ISDS 577

Master of Science Capstone Seminar

INSTACART RECOMMENDATION SYSTEM

FALL 2019

GUIDED BY

Dr. DANIEL SOPER

GROUP MEMBERS

AJAY YADAV

DEEPAM KHIRWAR

HARSHITA DIDWANIA

PRIYANKA KUNDER

SAILEE MENE

INDEX

1. Introduction and Project Overview:

Instacart, in essence, is a grocery delivery startup. They facilitate deliveries of on-demand grocery orders to homes within an hour in major US cities. As of June 2017, the company valuation is at around $3.4 billion. Food habits indeed define who you are, whether you plan to buy your staples from the carefully designed shopping lists or let your fancy decide your food choices. Instacart is essentially an app that helps you order groceries and get it delivered to your doorstep and intents to make it easy and convenient to stock your fridge and pantry with your unique favorites and staples when you need them. A crowdsourced marketplace model has been implemented by InstaCart similar to that of Uber or Lyft. The Instacart shopping experience is as follows: First, the user places his order via the app. Later, an order notification goes to the locally crowdsourced "shopper," after which he goes to a close-by store, buys the groceries, and delivers them to the user. There are three basic ways that Instacart generates revenue: delivery fees, membership fees, and mark-ups on in-store prices.

The data science team of Instacart plays a crucial role by providing its consumers with a pleasant shopping experience. Currently, they use transactional data to develop models that predict which products the user plans to reorder, try for the first time, or add to their cart during the next session. In 2017, the company announced its first public dataset release, which is incognito and contains a sample of around 3 million grocery orders from more than 200,000 Instacart users. For improving the overall shopping experience for the customers and making it more customized, the company expects to predict future orders of customers, namely which products a user will most likely buy next time in his or her future order. By

making such kind of predictions over a particular group of users, the company will be able to have a considerable understanding of the demand for its merchandise, thus appropriately adjusting for resupply level, eventually improving the customer's shopping experience. This dataset of groceries' data is perfect for market basket analysis.

The primary objectives of our project are:

- To use the anonymized data on the customer orders over time to build a model that will help us to predict what products will be reordered for each customer.
- Interpret and summarize the results to predict the overall demand for each product.

We have used Python along with Numpy, Pandas, and matplotlib libraries to load, explore, manipulate, and visualize the data.

2. Data Dictionary:

This dataset is a relational set of files describing customer orders' over the period. They are undisclosed and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users. For each user, it provided a sequence of orders purchased in the project, week and hour of the day when the order was placed, and a relative measure of time between the orders.

A total of six datasets were imported. Following sections will explore each dataset in details-

orders (3.4m rows, 206k users):

- order_id: order identifier
- user_id: customer identifier
- eval_set: which evaluation set this order belongs in (see SET described below)

- order_number: the order sequence number for this user (1 = first, n = nth)
- order_dow: the day of the week the order was placed on
- order_hour_of_day: the hour of the day the order was placed on

products (50k rows):

- product_id: product identifier
- product_name: name of the product
- aisle_id: foreign key
- department_id: foreign key

aisles (134 rows):

- aisle_id: aisle identifier
- aisle: the name of the aisle

departments (21 rows):

- department_id: department identifier
- department: the name of the department

order_products__SET (30m+ rows):

- order_id: foreign key
- product_id: foreign key
- add_to_cart_order: order in which each product was added to cart
- reordered: 1 if this user has ordered this product in the past, 0 otherwise

where SET is one of the four following evaluation sets (eval_set in orders):

- "prior": orders prior to that user's most recent order (~3.2m orders)
- "train": training data supplied to participants (~131k orders)
- "test": test data reserved for machine learning competitions (~75k orders.

2.1. Understanding dataset:

→ **Aisles:** There are 134 aisles in this dataset. Few sample names of the aisles are: ‘instant foods,’ ‘prepared soup salads,’ ‘other’ ‘specialty cheeses.’

→ **Department:** There are 21 departments in this dataset. Names of all departments are listed below alphabetically ordered-

"alcohol, babies, bakery, beverages, breakfast, bulk, canned goods, dairy eggs, deli, dry goods pasta, frozen, household, international, meat seafood, missing, other, pantry, personal care, pets, produce, snacks."

→ **Products:** There are around 49,688 products in the catalog within 134 aisles and 21 departments. Sample products are as shown below-

product_id	product_name	aisle_id	department_id
1	Chocolate sandwich cookies	61	19
2	All Seasons Salt	104	13
3	Green Chile Anytime Sauce	5	13

→ **Departments and their related products:** Products data frame is connected to Departments. Samples are as shown below:

Departments	three_examples_products
Beverages	Peach Iced tea Mix packets/Cayenne Lemonade/Sparkling Water
Alcohol	Bud Light Beer Cans/ Jamaican Style Lagger
International	Organic Ramen Noodles/Lo Mein Egg Noodles/Borscht

→ **Aisles and its related products:** The product dataframe is also related to its aisles. Each aisle consists of several products. When you join both aisles and product data frame, we get an idea regarding what type of products are available for each aisle. Shown below is the sample for three products for a few aisles.

Aisles	Three_examples_products
Frozen Dessert	Naked Coconut Organic Coconut Bliss|| Smooth and Creamy original Cream Cheesecake
Ice cream toppings	The Ultimate Caramel Sauce|| Classic Waffle Cones|| Gluten Free sugar Cones
Food Storage	Oven Bags|| Linkin Bag Clip||No Stick Heavy Duty Foil

→ **Orders:** There are over 3 million observations in the orders dataset. Each row represents a unique order.

1. Train Eval_Set
2. Test Eval_Set

→ **Order_products:** Each order contains several products that are ordered by the user. Instacart had classified the orders into 'train' and 'prior' in the SINGLE order dataset. However, the details of each order are split into two datasets:

- order_product_train: contains only details of product items of last order
- order_product_prior: includes details of product items of all prior orders

→ **Users:** There is no dedicated data frame for users. However, we can derive the number of unique users from the data frame from the order data frame. On grouping user_id and eval_set column, we found that there are around 75,000 test users and 131,209 train users.

2.2. Detailed Data description:

The table below shows us the detailed description of all the variables that we are focussing on-

Variable Name	Data Type	Variable Description
order_id	Num	Order Identification Number
product_id	Num	Product Identification Number
user_id	Num	User Identification Number
department_id	Num	Department Identification Number
aisle_id	Num	Aisle Identification Number
order_num	Num	Number of orders within each purchase
order_dow	Num	Order day of week
order_hour_of_day	Num	Order hour of the day
days_since_prior_order	Num	Days since the last order was made
product_name	Char	Name of product
aisle_name	Char	Name of aisles where product is placed
department_name	Char	Name of department that the aisle is in
add_to_cart_order	Num	Sequence of product placed into cart by each customer
reordered	Num	Dummy variable to indicate if the product is reordered or not.

3. Research Questions:

The dataset being utilized is related to the factors related to grocery shopping. It has multiple variables, and these variables form different columns in the dataset, and every column has either a numerical or categorical value. Despite the dissimilar data types, all these variables have an impact on the grocery shopping.

Some of the variables have a much greater effect on customer behavior, while others have less. Depending upon the impact of the factors, the user behavior fluctuates. Hence the project aims to understand how the different factors affect the shopping patterns of an individual and, more importantly, develop appropriate models to recommend products to the user. Below are the questions which we are going to address in this project:

i. Which department/Aisle's product has the highest order rates and name of the highest selling products?

   Deli is the department with the highest reorder ratio and the name of the highest selling product is Raw Veggie Wrapper.

	product_id	p_total_purchases	p_reorder_ratio	product_name	aisle_id	department_id
6430	6433	68	0.941176	Raw Veggie Wrappers	13	20

   Also, the highest selling product is banana which is under department 'Produce' was sold a total of 472565 times.

	product_id	p_total_purchases
24848	24852	472565

	product_id	product_name	aisle_id	department_id
24851	24852	Banana	24	4

ii. To predict which products are likely to be in a user's next order based on their previous purchased products.

We have used the LGBM, we have also submitted a Submission file which has all the order_id listed, and the train dataset has 75000 orders.

iii. What days and hours of the week are the most order placed? Which products are bought mainly during these peak hours? Is there any pattern?

The most orders are placed during the weekends, Saturdays and Sundays. The peak hours when the most orders are placed is between 8 AM to 5 PM.

4. Data Preprocessing:

4.1. Feature Engineering

The data set out by Instacart has more than 200,000 unique users. The number of orders for each user ranges from 4 to 100 with the orders in the dataset labeled as prior, train, or test.

The prior orders refer to the *past* behavior or activity of a user while the train and test orders refer to the *future* behavior or activity that we seek to predict. We have the results of the train orders revealed by Instacart, but we do not have such information for the test orders. From the prior orders, we know which products a user has purchased. We also know the order_id for the future orders of each user. Now, our goal is to find out which of those products will be in a user's future order.

Hence, we need to predict whether each pair of user and a product is a reorder or not which makes it a classification problem. This is indicated by a reorder variable, 1 if it is reordered or 0 when not reordered. Below is a sample in which,

user_id - Instacart users

product_id - Products in the prior orders

order_id - Future order(train/ test)

reordered - What we need to predict

user_id	product_id	order_id	reordered
1	196	1187899	1
1	10258	1187899	1
1	10326	1187899	0
2	17122	2125869	1
2	25133	2125869	0

We need to calculate predictor variables(X) that will help us in describing the characteristics of a product and the behavior of a user. This will be achieved by analyzing the prior orders

of the dataset. Then, train users will be used to create a predictive model, and test users will be used to make our prediction. The reordered column is our response variable(Y)

user_id , product_id - Primary key(products from prior orders)

order_id - Future orders

X - Predictor variables (based on prior orders)

Y - Reordered(Response Variable)

user_id	product_id	X	X	X	eval_set	order_id	Y
1	196				train	1187899	1
1	10258				train	1187899	1
1	10326				train	1187899	0
2	17122				test	2125869	
2	25133				test	2125869	

Now, we start our analysis by merging dataframes, orders and order_products_prior and saving it as a dataframe op. It is to note that the dataframe 'op' will only contain prior orders as the order_products_prior contains only prior orders.

