Predicting Mortality based on Heart Failure Patients

This project focuses on predicting mortality outcomes among heart failure patients using a combination of machine learning algorithms. The dataset undergoes preprocessing followed by training and evaluation using several classification models:

> The module is divided into the following steps:

1. Import Libraries
2. Load Dataset
3. Preprocessing
4. Support Vector Machine(SVM)
5. Decision Tree
6. Gaussian Naive Bayes (GNB)
7. Random Forest Classification (RF)
8. XGBoost Classification (XGB)
9. AdaBoost Classification
10. Artificial Neural Network (ANN)

Extras

1. Finding the worst Classification Algorithm using ROC AUC Score
2. Sampling using SMOTE with the worst Classification Algorithm
3. Using Explainable AI (Lime) to explain why one specific prediction was made (using misclassified sample).

```
!pip install imbalanced-learn lime xgboost

```

```
Requirement already satisfied: imbalanced-learn in /usr/local/lib/python3.11/dist-packages (0.13.0)
Requirement already satisfied: lime in /usr/local/lib/python3.11/dist-packages (0.2.0.1)
Requirement already satisfied: xgboost in /usr/local/lib/python3.11/dist-packages (2.1.4)
Requirement already satisfied: numpy<3,>=1.24.3 in /usr/local/lib/python3.11/dist-packages (from imbalanced-learn) (2.0.2)
Requirement already satisfied: scipy<2,>=1.10.1 in /usr/local/lib/python3.11/dist-packages (from imbalanced-learn) (1.15.3)
Requirement already satisfied: scikit-learn<2,>=1.3.2 in /usr/local/lib/python3.11/dist-packages (from imbalanced-learn) (1.6.1)
Requirement already satisfied: sklearn-compat<1,>=0.1 in /usr/local/lib/python3.11/dist-packages (from imbalanced-learn) (0.1.3)
Requirement already satisfied: joblib<2,>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from imbalanced-learn) (1.5.1)
Requirement already satisfied: threadpoolctl<4,>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from imbalanced-learn) (3.6.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (from lime) (3.10.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from lime) (4.67.1)
Requirement already satisfied: scikit-image>=0.12 in /usr/local/lib/python3.11/dist-packages (from lime) (0.25.2)
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.11/dist-packages (from xgboost) (2.21.5)
Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.11/dist-packages (from scikit-image>=0.12->lime) (3.5)
Requirement already satisfied: pillow>=10.1 in /usr/local/lib/python3.11/dist-packages (from scikit-image>=0.12->lime) (11.2.1)
Requirement already satisfied: imageio!=2.35.0,>=2.33 in /usr/local/lib/python3.11/dist-packages (from scikit-image>=0.12->lime) (2.37.0)
Requirement already satisfied: tifffile>=2022.8.12 in /usr/local/lib/python3.11/dist-packages (from scikit-image>=0.12->lime) (2025.6.11)
Requirement already satisfied: packaging>=21 in /usr/local/lib/python3.11/dist-packages (from scikit-image>=0.12->lime) (24.2)
Requirement already satisfied: lazy-loader>=0.4 in /usr/local/lib/python3.11/dist-packages (from scikit-image>=0.12->lime) (0.4)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->lime) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib->lime) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->lime) (4.58.4)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->lime) (1.4.8)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->lime) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib->lime) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib->lime) (1.17.0)
```

```
# Step 1: Importing necessary libraries

# Data and Preprocessing
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, StratifiedKFold, cross_val_score
from sklearn.preprocessing import StandardScaler

# Handling Imbalanced Data
from imblearn.over_sampling import SMOTE

# Models
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
import xgboost as xgb

# Evaluation
from sklearn.metrics import classification_report, roc_auc_score

# ANN
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras import callbacks

# Explainable AI
import shap
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")

# Set random seed for reproducibility
np.random.seed(42)

```

```
# Step 2 : Load the dataset and check the values
df = pd.read_csv("heart_failure_clinical_records.csv")
df.info()
```

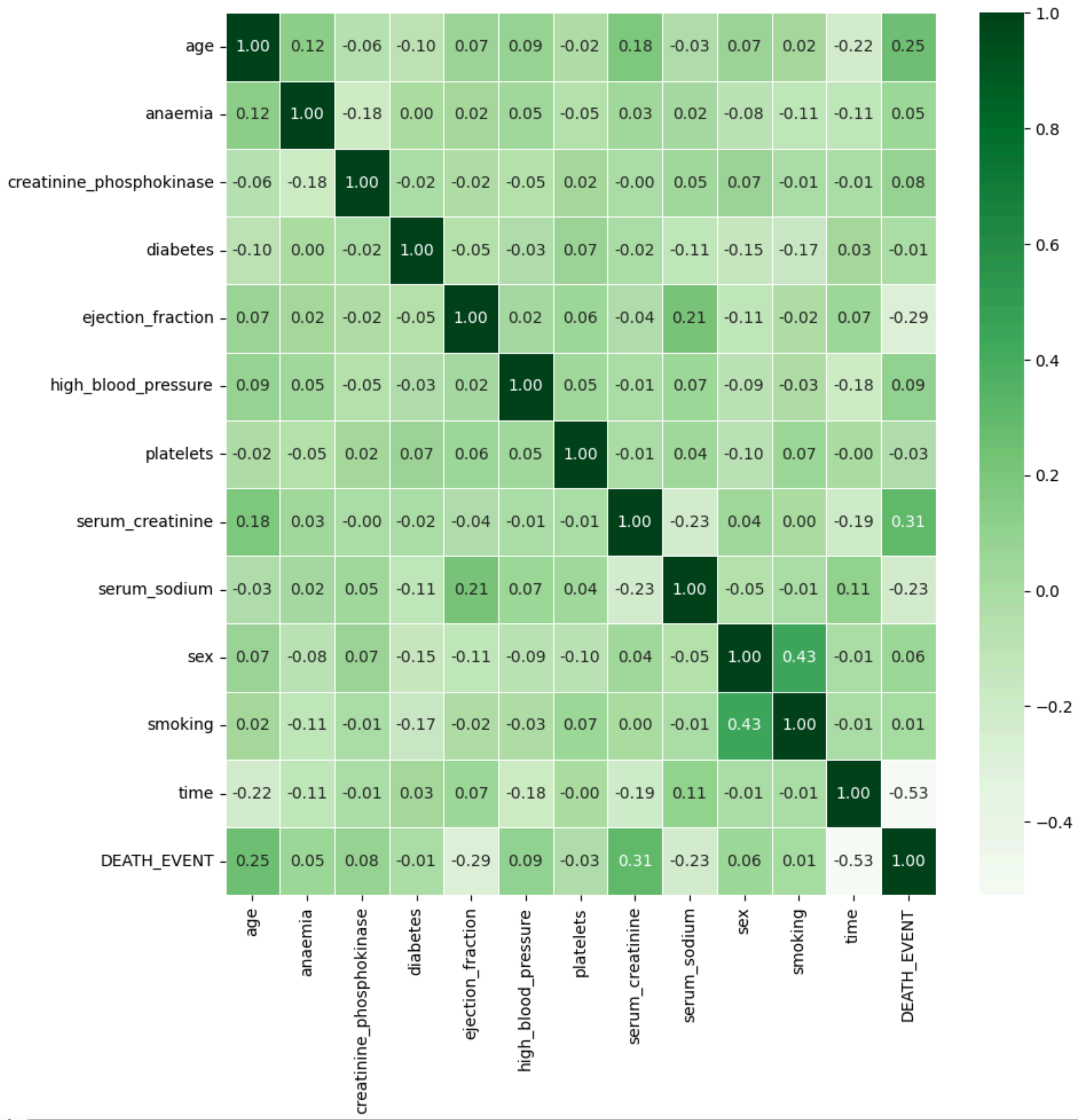
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   age                       5000 non-null   float64
 1   anaemia                   5000 non-null   int64
 2   creatinine_phosphokinase  5000 non-null   int64
 3   diabetes                  5000 non-null   int64
 4   ejection_fraction         5000 non-null   int64
 5   high_blood_pressure       5000 non-null   int64
 6   platelets                 5000 non-null   float64
 7   serum_creatinine          5000 non-null   float64
 8   serum_sodium              5000 non-null   int64
 9   sex                       5000 non-null   int64
 10  smoking                   5000 non-null   int64
 11  time                      5000 non-null   int64
```

