# CAPSTONE PROJECT - STARBUCKS

Udacity - Machine Learning Engineer Nanodegree Program

Leonardo Simões

10/21/2020

## Summary

## 1. Project definition

The project in question consists of the implementation of what was proposed in the previous work.

The chosen theme was one suggested by the course program, analyzing data from the Starbucks application and applying machine learning models to obtain a meaningful ranking that can provide insights into this part of the company's business. The project aims at the practical application of the theoretical and practical concepts taught in the course for an application similar to business, and not just academic.

The generated models aim to classify the type of offer offered by the application, based on some of its characteristics. Classification by type of offer leads to current and past business insights, and confirms future consistencies.

The classification of the type of offer will take into account the relevant features of transcript, portfolio and profile. The effectiveness of the solution will be assessed according to a specific metric, the accuracy, and will determine the viability of the solution and the choice of model.

The models used were LinearLearner and XGBoost, both seen in the course and usable in the AWS SageMaker environment. After being trained, each model will be evaluated according to the chosen metric on a test set.

The evaluation metric that was used to evaluate both models will be the accuracy. In the project, a multilabel classification (3 labels) will be carried out, so that with the chosen approach, this metric was adequate and sufficient. In addition to displaying the calculated value for accuracy, the confusion matrix plot was also made showing the number of correct classifications, false positives and false negatives for each label.

## 2. Datasets and Inputs

The data was provided by the Udacity platform in the section of the project proposal, with the objective and authorization to carry out this project. The dataset is a simplified sample of the data from the Starbucks app, and there are three separate sets, each in a different json file.

The description provided with the datasets includes that of each file and each column. The description provided by Udacity has been pasted below:

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

After preparing the data, two datasets were generated from the previous three, using their joining and filtering the type of value (offer or transaction)

For XGBoost model, the training entry is the file train.csv, whose first column is the label, and the others are features. For testing, the entry is the x_train.csv file. In both cases, data is loaded from the file using the sagemaker.s3_input function. The y_test.csv file is loaded with the pandas library.

For the Linear Learner model, the files x_train.csv, y_train.csv, x_test.csv and y_test.csv are used using the pandas library. After being loaded as data frames, they are converted to float arrays. The input for training the model is formatted with the record_set method of the predictor. The model output is formatted using a loop that processes the prediction result.

## 3. Analysis

The distributions of the labels 'bogo' and 'discount' are larger than that of 'informational', indicating that there will be a supposed superiority in the classification of the first two labels. This will be assessed after training and testing the models.

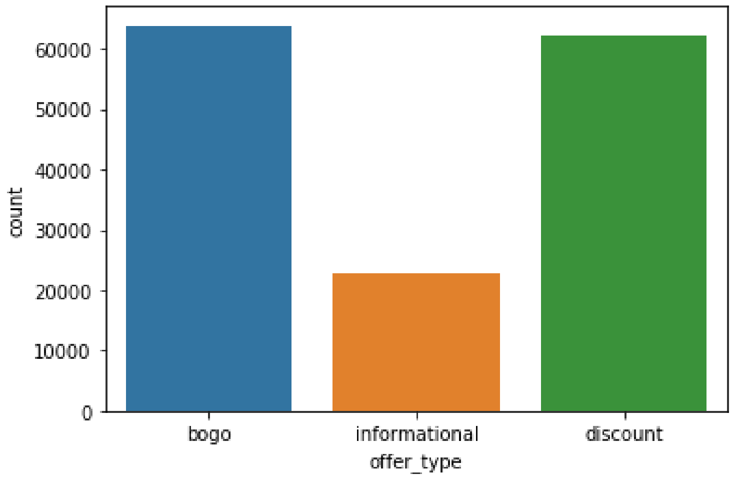

*Figure 1 – labels of the offer_type*

All offers were sent by email, but in general, they were sent by other means as well. Other means of delivery were social, mobile and web.

Initially, it was assumed that both models would have different results, even if approximately. No other result, other than the accuracy of the two models tested, is used as a reference.

The hyperparameters for each model are shown in the tables below. Some of these parameters refer to the sagemaker, others to the characterization of the problem (classification and number of classes), and the rest are 'default' options that have been adapted to similar problems.

| **Parameter** | **Value** |
|---|---|
| train_instance_count | 1 |
| train_instance_type | 'ml.c4.xlarge' |
| predictor_type | 'multiclass_classifier' |
| num_classes | 3 |
| epochs | 15 |

*Table 2 – Linear Learner parameters*

| **Parameter** | **Value** |
|---|---|
| objective | 'multi:softmax' |
| num_round | 100 |
| num_class | 3 |
| silent | 0 |
| subsample | 0.8 |
| min_child_weight | 6 |
| gamma | 4 |
| eta | 0.2 |
| max_depth | 5 |

*Table 2 – XGBoost parameters*

## 4. Methodology

The stages of the project were Data Wrangling, Feature Engineering, machine learning application. Each major stage of the project was implemented in a separate notebook.

The Data Wrangling stage was subdivided into gather, assessing, clean and tidiness. In the gather step, the datasets present in the .json files were loaded into dataframe objects in memory. In the assessing step, structural and data quality problems were verified and recorded, such as absence of values, if there were duplicates, formatting of cell values, consistency of columns. In the clean step, the data quality problems verified in the previous step were corrected. In the tidiness step, some dataframes were merged and others separated, in addition to solving data quality problems arising from these changes.

