Heart Disease Prediction using Logistic Regression – Detailed Explanation

1. Importing Required Libraries

```
import numpy as np

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score
```

Explanation:

- numpy: Used for numerical operations, especially arrays and reshaping.
- pandas: Used for loading and manipulating data in tabular (CSV) form.
- train_test_split: Used to divide data into training and testing parts.
- LogisticRegression: The classification algorithm we're using.
- accuracy_score: Calculates how accurate the model's predictions are.

2. Loading the Dataset

```
heart_data = pd.read_csv('/content/data.csv')
```

Explanation:

- Loads the CSV file into a Pandas DataFrame.
- heart_data now contains all rows and columns from the CSV file.

3. Basic Data Exploration

```
heart_data.head()

heart_data.tail()

heart_data.shape

heart_data.info()

heart_data.isnull().sum()

heart_data.describe()

heart_data['target'].value_counts()
```

Explanation:

- head(): Shows first 5 rows — to inspect the beginning of the dataset.
- tail(): Shows last 5 rows — to inspect the end of the dataset.
- shape: Returns (rows, columns). Example: (303, 14).
- info(): Displays data types, non-null counts — to check data health.
- isnull().sum(): Shows missing values in each column — should be 0 ideally.
- describe(): Gives statistical summary — mean, median, max, etc.
- value_counts() on target: Shows class distribution (0 = no disease, 1 = disease).

4. Separating Features and Labels

```
X = heart_data.drop(columns='target', axis=1)

Y = heart_data['target']
```

Explanation:

- X: All **independent variables** (features) — inputs used to make predictions.
- Y: The **dependent variable** (label/target) — output to be predicted.
- drop(columns='target'): Removes the target column from features.

5. Splitting the Dataset

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

Explanation:

- train_test_split: Splits the data into training (80%) and testing (20%).
- stratify=Y: Ensures both sets have same class distribution.
- random_state=2: For reproducibility (same split every time).
- X_train, Y_train: Used to train the model.
- X_test, Y_test: Used to test model performance.

6. Training the Model

```
model = LogisticRegression()

model.fit(X_train, Y_train)
```

Explanation:

- Creates a Logistic Regression model.
- .fit(): Trains the model using training data (X and Y).

7. Model Evaluation – Training Accuracy

```
X_train_prediction = model.predict(X_train)

training_data_accuracy = accuracy_score(X_train_prediction, Y_train)

print('Accuracy on Training data : ', training_data_accuracy)
```

Explanation:

- predict(X_train): Predicts the target for training set.
- accuracy_score(): Compares predicted vs actual targets.
- Gives accuracy of model on **training data**.

8. Model Evaluation – Test Accuracy

```
X_test_prediction = model.predict(X_test)

test_data_accuracy = accuracy_score(X_test_prediction, Y_test)

print('Accuracy on Test data : ', test_data_accuracy)
```

Explanation:

- predict(X_test): Predicts on test data (unseen by model).
- Checks if the model is overfitting or generalizing well.

9. Predicting for a New Data Sample

```
input_data = (62,0,0,140,268,0,0,160,0,3.6,0,2,2)

input_data_as_numpy_array = np.asarray(input_data)

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)
```

Explanation:

- input_data: A new patient's features (in correct order as dataset).
- np.asarray(): Converts tuple to numpy array.

- .reshape(1, -1): Reshapes the array to 2D as model expects multiple rows.
    - 1: number of rows (here, 1 person)
    - -1: auto-calculates number of columns
- model.predict(): Predicts whether this person has heart disease (0 or 1).

10. Final Output Display

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

Explanation:

- If model prediction is 0: No disease
- If model prediction is 1: Disease is present
- Human-readable output for end user

Complete code:

https://colab.research.google.com/drive/100RbLbvh8x8HmdA146oKw8mZ9twgIcgC?usp=sharing