Project_Report_Vickie

October 23, 2020

1 Starbucks Capstone Project

1.0.1 Octorber 10,2020

2 1.1 Domain Background

- Starbucks, an American Coffee Company, which offers discount to mobile users who purchase items through the mobile app. As a retail company that sells products to its consumers, increasing sales would be a major concern for them, and as a businesses look forward to increasing sales and revenue every year, establishing means to help with this using technology would be at the top of their list.
- Using a simplified version of the real Starbucks app based on a simulator we seek to be able to determine demographic groups of consumers and also how they best respond to offers.

3 1.2 Problem Statement

- The problem this project proposes to solve is finding the most appropriate offer for each one of the customers, which means finding the offer that is more likely to lead the customer to buy Starbucks products. In the context of this project, an appropriate offer is that one where the customer sees the offer received and buys products under its influence,completing the offer lifecycle.
- If a customer does not see an offer, it is not an appropriate one. If he or she sees the offer but does not complete it, it is not appropriate as well, since it did not lead the consumer to buy products. Similarly, if the customer buys some products, completes an offer, and receives a reward before visualizing that offer, it is not considered effective because the customer was not under the influence of that offer when decided to make a purchase.

4 1.3 Datasets and Inputs

```
In [2]: import pandas as pd
        import numpy as np
        import math
        import json

        %matplotlib inline
```

```
# read in the json files
portfolio = pd.read_json('data/portfolio.json', orient='records', lines=True)
profile = pd.read_json('data/profile.json', orient='records', lines=True)
transcript = pd.read_json('data/transcript.json', orient='records', lines=True)
```

The data is contained in three 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)

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

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 [3]: portfolio
```

Out[3]:

	reward	channels	difficulty	duration	offer_type
0	10	[email, mobile, social]	10	7	bogo
1	10	[web, email, mobile, social]	10	5	bogo
2	0	[web, email, mobile]	0	4	informational
3	5	[web, email, mobile]	5	7	bogo
4	5	[web, email]	20	10	discount
5	3	[web, email, mobile, social]	7	7	discount
6	2	[web, email, mobile, social]	10	10	discount
7	0	[email, mobile, social]	0	3	informational
8	5	[web, email, mobile, social]	5	5	bogo
9	2	[web, email, mobile]	10	7	discount

	id
0	ae264e3637204a6fb9bb56bc8210ddfd
1	4d5c57ea9a6940dd891ad53e9dbe8da0
2	3f207df678b143eea3cee63160fa8bed
3	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	0b1e1539f2cc45b7b9fa7c272da2e1d7
5	2298d6c36e964ae4a3e7e9706d1fb8c2
6	fafdcd668e3743c1bb461111dcafc2a4
7	5a8bc65990b245e5a138643cd4eb9837
8	f19421c1d4aa40978ebb69ca19b0e20d
9	2906b810c7d4411798c6938adc9daaa5

In [4]: profile

Out[4]:

	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
...	...	...	...	...	...
16995	F	45	6d5f3a774f3d4714ab0c092238f3a1d7	20180604	54000.0
16996	M	61	2cb4f97358b841b9a9773a7aa05a9d77	20180713	72000.0
16997	M	49	01d26f638c274aa0b965d24cefe3183f	20170126	73000.0
16998	F	83	9dc1421481194dcd9400aec7c9ae6366	20160307	50000.0
16999	F	62	e4052622e5ba45a8b96b59aba68cf068	20170722	82000.0

[17000 rows x 5 columns]

In [5]: transcript

Out[5]:

	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
...	...	...	...	...
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

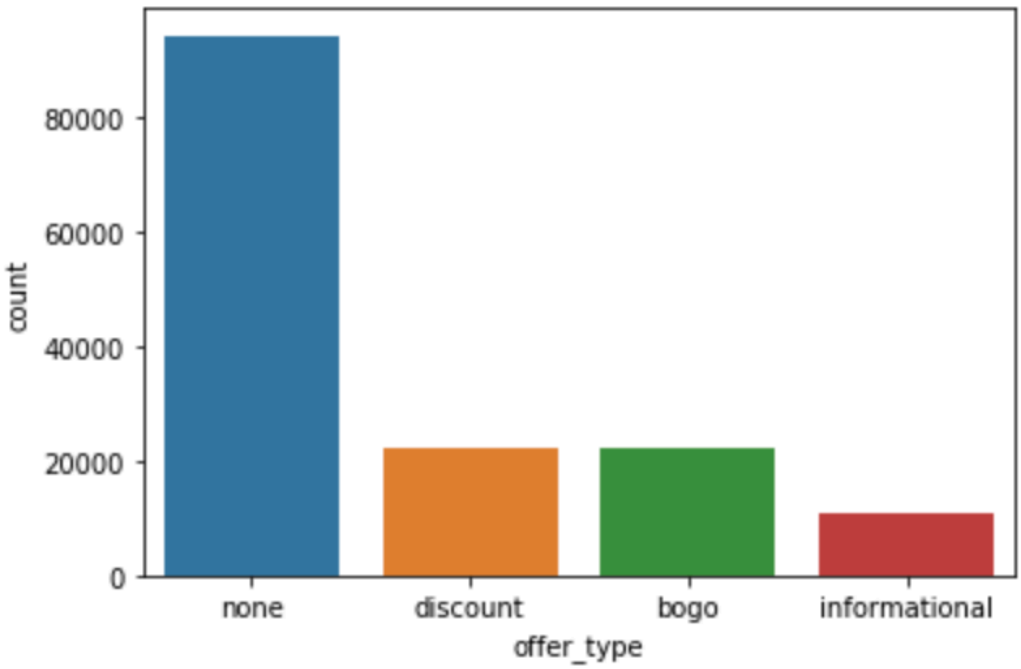
```
2       {'offer id': '2906b810c7d4411798c6938adc9daaa5'}     0
3       {'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}     0
4       {'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}     0
...                                                    ...   ...
306529                   {'amount': 1.5899999999999999}   714
306530                                {'amount': 9.53}   714
306531                                {'amount': 3.61}   714
306532                   {'amount': 3.5300000000000002}   714
306533                                {'amount': 4.05}   714

