ASDM Workshop: Week 8 - K-Nearest Neighbour classifier

- Charith Silva, Dr.Mo Saraee

Part 1: K-Nearest Neighbour classifier (KNN)

This workshop aims to detect a room's occupancy from the measurements of below given Temperature, Humidity, Light and CO2 in the room. The data set is called **Occupancy Detection Data Set** and it can be downloaded from the Blackboard Week 8 folder.

Sample data point

- Temperature = 24.15
- Humidity = 27.2675
- Light = 429.5
- CO2 = 715.00

You may work through the workshop as follows:

1. Download the **Occupancy_Detection_Dataset.zip** file from the Blackboard and save it in to a folder on your F: drive (eg:F:\ASDM)

2. Extract the **Occupancy_Detection_Dataset.zip** file and save all the CSV files to a sub folder (eg: `F:\ASDM\Week8).`

3. Start **RStudio**.

4. Change the working directory
   `File` → `More` → `Go To Working Directory…`
   In the `Go To Working Directory` dialogue, navigate and select the folder where you saved your data file eg: `F:\ASDM\Week8`. Click OK.

5. Click `Set as Working Directory option`.

6. Open a new R script window:
   `File` → `New File` → `R script`

7. Import the data file named "datatraining.txt" and create a data frame
```
occupancy_data<- read.table("datatraining.txt", sep = ",")
```

8. Inspect the dataset in R
Once the file has been imported to R we often want to do few things to explore the dataset:

```
names(occupancy_data)
head(occupancy_data)
tail(occupancy_data)
summary(occupancy_data)
str(occupancy_data)
```

9. Check the dimension and number of points in the "occupancy_data" dataset

```
#number of data rows in the data frame can be determined by
the nrow() function.
nrow(occupancy_data)


#number of columns in the data frame can be determined by
the ncol() function.
ncol(occupancy_data)


#dim() function Retrieve the dimension of an object.
dim(occupancy_data)
```

```
> nrow(occupancy_data)
[1] 8143
> #number of columns in the data frame can be determined by the ncol() function.
> ncol(occupancy_data)
[1] 7
> #dim() function Retrieve the dimension of an object.
> dim(occupancy_data)
[1] 8143    7
```

10. Create a new dataset only with below attributes.
"Temperature", "Humidity", "Light", "CO2"," Occupancy"

```
occupancy_data_f <- occupancy_data[,c(2,3,4,5,7)]
occupancy_data_f
```

11. Create a new column to store Euclidean distances

```
#create a new column named "euclidean_distance" and fill with "NA"
occupancy_data_f$euclidean_distance <- NA
occupancy_data_f
```

12. Create new variables to hold predefined Temperature, Humidity, Light and CO2 values.

```
tem<- 24.15
hum<- 27.2675
lit<- 429.5
co2<- 715.00
```

13. Calculate the Euclidean distances from a query point to each of the points in the dataset.

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

```
length <- nrow(occupancy_data_f) # get number of rows

# Calculate the Euclidean distances

for(i in 1:length)
{
 occupancy_data_f$euclidean_distance[i] =  sqrt(
                    (occupancy_data_f$Temperature[i]-tem)^2+
                    (occupancy_data_f$Humidity[i]-hum)^2+
                    (occupancy_data_f$Light[i]-lit)^2+
                    (occupancy_data_f$CO2[i]-co2)^2
                                                )
 #print(occupancy_data_f$euclidean_distance[i])
}
```

