TECHSCAPE E-COMMERCE

CUSTOMER´S BUYING BEHAVIOUR PREDICTIONS

PREPARED BY

Group 52

MACHINE LEARNING

26.Dec.2021

Table of Contents

Abstract

The following study has been conducted to develop a machine learning predictive model for Techscape, a Portuguese startup that specialises in digital detox products sold through their online shop. The goal is to analyse the online behaviour of their customers and to predict which customers are more likely of buying their products depending on their online behaviour.

Envisioning to obtain the highest possible score on our test dataset, the project started with a detailed data exploration and pre-processing, including checking for missing values and outliers, coherence checks, feature engineering and feature selection. Afterwards, a range of supervised learning algorithms were implemented such as, logistic regression, decision trees, neural networks, support vector machines and ensemble techniques. The scores of each of these models were tested through validation datasets and the results across models were compared. To improve the performance of the models, we have tried different combinations by adjusting parameters, scaling the data, using different techniques for imbalanced datasets and outlier detection methods. Finally, the best model performance was obtained through Random Forest with a train score of 0.835, a validation score of 0.712 and a test score of 0.7159.

Keywords: Machine Learning, Supervised Learning, Predictive Modelling, Binary Classification

1. Business understanding and Project plan

TechScape is a Portuguese online store startup company founded in 2020 that sells goods related to digital detox. Their products, such as, meditation kits, books, stress balls and retreats, allow their customers to stay focused on the most important things and improve the balance with technology use in their lives. such as meditation kits, books, stress balls and retreats. Aiming to increase their sales, TechScape wants to analyse the online behaviour of their customers and to predict which customers have a high probability of buying their products depending on their online actions.

Our goal is to build a predictive model that answers the question “Which customers are more likely to buy TechScape’s products?” using company’s data that contains general information concerning customers and their behaviour on the website from February 2020 until December 2020 (excluding April). We have a binary categorical output feature to predict which indicates a classification predictive modelling problem.

2. Data exploration and pre-processing

2.1 Initial data collection

Our team was provided with two data sets “train” and “test”. The “train” data set was used to train and validate our models so that we can then predict which customers contained in the “test” dataset are more likely to buy TechScape’s products.

2.2 Data exploration and quality verification

A description of the features that compose the datasets can be found in (A1). Our initial data exploration was done separately for both sets of data. The following main points can be taken from our data exploration:

- The "train" dataset contains observations concerning 9999 users whilst the "test" 2300.
- It is possible to group the features in two categories according to their data types - metric features and non-metric features:

Metric features	"AccountMng_Pages", "AccountMng_Duration", "FAQ_Pages", "FAQ_Duration", "Product_Pages", "Product_Duration", "GoogleAnalytics_BounceRate", "GoogleAnalytics_ExitRate" and "GoogleAnalytics_PageValue";
Non-metric features	"OS", "Country", "Date", "Type_of_Visitor", "Browser", "Type_of_Traffic"

- Considering the non-metric features our analysis reveals that:

  - Techspace users are based across two continents and nine countries. Most users are from Portugal and Spain and the proportion of buyers/non-buyers seems to be evenly distributed across countries (A2) which might be an indication that "Country" is not a good predictor.

  - The most popular operating system is "Windows", and there are clear differences in buyers' behaviour across operating systems(A3).

  - Their web platform is accessed by 3 types of visitors- returners, new users and other (eventually related with in house staff). Both returners and new users are potential buyers(A4).

  - Users access the website through a total of 15 different sources, buyers are distributed unevenly across the different sources(A5).

  - There are 13 different possibilities of browsers used. Two of them are more predominant, and only users from half of the browsers are buyers (A6).
- There are no missing values;
- There are no duplicated values;
- Nothing suggests the existence of inconsistent values;
- Some data types should be altered – "Date" should be converted to timestamp;
- The visual exploration of the data shows the possibility of outliers - we must further inspect and consider removing them.
- Visual inspection also suggests a high correlation level between a number of variables - this will be addressed during feature selection (A10).
- Considering the distribution of the target variable, we have an imbalanced dataset on a ratio of 8447:1552 with a clear predominance of non-buy behaviour.

2.3 Fix wrong data types

The variable "Date" has been converted and replaced by a new variable "Timestamp" to be included in the metric-features group.

2.4 Outliers

Considering the metric features only, outliers were detected through four different methods:

Manual Visual Inspection	Through the inspection of Histograms and Box-plots, values above a certain threshold have been dropped (A11). **Percentage of data kept after removing outliers**: 0.9757.
IQR	This method considered as outliers all values located 1.5 times the interquartile difference under or above the first and the third quartiles. **Percentage of data kept after removing outliers:** 0.4742.
Z-Score	Assuming that any z-score greater than 3 or less than -3 is considered to be an outlier. **Percentage of data kept after removing outliers:** 0.9652
DBSCAN	DBSCAN has detected a cluster with a total of 278 outliers (using a eps =300 and min_samples =18). **Percentage of data kept after removing outliers:** 0.9721

We opted for not using IQR in our models since the percentage of data removed as outliers was too high (over 50%). Outlier removal will exclusively be applied to the "train" dataset as we do not wish to remove any instances from the "test" data set.

