# TERM DEPOSIT TURNOVER

Letizia Ghilardi - Cecilia Paolini - Matteo Montrucchio

# INDEX

# Values

**Numerical Features**

- **Id:** Unique identifier for each record.
- **age:** Age of the client.
- **balance:** Account balance.
- **day:** Day of the month for the last contact.
- **duration:** Duration of the last contact in seconds.
- **campaign:** Number of contacts performed during this campaign.
- **pdays:** Number of days since the client was last contacted from a previous campaign.
- **previous:** Number of contacts performed before this campaign.

**Categorical Features**

- **job:** Type of job.
- **marital:** Marital status.
- **education:** Level of education.
- **default:** Has credit in default (yes/no).
- **housing:** Has a housing loan (yes/no).
- **loan:** Has a personal loan (yes/no).
- **contact:** Contact communication type.
- **month:** Month of the last contact.
- **poutcome:** Outcome of the previous marketing campaign.

**Target Variable**

**y:** Whether the client has subscribed to a term deposit (yes/no).

# Data Cleaning

# Data Cleaning

- Removing rows with missing data (NAs), 11 observations removed
- Removing "**id**" variable
- Removing "**duration**" variable:
  - it strongly influences the target (e.g., if duration=0, then y='no')
  - but, it is only known after the call is completed
  - by that point, the outcome ("**y**") is already determined

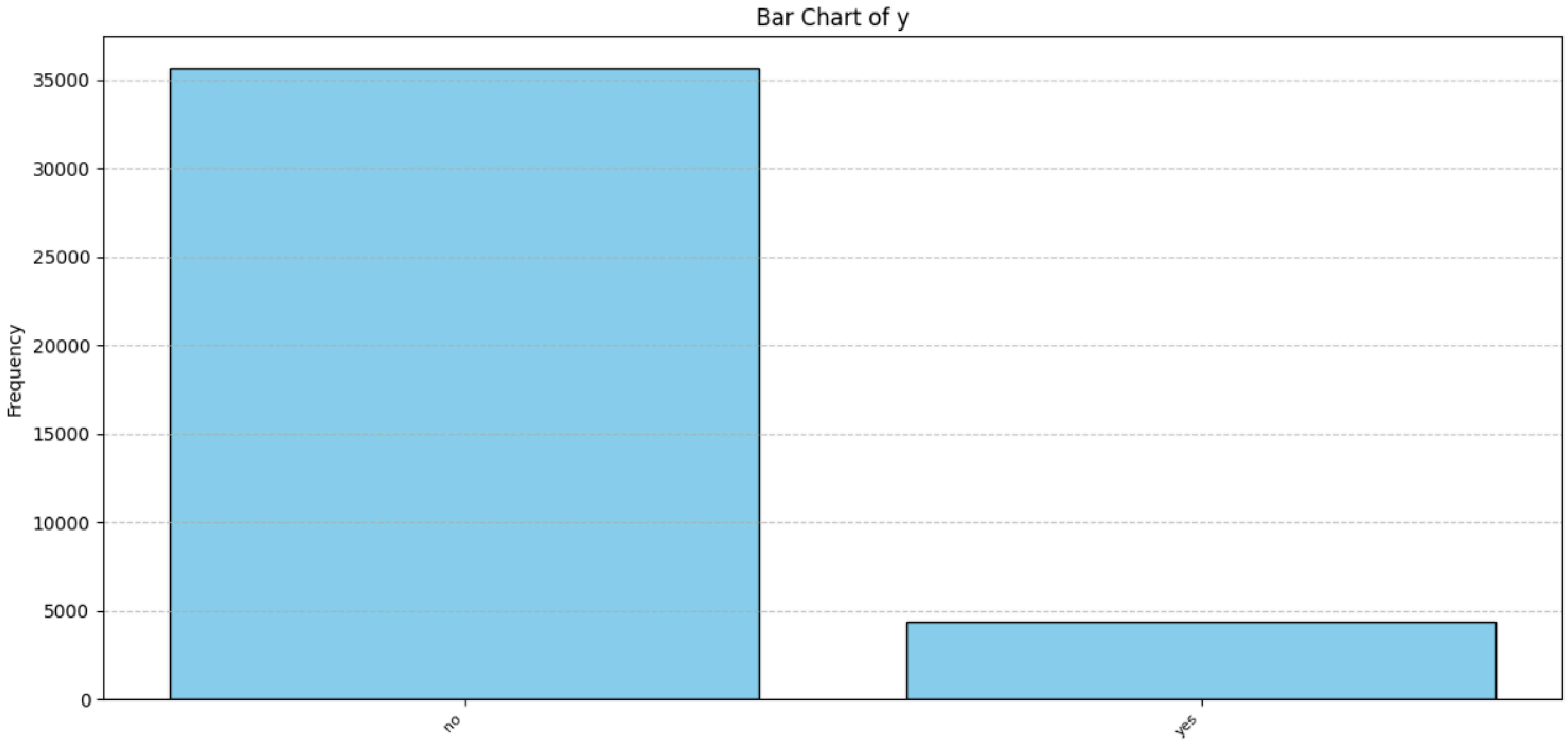
# Exploratory Data Analysis

# EDA

- The target variable is **highly imbalanced**

- “no” label: 39,911
- “yes label: 5,289

- Consequences:
  - our models tend to be biased toward the majority class
  - they may struggle to correctly predict the minority class ("yes").



# QQ-plot & Box Plots

- variables considered:
  - **“age”**
  - **“balance”**
  - **“campaign”**
  - **“previous”**

