Starbuck's Capstone Challenge

Definition

Project Overview

It is always a marketing topic to study our customers, especially in fast moving consumer goods domain where deals taken place at a very high frequency. So there are a bunch of data to be deeper mining to get useful information for further analysis.
This project created some dummy customer data of Starbucks, along with their actions data to different type of offers. It is very interesting to see how customer will act to different type of offers.

Problem Statement

As per the transcript dataset provided, there are four actions recorded in the dataset:

- offer received
- offer viewed
- transaction
- offer completed.

So there are potential several combinations among these actions, but I simply define these combination as follow three categories:

1. positive feedback on offer – customer will order after they know the offer, which including:
   a) offer received > offer viewed > transaction > * (for informational offer there is no offer completed)
2. negative feedback on offer – customer won't order even they know the offer, which including:
   a) offer received > offer viewed
3. neutral feedback on offer – customer will order regardless of awareness of offer, or they don't do anything after receiving offers, which including:
   a) offer received > transaction > * (for informational offer there is no offer completed)
   b) offer received

So in this regards I will build a classification model to predict whether a customer will be positive, negative or neutral feedback to a given offer.

Metrics

Since this is a classification problem, so it is very common to use accuracy as a metric. But it's

not good for imbalanced dataset, in which case F1 score is much better as a metric.
As accuracy is much easier to interpret and F1 score provide a better assessment of model performance, I will take both as metrics.

Analysis

Data Exploration

There are 3 datasets provided:

1. Portfolio

Overall, it is a very small dataset that only has 9 entries. There 4 BOGO offer, 4 discount offer and 2 informational offer with different channels, difficulty, duration and reward. Including following features:

- Channels – list-like object
- Difficulty – int64
- Duration – int64
- Id – object
- Offer_type – object
- Reward – int64

2. Profile

This dataset is all customer information with 17000 distinct entries, but there are 2175 entries are missing gender and income. There are also some 118 age customers looks a bit strange. Including following features:

- age – int64
- became_member_on – int64
- gender – object
- Id – object
- income – float64

3. Transcript

This is the trickiest dataset although there are only four columns with more than 300,000 rows. Only time is numeric value. Event and person are strings while value column is a dictionary structure. There are no missing values. Including following features:

- event – object
- person – object
- time – int64
- value – object

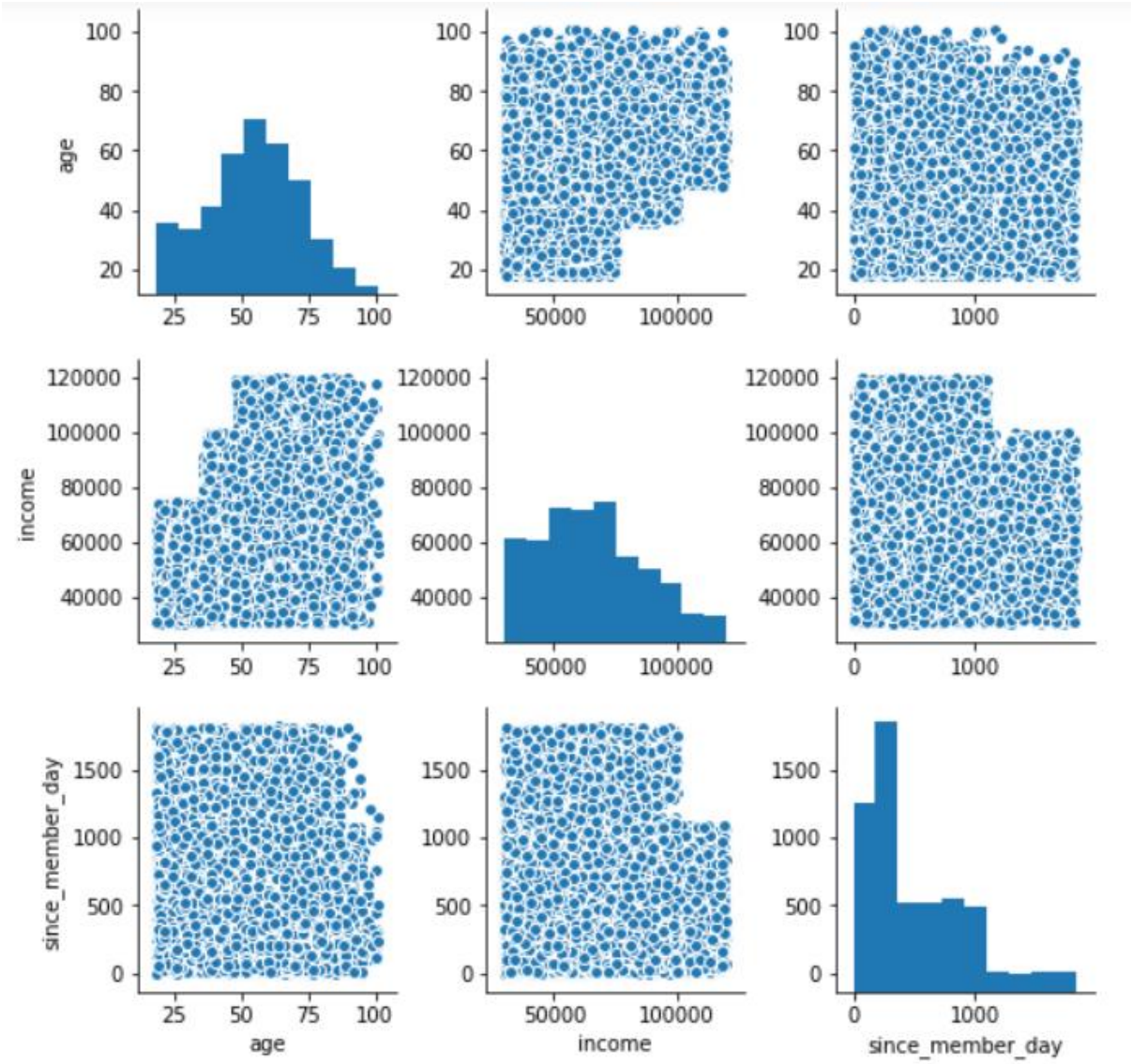
Data Visualization

At the very beginning only the profile dataset can have some visualization to analysis, for other dataset it will be better to explore visualization after preprocessing. So here I only focus on profile dataset. I performed two step on the dataset:

- Delete NaN value rows - as mentioned above there are 2175 missing values respectively

in column gender and income. These missing values appeared in pairs in these two columns and the corresponding age value are all 118, which doesn't make sense, so I delete these rows.

- Create a new feature since_number_day - for better analysis I create a new column - since_member_day as per became_member_on, which means the tenure days of a customer. In addition, I also set the most recent customer tenure as 0.



The findings are:

- The age and income are like normal distribution.
- The since_member_day column elaborates that newer customers are more than regular customers.
- Age doesn't have anything to do with customer tenure, but it does have relation to income – generally the elder the customer, the higher income they have.

Methodology

Data Preprocessing

1. Portfolio

For this dataset, the only preprocessing is to do one hot encode on the channels

2. Profile

The following steps are made to preprocess:

- delete all missing value rows
- Create a new feature since_number_day to better understand customer tenure

3. Transcript

This dataset is the most important and trickiest part in data preprocessing, which can be described as follow:

- Split the value feature to column by key in its dictionary-like object and save to cache
- Map the transactions to offer
  - ✧ For discount and BOGO offers, first get the pairs of offer completed and transaction with the same time taken place, then pull the offer id in offer completed and labeled offer id on transaction.
  - ✧ For informational offer, I marked all unlabeled transaction entries followed by informational offer received and informational offer viewed within validity of corresponding informational offer, and labeled these transaction with the informational offer id.
  - ✧ Save the data the cache
- Label the customer feedback result
  - ✧ As the transcript dataset now have all offer id mapped to their related transactions, the rest of unlabeled transaction are regarded as non-offer related. I am interested in how customer will respond to different type of offers, so as per statement in Problem Statement, then I can label the results of each customer whether they are positive, negative or neutral attitude to the offers.
  - ✧ First, I prepare a person – offer id pool, in which the distinct pairs of person and offer id are selected. This means all combination of person and offer id are considered in the transcript dataset.
  - ✧ For positive labels, I got the intersection set of offer viewed and transactions with same offer id. I labeled these entries as positive because they just follow the path: from offer received > offer viewed > transaction > offer completed, as what we expected.
  - ✧ For negative labels, I got the offer id rows only in offer viewed without appearing in transaction, since there is no transaction happened after offer viewed. So I labeled these entries as negative.
  - ✧ For neutral labels, the rest of combination in person – offer id pairs neither belongs to positive nor negative are labeled as neutral.
  - ✧ So the person – offer id dataset structure:
    - person (customer id)
    - offer id (offer id in portfolio dataset)
    - results (the label of positive, negative or neutral)

4. Merge to data to be trained

The steps are as follow:

- Merge profile data to labeled person – offer id on person
- Merge portfolio data to labeled person – offer id on offer id
- Perform one-hot encode on channel, gender, offer_type

5. Create new features

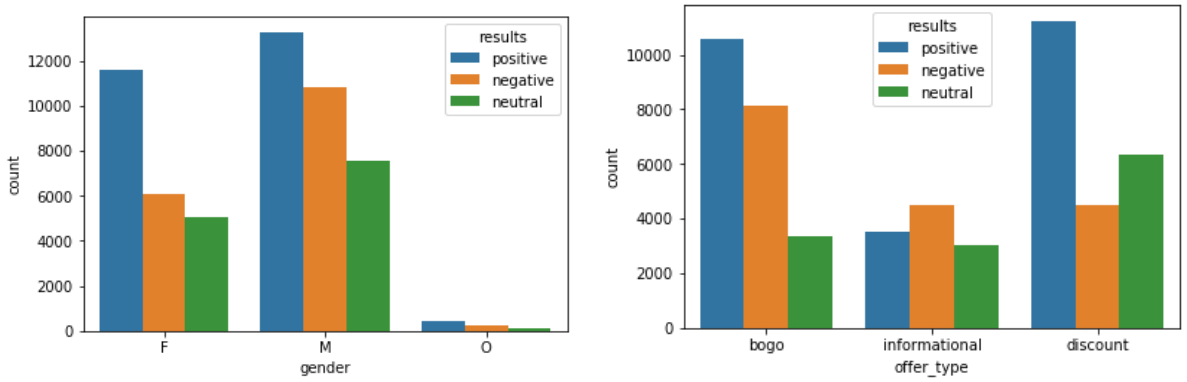
In addition to the features provided in given dataset, I am also interested in: How many transactions a customer has made without using an offer? How much money did they spent totally without using an offer? So I create below two features:

- no_offer_num, which means the total number of transactions that doesn't match to an offer from a customer
- no_offer_amount, which means the total amount of transactions that doesn't match to an offer from a customer

