# The Grocery Cart Analysis

## CAP5610 - Final Project Report

Dhanashree Dilip Shinde
dhanashrees@knights.ucf.edu

Aditya Pawar
aditya281198@knights.ucf.edu

## Abstract

Instacart is an online grocery delivery service. Customers using the app place orders for goods, which are then hand-delivered to their homes. This service is especially useful when a global pandemic has confined many people to their homes. Imagine a Sunday morning where you look at your schedule for the next few days after your socially distanced morning hike. Your refrigerator needs to be restocked, but the weekend flocks at the grocery store bore you. Monday and Tuesday are jam-packed with Zoom meetings, and you'll be supervising your kids' remote learning as well. In short, you're not going to the grocery store anytime soon. So, you take out your phone, launch the Instacart app, and choose your preferred grocery store. You go through your previously purchased items, look at specials, look for a new key-lime sparkling water that a friend recommended, and then choose a delivery window. We compared the accuracy of various models in this project. We present a thorough comparative study of various machine learning models on the Instacart dataset. It will enable us to provide a more credible recommended list that is tailored, personalized, and targeted to each user, as well as a better customer experience.

## Introduction

The coronavirus pandemic has catapulted online grocery, a remunerative but previously specific market industry, to the forefront. Customers' desire to avoid public spaces, government directives to stay at home, and the ongoing need for food and necessities have made online grocery delivery services from Walmart, Amazon, Target, and Instacart indispensable.

Prior to the pandemic, some customers resisted the shopping method because they liked to choose their own groceries and avoid additional costs. However, many have since changed their minds. Additionally, the sudden emphasis on food delivery is probably going to alter consumer behavior even after the pandemic is over, accelerating the industry's penetration in the US. Due to the extraordinary times, more people are choosing to do their shopping online. Online shopping is a great convenience, but for some people it can be difficult because the items are hard to see, or the payment is made in a hurried manner. To make the shopping experience more valuable for customers, online shopping apps also offer suggestions to users based on their prior data.

# Problem Statement and Goal

Many people now prefer to work from home instead of going out as frequently for things like shopping for groceries, which has been negatively impacted by most retailers due to the state of the world. People are now seeking out online shopping options as a direct consequence. Only a few examples of big and small online retailers include Amazon, Walmart, Instacart, and local shops. As the demand for these online applications increases, it's critical now more than ever to improve the user experience and deal with problems like:

- Consumers are unable to view recently released goods or goods that are comparable to what they've already bought.
- search results that are irrelevant or limited filtering options
- obtaining suggestions that are not helpful.

The inability of online retailers to thrive in this environment is exacerbated by these issues.

The primary research goals are to develop user segments based on time periods and a framework for product recommendations based on user preferences. The study's conclusions are anticipated to enhance inventory distribution on the supply side and raise the probability that buyers will receive essential goods without breaking the law against social distance.

Our objectives in this project are to:

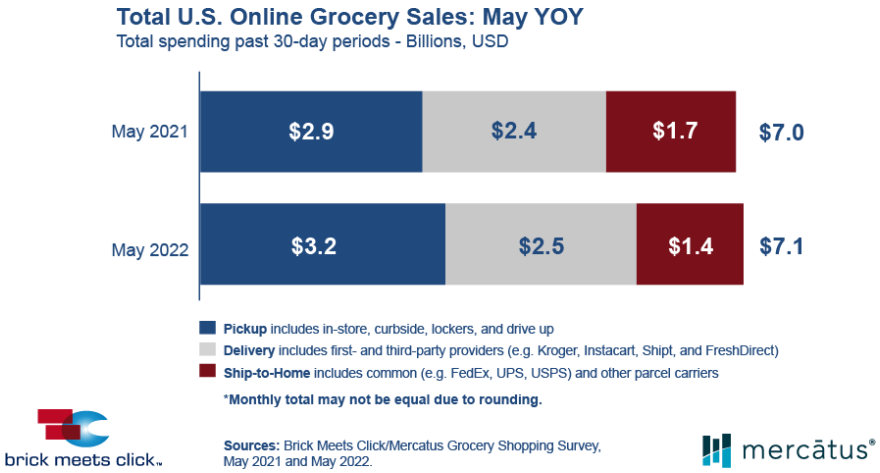
• To determine which products to suggest to users based on their prior purchases.

• To determine whether a product the user has previously purchased will be included in the user's upcoming order.

Most customers now face significant difficulties because of a lack of machine learning technology, which keeps businesses from making money. Machine learning algorithms will make all of this simpler by using recommendation features and recommending relevant items based on prior purchases, creating a more user-friendly shopping experience.

# Background Knowledge

- Consumer behavior changed because of the coronavirus pandemic, forcing food retailers to adapt rapidly. Numerous households will keep using e-commerce options after the crisis is over. Millions of households started ordering groceries online for pickup or home delivery.
- The grocery delivery and ordering app called Instacart is used. Today's working professionals and graduate students don't have time to go grocery shopping in person; instead, they prefer to order their food online because of the pandemic.
- According to the Brick Meets Click/Mercatus Grocery Shopping Survey conducted May 28–29, 2022, total online grocery sales ended the month of May at $7.1 billion, 1.7% higher than a year earlier as inflation is encouraging many customers to adapt where and how they shop online for groceries.
- The goal of Instacart is to make internet purchasing simple with a single click. After users choose their products via the Instacart app, sales associates assess the order, complete the in-store shopping for the customers, and deliver the goods.



# System’s Overview

## 1. Dataset

Anonymized orders from Instacart consumers over time beginning in 2017 are the main source of data. It contains the department file, aisles file, order file, inventory file, order and product file, and order and product file. Each element in the dataset has a distinct id assigned to it.

We're utilizing the Instacart dataset from a successful Kaggle competition for our trials.

