MIS_64060_Assingment_2

```
#ASSINGMENT_2

#Universal Bank (MIS 64060)

#Importing data set (Universal Bank CSV FILE)

library(readr)

UniversalBank <- read_csv("UniversalBank.csv")
```

```
## Rows: 5000 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (14): ID, Age, Experience, Income, ZIP Code, Family, CCAvg, Education, M...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
spec(UniversalBank)
```

```
## cols(
##   ID = col_double(),
##   Age = col_double(),
##   Experience = col_double(),
##   Income = col_double(),
##   `ZIP Code` = col_double(),
##   Family = col_double(),
##   CCAvg = col_double(),
##   Education = col_double(),
##   Mortgage = col_double(),
##   `Personal Loan` = col_double(),
##   `Securities Account` = col_double(),
##   `CD Account` = col_double(),
##   Online = col_double(),
##   CreditCard = col_double()
## )
```

```
##Assigning names to column

colnames(UniversalBank) <- c('ID', 'Age', 'Experience', 'Income', 'ZIP_Code', 'Family', 'CCAvg',
                             'Education', 'Mortgage', 'Personal_Loan',
                             'Securities_Account', 'CD_Account', 'Online', 'Credit_Card')

summary(UniversalBank)
```

```
##        ID              Age           Experience       Income          ZIP_Code
##  Min.   :   1   Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   : 9307
##  1st Qu.:1251   1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:91911
##  Median :2500   Median :45.00   Median :20.0   Median : 64.00   Median :93437
##  Mean   :2500   Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :93153
##  3rd Qu.:3750   3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:94608
##  Max.   :5000   Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :96651
##      Family          CCAvg          Education        Mortgage
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0
##  Median :2.000   Median : 1.500   Median :2.000   Median :  0.0
##  Mean   :2.396   Mean   : 1.938   Mean   :1.881   Mean   : 56.5
##  3rd Qu.:3.000   3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0
##  Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :635.0
##  Personal_Loan   Securities_Account   CD_Account         Online
##  Min.   :0.000   Min.   :0.0000     Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.000   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.000   Median :0.0000     Median :0.0000   Median :1.0000
##  Mean   :0.096   Mean   :0.1044     Mean   :0.0604   Mean   :0.5968
##  3rd Qu.:0.000   3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1.000   Max.   :1.0000     Max.   :1.0000   Max.   :1.0000
##   Credit_Card
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.294
##  3rd Qu.:1.000
##  Max.   :1.000
```

```
#Getting Rid of Zip Code and ID

UniversalBank$ID <-NULL

UniversalBank$ZIP_Code<-NULL

summary(UniversalBank)
```

```
##       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        Credit_Card
##  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
```

```
# Factoring Education and personal loan

UniversalBank$Education=as.factor(UniversalBank$Education)

UniversalBank$Personal_Loan=as.factor(UniversalBank$Personal_Loan)

summary(UniversalBank)
```

```
##       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 Securities_Account
##  Min.   : 0.000   1:2096    Min.   :  0.0   0:4520        Min.   :0.0000
##  1st Qu.: 0.700   2:1403    1st Qu.:  0.0   1: 480        1st Qu.:0.0000
##  Median : 1.500   3:1501    Median :  0.0                 Median :0.0000
##  Mean   : 1.938             Mean   : 56.5                 Mean   :0.1044
##  3rd Qu.: 2.500             3rd Qu.:101.0                 3rd Qu.:0.0000
##  Max.   :10.000             Max.   :635.0                 Max.   :1.0000
##    CD_Account         Online        Credit_Card
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :0.0000   Median :1.0000   Median :0.000
##  Mean   :0.0604   Mean   :0.5968   Mean   :0.294
##  3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.000
```

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(class)

dummies <- dummyVars(Personal_Loan  ~ ., data = UniversalBank)

UniversalBank_dummy=as.data.frame(predict(dummies, newdata=UniversalBank))
```

```
## Warning in model.frame.default(Terms, newdata, na.action = na.action, xlev =
## object$lvls): variable 'Personal_Loan' is not a factor
```

```
head(UniversalBank_dummy)
```

```
##   Age Experience Income Family CCAvg Education.1 Education.2 Education.3
## 1  25          1     49      4   1.6           1           0           0
## 2  45         19     34      3   1.5           1           0           0
## 3  39         15     11      1   1.0           1           0           0
## 4  35          9    100      1   2.7           0           1           0
## 5  35          8     45      4   1.0           0           1           0
## 6  37         13     29      4   0.4           0           1           0
##   Mortgage Securities_Account CD_Account Online Credit_Card
## 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      155                  0          0      1           0
```

```
#Normalizing Data

Norm_model <- preProcess(UniversalBank_dummy,method= c("center","scale"))