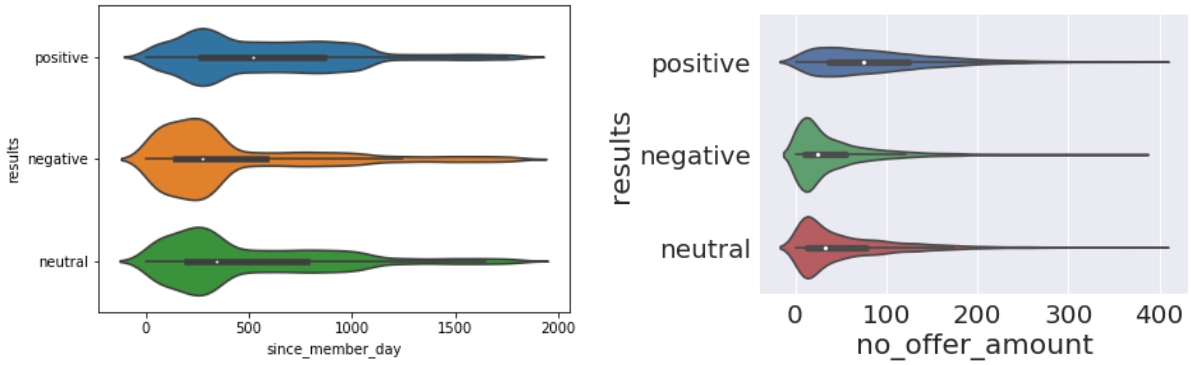
6. Further data visualization

We now can use the merged dataset with new features to do some further exploration.

- Discrete variables: Gender and offer type vs results

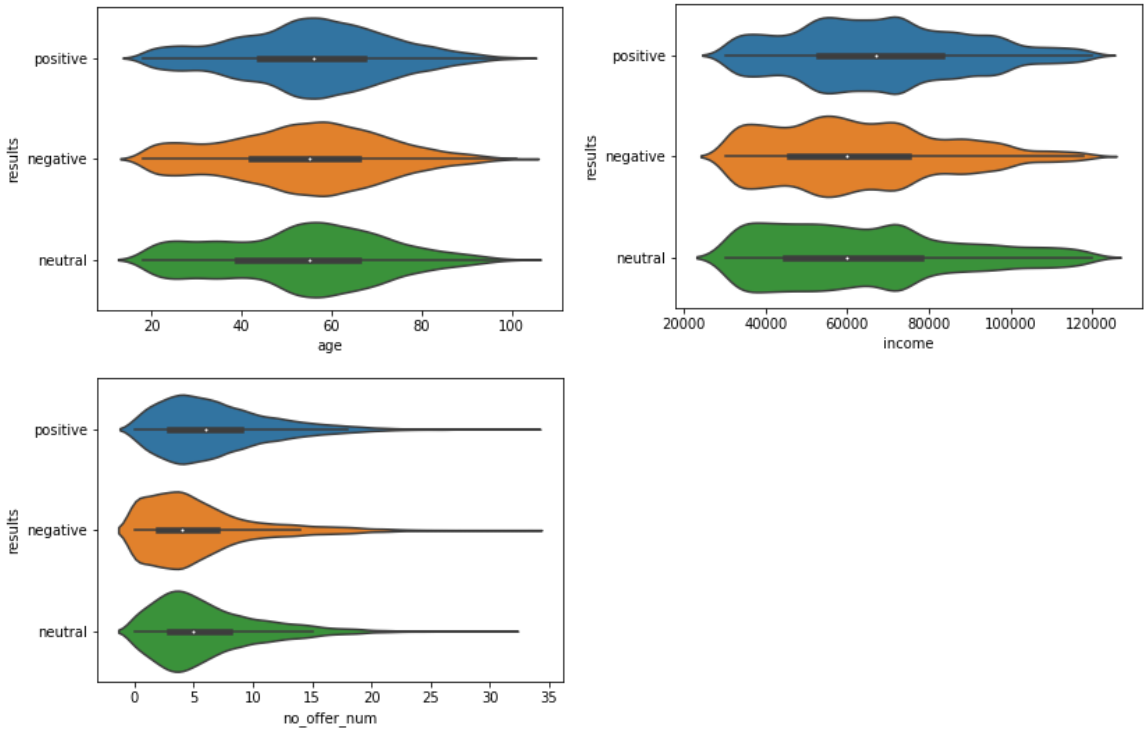


  - ✧ Offers are seemed to be more attractive to female than male as female has more percentage giving positive results.
  - ✧ Discount offers are the most attractive, BOGO are more attractive than informational offers.
- Since_member_day and no_offer_amount vs results



- ✧ It looks like regular customers give more positive feedback on offers, that mean customer tenure do have influence on the results.
- ✧ No_offer_amount is another feature that tells us regardless of offer, the more money customer spent, the more likelihood they will use an offer.

