SPICE ALLEY – Predicting Customer Behavior

20221617 | Diogo Charola
20221624 | Francisco Antelo
20222142 | Francisco Loureiro
20221622 | Joana Bravo
20222134 | Maria Beatriz Amado

Abstract

Two months after the successful deployment of Spice Alley's new marketing plan, the company is now relying on the same team of data scientists that were behind it to create a predictive model, which will be capable of driving profits for a sixth marketing campaign geared towards promoting a new product related to frozen food to the customer database, which has a potential of up to 10,000 customers.

Seven different algorithms will be tested in a dataset of 2,500 customers in order to predict the customer behavior of the remaining 7,500 customers. Based on the results, the chosen models will retrieve customers who are more likely to purchase the new product, while disregarding non-responders.

Keywords: Marketing campaign; Predictive models; Feature Selecting; Cross Validation

INDEX

1. Introduction

Spice Alley is committed to enhancing the customer experience through the analysis of its extensive data. Recognizing the rise of this potential powerhouse in the restaurant business, both in terms of popularity and its embrace of data analytics, the company is now attempting to identify consumer behavior patterns that can be leveraged to uncover new opportunities for business expansion in the future.

This project aims to build a predictive model that can target the most suitable customers for the sixth direct marketing campaign of Spice Alley, with the goal of maximizing profit from selling a new frozen food product. To construct this model, we will utilize a dataset containing information about the customers who were targeted in a pilot campaign, along with their behavior and response to the campaign. Ultimately, the model should be capable of identifying patterns and estimating which clients are more likely to purchase the new frozen food product.

2. Methodology

This project followed the OSEMN (Obtain, Scrub, Explore, Model, iNnterpret) methodology, with the aim of identifying groups of customers with similar characteristics to each other. The OSEMN methodology is a widely recognized and standardized model for conducting research in Data Science, offering a clear and coherent set of steps that are easy to follow (Dineva and Atanasova, 2018).

2.1. Obtain Data

To conduct the present analysis, two data sets were used: Historical Data, to build the machine learning models (File: historical.xlsx), and Predictive Data, to see how well the model performs on unseen data (File: predict.xlsx).

These were loaded under the names “historicalDF[1]” (Appendix I -) and “predictDF [2]” (Appendix II -).

2.2. *Scrub Data*

The starting point of this analysis is the ‘historicalDF’ dataframe. Before conducting the analysis, several data quality issues were addressed. Specifically, 18 duplicate values were detected, and there were missing values in the “Education” (32 instances), “Recency” (48 instances), and “MntDrinks” (21 instances) columns.

To improve the data quality, we performed data transformations. These transformations consisted of converting the dates to the correct format using the datetime format and correcting typographical errors

[1] 2500 rows and 30 columns.

[2] 2500 rows and 29 columns.

(e.g., converting "Basic" to "basic"). Additionally, misclassifications were rectified. For example, the date 2/29/2022 was corrected to 2/28/2022. These data transformations aimed to ensure that the data is in the appropriate format and accurately represents the intended values.

Furthermore, new variables were created to provide additional insights. One such variable was “Age”, which was derived from the “Birthyear” variable. Another variable was “Gender”, derived from the “Name” variable.

The missing values for “Education” were predicted with a KNN imputer, using “Income” and “Age” as predictors. Regarding the missing values in ‘MntDrinks’ and ‘Recency’, the predictors with the highest correlation between each of the 2 variables were used[3], respectively, to fill in the missing values using a Random Forest model.

2.3. Explore the data

After cleaning and filtering the data, it was necessary to classify the data into categorical and numerical variables (Appendix III -), and check if any of the numerical variables followed a normal distribution, in order to decide whether or not to normalize or standardize the data. The Jarque Bera statistical test was carried out for this purpose, and it concluded that no variables followed a normal distribution.

Additionally, both the correlation between all numerical variables (Appendix IV - Appendix IV -), where it was possible to see that many variables are correlated with each other. The summary statistics were also examined (Appendix V -), to better comprehend the dataset.

Finally, this section also explored Visual Data (Appendix VI -) about the distributions that describe the respective variables, confirming what was previously concluded in the Jarque Bera test, as well as allowing for a clearer visual of the distribution of the data.

2.4. Feature Selection (Transform Data)

Some steps were required before proceeding to Feature Selection. The StratifiedKFold Cross Validation method will be applied for further use in the feature selection techniques, with a total of 5 folds. Additionally, the variables 'Date_Adherence', 'BirthYear', and 'Name' were dropped from the ‘historicalDF’, as their values were deemed irrelevant for modeling.

At the beginning of feature selection, the data type of each variable was identified since different techniques are specific to different data types. For categorical features, a Chi-Square test was employed to determine the categorical variables that should be included in the subsequent models. Using the

[3] 'MntEntries', 'MntDesserts', 'NumStorePurchases', 'MntVegan&Vegetarian', 'Income' for MntDrinks, and 'DepVar', 'NumTakeAwayPurchases', 'Income', 'Kid_Younger6', 'MntMeat&Fish' for ‘Recency’.

previously defined StratifiedKFold (Appendix VII -), it was concluded that i) the variables 'Marital_Status', 'Response_Cmp2', 'Response_Cmp3', 'Response_Cmp4', 'Response_Cmp5', and 'Response_Cmp1' would be used in our models, and ii) the variables 'Gender', 'Education', and 'Complain' were removed (Appendix VIII -).

As for the numerical features (Appendix IX -), the variance of the data variables was examined. As a result, it was observed that the variables 'CostContact' and 'Revenue' exhibited zero variance. Consequently, these variables were excluded from further analysis.

To avoid redundant information in the numerical variables, the Spearman correlation was utilized. This analysis involved grouping variables based on their highest correlation with each other, effectively dividing them into two distinct groups (Appendix X -). More precisely, it was possible to observe high correlations within two groups of variables:

- Group 1: ‘MntDesserts’, ‘MntEntries’, ‘MntDrinks’
- Group 2: ‘MntMeat&Fish’, ‘Income’, ‘MntVegan&Vegetarian’, ‘NumAppPurchases’, ‘NumTakeAwayPurchases’, ‘NumStorePurchases’

In order to determine the most important variables, a decision tree classifier was utilized along with the StratifiedKFold technique. This approach helps identify the variables that consistently appear as the most important across multiple iterations, ranked in descending order. By analyzing the feature importance from the decision tree classifier using StratifiedKFold, it is possible to obtain insights into the variables that consistently contribute the most to the classification task. It selected ‘MntDesserts’ and ‘MntVegan&Vegetarian’ as the most important ones from group 1 and group 2, respectively (Appendix XI -).

The next step involved applying various feature selection techniques, namely RFE (Recursive Feature Elimination), Lasso Regression, and once again Decision Trees, without the previously excluded variables. These techniques were combined to determine the final set of features for further modelling, based on their performances in each of the three techniques (Appendix XII -). The outcome was that the variables 'Recency', 'MntVegan&Vegetarian', 'MntAddRequests', and 'NumAppVisitsMonth' should be considered in the model, while 'Kid_Younger6', 'Children_6to18', 'MntDesserts', 'NumOfferPurchases', and 'Age' were removed (Appendix XIII -).

In summary, we obtained two distinct datasets based on the feature selection techniques employed earlier. The first dataset, called ‘X’, comprises the most important variables determined by the feature selection techniques. These variables include 'Marital_Status', 'Recency', 'MntVegan&Vegetarian', 'MntAddRequests', and 'NumAppVisitsMonth', as well as the response variables ('Response_Cmp2', 'Response_Cmp3', 'Response_Cmp4', 'Response_Cmp5', and 'Response_Cmp1'). On the other hand, the second dataset, named ‘historicalDF2’, includes the same important variables as ‘X’, but also incorporates all the variables from the original ‘historicalDF’, apart from the previously removed 'Date_Adherence', 'BirthYear', and 'Name'.

2.5. Model Selection

In this section, we proceeded to train various predictive algorithms using the data that has been prepared up to this point. Subsequently, suitable model assessment metrics were employed to evaluate the performance of each algorithm and ascertain the optimal model for addressing the current problem.

The selection process for identifying potential customers who are more likely to purchase the new product involved testing several algorithms, such as Logistic Regression, K-Nearest Neighbor, Decision Trees, Neural Networks, Support-Vector Machine, XGBoost, and Random Forest (Appendix XVI -). These models and the respective hyperparameters can be described as following:

Logistic Regression: Logistic Regression is a statistical algorithm used for binary classification. It models the relationship between the features and the target variable using the logistic function and estimates the probabilities of the target class. This algorithm was attempted, but the F1 score was very low (0.248), so it was not considered. In the case of Logistic Regression, there are several hyperparameters that can be adjusted to optimize the model's performance. Some key hyperparameters include: Penalty (L1 and L2 regularization), C (Inverse of Regularization Strength) and Solver.

K-Nearest Neighbors (KNN): K-Nearest Neighbors (KNN) is a non-parametric algorithm that classifies data points based on their proximity to other data points. It assigns a label to a new data point by considering the majority label among its nearest neighbors. The hyperparameter n_neighbors in the KNN algorithm determines the number of neighbors to consider and significantly influences its performance and behavior. Choosing a small value for n_neighbors can make the algorithm more sensitive to noise and individual data points, potentially leading to overfitting. This can result in overly complex decision boundaries that may not generalize well to unseen data. To evaluate the KNN algorithm's performance for different values of n_neighbors, multiple instances of a KNeighborsClassifier were created, each with a specific value of n_neighbors. The average performance on the training and validation datasets was then calculated.

Decision Trees: Decision Trees are hierarchical structures that make decisions based on the values of features. They partition the data based on different feature thresholds and assign labels to the final leaf nodes. In the Decision Tree algorithm, the hyperparameter `max_depth` defines the number of levels our decision tree is going to have. Usually, a higher value leads to overfitting, while a lower value is prone to underfitting. Hence, different depths of the decision tree were tested.

Regarding the Neural Networks, SVM, and XGB algorithms, a thorough exploration of hyperparameters was conducted using techniques such as Grid Search and Random Search.

Neural Networks: Neural Networks are computational models inspired by the human brain's structure. They consist of interconnected layers of artificial neurons that learn from the data through a process called training. Neural Networks can be used for classification tasks. For Neural Networks, various hyperparameters were tested, including hidden layer sizes, learning rates, solvers, and activation functions. Hidden layer sizes determine the number of nodes in each layer, while learning rates control the step size in updating the weights during training. Solvers refer to the optimization algorithms used to train the neural

network, and activation functions define the nonlinear transformations applied to the input data at each node.

Support Vector Machine (SVM): SVM is a powerful algorithm that separates data points using a hyperplane. It aims to find the optimal hyperplane that maximizes the margin between different classes, leading to better classification performance. For SVM, different combinations of hyperparameters such as C, gammas, and Kernels were analyzed. The hyperparameter C controls the trade-off between maximizing the margin and minimizing the classification error. Gammas determine the influence of each training example and affect the smoothness of the decision boundary. Kernels define the type of function used to map the data into higher-dimensional spaces, allowing for more complex decision boundaries.

