Capstone Proposal
Machine Learning Nanodegree - Udacity
Jonatas Oliveira Lima da Silva
Teresina/PI, July 11st, 2021

Project Overview

This data set contains simulated data that mimics customer behavior on the Starbucks rewards mobile app. Once every few days, Starbucks sends out an offer to users of the mobile app. An offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). Some users might not receive any offer during certain weeks.
Not all users receive the same offer, and that is the challenge to solve with this data set.

The task is to combine transaction, demographic and offer data to determine which demographic groups respond best to which offer type. This data set is a simplified version of the real Starbucks app because the underlying simulator only has one product whereas Starbucks actually sells dozens of products.

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.

It will be given transactional data showing user purchases made on the app including the timestamp of purchase and the amount of money spent on a purchase. This transactional data also has a record for each offer that a user receives as well as a record for when a user actually views the offer. There are also records for when a user completes an offer.

Problem Statement

The goal of this project is help sending advertisement and offers to customers more efficiently. To achieve this goal, we take advantage of the transactions and demographics data to determine the offers that should be targeted to different groups of customers.

Metrics

To evaluate the performance of our methodology, was used in this work the accuracy of the predictions. The percentage of the right predictions by the false predictions. A confusion matrix was also defined to know the false positives and true positives. The recall and f1-score was also showed.

Data Exploration

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

The images bellow show how is the content of each data set.

portfolio.json

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0

profile.json

	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0

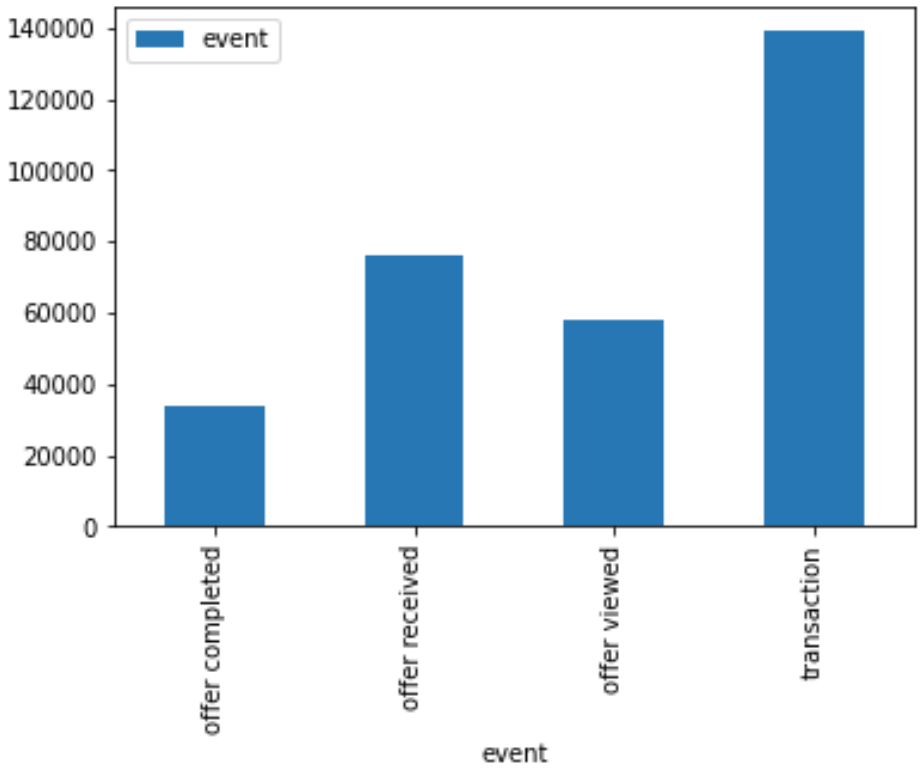
transcript.json

	person	event	value	time
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	0
1	a03223e636434f42ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0

Exploratory Visualisation

As part of this project, before write some code or make some predictions, is necessary understand in deep the data and all the attributes as well as the influence between the attributes.