2.5 Data normalisation

Our data features show varying degrees of magnitude, range, and units, which is a significant obstacle as some machine learning algorithms are highly sensitive to it. To avoid that a single variable steers the model performance in a certain direction, we gave equal weights/importance to each variable prior to model fitting. To do so, we used MinMax scaler to normalize our metric features in both data sets.

2.6 Data standardization

In the same way, to avoid bias by variables that are measured at different scales during model fitting, we tried to standardize our data by removing the mean and scaling to unit variance ($\mu=0$, $\sigma=1$). We also tried Robust Scaling as outliers can often influence the sample mean / variance in a negative way. In such cases, the median and the interquartile range often give better results.

2.7 Re-sampling techniques for imbalanced classification

Considering that we have an imbalanced dataset, which could negatively impact the results of our model, we tried four different techniques for imbalanced classification problems namely random undersampling, random oversampling, undersampling & oversampling (combined) and SMOTE - Synthetic Minority Oversampling Technique. We will try these different techniques in our models and check which one gives us the best results.

2.8 One-hot encoding

As many machine learning algorithms cannot operate on label data directly and require all variables to be numeric, we used one-hot encoder to convert categorical data to integer data so that categorical variables can be included in our models. This process has generated a total of 46 new features.

2.9 Feature engineering

Besides all the new features created through one-hot encoding we decided to engineer one more feature called "Season" in which we group all dates into seasons with the goal to verify if different season would have a direct effect on our target.

3. Feature selection

For feature selection, we employed a range of filter, wrapper and embedded methods to help us select the features that could be better contributors to our model. We separated our analysis into metric, non-metric and metric with encoded features (one-hot encoder) and we applied different methods to each segment.

Metric features: We started by applying Pearson and Spearman Correlations to access the correlation between features and their individual correlation with the target. We concluded that there is no independent variable highly correlated with the target. There are three pairs of variables highly correlated with each other, namely, FAQ_Duration vs FAQ_Pages (0.95), AccountMang_Pages vs AccountMang_Duration (0.94), GoogleAnalytics_BounceRate vs GoogleAnalytics_BounceRate (0.91) and Product_Pages vs Product_Duration (0.88). We need to drop some of these features. However, we will apply other feature selection techniques before deciding which variables we should keep. Additionally, we used Recursive Feature Elimination (RFE) to help us select the 4 more important features using logistic regression as our base algorithm. We also used Lasso and Ridge Regression, MAE, MSE and FRIEDMAN Importances and AdaBoost´s feature importance that is determined by the average feature importance provided by its base classifier, in this case we used a Decision Tree as a base classifier.

Non-metric features: The Chi-square test was used to test the independence between the categorical variables and our target variable. A higher Chi-Square value indicates that the feature is more dependent on the response, and that it can be selected for model training. We also considered the information we obtained through the histograms we built during the data exploratory phase, that related each categorical feature with our target variable “Buy”.

Metric and encoded features: For this analysis, we used all metric features, including the non-metric features encoded with one-hot encoder. The methods applied included ANOVA F-Values, Gini Importance and Entropy Importances, MAE, MSE and FRIEDMAN Importances, AdaBoost´s feature importance and Chi-square

The analysis of the different feature selection methods allowed us to see which predictors we should possibly include (YES) in the modelling phase, offering us the following final insights:

Numeric variables										
Predictor/ Model	AccountMng_Pages	AccountMng_Duration	FAQ_Pages	FAQ_Duration	Product_Pages	Product_ Duration	GoogleAnalytics_BounceRate	GoogleAnalytics_ExitRate	GoogleAnalytics_PageValue	Timestump
Pearson/ Spearman										
RFE						YES	YES	YES	YES	
Lasso			YES		YES	YES	YES	YES	YES	
Ridge			YES		YES	YES	YES		YES	YES
MAE,MSE, FRIEDMAN						YES	YES	YES	YES	YES
AdaBoost		YES				YES		YES	YES	YES
Total yes	0	1	2	0	2	5	4	5	5	3

Non-Metric variables					
Predictor/ Model	OS	Browser	Country	Type_of_Traffic	Type_of_Visitor
Chi-Square	YES	YES		YES	YES

Numeric variables including Encoded Variables

Model /Predictor	ANOVA F-Values	Feature Importances - TBC	Gini Importance | Entropy	MAE, MSE, FRIEDMAN	AdaBoost	Chi-square	Total yes
AccountMng_Pages	YES	YES	YES	YES			4
AccountMng_Duration	YES	YES	YES	YES			4
FAQ_Pages	YES			YES			2
FAQ_Duration				YES			1
Product_Pages	YES	YES	YES	YES	YES		5
Product_Duration	YES	YES	YES	YES	YES		5
GoogleAnalytics_BounceRate	YES	YES	YES				3
GoogleAnalytics_ExitRate	YES	YES	YES	YES	YES		5
GoogleAnalytics_PageValu	YES	YES	YES	YES	YES		5
x0_MacOSX						YES	1
x0_Windows						YES	1
x2_spring					YES	YES	2
x2_winter					YES	YES	2
x3_Returner	YES				YES	YES	3
x5_13						YES	1
x5_15						YES	1
x5_2	YES				YES	YES	3
x5_3						YES	1
x5_8					YES	YES	2

