I. Definition

Project Overview

In a world where advertising is becoming increasingly competitive, it is becoming also more and more important to target your audience accurately in order to ensure your business can continue to thrive.

With the recent advancements in computing, machine learning algorithms are now largely available in a number of frameworks that makes it easy for everybody to train and use. Leveraging this new technology to target your customers accurately is nowadays paramount.

Starbucks is a coffee company that was founded in 1971 in Seattle and despite his strong presence is the market, the recent pandemic hit the sector quite strongly.

Starbucks reported his first quarterly loss in 7 years (source: https://www.ft.com/content/4c876c5a-a03a-3f57-8a04-c364b12ad4b7) and many bars and restaurants had to close down as their profitability was strongly undermined by the social distancing guidelines put in place by world governments.

Problem Statement

Starbucks wants to improve the way it proposes deals and offers through the app to its customers.

The aim of this project is to leverage machine learning algorithms to increase revenue by targeting the right audience and drawing them to the stores.

The goal of this project is to analyse historical data to draw insights and create a model to better target customers.

Metrics

For the issue at hand we are going to use accuracy as metric to evaluate the machine learning algorithm.

The problem at hand is a binary classification problem, and the dataset at hand is balanced.

Furthermore, the business does not incur any severe consequence in case of a false positive or a false negative.

II. Analysis

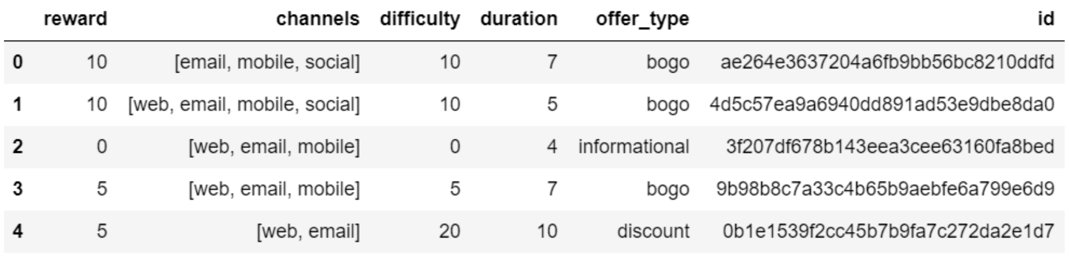
Data Exploration

In this first section we are going to explore the 3 datasets at hand.

We will also use plots to get a sense of what data is avaiable and how we might want to use it.

The goal of this phase is to get an insight on the datasets at hands and identify potential issues to tackle.

