Code

CS02 - Predicting Annual Air Pollution

Tyler Kurpanek, Advaith Ravishankar, Stephanie Ugochukwu, Akhil Subbarao, Ahmad Rakha

Introduction

Air pollution is a major public health concern with serious impacts on human health, environmental quality, and socioeconomic development. Our study focuses on particulate matter 2.5 (PM2.5), which are fine inhalable particles with a diameter of 2.5 micrometers or smaller [1]. Common sources include, but are not limited to, construction sites, unpaved roads, and fires [2]. Due to their small size, PM2.5 particles can penetrate deep into the lungs and enter the bloodstream, causing a variety of short and long term health effects, such as reduced lung function growth in children [3], nonfatal heart attacks, and premature death for people with heart or lung disease. [4] These health risks highlight the urgency in understanding and mitigating PM2.5 exposure.

In our case study, we answer the question of "With what accuracy can we predict U.S. annual average air pollution concentrations?". Having accurate predictions of PM2.5 levels at a zip code level can be an extremely helpful tool to protect public health and environmental quality, as communities will be able to take necessary action. If we can reliably identify what factors would lead to a region having high concentrations, policymakers can allocate resources, and create helpful policies more effectively.

Branching from our main question, we explore a secondary question, "How does air pollution in neighboring counties influence each other, given the geo-spatial nature of air quality data?". As PM2.5 does not follow geographical borders, it can spread across regions due to weather patterns, industrial activity, and urban development[5]. In fact, pollutants can travel hundreds of miles by wind, which can make it harder to meet pollution standards for areas that are downwind [6]. Having an understanding of these spatial dependencies is vital to design regional mitigation strategies to handle pollution.

Our goal with this case study is to investigate the factors that influence the U.S. annual average for PM2.5 concentrations, as well as the geospatial relationships between neighboring regions. By focusing on predictive accuracy at a zip code level, we provide actionable insights for communities to better protect public health. Further, by understanding the spread of PM2.5 across regional boundaries, our case study offers crucial context for developing effective mitigation strategies. This research provides a basis to aid policymakers in allocating resources, and designing targeted interventions that address both the sources and impacts of air pollution.

Questions

With what accuracy can we predict US annual average air pollution concentrations? As air pollution is a geo-spatial data, how do neighboring counties air pollution effect each other?

Load packages

Hide

```r
library(OCSdata)
library(datasauRus)
library(tidyverse)
library(ggplot2)
library(igraph)
library(geosphere)
library(dplyr)
library(olsrr)
library(tidymodels)
library(maps)
library(rnaturalearthdata)
library(skimr)
```

The Data

The data comes from the US Environmental Protection Agency (EPA) [7], the National Aeronautics and Space Administration (NASA) [8], the US Census 2010 [9], and the National Center for Health Statistics (NCHS) [10].

There are 48 features with values for 876 observations. Each record has variables in these categories: General, Emission (nei), Development (imp), Road Density (pri) and Education. Given below is a description of each variable.

General

1. ID: Monitor number

2. fips: Federal information processing standard number for the county where the monitor is located
3. lat: Latitude of the monitor in degrees
4. long: Longitude of the monitor in degrees
5. county: County where the monitor is located
6. city: City where the monitor is located
7. zcta: Zip Code Tabulation [11] Area where the monitor is located
8. zcta_area: Land area of the zip code area in meters squared
9. zcta_pop: Population in the zip code area
10. county_area: Land area of the county of the monitor in meters squared
11. county_pop: Population of the county of the monitor
12. popdens_zcta: Population density (number of people per kilometer squared area of zcta)
13. urc2013: 2013 Urban-rural classification [12] of the county where the monitor is located
14. urc2006: 2006 Urban-rural classification [13] of the county where the monitor is located
15. pov: Percentage of people in zcta area where the monitor is that lived in poverty in 2008 [14]
16. aod: Aerosol Optical Depth measurement from a NASA satellite
17. CMAQ: Estimated values of air pollution from a computational model called Community Multiscale Air Quality (CMAQ) [15]

Emission (nei)

The emission data follow theformat:

log_nei_2008_pmXX_sum_YYYY: Tons of emissions from major sources data base (annual data) sum of all sources within a circle with a radius of YYYY meters of distance around the monitor (Natural log) for $PM_{X.X}$

18. log_nei_2008_pm25_sum_10000
19. log_nei_2008_pm25_sum_15000
20. log_nei_2008_pm25_sum_25000
21. log_nei_2008_pm10_sum_10000
22. log_nei_2008_pm10_sum_15000
23. log_nei_2008_pm10_sum_25000

Development (imp)

The development data follow the format

imp_aXXXX: Impervious surface measure Within a circle with a radius of XXXX meters around the monitor

24. imp_a500
25. imp_a1000
26. imp_a5000
27. imp_a10000
28. imp_a15000

Road Density (pri)

29. Log_dist_to_prisec: Log (Natural log) distance to a primary or secondary road from the monitor

The following variables have the format: log_pri_length_XXXX where Count of primary road length in meters in a circle with a radius of XXXX meters around the monitor (Natural log)

30. log_pri_length_5000
31. log_pri_length_10000
32. log_pri_length_10000
33. log_pri_length_15000
34. log_pri_length_25000

The following variables have the format: log_prisec_length_XXXX Count of primary and secondary road length in meters in a circle with a radius of XXXX meters around the monitor (Natural log)

35. log_prisec_length_500
36. log_prisec_length_1000
37. log_prisec_length_1000
38. log_prisec_length_10000
39. log_prisec_length_15000
40. log_prisec_length_25000

Education (Data from Census)

41. nohs: Percentage of people in zcta area where the monitor is that do not have a high school degree
42. somehs: Percentage of people in zcta area where the monitor whose highest formal educational attainment was some high school education
43. hs: Percentage of people in zcta area where the monitor whose highest formal educational attainment was completing a high school degree
44. somecollege: Percentage of people in zcta area where the monitor whose highest formal educational attainment was completing some college education
45. associate: Percentage of people in zcta area where the monitor whose highest formal educational attainment was completing an associate degree
46. bachelor: Percentage of people in zcta area where the monitor whose highest formal educational attainment was a bachelor's degree
47. grad: Percentage of people in zcta area where the monitor whose highest formal educational attainment was a graduate degree
48. hs_orless: Percentage of people in zcta area where the monitor whose highest formal educational attainment was a high school degree or less (sum of nohs, somehs, and hs)

Data Import

Hide

```
pm <- read_csv("data/pm25_data.csv")
```

Data Wrangling

Hide

```
# Identify numeric columns
numeric_cols <- pm |>
  select(where(is.numeric))

# Check skewness of numeric columns
library(e1071)
skewness_values <- sapply(numeric_cols, skewness, na.rm = TRUE)

# Define threshold for skewness to log-transform (e.g., skewness > 1 or < -1)
skewed_cols <- names(skewness_values[abs(skewness_values) > 1])

variables_to_transform <- c("imp_a10000", "imp_a15000",
                            "nohs", "somehs", "somecollege",
                            "associate", "bachelor", "grad",
                            "pov", "aod")

# Log-transform the skewed columns that make sense(adding a small constant if necessary)
data_transformed <- pm |>
  mutate(across(all_of(variables_to_transform), ~ log(. + 1), .names = "log_{col}"))


data_long <- pivot_longer(
  data_transformed,
  cols = starts_with("imp_a"),
  names_to = "Measurement_imp_a",
  values_to = "Value_imp_a"
)

data_long <- pivot_longer(
  data_long,
  cols = contains("nei"),
  names_to = "Measurement_nei",
  values_to = "Value_nei"
)
data_long <- pivot_longer(
  data_long,
  cols = contains("pri"),
  names_to = "Measurement_pri",
  values_to = "Value_pri"
)
```

In our data preprocessing phase, we systematically evaluated the skewness of numeric variables using the e1071 package and identified specific variables requiring logarithmic transformation. We selected ten key variables for transformation: imp_a10000, imp_a15000 (development metrics), educational attainment percentages (nohs, somehs, somecollege, associate, bachelor, grad), poverty indicators (pov), and aerosol optical depth measurements (aod). While population-related variables showed skewness, we maintained their original scale to preserve direct interpretability of population effects in our models. The transformed data was then restructured into a long format using pivot_longer() for three categories: impervious surface measurements, National Emissions Inventory data, and road density metrics.

EDA

Variables

To visualize the types of variables in our dataset, we used the skimr package to understand the data we are working with.

Hide

```
skimr::skim(pm)
```

Data summary

Name	pm
Number of rows	876
Number of columns	50
_______________________	
Column type frequency:	
character	3
numeric	47

Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
state	0	1	4	20	0	49	0
county	0	1	3	20	0	471	0
city	0	1	4	48	0	607	0

Variable type: numeric

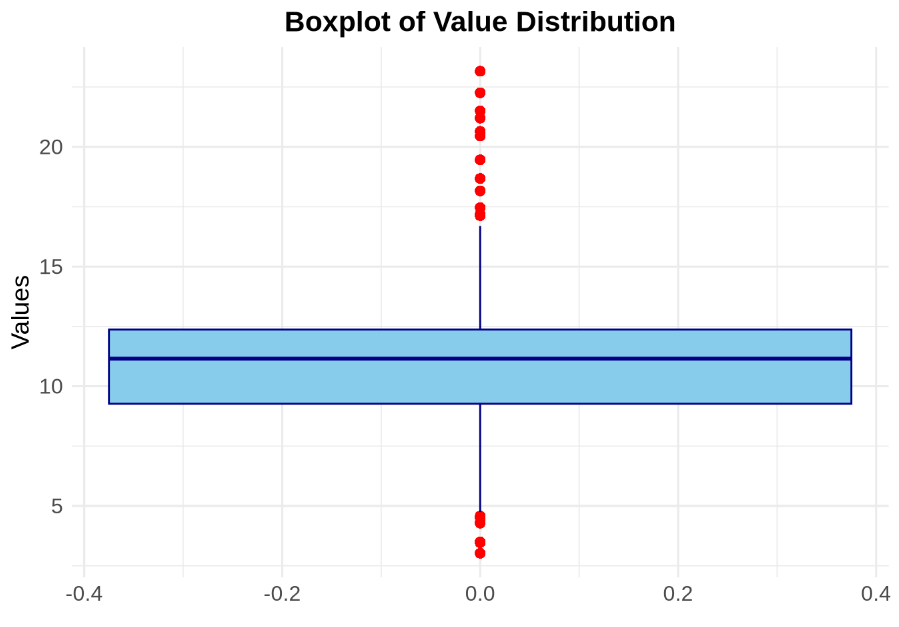
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	
id	0	1	26987.96	1.578761e+04	1003.00	13089.15	26132.00	39118.00	5.60391
value	0	1	10.81	2.580000e+00	3.02	9.27	11.15	12.37	2.31600
fips	0	1	26987.89	1.578763e+04	1003.00	13089.00	26132.00	39118.00	5.60390
lat	0	1	38.48	4.620000e+00	25.47	35.03	39.30	41.66	4.84000
lon	0	1	-91.74	1.496000e+01	-124.18	-99.16	-87.47	-80.69	-6.80400
CMAQ	0	1	8.41	2.970000e+00	1.63	6.53	8.62	10.24	2.31300
zcta	0	1	50890.29	2.778447e+04	1022.00	28788.25	48172.00	74371.00	9.92020
zcta_area	0	1	183173481.91	5.425989e+08	15459.00	14204601.75	37653560.50	160041508.25	8.164821
zcta_pop	0	1	24227.58	1.777216e+04	0.00	9797.00	22014.00	35004.75	9.53970
imp_a500	0	1	24.72	1.934000e+01	0.00	3.70	25.12	40.22	6.96100
imp_a1000	0	1	24.26	1.802000e+01	0.00	5.32	24.53	38.59	6.75000
imp_a5000	0	1	19.93	1.472000e+01	0.05	6.79	19.07	30.11	7.46000
imp_a10000	0	1	15.82	1.381000e+01	0.09	4.54	12.36	24.17	7.20900
imp_a15000	0	1	13.43	1.312000e+01	0.11	3.24	9.67	20.55	7.11000
county_area	0	1	3768701992.12	6.212830e+09	33703512.00	1116536297.50	1690826566.50	2878192209.00	5.19472
county_pop	0	1	687298.44	1.293489e+06	783.00	100948.00	280730.50	743159.00	9.81860
log_dist_to_prisec	0	1	6.19	1.410000e+00	-1.46	5.43	6.36	7.15	1.04500
log_pri_length_5000	0	1	9.82	1.080000e+00	8.52	8.52	10.05	10.73	1.20500
log_pri_length_10000	0	1	10.92	1.130000e+00	9.21	9.80	11.17	11.83	1.30200
log_pri_length_15000	0	1	11.50	1.150000e+00	9.62	10.87	11.72	12.40	1.35900
log_pri_length_25000	0	1	12.24	1.100000e+00	10.13	11.69	12.46	13.05	1.43600
log_prisec_length_500	0	1	6.99	9.500000e-01	6.21	6.21	6.21	7.82	9.40000
log_prisec_length_1000	0	1	8.56	7.900000e-01	7.60	7.60	8.66	9.20	1.04700
log_prisec_length_5000	0	1	11.28	7.800000e-01	8.52	10.91	11.42	11.83	1.27800
log_prisec_length_10000	0	1	12.41	7.300000e-01	9.21	11.99	12.53	12.94	1.38500
log_prisec_length_15000	0	1	13.03	7.200000e-01	9.62	12.59	13.13	13.57	1.44100
log_prisec_length_25000	0	1	13.82	7.000000e-01	10.13	13.38	13.92	14.35	1.52300
log_nei_2008_pm25_sum_10000	0	1	3.97	2.350000e+00	0.00	2.15	4.29	5.69	9.12000
log_nei_2008_pm25_sum_15000	0	1	4.72	2.250000e+00	0.00	3.47	5.00	6.35	9.42000
log_nei_2008_pm25_sum_25000	0	1	5.67	2.110000e+00	0.00	4.66	5.91	7.28	9.65000
log_nei_2008_pm10_sum_10000	0	1	4.35	2.320000e+00	0.00	2.69	4.62	6.07	9.34000
log_nei_2008_pm10_sum_15000	0	1	5.10	2.180000e+00	0.00	3.87	5.39	6.72	9.71000
log_nei_2008_pm10_sum_25000	0	1	6.07	2.010000e+00	0.00	5.10	6.37	7.52	9.88000
popdens_county	0	1	551.76	1.711510e+03	0.26	40.77	156.67	510.81	2.68219
popdens_zcta	0	1	1279.66	2.757490e+03	0.00	101.15	610.35	1382.52	3.04188
nohs	0	1	6.99	7.210000e+00	0.00	2.70	5.10	8.80	1.00000
somehs	0	1	10.17	6.200000e+00	0.00	5.90	9.40	13.90	7.22000

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	
hs	0	1	30.32	1.140000e+01	0.00	23.80	30.75	36.10	1.000000
somecollege	0	1	21.58	8.600000e+00	0.00	17.50	21.30	24.70	1.000000
associate	0	1	7.13	4.010000e+00	0.00	4.90	7.10	8.80	7
bachelor	0	1	14.90	9.710000e+00	0.00	8.80	12.95	19.22	1.000000
grad	0	1	8.91	8.650000e+00	0.00	3.90	6.70	11.00	1.000000
pov	0	1	14.95	1.133000e+01	0.00	6.50	12.10	21.22	6.590000
hs_orless	0	1	47.48	1.675000e+01	0.00	37.92	48.65	59.10	1.000000
urc2013	0	1	2.92	1.520000e+00	1.00	2.00	3.00	4.00	6.000000
urc2006	0	1	2.97	1.520000e+00	1.00	2.00	3.00	4.00	6.000000
aod	0	1	43.70	1.956000e+01	5.00	31.66	40.17	49.67	1.430000

Air Pollution Values Distribution

Hide

```
ggplot(data_long, aes(y = value)) +
  geom_boxplot(fill = "skyblue", color = "darkblue", outlier.color = "red", outlier.size = 2) +
  labs(
    title = "Boxplot of Value Distribution",
    y = "Values",
    x = ""
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```



From the distribution and skim of the dataset, the air pollution values are centered around 10.8 with a standard deviation of 2.6. Most of the data is concentrated between 9-15 with a large number of outlier measurements above 15.

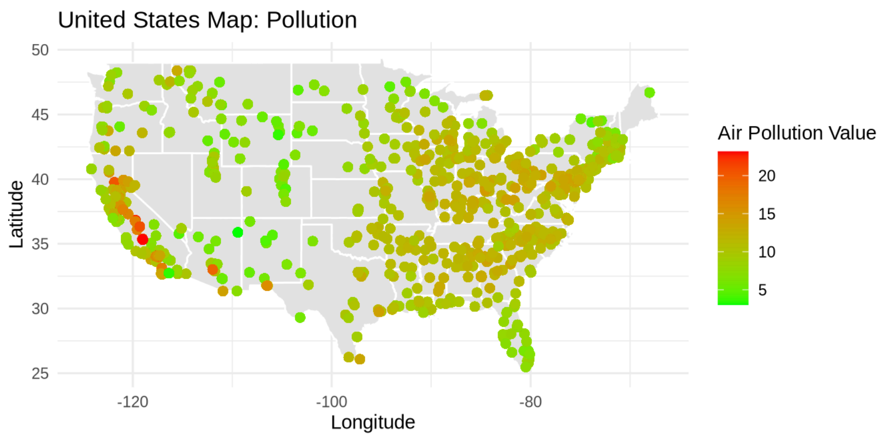
Data Distribution

Geospatial Pollution

Lets first visualize the distribution of our data across the United States of America

Hide

```r
# Get US map data
us_map <- map_data("state")

# Plot the map
ggplot() +
  geom_polygon(data = us_map, aes(x = long, y = lat, group = group),
               fill = "gray90", color = "white") +
  geom_point(data = data_long, aes(x = lon, y = lat, color = value),
             size = 2, alpha = 0.8) +
  scale_color_gradient(low = "green", high = "red", name = "Air Pollution Value") +
  theme_minimal() +
  labs(title = "United States Map: Pollution",
       x = "Longitude",
       y = "Latitude") +
  coord_fixed(1.3)
```



The map visualizes data points across the United States, with each point representing a geographical location's value (e.g., pollution level) using a color gradient (green to red). The points cover the coasts of the United States well, with denser clusters in areas with higher populations, such as the East Coast, the Midwest, and parts of California. However, there are a lack of data points observed in the mid to western non coastal United States, particularly in Nevada, Texas, Colorado and the rest of the middle of the country.

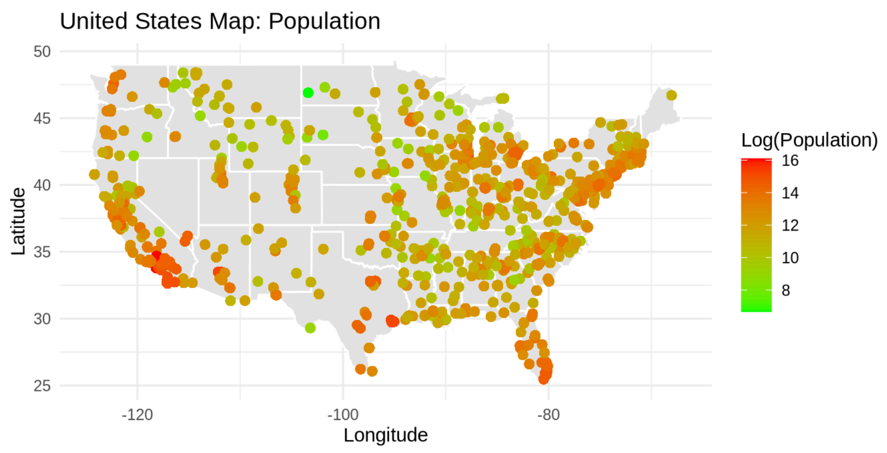
Urbanized areas (e.g., the East Coast, southern California) show more frequent yellow to red points, likely reflecting higher values in densely populated or industrial zones

Rural and less populated areas (e.g., parts of the Midwest and Mountain regions) show more green points, indicating lower values.

Geospatial Population

Hide

```r
# Get US map data
us_map <- map_data("state")

# Plot the map
ggplot() +
  geom_polygon(data = us_map, aes(x = long, y = lat, group = group),
               fill = "gray90", color = "white") +
  geom_point(data = data_long, aes(x = lon, y = lat, color = log(county_pop)),
             size = 2, alpha = 0.8) +
  scale_color_gradient(low = "green", high = "red", name = "Log(Population)") +
  theme_minimal() +
  labs(title = "United States Map: Population",
       x = "Longitude",
       y = "Latitude") +
  coord_fixed(1.3)
```



Both maps show a strong association between urbanized regions and higher values (red or orange points). Urban areas in southern California, the East Coast, and the Midwest tend to have higher pollution values and population density

Data Correlation

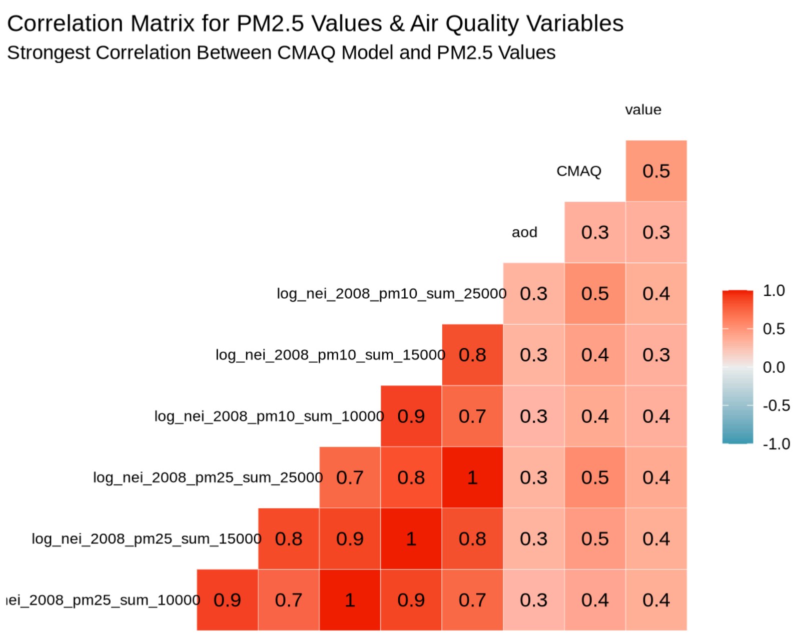
In our data set, there are variables that may influence the PM2.5 monitor values that are being measured in each county. In order to answer our main question, we must first determine what factors have the greatest correlation to the outcome measured by the PM2.5 monitors.

Outcome vs. Emissions (nei)

First, we'd like to explore the correlation between our outcome, and the different metrics that the data set includes that measure pollution. The CMAQ variable represents a model that measures the estimated values of air pollution, however this model does not utilize data from the PM2.5 monitor. Since CMAQ is still a pollution measuring metric, it is important that we determine if this model correlates to the PM2.5 monitor values and will be useful for building a new model to predict PM2.5 values. The NEI variables represent the values collected from the National Emissions Inventory. They represent the tons of emissions in a circle around the monitor. The radius of this circle is in meters and the numeric value at the end of each "nei" variable represents the size of the radius. The aod variable represents the Aerosol Optical Depth measurement from a NASA satellite. This metric approximates the amount of particulate pollution through the utilization of laser diffraction.

Hide

```
# producing pollution correlation matrix
data_transformed |>
  select(contains("nei"), aod, CMAQ, value) |> # selecting air quality variables
  GGally::ggcorr(hjust = .85, size = 3,
               layout.exp=2, label = TRUE) +
  ggplot2::labs(title = "Correlation Matrix for PM2.5 Values & Air Quality Variables",
              subtitle = "Strongest Correlation Between CMAQ Model and PM2.5 Values")
```



All of the pollution metrics had positive correlations to our PM2.5 outcome variable “value,” as demonstrated by the most right column being red. This means that as the values for these metrics increase, so will the PM2.5 values. The CMAQ model in particular had the strongest correlation, supporting its potential to be used to build a model that can predict PM2.5 values.

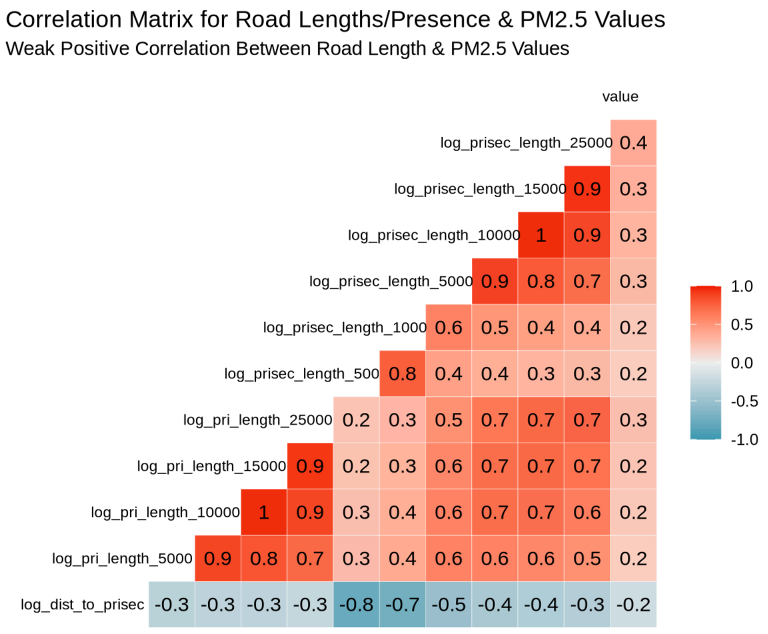
Therefore, all of these variables will be used for our analysis.

Outcome vs. Road Density (pri)

Next, we’d like to explore the influence of road length/presence on our outcome variable. The “Log_dist_to_prisec” variable represents the distance to a primary or secondary road from the monitor. The variables with “pri_length” represent the count of primary road length in meters in a circle with a radius the size of the value at the end of the variable name. The variables with “prisec_length” represent the count of primary and secondary road length in meters in a circle with a radius the size of the value at the end of the variable name. Roads and air quality may be a valuable relationship to explore for our model because of how the length and types of roads may signify the dependency of cars for the population residing in the county. On average, a typical passenger vehicle emits about 4.6 metric tons of CO2 per year [16]. Using roads as a way to explore how car emissions impacts air quality will provide important insight for building our model.

Hide

```
# producing road correlation matrix
data_transformed |>
  select(contains("pri"), value) |> # selecting all values associated with road density
  GGally::ggcorr(hjust = .85, size = 3,
               layout.exp=2, label = TRUE) +
  ggplot2::labs(title = "Correlation Matrix for Road Lengths/Presence & PM2.5 Values",
              subtitle = "Weak Positive Correlation Between Road Length & PM2.5 Values")
```



	log_dist_to_prisec	log_pri_length_5000	log_pri_length_10000	log_pri_length_15000	log_pri_length_25000	log_prisec_length_500	log_prisec_length_1000	log_prisec_length_5000	log_prisec_length_10000	log_prisec_length_15000	log_prisec_length_25000
log_prisec_length_25000											
log_prisec_length_15000											0.9
log_prisec_length_10000										1	0.9
log_prisec_length_5000									0.9	0.8	0.7
log_prisec_length_1000								0.6	0.5	0.4	0.4
log_prisec_length_500							0.8	0.4	0.4	0.3	0.3
log_pri_length_25000						0.2	0.3	0.5	0.7	0.7	0.7
log_pri_length_15000					0.9	0.2	0.3	0.6	0.7	0.7	0.7
log_pri_length_10000				1	0.9	0.3	0.4	0.6	0.7	0.7	0.6
log_pri_length_5000			0.9	0.8	0.7	0.3	0.4	0.6	0.6	0.6	0.5
value	-0.3	0.2	0.2	0.2	0.3	0.2	0.2	0.3	0.3	0.3	0.4

All of the road length variables have a positive correlation with the PM2.5 values, however this correlation is weak. This is demonstrated by their associated correlation values being pink in the most right column. The “Log_dist_to_prisec” variable has a negative correlation with the PM2.5 values meaning that as the distance between a primary/second road increases, there will be a small decrease in PM2.5 values. These correlation values imply that road length/presence do have an impact on the air quality pollution being measured by the monitors.

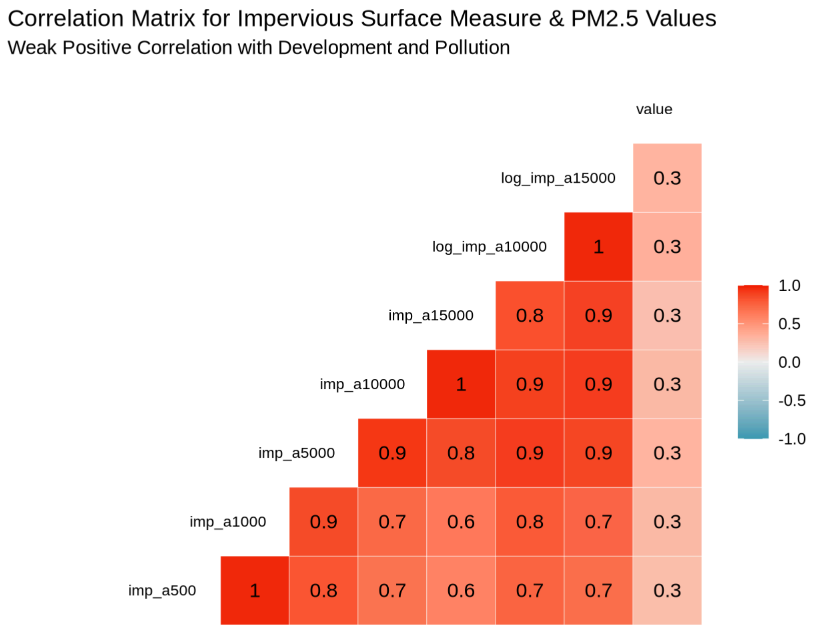
Roads alone aren’t the only indicator of polluter presence in an area. The next variables we’d like to explore are the impervious surface measure variables, represented by “imp”, which describe the development of the area the monitor is located in.

Outcome vs. Development (imp)

The imp variables describe the amount of impervious surface within a circle around the monitor. The radius of the circle is the value at the end of the variable name measured in meters. Impervious surfaces are roads, concrete, parking lots, and buildings. The increase in development and therefore industrial activity is associated with environmentally unfriendly outcomes such as higher emissions due to consumption of fossil fuels. We want to know if impervious surface area may serve as a good indicator for air quality for our model.

Hide

```
# producing development matrix
data_transformed |>
  select(contains("imp"), value) |> # selecting all values associated with impervious surface measure
  GGally::ggcorr(hjust = .85, size = 3,
               layout.exp=2, label = TRUE) +
  ggplot2::labs(title = "Correlation Matrix for Impervious Surface Measure & PM2.5 Values",
             subtitle = "Weak Positive Correlation with Development and Pollution")
```



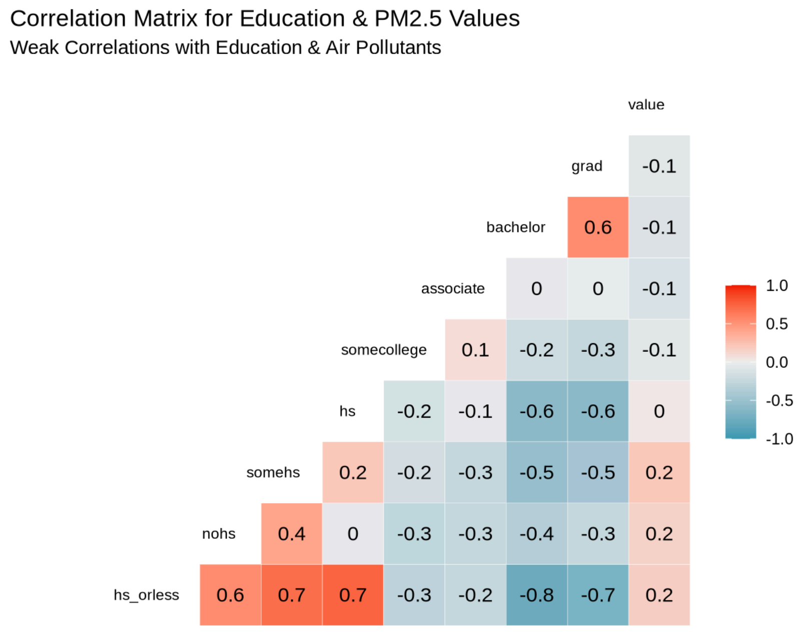
The matrix shows a weak positive correlation between impervious surfaces and our outcome variable. This means that as the amount of impervious surfaces increases, the monitor will measure more pollutants, however this relationship is quite weak. Development alone may not be a great indicator of pollutants, but maybe we can connect it to other results of development to build our model.

Outcome vs Education

Education describes the percentage of people's highest degree of education in a county. An education may inform people about harmful practices which cause high degrees of pollution so education level may have some relationship with our outcome.

Hide

```
# education correlation matrix
data_transformed |>
  select(hs_orless, nohs, somehs, hs, somecollege, associate, bachelor, grad, value) |> # selecting education variables
  GGally::ggcorr(hjust = .85, size = 3,
               layout.exp=2, label = TRUE) +
  ggplot2::labs(title = "Correlation Matrix for Education & PM2.5 Values",
             subtitle = "Weak Correlations with Education & Air Pollutants")
```



From the correlation matrix, all forms of education are loosely correlated with air pollution as the values are between -0.1 and 0.2 which signify little correlation. This gives us the insight that education parameters do no predict air pollution.

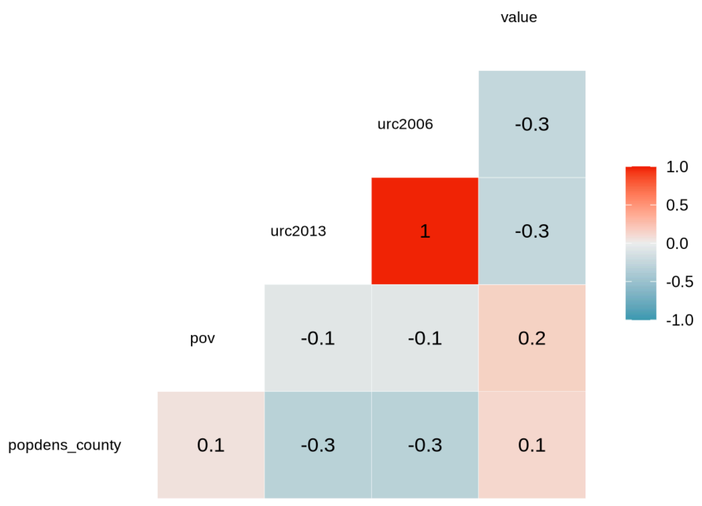
Outcome vs Remaining Factors

The remaining factors are urbanization classification rating, poverty of region and population density. Urbanization, poverty, and population density are all variables that are linked to development and may be able to provide stronger context to build an effective model for predicting air pollution concentrations.

Hide

```
# remaining factors matrix
data_transformed |>
  select(popdens_county, pov, urc2013, urc2006, value) |> # selecting poverty, urbanization, & population density variables
  GGally::ggcorr(hjust = .85, size = 3,
               layout.exp=2, label = TRUE) +
  ggplot2::labs(title = "Correlation Matrix for Urbanization, Population Desnity, Poverty, & PM2.5 Values",
             subtitle = "Weak Negative Correlation with URC Metric & Air Pollutants")
```

Correlation Matrix for Urbanization, Population Desnity, Poverty, & PM2.5 Values
Weak Negative Correlation with URC Metric & Air Pollutants



The URC metric is seen to have a small negative correlation to our outcome variable. This means that as a county becomes more rural, there will be less pollutants measured in the air. Poverty rates and population density have weak positive correlations to our outcome variable, so as these variables increase there will also be a slight increase pollutants measured in the air.

Now that we have a general idea of how the variables individually correlate to our outcome, we'd like to use them to build a model that can be use to predict air pollution concentrations. Based on the correlation matrices CMAQ will probably be the most useful variable for our model. However, while exploring the correlation of our variables with the outcome, we also saw strong correlations amongst the "nei" variables, the road length variables, and road length vs. road distance from the monitor. This result implies that when trying to build our model to predict air pollution concentration, it won't be enough to simply use one variable as a predictor. The mixing of these variables may prove to provide more context onto the state of air quality in the U.S.

Analysis

Overview

To predict the US annual average air pollution concentrations we made use of 2 models

1. **Multivariate Linear Regression**
2. **Random Forest Regression**.

We evaluated each model on based on R^2 for variance explanability, and root mean squared error for quantifying performance.

Multivariate Linear Regression

Setup

Hide

```
# Create train/test split
set.seed(1234)
pm_split <- initial_split(data = pm, prop = 3/4)
train_pm <- training(pm_split)
test_pm <- testing(pm_split)

# Create recipe for linear regression
lm_rec <- recipe(train_pm) |>
  update_role(everything(), new_role = "predictor") |>
  update_role(value, new_role = "outcome") |>
  update_role(id, new_role = "id variable") |>
  update_role("fips", new_role = "county id") |>
  step_string2factor("state", "county", "city") |>
  step_rm("city") |>
  step_rm("county") |>
  step_rm("zcta") |>
  step_corr(all_numeric()) |>
  step_nzv(all_numeric())

# Create linear model specification
lm_spec <- linear_reg() |>
  set_engine("lm") |>
  set_mode("regression")

# Create workflow
lm_wflow <- workflow() |>
  add_recipe(lm_rec) |>
  add_model(lm_spec)

# Fit the model
lm_fit <- fit(lm_wflow, data = train_pm)
```

Model Performance

Hide

```
# Cross validation
set.seed(123)
lm_resamples <- fit_resamples(
  lm_wflow,
  resamples = vfold_cv(train_pm, v = 4),
  metrics = metric_set(rmse, rsq)
)

# Collect metrics
collect_metrics(lm_resamples)
```

```
## # A tibble: 2 × 6
##   .metric .estimator  mean     n std_err .config
##   <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 rmse    standard   1.82      4  0.0402 Preprocessor1_Model1
## 2 rsq     standard   0.504     4  0.0330 Preprocessor1_Model1
```

Model Performance

The multivariate linear regression model demonstrates moderate predictive capability for US annual average air pollution concentrations:

The model achieves an R^2 value of 0.504, indicating it explains approximately 50.4% of the variance in PM2.5 concentrations. This suggests the model captures about half of the variability in air pollution levels.

The root mean square error (RMSE) is 1.82, meaning that on average, the model's predictions deviate by 1.82 μg/m^3 from the actual PM2.5 values. Given that PM2.5 values in the data set typically range from 3 to 23 μg/m^3, this represents a moderate level of prediction accuracy.

The model's performance metrics were obtained through 4-fold cross-validation, with relatively low standard errors (0.0402 for RMSE and 0.0330 for R^2), indicating stable performance across different subsets of the data.

This performance suggests that while the linear regression model captures significant patterns in air pollution variation, there remains substantial unexplained variability that might be better captured by more complex models like the random forest approach.

Hide

```
lm_fit |>
  extract_fit_parsnip() |>
  tidy() |>
  arrange(desc(abs(estimate))) |>
  head(10)
```

```
## # A tibble: 10 × 5
##    term              estimate std.error statistic       p.value
##    <chr>                <dbl>     <dbl>     <dbl>         <dbl>
##  1 (Intercept)         243.       94.5       2.57 0.0105
##  2 stateNew Mexico      -7.64      1.31     -5.85 0.00000000823
##  3 stateColorado        -6.49      1.13     -5.75 0.0000000141
##  4 stateWyoming         -5.99      1.39     -4.29 0.0000207
##  5 stateWashington      -5.57      1.85     -3.01 0.00274
##  6 stateArizona         -5.14      1.45     -3.55 0.000423
##  7 stateNevada          -5.12      1.84     -2.78 0.00558
##  8 stateNorth Dakota    -5.08      1.54     -3.29 0.00105
##  9 stateIdaho           -4.82      1.69     -2.85 0.00456
## 10 stateMaine           -4.78      2.23     -2.14 0.0326
```

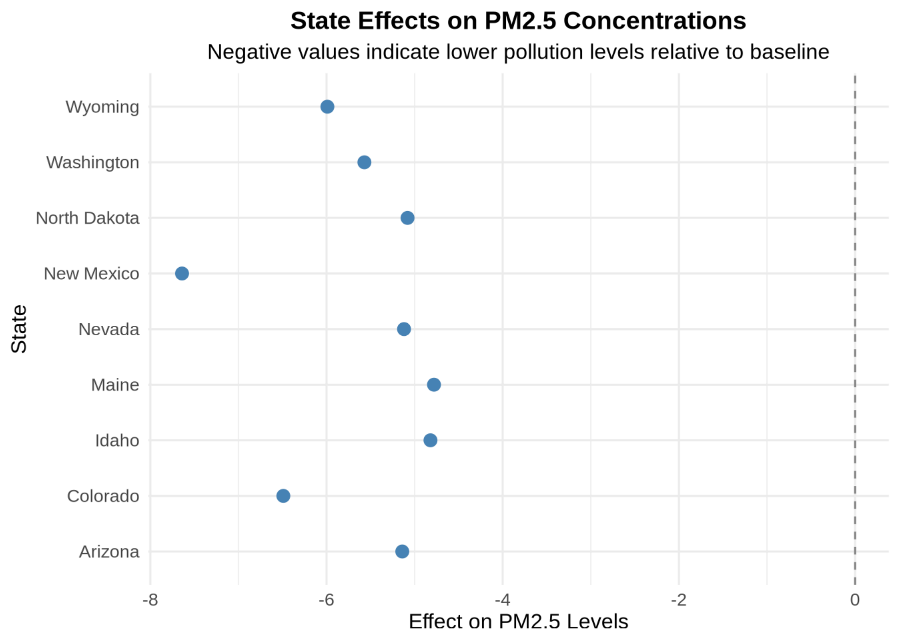
Strongest Negative Effects: New Mexico shows the strongest negative effect, with PM2.5 levels 7.64 units lower than the baseline state ($p < 0.001$) Colorado follows with concentrations 6.49 units lower ($p < 0.001$) Wyoming shows PM2.5 levels 5.99 units lower ($p < 0.001$)

Moderate Negative Effects Washington (-5.57 units, $p < 0.01$) Arizona (-5.14 units, $p < 0.001$) Nevada (-5.12 units, $p < 0.01$)

Geographic Pattern The coefficients reveal a clear geographic pattern where western and mountain states tend to have significantly lower PM2.5 concentrations compared to other regions. This aligns with expectations given factors like: Lower population density Higher elevation Different industrial profiles Geographic barriers affecting pollution dispersion The model's intercept of 243 ($p < 0.05$) represents the baseline PM2.5 level before accounting for state effects and other variables.

Hide

```
ggplot() +
  geom_point(aes(
    x = c(-7.64, -6.49, -5.99, -5.57, -5.14, -5.12, -5.08, -4.82, -4.78),
    y = c("New Mexico", "Colorado", "Wyoming", "Washington",
          "Arizona", "Nevada", "North Dakota", "Idaho", "Maine")),
    color = "steelblue", size = 3) +
  geom_vline(xintercept = 0, linetype = "dashed", color = "gray50") +
  labs(
    x = "Effect on PM2.5 Levels",
    y = "State",
    title = "State Effects on PM2.5 Concentrations",
    subtitle = "Negative values indicate lower pollution levels relative to baseline"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.text = element_text(size = 10),
    axis.title = element_text(size = 12)
  )
```



Hide

```
# Make predictions on the test data
test_predictions <- predict(lm_fit, new_data = test_pm)

# Create a data frame for plotting
error_data <- data.frame(
  Actual = test_pm$value,
  Predicted = test_predictions$.pred
)

# Function to calculate RMSE for a specific range
calculate_rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}

r_squared <- 1 - sum((error_data$Actual - error_data$Predicted)^2) /
                  sum((error_data$Actual - mean(error_data$Actual))^2)

# Overall RMSE on test

rmse <- calculate_rmse(
  actual = error_data$Actual,
  predicted = error_data$Predicted
)

cat("Linear Model RMSE on Test set: :", rmse)
```

```
## Linear Model RMSE on Test set: : 1.95045
```

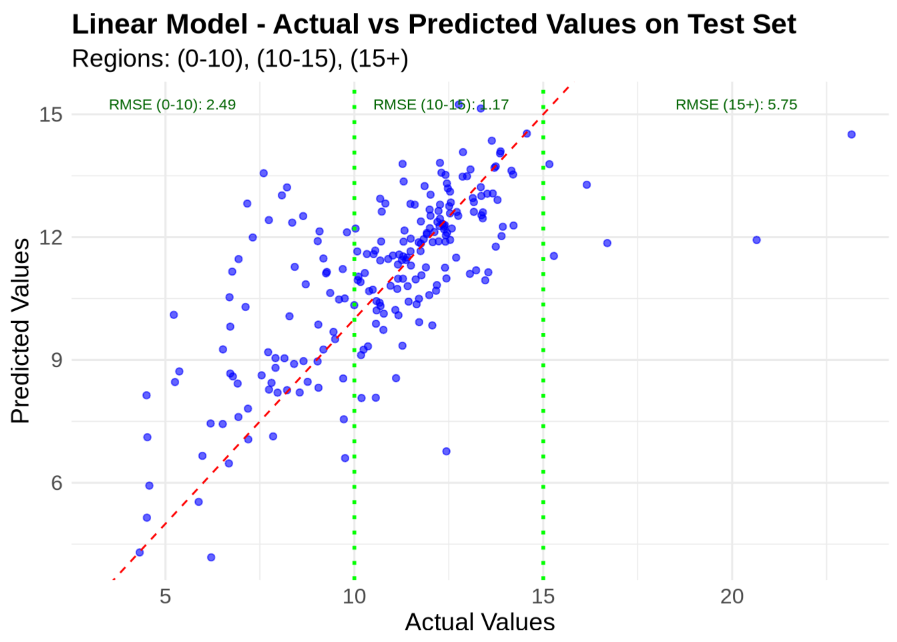
Hide

```
cat("\nLinear Model R^2 on Test set: :", r_squared)
```

```
##
## Linear Model R^2 on Test set: : 0.4641839
```

Hide

```
# Calculate RMSE for each region
rmse_0_10 <- calculate_rmse(
  actual = error_data$Actual[error_data$Actual <= 10],
  predicted = error_data$Predicted[error_data$Actual <= 10]
)

rmse_10_15 <- calculate_rmse(
  actual = error_data$Actual[error_data$Actual > 10 & error_data$Actual <= 15],
  predicted = error_data$Predicted[error_data$Actual > 10 & error_data$Actual <= 15]
)

rmse_15_plus <- calculate_rmse(
  actual = error_data$Actual[error_data$Actual > 15],
  predicted = error_data$Predicted[error_data$Actual > 15]
)

# Plot actual vs predicted values with vertical lines and RMSE annotations
ggplot(error_data, aes(x = Actual, y = Predicted)) +
  geom_point(color = "blue", alpha = 0.6) +
  geom_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed") +  # Line of equality
  geom_vline(xintercept = 10, color = "green", linetype = "dotted", size = 1) +  # Vertical line at x = 10
  geom_vline(xintercept = 15, color = "green", linetype = "dotted", size = 1) +  # Vertical line at x = 15
  annotate("text", x = 3.5, y = max(error_data$Predicted),
           label = paste("RMSE (0-10):", round(rmse_0_10, 2)),
           color = "darkgreen", size = 3, hjust = 0) +
  annotate("text", x = 10.5, y = max(error_data$Predicted),
           label = paste("RMSE (10-15):", round(rmse_10_15, 2)),
           color = "darkgreen", size = 3, hjust = 0) +
  annotate("text", x = 18.5, y = max(error_data$Predicted),
           label = paste("RMSE (15+):", round(rmse_15_plus, 2)),
           color = "darkgreen", size = 3, hjust = 0) +
  labs(
    title = "Linear Model - Actual vs Predicted Values on Test Set",
    subtitle = "Regions: (0-10), (10-15), (15+)",
    x = "Actual Values",
    y = "Predicted Values"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),
    plot.subtitle = element_text(size = 14),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```



In the graph, the red dashed line refers to the line where predicted values == actual values. If points are close to the line, they are good predctions, else they are not.

There is a clear positive linear trend in the scatter plot, indicating that the model capture the relationship between actual and predicted values well.

For lower actual values (0–10), the model over-predicts, as points deviate upwards from the red line. The rmse for this region is 2.49 $\mu g/m^3$ which means on average, the model is 2.49 $\mu g/m^3$ off the actual pollution score.

Values between 10-15 have strong predictions as most of the points are concentrated near the red dashed line on both under-predictions and over-predictions. The RMSE of this region is 1.17 $\mu g/m^3$ which means on average, the model is 1.17 $\mu g/m^3$ off the actual predicted score

For higher actual values (>15), there are fewer data points, and the model predictions under-predict. The RMSE of this region is 5.75 $\mu g/m^3$ which means on average, the model is 5.75 $\mu g/m^3$ off the actual predicted score This could be due to the lack of training data for high pollution scores as seen in the EDA section.

The model had an overall rmse of 1.95 and $R^2 = 0.46$ which means on average the model is 1.95 $\mu g/m^3$ off the actual value of air pollution. The model also explains 46% of the variablity in air pollution.

Overall, this model is a strong indicator for pollution values for the data set as within the region og 10-15 where most of the data is concentrated, the model regresses with high accuracy. However, for outlier at the both tails of the distribution, the model performs poorly.

Random Forest Model

Data Preprocessing

We first preprocess the data to work with the regression model. We mutate the data in order to fit the formatting for the regression, and split them into train and test set for the random forest model.

Hide

```
pm_RF<- pm |>
  mutate(across(c(id, fips, zcta), as.factor))

# Splitting the data
set.seed(1234)
pm_split <- initial_split(data = pm_RF, prop = 3/4)
pm_split
```

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

Hide

```
train_pm <- training(pm_split)
test_pm <- testing(pm_split)

# Recipe
RF_rec <- recipe(train_pm) |>
    update_role(everything(), new_role = "predictor")|>
    update_role(value, new_role = "outcome")|>
    update_role(id, new_role = "id variable") |>
    update_role("fips", new_role = "county id") |>
    step_novel("state") |>
    step_string2factor("state", "county", "city") |>
    step_rm("city") |>
    step_rm("county") |>
    step_rm("zcta") |>
    step_corr(all_numeric())|>
    step_nzv(all_numeric())


RF_rec
```

Hide

```
# Prepping the data
prepped_rec <- prep(RF_rec,
                    verbose = TRUE,
                    retain = TRUE)
```

```
## oper 1 step novel [training]
## oper 2 step string2factor [training]
## oper 3 step rm [training]
## oper 4 step rm [training]
## oper 5 step rm [training]
## oper 6 step corr [training]
## oper 7 step nzv [training]
## The retained training set is ~ 0.29 Mb  in memory.
```

Hide

```
baked_train <- bake(prepped_rec, new_data = NULL)
baked_test_pm <- bake(prepped_rec, new_data = test_pm)
```

Setup

We then set the model workflow for training and set a random seed for reproduciblity.

Hide

```
# 4 fold cross validation
set.seed(4321)
vfold_pm <- vfold_cv(data = train_pm, v = 4)

# Model Specification
RF_PM_model <- rand_forest(mtry = 10, min_n = 3) |>
  set_engine("randomForest") |>
  set_mode("regression")

RF_PM_model
```

```
## Random Forest Model Specification (regression)
##
## Main Arguments:
##   mtry = 10
##   min_n = 3
##
## Computational engine: randomForest
```

Hide

```
#Workflow
RF_wflow <- workflow() |>
  add_recipe(RF_rec) |>
  add_model(RF_PM_model)

RF_wflow
```

```
## ══ Workflow ═════════════════════════════════════════════════════════════════════
## Preprocessor: Recipe
## Model: rand_forest()
##
## ── Preprocessor ─────────────────────────────────────────────────────────────────
## 7 Recipe Steps
##
## • step_novel()
## • step_string2factor()
## • step_rm()
## • step_rm()
## • step_rm()
## • step_corr()
## • step_nzv()
##
## ── Model ────────────────────────────────────────────────────────────────────────
## Random Forest Model Specification (regression)
##
## Main Arguments:
##   mtry = 10
##   min_n = 3
##
## Computational engine: randomForest
```

Hide

```
#Fit the data
RF_wflow_fit <- fit(RF_wflow, data = train_pm)

RF_wflow_fit
```

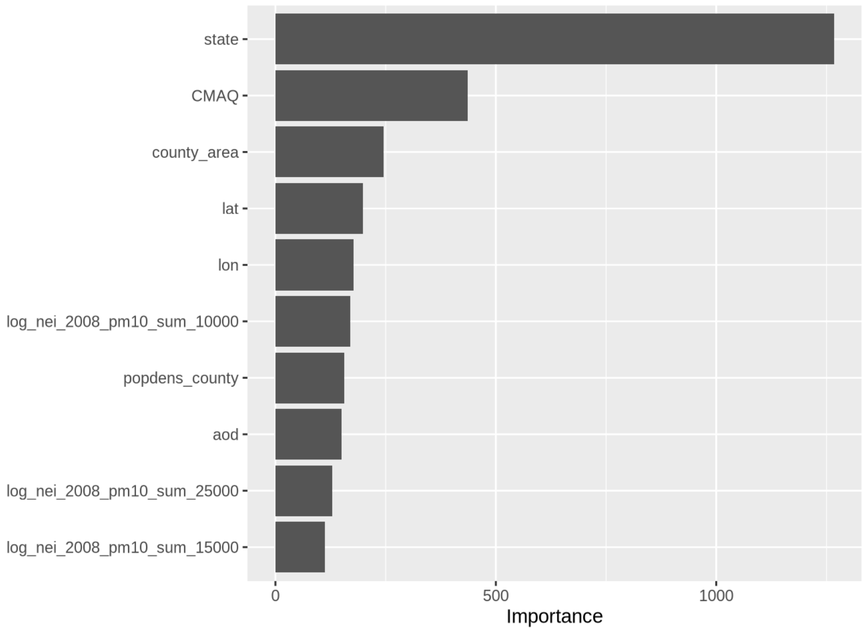
```
## ══ Workflow [trained] ══════════════════════════════════════════════════════════
## Preprocessor: Recipe
## Model: rand_forest()
##
## ── Preprocessor ────────────────────────────────────────────────────────────────
## 7 Recipe Steps
##
## • step_novel()
## • step_string2factor()
## • step_rm()
## • step_rm()
## • step_rm()
## • step_corr()
## • step_nzv()
##
## ── Model ───────────────────────────────────────────────────────────────────────
##
## Call:
##  randomForest(x = maybe_data_frame(x), y = y, mtry = min_cols(~10,      x), nodesize = min_rows(~3, x))
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 10
##
##           Mean of squared residuals: 2.484254
##                     % Var explained: 61.88
```

Feature importance

Based on the data, here is the breakdown of feature importance

Hide

```
RF_wflow_fit |>
  extract_fit_parsnip() |>
  vip::vip(num_features = 10)
```



State and CMAQ are the two highest importance variables which fits our eda as CMAQ has the highest correlation with our outcome (0.5) and state capture the geo-spatial nature of our data.

Model Performance

We then train the model as a test to check how it performes on the train set.

Hide

```
set.seed(456)
resample_RF_fit <- tune::fit_resamples(RF_wflow, vfold_pm)
collect_metrics(resample_RF_fit)
```

```
## # A tibble: 2 × 6
##   .metric .estimator  mean     n std_err .config
##   <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 rmse    standard   1.63      4  0.0772 Preprocessor1_Model1
## 2 rsq     standard   0.614     4  0.0354 Preprocessor1_Model1
```

With the current workflow, our model reaches an $R^2 = 0.61$ which means 61% of scores variability is explanied by our model. This paired with the $rmse = 1.62$ tells us that on average, our model mispredicts by 1.62 $\mu g/m^3$ (max 23.16 $\mu g/m^3$).

This can be improved upon.

Tune Model

In order to improve upon the predictions, we fine tune the model by the following pipeline:

Hide

```
tune_RF_model <- rand_forest(mtry = tune(), min_n = tune()) |>
  set_engine("randomForest") |>
  set_mode("regression")

tune_RF_model
```

```
## Random Forest Model Specification (regression)
##
## Main Arguments:
##   mtry = tune()
##   min_n = tune()
##
## Computational engine: randomForest
```

Hide

```
RF_tune_wflow <- workflows::workflow() |>
  workflows::add_recipe(RF_rec) |>
  workflows::add_model(tune_RF_model)

RF_tune_wflow
```

```
## ══ Workflow ════════════════════════════════════════════════════════════════════
## Preprocessor: Recipe
## Model: rand_forest()
##
## ── Preprocessor ────────────────────────────────────────────────────────────────
## 7 Recipe Steps
##
## • step_novel()
## • step_string2factor()
## • step_rm()
## • step_rm()
## • step_rm()
## • step_corr()
## • step_nzv()
##
## ── Model ───────────────────────────────────────────────────────────────────────
## Random Forest Model Specification (regression)
##
## Main Arguments:
##   mtry = tune()
##   min_n = tune()
##
## Computational engine: randomForest
```

Hide

```
n_cores <- parallel::detectCores()
n_cores
```

```
## [1] 256
```

Hide

```
doParallel::registerDoParallel(cores = n_cores)

set.seed(123)
tune_RF_results <- tune_grid(object = RF_tune_wflow, resamples = vfold_pm, grid = 20)
tune_RF_results
```

```
## # Tuning results
## # 4-fold cross-validation
## # A tibble: 4 × 4
##   splits            id    .metrics          .notes
##   <list>            <chr> <list>            <list>
## 1 <split [492/165]> Fold1 <tibble [40 × 6]> <tibble [0 × 3]>
## 2 <split [493/164]> Fold2 <tibble [40 × 6]> <tibble [0 × 3]>
## 3 <split [493/164]> Fold3 <tibble [40 × 6]> <tibble [1 × 3]>
## 4 <split [493/164]> Fold4 <tibble [40 × 6]> <tibble [0 × 3]>
##
## There were issues with some computations:
##
##   - Warning(s) x1: 35 columns were requested but there were 34 predictors in the dat...
##
## Run `show_notes(.Last.tune.result)` for more information.
```

We then extract the best model on root mean squred error and evaluate it on the test set:

Hide

```
tune_RF_results |>
  collect_metrics()
```

```
## # A tibble: 40 × 8
##     mtry min_n .metric .estimator  mean     n std_err .config
##    <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
##  1    12    33 rmse    standard   1.69      4  0.0777 Preprocessor1_Model01
##  2    12    33 rsq     standard   0.581     4  0.0374 Preprocessor1_Model01
##  3    27    35 rmse    standard   1.69      4  0.0817 Preprocessor1_Model02
##  4    27    35 rsq     standard   0.565     4  0.0272 Preprocessor1_Model02
##  5    22    40 rmse    standard   1.69      4  0.0796 Preprocessor1_Model03
##  6    22    40 rsq     standard   0.573     4  0.0296 Preprocessor1_Model03
##  7     1    27 rmse    standard   2.01      4  0.0797 Preprocessor1_Model04
##  8     1    27 rsq     standard   0.459     4  0.0568 Preprocessor1_Model04
##  9     6    32 rmse    standard   1.74      4  0.0838 Preprocessor1_Model05
## 10     6    32 rsq     standard   0.575     4  0.0461 Preprocessor1_Model05
## # ℹ 30 more rows
```

Hide

```
show_best(tune_RF_results, metric = "rmse", n = 1)
```

```
## # A tibble: 1 × 8
##    mtry min_n .metric .estimator  mean     n std_err .config
##   <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1    30     4 rmse    standard    1.63     4  0.0820 Preprocessor1_Model16
```

Hide

```
tuned_RF_values <- select_best(tune_RF_results, metric = "rmse")
tuned_RF_values
```

```
## # A tibble: 1 × 3
##    mtry min_n .config
##   <int> <int> <chr>
## 1    30     4 Preprocessor1_Model16
```

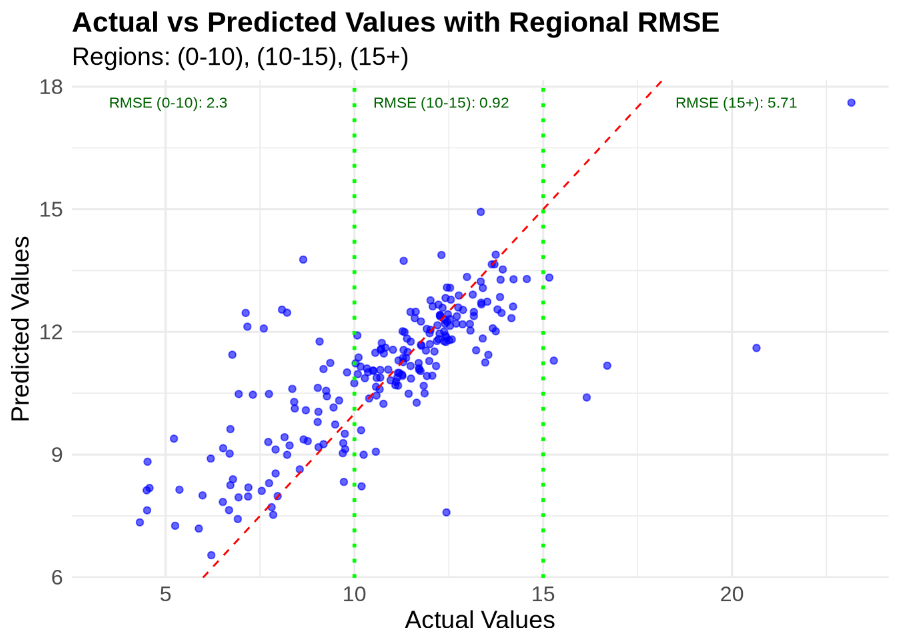
Hide

```
set.seed(456)
# specify best combination from tune in workflow
RF_tuned_wflow <-RF_tune_wflow |>
  tune::finalize_workflow(tuned_RF_values)

# fit model with those parameters on train AND test
overallfit <- RF_wflow |>
  tune::last_fit(pm_split)

collect_metrics(overallfit)
```

```
## # A tibble: 2 × 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 rmse    standard       1.78  Preprocessor1_Model1
## 2 rsq     standard       0.581 Preprocessor1_Model1
```

Hide

```
test_predictions <- collect_predictions(overallfit)


# Create a data frame for plotting
error_data <- data.frame(
  Actual = baked_test_pm$value,
  Predicted = test_predictions$.pred
)

# Function to calculate RMSE for a specific range
calculate_rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}

# Calculate RMSE for each region
rmse_0_10 <- calculate_rmse(
  actual = error_data$Actual[error_data$Actual <= 10],
  predicted = error_data$Predicted[error_data$Actual <= 10]
)

rmse_10_15 <- calculate_rmse(
  actual = error_data$Actual[error_data$Actual > 10 & error_data$Actual <= 15],
  predicted = error_data$Predicted[error_data$Actual > 10 & error_data$Actual <= 15]
)

rmse_15_plus <- calculate_rmse(
  actual = error_data$Actual[error_data$Actual > 15],
  predicted = error_data$Predicted[error_data$Actual > 15]
)

# Plot actual vs predicted values with vertical lines and RMSE annotations
ggplot(error_data, aes(x = Actual, y = Predicted)) +
  geom_point(color = "blue", alpha = 0.6) +
  geom_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed") +  # Line of equality
  geom_vline(xintercept = 10, color = "green", linetype = "dotted", size = 1) +  # Vertical line at x = 10
  geom_vline(xintercept = 15, color = "green", linetype = "dotted", size = 1) +  # Vertical line at x = 15
  annotate("text", x = 3.5, y = max(error_data$Predicted),
           label = paste("RMSE (0-10):", round(rmse_0_10, 2)),
           color = "darkgreen", size = 3, hjust = 0) +
  annotate("text", x = 10.5, y = max(error_data$Predicted),
           label = paste("RMSE (10-15):", round(rmse_10_15, 2)),
           color = "darkgreen", size = 3, hjust = 0) +
  annotate("text", x = 18.5, y = max(error_data$Predicted),
           label = paste("RMSE (15+):", round(rmse_15_plus, 2)),
           color = "darkgreen", size = 3, hjust = 0) +
  labs(
    title = "Actual vs Predicted Values with Regional RMSE",
    subtitle = "Regions: (0-10), (10-15), (15+)",
    x = "Actual Values",
    y = "Predicted Values"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),
    plot.subtitle = element_text(size = 14),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```



In the graph, the red dashed line refers to the line where predicted values == actual values. If points are close to the line, they are good predctions, else they are not.

There is a clear positive linear trend in the scatter plot, indicating that the model capture the relationship between actual and predicted values well.

For lower actual values (0–10), the model over-predicts, as points deviate upwards from the red line. The rmse for this region is 2.3 $\mu g/m^3$ which means on average, the model is 2.3 $\mu g/m^3$ off the actual pollution score.

Values between 10-15 have strong predictions as most of the points are concentrated near the red dashed line on both under-predictions and over-predictions. The RMSE of this region is 0.92 $\mu g/m^3$ which means on average, the model is 0.92 $\mu g/m^3$ off the actual predicted score

For higher actual values (>15), there are fewer data points, and the model predictions under-predict. The RMSE of this region is 5.71 $\mu g/m^3$ which means on average, the model is 5.71 $\mu g/m^3$ off the actual predicted score This could be due to the lack of training data for high pollution scores as seen in the EDA section.

The model had an overall rmse of 1.78 and $R^2 = 0.58$ which means on average the model is 1.78 $\mu g/m^3$ off the actual value of air pollution. The model also explains 58% of the variability in air pollution.

Overall, this model is a strong indicator for pollution values for the data set as within the region og 10-15 where most of the data is concentrated, the model regresses with high accuracy. However, for outliers at the tails of the distribution, the model performs poorly.

Model Comparision on Test Set

Hide

```
# Model comparison plot
model_comparison <- data.frame(
  Model = c("Linear Regression", "Random Forest"),
  RMSE_TEST = c(1.95, 1.78),
  RMSE_0_10 = c(2.49, 2.30),
  RMSE_10_15 = c(1.17, 0.92),
  RMSE_15_plus = c(5.75, 5.71),
  RSQ = c(0.46, 0.58)
)

model_comparison
```

```
##               Model RMSE_TEST RMSE_0_10 RMSE_10_15 RMSE_15_plus  RSQ
## 1 Linear Regression      1.95      2.49       1.17         5.75 0.46
## 2     Random Forest      1.78      2.30       0.92         5.71 0.58
```

For every metric on the test set, the Random Forest Model performs better than the linear regression model. The Random RMSE on the test is 0.17 $\mu g/m^3$ less than the RMSE of Linear regression and explains 12% more variability.

Therefore, the Random Forest Model is the better predictor that Multivariate linear regression.

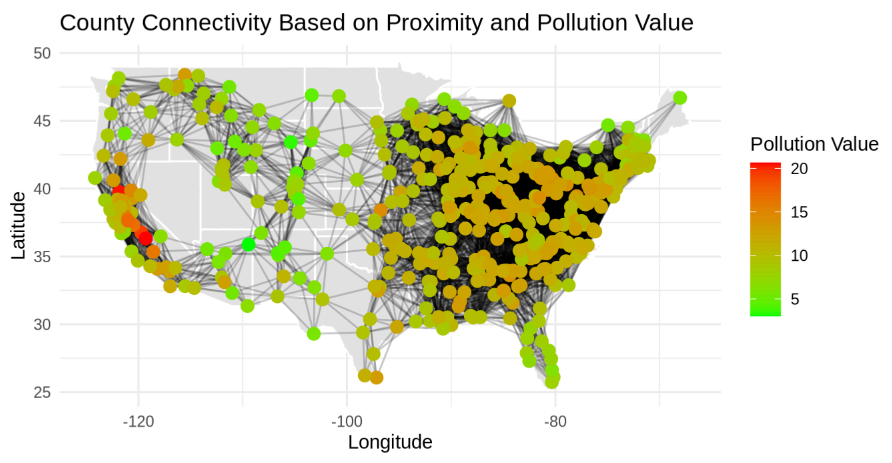
Extension: How do neigbouring air pollution values relate?

Visualization

One factor which is not captured by our models is neighboring counties. Air pollution scores are spatial information, so neighboring air pollution values influence each other. Therefore to capture this data, we created graph where each node is a data point based on latitude and longitude information. The color reflects the level of pollution.

Hide

```r
# Aggregate data by county
county_data <- data_long %>%
  group_by(county) %>%
  summarize(
    lat = mean(lat, na.rm = TRUE),
    lon = mean(lon, na.rm = TRUE),
    value = mean(value, na.rm = TRUE)
  ) %>%
  ungroup()

# Create a distance matrix for all counties based on lat/lon
dist_matrix <- distm(county_data[, c("lon", "lat")], fun = distHaversine)

# Define a distance threshold for nearby connections
threshold <- 500000

# Create an adjacency matrix where distances below the threshold are connected
adj_matrix <- dist_matrix < threshold
diag(adj_matrix) <- 0

# Create a graph object
graph <- graph_from_adjacency_matrix(adj_matrix, mode = "undirected")

# Add node attributes (pollution score)
V(graph)$value <- county_data$value

# Add colors to nodes based on value (normalized using quantiles)
breaks <- quantile(county_data$value, probs = seq(0, 1, 0.25), na.rm = TRUE)
labels <- c("blue", "green", "yellow", "red")

V(graph)$color <- cut(county_data$value,
                      breaks = breaks,
                      labels = labels,
                      include.lowest = TRUE)

edges <- as_data_frame(graph, what = "edges")
edges <- edges %>%
  mutate(
    from_lat = county_data$lat[edges$from],
    from_lon = county_data$lon[edges$from],
    to_lat = county_data$lat[edges$to],
    to_lon = county_data$lon[edges$to]
  )

# Get US map data
us_map <- map_data("state")

# Plot the US map with nodes and edges
ggplot() +
  geom_polygon(data = us_map, aes(x = long, y = lat, group = group),
               fill = "gray90", color = "white") +
  geom_segment(data = edges, aes(x = from_lon, y = from_lat, xend = to_lon, yend = to_lat),
               color = "black", alpha = 0.2) +
  geom_point(data = county_data, aes(x = lon, y = lat, color = value), size = 3) +
  scale_color_gradient(low = "green", high = "red", name = "Pollution Value") +
  coord_fixed(1.3) +
  theme_minimal() +
  labs(title = "County Connectivity Based on Proximity and Pollution Value",
       x = "Longitude", y = "Latitude")
```



If counties were close to each other (threshold of 500000 for distHaversine), they were connected. To train a model to test a relationship, for each county, we had a one-hop average pollution score. This calculated by taking all its connected neighbors' air pollution score and averaging it. We then trained a linear model for value ~ neighborhood average.

One-hop Neigbors Model

Hide

```
# Create an adjacency matrix where distances below the threshold are connected
adj_matrix <- dist_matrix < threshold
diag(adj_matrix) <- 0 # Remove self-loops

# Create a graph object
graph <- graph_from_adjacency_matrix(adj_matrix, mode = "undirected")

# Add node attributes (pollution score)
V(graph)$value <- county_data$value

# Calculate the average pollution score of each county's neighbors
neighborhood_average <- sapply(ego(graph, order = 1), function(neighbors) {
  # Exclude the current node from its neighbors
  neighbor_values <- V(graph)$value[neighbors]
  if (length(neighbor_values) > 1) {
    mean(neighbor_values[-1], na.rm = TRUE) # Exclude self from calculation
  } else {
    NA
  }
})

# Add neighborhood averages as a new column to the county_data dataframe
county_data$neighborhood_average <- neighborhood_average

# Create a linear model correlating pollution score with neighborhood average
linear_model <- lm(value ~ neighborhood_average, data = county_data)

r_squared <- summary(linear_model)$r.squared

# logging
cat("Linear Model Equation:\n")
```

```
## Linear Model Equation:
```

Hide

```
cat(sprintf("value = %.4f + %.4f * neighborhood_average\n",
            coef(linear_model)[1], coef(linear_model)[2]))
```

```
## value = -0.9058 + 1.0737 * neighborhood_average
```

Hide

```
cat('R_squared: ', r_squared)
```

```
## R_squared:  0.4514695
```

For each 1 unit increase in the neigbhorhood average pollution, the pollution of a county increases by 1.07 $\mu g/m^3$ (max 23.16 $\mu g/m^3$).

The $R^= 0.451$ which mean 45.1% of pollution's variance is explained by neighboring pollution values.

Therefore, nearby spatial pollution levels are a significant predictor of air pollution for a region with positive relationship.

Results & Discussion

Air Pollution Prediction

Multivariate Linear Regression

To predict annual air pollution, the linear model achieved an overall RMSE of **1.95** and an $R^2 = 0.46$ on the test set (25% if the data), indicating that the model is, on average, **1.95** $\mu g/m^3$ off the actual air pollution values. Additionally, the model explains **46% of the variability** in air pollution levels. Breaking this down further:

1. **For values between 0–10:** The model tends to over-predict, with an average error of 2.49 $\mu g/m^3$.
2. **For values between 10–15:** The model both over-predicts and underpredicts, with an average error of 1.17 $\mu g/m^3$.
3. **For values above 15:** The model consistently under-predicts, with an average error of 5.75 $\mu g/m^3$.

These results can be attributed to the distribution of the data set, which primarily contains values between 9 and 15, with significant outliers above 15. Since the Multivariate Linear Regression model learns patterns based on the underlying distribution of the data, it struggles to accurately predict outliers, as reflected in our empirical results.

NOTE: values range from 3.02 $\mu g/m^3$ to 23.16 $\mu g/m^3$

Random Forest Regression

To predict annual air pollution, we trained and fine-tuned a random forest model. The model achieved an overall RMSE of **1.78** and an $R^2 = 0.58$, indicating that the model is, on average, **1.78** $\mu g/m^3$ the actual air pollution values. Additionally, the model explains **58% of the variability** in air pollution levels. Breaking this down further:

1. **For values between 0–10:** The model tends to overpredict, with an average error of 2.3 $\mu g/m^3$.
2. **For values between 10–15:** The model both overpredicts and underpredicts, with an average error of 0.92 $\mu g/m^3$.
3. **For values above 15:** The model consistently underpredicts, with an average error of 5.75 $\mu g/m^3$.

NOTE: values range from 3.02 $\mu g/m^3$ to 23.16 $\mu g/m^3$

These results can be attributed to the distribution of the data set, which primarily contains values between 9 and 15, with significant outliers above 15. Since the random forest model learns patterns based on the underlying distribution of the data, it struggles to accurately predict outliers, as reflected in our empirical results.

Geo-spatial Relationship

When incorporating geo-spatial relationships using a one-hop neighbors model, the relationship is captured by the equation:

$$\text{Pollution} = -0.9058 + 1.0737 \times \text{neighborhood\_average}$$

This model achieves an $R^2 = 0.451$, meaning it explains **45.1% of the variance** in pollution values. The slope of **1.0737** indicates that if neighboring counties experience high air pollution, the county under investigation is also likely to have high pollution levels. This highlights the importance of **spatial information** as a critical factor in understanding air pollution. The $R^2 = 0.451$ reinforces the significance of this spatial relationship, demonstrating that neighboring pollution values play a substantial role in predicting pollution levels.

Limitations

Since we are modeling a distribution using a random forest model, it is crucial to have sufficient data to effectively learn the full range of the distribution. However, due to a lack of data representing outliers, the model struggles to accurately predict these extreme values. To address this, additional data on outliers is needed to enhance the model's performance and accuracy.

Additionally, our random forest model does not effectively capture the spatial relationships between neighboring regions. This is a limitation of the model, as it is not inherently designed to account for connectivity or geospatial interactions. A potential improvement could be the use of a Graph Neural Network (GNN), which is better equipped to capture nuanced information about spatial connectivity and relationships.

Lastly, the data set does not represent the entire United States uniformly. While there is ample data from coastal regions and areas with high population density, the central part of the country is underrepresented. To improve the model's ability to generalize across all regions, more data should be collected from these underrepresented areas.

Conclusion

This study demonstrates that annual average air pollution concentrations in the United States can be predicted with reasonable accuracy using a multivariate linear regression model and random forest model, achieving an error of **1.95** $\mu g/m^3$, **1.78** $\mu g/m^3$ and an $R^2 = 0.46$, $R^2 = 0.58$ respectively. As the Random Forest Model performs better than the linear regression model on our test set, it is the better model to use in they study.

Both models performs well for most of the data, particularly in the range of pollution values between 10 and 15, but they struggles with outliers and extreme values. This limitation arises from the dataset's skewed distribution, with the majority of data points falling within the 9–15 range and significant underrepresentation of values above 15. Addressing this issue requires collecting more data on outliers to enhance the model's ability to predict extreme pollution levels.

The geospatial analysis further highlights the importance of spatial relationships in understanding air pollution. By incorporating one-hop neighbor information, we demonstrated that pollution in neighboring counties strongly influences pollution levels in the county of interest. The resulting linear model, with an $R^2 = 0.451$ and a slope of **1.0737**, emphasizes the interconnected nature of air pollution. Counties with higher pollution levels tend to be surrounded by neighbors with similarly high pollution, reinforcing the critical role of spatial dynamics.

However, the study also reveals important limitations. The linear regression and random forest model are not inherently designed to capture spatial relationships, leading to missed opportunities to leverage geospatial connectivity more effectively. Future work could address this by employing more advanced models, such as **Graph Neural Networks (GNNs)**, which are better suited for capturing nuanced spatial interactions.

Finally, the data set itself poses challenges. While regions along the coasts and areas with high population density are well-represented, central parts of the United States remain under-sampled. This lack of uniform geographic representation limits the model's ability to generalize across the entire country. Expanding the data set to include more samples from underrepresented regions will be essential to building a truly robust and generalizeable model for predicting air pollution.

In conclusion, the random forest model provides valuable insights into air pollution prediction, predicting air pollution within error rates of 1.78 $\mu g/m^3$. However, it is important to address the data imbalances, improving spatial modeling capabilities, and enhancing geographic coverage are critical steps for advancing the accuracy and utility of these models in the future.

---

1. https://ww2.arb.ca.gov/resources/inhalable-particulate-matter-and-health (https://ww2.arb.ca.gov/resources/inhalable-particulate-matter-and-health)↩
2. https://www.epa.gov/pm-pollution/health-and-environmental-effects-particulate-matter-pm (https://www.epa.gov/pm-pollution/health-and-environmental-effects-particulate-matter-pm)↩
3. https://ww2.arb.ca.gov/resources/inhalable-particulate-matter-and-health (https://ww2.arb.ca.gov/resources/inhalable-particulate-matter-and-health)↩
4. https://www.epa.gov/pm-pollution/health-and-environmental-effects-particulate-matter-pm (https://www.epa.gov/pm-pollution/health-and-environmental-effects-particulate-matter-pm)↩
5. https://www.epa.gov/pm-pollution/particulate-matter-pm-basics (https://www.epa.gov/pm-pollution/particulate-matter-pm-basics)↩
6. https://www.clarity.io/blog/air-quality-measurements-series-wind-speed-and-direction (https://www.clarity.io/blog/air-quality-measurements-series-wind-speed-and-direction)↩
7. https://www.epa.gov/ (https://www.epa.gov/)↩
8. https://www.nasa.gov/ (https://www.nasa.gov/)↩
9. https://www.census.gov/about/what/census-at-a-glance.html (https://www.census.gov/about/what/census-at-a-glance.html)↩
10. https://www.cdc.gov/nchs/about/index.htm (https://www.cdc.gov/nchs/about/index.htm)↩
11. https://en.wikipedia.org/wiki/ZIP_Code_Tabulation_Area (https://en.wikipedia.org/wiki/ZIP_Code_Tabulation_Area)↩
12. https://www.cdc.gov/nchs/data/series/sr_02/sr02_166.pdf (https://www.cdc.gov/nchs/data/series/sr_02/sr02_166.pdf)↩
13. https://www.cdc.gov/nchs/data/series/sr_02/sr02_154.pdf (https://www.cdc.gov/nchs/data/series/sr_02/sr02_154.pdf)↩
14. https://www.cdc.gov/nchs/data/series/sr_02/sr02_154.pdf (https://www.cdc.gov/nchs/data/series/sr_02/sr02_154.pdf)↩
15. https://www.epa.gov/cmaq (https://www.epa.gov/cmaq)↩
16. https://www.epa.gov/greenvehicles/greenhouse-gas-emissions-typical-passenger-vehicle# (https://www.epa.gov/greenvehicles/greenhouse-gas-emissions-typical-passenger-vehicle#)↩