Create Predictor Variables

1. User Predictors

   It describes the behavior of a user, such as the total number of orders placed. Here, our aim is to find; the numbers of orders per customer and how frequent a customer has reordered products.

   Number of orders per customer can be achieved by grouping *user_id* and *order_number* and using .max() function in the op dataframe we created in the last step. The result is shown below:

   user

	user_id	order_number
0	1	10
1	2	14
2	3	12
3	4	5
4	5	4

To find how frequent a customer has reordered products,

$$\text{Probability reordered(user\_id)} = \frac{\text{total number of reorders}}{\text{total number of purchased products}}$$

We will first group *user_id* and *reordered* column and use the .mean() function to get the result. Below is the result:

u_reorder

	user_id	u_reordered_ratio
0	1	0.694915
1	2	0.476923
2	3	0.625000
3	4	0.055556
4	5	0.378378

This u_reorder data frame is merged with the user data frame by performing a left join to keep all these desired features and saving it as a new user data frame.

2. Product Predictors

   Here, we will aim to find the number of purchases for each product, and the probability of each product to be reordered. To calculate the number of purchases for each product, we group product_id and order_id columns in our op data frame and finally use the .count() function.

   prd

	product_id	p_total_purchases
0	1	1852
1	2	90
2	3	277
3	4	329
4	5	15

To find the probability of a product to be reordered, use

$$\text{Probability reordered(product\_id)} = \frac{\text{number of reorders}}{\text{total number of orders}}$$

In the next steps, we

i. Remove products with less than 40 purchases , by first creating groups of each product and then use filter function to keep only those groups of more than 40 rows.
ii. Group the products and calculating the mean of reorders. The product_id and the p_reorder_ratio are the two main columns.
iii. We merge these features with the prd dataframe through a left join.
iv. Fill NaN values with zero. These are due to the fact that we used left join with the prd dataframe and so all the rows that had products with less than 40 purchases will get NaN values.

The dataframe that we achieve after performing above steps will look as following

	product_id	p_total_purchases	p_reorder_ratio
0	1	1852	0.613391
1	2	90	0.133333
2	3	277	0.732852
3	4	329	0.446809
4	5	15	0.000000

3. User Product Predictors

Here we have found the following:

a. How many times a user bought a product
b. How frequently a user bought a product after its first purchase
c. Number of times a customer bought a product on their last 5 orders

I. The solution to this, groupby user_id and product_id. With .count() function, how many times each user bought a product. The result is saved in uxp dataframe

	user_id	product_id	uxp_total_bought
0	1	196	10
1	1	10258	9
2	1	10326	1
3	1	12427	10
4	1	13032	3

II. In order to find how frequently a user bought a product after its first purchase, we use the following approach,

$$\text{Probability reordered(user\_id, product\_id)} = \frac{\text{Times bought N}}{\text{Order range D}}$$

Times bought N - total numbers of order of each user

Order range D - total number of orders since the order was first placed for a product

Order range D is created using following:

- Total orders - Total number of orders of each user
- First order number - The order number where a user bought a product for the first time

Calculating the numerator Times bought N

This can be calculated simply by using a .groupby() function on user_id and product_id and counting the instances of order_id. The result is saved as times; below is the result

		Times_Bought_N
user_id	product_id	
	196	10
	10258	9
1	10326	1
	12427	10
	13032	3

Calculating the denominator Order Range D

Here, we first calculate the total numbers of the order of each customer, and we groupby() using user_id and order_number with the max() function. The result is total_orders attached below:

	total_orders
user_id	
1	10
2	14
3	12
4	5
5	4

Now to calculate the order number where a user bought a product for the first time, we groupby() *user_id* and *product_id* while retrieving the .min() value from the *order_number* column. The result is stored as first_order_no

	user_id	product_id	first_order_number
0	1	196	1
1	1	10258	2
2	1	10326	5
3	1	12427	1
4	1	13032	2

Next, we merge the first_order_no with the total_orders data frame using a right join and save the result as 'span' attached below:

	user_id	total_orders	product_id	first_order_number
0	1	10	196	1
1	1	10	10258	2
2	1	10	10326	5
3	1	10	12427	1
4	1	10	13032	2

The Order range D can now be created using some operations on the columns of the span data frame, such as span total orders - span first order + 1. Below is the attached screenshot:

	user_id	total_orders	product_id	first_order_number	Order_Range_D
0	1	10	196	1	10
1	1	10	10258	2	9
2	1	10	10326	5	6
3	1	10	12427	1	10
4	1	10	13032	2	9

Now, we merge times data frame which contains the numerator with the span data frame which includes the denominator and save the result as uxp_ratio.

	user_id	product_id	Times_Bought_N	total_orders	first_order_number	Order_Range_D
0	1	196	10	10	1	10
1	1	10258	9	10	2	9
2	1	10326	1	10	5	6
3	1	12427	10	10	1	10
4	1	13032	3	10	2	9

Now, we need to divide N and D. To do that, we set up a new column called uxp_reorder_ratio, which is Times_Bought_N / Order_Range_D. We only intend to keep user_id, product_id, and our new feature uxp_reorder_ratio while dropping other columns. Lastly, we merge the uxp_ratio data frame with the uxp data frame to get the final result attached below:

	user_id	product_id	uxp_total_bought	uxp_reorder_ratio
0	1	196	10	1.000000
1	1	10258	9	1.000000
2	1	10326	1	0.166667
3	1	12427	10	1.000000
4	1	13032	3	0.333333

III. Here, we want to keep the last five orders of a customer. To achieve that, we will first create order_number_back, a new variable that holds the order number for each order in reverse count. Then, we have kept only the last five orders. The next step is to perform the groupby functions on user and product to find how many times users had bought a product. Finally, we create the ratio

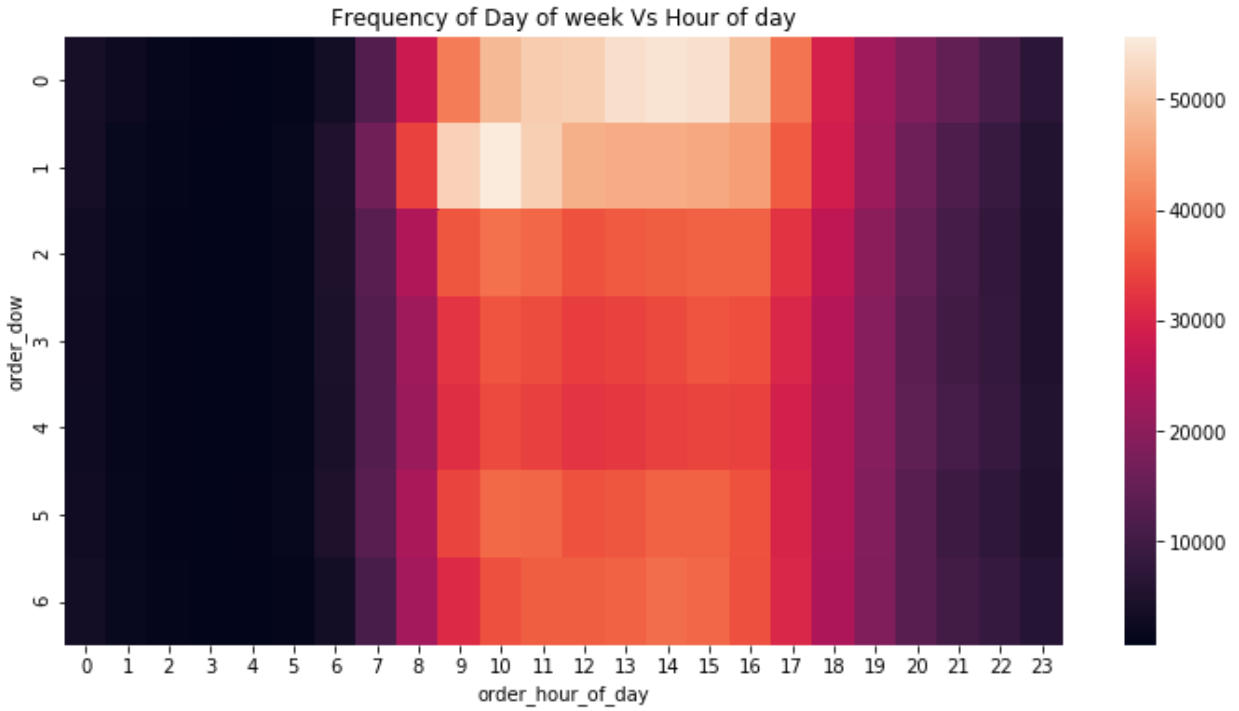
5. Exploratory Data Analysis:

Exploratory data analysis shall help us to understand the buying behavior of customers with some questions-

- What do people buy, and what do they typically reorder?
- When do they buy (day and time)? Is there a buying trend, and does that influence what they buy?

