FIT3152 Assignment 2

Karel Mackenzie Chandra - 30373867

This are the libraries used.

```
library(tree)
library(e1071)
library(ROCR)
library(randomForest)
```

```
randomForest 4.7-1
```

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

```
library(adabag)
```

```
Loading required package: rpart
```

```
Loading required package: caret
```

```
Loading required package: ggplot2
```

```


Attaching package: 'ggplot2'
```

```
The following object is masked from 'package:randomForest':

    margin
```

```
Loading required package: lattice
```

```
Loading required package: foreach
```

```
Loading required package: doParallel
```

```
Loading required package: iterators
```

```
Loading required package: parallel
```

```
library(rpart)
```

Report

This is the report for FIT3152 Assignment 2.

This block is given in the assignment spec

```
# Create Data Set
rm(list = ls())
WAUS <- read.csv("WarmerTomorrow2022.csv", stringsAsFactors = TRUE)
L <- as.data.frame(c(1:49))
set.seed(30373867) # Your Student ID is the random seed
L <- L[sample(nrow(L), 10, replace = FALSE),] # sample 10 locations
WAUS <- WAUS[(WAUS$Location %in% L),]
WAUS <- WAUS[sample(nrow(WAUS), 2000, replace = FALSE),] # sample 2000 rows
```

Q1

1.) Find the proportion of warmer days

```
# Find the number of warm days
# 1.) Find the proportion of warmer days
Warm <- sum(WAUS$WarmerTomorrow == 1, na.rm = TRUE)
Proportion <- (Warm/nrow(WAUS))*100
cat("Number of Warmer then previous days: ", Warm)
```

```
Number of Warmer then previous days:  1097
```

```
cat("Number of Warm Propotion: ", Proportion)
```

```
Number of Warm Propotion:  54.85
```

Here we can see the number of proportion of warmer days in compared to colder days is 54.85%

2.) Obtain description of Predictors

```
summary(WAUS)
```

```
      Day            Month             Year         Location    
 Min.   : 1.00   Min.   : 1.000   Min.   :2008   Min.   : 7.00  
 1st Qu.: 8.00   1st Qu.: 4.000   1st Qu.:2011   1st Qu.:19.00  
 Median :15.50   Median : 6.000   Median :2014   Median :31.00  
 Mean   :15.54   Mean   : 6.467   Mean   :2014   Mean   :28.41  
 3rd Qu.:23.00   3rd Qu.: 9.000   3rd Qu.:2017   3rd Qu.:43.00  
 Max.   :31.00   Max.   :12.000   Max.   :2019   Max.   :45.00  
 NA's   :16      NA's   :10       NA's   :12                    
    MinTemp         MaxTemp         Rainfall        Evaporation    
 Min.   :-3.10   Min.   : 8.50   Min.   :  0.000   Min.   : 0.000  
 1st Qu.: 7.10   1st Qu.:17.50   1st Qu.:  0.000   1st Qu.: 2.600  
 Median :11.00   Median :22.50   Median :  0.000   Median : 4.800  
 Mean   :11.65   Mean   :23.42   Mean   :  2.128   Mean   : 5.324  
 3rd Qu.:15.80   3rd Qu.:29.40   3rd Qu.:  0.600   3rd Qu.: 7.200  
 Max.   :28.30   Max.   :46.70   Max.   :110.800   Max.   :41.400  
 NA's   :22      NA's   :24      NA's   :60        NA's   :795     
    Sunshine        WindGustDir    WindGustSpeed     WindDir9am     WindDir3pm  
 Min.   : 0.000   N      : 247   Min.   :  9.0   N      : 235   S      : 200  
 1st Qu.: 4.400   SSW    : 180   1st Qu.: 30.0   SE     : 135   N      : 173  
 Median : 8.300   S      : 152   Median : 37.0   W      : 135   W      : 162  
 Mean   : 7.399   WSW    : 145   Mean   : 39.6   ENE    : 131   SW     : 149  
 3rd Qu.:10.700   SSE    : 137   3rd Qu.: 48.0   E      : 126   WNW    : 145  
 Max.   :13.900   (Other):1098   Max.   :102.0   (Other):1105   (Other):1129  
 NA's   :886      NA's   :  41   NA's   :38      NA's   : 133   NA's   :  42  
  WindSpeed9am    WindSpeed3pm    Humidity9am      Humidity3pm    
 Min.   : 0.00   Min.   : 0.00   Min.   : 12.00   Min.   :  3.00  
 1st Qu.: 7.00   1st Qu.:11.00   1st Qu.: 58.00   1st Qu.: 33.00  
 Median :13.00   Median :17.00   Median : 71.00   Median : 48.00  
 Mean   :13.88   Mean   :17.82   Mean   : 70.09   Mean   : 48.32  
 3rd Qu.:19.00   3rd Qu.:22.00   3rd Qu.: 83.25   3rd Qu.: 61.00  
 Max.   :61.00   Max.   :59.00   Max.   :100.00   Max.   :100.00  
 NA's   :29      NA's   :22      NA's   :44       NA's   :31      
  Pressure9am      Pressure3pm        Cloud9am        Cloud3pm    
 Min.   : 989.5   Min.   : 991.5   Min.   :0.000   Min.   :0.000  
 1st Qu.:1012.6   1st Qu.:1009.9   1st Qu.:1.000   1st Qu.:2.000  
 Median :1016.8   Median :1014.6   Median :6.000   Median :5.000  
 Mean   :1017.4   Mean   :1015.1   Mean   :4.584   Mean   :4.584  
 3rd Qu.:1022.7   3rd Qu.:1020.5   3rd Qu.:7.000   3rd Qu.:7.000  
 Max.   :1037.7   Max.   :1035.3   Max.   :8.000   Max.   :8.000  
 NA's   :427      NA's   :423      NA's   :770     NA's   :752    
    Temp9am         Temp3pm      WarmerTomorrow  
 Min.   :-0.60   Min.   : 7.20   Min.   :0.0000  
 1st Qu.:11.70   1st Qu.:16.20   1st Qu.:0.0000  
 Median :15.60   Median :20.75   Median :1.0000  
 Mean   :16.38   Mean   :21.93   Mean   :0.5549  
 3rd Qu.:20.50   3rd Qu.:27.70   3rd Qu.:1.0000  
 Max.   :34.50   Max.   :45.20   Max.   :1.0000  
 NA's   :30      NA's   :22      NA's   :23      
```

This is for the mean, Q1, Q3, etc a description of the predictor

```
apply(WAUS, 2, sd, na.rm = TRUE)
```

```
Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
na.rm): NAs introduced by coercion

Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
na.rm): NAs introduced by coercion

Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
na.rm): NAs introduced by coercion
```

```
          Day         Month          Year      Location       MinTemp
    8.8235172     3.3671968     3.3564444    12.9881223     6.4125937
      MaxTemp      Rainfall   Evaporation      Sunshine   WindGustDir
    7.3041308     7.1997725     3.7106345     3.8799284            NA
WindGustSpeed    WindDir9am    WindDir3pm  WindSpeed9am  WindSpeed3pm
   13.9704633            NA            NA     8.9293555     8.4533101
  Humidity9am   Humidity3pm   Pressure9am   Pressure3pm      Cloud9am
   18.2478703    20.2682628     7.3562530     7.2945975     2.8491894
     Cloud3pm       Temp9am       Temp3pm WarmerTomorrow
    2.6397076     6.4341571     7.1497243     0.4971047
```

