Exploring Data

On exploring the given data, it is calculated that 58.71% of days are warmer than the previous day and 41.29% are colder.

The real-valued attributes considered for further analysis are: MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm . Their descriptions are as follows:

MinTemp	
Min	-3.2
1st Qu.	8.5
Median	12.5
Mean	12.7
3rd Qu.	17.1
Max	29.2
NAs	42
SD	5.74

MaxTemp	
Min	7.6
1st Qu.	18.5
Median	23.0
Mean	23.7
3rd Qu.	28.4
Max	44.6
NAs	20
SD	6.70

Rainfall	
Min	0.0
1st Qu.	0.0
Median	0.0
Mean	2.1
3rd Qu.	0.6
Max	118.2
NAs	62
SD	6.84

Evaporation	
Min	0
1st Qu.	3
Median	5
Mean	7
3rd Qu.	9
Max	63
NAs	1230
SD	68.3

Sunshine	
Min	0
1st Qu.	6
Median	9
Mean	8
3rd Qu.	11
Max	14
NAs	1433
SD	3.82

WindGustSpeed	
Min	15.0
1st Qu.	31.0
Median	39.0
Mean	40.5
3rd Qu.	48.0
Max	115.0
NAs	223
SD	13.01

WindSpeed9am	
Min	0.0
1st Qu.	9.0
Median	15.0
Mean	15.1
3rd Qu.	20.0
Max	57.0
NAs	48
SD	9.04

WindSpeed3pm	
Min	0.0
1st Qu.	13.0
Median	19.0
Mean	18.9
3rd Qu.	24.0
Max	65.0
NAs	87
SD	9.09

Humidity9am	
Min	4.0
1st Qu.	55.0
Median	68.0
Mean	67.4
3rd Qu.	83.0
Max	100.0
NAs	51
SD	19.55

Humidity3pm	
Min	2.0
1st Qu.	31.0
Median	52.0
Mean	50.1
3rd Qu.	68.0
Max	100.0
NAs	96
SD	22.92

Pressure9am	
Min	993
1st Qu.	1014
Median	1018
Mean	1018
3rd Qu.	1023
Max	1039
NAs	219
SD	6.62

Pressure3pm	
Min	988
1st Qu.	1012
Median	1016
Mean	1016
3rd Qu.	1021
Max	1038
NAs	210
SD	6.59

Cloud9am	
Min	0
1st Qu.	1
Median	4
Mean	4
3rd Qu.	7
Max	8
NAs	1007

Cloud3pm	
Min	0
1st Qu.	1
Median	5
Mean	4
3rd Qu.	7
Max	8
NAs	1053

Temp9am	
Min	1.5
1st Qu.	12.9
Median	17.1
Mean	17.4
3rd Qu.	21.7
Max	35.6
NAs	32

SD	3.10

SD	2.76

SD	5.69

Temp3pm	
Min	6.9
1st Qu.	17.1
Median	21.4
Mean	22.1
3rd Qu.	26.5
Max	42.9
NAs	86
SD	6.55

It is interesting to note that ‘Sunshine’ has the maximum number of NAs. In fact, 71.1% of values in ‘Sunshine’ are NAs. Following suite are variables ‘Evaporation’, ‘Cloud3pm’ and ‘Cloud9am’ with more than 50% of NA values. In the above, ‘MaxTemp’ has the least amount of NA values.
Considering these, we will have to omit rows of values that have missing data.

Pre-processing

Number of Missing Values for other variables	
Location	0
WarmerTomorrow	15
WindDir3pm	113
WindGustDir	232
WindDir9am	169
Total number of missing Values in all variables	6427

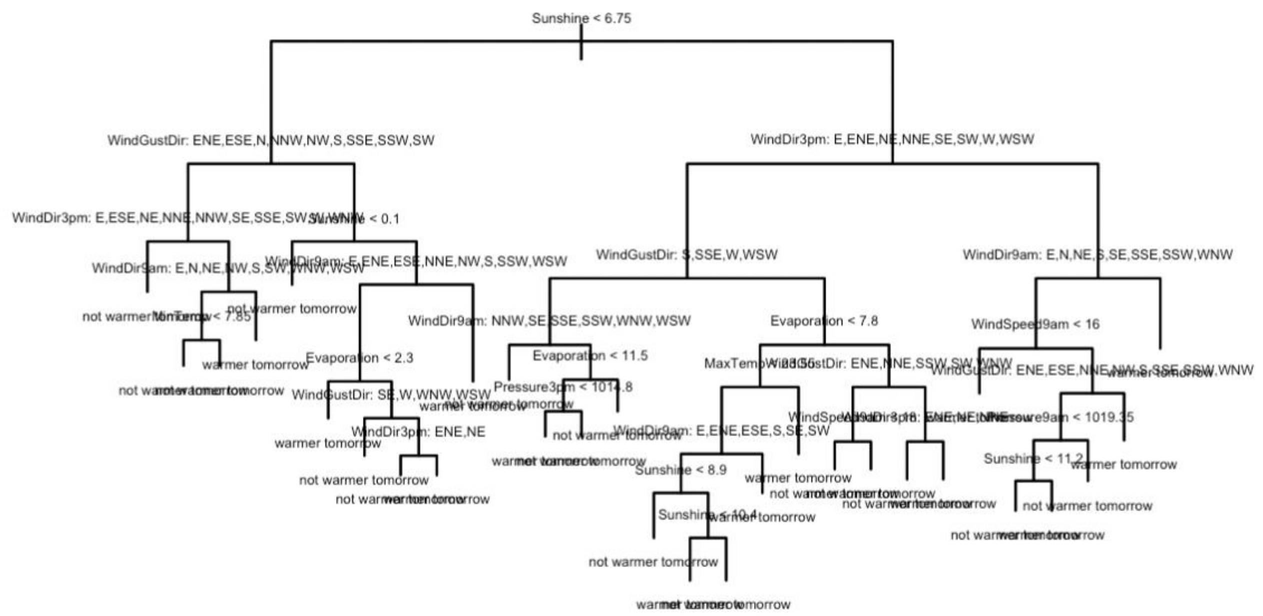
Considering that we are not doing a time related modelling, first, we drop ‘Day’, ‘Month’ and ‘Year’ columns from the data. After this, on calculation it seems that there are 6427 missing values in total.

