independent variable.

```
#Calculation correlation coeeficient
cor1 <- round(cor(inputdataset$Current_loan_amount, inputdataset$monthly_debt),3)
cor2 <- round(cor(inputdataset$Current_loan_amount, inputdataset$credit_score),3)
cor3 <-round(cor(inputdataset$Current_loan_amount,
inputdataset$Years_of_credit_history),3)
cor4 <- round(cor(inputdataset$Current_loan_amount,inputdataset$Number_of_Open_Accoun
ts),3)
cor5 <- round(cor(inputdataset$Current_loan_amount
,inputdataset$Current_Credit_Balance),3)

#selecting required columns
Resultdataset <- inputdataset %>% select (Current_loan_amount, credit_score,
Years_of_credit_history, Number_of_Open_Accounts, Current_Credit_Balance,
annual_income, Maximum_Open_Credit)

#correlation matrix
correlation <- cor(Resultdataset, use = 'pairwise')
correlation
corrplot(correlation, type='upper', col = brewer.pal(n=8, name = "Set3"))

#linear regression
l1 <- lm(inputdataset$Current_loan_amount~inputdataset$monthly_debt)
summary(l1)
l2<-lm(inputdataset$Current_loan_amount~inputdataset$credit_score)
l3<- lm(inputdataset$Current_loan_amount~inputdataset$Years_of_credit_history)
l4 <- lm(inputdataset$Current_loan_amount~ inputdataset$Number_of_Open_Accounts)
l5 <- lm(inputdataset$Current_loan_amount ~inputdataset$Current_Credit_Balance)

#Calculation determination coefficient
cord1 <- (cor1^2)
cord2 <- cor2^2
cord3 <- cor3^2
cord4 <- cor4^2
cord5 <- cor5^2
```

The linear regression model is given below:

```
Call:
lm(formula = inputdataset$Current_loan_amount ~ inputdataset$monthly_debt)

Residuals:
    Min      1Q  Median      3Q     Max
-13791156 -13536066 -13382228 -13121005  87187432

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)           1.381e+07  3.265e+05  42.309   <2e-16 *** inputdataset$monthly_debt -6.043e+00  1.448e+01  -0.417    0.676
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 33990000 on 38008 degrees of freedom
Multiple R-squared:  4.581e-06,         Adjusted R-squared:  -2.173e-05
F-statistic: 0.1741 on 1 and 38008 DF,  p-value: 0.6765
```

Here, we see that Current_loan_amount is highly significant and determination value is 0.000004581.

Multiple regression model is given below:

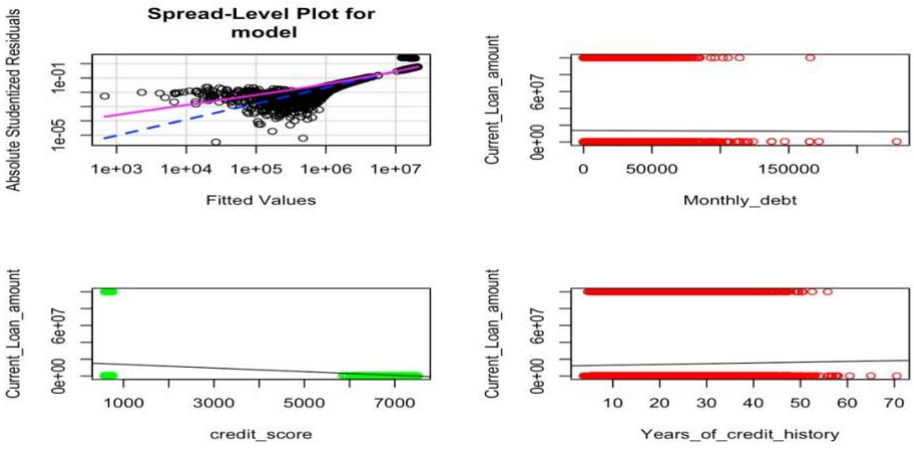
Here, Current_loan_amount is dependent variable, monthly_debt , credit_score, Years_of_credit_history, Number_of_Open_Accounts, Current_Credit_Balance are independent

