Bassel Ahmed

Nov 4th, 2020.

Capstone Project Definition

Project Overview

The project contains simulated data that mimics customer behavior on the Starbucks rewards mobile app. The goal of the project is to analyze this "historical" data in order to implement an algorithm that finds the most suiting offer type for each customer. Starbucks is one of the most famous companies in the world, it is a coffeehouse chain with more than 30 thousand stores all over the world. It is also very famous for offering its customers always the best service and the best experience. Like many companies nowadays they offer an app for their clients, where they can make orders, receive offers and information, etc. In this project we will focus on those offers. Every offer has a validity period before the offer expires. As an example, a BOGO offer might be valid for only 5 days. You'll see in the data set that informational offers have a validity period even though these ads are merely providing information about a product; for example, if an informational offer has 7 days of validity, you can assume the customer is feeling the influence of the offer for 7days after receiving the advertisement.

Problem Statement

The Starbucks company wants to figure out an effective way of sending special offers to their customers through the in-app notifications. Our mission is to create a Machine Learning model that based on historical data we can predict if we should send a discount offer to the user or not. The problem for this project is to solve and find the most appropriate offer for the customers, which means finding the right offer that will drive to a customer to buy Starbucks products.

Metrics

The accuracy of the models will be measured to evaluate the performance of the model. Since this is a classification problem we can use F1-score that combine precision and recall metrics to evaluate the model.

II. Analysis Data Exploration

The project contains simulated data that mimics customer behavior on the Starbucks rewards mobile app. The goal of the project is to analyze this "historical" data in order to implement an algorithm that finds the most suiting offer type for each customer. The data is contained in three files:

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)

portfolio.json

Offers sent during 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 reward
- duration: (numeric) time for offer to be open, in days
- offer_type: (string) bogo, discount, informational
- id: (string/hash)

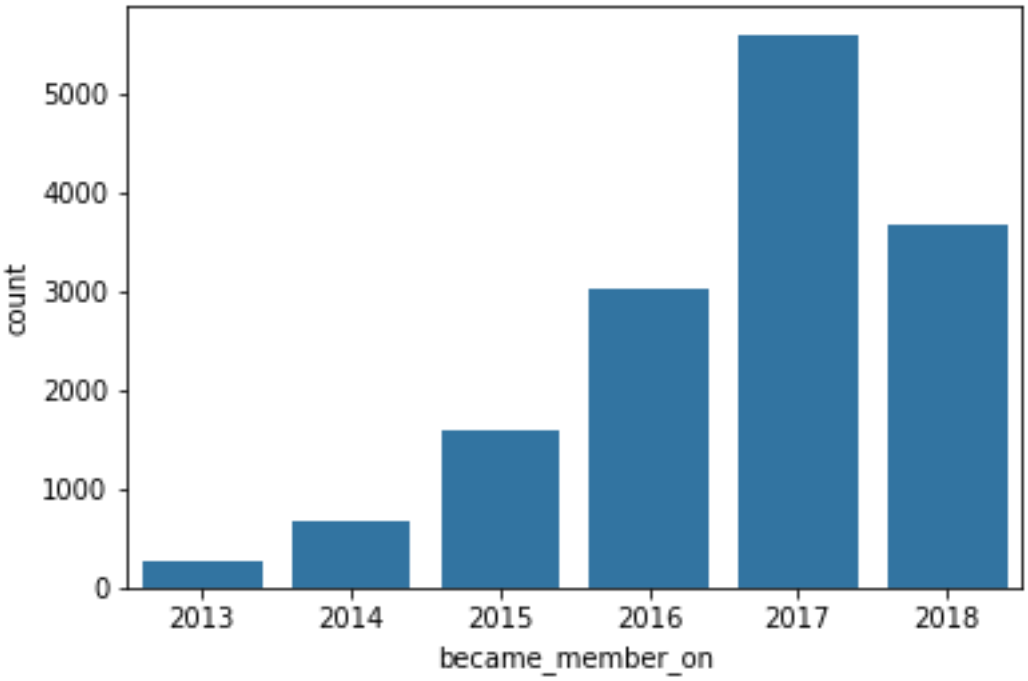
transcript.json

Event log (306648 events x 4 fields)