We will omit all rows that have missing data. Other ways to deal with missing data are imputation of closest possible value or mean of other values.

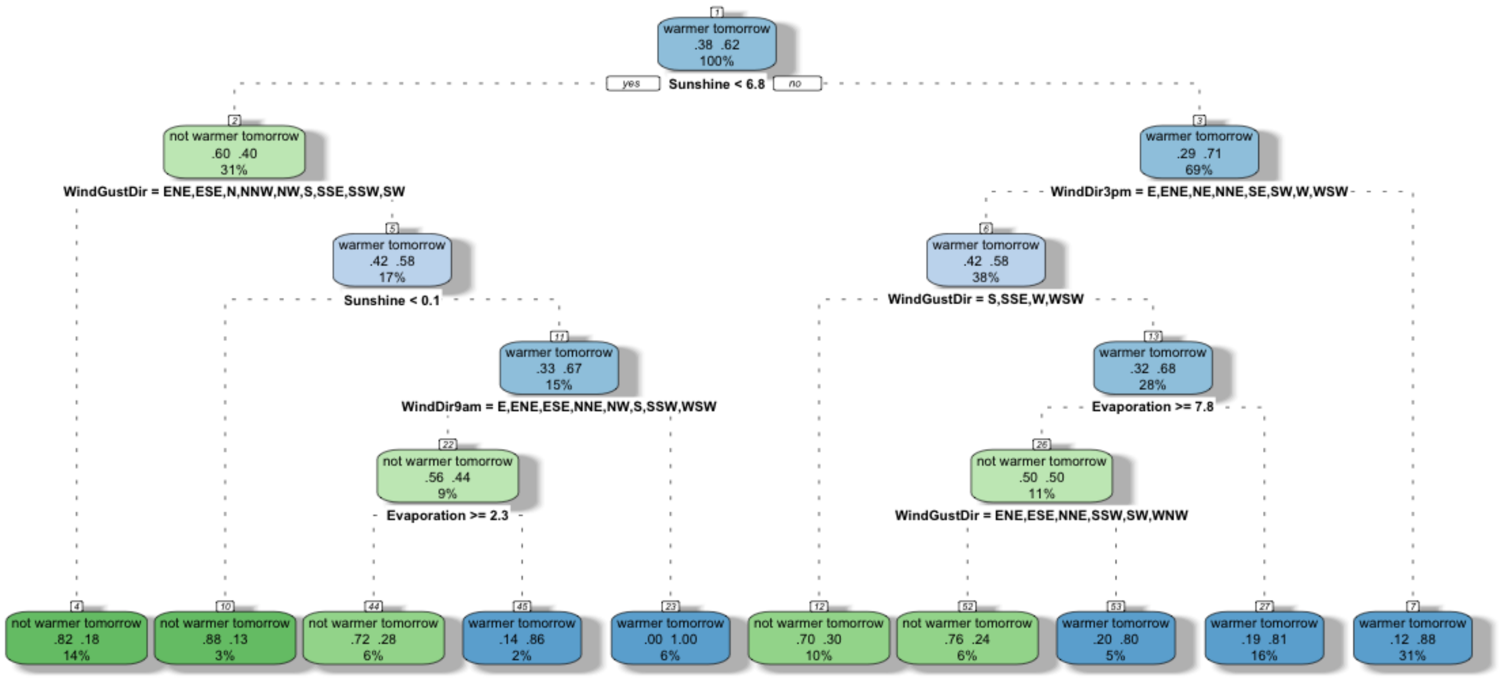
Upon dropping data that contained missing value, we have 410 rows of data remaining.

Model Fitting

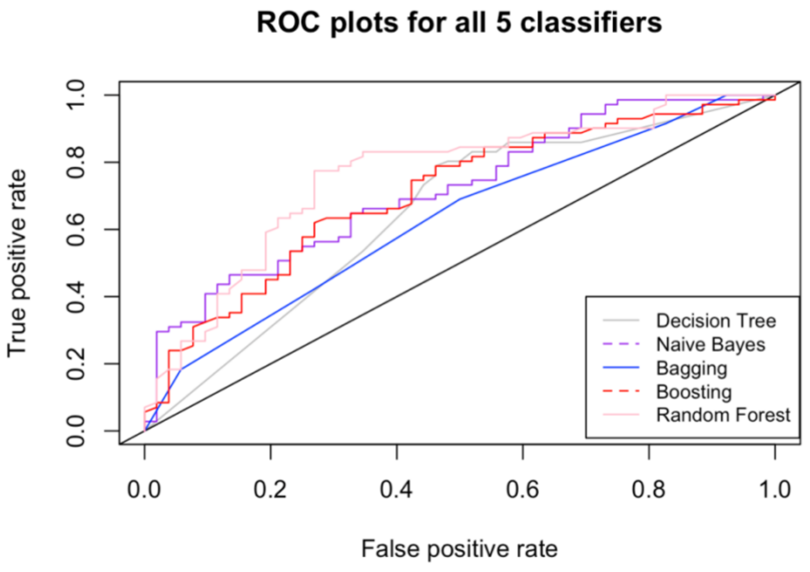
Decision Tree



Fancy Plotting for more readability



Other models fitted are: Naïve Bayes, Boosting, Bagging and Random Forest



Accuracy, ROC and Area Under Curve

To calculate the accuracy, a confusion matrix was drawn for each model. Using the True Positive and True Negative values as numerator and all the values in the denominator from the confusion matrix, accuracy was calculated.

