Capstone Project 1 Report - Predicting InstaCart Reordering

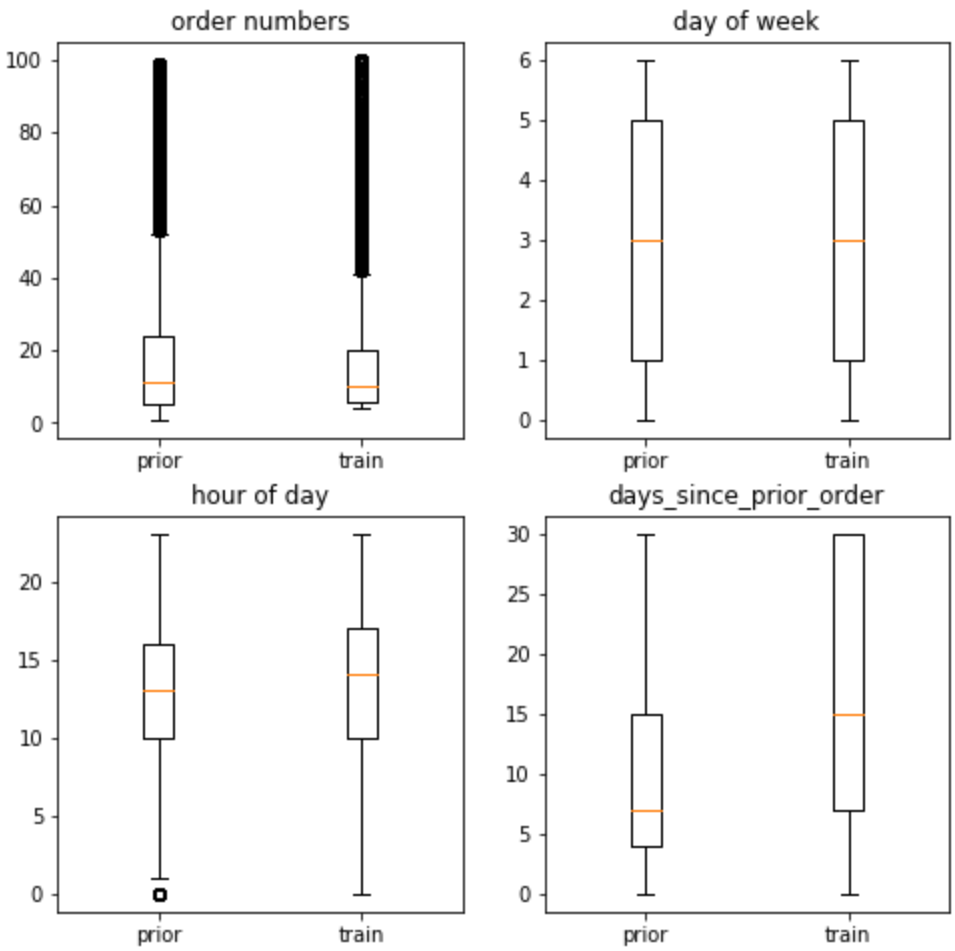
1. Background and motivation

Nowadays in the fast-paced world, it's trendy to shop grocery online and have them delivered to home. Instcart, a grocery ordering and delivery app, uses their historical transactional data to develop models that predict or recommend which products a user will buy again.

This project analyzes over 3 million orders from Instacart's database, explores patterns in customers' ordering habits and predicts what products people will reorder in their next orders. These information can help grocery stores to optimize their warehouse stock and build useful recommendation system for customers.

