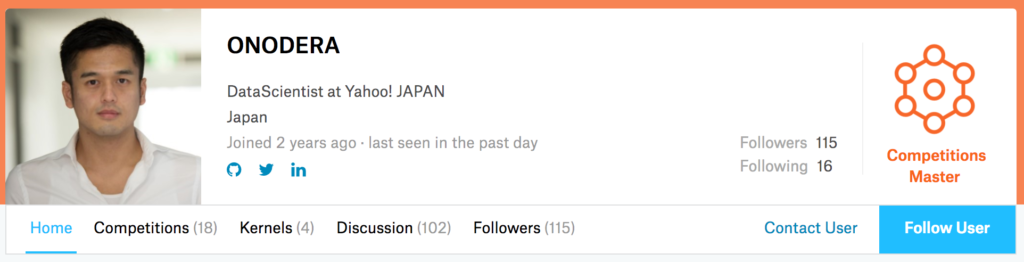
# Instacart Market Basket Analysis, Winner’s Interview: 2nd place, Kazuki Onodera Instacart市场篮分析，赢家访谈：第二名，小野寺一夫

原文链接：  
原文链接：

Our recent challenged Kagglers to predict which grocery products an Instacart consumer will purchase again and when. Imagine, for example, having milk ready to be added to your cart right when you run out, or knowing that it’s time to stock up again on your favorite ice cream.  
我们最近向Kagglers提出挑战，要求他们预测Instacart消费者将在何时再次购买哪些食品杂货。例如，想象一下，当你用完牛奶后，牛奶就可以被添加到你的购物车里，或者你知道是时候再去买你最喜欢的冰淇淋了。

This focus on understanding temporal behavior patterns makes the problem fairly different from standard item recommendation, where user needs and preferences are often assumed to be relatively constant across short windows of time. Whereas Netflix might be fine assuming you want to watch another movie similar to the one you just watched, it’s less clear that you’ll want to reorder a fresh batch of almond butter or toilet paper if you bought them yesterday.  
这种对理解时间行为模式的关注使得这个问题与标准的项目推荐大不相同，在标准的项目推荐中，用户的需求和偏好通常被认为在短时间内是相对恒定的。如果你想看另一部类似于你刚刚看的电影的话，Netflix可能会很好，但如果你昨天买了一批新鲜的杏仁酱或卫生纸的话，你想重新订购就不那么清楚了。

We interviewed (aka on Kaggle), a data scientist at Yahoo! JAPAN, to understand how he used complex feature engineering, gradient boosted tree models, and special modeling of the competition’s F1 evaluation metric to win 2nd place.  
我们采访了雅虎的数据科学家（又名Kaggle）！日本，为了了解他如何运用复杂的特征工程、梯度增强树模型和特殊的F1评价指标来赢得第二名。



img  
img公司

# Basics 基础

What was your background prior to entering this challenge?  
在参加这次挑战之前你的背景是什么？

I studied Economics in university, and I worked as a consultant in the financial industry for several years. In 2015, I won 2nd place in the KDD Cup 2015 challenge, where the goal of the challenge was to predict the probability that a student would drop out of a course in 10 days. Now I work as a data scientist for Yahoo! JAPAN.  
我在大学学的是经济学，在金融业做了几年顾问。2015年，我在KDD Cup 2015挑战赛中获得第二名，挑战的目标是预测学生在10天内辍学的概率。现在我在雅虎做数据科学家！日本。

How did you get started competing on Kaggle?  
你是怎么开始和卡格尔比赛的？

I joined Kaggle about 2 years ago after one of my colleagues mentioned it to me. My first competition was the Otto Product Classification Challenge. Since the features in that challenge were obfuscated, I couldn’t perform any exploratory data analysis or feature engineering, unlike what I did here.  
大约两年前，在我的一位同事向我提起这件事后，我加入了Kaggle我的第一个比赛是奥托产品分类挑战赛。由于这个挑战中的特性是模糊的，我无法执行任何探索性的数据分析或特性工程，这与我在这里所做的不同。

What made you decide to enter this competition?  
是什么让你决定参加这次比赛的？

First, I like e-commerce. I’m currently in charge of auction services at Yahoo! JAPAN.  
首先，我喜欢电子商务。我目前在雅虎负责拍卖服务！日本。

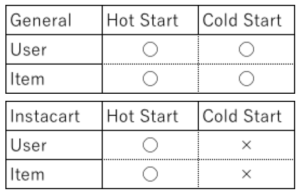
Second, this competition seemed to have clean data, and I thought that there might be a lot of room for feature engineering. I believe my strength is feature engineering, so I thought I’d be able to achieve good results with this kind of data.  
第二，这个竞争似乎有干净的数据，我认为功能工程可能有很大的空间。我相信我的优势是特性工程，所以我想我可以用这种数据取得好的结果。

# Diving Into The Solution 潜入解决方案

## Problem Overview 问题概述

The goal of this competition was to predict grocery reorders: given a user’s purchase history (a set of orders, and the products purchased within each order), which of their previously purchased products will they repurchase in their next order?  
这场竞争的目标是预测杂货店的重新订购：给定用户的购买历史（一组订单，以及每个订单中购买的产品），他们在下一个订单中会重新购买哪些以前购买的产品？

The problem is a little different from the general recommendation problem, where we often face a cold start issue of making predictions for new users and new items that we’ve never seen before. For example, a movie site may need to recommend new movies and make recommendations for new users.  
这个问题有点不同于一般的推荐问题，在这里我们经常面临一个冷启动的问题，即对新用户和新项目进行预测，这是我们以前从未见过的例如，电影网站可能需要推荐新电影，并为新用户提供推荐。



