MONASH
University

FIT 3152 Assignment 2

Tutorial group: Monday, 12pm – 2pm

Monash ID: 29634431

Name: Lai Wei Jian

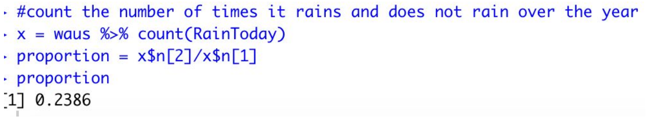
## Introduction

The WAUS2020 dataset contains 25 columns data which consists of Day, Month, Year, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Temp9am, Temp3pm, Cloud9am, Cloud3pm, RainToday and RainTomorrow. Before doing any pre-processing to the data, it has 100,000 rows of data with 25 variables mentioned earlier. Then, I have created my individual data using my seed ID to generate 2000 rows of data. The type of the variables Day, Month, Year, Location, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Cloud9am, Cloud3pm are Integer type, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, Pressure9am, Pressure3pm, Temp9am, Temp3pm are Num type and the remaining variables are Chr type which will then be converted to Factor type. **The library used are listed below:**

**library(tree)**
**library(e1071)**
**library(ROCR)**
**library(randomForest)**
**library(adabag)**
**library(rpart)**
**library(pastecs)**
**library(caret)**
**library(dplyr)**
**library(corrplot)**
**library(neuralnet)**
**library(car)**

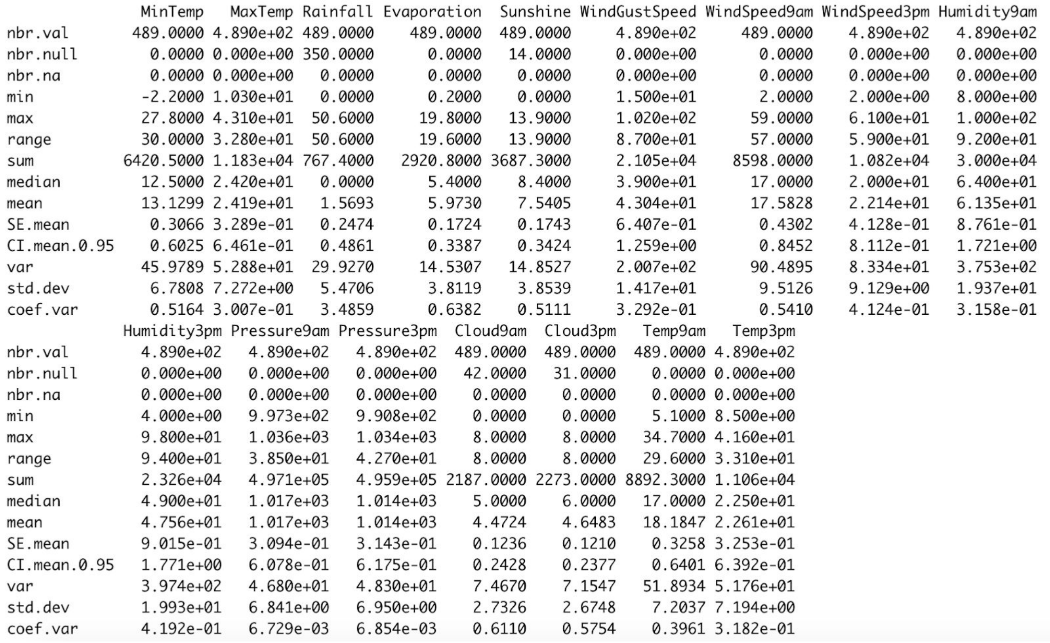
## Question 1

Before doing any pre-processing to the data, I first calculated the proportion of rainy days to fine days by using the library dplyr. As it can be seen below, the proportion value is 0.2386 which means that the odds of fine days happening are higher than rainy days giving such a low proportion value.



```
> #count the number of times it rains and does not rain over the year
> x = waus %>% count(RainToday)
> proportion = x$n[2]/x$n[1]
> proportion
[1] 0.2386
```