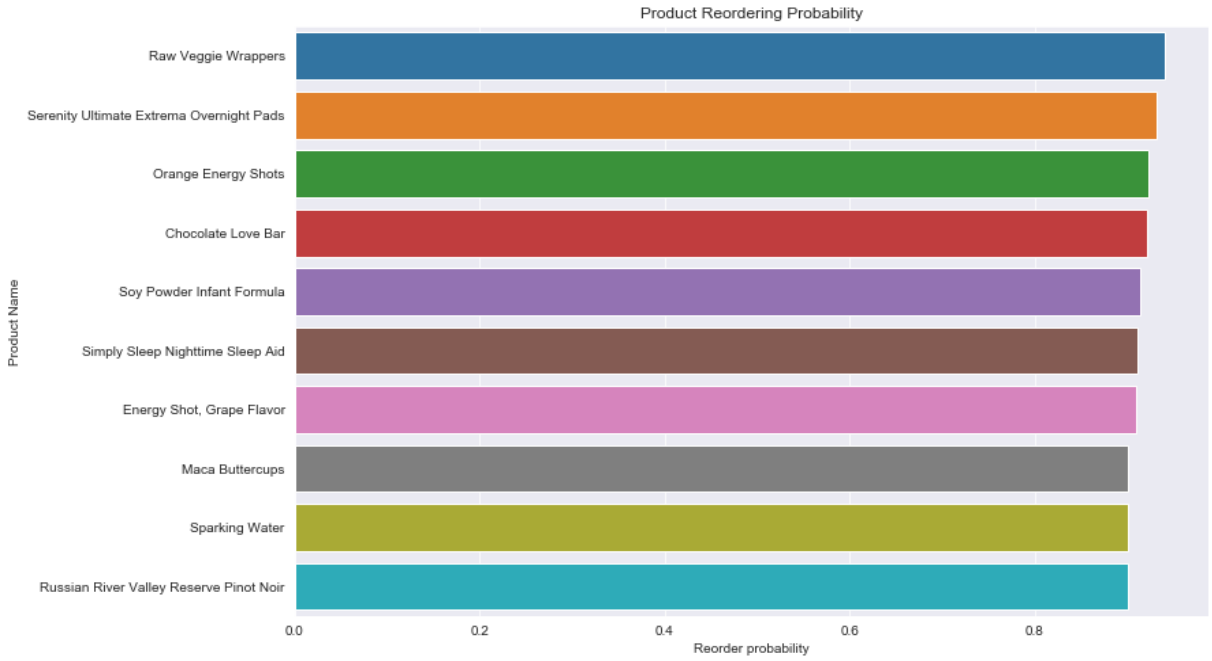
→ **Frequency of Day of week Vs. Hour of day and Frequency of order by hour of the day**

The heatmap below shows the day-to-day, hourly frequency for the orders. We can see that, on days 0 and 1, Saturday and Sunday respectively, we have a high rate of orders from 8 AM till 5 PM.



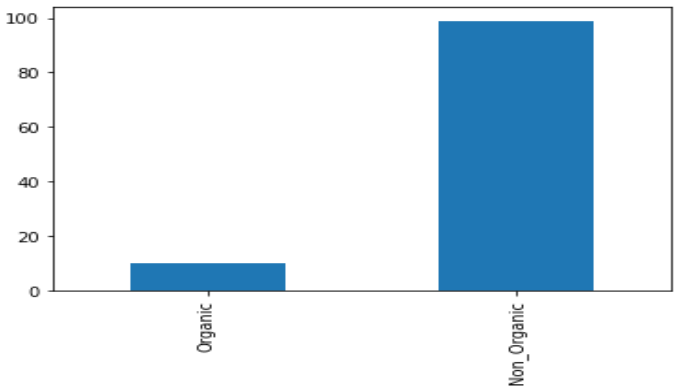
→ Number of orders for a product Vs. Probability of reordering.

The following graph shows the products with their reordering probability. The chart only displays the top ten products that have been ordered multiple times, and which might have a likelihood of being reordered.



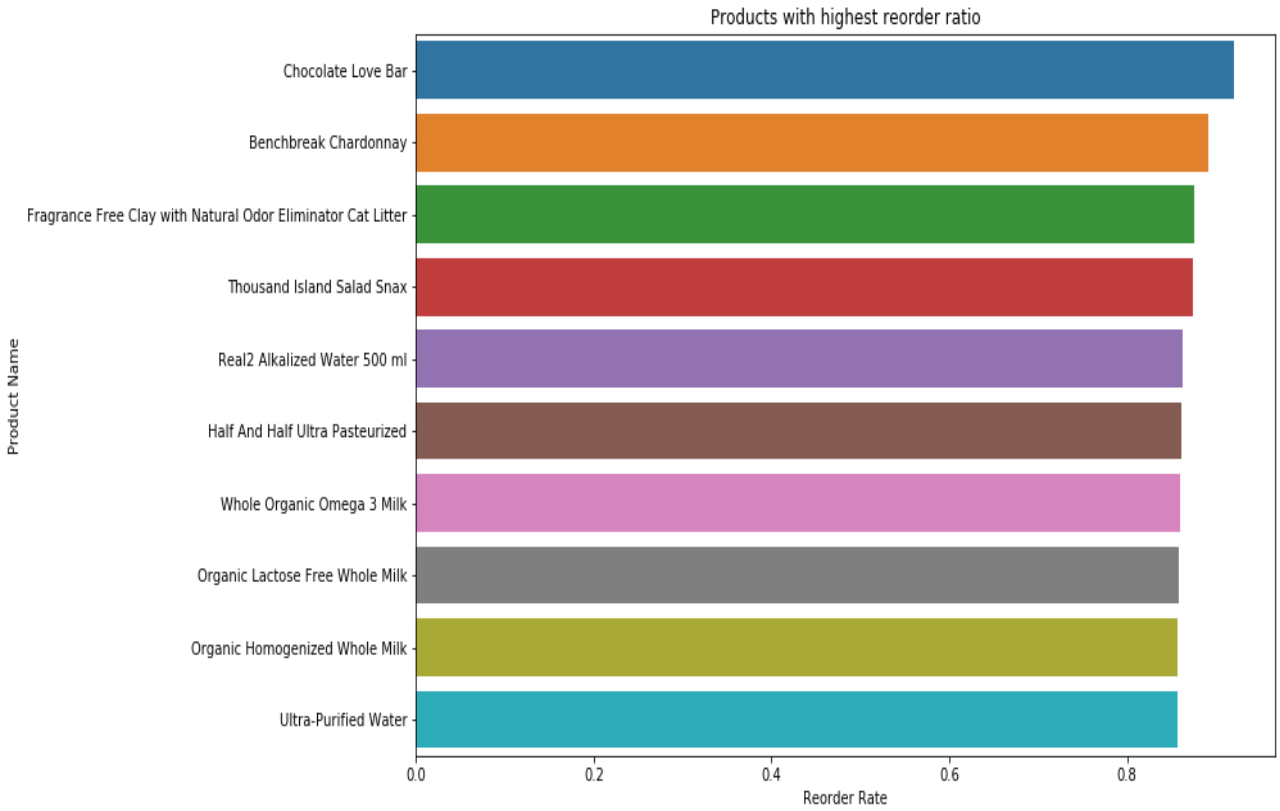
→ Percentage of organic Vs. non-organic products being ordered.

The bar plot below shows the ratio between the organic and non-organic products being ordered. Clearly, organic products are ordered just 10% of the time, whereas non-organic products are mostly being ordered.



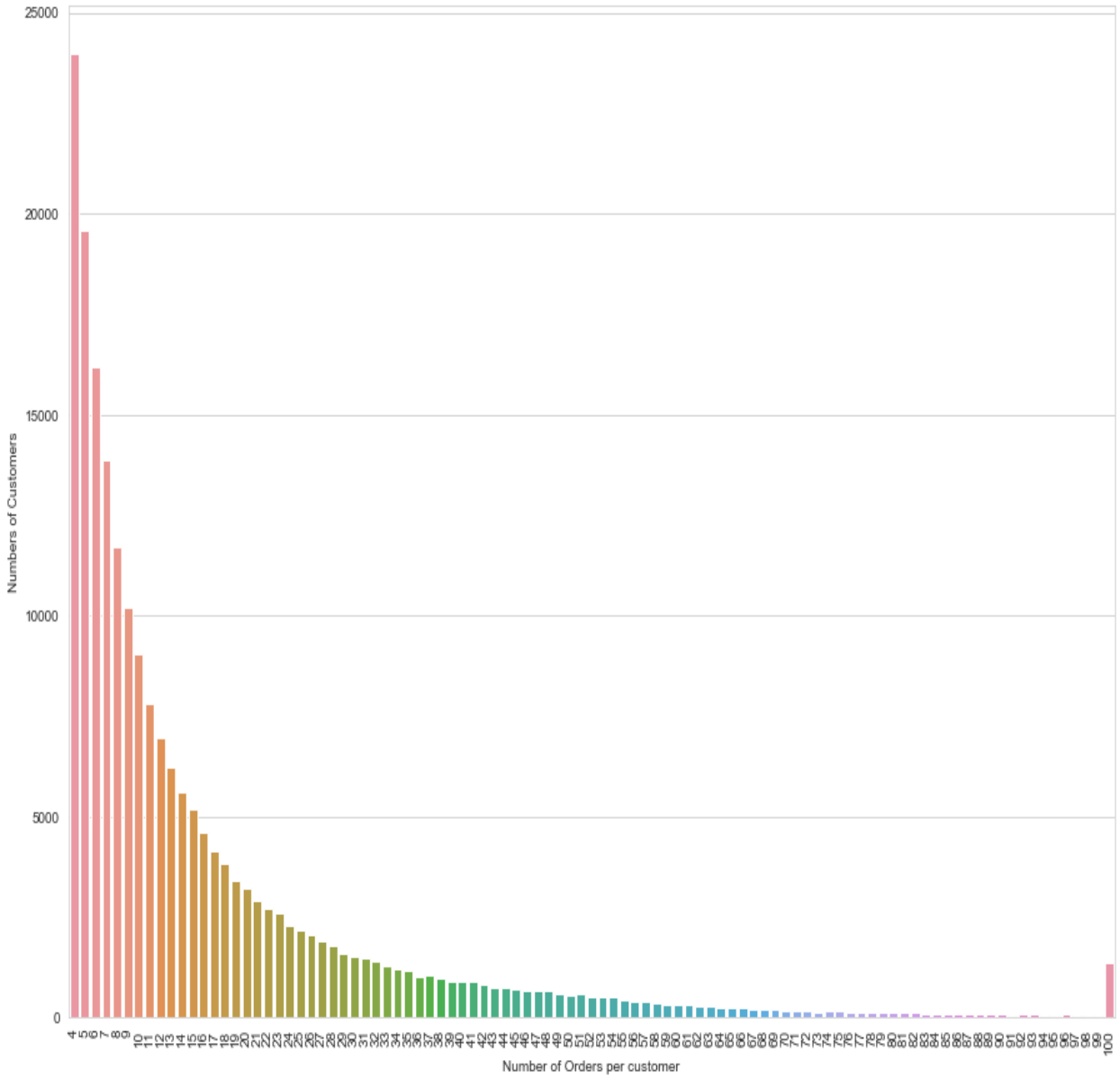
→ Most Reordered products Vs. Reorder probability.

