Regression

Problem Statement:

After identifying the customers who would default using a classification model,estimating the percentage of loss for each defaulted customer using Regression Analysis techniques.

Loading Libraries

```
library(caret)

library(dplyr)

library(corrplot)

library(glmnet)

library(tidyverse)

library(tidyr)

library(randomForest)
```

```
data<-read.csv('data_cleaned.csv')
```

Data Transformation

```
# Create a new column called 'default' with a value of 1 is loss is above 0 and 0 is loss is 0


data$default <- ifelse(data$loss == 0, 0, 1)
data$default<- as.factor(data$default)
```

Data Preparation

```
#Normalizing loss column by dividing with 100
data$loss <- (data$loss/100)


#Creating subset of customers who have defaulted (i.e loss > 0)

default_customers<- subset(data, data$default == 1)
```

Create a preprocessing model that eliminates near zero variance variables, highly correlated variables, and then does the imputation of missing values with the median

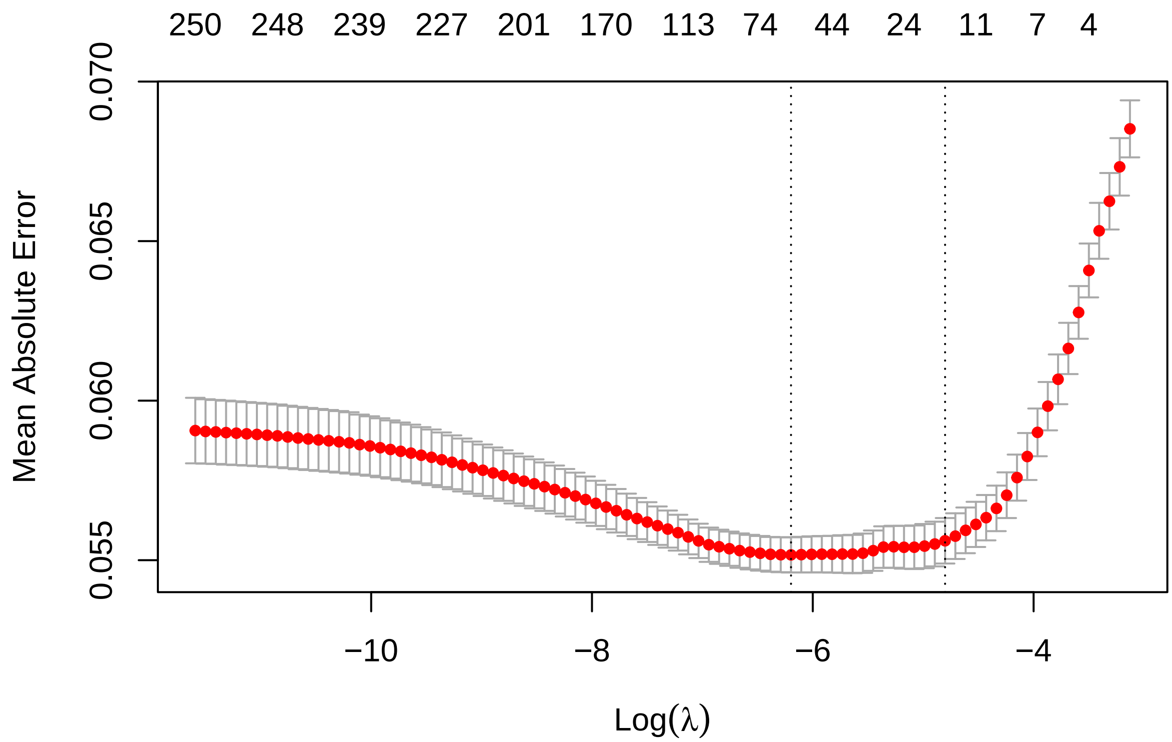
```r
data1<-select(default_customers,-c(f736,f764))

preProcessModel <- preProcess(data1[,-c(701,702)], method = c("nzv", "corr", "medianImpute"))
Preprocessed_default <- predict(preProcessModel, data1)
```