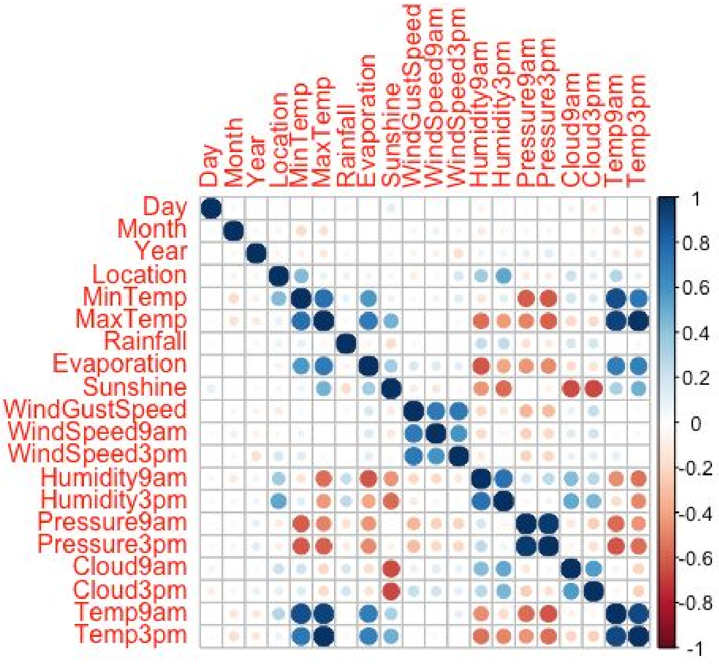
**Descriptions of the predictor variables for all real-valued attributes using stat.desc() function:**



```
               MinTemp   MaxTemp Rainfall Evaporation  Sunshine WindGustSpeed WindSpeed9am WindSpeed3pm Humidity9am
nbr.val       489.0000 4.890e+02 489.0000    489.0000  489.0000     4.890e+02     489.0000    4.890e+02   4.890e+02
nbr.null        0.0000 0.000e+00 350.0000      0.0000   14.0000     0.000e+00       0.0000    0.000e+00   0.000e+00
nbr.na          0.0000 0.000e+00   0.0000      0.0000    0.0000     0.000e+00       0.0000    0.000e+00   0.000e+00
min            -2.2000 1.030e+01   0.0000      0.2000    0.0000     1.500e+01       2.0000    2.000e+00   8.000e+00
max            27.8000 4.310e+01  50.6000     19.8000   13.9000     1.020e+02      59.0000    6.100e+01   1.000e+02
range          30.0000 3.280e+01  50.6000     19.6000   13.9000     8.700e+01      57.0000    5.900e+01   9.200e+01
sum          6420.5000 1.183e+04 767.4000   2920.8000 3687.3000     2.105e+04    8598.0000    1.082e+04   3.000e+04
median         12.5000 2.420e+01   0.0000      5.4000    8.4000     3.900e+01      17.0000    2.000e+01   6.400e+01
mean           13.1299 2.419e+01   1.5693      5.9730    7.5405     4.304e+01      17.5828    2.214e+01   6.135e+01
SE.mean         0.3066 3.289e-01   0.2474      0.1724    0.1743     6.407e-01       0.4302    4.128e-01   8.761e-01
CI.mean.0.95    0.6025 6.461e-01   0.4861      0.3387    0.3424     1.259e+00       0.8452    8.112e-01   1.721e+00
var            45.9789 5.288e+01  29.9270     14.5307   14.8527     2.007e+02      90.4895    8.334e+01   3.753e+02
std.dev         6.7808 7.272e+00   5.4706      3.8119    3.8539     1.417e+01       9.5126    9.129e+00   1.937e+01
coef.var        0.5164 3.007e-01   3.4859      0.6382    0.5111     3.292e-01       0.5410    4.124e-01   3.158e-01
             Humidity3pm Pressure9am Pressure3pm  Cloud9am  Cloud3pm   Temp9am   Temp3pm
nbr.val        4.890e+02   4.890e+02   4.890e+02  489.0000  489.0000  489.0000 4.890e+02
nbr.null       0.000e+00   0.000e+00   0.000e+00   42.0000   31.0000    0.0000 0.000e+00
nbr.na         0.000e+00   0.000e+00   0.000e+00    0.0000    0.0000    0.0000 0.000e+00
min            4.000e+00   9.973e+02   9.908e+02    0.0000    0.0000    5.1000 8.500e+00
max            9.800e+01   1.036e+03   1.034e+03    8.0000    8.0000   34.7000 4.160e+01
range          9.400e+01   3.850e+01   4.270e+01    8.0000    8.0000   29.6000 3.310e+01
sum            2.326e+04   4.971e+05   4.959e+05 2187.0000 2273.0000 8892.3000 1.106e+04
median         4.900e+01   1.017e+03   1.014e+03    5.0000    6.0000   17.0000 2.250e+01
mean           4.756e+01   1.017e+03   1.014e+03    4.4724    4.6483   18.1847 2.261e+01
SE.mean        9.015e-01   3.094e-01   3.143e-01    0.1236    0.1210    0.3258 3.253e-01
CI.mean.0.95   1.771e+00   6.078e-01   6.175e-01    0.2428    0.2377    0.6401 6.392e-01
var            3.974e+02   4.680e+01   4.830e+01    7.4670    7.1547   51.8934 5.176e+01
std.dev        1.993e+01   6.841e+00   6.950e+00    2.7326    2.6748    7.2037 7.194e+00
coef.var       4.192e-01   6.729e-03   6.854e-03    0.6110    0.5754    0.3961 3.182e-01
```

As I have removed all the non real-valued attributes before getting the description for each predictor variable, the table above only shows the descriptions of the predictor variables for all the real-valued attributes. This shows a description of each variable in the dataset such as the min, max, mean, standard deviation and etc. to provide more information about the dataset.

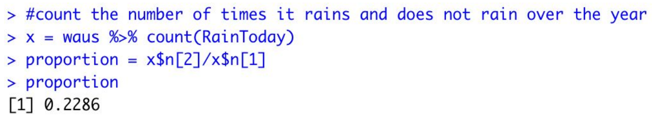
Heatmap showing the correlations of all the variables in the dataset



As it can be seen from the heatmap graph above,the attribute Day, Month and Year do ont have much correlation with other variables inside the dataset. Therefore, it will be omitted from the dataset because including the Day, Month and Year will only increase the workload which leads to an increase in the chance of errors. This is because these attributes are not used when building a model which means that it's not important to the classification model.

## Question 2

In order to tidy up and pre-process the data, I have removed the attributes Day, Month and Year from the dataset to exclude those irrelevant variables to improve them and also remove all the NA values in the dataset by using na.omit() function. The dataset will be left with 489 rows of data to be analysed. The proportion of the rainy days to fine days are again calculated after removing the NA values and the unused attributes, the value decreased to 0.2286 which means that the number of rainy days in the dataset decreases after tidying up the data.



```
> #count the number of times it rains and does not rain over the year
> x = waus %>% count(RainToday)
> proportion = x$n[2]/x$n[1]
> proportion
[1] 0.2286
```

## Question 3

By using the pre-processed dataset, the data will then be divided into a 70% training set and 30% test set where the training set can be used to build up a model and the test set is used to validate the model built. Therefore, the training dataset will later be used to build the classification models and the testing dataset will be used to determine how accurate the classification model is.

## Question 4

**Implement a decision tree:**

waus_decisiontree = tree(RainTomorrow ~., data = waus.train)

**Implement a naive bayes:**

waus_naivebayes = naiveBayes(RainTomorrow~., data = waus.train)

**Implement a bagging:**

waus_bagging = bagging(RainTomorrow ~. , data = waus.train, mfinal=5)

**Implement a boosting:**

waus_boosting = boosting(RainTomorrow ~. , data = waus.train, mfinal=10)

**Implement a random forest:**
waus_randomforest = randomForest(RainTomorrow ~. , data = waus.train, na.action = na.exclude)

All the classification models decision tree, naive bayes, bagging, boosting and random forest are implemented by using the tree(), naiveBayes(), bagging(), boosting(), randomForest() function respectively with inputting the RainTomorrow variable as the outcome value that we want to predict with all the other predictor variables in the dataset using the training dataset.