From the graph, it's observed that the Chocolate Love Bar has the highest reorder ratio amongst all.



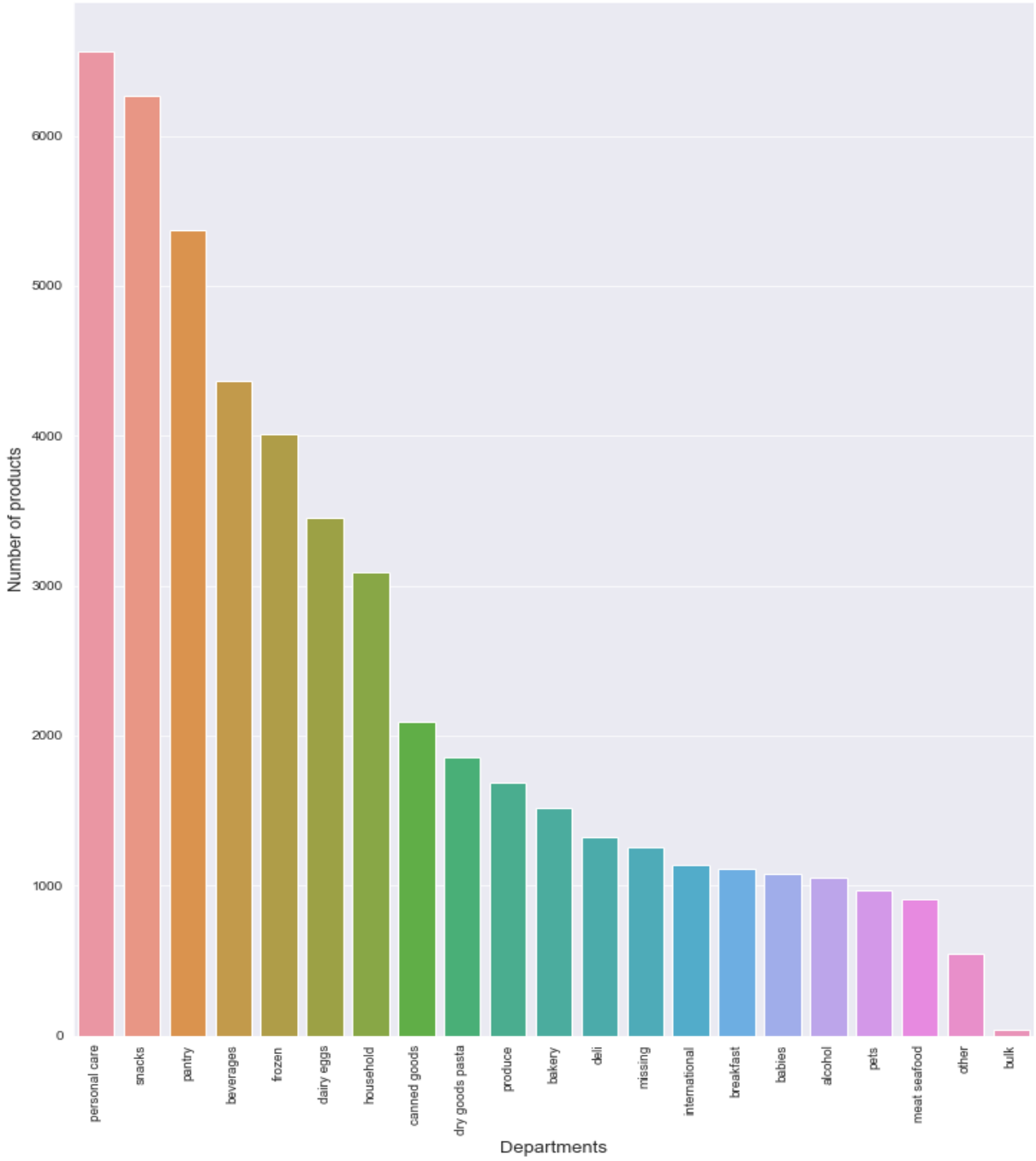
→ Number of customers Vs. Number of orders per customer.

Most of the users made few orders. The number of orders made by the user decreases significantly, along with the order numbers. On observing the graph, we can see that the maximum orders any user has made are 100.



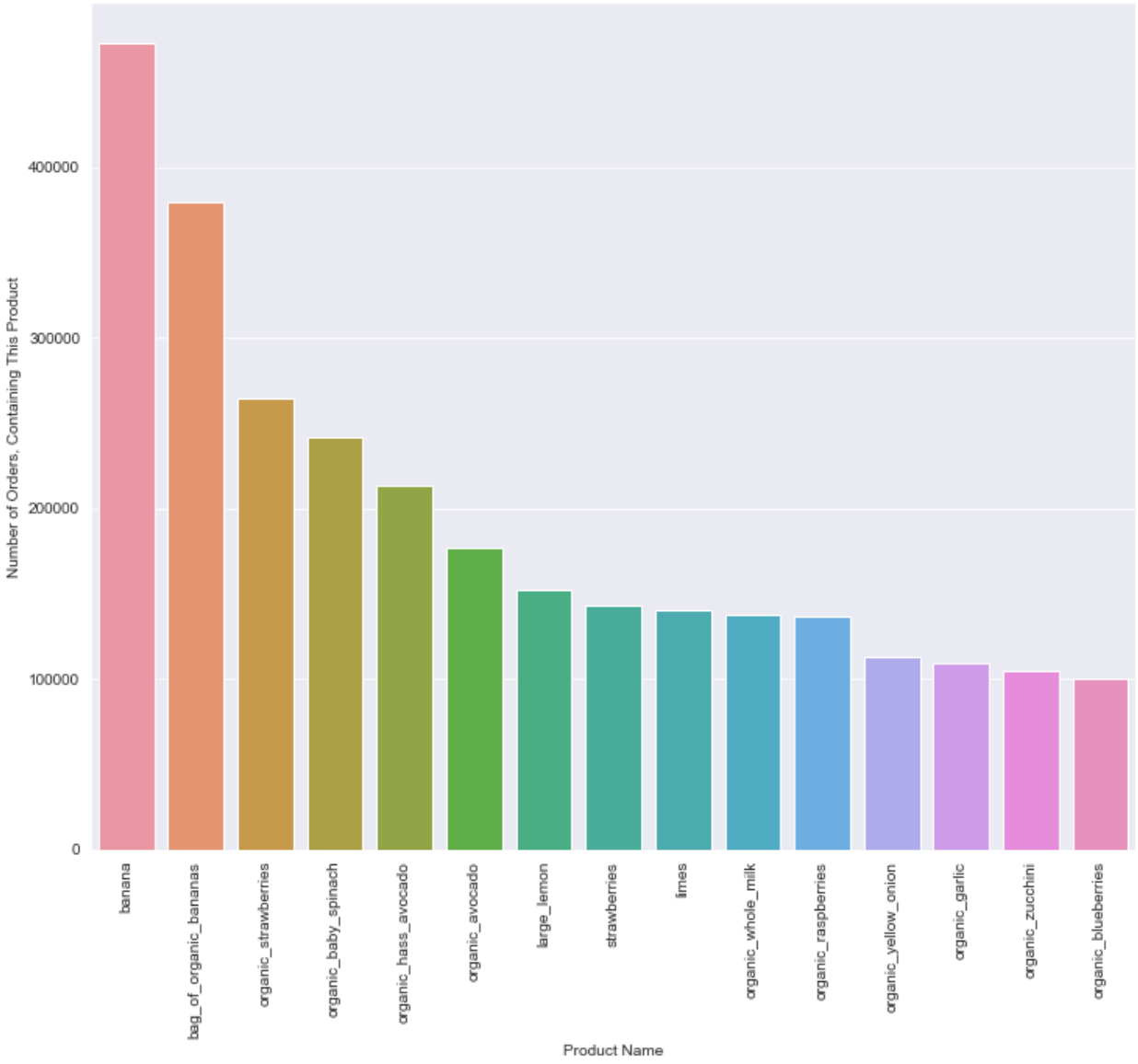
→ Departments Vs number of products available in each department.

