Machine Learning Engineer Nanodegree

Capstone Proposal

John David LEQUEUX
December 12th 2020

Starbucks Capstone Challenge

Definition

Project Overview

To provide the next best offer to customers is a main topic in marketing. The goal is to predict the optimal personalized product offer to each individualized customers. As a consequence it will improve the chance that a customer uses the offer and then it will increase the promotion response rate. Another benefit is to improve the customer satisfaction by avoiding spamming.

With a high number of data and features to analyze, it requires a real time automated self-learning decision. Using machine learning algorithms is then necessary to create such model.

Problem Statement

The aim of this project is to predict if an offer will be well received by different customers. In other words the goal is to determine if a customer will use an offer or not.

This binary problem can be solved by using a machine learning algorithm based on a binary classification. This is the solution we will investigate in this project.

Metrics

We are taking the hypothesis that the company is focusing on the recall of the model so that it can find as many positive instances as possible. We decide to give more importance on the recall than on the precision because our objective is also to attract the most number of customer.

The risk linked to choose the recall over the precision is that it could increase the number of misclassification on negative class as positive. We assume that this risk is acceptable for the classification of the next best offer for a customer.

Consequently we will considerate the recall as the evaluation metric of the model.

Analysis

Data Exploration

The inputs are obtain using simulated data that mimics customer behavior on the Starbucks rewards mobile app. It contains three files with different information.

Portfolio

This file contains a description of 10 different offers.

Main characteristics:
- An offer can be a BOGO (buy one get one), a discount or it can be informational only.
- An offer can use up to four channel: email, social, mobile and/or web.
- The difficulty, the reward and the duration are integers between 0 and 20. We can notice that the difficulty and the reward are null for the informational offers.

Profile

This file contains a list of customers with information related to the age, the gender, the customer id, the date when the customer created an app account and the customer's income.

Main characteristics:
- 17000 entries
- 2175 rows don't have any information concerning the gender and the income. Moreover the age associated for these rows is always 118. We can take the assumption that this a default age for customers who haven't populate their age, gender and income in the application.
- Each customers have a different customer id.

Transcript

This file contains a list of record transaction with the customer id, the time in hours since start of test, and values depending on the record (either an offer id or transaction amount).

Main characteristics:
- 306534 entries
- There is no empty data in the file
- A recorded event can be either an offer received, viewed, completed or a transaction
- An offer can be completed before being viewed
- We can notice that a customer can receive the same offer up to five times
- The column value can contain two type of dictionaries:
  - a dictionary with only "amount" as key, if the event is a transaction.
  - a dictionary with one or two keys, "offer id" and "reward", if the event is an offer. We can also notice that the key used for a completed offer uses the key "offer_id" whereas a viewed and received offer use the key "offer id".

One other characteristic found during the exploration of the file transcript, is that there are two types of transactions:

- a transaction without link to an offer.
- a transaction linked to an offer. In this case, the linked offer(s) has/have the same time and is/are located just below the transaction in the file

Class balance

For each customers, we consider an offer as really completed if the number of time the person has completed the offer after viewing it, is bigger than the number of time he has completed it after viewing it. Moreover we only consider the BOGO and discount offers for the project.

By analyzing the percentage of completed and not completed offers, we have the following result:

- completed offer: 41.5%
- not completed offer: 58.5%

As we can notice the class is well balanced and consequently, we can confirm that the metric used for the project, the recall and the precision are appropriate for the project.