## Question 5

By using the testing dataset and the created predicted classification models to predict the accuracy for each classification model. It will classify it into 'will rain tomorrow' or 'will not rain tomorrow'. In order to get the accuracy for each classification model,  I first predict the classification model with the testing dataset for each classification model with the type = 'class' to get the probability of Raining Tomorrow that return the results of either 'Yes' or 'No' for every row of data in the predicted model. Then, I will use this predicted output result and compare it with the testing dataset to create a confusion matrix. I also used the confusionMatrix() function that input the predicted classification model and the testing dataset with the RainTomorrow variables as the parameter for the function which can determine the Accuracy of the classification model. This is because the confusionMatrix() function will create a confusion matrix table and calculate the accuracy of the classification model.

### Confusion Matrix for Decision Tree

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **105 (TP)** | **20 (FN)** |
| | **Class = Yes** | **12 (FP)** | **10 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$

Accuracy for Decision Tree model = $\frac{105 + 10}{105 + 12 + 10 + 20}$

= 0.7823 $\approx$78.23%

This can determine the performance of the decision tree model. In other words, this gives the accuracy for the decision tree model to be 78.23% correct which means that the prediction it makes will be 78.23% correct.

**Confusion Matrix for Naive Bayes**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **95 (TP)** | **9 (FN)** |
| | **Class = Yes** | **22 (FP)** | **21 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$

Accuracy for Naive Bayes model = $\frac{95 + 21}{95 + 22 + 21 + 9}$

= 0..7755 $\approx$77.55%

This can determine the performance of the Naive Bayes model. In other words, this gives the accuracy for the Naive Bayes model to be 77.55% correct which means that the prediction it makes will be 77.55% correct.

**Confusion Matrix for Bagging**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **112 (TP)** | **23 (FN)** |
| | **Class = Yes** | **5 (FP)** | **7 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive +\ True\ Negative + False\ Negative}$

Accuracy for Bagging model = $\frac{112 + 7}{112 + 5 + 7 + 23}$

= 0.8095 $\approx$80.95%

This can determine the performance of the Bagging model. In other words, this gives the accuracy for the Bagging model to be 80.95% correct which means that the prediction it makes will be 80.95% correct.

**Confusion Matrix for Boosting**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **110 (TP)** | **19 (FN)** |
| | **Class = Yes** | **7 (FP)** | **11 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $$\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$$

Accuracy for Boosting model = $$\frac{110 + 11}{110 + 7 + 11 + 19}$$

= 0.8231 $\approx$82.31%

This can determine the performance of the Boosting model. In other words, this gives the accuracy for the Boosting model to be 82.31% correct which means that the prediction it makes will be 82.31% correct.

**Confusion Matrix for Random Forest**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **113 (TP)** | **20 (FN)** |
| | **Class = Yes** | **4 (FP)** | **9 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $$\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$$

Accuracy for Random Forest model = $$\frac{113 + 9}{113 + 4 + 9 + 20}$$

= 0.8367 $\approx$83.67%

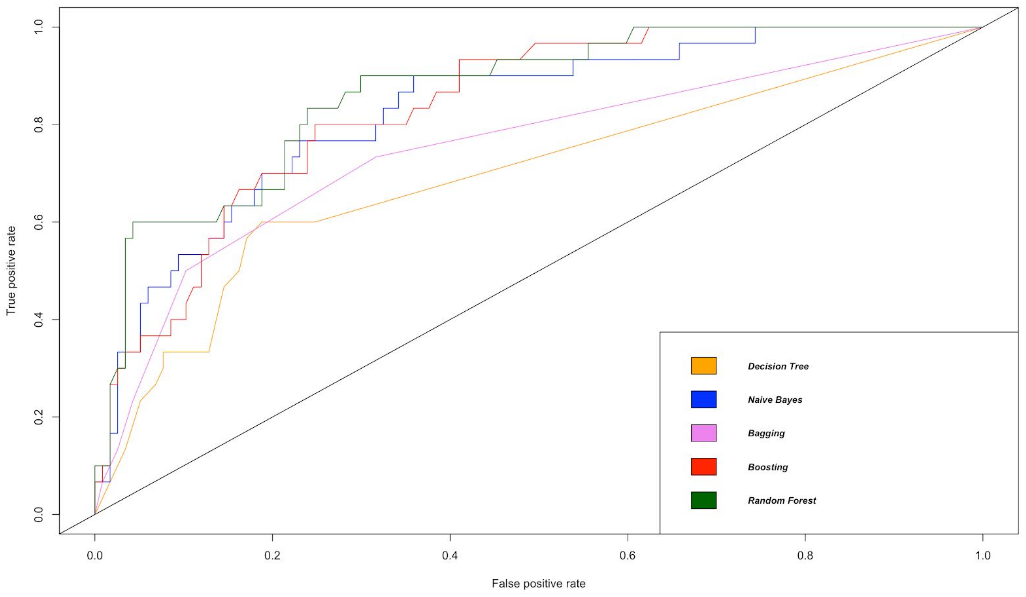
This can determine the performance of the Random Forest model. In other words, this gives the accuracy for the Random Forest model to be 83.67% correct which means that the prediction it makes will be 83.67% correct.

**In conclusion, by looking at the accuracy value for all the classification models mentioned above. We can conclude that Random Forest has the highest accuracy amongst the other classification models as it has the accuracy value of 83.67%.**

## Question 6

In this question, we are using the test data to predict the confidence of ‘will rain tomorrow’ and construct an ROC curve for each classifier which gives a more comprehensive comparison of several classifiers. It will construct the ROC curve by predicting each of the classification models created with the testing dataset with the type = ‘raw’. It will then get the predicted result of ‘it will rainTomorrow’ and compare it with the RainTomorrow variable in the testing dataset. After getting the result, it will then use a performance() function to input the data and output it as a coordinate value with (TPR,FPR) where TPR is the true positive rate in the x-axis and FPR is false positive rate in y-axis. This will then be plotted out into a curve to visualise the goodness of the classifier.