Feature selection for regression(loss) using Lasso

```r
set.seed(3456)

X1 <- as.matrix(Preprocessed_default[ ,-c(258,259)])
Y1 <- as.vector(Preprocessed_default$loss)


lasso_model <- cv.glmnet(X1, Y1, alpha = 1, family = "gaussian", nfolds = 10, type.measure = "mae")


summary(lasso_model)
```

```
##            Length Class  Mode
## lambda     92     -none- numeric
## cvm        92     -none- numeric
## cvsd       92     -none- numeric
## cvup       92     -none- numeric
## cvlo       92     -none- numeric
## nzero      92     -none- numeric
## call        7     -none- call
## name        1     -none- character
## glmnet.fit 12     elnet  list
## lambda.min  1     -none- numeric
## lambda.1se  1     -none- numeric
## index       2     -none- numeric
```

```r
plot(lasso_model)
```




```r
#Finding the minimum value of lambda

lasso_model$lambda.min
```

```
## [1] 0.002034776
```

```r
#Finding the coefficients at minimum lambda value

cv_lasso_coefs <- coef(lasso_model, s = "lambda.min")
cv_lasso_coefs
```

```
## 258 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept)  1.840101e-01
## id           .
## f1           .
## f3           .
## f5           .
## f6           .
## f13          5.471326e-03
## f16          .
## f19          .
## f25          .
## f26          .
## f29          .
```

```
## f31          .
## f32          .
## f43          .
## f44          .
## f47         -1.111446e-02
## f54          .
## f57         -1.471261e-02
## f64         -8.822112e-03
## f65          .
## f66          .
## f67          9.422909e-03
## f70         -5.297111e-02
## f71          1.464101e-04
## f73          .
## f76          6.826401e-04
## f80          .
## f81          .
## f82          .
## f90          .
## f92          .
## f94          .
## f99          .
## f100         .
## f102         .
## f104         .
## f109         .
## f110         .
## f112        -3.900654e-03
## f121         2.753497e-03
## f122        -2.826428e-04
## f124        -8.180303e-02
## f129        -6.057934e-02
## f130         .
## f131         .
## f132         .
## f133         .
## f139         .
## f140        -1.067397e-03
## f143         2.881755e-04
## f144         1.542220e-04
## f146         .
## f148         .
## f149         .
## f150         .
## f151         3.064095e-03
## f153         .
## f158         .
## f159         .
## f161        -1.239266e-03
## f163         .
## f168         .
## f170         .
## f171         .
## f173         .
```

```
## f178          .
## f180          .
## f181          .
## f183          .
## f188          .
## f189          .
## f190          .
## f191          .
## f193          .
## f198         -1.044576e-02
## f199          .
## f200          .
## f202          .
## f203          .
## f204          .
## f208          .
## f209          .
## f212          4.551465e-04
## f213          1.634143e-03
## f217          .
## f218          .
## f220          .
## f221          .
## f223          .
## f229          2.226747e-02
## f231          .
## f233          .
## f238          .
## f239          .
## f241          .
## f243          .
## f248          .
## f249          .
## f251          .
## f259          1.609275e-03
## f261         -5.784456e-03
## f268         -1.332081e-01
## f269          .
## f270          5.812275e-02
## f272          .
## f277          .
## f278          .
## f280          .
## f281          1.204297e-03
## f287          .
## f288          .
## f289          .
## f314          .
## f316          .
## f320          .
## f321          .
## f322          .
## f324          .
## f329         -1.589632e-02
```

```
## f330        -4.195500e-03
## f331         .
## f333         1.736368e-08
## f338         .
## f339         .
## f340         .
## f341        -1.737078e-03
## f357         .
## f358         .
## f361         .
## f366         .
## f367         .
## f374         .
## f378         .
## f382         7.138006e-13
## f383         .
## f384         .
## f385         .
## f391         1.057716e-44
## f393         .
## f398         .
## f402         1.021603e-02
## f403         .
## f411         .
## f412         .
## f413        -4.771116e-03
## f420         .
## f421         .
## f422         .
## f425         .
## f428         .
## f430         .
## f431         .
## f432         .
## f433         .
## f436         .
## f441         .
## f442         .
## f444         .
## f448         .
## f451         .
## f458         .
## f461         .
## f468         .
## f470         .
## f471         .
## f472         .
## f479        -1.665070e-03
## f489         .
## f499         .
## f509        -5.191314e-03
## f514        -1.084066e-04
## f516         .
## f518         .
```

```
## f522         .
## f523        -3.781328e-09
## f524         .
## f525         .
## f526         .
## f530         .
## f533         .
## f536         .
## f546        -3.579556e-02
## f556         .
## f566         .
## f567         .
## f587         .
## f588        -1.810972e-03
## f589         .
## f591         .
## f598        -8.164883e-03
## f600         .
## f601         .
## f609         .
## f611         .
## f612         .
## f613         .
## f614         .
## f618         .
## f621         .
## f623         5.140942e-13
## f628         .
## f629         2.074204e-02
## f631         .
## f634         .
## f636         5.962112e-07
## f637         .
## f638         .
## f639         .
## f640         .
## f643         .
## f646         .
## f647         .
## f648        -9.158841e-05
## f649         .
## f650         .
## f651         .
## f652         2.374468e-06
## f653         .
## f654         1.476647e-04
## f656         .
## f659         .
## f660         .
## f661         .
## f663         1.722405e-05
## f664         .
## f669         .
## f671         2.997383e-02
```

```
## f672         .
## f673         .
## f674        -5.577481e-06
## f675         .
## f677        -2.781395e-04
## f679         .
## f680         .
## f682         .
## f699         .
## f715         .
## f716         .
## f725         7.315022e-04
## f733         .
## f734        -5.516792e-04
## f735         .
## f739         .
## f740         .
## f742         .
## f743         .
## f744         .
## f746         .
## f755         .
## f756         .
## f760         .
## f763        -3.241678e-03
## f765         .
## f766         1.460551e-01
## f768         4.290787e-02
## f774        -5.832689e-02
## f775         .
```

```r
#Converting coefficients obtained into a dataframe

cv_lasso_coefs <- data.frame(name = cv_lasso_coefs@Dimnames[[1]][cv_lasso_coefs@i + 1], coefficient = c


#Removing the intercept from the coefficient data frame

cv_lasso_coefs <- cv_lasso_coefs[-1, ]


#Converting the coefficient data frame to vector

cv_lasso_coefs <- as.vector(cv_lasso_coefs$name)

#Adding loss variable back to the vector

cv_lasso_coefs1 <- c(cv_lasso_coefs,"loss")


#Combining the columns selected by lasso with variable selection and forming a new dataset

data_new<-select(default_customers,cv_lasso_coefs1)
```

