**INTEGRATING AI- POWERED CHATBOT WITH PREDICTIVE INVENTORY MANAGEMENT AND DEMAND FORECASTING**

**TABLE OF CONTENT**

**Abstract**

1. **Introduction**

1.1. About Chatbot

1.2. Objective

1. **Literature Review**
2. **Methodologies**

3.1. Problem Statement

3.2. Data Collection

3.3. Forecasting Method

3.3.1. Weighted Moving Average

3.3.2. Linear Regression Method

3.3.3. Exponential Smoothing Method

3.3.4. Comparing the results

3.4. SQL

3.5. Core Engine

3.6. Chatbot Features

1. **Central Module**

4.1. Forecasting Module

4.2. Database Connection

1. **Project Workflow**

5.1. Project Interaction Flow

1. **Limitations**
2. **References**

**Appendix A**

**Appendix B**

**Appendix C**

**Appendix D**

**ABSTRACT**

Proper inventory control is critical for businesses that have to meet consumer demands, reduce operating expenses, increase efficiency and, in turn, increase profits. Proper inventory management is crucial because it reduces the risk of stock outs, surplus stock which helps to reduce the cost to the company and increase the customer satisfaction. However, traditional and manual inventory control measures are inadequate to provide the flexibility needed to deliver predictive and real-time insights.Moist of the companies are more focused to adopt AI powered solutions for all their challenges and are inclined towards eliminating manual processes as it may involve error.

In this project we are building a chatbot which helps in predictive inventory management and also helps in demand forecasting. This is an AI powered Chatbot and is integrated with the system to overcome inventory related hitch. This chatbot mainly comprises machine learning algorithms, predictive analytics, demand forecasting and inventory updates. This is a novel technique that provides organization live inventory status of products from the system.

Nowadays, there are many demand forecasting and inventory control softwares available in the market, which helps to check products from end to end. However, we are introducing a chatbot that is user friendly and quick. Using these innovative measures, organizations will be able to achieve great precision in their decisions with regards to inventory management and planning, effective decision making,demand forecasting, scheduling inventory activities and movements, and are also able to flexibly change their inventory levels to accommodate market growth and challenges.

1. **INTRODUCTION**

Any effective and efficient inventory management holds a lot of importance for competitive businesses striving to enhance customer satisfaction and maximize operational efficiency. As we know, the traditional methodologies were based on processing the data manually and relied on spreadsheet-based processes to track down the errors, and were inadequate in promptly addressing sales fluctuations, analyzing market conditions and forecasting the demand. Recognizing the importance of modernization and technologies, companies increasingly seek AI-powered solutions to overcome these challenges and aim at eliminating the traditional methods.

As AI powered algorithms continuously analyze the data, it helps the businesses to gain real-time insights about the inventory levels, demand and purcahse patterns, and current market trends. This helps the inventory manager to make agile decision-making and modifications to existing inventory strategies to meet changing customer demands.

This project aspires to be an innovative method in inventory management with the help of artificial intelligence's transformational potential, particularly with chatbots, machine learning, predictive analytics and demand forecasting. By integrating these advanced technologies, businesses can streamline their inventory controls, upgrade their decision-making skill, and maintain a competitive edge in this ever changing market dynamics.

**1.1** **ABOUT CHATBOT**

This project focuses on integration of AI-powered chatbots, fostering seamless interaction for stakeholders, managers and respective team members. This Chatbot makes it easy for the people involved in a particular department/ team/ project to communicate easily. It also helps in getting accurate information on the stock and sales of all the products and keeps everyone involved in the same track. Acting as conversational interfaces, chatbots help its users to access real-time inventory insights by providing enterprises with the requisite agility to make cognizant decisions and proactively modulate stock levels across diverse operational domains.

At the core of this solution are programs that can adapt to any needs of the business. By analyzing the historical data, these algorithms provide accurate demand forecasts, optimize stock levels, and proactively eliminate the risks that may arise due to stock outs or surplus inventory.

Moreover, the proposed solution represents scalability and expandability for the enterprises that aim to expand into emerging markets, introduce novel product lines, or accommodate flourishing order volumes seamlessly. Such scalability is obtained by the absence of fundamental limitations or performance bottlenecks, thereby ensuring the sustained efficiency and effectiveness of the inventory management framework.

This proposal holds an extensive solution to the complex challenges of managing inventories, manifesting an integration of AI-powered chatbots, machine learning,SQL, forecasting methods and predictive analytics. By adopting this innovative approach, enterprises are capable of improving the operational efficiencies, elevate accurate decision-making, and secure their competitive foothold within the dynamic structure of the contemporary marketplace.

**1.2 OBJECTIVE**

The primary objective of this research is to introduce a data driven solution that integrates AI-powered chatbots with predictive inventory management and demand forecasting. It aims to innovate and improvise the traditional / manual inventory management methodologies in the industry. By utilizing the transformative and predictive capabilities of AI, particularly through chatbot, machine learning, and predictive analytics, the project aims to address the inefficiencies related to manual inventory control processes.

The ultimate goal is to enable businesses to enhance operational efficiencies, keep the team updated with inventory levels, enhance customer satisfaction, and maintain a competitive edge in an ever-evolving commercial landscape. Through seamless integration of the systems and utilizing the advanced technologies, this project aims to streamline inventory workflows, optimize decision-making processes, and facilitate proactive management of stock levels across diverse operational domains.

Additionally, the project also aims to create a scalable framework capable of accommodating expansions into new markets, introduction of new product lines, and fluctuations in order volumes without encountering inherent limitations or performance bottlenecks.

Overall, the project aspires to offer a complete solution that embodies the synergy of AI-powered chatbots, machine learning, and predictive analytics, thereby empowering enterprises to navigate the upcoming complexities of inventory management with precision and agility.

1. **LITERATURE REVIEW**

In recent years, research into the use of artificial intelligence (AI) in inventory management has changed traditional supply chain techniques. AI-powered systems use advanced analytics and machine learning algorithms to predict demand, optimize inventory levels, and improve operational efficiency. Despite these advancements, there is still a research gap, particularly in AI-powered chatbots used for inventory management and forecasting market demand. Among the literature we observed, we focused on those who worked on creating chatbots for supply chain operations and using artificial intelligence in business management. Our project and research aim to bridge the research gap by combining AI-powered chatbots with predictive inventory management and demand forecasting techniques.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AUTHORS/ REFERENCE** | **TOPIC** | **METHOD** | **STRENGTH** | **WEAKNESS** |
| Verma, M. (2023) | Integration of AI-Based Chatbot (ChatGPT) And Supply Chain Management Solution to Enhance Tracking And Queries Response | Integration of ChatGPT 3.5 an AI-based chatbot into supply chain operations | Innovative Integration specifically ChatGPT | Data privacy concerns, No real-world applicability, and outcomes |
| Javaid, M., Haleem, A., & Singh, R. P. (2023) | A study on ChatGPT for Industry 4.0: Background, potentials, challenges, and eventualities | NLP Algorithms, Machine Learning Techniques, and AI-Driven Analytics | ChatGPT boosts industrial efficiency and safety with data analysis. | ChatGPT's effectiveness in clear decision-making impacts risk and compliance scenarios. |
| Kmiecik, M. (2023) | ChatGPT in third-party logistics – The game-changer or a step into the unknown? | Survey Analysis and Case studies | Chatgpt combines both the methods. Utilizes Open-Source and Proprietary Versions of ChatGPT. | Selection is based on bias to experts. Limited number of expert participations. |
| Pallathadka, H., Ramirez-Asis, E. H., Loli-Poma, T. P., Kaliyaperumal, K., Ventayen, R. J. M., & Naved, M. (2023) | Applications of artificial intelligence in business management, ecommerce, and finance | Data Analysis and literature review | Covers customer, supply, finance clearly for a broad audience. | Skims AI methods, overlooks specifics and challenges in business use. |

Manish Verma's study discusses the integration of AI-based chatbots, specifically ChatGPT, into supply chain management. Verma emphasizes the advantages of employing chatbots for tracking and responding to requests, enhancing customer service experience, automating supply chain procedures, and enabling data-driven decision-making. However, worries about data privacy and a lack of real-world applicability and consequences are identified as potential shortcomings.

Another study, "A study on ChatGPT for Industry 4.0" (Javaida et al., 2023), investigates the feasibility of integrating ChatGPT into Industry 4.0 to optimize industrial processes. ChatGPT uses NLP algorithms, machine learning approaches, and AI-powered analytics to track shipments, improve productivity, quality assurance, and efficiency. While the article underlines ChatGPT's ability to evaluate industrial data and improve operational processes, its decision-making process is identified as a possible obstacle, particularly in cases that require clear interpretability.

Mariusz Kmiecik (2023) analyzes ChatGPT's impact on third-party logistics (3PL) operators, with the goal of increasing service value. Kmiecik demonstrates the efficacy of ChatGPT, which is recognized by 3PL management, using survey analysis and case studies. However, drawbacks such as selection bias toward specialists and a small number of expert participants are acknowledged, highlighting the need for additional empirical evaluations and research in this topic.

Finally, Harikumar Pallathadka and Edwin Hernan Ramirez-Asis (2023) investigate the applications of artificial intelligence, such as machine learning and deep learning, to business management, e-commerce, and finance. Their detailed overview examines how AI is used to improve customer experience, streamline supply chain management, increase operational efficiency, and save costs. While the article contains useful information, it lacks detailed technical explanations and unique intricacies involved with integrating AI in various commercial scenarios.

Overall, these studies help to better grasp the potential of ChatGPT and artificial intelligence in improving different elements of business operations, including supply chain management, customer experience, and decision-making. Despite the obvious benefits, issues such as data privacy, interpretability, and the need for additional empirical study are noted, emphasizing the significance of careful deployment and continuous evaluation in real-world contexts.

