Store Management for online shopping

**I Problem:**

Whether you're shopping from a well-planned grocery store or guided by whimsical grazing, our unique eating habits define who we are. When buying food, we often choose to buy back the goods they need. This is an accurate recommendation model that can help users quickly find their needs.

Recently, Instacart open sourced 3 million Instacart orders. We hope to analyze the transaction data from these orders to develop models and predict the products that users will buy again.

**II Goals:**

As the trendy of benefit maximization, store management should be optimized with machine learning. We choose history data information about Orders, Products, Hour of a Day, The Day of a Week and so forth as the dataset, to analyze and predict two things to help reducing the cost of managing a store.

Specifically, we set two goals to do this.

1. Predict the product that the user would buy again in the next purchase.
2. According to dataset we can predict in one week which day is the most busy and which day order number is the least. Accordingly, store supervisor would like to arrange schedule more reasonable according to this prediction

**III DataSet:**

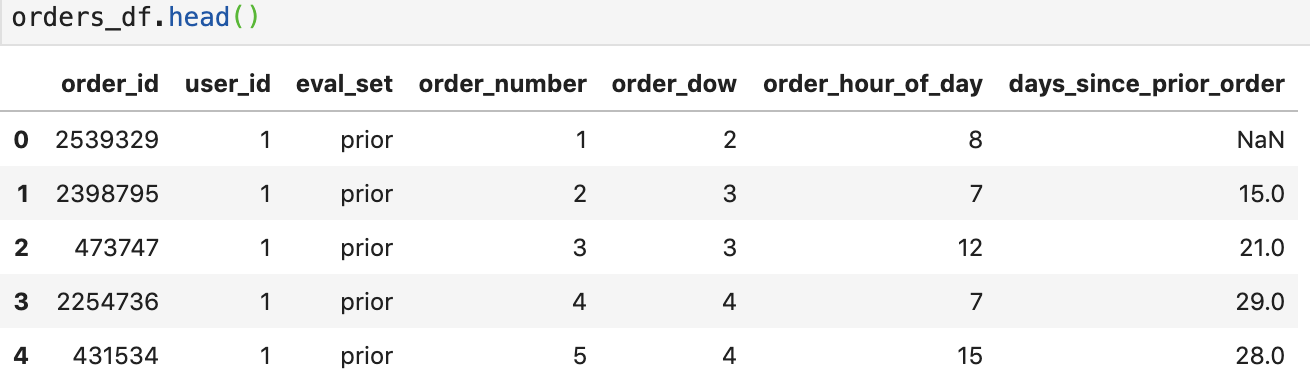
The dataset is a relational set of files describing customers' orders over time. The goal of the competition is to predict which products will be in a user's next order. The dataset is anonymized and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users. For each user, 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.

The data are given as CSV format, and the data amount is up to 200M.



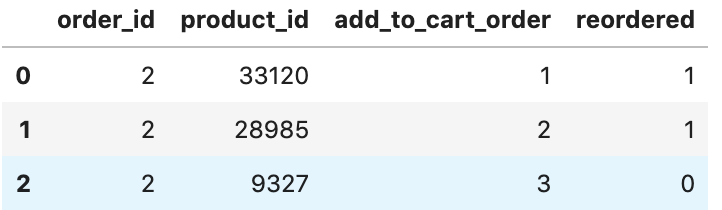
**orders.csv**

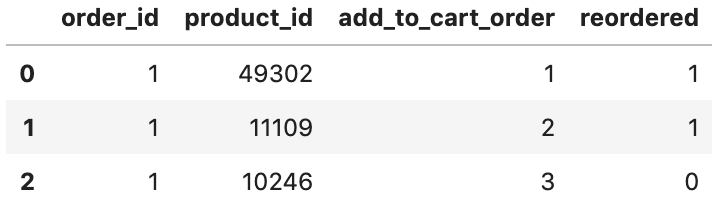
This file tells to which set (prior, train, test) an order belongs. You are predicting reordered items only for the test set orders. 'order\_dow' is the day of week.



