**Logistic Regression to Predict Admission for Student in University**

**Introduction: -**

This dataset is made for forecast of Graduate Admissions from an Indian point of view. The dataset contains a few parameters which are viewed as vital amid the application for master’s Programs. The parameters included are: 1. GRE Scores (out of 340) 2. TOEFL Scores (out of 120) 3. College Rating (out of 5) 4. Mission statement and Letter of Recommendation Strength (out of 5) 5. Undergrad GPA (out of 10) 6. Research Experience (either 0 or 1) 7. Possibility of Admit (extending from 0 to 1). This dataset is motivated by the UCLA Graduate Dataset. The test scores and GPA are in the more seasoned configuration. The dataset is possessed by Mohan S Acharya. This dataset was worked with the motivation behind helping understudies in shortlisting colleges with their profiles. The anticipated yield gives them a reasonable thought regarding their odds for a specific college.

**Motto: -**

Predict whether a student can get admit in any of top 5 rating universities or not.

**Data: -**

The data set considered contains 8 variables with 1100 observations. GRE Score, TOEFL Score, SOP, LOR, CGPA, University Rating, Research are independent variables and Admit is dependent. Research and Admit are of two levels 0 and 1. University Rating has 5 levels and remaining are numeric values.

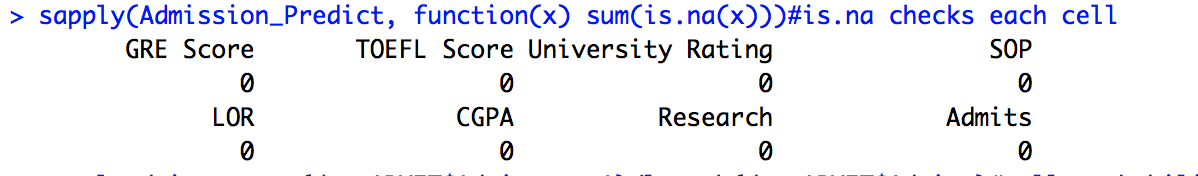
Let’s Check, number of empty cells in each variable in data

Figure 1-show there are no empty cells data

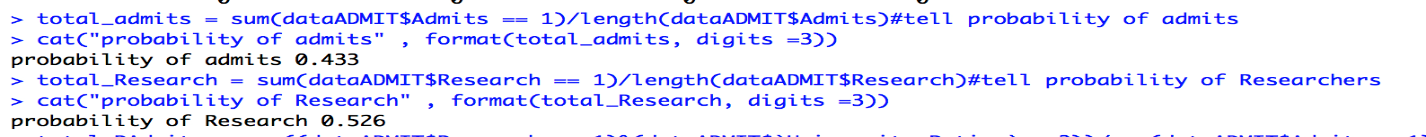
**Probability of students who got admit is 0.433 and students with research 0.56

Figure 2-show the probability of students who got admit and have research

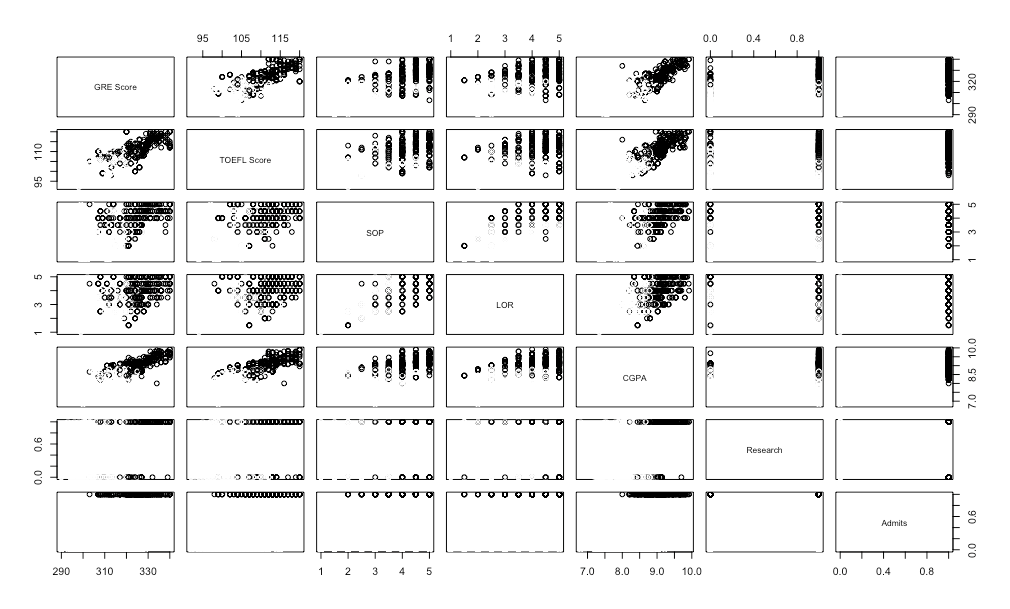


Figure 3-scatter plot of all variables

The relation between each independent variable with dependent variable seems linear and the relation between the independent variables seems almost non-linear.

**Methodology: -**

Data is cleaned, analyze the probability of admits on over all observations, over-fitting, find logistic regression model with train data and predict the values with test data. Find the best model and predict with k -fold cross validation after reducing the errors.

**Test and Train: -**

Data of 1100 observations is distributed in the ratio of 70:30 train and test respectively. Train data contains 770 observations and test data contains 330 observations.

**Logistic Regression model: -**

Conducting logistic regression using train data (70% of all observations). The data is divided into train and test of 0.7:0.3 of total observations. Logistic Regression model is performed using train data and this model is used to predict using test data.

Model with all the variables

Logit (Admission Probability) = log (Admission probability/ (1-Admission probability)) =

-70.59451 + (0.02823 x GRE SCORE) + (0.08071 x TOEFL SCORE) + (13.35869x University Rating 2) + (13.72304 x University Rating 3) + (14.21112 x University Rating 4) + (15.20100 x University Rating 5) + (0.75329 x SOP) + (0.30376 x LOR) + (4.03985 x CGPA) + (0.94784 x Research 1)

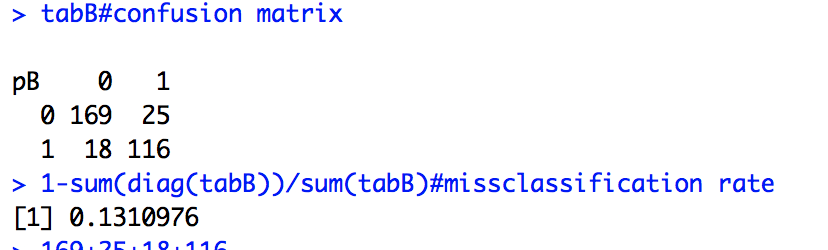


Figure 4-confusion matrix and misclassification rate

Model with overfitting- model2

The significance level of university rating variable and GRE score is high. After removing those variables, model-2 is developed with less misclassification error.

Logit (Admission Probability) =log (Admission probability/ (1-Admission probability))

-74.05221 + (0.09957 x TOEFL Score) + (0.91334 x SOP) + (0.40664 x LOR) + (4.12341 x CGPA) + (0.75205 x Research1)

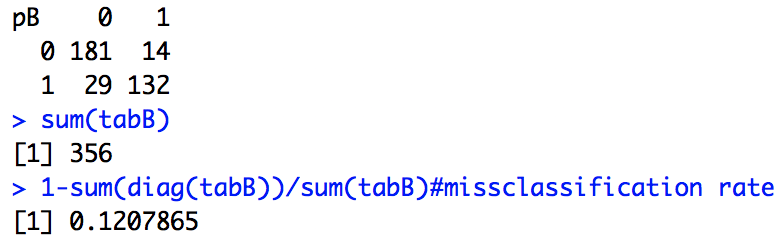


Figure 5-confusion matrix and misclassification after over fitting

The misclassification rate after overfitting is less when compared with all the variables. figure -8 shows the first five predictions. But these predictions show the results with default value 0.5 as cutoff. As our data is not distributed linear for all we need to find cutoff value.

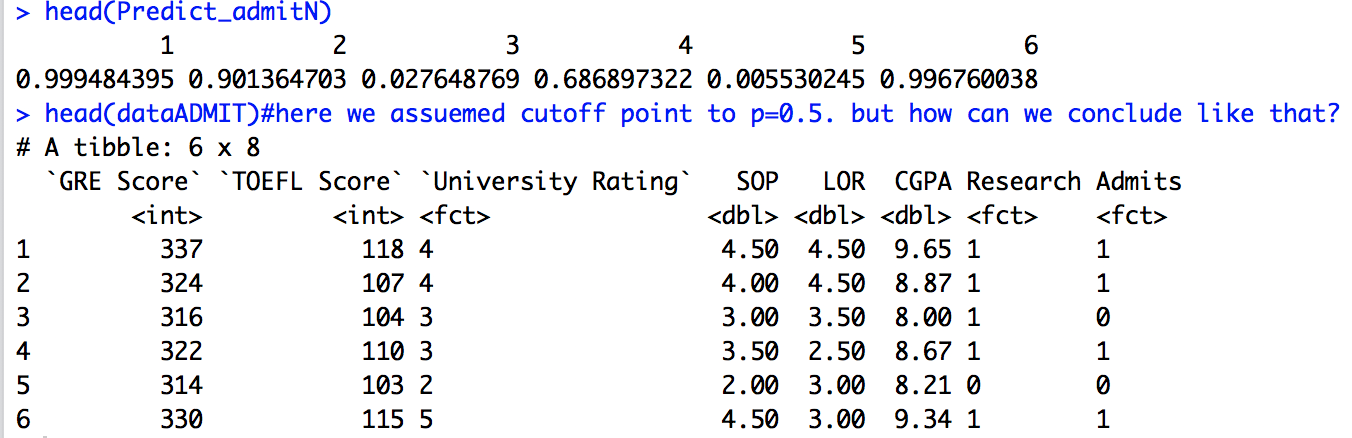


Figure 6- predicting using model-2

**Histogram: -**

Histogram shows how our data is spread from maximum to minimum values. A histogram is an accurate representation of the distribution of numerical data. It is an estimate of the probability distribution of a continuous variable.

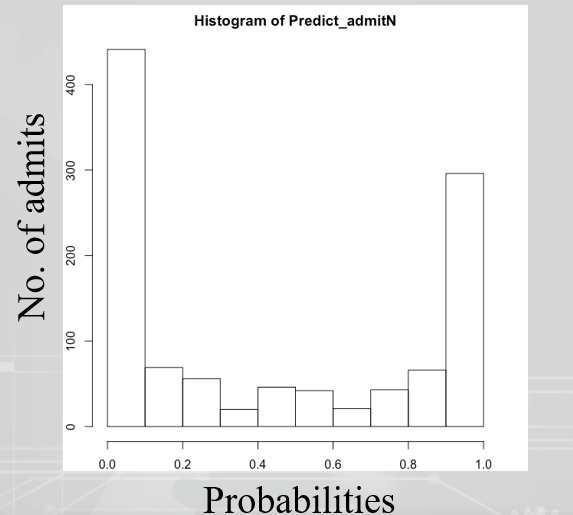
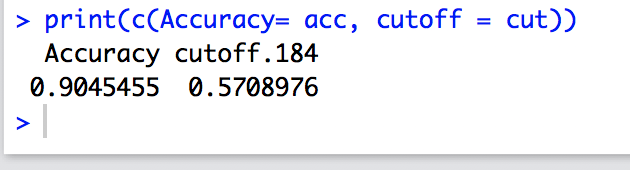


Figure 7 Histogram

**ROC Curve: -**

The graph below shows three ROC curves representing excellent, good, and worthless tests plotted on the same graph. The accuracy of the test depends on how well the test separates the group being tested into those with and without the disease in question. Accuracy is measured by the area under the ROC curve.

0.57089 is maximum cutoff value with accuracy of 0.90454



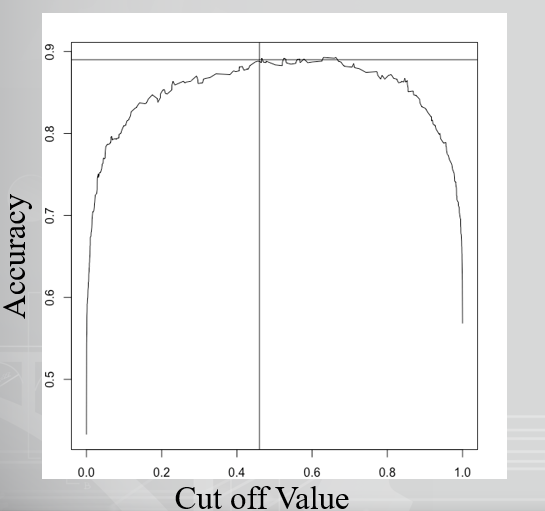


Figure 8 ROC Curve

In logistic regression as result will be in probabilities, we need to find cutoff value.

**AUC Curve: -**

AUC curve also known as Area Under the ROC Curve which is used to determine the accuracy of a quantitative diagnostic test. From the curve below we can see that curve-1 is plotted based on overfitted model with 0.9617 as AUC and curve-2 with AUC of 0.9603 is plotted based on the model developed by all the variable.

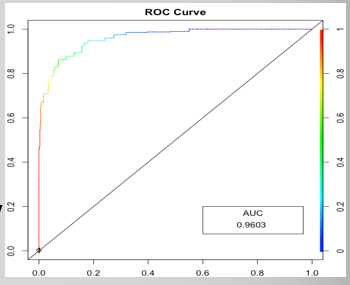
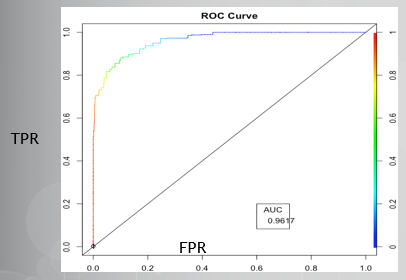
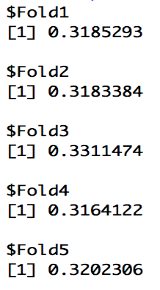


Figure 9 AUC Curve (1) Figure 10 AUC Curve (2)

**Cross-Validation: -**

Now, the logistics model is tested using cross validation for its efficiency. For this, k- fold is used to perform the cross- validation. The data is divided into train and test of 0.7:0.3 of total observations. The train data is used to form the k- fold validation and this model is used to predict using test data.

K- fold is performed for cross validation

K= N/(N\*no.of test data).

Where k represents number of folds and N represents total number of observations.

K= (1100/(1100\*0.30))=3.33 so performing 5 folds.

