Homework-4

Group 4

6/4/2020

Problem 2: Predicting Housing Median Prices k-NN

The file BostonHousing.xlsx contains information on over 500 census tracts in Boston, where for each tract 14 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 13 columns. Partition the data into training (60%) and validation (40%) sets.

(a) Perform a k-NN prediction with all 13 predictors (ignore the CAT.MEDV column), trying values of k from 1 to 5. Make sure to normalize the data (click "normalize input data"). What is the best k chosen? What does it mean?

```r
# Import the required packages
library(dplyr)
library(readxl)
library(e1071)
library(FNN)
library(caret)

# Import the BostonHousing.xlsx file
BostonHousing <- read_xlsx("BostonHousing.xlsx", sheet = "Data")

# Define the normalize function
normalize <- function(x) {
  return((x - min(x))) / (max(x) - min(x))
}

# Normalize the dataframe
df.norm <- as.data.frame(lapply(BostonHousing[1:13], normalize))

# Generate the training data indices
indices <- sample(seq_len(nrow(df.norm)), size = floor(0.6 * nrow(df.norm)))

# Get training and validation data
train_data <- df.norm[indices, ]
validation_data <- df.norm[-indices, ]

# Create a dataframe to keep track of k vs error
error.df <- data.frame("k" = 1:5, "error" = rep(0, 5))

# Loop for K = 1 to 5
```

```r
for (i in 1:5) {
  model <- knnreg(x = train_data[, 1:12], y = train_data[, 13], k = i)
  error.df[i, 2] <- RMSE(validation_data[, 13], predict(model, validation_data[, 1:12]))
}

# Get the K-value with the lowest RMSE error
best_k <- filter(error.df, error == min(error.df$error))$k
cat("The model with the best K is:", best_k, "\n")
```

```
## The model with the best K is: 4
```

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

```r
# Let's get the model with the best K
model <- knnreg(x = train_data[, 1:12], y = train_data[, 13], k = best_k)

# Create a dataframe for the given record
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
)

# Normalize the dataframe
df.norm <- as.data.frame(lapply(df[1:12], normalize))

# Predict the MEDV value for the new record.
prediction <- predict(model, df.norm)
cat("The MEDV prediction for the above record is:", prediction, "\n")
```

```
## The MEDV prediction for the above record is: 20.725
```

(c) Why is the error of the training data zero?

The error for training data will be zero at K = 1, since the single closest neighbor to the training sample vector will be itself. Hence the error will be zero.

```r
# Train the model with k = 1
model <- knnreg(x = train_data[, 1:12], y = train_data[, 13], k = 1)

# print
cat(
  "Error for Training Data at k = 1:",
  RMSE(train_data[, 13], predict(model, train_data[, 1:12])),
  "\n"
)
```

```
## Error for Training Data at k = 1: 0
```

```r
# remove all env variables
rm(list = ls())
```

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

We have chosen a model which can can perform best on the validation data. So it makes total sense that the error of the model on validation data is more optimistic as compared to it's error on 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.

- The disadvantage of using K-NN prediction for several thousand new tracts would be that the KNN algorithm will be slow. Calculating the euclidean distances for several thousand vectors would take a long time. Also, more sophisticated methods like Regression Trees and Neural Networks would perform better on such a large dataset.

The algorithm goes through the following operations in order to produce each operation -

For each record in the dataset, the algorithm -

1. Calculates the euclidean distance of that vector(record) with every other record in the dataset.
2. Sorts the euclidean distances from the lowest to the highest.
3. Takes the top K neighbors and then -
    - If it is a classification problem, it takes the classes of each of the K neighbors and assigns the majority class to the current record.
    - If it is a regression problem, it takes the average of the output variable of each of the K neighbors and assigns it to the current record.

Problem 3

The file Accidents.xlsx contains information on 42,183 actual automobile accidents in 2001 in the United States that involved one of three levels of injury: NO INJURY, INJURY, or FATALITY. For each accident, additional information is recorded, such as day of week, weather conditions, and road type. A firm might be interested in developing a system for quickly classifying the severity of an accident based on initial reports and associated data in the system (some of which rely on GPS-assisted reporting). Our goal here is to predict whether an accident just reported will involve an injury (MAX_SEV_IR = 1 or 2) or will not (MAX_SEV_IR = 0). For this purpose, create a dummy variable called INJURY that takes the value "yes" if MAX_SEV_IR = 1 or 2, and otherwise "no."

(a) Using the information in this dataset, if an accident has just been reported and no

further information is available, what should the prediction be? (INJURY = Yes or No?) Why?

```
# Import Accidents.xlsx
Accidents <- read_xlsx("Accidents.xlsx", sheet = "Data")

Accidents$INJURY <- Accidents$MAX_SEV_IR
Accidents <- Accidents %>%
  mutate_at(
    .vars = "INJURY",
    .funs = c(function(x) ifelse(x == "1" | x == "2", "Yes", "No"))
  )
```

```r
Accidents$INJURY <- as.factor(Accidents$INJURY)

# Look at the summary of Injury
summary(Accidents$INJURY)
```

```
##    No   Yes
## 20721 21462
```

Using the Naive Rule, we would predict a new accident to have an injury involved since INJURY = “yes” is more common than “no”.