The previous analysis and the high correlation verified through Person's and Spearman´s, allowed us to be in a more comfortable position to decide which variables will positively contribute to our prediction. Hence, considering the variables that are highly correlated and their level of importance to predict our target, we decided to keep "FAQ_Pages", "AccountMang_Duration", "GoogleAnalytics_ExitRate", "Product_Duration" and to drop the remaining highly correlated ones. Additionally, given their repeated importance across the feature methods "GoogleAnalytics_PageValue" will also be included in our modelling. Moreover, four categorical values have shown to be good predictors for our model. Therefore, "OS", "Type_of_Visitor", "Browser", "Type_of_Traffic" will also be considered for our modelling. Finally, two enconded "x3_Returner" and "x5_2" are also to be considered to our models. Consequently, we have decided for two groups of features to model:

Feature Selection 1	"AccountMng_Duration" "FAQ_Duration"**,** "Product_Duration" "GoogleAnalytics_ExitRate"**,** "GoogleAnalytics_PageValue"**,** "Os", "Type_of_Visitor"**,** "Browser"**,** "Type_of_Traffic"
Feature Selection 2	"AccountMng_Duration" "FAQ_Pages", "Product_Duration" "GoogleAnalytics_ExitRate", "GoogleAnalytics_PageValue", "Os", "Type_of_Visitor", "Browser", "Type_of_Traffic", "Seasons"
Feature Selection 3	AccountMng_Duration", "FAQ_Pages", "Product_Duration", "GoogleAnalytics_ExitRate", "GoogleAnalytics_PageValue", "x3_Returner", "x5_2"

4. Modelling

Our first step towards modelling was to partition our previously chosen datasets into train and validation sets. The train set will be used to train and make the model learn the hidden features/patterns in the data, whilst the validation part is used to evaluate our model performance during training and help us tune the model's hyperparameters and prevent overfitting. We split the data attributing 70% of the data to train and 30% to validation. We used K-Fold cross validation to split the training set into 10 folds. We have also made sure to select the option stratified based on our target variable to ensure that the proportion of each class was preserved, since we are in the presence of imbalanced classes.

We selected a total of 14 models to train and validate our data: Logistic regression, Decision trees, Neural Networks, AdaBoost, Bagging classifier KNN, Bagging classifier Decision trees, Random Forest, Gradient Boost, Stacking Classifier, Voting classifier, Support Vector Machine (SVM), K-Nearest Neighbours (KNN) classifier, and Passive aggressive classifier (A9). For each group of selected features, we have also tried to model them with and without outliers and using different resampling techniques for imbalanced classifications.

5. Results

For our results analysis, we used the F1 score to assess the weighted average of Precision and Recall. On the tables below, we compare the train and validation scores of all models for each set of features selected and the variations we experimented regarding outliers and imbalanced datasets techniques.

Feature Selection 1 with MinMax Scaler

	With Outliers		Without outliers (z-score)		With Outliers and Undersampling		With Outliers and Oversampling	
	Train Score	Validation score	Train Score	Validation score	Train Score	Validation score	Train Score	Validation score
Logistic regression	0.517	0.512	0.499	0.493	0.640	0.633	0.64	0.633
Decision trees	0.669	0.602	0.657	0.598	0.660	0.659	0.662	0.661
Neural networks	0.639	0.619	0.637	0.623	0.639	0.634	0.647	0.641
AdaBoost	0.619	0.608	0.626	0.611	0.655	0.656	0.658	0.657
Bagging	0.744	0.615	0.748	0.618	0.662	0.661	0.713	0.662
Random Forest	0.994	0.599	0.998	0.612	0.664	0.659	0.788	0.668
Gradient Boost	0.735	0.626	0.689	0.627	0.661	0.658	0.665	0.661
KNN classifier	0.773	0.614	0.687	0.616	0.664	0.593	0.652	0.599
Passive aggressive classifier	0.679	0.578	0.662	0.589	0.681	0.595	0.647	0.586

For our first modelling attempt, we used Feature selection 1 using MinMax scaling. The results of our attempts were not satisfying enough. First, we tried to model without removing any outliers. Afterwards, we tried to remove the outliers (using the z-score method). However, the results did not improve. At this point, we considered that maybe the results were not the best for the fact that our data set was imbalanced. Hence, we decided to apply a combination of oversampling and undersampling techniques to try to balance it and assess if the results of the predictive models would improve. We also tried to use the SMOTE technique for imbalanced datasets, but the results were similar to the oversampling ones. Despite our attempts, the results did not improve as low scores and overfitting are noticeable across the results. The best results we got are the ones marked in green. Bagging stood out in three of the combinations we tried (with outliers, z-score and oversampling). Random Forest had the best result using the oversampling method, and gradient boost with outliers also showed one of the best results. However, even on these “better results”, we see signs of overfitting.

As this line of modelling was not yielding good results, we decided to test Feature Selection 2, where we only replaced “FAQ_Duration” for “FAQ_Pages”, we didn’t remove any outliers. We also used a different scaling method - Robust Scaler – as this is a more appropriated method to reduce outlier sensitivity.

