INFO 411 - AIR POLLUTION

Name	Student No.	Contribution %
Angel Dela Cruz	7350211	100%
Brandon Louis Chia	7896530	100%
Sivathorn Siralert	6738564	100%
Thang Yu Ting Kym	7766270	100%
Li Biyang	7352670	100%
Felicia Gondo	7433086	100%

Table of contents

01 PRE-PROCESSING

02 RANDOM FOREST

03 LINEAR REGRESSION

04 DECISION TREE

05 CONCLUSION

INTRODUCTION

The US records daily ozone, SO2, CO and NO2 levels in several counties of every state.

The data set for this task contains the summary data for these readings, and associated meteorological data such as air quality index (AQI) and particulate matter (PM) index.

We are to find out the relationships between air pollution, the meteorological variables and the states and develop models to predict the number of days of PM > 2.5 concentrations.

01

Pre-processing

PRE-PROCESSING

READING & EXPLORING THE DATASET

```r
# get percentage of good / bad days
clean.data$percent_good <- (clean.data$Good.Days + clean.data$Moderate.Days) / clean.data$Days.with.AQI
clean.data$percent_bad <- (clean.data$Unhealthy.for.Sensitive.Groups.Days + clean.data$Unhealthy.Days +
                              clean.data$Very.Unhealthy.Days + clean.data$Hazardous.Days) /
                             clean.data$Days.with.AQI
```

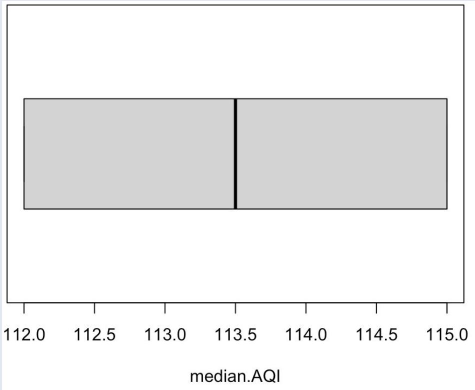
Created new columns to calculate the percentage of good / bad days respectively.

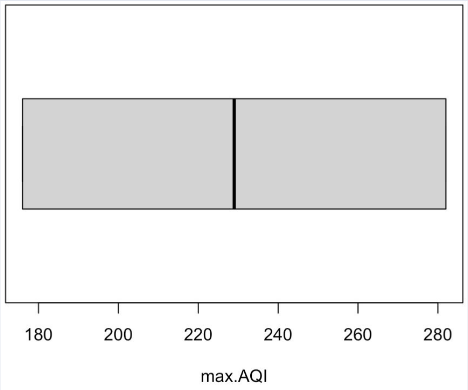
States with more bad days than good

	State	percent_good	percent_bad
1	Arizona	0.4508197	0.5491803
2	Country Of Mexico	0.4021739	0.5978261

PRE-PROCESSING

READING & EXPLORING THE DATASET





PRE-PROCESSING

CLEANING THE DATASET

```
data = read.csv('annual_aqi_by_county_2020.csv')
# summary of the dataset
str(data)
```

Removed *year* & *country* variables as all of the data is in 2020

```
> str(data)
'data.frame':	1003 obs. of  18 variables:
 $ State                              : chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
 $ County                             : chr  "Baldwin" "Clay" "DeKalb" "Elmore" ...
 $ Year                               : int  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 ...
 $ Days.with.AQI                      : int  269 108 364 197 278 366 90 364 348 361 ...
 $ Good.Days                          : int  250 99 350 197 260 212 87 307 256 162 ...
 $ Moderate.Days                      : int  19 9 14 0 18 151 3 57 92 197 ...
 $ Unhealthy.for.Sensitive.Groups.Days: int  0 0 0 0 0 3 0 0 0 2 ...
 $ Unhealthy.Days                     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Very.Unhealthy.Days                : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Hazardous.Days                     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Max.AQI                            : int  74 86 90 47 92 129 75 88 75 116 ...
 $ X90th.Percentile.AQI               : int  49 49 45 41 46 67 36 54 58 66 ...
 $ Median.AQI                         : int  36 26 36 31 34 48 18 40 42 52 ...
 $ Days.CO                            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Days.NO2                           : int  0 0 0 0 0 2 0 0 0 0 ...
 $ Days.Ozone                         : int  198 0 331 197 204 123 0 132 114 36 ...
 $ Days.PM2.5                         : int  71 108 33 0 74 241 90 227 234 325 ...
 $ Days.PM10                          : int  0 0 0 0 0 0 0 5 0 0 ...
```

PRE-PROCESSING

CLEANING THE DATASET

```r
# Label Encoding for State
clean.data$State <- as.integer(factor(clean.data$State))
```

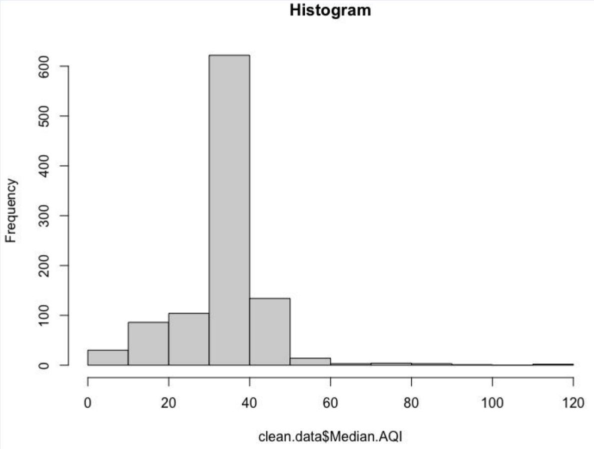
State
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
2
2
2