- Age, income and no_offer_num vs results



These three factors have far less impact on results compared with since_member_day and no_offer_amount. They are not significant factors.

Implementation

The dataset to be fed into model contains following features:

- 'age', 'income', 'difficulty', 'duration', 'reward'. These features are just from raw dataset
- 'since_member_day', 'no_offer_num', 'no_offer_amount'. These features are newly created.
- 'email', 'mobile', 'social', 'web', 'F', 'M' ,'O', 'bogo', 'discount', 'informational'. These features are generated from one-hot encoding from channel, gender and offer type.

Since this is a classification problem, so the target variable here is results (positive, negative and neutral). This is a multiclassification. At first I create a benchmark so as to do some comparison. The benchmark picked is just a LogisticRegression, after training the dataset with test set = 0.25, the accuracy is 0.58 with f1 score at 0.56.

Refinement

I tried the following step to refine the model performance:

- Try different algorithms other than LogisticRegression such as KNeighbors, SVM,

RandomForeset also followed by using standardization and normalization

- Using Grid Search to find best hyper-parameters
- Using cross validation to do final performance evaluation

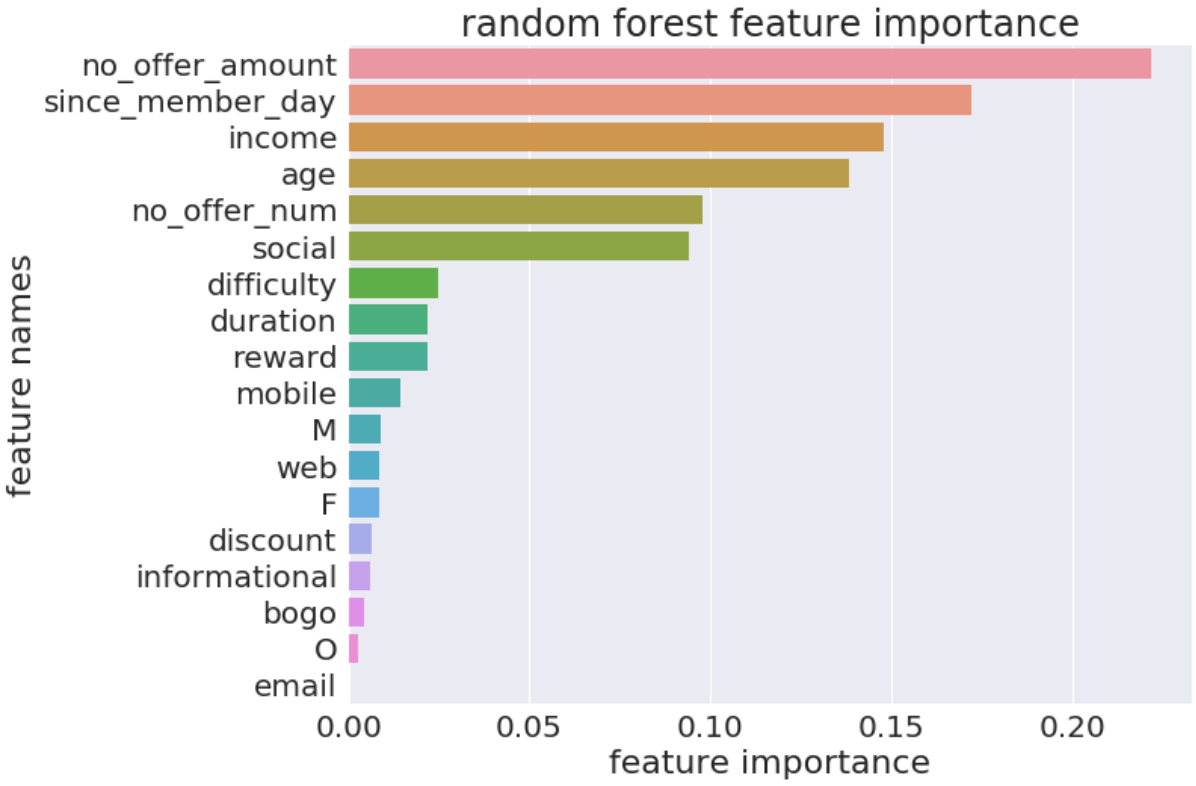
Results

Model Evaluation and Validation

The final model performance are listed as follow:

	Precision	Recall	F1-score	Accuracy
LogisticRegression	0.58	0.58	0.56	0.58
Kneighbors	0.62	0.62	0.62	0.62
SVM	0.63	0.62	0.61	0.62
RandomForest	0.64	0.64	0.64	0.64
RandomForest-Tuned	0.67	0.67	0.67	0.67

We can see in above table that all these algorithms have similar results, in which RandomForest has slightly better performance over the others. In addition, in order to better interpret the model, RandomForest is picked for further optimization.
After across validation of tuned RandomForest, the final accuracy and F1 score are reaching at 0.67 with 0.0 standard deviation, which means the model now should be stable enough.
Furthermore, below is a chart that displays the feature importance indicating which features are more important over others.



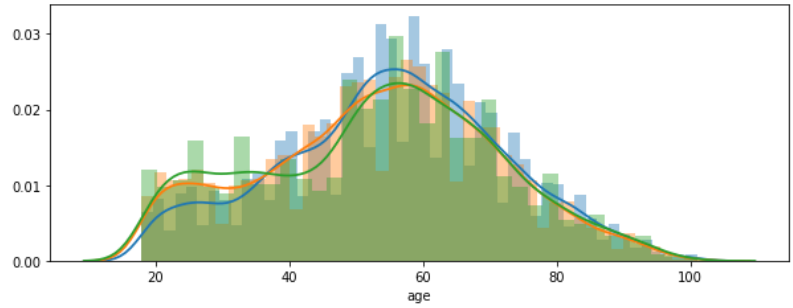
Justification

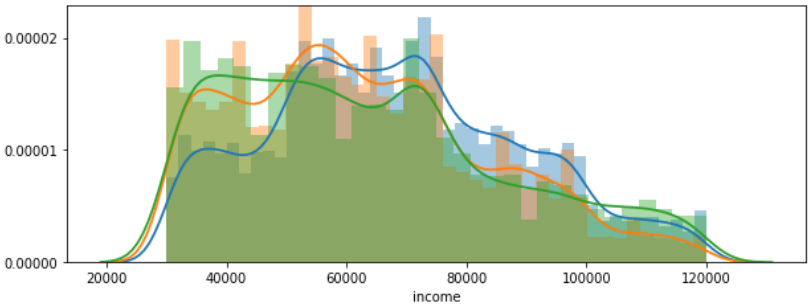
As RandomForest is a tree based algorithm, so scaling of the variables doesn't matter. In this dataset, as per feature importance, no_offer_amount, since_member_day, income and age are the 4 most important features influencing the target variable. And there are huge gaps among their scale. If no normalization performed on dataset, 'distance-related' algorithms like SVM would perform much worse results. So in this regards, there's no needs to worry about the data scale when using RandomForeset.
Besides, As an ensemble machine learning algorithm, RandomForest use random subspace method and bagging to prevent overfitting.

