NOVA IMS

BIG DATA ANALYSIS PROJECT

A ML, Graph and Streaming Project in Databricks using PySpark

Dinis Fernandes 20221848
Dinis Gaspar 20221869
Inês Santos 20221916
Luis Davila 20221949
Sara Ferrer 20221947

Introduction

In today’s world, the term “Big Data” has become significantly noteworthy for governments, hospitals, companies, it is everywhere.

Big Data refers to the vast volumes of structured and unstructured information generated at extraordinary rates by everyone and everything. Storing and analyzing this data can reveal unique opportunities, insights and provide competitiveness between companies.

When we talk about Big Data, we also need to refer to the 4 V’s associated with it, that were in our minds in the course of the creation of these notebooks.

- **Volume**: The huge amount of data generated daily.
- **Variety**: The diverse formats of data, including structured data like databases, semi-structured data like JSON files, and unstructured data like text and video.
- **Velocity**: The high speed at which data is produced. And the need to quickly generate insights from it.
- **Veracity**: The validation techniques to ensure accurate data and therefore its analysis.

Databricks and PySpark are essential for big modern data analytics, addressing the 4 Vs of Big Data and leveraging advanced technologies. They handle Volume by processing massive datasets across distributed systems using PySpark's RDDs and Map-Reduce paradigm, which enable fault-tolerant and scalable computations. For Velocity, real-time pipelines with Spark Structured Streaming process fast-moving data, supporting low-latency use cases like IoT and stock price monitoring. To manage Variety, PySpark supports diverse formats (e.g., JSON, Parquet, images) and integrates with sources like S3, Kafka, and Delta Lake, while Veracity is ensured through Delta Lake’s ACID transactions and schema enforcement, maintaining data consistency.

Additionally, Databricks supports DataFrames and SparkSQL for high-level abstractions, allowing efficient querying of structured and semi-structured data. It excels in Pipelines, Data Cleaning, and Data Engineering, enabling seamless workflows for ingestion, transformation, and cleaning, with PySpark automating repetitive tasks. PySpark’s MLlib offers scalable machine learning pipelines for classification, clustering, and recommendations, while GraphFrames enable advanced graph-based analytics like PageRank and community detection. Together, Databricks and PySpark combine distributed processing, real-time capabilities, and advanced analytics, making them indispensable for diverse and dynamic big data workloads.

To demonstrate the importance and capabilities of databricks and PySpark we develop three “projects” (notebooks) that illustrate all those functionalities.

1. A real estate price prediction project that shows an end-to-end machine learning workflow.
2. A graph analysis on yellow taxi movement during the most crowded month (December) in New York City in 2023.
3. Analysis of stock market streaming data using Yahoo Finance.

Real Estate Price Prediction (ML)

In this section of our work, we decided to use a dataset about real estate Portuguese listings to showcase the value of Pyspark and Databricks for Machine Learning problems. This dataset had 114623 rows. Even as this is not considered big data, we believed that it was important to use this dataset to show the importance of this kind of analysis.

To predict Portuguese real estate prices, we began by checking our data, descriptive statistics and some visualizations. From the visualizations we noticed some outliers, like negative values in areas and inconsistencies between variables. After treating the outliers (applying some filters) we did some visualizations and got the following insights:

- The more expensive houses, on average, are the ones that are in **Lisbon** (517 308 €), are of **Estate** type (1 538 885 €), have an energetic certificate of **A** (569 658 €) and have a **lift** (415 272 €).
- The houses with **3 rooms** in total, on average, are more expensive (758 655 €).

We then computed the Pearson correlation between the variables that we consider more important and understood which variables are highly correlated.

Afterwards, we analysed the missing values and decided to drop the columns and rows with more than 50% of nulls, transform Boolean columns into zeros and ones. When a null value represents absence (0) and not missing data, we put them to 0 and create a function to impute values using pipelines and a Random Forest model. Before we imputed the values with this function, we divided our dataset in train in test to ensure that there is no data leakage. We also one-hot-encoded the categorical variables and scaled the columns with the help of pipelines.

With the data ready, we train and tune the hyperparameters using grid search and evaluating using cross validation to have more reliable results on our metric RMSE (root mean squared error), and then results on the test set were the following:

- Gardient-Boosted Trees, RMSE: 259 765 €;
- Random Forest, RMSE: 259 689 €;
- Linear Regression, RMSE: 301 373 €;
- Neural Network, RMSE: 510 064 €.