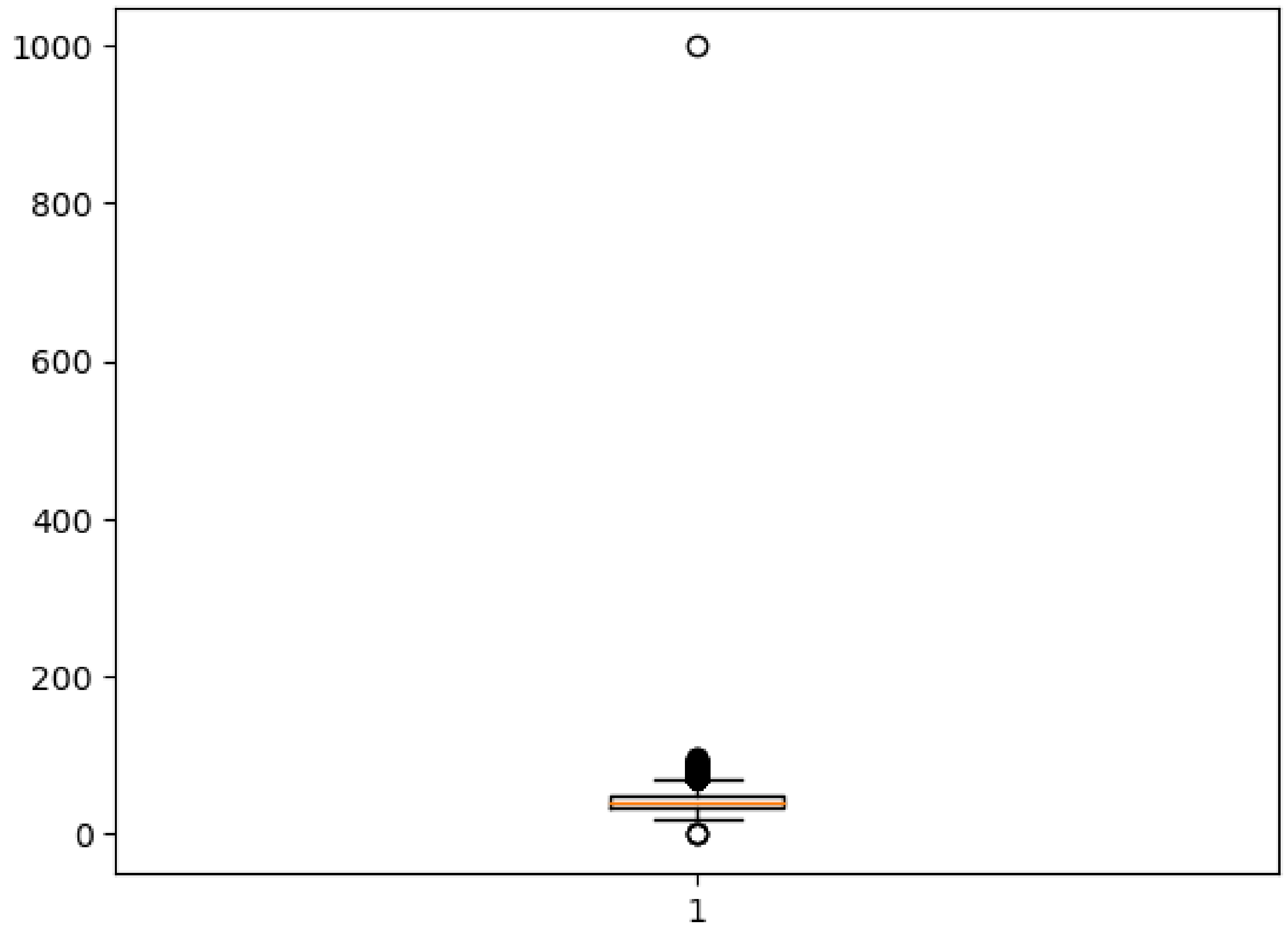
- outliers removed upon inspection (IRQ method)
  - outliers: $< Q1 - 1.5 \times IQR$ or $> Q3 + 1.5 \times IQR$

