Classification

Muhammad Shariq Azeem

10/06/2022

Classification:

Data:

For this assignment, I selected a data set that contains information about the room environment, such as, room temperature, humidity, CO2 levels, etc. I need to use that information to decide if the room is occupied or not.

Source for the data: https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+#

Note: The data sets provided through the link were not meeting the "at least 10K rows" requirement, so I used Excel to combine two data sets into one.

Cleaning the data:

- Got rid of the date column because I don't need that for my model.
- Converted 'Occupancy' attribute to a factor.

```
df <- read.csv("C:/Users/shari/Downloads/data.csv", header=T)
df <- df[,c(2,3,4,5,6,7)]
df$Occupancy <- factor(df$Occupancy)
```

Step A: Divide data into train and test

```
set.seed(1234)
i <- sample(1:nrow(df), 0.80*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]

knn.train <- train[,c(1,2,3,4,5)]
knn.test <- test[,c(1,2,3,4,5)]
knn.trainLabels <- train$Occupancy
knn.testLabels <- test$Occupancy
```

Step B: Statistical and Graphical Exploration on the training data

str():

str() tells us what type of data is stored in the table. In our case, training data consists of 5 number attributes and 1 attribute with two factors, 0 and 1.

```
str(train)
```

```
## 'data.frame':    9933 obs. of  6 variables:
##  $ Temperature  : num  20.9 21.4 22.6 21.8 21 ...
##  $ Humidity     : num  24.7 27.8 24.9 28.1 25.4 ...
##  $ Light        : num  0 0 732 0 14 0 0 454 433 0 ...
##  $ CO2          : num  572 566 588 594 522 ...
##  $ HumidityRatio: num  0.00377 0.00438 0.00423 0.00453 0.0039 ...
##  $ Occupancy    : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 2 2 1 ...
```

summary():

summary() gives us general statistics about the data. In our case, it tells us the minimum value, median/mean value, and maximum value of our quantitative attributes, and the counts of each factor of our qualitative attribute.

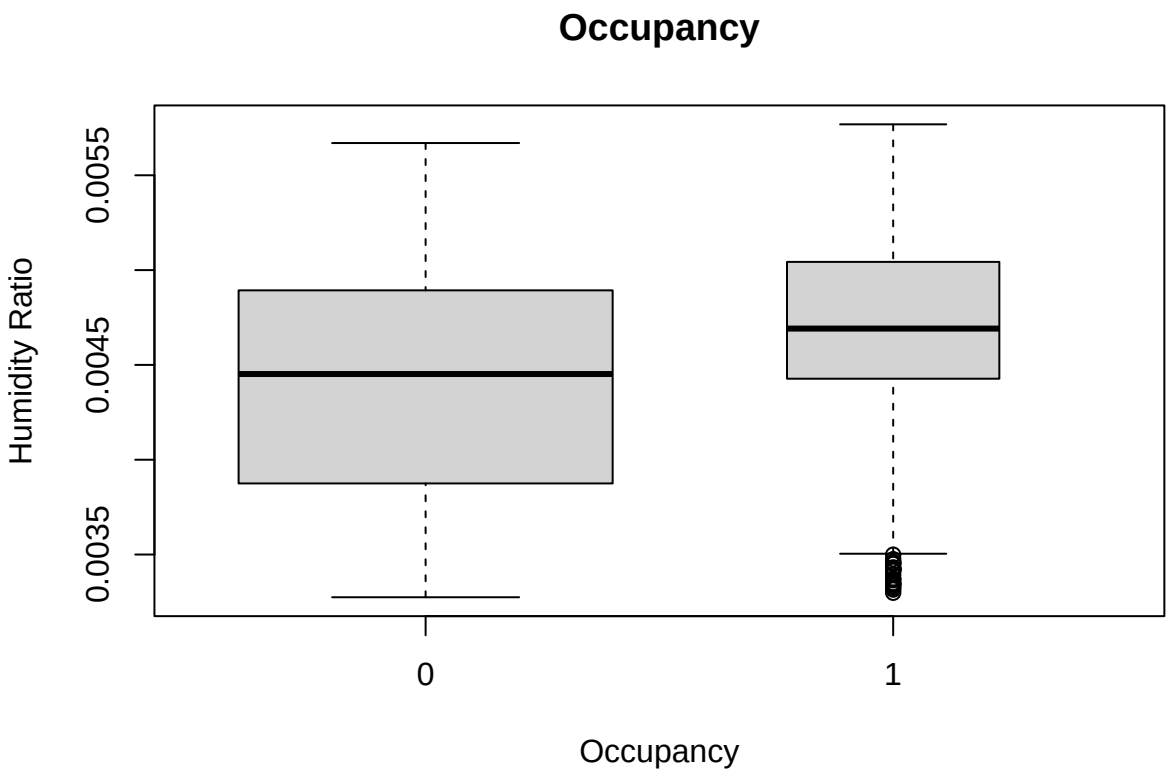
```
summary(train)
```

```
##   Temperature       Humidity         Light             CO2
##  Min.   :19.50   Min.   :21.86   Min.   :   0.0   Min.   : 427.5
##  1st Qu.:20.39   1st Qu.:25.39   1st Qu.:   0.0   1st Qu.: 528.0
##  Median :20.79   Median :28.63   Median :   0.0   Median : 632.7
##  Mean   :21.09   Mean   :28.92   Mean   : 138.1   Mean   : 745.6
##  3rd Qu.:21.67   3rd Qu.:31.86   3rd Qu.: 399.0   3rd Qu.: 857.0
##  Max.   :24.41   Max.   :39.50   Max.   :1581.0   Max.   :2076.5
##  HumidityRatio      Occupancy
##  Min.   :0.003275   0:7512
##  1st Qu.:0.003984   1:2421
##  Median :0.004512
##  Mean   :0.004468
##  3rd Qu.:0.004940
##  Max.   :0.005769
```

