Mini Project 1 : Banking Campaign Prediction

Harry Setiawan Hamjaya

2302000

***Abstract*— This project aims to predict the outcome of a banking campaign through direct marketing efforts using two machine learning algorithms: Logistic Regression and Random Forest Classifier. The dataset encompasses diverse attributes concerning bank clients, previous campaign interactions, and economic indicators. Preprocessing involved handling missing values, encoding categorical variables, and scaling numerical features. Models were trained with an 80-20 train-test split, targeting a minimum of 80% accuracy. Logistic Regression achieved 89.6% accuracy and 64.2% F1, while Random Forest Classifier reached 89.2% accuracy and 65.1% F1. This study underscores the viability of employing machine learning to forecast banking campaign outcomes, emphasizing the significance of algorithm selection in achieving accurate predictions.**

Keywords: Logistic Regression, Random Forest Classifier, Bank Clients, Campaign Interaction, and Economic Indicators.

I. Introduction

The efficacy of banking marketing campaigns hinges on accurately predicting client responses to direct marketing efforts, particularly regarding subscription to term deposits. In an environment where resources are limited, ensuring the optimal allocation of resources becomes imperative for campaign success. However, identifying the subset of clients most likely to respond positively to the campaign remains a challenging task. This problem underscores the need for sophisticated predictive models that can leverage available data on client demographics, previous campaign interactions, and broader economic indicators to make informed predictions about client behavior. By addressing this challenge, organizations can refine their targeting strategies, minimize resource wastage, and enhance the overall effectiveness of their marketing campaigns.

To tackle this problem, this study adopts a machine learning approach, leveraging algorithms such as Logistic Regression and Random Forest Classifier. These algorithms offer the capability to analyze complex datasets comprising diverse client attributes and historical campaign outcomes. Through comprehensive data preprocessing and model training, we aim to develop accurate predictive models capable of discerning patterns and relationships within the data. By harnessing the predictive power of these algorithms, organizations can gain insights into the factors driving client behavior, thereby enabling more targeted and effective marketing strategies.

The outcomes of this study hold significant implications for the banking industry and broader marketing practices. By harnessing the predictive capabilities of machine learning algorithms, organizations can optimize their marketing campaigns, improving client engagement and ultimately driving business growth. Furthermore, the methodologies and insights derived from this study can be extrapolated to other domains, providing a blueprint for leveraging data-driven approaches to enhance decision-making and strategy formulation across various industries.

II. Exploratory Data Analysis

A. Label

In the provided data, the imbalance in label distribution refers to the disproportionate representation of the two classes: "0" representing clients who did not subscribe to a term deposit (NO), and "1" representing clients who did subscribe (Yes). Specifically, there are 36,548 instances of class "0" and only 4,640 instances of class "1" as described in figure 1. This disparity suggests that the dataset is heavily skewed towards the negative outcome (clients not subscribing to a term deposit), making it an imbalanced dataset.

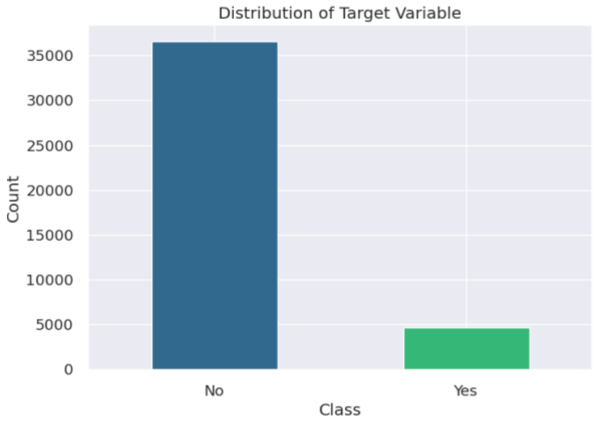


Figure 1: The distribution of Class label

Imbalanced data poses challenges for machine learning algorithms because they tend to be biased towards the majority class. In this case, the model may become overly focused on predicting the majority class (NO subscriptions), leading to poor performance in predicting the minority class (YES subscriptions). As a result, the model's ability to accurately identify clients likely to subscribe to a term deposit may be compromised.

B. Feature

In the dataset provided, there are two main types of features: numerical and categorical.