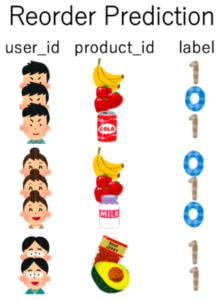
img  
img公司

The sequential and time-based nature of the problem also makes it interesting: how do we take the time since a user last purchased an item into account? Do users have specific purchase patterns, and do they buy different kinds of items at different times of the day? And the competition’s F1 evaluation metric makes sure our models have both high precision and high recall.  
这个问题的顺序性和基于时间的特性也让它变得有趣：我们如何将用户上次购买商品后的时间考虑在内？用户是否有特定的购买模式，是否在一天的不同时间购买不同种类的商品？而竞争对手的F1评价标准则确保了我们的模型既有高精度又有高召回率。

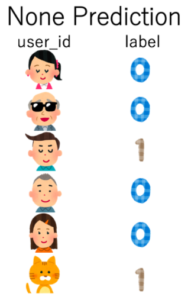
## Main Approach 主要途径

I used XGBoost to create two gradient boosted tree models:  
我使用XGBoost创建了两个梯度增强树模型：

1. Predicting reorders - which previously purchased products will be in the next order? This model depends on both the user and product.  
   预测再订购-哪些以前购买的产品将在下一个订单？这种模式取决于用户和产品。
2. Predicting None - will the user’s next order contain any previously purchased products? This model only depends on the user.  
   预测无-用户的下一个订单是否包含任何以前购买的产品？这个模型只取决于用户。

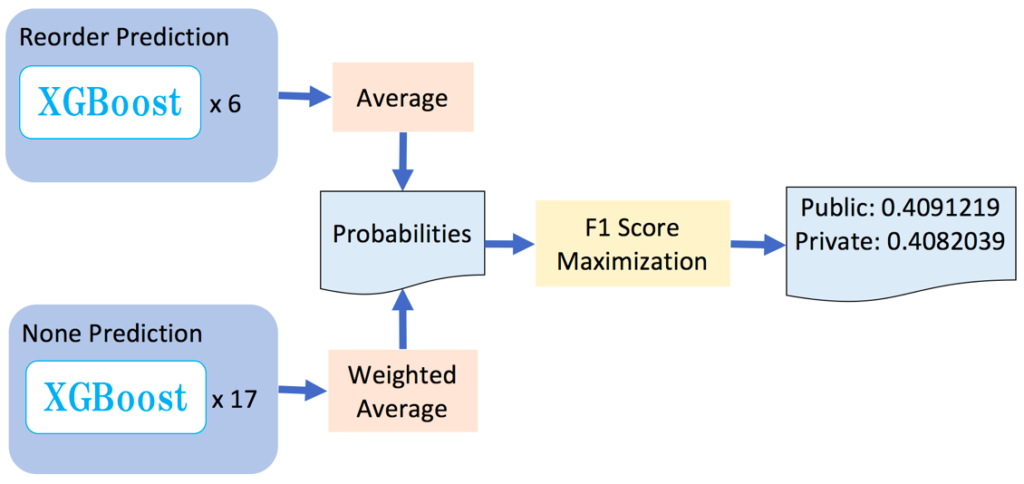


img  
img公司



img  
img公司

Here is a diagram of the model flow.  
这是模型流程图。



img  
img公司

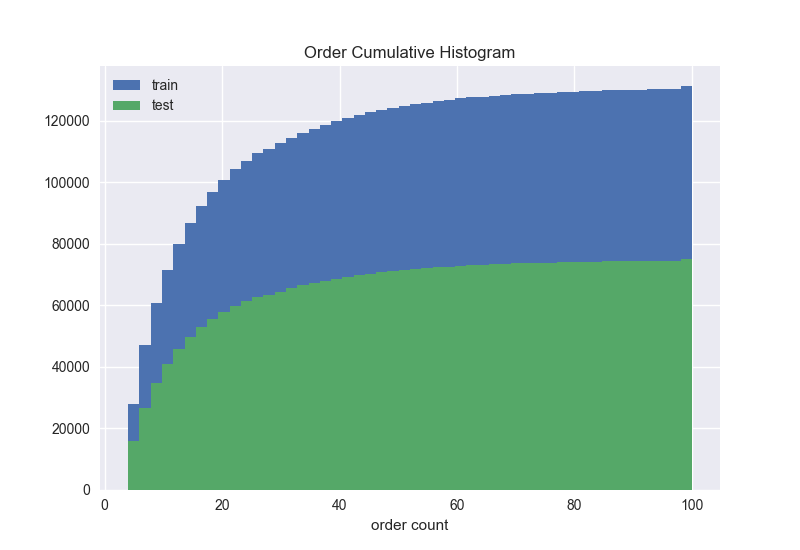
In words:  
换句话说：

* The reorder prediction model uses XGBoost to create six different gradient boosted tree models (each GBDT uses a different random seed). I average their predictions together to get the probability that User A will repurchase Item B in their next order.  
  重新排序预测模型使用XGBoost创建六个不同的梯度增强树模型（每个GBDT使用不同的随机种子）。我把他们的预测平均起来，得到用户A在下一个订单中重新购买项目B的概率。
* The None prediction model uses XGBoost to create seventeen different models. 11 of these use an eta parameter (a step size shrinkage) set to 0.01, and the others use an eta parameter set to 0.002. I take a weighted average of these predictions to get the probability that User A won’t repurchase any items in their next order.  
  无预测模型使用XGBoost创建17个不同的模型。其中11个使用设置为0.01的eta参数（步长收缩），其他使用设置为0.002的eta参数。我对这些预测进行加权平均，得到用户a在下一个订单中不会回购任何商品的概率。
* To convert these probabilities into binary Yes/No scores of which items User A will repurchase in their next order, I feed them into a special F1 Score Maximization algorithm that I created, detailed below.  
  为了将这些概率转换为二进制A是/否分数，用户A将在下一个订单中重新购买这些项目，我将它们输入到我创建的特殊F1分数最大化算法中，详细描述如下。

