# Project Final Report

## Table of Contents

## Introduction/Background

In today's competitive retail landscape, customers have no dearth of options. Boosting sales hinges on strategies spanning effective product placement, efficient inventory management that ensures product availability, and compelling promotions to induce customers to buy more. Analyzing purchase history can reveal valuable insights that are often not intuitive, such as the surprising correlation between diapers and beer found in a case study.

Instacart, a popular grocery ordering platform, has open-sourced their anonymized transactional data spanning 3 million grocery orders from more than 200,000 users. The Dataset is ideal for analysis as it provides user-specific purchase history and captures purchase sequence, re-ordered items, and order placement times. Previous work in this domain has utilized association rule algorithms and regression models. In this project, we will build on these approaches to identify frequently purchased items, re-ordering patterns, peak purchase times, and other trends.

# Dataset Description

The provided dataset consists of multiple csv files which capture information about the products users have ordered over time. The data includes detailed information on each product purchased, the timing of orders, and which items were reordered.

The following are descriptions of the files and their contents:

1. aisles.csv: Contains the list of different aisles in the Instacart store.
    - aisle_id: Unique identifier for each aisle.
    - aisle: Name of the aisle.
2. departments.csv: Contains department information.
    - department_id: Unique identifier for each department.
    - department: Name of the department (e.g., "dairy," "produce").
3. products.csv: Lists all products sold, with details including:
    - product_id: Unique identifier for each product.
    - product_name: Name of the product.
    - aisle_id: Identifier of the aisle where the product is found.
    - department_id: Identifier of the department associated with the product.
4. orders.csv: Contains information on each order placed by users
    - order_id: Unique identifier for each order.
    - user_id: Unique identifier for each customer.
    - order_number: Sequence of the order (e.g., 1st, 2nd order) for that user.
    - order_dow: Day of the week the order was placed.
    - order_hour_of_day: Hour when the order was placed.
    - days_since_prior_order: Number of days since the user's last order (useful for understanding order frequency).
5. order_products__prior.csv: Lists prior order details, with each row representing a product from a past order.
    - order_id: Identifier linking the product to a specific order.
    - product_id: Identifier linking to a specific product.
    - add_to_cart_order: Position of the product in the cart.
    - reordered: Flag indicating if the product was reordered (1) or a first-time purchase (0).
6. order_products__train.csv: Similar to order_products__prior.csv, but specifically contains products from orders used in the training set.

# Problem Definition

The objective of this project is to convert data given in the form of past customer orders to actionable knowledge/insights, which can be used to provide a better experience to the customers, while at the same time growing sales/revenue for the company. We aim to achieve this by targeting two objectives

1. Purchase Trend Discovery - Given a user and past user order history, identify purchasing patterns (ex. Clustering users with similar purchasing behavior for product recommendation) through unsupervised learning algorithms. This could be a useful input for product re-order prediction (T. Lim, 2021).
2. Product Re-order Prediction - Given a user, product and user order history, predict if the user will re-order the given product in their next order through supervised learning algorithms.

Solving for these objectives would provide valuable insights, which can be used for product inventory management and improving customer experience (through bundled offers and effective product recommendation).

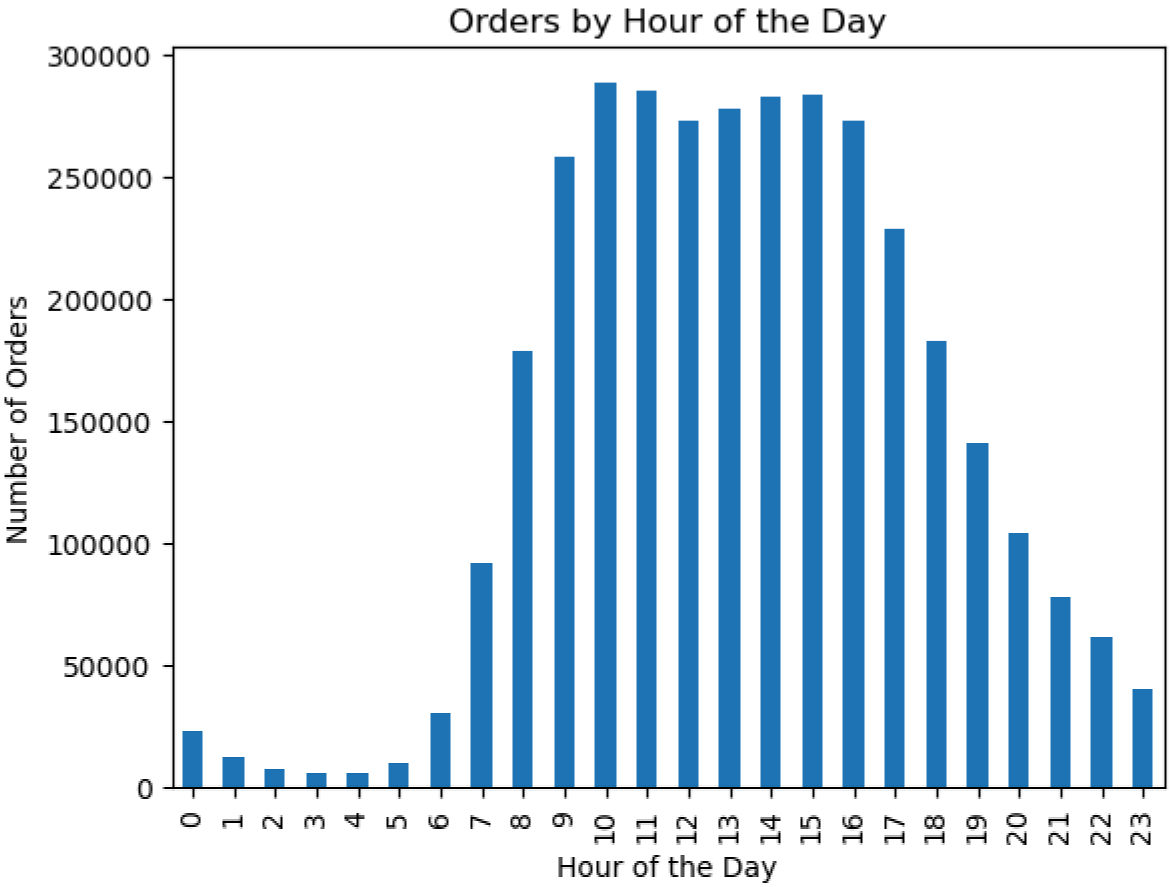
# Exploratory Data Analysis

In any data-driven project, Exploratory Data Analysis (EDA) begins to unravel the story that inevitably lurks within the data, revealing patterns, relationships, and potential trends that inform downstream analysis and modeling efforts. Through EDA, we can gain valuable insights into customer purchasing behavior, identify popular products, and discover temporal patterns that can support decision-making around stock management, promotions, and personalized recommendations.

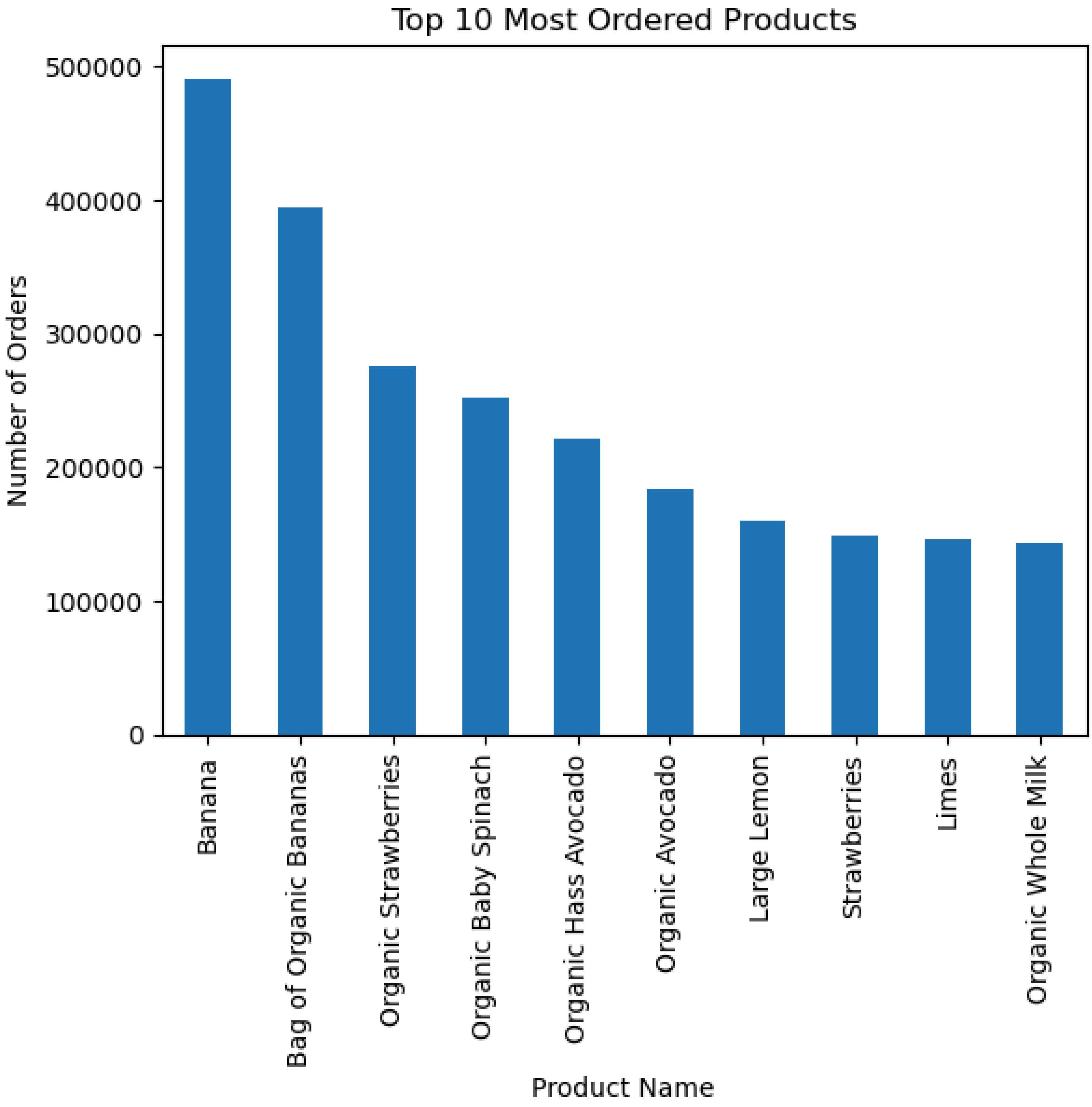
## Insights

1. **Shopping Behavior Insights -** The Exploratory Data Analysis has provided critical insights into customer shopping patterns and preferences. Analyzing order timing showed that most orders occur in the late morning and early afternoon, which can guide targeted promotions or stock management during peak hours. Observing the top 10 most ordered products highlighted essential items, such as bananas and milk, reinforcing the need for consistent availability of high-demand products.
2. **Reorder Patterns by Category -** Reorder patterns across departments revealed a strong customer loyalty to products in the dairy and produce sections, with high reorder ratios indicating a routine for these essentials. By understanding which departments drive repeat purchases, we can prioritize these items in recommendation models. Furthermore, aisle-level reorder ratios showed that certain aisles consistently include staple items that customers reorder frequently, making them ideal for predictive analysis in future orders.

