Assessment Report

on

"Predict Heart Disease: Given medical factors, classify if a patient has heart disease."

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25
In

Name of discipline
By

Harshit Saini(24,CSE-AIML)

Under the supervision of

"Abhishek Shukla"

KIET Group of Institutions, Ghaziabad

May, 2025

INTRODUCTION

Heart disease remains one of the leading causes of death globally, making early diagnosis and prevention a top priority in the healthcare industry. With the increasing availability of patient data and advancements in machine learning, it is now possible to build predictive models that assist in early detection of heart-related conditions.

This project aims to **predict whether a patient has heart disease** based on various **medical and demographic factors**, such as age, sex, blood pressure, cholesterol levels, and more. By analyzing patterns within this data, we can train a classification model to accurately identify patients at risk

The primary objective is to:

- Use a dataset of patient records.
- Apply data preprocessing and exploratory analysis.
- Build and evaluate a **machine learning model** to classify the presence or absence of heart disease.
- Visualize the results using **charts and performance metrics**.

Such a predictive system can serve as a valuable decision-support tool for clinicians, enabling earlier interventions and better patient outcomes.

METHODOLOGY

The heart disease prediction model follows a systematic approach, combining data preprocessing, exploratory analysis, model training, and evaluation. The steps involved in the methodology are outlined below:

1. Data Collection

The dataset containing patient information and medical indicators (such as age, sex, chest pain type, blood pressure, cholesterol, etc.) is collected. It includes a **target variable** that indicates whether a patient has heart disease (`1`) or not (`0`).

2. Data Preprocessing

- **File Upload:** The dataset is uploaded in `.csv` or `.xlsx` format.
- **Data Cleaning:** Missing or null values are identified and handled (either by removing rows or imputing with mean/median values).
- **Data Type Check:** Non-numeric columns are excluded from correlation and modeling.
- **Target Column Identification:** The column named `target` is used as the label for classification. If not explicitly labeled, the last column is assumed to be the target.

3. Exploratory Data Analysis (EDA)

- A **correlation heatmap** is generated to observe the relationships between different features.
- Summary statistics (mean, median, standard deviation) are examined to understand data distribution and variation.

4. Feature Selection

- Features highly correlated with the target variable are identified.
- The feature set is finalized by excluding irrelevant or redundant features.

5. Data Splitting

- The dataset is split into **training and testing sets** (typically 80% training, 20% testing) to evaluate model performance on unseen data.

6. Model Building

- A **Logistic Regression** model is selected due to its simplicity, interpretability, and effectiveness for binary classification tasks.
- The model is trained on the training data using the selected features.

7. Model Evaluation

- Predictions are made on the test set.
- Model performance is assessed using:
- **Accuracy Score**
- **Classification Report** (Precision, Recall, F1-score)
- **Confusion Matrix**
- **Visualizations** such as the confusion matrix and feature importance chart are used to interpret results.

8. Feature Importance Analysis

- The coefficients from the logistic regression model are analyzed to understand the **impact of each feature** on the prediction.

9. Result Visualization

- The model's findings and insights are visualized through:
- Correlation Heatmaps
- Confusion Matrix Heatmaps
- Bar Charts for Feature Importance