```
 12  DEATH_EVENT               5000 non-null   int64
dtypes: float64(3), int64(10)
memory usage: 507.9 KB
```

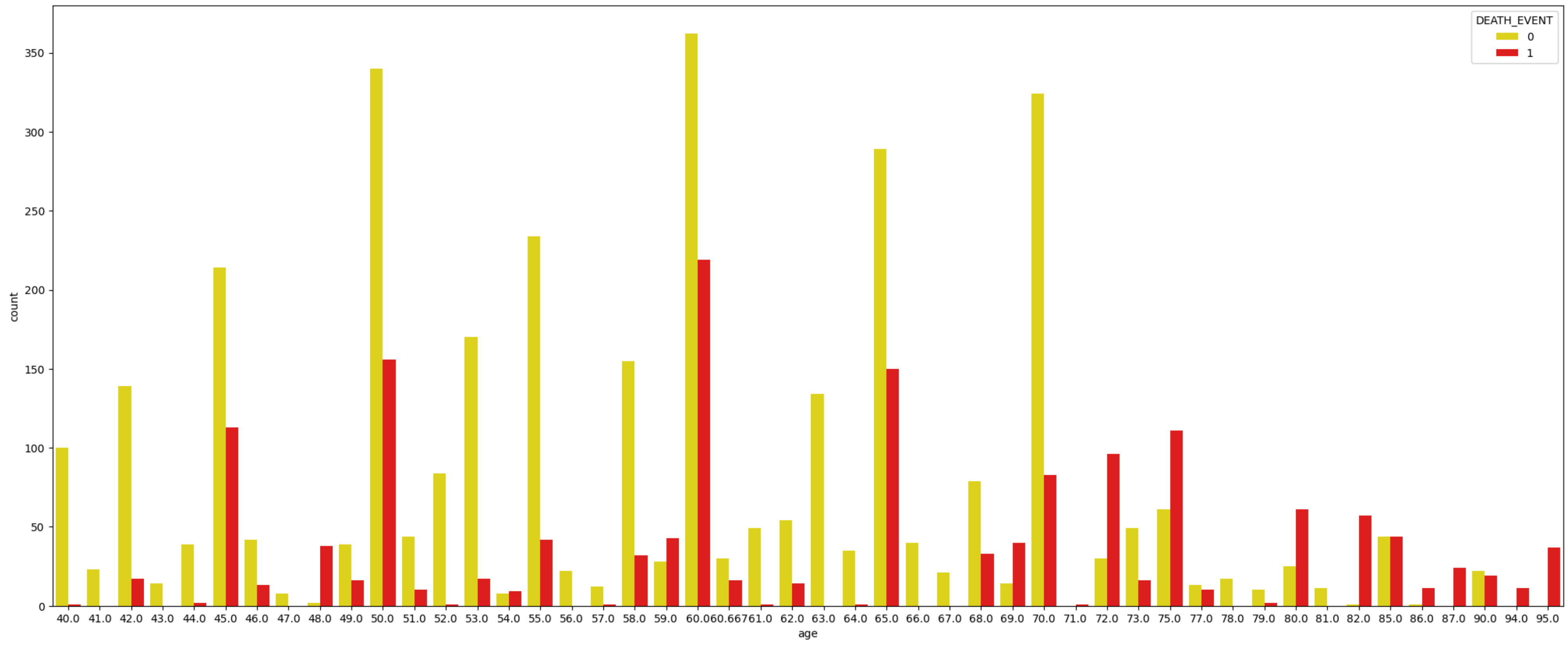
```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
age	5000.0	60.288736	11.697243	40.0	50.0	60.00	68.0	95.0
anaemia	5000.0	0.474400	0.499394	0.0	0.0	0.00	1.0	1.0
creatinine_phosphokinase	5000.0	586.760600	976.733979	23.0	121.0	248.00	582.0	7861.0
diabetes	5000.0	0.439400	0.496364	0.0	0.0	0.00	1.0	1.0
ejection_fraction	5000.0	37.734600	11.514855	14.0	30.0	38.00	45.0	80.0
high_blood_pressure	5000.0	0.364800	0.481422	0.0	0.0	0.00	1.0	1.0
platelets	5000.0	265075.404370	97999.758622	25100.0	215000.0	263358.03	310000.0	850000.0
serum_creatinine	5000.0	1.369106	1.009750	0.5	0.9	1.10	1.4	9.4
serum_sodium	5000.0	136.808200	4.464236	113.0	134.0	137.00	140.0	148.0
sex	5000.0	0.645600	0.478379	0.0	0.0	1.00	1.0	1.0
smoking	5000.0	0.311800	0.463275	0.0	0.0	0.00	1.0	1.0
time	5000.0	130.678800	77.325928	4.0	74.0	113.00	201.0	285.0
DEATH_EVENT	5000.0	0.313600	0.464002	0.0	0.0	0.00	1.0	1.0

```
plt.subplots(figsize=(10,10))
sns.heatmap(df.corr(),annot=True,cmap='Greens', fmt=".2f", linewidths=0.5, cbar=True)
plt.show()
```




```
cols = ['#FFF000', '#FF0000']
plt.figure(figsize=(25,10))
days_of_week = sns.countplot(x="age", data=df, hue = 'DEATH_EVENT', palette = cols)
plt.show()
```




```
# Step 3: Data Preprocessing
X = df.drop('DEATH_EVENT', axis=1)
y = df['DEATH_EVENT']

# Hold out a final test set
X_trainval, X_finaltest, y_trainval, y_finaltest = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)

# Feature scaling
scaler = StandardScaler()
X_trainval_scaled = scaler.fit_transform(X_trainval)
X_finaltest_scaled = scaler.transform(X_finaltest)
```

Support Vector Machine

```
svm_model = SVC(probability=True, kernel='rbf', random_state=42)
svm_model.fit(X_trainval_scaled, y_trainval)

svm_preds = svm_model.predict(X_finaltest_scaled)
svm_probs = svm_model.predict_proba(X_finaltest_scaled)[:, 1]

print("===== SVM Classification Report =====")
print(classification_report(y_finaltest, svm_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, svm_probs))
```

```
===== SVM Classification Report =====
              precision    recall  f1-score   support

           0       0.96      0.97      0.97       686
           1       0.94      0.92      0.93       314

    accuracy                           0.96      1000
   macro avg       0.95      0.95      0.95      1000
weighted avg       0.96      0.96      0.96      1000

ROC AUC Score: 0.9818759168817663
```

Decision Tree Classifier

```
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_trainval_scaled, y_trainval)

dt_preds = dt_model.predict(X_finaltest_scaled)
dt_probs = dt_model.predict_proba(X_finaltest_scaled)[:, 1]

print("\n===== Decision Tree Classification Report =====")
print(classification_report(y_finaltest, dt_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, dt_probs))
```

```
===== Decision Tree Classification Report =====
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       686
           1       0.98      0.97      0.97       314

    accuracy                           0.98      1000
   macro avg       0.98      0.98      0.98      1000
weighted avg       0.98      0.98      0.98      1000

ROC AUC Score: 0.9795871942953706
```

Gaussian Naive Bayes

```
gnb_model = GaussianNB()
gnb_model.fit(X_trainval_scaled, y_trainval)

gnb_preds = gnb_model.predict(X_finaltest_scaled)
gnb_probs = gnb_model.predict_proba(X_finaltest_scaled)[:, 1]

print("\n===== Gaussian Naive Bayes Classification Report =====")
```

```
print(classification_report(y_finaltest, gnb_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, gnb_probs))

```

```
===== Gaussian Naive Bayes Classification Report =====
              precision    recall  f1-score   support

           0       0.81      0.92      0.86       686
           1       0.74      0.52      0.61       314

    accuracy                           0.79      1000
   macro avg       0.77      0.72      0.73      1000
weighted avg       0.78      0.79      0.78      1000

ROC AUC Score: 0.884816437949156
```

Random Forest Classification

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_trainval_scaled, y_trainval)

rf_preds = rf_model.predict(X_finaltest_scaled)
rf_probs = rf_model.predict_proba(X_finaltest_scaled)[:, 1]

print("\n===== Random Forest Classification Report =====")
print(classification_report(y_finaltest, rf_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, rf_probs))

```

```
===== Random Forest Classification Report =====
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       686
           1       1.00      0.97      0.99       314

    accuracy                           0.99      1000
   macro avg       0.99      0.99      0.99      1000
weighted avg       0.99      0.99      0.99      1000

ROC AUC Score: 0.9998769753579321
```

XGBoost Classification

```
xgb_model = xgb.XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_state=42)
xgb_model.fit(X_trainval_scaled, y_trainval)

xgb_preds = xgb_model.predict(X_finaltest_scaled)
xgb_probs = xgb_model.predict_proba(X_finaltest_scaled)[:, 1]

print("\n===== XGBoost Classification Report =====")
print(classification_report(y_finaltest, xgb_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, xgb_probs))

```

```
===== XGBoost Classification Report =====
              precision    recall  f1-score   support

           0       0.99      1.00      1.00       686
           1       0.99      0.99      0.99       314

    accuracy                           0.99      1000
   macro avg       0.99      0.99      0.99      1000
weighted avg       0.99      0.99      0.99      1000

ROC AUC Score: 0.9998467995023306
```

AdaBoost Classification

```
ada_model = AdaBoostClassifier(n_estimators=100, random_state=42)
ada_model.fit(X_trainval_scaled, y_trainval)

ada_preds = ada_model.predict(X_finaltest_scaled)
ada_probs = ada_model.predict_proba(X_finaltest_scaled)[:, 1]

print("\n===== AdaBoost Classification Report =====")
print(classification_report(y_finaltest, ada_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, ada_probs))

```

```
===== AdaBoost Classification Report =====
              precision    recall  f1-score   support

           0       0.90      0.94      0.92       686
           1       0.86      0.76      0.81       314

    accuracy                           0.89      1000
   macro avg       0.88      0.85      0.87      1000
weighted avg       0.89      0.89      0.89      1000

ROC AUC Score: 0.962178046832928
```

Artificial Neural Network

```
# Early stopping to avoid overfitting
early_stopping = callbacks.EarlyStopping(
    min_delta=0.001,
    patience=30,
    verbose=1,
    restore_best_weights=True
)

# Define the model
ANN_model = Sequential()
ANN_model.add(Dense(32, input_dim=12, activation='relu', kernel_initializer='he_uniform'))
ANN_model.add(Dense(8, activation='relu', kernel_initializer='he_uniform'))
ANN_model.add(Dropout(0.25))
ANN_model.add(Dense(8, activation='relu', kernel_initializer='he_uniform'))
ANN_model.add(Dropout(0.25))
```

```
ANN_model.add(Dense(1, activation='sigmoid', kernel_initializer='glorot_uniform'))

```

```
ANN_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
ANN_model.summary()
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 32)	416
dense_5 (Dense)	(None, 8)	264
dropout_2 (Dropout)	(None, 8)	0
dense_6 (Dense)	(None, 8)	72
dropout_3 (Dropout)	(None, 8)	0
dense_7 (Dense)	(None, 1)	9

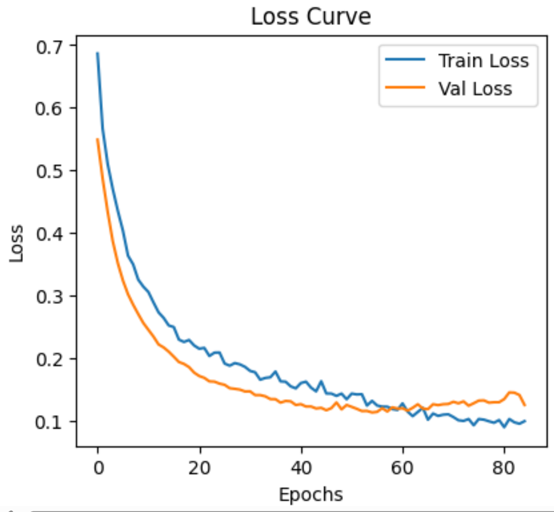
Total params: 761 (2.97 KB)

```
history = ANN_model.fit(
    X_trainval_scaled,
    y_trainval,
    validation_split=0.25,
    epochs=100,
    batch_size=25,
    callbacks=[early_stopping],
    verbose=1
)
```