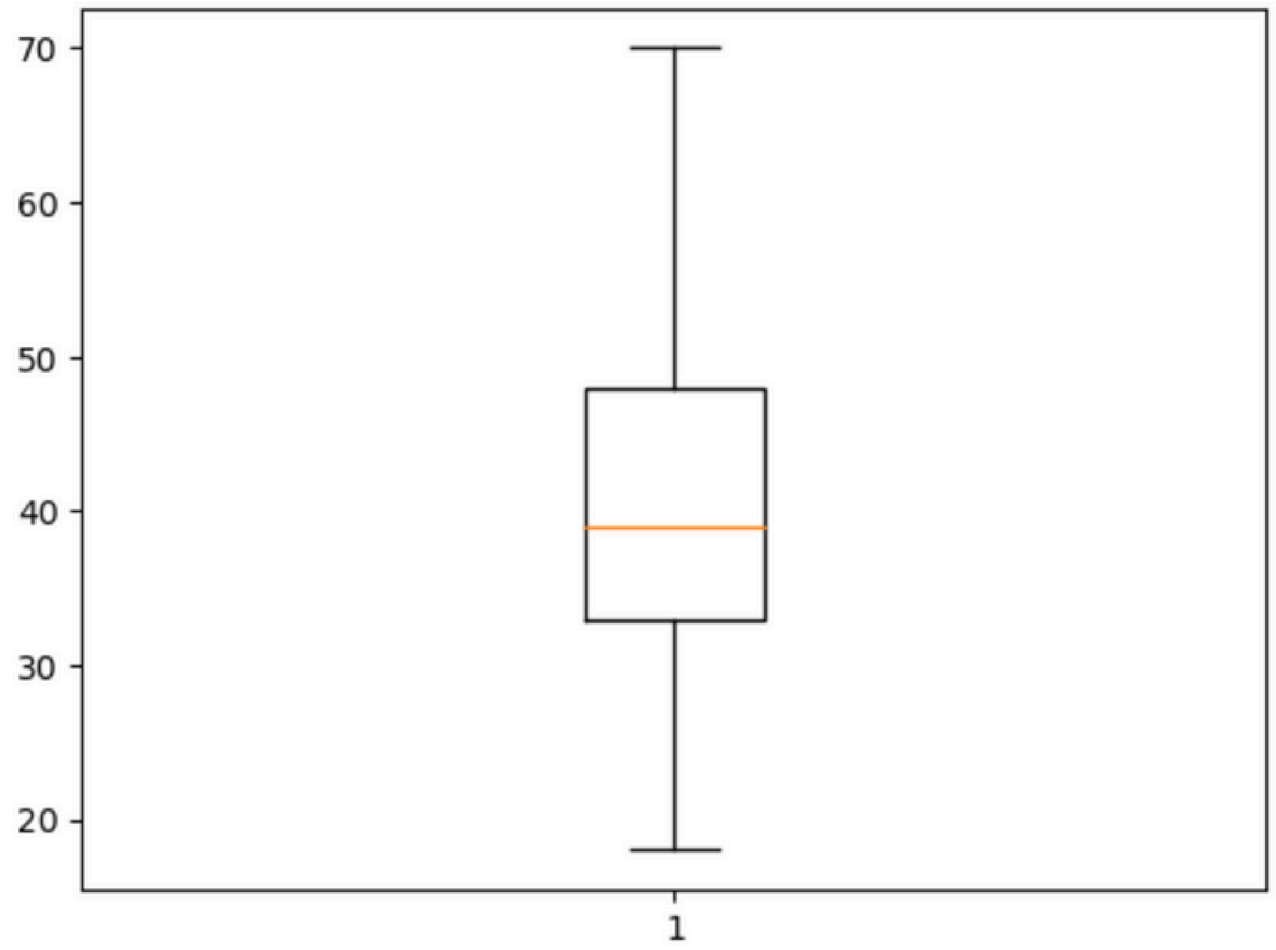
# Box plot illustration

Before

**Age**

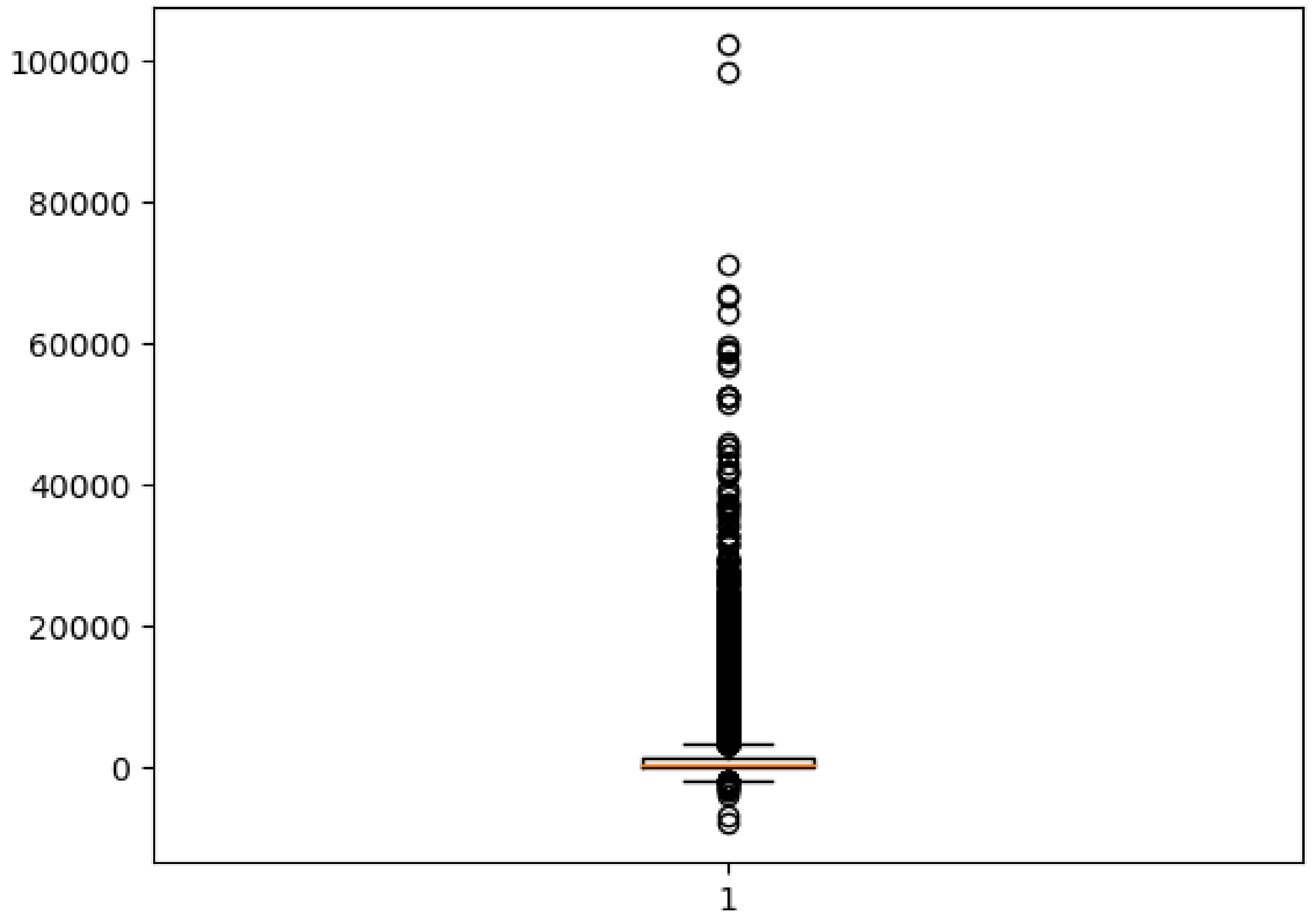
After

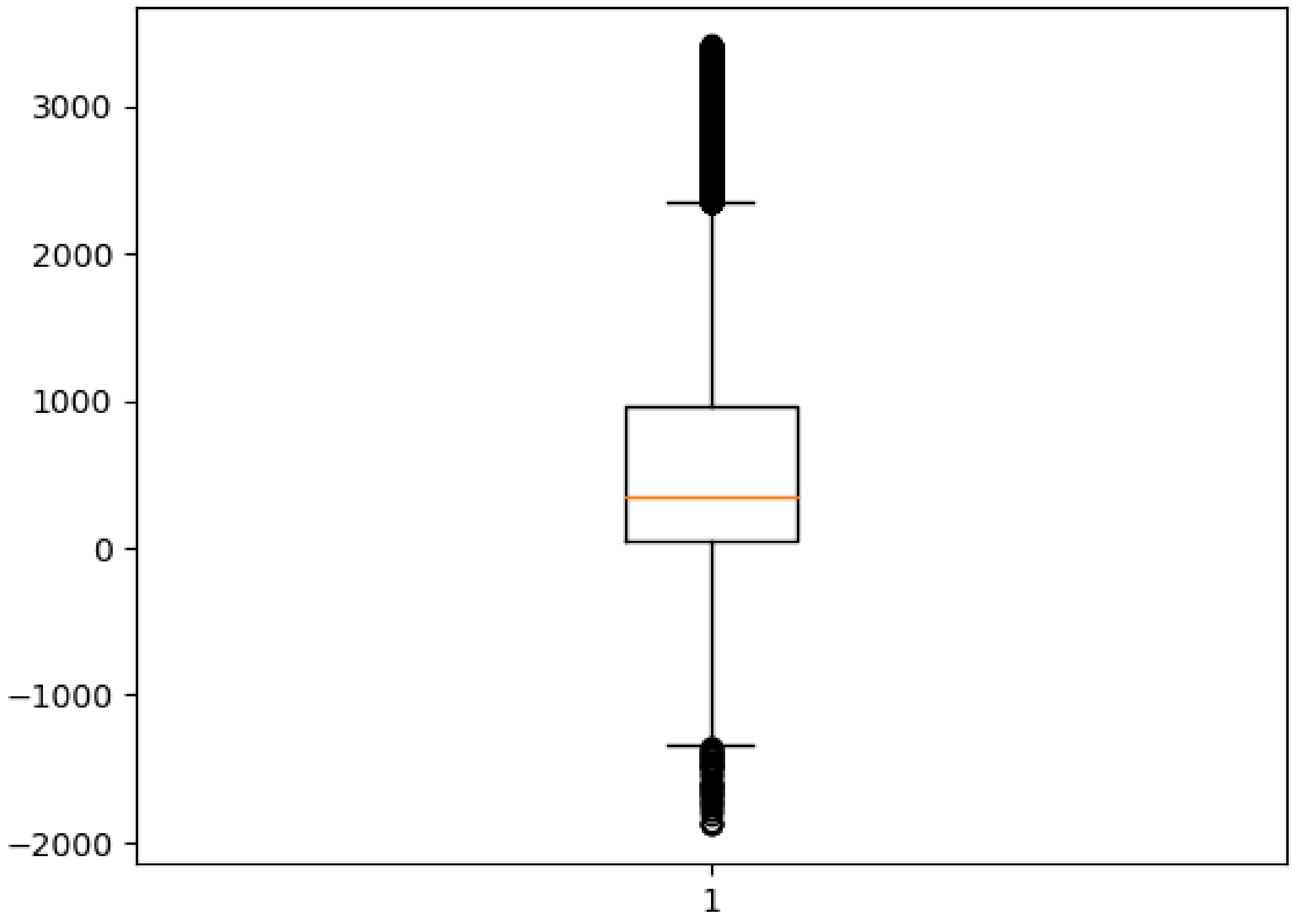


# Box plot illustration

Before **Balance** After



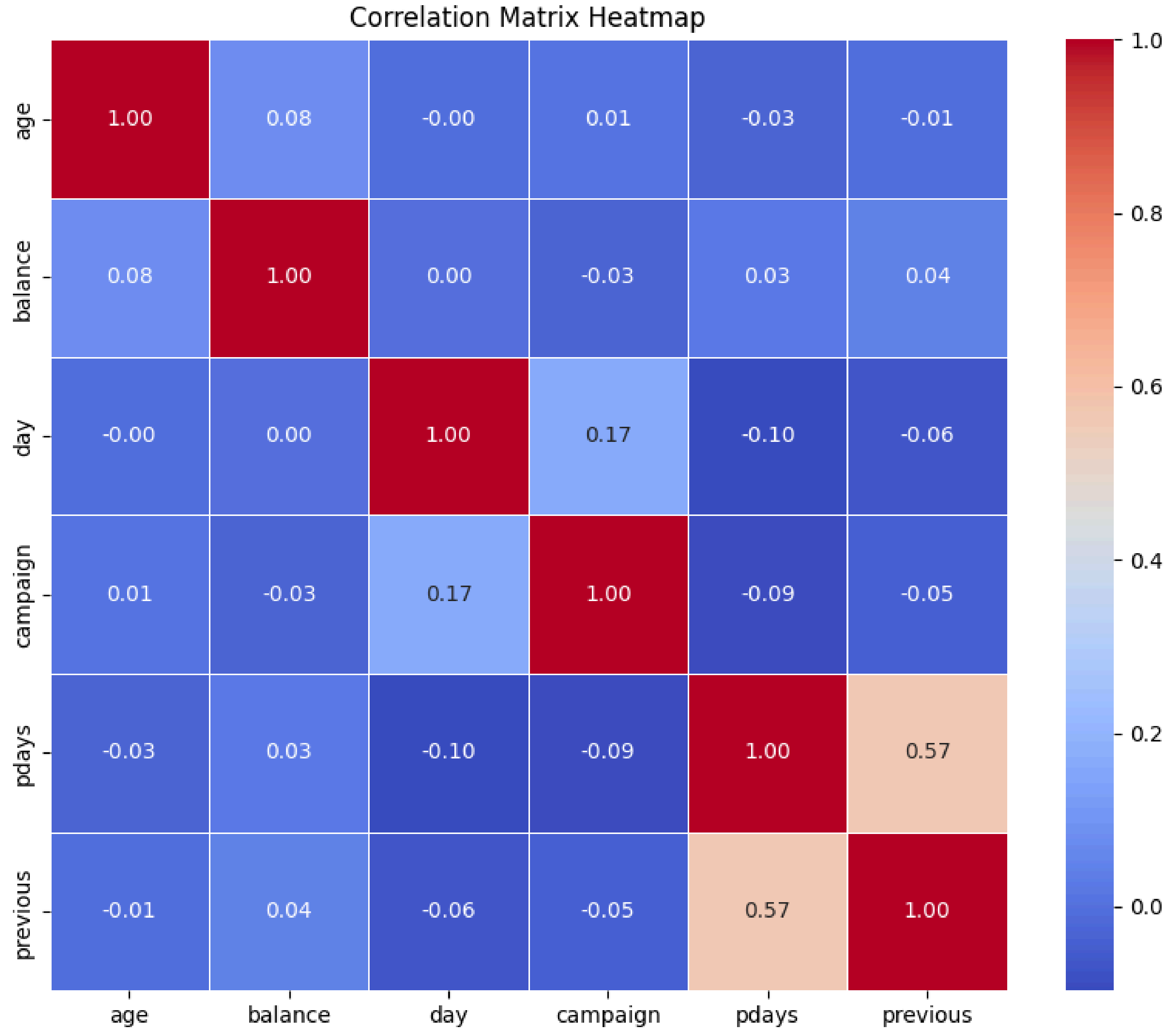



# Correlation Matrix

- Correlation between the variables quite weak (minimal linear relationships across dataset)

- the strongest positive correlation is between "**pdays**" and "**previous**":
  - they measure aspects of the client's previous contact history

- low correlations suggest that the variables are largely *independent*, (little redundancy among predictors)



# Encoding

# Label Encoding

- Applied to all variables that are categorical or contain string data
- It can imply an ordinal relationship (hierarchy among categories in a variable)
- Why not *one-hot encoding* (separate binary column for each category in a variable)?
  - increase dimensionality in the data & higher costs while performing the models (efficiency)
  - hard to compute (feasibility)
- Our models demonstrated satisfactory performance

