ENGSCI 255 Assignment 3

Mihnea Vlad (ID: 736248940)

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
source("ggpairs.R")
library(rpart)
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 4.1.3

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.3

## randomForest 4.7-1
```

```
## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:rattle':
##
##     importance

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(klaR)
```

```
## Warning: package 'klaR' was built under R version 4.1.3

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
# import required data
prices <- read_csv("prices.csv", col_names = TRUE)
```

```
## Rows: 35088 Columns: 5

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): TradingDate
## dbl (4): TradingPeriod, Benmore, Haywards, Otahuhu

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
banknotes <- read_csv("banknotes.csv", col_names = TRUE)
```

```
## Rows: 1372 Columns: 5
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): class
## dbl (4): var, skew, kurtosis, entropy

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
banknotesDis <- read_csv("banknotes-discrete.csv", col_names = TRUE)
```

```
## Rows: 1372 Columns: 5

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): class
## dbl (4): var, skew, kurtosis, entropy

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Question 1a)

```
prices$TradingDate = dmy(prices$TradingDate)
head(prices)
```

```
## # A tibble: 6 x 5
##   TradingDate TradingPeriod Benmore Haywards Otahuhu
##   <date>              <dbl>   <dbl>    <dbl>   <dbl>
## 1 2019-01-01              1    180.     182.    187.
## 2 2019-01-01              2    160.     162.    167.
## 3 2019-01-01              3    180.     181.    186.
## 4 2019-01-01              4    162.     164.    168
## 5 2019-01-01              5    151.     152.    156.
## 6 2019-01-01              6    146.     147.    151.
```

Question 1b)

```
prices_longer <- prices %>%
                 pivot_longer(c(Benmore, Haywards, Otahuhu),
                              names_to = "Nodes", values_to = "Prices")
head(prices_longer)
```

```
## # A tibble: 6 x 4
##   TradingDate TradingPeriod Nodes    Prices
##   <date>              <dbl> <chr>     <dbl>
```

```
## 1 2019-01-01             1 Benmore    180.
## 2 2019-01-01             1 Haywards   182.
## 3 2019-01-01             1 Otahuhu    187.
## 4 2019-01-01             2 Benmore    160.
## 5 2019-01-01             2 Haywards   162.
## 6 2019-01-01             2 Otahuhu    167.
```

Question 1c)

```
prices_longer <- prices_longer %>%
                 mutate(Year = year(prices_longer$TradingDate),
                        Month = month(prices_longer$TradingDate, label = TRUE),
                        Quarter = quarter(prices_longer$TradingDate,
                                          fiscal_start = 1))

prices_longer$Quarter <- paste("Q", prices_longer$Quarter, sep = '')

head(prices_longer)
```

```
## # A tibble: 6 x 7
##   TradingDate TradingPeriod Nodes    Prices  Year Month Quarter
##   <date>              <dbl> <chr>     <dbl> <dbl> <ord> <chr>
## 1 2019-01-01              1 Benmore    180.  2019 Jan   Q1
## 2 2019-01-01              1 Haywards   182.  2019 Jan   Q1
## 3 2019-01-01              1 Otahuhu    187.  2019 Jan   Q1
## 4 2019-01-01              2 Benmore    160.  2019 Jan   Q1
## 5 2019-01-01              2 Haywards   162.  2019 Jan   Q1
## 6 2019-01-01              2 Otahuhu    167.  2019 Jan   Q1
```

Question 1d)

```
prices_Benmore <- prices_longer %>%
                  filter(Nodes == 'Benmore') %>%
                  group_by(Month, Year) %>%
                  summarise(MeanPrice = mean(Prices)) %>%
                  pivot_wider(names_from = Year, values_from = MeanPrice)

prices_Benmore
```

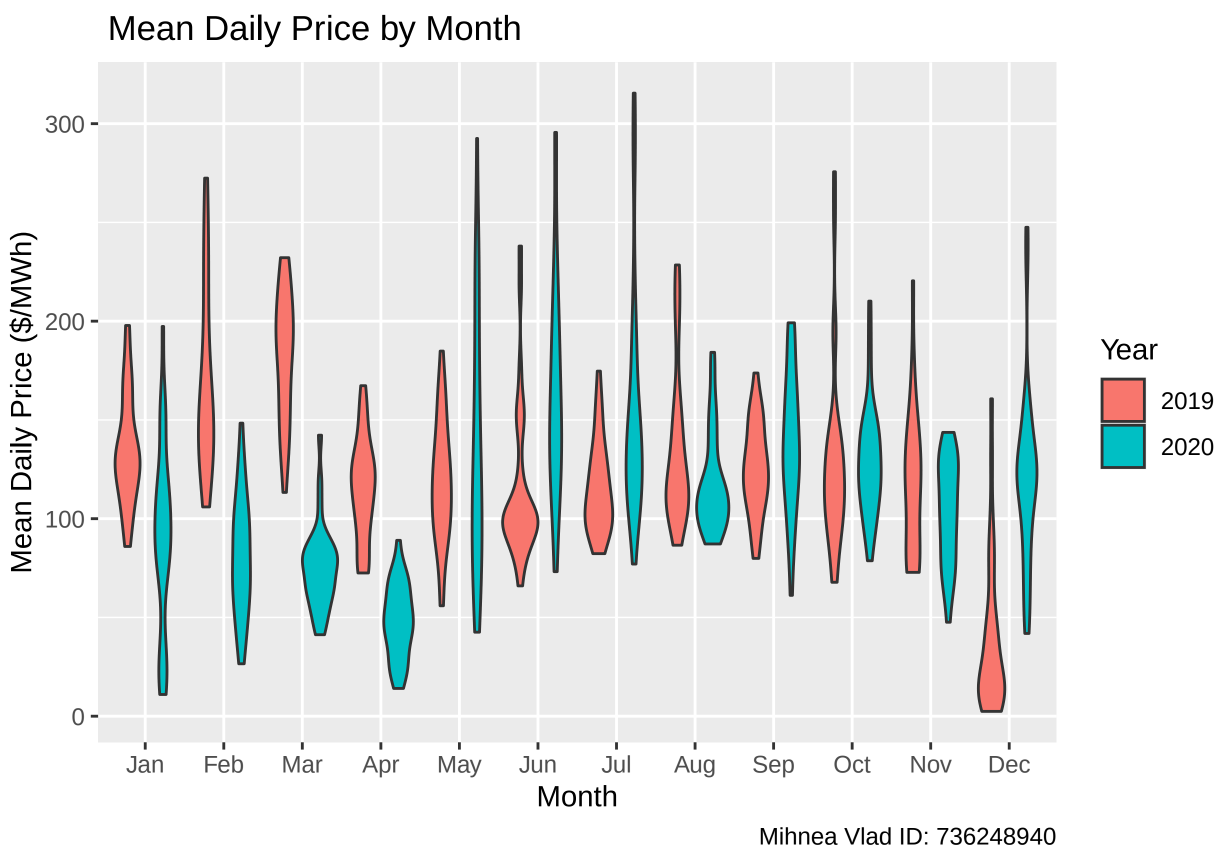
```
## # A tibble: 12 x 3
## # Groups:   Month [12]
##    Month `2019` `2020`
##    <ord>  <dbl>  <dbl>
##  1 Jan    122.    76.2
##  2 Feb    149.    37.8
##  3 Mar    165.    42.2
##  4 Apr     96.8   42.1
##  5 May     94.9  113.
##  6 Jun     94.1  151.
```

```
## 7 Jul     98.6  139.
## 8 Aug    120.   107.
## 9 Sep    119.   128.
## 10 Oct    119.   114.
## 11 Nov     93.7   89.5
## 12 Dec     24.9  107.
```

