Heart Disease Prediction Project Report

1. Problem Statement

Heart disease is one of the leading causes of death worldwide. Early diagnosis and intervention can save lives by allowing timely treatment and lifestyle changes. The goal of this project is to leverage machine learning to predict the presence of heart disease in a patient based on a set of clinical and demographic features.

The primary challenge is to build a robust, accurate, and generalizable model that can assist medical professionals in making quick and data-driven decisions..

2. Dataset Description

The dataset used for this project contains patient-level information, including clinical measurements
and diagnostic outcomes. Below are the key features:

Feature Name	Description	Type
age	Age of the patient (in years)	Numeric
sex	Gender (0 = Female, 1 = Male)	Categorical
Resting blood_pressure	Resting blood pressure in mmHg	Numeric
serum_cholesterol	Cholesterol level in mg/dL	Numeric
fasting_blood_sugar	Blood sugar > 120 mg/dL (1 = True, 0 = False)	Categorical
max_heart_rate	Maximum heart rate achieved during stress	Numeric
Exercise induced_angina	Angina induced by exercise (1 = Yes, 0 = No)	Categorical
target	Presence of heart disease (1 = Yes, 0 = No)	Categorical

3. Objective

The objective of this project is to build a machine learning pipeline that:

1. **Preprocesses the data**: This includes handling missing values, scaling numeric data, and encoding categorical variables.
2. **Trains a Random Forest Classifier**: Random Forest is selected due to its robustness and ability to handle both categorical and numeric data.
3. **Evaluates the model's performance**: Metrics such as accuracy, confusion matrix, precision, recall, and F1-score are used to assess the model's effectiveness.

4. Libraries Used

To accomplish the objectives, the following Python libraries were used:

- **Pandas**: For data loading, manipulation, and cleaning.
- **NumPy**: For numerical computations.
- **Matplotlib & Seaborn**: For creating visualizations like correlation heatmaps and data distributions.
- **Scikit-learn**: For data preprocessing, building pipelines, model training, and evaluation.

5. Methodology

5.1 Data Preprocessing

- Loading the Data: The dataset is read into a Pandas DataFrame.
- Feature-Target Separation: The features (X) and target variable (y) are separated.
- Train-Test Split: The data is split into 80% training and 20% testing sets to ensure the model generalizes well to unseen data.

5.2 Pipeline Creation

- Numeric Features: A pipeline using StandardScaler is created to scale numeric features like age, cholesterol, etc.
- Categorical Features: A pipeline using OneHotEncoder handles categorical variables such as sex and exercise_induced_angina.
- Preprocessing Pipeline: Both numeric and categorical pipelines are combined using ColumnTransformer for end-to-end preprocessing.

5.3 Model Training

- The preprocessed data is fed into a RandomForestClassifier.
- The pipeline is trained on the training dataset.
- The trained pipeline, including preprocessing steps and the model, is saved using joblib for future use.

5.4 Model Evaluation

- Predictions are generated for the test set.
- The following evaluation metrics are computed:
    - Accuracy: Percentage of correct predictions.
    - Confusion Matrix: A matrix showing True Positives, True Negatives, False Positives, and False Negatives.
    - Precision, Recall, F1-Score: Detailed evaluation metrics to understand performance on imbalanced datasets

6. Results

Confusion Matrix:

	Predicted: No Disease	**Predicted: Disease**
Actual: No Disease	TN	FP
Actual: Disease	FN	TP

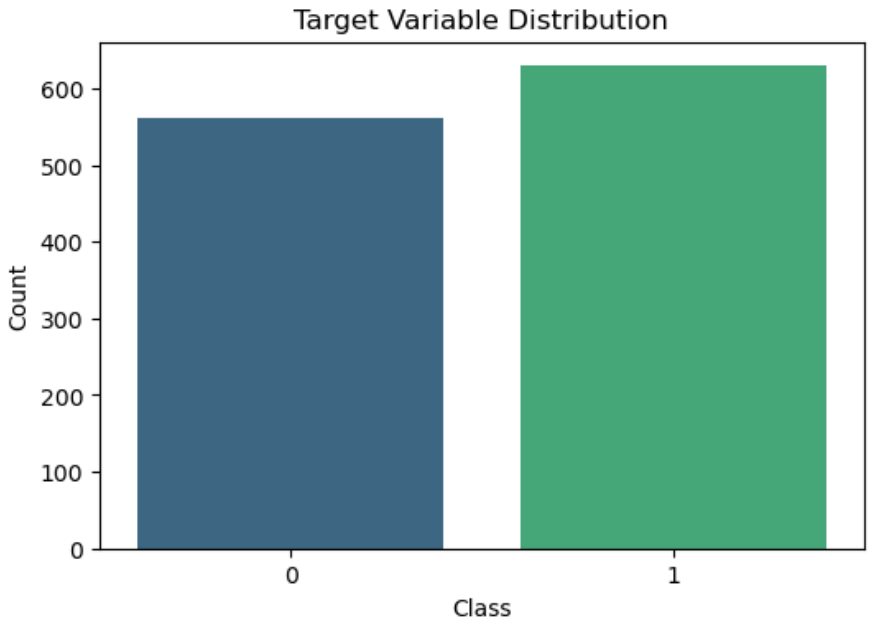
Classification Report: The classification report includes precision, recall, and F1-score for each class.

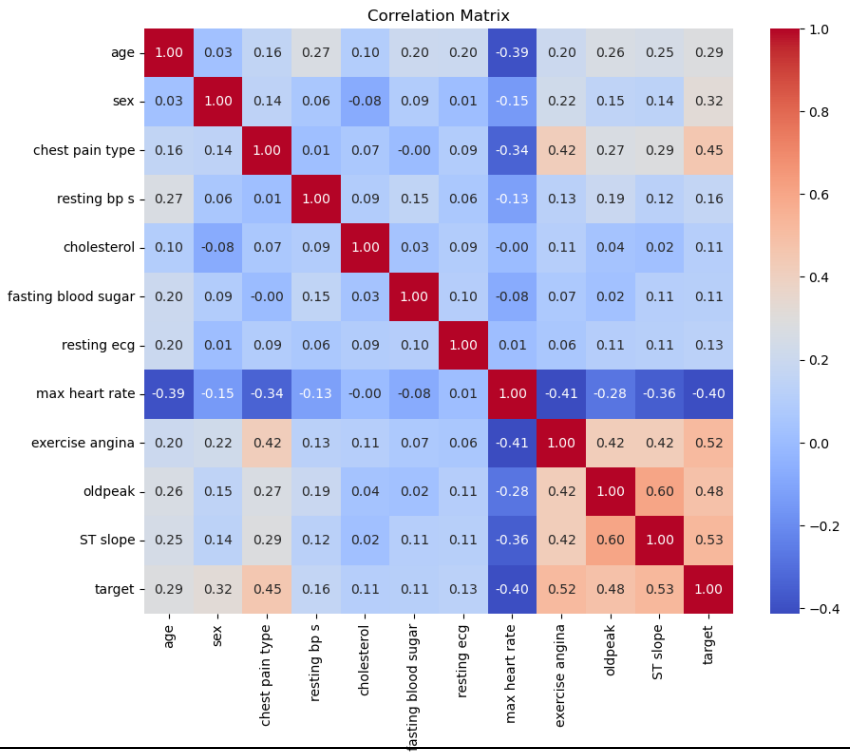
Accuracy: The model achieved an accuracy of **94%** on the test dataset.

