Starbucks Passive Customer Detection Based on Offer Data and Customer Clustering

Anjali Thomas **Daniel Nguyen** **Kyoun Huh** **Vignesh Perumal**
Masters in Computing Science with Big data Specialization
Simon Fraser University
Burnaby, BC V5A 1S6, Canada
ata111@sfu.ca; dna29@sfu.ca; kha104@sfu.ca; vpa12@sfu.ca

Abstract

Detection of passive customers was performed using six different supervised machine learning models. Support Vector Machine model was found to be the most effective at classifying passive and active customers based on a selected set of five features. K-Means and Mean Shift clustering techniques were used to segment customers based on their responses to different types of offers. Four clusters were obtained as a result, using K-Means algorithm.

1 Introduction

Passive customer is one of the biggest challenges for any organization. A passive customer is one who has not searched for information regarding the company and is not involved in any activities. The likelihood of a passive customer switching to a competitor is more than that of an active customer.

This project aims to detect potential passive customers, thereby helping the business apprise the effectiveness of the Starbucks reward program, assist in retaining active customers, and to engage passive customers with targeted marketing. Using supervised machine learning models, the customers were classified as active or passive based on their responsiveness to the offers they received from Starbucks. Furthermore, unsupervised machine learning models were applied in an attempt to gain insights about the customer segmentation of Starbucks based on the preference of offer type.

2 Methodology

2.1 Data set Interpretation

The project is based upon an open-source data set titled "Starbucks Customer Rewards Program data sets"[1] from Kaggle. It contains the following data:

Portfolio: This file contains offer ids and meta data about each offer (duration, type, reward, difficulty). The offers are of three types, namely: Buy One Get One (which will henceforth be referred to as BOGO), discount and informational. It also contains a column - channels which lists the different media via which the particular offer is available.

Profile: This file contains the demographic data for each customer like age, gender and income. It also contains the date when the customer created the app account along with the customer id.

Transcript: This file contains records for transactions, offers received, offers viewed, and offers completed. The record description is held under the event column, and the corresponding value column provides an offer id or transaction amount based on the event. Each record also contains the customer id of the person performing the action.

2.2 Technologies used

The programming language used in this project is **Python**, and the coding was done in **Jupyter Notebooks** using the **Google Colab (Colaboratory)** platform. The additional libraries that were used in this project are ***numpy***, ***pandas***, ***sklearn*** and ***seaborn***. **Github** was the version control platform used and the link of the repository is: https://github.com/p33ves/Starbucks-Passive-Customer-Detection

2.3 Data Cleaning and Preprocessing

The records under the profile data containing missing values and outliers with age greater than 100 were removed. The identification numbers in both the profile data and the portfolio data were renamed to *cust_id* and *offer_id* respectively so as to facilitate performing joins with the transcript data. The transcript table is also modified with offer_id and transaction_amt columns obtained as a result of unpacking the value field from the original data.

3 Experiment 1: Classifying Passive and Active Customers

3.1 Feature Extraction

One-hot encoding for the gender feature was done to split it into three different columns, as it was a categorical feature. Income column was scaled down to *income_in_k*. *became_member_on* was transformed into an integer column named *days_as_members* providing the actual number of days since the customer has started using the application. Total number and ratio of offers received, viewed, completed, and transactions were aggregated according to different channels or offer types. Average difficulty, duration, and transaction amount were also added to provide a normalized metric for observing the interaction between Starbucks and the customer.

3.2 Label Creation for classification

Total view ratio and total completion ratio were used as the metrics for label creation. They were calculated based on the total number of offers each customer received. Both the completion ratio and the view ratio were taken into account as parameters in classifying the customers because even the action of viewing the offer notification showed the customers' responsiveness to an offer from Starbucks. The customers who had Total View ratio less than 0.6 and Total Completion ratio less than 0.2 were labelled 'Passive' and the rest were labelled 'Active'. Using this decision boundary, the customer data was split into two uneven groups containing 9637 active and 5188 passive customers.