XGB: XGB, short for Extreme Gradient Boosting, is an ensemble algorithm based on gradient boosting. It combines the predictions of multiple weak models (decision trees) to form a strong predictive model. In the case of the XGB algorithm, different hyperparameters including depths, estimators, and gamma values were tested. The depth of an XGB model determines the maximum depth of each tree, while estimators refer to the number of boosting rounds. The gamma parameter controls the minimum loss reduction required for a split to occur during tree construction.

Random Forest: is an ensemble algorithm that combines multiple decision trees to create a robust predictive model. It leverages the power of decision trees by aggregating their predictions to make accurate and reliable predictions. In the case of Random Forest, various hyperparameters can be adjusted to optimize the model's performance. Some of the key hyperparameters include: i) number of Trees, ii) the maximum depth for each decision tree in the Random Forest, iii) number of features and iv) the minimum number of samples required to perform a split during tree construction.

To ensure a reliable assessment of these techniques, it was crucial to evaluate the performance of the chosen models on the data. Initially, default hyperparameter values were used for the models. The selection process involved identifying models with the highest f1-scores on the validation data while also considering the models with minimal differences between the training and validation datasets. This pre-filtering step aimed to exclude models that performed poorly and prioritize models that demonstrated both strong performance on the validation data and consistent performance across different datasets.

Based on the results and evaluation, the Neural Network emerged as the top-performing model among all the models that were tested. More precisely, the Neural Network is the one that presents the highest score in the validation subset, meaning that the model performs well on unseen data. Moreover, the model has a low delta between training and validation, suggesting that overfitting also might not be a concern. Lastly, it also has a high training f1-score, meaning that the model can capture complex patterns and relationships within the data.

The results can be summarized as follows:

Algorithms	Train	Validation	Delta Train - Validation
Logistic Regression	0.554+/-0.02	0.535+/-0.06	0.019

KNN	0.657+/-0.02	0.493+/-0.08	0.164
DT	0.528+/-0.06	0.479+/-0.1	0.049
NN	0.763+/-0.02	0.628+/-0.05	0.118
SVM	0.562+/-0.02	0.444+/-0.06	0.118
XGB	1.0+/-0.0	0.635+/-0.03	0.365
RF	1.0+/-0.0	0.611+/-0.03	0.389

Table 1 - The scores in the training and validation sets

A point plot was created for the Neural Network, which will determine the best values as well as the best number of layers to use moving forward (Appendix XV -).

To ensure the selection of the best model, it is vital to identify the optimal set of parameters. This involves fine-tuning the models with the objective of maximizing the f1-score and minimizing the discrepancy between the training dataset and the validation dataset. The optimal set of parameters can be identified by utilizing model optimization techniques such as Grid Search and Random Search. In this case, both techniques were evaluated, and it was determined that Random Search was more suitable due to its efficiency while providing similar parameter values as Grid Search.

After performing hyperparameter tuning and identifying the optimal set of hyperparameters for each algorithm, the models were retrained using the updated parameter values. The Neural Network had an improvement in the validation f1-score, while the train-validation- F1 difference reduced, leading to less overfitting.

In this case, the Logistic Regression, KNN, Decision Tree, Neural Network, SVM, XGB, and Random Forest models were also compared by plotting an ROC curve. Based on the results, it appears that the NN model outperformed the other four tested algorithms, making it the most favorable choice among the five models (Appendix XVII - Appendix XVII - Appendix XVII -).

In addition to the previous steps, we conducted an analysis to optimize the performance of our models by adjusting the prediction threshold. Our focus was to maximize the F1-Score, which provides a balanced measure of precision and recall. By default, the threshold for classifying predictions is set at 0.5, meaning that if the predicted probability of an observation belonging to the positive class is equal to or higher than 0.5, it is considered a positive prediction; otherwise, it is considered a negative prediction. However, we recognized that different scenarios may require different trade-offs between precision and recall. By adjusting the threshold, we could emphasize either precision or recall, depending on the specific requirements of the problem (Appendix XVIII -). Based on the obtained graph, the best threshold value is 0.297991.

In order to identify the optimal threshold for the best model, the precision_recall_curve function was used and corresponding thresholds were extracted from the precision-recall curve, allowing for a comprehensive analysis. The threshold value associated with the highest F1 score was determined. This F1 score represents the balance between precision and recall that yields the best overall performance for the model. Additionally, in an attempt to address the issue of class imbalance in our target variable, we

implemented the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE is an oversampling technique that generates synthetic samples from the minority class, aiming to achieve a more balanced distribution of classes in the training set. This technique involves creating synthetic instances based on the existing minority samples, effectively expanding the representation of the minority class.

However, despite the application of SMOTE, the performance did not meet our expectations. As a result, we decided not to proceed with it.

2.6. Interpret

Finally, after fine-tuning our algorithms and optimizing their parameters, we applied them to the predict dataset to make predictions on new data. To further enhance the predictive power and robustness of our models, we created an ensemble model. The ensemble model was constructed using a majority voting approach, where each algorithm in our collection of fine-tuned models had an equal vote. By aggregating the predictions from multiple models, we aimed to leverage the diverse strengths and capabilities of each individual algorithm. By combining the predictive power of multiple algorithms through the ensemble model, we sought to improve the overall performance and reliability of our predictions on the predicted dataset.

3. Conclusion

To emphasize the importance of this analysis, it is crucial to highlight the following key points. By employing predictive modeling, we successfully optimized the profitability of the upcoming sixth direct marketing campaign for the company. Our predictions were focused on gauging the customers' receptiveness towards the campaign, which aimed at promoting a new frozen food product.

To conclude, following a thorough evaluation of the data, we have determined that the Neural Network model provides the most accurate predictions for predicting client acceptance of the marketing campaign, with an f1-score of 0.39.

Based on the analysis and the performance of our model, we have decided to proactively reach out to a specific group of clients. We will leverage the insights and predictions provided by the model to target this group. Based on our estimations, we anticipate a positive response rate of 15.04%, which corresponds to approximately 376 clients.

By focusing our efforts on this particular group, we aim to maximize the effectiveness of our marketing or outreach campaign and optimize the allocation of resources. It is important to note that these estimations are based on the performance of the model and may be subject to some level of uncertainty.

Nevertheless, with a proactive approach and targeted engagement, we anticipate a positive response from a significant number of clients, contributing to the overall success of our campaign.

4. References

Abbott, D. (2014). Applied Predictive Analytics (1st ed.). Wiley. Retrieved April 2023 from https://www.perlego.com/book/999581/applied-predictive-analytics-principles-and-techniques-for-the-professional-data-analyst-pdf (Original work published 2014)

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

Mitchell, T., (1997) "Machine Learning", McGraw Hill.

5. Appendices

Appendix I - Information on the "historicalDF" data set

The "historicalDF" data set had a total of 2518 records distributed across 30 different columns with data concerning customers' spending patterns, from the costumers' expenses on each menu, to whether they do so in person or takeaway, their usage of the Spice Alley app by access and purchases through it, Recency and Complaints.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2518 entries, 0 to 2517
Data columns (total 30 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   CustomerID            2518 non-null   int64
 1   Name                  2518 non-null   object
 2   Birthyear             2518 non-null   int64
 3   Education             2485 non-null   object
 4   Marital_Status        2518 non-null   object
 5   Income                2518 non-null   int64
 6   Kid_Younger6          2518 non-null   int64
 7   Children_6to18        2518 non-null   int64
 8   Date_Adherence        2518 non-null   object
 9   Recency               2470 non-null   float64
 10  MntMeat&Fish          2518 non-null   int64
 11  MntEntries            2518 non-null   int64
 12  MntVegan&Vegetarian   2518 non-null   int64
 13  MntDrinks             2497 non-null   float64
 14  MntDesserts           2518 non-null   int64
 15  MntAdditionalRequests 2518 non-null   int64
 16  NumOfferPurchases     2518 non-null   int64
 17  NumAppPurchases       2518 non-null   int64
 18  NumTakeAwayPurchases  2518 non-null   int64
 19  NumStorePurchases     2518 non-null   int64
 20  NumAppVisitsMonth     2518 non-null   int64
 21  Response_Cmp2         2518 non-null   int64
 22  Response_Cmp3         2518 non-null   int64
 23  Response_Cmp4         2518 non-null   int64
 24  Response_Cmp5         2518 non-null   int64
 25  Response_Cmp1         2518 non-null   int64
 26  Complain              2518 non-null   int64
 27  CostContact           2518 non-null   int64
 28  Revenue               2518 non-null   int64
 29  DepVar                2518 non-null   int64
dtypes: float64(2), int64(24), object(4)
memory usage: 590.3+ KB
```

Figure 1 - Historical data information

Appendix II - Information on the "predictDF" data set

The "predictDF" data set had 2500 records across 29 columns. It follows the same structure as historicalDF, except in the non-null count and, date type, most importantly, without the ground truth associated with each customer (DepVar variable).

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2500 entries, 0 to 2499
Data columns (total 29 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   CustomerID             2500 non-null   int64
 1   Name                   2500 non-null   object
 2   Birthyear              2500 non-null   int64
 3   Education              2479 non-null   object
 4   Marital_Status         2500 non-null   object
 5   Income                 2500 non-null   int64
 6   Kid_Younger6           2500 non-null   int64
 7   Children_6to18         2500 non-null   int64
 8   Date_Adherence         2500 non-null   datetime64[ns]
 9   Recency                2488 non-null   float64
 10  MntMeat&Fish           2500 non-null   int64
 11  MntEntries             2500 non-null   int64
 12  MntVegan&Vegetarian    2500 non-null   int64
 13  MntDrinks              2493 non-null   float64
 14  MntDesserts            2500 non-null   int64
 15  MntAdditionalRequests  2500 non-null   int64
 16  NumOfferPurchases      2500 non-null   int64
 17  NumAppPurchases        2500 non-null   int64
 18  NumTakeAwayPurchases   2500 non-null   int64
 19  NumStorePurchases      2500 non-null   int64
 20  NumAppVisitsMonth      2500 non-null   int64
 21  Response_Cmp2          2500 non-null   int64
 22  Response_Cmp3          2500 non-null   int64
 23  Response_Cmp4          2500 non-null   int64
 24  Response_Cmp5          2500 non-null   int64
 25  Response_Cmp1          2500 non-null   int64
 26  Complain               2500 non-null   int64
 27  CostContact            2500 non-null   int64
 28  Revenue                2500 non-null   int64
dtypes: datetime64[ns](1), float64(2), int64(23), object(3)
memory usage: 566.5+ KB
```

Figure 2 - Predict data information

Appendix III - Historical data set: Categorical and Numerical Variables Distribution

	Categorical Variables	Numerical Variables
0	Name	CustomerID
1	Education	Birthyear
2	Marital_Status	Income
3	Date_Adherence	Kid_Younger6
4	None	Children_6to18
5	None	Recency
6	None	MntMeat&Fish
7	None	MntEntries
8	None	MntVegan&Vegetarian
9	None	MntDrinks
10	None	MntDesserts
11	None	MntAdditionalRequests
12	None	NumOfferPurchases
13	None	NumAppPurchases
14	None	NumTakeAwayPurchases
15	None	NumStorePurchases
16	None	NumAppVisitsMonth
17	None	Response_Cmp2
18	None	Response_Cmp3
19	None	Response_Cmp4
20	None	Response_Cmp5
21	None	Response_Cmp1
22	None	Complain
23	None	CostContact
24	None	Revenue
25	None	DepVar

