Project Report

Caroline Zhou,

Nathan Sehn,

David Yu

CPSC 571: Design and Implementation of Data Systems

Dr.Alhajj

December 22, 2020

Abstract

Data mining and time series analyses are considered some of the most powerful technologies available to help companies in any industry to extract meaningful information from their existing database for the purpose of maximizing profit. With a great inventory system, it can help the managers by providing different types of information such as products that are usually bought together, future selling pattern of each product and automated warnings if a product is out of stock or low in stock. Based on our knowledge and research of frequent data mining and time series analyses, FP-Growth and ARIMA are commonly used with great performance on large dataset.

INTRODUCTION

Problem Definition

An excellent inventory system optimizes space, deals with inaccurate estimation or changes that might affect the estimation, and considers the inventory needs during different times of the year. Stock management and estimation can be a tedious, repetitive, and time-consuming job for any individual, therefore, it increases the possibility for a person to make mistakes. Inaccurate inventory estimation may lead to unavailability of a product which may in turn affect a customer's or a client's experience with the company, and that can negatively impact the company's revenue. In the past, companies usually relied on customer relationships to collect inventory related data to predict demand of products so they could improve their sales. Nowadays, most companies use data mining techniques to help them predict future inventory needs based on the information that they have gathered throughout the years.

With tracks and records, companies can learn from the data about what their consumers' interest and demands are, and they can use this information to make sure to have the correct inventory in stock in order to satisfy their consumers' needs. Data mining techniques are used to reduce forecasting errors and improve customer satisfaction by increasing product availability, and this allows the companies to stay competitive in the market and maximize their profit.

Motivation

Poor inventory management may cause the company to lose money, and in severe cases, it can affect a patient's health condition. In order to illustrate the severity of the mismanagement, two cases will be presented to demonstrate the inventory errors. The first case is the incident that Tesla had to pause its production line, because one of their suppliers had not been able to deliver the set of USB cables that is an essential component of their product (AI, 2019). The second case illustrates the struggle of hospitals maintaining patient safety levels and minimizing the total inventory cost, because many hospitals only have limited space and budget. However, the companies who run these hospitals still need to consider the shortages of administered drugs, because a shortage of a required drug may affect their patients' lives (Kritchanchai & Meesamut, 2015). Kritchanchai and Meesamut (2015) also concluded from the study that the characteristics of item categories needs to be considered while the supply and demand economic model is unique to each industry. Based on the aforementioned cases, it is crucial to practice and develop data mining skill set with a focus on a specific inventory dataset in order to better understand the needs of that industry. Therefore, this project will focus on finding the best algorithm with the given dataset in order to develop a system that best suits its needs. There is probably a need to use multiple algorithms for different types of estimation and prediction tasks. All these methods

are aimed to provide the companies with required information to plan for downsizing or removal of an item with low selling patterns, because these items may negatively affect the sales of high ranked items.

Proposed Solution

To develop an application with a user-friendly interface in order to help users such as store managers to manage the stock of the store by providing different analyses and predictions based on a set of store data. The analyses will determine products frequently bought together, customers with similar purchasing behaviors, future sale trends of each item, and automated warnings for items that are low in stock or out of stock. For the ease of use, a simple interface with consistent style will be implemented for this project.

Outline of the Report

- Related Work
- Dataset Description and Data Preprocessing
- Database Setup
- Mythodology
- Testing & Discussion
- Conclusion & Future Development
- References

Related Work

For the sale prediction analysis on non-stationary data, we found several sources recommending Autoregressive Integrated Moving Average (ARIMA) as it is one of the most used method for time series analysis because of its flexibility with incorporating unobserved variables and fits well with short-term data (Li, 2018; Salinas, 2020; Singh, 2020). For frequent data mining on customer and products, FP-growth algorithm was chose over Apriori as FP-growth only scans the database twice to check for the support count of each item and itemset. With the size of our dataset, implementing Apriori algorithm might increase the cost significantly as it requires more disk I/O executions and computing memory. Based on a research in 2013 evaluating the performance difference between the two aforementioned algorithms, FP-Growth showed significantly better performance generally (S.Mythili & R. Mohamed Shanavas, 2013). This conclusion was formed by comparing the execution time with different number of transactions and various support levels.