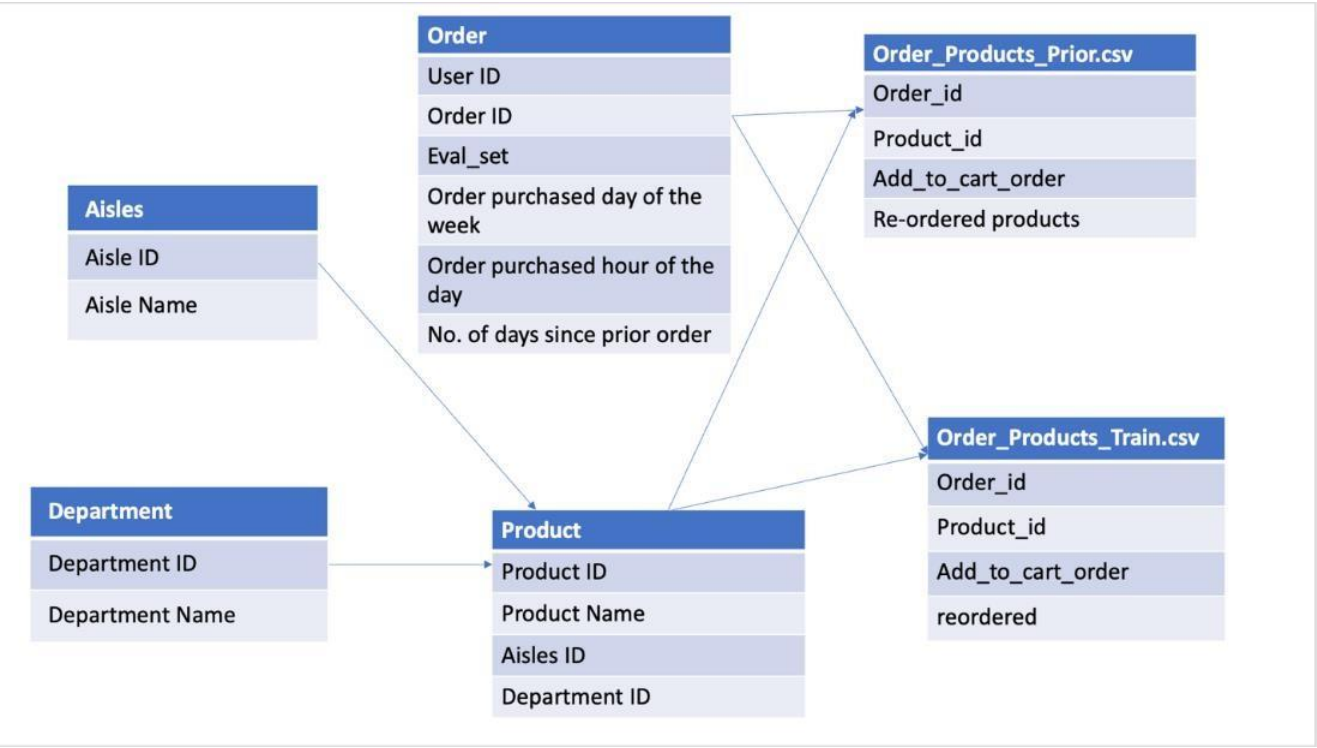
- Aisles.csv
- Departments.csv
- Orders.csv
- Products.csv
- order products
- Prior.csv
- order products train.csv

There are around 2 million Instacart customers who have placed 3 million grocery shopping. To retain the sequence in which they were ordered, only 4 to 100 orders per consumer are sampled. The provided dataset maintains consumer confidentiality by withholding the customers' identities.

If it is a first-time purchase, the days since the last transaction column will be NaN.

The names of connected departments are shown in the department table together with a specific department id. The aisle's table contains aisle ids and aisle names.

The aisles id, department id, product id, and product name are all included in the product table.



## 2. Outline

Task for the project:

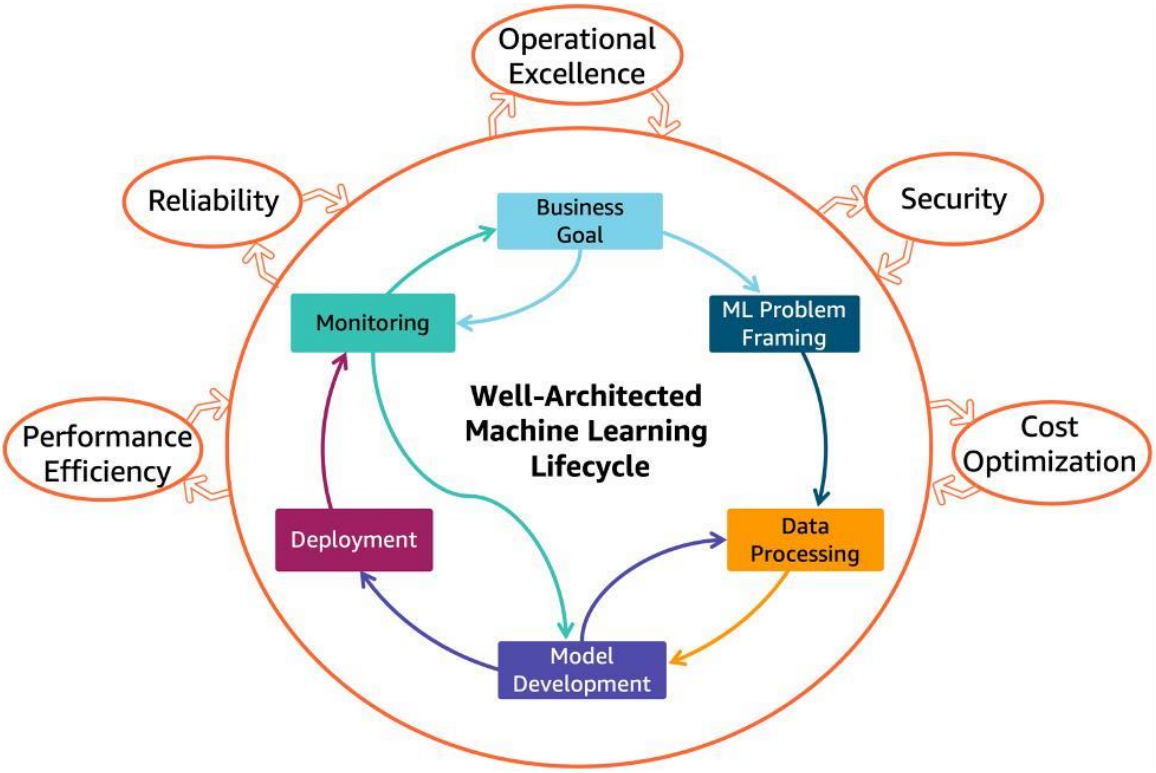
- Pre-processing of Dataset
- Analysis and Visualization of Dataset
- Feature Engineering
- Analysis of grocery carts using FP-Growth and Apriori algorithms
- Collaborative and Content filtering–based recommendation system.
- Applying Machine Learning and Data Learning models to get data of customers who tend to buy again based on previous purchases.

Our study has been divided into two parts:

- We examined the data distributions in our dataset, including the distribution of orders by day, by hour, by department, and so on. Along with other considerations, we examined the percentages of products that were reordered based on departments, day of the week, and hour of the day. The top 10 aisles and goods are also mentioned.
- In the second phase, we learned and modeled user behavior using a variety of strategies.

Our experiments are organized into three categories:

- To determine whether or not a product would be reordered and which department a user will visit next to make a purchase, we applied a range of machine learning algorithms.
- We were able to anticipate a subsequent product based on a number of predecessors using association rule mining.
- A product suggestion list is produced using collaborative-based filtering and product rating.



## 3. Technical Specifications

Due to the enormous amount of data that would be handled, the bare minimum proposed system will require a lot of processing power. We utilized a computer with a 1060 Nvidia GPU, at least 8 GB of RAM, and no background processes so that it could process everything without any delays.

Python is the major programming language required for this project, along with libraries like NumPy for solving algebraic equations, matplotlib for data visualization, panda for data processing, and a few libraries listed below for employing machine learning models and algorithms.

For binary classification tasks,

- Logistic Regression
- Naïve Bayes
- Linear SVM
- K-nearest Neighbor
- Random Forest
- AdaBoost
- Gradient Boosting
- Artificial Neural Network

For associative rule mining,

- Apriori
- FP Growth

Surprise library is used for product recommendations. And Keras Library is employed for Deep Learning models.

