Project Protocol

Bank Marketing

Almog Cohen

Introduction:

In today's competitive banking sector, understanding customer behavior is crucial for optimizing marketing efforts. This project analyzes data from direct marketing campaigns conducted by a Portuguese banking institution. These campaigns relied on phone calls to promote term deposit subscriptions, often requiring multiple contacts with potential customers.

The dataset, sourced from the UCI Machine Learning Repository via Kaggle, contains information on customer demographics, previous interactions, and the outcomes of past campaigns. The goal of this project is to develop a predictive model that determines whether a customer will subscribe to a term deposit based on these features. By leveraging machine learning, we aim to help banks improve their marketing strategies, increase efficiency, and better target potential clients.

Objectives:

In this project, my goal is to develop a model to determine whether a customer will subscribe to a term deposit based on various demographic, financial, and campaign-related features. To achieve this, I will address the following key questions:

- **What am I trying to find out?**
  I want to identify the characteristics and behaviors that influence a customer's decision to subscribe to a term deposit.
- **What do I already know?**
  I have access to data from previous marketing campaigns, including customer demographics, economic conditions, and past interactions. I also know that some features, such as duration, can significantly impact predictions but should be excluded for a realistic model.
- **What am I aiming to achieve?**
  My goal is to build an accurate and interpretable machine learning model that helps optimize marketing strategies by predicting which customers are most likely to subscribe.
- **What factors affect my results?**
  Several factors influence my predictions, including customer profiles (e.g., age, job, marital status), past campaign outcomes (poutcome, previous), economic indicators (euribor3m, emp.var.rate), and the number of previous contacts (pdays, campaign). My model’s performance will also depend on data quality, feature selection, and the choice of machine learning algorithms.

Data Preparation:

Before training any models, I first examined the dataset to understand its structure and ensure it was ready for analysis. The dataset was already well-organized, requiring minimal preprocessing. Below are the key steps I took during this phase:

Feature Overview

To better understand the dataset, I created a summary table outlining each feature, its datatype, and its role in the analysis:

Col	Explenation	Data Type	Numeric/Categorial?	Cluster
age	age	int	Numeric	**Bank client data**
job	job type	string	Categorial	
marital	status	string	Categorial	
education	education type	string	Categorial	
default	has credit in default?	string	categorial	
housing	has housing loan?	string	categorial	
loan	has personal loan?	string	categorial	
contact	contact communication type	string	categorial	**Related with the last contact of the current campaign**
month	last contact month of year	string	categorial	
day_of_week	last contact day of the week	string	categorial	
duration	last contact duration, in seconds	int	Numeric	
campaign	number of contacts performed during this campaign and for this client	int	Numeric	**Other attributes**
pdays	number of days that passed by after the client was last contacted from a previous campaign	int	Numeric	
previous	number of contacts performed before this campaign and for this client	int	Numeric	
poutcome	outcome of the previous marketing campaign	string	categorial	
emp.var.rate	employment variation rate - quarterly indicator	float	Numeric	**Social and economic context attributes**
cons.price.idx	consumer price index - monthly indicator	float	Numeric	
cons.conf.idx	consumer confidence index - monthly indicator	float	Numeric	
euribor3m	euribor 3 month rate - daily indicator	float	Numeric	
nr.employed	number of employees - quarterly indicator	int	Numeric	
y	has the client subscribed a term deposit?	bool	categorial	**TARGET**

Converting Object-Type Features to Their Appropriate Data Types

The dataset contained **11 categorical features** stored as object types (job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome, and y). Since these represent categorical data rather than free text, I converted them to the category datatype to optimize memory usage and improve performance during model training. Additionally, I encoded the target variable (y) as a binary category (0 for "no" and 1 for "yes").