[306534 rows x 4 columns]
```

4.0.1 Data Inputs

Although there are only three types of offer(bogo, discount, informational), acutually some people complete the transaction without refering to the offers. In order to exclude this condition, we have to consider the "none" type of offer. The histogram of offer type shows the amount of each certain offer.

In [8]:
```
from IPython.display import Image
Image('offer_type.png',width=600,height=400)
```

Out[8]:



In [19]:
```
Image('input2.png')
```

Out[19]:

```
data_input.head()
```

	person	offer_completed	id	gender	age	became_member_on	income
0	78afa995795e4d85b5d9ceeca43f5fef	1	9b98b8c7a33c4b65b9aebfe6a799e6d9	1	75	20170509	100000.0
1	e2127556f4f64592b11af22de27a7932	1	9b98b8c7a33c4b65b9aebfe6a799e6d9	-1	68	20180426	70000.0
3	389bc3fa690240e798340f5a15918d5c	1	9b98b8c7a33c4b65b9aebfe6a799e6d9	-1	65	20180209	53000.0
4	d058f73bf8674a26a95227db098147b1	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	1	56	20180428	88000.0
6	ebe7ef46ea6f4963a7dd49f501b26779	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	-1	59	20150121	41000.0

In [18]: Image('input1.png')

Out[18]:

In [63]:
```
data_input.head()
```

Out[63]:

id	gender	age	became_member_on	income	reward	difficulty	duration	offer_type	web	email	mobile	social	member_duration
9c7a33c4b65b9aebfe6a799e6d9	1	75	20170509	100000.0	5.0	5.0	7.0	bogo	1.0	1.0	1.0	0.0	1259
9c7a33c4b65b9aebfe6a799e6d9	-1	68	20180426	70000.0	5.0	5.0	7.0	bogo	1.0	1.0	1.0	0.0	907
9c7a33c4b65b9aebfe6a799e6d9	-1	65	20180209	53000.0	5.0	5.0	7.0	bogo	1.0	1.0	1.0	0.0	983
9c7a33c4b65b9aebfe6a799e6d9	1	56	20180428	88000.0	5.0	5.0	7.0	bogo	1.0	1.0	1.0	0.0	905
9c7a33c4b65b9aebfe6a799e6d9	-1	59	20150121	41000.0	5.0	5.0	7.0	bogo	1.0	1.0	1.0	0.0	2098

5 1.4 Solution Statement

A solution I propose would be to apply a Classification Machine Learning model after thorough analysis on demography data and transactions to determine if a user would respond to a particular offer or not.

With the information that there are 4 offers: * Bogo * Informational * Discount * None

This makes our model a multi-label classification model which would be useful in determine which of the offers (classes) listed above that a potential consumer would respond best to. With the knowledge that some consumers do not purchase an item with an offer in mind, they do not need to recieve any offer, is the reason why None is also listed as a class.

6 1.5 Benchmark Model

- In the terminology of machine learning, classification is considered an instance of supervised learning, i.e., learning where a training set of correctly identified observations is available.
- One of the basic algorithms of classification is logistic regression. Therefore, I would like to use logistic regression as my benchmark model.

Logistic Regression

- Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression. (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names. Analogous models with a different sigmoid function instead of the logistic function can also be used, such as the probit model; the defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable this generalizes the odds ratio.[1]

7 1.6 Evaluation Metrics