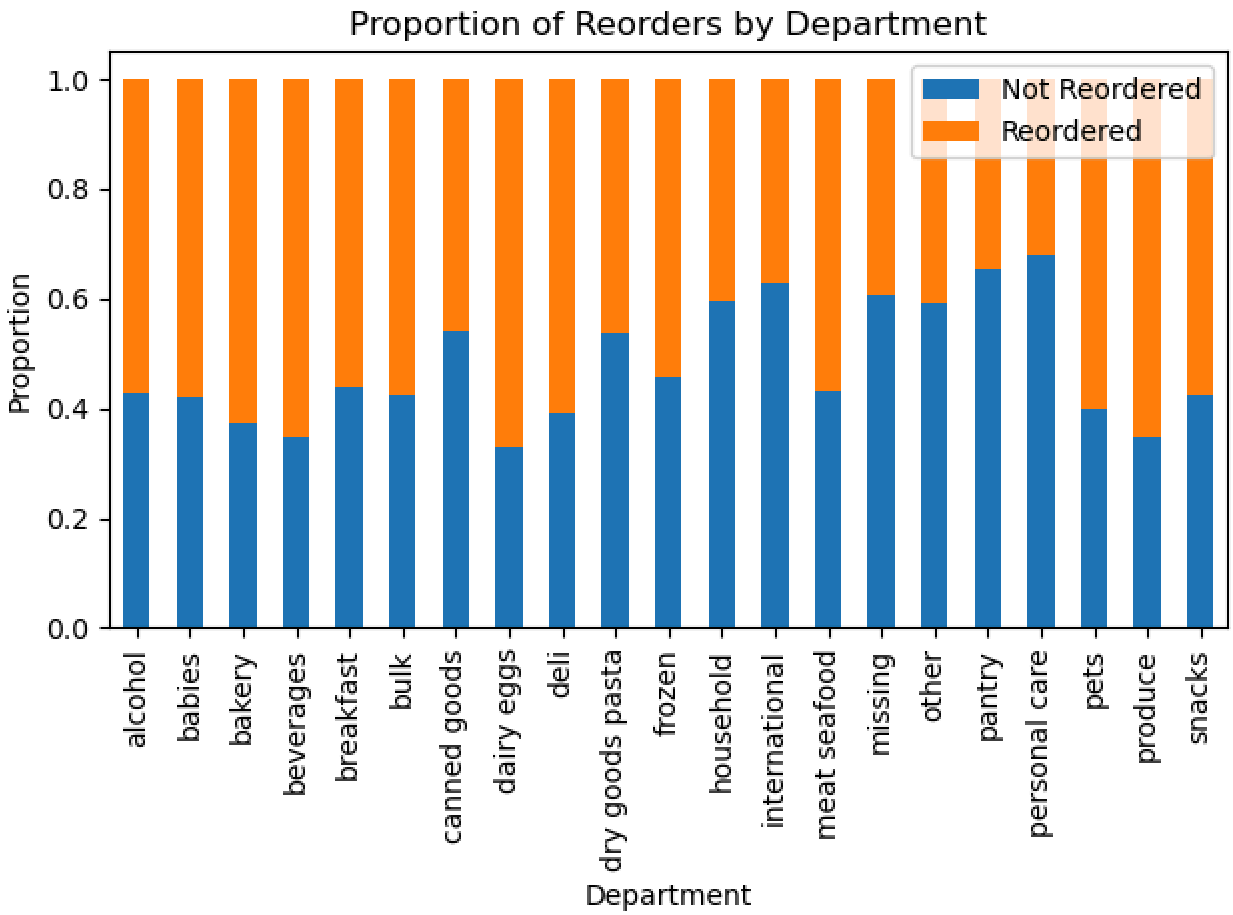
3. **Order Frequency and Implications -** The frequency of orders over time showed a typical weekly shopping cycle with a marked increase around Day 30, suggesting both weekly and monthly shopping habits. This data on time-based ordering habits will support our efforts to predict when customers are likely to reorder. Overall, these insights lay a solid foundation for developing feature-rich, data-informed models that enhance product recommendations and stock management, ensuring a more tailored shopping experience for customers.
4. **Top 10 Most Ordered Products -** Popular grocery items include bananas, milk, and eggs. This data can help in managing inventory to prioritize stocking for high-demand products.



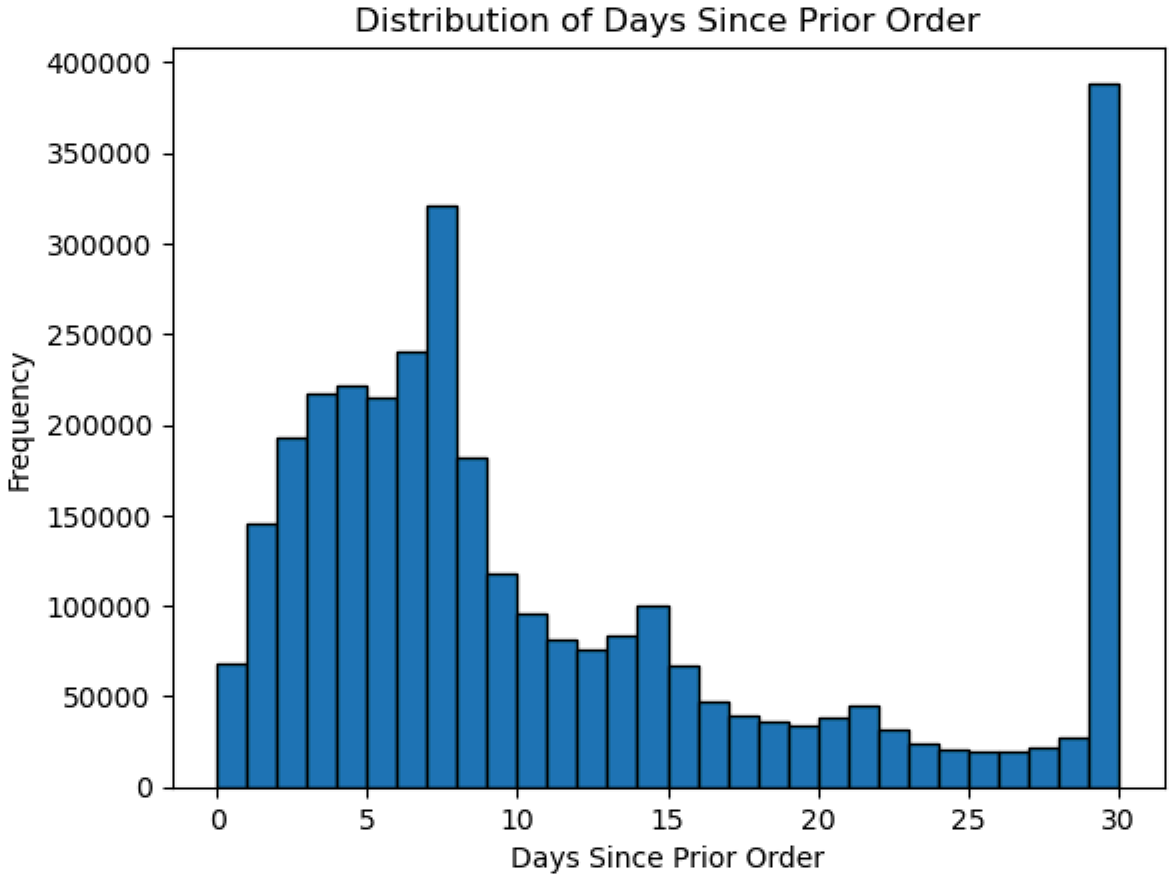
This graph shows the distribution of orders across different hours throughout the day, on average.



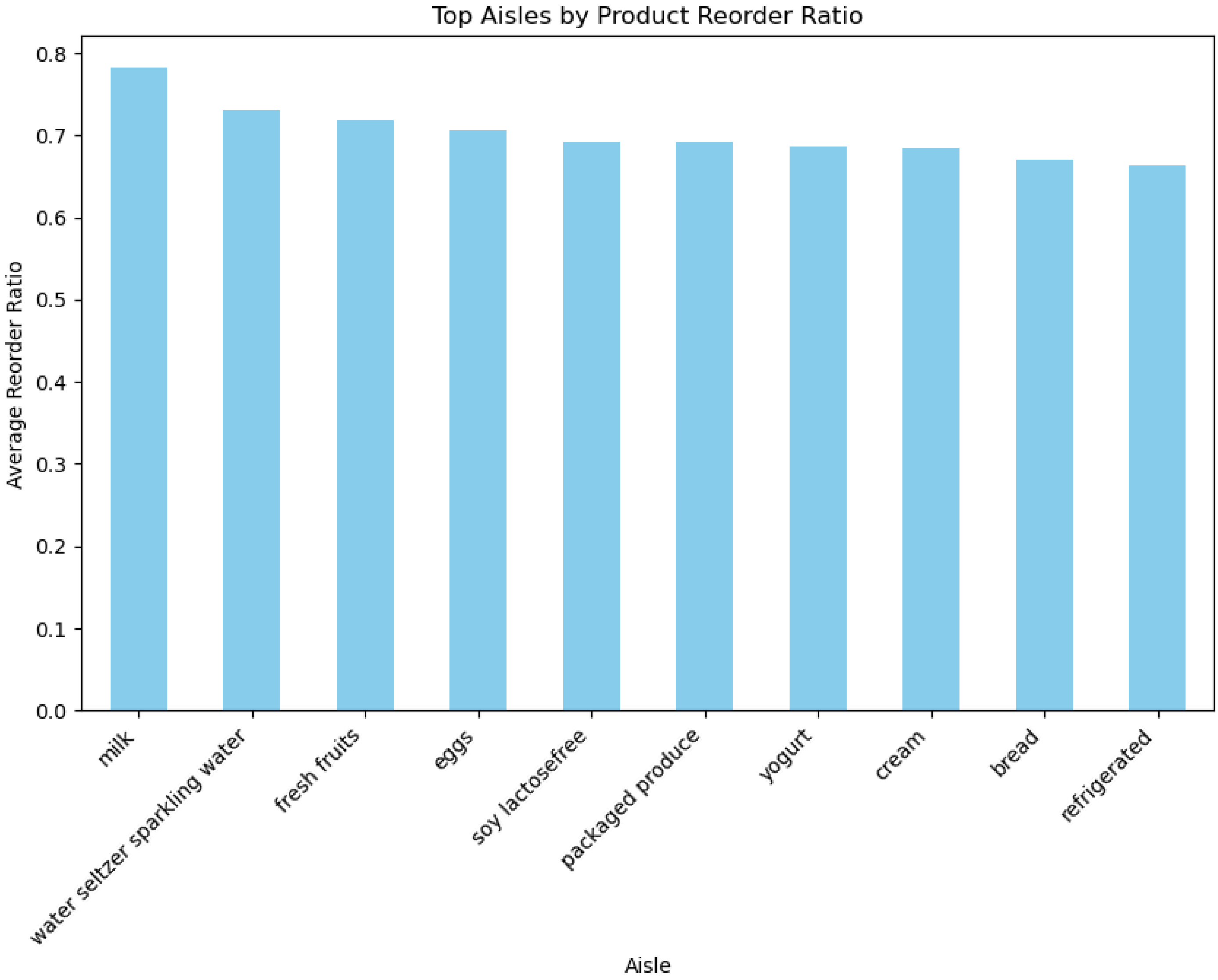
Top 10 most ordered products by count.