## Exploratory Data Analysis 探索性数据分析

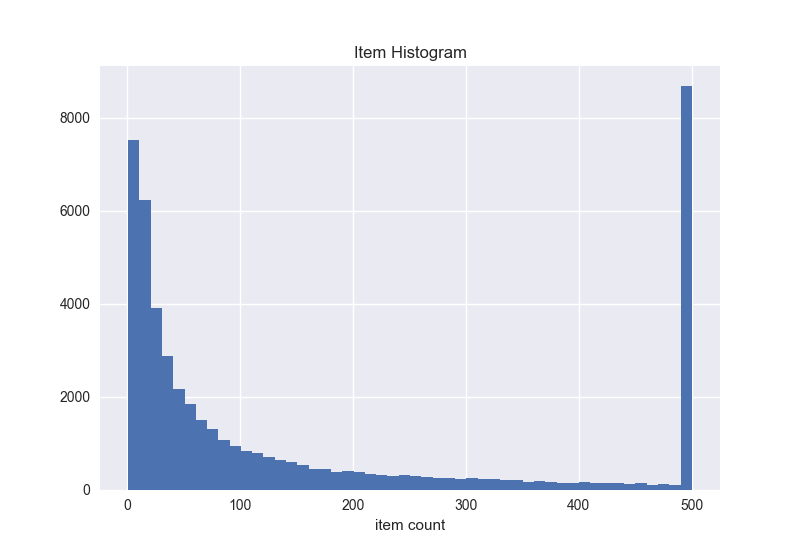
Let’s explore the data a little.  
让我们来研究一下数据。

How hot are users? How many orders do they make?   
用户有多热？他们下了多少订单？

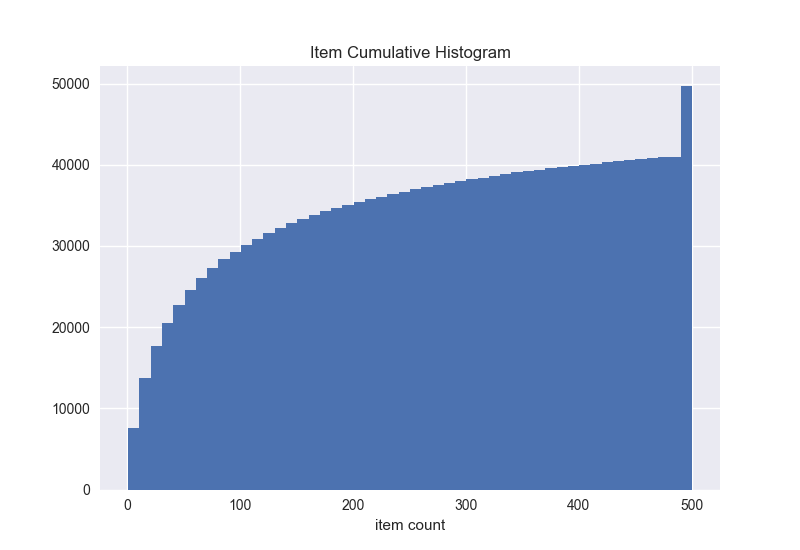


img  
img公司

How hot are items? How often are they ordered?  
物品有多热？他们多久点菜一次？



img  
img公司



img  
img公司

## Data Augmentation 数据扩充

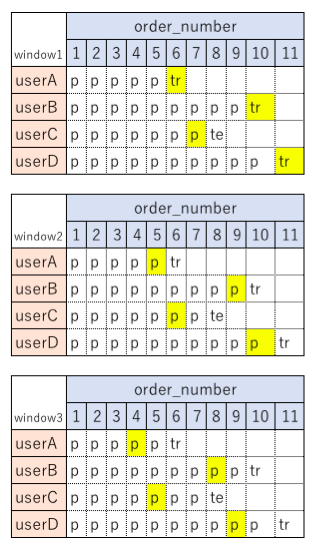
One of my thoughts was that more data would help me make better predictions. Thus, I decided to augment the amount of data I could train on.  
我的一个想法是，更多的数据可以帮助我做出更好的预测。因此，我决定增加我可以训练的数据量。

We were given three datasets:  
我们得到了三个数据集：

* A “prior” dataset containing user purchase histories.  
  包含用户购买历史记录的“先前”数据集。
* Training and test datasets consisting of future orders that we could train and test our models on.  
  培训和测试数据集，包括我们可以培训和测试模型的未来订单。

Rather than training my model only on the provided training set, I increased the amount of training data available to me by adding in each user’s 3 most recent orders as well.  
我没有只在提供的培训集上培训我的模型，而是通过添加每个用户的3个最新订单来增加可用的培训数据量。

This is best illustrated by the figure below.  
下图最好地说明了这一点。



img  
img公司

Instead of only using the provided training set (“tr”), I also looked a short window back in time (the cells shaded in yellow) to gather more data.  
我不仅使用了提供的训练集（“tr”），还查看了一个短时间的窗口（黄色阴影的单元格）来收集更多数据。

