Assessment Report

on

“Heart Disease Prediction”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-25

in

CSE(AI)

By

Rachit Gupta(202401100300189)

Rashika(202401100300196)

Pranjal Sharma(202401100300174)

Satyam Tyagi(202401100300220)

Mayan Prajapati(202401100300151)

Section: C

➢ A. Introduction

Heart disease is one of the leading causes of death globally. Detecting it early can save lives by enabling timely treatment. The goal of this project is to use machine learning techniques to predict whether a person has heart disease based on features such as age, cholesterol level, blood pressure, and other medical parameters.

This project uses a dataset from Kaggle which includes several medical attributes for patients. By training classification models, we aim to predict the presence of heart disease.

➢ B. Problem Statement

- Heart disease is a major cause of mortality worldwide. It is often difficult to diagnose due to the complex interaction of multiple medical factors. Early detection of heart disease is crucial for timely treatment and improving survival rates. Traditional diagnostic methods can be time-consuming and dependent on the expertise of healthcare professionals. Therefore, there is a need for an automated system that can accurately predict the presence of heart disease based on clinical parameters.
- This project focuses on developing a machine learning-based classification model that can analyze patient data and predict the likelihood of heart disease. The model will use features such as age, cholesterol level, blood pressure, and other medical indicators to make predictions.

➢ C. Objective

- To develop an efficient machine learning model that can classify whether a patient has heart disease based on clinical parameters.
- To preprocess and analyze the dataset, including handling missing values and scaling features.
- To perform exploratory data analysis (EDA) to understand the relationships between different medical features.
- To evaluate different classification algorithms such as Logistic Regression, Decision Tree, Random Forest, and KNN.
- To assess the performance of the models using metrics like accuracy, precision, recall, and F1-score.
- To visualize the data and model results using plots like heatmaps, confusion matrices, and ROC curves.
- To assist healthcare professionals in early and accurate diagnosis of heart disease using data-driven methods.

➢ D. Methodology

Step 1: Data Collection

Dataset downloaded from: Kaggle - Heart Disease Data

Step 2: Data Preprocessing

- Handle missing values (if any)
- Encode categorical features
- Normalize numerical features

Step 3: Exploratory Data Analysis (EDA)

- Visualize correlations using a heatmap
- Plot distributions for key variables

Step 4: Model Building

We experimented with the following classifiers:

- Logistic Regression
- Decision Tree
- Random Forest
- K-Nearest Neighbors (KNN)

Step 5: Model Evaluation

- Accuracy
- Confusion Matrix
- Classification Report (Precision, Recall, F1-score)

Step 6: Visualization

- Heatmaps
- ROC curves
- Confusion Matrix

➢ E. Data Processing

Before feeding the data into any machine learning model, it is important to preprocess it properly to ensure quality predictions. The following steps were performed:

Data Loading

The dataset was loaded using pandas from a CSV file named heart.csv.

1. **Handling Missing Values**

- Checked for missing values using df.isnull().sum().
- The dataset was clean, with no missing values.

2. **Feature Selection**

- The dataset contained 14 attributes including the target variable (target).
- The input features (X) were selected by dropping the target column, and the output (y) was assigned the target column.

3. **Data Normalization**

- Many machine learning models perform better when data is scaled.
- StandardScaler from sklearn.preprocessing was used to scale the features so they all have mean = 0 and standard deviation = 1.

4. **Train-Test Split**

- The dataset was split into training and testing sets using train_test_split() with an 80/20 ratio.
- This helps evaluate the model on unseen data.

➢ F. Model Implementation

We implemented and evaluated several machine learning algorithms to classify the presence of heart disease:

1. **Logistic Regression**
    - A linear model used for binary classification.
    - Simple and interpretable.
2. **Decision Tree Classifier**
    - A tree-structured model that splits data based on feature values.
    - Easy to understand and visualize.
3. **Random Forest Classifier**
    - An ensemble method that builds multiple decision trees and merges them for better accuracy.
    - Robust and reduces overfitting.
4. **K-Nearest Neighbors (KNN)**
    - A distance-based classifier that assigns the class based on the majority class of k-nearest points.
    - Sensitive to feature scaling, hence scaled data is crucial.