Ensuring Data Integrity

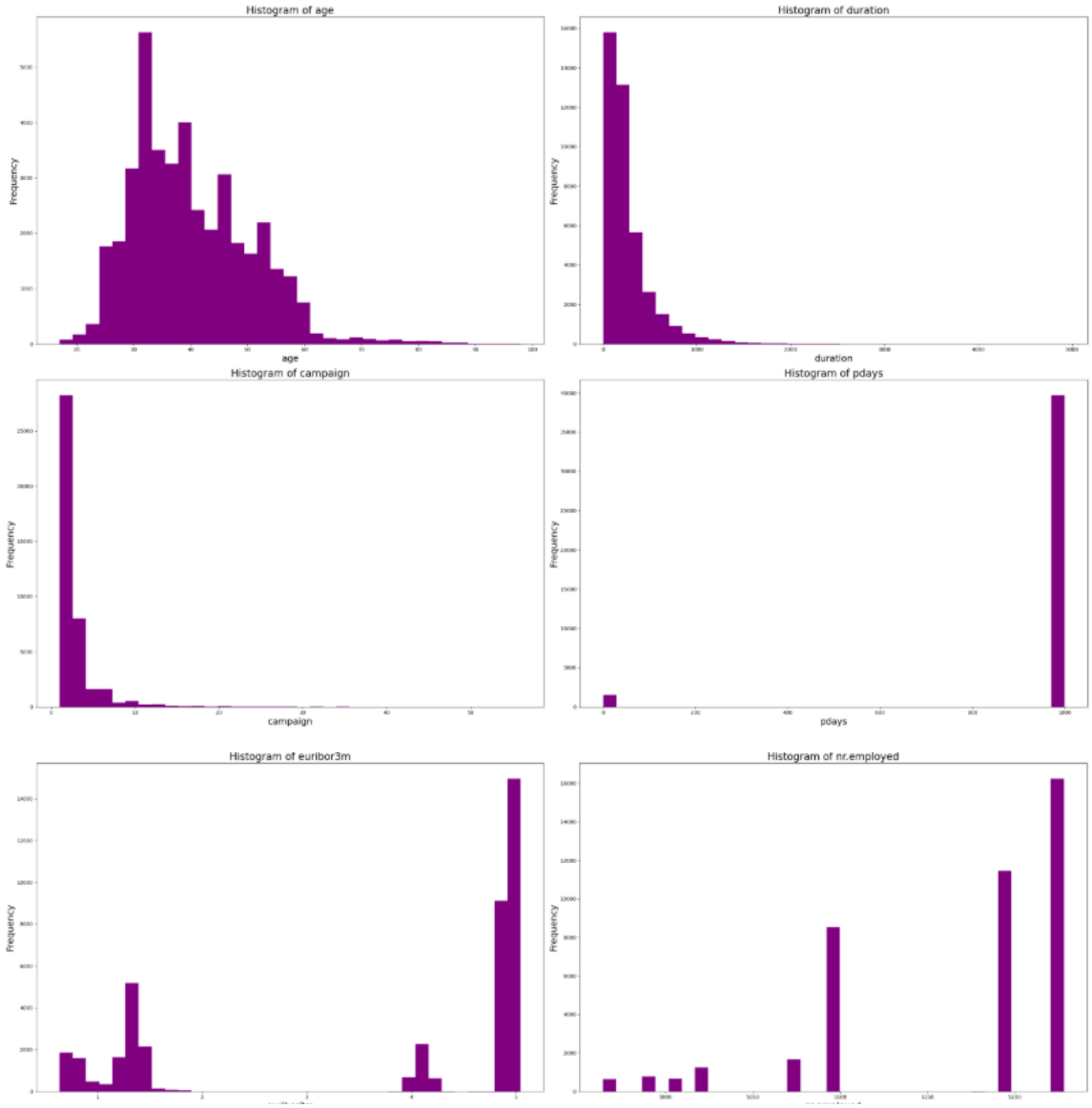
The dataset did not contain any missing values or inconsistencies, so no further cleaning was necessary.

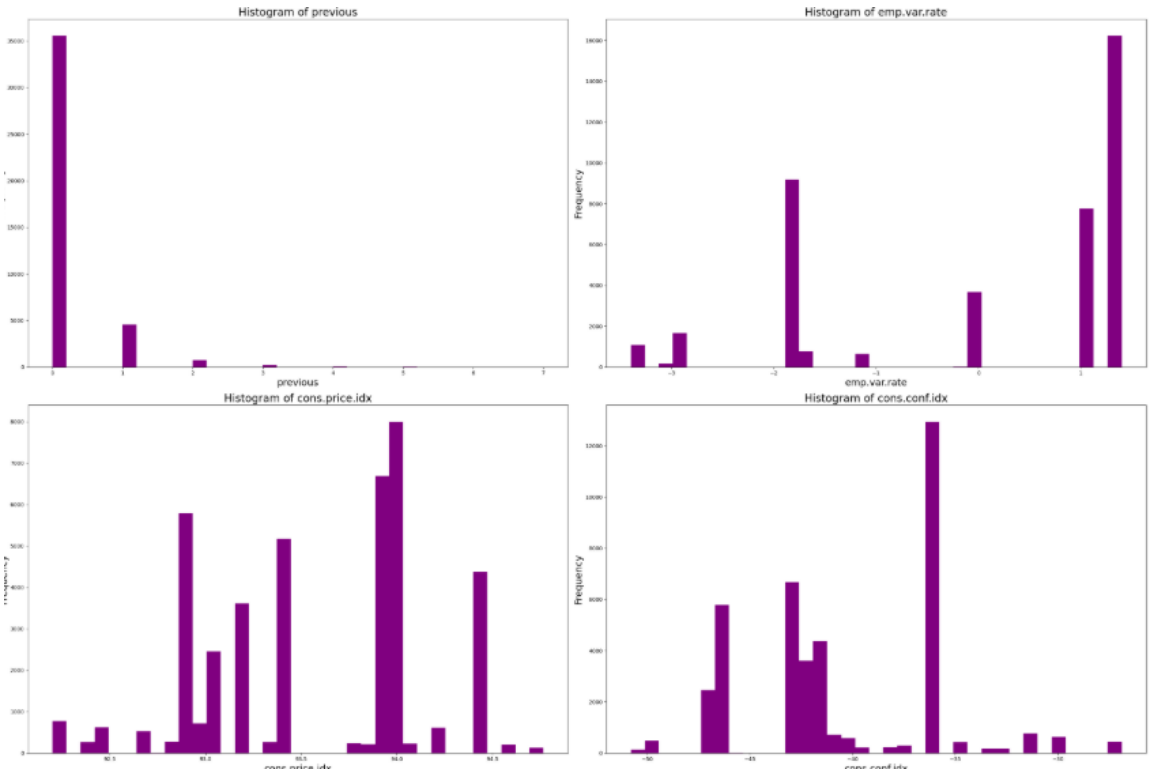
EDA:

As part of the initial exploration, I visualized the dataset's features by categorizing them into three types: numeric, categorical, and dummy variables.

Numeric Features:

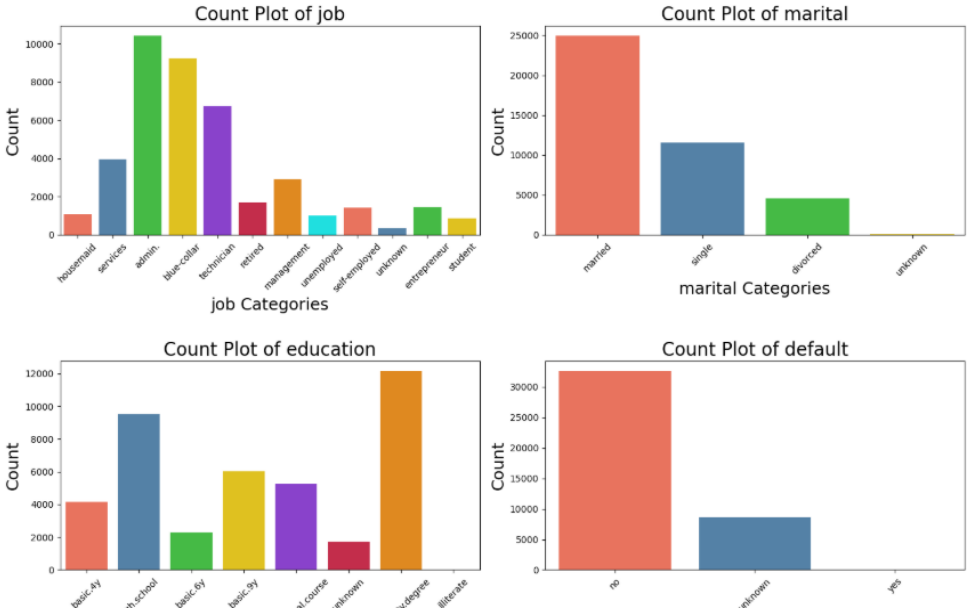
I started by examining the numeric features by visualizing their distribution as hisograms, such as **age**, **duration of the last call**, **campaign contacts**, and several **economic indicators** like the **consumer price index** and **employment rate**:

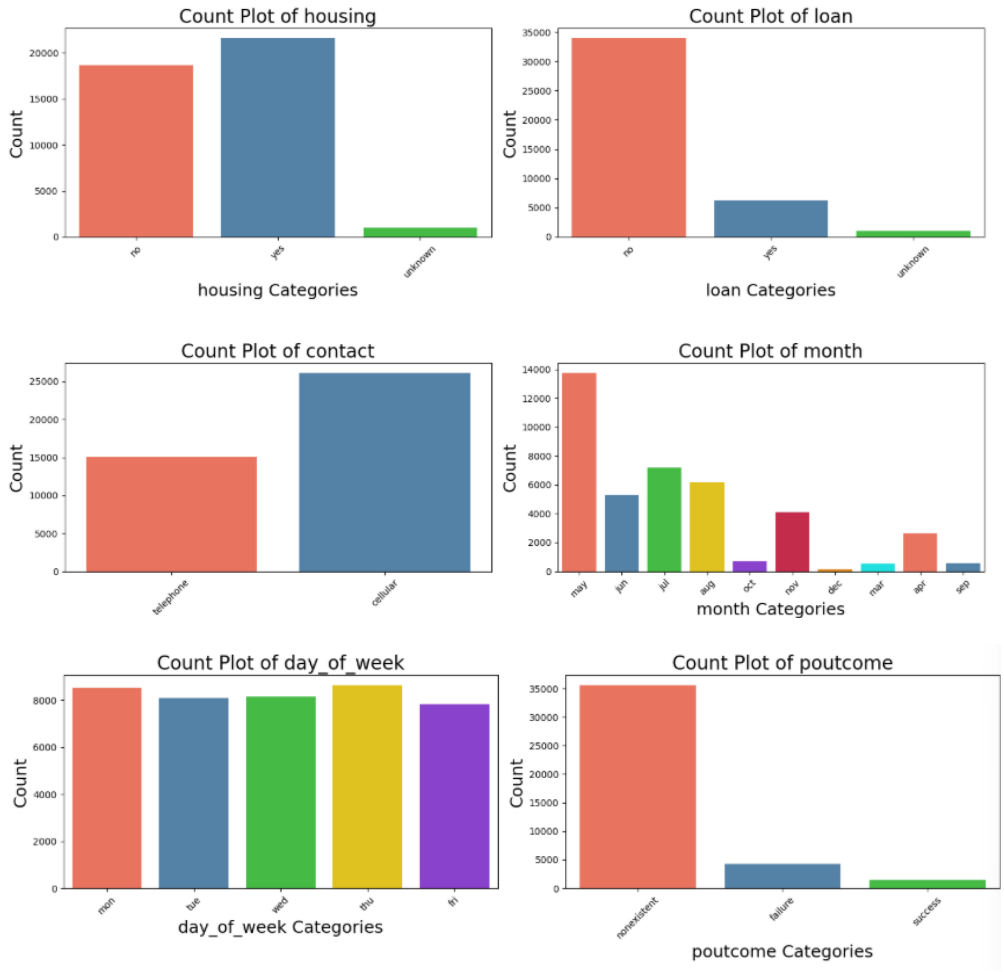




Categorical Features

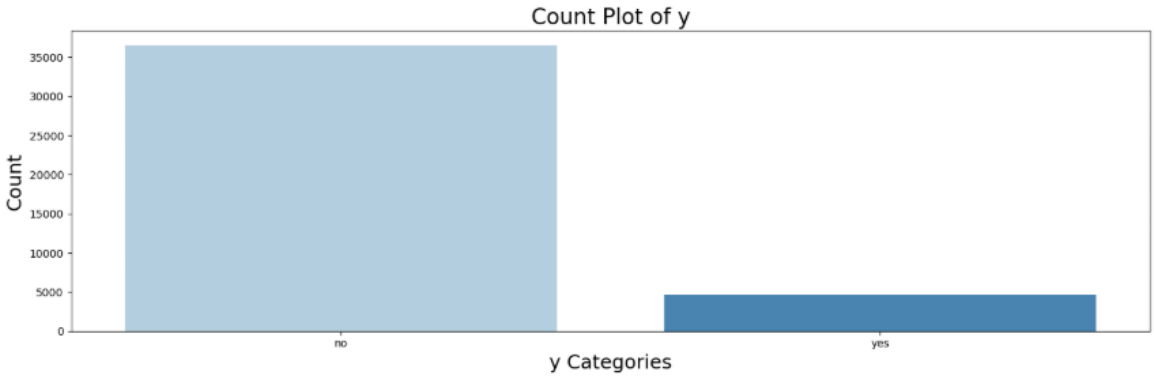
Next, I explored the **categorical features** in the dataset, which include variables such as **job**, **marital status**, **education**, and **contact method**. To visualize these features, I used **count plots** to see the distribution of different categories within each feature:





Dummy Feature ('y')

The final step in the data visualization was to examine the **dummy feature**, represented by the **'y'** variable. This boolean feature indicates whether a customer subscribed to a term deposit ('yes') or not ('no'). As the outcome variable in this dataset, it directly reflects the success or failure of the marketing campaign.



Skewness Analysis:

To understand the distribution of the numerical features in the dataset, I first examined their **skewness**. Skewness indicates the asymmetry of the data distribution, and by evaluating this, I can decide whether the data is approximately normal or requires transformation.

After running the skewness test, I found that most of the numerical features exhibit significant skewness. Specifically, **8 features** were **highly or moderately skewed**, while only **2 features** were approximately normal. This suggests that many of the variables deviate from a normal distribution, which can affect certain statistical techniques that assume normality.