This is for the standard deviation of the Predictor

Finding the best and worst predictor to predict WarmerTomorrow

```
fitted = lm(WAUS$WarmerTomorrow ~ WAUS$Day + WAUS$Month + WAUS$Year +
              WAUS$Location + WAUS$MinTemp + WAUS$MaxTemp + WAUS$Rainfall +
              WAUS$Evaporation + WAUS$Sunshine + WAUS$WindGustDir + WAUS$WindGustSp
eed +
              WAUS$WindDir9am + WAUS$WindDir3pm + WAUS$WindSpeed9am + WAUS$WindSpee
d3pm +
              WAUS$Humidity9am + WAUS$Humidity3pm + WAUS$Pressure9am + WAUS$Pressur
e3pm +
              WAUS$Cloud9am + WAUS$Cloud3pm + WAUS$Temp9am + WAUS$Temp3pm)
summary(fitted)
```

```


Call:
lm(formula = WAUS$WarmerTomorrow ~ WAUS$Day + WAUS$Month + WAUS$Year +
    WAUS$Location + WAUS$MinTemp + WAUS$MaxTemp + WAUS$Rainfall +
    WAUS$Evaporation + WAUS$Sunshine + WAUS$WindGustDir + WAUS$WindGustSpeed +
    WAUS$WindDir9am + WAUS$WindDir3pm + WAUS$WindSpeed9am + WAUS$WindSpeed3pm +
    WAUS$Humidity9am + WAUS$Humidity3pm + WAUS$Pressure9am +
    WAUS$Pressure3pm + WAUS$Cloud9am + WAUS$Cloud3pm + WAUS$Temp9am +
    WAUS$Temp3pm)

Residuals:
```

```
     Min       1Q   Median       3Q      Max 
-1.00375 -0.35762  0.07151  0.35088  1.03507 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)         -1.504e+01  1.025e+01  -1.466  0.14296    
WAUS$Day             1.875e-03  1.810e-03   1.036  0.30060    
WAUS$Month           5.939e-03  5.188e-03   1.145  0.25263    
WAUS$Year            2.602e-03  4.852e-03   0.536  0.59193    
WAUS$Location        1.006e-03  1.740e-03   0.579  0.56311    
WAUS$MinTemp        -2.218e-02  9.164e-03  -2.420  0.01576 *  
WAUS$MaxTemp         7.422e-02  1.655e-02   4.485  8.5e-06 ***
WAUS$Rainfall        2.388e-03  2.818e-03   0.847  0.39702    
WAUS$Evaporation    -6.154e-02  7.482e-03  -8.225  9.3e-16 ***
WAUS$Sunshine        4.088e-03  8.126e-03   0.503  0.61511    
WAUS$WindGustDirENE  1.171e-01  1.109e-01   1.056  0.29141    
WAUS$WindGustDirESE -4.178e-02  1.078e-01  -0.388  0.69848    
WAUS$WindGustDirN    1.005e-01  9.839e-02   1.021  0.30738    
WAUS$WindGustDirNE   8.878e-02  1.713e-01   0.518  0.60449    
WAUS$WindGustDirNNE  1.424e-01  1.213e-01   1.173  0.24103    
WAUS$WindGustDirNNW  1.099e-01  1.199e-01   0.917  0.35967    
WAUS$WindGustDirNW   1.981e-01  1.182e-01   1.676  0.09415 .  
WAUS$WindGustDirS    9.168e-02  1.064e-01   0.862  0.38897    
WAUS$WindGustDirSE  -3.788e-02  1.173e-01  -0.323  0.74684    
WAUS$WindGustDirSSE -7.997e-02  1.093e-01  -0.732  0.46464    
WAUS$WindGustDirSSW  3.423e-02  1.054e-01   0.325  0.74540    
WAUS$WindGustDirSW   2.642e-02  1.126e-01   0.235  0.81464    
WAUS$WindGustDirW    1.103e-01  1.111e-01   0.993  0.32123    
WAUS$WindGustDirWNW  3.404e-02  1.109e-01   0.307  0.75885    
WAUS$WindGustDirWSW -4.035e-02  1.085e-01  -0.372  0.71017    
WAUS$WindGustSpeed   2.611e-03  2.183e-03   1.196  0.23208    
WAUS$WindDir9amENE  -2.084e-03  8.200e-02  -0.025  0.97973    
WAUS$WindDir9amESE   1.472e-01  1.027e-01   1.432  0.15248    
WAUS$WindDir9amN     1.328e-01  8.647e-02   1.535  0.12516    
WAUS$WindDir9amNE    6.244e-02  9.173e-02   0.681  0.49626    
WAUS$WindDir9amNNE   3.958e-02  9.224e-02   0.429  0.66800    
WAUS$WindDir9amNNW  -3.132e-02  1.051e-01  -0.298  0.76578    
WAUS$WindDir9amNW    2.023e-02  1.153e-01   0.176  0.86069    
WAUS$WindDir9amS     1.587e-02  9.633e-02   0.165  0.86918    
WAUS$WindDir9amSE    8.579e-02  9.010e-02   0.952  0.34137    
WAUS$WindDir9amSSE   2.376e-02  9.681e-02   0.245  0.80618    
WAUS$WindDir9amSSW  -3.019e-02  1.042e-01  -0.290  0.77203    
WAUS$WindDir9amSW   -1.299e-02  1.005e-01  -0.129  0.89721    
WAUS$WindDir9amW     1.221e-01  9.427e-02   1.295  0.19567    
WAUS$WindDir9amWNW   2.950e-01  1.103e-01   2.676  0.00762 ** 
WAUS$WindDir9amWSW   4.887e-02  1.030e-01   0.474  0.63534    
WAUS$WindDir3pmENE   6.582e-02  1.434e-01   0.459  0.64642    
WAUS$WindDir3pmESE   1.439e-01  1.423e-01   1.011  0.31227    
WAUS$WindDir3pmN    -1.788e-02  1.136e-01  -0.157  0.87492    
WAUS$WindDir3pmNE    3.780e-02  1.309e-01   0.289  0.77284    
WAUS$WindDir3pmNNE   4.454e-02  1.276e-01   0.349  0.72724    
```

```
WAUS$WindDir3pmNNW  -2.946e-02  1.154e-01  -0.255  0.79851
WAUS$WindDir3pmNW   -5.470e-02  1.208e-01  -0.453  0.65080
WAUS$WindDir3pmS     1.132e-01  1.180e-01   0.959  0.33766
WAUS$WindDir3pmSE    6.137e-02  1.297e-01   0.473  0.63626
WAUS$WindDir3pmSSE   5.899e-02  1.241e-01   0.475  0.63459
WAUS$WindDir3pmSSW   1.673e-01  1.224e-01   1.366  0.17227
WAUS$WindDir3pmSW    1.130e-02  1.222e-01   0.093  0.92632
WAUS$WindDir3pmW     2.268e-02  1.234e-01   0.184  0.85429
WAUS$WindDir3pmWNW   8.336e-02  1.168e-01   0.714  0.47575
WAUS$WindDir3pmWSW   8.868e-02  1.205e-01   0.736  0.46190
WAUS$WindSpeed9am    1.923e-03  2.732e-03   0.704  0.48163
WAUS$WindSpeed3pm   -2.838e-04  3.172e-03  -0.089  0.92874
WAUS$Humidity9am     2.129e-04  1.884e-03   0.113  0.91006
WAUS$Humidity3pm    -4.240e-05  2.192e-03  -0.019  0.98457
WAUS$Pressure9am     2.116e-02  1.108e-02   1.910  0.05649 .
WAUS$Pressure3pm    -1.209e-02  1.115e-02  -1.084  0.27888
WAUS$Cloud9am        3.105e-04  8.229e-03   0.038  0.96991
WAUS$Cloud3pm        4.730e-03  8.830e-03   0.536  0.59234
WAUS$Temp9am        -1.655e-02  1.228e-02  -1.347  0.17845
WAUS$Temp3pm        -1.091e-02  1.815e-02  -0.601  0.54794
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4257 on 709 degrees of freedom
  (1225 observations deleted due to missingness)
Multiple R-squared:  0.3277,    Adjusted R-squared:  0.2661
F-statistic: 5.317 on 65 and 709 DF,  p-value: < 2.2e-16
```

Here we can see that Evaporation is the best predictor with the smallest Probabilty value, and Humidity 3pm is the worst predictor. Some models need the formula to be in Factor so WarmerTomorrow would be in Factor form, not only that but character data type would also be converted into factors.

Q2

```
WAUS <- na.omit(WAUS)
```

Here some rows are removed that contain NA so it is more suitable to use for the fitting it into model

Q3

Divide your data into a 70% training and 30% test

```
set.seed(30373867) #Student ID as random seed
train.row = sample(1:nrow(WAUS), 0.7*nrow(WAUS))
WAUS.train = WAUS[train.row,]
WAUS.test = WAUS[-train.row,]
WAUS.train$WarmerTomorrow <- as.factor(WAUS.train$WarmerTomorrow)
```

Here train and test data is generated from WAUS dataset

Q4

Implement a classification model

```
#  Calculate a descision tree
WAUS.tree = tree(WarmerTomorrow ~., data = WAUS.train)
```

Here Descision tree model is generated

```
# Calculate naive bayes
WAUS.bayes = naiveBayes(WarmerTomorrow ~. , data = WAUS.train)
```

Here Naive Bayes model is generated

```
# Bagging
WAUS.bag <- bagging(WarmerTomorrow ~. , data = WAUS.train, mfinal=5)
```

Here Bagging model is generated

```
#Boosting
WAUS.Boost <- boosting(WarmerTomorrow ~. , data = WAUS.train, mfinal=10)
```

Here Boosting model is generated

```
# Random Forest
WAUS.rf <- randomForest(WarmerTomorrow ~. , data = WAUS.train, na.action = na.exclu
de)
```

Here Random Forest model

Q5

Create a confusion matrix and report the accuracy of each model:

Decision Tree

```
# Decision Tree
# do predictions as classes and draw a table
WAUS.predtree = predict(WAUS.tree, WAUS.test, type = "class")
t1=table(Predicted_Class = WAUS.predtree, Actual_Class = WAUS.test$WarmerTomorrow)
accuracy_dt <- sum(t1[1], t1[4]) / sum(t1[1:4]) * 100
cat("\n# Decision Tree Confusion\n")
```

```

# Decision Tree Confusion
```

```
print(t1)
```

```
               Actual_Class
Predicted_Class  0  1
              0 52 38
              1 59 84
```

```
cat("Accuracy of Decision Tree: ", accuracy_dt)
```

```
Accuracy of Decision Tree:  58.3691
```

Here the accuracy of Decision Tree Clasifier is 58.4%, it have an average predicting power

