Wall Street Analytics Challenge 2023

Team Name : shaikabdulhadhi

College : B.V Raju Institute Of Technology.

Team Members
1. U.Rakshitha
2. Vijaya Rahul Sivapu
3. Kumara Swamy Inti

Milestone 1

Under Stock Estimation

Approach

The calculation of understock situations in different warehouses is performed by Comparing the products with their safety stock levels

Strategy

If the stock value is less than the the safety value we considered as a flagged situation

Observation

The most of the flagged cases are belong to the the understock condition

Over stock Estimation

Approach

For consideration of over stock detection we considered the complete stock data and added the stock data on the each date of warehouse and compare with the the capacity of the warehouse .

Strategy

If the stock value is more than the warehouse capacity we considered it as a overstock case.

Observations

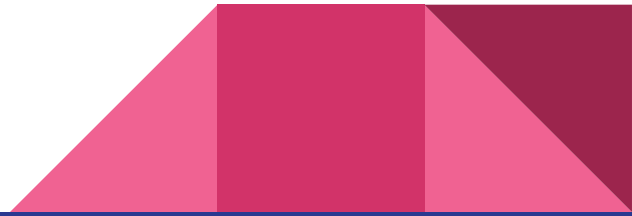
We observed the overstock condition is very less compared to other conditions

Out Of Stock Estimation

Approach: We have considered a situation where a warehouse has 1000 stock. If there are two simultaneous transaction happening where A wants 900 and B wants 1000. But, during the transaction B's transaction fail and the ordered is placed for the A. The output for B would be out of stock, even though there are 100 left over.

Strategy: We have considered the second instances of a warehouse of a particular product and checked if the quantity ordered is less than the current stock.

Observations: There were total 13 instances of out of stock



Ranking of Warehouses.

Approach

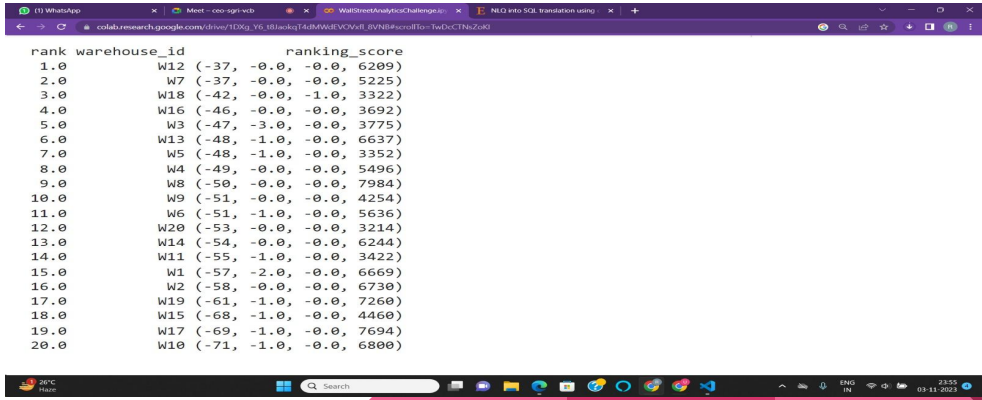
For ranking of warehouse we gone through all flagged cases consideration by adding the understock, overstock and out of stock situations.

The ranking of warehouses is done based on the correlation coefficients. To avoid conflicting ranks"warehouse capacity" is considered as the 4th parameter.

Observation:

W12 is the best performing warehouse.

