Instacart: Market Basket Analysis

Nusrat Islam

Problem Statement

- Instacart is an online grocery pick-up and delivery service that allows customers to order groceries from participating retailers with the shopping being done by a personal shopper.
- Given the nature of this delivery service, it will be useful to know what a customer is likely to reorder to stock up and be able to deliver on time.
- This project aims at predicting the reorder items of a customer.

Data

- The data is obtained from the Kaggle competition hosted by Instacart.
- It contains anonymized transactional data of 3 Million Instacart Orders
- The Instacart data include orders of 200,000 Instacart users with each user having between 4 and 100 orders.

- It also have information about the week and hour of day the order was placed, and a relative measure of time between orders.

- There are 5 files containing information about aisles, departments, and products in the orders as well as the time of order and prior order pattern.

    - **aisles.csv** - contains aisle id and aisle
    - **departments.csv** - contains department id and department
    - **order_products_*.csv** - These files specify which products were purchased in each order. order_products__prior.csv contains previous order contents for all customers. 'reordered' indicates that the customer has a previous order that contains the product.

    - **products.csv -** product_id, product_name. aisle_id, departmanet_id

    - **orders.csv** - This file tells to which set (prior, train, test) an order belongs. You are predicting reordered items only for the test set orders. 'order_dow' is the day of week.

Data Cleaning

- Descriptive statistics was used to check for missing values
- There were no missing values except for the days_since_prior_order feature.
- The missing values belonged to the first order for each of the customers as there was no prior information.
- We replaced that value with 0.

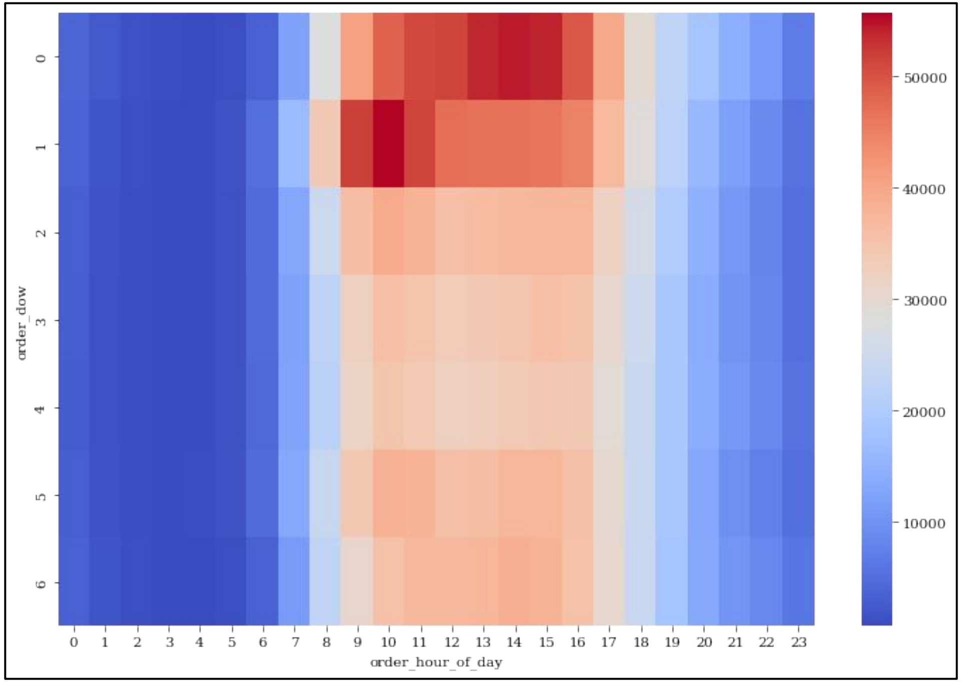
Exploratory data Analysis

- In this section we merged the aisle department and product data to see the top ordered aisles and departments.
- Personal care, dairy and beverages have the most aisles.