**order\_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.





**IV. Implementation**

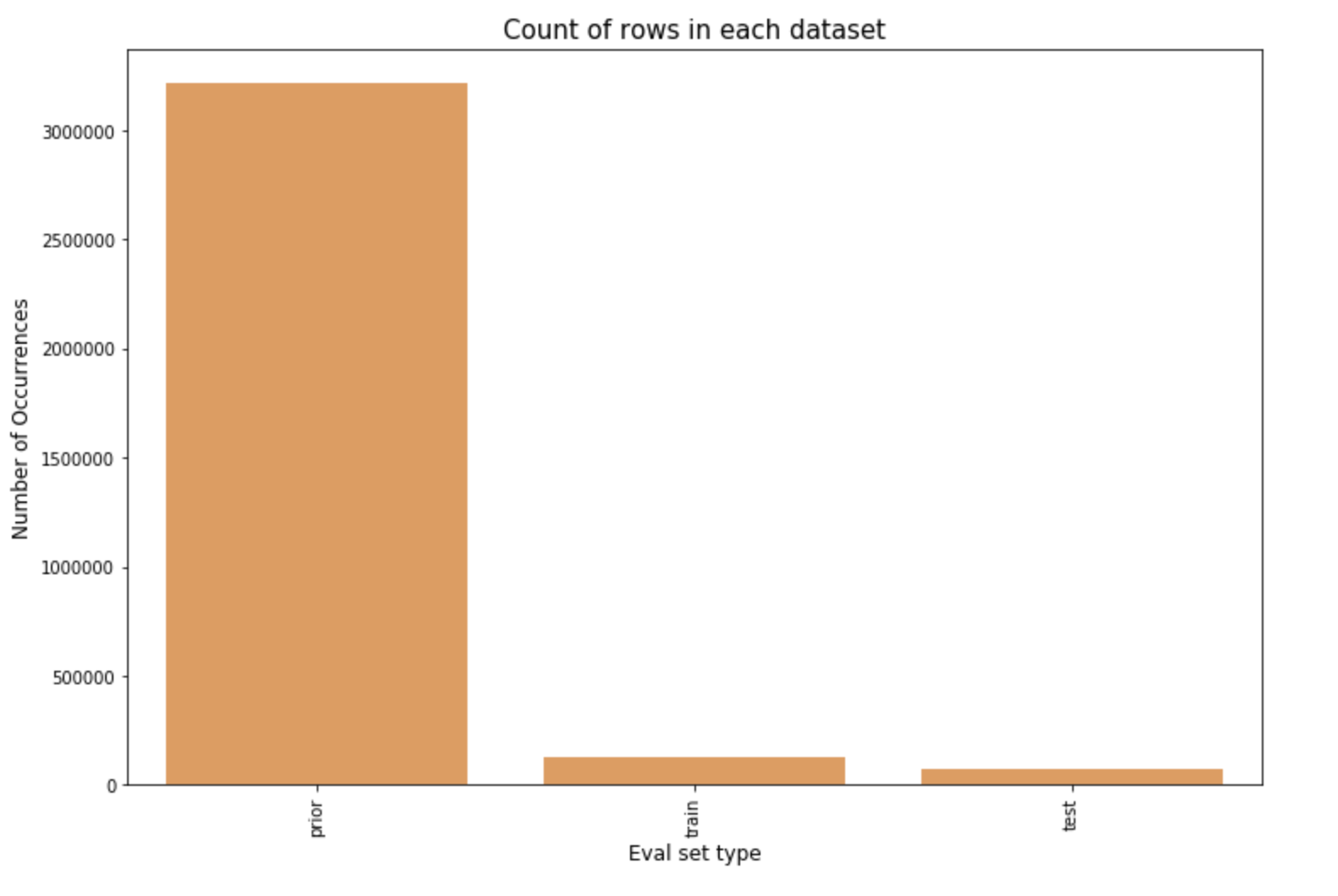
As mentioned above, we have used multiple data sets to advance and test our projects. Not only do we need to train our machine learning models through data sets, but we also need to verify our accuracy and models through data validation. Next, I will introduce the process of verifying the model accuracy using LightGBM and Xgboost algorithm implementation process.

**4.1 Data Processing**

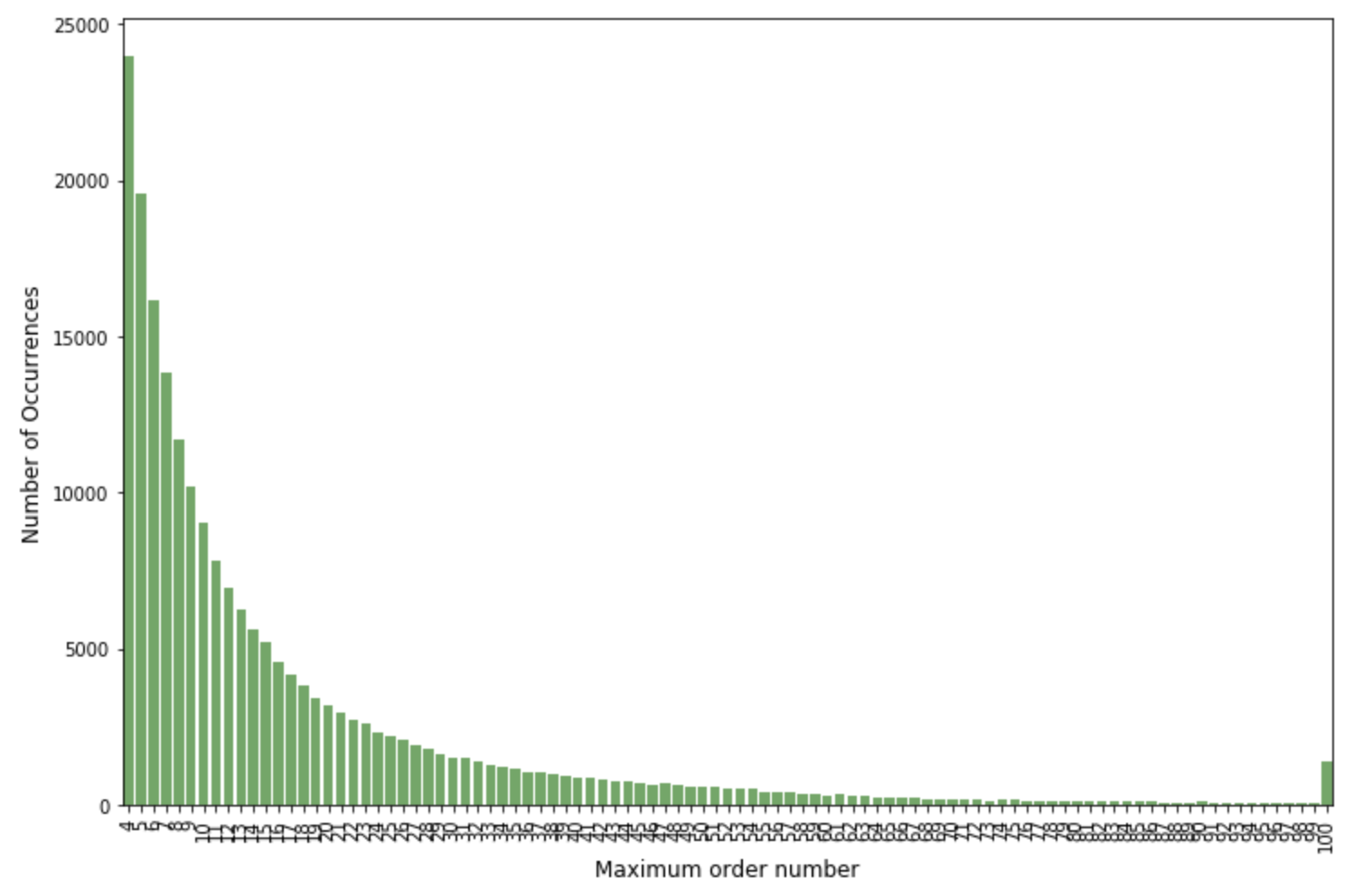
As mentioned earlier, in this dataset, 4 to 100 orders of a customer are given (we will look at this later) and we need to predict the products that will be re-ordered. So, the last order of the user has been taken out and divided into train and test sets. All the prior order information of the customer is present in order\_products\_prior file. We can also note that there is a column in orders.csv file called eval\_set which tells us as to which of the three datasets (prior, train or test) the given row goes to.

