InstaCart Market Basket Analysis

Jyoti Yadav - jyoti.yadav@studio.unibo.it

January 8, 2022

1 Introduction

Instacart is an American company that provides grocery delivery and pick-up service. The Company operates in the U.S and Canada. Instacart offers its services via a website and mobile app. Unlike another E-commerce website providing products directly from Seller to Customer. Instacart allows users to buy products from participating vendors. And this shopping is done by a Personal Shopper.The company is expanding its platform to cover 90 millions US household in 2018. With millions of transactions in real time, Instacart's problem is a representative of a problem I would like to work from the eye of data scientist in order to predict customer behaviors with large amount of data. Let's deal this problem with the machine learning to enhance customer's shopping experience.

2 Outline

1. Problem Overview
2. Data Source
3. File Description
4. Exploratory data Analysis
5. Feature Engineering
6. Creating Training and Testing data
7. Modeling Building
8. Evaluation Metrics
9. Submission
10. Future Work
11. References

3 Problem Overview

The main objective of this competition was to predict which previously purchased products will be in the user's next order.The problem is a little different

from the general recommendation problem, in this case, we are not recommending new products to the user instead of that we are going to recommend the product which is previously bought by that user. It is an interesting problem because here we need to find the user's reordered behavior so we can personalize the user's cart.

4 Data Source

The dataset for this competition is a relational set of files describing customers' orders over time. The goal of the competition is to predict which products will be in a user's next order. The dataset is anonymized and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users. For each user, we provide between 4 and 100 of their orders, with the sequence of products purchased in each order. We also provide the week and hour of day the order was placed, and a relative measure of time between orders. For more information, see the blog post accompanying its public release.

5 File Description

File descriptions Each entity (customer, product, order, aisle, etc.) has an associated unique id. Most of the files and variable names should be self-explanatory. This table includes all aisles. It has a single primary key aisle_id.

aisles.csv

```
aisle_id,aisle
1,prepared soups salads
2,specialty cheeses
3,energy granola bars
...
```

departments.csv

This table includes all departments. It has a single primary key department_id

```
department_id,department
1,frozen
2,other
3,bakery
...
```

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. Note that some orders will have no reordered items. You may predict an explicit 'None' value for orders with no reordered items. See the evaluation page for full details.

```
order_id,product_id,add_to_cart_order,reordered
1,49302,1,1
2,11109,2,1
3,10246,3,0
...
```

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.

```
order_id,users_id,eval_set,order_number,order_dow,order_hour_of_day,
days_since_prior_order
2539329,1,prior,1,2,08,
2398795,1,prior,2,3,07,15.0
473747,1,prior,3,3,12,21.0
...
```

orders.csv

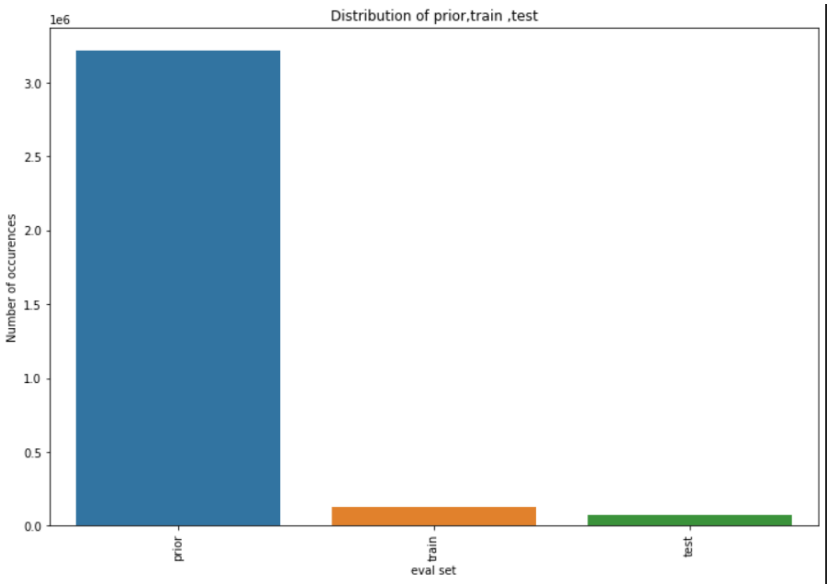
```
product_id,product_name,aisle_id,department_id
1,chocolate Sandwich cookies,61,19
2,All-Seasons,Salt,104,13
3,Robust Golden Unsweetened Oolong Tea,94,7
...
```