Further, ROC plots were made for all models and using that, area under curve (AUC) was calculated.

	Decision Tree	**Naïve Bayes**	**Bagging**	**Boosting**	**Random Forest**
Accuracy	68.3%	64.2%	61%	67.5%	64.2%
AUC	65%	72%	63%	71%	76%

Drawing conclusion from this, Decision Tree has highest accuracy and Random Forest has highest area under curve. Next in line for highest accuracy and AUC is Boosting.
To draw a better conclusion, a further calculation of recall has been done. Recall is the proportions of actual values that were identified correctly. Here is the result:

	Decision Tree	Naïve Bayes	Bagging	Boosting	Random Forest
Recall	81.7%	66.2%	69%	81.7%	90.1%

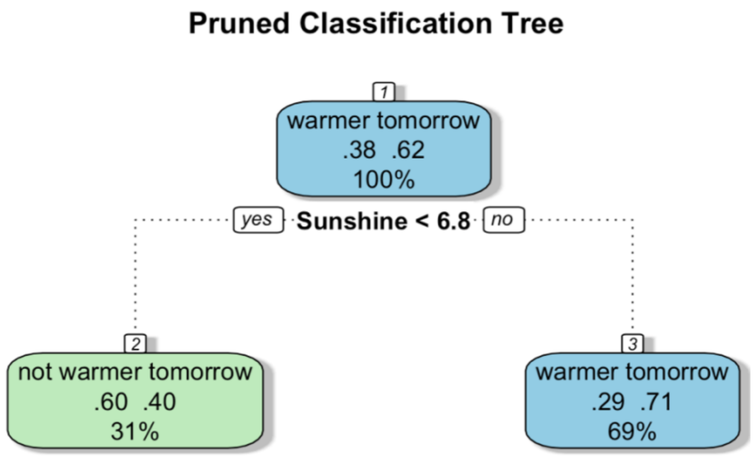
Here, Random Forest has the highest recall value. Therefore, based on accuracy, AUC and Recall, it is concluded that Random Forest is the best classifier.

Important Variables

Here is some of most and least important variables from the models Bagging, Boosting and Random Forest:

Most Important Variables		
Rank	**Bagging**	**Boosting**
1	WindGustDir	WindGustDir
2	WindDir3pm	WindDir3pm
3	WindDir9am	WindDir9am
Least Important Variables: 0.00 Importance		
Rank	**Bagging**	**Boosting**
15	Cloud9am	Cloud9am
16	Humidity9am	Humidity9am
17	Location	Location
18	MinTemp	MinTemp
19	Pressure9am	
20	Temp9am	

Based on the above, Cloud9am, Humidity9am, Location, MinTemp have 0.00 importance in both Bagging and Boosting models. Hence, it is safe to remove these variables.
In fact, on removing these variables and fitting a Bagging model, the accuracy of remained the same as 61% .

