Binary classification in Bank Marketing

Yixun Kang
Data Science Institute, Brown University
GitHub [1]

1 Introduction

The project explores the application of machine learning for binary classification in bank telemarketing. Central to the analysis is the 'Bank Marketing' dataset, initially from the UCI Machine Learning Repository and expanded upon in the study by Moro, Cortez, and Rita titled '*A Data-Driven Approach to Predict the Success of Bank Telemarketing*,' published in Decision Support Systems [2][3]. This dataset is notable for including five additional attributes representing social and economic indicators from a country with a population of approximately 10 million [2].

The objective is to develop a predictive model capable of accurately assessing the likelihood of clients subscribing to a term deposit. Such a model holds significant importance in bank marketing campaigns, which often involve contacting numerous clients, a process that is both time-consuming and resource-intensive. The use of a predictive model in this context aims to streamline the process, focusing on clients with a higher probability of positive response. This targeted approach is expected to conserve resources like time and human effort and potentially enhance the success rate of these campaigns.

To achieve this, our analysis will delve into a dataset containing 20 distinct input attributes. These attributes provide a comprehensive range of data, from basic client demographics, like age and job type, to more nuanced details from previous telemarketing interactions, as well as key socio-economic indicators. Ultimately, by utilizing the predictive power of this model, we aim to revolutionize the way banks approach telemarketing, making it more efficient and effective.

The study by Moro, Cortez, and Rita in 2014 is a key reference for this study. It evaluated four machine learning models—logistic regression, decision trees, neural networks, and support vector machines—using AUC and ALIFT as evaluation metrics [3]. The neural network model in their study achieved an AUC of 0.8 and an ALIFT of 0.67 [3], providing a valuable benchmark for performance comparison in this project.

2 EDA

2.1 Target Variable

In the dataset, the target variable, 'y', signifies whether a client will subscribe to a term deposit or not, with 'yes' and 'no' as the two possible values. The data exhibits an imbalance, with the majority (88.7%) of observations falling into class 0, indicating clients who will not subscribe to the term deposit.

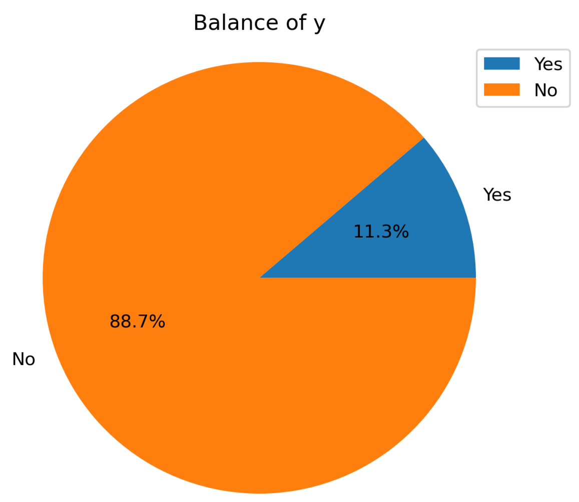

Figure 1. The ‘Bank Marketing’ data set is imbalanced. Only 11.3% of the observations belong to class 1.

2.2 Feature Analysis

To enhance model interpretability, it may be beneficial to consider removing some socio-economic features. While these features hold significance in financial problems, their explanation in human data can be challenging. The linear associations among these features are illustrated using a correlation heatmap.