sample_submission.csv

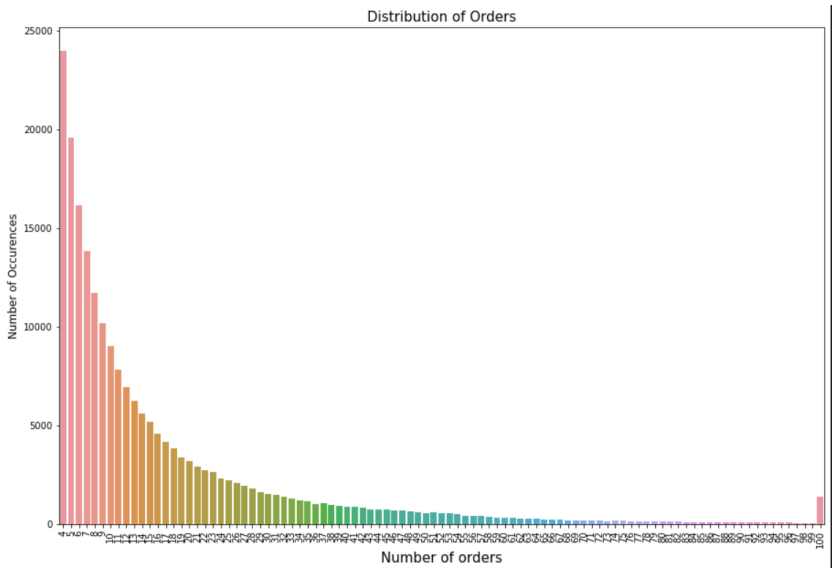
```
order_id,products
17,39276
34,39276
137,39726 ...
```

6 Exploratory Data Analysis

EDA is an important aspect of any machine learning or data science problem. It helps get insights in data through proper examination and is very important because it exposes trends, patterns, and relationships that are not readily apparent.Let's First look at the distribution of train ,test and prior.
Prior: 3214874 data points
Train: 131209 data points
Test: 75000 data points

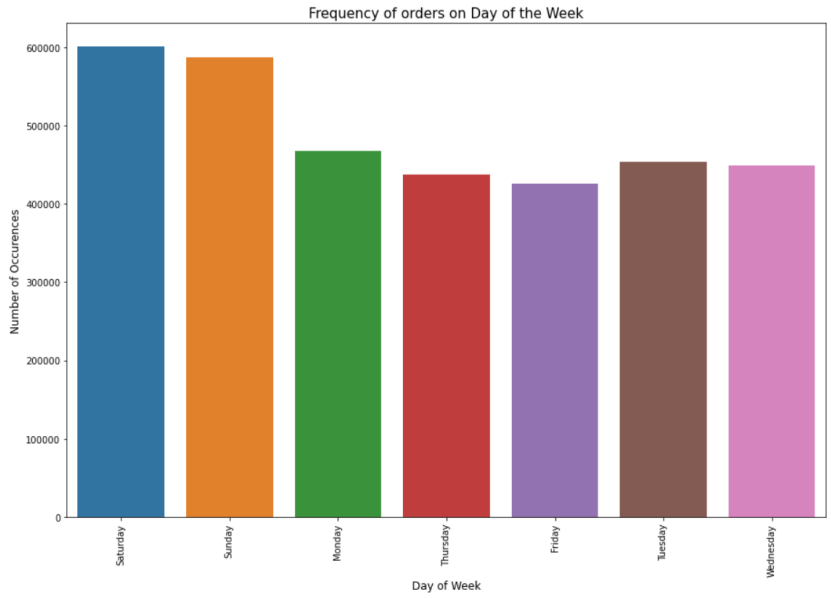


Let's check the Frequency Of Order Number Amongst The Users? Generally how many orders a user place?



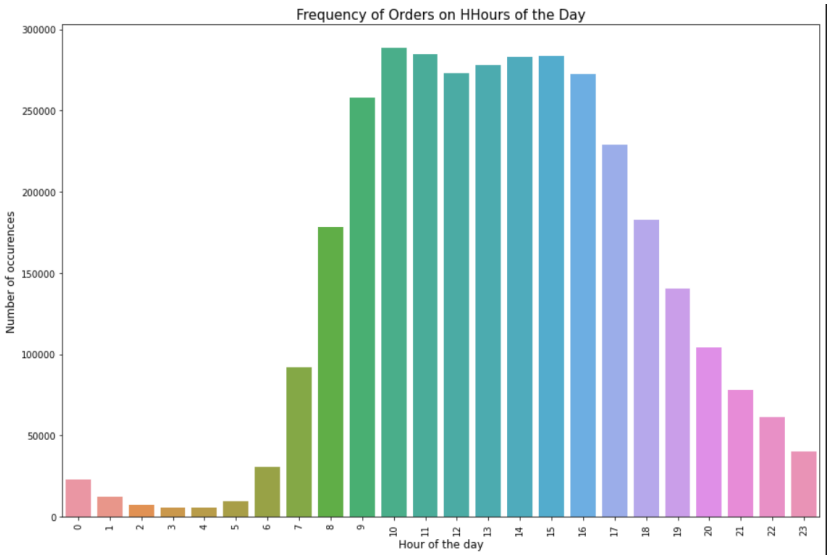
This plot explains, What's the order count for each user?. There is a high peak at 4,5,6,78 which tells us that users mostly orders 4 to 8 products in every order.

What Day Of The Week User Placed The Order?