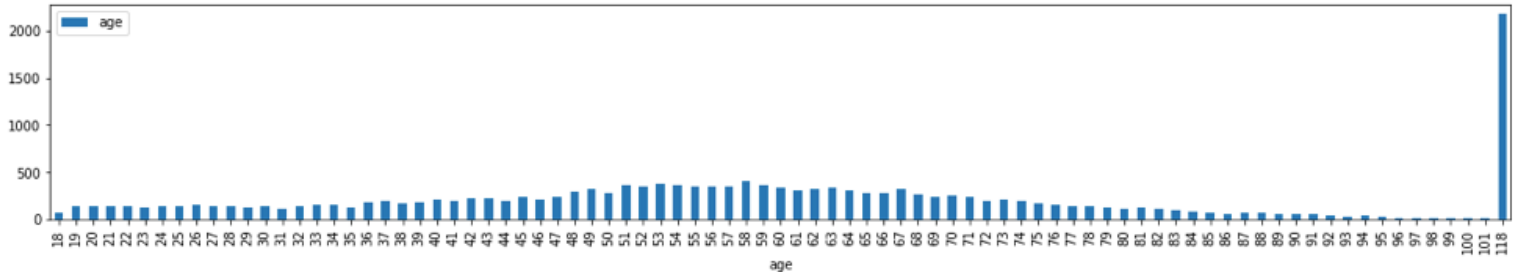
Transactions



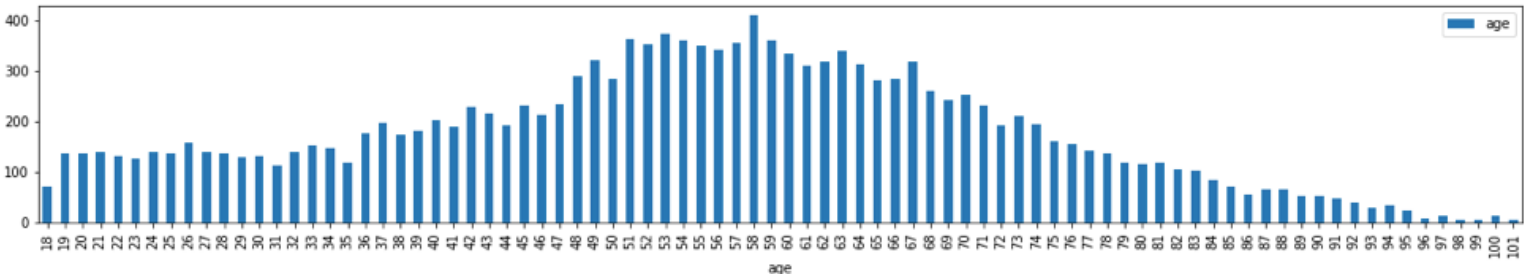
The most important attribute in all datasets is the event, from here we would like to predict the event, and knowing wich offer need be send for some person. But to extract this informations, is necessary to make some manipulation in this data set.

Demographic

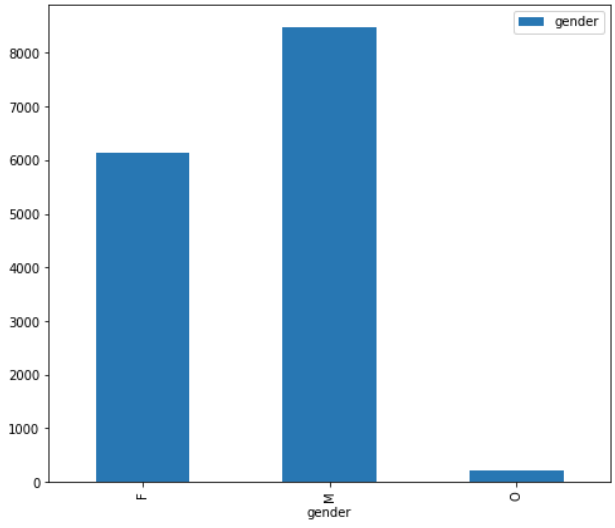
About the demographic, we will study about the gender, age and income.
With the age we have the follow graph, where we can se that we have an outlier.



