Mini project 1: air quality in U.S. cities

In a way, this project is simple: you are given some data on air quality in U.S. metropolitan areas over time together with several questions of interest, and your objective is to answer the questions.

However, unlike the homeworks and labs, there is no explicit instruction provided about *how* to answer the questions or where exactly to begin. Thus, you will need to discern for yourself how to manipulate and summarize the data in order to answer the questions of interest, and you will need to write your own codes from scratch to obtain results. It is recommended that you examine the data, consider the questions, and plan a rough approach before you begin doing any computations.

You have some latitude for creativity: **although there are accurate answers to each question** -- namely, those that are consistent with the data -- **there is no singularly correct answer**. Most students will perform similar operations and obtain similar answers, but there's no specific result that must be considered to answer the questions accurately. As a result, your approaches and answers may differ from those of your classmates. If you choose to discuss your work with others, you may even find that disagreements prove to be fertile learning opportunities.

The questions can be answered using computing skills taught in class so far and basic internet searches for domain background; for this project, you may wish to refer to HW1 and Lab1 for code examples and the EPA website on PM pollution for background. However, you are also encouraged to refer to external resources (package documentation, vignettes, stackexchange, internet searches, etc.) as needed -- this may be an especially good idea if you find yourself thinking, 'it would be really handy to do X, but I haven't seen that in class anywhere'.

The broader goal of these mini projects is to cultivate your problem-solving ability in an unstructured setting. Your work will be evaluated based on the following:

- choice of method(s) used to answer questions;
- clarity of presentation;
- code style and documentation.

Please write up your results separately from your codes; codes should be included at the end of the notebook.

Part I: Dataset

Merge the city information with the air quality data and tidy the dataset (see notes below). Write a brief description of the data.

In your description, answer the following questions:

- What is a CBSA (the geographic unit of measurement)?
- How many CBSA's are included in the data?
- In how many states and territories do the CBSA's reside? (*Hint:* `str.split()`)
- In which years were data values recorded?
- How many observations are recorded?
- How many variables are measured?
- Which variables are non-missing most of the time (*i.e.*, in at least 50% of instances)?
- What is PM 2.5 and why is it important?

Please write your description in narrative fashion; ***please do not list answers to the questions above one by one***. A few brief paragraphs should suffice; please limit your data description to three paragraphs or less.

Air quality data

CBSA stands for core-based statistical area. It's a geographic area that that consists of county(ies) around one urban area with at least 10,000 people. There are 351 CBSA's included in the CBSA dataset, there are two variables CBSA number and the geographic area. The CBSAs cover 86 unqiue states/territories, found using str.split() and other helper functions. Data was recorded from 2000 to 2019, a span of 20 years, and 1134 observations and 26 variables were recorded.

When examining the data for missing values, there are none across all the variables. Every observation (row) has values for all 26 variables. Upon looking at the EPA's website, PM 2.5 is an especially important particulate matter classification because of the risk it poses to our health. PM 2.5 are particulate matter whose diameter is less than 2.5 micrometers- they are fine, inhalable particles.

Part II: Descriptive analysis

Focus on the PM2.5 measurements that are non-missing most of the time. Answer each of the following questions in a brief paragraph or two. Your paragraph(s) should indicate both your answer and a description of how you obtained it; ***please do not include codes with your answers***.

Has PM 2.5 air pollution improved in the U.S. on the whole since 2000?

PM 2.5 air pollution has improved in the U.S. on the whole since 2000. By graphing the Weighted Annual Mean of each CBSA reporting PM 2.5 concentration throughout the years, we are able to see a slight downwards trend. I was able to graph the Weighted Annual Mean values by melting and faceting the data. There are some outliers in some of the years, but overall, the data seems to become centered around a smaller concentration value as the years go on.

Over time, has PM 2.5 pollution become more variable, less variable, or about equally variable from city to city in the U.S.?

Using the same PM 2.5 graph as used in the previous question, we're able to see that the concentration values for PM 2.5 get less variable as the years go on. Still faceting the data by year, we can see the Weighted Annual Mean values for each CBSA over time. The data seems to become centered around smaller concentration values as the years go on, as we saw previously. Additionally, the data becomes less spread out in the vertical axis, indicating less variability in concentration measurements as the years go on.

Which state has seen the greatest improvement in PM 2.5 pollution over time? Which city has seen the greatest improvement?

The territory that saw the greatest improvement in PM 2.5 pollution was TN-VA, while the individual state that saw the greatest improvement in PM 2.5 pollution was WV. Firstly, I defined improvement as the difference in the average of the Weighted Annual Mean across each unique state and territory. Firstly, I melted the data in order to create a Year column that would make the data easier to manipulate. I then filtered the data so that I only had 2000 and 2019 as the years of interest, before grouping by state and year and pivoting the means of the concentration value data (in order to get the unique states and territories as indices in the dataframe). I then created the difference column subtracting the concentration in 2019 and 2000. Looking at the min values of the new column, TN-VA had the greatest difference by magnitude (-10.20) for a single territory, while WV had the greatest difference by magnitude (-8.94) for a single state.

The city that saw the greatest improvement in PM 2.5 pollution over time was Portsmouth, OH. Here, improvement was defined in the same way- the difference in the Weighted Annual Means from 2019 and 2000 across each CBSA. Again, I melted the data so that the Years would be a single column. I then filtered the data so that I had 2000 and 2019 as years of interest, before pivoting the data to have the Core Based Statistical Area variable as the indicies (as each one was unique). I then created the improvement column by subtracting the concentrations from 2019 and 2000. Looking at the min value of the new column, Portsmouth, OH had the greatest difference by magnitude (-14.4).

Choose a location with some meaning to you (e.g. hometown, family lives there, took a vacation there, etc.). Was that location in compliance with EPA primary standards as of the most recent measurement?

The mean EPA standard value for PM 2.5 concentration is 12. The location I chose was San Francisco-Oakland-Hayward region which had a value of 7 in for most recent measurement (less than the EPA standard value), indicating that it was compliant with EPA primary standards.

Extra credit: Imputation

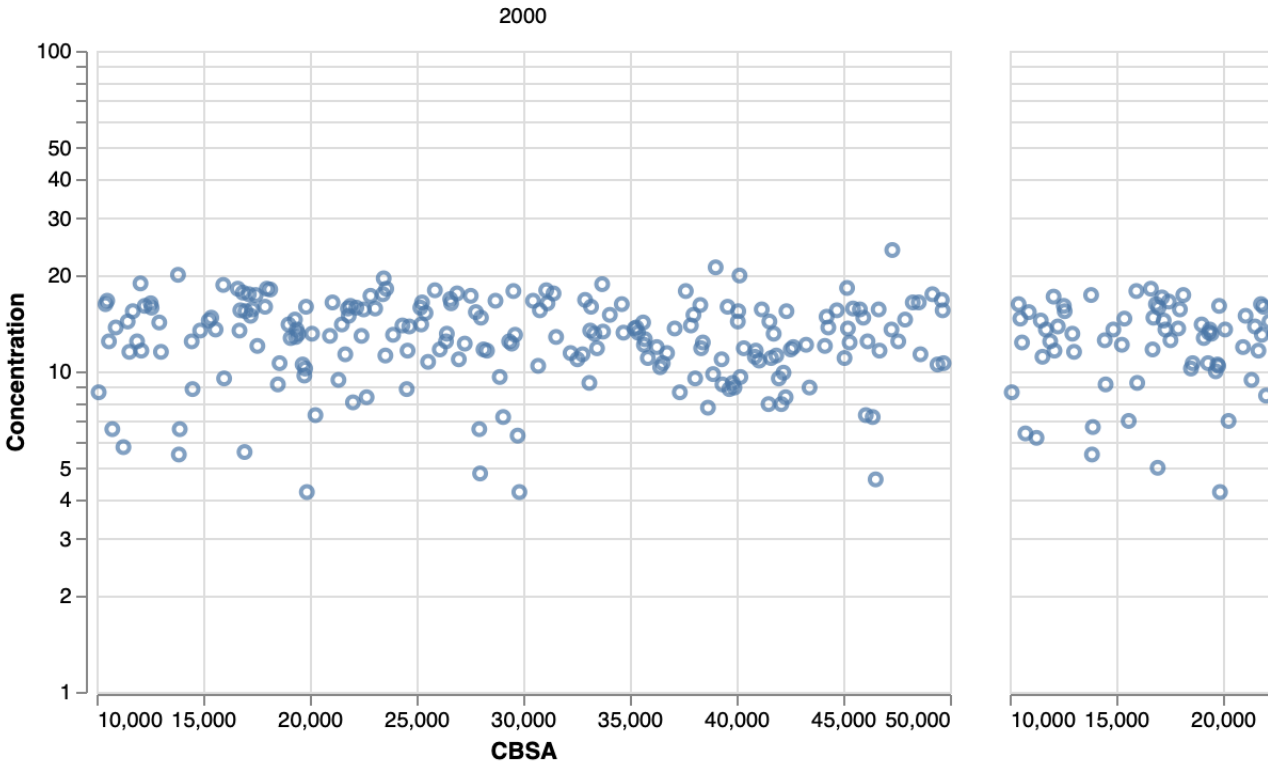
One strategy for filling in missing values ('imputation') is to use non-missing values to predict the missing ones; the success of this strategy depends in part on the strength of relationship between the variable(s) used as predictors of missing values.

Identify one other pollutant that might be a good candidate for imputation based on the PM 2.5 measurements and explain why you selected the variable you did. Can you envision any potential pitfalls to this technique?

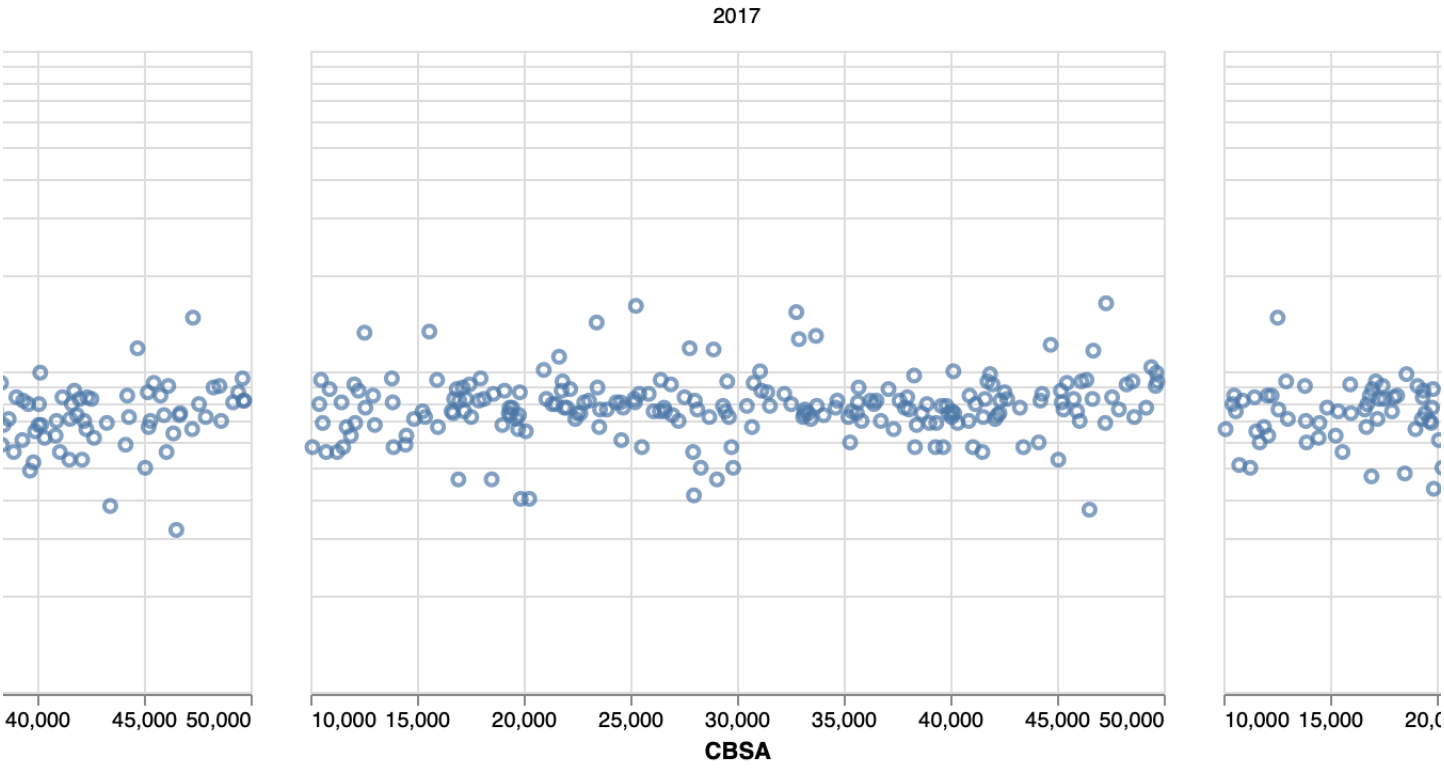
Codes

In [12]:
```
# packages
import numpy as np
import pandas as pd
import altair as alt

# raw data
air_raw = pd.read_csv('air-quality.csv')
cbsa_info = pd.read_csv('cbsa-info.csv')

## PART I
##########
data = pd.merge(air_raw, cbsa_info, how = 'left', on = 'CBSA')

def get_element(l, index): # helper function to split up city and state, to
    return l[index]

data['cities_covered'] = data['Core Based Statistical Area'].str.split(',').
data['state_terr_covered'] = data['Core Based Statistical Area'].str.split('
data['state_terr_covered'].unique().shape # figure out how many unique state

data.isna().sum(axis = 0) # figure out which variable has the most missing v

## PART II (continued in subsequent cells)
##########

# Has PM 2.5 air pollution improved in the U.S. on the whole since 2000?
# Scatter plots of PM2.5 over the years, using weighted average mean
yrs = range(2000, 2020)
yrs = list(map(str, range(2000,2020)))

plot_means = data[(data['Pollutant'] == 'PM2.5') & (data['Trend Statistic']
             columns = ['Pollutant', 'Number of Trends Sites', 'Core Based S
             ).melt(
             id_vars = ['CBSA'], value_vars = yrs, var_name = 'Year', value_
             ).reset_index()

alt.Chart(plot_means).mark_point().encode( # your turn here
    x = alt.X('CBSA', scale = alt.Scale(type = 'pow', zero = False)), # your
    y = alt.Y('Concentration', scale = alt.Scale(type = 'log')
)).facet(column = 'Year')
```

Out[12]:



In [13]:
```
alt.Chart(plot_means).mark_point().encode( # your turn here
    x = alt.X('CBSA', scale = alt.Scale(type = 'pow', zero = False)), # your
    y = alt.Y('Concentration', scale = alt.Scale(type = 'log')
)).facet(column = 'Year')
```



In [2]:
```
# city with the best improvement (2000-2019)

cities_diff = data[(data['Pollutant'] == 'PM2.5') & (data['Trend Statistic']
              columns = ['Pollutant', 'Number of Trends Sites']
              ).melt(
              id_vars = ['Core Based Statistical Area', 'CBSA', 'cities_cove
              ).reset_index()

cities_diff = cities_diff[(cities_diff.Year == '2000') | (cities_diff.Year =
cities_diff['improvement00-19'] = cities_diff['2019'] - cities_diff['2000']
cities_diff[cities_diff['improvement00-19'] == cities_diff['improvement00-19
```

Out[2]:

Year	2000	2019	improvement00-19
Core Based Statistical Area			
Portsmouth, OH	21.1	6.7	-14.4

