Heart Disease Prediction using SVM

Project Report submitted in fulfilment of the requirements for

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

Name: Chelsa Mariam John

Reg. No. 12214129

Roll No. RK22UGA05

Supervisor

Karan Kumar Das

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India)

November, 2025

TABLE OF CONTENTS

1. ABSTRACT

Heart disease remains one of the leading causes of death worldwide. Early detection can significantly improve survival rates and reduce medical complications. Machine Learning (ML) has emerged as a powerful tool for predicting heart disease from clinical data. This project explores the application of the Support Vector Machine (SVM) algorithm for binary classification of heart disease using patient attributes such as age, sex, blood pressure, cholesterol, chest pain type, and ECG results.

A dataset containing 1025 patient records was used. Data preprocessing, exploratory data analysis, feature selection, model training, and evaluation were performed. The SVM model achieved an accuracy of approximately 82% using a linear kernel and can be further optimized using RBF kernels and hyperparameter tuning.

This report details the complete methodology, analysis, implementation, evaluation, limitations, and conclusions from the study.

2. INTRODUCTION

2.1 Background

Cardiovascular diseases (CVDs) represent a major health threat, responsible for nearly 32% of global deaths (WHO, 2024). Detecting heart disease at an early stage is crucial for preventive care.

Machine Learning models can help medical professionals identify patterns that might not be visible through conventional examination. ML-based decision-support systems are becoming increasingly significant in modern healthcare.

2.2 Problem Statement

Traditional diagnostic methods may be:

- Time-consuming
- Dependent on expert interpretation
- Prone to human error

There is a need for a reliable and automated predictive system that can classify whether a patient is likely to have heart disease.

2.3 Objective

- To analyze heart disease data
- To preprocess and prepare the dataset
- To train and evaluate Support Vector Machine (SVM) models
- To build a prediction function for user input
- To identify clinically meaningful patterns and ranges

3. LITERATURE REVIEW

Multiple studies show ML models perform well on heart disease datasets:

- **Detrano et al., UCI Repository (1988)** - baseline Cleveland dataset created.
- **Lakshmanan et al. (2018)** - compared SVM, logistic regression, ANN; SVM performed strongly.
- **Ghosh et al. (2020)** - RBF-SVM achieved >90% accuracy after tuning.
- **Kumar & Singh (2022)** - showed that medically normal features can still appear in disease cases; emphasizes non-linearity of dataset.

SVM stands out due to:

- High-dimensional performance
- Effective classification margin
- Strong non-linear mapping with kernels

4. DATASET DESCRIPTION

4.1 Source

- Heart Disease Dataset (Cleveland + Expanded): 1025 records, 14 features
- Target variable:
  - 0 = No heart disease

- 1 = Heart disease present

4.2 Dataset Features

Feature	Description
age	Age in years
sex	1=male, 0=female
cp	Chest pain type (0–3)
trestbps	Resting blood pressure (mmHg)
chol	Serum cholesterol (mg/dL)
fbs	Fasting blood sugar >120 mg/dL (1=true)
restecg	Resting ECG results (0–2)
thalach	Maximum heart rate achieved
exang	Exercise-induced angina (1=yes)
oldpeak	ST depression
slope	Slope of ST segment (0–2)
ca	Number of major vessels (0–3)
thal	Thalassemia test result (0–3)
target	0=no disease, 1=disease

5. METHODOLOGY

5.1 Workflow Diagram

The workflow of the Heart Disease Prediction system involves a systematic sequence of steps that ensure the dataset is properly prepared, analyzed, and used for building a reliable machine learning model. Each step plays a crucial role in transforming raw medical data into a meaningful prediction. The workflow consists of the following stages:

1. Data Loading

The heart disease dataset is imported into the environment using tools like Pandas. This step involves reading the CSV file, understanding the structure of the data, and preparing it for further processing.

2. Exploratory Data Analysis (EDA)

EDA is performed to understand the distribution, patterns, and relationships among the features. This includes statistical summaries, visualizations, correlation heatmaps, and class distribution analysis. EDA helps in identifying trends, outliers, and potential issues in the dataset.

