AML Final Project Report – Customer Behavior Analysis

Group 16: Anne Lin (al4215), Chandan Suri (CS4090), Xingcheng Rong (xr2150), Yunchen Yao (yy2949), Ziwei Han (zh2484)

1. Project Overview

"Customer Behavior Analysis" dataset is a well-structured dataset about customers and their experiences. In marketing, customer segmentation can help businesses attract customers more effectively by suggesting specific types of campaigns for different groups of customers. Our dataset includes 2240 entries in total. Each entry has 29 attributes, which can be divided into 4 categories, namely people, product, promotion, and place.

The main objective of our project is to analyze and use machine learning methods to, firstly, predict whether a customer will be more willing to accept a campaign offer (and thus more inclined to buy) based on the characteristics above, and, secondly, cluster customers into specific groups and analyze the group-specific features separately which can help a product engineering team to look through the types of customer segments and devise product accordingly.

2. Exploratory Data Analysis

By checking the demographic of the customers, we can deduce the following:

a. The majority of the respondents are educated since most of the respondents are graduates having Ph.D. or Masters' degrees.
b. Most of the respondents are in some kind of relationship according to their "Marital Status". Also, single customers are less likely to accept campaigns compared with customers with partners.
c. By checking the continuous features, some of the features are correlated, for example, the number of purchases shows a linear correlation with expenses. There is no sharp difference between the distribution of money spent on each type of product, and if there are discounts, customers will mostly purchase at least once rather than none.
d. Furthermore, we noticed that customers tend to make more purchases in stores than online. In regard to their acceptance of any campaign offer, we realized that the groups are a little imbalanced in that 73% of them never accepted any offer. (73-27 ratio)
e. There are some outliers in the "Income" feature. Also, most of the customers are born between the 1960s and 1980s. Also, the number of days since the customer's last purchase is more or less uniformly distributed and it seems that this feature won't contribute much to the learning.
f. Lastly, it also seems that some features like "Expenses", "Number of accepted campaigns", "Income" and "Total Purchases" might be some of the most significant features that have a correlation with the final response label.

3. Data Preprocessing

We have dropped two columns that have the same value for all entries and thus, won't contribute to the predictive or segmentation models. Also, we remove customer ID as an index since it is unique for each customer. Then, we change the birthdate to age by subtracting it from the current year and changed the customer enrollment date to the number of enrolled days. We also encoded "Marital Status" according to whether the customer is single, in a relationship, or in an unconventional status, and encoded "Education" according to whether the customer is of undergraduate or postgraduate status. We also summed up the expenses and purchase amounts in different categories as it seemed more logical to keep those features as a linear combination of all of them. There is only missing data on income, which may be the result of the unwillingness of respondents to report their private information. Since the missing data only accounts for around 1% of the total, we decided to take the average value and then place that data for all the missing values for the income. Also, we "One Hot Encode" the categorical variables in our dataset. As both the columns just introduce 2-3 categories into our feature set, we don't have to pay much heed to the curse of dimensionality.

After preprocessing the data, we have 12 features in total. For classification, we use users' responses to the last champion as the target. The target is 0 or 1. 1 means accepting the offer from the last champion and 0 otherwise. And we use the other 11 features as training input. For clustering, we used all 12 features as training input as we need to form clusters and take the responses also in consideration for the clusters or customer segmentation. Additionally, we use the "Stratified Splitting" technique to stratify the train-test split (20% validation/training and 80% training) based on the responses label. This keeps the ratios of both the classes in both the splits in check as we have a considerable imbalance here.

Lastly, we also scale the data with the "Standard Scaler" for the Logistic Regression and the Neural network. We do this for the numerical columns and then stack them with the categorical ones. For other models, we don't need any kind of scaling. Also, we have used SMOTE for the 2 models (so we form 4 models in total) mentioned above to make the dataset balanced in terms of the 2 response classes.

4. Machine Learning Methods and Evaluation

We have tried eight different models (3 traditional ML classification, 1 neural network, 2 clustering algorithms, 2 classification methods with SMOTE) and tuned the hyperparameters based on the F-1 score on the test dataset, and also clustered the groups. We used the best model to predict if the respondent would intend to accept the campaign offer after training and found the most significant factors that would influence their decisions. Note that since our dataset is quite imbalanced (more customers who didn't accept campaign offers), we are using the F-1 score as our metric here. Furthermore, we are using the F-1 score here as we need to consider both the false positives (Precision) and false negatives (recall) equally for our use case here.