The average price in months 1-4 is significantly lower in 2020 than in 2019. This may be due to the impacts COVID-19 and the nationwide lock-down, reducing overall demand for electricity in Benmore as companies were closed, pushing the price down.

Question 1e)

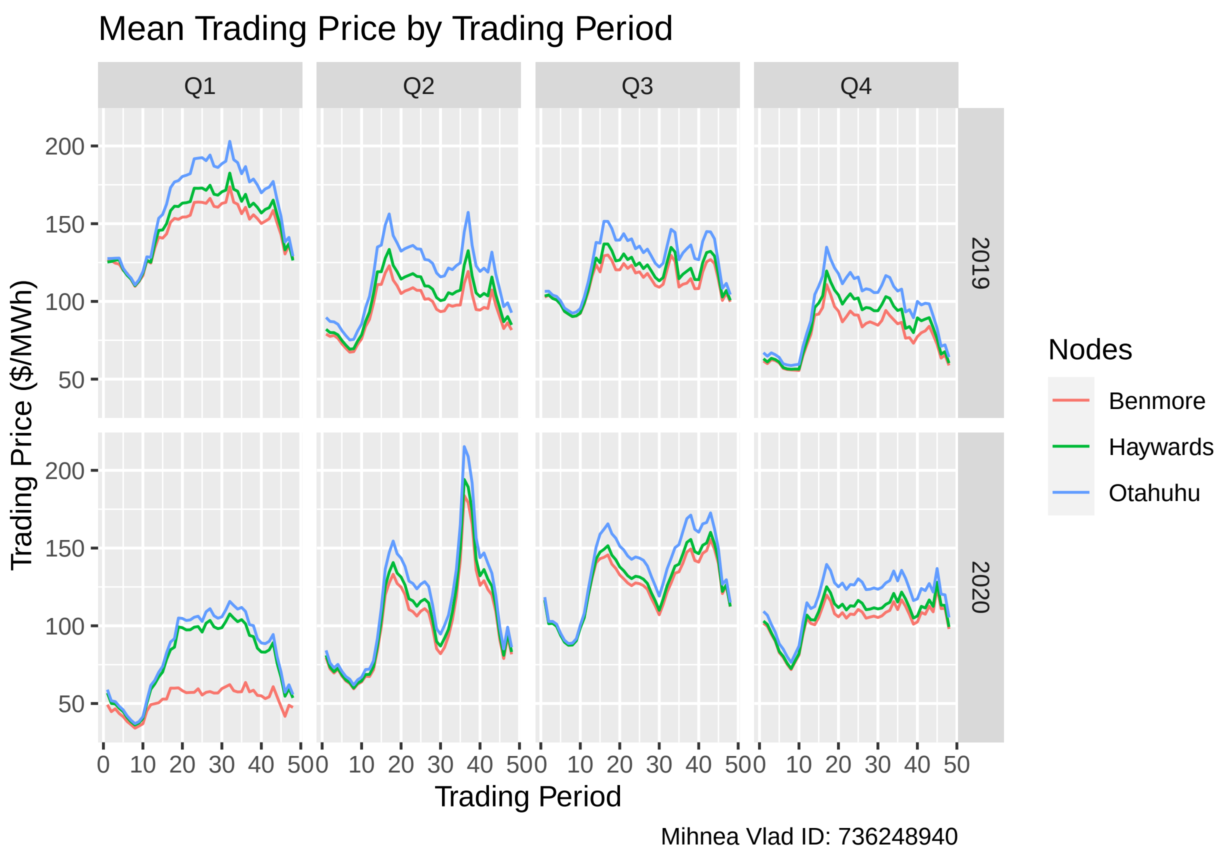
```
prices_longer <- prices_longer %>%
                 mutate(Day = day(prices_longer$TradingDate))
prices_longer$Year <- as.factor(prices_longer$Year)

OtahuhuPlot <- prices_longer %>%
               filter(Nodes == 'Otahuhu') %>%
               group_by(Day, Month, Year) %>%
               summarise(DailyPrice = mean(Prices)) %>%
               ggplot(aes(x=Month,y=DailyPrice)) +
               geom_violin(aes(fill=Year)) +
               labs(title=" Mean Daily Price by Month",
                    x="Month", y="Mean Daily Price ($/MWh)",
                    caption = "Mihnea Vlad ID: 736248940" )

OtahuhuPlot
```



In the months of Jan-May, the 2019 distributions are shifted higher than the 2020 distributions. There tends to be an gradual upward shift in the distributions of daily prices each month starting from month 5 and peaking in month 8. After month 8, the distribution start moving downwards again. This upward shift may be caused by people/companies in Otahuhu demanding more electricity for heating over the winter months, pushing the price up.