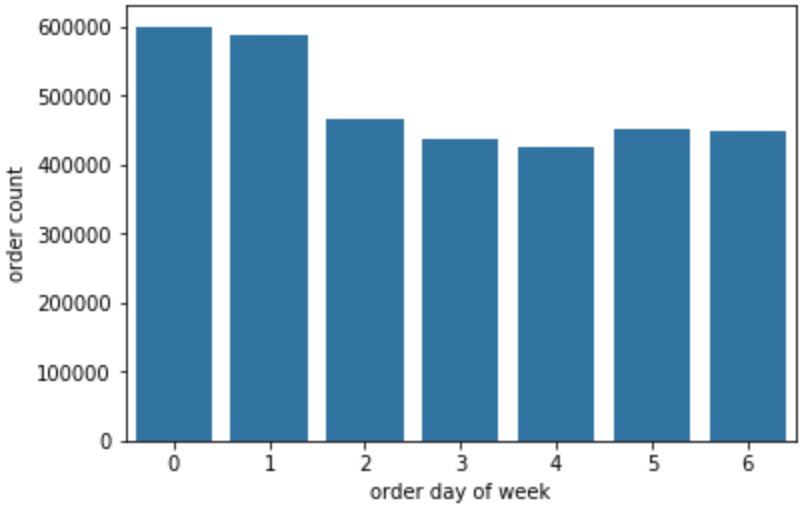
2. Data Wrangling

- There are 6 csv files containing the ordering data in InstaCart Market:
    - orders.csv includes orders in prior, train and test sets with information about order no, user id, order time, and days since prior order
    - aisles.csv includes aisle id and names
    - departments.csv includes department id and names
    - products.csv includes product id, names and corresponding aisle id and department id
    - order_products__prior.csv includes all 32434489 orders in prior dataset with information about order id, product id, add to cart order, whether it is a reorder
    - order_products__train.csv includes all 1384617 orders in train dataset with information about order id, product id, add to cart order, whether it is a reorder
- NaNs are found in column 'days_since_prior_order' of the dataset orders.csv, meaning these orders are first time orders.
- 206209 customers in the "prior" evaluation set of orders.csv are divided into 131209 customers in the "train" dataset and 75000 customers in the "test" set
- No outliner (unreasonable value) is found using boxplot

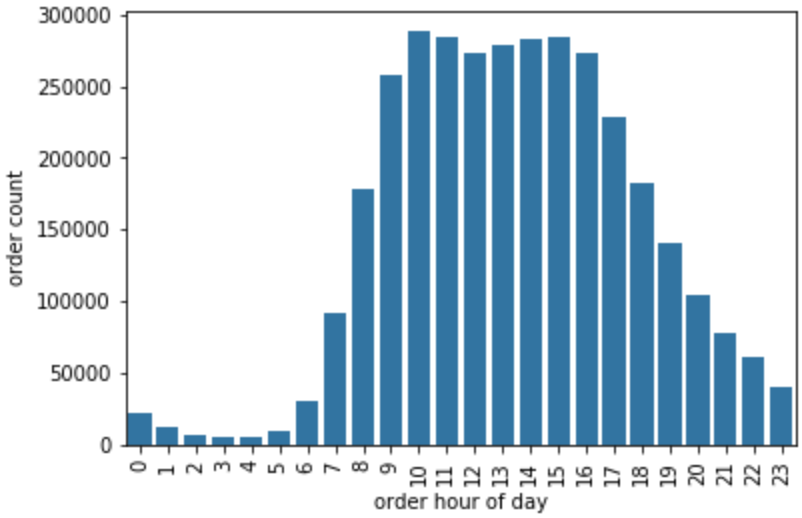


3. Exploratory Data Analysis

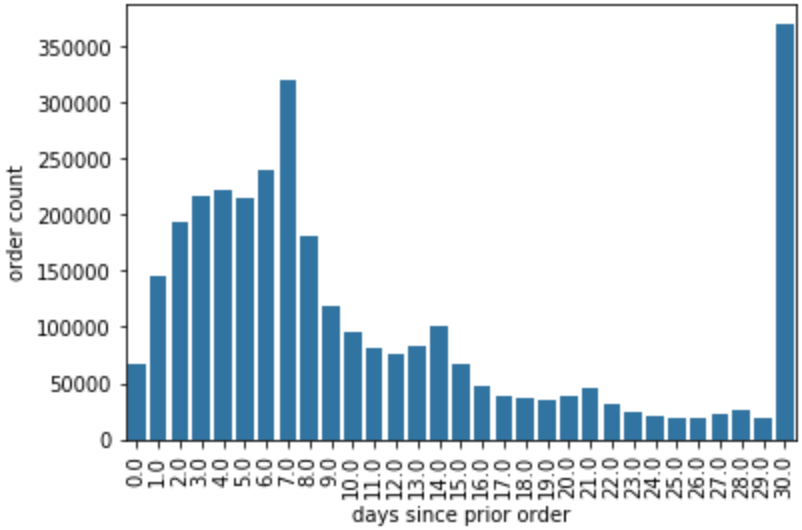
- Histogram of how orders distribute over order day of week shows Day 0 and 1 have higher orders, they should be Saturday and Sunday. Wednesday has the lowest order counts.



