Project 3: Study of online shoppers' purchasing intention

A. Mattiroli[1], A. Pera[1], and J. Štorek[2]

[1]University of Milano Bicocca
[2]Czech Technical University in Prague

November 2019

Abstract

In this report, an analysis of online shoppers' purchasing intentions has been performed, identifying a model which could predict whether a user completed or not the purchase while visiting an e-commerce website. In order to determine the model which fits the data best, supervised learning algorithms such as logistic regression, random forest, support vector machines and extreme gradient boosting (XGBoost) have been applied. Out of all the methods considered, XGBoost and random forest are found to be the ones achieving better results in terms of performance, measured through indices such as accuracy, area under the curve (AUC) and confusion matrices. In particular XGBoost produces the best predictions, resulting in an accuracy of 0.9019 and AUC equal to 78.35 %.

1 Introduction

The analysis of web pages visitors' behaviour plays a crucial role for an e-commerce. Extraction of relevant information serves as a feedback for the website itself, in order to know how well it performs and which web pages are directly connected to customers' purchases. The e-commerce can then focus on improving these pages aiming to make customers' experience even more positive.

Online shoppers' purchasing intention data set has been studied by applying logistic regression, random forests, support vector machines and XGBoost algorithms. All methods considered are explained in the methods section. After providing a detailed data set description, implementation of algorithms is summarized in the implementation section. Finally, results and conclusions are discussed in the results section which is followed by a short conclusion.

2 Methods

2.1 Logistic regression

Logistic regression is a widely-used classifier which allows to predict qualitative responses. In particular, logistic regression models the probability of Y to fall into a particular category rather than modelling directly the response itself.

The probability that a data point belongs to a category is given by the Sigmoid or logit function [1]

$$p(t) = \frac{1}{1+\exp(-t)} = \frac{\exp t}{1+\exp t} \quad , \tag{1}$$

which represents the likelihood of an event.

Logistic regression models are fit by applying the Maximum Likelihood Estimation (MLE) criterion. In a binary case with independent data points, the likelihood expression is

$$P(D|\vec{\beta}) = \prod_{i=1}^{n} [p(y_i = 1|x_i, \vec{\beta})]^{y_i} [1 - p(y_i = 1|x_i, \vec{\beta})]^{1-y_i} \tag{2}$$

from which it is possible to obtain the negative log-likelihood and the cost function, the so-called Cross Entropy, as

$$C(\vec{\beta}) = -\sum_{i=1}^{n} (y_i(\beta_0 + \beta_1 x_i) - \log(1 + \exp(\beta_0 + \beta_1 x_i)) \tag{3}$$

which is easier to minimize in order to find the parameter estimates.

The binary logistic regression has multiple-class and multinomial extensions in which the gradient descent method is typically used in order to find the optimal parameters' values.

2.2 Random forests

Random forest is a supervised learning ensemble algorithm which is based on the concept of decision trees. In particular, this kind of method aims to find the explanatory variables which are the most informative with respect to a specific response. The method then tries to split the dataset according to the values of these relevant features, providing another dataset for which the response variable is pure.

Decision trees are normally made of a root node, the interior nodes and final leaf nodes or leaves, which are connected by branches. In particular, leaf nodes contain predictions of new instances made after a training process of the data. The decision tree algorithm works by firstly considering a training dataset containing both explanatory variables and target features (response variables), used in order to train the model by applying a measure of information gain to determine which features can be considered informative. Secondly, the tree is grown until a stopping criterion is reached and therefore predictions for new instances can be made by re-running the tree until reaching the leaf nodes. Before actually considering the algorithm itself, it is needed to split the predictor space into K distinct and non-overlapping regions R_1, ..., R_K, each of which corresponding to a specific class $k = 0,1,...,K$. For every observation falling into the same region, the same prediction is made.

When building a classification tree, some useful criteria to split different nodes are taken into account. In general, it is possible to define a probability density function p_{mk} which represents the number of observations of class k in a region R_m which contains N_m observations. The likelihood is expressed in terms of proportion of observations of the given class in the specific region, which is [2]

$$p_{mk} = \frac{\sum_{x_i \in R_m} I(y_i = k)}{N_m} \tag{4}$$

The three main methods used for splitting the nodes are then the misclassification error, defined as

$$p_{mk} = \frac{\sum_{x_i \in R_m} I(y_i \neq k)}{N_m} = 1 - p_{mk} \tag{5}$$

the Gini index, which is

$$g = \sum_{k=1}^{K} p_{mk}(1 - p_{mk}) \tag{6}$$

and the information entropy, defined as

$$s = -\sum_{k=1}^{K} p_{mk} \log(p_{mk}) \quad . \tag{7}$$

The least two methods are the ones usually preferred in terms of sensitivity to node purity.

Random forest uses the decision trees method in the sense that it builds a certain number of trees based on bootstrapped training samples. In particular, every time a split is needed, random forest considers a random sample of m predictors from all the p as candidates for the split and it uses only one of those m to actually perform it. The algorithm itself is built by firstly determining the number of trees which are to be considered. Then, for each of them, a bootstrap training sample from the design matrix is drawn and used to build a tree by repeating the following steps until the maximum node size in reached:

- select m variables from the p predictors
- pick the best split among those either by using CART or ID3 algorithms and create a new node
- split the node into daughter nodes

At the end, the ensemble of trees can be considered to make predictions.

The difference between the applied CART and ID3 algorithms is that the latter considers information gain for classification.

Moreover, the CART algorithm uses a single feature k and a threshold t_k in order to split the data into two different subsets. Those quantities are defined by choosing the best combination of parameters which produces the purest subset by using either the Gini factor or the entropy, minimizing a cost function which considers the impurity and the number of instances in the left/right subsets. The process then continues iteratively among the subsets, until it reaches the maximum depth or cannot find a split that reduces impurity.

The ID3 algorithm, instead, learns the trees by building them top-down. Firstly, it evaluates each instance attribute in order to determine how well it is able to classify the training samples by itself. The feature selected after a statistical test is then used at the root node and descendent nodes are created by considering possible values of the attribute itself. The whole process is repeated and the algorithm never backtracks to reconsider earlier choices.

2.3 Support vector machines (SVM)

2.3.1 Maximal margin classifier

Support Vector Machines (SVM) are Machine Learning methods which can perform well both on regression and classification problems, being particularly effective on classification of small-medium size datasets.

The method relies on the definition of hyperplanes and of a margin which separates classes of variables.

In a p-dimensional space, a hyperplane is a flat affine subspace of a hyperplane of dimension p-1, where p is the number of features which classify the data points. In a simple, two-dimensional case, it is defined as [3]

$$x^T w + w_0 = 0 \tag{8}$$

And in a p-dimensional space

$$w_0 + w_1 x_1 + w_2 x_2 + ... + w_p x_p = 0 \tag{9}$$

Considering an output variable $y_i = \pm 1$ then, if $y_i = 1$

$$w_0 + w_1 x_1 + w_2 x_2 + ... + w_p x_p > 0 \tag{10}$$

holds, whereas

$$w_0 + w_1 x_1 + w_2 x_2 + ... + w_p x_p < 0 \tag{11}$$

holds if $y_i = -1$. In both cases,

$$(w_0 + w_1 x_1 + w_2 x_2 + ... + w_p x_p) y_i > 0 \tag{12}$$

In order to separate the classes there are many possible hyperplanes which can be taken into account and the main goal is to find the one that gives the maximum margin, that is the smallest of the perpendicular distances from each training observation to a given hyperplane. The maximal margin hyperplane is the separating hyperplane for which the margin is the largest. The points that have the smallest evaluated distances are called support vectors. Interestingly, the maximal margin hyperplane depends directly on the support vectors, but not on the other observations: a movement of any of the other observations would not affect the separating hyperplane. Hence, since the maximal margin hyperplane depends directly just on a small subset of the observations, it is often successful being quite robust to the behavior of observations that are far away from the hyperplane, although leading to possible overfitting when p is large.

