Bank Marketing Data Set

Project proposal 5

1st Artur Almeida 123196
Department of electronics, telecommunications and Informatics
University of Aveiro
Aprendizagem Automática
Instructor: Petia Georgieva
arturalmeida@ua.pt

2nd Rafael Morgado 104277
Department of electronics telecommunications and Informatics
University of Aveiro
Aprendizagem Automática
Instructor: Petia Georgieva
rafa.morgado@ua.pt

***Abstract*—This project aims to predict whether a client will subscribe to a term deposit using data from a Portuguese bank's marketing campaigns. We implemented a complete machine learning pipeline with data preprocessing, visualization, model training, and evaluation. Three classifiers—Logistic Regression, SVM, and Neural Networks—were tested and compared using metrics like accuracy, precision, recall, and ROC curves. The final goal is to derive insights into client behavior and optimize marketing efforts through predictive modeling.**

***Index Terms*—neural networks, logistic regression, support vector machine (SVM)**

I. Introduction

In the banking industry, customer acquisition and retention are key challenges. Direct marketing campaigns, such as phone calls, are commonly used to promote products like term deposits. However, contacting a large number of clients without knowing their potential interest can be inefficient and costly. Predictive modeling offers a solution by helping institutions target clients who are more likely to respond positively.

This project is based on the Bank Marketing Dataset, which contains data from marketing campaigns conducted by a Portuguese banking institution. The dataset includes a variety of client attributes, such as age, job, marital status, education, as well as campaign-specific variables like contact method, duration, and outcome. The classification objective is to predict whether a client will subscribe to a term deposit (”yes” or ”no”).

To achieve this, we implemented a full machine learning pipeline that includes data cleaning, transformation, visualization, model training, hyperparameter tuning, and evaluation. Three supervised models were compared: Logistic Regression, Support Vector Machines (SVM), and Multi-layer Perceptron Neural Networks. This study not only evaluates model performance but also highlights the importance of data preprocessing and systematic hyperparameter optimization in building effective classification systems.

II. Data Description and Preprocessing

A. Dataset Overview

The dataset used in this project was obtained from Kaggle and is based on direct phone call marketing campaigns conducted by a Portuguese banking institution. It contains a total of **11,162 observations** and **17 input features**, along with **1 binary target variable** called `deposit`, which indicates whether the client subscribed to a term deposit (`yes` or `no`).

The features are grouped into the following categories:

- **Client-related attributes:**
  - `age`, `job`, `marital`, `education`, `default`, `housing`, `loan`
- **Campaign-related attributes:**
  - `contact`, `month`, `day`, `duration`, `campaign`, `pdays`, `previous`, `poutcome`
- **Target variable:**
  - `deposit` (binary classification: `yes` or `no`)

B. Feature Description

In the table I we describe each variable included in the dataset along with their data types. This structured understanding of the features is essential for proper data preprocessing and model interpretation.

C. Motivation and Problem Definition

Nowadays, banks often run direct marketing campaigns to promote products like term deposits. However, calling a large number of clients without knowing if they are actually interested can be very time-consuming and inefficient. That's why we found this problem interesting, it's a real-world situation

TABLE I
DATASET VARIABLE DESCRIPTION

Variable	Type	Description
`age`	Integer	Age of the client.
`job`	Categorical	Type of job (e.g., admin., technician, student, etc.).
`marital`	Categorical	Marital status (e.g., married, single, divorced).
`education`	Categorical	Level of education (e.g., primary, secondary, tertiary or unknown).
`default`	Binary	Has credit in default? (yes/no).
`balance`	Integer	Average yearly balance (in euros).
`housing`	Binary	Has a housing loan? (yes/no).
`loan`	Binary	Has a personal loan? (yes/no).
`contact`	Categorical	Type of communication (cellular, telephone or unknown).
`day of week`	Date	Day of the week of the last contact.
`month`	Date	Month of the last contact.
`duration`	Integer	Duration of the last contact (in seconds). **Note:** this variable will be excluded from training as it leaks target information.
`campaign`	Integer	Number of contacts performed during this campaign.
`pdays`	Integer	Days since the client was last contacted. Value -1 means the client was not previously contacted.
`previous`	Integer	Number of contacts before this campaign.
`poutcome`	Categorical	Outcome of the previous marketing campaign (success, failure, other or unknown).
`deposit`	Binary (Target)	Has the client subscribed to a term deposit? (yes/no).

where machine learning can help make better decisions and save resources.