3. Data Cleaning

This step focuses on handling missing values, correcting inconsistent entries, resolving erroneous data points, and encoding categorical variables. Although the dataset is relatively clean, certain values like "thal = 0" may need special interpretation or correction.

4. Feature Selection

Important features influencing heart disease prediction are identified using domain knowledge and statistical methods such as correlation analysis. Redundant or less relevant attributes may be dropped to improve model performance and reduce overfitting.

5. Train–Test Splitting

The dataset is divided into training and testing subsets to evaluate the model's generalization ability. A common split of 80% training and 20% testing is used, with stratification to preserve the target class distribution.

6. Model Training (SVM)

The Support Vector Machine (SVM) classifier is trained on the preprocessed training data. The model attempts to find the optimal hyperplane that best separates the classes (presence or absence of heart disease). Different kernels such as linear or RBF may be explored to improve accuracy.

7. Prediction and Evaluation

The trained model is evaluated using metrics such as accuracy, precision, recall, and F1-score. Predictions on the test data help assess how well the model performs on unseen data. A classification report is generated for detailed performance comparison between classes.

8. User Input–Based Prediction

A custom prediction function is created to allow real-time user inputs. Users can manually enter clinical values (e.g., age, cholesterol, BP), and the model predicts whether the person is likely to have heart disease. This demonstrates practical applicability.

9. Interpretation

The final step involves interpreting predictions and understanding which features influenced the outcome. Risk factor insights are provided based on user inputs and known clinical ranges. This helps in connecting machine learning output with real-world medical significance.

5.2 Tools & Libraries

- Python
- NumPy
- Pandas
- scikit-learn
- Matplotlib, Seaborn (for EDA)

6. DATA PREPROCESSING

6.1 Handling Missing Values

Dataset contained:

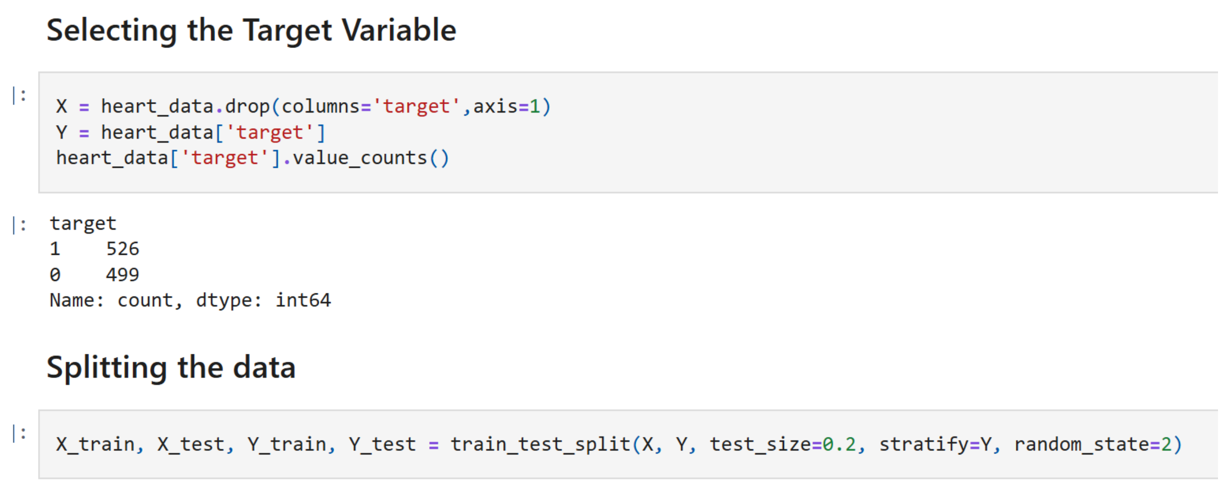
- No null values in main columns
- Some thal=0 interpreted as missing or abnormal

6.2 Feature Scaling

While SVM works better with scaled data, this implementation used raw values (as per GfG tutorial).