Reorder ratio across different departments.



Distribution of days since the last order.



Aisles that received the most reorders.

# Feature Engineering

Feature engineering is essential to come up with meaningful features that capture product-specific, user-centric insights from the raw transactional data. The main broad categories of the features explored are:

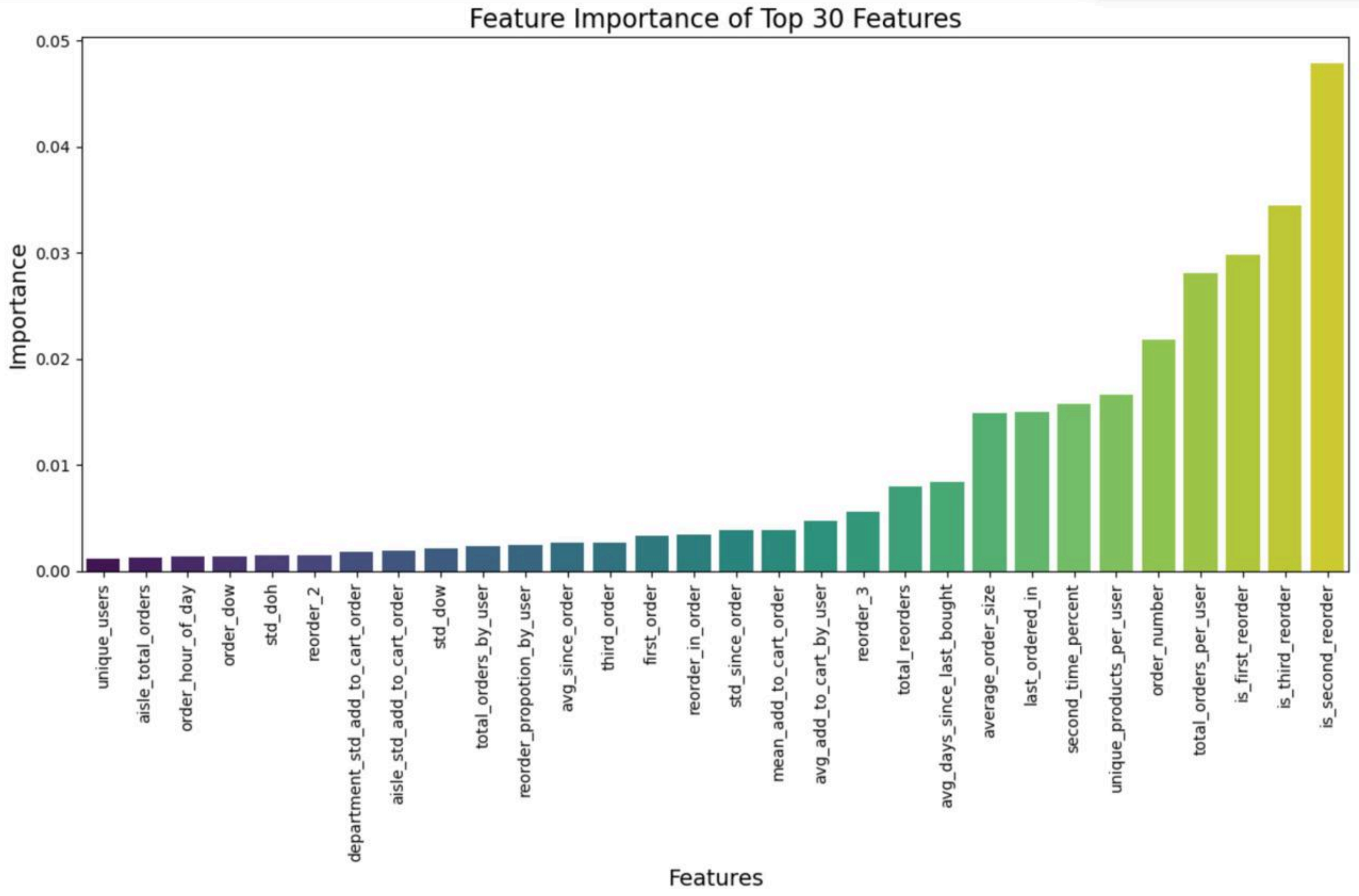
1. **Product-Level Features**
   - Aims to capture the popularity and purchase trends for a product.
   - Sample Features: reorder percentage, number of unique buyers, etc.

2. **User-Level Features**
   - Aims to capture user shopping cycles.
   - Sample Features: days before reorder, number of reordered items, etc.

3. **Aisle and Department Features**
   - Aims to capture the correlation of purchases from the same broader aisle/department.

In total, we came up with 69 features. We ran Lasso Regression to identify the top features. Some of the top features were:

## Feature Importance of Top 30 Features

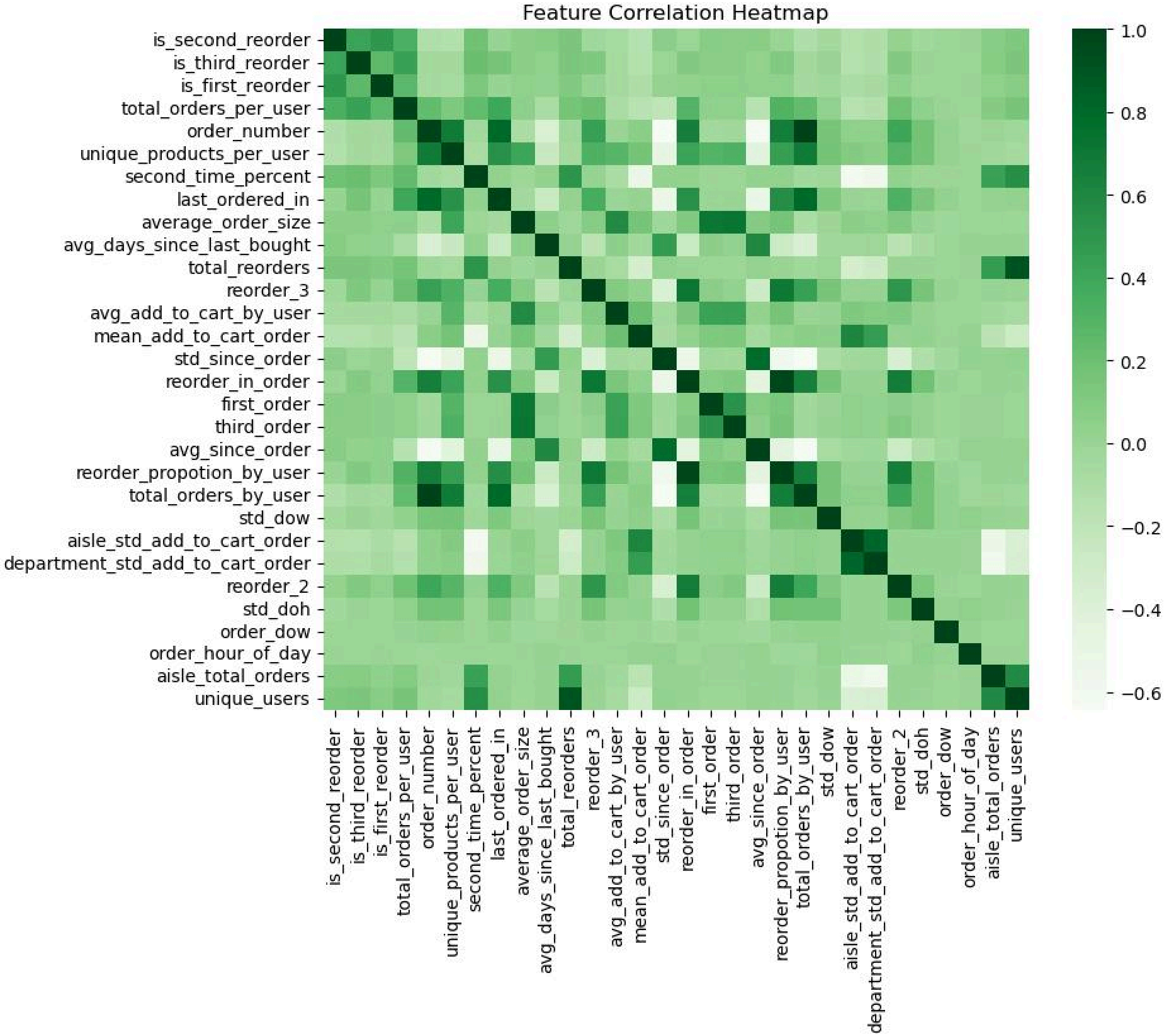


Some of the top features are:

- Whether a product was reordered in the last 3 orders
- Total number of orders a user placed
- The order index of a user
- Unique products purchased by the user
- The reorder percentage of a product
- The last time a product was ordered
- The average order size for a user

These features, as expected, characterize the user-product relationship. We plotted the correlation graph to verify that the top 30 features returned by the Lasso regression had limited correlation.

## Feature Correlation Heatmap



# Methods

## Customer Segmentation

Customer segmentation is the process of grouping customers based on common characteristics/behaviors [5]. Given our dataset of Instacart order histories for users, we can perform customer segmentation based on purchase habits as shown by frequency and categories of products purchased.

