

Assignment 2 Report

Rui Qin | FIT3152 | 18/05/2022

Contents

[Explore the data 2](#_Toc103981129)

[Data tidy 3](#_Toc103981130)

[Implement classification models 4](#_Toc103981131)

[Model performance 4](#_Toc103981132)

[Data performance 5](#_Toc103981133)

[Based on the decision tree 5](#_Toc103981134)

[Bagging, Boosting and Random Forest 6](#_Toc103981135)

[Top 5 important variables 6](#_Toc103981136)

[Models Improvement 7](#_Toc103981137)

[Prune Decision Tree 8](#_Toc103981138)

[Bagging Cross-Validation 8](#_Toc103981139)

[Boosting Cross-Validation 8](#_Toc103981140)

[Random Forest Cross-Validation 8](#_Toc103981141)

[Artificial Neural Network 9](#_Toc103981142)

[Appendix 10](#_Toc103981143)

[Confusion matrix 10](#_Toc103981144)

[Accuracy 11](#_Toc103981145)

[AUC 11](#_Toc103981146)

[Coding 11](#_Toc103981147)

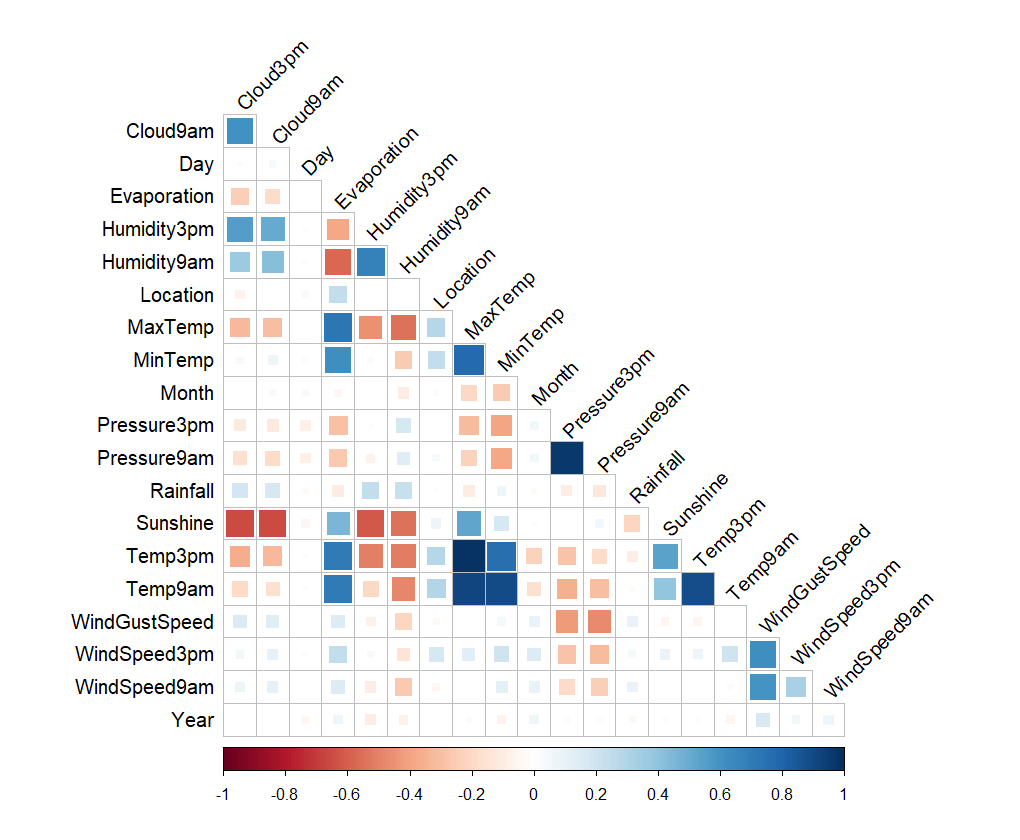
# Explore the data

After browsing the data, it is obvious that there is a lot of NAs value inside of data, which may bring the effect of the result. Before removing the NA value, we can check the proportion of warmer tomorrow, the result is 0.5524, which means almost half of the day is warmer than the last day.

Below is the table of attributes’ means and standard deviations, we remove the date attributes because the date will not affect the temperature.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Location | MinTemp | MaxTemp | Rainfall | Evaporation | Sunshine | WindGustDir |
| Stdv | 10.7382 | 5.9666 | 6.6622 | 7.8240 | 3.1769 | 3.6913 | NA |
| Mean | 28.7975 | 12.2718 | 22.7097 | 2.3676 | 4.7488 | 7.6185 | NA |
|  | WindGustSpeed | WindDir9am | WindDir3pm | WindSpeed9am | WindSpeed3pm | Humidity9am | Humidity3pm |
| Stdv | 13.5888 | NA | NA | 7.4681 | 7.9301 | 17.4299 | 18.8083 |
| Mean | 42.8146 | NA | NA | 16.7087 | 21.8364 | 67.3816 | 54.1526 |
|  | Pressure9am | Pressure3pm | Cloud9am | Cloud3pm | Temp9am | Temp3pm | Warmer  Tomorrow |
| Stdv | 7.4806 | 7.3587 | 2.7940 | 2.6798 | 6.3712 | 6.5216 | 0.4995 |
| Mean | 1017.3073 | 1015.1489 | 4.4377 | 4.4283 | 17.3835 | 21.1344 | 0.5296 |

Every attribute can have a relationship with warmer tomorrow, and they also have a relationship to cause other factors to change. We make a correlation graph of themselves.



This graph shows that day, month and year have a small direct relationship with temperature, pressure and other factors, which means they indirectly have a small relationship with the result. We also can notice location also has a low relation rate, we can remove date data and location. However, location data still has a small positive correlation with temperature and temperature at 3 am/pm, therefore we only remove date data.

# Data tidy

After removing date data and NA value, we only have 642 objects and 21 variables. In this part, we also factor the columns which form with character. Data is separated into two parts, the training part occupies 70% and the testing part is 30%.