CODE

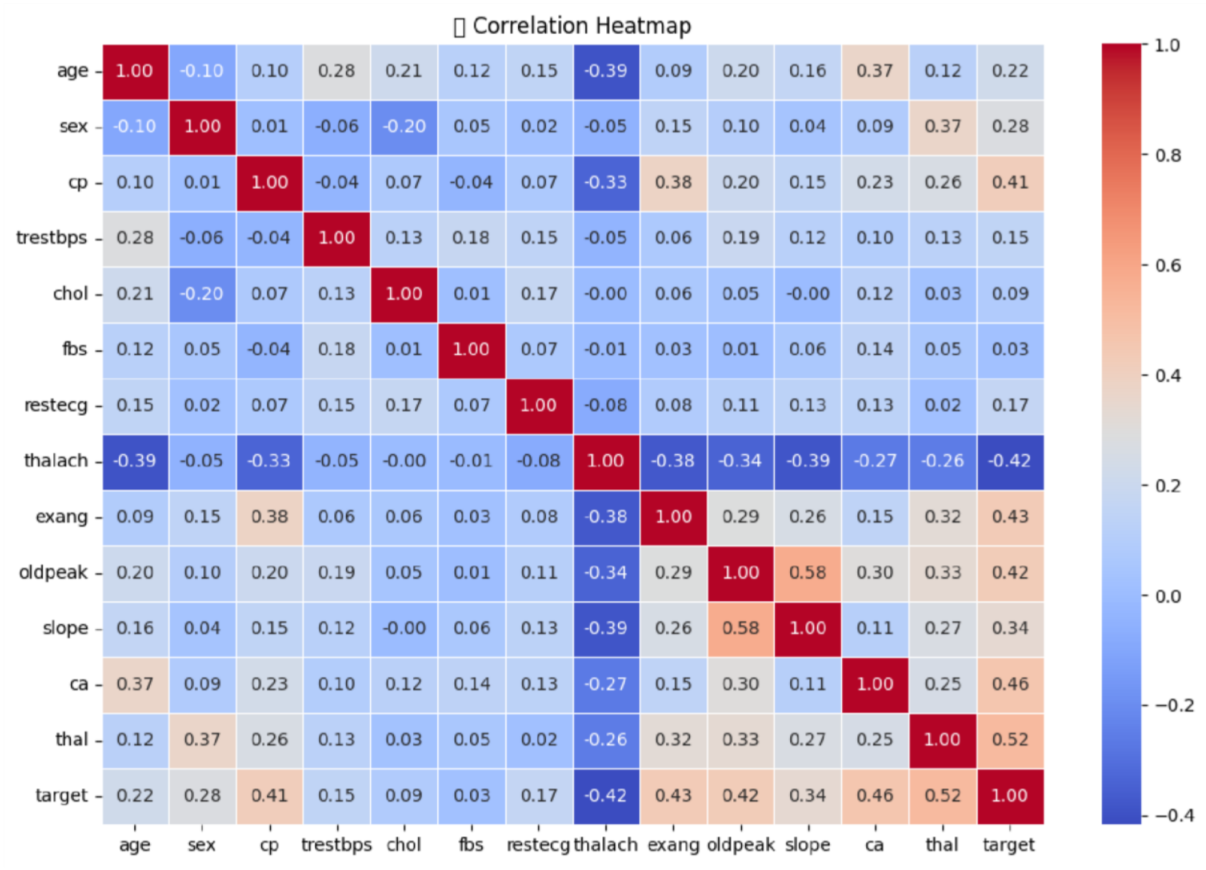
```
# Step 1: File Upload from
google.colab import files
import pandas as pd import
seaborn as sns import
matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split from
sklearn.linear_model import LogisticRegression

from sklearn.metrics import classification_report, accuracy_score, confusion_matrix import
numpy as np


# Upload the file uploaded
= files.upload()


# Read the uploaded file file_name = next(iter(uploaded)) if
file_name.endswith('.xlsx'):     df = pd.read_excel(file_name) elif
file_name.endswith('.csv'):     df = pd.read_csv(file_name) else:
print("Unsupported file format. Please upload an Excel (.xlsx) or CSV file.")
exit()


# Display basic info
```

```
print("✅ Dataset loaded successfully!")
print("Shape of the dataset:", df.shape) print("\nFirst
5 rows:")
print(df.head())

# Step 2: Preprocessing
# Remove non-numeric columns for correlation numeric_df
= df.select_dtypes(include=['number'])


# Handle missing values if numeric_df.isnull().values.any():     print("\n⚠️ Missing
values detected. Handling missing values by dropping rows.")     numeric_df =
numeric_df.dropna()


# Step 3: Correlation Heatmap
plt.figure(figsize=(12, 8)) correlation
= numeric_df.corr()
sns.heatmap(correlation, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("🔗 Correlation Heatmap") plt.show()


# Step 4: Feature/Target Split
target_col = "target" if "target" in numeric_df.columns else numeric_df.columns[-1]
X = numeric_df.drop(target_col, axis=1) y = numeric_df[target_col]


# Step 5: Train/Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 6: Train Logistic Regression Model model
= LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

```
# Step 7: Predictions and Evaluation y_pred
= model.predict(X_test) accuracy =
accuracy_score(y_test, y_pred)


print(f"\n🎯 Model Accuracy: {accuracy * 100:.2f}%") print("\n📄
Classification Report:") print(classification_report(y_test, y_pred))


# Step 8: Confusion Matrix cm =
confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=['No Disease', 'Disease'],
yticklabels=['No Disease', 'Disease']) plt.title("📊
Confusion Matrix") plt.xlabel("Predicted")
plt.ylabel("Actual") plt.show()