To obtain these results we experimented with a variety of hyperparameters for each model, until we obtained the highest result we could achieve. However, overfitting is still quite noticeable throughout the results.

Our best result was obtained using Random Forest with a train F1 score of 0.835 and a validation score of 0.712. Keeping the outliers, using a scaler method that is more resilient to outliers and balancing our data using a combination of under and oversampling gave the best results so far. To obtain this result we initialized a set of parameters so that we could create several trees based on different criteria. Afterwards, we used RandomizedSearchCV to look for the hyper parameters that better fitted our x and y train. This allowed us to find the best estimator, which was the one we used to build our random forest model.

Finally, we thought it would be interesting to experiment with a new set of selected features that includes specific features created through one hot-encoder- Feature Selection 3. We used this new feature selection to train and validate our models. We maintained the outliers and combined under and oversampling techniques. We kept Robust Scaler and the Manual inspection for the outliers, as this seemed to have shown the best result previously. The results are as follows:

Feature Selection 2 with Robust Scaler		
With Outliers and Combined Oversampling and Undersampling		
	Train Score	Validation Score
Logistic regression	0.506	0.485
Decision trees	0.670	0.643
Naive bayes	0.546	0.527
AdaBoost	0.646	0.610
Bagging classifier KNN	0.690	0.629
Bagging classifier trees	0.999	0.521
Stacking	0.736	0.629
Voting classifier	0.666	0.555
Random Forest	0.835	0.712
Gradient Boost	0.827	0.630
SVM	0.795	0.592
KNN classifier	0.698	0.562
Passive aggressive classifier	0.196	0.165

Feature Selection 3 with Robust Scaler		
With Outliers and Combined Oversampling and Undersampling		
	Train Score	Validation Score
Logistic regression	0.503	0.467
Decision trees	0.670	0.643
Naive bayes	0.542	0.529
AdaBoost	0.646	0.610
Bagging classifier KNN	0.675	0.632
Bagging classifier trees	0.823	0.629
Stacking	0.835	0.624
Voting classifier	0.841	0.562
Random Forest	0.699	0.579
Gradient Boost	0.729	0.607
SVM	0.591	0.530
KNN classifier	0.168	0.147
Passive aggressive classifier	0.503	0.467

The results for this set of features did not show any significant improvement from the previous ones. The best result was obtained through the Bagging Classifier with KNN. Nevertheless, it is still not as optimal as the one we got previously with Random Forest.

6. Final Model Selection

The comparison of all models made clear to us that Random Forest applied to Feature Selection 2, maintaining the outliers and combining under and oversampling techniques was the model that gave us the best results. When submitting all models on Kaggle, we could also confirm that this model was indeed the one that scored higher. This model showed a train score of 0.835, a validation score of 0.712 and a test score of 0.7159. Therefore, this is our final selected model to help us predict TechScape's buyers' customer behaviour.

7. Conclusion

In this study we developed an end-to-end machine learning project aiming to answer Techscape's question of "Which buyers are more likely to buy our product?". We performed a series of steps including data exploration, pre-processing, feature selection and modelling, aiming to develop the best possible model to respond to Techscape's business needs. The data provided by the company, had no missing, duplicated, or incoherent values. However, we had to consider the engineering of some features, the presence of outliers, the need to scale the data, and also the fact that we were exposed to an imbalanced dataset. We used a variety of feature selection methods that permitted us to choose three sets of features, using a mix of metric, non-metric, and encoded features. For our modelling process we experimented with a total of 14 different models, and we used F1 score to test our train and validation. The test scores permitted us to adjust the models' hyperparameters, as well as the presence or not of outliers and the use of different imbalanced dataset and scaling techniques. In the end we concluded that Random Forest applied to Feature Selection 2, maintaining the outliers, and combining under and oversampling techniques was the model that gave us the best results to help us predict TechScape customer's buyer behaviour.

8. Bibliography

Alam, M. (2020). *Data normalization in machine learning*. towards data science. https://towardsdatascience.com/data-normalization-in-machine-learning-395fdec69d02#:~:text=The%20short%20answer%20is%20%E2%80%94%20it,because%20they%20are%20bigger%20numbers

Arduino, E. (2019). *Passive-Aggressive Classifier for Embedded Devices*. hackster.io. https://www.hackster.io/news/passive-aggressive-classifier-for-embedded-devices-f97c3461fbee

Bhandari, A. (2020). *Feature Scaling for Machine Learning: Understanding the Difference Between Normalization vs. Standardization*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/

Brownlee, J. (2017), *Why One-Hot Encode Data in Machine Learning?*. Machine Learning Mastery. https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/

Cross Validated forum (2018). *Feature Value Importance - AdaBoost Classifier*. Stack exchange. https://stats.stackexchange.com/questions/324383/feature-value-importance-adaboost-classifier

Gajawada, S. K. (2019). *Chi-Square Test for Feature Selection in Machine learning*. towards data science. https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223

Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research 3 (2003)* (1157-1182). https://elearning.novaims.unl.pt/pluginfile.php/103407/mod_resource/content/0/guyon03a.pdf

Kelleher, J. D., & Namee, B. M., & D’arcy, A. (2015). *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. The MIT Press.

