FIT3152 Assignment 2

Name: Ko Ko Win

ID: 31842305

Question 1)

Q: ***What is the proportion of days when it is warmer than the previous day compared to those where it is cooler?***

A: According to the observation of data proportion of the days when it is warmer than the previous day is 52.43%. On the other hand, proportion of days where it is cooler than previous day is 47.57%. We can see that proportion of warmer days are higher than cooler days.

Q: ***Is there anything noteworthy in the data? Are there any attributes you need to consider omitting from your analysis?***

```
-- Data Summary ------------------------
                           Values
Name                       df_stat_sum
Number of rows             2000
Number of columns          16
_____________________
Column type frequency:
  numeric                  16
_____________________
Group variables            None

-- Variable type: numeric ---------------------------------------------------------------------------------------------------------------
   skim_variable n_missing complete_rate    mean    sd    p0    p25    p50    p75   p100 hist
 1 MinTemp              25         0.988   12.9   6.17  -3.9    8.5   12.4   16.9   30   <U+2581><U+2586><U+2587><U+2583><U+2582>
 2 MaxTemp              14         0.993   23.5   7.21   8.4   18.2   22.1   29.4   43.8 <U+2582><U+2587><U+2585><U+2585><U+2581>
 3 Rainfall             69         0.966    2.26  6.64   0     0      0      1     92.6 <U+2587><U+2581><U+2581><U+2581><U+2581>
 4 Evaporation        1038         0.481    5.40  3.67   0.2   3      4.8    7.2   43.6 <U+2587><U+2581><U+2581><U+2581><U+2581>
 5 Sunshine            854         0.573    8.19  3.67   0     5.6    9.1   11     14   <U+2582><U+2583><U+2583><U+2587><U+2585>
 6 WindGustSpeed        53         0.974   41.8  12.5   13    33     41     50     91   <U+2582><U+2587><U+2583><U+2581><U+2581>
 7 WindSpeed9am         22         0.989   16.2   8.42   0     9     15     22     48   <U+2585><U+2587><U+2585><U+2582><U+2581>
 8 WindSpeed3pm         22         0.989   19.7   8.01   0    13     19     24     56   <U+2582><U+2587><U+2583><U+2581><U+2581>
 9 Humidity9am          32         0.984   68.1  19.2    5    55.8   69     83    100   <U+2581><U+2582><U+2586><U+2587><U+2586>
10 Humidity3pm          33         0.984   51.7  21.1    4    34     53     66    100   <U+2583><U+2586><U+2587><U+2586><U+2582>
11 Pressure9am          23         0.988 1018.    7.07 990.  1013.  1018.  1023.  1039.  <U+2581><U+2582><U+2587><U+2586><U+2581>
12 Pressure3pm          22         0.989 1015.    7.02 986.  1010.  1016.  1020.  1038.  <U+2581><U+2582><U+2587><U+2586><U+2581>
13 Cloud9am            797         0.601    4.60  2.91   0     1      5      7      8    <U+2585><U+2582><U+2581><U+2583><U+2587>
14 Cloud3pm            836         0.582    4.67  2.79   0     2      5      7      8    <U+2585><U+2583><U+2581><U+2583><U+2587>
15 Temp9am              23         0.988   17.5   6.24  -0.5   12.8   17.1   22     34.7 <U+2581><U+2586><U+2587><U+2585><U+2582>
16 Temp3pm              21         0.990   22.0   7.07   4.7   16.8   20.6   27.7   42.1 <U+2581><U+2587><U+2586><U+2585><U+2581>
```

A:

To check the summary description of the predictors I have disregarded some columns which is not needed such as Day, Month and Year. Moreover, I have used skimr library to check the summary statistics rather than the built in summary function .From the table obtained above some of the notable things are

1) The data has 3884 missing values with variables Evaporation, Sunshine, Cloud9am and Cloud3am having large proportion of missing values.
2) Mean value of Pressure9am and Pressure3pm are significantly high.
3) Variable Rainfall median value is higher than the mean value which indicates that the variable is right skewed.

Question 2)

Q**: *Document any pre-processing required to make the data set suitable for the model fitting that follows***

A: The data has 3884 missing values, so I have omitted all the rows with NA values and I have changed the class of the target variable WarmerTomorrow to a factor for the model fitting in upcoming questions. This question was already done before proceeding to Q1.

Question 3)

Q: Divide your data into a 70% training and 30% test set

A: The data was divided into 2 parts where 70% will be used to train the model and 30% of the data will be used to test the model.

