STUDENT: Damir Valput

Nanodegree program: Machine Learning Engineer

CAPSTONE PROJECT REPORT: Starbucks Project

I. Definition _(approx. 1-2 pages)_

Project Overview

The problem I will solve in this capstone project is one of the three proposed ones dealing with Starbucks simulated data set.
The problem domain is in the field of targeted marketing (i.e. providing targeted offers), an area in which through data analysis and predictive modelling we wish to better understand different customer profiles and what types of offers they respond to. Many companies that offer specific products or services benefit from using targeted marketing strategies. Presenting customers with targeted offers is very beneficial to businesses as it encourages customers to choose that company over a competitor, increases chances that people spend money on a certain product, and yields more focused marketing efforts in terms of a higher ROI (Return on Investment) [1]. Customers should be targeted in the appropriate and relevant manner: too intensive targeting can result in their fatigue, whereas too weak advertising can result in profit loss or missed opportunities.
Data science, and artificial intelligence techniques in general, have been successfully applied in marketing for some time now, ranging through a variety of applications: lead targeting, customer segmentation, sentiment analysis, pricing strategy, product development, etc. [2] Data science and machine learning in targeted marketing are particularly useful as they can provide insights about customers' behaviour and decision making beyond what is intuitive to a human brain, which consequently allows business expansion into new markets or places. [2]
I find this problem particularly interesting as it focuses on applying machine learning to a problem in an area that is novel for me, and thus I believe it will enable me to learn a great deal of new concepts and techniques.

Problem Statement

In this problem, I am presented with three different datasets on Starbucks customers and the transactions they made in the past. My task is to combine transactional, demographic and offer data to determine which demographic groups respond best to which offer type.

I intend to perform market segmentation of the customers in the database, i.e. customers will be categorised into groups (clusters) with respect to their historical purchasing behaviour and demographic characteristics. This will be done taking into account solely the information on their historical transactions (recorded in the database) and their general socio-economic characteristics, and not taking into account how they respond to offers as my main idea with this task is to achieve a sort of baseline clustering of customers that could later be used in the task of predicting how they react to offers. For this task, I will employ a k-means clustering algorithm. Therefore, upon market segmentation, I will design a predictive model that answers the question whether a particular customer will respond to a certain type of offer (and consequently, whether an offer should be sent to them). To do this, I will implement a binary classifier using the XGBoost model.

Metrics

In the unsupervised task of customer clustering using k-means, I will rely on the elbow method [4] to determine what is a good number of clusters (a "good k"). In the supervised task of binary classification, I will use a number of standards evaluation metrics to measure the performance of my trained classifier and compare it to the performance of a benchmark model: accuracy, confusion matrix, recall, precision, F1 score, ROC AUC and AUC PR. As the dataset in question is fairly balanced, I'll use accuracy as the optimising metric (the one that is optimised during training), and the other metrics are going to be used as "satisficing metrics" with the goal of improving them over the benchmark model.

II. Analysis _(approx. 2-4 pages)_

Data exploration

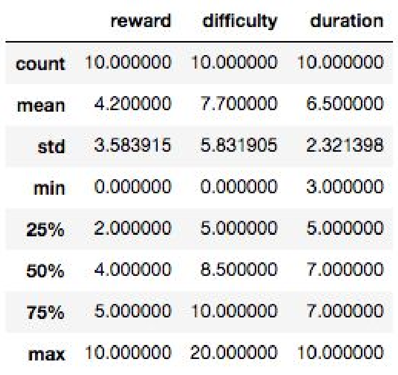
The dataset for this project was provided by Starbucks and it consists of simulated data that mimics customer behavior on the Starbucks rewards mobile app. These data are a simplified version of the real Starbucks app data because the underlying simulator only has one product, whereas Starbucks actually sells dozens of products. The dataset is structured (tabular data) and contains these three tables:

- "portfolio" - contains metadata on different offer types provided by Starbucks;
- "profile" - contains demographic data for each customer registered in the app, e.g. age, gender, income, etc.;
- "transcript" - consists of records of transactions, offers received, offers viewed, and offers completed for each customer in the "profile" table.

Each table is in .json format. The data will be loaded, cleaned and analysed via a standard exploratory data analysis process, with the objective of having them ready for their usage in the feature engineering process.
The data provided are fairly clean and almost ready to be used as they are. During the initial data, only a few anomalies and cleaning needs were identified (presented below). In continuation each of the three tables is shortly presented, usefulness of the table for the problem at hand is discussed, with references to useful features that could be extracted from each, and I present some descriptive statistics calculated for the dataset.