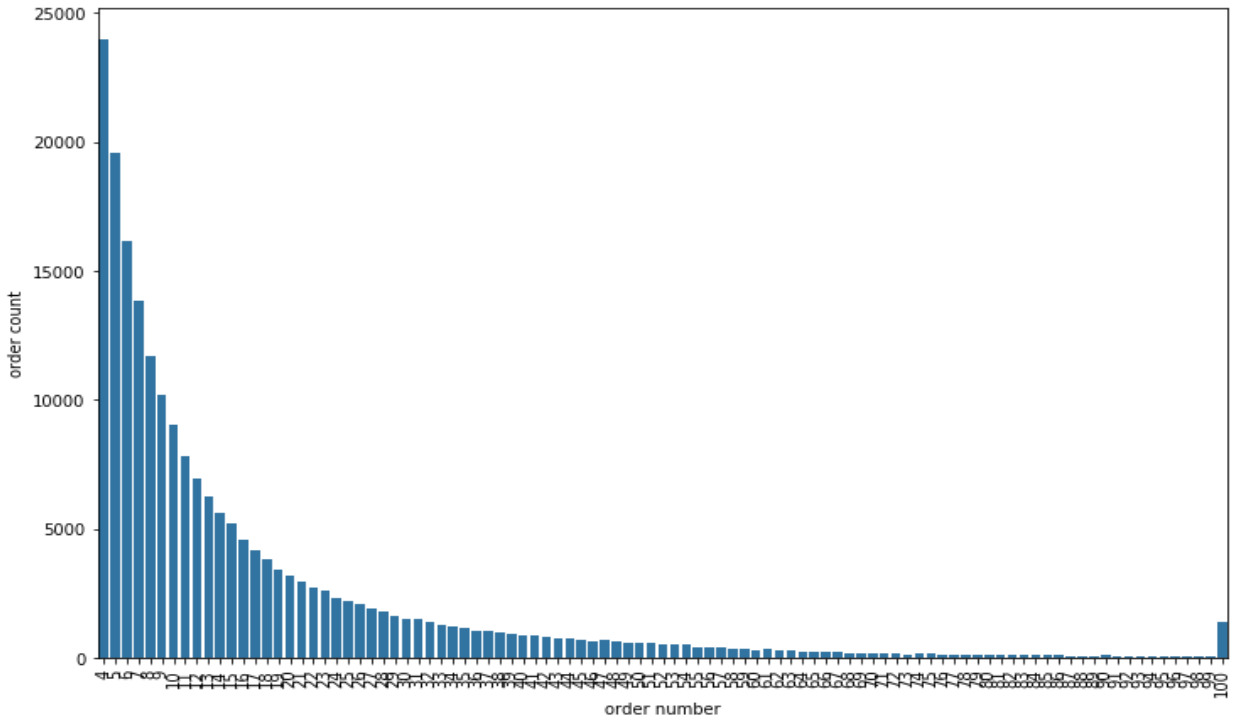
- Distribution of orders over order hour of day shows most orders happen during daytime about 9 am - 4 pm.



