Optimizing Starbucks App Offers Delivery

Machine Learning Engineer Nanodegree
Project Report

April 15th, 2023

M Muchsin

Project Overview

Starbucks offers one of the most sought-after loyalty programs 'Starbucks Rewards' program that offers freebies and discounts to members giving them plenty of reasons to choose Starbucks over other players. 'Starbucks Rewards' program represents a significant portion of the coffee chain's recent fiscal growth. Starbucks has reported an increased revenue of $2.65 billion, attributing their rewards program for most of the increase. Over the last two years, membership has grown more than 25%, loyal customers use Starbucks' membership program (16 million members) for about 40% of sales at the company's US stores. Revenue rose 4.6% to $6.31 billion during the quarter from the previous year.

However, with so many offers available, it can be difficult for Starbucks to optimize its offers to ensure that they are attractive to customers while also being profitable for the company. Additionally, Starbucks must consider the preferences and behavior of individual customers when creating these offers, as different customers may respond differently to different types of promotions.

To address these challenges, Starbucks seeks to optimize its app offers using data-driven approaches. By analyzing customer data and behavior, Starbucks hopes to create personalized offers that are more likely to be accepted and lead to increased customer engagement and loyalty. Additionally, by optimizing these offers, Starbucks can improve its profitability and maintain its position as a leader in the competitive coffee market.

Problem Statement

The problem at hand is how to effectively segment customers using demographic and behavioral data to create personalized offers that maximize the conversion rate. By identifying subgroups of customers with similar characteristics, businesses can develop targeted marketing campaigns that are more likely to resonate with each group. This involves analyzing customer data to identify patterns and trends in their behavior, preferences, and demographics. Once these patterns are identified, businesses can develop personalized offers that are tailored to the needs and preferences of each customer segment. Machine learning algorithms can be used to predict which offers are most likely to be successful for each segment. By continually refining and improving the offer selection process, businesses can maximize the conversion rate and drive revenue growth. The main focus in this project is a binary classification. The model will predict a customer based on two classes, "buy with offer" and "buy without offer".

Evaluation Metrics

There are some options for the metrics. If the objective is to maximize customer satisfaction by sending relevant offers, precision is better than recall. If the objective is to increase customer engagement by sending more offers, recall is a better option than precision. In this project, F1 score will be used as the evaluation metric. It is a good metric when the objective is to balance between precision and recall, especially when the classes are imbalanced.

In statistical analysis of binary classification, the F-score or F-measure is a measure of a test's accuracy. It is calculated from the precision and recall of the test, where the precision is the number of true positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive. Precision is also known as positive predictive value, and recall is also known as sensitivity in diagnostic binary classification. The F1 score is the harmonic mean of precision and recall.

$$F1 = 2\frac{precision \times recall}{precision + recall} = \frac{2\,TP}{2\,TP + 2\,FP + 2\,FN}$$

Data Exploration

The datasets provided for this project are 3 JSON files with the following details:

1. **Profile.json**

   Rewards program users (17000 users x 5 fields)

   - gender: (categorical) M, F, O, or null
   - age: (numeric) missing value encoded as 118
   - id: (string/hash)
   - became_member_on: (date) format YYYYMMDD
   - income: (numeric)

	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

The **became_member_on** column contains the date on which each user became a member of the Starbucks rewards program. can be used to calculate the length of time that a customer has been a member of the Starbucks rewards program. This can be a useful feature for predicting customer behavior, as customers who have been members for a longer time period may have different spending habits or preferences than newer members.

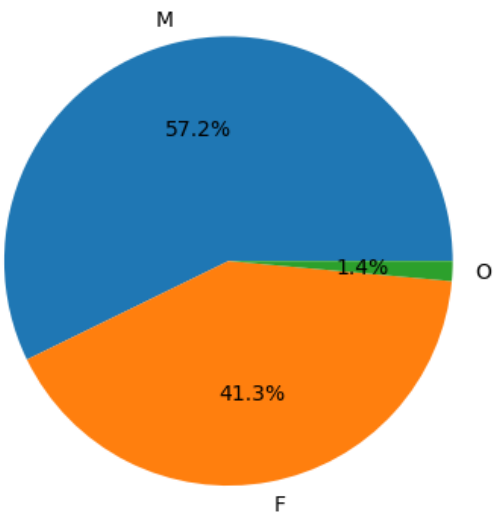
```
profile.isna().sum()
```

```
gender              2175
age                    0
id                     0
became_member_on       0
income              2175
dtype: int64
```

