Starbucks Offers: Customer Classification with Python

Project Overview

Driving customers sales behavior in the retail space via digital channels (such as email, mobile and social media, etc.) has been shown to be effective and has become an important tool in creating longevity of a brand, increasing profits and engagements, as well as building long lasting relationships with customers. Customer satisfaction drives business success and data analytics can provide insight into what customers think. Personalization is the process of using data to provide tailored experiences to shoppers hence improving the customer's experience. The phrase "360-degree customer view" refers to aggregating data describing a customer's purchases and customer service interactions. To fully give customers a personalized and special experience having data that provides a 360-degree view is paramount.

The Starbucks Udacity Data Scientist Nanodegree Capstone challenge data set is a simulation of customer behavior on the Starbucks rewards mobile application. Periodically, Starbucks sends offers to users that may be an advertisement, discount, or buy one get one free (BOGO). An important characteristic regarding this dataset is that not all users receive the same offer.

This data set contains three files. The first file, Offer Portfolio, describes offer characteristics including its duration, the amount a customer must spend to complete it (difficulty) and the channels where the promotional messages I sent through. The second file, Customer Profile, contains customer demographic data including their age, gender, income, and when they created an account on the Starbucks rewards mobile application. The third file describes customer purchases and when they received, viewed, and completed an offer. An offer can be showed to be completed in this data if a customer has met the difficulty requirements during the necessary duration. However, an offer will only be considered successful when a customer views an offer and meets or exceeds its difficulty within the offer's duration.

Problem Statement

The problem that I chose to solve was to build a model that predicts whether a customer will respond to an offer and the sort of offer the customer will respond to. I solved this using 3 major steps.

- First, I combined the offer portfolio, customer profile, and transaction data, and broke them into two data sets. One data set spoke to promotional offers and the other to informational offers. The rows of each dataset describe a customer's demographic data, the sort of offers they have received, the channels they've been engaged one and their response to these offers. For the promotional offer dataset, the response is a classification of the offers each customer has positively responded to i.e. either BOGO, discount, Both or None. While the Informational dataset checked whether a customer viewed the offer and their rate of viewing.
- I assessed the performances of each model based on the metrics specified below and compared them against each other.
- I refined the parameters of the best performing models to improve their performance. This was done for both models

Metrics

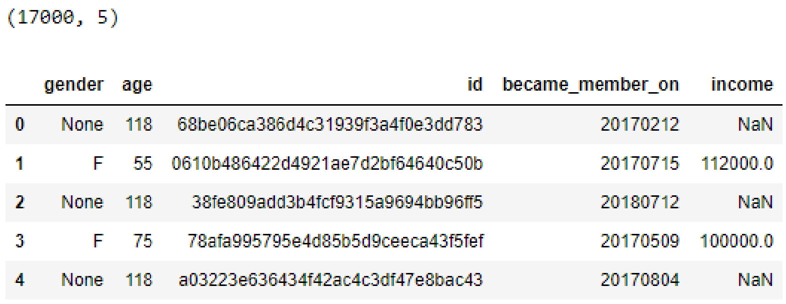
The metrics to be used to measure the performance of the solution will be accuracy, precision, recall and F-1 score. Accuracy, measures how well a model correctly predicts the sort of offer each customer is interested in. However, if the percentage of successful or unsuccessful offers is very low, accuracy will not be a good measure of model performance. For this situation, evaluating a model's precision and recall provides more insight to the model's performance. I chose the F1-score metric because it is "a weighted average of the precision and recall metrics" . These metrics were provided for the naïve predictors, logistic regressions and the other models used on the dataset. I assessed the precision and recall of a naive model that assumes all customers are interested in all offers received. This was done for both promotional type offers and informational offers. I also assessed a naïve logistic regression model for both types of offers. This provided me a baseline for evaluating the performance of models that was built.

Data Exploration/Exploratory Visualization

The goal of my exploration & visualization of the Starbucks Capstone Challenge data files is to identify the preprocessing steps that I need to apply prior to combining them.