Portfolio

The table containing the portfolio of offers, with the main features of each offer. It is a small table with only 10 entries (i.e. 10 different offer types), and for each offer type the following information is provided: offer id, offer type, difficulty of the offer, reward given completing the offer, and the duration (in days) for which an offer is going to be open. Additionally, a list of channels through which the offer is sent to the customer is given (email, web, etc.). In the portfolio given in this exercise, there are three possible offer types: 'buy one, get one' (BOGO), discount or informational. The basic descriptive statistics of these data are shown in the following Table: 1

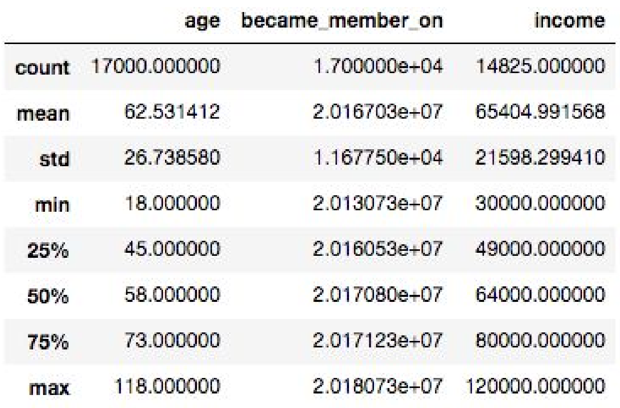


	reward	difficulty	duration
count	10.000000	10.000000	10.000000
mean	4.200000	7.700000	6.500000
std	3.583915	5.831905	2.321398
min	0.000000	0.000000	3.000000
25%	2.000000	5.000000	5.000000
50%	4.000000	8.500000	7.000000
75%	5.000000	10.000000	7.000000
max	10.000000	20.000000	10.000000

Table 1. Descriptive statistics of the table 'portfolio'

Profile

The table 'profile' contains the demographic data on the customers in the database, consisting of in total 17,000 different customer profiles. The demographic data contained in this table are not very extensive, but it should be sufficient to obtain some fundamental customer clusters: we have the data on the age, membership duration, gender and income of the customers. Mean income of the Starbucks customers is around 65,000$, and the mean age is quite high (62.53) (see Table 2. for descriptive statistics). However, I have discovered some outliers/anomalies in this table - a significant number of customers having age of 118 years and all of those having incomplete data rows (in fact, no other information but the age). Those will be removed as anomalies in the process of data cleaning.



	age	became_member_on	income
count	17000.000000	1.700000e+04	14825.000000
mean	62.531412	2.016703e+07	65404.991568
std	26.738580	1.167750e+04	21598.299410
min	18.000000	2.013073e+07	30000.000000
25%	45.000000	2.016053e+07	49000.000000
50%	58.000000	2.017080e+07	64000.000000
75%	73.000000	2.017123e+07	80000.000000
max	118.000000	2.018073e+07	120000.000000

Table 2. Descriptive statistics of the table 'profile'

Transcript

The table transcript contains information on different events recorded during the test, starting with event type (which can be a transaction, offer received, offer viewed, and offer completed). For each recorded event, there is the id of the customer who performed it, time in hours since the start of the test when the event was recorded (which will be useful to determine which offers were successfully completed) and a field named 'value' that, in general, can have two different types of information:

- Amount the customer spent if the event recorded is a transaction

- An offer id if the event if related to an offer (received, viewed, completed), with the reward obtained in case of completed offers

In total, there are 306,534 recorded transcripts.

For each of the above datasets, through exploratory data analysis has been run using library “pandas-profiling” in order to understand the characteristics of the data.

Exploratory Visualization

In this section, I provide some forms of visualization that summarizes more relevant characteristics about the data. In Figure 1 you can find a visualisation of the Pearson’s correlation coefficient for the numerical fields in the table ‘profile’.

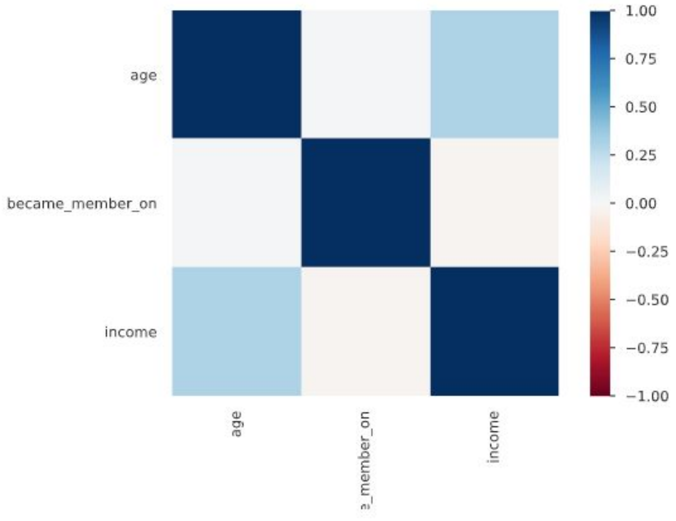


Figure 1. Pearson’s correlation coefficients: table ‘profile’

We can observe that age and income, as one would expect, have a weaker positive correlation with each other, while became_member_on is weakly negatively correlated with income and age. This means that there is a slight tendency that customers that joined more recently the Starbucks programme are slightly younger and naturally of lower income.

