Starbucks Capstone Project Proposal

Vanitha B Selvarajan | July, 2021

Udacity – Machine Learning Engineer Nanodegree

Domain Background

Starbucks is a largest and passionate vendor of coffee and other beverages, headquartered in Seattle, Washington. The corporation is ranked 125 in the list of 2021 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 project will aim towards maximizing the profits for starbucks by using Machine Learning model to predict 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. This prediction will eventually improve the conversion rate of the offers and in turn increase the profitability.

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 [2]:
```
#Access the portfolio data
portfolio.head(10)
```

Out[2]:

	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

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 [9]:
```
#Exploring the profile
profile.head()
```

Out[9]:

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

```
profile: Rows = 17000, Columns = 5
```

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

In [16]:
```
#Explore transcript data
transcript.head(10)
```

Out[16]:

	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'}
5	offer received	389bc3fa690240e798340f5a15918d5c	0	{'offer id': 'f19421c1d4aa40978ebb69ca19b0e20d'}
6	offer received	c4863c7985cf408faee930f111475da3	0	{'offer id': '2298d6c36e964ae4a3e7e9706d1fb8c2'}
7	offer received	2eeac8d8feae4a8cad5a6af0499a211d	0	{'offer id': '3f207df678b143eea3cee63160fa8bed'}
8	offer received	aa4862eba776480b8bb9c68455b8c2e1	0	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}
9	offer received	31dda685af34476cad5bc968bdb01c53	0	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}

In [17]:
```
print("transcript: Rows = {0}, Columns = {1}".format(str(transcript.shape[0]), str(transcript.shape[1])))
```

```
transcript: Rows = 306534, Columns = 4
```

Solution Statement

My Aim is to develop a Machine Learning model 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:

- ✓ most responded offer
- ✓ response to an offer
- ✓ age group & gender groups which are greatly interested in offers

Since the data sets falls under the supervised binary classification problem, To find out the appropriate response of a customer to an offer, I'll be leveraging models like RandomForestClassifier and DecisionTreeClassifier, AdaBoostClassifier 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 simple, a fast and considered standard method for binary classification machine learning problems and evaluate the model result using F1 score as the evaluation metric.

Since the KNN algorithm requires no training before making predictions, new data can be added seamlessly which will not impact the accuracy of the algorithm

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 and Presentation

Data Preprocessing:

This will involve removing or processing any missing values and cleaning the datasets to analyse the data better.

In [23]:
```
cleaned_portfolio
```

Out[23]:

	difficulty	duration	offer_id	reward	web	email	mobile	social	offer_bogo	offer_discount	offer_informational
0	10	168	ae264e3637204a6fb9bb56bc8210ddfd	10	0	1	1	1	1	0	0
1	10	120	4d5c57ea9a6940dd891ad53e9dbe8da0	10	1	1	1	1	1	0	0
2	0	96	3f207df678b143eea3cee63160fa8bed	0	1	1	1	0	0	0	1
3	5	168	9b98b8c7a33c4b65b9aebfe6a799e6d9	5	1	1	1	0	1	0	0
4	20	240	0b1e1539f2cc45b7b9fa7c272da2e1d7	5	1	1	0	0	0	1	0
5	7	168	2298d6c36e964ae4a3e7e9706d1fb8c2	3	1	1	1	1	0	1	0
6	10	240	fafdcd668e3743c1bb461111dcafc2a4	2	1	1	1	1	0	1	0

In [26]:
```
cleaned_profile
```

Out[26]:

	became_member_on	gender	customer_id	customer_income	memberdays	Age_group
1	20170715	F	0610b486422d4921ae7d2bf64640c50b	112000.0	1450	46-60
3	20170509	F	78afa995795e4d85b5d9ceeca43f5fef	100000.0	1517	61-80
5	20180426	M	e2127556f4f64592b11af22de27a7932	70000.0	1165	61-80
8	20180209	M	389bc3fa690240e798340f5a15918d5c	53000.0	1241	61-80
12	20171111	M	2eeac8d8feae4a8cad5a6af0499a211d	51000.0	1331	46-60
13	20170911	F	aa4862eba776480b8bb9c68455b8c2e1	57000.0	1392	61-80
14	20140213	M	e12aeaf2d47d42479ea1c4ac3d8286c6	46000.0	2698	20-45
15	20160211	F	31dda685af34476cad5bc968bdb01c53	71000.0	1970	61-80
16	20141113	M	62cf5e10845442329191fc246e7bcea3	52000.0	2425	46-60