Figure 3 - Categorical and Numerical variables' distribution

Appendix IV - Historical data set: Correlation between all variables

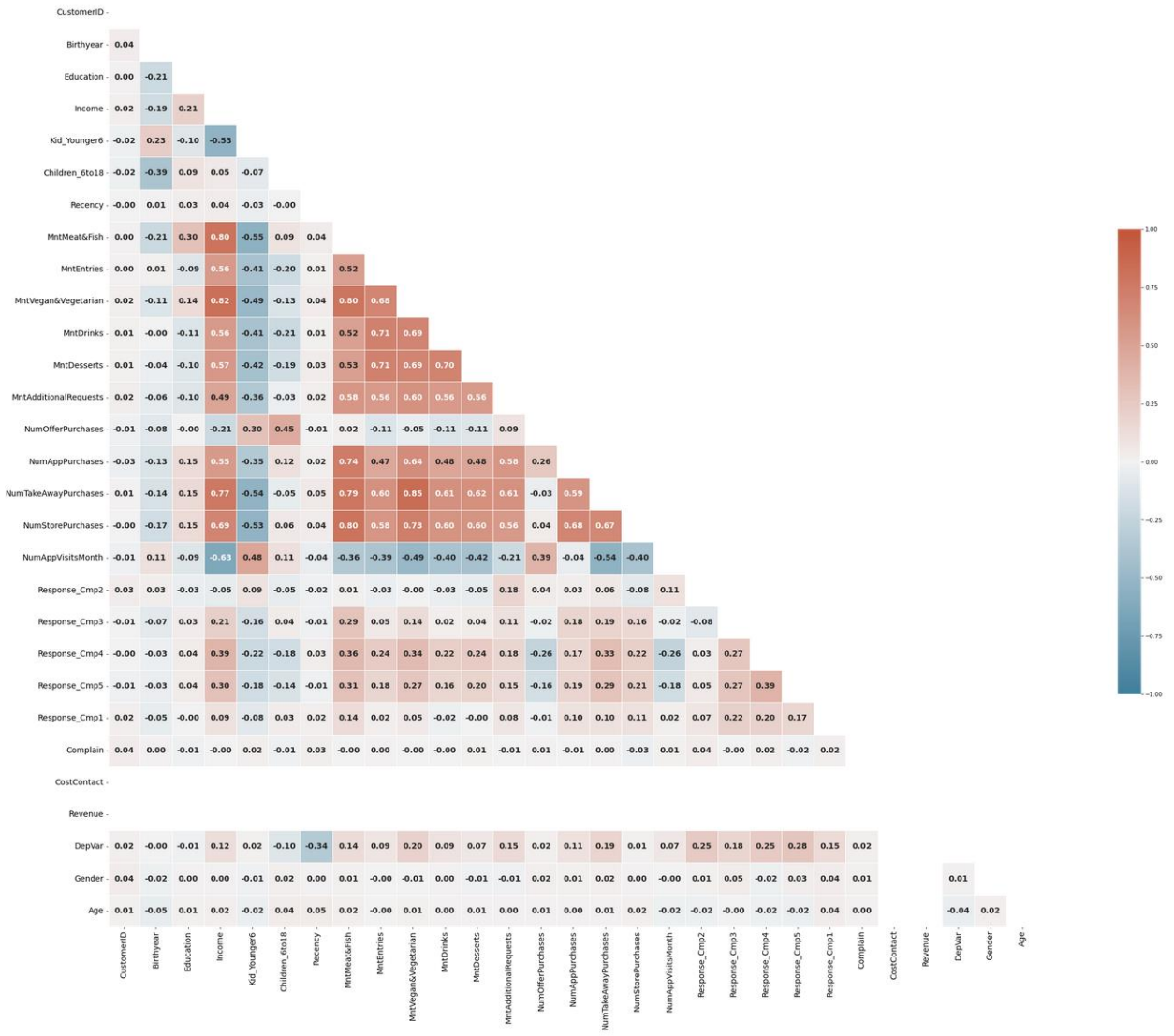


Figure 4 - Correlation between all variables

Appendix V - Historical dataframe: Summary statistics (count, mean, standard deviation, min, Q1 (25%), Q2(50%), Q3(75%) and max)

	count	mean	std	min	25%	50%	75%	max
CustomerID	2500.0	18190.798800	8722.377016	3114.0	10453.5	18232.5	25846.50	33093.0
Birthyear	2500.0	1976.620800	11.897631	1948.0	1967.0	1978.0	1985.00	2005.0
Income	2500.0	77557.227600	35505.417110	2678.0	50998.0	76684.5	101712.00	237117.0
Kid_Younger6	2500.0	0.439200	0.543343	0.0	0.0	0.0	1.00	2.0
Children_6to18	2500.0	0.500800	0.544535	0.0	0.0	0.0	1.00	2.0
Recency	2452.0	48.983279	28.636798	0.0	25.0	48.0	73.00	99.0
MntMeat&Fish	2500.0	3071.254400	3376.433081	0.0	240.0	1795.0	5050.00	14980.0
MntEntries	2500.0	526.582400	761.351600	0.0	40.0	180.0	680.00	3980.0
MntVegan&Vegetarian	2500.0	2748.278800	3875.425530	5.0	225.0	1110.0	3693.75	24886.0
MntDrinks	2479.0	545.916499	793.028804	0.0	40.0	180.0	700.00	3960.0
MntDesserts	2500.0	524.163200	763.868740	0.0	40.0	180.0	680.00	3980.0
MntAdditionalRequests	2500.0	42.555600	49.576031	0.0	8.0	24.0	57.00	249.0
NumOfferPurchases	2500.0	2.454400	2.300356	0.0	1.0	2.0	3.00	16.0
NumAppPurchases	2500.0	5.996800	2.757214	0.0	4.0	6.0	8.00	13.0
NumTakeAwayPurchases	2500.0	3.852400	3.425800	0.0	1.0	3.0	5.00	24.0
NumStorePurchases	2500.0	5.828400	3.339134	0.0	3.0	5.0	8.00	13.0
NumAppVisitsMonth	2500.0	5.292800	2.712860	0.0	3.0	6.0	7.00	20.0
Response_Cmp2	2500.0	0.081200	0.273197	0.0	0.0	0.0	0.00	1.0
Response_Cmp3	2500.0	0.068000	0.251796	0.0	0.0	0.0	0.00	1.0
Response_Cmp4	2500.0	0.078800	0.269480	0.0	0.0	0.0	0.00	1.0
Response_Cmp5	2500.0	0.063600	0.244088	0.0	0.0	0.0	0.00	1.0
Response_Cmp1	2500.0	0.012000	0.108907	0.0	0.0	0.0	0.00	1.0
Complain	2500.0	0.012800	0.112433	0.0	0.0	0.0	0.00	1.0
CostContact	2500.0	3.000000	0.000000	3.0	3.0	3.0	3.00	3.0
Revenue	2500.0	16.000000	0.000000	16.0	16.0	16.0	16.00	16.0
DepVar	2500.0	0.125200	0.331012	0.0	0.0	0.0	0.00	1.0