Logistic Regression

Here our classification is binary (accepted campaign or not which implies the buying habits of the customers), we first consider the logistic regression model. Firstly, we split the dataset using stratification based on the responses label and then scale the features in the dataset. Following this, we perform a "Grid Search" for finding the best set of hyperparameters based on the "F-1" scoring metric. After getting the best set of hyperparameters, we create the model with those parameters, train, and make predictions. Also, through observing the feature importance, we find that the features, "Total Accepted number of campaigns", "Income", "Number of days since they were the customers", and "Relationship Status" were the most important features in our dataset for the model. Following this, we get some decent results in terms of many metrics as shown in the table at the end.

Furthermore, as the F-1 score, was a little lower, we perform SMOTE on the dataset, find the best set of hyperparameters through Grid Search and fit the best model again. We get even better results this time after performing SMOTE which suggests that SMOTE can help us a lot with this kind of dataset. The evaluation metrics for the same are shown in the table at the end.

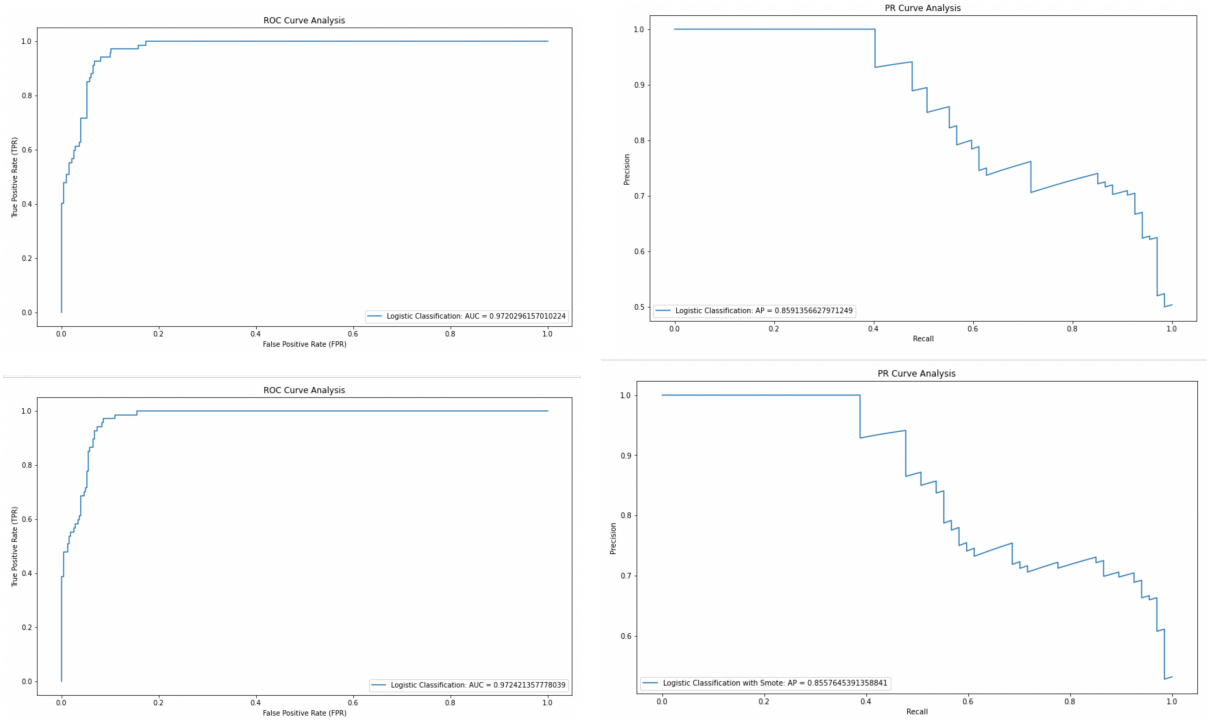


Figure 1: ROC & PR Curves for Logistic Regression models (top row – without SMOTE), (second row – with SMOTE)

Histogram-based Gradient Boosting

As the features have lots of useful information which can be segmented using some rules, it makes sense to try out a tree-based classification model. At first, we try the "Histogram-based Gradient Boosting". We split the dataset into 20% for testing and 80% for training. Firstly, we split the dataset using stratification based on the responses label and then scale the features in the dataset. Following this, we perform a "Grid Search" for finding the best set of hyperparameters (learning rate, max depth, and l2 regularization) based on the "F-1" scoring metric. After getting the best set of hyperparameters, we create the model with those parameters, train, and make predictions. The metrics for the best model with the best set of hyperparameters can be found in the table at the end. Also, we get the following ROC and PR curves as shown below. As we can see, we get decent results for the same. The model here performs comparably to the Logistic Regression, albeit a little worse. Additionally, for the marketing teams, the probability of a customer accepting to buy a product might be really useful so, we try to calibrate the probability distributions using "Platt Scaling" and "Isotonic Regression", but we find that these calibration methods weren't useful as the original distribution had the least Brier scores among the calibrated models (0.045, 0.048) and the uncalibrated one (0.044).