## 4. Evaluation Metrics

The associative rule mining data that was taken from the dataset was validated using the evaluation criteria listed below:

**1. Support**

By dividing the frequency by the dataset's scale, one may determine how frequently an itemset appears in a dataset.

**2. Confidence**

Confidence is computed by dividing the support of one or more items by the support of a subset of the numerator.

**3. Lift**

The lift is the differences between two or more items observed and expected frequencies. It displays if two or more things frequently occur together rather than randomly. A value larger than one denotes a non-random association.

Formulas are as follows:

$$Support(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Total\ number\ of\ transactions}$$

$$Confidence(\{X\} \rightarrow \{Y\}) = \frac{Transactions\ containing\ both\ X\ and\ Y}{Transactions\ containing\ X}$$

$$Lift(\{X\} \rightarrow \{Y\}) = \frac{(Transactions\ containing\ both\ X\ and\ Y)/(Transactions\ containing\ X)}{Fraction\ of\ transactions\ containing\ Y}$$

The effectiveness of these models was evaluated, and they were compared to one another, using the following evaluation criteria:

1. Accuracy: One of the easiest measurement metrics to understand is accuracy. It is also referred to as the proportion of accurate predictions among all predictions. The following formula can be used to calculate it:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

5. Log loss Score: The Log loss Score is also known as the Logistic Loss or Cross Entropy Loss. This is the logistic model's negative loglikelihood, which returns probabilities for the training set.

$$Logloss_i = -[y_i\ ln\ p_i + (1 - y_i)\ ln(1 - p_i)]$$

6. Test Score: The Test Score determines what proportion of items are bought from the suggested list. It can be calculated using the formula below:

$$\textbf{Test Score} = \frac{\boldsymbol{Count\ of\ Correct\ Recommendation}}{\boldsymbol{Total\ Length\ of\ Order}}$$

7. F1 Score: It is an expanded version of fundamental accuracy metrics that seeks to balance recall and precision.

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

F1 Score

The correlation of recommender system models was evaluated using the following evaluation metrics, which were then compared to one another:

1. Root Mean Square Error: This statistic shows how much the predicted and actual values differ from one another. Regression models are where this evaluation metric is most frequently used.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

2. The Mean Absolute Error: The MAE, or mean absolute error, is the average of the differences between the predicted and actual values. The error is determined linearly.

$$\text{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

# Approach

## 1. Data Loading

Various CSV files, including order, products, aisles, departments, order product prior, and order product prior, make up the Instacart dataset. Primary information on grocery items can be found in the products, aisles, and departments dataset.

Each customer's unique ID and the time the order was placed are listed in the orders data file along with the products they ordered. If the customer reordered a specific product after making an initial purchase, it is indicated in the order product prior and train data frame.

## 2. Data Cleaning

Dataset files are first examined for missing values as part of the data preprocessing process. It is clear from the image below that the dataset is accurate and devoid of any missing values. Therefore, no data imputation is required. Because this is a first-time order, there are no values in the "days since prior order" field; these null values can be ignored.

```
aisles.isna().sum()
```

```
aisle_id    0
aisle       0
dtype: int64
```

```
departments.isna().sum()
```

```
department_id    0
department       0
dtype: int64
```

```
orders.isna().sum()
```

```
order_id                      0
user_id                       0
eval_set                      0
order_number                  0
order_dow                     0
order_hour_of_day             0
days_since_prior_order    206209
dtype: int64
```

```
products.isna().sum()
```

```
product_id       0
product_name     0
aisle_id         0
department_id    0
dtype: int64
```

```
order_products__train.isna().sum()
```

```
order_id             0
product_id           0
add_to_cart_order    0
reordered            0
dtype: int64
```

```
order_products__prior.isna().sum()
```

```
order_id             0
product_id           0
add_to_cart_order    0
reordered            0
dtype: int64
```

## 3. Data Merging

The dataset's information is cluttered up in various files during the data loading section. Bits and pieces of attributes must be combined into a single data frame. Having completed the data merging process, we began working on the tasks outlined in our problem statement. The combined Order Product prior and Order Product Train data frame is displayed in the following tabulation. Here, the learning model is validated using Order Product Prior Train data.

```
order_products__prior.head()
```

| | order_id | product_id | add_to_cart_order | reordered | user_id | eval_set | order_number | order_dow | order_hour_of_day | days_since_prior_order | product_name | aisle_id | department_id | aisle | department |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2 | 33120 | 1 | 1 | 202279 | prior | 3 | 5 | 9 | 8.0 | Organic Egg Whites | 86 | 16 | eggs | dairy eggs |
| **1** | 2 | 28985 | 2 | 1 | 202279 | prior | 3 | 5 | 9 | 8.0 | Michigan Organic Kale | 83 | 4 | fresh vegetables | produce |
| **2** | 2 | 9327 | 3 | 0 | 202279 | prior | 3 | 5 | 9 | 8.0 | Garlic Powder | 104 | 13 | spices seasonings | pantry |
| **3** | 2 | 45918 | 4 | 1 | 202279 | prior | 3 | 5 | 9 | 8.0 | Coconut Butter | 19 | 13 | oils vinegars | pantry |
| **4** | 2 | 30035 | 5 | 0 | 202279 | prior | 3 | 5 | 9 | 8.0 | Natural Sweetener | 17 | 13 | baking ingredients | pantry |

```
order_products__train.head()
```

| | order_id | product_id | add_to_cart_order | reordered | user_id | eval_set | order_number | order_dow | order_hour_of_day | days_since_prior_order | product_name | aisle_id | department_id | aisle | department |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 49302 | 1 | 1 | 112108 | train | 4 | 4 | 10 | 9.0 | Bulgarian Yogurt | 120 | 16 | yogurt | dairy eggs |
| **1** | 1 | 11109 | 2 | 1 | 112108 | train | 4 | 4 | 10 | 9.0 | Organic 4% Milk Fat Whole Milk Cottage Cheese | 108 | 16 | other creams cheeses | dairy eggs |
| **2** | 1 | 10246 | 3 | 0 | 112108 | train | 4 | 4 | 10 | 9.0 | Organic Celery Hearts | 83 | 4 | fresh vegetables | produce |
| **3** | 1 | 49683 | 4 | 0 | 112108 | train | 4 | 4 | 10 | 9.0 | Cucumber Kirby | 83 | 4 | fresh vegetables | produce |
| **4** | 1 | 43633 | 5 | 1 | 112108 | train | 4 | 4 | 10 | 9.0 | Lightly Smoked Sardines in Olive Oil | 95 | 15 | canned meat seafood | canned goods |