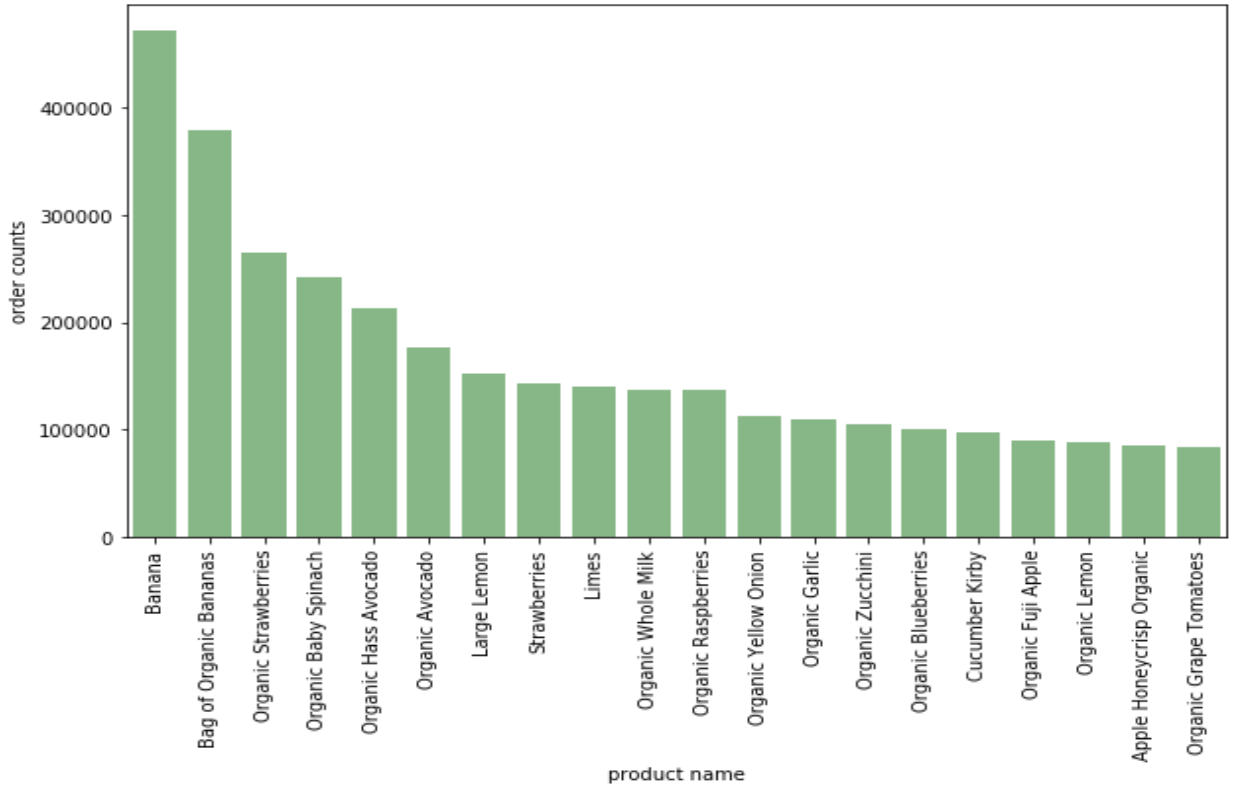
- Order frequency based on days since previous order shows customers most likely order again after 30 days since prior order, followed by 7 days.



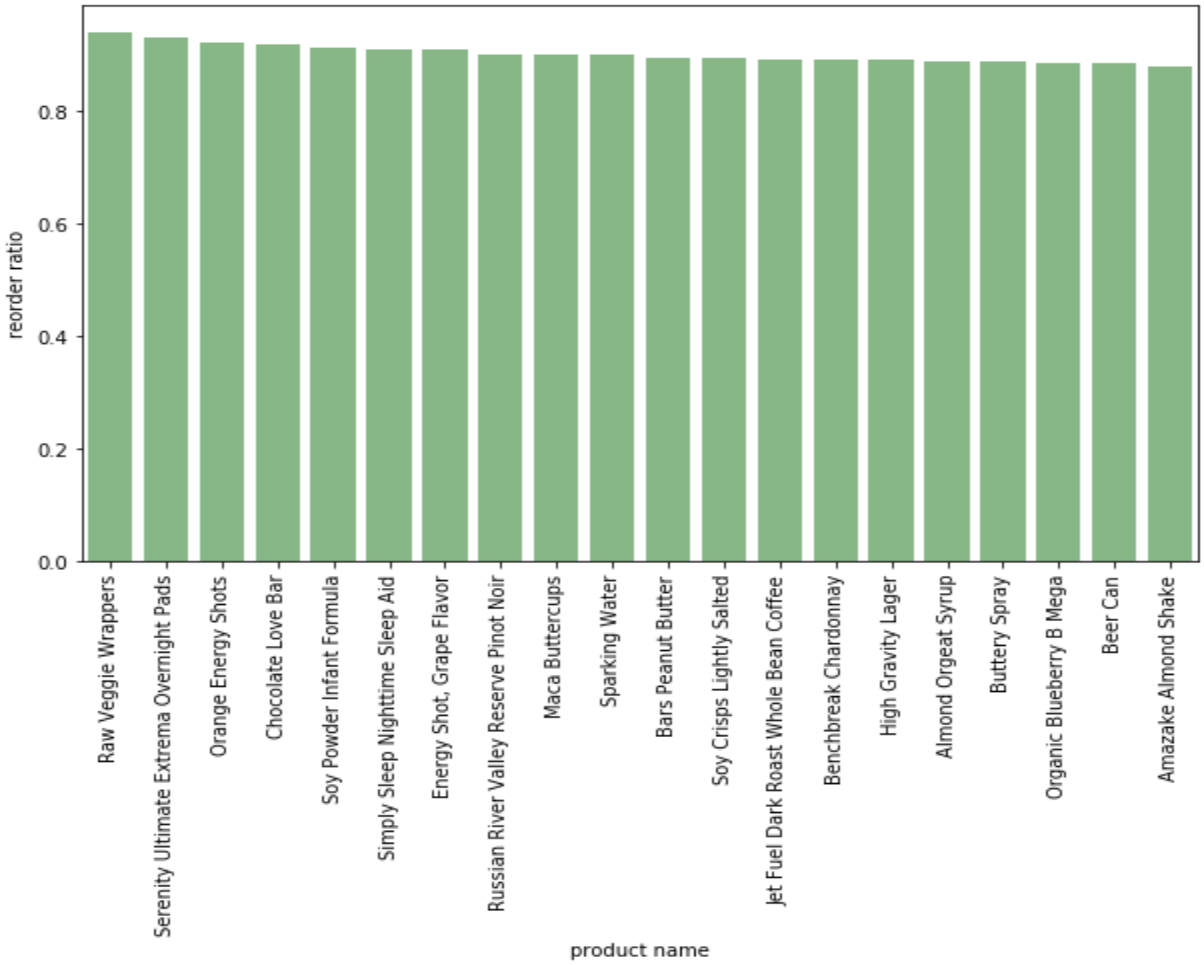
- In the dataset, about 59% of the orders are reorder.
- Customers make at least 4 orders and at most 100 orders.



