Default Risk Model Training

MediDeFi Project

01

OVERVIEW

OVERVIEW

Dataset

mediDeFi_clean_data.csv

Models Used

Logistic Regression, Random Forest, and XGBoost, compared based on performance.

Target Variable

default_risk : predicts whether a payment will fail (1) or succeed (0).

Project Goal

Identify patients most at risk of payment failure to improve financial prediction in MediDeFi

STEP 1

Load and Inspect Dataset

Loading the Dataset

```
================================================================================
DATASET HEAD
================================================================================
   patient_id  clinic_id  payment_amount  payment_amount_log payment_status  \
0        1102         11          304.66            5.722473      completed
1        1435         10          625.22            6.439702        pending
2        1860         45          541.86            6.296851      completed
3        1270         33           92.54            4.538389      completed
4        1106         36          425.44            6.055472         failed
5        1071         11          221.51            5.404972      completed
6        1700         40          154.72            5.048060      completed
7        1020         23           65.39            4.195546      completed
8        1614         11          250.39            5.527006      completed
9        1121         38          181.75            5.208119      completed

   payment_failed_before  payment_hour  payment_weekday  payment_month  \
0                      0            20                1             10
1                      0            22                6              7
2                      0            19                4              9
3                      0            19                1              6
4                      1             3                0              1
5                      0            23                4              9
6                      0             0                0              7
7                      0            20                0              3
...
6                   395.43                0.156       0.074
7                   459.23                0.057       0.601
8                   300.07                0.139       0.495
9                   157.36                0.035       0.683
```

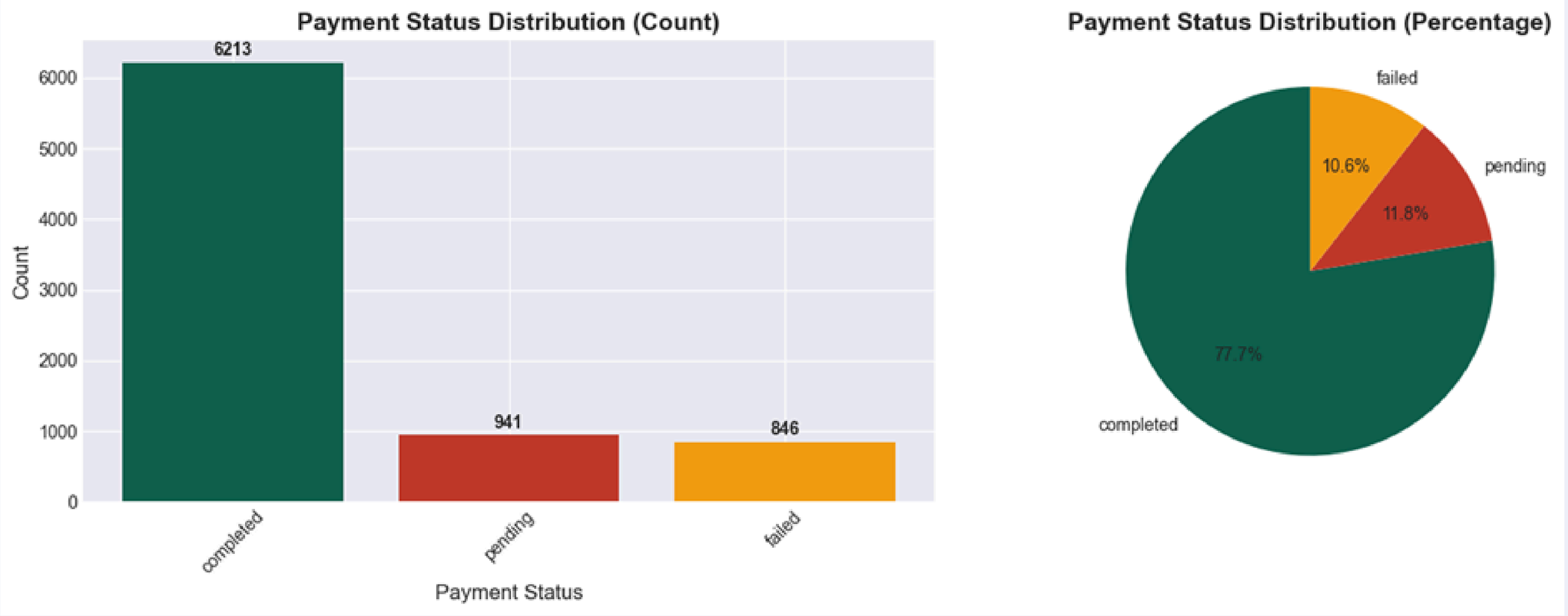
Display dataset shape

```
================================================================================
DATASET SHAPE
================================================================================
Rows: 8000
Columns: 14
Total samples: 8,000
```