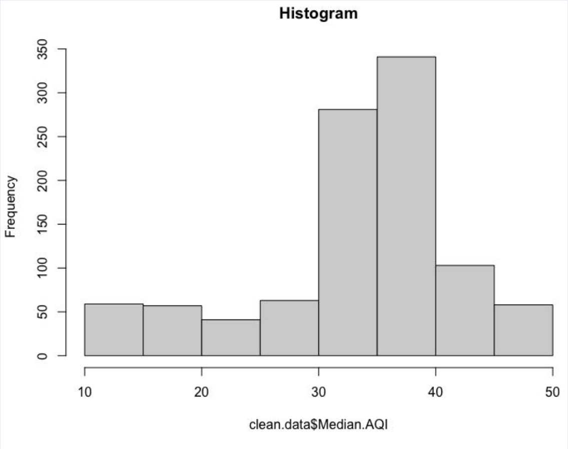
PRE-PROCESSING

CLEANING THE DATASET (Media AQI)

```
# Replacing outliers with 5% and 95% of interquartile range
median.AQI.quantile <- quantile(clean.data$Median.AQI,c(.05,0.95))
clean.data$Median.AQI <- squish(clean.data$Median.AQI, as.integer(median.AQI.quantile[1]), as.integer(median.AQI.quantile[2]))
```

“Squished” outliers into the 5% and 95% of the interquartile range



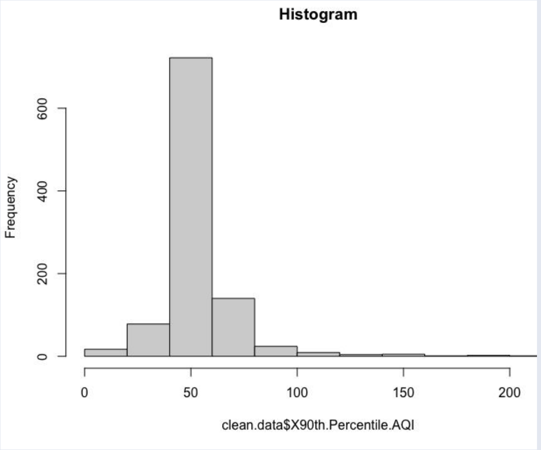


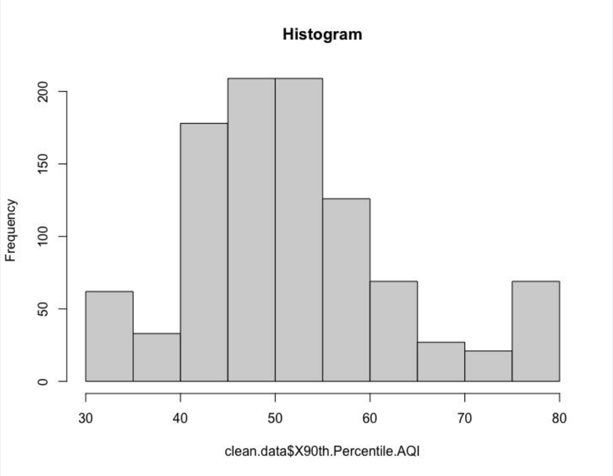
PRE-PROCESSING

CLEANING THE DATASET (90th Percentile AQI)

```
X90th.Percentile.AQI.quantile <- quantile(clean.data$X90th.Percentile.AQI,c(.05,0.95))
clean.data$X90th.Percentile.AQI <- squish(clean.data$X90th.Percentile.AQI,as.integer(X90th.Percentile.AQI.quantile[1]),
                                          as.integer(X90th.Percentile.AQI.quantile[2]))
```

