MULTIMEDIA UNIVERSITY

TDS 3301 DATA MINING

ASSIGNMENT 3

CLASSIFICATION

GROUP DETAILS

NAME	ID	EMAIL
Darrel Shakri Bin Ahmad Shakri	1141327906	menubearer@gmail.com
Nur Nadhirah Bt. Nazarudin	1142700151	nazanadhirah@gmail.com
Andy Yong Jun Jie	1142701565	Andyyong9611@gmail.com
Kirbashini Naidu a/p Ragavelu	1141127226	Kirba4796@gmail.com

Dataset Choice

The dataset chosen for this assignment is the Occupation Detection Dataset located at:

https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+#

It is a dataset divided into 3 files, the training data and two test data. It has 7 attributes consisting of the class variable, the data variable representing the row identifier, and 5 continuous attributes.

The following table details the attributes:

NAME	FEATURE TYPE	ROLE	COMMENT
date	Date/Time, String	None/Identifier	Formatted according to year, month , day and then hour, minute, second
Temperature	Numeric,continuous	Descriptor	Feature is in Celsius
Humidity	Numeric,continuous	Descriptor	Feature is in percentage (%)
Light	Numeric,continuous	Descriptor	Feature is in Lux
CO2	Numeric,continuous	Descriptor	Feature is in parts-per-million (ppm)
HumidityRatio	Numeric,continuous	Descriptor	Derived quantity from temperature and humidity
Occupancy	Integer,discrete	Response	0 = not occupied 1 = occupied

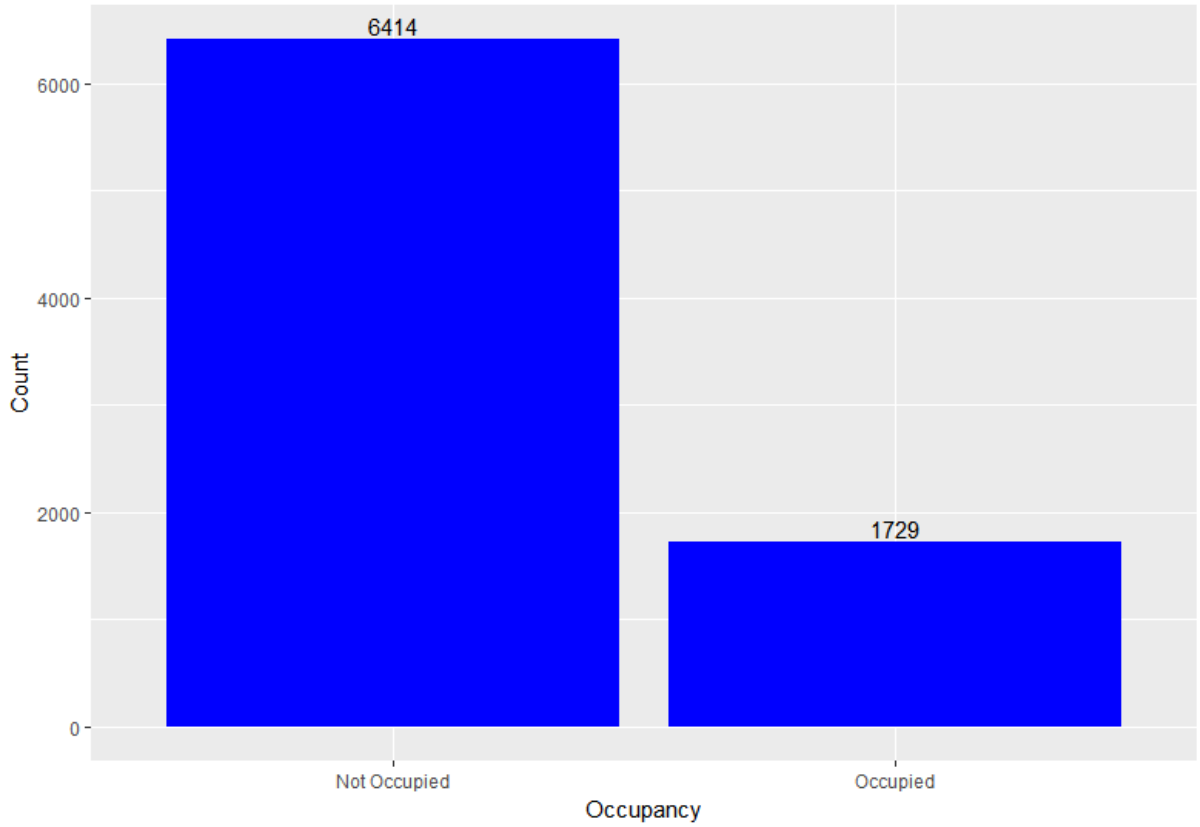
Exploratory Data Analysis

To load the training data for model construction, the following code is executed:

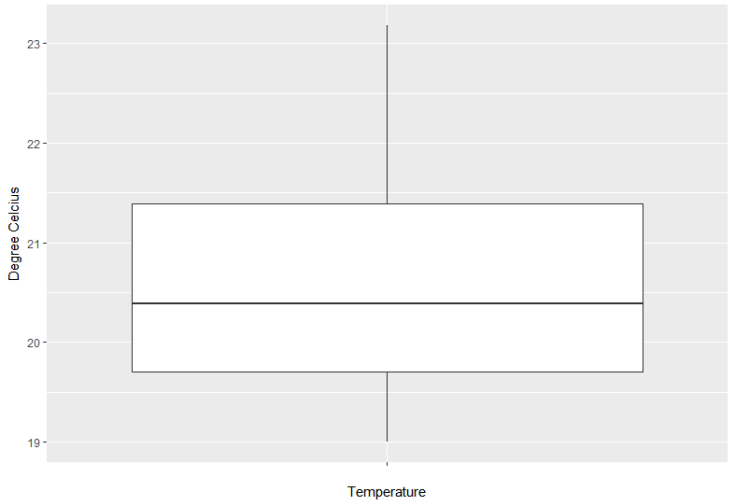
```
# load library
library(ggplot2) # plotting
library(caret) # scaling
library(rpart) # decision tree
library(rpart.plot) # plotting decision tree
library(ROCR) # ROC curve
library(arules) # discretize
library(e1071) # bayes
library('neuralnet') # ANN

# dates are not factors
datatraining <-
read.table("datatraining.txt",header=TRUE,sep=",",stringsAsFactors = FALSE)


# convert dates to POSIXct
datatraining$date <- as.POSIXct(strptime(datatraining$date,
"%Y-%m-%d %H:%M:%S"))
```

Once executed, the class distribution, and box-plot of the continuous variables are performed and shown below:

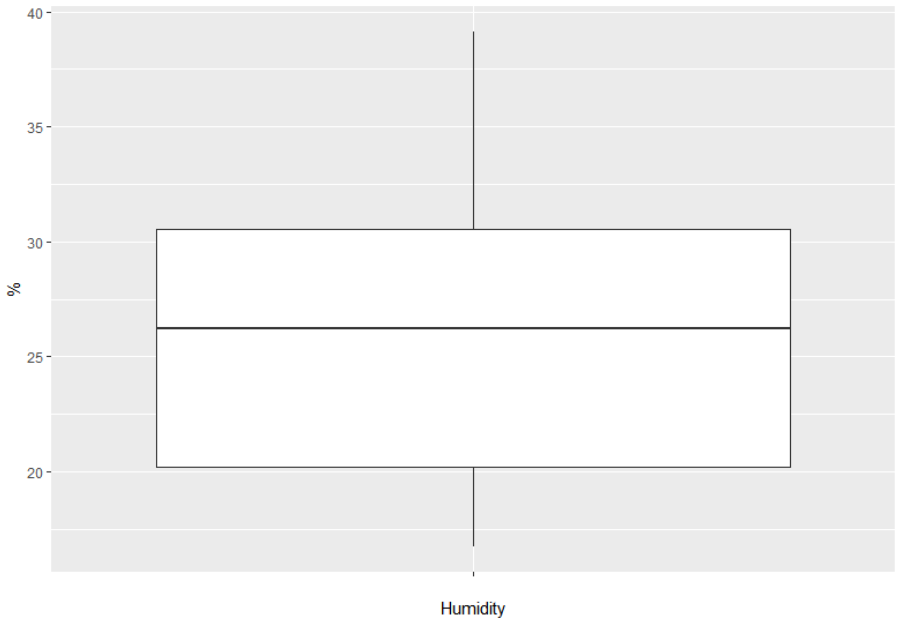
Class Distribution:



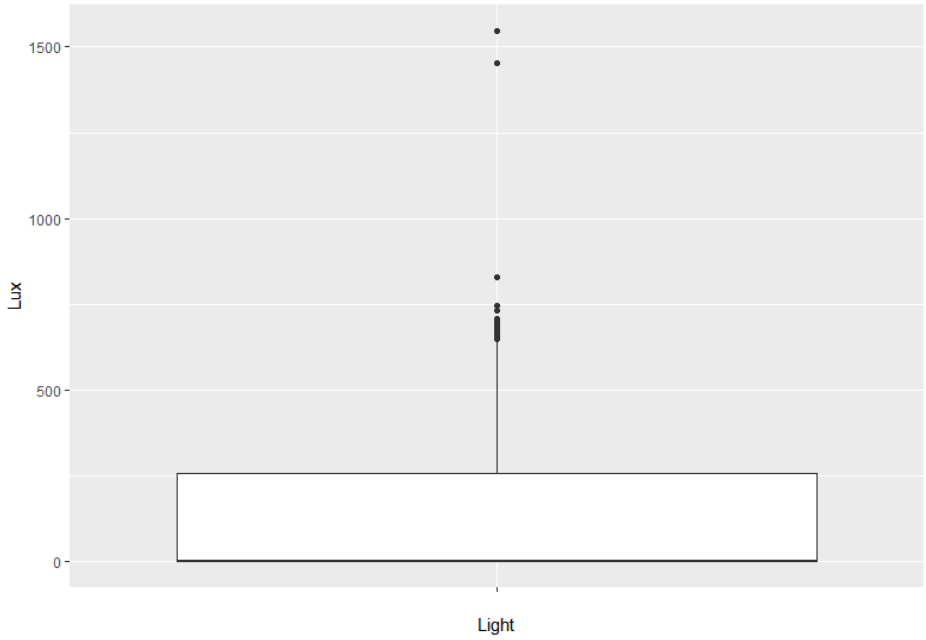
Temperature:



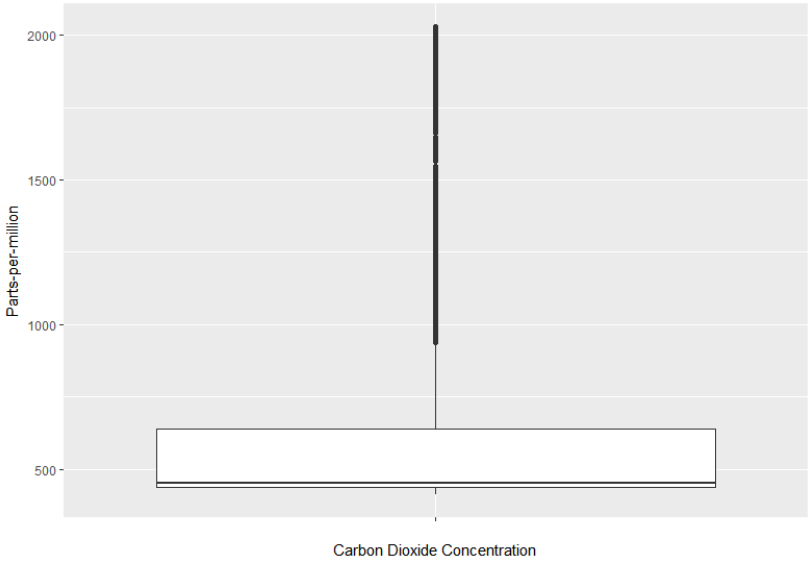
Humidity:



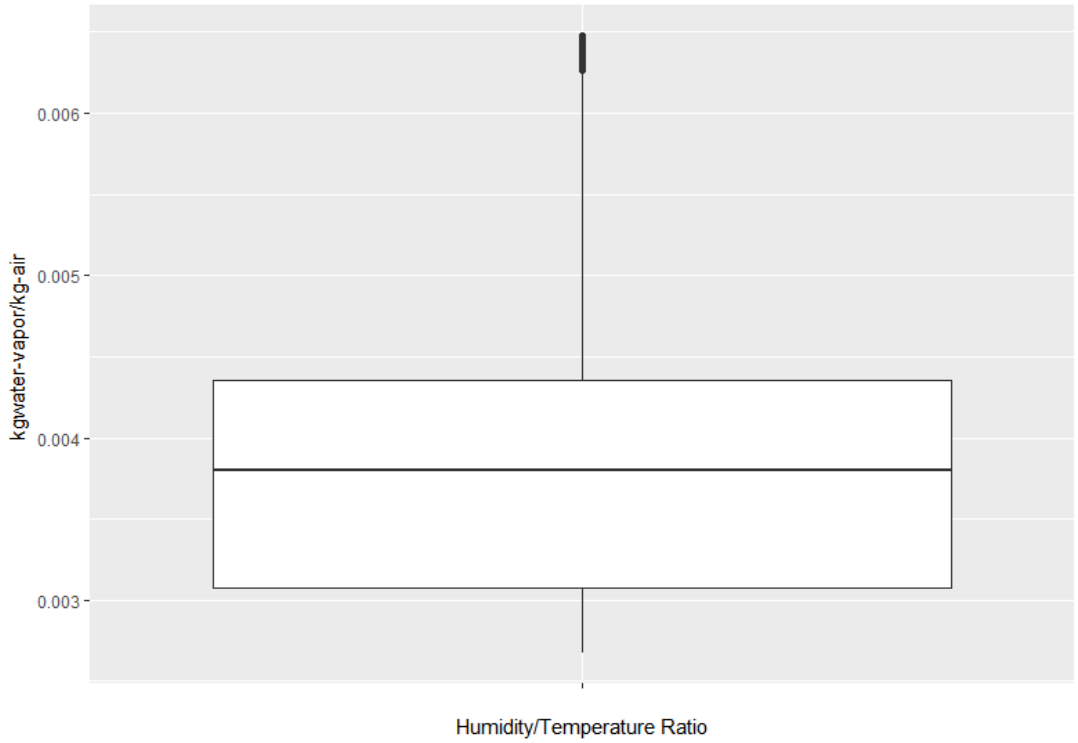
Light:



CO2:



HumidityRatio:



Pre-processing tasks

The training data seems to have no quality issues nor null values. Therefore, no data cleaning is necessary.

For **Decision Tree**, no pre-processing tasks are done because attribute splits are not affected by any kind of normalization.

For **Naive Bayes**, the continuous variables are discretized into categories based on the ranges of the continuous values. The same will be done to the test data.

For **Artifical Neural Network**, the continuous variables must be scaled between -1 and 1 and normalized to have a mean of 0 and standard deviation of 1. The same will be done to the test data.

The pre-processing tasks will be done on the section below, when constructing the model for testing.

Performance measures

For this exercise, we will do the ROC curve, and Sensivity/Specificity curve.

Constructing Classification Models

The following code creates the decision tree model:

```
# factorize the class variable
datatraining$Occupancy = as.factor(datatraining$Occupancy)


# create decision tree
dTree <- rpart(Occupancy ~ . - date, datatraining,control = rpart.control(cp
= 0.001))
prp(dTree, faclen = 0, cex = 0.8, extra = 1)

# pruning post-tree
bestcp <- dTree$cptable[which.min(dTree$cptable[,"xerror"]),"CP"]

# Step3: Prune the tree using the best cp.
tree.pruned <- prune(dTree, cp = bestcp)

prp(dTree, faclen = 0, cex = 0.8, extra = 1)
```

The following code creates the Naïve Bayes model:

```
datatrainingNB = datatraining

# discretize the continuous values
datatrainingNB$Temperature<- discretize(datatrainingNB$Temperature, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatrainingNB$Humidity <- discretize(datatrainingNB$Humidity, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatrainingNB$HumidityRatio <- discretize(datatrainingNB$HumidityRatio,
method = "interval",4,labels=c("Low","Medium","High","Very High"))
datatrainingNB$Light <- discretize(datatrainingNB$Light, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatrainingNB$CO2 <- discretize(datatrainingNB$CO2, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
```

```
bayesPred <- naiveBayes(Occupancy~ . -date, datatrainingNB)
```