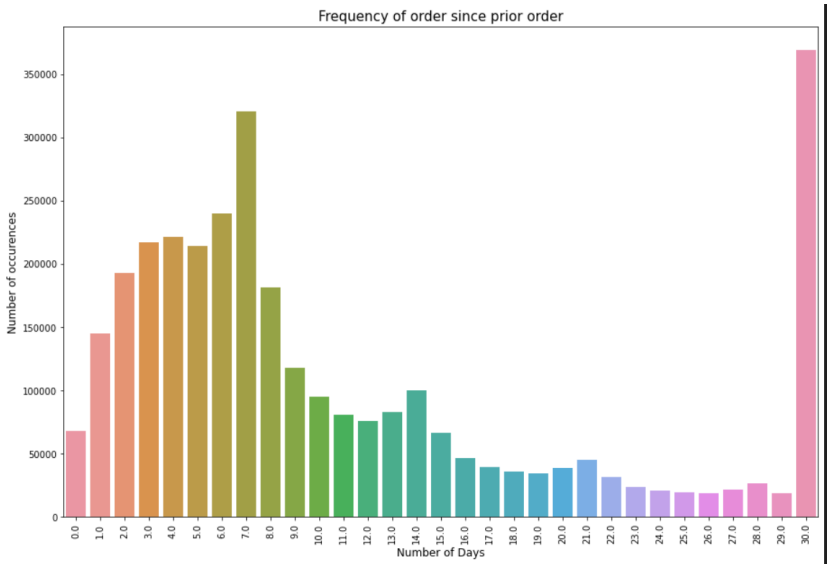
There are 7 days in a week most of the time user placed order on that starting days of the week i.e is on Saturday and Sunday where most of the users are free on that day and have time to place the orders Since after two days of the week frequency decreased a bit

In which hour user place the order



This plot tells us what's the user's preferred time for placing an order?. There are high peaks in the morning hours and it slowly gets decreases. Between 10 am to 15 pm there is a high order ratio amongst users.

How Many Days The User Takes To Place an Order?



After placing an order, how many days a user takes to order again?. Usually, there is a gap of 7 days after each order. If you see carefully there is a high peak

on the 7th day and after that on the 14th day and then at end of the month. The user takes a week to order once again.

Let's look for the uniqueness of the dataset

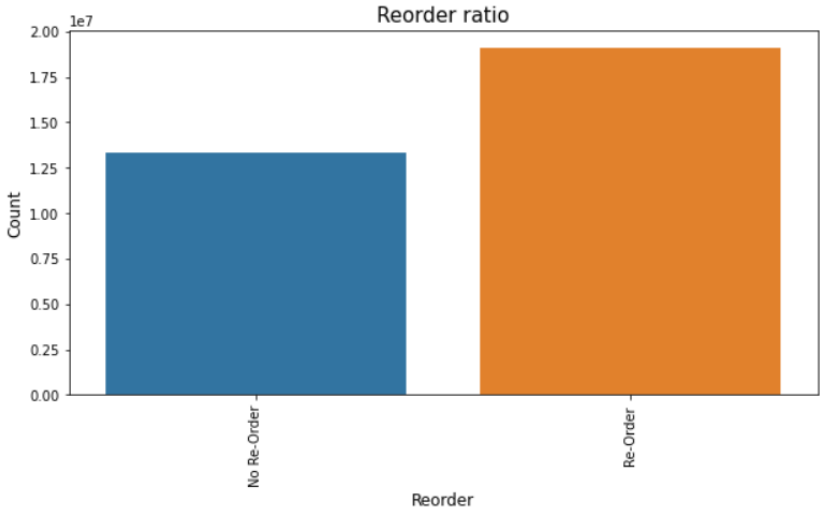
How many Unique Products in the Data?
Unique Number of products :49688

How many Unique departments in the data?
unique Department in the data :21

How many Unique aisles in the data?
unique Aisles in the data :134

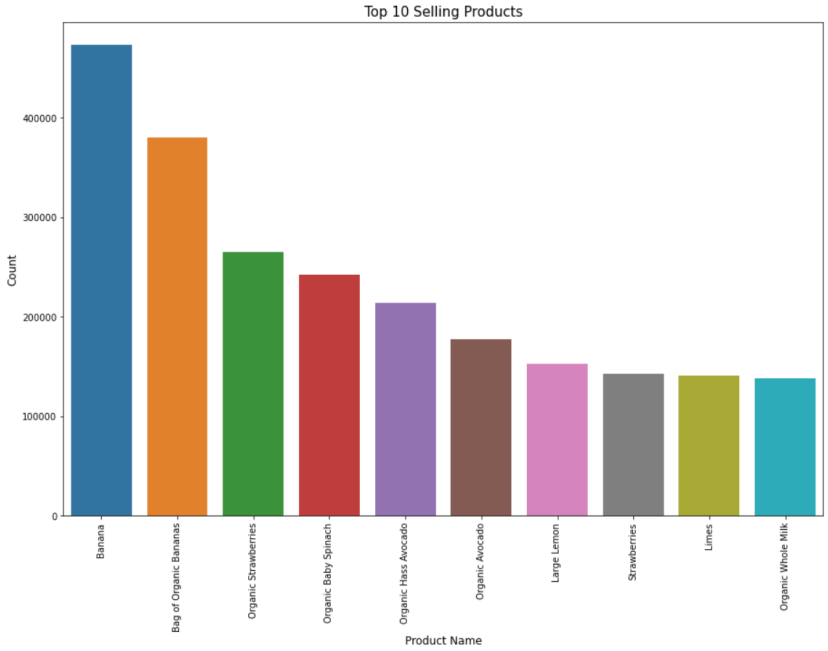
Let's merge the datasets by their primaryID's in order to find out the Products,Aisle and Department and explore the behaviour of customers towards their orders

After merging the three datasets we observed the how many times user have reorder the same item?