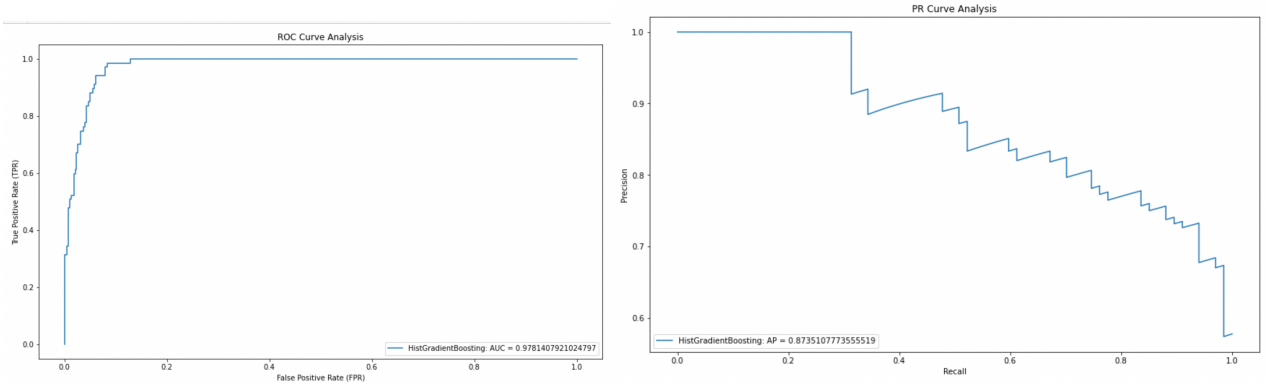


Figure 2: ROC and PR Curves for the Histogram-based Gradient Boosting Model

XG Boost

As we couldn't calibrate the probability distribution beyond a point, we tried looking into another tree-based model. We follow the same procedure as followed for the "Histogram-based Gradient Boosting" model including searching for the best set of hyperparameters (eta, max depth, and gamma). Also, we find that "Total Accepted Campaign numbers", "Related relationship status", "days since a customer", and "income" are found to be the most important features. After trying with just these important features, we find that the model's performance goes down, and thus, we consider all the features for now. This model performs a little better in terms of all the metrics in comparison to the previous tree-based model. Lastly, after performing 2 types of calibrations, we find that the probability distribution moves away from the perfect classifier probability distribution, and we get the best probability distribution possible with the original model (Brier score = 0.045) rather than the calibrated one (Brier scores = 0.051 & 0.046).

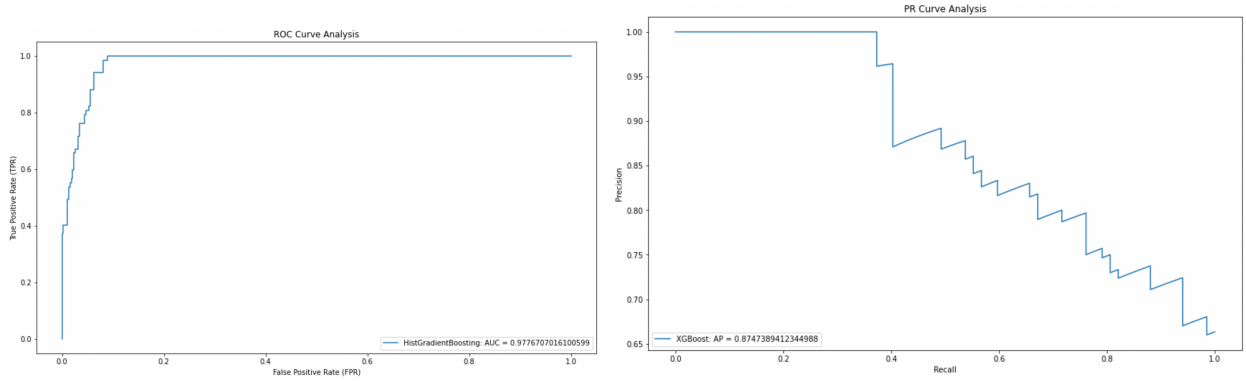
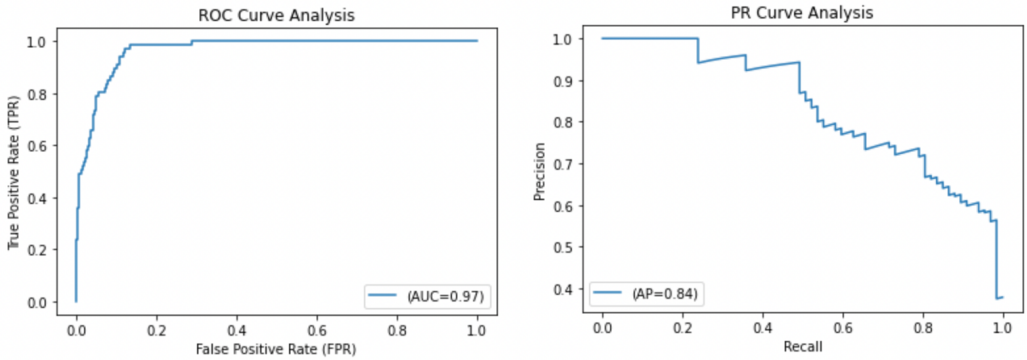