“Squished” outliers into the 5% and 95% of the interquartile range



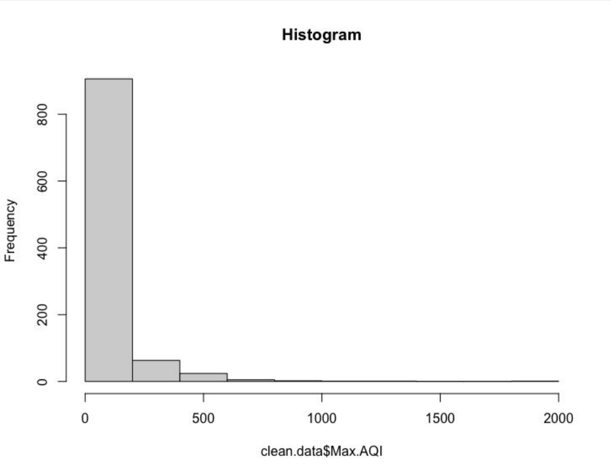


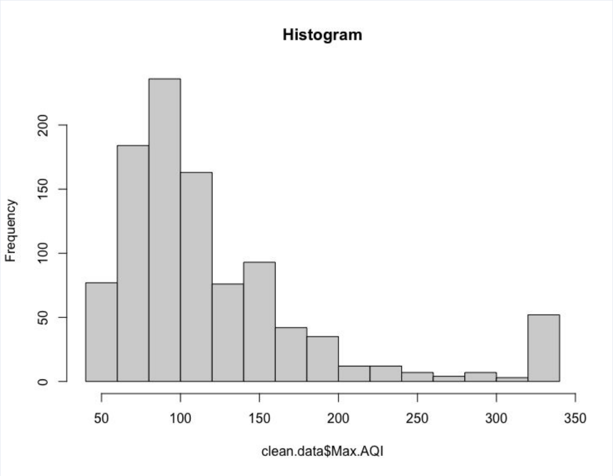
PRE-PROCESSING

CLEANING THE DATASET (Max AQI)

```
Max.AQI.quantile <- quantile(clean.data$Max.AQI,c(.05,0.95))
clean.data$Max.AQI <- squish(clean.data$Max.AQI,as.integer(Max.AQI.quantile[1]), as.integer(Max.AQI.quantile[2]))
```

“Squished” outliers into the 5% and 95% of the interquartile range





PRE-PROCESSING

FINDING THE CORRELATION

```
# Correlation in descending order
corrTable <- abs(cor(clean.data, y=clean.data$Days.PM2.5))
corrTable <- corrTable[order(corrTable, decreasing=TRUE),, drop=FALSE]
```

Days.PM2.5	1.000000000
Days.Ozone	0.683516487
Max.AQI	0.387826123
Moderate.Days	0.304622741
Days.with.AQI	0.280396864
Hazardous.Days	0.200807300
Median.AQI	0.198284899
X90th.Percentile.AQI	0.179486423
Unhealthy.Days	0.140857929
Days.PM10	0.124120002
Good.Days	0.110176064
Very.Unhealthy.Days	0.076463865
Days.CO	0.048699112
Unhealthy.for.Sensitive.Groups.Days	0.038796336
Days.NO2	0.034126264
State	0.007215575

PRE-PROCESSING

TRAIN & TEST DATA CREATION

```
# Set seed for reproducibility
set.seed(123)
```

Ensures reproducibility when generating a random number

PRE-PROCESSING

TRAIN & TEST DATA CREATION

```r
# Generate a vector of indices corresponding to the rows of your dataset
indices <- sample(1:nrow(clean.data), size = nrow(clean.data), replace = FALSE)
```

Vector of random indices created

PRE-PROCESSING

TRAIN & TEST DATA CREATION

```
# Calculate the number of rows for the training set (80%)
train_size <- round(0.8 * nrow(clean.data))

# Select the indices corresponding to the training set
train_indices <- indices[1:train_size]
```

- Set Training Set to 80% of dataset
- Select indices that corresponds to the Training Set

```
# Select the indices corresponding to the testing set
test_indices <- indices[(train_size + 1):nrow(clean.data)]
```

- Remaining goes to Test Set

PRE-PROCESSING

TRAIN & TEST DATA CREATION

```
# Create the training and testing datasets
train <- clean.data[train_indices, ]
test <- clean.data[test_indices, ]
```

test	201 obs. of 16 variables
train	802 obs. of 16 variables

02
Random Forest

Random Forest (Train)

```
> # RF Model on Train
> rf.train = randomForest(Days.PM2.5~., train, importance=TRUE)
> rf.train

Call:
 randomForest(formula = Days.PM2.5 ~ ., data = train, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 5

          Mean of squared residuals: 813.8069
                    % Var explained: 93.41
```

Mean of sq. residuals: 813.8069
R^2: 93.41%

Random Forest (Prediction)

Train

```
> # Predict train set and calculate RMSE
> rf.trainPred <- predict(rf.train, train)
> sqrt(mean((rf.trainPred - train$Days.PM2.5)^2))
[1] 12.74198
```

RMSE: 12.74198 days

Test

```
> # Predict test set and calculate RMSE
> rf.testPred <- predict(rf.train, test)
> sqrt(mean((rf.testPred - test$Days.PM2.5)^2))
[1] 34.79124
```

RMSE: 34.79124 days

Random Forest (Tuning)

