Machine Learning Engineer Nanodegree

Capstone Project

Jose Mendez

August 6th, 2020

I. 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 7 days 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 the following metrics to evaluate the model:

- Precision The proportion of positive cases that were correctly identified.
- Recall The proportion of actual positive cases which are correctly identified.
- F1-score, that combines the two previous measures.
- Roc_auc_score Area Under the ROC Curve (AUC), a measure that calculates the area
  under the Receiving Operating Characteristic Curve. This particular curve accounts that
  higher probabilities are associated with true positives and vice-versa.

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:

- 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

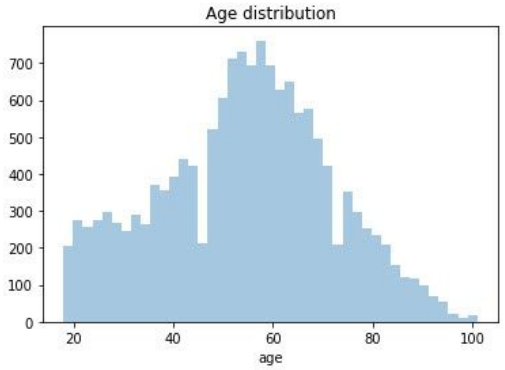
- age (int) - age of the customer
- became_member_on (int) - date when customer created an app account
- gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
- id (str) - customer id
- income (float) - customer's income

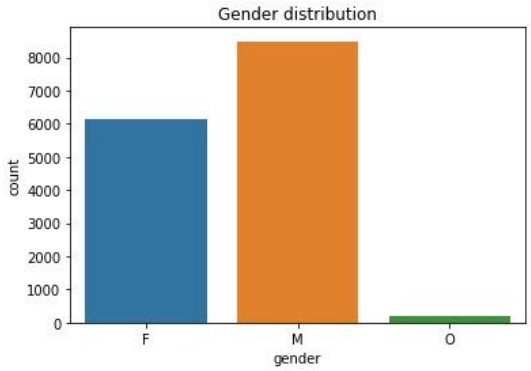
transcript.json

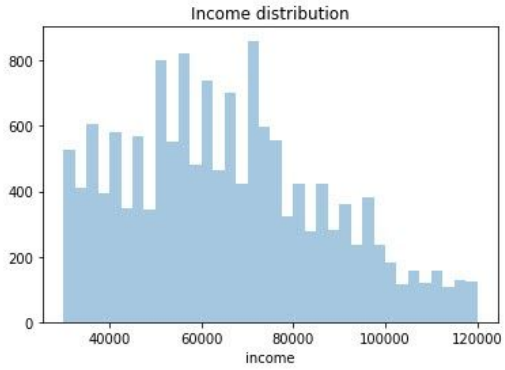
- event (str) - record description (ie transaction, offer received, offer viewed, etc.)
- person (str) - customer id
- time (int) - time in hours since start of test. The data begins at time t=0

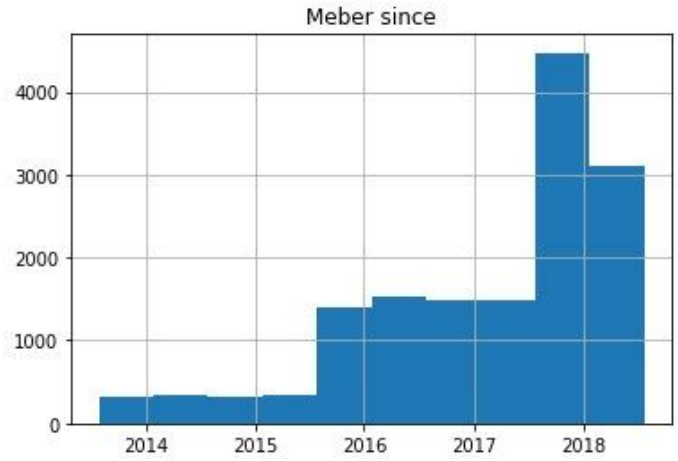
- value - (dict of strings) - either an offer id or transaction amount depending on the record