The following graph shows the number of products available in each department. The highest availability of products is in the personal care department, followed by the snacks department.



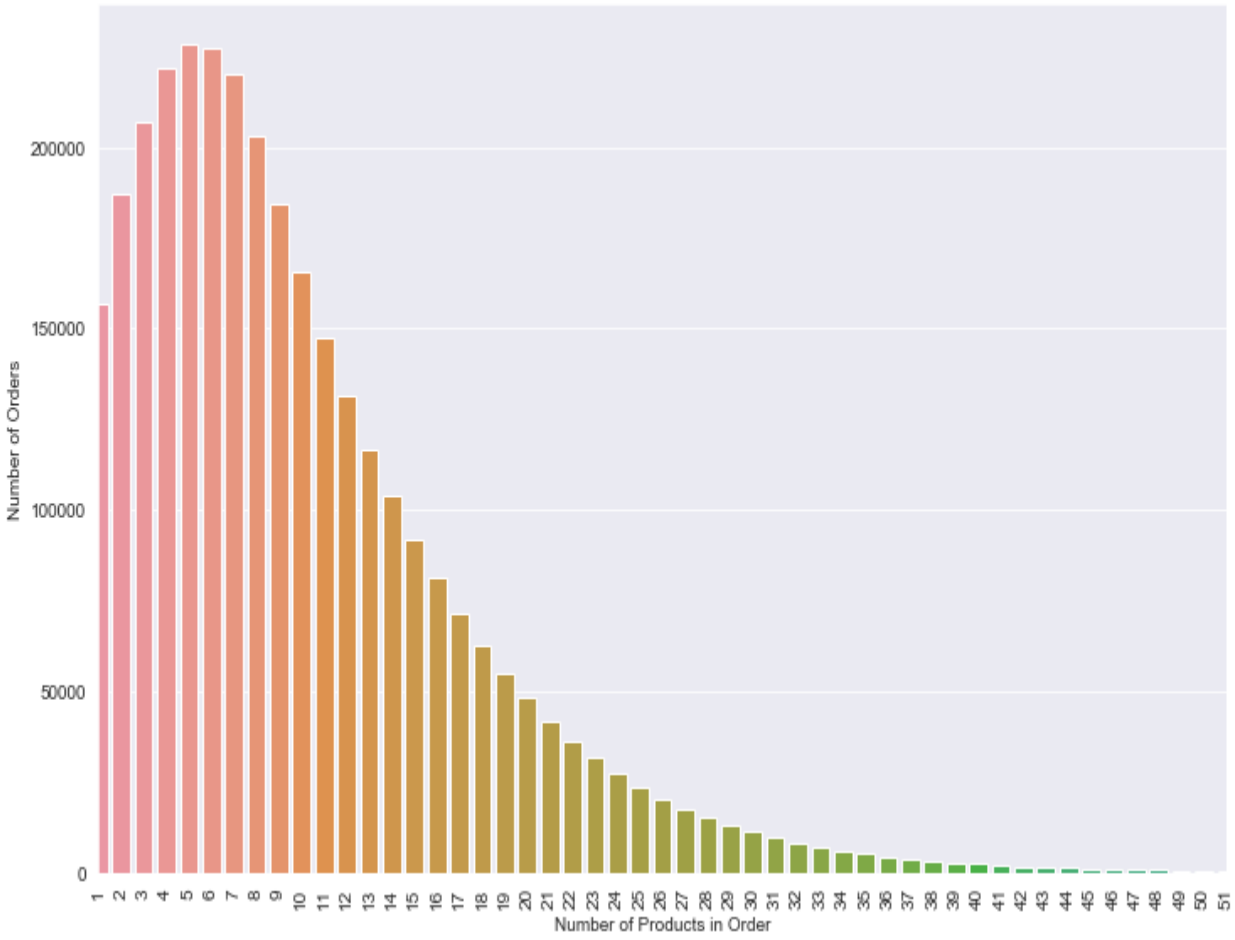
→ Most frequently ordered products Vs. Number of orders containing these products.

From the graph, we can see that Banana tops the list. The number of orders differs quite a lot for different products.



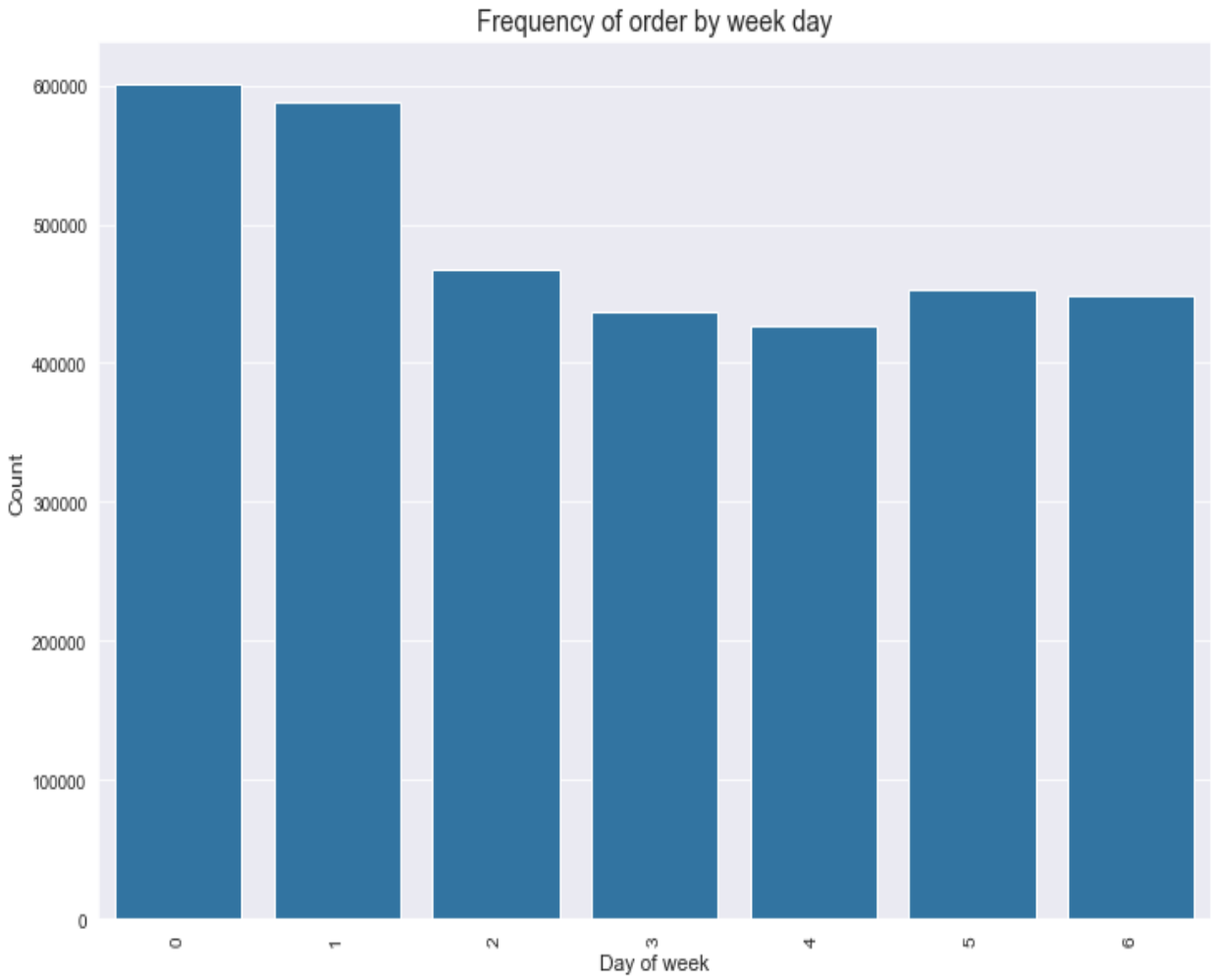
→ Number of Orders Vs Number of products in an order.

On observing the graph, we see that the average basket size is five, i.e., most of the orders contain typically five products.