Creating training and test partition with 70% for training and 30% for test

```
set.seed(6782)

Split_data <- createDataPartition(data_new$loss,p=.7,list=FALSE,times=1)
Training <- data_new[Split_data,]
Validation <- data_new[-Split_data,]
```

Modeling Strategy *Building Bagged Decision Tree model using Random Forest*

```
num_trees <- 100 #number of trees
sample_size <- 50 #size of the bootstrap sample used to grow each tree

# Building the Bagged Decision Tree Regression model

bagged_model <- randomForest(loss ~ ., data = Training,
                             ntree = num_trees,
                             mtry = 10,
                             sampsize = sample_size,
                             replace = TRUE)

summary(bagged_model)
```

```
##                 Length Class  Mode
## call               7   -none- call
## type               1   -none- character
## predicted       5167   -none- numeric
## mse              100   -none- numeric
## rsq              100   -none- numeric
## oob.times       5167   -none- numeric
## importance        58   -none- numeric
## importanceSD       0   -none- NULL
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            11   -none- list
## coefs              0   -none- NULL
## y               5167   -none- numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```
Predictions<- predict(bagged_model, Validation)
```

#Calculating Loss Metrics for the model

```
MAE<-MAE(Predictions,Validation$loss,na.rm=TRUE)
MAE
```

```
## [1] 0.06264001
```

Reading and preprocessing Test Data

```
data10<-read.csv("new_defaulted_test_customers.csv")

#Replacing null values with zeroes

data11 <- data10 %>% mutate_all(funs(replace_na(.,0)))

null_percent <- apply(data11 == 0, 2, mean)

#Removing columns having more than 30% null values

cols <- names(null_percent[null_percent <= 0.3])

new_test_file <- data11[, cols]


#Check if the columns with more than 30% null values are deleted

Sums<-(colSums(new_test_file==0)/nrow(new_test_file))*100

#Combining the variables from lasso model with test data to obtain a new test data with selected variab

Test_data_new<-select(new_test_file,cv_lasso_coefs)
```

Running the model on test data

```
Test_loss_Predictions<-predict(bagged_model, Test_data_new)


write.csv(Test_loss_Predictions,file="Final_Predictions.csv")
```