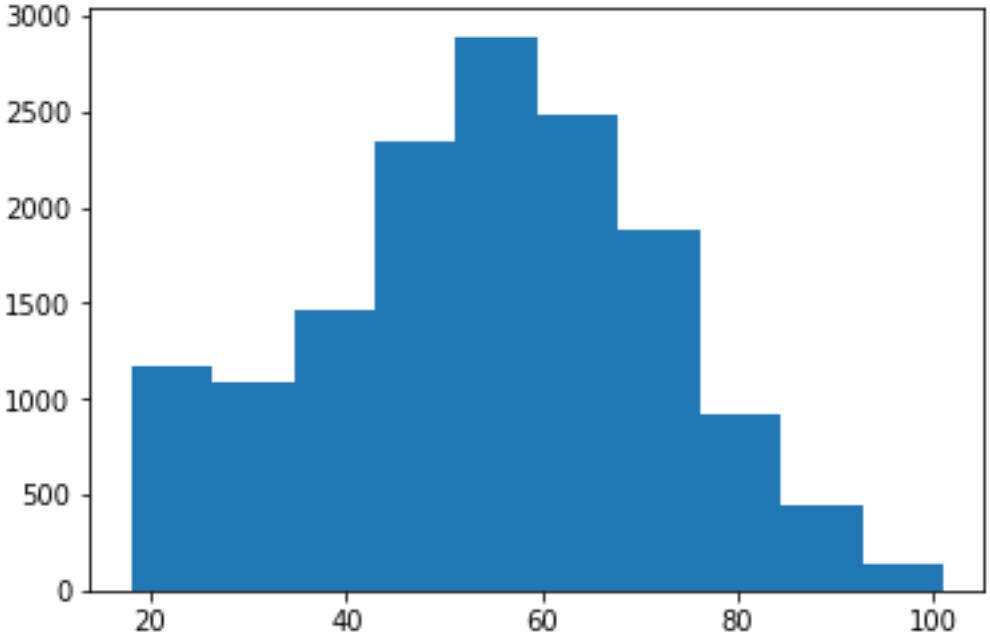
- person: (string/hash)
- event: (string) offer received, offer viewed, transaction, offer completed
- value: (dictionary) different values depending on event type
    - offer id: (string/hash) not associated with any "transaction"
    - amount: (numeric) money spent in "transaction"
    - reward: (numeric)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

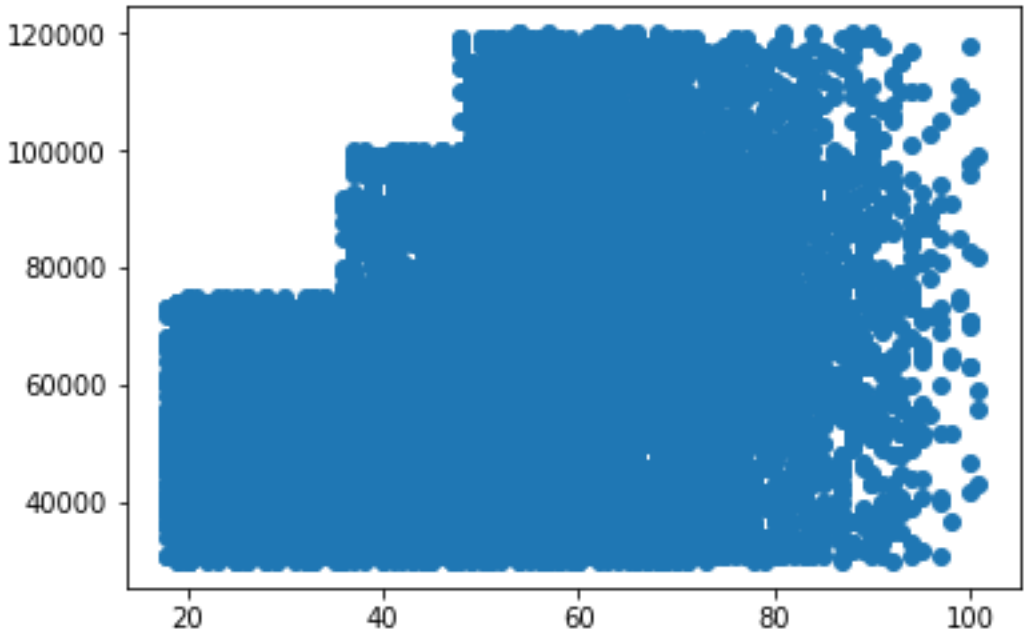
Exploratory Visualization

The next image describe the distribution of year when user became member of Starbucks reward app