Box Plot:

The Box Plot below shows us that the Humidity Ratio increases, as the room goes from being empty to being occupied.

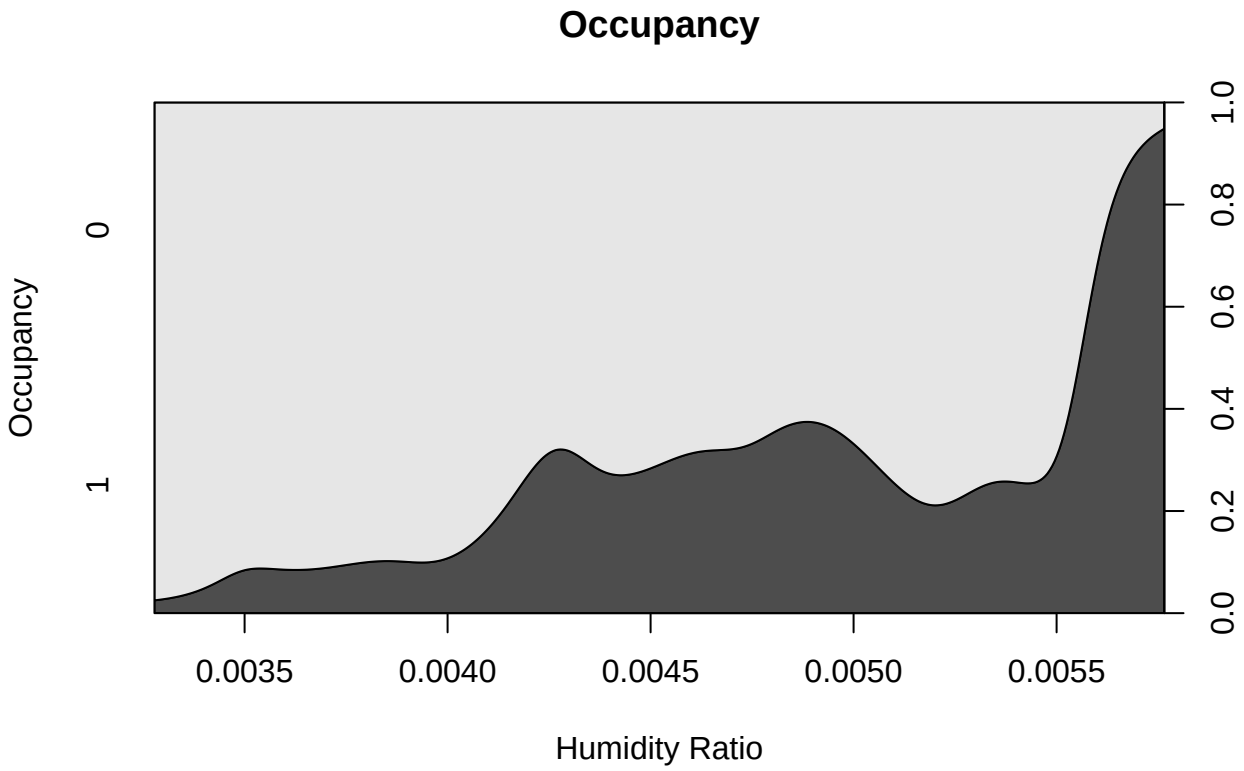
```
plot(HumidityRatio~Occupancy, data=train, main="Occupancy",
     xlab = "Occupancy", ylab = "Humidity Ratio", varwidth = TRUE)
```



CD Plot:

The conditional density (CD) plot below tells us the same thing as the box plot above, but it is just visualized differently. As the humidity ratio increases, there is more chance of the room being occupied.

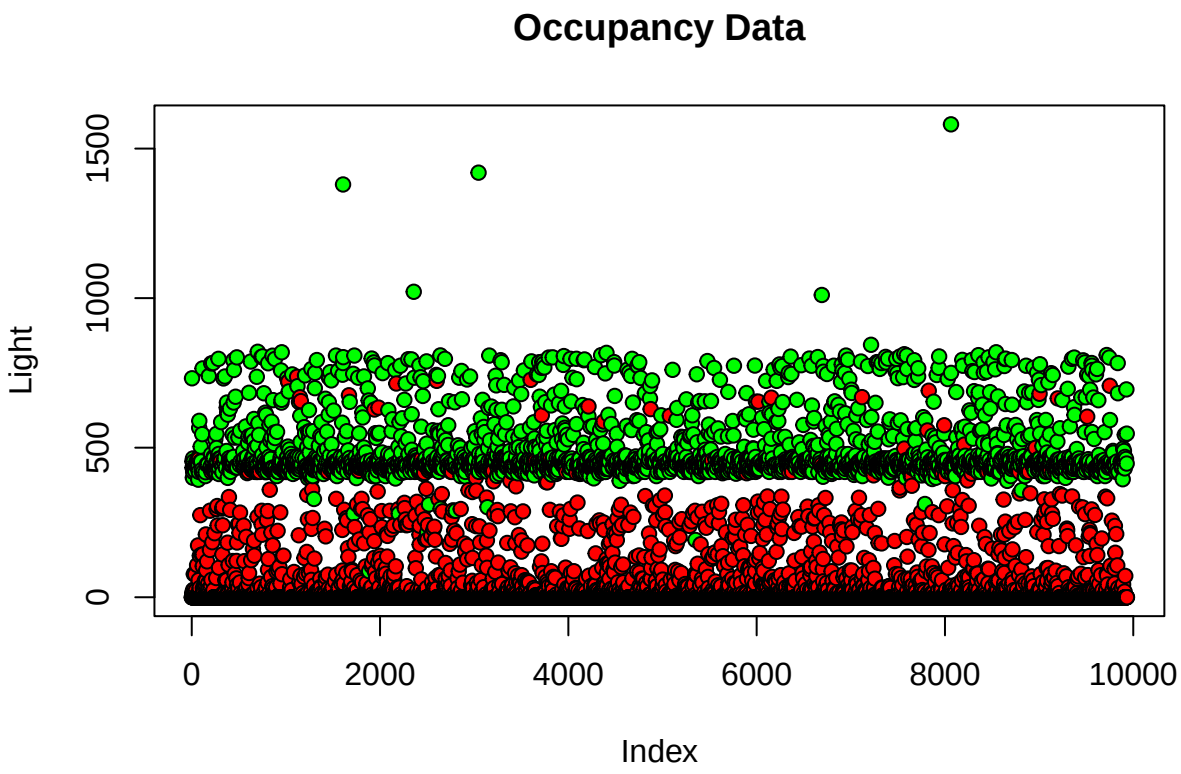
```
cdplot(train$Occupancy~train$HumidityRatio, main="Occupancy",
       ylab = "Occupancy", xlab = "Humidity Ratio")
```



Plot:

The plot below shows us that Light is a great predictor for predicting if a room is occupied or not because we can see a pretty good separation between empty rooms and occupied rooms, based on their Light value.

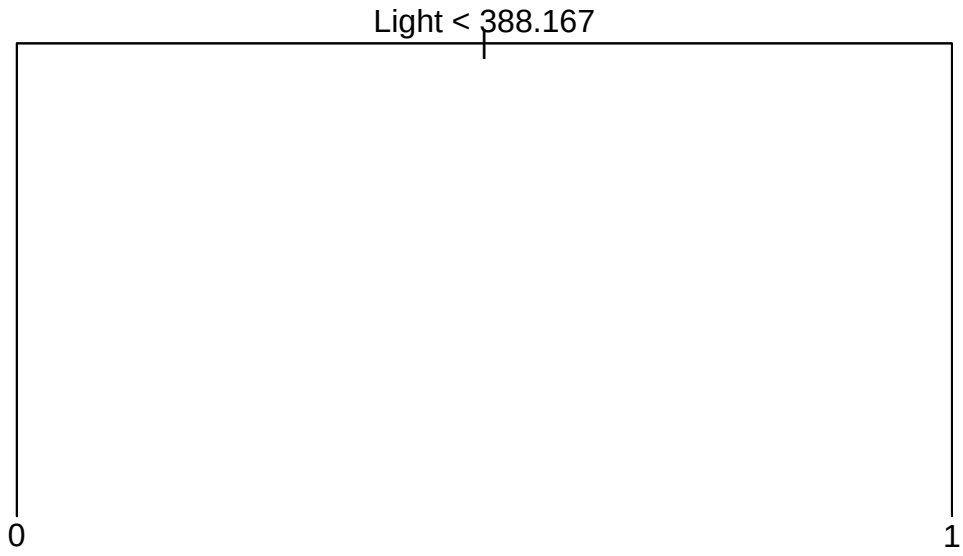
```
plot(train$Light, pch=21, bg=c("red","green")
     [unclass(train$Occupancy)], main="Occupancy Data", ylab = "Light")
```



Tree:

Even though I am predicting the Occupancy of a room based on all of its attributes, the tree below only has two branches, with the amount of light being the divider. This means that Light is a very good predictor of the room's occupancy, and adding the other predictors will not make the model much better.

```
library(tree)
tree <- tree(Occupancy~., data=train)
plot(tree)
text(tree, pretty=0)
```



Step C: Perform logistic regression

```
glm <- glm(Occupancy~., data=train, family="binomial")
probs <- predict(glm, newdata=test, type="response")
pred1 <- ifelse(probs > 0.5, 1, 0)

acc1 <- mean(pred1==test$Occupancy)
table(pred1, test$Occupancy)
```

```
##
## pred1    0    1
##     0 1871    2
##     1   13  598
```

```
print(paste("Logistic Regression Accuracy = ", acc1))
```

```
## [1] "Logistic Regression Accuracy =  0.993961352657005"
```

Step C: Perform kNN regression

Note: After checking several different k values, I selected 13 because it gives the best accuracy.

```
library(class)
pred2 <- knn(train = knn.train, test = knn.test, cl = knn.trainLabels, k = 13)

acc2 <- mean(pred2==knn.testLabels)
table(pred2, knn.testLabels)
```

```
##      knn.testLabels
## pred2    0    1
##     0 1875    3
##     1    9  597
```

```
print(paste("kNN Regression Accuracy = ", acc2))
```

```
## [1] "kNN Regression Accuracy =  0.995169082125604"
```

Step C: Perform Decision Tree regression

```
library(tree)
tree <- tree(Occupancy~., data=train)
pred3 <- predict(tree, newdata=test, type="class")

acc3 <- mean(pred3==test$Occupancy)
table(pred3, test$Occupancy)
```

```
##
## pred3    0    1
##     0 1871    1
##     1   13  599
```

```
print(paste("Decission Tree Regression Accuracy = ", acc3))
```

```
## [1] "Decission Tree Regression Accuracy =  0.994363929146538"
```

Step C: Compare the results

Note: The system considers an Unoccupied room to be the positive case.

Looking at the results for all three models above, we can see that all of them gave us almost 100% accuracy, with the kNN regression being the closest.

First, Logistic Regression Model had an overall accuracy of 99.396%, with 13 False Negatives and 2 False Positives.

Second, kNN Regression Model had an overall accuracy of 99.517%, with 9 False Negatives and 3 False Positives.

Lastly, Decision Tree Regression had an overall accuracy of 99.436%, with 13 False Negatives and only 1 False Positive.

Based on these metrics, it is safe to say that kNN algorithm is better at identifying True Positives, when compared to the other two models. Similarly, Decision Tree algorithm is better at identifying True Negatives.

Step D: Why these results were achieved

First, knowing how each of these algorithms work, I can say that the reason logistic regression had less accuracy than the other two models is that it tries to find a linear relationship between the predictors and the target variable, which can give us good results, but not better than the models that capture complex relationships without the assumption of linearity.

Next, it was surprising to see the decision tree algorithm to have around the same accuracy as the kNN algorithm because the decision tree is making the decision based on only one attribute. This makes me wonder how much affect the other attributes had on the results, and if it would be easier to just use that one predictor to classify the observations.

Now, between kNN and Decision Trees, if I would have chosen any other k value, then Decision Trees would have been more accurate; But since I selected $k = 13$, kNN became more accurate. This shows how important it is to select a good value for k because choosing a too high or too low value could make your model less accurate than other models.

Lastly, another reason kNN came out on top is that, before it made a decision, it looked at 13 nearest neighbors of an observation in a four dimensional space, which is better than predicting a class based on only one root, which is what the decision tree did.