Question 4)

Classification model used to analyse are Decision Tree, Naïve Bayes, Bagging, Boosting and Random Forest

Question 5)

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

A:

Classifiers	**Decision Tree**	**Naïve Bayes**	**Bagging**	**Boosting**	**Random Forest**
Accuracy (%)	55.31%	62.57%	58.1%	56.42%	65.36%

To calculate the accuracy of each classification model I have used the test data and each of test cases “warmer tomorrow” and “not warmer tomorrow”. Upon creating the confusion matrix for each model I have calculated the accuracy using the formula $\frac{TP + TN}{TP + TN + FP + FN}$.

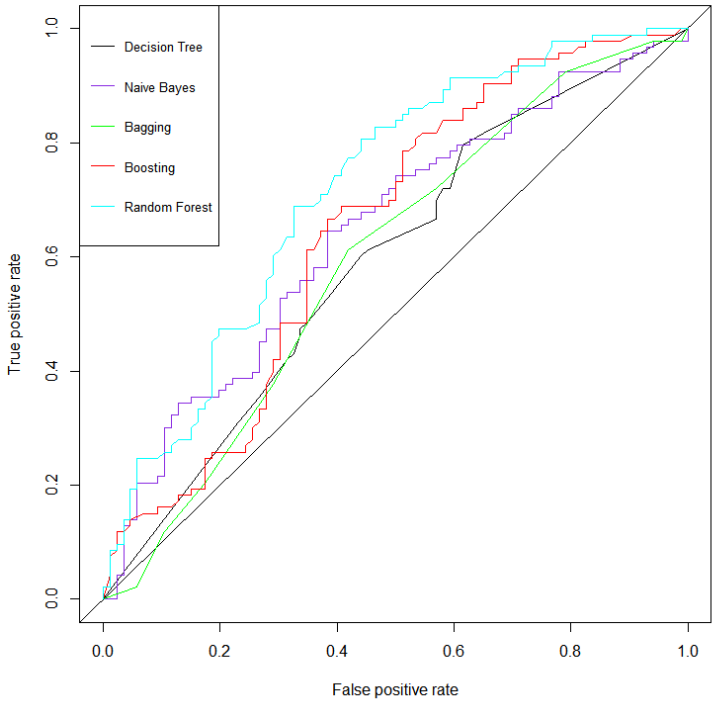
As we can see from the table above Random Forest has the highest accuracy and predicting whether it will be warm tomorrow or not. On the other hand, Decision Tree has the lowest accuracy in predicting among all the classifiers.

Question 6)

Q: Using the test data construct the ROC curve for all classifiers and calculate the AUC for each classifier.

A:

ROC curve for all the classifier:



AUC values for each classifier:

Classifiers	**Decision Tree**	**Naïve Bayes**	**Bagging**	**Boosting**	**Random Forest**
AUC	0.593	0.645	0.596	0.611	0.714

The AUC value of a classifier tells us the probability that the classifier will rank randomly chosen positive instance higher than a randomly chosen negative instance. As we can see from the table above Random Forest has the highest AUC value.

Question 7)

Q: Create a table comparing the results in parts 5 and 6 for all classifiers. Is there a single "best" classifier?

A:

Classifiers	**Decision Tree**	**Naïve Bayes**	**Bagging**	**Boosting**	**Random Forest**
Accuracy (%)	55.31%	62.57%	58.1%	56.42%	65.36%
AUC	0.593	0.645	0.596	0.611	0.714

As we can see from the table above Random Forest is the best classifier since it has the highest accuracy in correctly classifying the data and highest AUC value among all the classifiers. By further investigating the ROC plot, the curve for Random Forest is closer to the point (0,1) which indicates that Random Forest classifier is more accurate compared to all other classifiers. Thus, we can conclude that Random Forest is the single best classifier.

Question 8)

Q: Examining each of the models, determine the most important variables in predicting whether it will be warmer tomorrow or not. Which variables could be omitted from the data with very little effect on performance?

A: The important predictors for each model is shown below.

