Assignment 2

Brody Coyne

2022-10-03

Importing the data and showing the first 6 rows.

```
mydata<-read.csv("UniversalBank.csv")
head(mydata)
```

```
##   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
##   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
```

Changing the data from a class to a factor, adding a dummy variable for education, and getting rid of the ID and zipcode columns.

```
q<-class2ind(as.factor(mydata$Education))
colnames(q) <- c('Edu1','Edu2','Ed3')
new_mydata<-cbind(mydata[,2:4],mydata[,6:7],q,mydata[9],mydata[,11:14],mydata[10])
```

- Partitioning the data in a 60/40 split between training and validation sets.

```
Index_Train<-createDataPartition(new_mydata$Personal.Loan, p=0.6, list = FALSE)
Train <-new_mydata[Index_Train,]
Validation <-new_mydata[-Index_Train,]
```

-

Seperating the labels from the predictors

```
Train_Predictors<-Train[,1:13]
Validation_Predictors<-Validation[,1:13]
Train_labels <-Train[,14]
Validation_labels <-Validation[,14]
```

Importing the customer data that needs to analyzed so the customer can be classified.

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

Finding the knn at k=1

```
Predicted_Train_labels <-knn(Train_Predictors, Validation_Predictors, cl=Train_labels, k=1)
head(Predicted_Train_labels)
```

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

Finding the best knn. I chose k=9 as the best value here because it has the highest accuracy rate.

```
new_mydata$Personal.Loan = factor(new_mydata$Personal.Loan)
set.seed(123)
model<-train(Personal.Loan~ ., data = new_mydata, method="knn")
model
```

```
## k-Nearest Neighbors
##
## 5000 samples
##   13 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 5000, 5000, 5000, 5000, 5000, 5000, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.9008936  0.3714833
##   7  0.9040472  0.3714627
##   9  0.9043035  0.3613566
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

Finding the knn at the value I chose, k=9.

```
Predicted_Validation_labels<-knn(Validation_Predictors,Train_Predictors, cl=Validation_labels, k=9)
head(Predicted_Validation_labels)
```

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

Created a confusion matrix using the best k value. I would say that the model performed okay because only 176 cases out of the 2000 total cases were misclassified.

```
CrossTable(x=Validation_labels, y=Predicted_Train_labels, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##                   | Predicted_Train_labels
## Validation_labels |         0 |         1 | Row Total |
## ------------------|-----------|-----------|-----------|
##                 0 |      1724 |        82 |      1806 |
##                   |     0.955 |     0.045 |     0.903 |
##                   |     0.940 |     0.497 |           |
##                   |     0.862 |     0.041 |           |
## ------------------|-----------|-----------|-----------|
##                 1 |       111 |        83 |       194 |
##                   |     0.572 |     0.428 |     0.097 |
##                   |     0.060 |     0.503 |           |
##                   |     0.056 |     0.042 |           |
## ------------------|-----------|-----------|-----------|
##      Column Total |      1835 |       165 |      2000 |
##                   |     0.917 |     0.082 |           |
## ------------------|-----------|-----------|-----------|
##
##
```

Using the specific customer mentioned in the instructions and using k=1 to test if that customer will accept the loan offer or not. Based on the reponse being 0 it is predicted that they will not accept the loan.

```
Predicted_Customer<-knn(Train_Predictors,Customer_Data, cl=Train_labels, k=1)
head(Predicted_Customer)
```

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

Using the information for the same customer but this time using k=9. This also predicts that the customer will not accept the loan offer.

```
Predicted_Customer<-knn(Train_Predictors,Customer_Data, cl=Train_labels, k=9)
head(Predicted_Customer)
```

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

Repartitioning the data in a 50/30/20 split between training, validation, and testing data. This leaves 2500 observations in Train1, 1500 observations in Validation1, and 1000 observations in Test1.

```
Index_Train1<-createDataPartition(new_mydata$Personal.Loan, p=0.5, list = FALSE)
Train1 <-new_mydata[Index_Train1,]
ValidationandTest <-new_mydata[-Index_Train1,]
Index_Train2<-createDataPartition(ValidationandTest$Personal.Loan,p=0.6, list = FALSE)
Validation1 <-ValidationandTest[Index_Train2,]
Test1 <-ValidationandTest[-Index_Train2,]
```

After repartitioning the data I had to set the variables up so that I could make another confusion matrix.

```
Train_Predictors1<-Train[,1:13]
Test_Predictors1<-Validation[,1:13]
Train_labels1 <-Train[,14]
Test_labels1 <-Validation[,14]
Predicted_Test_labels1<-knn(Train_Predictors1,Test_Predictors1,cl=Train_labels1,k=9)
head(Predicted_Test_labels1)
```

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

Making a new confusion matrix with the repartitioned data. It is pretty similar to the one created earlier except it performed marginally better due to ther being only 164 cases being misclassified. I think that it performed a bit better due to there being more data to learn from.

```
CrossTable(x=Test_labels1, y=Predicted_Test_labels1, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##              | Predicted_Test_labels1
## Test_labels1 |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |      1763 |        43 |      1806 |
##              |     0.976 |     0.024 |     0.903 |
##              |     0.922 |     0.489 |           |
##              |     0.881 |     0.021 |           |
```

```
## -------------|-----------|-----------|-----------|
##            1 |       149 |        45 |       194 |
##              |     0.768 |     0.232 |     0.097 |
##              |     0.078 |     0.511 |           |
##              |     0.074 |     0.022 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1912 |        88 |      2000 |
##              |     0.956 |     0.044 |           |
## -------------|-----------|-----------|-----------|
##
##
```