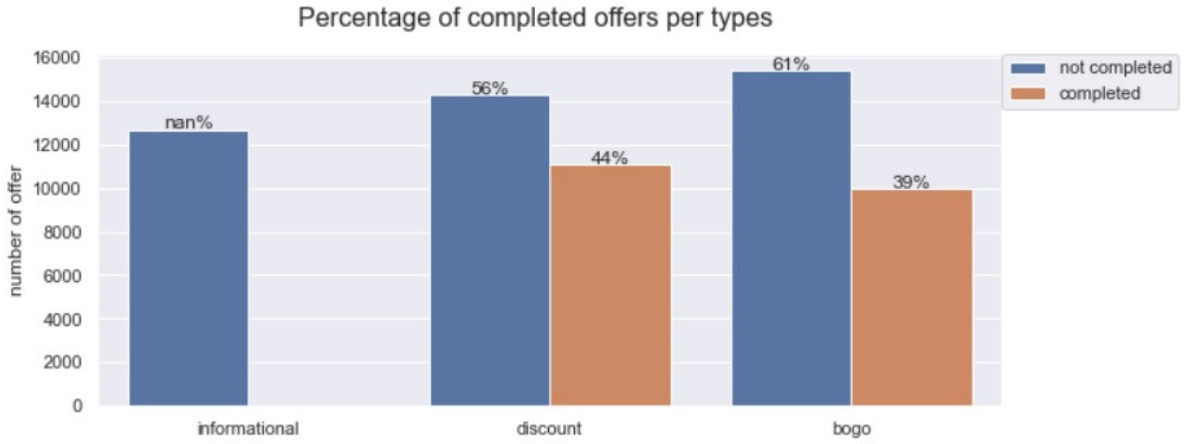
Exploratory Visualization

Since the aim of the project is to improve the promotion response rate, it is interesting to analyze how it is evolving following during features.

Notes:
- since we are focusing on the offers, the transactions not linked to an offer are removed from the analysis,
- for each customers, we consider an offer as really completed if the number of time the person has completed the offer after viewing it, is bigger than the number of time he has completed it after viewing it.

First we can analyze the percentage of completed offer following features related with the offers. Here below a set of plots with a discussion for each plots.

- Percentage of completed offers per types



On the above diagram we can easily figure out that a discount has more chance to be completed than a BOGO. Nevertheless, the difference is quite small and may not have a big impact in our classification model.

- Percentage of completed offers per channels

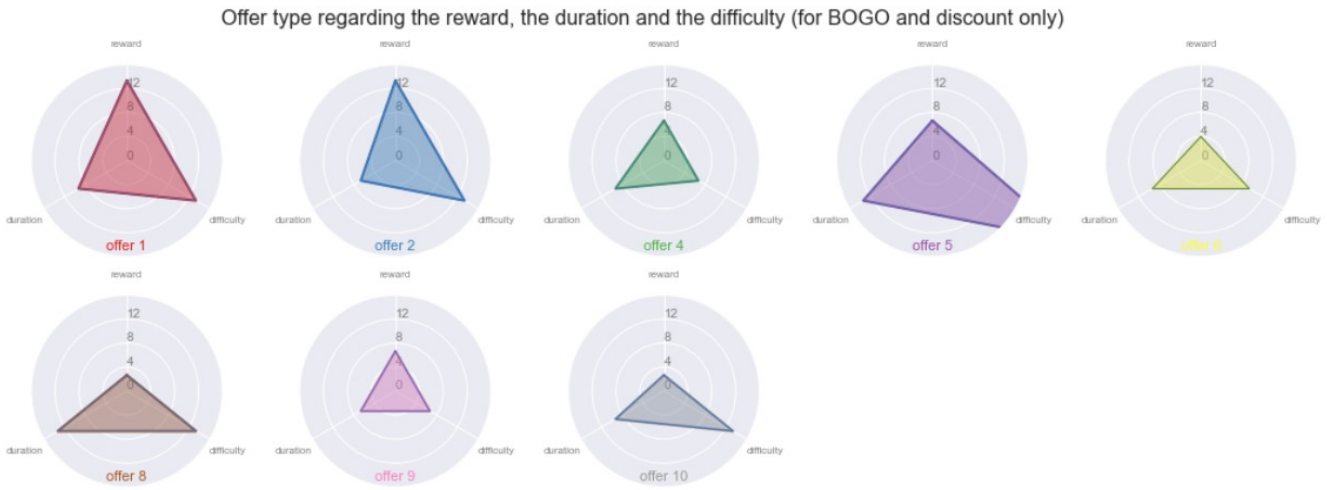
For a reminder, an offer uses at least two channels and can use the 4 types of channels. Moreover since the informational offers are not supposed to be completed, they are not taken into account for the following plots.

Taking each channel separately, we can notice that the web, the email and the mobile channels are linked to a completion rate of at least 55%. Only the social channel has a completion rate around 50%.

- Completed offers following the reward, the difficulty and the duration

We can notice the following characteristics:

- not surprisingly an offer with a lower difficulty will have more chance to be completed,
- the duration of an offer doesn't have a significant impact on the completion rate,
- the lower rewards have a higher completion rate. We can try to identify the reasons in the next radar chart.



These charts show the profile type of each BOGO and discount offers. We can notice that generally a higher difficulty involves a higher reward, and in the same way, a lower difficulty involves a lower reward. This the reason why a reward has more chance to be completed when looking on the previous diagram.

- Analysis of the completed offers regarding features linked to the customers

After analyzing the completion rate with features related to the offer, the analysis is performed now with features related to the customer profiles.