Decision Tree

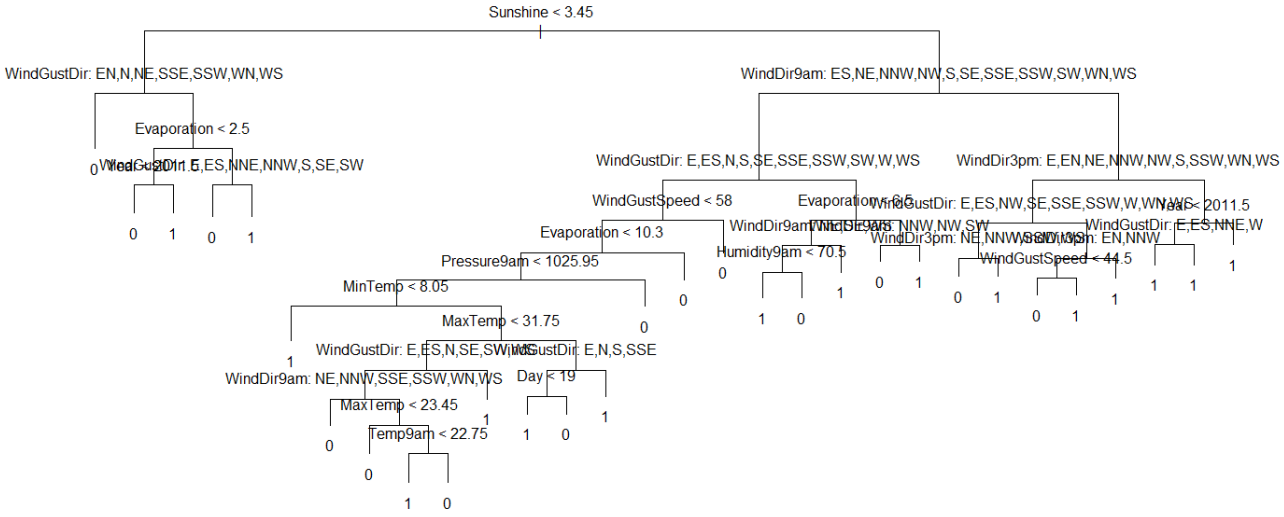
The important attributes used for the decision tree are:

Sunshine ,WindGustDir , Evaporation , Year , WindDir9am ,

WindGustSpeed , Pressure9am , MinTemp , MaxTemp , Temp9am ,

Day , Humidity9am , WindDir3pm.

To determine the most important predictors we will plot the decision tree



By looking at the tree plot we can see that Sunshine, WindGustDir, WindDir9am and WindDir3pm are the most important attribute.

Naïve Bayes

None. Because Naïve Bayes is not a tree-based classifier.

Bagging

	Var1	Freq
19	WindGustDir	18.9724974
18	WindDir9am	17.5297874
17	WindDir3pm	15.0849855
14	Sunshine	7.9038251
9	MinTemp	5.3395496
12	Pressure9am	4.6146474
2	Cloud9am	3.6205020
5	Humidity3pm	3.1125950
3	Day	3.0610921
11	Pressure3pm	2.9061510
8	MaxTemp	2.8863290
4	Evaporation	2.7357450
7	Location	1.8898039
20	WindGustSpeed	1.6802256
22	WindSpeed9am	1.6646505
1	Cloud3pm	1.5386367
10	Month	1.4818594
13	Rainfall	1.3389590
15	Temp3pm	1.3054370
21	WindSpeed3pm	0.5905171
6	Humidity9am	0.4399028
23	Year	0.3023014
16	Temp9am	0.0000000

For the bagging classifier the most important predictors are WindGustDir, WindDir9am, WindDir3pm and Sunshine.

Boosting

For the boosting classifier the most important predictors are WindDir3pm, WindGustDir, WindDir9am and Evaporation.

	Var1	Freq
17	WindDir3pm	20.1715723
19	WindGustDir	16.2797887
18	WindDir9am	16.1673935
4	Evaporation	8.2180838
14	Sunshine	3.9194609
3	Day	3.8092075
15	Temp3pm	3.5113832
8	MaxTemp	3.4991207
12	Pressure9am	3.3155574
5	Humidity3pm	3.1863950
9	MinTemp	3.1749068
21	WindSpeed3pm	2.1442181
23	Year	1.9311808
2	Cloud9am	1.9185930
6	Humidity9am	1.7804009
22	WindSpeed9am	1.7323779
10	Month	1.5530875
1	Cloud3pm	1.3972586
11	Pressure3pm	0.9552224
7	Location	0.7736263
16	Temp9am	0.5611646
13	Rainfall	0.0000000
20	WindGustSpeed	0.0000000

Random Forest

	Var1	Var2	Freq
12	WindDir9am	MeanDecreaseGini	22.434205
10	WindGustDir	MeanDecreaseGini	21.850219
13	WindDir3pm	MeanDecreaseGini	18.937964
9	Sunshine	MeanDecreaseGini	12.463925
23	Temp3pm	MeanDecreaseGini	10.930432
8	Evaporation	MeanDecreaseGini	10.786460
6	MaxTemp	MeanDecreaseGini	10.641345
5	MinTemp	MeanDecreaseGini	9.890025
18	Pressure9am	MeanDecreaseGini	8.563380
22	Temp9am	MeanDecreaseGini	8.058785
17	Humidity3pm	MeanDecreaseGini	7.902341
19	Pressure3pm	MeanDecreaseGini	7.482126
16	Humidity9am	MeanDecreaseGini	7.322248
1	Day	MeanDecreaseGini	6.612042
21	Cloud3pm	MeanDecreaseGini	6.018536
14	WindSpeed9am	MeanDecreaseGini	5.587849
11	WindGustSpeed	MeanDecreaseGini	5.509839
20	Cloud9am	MeanDecreaseGini	5.460732
15	WindSpeed3pm	MeanDecreaseGini	5.448210
3	Year	MeanDecreaseGini	4.755087
2	Month	MeanDecreaseGini	4.733678
7	Rainfall	MeanDecreaseGini	3.730486
4	Location	MeanDecreaseGini	1.837751

For the Random Forest classifier WindDir9am, WindGustDir, WindDir3pm and Sunshine are the most important predictors.