Order\_products\*csv file has more detailed information about the products that been bought in the given order along with the re-ordered status.

1. Let us first get the count of rows in each of the three sets.

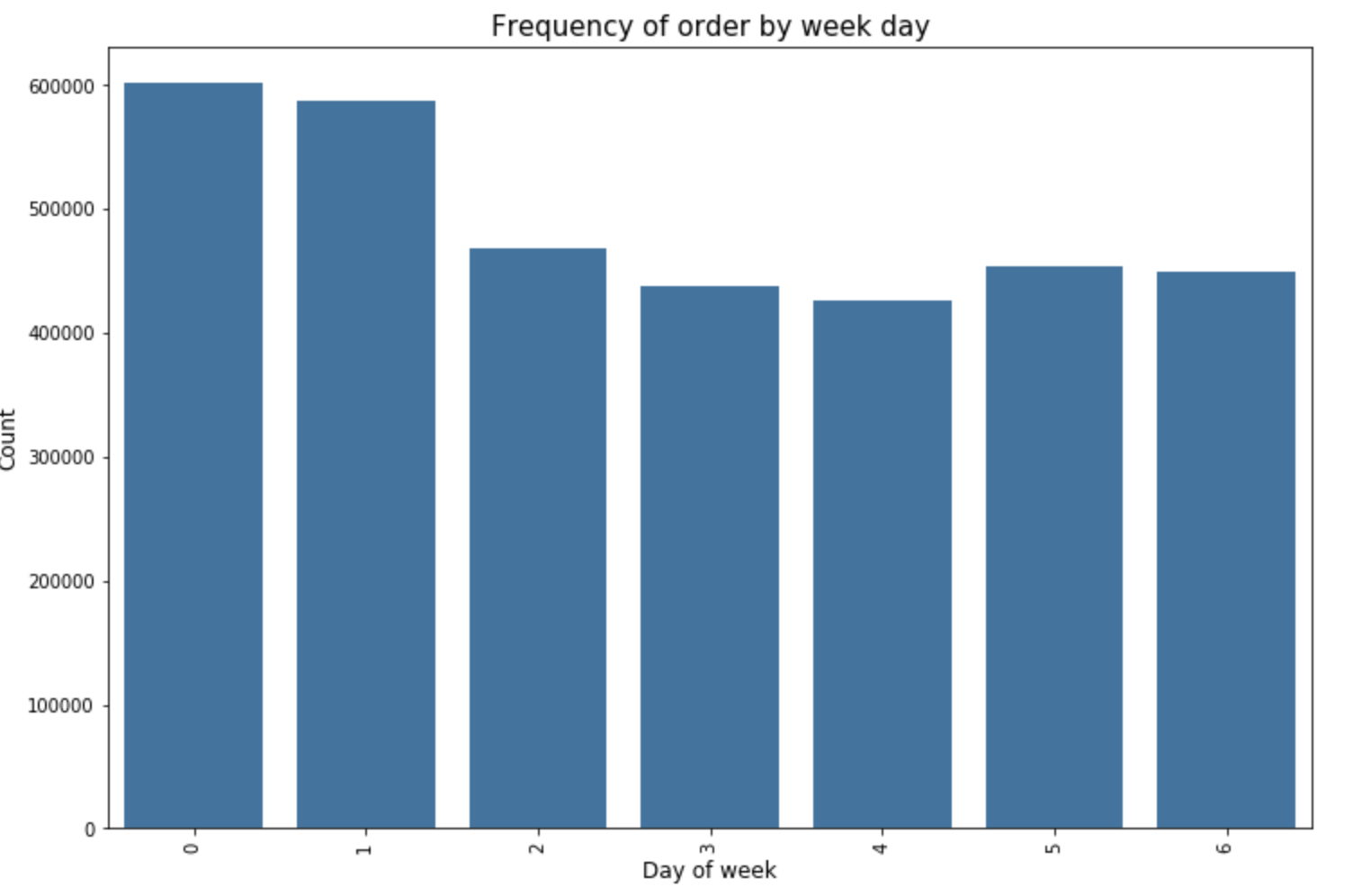


1. Now let us validate the claim that 4 to 100 orders of a customer are given.