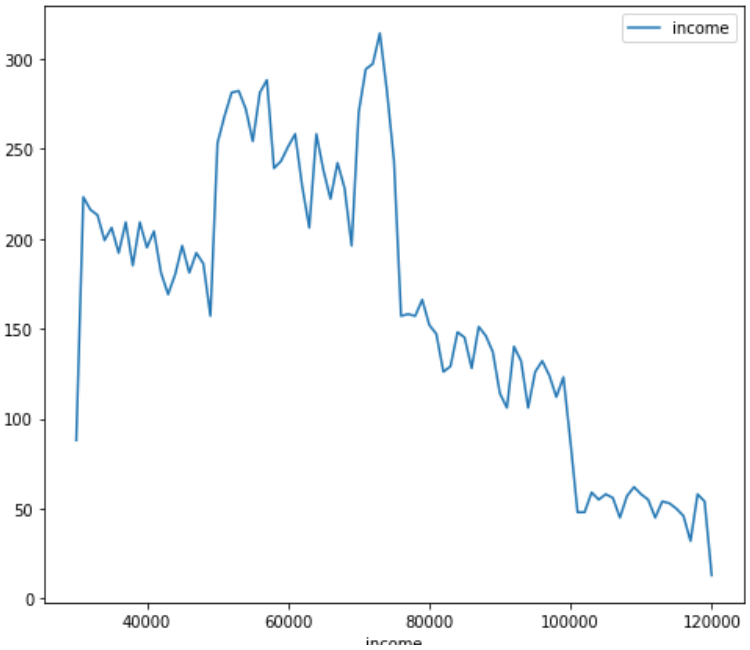
After remove the age 118, we have a more balanced graph as follow:



About the gender we have the follow graph, where we can see the gender M as the most present, and few samples with gender O, more informations about the gender O was not given, but we also use the samples with gender O in our experiments.



About the income we have the values exhibited in the image bellow. The Image shows that the income the be sliced.



Portfolio Offers

Analyse of the offers data. Here we have just 10 elements, thus we didn't need to plot any graph because it is easy to visualise all of them as in the image bellow. With the image we can also concluded that we have 4 different type of channels and 4 different type of offers.

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7
5	3	[web, email, mobile, social]	7	7	discount	2298d6c36e964ae4a3e7e9706d1fb8c2
6	2	[web, email, mobile, social]	10	10	discount	fafdcd668e3743c1bb461111dcafc2a4
7	0	[email, mobile, social]	0	3	informational	5a8bc65990b245e5a138643cd4eb9837
8	5	[web, email, mobile, social]	5	5	bogo	f19421c1d4aa40978ebb69ca19b0e20d
9	2	[web, email, mobile]	10	7	discount	2906b810c7d4411798c6938adc9daaa5

Algorithms and Techniques

Artificial Neural Network

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google’s search algorithm.

A typical training procedure for a neural network is as follows:

- Define the neural network that has some learnable parameters (or weights)
- Iterate over a dataset of inputs
- Process input through the network
- Compute the loss (how far is the output from being correct)
- Propagate gradients back into the network’s parameters
- Update the weights of the network, typically using a simple update rule: weight = weight - learning_rate * gradient

In our project we will use the library Keras.

Random Forest

Random forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of its simplicity and diversity (it can be used for both classification and regression tasks). In this post we'll learn how the random forest algorithm works, how it differs from other algorithms and how to use it.

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

One big advantage of random forest is that it can be used for both classification and regression problems, which form the majority of current machine learning systems. In our case we will use the classification; Let's look at random forest in classification, since classification is sometimes considered the building block of machine learning.

Benchmark

To have some paraters and define a border to our performance, we search at the literature similar works.
Stubseid and Arandjelovic (2018) studied about how consumers make decisions before to buy a product.
Using a large real world data set, they present a series of experiments, analyse and compare the performances of different machine learning techniques, and discuss the significance of the findings in the context of public policy and consumer education. As the result of their work, there are the tables bellow:

The figure below shows the instance structure used in their work:

Transaction #	Feature 1 (C_1)	Feature 2 (C_2)	...	Feature 72 (C_{72})	Consumer decision ($\mathcal{C}$)
1	BC5F4DF1E7	1582934400	...	-0.1216277	No purchase (0)
2	0AB04FC49F	1585612800	...	-0.5361754	Purchase (1)
⋮					
642,709	055D5DBE79	1596153600	...	+0.56486328	Purchase (1)

The figure below shows the results obtained in their work:

Measure	Naïve Bayes	Random forest
Accuracy	0.66	0.72
AUC	0.71	0.79
F1-score	0.66	0.72

Data Preprocessing

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, lacking in certain behaviors or trends, and is likely to contain many errors.

Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing. Data preprocessing is used in database-driven applications such as customer relationship management and rule-based applications (like neural networks). In Machine Learning (ML) processes, data preprocessing is critical to encode the dataset in a form that could be interpreted and parsed by the algorithm.

For example, when we remove the age 118 from the dataset profile.json, we made our first Preprocessing in our data. But some other manipulation is necessary yet before running the algorithm.
About the portfolio was made the follow steps:

- Tranform the offer_type in integers
- Create 4 new attibutes from channels

The final result is show in the image bellow:

	reward	difficulty	duration	id	web	email	mobile	social	offer_type
0	10	10	7	ae264e3637204a6fb9bb56bc8210ddfd	0	1	1	1	0
1	10	10	5	4d5c57ea9a6940dd891ad53e9dbe8da0	1	1	1	1	0
2	0	0	4	3f207df678b143eea3cee63160fa8bed	1	1	1	0	2

About the profile was made the follow steps:

- We already removed the samples with age 118
- Divide the attribute income in 3 groups
- Divide the attribute age in 3 groups
- Transform gender in integers
- Transform atribute became_member_on in days and then, divide it in 3 groups

The final result is show in the image bellow:

	age	id	income	gender	days_became_member
1	2	0610b486422d4921ae7d2bf64640c50b	3	0	1
3	3	78afa995795e4d85b5d9ceeca43f5fef	3	0	1
5	3	e2127556f4f64592b11af22de27a7932	2	1	1

About the transcript was made the follow steps:

- transform event into integers
- remove time with NaN values
- split time into 3 groups
- merge transcripts with same person and offer
  - after merge, define each element as a row

The final result is show in the image bellow:

	person	value	time	event
15561	78afa995795e4d85b5d9ceeca43f5fef	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	1	2
15562	a03223e636434f42ac4c3df47e8bac43	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	1	2
15564	102e9454054946fda62242d2e176fdce	{'offer_id': '4d5c57ea9a6940dd891ad53e9dbe8da0...	1	0

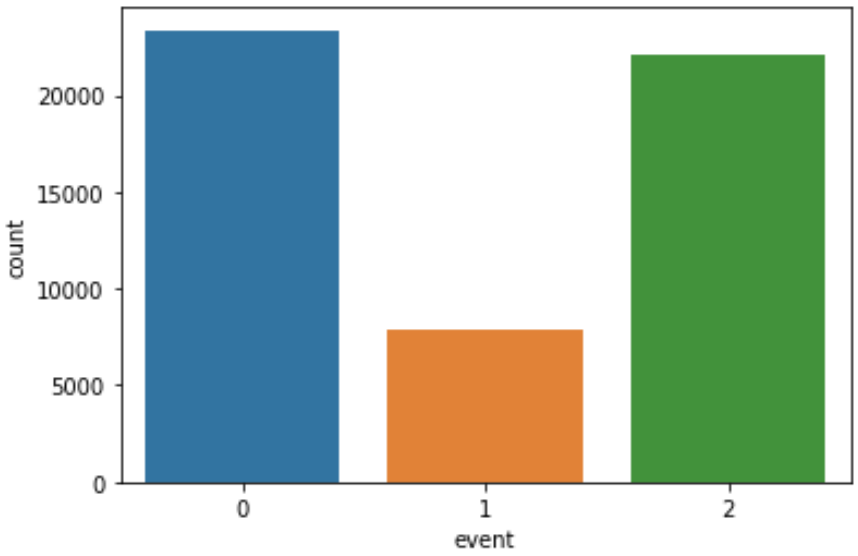
Finally, we need to merge the three datasets (portfolio, profile and transcript. The result is the image below:

	time	event	age	income	gender	days_became_member	reward	difficulty	duration	web	email	mobile	social	offer_type
0	3	0	2	2	1	1	2	10	7	1	1	1	0	1
1	3	1	1	2	0	2	2	10	7	1	1	1	0	1
2	2	1	2	1	1	1	2	10	7	1	1	1	0	1
3	3	2	2	3	0	1	2	10	7	1	1	1	0	1
4	3	0	1	2	1	3	2	10	7	1	1	1	0	1