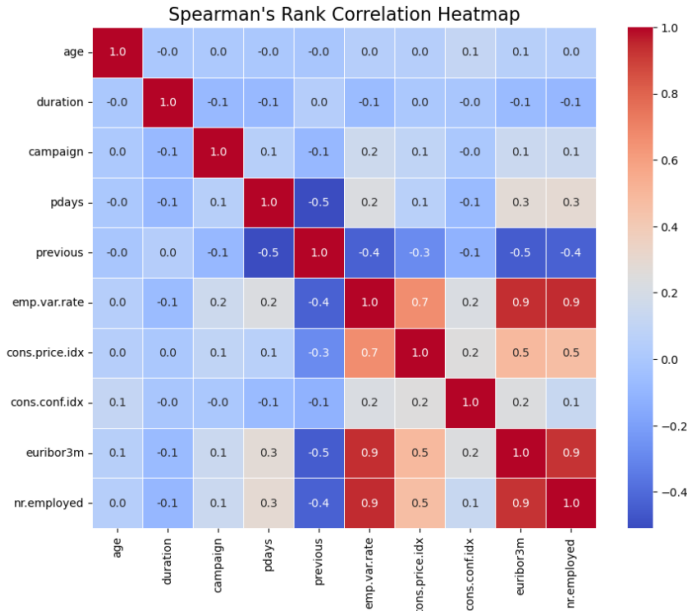
Here's a summary of the skewness results:

- **Moderately skewed**: age, emp.var.rate, euribor3m
- **Highly skewed**: duration, campaign, pdays, previous, nr.employed
- **Approximately normal**: cons.price.idx, cons.conf.idx

Given the findings, we will use the **Spearman correlation** method, which is suitable for non-normally distributed data, to evaluate the relationships between the numerical features.

Spearman Correlation Analysis:

To further understand the relationships between the numerical features, I conducted a **Spearman's Rank Correlation** test. Spearman's correlation is particularly useful for identifying monotonic relationships between variables, without assuming a linear relationship or normal distribution of the data.



The results from the **Spearman correlation matrix** were visualized in a heatmap for better clarity. From the analysis, I observed the following key correlations:

- **Very Strong Positive Correlations (≥ 0.9)**:
    - euribor3m (3-month Euribor rate) and emp.var.rate (employment variation rate)
    - nr.employed (number of employees) and emp.var.rate (employment variation rate)
    - nr.employed (number of employees) and euribor3m (3-month Euribor rate)
- **Strong Positive Correlation (≥ 0.7)**:
    - cons.price.idx (consumer price index) and emp.var.rate (employment variation rate)
- **Moderate Positive Correlations (≥ 0.5)**:
    - euribor3m (3-month Euribor rate) and cons.price.idx (consumer price index)
    - nr.employed (number of employees) and cons.price.idx (consumer price index)

These strong and moderate positive correlations suggest that certain economic factors, like the Euribor rate and employment variation rate, are strongly linked with the number of employees and consumer price index. Understanding these relationships is crucial for building a predictive model, as they could help inform important variables for classification.

Chi-Square Test (Categorical Features):

To evaluate the relationships between the categorical features, I conducted a **Chi-Square Test of Independence**. This test assesses whether two categorical variables are independent or if there is a significant association between them. A significant result indicates a relationship between the variables, while a non-significant result suggests independence.

The following key findings emerged from the test:

- **Strong Relationships (p-value = 0.0000)**:
    - Most pairs of categorical features show a **p-value of 0.0000**, suggesting that these variables are likely dependent and strongly related to each other. These relationships include various combinations of attributes like job, marital status, and previous campaign outcomes.
- **Non-Significant Relationships (p-value > 0.05)**:
    - **Job and Housing Loan**: No strong evidence that a person's job affects whether they have a housing loan.