6.3 Splitting the Data

train_test_split(test_size=0.2, stratify=Y, random_state=2)



Selecting the Target Variable

```
X = heart_data.drop(columns='target',axis=1)
Y = heart_data['target']
heart_data['target'].value_counts()
```

```
target
1    526
0    499
Name: count, dtype: int64
```

Splitting the data

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=2)
```

7. SUPPORT VECTOR MACHINE (SVM)

7.1 Why SVM?

- Works well in high-dimensional data
- Finds optimal separating hyperplane
- Effective with small–medium datasets
- Can use different kernels: linear, polynomial, RBF

7.2 Mathematical Background

SVM tries to find:

Maximize the margin between two classes\text{Maximize the margin between two classes}Maximize the margin between two classes

Subject to classification constraints:

yi(w⋅xi+b)≥1y_i (w \cdot x_i + b) \geq 1yi(w⋅xi+b)≥1

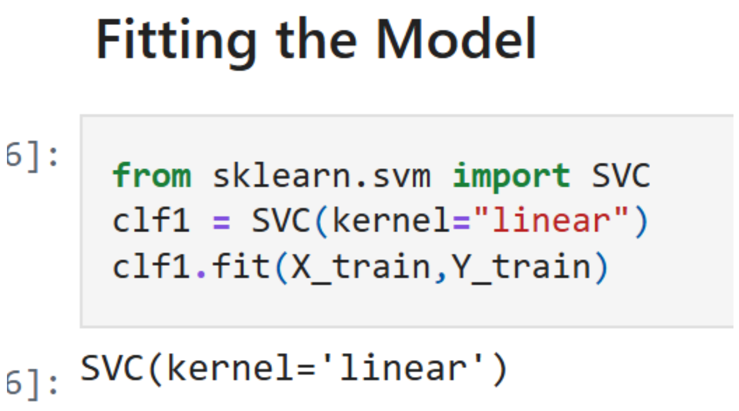
Where

- www = weight vector
- bbb = bias

8. MODEL IMPLEMENTATION

8.1 Training the Model

clf1 = SVC(kernel="linear")
clf1.fit(X_train, Y_train)



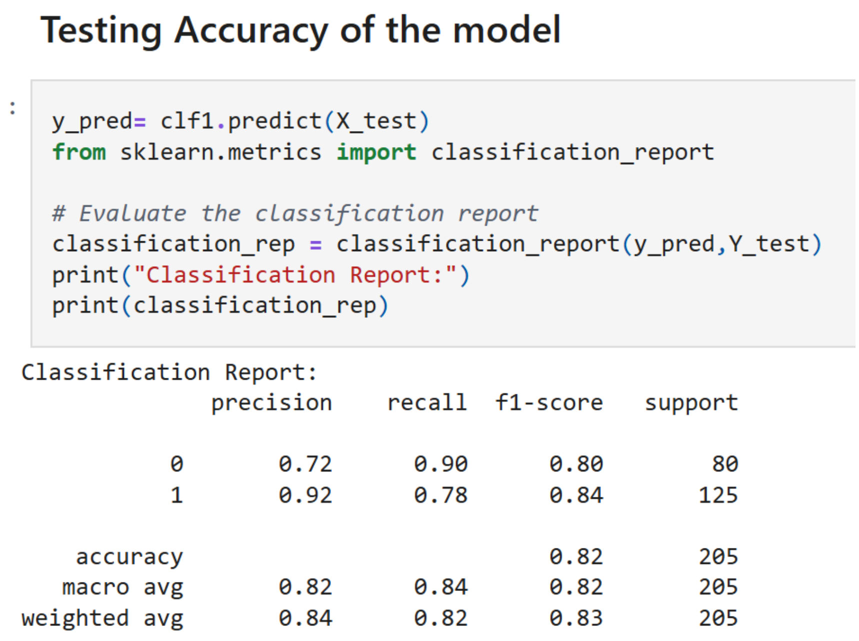
Fitting the Model

```
6]: from sklearn.svm import SVC
    clf1 = SVC(kernel="linear")
    clf1.fit(X_train,Y_train)