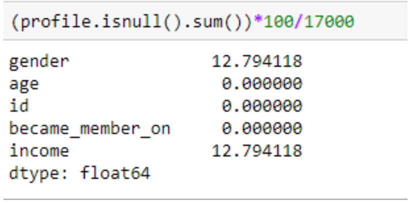
Customer Profile Data:

The customer profile data describes customer demographics. There are five characteristics of this data that I observed during my exploratory data analysis. First, gender and income have approximately 13% missing data. Second, customer age is 118 when customer income is missing (i.e. NaN). Third, customer gender is not specified for ~ 1.5% of the data. Fourth, the year that a customer became a rewards member is not uniformly distributed. This suggests that this feature may be a useful customer differentiator. Fifth, the month that a customer became a rewards member is approximately uniform. Therefore, this feature is probably not useful for predicting whether a customer offer was successful. I decided that if the customers with missing data have substantial transactions, they will be kept during the data clean up and their data replaced with values.



```
(17000, 5)
```

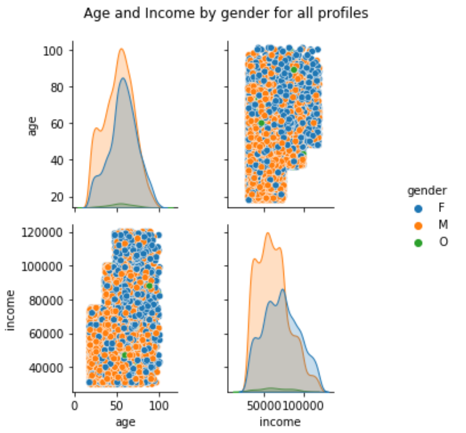
	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43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



```
(profile.isnull().sum())*100/17000
```

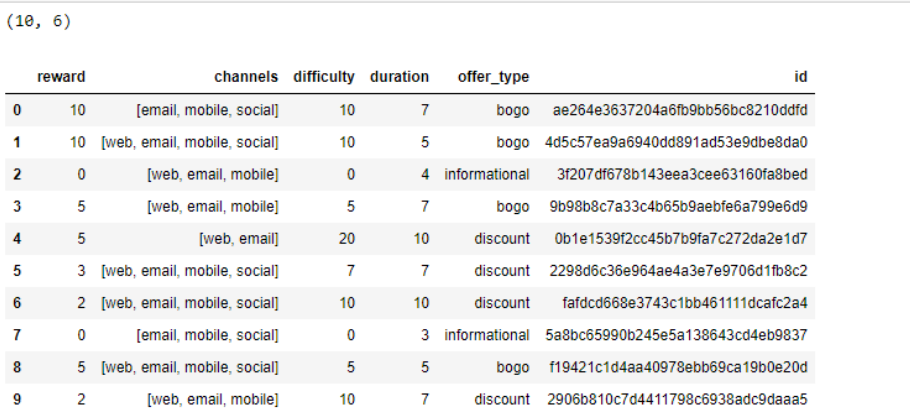
```
gender              12.794118
age                  0.000000
id                   0.000000
became_member_on     0.000000
income              12.794118
dtype: float64
```

The male population was larger in volume and age range than the female population. Customers between 18 and 25 years did not have incomes greater than 80k



Offer Portfolio Data:

- The Starbucks Capstone Challenge offer portfolio data summarizes offers shared with the customers. There are exactly 10 unique offers. Based on my exploratory analysis of this data, I decided that I would split the multi-label channels variable using the Scikit-learn MultiLabelBinarizer. Second, I should one hot encoded the offer types. Third, I need to rename the id variable to offer_id to distinguish it from customer identifiers.

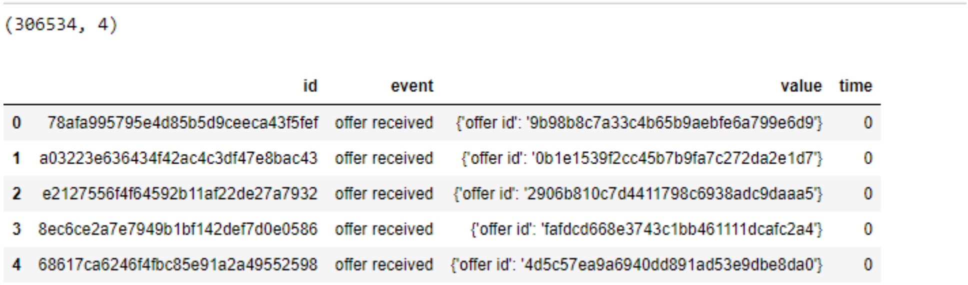


```
(10, 6)
```

	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

Customer Transaction Data:

The Starbucks Capstone Challenge customer transaction data describes customer purchases and when they received, viewed, and completed an offer. There are two conclusions that I drew from my exploratory analysis of this data. First, I need to separate customer offer and transactions data. Second, ~45% of the events are customer purchases and ~55% of events describe customer offers and responses.



```
(306534, 4)
```

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

Benchmark

A way to bench mark the model will be to use a Logistic Model Classifier model for both models. For the first model with 4 classes, an accuracy of 25% will show that the model is randomly guessing the classes. I also used a naïve predictor that assumes that all customers are interested in all Promotional offers (i.e. BOGO and discount).

For the second model, other than the logistic baseline model, an assumption will also be made that all customers are interested in receiving promotional offers. There are two classes for the second model so the benchmark will be 50% accuracy for the assumption.

Data preprocessing

Customer Profile Data:

The customer profile data describes customer demographics. The process I implemented for the data clean up:

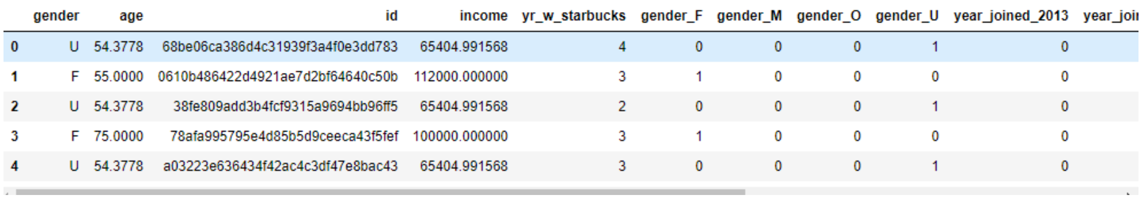
- I converted the membership date data to date time, then I created a new data point "years with Starbucks". This is a calculation of the years of membership.

- Second, I extracted the customers with Missing data and explored them separately. I joined them with the transaction data, to check if they had substantial transactions data. If they did, they would not be dropped from the profile list. The merge showed that they customers with incorrect ages, unknown income and gender had substantial transactions/interactions with Starbucks at 33,772 transactions/interactions for the 2175 profiles.

- The next step was to replace the missing gender data with U for unknown. Replace missing income data with the mean Income of the data set. The outlier ages were capped at 100

- One Hot encoding was done on customer membership year and gender and included back into the data frame.

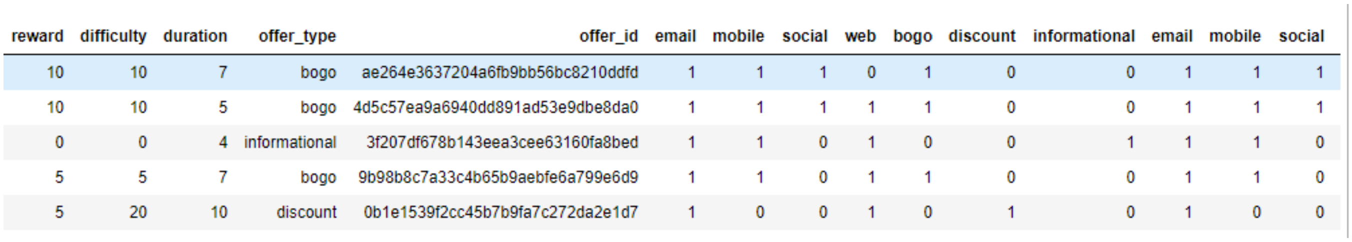
  Final Data frame looked like:



	gender	age	id	income	yr_w_starbucks	gender_F	gender_M	gender_O	gender_U	year_joined_2013	year_joi
0	U	54.3778	68be06ca386d4c31939f3a4f0e3dd783	65404.991568	4	0	0	0	1	0	
1	F	55.0000	0610b486422d4921ae7d2bf64640c50b	112000.000000	3	1	0	0	0	0	
2	U	54.3778	38fe809add3b4fcf9315a9694bb96ff5	65404.991568	2	0	0	0	1	0	
3	F	75.0000	78afa995795e4d85b5d9ceeca43f5fef	100000.000000	3	1	0	0	0	0	
4	U	54.3778	a03223e636434f42ac4c3df47e8bac43	65404.991568	3	0	0	0	1	0	