# Logistic Regression

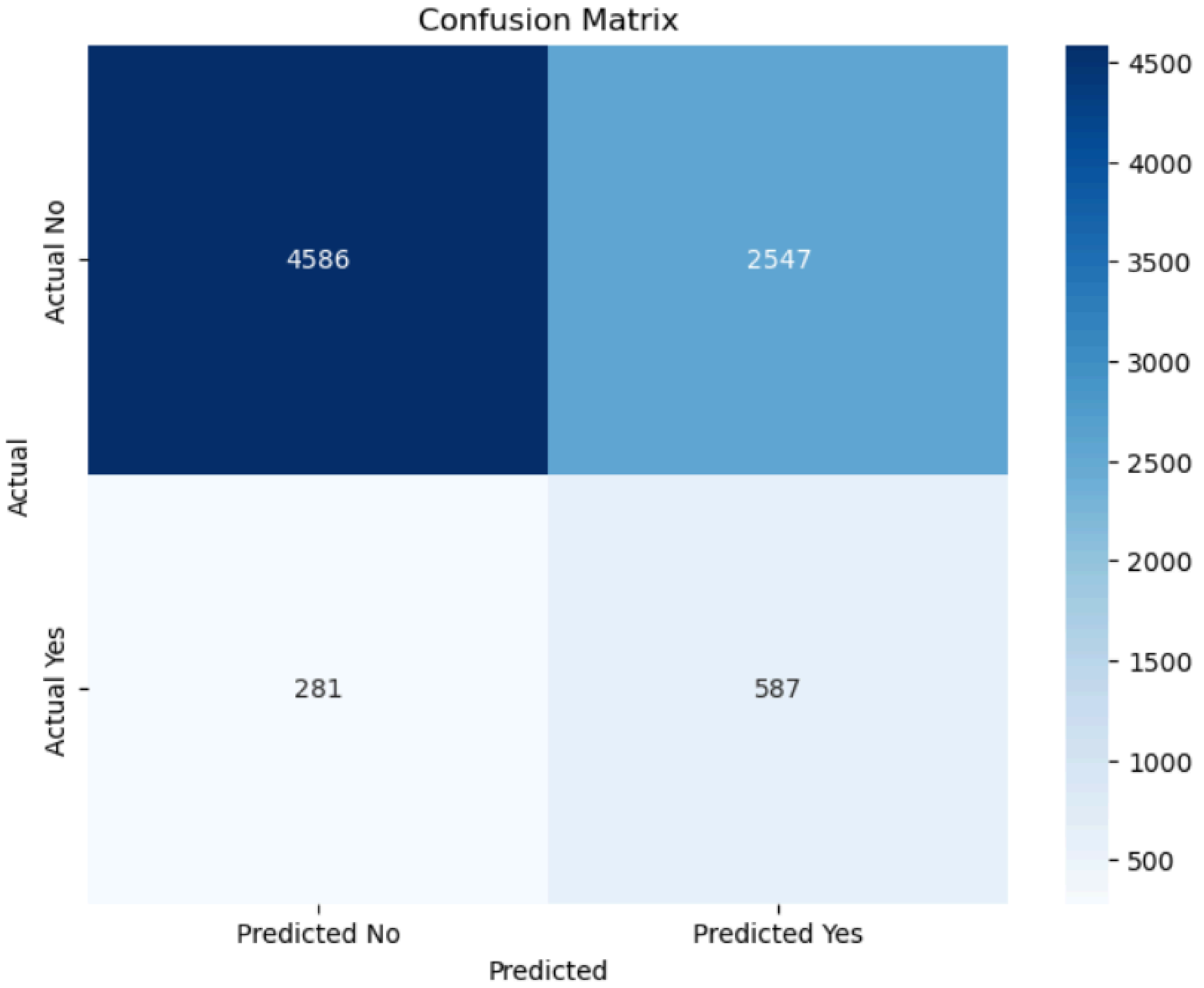
# Logistic Regression

- Logistic regression is interpretable but struggles with class imbalance.

- The model performs well for the majority class (class 0) but poorly for the minority class (class 1).

- Overall accuracy 64.65%

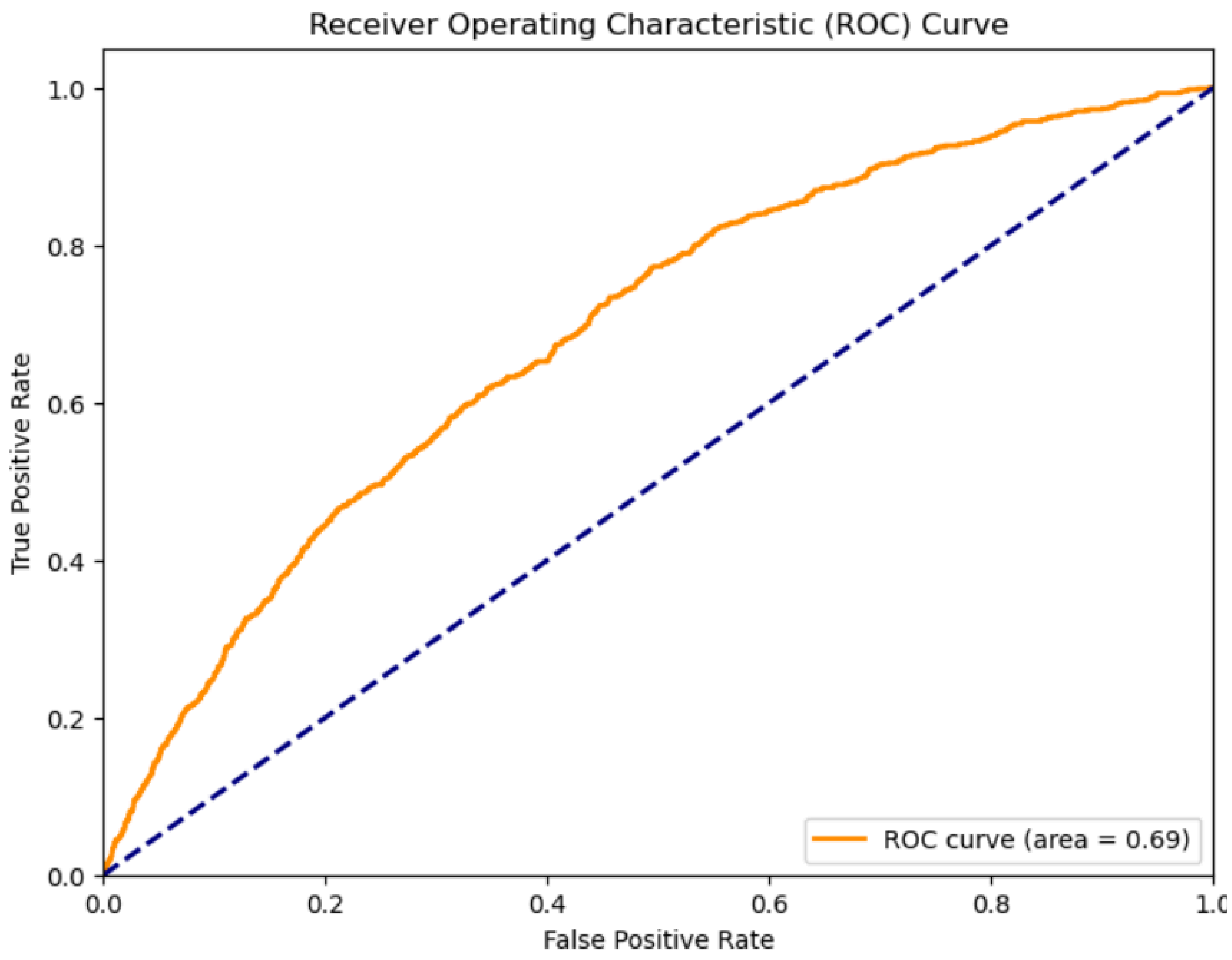
```
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.64      0.76      7133
           1       0.19      0.68      0.29       868

    accuracy                           0.65      8001
   macro avg       0.56      0.66      0.53      8001
weighted avg       0.86      0.65      0.71      8001


Accuracy Score: 0.6465441819772528

Confusion Matrix:
 [[4586 2547]
 [ 281  587]]
```



# ROC Curve

- AUC = 0.69 indicates moderate ability to distinguish between classes but far from optimal.

- Curve Shape:
  - Doesn't sharply rise toward the top-left corner.
  - Shows struggle to achieve high sensitivity without increasing false positives.