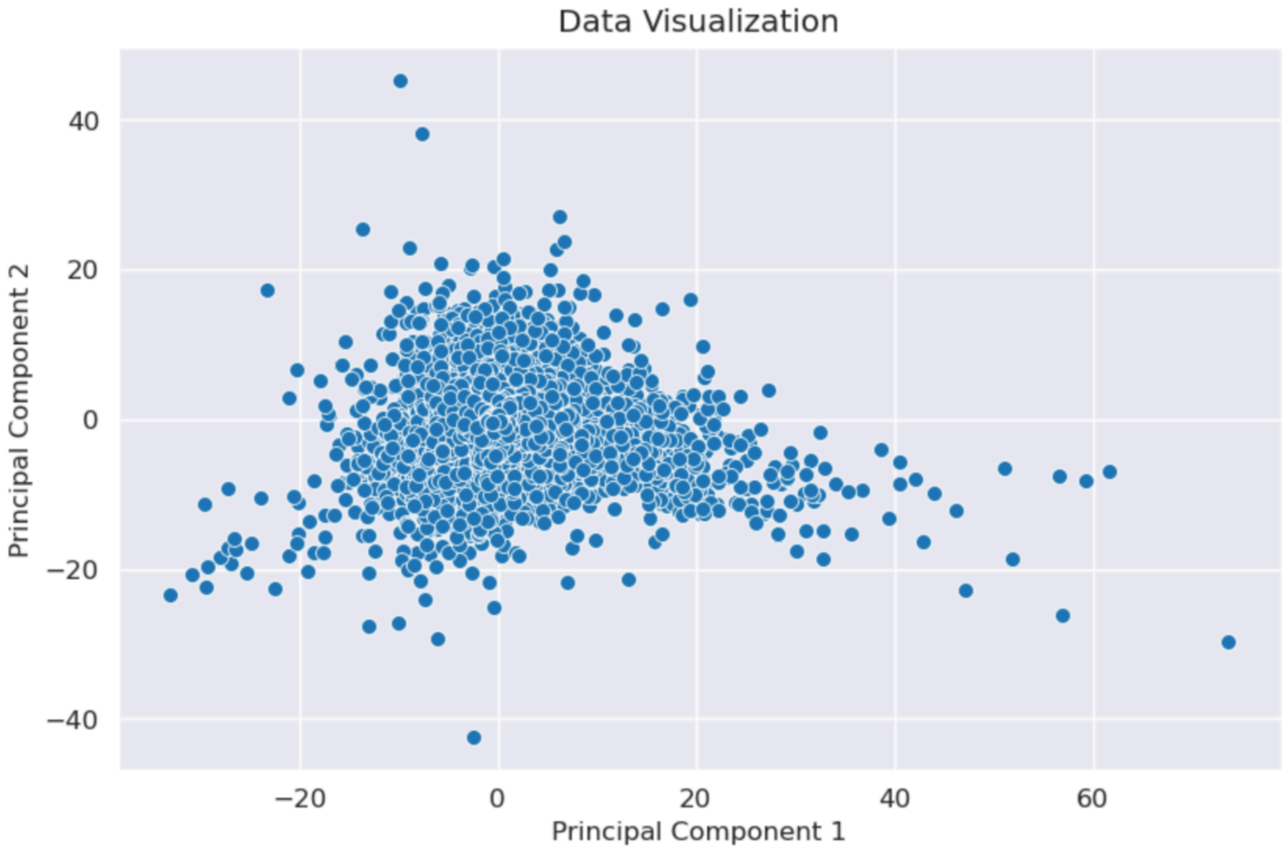
### Idea - Clustering Users Using K-Means

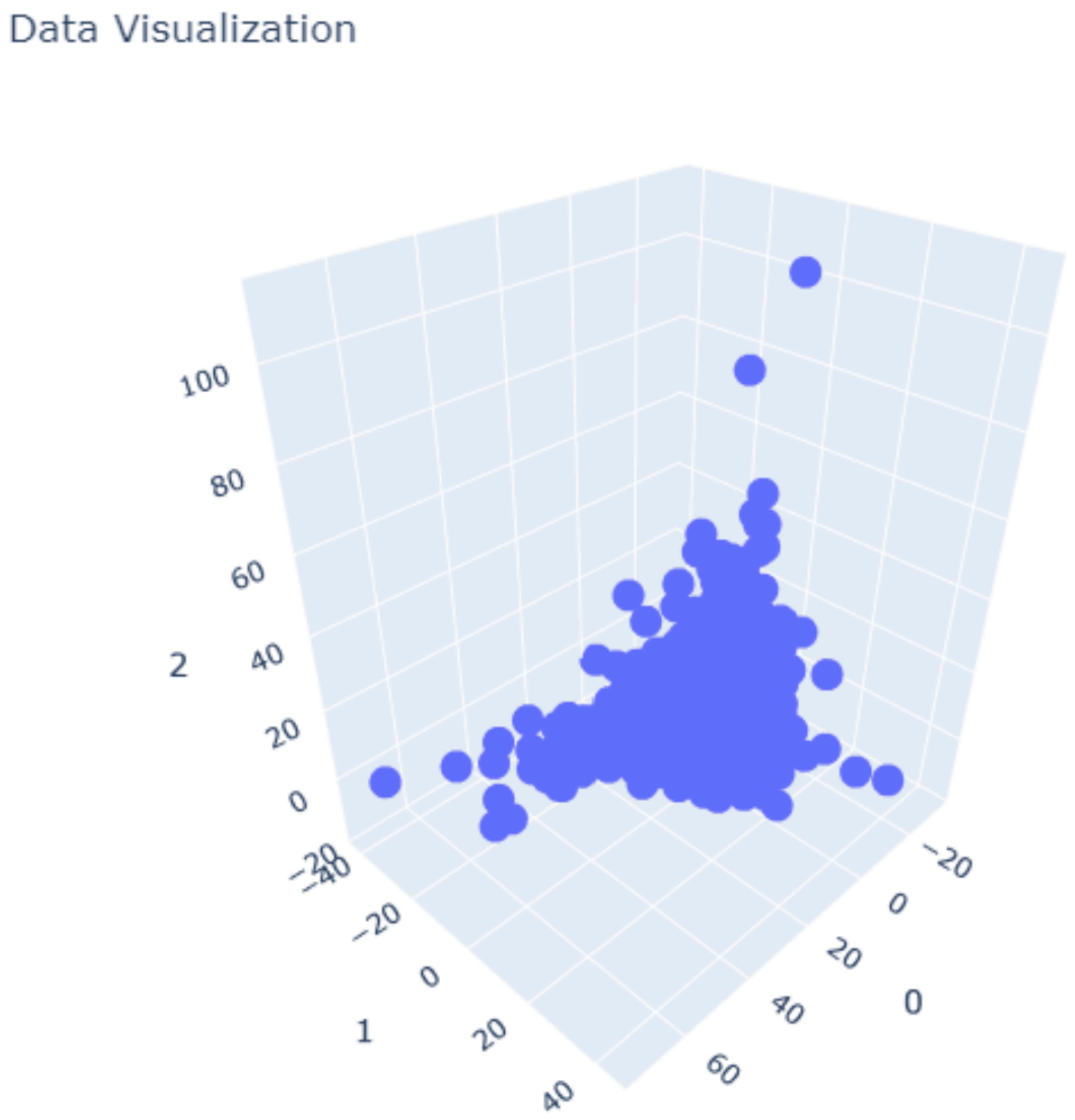
We use unsupervised learning methodologies to find inherent purchasing patterns in our dataset. Unsupervised learning algorithms are the obvious choice here, as there is no ground truth for clusters to which users might belong. An effective method to cluster similar users in our problem statement would be

K-means clustering. Due to its ease of implementation and efficient scaling for large datasets, K-means stood out as a prime choice for our use-case.

## Feature Reduction:

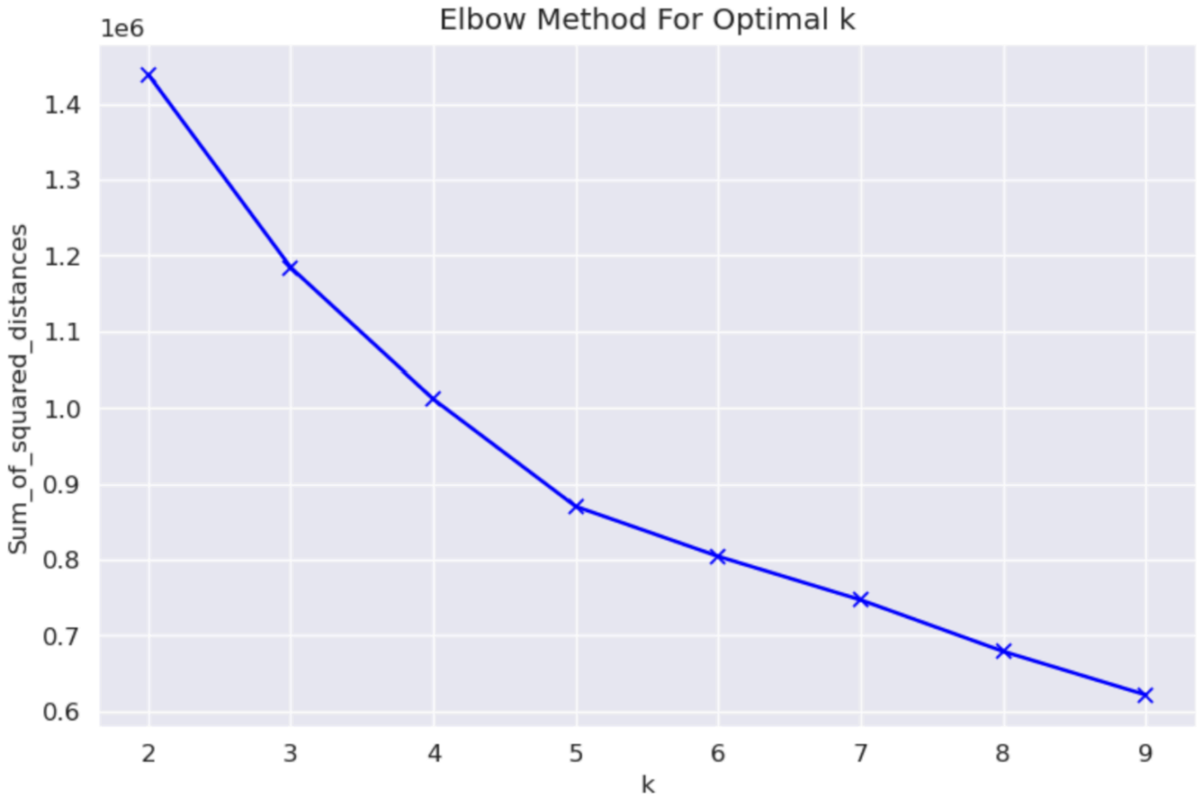
Since there are 134 product categories/aisles for each customer, this could have a negative impact on clustering results by introducing a lot of noise. There are many aisles that could be present in one-off orders, and do not add much information to cluster on. At the same time, some less frequently ordered aisles are the ones that could lead to discovering customer distinguishing characteristics. To capture such information and to reduce the impact of noisy aisles, we employ PCA for feature reduction. We focused on 5 principal components of the data which maximizes the variance and thus helping us to find meaningful clusters. In addition to fitting the data to 5 PCA components, we also discarded the first 2 components for clustering since we observed that components 3 to 5 provided the best clustering capability to the model (evaluated using Davies-Bouldin index).