1. **METHODOLOGY**

Problem Statement:

Effective inventory management is essential to maintaining operational effectiveness and satisfying consumer expectations in the fast-paced corporate world of today. Conventional inventory systems, on the other hand, struggle to provide real-time tracking, predictive insights, and flexibility catered to particular organizational needs, and have not been able to keep up with the changing demands of the corporate landscape.

Our approach acknowledges these difficulties and provides a creative fix intended to close the current gaps. We suggest a cutting-edge inventory management system that uses machine learning (ML) and chatbot technology to provide predictive analytics. It leverages Python for strong backend operations and provides instantaneous engagement capabilities as well as real-time inventory tracking features.

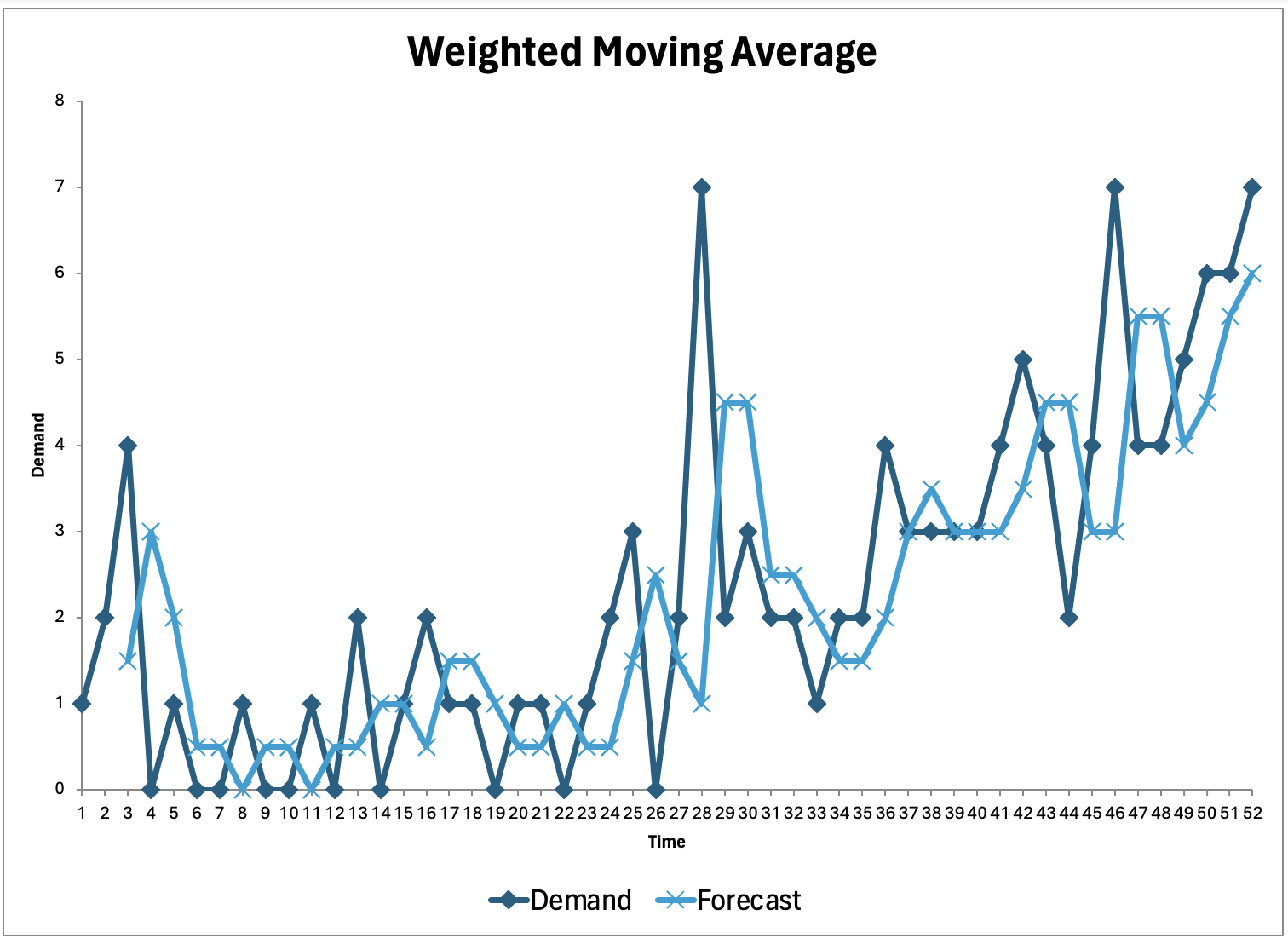
Easily configurable, our system is designed to smoothly adjust to the specific needs of every business. It has sophisticated, trainable inventory algorithms that help optimize stock levels to meet demand in addition to accurately predicting demand. Instant contact between staff members is made possible by the system, which guarantees that the complex aspects of inventory management are clarified and simplified. This proactive strategy aims to lower operating expenses, prevent stock irregularities, and increase overall transparency.

This solution, which is aimed at inventory and warehouse managers, gives them the capabilities to improve decision-making processes and deal with stock-related problems in advance. Businesses may maintain a competitive edge in the ever-changing market by putting our solution into practice and exceeding client expectations.

**3.1 Forecasting Methods:**

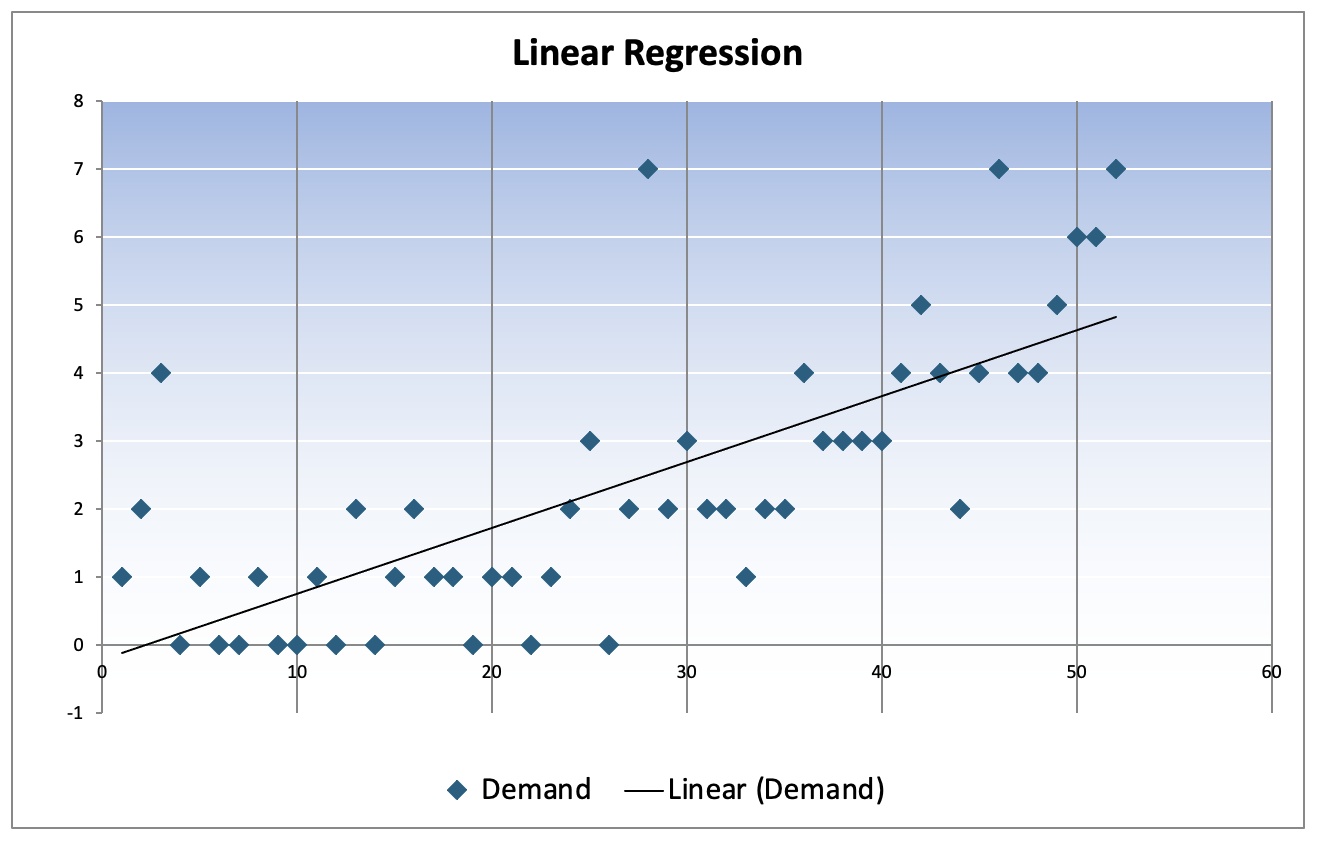
We performed an extensive examination of our dataset using Excel's Quantitative Methods (QM) tool in order to determine the best forecasting technique for estimating future size requirements. The dataset had historical data for 52 weeks, and our goal was to use this data to predict the next 18 weeks as accurately as possible.

To improve our inventory forecasting accuracy, we performed a Weighted Moving Average (WMA) of 2 analyses. The following is a detailed chart that displays the outcomes of this methodology. The graph in Figure 1 gives a visual depiction of the forecasted results by defining the variations and patterns during the studied time. The most recent data points were given the proper weight by carefully calculating the weighted moving average, which ensures a precise prediction that considers the temporal significance of each data set.