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 53403 entries, 0 to 53402
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   time                53403 non-null  int64
 1   event               53403 non-null  int64
 2   age                 53403 non-null  category
 3   income              53403 non-null  category
 4   gender              53403 non-null  int64
 5   days_became_member  53403 non-null  category
 6   reward              53403 non-null  object
 7   difficulty          53403 non-null  object
 8   duration            53403 non-null  object
 9   web                 53403 non-null  object
 10  email               53403 non-null  object
 11  mobile              53403 non-null  object
 12  social              53403 non-null  object
 13  offer_type          53403 non-null  int64
dtypes: category(3), int64(4), object(7)
memory usage: 5.0+ MB
```

Implementation One

Firstly for both algorithms we have the follow distribution of the classes:



where

- Class 0 = Offer completed
- Class 1 = Offer received
- Class 2 = Offer viewed

We will use the ANN algorithm and Random Forest. For both algorithms we make the follow steps:

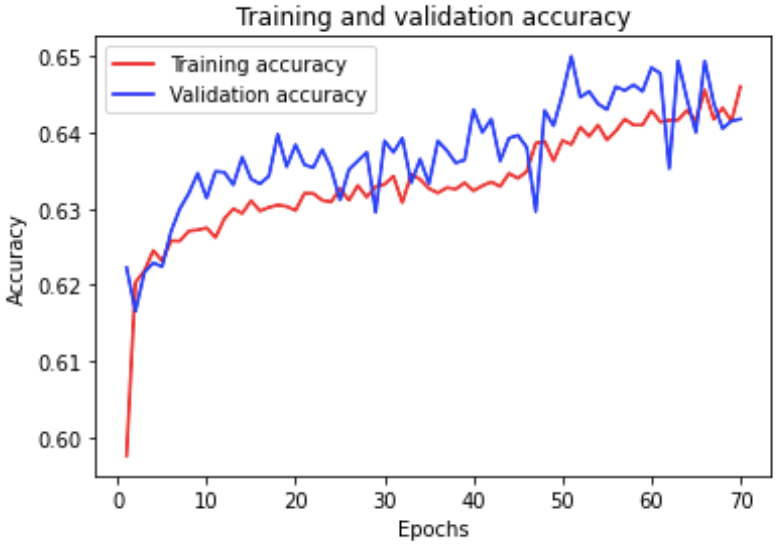
- split into X and Y with 53403 samples of X and Y
- split train and test with 25% of the total to tests.

Algorithm ANN

To the Neural Network with the library keras was configured as follow:

- Number of epochs as 100
- Batch size as 10
- One hidden layer with density 16
- Optimizer adam
- Crossentropy to calculate the Loss
- As metric as defined accuracy

Here we can see the graph of the accuracy of the training and validation.



The final result of the accuracy, recall and f1-score is showed in the table bellow:

	precision	recall	f1-score	support
0	0.64	0.79	0.70	5843
1	0.52	0.37	0.43	1975
2	0.69	0.60	0.64	5533
accuracy			0.65	13351
macro avg	0.62	0.58	0.59	13351
weighted avg	0.64	0.65	0.64	13351

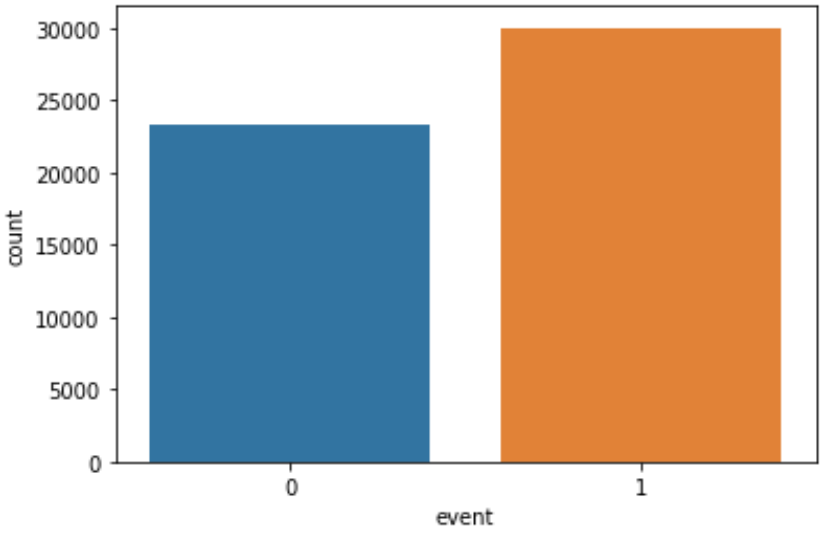
Algorithm Random Forest

With the algorithm random Forest we choose a similar configuration of the ANN. The results with the Random Forest was also very similar to the accuracy of the ANN. And the result can be saw in the image bellow:

	precision	recall	f1-score	support
0	0.64	0.77	0.70	5843
1	0.53	0.39	0.45	1975
2	0.69	0.60	0.64	5533
accuracy			0.64	13351
macro avg	0.62	0.59	0.59	13351
weighted avg	0.64	0.64	0.64	13351

Implementation Two

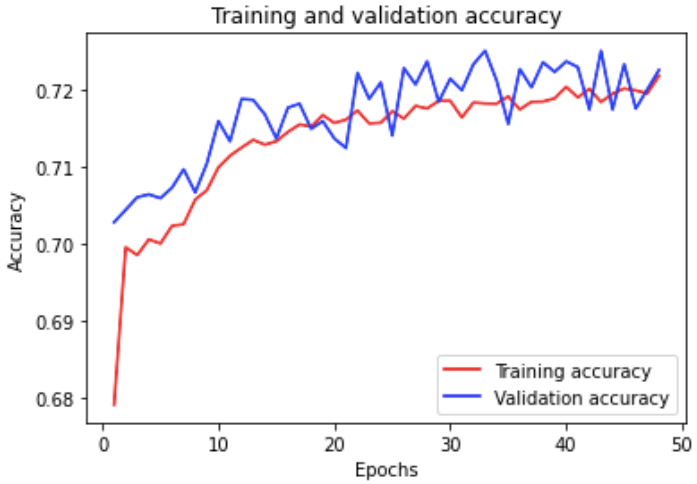
In the second implementation, was decided to mix the classes 1 and 2, because it represents the event viewed and received, and it was not purchased, so we can binarise our problem to know who purchases and who not purchases. The proportion of the two final classes can be viewed in the image below:



The follow steps is equals to the another showed above, to the algorithm ANN and to the Random Forest, so now we will just show the final results for the both approaches.

Algorithm ANN

Here we can see the graph of the accuracy of the training and validation.



The final result of the accuracy, recall and f1-score is showed in the table bellow:

	precision	recall	f1-score	support
0	0.66	0.74	0.70	5843
1	0.78	0.71	0.74	7508
accuracy			0.72	13351
macro avg	0.72	0.72	0.72	13351
weighted avg	0.73	0.72	0.72	13351

Algorithm Random Forest

The result can be saw in the image bellow:

	precision	recall	f1-score	support
0	0.66	0.72	0.69	5843
1	0.76	0.71	0.74	7508
accuracy			0.71	13351
macro avg	0.71	0.72	0.71	13351
weighted avg	0.72	0.71	0.72	13351

Conclusion

This project provided a way to implement a real recommendation system from scratch. The process required the implementation of all the steps of the data science methodology. It was interesting as well as difficult to decide what kind of data to use for the analysis. The project could have taken different directions if other aspects of the data were taken into account.

Another recommendation systems could have been explored. Modeling the data could have been another choice to recommend offers. The mathematical model would have needed considerable adjustments in order to be used in a production systems.

To improve the recommendation system, we could include other metrics. For instance, the frequency at which offers are completed would add an interesting dimension to the system.

To resume, what we did was:

- Study the problem in depth
- Search in the literature to similar works and find out how other research solved similar problems
- Produce a Data Visualizations of the attributes
- Preprocess the data, removing outliers, noises and incomplete informations
- Merge the data sets
- Produce three class to predict

- Implement an ANN with to solve the final dataset
- Implement a Random Forest to solve the final dataset

References

- https://www.ibm.com/cloud/learn/neural-networks
- https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html
- https://www.techopedia.com/definition/14650/data-preprocessing
- https://towardsdatascience.com/pytorch-tabular-multiclass-classification-9f8211a123ab
- https://medium.com/luca-chuangs-bapm-notes/build-a-neural-network-in-python-multi-class-classification-e940f74bd899
- https://builtin.com/data-science/random-forest-algorithm
- Stubseid, Saavi & Arandjelovic, Ognjen. (2018). Machine Learning Based Prediction of Consumer Purchasing Decisions: The Evidence and Its Significance.