Numerical Features: Numerical features represent quantitative data that can be measured or counted. In this dataset, numerical features include attributes such as 'age', 'duration' (the duration of the last contact in seconds), 'campaign' (the number of contacts made during the current campaign), 'pdays' (the number of days since the client was last contacted from a previous campaign), 'previous' (the number of contacts made before the current campaign), 'emp.var.rate' (employment variation rate), 'cons.price.idx' (consumer price index), 'cons.conf.idx' (consumer confidence index), 'euribor3m' (euribor 3 month rate), and 'nr.employed' (the number of employees). These features provide quantitative insights into client characteristics, previous campaign interactions, and economic indicators. In the notebook, it has been plotted for each categorical feature distribution, for example Age and Duration distribution in figure 2 and 3 respectively.

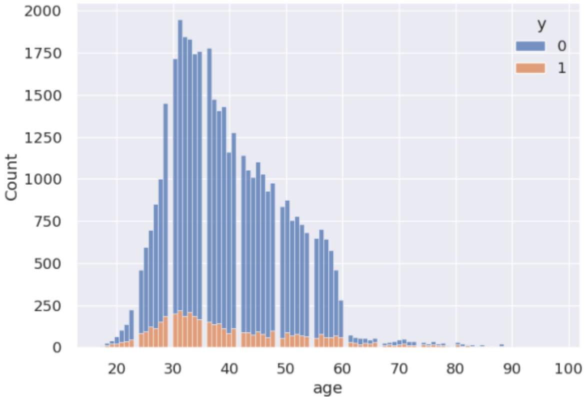


Figure 2: The distribution of age feature

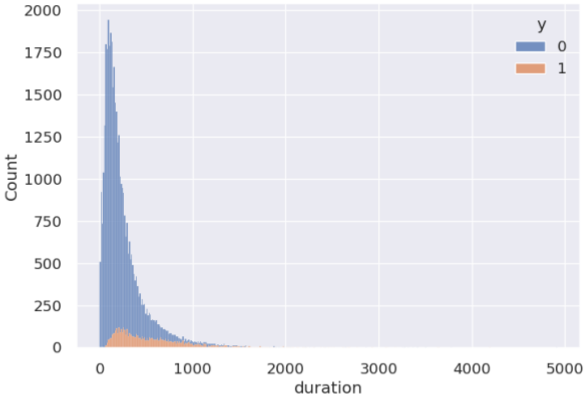


Figure 3: The distribution of duration feature

Categorical Features: Categorical features represent qualitative data that can take on a limited number of distinct categories or levels. In this dataset, categorical features include attributes such as 'job' (type of job), 'marital' (marital status), 'education', 'default' (whether the client has credit in default), 'housing' (whether the client has a housing loan), 'loan' (whether the client has a personal loan), 'contact' (communication type), 'month' (the month of the last contact), 'day_of_week' (the day of the week of the last contact), and 'poutcome' (outcome of the previous marketing campaign). These features provide qualitative information about client demographics, contact preferences, and previous campaign outcomes. In the notebook, it has been plotted for each categorical feature distribution, for example Job and Education distribution in figure 4 and 5 respectively.