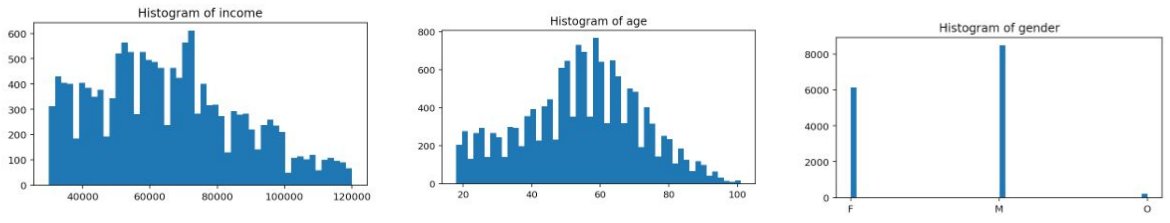


Figure 2. Histogram of customers’ gender, age, and income

From the histograms on Figure 2, we can observe that most customers declared as male (around 57%), around 42% are females, and the rest declared as “Other” (212 profiles). The largest number of customers is in the age range of 50 and 65 years

(the peak of the age histogram is around 60 years), followed by a demographic og younger customers. Both age and age group of a customer might be important features. Finally, the income is fairly uniformly distributed up to around 80k a year, after which it drops sharply. Most customers seem to be in the middle income range. All of these features, as well as the other complex features engineered from these profile characteristics, will be important to cluster the customer profiles and try to predict how they would react to a certain offer.
More visualisations using the processed data and engineered features are presented in the following sections.

Algorithms and Techniques

The project follows a standard machine learning workflow: from data loading and exploration to developing and evaluating two different machine learning models.
Firstly, an unsupervised clustering algorithm, concretely, k-means, has been employed to perform market segmentation, i.e. to cluster the customer into similar profile groups based on their demographic and purchasing characteristics. This has been done without taking into account how customers react to offers to capture only information on the different fundamental demographic clusters of customers.
In the second machine learning task, a supervised learning algorithm was developed and validated with the objective of predicting whether a particular offer to a certain customer would be successful, i.e. whether that offer should be sent to the customer. This is a binary classification problem. For this ML task adequate features were engineered and selected based on correlation analysis.
Moreover, the dataset is split into three parts to ensure proper model validation: train, validation and test sets. [3] The first two are used in model training, and the test dataset is used in the evaluation of the model.

Benchmark

A benchmark model for the supervised learning problem is designed to compare the developed binary classification model to. The benchmark model is in essence a random generator that decides whether a client will respond positively to an offer randomly without taking into account any information on the client or the offer, with the probability p of the positive outcome (completed offer). The probability p will be selected in such a way that it reflects the proportion of the positive examples in the original datasets, i.e. it adequately represents the sample given in the database. The performance of the benchmark model will be measured using the same set of evaluation metrics as for the defined solution.

III. Methodology _(approx. 3-5 pages)_

Data Preprocessing

In the data processing phase, my main efforts were focused on removing the anomalous data samples encountered in the data exploration phase, transforming some columns in provided data tables so that useful features can be extracted (e.g. one hot encoding), feature engineering, analysing and visualising those features, and labelling of the data for the binary classification problem (i.e. defining the ground truth).

First of all, 2175 profiles in the profile table were found to be anomalous since their information was incomplete (missing values in most of the fields) and their age was 118. I decided to discard them from further analysis as outliers/anomalies. That left me with only 5 female profiles aged over 100 (concretely, age 101) who did have other fields defined (not missing values), for example, they had varying incomes.

Moreover, data in certain columns had to be transformed so that it can be easily used in feature engineering and data labelling:

- In the table 'portfolio', the information in the column *channels* is given as a list of strings. I processed that column by defining a new column for each of the possible channels (email, mobile, social, web) and one-hot encoding the information provided in the column *channels* in the newly defined columns.
- Also in the table 'portfolio', I one-hot encoded the information on offer type in the same way.

In the table 'transcript', some more extensive processing has been done on the columns *event* and *value* so to extract information on whether an offer has been received, viewed or completed, and to obtain information on the transactions that the customers performed. Since the dataset didn't come with clearly defined ground truth, I had to perform the labelling myself. To do that, I set up a certain ground rules based on the following observation: Offer types 'bogo' and 'discount' have events 'offer received', 'offer viewed' and 'offer completed', but offer type 'informational' has only events 'offer received' and 'offer viewed'. Therefore, in this problem, for offer type 'bogo' and 'discount', I will consider the offer completed when there is an event 'offer completed' **within the offer duration time**. The offer type 'informational' I will consider completed if it has been viewed by the customer. Following those rules, I processed the data in the corresponding columns and extracted information on received, viewed and completed offers. This information is used to define the ground truth for the binary classification problem.

