Udacity Starbucks Capstone Project Report

Machine Learning Engineer Nanodegree

Victor Barros

Table of Contents

Introduction

Solutions that involve customized delivery of value to customers have been developed by companies in several sectors. With the amount of data generated by various technological tools that are now available in the hands of customers, it is possible to build predictive models that boost business and increase engagement with products or services.

Starbucks is an American multinational company, with the largest coffee chain in the world. In addition to cappuccino and espresso, Starbucks offers other types of drinks, such as a variety of teas, as well as foods such as sandwiches. Starbucks stores can be inside other stores, such as bookstores and shopping malls.

The company has released a data set that simulates the behavior of customers using the Starbucks rewards mobile app. The Starbucks app is able to share and advertise offers with frequent customers. Consumers can also use it as a payment method in stores. Often the company sends different types of offers in a given period (based on consumer behavior).

Promotions can be discount ads on a particular drink, or they can also be discount promotions or buy one and get one free, it is also important to know that all promotions have an expiration date. The company provided a series of information on the behavior, consumption and demographics of its customers, in such a way that the solution allows building customized promotions for each customer that encourage minimum consumption within relevant consumer preferences.

Problem Statement

The company is looking for a solution that will allow customers to increase their consumption through personalized advertising and promotions. Considering the fact that the company is able to collect and process various relevant data about individual consumption, location, frequency and other information, it can target the ads and promotions to each customer at the exact right time.

The possible solution to the problem is to develop a predictive model, using machine learning techniques, that can deliver an offer with a higher probability of conversion for a customer. For this type of problem, where the outcome of the model is purchase or non-purchase, Binary Classifier proves to be the best approach.

Binary classification can be defined as the task of classifying the elements of a dataset into two distinct groups using a classification rule. In machine learning, the model is responsible for finding patterns in the features of the input data and being capable of a higher level of prediction than chance.

It is important to remember that there are multiple sources of data that can be analyzed and studied to create a set of characteristics that represent the greatest likelihood of success for an offer. However, it is important that the model keep it simple, and in such a way that it is not overly biased to the point of being useless.

Evaluation Metrics

To assess the quality of the model, I believe that a composition of evaluation metrics is necessary, such as: accuracy, precision, recall or F1 score.

Accuracy is a metric for evaluating classification models. In binary classification, accuracy can also be calculated in terms of positive and negative outcomes - and can be informally explained as the proportion of predictions that our model got right.

Precision is the proportion of correctly predicted positive observations relative to the total positively predicted observations.

Recall is the proportion of positive observations correctly predicted for all observations in the real class.

The F1 score is the weighted average of Precision and Recall. This metric takes into account both false positives and false negatives, and it is best used if we have an unbalanced distribution.

The evaluation formulas can be found below:

$$\text{Recall} = \frac{\text{TP}}{\text{TP + FN}} \qquad \text{Precision} = \frac{\text{TP}}{\text{TP + FP}} \qquad \text{Accuracy} = \frac{\text{Correct Guesses}}{\text{Total Population}} = \frac{\text{TP + TN}}{\text{TP + TN + FP + FN}}$$

$$F_1 = \frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} = \frac{\mathrm{tp}}{\mathrm{tp} + \frac{1}{2}(\mathrm{fp} + \mathrm{fn})}.$$

Given that we have a dataset with a good balance, precision is a better metric to evaluate the performance of the models.

Datasets and Inputs

The company has made available a set of transactional data that reflect consumer behavior within the application. 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

The format and dictionary of the files were made available below:

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

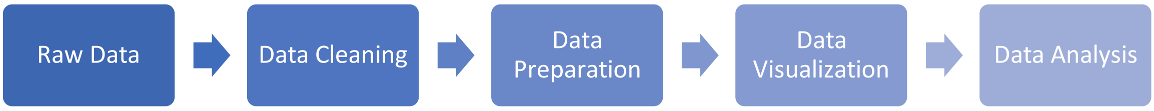
- age (int) - age of the customer
- became_member_on (int) - date when customer created an app account
- gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
- id (str) - customer id
- income (float) - customer's income

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

The dataset appears to be balanced, some fields have outliers that need to be addressed, and it will be important to understand which features will be even more meaningful for classification.

Using the available dataset, the following steps are followed to achieve a satisfactory result before modelling.



Data Cleaning

In order to better perform the exploration and analysis of the data set, it is necessary to clean and correct the columns of data, adjusting the different types of information present from the available models.

Cleaning portfolio.json

For this data set and its respective columns, the following adjustments will be necessary:

- id (string) – rename this column to “offer_id”
- offer_type (string) – transform to one hot encoded column
- channels (list of strings) – split de channels list of string in columns, transform to one hot encoded

Cleaning profile.json

For this data set and its respective columns, the following adjustments will be necessary:

- became_member_on (int) – adjust the column type to date type, split year into a separated column
- gender (str) – add “O” when there is no gender informed, transform to one hot encoded column
- id (str) – rename column to “customer_id”

Cleaning transcript.json

For this data set and its respective columns, the following adjustments will be necessary:

- event (str) – rename events value replacing white space for “_”
- person (str) – rename column to “customer_id”
- time (int) - time in hours since start of test. The data begins at time t=0
- value - (dict of strings) – create “offer_id” column and “amount” column with values, drop column “value”. Fill null column rows from “offer_id” and “amount” with 0

The data set was grouped taking into account "customer_id" and "offer_id" and "event", in such a way as to result in a consolidated view of performance per customer and offer.