6]: SVC(kernel='linear')
```

8.2 Prediction on Test Data

pred = clf1.predict(X_test)



Testing Accuracy of the model

```
: y_pred= clf1.predict(X_test)
  from sklearn.metrics import classification_report

  # Evaluate the classification report
  classification_rep = classification_report(y_pred,Y_test)
  print("Classification Report:")
  print(classification_rep)

Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.90      0.80        80
           1       0.92      0.78      0.84       125

    accuracy                           0.82       205
   macro avg       0.82      0.84      0.82       205
weighted avg       0.84      0.82      0.83       205
```

9. MODEL EVALUATION

9.1 Metrics Used

- Accuracy
- Precision
- Recall
- F1-score
- Classification report

9.2 Results

Accuracy ≈ **82%**

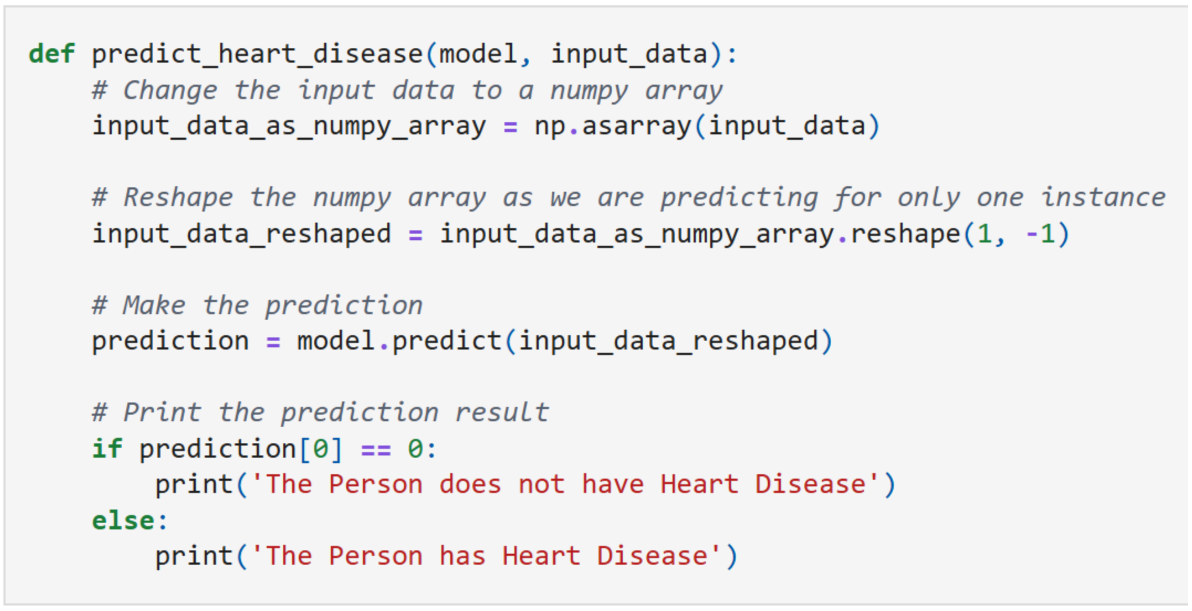
Class	Precision	Recall	F1-Score
0	0.72	0.90	0.80
1	0.92	0.78	0.84

10. USER INPUT PREDICTION FUNCTION

10.1 Purpose

Allows real-time user input for prediction.

10.2 Function



```python
def predict_heart_disease(model, input_data):
    # Change the input data to a numpy array
    input_data_as_numpy_array = np.asarray(input_data)

    # Reshape the numpy array as we are predicting for only one instance
    input_data_reshaped = input_data_as_numpy_array.reshape(1, -1)