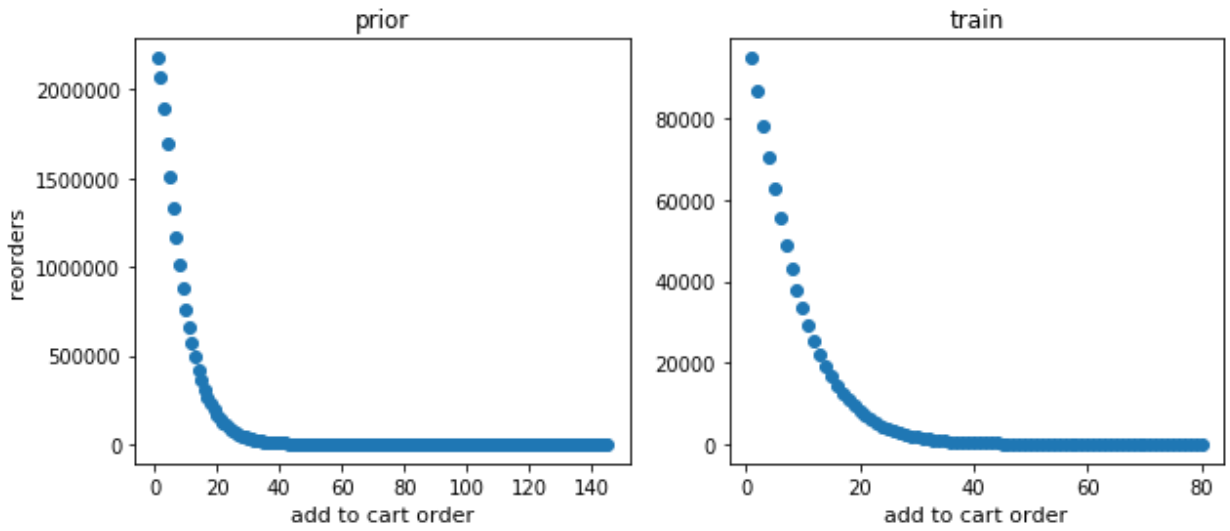
- The top 20 ordered products are mostly fresh fruits and vegetables.