Naive Bayes

```
# Naive Bayes
WAUS.predbayes = predict(WAUS.bayes, WAUS.test)
t2=table(Predicted_Class = WAUS.predbayes, Actual_Class = WAUS.test$WarmerTomorrow)
accuracy_nb <- sum(t2[1], t2[4]) / sum(t2[1:4]) * 100
cat("\n#NaiveBayes Confusion\n")
```

```

#NaiveBayes Confusion
```

```
print(t2)
```

```
               Actual_Class
Predicted_Class  0  1
              0 65 35
              1 46 87
```

```
cat("Accuracy of Naive Bayes: ", accuracy_nb)
```

```
Accuracy of Naive Bayes:  65.23605
```

The Accuracy of Naive Bayes is 65.24% it is slightly better then the decision tree but it is still not good enough

Bagging

```
# Bagging
WAUSpred.bag <- predict.bagging(WAUS.bag, WAUS.test)
accuracy_b <- sum(WAUSpred.bag$confusion[1], WAUSpred.bag$confusion[4]) / sum(WAUSp
red.bag$confusion[1:4])*100
cat("\n#Bagging Confusion\n")
```

```

#Bagging Confusion
```

```
print(WAUSpred.bag$confusion)
```

```
               Observed Class
Predicted Class  0  1
              0 56 29
              1 55 93
```

```
cat("Accuracy of Bagging: ", accuracy_b)
```

```
Accuracy of Bagging:  63.9485
```

The Accuracy of Bagging is 63.95% it is higher then Decision Tree but it is lower then Naive Bayes

Boosting

```
# Boosting
WAUSpred.boost <- predict.boosting(WAUS.Boost, newdata=WAUS.test)
accuracy_bs <- sum(WAUSpred.boost$confusion[1], WAUSpred.boost$confusion[4]) / sum(
WAUSpred.boost$confusion[1:4])*100
cat("\n#Boosting Confusion\n")
```

```

#Boosting Confusion
```

```
print(WAUSpred.boost$confusion)
```

```
               Observed Class
Predicted Class  0  1
              0 52 35
              1 59 87
```

```
cat("Accuracy of Boosting: ", accuracy_bs)
```

```
Accuracy of Boosting:  59.65665
```

The Accuracy of Boosting is 59.66%, it is higher then Decision Tree but it is still lower then Bagging and Naive Bayes

Random Forest

```
# Random Forest
WAUSpredrf <- predict(WAUS.rf, WAUS.test)
t3=table(Predicted_Class = WAUSpredrf, Actual_Class = WAUS.test$WarmerTomorrow)
accuracy_rf <- sum(t3[1], t3[4]) / sum(t3[1:4])*100
cat("\n#Random Forest Confusion\n")
```

```

#Random Forest Confusion
```

```
print(t3)
```

```
               Actual_Class
Predicted_Class   0   1
              0  51  15
              1  60 107
```

```
cat("Accuracy of Random Forest", accuracy_rf)
```

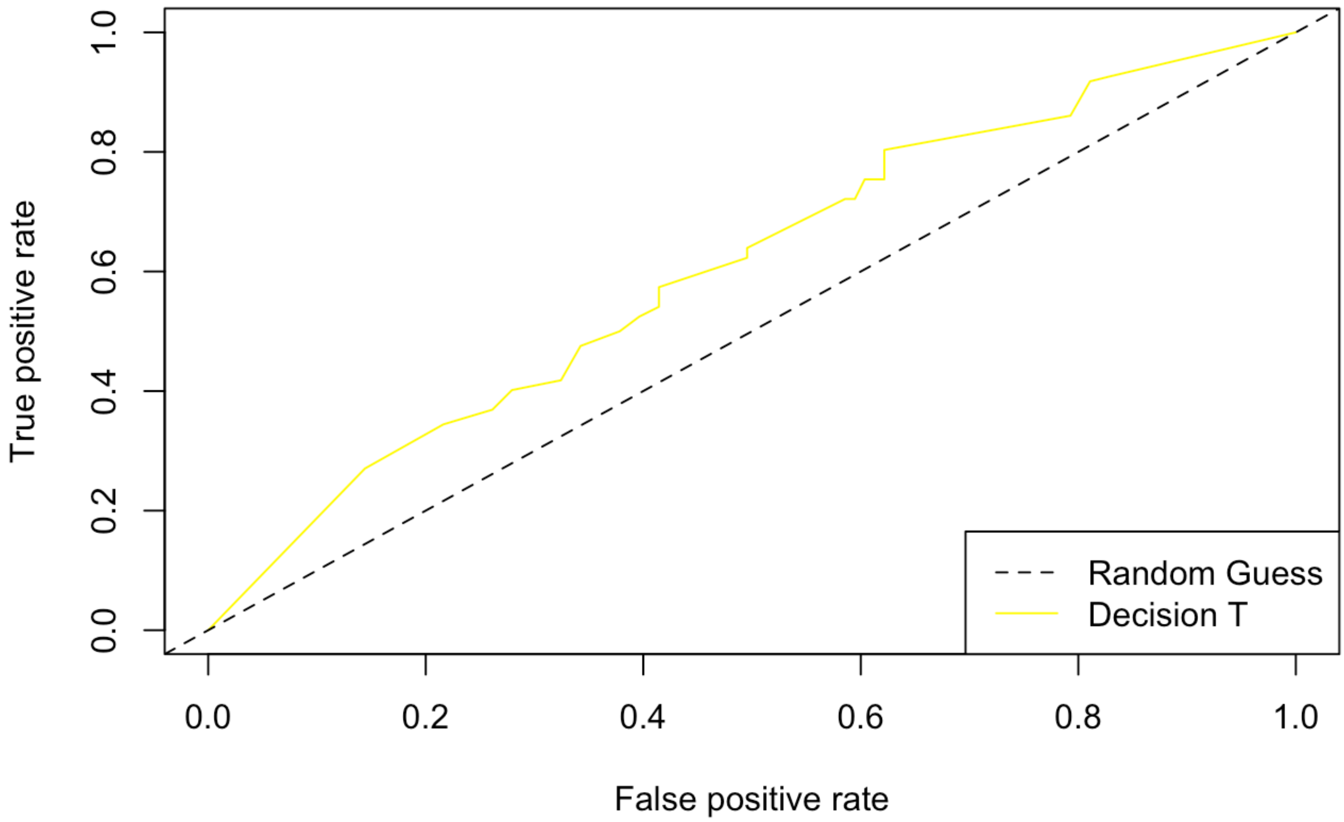
```
Accuracy of Random Forest 67.81116
```

From the above accuracies we can see that random forest as the highest accuracy against all the other classifiers model with 67.8%, this can be assume as the best classifier.

Q6

Create ROC and find AUC for each model.

```
#Decision tree
# do predictions as probabilities and draw ROCs
WAUS.pred.tree = predict(WAUS.tree, WAUS.test, type = "vector")
# computing a simple ROC curve (x-axis: fpr, y-axis: tpr)
# labels are actual values, predictors are probability of class
WAUSpred_dt <- prediction( WAUS.pred.tree[,2], WAUS.test$WarmerTomorrow)
WAUSperf_dt <- performance(WAUSpred_dt,"tpr","fpr")
plot(WAUSperf_dt, col = "yellow")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess","Decision T"),
       lty = c(2,1),
       col = c("black", "yellow"))
```