Offer Portfolio data:

- The Starbucks Capstone Challenge offer portfolio data summarizes customer offers. First, I split the multi-label channels variable using the Scikit-learn MultiLabelBinarizer. Second, I one hot encoded the offer types. Third, renamed the id variable to offer_id to distinguish it from customer identifiers. I dropped the Channels column



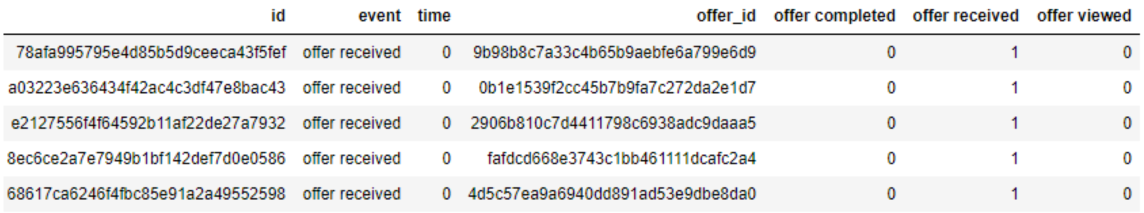
reward	difficulty	duration	offer_type	offer_id	email	mobile	social	web	bogo	discount	informational	email	mobile	social
10	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd	1	1	1	0	1	0	0	1	1	1
10	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0	1	1	1	1	1	0	0	1	1	1
0	0	4	informational	3f207df678b143eea3cee63160fa8bed	1	1	0	1	0	0	1	1	1	0
5	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9	1	1	0	1	1	0	0	1	1	0
5	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7	1	0	0	1	0	1	0	1	0	0

Customer Transaction/Interactions Data

The Starbucks Capstone Challenge customer transaction data describes customer purchases and when they received, viewed, and completed an offer. First, I renamed the 'person' column to 'id' to align with the profile data. Second, I separated customer offer and purchase data.

For the offer data, the steps I took below:

- I created a one hot encoding for the different event types: i.e. offer received, offer viewed and offer completed. I also dropped the value column as it had no financial values in it.



id	event	time	offer_id	offer completed	offer received	offer viewed
78afa995795e4d85b5d9ceeca43f5fef	offer received	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	0	1	0
a03223e636434f42ac4c3df47e8bac43	offer received	0	0b1e1539f2cc45b7b9fa7c272da2e1d7	0	1	0
e2127556f4f64592b11af22de27a7932	offer received	0	2906b810c7d4411798c6938adc9daaa5	0	1	0
8ec6ce2a7e7949b1bf142def7d0e0586	offer received	0	fafdcd668e3743c1bb461111dcafc2a4	0	1	0
68617ca6246f4fbc85e91a2a49552598	offer received	0	4d5c57ea9a6940dd891ad53e9dbe8da0	0	1	0

- I renamed the offer columns to offer_received, offer_completed, offer_viewed. For easier use in the feature engineering.

- Exploration showed that a customer can receive the same offer numerous times, so I picked the earliest time each customer received a certain offer using the minimum function. I also added 0.5 to all the times so that calculations will be easier and not bring up Inf values.

- I calculated the total offers received, viewed and completed per customer per promo. I also got the max time they received, viewed or completed an offer. I noticed some customers had Nan for completed or viewed offers. Which means they received it but did not view or complete, so I replaced Nan with zeros for the data frame. This solely populated the time_viewed, time_completed, offer_completed, offer_viewed columns.

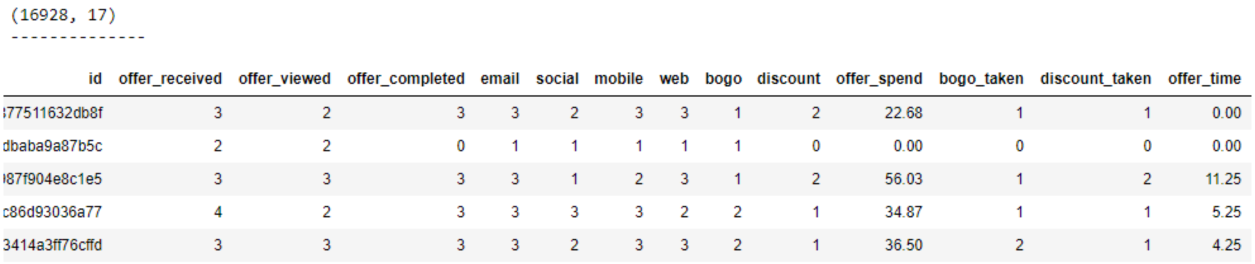
- I created a new data frame to calculate which transactions were completed after during a promotional offer. This was done by the following steps: a) merging the current offer data frame with the transaction data frame based on the customer's unique ID, dropping the event and value column, then comparing the time of the transaction completion to the time the offer was received. Transactions that did not fall within an offer reception time and offer completion time were considered non-offer related transactions.