After this consolidation, based on the occurrence of the "offer_completed" event, a new column called "succcessful" was created to record the conversions that were a consequence of the campaign.

It is important to emphasize that a small adjustment was made, considering only the conversions of offers that were viewed by the customer, and not conversions that occurred without interaction with the offer.

Data Exploration and Visualization

In order to better explore and analyze the data sets, it was necessary to create a crossover of the three data sets. This merger gave rise to a more complete set, bringing characteristics of the portfolio, customers and conversion aspects.

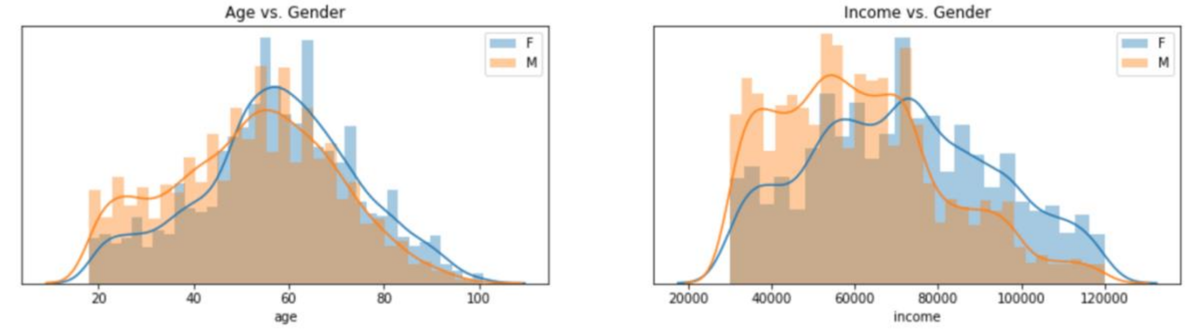
To assist in reading the analyzes, it was important to label the portfolio's offerings in order to facilitate the identification of the different types (BOGO, informational or discount).

Our final data set was organized as follows:

```
Int64Index: 79866 entries, 0 to 79865
Data columns (total 34 columns):
customer_id                 79866 non-null object
offer_id                    79866 non-null object
transaction                 16578 non-null float64
successful                  79866 non-null float64
age                         79866 non-null int64
became_member_on            79866 non-null datetime64[ns]
gender                      79866 non-null object
income                      69714 non-null float64
became_member_on_year       79866 non-null int64
gender_F                    79866 non-null uint8
gender_M                    79866 non-null uint8
gender_O                    79866 non-null uint8
difficulty                  63288 non-null float64
duration                    63288 non-null float64
offer_type                  63288 non-null object
reward                      63288 non-null float64
channel_email               63288 non-null float64
channel_mobile              63288 non-null float64
channel_social              63288 non-null float64
channel_web                 63288 non-null float64
offer_type_bogo             63288 non-null float64
offer_type_discount         63288 non-null float64
offer_type_informational    63288 non-null float64
offer_label                 63288 non-null object
offer_label_BOGO_1          79866 non-null uint8
offer_label_BOGO_2          79866 non-null uint8
offer_label_BOGO_3          79866 non-null uint8
offer_label_BOGO_4          79866 non-null uint8
offer_label_DISCOUNT_1      79866 non-null uint8
offer_label_DISCOUNT_2      79866 non-null uint8
offer_label_DISCOUNT_3      79866 non-null uint8
offer_label_DISCOUNT_4      79866 non-null uint8
offer_label_INFORM_1        79866 non-null uint8
offer_label_INFORM_2        79866 non-null uint8
dtypes: datetime64[ns](1), float64(13), int64(2), object(5), uint8(13)
```

Gender Analysis with Age and Income

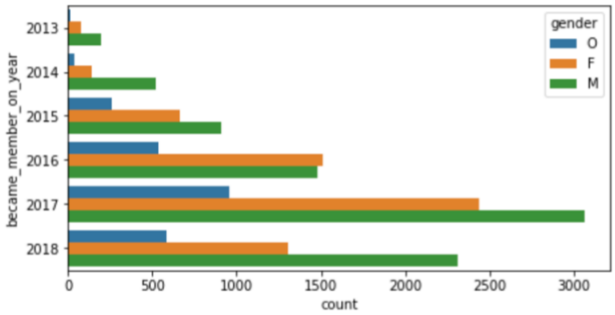
Looking at a cut in the dimensions of gender compared to age and income, we can see from the distribution graphs that there is a clear difference in the data sets.



We can conclude that an important group of the company's customers is in the age range of 50 to 65 years, with a predominance of female audience over 50 years and male audience under 50 years old.

It was also possible to identify the existence of points outside the curve that are over 100 years old and do not appear to represent real profile data.