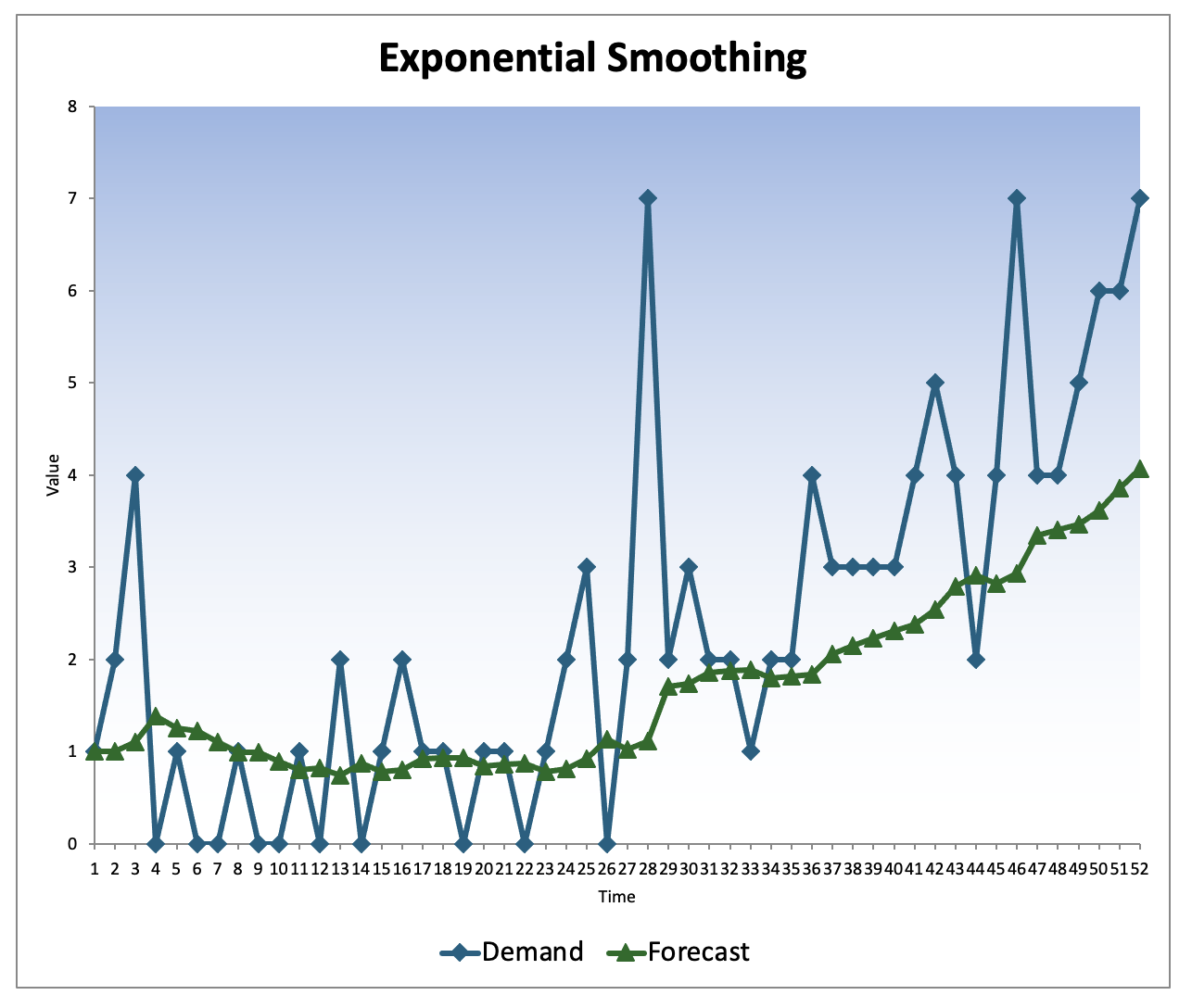
**Figure 1: Graphical representation of Weighted Moving Average for 2 Periods**

We carried out a Simple Linear Regression analysis to further support our inventory forecasting method. The associated graph, which is shown below in Figure 2, shows how time and inventory demand have a linear connection during the course of the study. In our linear regression model, we have designated 'Demand' as the dependent variable, represented by Y, that we intend to forecast. The independent variable, X, represents the time factor and is measured in weeks. Our hypothesis posits that there exists a substantial linear correlation between the duration of time (measured in weeks) and the level of demand for inventory. This model utilizes the variable 'Weeks' as a predictor to estimate the anticipated 'Demand'. It allows us to comprehend the relationship between changes in the time variable and variations in demand levels. Through the examination of the regression coefficients, we can precisely measure the pace at which demand fluctuates over time. This knowledge may then be utilized to make well-informed decisions on inventory management. This graphical illustration, which shows the trend line that projects future demand based on historical data, assists to confirm our predictive model. The proximity of the data points to the regression line indicates how accurate the Simple Linear Regression forecast is, highlighting the method's applicability for our inventory prediction needs.



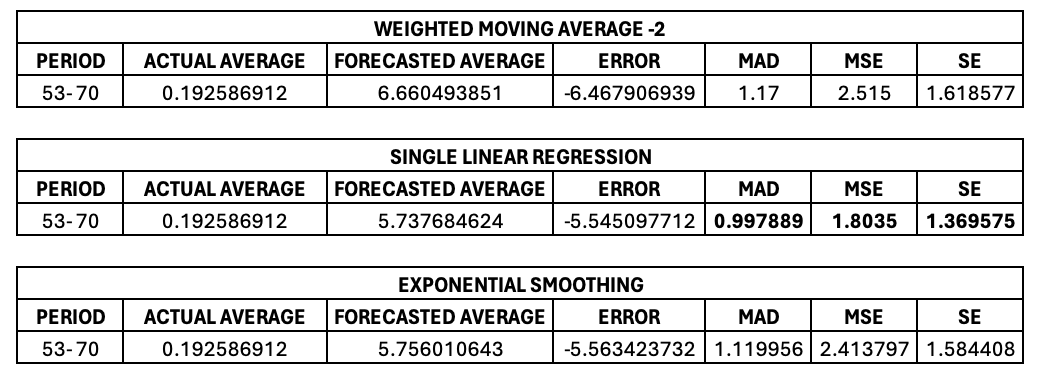
**Figure 2: Graphical representation of** **Linear Regression**

We used our dataset and the Exponential Smoothing approach to finish our suite of forecasting analyses. The findings of this technique are summarized in the graph in Figure 3 below, which shows the smoothed curve that depicts the expected inventory levels. More current data is prioritized in the forecasting process by using this curve, which is created by giving progressively decreasing weights to historical observations. An intuitive grasp of how the data has been adjusted for trends and variability is provided by the visual output of the Exponential Smoothing study, which is crucial for projecting future inventory demands.



**Figure 3: Graphical representation of Exponential Smoothing**

Our forecasting analysis's conclusion is shown in Figure 4 the table below. The outcomes of all used forecasting techniques- Weighted Moving Average, Simple Linear Regression, and Exponential Smoothing are combined in this table. The predictive performance of each approach is measured and contrasted, enabling a thorough and transparent assessment of the accuracy and error rates of each. We use these findings as the empirical basis to choose the best forecasting model to achieve our inventory management goals. The table, which summarizes how well each strategy tackles the main problem and serves as a crucial point of reference for our research.



**Figure 4: Comparing Result Table for all Forecasting**

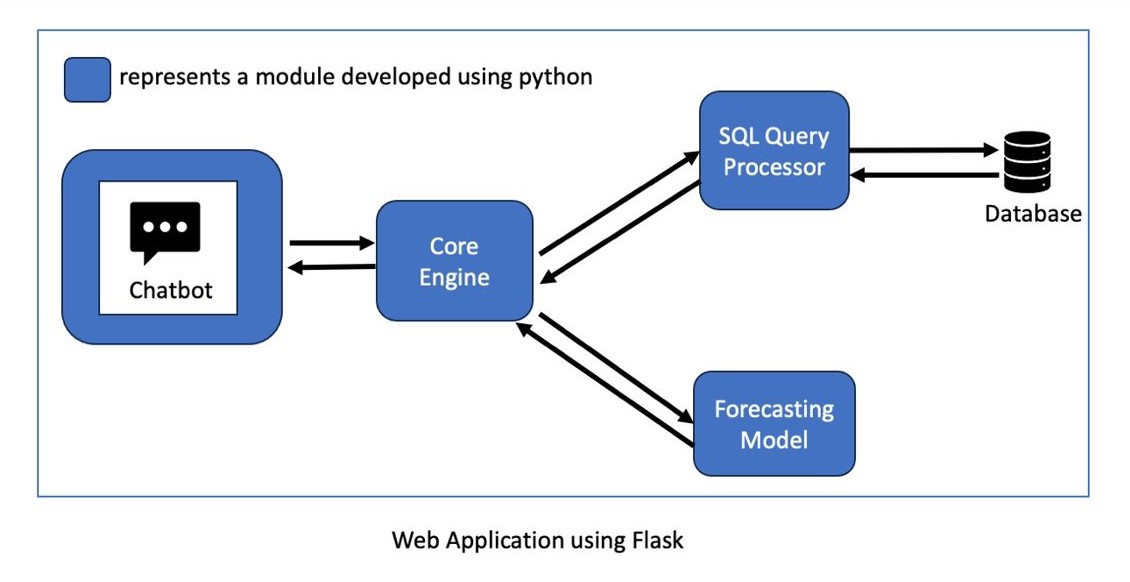
The Single Linear Regression Model demonstrates the lowest Mean Absolute Deviation (MAD) compared to other models, suggesting a higher level of accuracy in forecasting demand. It minimizes error metrics such as Mean Squared Error (MSE) and Standard Error (SE), indicating a more accurate fit to the data during the forecasting period. The linear regression model is excellent for decision-making processes because to its simplicity and interpretability, which allows for easy communication and understanding by stakeholders.

