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.

Link to an academic paper where machine learning was applied to this type of problem: http://ceur-ws.org/Vol-3026/paper18.pdf

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.)

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7
5	3	[web, email, mobile, social]	7	7	discount	2298d6c36e964ae4a3e7e9706d1fb8c2
6	2	[web, email, mobile, social]	10	10	discount	fafdcd668e3743c1bb461111dcafc2a4
7	0	[email, mobile, social]	0	3	informational	5a8bc65990b245e5a138643cd4eb9837
8	5	[web, email, mobile, social]	5	5	bogo	f19421c1d4aa40978ebb69ca19b0e20d
9	2	[web, email, mobile]	10	7	discount	2906b810c7d4411798c6938adc9daaa5

profile.json

The demographic data for each customer

	gender	age	id	became_member_on
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509
4	None	118	a03223e636434f42ac4c3df47e8bac43	20170804
...	...	...	...	...
16995	F	45	6d5f3a774f3d4714ab0c092238f3a1d7	20180604
16996	M	61	2cb4f97358b841b9a9773a7aa05a9d77	20180713
16997	M	49	01d26f638c274aa0b965d24cefe3183f	20170126
16998	F	83	9dc1421481194dcd9400aec7c9ae6366	20160307
16999	F	62	e4052622e5ba45a8b96b59aba68cf068	20170722

```
        income
0          NaN
1     112000.0
2          NaN
3     100000.0
4          NaN
...        ...
16995  54000.0
16996  72000.0
16997  73000.0
16998  50000.0
16999  82000.0

[17000 rows x 5 columns]
```

transcript.json

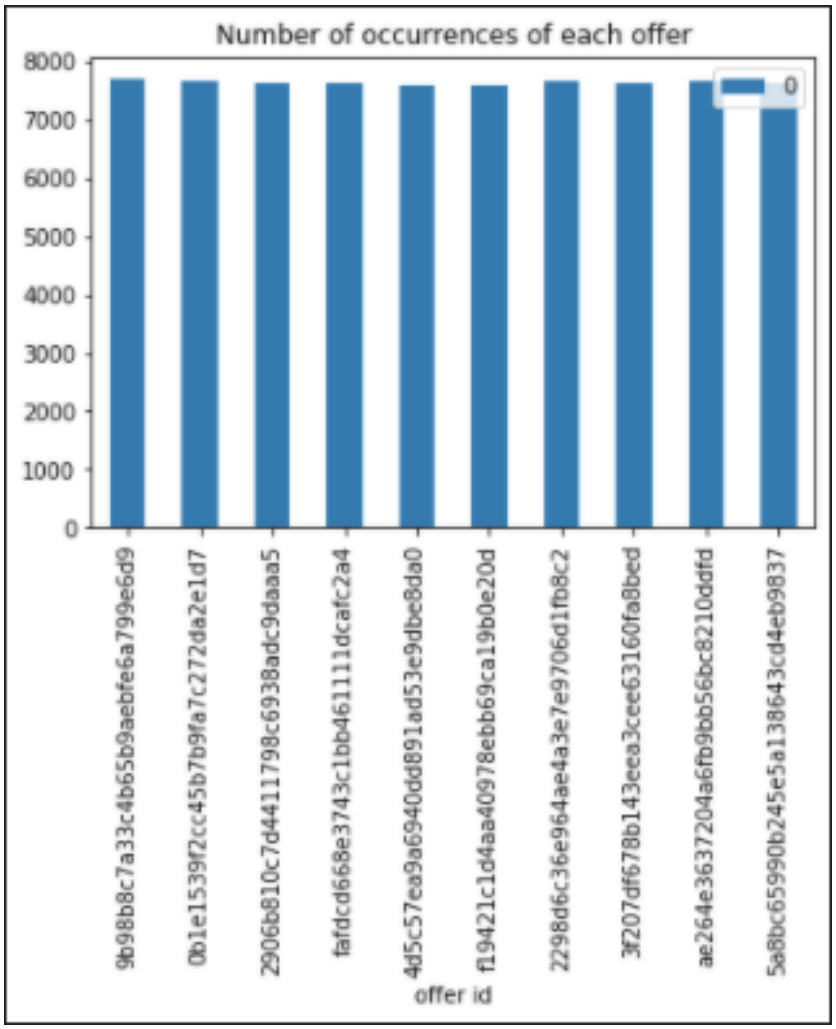
The records for transactions, offers received, offers viewed, and offers complete

```
                             person           event  \
0       78afa995795e4d85b5d9ceeca43f5fef  offer received
1       a03223e636434f42ac4c3df47e8bac43  offer received
2       e2127556f4f64592b11af22de27a7932  offer received
3       8ec6ce2a7e7949b1bf142def7d0e0586  offer received
4       68617ca6246f4fbc85e91a2a49552598  offer received
...                                  ...              ...
306529  b3a1272bc9904337b331bf348c3e8c17      transaction
306530  68213b08d99a4ae1b0dcb72aebd9aa35      transaction
306531  a00058cf10334a308c68e7631c529907      transaction
306532  76ddbd6576844afe811f1a3c0fbb5bec      transaction
306533  c02b10e8752c4d8e9b73f918558531f7      transaction

                                       value  time
0       {'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}     0
1       {'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}     0
2       {'offer id': '2906b810c7d4411798c6938adc9daaa5'}     0
3       {'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}     0
4       {'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}     0
...                                                    ...   ...
306529                {'amount': 1.5899999999999999}   714
306530                        {'amount': 9.53}   714
306531                        {'amount': 3.61}   714
306532                {'amount': 3.5300000000000002}   714
306533                        {'amount': 4.05}   714

[306534 rows x 4 columns]
```

Observe that the data is balanced if we consider the customers who received offers



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

portfolio.json
* id (string) - offer id
* offer_type (string) - a type of offer ie BOGO, 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) - time for the offer to be open, in days
* channels (list of strings)

profile.json
* 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
* 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

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 precision. 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. Note that in the transcripts file we only consider customers who received

offers. We will make category types for the data so models know they are not just numbers. Further, we will create a histogram of all features to show the distribution of each one relative to the data. This is part of the exploratory data analysis. Also Then upload the data to AWS S3. Declare our model training hyperparameter. Create our training estimator then fit our estimator. The XGBoost and LightGBM models could be good supervised learning approaches to try here. 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.