ML2

Gloria Stephen

2022-10-02

```
library(readr)
library(caret)
```

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

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

```
UniversalBank <- read_csv("C:/Users/idast/Downloads/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.
```

```
UniversalBank$Personal.Loan =as.factor(UniversalBank$Personal.Loan)
data1<-UniversalBank

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     Personal.Loan
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0   0:4520
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0   1: 480
##  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
##  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
```

```
View(data1)
```

##removing unwanted data from dataset

```
data2<-data1[,-c(1,5)]
View(data2)
```

#converting personal loan to factors

```
#data2$Personal.Loan =as.factor(data2$Personal.Loan)

View(data2)
```

#creating the dummy variables

```
Education_1 <- ifelse(data2$Education == 1, 1,0)
Education_2 <- ifelse(data2$Education == 2, 1,0)
Education_3 <- ifelse(data2$Education == 3, 1,0)
```

```
data3<-data.frame(Age=data2$Age,Experience=data2$Experience,Income=data2$Income,Family=data2$Family,CCA

View(data3)
```

##creating a partition of 60:40

```
Train_Index = createDataPartition(data3$Personal.Loan,p=0.6, list = FALSE)
Train.df =data3[Train_Index,]
Validation.df=data3[-Train_Index,]

nrow(Train.df)
```

```
## [1] 3000
```

```
nrow(Validation.df)
```

```
## [1] 2000
```

#normalization of the data

```
Norm_model <- preProcess(Train.df[,-(6:9)], method = c("center", "scale"))

training_norm<-predict(Norm_model,Train.df)

validation_norm<-predict(Norm_model,Validation.df)
```

#test data set

```
Test<-data.frame(Age=40,Experience=10,Income=84,Family=2,CCAvg=2,Education_1=0,Education_2=1,Education_


View(Test)

test_norm<-predict(Norm_model,Test)
```

#knn algorithm in dataset

```
library(class)

pred_train<-training_norm[,-9]
label_train<-training_norm[,9]

pred_valid<-validation_norm[,-9]
label_valid<-validation_norm[,9]



View(training_norm)

Model<-knn(pred_train,test_norm,cl=label_train,k=1)

Model
```

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

```
#since value of k=0, the customer will not accept the loan offer
```

#Finding the best value for k

```
set.seed(541)

searchGrid <- expand.grid(k=seq(1:30))

predicted <- train(Personal.Loan~.,training_norm,method="knn",tuneGrid=searchGrid)

predicted
```

```
## k-Nearest Neighbors
##
## 3000 samples
##   13 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 3000, 3000, 3000, 3000, 3000, 3000, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
```

```
##    1  0.9474226  0.6597318
##    2  0.9415773  0.6220057
##    3  0.9407644  0.6073046
##    4  0.9398258  0.5911270
##    5  0.9406885  0.5893467
##    6  0.9409011  0.5826246
##    7  0.9403928  0.5721435
##    8  0.9397173  0.5640636
##    9  0.9389523  0.5496778
##   10  0.9391248  0.5464590
##   11  0.9388502  0.5420046
##   12  0.9369975  0.5217099
##   13  0.9360908  0.5107912
##   14  0.9350713  0.5003536
##   15  0.9351159  0.4975322
##   16  0.9337495  0.4831006
##   17  0.9331800  0.4761293
##   18  0.9321740  0.4679611
##   19  0.9320925  0.4645789
##   20  0.9316627  0.4560454
##   21  0.9308652  0.4492359
##   22  0.9305707  0.4451149
##   23  0.9301748  0.4408971
##   24  0.9299997  0.4371372
##   25  0.9293147  0.4277107
##   26  0.9287329  0.4235959
##   27  0.9283323  0.4177395
##   28  0.9278963  0.4151312
##   29  0.9276433  0.4119218
##   30  0.9271669  0.4061719
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

```
best_k <- predicted$bestTune[[1]]

#this saves the best value for k
```

#the confusion matrix

```
model<-predict(predicted,validation_norm[,-9])

confusionMatrix(model,label_valid)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1790   60
##          1   18  132
##
##                Accuracy : 0.961
##                  95% CI : (0.9516, 0.9691)
```

```
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.751
##
##  Mcnemar's Test P-Value : 3.445e-06
##
##             Sensitivity : 0.9900
##             Specificity : 0.6875
##          Pos Pred Value : 0.9676
##          Neg Pred Value : 0.8800
##              Prevalence : 0.9040
##          Detection Rate : 0.8950
##    Detection Prevalence : 0.9250
##       Balanced Accuracy : 0.8388
##
##        'Positive' Class : 0
##
```

#knn for new customer

```
Prediction_new<-knn(pred_train,test_norm,cl=label_train,k=best_k)

Prediction_new
```

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

```
#k=0, new customer will not accept the loan offer
```

#5#splitting data to 50:30:20 ratio

set.seed(887)

part1<- createDataPartition(data3$Personal.Loan,p=.5,list=FALSE,times=1)

part2<- createDataPartition(data3$Personal.Loan,p=.3,list=FALSE,times=1)

part3<- createDataPartition(data3$Personal.Loan,p=.2,list=FALSE,times=1)

new_train <- data3[part1,]

new_valid <- data3[part2,]

new_test <- data3[part3,]

#normalization of the data

normalized <- preProcess(Train.df[,-(6:9)], method=c("center","scale"))

#training data

normalized_train <- predict(normalized,new_train)

#the validation data

normalized_valid<-predict(normalized,new_valid)

#the test data created

```
normalized_test<-predict(normalized,new_test)
#running knn for train,validation and test data
pred_train1=normalized_train[,-9]
label_train1=normalized_train[,9]
pred_valid1=normalized_valid[,-9]
label_valid1=normalized_valid[,9]
pred_test1=normalized_test[,-9]
label_test1=normalized_test[,9]
training_prediction<-knn(pred_train1,pred_train1,cl=label_train1,k=best_k)
validation_prediction<-knn(pred_train1,pred_valid1,cl=label_train1,k=best_k)
test_prediction<-knn(pred_train1,pred_test1,cl=label_train1,k=best_k)
#confusion matrix of training data
confusionMatrix(pred_train1,label_train1)
#confusion matrix of validation data
confusionMatrix(pred_valid1,label_valid1)
#confusion matrix of test data
confusionMatrix(pred_test1,label_test1)
```