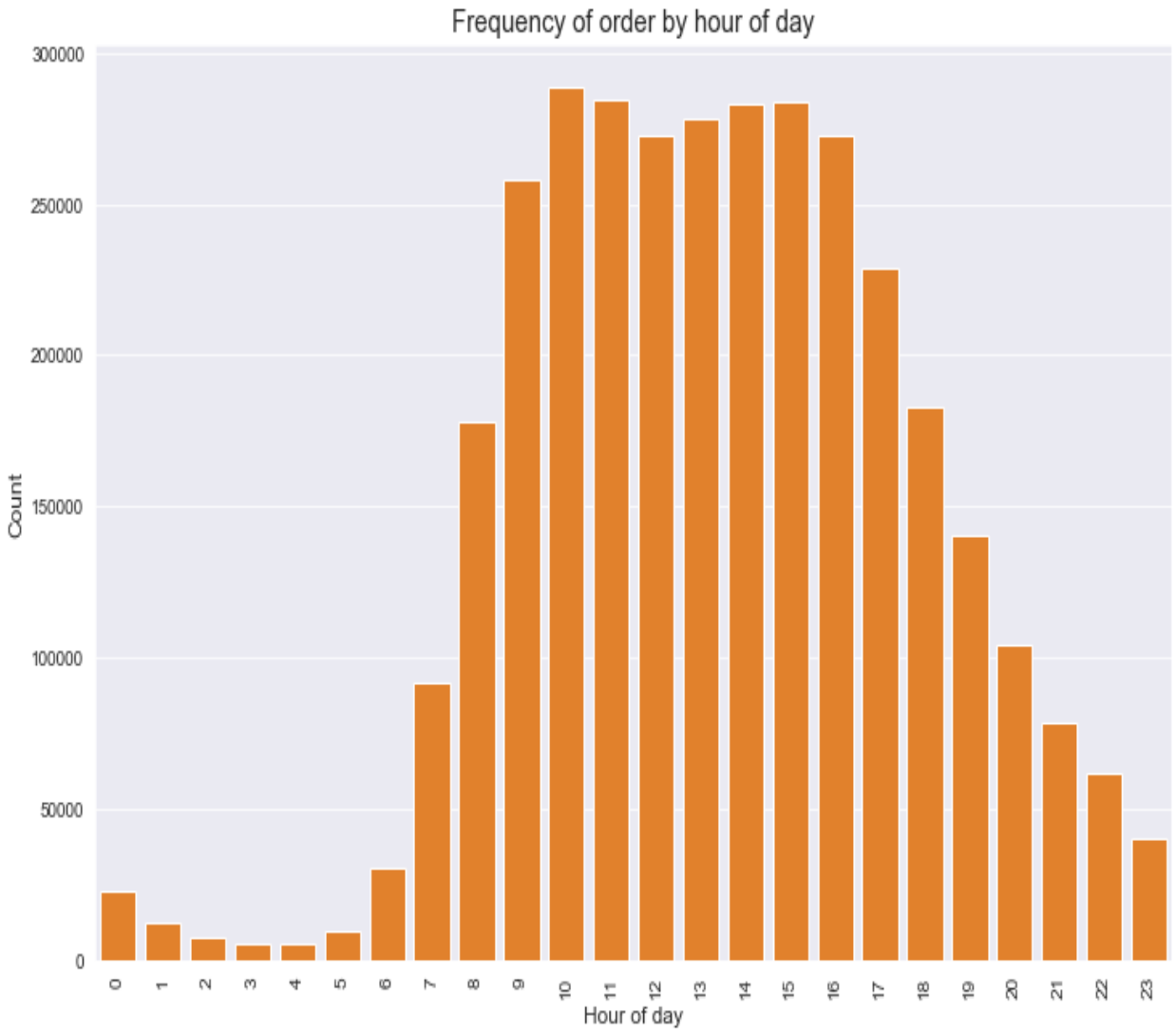
→ Orders Count Vs. Day of the Week

It's evident from the graph that the users mostly order on the 0th and 1st, which are Saturday and Sunday, respectively. The trend seems to decline post-weekend and then stays more or less the same during the weekdays.



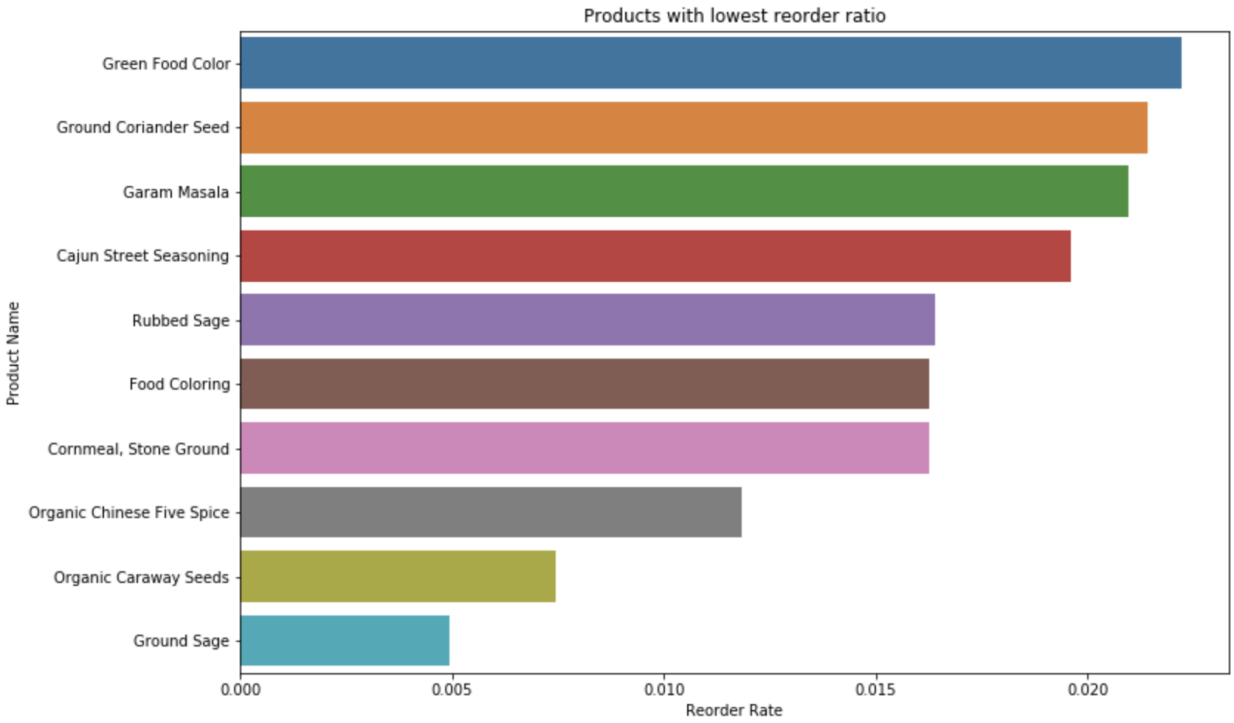
→ Orders Count Vs. Hour of the day.

On observing the graph, we can see that the ordering frequency is at the peak at 10 am and stays more or less similar up to 5 pm, after which we can see a gradual decline.

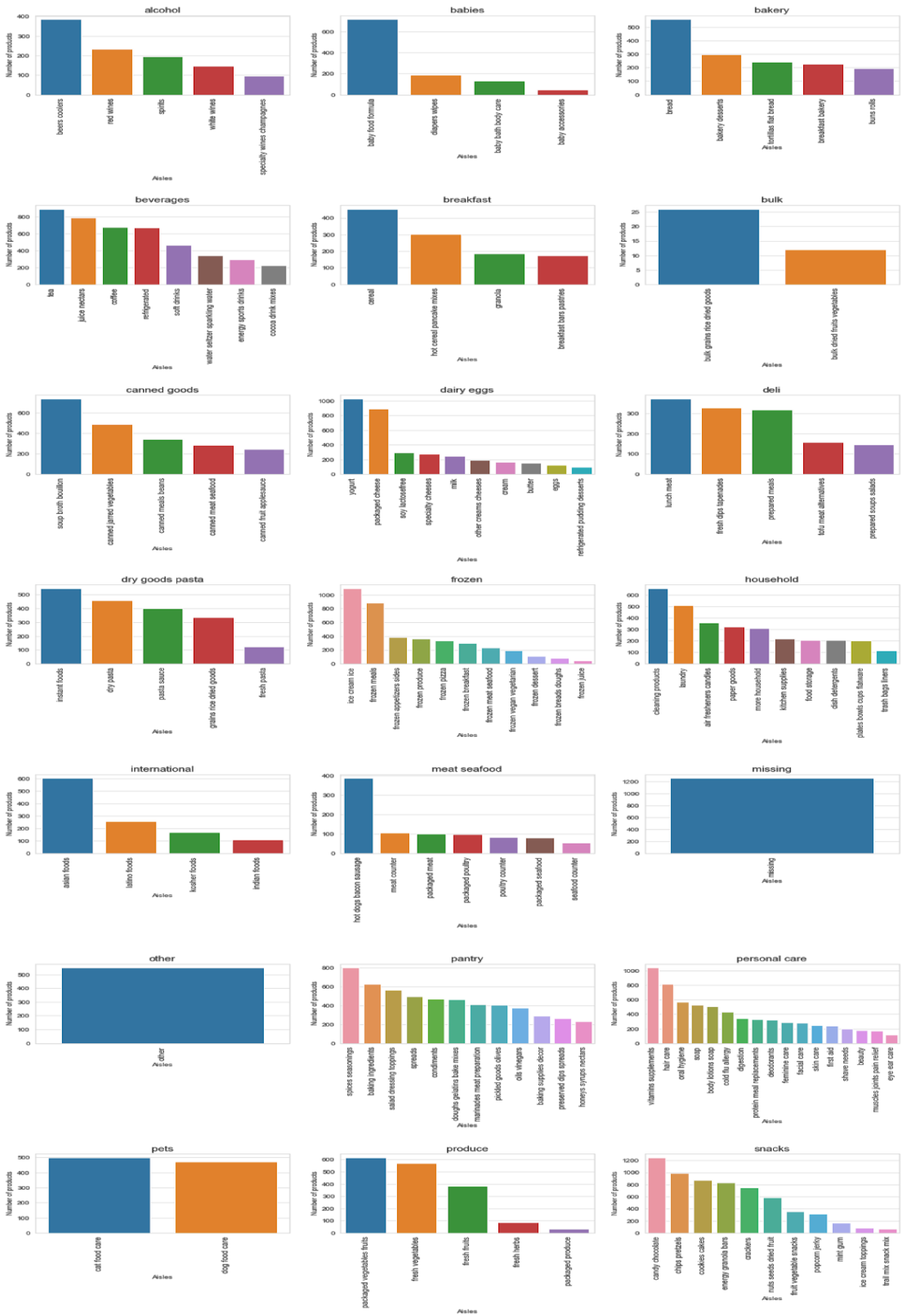


→ Least Reordered products vs Reorder probability.