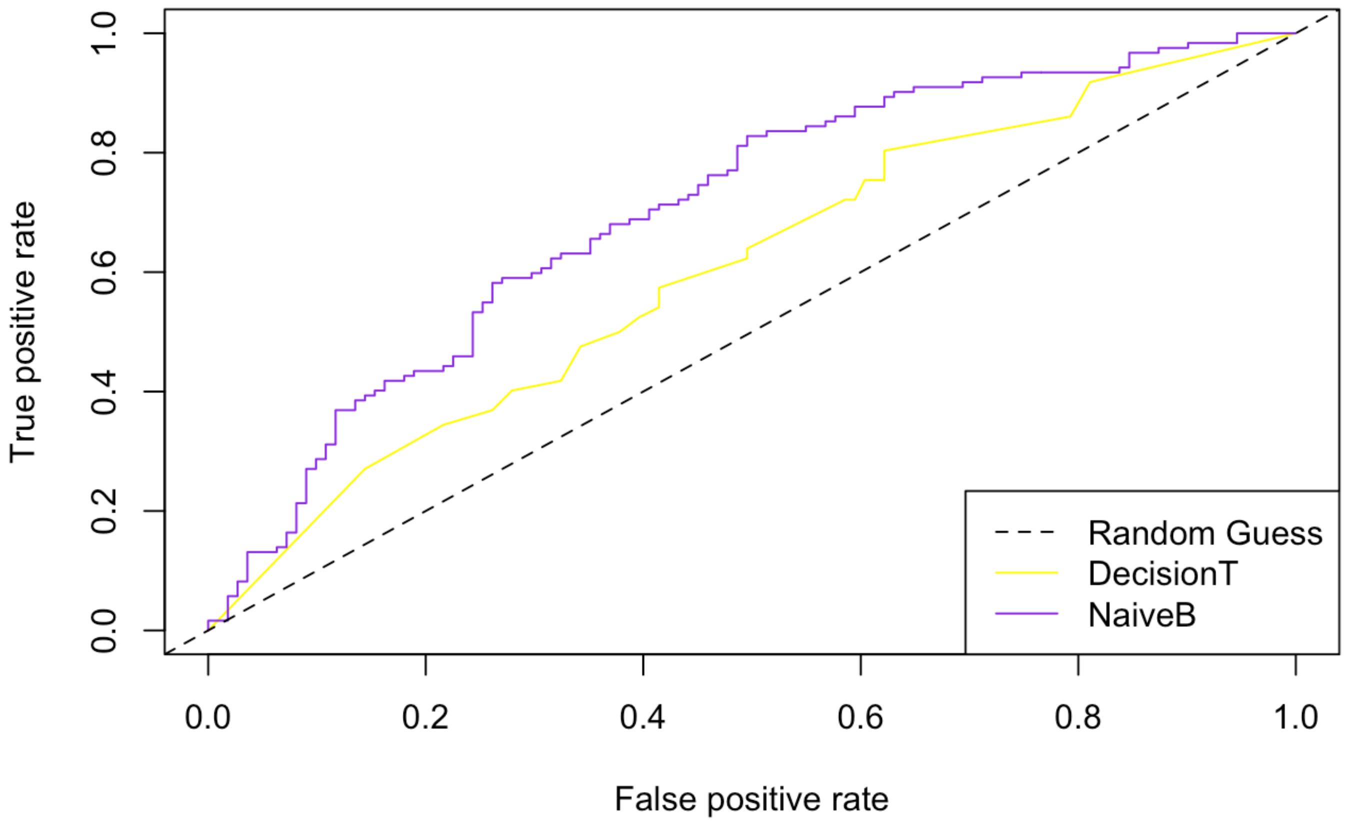



```
# calculate and print auc
cauc_dt = performance(WAUSpred_dt, "auc")
cat("The AUC of Decision Tree: ", as.numeric(cauc_dt@y.values))
```

```
The AUC of Decision Tree:  0.6040097
```

This ROC plot is for decision tree

```
# Naive Bayes
# outputs as confidence levels
WAUSpred.bayes = predict(WAUS.bayes, WAUS.test, type = 'raw')
WAUSpred_nb <- prediction( WAUSpred.bayes[,2], WAUS.test$WarmerTomorrow)
WAUSperf_nb <- performance(WAUSpred_nb,"tpr","fpr")
plot(WAUSperf_dt, col = "yellow")
plot(WAUSperf_nb, add = TRUE, col = "blueviolet")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess","DecisionT", "NaiveB"),
       lty = c(2,1,1),
       col = c("black", "yellow", "blueviolet"))
```

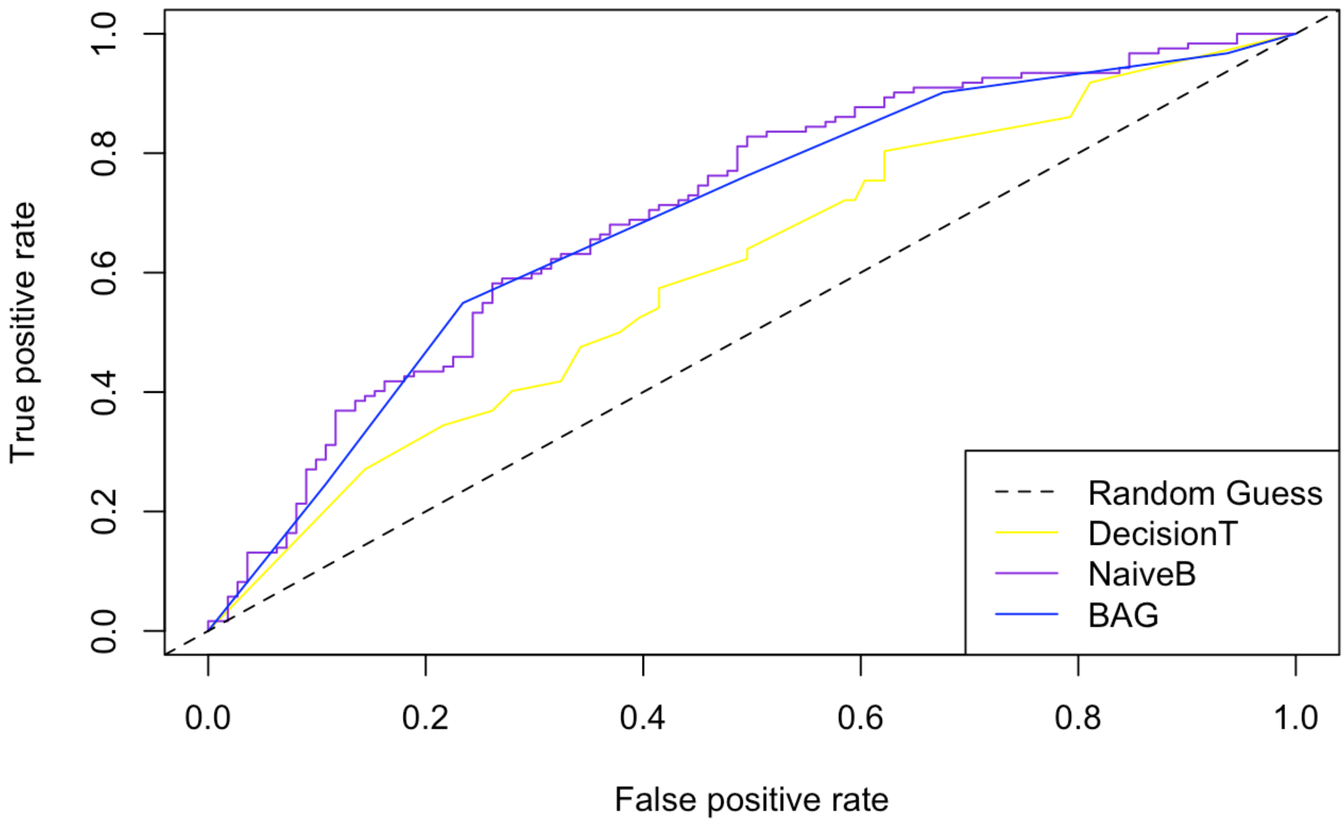



```
# calculate and print auc
cauc_nb = performance(WAUSpred_nb, "auc")
cat("The AUC of Naive Bayes: ", as.numeric(cauc_nb@y.values))
```

```
The AUC of Naive Bayes:  0.7041057
```

This ROC plot for Naive Bayes

```
# Bagging
WAUSBagpred <- prediction( WAUSpred.bag$prob[,2], WAUS.test$WarmerTomorrow)
WAUSBagperf <- performance(WAUSBagpred,"tpr","fpr")
plot(WAUSperf_dt, col = "yellow")
plot(WAUSperf_nb, add = TRUE, col = "blueviolet")
plot(WAUSBagperf, add=TRUE, col = "blue")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess","DecisionT", "NaiveB", "BAG"),
       lty = c(2,1,1,1),
       col = c("black","yellow", "blueviolet", "blue"))
```