By looking at the result of all classifiers we can conclude that WindDir9am (Direction of the wind at 9am), WindDir3pm (Direction of the wind at 3pm), WindGustDir (Direction of strongest wind gust over the day) , Sunshine (Hours of bright sunshine over the day) and Evaporation (The evaporation (mm) in the 24 hours to 9am.) are the most important predictors for our classifiers and it should not be omitted when fitting the model.

On the other hand, there are variable which are not much significant to our model. For instance, variable **Rainfall** can be omitted from our models since it is not contributed in Decision Tree and importance value of 0 in Boosting. Furthermore, as we can see from our question 1 variable **Rainfall** is right skewed which shows that it contains multiple outliers in the data .Moreover, it has second least importance value in Random Forest while it ranked 18th among 23 variables in Bagging in term of importance value. We will be omitting variable **Rainfall** from our 4 models above to examine the accuracy.

Classifiers	Decision Tree	Boosting	Bagging	Random Forest
Accuracy without Rainfall	55.31%	56.42%	60.89%	66.48%
Accuracy with Rainfall	55.31%	56.42%	58.10%	65.36%

As we can see from the table above the accuracy for Decision Tree stay unchanged because in the initial model variable “Rainfall” was not used and accuracy for Boosting remains unchanged as well since variable “Rainfall” has 0 importance. However, accuracy for Bagging increased by 2.79% and accuracy for Random Forest increased by 1.12% by omitting variable “Rainfall” .

Thus, we can conclude that by removing the predictor that has 0 importance or predictor that does not contribute to the model will not increase nor decrease the model accuracy. Furthermore, by removing the predictor that falls into top 5 lowest importance value will slightly increase the model accuracy.

Question 9)

Q: Starting with one of the classifiers you created in Part 4, create a classifier that is simple enough for a person to be able to classify whether it will be warmer tomorrow or not by hand

A:

Decision Tree classifier was chosen because it is simple enough for a person to work through the diagram for the prediction. To made the tree simpler for a person to classify by hand we will use pruning approach to prune the subtrees in our decision tree in the bottom-up approach. Before we begin to prune the tree we modelled in question 4, I have used cross validation approach to prune the misclassification which is done using cv.tree function in R. The below is the output of cross validation.

```
> test.fit = cv.tree(tree.fit, FUN = prune.misclass)
> print(test.fit)
$size
 [1] 30 28 26 24 22 17 12  4  3  2  1

$dev
 [1] 167 168 168 168 168 168 171 166 172 185 221

$k
 [1] -Inf  0.0  0.5  1.0  1.5  2.0  3.0  4.0 17.0 28.0 36.0

$method
[1] "misclass"

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

This diagram tells us that:

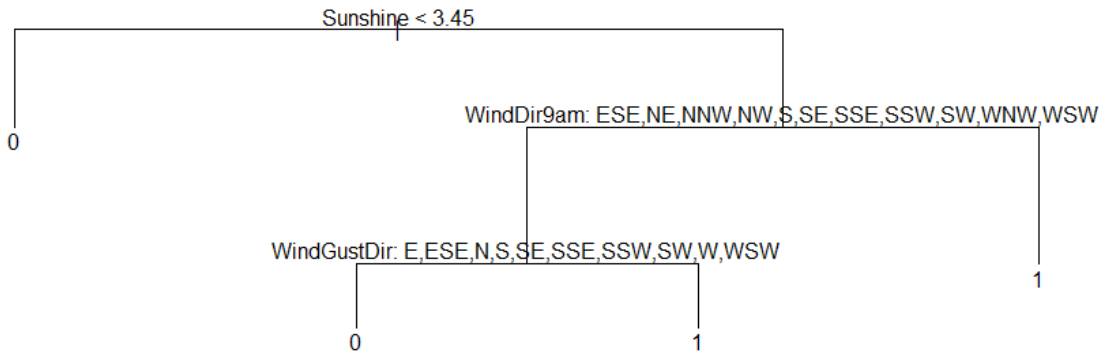
1) Size of the trees are 30, 28, 26 and etc.
2) Tree size of 4 has the lowest misclassification amount out of all the tree sizes which indicates tree size of 4 will give the best accuracy.

```
270 #prune down the tree size
271 prune.tree.fit = prune.misclass(tree.fit, best = 4)
272 summary(prune.tree.fit)
```

The above R function was used to pruned the decision tree. “best =4” indicates 4 terminal nodes is used because the result of cross validation tells us that tree size of 4 gives us the lowest misclassification value. This is crucial as it will gives us the best accuracy and AUC value.