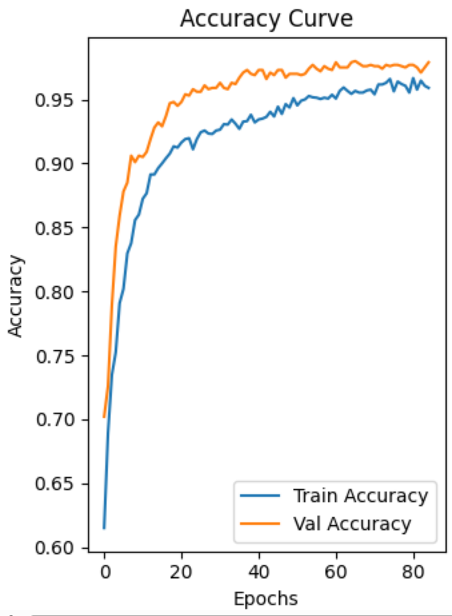
```
Epoch 58/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9518 - loss: 0.1286 - val_accuracy: 0.9760 - val_loss: 0.1141
Epoch 59/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.9536 - loss: 0.1216 - val_accuracy: 0.9740 - val_loss: 0.1208
Epoch 60/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.9527 - loss: 0.1125 - val_accuracy: 0.9730 - val_loss: 0.1188
Epoch 61/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.9526 - loss: 0.1210 - val_accuracy: 0.9790 - val_loss: 0.1192
Epoch 62/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.9603 - loss: 0.1243 - val_accuracy: 0.9750 - val_loss: 0.1154
Epoch 63/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9582 - loss: 0.1096 - val_accuracy: 0.9750 - val_loss: 0.1200
Epoch 64/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9527 - loss: 0.1217 - val_accuracy: 0.9750 - val_loss: 0.1256
Epoch 65/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9585 - loss: 0.1075 - val_accuracy: 0.9790 - val_loss: 0.1189
Epoch 66/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9552 - loss: 0.1058 - val_accuracy: 0.9800 - val_loss: 0.1179
Epoch 67/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9578 - loss: 0.0993 - val_accuracy: 0.9780 - val_loss: 0.1257
Epoch 68/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9552 - loss: 0.1024 - val_accuracy: 0.9760 - val_loss: 0.1243
Epoch 69/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9605 - loss: 0.0986 - val_accuracy: 0.9760 - val_loss: 0.1257
Epoch 70/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9526 - loss: 0.1212 - val_accuracy: 0.9770 - val_loss: 0.1261
Epoch 71/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9551 - loss: 0.1006 - val_accuracy: 0.9760 - val_loss: 0.1297
Epoch 72/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9545 - loss: 0.1104 - val_accuracy: 0.9760 - val_loss: 0.1272
Epoch 73/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9638 - loss: 0.0962 - val_accuracy: 0.9740 - val_loss: 0.1303
Epoch 74/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9663 - loss: 0.0965 - val_accuracy: 0.9770 - val_loss: 0.1233
Epoch 75/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9702 - loss: 0.0863 - val_accuracy: 0.9760 - val_loss: 0.1273
Epoch 76/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9524 - loss: 0.1028 - val_accuracy: 0.9770 - val_loss: 0.1315
Epoch 77/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9653 - loss: 0.1042 - val_accuracy: 0.9770 - val_loss: 0.1319
Epoch 78/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9638 - loss: 0.0894 - val_accuracy: 0.9760 - val_loss: 0.1282
Epoch 79/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9651 - loss: 0.0866 - val_accuracy: 0.9750 - val_loss: 0.1289
Epoch 80/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9558 - loss: 0.1001 - val_accuracy: 0.9770 - val_loss: 0.1289
Epoch 81/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9628 - loss: 0.1034 - val_accuracy: 0.9770 - val_loss: 0.1340
Epoch 82/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.9508 - loss: 0.1076 - val_accuracy: 0.9750 - val_loss: 0.1446
Epoch 83/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.9651 - loss: 0.0936 - val_accuracy: 0.9710 - val_loss: 0.1442
Epoch 84/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9573 - loss: 0.0968 - val_accuracy: 0.9750 - val_loss: 0.1404
Epoch 85/100
120/120 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9540 - loss: 0.1013 - val_accuracy: 0.9790 - val_loss: 0.1246
Epoch 85: early stopping
Restoring model weights from the end of the best epoch: 55.
```

```
# === Plot Loss Curves ===
history_df = pd.DataFrame(history.history)

plt.figure(figsize=(10, 4))
plt.subplot(1, 2, 1)
plt.plot(history_df['loss'], label='Train Loss')
plt.plot(history_df['val_loss'], label='Val Loss')
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.title("Loss Curve")
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7b91011c8bd0>
```




```
# === Plot Accuracy Curves ===
plt.subplot(1, 2, 2)
plt.plot(history_df['accuracy'], label='Train Accuracy')
plt.plot(history_df['val_accuracy'], label='Val Accuracy')
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.title("Accuracy Curve")
plt.legend()
plt.tight_layout()
plt.show()
```




```
# === Final Evaluation on True Test Set ===
y_ann_probs = ANN_model.predict(X_finaltest_scaled)
y_ann_preds = (y_ann_probs > 0.5).astype(int)

print("\n===== ANN Classification Report (Final Test Set) =====")
print(classification_report(y_finaltest, y_ann_preds))
print("ROC AUC Score:", roc_auc_score(y_finaltest, y_ann_probs))
```

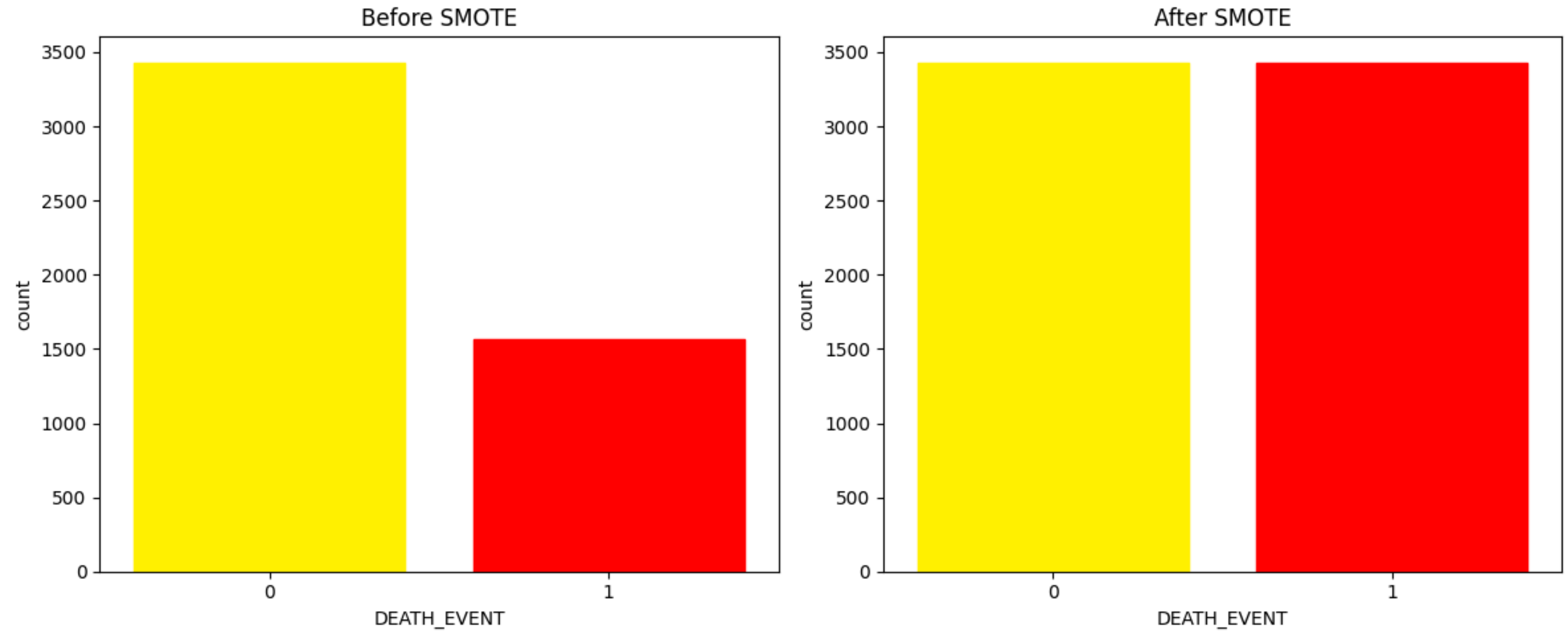
```
32/32 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step

===== ANN Classification Report (Final Test Set) =====
              precision    recall  f1-score   support

           0       0.98      0.97      0.98       686
           1       0.94      0.96      0.95       314

    accuracy                           0.97      1000
   macro avg       0.96      0.96      0.96      1000
weighted avg       0.97      0.97      0.97      1000

ROC AUC Score: 0.9892388256485488
```

So we can see that XGBoost ad Random Forest Classification gives us the highest ROC AUC Score for the given dataset. Now the dataset is not perfectly sampled i.e. The numner of **YES** and **NO** for **Death_Event** are not equal. Thus we'll apply SMOTE to balance the class and then work with Gaussian Naive Bayes and AdaBoost to see if the changes.

```
# Initialize SMOTE
smote = SMOTE(random_state=42)

# Resample features and target
X_resampled, y_resampled = smote.fit_resample(X, y)

# Train-test split after SMOTE
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X_resampled, y_resampled, test_size=0.2, random_state=42
)
```

```
# Colors
cols = ['#FFF000', '#FF0000']

# Create a new DataFrame from the resampled data
df_resampled = pd.DataFrame(X_resampled, columns=X.columns)
df_resampled['DEATH_EVENT'] = y_resampled

# Plot side-by-side
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Before SMOTE
```

```
ax1 = sns.countplot(x="DEATH_EVENT", data=df, ax=axes[0])
axes[0].set_title("Before SMOTE")
for bar, color in zip(ax1.patches, cols):
    bar.set_color(color)

# After SMOTE
ax2 = sns.countplot(x="DEATH_EVENT", data=df_resampled, ax=axes[1])
axes[1].set_title("After SMOTE")
for bar, color in zip(ax2.patches, cols):
    bar.set_color(color)

plt.tight_layout()
plt.show()

```




```
# Standard Scaling after SMOTE and train-test split
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Initialize and train the GaussianNB model
gnb_model = GaussianNB()
gnb_model.fit(X_train, y_train)

#  Make predictions and probability estimates
gnb_preds = gnb_model.predict(X_test)
gnb_probs = gnb_model.predict_proba(X_test)[:, 1]

# Evaluate the model
print("\n===== Gaussian Naive Bayes Classification Report =====")
print(classification_report(y_test, gnb_preds))
print("ROC AUC Score:", roc_auc_score(y_test, gnb_probs))
```

```
===== Gaussian Naive Bayes Classification Report =====
              precision    recall  f1-score   support

           0       0.76      0.91      0.83       693
           1       0.89      0.71      0.79       680

    accuracy                           0.81      1373
   macro avg       0.82      0.81      0.81      1373
weighted avg       0.82      0.81      0.81      1373

ROC AUC Score: 0.8867095322977676
```

```
# Initialize and train the AdaBoost model
ada_model = AdaBoostClassifier(n_estimators=100, random_state=42)
ada_model.fit(X_train, y_train)

# Predictions and probabilities
ada_preds = ada_model.predict(X_test)
ada_probs = ada_model.predict_proba(X_test)[:, 1]

# Evaluation
print("\n===== AdaBoost Classification Report =====")
print(classification_report(y_test, ada_preds))
print("ROC AUC Score:", roc_auc_score(y_test, ada_probs))
```

```
===== AdaBoost Classification Report =====
              precision    recall  f1-score   support

           0       0.91      0.90      0.90       693
           1       0.90      0.90      0.90       680

    accuracy                           0.90      1373
   macro avg       0.90      0.90      0.90      1373
weighted avg       0.90      0.90      0.90      1373

ROC AUC Score: 0.9711038961038962
```

Thus we can see applying SMOTE increased the ROC AUC Score for Gaussian Naive Bayes by 0.002 and for AdaBoost by 0.9.

