Predicting Term Deposit Subscriptions Using Machine Learning

A project on data-driven decision-making for bank telemarketing campaigns.

By Vasavi Busetty, Rupal Jha & Jyoti Bhardwaj

18 January 2025
Date: January 2025

Understanding about the Dataset

Objective:

Predict if a customer will subscribe to a term deposit (y: Yes/No)

Dataset Details:

- **Source:** Direct marketing campaigns of a Portuguese bank.
- **Train Data:** 40,000 rows, 17 columns.
- **Test Data:** 5,211 rows, 16 columns.

Key Features:

- **Client Data:** Age, job, marital status, education, loan status, balance.
- **Contact Details:** Last contact type (cellular/telephone), day, month, duration.
- **Campaign Performance:** Number of contacts, previous outcomes (poutcome).

Problem Approach

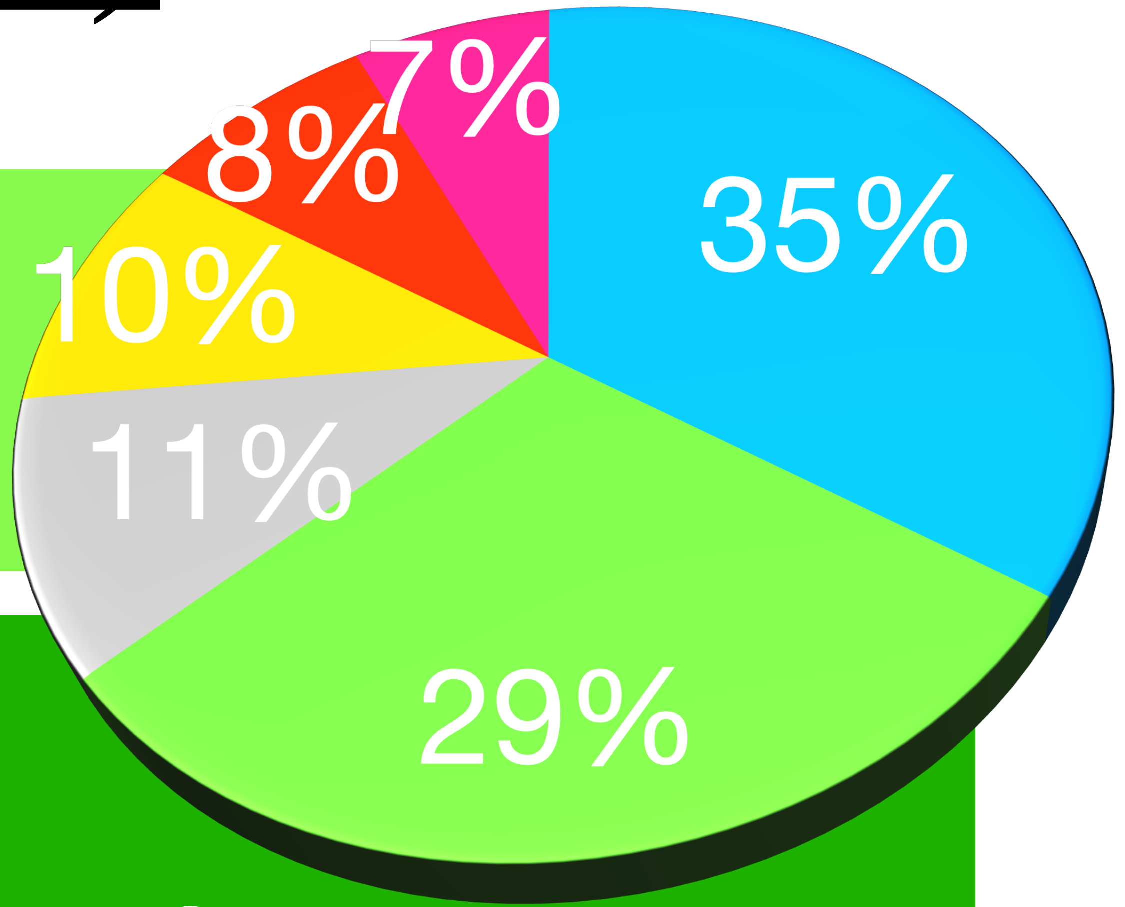
Define the Problem:

Identify customers most likely to subscribe to a term deposit to optimize telemarketing efforts.

Steps Taken:

- Data Cleaning
- Exploratory Data Analysis (EDA)
- Feature Engineering
- Model Selection & Tuning
- Evaluation & Inference

Exploratory Data Analysis (EDA)

