Air Quality Analysis In TamilNadu

In this part you will continue building our project.

* Perform the air quality analysis and create visualizations.
* Calculate average SO2, NO2, and RSPM/PM10 levels across different monitoring stations,cities, or areas. Identify pollution trends and areas with high pollution levels.
* Create visualizations using data visualization libraries (e.g., Matplotlib, Seaborn).

# To perform air quality analysis and create visualizations

**Data Collection:**

Obtain air quality data from relevant sources, such as government agencies or environmental organizations. You may find data for SO2, NO2, and RSPM/PM10 levels across different monitoring stations, cities, or areas.

**Data Preprocessing:**

Clean and prepare the data. This involves handling missing values, outliers, and ensuring data consistency.

**Calculate Averages:**

Calculate the average SO2, NO2, and RSPM/PM10 levels across the monitoring stations, cities, or areas you're interested in. You can use Python and libraries like Pandas for this.

**Identify Pollution Trends:**

Analyze the data to identify trends in pollution levels over time or across different locations. You can use statistical methods or time series analysis for this.

**Visualization:**

Create visualizations to convey your findings. You can use data visualization libraries like Matplotlib, Seaborn, or Plotly to generate charts and graphs. Some common visualizations include line charts, bar charts, heatmaps, and geographical maps.

**Interpretation:**

Interpret the visualizations to draw conclusions and insights about pollution trends and areas with high pollution levels.

**Report:**

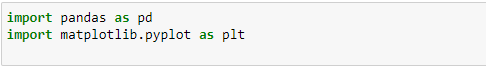
Document your analysis and findings in a report or presentation, including the visualizations. Make sure to communicate the results clearly to your audience.

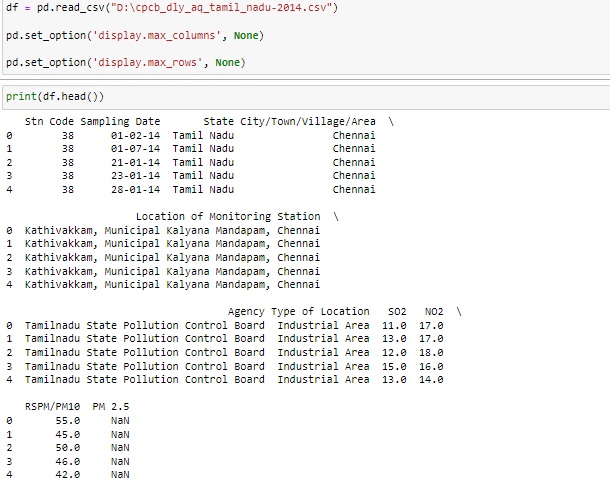
# Generating Histograms in Python With Matplotlib

* Matplotlib provides many out-of-the-box tools for quick and easy data visualization. For example, when analyzing a new data set, researchers are often interested in the distribution of values for a set of columns. One way to do so is through a histogram.
* Histograms are approximations to distributions generated through selecting values based on a set range and putting each set of values in a bin or bucket. Visualizing the distribution as a histogram is straightforward using Matplotlib.
* For our purposes, we will be working with the cpcb\_dly\_aq\_tamil\_nadu-2014 data set, which you can find here.
* To start, we need to import the Pandas library, which is a Python library used for data tasks such as statistical analysis and data wrangling:

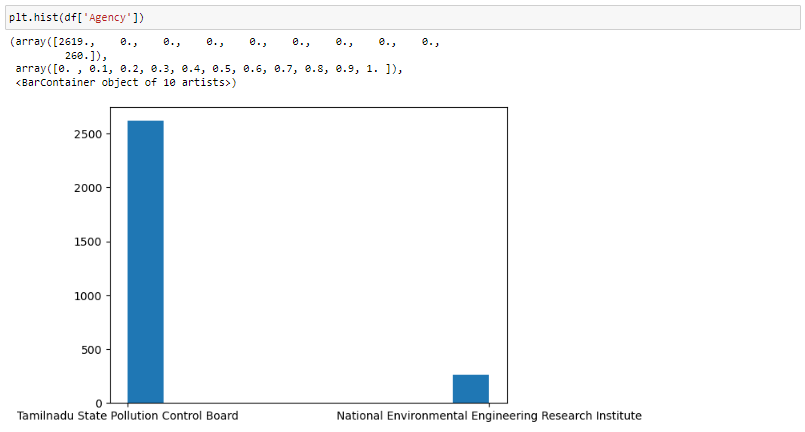
* import pandas as pd
* Next, we need to import the pyplot module from the Matplotlib library. It is custom to import it as plt:
* import matplotlib.pyplot as plt

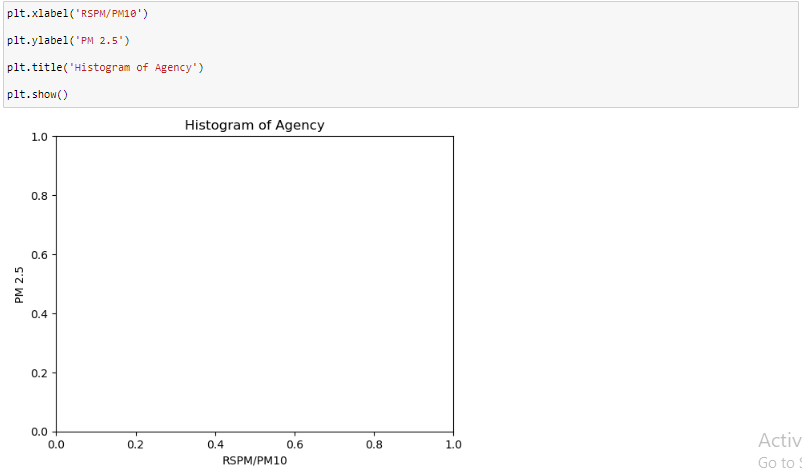
Now, let’s read our data into a Pandas dataframe. We will relax the limit on display columns and rows using the set\_option() method in Pandas:





We can generate a histogram for any of the numerical columns by calling the hist() method on the plt object and passing in the selected column in the data frame. Let’s do this for the Overall column, which corresponds to overall player rating:



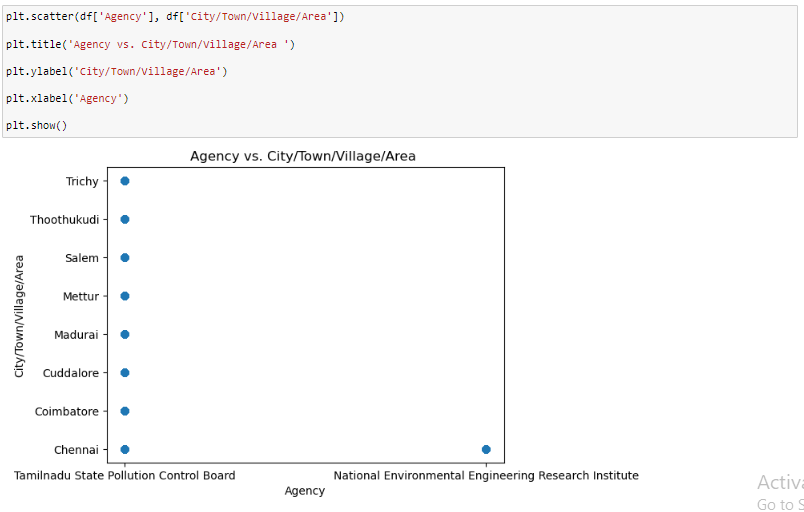
We can also label the x-axis, y-axis and title of the plot using the xlabel(), ylabel() and title() methods, respectively:

## Generating Scatter Plots in Python With Matplotlib

Scatter plots are a useful data visualization tool that helps with identifying variable dependence. For example, if we are interested in seeing if there is a positive relationship between wage and overall player rating, (i.e., if a Air Quality increases, does his rating also go up?) we can employ scatter plot visualization for insight.

Before we generate our scatter plot of wage versus overall rating, let’s convert the wage column from a string to a floating point numerical column.

To generate a scatter plot in Matplotlib, we simply use the scatter() method on the plt object. Let’s also label the axes and give our plot a title:



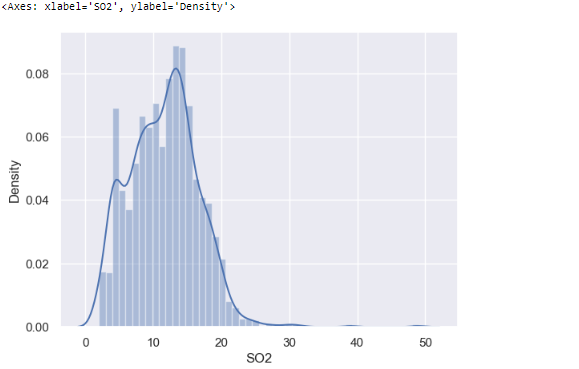
## Python Data Visualization With Seaborn

Seaborn is a library built on top of Matplotlib that enables more sophisticated visualization and aesthetic plot formatting. Once you’ve mastered Matplotlib, you may want to move up to Seaborn for more complex visualizations.

For example, simply using the Seaborn set() method can dramatically improve the appearance of your Matplotlib plots. Let’s take a look.

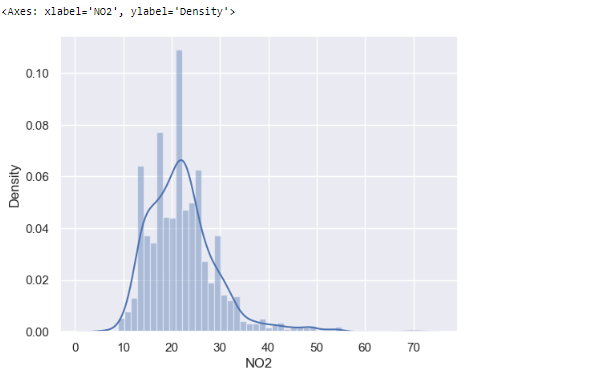
First, import Seaborn as sns and reformat all of the figures we generated. At the top of your script, write the following code and rerun:





And we can reuse the plt object for additional axis formatting and title setting:



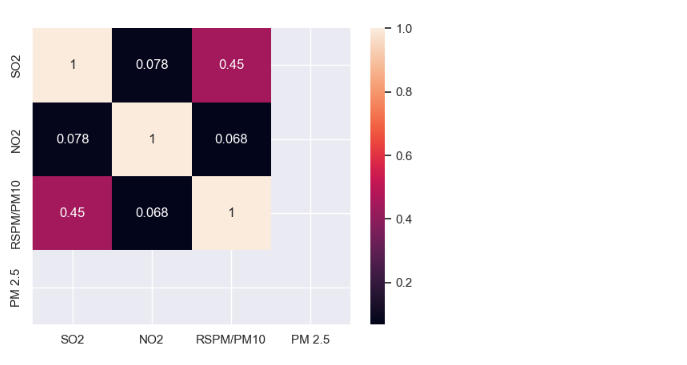


# Generating Heatmaps in Python With Seaborn

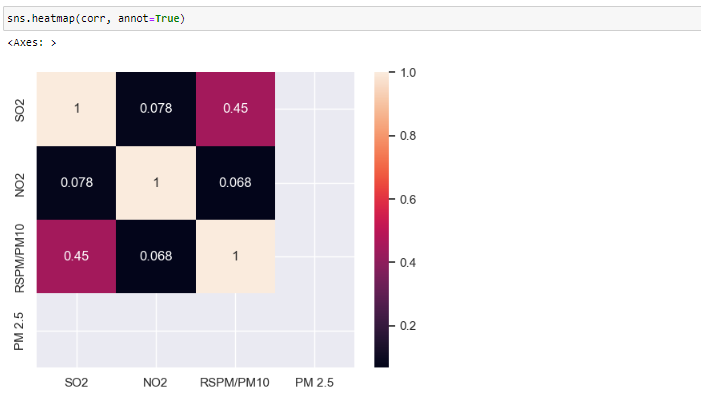
Seaborn is also known for making correlation heatmaps, which can be used to identify variable dependence. To generate one, first we need to calculate the correlation between a set of numerical columns. Let’s do this for age, overall, wage\_euro and skill moves:

A white screen with black text

Description automatically generated

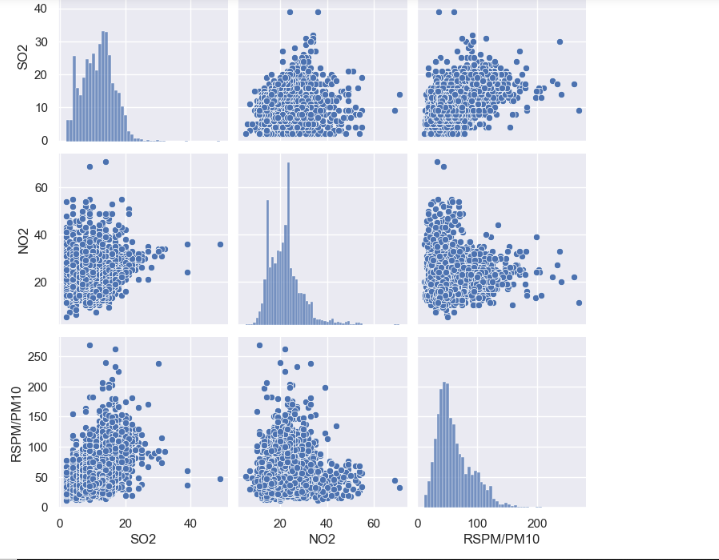


We can also set annot for annotate to true to see the correlation values:



# Generating Pairs Plots in Python With Seaborn

The last Seaborn tool I’ll discuss is the pairplot method. This allows you to generate a matrix of distributions and scatter plots for a set of numerical features. Let’s do this for age, overall and potential:



As you can see, this is a quick and easy way to visualize both the distribution in numerical values and relationships between variables through scatter plots.

# Visualizing in Python Using Matplotlib And Seaborn:

Overall, both Seaborn and Matplotlib are valuable tools for any data scientist. Matplotlib makes labeling, titling and formatting graphs simple, which is important for effective data communication. Further, it provides much of the basic tooling for visualizing data including histograms, scatter plots, pie charts and bar charts.

Seaborn is an important library to know because of its beautiful visuals and extensive statistical tooling. As you can see above, the plots generated in Seaborn, even if they communicate the same information, are much prettier than those generated in Matplotlib. Further, the tools provided by Seaborn allow for much more sophisticated analysis and visuals. Although I only discussed how to use Seaborn to generate heatmaps and pairwise plots, it can also be used to generate more complicated visuals like density maps for variables, line plots with confidence intervals, cluster maps and much more.

Matplotlib and Seaborn are two of the most widely used visualization libraries in Python. They both allow you to quickly perform data visualization for gaining statistical insights and telling a story with data. While there is significant overlap in the use cases for each of these libraries, having knowledge of both libraries can allow a data scientist to generate beautiful visuals that can tell an impactful story about the data being analyzed.

------------------------ -------------------- --------------------- -------------