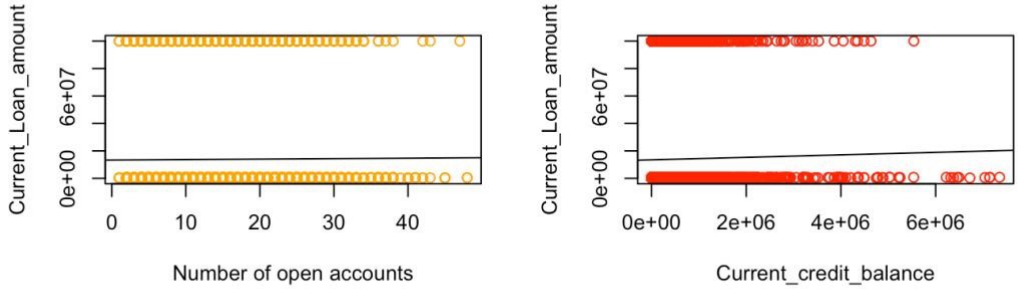
```
#multiple regression formula:
print(paste("Multiple regression y'=", a$coefficients[1],6,"+",
round(a$coefficients[2],6),"monthly_debt +",round(a$coefficients[3],6),"credit_score
+", round(a$coefficients[4],6),"Years_of_credit_history
+",round(a$coefficients[5],6),"Number_of_open_accounts +",
round(a$coefficients[6],6),"Current_credit_balance "))

a1 <- a$coefficients[1]
b1 <- a$coefficients[2]
b2 <- a$coefficients[3]
b3 <- a$coefficients[4]
b4 <- a$coefficients[5]
b5 <- a$coefficients[6]

#calculating the predicted and residual values
new_table <- mutate(Resultdataset,
predicted_value=round(a1+b1*inputdataset$monthly_debt+b2*inputdataset$credit_score+b3
*inputdataset$Years_of_credit_history+b4*inputdataset$Number_of_Open_Accounts+b5*inpu
tdataset$Current_Credit_Balance,3), Residual_value
=round(inputdataset$Current_loan_amount-predicted_value,3))
new_table
```

```
Call:
lm (formula = inputdataset$Current_loan_amount ~ inputdataset$monthly_debt +
inputdataset$credit_score + inputdataset$Years_of_credit_history +
   inputdataset$Number_of_Open_Accounts + inputdataset$Current_Credit_Balance)

Residuals:
    Min      1Q  Median      3Q     Max
-20280987 -14363368 -13878864 -13306069 87698556

Coefficients:
                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                      1.420e+07  6.472e+05  21.945  < 2e-16 ***
inputdataset$monthly_debt       -2.880e+01  1.702e+01  -1.693  0.09055 .
inputdataset$credit_score       -2.150e+03 1.199e+02 -17.937  < 2e-16 ***
inputdataset$Years_of_credit_history 8.201e+04  2.595e+04  3.160  0.00158 **
inputdataset$Number_of_Open_Accounts 4.529e+04 3.774e+04  1.200 0.23015
inputdataset$Current_Credit_Balance  8.734e-01  6.194e-01  1.410  0.15849
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 33840000 on 38004 degrees of freedom
Multiple R-squared:  0.008829,          Adjusted R-squared:  0.008699
F-statistic: 67.71 on 5 and 38004 DF,  p-value: < 2.2e-16
```

Here, we see that current_loan_amount, credit_score and years_of_open_accounts are significant. Pvalue is approximately equal to 0.00000000000000022.

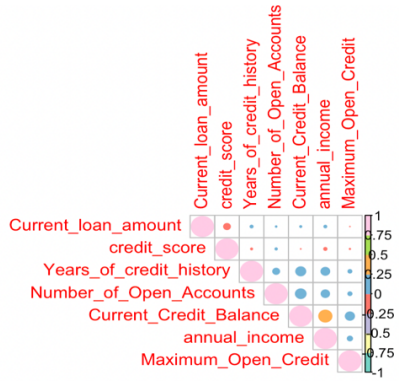
"Multiple regression y'= 14201754.8589036 6 + -28.803994 monthly_debt + -2150.086995 credit_score + 82009.855132 Years_of_credit_history + 45293.79792 Number_of_open_accounts + 0.873444 Current_credit_balance "