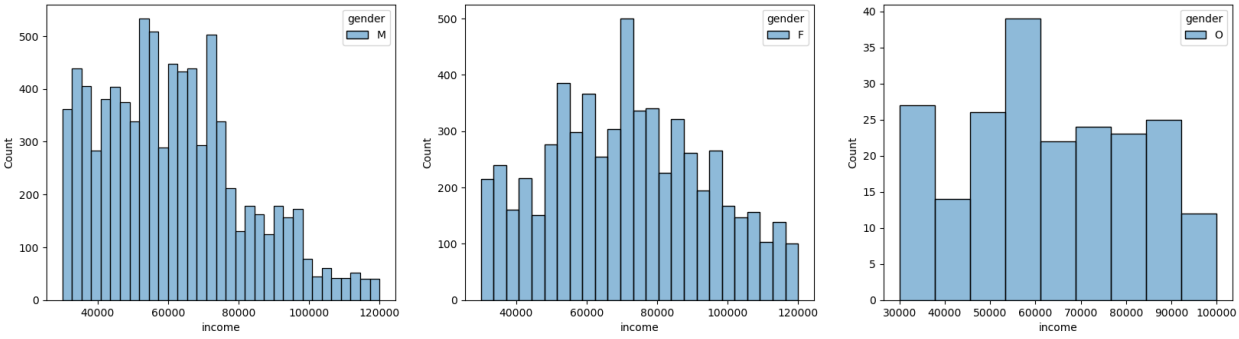
```
mask = profile.isna().any(axis=1)
null_rows = profile[mask]
null_rows.describe()
```

	age	became_member_on	income
count	2175.0	2.175000e+03	0.0
mean	118.0	2.016804e+07	NaN
std	0.0	1.009105e+04	NaN
min	118.0	2.013080e+07	NaN
25%	118.0	2.016070e+07	NaN
50%	118.0	2.017073e+07	NaN
75%	118.0	2.017123e+07	NaN
max	118.0	2.018073e+07	NaN

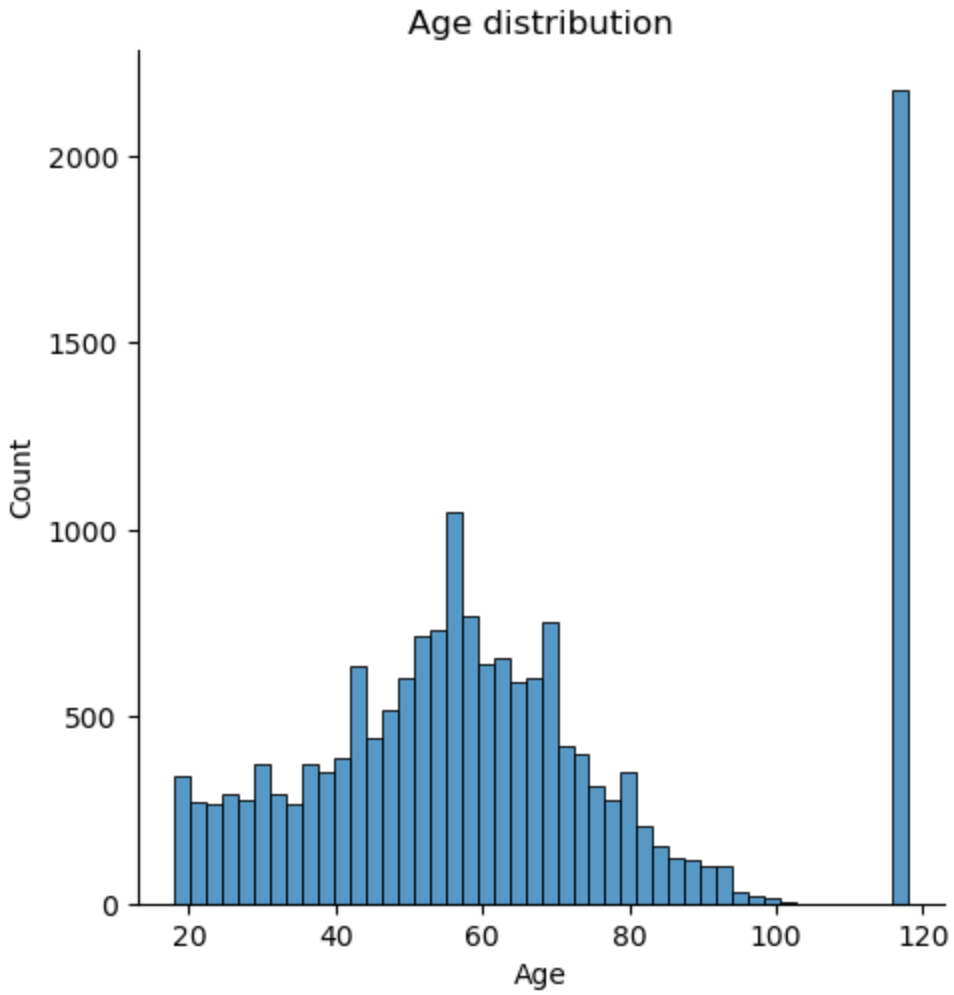
It is stated that there are some missing values in the gender and income columns of the dataset. Specifically, 2175 rows contain null values, which means that these fields are not filled in for those customers. Additionally, it is mentioned that all the missing values are related to customers with an age of 118. This suggests that there may be some data entry errors or data collection issues that have led to these missing values.