With this project, we wanted to use data from past campaigns to try and predict which clients are more likely to say ’yes’ to a term deposit. This can help banks focus their efforts on the right people and improve their success rate.

D. Machine Learning Problem Type

The problem we’ve been working on is a **binary classification task**. Basically, the goal is to predict whether a client will subscribe to a term deposit or not.

- **Inputs:** The dataset has 17 features that describe things like the client’s age, job, marital status, and details about how and when they were contacted by the bank.
- **Output:** A binary value (`yes` or `no`) in the `deposit` column, which shows if the client subscribed or not.

E. Problem Complexity

Despite the binary nature of the task, the problem is moderately complex due to a combination of factors: 17 original features, that will be expanded to 33 after encoding, categorical variables with many levels, class overlap, and subtle correlations. These elements require careful preprocessing and model selection to ensure generalization and robustness.

F. Data Visualization

To better understand the data before training any models, we created several visualizations.

- First, we made a pie chart to check the distribution of the target variable `deposit`. As expected, there is a slight imbalance, with around 52.6% of clients not subscribing and 47.4% subscribing.

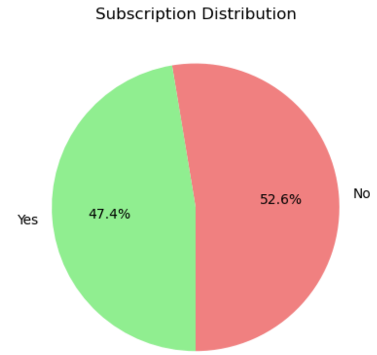

Fig. 1. Distribution of the `deposit` variable

- Then, we used bar graphs to analyze how the subscription rate varies across different categories. For example, when looking at the `job` attribute, we can see that clients in ’management’, ’retired’, and ’student’ roles tend to have a higher subscription rate.

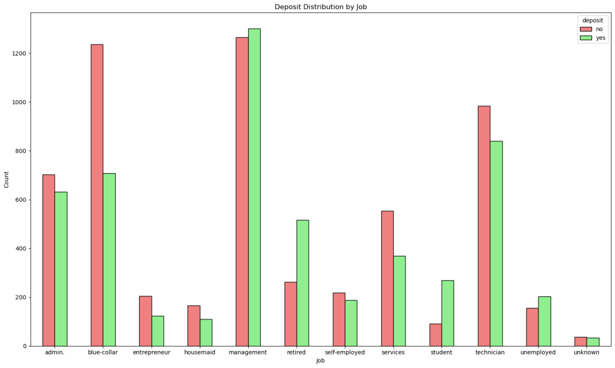

Fig. 2. Deposit distribution by job category

- We also plotted grouped bar charts for other categorical and binary variables like `marital`, `education`, `default`, `housing`, `loan`, `contact`, `month`, and `poutcome`. These helped identify which categories had the most successful conversions.

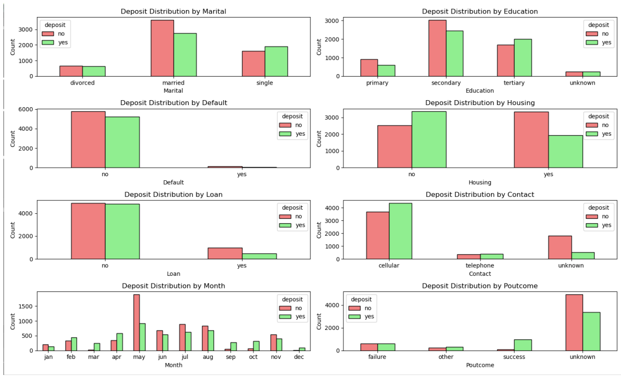

Fig. 3. Deposit distribution across other categorical and binary variables

- For the numerical features, we used histograms to look at the distribution of values. Most of them were unbalanced, especially `balance`, `duration`, and `pdays`, with a large number of zeros or small values.