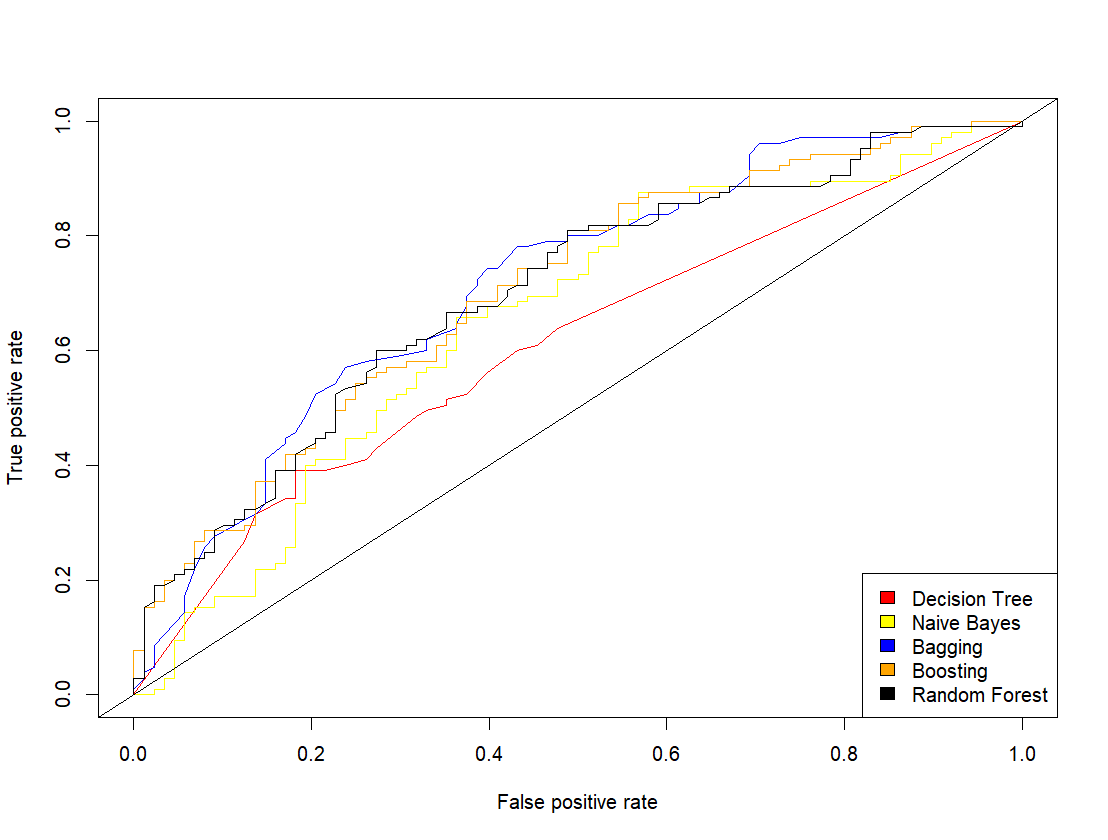
# Implement classification models

## Model performance

We apply the Decision Tree, Naïve Bayes, Bagging, Boosting and Random Forest to the data, and then we can start testing and getting the performance of these models. In order to get a good view of performance, we can calculate the accuracy base on the confusion matrix, and calculate AUC based on the ROC. The below table lists the accuracy and AUC of them

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Decision Tree | Naïve Bayes | Bagging | Boosting | Random Forest |
| Accuracy | 0.5544 | 0.601 | 0.6632 | 0.6269 | 0.6528 |
| AUC | 0.6541 | 0.6541 | 0.7152 | 0.7057 | 0.7004 |

Based on the above table, the bagging model has the best accuracy and AUC, which means it is the best model for the dataset. The better AUC means the model has a better ability to distinguish warmer tomorrow.



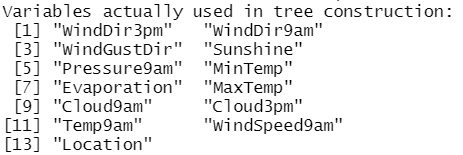
In the ROC diagram, if the curve closer to the top left means better performance. The decision tree has the worst performance, which has the lowest accuracy and AUC. Random forest and boosting perform better than Naïve Bayes.