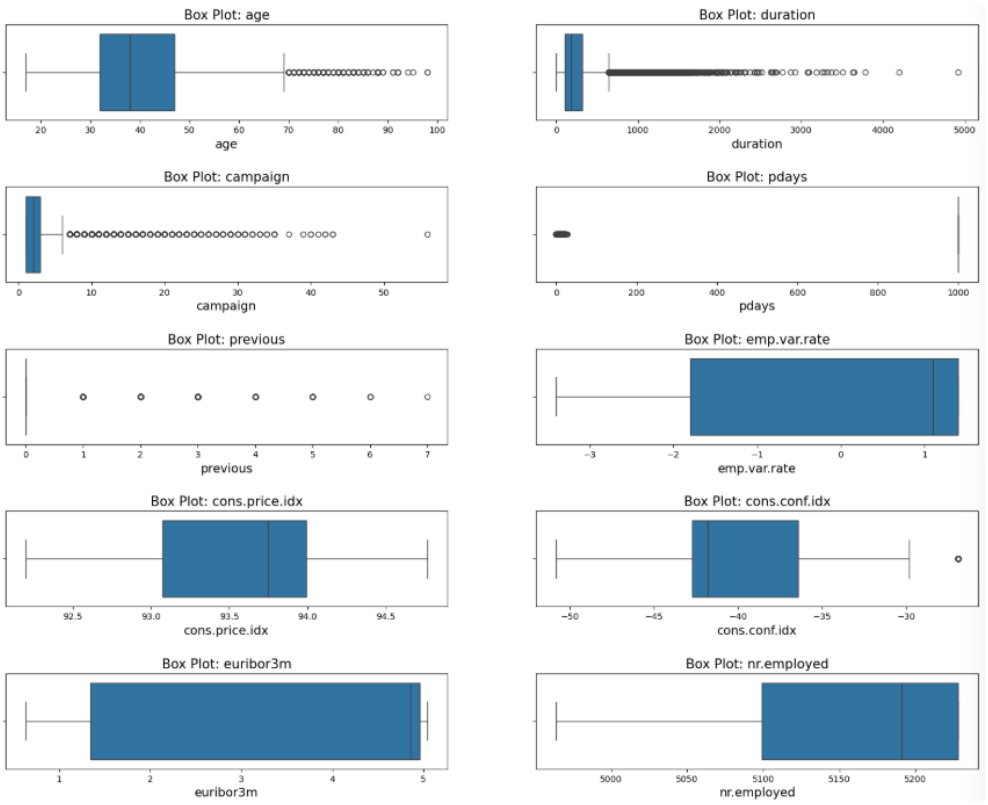
- **Job and Personal Loan**: No strong evidence that a person's job affects whether they have a personal loan.
- **Marital Status and Loan**: No significant relationship between marital status and having a loan.
- **Education Level and Loan**: No significant link between education level and having a loan.
- **Default Status and Personal Loan**: No evidence that having credit in default affects whether a person has a personal loan.
- **Loan Status and Day of Week**: No evidence that loan status correlates with the day of the week the contact was made.
- **Loan Status and Outcome of Previous Campaign**: No significant relationship between having a loan and the outcome of the previous marketing campaign.
- **Personal Loan and Subscription**: Having a personal loan does not appear to impact whether the person subscribes to the term deposit.
- **Housing Loan Status**: Close to significant (p-value just above 0.05), suggesting weak evidence of a relationship.

While many categorical features in the dataset exhibit strong relationships with each other, **loan-related variables** appear mostly independent. These insights provide valuable context for model development, indicating which features might be useful predictors and which may not contribute significantly to the target variable.

Outlier Detection and Treatment:

In this step, I focused on identifying potential outliers in the dataset. Outliers can sometimes distort statistical analyses, but it's essential to determine whether they meaningfully affect the distribution or relationship with the target variable (y).

Visualization:
To begin, I visualized all numeric features using **boxplots** to spot any obvious outliers. This helped in quickly identifying the spread and potential extreme values in each numeric feature.



Next, I implemented the **Interquartile Range (IQR)** method to systematically detect outliers. The IQR method identifies values that fall outside 1.5 times the interquartile range above the third quartile or below the first quartile. I then created a table to evaluate the impact of these outliers:

	feature	outliers_cnt	distribution_changed	mean_diff	drop
0	age	469	+	1.001961	no
1	duration	2963	+	332.346357	no
2	campaign	2406	+	-0.581361	no
3	pdays	1515	+	-192.078317	no
4	previous	5625	+	0.360299	no
5	emp.var.rate	0	-	-1.482324	no
6	cons.price.idx	0	-	-0.249371	no
7	cons.conf.idx	447	+	0.803312	no
8	euribor3m	0	-	-1.688356	no
9	nr.employed	0	-	-80.935674	no

Key Findings:
Upon reviewing the results, I found that the outliers in this dataset did **not** significantly affect the distribution of features or the relationship with the target variable. Specifically:

- **Distribution**: The outliers did not introduce a significant shift in the distribution of the features.
- **Relationship with Target**: The outliers did not break or weaken the correlation between the features and the target variable ('y').

Data Imputation:

During the exploration of the dataset, I found that there were no missing values to address. The categorical features included an "unknown" category to handle any potentially missing entries, while the numeric features did not contain any null values. As a result, no imputation was required for this dataset.

Feature Engineering:

In the feature engineering phase, I focused on encoding the categorical variables to make them suitable for model training. I applied One-Hot Encoding to categorical features that don't have a natural order, such as 'job', 'marital', 'contact', 'month', 'day_of_week', and 'poutcome'. For the 'education' feature, I used Ordinal Encoding to assign a numerical value based on the natural hierarchy of education levels. The boolean-like features ('default', 'housing', 'loan') were encoded as binary values, treating 'yes' as 1, 'no' as 0, and 'unknown' as -1. Lastly, the target variable 'y' was also encoded as a binary feature (1 for 'yes' and 0 for 'no').

As part of feature engineering, I made a few key adjustments to the dataset:

- I dropped the 'duration' feature, as it contained future information that could lead to data leakage, ensuring a more realistic model that reflects only the available features at the time of prediction.
- I transformed the 'pdays' feature into a categorical variable, replacing the value 999 with -1 to indicate that the client was not previously contacted. These changes were made to improve the model's reliability and realism in real-world scenarios.