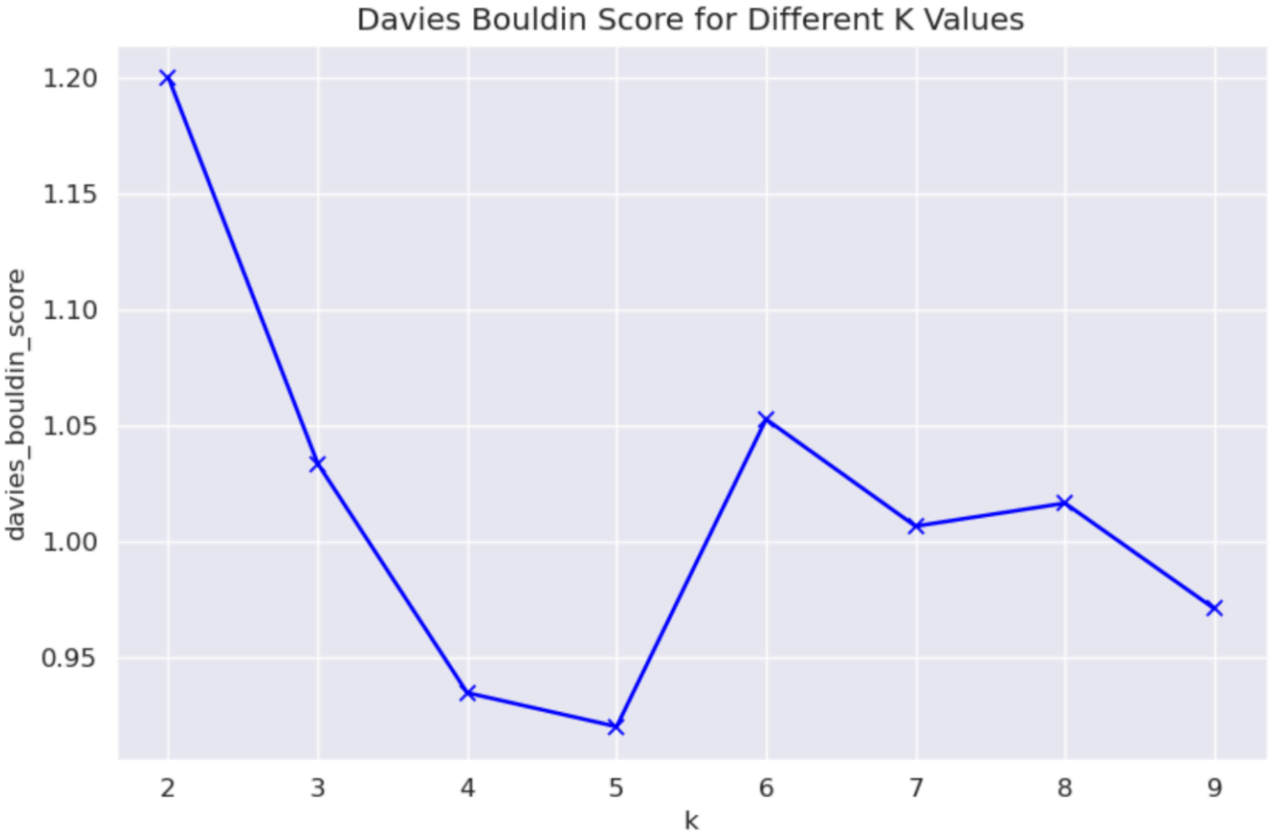




## Finding Optimal Cluster Count:

To find the optimal number of clusters in the data we used the elbow method on the sum of squared distances of the clusters for different K values and compared the Davies Bouldin (DB) Score for different K values to help disambiguate elbow points and ensure intra-cluster similarity. The plots for these 2 metrics are shown below. Based on the plots, we choose k=5 as the optimal cluster count.

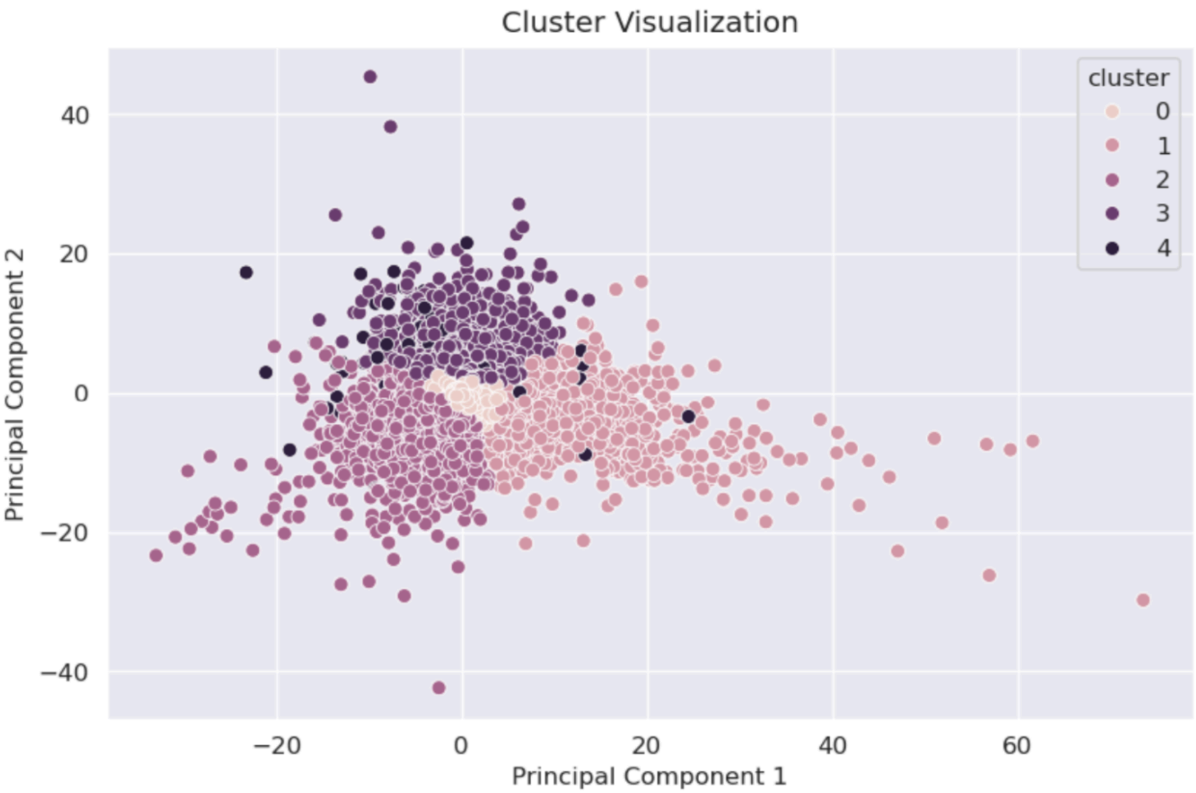


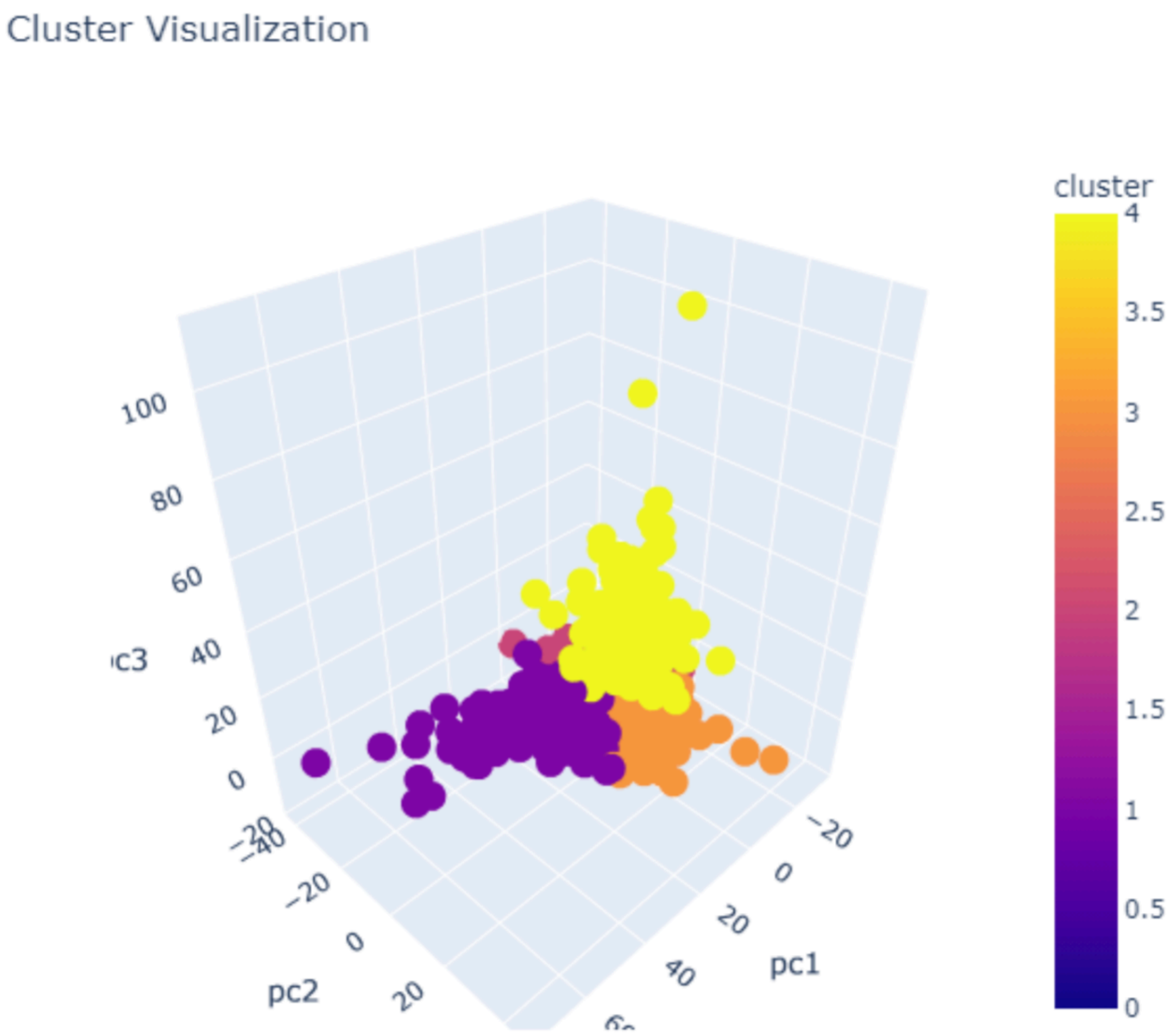


## Results:

### Visualization

The resulting 5 clusters created are visualized below.