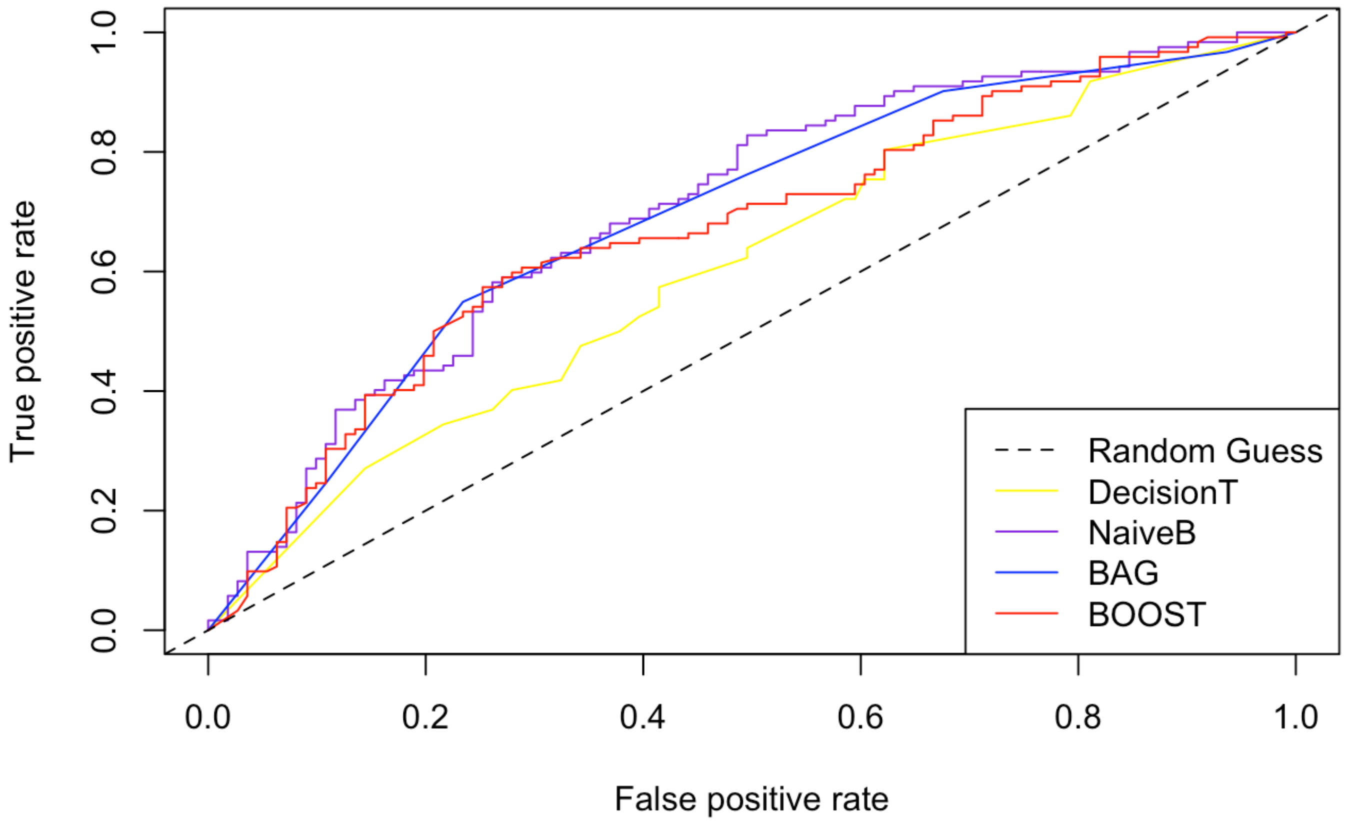



```
# calculate and print auc
cauc_bag = performance(WAUSBagpred, "auc")
cat("The AUC of Bagging: ", as.numeric(cauc_bag@y.values))
```

```
The AUC of Bagging:  0.6908138
```

This ROC plot is for bagging

```
# Boosting
WAUSBoostpred <- prediction( WAUSpred.boost$prob[,2], WAUS.test$WarmerTomorrow)
WAUSBoostperf <- performance(WAUSBoostpred,"tpr","fpr")
plot(WAUSperf_dt, col = "yellow")
plot(WAUSperf_nb, add = TRUE, col = "blueviolet")
plot(WAUSBagperf, add=TRUE, col = "blue")
plot(WAUSBoostperf, add=TRUE, col = "red")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess","DecisionT", "NaiveB", "BAG", "
BOOST"),
       lty = c(2,1,1,1,1),
       col = c("black", "yellow", "blueviolet", "blue", "red"))
```

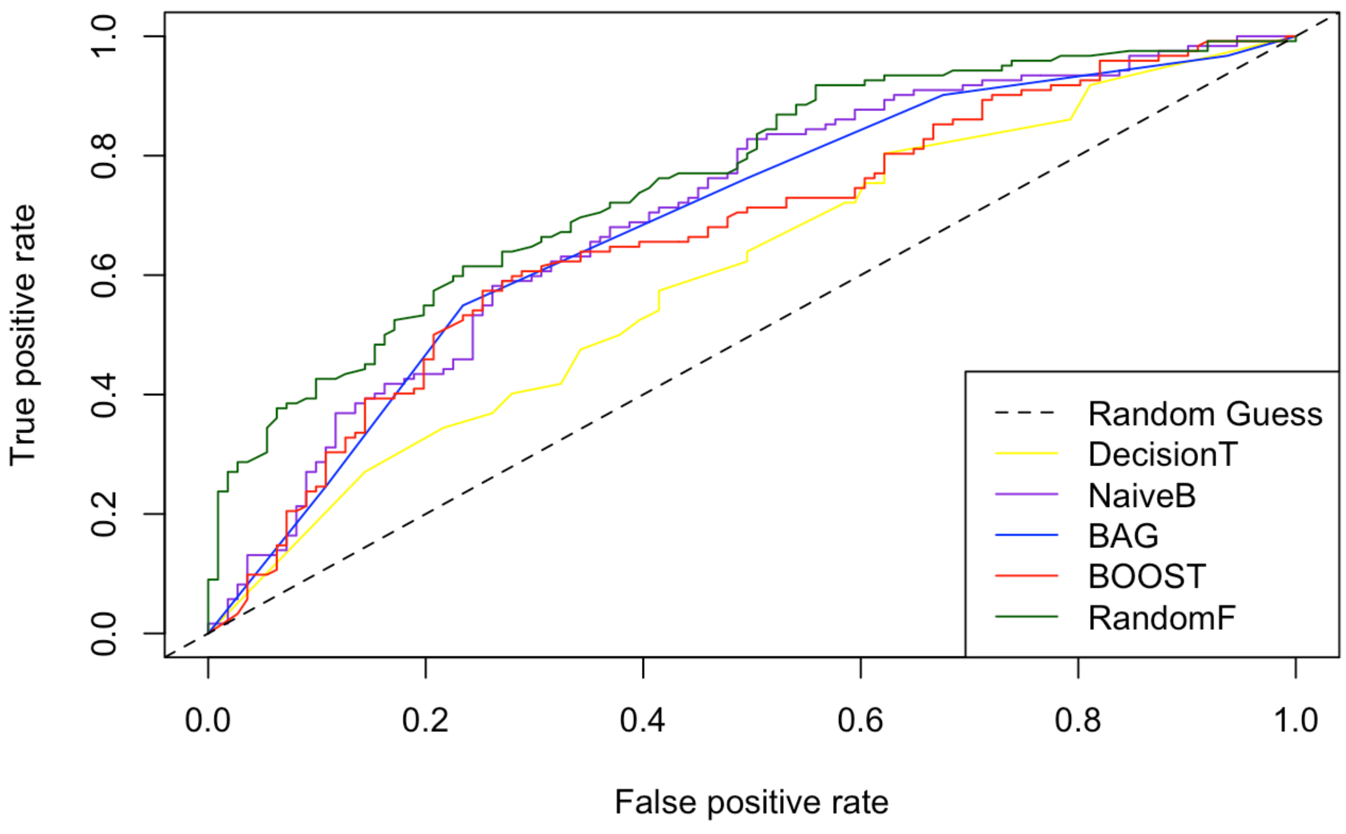



```
# calculate and print auc
cauc_boo = performance(WAUSBoostpred, "auc")
cat("The AUC of Boosting: ", as.numeric(cauc_boo@y.values))
```

```
The AUC of Boosting:  0.6711712
```

This ROC plot is for boosting

```
# Random Forest
WAUSpred.rf <- predict(WAUS.rf, WAUS.test, type="prob")
WAUSFpred <- prediction( WAUSpred.rf[,2], WAUS.test$WarmerTomorrow)
WAUSFperf <- performance(WAUSFpred,"tpr","fpr")
plot(WAUSperf_dt, col = "yellow")
plot(WAUSperf_nb, add = TRUE, col = "blueviolet")
plot(WAUSBagperf, add=TRUE, col = "blue")
plot(WAUSBoostperf, add=TRUE, col = "red")
plot(WAUSFperf, add=TRUE, col = "darkgreen")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess", "DecisionT", "NaiveB", "BAG",
"BOOST", "RandomF"),
       lty = c(2,1,1,1,1,1),
       col = c("black", "yellow", "blueviolet", "blue", "red", "darkgreen"))
```




```
# calculate and print auc
cauc_rf = performance(WAUSFpred, "auc")
cat("The AUC of Random Forest: ", as.numeric(cauc_rf@y.values))
```

```
The AUC of Random Forest:  0.7594521
```

This ROC plot is for random forest.

Here we can see that Random Forest has the best AUC and ROC curve in comparison with the rest

Q7

Create Table comparing all classifiers

```
Classifiers <- c("Decision Tree", "Naive Bayes", "Bagging", "Boosting", "Random For
est")
Accuracy <- c(accuracy_dt, accuracy_nb, accuracy_b, accuracy_bs, accuracy_rf)
AUC <- c(as.numeric(cauc_dt@y.values), as.numeric(cauc_nb@y.values), as.numeric(cau
c_bag@y.values), as.numeric(cauc_boo@y.values), as.numeric(cauc_rf@y.values))
df_c <- data.frame(Classifiers, Accuracy, AUC)
print(df_c)
```

```
    Classifiers Accuracy       AUC
1 Decision Tree 58.36910 0.6040097
2   Naive Bayes 65.23605 0.7041057
3       Bagging 63.94850 0.6908138
4      Boosting 59.65665 0.6711712
5 Random Forest 67.81116 0.7594521
```

From the table of comparison we can see that Random Forest is a single ‘best’ classifier with the highest AUC, Accuracy, and the most outer graph in the ROC chart.

