Implementation of a (Big) Data Management Backbone

Liam James Glennie England
Eduard Puga Creus

Introduction

The aim of this project is to build a well structured Big Data Management Backbone to be able to access it for descriptive and predictive analysis. This section of the project will focus on the formatted and exploitation zones and the process behind their construction. Figure 1 illustrates the structure of our solution.

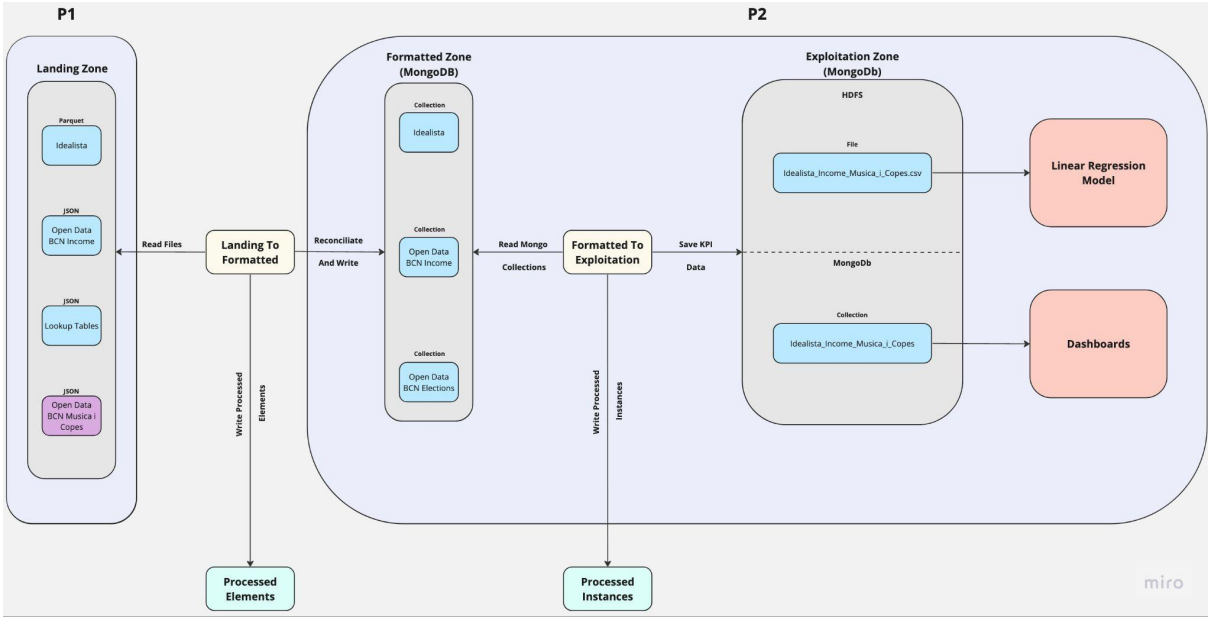

Figure 1. Project Stages

Data Sources

The given solution for part one of this project was taken. The data from this solution included multiple parquet files corresponding to the Idealista data. Next, the OpenDataBCN Income data had been converted to a MongoDB collection. Also, the Lookup Tables had also been converted to MongoDB collections. Finally, a third dataset was requested to be used in the solution. Despite the multitude of options, including academic performance divided by Barcelona districts, the OpenDatBCN Musica i Copes dataset was chosen. It contains the different establishments that can be found in Barcelona classified by neighbourhood and district. Some of the establishments include bars, cocktail bars, restaurants, fitness… The data came in JSON format and it assumed that the establishments found in the dataset are those that are registered in the Ajuntament de Barcelona as of the download date. That is if a bar were to be unregistered, the next time the file is downloaded, it will not contain this bar.

Formatted Zone

The idea behind the formatted zone is to retrieve the data from the landing zone and perform some type of reconciliation process and store in some distributed storage. First of all, the neighbourhood and district names and ids were modified to follow a standard, enforced by the lookup tables. So, for example, say that the Idealista data has as a neighbourhood name “Sants-Montjuïc” and OpenDataBCN Income has “Sants Monjuic”, the reconciled version is ¨sants montjuic¨. More importantly, the neighbourhood identifiers were also reconciled to enable joining the data sources later in the exploitation zone.