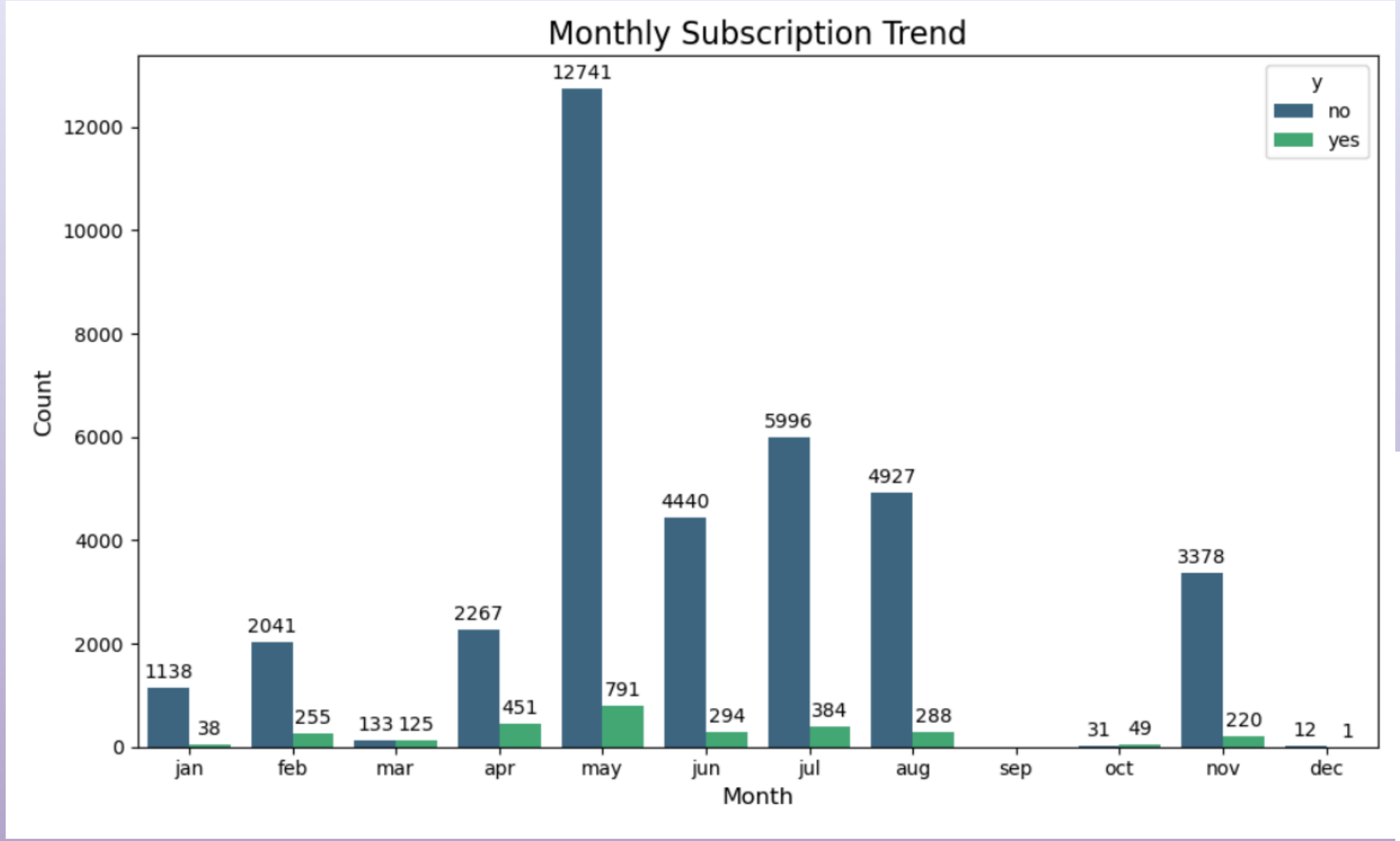


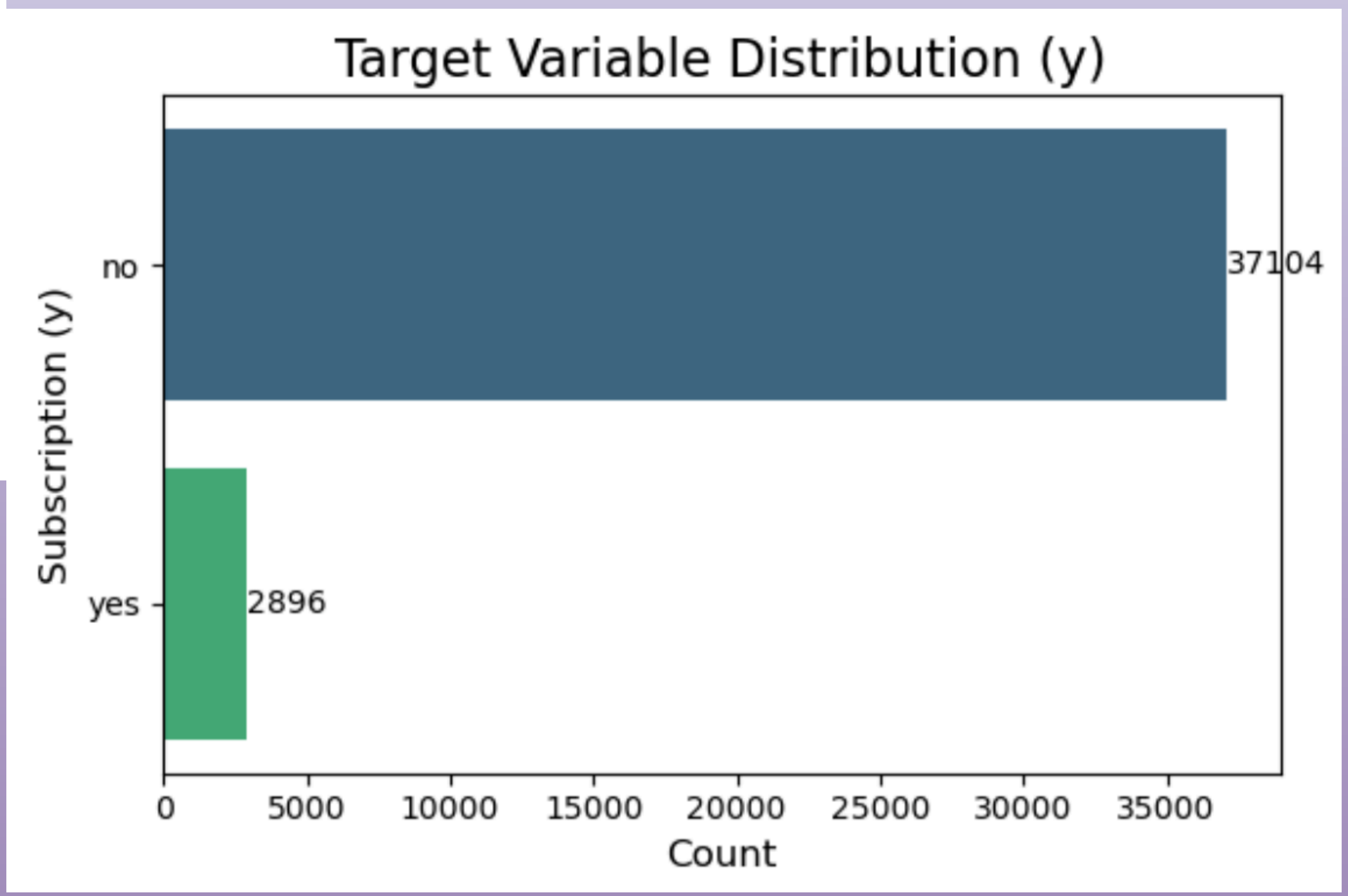
- **Goals of EDA:**
  - Identify patterns, trends, and anomalies using various libraries of python.
  - Understand relationships between features and the target variable (y).

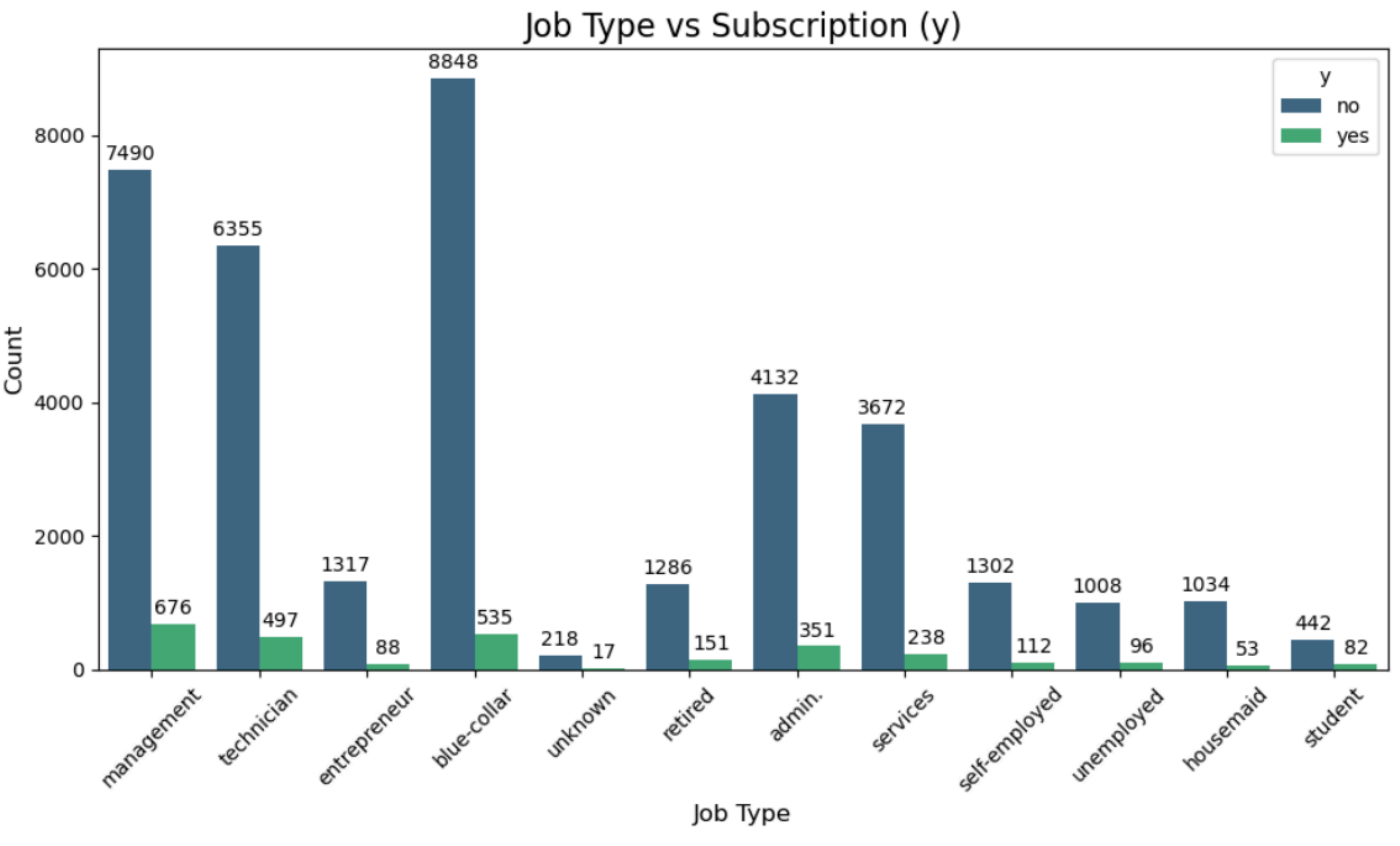
- **Key Insights:**
  - Most customers were contacted in **May and July**.
  - **Very few people i.e. 7.24% are taking** subscriptions.This indicates a significant imbalance in the dataset, suggesting the need for techniques like oversampling or undersampling during modeling.
  - Single customers are more likely to subscribe(9.4%) compared to married(6.1%)
  - Subscriptions are higher when contact is made via cellular (8.96%) compared to unknown methods (3.89%).

- **Visuals:**
- Included charts like:
  - Distribution of subscription(y) variable.
  - Subscription rate by job and education.
  - Relationship between poutcome and y.

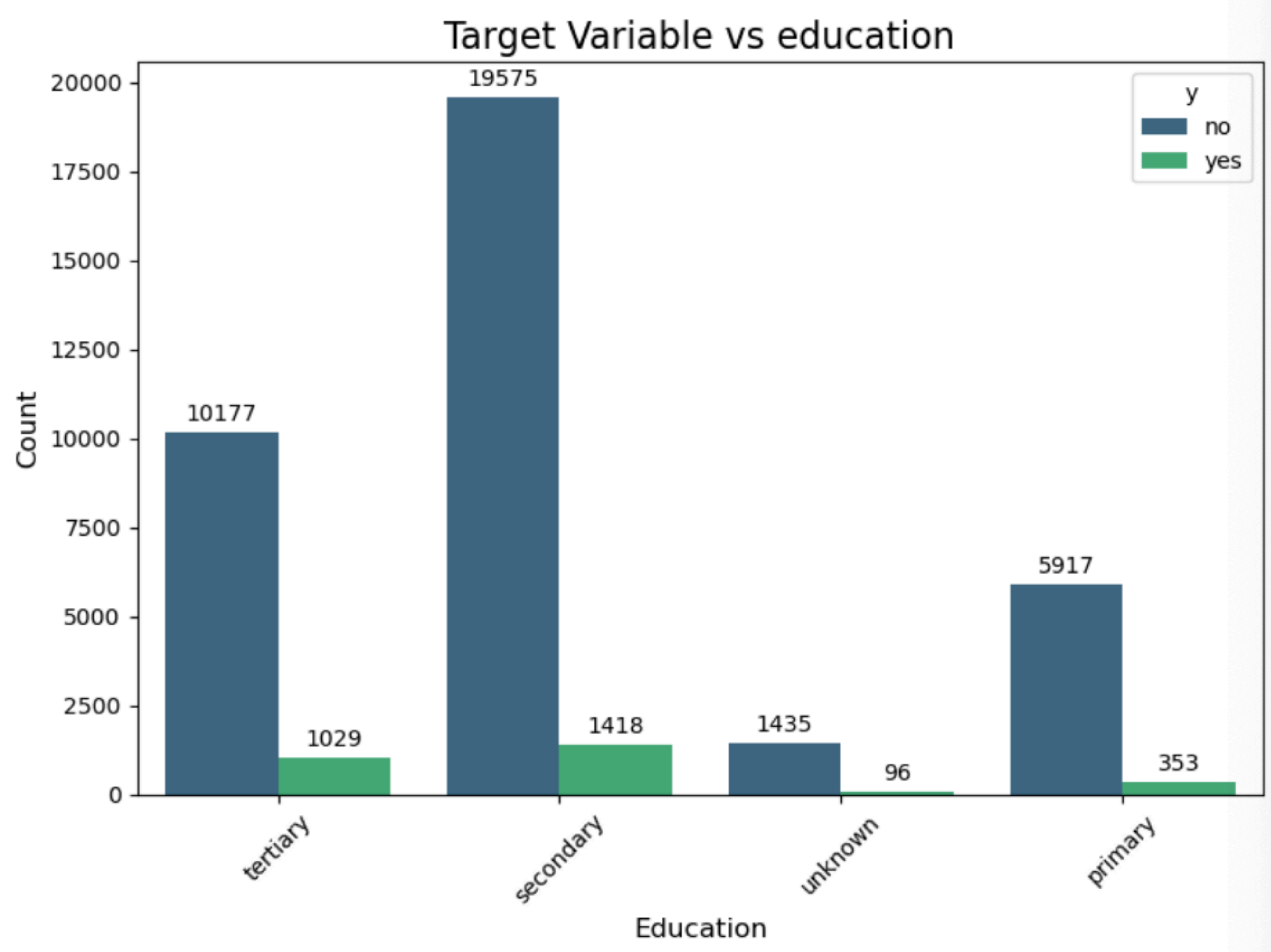
Key Visualisations



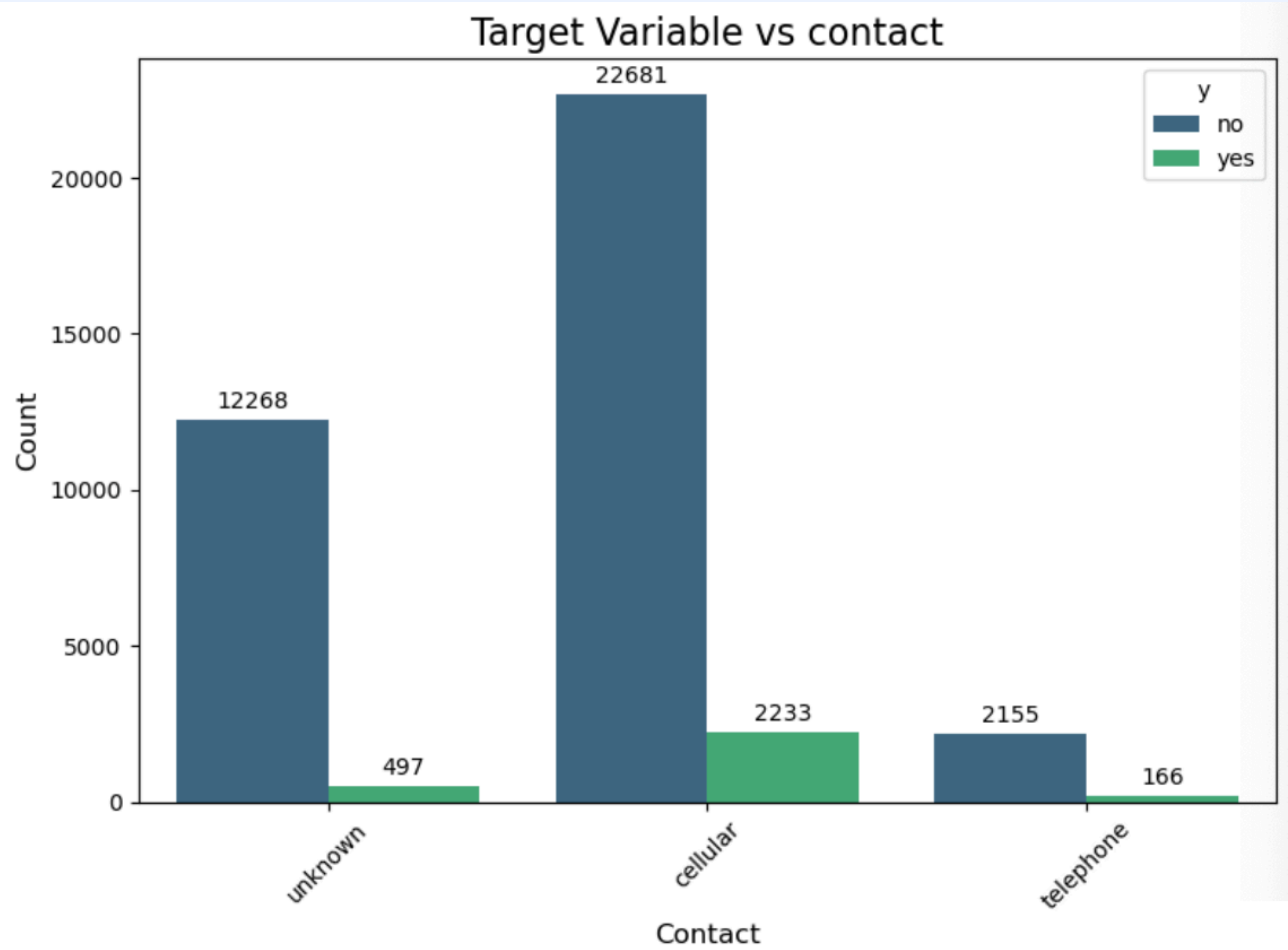


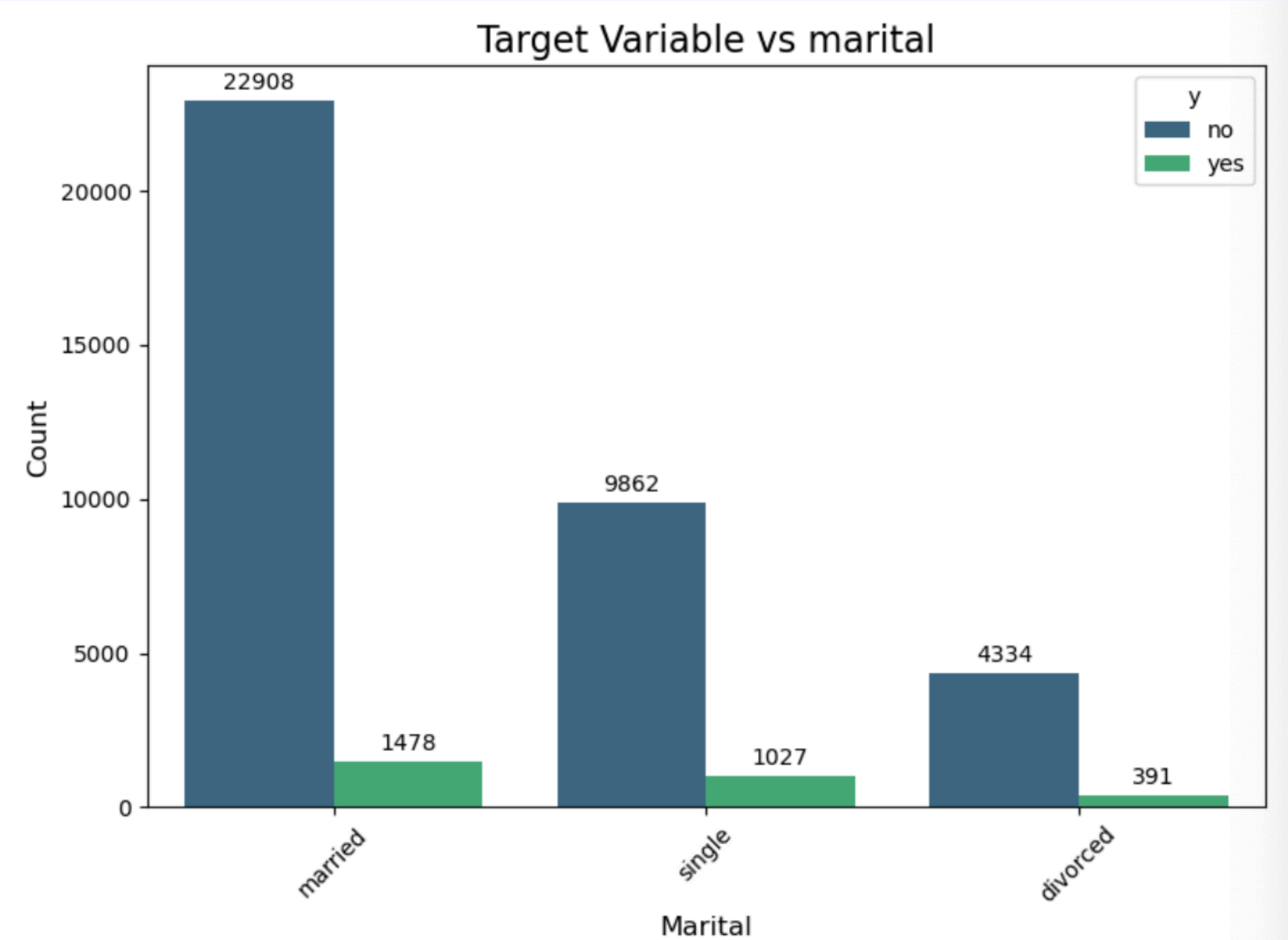


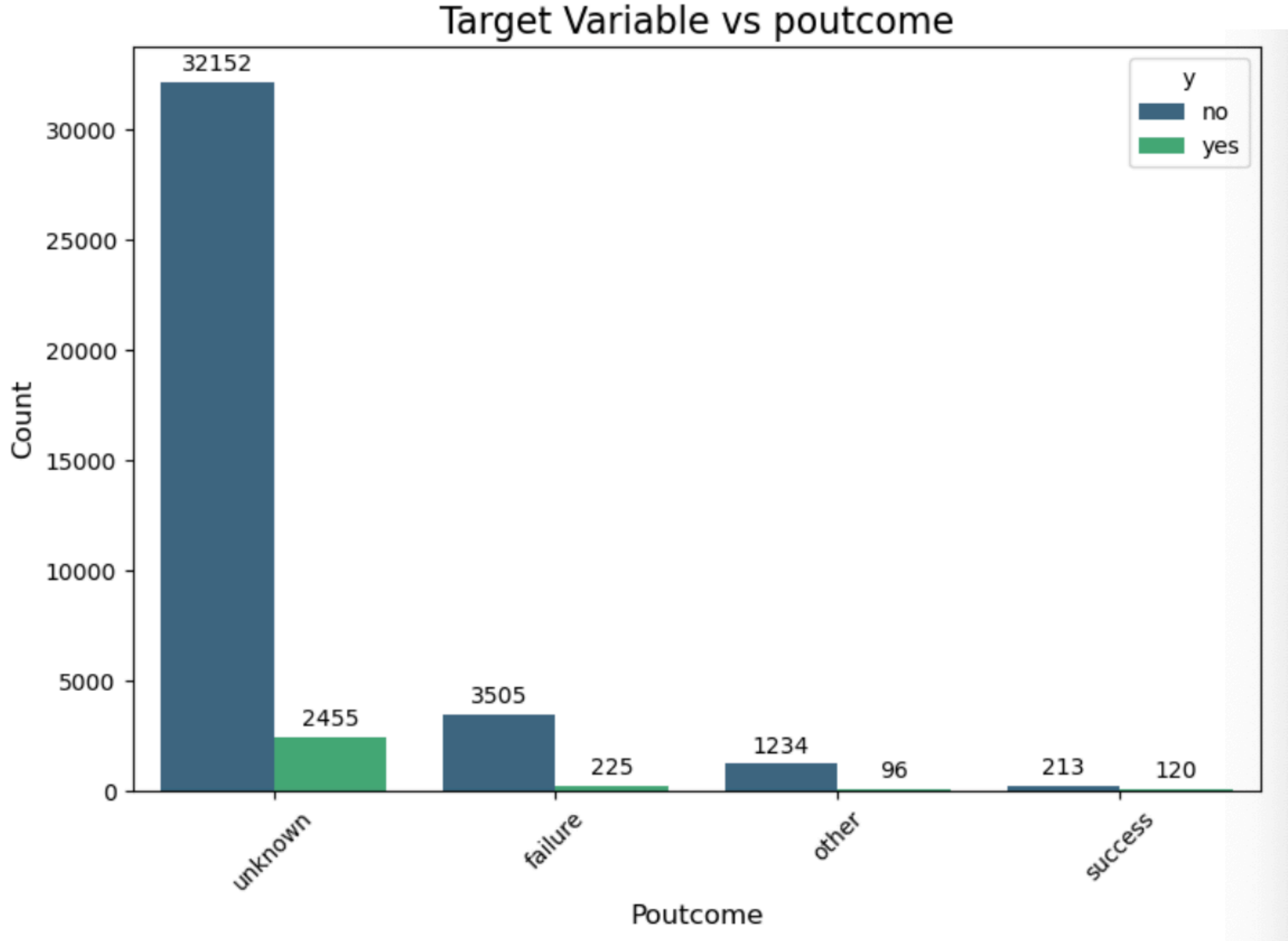
Job Type vs Subscription (y)



Target Variable vs education


Target Variable vs contact
y
no
yes
Count
0
5000
10000
15000
20000
12268
497
22681
2233
2155
166
unknown
cellular
telephone
Contact


Target Variable vs marital
y
no
yes
Count
0
5000
10000
15000
20000
22908
1478
9862
1027
4334
391
married
single
divorced
Marital


Target Variable vs poutcome
y
no
yes
Count
0
5000
10000
15000
20000
25000
30000
32152
2455
3505
225
1234
96
213
120
unknown
failure
other
success
Poutcome

Other Takeaways using EDA

Numeric Feature Summary:

- **Age**: Average age is 40.54 years, with some outliers above 70.
- **Balance**: Median account balance is ₹407, but outliers exist on both ends (e.g., as low as ₹-8,019 and as high as ₹102,127).
- **Duration**: The median call duration is 175 seconds, with a maximum of 4,918 seconds.

Outlier Analysis:

- **Balance**: 4,280 outliers exceed ₹3,216.
- **Duration**: 3,000 outliers surpass 632 seconds.
- **Campaign**: Campaign efforts over six attempts have 2,992 outliers.
- **Pdays/Previous**: A majority (-1 or 0 values) suggest clients not previously contacted.

Model Selection

- **Challenges Identified:**

  - Imbalanced dataset (y: Many "no", few "yes").
  - Mixed data types: Categorical and numeric.

- **Models Considered:**

  - Logistic Regression (simple, interpretable).
  - Random Forest (handles mixed data, avoids overfitting).
  - Gradient Boosting (XGBoost, LightGBM for better performance).

- **Final Model Chosen: Random Forest**

  - **Reason:** Strong balance of accuracy, interpretability, and speed.
  - **Metrics Used:** Accuracy, Precision, Recall, F1-Score, ROC-AUC.

Workflow of Machine Learning

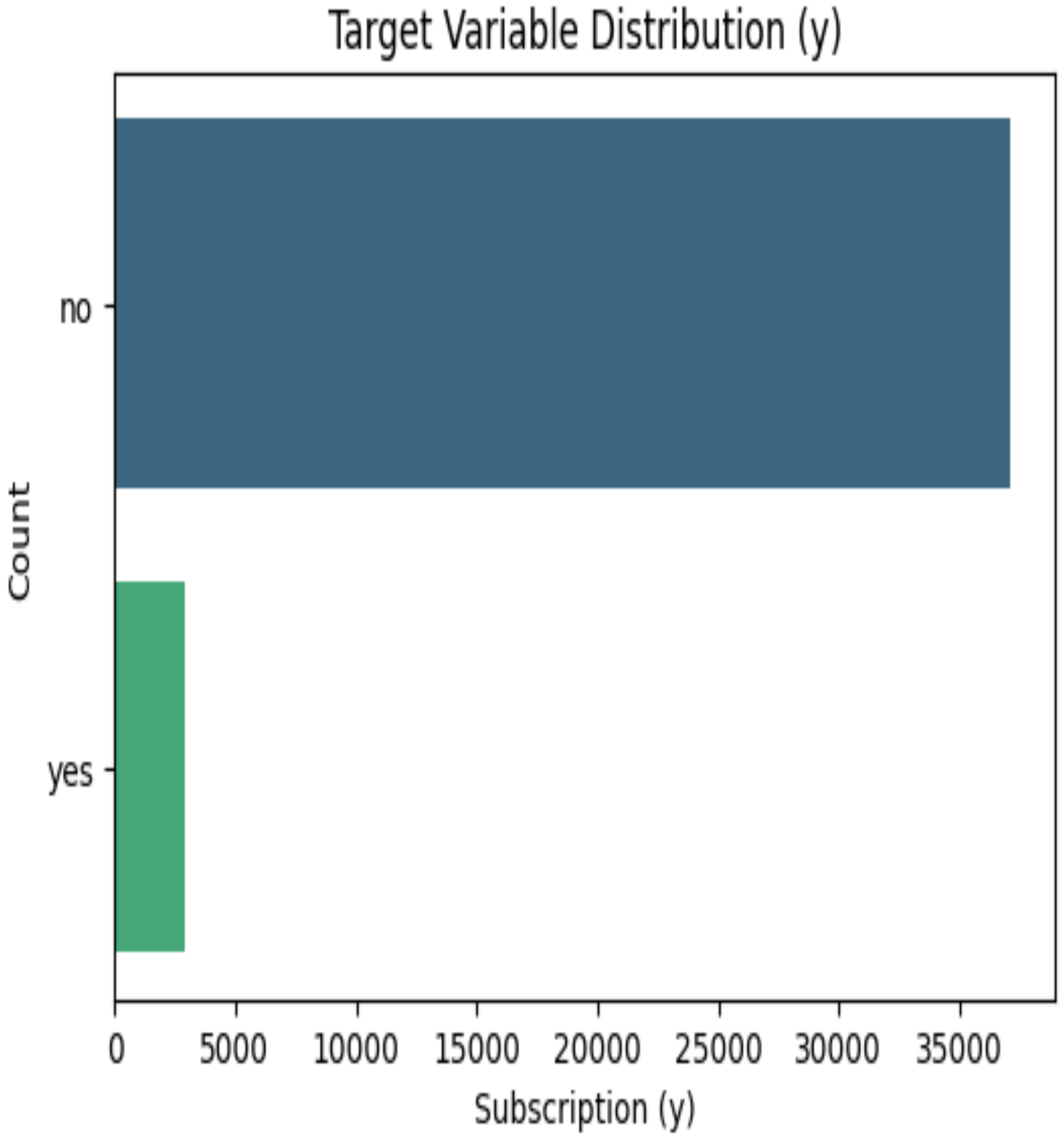
1. Data Loading and Initial Exploration-Import necessary libraries like pandas, NumPy, seaborn, and matplotlib.
2. Load the training and testing datasets using pd.read_csv(). Display the first few rows of the datasets using head().
3. Check for missing values using isnull().sum(). Perform initial data exploration by visualising the distribution of the target variable ('y') and other features using sns.countplot(), sns.histplot(), and sns.boxplot().
4. This helps understand the data characteristics and potential patterns.

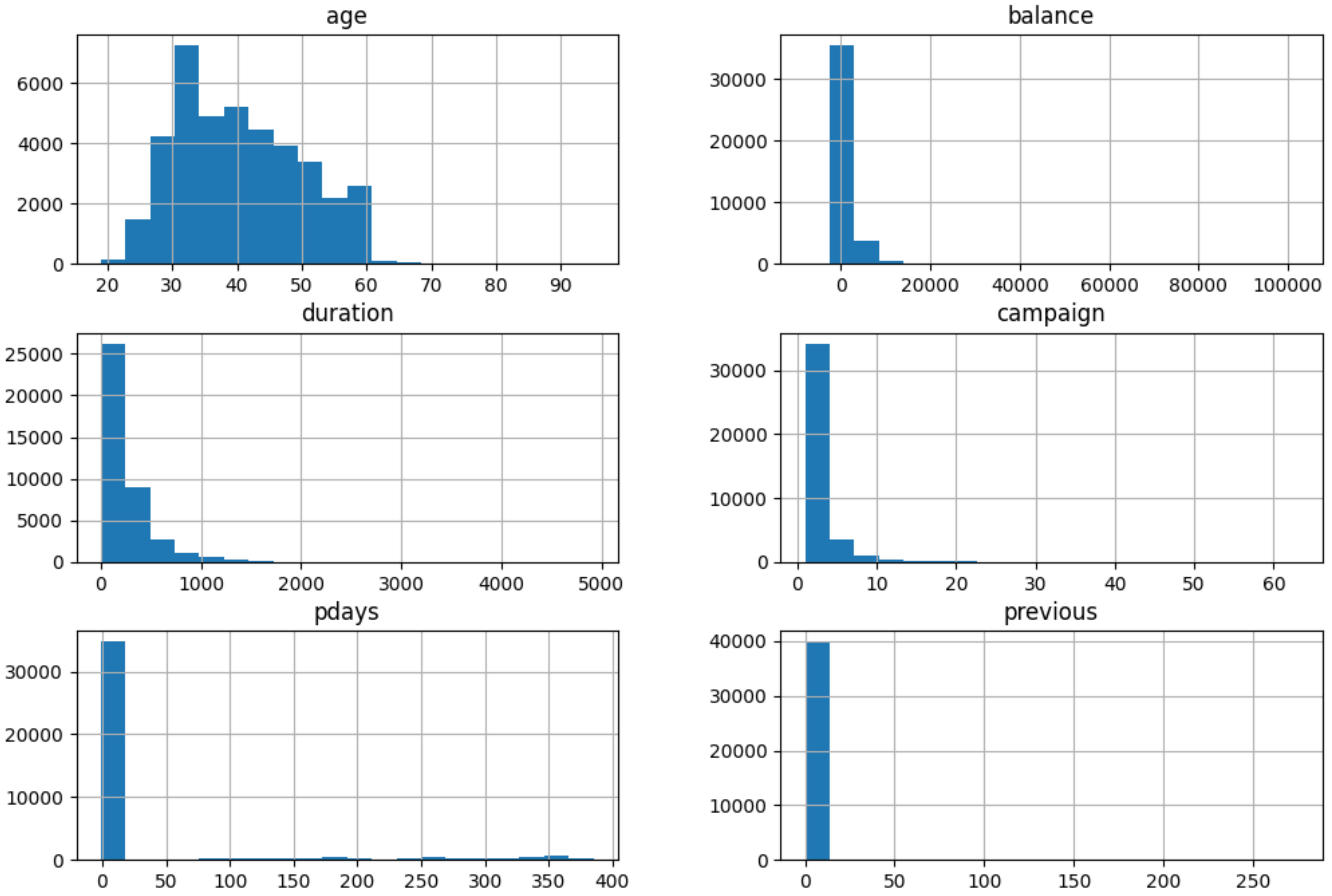
5. Data Preprocessing Combine the training and testing datasets for consistent preprocessing using pd.concat().
6. Identify categorical columns for one-hot encoding. Perform one-hot encoding on categorical columns using pd.get_job() to convert them into numerical representations.
7. Split the combined dataset back into training and testing sets. Convert the target variable ('y') to numerical values (0 and 1) using map().
8. Scale numerical features using StandardScaler() to ensure they have a similar range and prevent features with larger values from dominating the model.

9. Model Selection and Training Select a suitable machine learning model. In this case, RandomForestClassifier was chosen.
10. Split the training data into training and validation sets using train_test_split() to evaluate the model's performance.
11.Instantiate the model with desired hyperparameters. Train the model on the training data using fit().
12.Model Evaluation Make predictions on the validation set using predict(). Evaluate the model's performance using metrics like classification report, confusion matrix, and F1 score. These metrics provide insights into the model's accuracy, precision, recall, and overall effectiveness.

13. Hyperparameter Tuning: Fine-tune the model's hyperparameters using techniques like GridSearchCV to find the optimal settings that improve performance. Evaluate the tuned model's performance using the same metrics as before.
14. Prediction on Test Data Prepare the test data by ensuring it has the same features and preprocessing steps as the training data. Make predictions on the test data using the trained model. Convert the predictions back to the original format if necessary (e.g., from numerical values to 'yes' and 'no').

Target variable distribution





Model Output

Performance Metrics:

Accuracy: 0.93
Precision (Yes class): 0.64
Recall (Yes class): 0.06
F1-Score: 0.11
F1 Score (Tuned Model): 0.36868686868686867
ROC-AUC: 0.53

Model Performance

- **Accuracy:** 93% This indicates that the model correctly classified 93% of the instances in the dataset. While accuracy is a commonly used metric, it can be misleading, especially in cases of imbalanced datasets. Given the other metrics, the high accuracy may suggest that the model is good at predicting the majority class ("no" in this case), but struggles with the minority class ("yes").

- **Precision (Yes class):** 64% This means that out of all the instances predicted as "yes" by the model, 64% were actually "yes." In other words, when the model predicts a positive outcome, it's correct about 64% of the time. A higher precision is generally desirable, as it indicates fewer false positives.

- **Recall (Yes class):** 6% This indicates that out of all the actual "yes" instances in the dataset, the model only correctly identified 6% of them. This low recall suggests that the model is missing a significant number of positive cases. This is a critical issue, especially if the goal is to identify as many positive cases as possible (e.g., in medical diagnosis or fraud detection).

- **ROC-AUC:** 0.53 The ROC-AUC score of 0.53 is slightly above 0.5, which is the baseline for a random classifier. This suggests that the model has some predictive power, but it's only slightly better than random chance. A higher ROC-AUC score would indicate better discrimination between positive and negative cases. helpful

- **Overall Interpretation:** Based on these metrics, it appears that the model has high accuracy but suffers from low recall. This implies that it's good at predicting the majority class ("no"), but struggles to correctly identify the minority class ("yes"). Model is not performing well in identifying positive cases,needs to be improved.

Final Inference

1. **Key Takeaways:**
   - Customers with longer call durations and higher balances are more likely to subscribe.
   - Past campaign outcomes (`poutcome`) significantly impact the likelihood of subscription.
   - Over-contacting customers is counterproductive.
2. **Recommendations:**
   - Focus on customers with high balances and successful previous outcomes.
   - Optimize call duration for higher effectiveness.
   - Avoid repeated calls for non-responsive customers.
3. **Business Impact:**
   - Better targeting reduces marketing costs and improves conversion rates.

FUTURE SCOPE OF IMPROVEMENT

- **Class imbalance:** The dataset may have a significant imbalance between the "yes" and "no" classes, leading to the model being biased towards the majority class. Addressing class imbalance techniques like oversampling, undersampling, or using cost-sensitive learning can potentially improve the recall.
- **Model selection and hyperparameters:** The chosen model or its hyperparameters might not be suitable for the task. Experimenting with different models or tuning the hyperparameters could potentially enhance performance.
- **Feature engineering:** The existing features may not be informative enough for the model to effectively distinguish between classes. Exploring new features or transforming existing ones might be helpful.

THANK YOU