```
In [20]: Image('metrics.png')
```

Out[20]:

Table 1. Confusion Matrix for Binary Classification and the Corresponding Array Representation used in this Study

	Actual Positive Class	**Actual Negative Class**
Predicted Positive Class	True positive (*tp*)	False negative (*fn*)
Predicted Negative Class	False positive (*fp*)	True negative (*tn*)

7.0.1 1.6.1 Accuracy

In general, the accuracy metric measures the ratio of correct predictions over the total number of instances evaluated. * accuracy = (tp + tn) / (tp + fp + fn + tn)

7.0.2 1.6.2 F-Measure (FM)

Precision is used to measure the positive patterns that are correctly predicted from the total predicted patterns in a positive class. * precision = tp / (tp + fp)

Recall is used to measure the fraction of positive patterns that are correctly classified. * recall = tp / (tp + tn)

F-Measure (FM): This metric represents the harmonic mean between recall and precision values * fm = 2 * precision * recall / (precision + recall)

8 1.7 Project Design

8.0.1 1.7.1 Data Loading&cleaning

I would proceed to merge datasets together, transcript.json with both the porfolio.json and profile.json, as this all hold data that describe entities describe in the transcript.

I need to consider that some segment gatherings will make buys regardless of whether they don't get an offer. From a business point of view, if a client is going to make a 10 dollar buy without an offer at any rate, you wouldn't have any desire to send a purchase 10 dollars get 2 dollars off offer. I need to attempt to survey what a specific segment gathering will purchase when not accepting any offers. Therefore, there is a new type of "offer", named "none". This means that all the property values of this new type of offer are "0.0".

The following procedure is performed to complete the appropriate data preprocessing:

1. Shuffle the data (randomize entries)
2. clean the transcipt by person and offer_id. To clean the transcript, I need to group the dataframe by "id" and "person". If one person had finished the offer, the value of "offer_completed" would be '1', else '0'. Also, I need to record the amount of each transaction for each offer completion.

```
In [10]: pd.read_csv('cleaned_transcipt_byperson&offer2.csv',index_col=0)
```

```
Out[10]:                                  person  offer_completed  \
         0       78afa995795e4d85b5d9ceeca43f5fef                1
         1       78afa995795e4d85b5d9ceeca43f5fef                0
         2       78afa995795e4d85b5d9ceeca43f5fef                1
         3       78afa995795e4d85b5d9ceeca43f5fef                1
         4       78afa995795e4d85b5d9ceeca43f5fef                1
         ...                                  ...              ...
         171619  ebae5093b436466c9fbd097cb7434023                1
         171620  ebae5093b436466c9fbd097cb7434023                1
         171621  912b9f623b9e4b4eb99b6dc919f09a93                0
         171622  3045af4e98794a04a5542d3eac939b1f                0
         171623  da7a7c0dcfcb41a8acc7864a53cf60fb                1

