What's for Dinner?: Online Grocery Recommendations

Alan Flores-Lopez, Skip Perry, and Poorvi Bhargava
{alanf94, vperry, poorvib}@stanford.edu
Stanford University

Problem Statement

One of the major predictive questions facing retailers is what consumers will purchase and when. Does buying one item make you more likely to buy another? What patterns emerge during the shopping and buying process? Many such questions can be asked. We focused on one of them:

Given that a user has already placed *n* items in their shopping cart, what are the *k* next best items to advertise?

Data

An open-sourced dataset from Instacart with information on nearly 30 million orders. Data includes information about a product's aisle, department, and enumerates millions of user carts in the form:

Cart: (order_id, product_id, order in which product was added to cart)

and the corresponding order information:

Order: (order_id, user_id, day of week, hour of day, days since prior order)

We joined this data fuzzily with data from Open Food Facts to learn the nutrition score of food items. Crafted a data set of the following form from Instacart's raw data:

$$T = \{(\phi(p, u, o), y^{p,u,o}) : p \in P_{rec}, u \in U, o \in u\text{'s cart prefixes}\}$$

Phi extracts features from recommendation, user, partial cart tuples (see below). We train to classify possible recommendations with 0 or 1, in order to generate top 10. A label in T is 1 if a certain product recommendation came next after a partial cart in a user's purchase history, and 0 otherwise. Introduces problems: *uneven class samples*, *massive explicit training sets*. P_rec is the set we recommend from, the top 500 most popular items.

Feature Extraction and Clustering

1. Cart-Based: Features derived based on characteristics of cart itself.
- Number of Products in Cart
- Last Element placed in Cart
- Average of Product Embeddings in Cart
- One-Hot Representation of Cart

2. Product-Based: Features derived using the products in the cart.
- Product Embeddings
- Aisle number of product
- Department number of product
- Whether the product is a food item or not
- Nutriscore
- Which cluster (based on embeddings) did the product belong to?

3. User-Based: No user information was given, we extracted this data.
- Which cluster (based on shopping behavior and products in cart) did the user belong to?
- Nutrition Score of each user
- Time of Day user most frequently shopped
- Day of Week user most frequently shopped
- Which departments did each user frequent? (ex. Alcohol, baby, etc.)

Methods

1) Market Basket Analysis (Smart Baseline, Pairwise Analysis)

$$\text{Lift}(A \to B) = \text{Conf}(A \to B)/f(\text{Supp}(B))$$
$$\text{Confidence}(A \to B) = P(A \cup B) \qquad \text{Support}(A) = P(A)$$

a) Lift used scaling factor toward 1 for Support(B) due to extreme sparsity
b) Performed with and without user clustering
   i) K-means clustering, chose cluster size of 5 based on silhouette score
c) Tuned parameters: Minimum number of orders, scaling factor, clusters

2) Linear and Logistic Predictors

Sci-Kit learn's `SGDClassifier` with hinge and squared loss, as well as Sci-kit's `LogisticRegression` model. Architected system to scan batches of very large (millions) of examples in batches loadable into memory by using `partial_fit` function.

3) Neural Net with ReLU and logistic activations

Sci-kit learn's `MLPClassifier` with default settings. Both ReLU and logistic activations were attempted with better results from ReLU. Same as above, used batch architecture.

4) Cart-Weighted Linear Predictor

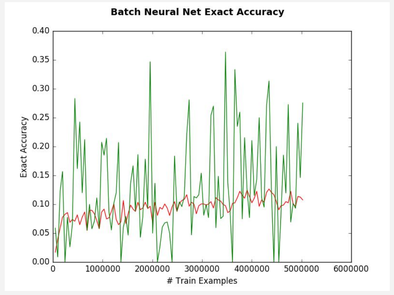
Experiment run with `SGDClassifier` with hinge loss. Minimize a weighted loss that depends on the current (partial) cart we are considering recommending for (dependence on entire recommendation tuple too expensive):