## Feature Engineering 特征工程

I created four types of features:  
我创建了四种类型的功能：

1. User features - what is this user like?  
   用户功能-这个用户是什么样的？
2. Item features - what is this item like?  
   项目功能-此项目是什么样的？
3. User x item features - how does this user feel about this item?  
   用户x项目功能-此用户对此项目有何看法？
4. Datetime features - what is this day and hour like?  
   日期时间特性-这一天和时间是什么样的？

Here are some of the ideas behind the features I created.  
以下是我创建的功能背后的一些想法。

User features  
用户功能

* How often the user reordered items  
  用户重新排序项目的频率
* Time between orders  
  订货间隔时间
* Time of day the user visits  
  用户访问的时间
* Whether the user ordered organic, gluten-free, or Asian items in the past  
  用户过去是否订购过有机、无麸质或亚洲商品
* Features based on order sizes  
  基于订单大小的特征
* How many of the user’s orders contained no previously purchased items  
  用户的订单中有多少没有以前购买过的商品

Item features  
项目特征

* How often the item is purchased  
  多久购买一次
* Position in the cart  
  在推车中的位置
* How many users buy it as “one shot” item  
  有多少用户将其作为“一次性”商品购买
* Stats on the number of items that co-occur with this item  
  与此项目同时出现的项目数的统计信息
* Stats on the order streak  
  连续定单统计
* Probability of being reordered within N orders  
  N阶内重新排序的概率
* Distribution of the day of week it is ordered  
  按周分配
* Probability it is reordered after the first order  
  在第一个订单之后重新排序的可能性
* Statistics around the time between orders  
  订单之间时间的统计

User x Item features  
用户x项功能

* Number of orders in which the user purchases the item  
  用户购买项目的订单数
* Days since the user last purchased the item  
  自用户上次购买项目后的天数
* Streak (number of orders in a row the user has purchased the item)  
  Streak（用户已购买商品的一行订单数）
* Position in the cart  
  在推车中的位置
* Whether the user already ordered the item today  
  用户今天是否已经订购了商品
* Co-occurrence statistics  
  共现统计
* Replacement items  
  替代品

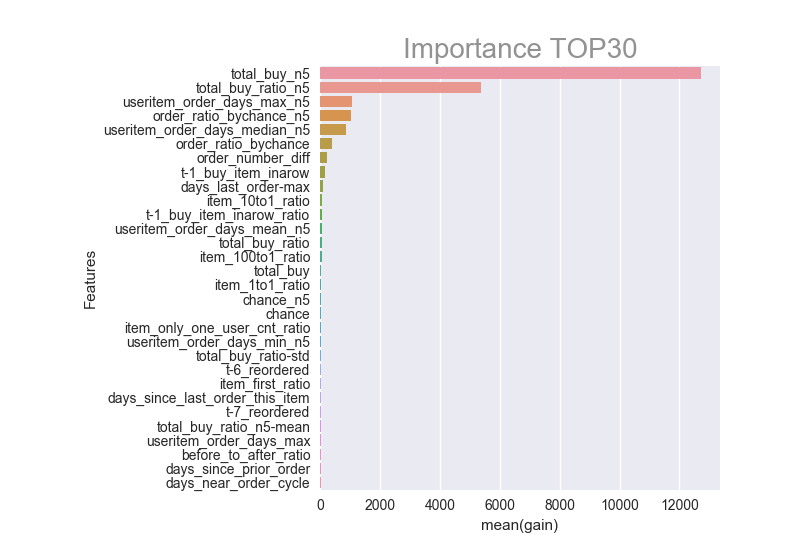
Datetime features  
日期时间功能

* Counts by day of week  
  按星期几计数
* Counts by hour  
  按小时计数

For a full list of all the features I used and how they were generated, see my .  
有关我使用的所有功能及其生成方式的完整列表，请参阅我的。

### Which features were the most useful? 哪些功能最有用？

For the reorder prediction model, we can see that the most important features were…  
对于再订购预测模型，我们可以看到最重要的特征是…



