Assignment - 2 k-NN for classification.

Avinash Ravipudi

2022-09-25

#install.packages("readr")  
library(readr)  
#install.packages("lattice")  
library(lattice)  
#install.packages("caret")  
library(caret)

## Loading required package: ggplot2

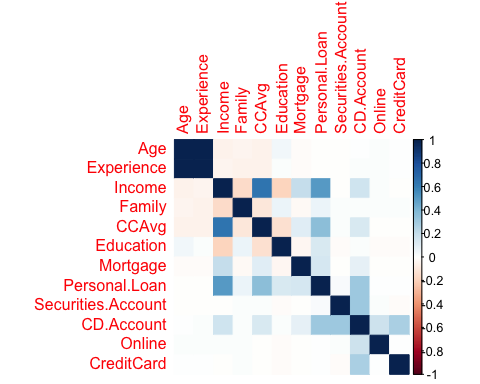
#install.packages("ISLR")  
library(ISLR)  
#install.packages("ggplot2")  
library(ggplot2)  
#install.packages("corrplot")  
library(corrplot)

## corrplot 0.92 loaded

#install.packages("fastDummies")  
library(fastDummies)  
#install.packages("FNN")  
library(FNN)  
#install.packages("plyr")  
library("plyr")  
#install.packages("gmodels")  
library(gmodels)  
#install.packages("ggplot2")  
library(ggplot2)

#Importing Data, Data visulization & Data Summary

options(stringsAsFactors = FALSE)  
UniversalBank <- read.csv("~/Desktop/FML/UniversalBank.csv")  
Universalbank\_num <-UniversalBank [, c(2:4,6:14)]   
corrplot(cor(Universalbank\_num), method="color")



summary(Universalbank\_num)

## Age Experience Income Family   
## Min. :23.00 Min. :-3.0 Min. : 8.00 Min. :1.000   
## 1st Qu.:35.00 1st Qu.:10.0 1st Qu.: 39.00 1st Qu.:1.000   
## Median :45.00 Median :20.0 Median : 64.00 Median :2.000   
## Mean :45.34 Mean :20.1 Mean : 73.77 Mean :2.396   
## 3rd Qu.:55.00 3rd Qu.:30.0 3rd Qu.: 98.00 3rd Qu.:3.000   
## Max. :67.00 Max. :43.0 Max. :224.00 Max. :4.000   
## CCAvg Education Mortgage Personal.Loan   
## Min. : 0.000 Min. :1.000 Min. : 0.0 Min. :0.000   
## 1st Qu.: 0.700 1st Qu.:1.000 1st Qu.: 0.0 1st Qu.:0.000   
## Median : 1.500 Median :2.000 Median : 0.0 Median :0.000   
## Mean : 1.938 Mean :1.881 Mean : 56.5 Mean :0.096   
## 3rd Qu.: 2.500 3rd Qu.:3.000 3rd Qu.:101.0 3rd Qu.:0.000   
## Max. :10.000 Max. :3.000 Max. :635.0 Max. :1.000   
## Securities.Account CD.Account Online CreditCard   
## Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. :0.000   
## 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.000   
## Median :0.0000 Median :0.0000 Median :1.0000 Median :0.000   
## Mean :0.1044 Mean :0.0604 Mean :0.5968 Mean :0.294   
## 3rd Qu.:0.0000 3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.000   
## Max. :1.0000 Max. :1.0000 Max. :1.0000 Max. :1.000

head(UniversalBank,10)

## ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage  
## 1 1 25 1 49 91107 4 1.6 1 0  
## 2 2 45 19 34 90089 3 1.5 1 0  
## 3 3 39 15 11 94720 1 1.0 1 0  
## 4 4 35 9 100 94112 1 2.7 2 0  
## 5 5 35 8 45 91330 4 1.0 2 0  
## 6 6 37 13 29 92121 4 0.4 2 155  
## 7 7 53 27 72 91711 2 1.5 2 0  
## 8 8 50 24 22 93943 1 0.3 3 0  
## 9 9 35 10 81 90089 3 0.6 2 104  
## 10 10 34 9 180 93023 1 8.9 3 0  
## Personal.Loan Securities.Account CD.Account Online CreditCard  
## 1 0 1 0 0 0  
## 2 0 1 0 0 0  
## 3 0 0 0 0 0  
## 4 0 0 0 0 0  
## 5 0 0 0 0 1  
## 6 0 0 0 1 0  
## 7 0 0 0 1 0  
## 8 0 0 0 0 1  
## 9 0 0 0 1 0  
## 10 1 0 0 0 0

#Convert Education to dummy variables

Universalbank\_dummy <- dummy\_cols(Universalbank\_num, select\_columns = "Education")

#Splitting data Training : 60% , Validation : 40%

set.seed(1)  
#splitting 60% of data into training & 40% of data into validation   
Train\_index <- createDataPartition(Universalbank\_dummy$'Personal.Loan', p=0.6, list=FALSE)  
Training\_data <-Universalbank\_dummy[Train\_index,]  
Validation\_data <-Universalbank\_dummy [-Train\_index,]  
summary(Training\_data)

## Age Experience Income Family   
## Min. :23.00 Min. :-3.00 Min. : 8.00 Min. :1.000   
## 1st Qu.:36.00 1st Qu.:10.00 1st Qu.: 39.00 1st Qu.:1.000   
## Median :45.00 Median :20.00 Median : 63.00 Median :2.000   
## Mean :45.43 Mean :20.19 Mean : 73.08 Mean :2.388   
## 3rd Qu.:55.00 3rd Qu.:30.00 3rd Qu.: 98.00 3rd Qu.:3.000   
## Max. :67.00 Max. :43.00 Max. :224.00 Max. :4.000   
## CCAvg Education Mortgage Personal.Loan   
## Min. : 0.000 Min. :1.00 Min. : 0.00 Min. :0.00000   
## 1st Qu.: 0.700 1st Qu.:1.00 1st Qu.: 0.00 1st Qu.:0.00000   
## Median : 1.500 Median :2.00 Median : 0.00 Median :0.00000   
## Mean : 1.915 Mean :1.88 Mean : 57.34 Mean :0.09167   
## 3rd Qu.: 2.500 3rd Qu.:3.00 3rd Qu.:102.00 3rd Qu.:0.00000   
## Max. :10.000 Max. :3.00 Max. :635.00 Max. :1.00000   
## Securities.Account CD.Account Online CreditCard   
## Min. :0.0000 Min. :0.00000 Min. :0.0000 Min. :0.0000   
## 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:0.0000 1st Qu.:0.0000   
## Median :0.0000 Median :0.00000 Median :1.0000 Median :0.0000   
## Mean :0.1003 Mean :0.05367 Mean :0.5847 Mean :0.2927   
## 3rd Qu.:0.0000 3rd Qu.:0.00000 3rd Qu.:1.0000 3rd Qu.:1.0000   
## Max. :1.0000 Max. :1.00000 Max. :1.0000 Max. :1.0000   
## Education\_1 Education\_2 Education\_3   
## Min. :0.0000 Min. :0.000 Min. :0.0000   
## 1st Qu.:0.0000 1st Qu.:0.000 1st Qu.:0.0000   
## Median :0.0000 Median :0.000 Median :0.0000   
## Mean :0.4173 Mean :0.285 Mean :0.2977   
## 3rd Qu.:1.0000 3rd Qu.:1.000 3rd Qu.:1.0000   
## Max. :1.0000 Max. :1.000 Max. :1.0000

summary(Validation\_data)