## Data performance

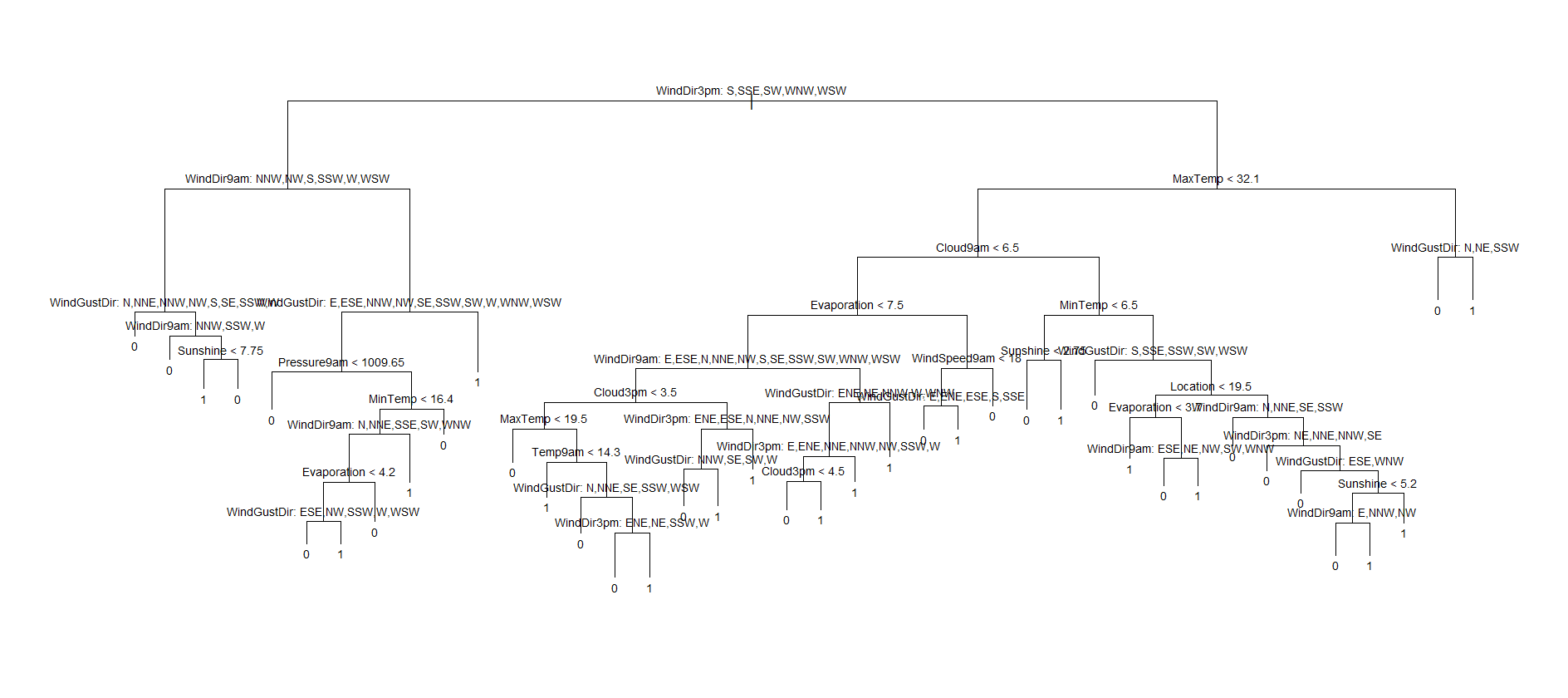
After analysing the model performance, we can start digging into the data performance base on the model.

### Based on the decision tree

The table below shows the variables used in the tree with their priority. On the top are WindDir3pm and WindDir9am, which are the most important for the decision tree, and location has the lowest priority for judging whether it is going to be warmer or not.

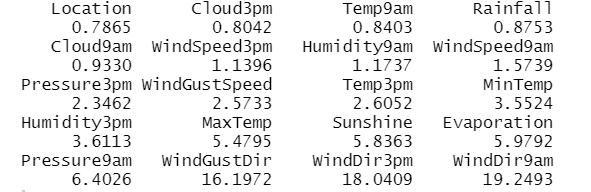


Below is the tree, windDir3pm is the first variable for the model to judge.

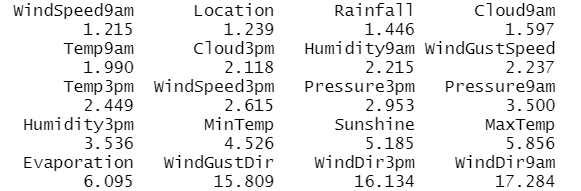


### Bagging, Boosting and Random Forest

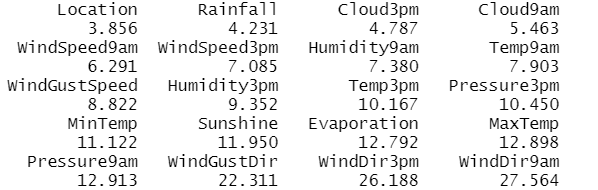
#### Bagging



#### Boosting



#### Random forest



### Top 5 important variables

The below table is the conclusion of the top 8 best variables base on the analysis above

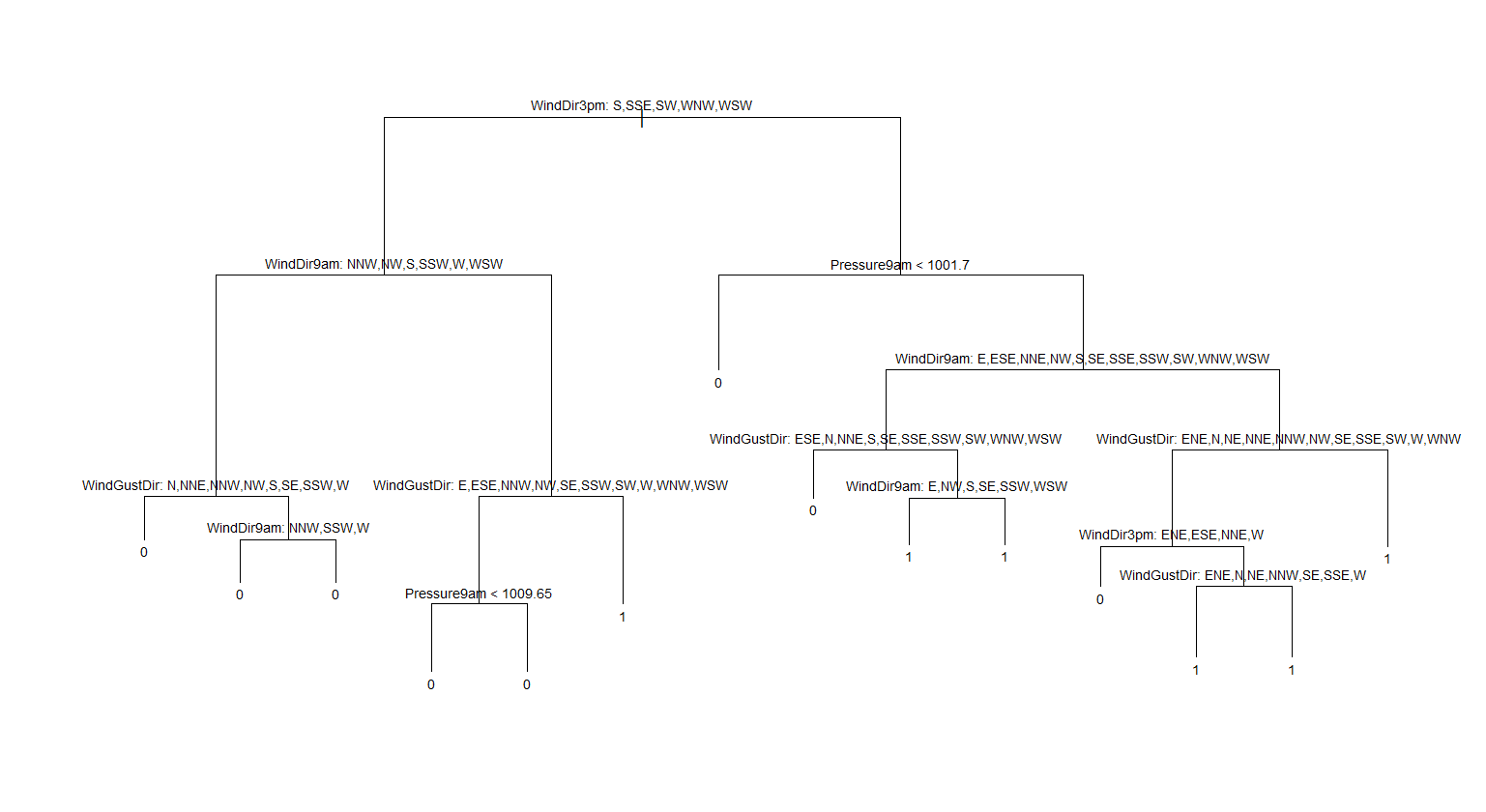
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Decision tree | Bagging | Boosting | Random forest |
| Top 1 | WindDir3pm | WindDir9am | WindDir9am | WindDir9am |
| Top 2 | WindDir9am | WindDir3pm | WindDir3pm | WindDir3pm |
| Top 3 | WindGustDir | WindGustDir | WindGustDir | WindGustDir |
| Top 4 | Sunshine | Pressure9am | Evaporation | Pressure9am |
| Top 5 | Pressure9am | Evaporation | MaxTemp | MaxTemp |
| Top 6 | MinTemp | Sunshine | Sunshine | Evaporation |
| Top 7 | Evaporation | MaxTemp | MinTemp | Sunshine |
| Top 8 | MaxTemp | Humidity3pm | Humidity3pm | MinTemp |