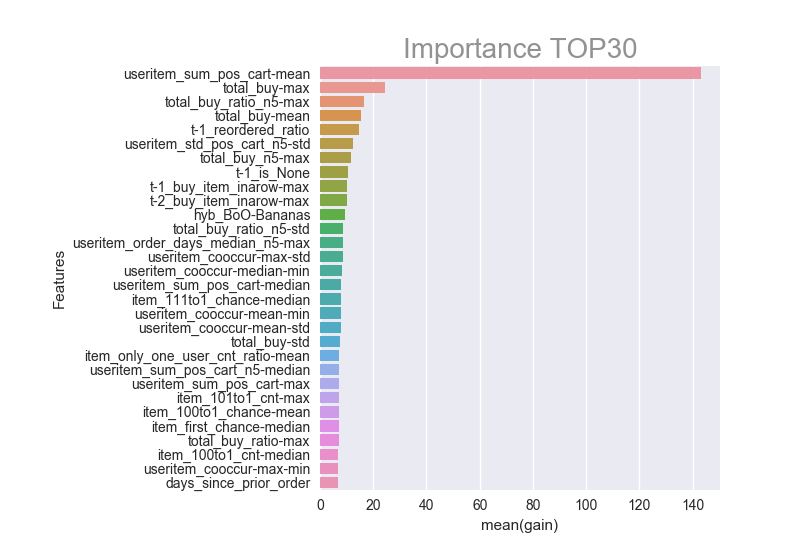
img  
img公司

To explain the top features:  
要解释最重要的功能：

* total\_buy\_n5(User A, Item B) is the total number of times User A bought Item B out of the 5 most recent orders.  
  total\_buy\_n5（用户A，项目B）是用户A在最近5个订单中购买项目B的总次数。
* total\_buy\_ratio\_n5 is the proportion of A’s 5 most recent orders in which A bought B.  
  total\_buy\_ratio\_n5是A最近5个订单中A购买B的比例。
* useritem\_order\_days\_max\_n5, described in more detail below, captures the longest that A has recently gone without buying B.  
  下面详细描述的useritem\_order\_days\_max\_n5捕获了A最近不购买B的最长时间。
* order\_ratio\_by\_chance\_n5 captures the proportion of recent orders in which A had the chance to buy B, and did indeed do so. (A “chance” refers to the number of opportunities the user had for buying the item after first encountering it. For example, if a user A had order numbers 1-5, and bought item B at order number 2, then the user had 4 chances to buy the item, at order numbers 2, 3, 4, and 5.)  
  order\_ratio\_by\_chance\_n5捕获了A有机会购买B的最近订单的比例，并且确实如此。（一个“机会”是指用户在第一次遇到商品后购买该商品的机会数量。例如，如果用户a的订单号为1-5，而B的订单号为2，则用户有4次机会购买该商品，订单号为2、3、4和5。）
* useritem\_order\_days\_median\_n5 is the median number of days that A has recently gone without buying B.  
  useritem\_order\_days\_median\_n5是A最近没有购买B的天数的中位数。

(Note: the suffix "\_n5" means “near5”, i.e., features extracted from the 5 most recent orders.)  
（注：后缀“\_n5”表示“near5”，即从5个最新订单中提取的特征。）

For the None prediction model, the most important features were…  
对于无预测模型，最重要的特征是…



img  
img公司

* useritem\_sum\_pos\_cart-mean(User A) is described in more detail below, and is a kind of measure of whether the user tends to buy a lot of items at once.  
  下面将更详细地描述useritem\_sum\_pos\_cart-mean（用户A），它是一种衡量用户是否倾向于同时购买大量商品的指标。
* total\_buy-max is the maximum number of times the user has bought any item.  
  ToalthBuy MAX是用户购买任何物品的最大次数。
* total\_buy\_ratio\_n5-max is the maximum proportion of the 5 most recent orders in which the user bought a certain item. For example, if there was an item the user bought in 4 out of their 5 most recent orders, but no other item more often than that, this feature would be 0.8.  
  ToopyBuyJuRosioON5-Max是用户购买某一物品的5个最近订单中的最大比例。例如，如果用户在最近的5个订单中购买了4个项目，但没有其他项目比这更频繁，则此功能将为0.8。
* total\_buy-mean is the mean number of times the user has bought any item.  
  total\_buy-mean是用户购买任何商品的平均次数。
* t-1\_reordered\_ratio is the proportion of items in the previous order that were repurchases.  
  t-1úu再订购比率是指前一顺序中被回购的项目所占的比例。

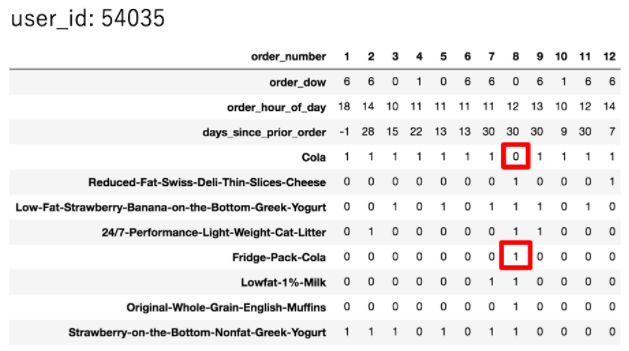
## Insights 洞察力

Here were some of my most important insights into the problem.  
以下是我对这个问题的一些最重要的见解。

### Important Finding for Reorders - #1 重新订购的重要发现-#1

Let’s think about the reordering problem. Common sense tells us that an item purchased many times in the past has a high probability of being reordered. However, there may be a pattern for when the item is not reordered. We can try to figure out this pattern and understand when a user doesn’t repurchase an item.  
让我们考虑一下重新排序问题。常识告诉我们，过去多次购买的商品很有可能被重新排序。但是，当项目未重新排序时，可能会有一个模式。我们可以尝试找出这种模式，并了解用户何时不重新购买项目。

For example, consider the following user.  
例如，考虑以下用户。



img  
img公司

This user pretty much always orders Cola. But at order number 8, the user didn’t. Why not? Probably because the user bought Fridge Pack Cola instead.  
这个用户几乎总是点可乐。但在8号订单上，用户没有。为什么没有？可能是因为用户买了冰箱包装可乐。

So I created features to capture this kind of behavior.  
所以我创建了一些特性来捕捉这种行为。

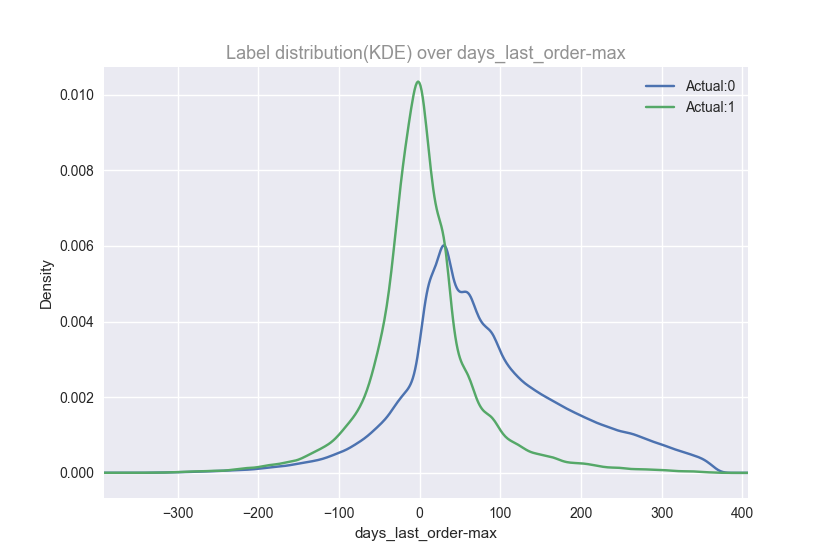
### Important Finding for Reorders - #2 重新订购的重要发现-#2