The maximal margin hyperplane is the solution to the optimization problem $max(M)$ subject to $\sum_{i=1}^{p} w_i^2 = 1$ and $(w_o + w_1 x_{i1} + w_2 x_{i2} + ... + w_p x_{ip}) y_i \geq M$ $\forall i = 1, ..., n$.

M then satisfies

$$y_i(W^T x_i + w_o) \geq M \parallel W \parallel \quad . \tag{13}$$

By scaling the equation so that $\parallel W \parallel = 1/M$, the problem is then reduced to finding the minimum of $\parallel W \parallel$ subject to $y_i(W^T x_i + w_o) \geq 1$. The margin is defined as the inverse of the norm of W. Lagrangian multipliers λ_i are used to solve the problem, setting the function

$$L(\lambda, w_0, W) = \frac{1}{2} W^T W - \sum_{i=1}^{n} \lambda_i [y_i(W^T x_i + w_0) - 1] \tag{14}$$

If $\lambda_i > 0$ then $y_i(W^T x_i + w_0) = 1$ and x_i is considered on the boundary.

If $y_i(W^T x_i + w_0) > 1$, $\lambda_i = 0$ and x_i is not on the boundary.

The vectors for which $\lambda_i > 0$ are the so-called support vectors and they define the margin by computing the coefficients

$$W = \sum_i \lambda_i y_i x_i \tag{15}$$

$$w_0 = \frac{1}{y_i} - W^T x_i. \tag{16}$$

2.3.2 Soft classifier

The maximal margin classifier is a very natural way to perform classification, in the case a separating hyperplane does exist. Otherwise, it is hard to exactly separate the classes. A possibility is to develop a hyperplane that almost separates the cases using a so-called soft margin which allows some points to be classified on the wrong side of the margin. In this case, $y_i(W^T x_i + w_0) = 1 - \xi_i$ where $\xi_i \geq 0$ is called the slack variable and $\sum_i \xi_i$ is the total violation.

If $\xi_i > 0$, then the i-th observation is on the wrong side of the margin, and it can be said that the i-th observation has violated the margin, e.g. has been misclassified. If $\xi_i > 1$ then the observation is on the wrong side of the hyperplane. The Lagrangian function considered for maximal margin classifier now becomes

$$L(\lambda, w_0, W) = \frac{1}{2}W^T W - \sum_{i=1}^{n} \lambda_i [y_i(W^T x_i + w_0) - (1-\xi)] + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \xi_i \gamma_i \quad (17)$$

where C bounds the sum of ξ, determining the number and severity of the violations to the margin (and to the hyperplane) that will be tolerated, hence $\sum_{i=1}^{n} \xi_i \leq C$.

C controls the bias-variance trade-off of the technique itself. When C is small, there will be narrow margins that are rarely violated; this leads to a classifier that is highly fitted to the training data and may have low bias but high variance. When C is larger, the margin is wider and more violations are allowed; this produces a classifier that is potentially more biased but may have lower variance.

As in the maximal margin classifier case, the hyperplane depends only on a small subset of the observations, the ones that either lie on the margin or that violate the margin, which are the so-called support vectors. Therefore C also determines how many observation will be involved in determining the hyperplane.

2.3.3 Non-linear decision boundaries

The support vector classification method described until now is good for two-classes cases with close to linear boundaries. In general, when facing non-linear class classifications it is possible to make the procedure more flexible by enlarging the feature space using functions of the predictors, such as quadratic and cubic terms, in order to address the non-linearity.

Support vector machine, following this intuition, allows to enlarge the feature space in a way that leads to efficient computations, avoiding the risk of having unmanageable ones.

In order to do so, SVM uses the so-called Kernel approach. Note that a kernel is a function which quantifies the similarity of two observations. For example, a polynomial kernel is given by

$$K(x_i, x_j) = (1 + \sum_{k=1}^{p} x_{ik} x_{jk})^d \quad (18)$$

It essentially leads to fit a support vector classifier in a higher-dimensional space involving polynomials of degree d, rather than in the original feature space. When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a support vector machine.

Another popular choice is the Gaussian radial kernel (infinite-dimensional).

$$K(x_i, x_j) = \exp(-\gamma \parallel x_i - x_j \parallel^2) \quad (19)$$

Where the parameter $\gamma > 0$ affects the variability, since small values lead to high variance and low bias and vice-versa.

2.4 XGBoost

Extreme Gradient Boosting (XGBoost) is an advanced implementation of the gradient boosting algorithm. It is an ensemble learning method that combines the predictive power of multiple learners [4]. The result is a single model (strong learner) given as an aggregation of the outputs of several different models (weak learners). This machine learning process has had a wide impact by solving several classification and regression problems in an extremely efficient way. The main two reasons why XGBoost is widely used are the execution speed and the model performance.

The method uses decision trees and boosting, being defined as "a scalable end-to-end tree boosting system" [5]. In contrast to bagging techniques such as Random Forest, in which trees are grown to their maximum extent, boosting makes use of small trees.

In particular, the iterative gradient boosting algorithm on which XGBoost is based on starts by initialing the estimates (F_0) after having minimized a specific loss function. Then, for a number of trees m which goes from 1 to M, $\tilde{y}$ is computed as the finite difference approximation to the derivative of the cost function. The model parameters a_m are defined, for each tree, by minimizing the squared difference between $\tilde{y}$ and the gradient $h(x_i, a)$, penalized by a weighting term for each node (β_i). These parameters represent the way in which each decision node makes a split. Afterwards, still for tree m, the weighting term ρ_m for each new fit h is computed as the one which minimizes the cost function between the target y and the estimate F_m. F_m is then generated by combining the model from the previous iteration, F_{m-1}, and the weighted gradient considering the parameters a_m. At the end of the loop, after M iteration, the final estimate of the model is given by

$$F_M = F_0 + \sum_{m=1}^{M} \rho_m h_m(x) \tag{20}$$

The trees are built sequentially such that each tree aims to reduce the errors of the previous one.

For each node, there is a factor β_i which defines the different impact of each branch of the split. Multiple models are created, each correcting the errors of the previous ones. Initially, all data point are given equal weights, whereas in later models the observations which were incorrectly predicted are given higher weights. The weak learners have high bias and would not perform well on the entire data set, but work well for some part of it. The final model, which is the weighted mean of all models created, brings down both the bias and the variance.

The loss function implied in XGBoost adds a regularization term to the usual cost function (for example, MSE for regression problems) in order to control the final learnt weights and to reduce over-fitting. Other techniques which aim to avoid over-fitting are used in the process, such as shrinkage which scales down new weights after each step of tree boosting, reducing the influence of individual trees and providing the possibility for new ones to improve the model. The other technique implied is column sub-sampling, which randomly sub-samples features used by trees for fitting. This last method is also used in random forest, although XGBoost allows to use row sub-sampling in order to fit on sub-samples of training data per tree instead of using the whole data itself.

In order to find the best split, XGBoost uses an approximation of the exact greeding algorithm, which consists of trying every possible split for the nodes and identifying the best one. The machine learning method is also able to deal with sparse data, since a default direction in tree nodes is added in order to classify missing instances.

2.5 SHapley Additive exPlanations (SHAP)

SHAP is a useful validation technique, being consistent and accurate, for interpreting complex models, in particular tree-based ones, by measuring the impact of different variables and how each feature affects the final predictions. SHAP relies on Shapley values, which are a solution for evaluating the features' contributions for single predictions in any machine learning model. A Shapley value is computed as the average marginal contribution of a feature value across different aggregations of features. This means that, for each combination of variables, the response is predicted both with and without one of the features and the difference is then considered in order to get marginal contributions. Their average corresponds to the Shapley value of that specific feature

$$\phi_{ij} = \sum_{S \subset (x_1,..,x_p)} \frac{|S|!(p-|S|-1)!}{p!} (\hat{y}_{i,S \cup x_j} - \hat{y}_{i,S}) \tag{21}$$

where S is a subset of the features not containing x_j. By running SHAP for each instance, it is possible to obtain a matrix of Shapley values containing one row per data instance and one column per feature. The entire model can be interpreted by analyzing the matrix itself. The feature importance is then evaluated by averaging the absolute Shapley values per feature across the data as

$$I_j = \sum_{i=1}^{n} |\phi_{ij}| \tag{22}$$

The feature effect is represented as an additive linear model

$$g(z) = \phi_0 + \sum_{j=1}^{M} \phi_j z_j \tag{23}$$

where g is the model, z an ensemble of features, M the maximum aggregation size and ϕ_j the Shapley value for feature j.