Treemap of aisles and departments

Ordering time

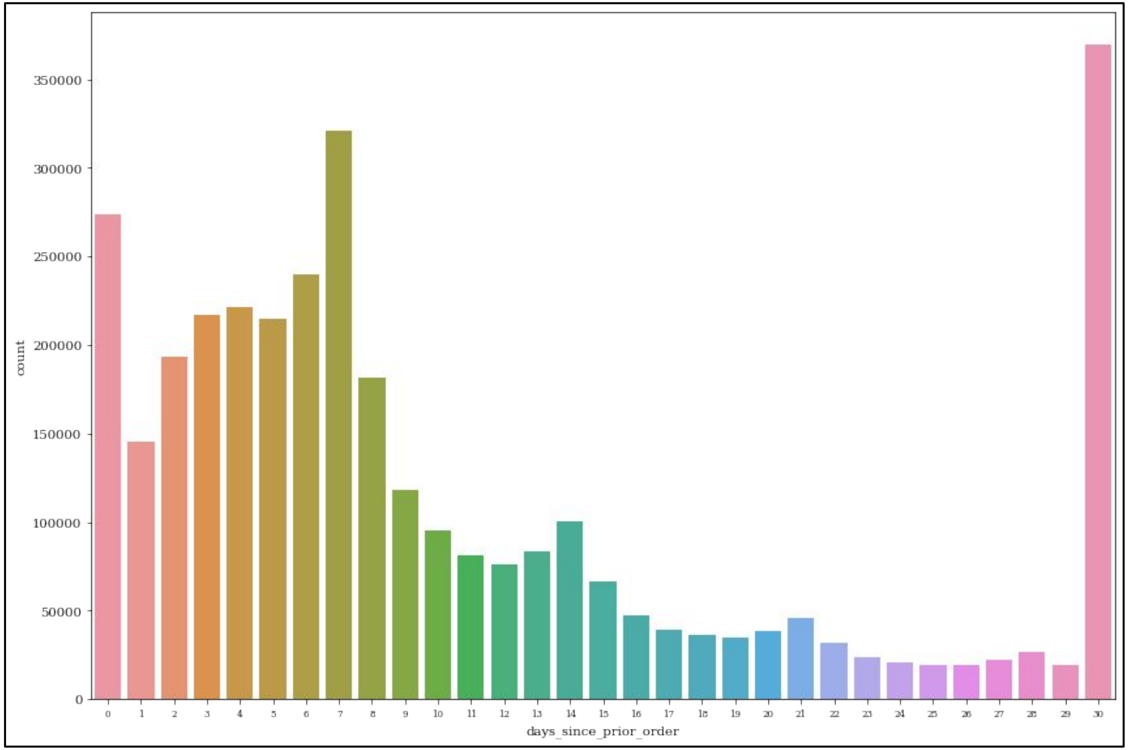
- Next we look at the times most orders are placed.
- The heatmap on the right shows order hour of day and order day of week
- We can see that people mostly ordered around the morning and afternoon during the weekends.



Heatmap showing ordering time

Reordering pattern

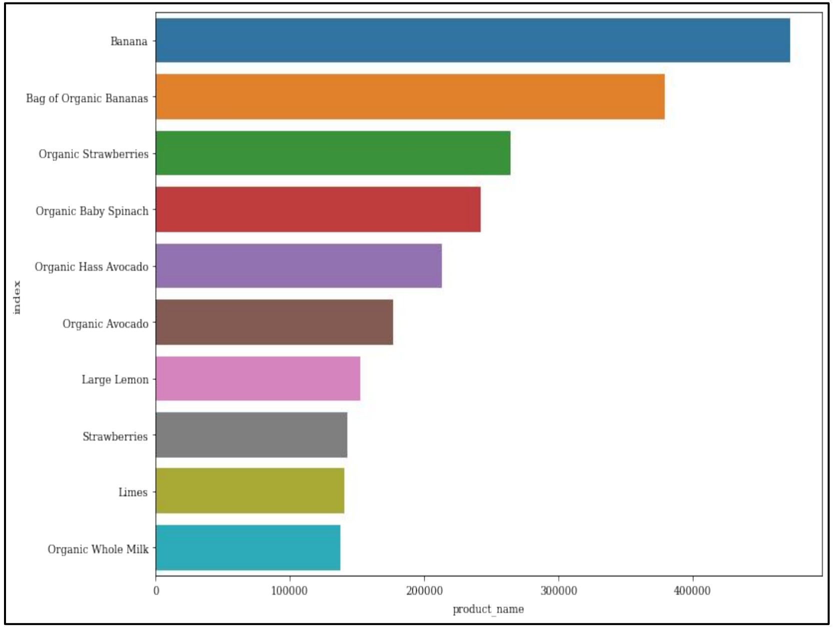
- People mostly ordered every week and also over a period of month.
- The spike around day 0 means Instacart also has a constant influx of new users.