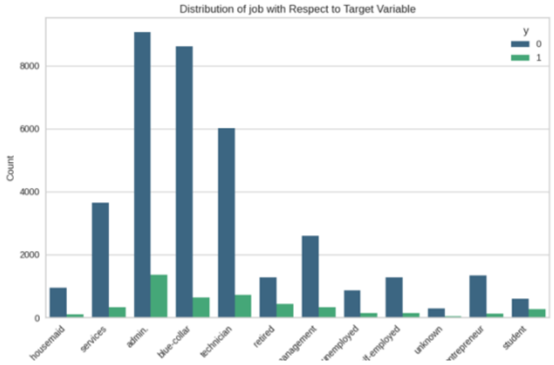


Figure 4: The distribution of job feature

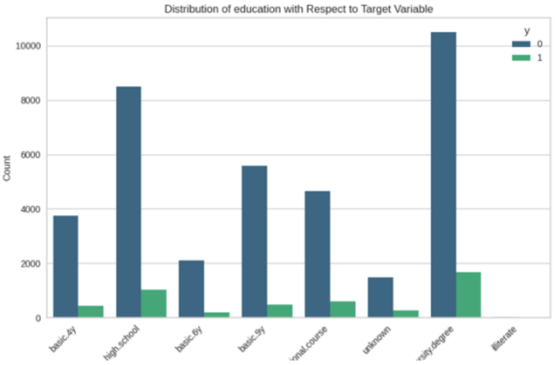


Figure 5: The distribution of education feature

III. DATA PROCESSING

A. Categorical Feature Encoding

Weight of Evidence (WOE) encoding plays a crucial role in predictive modeling, particularly in tasks like binary classification, as seen in this banking campaign prediction project. WOE encoding transforms categorical variables into a numeric format that effectively captures the relationship between the categorical variable and the target variable whether a client subscribed to a term deposit or not. By assigning numeric values based on the likelihood of each category resulting in the target event, WOE encoding provides a clear representation of the predictive power of each category. This enhances the interpretability of the encoded features while ensuring compatibility with machine learning algorithms that require numerical input.

Moreover, WOE encoding is important for mitigating the risks associated with sparse occurrences in categorical variables and overfitting. It achieves this by effectively handling categories with limited representation and reducing the dimensionality of the feature space. Ultimately, WOE encoding enhances the predictive performance and robustness of the models by capturing the underlying relationship between categorical variables and the target variable, thereby facilitating more accurate predictions in binary classification tasks.

For example, in the job category, every “admin” would be replace by “0.16052481239193747” numerical value instead of the category string “admin” value based on the likelihood of job category resulting in the target event, which is label Yes/No toward banking campaign

Table 1: WOE encoding on job categorical feature.

job	
'housemaid'	0.13331247843800917
'services'	0.3598215331177416
'admin’	0.16052481239193747
'blue-collar'	0.5391258365932243
'technician'	0.044722437998921664
'retired'	0.9776647282010504
'management'	0.004801370541626522
employed'	0.080487339056398
'unknown'	0.005342597474632559
'entrepreneur'	0.3102431865970874
'student'	1.2837535413486352

B. *Feature-Label Correlation*

There are several lists of features that must be created by our self, especially since we can’t use all the given feature data that was given. Two features that must be dropped:

Feature-label correlation refers to the relationship between individual features (independent variables) and the target label (dependent variable) in a dataset. It measures the degree of association or dependency between each feature and the target label, indicating how much influence a feature has on predicting the target variable. Understanding feature-label correlation is crucial in predictive modeling as it helps identify which features are most relevant or influential in predicting the target variable.

In binary classification tasks like the banking campaign prediction project, feature-label correlation typically involves assessing how each feature correlates with the binary outcome variable (e.g., whether a client subscribed to a term deposit or not). This correlation can be quantified using various statistical measures such as Pearson correlation coefficient for numerical features and point-biserial correlation coefficient for categorical features.

High feature-label correlation suggests that a particular feature has a strong association with the target label and is likely to be informative for prediction. Conversely, low correlation indicates that the feature may not contribute significantly to predicting the target variable. Identifying features with high correlation allows for better feature selection, where only the most relevant features are retained for model training, thereby improving model performance and interpretability.

Feature-label correlation analysis also helps uncover insights into the underlying relationship between features and the target variable, providing valuable insights into the factors driving the outcome of interest. Additionally, it can guide feature engineering efforts, where new features are created, or existing features are transformed to enhance their predictive power and improve model performance.

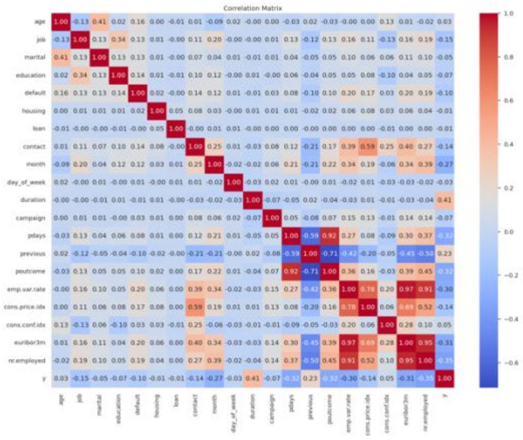


Figure 6: The correlation heatmap

In figure 6, we can see the correlation matrix of every feature and label with the heatmap format. We can list up the feature with the highest feature-label correlation by measuring the absolute value, and the result as table 2.

Table 2: Sorted highly correlated feature toward label.