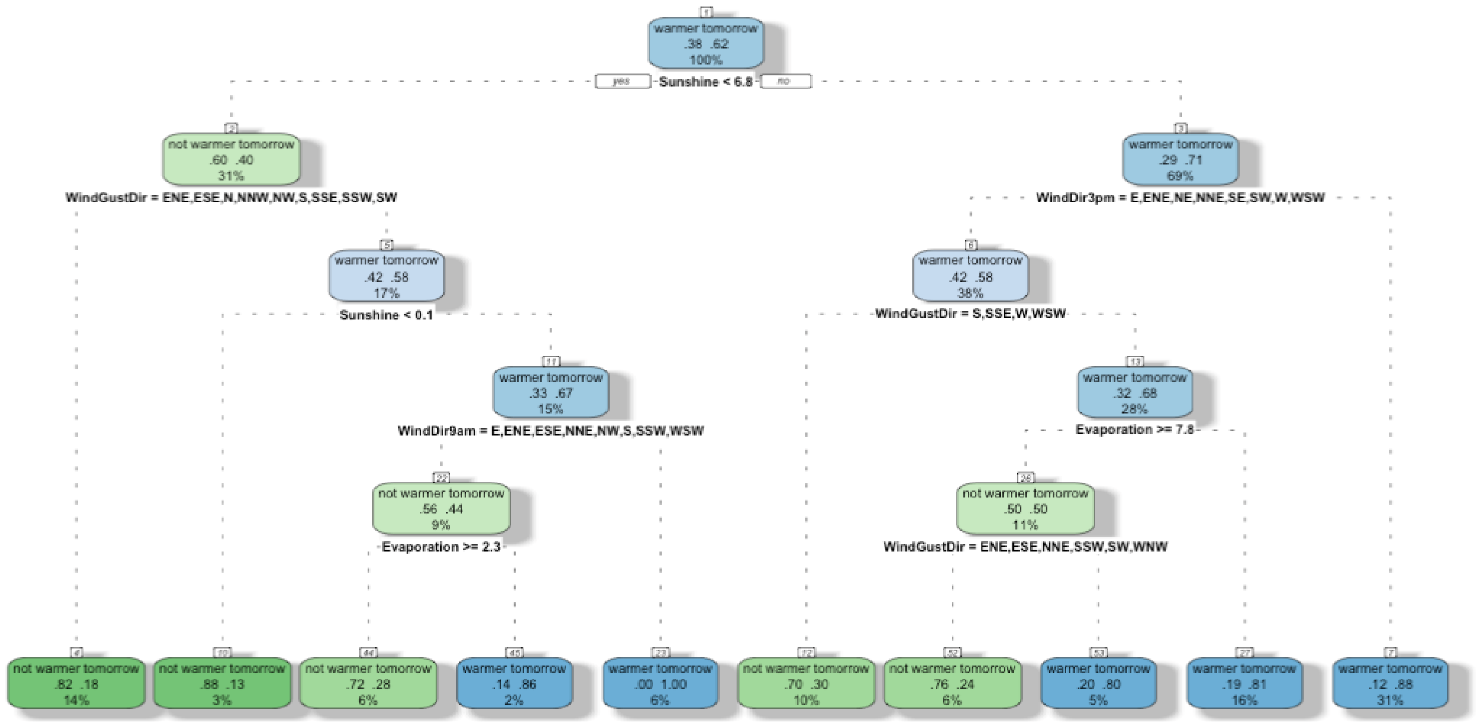


Simpler Classifier

Complexity parameter and cross validation were used to validate and then prune the model to simplify it. Pruning avoids overfitting, making a simpler and more readable tree. Using CP makes sure that there is least cross-validation error. The best CP value obtained for this is 0.0727, that is the least CP value, hence least error. Using this, a pruned tree is obtained. In fact, on testing the accuracy, it remained same as the unpruned trees, which is 68.3%.

The training and testing data used here was the new data set with the removed unimportant variables. Using CP value of least error gave a simpler tree with same accuracy as a complex tree. Removing unimportant variables removed the variables that were not contributing to the model.