Portfolio Dataset

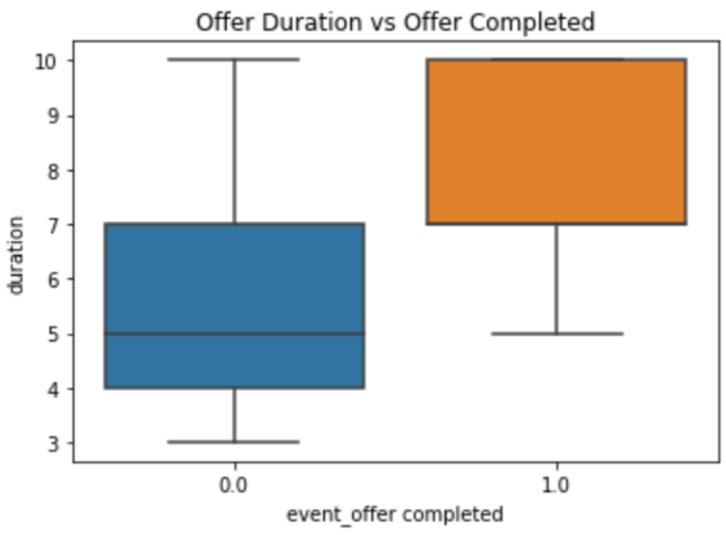


	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7

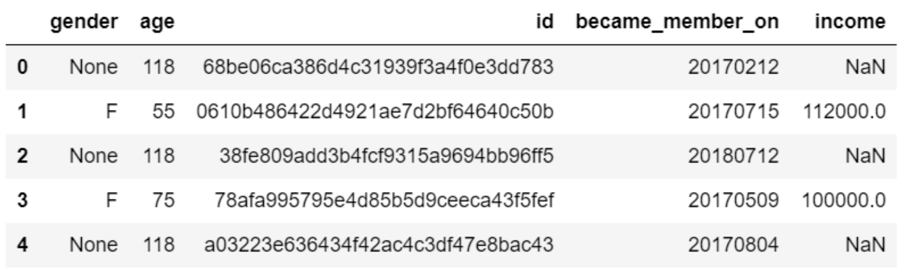
The portfolio has information about the offers Starbucks put forward. We see that 2 columns need to be one-hot encoded: Channels and offer_type.

The following are the offer types: email, mobile, social, web.

Some of the features inside the Portfolio dataset are highly informative for the problem we are trying to tackle, for instance the higher the duration, the more likely the offer is completed.



Profile Dataset

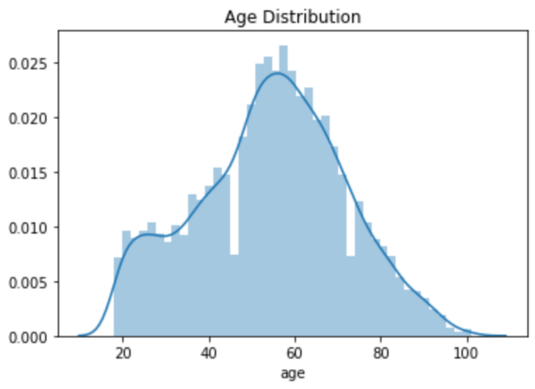


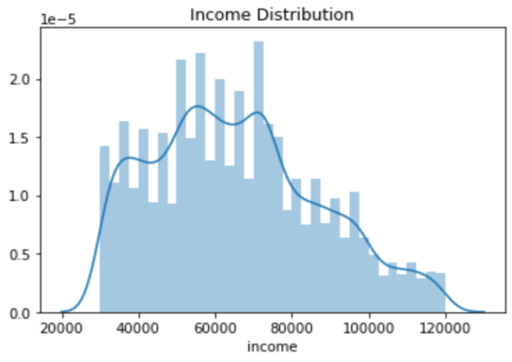
	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43	20170804	NaN

We have a number of missing values and all of them coincide with age 118. We have removed those.

We will need to one hot encode gender andtransform the variable "became_member_on" to something that tells us the customer tenure, such as number of days since customer joined.

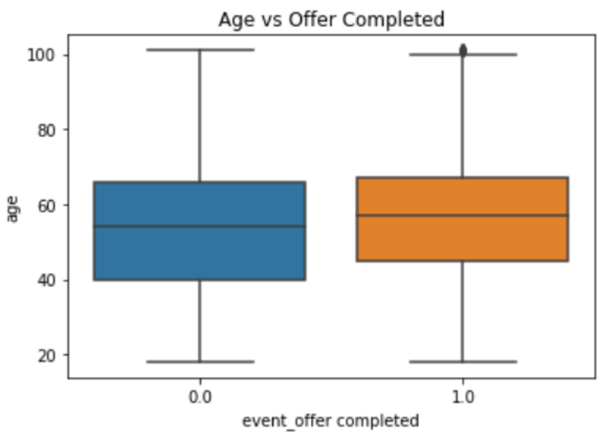
The following are the distributions of age and income with median values 55 and 64k respectively:

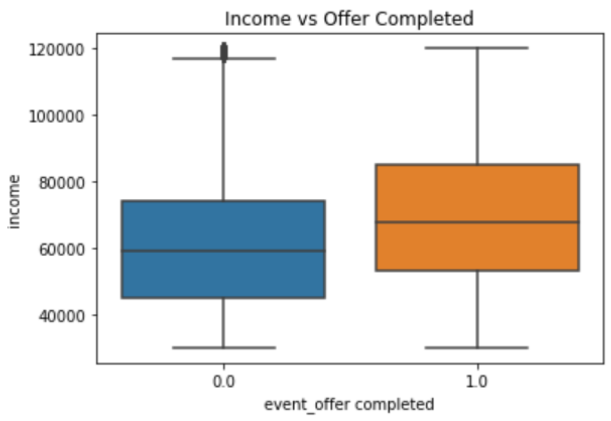




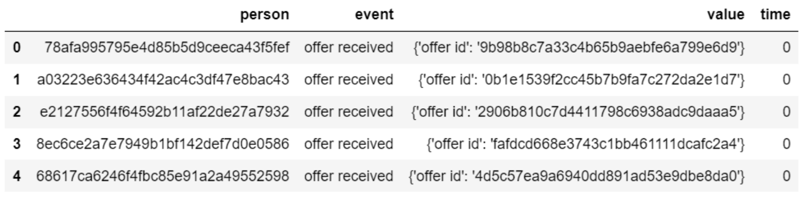
Income appears to be more influential than age in separating whether `an offer is completed.

Higher Income increases the chances of completing the order.





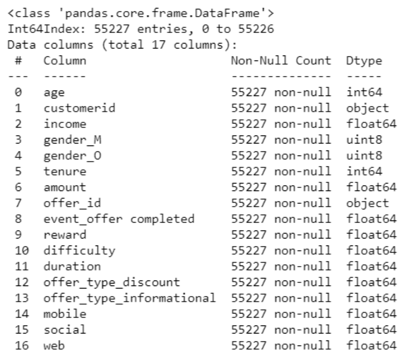
The transcript dataset is essentially a mapping between the event and the offer



	person	event	value	time
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	0
1	a03223e636434f42ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0
2	e2127556f4f64592b11af22de27a7932	offer received	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}	0
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer received	{'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}	0
4	68617ca6246f4fbc85e91a2a49552598	offer received	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	0

Combined dataset

This is what the dataset looks like once we have preprocessed them and combined them:



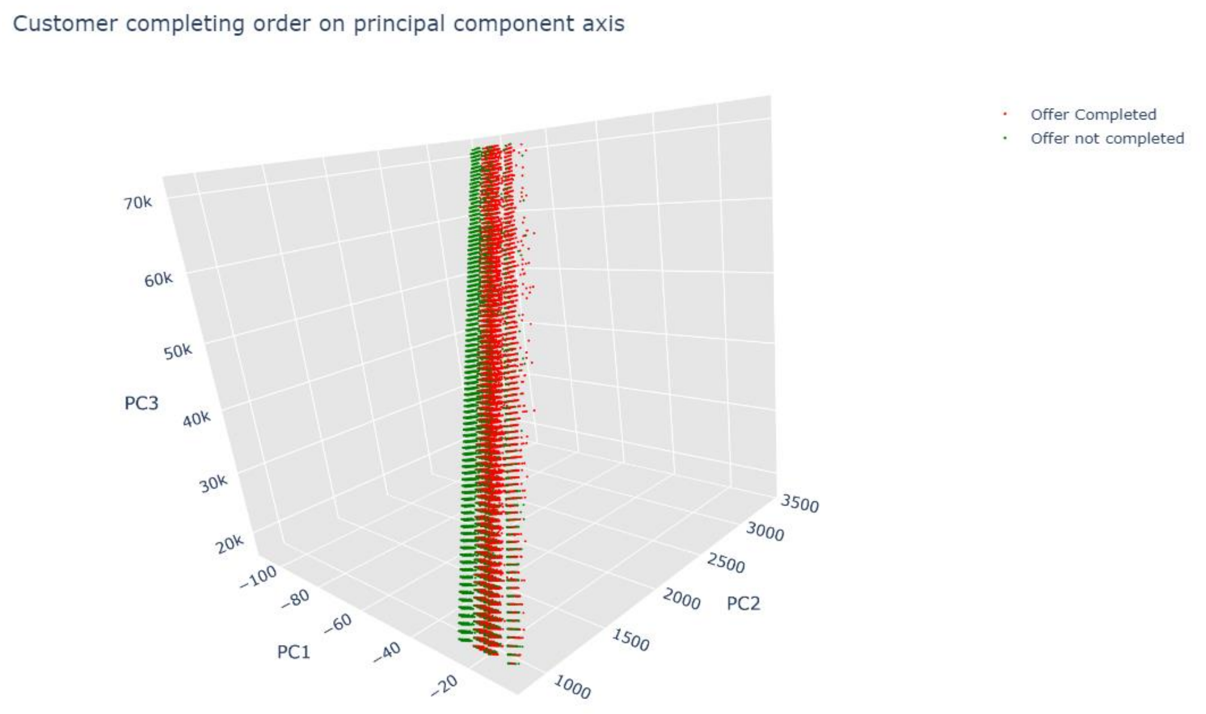
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 55227 entries, 0 to 55226
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   age                      55227 non-null  int64
 1   customerid               55227 non-null  object
 2   income                   55227 non-null  float64
 3   gender_M                 55227 non-null  uint8
 4   gender_O                 55227 non-null  uint8
 5   tenure                   55227 non-null  int64
 6   amount                   55227 non-null  float64
 7   offer_id                 55227 non-null  object
 8   event_offer completed    55227 non-null  float64
 9   reward                   55227 non-null  float64
 10  difficulty               55227 non-null  float64
 11  duration                 55227 non-null  float64
 12  offer_type_discount      55227 non-null  float64
 13  offer_type_informational 55227 non-null  float64
 14  mobile                   55227 non-null  float64
 15  social                   55227 non-null  float64
 16  web                      55227 non-null  float64
```

The dataset has now many dimensions and to further explore the separation between orders completed or not across all dimensions we are going to leverage a dimension reduction technique, Principal Component Analysis.

Using this technique, we can reduce the dimensionality of the data to 3 for the purpose of the visualization, retaining most of the variability in the original dataset.

Due to the high presence of categorical features, the points are sort of clustered towards the central axis.

But we do see a clear separation between the green and the red points, but also an area where they strongly overlap. We will need a strong classifier in a hyperdimensional space to achieve high accuracy.



III. Methodology

Data Preprocessing

The following pre-processing steps have been carried out:

1. One Hot encode the offer_type column from the portfolio dataset
2. One Hot encode the channels column from the portfolio dataset
3. One Hot encode the gender column from the profile dataset
4. Transform the became_member_on column into tenure days
5. Remove missing values from profile dataset
6. Remove events not offer related from transactions dataset
7. Extract offer ID from value column in transactions dataset
8. Reshape the transaction dataset to have the event type as one hot encoded column
9. Extract total amount spent from transactions for each customer
10. Create a column to mark whether an offer was successful or not
11. Join the 3 datasets and create a unique dataframe

The one hot encoding was necessary as algorithms only read numerical data. No data scaling was performed at this stage as the 2 algorithms that we have selected are tree based, random forest and XGBoost. They are not distance based algorithm such as SVM or KNN, or neural networks, so scaling is not necessarily required.

We have then split the dataset into training and test set using a 85:15 split.

Implementation

For the implementation we have leveraged the scikit learn library. It provides ready to use implementation of the algorithms we have chosen.

