MLPH Final Project

Monal Daterao, Devin Nathan

2025-04-14

About

The below code uses the Air Quality and Health Impact Dataset from Kaggle in order to examine the relationship between air quality and its impact on health.

The dataset can be found at: https://www.kaggle.com/datasets/rabieelkharoua/air-quality-and-health-impact-dataset/data

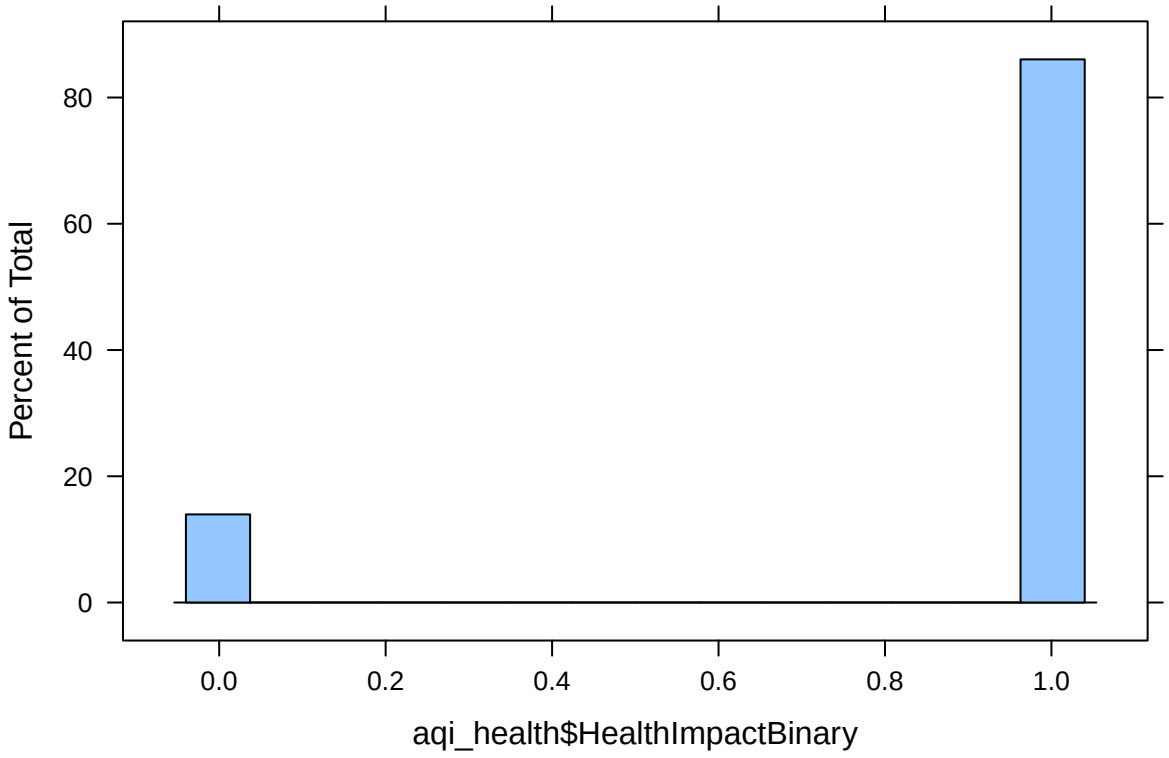
The opensource version of this code can be found on github at: https://github.com/mdaterao/final_project_group13

Load Libraries

```
library(tidyverse)
library(readr)
library(caret)
library(MASS)
library(boot)
library(glmnet)
library(caret)
library(plotmo)
library(tree)
library(randomForest)
```

Load Data

```
aqi_health <- read.csv("air_quality_health_impact_data.csv")

## Add binary HealthImpactScore variable:
aqi_health$HealthImpactBinary <- ifelse(aqi_health$HealthImpactScore >= 80, 1, 0)
# Histogram of HealthImpactBinary distribution
histogram(aqi_health$HealthImpactBinary)
```




```
#Table of HealthImpactBinary
table(aqi_health$HealthImpactBinary)
```

```
##
##    0    1
##  811 5000
```

Creation of binary HealthImpactClass Variable: HealthImpactBinary

Created a variable that dichotomized HealthImpactScore, which ranges from 0-100. We used the cut-off score of 80 for the binary variable because the HealthImpactScore values were very high overall. Doing this moderately helped to address the class imbalance we observed whent he cut-off score of 50 was used. However there is still class imbalance; High: 86%, Low: 14%

Exploratory Data Analysis

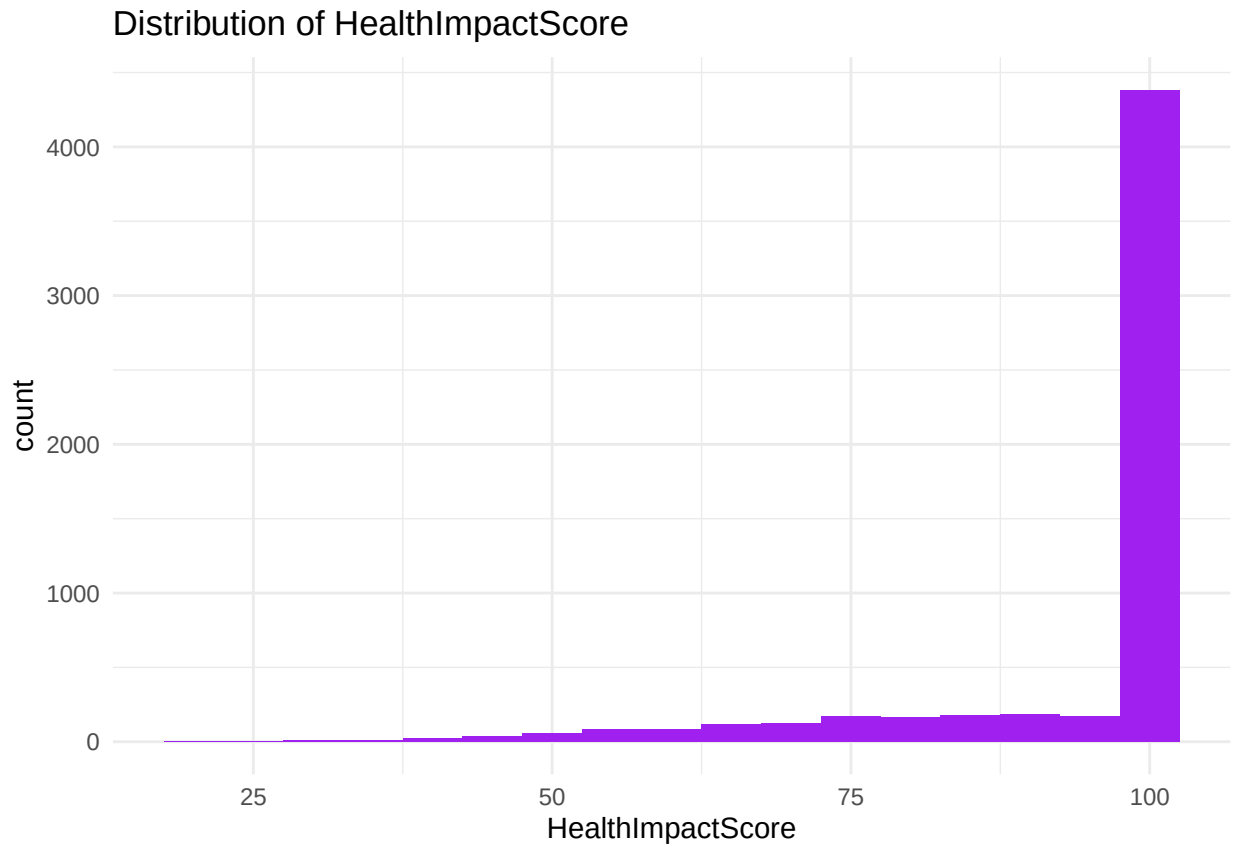
Data exploration was conducted on the predictor variables and outcome variables. Mean, Median, and Standard deviation of each variable was investigated. For the categorical variable of HealthImpactClass, the frequency and percentages were investigated. A histogram and bar plot were created for each to visually investigate the data.

```r
exploratory_analysis <- tibble("Value" = c("AQI", "PM10", "PM2_5", "NO2",
                                           "SO2", "O3", "Temperature", "Humidity",
                                           "WindSpeed", "RespiratoryCases",
                                           "CardiovascularCases", "HospitalAdmissions",
                                           "HealthImpactScore"),
                               "mean" = c(mean(aqi_health$AQI),
                                          mean(aqi_health$PM10),
                                          mean(aqi_health$PM2_5),
                                          mean(aqi_health$NO2),
                                          mean(aqi_health$SO2),
                                          mean(aqi_health$O3),
                                          mean(aqi_health$Temperature),
                                          mean(aqi_health$Humidity),
                                          mean(aqi_health$WindSpeed),
                                          mean(aqi_health$RespiratoryCases),
                                          mean(aqi_health$CardiovascularCases),
                                          mean(aqi_health$HospitalAdmissions),
                                          mean(aqi_health$HealthImpactScore)),
                               "median" = c(median(aqi_health$AQI),
                                            median(aqi_health$PM10),
                                            median(aqi_health$PM2_5),
                                            median(aqi_health$NO2),
                                            median(aqi_health$SO2),
                                            median(aqi_health$O3),
                                            median(aqi_health$Temperature),
                                            median(aqi_health$Humidity),
                                          median(aqi_health$WindSpeed),
                                            median(aqi_health$RespiratoryCases),
                                            median(aqi_health$CardiovascularCases),
                                            median(aqi_health$HospitalAdmissions),
                                          median(aqi_health$HealthImpactScore)),
                               "sd" = c(sd(aqi_health$AQI),
                                        sd(aqi_health$PM10),
                                        sd(aqi_health$PM2_5),
                                        sd(aqi_health$NO2),
                                        sd(aqi_health$SO2),
                                        sd(aqi_health$O3),
                                        sd(aqi_health$Temperature),
                                        sd(aqi_health$Humidity),
                                      sd(aqi_health$WindSpeed),
                                        sd(aqi_health$RespiratoryCases),
                                        sd(aqi_health$CardiovascularCases),
                                        sd(aqi_health$HospitalAdmissions),
                                      sd(aqi_health$HealthImpactScore)))

exploratory_analysis
```

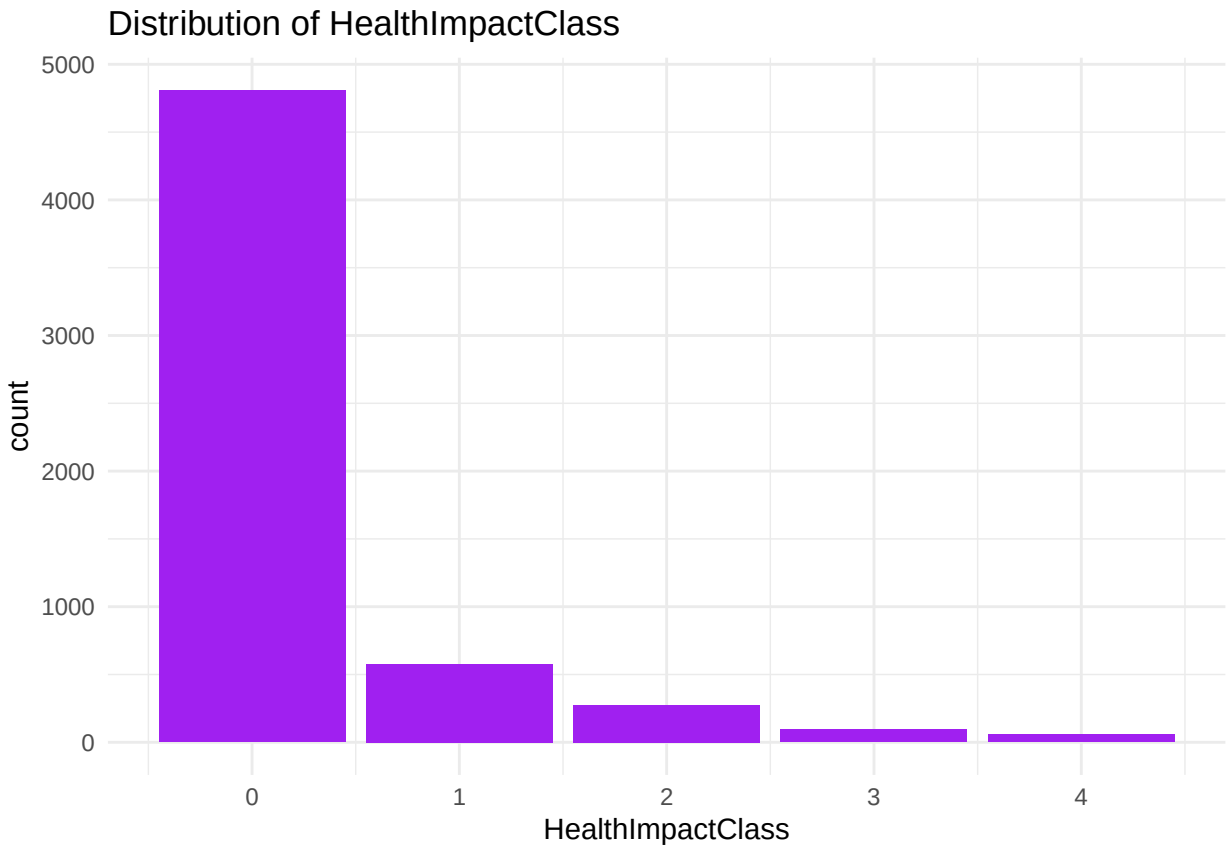
```
## # A tibble: 13 x 4
##    Value                mean median     sd
##    <chr>               <dbl>  <dbl>  <dbl>
##  1 AQI                 248.    249.  145.
##  2 PM10                149.    148.   85.7
##  3 PM2_5               100.    101.   58.1
```

```
##  4 NO2                 102.    103.   57.7
##  5 SO2                  49.5    49.5  28.5
##  6 O3                  149.    150.   86.5
##  7 Temperature          15.0    14.9  14.5
##  8 Humidity             54.8    54.5  26.0
##  9 WindSpeed             9.99   10.1   5.78
## 10 RespiratoryCases      9.97   10     3.13
## 11 CardiovascularCases   4.99    5     2.22
## 12 HospitalAdmissions    2.00    2     1.40
## 13 HealthImpactScore    93.8   100    13.3
```

```r
# Plot of the distribution of Health Impact Class
ggplot(data = aqi_health, mapping = aes(HealthImpactScore)) +
  geom_histogram(binwidth = 5, fill = "purple") +
  ggtitle("Distribution of HealthImpactScore") +
  theme_minimal()
```




```r
# Plot of the distribution of Health Impact Class
ggplot(data = aqi_health, mapping = aes(HealthImpactClass)) +
  geom_bar(fill = "purple") +
  ggtitle("Distribution of HealthImpactClass") +
  theme_minimal()
```




```
# Table of the percentages of Health Impact Class
n_tot <- nrow(aqi_health)

Health_Impact_percent <- aqi_health %>%
  count(HealthImpactClass)

Health_Impact_percent <- Health_Impact_percent %>%
  mutate(percentage = n/n_tot)

Health_Impact_percent
```

```
##   HealthImpactClass    n  percentage
## 1                 0 4808 0.827396317
## 2                 1  579 0.099638616
## 3                 2  273 0.046979866
## 4                 3   95 0.016348305
## 5                 4   56 0.009636896
```