Finally using LIME helps in understanding why a particular prediction was made. LIME doesn't explain the whole model — it explains why the model made a decision for one particular sample (a local explanation).

In this case I found the misclassified samples and used LIME to find which features made the model give a wrong prediction.

The bars and values show which features pushed the prediction towards **"Survived" (blue)** or toward **"Died" (orange)**

The values in the centre left table for each bar gives the magnitude of contribution of that feature interval to the final prediction.

🟦 Blue bars = Pushed prediction toward “Survived”

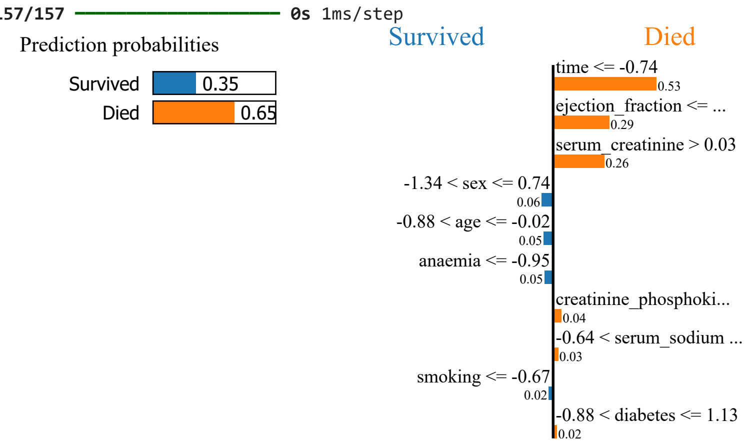
🟧 Orange bars = Pushed prediction toward “Died”

Explainable AI (EAI)

```
import lime
import lime.lime_tabular

feature_names = [
    'age', 'anaemia', 'creatinine_phosphokinase', 'diabetes', 'ejection_fraction',
    'high_blood_pressure', 'platelets', 'serum_creatinine', 'serum_sodium',
    'sex', 'smoking', 'time'
]

# LIME explainer
lime_explainer = lime.lime_tabular.LimeTabularExplainer(
    training_data=X_trainval_scaled,
    feature_names=feature_names,
    class_names=['Survived', 'Died'],
    mode='classification',
    discretize_continuous=True
)

# Wrapper to match ANN's probability shape
def predict_proba_wrapper(x):
    probs = ANN_model.predict(x)
    return np.hstack([1 - probs, probs])

```

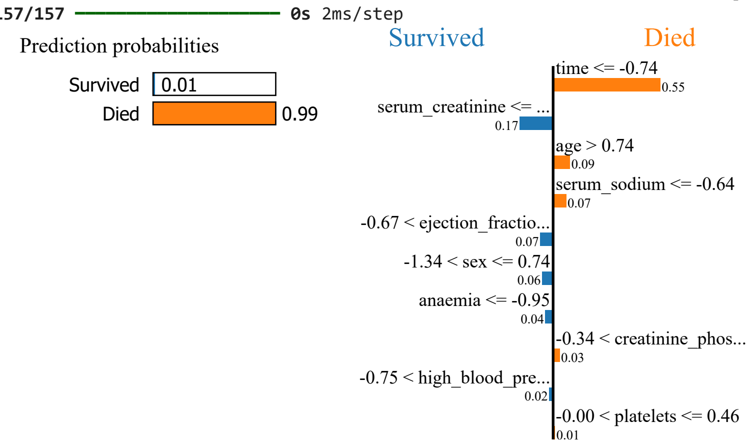
```
# Flatten predictions just to be safe
y_pred_binary = y_ann_preds.flatten()

# Find indices where predicted not equal to actual
misclassified_indices = np.where(y_pred_binary != y_finaltest.to_numpy())[0]

print(f"Total misclassified samples: {len(misclassified_indices)}")
print("Indices of misclassified samples:", misclassified_indices)

for i in misclassified_indices:
    print(f"\n--- Misclassified Sample #{i} ---")
    print(f"True Label     : {y_finaltest.iloc[i]}")
    print(f"Predicted Label: {y_pred_binary[i]}")
    print(f"Features (scaled):\n{X_finaltest_scaled[i]}")

# Explain with LIME
    lime_exp = lime_explainer.explain_instance(
        X_finaltest_scaled[i],
        predict_proba_wrapper,
        num_features=10
    )
    lime_exp.show_in_notebook()
```

```
Total misclassified samples: 34
Indices of misclassified samples: [  1  39 123 137 183 200 299 339 392 411 445 452 459 484 489 501 507 509
 540 556 599 639 702 713 742 765 825 870 880 883 897 942 956 977]

--- Misclassified Sample #1 ---
True Label      : 0
Predicted Label: 1
Features (scaled):
[-0.02395998 -0.94595802  2.17087895  1.13044431 -0.66643471 -0.74593581
  0.42348945  0.91986839  0.03258139  0.74471842 -0.67263127 -1.30608688]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```



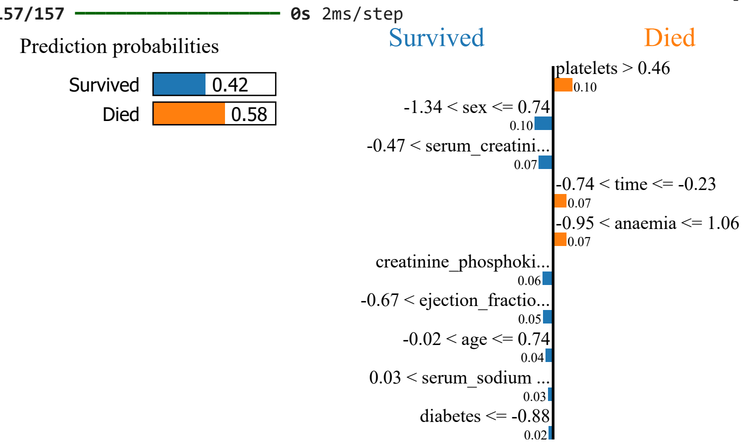
Feature	Value
time	-1.31
ejection_fraction	-0.67
serum_creatinine	0.92
sex	0.74
age	-0.02
anaemia	-0.95
creatinine_phosphokinase	2.17
serum_sodium	0.03
smoking	-0.67
diabetes	1.13

```
--- Misclassified Sample #39 ---
True Label      : 0
Predicted Label: 1
Features (scaled):
[ 1.85244778 -0.94595802 -0.02069111  1.13044431  0.02731645  1.34059793
  0.19809932 -0.76401054 -1.31098107  0.74471842  1.48669865 -0.73737163]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



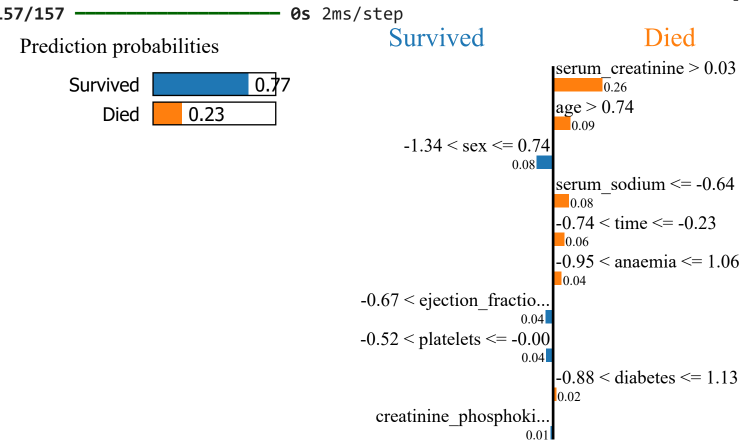
Feature	Value
time	-0.74
serum_creatinine	-0.76
age	1.85
serum_sodium	-1.31
ejection_fraction	0.03
sex	0.74
anaemia	-0.95
creatinine_phosphokinase	-0.02
high_blood_pressure	1.34
platelets	0.20

```
--- Misclassified Sample #123 ---
True Label      : 0
Predicted Label: 1
Features (scaled):
[ 0.40249633  1.05712937 -0.54324    -0.88460793 -0.23284023 -0.74593581
  1.03818979 -0.26875203  0.48043554  0.74471842  1.48669865 -0.25913381]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



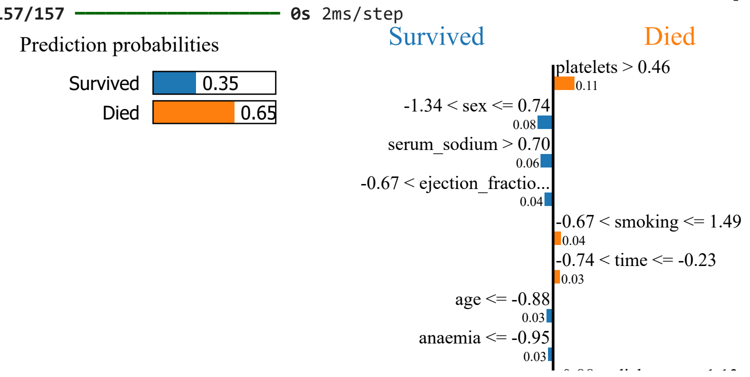
Feature	Value
platelets	1.04
sex	0.74
serum_creatinine	-0.27
time	-0.26
anaemia	1.06
creatinine_phosphokinase	-0.54
ejection_fraction	-0.23
age	0.40
serum_sodium	0.48
diabetes	-0.88