```
Classification tree:
snip.tree(tree = tree.fit, nodes = c(7L, 2L, 12L, 13L))
Variables actually used in tree construction:
[1] "Sunshine"    "WindDir9am"  "WindGustDir"
Number of terminal nodes:  4
Residual mean deviance:  1.224 = 503.1 / 411
Misclassification error rate: 0.3012 = 125 / 415
```

The summary of pruned tree tells us only 3 attributes are used in constructing the prune decision tree which are Sunshine, WindDir9am and WindGustDir. These attributes are also shown to be important attributes in our original decision tree.



After all the steps the model was generated as above. Furthermore, after obtaining the decision tree I have calculated the accuracy of the model and AUC of the model to compare with the initial decision tree we modelled. The results are shown as below:

Classifiers	**Accuracy (%)**	**AUC**
Original Decision Tree	55.31%	0.593
Pruned Decision Tree	59.20%	0.624

Compared to the decision tree model created in part 4 out pruned decision tree has greater accuracy in classifying the data correctly. By pruning the decision tree the accuracy has increased by 3.89%. On the other hand, AUC values tell us how many time the model rank the random positive example more than the random negative example. By looking at the AUC value of our pruned decision tree we can see that it has increased by 0.031 which is not very significant. To conclude, we can say prune decision tree performs better than our original decision tree from part 4.

Question 10)

Q: Create the best tree-based classifier you can

A:

Random Forest classifier was chosen to improve from one of the classifier in part 4. It is because Random Forest turns out to be the best performing model with highest accuracy and AUC value. Moreover, while overfitting is the major issue in Machine Learning models, Random Forest does not have over fitting issue.

To optimise our current Random Forest model from part 4 I have created an algorithm to fit model with different number of trees varying from 501 to 700.

```
#Algorithm to fit the number of trees from 501 to 700 by tuning different mtry value
for(ntrees in 500:700){
  set.seed(31842305)
  new_rf.fit = randomForest(WarmerTomorrow ~., data = imp.train, importance = TRUE, ntree = ntrees, mtry=2)
  new_rf.pred = predict(new_rf.fit, imp.test)
  new_rf.cfm = table(actual = imp.test$WarmerTomorrow, predicted = new_rf.pred)
  new_rf.acc = round(mean(new_rf.pred == imp.test$WarmerTomorrow)*100, digits = 2)

  #calculate confidence and AUC of the new Random Forest model
  new_rf.conf = predict(new_rf.fit, imp.test, type = "prob")
  new_rf.conf.pred = prediction(new_rf.conf[,2], imp.test$WarmerTomorrow)
  new_rf.auc = performance(new_rf.conf.pred, "auc")
  new_rf.auc = as.numeric(new_rf.auc@y.values)


  cat("Number of tree used is", new_rf.fit$ntree, "accuracy is: ", new_rf.acc , "AUC value: ", new_rf.auc ,"\n")

}
```

From the above diagram I have:

- Set the “Importance = TRUE”. Random Forest and other ensemble methods allow us to check variable importance. Thus, importance was set to TRUE to remove variables with low importance value.
- I have varied the mtry (number of variables randomly sampled as candidates at each split) from 2 to 10 manually and observed that value of 2 gives the best accuracy and AUC value.
- Lastly I have calculated the accuracy and AUC of our improved model. The results are shown in the table below.
- Cross Validation approach was not used because it has high computation cost to re-run the algorithm from scratch k-times

Classifiers	Accuracy (%)	AUC	Number of Trees	Number of Split
Original Random Forest	65.36 %	0.714	500	4
Improved Random Forest	68.16 %	0.720	537	2

By looking at the table above we can see that by increasing the number of trees from **500 to 537** and decreasing the number of splits from **4 to 2.** There is an increase of **2.8%** in accuracy and slight increase in AUC result. In conclusion we can see that tuning the number of splits, setting the importance to TRUE and increasing in number of trees gives us higher accuracy at the cost of slower learning.

Question 11)

Q: Using the insights from your analysis so far, implement an Artificial Neural Network classifier and report its performance

A:

Neural network consists of neurons which are later turns into multiple layers where the output from each layer is passed onto another layer and so on. Before we fit the data into the neural network it is necessary to carry out data pre-processing as it has some specific constraint. For this question some of the pre-processing I have done are:

- Used one hot encoding to convert non-numeric data such as WindDir9am, WinDir3pm and WinGustDir to numeric data type.
- Performed min-max normalisation technique to both training and testing data to adjust the data to the common scale as it will ensure in high accuracy while predicting.

Once data pre-processing was completed I have fitted the data using neuralnet package. Some of the attributes used in the function were:

- “hidden = 3” where the number of hidden layers was 3
- “linear.output” = FALSE to let the neural network know we are doing classification rather than regression
- “threshold = 0.01” to let the model know if the change in error during the iteration is less than 1% there is no need to further optimize.

After the model was obtained I generated the confusion matrix using the testing data set and accuracy of the Neural Network turns out to be **53.1%.** The accuracy of the neural network classifier is the lowest among all the other classifiers. Some of the reasons of the low accuracy rate might be as below.

1. Compare to other machine learning algorithms neural network classifiers requires millions of labelled samples. While we only have few thousands of data algorithms such as Random Forest or Naïve Bayes would be a better alternative compared to neural network.
2. Artificial Neural Network is mostly used in duplicating the performance of living organisms, recognizing speech and predicting handwritten digits. It is used to understand the complex relationships between the input and output. Thus, neural network might not be suitable in predicting if tomorrow will be warm or not.

To conclude everything that has been stated, artificial neural network classifier is the worst performing classifier among all other classifiers.

APPENDIX – R Code