## 4. Feature Extraction

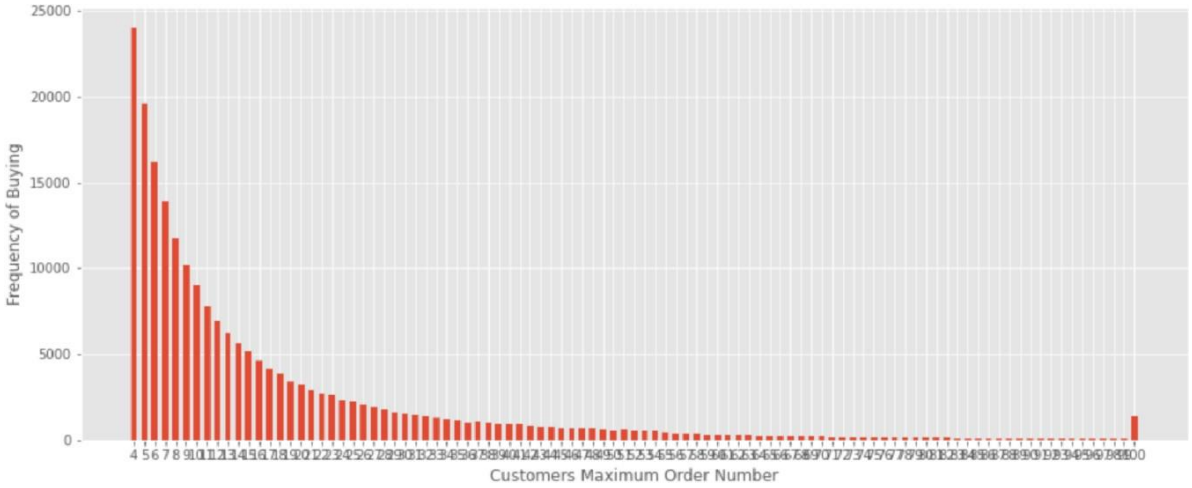
Although we have sufficiently prepared the data for model prediction, associative rule mining algorithms still need a set of products that have been ordered in each transaction as input. Order product dataset is grouped by distinct order ID in order to obtain ordered products in a transaction, and products column is added to an array for each order. The transaction dataset table is created in this manner for the purpose of rule mining.

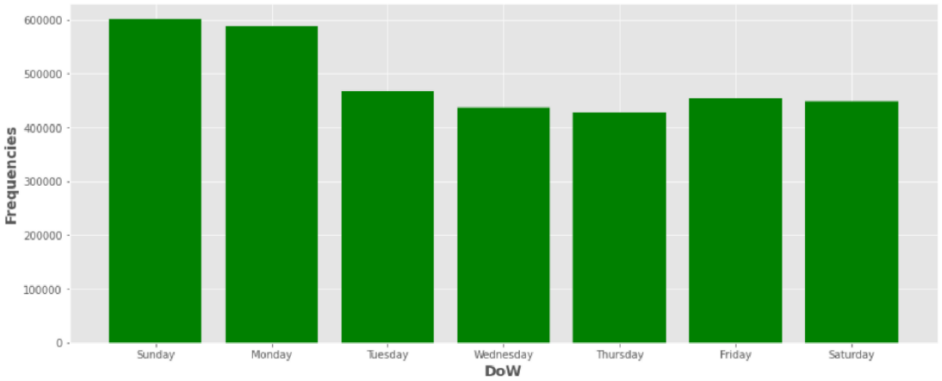
```
transaction.head()
```

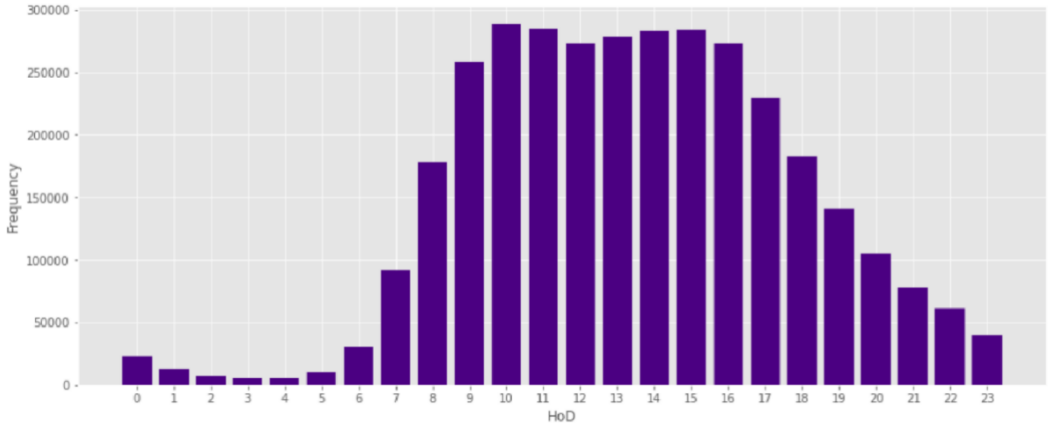
| | OrderID | Products |
|---|---|---|
| 0 | 2 | [33120, 28985, 9327, 45918, 30035, 17794, 4014... |
| 1 | 3 | [33754, 24838, 17704, 21903, 17668, 46667, 174... |
| 2 | 4 | [46842, 26434, 39758, 27761, 10054, 21351, 225... |
| 3 | 5 | [13176, 15005, 47329, 27966, 23909, 48370, 132... |
| 4 | 6 | [40462, 15873, 41897] |

## 5. Exploratory Data Analysis

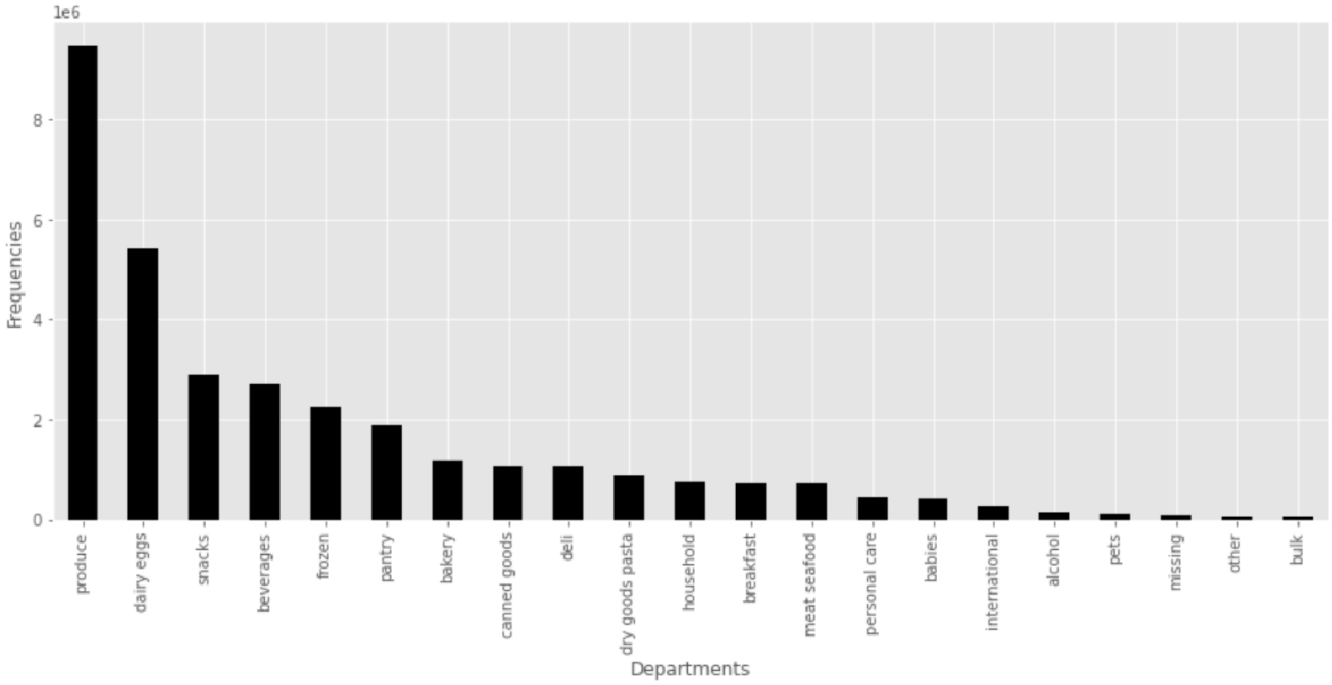
After feature extraction from the data and data preprocessing, we began our next task by making some interesting findings. We begin our data analysis by using the dataset file for orders, products, and aisles. These datasets provided in-depth information on a department's products as well as regular order totals for different time frames, such as month and day-by-day order counts. The visualization that follows offers a summary of the information.