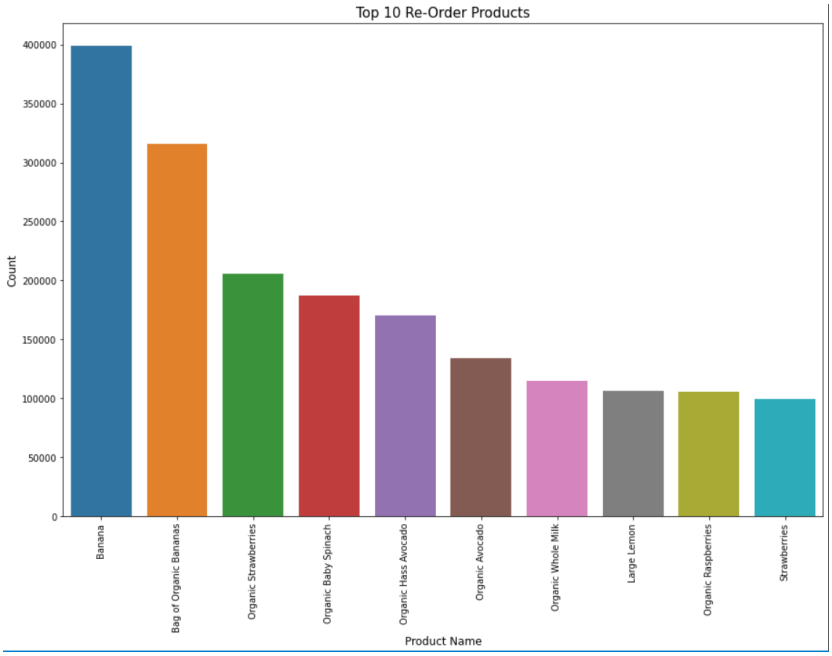
It was observed most of the time user reorder the same item

To further explore the dataset lets look at the top 10 selling and reordered items and also which item user added to the cart first



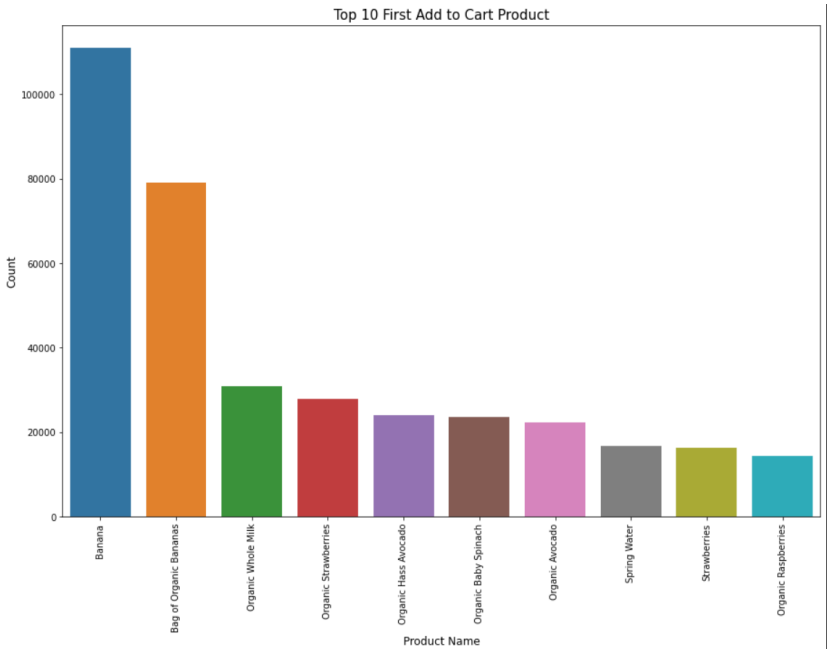
These are the top 10 selling products. We can clearly observe from the above graph that the top most selling product is Banana and Organic Banana Chips.

