Python: exploratory data analysis and visualization

by Olena Petrova, Data Analyst

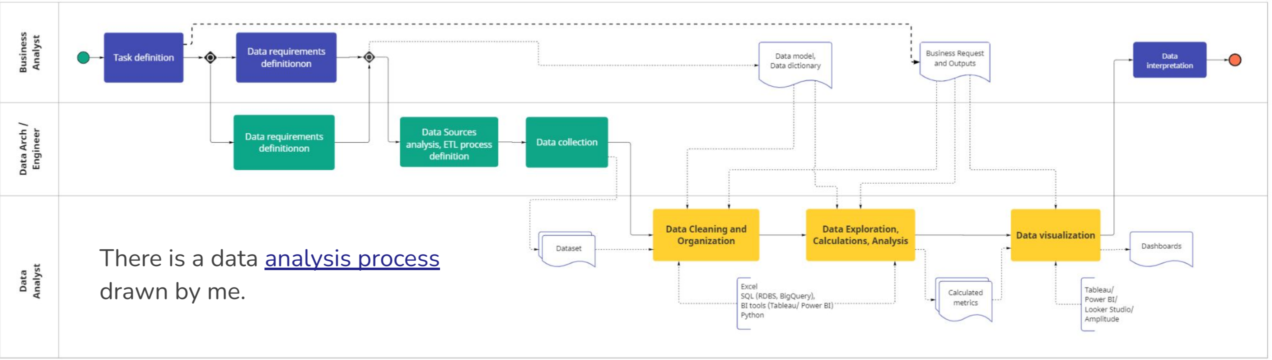
Project Scope

The current Case Study focuses on **exploratory data analysis and visualization with Python**.
There is no real-like business goal for this case, but there are **tasks to show particular data relations**.

My goal for this project is to demonstrate my **technical and problem solving abilities**.
Thus, it covers the following steps of the data analytics process:

- Outputs Requirements (p.3)
- Data Collection and Preparation (p. 4)
- Data visualization and required calculations (p. 5-9)

\+ Description of key factors that allowed me to solve technical issues (p.10)



There is a data analysis process drawn by me.

Output Requirements

The subject of analysis is the Facebook Ads campaigns data for the years 2021-2022. The data are stored in **two tables** in internal database: Campaigns and Ads daily data from Facebook.

It is required:

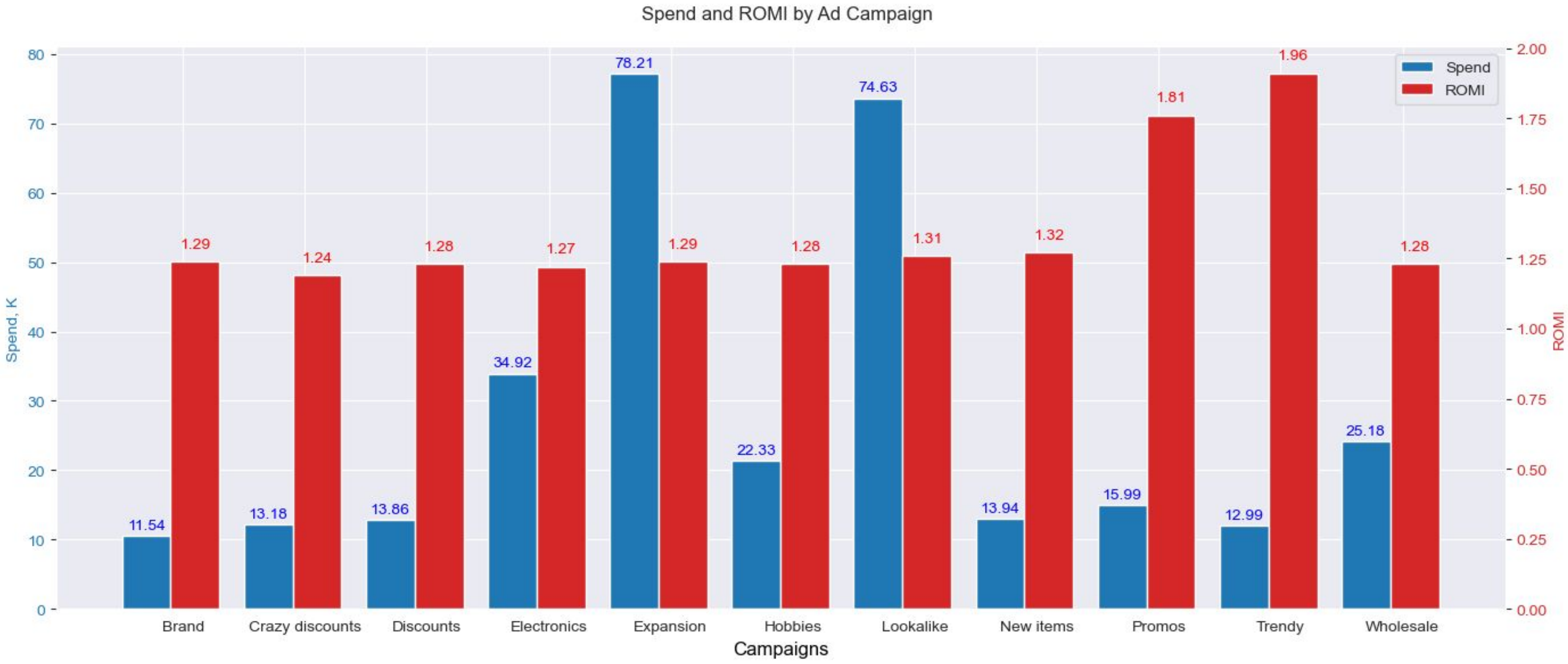
- For the **year 2021**: to show **effectiveness (ROMI) of each campaign** and compare it with **spends** for them.
- For the entire period (2021-2022): to **analyze dependency among the marketing metrics** (CPC, CPM, CTR, ROMI) and define:
    - if there are any correlations,
    - which dependency are the weakest
    - what does the metric “value” depend on
- Separately **visualize the relationship** of the metric “value” on defined metric(s)