If we focus on top4 to top8, we notice humidity3pm only shows up 2 times, and with the comparison, we can conclude that the list of the decision tree is the list of the top 8.

# Models Improvement

simple classifier

With the top 8 important variables, we can design our simple classifier. The classifier I choose is the decision tree because it is easy to normal people to understand and easy to visualize.

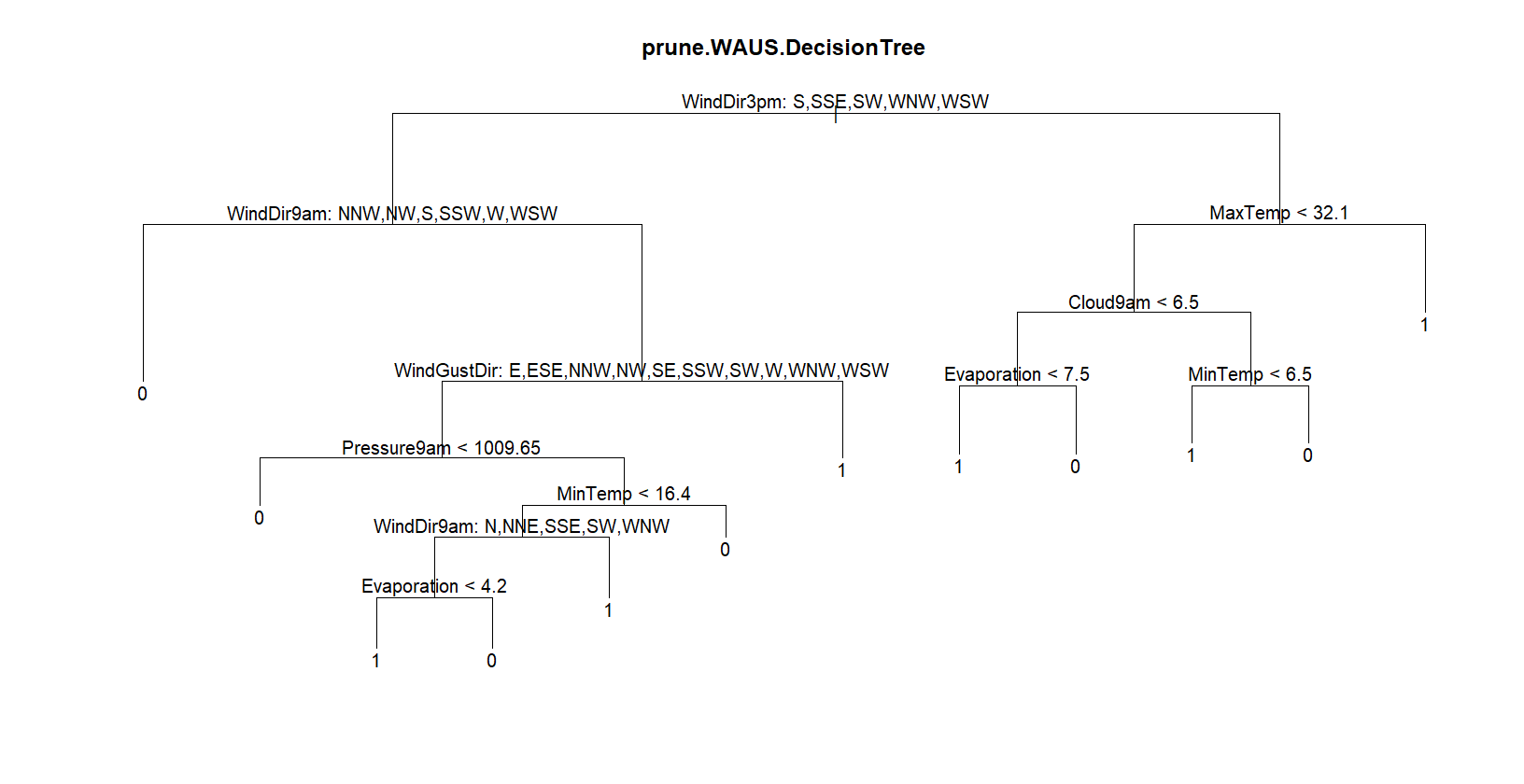


After we create a new tree, we can use it to go through the original testing data. Even if the tree is cut, the accuracy is 0.5389, and AUC is staying the same which is 0.6541

optimizing individual model

In this part, I optimize the model I create individually.

