##ABOUT THE DATASET###

##The data is related with direct marketing campaigns of a Portuguese banking institution.

#The marketing campaigns were based on phone calls. Often, more than one contact

#to the same client was required, in order to access if the product

#(bank term deposit) would be ('yes') or not ('no') subscribed.

#Dependent Variable: To predict if client will subscribe to the bank term deposit i.e. yes (1)

#Variables: From 2 to 15 are categorical variables rest are numeric

#Unbalanced dataset

install.packages("caTools")

install.packages("ROCR")

install.packages("gains")

install.packages("rpart")

install.packages("rpart.plot")

install.packages("randomForest")

install.packages("caret")

install.packages("ggplot2")

install.packages("e1071")

install.packages("corrplot")

install.packages("nnet")

library(Rcpp)

library(mlbench)

library(caret)

library(forecast)

library(corrplot)

library(ggplot2)

getwd()

###1. Loading the dataset

data = read.csv("winequality-red.csv")

nrow(data)

str(data)

table(data$quality)

#baseline = 1382/(1382+217) #86.42%

#Data Exploration - Check Summmary and Examine missing data

summary(data)

hist(data$fixed.acidity) #Highly Skewed

#Check correlations - An indication of multicollinearity

str(data)

correlations = cor(data[c(1:11)])

heatmap(round(correlations,2))

corrplot(correlations, method = "color", type="upper", order="hclust", addCoef.col = "black")

## Exploratory Data Analysis

table(data$quality)

par(mfrow=c(1,1))

hist(data$fixed.acidity)

boxplot(data$fixed.acidity, horizontal = TRUE)

hist(data$volatile.acidity)

boxplot(data$volatile.acidity, horizontal = TRUE)

hist(data$citric.acid)

boxplot(data$citric.acid, horizontal = TRUE)

hist(data$residual.sugar)

boxplot(data$residual.sugar, horizontal = TRUE)

hist(data$chlorides)

boxplot(data$chlorides, horizontal = TRUE)

hist(data$free.sulfur.dioxide)

boxplot(data$free.sulfur.dioxide, horizontal = TRUE)

hist(data$total.sulfur.dioxide)

boxplot(data$total.sulfur.dioxide, horizontal = TRUE)

hist(data$density)

boxplot(data$density, horizontal = TRUE)

hist(data$pH)

boxplot(data$pH, horizontal = TRUE)

hist(data$sulphates)

boxplot(data$sulphates, horizontal = TRUE)

hist(data$alcohol)

boxplot(data$alcohol, horizontal = TRUE)

#Pre Processing

#------------OUTLIER Treatment & Normalization-------------------- (Method Used = Capping {L:5%ile, O:95%ile})

cap\_outlier <- function(d) {

x <- d

qnt <- quantile(x, probs=c(.25, .75), na.rm = T)

caps <- quantile(x, probs=c(.05, .95), na.rm = T)

H <- 1.5 \* IQR(x, na.rm = T)

x[x < (qnt[1] - H)] <- caps[1]

x[x > (qnt[2] + H)] <- caps[2]

return(x)

}

data$fixed.acidity = cap\_outlier(data$fixed.acidity)

data$volatile.acidity = log(cap\_outlier(data$volatile.acidity))

data$citric.acid = log(cap\_outlier(data$citric.acid)+1)

data$residual.sugar = log(cap\_outlier(data$residual.sugar))

data$chlorides = log(cap\_outlier(data$chlorides))

data$free.sulfur.dioxide = log(cap\_outlier(data$free.sulfur.dioxide))

data$total.sulfur.dioxide = log(cap\_outlier(data$total.sulfur.dioxide))

data$density = log(cap\_outlier(data$density))

data$pH = cap\_outlier(data$pH)

data$sulphates = log(cap\_outlier(data$sulphates))

#-------------Encoding into Good(1) and Bad(0)--------------

data$quality = as.factor(ifelse(data$quality >= 7, 1, 0))

### Scaled Split of Data

split<- sample.split(data$quality,SplitRatio=0.7)

train= subset(data,split==TRUE)

test= subset(data,split==FALSE)

str(train)

str(test)

table(train$quality)

baseline.train = 967/(967+152) #86.41

table(test$quality)