```
#load libraries
library(pastecs)
library(tree)
library(rpart)
library(e1071)
library(adabag)
library(randomForest)
library(ROCR)
library(sf)
library(skimr)


rm(list = ls())
WAUS <- read.csv("WarmerTomorrow2022.csv", stringsAsFactors = T)
L <- as.data.frame(c(1:49))
set.seed(31842305) # 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

#Omit rows which are not needed for statistic description
df_stat_sum <-  subset(WAUS, select = -c(1,2,3,4,10,12,13,24))
skim(df_stat_sum)



#Omit rows with NA and set target variable to factor
WAUS = na.omit(WAUS)
WAUS$WarmerTomorrow = as.factor(WAUS$WarmerTomorrow)


###############Question 1##################

warm_tmr = as.data.frame(table(WAUS$WarmerTomorrow))

#calculate the proportion of warmer and cooler
warmer =  round((warm_tmr[2,2] / sum(warm_tmr[,2])) * 100, digits = 2)
#52.43
cooler = round((warm_tmr[1,2] / sum(warm_tmr[,2])) * 100, digits = 2) #
47.57

###############Question 2##################

#This part was done above already


###############Question 3##################

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

set.seed(31842305)
train.row = sample(1:nrow(WAUS), 0.7*nrow(WAUS))
train.data = WAUS[train.row,]
test.data = WAUS[-train.row,]
```

```
###############Question 4###################

#### Question: Implement a classification model using each of the following
techniques

#Decision Tree
tree.fit = tree(WarmerTomorrow ~., data = train.data, method = "class")

#Naïve Bayes
naive.fit = naiveBayes(WarmerTomorrow ~., data = train.data)

##Bagging
wbag.fit = bagging(WarmerTomorrow ~., train.data, mfinal = 10)

#Boosting
wboost.fit = boosting(WarmerTomorrow ~., train.data, mfinal = 10)

#Random Forest
rf.fit = randomForest(WarmerTomorrow ~., train.data)


###############Question 5###################

#Question: Using the test data, classify each of the test cases as 'warmer
tomorrow' or 'not warmer tomorrow'. Create a confusion matrix and report
the accuracy of each model.

#Decision Tree
tree.pred = predict(tree.fit, test.data, type = "class")
#confusion matrix
tree.cfm = table(actual = test.data$WarmerTomorrow, predicted = tree.pred)
#accuracy of model
tree.acc = round(mean(tree.pred == test.data$WarmerTomorrow)*100, digits =
2)
cat("Decision Tree accuracy is: ", tree.acc, "%")



#Naïve Bayes
naive.pred = predict(naive.fit, test.data)
#confusion matrix
naive.cfm = table(actual = test.data$WarmerTomorrow, predicted =
naive.pred)
#accuracy of model
naive.acc = round(mean(naive.pred == test.data$WarmerTomorrow)*100, digits
= 2)
cat("Naïve Bayes model accuracy is: ", naive.acc, "%")



##Bagging
wbag.fit_pred = predict.bagging(wbag.fit, test.data)
#confusion matrix
wbag.fit.cfm = wbag.fit_pred$confusion
#accuracy of model
wbag.fit.acc = round(mean(wbag.fit_pred$class ==
test.data$WarmerTomorrow)*100, digits = 2)
```

```
cat("Bagging ensemble model accuracy is: ", wbag.fit.acc, "%")



#Boosting
wboost.fit_pred = predict.boosting(wboost.fit, test.data)
#confusion matrix
wboost.fit.cfm = wboost.fit_pred$confusion
#accuracy of model
wboost.fit.acc = round(mean(wboost.fit_pred$class ==
test.data$WarmerTomorrow)*100, digits = 2)
cat("Boosting ensemble model accuracy is: ", wboost.fit.acc, "%")