Training and Testing Set

The data was split into 80% training and 20% testing. The output confirms data was split correctly.

```r
n <- nrow(aqi_health)

n_20 <- n * .80

aqi_health_train <- aqi_health[(1:n_20), ]
aqi_health_test <- aqi_health[-(1:n_20), ]

# Double check the numbers
nrow(aqi_health_train)
```

```
## [1] 4648
```

```r
nrow(aqi_health_test)
```

```
## [1] 1163
```

Lasso Regression

Perform a Lasso regression on the standardized data to see which features are most prominent

Standardization of data

```r
#drop the RecordID (1), HealthImpactScore (14), HealthImpactClass(15), and HealthImpactBinary (16)
x_train <- as.matrix(aqi_health_train[ , -c(1, 14, 15, 16)])

#Keep only the HealthImpactScore(14) - Predictor
# Return vectors only with attributes
y_train <- aqi_health_train[, 14, drop = T]

#Do the same steps for the test data
x_test <- as.matrix(aqi_health_test[, -c(1, 14, 15, 16)])
y_test <- aqi_health_test[, 14, drop = T]

standardized_fit <- preProcess(x_train, method = c("center", "scale"))
x_train_standardized <- predict(standardized_fit, x_train)
x_test_standardized <- predict(standardized_fit, x_test)
```

Training and Testing Error

```r
set.seed(0)
cv_fit_lasso <- cv.glmnet(x_train, y_train)
train_pred <- predict(cv_fit_lasso, newx = x_train)
test_pred <- predict(cv_fit_lasso, newx = x_test)
train_error <- mean((train_pred - y_train)^2)
test_error <- mean((test_pred - y_test)^2)

lasso_table <- data.frame("Error Type" = c("Training", "Testing"),
```

```
                           "Value" = c(train_error, test_error))
lasso_table
```

```
##   Error.Type    Value
## 1   Training 93.55742
## 2    Testing 87.64613
```

Coefficients

Obtain the cofficients from the lasso regression that were useful and their weights associated with each.

```
coef(cv_fit_lasso)
```

```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##                                s1
## (Intercept)         70.31625254
## AQI                  0.05208337
## PM10                 0.01749172
## PM2_5                0.03623523
## NO2                  0.01733098
## SO2                  .
## O3                   0.01647706
## Temperature          .
## Humidity             .
## WindSpeed            .
## RespiratoryCases     .
## CardiovascularCases  .
## HospitalAdmissions   .
```

Regression Methods

Linear Regression

Two Linear Regression models were fit. The first linear regression model was fit with all of the predictors besides the RecordID, HealthImpactClass, and HealthImpactBinary. RecordID is an identifier for each of the data's covariates and HealthImpactClass/HealthImpactBinary is another Target Variable. The second linear regression model was fit with only the predictors identified by the lasso regression. Lasso regression does automatic feature selection so the features that were not selected in the lasso were not included within the second model. From the lasso regression we see the variables AQI, PM10, PM2_5, NO2, and O3 are the most prominent features that effect the HealthImpactScore outcome. Both models will have their train and test score evaluated.

- Model 1: All coefficients besides RecordID, HealthImpactClass, and HealthImpactBinary
- Model 2: Select coefficients from Lasso Regression

Linear Regression: Model 1

Model 1 is fit with all coefficients

Model Setup

```
lm_model_1_HealthImpactScore_fit <- lm(HealthImpactScore ~ . -RecordID - HealthImpactClass -HealthImpac
```

Model 1 Training & Testing Error

```
# Training error
pred_HIS_train <- predict(lm_model_1_HealthImpactScore_fit, newdata = aqi_health_train)

train_error_mse <- mean((pred_HIS_train - aqi_health_train$HealthImpactScore)^2)

# Testing error
pred_HIS_test <- predict(lm_model_1_HealthImpactScore_fit, newdata = aqi_health_test)

test_error_mse <- mean((pred_HIS_test - aqi_health_test$HealthImpactScore)^2)

# Output Table
lm_model_1_table <- data.frame("Error_Type" = c("Training", "Testing"),
                               "Value" = c(train_error_mse, test_error_mse))
lm_model_1_table
```

```
##   Error_Type    Value
## 1   Training 90.64787
## 2    Testing 86.61993
```

Linear Regression: Model 2

Model 2 is fit with the AQI, PM10, PM2_5, NO2, and O3 coefficients as identified by the lasso regression.

Definitions: + HIS (Health Impact Score)

Model 2 Setup

```
lm_model_2_HealthImpactScore_fit <- lm(HealthImpactScore ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_hea
```

Model 2 Training & Testing Error

```
# Training error
pred_HIS_train <- predict(lm_model_2_HealthImpactScore_fit, newdata = aqi_health_train)

train_error_mse <- mean((pred_HIS_train - aqi_health_train$HealthImpactScore)^2)

#Testing error
pred_HIS_test <- predict(lm_model_2_HealthImpactScore_fit, newdata = aqi_health_test)

test_error_mse <- mean((pred_HIS_test - aqi_health_test$HealthImpactScore)^2)
```

```
# Output

lm_model_2_table <- data.frame("Error_Type" = c("Training", "Testing"),
                               "Value" = c(train_error_mse, test_error_mse))
lm_model_2_table
```

```
##   Error_Type    Value
## 1   Training 90.84229
## 2    Testing 86.95259
```

KNN Regression

Because KNN Regression is based on distances, we will use the standardized data from the Lasso regression section and rerun it here to ensure proper standardization.

Standardization of data

```
fit_knn_std <- preProcess(aqi_health_train, method = "scale")
aqi_health_train_std <- predict(fit_knn_std, newdata = aqi_health_train)
aqi_health_test_std <- predict(fit_knn_std, newdata = aqi_health_test)
```

Model 1 Setup, Training Error, Testing Error

Model 1 was fit with all the predictor variables on the standardized data. MSE was used to evaluate the training and testing error.

```
# Fit the model
knn_model_1_fit <- knnreg(HealthImpactScore ~. -HealthImpactClass - RecordID -HealthImpactBinary, data =

# Evaluate the training error
y_train_hat <- predict(knn_model_1_fit, newdata = aqi_health_train_std)
training_error <- sum((aqi_health_train_std$HealthImpactScore - y_train_hat)^2)

# Evaluate the testing error
y_test_hat <- predict(knn_model_1_fit, newdata = aqi_health_test_std)
testing_error <- sum((aqi_health_test_std$HealthImpactScore - y_test_hat)^2)

table <- data.frame("Error" = c("Training", "Testing"),
                    "Value" = c(training_error, testing_error))
table
```

```
##      Error     Value
## 1 Training 1292.4442
## 2  Testing  358.6207
```

Model 2 Setup, Training Error, Testing Error

Model 2 was fit with the selected predictor variables on the standardized data. MSE was used to evaluate the training and testing error.

```r
# Fit the model
knn_model_2_fit <- knnreg(HealthImpactScore ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_train_st

# Evaluate the training error
y_train_hat <- predict(knn_model_2_fit, newdata = aqi_health_train_std)
training_error <- sum((aqi_health_train_std$HealthImpactScore - y_train_hat)^2)

# Evaluate the testing error
y_test_hat <- predict(knn_model_2_fit, newdata = aqi_health_test_std)
testing_error <- sum((aqi_health_test_std$HealthImpactScore - y_test_hat)^2)

table <- data.frame("Error" = c("Training", "Testing"),
                    "Value" = c(training_error, testing_error))
table
```

```
##      Error     Value
## 1 Training 281.35683
## 2  Testing  79.39528
```

Decision Trees

The decision trees are fit the same way as linear regression, and KNN regression. Model 1 will include all of the coefficients besides the RecordId and the HealthImpactClass. Model 2 will only include the variables identified by Lasso Regression.

Cross-validation for Decision Trees to predict HealthImpactScore

Similar to all the other techniques we present two different models in which we compute the cross validation.

- Model 1: All variables besides the RecordID, HealthImpactClass, and HealthImpactBinary
- Model 2: Variable only selected by the variables identified in the lasso regression: AQI + PM10 + PM2_5 + NO2 + O3

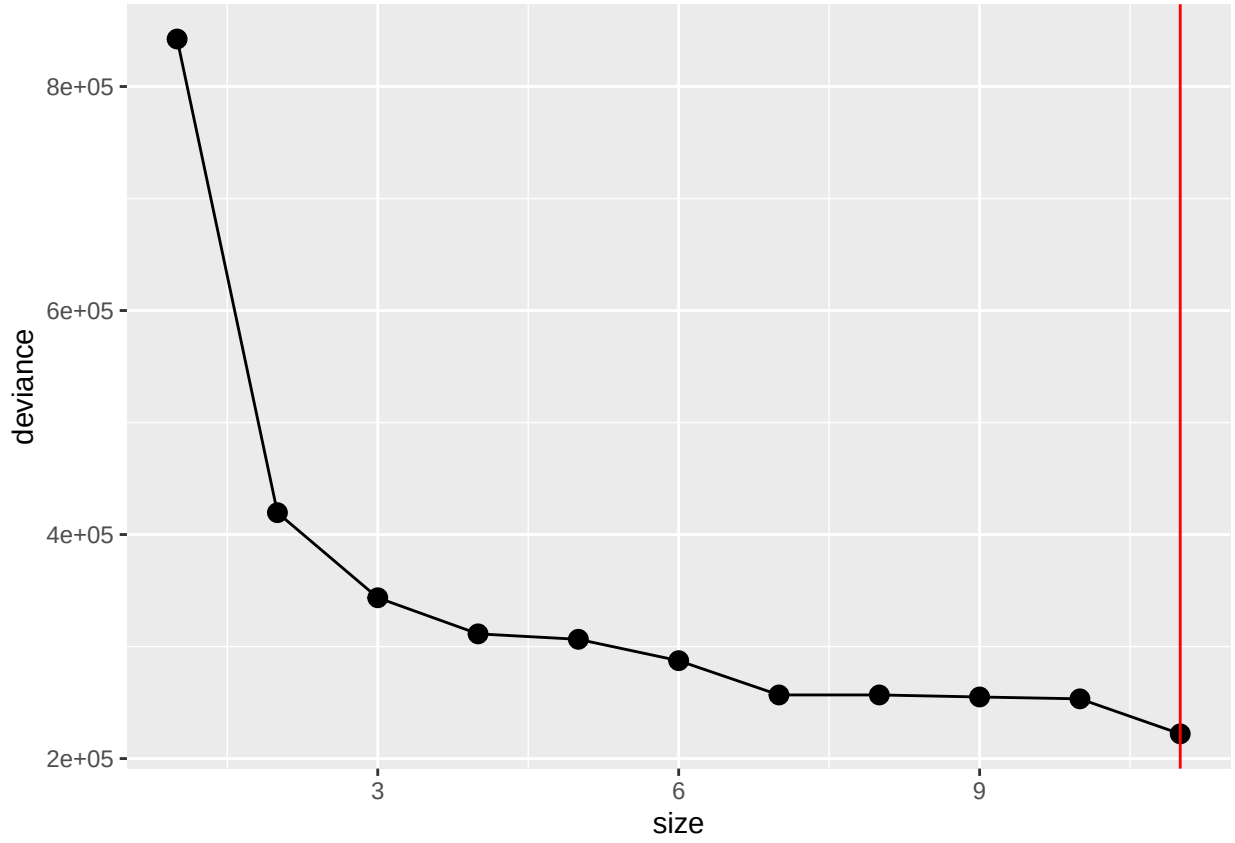
Model 1 Decision Tree - Cross Validation

The terminal node size was calculated using cross validation techniques. These terminal nodes were then used in the final pruned tree. From the pruned tree, the MSE was used to evaluate the training and testing error.