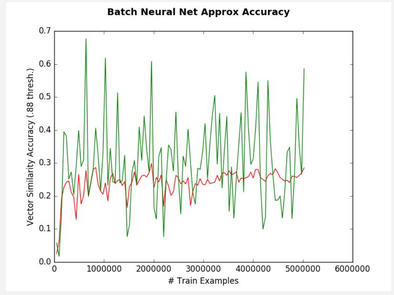
$$\sum_i w^{(i)} \text{Loss-hinge}(x^{(i)}, y^{(i)}) \qquad w^{(i)} = \exp\left(\frac{-(x^{(i)}.cart - x.cart)^2}{2\tau^2}\right)$$

Error Analysis

Qualitative: Analyzing Example Recommendations by User:
- **Partial Cart:** [Banana, Organic Cucumber, Dark Chocolate with Sea Salt & Almonds, Feta Cheese Crumbles, Coconut Fruit Bars, Organic Spring Mix, YoBaby Peach Pear Yogurt]
- **Next Item which User Purchased:** [Organic Strawberries]
- **Model Top 10 Recommendations:** [Strawberries, Lemon Hummus, Guacamole, Hass Avocado, Original Hummus, Hass Avocados, Organic Banana, Organic Strawberries, Broccoli Crown, Baby Arugula]

Quantitative: Analyzing Learning Curves (Green: train, Red: dev)





Results

Model (Training Set Size, Test Set Size)	Exact Accuracy [*Train*]	Exact Accuracy [*Test*]	Approx. Accuracy [*Train*]	Approx. Accuracy [*Test*]
Support (500k, 1k)	0.9%	1.0%	10.1%	10.2%
Confidence w/ User Clustering (500k, 1k)	9.6%	9.3%	21.9%	21.1%
Lift (500k, 1k)	9.9%	9.4%	22.4%	21.6%
Logistic (500k, 1k)	16.8%	10.3%	36.1%	26.0%
Weighted SVM (1000k, 100)	-	1.9%	-	28.2%
SVM (5000k, 1k)	8.0%	6.5%	27.8%	28.4%
Neural Net / ReLU (5000k, 1k)	12.8%	12.2%	30.4%	30.2%

Exact accuracy → the exact next item was in top 10 recs.
Approx. accuracy → next item word2vec similarity with top 10 recs above threshold (.88)

Discussion

- Extremely sparse products: 99%+ of products appear in <1% of orders
- Extremely sparse class labels: 1 to 1000 ratio for 1 labels versus 0 labels
- High standard: nearly 50,000 possible products could come next, exact match counts "banana" as different from "organic banana"
- Complicated consumer shopping patterns, best clusters were not obvious
- Still achieved ~10%+ success rate with recommendation set of 10
- Highest achieving methods learned mostly frequency, despite one-hot cart vectors or word2vec representations
- Weighted methods computationally heavy, need new classifier for example groups

Future Work

- Feature reduction, extraction, and kernels
- Discourage the recommendation of items based on frequency of appearance in training data
- Use better computing resources to train some of the models on more training examples and for longer (e.g. the weighted classifier)
- Find ways to generate train set T to have less class label skew
- Explore graph-based recommendation methods, since in practice, the order in which an item is added to a cart may not be important

References

- Instacart Online Grocery Shopping Dataset 2017. https://www.instacart.com/datasets/grocery-shopping-2017
- Open Food Facts. (2017). https://world.openfoodfacts.org/
- Aggarwal, Charu C. Recommender Systems: The Textbook, 2016. Internet resource.
- Silicon Valley Data Science. (2017). *Learning from Imbalanced Classes - Silicon Valley Data Science*. [online] Available at: https://svds.com/learning-imbalanced-classes/.
- Representation learning for very short texts using weighted word embedding aggregation. https://arxiv.org/pdf/1607.00570.pdf
- Hastie et al, *Elements of Statistical Learning*, Chapter 14 - Market basket analysis (accessed online 2017)