Capstone Project 1 Report - Predicting InstaCart Reordering

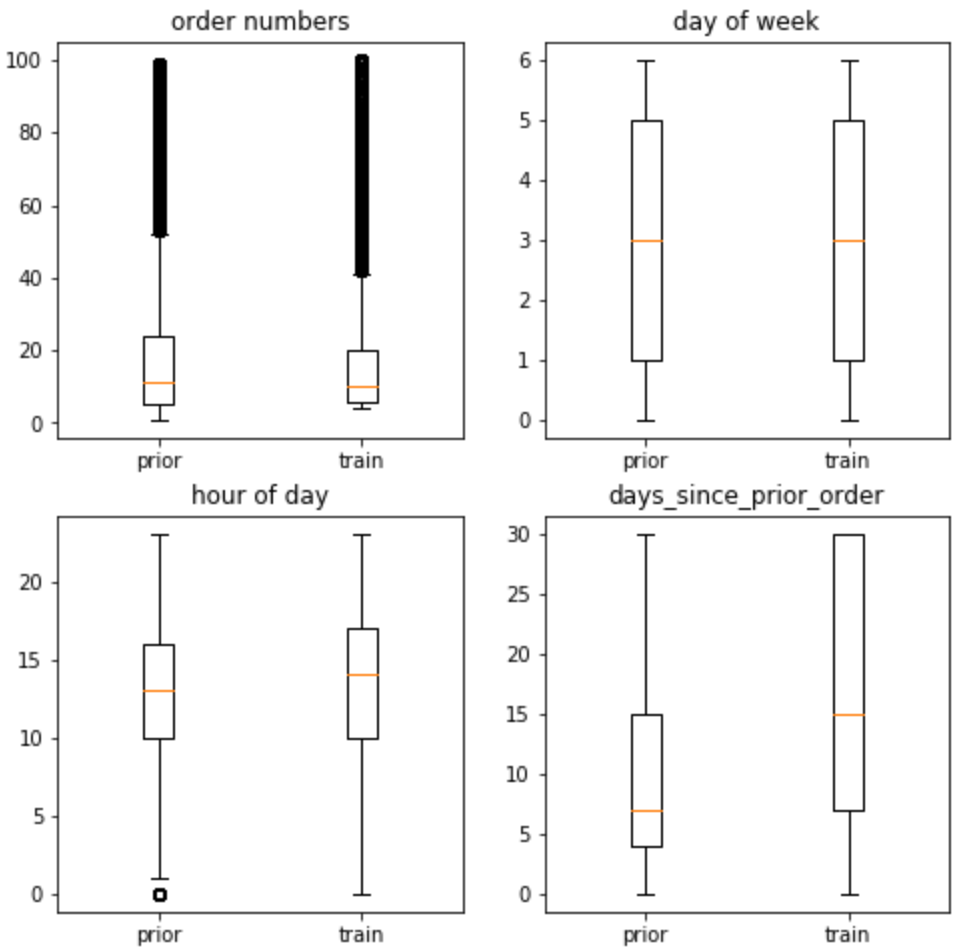
1. Background and motivation

Nowadays in the fast-paced world, it's trendy to shop grocery online and have them delivered to home. Instacart, 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.