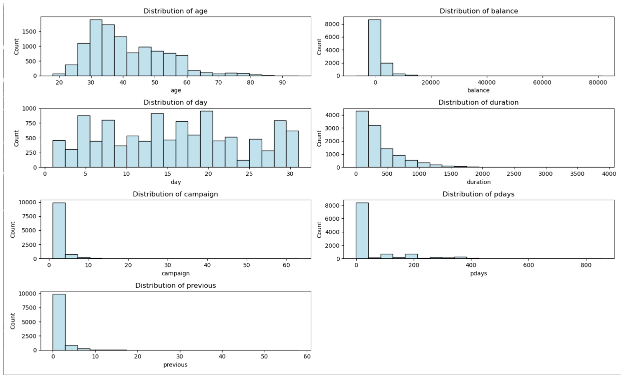

Fig. 4. Histograms of numerical features

- We also created boxplots to compare each numeric feature between the classes `yes` and `no`, and also the total distribution. This made it easier to see the spread of the data and some clear outliers, especially in `balance`, `duration` and `pdays`. Removing this outlier would result in a loss of **4605 entries** and significantly make the data unbalanced (37% `yes` and 63% `no`), which would negatively impact the model training. So, the outliers were retained to preserve data balance.

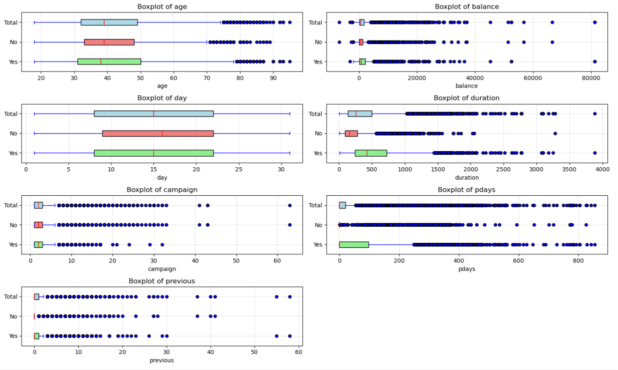

Fig. 5. Boxplots of numerical features by target class

- Finally, we plotted a correlation heatmap for the numerical variables. There were no very strong correlations, but some relationships like `previous` with `pdays`, or `duration` with `deposit`, stood out a bit more.

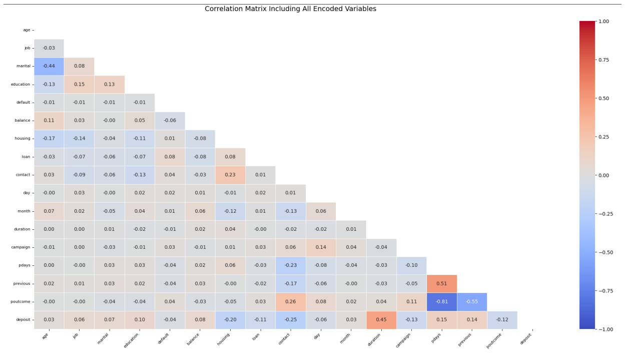

Fig. 6. Correlation heatmap of numerical variables

G. Data Preprocessing

Before training the classification models we have to preprocess the raw dataset to ensure that all features were properly encoded and suitable for machine learning algorithms.

First, the `duration` variable was removed from the dataset. Although it is the most correlated feature with the target variable, it is not known before the call is performed and thus model couldn't use this data to predict what would be the choice of the client.

The binary features like `deposit` (the target), `loan`, `default`, and `housing` were changed into numeric, using simple mapping: `yes` to 1 and `no` to 0.

The `education` feature, which has an ordinal nature, was mapped to values representing educational levels: `primary` to 0, `secondary` to 1, `tertiary` to 2, and `unknown` to -1.

The `month` features were changed into numeric, with month names mapped to their calendar values (e.g., `jan` to 1, `feb` to 2, etc).

The remaining nominal categorical variables, `job`, `marital`, `contact` and `poutcome` were converted into dummy variables using one-hot encoding, allowing the model to handle categorical information without assuming any ordinal relationship.

After the data preprocessing we will end up with 33 total features.

III. Model Training and Evaluation

A. Data Splitting

To start, the dataset was split into training and test sets using a stratified split, preserving the original class distribution of the target variable (`deposit`) in both sets to reduce the risk of biased model performance due to uneven class representation. The split became as follows:

- **Training set:** 80% (8,929 samples).
- **Test set:** 20% (2,233 samples).

B. Feature Standardization

After splitting the data, the features were Standardization using the `StandardScaler`, to transform the data to have zero mean and unit variance. This step is crucial for the training as the algorithms are sensitive to feature scales. The scaler was fitted on the training data to prevent an information leak from the test set.

C. Model Selection and Hyperparameter Tuning

We compared and trained 3 different machine learning:

1) **Logistic Regression (LR):** A basic baseline model for binary classification.
2) **Support Vector Machine (SVM):** A kernel-based classifier with flexibility in handling non-linear decision boundaries.
3) **Neural Network (NN):** A flexible model composed of layers of neurons that learns complex, non-linear patterns in data.

To optimize the hyperparameter for SVM and NN to find the ones with best accuracy, we did the following:

- **SVM Optimization:** A grid search testing different `C`, kernel types (`linear`, `rbf`) and gamma (`scale`, `auto`).
- **NN Optimization:** A grid search testing different combinations of hidden layers and neurons, lambda and learning rates.

The grid search used a 5 K-Fold cross-validation to ensure robust evaluation.

D. Model Performance

The model performances were evaluated with:

- **Classification report:** Includes precision, recall, F1-score, and support for each class.
- **Precision-Recall curve:** This curve shows the detection probability vs. recall for different detection thresholds.
- **Confusion matrix:** Number of True Positives (TP), False Positives, False Negatives and True Negatives (TN).
- **ROC curve:** Area under the Receiver Operating Characteristic (ROC) curve, measuring class separability, TN and TP. Having a bigger area the better it is at distinguishing classes.
- **Loss curve:** This metric was only used for the NN model and shows how the loss changes while training.

E. Results

Before showing the results that we got with the different models, is important to establish what we consider the best model for this dataset. A high accuracy desirable but the most important metric should be maximize the number of True Positives since missing actual positive cases (False Negatives) can lead to a loss of a client and having more False Positives just leads to a small loss of time in most cases.

Logistic Regression:

As shown in Table II, Logistic Regression achieved an overall accuracy of **68.65%**, but the recall for the positive class was only **61.25%**.

TABLE II
LR Classification Report

Class	Precision	Recall	F1-Score	Support
0 (No)	0.6834	0.7532	0.7166	1175
1 (Yes)	0.6908	0.6125	0.6493	1058
Accuracy			**0.6865**	2233
Macro avg	0.6871	0.6828	0.6829	2233
Weighted avg	0.6869	0.6865	0.6847	2233

The Precision-Recall Curve in Figure 7 presents an average precision of **0.74**, reflecting a good balance between precision and recall across thresholds. The Confusion Matrix (Figure 8) which shows that 410 of the positive class were predicted as negatives. The ROC Curve in Figure 9 shows an AUC of **0.74**, indicating fair class separability.

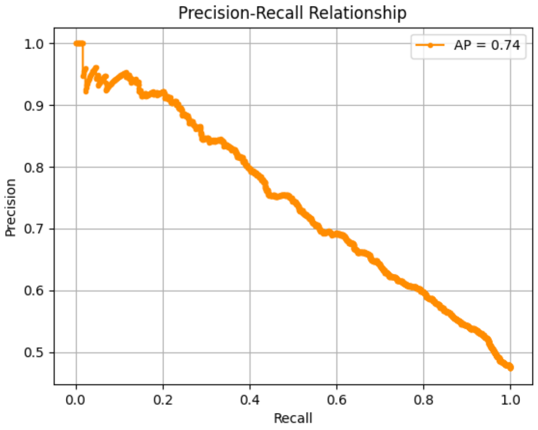

Fig. 7. LR Precision-Recall Curve

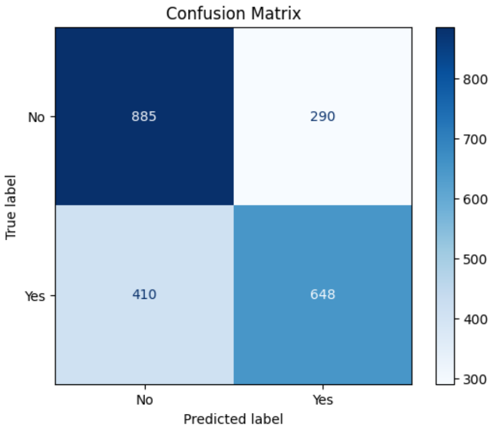

Fig. 8. LR Confusion Matrix

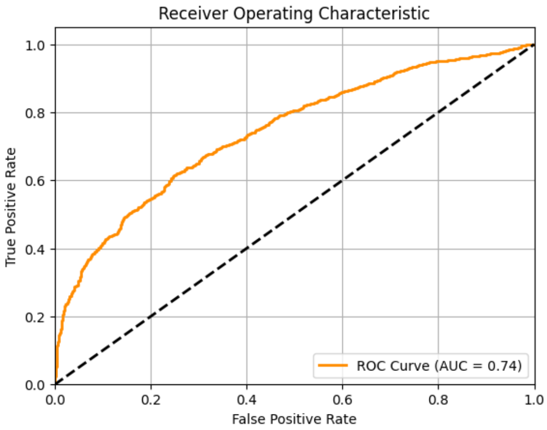

Fig. 9. LR ROC Curve

Support Vector Machine:

Using the best parameters obtained via grid search (`C = 1`, `kernel = rbf`, `gamma = scale`), the SVM model achieved slightly better accuracy at cost of positive recall as shown in Table III. the overall accuracy was **69.86%**, with a recall of **60.49%** for the positive class.

TABLE III
SVM CLASSIFICATION REPORT

Class	Precision	Recall	F1-Score	Support
0 (No)	0.6876	0.7830	0.7322	1175
1 (Yes)	0.7151	0.6049	0.6554	1058
Accuracy			**0.6986**	2233
Macro avg	0.7013	0.6939	0.6938	2233
Weighted avg	0.7006	0.6986	0.6958	2233

The Precision-Recall Curve in Figure 10 is slightly less smooth than LR's, likely due to tied decision scores or fewer confident positive predictions. The confusion matrix (Figure 11) shows that the model correctly predicted 640 out of 1058 positive cases which is less than the LR model. The ROC Curve (Figure 12) presents an AUC of **0.75**, slightly better than LR.

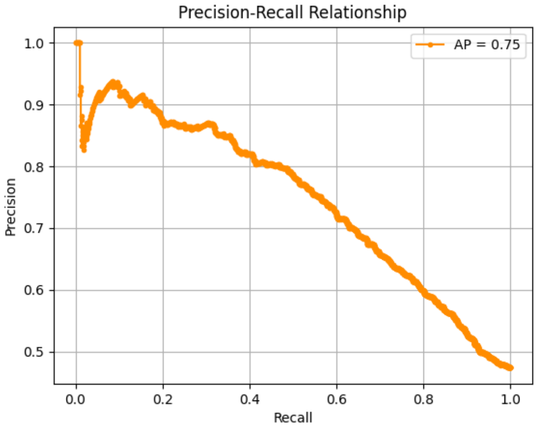

Fig. 10. SVM Precision-Recall Curve

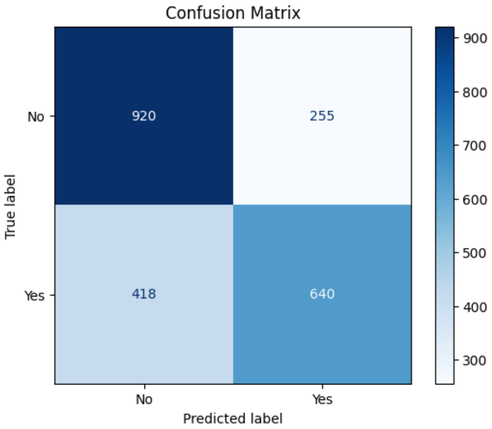

Fig. 11. SVM Confusion Matrix

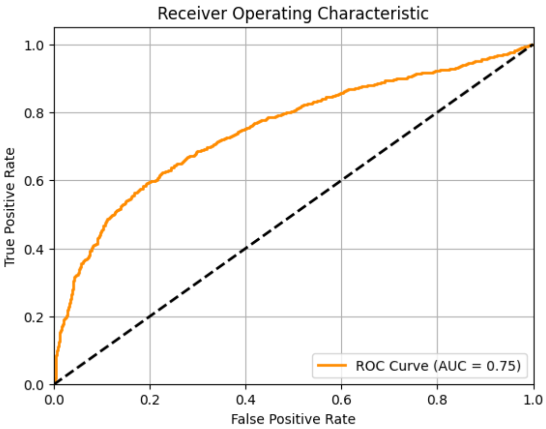

Fig. 12. SVM ROC Curve

Neural Networks:

Using the best parameters obtained via grid search (`Nodes = (10, 10, 20)`, `learning rate = 0.01`, `lambda = 0.01`), the Neural Network yielded the best results among all three models. Table IV shows an accuracy of **70.85%** and the highest recall for the positive class (**62.95%**).

TABLE IV
NN Classification Report

Class	Precision	Recall	F1-Score	Support
0 (No)	0.7003	0.7796	0.7378	1175
1 (Yes)	0.7200	0.6295	0.6717	1058
Accuracy			**0.7085**	2233
Macro avg	0.7102	0.7045	0.7048	2233
Weighted avg	0.7096	0.7085	0.7065	2233

The training loss (Figure 13) shows good convergence. The Precision-Recall curve, (figure 14) with the highest value of the 3, of **0.76**. The Confusion matrix, (figure 15) confirms that this is the model with less False Negatives. The ROC Curve (Figure 16) achieved the highest AUC of **0.77**, reinforcing the model's superior discriminative ability.