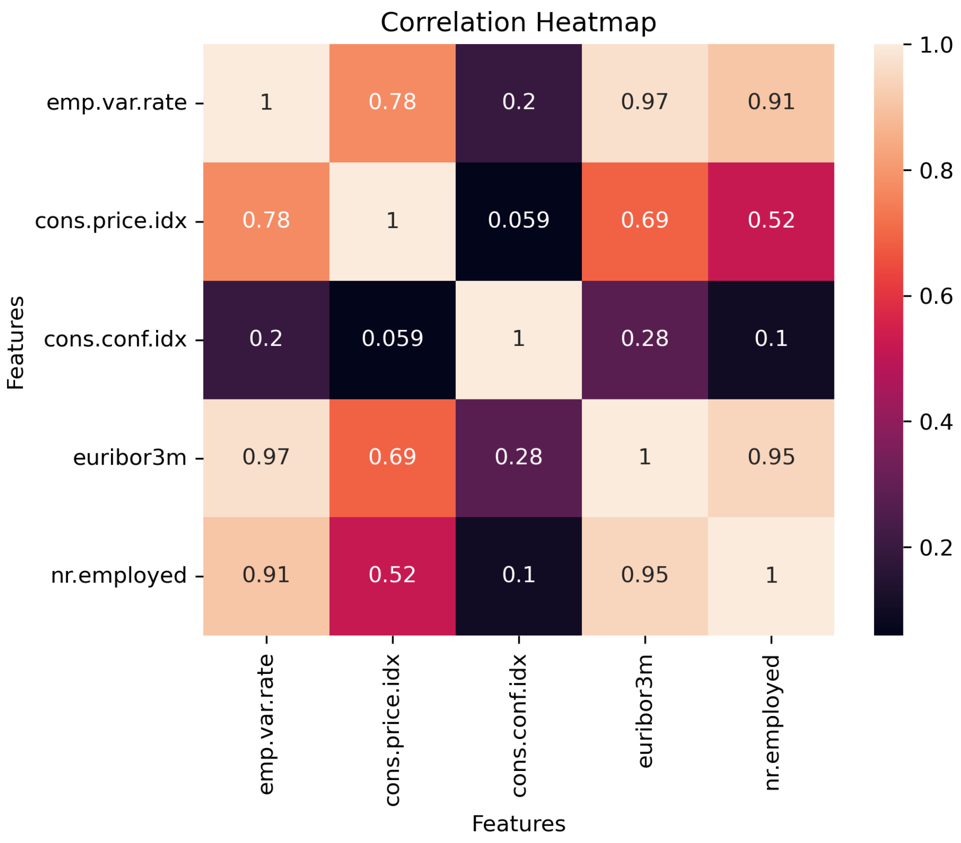

Figure 2. ‘euribor3m’ and ‘nr.employed’, ‘euribor3m’ and ‘emp.var.rate’, and ‘nr.employed’ and ‘emp.var.rate’ are strongly correlated.

The heatmap indicates strong correlations between 'euribor3m' and 'nr.employed', as well as between 'euribor3m' and 'emp.var.rate', and 'cons.price.idx' and 'emp.var.rate'. Based on these observations, it is suggested to remove 'emp.var.rate' and 'euribor3m' to improve the model's convergence time. This decision is guided by the aim of streamlining the model while maintaining its efficacy.

2.3 Future Information

This dataset includes information about future events, which presents a challenge in supervised machine learning due to the unavailability of such information at the time of model deployment. The ‘duration’ feature, representing the last contact duration in seconds, is unknown before a call is made. Similarly, the ‘campaign’ feature, denoting the number of contacts made during a specific campaign for a client, including the last contact, cannot be precisely known before making a call. To address this issue, the ‘duration’ feature is removed, and the ‘campaign’ feature is modified to ‘ncalls,’ representing the number of contacts already made during this campaign for the specific client.

2.4 Missing Values

As indicated in the data description, categorical features with missing values are filled with the label ‘unknown.’ Additionally, in the ‘pdays’ feature, a value of 999 is used as a placeholder to indicate that the client was not previously contacted in a prior campaign.

An evaluation of missing values in the dataset reveals 7 features with missing values—6 categorical and 1 numeric. Handling missing values in categorical features involves preprocessing techniques such as the application of one-hot encoding and ordinal encoding. The ‘pdays’ feature, with 96% of observations having missing values, presents a significant challenge and requires careful consideration in the data-handling strategy.

2.5 Modification in the ‘pdays’ Features

An approach is introduced to ordinalize certain numeric values within ‘pdays.’ For instance, values such as 1, 2, 3, 4, 5, 6, and 7 in the original 'pdays' are grouped under a new ordinal label named 'contacted within a week.' This transformation effectively converts 'pdays' from a numeric to an ordinal feature with six ordered classes. As a result, the most frequent category in 'pdays' now represents clients who were never contacted. The unique values and their respective counts in the revised ‘pdays’ are shown in the table below:

Classes in the Transformed ‘pdays’	**Values Counts**
contacted immediately	15
contacted within a week	1162
contacted between 8 to 14 days	276
contacted between 15 to 21 days	56
contacted between 22 to 28 days	6
never contacted	39673

Table 1. 96% of the observations are classified as ‘never contacted’.