## Age Experience Income Family   
## Min. :23.0 Min. :-3.00 Min. : 8.00 Min. :1.000   
## 1st Qu.:35.0 1st Qu.:10.00 1st Qu.: 39.00 1st Qu.:1.000   
## Median :45.0 Median :20.00 Median : 64.00 Median :2.000   
## Mean :45.2 Mean :19.97 Mean : 74.81 Mean :2.409   
## 3rd Qu.:55.0 3rd Qu.:30.00 3rd Qu.: 99.00 3rd Qu.:3.000   
## Max. :67.0 Max. :43.00 Max. :218.00 Max. :4.000   
## CCAvg Education Mortgage Personal.Loan   
## Min. : 0.000 Min. :1.000 Min. : 0.00 Min. :0.0000   
## 1st Qu.: 0.700 1st Qu.:1.000 1st Qu.: 0.00 1st Qu.:0.0000   
## Median : 1.600 Median :2.000 Median : 0.00 Median :0.0000   
## Mean : 1.973 Mean :1.882 Mean : 55.24 Mean :0.1025   
## 3rd Qu.: 2.600 3rd Qu.:3.000 3rd Qu.: 97.25 3rd Qu.:0.0000   
## Max. :10.000 Max. :3.000 Max. :617.00 Max. :1.0000   
## Securities.Account CD.Account Online CreditCard   
## Min. :0.0000 Min. :0.0000 Min. :0.000 Min. :0.000   
## 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.000 1st Qu.:0.000   
## Median :0.0000 Median :0.0000 Median :1.000 Median :0.000   
## Mean :0.1105 Mean :0.0705 Mean :0.615 Mean :0.296   
## 3rd Qu.:0.0000 3rd Qu.:0.0000 3rd Qu.:1.000 3rd Qu.:1.000   
## Max. :1.0000 Max. :1.0000 Max. :1.000 Max. :1.000   
## Education\_1 Education\_2 Education\_3   
## Min. :0.000 Min. :0.000 Min. :0.000   
## 1st Qu.:0.000 1st Qu.:0.000 1st Qu.:0.000   
## Median :0.000 Median :0.000 Median :0.000   
## Mean :0.422 Mean :0.274 Mean :0.304   
## 3rd Qu.:1.000 3rd Qu.:1.000 3rd Qu.:1.000   
## Max. :1.000 Max. :1.000 Max. :1.000

#checking Frequency of personal Loan splited properly or not  
count(Training\_data$`Personal.Loan`)

## x freq  
## 1 0 2725  
## 2 1 275

count(Validation\_data$`Personal.Loan`)

## x freq  
## 1 0 1795  
## 2 1 205

#Data Normalization

train.normalized.df <- Training\_data  
valid.normalized.df <- Validation\_data  
norm.values <- preProcess(Training\_data[, 1:7], method=c("center", "scale"))  
#Replacing columns with normalized values  
train.normalized.df [, 1:7] <- predict(norm.values,Training\_data[,1:7])   
valid.normalized.df [, 1:7] <- predict(norm.values, Validation\_data[,1:7])

#KNN Modeling

cl= as.data.frame(train.normalized.df[,8])  
tnf = as.data.frame(train.normalized.df)  
vnf = as.data.frame(valid.normalized.df)  
dim(cl)

## [1] 3000 1

dim(train.normalized.df[,1:7])

## [1] 3000 7

dim(valid.normalized.df[,1:7])

## [1] 2000 7

knn\_predict <- knn(tnf, vnf, cl=train.normalized.df$`Personal.Loan`, k =1)  
head(knn\_predict)

## [1] 0 0 0 0 1 0  
## Levels: 0 1

knn\_predict <- as.data.frame(knn\_predict)

#assess Data to model

customer\_df <- data.frame ("Age" =40, "Experience"=10, "Income"=84, "Family"=2, "CCAvg"=2, "Education\_1"=0, "Education\_2"=1, "Education\_3"=0, "Mortgage"=0, "Securities Account"=0, "CD Account"=0, "Online" =1, "Credit Card"=1)  
  
dim(tnf)

## [1] 3000 15

dim(customer\_df)