### Prune Decision Tree



After choosing the difference best variable, the above tree is based on the best 4 variables. The accuracy jumps up to 0.6166, and it is the best accuracy with the lowest number of variables.

### Bagging Cross-Validation

After cross-validation, the accuracy of bagging drops to 0.6481 from 0.6632.

### Boosting Cross-Validation

With boosting cross-validation, the accuracy improves to 0.6748 from 0.6269.

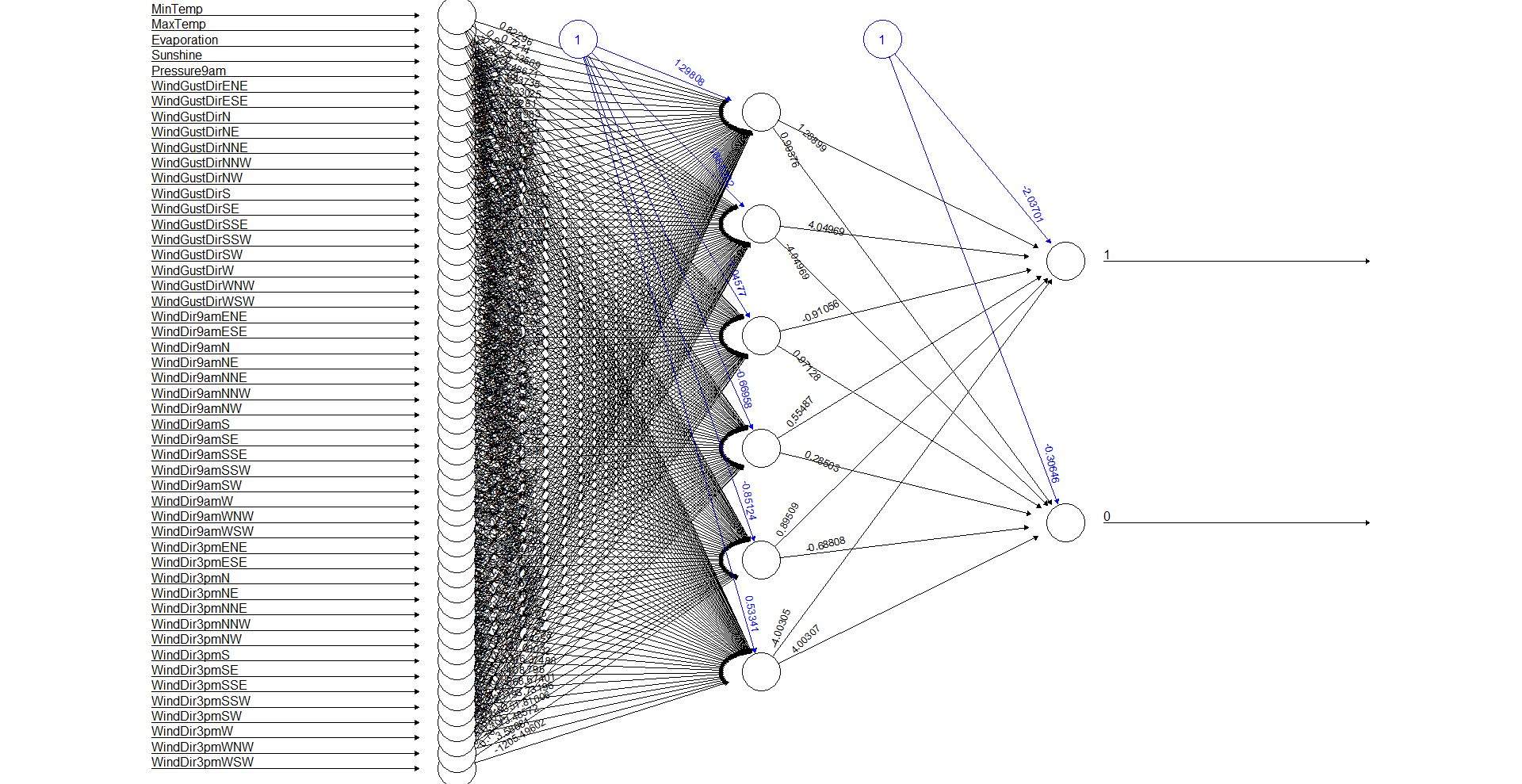
### Random Forest Cross-Validation



We notice if mtry=10, the random forest has the best performance. The accuracy improves to 0.6736 from 0.6528.

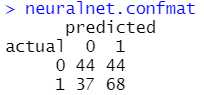
# Artificial Neural Network

The below network is based on the top 8 important variables.



We include Wind direction values are characters, I use the model. matrix to separate them into small pieces, and we have 6 hidden layers to decide whether it is going to be warmer or not.

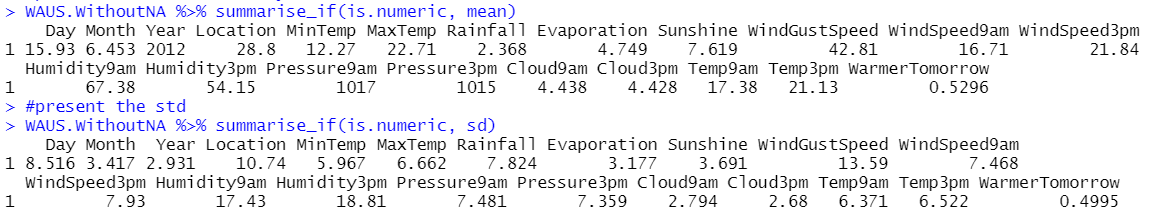
The table shown below is the confusion matrix of the neural network.