                                              id  amount
         0       9b98b8c7a33c4b65b9aebfe6a799e6d9   19.89
         1       5a8bc65990b245e5a138643cd4eb9837     NaN
         2       ae264e3637204a6fb9bb56bc8210ddfd   21.72
         3       f19421c1d4aa40978ebb69ca19b0e20d   21.72
         4                                  none   17.78
         ...                                 ...     ...
         171619  fafdcd668e3743c1bb461111dcafc2a4   12.64
         171620                             none    2.57
         171621  4d5c57ea9a6940dd891ad53e9dbe8da0     NaN
         171622  4d5c57ea9a6940dd891ad53e9dbe8da0     NaN
         171623                             none    0.35

         [171624 rows x 4 columns]
```

3. merge the dataframe and "portfolio" by id
4. merge the dataframe and "profile" by person

The ultimate cleaned dataframe should be consisted of following columns. * columns = ['person', 'offer_completed', 'id', 'amount', 'gender', 'age', 'became_member_on', 'income', 'reward', 'difficulty', 'duration', 'offer_type', 'web', 'email', 'mobile', 'social', 'member_duration']

pay attention to "none" type offer!!

```
In [9]: Image('none_type.png')
```

Out[9]:

```
In [372]: data_input[data_input.offer_type=='none']
```

Out[372]:

	person	offer_completed	id	amount	gender	age	became_member_on	income	reward	difficulty	duration	offer_type
59181	38f02eb17af04d55b4cff9dacba7e54c	1	none	14.06	-1	65	20151217	80000.0	0.0	0.0	0.0	none
96747	a6af69d4f60e49adb49022aba7a6392b	1	none	12.51	-1	51	20170308	85000.0	0.0	0.0	0.0	none
69091	fadd9b4164c84122bbd2bdde73aba541	1	none	1.08	-1	48	20180602	47000.0	0.0	0.0	0.0	none
64309	0ac5ffb0f1624297a453494b845e1a91	1	none	3.87	-1	57	20141104	46000.0	0.0	0.0	0.0	none
97694	a93cb9e04f5a4ffa8beeb6e6a51333ca	1	none	6.74	-1	40	20170210	41000.0	0.0	0.0	0.0	none
...	...	...	...	...	...	...	...	...	...	...	...	...
30573	6e2848db7c654376ad1da5616bba5395	1	none	6.28	-1	19	20170402	39000.0	0.0	0.0	0.0	none
06628	8524d450673b4c24869b6c94380006de	1	none	3.18	-1	70	20180604	75000.0	0.0	0.0	0.0	none
76569	a482640827114b1081c8c81fd1cd428b	1	none	10.60	1	57	20171104	71000.0	0.0	0.0	0.0	none
33605	04698a1182a24807ad4c36f12b67c5ef	1	none	19.82	1	73	20151128	74000.0	0.0	0.0	0.0	none
64895	46589de520eb4087a4715abf072ed490	1	none	9.57	-1	84	20171125	37000.0	0.0	0.0	0.0	none

l376 rows × 17 columns

8.0.2 1.7.2 Data Analysis & Visualizations

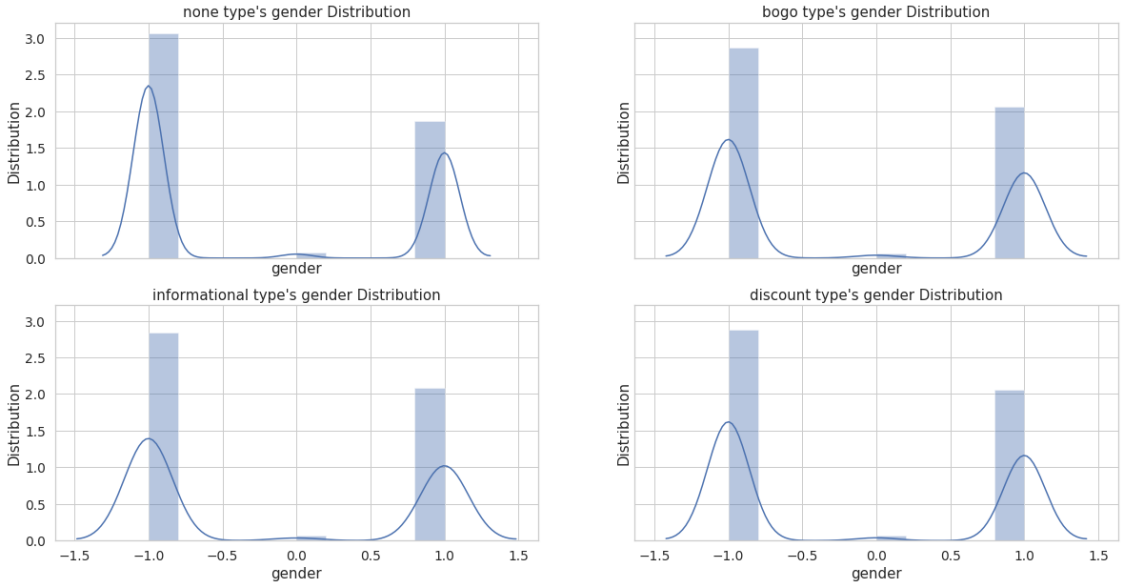
```
In [12]: data_input = pd.read_csv('data_input.csv',index_col=0)
```