Optimal Model

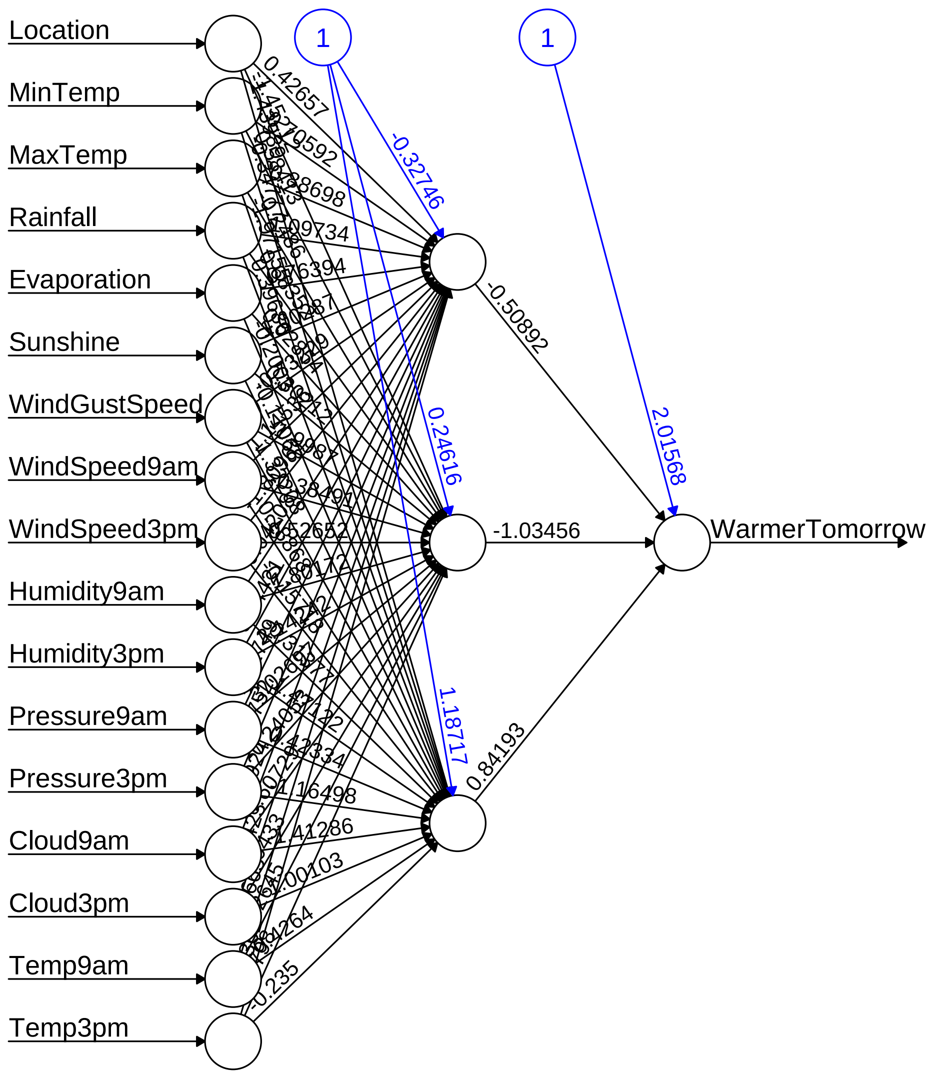


warmer tomorrow .38 .62 100%
yes
Sunshine < 6.8
no
not warmer tomorrow .60 .40 31%
WindGustDir = ENE,ESE,N,NNW,NW,S,SSE,SSW,SW
warmer tomorrow .29 .71 69%
WindDir3pm = E,ENE,NE,NNE,SE,SW,W,WSW
warmer tomorrow .42 .58 17%
Sunshine < 0.1
warmer tomorrow .42 .58 38%
WindGustDir = S,SSE,W,WSW
warmer tomorrow .33 .67 15%
WindDir9am = E,ENE,ESE,NNE,NW,S,SSW,WSW
warmer tomorrow .32 .68 28%
Evaporation >= 7.8
not warmer tomorrow .56 .44 9%
Evaporation >= 2.3
not warmer tomorrow .50 .50 11%
WindGustDir = ENE,ESE,NNE,SSW,SW,WNW
not warmer tomorrow .82 .18 14%
not warmer tomorrow .88 .13 3%
not warmer tomorrow .72 .28 6%
warmer tomorrow .14 .86 2%
warmer tomorrow .00 1.00 6%
not warmer tomorrow .70 .30 10%
not warmer tomorrow .76 .24 6%
warmer tomorrow .20 .80 5%
warmer tomorrow .19 .81 16%
warmer tomorrow .12 .88 31%

A recursive feature elimination was performed to make an optimal model. It is a backward selection process to eliminate features that are not relevant in prediction. Our original dataset has 21 features including “WarmerTomorrow”, which is the target variable. 10 fold cross validation is done, which performed fitting 10 time.
I chose, to improve on the Decision Tree model since it had the highest accuracy amongst the others.
Using recursive feature elimination on the data, the variables that were most relevant in predicting WarmerTomorrow were identified. This was then used to fit the Decision Tree model. The accuracy obtained was 90.6% which is very optimal compared to the previous models.

Artificial Neural Network

For data in neural network, firstly the prediction variable had to be changed to numeric from string. Hence, ‘warmer tomorrow’ was represented by 1 and ‘not warmer tomorrow’ was represented by 0. Secondly, all string data had to be removed since only numeric data can be used to perform ANN. "WindGustDir","WindDir9am", "WindDir3pm" were the String variables.
On performing ANN, an accuracy of 61.59% was achieved, which it highly optimized and high performing compared to other models. This ranks second to last, before Bagging.


Location
MinTemp
MaxTemp
Rainfall
Evaporation
Sunshine
WindGustSpeed
WindSpeed9am
WindSpeed3pm
Humidity9am
Humidity3pm
Pressure9am
Pressure3pm
Cloud9am
Cloud3pm
Temp9am
Temp3pm
1
1
0.42657
-0.32746
0.24616
1.18717
2.01568
-0.50892
-1.03456
0.84193
1.16498
1.41286
-0.235
WarmerTomorrow

Appendix [code]

```
# Reading data file and getting data according to sudent ID
rm(list = ls())
WAUS <- read.csv("WarmerTomorrow2022.csv",  stringsAsFactors = T)
L <- as.data.frame(c(1:49))
set.seed(29506751) # 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

##### Importing libraries used in the file #####
library(tidyverse)
library(dplyr)
library(adabag)
library(rpart)
library(neuralnet)
library(tree)
library(RColorBrewer)
library(rpart.plot)
library(rattle)
library(e1071)
library(randomForest)
library(caret)
library(ROCR)
############### 1 #####################

# Exploring data
# For number of days that are warmer tomorrow and not
# 1 represents 'warmer tomorrow'
# 0 represents 'not warmer tomorrow'
warm_cold <- table(WAUS$WarmerTomorrow)
num_of_days <- warm_cold[1] + warm_cold[2]
WAUS.warm_days <- warm_cold[2] # number of warmer days
WAUS.warm_days <- round((WAUS.warm_days/num_of_days)*100, digits = 2) #
percentage
WAUS.cooler_days <- warm_cold[1] # number of cooler days
WAUS.cooler_days <- round((WAUS.cooler_days/num_of_days)*100,  digits = 2) #
percentage
print(WAUS.warm_days)
print(WAUS.cooler_days)

# Predictors
# real-valued attributes: MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine,
# WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm,
# Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm
indices <- c(5:9,11,14:23)
options(digits=3)
summary(WAUS[,indices])
# Standard Deviation
```

```
sapply(WAUS[,indices], sd, na.rm=T)
####################################


############## 2 ####################
# Pre-processing data
# Removing Year, Month and day Day
drop <- c("Day","Month", "Year")
WAUS = WAUS[,!(names(WAUS) %in% drop)]
# Calculating number of missing values in total
missingValues <- (sapply(WAUS, function(x) sum(is.na(x))))
missingValues <- sum(missingValues)
# removing data that has NA values
WAUS <- WAUS %>% filter(complete.cases(.))
# Assiging 'Warmer Tomorrow' to 1 and 'Not warmer tomorrow' to 0
WarmerTomorrow = ifelse(WAUS$WarmerTomorrow == 1, "warmer tomorrow",
"not warmer tomorrow")
WarmerTomorrow = as.factor(WarmerTomorrow)
WAUS$WarmerTomorrow <- WarmerTomorrow
# Checking for any missing data in case
(sapply(WAUS, function(x) sum(is.na(x))))
####################################

############# 3 #####################
# dividing data into training and testing set
set.seed(29506751) #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,]
####################################

############## 4 ####################
# 4.1 Decision Tree
# fitting the model
DecisionTree = tree(WarmerTomorrow ~., data = WAUS.train)
summary(DecisionTree)
plot(DecisionTree)
text(DecisionTree, pretty = 0, cex = 0.35)

# Fancy plotting
fit<-rpart(WarmerTomorrow ~., data = WAUS.train, method = "class")
fancyRpartPlot(fit)

# 4.2 Naive-Bayers
set.seed(29506751)
NaiveBayes = naiveBayes(WarmerTomorrow ~., data = WAUS.train)

# 4.3 Bagging
# Bagging
set.seed(29506751)
```