As for the analysis of the clients' income, the male audience is more represented and concentrated in the range between 30,000 and 70,000, while the female audience was better distributed between 40,000 and 100,000.



If we analyze the year of entry of each client, we can see that most clients entered in 2018, 2017 and 2016. This parameter is not balanced and may represent a bias within the model.

The following analysis compared the frequency between gender, age and income, but considering a cut between offers that were successful in conversion and those that were not successful.

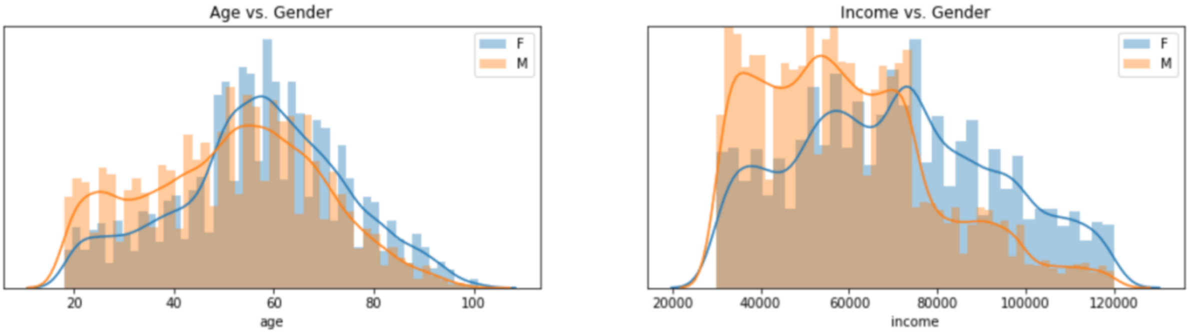


Figure 1 Unsuccessful offers by Gender with Age and Income

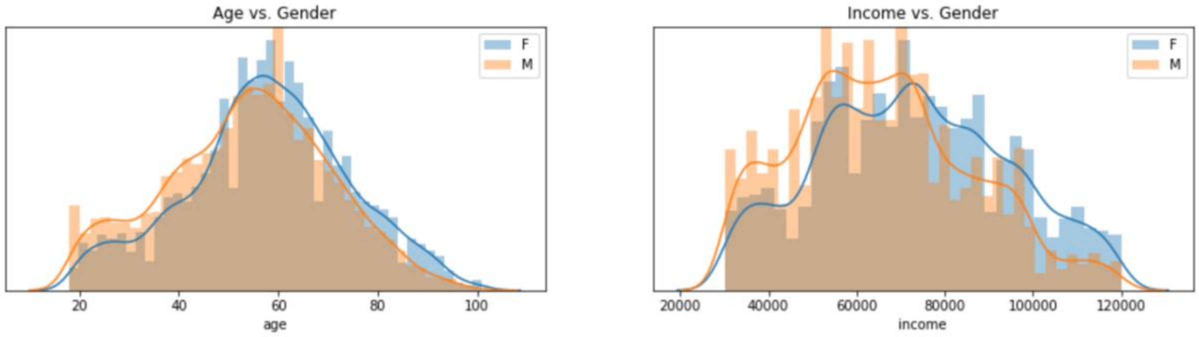
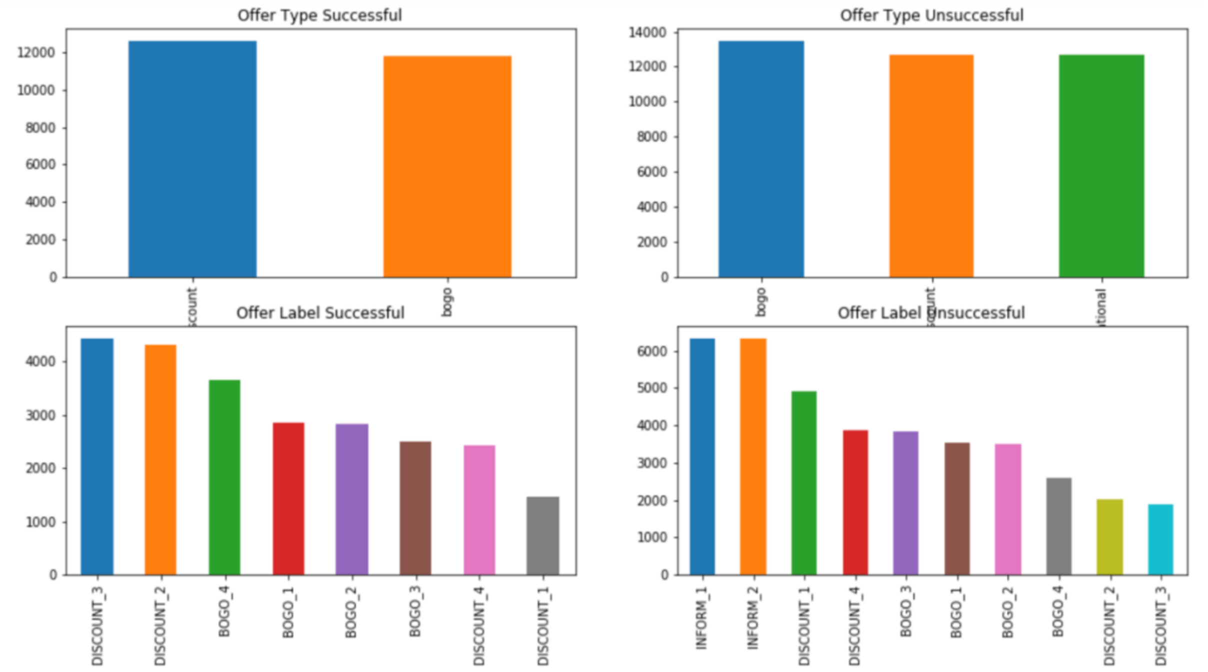