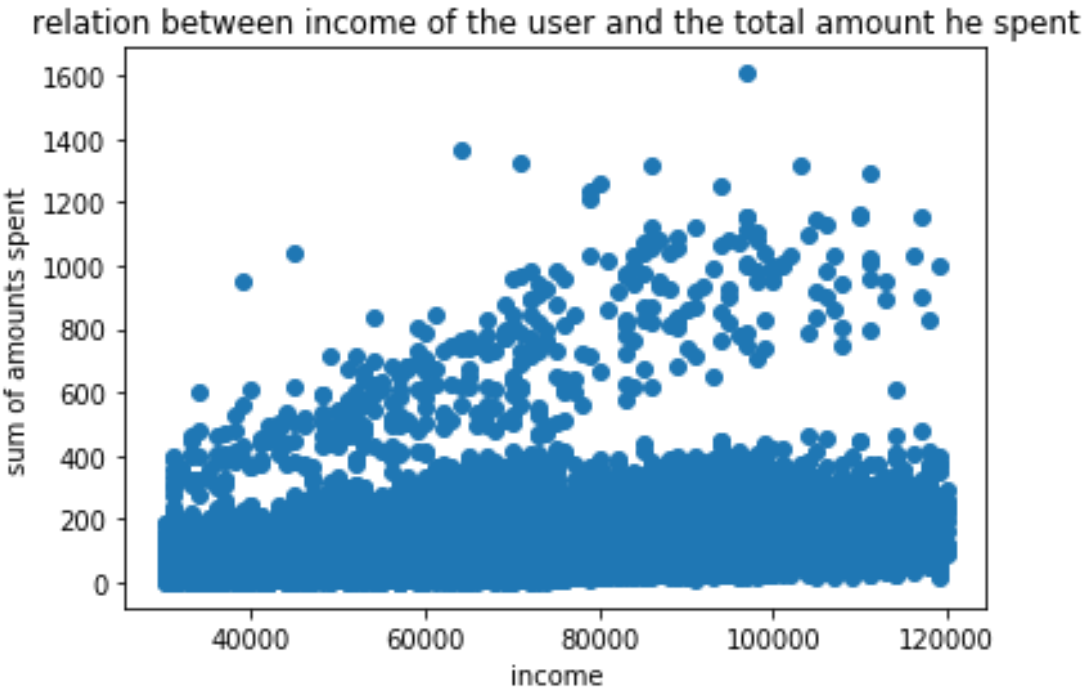
The next image describe the distribution of ages of users





The above image describe the relation between age of use and income of the user

The next image describe the relation between income of user and sum of amounts spent in purchases



Algorithms and Techniques

To address this problem and get the best offer for the customer we develop different machine learning models taking as input past data, with the corresponding label, the job of the model is to find the relationship between the input features and the labels. We explored two algorithms logistic regression and support vector machine to find the best models for each offer, a discount, a Buy One Get One (BOGO)or Informational.

Benchmark models

We will compare our results to other similar projects to compare performance of model of determining the most suitable offer for a customer to other models And here is work of others on the same problem and the same data to compare our results to

1 -https://towardsdatascience.com/starbucks-app-sending-the-right-offer-to-the-right-user-9d8d0e24e65c

2- https://towardsdatascience.com/using-starbucks-app-user-data-to-predict-effective-offers-20b799f3a6d5

Methodology

Data preprocessing

1 – **portfolio**
1.1 we create dummy variables for both offer type and channels feature

2 - **Profile data**

2.1 - remove all records in profile data that contain nan values in gender and income features and we found that these records are the same as ones with ages =118 which is suspicious.

2.2 – convert became_member_on to date_time feature type
2.3 - examining the relation between income and age of user
2.4 – examining relation between user gender and his income

2.5 – examining relation between gender of user and the year he became member user of the reward app

Transcript data

1 – Removing the users that have age of 118 from transcript data as we remove them from profile as they are outliers

2 – Creating 2 additional columns amount and offer id based on value feature

3 – separating the data into 2 transaction data in which users don' receive offers and offer data in which user receive offers

4 – Merging profile data with sum of amounts user spend (transaction data)

5 – Merging profile data with offer data

6 – Creating dummy variables for both gender event and year of when user became member in

Implementation

We use python and jupyter notebook. For the data preprocessing we used pandas library and for visualization we use matplotlib and seaborn libraries. For statistical analysis we used scipy and statsmodels library .Finally we use sklearn library for creating machine learning models evaluating them.

Refinement

We used hyperparameter tuning over support vector machine and logistic regression to improve results.