Feature Engineering Summary:
Before feature engineering, the dataset had 21 features. Through encoding and breaking down certain categorical features like "job," "month," "day_of_week," and "marital," the feature count increased to 44. These adjustments were made to enhance the model's ability to learn from various types of data, ensuring that it captures the most relevant information for prediction.

Feature Selection:

For this phase, we employed multiple models with different feature selection methods to identify the most important features for predicting the target variable. The models used were:

- Lasso Regression (L1 Regularization): Identified features with non-zero coefficients.
- Gradient Boosting Regressor: Chose features based on positive feature importances.
- Random Forest Regressor: Selected features with positive importances from decision trees.
- Ridge Regression (L2 Regularization): Identified features with non-zero coefficients.
- XGBoost Regressor: Chose features with positive feature importances from gradient boosting.

The output of the feature selection process is a table where each row represents a feature, and each column corresponds to a different model's feature selection outcome (either 0 for not selected or 1 for selected). The last column sums the selections from all models, providing a final ranking of features based on their importance across all models (maximum score = 5 as the number of the models).

Here's your feature selection summary in bullet points:

- **Features with a total score of 5**: These features were consistently selected by all models and were kept in the dataset.
- **Features with a total score of 4**: These features didn't get a score from Lasso, Given that Lasso is more rigid in feature selection due to its regularization, it might be better to rely on Random Forest and XGBoost, which tend to give more nuanced feature importance in classification tasks.
- **Features with a total score of 3**: These features were isolated and deleted as they were selected by only one or two models. The following features were removed:
    - 'job_entrepreneur'
    - 'job_housemaid'
    - 'job_retired'
    - 'month_nov'
    - 'poutcome_nonexistent'
    - 'loan_encoded'

Remaining features: After the feature selection process, the dataset now contains **38 features**.

Model Selection:

After preparing the dataset, I moved on to splitting it into three distinct subsets: training, validation, and test sets. This split was essential for evaluating the model's performance and ensuring its generalizability:

- Training set: 32,950 samples (80%) – Used to train the models.
- Validation set: 5,075 samples (10%) – Used to fine-tune hyperparameters and validate model performance.
- Test set: 8,238 samples (10%) – Used to evaluate the final model's performance on unseen data.

The selected models for training are:

- Logistic Regression
- Support Vector Machine (SVM)
- Random Forest
- Gradient Boosting
- K-Nearest Neighbors (KNN)
- XGBoost (XGB)

For each model I trained there is a decision boundaries plot that shows the classification clearly.

Summary tables:

After training multiple classification models, I compiled two key summary tables:

- **Confusion Matrix & AUC Summary** – This table provides a comparative view of how well each model distinguishes between positive and negative cases, including key metrics like True Positives (TP), False Positives (FP), False Negatives (FN), and the AUC score.

	Model	TP	FP	TN	FN	AUC
0	Logistic Regression	7250	63	807	118	0.763169
1	SVM	7222	91	786	139	0.564426
2	Random Forest	7058	255	737	188	0.711482
3	Gradient Boosting	7201	112	757	168	0.760751
4	K-Nearest Neighbors	7085	228	738	187	0.681179
5	XGBoost	7178	135	751	174	0.747014

- **Accuracy Score Summary** – A separate table was created to compare the overall accuracy of each model on the test set.

	Model	Test Accuracy
0	Logistic Regression	0.894392
1	SVM	0.893542
2	Random Forest	0.880918
3	Gradient Boosting	0.894513
4	K-Nearest Neighbors	0.882739
5	XGBoost	0.892450

From these analyses, **Logistic Regression** emerged as the best model. The decision was based on multiple factors:

- **Highest True Positives (TP)** – Logistic Regression correctly classified **7,250 positive cases**, the highest among all models.
  Lowest False Positives (FP) and False Negatives (FN) – It made the fewest errors in misclassifying both positive and negative cases.
  Competitive AUC Score (0.763) – While some models had slightly higher AUC scores, Logistic Regression still demonstrated strong performance in ranking predictions.
  Test Accuracy (0.8944) – While Gradient Boosting had a marginally higher accuracy (**0.8945**), the difference was negligible. Since accuracy alone is not the deciding factor, the higher TP count made Logistic Regression the better choice.