### ROC Curve for weather prediction for each classifier



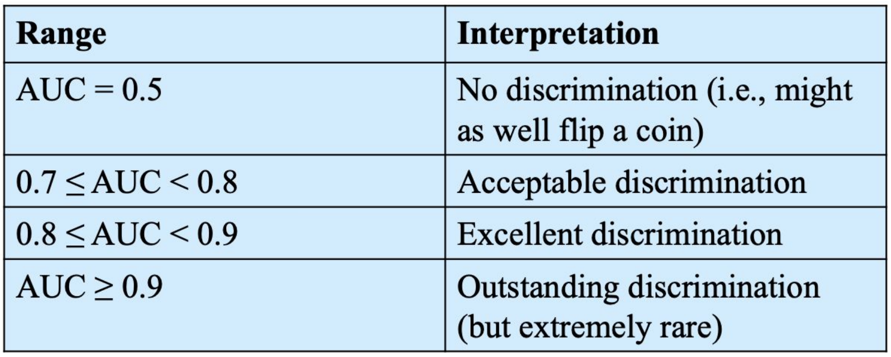
The closer the classifier is towards the True positive rate means that it will be a better model. So from the graph above, we can see that the dark green line - random forest is the closest to the true positive rate followed by red line - boosting, blue line - naive bayes, pink line - bagging and lastly orange line- decision tree. We can say that random forest is a great model by looking at the graph above, boosting and naive bayes will be a good model and bagging and decision trees will be considered as a bad model. However, no model will consistently outperform the other. For example, the blue line - naive bayes performs better for small FPR and the red line - boosting performs better for large FPR compared to the classifier naive bayes.

The AUC for each classifier is calculated and displayed into a table below:

| Classifier | Area Under Curve (AUC) |
|---|---|
| Decision Tree | 0.6964 |
| Naive Bayes | 0.8299 |
| Bagging | 0.7514 |
| Boosting | 0.8370 |
| Random Forest | 0.8651 |

From the table above, we can see that Random Forest has the highest value in AUC which is 0.8651 followed by Boosting, Naive Bayes, Bagging and Decision Tree.

Guidelines to interpret AUC:



| Range | Interpretation |
|---|---|
| AUC = 0.5 | No discrimination (i.e., might as well flip a coin) |
| 0.7 ≤ AUC < 0.8 | Acceptable discrimination |
| 0.8 ≤ AUC < 0.9 | Excellent discrimination |
| AUC ≥ 0.9 | Outstanding discrimination (but extremely rare) |

From the table above, we can say that Random Forest, Boosting and Naive Bayes are interpreted as excellent discrimination as the AUC falls within the range between 0.80 to 0.90. Bagging will be acceptable discrimination as the AUC falls within the range between 0.70 to 0.80. Although Decision Tree has AUC value of 0.6964 which is close to 0.70, the AUC is still below 0.70 which will still be classified as having no discrimination.

**Question 7**

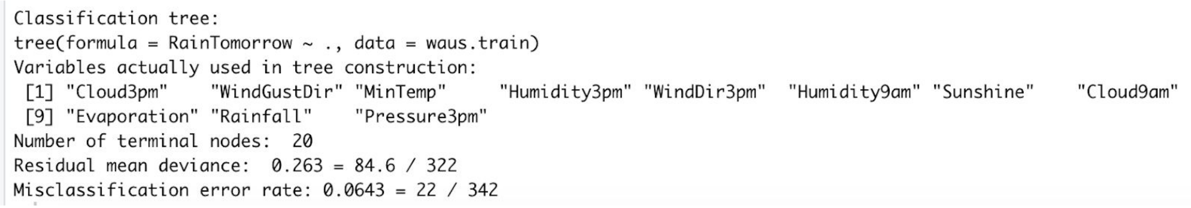
The table below shows the Accuracy of the classification model by using confusion matrix and the AUC accuracy for each classifier:

| Classifier | Confusion Matrix Accuracy | AUC |
|---|---|---|
| Decision Tree | 0.7891 | 0.6964 |
| Naive Bayes | 0.7891 | 0.8299 |
| Bagging | 0.8095 | 0.7514 |
| Boosting | 0.8231 | 0.8370 |
| Random Forest | 0.8367 | 0.8651 |

Based on the table above, we can see that Random Forest  scored really high in both the Confusion Matrix Accuracy and AUC which has the value of 0.8367 in Confusion Matrix Accuracy and 0.8651 in AUC. We can see that Naive Bayes, Boosting and Random Forest performs better and have higher value in AUC compared to the confusion matrix accuracy. Decision Tree and Bagging are the opposite of them, having a higher confusion matrix accuracy value than AUC. The Accuracy calculated by confusion matrix measures, for a given threshold, the percentage of points are classified correctly. AUC measures the performance of the classifier in which measuring the likelihood of two random points - one from the positive and one from the negative class which will then rank the point from the positive class higher than the negative class. The closer the AUC or Confusion Matrix Accuracy towards 1, the better the classifier. Hence, we can conclude that the single best classifier will be Random Forest as it has the highest AUC and confusion matrix accuracy value amongst the other classifiers. Decision tree will be the worst classifier as the value for both the accuracy are low and AUC is below the acceptable discrimination range.