Correlation	
duration	0.4052738
nr.employed	0.3546783
pdays	0.32491448
poutcome	0.31728483
euribor3m	0.3077714
emp.var.rate	0.29833443
month	0.26606421
previous	0.230181
job	0.14865814
contact	0.14477306
cons.price.idx	0.13621121
duration	0.4052738
nr.employed	0.3546783
pdays	0.32491448
poutcome	0.31728483
euribor3m	0.3077714
emp.var.rate	0.29833443
month	0.26606421
previous	0.230181
job	0.14865814

C. Standarization

Standardization, also known as z-score normalization, is a preprocessing technique used to rescale numerical features in a dataset to have a mean of 0 and a standard deviation of 1. This process ensures that all numerical features are on the same scale, which is particularly important for algorithms that are sensitive to the scale of the input features, such as gradient descent-based algorithms, support vector machines, and k-nearest neighbors.

In summary, while standardization is crucial for Logistic Regression to ensure faster convergence and improve coefficient interpretability, it can also benefit Random Forest by improving stability, performance, and the meaningfulness of feature splits. Therefore, applying standardization to input features is generally considered a good practice when working with both Logistic Regression and Random Forest algorithms.

D. Oversampling

Oversampling methods, including SMOTE (Synthetic Minority Over-sampling Technique), are techniques used to address class imbalance in datasets. Class imbalance occurs when one class (usually the minority class) is underrepresented compared to the other class(es) in a classification problem.

SMOTE is a popular oversampling technique that works by generating synthetic samples from the minority class. It creates new, synthetic instances by interpolating between existing minority class instances. This is achieved by selecting a sample from the minority class and finding its k nearest neighbors. New instances are then created by randomly selecting points along the line segments connecting the selected sample to its neighbors.

SMOTE has several advantages:

- It addresses class imbalance by creating synthetic samples, which helps in training more robust classifiers.
- It reduces the risk of overfitting that might occur when using simpler oversampling techniques (like duplication or replication).
- It introduces diversity in the dataset, reducing the risk of overfitting by creating slightly different samples rather than exact duplicates.

However, SMOTE also has limitations:

- It may generate noisy samples if the feature space is highly complex or noisy.
- SMOTE does not consider the underlying distribution of the data, which may result in generating unrealistic samples.

Overall, SMOTE is a powerful oversampling technique that can effectively address class imbalance in classification tasks, but it's essential to tune the parameters appropriately and evaluate the impact on model performance carefully.

IV. Modelling

In this project, we chose two machine learning algorithms for modeling: Logistic Regression and Random Forest Classifier.

Logistic Regression was selected for its simplicity, interpretability, and effectiveness in binary classification tasks. It assumes a linear relationship between the independent variables and the log-odds of the target variable. We trained the Logistic Regression model using the default parameters provided by the scikit-learn library.

Random Forest Classifier was chosen for its ability to handle non-linear relationships and complex datasets. It is an ensemble learning method that constructs multiple decision trees during training and combines their predictions through voting. To optimize the performance of the Random Forest Classifier, we performed hyperparameter tuning using grid search with cross-validation. This involved searching through different combinations of hyperparameters such as the number of trees (n_estimators), maximum depth of trees (max_depth), and minimum number of samples required to split a node (min_samples_split).

To improve the performance of our models, we employed several strategies:

1. Data Preprocessing: We conducted thorough data preprocessing steps, including handling missing values, encoding categorical variables, and scaling numerical features. This ensured that the data was in a suitable format for model training and reduced the impact of outliers.

2. Feature Engineering: We experimented with feature engineering techniques such as creating new features or transforming existing ones to capture additional information and improve model performance.

3. Hyperparameter Tuning: For Random Forest Classifier, we performed hyperparameter tuning using grid search with cross-validation to find the optimal combination of hyperparameters that maximized model performance.

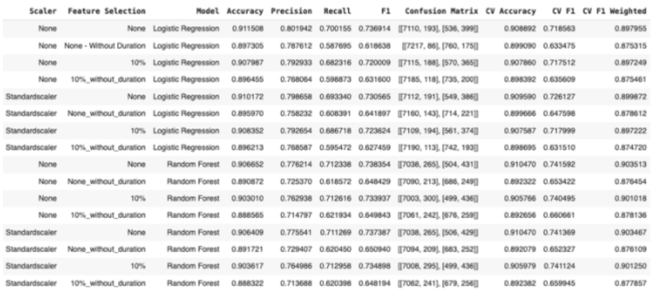
4. Evaluation Metrics: We used appropriate evaluation metrics such as accuracy, precision, recall, and F1-score to assess the performance of our models comprehensively. This allowed us to identify areas for improvement and fine-tune our models accordingly.