3 Methods

3.1 Splitting

In the implementation of Logistic Regression, Lasso, Ridge, Elastic Net, and Random Forest models, the sklearn’s `train_test_split` function with `stratify = y` is used to allocate 20% of the data points to the test set. This is a common practice for small datasets. The remaining 80% is processed using `StratifiedKFold` with `n_spits = 3` and `shuffle = True`. This k-fold cross-validation method enhances the robustness of model evaluation by training and validating each model on different subsets of the data, thereby mitigating the impact of any initial random data split.

For XGBoost models, a different approach is taken, dividing the dataset into three parts—training, validation, and test—in a 6:2:2 ratio, using `train_test_split` with `stratify = y`. This split ensures comprehensive evaluation, including hyperparameter tuning and testing on new data. This stratified division helps prevent overfitting and comprehensively evaluates the XGBoost model's effectiveness.

3.2 Preprocessing

All numeric features are normalized using the standard scaler to achieve a mean of 0 and a variance of 1, ensuring equal contribution to the model training. Categorical and ordinal features are preprocessed with one-hot encoding and ordinal encoding, respectively, followed by standard scaling for consistent feature scales.

3.3 ML Pipeline

Six machine learning algorithms are employed: Logistic Regression, Lasso, Ridge, Elastic Net, Random Forest, and XGBoost. For each algorithm, various hyperparameters are tuned, as illustrated in the table below:

Model	**Hyperparameters**
Logistic Regression	`class_weight`: [None, ‘balanced, {0:1, 1:5}, {0:1, 1:10}
Lasso	`C`: [0.001, 0.01, 0.1, 1, 10, 100] `class_weight`: [None, ‘balanced, {0:1, 1:5}, {0:1, 1:10}
Ridge	`C`: [0.001, 0.01, 0.1, 1, 10, 100] `class_weight`: [None, ‘balanced, {0:1, 1:5}, {0:1, 1:10}
Elastic Net	`C`: [0.001, 0.01, 0.1, 1, 10, 100] `l1_ratio`: [0.3, 0.5, 0.8] `class_weight`: [None, ‘balanced, {0:1, 1:5}, {0:1, 1:10}
Random Forest	`max_depth`: [1, 3, 10, 30, 100] `max_features`: [0.15, 0.5, 0.75, 1] `class_weight`: [None, ‘balanced, ‘balanced_subsampe’, {0:1, 1:5}, {0:1, 1:10}
XGBoost	`reg_alpha`: [0e0, 1e-2, 1e-1, 1e0, 1e1, 1e2] `max_depth`: [1, 3, 10, 30, 100] `scale_pos_weight`: [1, 7.87]

Table 2. Hyperparameter tuning options for various candidate models.

The chosen evaluation metric in this project is the F1 score. Given the imbalanced nature of the data, where the majority belongs to class 0 (indicating clients who will not subscribe to a term deposit), special attention is paid to mitigating false negatives. In the context of this financial problem, a false negative implies that the bank will not call the client who will make the subscription, potentially impacting the bank's profitability adversely. Additionally, minimizing false positives is equally crucial. A false positive scenario could lead to unnecessary calls to clients who won't subscribe, incurring costs for the bank. Therefore, striking a balance in minimizing both false negatives and false positives is essential for the success of the predictive model in this financial context.

Each algorithm is tested with three different random states to account for potential variations in model performance. Except for the XGBoost model, `GridSearchCV` is used, and its `cv` argument is set to the previously defined cross-validation pipeline. The use of multiple random states serves the purpose of quantifying uncertainties in the F1 score, accounting for both data splitting variations and the inherent non-deterministic nature of some machine learning methods. For XGBoost, `ParameterGrid` is used, with `early_stopping_rounds = 50`.

4 Results

4.1 Test Results

The table below shows the average F1 scores and standard deviations for various candidate algorithms. All algorithms outperform the baseline F1 score of 0.2025. Notably, the Random Forest model has the highest average F1 score at 0.5046, more than double the baseline, showing it performs the best. The Ridge model, with an average F1 score of 0.4562, stands 84.26 standard deviations above the baseline. Considering these results, the Random Forest model is selected as the final model due to its superior average performance.

Model	Mean F1 Score	Standard Deviation	Standard Deviation Above Baseline
Logistic Regression	0.4547	0.003276	76.98
Lasso	0.4685	0.01127	23.60
Ridge	0.4562	0.003011	84.26
Elastic Net	0.4648	0.01405	18.67
Random Forest	0.5046	0.004389	68.83
XGBoost	0.4493	0.01361	18.13

Table 3. Summary of test results across various candidate models.