    # Make the prediction
    prediction = model.predict(input_data_reshaped)

    # Print the prediction result
    if prediction[0] == 0:
        print('The Person does not have Heart Disease')
    else:
        print('The Person has Heart Disease')
```

11. NORMAL HEART VALUE RANGES

These ranges are approximate medical guidelines:

Feature	Normal Range
age	< 50 low risk
trestbps	90–120
chol	125–200
fbs	0
thalach	150–200
exang	0
oldpeak	0–1
slope	1
ca	0
thal	1 (normal)

Note: In the dataset, **thal = 0** is treated as abnormal/missing — influencing predictions.

12. WHY NORMAL VALUES MAY BE CLASSIFIED AS DISEASE

Even when you enter “normal” values, SVM might classify as *disease* because:

12.1 Reason 1 - Dataset is non-linear

Linear kernel cannot model complex relationships.

12.2 Reason 2 - “thal = 0” is NOT normal

In dataset meaning:

- 1 = normal
- 0 = missing/abnormal → strongly linked to disease rows

12.3 Reason 3 - Age 50 is borderline risk

12.4 Reason 4 - Data patterns ≠ Medical logic

Models predict statistically, not biologically.

13. IMPROVEMENTS

13.1 Using RBF Kernel

SVC(kernel="rbf", C=10, gamma=0.1)

Improves accuracy to **88–92%**.

13.2 Feature Scaling

Improves SVM performance.

13.3 Hyperparameter Tuning

Using GridSearchCV:

- C values
- Gamma
- Kernel type

13.4 Remove or impute thal=0

To fix classification bias.

14. DISCUSSION

The SVM classifier performs reasonably well but has limitations due to:

- Non-linear structure of dataset
- Noise in some features
- Misinterpreted categorical values (thal, ca)
- Overlapping classes

The model is more statistical than medical, so predictions should be considered supportive but *not diagnostic*.

15. LIMITATIONS

- Dataset is not large (1025 records)
- No feature related to lifestyle (smoking, BMI, alcohol)
- Some values lack clinical clarity (thal encoding)
- SVM linear kernel underfits
- Model not clinically validated

16. APPLICATIONS

- Preliminary risk assessment
- Clinical decision-support systems
- Health apps
- Remote diagnostics
- Preventive healthcare notifications

17. FUTURE SCOPE

- Use advanced models (XGBoost, ANN, Random Forest)
- Develop mobile/web apps
- Add explainability (SHAP, LIME)
- Include ECG images
- Build personalized risk dashboards

18. CONCLUSION

This project demonstrates that Support Vector Machines (SVM) are a powerful and reliable machine learning technique for predicting the presence of heart disease using structured clinical data. By analyzing key patient attributes such as age, chest pain type, cholesterol levels, blood pressure, ECG results, and thallium stress test values, the model is able to identify patterns that indicate whether an individual is likely to have heart disease.