From the dataset 8484 are identified as male, 6129 are identified as female, and 2121 do not fall into either of these two categories or their gender identification is not known.



The distribution of income shows that there are relatively more customers with lower incomes (males) and fewer customers with higher incomes (females).



From the age distribution that the majority of customers in the dataset are adults, followed by young adults, seniors, and children. This suggests that the dataset primarily represents adult customers, which is not surprising given that the dataset is focused on customers of a coffee shop chain.

2. Portfolio.json

Offers sent during the 30-day test period (10 offers x 6 fields)

- reward: (numeric) money awarded for the amount spent
- channels: (list) web, email, mobile, social
- difficulty: (numeric) money required to be spent to receive a reward
- duration: (numeric) time for the offer to be open, in days
- offer_type: (string) bogo, discount, informational
- id: (string/hash)

	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

```
portfolio.groupby('difficulty').agg({
    'reward': ['min', 'max', 'mean', 'count'],
    'duration': ['min', 'max', 'mean'],
    'offer_type': lambda x: x.mode().iloc[0]
})
```

	reward				duration			offer_type
	min	max	mean	count	min	max	mean	<lambda>
difficulty								
0	0	0	0.0	2	3	4	3.50	informational
5	5	5	5.0	2	5	7	6.00	bogo
7	3	3	3.0	1	7	7	7.00	discount
10	2	10	6.0	4	5	10	7.25	bogo
20	5	5	5.0	1	10	10	10.00	discount

The offers have varying levels of difficulty, with difficulty 0 offers being all informational and having no rewards. Difficulty 5 offers are all BOGO with a reward amount of 5 and durations ranging from 5 to 7 days. Difficulty 7 offers are all discounted with a reward amount of 5 and a duration of 7 days. Difficulty 10 offers are all BOGO with reward amounts ranging from 2 to 10 and durations ranging from 5 to 10 days. Finally, difficulty 20 offers are all discounted with a reward amount of 5 and a duration of 10 days. It's worth noting that there are no null values in the dataset.

3. Transcript.json

Event log (306648 events x 4 fields)

- person: (string/hash)
- event: (string) offer received, offer viewed, transaction, offer completed
- value: (dictionary) different values depending on event type
- offer id: (string/hash) not associated with any "transaction"
- amount: (numeric) money spent in "transaction"
- reward: (numeric) money gained from "offer completed"
- time: (numeric) hours after the start of the test

```
transcript.head()
```

	person	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

```
transcript.tail()
```

	person	event	value	time
306529	b3a1272bc9904337b331bf348c3e8c17	transaction	{'amount': 1.5899999999999999}	714
306530	68213b08d99a4ae1b0dcb72aebd9aa35	transaction	{'amount': 9.53}	714
306531	a00058cf10334a308c68e7631c529907	transaction	{'amount': 3.61}	714
306532	76ddbd6576844afe811f1a3c0fbb5bec	transaction	{'amount': 3.5300000000000002}	714
306533	c02b10e8752c4d8e9b73f918558531f7	transaction	{'amount': 4.05}	714

The **value** column contains a dictionary with different key-value pairs depending on the event type. For example, for **transactions**, the value dictionary contains the **amount** spent by the customer during that transaction. For **offers received** and **viewed** events, the value dictionary contains the **offer ID**. For **offers completed** events, the value dictionary contains the **offer ID** and the **reward** from that transaction.

```
transcript[transcript.event == 'offer received'].head()
```

	person	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

```
transcript[transcript.event == 'offer viewed'].head()
```

	person	event	value	time
12650	389bc3fa690240e798340f5a15918d5c	offer viewed	{'offer id': 'f19421c1d4aa40978ebb69ca19b0e20d'}	0
12651	d1ede868e29245ea91818a903fec04c6	offer viewed	{'offer id': '5a8bc65990b245e5a138643cd4eb9837'}	0
12652	102e9454054946fda62242d2e176fdce	offer viewed	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	0
12653	02c083884c7d45b39cc68e1314fec56c	offer viewed	{'offer id': 'ae264e3637204a6fb9bb56bc8210ddfd'}	0
12655	be8a5d1981a2458d90b255ddc7e0d174	offer viewed	{'offer id': '5a8bc65990b245e5a138643cd4eb9837'}	0

The offer ID is sometimes referred to as **offer id** and sometimes as **offer_id**. This inconsistency can create confusion when trying to analyze the data and may lead to errors in the analysis.