## **<u>Question 8</u>**

### Decision Tree



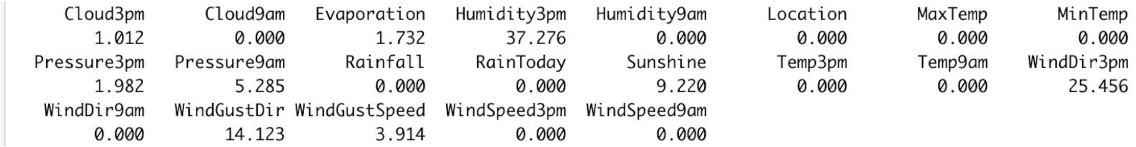
```
Classification tree:
tree(formula = RainTomorrow ~ ., data = waus.train)
Variables actually used in tree construction:
 [1] "Cloud3pm"    "WindGustDir" "MinTemp"     "Humidity3pm" "WindDir3pm"  "Humidity9am" "Sunshine"    "Cloud9am"
 [9] "Evaporation" "Rainfall"    "Pressure3pm"
Number of terminal nodes:  20
Residual mean deviance:  0.263 = 84.6 / 322
Misclassification error rate: 0.0643 = 22 / 342
```

From the screenshot above, we can see that the most important variables are Cloud3pm, WindGustDir, MinTemp, Humidity3pm, WindDir3pm, Humidity9am, Sunshine, Cloud9am, Evaporation, Rainfall and Pressure3pm. This is because all these variables are used in constructing the tree. The variables that can be omitted out from the data are Location, MaxTemp, WindDir9am, WindGustSpeed WindSpeed9am, WindSpeed3pm, Temp9am, Temp3pm, Pressure9am and RainToday as it is not used to construct the tree.

### Naive Bayes

After getting a summary of naive bayes classification models, there are no most important variables found from the analysis. This is due to the main limitation of Naive Bayes is the assumption of independent predictor features. Naive Bayes will assume all the attributes are mutually independent. The categorical variable in the testing dataset will be assigned with zero probability and will be unable to make a prediction as well because the categorical variable is not observed in the training dataset.

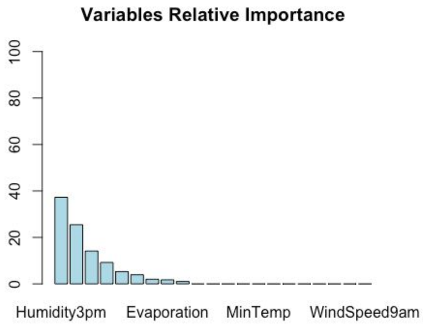
### Bagging



```
      Cloud3pm       Cloud9am    Evaporation    Humidity3pm    Humidity9am       Location        MaxTemp        MinTemp
         1.012          0.000          1.732         37.276          0.000          0.000          0.000          0.000
   Pressure3pm    Pressure9am       Rainfall      RainToday       Sunshine        Temp3pm        Temp9am     WindDir3pm
         1.982          5.285          0.000          0.000          9.220          0.000          0.000         25.456
    WindDir9am    WindGustDir  WindGustSpeed   WindSpeed3pm   WindSpeed9am
         0.000         14.123          3.914          0.000          0.000
```

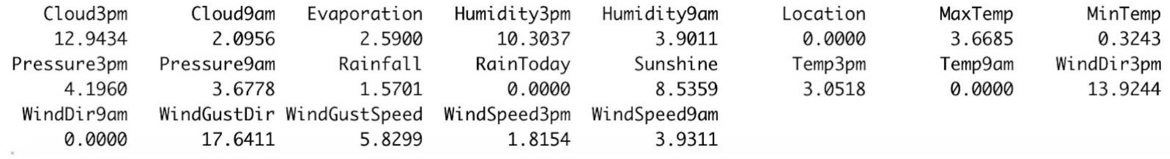
From the screenshot above, we can see that the most important variables for bagging are Humidity3pm, WindDir3pm, WindGustDir, Sunshine, Pressure9am, WindGustSpeed, Evaporation, Pressure3pm and Cloud3pm. Humidity3pm has the highest value of 37.276 compared to other variables which will be the most important variable in the bagging classification model. The variables that can be omitted from the data are Location, MinTemp, MaxTemp, Rainfall, Sunshine, Temp9am, WindDir9am, WindSpeed3pm, Cloud9am, Humidity9am and Pressure3pm. The reason is because they have a value of 0 which means that they are not important and have not been used by the model to make predictions.

The graph below shows the most important variables in descending orders:



We can clearly see from the graph that Humidity3pm has the highest value which means that it is the most important variable in the dataset when bagging classifiers are used.

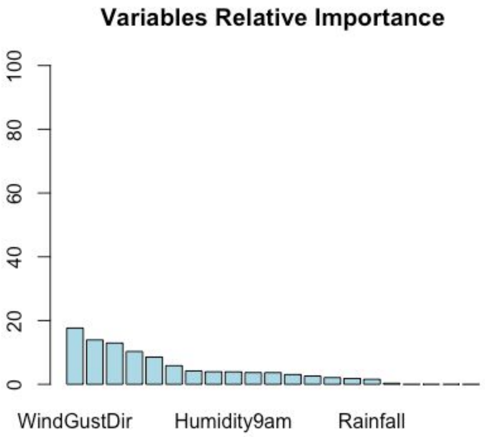
### Boosting



```
     Cloud3pm      Cloud9am   Evaporation   Humidity3pm   Humidity9am      Location       MaxTemp       MinTemp 
      12.9434        2.0956        2.5900       10.3037        3.9011        0.0000        3.6685        0.3243 
  Pressure3pm   Pressure9am      Rainfall     RainToday      Sunshine      Temp3pm       Temp9am    WindDir3pm 
       4.1960        3.6778        1.5701        0.0000        8.5359        3.0518        0.0000       13.9244 
   WindDir9am   WindGustDir WindGustSpeed  WindSpeed3pm  WindSpeed9am 
       0.0000       17.6411        5.8299        1.8154        3.9311 
```

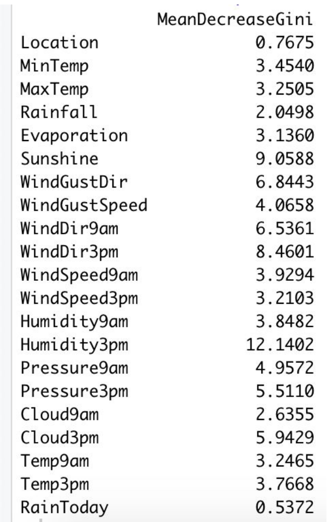
From the screenshot above, we can see that the most important variables for bagging are WindGustDir, WindDir3pm, Cloud3pm, Cloud9am, Humidity3pm, Sunshine, WindGustSpeed, Pressure3pm, WindGustDir, Sunshine, Pressure9am, Evaporation, Rainfall, Humidity9am, Temp3pm, MinTemp and MaxTemp. WindGustDir has the highest value of 17.6411 compared to other variables which will be the most important variable in the boosting classification model. The variables that can be omitted from the data are Location, RainToday, Temp9am and WindDir9am. The reason is because they have a value of 0 which means that they are not important and have not been used by the model to make predictions.