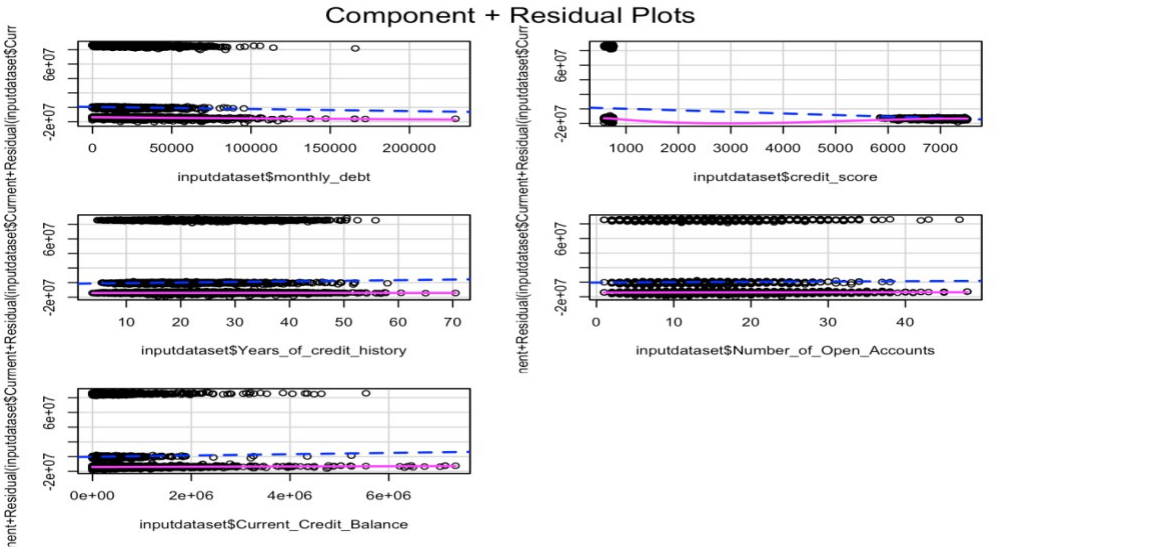


Here, we see that current_loan_amount is having negative correlation with credit_score.
Current_loan_amount is having negative correlation with monthly_debt.
Current_loan_amount is having positive correlation with number of open accounts.
Current_loan_amount is having positive correlation with credit_balance.

None of the values are falling within the correlation line. This is because of the data present in the dataset. Even after normalization and data clean up, we see that there is no strong correlation between the variables which implies that there are bad data in the dataset.



Here, in the correlation matrix, we see that current_loan_amount has a correlation of 1 (maximum correlation) with itself. Current_loan_amount is having a negative correlation of - 0.25 with credit score which is - 25%.



Partial residuals (component + residual) plot, the plots for the individual variables show that none of the component variables are linear: The dotted blue lines show the least squares fit,

and the pink lines indicate the real shape of the data. A component residual plot adds a line indicating where the line of best fit lies. A significant difference between the residual line and the component line indicates that the predictor does not have a linear relationship with the dependent variable. We see that there is a significant difference between the dependent variables and independent variables here and that is the reason the lines are not colliding. We only see that credit score merges with loan amount between 6000 to 7000.

Below shows the determination and F- statistic values. Here alpha is selected as 0.05.

```
Suggested power transformation:  -0.1191151
   value
67.70617
       value
1.075877e-70
[1] "coefficient of determination is 0.00882912114721729"
      value
"Reject Ho"
[1] "The conclusion is:  Reject Ho"
```

We can conclude that we have enough evidence to reject null hypothesis and to conclude that if credit score is good, applicant is good.

REGULARIZATION (LASSO/RIDGE)

Research Question 2: Build a model to predict the value of loan credit the applicant is eligible for.

Here, we have split the dataset into 2 halves, training, and test dataset.

```
#Split the data into a train and test set – refer to the Feature_Selection_R.pdf document for information on how
to split a dataset.
set.seed(222)
credit_trainIndex <- createDataPartition(credit_train$`Term`, p = 0.7, list = FALSE)
SampleTrain= credit_train[credit_trainIndex,]
SampleTest= credit_train[-credit_trainIndex,]
```

```
train_x= data.matrix(subset(SampleTrain, select = -c(`Maximum Open Credit`)))

test_x= data.matrix(subset(SampleTest, select = -c(`Maximum Open Credit`)))

train_y= SampleTrain$`Maximum Open Credit`
test_y= SampleTest$`Maximum Open Credit`
```

Maximum open credit is our dependent variable, and we are predicting its value using regularization methods.