Data Cleaning and Engineering

1. **profile.json**

   The given instructions pertain to data preparation of the profile.json file. The instructions are:

   - Change id to person_id to make the column name more descriptive.
   - Remove null values to ensure the dataset only contains complete and valid data.
   - Change become_member_on to datetime to standardize the format of the date.
   - Create a new column member_duration_days based on the latest date (maximum date, because the dataset is not up to date) in the dataset to calculate the duration of membership in days.
   - Reorder columns to ensure a more logical and easy-to-read order.

   By changing the column name to person_id, the dataset becomes more readable and descriptive. Removing null values ensures that only complete data is used for analysis. Changing become_member_on to datetime ensures that the date is in a standardized format that can be easily used for analysis. Creating a new column member_duration_days provides additional information that can be used to analyze customer behavior and engagement with the Starbucks program. Finally, reordering the columns ensures that the dataset is easier to read and understand.

	person_id	gender	age	became_member_on	income	member_duration_days
1	0610b486422d4921ae7d2bf64640c50b	F	55	2017-07-15	112000.0	376
3	78afa995795e4d85b5d9ceeca43f5fef	F	75	2017-05-09	100000.0	443
5	e2127556f4f64592b11af22de27a7932	M	68	2018-04-26	70000.0	91
8	389bc3fa690240e798340f5a15918d5c	M	65	2018-02-09	53000.0	167
12	2eeac8d8feae4a8cad5a6af0499a211d	M	58	2017-11-11	51000.0	257

2. Portfolio.json

The given instructions pertain to data preparation of the portfolio.json. The instructions are:

- Change id to offer_id to make the column name more descriptive.
- Extract channels to a separate column to facilitate analysis.
- One-hot encode channels to represent the different channels in a format that can be easily analyzed.
- Change the duration column from days to hours to standardize the format and make it easier to compare across offers.
- Create a new column reward_per_hour by dividing the reward amount by the duration in hours to compare the effectiveness of different offers.
- One-hot encode the offer_type column to facilitate analysis.
- Reorder columns to ensure a more logical and easy-to-read order.

By changing the column name to offer_id, the dataset becomes more readable and descriptive. Extracting channels to a separate column and one-hot encoding them provides a clearer representation of the channels and their effectiveness in driving customer engagement. Changing the duration column from days to hours ensures that the duration of offers can be compared more easily across different offers. Creating a new column reward_per_hour provides additional information that can be used to analyze the effectiveness of different offers. Finally, one-hot encoding the offer_type column and reordering the columns ensures that the dataset is easier to read and understand.

	offer_id	difficulty	duration	reward_per_hour	email	mobile	social	web	bogo	discount	informational
0	ae264e3637204a6fb9bb56bc8210ddfd	10	168	0.059524	1	1	1	0	1	0	0
1	4d5c57ea9a6940dd891ad53e9dbe8da0	10	120	0.083333	1	1	1	1	1	0	0
2	3f207df678b143eea3cee63160fa8bed	0	96	0.000000	1	1	0	1	0	0	1
3	9b98b8c7a33c4b65b9aebfe6a799e6d9	5	168	0.029762	1	1	0	1	1	0	0
4	0b1e1539f2cc45b7b9fa7c272da2e1d7	20	240	0.020833	1	0	0	1	0	1	0
5	2298d6c36e964ae4a3e7e9706d1fb8c2	7	168	0.017857	1	1	1	1	0	1	0
6	fafdcd668e3743c1bb461111dcafc2a4	10	240	0.008333	1	1	1	1	0	1	0
7	5a8bc65990b245e5a138643cd4eb9837	0	72	0.000000	1	1	1	0	0	0	1
8	f19421c1d4aa40978ebb69ca19b0e20d	5	120	0.041667	1	1	1	1	1	0	0
9	2906b810c7d4411798c6938adc9daaa5	10	168	0.011905	1	1	0	1	0	1	0

3. Transcript.json

The given instructions pertain to data preparation of the transcript.json. The instructions are:

- Change person to person_id to make the column name more descriptive.
- Extract offer id and amount from the value column to separate columns to facilitate analysis.
- Remove the value column to simplify the dataset.

