Overview

Students will work in teams to devise data-driven strategies for a grocer that targets improving customer experience, increasing complementary purchases, or both. Students teams will use a dataset previously open-sourced by Instacart as the primary data source for all analyses. (<https://www.kaggle.com/competitions/instacart-market-basket-analysis/data>) Faculty will provide an overview of the dataset in one or more of the seminar class sessions.

As mentioned previously, the goal of the project is for teams to devise data-driven strategies that target improving customer experience, increasing complementary purchases, or both. These objectives are purposefully broad, and student teams may approach the problem from a variety of vantage points.

As an example, student teams may pursue the design of models for a recommendation system that suggests purchases to customer based on past buying history, common complementary items, customer demographic, other novel metrics of product or customer similarity, or a combination of the mentioned features

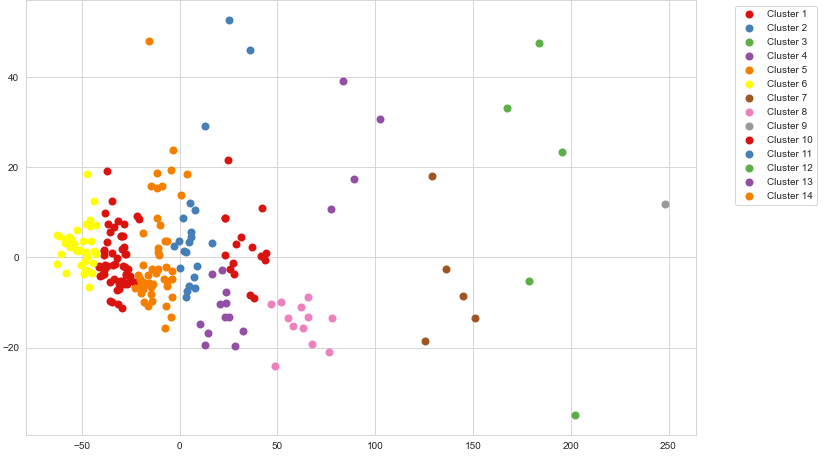
Goal

Develop a method for generating product recommendations for customers based on previous purchase history and shopping cart status.

Underlying Principles of the Recommendation Method

* Essentially we want to take a look at what is in the cart of the user and match that product to what other people put in their cart directly afterwards, but we had the user’s prior orders as well as time data regarding their orders. We can see what they purchased and when they purchased it. Therefore, we could utilize this additional time data to allow for more accurate recommendations. This means we can learn about our users on two dimensions instead of one, products they purchase and the time of their order
  + “We can all assume that people order different things at 11am on a Monday than 8pm on a Friday night.”
* Since we have this data, what is the next step?
* Cluster users based on average time of order (day of the week, time of day, day since prior order, number of products in an order) with a list of the products they ordered

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USER\_ID** | **DOW** | **HOUR OF DAY** | **DAYS SINCE** | **PRODUCTS** |
| 12345 | 5.3 | 12.54 | 10 | Ben & Jerry’s, Wine, Milk, Eggs, etc |

* + WHY?
    - We wanted to group users together by their purchasing tendencies as well as the products they ordered
* How did we include product names (strings) into the clustering process?
  + Enter Doc2Vec…
  + We wanted to get the overall trend of what user’s put in their cart
  + Then we wanted to convert that information to vectors so we would be able to include that sentiment in k means clustering process
* 
* Once we grouped our users into fifteen clusters we now have a way to segment new users (test data) into predefined groups, which should allow for more accurate recommendations
* 
* The next step was to use genism to create a dictionary of bigrams with a frequency score attached. The first word of the bigram would be a key in the dictionary and the second word of the bigram would be the key of a nested dictionary, with the frequency score as the value of the second word in the bigram
  + Here’s where the clustering comes into play…
  + We created two separate bigram dictionaries, one was a dictionary of bigrams for each individual cluster, the other was a dictionary of bigrams for all of the user data
* Using the test data
  + Once we had the bigram dictionaries, we assigned the test data users to clusters using the training data’s clustering algorithm.
* Once the users were assigned to clusters we created functions to match items in their current shopping cart with keys and values in our bigram dictionaries
  + We would look up their current cart items in their prospective cluster section of the clustered bigram dictionary, if no match was found we would try to find a match within the complete user bigram dictionary, if no match was found still, we developed a default recommendation algorithm
    - The default recommendation algorithm selected a random key in the cluster bigram dictionary and assigned the value as the recommendation. There is a level of randomness applied in the default recommendation algorithm, but we tried to counter-act that by only choosing from the products within that specific user’s cluster