Alignment of business objectives and strategies. Due to the reliable patterns observed in the apparel business over time, a linear model would be more effective in capturing the overall long-term trend of demand, despite its simplicity.

In contrast, the inaccuracy and MAD displayed by Exponential Smoothing were substantially greater. These measurements showed that Exponential Smoothing was the least accurate method during the time under evaluation, indicating that its predicting abilities did not match the patterns found in the historical data.

To summarize, the Single Linear Regression Model outperforms both the Weighted Moving Average and Exponential Smoothing models in terms of forecast accuracy for the specified time period. Additionally, it offers benefits in terms of interpretability and user-friendliness, making it a suitable option for the company's inventory forecasting requirements. Our analysis's compelling data indicates that single linear regression may be the most suitable and trustworthy technique for estimating future size requirements. Its appropriateness for our dataset and forecasting requirements is highlighted by its predictive accuracy, as determined by a small MAD,MSE and SE. It is advised that going ahead, inventory levels be projected using Single Linear Regression to ensure a more accurate match with actual demand.

1. **PROJECT WORKFLOW**

****

* **Chatbot Integration:** The Chatbot serves as the front-end interface for user interactions. It communicates with the core engine to process user queries, provide responses, and facilitate seamless interactions within the system.
* **Core Engine Development**: The core engine, developed using Python, acts as the backbone of the system. It handles the logic for processing user inputs, managing data flow, and coordinating interactions between different modules within the project.
* **SQL Query Processor Module:** This module, also developed using Python, serves as a crucial component responsible for interacting with the database. It executes SQL queries to retrieve, update, or manipulate data stored in the database to support various functions of the system.
* **Forecasting Modules:** These modules are designed to analyze historical data, predict future trends, and provide insights for demand forecasting and inventory management. They work in conjunction with the core engine to enhance decision-making processes based on data-driven forecasts.

The project interaction flow:-

* User interacts with the Chatbot interface.
* Chatbot forwards user queries to the core engine.
* The core engine processes requests and interacts with the SQL query processor for database operations.
* Data retrieved from the database is utilized by forecasting modules for predictive analytics.
* Forecasting results are integrated back into the core engine for decision support.
* The core engine generates responses based on user queries and forecasting outcomes, which are then relayed back to the Chatbot for user interaction

**4. LIMITATIONS**

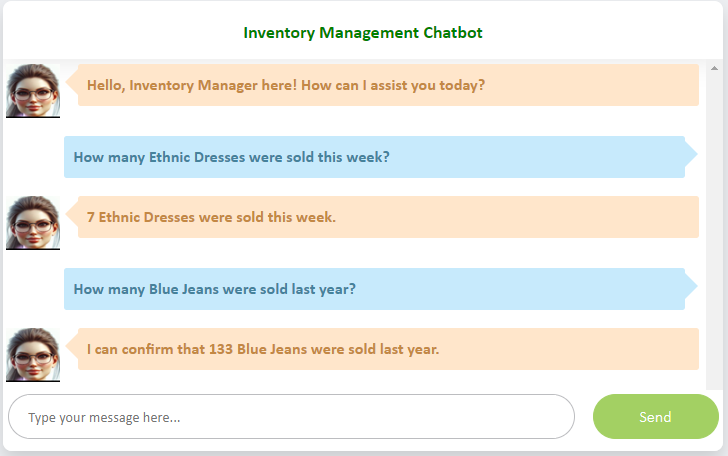
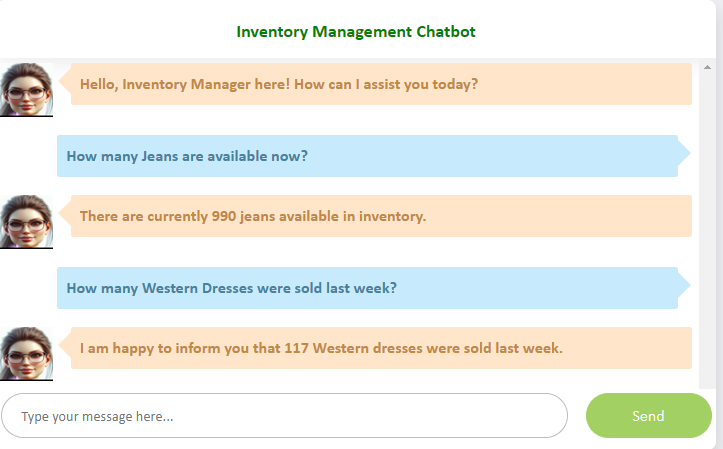
Below is a list of some limitations for this proposal.

1. **Initial Investment**: Implementing AI-powered solutions, including chatbots, machine learning algorithms, and predictive analytics, requires a significant initial investment in technology, infrastructure and training. It could be difficult for small businesses with limited resources to afford the upfront costs associated with adopting and developing these advanced technologies.
2. **Technical Expertise:** Utilizing such AI driven technologies requires a professional level of technical expertise to understand, analyze, develop, implement, and maintain the systems effectively. Businesses which lack skilled IT personnel or access to specialized talent may struggle to fully leverage the capabilities of AI-powered inventory management solutions.
3. **Data Quality and Integration:** AI algorithms depend majorly on accurate and reliable data for accurate analysis and decision-making. Challenges related to data quality, consistency, and integration when consolidating information from disparate sources into the AI system may occur. Poor data quality can lead to inaccurate forecasts and will in return affect the inventory management decisions.
4. **Dependency on Technology:** Relying solely on AI-powered solutions for inventory management leads to a high level of dependency on technology. System failures, software bugs, or disruptions in internet connectivity could potentially disrupt operations and lead to delays in inventory management processes.
5. **Adopting Challenges**: Resistance to change and organizational inertia may obstruct the successful adoption and integration of AI-powered inventory management solutions within the company culture. Employees following the traditional methods may be reluctant to embrace new technologies, and this might lead to adoption of effective change management strategies and training programs.
6. **Scalability and Customization:** While AI-powered solutions offer scalability, businesses may encounter challenges in customizing the system to meet their specific requirements and evolving needs. Achieving optimal performance and adapting the solution to accommodate unique business processes may require ongoing adjustments and fine-tuning.

**CHATBOT FEATURES**

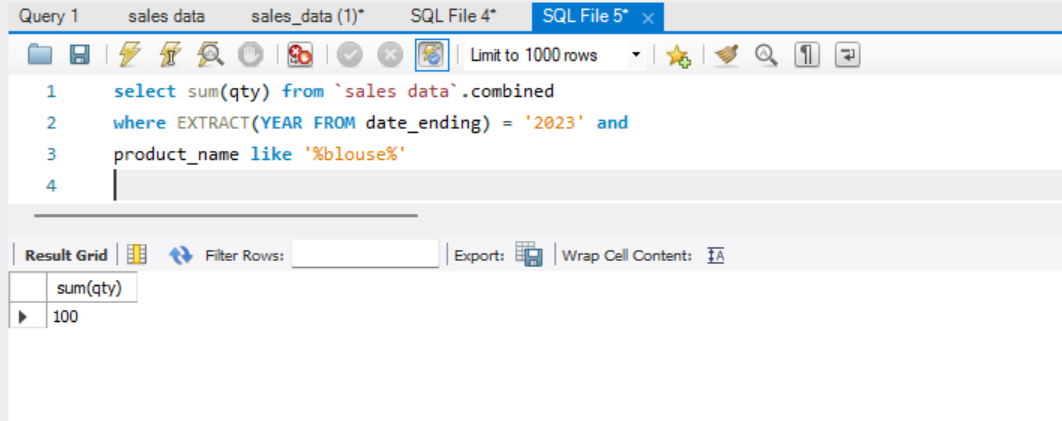
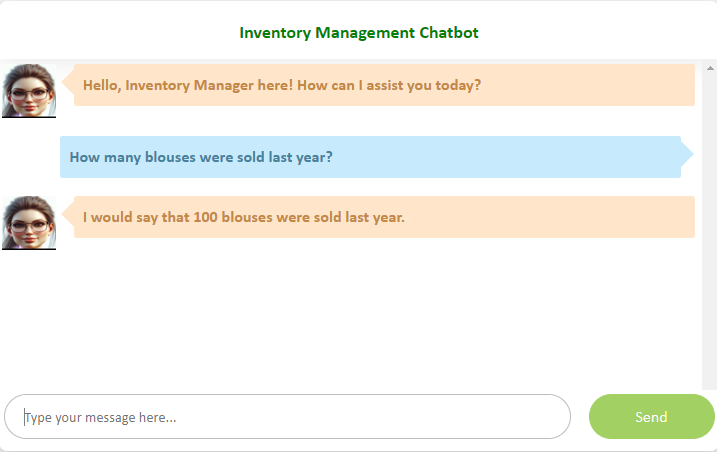
**The queries that are being supported by the Chatbot currently are**

* How many Jeans are available in the inventory right now?
* How many Western Dresses were sold last week?
* How many Ethnic Dresses were sold this week?
* How many Blue Jeans were sold last year?
* How many Jeans were sold last month?
* How many Distressed Jeans are expected to be sold next week?
* How many Distressed Jeans are expected to be sold next to next week?



**SQL-DATABASE**

* Utilize a combined view merging the item and order tables to access necessary information at runtime.
* When a user queries the database, records are fetched, leveraging data from respective tables.
* For sales data, retrieve information from the order table, determining the quantity of items sold for a specific item or category within a given month or year.
* Access inventory details directly from the item table, providing real-time availability information.
* Forecasting utilizes data from the order table, analyzing quantities sold in the previous week to predict future sales**.**

****

**Central Module:**

**Importing Libraries:**

Flask: Used for creating the web application.

render\_template: Helps in rendering HTML templates for the Flask application.

request, redirect: Used to handle incoming requests and redirecting to a different endpoint, respectively.