```
--- Misclassified Sample #137 ---
True Label      : 1
Predicted Label: 0
Features (scaled):
[ 1.59657399  1.05712937 -0.54742039  1.13044431  0.02731645  1.34059793
 -0.00313277  0.45432539 -0.63919984  0.74471842 -0.67263127 -0.68567024]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

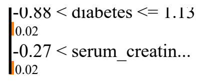


Feature	Value
serum_creatinine	0.45
age	1.60
sex	0.74
serum_sodium	-0.64
time	-0.69
anaemia	1.06
ejection_fraction	0.03
platelets	-0.00
diabetes	1.13
creatinine_phosphokinase	-0.55

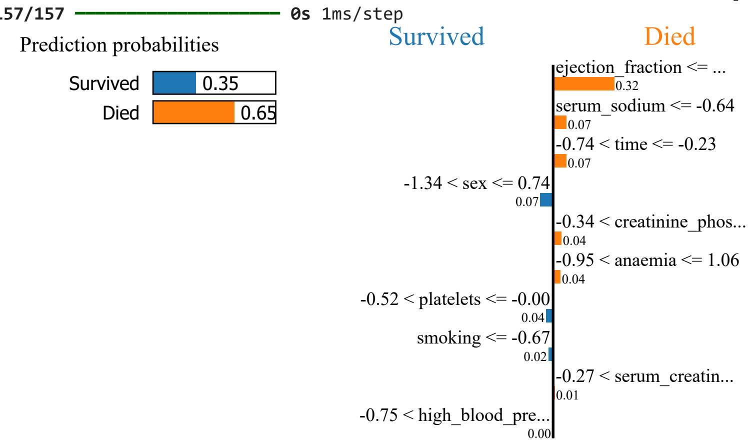
```
--- Misclassified Sample #183 ---
True Label      : 0
Predicted Label: 1
Features (scaled):
[-1.3033289  -0.94595802  1.95140842  1.13044431 -0.23284023 -0.74593581
  6.00701751 -0.07064863  1.15221677  0.74471842  1.48669865 -0.55641678]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



Feature	Value
platelets	6.01
sex	0.74
serum_sodium	1.15
ejection_fraction	-0.23
smoking	1.49
time	-0.56
age	-1.30
anaemia	-0.95
diabetes	1.13
serum_creatinine	-0.07

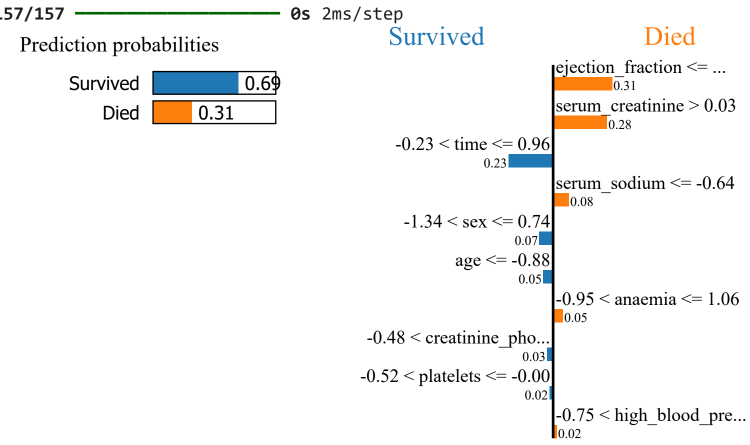



```
--- Misclassified Sample #200 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 0.23191381  1.05712937 -0.06772051  1.13044431 -1.10002918  1.34059793
 -0.09900584 -0.07064863 -0.63919984  0.74471842 -0.67263127 -0.62104351]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```



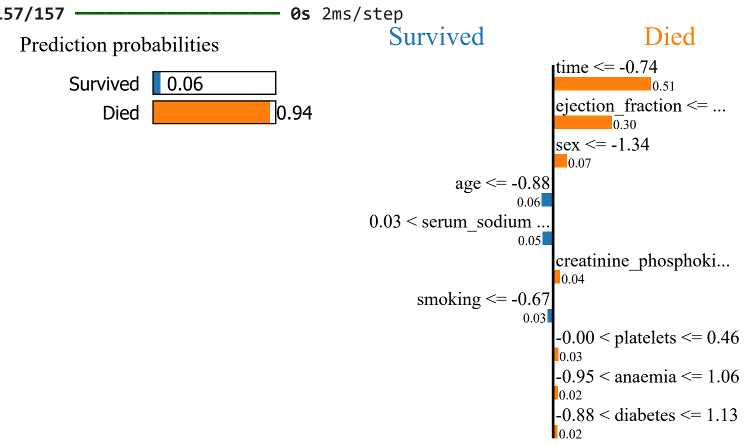
Feature	Value
ejection_fraction	-1.10
serum_sodium	-0.64
time	-0.62
sex	0.74
creatinine_phosphokinase	-0.07
anaemia	1.06
platelets	-0.10
smoking	-0.67
serum_creatinine	-0.07
high_blood_pressure	1.34

```
--- Misclassified Sample #299 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-1.04745512  1.05712937 -0.46799296 -0.88460793 -0.66643471  1.34059793
 -0.29366095  0.22650648 -1.53490814  0.74471842 -0.67263127  0.8007446 ]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



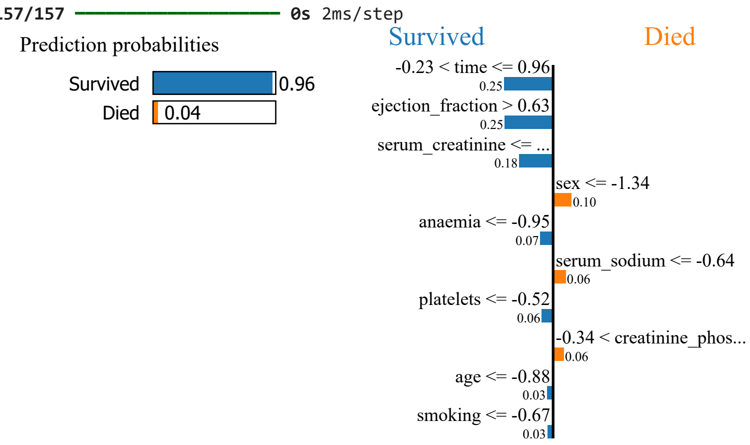
Feature	Value
ejection_fraction	-0.67
serum_creatinine	0.23
time	0.80
serum_sodium	-1.53
sex	0.74
age	-1.05
anaemia	1.06
creatinine_phosphokinase	-0.47
platelets	-0.29
high_blood_pressure	1.34

```
--- Misclassified Sample #339 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[-0.8768726   1.05712937  0.50394798  1.13044431 -0.66643471 -0.74593581
  0.39275443 -0.16970033  0.25650847 -1.34278941 -0.67263127 -1.31901222]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



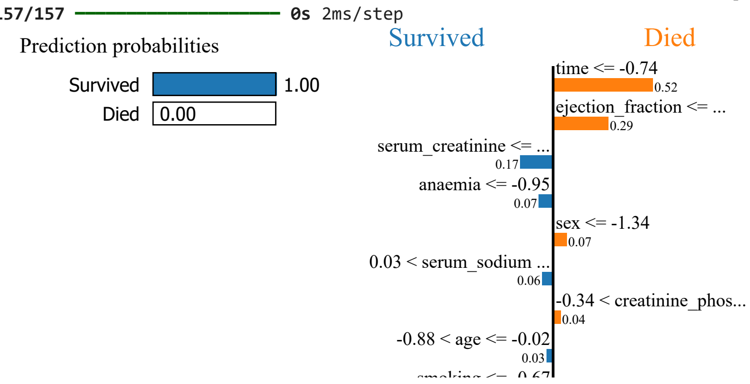
Feature	Value
time	-1.32
ejection_fraction	-0.67
sex	-1.34
age	-0.88
serum_sodium	0.26
creatinine_phosphokinase	0.50
smoking	-0.67
platelets	0.39
anaemia	1.06
diabetes	1.13

```
--- Misclassified Sample #392 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-0.8768726  -0.94595802  0.00334614  1.13044431  1.06794319 -0.74593581
 -1.13375141 -0.76401054 -0.63919984 -1.34278941 -0.67263127  0.52931233]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

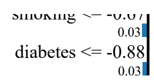


Feature	Value
time	0.53
ejection_fraction	1.07
serum_creatinine	-0.76
sex	-1.34
anaemia	-0.95
serum_sodium	-0.64
platelets	-1.13
creatinine_phosphokinase	0.00
age	-0.88
smoking	-0.67

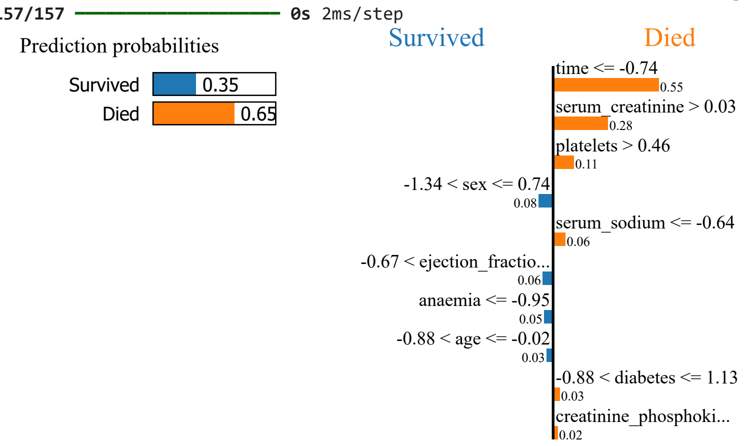
```
--- Misclassified Sample #411 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-0.02395998 -0.94595802  0.00334614 -0.88460793 -0.66643471  1.34059793
  0.30054938 -0.76401054  0.48043554 -1.34278941 -0.67263127 -1.17683341]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



Feature	Value
time	-1.18
ejection_fraction	-0.67
serum_creatinine	-0.76
anaemia	-0.95
sex	-1.34
serum_sodium	0.48
creatinine_phosphokinase	0.00
age	-0.02
smoking	-0.67
diabetes	-0.88