Check for missing values

```
================================================================================
MISSING VALUES
================================================================================
No missing values found!
```

PAYMENT STATUS DISTRIBUTION

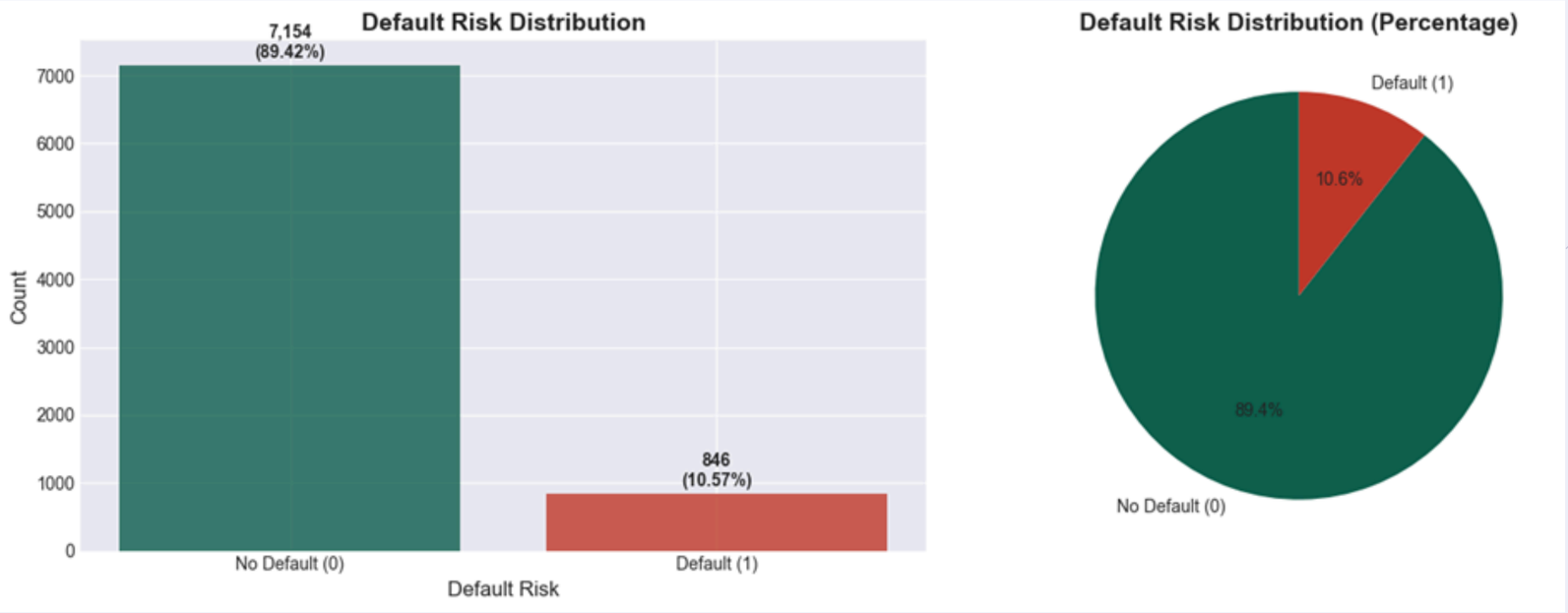


STEP 2

Create the Target Variable (default_risk)

Create the Target Variable (default_risk)

We'll create a binary target variable based on payment status.