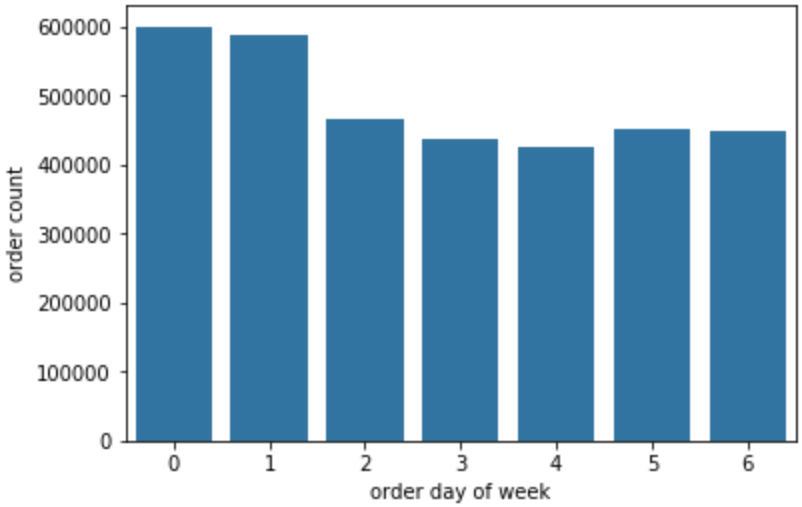
2. Data Wrangling

- 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

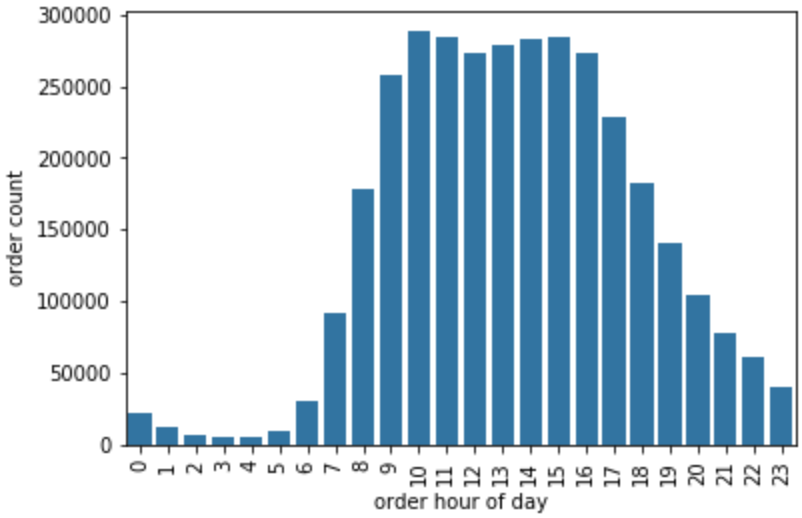


3. Exploratory Data Analysis

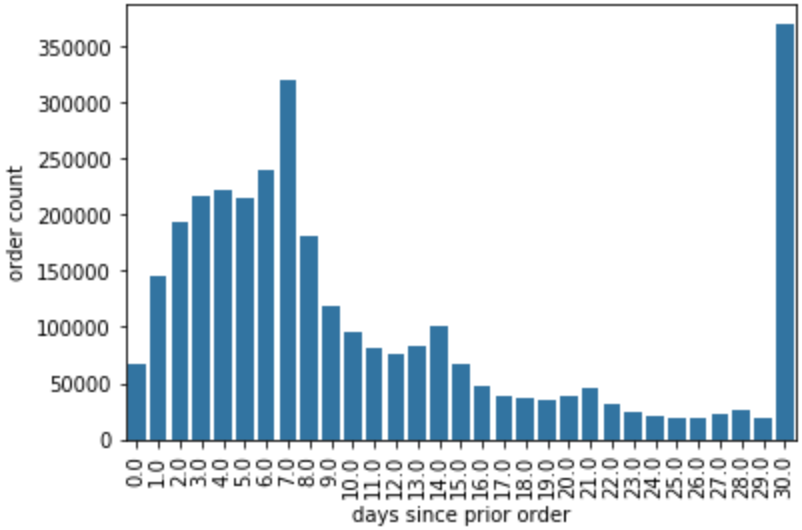
- Histogram of how orders distribute over order day of week shows Day 0 and 1 have higher orders, they should be Saturday and Sunday. Wednesday has the lowest order counts.