Top 10 Reordered products



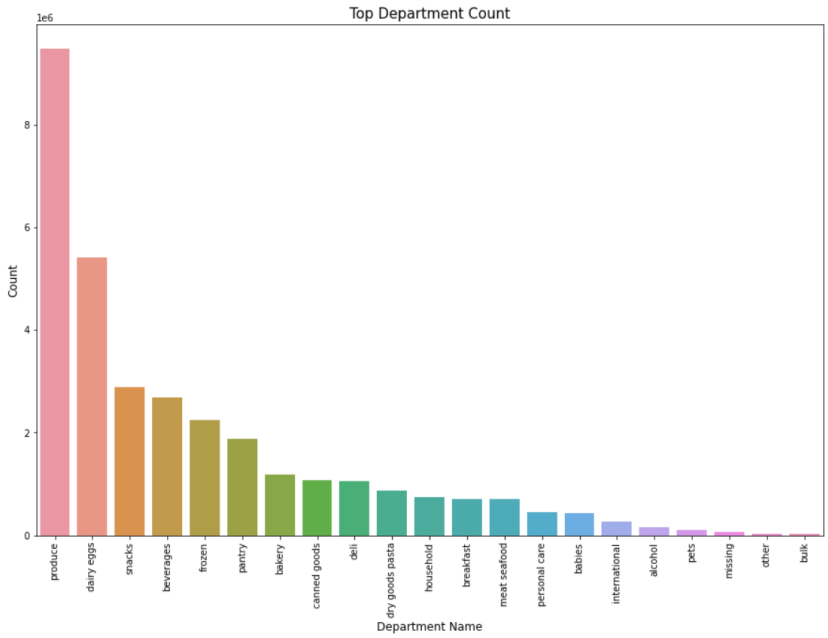
There is high peak on banana and oraganic banana chips looks like that that is most loving product and reorder products among users.

Top 10 First Add to Cart Product



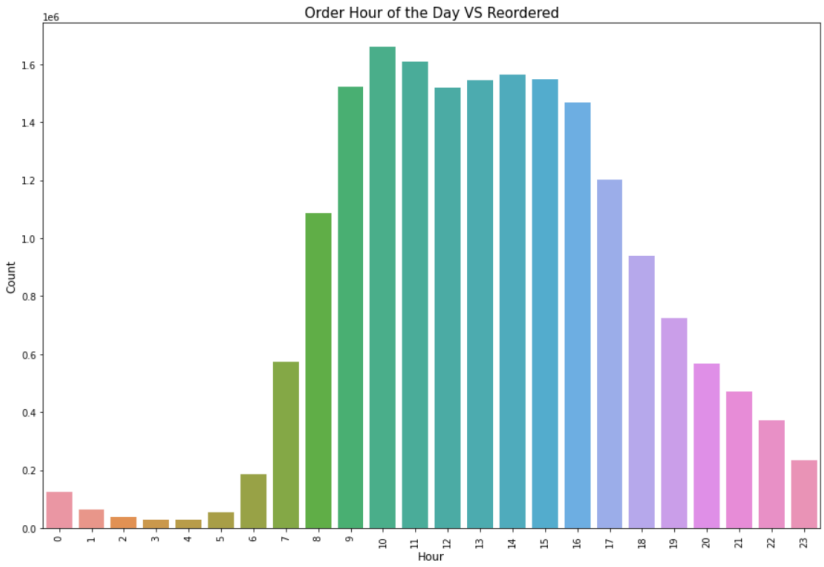
So it is clearly observed that top 10 selling products and 10 re-order product is almost same.If you see above 3 plots, all plots are almost same. Top selling,reordered and First product add to cart order product is Banana.From all this plot, the demand for produce product are high than any other Department.

Let's look at the distribution of Top departments



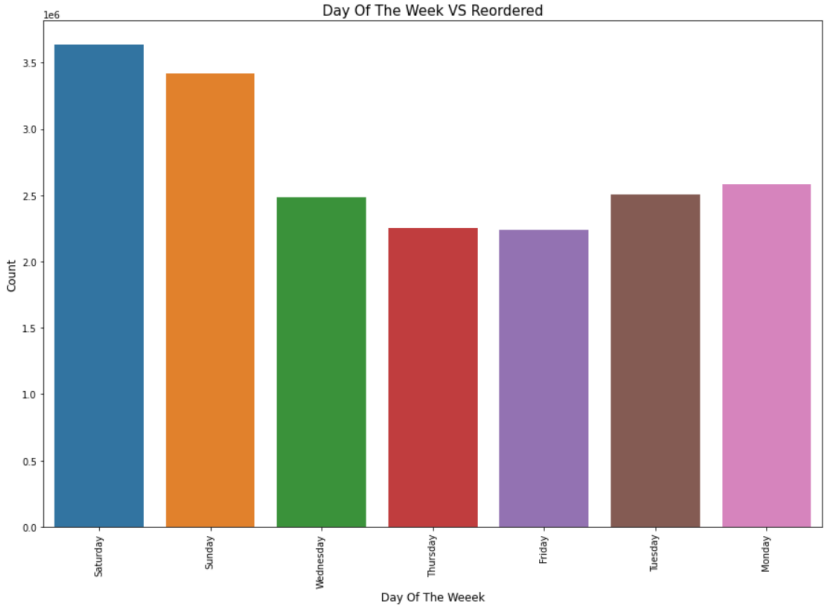
The First 2 Department have very high peak than any other department. As the First 2 Department are most important for any users because the Produce Department are about farm,crops and fruits and eggs are essential for the breakfast.