## [1] 1 13

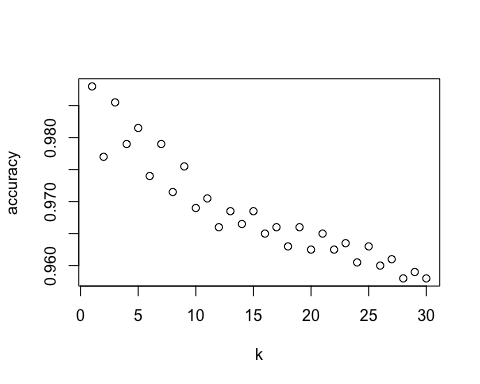
customerClass <- knn ((tnf[, c(-6, -8)]), (customer\_df), cl = train.normalized.df$`Personal.Loan`, k = 1, prob = 0.5)  
  
summary(customerClass) #CUSTOMER class is 1. Customer is likely to accept a personal loan according to this model.

## 1   
## 1

#library(lattice)  
#library(ggplot2)  
#library(caret)  
accuracy.df <- data.frame(k= seq (1, 30, 1), accuracy = rep(0, 30))  
for( i in 1:30) {  
 prediction <- knn ( tnf, vnf, cl = train.normalized.df$`Personal.Loan`, k = i)  
 accuracy.df[i, 2] <- confusionMatrix ( as.factor (prediction), as.factor( valid.normalized.df$`Personal.Loan`))$overall[1]  
}  
accuracy.df

## k accuracy  
## 1 1 0.9880  
## 2 2 0.9770  
## 3 3 0.9855  
## 4 4 0.9790  
## 5 5 0.9815  
## 6 6 0.9740  
## 7 7 0.9790  
## 8 8 0.9715  
## 9 9 0.9755  
## 10 10 0.9690  
## 11 11 0.9705  
## 12 12 0.9660  
## 13 13 0.9685  
## 14 14 0.9665  
## 15 15 0.9685  
## 16 16 0.9650  
## 17 17 0.9660  
## 18 18 0.9630  
## 19 19 0.9660  
## 20 20 0.9625  
## 21 21 0.9650  
## 22 22 0.9625  
## 23 23 0.9635  
## 24 24 0.9605  
## 25 25 0.9630  
## 26 26 0.9600  
## 27 27 0.9610  
## 28 28 0.9580  
## 29 29 0.9590  
## 30 30 0.9580

plot(accuracy.df)

 #Confusion Matrix

#library(gmodels)  
valid\_labels <-as.data.frame( vnf[,8])  
  
#Model accuracy = TP+TN/Total= 99%, specifity= 99.7%, percision= 98%  
CrossTable( valid\_labels$`vnf[, 8]`, knn\_predict$knn\_predict, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 2000   
##   
##   
## | knn\_predict$knn\_predict   
## valid\_labels$`vnf[, 8]` | 0 | 1 | Row Total |   
## ------------------------|-----------|-----------|-----------|  
## 0 | 1795 | 0 | 1795 |   
## | 1.000 | 0.000 | 0.897 |   
## | 0.987 | 0.000 | |   
## | 0.897 | 0.000 | |   
## ------------------------|-----------|-----------|-----------|  
## 1 | 24 | 181 | 205 |   
## | 0.117 | 0.883 | 0.102 |   
## | 0.013 | 1.000 | |   
## | 0.012 | 0.090 | |   
## ------------------------|-----------|-----------|-----------|  
## Column Total | 1819 | 181 | 2000 |   
## | 0.909 | 0.090 | |   
## ------------------------|-----------|-----------|-----------|  
##   
##

#Data Plinting into Training as 50% , Validation as 30% , Testing as 20%

set.seed(12)  
Train\_index2 <- createDataPartition(Universalbank\_dummy$`Personal.Loan`, p=0.50, list=FALSE)  
Training\_data2 <- Universalbank\_dummy[Train\_index2,]  
  
CombinedValidation\_test <- Universalbank\_dummy [-Train\_index2,]  
  