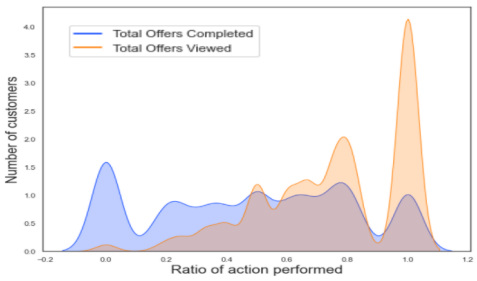

Figure 1: View ratio vs Completion ratio

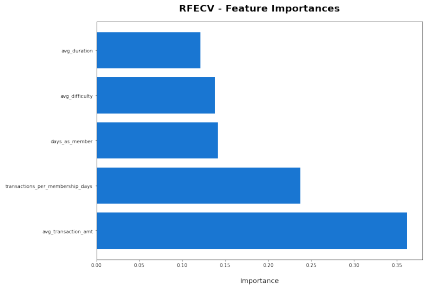

Figure 2: Feature importance chart

3.3 Feature Selection

Recursive Feature Elimination with Cross Validation (RFECV) was the major technique that was used for feature selection in this project. It is a backward selection technique which begins by building a model on the entire set of predictors and computing an importance score for each predictor. The least important predictor(s) are then removed, the model is re-built, and importance scores are computed again. The subset size of the input features that optimizes the performance criteria is used to select

the predictors based on the importance rankings. The optimal subset is then used to train the final model.

Since RFECV is expensive to run, some input features were eliminated using the correlation method prior to being used in RFECV. With the given set of input features, a correlation matrix was computed and all features having correlation co-efficient greater than 0.8 were removed. Features containing the offer counts were removed due to their apparent correlation with the target column, while other features like customer id and offer id were also removed because of them being arbitrary values used to identify each customer and offer distinctly. The remaining set of features were used to perform RFECV with Decision Tree Classifier as well as other classifier algorithms. Due to the stochastic nature of the process and evaluation procedure, there were some variations in the results, but with trial and error method the optimum number of input features was set to 5. As given in the **Figure 2**, the five selected features in the increasing order of feature importance are:

1. *'avg_duration'*: the average duration of different offers a person has received.
2. *'avg_difficulty'*: the average difficulty score of different offers a person has received.
3. *'days_as_members'*: the number of days a customer has been a member of Starbucks.
4. *'transactions_per_membership_days'*: the average number of transactions a customer performed throughout their days as a member.
5. *'avg_transaction_amt'*: average amount a customer spends per transaction.

3.4 Handling Imbalanced Data Set

Before a Machine Learning model was applied, the data was checked for balanced 'Customer_type' labeling in the training set. Our training target class y_train was imbalanced about 65% customer labelled 'active' and 35% 'passive' customers. A data oversampling SMOTE (Synthetic Minority Oversampling Technique) was applied on the training data set to get balanced data classification. The technique simply duplicates and oversamples the minority class, without adding any additional information in the data set. After the process, the training data had an equal classification of 'active' and 'passive' customers.

3.5 Model Selection and Tuning

The data was split into training and test data sets with a 75-25 percent ratio. Classification Machine Learning models that were used and compared are Logistic Regression, Support Vector Machine, K-Nearest Neighbor, Random Forest Classifier, Gradient Boosting Classifier, and Ada Boost Classifier.

3.5.1 Hyperparameter Optimization Results

To improve and compare the accuracies of the better performing models, hyperparameter tuning was applied to Random Forest, Gradient Boosting, Support Vector Machine, and K-Nearest Neighbours Classifiers.

GridSearchCV or the 'parameter sweep' model selection library in Scikit-learn was used to train the classifiers with hyperparameter tuning. With GridSearchCV, the hyperparameters have to be manually set as a list of values for each parameter and the algorithm returns the optimal values for each parameter. The following are the Optimal set of values returned by GridSearchCV:

Random Forest Classifier

- n_estimators=500; this sets the number of decision trees to be used in the forest. Values used were [100, 120, 300, 500, 800, 1200].
- max_depth=30; this sets the maximum depth of each tree. Values used were [5, 8, 15, 25, 30, None].
- min_samples_split=2; this is for the minimum number of samples needed before a split in the decision tree. Values given are [1,2,5,10,15,100].
- min_samples_leaf=1; this is the minimum number of samples needed to create a leaf. Values given were [1,2,5,10].

Gradient Boosting Classifier

- learning_rate=0.25; this shrinks the contribution of each tree by the learning rates. Default value is 0.1. Values set were learning_rate = [1, 0.5, 0.25, 0.1, 0.05, 0.01].
- n_estimators=200; this sets the number of boosting stages. default=100. Values used were n_estimators = [1, 2, 4, 8, 16, 32, 64, 100, 200].
- max_depth=15; this maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree. Default is 3. Values set were max_depths = [5, 8, 15, 25, 30, None].