  To track the transactions that were due to an offer was to use an algorithm on each type of offer. For BOGO and discounts; check if the spend during a promotion is greater than zero. If it is, then check if offer_viewed and offer_completed is greater than zero. if true then the offer is assumed to be completed, and the number of times It was completed is captured as Bogo_taken or discount_taken. For informational Offers, the algorithm checks if information was received and viewed.

- Time taken to react positively to an offer which was calculated in days by, ((time_completed) - (time viewed))/24. This was calculated for only customers who had either taken up a BOGO or Discount offer. Else the offer time was replaced with zero as the condition did not apply to Informational offers.

- Time taken to view an informational offer was also calculated by subtracting the time viewed from time received and dividing by 24. Customers actions were then grouped into two: Reactions to promotional offers and reactions to informational offers.

- For Promotional Offers: I aggregated the total promotional offers received, viewed, total spent during offers, total spent on products and the medium they were received per customer.



```
(16928, 17)
--------------
```

id	offer_received	offer_viewed	offer_completed	email	social	mobile	web	bogo	discount	offer_spend	bogo_taken	discount_taken	offer_time
77511632db8f	3	2	3	3	2	3	3	1	2	22.68	1	1	0.00
dbaba9a87b5c	2	2	0	1	1	1	1	1	0	0.00	0	0	0.00
87f904e8c1e5	3	3	3	3	1	2	3	1	2	56.03	1	2	11.25
c86d93036a77	4	2	3	3	3	3	2	2	1	34.87	1	1	5.25
3414a3ff76cffd	3	3	3	3	2	3	3	2	1	36.50	2	1	4.25

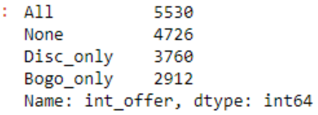
I also calculated the rate of conversion, i.e. total promotional offers received vs total that were viewed and completed. I also merged the customer profile data to this, and the total amount spent by each customer.

Merging all Files and Transformations:

I merged all the data frames and transformations done into two Data frames. One for the promotional offer and the other for the informational offers. For the promotional offer I merged the Profile, portfolio, Total Transactions and promotional offer conversion rate data together. For the Informational offer, I merged, Profile, portfolio, Total Transactions and informational offer conversion rate data together.

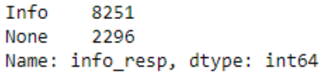
The final transformation done was to group customers according to their reactions/response to offers. For the promotional offers, this was grouped into 4: All (Bogo and Discount), None, Discount only and BOGO Only. This was to properly classify customers into the categories they were most suited or interested in. The classification was done by checking if they had viewed and completed only a BOGO offer, Only a discount offer, both or none of them

Distribution of customers and the types of offers they took up:



```
: All          5530
  None         4726
  Disc_only    3760
  Bogo_only    2912
  Name: int_offer, dtype: int64
```

For the Informational offers data frame, the final transformation was to group customers according to their receiving and viewing informational offers. Distribution of customers who view informational offers vs those who don't:



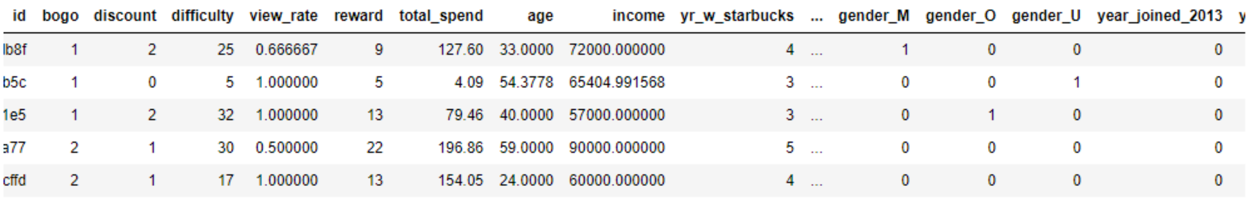
```
Info    8251
None    2296
Name: info_resp, dtype: int64
```

Predict Customer’s positive reaction to Promotional and informational offers

The problem that I chose to solve was to build a model that predicts whether a customer will respond to an offer and the sort of offer the customer will respond to. My strategy for solving this problem has several steps. For promotional offers and Informational offers. The steps for modelling were identical.