Figure 5 - Summary statistics

Appendix VI - Visual Exploration of Numerical Variables

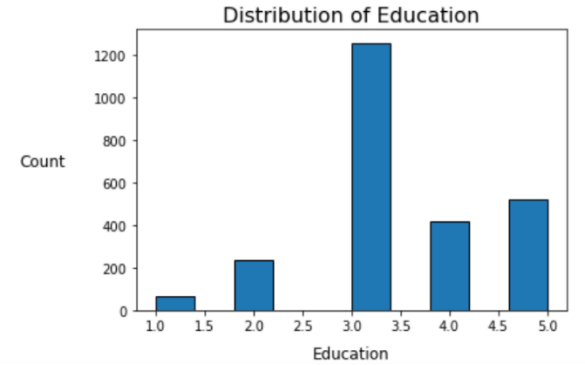


Figure 6 - Education Histogram

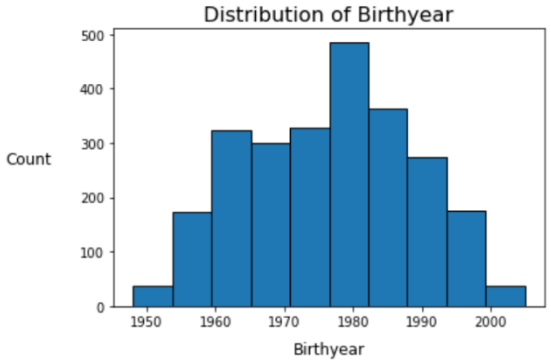


Figure 7 - Birthyear Histogram

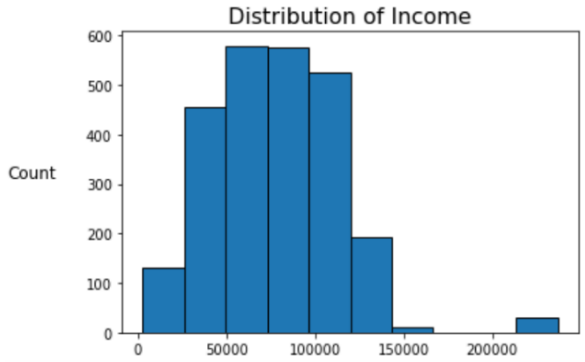


Figure 8 – Income Histogram

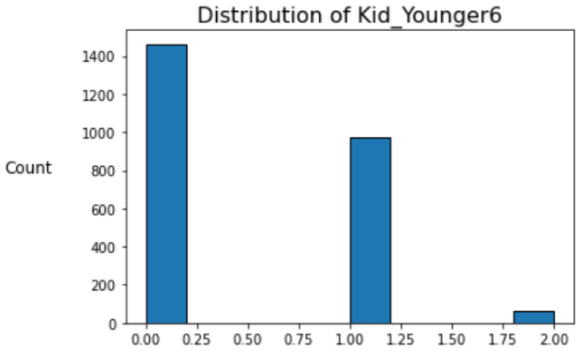


Figure 9 – Nº of kids younger than 6 Histogram

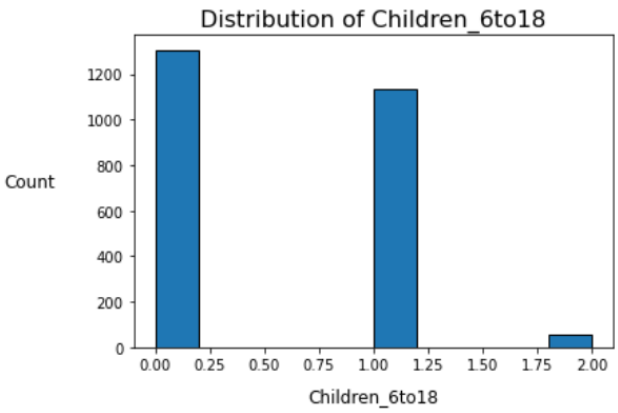


Figure 10 - Nº of kids between 6-18 years old Histogram

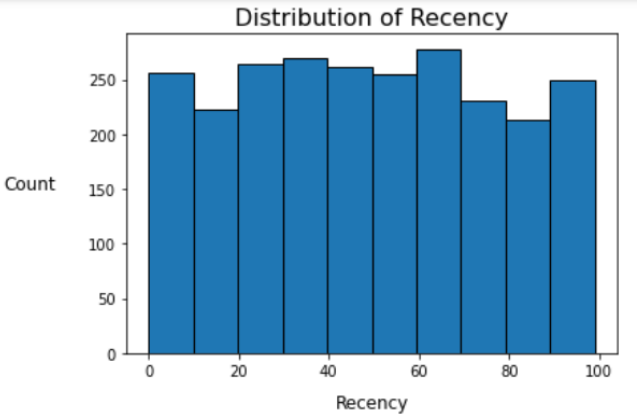


Figure 11 - Recency Histogram

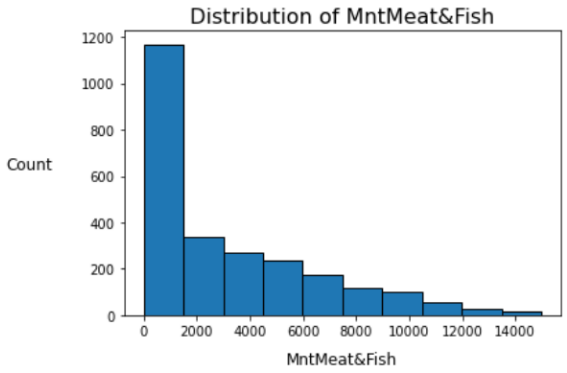


Figure 12 - MntMeat&Fish Histogram

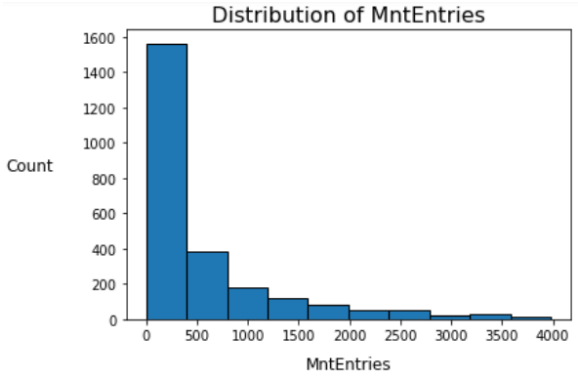


Figure 13 - MntEntries Histogram

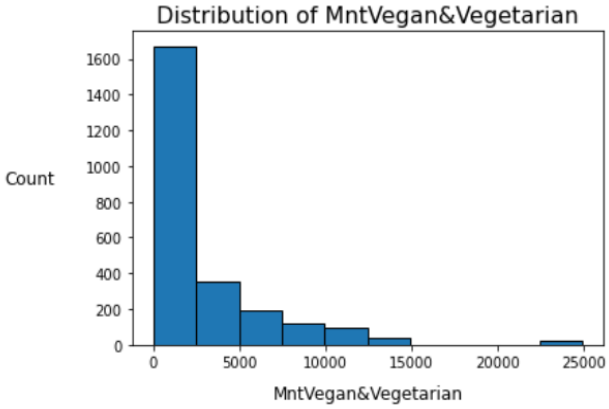


Figure 15 - MntVegan&Vegetarian Histogram

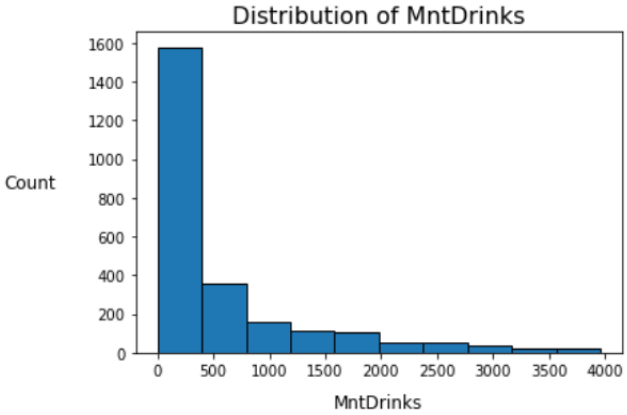


Figure 14 - MntDrinks Histogram

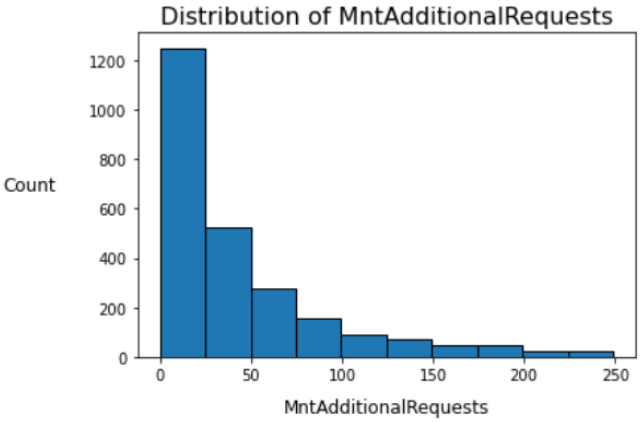


Figure 16 – MntAdditionalRequests Histogram

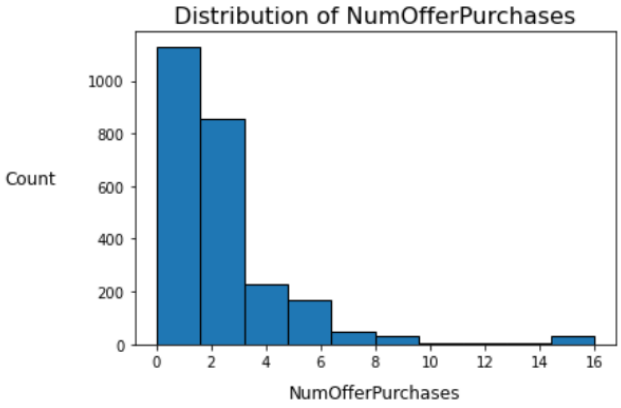


Figure 17 - NumOfferPurchases Histogram

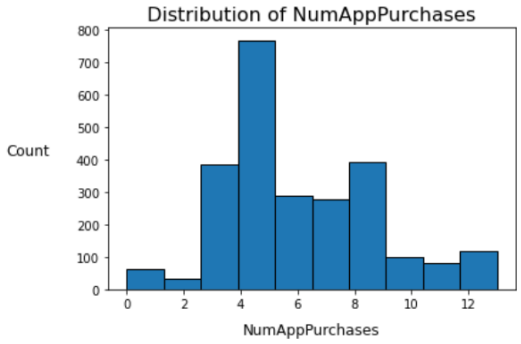


Figure 18 - NumAppPurchases Histogram

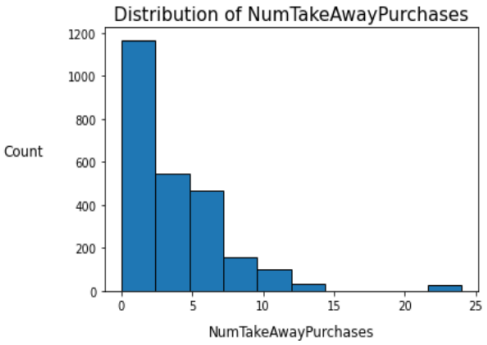


Figure 19 - NumTakeAwayPurchases Histogram

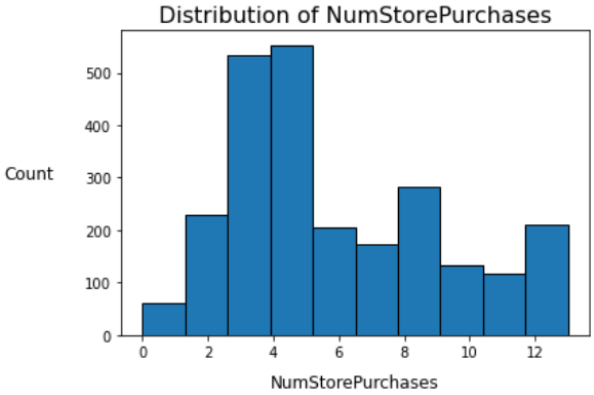


Figure 20 – NumStorePurchases Histogram

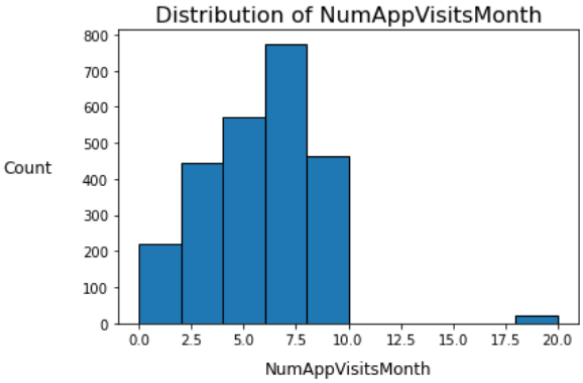


Figure 21 - NumAppVisitsMonth Histogram

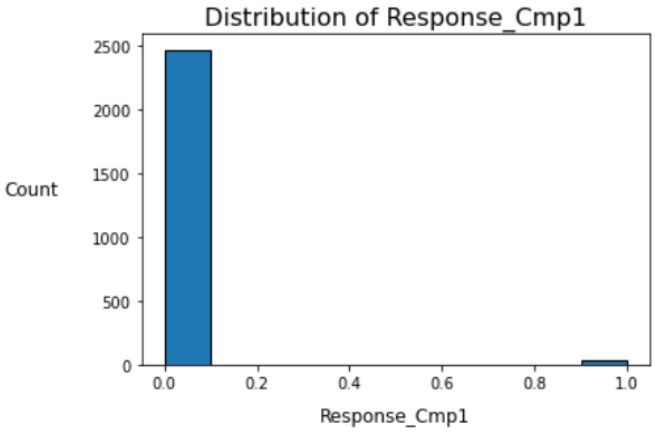


Figure 22 - Response_Cmp1 Histogram

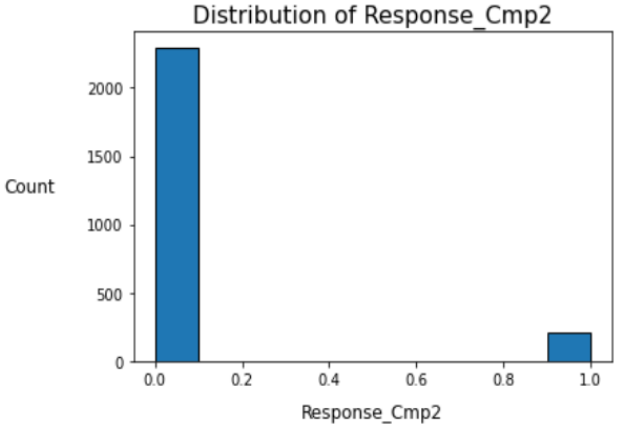


Figure 23 - Response_Cmp2 Histogram

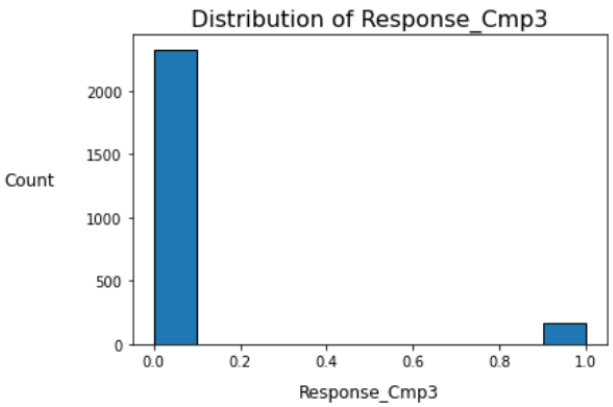


Figure 25 - Response_Cmp3 Histogram

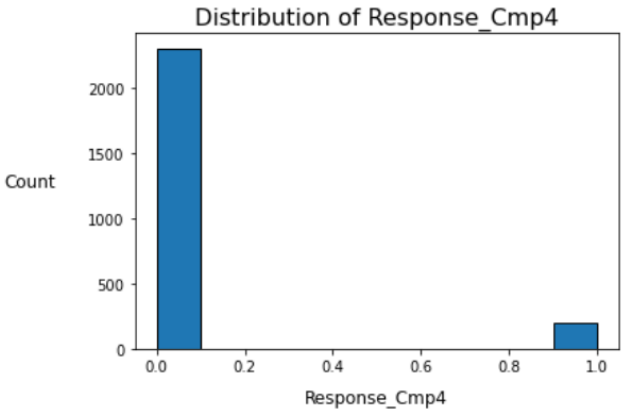


Figure 24 - Response_Cmp4 Histogram

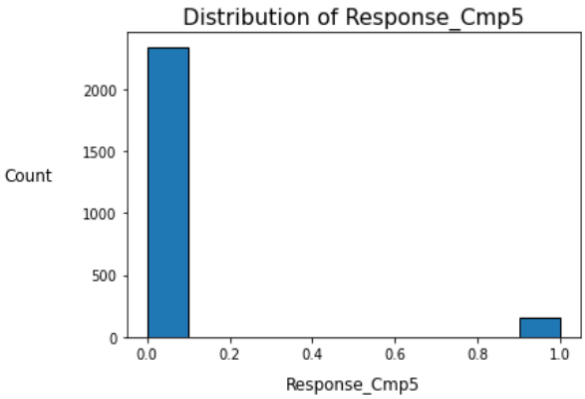


Figure 27 - Response_Cmp5 Histogram

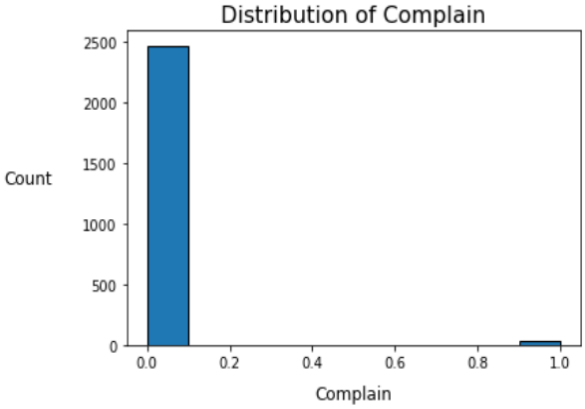


Figure 26 - Complain Histogram

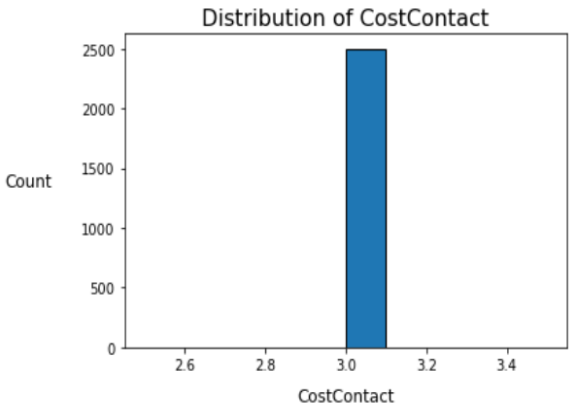


Figure 28 - CostContact Histogram

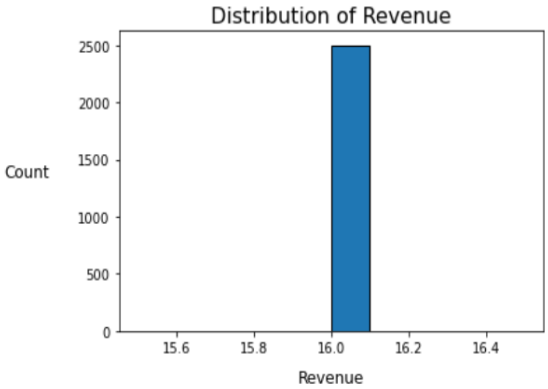


Figure 29 - Revenue Histogram

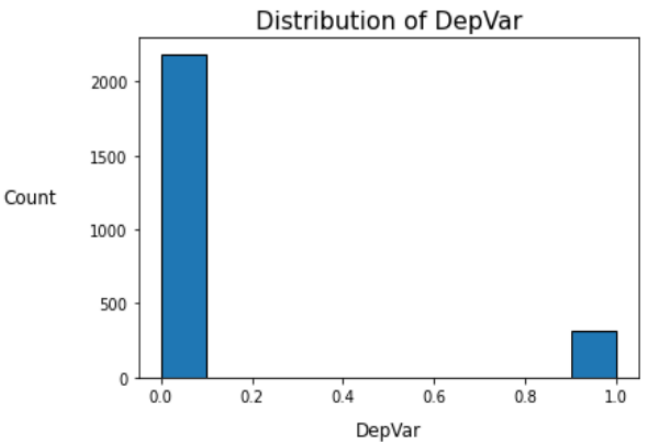


Figure 31 - DepVar Histogram

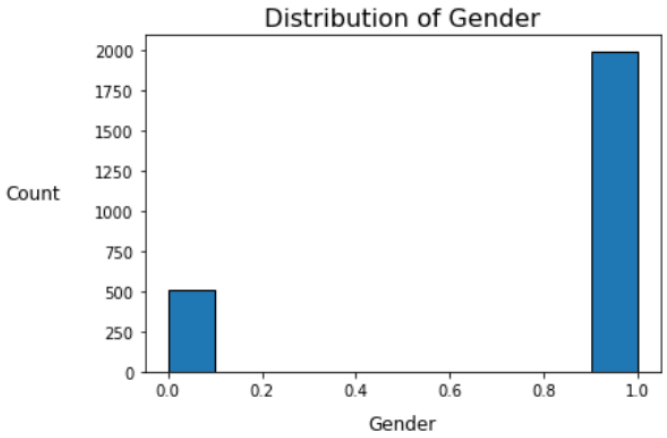


Figure 30 - Gender Histogram

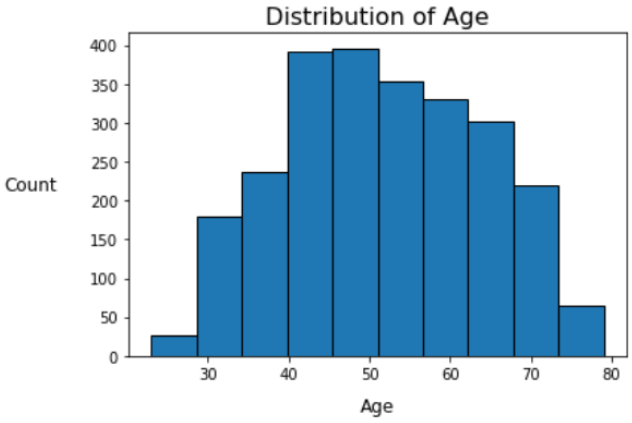


Figure 32 - Age Histogram

Appendix VII - SPLITS Categorical variables

SPLIT 1

```
Response_Cmp2 is IMPORTANT for Prediction
Response_Cmp3 is IMPORTANT for Prediction
Response_Cmp4 is IMPORTANT for Prediction
Response_Cmp5 is IMPORTANT for Prediction
Response_Cmp1 is IMPORTANT for Prediction
Complain is NOT an important predictor. (Discard Complain from model)