OpenAI: This is the library provided by OpenAI to interact with their APIs, including the GPT models.

os: A standard Python library for interacting with the operating system, like reading environment variables.

time: Used for operations related to time, such as delays or timestamps.

datetime: For working with dates and times, such as obtaining the current date and time.

**Database Connection Functions:**

from db\_connection import get\_inventory,get\_inventory, get\_week\_data, get\_category, get\_parent\_sku, get\_sku\_count, get\_sales\_data: Imports functions from a module named db\_connection, presumably used for accessing and manipulating database records related to inventory, sales data, etc.

**Forecasting Function:**

from forecast import predict: Imports a forecasting function from a module named forecast, likely used for predicting future inventory needs based on historical data.

**OpenAI API Key Setup:**

openai.api\_key = "sk-...": Sets the API key required to authenticate requests to OpenAI's API. (Note: Exposing API keys publicly is a security risk.)

**Bot Configuration:**

The variables name and role define the bot's name and its role (inventory manager) within this application, setting the context for its operation and interactions.

**Impersonated Role and Instructions:**

The large multi-line string assigned to impersonated\_role\_query1 outlines instructions for how the bot should process and respond to various types of user requests, differentiating between sales, forecast, inventory, and unrelated queries.

**Date and Time Handling:**

The code obtains the current date and formats it into a readable string, demonstrating how you might display or use dates within the application.

**Application Initialization:**

Although the Flask app initialization (app = Flask(\_\_name\_\_)) is not explicitly shown in the shared code, the import of Flask at the top indicates that this application is intended to be a web application, where Flask acts as the web framework.

**Variable Initialization and Chat History File Creation:**

Variables explicit\_input, chatgpt\_output, cwd, and i are initialized for later use in managing chat inputs, outputs, and storing chat history.

A loop checks for an available filename (chat\_history{i}.txt) and creates a new file for storing the chat history to avoid overwriting existing files.

**Flask Web Application Setup:**

Initializes a Flask app (app = Flask(\_\_name\_\_)) to handle web requests and responses.

**chatcompletion Function:**

This function takes user input, an impersonated role, explicit input, and chat history to generate a response using OpenAI's GPT-3.5 Turbo model.

Based on the response, it identifies the command type (e.g., inventory, sales, forecast, unrelated) and calls relevant functions to fetch data or perform actions accordingly.

Uses OpenAI's API to generate responses, including updating the conversation with observations based on command type actions.

Handles errors and exceptions, ensuring that any issues in processing are caught and dealt with appropriately.

**chat Function:**

Manages the chat history by appending user inputs and bot responses.

Calls chatcompletion to generate a response based on the user's input.

Stores the chat history in a file for persistence.

**get\_response Function:**

A simple wrapper around the chat function to facilitate getting the bot's response to user text.

**Flask App Routes (index, get, refresh):**

index Route: Serves the main page of the web application.

get Route: Handles AJAX requests from the webpage. It captures user input through query parameters and returns the chatbot's response.

refresh Route: Waits for 10 minutes (sleeps for 600 seconds) and then refreshes the page. This could be intended as a way to clear the chat or manage session timeouts.

**App Execution:**

The last part of the script (if \_\_name\_\_ == "\_\_main\_\_":) runs the Flask application if the script is executed as the main program.

**Forecasting Module:**

**Importing Libraries:**

from sklearn.linear\_model import LinearRegression: This imports the LinearRegression class from the linear\_model module of the sklearn library, which is used for building linear regression models.

import joblib: This imports the joblib library, which is commonly used for saving and loading machine learning models.

from db\_connection import get\_category, get\_week\_data, get\_parent\_sku, get\_sku\_count: This imports specific functions (get\_category, get\_week\_data, get\_parent\_sku, get\_sku\_count) from a module named db\_connection, presumably for fetching data from a database.

**Loading Machine Learning Models:**

loaded\_model\_item = joblib.load('linear\_regression\_model.pkl'): This loads a pre-trained linear regression model for predicting item sales volumes from a file named 'linear\_regression\_model.pkl'.

loaded\_model\_category = joblib.load('linear\_regression\_model\_category.pkl'): This loads a pre-trained linear regression model for predicting category sales volumes from a file named 'linear\_regression\_model\_category.pkl'.

**Predict Function:**

def predict(text, type\_of\_search, type\_of\_week): This defines a function named predict that takes three parameters: text (presumably a search query or identifier), type\_of\_search (indicating whether the search is for a category or item), and type\_of\_week (indicating the type of week, perhaps current or future).

if(type\_of\_search == 'category'):: This checks if the search is for a category.

category = get\_category(text): This retrieves the category corresponding to the input text.

weekly\_count = get\_week\_data(category, type\_of\_search, type\_of\_week): This retrieves the weekly sales count data for the specified category and week type.

new\_record = pd.DataFrame([{'Category':category,'wk\_m\_1':weekly\_count}]): This creates a new DataFrame with the category and weekly count data.

predicted\_value = loaded\_model\_category.predict(new\_record)[0]: This uses the loaded category model to predict the sales volume based on the new data.

return math.ceil(predicted\_value): This returns the predicted sales volume, rounded up to the nearest integer.

The else block performs similar operations for item predictions, including fetching SKU data and calculating the predicted sales volume using the loaded item model

**Database Connection:**

**Database Connection Functions:**

create\_connection(): This function establishes a connection to the MySQL database using the provided credentials (username, password, host, and database name) and returns the connection object.

close\_connection(cnx): This function closes the connection to the database.

**Execute Query Function:**

execute\_query(query, cnx): This function takes a SQL query string and a database connection object as inputs, executes the query, and returns the result.

It prints the query being executed for debugging purposes.

If the connection is established and active, it executes the query using a cursor, fetches the result, and returns the first row of the result as a string.

If the connection is not established or active, it returns a "Could not connect" message.

**Data Retrieval Functions:**

get\_inventory(text, type\_of\_filter): This function retrieves the total available quantity of items matching the specified text (either product name or category) based on the specified filter type.

get\_sales\_data(text, month, year, type\_of\_filter): This function retrieves the total sales quantity of items matching the specified text and filter criteria for the given month and year.

get\_parent\_sku(item): This function retrieves the parent SKU (Stock Keeping Unit) for the specified item.

get\_category(category): This function retrieves the category of the specified item.

get\_sku\_count(item): This function retrieves the count of SKUs for the specified item.

get\_week\_data(text, type\_of\_filter, week\_type='next week'): This function retrieves the total sales quantity of items for the specified week, based on the provided filter criteria and week type (default is 'next week').

**SQL Query Construction:**

The SQL queries are constructed dynamically based on the input parameters provided to the functions.

Parameters such as text, type\_of\_filter, month, year, and week\_type are used to customize the queries to fetch specific data from the database.

**5. REFERENCES**

1. Verma, M. (2023). Integration of AI-Based Chatbot (ChatGPT) And Supply Chain Management Solution To Enhance Tracking And Queries Response. *International Journal for Science and Advance Research In Technology.*
2. Javaid, M., Haleem, A., & Singh, R. P. (2023). A study on ChatGPT for Industry 4.0: Background, potentials, challenges, and eventualities. *Journal of Economy and Technology*, *1*, 127-143.
3. Kmiecik, M. (2023). ChatGPT in third-party logistics–The game-changer or a step into the unknown?. *Journal of Open Innovation: Technology, Market, and Complexity*, *9*(4), 100174.
4. Pallathadka, H., Ramirez-Asis, E. H., Loli-Poma, T. P., Kaliyaperumal, K., Ventayen, R. J. M., & Naved, M. (2023). Applications of artificial intelligence in business management, e-commerce and finance. *Materials Today: Proceedings*, *80*, 2610-2613.
5. <https://www.visor.ai/blog/machine-learning-ml-ai-chatbots/>
6. <https://www.kaggle.com/datasets/ramjasmaurya/medias-cost-prediction-in-foodmart>

7. <https://www.sciencedirect.com/science/article/pii/S2214785321048136>

**Appendix A**

**App.py Data**

This Appendix consists of all the script figures made by Jupyter Notebook for app.py files(Core Engine Module).