In [32]:
```
cleaned_transcript
```

Out[32]:

	event	customer_id	time	value	offer_id	money_gained
0	offer received	78afa995795e4d85b5d9ceeca43f5fef	0	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	9b98b8c7a33c4b65b9aebfe6a799e6d9	0.0
1	offer received	a03223e636434f42ac4c3df47e8bac43	0	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0b1e1539f2cc45b7b9fa7c272da2e1d7	0.0
2	offer received	e2127556f4f64592b11af22de27a7932	0	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}	2906b810c7d4411798c6938adc9daaa5	0.0
3	offer received	8ec6ce2a7e7949b1bf142def7d0e0586	0	{'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}	fafdcd668e3743c1bb461111dcafc2a4	0.0
4	offer received	68617ca6246f4fbc85e91a2a49552598	0	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	4d5c57ea9a6940dd891ad53e9dbe8da0	0.0
5	offer	[illegible]	0	{'offer id':	[illegible]	0.0

Data Merging:

At this stage, preparing the final dataframe by merging the cleaned data sets and cleaning the final data.

```
In [36]: merged_df.head()
```

Out[36]:

	difficulty	duration	offer_id	reward	web	email	mobile	social	offer_bogo	offer_discount	...	customer_id
0	10	168	ae264e3637204a6fb9bb56bc8210ddfd	10	0	1	1	1	1	0	...	4b0da7e80e5945209a1fdddfe813c
1	10	168	ae264e3637204a6fb9bb56bc8210ddfd	10	0	1	1	1	1	0	...	4b0da7e80e5945209a1fdddfe813c
2	10	168	ae264e3637204a6fb9bb56bc8210ddfd	10	0	1	1	1	1	0	...	4b0da7e80e5945209a1fdddfe813c
3	10	168	ae264e3637204a6fb9bb56bc8210ddfd	10	0	1	1	1	1	0	...	4b0da7e80e5945209a1fdddfe813c
4	10	168	ae264e3637204a6fb9bb56bc8210ddfd	10	0	1	1	1	1	0	...	4b0da7e80e5945209a1fdddfe813c

5 rows × 22 columns

```
In [69]: cleaned_final_data.head(10)
```

Out[69]:

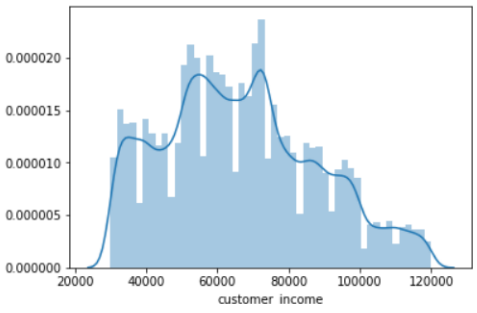
	difficulty	duration	offer_id	reward	web	email	mobile	social	offer_bogo	offer_discount	...	memberdays	gender_F	gender_M	gender_O	Age_group_45
0	0.5	0.571429	0	1.0	0	1	1	1	1	0	...	1394	0	1	0	0
1	0.5	0.571429	0	1.0	0	1	1	1	1	0	...	1394	0	1	0	0
2	0.5	0.571429	0	1.0	0	1	1	1	1	0	...	1394	0	1	0	0
3	0.5	0.571429	0	1.0	0	1	1	1	1	0	...	1394	0	1	0	0
4	0.5	0.571429	0	1.0	0	1	1	1	1	0	...	1394	0	1	0	0
5	0.0	0.142857	1	0.0	1	1	1	0	0	0	...	1394	0	1	0	0
6	0.0	0.142857	1	0.0	1	1	1	0	0	0	...	1394	0	1	0	0
7	1.0	1.000000	2	0.5	1	1	0	0	0	1	...	1394	0	1	0	0
8	1.0	1.000000	2	0.5	1	1	0	0	0	1	...	1394	0	1	0	0
9	1.0	1.000000	2	0.5	1	1	0	0	0	1	...	1394	0	1	0	0

