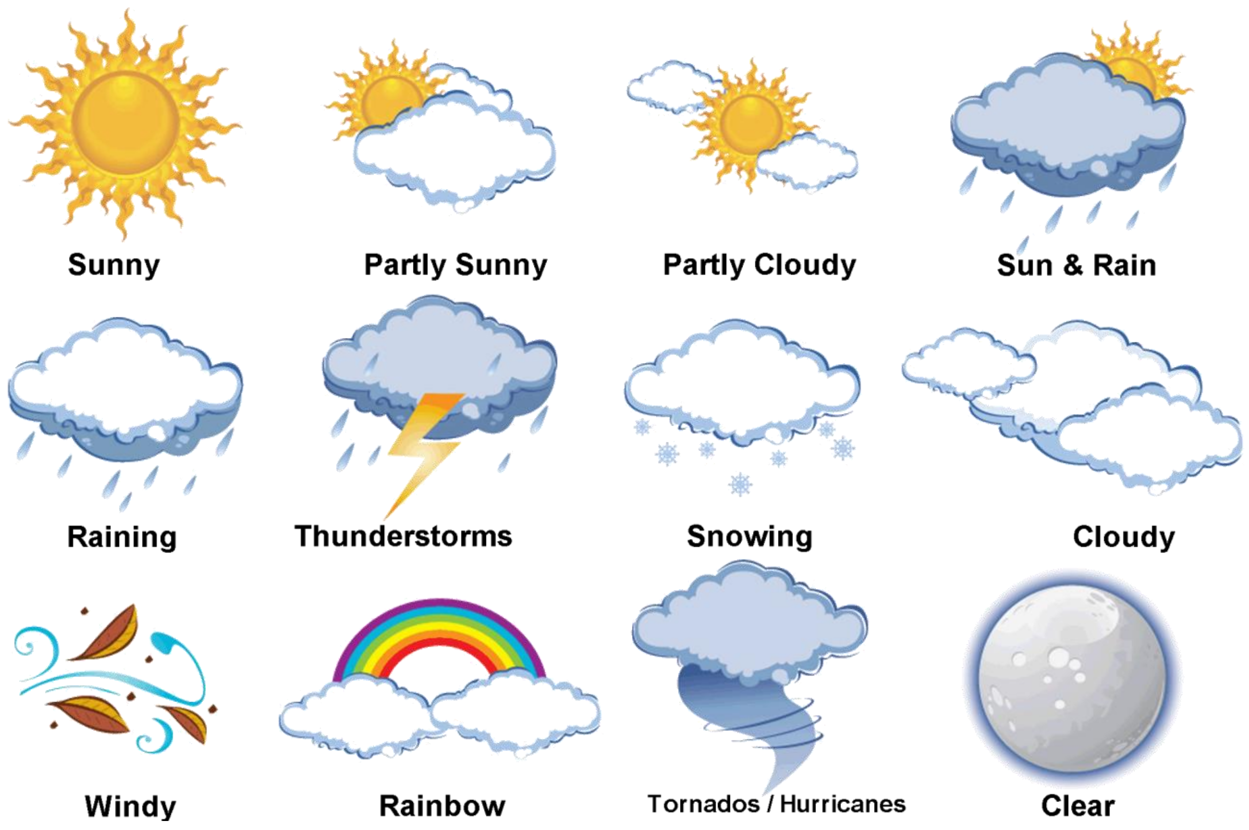
# Data Analyst Nanodegree

UDACITY



# Explore Weather Trends

By

Anuj Dutt

## **Extract Data from Database** (using SQL)

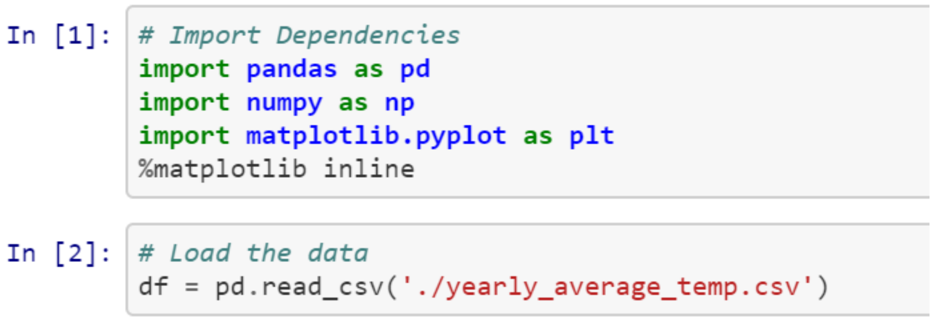
| Data Type | SQL Command for Extracting Data |
|---|---|
| To select the Country Name in “city_list”. | SELECT *<br>FROM city_list<br>WHERE country = 'India'; |
| To Alter the name for “avg_temp” in City_data to “city_avg_temp”. | ALTER TABLE city_data<br>RENAME COLUMN avg_temp to city_avg_temp; |
| To Alter the name for “avg_temp” in Global_data to “global_avg_temp”. | ALTER TABLE global_data<br>RENAME COLUMN avg_temp to global_avg_temp; |
| To join two tables “global_data” and “city_data” and extract three columns of relevant data i.e. “year”, “global_avg_temp” and “city_avg_temp” for a city in India. | SELECT gd.year, gd.global_avg_temp, cd.city_avg_temp<br>FROM global_data AS gd<br>INNER JOIN city_data as cd<br>ON gd.year = cd.year<br>WHERE city = 'Delhi'; |

**Approach to extract the desired data:**

- I used the above SQL commands to get the final CSV file containing the columns “year”, “global_average_temp” and “city_average_temp”.
- After downloading the CSV file, I performed analysis in three ways i.e. using pandas with Python, Excel and Tableau.

## Analysis Using Python and Pandas

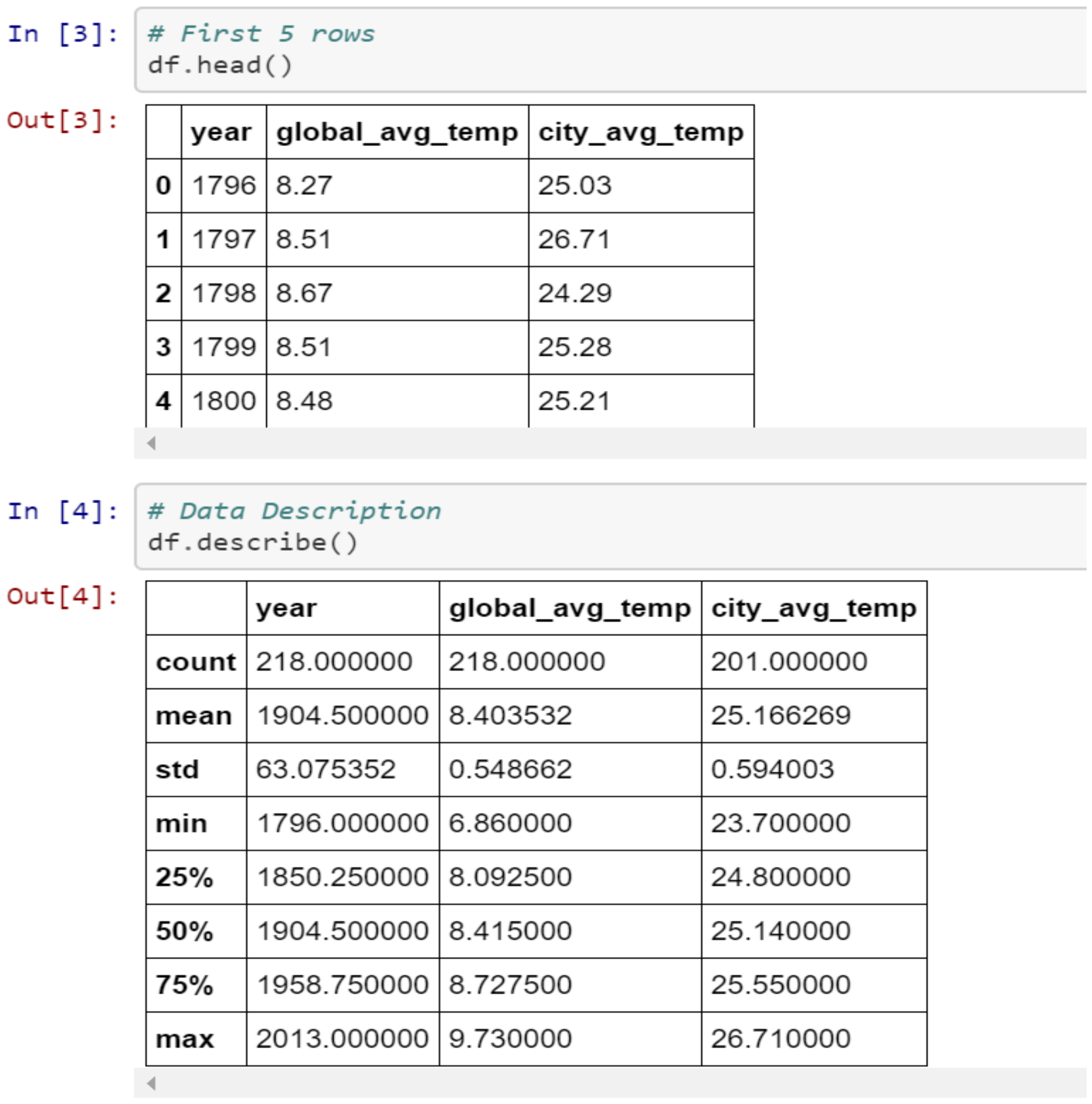
- For this step, I downloaded the CSV file and imported that in my Jupyter Notebook using Pandas as follows:



```
In [1]: # Import Dependencies
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline

In [2]: # Load the data
        df = pd.read_csv('./yearly_average_temp.csv')
```

- After reading in the data, I first checked the data for any imbalance in classes i.e. missing values in data using the following:



```
In [3]: # First 5 rows
        df.head()
```

Out[3]:

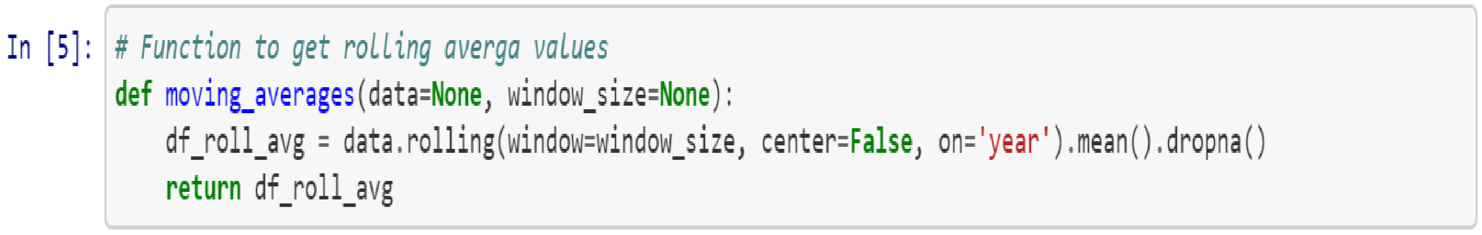
| | year | global_avg_temp | city_avg_temp |
|---|---|---|---|
| **0** | 1796 | 8.27 | 25.03 |
| **1** | 1797 | 8.51 | 26.71 |
| **2** | 1798 | 8.67 | 24.29 |
| **3** | 1799 | 8.51 | 25.28 |
| **4** | 1800 | 8.48 | 25.21 |

```
In [4]: # Data Description
        df.describe()
```

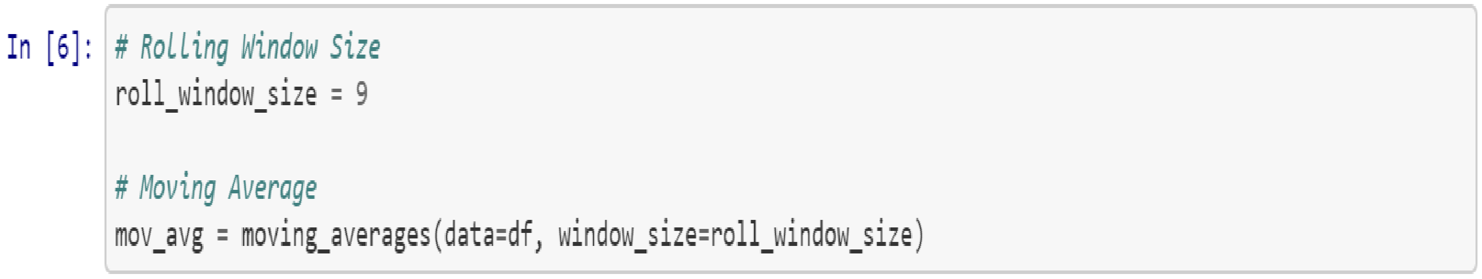
Out[4]:

| | year | global_avg_temp | city_avg_temp |
|---|---|---|---|
| **count** | 218.000000 | 218.000000 | 201.000000 |
| **mean** | 1904.500000 | 8.403532 | 25.166269 |
| **std** | 63.075352 | 0.548662 | 0.594003 |
| **min** | 1796.000000 | 6.860000 | 23.700000 |
| **25%** | 1850.250000 | 8.092500 | 24.800000 |
| **50%** | 1904.500000 | 8.415000 | 25.140000 |
| **75%** | 1958.750000 | 8.727500 | 25.550000 |
| **max** | 2013.000000 | 9.730000 | 26.710000 |

- From the above tables, I could see that the “year” and “global_avg_temp” had same number of values but the column for “city_avg_temp” had 7 missing values. So, in order to balance the data, I decided to drop the rows with “NaN” for the “city_avg_temp” values.
- I wrote a function to calculate the **Moving Averages** for “global_avg_temp” and “city_avg_temp” columns to see the trend in the trends in temperature readings over years using rolling function from pandas as follows:



```
In [5]: # Function to get rolling averga values
        def moving_averages(data=None, window_size=None):
            df_roll_avg = data.rolling(window=window_size, center=False, on='year').mean().dropna()
            return df_roll_avg
```

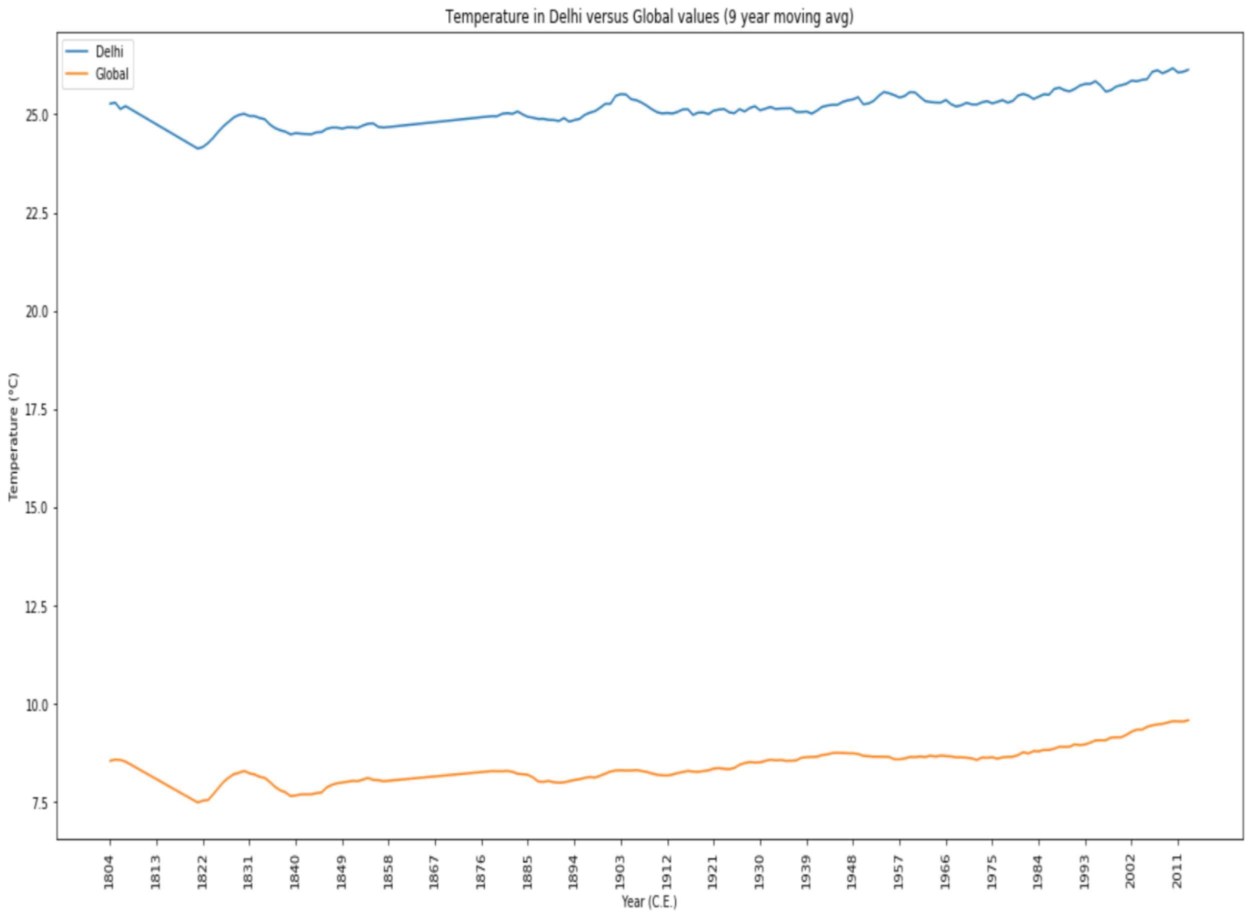


```
In [6]: # Rolling Window Size
        roll_window_size = 9

        # Moving Average
        mov_avg = moving_averages(data=df, window_size=roll_window_size)
```

- In above function, the inputs to the “moving_averages” function are “data” i.e. the original dataframe and “window_size” i.e. size of the moving window. This is the number of observations used for calculating the statistic. Each window is of a fixed size.
- Additionally, this function takes care of dropping the rows for which the “city_avg_temp” is a “NaN” by using **“dropna()”**.
- For this dataset, I tried different values for the rolling window sizes but settled down for a value of 9.
- Next, I plot the moving average temperature using matplotlib. The final plot looks like follows:

```
Out[8]: Text(0.5,1,'Temperature in Delhi versus Global values (9 year moving avg)')
```

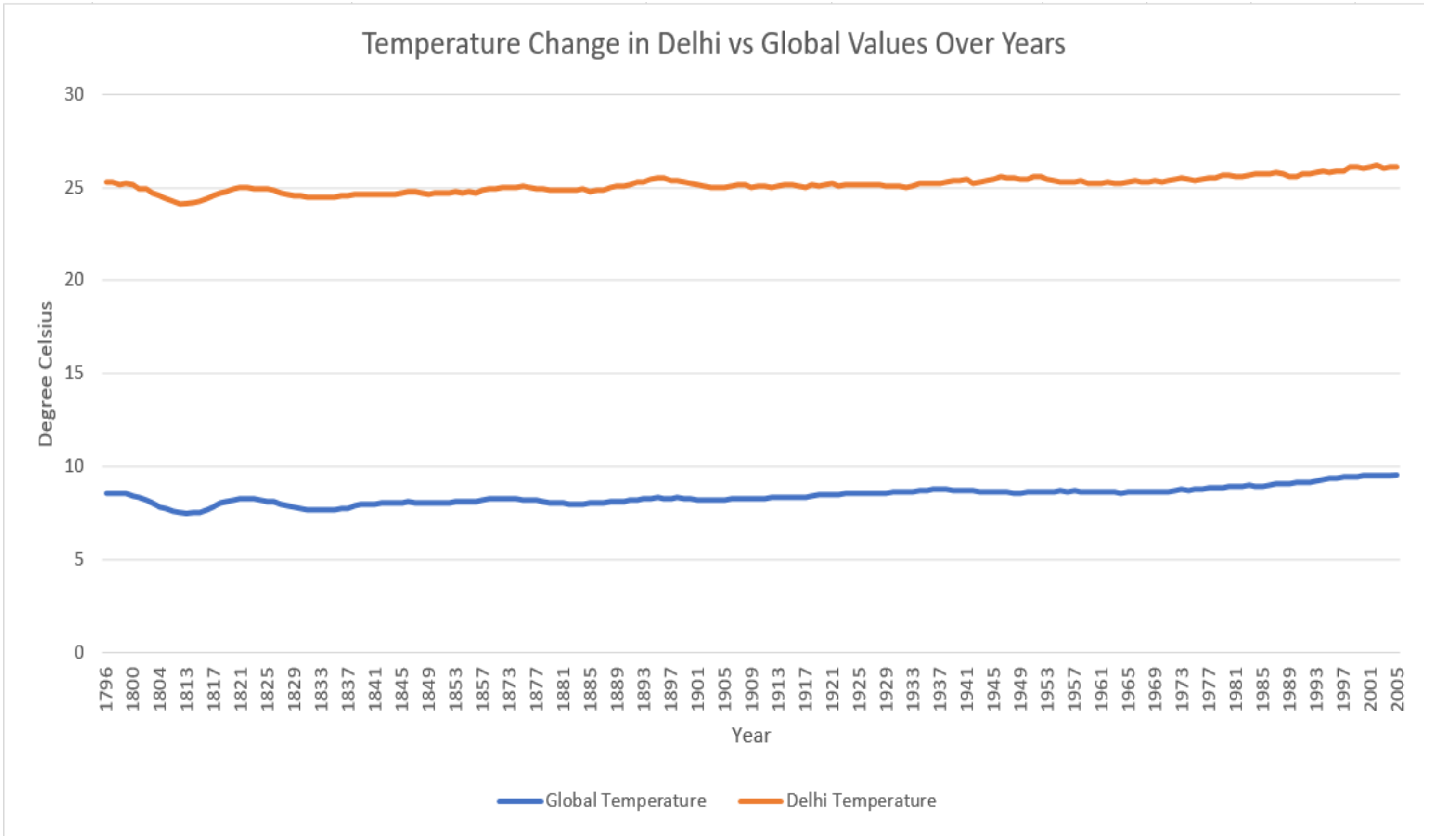


**Observations:**

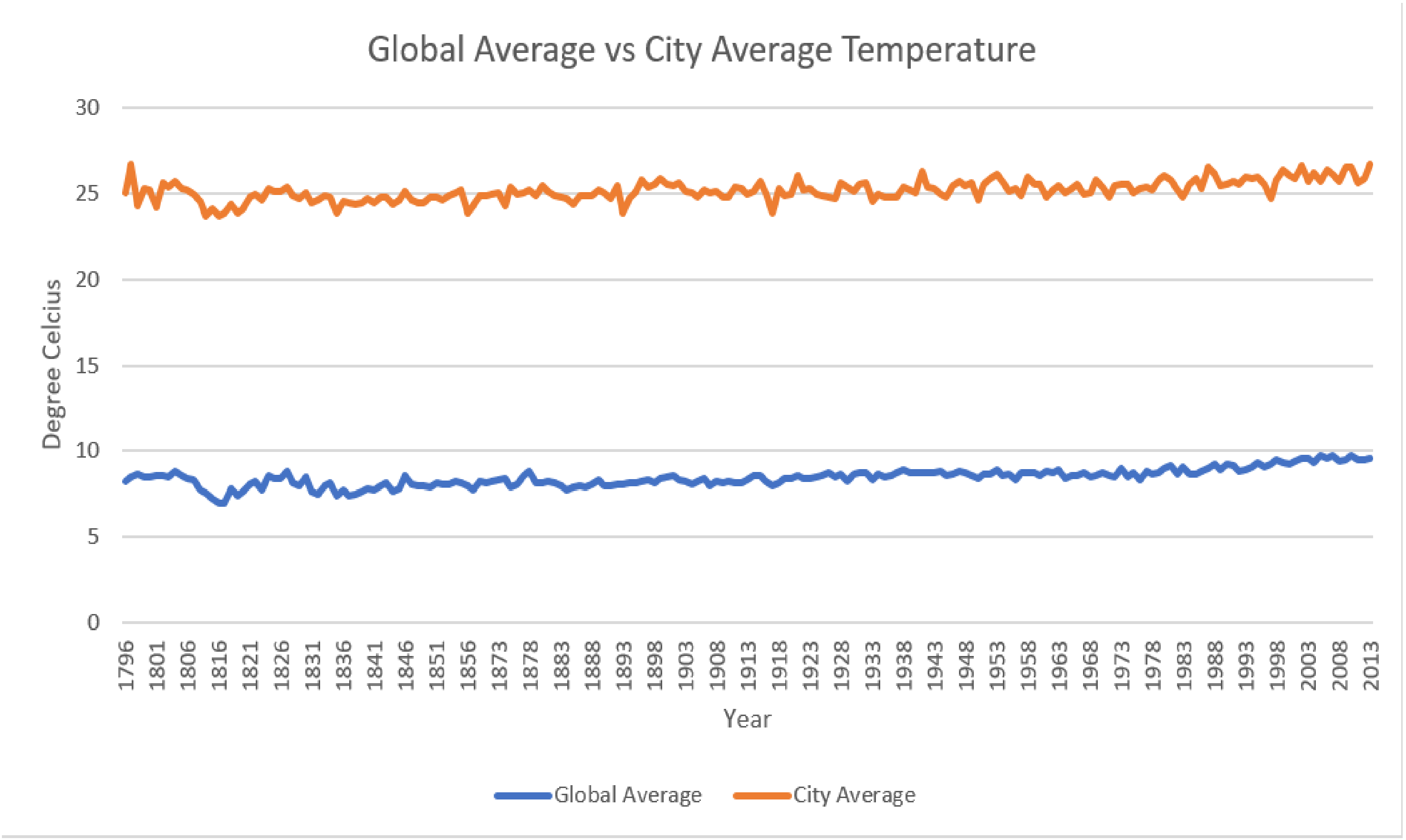
- From the above plot, we can see that the global moving average temperature lies between **7.15 to 8.67** Degrees Celsius.
- For Delhi, the moving average temperature lies in the range of **23.8 to 26.5** Degree Celsius.
- The above plot shows that the moving average temperature for Delhi is more than the Global moving average temperature which infers that **Delhi is hotter** in comparison of the **Global temperature**.

## Analysis Using Excel

- For this analysis, I used the CSV data obtained from above after extraction and used excel tools to get the moving averages for the data i.e. city and global temperatures moving averages.
- Just like the previous approach, I kept the moving average window to 9.
- The line chart for the global temperature vs the city temperature over years looks like following:



- Similarly, the plot for the global averages vs the city average looks something like this:

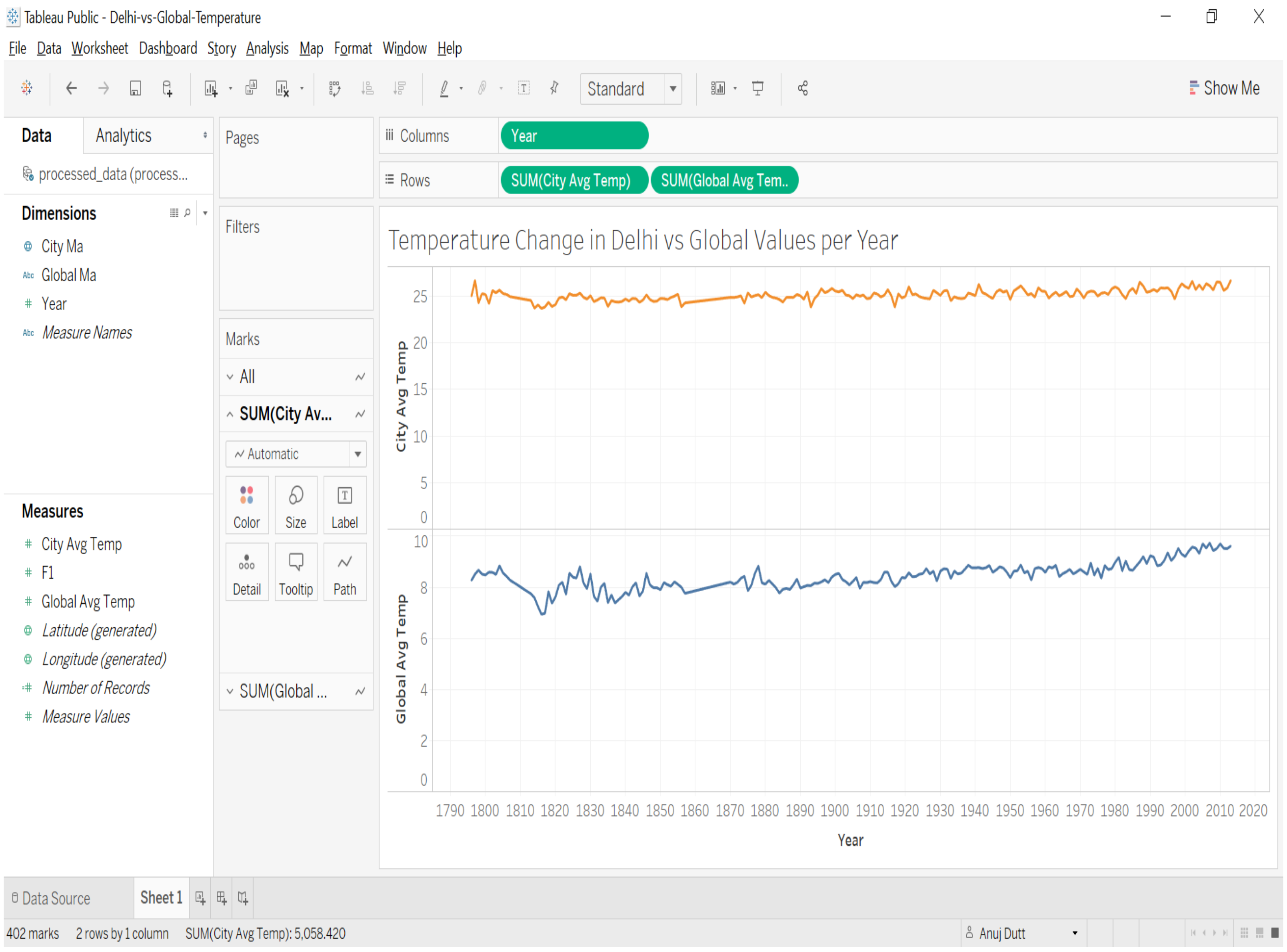


**Observations:**

- According to the above two plots, we can see that both the Global temperatures and the city temperatures have been consistent over time.
- During the period of **1806 to 1823**, both the Global as well as Delhi temperatures went down.
- A similar trend is seen for the years **1826 to 1845** where both the graphs move downwards.
- Since then, there have been some small ups and downs in both the Global and Delhi temperatures as well as their moving averages but mostly, the temperature has been increasing consistently.
- This proves that both the Global and Delhi's temperature is rising at a high rate based on the increasing trend as seen in the plots above starting from about **1846 till today**.

**Hence, Global Warming is Real!!!**

# Basic Data Visualization using Tableau



- The above plots in tableau shows the City and the Global average temperature plots over years.
- These plots provide us the same information as the plots above with excel and python.