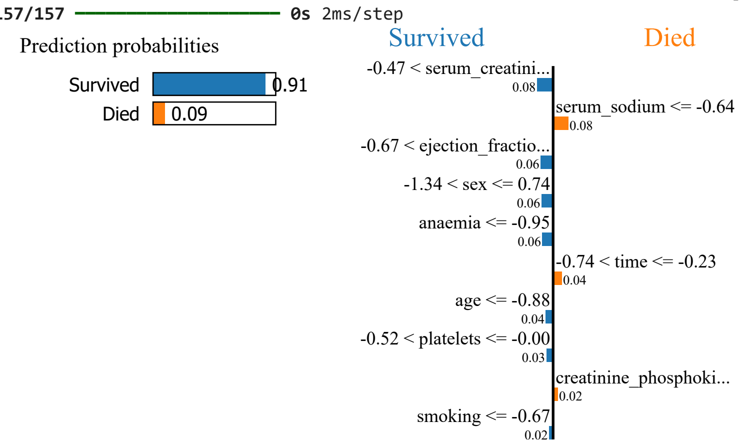



```
--- Misclassified Sample #445 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[-0.02395998 -0.94595802  2.17087895  1.13044431 -0.23284023 -0.74593581
  0.8845147   0.72176499 -0.63919984  0.74471842 -0.67263127 -1.30608688]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



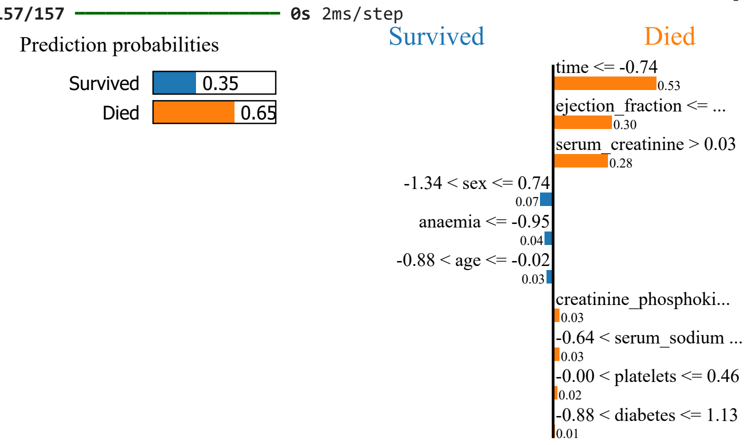
Feature	Value
time	-1.31
serum_creatinine	0.72
platelets	0.88
sex	0.74
serum_sodium	-0.64
ejection_fraction	-0.23
anaemia	-0.95
age	-0.02
diabetes	1.13
creatinine_phosphokinase	2.17

```
--- Misclassified Sample #452 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-0.8768726  -0.94595802  0.26775588 -0.88460793  0.02731645 -0.74593581
 -0.00313277 -0.26875203 -0.63919984  0.74471842 -0.67263127 -0.2849845 ]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



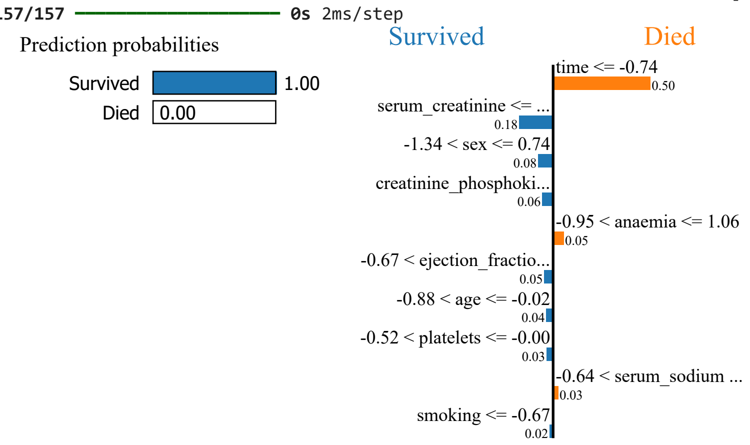
Feature	Value
serum_creatinine	-0.27
serum_sodium	-0.64
ejection_fraction	0.03
sex	0.74
anaemia	-0.95
time	-0.28
age	-0.88
platelets	-0.00
creatinine_phosphokinase	0.27
smoking	-0.67

```
--- Misclassified Sample #459 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[-0.02395998 -0.94595802  2.17087895  1.13044431 -0.66643471 -0.74593581
  0.42348945  0.91986839  0.03258139  0.74471842 -0.67263127 -1.30608688]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



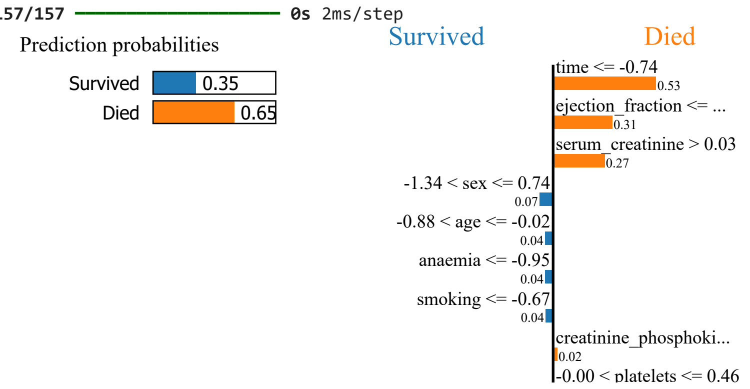
Feature	Value
time	-1.31
ejection_fraction	-0.67
serum_creatinine	0.92
sex	0.74
anaemia	-0.95
age	-0.02
creatinine_phosphokinase	2.17
serum_sodium	0.03
platelets	0.42
diabetes	1.13

```
--- Misclassified Sample #484 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-0.02395998  1.05712937 -0.55578117 -0.88460793  0.02731645 -0.74593581
 -0.40635601 -0.66495884  0.03258139  0.74471842 -0.67263127 -0.75029698]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

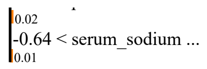


Feature	Value
time	-0.75
serum_creatinine	-0.66
sex	0.74
creatinine_phosphokinase	-0.56
anaemia	1.06
ejection_fraction	0.03
age	-0.02
platelets	-0.41
serum_sodium	0.03
smoking	-0.67

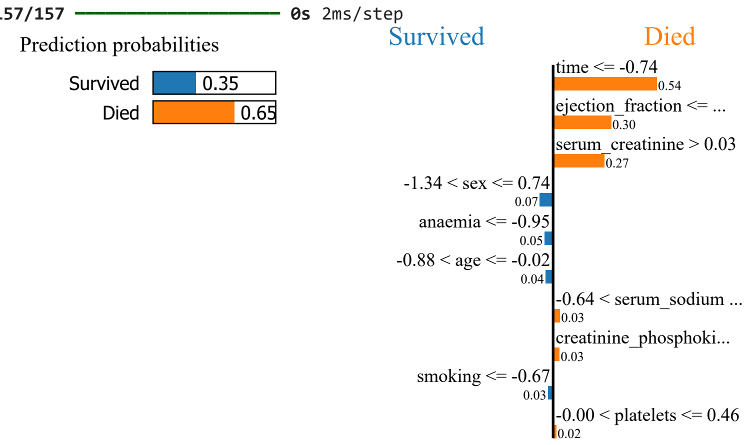
```
--- Misclassified Sample #489 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[-0.02395998 -0.94595802  2.17087895  1.13044431 -0.66643471 -0.74593581
  0.42348945  0.91986839  0.03258139  0.74471842 -0.67263127 -1.30608688]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



Feature	Value
time	-1.31
ejection_fraction	-0.67
serum_creatinine	0.92
sex	0.74
age	-0.02
anaemia	-0.95
smoking	-0.67
creatinine_phosphokinase	2.17
platelets	0.42
serum_sodium	0.03

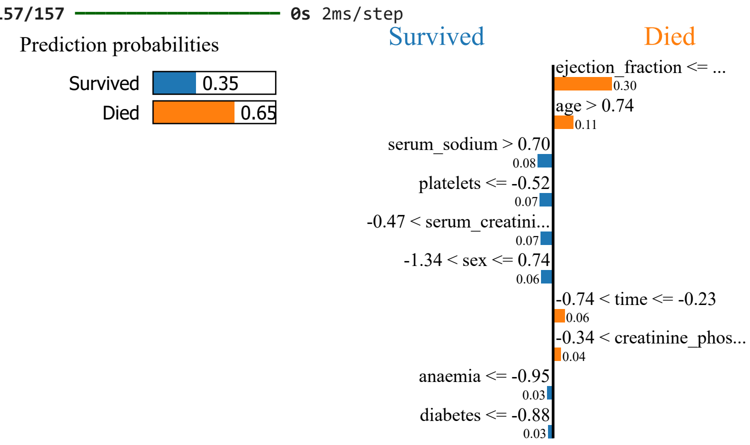



```
--- Misclassified Sample #501 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[-0.02395998 -0.94595802  2.17087895  1.13044431 -0.66643471 -0.74593581
  0.42348945  0.91986839  0.03258139  0.74471842 -0.67263127 -1.30608688]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



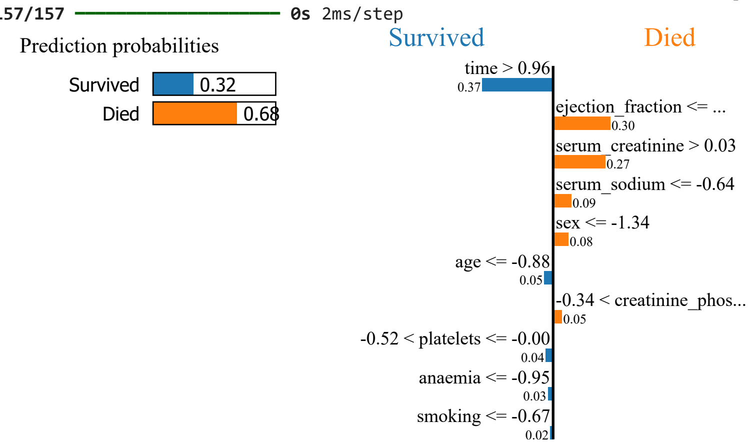
Feature	Value
time	-1.31
ejection_fraction	-0.67
serum_creatinine	0.92
sex	0.74
anaemia	-0.95
age	-0.02
serum_sodium	0.03
creatinine_phosphokinase	2.17
smoking	-0.67
platelets	0.42

```
--- Misclassified Sample #507 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 1.68186525 -0.94595802  0.00334614 -0.88460793 -1.10002918 -0.74593581
 -1.17473143 -0.26875203  1.60007092  0.74471842  1.48669865 -0.56934213]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



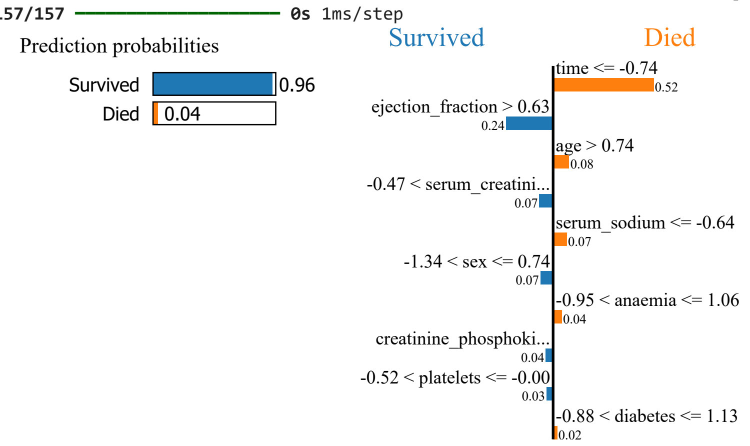
Feature	Value
ejection_fraction	-1.10
age	1.68
serum_sodium	1.60
platelets	-1.17
serum_creatinine	-0.27
sex	0.74
time	-0.57
creatinine_phosphokinase	0.00
anaemia	-0.95
diabetes	-0.88