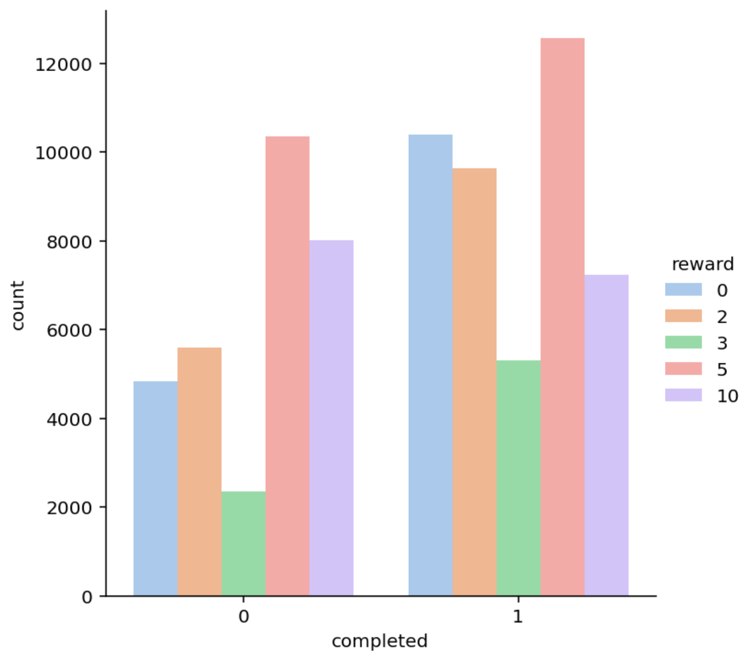


Figure 3. Number of completed (1) and not completed (0) offers grouped by rewards

It is interesting to observe (see Figure 3), upon the data labelling, how completion of an offer changes with respect to the reward offered. Offers with highest rewards do not have the highest completion rate, which is probably influenced by the difficulty of such rewards. In fact, offers with lowest rewards (from 0 to 3) have much better completion rates.
In the process of feature engineering, I created some more complex features relying on the data provided on customers and their purchases in the tables 'profile' and 'transcript'. These features will be used in both tasks of customer segmentation and binary classification. They encompass demographic features of the customers such as relation of their income to their age, age group, the duration of their Starbucks membership, etc., as well as some features that capture their spending patterns like average amount they spend per transaction, buying frequency (how often do they make a purchase), etc. The full list of these features and those selected for the modelling tasks can be found in the section on Implementation, when discussing features that are going to be selected.

Upon data cleaning, processing and labelling, in the final obtained set of examples I got 76,277 received offers, out of which 58,385 have been labelled as viewed, and 45,135 have been labelled as completed. That means that around 59% of the offers have been completed, the value that will be used as p in the benchmark model. That also means that the data for the binary classification model is fairly well balanced, i.e. I don't have a case of imbalanced data for binary classification. Figure 3 shows that most offers received by customers are BOGO and discount offers, and Figure 4 shows that information offers have in general the highest completion rate (almost 70%), whereas BOGO offers have the lowest completion rate of about 53%.

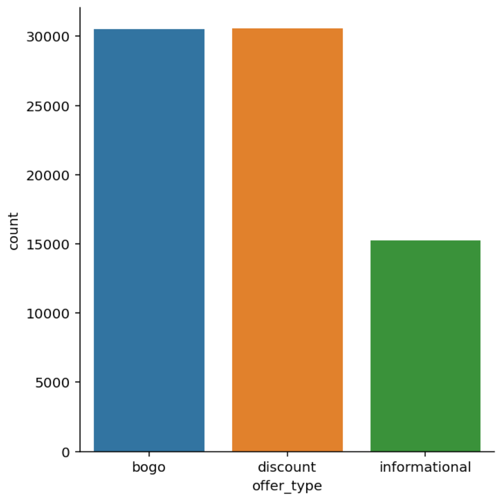


Figure 4. Number of different offer types among received offers

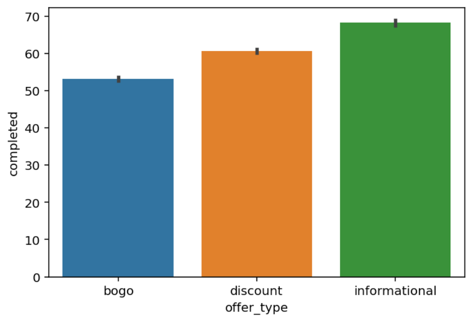


Figure 5. % of completed offers by each offer type

Implementation

In this section, I discuss the implementation details of the two models I tried in this project: k-means customer clustering and binary classifier, as well as the method used to split the data for the model training and evaluation.

k-Means clustering for customer segmentation

For the first task, my intention was to use k-means clustering to segment the customers using their profile features on gender, age, income, Starbucks membership, etc. With this, I wish to obtain a division of customers into baseline groups according to their basic purchasing patterns (excluding the data on how they react to offers) and demographic and socio-economic characteristics. That way, each customer will be assigned to their “customer group” that could be used for various purposes: to better understand different groups of customers in Starbucks, to label them for different marketing purposes, etc.
To choose a "good k", I relied on the elbow method. Since this is an unsupervised learning method, there is nothing to test since there is no ground truth, and only the elbow method will be used to fine-tune the model. The range of values of k that has been tried out is from k=4 to k=15. Due to a fairly small feature space (around 10 features), the clustering is performed directly on the created features without using dimensionality reduction (PCA). The results of the clustering are presented in the

following section on results. Upon the clustering, a label of the cluster is assigned to each customer in the database.