Q8

Finding importance of each model

```
# Decision Tree
Waus_dt <- tree(WarmerTomorrow ~ ., WAUS.train)
summary(Waus_dt)
```

```


Classification tree:
tree(formula = WarmerTomorrow ~ ., data = WAUS.train)
Variables actually used in tree construction:
 [1] "WindDir9am"    "Evaporation"   "Pressure9am"   "MaxTemp"
 [5] "WindGustDir"   "WindDir3pm"    "WindGustSpeed" "Sunshine"
 [9] "WindSpeed9am"  "Temp9am"       "Humidity3pm"   "WindSpeed3pm"
[13] "Month"         "Humidity9am"
Number of terminal nodes:  34
Residual mean deviance:  0.612 = 310.9 / 508
Misclassification error rate: 0.1292 = 70 / 542
```

Use to find the variable used in Decision Tree

```
# Bagging
WAUS.bag$importance
```

```
     Cloud3pm      Cloud9am           Day   Evaporation   Humidity3pm 
    1.3243465     0.0000000     1.7603269     6.2600568     6.5588906 
  Humidity9am      Location       MaxTemp       MinTemp         Month 
    0.0000000     1.3232150     4.1456439     0.0000000     0.0000000 
  Pressure3pm   Pressure9am      Rainfall      Sunshine       Temp3pm 
    6.2567104     7.6171224     0.0000000     4.5879036     8.4253604 
      Temp9am    WindDir3pm    WindDir9am   WindGustDir WindGustSpeed 
    2.9899348     9.8712258    17.8234226    14.9137269     0.0000000 
  WindSpeed3pm  WindSpeed9am          Year 
    0.0000000     5.2153805     0.9267328 
```

Use to find the important variables on Bagging

```
# Boosting
WAUS.Boost$importance
```

```
     Cloud3pm      Cloud9am           Day   Evaporation   Humidity3pm 
    0.8695770     0.0000000     2.0479485     4.1674206     3.8844222 
  Humidity9am      Location       MaxTemp       MinTemp         Month 
    0.9894385     2.4520809     6.2465139     3.1596976     1.9596566 
  Pressure3pm   Pressure9am      Rainfall      Sunshine       Temp3pm 
    1.0978066     3.2636373     0.3065639     5.5173593     4.1480147 
      Temp9am    WindDir3pm    WindDir9am   WindGustDir WindGustSpeed 
    3.8981448    16.3670325    18.5584061    14.4400995     1.9948349 
  WindSpeed3pm  WindSpeed9am          Year 
    2.4209192     1.2153694     0.9950560 
```

Use to find the important variables on Boosting

```
# Random Forest
print(WAUS.rf$importance)
```

```
               MeanDecreaseGini
Day                    7.655641
Month                  5.718535
Year                   6.107831
Location               4.101221
MinTemp               11.188893
MaxTemp               13.805936
Rainfall               4.099077
Evaporation           13.712702
Sunshine              13.245452
WindGustDir           24.301272
WindGustSpeed          7.208276
WindDir9am            30.430186
WindDir3pm            23.935020
WindSpeed9am           8.481398
WindSpeed3pm           7.430196
Humidity9am            7.753982
Humidity3pm           13.502740
Pressure9am           13.415599
Pressure3pm           10.783107
Cloud9am               5.401629
Cloud3pm               6.572308
Temp9am               10.074577
Temp3pm               15.553601
```

Use to find the important variables on Random Forest

After examining each model, I can say that, variables WindGustDir, WindDir9am, WindDir3pm are the most important to predict whether it will be warmer tomorrow or not. I can say that because First the decision tree uses it as the root, not only that but each models importance also shows that this three values is has the highest importance among the other values. For the variables that have a very little effect on performance are: Location, Rainfall, Cloud9am, Cloud3pm, I can say this because looking at the decision tree it is not use as any root, for the rest model this variables have the lowest importance among the rest.

Q9

```
# Q9
# Exclude not important predictors
WAUS.train_b <- subset(WAUS.train, select = -c(Cloud3pm, Cloud9am,
                                               Day, Humidity9am,
                                               Location, MinTemp,
                                               Month, Rainfall,
                                               WindGustSpeed, Year,
                                               WindDir9am, WindDir3pm, WindGustDir)
)
WAUS.test_b <- subset(WAUS.test, select = -c(Cloud3pm, Cloud9am,
                                             Day, Humidity9am,
                                             Location, MinTemp,
                                             Month, Rainfall,
                                             WindGustSpeed, Year,
                                             WindDir9am, WindDir3pm, WindGustDir))
WAUS.train_b$WarmerTomorrow <- as.factor(WAUS.train_b$WarmerTomorrow)
```

Here I exclude the attributes from above, by looking at the importance of each particular attributes to the model, through Q8 and the readability of the tree when it is produced, when the variable above are included it would be a lot harder to read the tree.

```
Waus_dt_b<- tree(WarmerTomorrow ~ ., WAUS.train_b)
summary(Waus_dt_b)
```

```
Classification tree:
tree(formula = WarmerTomorrow ~ ., data = WAUS.train_b)
Variables actually used in tree construction:
[1] "Temp3pm"      "Evaporation"  "Sunshine"     "Humidity3pm"  "Pressure9am"
[6] "Temp9am"      "Pressure3pm"  "WindSpeed9am" "MaxTemp"
Number of terminal nodes:  21
Residual mean deviance:  0.8505 = 443.1 / 521
Misclassification error rate: 0.2362 = 128 / 542
```

Here I choose decision tree as a simple classifier because, it is doable by hand, and it can check and eliminate unimportant attribute through the summary. This classifier has a lower terminal node in comparison to the original decision tree, making it easier by hand and increases the readability of the tree

```
# do predictions as classes and draw a table
Waus_dt_b.predtree = predict(Waus_dt_b, WAUS.test_b, type = "class")
t1=table(Actual_Class = WAUS.test_b$WarmerTomorrow, Predicted_Class =Waus_dt_b.pred
tree)
cat("\n#Decsion Tree Confusion Better\n")
```

```
#Decsion Tree Confusion Better
```

```
print(t1)
```

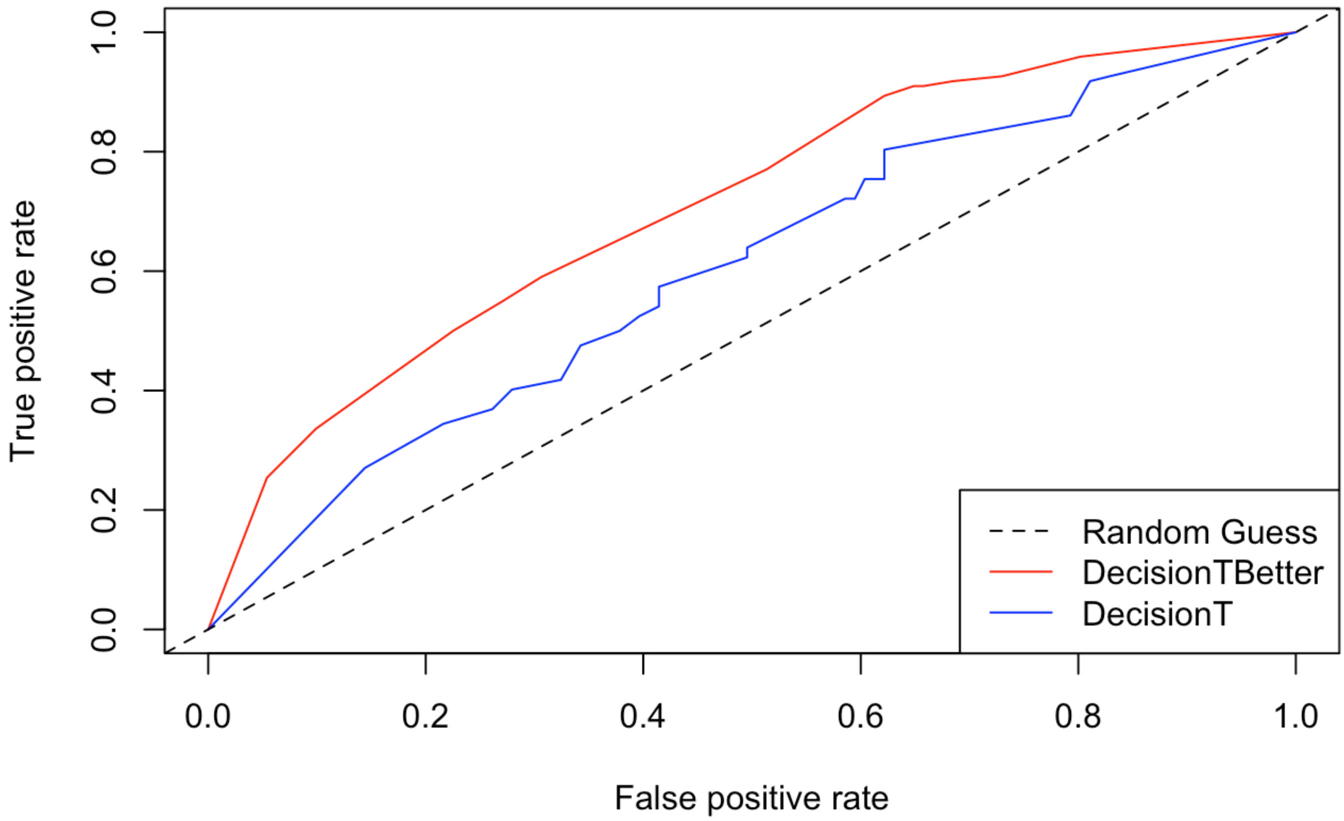
```
            Predicted_Class
Actual_Class   0   1
           0  42  69
           1  13 109
```

```
accuracy_dt_b <- sum(t1[1], t1[4]) / sum(t1[1:4])*100
cat("Accuracy of Simple Tree: ", accuracy_dt_b)
```

```
Accuracy of Simple Tree:  64.80687
```