(b) Select the first 12 records in the dataset and look only at the response (INJURY) and

the two predictors WEATHER_R and TRAF_CON_R.

i. Using Excel tools create a pivot table that examines INJURY as a function of the 2 predictors for these 12 records. Use all 3 variables in the pivot table as rows/columns, and use counts for the cells.

```r
# Select the required columns and the 12 records
df <- select(Accidents, c("WEATHER_R", "TRAF_CON_R", "INJURY"))[1:12, ]

# Create the pivot table
pivot_table <- as.data.frame(table(df$WEATHER_R, df$TRAF_CON_R, df$INJURY,
  dnn = c("WEATHER_R", "TRAF_CON_R", "INJURY")
))

print(pivot_table)
```

```
##    WEATHER_R TRAF_CON_R INJURY Freq
## 1          1          0     No    1
## 2          2          0     No    5
## 3          1          1     No    1
## 4          2          1     No    1
## 5          1          2     No    1
## 6          2          2     No    0
## 7          1          0    Yes    2
## 8          2          0    Yes    1
## 9          1          1    Yes    0
## 10         2          1    Yes    0
## 11         1          2    Yes    0
## 12         2          2    Yes    0
```

ii. Compute the exact Bayes conditional probabilities of an injury (INJURY =Yes) given the six possible combinations of the predictors.

```r
# Function to calculate probability manually
calculate_prob <- function(weather_r, traf_con_r) {
  filter(pivot_table, WEATHER_R == weather_r & TRAF_CON_R == traf_con_r & INJURY == "Yes")$Freq /
    sum(filter(pivot_table, WEATHER_R == weather_r & TRAF_CON_R == traf_con_r)$Freq)
}


cat("P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 0):", calculate_prob(1, 0), "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 0): 0.6666667
```

```r
cat("P(INJURY = Yes | WEATHER_R = 2, TRAF_CON_R = 0):", calculate_prob(2, 0), "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 2, TRAF_CON_R = 0): 0.1666667
```

```r
cat("P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 1):", calculate_prob(1, 1), "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 1): 0
```

```r
cat("P(INJURY = Yes | WEATHER_R = 2, TRAF_CON_R = 1):", calculate_prob(2, 1), "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 2, TRAF_CON_R = 1): 0
```

```r
cat("P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 2):", calculate_prob(1, 2), "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 2): 0
```

```r
cat("P(INJURY = Yes | WEATHER_R = 2, TRAF_CON_R = 2):", calculate_prob(2, 2), "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 2, TRAF_CON_R = 2): NaN
```

iii. Classify the 12 accidents using these probabilities and a cutoff of 0.5.

Since the cutoff is 0.5, the only combination of attributes that has probability greater than 0.5 is when WEATHER_R = 1 and TRAF_CON_R = 0

```r
# add a predictions column
df <- df %>%
  mutate(predictions = ifelse(WEATHER_R == 1 & TRAF_CON_R == 0, "Yes", "No"))

cat("The predictions are: \n")
```

```
## The predictions are:
```

```r
print(df)
```

```
## # A tibble: 12 x 4
##    WEATHER_R TRAF_CON_R INJURY predictions
##        <dbl>      <dbl> <fct>  <chr>
##  1         1          0 Yes    Yes
##  2         2          0 No     No
##  3         2          1 No     No
##  4         1          1 No     No
##  5         1          0 No     Yes
##  6         2          0 Yes    No
##  7         2          0 No     No
##  8         1          0 Yes    Yes
##  9         2          0 No     No
## 10         2          0 No     No
## 11         2          0 No     No
## 12         1          2 No     No
```

iv. Compute manually the naive Bayes conditional probability of an injury given WEATHER_R = 1 and TRAF_CON_R = 1.

By looking at the pivot table, we can calculate P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 1)

```r
# Manually calculate the probability
prob <- ((3 / 12) * ((2 / 3) * (0 / 3))) / (((3 / 12) * ((2 / 3) * (0 / 3))) + ((9 / 12) * ((3 / 9) * (

cat("P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 1):", prob, "\n")
```

```
## P(INJURY = Yes | WEATHER_R = 1, TRAF_CON_R = 1): 0
```