10 rows × 27 columns

Data Exploration:

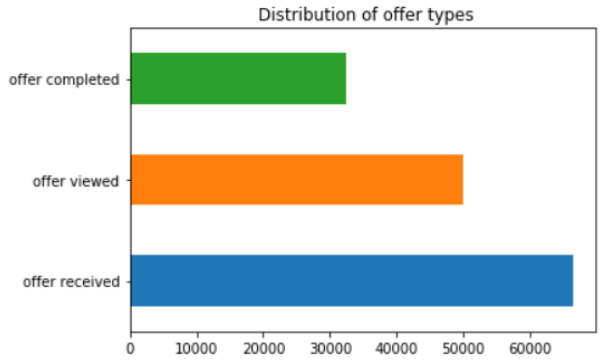
This will provide us a deep knowledge in the distribution of data based on various factors.

```
In [37]: sns.distplot(merged_df['customer_income'], bins=50, hist_kws={'alpha': 0.4});
```



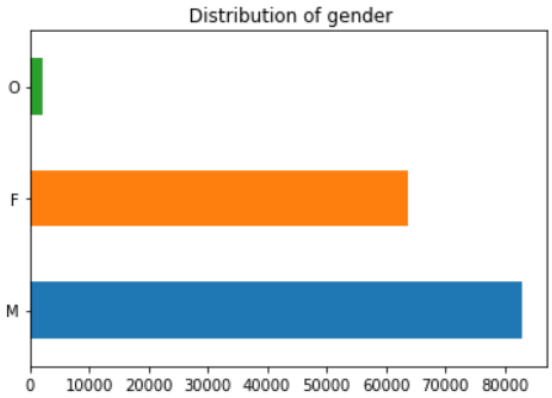
In [38]:
```
merged_df['event'].value_counts().plot.barh(title=' Distribution of offer types')
```

Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0x7f16c59ebd68>



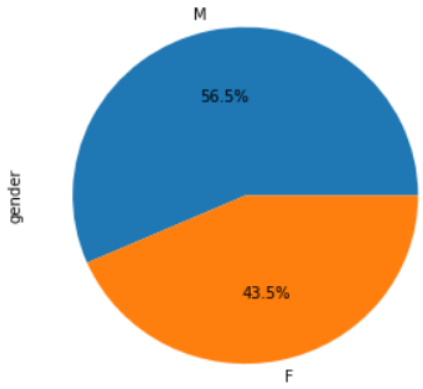
In [39]:
```
merged_df['gender'].value_counts().plot.barh(title=' Distribution of gender')
```

Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x7f16c5a6ecf8>



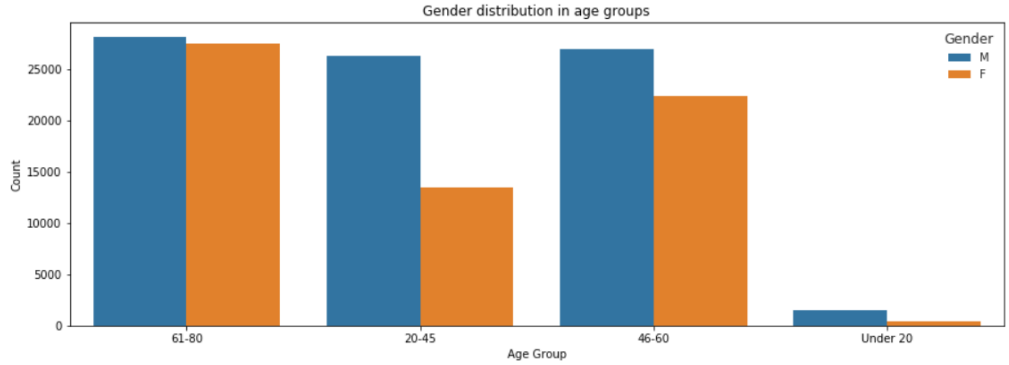
In [42]:
```
plot_gender.gender.value_counts().plot(kind='pie' , figsize=(5, 5), autopct='%1.1f%%')
```