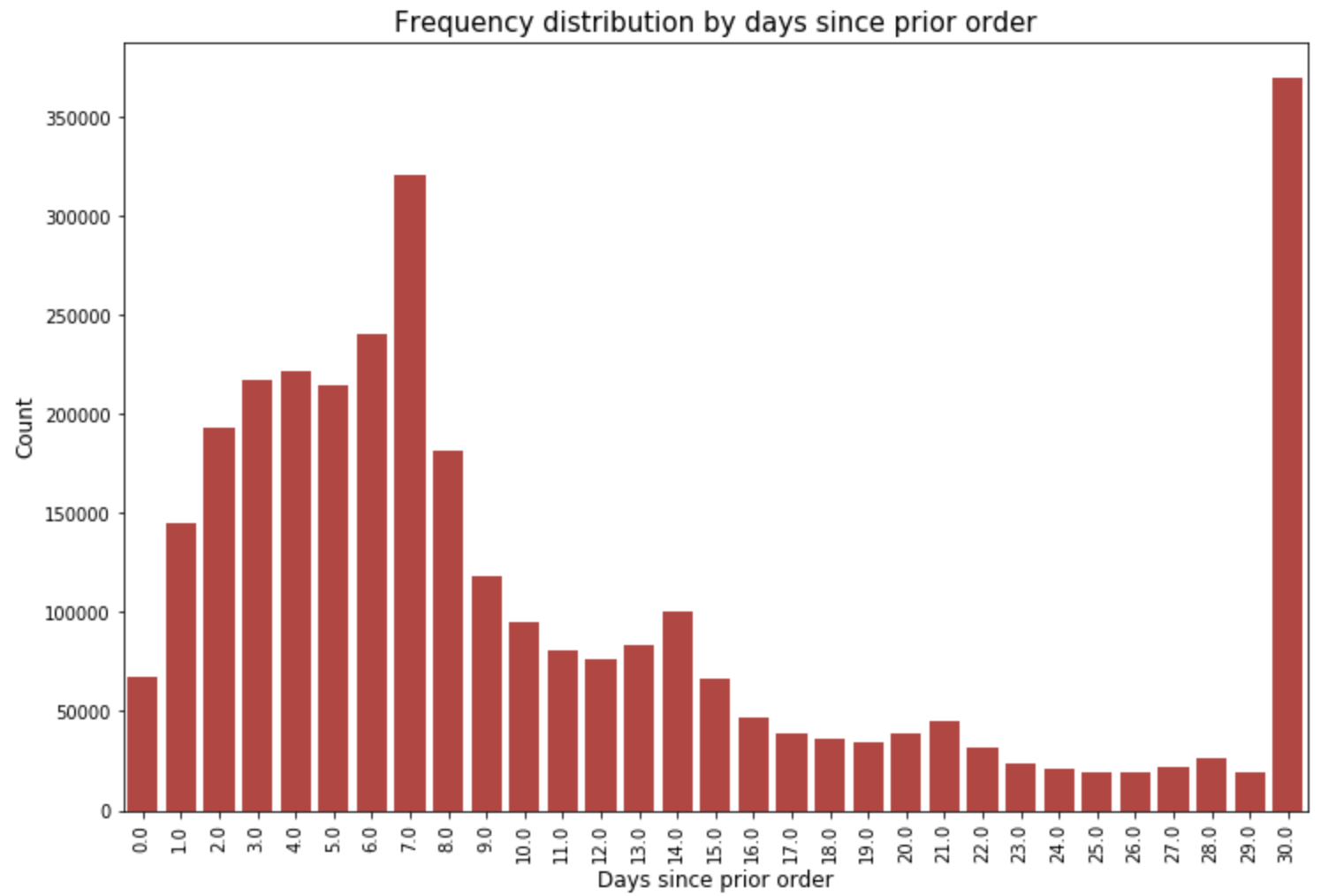
So, there are no orders less than 4 and is max capped at 100 as given in the data page.

1. Now let us see how the ordering habit changes with day of week.



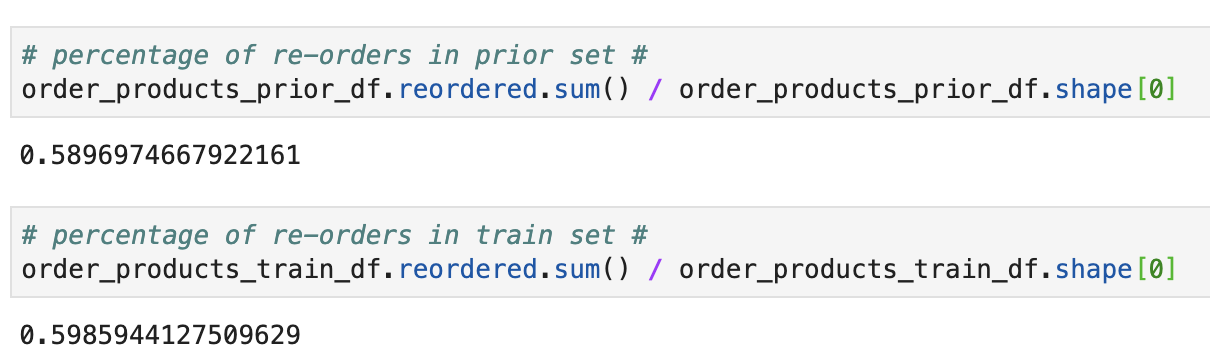
Seems like 0 and 1 is Saturday and Sunday when the orders are high and low during Wednesday.