The graph below shows the most important variables in descending orders:



We can clearly see from the graph that WindGustDir has the highest value which means that it is the most important variable in the dataset when Boosting classifier is used.

## Random Forest



```
              MeanDecreaseGini
Location                0.7675
MinTemp                 3.4540
MaxTemp                 3.2505
Rainfall                2.0498
Evaporation             3.1360
Sunshine                9.0588
WindGustDir             6.8443
WindGustSpeed           4.0658
WindDir9am              6.5361
WindDir3pm              8.4601
WindSpeed9am            3.9294
WindSpeed3pm            3.2103
Humidity9am             3.8482
Humidity3pm            12.1402
Pressure9am             4.9572
Pressure3pm             5.5110
Cloud9am                2.6355
Cloud3pm                5.9429
Temp9am                 3.2465
Temp3pm                 3.7668
RainToday               0.5372
```

From the screenshot above, we can see that the most important variables for bagging are Humidity3pm, WindGustDir, WindDir3pm, Cloud3pm, Cloud9am, Sunshine, WindGustSpeed, Pressure3pm, WindGustDir, Sunshine, Pressure9am, Evaporation, Rainfall, Humidity9am, Temp9am, WindDir9am, Temp3pm, MinTemp and MaxTemp. Humidity3pm has the highest value of 12.1402 compared to other variables which will be the most important variable in the boosting classification model. The variables that can be omitted from the data are Location and RainToday. The reason is because they have a value of close to 0 which means that they are not important and have not been used by the model to make predictions.

Variables to omit: Location, WindDir9am, RainToday

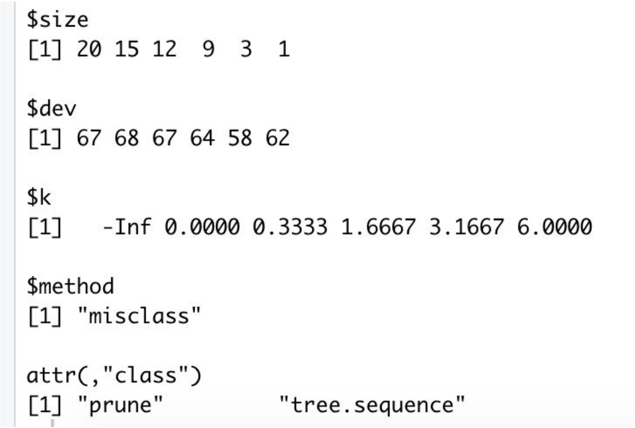
In conclusion, we can see that RainToday and Location variables have really little performance on most of the classifiers such as Decision Tree, Bagging, Boosting and Random Forest. The variable WindDir9am has a low value in both Bagging and Boosting classifiers and it's not included from the Decision Tree when constructing a tree. Hence, we can omit the variable WindDir9am, RainToday and Location out of the dataset. RainToday and Location are not a part of our analysis. Including them will only increase the workload, which will increase the chances of error and reduce the accuracy value in making predictions. We can say that Humidity3pm is the most important variable in the dataset as it has the highest value in Bagging and Random forest classifiers and it has high values in the other classifiers. It is also an important variable in every classifier mentioned above. Therefore, we can conclude that Humidity3pm is the most important variable in predicting whether it will rain tomorrow. However, all the other most important variables mentioned earlier in each classifier are still important and play a role in helping to predict whether it will rain tomorrow.

**<u>Question 9</u>**

Our dataset has dropped three attributes, Location, RainToday and WindDir9am before moving on to create the best tree classifier. This is to remove those attributes that are not helpful in predicting whether it will rain tomorrow to avoid overfitting. I will be using cross validation for  and pruning the trees to create the best tree-based classifier. By doing that, I will perform cross validation to all the tree-based classifiers and also prune the decision tree and see which tree-based model gives the highest accuracy value. The most important factor in my decision is the accuracy of the classifiers.

### Decision Tree

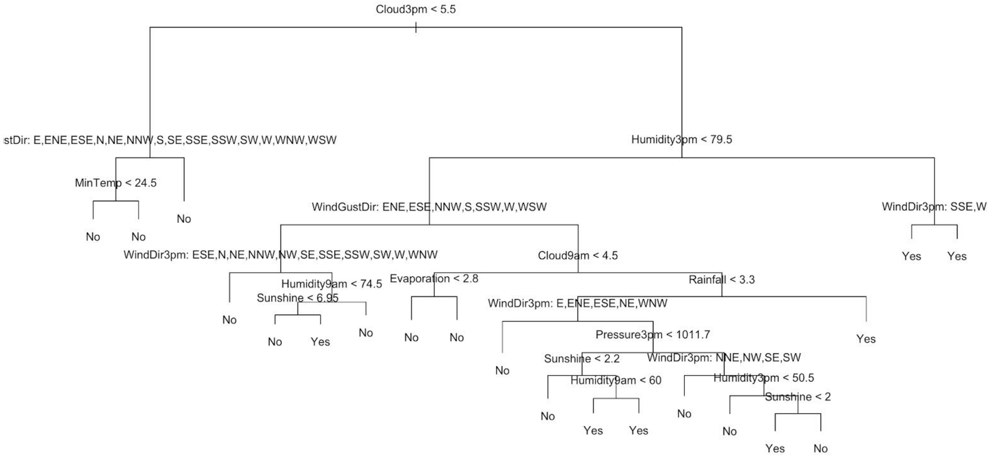
First, I have used a cross validation function at different tree sizes with inputting the original tree and the prune.misclass as the performance measure as the input parameters.