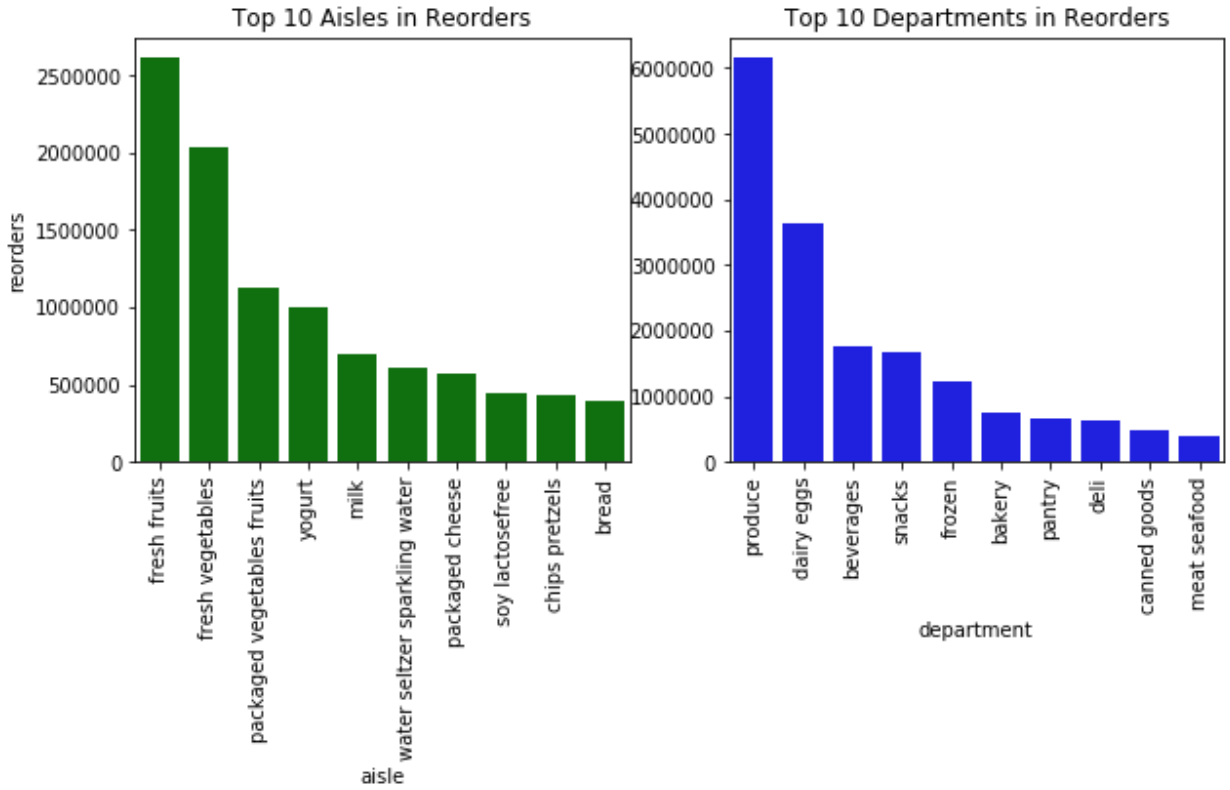
- Products with high reorder ratio are those items whose buyers are "loyal" or have high preference to them. As seen from the plot above, most of the products with highest reorder ratio are kind of snacks, not daily necessaries.



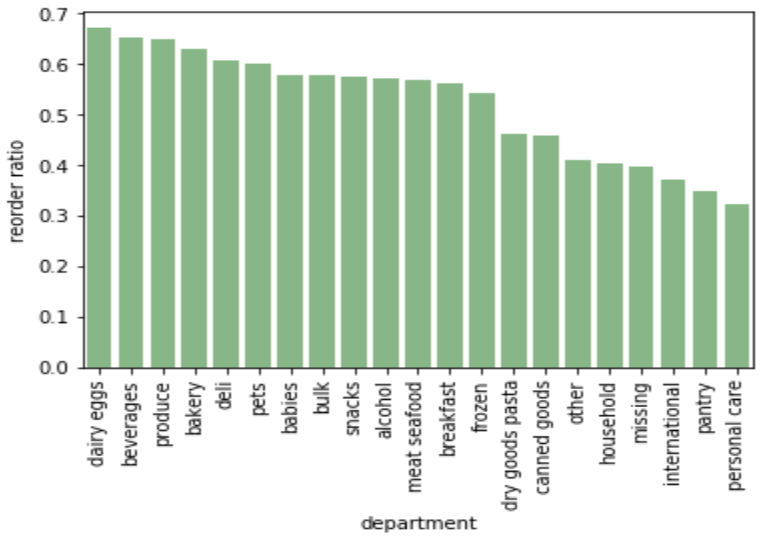
- There is a clear nonlinear pattern between reorder counts and add to cart orders. The products adding to cart first are reordered the most.



- The most reordered aisles are fresh fruits and vegetables. The most reordered departments are produce and dairy eggs.