# Artificial
# Neural Network

# ANN Code

```
model3 = Sequential([
    Dense(128, input_dim=15, activation='relu'),
    Dropout(0.1),
    Dense(64, activation='relu'),
    Dropout(0.1),
    Dense(32, activation='relu'),
    Dropout(0.1),
    Dense(1, activation='sigmoid')
])

model3.compile(optimizer=Adam(learning_rate=0.001),
               loss='binary_crossentropy',
               metrics=['accuracy'])

history = model3.fit(X_train_balanced, y_train_balanced,
                     validation_data=(X_test, y_test),
                     epochs=50,
                     batch_size=32,
                     verbose=1)
```

- We have realized 8 different versions, but the one on the left is the one that gave us the higher accuracy

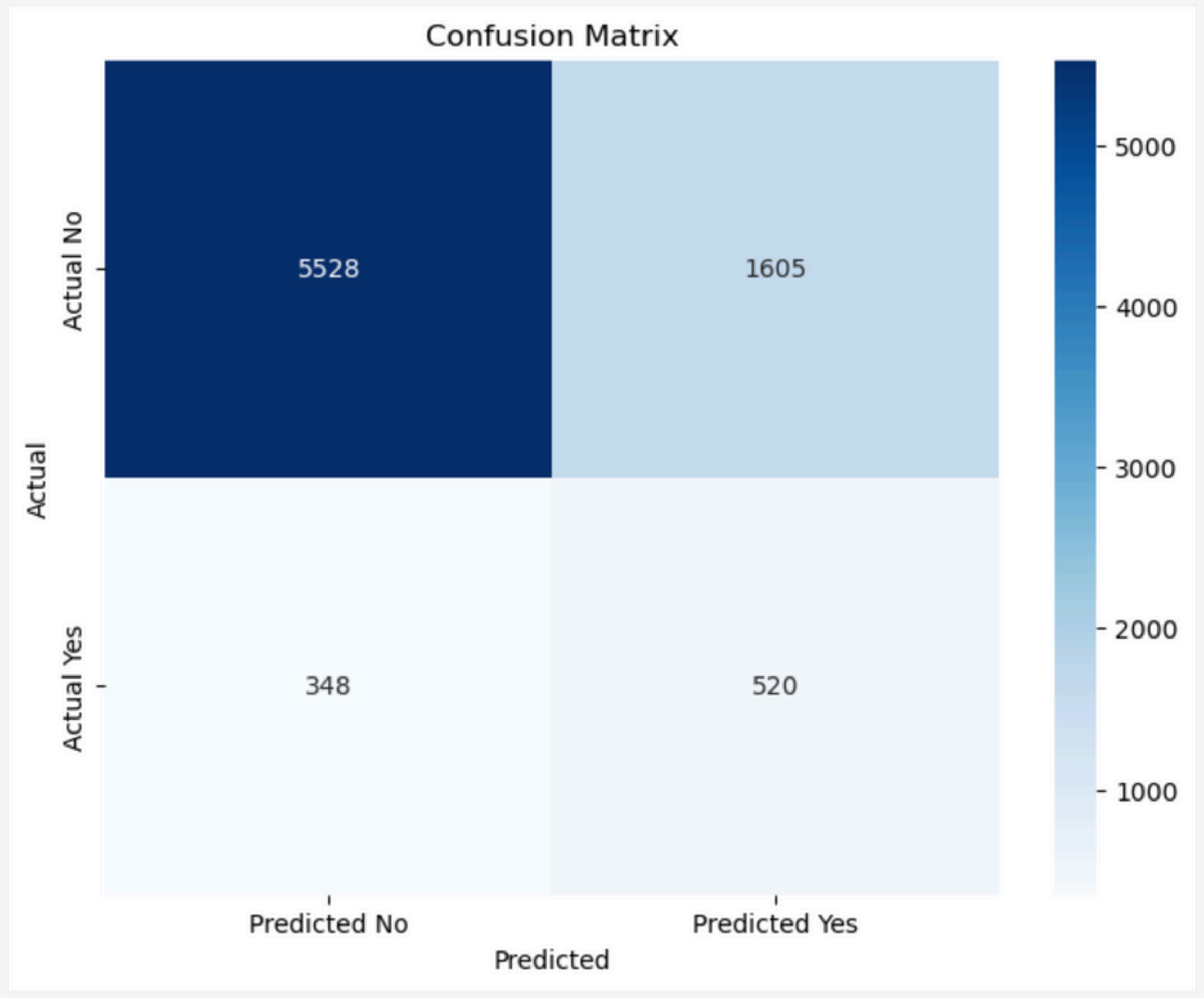
- The main differences from the other versions are the two hidden layers and a dropout of 0.1 and not 0.3

- Overall accuracy 75.59%

# Classification Report and Confusion Matrix

- Similarly to the logistic regression, the neural network is interpretable but struggles with class imbalance.
- The model is able to perform better for the class label 0, but it performs poorly for the class label 1.

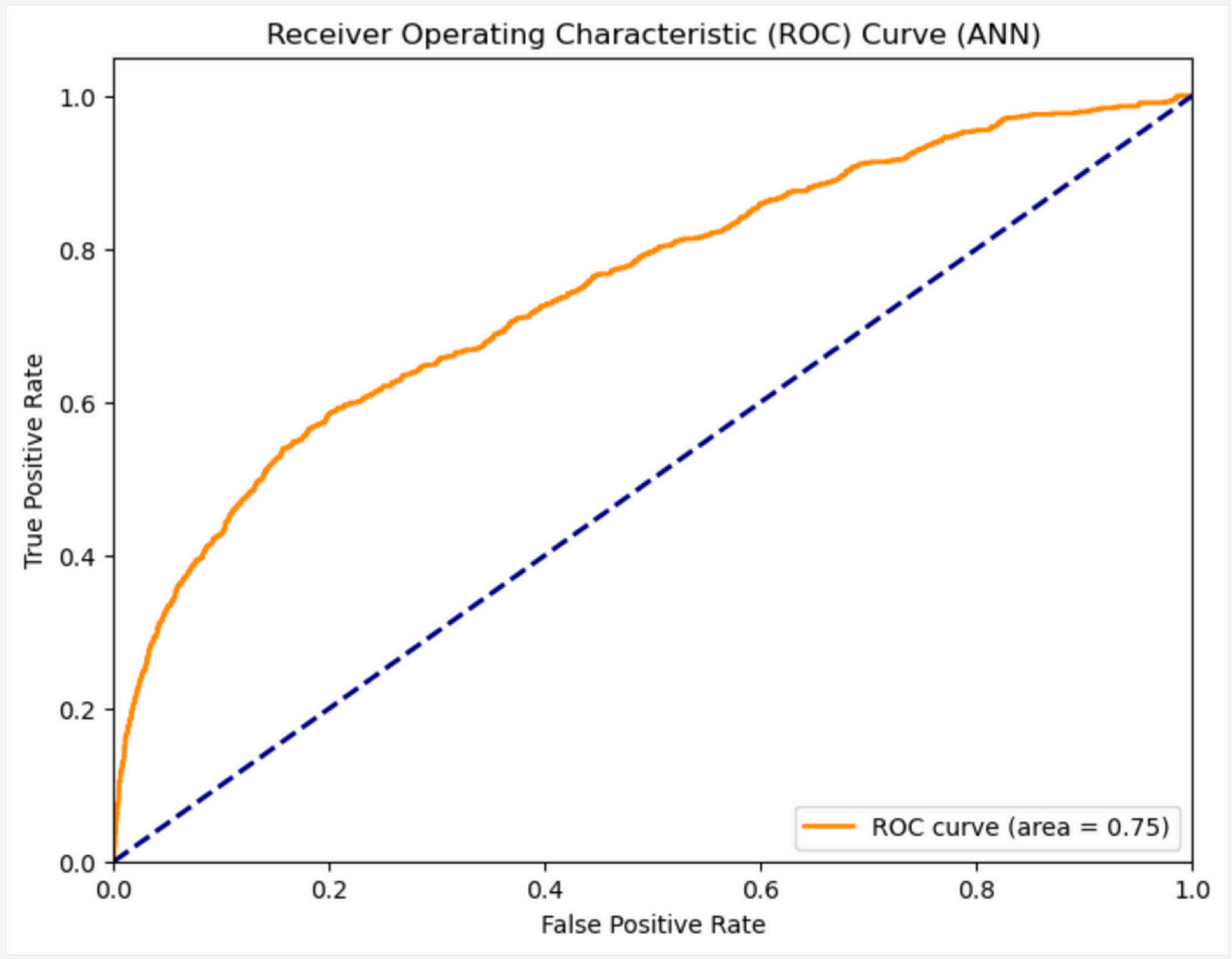
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.94 | 0.77 | 0.85 | 7133 |
| 1 | 0.24 | 0.60 | 0.35 | 868 |
| accuracy | | | 0.76 | 8001 |
| macro avg | 0.59 | 0.69 | 0.60 | 8001 |
| weighted avg | 0.87 | 0.76 | 0.80 | 8001 |