2.6 Performance measurements

In the machine learning context, checking the performance of a classification model is an important task. In order to evaluate and compare the performance of different methods, several metrics can be used.

Firstly, considering the number of correctly guessed targets over the total amount of targets, the accuracy score metric is defined as [6]

$$A(t_i, y_i) = \frac{\sum_{i=1}^{n} I(t_i = y_i)}{n} \tag{24}$$

where n is the total number of targets and I is indicator function which returns 1 for $t_i = y_i$ and 0 otherwise . Due to their definition, accuracy score values lay in the interval [0,1], where 1 represents the accuracy score of the best possible classification method.

Another method which can be applied is called Area Under Curve (AUC), which computes the area under a Receiving Operating Characteristics (ROC) curve. ROC is a probability curve used to understand the extent to which the model is capable to distinguish between different classes. This curve plots two parameters: the true positive rate (TPR) and the false positive rate (FPR). The first one is the ratio of true positive to all positives, and the second is the ratio of false positives to all negatives (binary classification). AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0, while an AUC of 0.5 reflects random predictions.

Finally, confusion matrices can be produced in order to study the amount of correctly guessed targets after the use of a specific predictive algorithm. A confusion matrix is a table which allows to show the way the classification method is confused when making predictions and it is strictly connected to the concept of accuracy score. In particular, the the diagonals of the table contain the number of correctly predicted targets in all the classes studied and the number of wrong guesses.

2.7 Dataset

The dataset [7] contains information about the purchasing intention of online shoppers. Every line corresponds to one session of a single user. In particular, a session is defined as the time block in which the user actively accesses websites and this block normally does not exceed several hours. Data has been recorded throughout the all year (2018) in order to avoid unnecessary bias.

The dataset consists of 10 numerical and 8 categorical variables. All variables are described in Table 3 in Appendix, where they are also statistically described with results obtained after the data pre-processing.

As an additional explanation, ”bounce rate”, ”exit rate” and ”page value” variables are extracted from the Google analytics feature [9]. The ”Bounce rate” variable is an average value of all visited web pages' bounce rates by the users.

The bounce rate of a web page is the percentage of page visits when visitor quits the page without triggering any action.

The "Exit rate" variable is an average value of all visited web pages' exit rates by the users. Exit rate of a web page is the ratio of the particular page being the last in the session over all the page views.

The "Page value" is an average value of all the visited web pages' page values by the users. The page value is calculated based on whether visiting the specific web page precedes the purchase on the e-commerce or not. If the page was not involved in an e-commerce transaction in any way, then the page value is 0. The variable reaches high values when the page is frequently visited before purchases and its value is weighted by the amount of money spent in e-commerce. A detailed description of page value calculation can be seen in [9].

Finally, "special day" variable expresses the likelihood that the visitor is going to buy an item in relation to a special day, such as Valentine's day or Mother's day. For example, for Valentine's day, this variable takes a nonzero value between February 2 and February 12, zero before and after this date (unless it is close to another special day), and the maximum value of 1 on February 8 [7].

Explanation of all categories in categorical variables has not been provided.

3 Implementation

First, bad values have been removed from the analyzed dataset in the pre-processing stage, next methods such as logistic regression, random forests, support vector machines and XGBoost have been applied to fit the data. In order to be able to compare results from all these different methods, a random seed (state) has been chosen as random_state = 42, being an argument in all applied methods.

3.1 Exploratory data analysis and data pre-processing

Firstly, the data set is read by importing the csv file [7] as a Data frame. Some relevant plots, such as revenue by month, by visitor type and revenue rate, are created by using the pyplot functionality from matplotlib and the seaborn library. A brief description of the dataset is given by using panda's data frames functionalities.

Secondly, the data is processed by initially counting the number of N/A (not available) values per column and deleting rows which contain N/A values. Since it has been noticed that there are several duration features (such as administrative duration, informational duration and product related duration), it has been decided to drop rows which contain negative values for those variables. Moreover, since some of the duration values are different from zero even though the number of accesses to a specific type of page is recorded to be zero, all the rows which behave in the described way have been deleted from the data set. The whole analysis of rows to drop is processed by a loop which adds the elements to drop to a list, being the starting point for the actual elimination of selected instances from the original data frame (see drop_rows.py code on the GitHub link [8]).

In order to deal with categorical variables, some dummy features are created. In particular, pandas' data frame feature get_dummies is applied for the purpose. After defining the difference between response variable and explanatory ones, the design matrix containing pre-processed predictors is standardized by applying the StandardScaler function from sklearn.

In order to get an overview of the variable's relationships, a correlation matrix is produced by using the seaborn library. After having analyzed all different correlations among the predictors, a Principal Component Analysis (PCA) is performed in order to verify the existence of a useful reduction of dimensionality.

The data is then split into training and test subsets by applying sklearn's train_test_split function and considering a test size of 20%.

3.2 Logistic Regression, Random Forest and Support Vector Machines

In order to find the best model aiming to predict the customers' shopping intention, several machine learning algorithms have been applied. Among them, logistic regression, random forest and support vector machines. The implementation of these methods relies on the use of sklearn's functionalities and results have been analyzed by tuning the parameters of the functions themselves. In particular, Random forest is tested under an increasing number of trees (analysing changes in [1,10,100,500,1000,2000]) and Support Vector Machine is performed for all the possible combinations of Kernels between linear, polynomial and radial with the C parameter changing in [0.1,1,10,100] and gamma varying in [0.001,0.01,0.1].

3.3 XGBoost

XGBoost algorithm requires a specific library, xgboost (xgb), in order to be performed. In particular, the XGBClassifier method from that library is used to find the combination of parameters which leads to the best model to predict results. In this sense, a number of trees changing in [10, 50, 100, 200, 500, 1000] and learning rates η's in [0.0001, 0.001, 0.01, 0.1, 0.3, 1] are analyzed. The algorithm is then optimized by examining early stopping through loss functions such as negative log-likelihood ("logloss" in eval_metrics option of xgb.fit) and binary classification error rate ("error" in eval_metrics option of xgb.fit). A more explicit outlook is then given by considering to stop when the loss over a range of 10 trees is too large, using early_stopping_rounds = 10 and eval_metrics = "logloss" in xgb.fit (see GitHub link [8]).

3.4 Final plots and comparisons

In order to compare all the methods analyzed, ROC curves are considered as useful indicators of the algorithms' performance. Aiming to plot ROC curves for all the different techniques used, matplotlib pyplot functionality has been used.

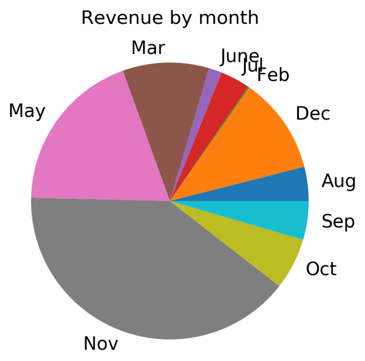
4 Results and discussion

4.1 Exploratory plots

In order to have a first insight into the data, different exploratory plots have been produced. Among these, the heatmap in Figure 11 (see Appendix) displays correlations among all the features. It is important to verify that explanatory variables are not too correlated, because of possible collinearity and unreliable results. From the correlation map, it is possible to notice that most of the covariates have correlations smaller than 0.3 and that the only significant ones are between Informational_Duration and Information (correlation equal to 0.62), between ProductRelated_Duration and ProductRelated (value of 0.86) and between ExitRates and BounceRates (value of 0.91). A study of the correlation between predictors and the revenue value can be seen in Figure 10 in section 4.4.