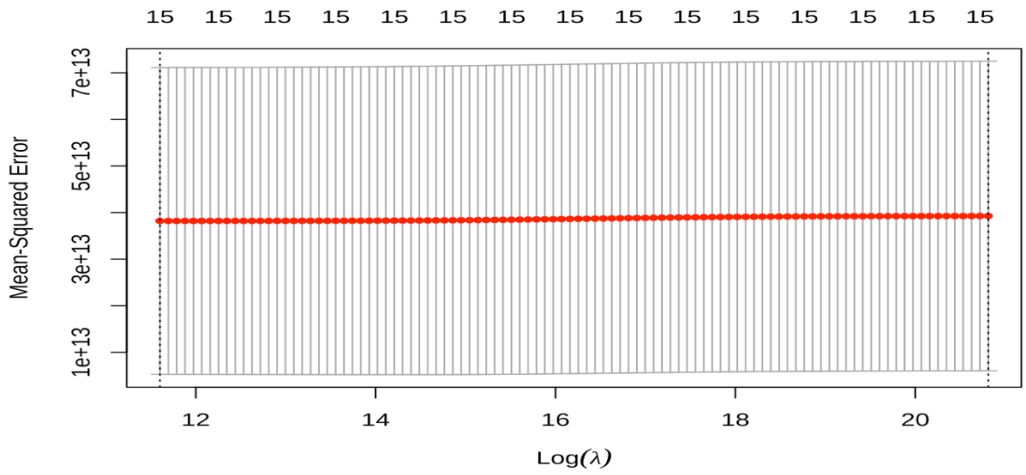
RIDGE REGULARIZATION

In Ridge regression, we add a penalty term which is equal to the square of the coefficient. Using cv.glmnet(), we are performing cross validation to produce a value for lamba.

```
#RIDGE REGRESSION
#Using Glmnet function finding best value for lambda
set.seed(222)
ridge= cv.glmnet(train_x,train_y,nfolds = 10,alpha=0)
print(paste("lamnda Minimum is",ridge$lambda.min))
```

```
"lamnda Minimum is 109352.629747606"
"lamnda 1 Standard error is 1093526297.47606"
```

We see that lambda.min gives minimum mean cross-validated error and lambda.1se gives the most regularized model such that error is within one standard error of the minimum.



The above plot shows that lambda.min value is 1.60233 and lambda.1se value is 20.81267.

```
#Model with the 1se lambda value
RSE1Model=glmnet(train_x,train_y,alpha = 0,lambda = ridge$lambda.1se)
#MinModel coefficients
coefficients(RSE1Model)
```

Model with the minimum lambda value

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                                        s0
## (Intercept)                -9.141847e+04
## Loan ID                    -3.246570e+01
## Customer ID                 2.145521e+01
## Loan Status                 8.329112e+04
## Current Loan Amount        -7.172723e-05
## Term                        8.106377e+04
```