At what time user place the order again



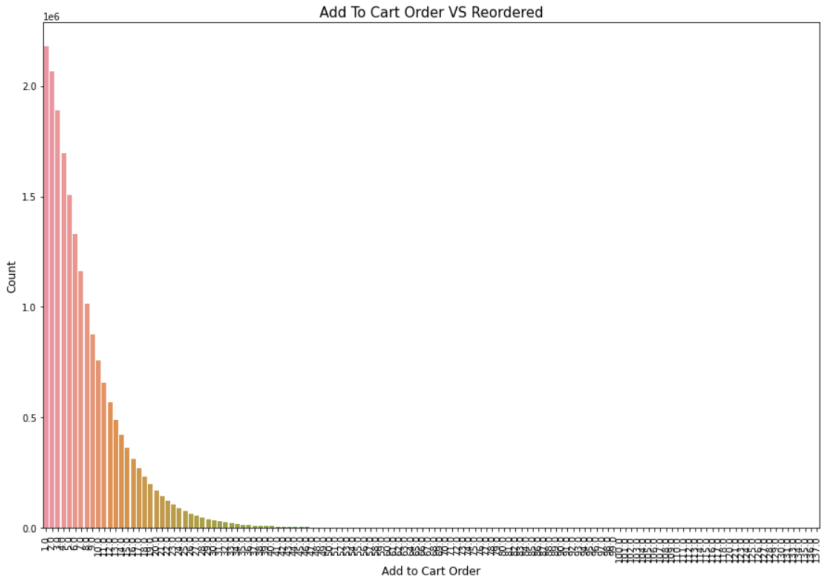
Here I plot at what time most reordered products are placed. Most of the orders are placed between 8'0 clock in the morning and 5'0 clock in the evening. This plot is almost same as Frequency of the order day. It shows user tends to order/reorder between this hours. This feature could help to predict reorder.

At what Day Most User place order again?



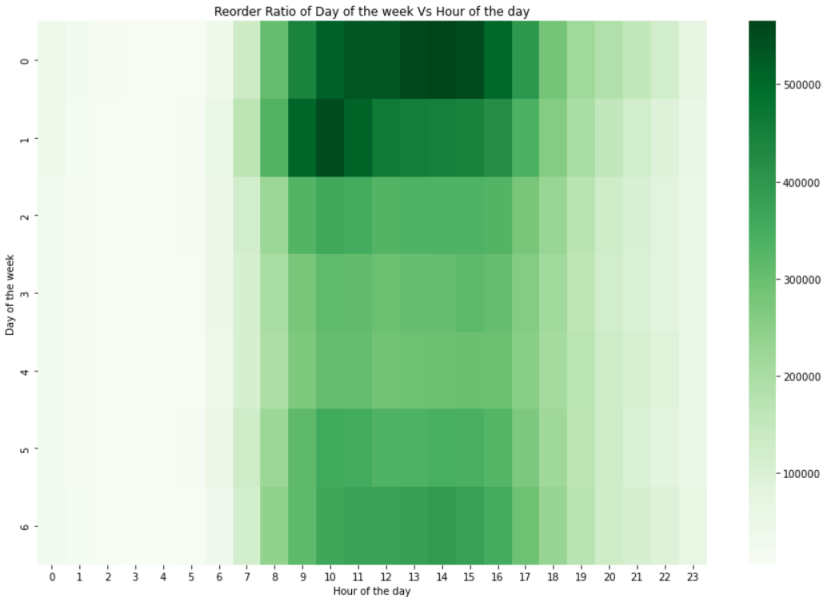
It was observed usually the people place the order again on weekends i.e on saturday and sunday. The Order and Re-Order ration are almost same. The user tends to place order again on Saturday and Sunday. Saturday and Sunday have the high ratio of Order/Re-Order getting placed.

Add To Cart Order VS Reordered



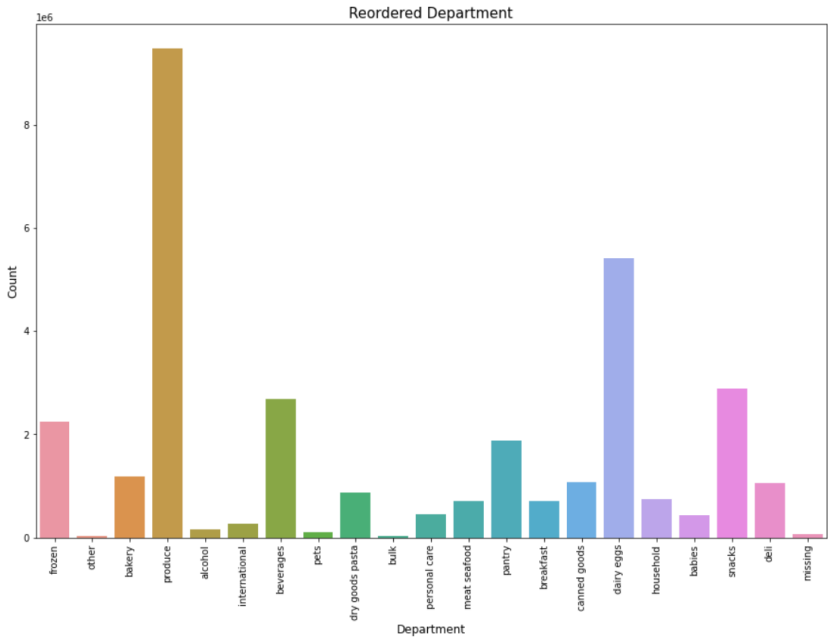
The order which are added at 1th and 4th position in the cart have high chance of Re-Order by users.