Figure 3: ROC and PR Curves for XG Boost Classifier

Neural Network

After doing all the pre-processing including the scaling and splitting, we create a model from scratch that uses one input dense layer with 128 units, one hidden dense layer with 64 units, one hidden dense layer with 32 units, and one output layer with 2 units. We also added batch normalization to prevent possible overfitting. Following that, we tune the hyperparameters (batch size and optimizer) using Grid Search. After finding the best set of hyperparameters, we train the model for 30 epochs. The metric scores can be seen in the table below. Also, the ROC and PR curves are shown below.

Furthermore, to improve the performance of our model, we also tried SMOTE on the development set. This results in a performance boost. As we can see from the accuracy and loss curves below, this model reaches the best performance among all our models for classification in terms of all the metrics involved.



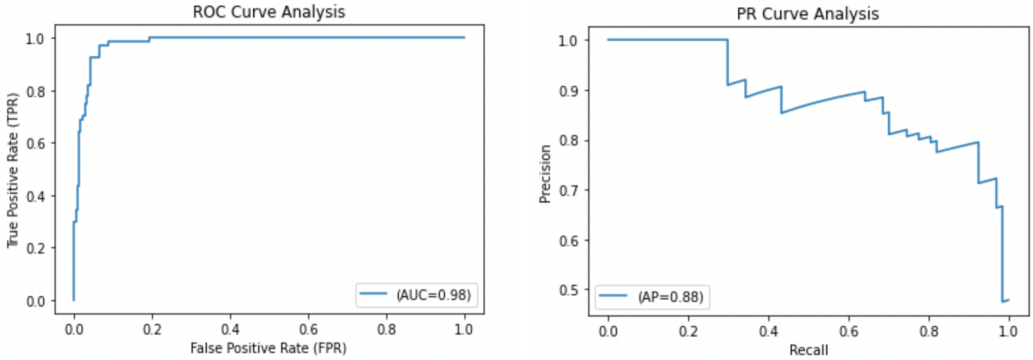


Figure 4: ROC and PR Curves for Neural Network model (top row – without SMOTE), (second row – with SMOTE)

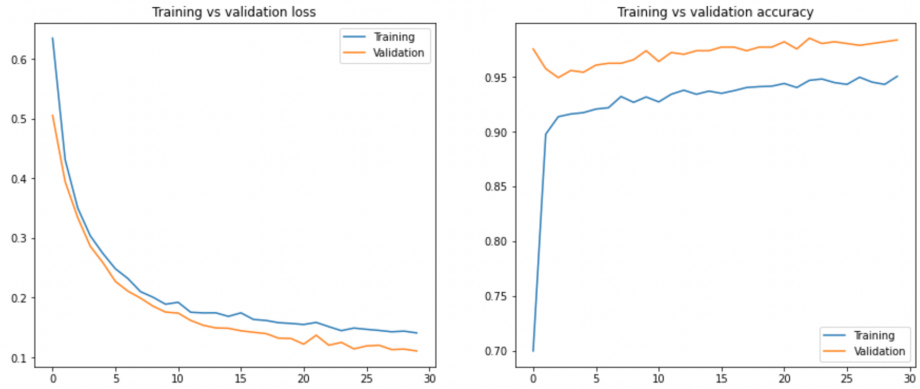


Figure 5: Loss and Accuracy curves for Neural Network with SMOTE

Metrics for all the Classification Models:

Model Name	F-1 Score	AUROC	AP	Accuracy (%)
Logistic (without SMOTE)	0.688	0.9720	0.8591	91.29
Logistic (with SMOTE)	0.7826	0.9724	0.8557	92.19
Hist-Gradient Boosting	0.7538	0.9781	0.8735	92.86
XG Boost	0.7596	0.9776	0.8747	93.08
Neural Network (without SMOTE)	0.7552	0.97	0.84	92.19
Neural Network (with SMOTE)	0.8052	0.98	0.88	93.30

Clustering Techniques

To further process the data for clustering, we first removed outliers from the data. Then we tried two different clustering methods, KMeans and DBxScan. We used the elbow method and silhouette score to find the best number of clusters to be two for both methods.