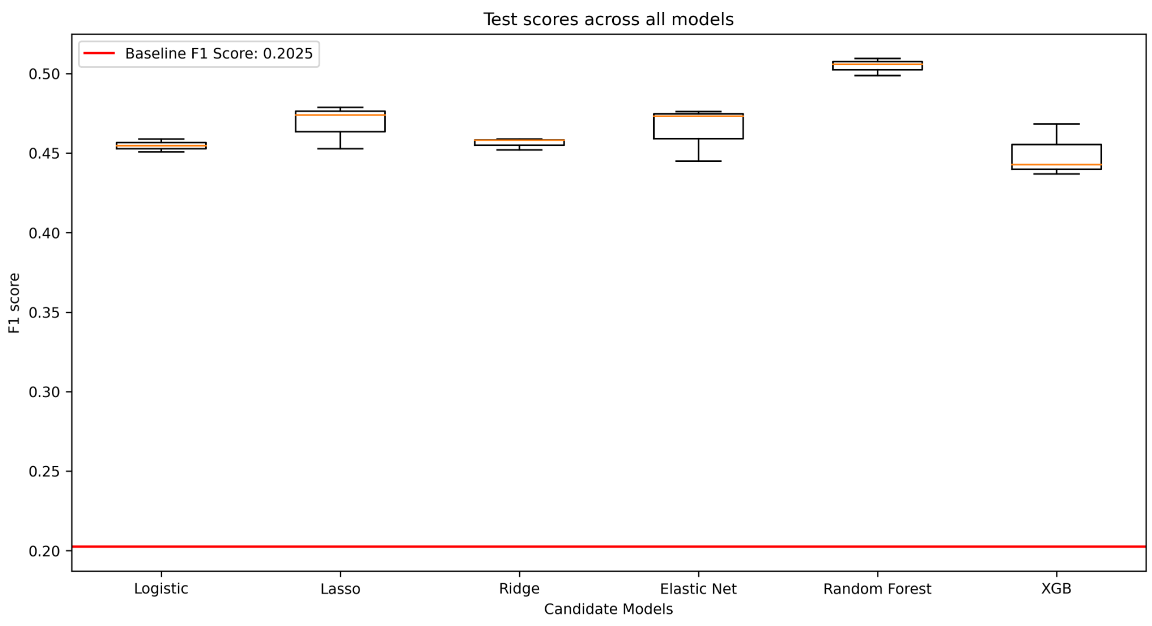

Figure 2. Visualization of test results across various candidate models.

4.2 Global Features Importances

Permutation feature importance, `RandomForestClassification.feature_importances_`, and SHAP are used to determine which features have a strong impact on the predictions.