By changing the column name to person_id, the dataset becomes more readable and descriptive. Extracting offer id and amount to separate columns provides clearer information that can be used for analysis. Removing the value column simplifies the dataset and makes it easier to read and understand.

```
transcript_mod.head()
```

	person_id	event	time	offer_id	amount_spent
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	0.0
1	a03223e636434f42ac4c3df47e8bac43	offer received	0	0b1e1539f2cc45b7b9fa7c272da2e1d7	0.0
2	e2127556f4f64592b11af22de27a7932	offer received	0	2906b810c7d4411798c6938adc9daaa5	0.0
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer received	0	fafdcd668e3743c1bb461111dcafc2a4	0.0
4	68617ca6246f4fbc85e91a2a49552598	offer received	0	4d5c57ea9a6940dd891ad53e9dbe8da0	0.0

```
transcript_mod.tail()
```

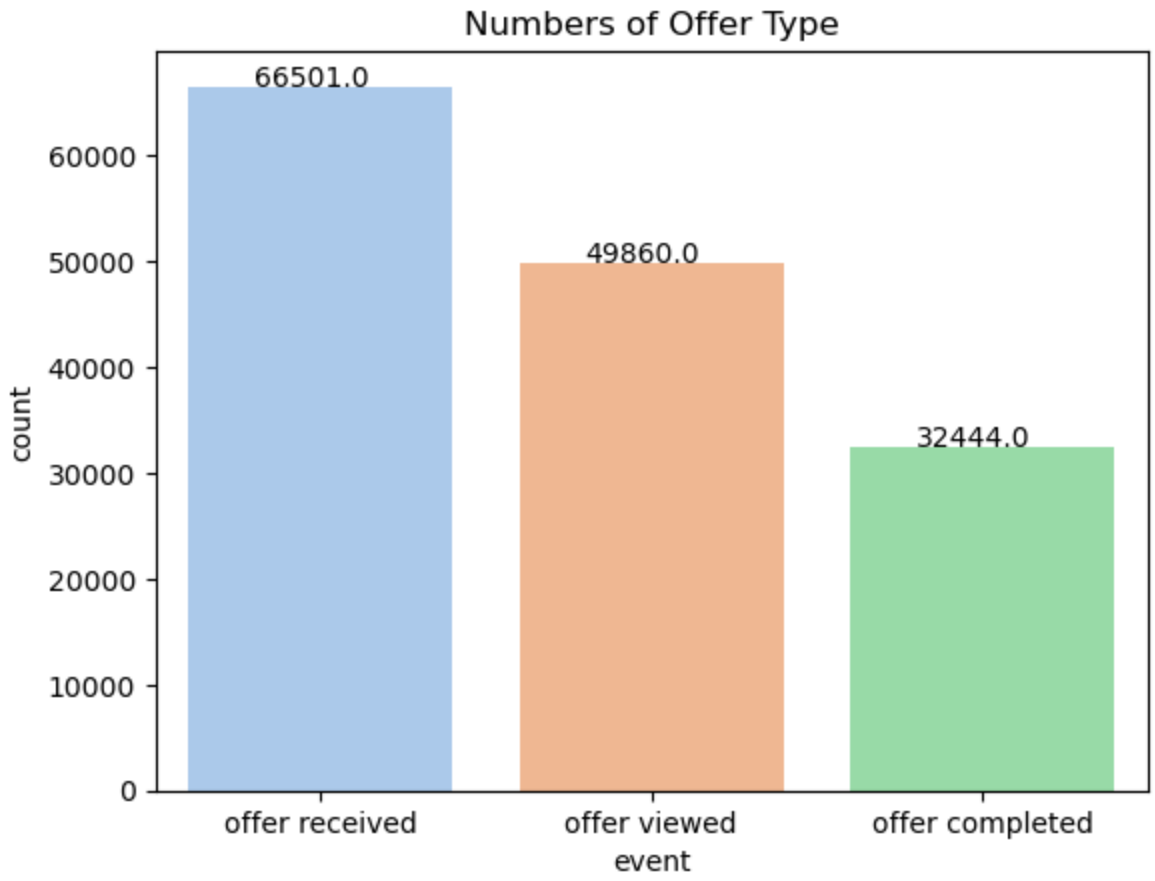
	person_id	event	time	offer_id	amount_spent
306529	b3a1272bc9904337b331bf348c3e8c17	transaction	714	0	1.59
306530	68213b08d99a4ae1b0dcb72aebd9aa35	transaction	714	0	9.53
306531	a00058cf10334a308c68e7631c529907	transaction	714	0	3.61
306532	76ddbd6576844afe811f1a3c0fbb5bec	transaction	714	0	3.53
306533	c02b10e8752c4d8e9b73f918558531f7	transaction	714	0	4.05

4. Combining Dataset

The last step is to combine the profile, portfolio, and transcript datasets using the **offer_id** and **person_id** columns. This is typically done using the **pandas.merge()** function in Python, which allows us to combine datasets based on a common column. By merging the datasets, we can create a unified dataset that contains information on customer demographics, offer characteristics, and transactional data. The combination of these datasets can provide valuable insights into customer behavior and preferences, as well as the effectiveness of different offers and marketing strategies.