Why Our Default Label Definition Makes Sense

Failed Payments = Default (1)

- Represents a true default event where the patient did not complete the payment
- Indicates high financial risk for the system
- Directly affects cash flow, revenue, and system reliability
- These cases are exactly what the model must detect and prevent

Completed Payments = Non-Default (0)

- These transactions are successful and carry no default risk
- Provide strong positive examples for the model
- Help the model learn normal, healthy payment behavior
- Represent stable transactions with no financial threat

Pending Payments = Non-Default (0)

- Payment not finished yet, but not failed
- Cannot be labeled as default since the outcome is still open
- Useful for training: shows the model in-progress behavior
- While pending could fail later, they must be treated as non-default for training consistency

Define features to use in the modal

```
================================================================================
FEATURE SELECTION
================================================================================
Selected features (11):
   1. payment_amount
   2. payment_amount_log
   3. payment_failed_before
   4. payment_hour
   5. payment_weekday
   6. payment_month
   7. patient_total_payments
   8. patient_failed_payments
   9. patient_avg_payment_amount
  10. clinic_default_rate
  11. risk_score
```

STEP 3

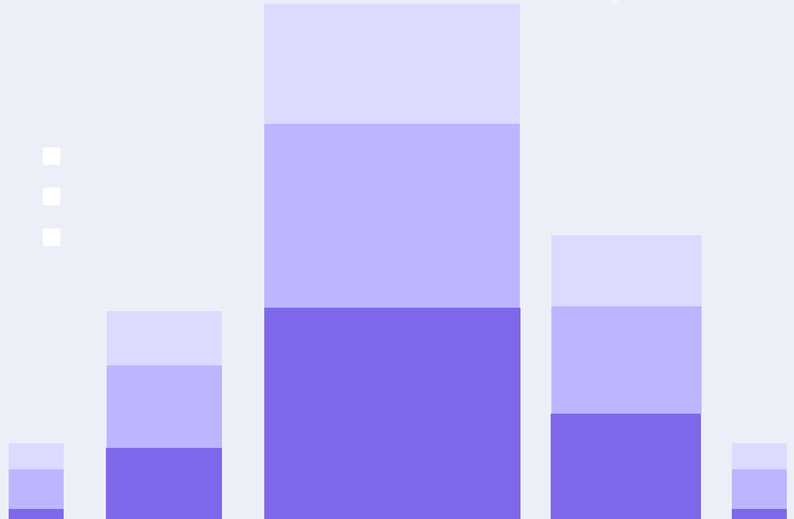
Train/Test Split

Train/Test Split

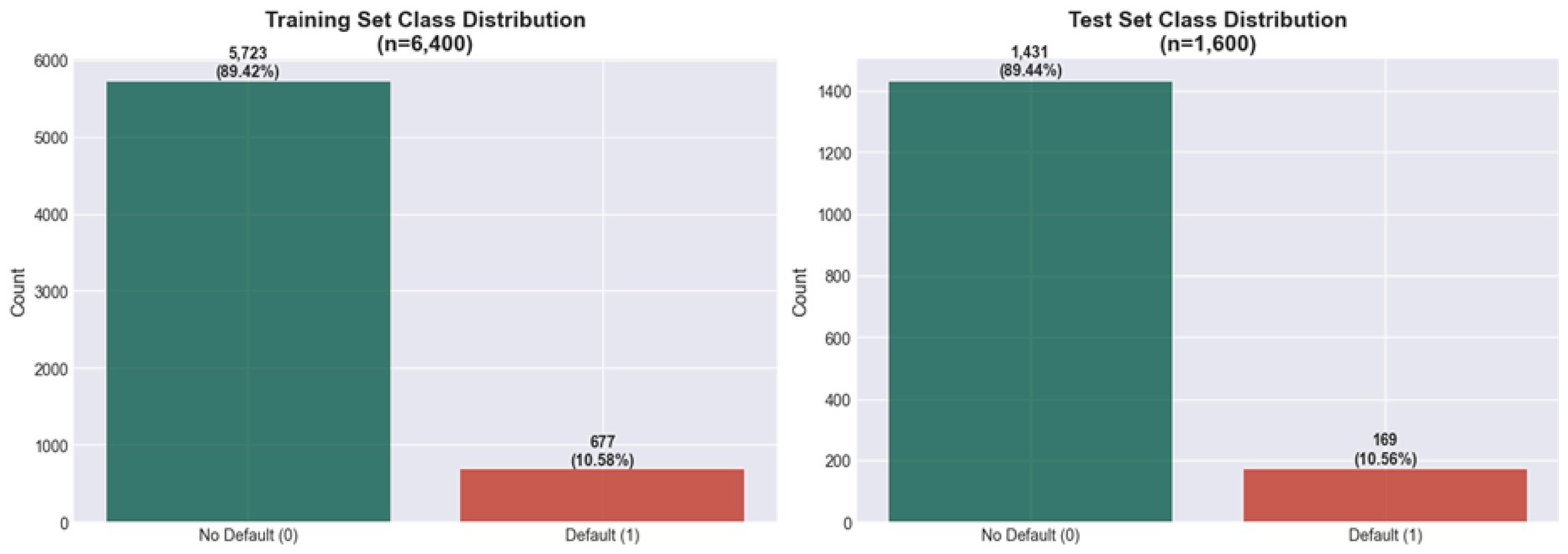
We'll split the data into training and testing sets using stratification to maintain class balance.

```
================================================================================
TRAIN/TEST SPLIT
================================================================================
Training set shape: (6400, 11)
Test set shape: (1600, 11)

Training set size: 6,400 samples (80.0%)
Test set size: 1,600 samples (20.0%)
```