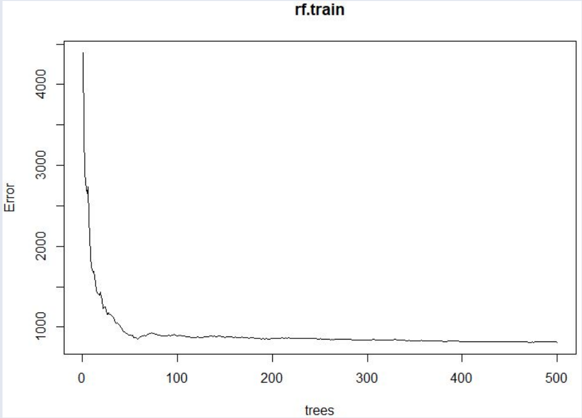
Parameter

mTry = 15

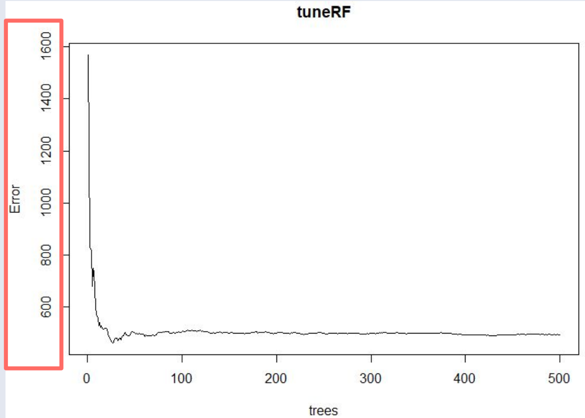
```
> # tuning the random forest with parameters:
> tuneRF <- tuneRF(
+   x           = train[features],
+   y           = train$Days.PM2.5,
+   ntreeTry    = 500,   # No. of trees
+   mtryStart   = 1,     # Starting value of mtry
+   stepFactor  = 3,   # mTry step factor
+   improve     = 0.05,   # Improvement to continue
+   trace       = TRUE,  # Shows progress
+   doBest      = TRUE,  # Returns tree with optimal mTry
+ )
mtry = 1  OOB error = 3652.294
Searching left ...
Searching right ...
mtry = 3        OOB error = 1210.595
0.6685385 0.05
mtry = 9        OOB error = 533.5601
0.5592579 0.05
mtry = 15       OOB error = 485.7787
0.08955199 0.05
```

Random Forest (Post-tuning)

Before



After



- Lower range of Error
- Less inaccuracies

Random Forest (Post-tuning)

Before

```
Call:
 randomForest(formula = Days.PM2.5 ~ ., data = train, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 5

          Mean of squared residuals: 813.8069
                    % Var explained: 93.41
```

After

```
Call:
 randomForest(x = x, y = y, mtry = res[which.min(res[, 2]), 1])
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 15

          Mean of squared residuals: 491.4446
                    % Var explained: 96.02
```

Mean of sq. residuals: -324.99900
R^2: +2.63%

Random Forest (Post-tuning)

Before

Train

```
> # Predict train set and calculate RMSE
> rf.trainPred <- predict(rf.train, train)
> sqrt(mean((rf.trainPred - train$Days.PM2.5)^2))
[1] 12.74198
```

Test

```
> # Predict test set and calculate RMSE
> rf.testPred <- predict(rf.train, test)
> sqrt(mean((rf.testPred - test$Days.PM2.5)^2))
[1] 34.79124
```

After

Train

```
> # Predict train set and calculate RMSE
> rfTuned.trainPred <- predict(tuneRF, train)
> sqrt(mean((rfTuned.trainPred - train$Days.PM2.5)^2))
[1] 8.995248
```

-3.746732

Test

```
> # Predict test set and calculate RMSE
> rfTuned.testPred <- predict(tuneRF, test)
> sqrt(mean((rfTuned.testPred - test$Days.PM2.5)^2))
[1] 22.11574
```

-12.6755

03
Decision Tree

Building the tree model

```
> tree = rpart(Days.PM2.5 ~ ., data=train, method ="anova")
> tree
```

```
n= 802

node), split, n, deviance, yval
      * denotes terminal node

 1) root 802 9890650.00 121.20700
   2) Days.Ozone>=196.5 451 1612052.00  57.03104
     4) Days.Ozone>=234.5 304  558136.30  31.85197
       8) Moderate.Days< 26.5 171  103208.50  10.09357 *
       9) Moderate.Days>=26.5 133  269885.00  59.82707
        18) Days.Ozone>=295.5 57   28340.56  20.75439 *
        19) Days.Ozone< 295.5 76   89258.68  89.13158 *
     5) Days.Ozone< 234.5 147  462609.50 109.10200
      10) Days.with.AQI< 313.5 34   28324.03  23.38235 *
      11) Days.with.AQI>=313.5 113  109288.70 134.89380 *
   3) Days.Ozone< 196.5 351 4034472.00 203.66670
     6) Days.with.AQI< 271.5 73  205237.50  81.91781 *
     7) Days.with.AQI>=271.5 278 2463032.00 235.63670
      14) Days.Ozone>=109.5 146  142787.10 186.69180 *
      15) Days.Ozone< 109.5 132 1583633.00 289.77270
        30) Days.PM10>=166.5 13   42064.31  45.23077 *
        31) Days.PM10< 166.5 119  679231.70 316.48740
          62) X90th.Percentile.AQI< 32.5 8  119359.90 111.37500 *
          63) X90th.Percentile.AQI>=32.5 111  199045.90 331.27030 *
```

rpart is used to build the decision tree

n means the number of observations that is used in the model which is 802

The 1st split is based on the Days.Ozone at >=196.5 and <196.5

451 node cases are Days.Ozone >=196.5

With 57.03104 the number of days predicted for this node.

First Prediction

I have used both training set and test set to do the prediction. Then I computed out the RMSE.

```
> #RMSE of the train prediction
> sqrt(mean((tree.pred - train$Days.PM2.5)^2))
[1] 36.47353
```

The RMSE for training set is 36.47 days.

```
> tree.pred = predict(tree, test, method = "anova")
> #RMSE of the test prediction
> sqrt(mean((tree.pred - test$Days.PM2.5)^2))
[1] 50.56401
```

The RMSE for test set is 50.56 days.