#Random Forest
rf.fit_pred = predict(rf.fit, test.data)
#confusion matrix
rf.fit.cfm = table(actual = test.data$WarmerTomorrow, predicted =
rf.fit_pred)
#accuracy of model
rf.fit.acc = round(mean(rf.fit_pred == test.data$WarmerTomorrow)*100,
digits = 2)
cat("Random Forest ensemble model accuracy is: ", rf.fit.acc, "%")


###############Question 6###################
#Question: Calculate confidence and construct ROC curve

#Decision Tree
tree.conf = predict(tree.fit, test.data, type = "vector")
tree.conf.pred = prediction(tree.conf[,2], test.data$WarmerTomorrow)
tree.perf = performance(tree.conf.pred, "tpr", "fpr")
plot(tree.perf, col = "black")
abline(0,1)



#Naïve Bayes
naive.conf = predict(naive.fit, test.data, type = "raw")
naive.conf.pred = prediction(naive.conf[,2], test.data$WarmerTomorrow)
naive.perf = performance(naive.conf.pred, "tpr", "fpr")
plot(naive.perf,add = TRUE, col = "blueviolet")


#Bagging
bag.conf = prediction(wbag.fit_pred$prob[,2], test.data$WarmerTomorrow)
bag.perf = performance(bag.conf, "tpr", "fpr")
plot(bag.perf, col = "green", add = TRUE)


#Boosting
boost.conf = prediction(wboost.fit_pred$prob[,2], test.data$WarmerTomorrow)
boost.perf = performance(boost.conf, "tpr", "fpr")
plot(boost.perf, col ="red", add = TRUE)


#Random Forest
```

```
rf.conf = predict(rf.fit, test.data, type = "prob")
rf.conf.pred = prediction(rf.conf[,2], test.data$WarmerTomorrow)
rf.perf = performance(rf.conf.pred, "tpr", "fpr")
plot(rf.perf, col = "cyan", add = TRUE)


#add legend to ROC plot
legend("topleft", legend = c("Decision Tree", "Naive Bayes", "Bagging",
"Boosting", "Random Forest"),col =
c("black","blueviolet","green","red","cyan"), cex = 0.8, lty = 1)


#Calculate the AUC

#Decision Tree
tree.auc = performance(tree.conf.pred, "auc")
tree.auc = as.numeric(tree.auc@y.values)
cat("Decision Tree AUC is: ", tree.auc)

#Naive Bayes
nv.auc = performance(naive.conf.pred, "auc")
nv.auc = as.numeric(nv.auc@y.values)
cat("Naive Bayes AUC is :", nv.auc)

#Bagging
bag.auc = performance(bag.conf, "auc")
bag.auc = as.numeric(bag.auc@y.values)
cat("Bagging AUC is : ", bag.auc)

#Boosting
boost.auc = performance(boost.conf, "auc")
boost.auc = as.numeric(boost.auc@y.values)
cat("Boosting AUC is: ", boost.auc)

#Random Forest
rf.auc = performance(rf.conf.pred, "auc")
rf.auc = as.numeric(rf.auc@y.values)
cat("Random Forest is : ", rf.auc)


###############Question 8###################

#Decision Tree
summary(tree.fit)

#Naive Bayes
#Cannot calculate as it is not a tree based  classifier.

#Bagging
bagging_importance = as.data.frame(as.table(wbag.fit$importance))

#Boosting
boosting_importance = as.data.frame(as.table(wboost.fit$importance))

#Random Forest
rf_importance = as.data.frame(as.table(rf.fit$importance))
```

```
#Remove variable "Rainfall" from train and test data.
new_train.data = train.data[,-7]
new_test.data = test.data[, -7]

##### Fit the model to the new training and test data after removing
variable "Rainfall"####

#Decision Tree
imp_tree.fit = tree(WarmerTomorrow ~., data =new_train.data, method =
"class")
imp_tree.pred = predict(imp_tree.fit, new_test.data, type = "class")
#confusion matrix
imp_tree.cfm = table(actual = new_test.data$WarmerTomorrow, predicted =
imp_tree.pred)
#accuracy of model
imp_tree.acc = round(mean(imp_tree.pred ==
new_test.data$WarmerTomorrow)*100, digits = 2)
cat("Decision Tree accuracy is: ", imp_tree.acc, "%")



##Bagging
imp_wbag.fit = bagging(WarmerTomorrow ~., new_train.data, mfinal = 10)
imp_wbag.fit_pred = predict.bagging(imp_wbag.fit, new_test.data)
#confusion matrix
imp_wbag.fit.cfm = imp_wbag.fit_pred$confusion
#accuracy of model
imp_wbag.fit.acc = round(mean(imp_wbag.fit_pred$class ==
new_test.data$WarmerTomorrow)*100, digits = 2)
cat("Bagging ensemble model accuracy is: ", imp_wbag.fit.acc, "%")



