DATA ANALYTICS DESIGN REPORT

The Insights Group (Chris Gao, Huy Nguyen, Shuoyang Li, Shuwei Liu)

ABSTRACT:

Instacart is an internet company that operates same-day grocery delivery service. In this project, we explored ways to predict whether an Instacart user would reorder a certain product in his or her next order, which can be used to implement highly personalized marketing strategies. After consolidating and exploring the Instacart datasets, we identified three feature categories, namely user-behavior-based, product-based and user-product-cross-based features. Having processed all available data, we finally decided on using the XGBoost algorithm as the primary model as it yielded the best accuracy of 75%.

DATASET DESCRIPTION:

The dataset we used for this project includes 3 million of Instacart's anonymized customer orders from 200,000 Instacart users (4 - 100 orders for each user). Specifically, "order.csv" specifies all order ids of all customers as well as the time at the order and the time since a customer's last order. "Order_products_prior.csv" expands each order id that is not the last order of a customer, including in every row the product's name and sequence he/she puts it in a cart. "Order_products_train.csv" expands all the last order ids of customers, which we turned into labels used for prediction. The data is located on the web and can be found at: Instacart Data (Kaggle).

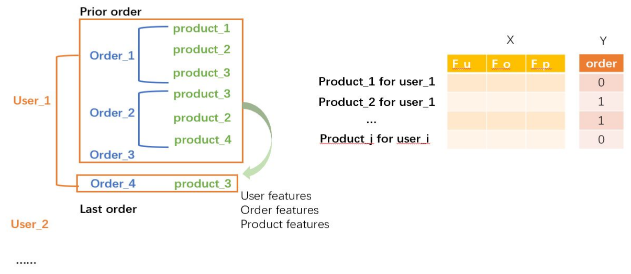
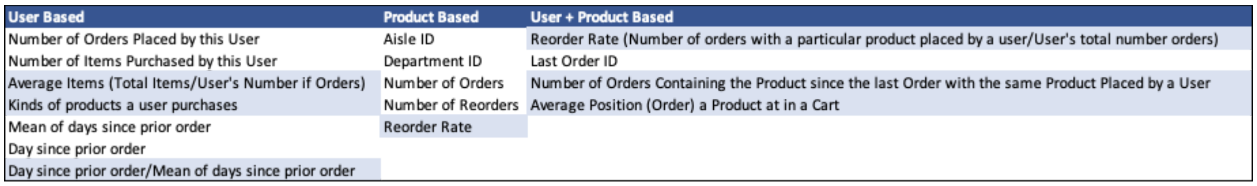


Figure 1. The Structure of Instacart's dataset before and after transformation

DATA PROCESSING PROCEDURES:

Since the Instacart data came in the form of multiple CSV files, we consolidated these data by merging them into a single data frame. We began by categorizing the Instacart data into three groups: user-behavior-based, product-based, and user-product-cross based features. Table 1 shows a detailed list of all features we used in the model. Note that in addition to the provided features, we also **created our own features** for better analysis (which are shaded in the table). For instance, we created the "Reorder Rate" of a product under the user-product-cross-based category. We obtained this number by dividing the number of orders containing a particular product placed by a user by this user's total number of orders. This turned out to be a key feature.



User Based	Product Based	User + Product Based
Number of Orders Placed by this User	Aisle ID	Reorder Rate (Number of orders with a particular product placed by a user/User's total number orders)
Number of Items Purchased by this User	Department ID	Last Order ID
Average Items (Total Items/User's Number if Orders)	Number of Orders	Number of Orders Containing the Product since the last Order with the same Product Placed by a User
Kinds of products a user purchases	Number of Reorders	Average Position (Order) a Product at in a Cart
Mean of days since prior order	Reorder Rate	
Day since prior order		
Day since prior order/Mean of days since prior order		

Table 1. List of all input features

While we cleaned the data, we also visualized some features and discovered a few interesting observations. The left plot shows the distribution of orders by days since a prior order. Note that day 7 and day 30+ have extremely high number of orders, suggesting that a user have a higher probability to place an order if he or she hasn't done so for the past 7 or 30+ days (weekly and monthly basis, respectively). The plot on the right shows the distribution of orders among departments. Here we see that products under the "produce" department have a higher probability to be purchased.