Gender is NOT an important predictor. (Discard Gender from model)
Education is NOT an important predictor. (Discard Education from model)
Marital_Status is IMPORTANT for Prediction
```

SPLIT 2

```
Response_Cmp2 is IMPORTANT for Prediction
Response_Cmp3 is IMPORTANT for Prediction
Response_Cmp4 is IMPORTANT for Prediction
Response_Cmp5 is IMPORTANT for Prediction
Response_Cmp1 is IMPORTANT for Prediction
Complain is NOT an important predictor. (Discard Complain from model)
Gender is NOT an important predictor. (Discard Gender from model)
Education is NOT an important predictor. (Discard Education from model)
Marital_Status is IMPORTANT for Prediction
```

SPLIT 3

```
Response_Cmp2 is IMPORTANT for Prediction
Response_Cmp3 is IMPORTANT for Prediction
Response_Cmp4 is IMPORTANT for Prediction
Response_Cmp5 is IMPORTANT for Prediction
Response_Cmp1 is IMPORTANT for Prediction
Complain is NOT an important predictor. (Discard Complain from model)
Gender is NOT an important predictor. (Discard Gender from model)
Education is NOT an important predictor. (Discard Education from model)
Marital_Status is NOT an important predictor. (Discard Marital_Status from model)
```

SPLIT 4

```
Response_Cmp2 is IMPORTANT for Prediction
Response_Cmp3 is IMPORTANT for Prediction
Response_Cmp4 is IMPORTANT for Prediction
Response_Cmp5 is IMPORTANT for Prediction
Response_Cmp1 is IMPORTANT for Prediction
Complain is NOT an important predictor. (Discard Complain from model)
Gender is NOT an important predictor. (Discard Gender from model)
Education is NOT an important predictor. (Discard Education from model)
Marital_Status is IMPORTANT for Prediction
```

```
                                        SPLIT 5
_____________________________________________________________________________________________________
Response_Cmp2 is IMPORTANT for Prediction
Response_Cmp3 is IMPORTANT for Prediction
Response_Cmp4 is IMPORTANT for Prediction
Response_Cmp5 is IMPORTANT for Prediction
Response_Cmp1 is IMPORTANT for Prediction
Complain is NOT an important predictor. (Discard Complain from model)
Gender is NOT an important predictor. (Discard Gender from model)
Education is IMPORTANT for Prediction
Marital_Status is IMPORTANT for Prediction
```

Appendix VIII - The best feature for categorical variables

Predictor	Chi-Square	What to do? (One possible way to "solve")
Gender	5 NO	Remove
Education	1 YES & 4 NO	Remove
Marital_Status	4 YES & 1 No	Keep
Response_Cmp2	5 YES	Keep
Response_Cmp3	5 YES	Keep
Response_Cmp4	5 YES	Keep
Response_Cmp5	5 YES	Keep
Response_Cmp1	5 YES	Keep
Complain	5 No	Remove

