Starbucks Capstone Project Proposal

Anirudh Sharma | April 06, 2020
Udacity – Machine Learning Engineer Nanodegree

Domain Background

Starbucks is a passionate purveyor of coffee and other beverages, headquartered in Seattle, Washington. The corporation is ranked 121st in the list of 2019 Fortune 500 companies. They have a mobile application where registered users can use it to order coffee for pickup while mobile, pay in-store directly using the app, and collect rewards points. This app also offers promotions for bonus points to these users. The promotional offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). This project is focused on tailoring the promotional offers for customers based on their responses to the previous offers and find out which of them are most likely to respond to an offer.

Problem Statement

The goal that I have to achieve here is to best determine which kind of offer to send to each user based on their response to the previously sent offers. Not all users receive the same offer, and that is the challenge to solve using the data set that is provided by Starbucks, which was captured over 30 days. I'll also build a machine learning model that will predict the response of a customer to an offer.

Datasets and Inputs

This data set contains simulated data that mimics customer behavior on the Starbucks rewards mobile app. Once every few days, Starbucks sends out an offer to users of the mobile app. The data set is provided in form of three JSON files:

- portfolio.json - containing offer ids and meta data about each offer (duration, type, etc.)
- profile.json - demographic data for each customer
- transcript.json - records for transactions, offers received, offers viewed, and offers completed

Here is the schema and explanation of each variable in the files:

portfolio.json

- id (string) - offer id
- offer_type (string) - 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)

```
In [8]: portfolio.head(10)
```

Out[8]:

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

```
In [9]: print("portfolio: Rows = {0}, Columns = {1}".format(str(portfolio.shape[0]), str(portfolio.shape[1])))
portfolio: Rows = 10, Columns = 6
```

portfolio and its number of rows & columns

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

```
In [23]: profile.head()
```

Out[23]:

	age	became_member_on	gender	id	income
0	118	20170212	None	68be06ca386d4c31939f3a4f0e3dd783	NaN
1	55	20170715	F	0610b486422d4921ae7d2bf64640c50b	112000.0
2	118	20180712	None	38fe809add3b4fcf9315a9694bb96ff5	NaN
3	75	20170509	F	78afa995795e4d85b5d9ceeca43f5fef	100000.0
4	118	20170804	None	a03223e636434f42ac4c3df47e8bac43	NaN

```
In [24]: print("profile: Rows = {0}, Columns = {1}".format(str(profile.shape[0]), str(profile.shape[1])))
profile: Rows = 17000, Columns = 5
```

Head of profile and its number of rows & columns

transcript.json

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

```
In [25]: transcript.head()
```

Out[25]:

	event	person	time	value
0	offer received	78afa995795e4d85b5d9ceeca43f5fef	0	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}
1	offer received	a03223e636434f42ac4c3df47e8bac43	0	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}
2	offer received	e2127556f4f64592b11af22de27a7932	0	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}
3	offer received	8ec6ce2a7e7949b1bf142def7d0e0586	0	{'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}
4	offer received	68617ca6246f4fbc85e91a2a49552598	0	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}

```
In [26]: print("transcript: Rows = {0}, Columns = {1}".format(str(transcript.shape[0]), str(transcript.shape[1])))
transcript: Rows = 306534, Columns = 4
```

Head of transcript and its number of rows & columns

The portfolio.json contains offer_type column, which describes the types of offers that Starbucks is looking to potentially send its customers:

1) BOGO (Buy-One-Get-One): This offer enables a customer to receive an extra and equal product at no additional cost. The customer must spend a certain threshold in order to make this reward available.
2) Informational: This offer doesn't necessarily include a reward, but rather an opportunity for a customer to purchase a certain object given a requisite amount of money.
3) Discount: With this offer, a customer is given a reward that knocks a certain percentage off the original cost of the product they're choosing to purchase, subject to limitations.

▪ Solution Statement

To find out which offers are to be sent to the customers, I'll find out the offers that interests them the most, and consider Exploratory Data Analysis to cover a few points like:

1) most responded offer
2) response to an offer
3) age & gender groups which are greatly interested in offers

These points will be discussed for the combined population, and for the individual personalized level as well.

To find out the appropriate response of a customer to an offer, I'll be leveraging models like RandomForestClassifier and DecisionTreeClassifier, to determine which model best represents our data on hand.

▪ Benchmark Model

A quick and fairly accurate model can be considered as a benchmark. I will use the KNeighborsClassifier to build the benchmark, as it is a fast and standard method for binary classification machine learning problems and evaluate the model result using F1 score as the evaluation metric.

- ## Evaluation Metrics

  I will consider the F1 score as the model metric to assess the quality of the approach and determine which model gives the best results. It can be interpreted as the weighted average of the precision and recall. The traditional or balanced F-score (F1 score) is the harmonic mean of precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

- ## Project Design

  Here is the general flow for how I will be conducting this project:

  1) Establishing the workspace in a Jupyter environment
  2) Cleaning up the data as needed for the modeling purposes.
  3) Performing a deep-dive exploratory analysis on the data
  4) Building different models to determine the most appropriate one for the data
  5) Leveraging benchmark model and evaluation metric to ensure sanity.
  6) Summarise the findings and project work in a detailed blog post.