The Feature Engineering stage processed and modified the features and offer labels in order to allow the use of machine learning models for classification. In this step, the label and some features were transformed to numeric values, some columns were dropped, some new features were created. After this processing, the dataset was divided into train and test sets, for X and Y, and then saved in .csv files.

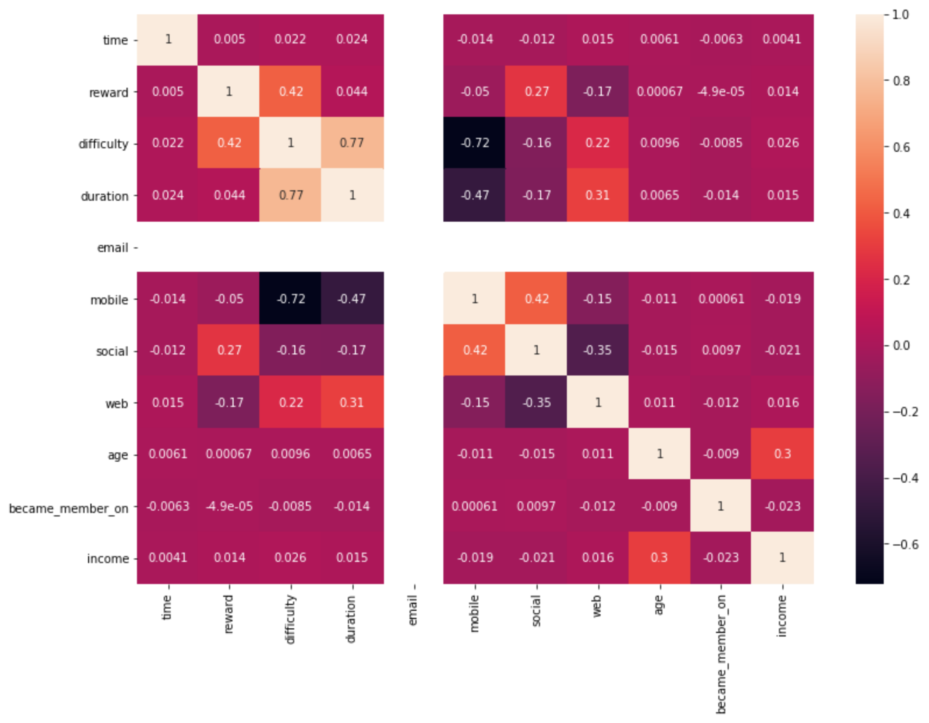

*Figure 3 - Correlation Matrix*

In the machine learning application stage, the procedures were analogous for the two chosen algorithms. Initially, there was configuration referring to AWS SageMaker how to define the variables of session, role, bucket, input_data. Then the previously generated training and test data was loaded. Then, an object was created for the model predictor with defined hyperparameters, which was trained with formatted data, and then it was deployed. The predictor makes the classification of the test set, and the result is compared with the real labels, enabling the calculation of accuracy and the plot of the confusion matrix.

There were no intermediate solutions for the models, that is, the initial solutions were the same as the final solutions. There was no optimization process for hyperparameters or changes in the models after being trained and tested.

## 5. Results

Both applied algorithms, Linear Learner and XGBoost, showed 100% accuracy for the multilabel classification problem. This result is very unusual in complex and / or realistic problems, but it occurred in this case due to some factors. One of the most likely justifications for the result was the strong correlation of the label with some of the features. Due to the similarity of the accuracy, it is not possible to verify differences between the two classification algorithms used.

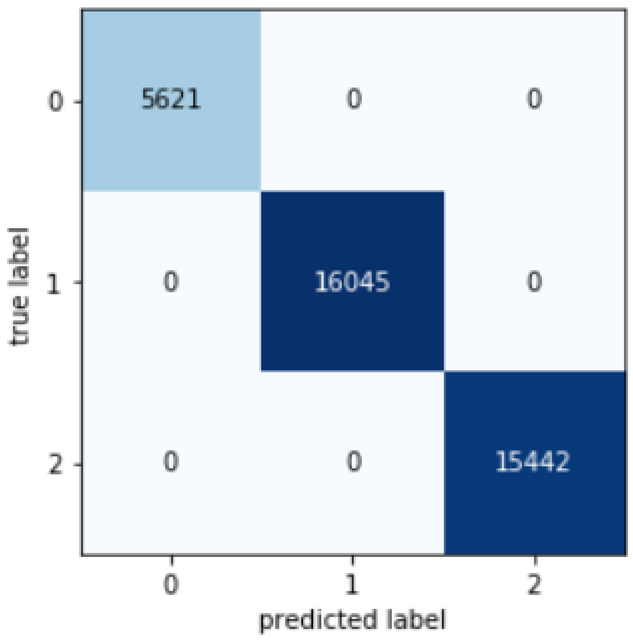

*Figure 4 - Confusion matrix*

The final models exceeded initial expectations. Hyperparameters proved to be satisfactory for the problem. The solution found was sufficient to solve the identified problem.

## REFERENCES

**UDACITY - Machine Learning Engineer Nanodegree:** https://www.udacity.com/course/machine-learning-engineer-nanodegree--nd009t