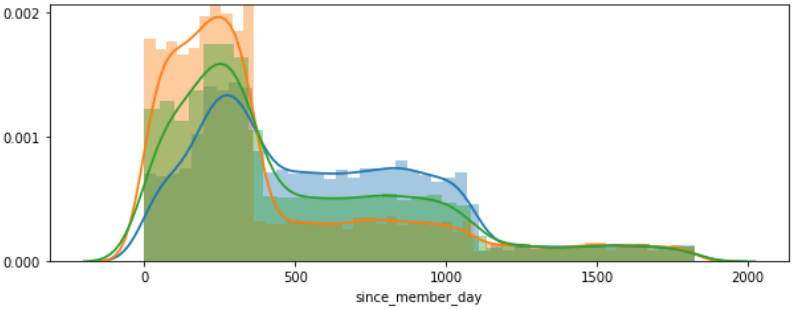
Conclusion

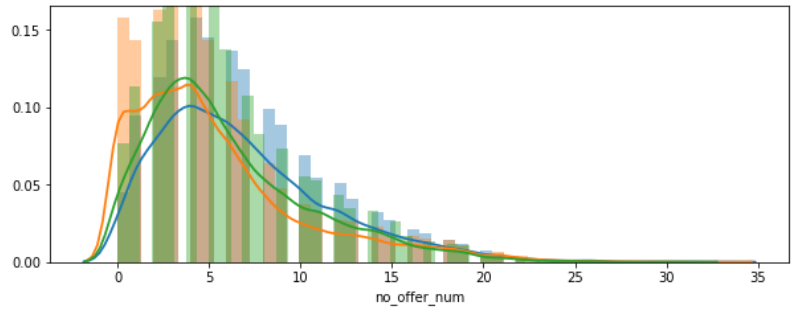
Reflection

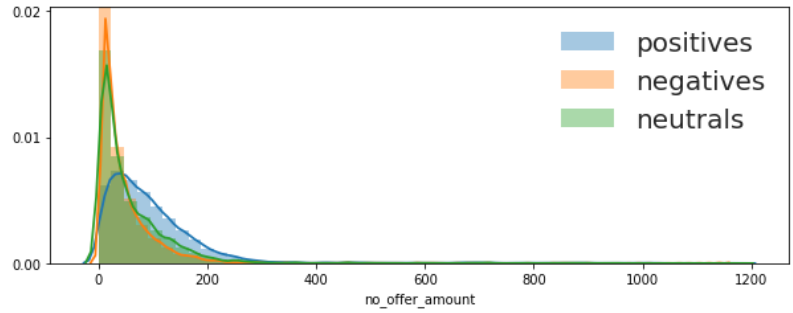
We can also have another view of the visualization on the distribution on different feedback results.


0.03
0.02
0.01
0.00
20
40
60
80
100
age


0.00002
0.00001
0.00000
20000
40000
60000
80000
100000
120000
income


0.002
0.001
0.000
0
500
1000
1500
2000
since_member_day


0.15
0.10
0.05
0.00
0
5
10
15
20
25
30
35
no_offer_num


0.02
0.01
0.00
0
200
400
600
800
1000
1200
no_offer_amount
positives
negatives
neutrals

I have built a model to predict whether a customer will respond to a particular offer. As per visualizations displayed above, we already know that female, regular customers are the most active group to give positive feedbacks on an offer. As they are the most active customers, so they usually make more transaction than those less active ones even if when they have no offers available, which is reflected in the feature no_offer_amount.

From the offer type perspective of view, discount offers are more attractive than BOGO. This perhaps due to not everyone need extra copy of coffee drink, but everyone can get the discount benefit. As informational offer do not have financial benefit, so they are the least attractive.

In addition, it is hard to define what is neutral feedback. They probably do not care about offers, or they do not want to receive offers at all? As there is no more information inherit from the dataset, its difficult to get further deeper insights in this point.

Improvement

In data preprocessing, as there will be the case that multiple order completed at same time only with one transaction, so I only map the default offer id in this situation, this can be done in detail to be more precise.
There is more information can be extract from the transcript dataset, e.g. how many times/percentage the customer viewed offer, completed that offer. And to perform their time analysis on offer viewed time, offer completed time should also somehow benefits the model. Last but not least, if we have more data about our customer, this should definitely help to improved the model performance.