Feature selection and data splitting

To prepare the final dataset for model training and evaluation, I performed correlation analysis (see correlation matrix on Figure 6) and I split the data into three sets: train, validation and test. To create the table of features for training, I dropped irrelevant features such as person or offer id, or features that could cause data leakage such as the column ‘viewed’. Correlation analysis revealed that the column ‘email’ is always 1 and thus it can be dropped. Other than that, all of the remaining features have comparable correlation coefficients with the target (well below 0.5), so I decided to train the binary classifier.
Since around 59% of the training examples are positive, I can conclude that I have at hands a pretty well balanced data set. Nevertheless, when splitting the data for training and testing, I will honor the class representation and perform stratified sampling to make sure that the shares of each class in each of the subsets (train, test, val) reflects their shares in the data. The features have also been scaled to (0,1) range.

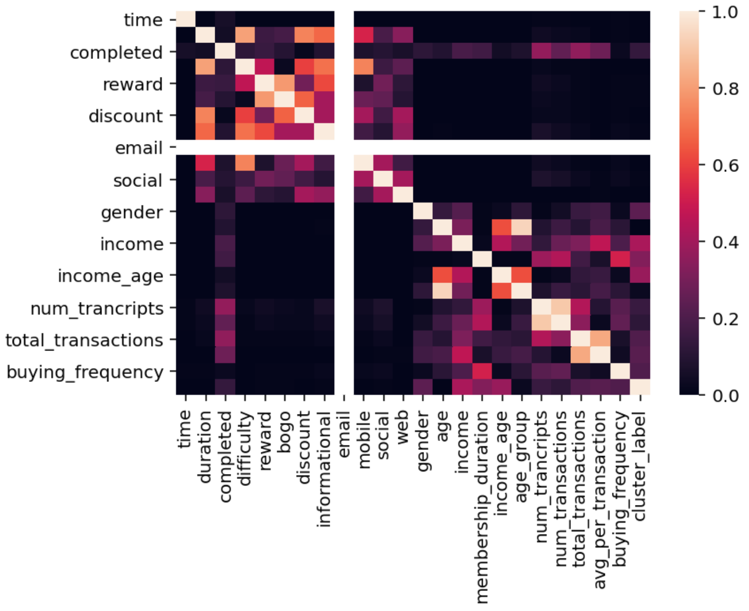


Figure 6. Feature correlation matrix

Binary classifier for predicting success of an offer

To implement a binary classifier that tries to predict whether an offer will be completed or not, I chose to rely on the XGBoost model due to its remarkable performance demonstrated across a number of Kaggle competitions and real-world problems [5]. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance, very suitable for both classification and

regression problems [6]. Today it is often one of the strongest tools in the arsenal of a data scientist. To implement it, I used a built-in algorithm provided by Amazon Sagemaker [7]. Their implementation has a smaller memory footprint, easy to implement hyperparameter tuning and a very large set of available validation metrics. That makes the implementation fairly easy and convenient, albeit giving up on some of the flexibility that I would get when not using a built-in algorithm. However, since the problem I have at hands is a very standard binary classification problem, I believe this approach will give me a very good performance without needing to invest a lot of time to implement and tune the model.
The hyperparameter tuning and deployment is described in more details in the next section on refinement.

Refinement

Upon the definition of the XGB estimator, a Hyperparameter Tuner object has been defined with a set of hyperparameters that I wish to tune: max_depth, eta, min_child_weight, subsample, gamma and num_round. Number of models to construct (max_jobs) has been set to 15, with at most 3 models trained in parallel. This has taken 20-30 minutes to execute, and in future a wider hyperparameter search could be performed to see if the current results can be improved. However, for this exercise, I decided to go forward with the obtained results since they were pretty satisfactory over the benchmark model's results (see the following section on Results).
The hyperparameter tuning has been performed using the accuracy metric (considering this is a binary classification problem with balanced data) on the validation data set to compare the performance of various trained models. I also tried ACU and ACU PR as objective metrics, and very similar results were obtained.
Lastly, the model from the best training job has been selected. That model achieved the accuracy of around 83% of both training and validation datasets, after 68 rounds (early stopping is implemented after 10 training rounds). The trained model was first tested using SageMaker's Batch Transform functionality just to make sure that it is working fairly along the lines of what was expected before deploying it to an endpoint, to which I sent the test data in order to evaluate the model (see the Evaluation section).