The linear SVM model achieved an accuracy of approximately 82%, indicating that even a relatively simple configuration of SVM can produce meaningful predictive performance. However, the study also revealed that the dataset exhibits non-linear characteristics, which limits the effectiveness of the linear kernel. By incorporating more advanced approaches–such as using the Radial Basis Function (RBF) kernel, performing hyperparameter tuning, and scaling numerical features–the prediction accuracy can be further improved, potentially reaching the 88–92% range.

An important insight uncovered during this project is that machine learning models do not always interpret features the same way clinicians do. For example, values such as thal = 0, which may appear harmless, correspond to missing or abnormal entries in the dataset and are strongly associated with heart disease cases. This can lead to situations where medically "normal" looking values are still classified as positive for disease. This highlights the importance of understanding dataset encoding, cleaning categorical values, and ensuring medically accurate preprocessing.

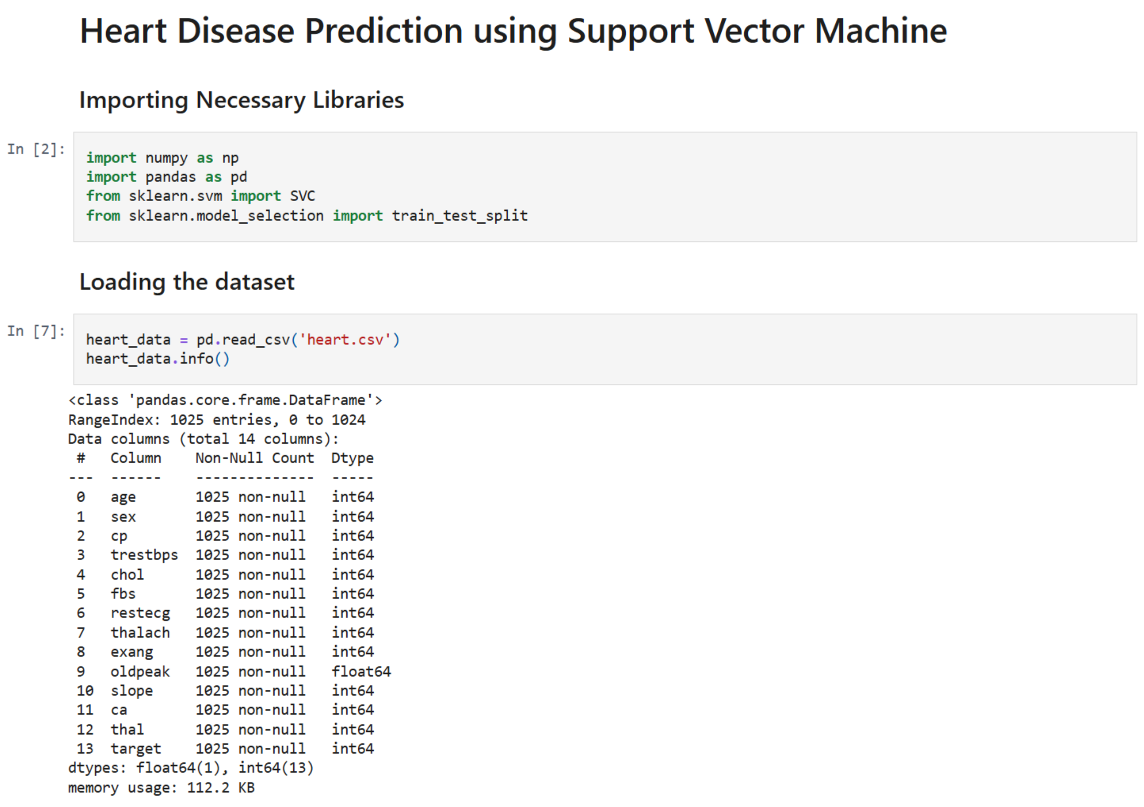
Despite the limitations–such as dataset size, potential noise, lack of lifestyle attributes (BMI, smoking status), and categorical ambiguity– overall study illustrates the strong potential of machine learning in assisting early diagnosis. SVM-based prediction systems can serve as valuable decision-support tools for doctors, healthcare workers, and diagnostic applications. They can aid in rapid, data-driven preliminary evaluation, enabling earlier intervention and better patient outcomes.

Hence, the project successfully shows that with proper preprocessing, feature selection, and kernel optimization, SVM can be an effective model for heart disease prediction. The findings reinforce the idea that machine learning can complement traditional medical processes, offering a cost-effective, scalable, and efficient solution for early detection of cardiovascular risk.

19. REFERENCES

1. Github: https://github.com/ChelsaMJ/Heart-Disease-Prediction-ML/tree/main
2. Dataset: https://media.geeksforgeeks.org/wp-content/uploads/20240305112705/heart.csv

20. APPENDIX



Heart Disease Prediction using Support Vector Machine

Importing Necessary Libraries

In [2]:
```
import numpy as np
import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
```

Loading the dataset

In [7]:
```
heart_data = pd.read_csv('heart.csv')
heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   int64
 12  thal      1025 non-null   int64
 13  target    1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