The seed was set to (0) for all models for reproducibility.

```r
#set seed to zero for reproducibility
set.seed(0)

#fit the tree
HIS_tree_model_1 <- tree(HealthImpactScore ~ . -RecordID - HealthImpactClass -HealthImpactBinary, data =

#use cross validation to understand the best terminal node size
cv_HIS <- cv.tree(HIS_tree_model_1)

#create a dataframe with the size and deviation
```

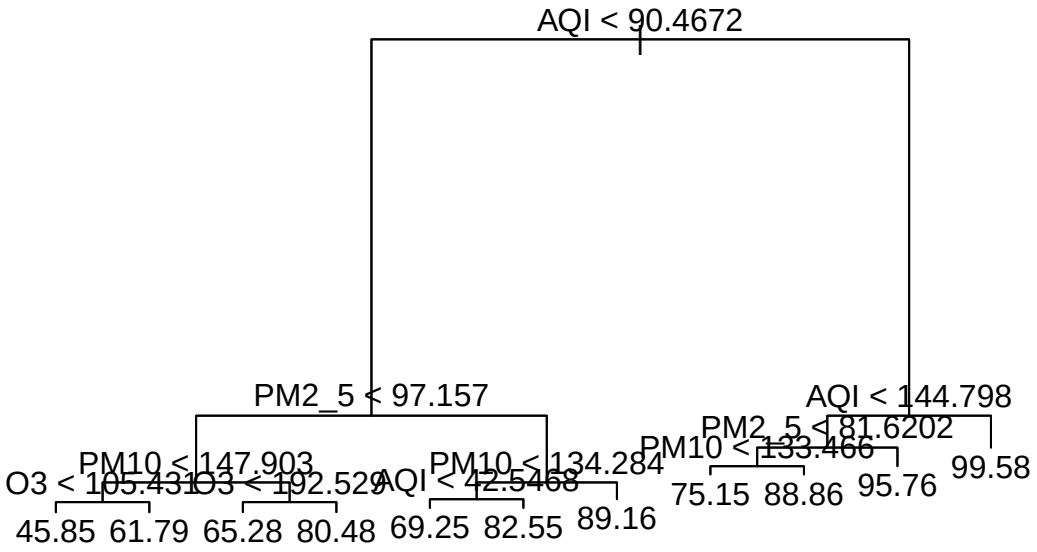
```r
cv_HIS_df <- data.frame(size = cv_HIS$size, deviance = cv_HIS$dev)

#find the best terminal node size
best_size <- cv_HIS$size[which.min(cv_HIS$dev)]

#plot to visually see the best size
ggplot(cv_HIS_df, mapping = aes(x = size, y = deviance)) +
  geom_point(size = 3) +
  geom_line() +
  geom_vline(xintercept = best_size, col = "red")
```



Visualizing the pruned regression tree

The pruned regression tree for Model 1 is visualized.

```r
HIS_tree_final <- prune.tree(HIS_tree_model_1, best = best_size) #The best_size identified by the above
plot(HIS_tree_final)
text(HIS_tree_final)
```



Compute the training and testing error

MSE was used to compute the training and testing error on the Model 1 pruned tree.

```
# Training Error
pred_HIS <- predict(HIS_tree_final, newdata = aqi_health_train)
train_error <- mean((pred_HIS - aqi_health_train$HealthImpactScore)^2)

#Testing Error
pred_HIS <- predict(HIS_tree_final, newdata = aqi_health_test)
test_error <- mean((pred_HIS - aqi_health_test$HealthImpactScore)^2)

table <- data.frame("Error Type" = c("Training", "Testing"),
                    "Value" = c(train_error, test_error))
table
```

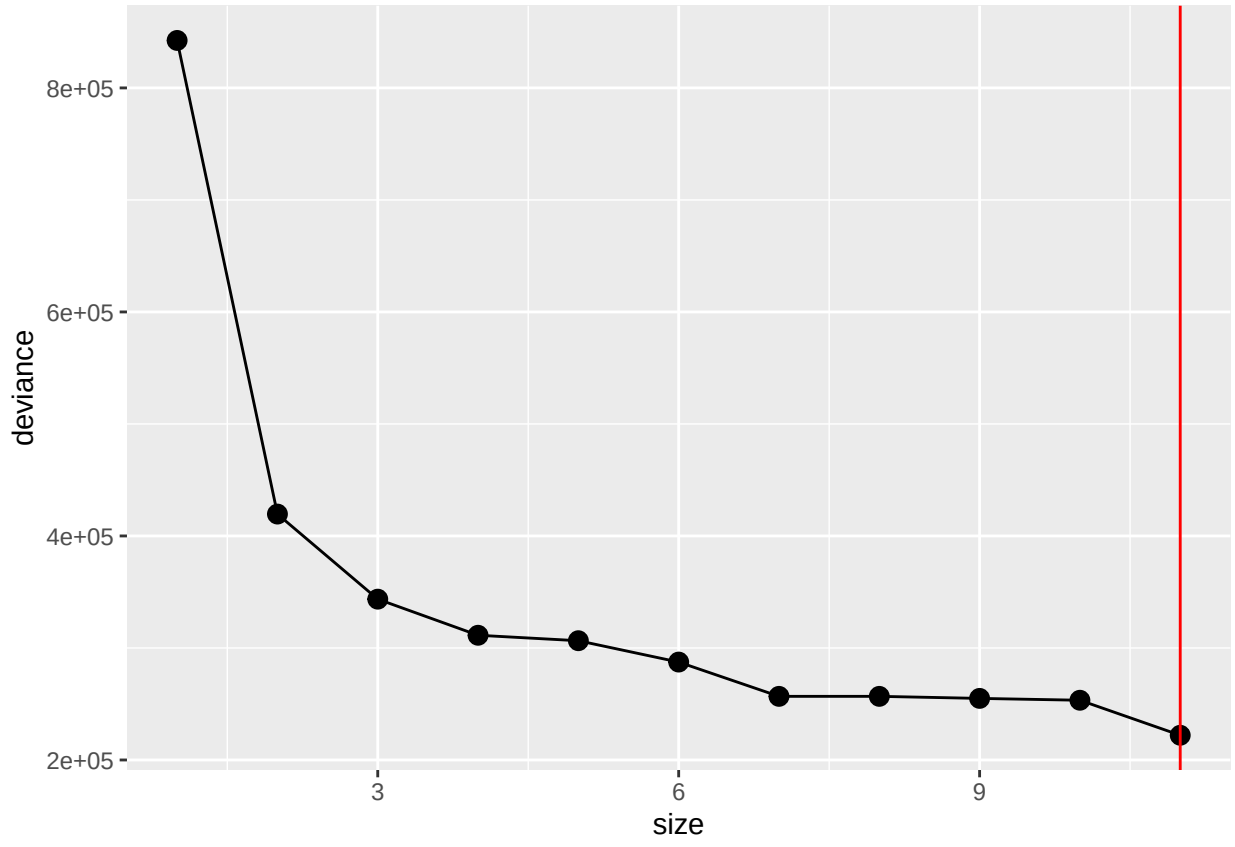
```
##   Error.Type    Value
## 1   Training 39.23049
## 2    Testing 45.22308
```

Model 2 Decision Tree - Cross Validation

The terminal node size was calculated using cross validation techniques for the predictors within Model 2. These terminal nodes were then used in the final pruned tree. From the pruned tree, the MSE was used to evaluate the training and testing error.

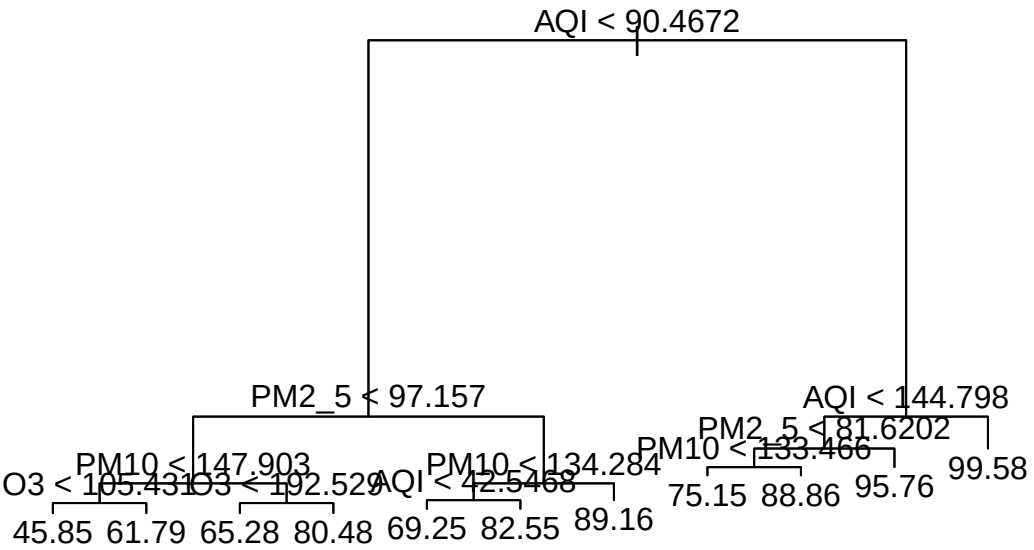
The seed was set to (0) for all models for reproducibility.

```r
#set seed to zero for reproducibility
set.seed(0)

#fit the tree to model 2
HIS_tree_model_2 <- tree(HealthImpactScore ~  AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_train)

#use cross validation to understand the best terminal node size
cv_HIS_ltd <- cv.tree(HIS_tree_model_2)

#create a dataframe with the size and deviation to find the best terminal node size
cv_HIS_ltd_df <- data.frame(size = cv_HIS_ltd$size, deviance = cv_HIS_ltd$dev)

#find the best terminal node size
best_size <- cv_HIS_ltd$size[which.min(cv_HIS_ltd$dev)]

#plot to visually see the best size
ggplot(cv_HIS_ltd_df, mapping = aes(x = size, y = deviance)) +
  geom_point(size = 3) +
  geom_line() +
  geom_vline(xintercept = best_size, col = "red")
```



Visualizing the pruned regression tree

```
HIS_tree_final_model_2 <- prune.tree(HIS_tree_model_2, best = best_size) #The subtree with best_size te
plot(HIS_tree_final_model_2)
text(HIS_tree_final_model_2)
```



Compute the training and test error

```
#Training Error
pred_HIS <- predict(HIS_tree_final_model_2, newdata = aqi_health_train)
train_error <- mean((pred_HIS - aqi_health_train$HealthImpactScore)^2)

#Testing Error
pred_HIS <- predict(HIS_tree_final_model_2, newdata = aqi_health_test)
test_error <- mean((pred_HIS - aqi_health_test$HealthImpactScore)^2)

table <- data.frame("Error Type" = c("Training", "Testing"),
                    "Value" = c(train_error, test_error))
table
```

```
##   Error.Type    Value
## 1   Training 39.23049
## 2    Testing 45.22308
```

Bagging

Similar to all the other techniques we present two different models in which we compute the bagging. For bagging we set p as equal to the number of predictors and specify 'mtry' as the number of variables randomly assigned as candidates for each split.