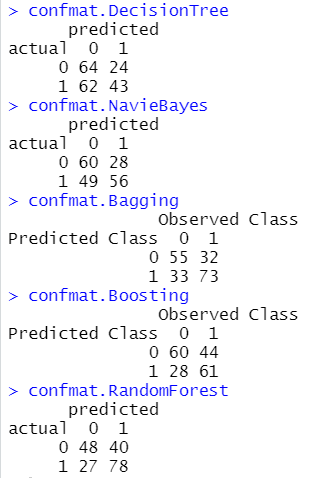
The accuracy is 0.5803 which is lower than the bagging model. Therefore, the bagging classifier performs better than others. This is maybe the wind direction situation effect the ANN model, which brings a lot of indeterminacy

# Appendix

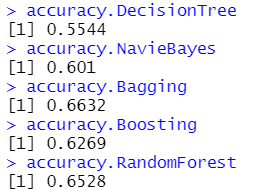




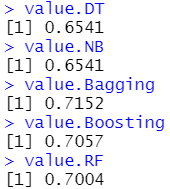
## Confusion matrix



## Accuracy



## AUC



## Coding

library(dplyr)

library(rpart)

library(rpart.plot)

library(corrplot)

library(randomForest)

library(adabag)

library(tree)

library(e1071)

library(ROCR)

setwd("C:/Users/aud/My Drive/Documents/Assignment/2-SEM\_1/FIT3152/A2 (20%)")

rm(list = ls())

WAUS <- read.csv("WarmerTomorrow2022.csv")

L <- as.data.frame(c(1:49))

set.seed(30874157) # 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

#the WAUS without NA value object

WAUS.WithoutNA = na.omit(WAUS)

#setting 4 decimals

options(digits=4)

#----------

#Question 1

#----------

#delete Warmer Tomorrow NA value

bad.WAUS = is.na(WAUS$WarmerTomorrow)

WAUS.WTWithoutNA=WAUS[!bad.WAUS,]

#proportion warmer vs not warmer

count.WT = WAUS.WTWithoutNA %>% count(WarmerTomorrow)

proportion = count.WT$n[2]/(count.WT$n[1]+count.WT$n[2])

proportion

remove(WAUS.WTWithoutNA)

#present the mean of every elements relate to warmer

WAUS.WithoutNA %>% summarise\_if(is.numeric, mean)

#present the std

WAUS.WithoutNA %>% summarise\_if(is.numeric, sd)

#Calculate the correlation of them to find any element should remove

A2\_Correlation <- na.omit(subset(WAUS, select = -c(WindGustDir,

WindDir9am,

WindDir3pm,

WarmerTomorrow)))

A2\_Correlation = cor(A2\_Correlation)

corrplot(A2\_Correlation,

method = 'square',

order = 'alphabet',

type = 'lower',

tl.col= "black",

tl.srt= 45,

diag = FALSE)

#----------

#Question 2

#----------

#Remove Day, Year, Month, Location

#Remove NA value

WAUS.WithoutNA$Day<-NULL

WAUS.WithoutNA$Month<-NULL

WAUS.WithoutNA$Year<-NULL

#factor

WAUS.WithoutNA$WindGustDir = as.factor(WAUS.WithoutNA$WindGustDir)

WAUS.WithoutNA$WindDir9am = as.factor(WAUS.WithoutNA$WindDir9am)

WAUS.WithoutNA$WindDir3pm = as.factor(WAUS.WithoutNA$WindDir3pm)

WAUS.WithoutNA$WarmerTomorrow = as.factor(WAUS.WithoutNA$WarmerTomorrow)

#----------

#Question 3

#----------

set.seed(30874157) #Student ID as random seed

train.row = sample(1:nrow(WAUS.WithoutNA), 0.7\*nrow(WAUS.WithoutNA))

WAUS.train = WAUS.WithoutNA[train.row,]

WAUS.test = WAUS.WithoutNA[-train.row,]

remove(A2\_Correlation)

remove(count.WT)

#----------

#Question 4

#----------

#decision tree

WAUS.DecisionTree = tree(WarmerTomorrow~., data = WAUS.train)

plot(WAUS.DecisionTree)

WAUS.DecisionTree

summary(WAUS.DecisionTree)

#naive bayes

WAUS.NavieBayes = naiveBayes(WarmerTomorrow~., data = WAUS.train)

#bagging

WAUS.Bagging = bagging(WarmerTomorrow ~. , data = WAUS.train)

#boosting

WAUS.Boosting = boosting(WarmerTomorrow ~. , data = WAUS.train)

#random Forest

WAUS.RandomForest = randomForest(WarmerTomorrow ~. ,

data = WAUS.train,

na.action = na.exclude)

#----------

#Question 5

#----------

#prepare predict data

WAUS.predict.DecisionTree = predict(WAUS.DecisionTree,

WAUS.test,type = "class")

WAUS.predict.NavieBayes = predict(WAUS.NavieBayes, WAUS.test)

WAUS.predict.Bagging = predict.bagging(WAUS.Bagging,WAUS.test)

WAUS.predict.Boosting = predict.boosting(WAUS.Boosting, WAUS.test)

WAUS.predict.RandomForest = predict(WAUS.RandomForest, WAUS.test)

#confusion matrix of them

confmat.DecisionTree = table(actual = WAUS.test$WarmerTomorrow,

predicted = WAUS.predict.DecisionTree)

confmat.NavieBayes = table(actual = WAUS.test$WarmerTomorrow,

predicted = WAUS.predict.NavieBayes)

confmat.Bagging = WAUS.predict.Bagging$confusion

confmat.Boosting = WAUS.predict.Boosting$confusion

confmat.RandomForest = table(actual = WAUS.test$WarmerTomorrow,

predicted = WAUS.predict.RandomForest)