While we aimed to reach the required accuracy in the first training round, it is common to iterate on model development, fine-tuning hyperparameters, and refining preprocessing steps to achieve the desired performance. But in this case, we managed to get the 80% accuracy threshold because the data is highly imbalanced toward the “Yes”. It is uncommon, but the main reason is because the Evaluation metric criteria that is challenged shouldn’t be Accuracy but Recall or F1 which can show the model robustness toward the imbalanced dataset.

V. Training and Evaluation

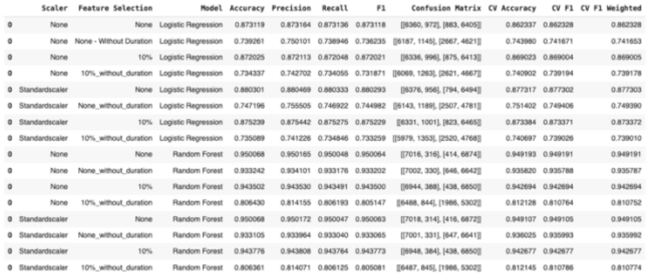
Every batch of training would produce the evaluation table such as the Table 3 as the Model Evaluation Metrics when training is done without oversampling and Table 4 as the Model Evaluation Metrics when training is done with the oversampling. There are several scenarios that we have tested, from the feature selection usage (only the feature with 10% or more that correlated to label), duration feature usage (as it is from the specification, that it shouldn’t be included), the application of standardization and the usage of tow different models, which in our case it is Logistic Regression and Random Forest Classifier.

Table 3: Model Evaluation Metrics without Oversampling



Scaler	Feature Selection	Model	Accuracy	Precision	Recall	F1	Confusion Matrix	CV Accuracy	CV F1	CV F1 Weighted
None	None	Logistic Regression	0.911508	0.801942	0.700155	0.736914	[[7110, 193], [536, 399]]	0.908892	0.718563	0.897955
None	None - Without Duration	Logistic Regression	0.897305	0.787612	0.587695	0.618638	[[7217, 86], [760, 175]]	0.899090	0.633475	0.875315
None	10%	Logistic Regression	0.907987	0.792933	0.682316	0.720009	[[7115, 188], [570, 365]]	0.907860	0.717512	0.897249
None	10%_without_duration	Logistic Regression	0.896455	0.768064	0.598873	0.631600	[[7185, 118], [735, 200]]	0.898392	0.635609	0.875461
Standardscaler	None	Logistic Regression	0.910172	0.798658	0.693340	0.730565	[[7112, 191], [549, 386]]	0.909590	0.726127	0.899872
Standardscaler	None_without_duration	Logistic Regression	0.895970	0.758232	0.608391	0.641897	[[7160, 143], [714, 221]]	0.899666	0.647598	0.878612
Standardscaler	10%	Logistic Regression	0.908352	0.792654	0.686718	0.723624	[[7109, 194], [561, 374]]	0.907587	0.717999	0.897222
Standardscaler	10%_without_duration	Logistic Regression	0.896213	0.768587	0.595472	0.627459	[[7190, 113], [742, 193]]	0.898695	0.631510	0.874720
None	None	Random Forest	0.906652	0.776214	0.712338	0.738354	[[7038, 265], [504, 431]]	0.910470	0.741592	0.903513
None	None_without_duration	Random Forest	0.890872	0.725370	0.618572	0.648429	[[7090, 213], [686, 249]]	0.892322	0.653422	0.876454
None	10%	Random Forest	0.903010	0.762938	0.712616	0.733937	[[7003, 300], [499, 436]]	0.905766	0.740495	0.901018
None	10%_without_duration	Random Forest	0.888565	0.714797	0.621934	0.649843	[[7061, 242], [676, 259]]	0.892656	0.660661	0.878136
Standardscaler	None	Random Forest	0.906409	0.775541	0.711269	0.737387	[[7038, 265], [506, 429]]	0.910470	0.741369	0.903467
Standardscaler	None_without_duration	Random Forest	0.891721	0.729407	0.620450	0.650940	[[7094, 209], [683, 252]]	0.892079	0.652327	0.876109
Standardscaler	10%	Random Forest	0.903617	0.764986	0.712958	0.734898	[[7008, 295], [499, 436]]	0.905979	0.741124	0.901250
Standardscaler	10%_without_duration	Random Forest	0.888322	0.713688	0.620398	0.648194	[[7062, 241], [679, 256]]	0.892382	0.659945	0.877857