W10 is the worst performing warehouse.



rank	warehouse_id	ranking_score
1.0	W12	(-37, -0.0, -0.0, 6209)
2.0	W7	(-37, -0.0, -0.0, 5225)
3.0	W18	(-42, -0.0, -1.0, 3322)
4.0	W16	(-46, -0.0, -0.0, 3692)
5.0	W3	(-47, -3.0, -0.0, 3775)
6.0	W13	(-48, -1.0, -0.0, 6637)
7.0	W5	(-48, -1.0, -0.0, 3352)
8.0	W4	(-49, -0.0, -0.0, 5496)
9.0	W8	(-50, -0.0, -0.0, 7984)
10.0	W9	(-51, -0.0, -0.0, 4254)
11.0	W6	(-51, -1.0, -0.0, 5636)
12.0	W20	(-53, -0.0, -0.0, 3214)
13.0	W14	(-54, -0.0, -0.0, 6244)
14.0	W11	(-55, -1.0, -0.0, 3422)
15.0	W1	(-57, -2.0, -0.0, 6669)
16.0	W2	(-58, -0.0, -0.0, 6730)
17.0	W19	(-61, -1.0, -0.0, 7260)
18.0	W15	(-68, -1.0, -0.0, 4460)
19.0	W17	(-69, -1.0, -0.0, 7694)
20.0	W10	(-71, -1.0, -0.0, 6800)

Visualizations

We produce a line plot for each product in each warehouse showing it’s stock levels and quantity ordered over a period of 2 years.

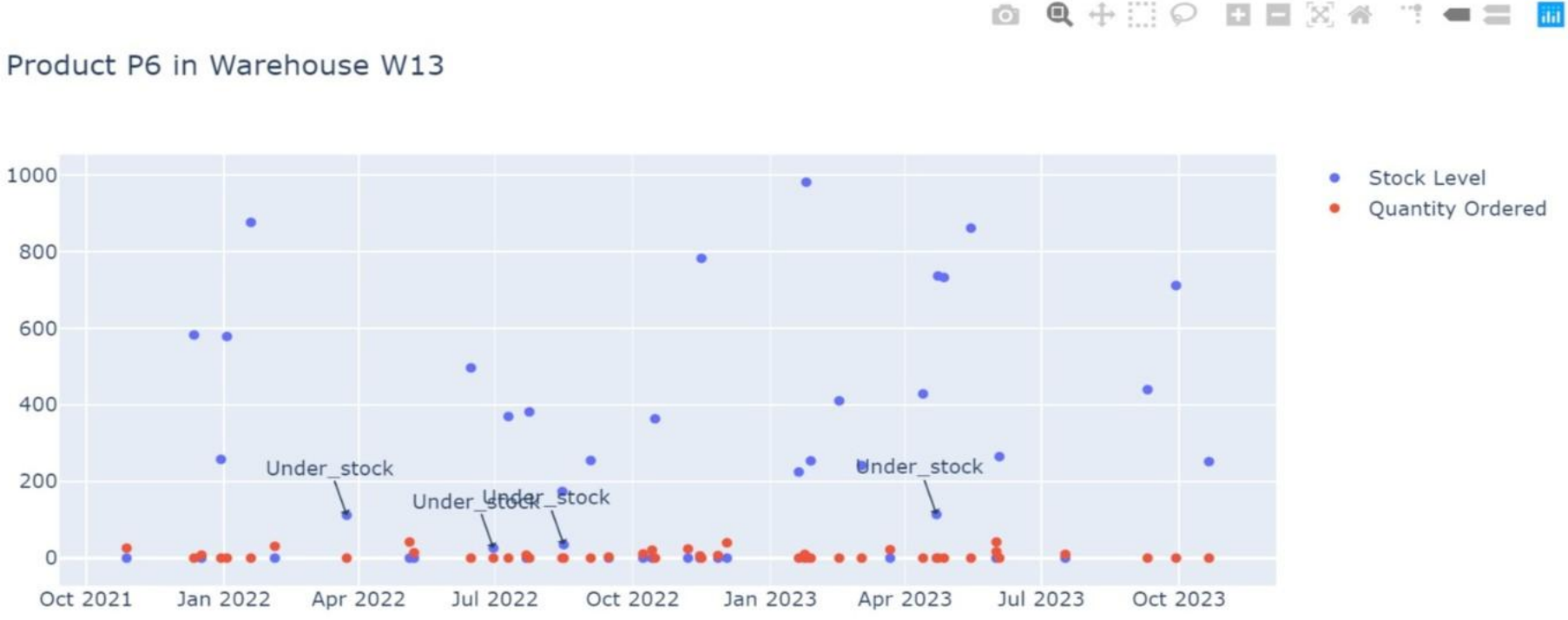
We highlighted 3 conditions.