Here below, a set of three charts with the following setting:

- all profile with 118 years old as age are removed from the analysis
- the informational offers are also removed from the analysis

We can notice that the group of population which respond the best to the offers have the following characteristics:

- age between 50 and 65 years old
- income between 65000 and 80000 $
- they spend between 9 and 12 $ for each offer

We can notice that the people who didn't populate the form for the age, are the group with the lowest completion rate.
Consequently it could be interesting to focus the promotion offer on people who populates correctly the application form.

Algorithms and Techniques

After the exploration of the data, we will create a label data set and an input data set which will be used to train a binary classification model. We will use LinearLearner from Sagemaker which provides a solution for binary classification problems.

The algorithm used by LinearLearner allows to simulate different training objectives and more specifically it allows to optimize the recall of the model, which is the metric used for the project.

Benchmark

We will use a naive predictor for our model. As we have seen in the part Class balance, the promotion response rate is about 42 percents.

Our goal will be to improve this rate. To do this comparison we will compare the actual rate with the precision of the model. Indeed the precision will give us an idea about how would be the response rate if an offer has been sent to a customer in accordance with the prediction of the model.

Methodology

Data Preprocessing

Each files needs to be clean up and normalize before being able to use them in the binary classification. Here below an explanation about the exploration performed on the data set.

- Portfolio

The column “channel” is divided into four columns: web, email, mobile and social with the value 0 or 1 following the type of channel used.

We convert all offer type labels to numerical labels according to the following rules:

- 1 for informational
- 2 for discount
- 3 for BOGO

To improve the readiness of the file, we convert the offer ids into simple numerical values, and we use these new references as indexes for the file.

Consequently two dictionaries, containing the link between the offer ids and the offer type and the new references, are created for reference.

Here below the result of the portfolio after exploration:

	reward	difficulty	duration	offer_type	web	email	mobile	social
1	10	10	7	3	0	1	1	1
2	10	10	5	3	1	1	1	1
3	0	0	4	1	1	1	1	0
4	5	5	7	3	1	1	1	0
5	5	20	10	2	1	1	0	0
6	3	7	7	2	1	1	1	1
7	2	10	10	2	1	1	1	1
8	0	0	3	1	0	1	1	1
9	5	5	5	3	1	1	1	1
10	2	10	7	2	1	1	1	0

- Profile

Like for the portfolio file, we modify the ids for each customers, and the genders are converted as follow:

- 0 for None
- 1 for O
- 2 for F
- 3 for M

Two dictionaries containing the correlation between the different data will also be created.

For missing incomes in the last column, we replace these missing information by the mean of the global group.

Sample of the profile file after exploration:

	gender	age	became_member_on	income
0	0	118	20170212	65404.991568
1	2	55	20170715	112000.000000
2	0	118	20180712	65404.991568
3	2	75	20170509	100000.000000
4	0	118	20170804	65404.991568

- Transcript

The first step is to clean the column value. We split this column into three new columns: “offer id”, “amount” and “reward”. During this step we replace the offer and person ids by their new references from the reference dictionaries created during the exploration of profile and portfolio.
To ease the future jobs on the data, we create two new references for each lines:

- One using a combination of the person id and the offer id. We will call it: pers-offer.
- And another using a combination of the person id, the offer id and the associated time. We will call this reference: pers-offer-time.

The idea behind creating these two new references is that at the end of the transcript exploration, we would like to have a table where each lines corresponding to one couple person/ offer or one couple person/ transaction. Pers-offer-time will be used as index for our new table.

“0” is used as reference for the transaction id.

The second step is to determine if a transaction is linked to an offer or not. During this step we are comparing the time associated to each transactions to the other events, and if the compared event is a completed offer with the same time, we consider that this completed offer and this transaction are linked. In case of match:

- we update the pers-offer reference of the transaction with the same as the completed offer
- we modify the event type of the transaction into “transaction linked”
- we report the value of the amount of the transaction into the completed offer.

We notice during the exploration that a transaction can be linked to several offers. Consequently we decide to equally divide the amount of the transaction into each associated offers.

The last step is to create a data fame with the following characteristics:

- the event column is split into: received, viewed and completed
- a new column "completed before viewed" is added to check if an offer has been completed before being viewed
- for each couple person/offer or person/transaction, we will count how many time the offer has been received, viewed, completed and completed before being viewed,
- the default value for the couple transaction will be 1
- if an offer has been completed several times, the sum of the amounts will be added, same process for the rewards
- the goal is to have one line per person/offer (or person/ transaction).

The reason behind counting the number of time if an offer has been viewed, received or completed instead of using Boolean values is that an offer can be send several times to the same customer.