Table 4: Model Evaluation Metrics with Oversampling



	Scaler	Feature Selection	Model	Accuracy	Precision	Recall	F1	Confusion Matrix	CV Accuracy	CV F1	CV F1 Weighted
0	None	None	Logistic Regression	0.873119	0.873164	0.873136	0.873118	[[6360, 972], [883, 6405]]	0.862337	0.862328	0.862328
0	None	None - Without Duration	Logistic Regression	0.739261	0.750101	0.738946	0.736235	[[6187, 1145], [2667, 4621]]	0.743980	0.741671	0.741653
0	None	10%	Logistic Regression	0.872025	0.872113	0.872048	0.872021	[[6336, 996], [875, 6413]]	0.869023	0.869004	0.869005
0	None	10%_without_duration	Logistic Regression	0.734337	0.742702	0.734055	0.731871	[[6069, 1263], [2621, 4667]]	0.740902	0.739194	0.739178
0	Standardscaler	None	Logistic Regression	0.880301	0.880469	0.880333	0.880293	[[6376, 956], [794, 6494]]	0.877317	0.877302	0.877303
0	Standardscaler	None_without_duration	Logistic Regression	0.747196	0.755505	0.746922	0.744982	[[6143, 1189], [2507, 4781]]	0.751402	0.749406	0.749390
0	Standardscaler	10%	Logistic Regression	0.875239	0.875442	0.875275	0.875229	[[6331, 1001], [823, 6465]]	0.873384	0.873371	0.873372
0	Standardscaler	10%_without_duration	Logistic Regression	0.735089	0.741226	0.734846	0.733259	[[5979, 1353], [2520, 4768]]	0.740697	0.739026	0.739010
0	None	None	Random Forest	0.950068	0.950165	0.950048	0.950064	[[7016, 316], [414, 6874]]	0.949193	0.949191	0.949191
0	None	None_without_duration	Random Forest	0.933242	0.934101	0.933176	0.933202	[[7002, 330], [646, 6642]]	0.935820	0.935788	0.935787
0	None	10%	Random Forest	0.943502	0.943530	0.943491	0.943500	[[6944, 388], [438, 6850]]	0.942694	0.942694	0.942694
0	None	10%_without_duration	Random Forest	0.806430	0.814155	0.806193	0.805147	[[6488, 844], [1986, 5302]]	0.812128	0.810764	0.810752
0	Standardscaler	None	Random Forest	0.950068	0.950172	0.950047	0.950063	[[7018, 314], [416, 6872]]	0.949107	0.949105	0.949105
0	Standardscaler	None_without_duration	Random Forest	0.933105	0.933964	0.933040	0.933065	[[7001, 331], [647, 6641]]	0.936025	0.935993	0.935992
0	Standardscaler	10%	Random Forest	0.943776	0.943808	0.943764	0.943773	[[6948, 384], [438, 6850]]	0.942677	0.942677	0.942677
0	Standardscaler	10%_without_duration	Random Forest	0.806361	0.814071	0.806125	0.805081	[[6487, 845], [1986, 5302]]	0.812145	0.810786	0.810774

Based on table 3 as we can see the accuracy between Random Forest and Logistic Regression didn’t differ more than 5% which means it is not significant compared to the table 4, as we can see the table 4 shows that the random forest give more than 90% accuracy, while logistic regression have the average of 87% while using duration feature, while having 73% when duration feature is removed. From this point of view, we can note that random forest shows that it is more robust compared to the logistic regression, especially when we didn’t do the hyperparameter tuning.

From the perspective of training data and cross-validation as shown in figure 7, it appears that feature selection did not significantly improve the performance of both Logistic Regression and Random Forest models. This is evidenced by the range of results not varying much compared to models without feature selection. Similarly, standardization using StandardScaler did not have a significant impact on model performance, as indicated by the cross-validation accuracy comparison chart. However, the inclusion of the 'duration' feature notably affected the model's recall and F1 score as shown in figure 8, although it did not cause a significant change in accuracy. This discrepancy is apparent when comparing the 'duration' feature with other indicators such as feature selection and standardization.