1. Now let us check the time interval between the orders.



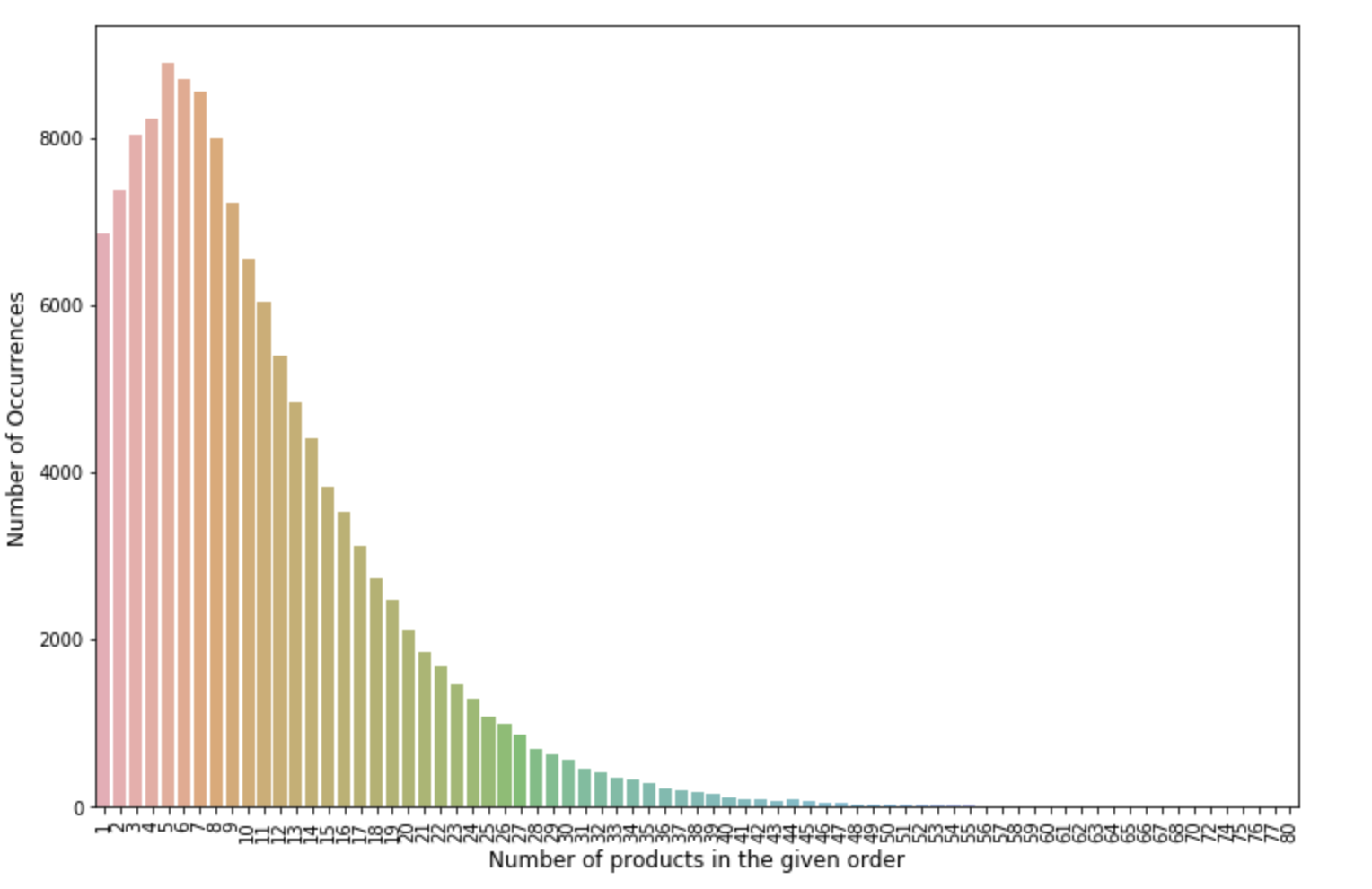
Looks like customers order once in every week (check the peak at 7 days) or once in a month (peak at 30 days). We could also see smaller peaks at 14, 21 and 28 days (weekly intervals).

1. Since our objective is to figure out the re-orders, let us check out the re-order percentage in prior set and train set.



On an average, about 59% of the products in an order are re-ordered products.

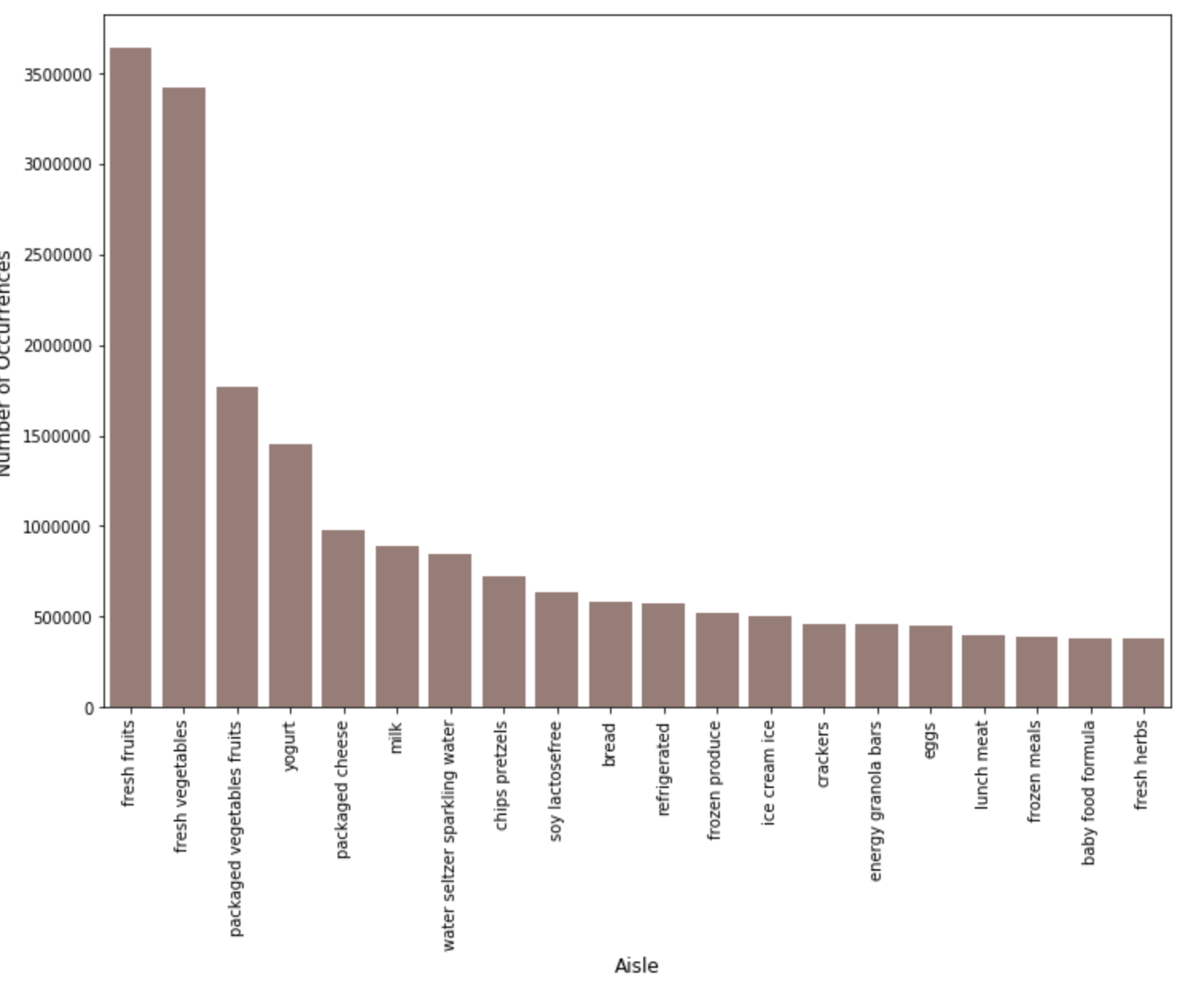
1. Now let us see the number of products bought in each order.