UniversalBank_norm = predict(Norm_model, UniversalBank_dummy)


summary(UniversalBank_norm)
```

```
##       Age              Experience           Income            Family       
##  Min.   :-1.94871   Min.   :-2.014710   Min.   :-1.4288   Min.   :-1.2167  
##  1st Qu.:-0.90188   1st Qu.:-0.881116   1st Qu.:-0.7554   1st Qu.:-1.2167  
##  Median :-0.02952   Median :-0.009121   Median :-0.2123   Median :-0.3454  
##  Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.0000   Mean   : 0.0000  
##  3rd Qu.: 0.84284   3rd Qu.: 0.862874   3rd Qu.: 0.5263   3rd Qu.: 0.5259  
##  Max.   : 1.88967   Max.   : 1.996468   Max.   : 3.2634   Max.   : 1.3973  
##      CCAvg          Education.1       Education.2       Education.3      
##  Min.   :-1.1089   Min.   :-0.8495   Min.   :-0.6245   Min.   :-0.6549  
##  1st Qu.:-0.7083   1st Qu.:-0.8495   1st Qu.:-0.6245   1st Qu.:-0.6549  
##  Median :-0.2506   Median :-0.8495   Median :-0.6245   Median :-0.6549  
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000  
##  3rd Qu.: 0.3216   3rd Qu.: 1.1770   3rd Qu.: 1.6010   3rd Qu.: 1.5266  
##  Max.   : 4.6131   Max.   : 1.1770   Max.   : 1.6010   Max.   : 1.5266  
##     Mortgage       Securities_Account   CD_Account          Online       
##  Min.   :-0.5555   Min.   :-0.3414    Min.   :-0.2535   Min.   :-1.2165  
##  1st Qu.:-0.5555   1st Qu.:-0.3414    1st Qu.:-0.2535   1st Qu.:-1.2165  
##  Median :-0.5555   Median :-0.3414    Median :-0.2535   Median : 0.8219  
##  Mean   : 0.0000   Mean   : 0.0000    Mean   : 0.0000   Mean   : 0.0000  
##  3rd Qu.: 0.4375   3rd Qu.:-0.3414    3rd Qu.:-0.2535   3rd Qu.: 0.8219  
##  Max.   : 5.6875   Max.   : 2.9286    Max.   : 3.9438   Max.   : 0.8219  
##   Credit_Card      
##  Min.   :-0.6452  
##  1st Qu.:-0.6452  
##  Median :-0.6452
```

```
##  Mean   : 0.0000
##  3rd Qu.: 1.5495
##  Max.   : 1.5495
```

```
#Adding the target attribute

UniversalBank_norm$Personal_Loan=UniversalBank$Personal_Loan
```

```
#Dividing the data into train, test and validation.(60/40)

Train1_Index = createDataPartition(UniversalBank$Personal_Loan,p=0.6, list=FALSE) # 60% reserved for Tr
Train1.df=UniversalBank_norm[Train1_Index,]
Validation.df=UniversalBank_norm[-Train1_Index,]
```

```
#Task 1 (a k-NN classification with all predictors except ID and ZIP code using k = 1. How would this c

# (Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Edu

To_Predict = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education.1 = 0,

print(To_Predict)
```

```
##   Age Experience Income Family CCAvg Education.1 Education.2 Education.3
## 1  40         10     84      2     2           0           1           0
##   Mortgage Securities_Account CD_Account Online Credit_Card
## 1        0                  0          0      1           1
```

```
#Applying Normalization

To_Predict_norm= predict(Norm_model,To_Predict)

print(To_Predict_norm)
```

```
##          Age Experience    Income     Family     CCAvg Education.1 Education.2
## 1 -0.4657003 -0.8811162 0.2221371 -0.3453975 0.0355115  -0.8494814    1.601024
##   Education.3   Mortgage Securities_Account CD_Account    Online Credit_Card
## 1  -0.6548999 -0.5554684         -0.3413892 -0.2535149 0.8218687    1.549477
```

```
print(Norm_model)
```

```
## Created from 5000 samples and 13 variables
##
## Pre-processing:
##   - centered (13)
##   - ignored (0)
##   - scaled (13)
```

```
#Using knn for Prediction

Prediction <-knn(train=Train1.df[,1:13],
                 test=To_Predict_norm[,1:13],
```

```
                cl=Train1.df$Personal_Loan,
                k=1)


print(Prediction)
```

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

```
## TASK 2


##Right choice of k reducing the effect of overfitting and underfitting

##k=Number of crossfold Validation

#setting random number variables for reproducible results

set.seed(123)

fitControl <- trainControl(method = "repeatedcv",
                           number = 3,
                           repeats = 2)

searchGrid=expand.grid(k = 1:10)

Knn.model=train(Personal_Loan~.,
                data=Train1.df,
                method='knn',
                tuneGrid=searchGrid,
                trControl = fitControl)