- Model 1: All variables besides the RecordID, HealthImpactClass, and HealthImpactBinary
- Model 2: Variable only selected by the variables identified in the lasso regression: AQI + PM10 + PM2_5 + NO2 + O3

Model 1 Setup, Training Error, and Testing Error

```
set.seed(0)

p <- ncol(aqi_health_train) - 4 #set p to the number of predictors. In this case it is 12 because we re
##Setting mtry = p for bagging
bag_aqi_health <- randomForest(HealthImpactScore ~. -RecordID -HealthImpactClass -HealthImpactBinary, da
bag_aqi_health
```

```
##
## Call:
##  randomForest(formula = HealthImpactScore ~ . - RecordID - HealthImpactClass -      HealthImpactBina
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 12
##
##           Mean of squared residuals: 8.505082
##                     % Var explained: 95.31
```

```
#Training Error
yhat_bag_train <- predict(bag_aqi_health, newdata = aqi_health_train)
bag_train_error <- mean((yhat_bag_train - aqi_health_train$HealthImpactScore)^2)

# Testing Error
yhat_bag_test <- predict(bag_aqi_health, newdata = aqi_health_test)
bag_test_error <- mean((yhat_bag_test - aqi_health_test$HealthImpactScore)^2)

bag_table_model_1 <- data.frame("Error" = c("Training","Testing"),
                                "Value" = c(bag_train_error, bag_test_error))
bag_table_model_1
```

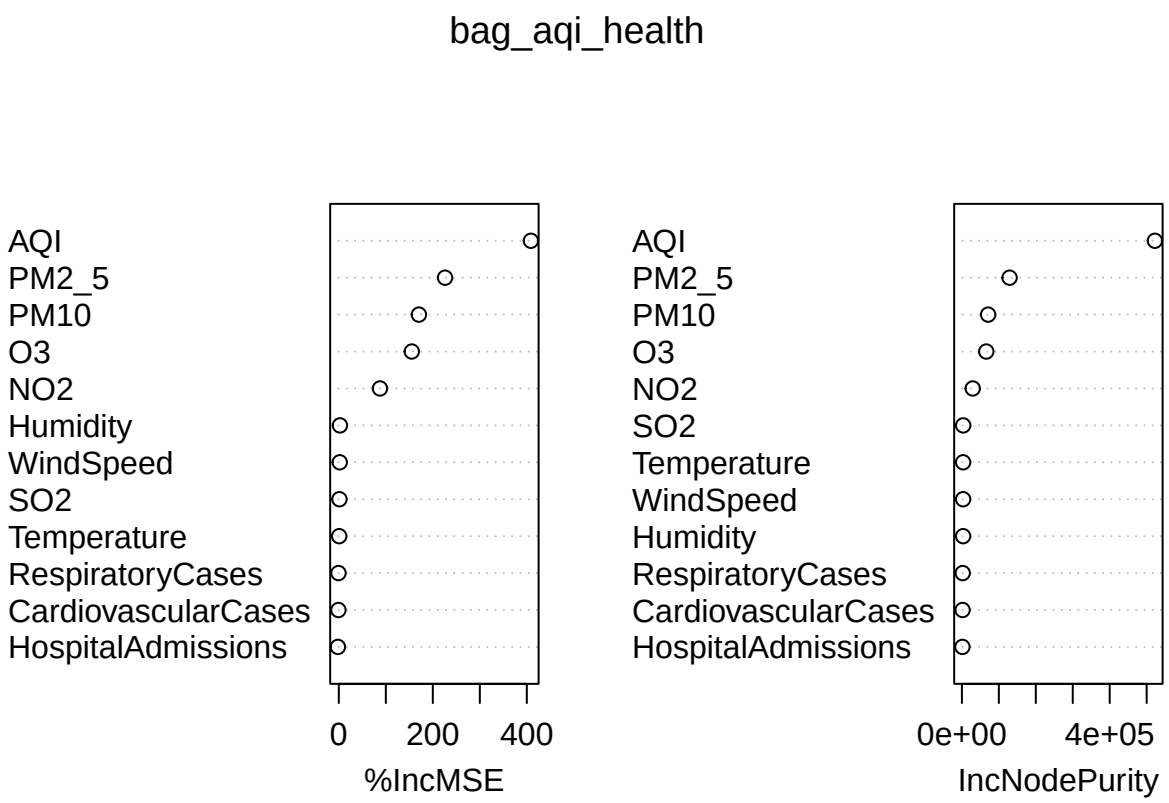
```
##      Error    Value
## 1 Training 1.397550
## 2  Testing 9.050325
```

```
importance(bag_aqi_health)
```

```
##                         %IncMSE IncNodePurity
## AQI                 408.6425221    522274.746
## PM10                170.3503416     70991.424
```

```
## PM2_5               226.0233923    129061.751
## NO2                  87.5294791     29323.326
## SO2                   1.3381347      3438.307
## O3                  155.2246586     65995.337
## Temperature           0.7616277      3302.082
## Humidity              2.4105160      3237.078
## WindSpeed             1.9780153      3244.916
## RespiratoryCases     -0.7030158      2130.208
## CardiovascularCases  -0.8517347      1782.936
## HospitalAdmissions   -1.9583021      1363.878
```

```
varImpPlot(bag_aqi_health)
```



Model 2 Setup, Training Error, and Testing Error

```
set.seed(0)

bag_aqi_health <- randomForest(HealthImpactScore ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_tra
bag_aqi_health
```

```
##
## Call:
##  randomForest(formula = HealthImpactScore ~ AQI + PM10 + PM2_5 +      NO2 + O3, data = aqi_health_tra
```

```
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 6.693066
##                     % Var explained: 96.31
```

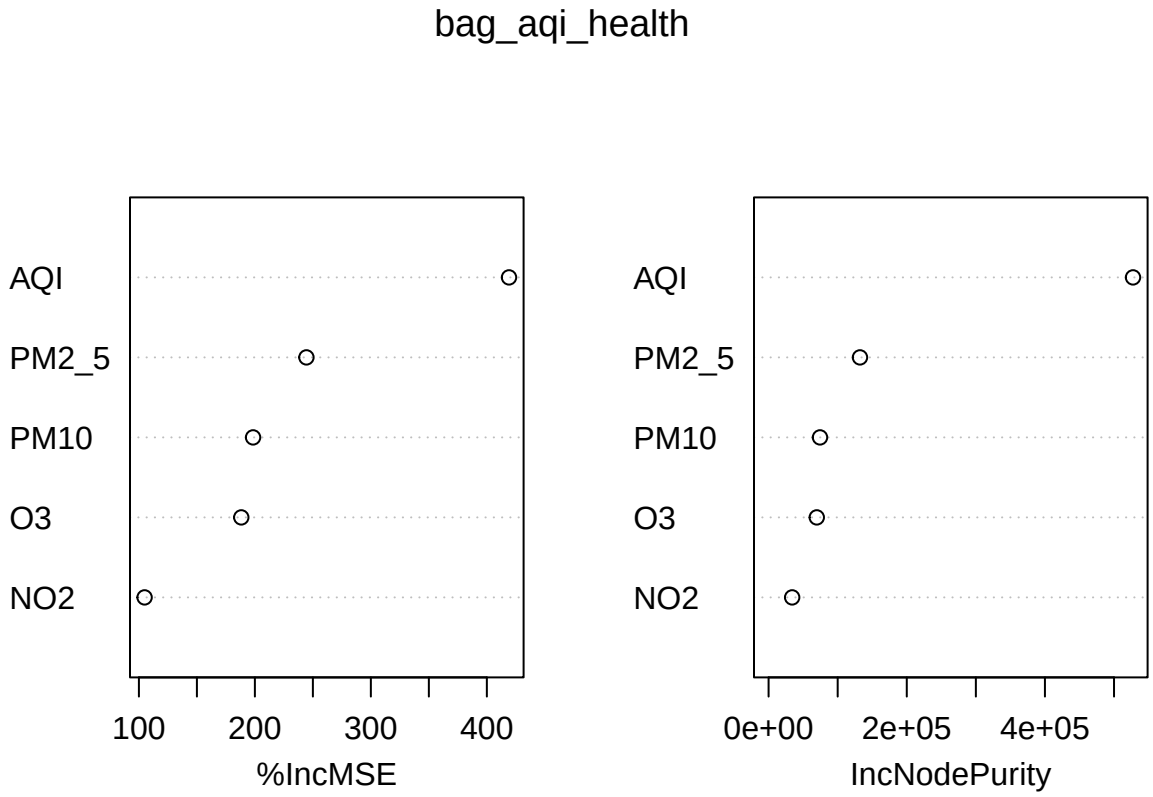
```r
#Training Error
yhat_bag_train <- predict(bag_aqi_health, newdata = aqi_health_train)
bag_train_error <- mean((yhat_bag_train - aqi_health_train$HealthImpactScore)^2)

# Testing Error
yhat_bag_test <- predict(bag_aqi_health, newdata = aqi_health_test)
bag_test_error <- mean((yhat_bag_test - aqi_health_test$HealthImpactScore)^2)

bag_table_model_2 <- data.frame("Error" = c("Training","Testing"),
                                "Value" = c(bag_train_error, bag_test_error))
bag_table_model_2
```

```
##      Error    Value
## 1 Training 1.162930
## 2  Testing 7.317396
```

```r
importance(bag_aqi_health)
```

```
##        %IncMSE IncNodePurity
## AQI   419.1240     527477.09
## PM10  198.4425      74425.74
## PM2_5 244.4332     132281.67
## NO2   104.8919      34137.42
## O3    188.2809      69741.29
```

```r
varImpPlot(bag_aqi_health)
```



Random Forest

Similar to all the other techniques we present two different models in which we compute for the Random Forest. Unlike bagging, we do not specificy "mtry = p" where p is the number of predictors.

- Model 1: All variables besides the RecordID, HealthImpactClass, and HealthImpactBinary
- Model 2: Variable only selected by the variables identified in the lasso regression: AQI + PM10 + PM2_5 + NO2 + O3

Model 1 Setup, Training Error, and Testing Error

```
set.seed(0)

rf_aqi_health_model_1 <- randomForest(HealthImpactScore ~. -RecordID -HealthImpactClass -HealthImpactBi
rf_aqi_health_model_1
```

```
##
## Call:
##  randomForest(formula = HealthImpactScore ~ . - RecordID - HealthImpactClass -      HealthImpactBina
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
```

```
##
##           Mean of squared residuals: 11.07616
##                     % Var explained: 93.89
```

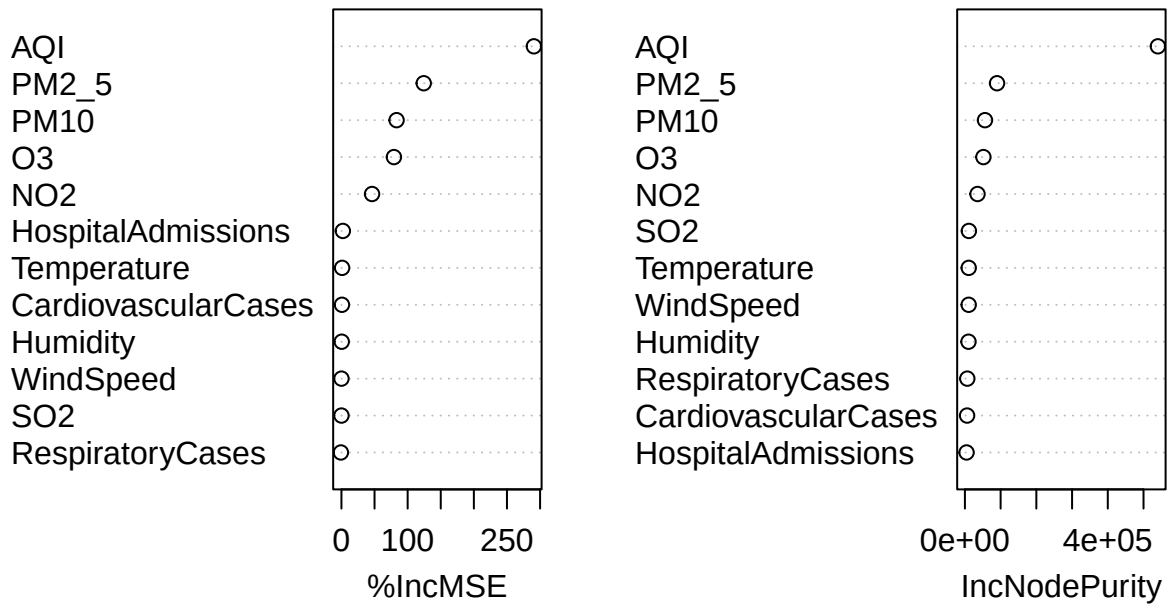
```r
#Training Error
yhat_rf_train <- predict(rf_aqi_health_model_1, newdata = aqi_health_train)
rf_train_error <- mean((yhat_rf_train - aqi_health_train$HealthImpactScore)^2)

# Testing Error
yhat_rf_test <- predict(rf_aqi_health_model_1, newdata = aqi_health_test)
rf_test_error <- mean((yhat_rf_test - aqi_health_test$HealthImpactScore)^2)

rf_table_model_1 <- data.frame("Error" = c("Training","Testing"),
                               "Value" = c(rf_train_error, rf_test_error))
rf_table_model_1
```

```
##      Error     Value
## 1 Training  1.998979
## 2  Testing 10.819194
```

```r
importance(rf_aqi_health_model_1)
```

```
##                         %IncMSE IncNodePurity
## AQI                 290.63750257    540218.510
## PM10                 83.29186371     56215.073
## PM2_5               124.30781877     89967.501
## NO2                  46.22771756     35354.061
## SO2                   0.03104027     11140.147
## O3                   79.24385946     51762.142
## Temperature           1.02232485     10709.798
## Humidity              0.45856142     10102.294
## WindSpeed             0.04567238     10603.359
## RespiratoryCases     -0.80507890      6749.820
## CardiovascularCases   0.76217424      6136.666
## HospitalAdmissions    2.13449286      4629.298
```

```r
varImpPlot(rf_aqi_health_model_1)
```