Figure 33 - The best feature for categorical variables

Appendix IX - SPLITS Numerical variables

SPLIT 1

```
Income                  1.280089e+09
Kid_Younger6            2.953654e-01
Children_6to18          2.931343e-01
Recency                 8.000596e+02
MntMeat&Fish            1.133034e+07
MntEntries              5.661167e+05
MntVegan&Vegetarian     1.531684e+07
MntDrinks               6.164453e+05
MntDesserts             5.681426e+05
MntAdditionalRequests   2.402257e+03
NumOfferPurchases       5.439750e+00
NumAppPurchases         7.553717e+00
NumTakeAwayPurchases    1.179090e+01
NumStorePurchases       1.099567e+01
NumAppVisitsMonth       7.420221e+00
Response_Cmp2           7.405678e-02
Response_Cmp3           6.340770e-02
Response_Cmp4           7.321636e-02
Response_Cmp5           5.993397e-02
Response_Cmp1           1.283742e-02
CostContact             0.000000e+00
Revenue                 0.000000e+00
Age                     1.470914e+02
dtype: float64
```

SPLIT 2

```
Income                  1.238134e+09
Kid_Younger6            2.873677e-01
Children_6to18          2.971396e-01
Recency                 8.052042e+02
MntMeat&Fish            1.151812e+07
MntEntries              5.781923e+05
MntVegan&Vegetarian     1.497428e+07
MntDrinks               6.446142e+05
MntDesserts             5.856480e+05
MntAdditionalRequests   2.513161e+03
NumOfferPurchases       5.056388e+00
NumAppPurchases         7.597849e+00
NumTakeAwayPurchases    1.153812e+01
NumStorePurchases       1.154115e+01
NumAppVisitsMonth       7.411681e+00
Response_Cmp2           7.025913e-02
Response_Cmp3           6.727739e-02
Response_Cmp4           7.447624e-02
Response_Cmp5           6.340770e-02
Response_Cmp1           1.186193e-02
CostContact             0.000000e+00
Revenue                 0.000000e+00
Age                     1.442374e+02
dtype: float64
```

SPLIT 3

```
Income                  1.241573e+09
Kid_Younger6            2.942311e-01
Children_6to18          2.961278e-01
Recency                 8.182410e+02
MntMeat&Fish            1.146822e+07
MntEntries              5.944042e+05
MntVegan&Vegetarian     1.429153e+07
MntDrinks               6.246932e+05
MntDesserts             5.752959e+05
MntAdditionalRequests   2.461895e+03
NumOfferPurchases       5.074008e+00
NumAppPurchases         7.553416e+00
NumTakeAwayPurchases    1.141514e+01
NumStorePurchases       1.098193e+01
NumAppVisitsMonth       7.320164e+00
Response_Cmp2           7.988369e-02
Response_Cmp3           5.993397e-02
Response_Cmp4           7.195198e-02
Response_Cmp5           5.510530e-02
Response_Cmp1           1.137344e-02
CostContact             0.000000e+00
Revenue                 0.000000e+00
Age                     1.420394e+02
dtype: float64
```

SPLIT 4

```
Income                  1.271136e+09
Kid_Younger6            2.983669e-01
Children_6to18          2.951436e-01
Recency                 8.080442e+02
MntMeat&Fish            1.131487e+07
MntEntries              5.902400e+05
MntVegan&Vegetarian     1.514236e+07
MntDrinks               6.372638e+05
MntDesserts             6.068435e+05
MntAdditionalRequests   2.459510e+03
NumOfferPurchases       5.591415e+00
NumAppPurchases         7.693136e+00
NumTakeAwayPurchases    1.200501e+01
NumStorePurchases       1.110810e+01
NumAppVisitsMonth       7.541811e+00
Response_Cmp2           7.279540e-02
Response_Cmp3           6.513257e-02
Response_Cmp4           7.321636e-02
Response_Cmp5           6.167484e-02
Response_Cmp1           1.186193e-02
CostContact             0.000000e+00
Revenue                 0.000000e+00
Age                     1.426596e+02
dtype: float64
```

SPLIT 5

```
Income                  1.272019e+09
Kid_Younger6            3.005493e-01
Children_6to18          3.011346e-01
Recency                 8.033280e+02
MntMeat&Fish            1.136555e+07
MntEntries              5.693332e+05
MntVegan&Vegetarian     1.536732e+07
MntDrinks               6.070566e+05
MntDesserts             5.812907e+05
MntAdditionalRequests   2.451444e+03
NumOfferPurchases       5.296486e+00
NumAppPurchases         7.611375e+00
NumTakeAwayPurchases    1.192970e+01
NumStorePurchases       1.111269e+01
NumAppVisitsMonth       7.097492e+00
Response_Cmp2           7.614907e-02
Response_Cmp3           6.124037e-02
Response_Cmp4           7.025913e-02
Response_Cmp5           5.774662e-02
Response_Cmp1           1.137344e-02
CostContact             0.000000e+00
Revenue                 0.000000e+00
Age                     1.456533e+02
dtype: float64
```

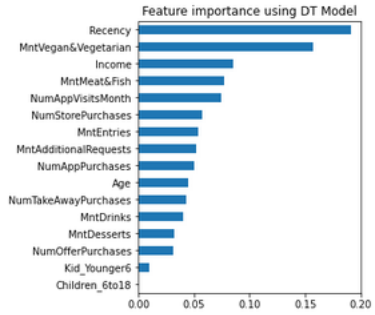
Appendix X - The Spearman correlation

Redundant Variables - Spearman Correlation (Correlation higher than |0.7|)

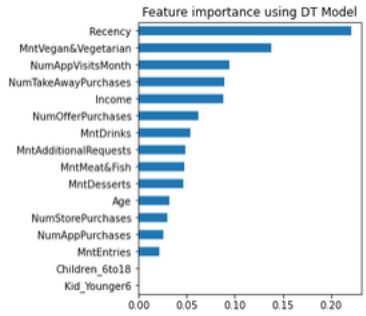
	Feature 1	Feature 2	split 1	split 2	split 3	split 4	split 5	total
0	Income	MntMeat&Fish	1	1	1	1	1	5
1	Income	MntVegan&Vegetarian	1	1	1	1	1	5
2	Income	NumTakeAwayPurchases	1	1	1	1	1	5
3	MntDesserts	MntDrinks	1	1	1	0	0	3
4	MntDesserts	MntEntries	1	1	1	1	1	5
5	MntDrinks	MntDesserts	1	1	1	0	0	3
6	MntDrinks	MntEntries	1	1	1	1	1	5
7	MntEntries	MntDesserts	1	1	1	1	1	5
8	MntEntries	MntDrinks	1	1	1	1	1	5
9	MntMeat&Fish	Income	1	1	1	1	1	5
10	MntMeat&Fish	MntVegan&Vegetarian	1	1	1	1	1	5
11	MntMeat&Fish	NumAppPurchases	1	1	1	1	1	5
12	MntMeat&Fish	NumStorePurchases	1	1	1	1	1	5
13	MntMeat&Fish	NumTakeAwayPurchases	1	1	1	1	1	5
14	MntVegan&Vegetarian	Income	1	1	1	1	1	5
15	MntVegan&Vegetarian	MntMeat&Fish	1	1	1	1	1	5
16	MntVegan&Vegetarian	NumStorePurchases	1	1	1	1	0	4
17	MntVegan&Vegetarian	NumTakeAwayPurchases	1	1	1	1	1	5
18	NumAppPurchases	MntMeat&Fish	1	1	1	1	1	5
19	NumStorePurchases	MntMeat&Fish	1	1	1	1	1	5
20	NumStorePurchases	MntVegan&Vegetarian	1	1	1	1	0	4
21	NumTakeAwayPurchases	Income	1	1	1	1	1	5
22	NumTakeAwayPurchases	MntMeat&Fish	1	1	1	1	1	5
23	NumTakeAwayPurchases	MntVegan&Vegetarian	1	1	1	1	1	5

Appendix XI - Feature importance using DT Model

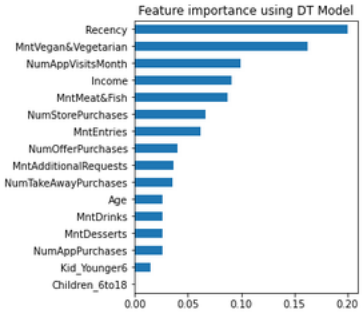
SPLIT 1



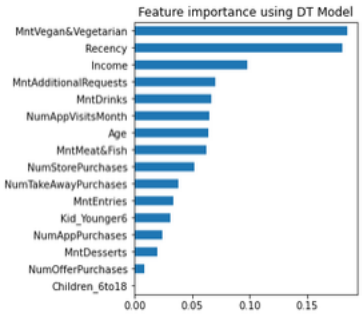
SPLIT 2



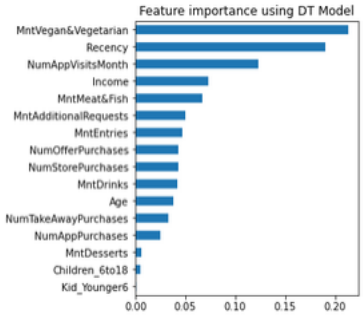
SPLIT 3



SPLIT 4



SPLIT 5

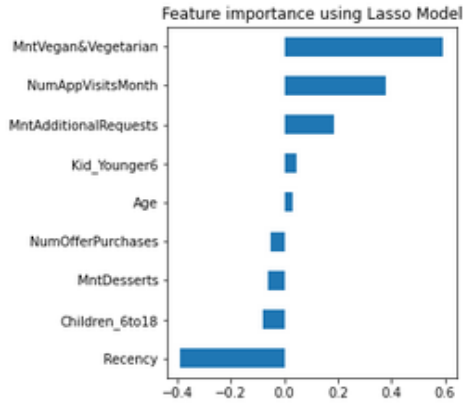

Feature importance using DT Model
MntVegan&Vegetarian
Recency
NumAppVisitsMonth
Income
MntMeat&Fish
MntAdditionalRequests
MntEntries
NumOfferPurchases
NumStorePurchases
MntDrinks
Age
NumTakeAwayPurchases
NumAppPurchases
MntDesserts
Children_6to18
Kid_Younger6
0.00
0.05
0.10
0.15
0.20

Appendix XII - Feature importance using RFE, Lasso and Decision Trees Models

SPLIT 1

```
---------- RFE ----------
Kid_Younger6             False
Children_6to18            True
Recency                   True
MntVegan&Vegetarian       True
MntDesserts              False
MntAdditionalRequests     True
NumOfferPurchases        False
NumAppVisitsMonth         True
Age                      False
dtype: bool
```

---------- LASSO ----------



---------- DT ----------

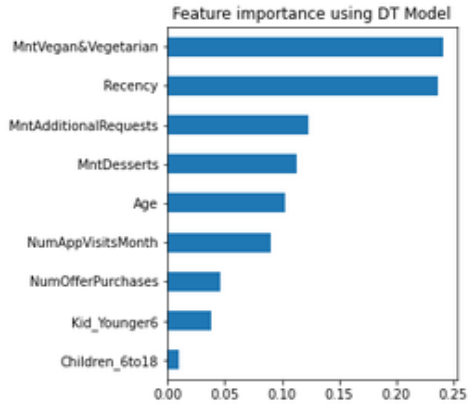


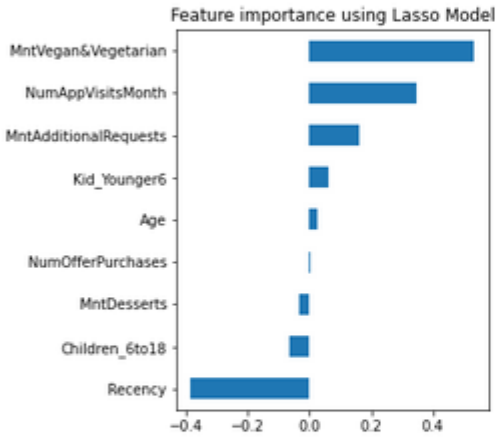
Figure 34 - Split 1

SPLIT 2

---------- RFE ----------

```
Kid_Younger6             False
Children_6to18            True
Recency                   True
MntVegan&Vegetarian       True
MntDesserts              False
MntAdditionalRequests     True
NumOfferPurchases        False
NumAppVisitsMonth         True
Age                      False
dtype: bool
```