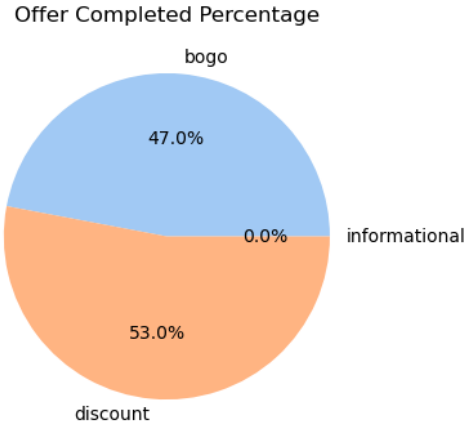
	person_id	event	time	offer_id	amount_spent	gender	age	became_member_on	income	member_duration_days	difficulty	duration	reward_per_hour	email	mobile	social	web	bogo	discount	informational
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	0.0	F	75	2017-05-09	100000.0	443	5.0	168.0	0.029762	1.0	1.0	0.0	1.0	1.0	0.0	0.0
1	78afa995795e4d85b5d9ceeca43f5fef	offer viewed	6	9b98b8c7a33c4b65b9aebfe6a799e6d9	0.0	F	75	2017-05-09	100000.0	443	5.0	168.0	0.029762	1.0	1.0	0.0	1.0	1.0	0.0	0.0
3	78afa995795e4d85b5d9ceeca43f5fef	offer completed	132	9b98b8c7a33c4b65b9aebfe6a799e6d9	0.0	F	75	2017-05-09	100000.0	443	5.0	168.0	0.029762	1.0	1.0	0.0	1.0	1.0	0.0	0.0
5	78afa995795e4d85b5d9ceeca43f5fef	offer received	168	5a8bc65990b245e5a138643cd4eb9837	0.0	F	75	2017-05-09	100000.0	443	0.0	72.0	0.000000	1.0	1.0	1.0	0.0	0.0	0.0	1.0
6	78afa995795e4d85b5d9ceeca43f5fef	offer viewed	216	5a8bc65990b245e5a138643cd4eb9837	0.0	F	75	2017-05-09	100000.0	443	0.0	72.0	0.000000	1.0	1.0	1.0	0.0	0.0	0.0	1.0