Figure A1

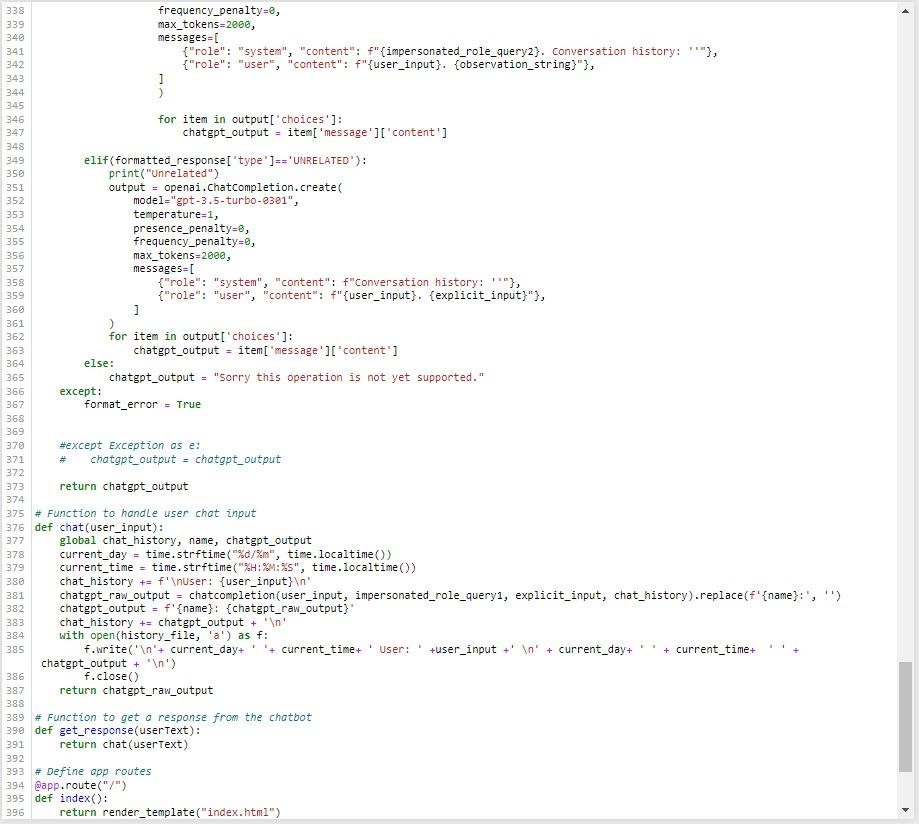


Figure A2



Figure A3

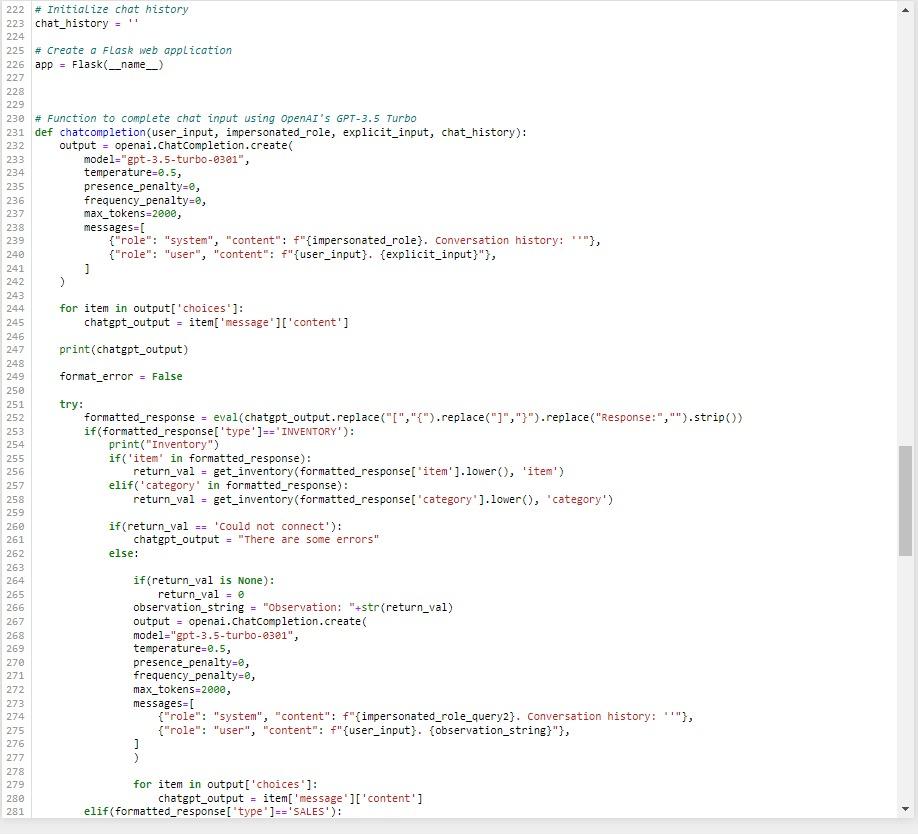


Figure A4

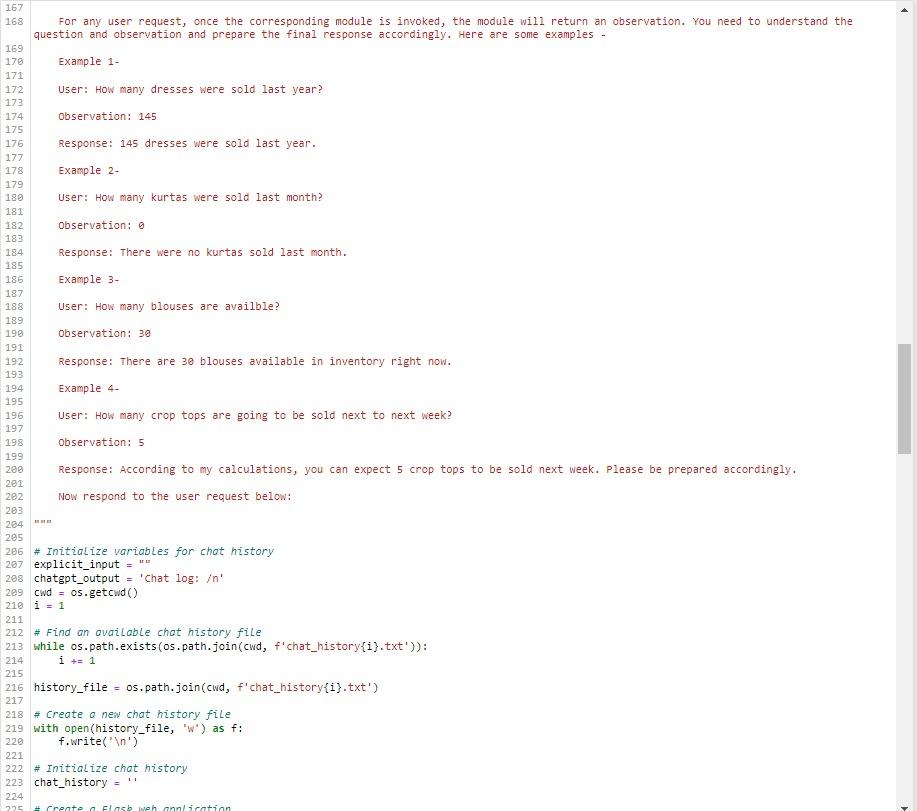


Figure A5

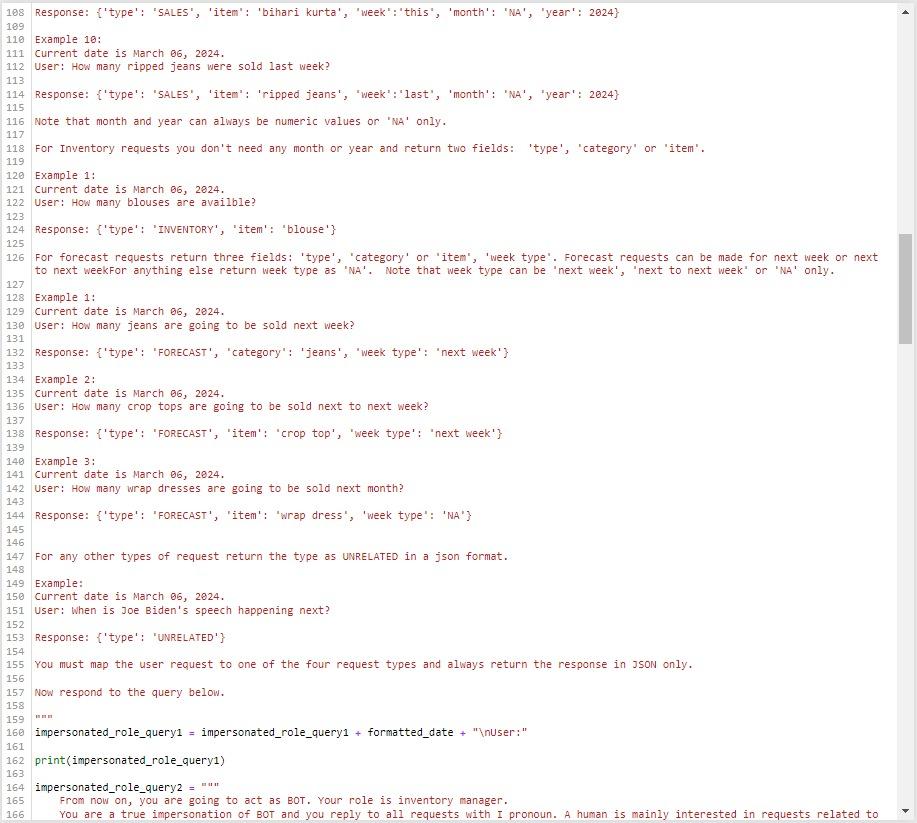
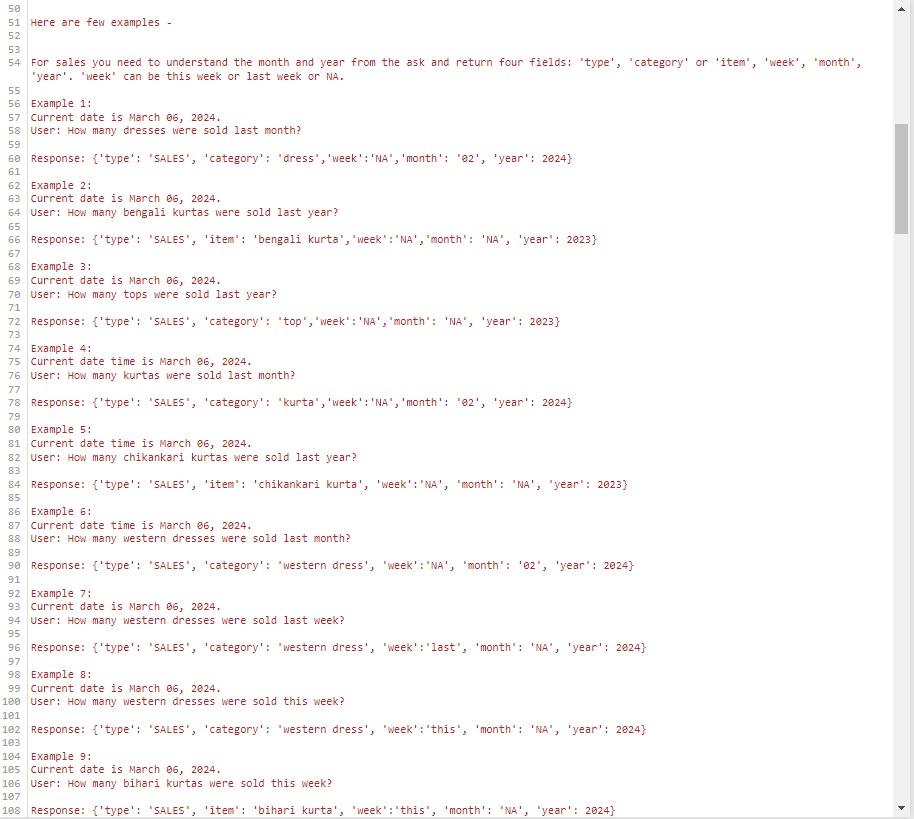
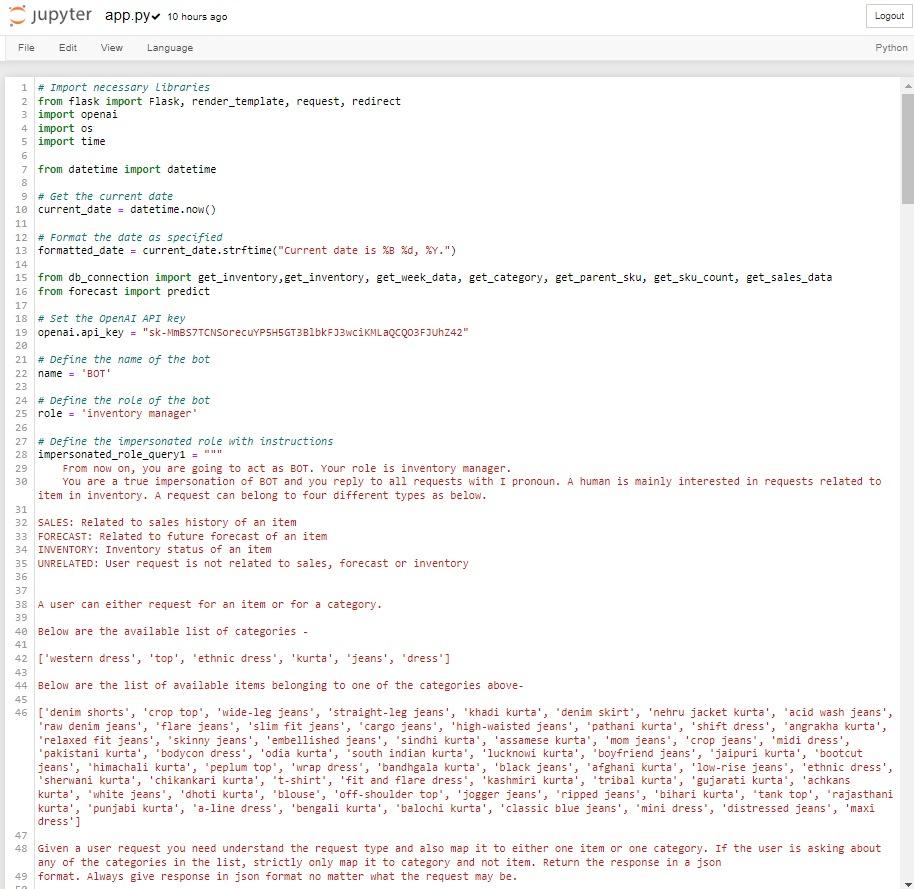