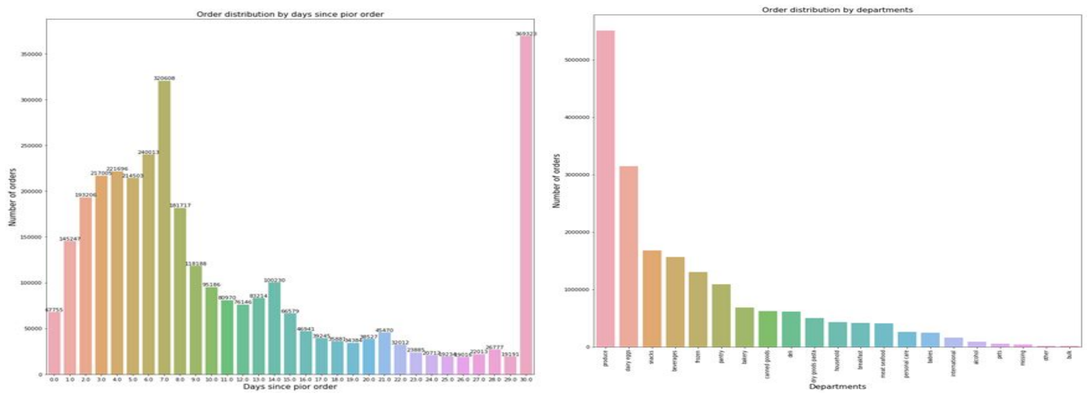


Figure 2. Order distribution by date since previous order (left) and department (right)

We wanted to build a model for every input pair of (user_id, product_id) from previous order. First we generated a basket containing only the products that this user has bought, because the prediction of the products that did not appear in the user's previous order were not useful. Our model will output whether that customer has put those products in the basket in his or her latest order. To achieve this, we first created the labels by checking whether the product was in the latest order.

Every row of our data frame represents a unique set of user and product with paired index. This was a challenge because we dealt with a massive dataset and could not use for-loops in searching for data and building our final data frame. On the other hand, for the user and product features, we used functions like "groupby" and "join" which was less difficult. The prime objective for us was **how to construct User Product cross-based features**. For instance, to get the number of times a customer bought a certain product, we created unique ids that represented a unique set of user and product pair and created a dictionary that recorded the total number of times that the customer bought the product in the past (which was the only for loop we used) and mapped the results to our final data frame based on the unique id. In the end, our program managed to convert the raw data into our final data frame with the size of 8 million rows by 16 columns in less than 5 minutes, which is a very satisfying result.

PREDICTIVE METHODS:

For this project we used XGBoost (Extreme Gradient Boosting) model. Gradient boosting tree is an ensemble learning algorithm that iteratively learn weak classifiers and add them to a final strong classifier to produce final predictions. In each round we learned a new tree to approximate the negative gradient and minimize the losses. The parameters we used for the XGBoost included: "min_child_weight" is the minimum weight that a node can have; "gamma" is the tree size penalty; "subsample" indicates bagging (sampling with replacement a proportion of the training set) to reduce the level of overfitting; "colsample_bytree" allows us to perform bagging on a proportion of features to construct the trees, which also helps to reduce overfitting; "max_depth" is the maximum depth of the tree. The disadvantages of XGBoost are that it is not good at extrapolating unseen values. Therefore if a user nevers bought a product before, our model would never predict it in that user's reorder basket. Moreover, we split the data into training set and testing set and also did cross-validation to prevent overfitting.

DISCUSSION AND RESULTS:

The AUROC score for our model using XGBoost is 83%. We also analyzed the feature importance to figure out which features contribute the most to the prediction and explain the most variance in our labels. It turned out that the most important feature was **the number of orders since the previous order**. This feature is a user-product-cross-based feature which was generated by us. Intuitively, it is correct to think that a customer is more likely to buy a product if he has bought it just recently. The second and third important feature are **the reorder rate of a product** and the **total number of different product the customer has bought before**. We can see from the result that the most three important features are from each of the three feature categories respectively. This demonstrated how our feature engineering work could help increase the prediction accuracy. In addition to predictions, we suggest Instacart to increase marketing efforts to customers with shorter purchase frequency and more diverse basket to maximize their revenues. They can also look into their product portfolio to improve or cut products with low reorder rates to reduce costs. Moreover, based on the prediction results, Instacart can produce the most relevant and personalized recommendations for users when they use the app.