In terms of the storage technology, both HDFS and MongoDB were evaluated. Nonetheless, MongoDB was considered to be more suited for our solution due to its flexibility in terms of storing schema-less data allowing for schema evolution. Next, HDFS is more suited for large-scale distributed data, meaning that with the small size of the data to be processed, it seemed like the advantages of HDFS would not be exploited. Finally, as no real-time analysis was going to be performed, paying the price of HDFS’ strong consistency was not needed. Coupling these ideas with MongoDB’s ease of use, it seemed that it was the correct choice for the formatted zone.

Finally, RDDs (Resilient Distributed Datasets) were chosen over Dataframes to gain experience using this data structure. Gaining experience also played a role in our choice of MongoDB over HDFS.

KPIs

Model

The KPIs calculated can be divided into two categories. First, the predictive KPI involves predicting the price of an Idealista listing based on the attributes of the flat/house, and information of the neighbourhood, like the average family income, population and types of establishments, as well as the number of each type of establishments. A linear regression model was developed to calculate this KPI. Some preprocessing steps were made, such as removing missing values, one-hot encoding of categorical variables, removing outliers. The resulting model produced a RMSE of 188061.42€ on the validation set and 192959.90€ on the test set. The model does not produce any useful results, however as optimising it is not the objective of the project, no further work was done. After validation, the model was persisted into disk. The model was developed in a Jupyter notebook called model.ipynb.s

Dashboard

The next two KPIs are more related to descriptive analysis, and Tableau was used as a visualization tool. The first KPI, uses data from the Idealista and OpenDataBCN income dataset in order to find a correlation between an Idealista listing price with the family income within the neighbourhood. For this purpose a scatter plot was created, as it can be seen in the image below as the RDF increases, there is a slight tendency for the sale price to ascend too.

Figure 2. Scatter plot correlation between RDF and Price

The second KPI aims to find which property type is a listing more likely to be based on the number of establishments types within the same neighbourhood. We started by creating a dashboard for each establishment type showing the percentage of sales properties that were from a specific type (flat, chalet…), the problem was that for all the establishments types the vast majority was type flat, and there was not too much relevant information that could be extracted. So we decided for each type of property how many establishments were of each kind (each kind with his own scale because there are many more flats that the other property types combine). As can be seen in the image below, the establishment type where there are more chalets is “Bars and Pubs musicals”.