They are under stock, out of stock and over stock.

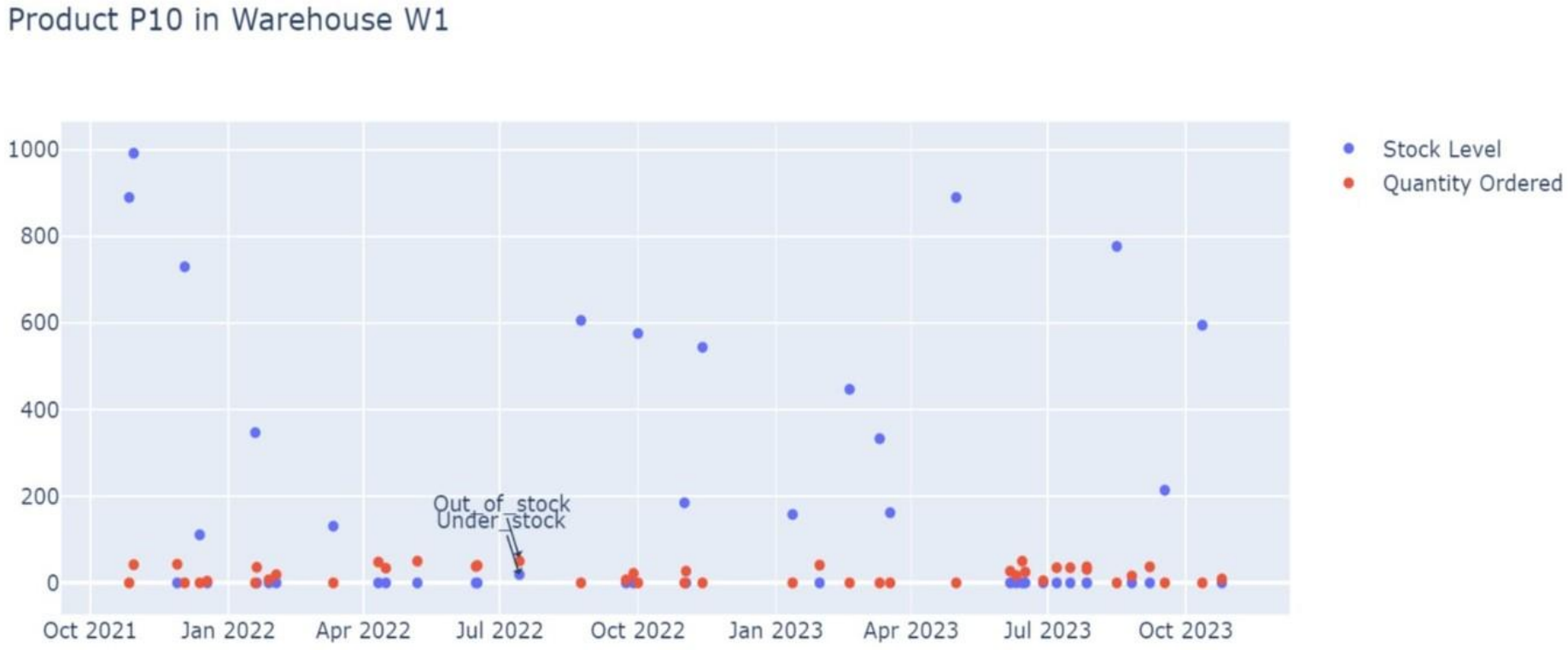
We used **plotly** library for visualizing the line plot.

We have created 2 dropdowns for the user to select a specific product in a specific warehouse so that, the specific product visualization is generated.