Check class balance in train and test sets



STEP 4

Model 1 — Logistic Regression

Test Set Evaluation

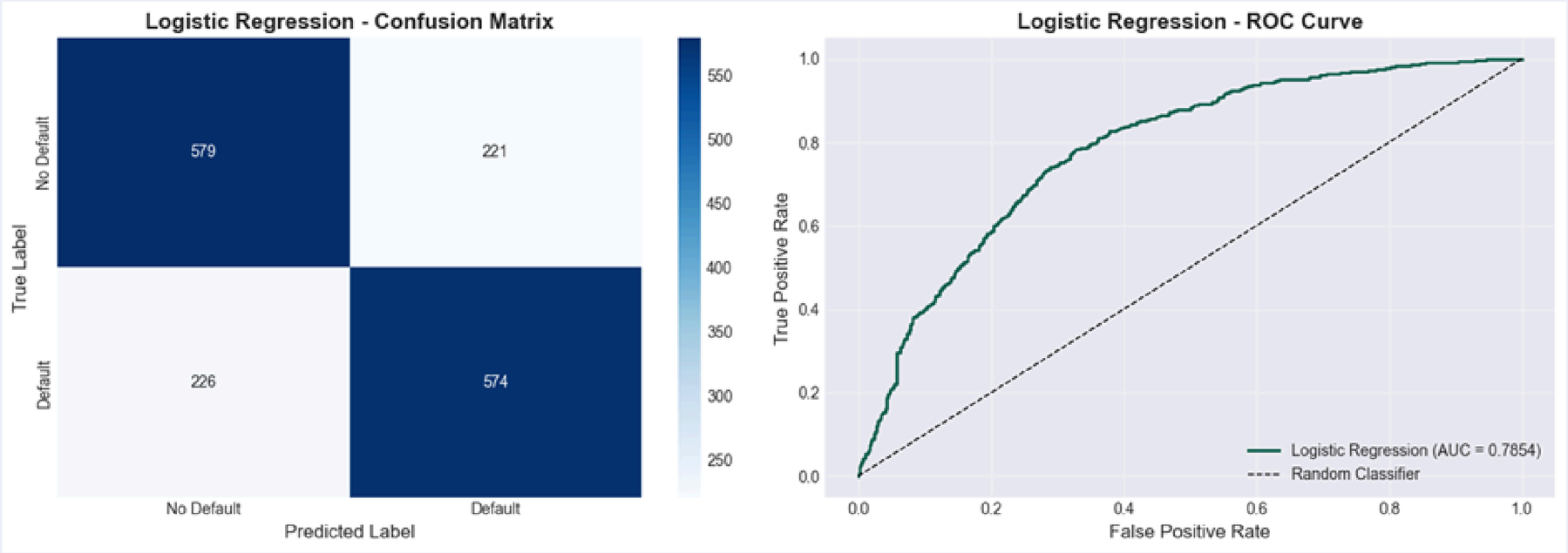
```
================================================================================
LOGISTIC REGRESSION - TEST SET EVALUATION
================================================================================
Accuracy:  0.7206 (72.06%)
Precision: 0.7220 (72.20%)
Recall:    0.7175 (71.75%)
F1 Score:  0.7197 (71.97%)
ROC-AUC:   0.7854 (78.54%)

================================================================================
CLASSIFICATION REPORT
================================================================================
              precision    recall  f1-score   support

  No Default       0.72      0.72      0.72       800
     Default       0.72      0.72      0.72       800

    accuracy                           0.72      1600
   macro avg       0.72      0.72      0.72      1600
weighted avg       0.72      0.72      0.72      1600
```