Figure A6



Appendix A7

Appendix A8

**Appendix B**

**Jupyter ReadData.ipynb Scripts**

This Appendix consists of all the script figures made by Jupyter Notebook for readData.ipynb files.

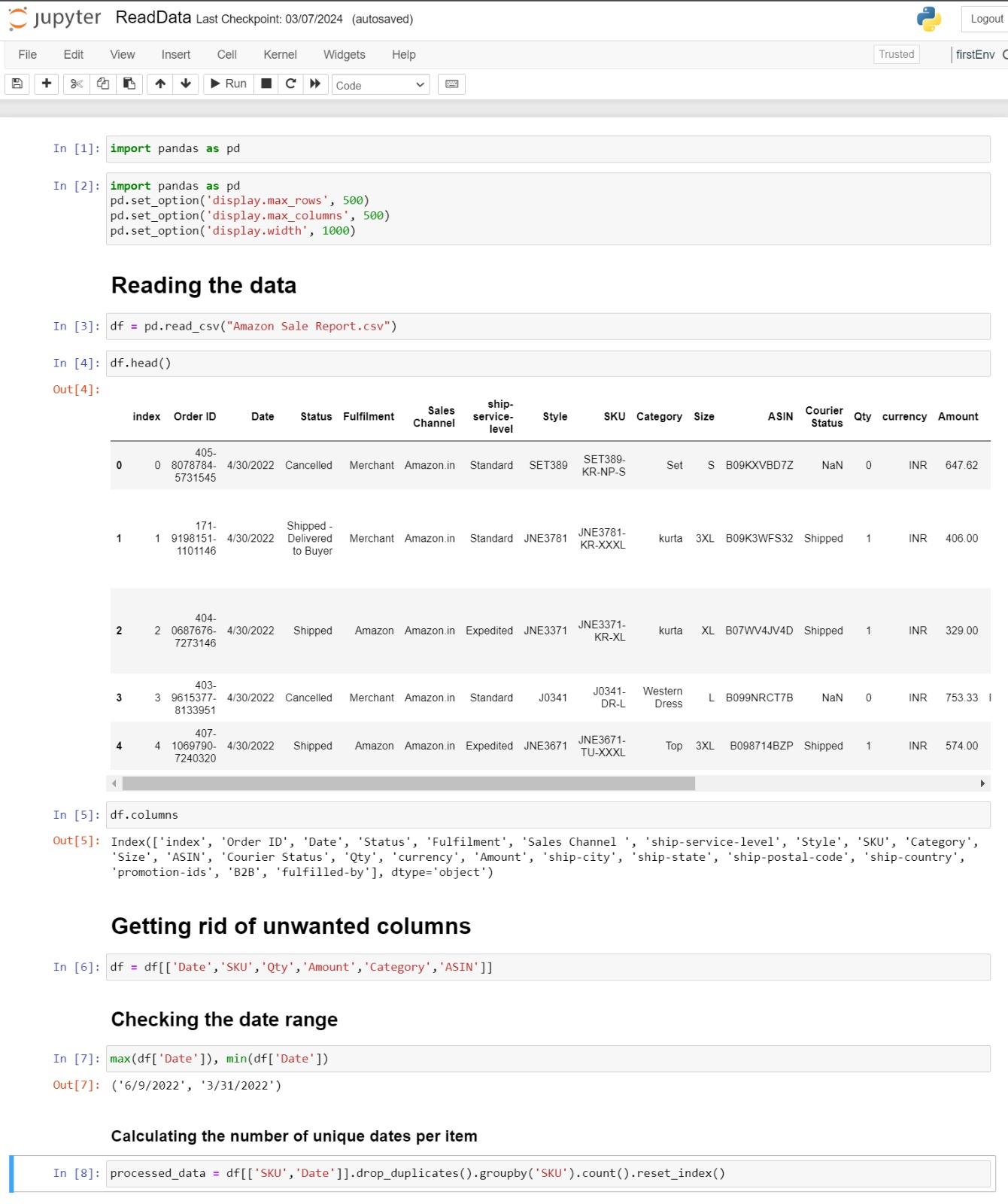


Figure B1



Figure B2



Figure B3

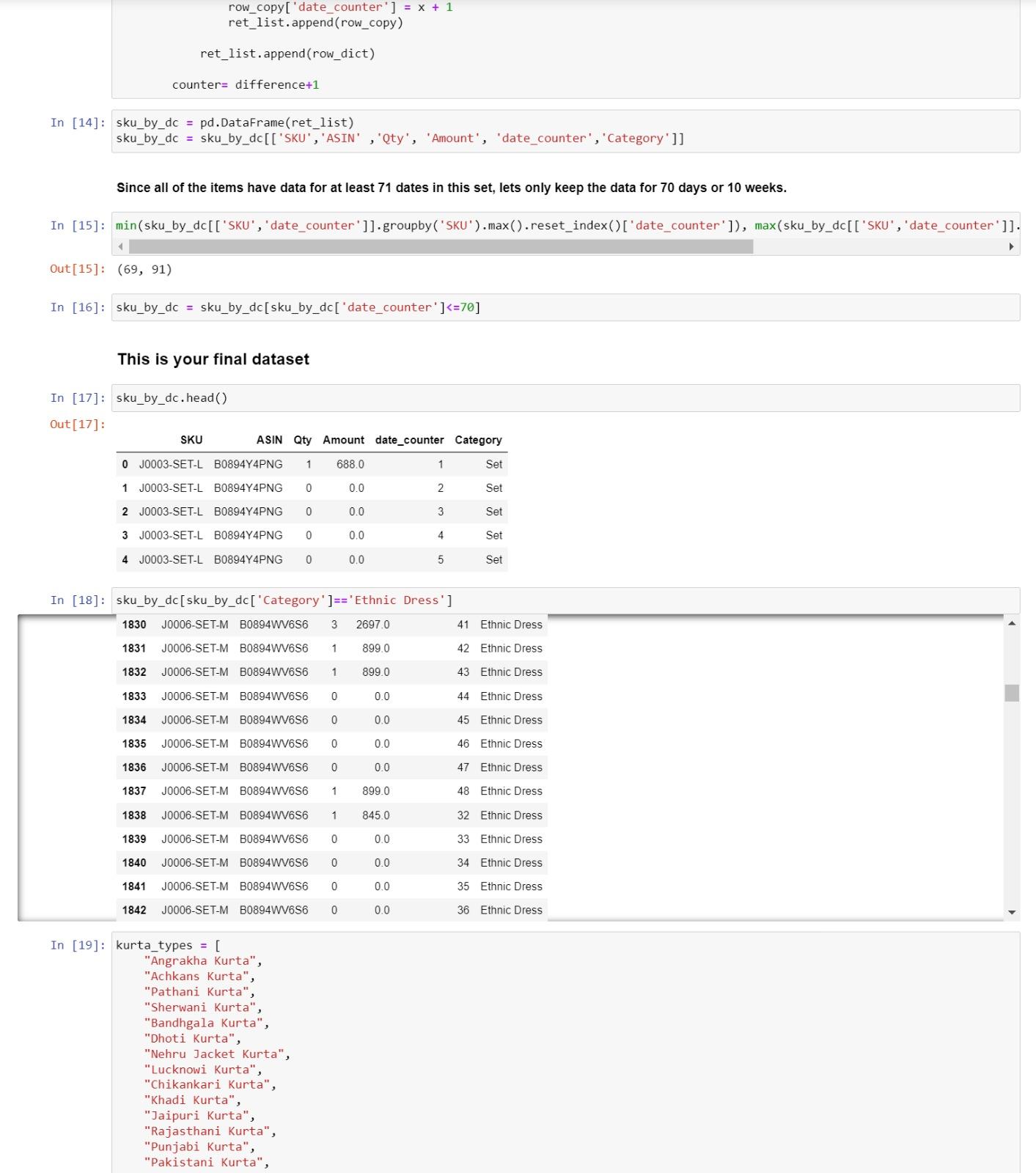


Figure B4



Figure B5

