# Starbucks Rewards Recommender

## Introduction

As a coffee lover, I always check out the Starbucks mobile app for offers and discounts for my next coffee purchase. I was greatly motivated to see datasets containing transaction, demographic information and offers that mimics customer behavior on Starbucks rewards mobile app. I decided to take this as a challenge for my Machine Learning Nanodegree program, explore the datasets, build a supervised model and deploy the same to predict influencing offers for Starbucks coffee lovers.

### Domain background

A recommendation system falls under the broad umbrella of information filtering system. The primary goal or application of a such systems is to suggest an attractive list of product recommendations for commercial purposes. A company can use these suggested recommendations and advertise its products to targeted audience. What makes this problem challenging is the plethora of industries that require such recommendation services. Each domain has its own set of consumers and product types to experiment with. On a positive note, there is one common question that needs an answer across the board for any industry i.e. How to attract new customers and keep existing customers to buy products and services? In this project, I would like to explore this question from the perspective of a famous Coffee company, “Starbucks”.

### Problem statement

We will answer the query of how to make Starbucks customers happy and keep them interested in purchasing its products with the help of the rewards app that the coffee franchise offers to its customers. The goal is to make the rewards app provide different offers that interests its customers and keeps them buying more coffee and related products from Starbucks. One thing to remember is that the app can not only recommend offers to existing customers but also draw new customers based on the machine learning model which can recommends offers to new customers who have similar demographic attributes as existing customers.

The specific question that I wanted to answer in this project is,

What specific offers should we recommend to customers with certain demographic attributes. Meaning, a very specific query we might want to answer could be,

What offers will attract customers < 40 years of age and yearly income > 70K?

I will explore at least two or more queries like this in this project.

I also expect the model to predict one or more offers we want to recommend our customers with a given set of attributes. Also, before training the model using the provided datasets, I would want to know if a previously recommended offer actually influenced the customers on their next purchase. We will see later how an indicator attribute, something like influencing_offer can be useful for our analysis.

## Datasets and Inputs

The datasets for this project are contained in three files. These files were provided as part of the Starbucks project workspace on Udacity's website. I have referenced a short video in the reference section showing the motivation of this capstone project as shared by a Data Scientist at Starbucks.

The data names, shapes, schema and explanation of each variable in these files are below:

1. **portfolio.json** - containing offer ids and meta data about each offer (duration, type, etc.)

***Columns***:

id (string) - offer id
offer_type (string) - type of offer i.e. 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

**Numeric** – difficulty, reward, duration

**Text** – id, offer_type, channels

***Rows***: 10

As mentioned, the portfolio data has details of offer metadata. The tabular representation of this data is shown next. This is just metadata so we need not worry about balanced and unbalanced nature of this table.

| | 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 |

2. **profile.json** - demographic data for each customer

***Columns***:

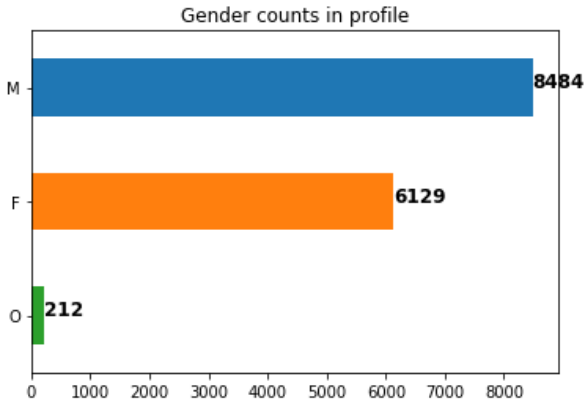
age (int) - age of the customer
became_member_on (int) - date when customer created an app account
gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
id (str) - customer id
income (float) - customer's income

**Numeric** – *age, id and income*

**Text** – *gender*

***Rows***: 17,000

The dataset is little imbalanced as there are more males and females than other gender types. We could either sample this data to work on subset cases targeting specific gender types and also augment the dataset with some more cases for the other gender category so as to balance for few cases with the other gender category.




The first few rows of the profile dataset are shown below,

| | age | became_member_on | gender | id | income | has_transacted |
|---|---|---|---|---|---|---|
| **1** | 55 | 20170715 | F | 0610b486422d4921ae7d2bf64640c50b | 112000.0 | True |
| **3** | 75 | 20170509 | F | 78afa995795e4d85b5d9ceeca43f5fef | 100000.0 | True |
| **5** | 68 | 20180426 | M | e2127556f4f64592b11af22de27a7932 | 70000.0 | True |
| **8** | 65 | 20180209 | M | 389bc3fa690240e798340f5a15918d5c | 53000.0 | True |
| **12** | 58 | 20171111 | M | 2eeac8d8feae4a8cad5a6af0499a211d | 51000.0 | True |

3. **transcript.json** - records for transactions, offers received, offers viewed, and offers completed

*Columns*:

event (string) - record description (i.e. transaction, offer received, offer viewed, etc.)

person (string) - 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

**Numeric** – *time (this is time in hours since test started)*

**Text** – event, person, value

*Rows*: 306,534

The transactions data is balanced as such as it does not involve any gender columns like in the profile data above. But we surely want to normalize the transactions if we want to query and find any influencing offers based on gender with particular age and annual income range.

