Instacart Market Basket Analysis 2nd place solution

I made two models for predicting reorder & None. Following are the features I made.

Features

User feature

- How often the user reordered items
- Time between orders
- Time of day the user visits
- Whether the user ordered organic, gluten-free, or Asian items in the past
- Features based on order sizes
- How many of the user’s orders contained no previously purchased items

Item feature

- How often the item is purchased
- Position in the cart
- How many users buy it as "one shot" item
- Stats on the number of items that co-occur with this item
- Stats on the order streak
- Probability of being reordered within N orders
- Distribution of the day of week it is ordered
- Probability it is reordered after the first order
- Statistics around the time between orders

User x Item feature

- Number of orders in which the user purchases the item
- Days since the user last purchased the item
- Streak (number of orders in a row the user has purchased the item)
- Position in the cart
- Whether the user already ordered the item today
- Co-occurrence statistics
- Replacement items

datetime feature

- Counts by day of week
- Counts by hour

More detail, please refer to codes.

F1 maximization

Regarding F1 maximization, I hadn't read that paper until Faron had published the kernel. But I got high score because of my F1 maximization. Let me explain it. For maximizing F1, I generate y_true according to predicted prob. And check F1 from higher prob. For example, lets say we have ordered item and prob, like {A: 0.3, B:0.5, C:0.4}. Then generate y_true in many times. In my case, generated 9999 times. So now we have many of y_true, like [[A,B],[B],[B,C],[C],[B],[None].....]. As I mentioned above, next thing we do is to check F1 from [B], [B,C], [B,C,A]. Then we can estimate F1 peak out, and stop calculation, and go next order. You may know, in this method, we don't need to check all pattern, like [A],[A,B],[A,B,C],[B]... I guess some might have figured out this method from my comment of "tips to go farther". However, this method is time consuming as well as depends on seed. So finally I used Faron's kernel. Fortunatelly or not, I got almost same result using Faron's kernel. Please refer to py_model/pyx_get_best_items.pyx

How to run

- cd py_feature
- python 901_run_feature.py
- python 902_run_concat.py
- cd ../py_model
- python 999_run.py

Requirements

Around 300 GB RAM needed(sorry). But I confirmed we can get 0.4073 on private LB with only around 60 GB RAM. Also if you don't have enough memory and want to get high score, try continuous training using xgb_model of xgb.train.

Python packages:

- numpy==1.12.1
- pandas==0.19.2
- scipy==0.19.0
- tqdm==4.11.2
- xgboost==0.6