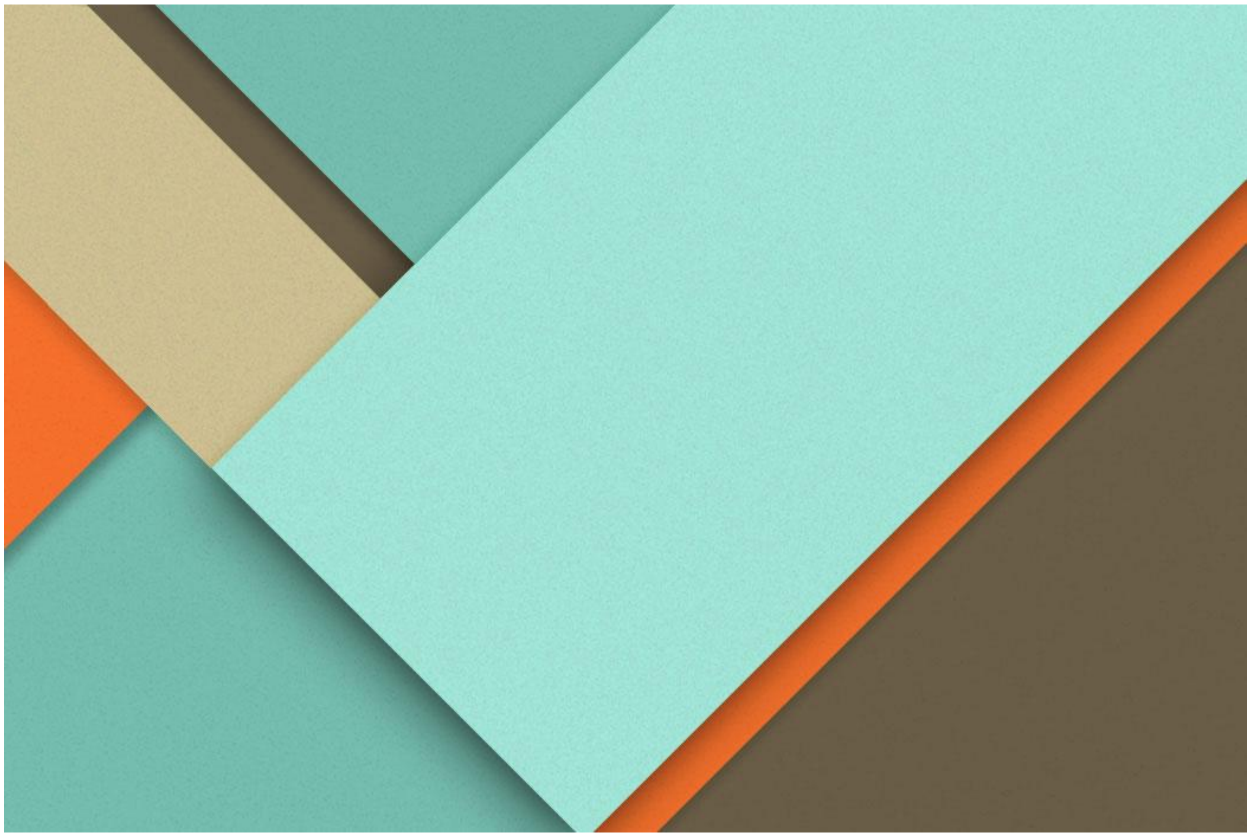

Capstone Project Proposal

17.06.2021

Arunn Thevapalan

Machine Learning Engineer Nanodegree

Domain Background

Once every few days, Starbucks sends out an offer to users of the mobile app. An offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). Some users might not receive any offers during certain weeks. A customer who received the offer would be rewarded when his/her cumulated consumption over the designed threshold. Starbucks sends different types of advertisements and offers once every few days to their customers. A customer might get one of the following:

- Informational offer (i.e., mere advertisement)
- Discount offer
- Buy one get one free (BOGO) offer

Every offer has a validity period before the offer expires. As an example, a BOGO offer might be valid for only 5 days. We'll see in the data set that informational offers have a validity period even though these ads are merely providing information about a product; for example, if an informational offer has 7 days of validity, we can assume the customer is feeling the influence of the offer for 7 days after receiving the advertisement.