# ROC Curve

- AUC = 0.75 indicates good ability to distinguish between classes but far from optimal.

- Curve Shape:
  - Doesn't sharply rise toward the top-left corner.
  - Shows room for improvement in sensitivity and specificity balance.

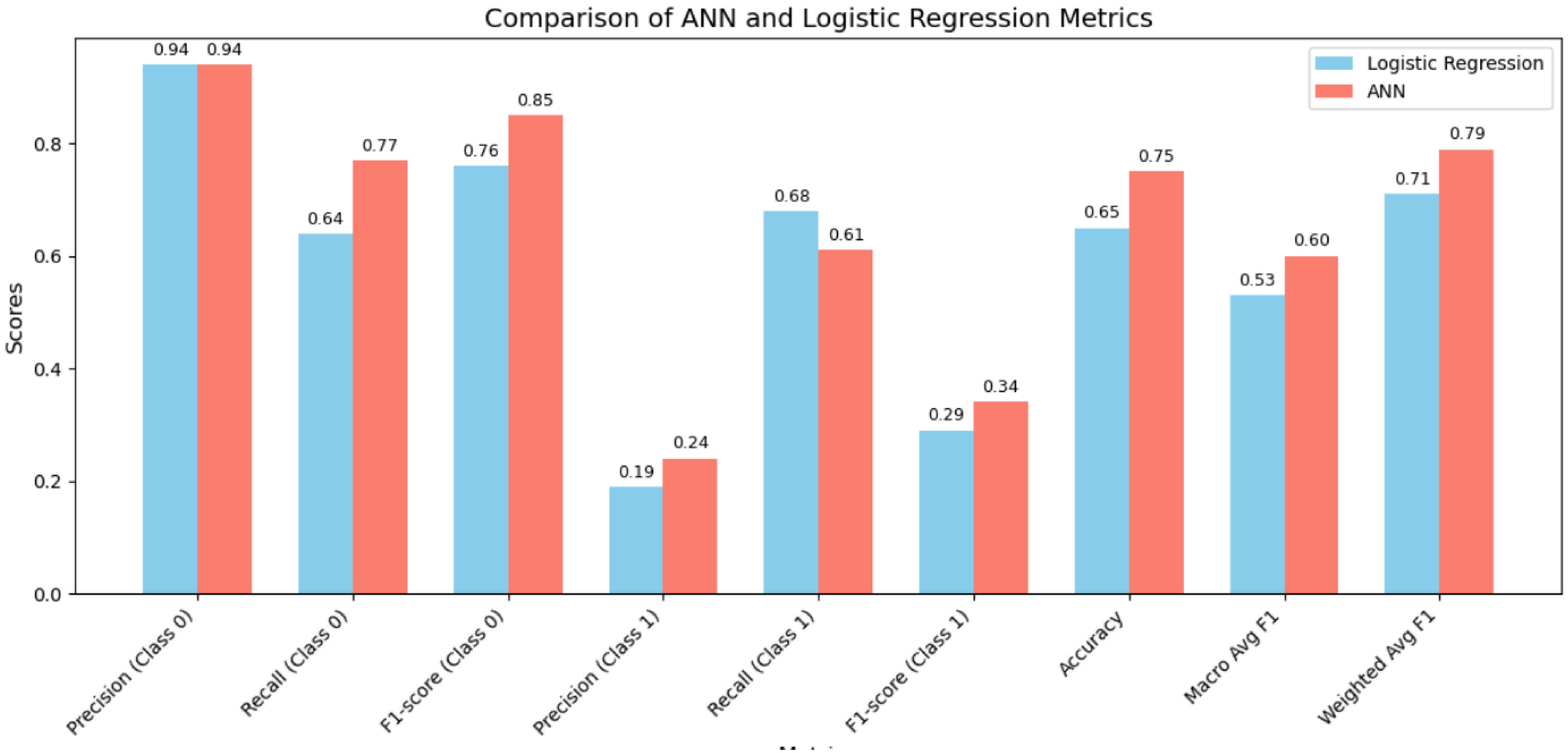


# Logistic Regression VS Ann

# Logistic Regression - ANN

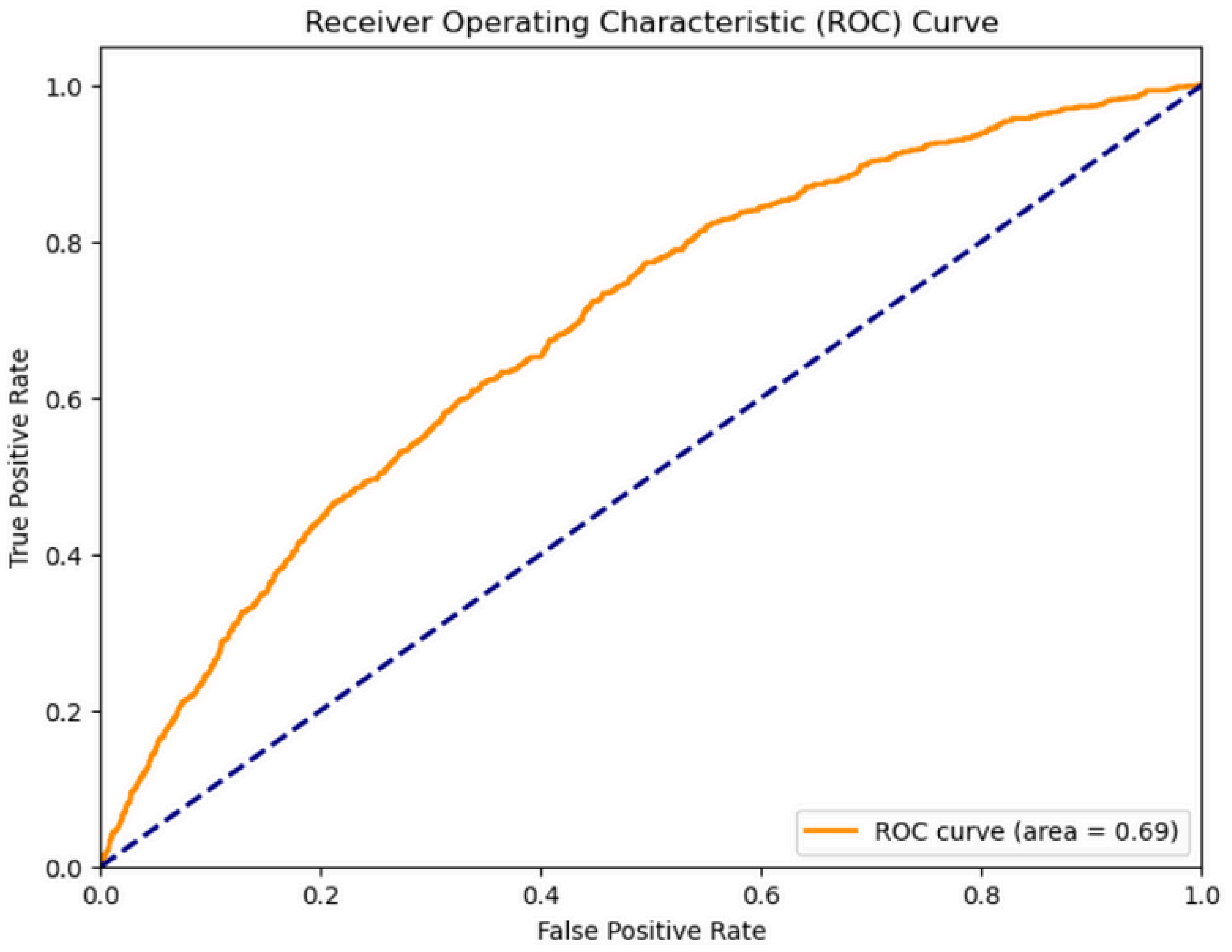
- Accuracy: ANN (75%) vs Logistic Regression (65%).