A right tailed distribution with the maximum value at 5.

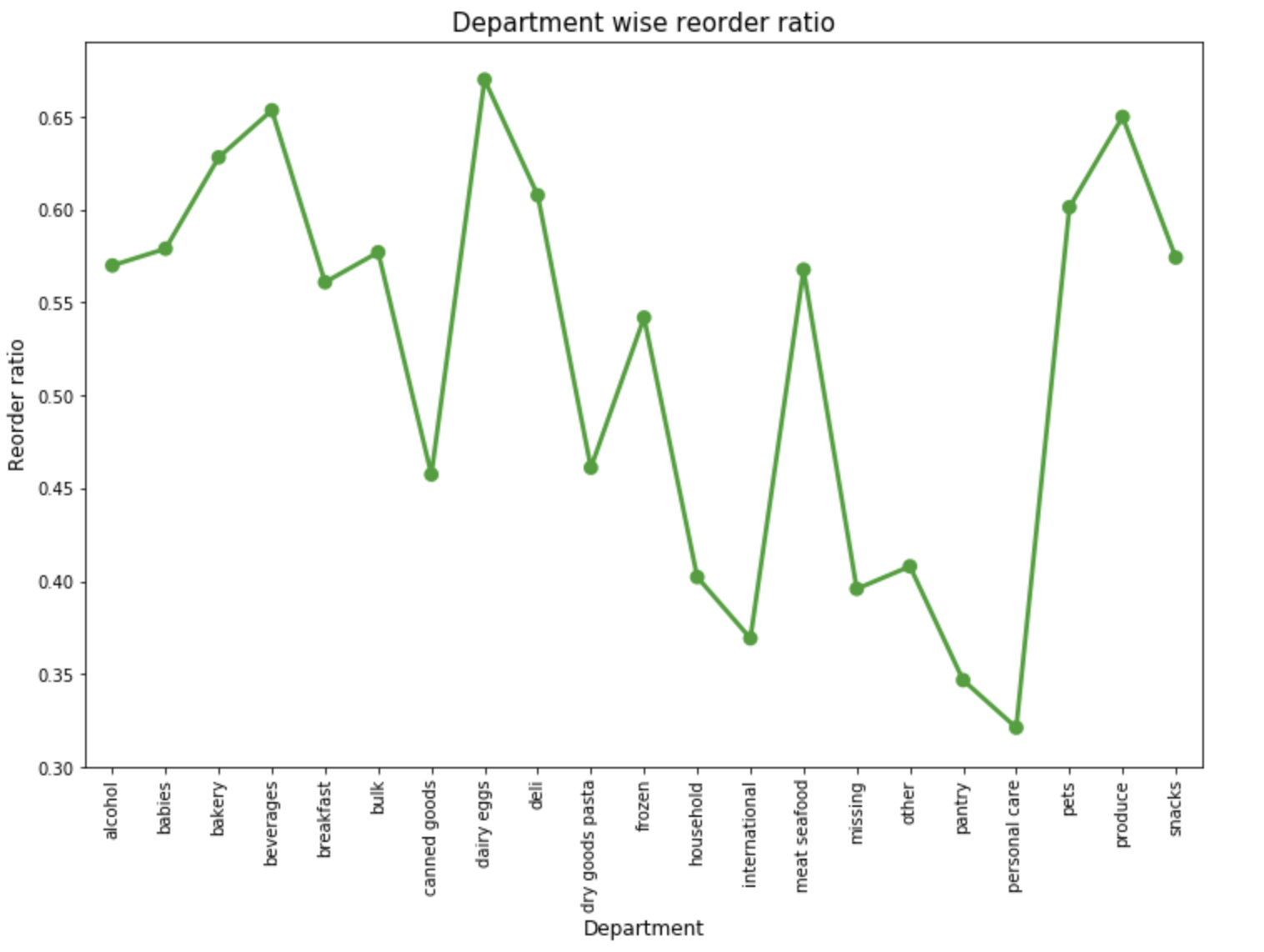
1. Now let us merge these product details with the order\_prior details and look at the important aisles.





The top two aisles are fresh fruits and fresh vegetables.

1. Now let us check the reordered percentage of each department.



Personal care has lowest reorder ratio and dairy eggs have highest reorder ratio.

**4.2 Implement of Classifications**

1. **LightGBM**
2. **XGboost**

XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data.

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

XGBoost stands for extreme Gradient Boosting.

**Model Features**

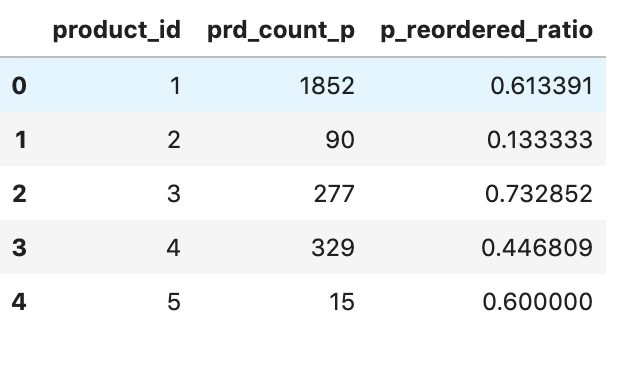
The implementation of the model supports the features of the scikit-learn and R implementations, with new additions like regularization. Three main forms of gradient boosting are supported:

Gradient Boosting algorithm also called gradient boosting machine including the learning rate.