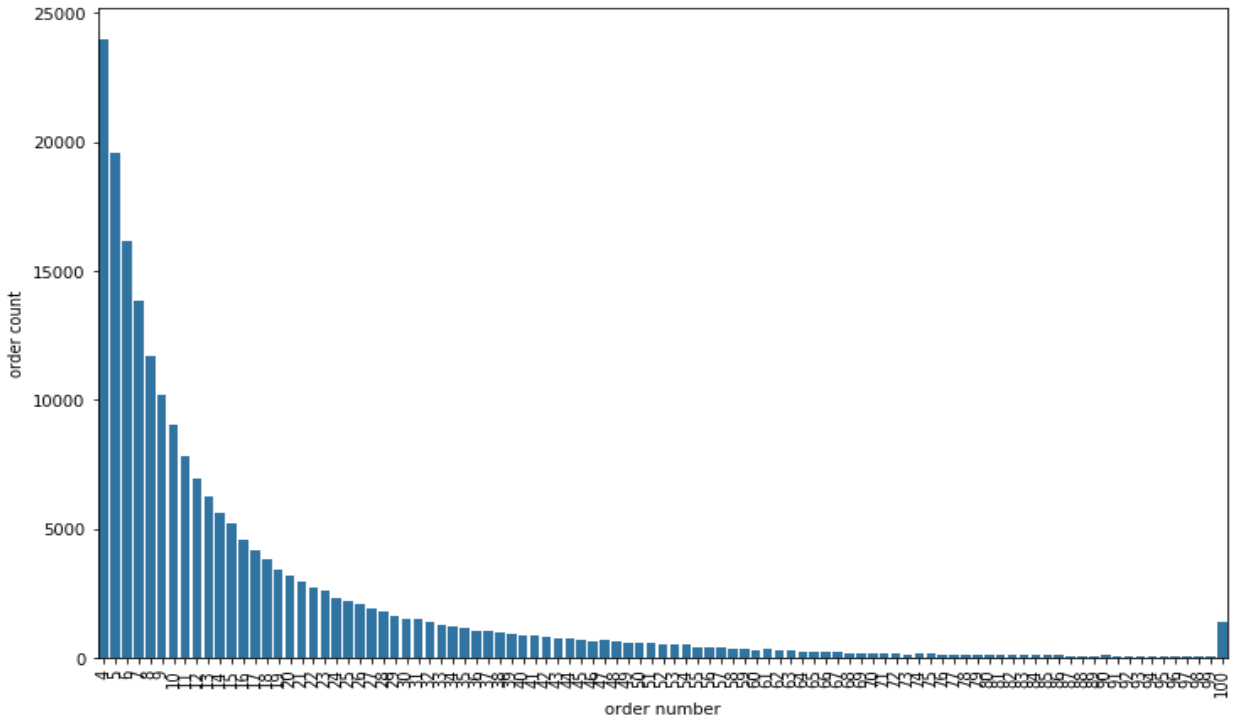
- Distribution of orders over order hour of day shows most orders happen during daytime about 9 am - 4 pm.