Test Set Evaluation



STEP 5

Model 2— Random Forest

Test Set Evaluation

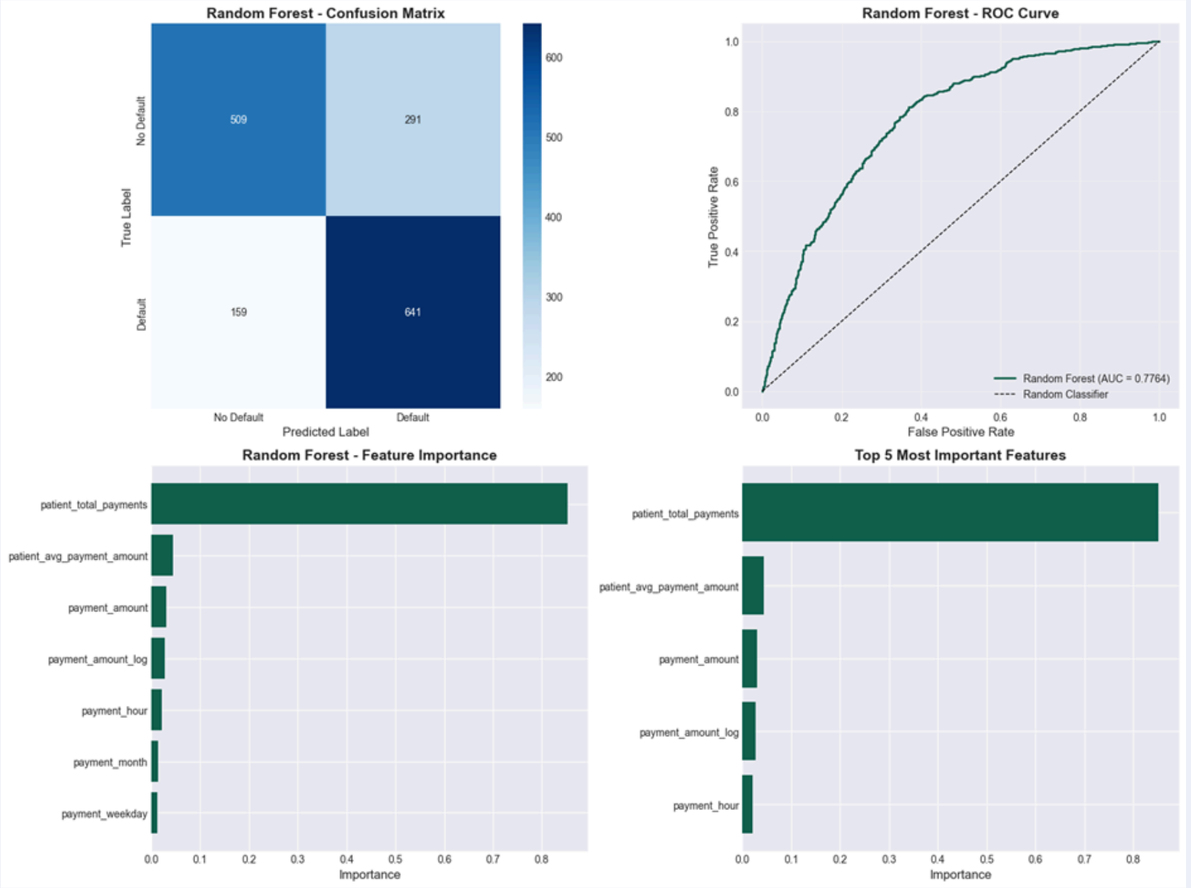
```
================================================================================
RANDOM FOREST - TEST SET EVALUATION
================================================================================
Accuracy:  0.7188 (71.88%)
Precision: 0.6878 (68.78%)
Recall:    0.8013 (80.12%)
F1 Score:  0.7402 (74.02%)
ROC-AUC:   0.7764 (77.64%)

================================================================================
CLASSIFICATION REPORT
================================================================================
              precision    recall  f1-score   support

  No Default       0.76      0.64      0.69       800
     Default       0.69      0.80      0.74       800

    accuracy                           0.72      1600
   macro avg       0.72      0.72      0.72      1600
weighted avg       0.72      0.72      0.72      1600
```