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

# 4.4 Boosting
#Boosting
set.seed(29506751)
Boosting <- boosting(WarmerTomorrow ~., data = WAUS.train, mfinal=10)

# 4.5 Random Forest
# Random Forest
set.seed(29506751)
RandomForest <- randomForest(WarmerTomorrow ~., data = WAUS.train, na.action =
na.exclude)
####################################

############## 5 ####################
# 5 Confusion Matrix and accuracy
# Decision Tree
set.seed(29506751)
DecisionTree.predict = predict(DecisionTree, WAUS.test, type = "class")
DecisionTree.confusionM = table(Predicted_Class = DecisionTree.predict, Actual_Class =
WAUS.test$WarmerTomorrow)
cat("\n#Decision Tree Confusion\n")
print(DecisionTree.confusionM)
# Accuracy = (TP+TN)/(TP+TN+FN+FP)
DecisionTree.accuracy =
(DecisionTree.confusionM[1,1]+DecisionTree.confusionM[2,2])/(DecisionTree.confusionM[
1,1]+DecisionTree.confusionM[2,2]+DecisionTree.confusionM[1,2]+DecisionTree.confusio
nM[2,1])
DecisionTree.accuracy <- DecisionTree.accuracy*100

# Naive-Bayes
NaiveBayes.predict = predict(NaiveBayes, WAUS.test)
NaiveBayes.confusionM=table(Predicted_Class = NaiveBayes.predict, Actual_Class =
WAUS.test$WarmerTomorrow)
cat("\n#NaiveBayes Confusion\n")
print(NaiveBayes.confusionM)
# Accuracy = (TP+TN)/(TP+TN+FN+FP)
NaiveBayes.accuracy =
(NaiveBayes.confusionM[1,1]+NaiveBayes.confusionM[2,2])/(NaiveBayes.confusionM[1,1]
+NaiveBayes.confusionM[2,2]+NaiveBayes.confusionM[1,2]+NaiveBayes.confusionM[2,1])
NaiveBayes.accuracy <- NaiveBayes.accuracy*100

# Bagging
Bagging.predict <- predict.bagging(Bagging, WAUS.test)
Bagging.confusionM <- Bagging.predict$confusion
cat("\n#Bagging Confusion\n")
print(Bagging.confusionM)
# Accuracy = (TP+TN)/(TP+TN+FN+FP)
```

```
Bagging.accuracy = (Bagging.confusionM[1,1]+
Bagging.confusionM[2,2])/(Bagging.confusionM[1,1]+ Bagging.confusionM[2,2]+
Bagging.confusionM[1,2]+ Bagging.confusionM[2,1])
Bagging.accuracy <- Bagging.accuracy*100

# Boosting
Boosting.predict <- predict.boosting(Boosting, WAUS.test)
Boosting.confusionM <- Boosting.predict$confusion
cat("\n#Boosting Confusion\n")
print(Boosting.confusionM)
# Accuracy = (TP+TN)/(TP+TN+FN+FP)
Boosting.accuracy = (Boosting.confusionM[1,1]+
Boosting.confusionM[2,2])/(Boosting.confusionM[1,1]+ Boosting.confusionM[2,2]+
Boosting.confusionM[1,2]+ Boosting.confusionM[2,1])
Boosting.accuracy <- Boosting.accuracy*100

# Random forest
RandomForest.predict <- predict(RandomForest, WAUS.test)
RandomForest.confusionM = table(Predicted_Class = RandomForest.predict, Actual_Class =
WAUS.test$WarmerTomorrow)
cat("\n#Random Forest Confusion\n")
print(RandomForest.confusionM)

# Accuracy = (TP+TN)/(TP+TN+FN+FP)
RandomForest.accuracy = (RandomForest.confusionM[1,1]+
RandomForest.confusionM[2,2])/(RandomForest.confusionM[1,1]+RandomForest.confusion
M[2,2]+ RandomForest.confusionM[1,2]+ RandomForest.confusionM[2,1])
RandomForest.accuracy <- RandomForest.accuracy*100

# Accuracies
accuracy = data.frame(Decision_Tree = DecisionTree.accuracy, Naive_Bayes=
NaiveBayes.accuracy,
                  Bagging = Bagging.accuracy, Boosting = Boosting.accuracy,
                  RandomForest = RandomForest.accuracy)
print(accuracy)
####################################


############### 6 ###################
# Confidence and ROC

# Decision Tree
DecisionTree.confidence = predict(DecisionTree, WAUS.test, type = "vector")
# computing a simple ROC curve (x-axis: fpr, y-axis: tpr)
# labels are actual values, predictors are probability of class
detach("package:neuralnet", unload=TRUE)
DecisionTree.prediction <- prediction( DecisionTree.confidence[,2],
WAUS.test$WarmerTomorrow)
DecisionTree.performance <- performance(DecisionTree.prediction,"tpr","fpr")
```

```
plot(DecisionTree.performance, col = "gray")
abline(0,1)

# Naive-Bayes
NaiveBayes.confidence <- predict(NaiveBayes, WAUS.test, type = 'raw')
NaiveBayes.prediction <- prediction(NaiveBayes.confidence[,2],
WAUS.test$WarmerTomorrow)
NaiveBayes.performance <- performance(NaiveBayes.prediction,"tpr","fpr")
plot(NaiveBayes.performance, add=TRUE, col = "blueviolet")