Cluster Visualization
cluster
0
1
2
3
4
Principal Component 2
Principal Component 1
40
20
0
−20
−40
−20
0
20
40
60



Cluster Visualization

## Metrics

The above clusters resulted in the following cluster evaluation metrics

| Metric | Value |
|---|---|
| DB Score | 0.921 |
| Silhouette Score | 0.583 |

- What do these mean?

A Davies Bouldin index value below 1 suggests that the clusters are compact and well-separated [7]. With our modelling we see a score of 0.921 which seems to indicate that the formed clusters correctly identify customers with similar purchasing habbits as belonging to one cluster

When using the Silhouette Score, a clustering with an average silhouette width of over 0.7 is considered to be "strong", a value over 0.5 "reasonable" and over 0.25 "weak" [8]. With our modelling we have acheived a score of 0.583 which indicates a above reasonable clustering performance.

- Why did they perform well or poorly?

Our metrics indicate an above reasonable clustering performance for our model. We believe that the clustering results are not excellent due to the underlying nature of the use-case. For most purchases, customers tend to order the usual highly consumed food products like fruits, vegetables, milk, eggs etc. This causes our cluster formation to not have sharp and distinct cluster boundaries.

- Future Scope

Our model has performed at par on the clustering task and we believe that it can provide effective support to the reorder prediction model.

### Clustering Results and Interpretation:

The the clusters can be summarized as follows -

1. Cluster 1 - Customers belonging to this cluster usually purchase more healthy and fresh food products but the data suggests that they order less frequently than customers from other clusters
2. Cluster 2 - Customers belonging to this cluster usually purchase junk food products and packaged food items such as candies, chips pretzels, granola bars etc. indicating a relatively younger audience
3. Cluster 3 - Customers belonging to this cluster purchase healthier products more often and they tend to order more frequently than customers of any other clusters
4. Cluster 4 - Customers belonging to this cluster tend to purchase baby products and healthier food products indicating a family oriented audience with young children in the household
5. Cluster 5 - Customers belonging to this cluster tend to purchase alcoholic beverages indicating a high-earning younger audience who make more frequent leisure purchases

Another point to note is the presence of fresh fruits and vegetables in almost all clusters, indicating all customer segments order these.

# Product Reorder Prediction

The objective here is to make use of the provided data and engineered features, to predict which previously ordered item would be present in any users future cart. To achieve this, we employ machine learning techniques of Neural Networks and Gradient Boosted Treess.

## Neural Networks

### Why this model?

For the Instacart Market Basket Challenge, neural networks are a natural choice due to their ability to learn complex, non-linear relationships within high-dimensional data. Neural networks can model complex feature interactions. Predicting future purchases involves understanding intricate relationships between user behavior, product characteristics, and order patterns. As we have created relevant features in the feature engineering phase, neural networks can capture these interactions. Neural networks often outperform traditional models in complex predictive tasks. As most commonly employed models for this task are gradient boosted trees, it would be interesting to observe the performance comparisons of neural networks. By leveraging these strengths, neural networks can provide a robust framework for tackling the challenge of predicting users' next orders with high accuracy and scalability.

## Model Architecture

The neural network architecture that we ended up using is a network of 4 densely connected layers with 64, 15, 4, and 1 neurons in the 1st, 2nd, 3rd and 4th layers respectively and a total parameter count of 5,460. To introduce non-linearity into our model, we used 'relu' activation function for layers 1 through 3 and we used 'sigmoid' activation function in our final (4th) layer to get a value between 0 and 1 as the output of our model. As for the loss function, we used the 'Binary Cross Entropy' loss due to its compatibility with the binary classification objective ('1' - reorder, '0' - not a reorder). BCE is ideal to deal with continuous values between 0 and 1 (probabilities) which represents how confident the model is in its prediction. To construct our model we used tensorflow with keras wrapper. The output of our model indicates the probability of reordering a product by a customer and we used '0.7' as the threshold to classify a product as reorder. This threshold was evaluated by calculating metrics like accuracy and F1 score of the final trained model over a test set.

The architecture can be summarized as -

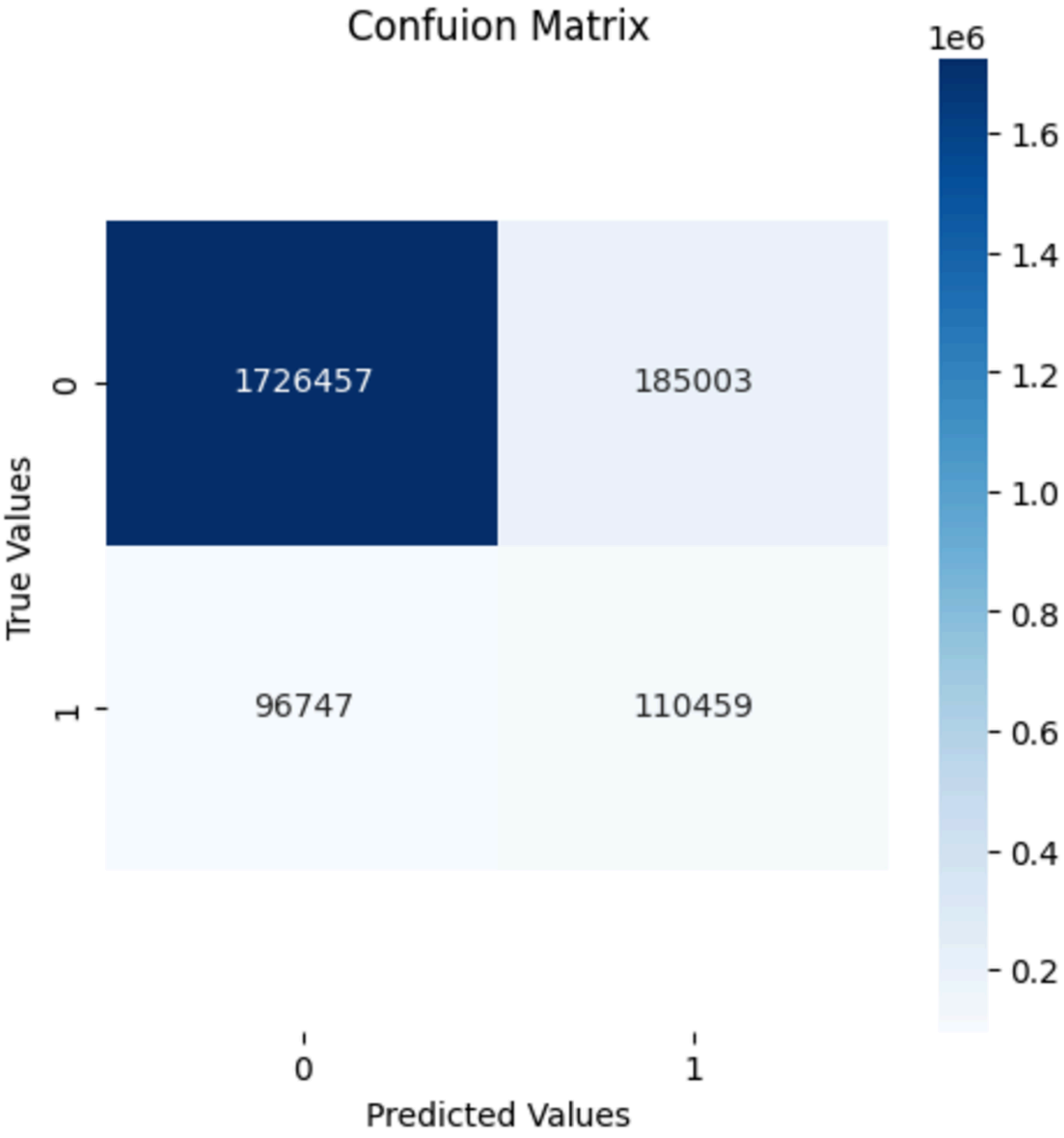
| Layer (type) | Output Shape | Param # |
|---|---|---|
| dense (Dense) | (None, 64) | 4,416 |
| dense_1 (Dense) | (None, 15) | 975 |
| dense_2 (Dense) | (None, 4) | 64 |
| dense_3 (Dense) | (None, 1) | 5 |

**Total params:** 5,460 (21.33 KB)
**Trainable params:** 5,460 (21.33 KB)
**Non-trainable params:** 0 (0.00 B)

## Results

- Accuracy - 86.7%
- F1 - 0.44

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0.0 | 0.95 | 0.90 | 0.92 | 1,911,460 |
| 1.0 | 0.37 | 0.53 | **0.44** | 207,206 |
| **Accuracy** | | | **0.87** | 2,118,666 |
| **Macro Avg** | 0.66 | 0.72 | 0.68 | 2,118,666 |
| **Weighted Avg** | 0.89 | 0.87 | 0.88 | 2,118,666 |



## Gradient Boosted Trees

### Introduction

Gradient boosted trees is a machine learning technique that uses decision trees. It is an ensemble method that performs regression or classification by combining the outputs from individual trees. The technique uses a method called boosting to build individual trees. Boosting combines simple decision trees that have very few splits i.e combines weak "learners" into a single strong learner iteratively.

## Why this model?

Gradient Boosted Trees offer identification of complex associations that are inherent to decision trees, bias reduction through sequential incorporation of the weak learners, great accuracy by fitting new trees to compensate for the errors of those already incorporated into the larger model, and faster training time on large data sets by giving precedence to those features that increase the model's predictive accuracy. This makes them an apt fit for market basket analysis problem.

## Model Details

XGBoost(Extreme Gradient Boosting) is a scalable, portable and distributed gradient boosting open source library. It has become popular due to its ability to handle large datasets and to achieve state-of-the-art performance in many machine learning tasks. We have leveraged it as part of the solution.

An important step was to define the model parameters. The parameters accepted by the algorithm include:

- **objective:** defines the type of task, say regression or classification. Since reordering is a classification problem, the 'binary:logistic' objective function was used.
- **eval_metric:** measures the difference between predicted probabilities and actual class labels. logarithmic loss (log-loss) was used to evaluate the model's performance during training.
- **colsample_bytree:** the subsample of columns when constructing each tree. It helps reduce overfitting and speeds up training by limiting the feature space for each tree.
- **subsample:** size of the random sample used for training in each iteration. This parameter introduces diversity to the training process.
- **learning_rate:** that determines how fast or slow the model will learn. A moderately low value was chosen to balance between convergence speed and preventing overfitting.
- **max_depth:** indicates the maximum depth for each tree. The value was set keeping in mind the trade off between model complexity and risk of overfitting.
- **min_child_weight:** specifies the minimum sum of instance weights( importance of a specific datapoint in the tree) required inorder to create a new leaf node. This prevents overfitting on small or noisy subsets of data.
- **alpha, lambda:** the L1, L2 regularization on weights reducing the influence of less important features.
- **gamma:** tree pruning regularizer which defines the minimum reduction in loss required to make a new split in the tree.
- **scale_pos_weight:** used to balance the positive and negative classes in imbalanced datasets. Since reorder events are often less frequent than non-reorder event, this parameters allows us to provide

more weight for positive samples.

- **n_estimators:** the number of decision trees to fit.

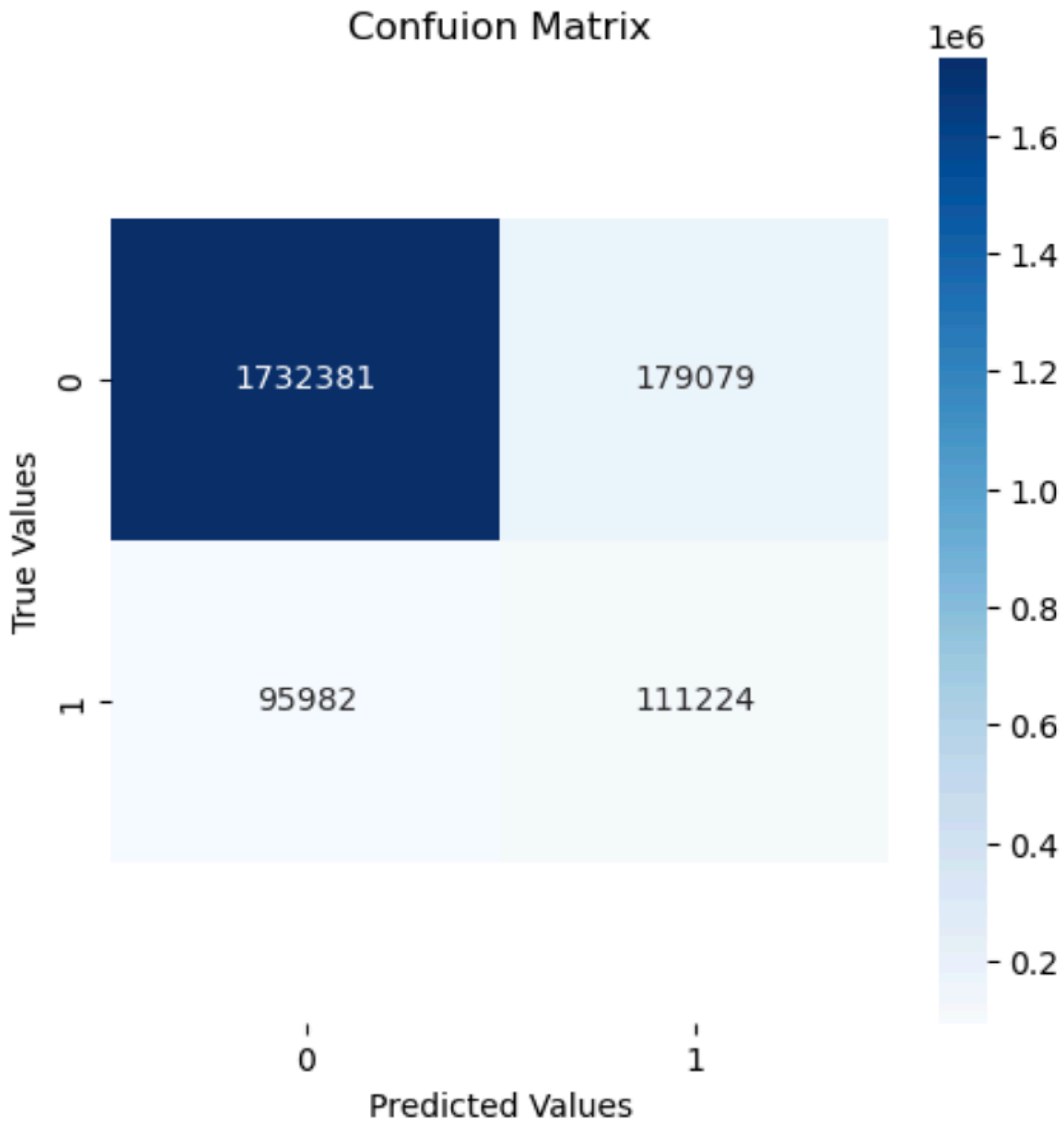
We fine tuned the parameters until we achieved acceptable performance.

## Results

### Performance

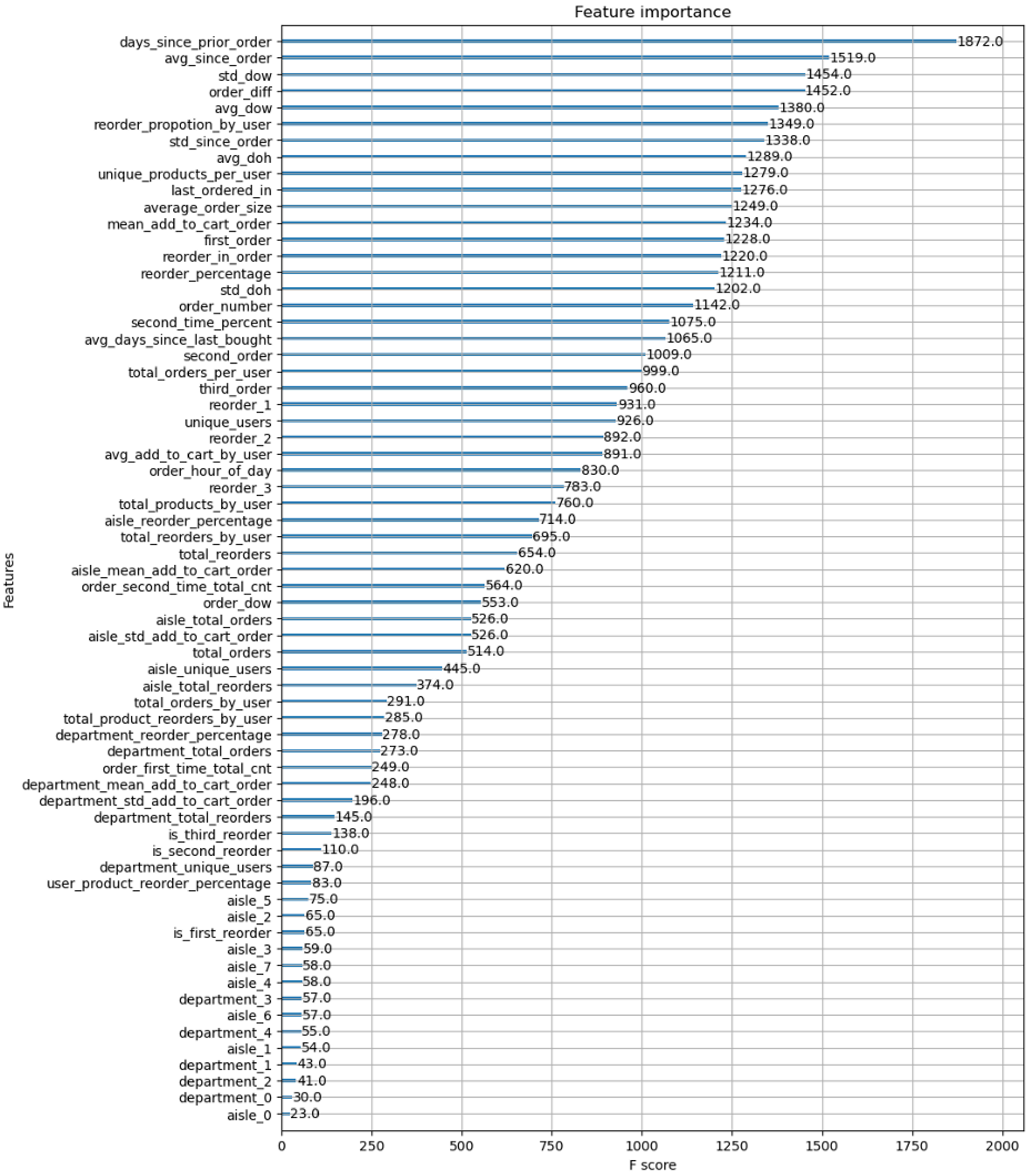
- Accuracy - 87.02%
- F1 - 0.45

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| 0.0 | 0.95 | 0.91 | 0.93 | 1,911,460 |
| 1.0 | 0.38 | 0.54 | **0.45** | 207,206 |
| **Accuracy** | | | **0.87** | 2,118,666 |
| **Macro Avg** | 0.67 | 0.72 | 0.69 | 2,118,666 |
| **Weighted Avg** | 0.89 | 0.87 | 0.88 | 2,118,666 |



**Feature Importance**

The XGBoost models also allow one to obtain the feature importances via the `feature_importances_` attribute.



Top features were days since prior order, day of week, reorder proportion for the user, etc which is inline with our expectations.

## Model Comparison

### Supervised Learning Models

**Strength of XGBoost**

- Our machine problem consists of structured/tabular dataset for which XGBoost is well suited

- While training the models we observed faster training time for XGBoost

#### Limitation of XGBoost

- Our dataset consists of more than 3 million records and XGBoost is not the ideal choice for very large dataset
- XGBoost does not scale effectively with high dimensional dataset

#### Strength of Neural Network

- Neural networks have the ability to learn hidden features/patterns which are not intuitive during feature engineering
- Our dataset consists of more than 3 million records and neural networks are an ideal choice for very large datasets

#### Limitation of Neural Network

- Neural Networks are more suited for complex data forms such as image, text, sensor data which is not the case for our problem
- While training the models we observed longer training time for neural network

### Unsupervised Learning Model

#### Strength of K-Means Clustering

- Simple implementation and testing
- Scales well to large dataset

#### Limitation of K-Means Clustering

- Pre-determining the number of clusters requires domain knowledge or experimentation
- Is prone to the curse of high dimensionality

## Understanding the Results

As can be observed from the results, both the models performed similarly according to our evaluation metrics. The Neural Network model was able to achieve a prediction accuracy of 86.7% / F1 Score of 0.44 and the XGBoost model was able to achieve a prediction accuracy of 87.02% and an F1 Score of 0.45.