Test Set Evaluation



STEP 6

Model 3 — XGBoost

Test Set Evaluation

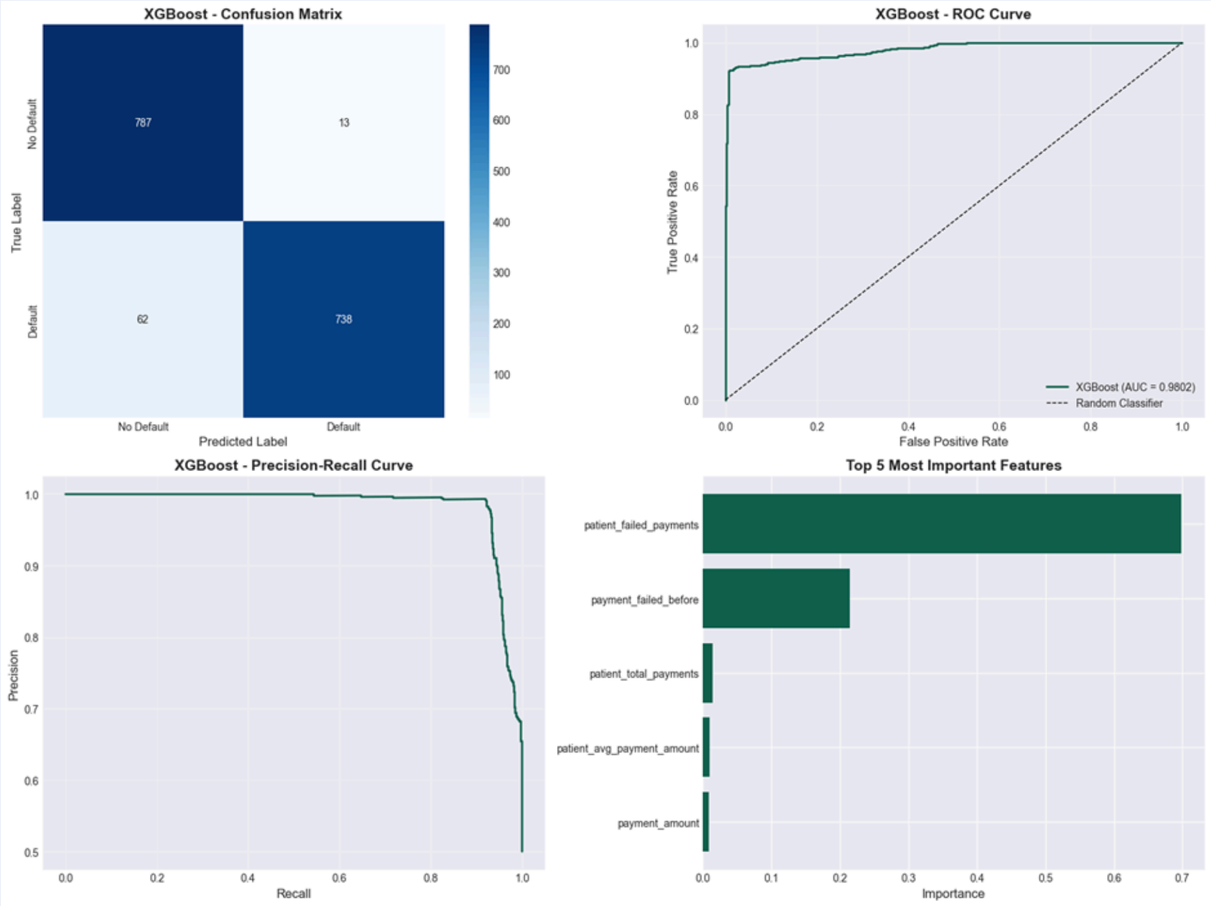
```
================================================================================
XGBOOST - TEST SET EVALUATION
================================================================================
Accuracy:  0.9531 (95.31%)
Precision: 0.9827 (98.27%)
Recall:    0.9225 (92.25%)
F1 Score:  0.9516 (95.16%)
ROC-AUC:   0.9802 (98.02%)

================================================================================
CLASSIFICATION REPORT
================================================================================
              precision    recall  f1-score   support

  No Default       0.93      0.98      0.95       800
     Default       0.98      0.92      0.95       800

    accuracy                           0.95      1600
   macro avg       0.95      0.95      0.95      1600
weighted avg       0.95      0.95      0.95      1600
```