Problem Statement

The problem at hand is to combine transaction, demographic, and offer data to determine which demographic groups respond best to which offer type.

Hence this problem can be framed as a recommendation problem, where user groups are clustered based on demographics first, and based on the demographics, recommendations on which offer to provide are given. This is indeed a different type of setting to traditional recommender system use-case but nevertheless works very well in our case.

The work-in-progress GitHub repository can be found here along with a README.

Datasets & Inputs

The dataset simulated by Starbucks was given by Udacity and has detailed information. The data can also be found from the project's repository here. The data is contained in three files and their schema are as below,

portfolio.json — containing offer ids and metadata about each offer (duration, type, etc.).

- id (string) - offer id
- offer_type (string) - a type of offer ie BOGO, discount, informational
- difficulty (int) - minimum required spend to complete an offer
- reward (int) - reward given for completing an offer
- duration (int) - time for offer to be open, in days
- channels (list of strings)

profile.json — demographic data for each customer.

- age (int) - age of the customer
- became_member_on (int) - date when customer created an app account
- gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
- id (str) - customer id
- income (float) - customer's income

transcript.json — records for transactions, offers received, offers viewed, and offers complete.

- event (str) - record description (ie transaction, offer received, offer viewed, etc.)
- person (str) - customer id
- time (int) - time in hours since the start of the test. The data begins at time t=0
- value - (dict of strings) - either an offer id or transaction amount depending on the record

Upon further investigation, we notice that the simulated data was obtained for a period of 29.75 days, with 306,534 events.

Solution Statement

A potential solution presets modification to simple knowledge-based recommendation systems. The reason is, recommender systems are often complex and require high computational power. The plan is to create a basic knowledge-based system. Then identify the trends and features through an in-depth exploratory data analysis. Based on the analysis, use few filter-based recommendations.

Further development would be to use collaborative filtering algorithms to fine-tune the results.

We need to come up with metrics that can drive profit to Starbucks and in turn, will result in high customer engagement. These metrics will be described in a later section.

All code and datasets will be provided on GitHub making this project fully replicable.

Benchmarks

Let us first build a basic recommender system that recommends the most popular offers to the customers. This is the basis of Knowledge-based recommendations. Here we sort offers based on the ones that result in the highest net expense for Starbucks. This function achieves this returning the set of best offers.

```
def get_most_popular_offers(customers, n_top=2, q=0.5, offers=None):

  if not offers:
      offers = ['I1', 'I2', 'B1', 'B2', 'B3',
                'B4', 'D1', 'D2', 'D3', 'D4']
  offers.sort(key=lambda x: get_net_expense(customers, x, q), reverse=True)
  offers_dict = {o: get_net_expense(customers, o, q) for o in offers}
  return offers[:n_top], offers_dict
```

Evaluation Metrics

Since we have defined our problem, it is vital to decide on a metric and justify the same. What would be a good metric to evaluate the performance of our recommender system? The chosen metrics are:

- Net Expense (total expense - reward received)
- Average Transaction Values (total expense/total transactions)

Maximizing these two metrics would result in a profit to Starbucks and will result in high customer engagement.

Project Design

The strategy employed to solve this problem is as below,

1. Business Understanding of Starbucks
2. Understanding the simulated data by Starbucks
3. Pre-process the data, extract meaningful information combining all datasets
4. Exploratory Data Analysis
5. Build Recommendation models
6. Evaluate the results and iterate
7. Conclude with future improvements
8. Communicate the results

The proposed plan revolves around the blueprint given above. This was derived from a typical machine learning workflow. The idea is to elaborate on each of the sections above, giving more context as we proceed into the project.