Dataset Description & Data Preprocessing

We used a subset of the dataset Instacart Market Basket Analysis from Kaggle for development, this subset includes several csv files to provide information about the aisles, departments, products and orders.

- The aisles.csv file includes unique aisle id along with the name of the aisle, we added a column called min_amount indicating the minimum amount of product that the store should have for each product in that specific aisle.

- The departments.csv fie includes unique department id along with the name of the department, we also added another column named storage_days which contains data regarding to the number of days a product will last in that department upon arrival.
- The products.csv file includes unique product id, name of the product, aisle id, and department id, we also added a default current stock quantity of 200 to this table.
- The orders_products_prior.csv includes many columns, we chose a subset of these such as the unique order id, product id, and user id, we also added a timestamp column to the table for several analyses; and aggregated the product id columns so that all the products that got purchased for an order is store together in same row with data type string.
- The orders_products_test.csv includes many columns, we only kept one column called product_id and aggregated this column by counting the occurrence of each unique product id. This is used as new order in automated warning section.

Database Setup

See final database structure and csv files that we used for the project down below.

We decided to use several csv files for analysis rather than importing and including these files as a part of the database, because it is very time consuming (~5-7 hours) to import small fractions of these files (see Future Development). We considered using only a subset of the following csv files, however, there are several reasons behind our decision of not doing that.

1. Based on data exploration at the beginning of the development stage, we noticed that the orders' data is sparse in way that there are so many products (~50,000) within the store, so most of the product was not being purchased very often based on the data provided for us. We also wanted to capture the entirety of the dataset

to hopefully produce meaningful analyses and predictions considering the algorithm that we use might overfit information if we decided to only take 10% of the dataset to use.

2. We also considered the increase in runtime changing from reading data from the SQL server, and reading data from an csv file on disk. However, for the purpose of this project and development, we thought the slight increase in wait time for some components is acceptable for us and the teacher assistant or the professor to test our application.

CSV files:

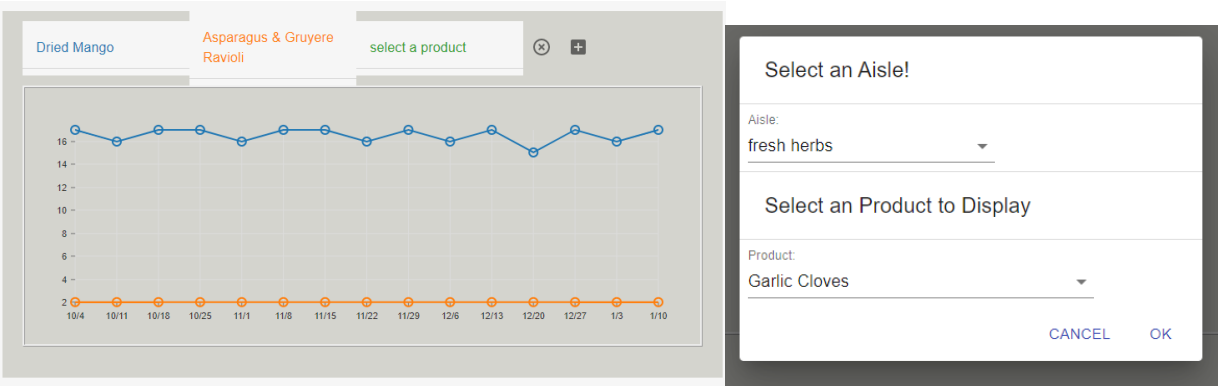
- orders_for_db.csv
    - <order_id (int), product_id (string), timestamp (datetime), user_id (int)>
    - Used for product analysis and customer analysis implemented using apriori algorithm
- time_series_data.csv
    - <product_id (int), timestamp (datetime)>
    - Used for time series analysis to predict the future sale pattern of a product.
- new_orders.csv
    - <product_id (int), quantity (int)>
    - Used for automated warning

Database:

- aisles, <id, name>

- departments, <id, name, storage_days>
- products, <id, name, aisle_id, department_id, current_stock>
- warn, <id, date>

Methodology



A. Item Sale Prediction

- Frontend:

This component consists a maximum of four product selection boxes on the top, and a graph corresponding to the selected products' future selling patterns. We matched the color of the selection box's text to the color of the line in the graph for the ease of use.