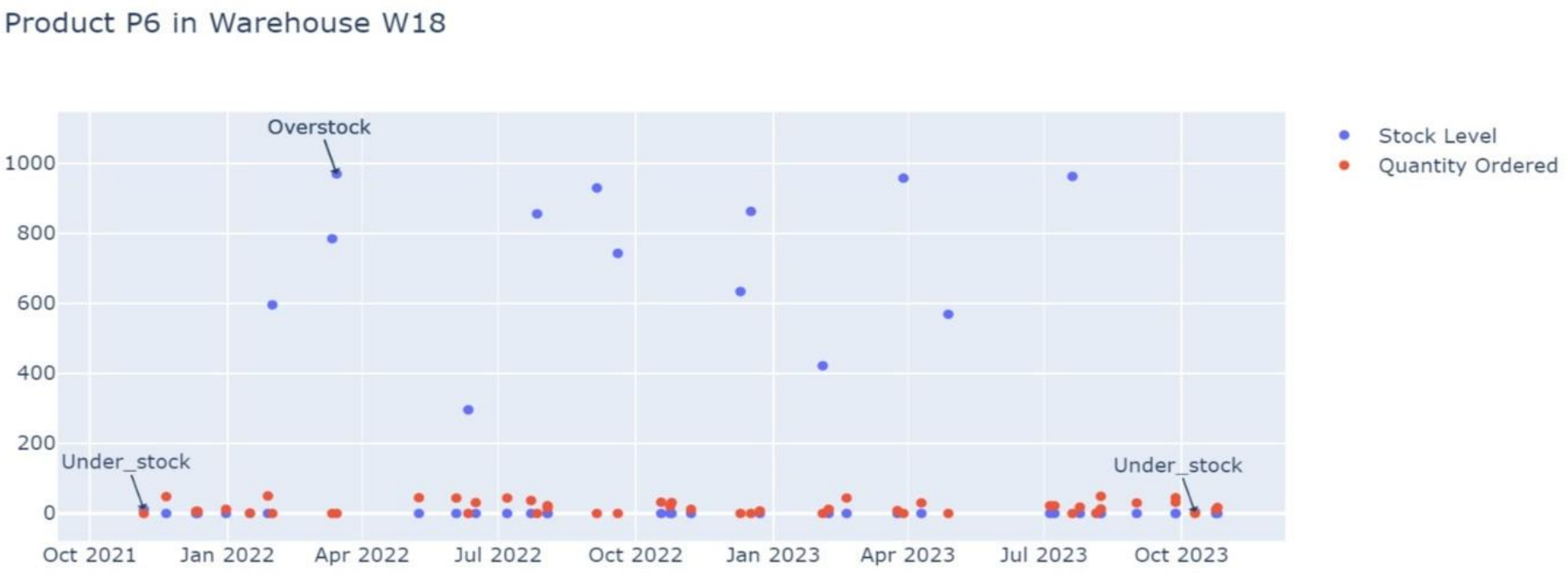
Under_Stock


Product P6 in Warehouse W13
1000
800
600
400
200
0
Under_stock
Under_stock
Under_stock
Under_stock
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Apr 2023
Jul 2023
Oct 2023
Stock Level
Quantity Ordered

Out_Of_Stock


Product P10 in Warehouse W1
1000
800
600
400
200
0
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Apr 2023
Jul 2023
Oct 2023
Out_of_stock
Under_stock
Stock Level
Quantity Ordered

Over_Stock


Product P6 in Warehouse W18
Overstock
Under_stock
Under_stock
1000
800
600
400
200
0
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Oct 2022
Jan 2023
Apr 2023
Jul 2023
Oct 2023
Stock Level
Quantity Ordered

MileStone 2

In the first step where we have to convert the natural language query into a SQL query we have used the python library “spacy”.

We have used SingleStoreDB as our database.

In order to vectorize the dataset we have used Postman.

In the Postman, we have first set the the method as POST and then paste the url from the OpenAI.

The model we choose for the conversion is text-embedding-ada-002.

We then set it in JSON mode and give the model and input, and we send the request.