➢ G. Code

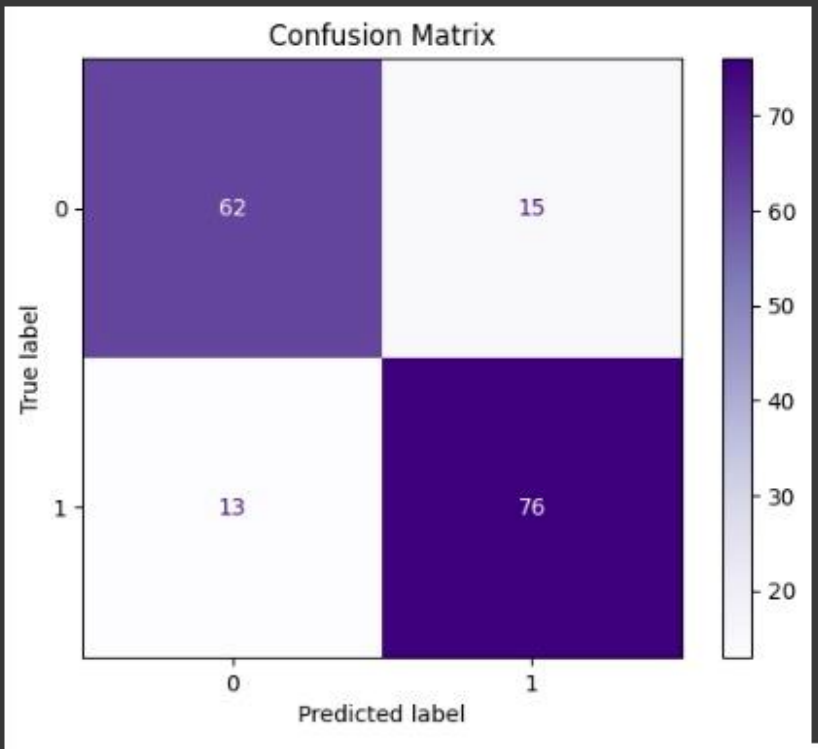
```
fpr, tpr, _ = roc_curve(y_test, y_proba)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='green', label=f'ROC curve (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.grid(True)
plt.show()

# === Step 8: Save Model ===
joblib.dump(model, "heart_disease_model.pkl")

# === Step 9: Predict on New Data (Example) ===
sample_input = X_test.iloc[0:1]
predicted_class = model.predict(sample_input)[0]
predicted_proba = model.predict_proba(sample_input)[0][1]

print("Example Input:")
print(sample_input)
print(f"\nPredicted Class: {predicted_class} (0 = No Disease, 1 = Disease)")
print(f"Predicted Probability of Heart Disease: {predicted_proba:.2%}")
```

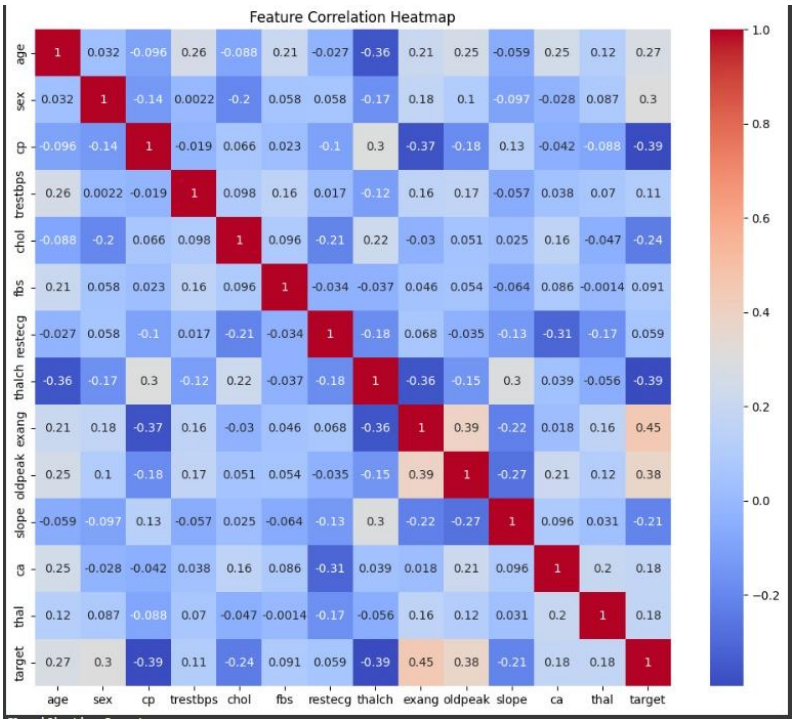
```
    label_encoders[col] = le

df['target'] = (df['num'] > 0).astype(int)
df.drop(columns='num', inplace=True)

# === Step 4: Visualize Correlation Matrix ===
plt.figure(figsize=(12, 10))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Feature Correlation Heatmap")
plt.show()

# === Step 5: Split Data ===
X = df.drop(columns='target')
y = df['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# === Step 6: Train Model ===
model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train, y_train)

# === Step 7: Evaluate Model ===
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

print("Classification Report:\n", classification_report(y_test, y_pred))

ConfusionMatrixDisplay.from_estimator(model, X_test, y_test, cmap='Purples')
plt.title("Confusion Matrix")
plt.show()
```