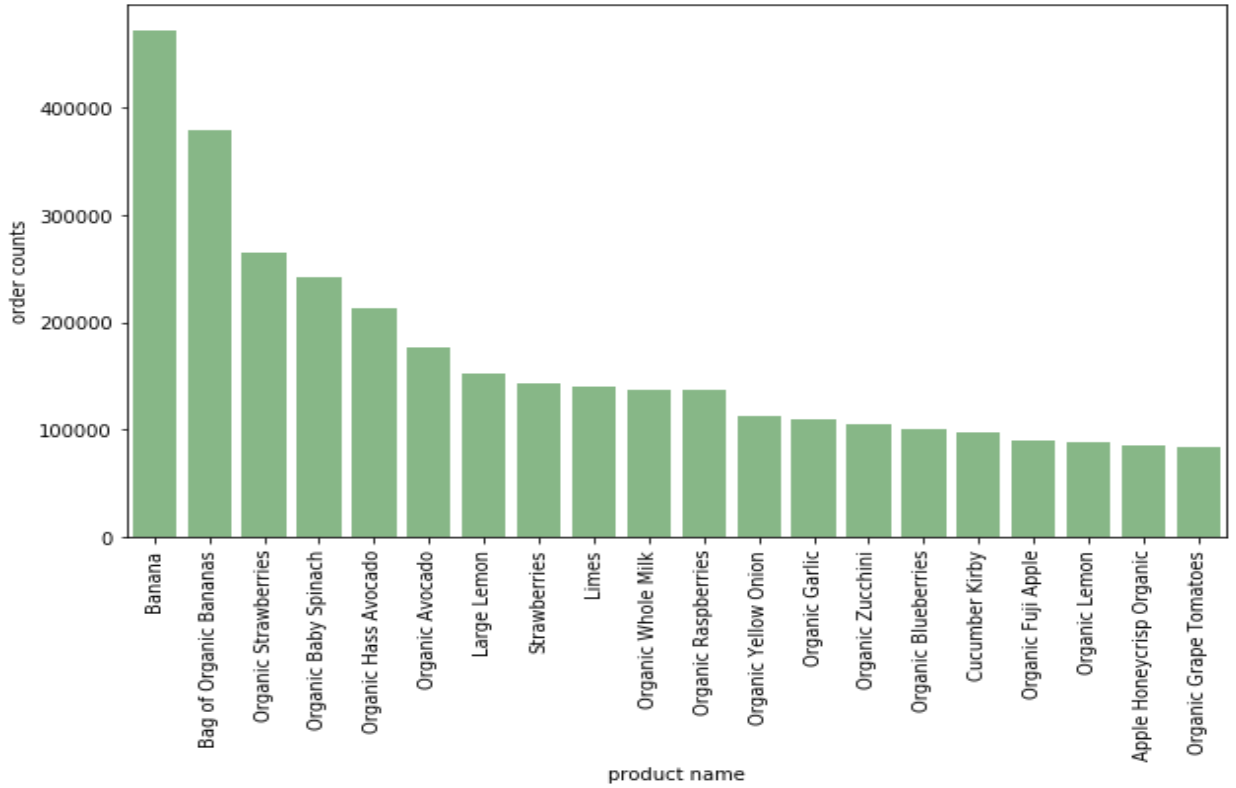
- Order frequency based on days since previous order shows customers most likely order again after 30 days since prior order, followed by 7 days.