Reorder Ratio of Day of the Week Vs Hour of the Day



This Heatmap tells everything, Lot of users place order on Saturday and Sunday in between 10'clock to 15'clock.

Top Reordered Department



Most Users reorder from Produce and Dairy Eggs Department as they are essential for every day that's why they have high peak than any other department.

7 Feature Engineering

Let's start by merging the dataframe order and orders_products_prior.
We created three types of features.
1. user_id
2. Product_id
3. user_id and product_id

7.1 Creating Features using user_id

We have performed the following steps in order to create the user_id features

1. Let's count the Total number of orders placed by each users. Max of the order_number column.
2. Average number of products bought in each orders.
3. Day of the week the users orders the most.
4. Hour of the day the user has placed most of his/her orders.
5. Reordered ratio of each user.

6. Average days since prior order.
7. Total items bought by user.
8. Merging all the created features into the users dataset.

	user_id	u_num_of_orders	u_avg_prd	u_orders_the_most	hod_u_most_orders	u_reorder_ratio	average_days_between_orders	u_total_items_bought
0	7	17	11.0	0	10	0.818359	28.0	11
1	13	4	5.0	1	12	0.600098	9.0	5
2	23	2	9.0	0	10	0.000000	9.0	9
3	27	63	13.0	3	10	0.461426	1.0	13
4	36	10	3.0	1	18	0.333252	1.0	3

7.2 Creating Features using product_id

1. Number of times the product has been purchased by the users.
2. Reorder ratio of each products. Number of times the product was reordered number of times it was purchased.
3. Average add to cart order for each product.
4. Merging all the created features into the prd dataset.

7.3 Creating Features using user_id and product_id

1. How many times a User has bought a product.
2. How many times a user bought a product after its first purchase.
3. Finding when the user has bought a product for the first time.
4. Merging all the created features into the uxp dataset.
5. How many times a customer bought a product on its last 5 orders.
6. product bought by users in the last_five orders.
7. Ratio of the products bought in the last_five orders.

7.4 Merging the features users,prd and uxp dataframes

Merging all the three features into dataset called **data** in order to create the training and the testing datasets

8 Creating Training and Testing datasets

In order to create the train and test set from the orders. We keep only the train and test from the orders dataset. We merge the dataset created with the features and from the orders. Let's create the training data by using eval_set as 'Train'. Let's create the testing data by using eval_set as 'Test'. Merging product data into data_train and data_test.

One very common step in any feature engineering task is converting categorical features into numerical. This is called encoding and although there

are several encoding techniques like OneHotEncoding ,Label Encoding and Frequency Encoding But i decided to use the Mean Encoding.It tries to approach the problem in a more logical way.In a nutshell,it uses the target variable as the basis to generate the new encoded feature.

We create the X and y variables. X variables are created by dropping the reordered,uxp_ratio_last_five
y variable can be created by using only the reordered column
We split the data using the function train_test_split i.e 70% of Training data and 30% Testing data.

9 Model Building

We are trying to predict will the user reorder again?.This makes it a binary classification problem.There are various algorithms we can use for the binary classification task.I decided to check few models and compare their accuracy with the same threshold value. The most common algorithms that can be used for binary classification problems are :

1. Logistic Regression
Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability.We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the 'Sigmoid function' or also known as the 'logistic function' instead of a linear function.

```
log_reg = LogisticRegression(random_state=0,n_jobs=-1)
log_reg.fit(X_train,y_train)
#setting a threshold.
y_pred = (log_reg.predict_proba(X_test)[:, 1] >= 0.21).
astype('int')
```

2. Decision Tree
The decision tree Algorithm belongs to the family of supervised machine learning algorithms. It can be used for both a classification problem as well as for regression problem.The goal of this algorithm is to create a model that predicts the value of a target variable, for which the decision tree uses the tree representation to solve the problem in which the leaf node corresponds to a class label and attributes are represented on the internal node of the tree.