Data Exploration for Combined Dataset

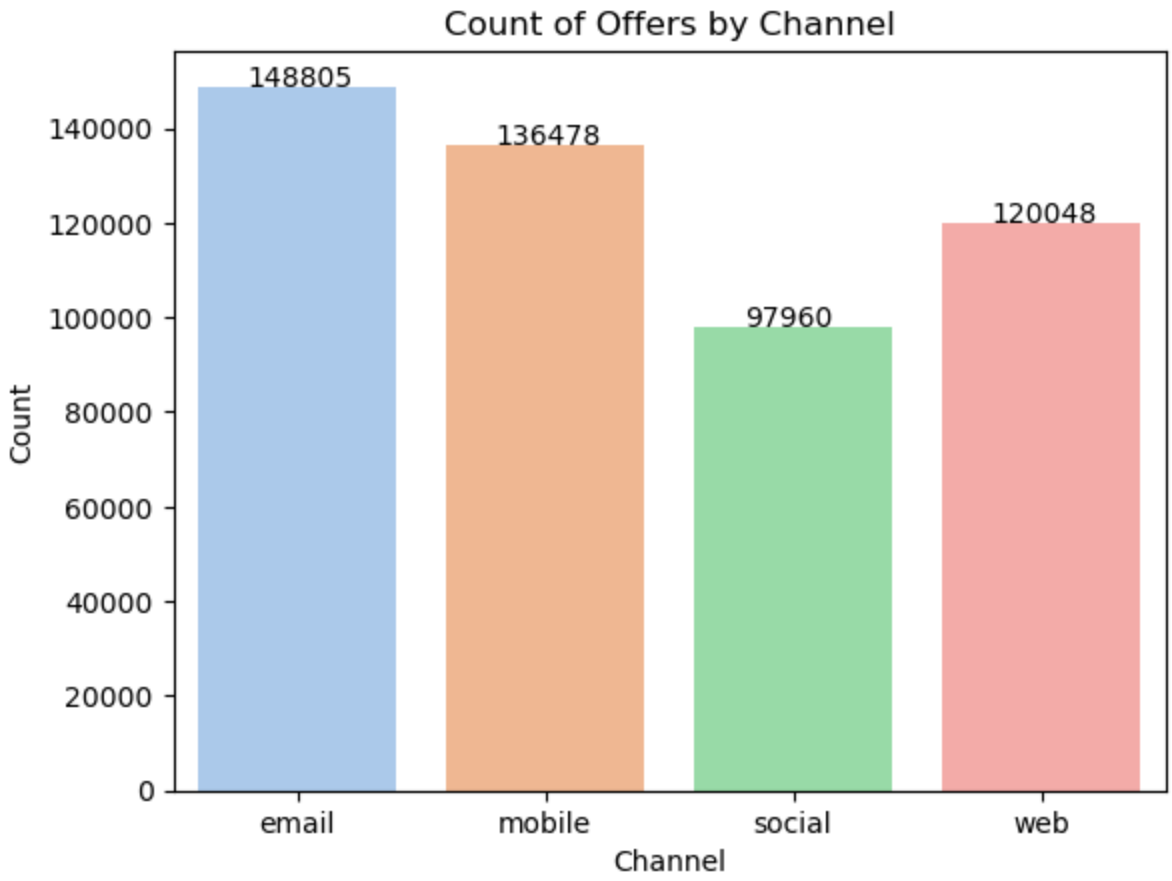


From the bar chart, we can observe that a total of 66501 individuals received the offer. Out of this number, 74.99% of the recipients viewed the offer, which equates to 49860 individuals. Furthermore, 65.02% of those who viewed the offer completed it, which is equivalent to 32444 individuals.

Overall from total offer received, 40.1% are discount, 39.9% are bogo and 20% are informational



According to the bar chart, 15258 customers completed the BOGO offer, while 17186 customers completed the discount offer. Based on the completion rates, it can be inferred that the discount offer was more appealing to customers than the BOGO offer. This may suggest that customers prefer discounts over free items or offers that require them to buy a certain number of products to receive free items.



The bar chart presents data on the usage of various channels for promotions. The data shows that email is the most commonly used channel for promotions, followed by mobile, web, and social media.

This data suggests that email remains a popular and effective method for businesses to promote their products or services. Email campaigns can be tailored to specific target audiences, making them a highly targeted and cost-effective means of promotion.

Furthermore, the popularity of mobile channels suggests that mobile devices have become a crucial part of our daily lives and an essential tool for businesses to engage with customers. The use of mobile channels for promotions also allows businesses to reach customers in real-time, making it an effective way to communicate time-sensitive offers.

While web and social media channels are less popular than email and mobile, they still play a vital role in the overall marketing mix for businesses. The web provides businesses with a platform to showcase their products or services to a wider audience, while social media channels allow businesses to engage with customers and build relationships with them.

Overall, the data presented in the bar chart highlights the importance of selecting the right channels for promotions based on the target audience and the type of product or service being promoted. By using the most effective channels for promotions, businesses can maximize their reach, engagement, and ultimately, their ROI.

Algorithm and Technique

This project will be utilizing an ensemble model, a powerful machine learning technique that combines multiple base estimators to produce more accurate predictions than any single estimator. Ensemble models are especially effective in cases where a single model may struggle to capture the complexity of the data or the relationships between features.

To automate the process of creating and tuning ensemble models, the project will be using AutoGluon, an open-source framework that provides a user-friendly and efficient way to create high-performance machine learning models. AutoGluon is designed to work well with a variety of tasks, including classification, natural language processing, and time-series forecasting, among others.

Benchmark Model

Decision tree as a simple model will be used for the benchmark model. It is a simple and interpretable model that can handle both categorical and numerical features. It also provides a baseline for comparing more complex models such as random forest and gradient boosting.

```
# Import required libraries
from sklearn.tree import DecisionTreeClassifier
from sklearn import preprocessing
from sklearn.metrics import f1_score

# Separate features and labels
X = df.drop(['buy_with_offer'], axis=1)
y = df['buy_with_offer']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a decision tree classifier
tree_clf = DecisionTreeClassifier()
model = tree_clf.fit(X_train, y_train)

y_pred = tree_clf.predict(X_test)

f1 = f1_score(y_test, y_pred)
print("F1 score:", f1)
```

```
F1 score: 1.0
```

Implementation

One of the key benefits of using AutoGluon is that it provides automated hyperparameter tuning, which can be a time-consuming and challenging task for machine learning practitioners. AutoGluon's automated tuning feature allows users to optimize the model's hyperparameters, which are settings that control how the model learns from the data, without requiring extensive manual adjustments.