K-Nearest Neighbors

- metric='manhattan'; this is the distance metric to use for the tree.
- n_neighbors=19; this gives number of neighbors to use. The lowest n-neighbor with the lowest error and highest accuracy was chosen, the number of n-neighbors can be found using the error chart.
- weights='uniform'; the weight function used in prediction, 'uniform', 'distance' or callable, default is 'uniform'.
- algorithm='auto'; this is used to compute the nearest neighbors, Values used are ['auto', 'ball_tree', 'kd_tree', 'brute'].

Support Vector Machine

- kernel='rbf'; this specifies the type of kernel to be used in the algorithm.
- SVM__C=100; this sets the regularization parameter. This must be strictly positive. The penalty is a squared l2 penalty.
- SVM__gamma=0.1; this is the kernel coefficient.

For all the classifiers the F1 scores increased after Hyperparameter optimization.

3.6 Prediction and Performance of Classifiers

Method	Before Tuning	After Tuning
Logistic Regression	67.91	67.03
K-Nearest Neighbors	66.91	71.29
Ada Boost Classifier	73.82	77.82
Gradient Boosting Classifier	73.82	78.93
Random Forest Classifier	79.19	79.49
Support Vector Machine	61.01	80.13

Table 1: Weighted Average F1 Score

3.7 Results and Inferences

- Among all the six machine learning models used, the Support Vector Machine gave the highest F1 score of 80.13%. Its performance increased significantly with hyperparameter optimization.
- All decision tree models showed fairly higher accuracies with and without hyperparameter optimization. The Random Forest Classifier and Gradient Boosting Classifiers gave higher F1 scores of about 79.49% and 78.93% respectively with the test data. This could be due to the fact that feature selection was using the Decision Tree algorithm in RFECV.
- Logistic Regression gave the least F1 score of only about 67.03%.
- Since the labels for classification was arbitrarily chosen by us, the result can be interpreted as the probability of responsiveness of the customer in performing actions such as viewing the offer and completing the offer.

- Initially, *income_in_k* was also used as an input feature during classification. It was the sixth important feature as computed by RFECV. But upon eliminating it from the input and re-running the models we observed that:
  - It did not affect the f1 and accuracy scores of the models significantly (by about 0.1%), so we can conclude that the offer responsiveness of a customer is primarily based on their spending patterns more than any demographic information.
  - Since the input features consisted only of the real-time transactions data, we can infer that there is a definite statistical relationship between the transaction history and generation of offers by Starbucks.

4 Experiment 2: Clustering Customers Based on Offers Type

Unsupervised Machine Learning models were used for clustering customers into different segments. This helps identify and analyze the groups of people that are interested and would potentially respond to a particular type of offer, based on their demographic information, so that similar offers can be targeted towards those groups in the future.

Clustering was performed with the following unsupervised machine learning models:

1. Mean Shift Clustering
2. DBSCAN Clustering
3. K-Means Clustering

4.1 Feature Extraction and Data Preprocessing

The master data set used for the first use case was reused as input to the clustering algorithms as well, with some features dropped and a few new features added to accentuate and segregate customer response to the different types of offers.

Features like the total Count of offers received, viewed and completed by each customer along with the counts specific to each offer are already present in the master Dataframe. Eight new features were added for computing the completion percentage of each type of offer, computed as:

$$\frac{\text{Total count of Offer X (received/viewed/completed)}}{\text{Total number of Offers (received/viewed/completed)}}$$

Five new features were added for ratios of completion and viewing which are computed as:

$$\frac{\text{Total count of Offer X (viewed/completed)}}{\text{Total count of Offer X received}}$$

where X = Bogo/Discount/Informational

Note: Informational Offers cannot be completed

'cust_id', 'became_member_on' (since 'days_as_member' feature is used), all variables related to communication channels like email, sms, mobile web was dropped as these wouldn't contribute to clustering customers for the offer types.

Handling Missing Data SimpleImputer class of Scikit-learn impute library was used to handle missing values by replacing 'Nan' values with the median values along each column.

Scaling the Data StandardScaler utility class of Scikit-learn preprocessing library was used to standardize and scale all feature values.