Question 1f)

```
TradingPlot <- prices_longer %>%
               filter(TradingPeriod < 49) %>%
               group_by(TradingPeriod, Quarter, Year, Nodes) %>%
               summarise(TradingPrice = mean(Prices)) %>%
               ggplot() +
               geom_line(aes(x=TradingPeriod, y=TradingPrice, color=Nodes)) +
               labs(title='Mean Trading Price by Trading Period',
                    x='Trading Period', y='Trading Price ($/MWh)',
                    caption = "Mihnea Vlad ID: 736248940") +
               facet_grid(Year~Quarter)

TradingPlot
```



All 3 nodes follow essentially the same pattern over the 48 trading periods. Benmore experiences unusually low prices in quarter 1 of 2020. The order of the nodes also seems to stay relatively consistent throughout the day with Benmore always having the lowest average trading price in a quarter and Otahuhu having the highest. The most notable feature across all of the plots is the dip in trading price from trading period 0 until approximately trading period 10 before the trading price begins to rise. This is explained by the time of day these trading periods correspond to (Trading periods 1-10 correspond to 12am - 5am). Most people will be sleeping at this time of the day so there will be little activity within the towns, reducing for demand for electricity, pushing the price down. As people begin to wake up after trading period 10, activity resumes and demand for electricity increases, pushing the price up again.

Question 1g)

```r
prices_longer <- prices_longer %>%
                 mutate(Day = ifelse((wday(TradingDate, label = TRUE) == 'Sat' |
                                      wday(TradingDate, label = TRUE) == 'Sun'),
                                     'Weekend', 'Weekday'),
                        DayLabel = wday(TradingDate, label = TRUE))

prices_longer %>% slice(1:6)
```

```
## # A tibble: 6 x 9
##   TradingDate TradingPeriod Nodes    Prices Year  Month Quarter Day     DayLabel
##   <date>              <dbl> <chr>     <dbl> <fct> <ord> <chr>   <chr>   <ord>
## 1 2019-01-01              1 Benmore    180. 2019  Jan   Q1      Weekday Tue
```

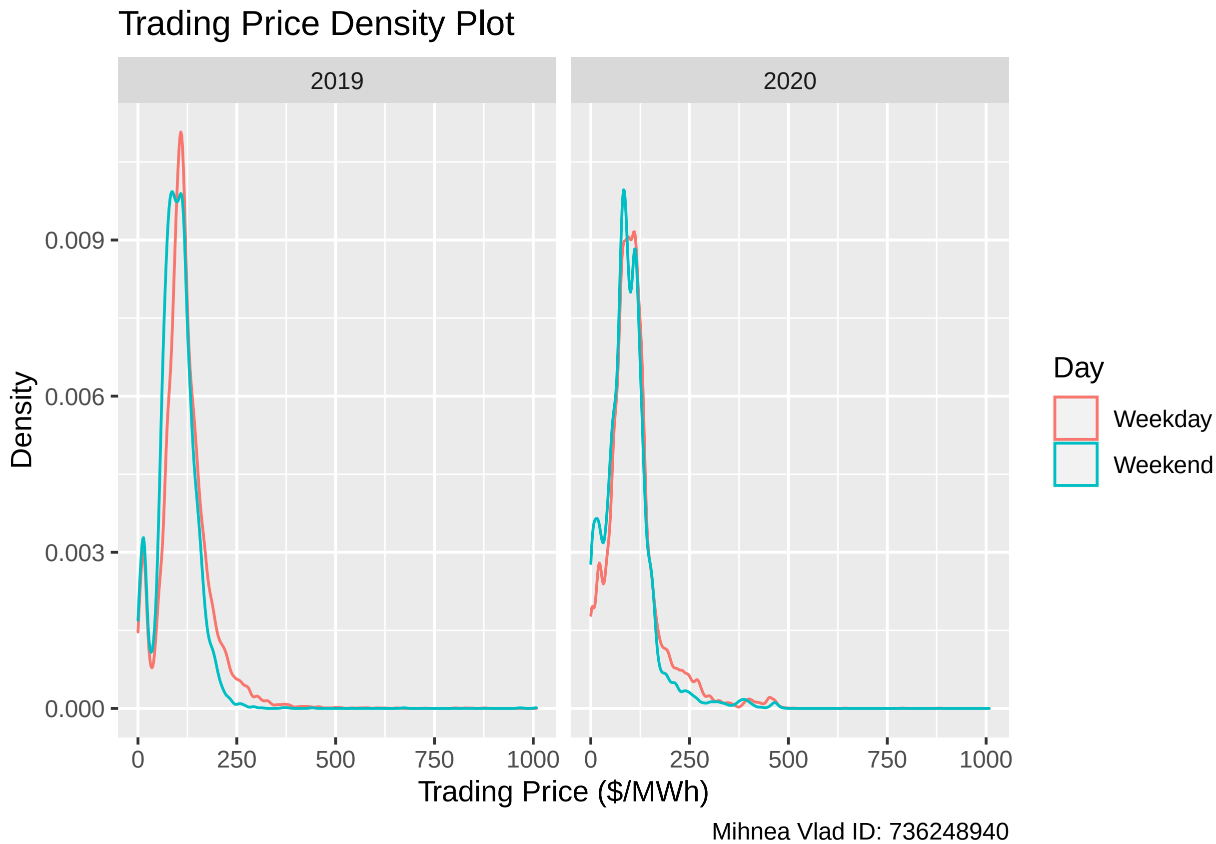
```
## 2 2019-01-01              1 Haywards   182. 2019  Jan   Q1      Weekday Tue
## 3 2019-01-01              1 Otahuhu    187. 2019  Jan   Q1      Weekday Tue
## 4 2019-01-01              2 Benmore    160. 2019  Jan   Q1      Weekday Tue
## 5 2019-01-01              2 Haywards   162. 2019  Jan   Q1      Weekday Tue
## 6 2019-01-01              2 Otahuhu    167. 2019  Jan   Q1      Weekday Tue
```

```
prices_longer %>% slice(577:582)
```

```
## # A tibble: 6 x 9
##   TradingDate TradingPeriod Nodes    Prices Year  Month Quarter Day     DayLabel
##   <date>              <dbl> <chr>     <dbl> <fct> <ord> <chr>   <chr>   <ord>
## 1 2019-01-05              1 Benmore    147. 2019  Jan   Q1      Weekend Sat
## 2 2019-01-05              1 Haywards   149. 2019  Jan   Q1      Weekend Sat
## 3 2019-01-05              1 Otahuhu    154. 2019  Jan   Q1      Weekend Sat
## 4 2019-01-05              2 Benmore    147. 2019  Jan   Q1      Weekend Sat
## 5 2019-01-05              2 Haywards   149. 2019  Jan   Q1      Weekend Sat
## 6 2019-01-05              2 Otahuhu    153  2019  Jan   Q1      Weekend Sat
```