In [3]:
```
# territory with the best improvement (2000-2019)

state_terr_diff = data[(data['Pollutant'] == 'PM2.5') & (data['Trend Statist
              columns = ['Pollutant', 'Number of Trends Sites']
              ).melt(
              id_vars = ['state_terr_covered', 'CBSA'], value_vars = yrs, va
              ).reset_index()

state_terr_diff = state_terr_diff[(state_terr_diff.Year == '2000') | (state_
    ['state_terr_covered', 'Year']
    ).mean().drop(columns = ['index', 'CBSA']).reset_index(
    ).pivot(index = 'state_terr_covered', columns = 'Year', values = 'Concen

state_terr_diff['improvement00-19'] = state_terr_diff['2019'] - state_terr_d
state_terr_diff[state_terr_diff['improvement00-19'] == state_terr_diff['impr
```

Out[3]:

Year	2000	2019	improvement00-19
state_terr_covered			
TN-VA	16.6	6.4	-10.2

In [4]:
```
# find state with the best improvement (2000-2019)

state_terr_diff.sort_values('improvement00-19', ascending = True)
```

Out[4]:

Year	2000	2019	improvement00-19
state_terr_covered			
TN-VA	16.600000	6.400000	-10.20
GA-AL	18.100000	8.800000	-9.30
WV-KY-OH	16.800000	7.700000	-9.10
TN-GA	17.600000	8.600000	-9.00
WV	16.360000	7.420000	-8.94
...	...	...	...
ND-MN	8.000000	6.500000	-1.50
ND	5.400000	3.900000	-1.50
HI	4.700000	3.550000	-1.15
NM	6.450000	5.450000	-1.00
OR	10.733333	11.333333	0.60

73 rows × 3 columns

In [5]:
```
# find PM 2.5 concentration in 2019 for SF-Oakland-Hayward

data = data[(data.cities_covered == 'San Francisco-Oakland-Hayward') & (data
      id_vars = ['cities_covered'], value_vars = yrs, var_name = 'Year', val
      )

data[data.Year == '2019']
```

Out[5]:

	cities_covered	Year	Concentration
19	San Francisco-Oakland-Hayward	2019	7.0

Notes on merging (keep at bottom of notebook)

To combine datasets based on shared information, you can use the `pd.merge(A, B, how = ..., on = SHARED_COLS)` function, which will match the rows of `A` and `B` based on the shared columns `SHARED_COLS`. If `how = 'left'`, then only rows in `A` will be retained in the output (so `B` will be merged *to* `A`); conversely, if `how = 'right'`, then only rows in `B` will be retained in the output (so `A` will be merged *to* `B`).

A simple example of the use of `pd.merge` is illustrated below:

In [6]:
```
# toy data frames
A = pd.DataFrame(
    {'shared_col': ['a', 'b', 'c'],
    'x1': [1, 2, 3],
    'x2': [4, 5, 6]}
)

B = pd.DataFrame(
    {'shared_col': ['a', 'b'],
    'y1': [7, 8]}
)
```

In [7]:
```
A
```

Out[7]:

	shared_col	x1	x2
0	a	1	4
1	b	2	5
2	c	3	6

In [8]:
```
B
```

Out[8]:

	shared_col	y1
0	a	7
1	b	8

Below, if `A` and `B` are merged retaining the rows in `A`, notice that a missing value is input because `B` has no row where the shared column (on which the merging is done) has value `c`. In other words, the third row of `A` has no match in `B`.

In [9]:
```
# left join
pd.merge(A, B, how = 'left', on = 'shared_col')
```

Out[9]:

	shared_col	x1	x2	y1
0	a	1	4	7.0
1	b	2	5	8.0
2	c	3	6	NaN

If the direction of merging is reversed, and the row structure of `B` is dominant, then the third row of `A` is dropped altogether because it has no match in `B`.

In [10]:
```
# right join
pd.merge(A, B, how = 'right', on = 'shared_col')
```

Out[10]:

	shared_col	x1	x2	y1
0	a	1	4	7
1	b	2	5	8

Submission Checklist

1. Save file to confirm all changes are on disk
2. Run *Kernel > Restart & Run All* to execute all code from top to bottom
3. Save file again to write any new output to disk
4. Select *File > Download as > HTML*.
5. Open in Google Chrome and print to PDF.
6. Submit to Gradescope