Considering the correlation set, Principal Components Analysis (PCA) has been performed in order to verify whether a dimensionality reduction approach would have been useful in the analysis or not. Starting from an original variable space of 26 features, the technique leads to a set of 21 principal components. This reduction is not very significant, as it is observed that variables are not, in general, extremely highly correlated. Due to this result, it has been decided not to use the reduction obtained with PCA, but to conduct further analysis with the original set of explanatory variables.

Other relevant plots are connected to the analysis of the revenue of customers in association to other feautures. For example, it is possible to state that the month which has the highest number of costumers visiting online shopping websites is November, followed by May (see left graph in Figure 1). The one that has the least number of visits is February. Moreover, it is clear that most of the analyzed costumers are returning visitors (right graph in Figure 1) and that the number of false revenues (costumers which did not finalize the purchase) is greater than the number of true ones (Figure 2).



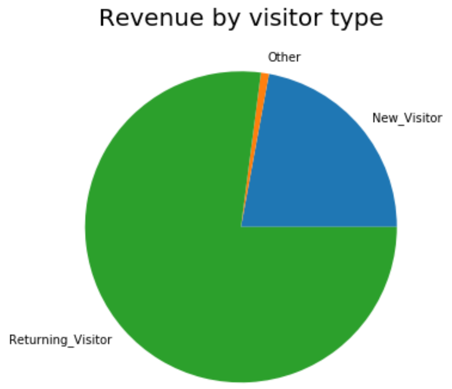


Figure 1: Left:Revenue by month, Right:Revenue by visitor type

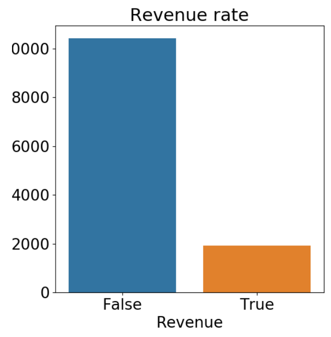


Figure 2: Revenue rate

4.2 Comparison of the methods

Different machine learning methods have been compared in order to determine which one leads to the most accurate fitting of the data. In particular, both accuracy scores and AUC values have been analyzed as performance measurements and confusion matrices have been produced in order to visualize the amount of correctly guessed targets.

4.2.1 Logistic regression

The first method applied, as well as the simplest, is logistic regression. The logistic regression accuracy results in 0.8832 and the AUC equals 66.82%. The produced confusion matrix (left graph in Figure 3) shows that the method performs really well in classifying the false revenues, since the correct guesses rate is 0.98. However, the classification of true revenues is not satisfactory, since 64% of costumers who finalized the purchase are wrongly classified as false revenue. Since the performance measurements do not seem totally adequate in terms of model fitting, more complex methods are evaluated.

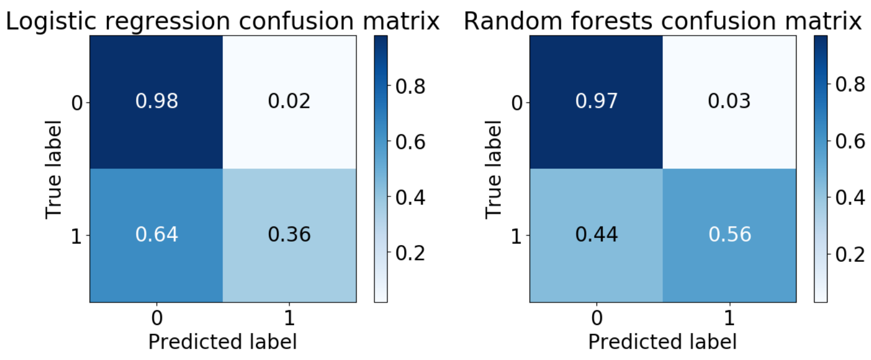


Figure 3: Confusion matrix for logistic regression (left) and random forests (right).

4.2.2 Random Forests

Dealing with Random Forest, the algorithm has been tested with different numbers of trees and its performance has been evaluated in all the analyzed cases (Figure 4). In particular, the number of trees which leads to the best results is found to be 100. In fact, the model which uses this number of trees returns the highest accuracy (0.9015) and an AUC value equal to 75.83%, whereas the best AUC (which is 75.84%) is given by the model using 2000 trees. Since the difference in terms of AUC is minimum and because of efficiency matters, it has been chosen to consider 100 as the number of trees which yields the most accurate result; in fact, performing random forest with 2000 trees requires 14.90 seconds of process time, while the same algorithm with 100 trees only takes 0.80 seconds. The confusion matrix produced by fitting the model through a random forest procedure, which considers 100 trees, can be seen in left graph in Figure 3. The result is more satisfactory than the one obtained for logistic regression, since the former method is more accurate in the classification of true revenues, performing about 20% better in their prediction.

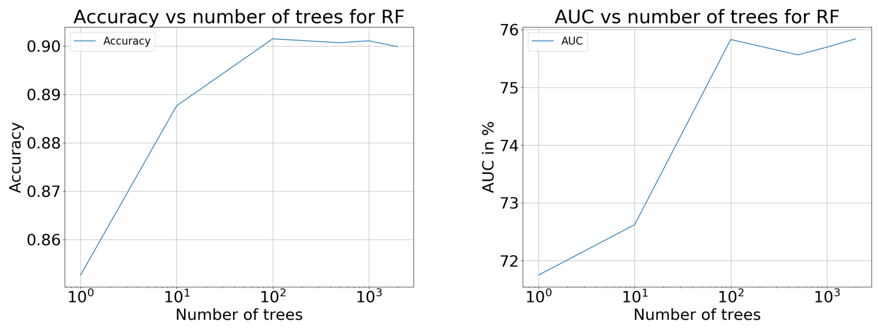


Figure 4: Performance of random forests with dependence on number of trees in terms of accuracy (left) and AUC (right).

4.2.3 Support Vector Machines (SVM)

Another method applied is Support Vector Machines (SVM), for which different combinations of kernel, C and γ parameters are studied. In particular, the analysis determines C=10, γ=0.1 and a radial basis function (rbf) kernel as the best possible combination in terms of the AUC value (equal to 72.40%, see right graph in Figure 5). The accuracy score can behave as a biased index in the best model determination, hence it is more desirable to evaluate AUC values in order to provide the most relevant solution. In particular, the mentioned combination of parameters also gives one of the highest accuracy values which has been found to be equal to 0.8856 (left graph in Figure 5). The confusion matrix produced by SVM with the given combination of parameters (left graph in Figure 6) displays a weaker classifier in comparison to the one returned by random forest, although the prediction ability appears improved if compared to the one obtained through the logistic regression method.

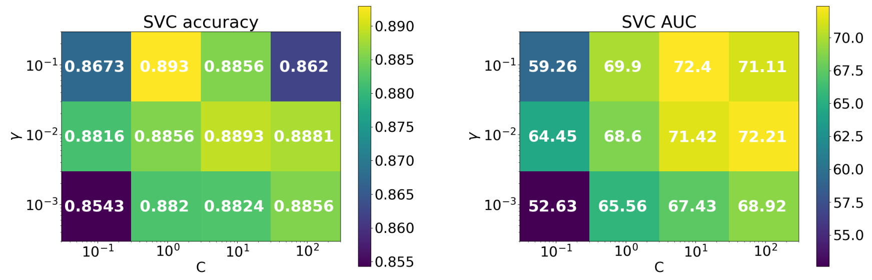


Figure 5: Performance of Support Vector Machines with dependence on parameters C and γ in terms of accuracy (left) and AUC (right).