From the graph, it's observed that the Green Food Color has the lowest reorder ratio amongst all, followed by the Ground Coriander seed.



→ Number of products based on department and aisle: The following plots divide the number of products among aisles and departments. The highest number of products in the personal care department is vitamin supplements, which also is one of the highest numbers of products based on aisles. Similarly, candy chocolates are the highest under the Snacks department.


alcohol
Number of products
beers coolers
red wines
spirits
white wines
specialty wines champagnes
Aisles
babies
Number of products
baby food formula
diapers wipes
baby bath body care
baby accessories
Aisles
bakery
Number of products
bread
bakery desserts
tortillas flat bread
breakfast bakery
buns rolls
Aisles
beverages
Number of products
tea
juice nectars
coffee
refrigerated
soft drinks
water seltzer sparkling water
energy sports drinks
cocoa drink mixes
Aisles
breakfast
Number of products
cereal
hot cereal pancake mixes
granola
breakfast bars pastries
Aisles
bulk
Number of products
bulk grains rice dried goods
bulk dried fruits vegetables
Aisles
canned goods
Number of products
soup broth bouillon
canned jarred vegetables
canned meals beans
canned meat seafood
canned fruit applesauce
Aisles
dairy eggs
Number of products
yogurt
packaged cheese
soy lactosefree
specialty cheeses
milk
other creams cheeses
cream
butter
eggs
refrigerated pudding desserts
Aisles
deli
Number of products
lunch meat
fresh dips tapenades
prepared meals
tofu meat alternatives
prepared soups salads
Aisles
dry goods pasta
Number of products
instant foods
dry pasta
pasta sauce
grains rice dried goods
fresh pasta
Aisles
frozen
Number of products
ice cream ice
frozen meals
frozen appetizers sides
frozen produce
frozen pizza
frozen breakfast
frozen meat seafood
frozen vegan vegetarian
frozen dessert
frozen breads doughs
frozen juice
Aisles
household
Number of products
cleaning products
laundry
air fresheners candles
paper goods
more household
kitchen supplies
food storage
dish detergents
plates bowls cups flatware
trash bags liners
Aisles
international
Number of products
asian foods
latino foods
kosher foods
indian foods
Aisles
meat seafood
Number of products
hot dogs bacon sausage
meat counter
packaged meat
packaged poultry
poultry counter
packaged seafood
seafood counter
Aisles
missing
Number of products
missing
Aisles
other
Number of products
other
Aisles
pantry
Number of products
spices seasonings
baking ingredients
salad dressing toppings
spreads
condiments
doughs gelatins bake mixes
marinades meat preparation
pickled goods olives
oils vinegars
baking supplies decor
preserved dips spreads
honeys syrups nectars
Aisles
personal care
Number of products
vitamins supplements
hair care
oral hygiene
soap
body lotions soap
cold flu allergy
digestion
protein meal replacements
deodorants
feminine care
facial care
skin care
first aid
shave needs
beauty
muscles joints pain relief
eye ear care
Aisles
pets
Number of products
cat food care
dog food care
Aisles
produce
Number of products
packaged vegetables fruits
fresh vegetables
fresh fruits
fresh herbs
packaged produce
Aisles
snacks
Number of products
candy chocolate
chips pretzels
cookies cakes
energy granola bars
crackers
nuts seeds dried fruit
fruit vegetable snacks
popcorn jerky
meat gum
ice cream toppings
trail mix snack mix
Aisles

6. Light Gradient Boosting Model

The basic principle behind the working of the boosting algorithm is to generate multiple weak learners and combine their predictions to form one strong rule. In gradient boosting base, learners are generated sequentially in such a way that the present base learner is more effective than the previous one. It optimized the loss function of the former learner.

It has three components:

- Loss Function that needs to be improved
- Weak learner for computing predictions and forming strong learners
- An additive model that will regularize the 'Loss Function.'

LightGBM uses a gradient-based one side sampling (GOSS), a technique to filter the data observations to find a split value while XGBoost uses a pre-sorted algorithm and Histogram based algorithm to determine the best split. Microsoft developed it in 2017. The way pre-sorting works is that for each node, boosting enumerates over all the features. After that, for each feature, the observations are sorted based on the feature value. Then the best split among all the features is decided.

Gradient-based one side sampling

In Adaboost, giving weights to the samples is a good way to increase accuracy. But, in GDBT, there are no weights given, so gradient-based sampling is done to improve efficiency. A gradient is the slope of the tangent of the loss function, so if the gradient is higher, that means the error is higher. GOSS keeps all the observations with a high gradient and performs random sampling on the observations with small gradients.

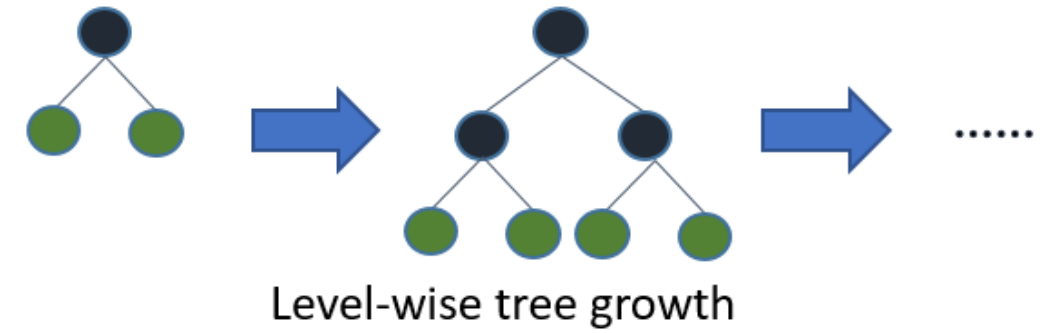
The main reason to use this model is its:

- Faster training speed and higher efficiency.
- Better Accuracy
- Supports parallel and GPU learning
- Capable of handling large dataset
- Lower memory usage

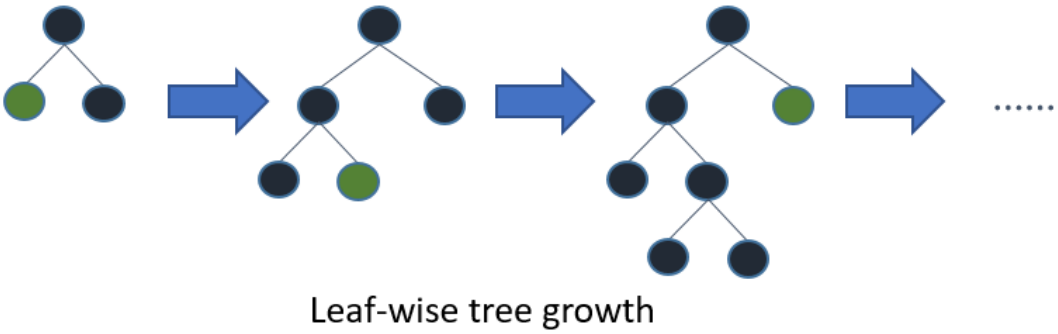
LGBM splits the tree leaf wise with the best fit, whereas other boosting algorithms split the tree vertically. It takes the leaf with the highest error and split it to reduce the loss. Leaf wise splits can increase complexity and overfitting the small datasets. It is recommended that LGBM used on dataset 10000+ rows.

The important parameter of LightGBM are:

- Task: If it is train or prediction
- Application: Binary classification
- Data: training data
- Num_iterations: number of boosting iterations. We took 100
- Num_leaves: Number of leaves in a tree: default is 31. We took 40
- Max_dept: Specify the max depth the tree will grow
- Categorical features: Specify the categorical features we want to use for training our model