Figure 2 Successful offers by Gender with Age and Income

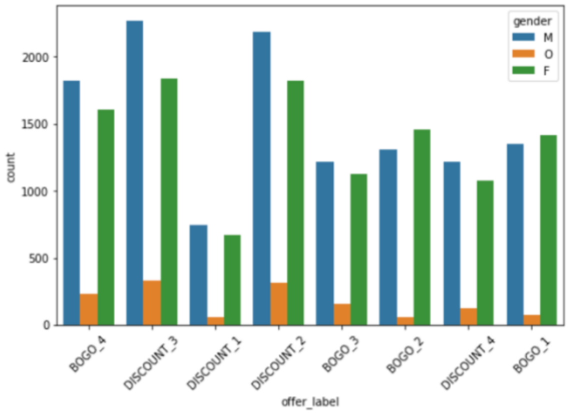
The bids that have been successful in implementation are relatively balanced and do not appear to have biases that could hinder the construction of the model. It is also important to note that they represent a representative set that is quite similar to the original data set.

Offer Analysis

The analysis based on the offers and their conversion success or not, shows that the BOGO and the discount offers are balanced and have a good sample that can be used for training the model.



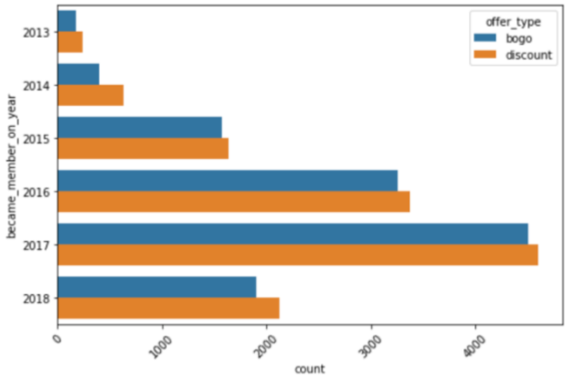
If we only consider the subset of offers with successful conversion, information offers are not considered, and discount offers are more prevalent than BOGO.



If we look at the successful bids according to the gender dimension, there is an excellent balance between men and women. There are very few cases that have no gender.

Become Member with Offer Conversion

Considering only successful offers, there is a good balance between the period of customer entry and the conversion of each offer.



Outliers and Missing Values

It was possible to identify during the analysis of the data that age has some non-standard data points, in this case, customers with 118 years of age, which is not coherent.

It was also possible to identify missing values in the income and gender columns. My solution to deal with this scenario was to define an “O” value for the gender, and in the case of the income field, fill in the average of the data set.

Data Preparation, Modelling and Benchmark

Data Preparation

It was necessary to remove the columns that would not represent compatible data for the construction of the model, or even, it could bias the training.

The following columns have been removed from the data set:

```
customer_id                 79866 non-null object
became_member_on            79866 non-null datetime64[ns]
gender                      79866 non-null object
offer_type                  63288 non-null object
offer_type_bogo             63288 non-null float64
offer_type_discount         63288 non-null float64
offer_type_informational    63288 non-null float64
transaction                 16578 non-null float64
difficulty                  63288 non-null float64
duration                    63288 non-null float64
reward                      63288 non-null float64
```

For model training, we only consider data that refer to offers, and not related to the transaction. It was also necessary to fill in all the lines where 'income' has no value with the median.

Our final data set for entering the training model has the following structure:

```
successful                63288 non-null float64
age                       63288 non-null int64
income                    63288 non-null float64
became_member_on_year     63288 non-null int64
gender_F                  63288 non-null uint8
gender_M                  63288 non-null uint8
gender_O                  63288 non-null uint8
channel_email             63288 non-null float64
channel_mobile            63288 non-null float64
channel_social            63288 non-null float64
channel_web               63288 non-null float64
offer_label_BOGO_1        63288 non-null uint8
offer_label_BOGO_2        63288 non-null uint8
offer_label_BOGO_3        63288 non-null uint8
offer_label_BOGO_4        63288 non-null uint8
offer_label_DISCOUNT_1    63288 non-null uint8
offer_label_DISCOUNT_2    63288 non-null uint8
offer_label_DISCOUNT_3    63288 non-null uint8
offer_label_DISCOUNT_4    63288 non-null uint8
offer_label_INFORM_1      63288 non-null uint8
offer_label_INFORM_2      63288 non-null uint8
```