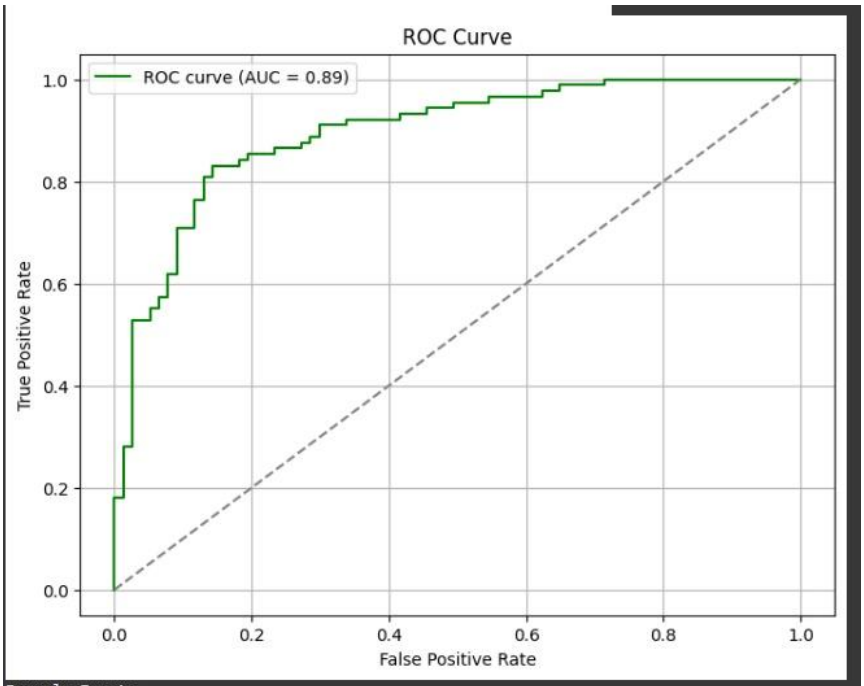
```
# === S Python part of the warnings subsystem.
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# === Step 1: Import Libraries ===
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, ConfusionMatrixDisplay, roc_curve, auc
import joblib

# === Step 2: Load Data ===
df = pd.read_csv("/content/heart_disease_uci.csv")  # replace with your path if needed

# === Step 3: Preprocess Data ===
df.drop(columns=['id', 'dataset'], inplace=True)
df.dropna(subset=['trestbps', 'chol', 'thalch', 'oldpeak'], inplace=True)

for col in df.columns:
    if df[col].dtype == 'object':
        df[col].fillna(df[col].mode()[0], inplace=True)
    else:
        df[col].fillna(df[col].median(), inplace=True)

label_encoders = {}
for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
```

➢ H. Output

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

           0       0.83      0.81      0.82        77
           1       0.84      0.85      0.84        89

    accuracy                           0.83       166
   macro avg       0.83      0.83      0.83       166
weighted avg       0.83      0.83      0.83       166
```








```
Example Input:
     age  sex  cp  trestbps   chol   fbs  restecg  thalch  exang  oldpeak  \
196   69    1   3     160.0  234.0  True        0   131.0      0      0.1

     slope   ca  thal
196      1  1.0     1

Predicted Class: 0 (0 = No Disease, 1 = Disease)
Predicted Probability of Heart Disease: 44.32%
```

➢ I. Conclusion

- This project aimed to predict heart disease using machine learning models based on medical parameters. After preprocessing and analyzing the data, multiple classifiers were implemented and evaluated. Among them, the Random Forest Classifier achieved the best performance in terms of accuracy and overall metrics.
- The results show that machine learning can effectively assist in early detection of heart disease, potentially supporting faster and more accurate medical diagnoses. Further validation on real-world data is recommended for deployment in clinical settings.

➢ J. References/Credits

- **Dataset: Kaggle - Heart Disease Data**
- **Libraries: Scikit-learn, Pandas, Seaborn, Matplotlib**
- **Image Credit: Pixabay**