This model also produces a better accuracy in comparison with the original model

```
# do predictions as probabilities and draw ROC
Waus_dt_b.pred.tree = predict(Waus_dt_b, WAUS.test_b, type = "vector")
Waus_dt_bDpred <- prediction( Waus_dt_b.pred.tree[,2], WAUS.test_b$WarmerTomorrow)
Waus_dt_bDperf <- performance(Waus_dt_bDpred,"tpr","fpr")
plot(Waus_dt_bDperf, col = "red")
plot(WAUSperf_dt, add=TRUE, col = "blue")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess", "DecisionTBetter", "DecisionT"
),
       lty = c(2,1,1),
       col = c("black", "red","blue"))
```




```
# calculate and print auc
cauc_dt_b = performance(Waus_dt_bDpred, "auc")
cat("The AUC of Simple Tree: ", as.numeric(cauc_dt_b@y.values))
```

```
The AUC of Simple Tree:  0.7088318
```

This model also has a higher overall AUC as it can be seen from the ROC plot above, it is closer to the TPR then the original decision tree.

```
Classifiers <- c("Original Decision Tree", "Better Decision Tree")
Accuracy <- c(accuracy_dt,accuracy_dt_b)
AUC <- c(as.numeric(cauc_dt@y.values), as.numeric(cauc_dt_b@y.values))
df_c9 <- data.frame(Classifiers, Accuracy, AUC)
print(df_c9)
```

```
             Classifiers Accuracy       AUC
1 Original Decision Tree 58.36910 0.6040097
2   Better Decision Tree 64.80687 0.7088318
```

In this table we can see that Accuracy and AUC increases in the better/simple model. It can be say that making a simple doable tree by eliminating unimportant variable makes the classifier better then the original Decision Tree. Overall choosing the right predictor and classifier are the most important factors when creating a simpler model, by deselecting some predictor it increases the overall accuracy of the model, the attributes that are selected are based on the importance model in Q8. And also by choosing the right base classifier to be worked on could easily be done by hand. Achieving a simple model criteria.

Q10

Create the best tree classifier

```
# Exclude not important predictors
WAUS.train_cv <- subset(WAUS.train, select = -c(Location, Rainfall, Month, Cloud9am
))
WAUS.test_cv <- subset(WAUS.test, select = -c(Location, Rainfall, Month, Cloud9am))
```

Here I choose from the single best classifier from Q7 which is Random Forest but to make it even better I excluded first some attributes/predictor from the data. Here I choose the attributed from above after looking through the importance of attributes in Random Forest from Q8. I found that this 4 attributed comes one of the lowest among the other attributes. Hence by removing those it will automatically increase the Accuracy and AUC of the classifier making it the best. I choose this model because it is the best classifier among the rest it also have a feature where we can check which attribute our least important making the classifier Accuracy and AUC even better from before.

```
# Random Forest
WAUS.rf_cv <- randomForest(WarmerTomorrow ~. , data = WAUS.train_cv, na.action = na
.exclude)
```

Here we predict with the choosen model

```
Waus.pred = predict(WAUS.rf_cv, WAUS.test_cv)
tP <- table(actual = WAUS.test_cv$WarmerTomorrow, predicted = Waus.pred)
accuracy_rf_cv <- sum(tP[1], tP[4]) / sum(tP[1:4])*100
cat("\n#Random Forest Confusion Better\n")
```

```


#Random Forest Confusion Better
```

```
print(tP)
```

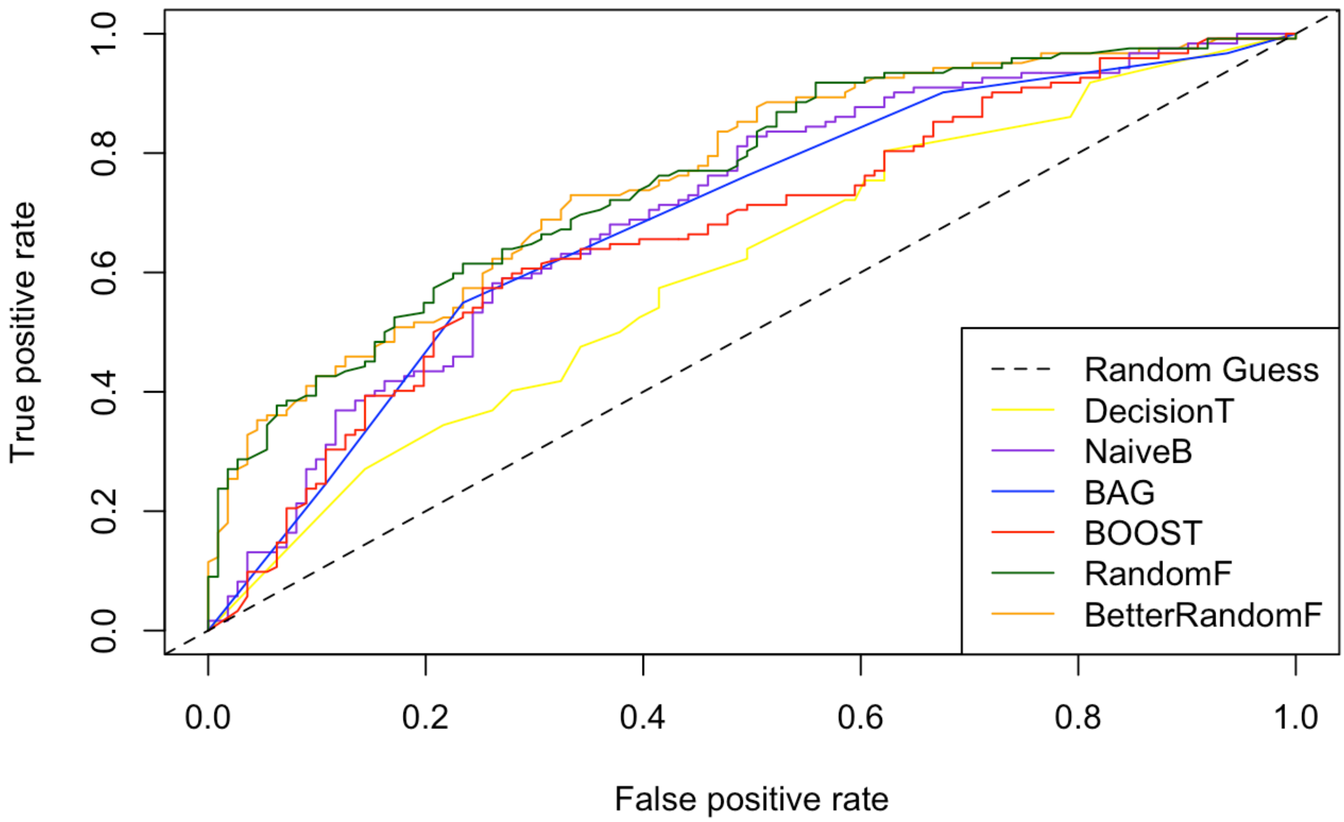
```
      predicted
actual   0   1
     0  53  58
     1  14 108
```

```
cat("Accuracy of Best Classifier: ", accuracy_rf_cv)
```

```
Accuracy of Best Classifier:  69.09871
```

Then we produce the confusion matrix along side the Accuracy of the classifier

```
Waus.pred_cv = predict(WAUS.rf_cv, WAUS.test_cv, type="prob")
WAUSpred_cv <- prediction(Waus.pred_cv[,2], WAUS.test_cv$WarmerTomorrow)
WAUSperf_cv <- performance(WAUSpred_cv,"tpr","fpr")
plot(WAUSperf_cv, col = "orange")
plot(WAUSperf_dt, add = TRUE, col = "yellow")
plot(WAUSperf_nb, add = TRUE, col = "blueviolet")
plot(WAUSBagperf, add=TRUE, col = "blue")
plot(WAUSBoostperf, add=TRUE, col = "red")
plot(WAUSFperf, add=TRUE, col = "darkgreen")
abline(0,1, lty = 2)
#Legend
legend(x = "bottomright", legend = c("Random Guess", "DecisionT", "NaiveB", "BAG",
"BOOST", "RandomF", "BetterRandomF"),
       lty = c(2,1,1,1,1,1,1),
       col = c("black", "yellow", "blueviolet", "blue", "red", "darkgreen", "orange
"))
```



```
# calculate and print auc
cauc_rf_cv = performance(WAUSpred_cv, "auc")
cat("The AUC of Best Classifier: ", as.numeric(cauc_rf_cv@y.values))
```

```
The AUC of Best Classifier:  0.7614459
```

Here we compare the best classifier against the once found in Q4

```
Classifiers <- c("Decision Tree", "Naive Bayes", "Bagging", "Boosting", "Random For
est", "Better Random Forest")
Accuracy <- c(accuracy_dt, accuracy_nb, accuracy_b, accuracy_bs, accuracy_rf, accur
acy_rf_cv)
AUC <- c(as.numeric(cauc_dt@y.values), as.numeric(cauc_nb@y.values), as.numeric(cau
c_bag@y.values), as.numeric(cauc_boo@y.values), as.numeric(cauc_rf@y.values), as.nu
meric(cauc_rf_cv@y.values))
df_c10 <- data.frame(Classifiers, Accuracy, AUC)
print(df_c10)
```

```
           Classifiers Accuracy       AUC
1        Decision Tree 58.36910 0.6040097
2          Naive Bayes 65.23605 0.7041057
3              Bagging 63.94850 0.6908138
4             Boosting 59.65665 0.6711712
5        Random Forest 67.81116 0.7594521
6 Better Random Forest 69.09871 0.7614459
```

From the table above we can see that this classifier is the best among the rest, with a few adjustments in its parameter by looking and removing some unfit/less important attributes, it is better then the original Random Forest Classifier in Accuracy and AUC wise, not only that but in the ROC plot it is closer to TPR then the rest of the classifier.