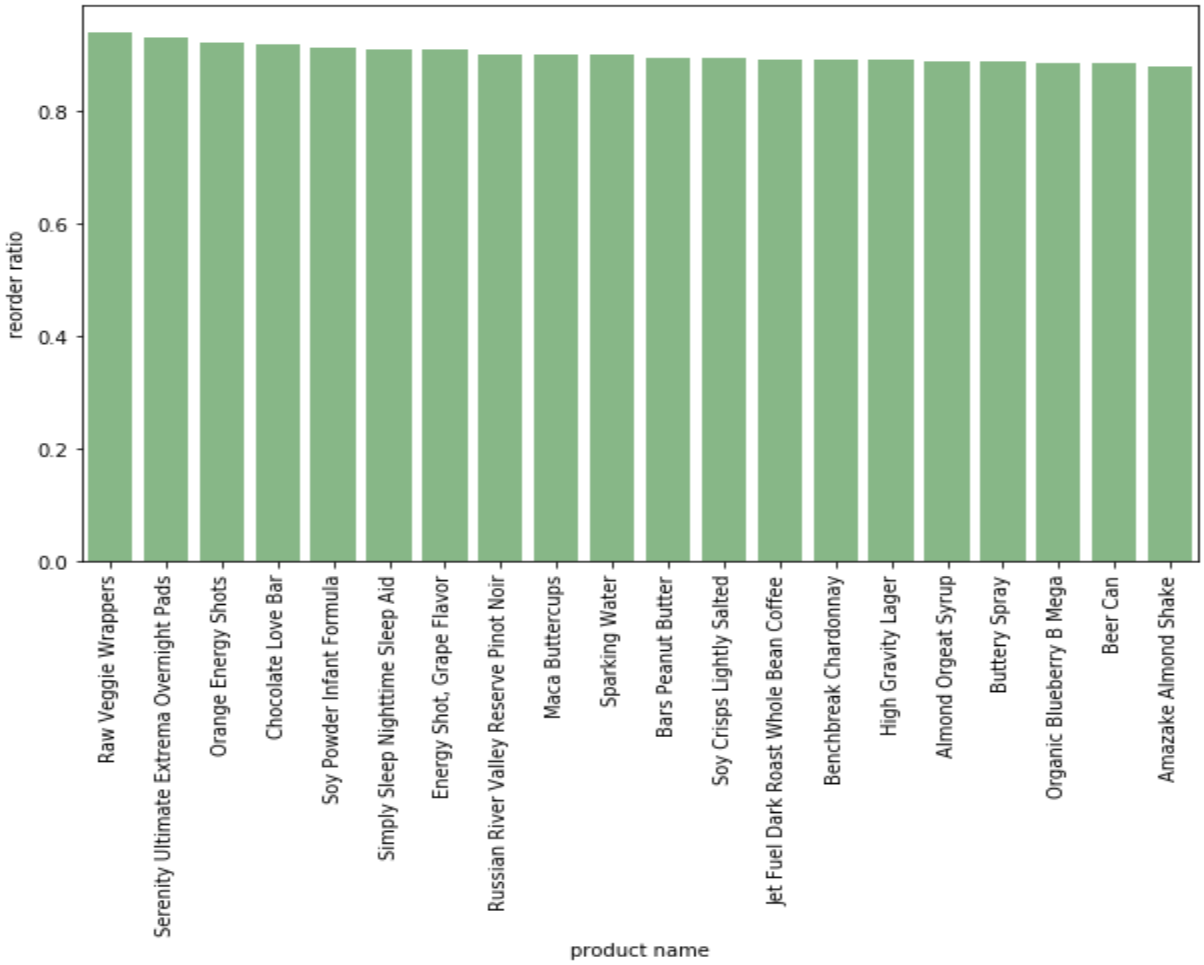
- In the dataset, about 59% of the orders are reorder.
- Customers make at least 4 orders and at most 100 orders.



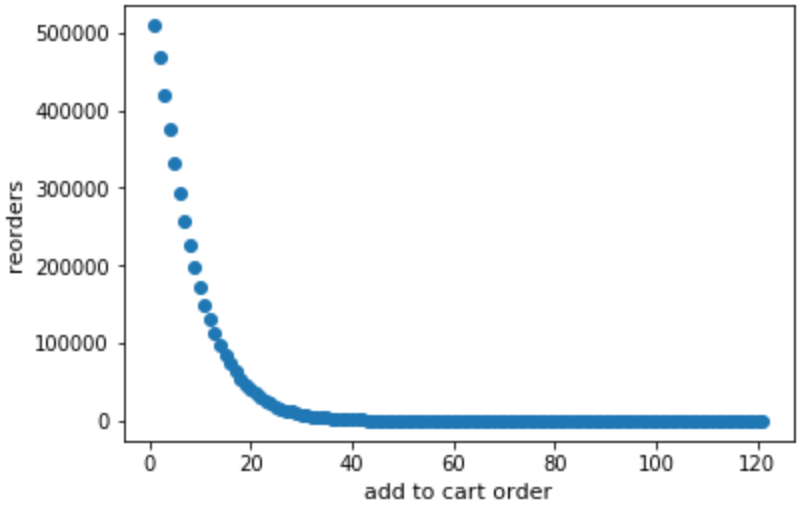
- The top 20 ordered products are mostly fresh fruits and vegetables.