4.2 Model Selection

4.2.1 K-Means Clustering

The K-Means Clustering algorithm clusters the entire data into k clusters. It initially picks k random points from the dataset and assigns the rest of the data into one of the k clusters based on the least

distance of the data point from each cluster(the Sum of Squared Distance (SSE) is calculated). Once this is done the average value of each cluster is computed and the process is repeated until there is no change in the output and the clusters are optimized.

Choosing the optimal number of clusters

To find the optimal k, the K-Means algorithm was fitted on the data iteratively for a range of k values (here values for k from 1 to 12). Based on the SSE scores for different values of k, the Elbow Method was applied and both the Silhouette score and Calinski Harabasz scores were calculated.

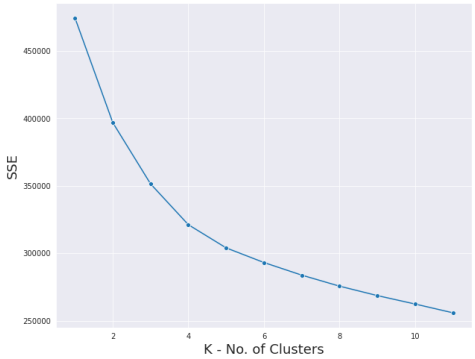

Figure 3: Elbow Method for optimal k

k	Silhouette Score	Calinski Harabasz Score
2	0.14849762254725	2896.60409020
3	0.142629205412459	2593.25953438
4	0.142925934731120	2357.03747857
5	0.123684772275592	2077.02109836
6	0.116120483131220	1832.63355552
7	0.100250659905370	1658.02313534
8	0.102205904773770	1522.40226076
9	0.102385273698649	1421.23391153

Table 2: Silhouette and Calinski Harabasz scores for optimal k

The Elbow Method showed the graph to bend at k=4 or k=5. From the values of Silhouette Scores and Calinski Harabasz Scores for a range of k values, the scores peak at k=4 and drops considerably after k=4. So we chose the optimal value of k as 4.

4.2.2 Mean Shift Clustering

Mean Shift Clustering seems to not work very well with high dimensional dataset, as it groups almost all of the data into 1 big cluster, and the other 2 clusters are so small that they are not visible in our visualizations. It can detect outliers very well, but cannot identify the different segments of customers based on all of the features we provide.

Therefore, a different goal was set where a group of 3 features were chosen at a time to find outliers of the data. The outliers can then be removed prior to running a supervised machine learning model, but they can also provide useful insights.

Mean Shift by its default estimate_bandwidth method is often not able to find more than 1 cluster with our sets of features. The bandwidth has to be reduced to 1 or 5 for it to come up with more than 2 clusters.

4.2.3 DBSCAN Clustering

With the entire set of features, DBSCAN algorithm could not find any clusters without us changing the parameter eps and sample_range, and it also runs into the same problem as Mean Shift where it creates one big cluster that is much larger than the other clusters, to the point that they cannot be visualized effectively.

A different approach was used where a set of 2 features were chosen at a time and fit into the model, and our goal is to detect outliers and investigate. However, DBSCAN did not offer any insight that the other 2 models do not have, so it is not included in our analysis in the end.

4.3 Clustering results

4.3.1 K-Means Clustering

t-distributed stochastic neighbor embedding The t-SNE algorithm was obtained to get an overall 3D visualization of the four clusters, which better visualization the clusters than the PCA

dimensionality reduction algorithm. Also, the top 5 clustering features were obtained (refer to GitHub Notebook).

