**SOURCE CODE IN R\_ONLINE PURCHASE INTENTION**

**GROUP 3-MGSC-5126-12**

**#Data Cleansing & Manipulation**

my\_data<-read.csv(url("https://archive.ics.uci.edu/ml/machine-learning-databases/00468/online\_shoppers\_intention.csv"))

str(my\_data)

install.packages("plyr")

library(plyr)

my\_data[,12:15] <- lapply(my\_data[,12:15] , factor)

my\_data$VisitorType = factor(my\_data$VisitorType, levels = c('New\_Visitor','Returning\_Visitor','Other'), labels = c(1,2,3))

my\_data$Month = factor(my\_data$Month, levels = c('Jan','Feb','Mar','Apr','May','June','Jul','Aug','Sep','Oct','Nov','Dec'), labels = c(1,2,3,4,5,6,7,8,9,10,11,12))

my\_data$Weekend = factor(my\_data$Weekend, levels = c('TRUE','FALSE'), labels = c(1,2))

my\_data$Revenue = factor(my\_data$Revenue, levels = c('TRUE','FALSE'), labels = c(1,2))

str(my\_data)

my\_data <-my\_data[!(my\_data $VisitorType=='3'),]

head(my\_data)

install.packages("Amelia")

library(Amelia)

table(is.na (my\_data))

**#Data Normalization**

summary(my\_data)

quantile(my\_data$Administrative\_Duration)

quantile(my\_data$Informational\_Duration)

quantile(my\_data$ProductRelated\_Duration)

quantile(my\_data$BounceRates)

quantile(my\_data$ExitRates)

quantile(my\_data$PageValues)

normalize <- function(x) {(x - min(x, na.rm=TRUE))/(max(x,na.rm=TRUE) -min(x, na.rm=TRUE))}

install.packages('dplyr')

library(dplyr)

my\_data\_norm<-my\_data%>%mutate\_if(is.numeric, normalize)

lapply(my\_data\_norm[,c(1:10)], range)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#Data Visualisation – Scatter plot-Correlation**

plot(my\_data\_norm$BounceRates,my\_data\_norm$PageValues,main="PageValues vs BounceRates of a WebPage", ylab="Page Values",xlab="Bounce Rates",pch=18,col=c('light pink'))

attach(my\_data\_norm)

lines(lowess(PageValues,BounceRates), col="red")

X<-my\_data\_norm$BounceRates

Y<-my\_data\_norm$PageValues

cor(X,Y,method='pearson')

plot(my\_data\_norm$ExitRates,my\_data\_norm$PageValues,main="PageValues vs ExitRates of a WebPage", ylab="Page Values",xlab="Exit Rates",pch=18,col=c('grey'))

attach(my\_data\_norm)

lines(lowess(PageValues,ExitRates), col="blue")

X1<-my\_data\_norm$ExitRates

Y<-my\_data\_norm$PageValues

cor(X1,Y,method='pearson')

plot(my\_data\_norm$SpecialDay,my\_data\_norm$PageValues,main="PageValues vs SpecialDay", ylab="Page Values",xlab="SpecialDay",pch=18,col=c('light green'))

attach(my\_data\_norm)

SD<-my\_data\_norm$SpecialDay

Y<-my\_data\_norm$PageValues

cor(SD,Y,method='pearson')

**#Splitting the data**

install.packages("caTools")

library(caTools)

install.packages("caTools")

library(caTools)

Split\_Data\_0.7 <- sample.split(my\_data\_norm$Revenue, SplitRatio = 0.7)

train\_data\_0.7 = my\_data\_norm[Split\_Data\_0.7,]

test\_data\_0.7 = my\_data\_norm[!Split\_Data\_0.7,]

dim(train\_data\_0.7)

dim(test\_data\_0.7)

Split\_Data\_0.8 <- sample.split(my\_data\_norm$Revenue, SplitRatio = 0.8)

train\_data\_0.8 = my\_data\_norm[Split\_Data\_0.8,]

test\_data\_0.8 = my\_data\_norm[!Split\_Data\_0.8,]

dim(train\_data\_0.8)

dim(test\_data\_0.8)

Split\_Data\_0.7 <- sample.split(my\_data\_discretized$Revenue, SplitRatio = 0.7)

train\_data\_0.7 = my\_data\_discretized[Split\_Data\_0.7,]

test\_data\_0.7 = my\_data\_discretized[!Split\_Data\_0.7,]

dim(train\_data\_0.7)

dim(test\_data\_0.7)

Split\_Data\_0.8 <- sample.split(my\_data\_discretized$Revenue, SplitRatio = 0.8)

train\_data\_0.8 = my\_data\_discretized[Split\_Data\_0.8,]

test\_data\_0.8 = my\_data\_discretized[!Split\_Data\_0.8,]

dim(train\_data\_0.8)

dim(test\_data\_0.8)

**#Creating Decision Trees**

install.packages("party")

library("party")

Train\_tree\_0.7 = ctree(formula=Revenue~., data = train\_data\_0.7)

print(Train\_tree\_0.7)

plot(Train\_tree\_0.7)

tree\_predict\_0.7 <- predict(Train\_tree\_0.7, newdata = test\_data\_0.7)

Train\_tree\_0.8 = ctree(formula=Revenue~., data = train\_data\_0.8)

print(Train\_tree\_0.8)

plot(Train\_tree\_0.8)

tree\_predict\_0.8 <- predict(Train\_tree\_0.8, newdata = test\_data\_0.8)

**#Generating Confusion Matrix – Decision Tree**

install.packages("caret")

library(caret)

Tree\_CM\_0.7 = confusionMatrix(tree\_predict\_0.7, test\_data\_0.7$Revenue)

print(Tree\_CM\_0.7)

Tree\_CM\_0.8 = confusionMatrix(tree\_predict\_0.8, test\_data\_0.8$Revenue)

print(Tree\_CM\_0.8)

**## KNN**

install.packages("class")

library(class)

shop = my\_data\_norm

summary(shop)

str(shop)

index\_70 <- createDataPartition(shop$Revenue, p=0.70, list=FALSE)

train\_70 <- shop[index\_70,]

test\_70 <- shop[-index\_70,]

knn\_pred70 <- knn(train\_70, test\_70, train\_70$Revenue, k=1)

confusionMatrix(table(knn\_pred70, test\_70$Revenue))

**#Generating Confusion Matrix - KNN**

index\_80 <- createDataPartition(shop$Revenue, p=0.80, list=FALSE)

train\_80 <- shop[index\_80,]

test\_80 <- shop[-index\_80,]

knn\_pred80 <- knn(train\_80, test\_80, train\_80$Revenue, k=1)

confusionMatrix(table(knn\_pred80, test\_80$Revenue))