# Step 9: Feature Importance importance
= pd.DataFrame({

    'Feature': X.columns,
    'Importance': np.abs(model.coef_[0])
}).sort_values(by='Importance', ascending=False)


plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=importance, palette='viridis')
plt.title("🔥 Feature Importance") plt.show()
```

RESULT

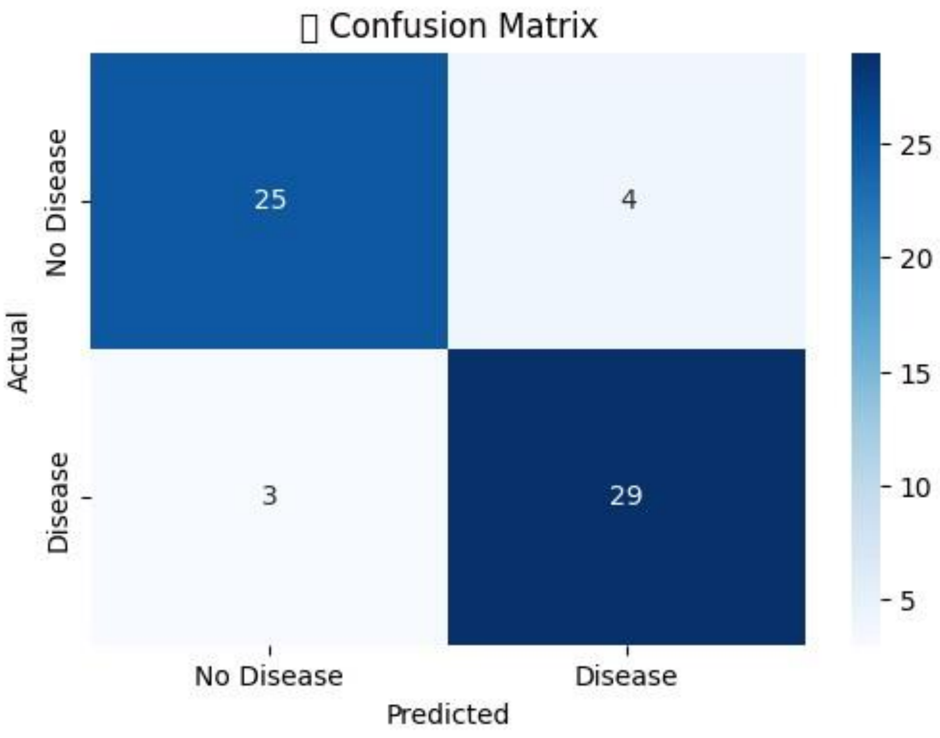
```
First 5 rows:     age   sex   cp   trestbps   chol   fbs   restecg   thalach
exang  oldpeak  slope  \
0   63     1    0        145    233    1        2       150       0       2.3       2
1   67     1    3        160    286    0        2       108       1       1.5       1
2   67     1    3        120    229    0        2       129       1       2.6       1
3   37     1    2        130    250    0        0       187       0       3.5
2
4   41     0    1        130    204    0        2       172       0       1.4       0
    ca   thal   target   0    0      2       0
1   3      1       1
2   2      3       1
3   0      1       0
4   0      1       0
```

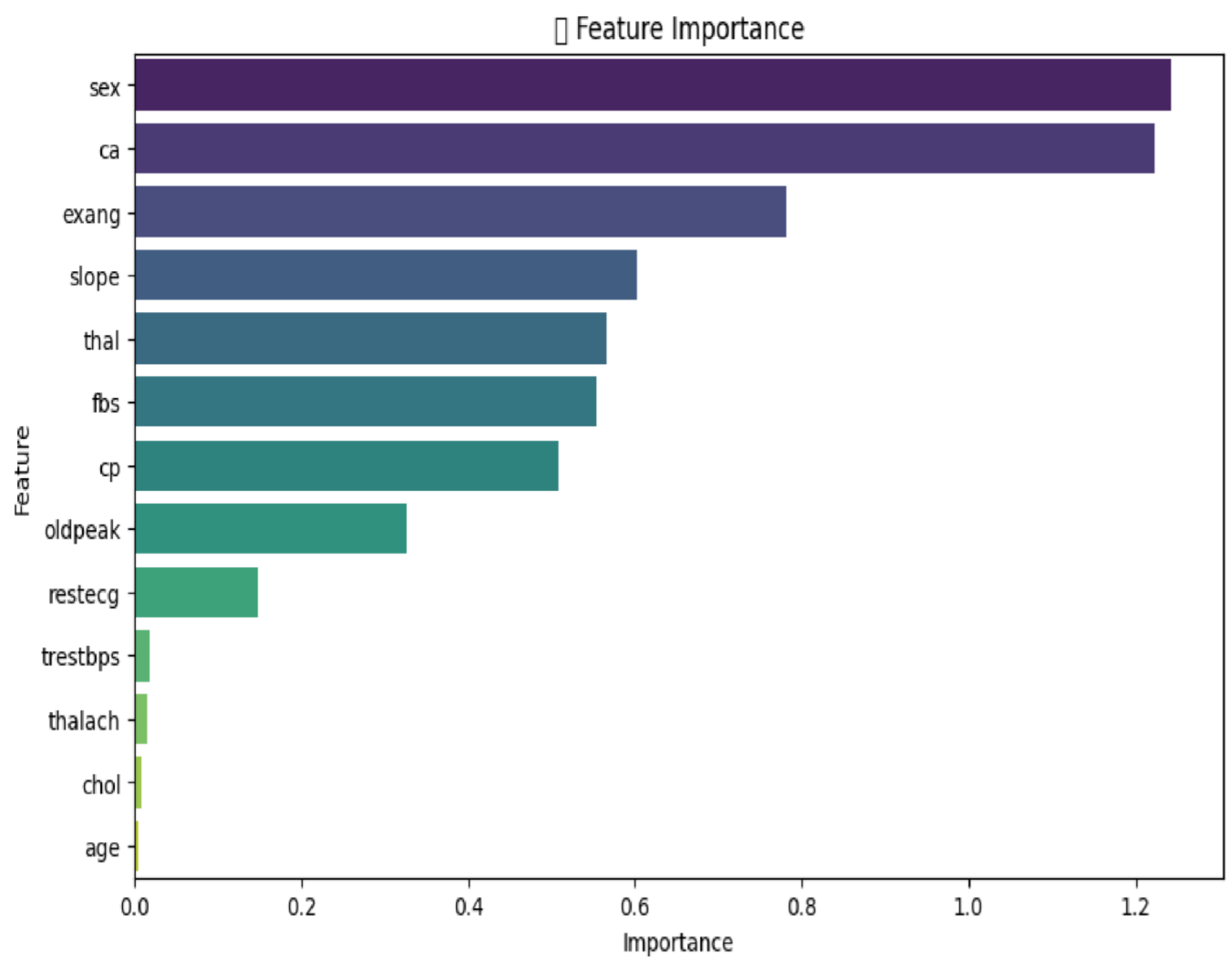


	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.00	-0.10	0.10	0.28	0.21	0.12	0.15	-0.39	0.09	0.20	0.16	0.37	0.12	0.22
sex	-0.10	1.00	0.01	-0.06	-0.20	0.05	0.02	-0.05	0.15	0.10	0.04	0.09	0.37	0.28