The first few rows of the transcript dataset are shown below,

| | event | person | time | value |
|---|---|---|---|---|
| **12654** | transaction | 02c083884c7d45b39cc68e1314fec56c | 0 | {'amount': 0.8300000000000001} |
| **12657** | transaction | 9fa9ae8f57894cc9a3b8a9bbe0fc1b2f | 0 | {'amount': 34.56} |
| **12659** | transaction | 54890f68699049c2a04d415abc25e717 | 0 | {'amount': 13.23} |
| **12670** | transaction | b2f1cd155b864803ad8334cdf13c4bd2 | 0 | {'amount': 19.51} |
| **12671** | transaction | fe97aa22dd3e48c8b143116a8403dd52 | 0 | {'amount': 18.97} |

## Solution statement

To solve our problem of what offers should we must recommend our customers we will follow the steps below,

1. Find a list of transacted users form our transactions dataset. This will be part of our preprocessing step to get the records that have transacted and use it for training purposes. We can then use the other records that have not transacted at all as test cases.

2. Next, we find, influencing offers based on the transacted records by looking at the completed offers and also making sure the customer viewed the offer before they completed the offer based on the key offer_id in the transcript data frame.

   *Note:*

   a. *Someone using the app might make a purchase through the app without having received an offer or seen an offer. Influencing offers are found by looking at the completed offers and also making sure the customer viewed the offer before they completed it. The "event" and "time" columns in the transcript dataset can be used to find such influential offers for each customer.*

   b. *Another tricky part here could be that in the "value" column of transcript data frame, has dictionary keys were "offer id" i.e. with a space in the middle, for offer received and offer viewed entries, but is "offer_id" i.e. with an underscore in the middle, for offer completed entries. So, we have to be mindful of these key names when finding offers from transactions.*

3. We could now expand the influencing offer_id into one hot encoded fields of all offers. These columns will have a 1 if an offer influenced a transaction or 0 if not.

4. Using the above transformed data with influencing offers for each transaction transactions we can now train and build a machine learning model. I plan to build a **sklearn Random Forest Classifier** model. The input dataset for our model will be a transformed dataset derived from the transaction and portfolio datasets including new columns that one-hot encode influencing_offers we created in step 2.

   a. Our model input variables could be age, gender and income and the response will be multi-class representation of one hot encoded offer column that influences such transaction.

   b. We adjust the Random Forest model hyperparameters like split criterion, max_depth and max_features using a tuning function that runs training over several iterations using cross-validated data samples until certain threshold is reached. We either specify the number of iterations to run this tuning function or use a specific threshold on accuracy or recall of the model where the iteration stops. The assumption here is that we have reached a point where our model cannot be tuned further.

   c. Finally, we will measure and evaluate model quality metrics such as AUC, Precision and Recall (from confusion matrix) and F1 scores of our Random Forest model and compare it with the benchmark model explained next.

We expect our recommendation model to predict offers for customers based on the given demographic attributes. Our model should work in such a way that the

predicted offers have a greater chance of actually influencing a customer with their next purchase.

## Benchmark model

As a benchmark model we will also build sklearn multinomial logistic Regression classifier with the same training dataset we used for our Random Forest classifier. As part of measuring the model quality we will also generate the AUC, Confusion Matrix and F1 scores for this model and compare these values with our Random Forest model to conclude which model performed the best.

We will call the model with better accuracy i.e. one with better precision and recall as our Champion Challenger and use it for model deployment.

## Evaluation metrics

We will use the following evaluation metrics for our model. Once we know our model performs reasonably, we will deploy it on new customer data and predict offers.

1. Confusion Matrix — The confusion matrix provides a great place to start model evaluation for classification models. In essence we look at the proportions of predicted and observed classes and build out a matrix showing True Positives, True Negatives, False Positives and False Negatives. From this we can then compute our model's precision and recall. A precise model with a high recall rate is always better to have. Understanding the confusion matrix is a much bigger topic. All I will say here is, we can trust our classification model to recommend an offer with high confidence if it had a high precision and a better recall rate.

2. Area Under the Curve (AOC) — This metric will also fit in well as a performance metric for our classifier. In short, the AUC-ROC curve is a performance measure that tells us how better our model is in distinguishing the different categories or classes that we are exploring using the model. The higher the AUC the better is the model.

3. F1 score — This 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

The solution will be designed on Jupyter notebooks that reads the previously mentioned datasets and transforms it to create a normalized data table of customer transaction information. Also, I will one hot encode the response variables i.e. offer types for each transaction and include an indicator variable to track if an offer had influence on a transaction. We will create this indicator variable using the following criteria,

*Influencing offers are found by looking at the completed offers and also making sure the customer viewed the offer before they completed it. The "event" and "time" columns in the transcript dataset will be used to find such influential offers for each customer.*

In short, I will focus on the following key procedures to logically analyze the datasets and gain more insights to provide an offer recommendation to the customers.

1. Data Exploration
2. Data Preparation, Cleaning and Transformation
3. Descriptive data analysis.
4. Model Building with hyperparameter tuning
5. Model Deployment

I also plan to deploy these Jupyter notebooks on Amazon Sagemaker and create a model endpoint of the trained model. I will finally invoke this model endpoint on some test cases to generate offers we want to send out to our customers.

## Conclusion

In this project, I want to use data mining and machine learning techniques to explore the given datasets, build a predictive analytics model and finally deploy it on test samples to generate recommendations of influencing offers to Starbucks customers.

I plan to study this use case using AWS cloud services like Amazon Sagemaker and Amazon S3. The project should provide me a good opportunity to leverage my understanding of data mining and machine learning techniques using AWS cloud services.

## References


1. https://en.wikipedia.org/wiki/Recommender_system
2. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
3. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
4. https://barnraisersllc.com/2018/10/01/data-mining-process-essential-steps/
5. https://classroom.udacity.com/nanodegrees/nd009t/parts/2f120d8a-e90a-4bc0-9f4e-43c71c504879/modules/2c37ba18-d9dc-4a94-abb9-066216ccace1/lessons/4f0118c0-20fc-482a-81d6-b27507355985/concepts/480e9dc2-4726-4582-81d7-3b8e6a863450