IV. Results _(approx. 2-3 pages)_

Model Evaluation and Validation

k-Means clustering for customer segmentation

Since customer clustering using k-Means is an unsupervised learning method, the testing and validation can not be done as in the case of supervised methods. However, the model itself can be evaluated using the elbow method, as I described in the section on implementation. Upon training k-Means for various k, I obtained the elbow graph as shown on Figure 7.

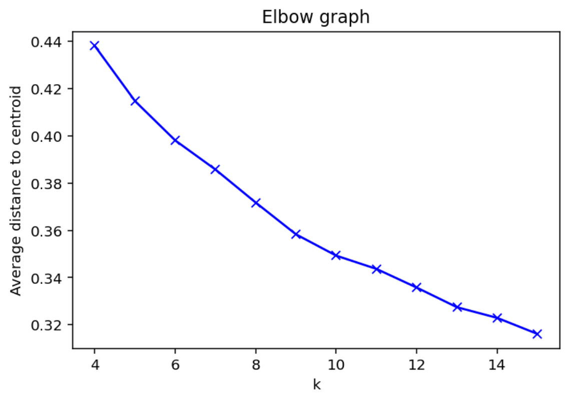


Figure 7. Elbow graph for k-Means customer clustering

The elbow graph can help us determine what would be a **good k** to choose for clustering our data. While the graph I obtained doesn't have a very distinctly visible "elbow", i.e. there is not a super clear k for which the centroid distance stops decreasing at a very sharp rate, I can observe that after k=10 the decrease noticeably slows down. Therefore, I decided to select a k=10 as a "good value of k". This might be due to a fairly small feature space that I had to work with regarding customer demographic data, and thus I need to have in mind that this kind of elbow graph might be indicative of clusters that are not very distinct.

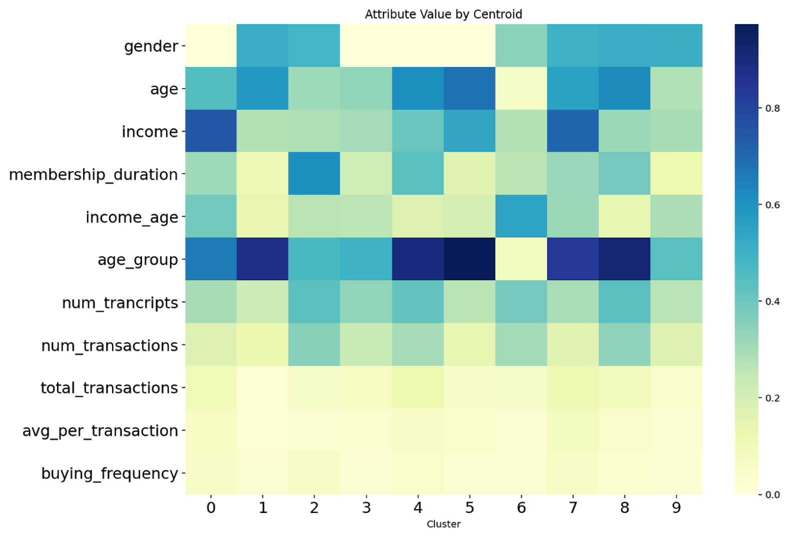


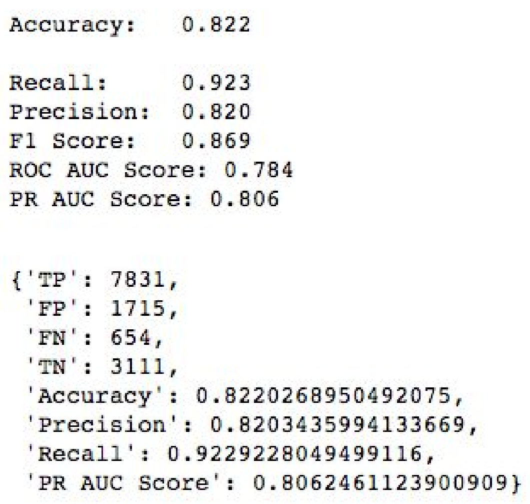
Figure 9. Heatmap of centroid in component space

Visualising centroids in component space as in Figure 9 can help us get an idea of how the clustering process was performed, what were the most determining features when defining the clusters and how they change across the clusters. Age group indeed seems to be one of the most determining features as in a way it is a clustering of the customers per se, and as if we have seen through the analysis performed so far that customers in different age groups have quite distinct socio-economic and purchasing characteristics. Additionally, income and gender seem to be very relevant too.