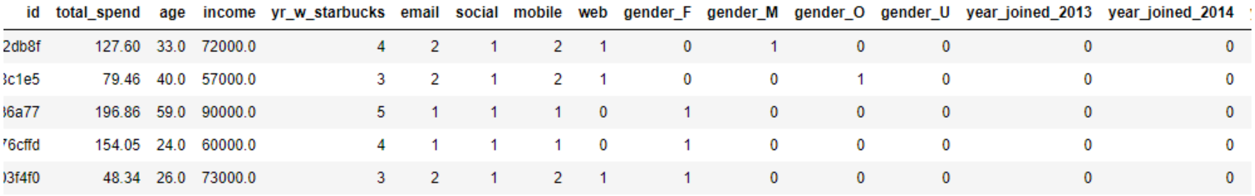
First, I combined offer portfolio, customer profile, and transaction data, then separated them to informational offer data set and promotional offer datasets. Both data sets were imbalanced which is why I went with tree-based models, as they perform well on imbalanced data sets.

Promotional dataset:



id	bogo	discount	difficulty	view_rate	reward	total_spend	age	income	yr_w_starbucks	...	gender_M	gender_O	gender_U	year_joined_2013	y
lb8f	1	2	25	0.666667	9	127.60	33.0000	72000.000000	4	...	1	0	0	0	
b5c	1	0	5	1.000000	5	4.09	54.3778	65404.991568	3	...	0	0	1	0	
1e5	1	2	32	1.000000	13	79.46	40.0000	57000.000000	3	...	0	1	0	0	
a77	2	1	30	0.500000	22	196.86	59.0000	90000.000000	5	...	0	0	0	0	
cffd	2	1	17	1.000000	13	154.05	24.0000	60000.000000	4	...	0	0	0	0	

Informational dataset:

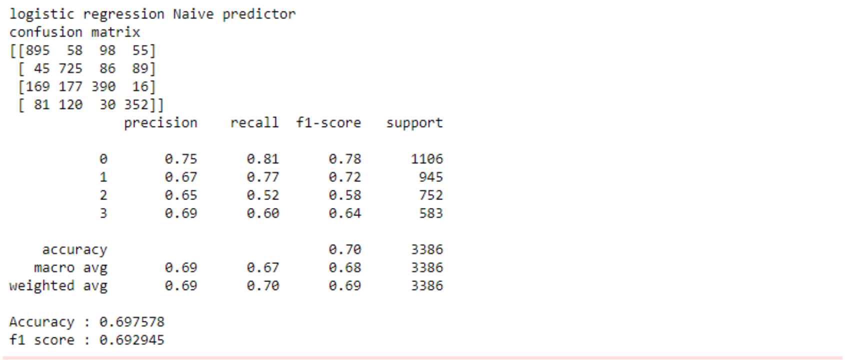


id	total_spend	age	income	yr_w_starbucks	email	social	mobile	web	gender_F	gender_M	gender_O	gender_U	year_joined_2013	year_joined_2014
2db8f	127.60	33.0	72000.0	4	2	1	2	1	0	1	0	0	0	0
3c1e5	79.46	40.0	57000.0	3	2	1	2	1	0	0	1	0	0	0
36a77	196.86	59.0	90000.0	5	1	1	1	0	1	0	0	0	0	0
76cffd	154.05	24.0	60000.0	4	1	1	1	0	1	0	0	0	0	0
03f4f0	48.34	26.0	73000.0	3	2	1	2	1	1	0	0	0	0	0

Second, I performed a Min-Max Scaler transformation on both dataset due to attributes like Income and customer total spend, which might cause bias in the model.