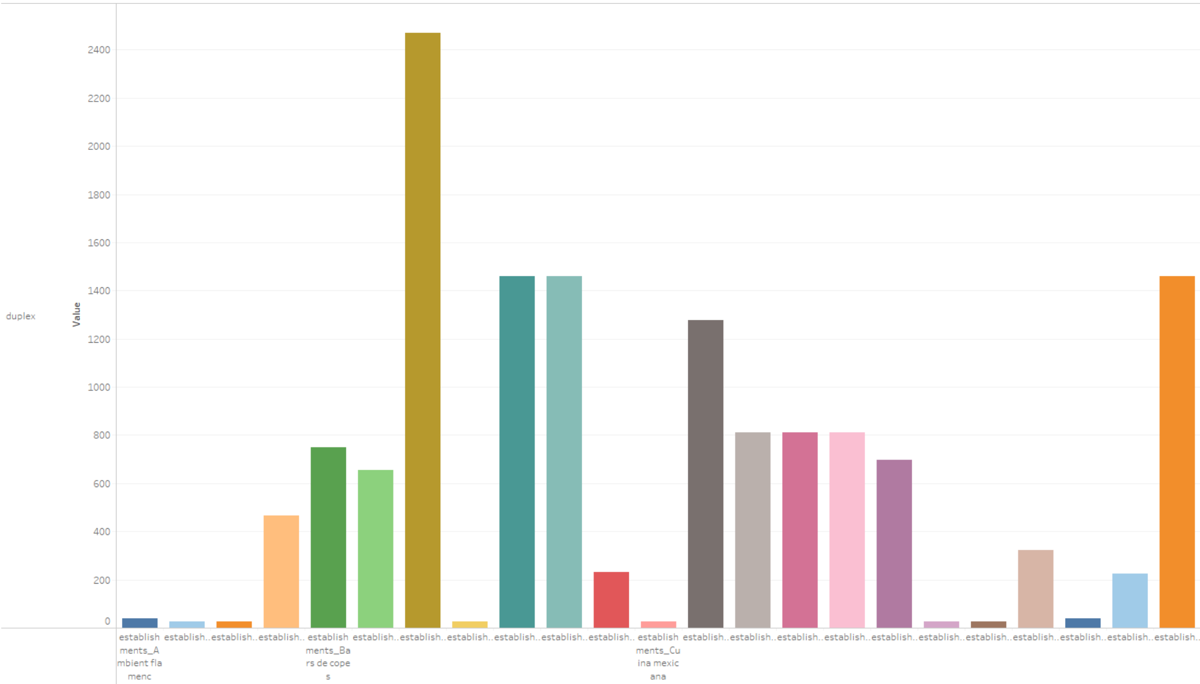

Figure 3. Histogram of the property type and establishments types

For visual representation, please refer to the example video, where it can be seen the three dashboard completes.
https://drive.google.com/file/d/1bYFBAshcRcz0CLk2cPDjVyRHsX6rG3R1/view?usp=sharing

Exploitation Zone

The exploitation zone is the region where the data from which the KPIs are extracted from. In our solution, there is a split between the two categories of KPIs: predictive and descriptive. We will comment on the implementation of each type separately.

As a reminder, the predictive KPI involved using attributes from all the data sources to try and predict the price of an Idealista listing. Therefore, a join operation had to be performed between the three mongo collections: Idealista, OpenDataBCN Income and OpenDataBCN Musica i Copes. So, first, all *None* identifiers were removed from all three datasets. Next, as the timelines of the data available does not coincide, the temporal aspect was ignored. Therefore, the Income dataset was aggregated obtaining the average population and RFD per neighbourhood. Aggregation was also performed on the Musica i Copes dataset by performing a count of each type of establishment. Lets explain the structure of this dataset to further understand how this aggregation was applied. As can be seen in Figure 4, each establishment could have multiple establishment types. The establishment type can be found in the attribute "secondary filters data". Thus, the data of each establishment was converted to a tuple `((neighborhood_id, establishment), 1)` where the one was eventually used to perform the count aggregation. The final shape after all the operations were performed resulted in: `(neighborhood_id, {'establishments': {'Bar': 3, 'Cocteleria': 2, 'Restaurant': 1}}).`

```
Unset
[{ "registerID": 1,
  "secondary_filters_data: [
       {"id": 1, "name":"Bar", ...},
       {"id": 2, "name":"Cocteleria", ...},
       {"id": 3, "name":"Restaurant", ...},
  ]
 },
 {...}
]
```

Figure 4. Musica i Copes data structure

Next, for joining the data sources, Income and Musica i Copes were joined first as they were of a similar size. Before joining the resulting RDD with the Idealista data, a repartition between four partitions was performed on the idealista RDD because it was much bigger than the other RDD. This difference in size between two RDD when joining can lead to performance issues due to inefficient data distribution, imbalance between workers, and data reshuffling.

Finally, this RDD was flattened and exported to a CSV which was then uploaded to an HDFS server. As the data was going to be fed to a machine learning model, it had to be converted to a tabular format and for ease of use CSV was elected as importing them to Spark Dataframes is relatively easy. As CSV was the elected format, we chose HDFS over MongoDB because we could store directly in the server without having to change the format.

For the descriptive KPIS, we also needed to integrate the three datasets to examine the correlation between the Idealista sales prices and with the family income within the neighborhood, and to be able to observe the relationship between the establishment types of a neighborhood and the property type of the Idealista listings. Once we had joined all three datasets with RDD operations, we needed to store the data in a database for later extracting and visualising through dashboards.
We selected MongoDB for this part of the exploitation zone for several reasons. First, it's schema-less, allowing us to store and retrieve data in a very flexible way, which is ideal for evolving data structures. This flexibility makes it ideal in case the dataset changes or we want to add new parameters in the dashboarding part. Second, MongoDB is designed horizontally, enabling the distribution of data across multiple servers as the volume increases. Lastly, MongoDB is known for its high performance in large-scale data processing, making it extremely efficient for data retrieval in visualization tools, such as Tableau, which we used for this project.