Model 2 Setup, Training Error, and Testing Error

```
set.seed(0)

rf_aqi_health_model_2 <- randomForest(HealthImpactScore ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_heal
rf_aqi_health_model_2
```

```
##
## Call:
##  randomForest(formula = HealthImpactScore ~ AQI + PM10 + PM2_5 +      NO2 + O3, data = aqi_health_tra
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 14.81576
##                     % Var explained: 91.82
```

```
#Training Error
yhat_rf_train <- predict(rf_aqi_health_model_2, newdata = aqi_health_train)
rf_train_error <- mean((yhat_rf_train - aqi_health_train$HealthImpactScore)^2)

# Testing Error
yhat_rf_test <- predict(rf_aqi_health_model_2, newdata = aqi_health_test)
```

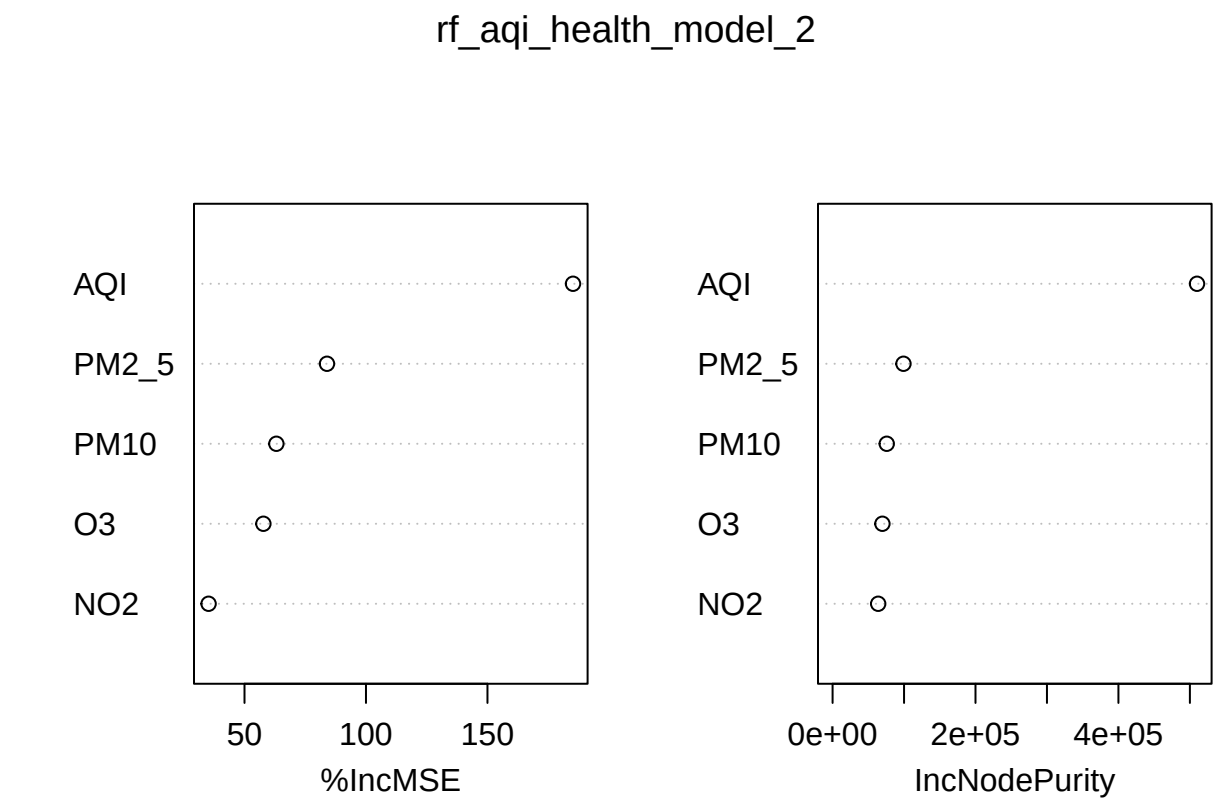
```r
rf_test_error <- mean((yhat_rf_test - aqi_health_test$HealthImpactScore)^2)

rf_table_model_1 <- data.frame("Error" = c("Training","Testing"),
                               "Value" = c(rf_train_error, rf_test_error))
rf_table_model_1
```

```
##      Error     Value
## 1 Training  3.307949
## 2  Testing 14.573818
```

```r
importance(rf_aqi_health_model_2)
```

```
##         %IncMSE IncNodePurity
## AQI   185.21903     509969.25
## PM10   63.11815      75808.20
## PM2_5  83.95087      99204.77
## NO2    35.21143      63838.77
## O3     57.77384      69716.40
```

```r
varImpPlot(rf_aqi_health_model_2)
```



Classification Methods

Preparing data

```
# Ensure outcome variable is a factor
aqi_health_train$HealthImpactBinary <- as.factor(aqi_health_train$HealthImpactBinary)
aqi_health_test$HealthImpactBinary <- as.factor(aqi_health_test$HealthImpactBinary)

# Look at distributions
table(aqi_health_train$HealthImpactClass)
```

```
##
##    0    1    2    3
## 3982  456  175   35
```

```
table(aqi_health_train$HealthImpactBinary)
```

```
##
##    0    1
##  666 3982
```

Logistic Regression

Two Logistic Regression models were fit to predict HealthImpactBinary. The first logistic regression model was fit with all of the predictors besides the variables RecordID, HealthImpactClass, and HealthImpactScore. RecordID is an identifier for each of the data's covariates. The second linear regression model was fit with only the predictors identified by the lasso regression. Lasso regression does automatic feature selection so the features that were not selected in the lasso were not included within the second model. From the lasso regression we see the variables AQI, PM10, PM2_5, NO2, and O3 are the most prominent features that effect the HealthImpactScore outcome. Both models will have their train and test score evaluated.

- Model 1: All coefficients besides RecordID, HealthImpactClass, and HealthImpactScore
- Model 2: Select coefficients from Lasso Regression: AQI, PM10, PM2_5, NO2, and O3

Logistic Regression: Model 1

Model Setup

```
glm_model_1_HealthImpactClass_fit <- glm(HealthImpactBinary ~ . -RecordID -HealthImpactClass -HealthImpa
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm_model_1_HealthImpactClass_fit)
```

```
##
## Call:
## glm(formula = HealthImpactBinary ~ . - RecordID - HealthImpactClass -
##     HealthImpactScore, family = binomial, data = aqi_health_train)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3471.2244 24947.3752  -0.139    0.889
## AQI                    12.6875    90.7450   0.140    0.889
## PM10                    4.2147    30.3742   0.139    0.890
## PM2_5                   8.3735    59.7903   0.140    0.889
## NO2                     4.3731    31.5905   0.138    0.890
## SO2                     2.2984    23.8938   0.096    0.923
## O3                      4.3022    31.4591   0.137    0.891
## Temperature            -1.6186    21.6927  -0.075    0.941
## Humidity                0.3454    11.6131   0.030    0.976
## WindSpeed              -2.4018    69.1609  -0.035    0.972
## RespiratoryCases        0.8506   117.8240   0.007    0.994
## CardiovascularCases     4.0629   139.9004   0.029    0.977
## HospitalAdmissions      3.6877    94.6751   0.039    0.969
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3.8196e+03  on 4647  degrees of freedom
## Residual deviance: 9.9913e-05  on 4635  degrees of freedom
## AIC: 26
##
## Number of Fisher Scoring iterations: 25
```

Training and Testing Error

```
# Training Error
glm_model1_train_prob <- predict(glm_model_1_HealthImpactClass_fit, type = "response")
glm_model1_train_class <- ifelse(glm_model1_train_prob > 0.5, 1, 0)
glm_model1_train_train_error <- mean(glm_model1_train_class != aqi_health_train$HealthImpactBinary)
glm_model1_train_train_error
```

```
## [1] 0
```

```
# Testing Error
glm_model1_test_prob <- predict(glm_model_1_HealthImpactClass_fit, newdata = aqi_health_test, type = "r
glm_model1_test_class <- ifelse(glm_model1_test_prob > 0.5, 1, 0)
glm_model1_test_train_error <- mean(glm_model1_test_class != aqi_health_test$HealthImpactBinary)
glm_model1_test_train_error
```

```
## [1] 0.006018917
```

```
log_model_1 <- data.frame("Error" = c("Training","Testing"),
                          "Value" = c(glm_model1_train_train_error, glm_model1_test_train_error))
log_model_1
```

```
##      Error       Value
## 1 Training 0.000000000
## 2  Testing 0.006018917
```

Logistic Regression: Model 2

Model Setup

```
glm_model_2_HealthImpactClass_fit <- glm(HealthImpactBinary ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(glm_model_2_HealthImpactClass_fit)
```

```
##
## Call:
## glm(formula = HealthImpactBinary ~ AQI + PM10 + PM2_5 + NO2 +
##     O3, family = binomial, data = aqi_health_train)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -87.46152    9.86495  -8.866   <2e-16 ***
## AQI           0.33362    0.03772   8.845   <2e-16 ***
## PM10          0.10957    0.01266   8.657   <2e-16 ***
## PM2_5         0.21961    0.02504   8.770   <2e-16 ***
## NO2           0.11358    0.01311   8.664   <2e-16 ***
## O3            0.11065    0.01261   8.778   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3819.60  on 4647  degrees of freedom
## Residual deviance:  164.26  on 4642  degrees of freedom
## AIC: 176.26
##
## Number of Fisher Scoring iterations: 14
```

Training and Testing Error

```
# Training Error
glm_model2_train_prob <- predict(glm_model_2_HealthImpactClass_fit, type = "response")
glm_model2_train_class <- ifelse(glm_model2_train_prob > 0.5, 1, 0)
glm_model2_train_train_error <- mean(glm_model2_train_class != aqi_health_train$HealthImpactBinary)
glm_model2_train_train_error
```

```
## [1] 0.007099828
```

```r
# Testing Error
glm_model2_test_prob <- predict(glm_model_2_HealthImpactClass_fit, newdata = aqi_health_test, type = "r
glm_model2_test_class <- ifelse(glm_model2_test_prob > 0.5, 1, 0)
glm_model2_test_train_error <- mean(glm_model2_test_class != aqi_health_test$HealthImpactBinary)
glm_model2_test_train_error
```

```
## [1] 0.01203783
```

```r
log_model_2 <- data.frame("Error" = c("Training","Testing"),
                          "Value" = c(glm_model2_train_train_error, glm_model2_test_train_error))
log_model_2
```

```
##      Error       Value
## 1 Training 0.007099828
## 2  Testing 0.012037833
```

KNN Classification

Standardize data

```r
# Identify test/train predictors and response variables
train_predictors <- aqi_health_train[, !(names(aqi_health_train) %in% c("HealthImpactBinary", "HealthIm
train_response <- aqi_health_train$HealthImpactBinary

test_predictors <- aqi_health_test[, !(names(aqi_health_test) %in% c("HealthImpactBinary", "HealthImpac
test_response <- aqi_health_test$HealthImpactBinary

# Standardize
fit_knn_std <- preProcess(train_predictors, method = c("center", "scale"))
train_predictors_std <- predict(fit_knn_std, train_predictors)
test_predictors_std <- predict(fit_knn_std, test_predictors)

# recombine HealthImpactBinary outcome variable with standardized predictors
aqi_health_train_std_class <- cbind(train_predictors_std, HealthImpactBinary = train_response)
aqi_health_test_std_class <- cbind(test_predictors_std, HealthImpactBinary = test_response)
```

KNN Classification Model 1: All predictors except RecordID, HealthImpactClass, and HealthImpactScore (Model 1)

Model 1 was fit with all the predictor variables on the standardized data. Classification error was calculated for the testing and training datasets.

```r
### KNN Model 1 Setup, Training Error, Testing Error

# Fit the model
knn_class_mod1 <- knn3(HealthImpactBinary ~. , data = aqi_health_train_std_class, k = 10)

# Evaluate the training error
knn_mod1_train_pred <- predict(knn_class_mod1, newdata = aqi_health_train_std_class, type = "class")
```

```r
knn_mod1_train_err <- mean(knn_mod1_train_pred != aqi_health_train_std_class$HealthImpactBinary)

# Evaluate the testing error
knn_mod1_test_pred <- predict(knn_class_mod1, newdata = aqi_health_test_std_class, type = "class")
knn_mod1_test_err <- mean(knn_mod1_test_pred != aqi_health_test_std_class$HealthImpactBinary)

table_knn1 <- data.frame("Error" = c("Training", "Testing"),
                         "Value" = c(knn_mod1_train_err, knn_mod1_test_err))
table_knn1
```

```
##      Error      Value
## 1 Training 0.07142857
## 2  Testing 0.08168530
```

KNN Classification Model 2: AQI, PM10, PM2_5, NO2, and O3 predictors ONLY (Model 2)

Model 2 was fit with selected predictor variables on the standardized data. Classification error was calculated for the testing and training datasets.

```r
### KNN Model 2 Setup, Training Error, Testing Error

# Fit the model
knn_class_mod2 <- knn3(HealthImpactBinary ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_train_std_

# Evaluate the training error
knn_mod2_train_pred <- predict(knn_class_mod2, newdata = aqi_health_train_std_class, type = "class")
knn_mod2_train_err <- mean(knn_mod2_train_pred != aqi_health_train_std_class$HealthImpactBinary)

# Evaluate the testing error
knn_mod2_test_pred <- predict(knn_class_mod2, newdata = aqi_health_test_std_class, type = "class")
knn_mod2_test_err <- mean(knn_mod2_test_pred != aqi_health_test_std_class$HealthImpactBinary)

table_knn2 <- data.frame("Error" = c("Training", "Testing"),
                         "Value" = c(knn_mod2_train_err, knn_mod2_test_err))
table_knn2
```

```
##      Error      Value
## 1 Training 0.02775387
## 2  Testing 0.03181427
```

Decision Trees: Classification

The decision classification trees are fit the same way as logistic regression and KNN classification. Model 1 will include all of the coefficients besides the RecordId, HealthImpactClass, and HealthImpactScore. Model 2 will only include the variables identified by Lasso Regression.