# Bagging
Bagging.confidence <-  predict(Bagging, WAUS.test, type = 'raw')$prob
Bagging.prediction <- prediction(Bagging.confidence[,2], WAUS.test$WarmerTomorrow)
Bagging.performance <- performance(Bagging.prediction,"tpr","fpr")
plot(Bagging.performance, add=TRUE, col = "blue")

# Boosting
Boosting.confidence <-  predict(Boosting, WAUS.test, type = 'raw')$prob
Boosting.prediction <- prediction(Boosting.confidence[,2], WAUS.test$WarmerTomorrow)
Boosting.performance <- performance(Boosting.prediction,"tpr","fpr")
plot(Boosting.performance, add=TRUE, col = "red")

# Random Forest
RandomForest.confidence <- predict(RandomForest, WAUS.test, type="prob")
RandomForest.prediction <- prediction( RandomForest.confidence[,2],
WAUS.test$WarmerTomorrow)
RandomForest.performance <- performance(RandomForest.prediction,"tpr","fpr")
plot(RandomForest.performance, add=TRUE, col = "pink")

legend(0.7, 0.4, legend=c("Decision Tree", "Naive Bayes", "Bagging", "Boosting", "Random
Forest"),
      col=c("gray", "blueviolet","blue","red","pink"), lty=1:2, cex=0.8)
title("ROC plots for all 5 classifiers")

# AUC: Area Under Curve

# Decision Tree
DecisionTree.auc = performance(DecisionTree.prediction, "auc")
DecisionTree.auc = round((as.numeric(DecisionTree.auc@y.values)),2)
DecisionTree.auc = DecisionTree.auc*100

# Naive Bayes
NaiveBayes.auc = performance(NaiveBayes.prediction, "auc")
NaiveBayes.auc =  round((as.numeric(NaiveBayes.auc@y.values)),2)
NaiveBayes.auc = NaiveBayes.auc*100

# Bagging
Bagging.auc = performance(Bagging.prediction, "auc")
Bagging.auc =  round((as.numeric(Bagging.auc@y.values)),2)
Bagging.auc = Bagging.auc*100
```

```
# Boosting
Boosting.auc = performance(Boosting.prediction, "auc")
Boosting.auc =  round((as.numeric(Boosting.auc@y.values)),2)
Boosting.auc = Boosting.auc*100

# Random Forest
RandomForest.auc = performance(RandomForest.prediction, "auc")
RandomForest.auc =  round((as.numeric(RandomForest.auc@y.values)),2)
RandomForest.auc = RandomForest.auc*100
###############################


############ 7 ###################
# Comparing Results
# Accuracy
auc = data.frame(Decision_Tree = DecisionTree.auc, Naive_Bayes= NaiveBayes.auc,
                Bagging = Bagging.auc, Boosting = Boosting.auc,
                RandomForest = RandomForest.auc)
comparision_table <- rbind(accuracy, auc)
print(comparision_table)

## Recall: how good our model is at correctly predicting positive classes.

# Decision Tree
dt_numerator <-  DecisionTree.confusionM[2,2]
dt_denominator <- DecisionTree.confusionM[1,2] + DecisionTree.confusionM[2,2]
DecisionTree.recall <- round((dt_numerator/dt_denominator) * 100, 2)

# Naive Bayes
nb_numerator <-  NaiveBayes.confusionM[2,2]
nb_denominator <- NaiveBayes.confusionM[1,2] + NaiveBayes.confusionM[2,2]
NaiveBayes.recall <- round((nb_numerator/nb_denominator) * 100, 2)

# Bagging
bag_numerator <-  Bagging.confusionM[2,2]
bag_denominator <- Bagging.confusionM[1,2] + Bagging.confusionM[2,2]
Bagging.recall <- round((bag_numerator/bag_denominator) * 100, 2)

# Boosting
boost_numerator <-  Boosting.confusionM[2,2]
boost_denominator <- Boosting.confusionM[1,2] + Boosting.confusionM[2,2]
Boosting.recall <- round((boost_numerator/boost_denominator) * 100, 2)

# Random Forest
rf_numerator <-  RandomForest.confusionM[2,2]
rf_denominator <- RandomForest.confusionM[1,2] + RandomForest.confusionM[2,2]
RandomForest.recall <- round((rf_numerator/rf_denominator) * 100, 2)

recall = data.frame(Decision_Tree = DecisionTree.recall, Naive_Bayes= NaiveBayes.recall,
            Bagging = Bagging.recall, Boosting = Boosting.recall,
            RandomForest = RandomForest.recall)

comparision_table <- rbind(comparision_table, recall)
print(comparision_table)
################################

############ 8 ###################
# Variables
Bagging.variables <- sort(Bagging$importance)
Boosting.variables <- sort(Boosting$importance)
RandomForest.variables <- varImp(RandomForest)
variables = data.frame(Bagging = Bagging.variables, Boosting = Boosting.variables,
              RandomForest = RandomForest.variables)
print(cbind(Bagging.variables,Boosting.variables))
# Cloud9am, Humidity9am, Location, MinTemp are of 0.00 importance

# Plotting and checking accuracy of the model without variables that are unimportance
newWAUS <- WAUS
drop <- c( "Cloud9am", "Humidity9am", "Location", "MinTemp")
newWAUS = newWAUS[,!(names(newWAUS) %in% drop)]

# dividing data into training and testing set
set.seed(29506751) #Student ID as random seed
train.row = sample(1:nrow(newWAUS), 0.7*nrow(newWAUS))
newWAUS.train = newWAUS[train.row,]
newWAUS.test = newWAUS[-train.row,]