IV Results

Model validation and evaluation

We trained the 2 models on f1 score and evaluated on the same score on test part after splitting data with percentage of 20% for test data

We obtained very high score of both models after run both of them on

Data of bogo offer type, data of discount offer type and data of informational offer type

And this the results of models on bogo data

Support vector machine with f1 score =1


```
{'mean_fit_time': array([0.04752135, 0.03551579]), 'std_fit_time'
: array([0.01351595, 0.00249267]), 'mean_score_time': array([0.02
853918, 0.02602768]), 'std_score_time': array([0.00450325, 0.0010
0994]), 'param_C': masked_array(data=[10, 20],
             mask=[False, False],
       fill_value='?',
            dtype=object), 'param_kernel': masked_array(data=['li
near', 'linear'],
             mask=[False, False],
       fill_value='?',
            dtype=object), 'params': [{'C': 10, 'kernel': 'linear
'}, {'C': 20, 'kernel': 'linear'}], 'split0_test_score': array([1
., 1.]), 'split1_test_score': array([1., 1.]), 'mean_test_score':
array([1., 1.]), 'std_test_score': array([0., 0.]), 'rank_test_sc
ore': array([1, 1]), 'split0_train_score': array([1., 1.]), 'spli
t1_train_score': array([1., 1.]), 'mean_train_score': array([1.,
1.]), 'std_train_score': array([0., 0.])}
```


logistic regression with f1-score = 1


```
{'mean_fit_time': array([0.14860725]), 'std_fit_time': array([0.0
0349784]), 'mean_score_time': array([0.01300657]), 'std_score_tim
e': array([0.00200522]), 'param_C': masked_array(data=[1.2],
             mask=[False],
       fill_value='?',
            dtype=object), 'param_max_iter': masked_array(data=[1
500],
             mask=[False],
       fill_value='?',
            dtype=object), 'params': [{'C': 1.2, 'max_iter': 1500
}], 'split0_test_score': array([1.]), 'split1_test_score': array(
[1.]), 'mean_test_score': array([1.]), 'std_test_score': array([0
.]), 'rank_test_score': array([1]), 'split0_train_score': array([
1.]), 'split1_train_score': array([1.]), 'mean_train_score': arra
y([1.]), 'std_train_score': array([0.])}
```


results on discount data with f1 score = 1


```
{'mean_fit_time': array([0.03452122, 0.0345248 ]),
 'mean_score_time': array([0.0295217 , 0.02802992]),
 'mean_test_score': array([1., 1.]),
 'mean_train_score': array([1., 1.]),
 'param_C': masked_array(data=[10, 20],
              mask=[False, False],
        fill_value='?',
             dtype=object),
```

```
 'param_kernel': masked_array(data=['linear', 'linear'],
              mask=[False, False],
        fill_value='?',
             dtype=object),
 'params': [{'C': 10, 'kernel': 'linear'}, {'C': 20, 'kernel': 'linea
r'}],
 'rank_test_score': array([1, 1]),
 'split0_test_score': array([1., 1.]),
 'split0_train_score': array([1., 1.]),
 'split1_test_score': array([1., 1.]),
 'split1_train_score': array([1., 1.]),
 'std_fit_time': array([0.00349867, 0.00150144]),
 'std_score_time': array([0.00750637, 0.00201201]),
 'std_test_score': array([0., 0.]),
 'std_train_score': array([0., 0.])}
```

Results on informational data

Support vector machine with f1 score = 1

```
'mean_fit_time': array([0.01051807, 0.01251173]),
 'mean_score_time': array([0.01099491, 0.00950468]),
 'mean_test_score': array([1., 1.]),
 'mean_train_score': array([1., 1.]),
 'param_C': masked_array(data=[10, 20],
              mask=[False, False],
        fill_value='?',
             dtype=object),
 'param_kernel': masked_array(data=['linear', 'linear'],
              mask=[False, False],
        fill_value='?',
             dtype=object),
 'params': [{'C': 10, 'kernel': 'linear'}, {'C': 20, 'kernel': 'l
inear'}],
 'rank_test_score': array([1, 1]),
 'split0_test_score': array([1., 1.]),
 'split0_train_score': array([1., 1.]),
 'split1_test_score': array([1., 1.]),
 'split1_train_score': array([1., 1.]),
 'std_fit_time': array([0.00149131, 0.00350523]),
 'std_score_time': array([0.00101042, 0.0004977 ]),
 'std_test_score': array([0., 0.]),
 'std_train_score': array([0., 0.])}
```

Conclusion

Reflection

We can summarize the steps followed in this process

- Loading the 3 data files
2- Assessing data quality
3- Cleaning the data for instance removing duplicates, missing values, outlies, and resolving any inconsistencies in feature naming or representation of values

4- Statistical analysis of the 3 files
4.1- Profile data
4.2 – portfolio
4.3 - tarnscript
5- Preparing data for classification models
6- Trying different classification algorithms SVM and logistic regression
7- Evaluating the models on F1-score

I think personally wrangling the data and cleaning and creating featires necessary for machine learning model is the most difficult part of the project.

Improvements

The most relevant thing we could do is getting more data about products user purchase to get insights about user behavior like what are products are most purchased but certain demographic groups.

We could do further statistical analysis of profile data for example for each year of when use became member on what are moths that have highest number of users