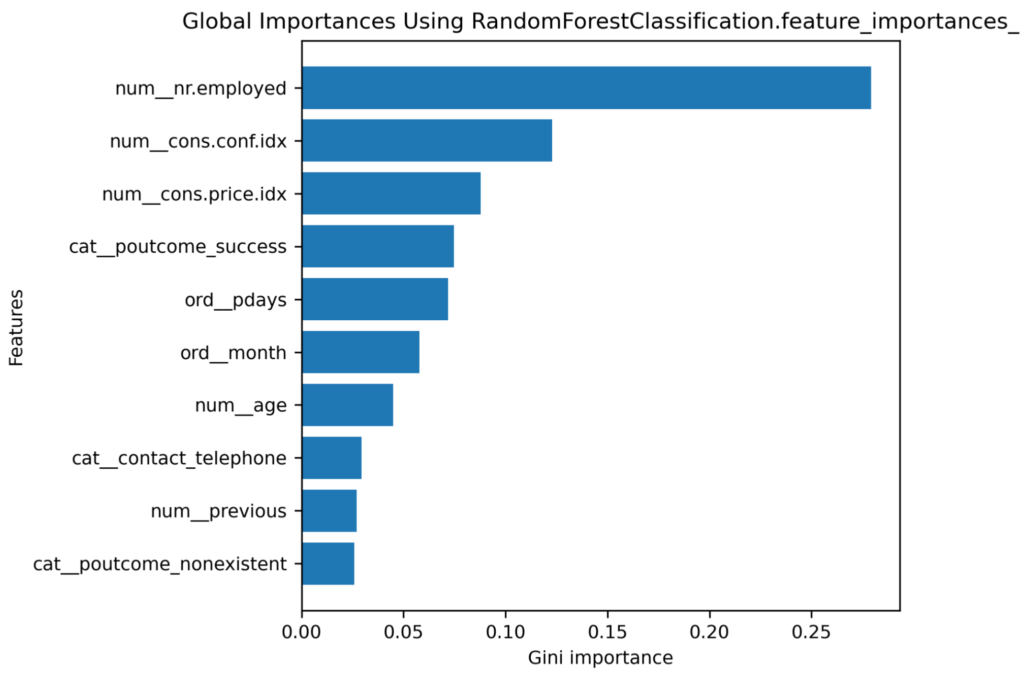

Figure 3. Top 10 important features using `RandomForestClassification.feature_importances_`.

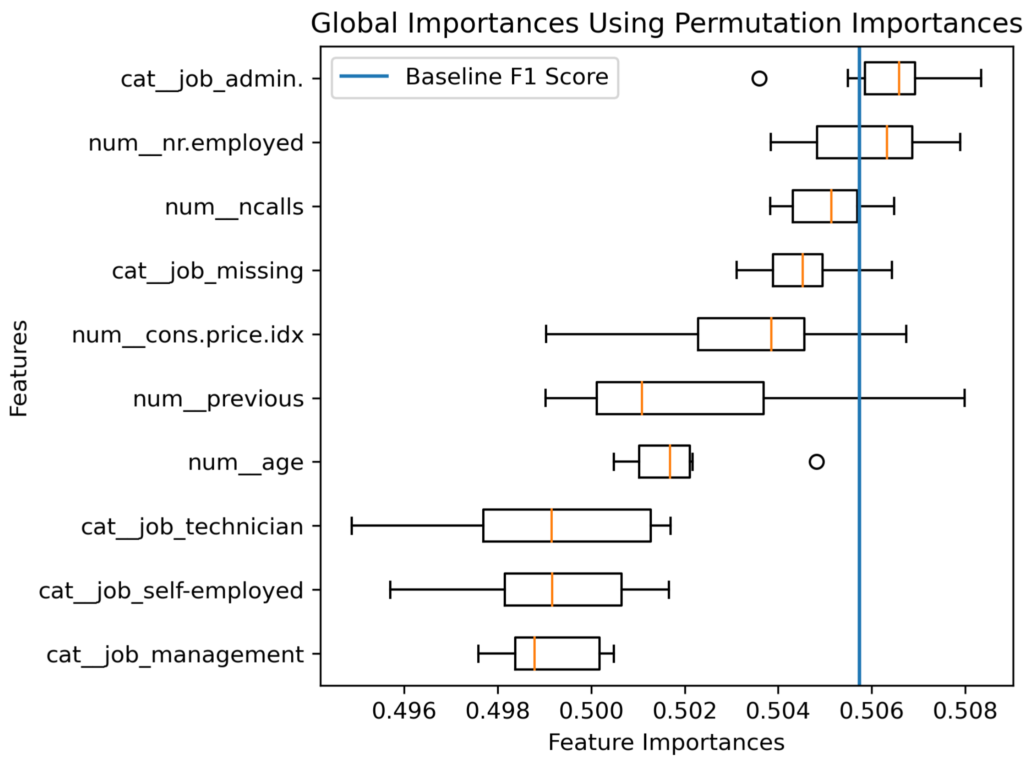

Figure 4. Top 10 important features using permutation feature importance.