14. Sort the “euclidean_distance” column in ascending order

```
occupancy_data_f<-
occupancy_data_f[order(occupancy_data_f$euclidean_distance),]
occupancy_data_f
```

```
> occupancy_data_f
     Temperature Humidity       Light       CO2 Occupancy euclidean_distance
2       23.15000 27.26750  429.500000  714.0000         1            1.414214
3       23.15000 27.24500  426.000000  713.5000         1            3.937068
1       23.18000 27.27200  426.000000  721.2500         1            7.228653
4       23.15000 27.20000  426.000000  708.2500         1            7.669228
887     21.34000 24.29000  433.000000  709.0000         1            8.062978
888     21.34000 24.29000  433.000000  721.5000         1            8.441659
8130    20.95875 35.68375  433.000000  712.6250         1            9.945248
8131    21.00000 35.65000  433.000000  719.0000         1           10.413396
5       23.10000 27.20000  426.000000  704.5000         1           11.117871
886     21.29000 24.29000  433.000000  704.5000         1           11.812921
889     21.32333 24.29000  426.666667  727.0000         1           12.995512
8129    20.91750 35.71750  433.000000  706.2500         1           13.063845
885     21.34000 24.22250  433.000000  703.0000         1           13.168832
2360    20.92667 20.39000  442.333333  712.0000         1           15.211322
2361    20.94500 20.50000  443.500000  710.0000         1           16.645452
7       23.10000 27.20000  419.000000  701.6667         1           17.003965
6       23.10000 27.20000  419.000000  701.0000         1           17.531602
8132    21.00000 35.70000  433.000000  730.2500         1           18.051096
```

Show the k nearest neighbours, here we use k=5

```
k <- 5
occupancy_data_f[1:k,]
```

```
> occupancy_data_f[1:k,]
    Temperature Humidity Light    CO2 Occupancy euclidean_distance
2         23.15  27.2675 429.5 714.00         1           1.414214
3         23.15  27.2450 426.0 713.50         1           3.937068
1         23.18  27.2720 426.0 721.25         1           7.228653
4         23.15  27.2000 426.0 708.25         1           7.669228
887       21.34  24.2900 433.0 709.00         1           8.062978
```

15. Determine the class label for the query data point

```
Class1 <- sum(occupancy_data_f$Occupancy[1:k]==1)

if (Class1 > k/2)
{
  print("The query point belongs to class 1")
} else
{
  print("The query point belongs to class 0")
}
```

Part 2: Exercises

Exercise 1 :

This exercise aims to detect a room's occupancy from the measurements of Temperature, Humidity, Light and CO2 in an office room, so smart building energy management system can be developed. The data set is called Occupancy Detection Data Set and it can be downloaded from Blackboard Week 8 folder. You should use "datatraining.txt" dataset to build the models and "datatest.txt", "datatest2.txt" datasets to validate the models.

1. Calculate classification accuracy percentage on test dataset 1 and 2 separately using K-Nearest Neighbour algorithm.

2. Calculate classification accuracy percentage on test dataset 1 and 2 separately using Decision Tree algorithm.

3. Compare the both set of classification accuracy percentages and find the best algorithm to develop smart building energy management system.

Exercise 2:

Repeat the exercise 1 with the normalised data and compare the accuracy percentages.

```
#--------------------ASDM Workshop 2019 - Part 2: Exercises 1 --------------
#This exercise aims to detect a room's occupancy from the measurements of
Temperature, Humidity,
#Light and CO2 in an office room, so smart building energy management system
can be developed.
#The data set is called Occupancy Detection Data Set and it can be
downloaded from Blackboard Week 3 folder.
#You should use "datatraining.txt" dataset to build the models and
"datatest.txt", "datatest2.txt" datasets to validate the models.

#########################################################################
############ 1. Calculate classification accuracy percentage on test dataset
1 and 2 separately using K-Nearest Neighbour algorithm.###################

#load training data
occupancy_data<- read.table("datatraining.txt",sep = ",")
occupancy_data

#Inspect the dataset
names(occupancy_data)
head(occupancy_data)
tail(occupancy_data)
summary(occupancy_data)
str(occupancy_data)

#Check the dimension
dim(occupancy_data)

#select only Temperature, Humidity, Light, CO2 and Occupancy columns
occupancy_data_f<-occupancy_data[,c(2,3,4,5,7)]
occupancy_data_f
head(occupancy_data_f)

#create new column to store euclidean distance
occupancy_data_f$euclidean_distance <- NA
occupancy_data_f

#load test data
occupancy_test<- read.table("datatest.txt",sep = ",")
occupancy_test

#select only Temperature, Humidity, Light, CO2 and Occupancy columns
occupancy_test_f<-occupancy_test[,c(2,3,4,5,7)]
occupancy_test_f

#create new column to store KNN Results
occupancy_test_f$knn_result <- NA
occupancy_test_f
```

```
length_test <- nrow(occupancy_test_f)
k <- 5


#Outer loop
for(j in 1:length_test)
{

  tem<- occupancy_test_f$Temperature[j]
  hum<- occupancy_test_f$Humidity[j]
  lit<- occupancy_test_f$Light[j]
  co2 <-occupancy_test_f$CO2[j]


  length <- nrow(occupancy_data_f)

  #create new column to store euclidean distance
  occupancy_data_f$euclidean_distance <- NA

  #Inner loop
  for(i in 1:length)
  {
    occupancy_data_f$euclidean_distance[i] =  sqrt(
      (occupancy_data_f$Temperature[i]-tem)^2+
        (occupancy_data_f$Humidity[i]-hum)^2+
        (occupancy_data_f$Light[i]-lit)^2+
        (occupancy_data_f$CO2[i]-co2)^2
    )

  }


  occupancy_data_f<-
occupancy_data_f[order(occupancy_data_f$euclidean_distance),]

  # determine the class label for the query data point
  Class1 <- sum(occupancy_data_f$Occupancy[1:k]==1)

  if (Class1 > k/2)
  {
     occupancy_test_f$knn_result[j]<-1
  }
  else
  {
     occupancy_test_f$knn_result[j]<-0
  }

}
```

```
#Calculate classification accuracy

correct = 0
for(l in 1:length_test)
{
  if(occupancy_test_f[l,5] == occupancy_test_f[l,6])
  {
    correct = correct+1
  }
}

Accuracy <-(correct/length_test) * 100

print(correct)
print(length_test)
print(Accuracy)


############################ K-Nearest Neighbour Using KNN Function in the
Class Package#########################################


#install.packages("class")
library(class)

#load training data
occupancy_training<- read.table("datatraining.txt",sep = ",")
occupancy_training

#load test data set 1
occupancy_test<- read.table("datatest.txt",sep = ",")
occupancy_test

#load test data set 2
occupancy_test2<- read.table("datatest2.txt",sep = ",")
occupancy_test2

#Inspect the occupancy_training data set
names(occupancy_training)
head(occupancy_training)

#Inspect the occupancy_test data set 1
names(occupancy_test)
head(occupancy_test)

#Inspect the occupancy_test data set 2
names(occupancy_test2)
head(occupancy_test2)
```

```
#Filter occupancy_training data set
occupancy_training_f<-occupancy_training[,c(2,3,4,5,7)]
head(occupancy_training_f)

#Filter occupancy_test data set
occupancy_test_f<-occupancy_test[,c(2,3,4,5,7)]
head(occupancy_test_f)

#Filter occupancy_test data set 2
occupancy_test_f2<-occupancy_test2[,c(2,3,4,5,7)]
head(occupancy_test_f2)

#convert the Occupancy variable to categorical form  in both data sets
occupancy_training_f$Occupancy2 <- as.factor(occupancy_training_f$Occupancy)
occupancy_test_f$Occupancy2 <- as.factor(occupancy_test_f$Occupancy)
occupancy_test_f2$Occupancy2 <- as.factor(occupancy_test_f2$Occupancy)

#inspect the structure of the data sets
str(occupancy_training_f)
str(occupancy_test_f)
str(occupancy_test_f2)


#test data set 1
knn_results <-knn(occupancy_training_f, occupancy_test_f,