Q11

Create ANN classifier

```
# clean up the environment before starting
rm(list = ls())
library(neuralnet)
```

```
Attaching package: 'neuralnet'
```

```
The following object is masked from 'package:ROCR':

    prediction
```

```
options(digits=4)
WAUS <- read.csv("WarmerTomorrow2022.csv")
L <- as.data.frame(c(1:49))
set.seed(30373867) # Your Student ID is the random seed
L <- L[sample(nrow(L), 10, replace = FALSE),] # sample 10 locations
WAUS <- WAUS[(WAUS$Location %in% L),]
WAUS <- WAUS[sample(nrow(WAUS), 2000, replace = FALSE),] # sample 2000 rows
```

Perform data preprocessing before using with ANN model

```
# Remove NA value so that model can work
WAUS <- na.omit(WAUS)
# Move not integer type to new df
Waus_f <- WAUS[, c(10,12,13)]
WAUS <- WAUS[, -c(10,12,13)]
# Create Normalize function
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
# Use the function
WausNorm <- as.data.frame(lapply(WAUS, normalize))
# Return back factor data
WausNorm$WindGustDir <- Waus_f$WindGustDir
WausNorm$WindDir9am <- Waus_f$WindDir9am
WausNorm$WindDir3pm <- Waus_f$WindDir3pm
# convert WarmerTomorrow to a numerical form
WausNorm$WarmerTomorrow = as.numeric(WausNorm$WarmerTomorrow)
```

Create Train and Test data for the classifier

```
# make training and test sets
set.seed(30373867) #Student ID as random seed
train.row = sample(1:nrow(WausNorm), 0.8*nrow(WausNorm))
WAUS.train = WausNorm[train.row,]
WAUS.test = WausNorm[-train.row,]
```

Use the ANN classifier

```
# From the tutorial
##########################################################################
#Abishek's improved solution
#Binomial classification: predict the probability of belonging to class 1 and if th
e probability is less than 0.5 consider it predicted as class 0
WAUS.nn = neuralnet(WarmerTomorrow ~ Evaporation + Humidity3pm + MaxTemp
                    + Pressure3pm + Pressure9am + Sunshine
                    + Temp3pm + Temp9am + WindSpeed9am,
                    WAUS.train, hidden=3, linear.output = FALSE)
#Neural Network
WAUS.pred = compute(WAUS.nn, WAUS.test[c(6, 8, 9, 11, 14, 15, 16, 19, 20)])
prob <- WAUS.pred$net.result
pred <- ifelse(prob>0.5, 1, 0)
```

Create the confusion matrix and accuracy

```
#confusion matrix
tA <- table(observed = WAUS.test$WarmerTomorrow, predicted = pred)
# Find the accuracy
accuracy_ann <- sum(tA[1], tA[4]) / sum(tA[1:4])*100
cat("\n#ANN Confusion\n")
```

```

#ANN Confusion
```

```
print(tA)
```

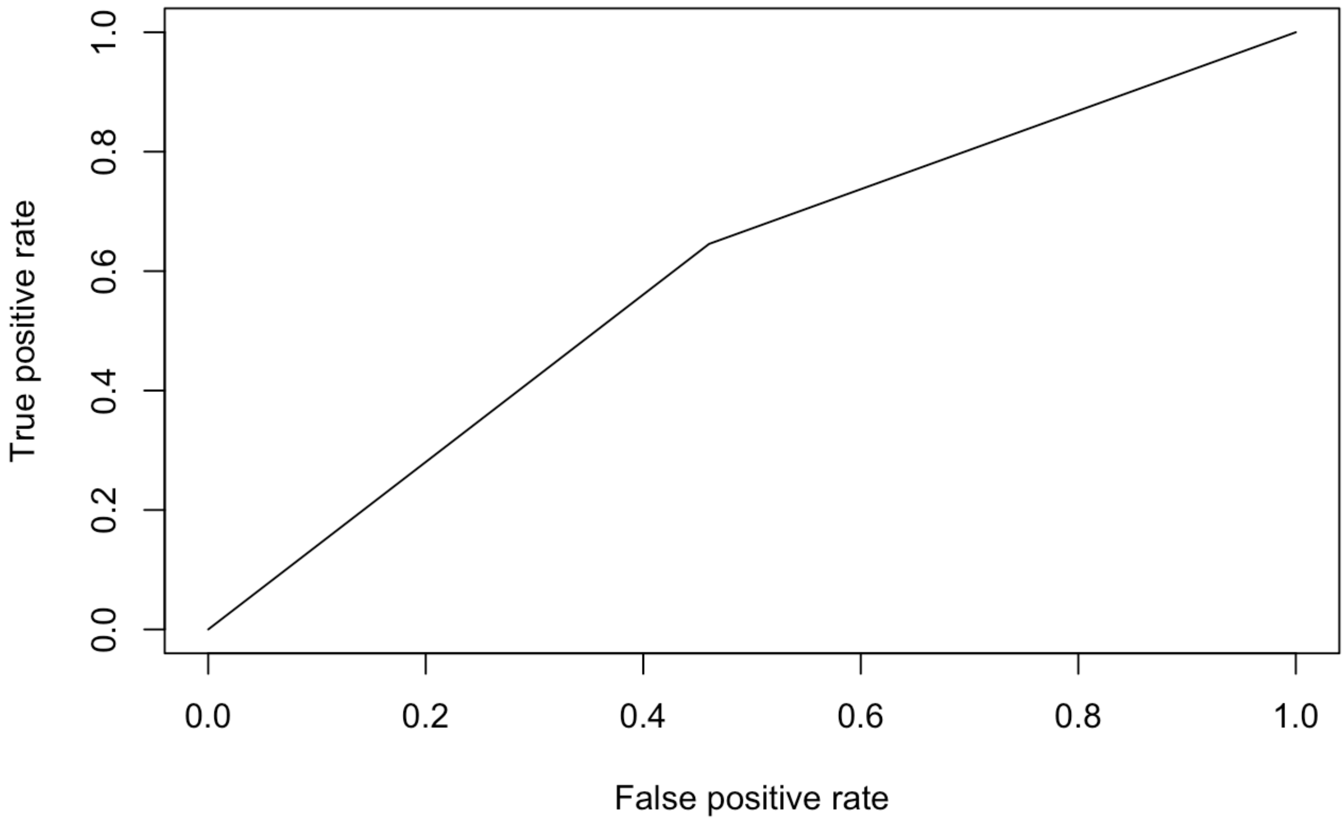
```
        predicted
observed  0  1
       0 41 35
       1 28 51
```

```
cat("Accuracy of ANN: ", accuracy_ann)
```

```
Accuracy of ANN:  59.35
```

Create an ROC plot and find AUC for ANN classifier

```
detach(package:neuralnet,unload = T)
library(ROCR)
WAUSnn.pred <- prediction(pred, WAUS.test$WarmerTomorrow)
WAUSnn.pref <- performance(WAUSnn.pred, "tpr", "fpr")
plot(WAUSnn.pref)
```




```
# Find the AUC
cauc_ann = performance(WAUSnn.pred, "auc")
cat("The AUC of ANN: ", as.numeric(cauc_ann@y.values))
```

```
The AUC of ANN:  0.5925
```

In this question I use all the attribute that are important from each models in Q8, though I only use numerical values as the function won't accept non-numerical values. However the numerical values used are still in the high importance. meaning it full fills the condition. In the data preprocessing part first I remove all rows that contain NAs, then I move non-numerical data into new dataframe for normalization, after normalization, I assign the non-numerical data back, then split the data into Train and Test datasets of 80%-20%. Comparing against other classifier ANN accuracy is slightly better then basic Decision Tree but it is no better then the rest e.g. Naive bayes, Bagging, Boosting, Random Forest. It can be say that this Classifier is just an average Classifier.