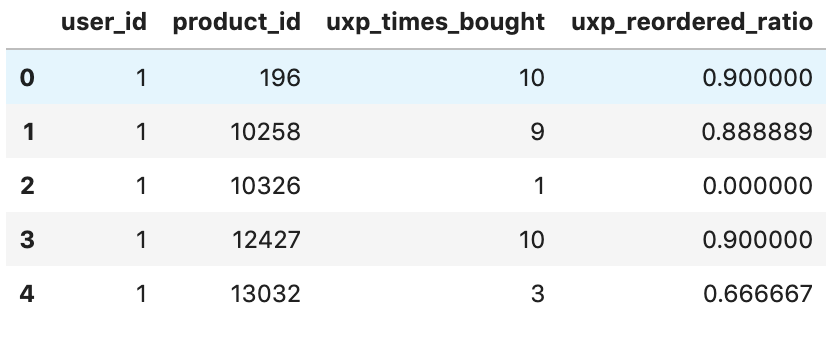
Stochastic Gradient Boosting with sub-sampling at the row, column and column per split levels.

Regularized Gradient Boosting with both L1 and L2 regularization

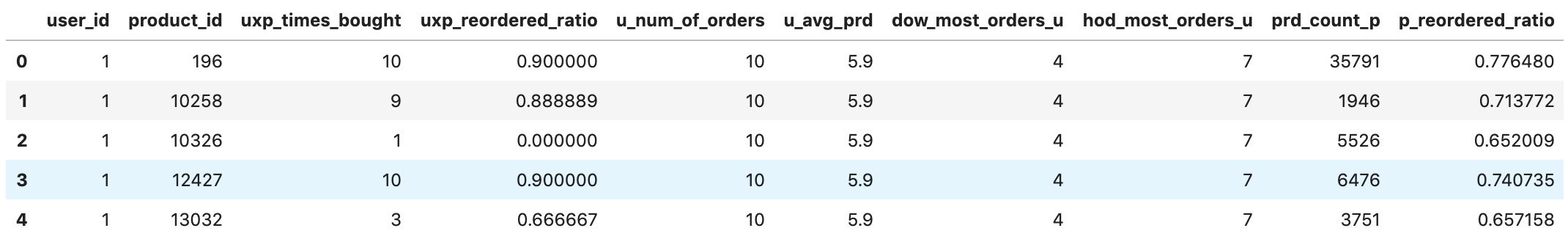
**Creating features related to the products using product\_id**

****

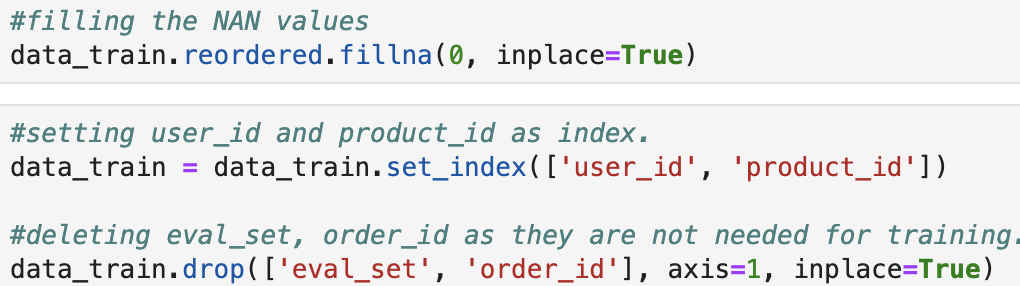
**Creating user-product features**

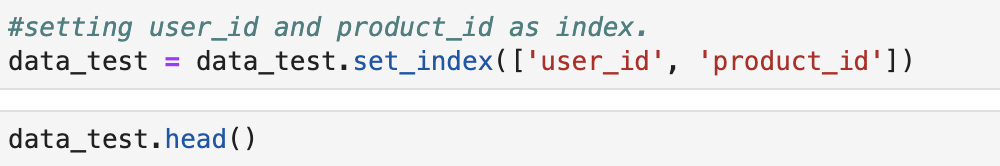
****

**Merging all the features into data DF**

****

**Creating train and test dataset**

****

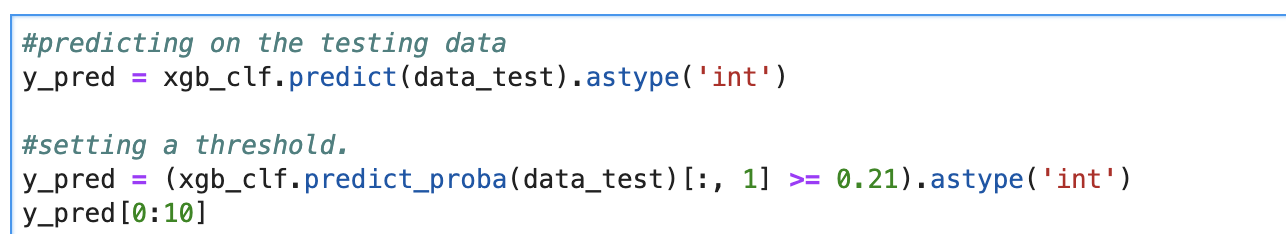
****

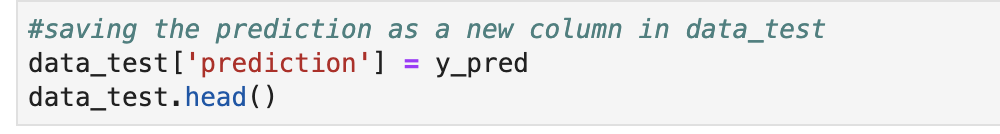
****

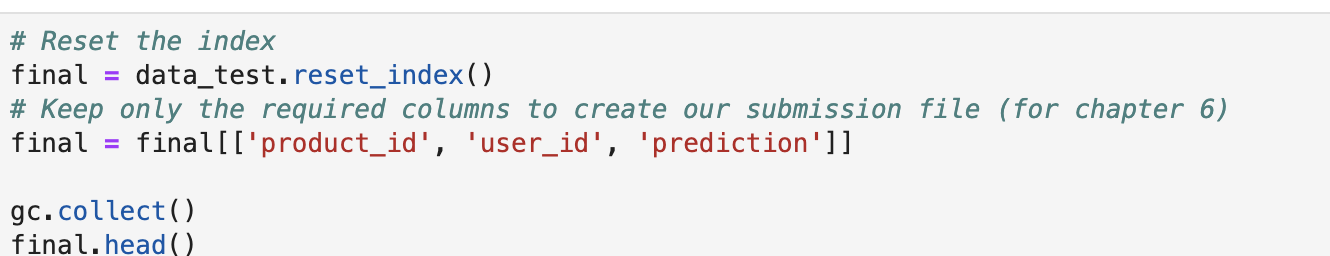
**Creating Predictive model --- XGBoost**