---------- LASSO ----------



---------- DT ----------

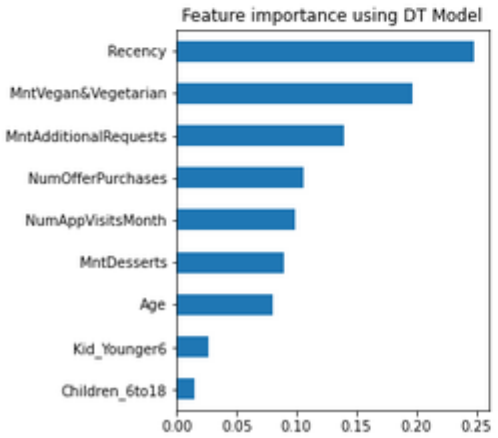


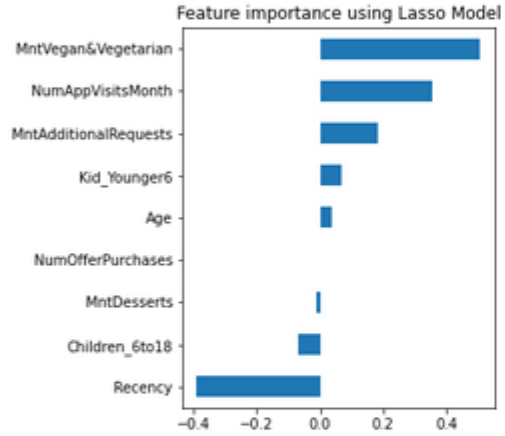
Figure 35 - Split 2

SPLIT 3

---------- RFE ----------

```
Kid_Younger6             False
Children_6to18            True
Recency                   True
MntVegan&Vegetarian       True
MntDesserts              False
MntAdditionalRequests     True
NumOfferPurchases        False
NumAppVisitsMonth         True
Age                      False
dtype: bool
```

---------- LASSO ----------



---------- DT ----------

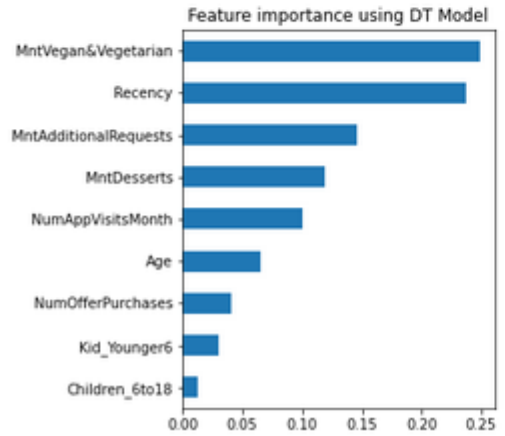
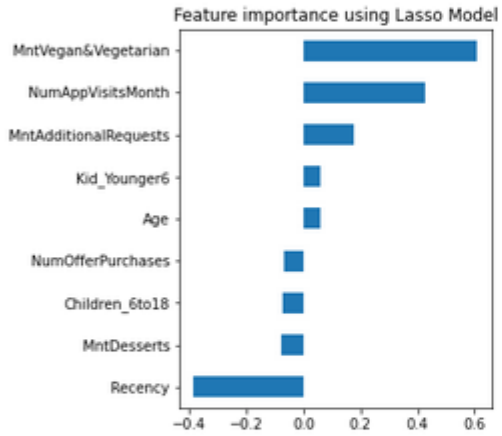


Figure 36 - Split 3

SPLIT 4

```
----------------- RFE ----------------------
Kid_Younger6              False
Children_6to18             True
Recency                    True
MntVegan&Vegetarian        True
MntDesserts               False
MntAdditionalRequests      True
NumOfferPurchases         False
NumAppVisitsMonth          True
Age                       False
dtype: bool
```

```
----------------- LASSO ----------------------
```




```
----------------- DT ----------------------
```

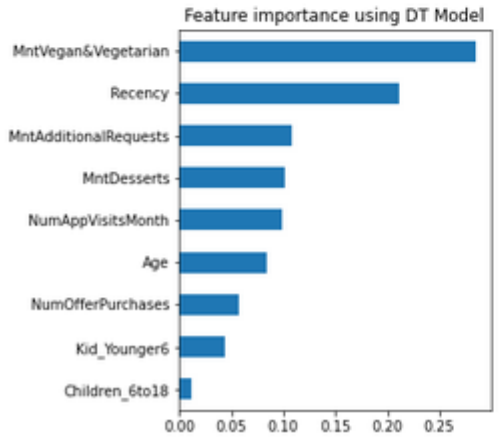
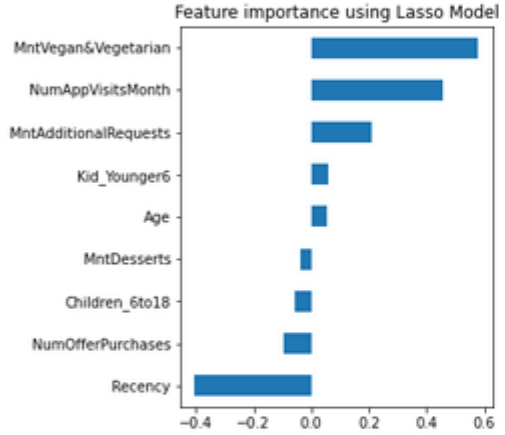


Figure 37 - Split 4

```
SPLIT 5
```

```
---------- RFE ----------
Kid_Younger6             False
Children_6to18            True
Recency                   True
MntVegan&Vegetarian       True
MntDesserts              False
MntAdditionalRequests     True
NumOfferPurchases        False
NumAppVisitsMonth         True
Age                      False
dtype: bool
```

```
---------- LASSO ----------
```




```
---------- DT ----------
```

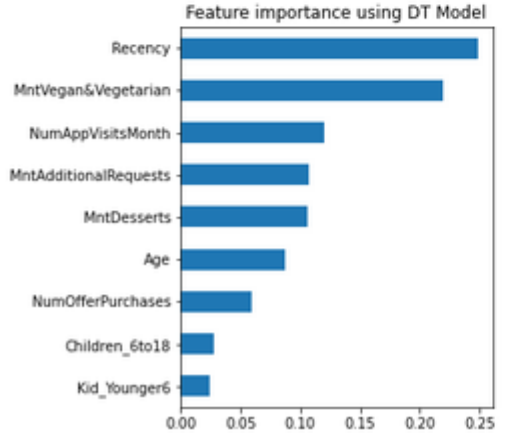


Figure 38 - Split 5

Appendix XIII - RFE, Lasso, and DT final results

Feature	RFE	LASSO	DT	Total	
Kid_Younger6	0	1	0	1	
Children_6to18	0	4	0	4	
Recency	5	5	5	15	Keep
MntVegan&Vegetarian	5	5	5	15	Keep
MntDesserts	0	0	2	3	
MntAdditionalRequests	0	5	5	10	Keep
NumOfferPurchases	0	0	1	1	
NumAppVisitsMonth	5	5	3	13	Keep
Age	0	0	4	4	

Figure 39 - The best features for numerical variables

Appendix XIV - Models Score without optimization.

	Train	Validation	Delta Train - Validation
Logistic Regression	0.58+/-0.02	0.539+/-0.05	0.041
KNN	0.519+/-0.02	0.308+/-0.08	0.211
DT	0.499+/-0.05	0.438+/-0.02	0.061
NN	0.99+/-0.0	0.777+/-0.02	0.213
SVM	0.663+/-0.03	0.483+/-0.05	0.180
XGB	1.0+/-0.0	0.69+/-0.06	0.310

Figure 40 – Model score for ‘historicalDF2’

	Train	Validation	Delta Train - Validation
Logistic Regression	0.58+/-0.02	0.539+/-0.05	0.041
KNN	0.519+/-0.02	0.308+/-0.08	0.211
DT	0.499+/-0.05	0.438+/-0.02	0.061
NN	0.99+/-0.0	0.777+/-0.02	0.213
SVM	0.663+/-0.03	0.483+/-0.05	0.180
XGB	1.0+/-0.0	0.69+/-0.06	0.310

Figure 38 – Model score for ‘X’

Appendix XV - Neural Network point plot

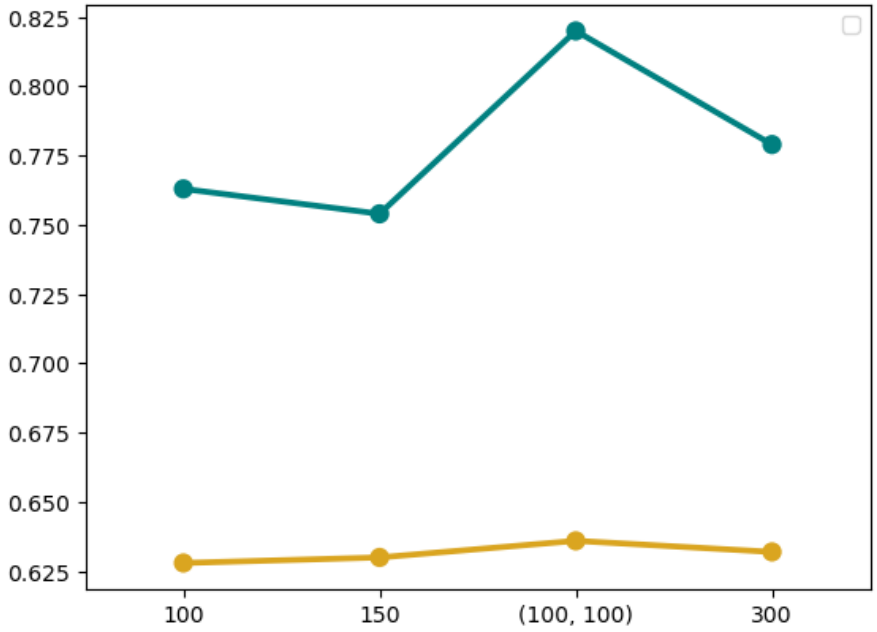


Figure 41 - Figure 41 - Neural Network

Appendix XVI - Algorithms

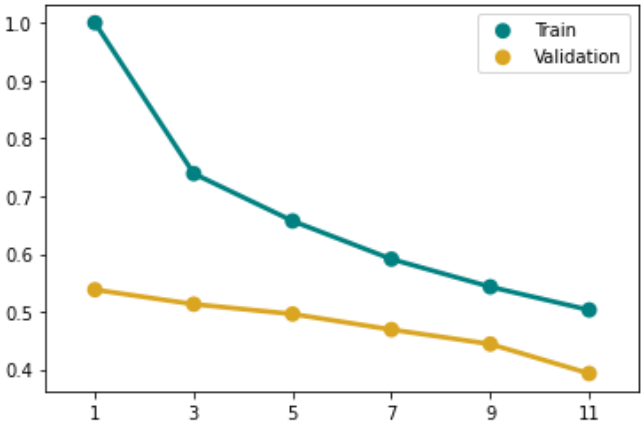


Figure 42- KNN (the best value of number of neighbors is 5)

