Domain Background

The capstone project is an experiment that determines how do we take this experimental data and discover what are the offers that excite people? So, the Capstone Project is about discovering what is the most valuable offer there is, not just for the customers as a whole but at an individual personal level.

Problem Statement

To predict if someone will reply to an offer, transaction data and demographic information must be combined. GitHub customers will also have access to the data and in the Data Sets section of the proposal, it is clearly stated that the repository and data sets are available.

Note that every offer will have an expiration date. For example, a buy one gets one free offer might be valid for only a certain number of days. You'll see in the data set that informational offers have a validity period even though these advertisements are providing information about a product; for example, if an informational offer has 10 days of validity, you can assume that after receiving the advertisement the customer will be aware of the offer for 10 days.

Solution Statement

Using Gradient Boosting, a supervised learning method, we will analyze the attributes of customers to create customer classifications.

Datasets and Inputs

Starbucks rewards mobile app customers are simulated with this simulated data. An offer could be an advertisement for a drink, a discount, or a buy one get one free deal. Some customers may

not be sent offers during certain days or weeks and not all customers will receive the same offer. That is the challenge to solve using this dataset, who gets what offer?

Also, this transactional data contains a record for each offer that a customer receives as well as a record for when a customer sees the offer. There are also records for when a customer completes the offer that is viewed.

The data is in three files:

* portfolio.json - has offer ids and metadata about each offer (duration, type, etc.)

* profile.json - the demographic data for each customer

* transcript.json - the records for transactions, offers received, offers viewed, and offers complete

Each variable in the files is described in the following schema:

* portfolio.json

* id (string) - the offer id

* offer_type (string) - a type of offer i.e. by one get one free, discount, informational

* difficulty (int) - the minimum required to spend to complete an offer

* reward (int) - the reward is given for completing an offer

* duration (int) - the time for the offer to be open, in days

* channels (list of strings)

* profile.json

* age (int) - the age of the customer

* became_member_on (int) - the date when the customer created an app account

* gender (string) - the gender of the customer (some entries contain O rather than M or F)

* id (string) - the customer-id

* income (float) - income of customers

* transcript.json

* event (string) - the record description (e.g. transaction, offer received, offer viewed, etc.)

* person (string) - the customer-id

* time (int) - the time in hours since the start of the test (begins at time t=0)

* value - (dict of strings) - offer id or transaction amount

To use the dataset:

```
import pandas as pd

import numpy as np

import math

import json



portfolio = pd.read_json('data/portfolio.json', orient='records', lines=True)

profile = pd.read_json('data/profile.json', orient='records', lines=True)

transcript = pd.read_json('data/transcript.json', orient='records', lines=True)
```

Benchmark Model

We will be using multiclass logistic regression against which we can benchmark.

Evaluation Metrics

This is a multi-class classification problem so the key metric we will use is accuracy. The simulating dataset only has one product, while Starbucks offers dozens of products. Therefore, this data

set is a simplified version of the real Starbucks app.

Project Design

Perform data cleaning or data preprocessing on the JSON files above then upload the data to AWS S3. Declare our model training hyperparameter. Create our training estimator then fit our estimator. Finally, for standout suggestions, we will perform hyperparameter tuning to increase the performance of our model. We can also deploy our model to an endpoint and then query that endpoint to get a result.