Days\_since\_last\_order\_this\_item(User A, Item B) is a feature I created that measures the number of days that have passed since User A last ordered Item B.  
自上次订购以来的天数此项目（用户A，项目B）是我创建的一个功能，用于测量自用户A上次订购项目B以来经过的天数。

Useritem\_orders\_days\_max(User A, Item B) is the maximum of the above feature across time, i.e., the longest that User A has ever gone without ordering B.  
UsReTeMyOrthsSoDaysSyxMax（用户A，项目B）是跨越时间的上述特征的最大值，即用户A没有订购B时所经历的最长时间。

Days\_last\_order-max(User A, Item B) is the difference between these two features. So this feature tells us how ready the user is to repurchase the item.  
Days\_last\_order-max（用户A，项目B）是这两个功能之间的区别。所以这个功能告诉我们用户准备如何重新购买商品。

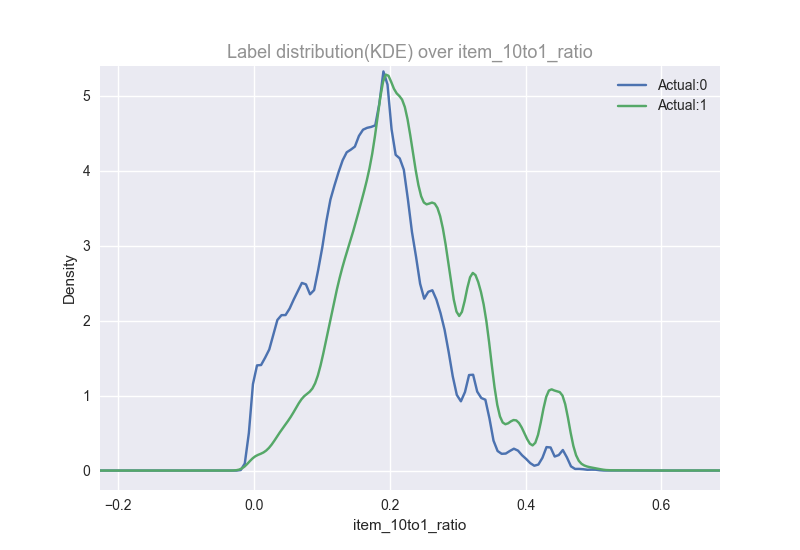
Indeed, if we plot the distribution of the feature, we can see that it’s highly predictive of our target value.  
事实上，如果我们绘制特征的分布图，我们可以看到它对我们的目标值具有很高的预测性。



img  
img公司

### Important Finding for Reorders - #3 重新订购的重要发现-#3

We already know that fruits are reordered more frequently than vegetables (see ). I wanted to know how often, so I made a item\_10to1\_ratio feature that’s defined as the reorder ratio after an item is ordered vs. not ordered.  
我们已经知道水果比蔬菜更频繁地重新排序（见）。我想知道多久一次，所以我制作了一个item-to1-u-ratio特性，它定义为在一个item被订购后的重新订购比率与未订购后的重新订购比率。



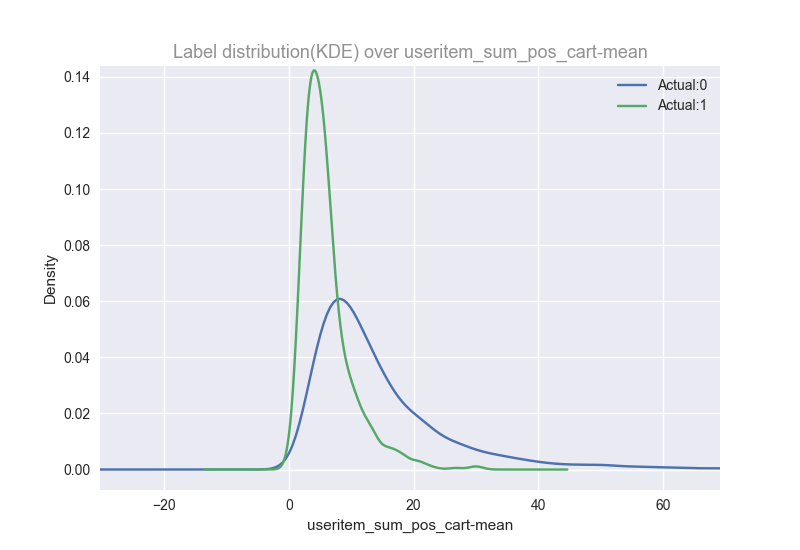
img  
img公司

### Important Finding for None - #1 无重要发现1

Useritem\_sum\_pos\_cart(User A, Item B) is the sum across orders of the position in User A’s cart that Item B falls into.  
User Item\_sum\_pos\_cart（用户A，项目B）是项目B所属的用户A的购物车中位置的各个订单的总和。

Useritem\_sum\_pos\_cart-mean(User A) is the mean of the above feature across all items.  
Useritem\_sum\_pos\_cart-mean（用户A）是上述功能在所有项目中的平均值。