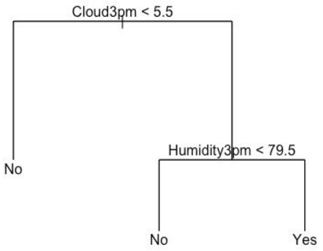
```
$size
[1] 20 15 12  9  3  1

$dev
[1] 67 68 67 64 58 62

$k
[1]   -Inf 0.0000 0.3333 1.6667 3.1667 6.0000

$method
[1] "misclass"

attr(,"class")
[1] "prune"         "tree.sequence"
```

The result above shows that using the tree size of 3 will be the best to fit the decision tree as it has the lowest dev value. Having high dev value which counts the misclass at each size will cause underfitting and tree size of 1 will cause overfitting. Therefore, the tree size of 3 will be the best.

Decision Tree before using cross validation and pruning:



Decision Tree after using cross validation and pruning:



As it can be seen from the two trees above, the decision tree after performing cross validation and pruning will be left with the tree size of 3.

Table below showing the Accuracy and AUC value before after performing cross validation and pruning:

| | Confusion Matrix Accuracy | AUC |
|---|---|---|
| Decision Tree | 0.7891 | 0.6964 |
| Cross Validation and Pruned tree | 0.8095 | 0.7534 |

We can see that the Accuracy is 2% higher than before performing cross validation and pruning. It is close to 6% increase in the AUC value which means that decision tree falls under the acceptable discrimination now as the AUC is within the range of

0.70 to 0.80. This proves that by performing the cross validation and pruning the tree will improve the accuracy of the classification model. However, the misclassification error rate increases from 0.0643 to 0.137 Let’s move on to check for the other tree-based models.

**Bagging**

I have used the bagging.cv function with 5 v-fold cross validation and 10 trees are used. This will give me the result of a confusion matrix for bagging which showed below.

**Confusion Matrix for Bagging**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **278 (TP)** | **44 (FN)** |
| | **Class = Yes** | **5 (FP)** | **15 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $$\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$$

Accuracy for Bagging model (After CV) = $$\frac{278 + 15}{278 + 6 + 15 + 44}$$

= 0.8567 $\approx$85.67%

This can determine the performance of the Bagging model. In other words, this gives the accuracy for the Bagging model to be 85.09% correct which means that the prediction it makes will be 85.09% correct. We can see that after performing cross validation, the accuracy improves significantly from 80% to 85.67%. The error decreases too from 0.1905 to 0.1433 after cross validation.

**Boosting**

I have used the boosting.cv function with 5 v-fold cross validation and 10 trees are used. This will give me the result of a confusion matrix for boosting which showed below.

**Confusion Matrix for Boosting**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **269 (TP)** | **37 (FN)** |
| | **Class = Yes** | **14 (FP)** | **22 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $$\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$$

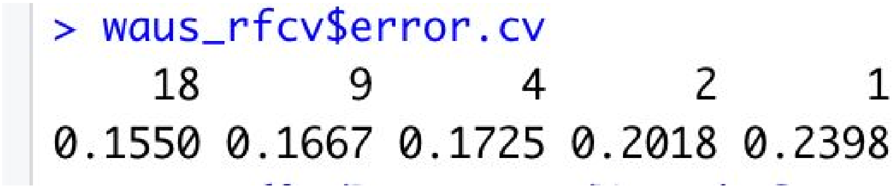
Accuracy for Boosting model (After CV) = $$\frac{269 + 14}{269 + 14 + 22 + 37}$$

= 0.8509 $\approx$85.09%

This can determine the performance of the Boosting model. In other words, this gives the accuracy for the Boosting model to be 85.09% correct which means that the prediction it makes will be 85.09% correct. The error decreases too from 0.1769 to 0.1491 after cross validation. We can see that after performing cross validation, the accuracy improves from 82.30% to 85.09%.

**Random Forest**

I have used the rfcv function with trainx with all the predictor attributes and trainy with the output result, RainTomorrow attribute. 5 v-fold cross validation and 10 trees are used. This will give me the result of a number of mtry with the error value which is shown below.



```
> waus_rfcv$error.cv
    18      9      4      2      1 
0.1550 0.1667 0.1725 0.2018 0.2398 
```

From the screenshot above, after i perform the cross validation for random forest classifier. It shows that using 18 predictors in the random forest will give the lowest errors which is 0.1550. Therefore, mtry of 18 is used.

**Confusion Matrix for Random Forest**

| | | **Observed** | |
|---|---|---|---|
| | | **Class = No** | **Class = Yes** |
| **Predicted** | **Class = No** | **270 (TP)** | **13 (FN)** |
| | **Class = Yes** | **40 (FP)** | **19 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

Accuracy = $\frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$

Accuracy for Random Forest model (After CV) = $\frac{270 + 19}{270 + 40 + 19 + 13}$

= 0.8435 $\approx$84.35%

This can determine the performance of the Random Forest model. In other words, this gives the accuracy for the Boosting model to be 84.35% correct which means that the prediction it makes will be 84.35% correct. We can see that after performing cross validation, the accuracy improves significantly from 83.67% to 84.35%. The possible reason why the accuracy value of random forest increases by only 1% might be because of the small dataset.