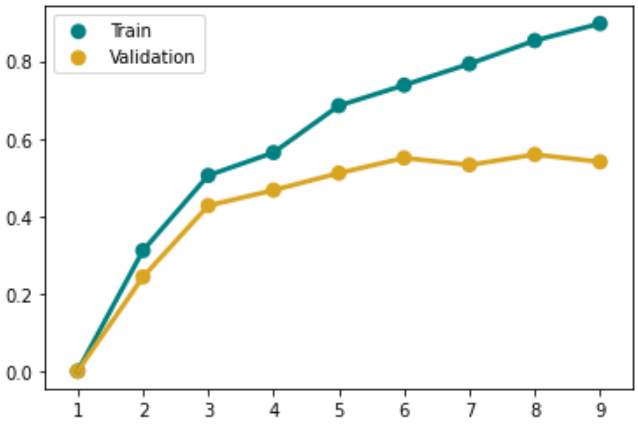


Figure 43 - Decision Trees (the best value of levels to keep is 5)

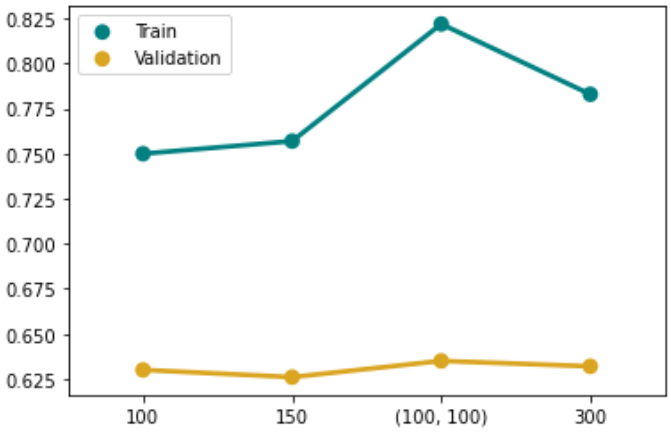


Figure 39- Neural Networks (the best value is 100)

Appendix XVII - ROC Curves: KNN, DT, NN, SVM and XGB, and individual Neural Network

The ROC Curve provides a graphical representation of the trade-off between false positives (x-axis) and true positives (y-axis). It offers a comprehensive visualization of the performance of the five models by plotting multiple confusion matrices. Each confusion matrix corresponds to a different threshold applied to the predicted probabilities, ranging from 0 to 1.

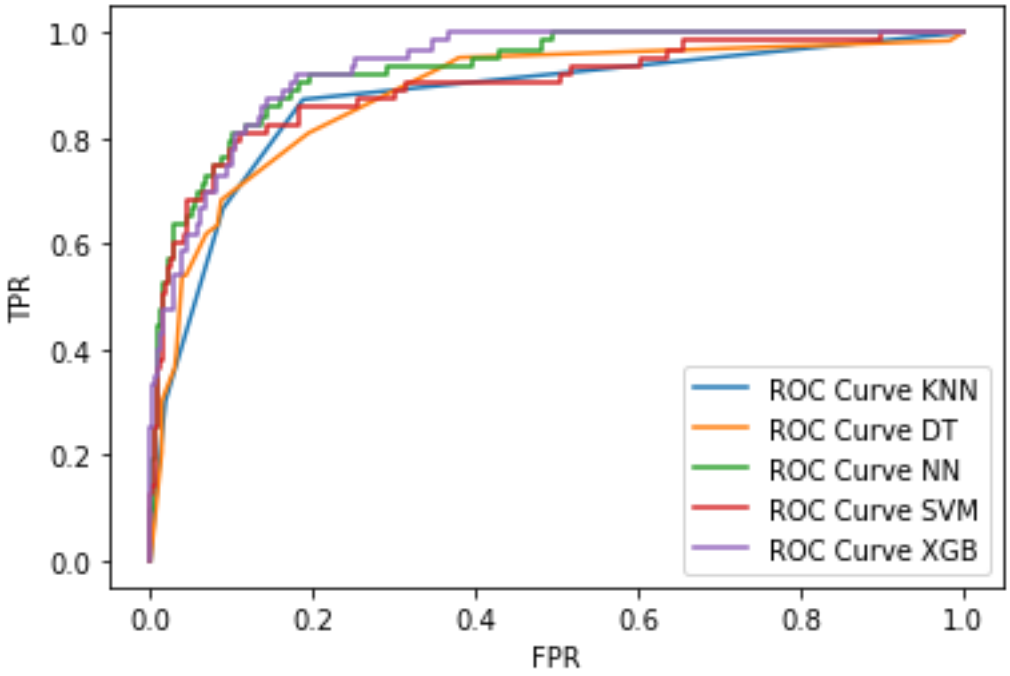


Figure 40- ROC Curve KNN, DT, NN, SVM and XGB

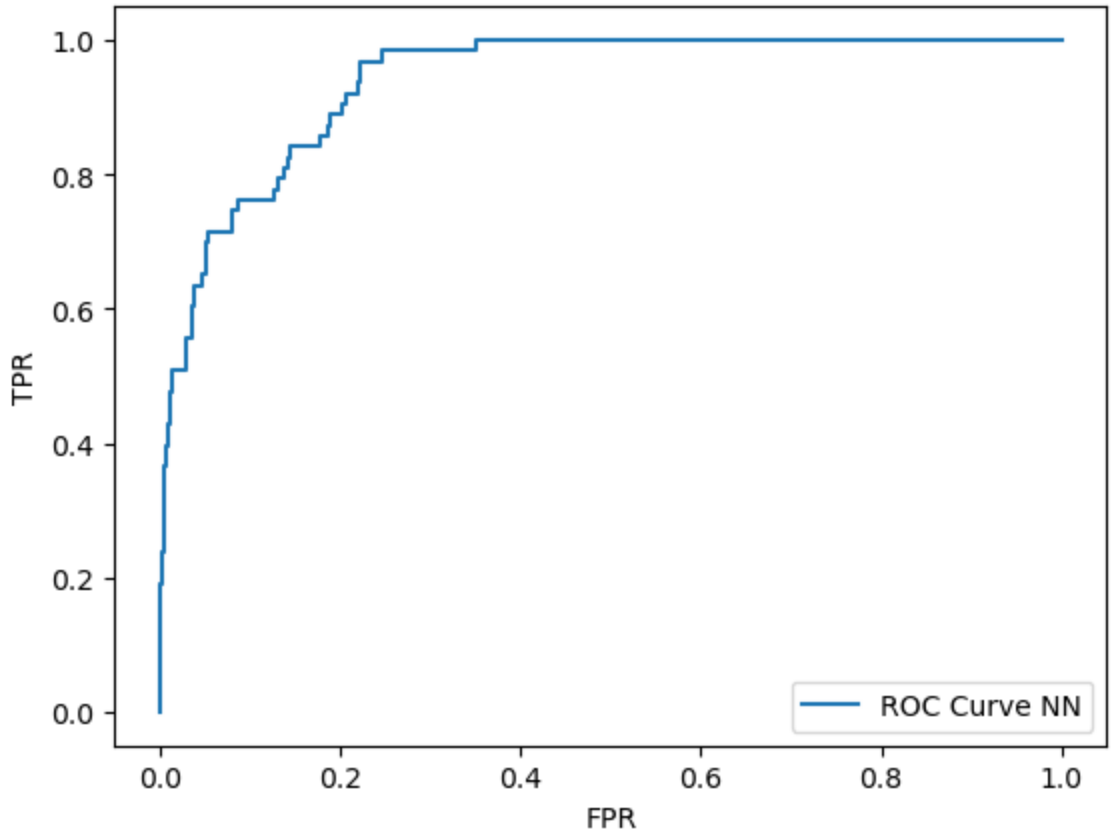


Figure 44 - ROC Curve Neural Network

Appendix XVIII - Neural Network best threshold

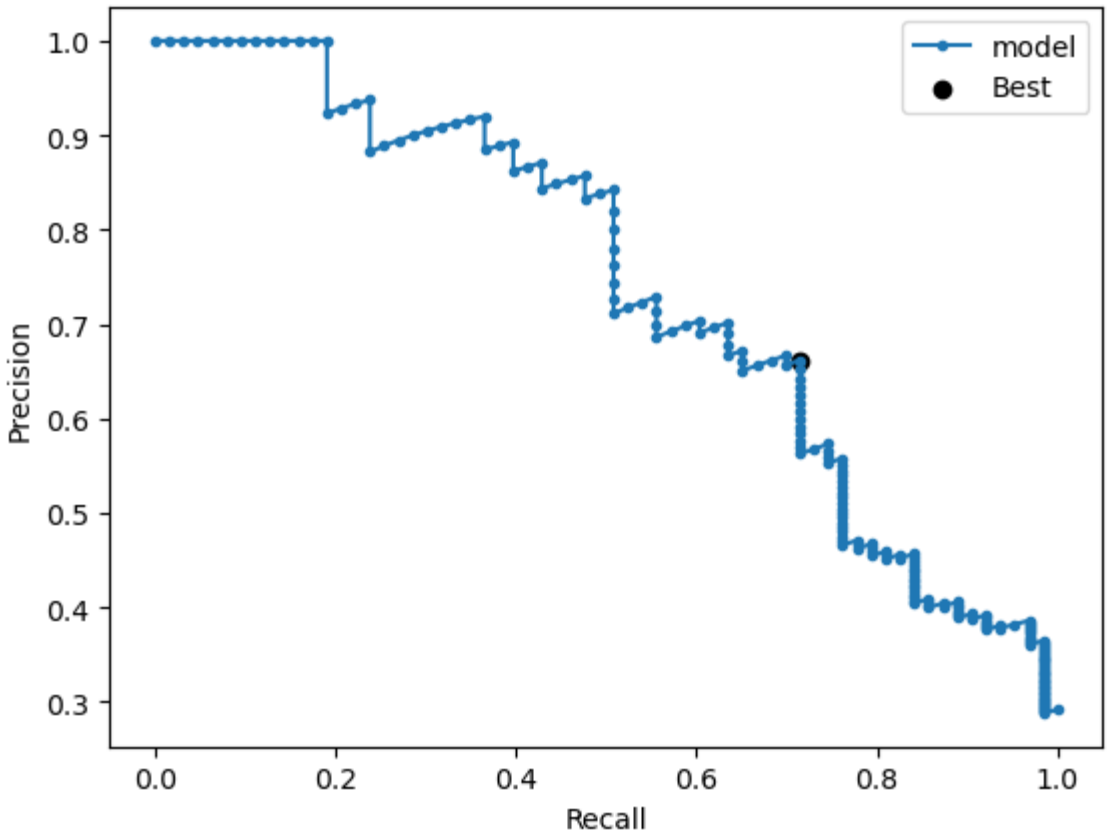


Figure 45 - Best threshold