Moving on to the evaluation of the test data, Random Forest with all features without resampling achieved the highest F1-score of approximately 73.8%, with an accuracy of around 90.6%. However, considering that the 'duration' feature cannot be used in a realistic predictive model, Random Forest with StandardScaler yielded the best result, with a 65.1% F1 score and 89.2% accuracy. Conversely, Logistic Regression performed best with all features, achieving an F1-score of about 73.7% and an accuracy of 91.2%. Nevertheless, similar to Random Forest, Logistic Regression with StandardScaler performed optimally without the 'duration' feature, yielding a 64.2% F1 score and 89.6% accuracy. Additionally, the Cross Validation Average Boxplot Comparison indicates that Random Forest outperforms Logistic Regression, particularly in terms of F1 score and predictive capability towards the "Yes" label.

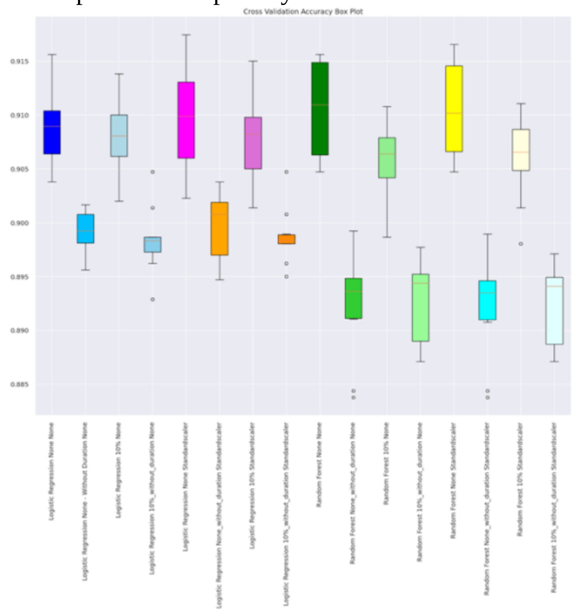


Figure 7: Accuracy cross validation plot

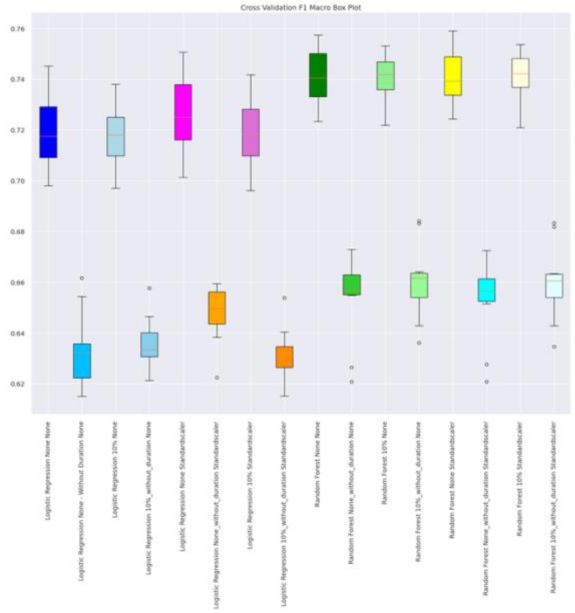


Figure 8: F1 cross validation plot

The selection of important features by each model, Logistic Regression and Random Forest, sheds light on the different aspects of the dataset that influence the prediction outcome.

In Logistic Regression, the employment variation rate and consumer price index emerge as significant features as shown in figure 9. The employment variation rate is a quarterly indicator reflecting changes in the employment market. It is a critical economic indicator that directly impacts consumer behavior and their likelihood to subscribe to a term deposit. Similarly, the consumer price index, a monthly indicator, provides insights into the overall inflationary pressures in the economy. Fluctuations in consumer prices can affect consumer confidence and purchasing power, thereby influencing their decision to subscribe to financial products such as term deposits. Hence, Logistic Regression prioritizes these economic indicators, recognizing their direct impact on consumer behavior and subscription outcomes.