#print out confusion matrix

confmat.DecisionTree

confmat.NavieBayes

confmat.Bagging

confmat.Boosting

confmat.RandomForest

#calculate accuracy

accuracy.DecisionTree = (sum(diag(confmat.DecisionTree))/

sum(confmat.DecisionTree))

accuracy.NavieBayes = (sum(diag(confmat.NavieBayes))/

sum(confmat.NavieBayes))

accuracy.Bagging = (sum(diag(confmat.Bagging))/

sum(confmat.Bagging))

accuracy.Boosting =(sum(diag(confmat.Boosting))/

sum(confmat.Boosting))

accuracy.RandomForest =(sum(diag(confmat.RandomForest))/

sum(confmat.RandomForest))

#print out accuracy (Bagging and RandomForest)

accuracy.DecisionTree

accuracy.NavieBayes

accuracy.Bagging

accuracy.Boosting

accuracy.RandomForest

#----------

#Question 6

#----------

#####ROC#####

###Decision Tree

WAUS.pred.tree = predict(WAUS.DecisionTree, WAUS.test, type = "vector")

prediction.DT = prediction( WAUS.pred.tree[,2], WAUS.test$WarmerTomorrow)

performance.DT = performance(prediction.DT,"tpr","fpr")

plot(performance.DT, col='red')

###Navie Bayes

WAUS.pred.bayes = predict(WAUS.NavieBayes, WAUS.test, type = "raw")

predict.bayes = prediction( WAUS.pred.bayes[,2], WAUS.test$WarmerTomorrow)

performance.NB = performance(predict.bayes,"tpr","fpr")

plot(performance.NB, add=TRUE, col='yellow')

###Bagging

predict.Bagging = prediction( WAUS.predict.Bagging$prob[,2],

WAUS.test$WarmerTomorrow)

performance.Bagging = performance(predict.Bagging,"tpr","fpr")

plot(performance.Bagging, add=TRUE, col='blue')

###Boosting

predict.Boosting = prediction( WAUS.predict.Boosting$prob[,2],

WAUS.test$WarmerTomorrow)

performance.Boosting = performance(predict.Boosting,"tpr","fpr")

plot(performance.Boosting, add=TRUE, col='orange')

###Random Forest

WAUS.pred.forest = predict(WAUS.RandomForest, WAUS.test, type = "prob")

predict.RF = prediction( WAUS.pred.forest[,2], WAUS.test$WarmerTomorrow)

performance.RF = performance(predict.RF,"tpr","fpr")

plot(performance.RF, add=TRUE, col='black')

abline(0,1)

legend("bottomright",

legend=c("Decision Tree",

"Naive Bayes",

"Bagging",

"Boosting",

"Random Forest"),

fill=c('red',

'yellow',

'blue',

'orange',

'black'))

#####AUC#####(Bagging is the best)

###Decision Tree

waus.AUC.DT = performance(predict.bayes, "auc")

value.DT = as.numeric(waus.AUC.DT@y.values)

###Navie Bayes

waus.AUC.NB = performance(predict.bayes, "auc")

value.NB = as.numeric(waus.AUC.NB@y.values)

###Bagging

waus.AUC.Bagging = performance(predict.Bagging, "auc")

value.Bagging = as.numeric(waus.AUC.Bagging@y.values)

###Boosting

waus.AUC.Boosting = performance(predict.Boosting, "auc")

value.Boosting = as.numeric(waus.AUC.Boosting@y.values)

###Random Forest

waus.AUC.RF = performance(predict.RF, "auc")

value.RF = as.numeric(waus.AUC.RF@y.values)

#print result

value.DT

value.NB

value.Bagging

value.Boosting

value.RF

#----------

#Question 7

#----------

#Bagging

#----------

#Question 8

#----------

###Decision Tree

plot(WAUS.DecisionTree)

text(WAUS.DecisionTree, pretty = 0,cex = 0.7)

title("WAUS.DecisionTree")

summary(WAUS.DecisionTree)

###Navie Bayes (Not found)

###Bagging

sort(WAUS.Bagging$importance)

###Boosting

sort(WAUS.Boosting$importance)

###Random Forest

WAUS.RandomForest$importance[order(WAUS.RandomForest$importance),]

#----------

#Question 9

#----------

set.seed(30874157) #Student ID as random seed

WAUS.Q10 = WAUS.WithoutNA

WAUS.Q10 = WAUS.Q10[,c(5,7,9,10,15,21)]

WAUS.Q10 = na.omit(WAUS.Q10)

train.row = sample(1:nrow(WAUS.Q10), 0.7\*nrow(WAUS.Q10))

WAUS.train.Q10 = WAUS.Q10[train.row,]

WAUS.test.Q10 = WAUS.Q10[-train.row,]

WAUS.DecisionTree2 = tree(WarmerTomorrow~., data = WAUS.train.Q10,method = "vector")

plot(WAUS.DecisionTree2)

text(WAUS.DecisionTree2,pretty = 0,cex = 0.7)

#testing in origin data

WAUS.predict.DecisionTree2 = predict(WAUS.DecisionTree2,

WAUS.test,type = "class")

#confmat

confmat.DecisionTree2 = table(actual = WAUS.test$WarmerTomorrow,

predicted = WAUS.predict.DecisionTree2)

#accuracy

accuracy.DecisionTree2 = (sum(diag(confmat.DecisionTree2))/

sum(confmat.DecisionTree2))

accuracy.DecisionTree2

WAUS.pred.tree2 = predict(WAUS.DecisionTree2, WAUS.test, type = "vector")

prediction.DT2 = prediction( WAUS.pred.tree2[,2], WAUS.test$WarmerTomorrow)

performance.DT2 = performance(prediction.DT2,"tpr","fpr")

#AUC

waus.AUC.DT2 = performance(predict.bayes, "auc")

as.numeric(waus.AUC.DT@y.values)