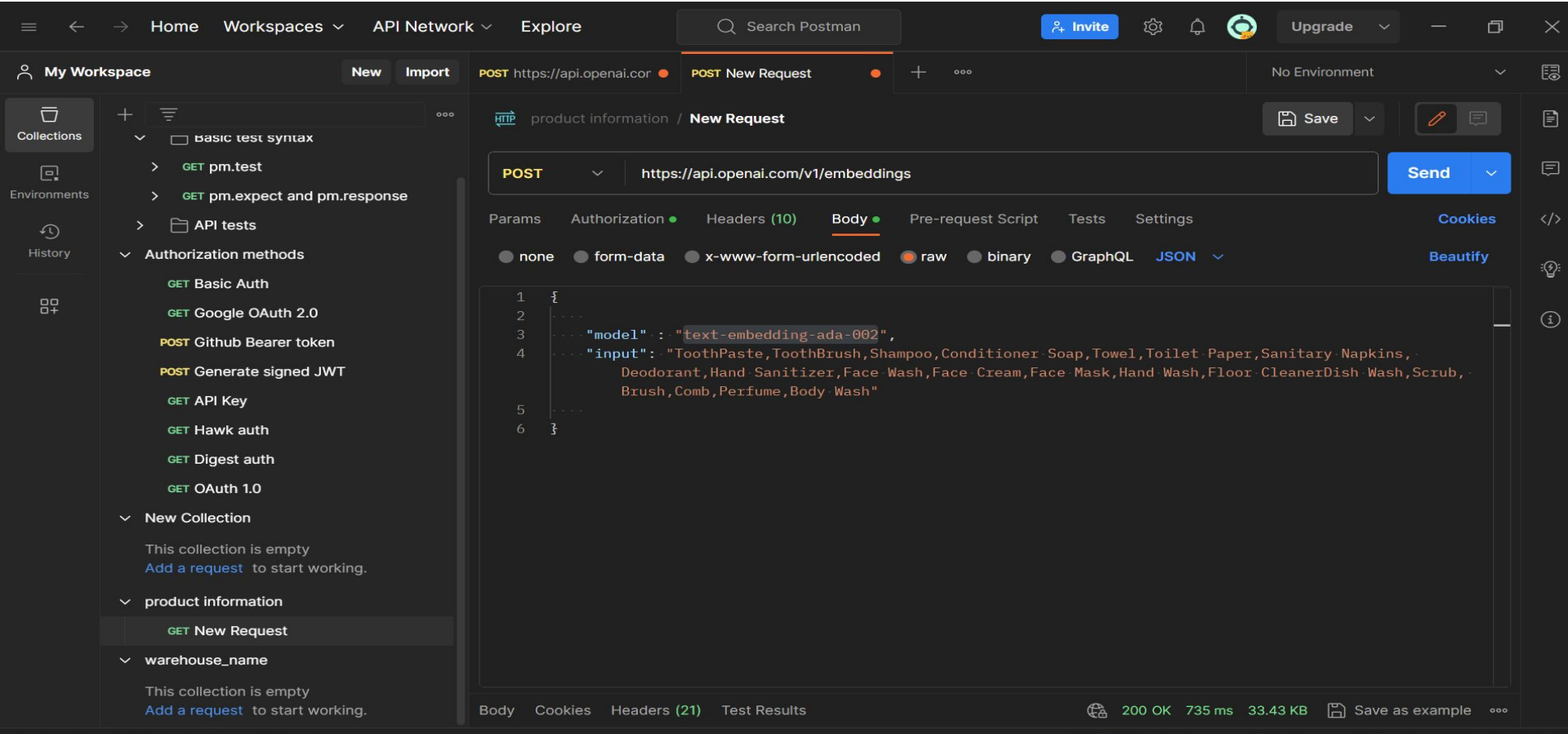
And based on the request we get the vector dataset.

For entering the queries we have used the streamlit library with the Langchain LLM.

For fine-tuning the model, where we can see a significant change in the model we need atleast 50-100 dataset.

The

General setup of Postman



This is the vectorized data