Cross-validation for Decision Trees to predict HealthImpactBinary

Similar to all the other techniques we present two different models in which we compute the cross validation.

- Model 1: All variables besides the RecordID, HealthImpactClass, and HealthImpactScore

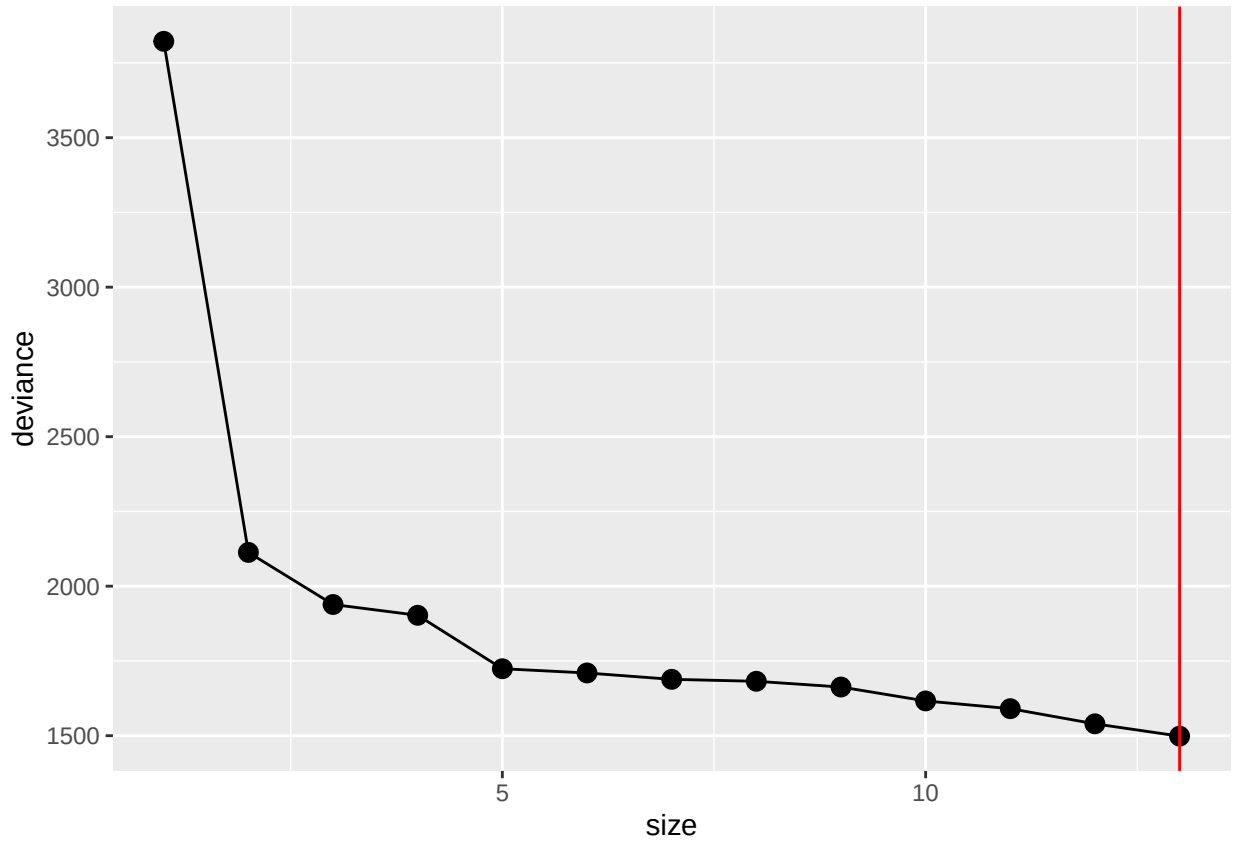
- Model 2: Variable only selected by the variables identified in the lasso regression: AQI + PM10 + PM2_5 + NO2 + O3

Model 1 Decision Classification Tree - Cross Validation

The terminal node size was calculated using cross validation techniques. These terminal nodes were then used in the final pruned tree. From the pruned tree, the classification error for the testing and training datasets was calculated.

The seed was set to (0) for all models for reproducibility.

```
#set seed to zero for reproducibility
set.seed(0)

#fit the tree
HIB_tree_model_1 <- tree(HealthImpactBinary ~ . -RecordID -HealthImpactClass -HealthImpactScore, data =

#use cross validation to understand the best terminal node size
cv_HIB <- cv.tree(HIB_tree_model_1)

#create a dataframe with the size and deviation
cv_HIB_df <- data.frame(size = cv_HIB$size, deviance = cv_HIB$dev)

#find the best terminal node size
best_size_HIB1 <- cv_HIB$size[which.min(cv_HIB$dev)]

#plot to visually see the best size
ggplot(cv_HIB_df, mapping = aes(x = size, y = deviance)) +
  geom_point(size = 3) +
  geom_line() +
  geom_vline(xintercept = best_size_HIB1, col = "red")
```



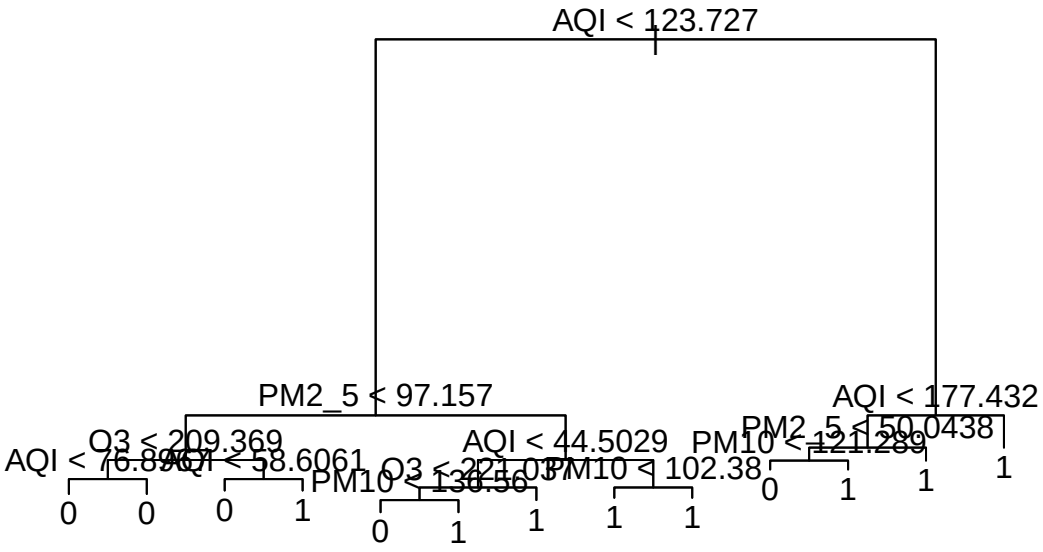

```
cat('CV leads to the optimal tree size as ', best_size_HIB1,'\n')
```

```
## CV leads to the optimal tree size as  13
```

Visualizing the pruned regression tree

The pruned regression tree for Model 1 is visualized.

```
HIB_tree_final <- prune.tree(HIB_tree_model_1, best = best_size_HIB1) #The subtree with best_size termi
plot(HIB_tree_final)
text(HIB_tree_final)
```



Compute the training and testing error

Compute the classification training and testing error on the Model 1 pruned tree.

```
# Training Error
pred_HIB1 <- predict(HIB_tree_final, newdata = aqi_health_train, type = "class")
train_error_HIB1 <- mean(pred_HIB1 != aqi_health_train$HealthImpactBinary)

#Testing Error
pred_HIB1_test <- predict(HIB_tree_final, newdata = aqi_health_test, type = "class")
test_error_HIB1 <- mean(pred_HIB1_test != aqi_health_test$HealthImpactBinary)

HIB1_errors <- data.frame("Error Type" = c("Training", "Testing"),
                  "Value" = c(train_error_HIB1, test_error_HIB1))
HIB1_errors
```

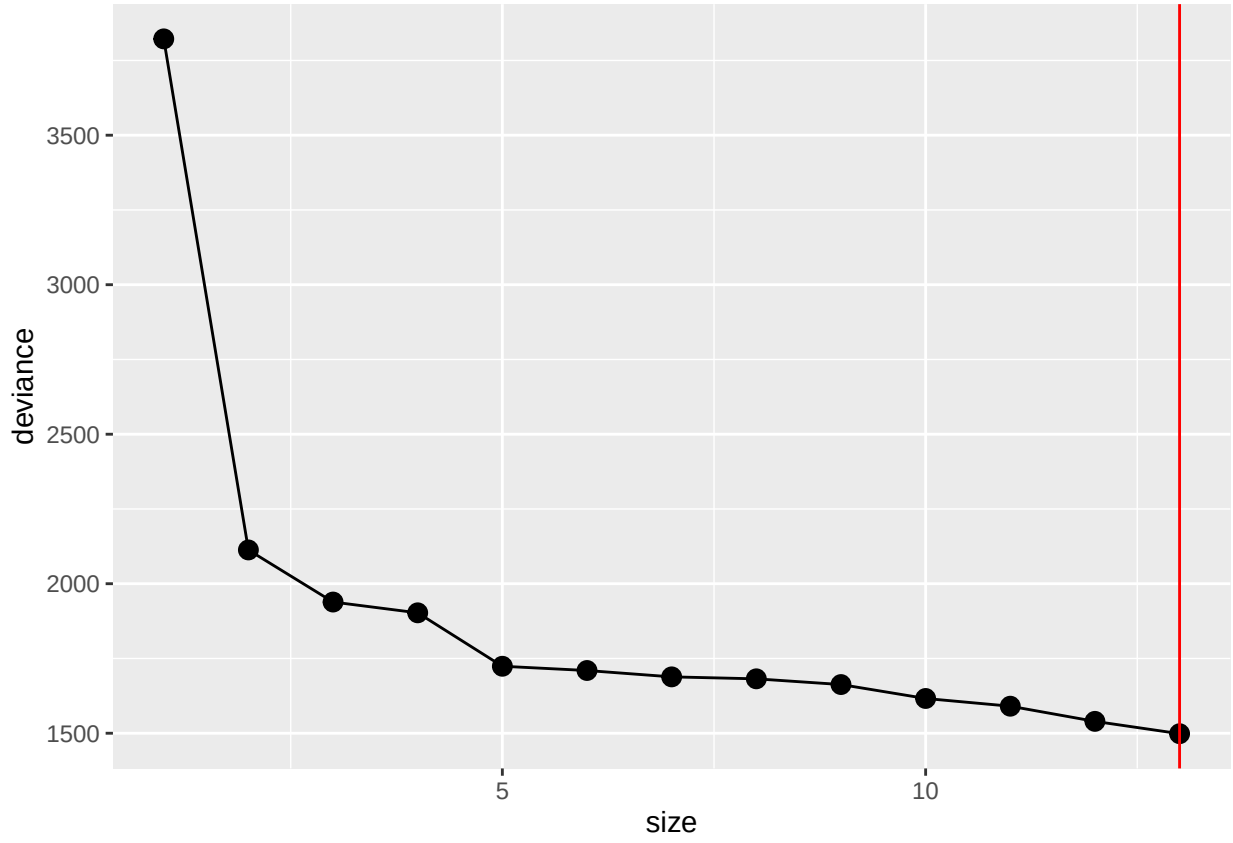
```
##   Error.Type      Value
## 1   Training 0.05271084
## 2    Testing 0.05331040
```

Model 2 Decision Classification Tree - Cross Validation

The terminal node size was calculated using cross validation techniques for the predictors within Model 2. These terminal nodes were then used in the final pruned tree. From the pruned tree, the training and testing classification errors were calculated.