occupancy_training_f$Occupancy2, k = 5, prob=TRUE)
test_predict_knn<-table(knn_results, occupancy_test_f$Occupancy2)
print(test_predict_knn)

#Calculate KNN classification accuracy and error on test data set.
sum(diag(test_predict_knn))/sum(test_predict_knn)
1-sum(diag(test_predict_knn))/sum(test_predict_knn)



# test data set 2
knn_results2 <-knn(occupancy_training_f, occupancy_test_f2,
occupancy_training_f$Occupancy2, k = 5, prob=TRUE)
test_predict_knn2<-table(knn_results2, occupancy_test_f2$Occupancy2)
print(test_predict_knn2)

#Calculate KNN classification accuracy and error on test data set 2.
sum(diag(test_predict_knn2))/sum(test_predict_knn2)
1-sum(diag(test_predict_knn2))/sum(test_predict_knn2)
```

```
#################### 2. Calculate classification accuracy percentage on test
dataset 1 and 2 separately using Decision Tree
algorithm.############################################################

#load training data
occupancy_training<- read.table("datatraining.txt",sep = ",")
occupancy_training

#load test data set 1
occupancy_test<- read.table("datatest.txt",sep = ",")
occupancy_test

#load test data set 2
occupancy_test2<- read.table("datatest2.txt",sep = ",")
occupancy_test2

#Inspect the occupancy_training data set
names(occupancy_training)
head(occupancy_training)

#Inspect the occupancy_test data set 1
names(occupancy_test)
head(occupancy_test)

#Inspect the occupancy_test data set 2
names(occupancy_test2)
head(occupancy_test2)

#Filter occupancy_training data set
occupancy_training_f<-occupancy_training[,c(2,3,4,5,7)]
head(occupancy_training_f)

#Filter occupancy_test data set
occupancy_test_f<-occupancy_test[,c(2,3,4,5,7)]
head(occupancy_test_f)

#Filter occupancy_test data set 2
occupancy_test_f2<-occupancy_test2[,c(2,3,4,5,7)]
head(occupancy_test_f2)

#convert the Occupancy variable to categorical form  in both data sets
occupancy_training_f$Occupancy2 <- as.factor(occupancy_training_f$Occupancy)
occupancy_test_f$Occupancy2 <- as.factor(occupancy_test_f$Occupancy)
occupancy_test_f2$Occupancy2 <- as.factor(occupancy_test_f2$Occupancy)

#inspect the structure of the data sets
str(occupancy_training_f)
str(occupancy_test_f)
```

```
str(occupancy_test_f2)

# install "party" package. Ignore instalation if "party" was already
installed. You can run library() to find out this.
#install.packages("party")
library(party)              # activate "party" package


#Train the tree
occupancy_tree <-
ctree(Occupancy2~Temperature+Humidity+Light+CO2,occupancy_training_f)
occupancy_tree

print(occupancy_tree)
plot(occupancy_tree)        # draw the tree

plot(occupancy_tree, type="simple")

#prediction on train data itself
tab<-table(predict(occupancy_tree), occupancy_training_f$Occupancy2)
print(tab)

#Calculate classification accuracy and error on train data itself.
sum(diag(tab))/sum(tab)
1-sum(diag(tab))/sum(tab)

#validate the model on test data set 1
test_predict <- table(predict(occupancy_tree, newdata= occupancy_test_f),
occupancy_test_f$Occupancy2)
print(test_predict)

#Calculate classification accuracy and error on test data set 1
sum(diag(test_predict))/sum(test_predict)
1-sum(diag(test_predict))/sum(test_predict)


#validate the model on test data set 2
test_predict2 <- table(predict(occupancy_tree, newdata= occupancy_test_f2),
occupancy_test_f2$Occupancy2)
print(test_predict2)


#Calculate classification accuracy and error on test data set2.
sum(diag(test_predict2))/sum(test_predict2)
1-sum(diag(test_predict2))/sum(test_predict2)
```

```
#############################################################################

#---------------ASDM Workshop 2019 - Part 2: Exercises 2 -----------------#
#Repeat the exercise 1 with the normalised data and compare the accuracy
percentages.


normalise <- function(df)
{
  return(((df- min(df)) /(max(df)-min(df))*(1-0))+0)
}


#Filter occupancy_training data set
occupancy_training_f<-occupancy_training[,c(2,3,4,5,7)]
occupancy_training_f<-as.data.frame(lapply(occupancy_training_f,normalise))
occupancy_training_f
head(occupancy_training_f)

#Filter occupancy_test data set
occupancy_test_f<-occupancy_test[,c(2,3,4,5,7)]
occupancy_test_f<-as.data.frame(lapply(occupancy_test_f,normalise))
occupancy_test_f
head(occupancy_test_f)

#Filter occupancy_test data set 2
occupancy_test_f2<-occupancy_test2[,c(2,3,4,5,7)]
occupancy_test_f2<-as.data.frame(lapply(occupancy_test_f2,normalise))
occupancy_test_f2
head(occupancy_test_f2)
```