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x7f16c5c68dd8>



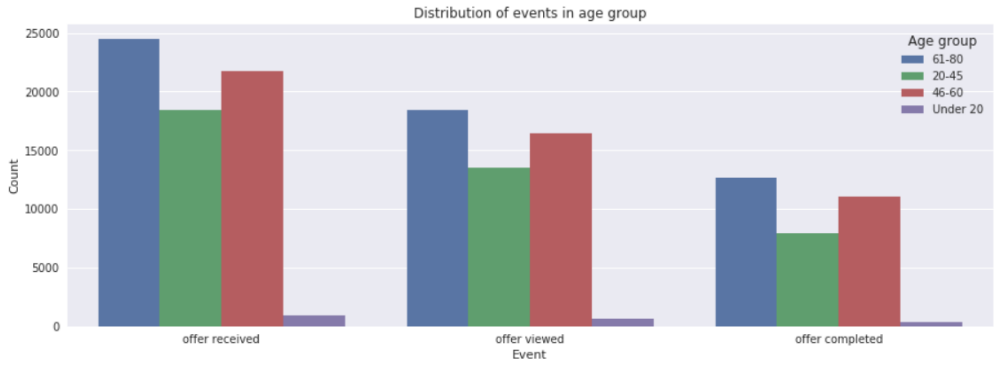
In [43]:
```
#plotting Gender distribution
plt.figure(figsize=(15, 5))
sns.countplot(x= "Age_group", hue= "gender", data=plot_gender)
sns.set(style="darkgrid")
plt.title('Gender distribution in age groups')
plt.ylabel('Count')
plt.xlabel('Age Group')
plt.legend(title='Gender')
```

Out[43]: <matplotlib.legend.Legend at 0x7f16c5bf3240>



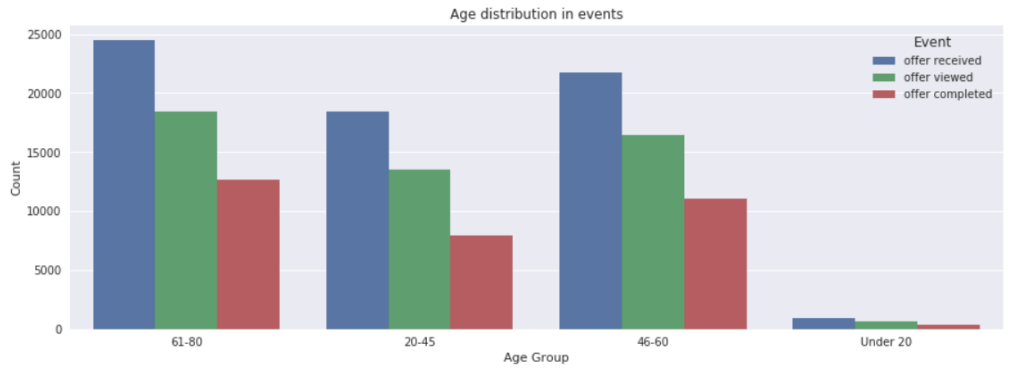
In [44]:
```
#Plotting evnts distribution in age group
plt.figure(figsize=(15, 5))
sns.countplot(x= "event", hue= "Age_group", data=plot_gender)
sns.set(style="darkgrid")
plt.title('Distribution of events in age group')
plt.ylabel('Count')
plt.xlabel('Event')
plt.legend(title='Age group')
```

Out[44]: <matplotlib.legend.Legend at 0x7f16c5c30c50>



In [45]:
```
plt.figure(figsize=(15, 5))
sns.countplot(x= "Age_group", hue= "event", data=plot_gender)
sns.set(style="darkgrid")
plt.title('Age distribution in events')
plt.ylabel('Count')
plt.xlabel('Age Group')
plt.legend(title='Event')
```

Out[45]: <matplotlib.legend.Legend at 0x7f16c5c303c8>



Building a Model:

At this stage, we will build a Machine Learning model based on the above understanding of the data sets.Since it falls under the supervised binary classification problem, we can try any of the models like RandomForestClassifier, DecisionTreeClassifier etc and determine the accuracy

Evauation:

At this stage, we will compare the models and evaluation metric to ensure sanity and determine the model which provides maximum accuracy in the prediction.