```
--- Misclassified Sample #509 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[-1.38862017 -0.94595802  0.00334614  1.13044431 -0.66643471  1.34059793
 -0.20145589  0.22650648 -1.53490814 -1.34278941 -0.67263127  1.45993728]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



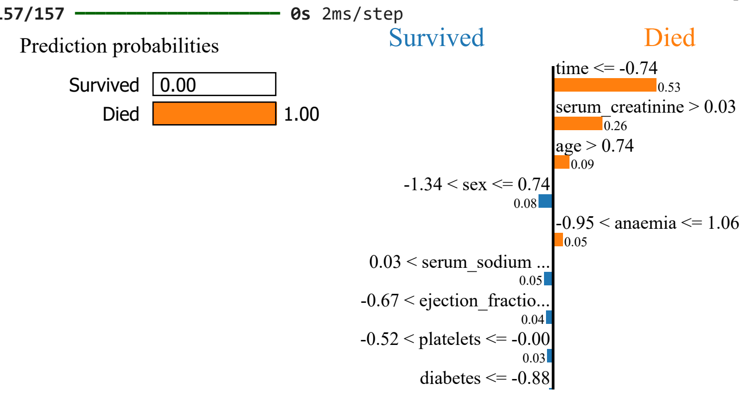
Feature	Value
time	1.46
ejection_fraction	-0.67
serum_creatinine	0.23
serum_sodium	-1.53
sex	-1.34
age	-1.39
creatinine_phosphokinase	0.00
platelets	-0.20
anaemia	-0.95
smoking	-0.67

```
--- Misclassified Sample #540 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[ 0.82895264  1.05712937 -0.53278902  1.13044431  1.06794319  1.34059793
 -0.385866   -0.36780373 -0.63919984  0.74471842 -0.67263127 -1.12513203]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```

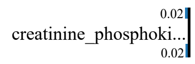


Feature	Value
time	-1.13
ejection_fraction	1.07
age	0.83
serum_creatinine	-0.37
serum_sodium	-0.64
sex	0.74
anaemia	1.06
creatinine_phosphokinase	-0.53
platelets	-0.39
diabetes	1.13

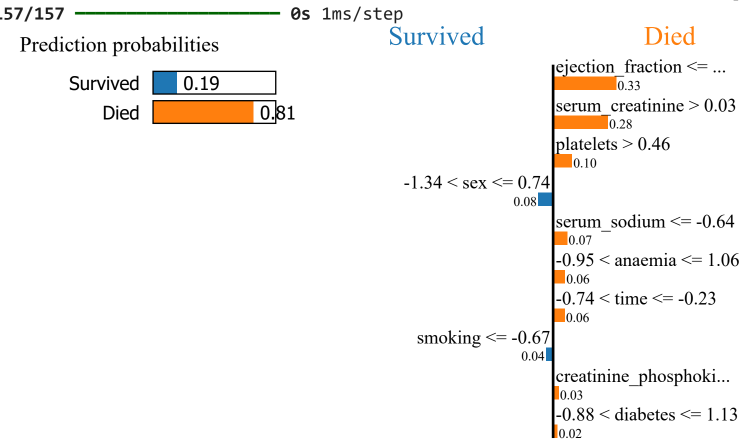
```
--- Misclassified Sample #556 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 0.82895264  1.05712937 -0.53487922 -0.88460793 -0.23284023 -0.74593581
 -0.41660101  1.3160752   0.25650847  0.74471842  1.48669865 -0.99587856]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



Feature	Value
time	-1.00
serum_creatinine	1.32
age	0.83
sex	0.74
anaemia	1.06
serum_sodium	0.26
ejection_fraction	-0.23
platelets	-0.42
diabetes	-0.88
creatinine_phosphokinase	-0.53

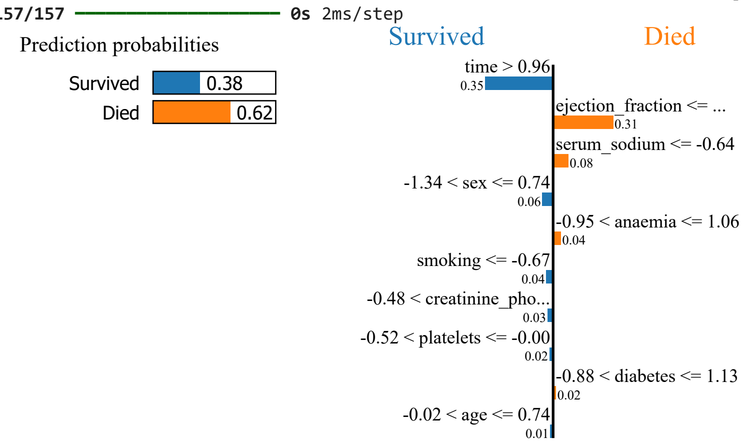



```
--- Misclassified Sample #599 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 0.65837011  1.05712937  0.0702324   1.13044431 -1.10002918 -0.74593581
  1.43774501  0.91986839 -0.63919984  0.74471842 -0.67263127 -0.29790985]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```



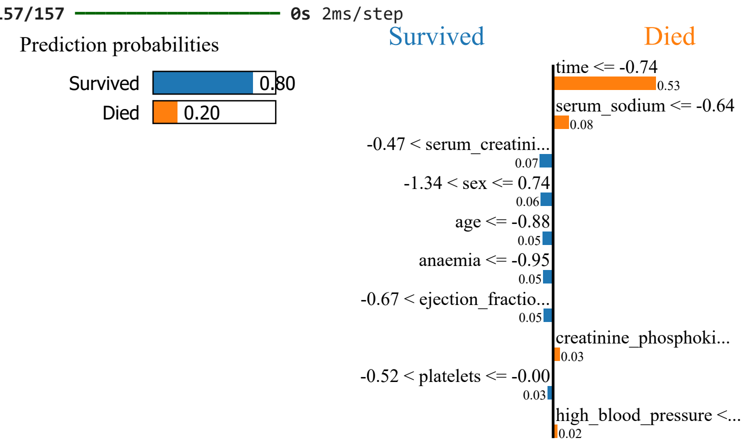
Feature	Value
ejection_fraction	-1.10
serum_creatinine	0.92
platelets	1.44
sex	0.74
serum_sodium	-0.64
anaemia	1.06
time	-0.30
smoking	-0.67
creatinine_phosphokinase	0.07
diabetes	1.13

```
--- Misclassified Sample #639 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 0.65837011  1.05712937 -0.44082042  1.13044431 -1.10002918 -0.74593581
 -0.09900584 -0.07064863 -0.63919984  0.74471842 -0.67263127  1.36945985]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



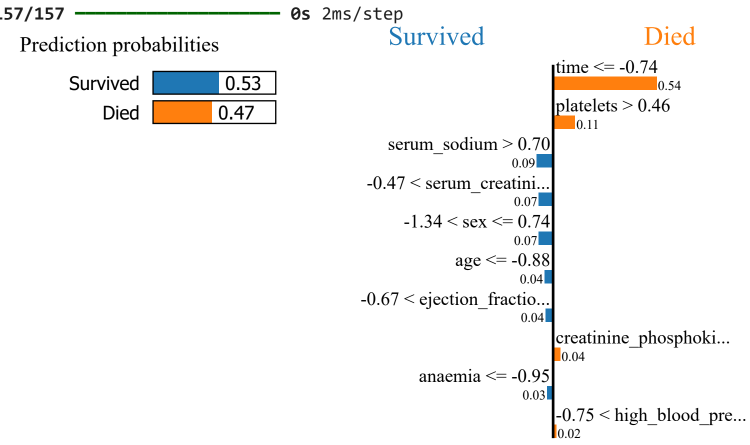
Feature	Value
time	1.37
ejection_fraction	-1.10
serum_sodium	-0.64
sex	0.74
anaemia	1.06
smoking	-0.67
creatinine_phosphokinase	-0.44
platelets	-0.10
diabetes	1.13
age	0.66

```
--- Misclassified Sample #702 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-1.3033289  -0.94595802  1.91692019 -0.88460793 -0.23284023 -0.74593581
 -0.2117009  -0.36780373 -1.08705399  0.74471842  1.48669865 -0.76322232]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



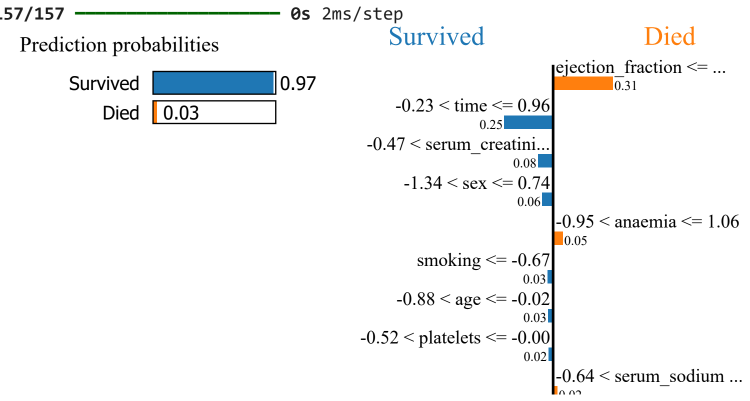
Feature	Value
time	-0.76
serum_sodium	-1.09
serum_creatinine	-0.37
sex	0.74
age	-1.30
anaemia	-0.95
ejection_fraction	-0.23
creatinine_phosphokinase	1.92
platelets	-0.21
high_blood_pressure	-0.75