Question 1h)

```
prices_longer$Year <- as.factor(prices_longer$Year)


DayPlot <- prices_longer %>%
           filter(Nodes == 'Haywards') %>%
           group_by(Day, Year) %>%
           ggplot(aes(x=Prices, color=Day)) +
           geom_density() +
           labs(title='Trading Price Density Plot',
                x='Trading Price ($/MWh)', y='Density',
                caption = "Mihnea Vlad ID: 736248940") +
           facet_wrap(~Year, ncol = 2)

DayPlot
```



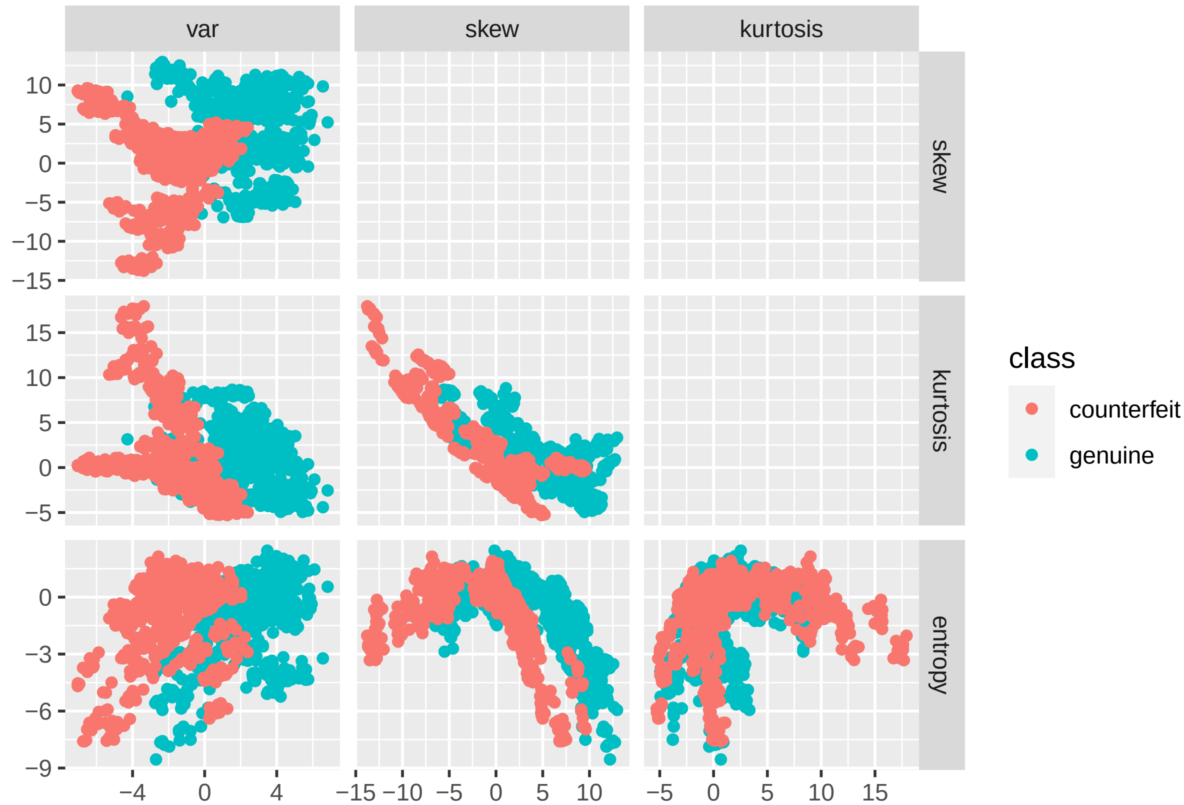
The density plots are essentially the same on both weekends and weekdays. The density in both years dips when the prices are close to 0 before going back up again. This dip is more pronounced in 2019 than in 2020. Resultingly, the main density peak is slightly higher in 2019 than in 2020.

```
head(banknotes)
```

```
## # A tibble: 6 x 5
##     var  skew kurtosis entropy class
##   <dbl> <dbl>    <dbl>   <dbl> <chr>
## 1 3.62   8.67    -2.81  -0.447 genuine
## 2 4.55   8.17    -2.46  -1.46  genuine
## 3 3.87  -2.64     1.92   0.106 genuine
## 4 3.46   9.52    -4.01  -3.59  genuine
## 5 0.329 -4.46     4.57  -0.989 genuine
## 6 4.37   9.67    -3.96  -3.16  genuine
```

Question 2a)

```
NotePairs <- banknotes %>%
             ggpairs(where(is.double), color=class, )

NotePairs
```



Both counterfeit and genuine notes follow similar distribution pattern, the main difference being a shift in these distributions depending on the class. This shift is most visible with *var* plotted along the horizontal axis. The genuine class has a visible right shift in the data points, regardless of the other attribute. This right shift is slightly visible when *skew* is plotted along the horizontal axis but not to the same extent. No visible pattern is observed between *kurtosis* and *entropy* as there is a large overlap between classes when observing these attributes.