Tuning the model

```
> control <- rpart.control(minsplit = 4,
+                          minbucket = round(5 / 3),
+                          maxdepth = 16,
+                          cp = 0)
> #Fitting of tuned model
> tune_fit <- rpart(Days.PM2.5~., data = train, method = 'anova', control = control)
```

Prediction After Tuning

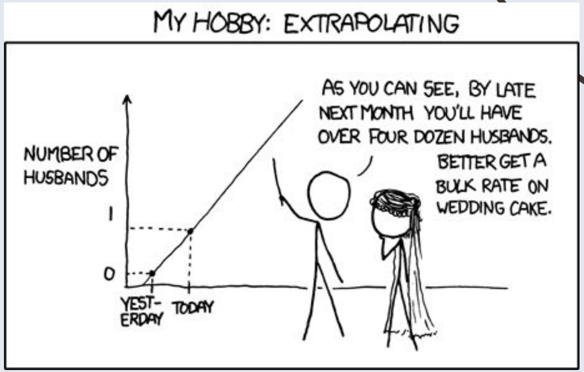
```
> #train
> tune.pred = predict(tune_fit, train, method = "anova")
> sqrt(mean((tune.pred - train$Days.PM2.5)^2))
[1] 9.411476
```

```
> #test
> tune.pred = predict(tune_fit, test, method = "anova")
> sqrt(mean((tune.pred - test$Days.PM2.5)^2))
[1] 44.73216
```

The RMSE for both training and test sets are slight lower than before.

Therefore, the model is overfitted.

04 Linear Regression



Training the model

```
> lr.train1 <- lm(Days.PM2.5 ~.-Days.NO2, data = train)
> summary(lr.train1)

Call:
lm(formula = Days.PM2.5 ~ . - Days.NO2, data = train)

Residuals:
    Min      1Q  Median      3Q     Max 
-331.34   -2.13    3.00    7.45   34.79 
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 24.1 on 788 degrees of freedom
Multiple R-squared:  0.9538,	Adjusted R-squared:  0.953 
F-statistic:  1251 on 13 and 788 DF,  p-value: < 2.2e-16
```

Important takeaways:

-fstat value of 1251

-R^2 value of 0.9538

Predicting using the model

```
> lr.predict1 = predict(lr.train1, newdata = test)
> # Calculate RMSE
> #RMSE(lr.predict1, test$Days.PM2.5)
> rmse <- sqrt(mean((lr.predict1 - test$Days.PM2.5)^2))
> # Print RMSE
> rmse
[1] 28.85616
```

RMSE value of 28.86

Fine-tuning the model

```
> # 2nd model, using only attribute with a significant p-value
> lr.train2 <-stepAIC(lr.train1,direction="both")
> summary(lr.train2)

Call:
lm(formula = Days.PM2.5 ~ State + Days.with.AQI + Max.AQI + Median.AQI +
    Days.CO + Days.Ozone + Days.PM10 + Unhealthy.for.Sensitive.Groups.Days,
    data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-334.16   -2.52    2.85    7.22   34.99
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 24.03 on 793 degrees of freedom
Multiple R-squared:  0.9521   Adjusted R-squared:  0.954
F-statistic:  2045 on 8 and 793 DF,  p-value: < 2.2e-16
```

Important takeaways:

-fstat value of 2045

-R^2 value of 0.9521

Predicting using the model

```
> lr.predict2 = predict(lr.train2, newdata = test)
> # Calculate RMSE
> #RMSE(lr.predict2, test$Days.PM2.5)
> rmse <- sqrt(mean((lr.predict2 - test$Days.PM2.5)^2))
> # Print RMSE
> rmse
[1] 27.90545
```

RMSE value of 27.91

Fine-tuning the model

Important takeaways:

-fstat value of 2170

-R^2 value of 0.9526

```
> # 3rd model, remove attributes with p-value< 0.05
> lr.train3 =update(lr.train2, ~. -StateNew.York-StateLouisiana, data =
train)
> summary(lr.train3)

Call:
lm(formula = Days.PM2.5 ~ State + Days.with.AQI + Max.AQI + Median.AQI +
    Days.CO + Days.Ozone + Days.PM10 + Unhealthy.for.Sensitive.Groups.Days,
    data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-334.52   -2.48    2.85    7.11   35.03
```

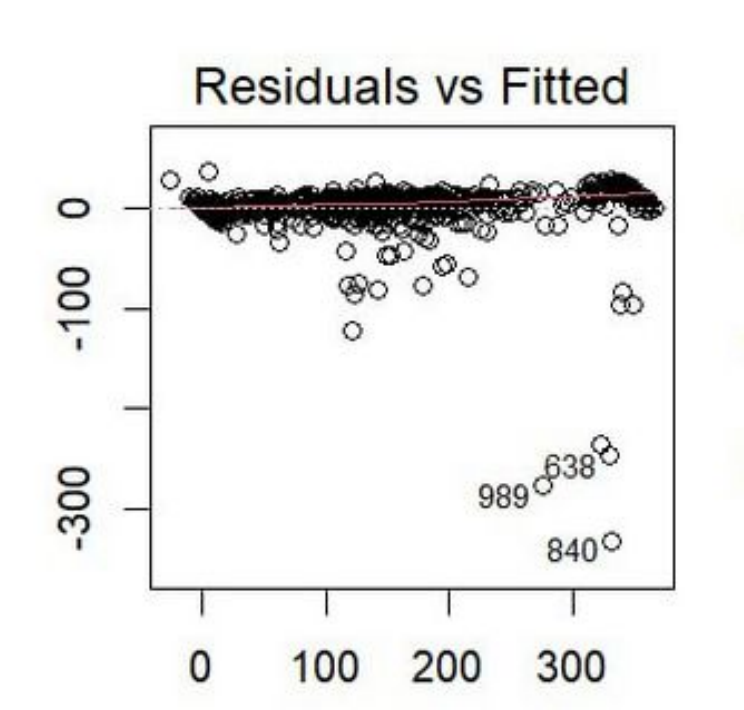
```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 24.03 on 798 degrees of freedom
Multiple R-squared:  0.9526,  Adjusted R-squared:  0.9533
F-statistic:  2170 on 8 and 793 DF,  p-value: < 2.2e-16
```

Predicting using the model

```
> lr.predict3 = predict(lr.train3, newdata = test)
> # Calculate RMSE
> #RMSE(lr.predict3, test$Days.PM2.5)
> rmse <- sqrt(mean((lr.predict3 - test$Days.PM2.5)^2))
> # Print RMSE
> rmse
[1] 27.41584
```

RMSE value of 27.42

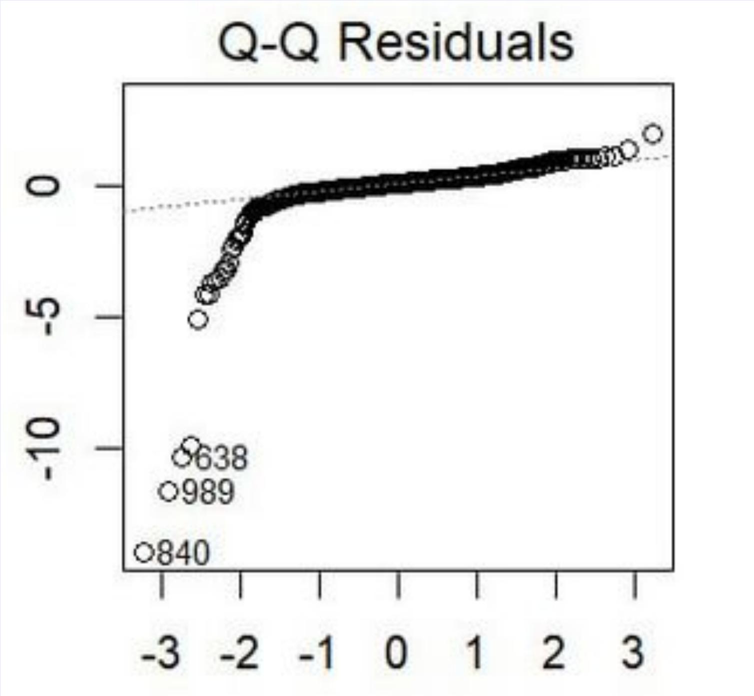
Linear Regression - Model Analysis



Residuals vs Fitted Plot:

- Shows relationship between residuals and fitted values.
- Random scatter suggests linearity and constant variance.
- Average of residuals should be close to 0.

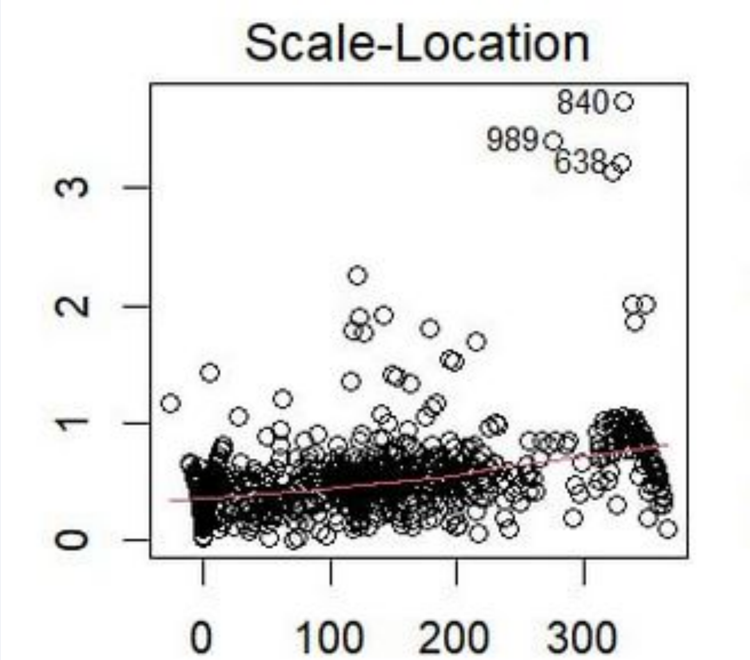
Linear Regression - Model Analysis



Residual Q-Q Plot:

- Compares residuals' quantiles to theoretical normal distribution.
- Points close to diagonal line indicate normality of residuals.
- Departures from line suggest deviations from normality.