Model Pipeline

To optimize the training of the models, I chose to transform all values using *MinMaxScaler*. With that, all the columns had their values transformed and the lines were randomly mixed to segregate the training and testing data. For more information about shuffle: https://scikit-learn.org/stable/modules/generated/sklearn.utils.shuffle.html

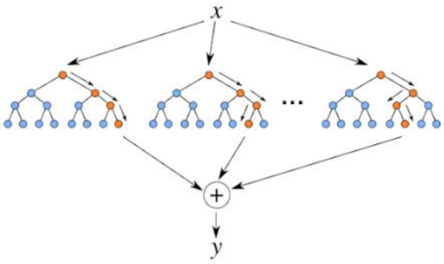
I chose to separate 80% of the data set for training and 20% for testing, ensuring adequate sampling for the chosen models.

Algorithms and Techniques

For all models, I performed fine-tuning before finding the best training parameters. I also decided to standardize the analysis by using *classification_report* to extract the performance metrics from the model.

RandomForestClassifier

Random Forest is a powerful and most widely used supervised learning algorithm in Machine Learning. It enables rapid identification of significant information from large datasets.



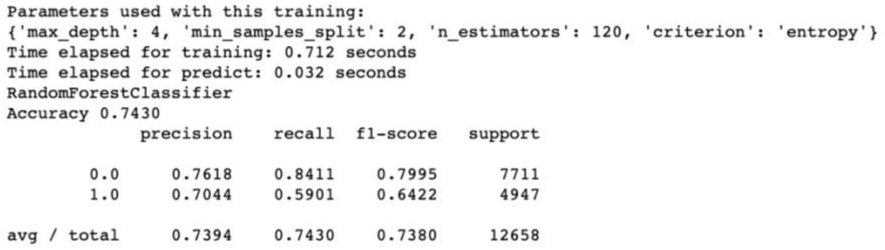
The advantages are:

- Works well with data sets with outliers;
- Works well with non-linear data;
- Less risk of overfitting after training;
- Works efficiently on a large data set;
- Better accuracy than other classification algorithms.

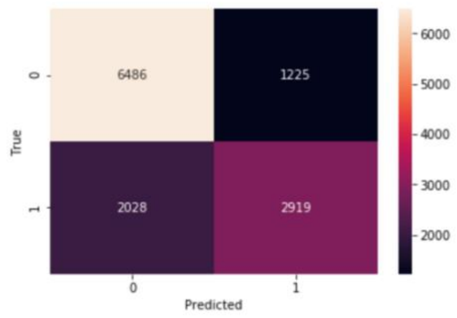
The disadvantages are:

- Random Forests are biased when working with categorical variables;
- Slow training;
- Not suitable for linear methods with many sparse resources.

For this model was decided to use *GridSearchCV* to find the best parameters for using the model.

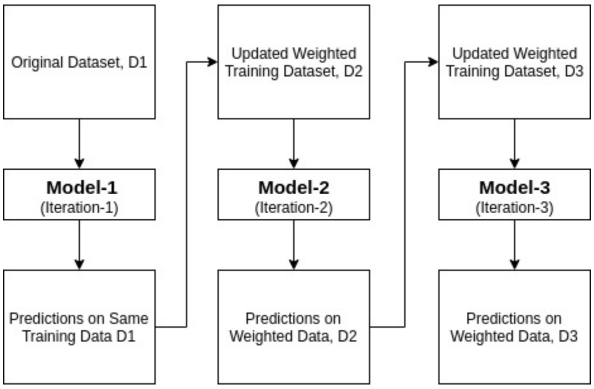


```
Parameters used with this training:
{'max_depth': 4, 'min_samples_split': 2, 'n_estimators': 120, 'criterion': 'entropy'}
Time elapsed for training: 0.712 seconds
Time elapsed for predict: 0.032 seconds
RandomForestClassifier
Accuracy 0.7430
             precision    recall  f1-score   support

        0.0     0.7618    0.8411    0.7995      7711
        1.0     0.7044    0.5901    0.6422      4947

avg / total     0.7394    0.7430    0.7380     12658
```



AdaBoostClassifier

AdaBoost, short for Adaptive Boosting, is a machine learning meta-algorithm that can be used in combination with many other types of learning algorithms to improve performance. This model is adaptive in the sense that subsequent weak learners are optimized in favor of instances that were misclassified by previous classifiers. It is best described as a decision tree algorithm that does not require scaled data.


Original Dataset, D1
Model-1
(Iteration-1)
Predictions on Same Training Data D1
Updated Weighted Training Dataset, D2
Model-2
(Iteration-2)
Predictions on Weighted Data, D2
Updated Weighted Training Dataset, D3
Model-3
(Iteration-3)
Predictions on Weighted Data, D3

The advantages are:

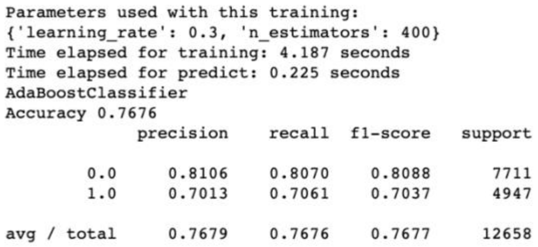
- Easy to implement;
- It iteratively corrects weak classifier errors and improves accuracy by combining weak learners;
- You can use many basic classifiers with the model;
- Not prone to overfitting;

The disadvantages are:

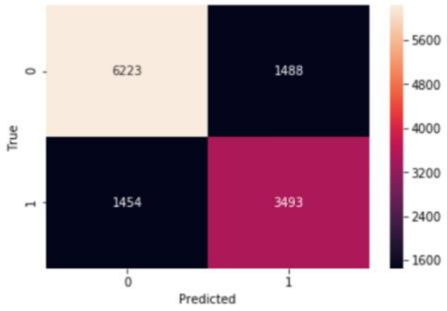
- Sensitive to noise data;

- Highly affected by outliers because it tries to fit each point perfectly;
- Slower compared to other models;

For this model was decided to use *GridSearchCV* to find the best parameters for using the model.

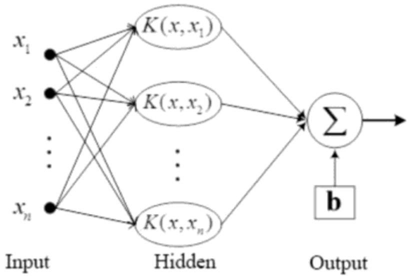


```
Parameters used with this training:
{'learning_rate': 0.3, 'n_estimators': 400}
Time elapsed for training: 4.187 seconds
Time elapsed for predict: 0.225 seconds
AdaBoostClassifier
Accuracy 0.7676
             precision    recall  f1-score   support

        0.0     0.8106    0.8070    0.8088      7711
        1.0     0.7013    0.7061    0.7037      4947

avg / total     0.7679    0.7676    0.7677     12658
```



SVC

SVC (Support Vector Classifier) goal is to fit the data provided and return a "best fit" hyperplane that divides or categorizes input data, that maximize the decision boundary margin between classes. This model can be used for both classification or regression challenges.

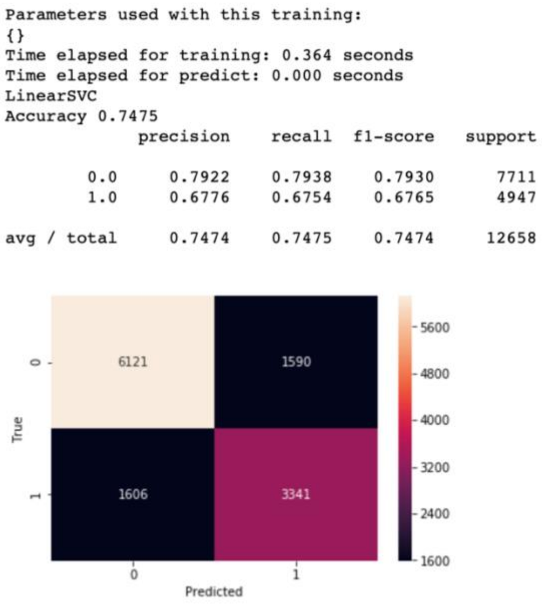


The advantages are:

- Works great with a clear dividing line;
- Effective in high dimensional spaces;
- Effective in cases where the number of dimensions is greater than the number of samples.

The disadvantages are:

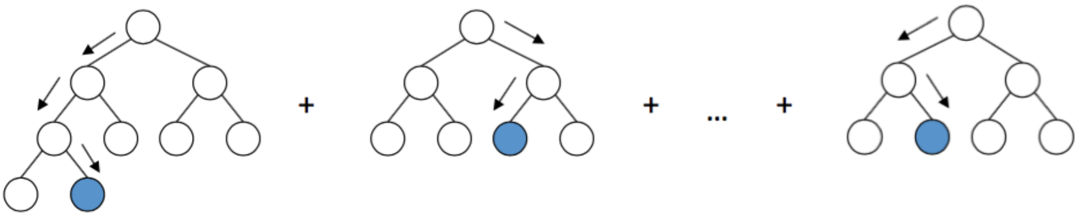
- Does not perform well when we have a large data set because the training time required is higher
- Does not perform well when the dataset contains more noise



```
Parameters used with this training:
{}
Time elapsed for training: 0.364 seconds
Time elapsed for predict: 0.000 seconds
LinearSVC
Accuracy 0.7475
             precision    recall  f1-score   support

        0.0     0.7922    0.7938    0.7930      7711
        1.0     0.6776    0.6754    0.6765      4947

avg / total     0.7474    0.7475    0.7474     12658
```

GradientBoostingClassifier

Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models to create a strong predictive model, and they are effective at classifying complex datasets.



The concept behind gradient boosting is to take a weak learning algorithm and apply a series of optimizations to it that improve the strength of the learner. It builds an additive model forward stepwise, it allows the optimization of arbitrary differentiable loss functions.

The advantages are:

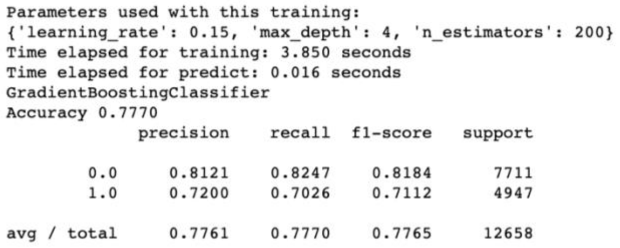
- Often provides predictive accuracy that cannot be trumped;

- Flexible, can optimize for different loss functions, and offers multiple options for setting hyperparameters that make fitting the function very flexible;
- No data preprocessing required, often works great with categorical and numeric values in their original state.

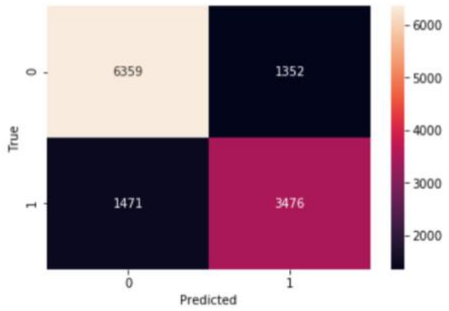
The disadvantages are:

- Continually improve to minimize all errors. This can overemphasize outliers and lead to overfitting;
- Computationally expensive, often requires many trees, which can be time and memory consuming;
- Flexibility leads to many parameters that interact and greatly affect the behavior of the approach Requires a large grid search during tuning.

For this model was decided to use *GridSearchCV* to find the best parameters for using the model.

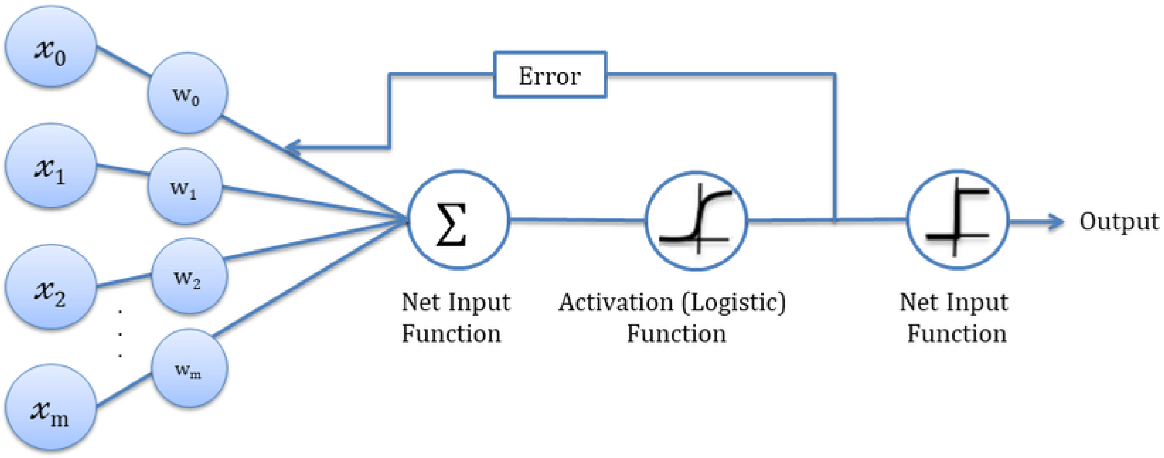


```
Parameters used with this training:
{'learning_rate': 0.15, 'max_depth': 4, 'n_estimators': 200}
Time elapsed for training: 3.850 seconds
Time elapsed for predict: 0.016 seconds
GradientBoostingClassifier
Accuracy 0.7770
             precision    recall  f1-score   support

        0.0     0.8121    0.8247    0.8184      7711
        1.0     0.7200    0.7026    0.7112      4947

avg / total     0.7761    0.7770    0.7765     12658
```



LogisticRegression

This type of model is excellent for binary classification problems. They have some underlying assumptions, they assume that class outcomes are interdependent and require little or no colinearity between the independent variables, the dependent variables or features should not be highly correlated.



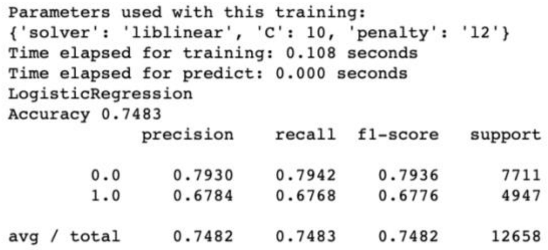
The advantages are:

- Logistic regression is easier to model, read and interpret, and train very efficiently;
- It makes no assumptions about distributions of classes in the feature space;
- Good accuracy for many simple data sets and it performs well if the data set is linearly separable;
- It is very fast at classifying unknown data sets;
- Logistic regression is less prone to overfitting, but it can overfit in high-dimensional data sets.

