problem7-3

Mehrab Atighi

11/19/2021

7.3 **Predicting Housing Median Prices.** The file BostonHousing.csv contains information on over 500 census tracts in Boston, where for each tract multiple variables are recorded. The last column (CAT.MEDV) was derived from MEDV, such that it obtains the value 1 if MEDV > 30 and 0 otherwise. Consider the goal of predicting the median value (MEDV) of a tract, given the information in the first 12 columns. Partition the data into training (60%) and validation (40%) sets.

a. Perform a k-NN prediction with all 12 predictors (ignore the CAT.MEDV column), trying values of k from 1 to 5. Make sure to normalize the data, and choose function knn() from package class rather than package FNN. To make sure R is using the class package (when both packages are loaded), use class::knn(). What is the best k? What does it mean?

problem

b. Predict the MEDV for a tract with the following information, using the best k:

CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	LSTAT
0.2	0	7	0	0.538	6	62	4.7	4	307	21	10

c. If we used the above k-NN algorithm to score the training data, what would be the error of the training set?

d. Why is the validation data error overly optimistic compared to the error rate when applying this k-NN predictor to new data?

e. If the purpose is to predict MEDV for several thousands of new tracts, what would be the disadvantage of using k-NN prediction? List the operations that the algorithm goes through in order to produce each prediction.

solution

```
Data<-read.csv("F:/lessons/Data mining/Data/BostonHousing.csv")
data=Data[,-13]

head(data,4)
```

```
##      CRIM ZN INDUS CHAS   NOX    RM  AGE    DIS RAD TAX PTRAT
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18
```

```
dim(data)
```

```
## [1] 506  13
```

solution

```
set.seed(5)
n = sample(c(0,1),nrow(data) ,prob = c(0.6,0.4)   ,replace =
train_data=data[which(n==0),]
test_data=data[which(n==1),]
```

solution

```
#a)
train_normal<- train_data
test_normal<-test_data
data_normal<-data
library(caret)
```

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

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

```
norm.values<-preProcess(train_data[,-13] , method = c("center" ,"scale"
train_normal<-predict(norm.values , train_data[,-13])
test_normal<-predict(norm.values , test_data[,-13])
data_normal<-predict(norm.values , data[,-13])
library(class)
accuracy_data=data.frame(k = seq(1,5,1) , accuracy=rep(0,5))
system.time(
for(i in 1:5){
  knn.predict<-class::knn(train =train_normal[,-13] ,test = test_normal
                          cl = train_data[,13] ,k=i)
  accuracy_data[i , 2] = confusionMatrix(data = knn.predict[1:nrow(test
                                         factor(test_data[,13]) ,)$overa
    })
```

solution

```
accuracy_data
```

```
##   k accuracy
## 1 1    0.950
## 2 2    0.940
## 3 3    0.945
## 4 4    0.955
## 5 5    0.965
```

```
(best = which.max(accuracy_data[,2]))
```

```
## [1] 5
```

solution

```
#b)
new.df<-data.frame(CRIM=0.2 , ZN = 0 , INDUS = 7 ,
                   CHAS = 0 , NOX = 0.538 , RM = 6 ,
                   AGE = 62 , DIS = 4.7 , RAD = 4 ,
                   TAX  =307 , PTRATIO = 21 , LSTAT = 10)

new.norm.df <-predict(norm.values, new.df)
class::knn(train_normal[,-13] ,test = new.df ,
           cl = train_data[,13] , k= best)
```

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

solution

```
#c)
knn.2<-class::knn(train_normal[,-13] ,
                  test = train_normal[,-13] ,
                  cl = train_data[,13] , k= best)
confusionMatrix(data = knn.2[1:nrow(train_normal)],
                factor(train_data[,13]) ,)$overall[1]
```

```
##  Accuracy
## 0.9542484
```

solution

d. when we calculated the best k with all of the data, and now we are calculating the accuracy for training data we are using the optimal k and for the all data accuracy we hade alot of data and the data coverage all over each but in calculating training accuracy we just have about 60% of data thus its Ok.

solution

e. if we want to predict MEDV for several thousands of new tracts we should do these steps.

steps

1. for each new observation we should find the new points distance with all of others training data(model data).
2. now we should sort the distances
3. thus, selecting the 1:k distances
4. find the k selected distance observation 5-1. if the response is categorical we will predict the new response that have the most repeat in the k response (voting). 5-2. if the response is numercial we will predict the new response as the meaning of k response.