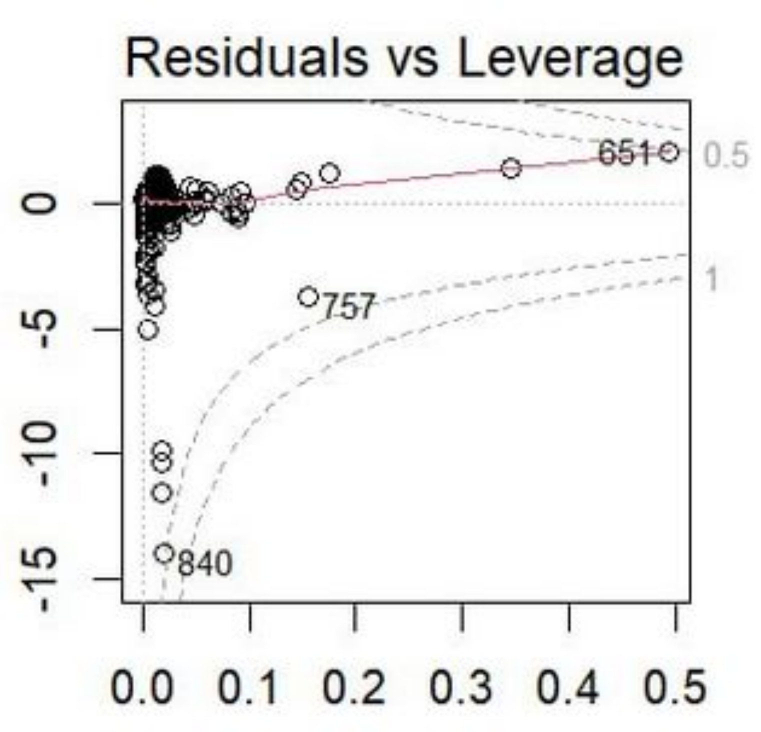
Linear Regression - Model Analysis



Scale-Location Plot

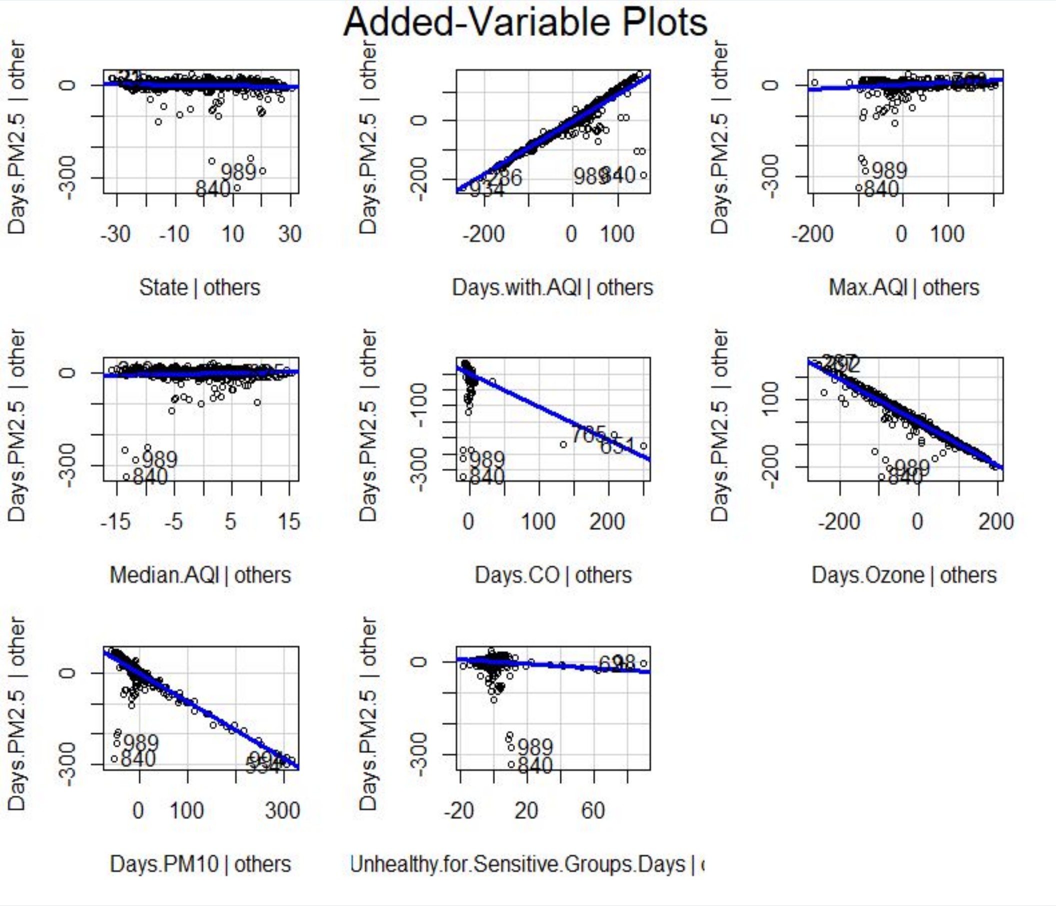
- Displays spread of residuals against fitted values.
- Horizontal line with equally spread points indicates homoscedasticity.
- Straight red line = equal spread of residuals across fitted values = good.

Linear Regression - Model Analysis



Residuals vs Leverage Plot:

- Assesses influence of each data point on regression coefficients.
- Points outside dashed lines have high leverage.
- Identifies influential observations impacting regression model.



Added Variable Plots

- Blue line represents the relationship between variable and what we want to predict.
- For each variable plot, other predictor values are constant. Only changing what we are looking at.
- Generally, there is a negative relationship between the plots.

05
Conclusion

Conclusion

Random Forest Model

Before

Train

```
> # Predict train set and calculate RMSE
> rf.trainPred <- predict(rf.train, train)
> sqrt(mean((rf.trainPred - train$Days.PM2.5)^2))
[1] 12.74198
```

Test

```
> # Predict test set and calculate RMSE
> rf.testPred <- predict(rf.train, test)
> sqrt(mean((rf.testPred - test$Days.PM2.5)^2))
[1] 34.79124
```

After

Train

```
> # Predict train set and calculate RMSE
> rfTuned.trainPred <- predict(tuneRF, train)
> sqrt(mean((rfTuned.trainPred - train$Days.PM2.5)^2))
[1] 8.995248
```

-3.746732

Test

```
> # Predict test set and calculate RMSE
> rfTuned.testPred <- predict(tuneRF, test)
> sqrt(mean((rfTuned.testPred - test$Days.PM2.5)^2))
[1] 22.11574
```

-12.6755

Conclusion

Random Forest Model

Strengths

- Avoids & prevents overfitting by using multiple trees
- More accurate

Weakness

- Slower training time as complex model

Conclusion

Decision Tree Model

Before

Train

```
> #RMSE of the train prediction
> sqrt(mean((tree.pred - train$Days.PM2.5)^2))
[1] 36.47353
```

Test

```
> tree.pred = predict(tree, test, method = "anova")
> #RMSE of the test prediction
> sqrt(mean((tree.pred - test$Days.PM2.5)^2))
[1] 50.56401
```

After

Train

```
> #train
> tune.pred = predict(tune_fit, train, method = "anova")
> sqrt(mean((tune.pred - train$Days.PM2.5)^2))
[1] 9.411476
```

-27.06205

Test

```
> #test
> tune.pred = predict(tune_fit, test, method = "anova")
> sqrt(mean((tune.pred - test$Days.PM2.5)^2))
[1] 44.73216
```

-5.83185

Conclusion

Decision Tree Model

Strengths

- Requires less effort for data preparation during pre-processing

Weakness

- Slower training time as complex model
- Small changes in data can cause large changes in the structure of the decision tree

Conclusion

Linear Regression Model

Before

```
> lr.predict1 = predict(lr.train1, newdata = test)
> # Calculate RMSE
> #RMSE(lr.predict1, test$Days.PM2.5)
> rmse <- sqrt(mean((lr.predict1 - test$Days.PM2.5)^2))
> # Print RMSE
> rmse
[1] 28.85616
```

RMSE value is 28.87

After

```
> lr.predict3 = predict(lr.train3, newdata = test)
> # Calculate RMSE
> #RMSE(lr.predict3, test$Days.PM2.5)
> rmse <- sqrt(mean((lr.predict3 - test$Days.PM2.5)^2))
> # Print RMSE
> rmse
[1] 27.41584
```

RMSE value is 27.42

Conclusion

Linear Regression Model

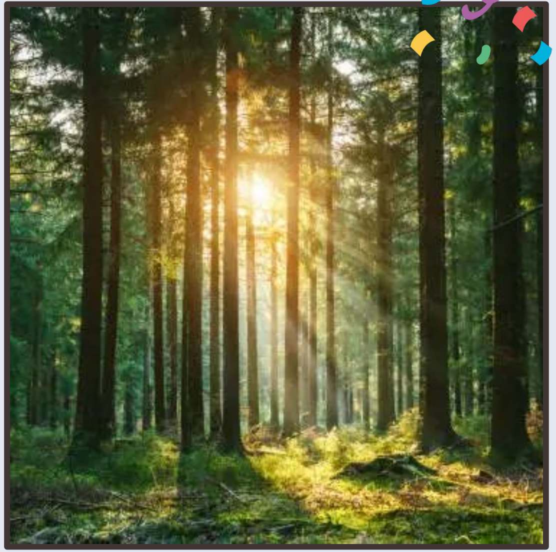
Strengths

- Requires less effort for data preparation during pre-processing

Weakness

- Sensitive to outliers

Best model:
Random Forest



Train

```
> # Predict train set and calculate RMSE
> rfTuned.trainPred <- predict(tuneRF, train)
> sqrt(mean((rfTuned.trainPred - train$Days.PM2.5)^2))
[1] 8.995248
```

Test

```
> # Predict test set and calculate RMSE
> rfTuned.testPred <- predict(tuneRF, test)
> sqrt(mean((rfTuned.testPred - test$Days.PM2.5)^2))
[1] 22.11574
```

Thank you