The following code creates the ANN model:

```
# get maximum and minimum of column values
maximums <- apply(datatraining[,2:6],2, max)
minimums <- apply(datatraining[,2:6],2, min)

# begin scaling

# Use scale() and convert the resulting matrix to a data frame
datatrainingANN <- as.data.frame(scale(datatraining[,2:6],center = minimums,
scale = maximums - minimums))

# add the class variable to the ANN training
datatrainingANN <- cbind(Occupancy=as.numeric(datatraining$Occupancy)-1,
datatrainingANN)


# getting the formula
feats <- names(datatrainingANN)
# Concatenate strings
f <- paste(feats[2:6],collapse=' + ')
f <- paste(feats[1], ' ~ ',f)

# Convert to formula
f <- as.formula(f)



nn <- neuralnet(f,datatrainingANN,hidden=c(3,2,1), linear.output = FALSE)
```

Testing and Performance

The following code loads the test data (testdata.txt) and use the constructed decision tree for predicting the class attribute:

```
# load test data
datatestTREE <-
read.table("datatest2.txt",header=TRUE,sep=",",stringsAsFactors = FALSE)


# convert dates to POSIXct
datatestTREE$date <- as.POSIXct(strptime(datatestTREE$date,
"%Y-%m-%d %H:%M:%S"))

# do prediction
predT <- predict(dTree, newdata = datatestTREE, type = 'class')
```

```
# get confusion matrix
confusionMatrix(predT, datatestTREE$Occupancy)

# get ROC curve
# need to turn factors to binary values
roc_predDT <- ROCR::prediction(as.numeric(predT),
as.numeric(datatestTREE$Occupancy))
plot(performance(roc_predDT, measure="tpr", x.measure="fpr"), colorize=TRUE)

# precision recall curve
plot(performance(roc_predDT, measure="sens", x.measure="spec"),
colorize=TRUE)
```

Then we do the same with the Naive Bayes models, including discretizing the test data:

```
# load test data
datatestNB <- read.table("datatest2.txt",header=TRUE,sep=",",stringsAsFactors
= FALSE)

# factorize the class variable
datatestNB$Occupancy = as.factor(datatestNB$Occupancy)

# discretize the continuous values
datatestNB$Temperature<- discretize(datatestNB$Temperature, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatestNB$Humidity <- discretize(datatestNB$Humidity, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatestNB$HumidityRatio <- discretize(datatestNB$HumidityRatio, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatestNB$Light <- discretize(datatestNB$Light, method =
"interval",4,labels=c("Low","Medium","High","Very High"))
datatestNB$CO2 <- discretize(datatestNB$CO2, method =
"interval",4,labels=c("Low","Medium","High","Very High"))

# convert dates to POSIXct
datatestNB$date <- as.POSIXct(strptime(datatestNB$date, "%Y-%m-%d %H:%M:%S"))

# do prediction
predB <- predict(bayesPred, newdata = datatestNB[,2:6])

# get confusion matrix
confusionMatrix(predB, datatestNB$Occupancy)

# get ROC curve
# need to turn factors to binary values
roc_predNB <- ROCR::prediction(as.numeric(predB),
as.numeric(datatestNB$Occupancy))
plot(performance(roc_predNB, measure="tpr", x.measure="fpr"), colorize=TRUE)

# precision recall curve
plot(performance(roc_predNB, measure="sens", x.measure="spec"),
colorize=TRUE)
```

Finally, we load the test data, and modify it so that the ANN can be used for prediction:

```
# load test data
datatestANN <-
read.table("datatest2.txt",header=TRUE,sep=",",stringsAsFactors = FALSE)

# numerize the class variable
datatestANN$Occupancy = as.numeric(datatestANN$Occupancy)

# get maximum and minimum of column values
maximums <- apply(datatestANN[,2:6],2, max)
minimums <- apply(datatestANN[,2:6],2, min)

# begin scaling
# Use scale() and convert the resulting matrix to a data frame
scaled.dataANN <- as.data.frame(scale(datatestANN[,2:6],center = minimums,
scale = maximums - minimums))

# add the class variable to the ANN training
datatestANN <- cbind(Occupancy=datatestANN$Occupancy, scaled.dataANN )

# prediction
predN <- compute(nn,datatestANN[,2:6])

# round off results
predN <- sapply(predN$net.result,round,digits=0)



# get confusion matrix
confusionMatrix(predN, datatestANN$Occupancy)
```

The following graphs show the performance measures: