Big Data Management
Master's in Data Science
BDM - Project P2
June 21st, 2022

FIB

Meritxell Arbiol……..…….meritxell.arbiol@estudiantat.upc.edu
Emmanuel F. Werr…….....emmanuel.werr@estudiantat.upc.edu

Introduction

For this second part of the BDM project, we are picking up from the persistent zone. We have utilized the given solution by the professor where Idealista listings are in .parquet files from HDFS and OpenDataBCN data is in .json files ingested already in mongo DB. For P2 of this project we are tasked with:

- Integrate and reconcile the data stored in MongoDB and HBase by saving it in the Formatted zone using technologies such as Spark.
- Calculate 3 KPIs from this integration, in which we are going to do descriptive and predictive analysis, and store them in the Exploitation zone.
- Work with data that comes to us by streaming and make real-time predictions with it using the model generated by the predictive analysis calculated previously.
- Visualize the data of the KPIs calculated for our descriptive analysis so that the results are more understandable.

Pipeline implemented:

The Pipeline implemented to integrate, calculate and store the KPIs we have generated is:

1. **alldata_format.py:** Here we take the landing zone data and transform it to integrate it into the formatted zone.
2. **KPI_exploit.py:** Once the data has been integrated with our formatted zone, we calculate the 3 KPIs, where we have 2 descriptive and one predictive.
3. **streaming_spark.py:** Here we predict with the model that we have generated in the previous step the data that is coming to us in real time by streaming.
4. **visualization.py:** In this part, we have the visualization of the descriptive KPIs that we have saved in step 2.

Datasets used:

Our datasets for this project are:

- **Idealista**
    - Downloaded data from their search API as JSON documents, each containing Idealista listings in a circular area (1.5km radius) with the center being a random point within the Barcelona grid.
    - API requests were made on a daily basis.
- **OpendataBCN - Income**
    - Territorial income distribution in the city of Barcelona at the neighborhood level.
    - They are updated yearly and are available from 2007 to 2017.
- **OpendataBCN - Lloguer Preu**
    - List of average apartment rent prices per neighborhood.
    - They are updated yearly and are available from 2014 to 2021.

In the third point, we have our extra dataset, called "Lloguer Preu". In it, we have the list of monthly rents and per square meter (m2) according to the different neighborhoods and districts of Barcelona from 2014 to 2021 divided by quarters.

KPIs

The KPIs that have been calculated are two for descriptive analysis and one for predictive analysis.

- **KPI 1:** We have calculated the average rental prices requested, and the list of apartments that are for rent according to the neighborhood. In this way, we have the list of neighborhoods, and for each neighborhood we have the average price asked for rent and the list of flats/apartments that are for rent in that same neighborhood.
- **KPI 2:** We have calculated the correlation between the monthly rental price and the family income index since we believe that there should be a high correlation between these two values, and indeed there is since we obtain a 98% correlation between both variables.
- **KPI 3:** Here we have created a model to predict the number of rooms a flat/apartment will have according to the price and the neighborhood in which it is located.

Data Structure in the Formatted Zone

For the integration of the three datasets that we have uploaded to the formatted zone, we have decided to create two different structures. We have decided to do it like this because for two of the three KPIs one way of structuring the data works well, and for the third KPI, the other way works better.

We know that in the formatted zone we must upload the data already transformed so that when we want to apply the KPIs we do not have to do more transformations than those already done when uploading the data to the formatted zone. For the formatted zone we have worked in mongo, it is where we have uploaded the data, and to make it more understandable, the DB is called "formatted", where inside it we have two collections, one called "data", and the other called "nested_data".

In both collections, we have exactly the same information only the structure changes, where one is in nested format, and the other is not. The data that we have selected to store in the formatted zone because they are relevant to us are: *“Neighborhood”, “Neighborhood Id”, “Neighborhood Code”, “District Name”, “District Code”, “Property Code”, “Date”, “Price”, “Size”, “Rooms”, “Bathrooms”, “Latitude”, “Longitude”, “Operation”, “Property Code”, “Floor”, “RFD”, “POP”, “Surface Price (€/m2)” and “Monthly Price (€/month)”.*

The above-mentioned data seem to us important, some of them for the analyses we have done, and others more thinking in possible future other analyses that could be done since the data are of interest.