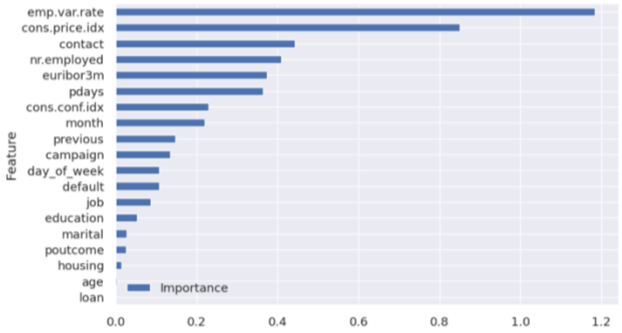


Figure 9: Logistic Regression feature importance

Conversely, Random Forest identifies age and euribor 3-month rate as the most important features as shown in figure 10. Age, being a demographic variable, often captures life stage-related factors such as financial stability, investment goals, and risk tolerance. Younger individuals may have different financial priorities and risk perceptions compared to older age groups, affecting their propensity to subscribe to term deposits. On the other hand, the euribor 3-month rate, a daily indicator, represents the prevailing interest rates in the Eurozone. Fluctuations in interest rates can influence consumer savings and investment decisions, making it a crucial factor in predicting subscription outcomes. Therefore, Random Forest prioritizes demographic factors such as age alongside macroeconomic indicators like interest rates, recognizing their collective impact on subscription behavior.

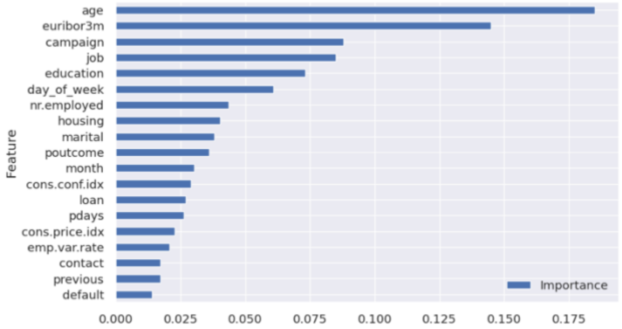


Figure 10: Random Forest feature importance

In summary, the selection of employment variation rate and consumer price index by Logistic Regression emphasizes the significance of economic indicators, while the inclusion of age and euribor 3-month rate by Random Forest underscores the importance of demographic and macroeconomic factors in predicting subscription outcomes. These features encapsulate diverse aspects of consumer behavior and economic conditions, providing valuable insights into the drivers of subscription decisions in the banking sector.

VI. Conclusion

In conclusion, several scientific bottlenecks were encountered during the course of this project, primarily cantered around feature selection, model performance optimization, and algorithm selection. One of the main challenges was determining the most influential features for predicting subscription outcomes accurately. Despite attempts to use feature selection techniques, such as cross-validation and analysis of feature importance, it was observed that the selected features did not consistently improve model performance across different algorithms.

The real bottleneck of the task is the task didn’t describe the evaluation method properly, especially since the dataset is imbalanced. Achieving the high accuracy without considering the imbalanced value is inappropriate since it might mislead the decision. For this context, resampling is needed either over sampling and or under sampling to check if the model is able to detect the balanced dataset instead of imbalanced one.

To overcome these bottlenecks, various strategies were employed. Firstly, we conducted thorough data pre-processing to ensure the quality and consistency of the dataset. This involved handling missing values, encoding categorical variables, and scaling numerical features. Additionally, we experimented with different feature engineering techniques to create new features and transform existing ones, aiming to capture additional information and improve model performance.

In terms of algorithm selection, both Logistic Regression and Random Forest were evaluated extensively. While Logistic Regression offers simplicity and interpretability, Random Forest excels in capturing complex relationships in the data. Through iterative experimentation and fine-tuning, we aimed to identify the algorithm that best suited the characteristics of the dataset and provided the most accurate predictions.

Ultimately, the choice between the two algorithms depends on the specific requirements and priorities of the task at hand. In this project, Random Forest with standardization and all dataset without duration (since the specification asked for it) demonstrated superior performance in terms of F1 score and predictive capability towards the "Yes" label compared to Logistic Regression. Moreover, random forest model offers the more stable and robust model as it performs better toward the oversampling data, although it is not chosen as the best model since the limitation of SMOTE algorithm in generating data which might be outlier because it is out the real data distribution

In summary, despite encountering scientific bottlenecks throughout the project, thorough experimentation, data pre-processing, and model optimization techniques allowed us to overcome these challenges and develop predictive models capable of accurately forecasting subscription outcomes in banking campaigns. While Random Forest emerged as the better-performing algorithm in this particular scenario, the selection of the most suitable algorithm ultimately depends on the specific requirements and constraints of the problem domain