```
## Credit Score                    .
## Annual Income                   .
## Years in current job          1.679702e+04
## Home Ownership               -9.621988e+04
## Purpose                       9.607698e+04
## Monthly Debt                 -2.706639e+01
## Years of Credit History       9.725063e+03
## Months since last delinquent -4.614987e+03
## Number of Open Accounts      -6.484075e+02
## Number of Credit Problems    -2.356814e+04
## Current Credit Balance        3.414152e+00
## Bankruptcies                 -1.269041e+03
## Tax Liens                       .
```

Model with the 1se lambda value

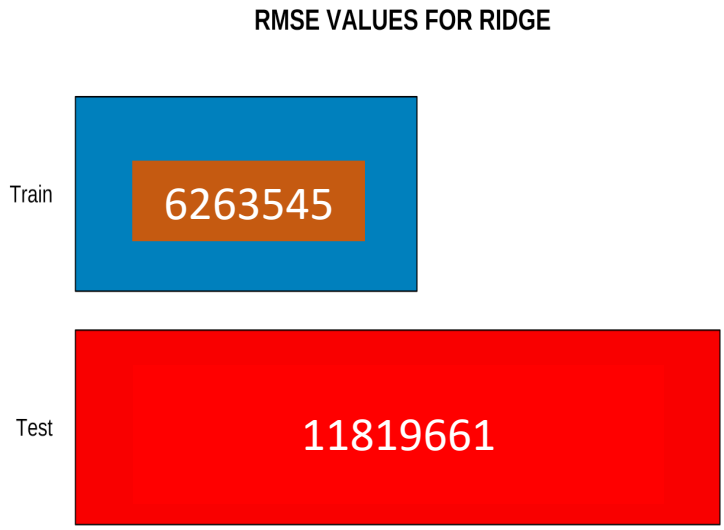
```
19 x 1 sparse Matrix of class "dgCMatrix"
##                                       s0
## (Intercept)                   7.690501e+05
## Loan ID                      -1.909007e-01
## Customer ID                   1.713110e-01
## Loan Status                   9.430353e+02
## Current Loan Amount           1.540664e-06
## Term                         -8.111963e+02
## Credit Score                    .
## Annual Income                   .
## Years in current job          8.153984e+01
## Home Ownership               -1.491370e+03
## Purpose                       4.301876e+02
## Monthly Debt                  1.236068e-01
## Years of Credit History       2.528326e+02
## Months since last delinquent -8.079629e+00
## Number of Open Accounts       1.632680e+02
## Number of Credit Problems    -1.563189e+03
## Current Credit Balance        1.735174e-02
## Bankruptcies                 -1.685960e+03
## Tax Liens                       .
```

Training dataset prediction and calculating RMSE

The RMSE value for training dataset is **6263545**.

Test set prediction and calculating RMSE

The RMSE value for test dataset is **11819661**.



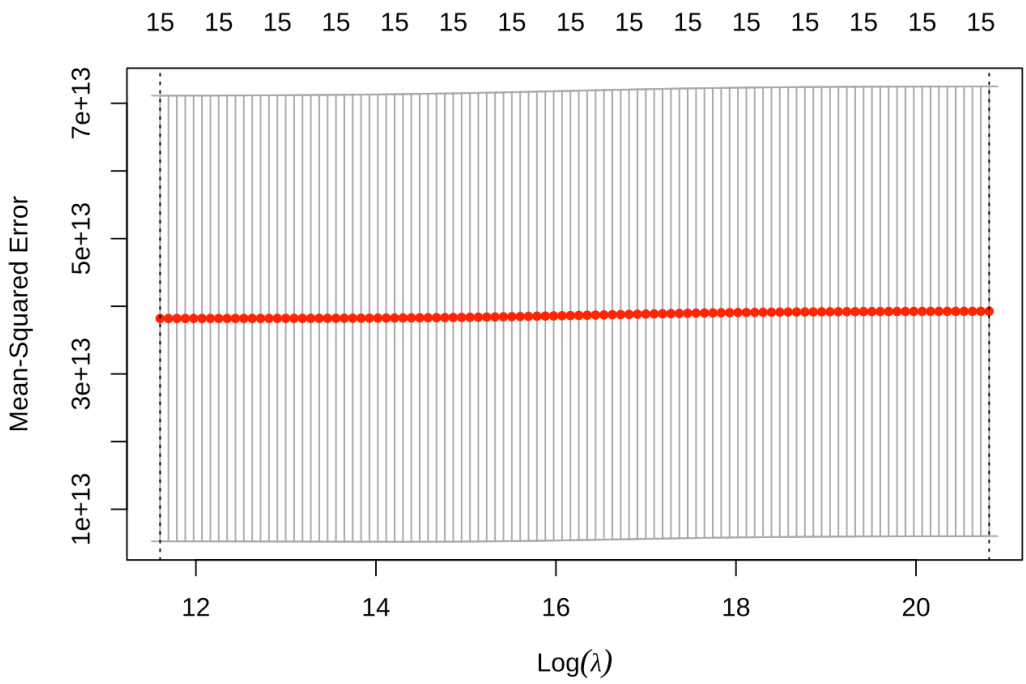
Ridge model does not remove unwanted variables in the model and makes use of it for the prediction. This can cause issue because we can end up designing a bad model.

LASSO REGULARIZATION

Lasso regression stands for Least Absolute Shrinkage and Selection Operator. Using Glmnet function finding best value for lambda

```
#LASSO REGRESSION
#Using Glmnet function finding best value for lambda
set.seed(123)
lasso= cv.glmnet(train_x,train_y,nfolds = 10,alpha=1)
print(paste("lamnda Minimum is",lasso$lambda.min))
```

```
"lamnda Minimum is 34984.8274783239"
"lamnda 1 Standard error is 1093526.29747605"
```



Model with the minimum lambda value

