Assignment_2

Matt Kostoff

2022-10-02

Question #1

Partition Data into training (60%) and validation (40%)

```
library (class)
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library (ISLR)
bank.df<-read.csv("universalbank.csv")
bank.df<-subset(bank.df, select=-ID)
bank.df<-subset(bank.df, select=-ZIP.Code)
head(bank.df)

##   Age Experience Income Family CCAvg Education Mortgage Personal.Loan
## 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         2        0             0
## 5  35          8     45      4   1.0         2        0             0
## 6  37         13     29      4   0.4         2      155             0
##   Securities.Account CD.Account Online CreditCard
## 1                  1          0      0          0
## 2                  1          0      0          0
## 3                  0          0      0          0
## 4                  0          0      0          0
## 5                  0          0      0          1
## 6                  0          0      1          0

set.seed(123)
train.index=(createDataPartition(bank.df$Age, p = 0.6, list=FALSE))
train.df<-bank.df[train.index,]
valid.df<-bank.df[-train.index,]
```

Normalization

```
train.norm.df<-train.df
valid.norm.df<-valid.df
bank.norm.df<-bank.df
norm.values<-preProcess(train.df[, -8], method=c("center", "scale"))
train.norm.df[, -8]<-predict(norm.values, train.df[,-8])
```

```
valid.norm.df[, -8]<-predict(norm.values, valid.df[, -8])
bank.norm.df[, -8]<-predict(norm.values, bank.df[, -8])
```

Classification of customer

```
library(FNN)

##
## Attaching package: 'FNN'

## The following objects are masked from 'package:class':
##
##     knn, knn.cv

new.df<-data.frame(40,10,84,2,2,0,0,0,0,1,1)
names(new.df)<-names(train.norm.df)[-8]
new.norm.values<-preProcess(new.df, method=c("center","scale"))

## Warning in preProcess.default(new.df, method = c("center", "scale")): Std.
## deviations could not be computed for: Age, Experience, Income, Family,
CCAvg,
## Education, Mortgage, Securities.Account, CD.Account, Online, CreditCard

new.norm.df<-predict(new.norm.values, newdata=new.df)
new.knn.pred <- class::knn(train = train.norm.df[,-8], test = new.norm.df, cl
= train.df$Personal.Loan, k = 1)
new.knn.pred

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

# Customer is classified as Personal Loan = 0, which means they would not
accept
```

Question #2

```
accuracy.df <- data.frame(k = seq(1, 14, 1), RSME = rep (0, 14))
for(i in 1:14){
knn.pred<-class::knn(train = train.norm.df[,-8],test = valid.norm.df[,-8], cl
= train.df[,8], k = i)
accuracy.df[i,2]<-RMSE(as.numeric(as.character(knn.pred)),valid.df[,8])
}
accuracy.df

##     k      RSME
## 1   1 0.2098142
## 2   2 0.2280921
## 3   3 0.2133606
## 4   4 0.2168491
## 5   5 0.2156925
## 6   6 0.2145297
## 7   7 0.2202822
## 8   8 0.2214148
## 9   9 0.2247783
## 10 10 0.2225416
## 11 11 0.2236627
## 12 12 0.2302749
## 13 13 0.2313585
## 14 14 0.2356433
```

```
#k=3 is the next lowest RSME value, so it provides a good balance between
overfitting and ignoring the predictor information
```

Question #3 - confusion matrix for validation data (k=3)

```
Train_Predictors<-train.df[,-8]
Val_Predcitors<-valid.df[,-8]
Val_Predcitors<-valid.df[,-8]
Train_labels<-train.df[,8]
Val_labels<-valid.df[,8]
Predicted_Val_labels<-knn(Train_Predictors,Val_Predcitors,cl=Train_labels,
k=3)
head(Predicted_Val_labels)
```

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

```
library("gmodels")
CrossTable(x=Val_labels, y=Predicted_Val_labels, prop.chisq = FALSE)
```

```
## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  1999 
## 
## 
##              | Predicted_Val_labels 
##   Val_labels |         0 |         1 | Row Total | 
## -------------|-----------|-----------|-----------|
##            0 |      1730 |        70 |      1800 | 
##              |     0.961 |     0.039 |     0.900 | 
##              |     0.934 |     0.476 |           | 
##              |     0.865 |     0.035 |           | 
## -------------|-----------|-----------|-----------|
##            1 |       122 |        77 |       199 | 
##              |     0.613 |     0.387 |     0.100 | 
##              |     0.066 |     0.524 |           | 
##              |     0.061 |     0.039 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1852 |       147 |      1999 |
##              |     0.926 |     0.074 |           |
## -------------|-----------|-----------|-----------|
##
##
```