Sample of transcript file after exploration:

	pers-offer-time	received	viewed	completed	completed before viewed	amount	reward	person	offer	pers-offer
0	3-4-0	1	1	1	0	19.89	5	3	4	3-4
30959	3-0-144	1	1	1	0	17.78	0	3	0	3-0-49502
33917	3-8-168	1	1	0	0	0.00	0	3	8	3-8
52919	3-0-222	1	1	1	0	19.67	0	3	0	3-0-87134
55926	3-0-240	1	1	1	0	29.72	0	3	0	3-0-92104
86242	3-0-378	1	1	1	0	23.93	0	3	0	3-0-141566
91333	3-1-408	1	1	1	0	10.86	10	3	1	3-1
117943	3-9-504	1	1	1	1	10.86	5	3	9	3-9
131907	3-0-534	1	1	1	0	26.56	0	3	0	3-0-230412

- Data merge and creation of pers_offer_df, class_df and data_input_df

Portfolio, profile and transcript are first merged together. After the normalization of the data so that they can fall between 0 and 1, three new data frames are created:

- pers_offer_df contains the information concerning the number of time an offer has been received, viewed and completed, the type of offer and the person gender. Here below a sample:

pers-offer-time	received	viewed	completed	completed before viewed	person	offer	pers-offer	offer_type	gender
3-4-0	1	1	1	0	3	4	3-4	2	2
4-5-0	3	2	0	0	4	5	4-5	1	0
5-10-0	1	1	0	0	5	10	5-10	1	3
6-7-0	2	2	0	0	6	7	6-7	1	0
7-2-0	1	1	0	0	7	2	7-2	2	0

- class_df contains the labels for our binary classification. We consider pers-offer-time as “really” completed if the value “completed” is bigger that 1 and bigger than the value “completed before viewed”.
- data_input_df is the input data for our model and contains all the normalized numerical values. Here below a sample:

pers-offer-time	amount	reward	difficulty	duration	web	email	mobile	social	age	became member on	income
3-4-0	0.018724	0.5	0.25	0.571429	1	1	1	0	0.57	0.795648	0.777778
4-5-0	0.000000	0.5	1.00	1.000000	1	1	0	0	1.00	0.801548	0.393389
5-10-0	0.000000	0.2	0.50	0.571429	1	1	1	0	0.50	0.994000	0.444444
6-7-0	0.000000	0.2	0.50	1.000000	1	1	1	1	1.00	0.803968	0.393389
7-2-0	0.000000	1.0	0.50	0.285714	1	1	1	1	1.00	0.805508	0.393389

Note: before creating pers_offer_df, class_df and data_input_df, we remove all the transaction from the merged data. The reason behind is that we don’t want our model to be biased by the transactions since they are always consider as completed in our data set.
Moreover since the aim is to study the promotion response rate of different offers, it makes more sense to leave the transactions out of the binary classification.

Implementation

data_input_df and class_df are split into a train and test set using “train_test_split” from sklearn.model_selection. We split the data set into 2/3 training and 1/3 testing sets.

After converting the train features and labels into recordset, the estimator is then trained and deployed and a predictor is created. Here below the details of the initialization of LinearLearner.

```
# instantiate LinearLearner
linear = LinearLearner(role=role,
                                train_instance_count=1,
                                train_instance_type='ml.c4.xlarge',
                                predictor_type='binary_classifier',
                                output_path=output_path,
                                sagemaker_session=sagemaker_session,
                                epochs=15)
```

Refinement

We refine the model by using the functionalities of LinearLearner. More specifically we compare the result with a model without the target on the recall, and two models with target on the recall using two different value on the recall.

Here below the setting we will modify for the refinement.

```
binary_classifier_model_selection_criteria='precision_at_target_recall', # target recall
target_recall=0.85,
positive_example_weight_mult = 'balanced')
```

Results

Model Evaluation and Validation

We use a function which evaluate the recall, the precision and the accuracy of a binary classification model. This function determines the number of true positive, true negative, false positive and false negative and then return the associated recall, precision and accuracy of the model.

All the model are evaluate with 15 epochs.

- Evaluation of the model without target on the recall:

		prediction	
		0	1
Actual classification	0	11859	2050
	1	1495	5482

Recall: 78.6%
Precision: 77.8%
Accuracy: 83.0%

The number of false negative is quite high. Consequently we set LinearLearner as target on the recall to improve the result.

- Evaluation of the model target on the recall (target: 98%):

		prediction	
		0	1
Actual classification	0	8940	4969
	1	151	6826

Recall: 97.8%
Precision: 57.9%
Accuracy: 75.5%

The number of false negative has been reduced by about 90%. Nevertheless we can notice that the precision and the accuracy are very low. Those numbers may not be acceptable if we want to improve the customer satisfaction by reducing spamming.