- F1-scores: ANN (0.79) vs Logistic Regression (0.71).



# Logistic Regression - ANN

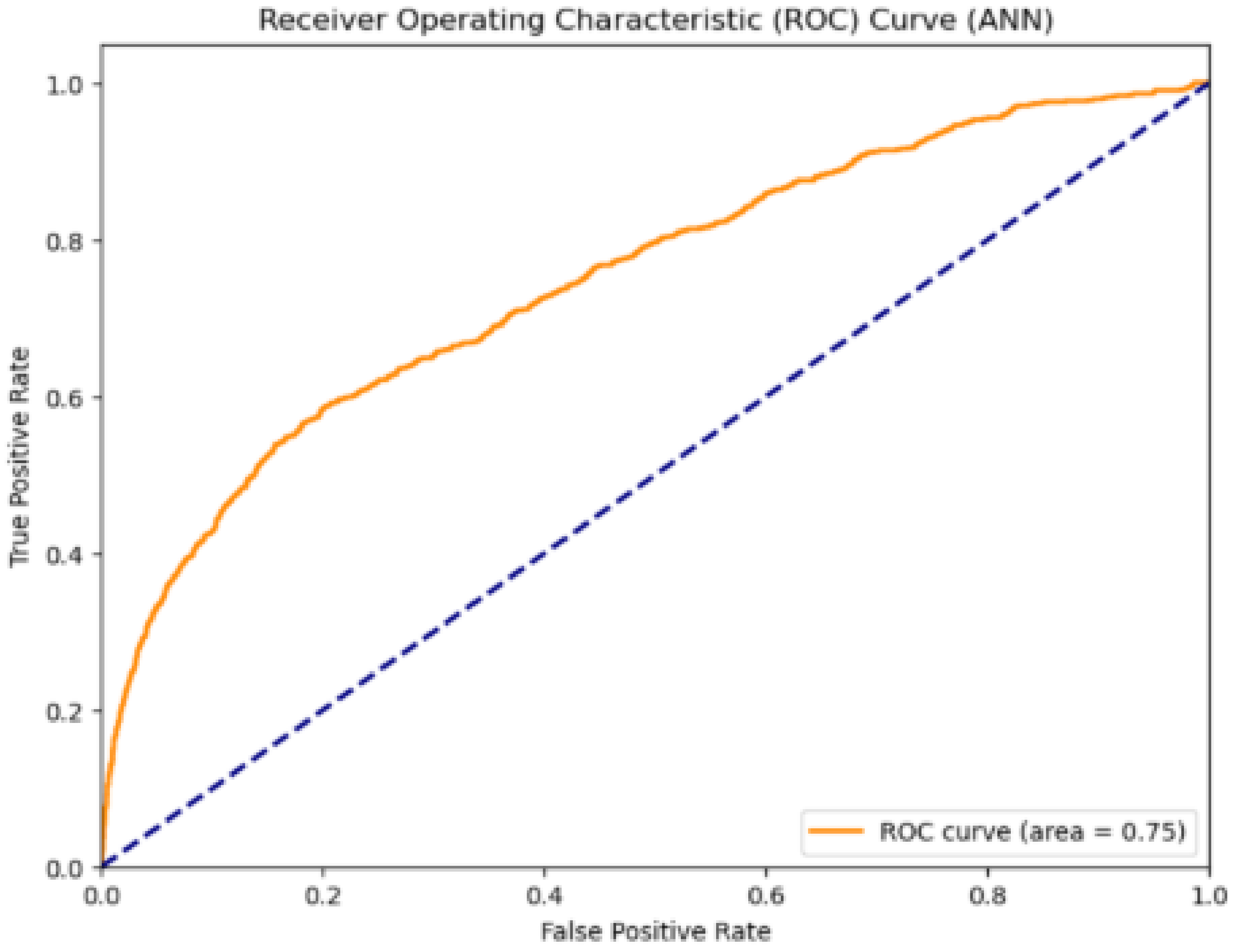
LOGISTIC




- Logistic Regression:
  - Best for straightforward, interpretable problems.
  - Suitable for small datasets with linear relationships.

- Artificial Neural Networks (ANN):
  - Preferred for complex tasks with large datasets.
  - Excels in handling class imbalance and non-linear patterns.

ANN



# THANK YOU!

for your attention