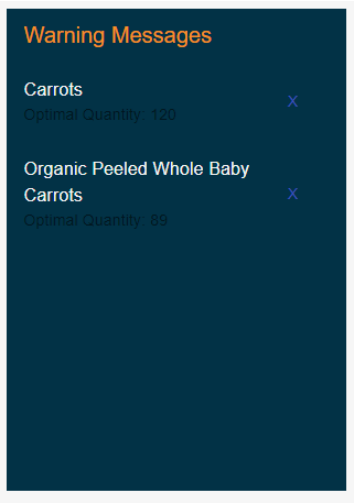
Once the user clicked the selection box, a dialog popup will appear in the middle of the application for the user to pick an aisle first in order to select a

product from that aisle. The user must select an aisle first, we designed it this way to reduce the amount of information being send back and for

- Backend:

We implemented Autoregressive Integrated Moving Average (ARIMA) for item sale prediction, one of the most used method for time series analysis (Li, 2018; Salinas, 2020; Singh, 2020). We started with data exploration to better understand our dataset for hyperparameter tuning for the model. Then we prepare the data for the model by applying data transformation techniques on the initial dataset.

After that, we used grid search to tune the hyperparameters for the model by finding the best combination of parameters. We evaluate the best parameters by comparing the Akaike information criterion score between all the parameters in order to choose the combination of parameters with the lowest score. Then we simply fit the model with the best parameter to get the result of the prediction.



B. Automated Warnings

- Frontend:

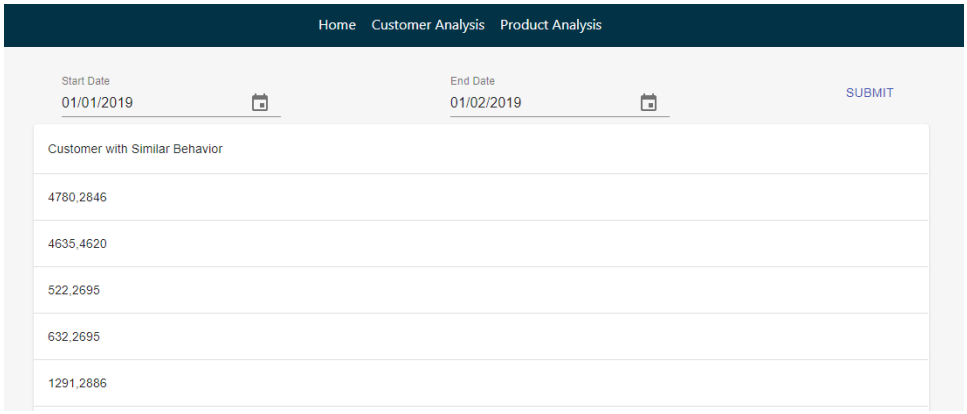
The automated warnings box displays all the low stock or out of stock items along with recommended amount to purchase in the next order by checking and getting the entries in the ‘warn’ table every 10 seconds, and joins that with the names from the product table. The user may remove any warning messages by clicking the delete icon after the item name.

- Backend:

Other than get and delete warnings from the ‘warn’ table for the frontend. We also implemented a new orders simulation functionality to process new orders coming in and updating the ‘warn” table in the database accordingly in newOrdersSimulation.py in the /Backend/python/ directory. This script reads a

new_order.csv file (see Database Setup section), and calculate the optimal quantity to order for a specific item next week.

The optimal quantity is calculated by considering the product's order amount, next week's expected sale for the product, empty storage space within the store, and the shelf life of that product.



C. Customer Analysis

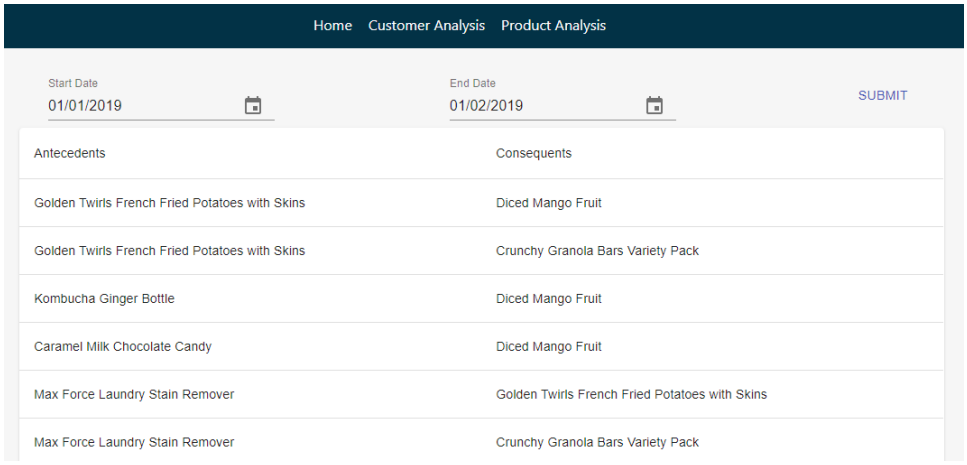
- Frontend:

This component consists of two date pickers, a submit button and a table to display customers with similar behavior.

- Backend:

We prepare the dataset orders_for_db.csv (see Database Setup section) and use the data in the correct format as input for the FP Growth model to

generate a list of customer ids who purchased the same set of products with a maximum of 100 sets.



D. Product Analysis

- Frontend:

  Similar to customer analysis except that this one generates a table that displays the relationship between antecedents and consequents. For example, if the antecedent is apple and consequent is peanut butter, then this represents that if a customer bought apple, then it is very likely that this same customer also bought peanut butter.

- Backend:

  Similar to customer analysis except we also include the confident factor in the apriori algorithm.

Testing & Discussion

The objectives are identifying existing problems within the program and evaluating implemented data mining optimization algorithm. The information gathered from each cycle of testing may be used to improve user experience and prediction accuracy of the program.

Constraints: The testing data is limited to the initial dataset given to the team.

Coverage

- List of features to be tested automatically
    - That our algorithm determines the most optimal spending and inventory management. Using a small simple but known dataset
- List of features which will not be tested automatically
    - Frontend components such as graphs, catalogues, message boxes and dropdown menu for item selection of the graph.

Unit testing was used throughout the development of the system to ensure each unit of the program performs as designed such as all front-end components before its integration with the backend. Testing was done by the members that created each section as new features were added. Any mistakes that were found were quickly fixed and communicated properly. Once the program reached a desired prediction accuracy using the chosen algorithm, we have determined the App is complete.

Conclusion & Future Development

In conclusion, if implemented in a large scale, this system could prevent stores from running out of stock on items. Our system runs on a single item as we are condensing many weeks into a few minutes. In the real world, the system will run on all items but would handle data real time as purchases happen.

If given more development time we would move our datafiles from CSV format to the database so that we can use better data retrieval methods. With our current setup, (allocated memory, clock speed, etc.) It took 7 hours to import 400,000 lines from the CSV to a database table. With 40,000,000 lines between the 3 primary CSV sheets, it was infeasible to import the data to a database which would take ~700 hours, or 29 days. With more time we could find and adjust the allocated memory variables in our MySQL environment to try and speed this import up significantly, but we decided to prioritize the completeness of the project.

CSV files could also be modified first then implemented into the database to fit with future development. With more time, we could also improve our dataset by sourcing real world minimum amount data, and storage days data for each product.

References

AI, R. (2019, September 25). Artificial intelligence for inventory management. Medium. https://medium.com/@RemiStudios/artificial-intelligence-for-inventory-management-c8a9c0c2a694

Kritchanchai, D., & Meesamut, W. (2015, June). Developing Inventory Management in Hospital. International Journal of Supply Chain Management, 4(2), 11-19. https://core.ac.uk/download/pdf/230741788.pdf

Li, S. (2018, September 5). *An end-to-end project on time series analysis and forecasting with python.* Medium. https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b

Salinas, D. (2020, December 17). *Python sales forecasting Kaggle competition.* Medium. https://towardsdatascience.com/python-sales-forecasting-kaggle-competition-40726b2ee047?gi=382b989079b9

Singh, B. P. (2020, June 23). *Predicting sales: Time series analysis & forecasting with Python.* Medium. https://medium.com/analytics-vidhya/predicting-sales-time-series-analysis-forecasting-with-python-b81d3e8ff03f

S.Mythili, M., & R. Mohamed Shanavas, A. (2013). Performance evaluation of Apriori and FP-growth algorithms. *International Journal of Computer Application, 79*(10), 34-37. https://doi.org/10.5120/13779-1650