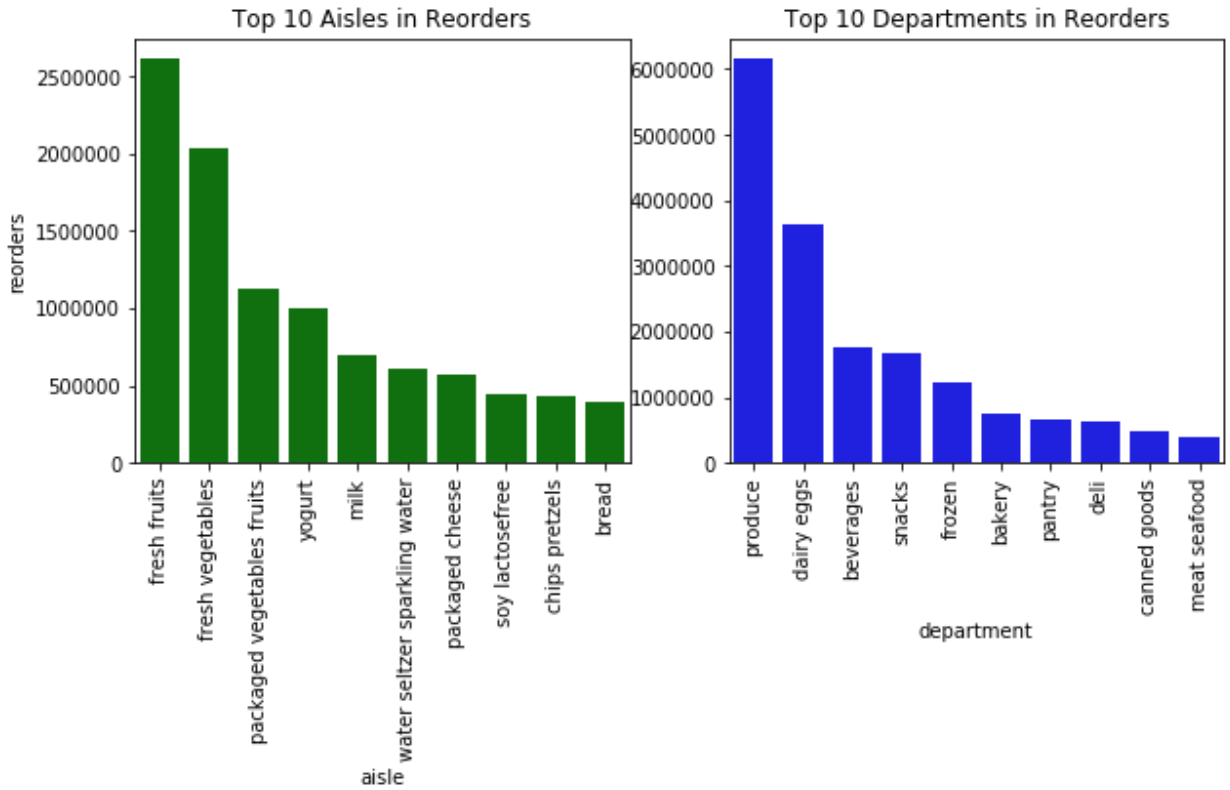
- Products with high reorder ratio are those items whose buyers are "loyal" or have high preference to them. As seen from the plot above, most of the products with highest reorder ratio are kind of snacks, not daily necessaries.



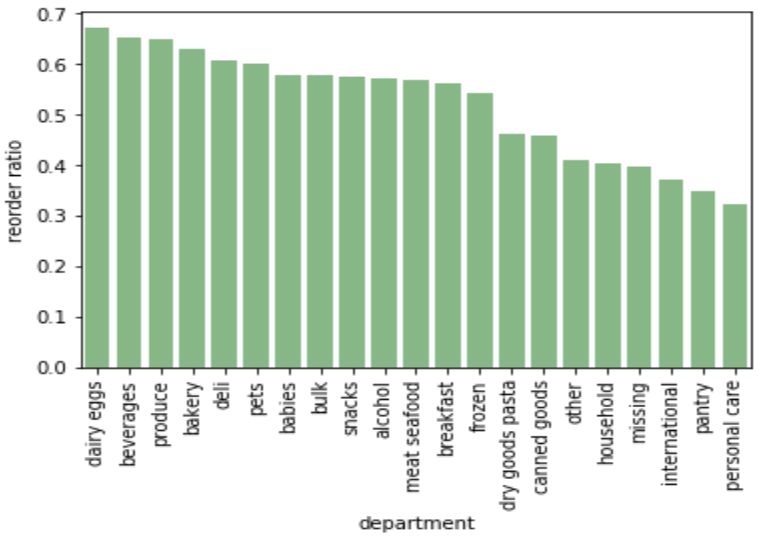
- There is a clear nonlinear pattern between reorder counts and add to cart orders. The products adding to cart first are reordered the most.