Question 2b)

```
training = banknotes[c(1:350,1093:1372),]
test = banknotes[c(351:1092),]

view(training)
view(test)
```

Question 2c i)

```
tree1 <- rpart(class~.,data=training,method="class",
               control=rpart.control(minsplit=1, maxdepth=2, cp=0))
```

```
tree2 <- rpart(class~.,data=training,method="class",
               control=rpart.control(minsplit=1, maxdepth=5, cp=0))
```

```r
tree3 <- rpart(class~.,data=training,method="class",
               control=rpart.control(minsplit=1, maxdepth=8, cp=0))
```

Question 2c ii)

```r
# In Sample Tree 1
tree1.InPredict = predict(tree1,training,type="class")
tree1.InSample = table(Class=training$class, Prediction=tree1.InPredict)

# Out of Sample Tree 1
tree1.OutPredict = predict(tree1,test,type="class")
tree1.OutSample = table(Class=test$class, Prediction=tree1.OutPredict)

# In Sample Tree 2
tree2.InPredict = predict(tree2,training,type="class")
tree2.InSample = table(Class=training$class, Prediction=tree2.InPredict)

# Out of Sample Tree 2
tree2.OutPredict = predict(tree2,test,type="class")
tree2.OutSample = table(Class=test$class, Prediction=tree2.OutPredict)

# In Sample Tree 3
tree3.InPredict = predict(tree3,training,type="class")
tree3.InSample = table(Class=training$class, Prediction=tree3.InPredict)

# Out of Sample Tree 3
tree3.OutPredict = predict(tree3,test,type="class")
tree3.OutSample = table(Class=test$class, Prediction=tree3.OutPredict)
```

Tree 1 In-Sample Confusion Matrix:

```
##              Prediction
## Class         counterfeit genuine
##   counterfeit         258      22
##   genuine              35     315
```

Tree 1 Out-Of-Sample Confusion Matrix:

```
##              Prediction
## Class         counterfeit genuine
##   counterfeit         305      25
##   genuine              51     361
```

Tree 2 In-Sample Confusion Matrix:

```
##              Prediction
## Class         counterfeit genuine
##   counterfeit         280       0
##   genuine               2     348
```

Tree 2 Out-Of-Sample Confusion Matrix:

```
##             Prediction
## Class         counterfeit genuine
##   counterfeit         317      13
##   genuine              12     400
```

Tree 3 In-Sample Confusion Matrix:

```
##             Prediction
## Class         counterfeit genuine
##   counterfeit         280       0
##   genuine               0     350
```

Tree 3 Out-Of-Sample Confusion Matrix:

```
##             Prediction
## Class         counterfeit genuine
##   counterfeit         314      16
##   genuine              11     401
```

Question 2c iii)

```
# Tree 1 Accuracy calculations:
tree1.InSample.Acc <- (tree1.InSample[1] + tree1.InSample[4])/sum(tree1.InSample)
tree1.OutSample.Acc <- (tree1.OutSample[1] + tree1.OutSample[4])/sum(tree1.OutSample)

# Tree 2 Accuracy calculations:
tree2.InSample.Acc <- (tree2.InSample[1] + tree2.InSample[4])/sum(tree2.InSample)
tree2.OutSample.Acc <- (tree2.OutSample[1] + tree2.OutSample[4])/sum(tree2.OutSample)

# Tree 3 Accuracy calculations:
tree3.InSample.Acc <- (tree3.InSample[1] + tree3.InSample[4])/sum(tree3.InSample)
tree3.OutSample.Acc <- (tree3.OutSample[1] + tree3.OutSample[4])/sum(tree3.OutSample)

# Creates table columns
maxDepth = c('2', '5', '8')
InSample = c(tree1.InSample.Acc, tree2.InSample.Acc, tree3.InSample.Acc)
OutSample = c(tree1.OutSample.Acc, tree2.OutSample.Acc, tree3.OutSample.Acc)

# Creates table
AccuracyTable = tibble(TreeMaxDepth=maxDepth, InSampleAccuracy = InSample,
                       OutOfSampleAccuracy = OutSample)
AccuracyTable
```

```
## # A tibble: 3 x 3
##   TreeMaxDepth InSampleAccuracy OutOfSampleAccuracy
##   <chr>                   <dbl>               <dbl>
## 1 2                       0.910               0.898
## 2 5                       0.997               0.966
## 3 8                       1                   0.964
```

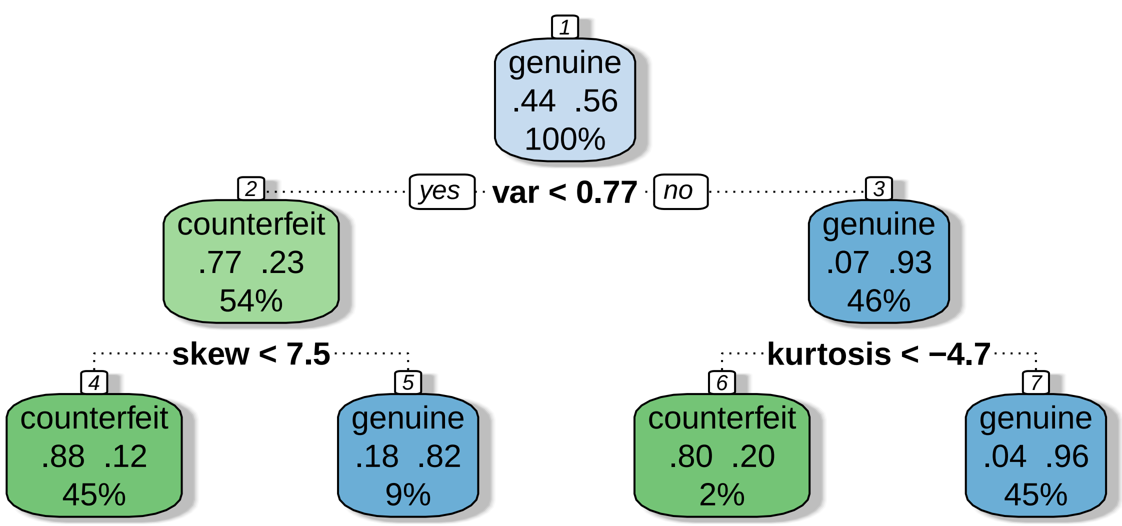
Question 2c iv)

As MaxDepth increases, the In-Sample accuracy also increases. In-Sample accuracy starts at 0.910 for a MaxDepth of 2 and increases to 0.997 for a MaxDepth of 5 before increasing to a maximum value of 1.00 for a MaxDepth of 8. This occurs because a higher MaxDepth allows a tree to have more branches and sorting conditions which separates In-Sample data points with a higher degree of accuracy. Since the In-Sample data was used to create the tree, the In-Sample accuracy can reach the maximum value of 1 because the tree is specific to that data set.