Figure 11-RMSE values

Among all the RMSE values fold5 has the lowest. so, let’s consider fold 5 for model and prediction.

The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data–how close the observed data points are to the model's predicted values. Whereas R-squared is a relative measure of fit, RMSE is an absolute measure of fit. so, consider RMSE with less value.

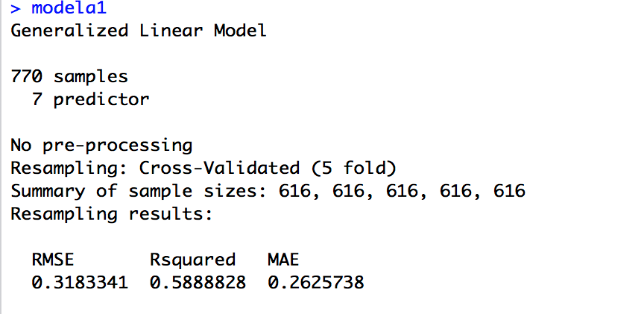
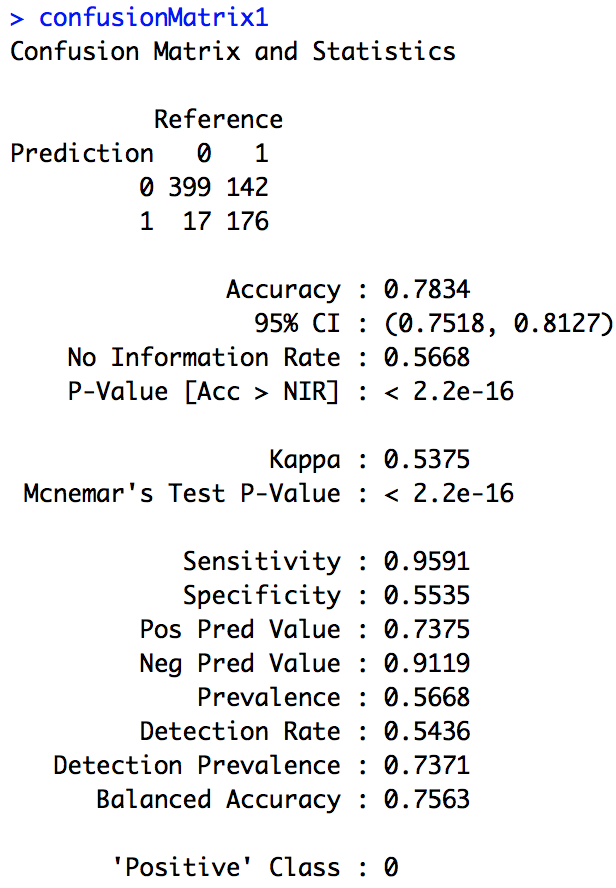


Figure 12- How each fold is divided

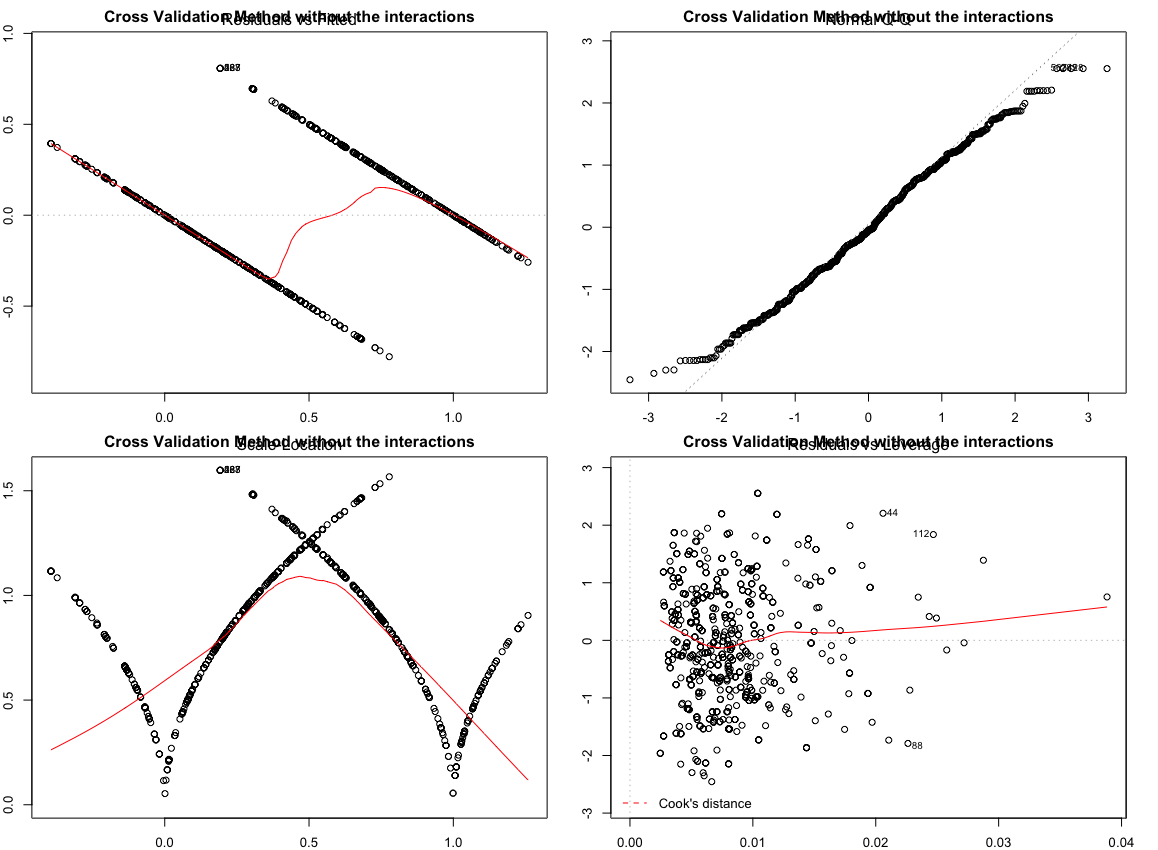


Figure 13- scatter plot of cross validation

**Results:-**

* Probability of admits = 0.433, Probability of Research = 0.526
* Miss classification rate as 0.1207
* Based on missclassification, P value, Significance of each independent variable, their realtionship as discussed using scatter plot we decided to use only TOEFL SCORE,LOR, SOP, CGPA, Research as independent variables.
* In ROC curve, the accuracy is 0.9045 and cutoff is 0.5708.
* AUC as 0.9617.
* In cross validation best model is fold-05 with RMSE value of 0.3101 with TOEFL SCORE,LOR, SOP, CGPA, Research as independent variables. With accuracy rate of 0.9001.

**Conclusion:-**

* Students with the probability greater than 0.5708 will get admit in university with an accuracy rate of 0.90.
* Model 2 can be used to know whether they can get admit or not with accurate of 90%.

R- output- final results of perdiction

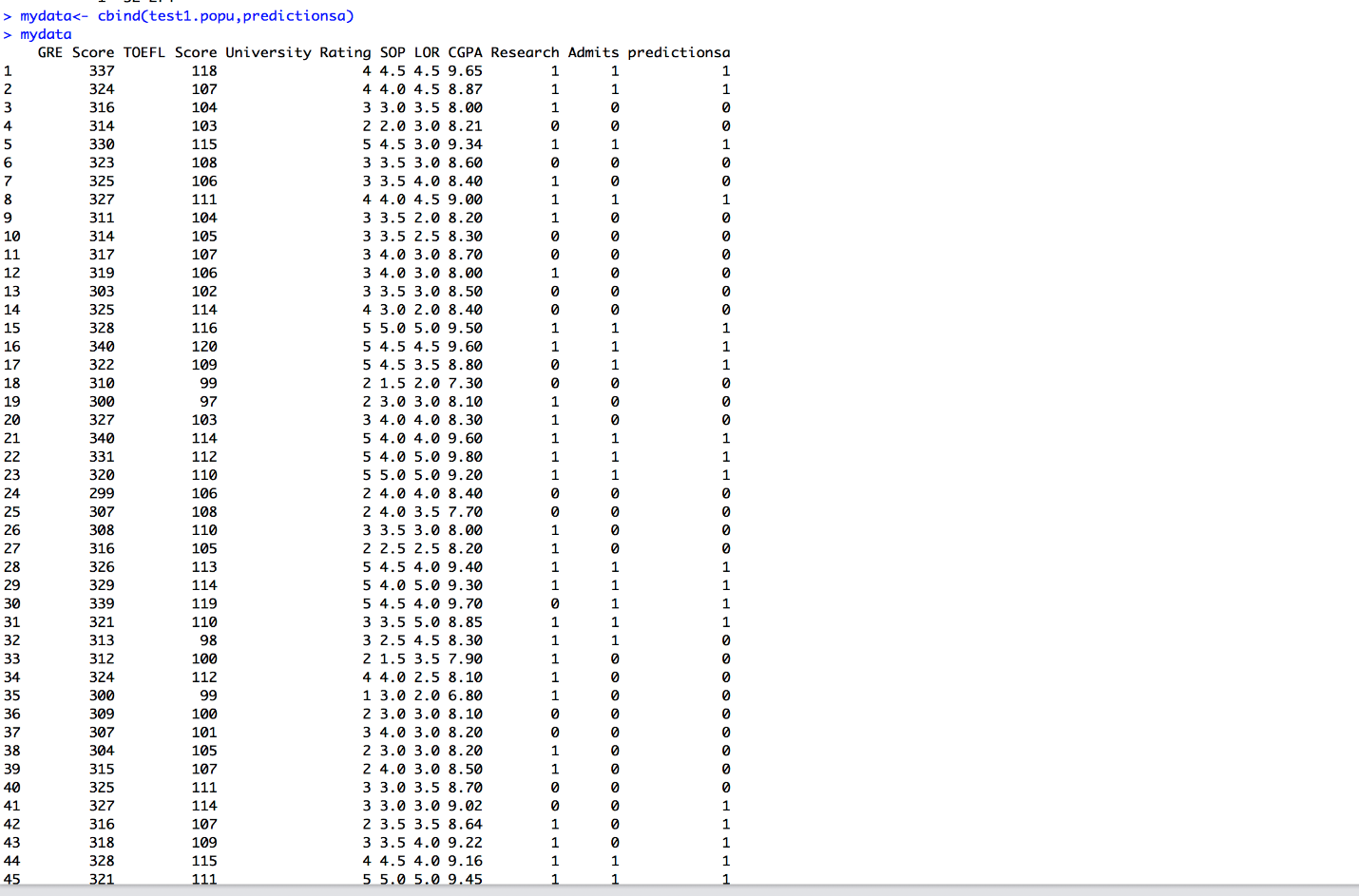
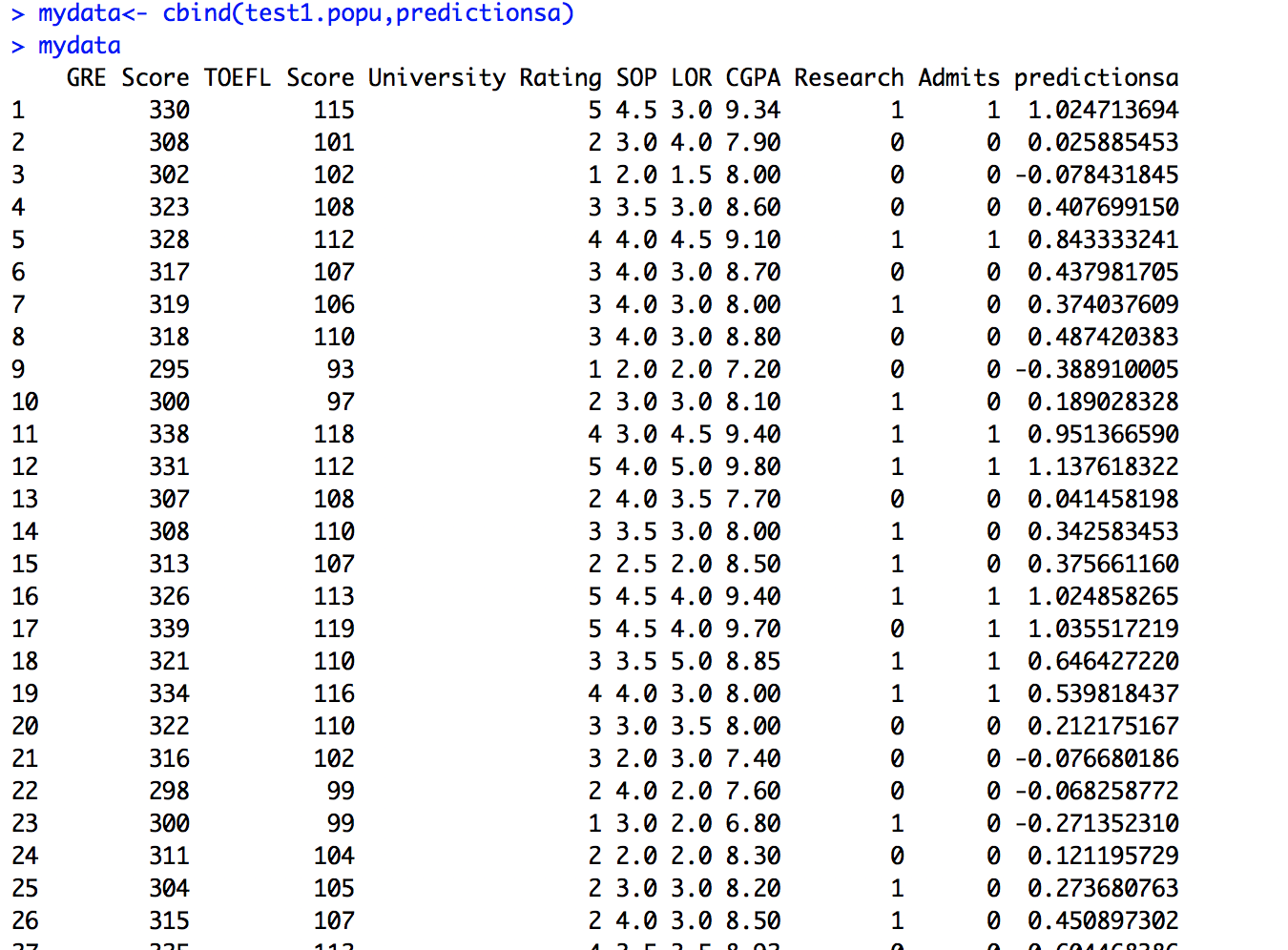


Figure 15- results between prediction and actual values

In figure 14 the predicted values are in factor class.

In figure 15 the predicted values are in numeric class.



**Reference: -**

* <https://www.r-bloggers.com/illustrated-guide-to-roc-and-auc/>
* <https://www.kaggle.com/mohansacharya/graduate-admissions/data>
* https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/

**Appendix: -**

**R- code**

#setting up the data

View(Admission\_Predict)

head(Admission\_Predict)

summary(Admission\_Predict)

length(Admission\_Predict$SOP)

dataADMIT<-Admission\_Predict

str(dataADMIT)

#1.data cleaning

dataADMIT$Research <- as.factor(dataADMIT$Research)

dataADMIT$Admits <- as.factor(dataADMIT$Admits)

dataADMIT$`University Rating` <- as.factor(dataADMIT$`University Rating`)

#replace empty spaces with NA

Admission\_Predict[Admission\_Predict==""] <- NA

#check for missing values using sapply() fun

sapply(Admission\_Predict, function(x) sum(is.na(x)))#is.na checks each cell

#colSums(is.na(train))#does same as above

#DATA ANALYSING

#few basic ratios from data

total\_admits = sum(dataADMIT$Admits == 1)/length(dataADMIT$Admits)#tell probability of admits

cat("probability of admits" , format(total\_admits, digits =3))

total\_Research = sum(dataADMIT$Research == 1)/length(dataADMIT$Research)#tell probability of Researchers

cat("probability of Research" , format(total\_Research, digits =3))

#drop columns we are not using

dataADMIT1 <- subset(Admission\_Predict, select = c(1,2,4,5,6,7,8))

#2.train and test data

ind.Admit <- sample(2, nrow(dataADMIT), replace = TRUE, prob = c(0.7,0.3))

train.popu <- dataADMIT[ind.Admit==1,]

test.popu <- dataADMIT[ind.Admit==2,]

set.seed(1234)

#ind.Admit<-createDataPartition(dataADMIT$Admits, p=2/3,list = FALSE)

#train1.popu <- dataADMIT[-ind.Admit,]

#test1.popu <- dataADMIT[ind.Admit,]

head(train.popu)

head(test.popu)

#check for strongly correlated variable

pairs(dataADMIT1[,1:7], col= dataADMIT1$`Admits`)

plot()

# 3. Ordinal Logitic Regression or Proportional Odds Logistic Regression

library(MASS,quietly = TRUE)

attach(train.popu)

modeladmi1 <- glm(`Admits`~ `TOEFL Score`+SOP+CGPA+Research+LOR, train.popu, family ='binomial')

summary(modeladmi1)

modeladmiall <- glm(`Admits`~ `TOEFL Score`+SOP+CGPA+Research+LOR, train.popu, family ='binomial')

summary(modeladmiall)

#4.prediction

res<- predict(modeladmi1,test.popu,type= "response")

res

table1<-table(actualvalue =test.popu$Admits,predictvalue = res>0.5342)#confusion matrix

1-sum(diag(table1))/sum(table1)#missclassification

#5. confusion matrix and error for test data

#predTest<- predict(modeladmi1,train.popu)

#predTest

#tabtest<- table(predTest,train.popu$Admits)#confusion matrix with train

#tabtest

#sum(diag(tabtest))/sum(tabtest)#compare this with train

#whether student is admited or not

#model performance evaluation

#logistic regression model(multinoimnal)

library(nnet)

modeladmit<- multinom(`Admits`~.-`GRE Score`-`University Rating`, data=train.popu )

#prediction

predTrain<- predict(modeladmit,train.popu, type= 'prob')#prob for each and everyone

print(predTrain, digits = 3)

#confusion matrix for train data

tabAdmit<- table(predTrain,train.popu$Admits)

#tabAdmit

#1-sum(diag(tabAdmit))/sum(tabAdmit)#missclassification rate

#missclassification rate for all data

pB<- predict(modeladmit, test.popu)

tabB<- table(pB, test.popu$`Admits`)

tabB#confusion matrix

1-sum(diag(tabB))/sum(tabB)#missclassification rate

#2-tailed z test

z <- summary(modeladmit)$coefficients/summary(modeladmit)$standard.errors

p <- (1 - pnorm(abs(z), 0, 1)) \* 2

p

#for better prediction- using ROCR-----------------

install.packages("ROCR")

library(ROCR)

Predict\_admitN<- predict(modeladmit, dataADMIT, type = 'prob')

head(Predict\_admitN)

head(dataADMIT)#here we assuemed cutoff point to p=0.5. but how can we conclude like that?