Figure 4: Age vs Income

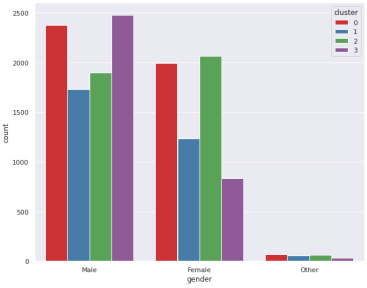

Figure 5: Gender Distribution

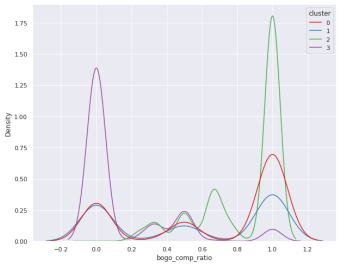

Figure 6: Bogo Completed

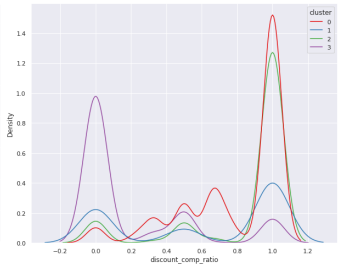

Figure 7: Discount Completed

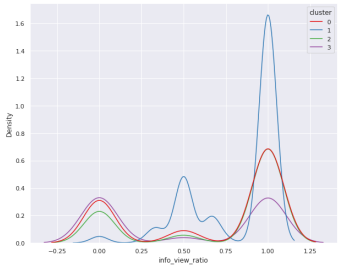

Figure 8: Info Viewed

Cluster 0 - Red, Cluster 1 - Blue, Cluster 2 - Green, Cluster 3 - Purple

Based on the visualizations we obtained, below are some of the conclusions that can be drawn about the four different clusters: **Cluster 0** - This cluster contains the most number of people, and it also includes a higher number of the older population with age greater than 60. While the income spread follows the general correlation with age, it seems like the customers in this cluster strongly prefer completing discount offers as compared to BOGO ones.

Cluster 1 - Containing the least number of customers, this cluster is made up of middle-income earners who seem to be the least responsive both in terms of viewing the offers and completing them. The only action they perform better than the rest is viewing the informational offers.

Cluster 2 - The only cluster that contains more female customers than male, it is made up of relatively higher income earners. This cluster consists of the most active customers, completing a high percentage of offers with a noticeably higher preference to BOGO offers as compared to discount offers.

Cluster 3 - Consists of a high ratio of male customers as compared to females, and possesses many of the younger population of the customer base with age less than 30 years. The average income of the people belonging to this cluster is also significantly lower than other clusters. This reflects in their offer responsiveness as their offer view percentages are very high (could be due to their high usage of mobile phones and internet) but they do not complete many of the offers provided to them (could be due to their economic situation).

4.3.2 Mean Shift Clusters

The following can be deduced from the below visualizations of clusters:

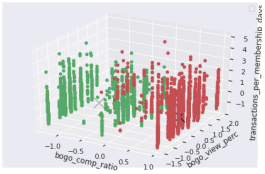

Figure 9: BOGO

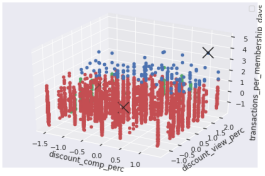

Figure 10: Discount

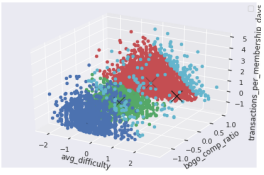

Figure 11: BOGO

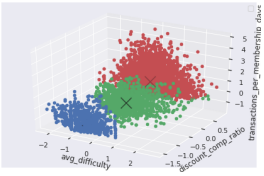

Figure 12: Discount

- **Figure 9:** Both clusters have decent viewing percentages and low count of visits to Starbucks. The green cluster is not completing their offers, but it does not seem to affect their count of visits to Starbucks. BOGO offers may not help increase customers' visits and spending at Starbucks.
- **Figure 10:** The blue cluster represents a group of customers who are willing to visit Starbucks often while participating in the rewards program. They can be considered loyal customers, and it should be the goal to convert other customers into this blue cluster. Discount offers seem to help distinguish different segments of customers.
- **Figure 11:** The light blue cluster represents a group of customers who are willing to visit Starbucks often and redeem their BOGO offers, despite these offers having an above-average difficulty score. This cluster can be considered loyal customers, and it should be the goal to convert other customers into this light blue cluster. For BOGO offers, difficulty score matters. However, the clusters do not separate well in this graph and there are a lot of outliers, which means customer loyalty is harder to estimate using BOGO offers.
- **Figure 12:** The blue cluster represents a group of customers who are disinterested in Starbucks and its reward program, even though the offers are easy to complete; they are not taking action on them, and also not making trips to Starbucks. This cluster should be investigated to see whether they can be better enticed. The red cluster is larger than that of the BOGO offers, so it can be deduced that customers who complete more discount offers visit Starbucks more often, even though the mean difficulty score of discount offers is higher than that of the BOGO offers.