Question #4

#Classify customer using best k, which is k = 3

```
library(FNN)
new.df<-data.frame(40,10,84,2,2,0,0,0,0,1,1)
names(new.df)<-names(train.norm.df)[-8]
new.norm.values<-preProcess(new.df, method=c("center","scale"))
```

```
## Warning in preProcess.default(new.df, method = c("center", "scale")): Std.
## deviations could not be computed for: Age, Experience, Income, Family,
CCAvg,
## Education, Mortgage, Securities.Account, CD.Account, Online, CreditCard
```

```
new.norm.df<-predict(new.norm.values, newdata=new.df)
new.knn.pred <- class::knn(train = train.norm.df[,-8], test = new.norm.df, cl
= train.df$Personal.Loan, k = 3)
new.knn.pred
```

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

```
# Customer is classified as Personal Loan = 0, which means they would not
accept
```

Question #5 - repartition (50:30:20%)

```
set.seed(123)
train.rows<-sample(rownames(bank.df), dim(bank.df)[1]*0.5)
valid.rows<-sample(setdiff(rownames(bank.df), train.rows),
dim(bank.df)[1]*0.3)
test.rows<-setdiff(rownames(bank.df), union(train.rows, valid.rows))
train.df<-bank.df[train.rows,]
valid.df<-bank.df[valid.rows,]
test.df<-bank.df[test.rows,]
#Normalize
train.norm.df<-train.df
valid.norm.df<-valid.df
test.norm.df<-test.df
norm.values<-preProcess(train.df[, -8], method=c("center", "scale"))
train.norm.df[, -8]<-predict(norm.values, train.df[,-8])
valid.norm.df[, -8]<-predict(norm.values, valid.df[, -8])
test.norm.df[, -8]<-predict(norm.values, test.df[, -8])
#Confusion Matrix - Validation
```

```
Train_Predictors<-train.df[,-8]
Val_Predcitors<-valid.df[,-8]
Val_Predcitors<-valid.df[,-8]
Train_labels<-train.df[,8]
Val_labels<-valid.df[,8]
Predicted_Val_labels<-knn(Train_Predictors,Val_Predcitors,cl=Train_labels,
k=3)
head(Predicted_Val_labels)
```

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

```
library("gmodels")
CrossTable(x=Val_labels, y=Predicted_Val_labels, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1500
##
##
##              | Predicted_Val_labels
##   Val_labels |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |      1295 |        62 |      1357 |
##              |     0.954 |     0.046 |     0.905 |
##              |     0.936 |     0.534 |           |
##              |     0.863 |     0.041 |           |
## -------------|-----------|-----------|-----------|
##            1 |        89 |        54 |       143 |
##              |     0.622 |     0.378 |     0.095 |
##              |     0.064 |     0.466 |           |
##              |     0.059 |     0.036 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1384 |       116 |      1500 |
##              |     0.923 |     0.077 |           |
## -------------|-----------|-----------|-----------|
##
##
```

```
#Confusion Matrix - Test
Train_Predictors<-train.df[,-8]
Test_Predcitors<-test.df[,-8]
```

```
Test_Predcitors<-test.df[,-8]
Train_labels<-train.df[,8]
Test_labels<-test.df[,8]
Predicted_Test_labels<-knn(Train_Predictors,Test_Predcitors,cl=Train_labels,
k=3)
head(Predicted_Test_labels)
```

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

```
library("gmodels")
CrossTable(x=Test_labels, y=Predicted_Test_labels, prop.chisq = FALSE)
```

```
## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  1000 
## 
## 
##              | Predicted_Test_labels 
##  Test_labels |         0 |         1 | Row Total | 
## -------------|-----------|-----------|-----------|
##            0 |       845 |        47 |       892 | 
##              |     0.947 |     0.053 |     0.892 | 
##              |     0.927 |     0.534 |           | 
##              |     0.845 |     0.047 |           | 
## -------------|-----------|-----------|-----------|
##            1 |        67 |        41 |       108 | 
##              |     0.620 |     0.380 |     0.108 | 
##              |     0.073 |     0.466 |           | 
##              |     0.067 |     0.041 |           | 
## -------------|-----------|-----------|-----------|
## Column Total |       912 |        88 |      1000 | 
##              |     0.912 |     0.088 |           | 
## -------------|-----------|-----------|-----------|
## 
## 
```

Question #5 Comments

Fewer false negatives and false positives in test observations due to lower total volume, but lower rate of false negatives and false positives in validation observations due more total observations