Third, I broke the datasets into to a train and test set prior to assessing accuracy and F1 score of the two naïve predictors. Fourth, I tried the naïve predictors and noted their values as my benchmark for building my model. The performance of both Naïve predictors:

Promotional Data:

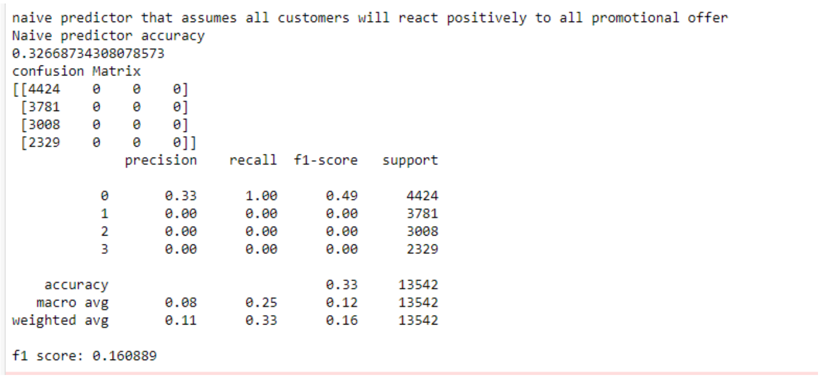


```
logistic regression Naive predictor
confusion matrix
[[895  58  98  55]
 [ 45 725  86  89]
 [169 177 390  16]
 [ 81 120  30 352]]
              precision    recall  f1-score   support

           0       0.75      0.81      0.78      1106
           1       0.67      0.77      0.72       945
           2       0.65      0.52      0.58       752
           3       0.69      0.60      0.64       583

    accuracy                           0.70      3386
   macro avg       0.69      0.67      0.68      3386
weighted avg       0.69      0.70      0.69      3386

Accuracy : 0.697578
f1 score : 0.692945
```



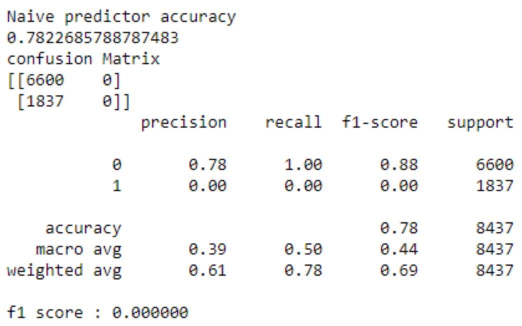
```
naive predictor that assumes all customers will react positively to all promotional offer
Naive predictor accuracy
0.32668734308078573
confusion Matrix
[[4424    0    0    0]
 [3781    0    0    0]
 [3008    0    0    0]
 [2329    0    0    0]]
              precision    recall  f1-score   support

           0       0.33      1.00      0.49      4424
           1       0.00      0.00      0.00      3781
           2       0.00      0.00      0.00      3008
           3       0.00      0.00      0.00      2329

    accuracy                           0.33     13542
   macro avg       0.08      0.25      0.12     13542
weighted avg       0.11      0.33      0.16     13542

f1 score: 0.160889
```

Informational Data:



```
Naive predictor accuracy
0.7822685788787483
confusion Matrix
[[6600    0]
 [1837    0]]
              precision    recall  f1-score   support

           0       0.78      1.00      0.88      6600
           1       0.00      0.00      0.00      1837

    accuracy                           0.78      8437
   macro avg       0.39      0.50      0.44      8437
weighted avg       0.61      0.78      0.69      8437

f1 score : 0.000000
```

Next, I tried a list of decision tree-based models for both Data sets because they were both imbalanced. I also tried using SMOTE resampling on both dataset but the performance (especially precision and recall) was better when the datasets were imbalanced. For the Promotional Dataset, I tried Random forest, Cat Boost and Gradient boosting models. For the promotional Offer, the best performing model was the Cat Boost Classifier, using a random search and K-fold classifiers did not improve the model performance at F1-score of 0.764 and accuracy of 0.761.

Model performance for promotional offers:

Model	Accuracy	F1 Score
RandomForestClassifier	0.761075	0.756533
GradientBoostingClassifier	0.764028	0.760893
CatBoostClassifier	0.764324	0.761438

For the Informational Offer, the best model in the decision classifier models used was the Gradient Boosting Classifier, after Using a random search for hyperparameter tuning, the Accuracy and F-1 score performance moved from 0.826 to 0.83 and 0.587 to 0.60 respectively.