Knn.model
```

```
## k-Nearest Neighbors
##
## 3000 samples
##   13 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold, repeated 2 times)
## Summary of sample sizes: 2000, 2000, 2000, 2000, 2000, 2000, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.9536667  0.7037609
##    2  0.9511667  0.6907851
##    3  0.9566667  0.7049159
##    4  0.9523333  0.6698503
##    5  0.9518333  0.6572339
##    6  0.9500000  0.6417885
##    7  0.9488333  0.6223700
```

```
##    8  0.9490000  0.6225021
##    9  0.9470000  0.6047018
##   10  0.9455000  0.5873667
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
```

```
##RMSE was used to select the optimal model using the smallest value. The final value used for the mode
```

```
##TASK 3

##Confusion matrix for the validation data that results from using the best k

Predictions<- predict(Knn.model,Validation.df)

confusionMatrix(Predictions, Validation.df$Personal_Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1797   65
##          1   11  127
##
##                Accuracy : 0.962
##                  95% CI : (0.9527, 0.9699)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7496
##
##  Mcnemar's Test P-Value : 1.205e-09
##
##             Sensitivity : 0.9939
##             Specificity : 0.6615
##          Pos Pred Value : 0.9651
##          Neg Pred Value : 0.9203
##              Prevalence : 0.9040
##          Detection Rate : 0.8985
##    Detection Prevalence : 0.9310
##       Balanced Accuracy : 0.8277
##
##        'Positive' Class : 0
##
```

```
##TASK 4

##classifying customers using best k

# considerations = (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 and Credit
# Card = 1.)
```

```
Prediction2 <-knn(train=Train1.df[,1:13],
                  test=To_Predict_norm[,1:13],
                  cl=Train1.df$Personal_Loan,
                  k=3)


print(Prediction2)
```

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

```
##TASK 5

#Repartition of the data into train, test and validation.(50/30/20)

Train2_Index = createDataPartition(UniversalBank$Personal_Loan,p=0.5, list=FALSE) # 50% reserved for Tr
Train2.df=UniversalBank_norm[Train2_Index,]
validation1.df=UniversalBank_norm[-Train2_Index,]

validation1_Index = createDataPartition(validation1.df$Personal_Loan,p=0.6, list=FALSE) # 60% reserved
validation2.df=validation1.df[validation1_Index,]
Test1.df=validation1.df[-validation1_Index,]
```

```
# (Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Edu

To_Predict1 = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education.1 = 0

print(To_Predict1)
```

```
##   Age Experience Income Family CCAvg Education.1 Education.2 Education.3
## 1  40         10     84      2     2           0           1           0
##   Mortgage Securities_Account CD_Account Online Credit_Card
## 1        0                  0          0      1           1
```

```
#Applying Normalization


Norm_model2 <- preProcess(Train2.df[,-13], method = c("center", "scale"))

Train2_Norm <- predict(Norm_model2, Train2.df[,-13])

Validation2_Norm <- predict(Norm_model2, validation2.df [,-13])

Test1_Norm <- predict(Norm_model2, Test1.df[,-13])

Prediction3 <- knn(Train2_Norm, Validation2_Norm , cl=Train2.df$`Personal_Loan`, k=3,)

Prediction3
```

```
##    [1] 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##   [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
```

```
##   [75] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0
##  [112] 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
##  [149] 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [186] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
##  [223] 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [260] 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1
##  [297] 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
##  [334] 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
##  [371] 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0
##  [408] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
##  [445] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 0 0
##  [482] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
##  [519] 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [556] 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
##  [593] 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
##  [630] 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [667] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0
##  [704] 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [741] 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [778] 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
##  [815] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [852] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0
##  [889] 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0
##  [926] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 1
##  [963] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
## [1000] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [1037] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0
## [1074] 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
## [1111] 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0
## [1148] 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [1185] 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
## [1222] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [1259] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
## [1296] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
## [1333] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
## [1370] 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [1407] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [1444] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [1481] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## Levels: 0 1
```

```
confusionMatrix(Prediction3, validation2.df$`Personal_Loan`)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1356   37
##          1    0  107
##
##                Accuracy : 0.9753
##                  95% CI : (0.9662, 0.9826)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
```

```
##
##                  Kappa : 0.8394
##
##  Mcnemar's Test P-Value : 3.252e-09
##
##             Sensitivity : 1.0000
##             Specificity : 0.7431
##          Pos Pred Value : 0.9734
##          Neg Pred Value : 1.0000
##              Prevalence : 0.9040
##          Detection Rate : 0.9040
##    Detection Prevalence : 0.9287
##       Balanced Accuracy : 0.8715
##
##        'Positive' Class : 0
##
```

```
##Comaprision-

# (Here we can see difference in test set  with validation and training set. Major statistical differen

#(Overall we can see better result in set with test, validation and train set as using same k also, the
```