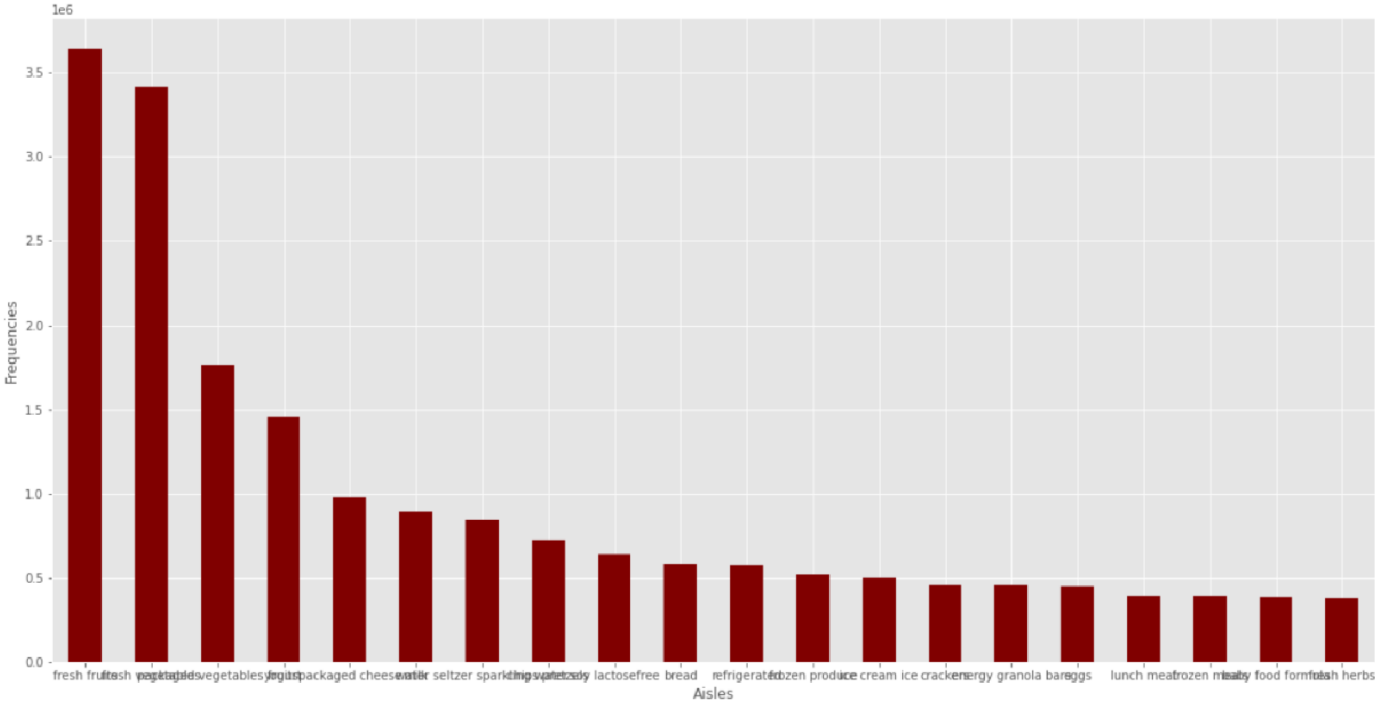


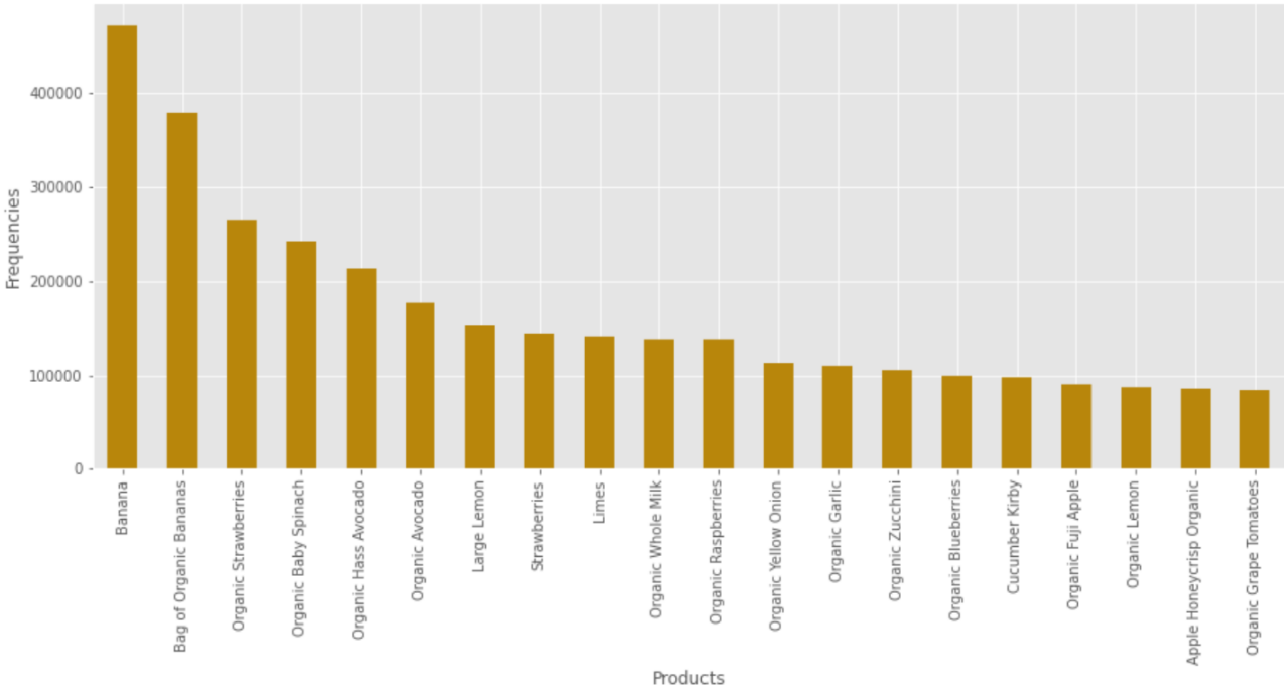





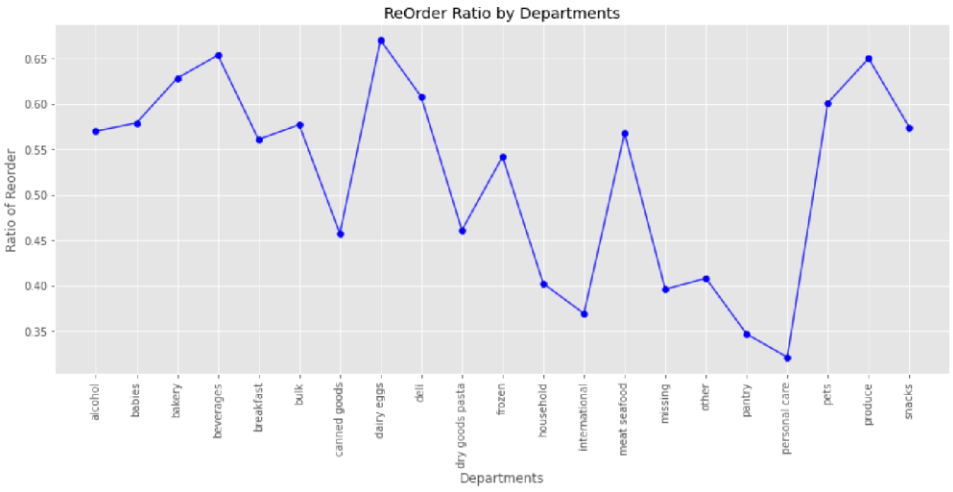

According to the the above figures, which represent the maximum number of orders placed by customers, the highest order exceeds 2000. The weekday plot demonstrates the regularity of customer orders. This graph demonstrates that orders are placed in the greatest volume on the weekends. Additionally, the frequency of orders is displayed in relation to each hour of the day. The distribution of orders by frequency and customer movement throughout the day are closely related.

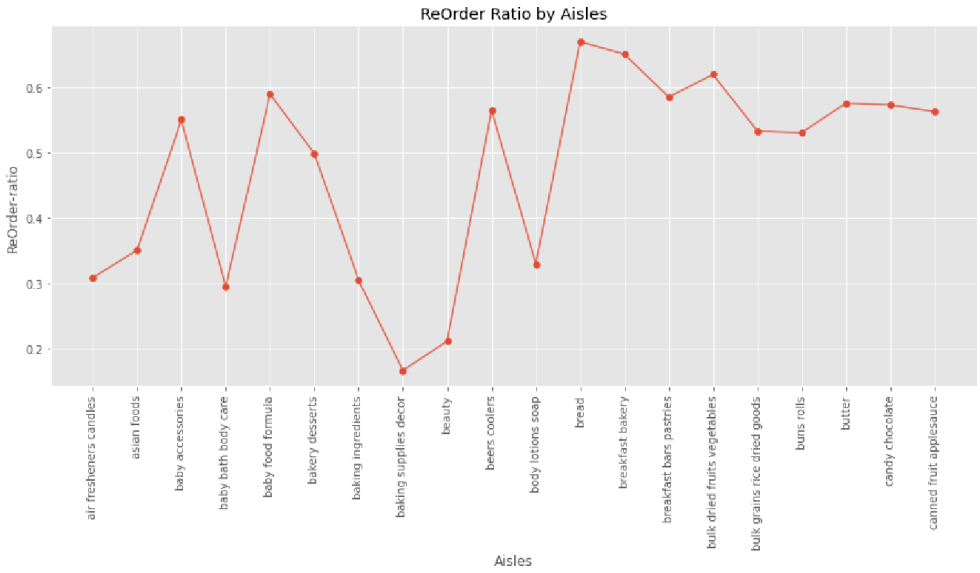





The image above displays the most frequently ordered product according to a dataset that includes data on product category, department, and aisle organization in stores. It is evident that organic products are the most ordered in total transactions by visualizing plots using all of this dataset information.




## Methods Followed

A variety of techniques were used to learn and model user actions from the Instacart dataset. For the classification task, we used Machine Learning algorithms such as Logistic Regression, Random Forest, AdaBoost, KNN, Gradient Boosting, Nave Bayes, Linear SVM, and Artificial Neural Network.

While Logistic Regression employs a straightforward model that seeks to predict the class of an item using a logistic function, Random Forest makes use of decision trees to do the same. AdaBoost and Gradient Boost are two additional ensemble learning models that can be used to further boost the precision of these algorithms. Aside from that, linear SVM and KNN are used for classification. Our work included implementing an ANN model in the classification task to broaden the project's scope.

To predict goods (consequent) for the customer based on the goods already in the basket, we used two widely used algorithms: Apriori and FP-Growth (antecedent). Two popular algorithms for finding patterns are FP-Growth and Apriori.

The two most popular algorithms for finding patterns are FP-Growth and Apriori. They rely on a figure called the item's "help count," or the quantity of transactions that include it. It was necessary to filter out only the objects that happen frequently enough. Insufficient occurrences of an object will result in no interesting trends.