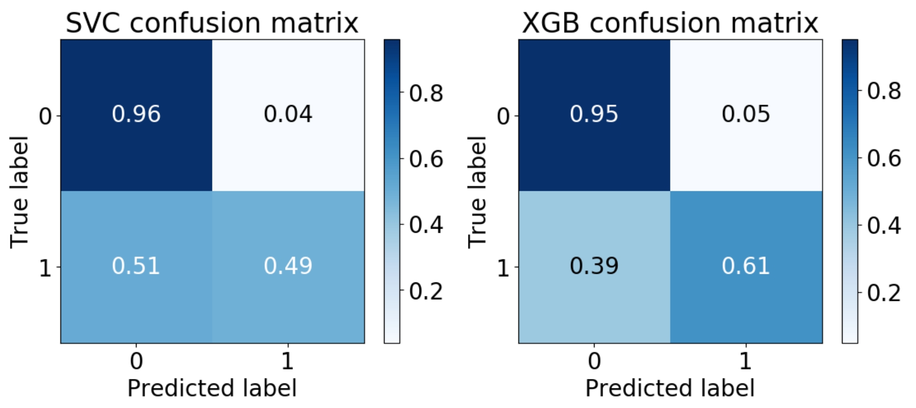


Figure 6: Confusion matrix for Support Vector Machines (left) and XGBoost (right).

4.2.4 XGBoost

The last algorithm analyzed is XGBoost, for which different combinations of learning rate η and number of trees are explored. Interestingly, from the study of parameters, it is clear that the learning rate does not lead to a considerable difference in terms of performance, since the number of trees is the only specification which determines conspicuous changes in that sense (Figure 7). The best results are provided by the model which considers 100 trees, leaving the learning rate as default (equal to 0.3). In particular, the accuracy score is 0.9008 and the AUC value is computed to be 78.06%. From an optimization perspective, different numbers of trees and relative models are analyzed by considering negative log-likelihood (logloss) and binary classification error rate (error) as evaluation metrics. The plots in Figure 8, related to the results obtained by these metrics, do not provide a clear insight into the optimal number of trees. In order to determine this value and get a more precise understanding, the logloss metric is applied to perform early stopping when the loss over a range of 10 estimators (trees) is too high. The result of this analysis indicates that overfitting is observed between 90 and 100 trees and that 92 is the best iteration, meaning that the smallest loss is obtained by using 92 trees. Fitting the XGBoost model with 92 trees determines an accuracy score of 0.9019 and an AUC value of 78.35%. The confusion matrix produced by XGBoost (right graph in Figure 6) using 92 trees is the most accurate among all used methods, since it allows to classify correctly 95% of false revenues and 61% of true ones.

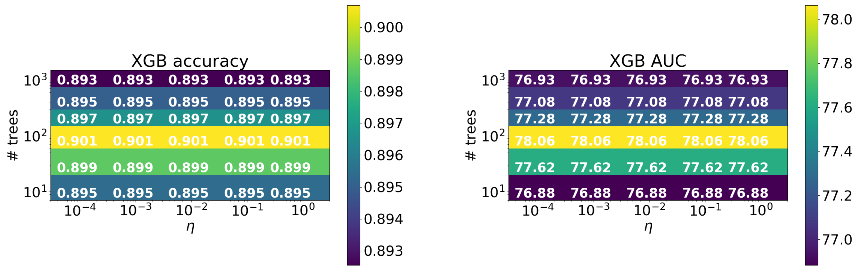


Figure 7: Performance of XGBoost with dependence on number of trees and γ parameter in terms of accuracy (left) and AUC (right).

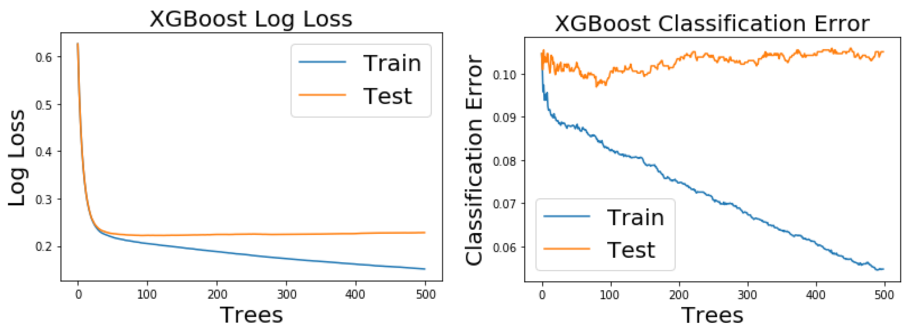


Figure 8: Study of overfitting through log-loss (left) and through classification error (right)

4.3 Summary of results

A summary of the obtained results is given in Table 1. XGBoost emerges to be the best method, followed by Random Forest, Support Vector Machine and then Logistic regression. The outcomes of the first two tree-based methods are similar. Moreover, both Logistic regression and Support Vector Machines have very low sensitivity to observations far from the decision boundary and their conclusions are therefore similar.

Receiver operating characteristic curves in Figure 9 offer a graphical visualization of obtained results from which it can be endorsed that XGBoost and random forests perform better than the other methods.

The results of the analysis also match well with the reference values from the original article [7] (Table 2). In terms of accuracy, presented results are even better than the reference ones. True-positive rate and true-negative rate, as referred to in the article, can be compared to diagonal elements of confusion matrices for random forests and SVM, which are applied in the original study as well (see Figure 3 and 6). In particular, in the case of the random forest method, results appear to be almost the same, whereas for SVM slightly better results have been obtained in this report. However, it has to be remarked that advanced data analysis techniques, such as oversampling and feature selection, yielding superior results compared to the ones presented have not been performed.

Method	Accuracy	AUC in %	Process time [s]
Logistic regression	0.8832	66.82	0.03
Support Vector Machines	0.8856	72.40	2.32
Random forests	0.9015	75.83	0.80
XGBoost	0.9019	78.35	1.61

Table 1: Summary of accuracy and AUC results of studied methods as well as time demandingness of every method ordered from the least precise method to the most precise one.

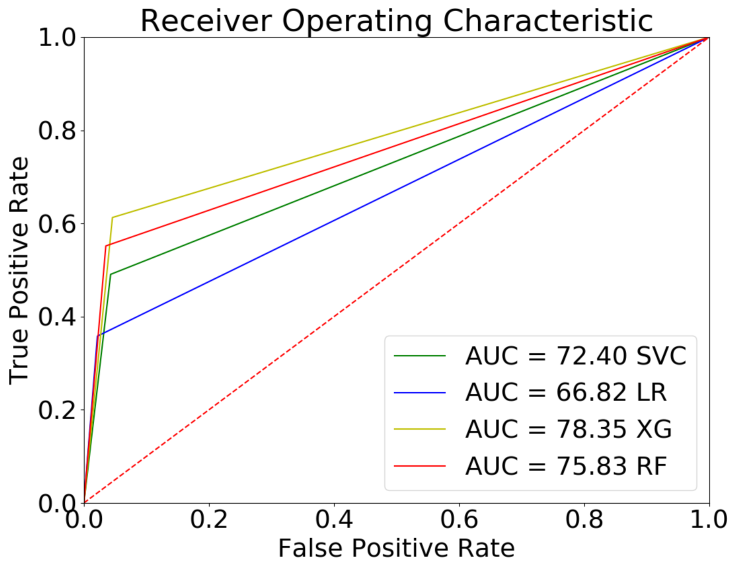


Figure 9: Receiver operating characteristics for Support vector classifier (SVC), Logistic regression (LR), XGBoost (XG) and Random forests (RF).

Method	Accuracy	TPR	TNR
Random forests	0.8951	0.57	0.96
Support Vector Machines	0.8614	0.46	0.92

Table 2: Summary of reference accuracy, true-positive rate (TPR) and true-negative rate (TNR) results from [7]. For Support Vector Machines RBF kernel has been used.

4.4 Further analysis for the best models

An interesting insight can be given by determining which explanatory variable has the most influence in predicting customers' revenue. This relationship has been evaluated through different processes. A first result is given by considering the correlations between each feature and the response variable. As diplayed in Figure 10, this raw method indicates PageValues as the most correlated feature to the dependent variable. Another evaluation is provided by analyzing the feature importance in the XGBoost fitted model using three different options:

- Type weight, in which the importance is considered as the number of times an explanatory variable is used to split the data across all trees. Figure 12 shows that in this case, PageValues feature is identified as the most relevant one, followed by Administrative_Duration and ProductRelated_Duration.