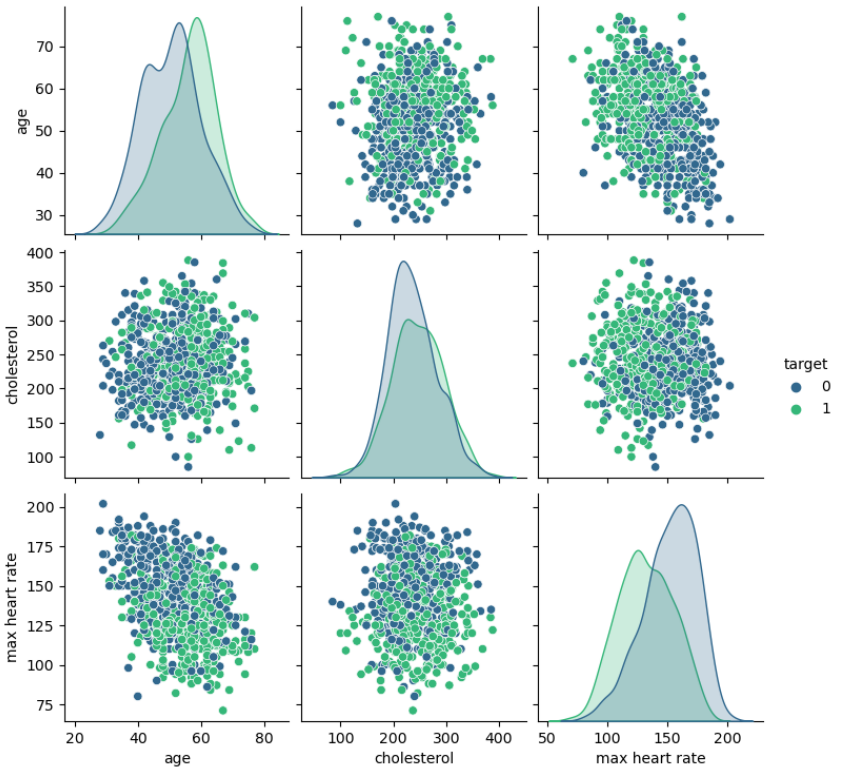
7. Key Insights

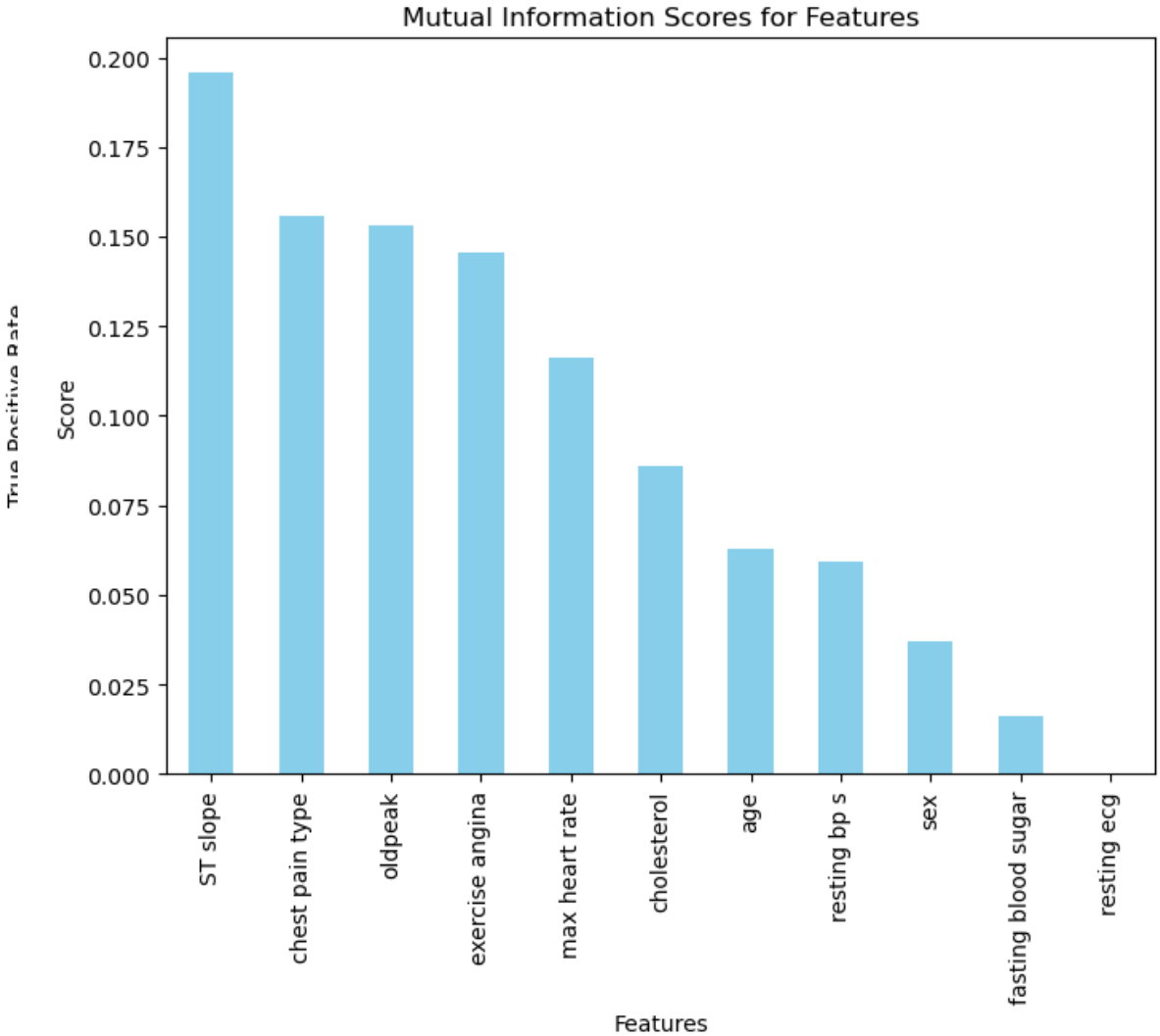
- **Feature Importance**: The Random Forest algorithm highlights the most important features contributing to the predictions.
- **Confusion Matrix Analysis**: Observations of False Positives and False Negatives can help refine the model further.
- **Pipeline Efficiency**: Combining preprocessing and modeling in a single pipeline improves reproducibility and reduces errors.

