Research Project 2

```r
#load packages
library(tidyverse)
library(dplyr)
library(caret)
library(tidymodels)
library(rpart)
library(knitr)

#read in the data
dat <- read_csv("https://github.com/rdpeng/stat322E_public/raw/main/data/pm25_data.csv.gz")
dat1 <- dat
```

Analyzing and Predicting Ground-Level Pollution Metrics across the US

By: Tanvi Panchumarthy, Elaina Partida, and Anish Sunk

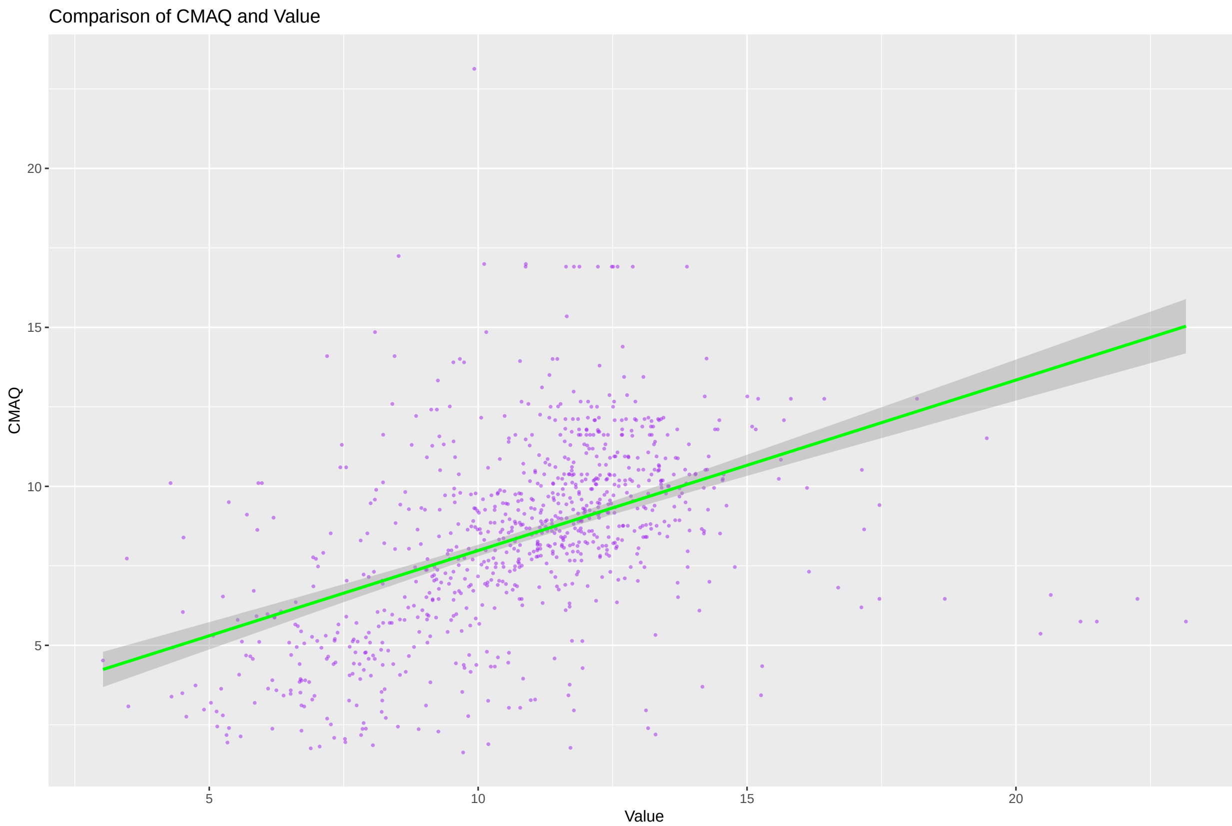
Introduction

Monitoring pollution levels has been an important task for government agencies since the Industrial Revolution. PM2.5 values have traditionally been used as the standard metric for ground level pollution. However, there have been recent advancements in pollution quantification methods that aim to improve the cost-effectiveness of pollution level monitoring in the US, and two new methods for pollution have been introduced: AOD and CMAQ.

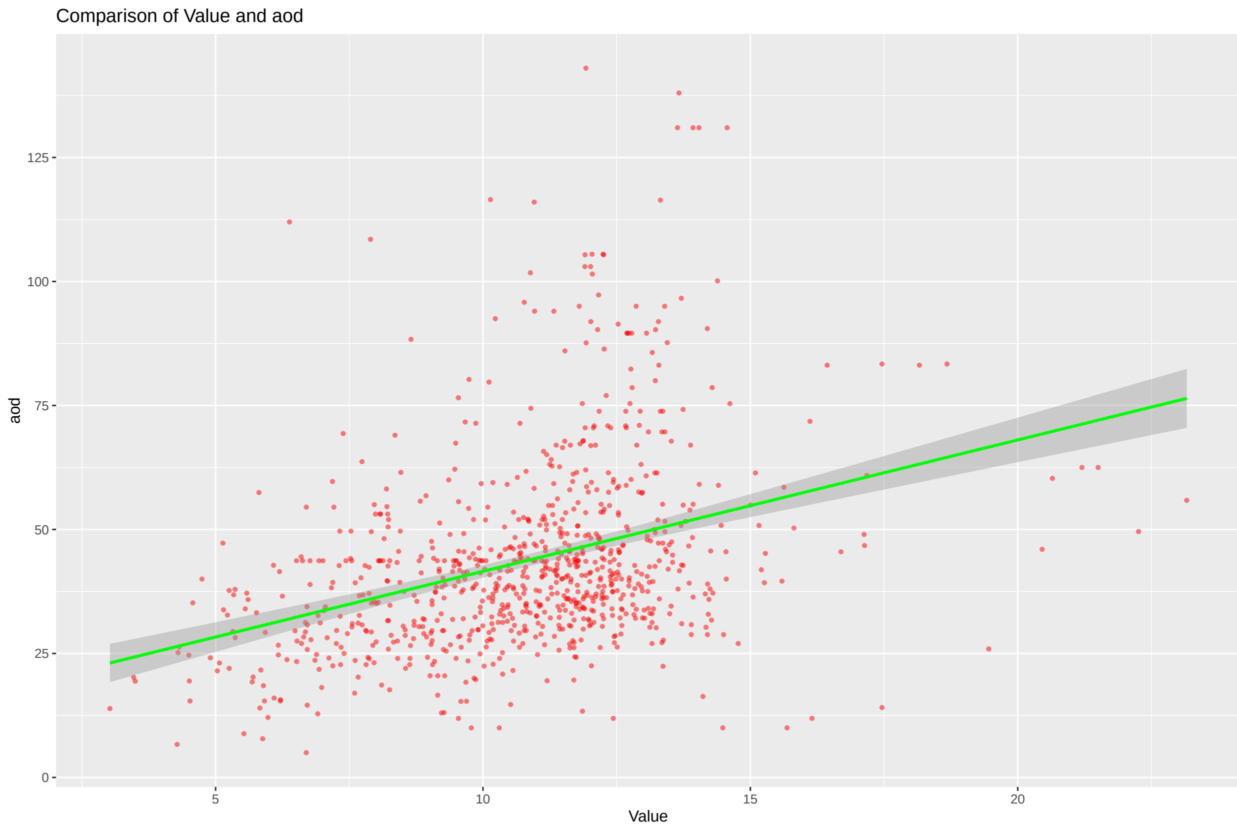
Introduction: Exploratory Data Analysis

Before we began the analysis, we took a look at the data.

```r
##plot CMAQ and Value for data analysis
dat1%>%
  ggplot (aes(x= value, y = CMAQ))+
  geom_point(color="purple", fill="purple", size= 0.55, alpha= 0.5)+
labs (x = "Value")+
  labs (y ="CMAQ")+
  labs (title = "Comparison of CMAQ and Value")+
  geom_smooth(method ="lm", linewidth = 1, color= "green")+
  scale_y_continuous( breaks = seq(0,35,5))
```




```
##plot AOD and Value
dat1%>%
  ggplot (aes(x= value, y = aod))+
  geom_point(color="red", fill="red", size= 1, alpha= 0.5)+
labs (x = "Value")+
  labs (y ="aod")+
  labs (title = "Comparison of Value and aod")+
  geom_smooth(method ="lm", linewidth = 1, color= "green")+
      scale_x_continuous( breaks = seq(0,25,5))+
  scale_y_continuous( breaks =seq(0,150,25))
```



Here, we see that there is somewhat of a positive correlation between the CMAQ and value variables as well as between the AOD and value variables. This indicates that linear regression would be a good start for our modelling, and that we should predict the PM2.5 pollution levels on the CMAQ and AOD.

Introduction: Models Used **We first predicted the PM2.5 value with a linear regression model. Then we used k-NN, random forest, and boosted trees models to try to fit the data as well.**

Wrangling

```r
#group the data by the state variable
dat1 <- dat %>%
  select(value, state, CMAQ, county_pop, aod) %>%
  group_by(state)
dat1
```

```
## # A tibble: 876 x 5
## # Groups:   state [49]
##    value state    CMAQ county_pop   aod
##    <dbl> <chr>   <dbl>      <dbl> <dbl>
##  1  9.60 Alabama  8.10     182265  37.4
##  2 10.8  Alabama  9.77      13932  34.8
##  3 11.2  Alabama  9.40      54428  36
##  4 11.7  Alabama  8.53      71109  33.1
##  5 12.4  Alabama  9.24     104430  43.4
```

```
##  6 10.5  Alabama  9.12     101547  33
##  7 15.6  Alabama 10.2      658466  39.6
##  8 12.4  Alabama 10.2      658466  38.8
##  9 11.1  Alabama  8.16     194656  40.4
## 10 13.1  Alabama  9.30     658466  42.5
## # i 866 more rows
```

```
#summarize the data for all monitors in each state
dat1 <- dat1 %>%
  group_by(state) %>%
  summarize(across(value:aod, mean))

dat1
```

```
## # A tibble: 49 x 5
##    state                value  CMAQ county_pop   aod
##    <chr>                <dbl> <dbl>      <dbl> <dbl>
##  1 Alabama              11.8   9.34    315426.  38.8
##  2 Arizona               8.57 10.2    1362653.  29.3
##  3 Arkansas             11.2   9.49    125542.  40.0
##  4 California           12.2   7.92   2078390.  48.0
##  5 Colorado              7.34  4.74    379611.  38.6
##  6 Connecticut          10.6   7.40    744403.  35.2
##  7 Delaware             12.2   9.52    382240.  55.5
##  8 District Of Columbia 12.1  11.1     601723   62.6
##  9 Florida               7.90  7.87   1007859.  38.4
## 10 Georgia              12.5  10.3     301249   39.5
## # i 39 more rows
```

Originally, the dataset had observations such that each row of data in the dataset represented the data from a single monitor and the dataset had an ID variable. However, we decided to group the data so that we could get data by state and take the average of the PM2.5 levels and other metrics for all of the monitors in a given state. Then, we could perform an analysis on pollution levels in different states and predict the values for new monitors in a given state.

```
#scale county_pop variable in the dataset
dat1$county_pop_scaled <- scale(dat1$county_pop)
dat1$county_pop <- dat1$county_pop_scaled
dat1$county_pop_scaled <- NULL
# Now dat1 has the scaled county_pop values in the county_pop column.
dat1
```

```
## # A tibble: 49 x 5
##    state                value  CMAQ county_pop[,1]   aod
##    <chr>                <dbl> <dbl>          <dbl> <dbl>
##  1 Alabama              11.8   9.34         -0.383  38.8
##  2 Arizona               8.57 10.2           1.89   29.3
##  3 Arkansas             11.2   9.49         -0.795  40.0
##  4 California           12.2   7.92          3.44   48.0
##  5 Colorado              7.34  4.74         -0.244  38.6
##  6 Connecticut          10.6   7.40          0.546  35.2
##  7 Delaware             12.2   9.52         -0.238  55.5
##  8 District Of Columbia 12.1  11.1           0.237  62.6
```

```
##   9 Florida                7.90  7.87           1.12   38.4
## 10 Georgia               12.5   10.3           -0.414  39.5
## # i 39 more rows
```

```
set.seed(123)
#split data into training and testing data
trainIndex <- createDataPartition(dat1$value, p = 0.5, list = FALSE)
trainData <- dat1[trainIndex, ]
testData <- dat1[-trainIndex, ]

#take a look at training and testing data
head(testData)
```

```
## # A tibble: 6 x 5
##   state       value  CMAQ county_pop[,1]   aod
##   <chr>       <dbl> <dbl>          <dbl> <dbl>
## 1 Alabama     11.8   9.34         -0.383  38.8
## 2 Arizona      8.57 10.2           1.89   29.3
## 3 Arkansas    11.2   9.49         -0.795  40.0
## 4 California  12.2   7.92          3.44   48.0
## 5 Colorado     7.34  4.74         -0.244  38.6
## 6 Connecticut 10.6   7.40          0.546  35.2
```

```
head(trainData)
```

```
## # A tibble: 6 x 5
##   state                value  CMAQ county_pop[,1]   aod
##   <chr>                <dbl> <dbl>          <dbl> <dbl>
## 1 Delaware             12.2   9.52         -0.238  55.5
## 2 District Of Columbia 12.1  11.1           0.237  62.6
## 3 Florida               7.90  7.87          1.12   38.4
## 4 Idaho                 8.21  3.92         -0.796  36.0
## 5 Illinois             11.6  11.0           2.89   53.8
## 6 Indiana              12.4  11.9          -0.401  45.7
```

Model 1: Linear Regression

```
model <- lm(value ~ CMAQ + aod + county_pop, data = trainData)
predicted <- predict(model, newdata = testData)
summary(model)
```

```
##
## Call:
## lm(formula = value ~ CMAQ + aod + county_pop, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2352 -0.5309  0.2301  0.6822  1.9085
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.16025    0.89391   4.654 0.000136 ***
```

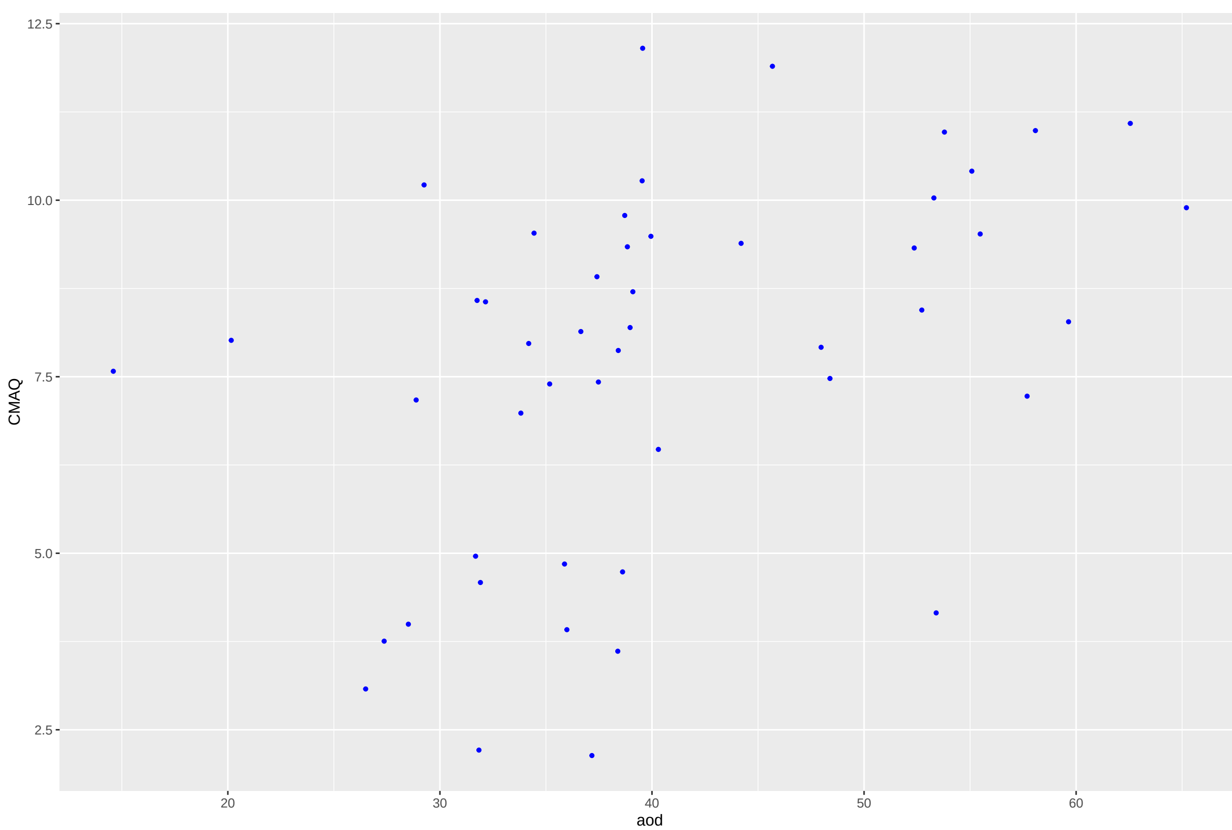
```
## CMAQ         0.45589    0.10257   4.445 0.000224 ***
## aod          0.05702    0.02322   2.456 0.022848 *
## county_pop  -0.26733    0.23554  -1.135 0.269174
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.066 on 21 degrees of freedom
## Multiple R-squared:  0.7214, Adjusted R-squared:  0.6817
## F-statistic: 18.13 on 3 and 21 DF,  p-value: 4.851e-06
```

```r
RMSE <- sqrt(mean((predicted - testData$value)^2))
RMSE
```

```
## [1] 1.425438
```

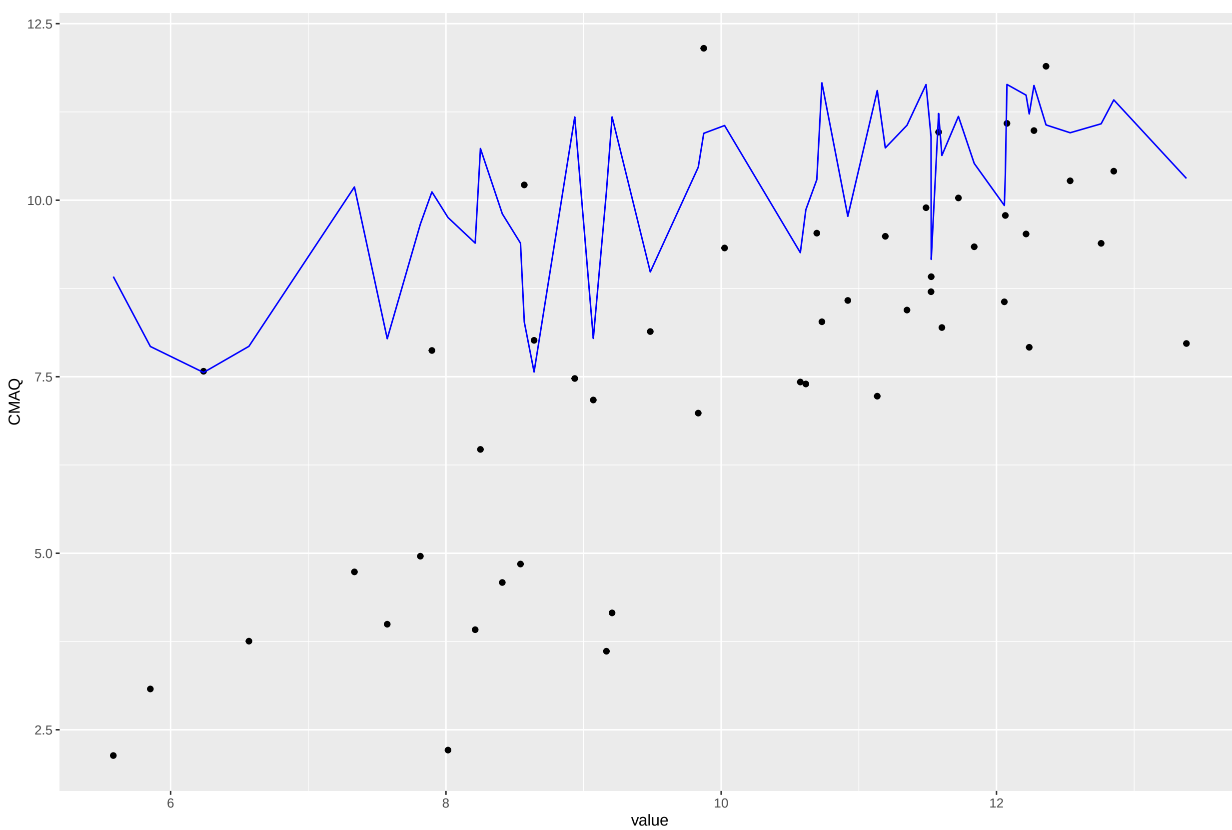
Overall, in the linear model, we predict the value of PM2.5 pollution with an RMSE value of 1.425438 which means that the model predicts the value of PM2.5 within about 1-1.5 micrograms per cubic meter. This is a decent model, but the error is still a bit high compared to what we wish to see.

```r
#create plot of aod and CMAQ values
dat1 %>%
    ggplot(aes(x= aod, y= CMAQ)) +
    geom_point(color= "blue", size = 1, alpha= 5) +
    labs(x = "aod", y = "CMAQ")
```



Model 2: k-NN

```r
#create the k-NN model)
rec <- recipe(value ~ aod,
              data = dat1)

model <- nearest_neighbor(neighbors = 10) %>%
    set_engine("kknn") %>%
    set_mode("regression")

wf <- workflow() %>%
    add_recipe(rec) %>%
    add_model(model)

model_fit <- fit(wf, data = dat1)


model_fit %>%
    extract_fit_parsnip() %>%
    augment(dat1) %>%
    arrange(value) %>%
    ggplot(aes(value, CMAQ)) +
    geom_point() +
    geom_line( aes(value, .pred),
              color = "blue")
```



```r
data(dat1)

#separating the data into test and train by selecting variables and splitting
set.seed(123)
dat1 <- dat %>%
  select(value, state, CMAQ, county_pop, aod)
dat1
```

```
## # A tibble: 876 x 5
##    value state    CMAQ county_pop   aod
##    <dbl> <chr>   <dbl>      <dbl> <dbl>
##  1  9.60 Alabama  8.10     182265  37.4
##  2 10.8  Alabama  9.77      13932  34.8
##  3 11.2  Alabama  9.40      54428  36
##  4 11.7  Alabama  8.53      71109  33.1
##  5 12.4  Alabama  9.24     104430  43.4
##  6 10.5  Alabama  9.12     101547  33
##  7 15.6  Alabama 10.2      658466  39.6
##  8 12.4  Alabama 10.2      658466  38.8
##  9 11.1  Alabama  8.16     194656  40.4
## 10 13.1  Alabama  9.30     658466  42.5
## # i 866 more rows
```

```r
dat1_split <- initial_split(dat1)
dat1_train <- training(dat1_split)


dat1_split
```

```
## <Training/Testing/Total>
## <657/219/876>
```

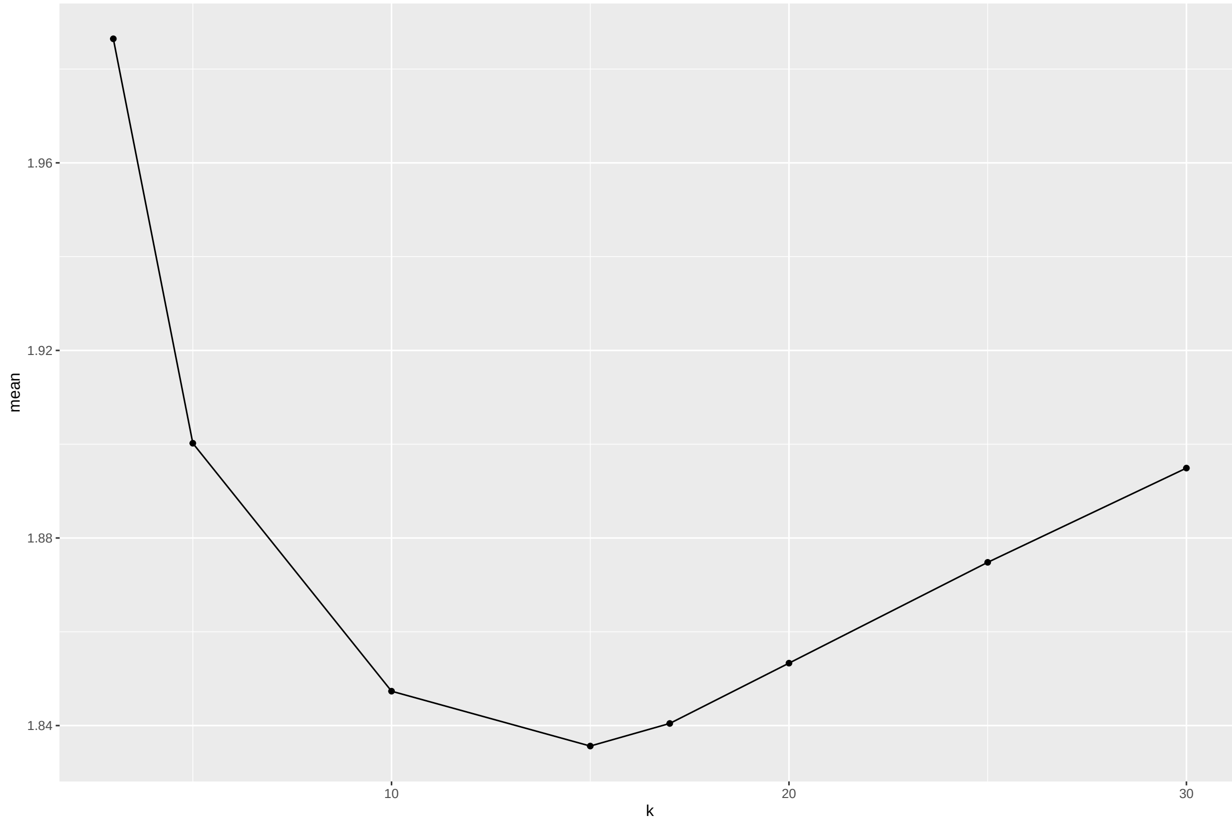
```r
#creating the recipe
rec <- dat1_train %>%
    recipe(value ~ .) %>%
    step_normalize()

#creating the KNN model using 15 as the optimal K value
model <- nearest_neighbor(neighbors = 15) %>%
    set_engine("kknn") %>%
    set_mode("regression")

wf <- workflow() %>%
    add_model(model) %>%
    add_recipe(rec)

folds <- vfold_cv(dat1_train, v = 6)

res <- fit_resamples(wf, resamples = folds)

res %>%
    collect_metrics()
```

```
## # A tibble: 2 x 6
##   .metric .estimator  mean     n std_err .config
##   <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 rmse    standard   1.83      5  0.107  Preprocessor1_Model1
## 2 rsq     standard   0.494     5  0.0231 Preprocessor1_Model1
```

```
##Tuning to find the best k value

model <- nearest_neighbor(neighbors = tune("k")) %>%
    set_engine("kknn") %>%
    set_mode("regression")

wf <- workflow() %>%
    add_model(model) %>%
    add_recipe(rec)
folds <- vfold_cv(dat1_train, v =6 )

res <- tune_grid(wf, resamples = folds,
                 grid = tibble(k = c(3,5, 10, 17,15, 20, 25, 30)))


res %>%
    collect_metrics()
```

```
## # A tibble: 16 x 7
##        k .metric .estimator  mean     n std_err .config
##    <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
##  1     3 rmse    standard   1.99      5  0.138  Preprocessor1_Model1
##  2     3 rsq     standard   0.461     5  0.0375 Preprocessor1_Model1
##  3     5 rmse    standard   1.90      5  0.161  Preprocessor1_Model2
##  4     5 rsq     standard   0.486     5  0.0494 Preprocessor1_Model2
##  5    10 rmse    standard   1.85      5  0.178  Preprocessor1_Model3
##  6    10 rsq     standard   0.500     5  0.0582 Preprocessor1_Model3
##  7    15 rmse    standard   1.84      5  0.179  Preprocessor1_Model4
##  8    15 rsq     standard   0.504     5  0.0596 Preprocessor1_Model4
##  9    17 rmse    standard   1.84      5  0.178  Preprocessor1_Model5
## 10    17 rsq     standard   0.502     5  0.0594 Preprocessor1_Model5
## 11    20 rmse    standard   1.85      5  0.177  Preprocessor1_Model6
## 12    20 rsq     standard   0.496     5  0.0597 Preprocessor1_Model6
## 13    25 rmse    standard   1.87      5  0.176  Preprocessor1_Model7
## 14    25 rsq     standard   0.486     5  0.0601 Preprocessor1_Model7
## 15    30 rmse    standard   1.89      5  0.178  Preprocessor1_Model8
## 16    30 rsq     standard   0.477     5  0.0619 Preprocessor1_Model8
```

```
res %>%
    collect_metrics() %>%
    filter(.metric == "rmse") %>%
    ggplot(aes(k, mean)) +
    geom_point() +
    geom_line()
```




```r
#display rmse values of top performing k values in kNN model
res %>%
    show_best(metric = "rmse")
```

```
## # A tibble: 5 x 7
##       k .metric .estimator  mean     n std_err .config
##   <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1    15 rmse    standard    1.84     5   0.179 Preprocessor1_Model4
## 2    17 rmse    standard    1.84     5   0.178 Preprocessor1_Model5
## 3    10 rmse    standard    1.85     5   0.178 Preprocessor1_Model3
## 4    20 rmse    standard    1.85     5   0.177 Preprocessor1_Model6
## 5    25 rmse    standard    1.87     5   0.176 Preprocessor1_Model7
```

After running the tuning algorithms, we see that using 15 nearest neighbors in the kNN model yields the lowest RMSE value of 1.8356, so we decide to use 15 neighbors for the model.

Model 3: Random Forest

```r
rec_rf <- dat1_train %>%
    recipe(value ~ .) %>%
    step_normalize()

model_rf <- rand_forest(mtry = 6) %>%
    set_engine("ranger") %>%
    set_mode("regression")
```

```r
wf_rf <- workflow() %>%
    add_recipe(rec_rf) %>%
    add_model(model_rf)

folds <- vfold_cv(dat1_train, v = 6)

res <- fit_resamples(wf_rf, resamples = folds)

#look at RMSE values only + arrange ascending to get smallest value for RMSE
res %>%
  collect_metrics() %>%
  filter(.metric == "rmse") %>%
  arrange(mean)
```

```
## # A tibble: 1 x 6
##   .metric .estimator  mean     n std_err .config
##   <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 rmse    standard    1.92     5   0.118 Preprocessor1_Model1
```

Model 4: Boosted Trees

```r
dat1_train
```

```
## # A tibble: 657 x 5
##    value state                 CMAQ county_pop   aod
##    <dbl> <chr>                <dbl>      <dbl> <dbl>
##  1 11.2  Massachusetts         8.04     722023  52
##  2  6.70 Minnesota             3.94     200226  54.5
##  3 12.1  District Of Columbia 12.5      601723  67
##  4  8.55 New Hampshire         4.07      43742  28.6
##  5  7.02 Florida               7.48     618754  33.6
##  6 13.7  Virginia              8.94     242803 138
##  7 17.5  California            9.41    3095313  14.1
##  8 11.6  Indiana              16.9      496005  67
##  9  9.55 Georgia               9.93     109233  42.9
## 10  9.72 Idaho                 1.63       7936  23.5
## # i 647 more rows
```

```r
rec_bt <- dat1_train %>%
    recipe(value ~ CMAQ + county_pop + aod) %>%
    step_normalize()

model_bt <- boost_tree(mode = "regression") %>%
    set_engine("xgboost")

wf_bt <- workflow() %>%
    add_recipe(rec_bt) %>%
    add_model(model_bt)

folds <- vfold_cv(dat1_train, v = 6)

res <- fit_resamples(wf_bt, resamples = folds)
```

```r
#look at RMSE values only + arrange ascending to get smallest value for RMSE
res %>%
  collect_metrics() %>%
  filter(.metric == "rmse") %>%
  arrange(mean)
```

```
## # A tibble: 1 x 6
##   .metric .estimator  mean     n std_err .config
##   <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 rmse    standard    1.90     6  0.0645 Preprocessor1_Model1
```

Best and Final Model: Linear Regression

Determining the Best Model with RMSE Values *We determined that linear regression was the best model to use on this dataset since it produces the lowest RMSE value on the training dataset. Now, we apply the linear regression model to the testing data to see how the model will perform on new data.*

Best and Final Model

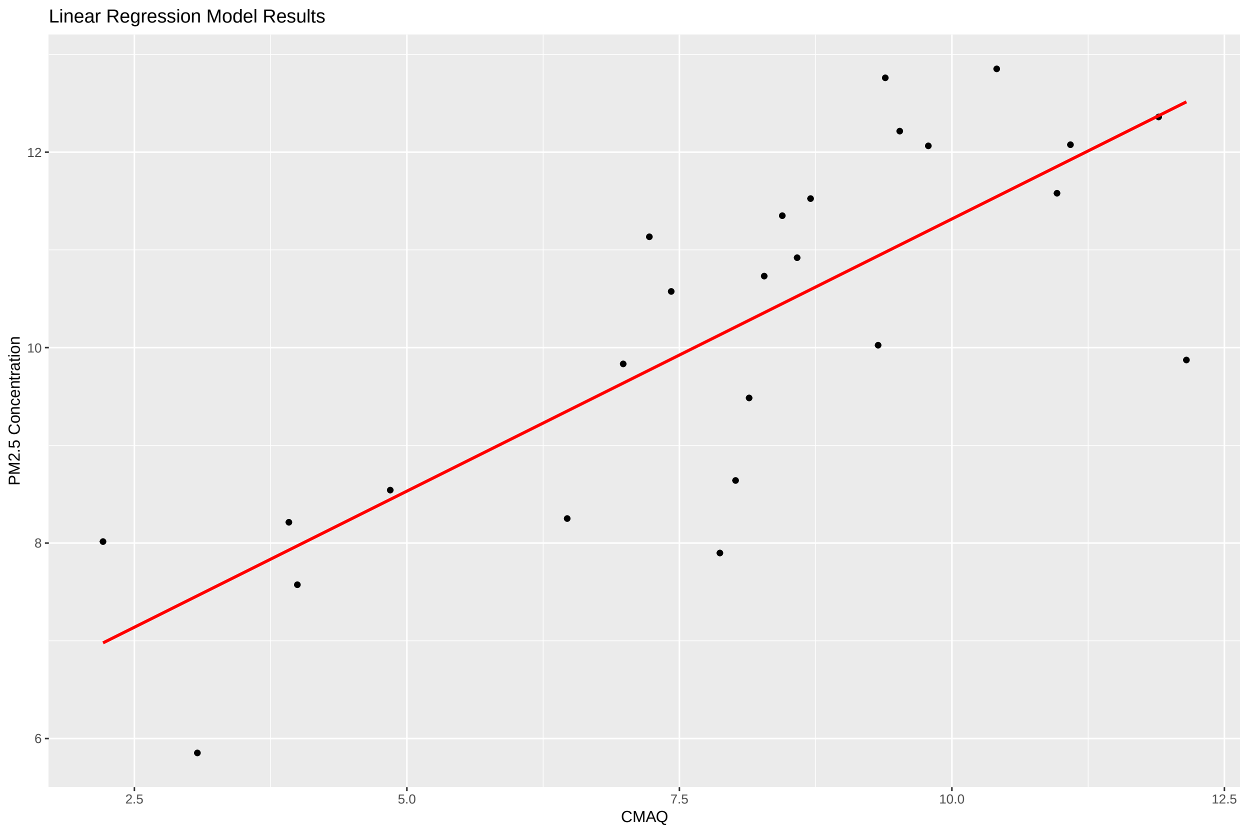
We determined that linear regression was the best model to use on this dataset since it produces the lowest RMSE value on the training dataset.

```r
#create a data frame with model names and RMSE values
library(knitr)

#create dataframe
Model_Name <- c('Linear Regression', 'kNN', 'Random Forest', 'Boosted Trees')
RMSE_values <- c(1.425438, 1.835633, 1.906, 1.878)
model_results <- data.frame(Model_Name, RMSE_values)

#use the kable function to create a formatted table
kable(model_results)
```

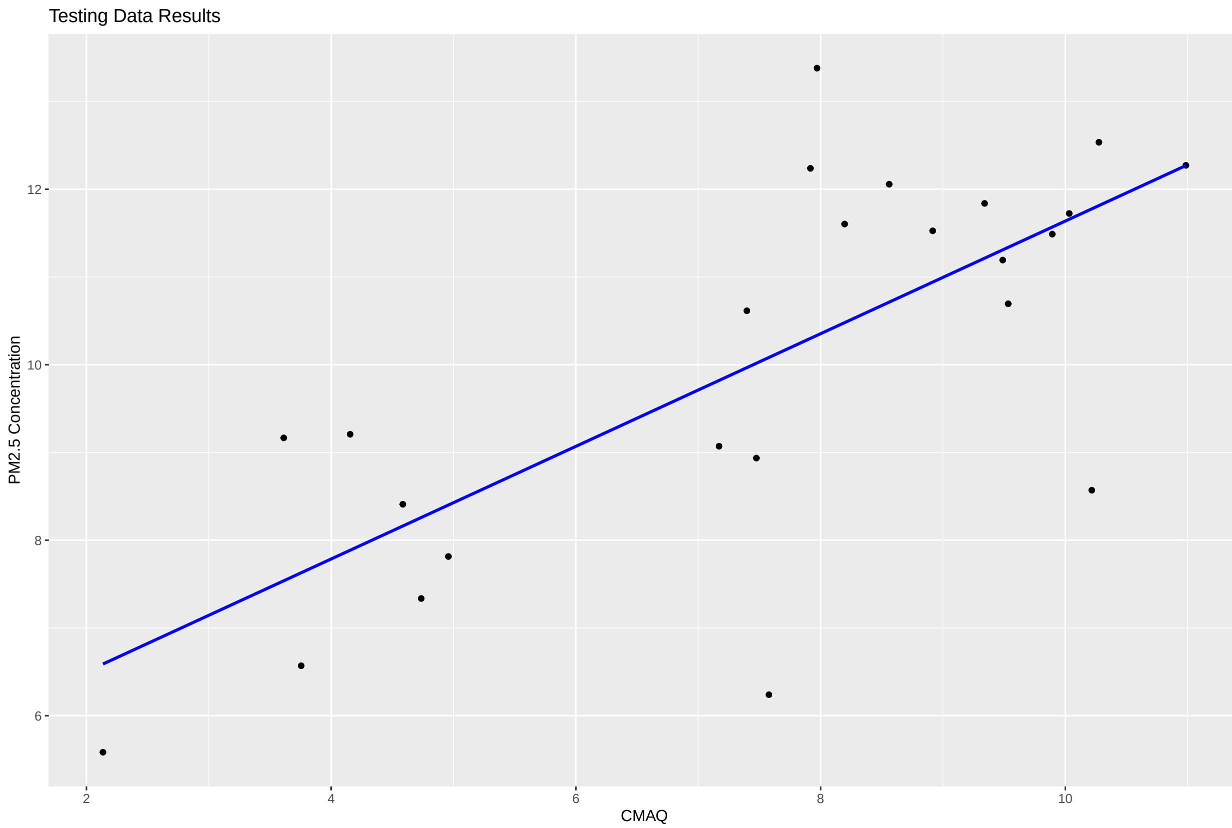
Model_Name	RMSE_values
Linear Regression	1.425438
kNN	1.835633
Random Forest	1.906000
Boosted Trees	1.878000

Now, we apply the linear regression model to the testing data to see the predicted values against the actual values of our best model and graph the results to compare the differences.

```r
#plot the predicted values against the actual values for our
#linear regression mode
ggplot(trainData, aes(x = CMAQ, y = value)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, color = "red") +
  labs(title = "Linear Regression Model Results",
       x = "CMAQ",
       y = "PM2.5 Concentration")
```




```
#plot the predicted values against the actual values for the testing data
ggplot(testData, aes(x = CMAQ, y = value)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, color = "blue") +
  labs(title = "Testing Data Results",
       x = "CMAQ",
       y = "PM2.5 Concentration")
```



Discussion of Performance of Best and Final Model

Discussion (Questions)

```
#create a new linear model without predicting on aod or CMAQ

model_less <- lm(value ~ county_pop, data = trainData)
predicted_less <- predict(model_less, newdata = testData)
summary(model_less)
```

Discussion (Questions): Policy Question

```
##
## Call:
## lm(formula = value ~ county_pop, data = trainData)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.071 -1.772  0.439  1.470  2.650
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  10.1611     0.3835  26.495   <2e-16 ***
## county_pop    0.2411     0.3990   0.604    0.552
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.915 on 23 degrees of freedom
## Multiple R-squared:  0.01562,    Adjusted R-squared:  -0.02718
## F-statistic: 0.365 on 1 and 23 DF,  p-value: 0.5517
```

```r
#find RMSE of less lm model
RMSE_less <- sqrt(mean((predicted_less - testData$value)^2))
RMSE_less
```

```
## [1] 2.137663
```

```r
RMSE
```

```
## [1] 1.425438
```

The RMSE value when we predict the value of the PM2.5 value variable is 2.13766 when we predict without using the aod or the QMAC variables as compared to 1.435438 when we predict with those variables. This means that using the aod and QMAC as predictors is quite helpful to the model and helps reduce the error in the model. The results of this analysis support the recent interests in policy for improving cost-effective methods for ground-based monitors since satellite predictions help predict ground-level concentrations according to the model above.

Discussion (Questions): Locations Closest and Furthest from Observed Values *In order to see where the model is more or less accurate, we did some testing on how the model performs in different geographic regions in the US.*

```r
#Code for selecting 8 states and finding RMSE using linear regression model. The only error that's comi

#filter the dataset for the 8 east coast states
east_coast <- c("Georgia", "Florida", "North Carolina", "South Carolina", "Maryland", "New Jersey", "Ne
east_coast_data <- dat1[dat1$state %in% east_coast, ]

#subset the test data for the east coast states
east_coast_test <- testData[testData$state %in% east_coast, ]

#fit the linear regression model using the east coast data
linear_model_east_coast <- lm(value ~ CMAQ + aod, data = east_coast_data)

#make predictions on the east coast test data using the linear regression model
east_coast_pred <- predict(linear_model_east_coast, newdata = east_coast_test)

#calculate the RMSE for the east coast test data
east_coast_rmse <- sqrt(mean(east_coast_test$value - east_coast_pred)^2)

#print the RMSE for the east coast test data
cat("RMSE for east coast states using linear regression model:", east_coast_rmse)
```

```
## RMSE for east coast states using linear regression model: 0.4172563
```

After predicting the values for 8 East Coast states using the linear model from above, we get a much lower RMSE value of 0.4172563 than the average RMSE for the model on the full testing set, which was RMSE ~ 1.4. This indicates that the model can much more accurately predict the PM2.5 values for East Coast states as compared to the rest of the contiguous US. We hypothesize that this is potentially due to there being a higher density of monitors placed in the East Coast and satellite data tracking more over East Coast region meaning that the CMAQ and AOD metrics more accurately predict the PM2.5 values.

Discussion (Questions): Which Variables Could Predict Model Performance *Once we create the models, knowing where our model's predictions would be more or less accurate is helpful for using our predictions properly. After referring to our exploratory data analysis again, we hypothesize that the population variable is the main variable that could predict how well our model performs since areas with higher populations are likely to have better measurement devices in those areas. CMAQ and AOD are likely not as helpful in differentiating how well the model performs because these metrics are meant to be compute in a standardized manner across all monitors and sites.*

Discussion (Questions): Extrapolating Results of the Analysis *In terms of using these models to predict other results, we are not very confident that this model would extend well to other regions of the United States. For example, there were not any monitors in Hawaii or Alaska, and we do not think that the model would necessarily be able to accurately predict data for these two states because these two states are not in the continental US and likely have lower levels of pollution than in the mainland US due to their lower populations as well.*

Discussion

Discussion: Using Data to Answer Analysis Question *CMAQ and AOD seem like promising candidates for predicting PM2.5 concentrations. If CMAQ and AOD are more cost effective for policy makers, governments can definitely consider using them for monitoring air pollution to avoid the costs of constructing monitors and maintaining existing ground-level PM2.5 monitors. The RMSE metrics for our best model with and without AOD and CMAQ data were computed and analyzed above, and they support this conclusion.*

Discussion: Reflect on Process of Conducting Project *Conducting this project was more challenging than we initially expected. However, after we mastered the steps for creating the various models, tweaking them to test different recipes and to perform cross validation became quite simple. This project definitely helped lay the foundation for future projects we could take on in data analysis with machine learning.*

Discussion: Reflect on Performance of Model *The model performed well, but not as well as we would have hoped that it would. It was a bit surprising to see that the first model that we used, linear regression, performed the best on this dataset. Using the other 3 methods (k-NN, random forest, and boosted trees), we had hoped that we would be able to increase the accuracy of the model, but the RMSE values for these models were actually lower than for the linear regression. Perhaps including more predictors in each of the models could have improved performacnce. Having a limited dataset, since we had less than 50 observations after calculating the mean of the monitors for each of the states, could also be one of the reasons the model performance was not as high as we would have hoped.*

Dicussion: Contributions of Group Members *The work for the project was split well amongst the group members and we collaborated effectively throughout the project. Tanvi wrote the introduction and discussion portions of the report and created model 4. Anish performed the wrangling, created model 1, and answered the questions portion for the best and final model. Elaina performed the expolratory data analysis and created models 2 and 3.*

Discussion: Acknowledgements *We would like to thank the TA Raquel for their help in configuring the coding for the project and to Dr. Peng for his guidance throughout.*