#Boosting
imp_wboost.fit = boosting(WarmerTomorrow ~., new_train.data, mfinal = 10)
imp_wboost.fit_pred = predict.boosting(imp_wboost.fit, new_test.data)
#confusion matrix
imp_wboost.fit.cfm = imp_wboost.fit_pred$confusion
#accuracy of model
imp_wboost.fit.acc = round(mean(imp_wboost.fit_pred$class ==
new_test.data$WarmerTomorrow)*100, digits = 2)
cat("Boosting ensemble model accuracy is: ", imp_wboost.fit.acc, "%")


#Random Forest
imp_rf.fit = randomForest(WarmerTomorrow ~., new_train.data)
imp_rf.fit_pred = predict(imp_rf.fit, new_test.data)
#confusion matrix
imp_rf.fit.cfm = tree.cfm = table(actual = new_test.data$WarmerTomorrow,
predicted = imp_rf.fit_pred)
#accuracy of model
imp_rf.fit.acc = round(mean(imp_rf.fit_pred ==
new_test.data$WarmerTomorrow)*100, digits = 2)
cat("Random Forest ensemble model accuracy is: ", imp_rf.fit.acc, "%")
```

```
###############Question 9###################

#perform cross validation for our decision tree
test.fit = cv.tree(tree.fit, FUN = prune.misclass)
print(test.fit)

#prune down the tree size
prune.tree.fit = prune.misclass(tree.fit, best = 4)
summary(prune.tree.fit)

#plot prune decision tree
plot(prune.tree.fit)
text(prune.tree.fit, pretty = 0)

#calculate accuracy of prune decision tree
ppredict = predict(prune.tree.fit, test.data, type="class")
table(actual = test.data$WarmerTomorrow, predicted = ppredict)
#accuracy = 59.2

#calculate confidence and AUC of prune decision tree
prune.tree.conf = predict(prune.tree.fit, test.data, type = "vector")
prune.tree.pred = prediction(prune.tree.conf[,2], test.data$WarmerTomorrow)
ptree.auc = performance(prune.tree.pred, "auc")
ptree.auc = as.numeric(ptree.auc@y.values)
cat("Prune Decision Tree AUC is: ", ptree.auc)
#AUC = 0.624


###############Question 10###################
#Find original number of trees used
rf.fit$ntree #500

#save training and testing data in new variable
imp.train = train.data
imp.test = test.data

#Algorithm to fit the number of trees from 501 to 700 by tuning different
mtry value
for(ntrees in 500:700){
  set.seed(31842305)
  new_rf.fit = randomForest(WarmerTomorrow ~., data = imp.train, importance
= TRUE, ntree = ntrees, mtry=2)
  new_rf.pred = predict(new_rf.fit, imp.test)
  new_rf.cfm = table(actual = imp.test$WarmerTomorrow, predicted =
new_rf.pred)
  new_rf.acc = round(mean(new_rf.pred == imp.test$WarmerTomorrow)*100,
digits = 2)

  #calculate confidence and AUC of the new Random Forest model
  new_rf.conf = predict(new_rf.fit, imp.test, type = "prob")
  new_rf.conf.pred = prediction(new_rf.conf[,2], imp.test$WarmerTomorrow)
  new_rf.auc = performance(new_rf.conf.pred, "auc")
  new_rf.auc = as.numeric(new_rf.auc@y.values)

  cat("Best Tree is: ", new_rf.fit$ntree, "accuracy is: ", new_rf.acc ,
"AUC value: ", new_rf.auc ,"\n")
```

```
}


###############Question 11###################
#Package to train the classifier
install.packages("neuralnet")
library(neuralnet)
library(caret)

#Store train data and test data in new variable
nn_train = train.data
nn_test = test.data

#one hot encoding for categorical variable
dummy <- dummyVars("~.", data = nn_train[,1:24])
dummy2 <- dummyVars("~.", data = nn_test[,1:24])
nn_train = data.frame(predict(dummy, newdata = nn_train))
nn_test = data.frame(predict(dummy, newdata = nn_test))

#Function for max-min normalise
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

#Normalize the data using max-min method
ann_train = as.data.frame(lapply(nn_train, FUN = normalize))
ann_test = as.data.frame(lapply(nn_test, FUN = normalize))

#Fit the classifier
ws.nn = neuralnet(WarmerTomorrow.0 + WarmerTomorrow.1 ~., data = ann_train,
hidden = 3 , linear.output = FALSE, threshold = 0.01, stepmax = 1e7)

#Visualize result
plot(ws.nn)

#Evaluate model performance
ws_pred = compute(ws_nn, ann_test)
ws_pred = as.data.frame(round(ws_pred$net.result, 0))
table(actual = ann_test$WarmerTomorrow.1 , predicted = ws_pred$V1)
```