Assignment2 Solution

Zhijin Zhou

4/28/2018

Assignment2

The file Housing.csv contains information on over 500 census tracts in Boston, where for each tract multiplevariables are recorded. The last column (CAT.MEDV) was derived from MEDV (the median value), such that it obtains the value “High” if MEDV > 30 and “Low” otherwise. Consider the goal of predicting CAT.MEDV of a tract, given the information in the first 12 columns. Use R to analyze the data.

```
housing.df <- read.csv("Housing.csv")
```

Question 1

Partition the data into 60% Training and 40% Validation with a seed of 5.

```
set.seed(5)
train.index <- sample(1:nrow(housing.df), 0.6*nrow(housing.df))
train.df <- housing.df[train.index, ]
valid.df <- housing.df[-train.index,]
```

Question 2

Use the training set to perform a k-NN classification with the first 12 predictors (ignore the MEDV column), trying values of k from 1 to 5. Make sure to first normalize the data. What is the best k chosen? What does this k mean?

Step 1: normalize the data

```
# initialize normalized training, validation data, complete data frames to originals

train.norm.df <- train.df
valid.norm.df <- valid.df
housing.norm.df <- housing.df
```

```r
#load package
library(caret)
norm.values <- preProcess(train.df[, 1:12], method=c("center", "scale"))

#normalize data
train.norm.df[, 1:12] <- predict(norm.values, train.df[, 1:12])
valid.norm.df[, 1:12] <- predict(norm.values, valid.df[, 1:12])
```

Step 2: Perform knn analysis for different values of k using `knn` function.

```r
# initialize a data frame with two columns: k, and accuracy to store the accuracy for
accuracy.df <- data.frame(k = 1:5, accuracy = rep(0, 5))

# compute knn for different k on validation.
library(FNN)
for(i in 1:5) {
  # use ?knn to find out more information
  # It worth noting that the input argument cl must be a factor!
  knn.pred <- knn(train.norm.df[, 1:12], valid.norm.df[, 1:12],
                  cl = train.norm.df[, 14], k = i)
  accuracy.df[i, 2] <- confusionMatrix(knn.pred, valid.norm.df[, 14])$overall[1]
}

#print out the accuracy matrix
accuracy.df
```

```
##   k  accuracy
## 1 1 0.9261084
## 2 2 0.9359606
## 3 3 0.8916256
## 4 4 0.9064039
## 5 5 0.8768473
```

k	accuracy
1	0.9261084
2	0.9359606
3	0.8916256
4	0.9064039
5	0.8768473

The best k is 2. This means that using the 2 nearest records gives the best prediction result.

Question 3

Predict the CAT.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

Step 1: First load the

```
# Predict for the new household using the best k (=2)
new.df <- data.frame(CRIM = 0.2, ZN = 0, INDUS = 7, CHAS = 0, NOX = 0.538, RM = 6, AG
```

Step 2: NORMALIZE the new record

```
new.norm.df <- predict(norm.values, new.df)
```

Step 3: Predict the outcome using the best k.

```
knn.pred.new <- knn(train.norm.df[, 1:12], new.norm.df,
                    cl = train.norm.df[, 14], k = 2)
# See the result
knn.pred.new[1]
```

```
## [1] Low
## Levels: Low
```

Question 4

If we use the above k-NN algorithm (with the best k) to predict CAT.MEDV for the training data, what would be the confusion matrix of the training dataset? What is the training accuracy?

```
knn.pred.train <- knn(train.norm.df[, 1:12], train.norm.df[, 1:12],
                      cl = train.norm.df[, 14], k = 2)
#Confusion Matrix
confusionMatrix(knn.pred.train, train.norm.df[, 14])$table
```

```
##            Reference
## Prediction High Low
##       High   46   9
##       Low     0 248
```

```
#Accuracy
confusionMatrix(knn.pred.train, train.norm.df[, 14])$overall[1]
```

```
## Accuracy
## 0.970297
```

Question 5

If the purpose is to predict CAT.MEDV for several billions 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.

The k-NN algorithm to make each prediction: (1) Calculate the distance between a new record and every record in the training set; (2) Based on the result of (1), determine the k-nearby records (k-nearest neighbors) of the new record; (3) Classify the new record as the predominant class among the k-nearby records.

Therefore, if we have several billions of new tracts, we need to repeat steps (1) ~ (3) billions of times. This might take long since step (1) requires much calculation time if we have a large training set.