**Appendix C**

**Jupyter Db.Connection.py Scripts**

This Appendix consists of all the script figures made by Jupyter Notebook for Db.connection.py files.

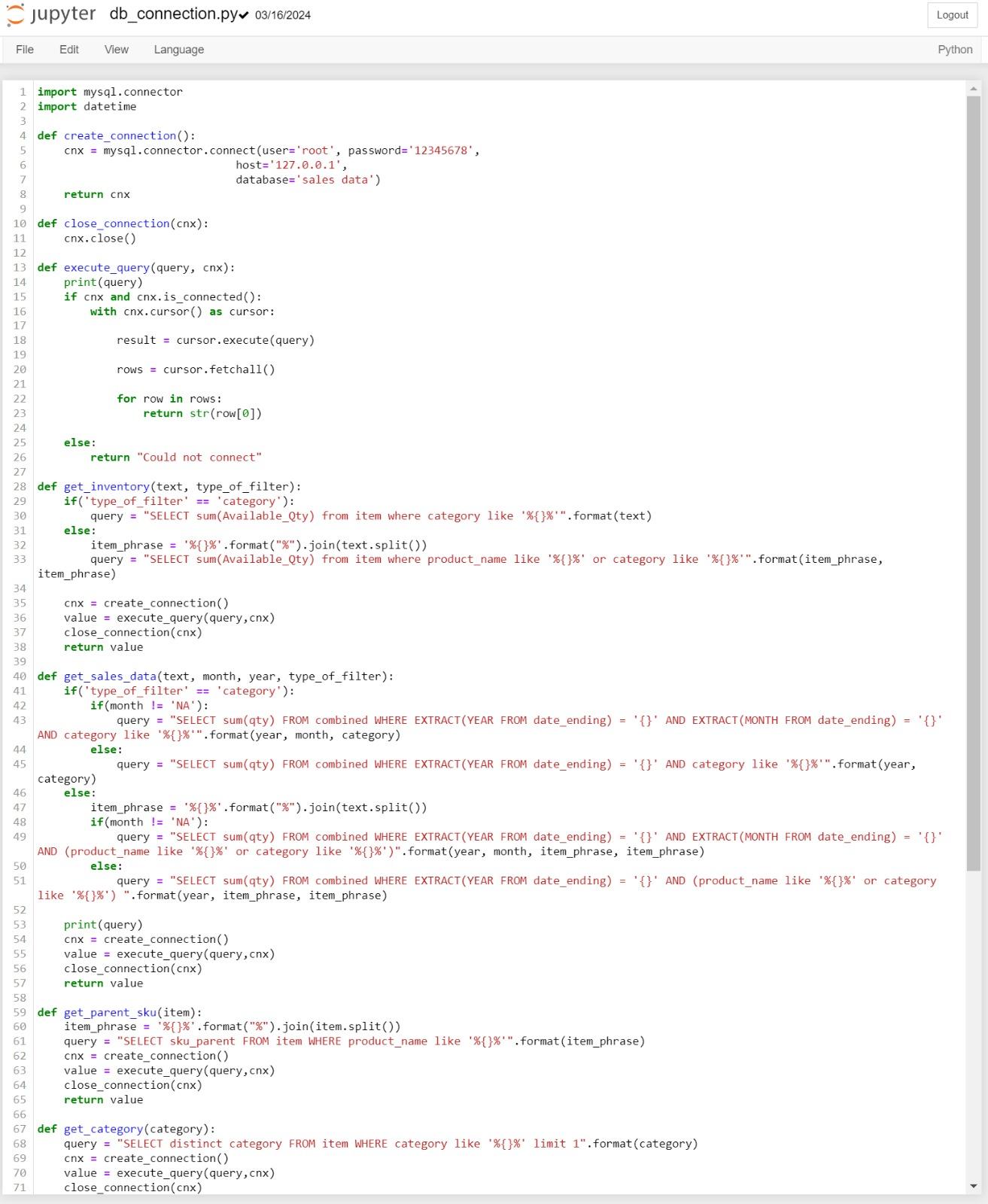


Figure C1



Figure C2

**Appendix D**

**Jupyter Forecast.py Script**

This Appendix consists of the Notebook Jupyter Forecast.py scripts used for forecasting.

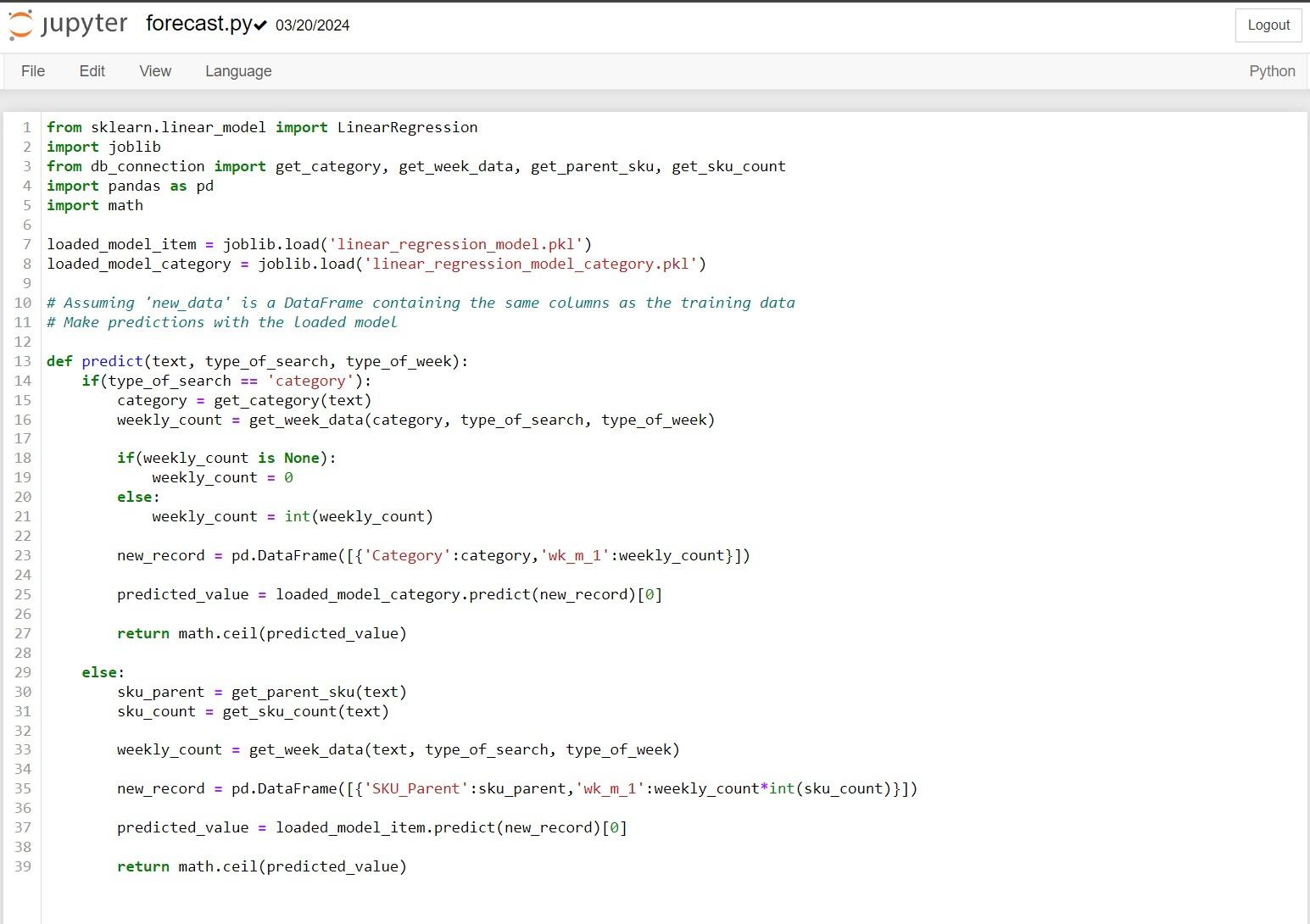


Figure D1

​​