v. Run a naive Bayes classifier on the 12 records and 2 predictors. Obtain probabilities and classifications for all 12 records. Compare this to the exact Bayes classification. Are the resulting classifications equivalent? Is the ranking (= ordering) of observations equivalent?

```r
# Train the Naive Bayes Classifier
nb <- naiveBayes(INJURY ~ WEATHER_R + TRAF_CON_R, data = df[, 1:3])

pred.prob <- predict(nb, newdata = df[, 1:3], type = "raw")

pred.class <- data.frame(ifelse(pred.prob[, 1] - pred.prob[2] < 0, "Yes", "No"))
colnames(pred.class) <- "class"

actual_vs_predicted <- data.frame("actual" = df$INJURY, "predicted" = pred.class$class)
actual_vs_predicted$exact_bayes <- df$predictions
actual_vs_predicted$no_prob <- pred.prob[, 1]
actual_vs_predicted$yes_prob <- pred.prob[, 2]

cat("Actual vs Predicted Probabilities are: \n")
```

```
## Actual vs Predicted Probabilities are:
```

```r
print(actual_vs_predicted)
```

```
##    actual predicted exact_bayes     no_prob     yes_prob
## 1     Yes       Yes         Yes 0.001916916 0.9980830837
## 2      No        No          No 0.006129754 0.9938702459
## 3      No        No          No 0.999548668 0.0004513316
## 4      No        No          No 0.998552097 0.0014479028
## 5      No       Yes         Yes 0.001916916 0.9980830837
## 6     Yes        No          No 0.006129754 0.9938702459
## 7      No        No          No 0.006129754 0.9938702459
## 8     Yes       Yes         Yes 0.001916916 0.9980830837
## 9      No        No          No 0.006129754 0.9938702459
## 10     No        No          No 0.006129754 0.9938702459
## 11     No        No          No 0.006129754 0.9938702459
## 12     No        No          No 0.989399428 0.0106005719
```

The resulting classifications of the Naive Bayes Classifier and the Exact Bayes classifier are equivalent. Their ranking and ordering is also equivalent.

(c) Let us now return to the entire dataset. Partition the data into training/validation sets.

i. Assuming that no information or initial reports about the accident itself are available at the time of prediction (only location characteristics, weather conditions, etc.), which predictors can we include in the analysis? (Use the Data_Codes sheet.)

If no information about the accident itself is available, we can include these predictor in the analysis - HOUR_I_R, ALIGN_R, WRK_ZONE, WKDY_I_R, INT_HWY, LIGHTCON_I_R, REL_RWY_R, SPD_LIM, SUR_CON, TRAF_CON_R, WEATHER_R.

```r
# Generate the training data indices
df <- Accidents[, c(1, 3, 5, 6, 7, 8, 11, 12, 14, 15, 16, 20, 25)]
indices <- sample(seq_len(nrow(df)), size = floor(0.6 * nrow(df)))
```

```r
# Get training and validation data
train_data <- df[indices, ]
validation_data <- df[-indices, ]
```

ii. Run a naive Bayes classifier on the complete training set with the relevant predictors (and INJURY as the response). Note that all predictors are categorical. Show the classification matrix.

```r
nb <- naiveBayes(INJURY ~ ., train_data)

pred.class <- predict(nb, newdata = train_data)

cat("The Classification Matrix is :\n")
```

```
## The Classification Matrix is :
```

```r
confusionMatrix(data = pred.class, reference = train_data$INJURY)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    No   Yes
##        No  12417   273
##        Yes     0 12619
##
##                Accuracy : 0.9892
##                  95% CI : (0.9879, 0.9904)
##     No Information Rate : 0.5094
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9784
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.9788
##          Pos Pred Value : 0.9785
##          Neg Pred Value : 1.0000
##              Prevalence : 0.4906
##          Detection Rate : 0.4906
##    Detection Prevalence : 0.5014
##       Balanced Accuracy : 0.9894
##
##        'Positive' Class : No
##
```

iii. What is the overall error for the validation set?

```r
pred.class <- predict(nb, newdata = validation_data)

cf <- confusionMatrix(data = pred.class, reference = validation_data$INJURY)

cat("The error rate is :", paste0(round(100 * (1 - cf$overall[[1]]), 4), "%"), "\n")
```

```
## The error rate is : 1.1438%
```

iv. What is the percent improvement relative to the naive rule (using the validation set)?

```r
naive_cf <- confusionMatrix(
  data = as.factor(rep("Yes", dim(validation_data)[[1]])),
  reference = validation_data$INJURY
)

cat(
  "The difference between the Naive Bayes Classifier accuracy and the Naive Rule accuracy is:",
  paste0(round(100 * (cf$overall[[1]] - naive_cf$overall[[1]]), 4), "%"), "\n"
)
```

```
## The difference between the Naive Bayes Classifier accuracy and the Naive Rule accuracy is: 48.068%
```

v. Examine the conditional probabilities output. Why do we get a probability of zero for P(INJURY = No | SPD_LIM = 5)?

```r
pivot_table <- as.data.frame(table(validation_data$INJURY, validation_data$SPD_LIM,
  dnn = c("INJURY", "SPD_LIM")
))


cat(
  "P(INJURY = No | SPD_LIM = 5):",
  filter(pivot_table, SPD_LIM == 5 & INJURY == "No")$Freq /
    sum(filter(pivot_table, SPD_LIM == 5)$Freq)
)
```

```
## P(INJURY = No | SPD_LIM = 5): 0.6666667
```

```r
rm(list = ls())
```

The entire dataset of nearly 17000 records has only one case with INJURY = No and SPD_LIM = 5 so the probability is approximately 0.