As MaxDepth increases, the Out-Of-Sample Accuracy starts at 0.900 for a MaxDepth of 2, increasing to 0.966 for a MaxDepth of 5 before decreasing slightly to 0.964 for a MaxDepth of 8. As MaxDepth becomes too high, the Out-Of-Sample accuracy reaches a plateau as the tree begins to overfit the training data. This occurs when the tree is so specific to the training set data that it is no longer able to make accurate predictions on unseen data sets. A MaxDepth of 5 seems to be optimal for this data set whereas a MaxDepth of 8 overfits the training set data.

Question 2d i)

```
# Visualise Tree with MaxDepth 2
fancyRpartPlot(tree1, caption = "Mihnea Vlad ID: 736248940")
```



Mihnea Vlad ID: 736248940

Question 2d ii)

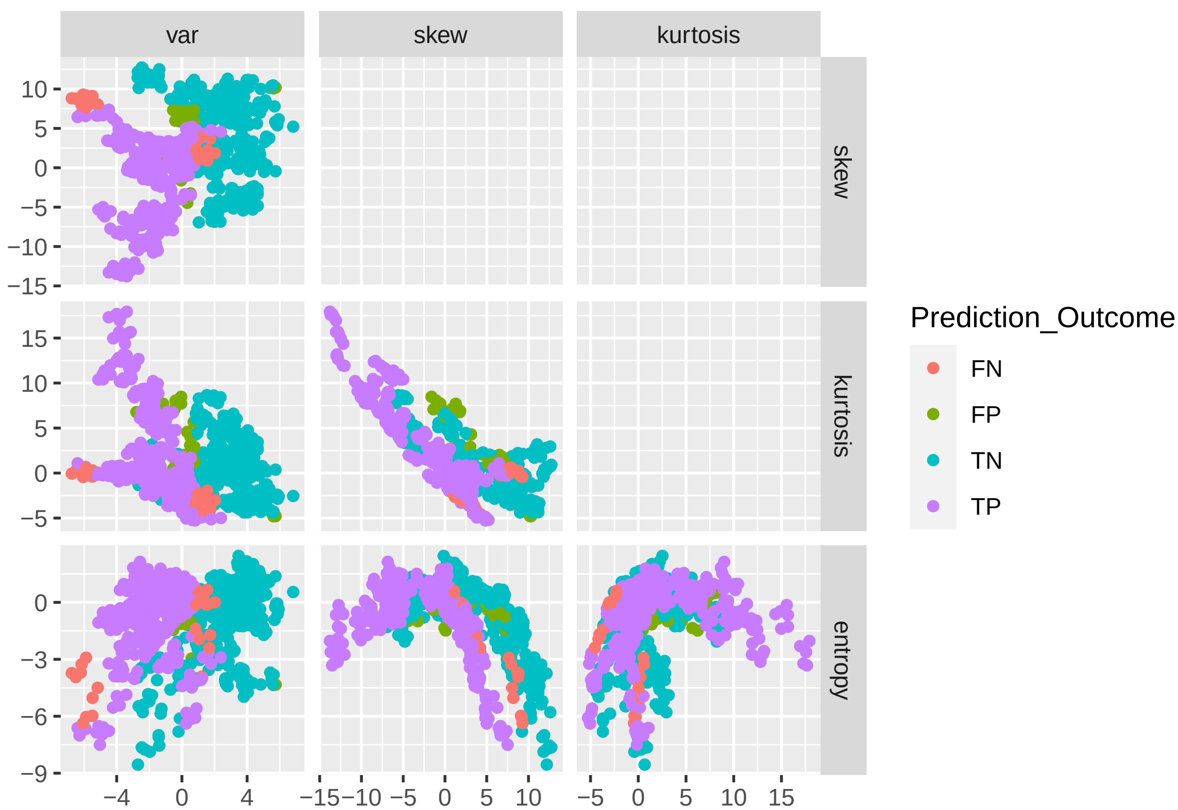
```
test <- test %>%
        mutate(Tree1_Prediction = tree1.OutPredict) %>%
```

```
        mutate(Prediction_Outcome = ifelse(class==Tree1_Prediction,
                                    ifelse(class=='genuine', 'TN', 'TP'),
                                    ifelse(class=='genuine', 'FP', 'FN')))


head(test)
```

```
## # A tibble: 6 x 7
##       var  skew kurtosis entropy class   Tree1_Prediction Prediction_Outcome
##     <dbl> <dbl>    <dbl>   <dbl> <chr>   <fct>            <chr>
## 1 -1.25  10.2      2.18   -5.60  genuine genuine          TN
## 2  0.520 -3.26     3.09   -0.985 genuine counterfeit      FP
## 3  0.329 -4.46     4.57   -0.989 genuine counterfeit      FP
## 4  0.889  5.34     2.04   -0.194 genuine genuine          TN
## 5  3.55   9.37    -4.04   -3.96  genuine genuine          TN
## 6 -0.217  8.03     1.88   -3.89  genuine genuine          TN
```

Question 2d iii)

```
PredictionPairs <- test %>%
                   ggpairs(where(is.double), color=Prediction_Outcome)

PredictionPairs
```



The misclassified points are shown the red and green dots. The tree in (d)i only uses the *var*, *skew* and *kurtosis* attributes to classify data points. Consequently, the misclassified points are seen in clusters on the plots which show the correlations between these attributes (Plots in row 1, col 1 and row 2, col 1). These clusters are located close to the values used as thresholds for those particular attributes on the tree in (d)i (i.e. $var = 0.77$, $skew = 7.5$ and $kurtosis = 4.7$). This is probably because these threshold values cannot be chosen to satisfy every single data point, so there will always be some misclassification for data points with values close to the threshold values for each attribute.

There are no visible groupings of misclassified points in the entropy plots as it wasn't used as a classification attribute on the tree in (d)i.