It is also worth mentioning that each of the datasets has different years, so it was very complicated to take the year into account and join the different datasets according to the year, so what we have done is to keep the data of the most recent year as we believe it is the most realistic value at present.

These are the two structures stored in the formatted zone, each one in a different collection:

- ***data* collection:** We have a total of 3905 lines, as this is the number of apartments we have, excluding those that are repeated, always keeping the most recent. This format works well for our predictive analysis as we take into account the listing of the different apartments, so it is easier to structure in which we have an apartment with all its data per line.
- ***nested_data* collection:** We have a total of 50 lines, as we have a neighborhood for each line. Within each neighborhood, we have all the information about this, and we also have a nested list with the different apartments that are for rent for that neighborhood, along with their characteristics. This second collection suits us well for the two descriptive analyses. We don't take into account the list of apartments that are for rent, so we only use some of the variables that we have in these 50 lines, and not 3905, so the problem is simplified quite a bit. In addition, from a visual point of view, it is also much more understandable since everything is more organized, although it is also true that the level of metadata used for this second structure is higher than for the previous one.

Data Structure in the Exploitation Zone

In the exploitation zone, we have our KPI tables saved. KPI 1 and KPI 2 (these are the two descriptive analyses mentioned above) are stored in mongo, in a DB called "exploitation", where we have two collections there: “kpi1” and “kpi2”.

In the “kpi1” collection we have for each line of the neighborhood, the average price of that neighborhood, the number of apartments for rent there, and so on for each, having a total of 50 lines (one per neighborhood).

In the “kpi2” collection we have two columns (one for the average monthly price of listings for each neighborhood, and one for the most recent RFD for each neighborhood). We use these attributes to calculate the Pearson correlation between them.

For “kpi3” it is a bit different regarding the way of saving the data compared to the previous two. In this case, we have saved a model in a folder called *"exploitation"*, since it is also part of our exploitation zone. The model that we have created is a Random Forest Classifier, to predict the number of rooms that any apartment will have depending on the “*Neighborhood Id”* and “*Price”*.
In this same part is where we divide the data into the train (70% of the data) and test (30% of the data) to train the model and then validate it using accuracy measures. Once validated with the test data, the accuracy obtained is ~0.4, so the Test Error is ~0.6.

Data Stream Predictions

We have used the model generated in the previous section that is stored in the exploitation zone, to predict the data that is coming to us by streaming, and printing the data that is coming to us on the screen (*“Neighborhood Id”* and *“Price”*) together with the prediction of the number of rooms.

When executing the straming_spark.py file, so that it is not running infinitely, we ask the user for how many seconds he/she wants to be receiving values by the stream along with the prediction made by the model. When this time passes, the execution will end.

KPI Visualization Dashboard

As the final step in our pipeline, we developed a script called ‘visualize.py’. This script connects to the Exploitation Zone in MongoDB and imports the collections for KPI 1 and KPI 2. Then it creates the dashboard visuals for the information we wanted to convey using Dash and Plotly. The dashboard remains open and running in Localhost until a Keyboard interrupt. A summary of the information portrayed in the dashboard is given below:

Graph #1 (KPI 2) - Correlation between monthly average price and RFD of the previous year for each neighborhood. They have a Pearson correlation of ~0.98.

Graph #2 (KPI 1) - Bar plot showing the average listing price of all listings for each given neighborhood. The neighborhoods were sorted by average price in order to provide a nice visualization. Pedralbes has the highest average price of all listings at ~2.3 Million Euros.

Graph #3 (KPI 1) - Bar plot showing the total number of available listings for each given neighborhood. Here the neighborhoods are still sorted by average price (same as in the graph above) so the visualization doesn’t look so clean. However, this was done intentionally so that it is easy to compare average price and total listings of the same neighborhood using both plots simultaneously (it is easier because the neighborhoods are in the same location). Sants and Dreta de l’Eixample have the highest number of available listings at 350, but as we can see from the graph above, a big difference in the affordability of listings.