The tasks should be completed **with Python**. The results should be represented as **reasonable quantity of pictures** for further presentation.

Data Collection and Preparation

- *DBeaver* was used to connect to database and data preparation and calculations (**SQL**)
  - The **tables were united** and required data were selected using CTE
  - Marketing **metrics were calculated**, CASE method was used to reasonably clean data
  - Dataset was exported to .csv file
- Dataset was imported into *Jupyter Notebook*.
  - Data were checked: the **column headers** are in place, **data format** is suitable for further calculations
  - Python libraries were imported: Pandas, Matplotlib, Seaborn
- Three visuals were defined to create:
  - Spend and ROMI totals by Campaign - **bar chart**;
  - **Correlation heatma**p of all the ads metrics with specification of 3 the strongest and 3 the weakest dependencies + specification of the metric that impacts Value data
  - **Scatter plot with the linear regression** for visualization of Value dependency

Spend and ROMI totals by Campaign



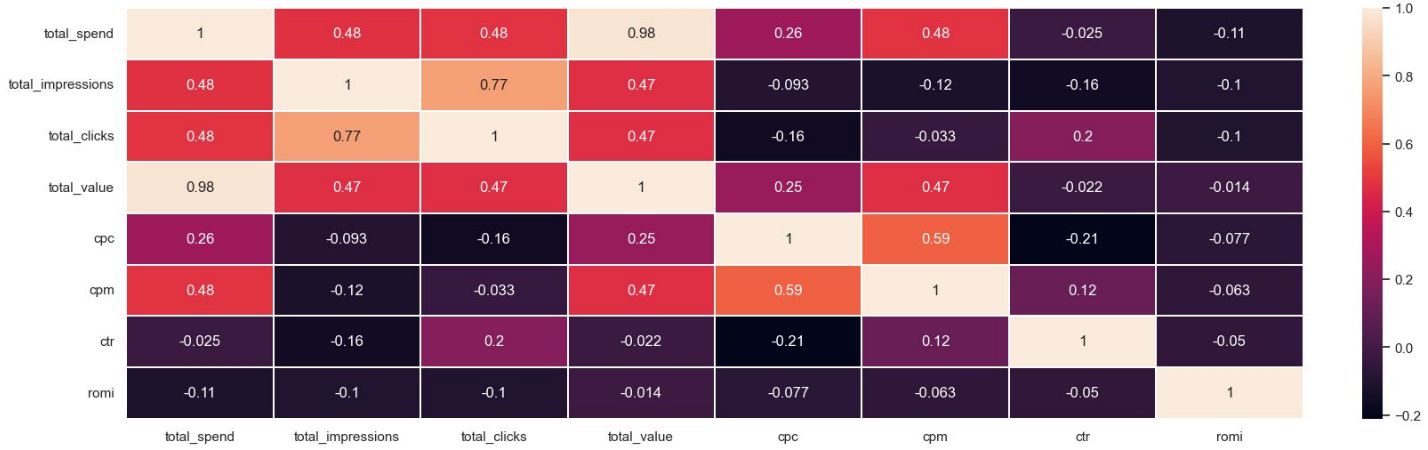
The challenges and applied approach are described on the next page

Spend and ROMI totals by Campaign

Approach and challenges

- **Grouped all the records** by Campaign column value. A subset with 3 calculated metrics was created. Campaigns were set as indices.
- **Two bar chart** subplots were created and placed one in front of another **sharing the x-axis**. That called the following challenges:
  - Two y-axis with different scales were created
  - Coordinates of bars were modified to place them only aside another
  - Grid of the front graph was disabled
- **Legend** was added (that required some time to find solution for such a viz).
- **Bar values** were added using **a custom function** (it also required effort to implement it in a user-friendly way, considering bars coordinates modification)