#----------

#Question 10

#----------

######Decision Tree######

test.WAUS.fit=cv.tree(WAUS.DecisionTree, FUN=prune.misclass)

test.WAUS.fit

#After testing, best = 4 has lowest dev

prune.WAUS.DecisionTree = prune.misclass(WAUS.DecisionTree, best=9)

summary(prune.WAUS.DecisionTree)

plot(prune.WAUS.DecisionTree)

text(prune.WAUS.DecisionTree, pretty=0)

title("prune.WAUS.DecisionTree")

#test accuracy after pruning

WAUS.prune.predict = predict(prune.WAUS.DecisionTree,WAUS.test,type = "class")

confmat.prune.DecisionTree = table(predicted = WAUS.prune.predict,

actual = WAUS.test$WarmerTomorrow)

confmat.prune.DecisionTree

confmat.DecisionTree

accuracy.prune.DecisionTree = (sum(diag(confmat.prune.DecisionTree))/

sum(confmat.prune.DecisionTree))

accuracy.prune.DecisionTree

accuracy.DecisionTree

######Bagging Cross Validation######

WAUS.Bagging.cv = bagging.cv(WarmerTomorrow ~ .,

v = 10,

WAUS.train)

confmat.Bagging = WAUS.Bagging.cv$confusion

confmat.Bagging

accuracy.Bagging.cv = (sum(diag(confmat.Bagging))/

sum(confmat.Bagging))

#Compare accuracy

accuracy.Bagging

accuracy.Bagging.cv

######Boosting Cross Validation######

WAUS.Boosting.cv = boosting.cv(WarmerTomorrow ~ .,

v = 10,

WAUS.train)

confmat.Boosting = WAUS.Boosting.cv$confusion

confmat.Boosting

accuracy.Boosting.cv = (sum(diag(confmat.Boosting))/

sum(confmat.Boosting))

#Compare accuracy

accuracy.Boosting

accuracy.Boosting.cv

######Random Forest######

rf.cv = rfcv(WAUS.train[,-c(19)],

WAUS.train[,c(19)],

cv.fold=10,

scale="log",

step=0.5,

mtry=function(p) max(1, floor(sqrt(p))),

recursive=FALSE)

rf.cv$error.cv

with(rf.cv, plot(n.var, error.cv))

#mtry = 10

WAUS.CV.RandomForest = randomForest(WarmerTomorrow~.,

WAUS.train,

na.action = na.exclude,

mtry=10)

WAUS.CV.RandomForest

WAUS.predict.CV.RandomForest = predict(WAUS.CV.RandomForest, WAUS.test)

confmat.CV.RandomForest = table(actual = WAUS.test$WarmerTomorrow,

predicted = WAUS.predict.CV.RandomForest)

confmat.RandomForest

confmat.CV.RandomForest

accuracy.CV.RandomForest =(sum(diag(confmat.CV.RandomForest))/

sum(confmat.CV.RandomForest))

accuracy.RandomForest

accuracy.CV.RandomForest

#----------

#Question 11

#----------

library(neuralnet)

ANN.WAUS = WAUS

ANN.WAUS = na.omit(ANN.WAUS)

ANN.m = as.data.frame(model.matrix(~WindGustDir+WindDir9am+WindDir3pm, data=ANN.WAUS))

#factor

ANN.WAUS = cbind(ANN.WAUS,ANN.m)

ANN.WAUS$WarmerTomorrow = factor(ANN.WAUS$WarmerTomorrow)

ANN.WAUS

ANN.WAUS = ANN.WAUS[,-c(1:4,7,10:17,19:23)]

ANN.WAUS$`(Intercept)`<-NULL

#ANN.WAUS = ANN.WAUS %>% relocate(WarmerTomorrow, .after = WindDir3pmWNW)

ANN.WAUS

set.seed(30874157)

train.row = sample(1:nrow(ANN.WAUS), 0.7\*nrow(ANN.WAUS))

ANN.WAUS.train = ANN.WAUS[train.row,]

ANN.WAUS.test = ANN.WAUS[-train.row,]

neuralnet.WAUS = neuralnet(WarmerTomorrow~

MinTemp+MaxTemp+Evaporation+Sunshine+Pressure9am +

WindGustDirENE+WindGustDirESE+WindGustDirN+

WindGustDirNE+WindGustDirNNE+WindGustDirNNW+

WindGustDirNW+WindGustDirS+WindGustDirSE+

WindGustDirSSE+WindGustDirSSW+WindGustDirSW+

WindGustDirW+WindGustDirWNW+WindGustDirWSW+

WindDir9amENE+WindDir9amESE+WindDir9amN+

WindDir9amNE+WindDir9amNNE+WindDir9amNNW+

WindDir9amNW+WindDir9amS+WindDir9amSE+

WindDir9amSSE+WindDir9amSSW+WindDir9amSW+

WindDir9amW+WindDir9amWNW+WindDir9amWSW+

WindDir3pmENE+WindDir3pmESE+WindDir3pmN+

WindDir3pmNE+WindDir3pmNNE+WindDir3pmNNW+

WindDir3pmNW+WindDir3pmS+WindDir3pmSE+

WindDir3pmSSE+WindDir3pmSSW+WindDir3pmSW+

WindDir3pmW+WindDir3pmWNW+WindDir3pmWSW,

ANN.WAUS.train,

hidden = 6,

linear.output = FALSE)

#tidy data

neuralnet.WAUS$result.matrix

neuralnet.WAUS.predict = neuralnet::compute(neuralnet.WAUS,ANN.WAUS.test)

neuralnet.WAUS.predict$net.result

neuralnet.WAUS.predict = round(neuralnet.WAUS.predict$net.result,0)

# Create confusion matrix and calculate the accuracy

neuralnet.confmat = table(actual=ANN.WAUS.test$WarmerTomorrow,

predicted=neuralnet.WAUS.predict)

neuralnet.confmat

neuralnet.WAUS.accuracy = (sum(diag(neuralnet.confmat))/

sum(neuralnet.confmat))

neuralnet.WAUS.accuracy

plot(neuralnet.WAUS)