Exploratory Visualization

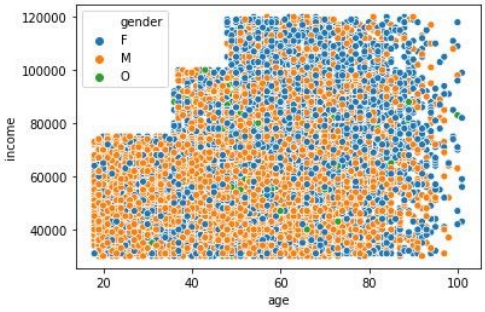









```
Members started on: 2013-07-29 - last member on 2018-07-26
```



Things to notice:

- there are a third gender o with very fwe records
- The age has a normal distribution, while income has irregular distribution
- The users are quite new members, the higher part of customers' memberships are concentrated from the last part of 2017 beginning of 2018
- The age income distribution is quite tight (no outliers)

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 **Linear Learner** and **XGBoost** to find the best models for for each offer, a Discount, a Buy One Get One (BOGO)
or Informational.

Benchmark

As the benchmark measure, we decided to take the current conversion rate for each type of offer.
After developing the models,
We are going to compare the results of the models' precision scores, and make some refinements using the Hyper parameter Tuning.

III. Methodology

Data Preprocessing

Starting from the journey dataset described in Section 2 - Data Exploration we create 3 different datasets, one for each type of offer. Each of these datasets contains only the data portion relative to viewed offers of the same type: for example. **bogo** dataset contains all the viewed BOGO offers with the relative target about offer completion. After that, we need to apply some data pre-processing steps to each dataset. These steps are:

- **Data sampling**: since 2 out of 3 targets are unbalanced (discount is 66%, info 39%) we can sample data to have better performances on the model. In particular we apply the undersampling method: we take all the records of the lowest frequent class (0 for discount, 1 for info) and a random subset of the most frequent, in order to have a balanced 50-50% ratio

- **Train-valid-test split:** divide the input data into 3 parts, each one stratified by the target (the event percentage is the same both on the original dataset and the 3 derived ones). The train set is the "real" input dataset for the algorithm: all the weights and the formula will derive from the relations found on this set. The valid set is to tune the hyperparameters of each model (more on this in the next paragraph). Finally, the test set is to compare the results of the several developed models to choose the best one

- **Missing imputation:** substitute missing values in features, approximating the possible real value. The methods to impute are several: we use a constant value for the only categorical variable (missing gender is replaced by "O") and the median value for the numeric features. Another common way to address missing numerical values is using the mean, but we preferred the median because it is a robust statistics even with skewed data and with outliers

- **Categorical encoding:** a Machine Learning algorithm can use only numerical data, so we must transform the gender variable into a numeric feature. Since we've got only one categorical input with only 3 categories, we decided to use One Hot Encoding: this method creates a binary feature for each category of the original variable, and each binary feature is equal to 1 if the relative category is present on that record.

Preprocessing with scikit-learn

- Encode categorical variable gender (One Hot encoder)
- Impute missing values
- Normalize feature distributions

Things to notice:

- There are 3 genres recorded in the dataset (F, M, O) which is going to be encoded during the feature engineering phase.

- The scatterplot shows some odd structure but it should be because the data was simulated, also it doesn't look to be outliers.

Implementation

The data pre-processing and the model were developed with aws cloud, with Sagemaker.
We use python and jupyter notebook on the cloud.
For the data preprocessing we used **SKLearnProcessor**. The hyperparameter tuning was done using **HyperparameterTuner.** The model query for the result using the Batch Transformation.
Finally we evaluate the model we measure and compare the performances of the models with the current benchmark, to learn if the proposed solution is viable for the offer attribution process.