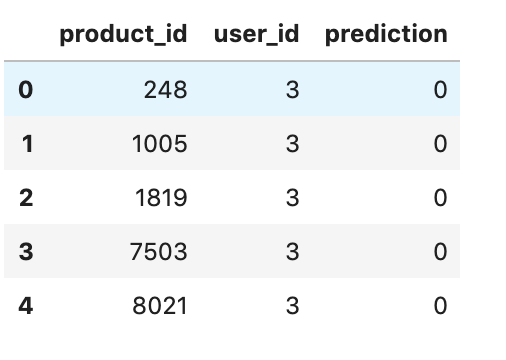
****

**predicting on the testing data**

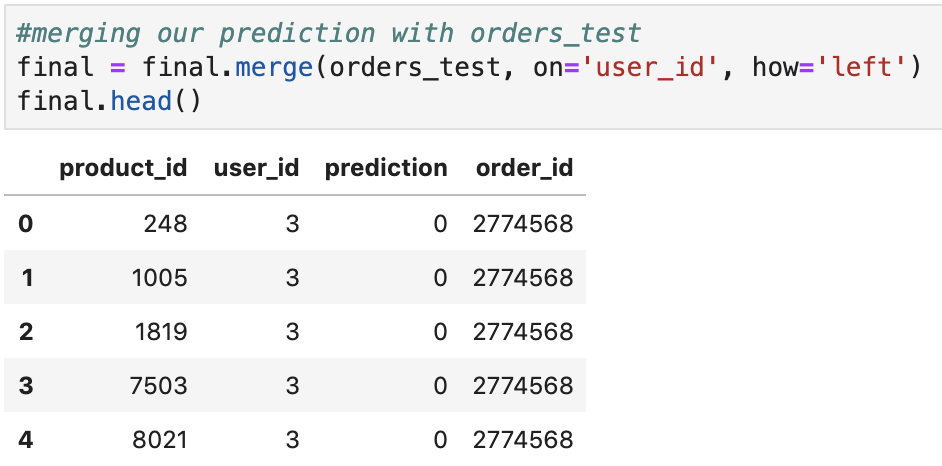
****

****

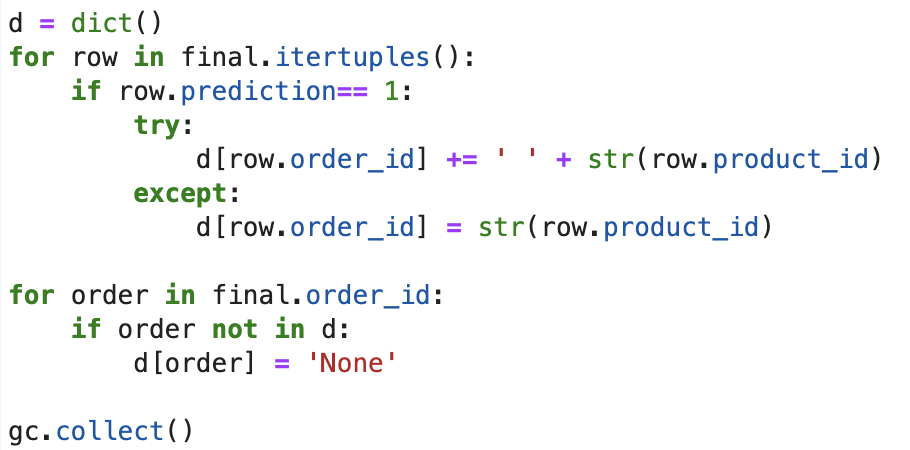
****

****

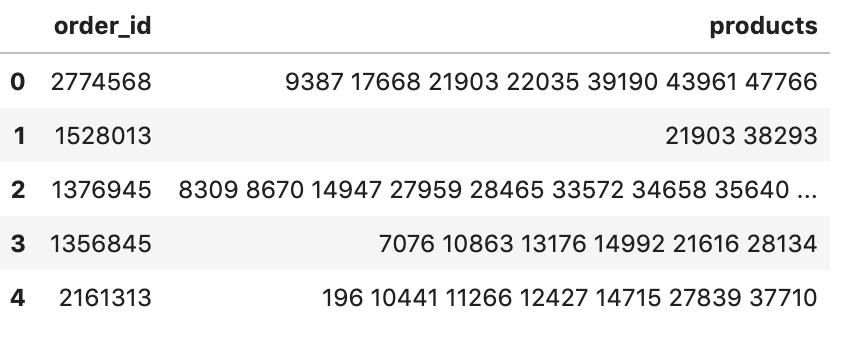
****

****

**We now check how the dictionary were populated (open hidden output)**

****

****

****

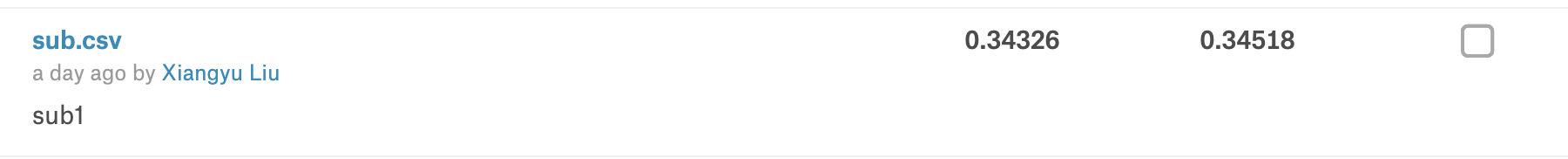
**V. Results**

**5.1 Goal\_1**

**1) LightBGM**

**2) XGboost**

The final results are output as CSV format and after submission in Kaggle, the f1 score is 0.34518.

****

The top 10 popular products according to the prediction