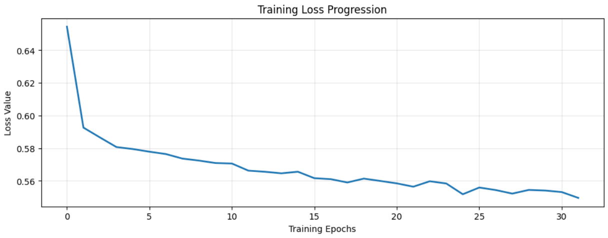

Fig. 13. NN Loss

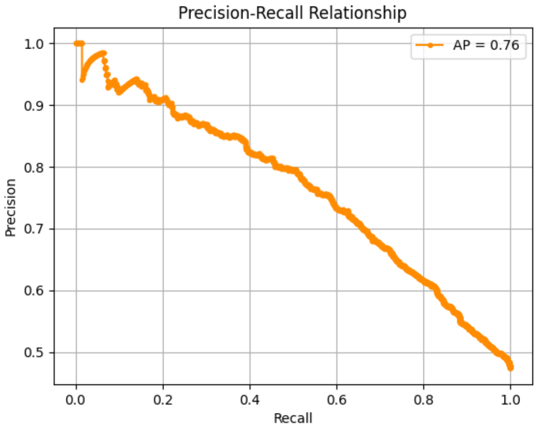

Fig. 14. NN Precision-Recall Curve

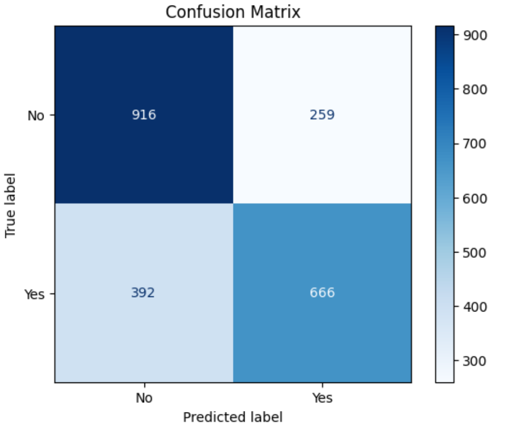

Fig. 15. NN Confusion Matrix

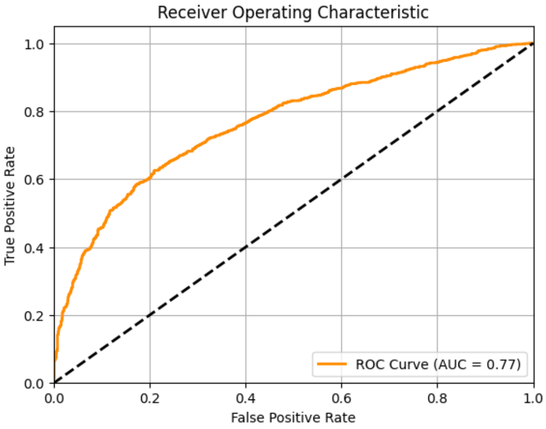

Fig. 16. NN ROC Curve

F. Model Comparison

In the table V we can see the most important metrics of the models. The Neural Network was the most effective model overall, providing the higher probability of correctly guessing if a person will subscribe for a deposit. The SVM was the worst of the 3, getting the least TP of the models.

TABLE V
Model Performance Comparison

Model	Accuracy	Precision (1)	Recall (1)	ROC-AUC
Logistic Regression	68.65%	69.08%	61.25%	0.74
SVM	69.86%	71.51%	60.49%	0.75
Neural Network	**70.85%**	**72%**	**62.95%**	**0.77**

G. Other tests

During the project we tried to manipulate the data in different ways to get better results but they were not successful, for example, we tried removing the features that had a correlation less than 0.1 with deposit, it made the models less significantly less accurate and thus we discarded that option. Also in the beginning of the project we were using the duration feature and the models were getting accuracies around 80% which confirms the correlation of this feature.

H. Comparison with other works

On Kaggle, where we obtained this dataset, it is possible to find the work of other users. After obtaining our results, we decided to compare them with those.

To our surprise, most of the notebooks we found on Kaggle did not exclude the `duration` feature from the dataset. This variable has a data leakage problem, and including it leads to inflated model accuracy and for that reason, their results are not comparable to ours.

However, in this notebook by the user `Murat Mert`, the `duration` feature was removed, just as we did. Unfortunately, the only model we had in common was Logistic Regression. His model performed slightly better than ours, with an accuracy of 70.71% compared to our 68.65%. This difference is likely due to hyperparameter tuning, which we did not apply to our Logistic Regression model. However, the difference is not significant.

His best-performing model was the XGBoost Classifier, which achieved an accuracy of 72.50%, outperforming our best model (Neural Network, 70.85%).

We chose to compare only with Kaggle implementations because the dataset on Kaggle has been modified from the original version from the UCI repository, which we only discovered later.

IV. Conclusion

This project addressed a real-world classification issue in the banking industry: predicting if a customer would accept a term deposit or not based on characteristics of the person and campaign. By applying a machine learning pipeline — data preprocessing, visualization, feature encoding, model training, evaluation — we contrasted three models: Logistic Regression, Support Vector Machine, and Neural Networks.

The results revealed that while all models were satisfactory, the Neural Network performed the best on all of the most critical measures. It achieved the highest accuracy (70.85%), positive class precision (72%), positive class precision (62.95%) , and ROC-AUC (0.77) and was the best at correctly identifying clients who will likely subscribe. This would align with the business objective of optimizing true positives since missing potential subscribers would translate to lost business opportunities.

Logistic Regression provided a good and understandable baseline, while SVM, although slightly more precise, was worse at detecting true positives. Additional experiments confirmed the importance of careful feature selection and the justification for removing the `duration` variable to avoid target leakage.

In conclusion, machine learning is an efficient instrument to optimize the direct marketing in banking, and the approach described in this project can be implemented or improved for forthcoming campaigns or other areas of interest.

Disclaimer: This report was assisted by AI-based tools such as ChatGPT and DeepSeek. They were used to improve code formatting, ease of generating graph plots and refine the English writing.