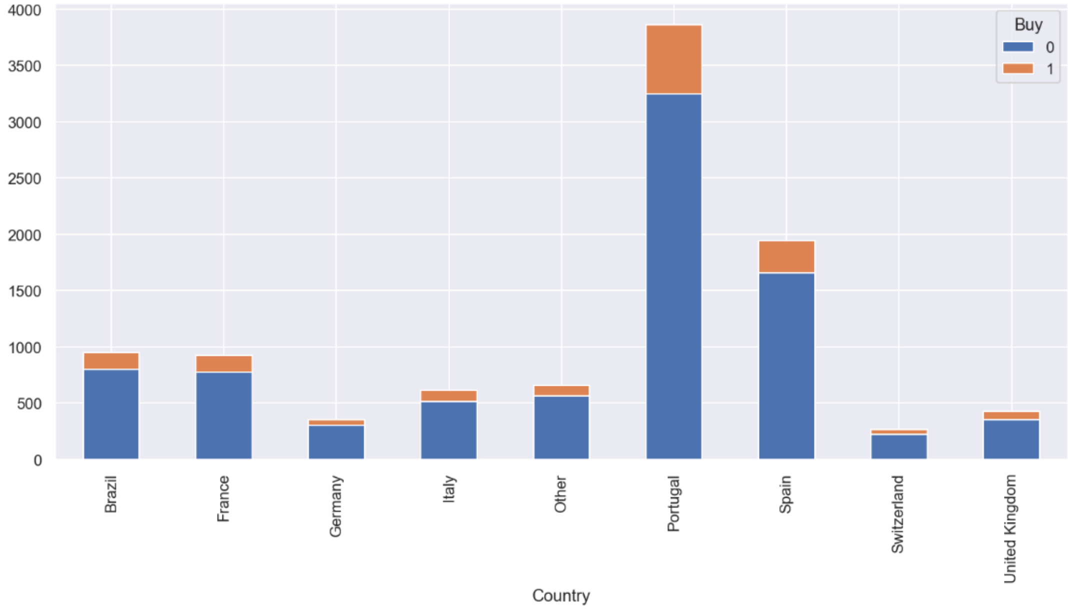
Loukas, S. (2020). *How and why to Standardize your data: A python tutorial*. towards data science. https://towardsdatascience.com/how-and-why-to-standardize-your-data-996926c2c832

Annexes

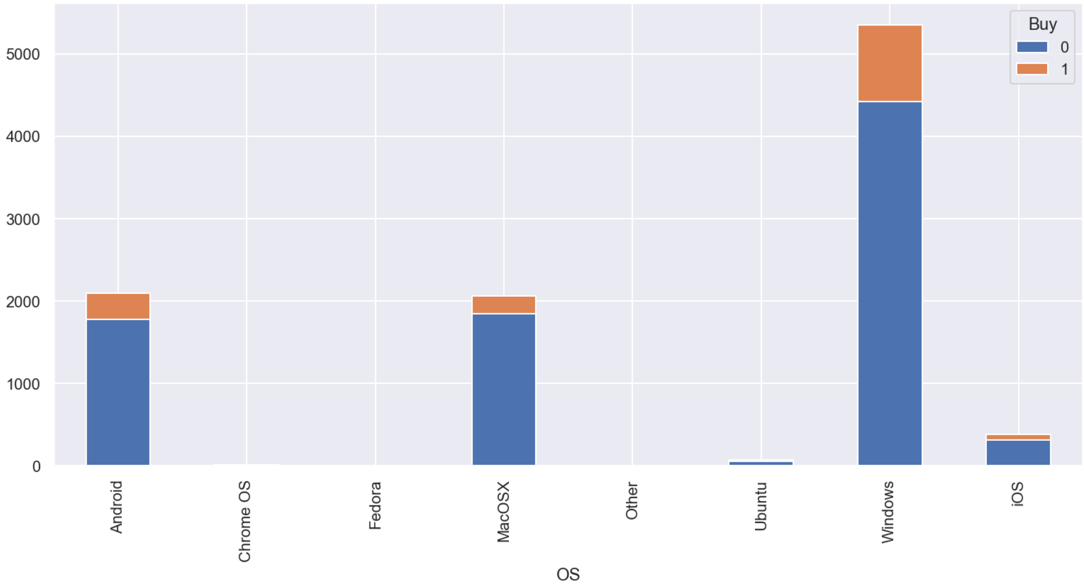
A1 – Data

Access_ID	**Unique identification of the user access to the website**
Date	Website visit date
AccountMng_Pages	Number of pages visited by the user about account management
AccountMng_Duration	Total amount of time (seconds) spent by the user on account management related pages
FAQ_Pages	Number of pages visited by the user about frequently asked questions, shipping information and company related pages
FAQ_Duration	Total amount of time (seconds) spent by the user on FAQ pages
Product_Pages	Number of pages visited by the user about products and services offered by the company
Product_Duration	Total amount in time (seconds) spent by the user on products and services related pages
GoogleAnalytics_BounceRate	Average bounce rate value of the pages visited by the user, provided by google analytics
GoogleAnalytics_ExitRate	Average exit rate value of the pages visited by the user, provided by google analytics
GoogleAnalytics_PageValue	Average page value of the pages visited by the user, provided by google analytics
OS	Operating System of the user
Browser	Browser used to access the webpage
Country	The country of the user
Type_of_Traffic	Traffic Source by which the user has accessed the website (e.g., email, banner, direct)
Type_of_Visitor	User type as "New access", "Returner" or "Other"
Buy	Class label indicating if the user finalized their actions in the website with a transaction