- Type cover, in which the importance is again given as the number of times an explanatory variable is used to split the data across all trees, but weighted by the number of training data point passing through those splits. In this case, the most important feature is Month_nov, followed by PageValues and Month_Feb (Figure 13).

- Type gain, in which the importance is evaluated as the average training loss reduction obtained using a certain variable for splitting. In Figure 14, it can be seen that PageValues stands out as the most significant feature and all the others seem relatively unimportant.

The tree indexes are not totally in agreement and, not knowing which one is to be trusted, it is best to rely on another method for the feature evaluation. The SHAP index can be used to that aim for tree-based models and it is considered more reliable due to its consistency and accuracy. From the bar plot in Figure 16, PageValues feature emerges to be the most crucial feature, in agreement with the importance of type gain. The right plot in Figure 16 combines feature importance with feature effects, representing the variable relevance for every point of the dataset. From this graph, which shows the same priority order as the bar plot, it is also possible to get a first insight into the relationship between the value of a feature and the impact on the prediction. In particular, lower values of PageValues lead to lower values of the response variable and vice versa, whereas lower values of ExitRates make predictions higher and elevated values of ProductRelated_Duration produce lower responses.

Finally, the SHAP index is also used to evaluate the main feature in the second best model, obtained by applying random forest. The results are similar to the ones obtained for the XGBoost model and, again, PageValues is indicated as the most relevant feature. The bar plot in Figure 18 also shows that the variables have equal importance in predicting positive revenue and negative without distinctions.

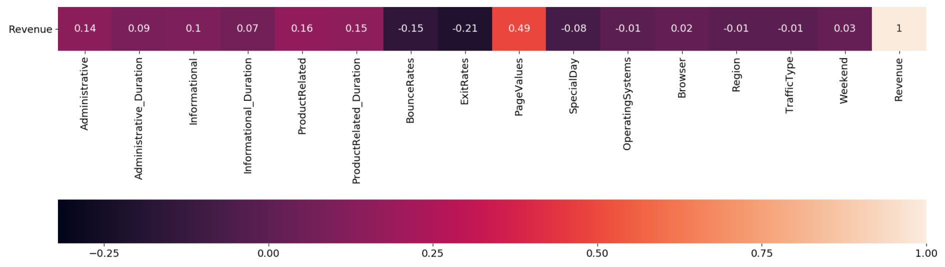


Figure 10: Correlation heatmap for revenue

5 Conclusions

Supervised learning methods have been applied aiming to perform a fit of online shoppers' intentions. Thanks to these features, e-commerce websites can study users' behaviour and change their content in order to increase chances of finalizing a purchase.

The algorithms studied are Logistic regression, Random Forest, Support Vector Machines and Extreme Gradient Boosting (XGBoost). The analysis has underlined that XGBoost is the best method in terms of predictions of target values, leading to more impressive performance results, namely accuracy of 0.9019 and AUC 78.35 %. In particular, this machine learning tool includes regularization and overfitting check, as well as fast processing, as main features. Moreover, it is highly flexible in the sense that it allows to customize optimization parameters and evaluation criteria, being able to handle missing values by default. In addition, XGBoost performs tree pruning by removing splits which do not correspond to a positive gain, while allowing to set a cross-validation procedure at each iteration in order to get the right optimum number of boosting iteration in an individual run.

Random forest tends to perform similarly to XGBoost, since its ability in classifying the targets correctly can be considered remarkable, whereas logistic regression and support vector machines appear less notable.

Based on the evaluation of performance measurements, it can be concluded that tree-based models are the ones that fit the data best and that the model obtained by considering XGBoost could be applied to predict whether a new user would finalize or not his purchase on a specific e-commerce.

It would be interesting to study how results can be improved by applying more advanced techniques of feature selection or oversampling, as implemented in the article [7]. In addition, the main goal of a predictive model would be to classify real-time users' intentions and a future development of the problem could be in this sense.

References

[1] Morten Hjorth-Jensen, (2019), Lectures Notes in FYS-STK4155. Data Analysis and Machine Learning: Logistic Regression, https://compphysics.github.io/MachineLearning/doc/pub/LogReg/html/LogReg.html

[2] Morten Hjorth-Jensen, (2019), Lectures Notes in FYS-STK4155. Data Analysis and Machine Learning: From Decision Trees to Forests and all that, https://compphysics.github.io/MachineLearning/doc/pub/DecisionTrees/html/DecisionTrees.html

[3] Morten Hjorth-Jensen, (2019), Lectures Notes in FYS-STK4155. Data Analysis and Machine Learning: Support Vector Machines, https://compphysics.github.io/MachineLearning/doc/pub/svm/html/svm.html

[4] Sundaram , Ramya Bhaskar. "Understanding the Math behind the XG-Boost Algorithm." Analytics Vidhya, 10 Jun. 2016, https://arxiv.org/pdf/1603.02754.pdf.

[5] Chen, Guestrin. "XGBoost: A Scalable Tree Boosting System." Analytics Vidhya, 7 Aug. 2019, www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/.

[6] Morten Hjorth-Jensen, (2019), Lectures Notes in FYS-STK4155. Data Analysis and Machine Learning: Neural networks, from the simple perceptron to deep learning, https://compphysics.github.io/MachineLearning/doc/pub/NeuralNet/html/NeuralNet.html

[7] Sakar, C. O., Polat, S. O., Katircioglu, M., Kastro, Y. (2018). Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. Neural Computing and Applications, 31(10), 6893–6908. https://doi.org/10.1007/s00521-018-3523-0

[8] Project 3 repository location https://github.com/storejar/Data_analysis_and_machine_learning/tree/master/Project3

[9] Clifton, B. (2010). Advanced Web metrics with Google Analytics. Indianapolis, Ind: Wiley Pub.

[10] James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer Texts in Statistics. Springer New York. https://doi.org/10.1007/978-1-4614-7138-7

[11] Molnar, C. (n.d.). Interpretable machine learning. https://christophm.github.io/interpretable-ml-book/

6 Appendix

Variables	Description	Categories	Min. value	Max. value	SD
Administrative	Number of pages visited relative to the account management		0	27	3.32
Administrative duration	Seconds spent on pages related to the account management		0	3398	176.70
Informational	Number of pages visited regarding the Web site, communication and address information of the shopping site		0	24	1.26
Informational duration	Seconds spent on informational pages		0	2549	140.64
Product related	Number of visited pages related to products		0	705	44.45
Product related duration	Seconds spent on product related pages		0	63.973	1912.25
Bounce rate	Average of the bounce rates relative to the pages visited by the user		0	0.2	0.04
Exit rate	Average of the exit rates relative to the pages visited by the user		0	0.2	0.05
Page value	Average of the page values relative to the pages visited by the user		0	361	18.55
Special day	Proximity of the visiting date to a special day		0	1	0.19
Operating Systems	Type of operating system used by the visitor	8			
Browser	Type of browser used for the access	13			
Region	Geographic region from which the access occurs	9			
Traffic type	Type of source from which the user has arrived to the web page	20			
Visitor type	New Visitor, Returning Visitor, or Other	3			
Weekend	Boolean value indicationg if the visit happened during the weekend	2			
Month	Month in which the visit occurs	12			
Revenue	Class label indicating whether during the visit a purchase happened or not	2			

Table 3: Variables description including their minimum and maximum values with standard deviation (SD). For categorical variables number of categories is shown.

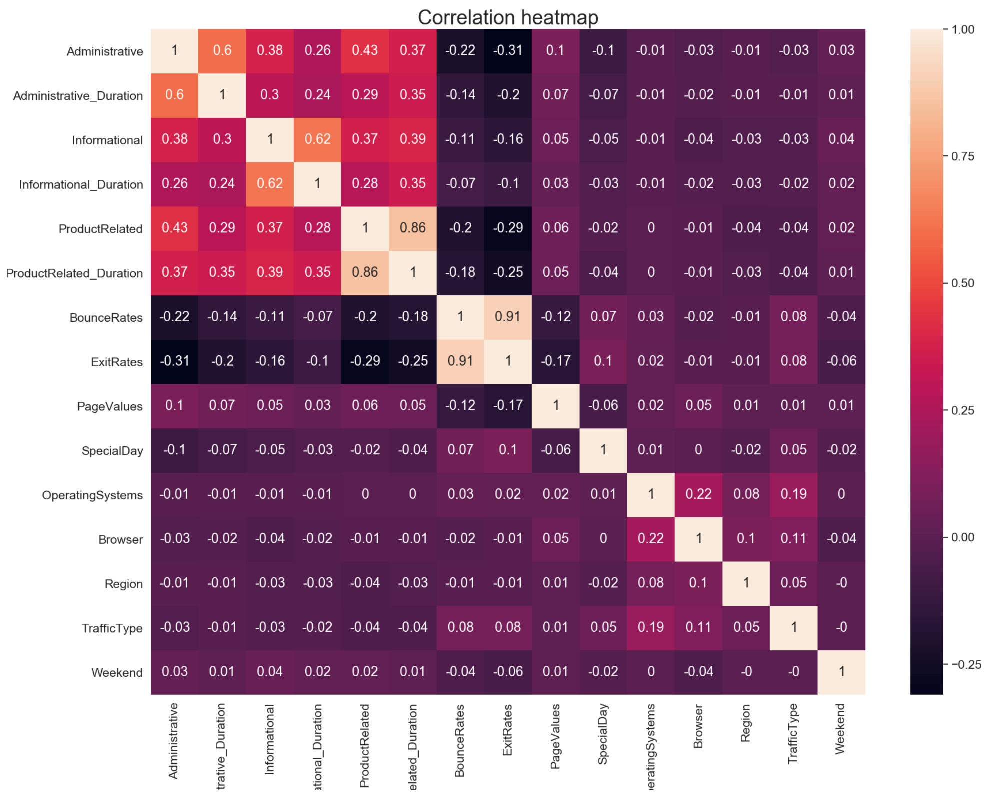


Figure 11: Correlation matrix heatmap

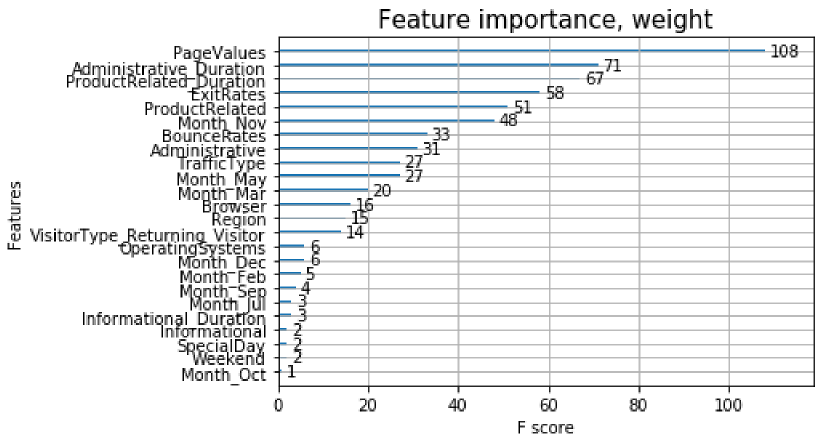


Figure 12: XGBoost feature importance, type weight

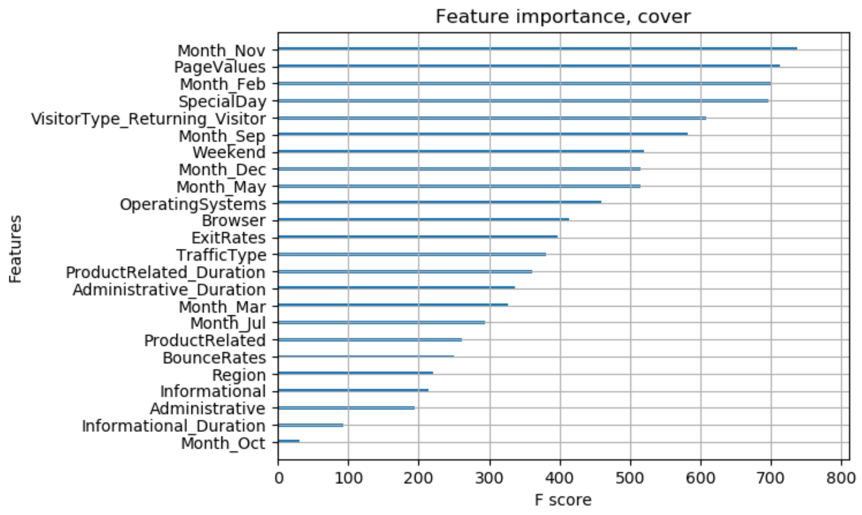


Figure 13: XGBoost feature importance, type cover

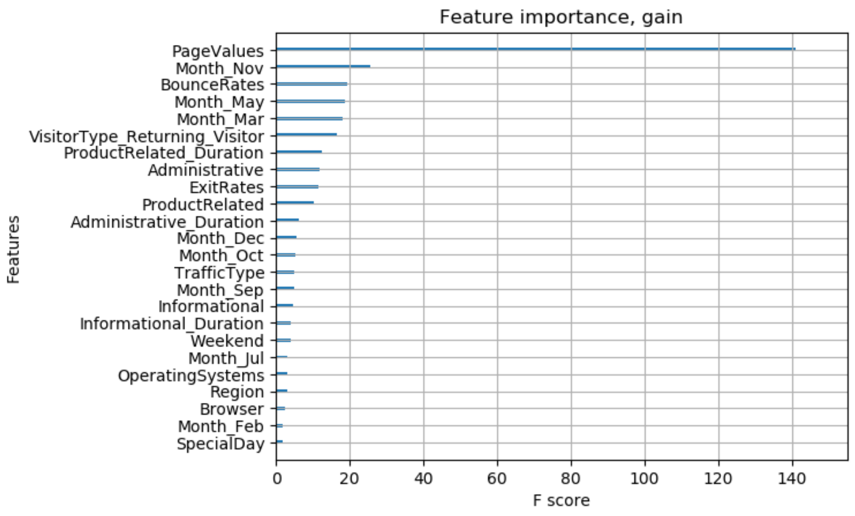


Figure 14: XGBoost feature importance, type gain

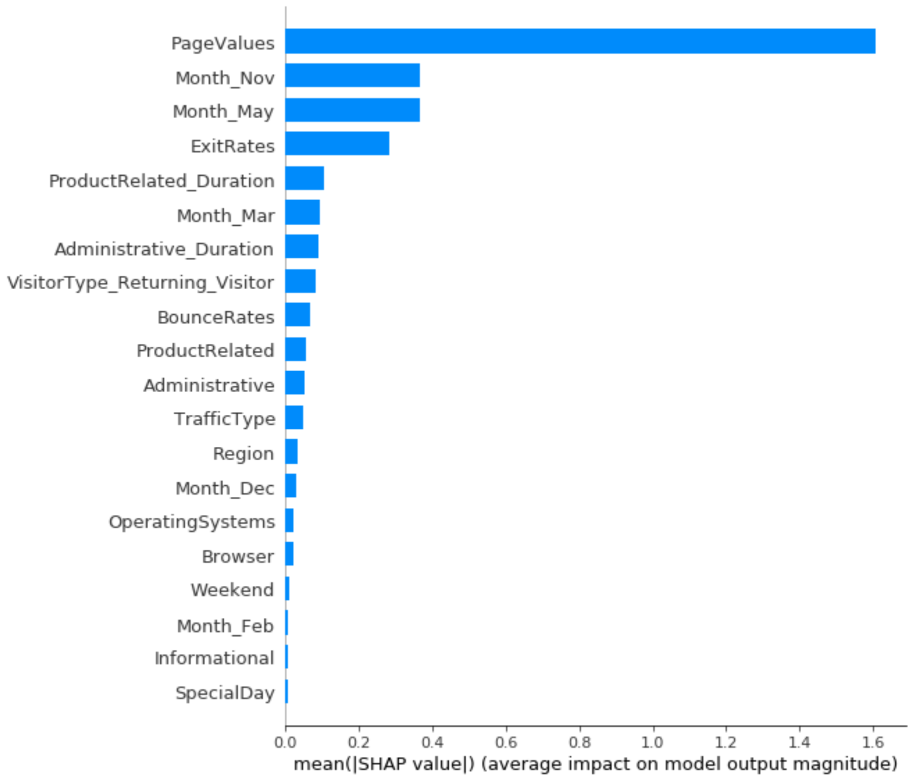


Figure 15: Bar SHAP for XGBoost.