The seed was set to (0) for all models for reproducibility.

```
#set seed to zero for reproducibility
set.seed(0)

#fit the tree
HIB_tree_model_2 <- tree(HealthImpactBinary ~  AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_train)

#use cross validation to understand the best terminal node size
cv_HIB_ltd <- cv.tree(HIB_tree_model_2)

#create a dataframe with the size and deviation
cv_HIB_ltd_df <- data.frame(size = cv_HIB_ltd$size, deviance = cv_HIB_ltd$dev)

#find the best terminal node size
best_size_HIB2 <- cv_HIB_ltd$size[which.min(cv_HIB_ltd$dev)]

#plot to visually see the best size
ggplot(cv_HIB_ltd_df, mapping = aes(x = size, y = deviance)) +
  geom_point(size = 3) +
  geom_line() +
  geom_vline(xintercept = best_size_HIB2, col = "red")
```



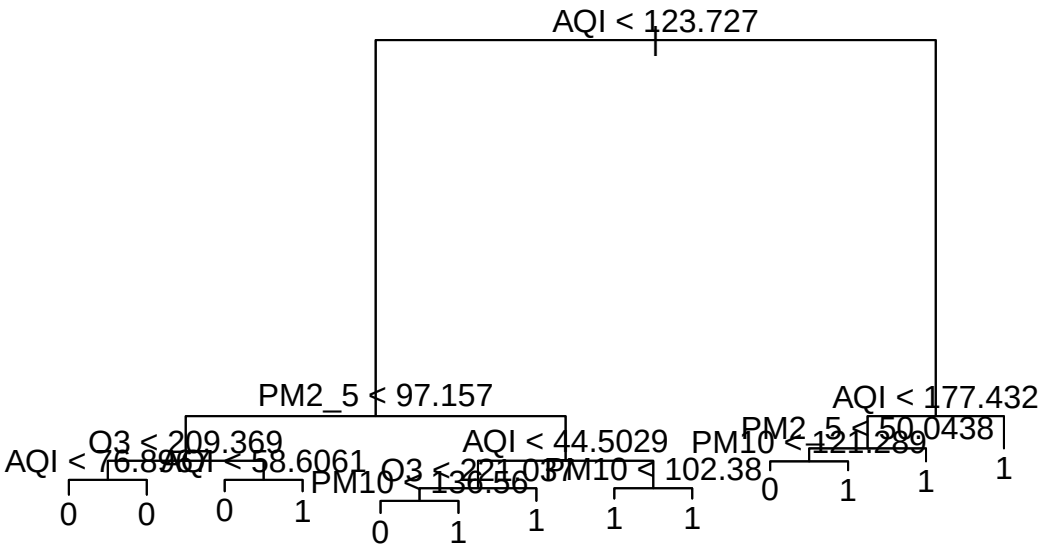

```
cat('CV leads to the optimal tree size as ', best_size_HIB2,'\n')
```

```
## CV leads to the optimal tree size as  13
```

Visualizing the pruned regression tree

The pruned regression tree for Model 2 is visualized.

```
HIB_tree_final_model_2 <- prune.tree(HIB_tree_model_2, best = best_size_HIB2) #The subtree with best_si
plot(HIB_tree_final_model_2)
text(HIB_tree_final_model_2)
```



Compute the training and testing error

Compute the classification training and testing error on the Model 2 pruned tree.

```
# Training Error
pred_HIB2 <- predict(HIB_tree_final_model_2, newdata = aqi_health_train, type = "class")
train_error_HIB2 <- mean(pred_HIB2 != aqi_health_train$HealthImpactBinary)

#Testing Error
pred_HIB2_test <- predict(HIB_tree_final_model_2, newdata = aqi_health_test, type = "class")
test_error_HIB2 <- mean(pred_HIB2_test != aqi_health_test$HealthImpactBinary)

HIB2_errors <- data.frame("Error Type" = c("Training", "Testing"),
                  "Value" = c(train_error_HIB2, test_error_HIB2))
HIB2_errors
```

```
##   Error.Type      Value
```

```
## 1   Training 0.05271084
## 2    Testing 0.05331040
```

Bagging

Similar to all the other techniques we present two different models in which we compute the bagging. For bagging we set p as equal to the number of predictors and specify 'mtry' as the number of variables randomly assigned as candidates for each split.

- Model 1: All variables besides the RecordID, HealthImpactClass, and HealthImpactScore
- Model 2: Variable only selected by the variables identified in the lasso regression: AQI + PM10 + PM2_5 + NO2 + O3

Model 1 Setup, Training Error, and Testing Error

```
set.seed(0)

p <- ncol(aqi_health_train) - 4 #set p to the number of predictors. In this case it is 12 because we re
##Setting mtry = p for bagging

bag_hib1 <- randomForest(HealthImpactBinary ~. -RecordID -HealthImpactClass -HealthImpactScore, data = a
bag_hib1
```

```
##
## Call:
##  randomForest(formula = HealthImpactBinary ~ . - RecordID - HealthImpactClass -      HealthImpactSco
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 12
##
##         OOB estimate of  error rate: 3.14%
## Confusion matrix:
##     0    1 class.error
## 0 575   91  0.13663664
## 1  55 3927  0.01381215
```

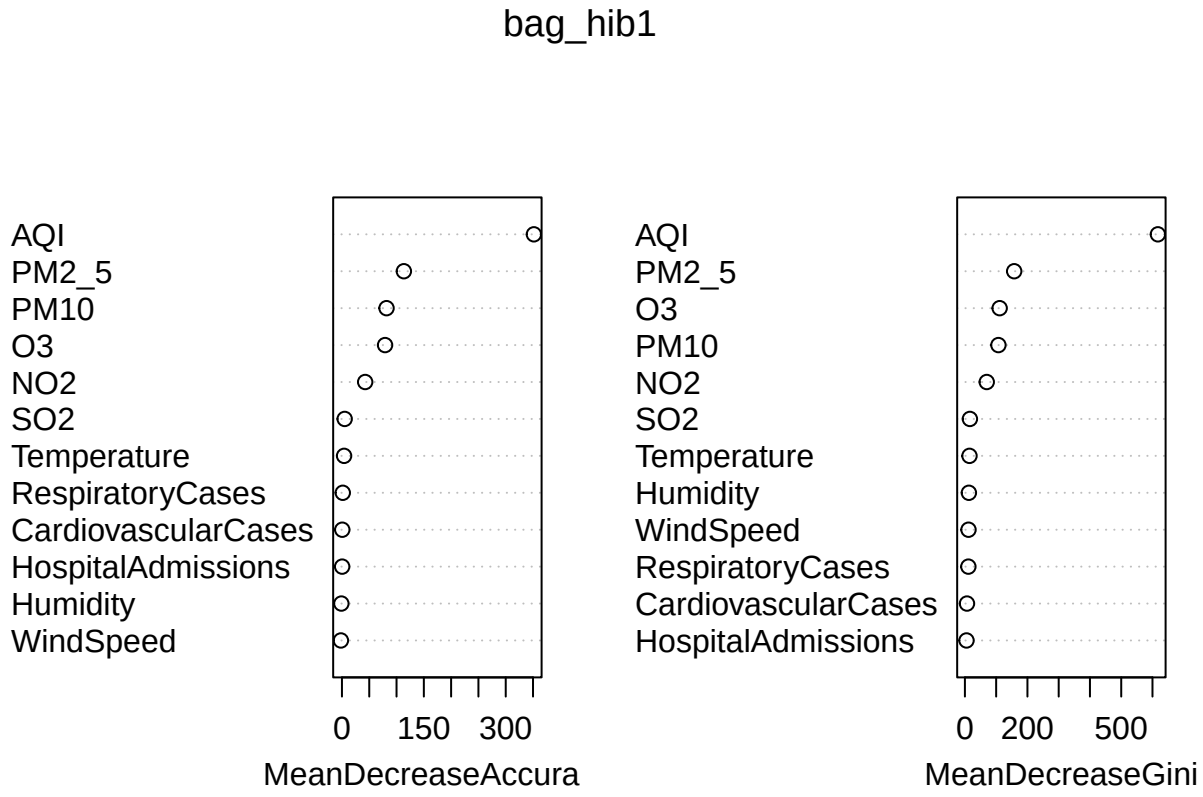
```
#Training Error
yhat_bag_train_class1 <- predict(bag_hib1, newdata = aqi_health_train, type = "class")
bag_train_error1 <- mean(yhat_bag_train_class1 != aqi_health_train$HealthImpactBinary)

# Testing Error
yhat_bag_test_class1 <- predict(bag_hib1, newdata = aqi_health_test)
bag_test_error1 <- mean(yhat_bag_test_class1 != aqi_health_test$HealthImpactBinary)

bag_hib1_errors <- data.frame("Error" = c("Training","Testing"),
                              "Value" = c(bag_train_error1, bag_test_error1))
bag_hib1_errors
```

```
##      Error      Value
## 1 Training 0.00000000
## 2  Testing 0.02321582
```

```r
importance(bag_hib1)
```

```
##                              0           1 MeanDecreaseAccuracy
## AQI                 370.9130145 256.4040694           351.5724957
## PM10                 64.4498798  53.4904773            81.6545794
## PM2_5                89.8810790  82.9800422           113.4124327
## NO2                  36.5998496  26.1248113            42.5627232
## SO2                   2.0111840   4.9643723             4.9945734
## O3                   55.7994789  57.5077524            78.9750005
## Temperature           3.9837606   1.4275743             3.9123015
## Humidity             -0.3070747  -1.1473464            -1.0474762
## WindSpeed            -0.5474278  -2.0837877            -1.9070614
## RespiratoryCases     -0.4257376   2.3321964             1.5171052
## CardiovascularCases  -2.9550630   3.3028846             0.5918332
## HospitalAdmissions    0.2316311   0.3602387             0.3951752
##                     MeanDecreaseGini
## AQI                       617.251738
## PM10                      107.257211
## PM2_5                     157.638971
## NO2                        69.855315
## SO2                        15.975826
## O3                        111.174642
## Temperature                14.676949
## Humidity                   12.714567
## WindSpeed                  11.511240
## RespiratoryCases           11.104305
## CardiovascularCases         6.629373
## HospitalAdmissions          5.133111
```

```r
varImpPlot(bag_hib1)
```



Model 2 Setup, Training Error, and Testing Error

```
set.seed(0)

# Model Setup
bag_hib2 <- randomForest(HealthImpactBinary ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_train, m
bag_hib2
```

```
##
## Call:
##  randomForest(formula = HealthImpactBinary ~ AQI + PM10 + PM2_5 +      NO2 + O3, data = aqi_health_t
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 3.03%
## Confusion matrix:
##     0    1 class.error
## 0 577   89  0.13363363
## 1  52 3930  0.01305876
```

```
#Training Error
yhat_bag_train_class2 <- predict(bag_hib2, newdata = aqi_health_train, type = "class")
```

