# Data Analyst Nanodegree

# Project 1 – Explore weather data trend

**Author: Rogério da Silva Carneiro** [**https://github.com/rogeriocarneiro/Udacity\_NanoDegree\_DataAnalyst**](https://github.com/rogeriocarneiro/Udacity_NanoDegree_DataAnalyst)

**Summary**

[Data Analyst Nanodegree 1](#_Toc99733124)

[Project 1 – Explore weather data trend 1](#_Toc99733125)

[Step 1 – Extracting the data from the database. 2](#_Toc99733126)

[Step 2 – Scanning the data 3](#_Toc99733127)

[Step 3 – Calculating the moving averages and data visualization 3](#_Toc99733128)

[Step 4 – Data analysis 5](#_Toc99733129)

[Correlation coefficient 5](#_Toc99733130)

[Linear Regression on temperature relation 5](#_Toc99733131)

[Derivative analysis 7](#_Toc99733132)

## Step 1 – Extracting the data from the database.

It’s shown below the SQL codes run to extract the data from the database provided and the 10 first rows from the query results.

|  |  |
| --- | --- |
| EXTRACTING CITY TEMPERATURE | EXTRACTING GLOBAL TEMPERATURE |
| select  cd.year  , cl.city  , cl.country  , cd.avg\_temp  from city\_list cl  left join city\_data cd on cd.city = cl.city  where cl.country = 'Brazil'  and cl.city ilike 'Rio De Janeiro' | select \* from global\_data |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | year | city | country | avg\_temp | | 0 | 1832 | Rio De Janeiro | Brazil | 23.05 | | 1 | 1833 | Rio De Janeiro | Brazil | 24.11 | | 2 | 1834 | Rio De Janeiro | Brazil | 23.27 | | 3 | 1835 | Rio De Janeiro | Brazil | 22.73 | | 4 | 1836 | Rio De Janeiro | Brazil | 22.91 | | 5 | 1837 | Rio De Janeiro | Brazil | 22.29 | | 6 | 1838 | Rio De Janeiro | Brazil | 22.81 | | 7 | 1839 | Rio De Janeiro | Brazil | 22.54 | | 8 | 1840 | Rio De Janeiro | Brazil | 23.32 | | 9 | 1841 | Rio De Janeiro | Brazil | 22.97 | | |  |  |  | | --- | --- | --- | |  | year | avg\_temp | | 0 | 1750 | 8.72 | | 1 | 1751 | 7.98 | | 2 | 1752 | 5.78 | | 3 | 1753 | 8.39 | | 4 | 1754 | 8.47 | | 5 | 1755 | 8.36 | | 6 | 1756 | 8.85 | | 7 | 1757 | 9.02 | | 8 | 1758 | 6.74 | | 9 | 1759 | 7.99 | |

## Step 2 – Scanning the data

This way we can have a first picture of what’s inside the datasets, as the count of rows with no non-null values for each series.

Firstly, the shape information of each dataset was retrieved. Then both data frames were merged using the **‘year’** column in order to have them sharing the same time span, with or without non-null values.

With the data frame normalized using the year column, it’s verified the first year where both temperatures were measured. This way the analysis can be carried out from that year forward, in order to minimize the chance of using a not balanced data frame. This year is 1832 as seen in the third row at the table below.

|  |  |  |
| --- | --- | --- |
|  | CITY TEMPERATURE | GLOBAL TEMPERATURE |
| df.shape() | Columns: 4  Rows: 182 | Columns: 2  Rows: 262 |
| df.info() | Data columns (total 5 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 year 266 non-null int64  1 global\_temp 266 non-null float64  2 city 182 non-null object  3 country 182 non-null object  4 city\_temp 175 non-null float64  dtypes: float64(2), int64(1), object(2) | |
| df.loc[df[\_].first\_valid\_index()].iloc[0] | 1832 | 1750 |

## Step 3 – Calculating the moving averages and data visualization

The moving average chosen to perform this analysis considers a window of 10 years. The moving averages of the city\_temp and global\_temp and the difference between them were calculated and appended to the data frame simply using the rolling() and mean() methods as show below:

df['global\_10y\_MA'] = df['global\_temp'].rolling(10).mean()

df['city\_10y\_MA'] = df['city\_temp'].rolling(10).mean()

df['diff'] = df['city\_temp'] - df['global\_temp']

df['diff\_MA'] = df['diff'].rolling(10).mean()

Once these calculations were made, it was possible to generate a reasonable visualization on what the datasets contains.