## Apriori Algorithm

The Apriori algorithm makes use of the apriori property. This property states that every subset of a frequent itemset is also frequent, and every superset of an infrequent itemset can also be infrequent. The algorithm takes advantage of this property by counting the common items between smaller frequent item sets and the larger itemset.

## FP Growth Algorithm

Each transaction in FP-Growth is sorted by the support values contained in it after the items have been filtered. The transactions are then used by the algorithm to build a tree. We can follow the edges of the constructed tree using the items included in each transaction. The algorithm then extracts the most frequent patterns from the tree.

These two algorithms were employed for rule mining of product relationships and purchasing patterns. These connections can be used to predict which product will be bought after that or which products are most likely to be bought together.

# Recommender Systems

Later we applied collaborative filtering algorithms to the Instacart dataset to make product recommendations. It is a well-known recommendation algorithm that focuses its recommendations on the actions of other system users. Most collaborative filtering algorithms function by predicting the user's preferences and then recommending items based on metrics that measure similarity between users or items that the user interacted with.

An algorithmic interpretation of how similar users are to one another is called user-user CF. Finding users whose interactions or purchases are similar to the other users' is the goal of this technique. K-NN algorithm is used to cluster users. This system's limitation is its difficulty in expanding the model to accommodate an increasing user base. In contrast, the filter in Item-Item CF makes use of similarities in the rating patterns of the items. To find the most similar item to recommend, we used two functions.

```
matrix_train.head()
```

| | user_id | product_name | rating |
|---|---|---|---|
| 0 | 112108 | Bulgarian Yogurt | 1 |
| 1 | 112108 | Organic 4% Milk Fat Whole Milk Cottage Cheese | 1 |
| 2 | 112108 | Organic Celery Hearts | 1 |
| 3 | 112108 | Cucumber Kirby | 1 |
| 4 | 112108 | Lightly Smoked Sardines in Olive Oil | 1 |

# Outcomes

The report's earlier sections covered the implementation of our project's methodologies and workflow.

