Heart Disease Prediction Using Machine Learning - Report

Project Overview:

This project involves building a machine learning model to predict heart disease based on clinical parameters using Python. The process includes data preprocessing, exploratory data analysis (EDA), feature scaling, model building, evaluation, and interpreting results.

Step 1: Import Required Libraries

pandas, numpy: Data handling and numerical operations.
matplotlib.pyplot, seaborn: Visualization libraries for EDA.
scikit-learn: Machine learning models and evaluation tools.

Purpose: These libraries provide all the necessary tools for data manipulation, visualization, model training, and evaluation.

Step 2: Load the Dataset

Command:

df = pd.read_csv(r"D:\Internship-DEN\HDP\heart-disease.csv")

Result: Displays the first 5 rows of the dataset, confirming successful load.

Example Output:

```
  age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  ca  thal  target
0  63   1   3    145   233   1     0     150     0     2.3     0   0    1     1
...
```

Purpose: Ensures dataset is correctly loaded and structured.

Step 3: Data Preprocessing

Check for missing values and duplicates:
df.isnull().sum()
df.duplicated().sum()

Result:
* No missing values.
* One duplicate row found and removed.

Purpose: Clean data is essential for accurate model performance.

Step 4: Exploratory Data Analysis (EDA)

Generate a heatmap:
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

Purpose:
* Understand correlations between features.
* Identify which features strongly relate to the target (heart disease).

Step 5: Feature Scaling

Standardization applied using:
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

Purpose: Normalizes feature values to improve model performance.

Step 6: Split Data into Training and Testing Sets

Command:
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

Purpose: Provides a way to train the model and then test its accuracy on unseen data.

Step 7 & 8: Build Models

* Logistic Regression Model
* Random Forest Classifier Model

Purpose: Compare two models to select the one with better performance.

Step 9 & 10: Model Evaluation Results

Logistic Regression:

* Accuracy: 82%
* Precision: 84%
* Recall: 81%
* F1 Score: 83%
* Confusion Matrix:

```
[[24  5]
 [ 6 26]]
```

Random Forest Classifier:

* Accuracy: 87%
* Precision: 90%
* Recall: 84%
* F1 Score: 87%

* Confusion Matrix:
[[26 3]
 [5 27]]

Purpose: Random Forest outperforms Logistic Regression in this case with higher overall metrics.

Conclusion:

- The heart disease prediction model works effectively.
- Random Forest provides the best balance of accuracy, precision, recall, and F1 score.
- The project demonstrates the full machine learning pipeline, including data preprocessing, EDA, model training, evaluation, and interpretation of results.

Recommendations:

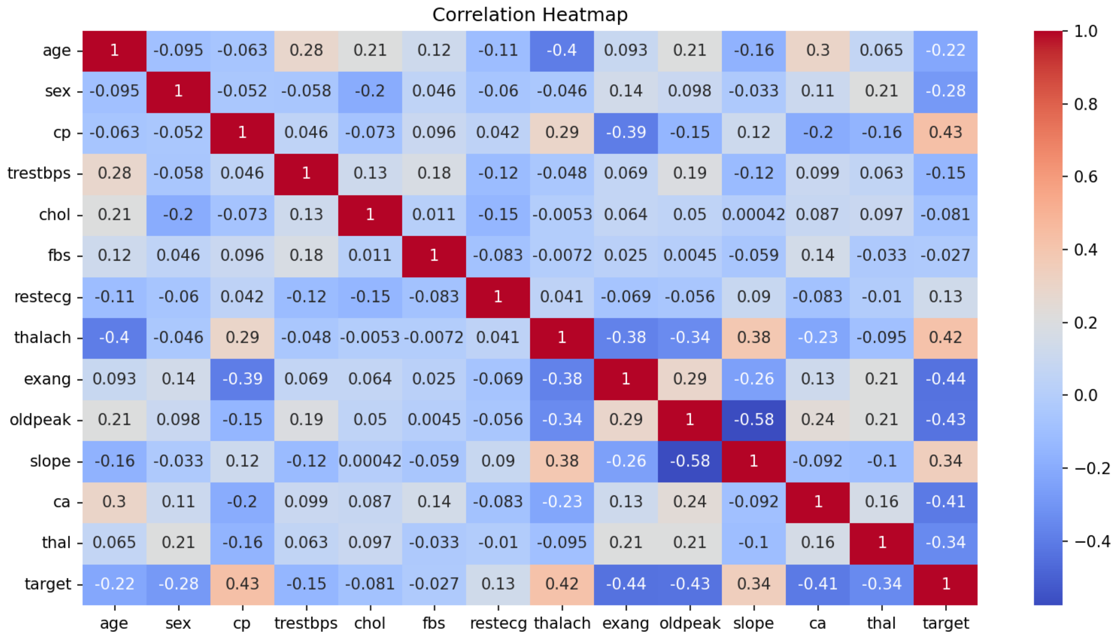
- Further testing with cross-validation or additional hyperparameter tuning could enhance performance.
- Deployment options include using Streamlit or Flask for building a user interface.