In order to evaluate them, we chose to use cross validation techniques.

Fitting the algorithms on the training set alone can lead to overfitting and it’s important to see how these algorithm generalize.

We have therefore leveraged stratified cross validation and obtained the mean accuracy achieved on the out of sample prediction.

Refinement

We then proceeded to fine tune the hyperparameters of XGBoost and Random Forest to improve the generalization on the test set.

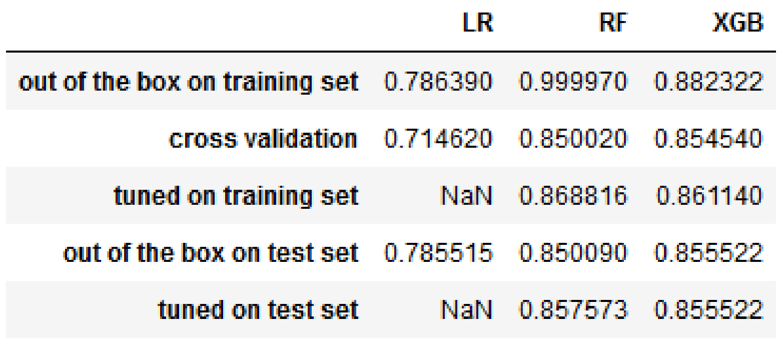
Hyperparameter tuning has been implemented using Randomized search over a set of combination of parameters. 50 combinations have been searched and the best has been fitted.

We have then fitted the tuned models and evaluate their performance on the test set.

IV. Results

We have evaluated the accuracy of our models using a test set, data that the model has not trained on and that represents the performance on out of sample data.

The following are the accuracies achieved by the models we have trained:

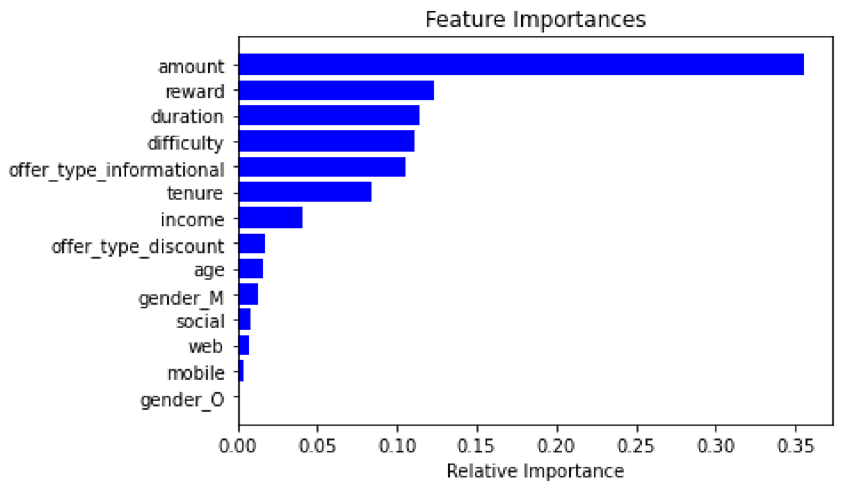


	LR	RF	XGB
out of the box on training set	0.786390	0.999970	0.882322
cross validation	0.714620	0.850020	0.854540
tuned on training set	NaN	0.868816	0.861140
out of the box on test set	0.785515	0.850090	0.855522
tuned on test set	NaN	0.857573	0.855522

The best performing model is RandomForest classifier, with the tuned hyperparameters. We reached this conclusion after looking at the accuracy it achieved on the test set, which is 85.7%.

The performance of this new classifier far outperforms that of our benchmark, logistic regression which achieved 78.5% accuracy on the test set and it is also higher than the current performance of the starbucks model, where only 59% of the offers gets completed.

Furthermore, our recommendation is to focus on the amount,reward and duration of the offer as they the most influential features in determining whether an offer gets completed or not. See Feature importance extract of the tuned classifier below:



V. Conclusion

In this project I have imported,cleaned and analysed a dataset provided by Starbucks.

We have engineered the data to extract features for the algorithms, and I was able to create a Supervised Learning model with an accuracy of 85.7% with a big improvement over the current 59% success rate of the model employed by Starbucks.

This is a considerable achievement as this model could boost revenue if used correctly to attract customers.

Further Steps:

It is important to note that the possibility with a dataset are endless and there are many more things that we could have tried to do. I hereby list a few:

1) check other classification algorithms

2) fine tune the models again by searching for longer to find better hyper parameters

3) perform further feature engineering to create more features that might help the algorithm increase accuracy

4) perform error analysis by looking at false positives or false negatives to understand why they were wrongly classified

5) try more complex models such as neural networks