Binary classifier

The deployed XGBoost model tested on the test dataset has achieved a fairly satisfactory performance with an accuracy of around 82,2%, which is comparable to the accuracy achieved on the validation data set during model training and tuning.The rest of the metrics obtained while evaluating the model on the test data can be observed on Figure 10.

As you can see, the classifier achieves a quite high precision and recall (and consequently, F1 score), as well as a fairly high ROC AUC Score of 0.784. Recall (0.923) is noticeably better than precision (0.823), indicating that the model is very good in identifying actual positives, but a bit worse when it comes to producing false positives, i.e. perhaps too optimistic in forecasting that some customers will react positively to some offers. In future work, it would be an interesting task to see if both precision and recall could be brought above 90%. However, while these numbers seem very encouraging and in general present good results, their analysis will make much more sense when put into the context of comparison with the benchmark model in the next subsection.

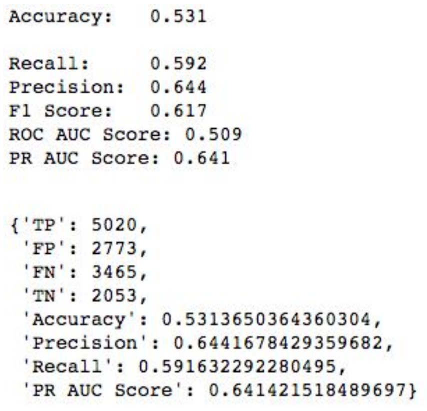


```
Accuracy:   0.822

Recall:     0.923
Precision:  0.820
F1 Score:   0.869
ROC AUC Score: 0.784
PR AUC Score: 0.806


{'TP': 7831,
 'FP': 1715,
 'FN': 654,
 'TN': 3111,
 'Accuracy': 0.8220268950492075,
 'Precision': 0.8203435994133669,
 'Recall': 0.9229228049499116,
 'PR AUC Score': 0.8062461123900909}
```

Figure 10. Evaluation metrics for XGBoost binary classifier (test data)

Justification

In this section, I compare the model's final solution and its results to the benchmark established earlier in the project using the same set of evaluation metrics that were used to evaluate the performance of the XGBoost classifier. The result of those metrics on the benchmark model (test data) is shown on Figure 11.



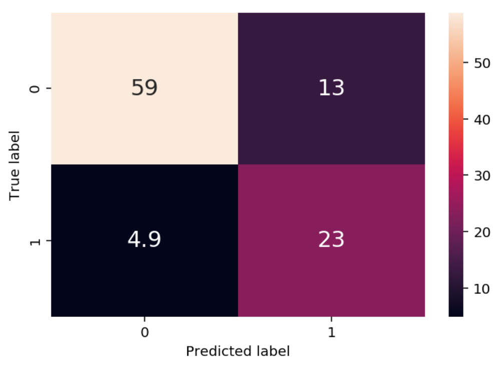
```
Accuracy:   0.531

Recall:     0.592
Precision:  0.644
F1 Score:   0.617
ROC AUC Score: 0.509
PR AUC Score: 0.641


{'TP': 5020,
 'FP': 2773,
 'FN': 3465,
 'TN': 2053,
 'Accuracy': 0.5313650364360304,
 'Precision': 0.6441678429359682,
 'Recall': 0.591632292280495,
 'PR AUC Score': 0.641421518489697}
```

Figure 11. Evaluation metrics for the benchmark model (test data)

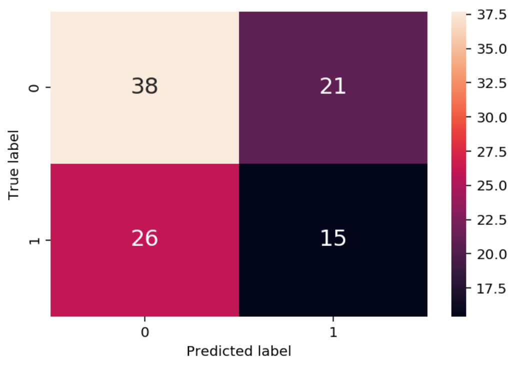
We can observe that the benchmark model (which sets a baseline for model evaluation as a random predictor who tries to blindly guess customers' response to an offer) achieves an accuracy of around 53% on the test data, the same data for which the ML trained model achieved 82% accuracy. ROC AUC Score is also a good indicator of the improvement achieved by using machine learning: the benchmark model obtained there 0.5 (as one would expect from random guessing), compared to 0.8 score of the XGBoost. Additionally, all the other metrics such as precision, recall, F1 score, etc. have been significantly improved over the benchmark model. Since the final results are much stronger than the benchmark results, I can consider the final solution significant enough to have solved the problem. Of course, further improvements of the model would be possible and desirable even, and I will discuss those in the following section.