Question 2e i)

```r
# Create tree with MaxDepth 3 and loss matrix to avoid False Positives
tree4 <- rpart(class~.,data=training,method="class",
               control=rpart.control(minsplit=1, maxdepth=3, cp=0),
               parms=list(loss=matrix(c(0,92,8,0), nrow=2)))
```

Question 2e ii)

Tree 4 In-Sample Confusion Matrix:

```r
# In Sample Tree 4
tree4.InPredict = predict(tree4,training,type="class")
tree4.InSample = table(Class=training$class, Prediction=tree4.InPredict)
tree4.InSample
```

```
##               Prediction
## Class          counterfeit genuine
##   counterfeit         192      88
##   genuine               2     348
```

```r
# Calculate in-sample specificity
specificity = tree4.InSample[4]/(tree4.InSample[2]+tree4.InSample[4])
cat("In-Sample specificity is", specificity)
```

```
## In-Sample specificty is 0.9942857
```

Tree 4 Out-Of-Sample Confusion Matrix:

```r
# Out of Sample Tree 4
tree4.OutPredict = predict(tree4,test,type="class")
tree4.OutSample = table(Class=test$class, Prediction=tree4.OutPredict)

tree4.OutSample
```

```
##               Prediction
## Class          counterfeit genuine
##   counterfeit         206     124
##   genuine               6     406
```

Question 2f)

10 Tree forests

```r
banknotes$class <- as.factor(banknotes$class)
set.seed(100)
# 10 Tree forests:
rf1.10Trees <- randomForest(class~., banknotes, ntree=10)
rf2.10Trees <- randomForest(class~., banknotes, ntree=10)
rf3.10Trees <- randomForest(class~., banknotes, ntree=10)
rf4.10Trees <- randomForest(class~., banknotes, ntree=10)
rf5.10Trees <- randomForest(class~., banknotes, ntree=10)

rf1.10Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 10)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 1.03%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         594       7 0.011647255
## genuine               7     746 0.009296149
```

```r
rf2.10Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 10)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 1.55%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         597       7  0.01158940
## genuine              14     737  0.01864181
```

```r
rf3.10Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 10)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 2
```

```
##
##         OOB estimate of  error rate: 1.11%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         602       3 0.004958678
## genuine              12     740 0.015957447
```

```r
rf4.10Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 10)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.89%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         596       5 0.008319468
## genuine               7     746 0.009296149
```

```r
rf5.10Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 10)
##                Type of random forest: classification
##                      Number of trees: 10
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 1.25%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         598       4 0.006644518
## genuine              13     742 0.017218543
```

50 Tree forests

```r
banknotes$class <- as.factor(banknotes$class)
set.seed(100)
# 50 Tree forests:
rf1.50Trees <- randomForest(class~., banknotes, ntree=50)
rf2.50Trees <- randomForest(class~., banknotes, ntree=50)
rf3.50Trees <- randomForest(class~., banknotes, ntree=50)
rf4.50Trees <- randomForest(class~., banknotes, ntree=50)
rf5.50Trees <- randomForest(class~., banknotes, ntree=50)

rf1.50Trees
```

```
##
```

```
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 50)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.73%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         607       3 0.004918033
## genuine               7     755 0.009186352
```

```
rf2.50Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 50)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.66%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         607       3 0.004918033
## genuine               6     756 0.007874016
```

```
rf3.50Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 50)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.73%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         606       4 0.006557377
## genuine               6     756 0.007874016
```

```
rf4.50Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 50)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.8%
## Confusion matrix:
```

```
##              counterfeit genuine class.error
## counterfeit          606       4 0.006557377
## genuine                7     755 0.009186352
```

```r
rf5.50Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 50)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.51%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         610       0 0.000000000
## genuine               7     755 0.009186352
```

250 Tree forests

```r
banknotes$class <- as.factor(banknotes$class)
set.seed(100)
# 250 Tree forests:
rf1.250Trees <- randomForest(class~., banknotes, ntree=250)
rf2.250Trees <- randomForest(class~., banknotes, ntree=250)
rf3.250Trees <- randomForest(class~., banknotes, ntree=250)
rf4.250Trees <- randomForest(class~., banknotes, ntree=250)
rf5.250Trees <- randomForest(class~., banknotes, ntree=250)

rf1.250Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 250)
##                Type of random forest: classification
##                      Number of trees: 250
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.58%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         608       2 0.003278689
## genuine               6     756 0.007874016
```

```r
rf2.250Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 250)
##                Type of random forest: classification
```

```
##                      Number of trees: 250
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.66%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         607       3 0.004918033
## genuine               6     756 0.007874016
```

```r
rf3.250Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 250)
##                Type of random forest: classification
##                      Number of trees: 250
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.66%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         607       3 0.004918033
## genuine               6     756 0.007874016
```

```r
rf4.250Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 250)
##                Type of random forest: classification
##                      Number of trees: 250
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.66%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         607       3 0.004918033
## genuine               6     756 0.007874016
```

```r
rf5.250Trees
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = banknotes, ntree = 250)
##                Type of random forest: classification
##                      Number of trees: 250
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.58%
## Confusion matrix:
##             counterfeit genuine class.error
## counterfeit         608       2 0.003278689
## genuine               6     756 0.007874016
```