After analysing the results, we can state that Random Forest is the best model (even though it is missing by 259 689 € the price of the house on average). It is also important to state that the Linear Regression and the Gradient-Boosted Trees predicted negative values for house prices, which doesn't make sense in the real scenario (just poor performing models).

During the development of this notebook we encountered some problems because we were using the community edition (that is free), the main one being that the cluster that was connected would occasionally crash, most probably due to memory overload, that we solved by deleting variables that would not be used anymore and saving variables that are needed to go through several steps in the disk instead of the memory.

Yellow Taxi movement during the most crowded month (December) in NYC in 2023 (Graph)

In this section of our work, we decided to work with taxi trips in New York City during the most crowded month in 2023. Our notebook is composed of two datasets, one where we have the locations id, borough, zone and service zone(taxi_zone_lookup), another dataset which is composed of all taxi trips, information like Pull up location id and drop off location id (yellow_taxis_dec). For us to have the names of the pull up and drop off boroughs, zones and service zones we performed a left join to ensure that all rows from yellow_taxis_dec were retained, and only matching rows from taxi_zone_lookup were added. Our resulting dataset is composed of 3158863 rows that didn't have missing values but had some incoherences that were dealt with using built-in SQL functions.

For us to have a notion of the dataset we were dealing with, we performed some visualizations using functions from SparkSQL and MapReduce (and concluded that using SparkSQL leads to faster results due to its in-memory processing). From the visualizations we arrived to the following:

- People mostly pay with credit cards.
- People usually tip.
- People from Suburban zones with older demographics and people who travel, on average, tip more.
- There's always a small decrease in usage on Sundays and Mondays.
- There’s a clear drop in the week prior to Christmas day, with Christmas day as the day with the lowest usage.

To perform an effective large-scale analysis of the taxi trip data we decided to combine PySpark with GraphFrames. For more interesting insights we decided to perform three analyses, one for the whole month of December, another for Christmas Eve and the final one for New Year’s Eve. To perform GraphFrame in our dataset we needed to define the edges (the trips) and the vertices (the zones).

Our first goal was to identify key hotspots, to do so we applied pageRank, an algorithm used by Google Search to rank web pages from 0 to 10 according to their importance. In our case we concluded that the zones with the most importance are in general high-cost living places and Shopping areas.

Our second goal was to identify locations closely linked using labelPropagation. In our case we noticed that all zones are strongly connected, apart from very specific zones like correctional centers, wildlife parks and cemeteries.

Our third goal was to identify common trip paths, using only the Edges.

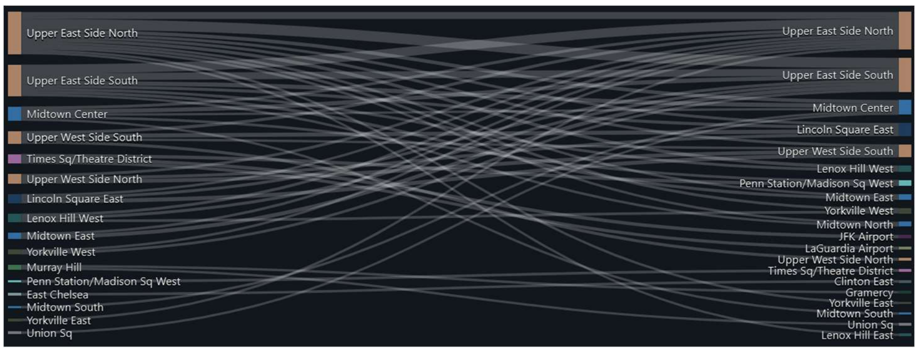

Figure 1 - Common Trip Paths Graph.

From the graph above we can conclude that High-income individuals travel between high-cost living places and shopping areas.

One of the most common functionalities of GraphFrame is the degrees, to identify popular locations. As predicted the High-cost living places, shopping areas and tourist areas are the ones with most edges pointed to them. Using out degrees, we noticed that most transportations hubs appear in the top 10 which makes sense since we are talking about people arriving in NYC.

To study and visualize the structure and dynamics of our borough network we used networkx.

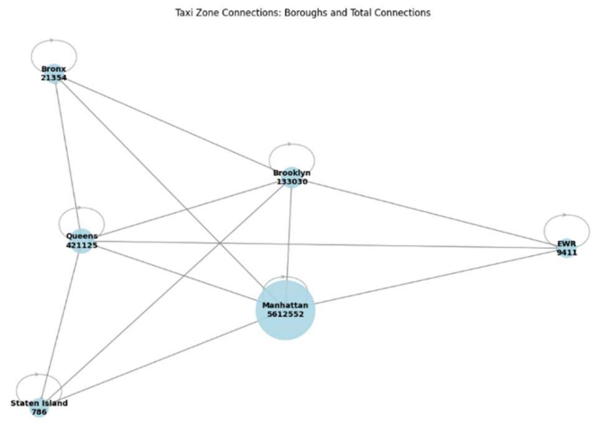

Figure 2 - Boroughs and Total Connections Visualization.

From the visualization above we can conclude that Manhattan is the borough with most connections, Bronx doesn’t have connections with Newark Airport because of its proximity to LaGuardia Airport, Staten Island is not directly connected to the Bronx by the fact that they are very distinct communities and is a big trip between them and Staten Island is not connected to the Newark Airport probably because the people that live there are people with their own means of transportation.

For the GraphFrame of Christmas Eve (24 of December). Using pageRank we noticed that Newark Airport appeared in the top 5 because of Holiday travels. With LabelPropagation we noticed that all zones are strongly connected with exception to locations in Staten Island which have in its majority unique labels, this can be because Staten Island is a family Suburban area with different patterns in the holidays.

Seeing the number of connections between zones we can conclude that High-income individuals travel between high-cost living places probably for private parties and there's a direct path between JFK Airport and tourist areas like Time Square and Theatre District. Using indegree Midtown and Time Square appear higher, probably because of late Christmas shopping and the Rockerfeller center.

The process and conclusions for the New Years Eve were similar to the Christmas Eve with the exception that Midtown appears first using pageRank most likely because of its proximity to Time Square (Ball drop celebration) and the major transportation hubs appear in top 10 using outdegree due to people coming to NYC to celebrate the New Years.

From the analysis the group concluded that the number of taxis needs to be reinforced in the Upper East Side and in tourists and Transportation hubs during the holidays. Apart from that we noticed through taxi paths that we can identify very specific zones with different characteristics and identify patterns within High-income individuals.

Analysis of Stock Market Data from Yahoo Finance (Streaming)

In this project we simulate a streaming process that can acquire, stream, process and analyse stock data using spark streaming.

An arbitrary stock (Microsoft, MSFT symbol) was selected, the data was fetched from yahoo finance (we can define the frequency of the data and how many periods back we want to collect), it contains the timestamp, open, high, low and close of the price, volume, dividends and stock split, we will only use the timestamp and the information of the price (close, high and low) to do our analysis, so we will only save that information and as it is being updated/fetched, the file is updated and the rest of the metrics are being updated. To guarantee that the data that came late after ingestion (when a change is made) is still considered, we created a watermark of 10 min to give that patience of delay.

Now that we have the data and it is in stream (we implement a spark session and read the file as stream), we then process it to have more meaningful metrics. In our case we aggregate the data in 1h time intervals, calculate the price window averages, volatility (standard deviation on that window), and Bollinger Bands (the price plus or minus 2 times the standard deviation of that hour), and outliers will be the price points outside the bands defined above.

We started a streaming query to be able to have a temporary table and analyse the data created using SparkSQL, to see those outliers. Afterwards we can count it, aggregate it by hours and see how many there were per hour; we can also do the same with other stocks and compare them. We didn't include any concrete conclusions about results since they will be updated with new data coming in.

In conclusion, we just mention how we can implement streaming analysis and preprocessing in Databricks using spark streaming.

Conclusion

In conclusion, the projects developed highlight the power of Databricks and PySpark in addressing the 4Vs of Big Data, showcasing their scalability, flexibility, and real-time capabilities. These notebooks demonstrate practical applications in machine learning, graph analysis, streaming, ETL, pipelines and EDA, emphasizing their role in unlocking insights and driving informed decisions across diverse industries.

References

https://www.kaggle.com/datasets/luvathoms/portugal-real-estate-2024?resource=download

https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

https://pypi.org/project/yfinance/