This chapter focuses on the conclusions drawn from each model's various parameterizations. According to the goal of our task, which discusses the outcomes of the models used in each task, the subsections are as follows:

# Associative Rule Mining

## Apriori Algorithm

Following the implementation of the apriori algorithm, pattern mining is carried out using different algorithmic parameters like lift, support, and confidence. First, the Apriori algorithm was applied, with a support and confidence level of 0.01 and 0.2, respectively. The support was then reduced to 0.005 with the same confidence level of 0.2 in order to extract additional rules.

The top 10 rules identified by the Apriori algorithm for the two distinct instances are shown in the figures below.

| | Product_A | Product_B | Support_A | Support_B | Support_AB | Confidence_AB | Lift_AB |
|---|---|---|---|---|---|---|---|
| 7 | Organic Fuji Apple | Banana | 0.027942 | 0.147194 | 0.010630 | 0.380431 | 2.584554 |
| 5 | Cucumber Kirby | Banana | 0.030218 | 0.147194 | 0.010024 | 0.331723 | 2.253644 |
| 3 | Organic Avocado | Banana | 0.054564 | 0.147194 | 0.016328 | 0.299245 | 2.032997 |
| 0 | Organic Raspberries | Bag of Organic Bananas | 0.042334 | 0.117802 | 0.012428 | 0.293570 | 2.492064 |
| 1 | Organic Hass Avocado | Bag of Organic Bananas | 0.066218 | 0.117802 | 0.019260 | 0.290857 | 2.469037 |
| 8 | Strawberries | Banana | 0.044496 | 0.147194 | 0.012902 | 0.289959 | 1.969908 |
| 6 | Large Lemon | Banana | 0.047718 | 0.147194 | 0.012692 | 0.265979 | 1.806998 |
| 11 | Organic Raspberries | Organic Strawberries | 0.042334 | 0.082550 | 0.010524 | 0.248595 | 3.011442 |
| 10 | Organic Strawberries | Bag of Organic Bananas | 0.082550 | 0.117802 | 0.019122 | 0.231641 | 1.966362 |
| 4 | Organic Baby Spinach | Banana | 0.075000 | 0.147194 | 0.016016 | 0.213547 | 1.450784 |
| 2 | Organic Strawberries | Banana | 0.082550 | 0.147194 | 0.017596 | 0.213156 | 1.448127 |
| 9 | Organic Baby Spinach | Bag of Organic Bananas | 0.075000 | 0.117802 | 0.015500 | 0.206667 | 1.754356 |

Support = 0.01 and Confidence = 0.2 – Sorting by Confidence

| | Product_A | Product_B | Support_A | Support_B | Support_AB | Confidence_AB | Lift_AB |
|---|---|---|---|---|---|---|---|
| 16 | Organic Fuji Apple | Banana | 0.027942 | 0.147194 | 0.010630 | 0.380431 | 2.584554 |
| 19 | Honeycrisp Apple | Banana | 0.024774 | 0.147194 | 0.008832 | 0.356503 | 2.421993 |
| 6 | Cucumber Kirby | Banana | 0.030218 | 0.147194 | 0.010024 | 0.331723 | 2.253644 |
| 27 | Organic Large Extra Fancy Fuji Apple | Bag of Organic Bananas | 0.023374 | 0.117802 | 0.007330 | 0.313596 | 2.662063 |
| 4 | Organic Avocado | Banana | 0.054564 | 0.147194 | 0.016328 | 0.299245 | 2.032997 |
| 35 | Seedless Red Grapes | Banana | 0.026054 | 0.147194 | 0.007750 | 0.297459 | 2.020864 |
| 2 | Yellow Onions | Banana | 0.022814 | 0.147194 | 0.006750 | 0.295871 | 2.010075 |
| 0 | Organic Raspberries | Bag of Organic Bananas | 0.042334 | 0.117802 | 0.012428 | 0.293570 | 2.492064 |
| 20 | Blueberries | Banana | 0.017674 | 0.147194 | 0.005186 | 0.293425 | 1.993460 |
| 1 | Organic Hass Avocado | Bag of Organic Bananas | 0.066218 | 0.117802 | 0.019260 | 0.290857 | 2.469037 |

Support = 0. 005 and Confidence = 0.2 - Sorting by Confidence

## FP Growth Algorithm

The FP-Growth algorithm, an improvised version of the Apriori algorithm, undertake the same testing conditions as the Apriori algorithm. The results from running this model with various parameters are as follows.

| | Product_A | Product_B | Support_A | Support_B | Support_AB | Confidence_AB | Lift_AB |
|---|---|---|---|---|---|---|---|
| 0 | Organic Fuji Apple | Banana | 0.027942 | 0.147194 | 0.010630 | 0.380431 | 2.584554 |
| 1 | Cucumber Kirby | Banana | 0.030218 | 0.147194 | 0.010024 | 0.331723 | 2.253644 |
| 4 | Organic Avocado | Banana | 0.054564 | 0.147194 | 0.016328 | 0.299245 | 2.032997 |
| 2 | Strawberries | Banana | 0.044496 | 0.147194 | 0.012902 | 0.289959 | 1.969908 |
| 3 | Large Lemon | Banana | 0.047718 | 0.147194 | 0.012692 | 0.265979 | 1.806998 |

Support = 0.01 and Confidence = 0.2 – Sorting by Confidence

| | Product_A | Product_B | Support_A | Support_B | Support_AB | Confidence_AB | Lift_AB |
|---|---|---|---|---|---|---|---|
| 9 | Organic Fuji Apple | Banana | 0.027942 | 0.147194 | 0.010630 | 0.380431 | 2.584554 |
| 7 | Honeycrisp Apple | Banana | 0.024774 | 0.147194 | 0.008832 | 0.356503 | 2.421993 |
| 10 | Cucumber Kirby | Banana | 0.030218 | 0.147194 | 0.010024 | 0.331723 | 2.253644 |
| 4 | Organic Large Extra Fancy Fuji Apple | Bag of Organic Bananas | 0.023374 | 0.117802 | 0.007330 | 0.313596 | 2.662063 |
| 8 | Seedless Red Grapes | Banana | 0.026054 | 0.147194 | 0.007750 | 0.297459 | 2.020864 |
| 3 | Yellow Onions | Banana | 0.022814 | 0.147194 | 0.006750 | 0.295871 | 2.010075 |
| 0 | Blueberries | Banana | 0.017674 | 0.147194 | 0.005186 | 0.293425 | 1.993460 |
| 2 | Original Hummus | Banana | 0.022202 | 0.147194 | 0.005684 | 0.256013 | 1.739289 |
| 1 | Organic Cilantro | Limes | 0.021334 | 0.043640 | 0.005342 | 0.250398 | 5.737819 |
| 6 | Organic Baby Carrots | Banana | 0.023872 | 0.147194 | 0.005334 | 0.223442 | 1.518008 |
| 5 | Sparkling Water Grapefruit | Banana | 0.023752 | 0.147194 | 0.005018 | 0.211266 | 1.435292 |

Support = 0.005 and Confidence = 0.2 – Sorting by Confidence

Comparison of FP-growth and Apriori

• When compared to Apriori, FP-Growth is rapid.