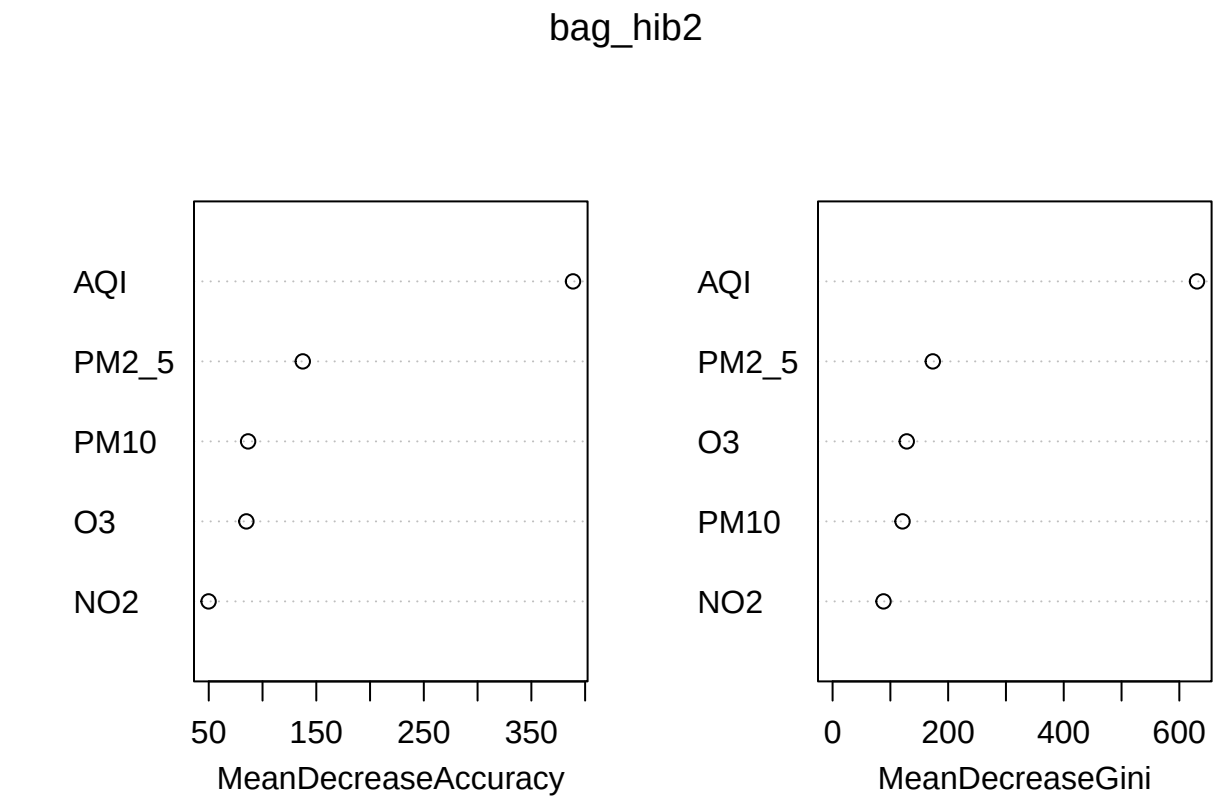
```r
bag_train_error2 <- mean(yhat_bag_train_class2 != aqi_health_train$HealthImpactBinary)

# Testing Error
yhat_bag_test_class2 <- predict(bag_hib2, newdata = aqi_health_test)
bag_test_error2 <- mean(yhat_bag_test_class2 != aqi_health_test$HealthImpactBinary)

bag_hib2_errors <- data.frame("Error" = c("Training","Testing"),
                              "Value" = c(bag_train_error2, bag_test_error2))
bag_hib2_errors
```

```
##      Error      Value
## 1 Training 0.00000000
## 2  Testing 0.02407567
```

```r
importance(bag_hib2)
```

```
##               0         1 MeanDecreaseAccuracy MeanDecreaseGini
## AQI   408.74734 277.65714            388.75508         630.6135
## PM10   69.60908  53.72790             86.60505         121.0591
## PM2_5 103.91337  95.28430            137.43685         173.3960
## NO2    40.55034  30.48472             49.83209          88.1375
## O3     65.05096  59.61832             84.98232         128.3402
```

```r
varImpPlot(bag_hib2)
```



Random Forest

Similar to all the other techniques we present two different models in which we compute for the Random Forest. Unlike bagging, we do not specificy "mtry = p" where p is the number of predictors.

- Model 1: All variables besides the RecordID, HealthImpactClass, and HealthImpactScore
- Model 2: Variable only selected by the variables identified in the lasso regression: AQI + PM10 + PM2_5 + NO2 + O3

Model 1 Setup, Training Error, and Testing Error

```
set.seed(0)

rf_hib_model_1 <- randomForest(HealthImpactBinary ~. -RecordID -HealthImpactClass -HealthImpactScore, da
rf_hib_model_1
```

```
##
## Call:
##  randomForest(formula = HealthImpactBinary ~ . - RecordID - HealthImpactClass -      HealthImpactSco
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 3.16%
## Confusion matrix:
##     0    1 class.error
## 0 560  106  0.15915916
## 1  41 3941  0.01029633
```

```
#Training Error
yhat_rf_train_class1 <- predict(rf_hib_model_1, newdata = aqi_health_train, type = "class")
rf_train_error1 <- mean(yhat_rf_train_class1 != aqi_health_train$HealthImpactBinary)

# Testing Error
yhat_rf_test_class1 <- predict(rf_hib_model_1, newdata = aqi_health_test, type = "class")
rf_test_error1 <- mean(yhat_rf_test_class1 != aqi_health_test$HealthImpactBinary)

rf_hib1_errors <- data.frame("Error" = c("Training","Testing"),
                             "Value" = c(rf_train_error1, rf_test_error1))
rf_hib1_errors
```

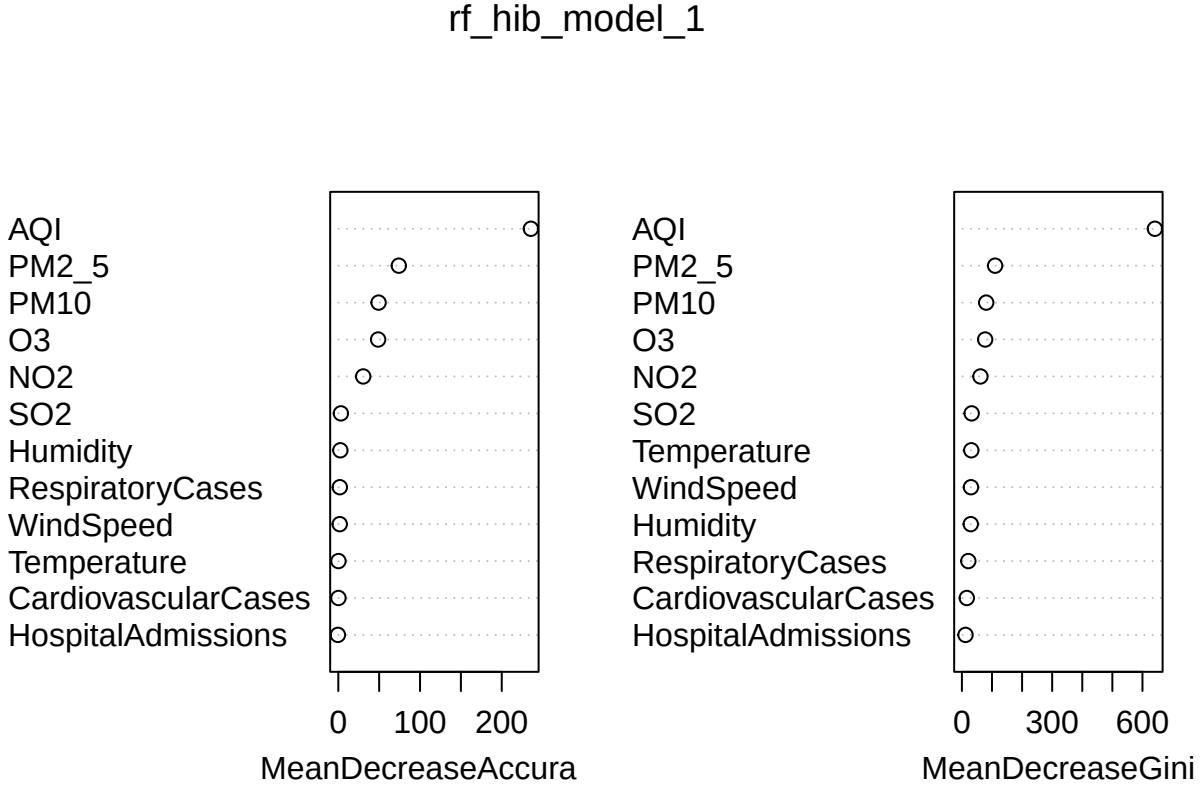
```
##      Error      Value
## 1 Training 0.00000000
## 2  Testing 0.02407567
```

```
importance(rf_hib_model_1)
```

```
##                              0           1 MeanDecreaseAccuracy
## AQI                232.9826466 191.7346944          235.6687378
## PM10                42.0957722  35.2019193           49.2739020
```

```
## PM2_5                59.1754417  60.4479410           74.0227852
## NO2                  27.9467785  17.4667649           30.4564730
## SO2                   0.6150009   3.4048277            3.1551816
## O3                   38.8802466  35.9484244           48.6969610
## Temperature           0.4182126  -0.2160922            0.1775970
## Humidity              2.8321712   0.7705320            2.4936177
## WindSpeed             1.9684379   0.5639951            1.7530409
## RespiratoryCases      0.4924917   1.9917566            1.8598056
## CardiovascularCases  -1.0610503   0.9625386            0.1604442
## HospitalAdmissions    0.9500071  -1.4150004           -0.5067760
##                     MeanDecreaseGini
## AQI                       641.26896
## PM10                       80.76921
## PM2_5                     110.33872
## NO2                        61.56821
## SO2                        32.41578
## O3                         77.23973
## Temperature                31.23478
## Humidity                   29.40660
## WindSpeed                  30.11892
## RespiratoryCases           21.26579
## CardiovascularCases        16.26035
## HospitalAdmissions         12.00743
```

```
varImpPlot(rf_hib_model_1)
```



Model 2 Setup, Training Error, and Testing Error

```
set.seed(0)

rf_hib_model_2 <- randomForest(HealthImpactBinary ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_health_tra
rf_hib_model_2
```

```
##
## Call:
##  randomForest(formula = HealthImpactBinary ~ AQI + PM10 + PM2_5 +      NO2 + O3, data = aqi_health_t
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 2.82%
## Confusion matrix:
##     0    1 class.error
## 0 586   80  0.12012012
## 1  51 3931  0.01280763
```

```
#Training Error
yhat_rf_train_class2 <- predict(rf_hib_model_2, newdata = aqi_health_train, type = "class")
rf_train_error2 <- mean(yhat_rf_train_class2 != aqi_health_train$HealthImpactBinary)

# Testing Error
yhat_rf_test_class2 <- predict(rf_hib_model_2, newdata = aqi_health_test, type = "class")
rf_test_error2 <- mean(yhat_rf_test_class2 != aqi_health_test$HealthImpactBinary)

rf_hib2_errors <- data.frame("Error" = c("Training","Testing"),
                             "Value" = c(rf_train_error2, rf_test_error2))
rf_hib2_errors
```

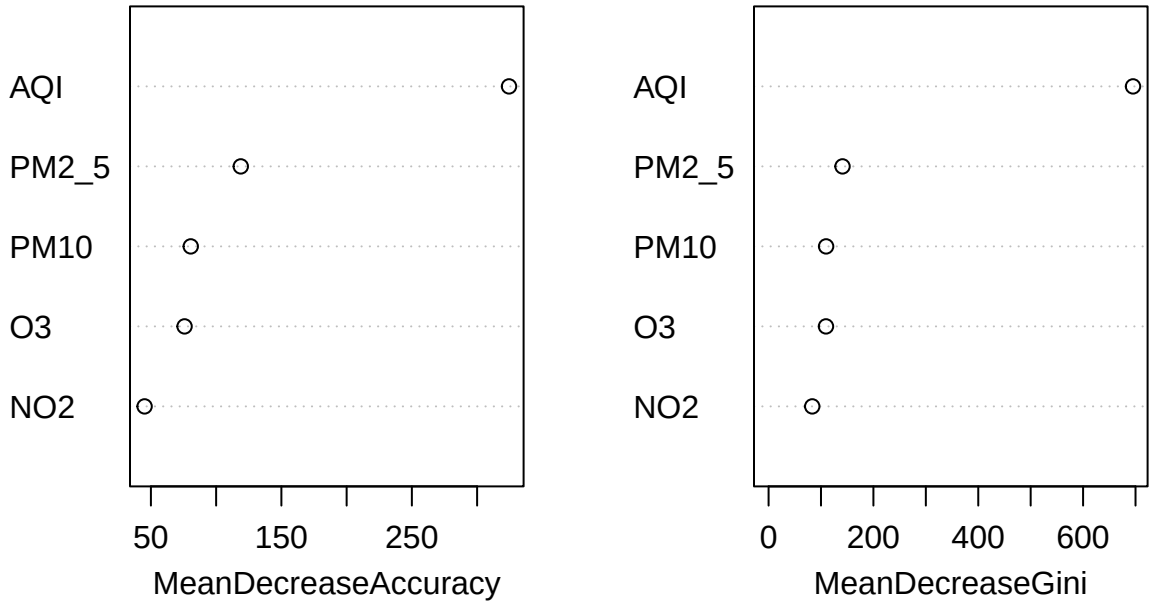
```
##      Error      Value
## 1 Training 0.00000000
## 2  Testing 0.02235598
```

```
importance(rf_hib_model_2)
```

```
##               0         1 MeanDecreaseAccuracy MeanDecreaseGini
## AQI   353.66521 240.90786            324.44261        695.23820
## PM10   63.55809  51.44001             80.54863        109.74554
## PM2_5  88.46385  86.49701            118.88545        141.04469
## NO2    38.79882  27.77278             45.15954         83.31214
## O3     56.81093  57.63154             75.79561        109.48371
```

```
varImpPlot(rf_hib_model_2)
```

rf_hib_model_2




```
rf_aqi_health_model_2 <- randomForest(HealthImpactScore ~ AQI + PM10 + PM2_5 + NO2 + O3, data = aqi_hea
rf_aqi_health_model_2
```

```
##
## Call:
##  randomForest(formula = HealthImpactScore ~ AQI + PM10 + PM2_5 +      NO2 + O3, data = aqi_health_tra
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 15.21548
##                     % Var explained: 91.6
```