V. Conclusion _(approx. 1-2 pages)_

Free-Form Visualization



a)



b)

Figure 12. Normalised confusion matrix (percentages) obtained, test data, for:
a) XGBoost classifier; b) benchmark model

From Figure 12, we can extract a few significant final remarks about the trained classifier and its performance (the values shown are percentages of all the examples in the test dataset). The trained XGBoost classifier has a probability of around 5% of wrongly classifying a customer who would respond to an offer, compared to 26% of the benchmark model. Moreover, the developed classifier will missclassify a customer who would not respond to an offer as someone who will respond to it with a probability of 13%, compared to 21% with the benchmark model. Therefore, we can observe that the trained model learnt much better to classify customers who complete offers than those who do not.
In general, it captures very well the customers who would respond positively to an offer, missing only 5% of them. However, whether that is more relevant for the business objectives of Starbucks than sending offers to wrong customers (who will not respond to them) it's hard to say without any consultation and insight into the business goals of a particular company. Intuitively, I would say in this case it'd be better to indeed aim at a lower rate of false negatives so not to lose any customers who would complete an offer by not sending it to them. In further model tuning, these business objectives could easily be integrated as goals into this model by setting desired levels of recall and precision.

Reflection

In this project, I explored a number of different analytical and modelling techniques on the provided Starbucks data: Starting with the loading of the data (given in json format), exploring it and visualising some aspects of it, detecting anomalous and erroneous data, cleaning it and transforming it so that it can be easily used for the envisioned tasks, followed by additional visualisation and analysis to deepen the understanding of the data. Lastly, I decided to develop and test two machine learning techniques: an unsupervised one of clustering for the purposes of market (customer)

segmentation and another supervised technique of developing a binary classifier to try to predict whether a customer will complete a particular offer extended to them, for which I also had to perform labelling of the data (assigning targets that the ML model needs to learn).
I've chosen those case studies due to the fact that they were fairly novel to me as I have never applied machine learning in an area such as targeted marketing, customer profiling or another similar commercially oriented application. To get the most out of this project, I wanted to test two different ML techniques with two slightly different objectives, and even though that added some significant workload to this capstone project, I feel it was worth it as I was able to explore different ways these kind of commercial data can be used in very hands-on commercial applications. Perhaps one of the most difficult aspects of this project was to understand the offering scheme of Starbucks from the data and to define ground truth labels that would make sense, as well as useful features, without having a lot of inside knowledge of the Starbucks' business schemes. Nevertheless, I believe the results I obtained are quite interesting and promising for the first prototype of such models, and I feel motivated to continue working on this project by finely tuning the developed models, creating novel features, trying out new ML models, or even enriching the dataset with new data that could boost the performance. Moreover, it would be a very interesting line of future work to combine this work with a set of business key performance indicators that would drive the development and fine tuning of these predictive models and could make this kind of project even more realistic and challenging, marrying its technical challenges with the business ones.

Improvement

There are several things I would like to try in the future work, starting with exploring additional machine learning models to solve this problem, as well as using more advanced ML techniques such as perhaps ensembling. Concretely, I would like to try, among other models, a custom designed PyTorch neural network classifier to see how it performs compared to the XGBoost classifier on the created set of features. I also leave feature importance for future work due to time restrictions, as I think it will be interesting to see how different features contribute to the accuracy of the binary classifier and even perhaps if that ranking changes over different models. Moreover, additional data sources could be of great help and interest to improve the results; especially in the unsupervised customer clustering task where the feature space was fairly small and I feel the model would profit significantly from having more socio-economic data on the customers. As I already mentioned in the previous section, I think it would be also really exciting to expand those scope of this project into a more business oriented area and have to work with a set of business key performance indicators (KPIs) that could guide and even challenge further the development of these models. Finally, the solution I developed here could be used as the new benchmark and a new even better solution could be obtained probably with an ensemble of various models.

References

1. Targeted Marketing: Explore the Strategy of Targeted Marketing. Last updated: November 28, 2020. Available at: https://www.marketing-schools.org/types-of-marketing/targeted-marketing/#:~:text=Targeted%20marketing%20identifies%20an%20audience,for%20those%20preferred%20market%20segments.
2. Data Science in Marketing: A Comprehensive Guide (with examples). Published on: May 26, 2020. Available at: https://nogood.io/2020/05/26/data-science-marketing-guide/
3. How to Avoid Data Leakage When Performing Data Preparation. Updated on: August 17, 2020. Available at: https://machinelearningmastery.com/data-preparation-without-data-leakage/
4. Elbow method (clustering). Wikipedia. Available at: https://en.wikipedia.org/wiki/Elbow_method_(clustering)
5. XGBoost hyperparameter tuning with Bayesian optimisation using Python. Simon Löw. August 2019. Available at: https://aiinpractice.com/xgboost-hyperparameter-tuning-with-bayesian-optimization/
6. A Gentle Introduction to XGBoost for Applied Machine Learning. Jason Brownlee. August 2017. Available at: https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/
7. XGBoost Algorithm, AWS. Available at: https://docs.aws.amazon.com/sagemaker/latest/dg/xgboost.html