5 Challenges and Learning

- While trying to understand the Starbucks data set, some of the challenges we faced were:
  - The flow of transaction events from offer received to offer viewed, then transactions and finally offer completed were not always in order. For example, there were numerous occasions where offers were completed without any transactions and times when an offer was completed without being viewed. The implications of these abnormal event flows were ignored since only the count of the event occurrences were required for aggregations of total and average values.
  - Records with 'Offer completed' event sometimes occurred after the duration for the offer, meaning the offer was expired. These records were still considered since the customer did express their willingness in utilizing the offer, if possible.
  - Reward that was awarded for offer completion did not match with the reward defined in the portfolio. Since this was not a part of the set of input features selected, it was ignored.
- The decision boundary for classification could have been selected in such a way that it splits the data into two groups of nearly equal sizes so that uneven distribution of data points between the groups could have been prevented.
- Visualizing the clusters obtained from unsupervised learning models felt challenging as it required analyzing across multiple features to identify the key differences.
- Individual groups obtained as a result of clustering did not exhibit plainly discernible differences, which meant that further study into feature selection was required in addition to selecting more suitable clustering methods.
- Another major learning from the project was that unsupervised learning could be used for label creation for classification to detect customers who exhibited passive behavior towards the Starbucks offers.

6 Summary of individual contributions

- Daniel Nguyen : contributed to the creation of new features, the trial and errors process of features selection, the development of the Support Vector Machine, K-Nearest Neighbor, Mean Shift and DBSCAN models, as well as their visualizations.
- Vignesh Perumal : contributed to data cleansing, preprocessing, label creation, feature extraction in both experiments, feature selection using RFECV and trail and error approach using different tree based classification algorithms, optimum number calculation and analysis of K-means cluster results, along with deriving inferences and documenting challenges and learning.
- Kyoun Huh : contributed to data cleansing, preprocessing, trial and error process of feature selection, development and hyperparameter optimization of the Support Vector Machine, K-Nearest Neighbor models and experimented Mean Shift clustering model in comparison with K-Means clustering model, as well as their visualizations.
- Anjali Thomas : contributed to data cleaning, implemented SMOTE, implemented machine learning algorithms for experiment 1 - Linear Regression, Random Forest Classifier, Gradient Boosting Classifier, Ada Boosting Classifier using GridSearchCV, Hyperparameter Optimization of Random Forest and Gradient Boosting classifiers and their prediction results with different feature sets, implemented feature extraction and data preprocessing for experiment 2, implemented K-Means Clustering for experiment 2, contributed to data visualizations for both experiments.

Poster making and report creation were completed through shared effort by all the team members.

References

[1] Kaggle," Starbucks Customer Rewards Program Datasets",
https://www.kaggle.com/blacktile/starbucks-app-customer-reward-program-data

[2] Roos, Inger & Gustafsson, Anders (2011) *The influence of active and passive customer behavior on switching in customer relationships*,
https://www.researchgate.net/publication/242348427_The_influence_of_active_and_passive_customer_behavior_on_switching_in_customer_relationships

[3] Linda, G.Schneider & Imran, S.Currim (1991) *Consumer purchase behaviors associated with active and passive deal-proneness*, International Journal of Research in Marketing, Volume 8, Issue 3, pp. 205–222.

[4] Siavoshi, Mehrnaz (2020) *To offer or not to offer? A Starbucks Machine Learning Project*,
https://towardsdatascience.com/to-offer-or-not-to-offer-a-starbucks-machine-learning-project-c21bf9c398bc

[5] Brownlee, Jason (2020) *Recursive Feature Elimination (RFE) for Feature Selection in Python*,
https://machinelearningmastery.com/rfe-feature-selection-in-python/

[6] Sharma, Mathanraj (2020) *Grid Search for Hyperparameter Tuning*,
https://towardsdatascience.com/grid-search-for-hyperparameter-tuning-9f63945e8fec

[7] Cornellius Yudha Wijaya (2020) *5 SMOTE Techniques for Oversampling your Imbalance Data*,
https://towardsdatascience.com/5-smote-techniques-for-oversampling-your-imbalance-data-b8155bdbe2b5

[8] Dennis T. (2019) *Confusion Matrix Visualization*,
https://medium.com/@dtuk81/confusion-matrix-visualization-fc31e3f30fea

[9] Andrea Xue (2019) *Starbucks Offer Personalization — Sending the right offer to the right customer*,
https://towardsdatascience.com/starbucks-offer-personalization-sending-the-right-offer-to-the-right-customer-14d4fbc20575