```
In [16]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.set(style = 'whitegrid', font_scale = 1.25)
         palette = sns.color_palette()
```

```
In [34]: tags = ['gender', 'age', 'member_duration', 'income']
         for tag in tags:
             print(tag)
             fig, ax = plt.subplots(figsize=(20,10), nrows = 2, ncols = 2, sharex = True, share
             plt.sca(ax[0,0])
             sns.distplot(data_input[data_input['offer_type'] == 'none'][tag], bins = 10, color
             plt.title('none type\'s %s Distribution' % tag)
             plt.xlabel(tag)
             plt.ylabel('Distribution')

             plt.sca(ax[0,1])
             sns.distplot(data_input[data_input['offer_type'] == 'bogo'][tag], bins = 10, color
```

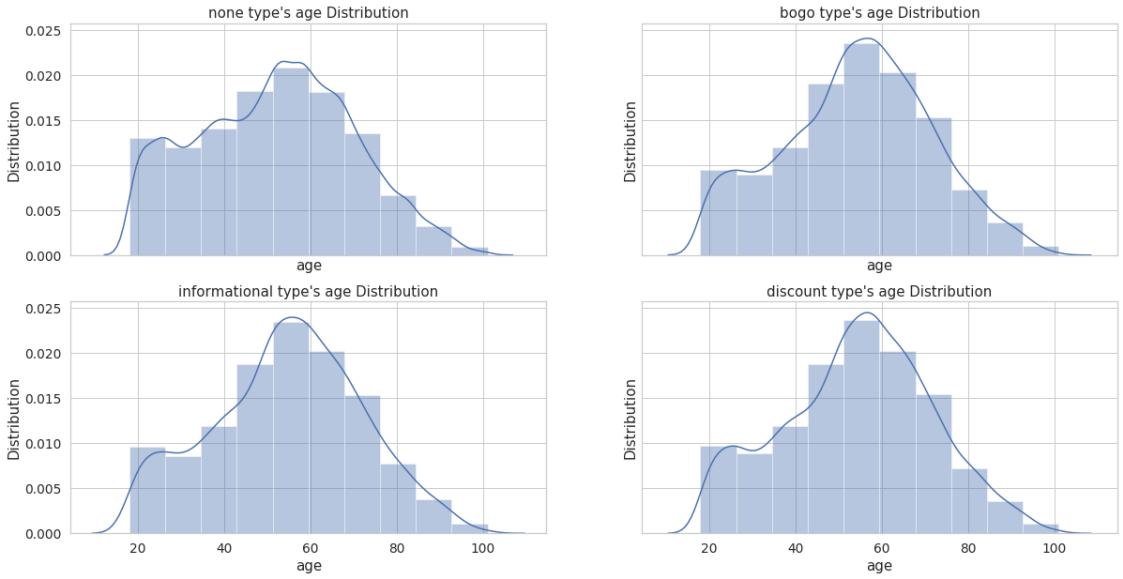
```
            plt.title('bogo type\'s %s Distribution' %tag)
            plt.xlabel(tag)
            plt.ylabel('Distribution')