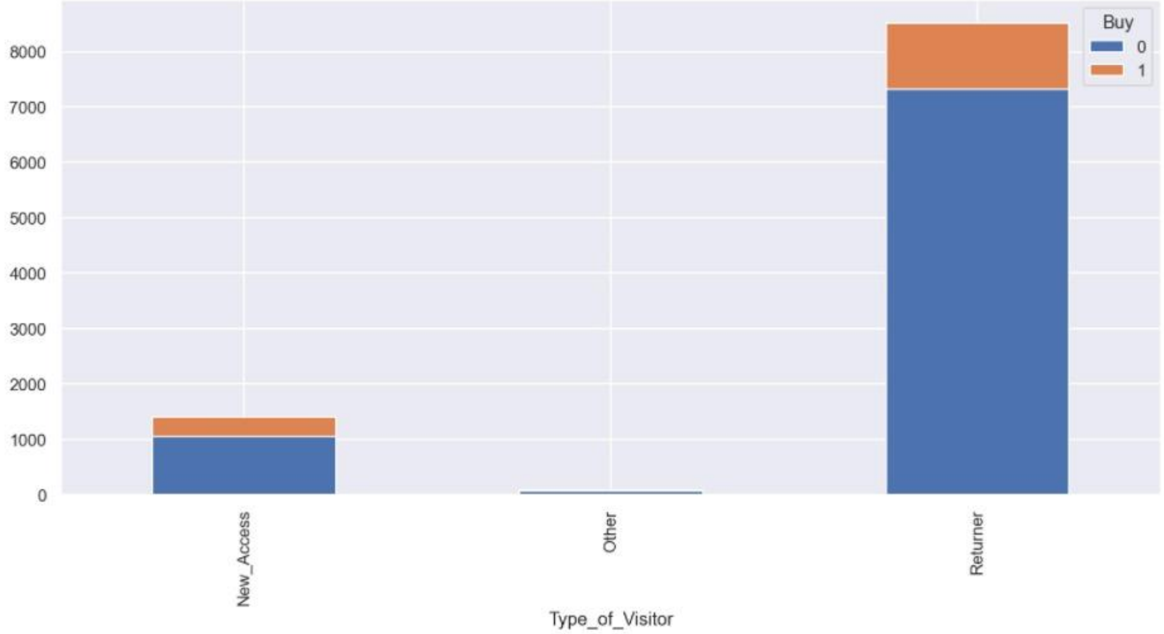
A2 – Distribution of buying behaviour per country



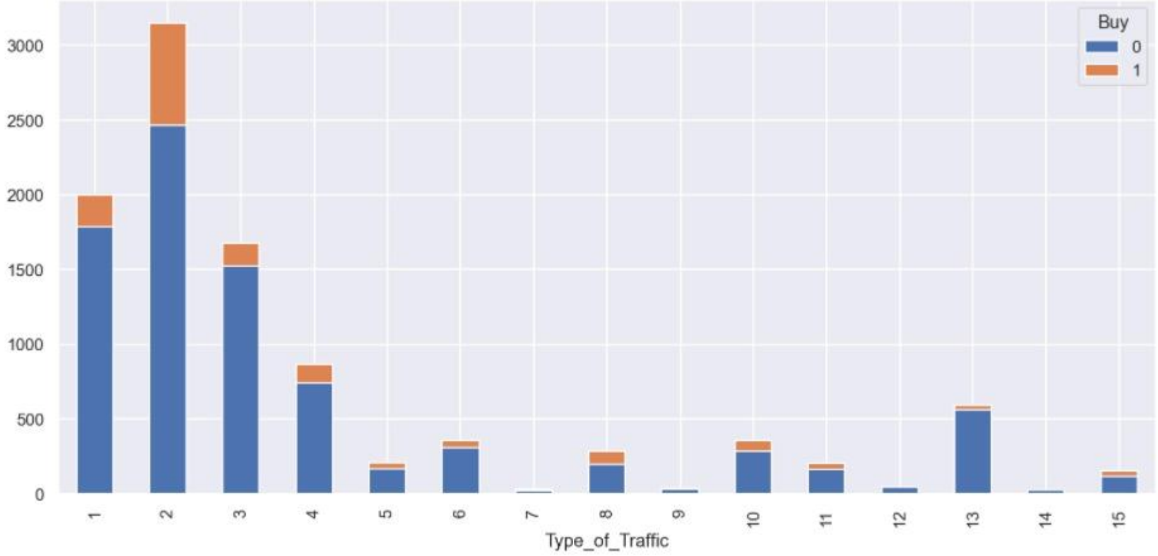
A3 - Distribution of buying behaviour per Operative System



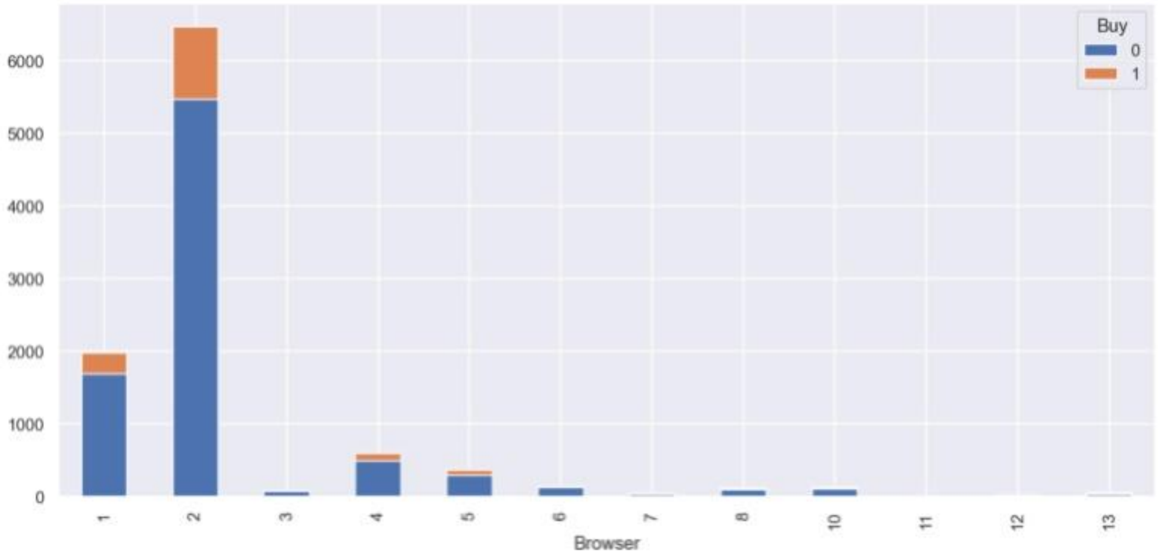
A4- Distribution of buying behaviour per Type of Visitor



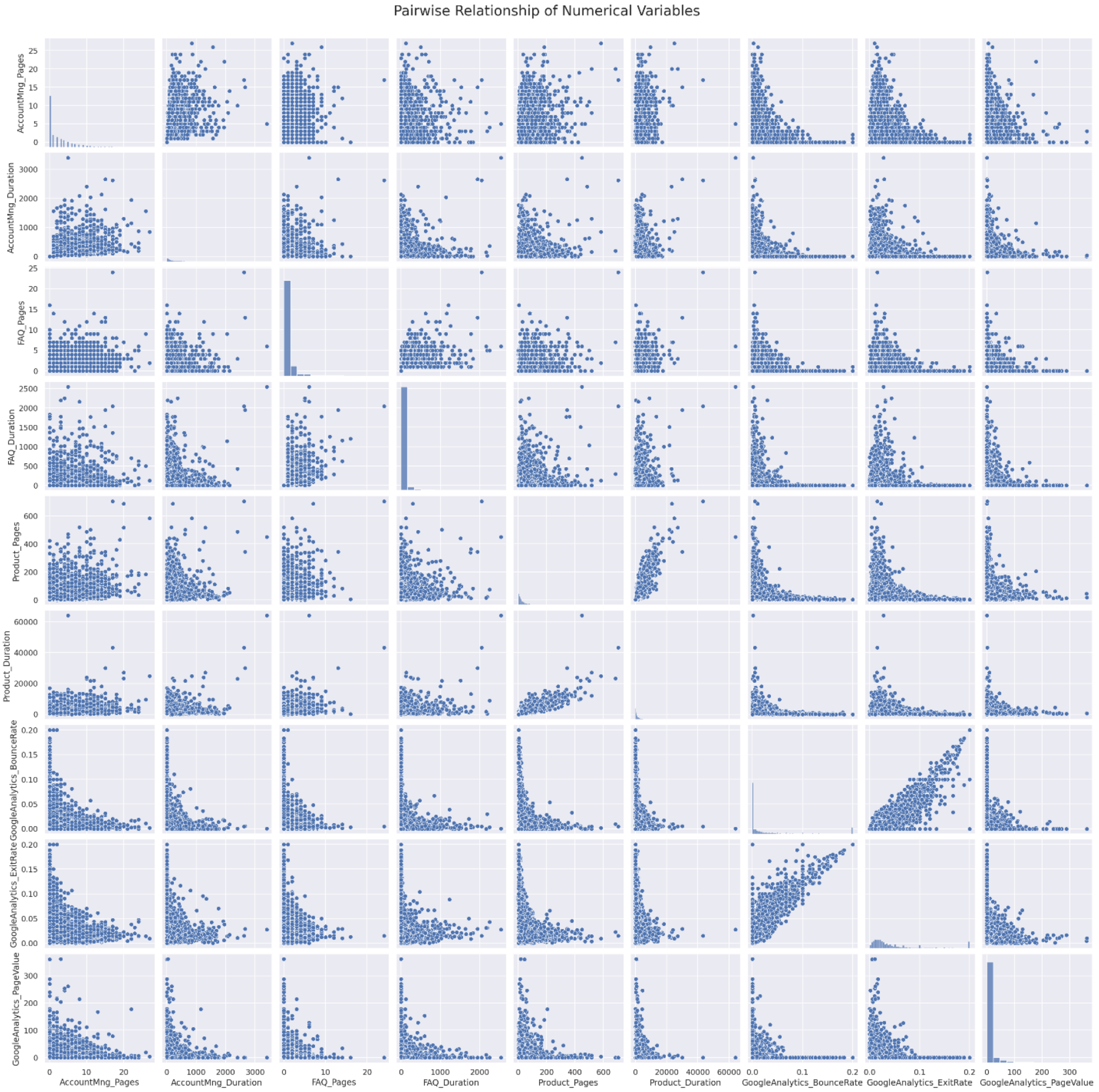
A5- Distribution of buying behaviour per Type of Traffic