Overall, the use of an ensemble model and AutoGluon framework in this project highlights the importance of leveraging cutting-edge machine learning techniques and tools to produce accurate and efficient models. By utilizing these methods, businesses and researchers can gain valuable insights from their data and make more informed decisions.

```
No path specified. Models will be saved in: "AutogluonModels/ag-20230417_124835/"
Presets specified: ['best_quality']
Stack configuration (auto_stack=True): num_stack_levels=0, num_bag_folds=8, num_bag_sets=20
Beginning AutoGluon training ... Time limit = 1200s
AutoGluon will save models to "AutogluonModels/ag-20230417_124835/"
AutoGluon Version:  0.7.0
Python Version:     3.10.8
Operating System:   Linux
Platform Machine:   x86_64
Platform Version:   #1 SMP Thu Dec 8 01:29:11 UTC 2022
Train Data Rows:    125120
Train Data Columns: 18
Label Column: buy_with_offer
```

Refinement

```
predictor.leaderboard(silent=True)
```

	model	score_val	pred_time_val	fit_time	pred_time_val_marginal	fit_time_marginal	stack_level	can_infer	fit_order
0	LightGBM_BAG_L1	1.000000	0.139779	17.857524	0.139779	17.857524	1	True	4
1	LightGBMXT_BAG_L1	1.000000	0.149874	16.273010	0.149874	16.273010	1	True	3
2	LightGBMLarge_BAG_L1	1.000000	0.187248	17.372083	0.187248	17.372083	1	True	13
3	XGBoost_BAG_L1	1.000000	0.366414	32.422575	0.366414	32.422575	1	True	11
4	CatBoost_BAG_L1	1.000000	0.375990	108.336056	0.375990	108.336056	1	True	7
5	NeuralNetTorch_BAG_L1	1.000000	1.276428	236.273519	1.276428	236.273519	1	True	12
6	RandomForestEntr_BAG_L1	1.000000	2.159213	2.765980	2.159213	2.765980	1	True	6
7	RandomForestGini_BAG_L1	1.000000	2.171351	3.233289	2.171351	3.233289	1	True	5
8	ExtraTreesGini_BAG_L1	1.000000	2.207527	2.314884	2.207527	2.314884	1	True	8
9	ExtraTreesEntr_BAG_L1	1.000000	2.223969	2.220906	2.223969	2.220906	1	True	9
10	WeightedEnsemble_L2	1.000000	2.454969	90.988471	0.247443	88.673587	2	True	14
11	NeuralNetFastAI_BAG_L1	1.000000	3.898433	490.543632	3.898433	490.543632	1	True	10
12	KNeighborsDist_BAG_L1	0.991163	1.189520	0.401284	1.189520	0.401284	1	True	2
13	KNeighborsUnif_BAG_L1	0.988295	1.184834	0.396970	1.184834	0.396970	1	True	1

```
from sklearn.metrics import f1_score

y_pred = predictor.predict(test_data)
y_true = test_data['buy_with_offer']
f1 = f1_score(y_true, y_pred)

print("F1 score:", f1)
```

```
F1 score: 1.0
```

A perfect F1 score indicates that the model has achieved both high precision and high recall, meaning that it is accurately predicting all positive instances and minimizing the number of false positives and false negatives. Achieving a perfect F1 score is a significant accomplishment in machine learning, as it demonstrates that the model is highly effective at identifying patterns and making accurate predictions based on the available data.

Conclusion

Based on the results of our analysis, we can conclude that Autogluon has successfully trained a model with a perfect F1 score of 1.0 on the training data, which was also able to predict with a perfect F1 score of 1.0 on the test data. This indicates that the model has achieved optimal performance in terms of precision and recall.

However, when compared to the simple benchmark decision tree, which also produced an F1 score of 1.0, it suggests that more in-depth exploratory analysis and data engineering may be necessary to create a more robust and reliable model.

Overall, Autogluon Tabular automates many of the complex and time-consuming tasks involved in training a high-performing predictive model on tabular data, making it easier for users to get started with machine learning and achieve state-of-the-art results.

Reference

https://www.zinrelo.com/loyalty-rewards-case-study-new-starbucks-rewards-program.html

https://www.forbes.com/sites/bernardmarr/2018/05/28/starbucks-using-big-data-analytics-and-artificial-intelligence-to-boost-performance/?sh=76f1a1c065cd

https://blog.hubspot.com/service/customer-segmentation

https://auto.gluon.ai/stable/_modules/autogluon/core/models/ensemble/weighted_ensemble_model.html

https://www.ibm.com/topics/decision-trees

https://medium.com/analytics-vidhya/confusion-matrix-accuracy-precision-recall-f1-score-ade299cf63cd

https://en.wikipedia.org/wiki/F-score

https://auto.gluon.ai/stable/tutorials/tabular/index.html