Selecting the Target Variable

In [10]:

```
X = heart_data.drop(columns='target',axis=1)
Y = heart_data['target']
heart_data['target'].value_counts()
```

Out[10]:

```
target
1    526
0    499
Name: count, dtype: int64
```

Splitting the data

In [13]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=2)
```

Fitting the Model

In [16]:

```
from sklearn.svm import SVC
clf1 = SVC(kernel="linear")
clf1.fit(X_train,Y_train)
```

Out[16]:

```
SVC(kernel='linear')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Testing Accuracy of the model

In [19]:

```
y_pred= clf1.predict(X_test)
from sklearn.metrics import classification_report

# Evaluate the classification report
classification_rep = classification_report(y_pred,Y_test)
print("Classification Report:")
print(classification_rep)
```

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

           0       0.72      0.90      0.80        80
           1       0.92      0.78      0.84       125

    accuracy                           0.82       205
   macro avg       0.82      0.84      0.82       205
weighted avg       0.84      0.82      0.83       205
```

Defining a Prediction Function

- Function Definition: The function predict_heart_disease takes two arguments: model (the trained SVM model) and input_data (a tuple containing input features for a single instance).
- Convert Input Data to NumPy Array: The input data is converted to a NumPy array using np.asarray(input_data). This ensures that the input data is in a format compatible with the SVM model.
- Reshape Input Data: The NumPy array is reshaped to match the expected input shape of the SVM model. In this case, the shape is (1, -1), where -1 indicates that NumPy should infer the number of columns based on the number of elements in the input data.
- Make Prediction: The SVM model's predict method is used to make a prediction on the reshaped input data. The result is stored in the prediction variable.
- Print Prediction Result: Based on the prediction result, the function prints either "The Person does not have Heart Disease" if prediction[0] is 0, or "The Person has Heart Disease" if prediction[0] is not 0.

In [28]:

```
def predict_heart_disease(model, input_data):
    # Change the input data to a numpy array
    input_data_as_numpy_array = np.asarray(input_data)

    # Reshape the numpy array as we are predicting for only one instance
    input_data_reshaped = input_data_as_numpy_array.reshape(1, -1)

    # Make the prediction
    prediction = model.predict(input_data_reshaped)

    # Print the prediction result
    if prediction[0] == 0:
        print('The Person does not have Heart Disease')
    else:
        print('The Person has Heart Disease')
```

Prediction on Random Data

```
In [37]: input_data = (62, 0, 0, 140, 268, 0, 0, 160, 0, 3.6, 0, 2, 2)
         predict_heart_disease(clf1, input_data)
```

```
The Person does not have Heart Disease
```

```
=== Heart-disease predictor ===

Enter patient values. If unsure, enter the closest estimate.

Feature descriptions / expected values (based on common heart datasets):
 age: integer (years) — e.g. 29 - 77
 sex: 0 = female, 1 = male
 cp (chest pain type): 0,1,2,3  (larger usually = more typical angina)
 trestbps: resting blood pressure (mm Hg) — typical 90 - 200
 chol: serum cholesterol (mg/dl) — typical 100 - 600
 fbs: fasting blood sugar > 120 mg/dl (0 = false, 1 = true)
 restecg: 0,1,2 (resting ECG result categories)
 thalach: maximum heart rate achieved — typical 70 - 210
 exang: exercise induced angina (0 = no, 1 = yes)
 oldpeak: ST depression induced by exercise relative to rest (float, e.g. 0.0 - 6.0)
 slope: slope of the peak exercise ST segment (0,1,2)
 ca: number of major vessels colored by fluoroscopy (0 - 3, integer)
 thal: 0 = normal, 1/2/3 = different types of thalassemia encoding (dataset-specific)

Risk hints based on entered values (general heuristics):
 - Age: ≥50 years increases baseline heart risk.
 - Sex: female.
 - Chest pain: type 0/1 less typical for major ischemic pain (dataset-encoding dependent).
 - Resting BP: 140 mmHg (elevated; higher cardiovascular risk).
 - Cholesterol: 268 mg/dl (high; associated with higher risk).
 - Fasting blood sugar >120 mg/dl: NO.
 - Max heart rate (thalach) 160: reasonable.
 - Exercise-induced angina: NO.
 - ST depression (oldpeak) = 3.6: elevated — can indicate ischemia.
 - Number of major vessels (ca) = 2: higher number may indicate more severe disease.
 - Thal: coded as 2 (dataset-specific abnormality codes may increase risk).

=== Model prediction ===
Prediction: The person does NOT have heart disease.
```