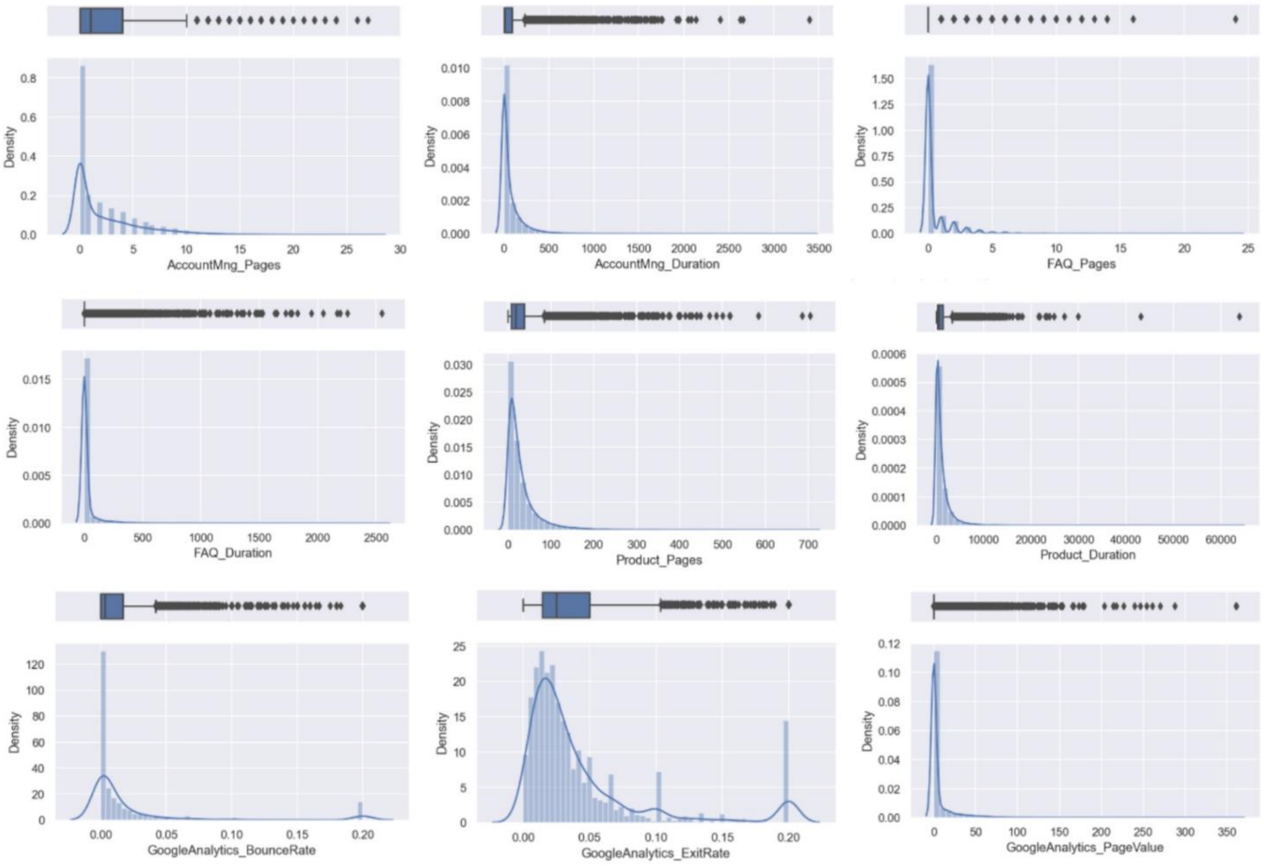
A6- Distribution of buying behaviour per Browser



A7 - Pairwise Relationship of Numerical Variables – features with high correlations



A8 - Histograms and Box-plots for the numerical variables



A9 - Passive aggressive classifier

Proposed by Crammer at al., the passive aggressive classifier is an ideal model to work with memory-constrained devices, as it works by incremental inspecting one training sample at a time, rather than all at once. It can be used for both classification and regression tasks. The main disadvantage is that we don't have the "big picture" of the data and the result can be affected by the order of presentation of the samples. We may also not be able to achieve the best accuracy. The core concept of this classifier is that it adjusts its weight vector for each misclassified training sample it receives, trying to correct it.