            plt.sca(ax[1,0])
            sns.distplot(data_input[data_input['offer_type'] == 'informational'][tag], bins =
            plt.title('informational type\'s %s Distribution'%tag)
            plt.xlabel(tag)
            plt.ylabel('Distribution')

            plt.sca(ax[1,1])
            sns.distplot(data_input[data_input['offer_type'] == 'discount'][tag], bins = 10, c
            plt.title('discount type\'s %s Distribution'%tag)
            plt.xlabel(tag)
            plt.ylabel('Distribution')

            plt.show()
            plt.clf()
```

gender



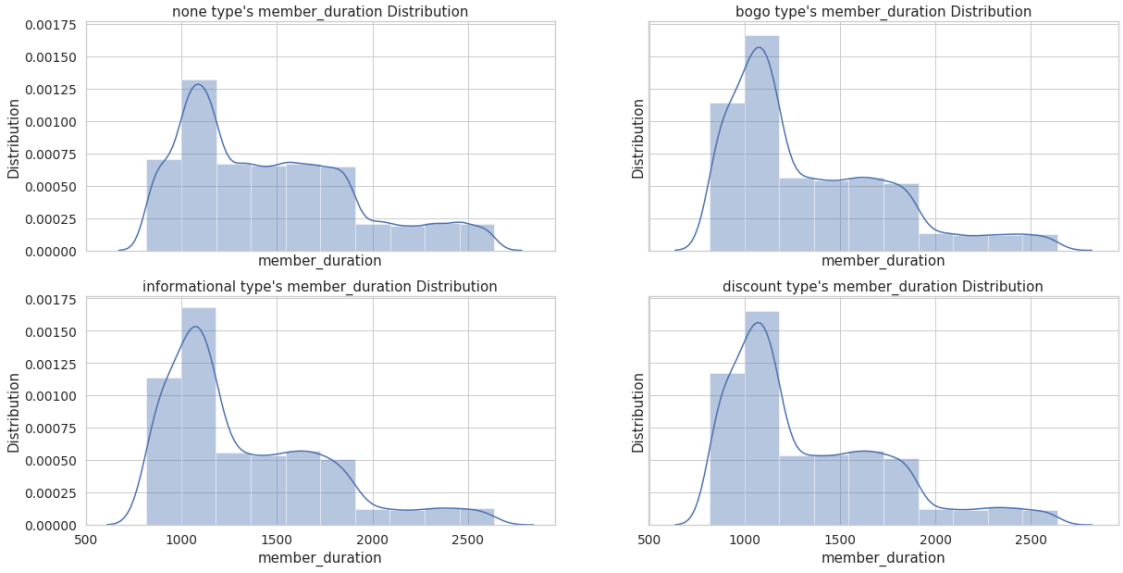
age

```
<Figure size 432x288 with 0 Axes>
```



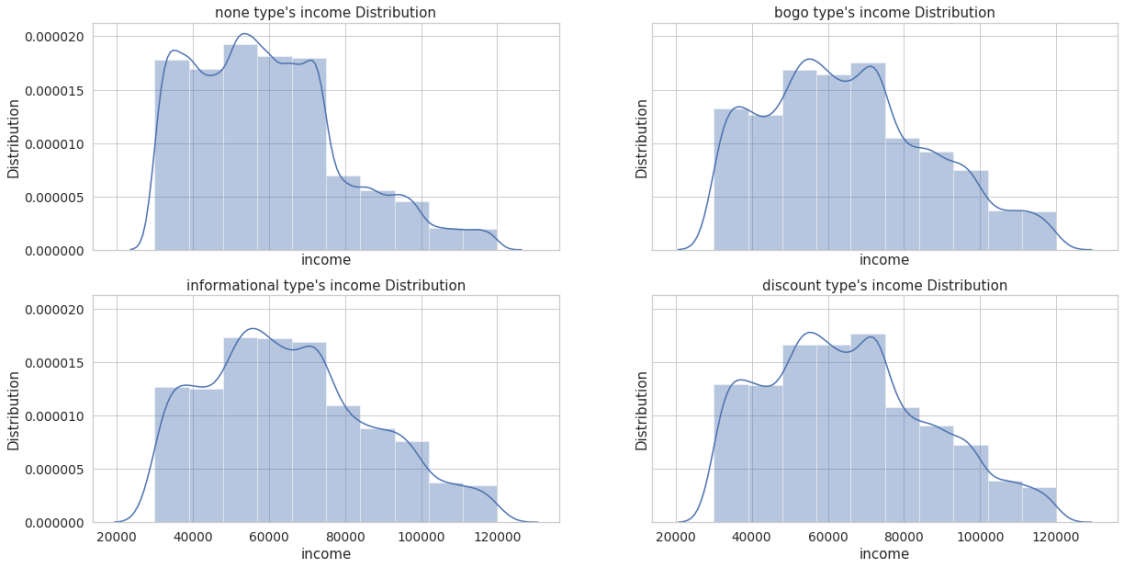