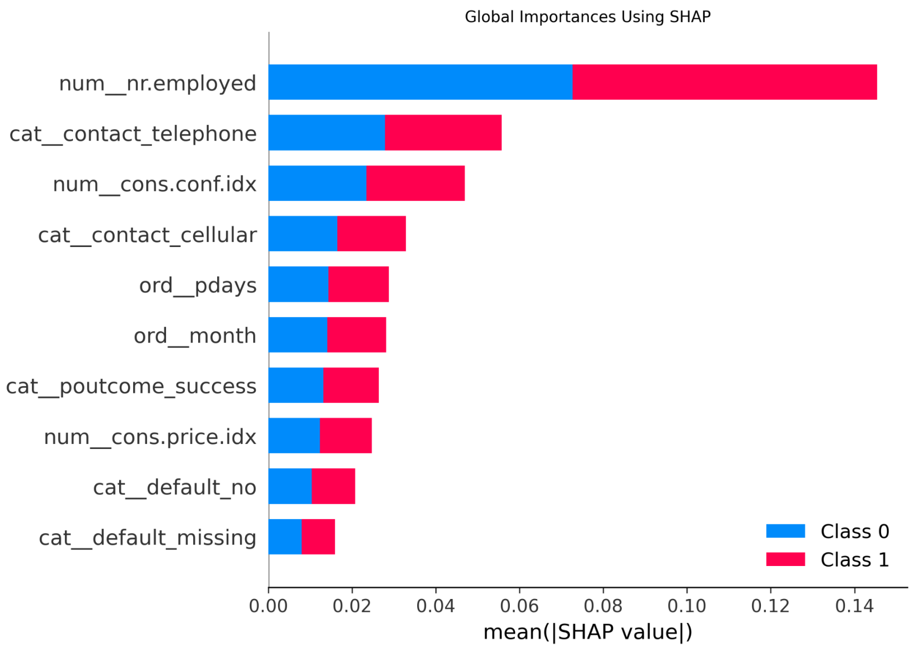

Figure 5. Top 10 important features using `shap.summary_plot`.

In the three figures, it's evident that 'nr.employed' is the most significant feature in the model. In Figure 3 and Figure 5, there is an overlap of seven features, though their rankings vary. Moreover, Figure 4 presents the most varied set of top ten significant features. Therefore, it is hard to determine the least influential feature due to the various rankings.

4.3 Local Features Importances

For index 872, labeled as 'yes', the force plot reveals that ‘num__nr.employed’, ‘cat__job_blue-collar’, and ‘cat__poutcome_failure’ significantly reduce the predictive score to 0.27, leading to an incorrect prediction of ‘0’. This outcome occurs despite the positive influence of features like ‘cat__contact_telephone’ and ‘ord__month’, underscoring the strong negative impact of the former set of features.

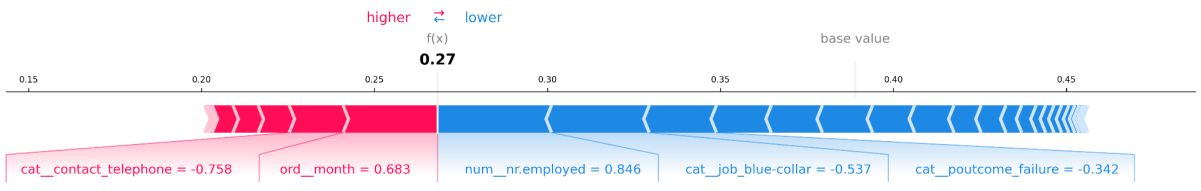

Figure 5. Local features importances for index 872 in the test set.

In the context of bank marketing, the model predicts that because this client is a blue-collar and the previous campaign targeting this client was unsuccessful, they are unlikely to subscribe to a term deposit. However, this turns out to be an incorrect prediction, suggesting that the final model struggles to accurately discern subtleties in individual client profiles and historical campaign interactions.

5 Outlook

The SHAP force plot above reveals that 'nr.employed' is a key factor in the model's prediction. However, it is hard to interpret ‘number of employees’ in the bank marketing context. Removing the three remaining socio-economic attributes might enhance the interpretability of the model.

The method of converting 'pdays' into an ordinal feature is uncommon. Typically, dealing with numeric missing values involves using a reduced-features model. But, given the imbalanced nature of this dataset, using such a model could decrease the predictive power.

To enhance the model's predictive capabilities, exploring neural networks and support vector machines, as seen in the study by Moro, Cortez, and Rita, could be beneficial. Neural networks, a type of deep learning, are particularly suitable for this project due to their capacity for capturing complex patterns and relationships within the data.

References

[1] Ihiroo. (n.d.). *Ihiroo/data1030_bank_marketing*. GitHub.
https://github.com/ihiroo/DATA1030_Bank_Marketing.git
[2] Moro,S., Rita,P., and Cortez,P.. (2012). Bank Marketing. UCI Machine Learning Repository. https://doi.org/10.24432/C5K306.
[3] Sérgio Moro, Paulo Cortez, Paulo Rita, A data-driven approach to predict the success of bank telemarketing, Decision Support Systems, Volume 62, 2014, Pages 22-31, ISSN 0167-9236, https://doi.org/10.1016/j.dss.2014.03.001.