Level-wise tree growth in XGBOOST



Leaf wise tree growth in Light GBM

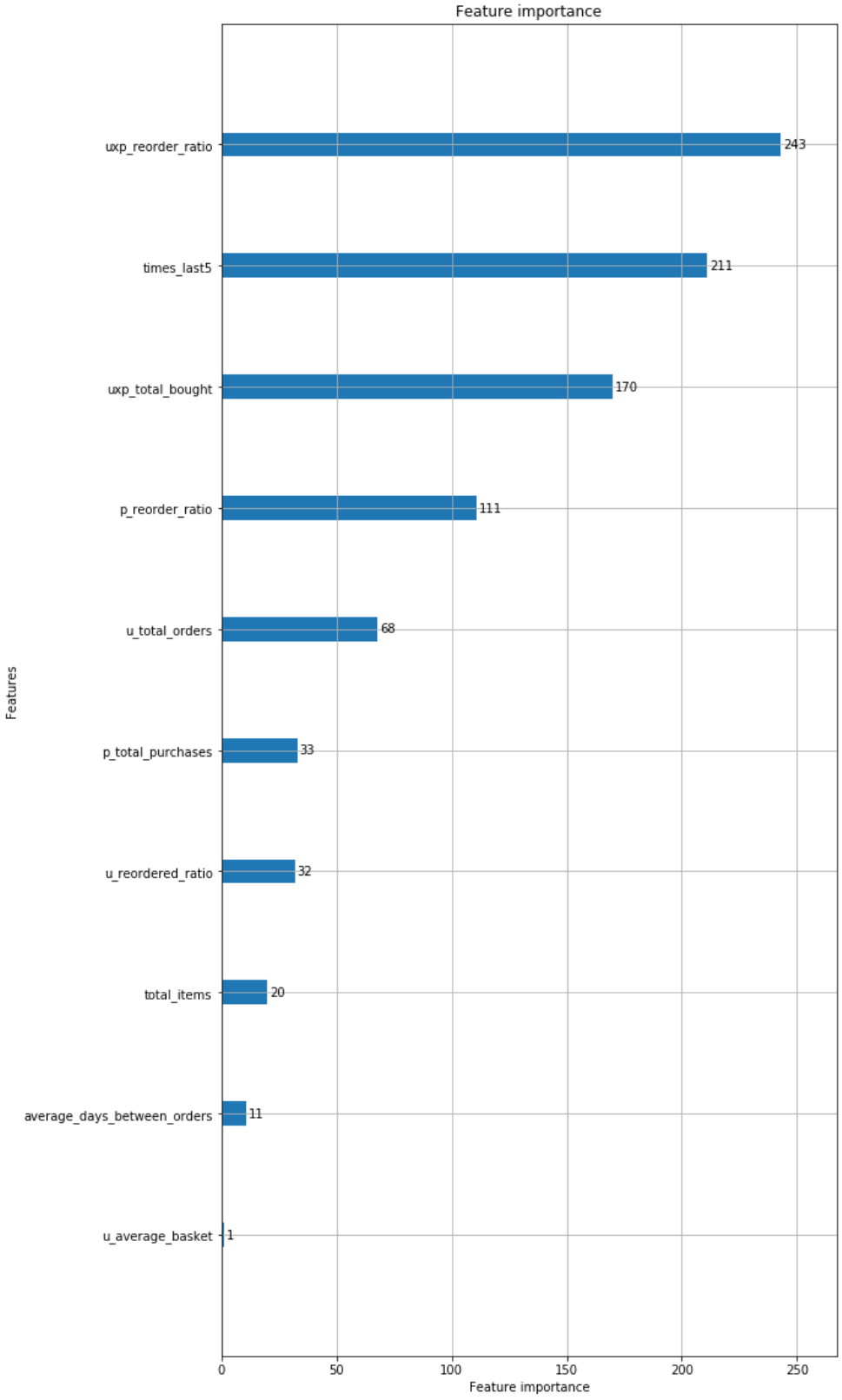
Binary Log Loss and F-1 Metric

- Accuracy: It is the number of predictions where predicted value is equal to the actual value. It is not always a good indicator because of its yes or no value.
- Log Loss: It takes the uncertainty of the prediction based on how much it varies from the actual label, and thereby giving a better look at the performance of our model. It is a probability between 0 and 1. Our goal is to minimize the value.

The binary log loss which we got was after cross validating the training data was 0.24849

F-1 metric was used in Kaggle to predict the accuracy of our predicted data, which came out to be as 0.3769. The F-1 score is the weighted average of precision and recall. It takes both false positives and false negatives, due to which the F1 score is more accurate than the accuracy. If you have uneven class distribution, this F-1 score is low, but it is hard to predict a shopper's behavior.

Below is the tree we got, and the feature importance graphs.



7. Market basket Analysis:

Market basket analysis is one of the primary techniques used by large retailers these days, which is used to uncover associations between products by looking for combinations of products that frequently co-occur in transactions. In other words, it allows supermarkets to identify relationships between products that people buy. For instance- customers that buy flour and sugar, are likely to buy eggs (because a high proportion of them are likely to bake a cake) Retailers can use these insights from Market Basket Analysis in multiple ways including-

1. Classifying products that co-exist in the design of the store's layout to increase the chance of cross-selling
2. Managing online recommendation engines
3. Focussing on the marketing campaigns by sending out promotional offers to customers for products related to the items they recently ordered.

In Market Basket Analysis, transactions are analyzed to identify the rules of the association. For instance – a rule could be {flour, sugar} → {eggs}, which means that if a customer has a transaction that contains flour and sugar, then they are most likely interested in buying eggs. In Market Basket Analysis, we need to measure the strength of the rule by calculating three commonly used metrics which are as follows:

1. Support: It is the percentage of transactions that contain all of the items in an itemset (e.g., flour, sugar, eggs). The bigger the support, the more frequently the itemset

occurs. Rules with great support are preferred since they are likely to apply to a large number of future transactions.

2. Confidence: The probability that a transaction contains the items on the left-hand side of the rule (in our example, flour, and sugar) also include the item on the right-hand side (eggs). The bigger the confidence, the higher the likelihood that the item on the right-hand side will be purchased or, in other words, the higher the return rate you can expect for a given rule.

3. Lift: It is the probability that all items occurring in a rule together, divided by the product of the chances of items on the left and right-hand side occurring as if there was no association between them. Overall, lift encapsulates the strength of the relationship between products on the left and right-hand side of the rule. The more significant the lift, the higher the link between the two products. To perform Market Basket Analysis and identify potential rules, we have used a data mining algorithm called the Apriori algorithm.

8. Apriori Algorithm:

Apriori Algorithm is an algorithm that finds its usage primarily in identifying frequent itemsets or item pairs. It does so using a "bottom-up" strategy, first defining objects that reach a minimum threshold of occurrence. It then expands the collection of items, adds one item at a time, and tests whether the resulting set of items still meets the defined threshold. When there are no more things to add, the algorithm stops. The name of the algorithm was coined Apriori because it uses prior knowledge of frequent itemset properties.

The rapid growth of e-commerce applications has led to vast amounts of data being accumulated in months instead of years. Data Mining, which is also known as the Knowledge Discovery in Databases (KDD), is designed to find any anomalies, correlations, patterns, and trends to predict results.
Apriori algorithm is a classic data mining algorithm. It is used for regular itemsets and association rules to be used for mining.

To avoid the kernel crash, rules were generated with the help of two item pairs. Since the Apriori algorithm is not a standard Python machine learning library, we also used Python Generators to make sure that the generators yield one value at a time rather than all values at once to avoid the kernel crashing. After running the algorithm, this is the set of rules which were generated.

	itemA	itemB	freqAB	supportAB	freqA	supportA	freqB	supportB	confidenceAtoB	confidenceBtoA	lift
0	Organic Raspberry Yogurt	Organic Wildberry Yogurt	40	0.010494	141	0.036992	142	0.037254	0.283688	0.281690	7.614904
1	Yerba Mate Sparkling Classic Gold	Cranberry Pomegranate Sparkling Yerba Mate	43	0.011281	168	0.044076	147	0.038566	0.255952	0.292517	6.636723
2	Organic Grapefruit Ginger Sparkling Yerba Mate	Cranberry Pomegranate Sparkling Yerba Mate	54	0.014167	213	0.055881	147	0.038566	0.253521	0.367347	6.573682
4	Baby Food Pouch - Roasted Carrot Spinach & Beans	Baby Food Pouch - Spinach Pumpkin & Chickpea	44	0.011544	236	0.061916	120	0.031483	0.186441	0.366667	5.922040
5	Baby Food Pouch - Roasted Carrot Spinach & Beans	Baby Food Pouch - Butternut Squash, Carrot & C...	60	0.015741	236	0.061916	180	0.047224	0.254237	0.333333	5.383672
...	...	...	...	...	...	...	...	...	...	...	...
11439	Organic Strawberries	Strawberries	71	0.018627	33331	8.744530	17838	4.679875	0.002130	0.003980	0.000455
5920	Organic Hass Avocado	Organic Avocado	59	0.015479	26772	7.023748	22187	5.820854	0.002204	0.002659	0.000379
3300	Organic Avocado	Organic Hass Avocado	49	0.012855	22187	5.820854	26772	7.023748	0.002209	0.001830	0.000314
1536	Banana	Bag of Organic Bananas	80	0.020988	59503	15.610866	47589	12.485177	0.001344	0.001681	0.000108
4278	Bag of Organic Bananas	Banana	62	0.016266	47589	12.485177	59503	15.610866	0.001303	0.001042	0.000083

48919 rows × 11 columns

9. Conclusion

Besides being a robust model, it helps us to pinpoint the future buying trends and product preferences for each user. Also, it shall help us to identify which previously bought product shall be ordered again. This model will give us better insights regarding which products need to be supplied. This model has overall helped us to leverage past purchase history in order to predict the subsequent behavior of the consumers. Also, as far as the store layout is concerned, the related products and aisles can be placed together to enhance the shopping experience of the consumers, along with boosting the profit margin for Instacart.

10. Future Scope

1. Adding geolocation in the data would help a great deal in predicting customer behavior and customer frequency. A mobile app is a very convenient way to track their location and send advertising texts or emails based on their location saying

"Hey! You bought eggs a week ago, maybe you need sunny side up eggs tomorrow morning". This helps in reaching out to better target customers.

2. Developing a better-mixed marketing attribution models. Better customer segments segregation provides us a means to advertise to the customer base through various communication means.
3. Furthermore, recent developments in digital marketing analysis have revealed the importance of social media. Learning customer characteristics using social media scrapping will help Instacart recommend better basket products.
4. Partnering with more local & remote grocery stores will help reach a more significant customer base.

11. References

1. "LightGM Documentation." LightGBM, https://lightgbm.readthedocs.io/en/latest/
2. "Features." LightGBM, https://lightgbm.readthedocs.io/en/latest/Features.html
3. "Python API." LightGBM, https://lightgbm.readthedocs.io/en/latest/Python-API.html
4. "Which algorithm takes the crown: Light GBM vs XGBOOST." Analytics Vidhya, https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/
5. "Microsoft LightGBM." GitHub, https://github.com/microsoft/LightGBM

6. “What is LightGBM, How to implement it? How to fine tune the parameters?” Medium, https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc
7. “LightGBM: A Highly Efficient Gradient Boosting Decision Tree” https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf