Assignment 2

Julia Thacker

10/3/2021

Imported UniversalBank.csv

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(ISLR)  
UniversalBank <-read.csv("UniversalBank.csv")  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

m\_UniversalBank<-select(UniversalBank,Age,Experience,Income,Family,CCAvg,Education,Mortgage,Personal.Loan,Securities.Account,CD.Account,Online,CreditCard)

Created a data set excluding the ID and Zip code

m\_UniversalBank$Education<-as.factor(m\_UniversalBank$Education)  
class(m\_UniversalBank$Education)

## [1] "factor"

Converted the education data into factors and checked their class

dummy\_model<-dummyVars(~Education,data=m\_UniversalBank)  
head(predict(dummy\_model,m\_UniversalBank))

## Education.1 Education.2 Education.3  
## 1 1 0 0  
## 2 1 0 0  
## 3 1 0 0  
## 4 0 1 0  
## 5 0 1 0  
## 6 0 1 0

Transformed Education data into dummy variables

Education <- dummyVars(~Education,m\_UniversalBank)  
EduDV<-predict(Education,m\_UniversalBank)  
m\_UniversalBank<-cbind(m\_UniversalBank,EduDV)  
m\_UniversalBank<-m\_UniversalBank[,-6]

Added the three education columns that are now dummy variables, into the original data set and removed the original education column

Test\_Data<-data.frame(Age=40,Experience=10,Income=84,Family=2,CCAvg=2,Mortgage=0,Securities.Account=0,CD.Account=0,Online=1,CreditCard=1,Education.1=0,Education.2=1,Education.3=0)

Added a data frame for the example customer test data

set.seed(123)  
Train\_Index=createDataPartition(m\_UniversalBank$Age,p=0.6,list=FALSE)  
Train\_Data=m\_UniversalBank[Train\_Index,]  
Validation\_Data=Train\_Data[-Train\_Index,]  
Traval\_Data=m\_UniversalBank

Partitioned the data into 60% training data and 40% validation

train.norm.df<-Train\_Data  
valid.norm.df<-Validation\_Data  
test.norm.df<-Test\_Data  
allbankdata.norm.df<-m\_UniversalBank  
traval.norm.df<-Traval\_Data

Copied the original data

norm.values<-preProcess(Train\_Data[,-7],method=c("center","scale"))  
train.norm.df[,-7]<-predict(norm.values,Train\_Data[,-7])  
valid.norm.df[,-7]<-predict(norm.values,Validation\_Data[,-7])  
test.norm.df<-predict(norm.values,Test\_Data)  
allbankdata.norm.df<-predict(norm.values,m\_UniversalBank[,-7])  
traval.norm.df[,-7]<-predict(norm.values,traval.norm.df[,-7])

Normalized all of the data

library(FNN)  
set.seed(123)  
nn<-knn(train=train.norm.df[,c(1:6,8:14)],test=test.norm.df,cl=train.norm.df[,7],k=1,prob=TRUE)  
actual=valid.norm.df$Personal.Loan  
nn\_pred=attr(nn,"prob")  
row.names(Train\_Data)[attr(nn,"nn.idex")]

## character(0)

nn

## [1] 0  
## attr(,"prob")  
## [1] 1  
## attr(,"nn.index")  
## [,1]  
## [1,] 2655  
## attr(,"nn.dist")  
## [,1]  
## [1,] 0.4975307  
## Levels: 0

Performed k-NN classification with default cutoff value of 0.5 and a success class of 1. This customer would accept a personal loan.

accuracy.df<-data.frame(k=seq(1,14,1),accuracy=rep(0,14))  
for(i in 1:14){  
 knn.pred<-knn(train=train.norm.df[,-7],test=valid.norm.df[,-7],  
 cl=train.norm.df[,7],k=i,prob=TRUE)  
 accuracy.df[i,2]<-mean(knn.pred==actual)  
}  
accuracy.df

## k accuracy  
## 1 1 1.0000000  
## 2 2 0.9761905  
## 3 3 0.9821429  
## 4 4 0.9693878  
## 5 5 0.9761905  
## 6 6 0.9642857  
## 7 7 0.9685374  
## 8 8 0.9574830  
## 9 9 0.9625850  
## 10 10 0.9540816  
## 11 11 0.9591837  
## 12 12 0.9540816  
## 13 13 0.9574830  
## 14 14 0.9549320

accuracy.df<-data.frame(k=seq(1,14,1),accuracy=rep(0,14))  
for(i in 1:14){  
 knn.pred<-knn(train.norm.df[,-7], valid.norm.df[,-7],  
 cl=train.norm.df[,7],k=i)  
 accuracy.df[i,2]<-confusionMatrix(knn.pred, as.factor(valid.norm.df[,7]))$overall[1]  
}  
accuracy.df

## k accuracy  
## 1 1 1.0000000  
## 2 2 0.9761905  
## 3 3 0.9821429  
## 4 4 0.9693878  
## 5 5 0.9761905  
## 6 6 0.9642857  
## 7 7 0.9685374  
## 8 8 0.9574830  
## 9 9 0.9625850  
## 10 10 0.9540816  
## 11 11 0.9591837  
## 12 12 0.9540816  
## 13 13 0.9574830  
## 14 14 0.9549320

Found the best value of K that balances between overfitting and ignoring the predictor information. Tried it two different ways with the same results. accuracy.df shows the accuracy of each K value with K=1 being the most accurate with an accuracy of 1.0.

table(knn.pred,actual)

## actual  
## knn.pred 0 1  
## 0 1064 51  
## 1 2 59

mean(knn.pred==actual)

## [1] 0.954932

Confusion Matrix for best value K=1

Test\_Data2.df<-data.frame(Age=40,Experience=10,Income=84,Family=2,CCAvg=2,Mortgage=0,Securities.Account=0,CD.Account=0,Online=1,CreditCard=1,Education.1=0,Education.2=1,Education.3=0)

Added a data frame for the second example customer test data set

nn2<-knn(train=train.norm.df[,c(1:6,8:14)],test=Test\_Data2.df,cl=train.norm.df[,7],k=1,prob=TRUE)  
nn2

## [1] 1  
## attr(,"prob")  
## [1] 1  
## attr(,"nn.index")  
## [,1]  
## [1,] 551  
## attr(,"nn.dist")  
## [,1]  
## [1,] 90.53694  
## Levels: 1

Classified customer number 2 using best K. They would accept the loan offer based on the success class of 1 and default cutoff value of 0.5.

set.seed(123)  
  
  
Test\_Index2=createDataPartition(m\_UniversalBank$Age,p=0.2,list=FALSE)  
Test\_Data2=m\_UniversalBank[Test\_Index2,]  
  
Remaining\_Data=m\_UniversalBank[-Test\_Index2,]  
  
Train\_Index2=createDataPartition(Remaining\_Data$Age,p=0.5,list=FALSE)  
   
Train\_Data2=Remaining\_Data[Train\_Index2,]  
  
Validation\_Data2=Remaining\_Data[-Train\_Index2,]

Repartitioned the data into 50% training, 30% validation, and 20% test data

train.norm2.df<-Train\_Data2  
valid.norm2.df<-Validation\_Data2  
test.norm2.df<-Test\_Data2  
remainingdata.norm.df<-Remaining\_Data  
norm.values2<-preProcess(Train\_Data2[,-7],method=c("center","scale"))  
train.norm2.df[,-7]<-predict(norm.values2,Train\_Data2[,-7])  
valid.norm2.df[,-7]<-predict(norm.values2,Validation\_Data2[,-7])  
test.norm2.df[,-7]<-predict(norm.values2,Test\_Data2[,-7])  
remainingdata.norm.df[,-7]<-predict(norm.values2,Remaining\_Data[,-7])

Normalized the repartitioned data

set.seed(123)  
nn2<-knn(train=train.norm2.df[,-7],test=valid.norm2.df[,-7],cl=train.norm2.df[,7],k=1,prob=TRUE)  
actual=valid.norm2.df$Personal.Loan  
nn2\_pred=attr(nn2,"prob")  
table(nn2,actual)

## actual  
## nn2 0 1  
## 0 1772 74  
## 1 23 130

mean(nn2==actual)

## [1] 0.9514757

Used k=1 with the new reparition of the data

nn3<-knn(train=remainingdata.norm.df[,-7],test=test.norm2.df[,-7],cl=remainingdata.norm.df[,7],k=1,prob=TRUE)  
actual=test.norm2.df$Personal.Loan  
nn3\_pred=attr(nn3,"prob")  
table(nn3,actual)

## actual  
## nn3 0 1  
## 0 903 24  
## 1 9 65

mean(nn3==actual)

## [1] 0.967033

The model performed better with the test data set. It produced a value of 0.967 compared to 0.951 in the training and validation data.