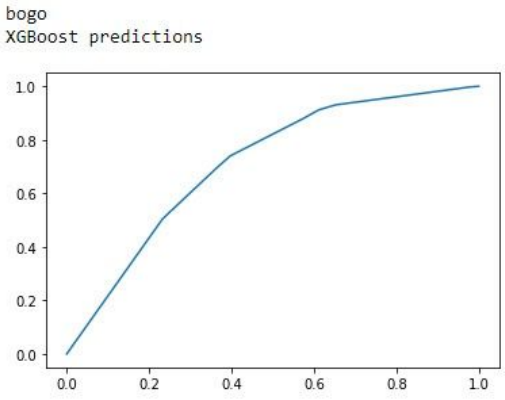
Refinement

The first time, we did not choose to sample the data, since the unbalance is not so strong, we tried to develop models with the original target distribution, However we noticed that there were unbalanced predictions.
We make some hyperparameter changes to verify if the performance improved or not.
At first the **Discount** model over predicted the positive event, and the info one over predicted the negative event. For this reason, we decided to do undersampling and we improved the performance.

IV. Results

Model Evaluation and Validation

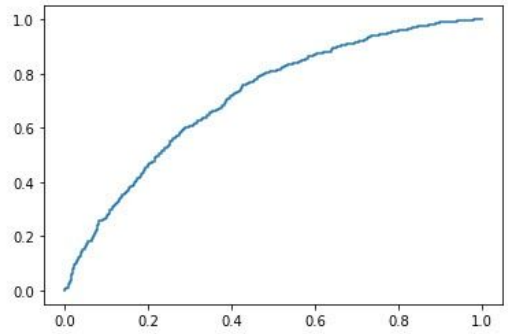
The XGBoost tuning for BOGO resulted in F1-scores on the validation set ranging from 0.7099 to 0.7366. The best one has these hyperparameters:
colsample_bytree: 0.665418398434084
eta: 0.3024061838045679
gamma: 0.15467677184991543
max_depth: 3
min_child_weight: 8
subsample: 0.7285079986427208



col_0	0	1
0		
0.0	0.602804	0.397196
1.0	0.259813	0.740187

	value
accuracy	0.671495
balanced_accuracy	0.671495
precision	0.650781
recall	0.740187
f1	0.692610
average_precision	0.649445
AUC	0.711577

bogo
LinearLearner predictions

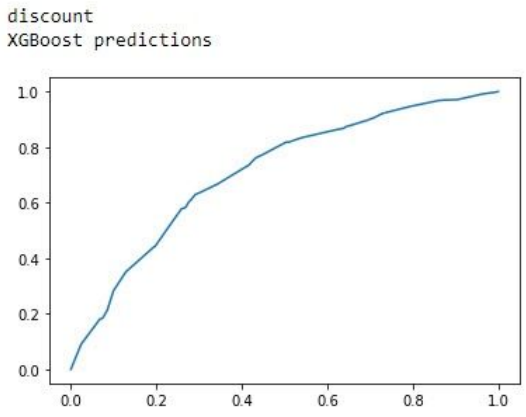


col_0	0	1
0		
0.0	0.417757	0.582243
1.0	0.142056	0.857944

	value
accuracy	0.637850
balanced_accuracy	0.637850
precision	0.595717
recall	0.857944
f1	0.703179
average_precision	0.680886
AUC	0.712177

The XGBoost tuning for discount resulted in F1-scores validation set ranging from 0.6980 to 0.7282. The best one has these hyperparameters:

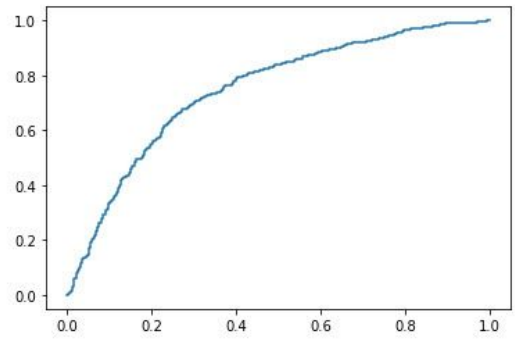
colsample_bytree: 0.778622530979818
eta: 0.4561709659187462
gamma: 4.844954586874898
max_depth: 5
min_child_weight: 2
subsample: 0.9165187239747279