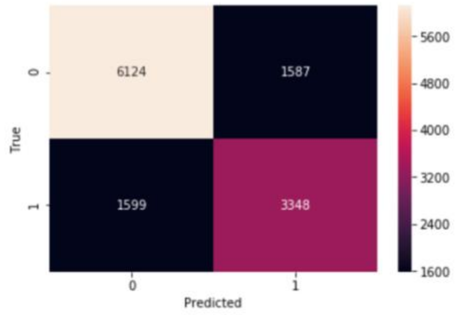
The disadvantages are:

- It constructs linear boundaries
- A major limitation is the assumption of linearity between the dependent variable and the independent variables
- Requires average or no multicollinearity between independent variables
- It is difficult to obtain complex relationships

For this model I decided to use *GridSearchCV* to find the best parameters for using the model.

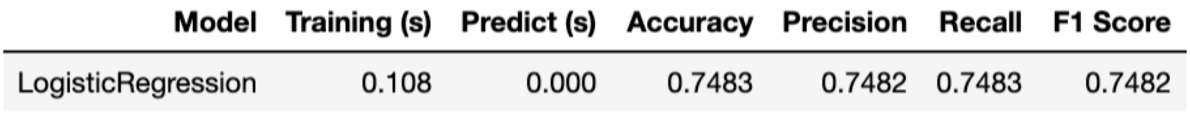


```
Parameters used with this training:
{'solver': 'liblinear', 'C': 10, 'penalty': 'l2'}
Time elapsed for training: 0.108 seconds
Time elapsed for predict: 0.000 seconds
LogisticRegression
Accuracy 0.7483
             precision    recall  f1-score   support

        0.0     0.7930    0.7942    0.7936      7711
        1.0     0.6784    0.6768    0.6776      4947

avg / total     0.7482    0.7483    0.7482     12658
```



Benchmark

As a benchmark, the *LogisticRegression* model turns out to be the best choice for binary classification problems. Easier to model, read and interpret, and train very efficiently.



Model	Training (s)	Predict (s)	Accuracy	Precision	Recall	F1 Score
LogisticRegression	0.108	0.000	0.7483	0.7482	0.7483	0.7482

The trained benchmark has an accuracy of 74.83%, which I think is a good first reference for a new model to give a better result.

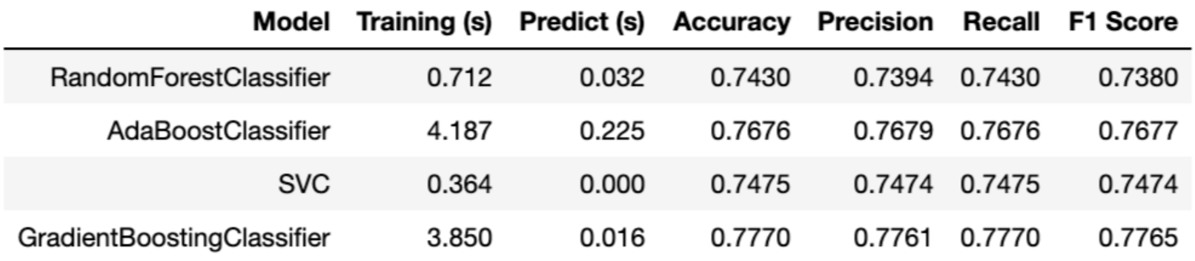
The parameters used for training the model were:

- `solver: liblinear`
- `C: 10`
- `penalty: l2`

Interestingly, the prediction time was 0s, which is an excellent performance, but I don't consider it negligible that a new model has a relatively small processing time.

Results

Considering the collection of results from all models chosen to build the analysis of the best result for this type of problem. The tabulation of the results follows:



Model	Training (s)	Predict (s)	Accuracy	Precision	Recall	F1 Score
RandomForestClassifier	0.712	0.032	0.7430	0.7394	0.7430	0.7380
AdaBoostClassifier	4.187	0.225	0.7676	0.7679	0.7676	0.7677
SVC	0.364	0.000	0.7475	0.7474	0.7475	0.7474
GradientBoostingClassifier	3.850	0.016	0.7770	0.7761	0.7770	0.7765

Justification

All models have good predictive accuracy, so this criterion alone is not sufficient to select the best model.

Since this solution is applied to a very large volume of clients on a daily basis, it is important that the model has a shorter prediction processing time. In this case, the *GradientBoostingClassifier* model has an excellent prediction time (0.016s) and the best precision of all models (77.7%). This model also has the best recall rates and f1 score.

The only negative point is that the training time of this model ends up being longer, but since it is done sporadically, it does not become a problem.

Conclusions

This project provided an opportunity to implement a real recommender system from the ground. It was both interesting and difficult to decide what kind of data to use for analysis, mainly due to the difficulty of understanding well the models used as input.

When using predictive models, we were limited by the number of features we had for the client. If more features were available, we could find more optimal demographic features and contribute to better classification results.

For this first version, my analysis did not include information on transaction amounts. To improve, it might be useful to use predictive models to predict the transaction amount for a user in response to different offers. This information would be very useful to evaluate the different offers, as some offers might have poor conversion rates but could generate large transaction amounts.