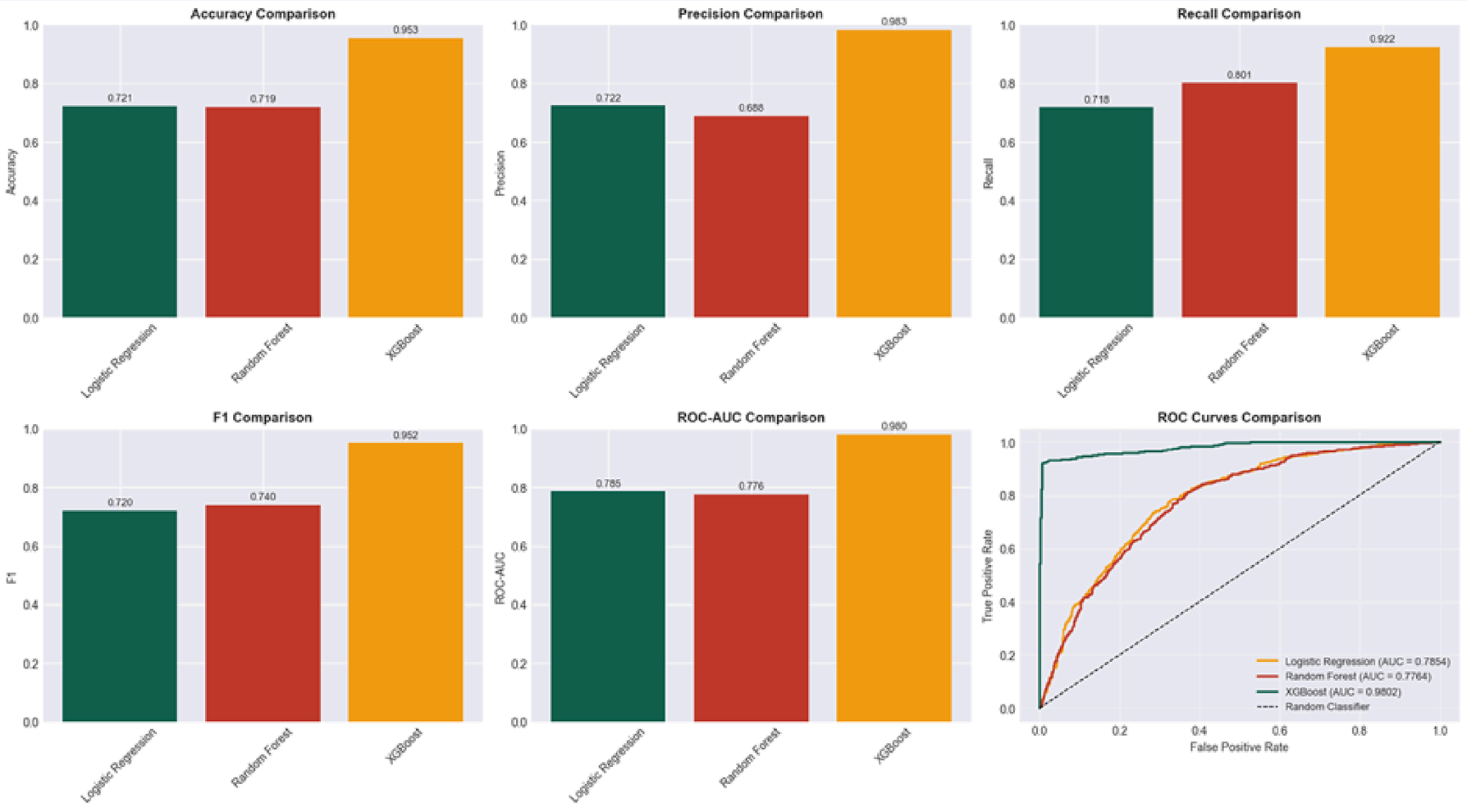
Test Set Evaluation



STEP 7

Model Comparison Table

Test Set Evaluation



Limitations of Logistic Regression

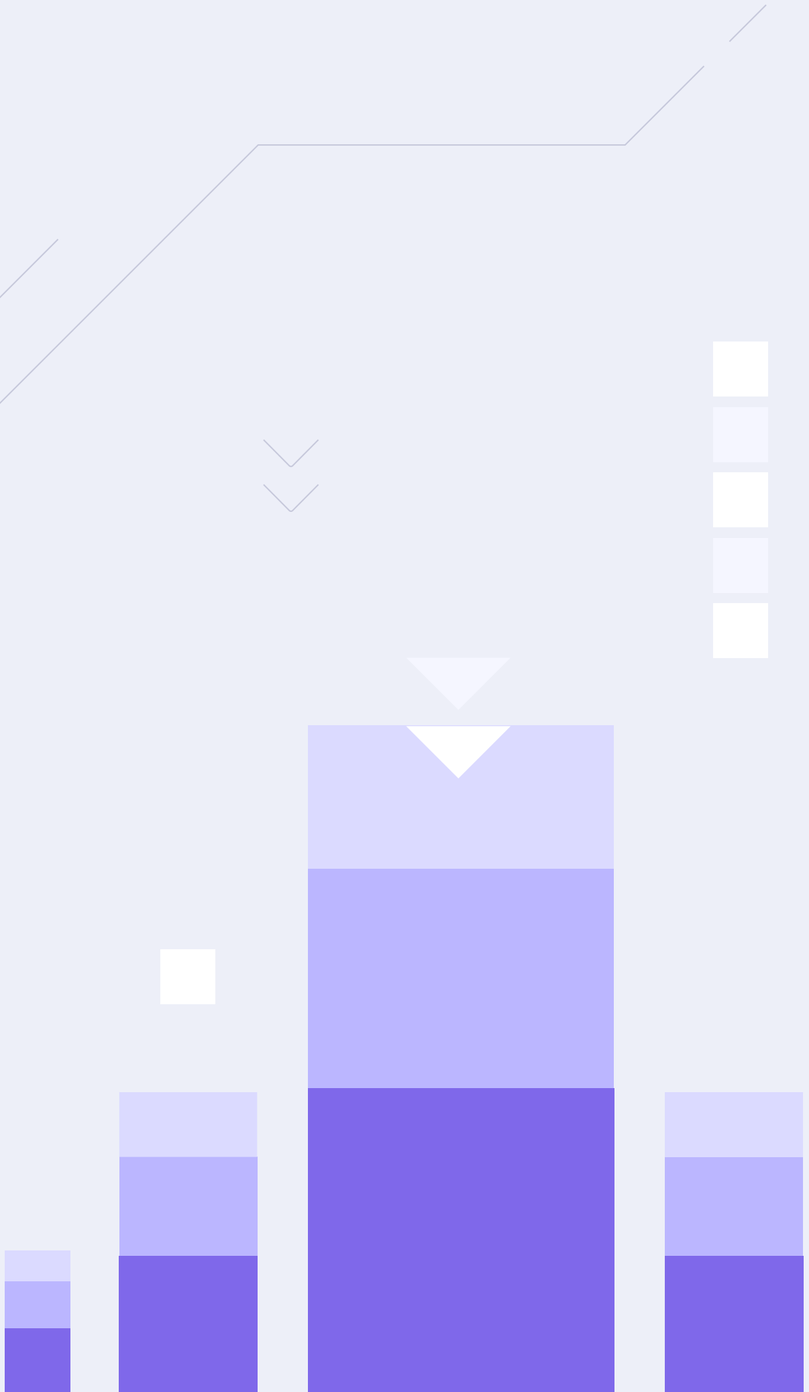
- It assumes relationships are straight and linear, but real payment behavior is not.
- It cannot capture complex patterns between time, behavior, and risk.
- It misses many default cases, giving low recall.
- With only a few simple features, it cannot learn deep risk signals.

Why Random Forest Is NOT Chosen

- It captures some non-linear patterns, but struggles on minority cases like defaults.
- Performance becomes less stable on imbalanced data, unlike XGBoost.
- Its probability estimates are less reliable, making it weaker for risk scoring.
- With restricted settings (fewer trees, shallow depth), it cannot learn the more subtle risk patterns in MediDeFi.

Why XGBoost Is Chosen

- It provides the best overall balance of accuracy, recall, and ROC-AUC
- It learns subtle patterns in patient and clinic behavior thanks to gradient boosting.
- It handles imbalanced data better than the other models through weighted loss.
- It is more stable and robust because its regularization (L1/L2) reduces overfitting.
- It is an industry standard for financial risk, fraud, and default prediction.



Thanks!