########################### CHURN PREDICTION ##########################

library(devtools)

library(woe)

data=read.csv(file.choose()) **# To read the Churn csv file from the system**

iv.mult(data,"Churn",TRUE) **# To get the Information value of all variables**

str(data) **#Summary of data**

table(data$Churn) **#table showing no. of 1s and 0s in data**

index<-sample(1:nrow(data),0.8\*nrow(data)) **# Sampling 80% data for train\_data**

train\_data<-data[index,-21] **# Storing to train\_data**

test\_data<-data[-index,-c(8,21)] **#Storing the rest 20% to test\_data**

str(train\_data) **#Summary of train\_data**

str(test\_data) **#Summary of test\_data**

glm\_model<-glm(Churn~.,family=binomial,data=train\_data)#Creating the regressin model

glm\_model

pred<-predict(glm\_model,test\_data,type = "response")#Predicting on testdata using model

pred

hist(pred) **#Graphical representations**

plot(data$Churn[-index]~pred)

gg=floor(pred+0.5) **# Using floor operator to make probabilities to 1s & 0s...**

gg **# by using 0.5 cutoff**

table(gg) **# displays no. of 1s and 0s.**

ttt=table(data$Churn[-index],gg)

ttt  **#Original Outcome vs Prediction**

**########################## Overcoming Imbalanced dataset ###################**

data\_1<-data[data$Churn==1,]

ind\_1<-sample(rownames(data\_1),483)

data\_0<-data[data$Churn==0,]

ind\_0<-sample(rownames(data\_0),483)

train\_data1<-data[c(ind\_1,ind\_0), -21 ] **#Creating train\_data with equal sample of**

**# 1 & 0 churns to make balanced traindata**

str(train\_data1) **#Summary of train\_data1**

table(train\_data1$Churn) #**Table showing equal no of value in 1s and 0s of traindata**

glm\_model1<-glm(Churn~.,family=binomial,train\_data1)#Creating the regressin model

glm\_model1

pred1<-predict(glm\_model1,test\_data,type = "response")# Prediction using the NEW model

pred1

hist(pred1) **# Graphical representations**

plot(data$Churn[-index]-pred1)

gg1=floor(pred1+0.5) **# Using floor operator to make probabilities to 1s & 0s...**

gg1 **# by using 0.5 cutoff**

table(gg1) **# displays no. of 1s and 0s.**

ttt1=table(data$Churn[-index],gg1)

ttt1 **#Original Outcome vs Prediction for balanced dataset**

exp(cbind(Odds\_and\_OR=coef(glm\_model1),confint(glm\_model1)))

**# Odds ratio of all variables**

**############################ Accuracy measures #############################**

acc<-(ttt1[1]+ttt1[4])/(ttt1[1]+ttt1[3]+ttt1[2]+ttt1[4]) # Accuracy=tp+tn/(tp+fp+tn+fn)

acc

library(caret)

confusionMatrix(ttt1) **#shows some accuracy measures**

sens=ttt1[1]/(ttt1[1]+ttt1[3]) **# Sensitivity=tp/tp+fn**

sens

spec=ttt1[4]/(ttt1[4]+ttt1[2]) **# Specificity=tn/tn+fp**

spec

**######################## Visualizing Performance Tradeoffs ###################**

install.packages("ROCR")

library(ROCR) **#package for ROC Curve**

pred5<-predict(glm\_model1,train\_data1,type = "response")

pred2<-prediction(predictions = pred5,labels = train\_data1$Churn )

pred2

eval<- performance(pred2,measure = "tpr", x.measure = "fpr")

plot(eval,colorize=T, main="ROC curve",col="blue",lwd=5) **# Plotting ROC curve for...**

**# different cutoffs colorized accordingly**.

auc<-performance(pred2,measure = "auc")

**# Finding accuracy measure- AREA UNDER CURVE(AUC)**

str(auc) **# Summary of auc**

as.numeric(auc@y.values) **#Printing auc value**

**###################### Optimum cutoff & Maximum accuracy ##################**

eval1=performance(pred2,measure = "acc")

plot(eval1) **# plots accuracy vs cutoff**

max<- which.max(slot(eval1,"y.values")[[1]])

acc<- slot(eval1,"y.values")[[1]][max] **#Identifying maximum accuracy &**

cut<- slot(eval1,"x.values")[[1]][max] **#its respective cutoff from the graph**

print(c(Accuracy=acc, Cutoff=cut)) **# printing the max. cutoff & Accuracy**

**######################### Connecting R to Tableau ###########################**

install.packages("Rserve")

library(Rserve)

Rserve()