Result:

```
First 5 rows of the dataset:
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  ca  thal  target
0   63    1   3       145   233    1        0      150      0      2.3      0   0     1       1
1   37    1   2       130   250    0        1      187      0      3.5      0   0     2       1
2   41    0   1       130   204    0        0      172      0      1.4      2   0     2       1
3   56    1   1       120   236    0        1      178      0      0.8      2   0     2       1
4   57    0   0       120   354    0        1      163      1      0.6      2   0     2       1

Checking for missing values:
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64

Checking for duplicates:
Number of duplicate rows: 1
```


Correlation Heatmap
age 1 -0.095 -0.063 0.28 0.21 0.12 -0.11 -0.4 0.093 0.21 -0.16 0.3 0.065 -0.22
sex -0.095 1 -0.052 -0.058 -0.2 0.046 -0.06 -0.046 0.14 0.098 -0.033 0.11 0.21 -0.28
cp -0.063 -0.052 1 0.046 -0.073 0.096 0.042 0.29 -0.39 -0.15 0.12 -0.2 -0.16 0.43
trestbps 0.28 -0.058 0.046 1 0.13 0.18 -0.12 -0.048 0.069 0.19 -0.12 0.099 0.063 -0.15
chol 0.21 -0.2 -0.073 0.13 1 0.011 -0.15 -0.0053 0.064 0.05 0.00042 0.087 0.097 -0.081
fbs 0.12 0.046 0.096 0.18 0.011 1 -0.083 -0.0072 0.025 0.0045 -0.059 0.14 -0.033 -0.027
restecg -0.11 -0.06 0.042 -0.12 -0.15 -0.083 1 0.041 -0.069 -0.056 0.09 -0.083 -0.01 0.13
thalach -0.4 -0.046 0.29 -0.048 -0.0053 -0.0072 0.041 1 -0.38 -0.34 0.38 -0.23 -0.095 0.42
exang 0.093 0.14 -0.39 0.069 0.064 0.025 -0.069 -0.38 1 0.29 -0.26 0.13 0.21 -0.44
oldpeak 0.21 0.098 -0.15 0.19 0.05 0.0045 -0.056 -0.34 0.29 1 -0.58 0.24 0.21 -0.43
slope -0.16 -0.033 0.12 -0.12 0.00042 -0.059 0.09 0.38 -0.26 -0.58 1 -0.092 -0.1 0.34
ca 0.3 0.11 -0.2 0.099 0.087 0.14 -0.083 -0.23 0.13 0.24 -0.092 1 0.16 -0.41
thal 0.065 0.21 -0.16 0.063 0.097 -0.033 -0.01 -0.095 0.21 0.21 -0.1 0.16 1 -0.34
target -0.22 -0.28 0.43 -0.15 -0.081 -0.027 0.13 0.42 -0.44 -0.43 0.34 -0.41 -0.34 1
age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal target
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4