For KMeans, the number of samples in each cluster seems almost balanced. Also, we have generated distributions of features in each cluster for interpretation. Looking at the clusters above, we can see some customer behavior in the 2 clusters. This information can be used to cluster and form different customer segments according to their characteristics. The clusters formed are visualized in the figure below.

1. Income: In terms of Income, Cluster 2 is the customer segment that has a higher income than the customers in the 1st segment. Also, the cluster with the higher income is constituted of people with slightly less children and larger age.
2. Expenses: In Cluster 2, we have customers that have a higher spending habit in comparison to the customers in the 1st cluster. This can also be directly correlated with the fact that the customers having higher income would have high expenses as well. Thus, these customers would tend to buy a lot.
3. Total Number of Purchases: In cluster 2, we have customers that tend to make more purchases as well.

Thus, as we can see above, these customer segments are formed on the basis of the spending habits and on the financial situation of the customers. This kind of behavioral analysis would be very helpful for us to segment the customers in the clusters accordingly which can help the marketing team a lot (advertising too!)

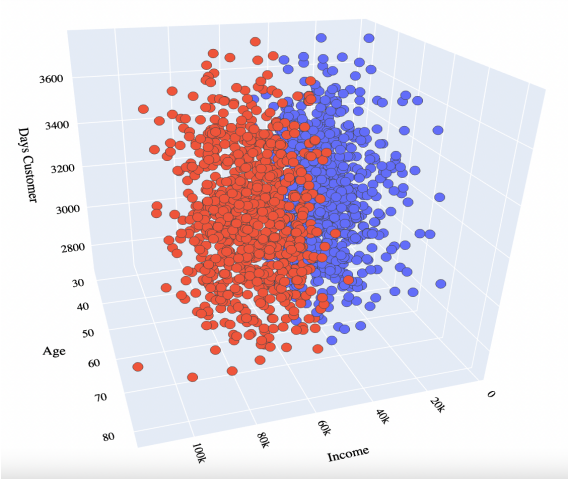


Figure 6: Two clusters formed after K-Means Clustering

For DBScan, the number of samples in the first cluster is around half of the number of samples in the second cluster. We also perform PCA to reduce the dimensionality of the data here. Along with that, we perform a knee-based analysis to find the best number of clusters for this. Following all this, we generate the distribution of each feature in the two clusters. From the graph, the clusters for DBScan are generated based on marital status. Cluster two contains samples in a relationship whereas cluster one contains the remaining samples. The other features don't show distinct differences between these two clusters, with cluster two having a slightly lower average number of total purchases.

5. Conclusion

Classification: After trying out six different classification pipelines, we find out that the Neural Network performs the best among all the classification models. It has the highest F-1 score about which we care the most. Also, this model performs the best in terms of all the other metrics such as AUROC, AP, Precision, Recall, and Accuracy. Although, if we had to use a traditional ML-based algorithm, I would choose XG Boost as the best one. Although it has a lesser F-1 score than the logistic regression with SMOTE pipeline, it has other comparable metrics with a larger AP. For our use case here, we consider the F-1. Still, precision would be a little bit more useful for our use case here if combined with the availability of the probability distribution. Also, we can find calibrated probability distribution from the classes here, which would be very useful for our use case of customer behavior analysis.

Clustering: Looking at the results for clustering techniques above, we can indeed say that KMeans performs much better than DBSCAN. From the clusters formed above, we can say that KMeans does a better job that can be used for segmenting the customers into different segments and thus, helping in targeted advertisements accordingly.

Conclusively, we could find at least one good classification and one clustering model to achieve the two goals for our project. The classification model achieved the goal of classifying and predicting whether a customer will attend the next campaign and buy more products in the future based on the customer and other characteristics. And the clustering model could successfully form segments of customers that can be used for targeted advertisements. We could even create different types of clusters by considering different subsets of the features if we had more data at our disposal. But still, this still shows much promise in the area.