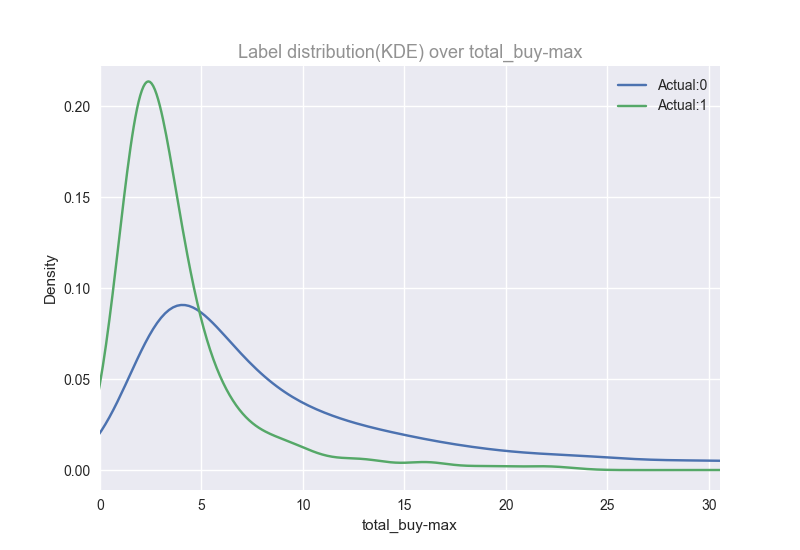
This feature says that users who don’t buy many items all at once are more likely to be None.  
这项功能表示，不同时购买很多商品的用户更有可能一件也不买。



img  
img公司

### Important Finding for None - #2 无重要发现2

Total\_buy-max(User A) is the total number of times User A has purchased any item. We can see that it predicts whether or not a user will make a reorder.  
Total\_buy-max（用户A）是用户A购买任何项目的总次数。我们可以看到它可以预测用户是否会进行重新排序。

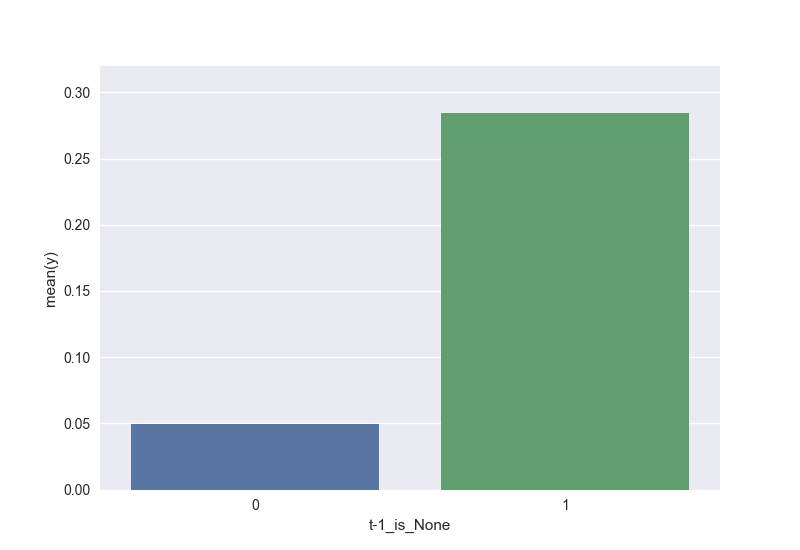


img  
img公司

### Important Finding for None - #3 无重要发现-#3

t-1\_is\_None(User A) is a binary feature that says whether or not the user’s previous order was None (i.e., contained no reordered products).  
t-1\_i s\_None（用户A）是一个二进制特性，表示用户以前的订单是否为None（即，不包含重新排序的产品）。

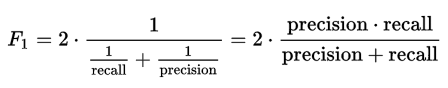
If the previous order is None, then the next order will also be None with 30% probability.  
如果上一个订单为“无”，则下一个订单也将为“无”，概率为30%。



img  
img公司

## F1 Maximization F1最大化

In this competition, the evaluation metric was an , which is a way of capturing both precision and recall in a single metric.  
在本次比赛中，评估指标是一种，它是一种在单一指标中同时获取精确性和召回率的方法。

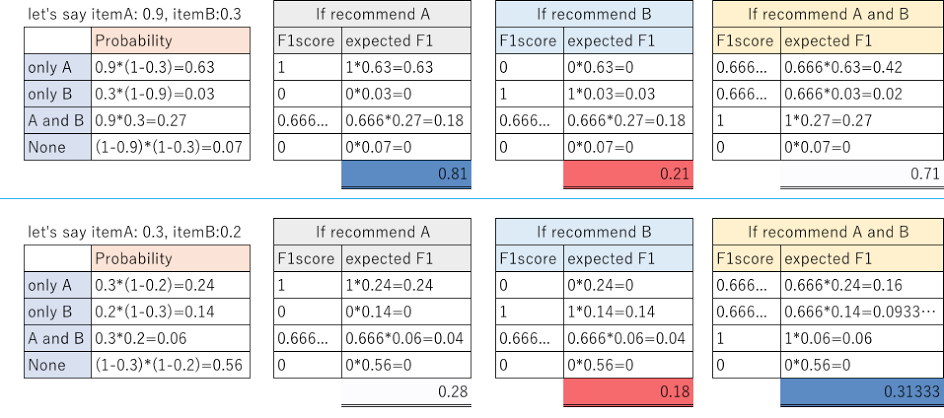


img  
img公司

Thus, instead of returning reorder probabilities, we need to convert them into binary 1/0 (Yes/No) numbers.  
因此，我们需要将它们转换成二进制1/0（是/否）数字，而不是返回重新排序概率。