• Nearly identical data mining rules have been offered by FP-Growth and Apriori.

• In comparison to Apriori, the FP-Growth algorithm produced fewer data mining rules, and its rules are a subset of Apriori rules. This suggests that FP-Growth only offers the most crucial rules.

| | Apriori | FP-Growth |
|---|---|---|
| **support = 0.01** | 2m 28s | 2m 37s |
| **support = 0.005** | 14m 22s | 5m 12s |
| | | |

# Collaborative Filtering

## Probabilistic Matrix Factorization

```
predictions_pmf = pmf_svd.test(x_test)
accuracy.mae(predictions_pmf)
accuracy.rmse(predictions_pmf)
```

```
MAE:  0.0019
RMSE: 0.0182
0.018222349534105484
```

```
matrix_train[matrix_train['user_id'] == 112108]
```

| | user_id | product_name | rating |
|---|---|---|---|
| 0 | 112108 | Bulgarian Yogurt | 1 |
| 1 | 112108 | Organic 4% Milk Fat Whole Milk Cottage Cheese | 1 |
| 2 | 112108 | Organic Celery Hearts | 1 |
| 3 | 112108 | Cucumber Kirby | 1 |
| 4 | 112108 | Lightly Smoked Sardines in Olive Oil | 1 |
| 5 | 112108 | Bag of Organic Bananas | 1 |
| 6 | 112108 | Organic Hass Avocado | 1 |
| 7 | 112108 | Organic Whole String Cheese | 1 |

## User Based CF

```
predictions_ub = user_based_cf.test(x_test)
accuracy.mae(predictions_ub)
accuracy.rmse(predictions_ub)
```

```
MAE:  0.0000
RMSE: 0.0000
0.0
```

## Item Based CF

```
predictions_ib = item_based_cf.test(x_test)
accuracy.mae(predictions_ib)
accuracy.rmse(predictions_ib)
```

```
MAE:  0.0000
RMSE: 0.0000
0.0
```

The comparison of evaluation metrics for different collaborative filtering models is shown in the table below. The lower the value of RMSE, the better the model's fit.

| Model | MAE | RMSE |
|---|---|---|
| User-based filtering | 0.0019 | 0.0182 |
| Item-based filtering | 0 | 0 |
| Probabilistic Matrix Factorization | 0 | 0 |

The training dataset is used to build similarity matrices for every user and item pair in the recommender system model. The validation test set is used to test the recommender system model after it has been constructed. The system's findings for user 1 are as follows:

```
pred_dict_ib[1]
```

```
['Organic String Cheese',
 'Cinnamon Toast Crunch',
 'Aged White Cheddar Popcorn']
```

```
pred_dict_ib[112108]
```

```
['Organic 4% Milk Fat Whole Milk Cottage Cheese',
 'Lightly Smoked Sardines in Olive Oil']
```

# Classification Models

To train a classifier model that can predict whether a customer will place another order for a product, the dataset "Order Product Prior" is used. Utilizing metrics such as accuracy score, precision, recall, and confusion matrix graph, each classifier model is evaluated for accuracy. The results of the experiment are displayed in the table below.

The performance of the models Linear SVM, Naive Bayes, and Logistic Regression is biased, according to the test set's results. These binary classifiers predict primarily positive outcomes, which is inconsistent with the actual labels. As a result, they perform poorly. However, KNN performs significantly better than the latter classifiers. Additionally, ensemble classifiers like AdaBoost, Gradient, and Random forest are trained on the reorder dataset. According to the results tabulation, Random Forest performs the best overall of all ensemble classifiers.

Furthermore, we developed an artificial neural network (ANN) deep learning model (ANN). As the classification problem is highly non-linear, the architecture for the ANN is as follows. We utilised three hidden layers between the input and output layers. There are 30 neurons in each of the first and second hidden states and 20 neurons in the third hidden layer. To prevent overfitting, a dropout layer with a dropout rate of 0.2 is applied across each hidden layer. In order to introduce nonlinearity, the rectifier (ReLu) activation function has been used in each of the hidden states.

The Random Forest model, which was used for our project, is the best model overall, having a good overall F1-Score and a good prediction rate when assessed at the class level.

## Related Work

The fast growth in the quantity of digital content and the number of Users of internet has led to a potential information overload problem that may limit timely access to online resources of interest.

This problem has been somewhat resolved by information retrieval systems like Google, Devil Finder, and Altavista, but prioritizing and personalization of the information are still absent. As a result, recommender systems are more in demand than ever. Based on the customer's predictions, preferences, or observed activities, recommender systems being the information filtering systems that deal with the issue of information overload by identifying important information or pieces from a large amount of constantly created data.

A recommender system will determine whether or not a certain user will appreciate an item based on the customer's profile. Programs that promote products or services to customers are profitable for both parties. They make product selection and product research less expensive while purchasing online. It has also been demonstrated that recommendation systems speed up and improve decision-making. In an e-commerce setting, recommender systems boost revenue since they are an effective technique to sell more products.

- Recommender systems survey. Knowledge-based systems by Bobadilla, J., Ortega, F., Hernando, A. and Gutiérrez, A., 2013.
- Del Olmo, F.H. and Gaudioso, E., 2008. Evaluation of recommender systems: A new approach. Expert Systems with Applications

These two papers have also helped us a lot in understating the concepts really well.

## Conclusion

The research we did for this project allowed us to predict potential products for a recommendation list and give users more sensible options. These algorithms may be used to run targeted advertising campaigns, which will increase the likelihood of receiving favorable reviews from customers of these online shopping apps.

Based on these discoveries, we might enhance an user's shopping experience by suggesting products that might be of interest to them. To further assist customers, we always keep in stock regularly ordered / reordered goods. By assigning more staff members to the crowded aisles and divisions, we can also reduce checkout times. We can also run targeted advertising campaigns based on certain demographics. Additionally, we can run ad campaigns that cater to specific clients and help them recommend products. These outcomes could also be used to send customers personalized alerts about products they frequently reorder.

In addition, the retailer might profit from the outcomes. First and foremost, enhancing a customer's shopping experience will raise service quality, which will ultimately lead to higher sales for the business. Second, retailers may use related products as a promotional item to pair new products/flavors with. Additionally, the revenue will rise as a result of this.

Code Link: https://github.com/shishir047/CAP5610-Machine-Learning/blob/main/ML_Project_%20grocery-cart-analysis.ipynb

# References


1. Instacart, “Instacart Market Basket Analysis.” 2017. [Online]. Available: https://www.kaggle.com/c/instacart-market-basket-analysis/data
2. https://docs.aws.amazon.com/wellarchitected/latest/machine-learning-lens/well-architected-machine-learning-lifecycle.html
3. https://www.brickmeetsclick.com/total-u-s--egrocery-sales-for-may-2022-up-almost-2--versus-last-year-to--7-1-billion
4. Breiman, Leo. "Random forests." UC Berkeley TR567
5. Wright, Raymond E. "Logistic regression." (1995
6. Data Mining Concepts: By Jiawei han