EatsEncore!

Details About the Food Item Recommendation Engine

1. Overview

First, we analyze food purchases at a restaurant inside a large entertainment venue to identify items that are commonly bought together. Then we use this information to recommend additional items that customers may want to add to their order. We employ a constrained version of Market Basket Analysis, which is an unsupervised machine learning method based on item associations. We implement Market Basket Analysis with the efficient Apriori algorithm.

In short, Market Basket Analysis applies to the following recommendation scenario:
"Customers who have purchased item X, have also purchased item Y."

2. Constrained Version of Market Basket Analysis

Our model is constrained in two important ways: by age and the use of strict categorical rules wherein a user may only have 0 or 1 item per category in each category set, which is different than traditional market basket analysis.

Age Groups	Offerings	Category Sets
Kids = 0–14	• Includes Kids items • Excludes coffee/tea • Excludes adult beverages *(i.e., alcohol)*	kids, side, beverage, dessert
Teens/Young Adults = 15–20	• Excludes Kids items • Excludes adult beverages	entree, side, beverage, dessert
Adults = 21+	• Includes adult beverages • Excludes Kids items	entree, side, beverage, dessert, alcoholic beverage

These constraints result in a need for three separate models—one for each age group. This requires substantial preprocessing for each age group as we will detail in the following sections.

3. Data Engineering

Before feeding the data into our machine learning model or even into our SQLite3 product database used by Flask, extensive data engineering was necessary.

- **Multiple Item IDs:** There were many items with separate item IDs, yet very similar or identical names. They had to be combined by into a single item ID by extensive manual labor.
- **Miscategorized Items:** Multiple items were miscategorized and had to be manually recategorized.
- **Lengthy Item Descriptions:** Many item descriptions were overly complex and verbose and required manual simplification.
- **Multiple IDs in an Order:** Many orders include multiple items of the same item ID. We remove these.

Further, each age group required special choices:

- **Kids**

  - Considered orders that have only kids items, but no entrées. Because many orders involving Kids items also include regular entrées, plus sides, beverages, and desserts. With entrées, this makes it impossible to clearly associate the Kids items with non-entree items.
  - Removed all Kids orders that included alcoholic beverages.

  These choices reduced the number of Kids orders to work with *(each order has multiple items)* to about 2,000.

- **Teens / Young Adults**
  - Kept all orders, but removed alcoholic beverages and kids items.

  This simpler approach gave us about 14,000 orders to work with for those aged 15–20.

- **Adults**
  - Removed all kids items from our orders.

  Again, followed the simpler approach and were able to work with 17,500 orders.

In many orders, there are more than one item per category. In this case, we dissect the order and generate a multitude of orders using all possible unique combinations of categorized items, strictly adhering to 0 or 1 item per category per order.

4. Constrained Market Basket Analysis Execution

In market basket analysis, for a combination of items {X, Y, …}, we are interested in its *support*, *confidence*, and *lift*.

Support is simply the fraction of the combination (for simplicity we consider just pairs here) {X,Y} of all N orders.

$$\text{Support} = \frac{\sum\{X,Y\}}{N}$$

Confidence is a measure of how likely Y is purchased if X is purchased.

$$\text{Confidence} = \frac{\sum\{X,Y\}}{\sum\{X\}}$$

Finally, *Lift* is how likely the pair {X,Y} is, controlling for the popularity of {Y}

$$\text{Lift} = \frac{\text{Confidence}(\{X,Y\})}{\text{Support}(\{Y\})}$$

For the {X,Y} association to be significant, Lift should be > 1.

After building a two-dimensional list of all possible combinations of categorized items for our order data set, we employ the Apriori algorithm as implemented in the apyori Python package.

The hyperparameters that can be tuned are the following:

- Minimum Support
- Minimum Confidence
- Minimum Lift
- Maximum item group length

We did not have sufficient time to go through rigorous hyperparameter tuning. Instead, we employ a heuristic approach. Our dataset of orders is rather limited (~18,000) while our set of products allows a very large number of combinations (for pairs, roughly 1000). Because of this we limit ourselves to item pairs (maximum item group length of two) since larger groups would not have statistically significant data.

Further to this, in this analysis phase, we place rather lax limits on support (typically 0.001) and confidence (typically 0.01)—again, because many combinations are possible and our dataset is limited. Thus, we always require a Lift that is greater than 1.

5. Predictive Model

We iterate through the apyori output and build a dictionary of dictionaries (the keys are hashed and access is fast). The primary keys are the item IDs. Each item ID has a dictionary attached that contains all categories as keys. These keys have values as follows:

- The item's own category has a value of -1, since we never want to recommend items of the same category.
- If apyroi finds significant associations, the value assigned to the category is a list of associations, each being a dictionary containing the item id, plus significance, confidence, and lift for the association.
- If apyori did not find any significant association with an item of the category, the value is set to -2 for that category. If we need to propose an item of that category to accompany this item, we use the popularity model *(see below).*

When deciding on proposed additions to a food order we loop over all open food categories *(i.e., categories without item)* and generate a recommendation for each category.

First, we build a list of all market basket model-based recommendations on the basis of the items in the order. Then, from this list, we pick the recommendation that has the highest value of the product support * confidence * lift.

If there is no market basket model-based recommendation available, we use the popularity model to recommend the most popular item in the category in question.

6. Popularity Model

As a fallback recommendation model, we choose the simple *popularity model*—meaning if we don't have market basket predictions for a given category, the recommendation is the most popular item *(i.e. the most ordered item)* in that category.

7. Limitations and Future Improvements

Due to time constraints, our analysis has been very preliminary. Because of the rules of combinatorics, the size of our dataset, and the size of our product list (menu), we realize that we stand little chance to produce highly significant associations.

In fact, in each of our age-group cases, we found less than significant associations (based on the hyperparameters chosen) for about 10% of possible item pairs. Clearly, there is more to be learned and assessed, in particular when it comes to more complex relationships of three or more items. For this, we would need much larger datasets.

One drawback of our machine learning approach in this project is that we are not creating separate training and test sets. While we had originally planned to do so, we felt that the very limited size of the available dataset made it imperative that we use all the available data to train our model. Since the Apriori algorithm is purely deterministic and straightforward, the only issues could be associated with the customized and constrained setup of our order dataset.

While we have taken great care and done in-situ testing of our code, there is, of course, always room for systematic error. This is something that should be checked in follow-up work. Furthermore, work on hyperparameter tuning and additional data engineering work could improve our results.

Meet the Team

Christian Ott
A computational scientist with a PhD in Theoretical Astrophysics and 15+ years of experience in scientific computing, Christian has a strong understanding of mathematics, numerical methods, and statistics. Additionally he loves cats!

Charla Myers
A content-focused creative director, Charla sees data as a reflection of human behavior. Her goal is to combine her skills for creative development and user experience with analytics to become a data storyteller. Plus she's lived in 6 major American cities.

Eric Jung
A digital product manager who focuses on data analytics & strategy. Eric loves building products that enhance the magic of a customer's experience with data-driven insights. And he's a huge Disney fan.

Website: **https://eatsencore.herokuapp.com/**