Facebook ads metrics dependency



The strongest correlation (>=0.6 or <=-0.6)

metric	pair	corr_index	corr_quality
total_value	total_spend	0.98	Strong
total_clicks	total_impressions	0.77	Strong
cpc	cpm	0.59	Moderate

The weakest correlation (the closest to 0)

metric	pair	corr_index	corr_quality
total_value	romi	-0.01	Weak
total_value	ctr	-0.02	Weak
ctr	total_spend	-0.03	Weak

"total_value" correlates with "total_spend" for 0.98

The challenges and applied approach are described on the next page

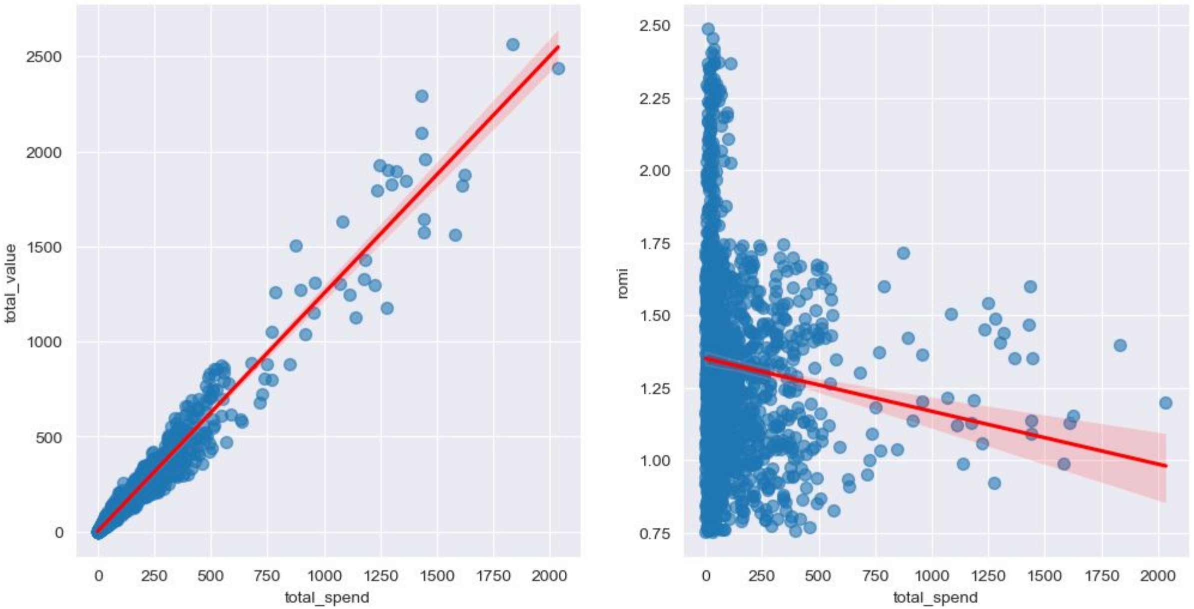
Facebook ads metrics dependency

Approach and challenges

- **Data Preparation and Calculation**
  - **Corr()** was applied to dataset considering numeric values only. Resulting dataframe was **unnested**, columns were named, numeric values were rounded
  - **Duplicated dependencies were removed,** such as, for example, ‘AA’ - parameters self-reference, ‘AB’ and ‘BA’ - the same parameters references. That wasn’t an easy issue to resolve.
  - Correlation quality parameter was added to the table. Based on general **correlation evaluation** >=0.6 or <=-0.6, but added the third quality degree and therefore had to use more complex where().
  - Selected 3 **the most strong** and 3 **the most weak** dependencies using **abs()**
  - The parameter that impacts “Value” was found. Its name and correlation index was extracted from the record.
- **Visualization**
  - A figure with **4 subplots of different sizes** was created.
  - Seaborn heatmap was placed in the first subplot. No problem.
  - Two tables in the 2nd and 3rd subplots called efforts to locate tables, to adjust cell size, and to add annotation
  - The f-string with the ‘Value’ explanatory parameter name and correlation index was placed in the 4th subplot

Value dependency visualization

Nothing of specific was with this graphs.
Just decided to demonstrate visual difference between strong and weak correlation.



Approach and problem solving

While creating visualizations I apield to google search and ChapGPT looking for specifications and solutions. Then I successfully applied the to my Python code, changing the proposed solution to fit my case.

These efforts success was achieved due to the following skills:

- Ability to specify the expected result
- Understanding the structure and logic of the programming language
- Ability to formulate the question
- Understanding of dependency of various parts of the code
- Ability to write a maintainable code