In order to perform this conversion, we need to know a threshold. At first, I used grid search to find a universal threshold of 0.2. However, then I saw comments on the Kaggle discussion boards suggesting that different orders should have different thresholds.  
为了执行这个转换，我们需要知道一个阈值。起初，我使用网格搜索来找到0.2的通用阈值。然而，后来我在Kaggle讨论板上看到一些评论，建议不同的订单应该有不同的阈值。

To understand why, let’s look at an example.  
为了理解原因，让我们看一个例子。



img  
img公司

Take the order in the first row. Let’s say our model predicts that Item A will be reordered with 0.9 probability, and Item B will be reordered with 0.3 probability. If we predict that only A will be reordered, then our expected F1 score is 0.81; if we predict that only B will be reordered, then our expected F1 score is 0.21; and if we predict that both A and B will be reordered, then our expected F1 score is 0.71.  
按第一排的顺序。假设我们的模型预测A项的重新排序概率为0.9，B项的重新排序概率为0.3如果我们预测只有A将被重新排序，那么我们的预期F1分数是0.81；如果我们预测只有B将被重新排序，那么我们的预期F1分数是0.21；如果我们预测A和B都将被重新排序，那么我们的预期F1分数是0.71。

Thus, we should predict that Item A and only Item A will be reordered. This will happen if we use a threshold between 0.3 and 0.9.  
因此，我们应该预测A项，并且只有A项将被重新排序。如果我们使用0.3到0.9之间的阈值，就会发生这种情况。

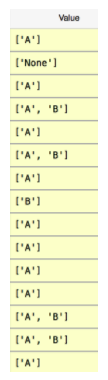
Similarly, for the order in the second row, our optimal choice is to predict that Items A and B will both be reordered. This will happen is long as the threshold is less than 0.2 (the probability that Item B will be reordered).  
同样，对于第二行的订单，我们的最佳选择是预测项目A和B都将重新排序。只要阈值小于0.2（B项重新排序的概率），就会发生这种情况。

What this illustrates is that each order should have its own threshold.  
这说明每个订单都应该有自己的阈值。

### Finding Thresholds 查找阈值

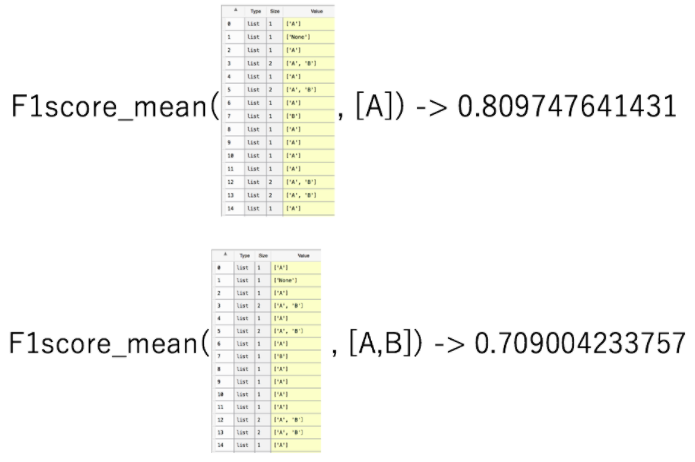
How do we determine this threshold? I wrote a simulation algorithm as follows.  
我们如何确定这个阈值？我写了一个模拟算法如下。

Let’s say our model predicts that Item A will be reordered with probability 0.9, and Item B with probability 0.3. I then simulate 9,999 target labels (whether A and B will be ordered or not) using these probabilities. For example, the simulated labels might look like this.  
假设我们的模型预测A项的重新排序概率为0.9，B项的重新排序概率为0.3。然后，我使用这些概率模拟9999个目标标签（无论A和B是否被订购）。例如，模拟的标签可能是这样的。



img  
img公司

I then calculate the expected F1 score for each set of labels, starting from the highest probability items, and then adding items (e.g., [A], then [A, B], then [A, B, C], etc) until the F1 score peaks and then decreases.  
然后，我计算每组标签的预期F1分数，从概率最高的项目开始，然后添加项目（例如，[A]，然后[A，B]，然后[A，B，C]等），直到F1分数达到峰值，然后降低。



img  
img公司

## Predicting None 没有预测

One way to think about None is as the probability (1 - Item A) \* (1 - Item B) \* …  
一种认为没有的方法是概率（1项A）\*（1项B）\*…

But another method is to try to predict None as a special case. By creating a None model and treating None as just another item, I was able to boost my F1 score from 0.400 to 0.407.  
但另一种方法是尝试将无预测作为特例。通过创建一个None模型并将None视为另一个项目，我可以将F1的得分从0.400提高到0.407。

# Words of wisdom 智慧的言语

What have you taken away from this competition?  
你从这次比赛中得到了什么？

All metrics can be hacked, I think. Especially metrics where we have to convert probabilities to binary scores. (Although metrics like AUC are rarely hacked.)  
我认为所有的指标都可以被破解。尤其是我们必须将概率转换为二进制分数的度量。（尽管像AUC这样的指标很少受到攻击。）

**Do you have any advice for those just getting started in data science?**

Join the competitions you like. But never give up before the end, and try every approach you come up with. I know it’s a tradeoff between sleep and your leaderboard ranking. It’s common for features that take a lot of time to construct to wind up doing nothing. But we can’t know the result if we don’t do anything. So the most important thing is to participate in the delusion that you’ll get a better result if you try!