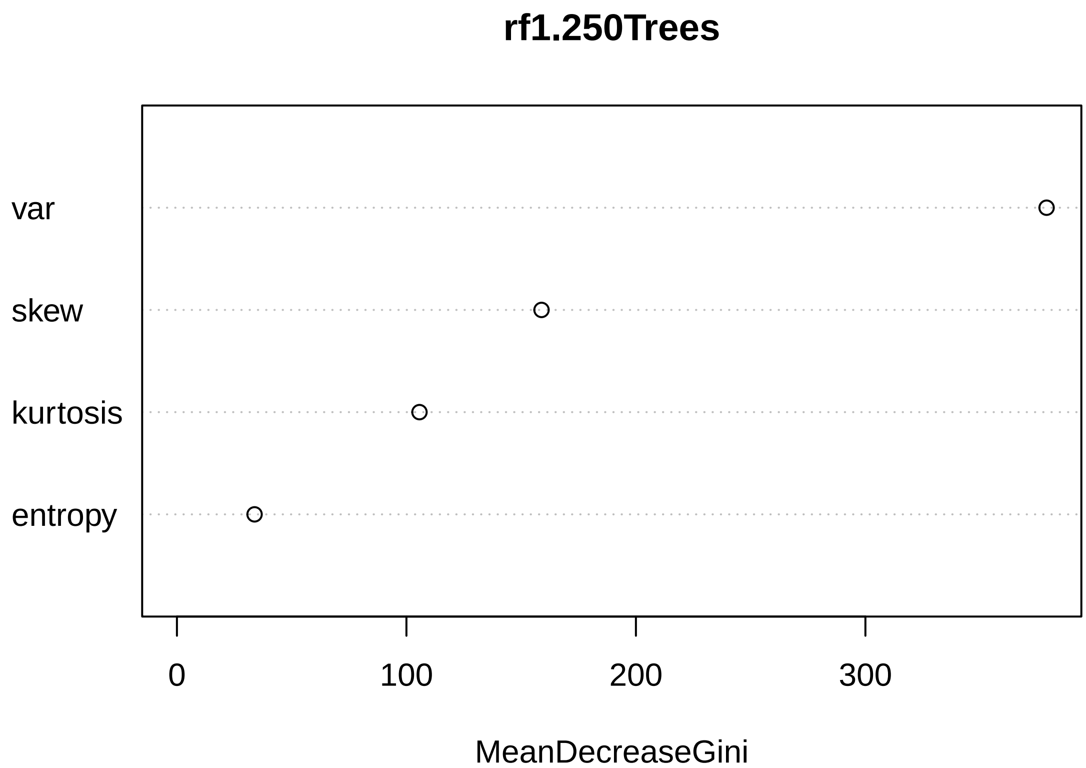
```r
# Creates table columns
nTrees = c(10, 50, 250)
rf1_OOB = c(1.03, 0.73, 0.58)
rf2_OOB = c(1.55, 0.66, 0.66)
rf3_OOB = c(1.11, 0.73, 0.66)
rf4_OOB = c(0.89, 0.8, 0.66)
rf5_OOB = c(1.25, 0.51, 0.58)

# Creates table
OOBTable <- tibble(nTrees, rf1_OOB, rf2_OOB, rf3_OOB, rf4_OOB, rf5_OOB)

OOBSummary <- OOBTable %>%
              pivot_longer(contains('f'), names_to = 'RF', values_to = 'OOB') %>%
              group_by(nTrees) %>%
              summarise(Mean_OOB = mean(OOB))


OOBTable <- OOBTable %>%
            mutate(dplyr::select(OOBSummary,2))

OOBTable
```

```
## # A tibble: 3 x 7
##   nTrees rf1_OOB rf2_OOB rf3_OOB rf4_OOB rf5_OOB Mean_OOB
##    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>    <dbl>
## 1     10    1.03    1.55    1.11    0.89    1.25    1.17
## 2     50    0.73    0.66    0.73    0.8     0.51    0.686
## 3    250    0.58    0.66    0.66    0.66    0.58    0.628
```

As the number of trees in a random forest increases, the mean OOB decreases. This makes sense as larger forests will use a larger number of trees to predict an outcome, reducing the likelihood of misclassification which leads to a decrease in mean OOB.

Question 2g)

```r
ImPlot = varImpPlot(rf1.250Trees)
```




```
ImPlot
```

```
##          MeanDecreaseGini
## var             378.93483
## skew            158.85615
## kurtosis        105.68100
## entropy          33.82188
```

var is, by far, the most useful attribute to detect forgeries because it has the highest mean decrease in Gini score of 372.5 (When a split is made using this attribute, the mean decrease in the Gini score is 372.5). The second most useful attribute is *skew* with a mean decrease in Gini score of 162.8, the third most useful attribute is *kurtosis* with a mean decrease in Gini score of 107.5 and the least useful attribute is *entropy* with a mean decrease in Gini score of 34.3. Based on this plot, a relatively accurate random forest could be generated without including the entropy attribute as it is not very useful in detecting forgeries.

Question 2h)

```
# Converts all attributes to factors
banknotesDis$var <- as.factor(banknotesDis$var)
banknotesDis$skew <- as.factor(banknotesDis$skew)
banknotesDis$kurtosis <- as.factor(banknotesDis$kurtosis)
banknotesDis$entropy <- as.factor(banknotesDis$entropy)
banknotesDis$class <- as.factor(banknotesDis$class)
```

```r
# Creates test set and training set
trainingD = banknotesDis[c(1:350,1093:1372),]
testD = banknotesDis[c(351:1092),]

# Creates Naive Bayes model
banknotesDis.nb = NaiveBayes(class~.,data=trainingD)
```

Question 2i)

```r
# In-sample predictions
InSample <- suppressWarnings(predict(banknotesDis.nb, trainingD)$class)
InSample.conf <- table(class=trainingD$class,prediction=InSample)

InSample.conf
```

```
##                prediction
## class           counterfeit genuine
##   counterfeit           252      28
##   genuine                17     333
```

```r
InSample.Acc <- (InSample.conf[1] + InSample.conf[4])/sum(InSample.conf)

# Out-sample predictions
OutSample <- suppressWarnings(predict(banknotesDis.nb, testD)$class)
OutSample.conf <- table(class=testD$class,prediction=OutSample)

OutSample.conf
```

```
##                prediction
## class           counterfeit genuine
##   counterfeit           294      36
##   genuine                24     388
```

```r
OutSample.Acc <- (OutSample.conf[1] + OutSample.conf[4])/sum(OutSample.conf)

noquote("")
```

```
## [1]
```

```r
noquote("In Sample Accuracy:")
```

```
## [1] In Sample Accuracy:
```

```r
InSample.Acc
```

```
## [1] 0.9285714
```

```r
noquote("")
```

```
## [1]
```

```r
noquote("Out of Sample Accuracy:")
```

```
## [1] Out of Sample Accuracy:
```

```r
OutSample.Acc
```

```
## [1] 0.9191375
```

Question 2j)

```r
noquote('Naive-Bayes:')
```

```
## [1] Naive-Bayes:
```

```r
OutSample.conf
```

```
##              prediction
## class         counterfeit genuine
##   counterfeit         294      36
##   genuine              24     388
```

```r
noquote('Tree - depth 5:')
```

```
## [1] Tree - depth 5:
```

```r
tree2.OutSample
```

```
##              Prediction
## Class         counterfeit genuine
##   counterfeit         317      13
##   genuine              12     400
```

The Naive Bayes model has an out sample accuracy of 0.92 whereas the classification tree with depth 5 has an out sample accuracy of 0.97 which is slightly higher than the Naive Bayes model.

Question 2k)

Naive Bayes assumes conditional independence among the predictor variables. This is unlikely to be true in this scenario because relationships between the predictor variables exist as seen on the pairs plot in q2 d iii. This probably accounts for the accuracy differences.