EXPLORE WEATHER TRENDS

April 30 -2020

Udacity -Data Analyst Nanodegree

NIMMY GEORGE

Project 1

Explore Weather Trends

Overview

In this project, I have analyzed local temperature of San Jose, United States in accordance with the global temperature data and compared. I had been provided with a database on Udacity portal from where I have to extract, manipulate and visualize the data as in the following goals.

Goals

1. Extraction of data from the database and export to CSV file
2. Making a chart visualization based on extracted data
3. Observation based on chart

Tools Used

1. SQL: To extract the data from the database
2. Python: For calculating the moving average and plotting a line chart

3. ANACONDA - Jupyter Notebook: For writing python code and making observations

4. Google Sheets: Having a look at the data and writing project

5. Google Docs: To use DOCS colour formatting to write pretty codes.

Steps

STEP 1 - Extraction of Data from provided Database

I have done the following activity in order to make a relevant dataset. I have learnt the SQL basics from lessons provided before this project. I have also done an introductory course on SQL and relational database from which I have used some concepts.

1. To see which cities are available for "San Jose" in the given dataset:

```
SELECT * FROM city_list WHERE city = ‘San Jose’;
```

2. To make a relevant dataset by joining the two tables. But, I found from the SCHEMA that both city_data and global_data contains the same column named 'avg_temp'. So I have changed the names of the columns respectively in order to have distinct columns.

```
ALTER TABLE city_data RENAME COLUMN avg_temp to  city_average_temp;
ALTER TABLE global_data RENAME COLUMN avg_temp to  global_average_temp;
```

3. Now I have written the following code in order to join the two tables and have the relevant data:

```
SELECT global_data.year, global_data.global_average_temp, city_data.city_average_temp FROM global_data JOIN city_data ON global_data.year = city_data.year WHERE city LIKE 'San Jose' ;
```

Now I have got an option of downloading the file as CSV format. Downloaded as “results.csv”.

STEP 2 - Python Code for Making the Line Chart

For making the line chart, I have used some python libraries and wrote these codes on Jupyter Notebook.

```
import numpy as np                    # Importing the important Libraries
```

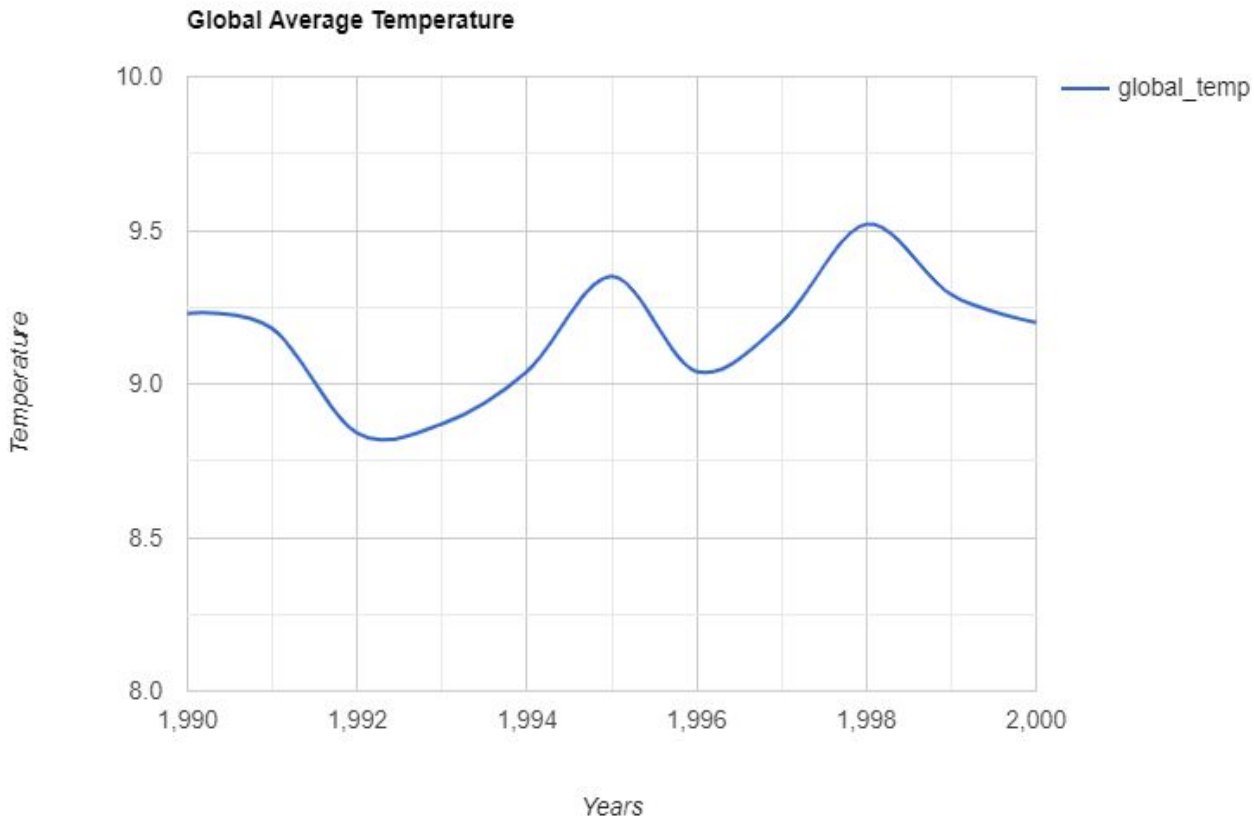
```
import pandas as pd                          # for loading data into the notebook
from matplotlib import pyplot as plt   #for making a line chart
data = pd.read_csv( "results.csv" )       # Importing the extracted Data Set
```

Now I have defined a function for the calculation of moving averages in order to get smooth graph.

```
 def moving_avg (mA_range, data_input):    # function that calculates the MOVING AVERAGE
Output=data_input.rolling(window = mA_range, on = "city_average_temp" ).mean().dropna()
 return output
mA_value = 150
chart_moving_avg = moving_avg(mA_value, data) # Function Calling with the range of moving Average


plt.plot(chart_moving_avg [ 'year' ], chart_moving_avg [ 'global_average_temp' ], label = 'Global' )                                    # Drawing the graph: Global Temperature
plt.legend()
plt.xlabel ( "Years" )
plt.ylabel ( "Temperature (°C)" )
plt.title ( "global_average_temp" )
plt.show ()
```

So I have got the following output:

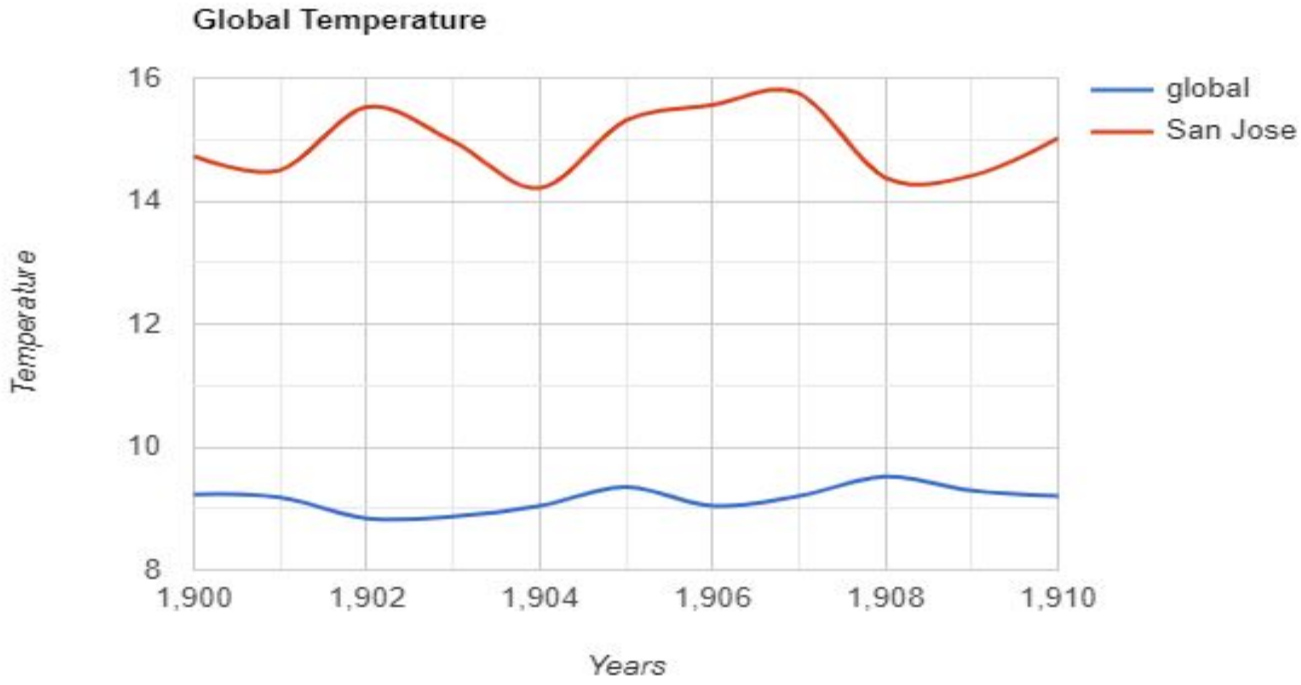


I have separately analysed the global data in order to check and distinguish it from combined data of San Jose and Global Average temperatures.

Now combined with New Delhi data,

```
# Drawing the graph: New Delhi and Global Temperature
# Introducing this line in the previous code above the "global_data plot command"
plt.plot(chart_moving_avg [ 'year' ], chart_moving_avg [ 'city_average_temp' ], label = 'San Jose' )
```

I got the following graph:



RESULT - Observations:

I have observed that, if I choose a short range for moving average, say 10, I will get messy line in the graph. Also, the range of the "Years" on x-axis becomes short. And if I use a larger moving average, say 100, I will get a relatively smooth graph and range of x-axis is longer.

Observation from the Line Charts:

1. The chart of San Jose Vs Global Temperature: Very big difference between the average temperature of San Jose and that of the world.

2. Since I have got a slightly inclined straight line for global data. I have separately plotted the graph of global data. (The first graph)

3. From the first graph: I noticed that global temp. is increasing quiet constantly with

years by 0.1-degree centigrade.

4. Again coming back to the second chart, I observe that San Jose have

temperatures greater than the global average.

5. If I draw a tangent line touching the two troughs or crests of the line of San Jose, I see that there is a consistent change between this line and the line of global average

over time.

6. The city of New Delhi seems to be hotter than any other cool place in the world.

From this, I came to the conclusion that the regions that lie between the tropic of Capricorn and the tropic of cancer will have greater temperatures as compared to the global average.

7. Seeing the graph, the temperature of the world is on a constant rise.

This is all of my observations for the data used in this project.

Key Considerations:

1. Unit of Temperature: Centigrade, on Y-axis

2. Years showed on X-axis

3. Different colours of lines for the city and global average

4. Use of Matplotlib library for visualization

5. Applied moving average on City data in order to get a relatively smooth line

6. Defined a function for easy code

7. Saved all of the codes in .ipynb files (Jupyter Notebook) for later reference and

regenerations or revisions.

REFERENCES:

1. Change Column Name:

https://www.1keydata.com/sql/alter-table-rename-column.html.

2. Joining the tables for better analysing:

http://www.dofactory.com/sql/join.

3. Calculation command for Moving Average in Python used for drawing graphs:

http://www.learndatasci.com/python-finance-part-3-moving-average-tradingstrategy/

Parameters used for .rolling().

https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame

.rolling.html.