col_0	0	1
0		
0.0	0.708535	0.291465
1.0	0.370370	0.629630

	value
accuracy	0.669082
balanced_accuracy	0.669082
precision	0.683566
recall	0.629630
f1	0.655490
average_precision	0.674997
AUC	0.710184

discount
LinearLearner predictions



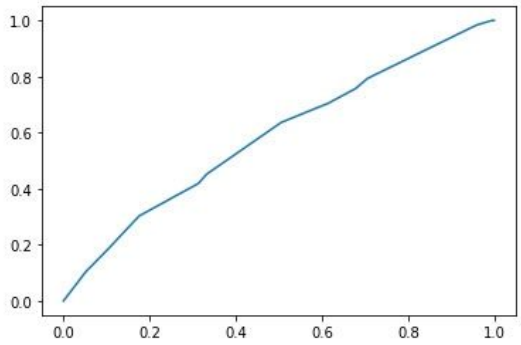
col_0	0	1
0		
0.0	0.439614	0.560386
1.0	0.132045	0.867955

	value
accuracy	0.653784
balanced_accuracy	0.653784
precision	0.607666
recall	0.867955
f1	0.714854
average_precision	0.706193
AUC	0.747288

In this case the best model is the **LinearLearner**, we used these hyperparameters:

epochs: 100
learning_rate: 0.01
loss: auto
mini_batch_size: 1000
optimizer: adam

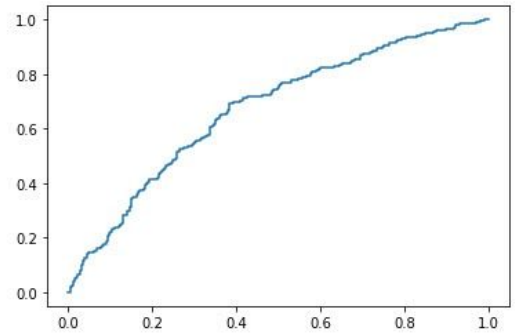
```
info
XGBoost predictions
```



col_0	0	1
0		
0.0	0.667582	0.332418
1.0	0.548209	0.451791

	value
accuracy	0.559835
balanced_accuracy	0.559687
precision	0.575439
recall	0.451791
f1	0.506173
average_precision	0.572414
AUC	0.586603

info
LinearLearner predictions



col_0	0	1
0		
0.0	0.192308	0.807692
1.0	0.066116	0.933884

	value
accuracy	0.562586
balanced_accuracy	0.563096
precision	0.535545
recall	0.933884
f1	0.680723
average_precision	0.642065
AUC	0.671124

Justification

We can see that the precisions of 2 out of 3 models are higher than our benchmark, the current conversion rate:

Offer	Current CR	Model precision	Delta
BOGO	50.47%	60.11%	+9.64%
Discount	65.73%	64,01%	-1.72%
Informational	38.82%	53,47%	+14.65%

The only negative case, **Discount**, has a small negative delta; the other two cases have a real positive delta. I think that the **Informational** is especially important, since on these types of offers Starbucks does not have to give money as a reward.

V. Conclusion

Reflection

Let's summarize all the different steps followed in this process.

1. We analyse the 3 different datasets containing information about offers in Starbucks app
2. Then we reconstruct the customer's journey through offer view, completion and transaction
3. After that, we create new input features to better understand one customer's behavior
4. We made some pre-process on input data
5. We develop different Machine Learning models, comparing them and choosing one champion model for each type of offer

I think that wrangling the data and understanding all the different special cases to reconstruct customer' journey was the most difficult and time consuming part. I personally took several iterations, founding every time some edge case was not taken under consideration. Also the model development through the Sagemaker platform was challenging, but very rewarding.

Improvement

We could achieve further improvements in several ways:

- The most relevant thing could be having more input features. Data about app usage and about which types of products a customer buys could really help understanding one's behavior
- also having more records at our disposal could improve the model's performances
- we could set a minimum precision for the Discount model in order to have a non-negative delta with the current benchmark
- we could try other algorithms, such as Neural Networks, SVM and Random Forests, as they could better suit this particular use case