### Interpretation of Results

Here it is important for us to look at the metrics for Class ‘1’ (which represents a customer re-order). Taking a look at the recall metric, the neural network model was correctly able to predict a reorder ~53% of the times and XGBoost was correctly able to predict a reorder ~54% of the times out of the total number of reorders in the test dataset. At first glance this might seem low but if we try to understand the problem use-case in depth, one might feel otherwise. For this particular machine learning problem we are trying to supplement the existing infrastructure and processes that are in-place. Even with capturing 54% of the total re-orders correctly the company may be able to benefit immensely due to the sheer volume at which they are operating.

Similarly, considering the precision for Class '1'. The neural network has a precision of ~37% and the xgboost has a precision of ~38%. This means that a large number of non-reorders are being classified as reorders by our model. In the context of this problem this is acceptable because one of our objectives is to improve customer experience and product sales by product recommendation and bundling offers. With a higher number of false positives, the customers are provided with more offers and recommendations that might be relevant but the customer had never ordered such products before. This sits well with the real world scenarios where the customers might buy previously unbought products like when cooking a dish from a new cuisine.

Overall, XGBoost performed slightly better than the Neural Network model but the results of both the models is at par with the expected results for this machine learning problem.

# Next Steps

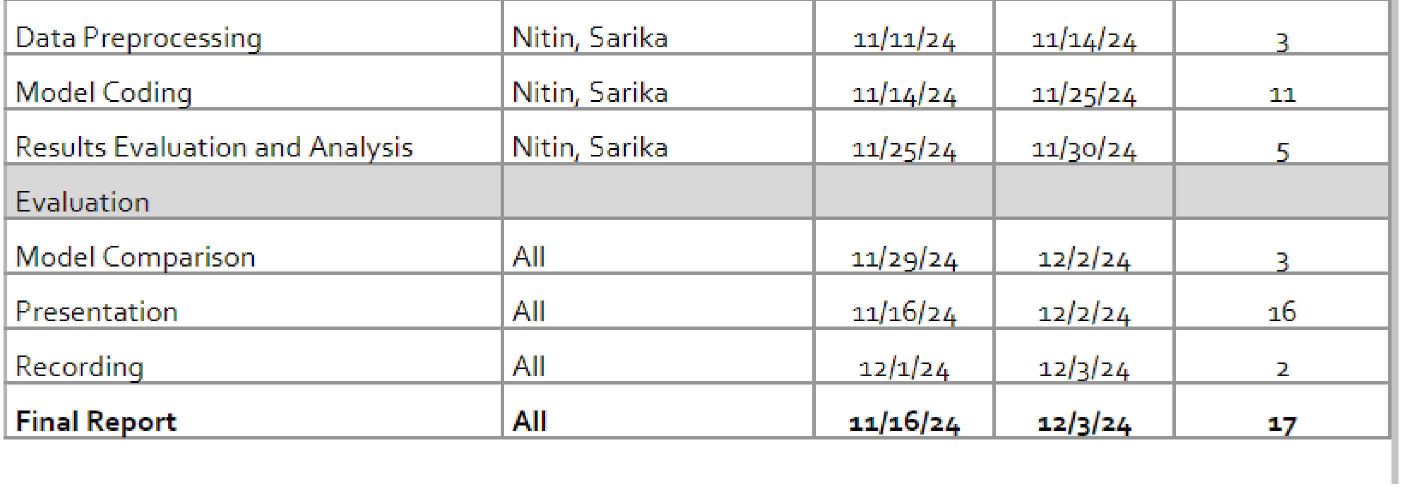
Our models provides customer segmentation, reordering prediction, and sales trend forecasting. The next logical step is to present these insights in a clear and actionable way for stakeholders by realtime sales trends visualizations, tracking customer sements and monitoring product popularity. This well-designed dashboard can provide real-time visibility into key sales trends, customer behavior, and forecasts, enabling data-driven decision-making.

# Group Gantt Chart

# GANTT CHART

| PROJECT TITLE | Predictive Analytics for Purchase Patterns |
|---|---|

| TASK TITLE | TASK OWNER | START DATE | DUE DATE | DURATION |
|---|---|---|---|---|
| Important Dates | N/A | | | |
| Proposal Due | N/A | 9/23/24 | 10/4/24 | 11 |
| Midpoint Report | N/A | 9/23/24 | 11/8/24 | 45 |
| Final Project | N/A | 9/23/24 | 12/3/24 | 70 |
| **Project Proposal** | | | | |
| Data Sourcing | Sarika | 9/23/24 | 9/27/24 | 4 |
| Project Research | Sarika, Nitin | 9/23/24 | 10/1/24 | 8 |
| Gantt Chart/Contribution Table | Sarika, Nitin | 10/1/24 | 10/3/24 | 2 |
| Proposal Write-Up | All | 10/1/24 | 10/4/24 | 3 |
| GitHub Page | Yashasvini | 9/27/24 | 10/4/24 | 7 |
| Video Recording | Aparna, Yashasvini | 10/1/24 | 10/4/24 | 3 |
| **Exploratory Data Analysis** | Benjamin | 10/7/24 | 10/14/24 | 7 |
| **Data Collection/Processing** | | | | |
| Data Cleaning | Yashasvini, Aparna | 10/7/24 | 10/14/24 | 7 |
| Data Transformation | Yashasvini, Aparna | 10/14/24 | 10/21/24 | 7 |
| Feature Engineering | Yashasvini, Aparna | 10/14/24 | 10/21/24 | 7 |
| **Clustering** | | | | |
| Model Selection | Nitin, Sarika | 10/14/24 | 10/16/24 | 2 |
| Data Preprocessing | Nitin, Sarika | 10/16/24 | 10/18/24 | 2 |
| Model Coding | Nitin, Sarika | 10/18/24 | 11/2/24 | 14 |
| Results Evaluation and Analysis | Nitin, Sarika | 11/2/24 | 11/5/24 | 3 |
| **Classification** | | | | |
| Model 1 - Classification | | | | |
| Model Selection | Yashasvini, Aparna, Benjamin | 10/14/24 | 10/19/24 | 5 |
| Data Preprocessing | Yashasvini, Aparna, Benjamin | 10/19/24 | 10/25/24 | 6 |
| Model Coding | Yashasvini, Aparna, Benjamin | 10/25/24 | 11/11/24 | 16 |
| Results Evaluation and Analysis | Yashasvini, Aparna, Benjamin | 11/11/24 | 11/16/24 | 5 |
| **Midterm Report** | All | 11/4/24 | 11/8/24 | 4 |
| Model 2 - Classification | | | | |
| Model Selection | Nitin, Sarika | 11/8/24 | 11/11/24 | 3 |



| | | | | |
|---|---|---|---|---|
| Data Preprocessing | Nitin, Sarika | 11/11/24 | 11/14/24 | 3 |
| Model Coding | Nitin, Sarika | 11/14/24 | 11/25/24 | 11 |
| Results Evaluation and Analysis | Nitin, Sarika | 11/25/24 | 11/30/24 | 5 |
| Evaluation | | | | |
| Model Comparison | All | 11/29/24 | 12/2/24 | 3 |
| Presentation | All | 11/16/24 | 12/2/24 | 16 |
| Recording | All | 12/1/24 | 12/3/24 | 2 |
| **Final Report** | **All** | **11/16/24** | **12/3/24** | **17** |

Link to Gantt Chart

# Contributions

| Name | Project Proposal | Midterm Progress | Final Submission |
|---|---|---|---|
| Aparna Manoj | Introduction<br>Presentation<br>Video Recording<br>Proposal Write-Up | Feature Engineering<br>Report Writing | XGBoost Model<br>Report Writing<br>Video Recording |
| Benjamin Jiras | Potential Results<br>Proposal Write-Up | Exploratory Data Analysis<br>Report Writing | Presentation Deck<br>Video Recording |
| Nitin Chawla | Project Research<br>Gantt Chart<br>Proposal Write-Up | K-Means Clustering<br>Report Writing | Neural Network Model<br>Report Writing |
| Sarika Singh | Gantt Chart<br>Data Sourcing<br>Project Research<br>Proposal Write-Up | K-Means Clustering<br>Report Writing | Neural Network Model<br>Report Writing<br>Video Recording |
| Yashasvini Pratyaqsha | Project Webpage Creation<br>Video Recording<br>Proposal Write-Up | Feature Engineering<br>Report Writing | XGBoost Model<br>Report Writing |

# References

[1] Swoyer, S. (2016, November 15). Beer and diapers: The impossible correlation. TDWI. https://tdwi.org/articles/2016/11/15/beer-and-diapers-impossible-correlation.aspx

[2] A. Bhargav, R. P. Mathur and M. Bhargav, "Market basket analysis using artificial neural network," International Conference for Convergence for Technology-2014, Pune, India, 2014, pp. 1-6, doi: 10.1109/I2CT.2014.7092091. https://ieeexplore.ieee.org/document/7092091.

[3] T. Lim, "K-Means Clustering-Based Market Basket Analysis: U.K. Online E-Commerce Retailer," 2021 International Conference on Information Technology (ICIT), Amman, Jordan, 2021, pp. 126-131, doi: 10.1109/ICIT52682.2021.9491703.

[4] P. Sirohi, N. Singhal, P. Kumar and M. Alam, "Leveraging knowledge of Previous Baskets to Anticipate online Buyer Behaviour," 2023 International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT), Faridabad, India, 2023, pp. 968-973, doi: 10.1109/ICAICCIT60255.2023.10465793.

[5] Massaro, Alessandro & Panarese, Antonio & Giannone, Daniele & Galiano, Angelo. (2021). Augmented Data and XGBoost Improvement for Sales Forecasting in the Large-Scale Retail Sector. Applied Sciences. 11. 7793. 10.3390/app11177793.

[6] https://www.qualtrics.com/experience-management/brand/customer-segmentation/#:~:text=Customer%20segmentation%20is%20the%20process,to%20those%20customers%20more%20effectively.

[7] https://scikit-learn.org/1.5/modules/generated/sklearn.cluster.KMeans.html

[8] https://tomron.net/2016/11/30/davies-bouldin-index/#:~:text=The%20intuition%20behind%20Davies%2DBouldin,1%2C%20lower%20score%20is%20better.

[9] https://en.wikipedia.org/wiki/Silhouette_(clustering)#:~:text=A%20clustering%20with%20an%20average,the%20distances%20become%20more%20similar.

[10] A. Bhargav, R. P. Mathur and M. Bhargav, "Market basket analysis using artificial neural network," International Conference for Convergence for Technology-2014, Pune, India, 2014, pp. 1-6, doi: 10.1109/I2CT.2014.7092091. keywords: {Neurons;Joining processes;Algorithm design and analysis;Databases;Biological neural networks;Convergence;Training;market basket analysis;feed-forward neural network;apriori algorithm},