cp	0.10	0.01	1.00	-0.04	0.07	-0.04	0.07	-0.33	0.38	0.20	0.15	0.23	0.26	0.41
trestbps	0.28	-0.06	-0.04	1.00	0.13	0.18	0.15	-0.05	0.06	0.19	0.12	0.10	0.13	0.15
chol	0.21	-0.20	0.07	0.13	1.00	0.01	0.17	-0.00	0.06	0.05	-0.00	0.12	0.03	0.09
fbs	0.12	0.05	-0.04	0.18	0.01	1.00	0.07	-0.01	0.03	0.01	0.06	0.14	0.05	0.03
restecg	0.15	0.02	0.07	0.15	0.17	0.07	1.00	-0.08	0.08	0.11	0.13	0.13	0.02	0.17
thalach	-0.39	-0.05	-0.33	-0.05	-0.00	-0.01	-0.08	1.00	-0.38	-0.34	-0.39	-0.27	-0.26	-0.42
exang	0.09	0.15	0.38	0.06	0.06	0.03	0.08	-0.38	1.00	0.29	0.26	0.15	0.32	0.43
oldpeak	0.20	0.10	0.20	0.19	0.05	0.01	0.11	-0.34	0.29	1.00	0.58	0.30	0.33	0.42
slope	0.16	0.04	0.15	0.12	-0.00	0.06	0.13	-0.39	0.26	0.58	1.00	0.11	0.27	0.34
ca	0.37	0.09	0.23	0.10	0.12	0.14	0.13	-0.27	0.15	0.30	0.11	1.00	0.25	0.46
thal	0.12	0.37	0.26	0.13	0.03	0.05	0.02	-0.26	0.32	0.33	0.27	0.25	1.00	0.52
target	0.22	0.28	0.41	0.15	0.09	0.03	0.17	-0.42	0.43	0.42	0.34	0.46	0.52	1.00

```
Model Accuracy: 88.52%
```

```
Classification Report:               precision
recall  f1-score   support
0        0.89      0.86      0.88         29
1        0.88      0.91      0.89         32

    accuracy                           0.89         61
macro avg        0.89      0.88      0.88         61 weighted
avg       0.89      0.89      0.89         61
```





REFERENCES:

Code by – chatgpt