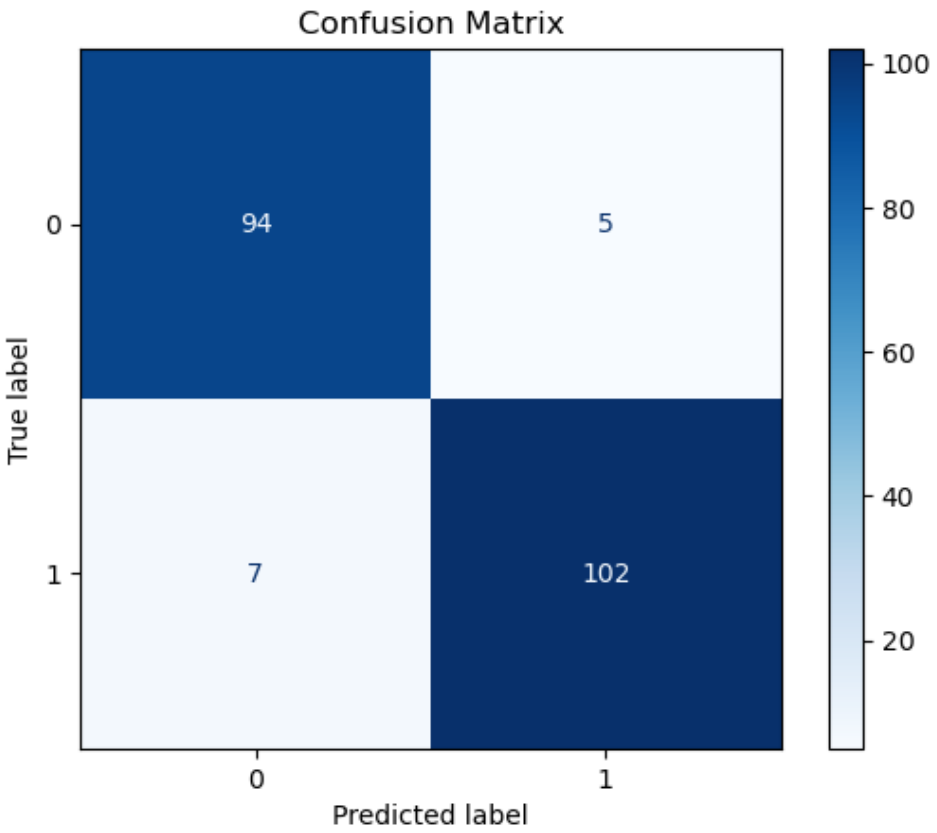
8. Screenshots






age
cholesterol
max heart rate
target
0
1


Mutual Information Scores for Features
Score
Features
ST slope
chest pain type
oldpeak
exercise angina
max heart rate
cholesterol
age
resting bp s
sex
fasting blood sugar
resting ecg



9. Future Enhancements

- **Hyperparameter Tuning**: Explore GridSearchCV or RandomizedSearchCV to fine-tune model parameters.
- **Algorithm Exploration**: Test more advanced algorithms like Gradient Boosting, XGBoost, or LightGBM.
- **Deployment**: Host the model as a web application using cloud platforms like AWS or Azure for real-time predictions.
- **Real-Time Data**: Integrate real-time patient monitoring data for live predictions.

10. References

- Scikit-learn documentation
- Official Pandas documentation