Valid\_index2 <- createDataPartition (CombinedValidation\_test$`Personal.Loan`, p=0.30, list=FALSE)  
Validation\_data2 <- CombinedValidation\_test[Valid\_index2,]  
Test\_data2 <- CombinedValidation\_test[-Valid\_index2,]

#Data Normalization

train.normalized.df2 <- Training\_data2  
valid.normalized.df2 <- Validation\_data2  
Test.normalized.df2 <- Test\_data2  
Combined\_normalized2<-CombinedValidation\_test  
  
norm.values2 <- preProcess(Training\_data2[, 1:7], method=c("center", "scale"))  
  
train.normalized.df2 [, 1:7] <- predict(norm.values2, Training\_data2[,1:7]) # Replace columns with normalized values  
valid.normalized.df2 [, 1:7] <- predict(norm.values2, Validation\_data2[,1:7])  
  
Test.normalized.df2 [, 1:7] <- predict(norm.values2, Test\_data2[, 1:7])  
  
Combined\_normalized2[, 1:7] <- predict(norm.values2, CombinedValidation\_test[,1:7])

#Modeling k-NN with validation data

#library(FNN)  
cl2= as.data.frame(train.normalized.df2[,8])  
tnf2 = as.data.frame(train.normalized.df2)  
vnf2= as.data.frame(valid.normalized.df2)  
dim(cl2)

## [1] 2500 1

dim(train.normalized.df2[,1:7])

## [1] 2500 7

dim(valid.normalized.df2[,1:7])

## [1] 750 7

knn\_predict2 <- knn(tnf2, vnf2, cl=train.normalized.df2$`Personal.Loan`, k =1)  
head(knn\_predict2)

## [1] 0 0 0 0 0 1  
## Levels: 0 1

knn\_predict2 <- as.data.frame(knn\_predict2)

#predicting KNN using validation and test data

cl2= as.data.frame(train.normalized.df2[,8])  
tnf2 = as.data.frame(train.normalized.df2)  
cnf3= as.data.frame(Combined\_normalized2)  
dim(cl2)

## [1] 2500 1

dim(train.normalized.df2[,1:7])

## [1] 2500 7

dim(Combined\_normalized2[,1:7])

## [1] 2500 7

knn\_predict3 <- knn(tnf2, cnf3, cl=train.normalized.df2$`Personal.Loan`, k =1)  
head(knn\_predict3)

## [1] 0 0 0 0 0 0  
## Levels: 0 1

knn\_predict3 <- as.data.frame(knn\_predict3)  
  
  
summary(knn\_predict3)

## knn\_predict3  
## 0:2295   
## 1: 205

#Customer class

customer\_df2 <- data.frame ("Age" =40, "Experience"=10, "Income"=84, "Family"=2, "CCAvg"=2, "Education\_1"=0, "Education\_2"=1, "Education\_3"=0, "Mortgage"=0, "Securities Account"=0, "CD Account"=0, "Online" =1, "Credit Card"=1)  
  
dim(tnf2)

## [1] 2500 15

dim(customer\_df2)

## [1] 1 13

customerClass2 <- knn ((tnf2[, c(-6, -8)]), (customer\_df2), cl = Combined\_normalized2$`Personal.Loan`, k = 1, prob = 0.5)  
 #CUSTOMER class is 0. Customer is NOT likely to accept a personal loan according to this model  
summary(customerClass)

## 1   
## 1

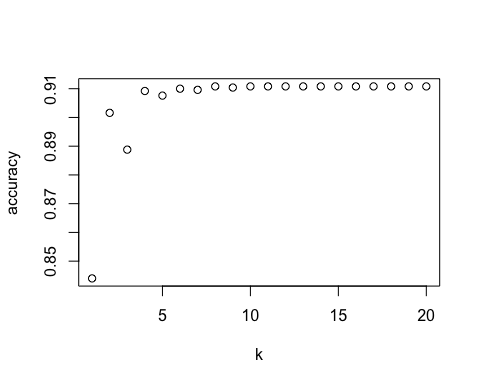
# k= 8 gives the highest accuracy percentage of 91%  
accuracy.df2 <- data.frame(k= seq (1, 20, 1), accuracy = rep(0, 20))  
  
for( y in 1:20){  
 prediction2 <- knn (tnf2, cnf3, cl= Combined\_normalized2$`Personal.Loan`, k = y)  
 accuracy.df2[y, 2] <- confusionMatrix ( as.factor(prediction2) , as.factor(Combined\_normalized2$`Personal.Loan`))$overall[1]  
}

## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.  
  
## Warning in confusionMatrix.default(as.factor(prediction2),  
## as.factor(Combined\_normalized2$Personal.Loan)): Levels are not in the same order  
## for reference and data. Refactoring data to match.

accuracy.df2

## k accuracy  
## 1 1 0.8440  
## 2 2 0.9016  
## 3 3 0.8888  
## 4 4 0.9092  
## 5 5 0.9076  
## 6 6 0.9100  
## 7 7 0.9096  
## 8 8 0.9108  
## 9 9 0.9104  
## 10 10 0.9108  
## 11 11 0.9108  
## 12 12 0.9108  
## 13 13 0.9108  
## 14 14 0.9108  
## 15 15 0.9108  
## 16 16 0.9108  
## 17 17 0.9108  
## 18 18 0.9108  
## 19 19 0.9108  
## 20 20 0.9108

plot(accuracy.df2)



#Using only validation dataset

valid\_labels2 <-as.data.frame( vnf2[,8])  
  
CrossTable( valid\_labels2$`vnf2[, 8]`, knn\_predict2$knn\_predict2, prop.chisq = FALSE) #Model accuracy = TP+TN/Total= 99%, specifity= 99.9%, percision= 99%, sesitivity =93%

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 750   
##   
##   
## | knn\_predict2$knn\_predict2   
## valid\_labels2$`vnf2[, 8]` | 0 | 1 | Row Total |   
## --------------------------|-----------|-----------|-----------|  
## 0 | 670 | 1 | 671 |   
## | 0.999 | 0.001 | 0.895 |   
## | 0.994 | 0.013 | |   
## | 0.893 | 0.001 | |   
## --------------------------|-----------|-----------|-----------|  
## 1 | 4 | 75 | 79 |   
## | 0.051 | 0.949 | 0.105 |   
## | 0.006 | 0.987 | |   
## | 0.005 | 0.100 | |   
## --------------------------|-----------|-----------|-----------|  
## Column Total | 674 | 76 | 750 |   
## | 0.899 | 0.101 | |   
## --------------------------|-----------|-----------|-----------|  
##   
##

#Using combined validation and test datasets

valid\_labels2 <-as.data.frame(cnf3[,8])  
CrossTable( valid\_labels2$`cnf3[, 8]`, knn\_predict3$knn\_predict3, prop.chisq = FALSE ) #Model accuracy = TP+TN/Total= 99.9%, specifity= 99.9%, percision= 98.7%, sesitivity =91% This model give highest results.

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 2500   
##   
##   
## | knn\_predict3$knn\_predict3   
## valid\_labels2$`cnf3[, 8]` | 0 | 1 | Row Total |   
## --------------------------|-----------|-----------|-----------|  
## 0 | 2274 | 3 | 2277 |   
## | 0.999 | 0.001 | 0.911 |   
## | 0.991 | 0.015 | |   
## | 0.910 | 0.001 | |   
## --------------------------|-----------|-----------|-----------|  
## 1 | 21 | 202 | 223 |   
## | 0.094 | 0.906 | 0.089 |   
## | 0.009 | 0.985 | |   
## | 0.008 | 0.081 | |   
## --------------------------|-----------|-----------|-----------|  
## Column Total | 2295 | 205 | 2500 |   
## | 0.918 | 0.082 | |   
## --------------------------|-----------|-----------|-----------|  
##   
##