Model Performance for Informational offers:

Model	Accuracy	F1 Score
RandomForestClassifier	0.811848	0.557268
GradientBoostingClassifier	0.825592	0.585705
CatBoostClassifier	0.819431	0.572168

Bias and variance are two characteristics of a machine learning model. Bias refers to inherent model assumptions regarding the decision boundary between different classes. On the other hand, variance refers a model's sensitivity to changes in its inputs. However, my exploratory analysis of customer demographics for each offer suggests that this decision boundary will be non-linear. Therefore, an ensemble method based on decision trees should perform better.
All three models, random forest, Cat boost and gradient boosting models are a combination of multiple decision trees. A random forest classifier randomly samples the training data with replacement to construct a set of decision trees that are combined using majority

voting. In contrast, gradient boosting iteratively constructs a set of decision trees with the goal of reducing the number of misclassified training data samples from the previous iteration. Lastly, Cat Boost Algorithm focuses on the maximizing the accuracy of split scoring with weighted sampling happening instead at tree level. A consequence of these model construction strategies is that the depth of decision trees generated during random forest model training are typically greater than gradient boosting weak learner depth to minimize model variance. Typically, gradient boosting performs better than a random forest classifier. However, gradient boosting may overfit the training data and requires additional effort to tune. A random forest classifier is less prone to overfitting because it constructs decision trees from random training data samples. Cat Boost is less prone to overfitting but performs faster than the other two algorithms, it performs best when there are categorical features n the dataset and one-hot-max-size is well tuned.

CONCLUSION:

The problem that I chose to solve was to build a model that predicts whether a customer will respond to an offer and the sort of offer the customer will respond to. My strategy for solving this problem has seven steps. For promotional offers:

First, I combined offer portfolio, customer profile, and transaction data, then separated them to informational offer data set and promotional offer datasets. Both data sets were imbalanced which is why I went with tree-based models, as they perform well on imbalanced data sets. Second, I assessed the accuracy and F1-score of two naïve models; a naive model that assumed customers were interested in all promotional offers and a baseline logistic regression model.

Third, I compared the performance of Random forest, Cat Boost and Gradient boosting models. This analysis suggests that Cat-Boost model has the best training data accuracy and F1-score. Fourth, I refined Cat boost model hyperparameters using a Random search and K-fold Cross-validation. My analysis suggests that the model test data set accuracy of 0.761 and F1-score of 0.761 suggesting that model did not overfit the training data.

Fifth, For the Informational Offers, I assessed the accuracy and F1-score of two naïve models; a naive model that assumed customers would view all informational offers and a baseline logistic regression model. Step six, I compared the performance of Decision Tree, Random Forest, Gradient Boosting and Cat Boost Classifiers. The best performing model in this second instance was the Gradient boosting Classifier. Using a random search grid, I was able to improve the performance of the testing from Accuracy of 0.826 and F-1 of 0.587 to 0.83 and 0.60 respectively.

"Feature importance" refers to a numerical value that describes a feature's contribution to building a model that maximizes its evaluation metric. A random forest classifier is an example of a model that estimates feature importance during training. My analysis of the Starbucks Capstone Challenge customer offer effectiveness training data suggests that the top five features for promotional offers based on their importance are:

```
1. How much the customer has spent with Starbucks
2. If the offer is a discount offer
3. If a customer generally views offers sent.
4. If the offer is a BOGO offer
5. Income of the customer
```

Since the top 4 features includes both types of offers, it may be necessary to remove the features of discount and BOGO and check if that will affect the performance. I also noticed after documentations that some of the customers who were classified as BOGO only or Discount Only, received only one type of offer from Starbucks. This means that we might not know how they will act until they have been offered both promotional offer types. So an option will be to build a model on customers who have received both promotional offer type and check performance. This will be a more accurate model to use. Lastly, Catboost is said to work best when there is categorical variable that is not one-hot Encoded as its method of encoding using one_hot_max_size produces the best result.

For Informational offers the top five features based on their importance were:

```
1. Offers sent via social media
2. Income of customer
3. Total the customer has already spent with Starbucks
4. Age of customer
5. Customers with unknown genders.
```

Since one of the top features are customers with unknown genders, it might be interesting to either assign these customers a gender or to drop them completely from the data set. The performance might be a more accurate predictor of customer’s likelihood to view informational offers.