```
19 x 1 sparse Matrix of class "dgCMatrix"
##                                       s0
## (Intercept)                  78490.206173
## Loan ID                        -15.829887
## Customer ID                      5.460328
## Loan Status                  17279.638429
## Current Loan Amount                 .
## Term                         14541.787314
## Credit Score                        .
## Annual Income                       .
## Years in current job          6326.187479
## Home Ownership              -60689.312813
## Purpose                      83351.538646
## Monthly Debt                   -22.819719
## Years of Credit History       5804.565322
## Months since last delinquent -3488.799273
## Number of Open Accounts             .
## Number of Credit Problems           .
## Current Credit Balance           3.322253
## Bankruptcies                        .
```

```
## Tax Liens                          .
```

Model with the 1se lambda value

```
19 x 1 sparse Matrix of class "dgCMatrix"
##                                       s0
## (Intercept)                 7.784415e+05
## Loan ID                     .
## Customer ID                 .
## Loan Status                 .
## Current Loan Amount         .
## Term                        .
## Credit Score                .
## Annual Income               .
## Years in current job        .
## Home Ownership              .
## Purpose                     .
## Monthly Debt                .
## Years of Credit History     .
## Months since last delinquent .
## Number of Open Accounts     .
## Number of Credit Problems   .
## Current Credit Balance      6.306698e-15
## Bankruptcies                .
## Tax Liens                   .
```

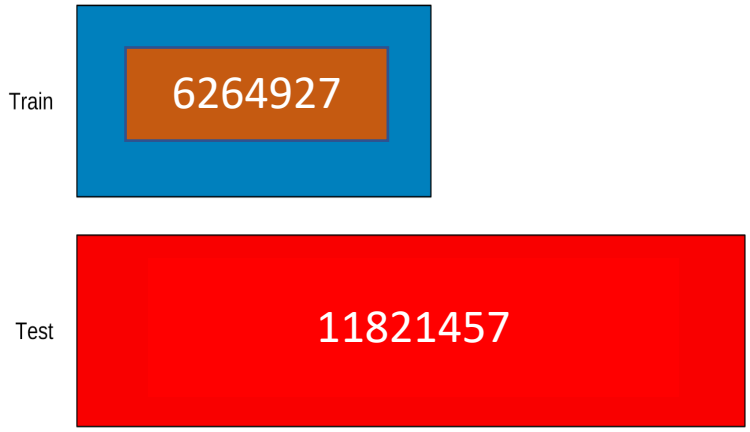
Training dataset prediction and calculating RMSE

The RMSE value of training dataset is **6264927.**

Testing dataset prediction and calculating RMSE

The RMSE value of test dataset is **11821457.**

RMSE VALUES FOR LASSO



Here, we see that lasso model has removed most of the variables and included only the ones that are significant for predicted model. This is useful since we get rid of unwanted variables.

From both the regularization model, we see negligible difference in value w.r.t RMSE. Lasso model is having lower lambda values when compared to Ridge model. Ridge model is having lower RMSE value for training dataset when compared to LASSO and lower RMSE value for testing dataset. Since there is a huge difference in the RMSE value, both are not a good model to perform prediction.

CONCLUSION

We have used the dataset from Kaggle to perform our analysis. We have used 3 models, GLM, Linear & Multiple Regression and LASSO/RIDGE Regularization to predict our analysis. We have proved that if the credit score is good, applicant is a good customer using multiple regression model. We have used Lasso and ridge method to predict the maximum open credit value that the customer is eligible for. We used Logistic regression model to predict if loan amount should be approved or not for a customer. We concluded that both ridge and lasso model is not great one to predict the value due to high difference in RMSE value for the training dataset. Based on our analysis we found that the data is untidy, and we can see it clearly in the models. If the data was good even after data clean up and normalization, we would have a better model to predict the required data.

REFERENCES

Dataset for Bank Loan Prediction:
https://www.kaggle.com/datasets/omkar5/dataset-for-bank-loan-prediction

Krishan Kumar Pandey, Abhishek Giri, Saket Sharma, Anurag Singh, "Predictive Analysis of Classification Algorithms on Banking Data", Computing Power and Communication Technologies (GUCON) 2021 IEEE 4th International Conference on, pp. 1-5, 2021.

Ambika, Santosh Biradar, "Survey on Prediction of Loan Approval Using Machine Learning Techniques", International Journal of Advanced Research in Science, Communication and Technology, pp. 449, 2021.

APPENDIX

Below is the name of the R script (RMD file) written for the given problem statements.

Multiple Regression.Rmd

EDA.Rmd

Capstone (lasso and Ridge).Rmd

Logistic.Rmd

These files are attached in canvas.