#Bagging
set.seed(29506751)
newBagging <- bagging(WarmerTomorrow ~., data = newWAUS.train,mfinal=5)

# Prediction and accuracy
newBagging.predict <- predict.bagging(newBagging, newWAUS.test)
newBagging.confusionM <- newBagging.predict$confusion
cat("\n#Bagging Confusion\n")
print(newBagging.confusionM)
# Accuracy = (TP+TN)/(TP+TN+FN+FP)
newBagging.accuracy = (newBagging.confusionM[1,1]+
newBagging.confusionM[2,2])/(newBagging.confusionM[1,1]+
newBagging.confusionM[2,2]+ newBagging.confusionM[1,2]+
newBagging.confusionM[2,1])
newBagging.accuracy <- newBagging.accuracy*100
newBagging.accuracy
# Same accuracy on dropping least important variables
################################


############# 9 ##################
# CP and Pruning to make Simpler Decision Tree
# getting value of least CP error
printcp(fit)
cp <- fit$cptable[which.min(fit$cptable[,"xerror"]),"CP"] # Optimal Cp value is 0.0727

# Pruning
ptree<- prune(fit, cp= cp,"CP")
tree(ptree)

# plotting
fancyRpartPlot(ptree, uniform=TRUE,main="Pruned Classification Tree")

# prediction and accuracy
set.seed(29506751)
simpler_tree <- tree(WarmerTomorrow~., newWAUS.train)
simpler_tree.predict = predict(simpler_tree, newWAUS.test, type = "class")
simpler_tree.confusionM = table(Predicted_Class = simpler_tree.predict, Actual_Class =
newWAUS.test$WarmerTomorrow)
cat("\n#Decision Tree Confusion\n")
print(simpler_tree.confusionM)

# Accuracy = (TP+TN)/(TP+TN+FN+FP)
simpler_tree.accuracy =
(simpler_tree.confusionM[1,1]+simpler_tree.confusionM[2,2])/(simpler_tree.confusionM[1,1
]+simpler_tree.confusionM[2,2]+simpler_tree.confusionM[1,2]+simpler_tree.confusionM[2,
1])
simpler_tree.accuracy <- simpler_tree.accuracy*100 # same accuracy as the complext
decision tree
################################


############# 10 ##################
# BETTER ACCURACY

# Automatic feature selection
set.seed(29506751)


###################### 10 ###################
# BETTER ACCURACY

# Automatic feature elemination
set.seed(29506751)
#automatic feature selection
control <- rfeControl(functions=rfFuncs, method="cv", number=10)
results <- rfe(WAUS.train[,1:20], WAUS.train[,21], rfeControl=control)

# Top Features
print(predictors(results))
cols=predictors(results)
```

```
cols=c(cols,"WarmerTomorrow")
index=which(colnames(WAUS.train) %in% cols)
WAUS_dt_train=WAUS.train[,index]
WAUS_dt_test=WAUS.train[,index]

# Fitting Decision tree
fit_tree_mod = tree(WarmerTomorrow ~., data = WAUS_dt_train, method = "class")
plot(fit_tree_mod)
text(fit_tree_mod, pretty = 0, cex=0.5)
# Fancy plotting
fit<-rpart(WarmerTomorrow ~., data = WAUS_dt_train, method = "class")
fancyRpartPlot(fit)

# Prediction and accuracy
set.seed(29506751)
fit_tree_mod.predict = predict(fit_tree_mod, WAUS_dt_test, type = "class")
fit_tree_mod.confusionM = table(Predicted_Class = fit_tree_mod.predict, Actual_Class =
WAUS_dt_test$WarmerTomorrow)
cat("\n#Decision Tree Confusion\n")
print(fit_tree_mod.confusionM)
# Accuracy = (TP+TN)/(TP+TN+FN+FP)
fit_tree_mod.accuracy =
(fit_tree_mod.confusionM[1,1]+fit_tree_mod.confusionM[2,2])/(fit_tree_mod.confusionM[1,
1]+fit_tree_mod.confusionM[2,2]+fit_tree_mod.confusionM[1,2]+fit_tree_mod.confusionM[
2,1])
fit_tree_mod.accuracy <- fit_tree_mod.accuracy*100
fit_tree_mod.accuracy
##################


########## 11 ##########
ANN_WAUS <- WAUS
ANN_WAUS <- ANN_WAUS %>% mutate(WarmerTomorrow=recode(WarmerTomorrow,
                    'warmer tomorrow'= 1,
                    'not warmer tomorrow'=0))
ANN_WAUS$WarmerTomorrow = as.numeric(ANN_WAUS$WarmerTomorrow)

# dropping string
drop <- c("WindGustDir","WindDir9am", "WindDir3pm")
ANN_WAUS = ANN_WAUS[,!(names(ANN_WAUS) %in% drop)]

# make training and test sets
set.seed(29506751)
train.row = sample(1:nrow(ANN_WAUS), 0.8*nrow(ANN_WAUS))
ANN_WAUS.train = ANN_WAUS[train.row,]
ANN_WAUS.test = ANN_WAUS[-train.row,]

library(neuralnet)
ANN_WAUS.nn = neuralnet(WarmerTomorrow~., ANN_WAUS.train, hidden=(3),act.fct =
"logistic",linear.output = FALSE)
```

```
plot(ANN_WAUS.nn)
ANN.predict = predict(ANN_WAUS.nn, ANN_WAUS.test, type = "class")
ANN.confusionM = table(Predicted_Class = ANN.predict, Actual_Class =
as.array(ANN_WAUS.test$WarmerTomorrow))
print(ANN.confusionM)
```