```
member_duration
```

```
<Figure size 432x288 with 0 Axes>
```




```
income
```

```
<Figure size 432x288 with 0 Axes>
```




```
<Figure size 432x288 with 0 Axes>
```

8.0.3 1.7.3 Model Selection

There are 5 machine learning models for classification that will be examined thoroughly. These are:

- Logistic Regression
- Decision Tree
- Adaptive Boosting
- Random Forest
- XGBoost

For each of the model, the cross validation technique of 10-fold is used to calculate the accuracy of each model. These values are going to be shown in a DataFrame model_df. The dataframe is sorted based on the model test accuracy (the highest at the top).

```
In [1]: from IPython.display import Image
        Image('model_result.png')
```

```
Out[1]:
```

```
In [160]: model_result.sort_values('test_accuracy',ascending=False)
```

Out[160]:

	model_parameter	accuracy	f1-score	precision	test_accuracy	test_f1-score	test_precision
XGBClassifier	{'max_depth': 2, 'eta': 0.2, 'gamma': 4, 'min_...	0.840487	0.909103	0.974573	0.837059	0.906848	0.849091
AdaBoostClassifier	{'algorithm': 'SAMME.R', 'base_estimator': Non...	0.835318	0.906195	0.848861	0.832193	0.904195	0.845616
RandomForestClassifier	{'bootstrap': True, 'ccp_alpha': 0.0, 'class_w...	0.829819	0.900171	0.865451	0.830668	0.900476	0.864306
LogisticRegression	{'C': 1.0, 'class_weight': None, 'dual': False...	0.818491	0.900187	0.818491	0.815107	0.898137	0.815107
DecisionTreeClassifier	{'ccp_alpha': 0.0, 'class_weight': None, 'crit...	0.798713	0.879556	0.861909	0.804626	0.88315	0.861614

8.0.4 1.7.4 Conclusion

XGBClassifier shows that the three following features to be the most important:

1. offer_duation
2. amount
3. channel_social

Let's take a look at each of these features. Offer duration might be significant since it specifies the number of days the duration of the offer lasts. The longer the offer lasts, the higher the chance for a particular customer to fulfill its requirement. How about money spent(amount)? Unlike income, the money spent by a user typically reflects a closer lifestyle of that person. The more often you buy, the higher the chance that you will buy enough to meet the offer conditions. This is more informational than the income information, since it only gives an idea of how much money they make, not how much money they are willing to spend.

Finally, the social channel feature. The fact that this comes in the top three features signifies how social channel is the one that people pay more attention to (obviously does not apply for everyone). This can be used as a reasoning for Starbucks to concentrate more on advertising/sending their offers through these social channels, instead of through web for example. One thing to note though, the email channel is not included in the training data since it contains little to no variation between datasets. Therefore, it might interesting to do a little further experiment to see if whether email is a more effective and efficient method of sending Starbucks offers.

Among different models, XGB Classifier comes out on top. The training and validation scores converge to a similar value, just right so that this model can generalize well to unseen data. Due to its high accuracy score and the ideal learning curve, XGB Classifier is chosen as the machine learning model for this project.

```
In [ ]:
```