**Table below shows the tree-based classifier:**

| Classifier | Confusion Matrix Accuracy |
|---|---|
| Decision Tree | 0.7891 |

| | |
|---|---|
| Bagging | 0.8095 |
| Boosting | 0.8231 |
| Random Forest | 0.8367 |

**Table below shows the improved version of tree-based classifier using cross validation:**

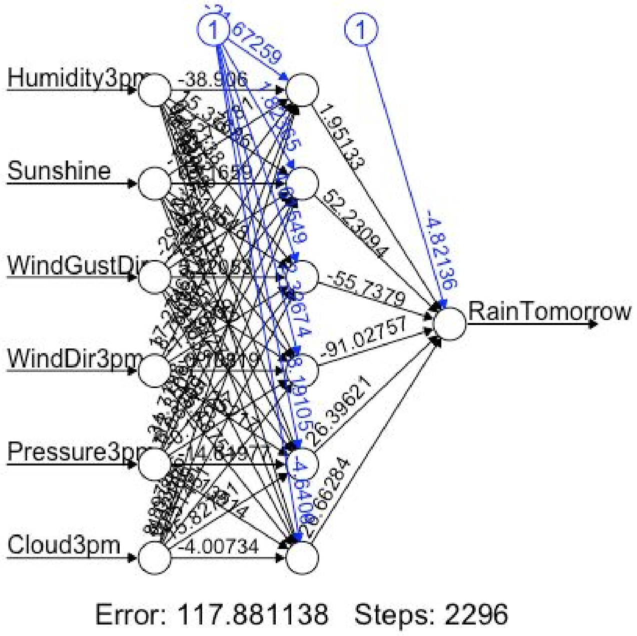
| Classifier | Confusion Matrix Accuracy |
|---|---|
| Decision Tree | 0.8095 |
| Bagging | 0.8567 |
| Boosting | 0.8509 |
| Random Forest | 0.8435 |

By comparing the Accuracy value from the two tables above, we can see that all the improved versions of tree-based classifiers have improved in the accuracy value after performing cross validation. From the first table, we can see that random forest has the highest accuracy, 0.8367, compared to the other 4 classifiers. However, bagging has the highest accuracy, 0.8567, on the improved version of the tree-based classifier. Since bagging’s accuracy is higher than random forest before and after performing the cross validation. Therefore, we can conclude that bagging is the best classifier in terms of accuracy factor. We know that random forest is actually better than bagging and boosting as random forest uses random feature selection, the trees are more independent of each other compared to bagging which will often have better predictive performance. In this case, the reason why bagging and boosting are better than random forest might be due to a really small dataset used. Hence, the bagging classifier will be used as it has the highest accuracy value compared to the other 3 tree-based classifiers. This is because accuracy of the classifiers are the most important factor in determining the best-tree classifier.

The reason why I chose the attributes that I used to find the best tree-based classifier is because as from the previous question, we have already found out the most important variables and variables that can be omitted from the dataset by looking at the importance of the variable for each classifier. We have seen that the variables, Location, RainToday and WindDir9am are not important in predicting whether it will rain Tomorrow. Therefore, these attributes such as Location, RainToday and WindDir9am are removed from the dataset to further analyse and find the best tree-based classifier. The remaining attributes in the dataset will be important or will affect the accuracy of the classifier when predicting the class attribute, RainTomorrow. Hence, it’s used in the dataset.

## Question 10

The Artificial Neural Network Classifier is implemented below using 6 attributes, Humidity3pm, Sunshine, WindGustDie, WindDir3pm, Pressure3pm and Cloud3pm. The attributes used are obtained from **Question 8,** which get the top 6 most important attributes from all the tree-based classifiers. The 6 attributes are obtained when it has a high value in Bagging, Boosting and Random Forest classifiers. First, I preprocessed the data by calling the original dataset again and used a subset function to get the 6 attributes + the RainTomorrow attribute into the new dataset. The less important attributes will not be added into the dataset to improve the accuracy value for the ANN classifier. Next, I will remove the NA values from the dataset and also convert all the used categorical data such as WindGustDIr, WIndDir3pn and RainTomorrow to binary columns as indicator variables. The values in the attribute will be converted into numerical values. Then, the data will be normalised by using scale function for all the attributes in the dataset as I have previously filtered all the important attributes into this dataset. The dataset is now ready to be used to build an ANN classification model.



The graph above shows the ANN diagram showing the weights at each synapse and the threshold of each neuron to produce output results close to the training dataset. It can be seen that there are 6 input neurons where each represents one input variable and one output neuron for the output class, RainTomorrow to determine whether it will rain tomorrow.

The dataset is first divided into 70% training dataset and 30% testing dataset before using the dataset to build the Artificial Neural Network classification model. Then, it will build an ANN model by taking the training dataset as an input into the neuralnet() function and call the 6 predictor variables, Humidity3pm, Sunshine, WindGustDie, WindDir3pm, Pressure3pm and Cloud3pm. The number of hidden layers is set to 6 which is used to determine the complexity of problems the ANN can address. This allows the neural network to decode complex, non-linear problems. Then, compute the built ANN model with the testing dataset with all the predictor variables except for the output class, RainTomorrow. The confusion matrix table will be created and shown below:

| | Predicted | |
|---|---|---|
| observed | **No** | **Yes** |
| **No** | **176 (TP)** | **18 (FN)** |
| **Yes** | **18 (FP)** | **16 (TN)** |

According to the confusion matrix table obtained above, we can calculate the accuracy of the classification model manually by using the formula below:

$$\text{Accuracy} = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + \ True\ Negative + False\ Negative}$$

$$\text{Accuracy for ANN model} = \frac{176 + 16}{176 + 18 + 16 + 18}$$

$$= 0.8421 \approx 84.21\%$$

This can determine the performance of the ANN model. The performance is considered quite good as it has 84.21% accuracy to be correct which means that the prediction it makes will be 84.21% correct.

**Table shows the Tree Based Accuracy Before and After Improved:**

| Classifier | Tree-Based Accuracy Before Improved | Tree-Based Accuracy After Improved |
|---|---|---|
| Decision Tree | 0.7891 | 0.8095 |
| Bagging | 0.8095 | 0.8567 |
| Boosting | 0.8231 | 0.8509 |
| Random Forest | 0.8367 | 0.8435 |

As ANN classifier has 0.8421 Accuracy value and according to the table above, the accuracy value of ANN classifier is better than all the tree-based classifiers before improving it using cross validation and pruning. The accuracy is slightly lower than the improved Bagging, Boosting, Random Forest tree-based classifiers. The difference between them is just a 1% difference. This means that the bagging, boosting and random forest classifiers are slightly better than ANN classifiers. Therefore, we can conclude that although ANN classifier and the performance is not the best when it's compared to all the tree-based classifiers, it is still considered a good classifier as the accuracy value is high.