We optimize the hyperparameter using the RandomizeSearchCV.Random search is a technique where random combinations of hyperparameters are used to

find the best solution for the built model.It reduces the danger of overfitting, and is likely to provide more accurate long term results with the larger datasets—especially when there are a smaller number of significant hyperparameters.

```
param_grid = {}
param_grid['max_depth'] = [5,10,15,20]
param_grid['min_samples_split'] = [2,3,4,5]

dt_clf = DecisionTreeClassifier()
r_search = RandomizedSearchCV(dt_clf,
param_distributions=param_grid, cv=5, verbose=True,
n_jobs=-1)
r_search.fit(X_train, y_train)
#setting a threshold
y_pred3 =(r_search.predict_proba(X_test)[:,1]>=0.21)
.astype('int') .
```

3. Random Forest classifier

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

We decided to calibrate a classifier using the CalibratedClassifierCV class.It can predict the probabilities for each class label.This provides some flexibility both in the way predictions are interpreted and presented (choice of threshold and prediction uncertainty) and in the way the model is evaluated.We fit the model on a training dataset and calibrate it using the sigmoid.

```
clf=RandomForestClassifier(n_estimators=25,
random_state=42,n_jobs=-1)
clf.fit(X_train, y_train)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
sig_clf.fit(X_train, y_train)
#setting a threshold.
y_pred4 = (sig_clf.predict_proba(X_test)[:, 1] >= 0.21)
.astype('int')
```

4. XGBoost

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks.I have used XGBoost. It is a great algorithm as it provides a range of wide parameters that give fine-grained control over the model training procedure.The XGBoost returns the probabilistic values for a binary classification problem.

```
#setting boosters parameters
parameters = {'eavl_metric':'logloss','max_depth':5,
'colsample_bytree':0.4,'subsample':0.8}
```

```
xgb = xgb.XGBClassifier(objective='binary:logistic',
parameters=parameters, num_boost_round=10)
#fitting the model.
xgb.fit(X_train, y_train)
#setting a threshold.
y_pred2 = (xgb.predict_proba(X_test)[:, 1] >= 0.21)
.astype('int')
```

From the above models it was observed the XGBoost and Random Forest Classifier both of them performed well.But XGBoost had the F1-Score slighly higher than the Random Forest Hence we decided to take the predictions of the XGBoost.

```
F1 Score: 0.3904839721055732
```

10 Threshold

taking a threshold for classification.I have used Threshold greater than or equal to 0.21.These thresholds are selected based on:
Adhoc approach: I have applied the above three thresholds to the above models and check the F1-Score. It was observed using the the threshold value of 0.21 the F1-Score was higher.I took the reference to this
https://www.kaggle.com/c/instacart-market-basket-analysis/discussion/34977
to know about the threshold so it will maximize the score.

11 Evaluation Metric

The Kaggle evaluation metric is the Mean F1-Score and not the accuracy because accuracy is an inappropriate measure with unbalanced classes. Computing the F1 score, also known as balanced F-score or F-measure.The F1 score can be interpreted as a harmonic mean of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is:

F1 = 2 * (precision * recall) / (precision + recall).

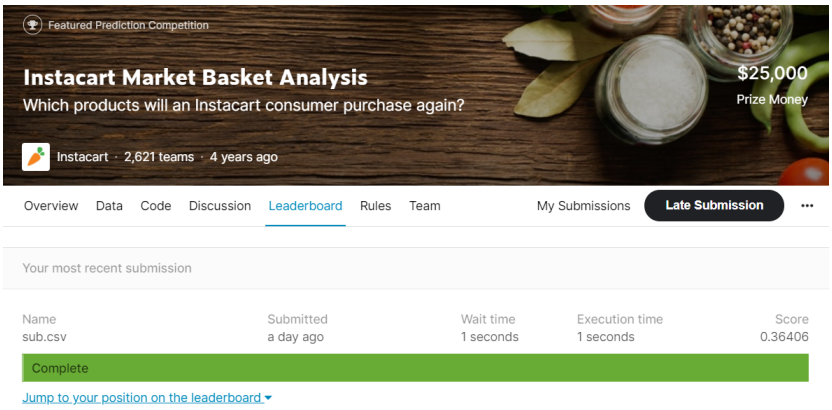
To calculate the Mean F1- Score we have used this (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html).

12 Submission

The final Submission file looks like this

	order_id	products
0	2774568	17668 18599 21903 39190 43961 47766
1	1528013	21903 38293
2	1376945	8309 14947 27959 28465 33572 34658 35948 44632
3	1356845	7076 10863 13176 14992
4	2161313	196 10441 11266 12427 14715 27839 37710

So, the Final Kaggle Score is 0.36406



13 Future Work

Here I used only a Machine Learning algorithm but in the future, I would like to try out some deep learning models and I think it will improve the score a lot.Also, I am looking to try the Apriori Algorithm to get some more features and also Some of the techniques are like F1 Optimization or Maximization as per discussed by the winner of the challenge here .

14 References

https://icml.cc/Conferences/2012/papers/175.pdf https://github.com/KazukiOnodera/Instacart https://www.kaggle.com/c/instacart-market-basket-analysis/discussion?sort=votes https://www.kdd.org/kdd2016/papers/files/adf0160-liuA.pdf https://www.kaggle.com/c/instacart-market-basket-analysis/ https://www.kaggle.com/mmueller/f1-score-expectation-maximization-in-o-n https://www.kaggle.com/c/instacart-market-basket-analysis/discussion/35468 https://www.kaggle.com/paulantoine/light-gbm-benchmark-0-3692 https://www.kaggle.com/c/instacart-market-basket-analysis/discussion/35048