###3. Logistic Regression

fit\_lr.train = glm(quality ~ ., data=train, family=binomial) #Fit Logistic Regression Model

summary(fit\_lr.train) #summarize the fit

options(scipen=999)

fit\_lr.train$fitted.values #Check fitted values

table(train$quality, fit\_lr.train$fitted.values>0.1) #Check Accuracy - 91.19%

fit\_lr.test = predict(fit\_lr.train, type="response", newdata=test) #Make Predictions

table(test$quality, fit\_lr.test > 0.087) #Check Accuracy on Test data - 91.23%

#Remove highly correlated variables and run logistic regression again

fit\_lr.train = glm(quality ~ . - fixed.acidity, data=train, family=binomial)

summary(fit\_lr.train)

fit\_lr.train$fitted.values

table(train$quality, fit\_lr.train$fitted.values>0.8) #Check Accuracy - 91.10%

fit\_lr.test = predict(fit\_lr.train, type="response", newdata=test) #Make Predictions

table(test$quality, fit\_lr.test > 0.1) #Check Accuracy on Test data - 91.17%

liftdata = cbind(train$quality, fit\_lr.train$fitted.values)

write.csv(liftdata, "wine\_liftdata.csv")

#Change cut-offs to plot ROCR Curve

table(train$quality, fit\_lr.train$fitted.values>0.1)

table(train$quality, fit\_lr.train$fitted.values>0.2)

table(train$quality, fit\_lr.train$fitted.values>0.3)

table(train$quality, fit\_lr.train$fitted.values>0.4)

table(train$quality, fit\_lr.train$fitted.values>0.5)

table(train$quality, fit\_lr.train$fitted.values>0.6)

table(train$quality, fit\_lr.train$fitted.values>0.7)

table(train$quality, fit\_lr.train$fitted.values>0.8)

table(train$quality, fit\_lr.train$fitted.values>0.9)

table(train$quality, fit\_lr.train$fitted.values>0.08762868) #Optimal cut-off

###4. Classsification Trees

#library(rpart)

#library(rpart.plot)

#library(caret)

#fit\_ct.train = rpart(quality ~ ., data=train, method ="class", control = rpart.control(maxdepth = 10, xval = 5))

#prp(fit\_ct.train)

#fit\_ct.train.pred = predict(fit\_ct.train, data = train, type="class")

#confusionMatrix(fit\_ct.train.pred, train$quality)

#fit\_ct.test.pred = predict(fit\_ct.train, newdata = test, type="class")

#confusionMatrix(fit\_ct.test.pred, test$quality)

#Improving Accuracy using Random Forest Modeling - Make Sure that Outcome Variable is a Factor for conducting Random Forests

library(randomForest)

str(train)

fit\_rf.train = randomForest(quality ~ ., data=train, ntree=100)

fit\_rf.train.pred = predict(fit\_rf.train, data = train)

confusionMatrix(train$quality, fit\_rf.train.pred) #91.26%

fit\_rf.test.pred = predict(fit\_rf.train, newdata = test)

confusionMatrix(test$quality, fit\_rf.test.pred) #91.55%

###6. Judging Classification Performance: ROCR Curve

library(ROCR)

best\_model = fit\_lr.test

ROCRpred = prediction(best\_model, test$quality)

ROCRperf = performance(ROCRpred, "tpr","fpr")

plot(ROCRperf, col="black", lty=2, lwd=1)

plot(ROCRperf, col=rainbow(4))

lines(c(0,1),c(0,1))

as.numeric(performance(ROCRpred, "auc")@y.values) #88.92%

#Determining Optimal cut-off - Given by Youden's Index = max(sensitivity+specificity-1) = max(tpr-fpr)

str(ROCRperf)

ROCRperf@alpha.values[[1]]

ROCRperf@x.values[[1]]

ROCRperf@y.values[[1]]

difference = ROCRperf@y.values[[1]] - ROCRperf@x.values[[1]]

youden = max(difference)

youdendata = as.data.frame(cbind(alpha=ROCRperf@alpha.values[[1]],difference))

which.max(youdendata$difference) #3200

youdendata[3200,]

write.csv(youdendata,"youdendata.csv")

###7. Judging Ranking Performance: Building the Lift Chart

library(gains)

best\_model = fit\_lr.test

gain = gains(as.integer(test$y)-1, best\_model) #Subtracted 1 as converting factor variable into integer converts it into 1 and 2 rather than 0 and 1

plot(gain$cume.lift)

plot(gain$cume.obs)

plot(c(0,gain$cume.pct.of.total\*sum(as.numeric(test$y)-1))~c(0,gain$cume.obs),

xlab="# cases", ylab="Cumulative", main="", type="l")

lines(c(0,sum(as.integer(test$y)-1))~c(0, dim(test)[1]), lty=2)

str(test)

#Plot lift Chart - Alternate Code

lift = as.data.frame(cbind("Actual"=test$y, "Predicted"= best\_model))

lift = lift[order(-lift$Predicted),] # - is added for ordering in descending order

lift$cumsum = cumsum(lift$Actual-1) #Subtracted 1 so that 1 and 2 are taken as 0 and 1

plot(lift$cumsum, type="l", lty=1)

lines(c(0,sum(as.integer(test$y)-1))~c(0, dim(test)[1]), lty=2)

write.csv(lift,"liftdata.csv")