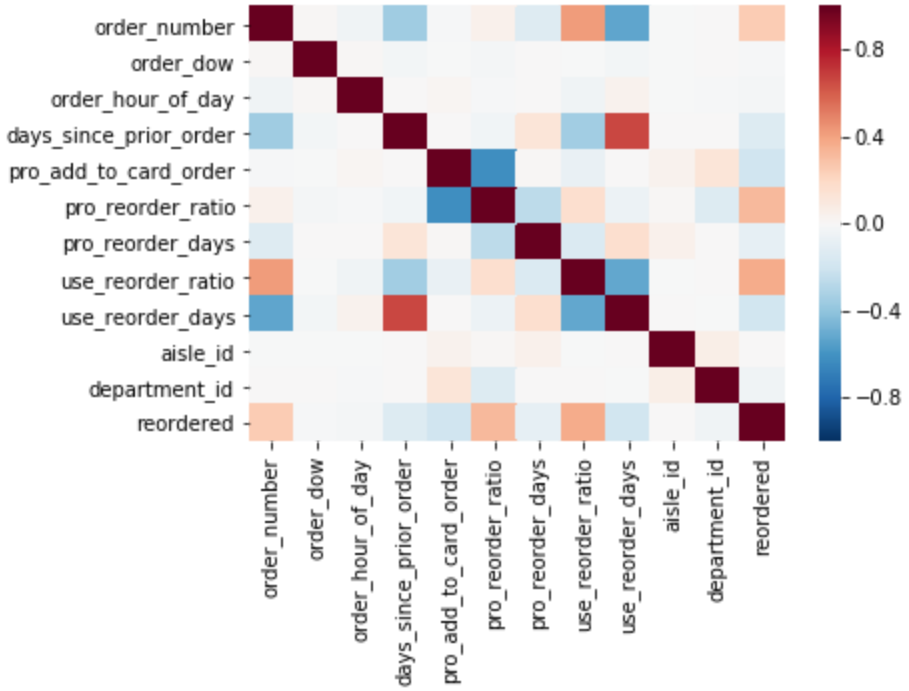
- Reorder ratio is high in dairy eggs, beverages, and produce and lowest in personal care.



- Some customers show strong reordering habits. There are 971 customers always reorder what they buy before. While 3045 customers never reorder what they buy before.

4. Modeling and analysis

- 11 features are selected to predict what customers will reorder in their next order: *order related information* (order_number, order_day_of_week, order_hour_of_day, days_since_prior_order), *product related information* (product_add_to_cart_order, department_id, aisle_id, product_reorder_ratio, product_reorder_days) and *user related information* (user_reorder_ratio, user_reorder_days).
- Among the 11 features, there is noticeable correlation between days_since_prior_order and user_reorder_days. It is expected since user_reorder_days is aggregated from days_since_prior_order at user level. But as it is an important feature characterizing users' reorder habits, both are kept in the model.



- As the transaction data is large and there is limited computation resources (8GB RAM), use a subsample to choose model.
- Logistic regression is firstly used to predict reordering since the output is a binary classification. Scaling is performed first with the predicting variables since the magnitudes of the features vary a lot. A pipeline is built including StandardScaler and LogisticRegression.
- The model is fit to the *prior* set first in order to get parameters of all users. Then the model's performance is evaluated using the *train* set. The model yields reasonable results on both data sets:
  accuracy score on prior set is: 0.7302488238072868
  f1 score on prior set is: 0.8003747416520519
  accuracy score on train set is: 0.6843191648713239
  f1 score on train set is: 0.7257596177253757
- Random Forest is also used to model reordering. Default parameters in Random Forest lead to overfitting. Therefore, grid search cross validation is used to tune hyperparameters including number of trees (n_estimators), max feature in each tree (max_features), max depth of each tree (max_depth), min no of samples to be a leaf (min_samples_leaf). After carefully choosing hyperparameters, Random Forest is able to return slightly better results than Logistic Regression:
  accuracy score on train set is: 0.692741085699057
  f1 score on train set is: 0.7416314173708934
  Random Forest also returns feature importance and it is found user reorder ratio is the most important feature among the 11.

- Then the pipeline is performed to *test* set, and predict products that customers will reorder and output to “submission.csv” in desired format.

5. Summary

Random Forest and Logistic Regression both return reasonable results. Random Forest slightly outperform Logistic Regression in accuracy and f1 score after hyperparameter tuning. Logistic Regression is more computationally effective.