Project Name

Explore Weather Trends

Mohammed Rayeez
India Chennai

My Notebook for this Project

Overview

In this project, you will analyze local and global temperature data and compare the temperature trends where you live to overall global temperature trends.

Goals

1. To create a visualization
2. Prepare a write up describing the similarities and differences between global temperature trends and temperature trends in the closest big city to where you live..

Tool's Used

SQL : **TO Extract Data From Database**

Python library :

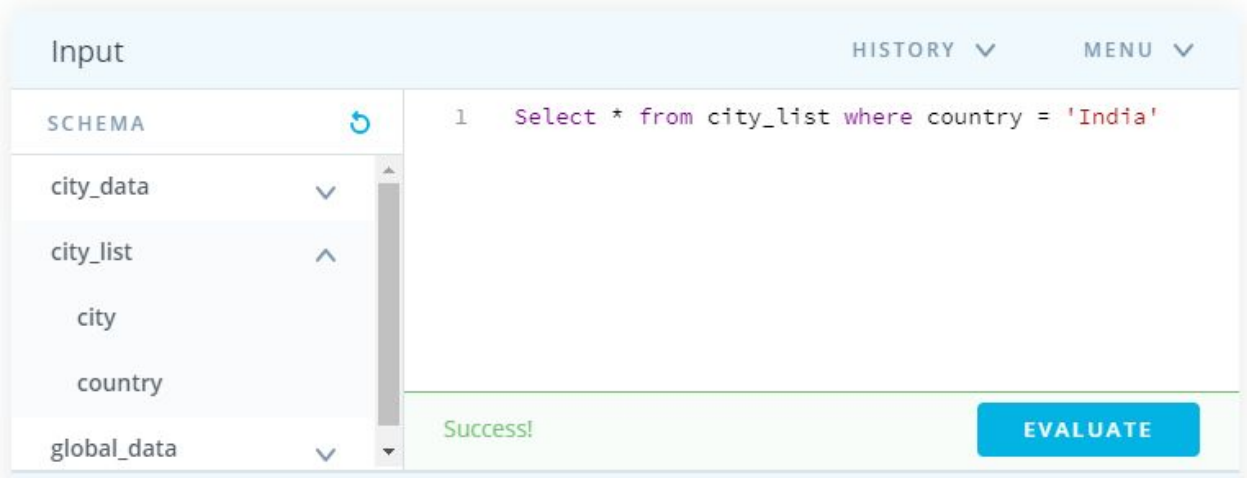
Pandas : **For manipulation of data and arrange the data based on my requirement**

Matplot : **For plotting Graphs**

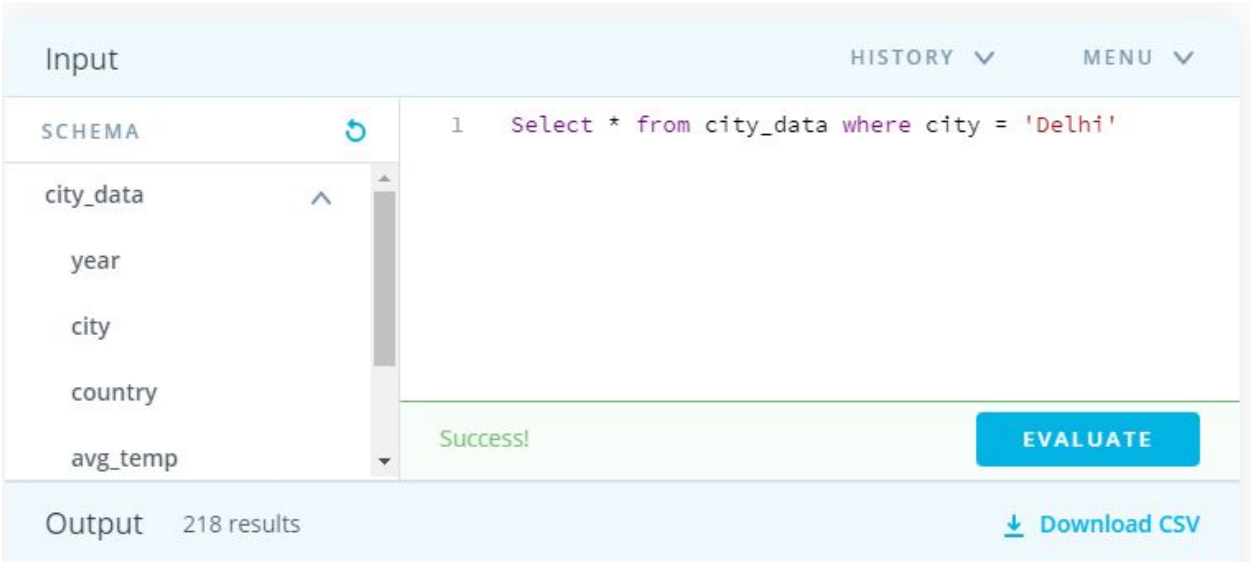
Process

Extracting data from Udacity Database (Using SQL)

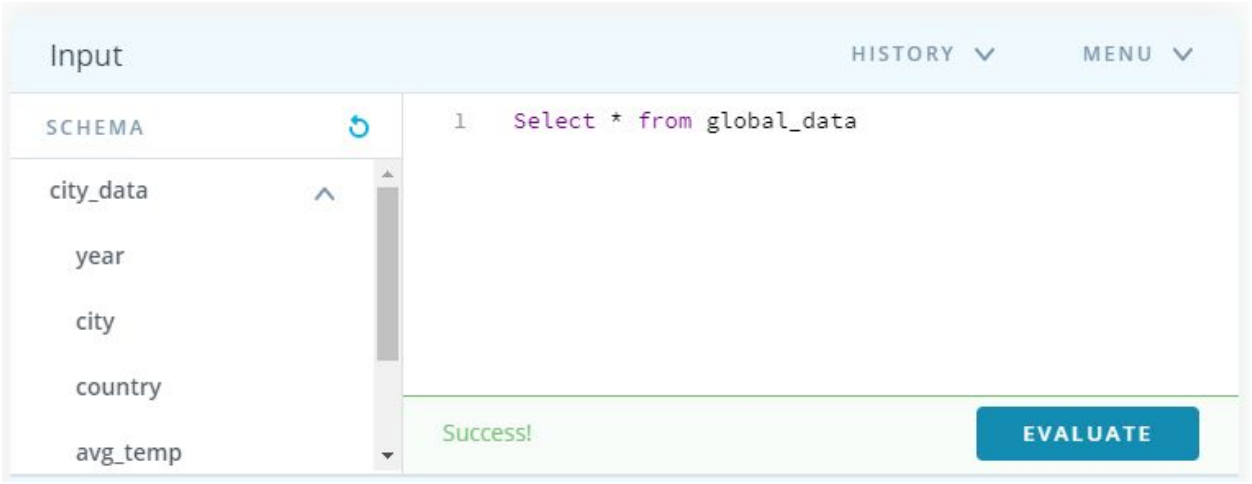
Step 1 (**Finding My City using Country**)



Step 2(**Fetching Dataset from database**)



Step 3(**Fetching Global Dataset from database**)



Processing and Plotting Data (Using Jupyter NoteBook)

Step 1

Importing Lib

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Step 2

Saving Global Dataset and My City Dataset from CSV file to respective variable's

```
global_ds = pd.read_csv("/content/drive/My Drive/Udacity Colab/Gloabl.csv")
city_ds = pd.read_csv("/content/drive/My Drive/Udacity Colab/Delhi.csv")
```

Step 3

Setting Year as index in both dataset

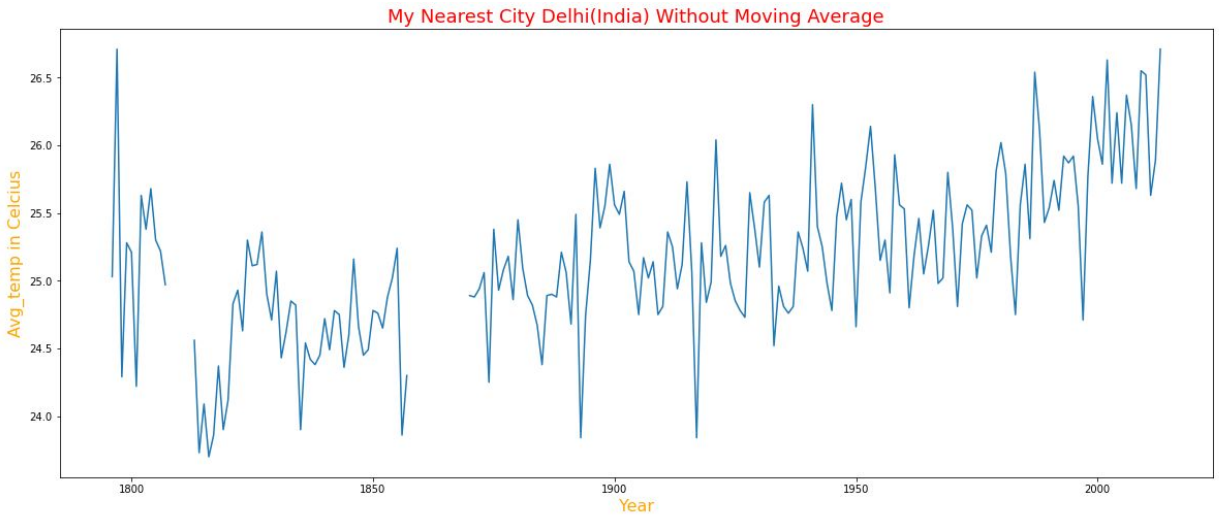
```
global_ds = global_ds.set_index('year')
city_ds = city_ds.set_index('year')
```

Step 4

Plotting My City DataSet without Moving Average

```
city_ds.avg_temp.plot(figsize=(20,8))
plt.title('My Nearest City Delhi(India) Without Moving Average' , fontdict={'fontsize': 18, 'color' : 'Red'})
plt.xlabel('Year' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
plt.ylabel('Avg_temp' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
```

which gives significant fluctuations

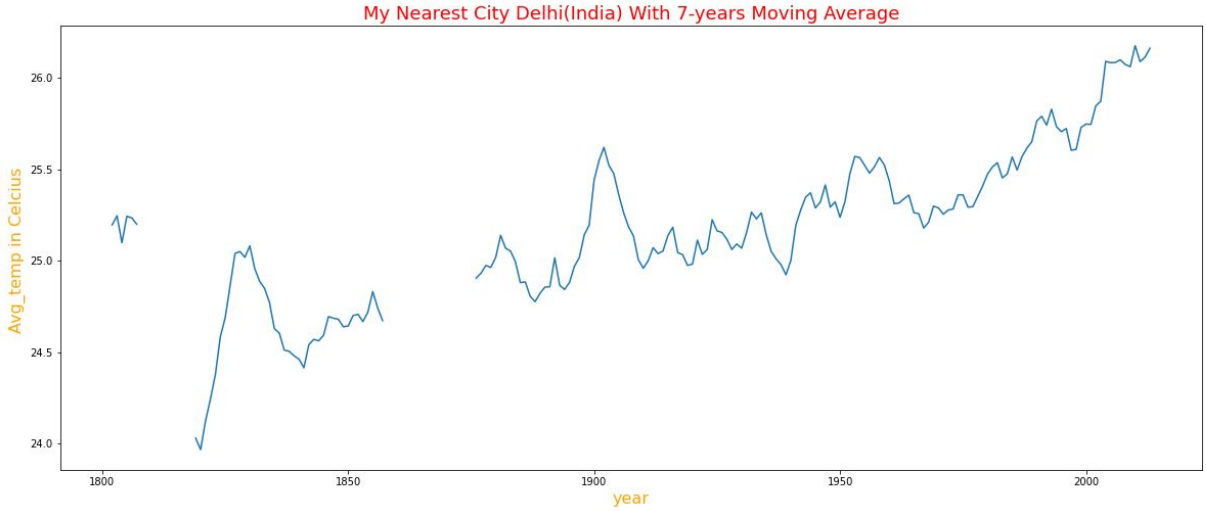


Step 5

We changing the normal distribution dataset with the rolling window size of 7

```
plt.title('My Nearest City Delhi(India) With Moving Average' , fontdict={'fontsize': 18, 'color' : 'Red'})
plt.xlabel('Year' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
plt.ylabel('Avg_temp' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
city_ds.avg_temp.rolling(window=7).mean().plot(figsize=(20,8))
```

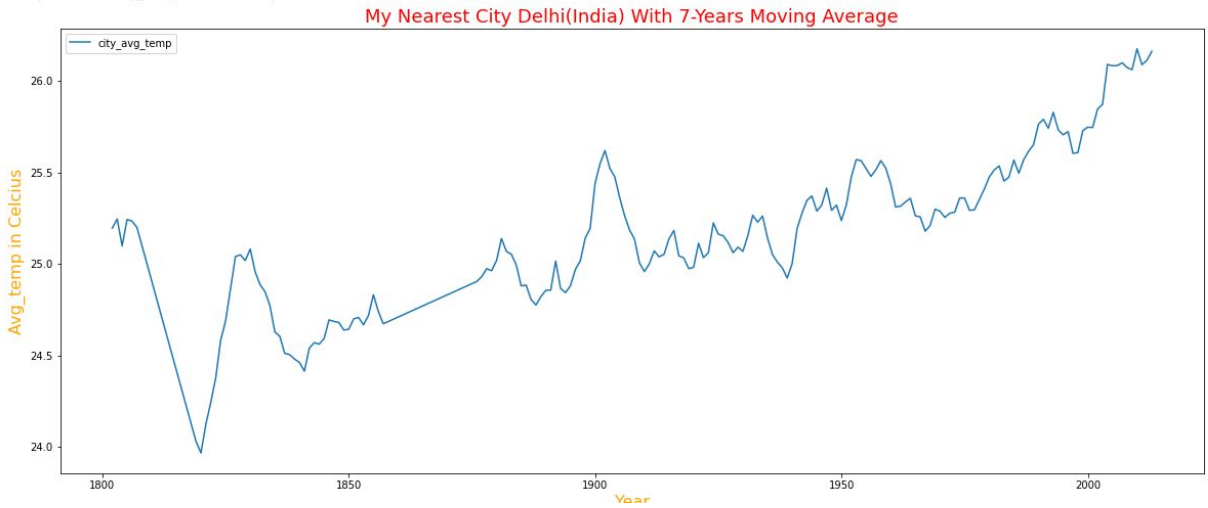
The trend got better by using a rolling average, and if we cover the gab which was generated by moving average it will give some figures



Step 6

Removing Nan Value we get this result in My City

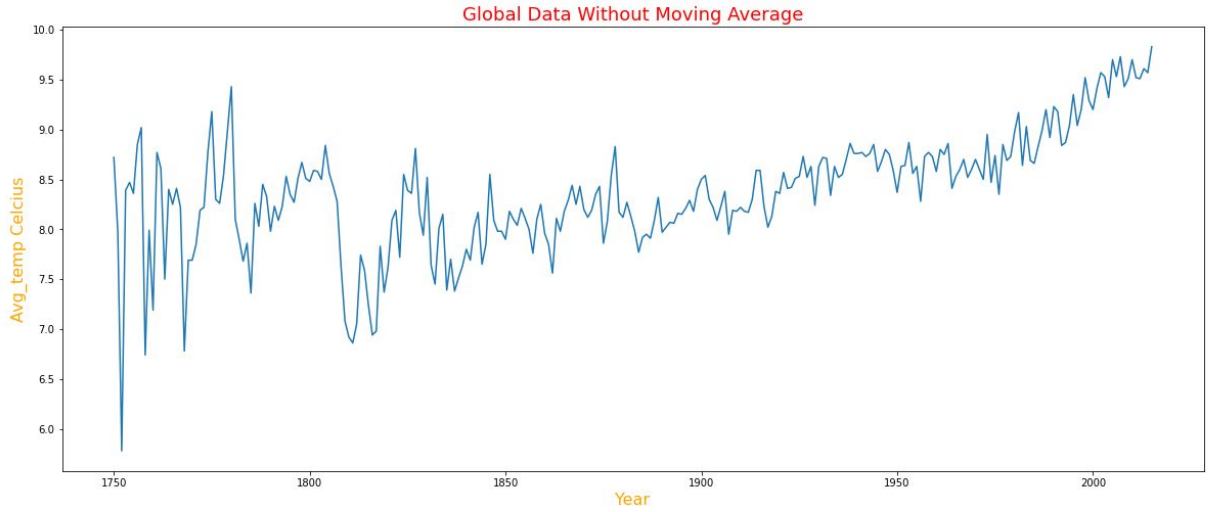
```
df=city_ds.rolling(window=7).mean()
df.dropna().plot(figsize=(20,8))
plt.title('My Nearest City Delhi(India) With Moving Average' , fontdict={'fontsize': 18, 'color' : 'Red'})
plt.xlabel('Year' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
plt.ylabel('Avg_temp' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
```



Step 7

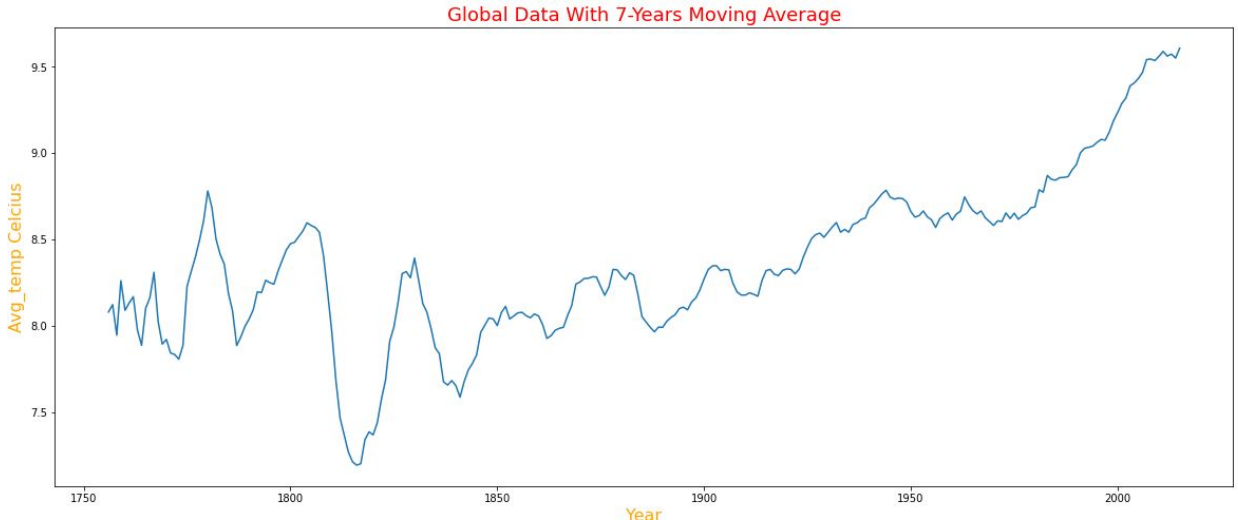
Plotting Global DataSet without Moving Average

```
1  global_ds.avg_temp.plot(figsize=(20,8))
2  plt.title('Global Data Without Moving Average' , fontdict={'fontsize': 18, 'color' : 'Red'})
3  plt.xlabel('Year' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
4  plt.ylabel('Avg_temp' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
```



Step 8

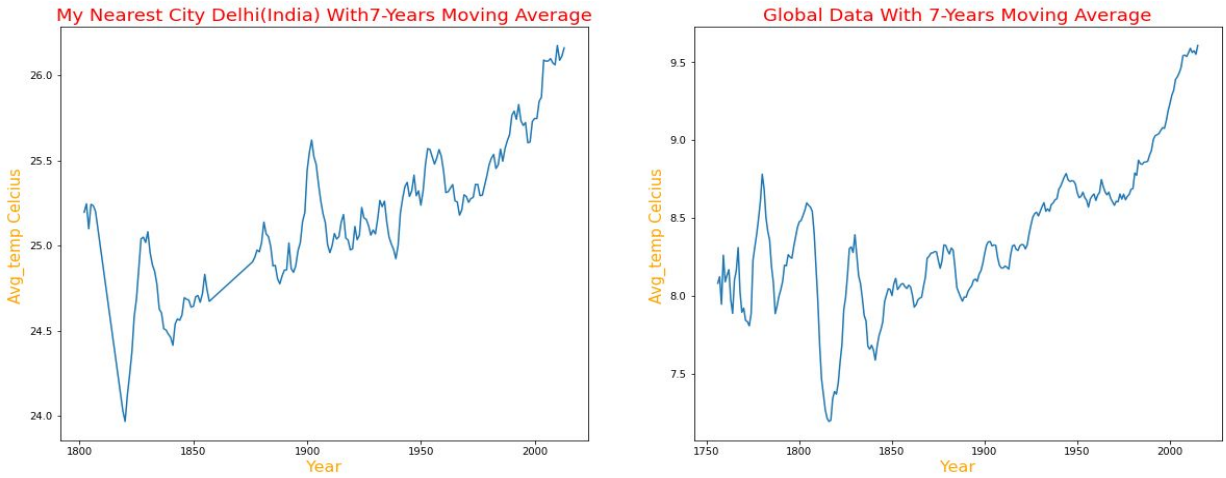
Plotting Global DataSet with Moving Average
& we don't find any space in between the avg_temp after setting moving average so we left it as it is



Step 9

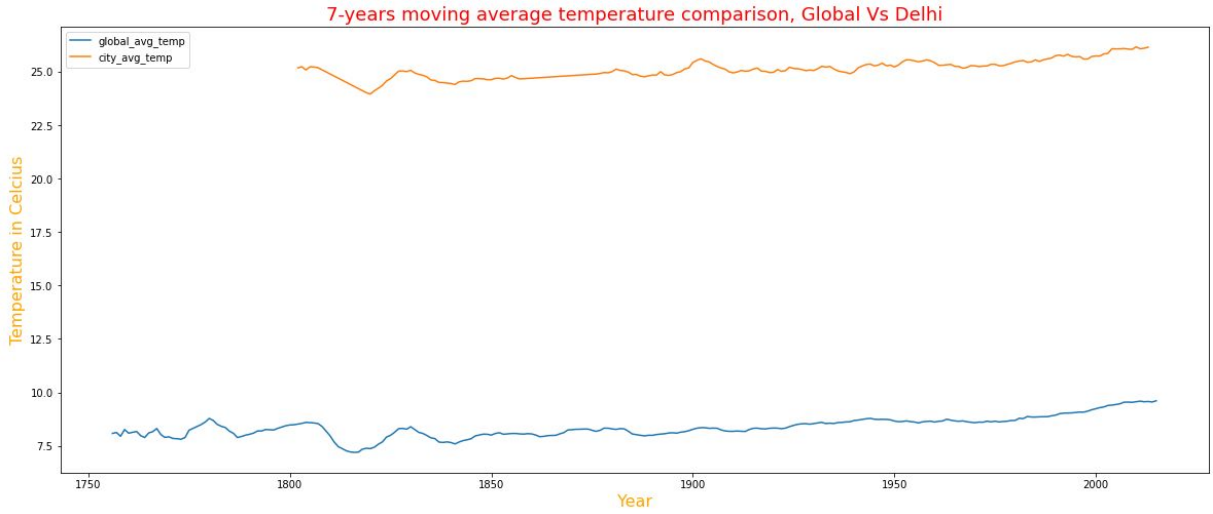
Before Merging two-liner lines in one graph I plot it side by side so we have some visual clarification

```
# First Graph My City

plt.subplot(1, 2, 1)
df=city_ds.avg_temp.rolling(window=7).mean()
df.dropna().plot(figsize=(20,8))
plt.title('My Nearest City Delhi(India) With Moving Average' , fontdict={'fontsize': 18, 'color' : 'Red'})
plt.xlabel('Year' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
plt.ylabel('Avg_temp' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})

# Secong Graph Global

plt.subplot(1, 2, 2)
global_ds.avg_temp.rolling(window=7).mean().plot(figsize=(20,8))
plt.title('Global Data With Moving Average' , fontdict={'fontsize': 18, 'color' : 'Red'})
plt.xlabel('Year' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
plt.ylabel('Avg_temp' ,  fontdict={'fontsize': 16, 'color' : 'Orange'})
```



Step 10

Merging two-linear lines in one graph **Orange** is My city and **Blue** line is Global



Observations :

1) **As per the five quartile range, we can see the significant difference between Global Avg_temp and City Avg_temp**

```
1  global_.describe()
```

	year	avg_temp
count	266.000000	266.000000
mean	1882.500000	8.369474
std	76.931788	0.584747
min	1750.000000	5.780000
25%	1816.250000	8.082500
50%	1882.500000	8.375000
75%	1948.750000	8.707500
max	2015.000000	9.830000

```
1  city_.describe()
```

	year	avg_temp
count	218.000000	201.000000
mean	1904.500000	25.166269
std	63.075352	0.594003
min	1796.000000	23.700000
25%	1850.250000	24.800000
50%	1904.500000	25.140000
75%	1958.750000	25.550000
max	2013.000000	26.710000

If we notice the min temp of city and the max temp of global itself has a huge difference so we can

2) **Based on the data we can say My Nearest City (Delhi) Shows Significant Oscillate in temperature when compared to Global temperature**

3) **Global Temp doesn't show huge fluctuation but it significantly rising up eventually the city temp also raising up**

4) **We can't match the temp of Global and My City because the min temp of the city is for greater than global max temp**

```
global_.sort_values(by=['avg_temp'],ascending=False).head()
```

	year	avg_temp
265	2015	9.83
257	2007	9.73
260	2010	9.70
255	2005	9.70
263	2013	9.61

```
city_.sort_values(by=['avg_temp'],ascending=False).head()
```

	year	city	country	avg_temp
217	2013	Delhi	India	26.71
1	1797	Delhi	India	26.71
206	2002	Delhi	India	26.63
213	2009	Delhi	India	26.55
191	1987	Delhi	India	26.54

5) **Globally Huge temp changes taken place from 2005 to 2015 and Globally the temp spike up on the year of 2013 based on my city and globally**

This Conclude My Observation

Thank You ...