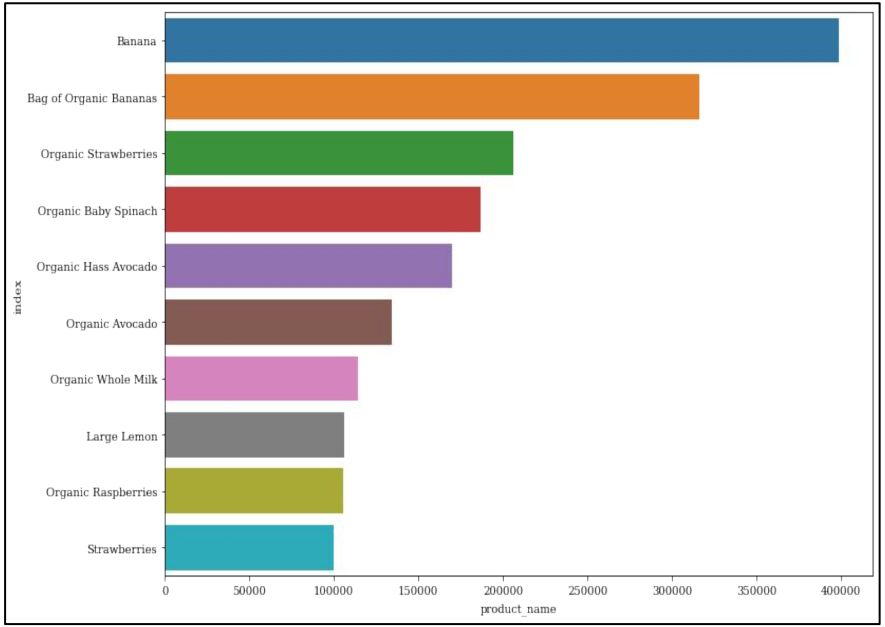


Reordering Pattern

Products ordered vs reordered

- All items on both ordered and re-ordered are from produce section

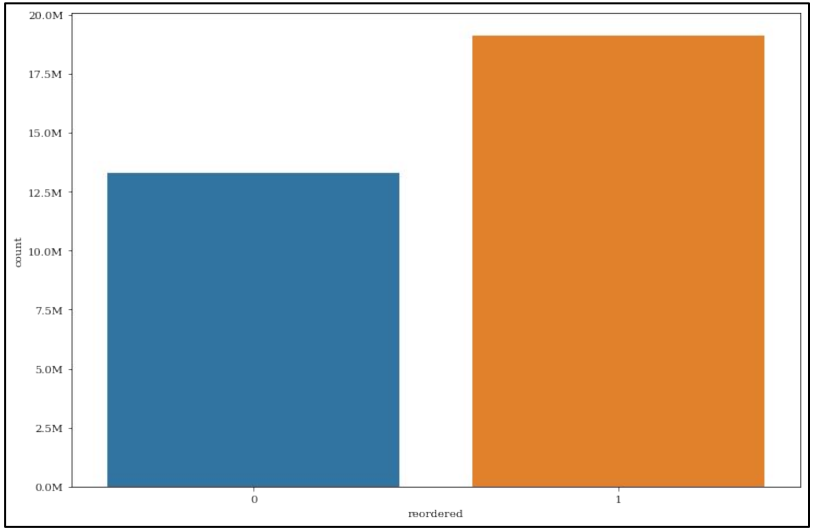




Top items ordered and reorderd

Class balance

- The last thing we looked before moving on to feature engineering was to ensure the prediction classes were balanced.



Number of products reordered*

** 0 :not reordered ; 1: reordered*

Feature engineering

- Since we want to predict whether a user will reorder a product or not we have organized the dataset to have each row represent a unique user-product behavior
- Maximum products per order was used to generate a feature average product per order per user and we named it **u_avg_prd**
- Maximum number of products in an order per user was used to generate **u_num_of_orders.**
- We calculated the day of week and the hour of day a user ordered the most and named these features **dow_most_orders_u** and **hod_most_orders_u** respectively.
- Using the information of aisle and department we also obtained **frequent_aisles** per user and **frequent_department** per user
- We then combined all these information aggregated by users into a dataframe and called it users.

Feature engineering (contd.)

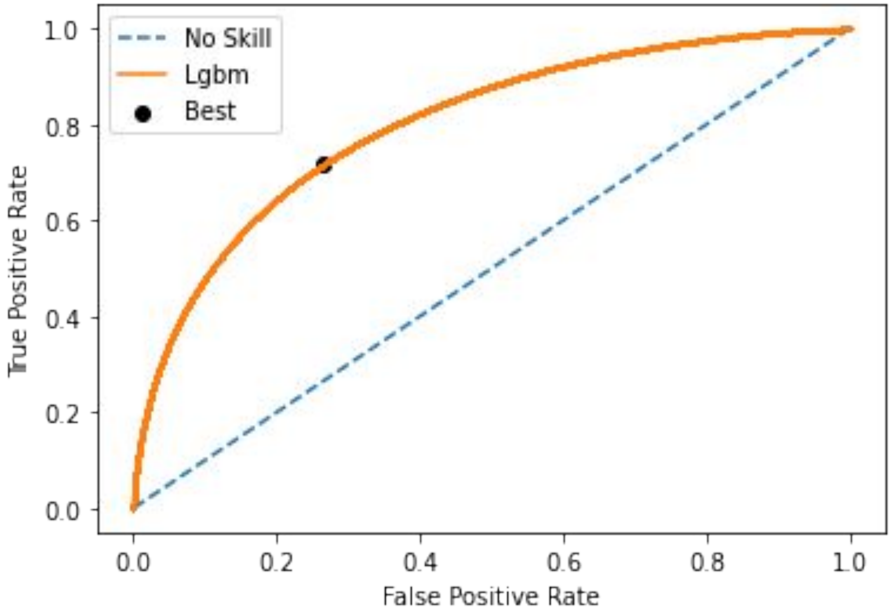
- Next we focused on the product side
- We calculated the number of times a product was ordered and named it **prd_count_p**
- We also obtained the number of times a product was re-ordered and named it **p_reorderd_ratio**
- We calculated the number of times a user has bought a particular product as well as the ratio of reordering that product. We called this dataframe **uxp**
- We then joined the **users** and **uxp** dataframe on **user_id** and **product_id** to create a new dataframe **df**
- Each row of this dataframe contains information of each user and each product based on the prior purchase behavior of the user

Algorithm and ML model

- We used a light gradient boosting classifier algorithm for this classification task.
- We used 5 fold cross validation with hyper parameter tuning.
- The parameters that were used to tune the model were:
    - max_depth:
    - bagging_fraction
    - learning rate
- The metric for the model was ‘log-loss’ and ‘roc-auc’ was used as scoring for cross-validation.

Predictions

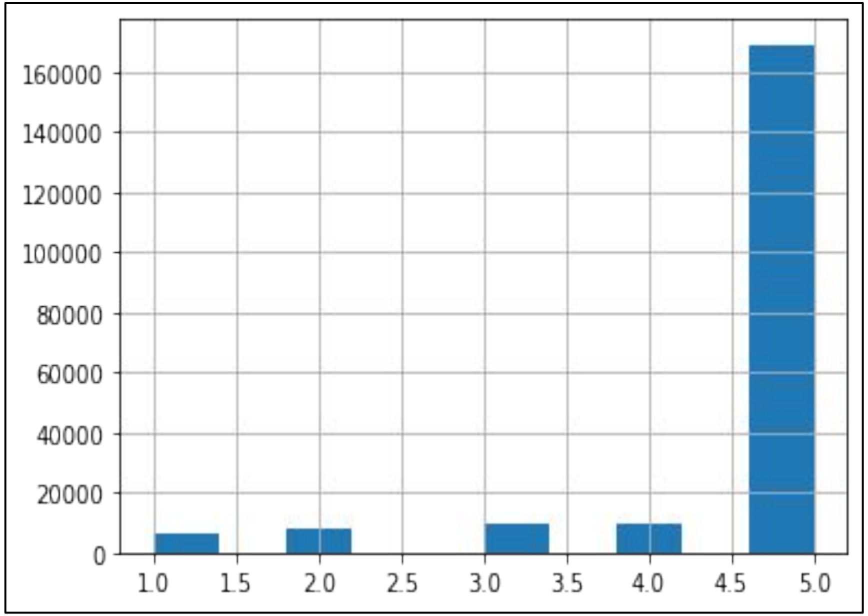
- We now have to choose a threshold to convert the probability from the model into predictions.
- We wanted to suggest at least 5 items to all the users we focus more on precision and less on recall.
- We used threshold tuning to chose a threshold that maximizes precision while having a decent recall. This point was drawn from the ROC-AUC curve and was found as 0.062



ROC-AUC curve with best threshold

Recommendations from the model

- Using this best threshold our prediction accuracy is 80%.
- We also looked at the top 5 items reordered by each user.
- More than 80% of the users would get 5 recommendations based on their prior purchase.



No. of recommendations per user

Conclusion

- Using the user transaction history and times of ordering we were able to develop a model that is capable of providing top 5 product recommendations based on prior order with 80% accuracy
- For new users the demographics of users could be used to make suggestions.
- The main trade-off for our model was the threshold we chose.
- Since we wanted to maximize precision we did not focus on whether all the top 5 suggestions were likely to be reordered by users.
- For further improvements, we can run an A/B test with users getting top 5 recommendations based on a threshold that maximizes precision vs top 5 recommendations that maximizes recall. We could use this test to then evaluate which recommendation model provides generate more revenue.
- Also, a further step could be added to filter products that were suggested to an user but they did not buy.