```
--- Misclassified Sample #713 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-1.3033289  -0.94595802  7.4444424  -0.88460793 -0.23284023  1.34059793
  1.2430899  -0.36780373  1.823998    0.74471842 -0.67263127 -0.90540114]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

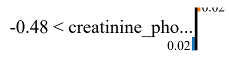


Feature	Value
time	-0.91
platelets	1.24
serum_sodium	1.82
serum_creatinine	-0.37
sex	0.74
age	-1.30
ejection_fraction	-0.23
creatinine_phosphokinase	7.44
anaemia	-0.95
high_blood_pressure	1.34

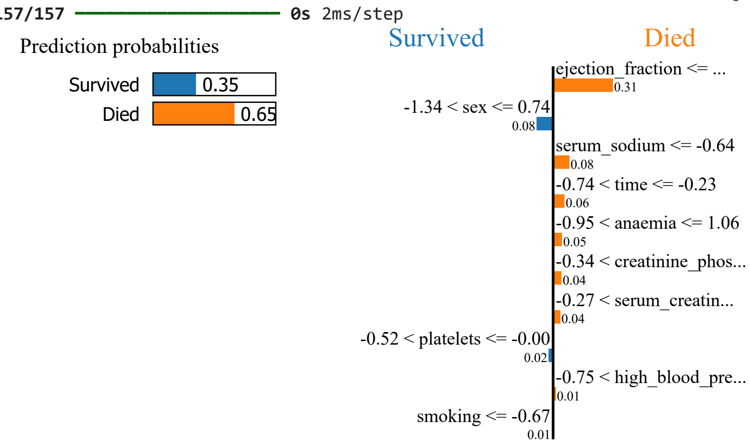
```
--- Misclassified Sample #742 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[-0.10925124  1.05712937 -0.42096356  1.13044431 -1.10002918 -0.74593581
 -0.43709102 -0.36780373 -0.19134569  0.74471842 -0.67263127  0.2449547 ]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



Feature	Value
ejection_fraction	-1.10
time	0.24
serum_creatinine	-0.37
sex	0.74
anaemia	1.06
smoking	-0.67
age	-0.11
platelets	-0.44
serum_sodium	-0.19
creatinine_phosphokinase	-0.42

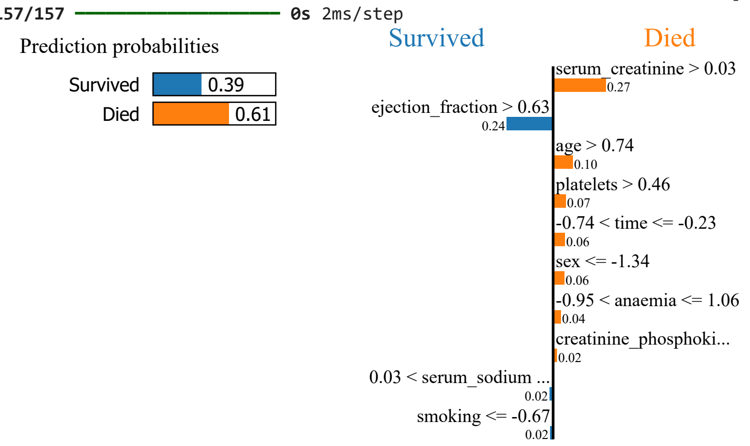



```
--- Misclassified Sample #765 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 0.23191381  1.05712937 -0.06772051  1.13044431 -1.10002918  1.34059793
 -0.09900584 -0.07064863 -0.63919984  0.74471842 -0.67263127 -0.62104351]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



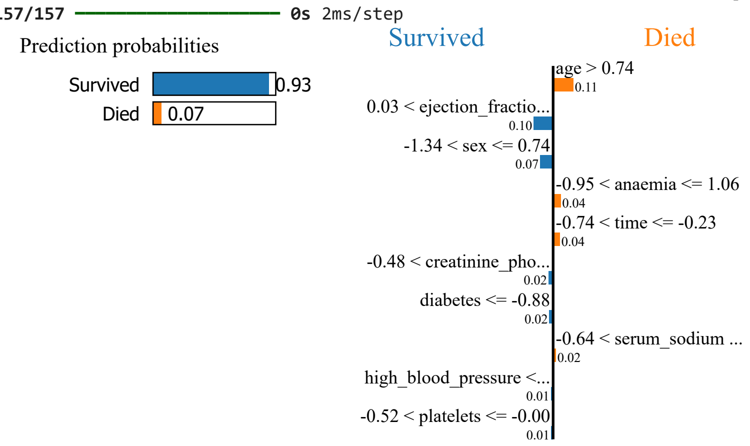
Feature	Value
ejection_fraction	-1.10
sex	0.74
serum_sodium	-0.64
time	-0.62
anaemia	1.06
creatinine_phosphokinase	-0.07
serum_creatinine	-0.07
platelets	-0.10
high_blood_pressure	1.34
smoking	-0.67

```
--- Misclassified Sample #825 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 2.10832156  1.05712937  0.51439896 -0.88460793  1.93513214 -0.74593581
  2.49298058  1.81133371  0.25650847 -1.34278941 -0.67263127 -0.4788647 ]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



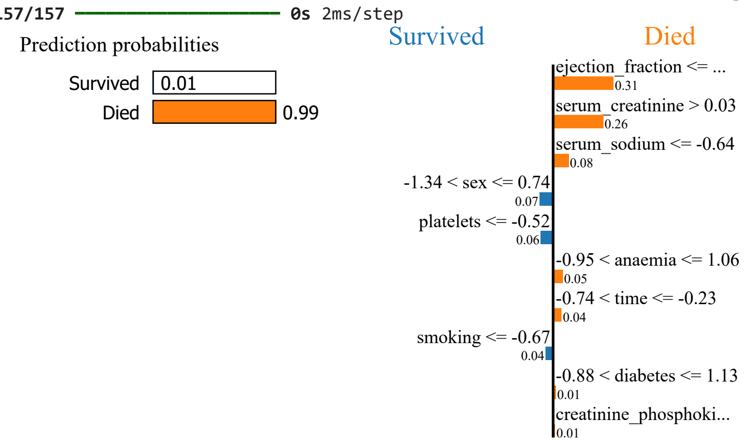
Feature	Value
serum_creatinine	1.81
ejection_fraction	1.94
age	2.11
platelets	2.49
time	-0.48
sex	-1.34
anaemia	1.06
creatinine_phosphokinase	0.51
serum_sodium	0.26
smoking	-0.67

```
--- Misclassified Sample #870 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[ 0.82895264  1.05712937 -0.45545179 -0.88460793  0.63434872 -0.74593581
 -0.41660101 -0.07064863  0.03258139  0.74471842 -0.67263127 -0.53056609]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



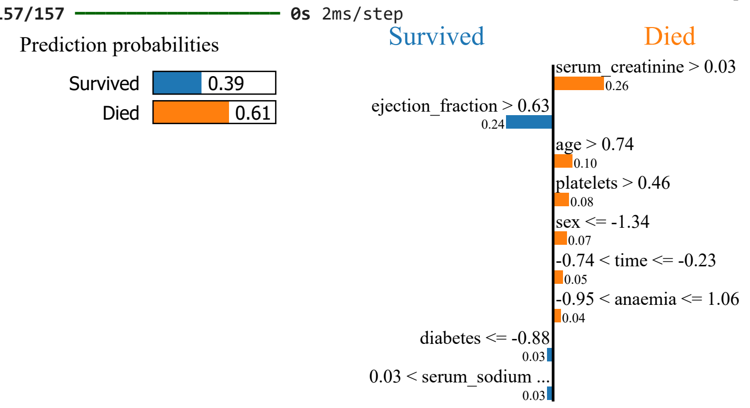
Feature	Value
age	0.83
ejection_fraction	0.63
sex	0.74
anaemia	1.06
time	-0.53
creatinine_phosphokinase	-0.46
diabetes	-0.88
serum_sodium	0.03
high_blood_pressure	-0.75
platelets	-0.42

```
--- Misclassified Sample #880 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 0.65837011  1.05712937  0.0702324   1.13044431 -1.10002918 -0.74593581
 -1.40012155  0.72176499 -1.53490814  0.74471842 -0.67263127 -0.29790985]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

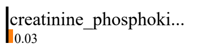


Feature	Value
ejection_fraction	-1.10
serum_creatinine	0.72
serum_sodium	-1.53
sex	0.74
platelets	-1.40
anaemia	1.06
time	-0.30
smoking	-0.67
diabetes	1.13
creatinine_phosphokinase	0.07

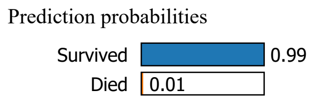
```
--- Misclassified Sample #883 ---
True Label     : 0
Predicted Label: 1
Features (scaled):
[ 2.10832156  1.05712937  0.51439896 -0.88460793  1.93513214 -0.74593581
  2.49298058  1.81133371  0.25650847 -1.34278941 -0.67263127 -0.4788647 ]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

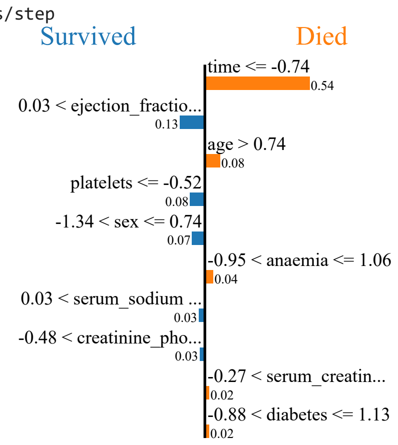


Feature	Value
serum_creatinine	1.81
ejection_fraction	1.94
age	2.11
platelets	2.49
sex	-1.34
time	-0.48
anaemia	1.06
diabetes	-0.88
serum_sodium	0.26
creatinine_phosphokinase	0.51




```
--- Misclassified Sample #897 ---
True Label     : 1
Predicted Label: 0
Features (scaled):
[ 0.82895264  1.05712937 -0.47426355  1.13044431  0.63434872  1.34059793
 -0.80591123 -0.16970033  0.48043554  0.74471842  1.48669865 -0.8407744 ]
157/157 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```





Feature	Value
time	-0.84
ejection_fraction	0.63
age	0.83
platelets	-0.81
sex	0.74
anaemia	1.06
serum_sodium	0.48
creatinine_phosphokinase	-0.47
serum_creatinine	-0.17
diabetes	1.13

```
--- Misclassified Sample #942 ---
True Label     : 1
```