- Evaluation of the model target on the recall (target: 85%):

		prediction	
		0	1
Actual classification	0	11637	2272
	1	980	5997

Recall: 86.0%
Precision: 72.5%
Accuracy: 84.4%

This last setting has the best trade-off between the recall and the precision. The number of false negative is better that the first model and the precision and accuracy are reasonably high.

Consequently a model with the same setting will be an appropriate solution for our binary classification problem.

Justification

If we compare the evaluation of our solution to our benchmark result, we can confirm that it is in accordance with our our goal: having a high recall and improving the promotion response rate.

The promotion rate of the input data is around 42%. If we use the prediction of our binary classification before sending the offers, we could expect to improve the promotion rate to about 70% and to optimize the false negative.

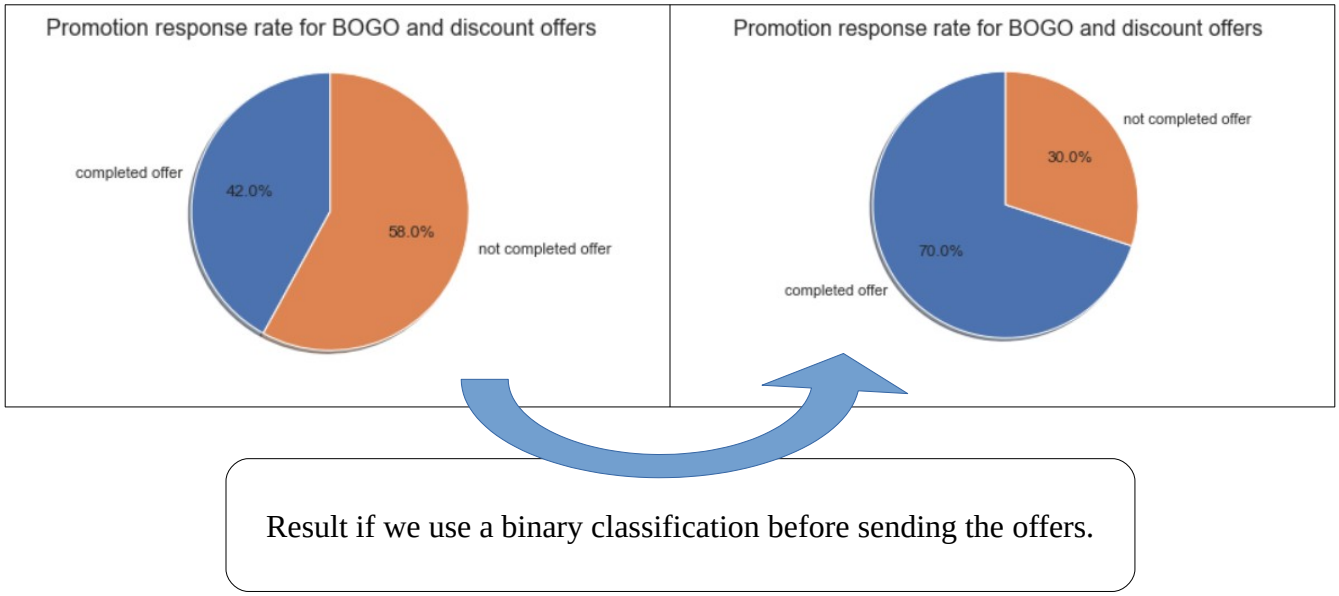
Conclusion

Reflection

After analyzing and cleaning the input data, we have used these transformed data to train a binary classification model.

We have performed the evaluation of the model with different setting of the model and we have choose a configuration with a target on the recall while keeping a high precision.

It’s interesting to see that the model created can be used to improve the promotion response rate of the BOGO and discount offers.



The exploration step was challenging with the high number of data and it has required to go through several steps of cleaning and transformation.

The result of the final model fit the expectations of the problem by providing a prediction that can be used to improve the promotion response rate of an offer campaign.

Improvement

The input data given to the trained model could be improved by adding more features to the model and using a PCA (Principal component analysis) afterward. Such features could be:

- cluster of group of people with different characteristics
- the time spent to complete an offer

Another way to improve the promotion response rate could be to create a model to identify the best type of customer profile for each offer.

Reference

In our Jupyter notebook:

- the function “make_spider” has been taken from taken from https://python-graph-gallery.com/392-use-faceting-for-radar-chart/

- the function “evaluate” has been taken from the Fraud_Detection_Exercise of this Nanodegree