The chart is presented in the next page, as follows.

Chart, line chart

Description automatically generated

## Step 4 – Data analysis

The boundary values for the moving averages series are:

|  |  |  |  |
| --- | --- | --- | --- |
|  | CITY TEMPERATURE | GLOBAL TEMPERATURE | DIFFERENCE |
| MIN (YEAR) | 22.98 (1843) | 7.67 (1841) | 14.99 (1867) |
| MAX (YEAR) | 24.76 (2010) | 9.59 (2015) | 15.5 (1967) |
| VARIATION | 1.784 | 1.923 | 0.510 |

Other overall value that could explain more the behavior of the temperatures is the trend of the temperature change over the years and how many years it would take to the temperature rise in 1 oC. The calculated trend consists in a linear regression of both temperatures.

|  |  |  |
| --- | --- | --- |
|  | Slope ( oC / year ) | Slope ( years / oC ) |
| City temperature | ~ 0.0086 | ~ 116.7116 |
| Global temperature | ~ 0.0080 | ~ 124.6363 |
| Absolute difference | 0.0005447855984791923  or **5.45 x 10^(-4)** |  |

### Correlation coefficient

The very small difference in the slope of the trends for both measures leads to a expectation about the **Pearson's correlation coefficient** being high. This can be verified by the following method, still using the moving averages:

df['global\_temp\_10y\_MA'].corr(df['city\_temp\_10y\_MA'],

method = "pearson")

= **0.9677**548684994313

It confirms that the temperatures of the city of Rio de Janeiro and the temperature presented as global temperature are highly bind in a correlational way.

### Linear Regression on temperature relation

Using this high correlation, despite of the cause, it’s possible to predict with a certain level of accuracy for a still basic analysis. As in order to perform a linear regression using Numpy, it’s mandatory to consider only the non-null values of the moving averages.

mask = np.isfinite(y\_city\_MA) & np.isfinite(y\_global\_MA)

slope,interception = np.polyfit(y\_global\_MA[mask], y\_city\_MA[mask], deg = 1)

reg = slope,interception

trend = np.polyval(reg, y\_global\_MA)

The visualization of this linear regressions is show as follow:

Chart, scatter chart

Description automatically generated

The linear equation that represents this regression is:

City temperature = 1.0021917554158453 \* years + 15.259902131042605

Meaning that for each degree Celsius increase in global temperature, 1.0021917554158453 degree Celsius is increased in Rio de Janeiro temperature, keeping a difference of 15.259902131042605 degree Celsius between them.

Once defined a linear equation that represents data with a high correlation, it can be used to perform a basic predictable analysis – disregarding further technical rigor.

Creating the function below and applying an higher value to it, one can get the temperature in the Rio de Janeiro city based in the global temperature.

def rio\_temp (temp\_global):

return slope\*temp\_global + interception

rio\_temp(30)

# 45.32565479351796

### Derivative analysis

Given the current context of more concern around the human responsibility in climate changes, a valid analysis would be to verify any increment of temperature increase rate after some point in time.

From calculus, the temperature change rate on time can be related to the concept of first derivative of the temperature(year) function. So, any behavior variation on that first derivative would be captured calculating it’s derivative (the second derivative of temperature(year) function).

Text

Description automatically generatedBelow are printed the first and second derivative of Rio de Janeiro temperature over the years and the analog measures for Global temperature. The values in x axis represent the number of the measurement, not the year of the measurement.

As can be observed, at no point in time the second derivatives explicitly show a relevant increase. For the data gathered and shown, it’s uncertain to estimate any increase in global or local temperature due to human action.