Fine-Tuning Logistic Regression:

After selecting **Logistic Regression** as the best-performing model, I explored **hyperparameter tuning** to see if further optimization could improve its performance.
To do so I used **GridSearchCV**, a method that systematically tests multiple hyperparameter combinations to find the best configuration. Despite testing various hyperparameter values, the model's **performance remained unchanged**

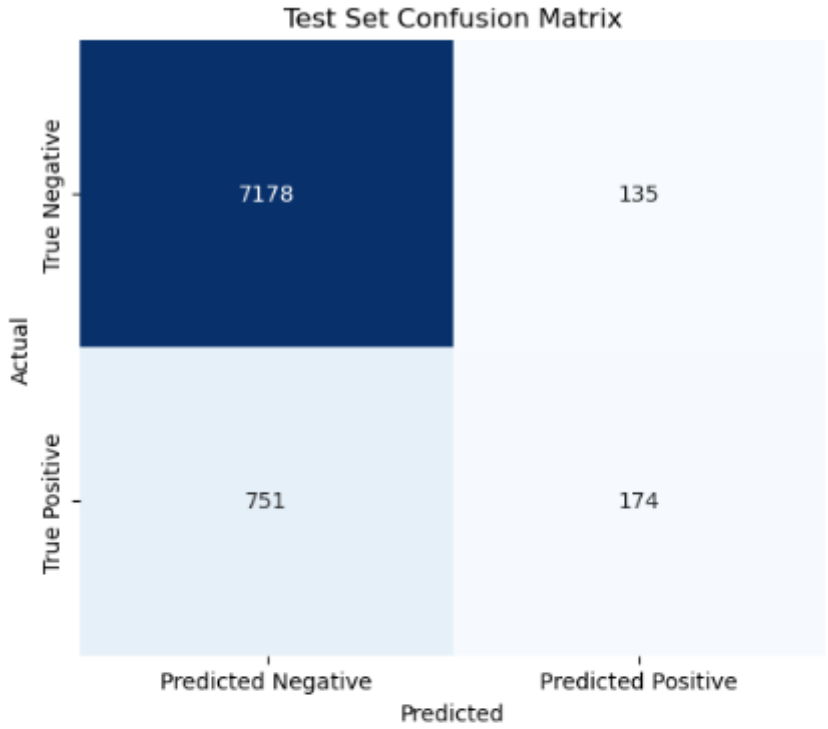
Final Model Evaluation & Feature Importance Analysis:

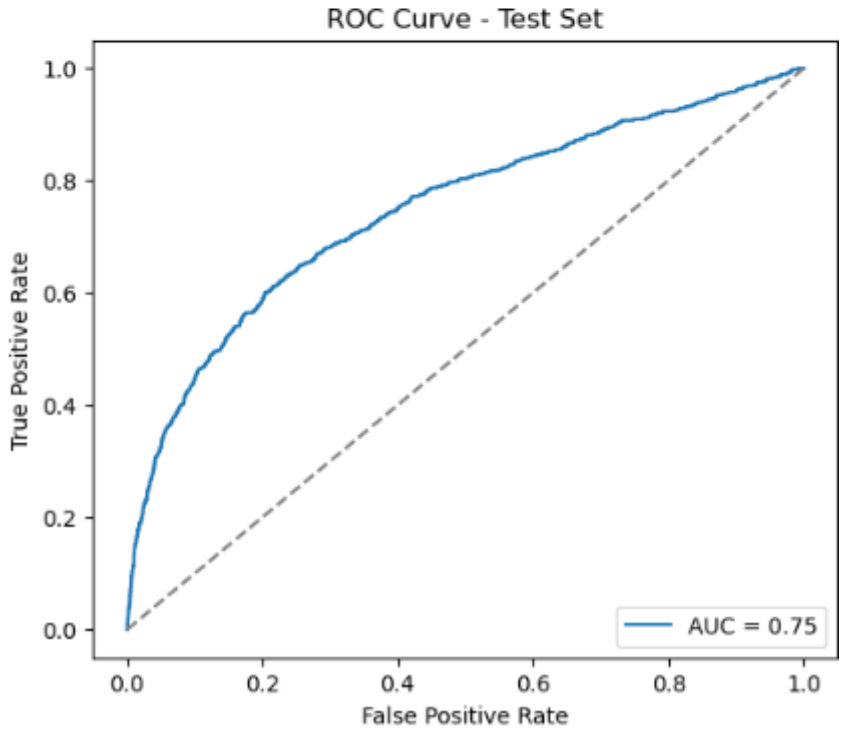
After selecting **Logistic Regression** as the best model, I evaluated its performance on the test set to confirm its effectiveness.

Test Set Performance:

- Logistic Regression correctly classified **89.24%** of cases.
- A **Confusion Matrix** was generated to visualize the classification results.

- An **ROC Curve** was plotted, showing a strong **AUC score**, reinforcing the model's reliability in distinguishing between positive and negative cases.





Feature Importance Analysis:

- Although Logistic Regression was chosen, I used **XGBoost** to analyze feature importance.
- XGBoost's ability to capture non-linear relationships makes it a great tool for understanding which features influence predictions the most.
- The analysis revealed that **"number of employees per quarter"** had the highest importance.
- Given that this campaign was managed through **phone calls**, the number of available employees likely played a crucial role in its success.

Thank You