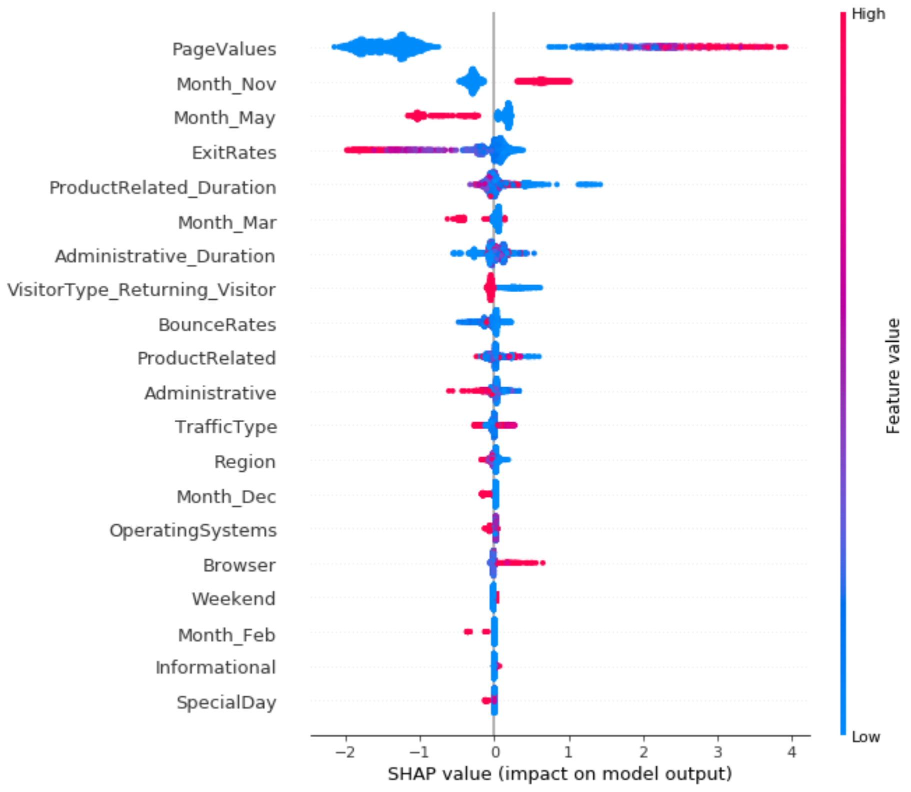


Figure 16: SHAP for XGBoost.

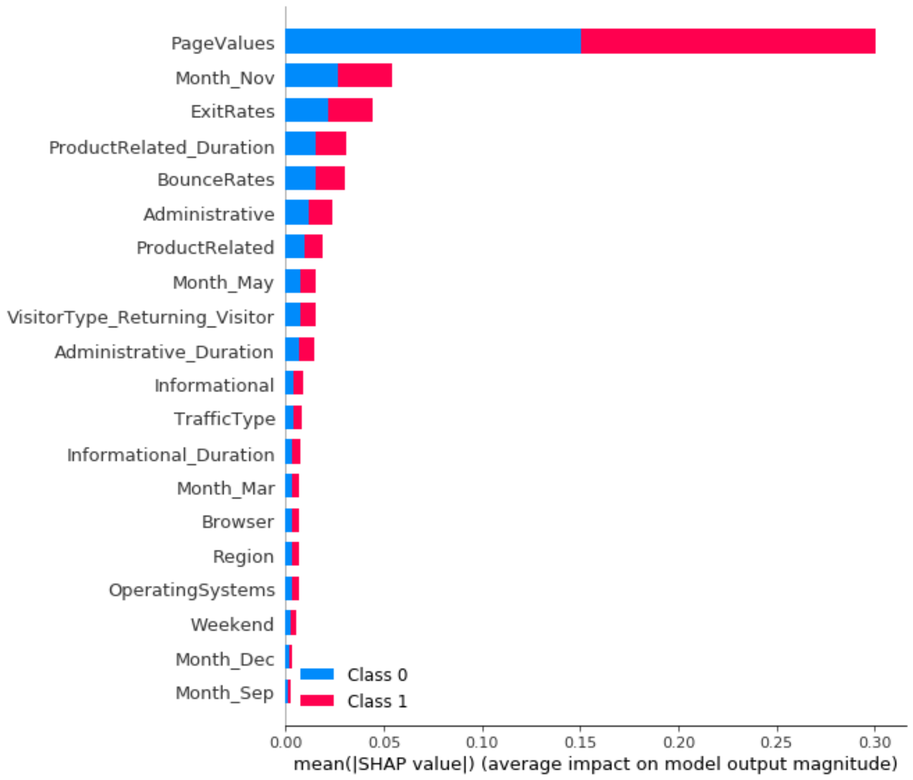


Figure 17: Bar SHAP for Random Forest.

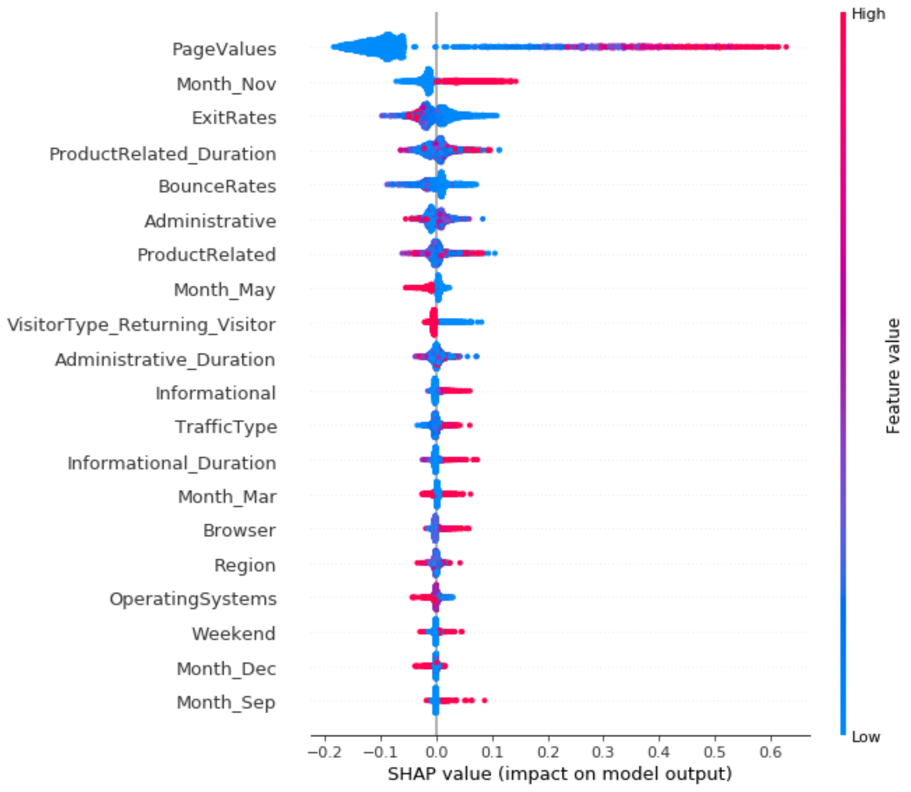


Figure 18: SHAP for Random Forest.