- 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 3.27% customers always reorder what they buy before. While 1.08% customers never reorder what they buy before.

4. Feature Selection

- Dataset to train models include orders from “train” set and 3 most recent orders from “prior” set. If a user does not order in “train” set then 4 most recent orders from “prior” set are used.
- Four groups of feature are selected to predict whether a product will be reordered:

  User features:

  - how many days between user's orders
  - user order size (how many items the user buy in one order)
  - order organic products

  Product features:

  - product order frequency
  - product purchase probability across a week
  - product purchase probability in a day
  - product add to cart order
  - aisle
  - department

  User-product features:

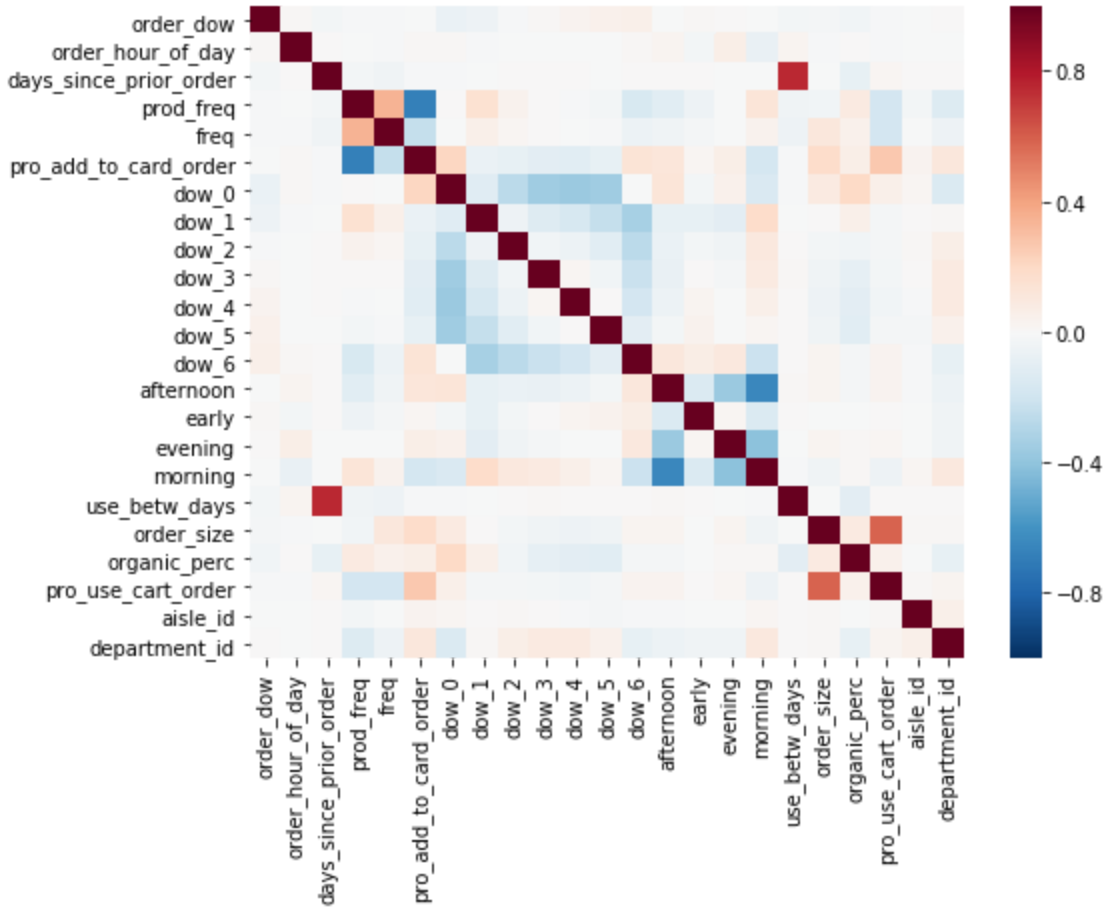
  - add to cart order at user level

- frequency of purchasing a product for a user

Date-time information:

- order dow
- order time
- days_since_prior_order

- 23 features are calculated to represent the four groups of information:
  - Date time: 'order_dow', 'order_hour_of_day','days_since_prior_order'
  - User: 'use_betw_days', 'order_size', 'organic_perc'
  - Product: 'prod_freq', 'pro_add_to_card_order', 'dow_0', 'dow_1', 'dow_2', 'dow_3', 'dow_4', 'dow_5', 'dow_6', 'afternoon', 'early', 'evening', 'morning','aisle_id','department_id'
  - User-product: 'freq','pro_use_cart_order'
- As feature "dow_6" and "early" can be calculated from other features, they are not kept as independent variables. Although user in-between days is correlated with days_since_prior_order, it represents user ordering habits and is kept as an independent variable.



5. Modeling and analysis

- As the transaction data is large and there is limited computation resources (8GB RAM), use a subsample (10%) for model selection and hyperparameter tuning.
- **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 varies. A pipeline is built including StandardScaler and LogisticRegression.
- The model is fit to the train dataset and yields decent results :
  accuracy score on train set is: 0.7622206294563398
  f1 score on train set is: 0.7850949596092045
  area under roc curve on train set is: 0.7644140378161415