#find cutoff probability point

hist(Predict\_admitN)

Predict\_admitN<- prediction(Predict\_admitN, dataADMIT$`Admits`)

evaluation<- performance(Predict\_admitN, "acc")

plot(evaluation)

abline(h=0.89,v= 0.46)

#identify best value

evaluation

max<- which.max(slot(evaluation, "y.values")[[1]])

acc<- slot(evaluation, "y.values")[[1]][max]

cut<-slot(evaluation, "x.values")[[1]][max]

print(c(Accuracy= acc, cutoff = cut))

#prediction using ROC curve---------------

Predict\_admitN<- predict(modeladmit, dataADMIT, type = 'prob')

Predict\_admitN <- prediction(Predict\_admitN, dataADMIT$Admits)

rocAdmit<-performance(Predict\_admitN, "tpr","fpr")

rocAdmit

plot(rocAdmit,

colorize=T,print.cutoffs.at=seq(0.1),by=0.1,

main="ROC Curve",

ylab= "sensitivity",

xlab= "1-sensitivity")

abline(a=0, b=1)

#calc. are under the curve(AUC) as area is more the performance is good

auc<-performance(Predict\_admitN, "auc")

auc<-unlist(slot(auc,"y.values"))

auc <- round(auc,4)

legend(0.6,.2,auc,title="AUC")

# load libraries

library(caret)

library(rpart)

# load the dataset

View(Admission\_Predict)

df<-Admission\_Predict

# CLEAN DATA

str(df)

df$Research <- as.factor(df$Research)

df$Admits <- as.factors(df$Admits)

df$`University Rating` <- as.factor(df$`University Rating`)

#data partition

set.seed(1234)

ind.Admit<-createDataPartition(df$Admits, p=c(0.7,0.3),list = FALSE)

train1.popu <- df[ind.Admit,]

test1.popu <- df[-ind.Admit,]

head(train1.popu)

head(test1.popu)

# cross validation- train

train\_control<- trainControl(method="cv", number=5, savePredictions = TRUE)

train\_control

# train the model - fit navie bayes model

parametergrid<- expand.grid(mtry=c(1,2,3,4,5,6,7))

parametergrid

modela<- train(Admits~.-`GRE Score`-`University Rating`, data=train1.popu, trControl=train\_control, method="glm",tuneGrid = parametergrid)

modela

modela1<- train(Admits~.-`GRE Score`-`University Rating`, data=train1.popu, trControl=train\_control, method="glm")

modela1

#`GRE Score`+`TOEFL Score`+`University Rating`+SOP+CGPA+Research+LOR

#PREDICTION OF MODEL FROM TRAIN DATA WITH TEST DATA AND TRAIN DATA

# make predictions with test data

predictionsa<- predict(modela1,test1.popu)

head(predictionsa)

tabtestcross<- table(predictionsa,test1.popu$Admits)

tabtestcross

# append predictions

mydata<- cbind(test1.popu,predictionsa)

mydata

# summarize results

confusionMatrix1<- confusionMatrix(mydata$predictionsa,test1.popu$Admits)

confusionMatrix1

#////////////////////////////////////////////////////

# make predictions with train data

predictionsa<- predict(modela1,train1.popu)

head(predictionsa)

tabtestcross<- table(predictionsa,train1.popu$Admits)

tabtestcross

# append predictions

mydata<- cbind(train1.popu,predictionsa)

mydata

# summarize results

#confusionMatrix1<- confusionMatrix(mydata$predictionsa,train1.popu$Admits)

#confusionMatrix1

#///////////////////////////////////////////////

#cross-validation- kfold

folds\_simple\_modelmy = createFolds(df$Admits, 5)

folds\_simple\_modelmy

df$Research <- as.numeric(df$Research)

df$Admits <- as.numeric(df$Admits)

df$`University Rating` <- as.numeric(df$`University Rating`)

cv\_simple\_modelmy = lapply(folds\_simple\_modelmy, function(x)

{

trainCV = df[-x,]

validCV = df[x,]

modelCV = glm(Admits~.-`GRE Score`-`University Rating`, data = trainCV)

predict.cv = predict(modelCV, validCV)

rmse\_simple\_cv = RMSE(validCV$Admits, predict.cv)

return(rmse\_simple\_cv)

})

cv\_simple\_modelmy

##As we can seee fold numer 05 has less error therefore we will use fold01 to train model

trainCV\_1 = df[-folds\_simple\_modelmy$Fold1,]

validCV\_1 = df[folds\_simple\_modelmy$Fold1,]

modelCV\_1 = glm(Admits~-`GRE Score`-`University Rating`, data = trainCV\_1)

predictCV\_1 = predict(modelCV\_1,validCV\_1)

rmseCV\_1 = RMSE(validCV\_1$Admits, predictCV\_1)

rmseCV\_1

summary(modelCV\_1)

#coeefffcient(modelCV\_1)

#confint(modelCV\_1)

par(mfrow = c(2,2))

#x<-plot(modelCV\_1, main = "Cross Validation Method without the interactions")

## Now we will make another models with another interactions

#Model with interaction-1

MM = glm(Admits~.-`GRE Score`-`University Rating`, data = train1.popu)

summary(MM)

#Prediction on model1

predict.interactions1 = predict(MM,test1.popu)

#Error in model1

rmse\_MM =RMSE(test1.popu$Admits, predict.interactions1)

rmse MM

mydata