  Then the model predicts product reordering on the test dataset. The prediction output is formatted as required. The results from Logistic Regression were submitted to Kaggle, scoring 0.3317, compared to the 1st place of 0.4.
- **Random Forest** is also used to model reordering. 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 f1 score than Logistic Regression:
  accuracy score on train set is: 0.7601503298624064
  f1 score on train set is: 0.7919928182151138
  area under roc curve on train set is: 0.7547594811199003
  Based on the feature importance score, *add to cart order* and *product order frequency at user level* are the most important features.
- Then the model is used to predict reorders of test set, and predicted results yield a score of 0.3335 after submission to Kaggle, which is slightly better than that from Logistic Regression.
- **Stochastic Gradient Descent** is used to predict reordering. A grid search cross validation is performed to select loss function: 'log', 'squared_hinge', alpha: [0.001,0.01], max_iter: [500, 1000]. In the end, best parameters are 'log', alpha = 0.01, max_iter = 500.
- F1 score on the training set are slightly better than the original Logistic Regression, but lower than that of the Random Forest.
  accuracy score on train set is: 0.7613826510492715
  f1 score on train set is: 0.7859113380571253
  area under roc curve on train set is: 0.7622414740360158

- Then the model predicts reorders of test set, and predicted results yield a score of 0.3326, which is slightly better than that from Logistic Regression but lower than Random Forest.

6. Summary

Random Forest, Logistic Regression and Stochastic Gradient Descent all return reasonable results. Random Forest slightly outperform Logistic Regression in f1 score after hyperparameter tuning. Logistic Regression is more computationally effective. Logistic Regression using Stochastic Gradient Descent returns slightly better f1 score for training and test than original Logistic Regression, not as good as result of Random Forest though.

The model can be improved by including more features such as in-between days for products. It is not incorporated in the current model due to computational limitation since calculation of this feature is time consuming.

Reorder predicting is useful to help the InstaCart to recommend products to users and make sure enough items are in stock when customers are likely to reorder, save space and avoid waste when customers are less likely to purchase.