

## **KSI: Machine Learning & AI Dual certification**

Project build under the guidance of Professor Nitesh Karmakar

### **Project 1: Heart Disease diagnosis**

*Machine Learning: Logistic Regression*

**Name:** Mohammed Faraz Hussain

**College:** Terna Engineering College, Nerul

**Roll number:** TE-B-24

**ID number:** TU3F1920077

**Email:** [farazhussain503@gmail.com](mailto:farazhussain503@gmail.com)

**Terna Mail:** [mohammedhussain@ternaengg.ac.in](mailto:mohammedhussain@ternaengg.ac.in)

**Phone number:** +91 9082905016 / +91 7303272780

# Table of Contents

<b>Sr.No</b>	<b>Title</b>
1	Introduction
2	Aim and Objective
3	About Dataset
4	Design
5	Implementation
6	Conclusion
7	References

# **Introduction**

Every day, the average human heart beats around 100,000 times, pumping 2,000 gallons of blood through the body. Inside your body there are 60,000 miles of blood vessels.

The signs of a woman having a heart attack are much less noticeable than the signs of a male. In women, heart attacks may feel uncomfortable squeezing, pressure, fullness, or pain in the center of the chest. It may also cause pain in one or both arms, the back, neck, jaw or stomach, shortness of breath, nausea and other symptoms. Men experience typical symptoms of heart attack, such as chest pain, discomfort, and stress. They may also experience pain in other areas, such as arms, neck, back, and jaw, and shortness of breath, sweating, and discomfort that mimics heartburn.

The World Health Organization has estimated that four out of five cardiovascular diseases (CVD) deaths are due to heart attacks.

## **Aim and Objective**

The prediction of heart disease is considered one of the most important topics in the health domain.

Also, The amount of data in the healthcare industry is huge.

Heart disease is a major concern to be dealt with. But it is difficult to identify heart disease because of several contributory risk factors such as diabetes, high blood pressure, high cholesterol, abnormal pulse rate, and many other factors. Due to such constraints, scientists have turned towards modern approaches like Data Mining and Machine Learning for predicting the disease.

Hence,

With machine learning algorithms and having large amounts of data, it is possible to extrapolate information that can help doctors make more accurate predictions.

## **About Dataset**

Heart disease describes a range of conditions that affect your heart.

- **Age:** The person's age in years
- **Sex:** The person's sex (1 = male, 0 = female)
- **Chest Pain/Angina:** The chest pain experienced
  - ◆ Value 1: typical
  - ◆ Value 2: nontypical
  - ◆ Value 3: asymptomatic
  - ◆ Value 4: nonanginal
- **RestBP:** The person's resting blood pressure (mm Hg on admission to the hospital)
- **Chol:** The person's cholesterol measurement in mg/dl
- **Fbs:** The person's fasting blood sugar (> 120 mg/dl, 1 = true; 0 = false)
- **RestECG:** Resting electrocardiographic measurement
  - ◆ 0 = normal
  - ◆ 1 = having ST-T wave abnormality
  - ◆ 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria

- **MaxHR:** The person's maximum heart rate achieved
- **ExAng:** Exercise induced angina (1 = yes; 0 = no)
- **Oldpeak:** ST depression induced by exercise relative to rest ('ST' relates to positions on the ECG plot.
- **Slope:** the slope of the peak exercise ST segment
  - ◆ Value 1: upsloping
  - ◆ Value 2: flat
  - ◆ Value 3: downsloping
- **Ca:** The number of major vessels (0-3)
- **Thal:** A blood disorder called thalassemia
  - ◆ Value 1: normal
  - ◆ Value 2: fixed defect
  - ◆ Value 3: reversible defect

# Design

Since the AHD column exists, which depends on the independent 13 features. Hence, it is a dependent variable.

As a dependent variable exists, it falls into the category of supervised learning.

Also, the output of the model is either **Yes** or **No**. It is a binary classification.

Hence, we will use Logistic Regression for this problem.

Logistic Regression is a statistical and machine-learning technique classifying records of a dataset based on the values of the input fields.

It predicts a dependent variable based on one or more sets of independent variables to predict outcomes .

It can be used both for binary classification and multi-class classification

In this project, we have to predict whether the person has heart disease or not, which means, it falls into the category of binary classification(Since, we have to predict- Yes/No).

# Implementation

To solve a machine learning model, we need to follow few steps. They are as follows:

1. Importing libraries
2. Importing Dataset and Read the data (from csv)
3. Identify the dependent and independent variables.
4. Check if the data has missing values or the data is categorical or not.
5. Visualize the data.
6. Now split the data into the groups of training and testing for the respective purpose.
7. After splitting data, fit it to a most suitable model.
8. Prediction
9. Model Evaluation



## 1. Importing Libraries

```
In [1]: 1 import numpy as np
        2 import pandas as pd
        3 import seaborn as sns
        4 import matplotlib.pyplot as plt
        5 import seaborn as sns
        6 import warnings

In [2]: 1 warnings.filterwarnings('ignore')
```

## 2. Importing Dataset and Read the data (from csv)

```
In [3]: 1 # importing csv data and view data
        2 data = pd.read_csv("heart_disease.csv.txt")
        3 data
```

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
299	45	1	typical	110	264	0	0	132	0	1.2	2	0.0	reversable	Yes
300	68	1	asymptomatic	144	193	1	0	141	0	3.4	2	2.0	reversable	Yes
301	57	1	asymptomatic	130	131	0	0	115	1	1.2	2	1.0	reversable	Yes
302	57	0	nontypical	130	236	0	2	174	0	0.0	2	1.0	normal	Yes
303	38	1	nonanginal	138	175	0	0	173	0	0.0	1	NaN	normal	No

303 rows x 14 columns

Acti  
Go to

## 2.1 Checking rows and columns

```
In [4]: 1 print("(Rows, columns): " + str(data.shape)) # rows = 303, columns = 14
        2 data.columns # features

(Rows, columns): (303, 14)

Index(['Age', 'Sex', 'ChestPain', 'RestBP', 'Chol', 'Fbs', 'RestECG', 'MaxHR',
       'ExAng', 'Oldpeak', 'Slope', 'Ca', 'Thal', 'AHD'],
      dtype='object')
```

## 2.2 Checking first 5 values from the imported dataset

```
In [5]: 1 data.head()
```

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No

## 3. Identify the dependent and independent variables.

Independent variables:

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal
1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed
2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal
3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable
4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal
5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal

**Dependent variables:**

AHD
No
Yes
Yes
No
No

## 4. Check if the data has missing values or the data is categorical or not.

### Changing string data type to numeric data type:

- Data generally needs to be put into numeric form for machine learning algorithms to use the data to make predictions.
  - Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers.
- 

```
In [6]: 1 from sklearn.preprocessing import LabelEncoder
        2 lb = LabelEncoder()
```

```
In [7]: 1 # converting the string data of chestpain into integer data
        2 # ChestPain = {
        3 #     'typical': 0,
        4 #     'nontypical': 1,
        5 #     'asymptomatic': 2,
        6 #     'nonanginal': 3
        7 # }
        8 # data.ChestPain = [ChestPain[item] for item in data.ChestPain]
        9
       10 # -----
       11
       12 data["ChestPain"] = lb.fit_transform(data["ChestPain"])
```

## Before LabelEncoding:

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal
1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed
2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal
3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable
4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal
5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal

## After LabelEncoding:

```
In [8]: 1 data.head()
```

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
1	63	1	3	145	233	1	2	150	0	2.3	3	0.0	fixed	No
2	67	1	0	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
3	67	1	0	120	229	0	2	129	1	2.6	2	2.0	reversible	Yes
4	37	1	1	130	250	0	0	187	0	3.5	3	0.0	normal	No
5	41	0	2	130	204	0	2	172	0	1.4	1	0.0	normal	No

-----

```

In [10]: 1 # converting the string data of thal into int data
          2 # Thal = {
          3 #     'fixed': 0,
          4 #     'normal': 1,
          5 #     'reversible': 2
          6 # }
          7 # data.Thal = [Thal[item] for item in data.Thal]
          8
          9 # -----
         10
         11 data["Thal"] = lb.fit_transform(data["Thal"])

```

Before LabelEncoding:

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal
1	63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed
2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal
3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversible
4	37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal
5	41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal

## After LabelEncoding:

```
In [11]: 1 data.head()
```

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
1	63	1	3	145	233	1	2	150	0	2.3	3	0.0	0	No
2	67	1	0	160	286	0	2	108	1	1.5	2	3.0	1	Yes
3	67	1	0	120	229	0	2	129	1	2.6	2	2.0	2	Yes
4	37	1	1	130	250	0	0	187	0	3.5	3	0.0	1	No
5	41	0	2	130	204	0	2	172	0	1.4	1	0.0	1	No

## 4.1: Dropping NaN/Null Values

```
In [11]: 1 # dropping NaN values  
2 data = data.dropna()  
3 data.isna().sum()
```

```
Age      0  
Sex      0  
ChestPain 0  
RestBP   0  
Chol     0  
Fbs      0  
RestECG  0  
MaxHR    0  
ExAng    0  
Oldpeak  0  
Slope    0  
Ca       0  
Thal     0  
AHD      0  
dtype: int64
```

## 4.2: Unique values for each variable:

```
In [12]: 1 # returning the number of unique values for each variable.  
        2 data.nunique(axis=0)
```

```
Age      41  
Sex       2  
ChestPain  4  
RestBP    50  
Chol     152  
Fbs       2  
RestECG   3  
MaxHR     91  
ExAng     2  
Oldpeak   40  
Slope     3  
Ca        4  
Thal      4  
AHD       2  
dtype: int64
```



### 4.3: Concise summary of data

```
In [13]: 1 # checking the concise summary of data
          2 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 299 entries, 1 to 302
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Age         299 non-null    int64
1   Sex         299 non-null    int64
2   ChestPain   299 non-null    int32
3   RestBP      299 non-null    int64
4   Chol        299 non-null    int64
5   Fbs         299 non-null    int64
6   RestECG     299 non-null    int64
7   MaxHR       299 non-null    int64
8   ExAng       299 non-null    int64
9   Oldpeak     299 non-null    float64
10  Slope       299 non-null    int64
11  Ca          299 non-null    float64
12  Thal        299 non-null    int32
13  AHD         299 non-null    object
dtypes: float64(2), int32(2), int64(9), object(1)
memory usage: 32.7+ KB
```

### 4.4: Rows and columns in the dataset after dropping NaN values

```
In [16]: 1 # number of rows and columns in the dataset
          2 data.shape
```

```
(299, 14)
```

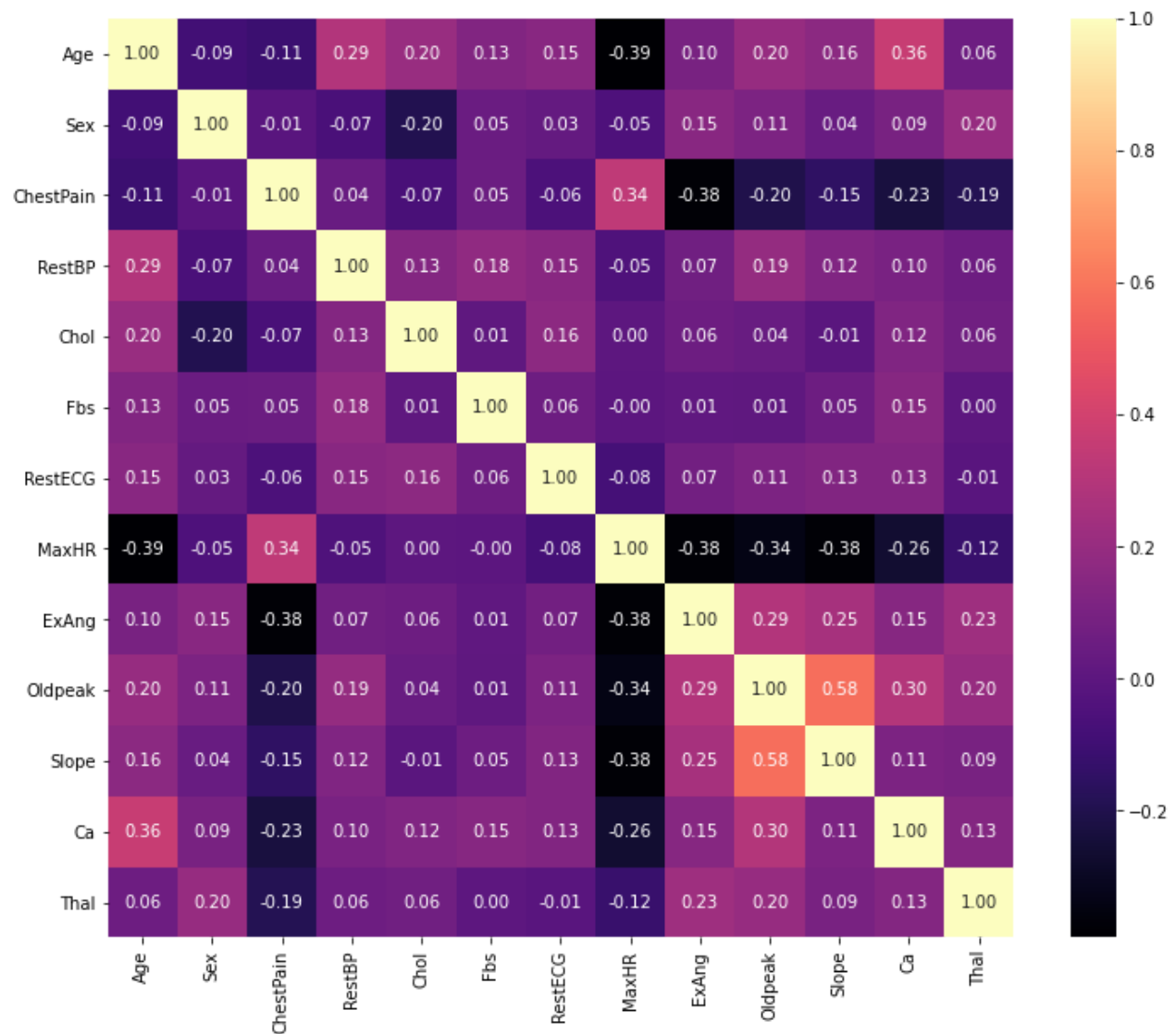
## 4.5: Statistical measures about the data

```
In [19]: 1 # statistical measures about the data
        2 data.describe()
```

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG
count	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000
mean	54.528428	0.675585	0.839465	131.668896	247.100334	0.147157	0.996656
std	9.020950	0.468941	0.962893	17.705668	51.914779	0.354856	0.994948
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000
25%	48.000000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000
50%	56.000000	1.000000	1.000000	130.000000	242.000000	0.000000	1.000000
75%	61.000000	1.000000	1.000000	140.000000	275.500000	0.000000	2.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000

MaxHR	ExAng	Oldpeak	Slope	Ca	Thal
299.000000	299.000000	299.000000	299.000000	299.000000	299.000000
149.505017	0.327759	1.051839	1.602007	0.672241	1.337793
22.954927	0.470183	1.163809	0.617526	0.937438	0.598889
71.000000	0.000000	0.000000	1.000000	0.000000	0.000000
133.000000	0.000000	0.000000	1.000000	0.000000	1.000000
153.000000	0.000000	0.800000	2.000000	0.000000	1.000000
165.500000	1.000000	1.600000	2.000000	1.000000	2.000000
202.000000	1.000000	6.200000	3.000000	3.000000	3.000000

#### 4.6: Correlation matrix: Correlation indicates how the features are related to each other or to the target variable.



## 5. Visualizing the data

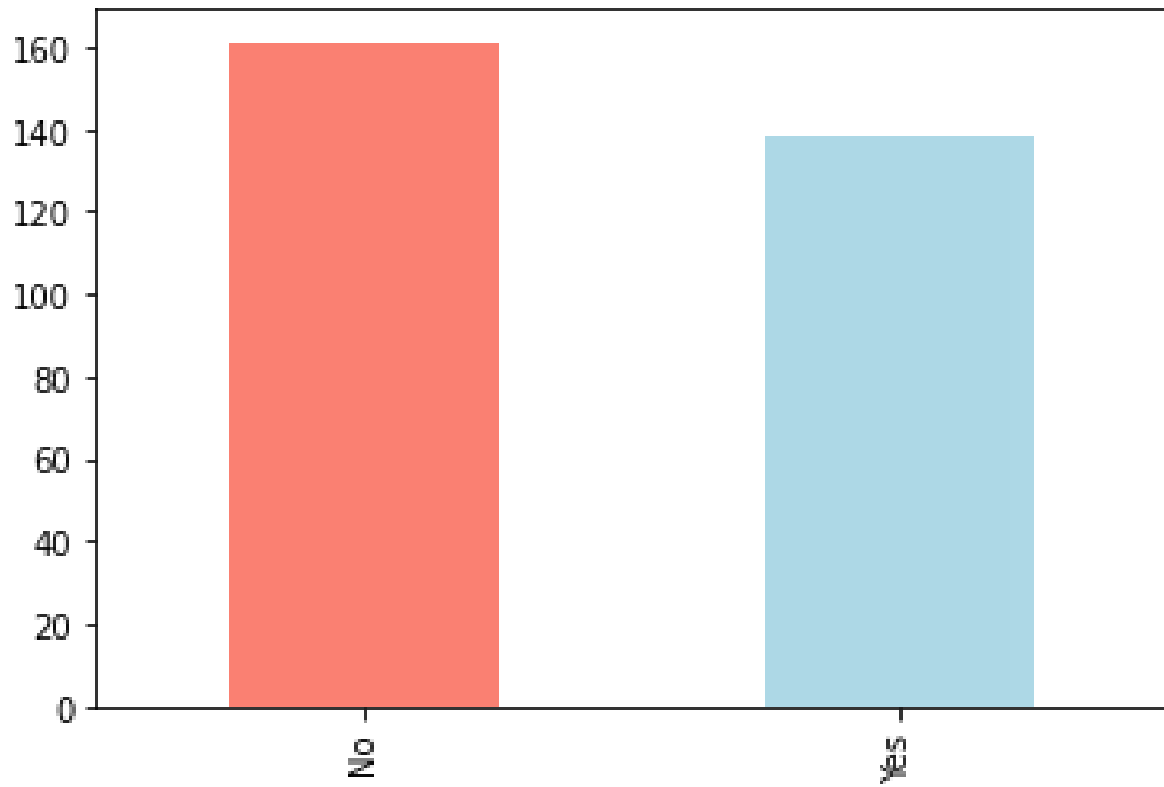
```
1. Counting the AHD value and plotting the graph for number of pateints having heart disease
```

```
In [21]: 1 # checking the distribution of AHD Variable
          2 data['AHD'].value_counts()
```

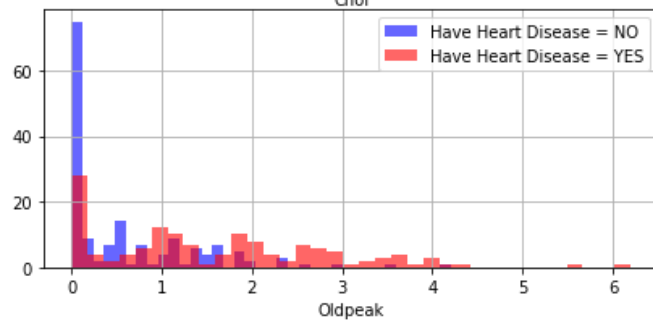
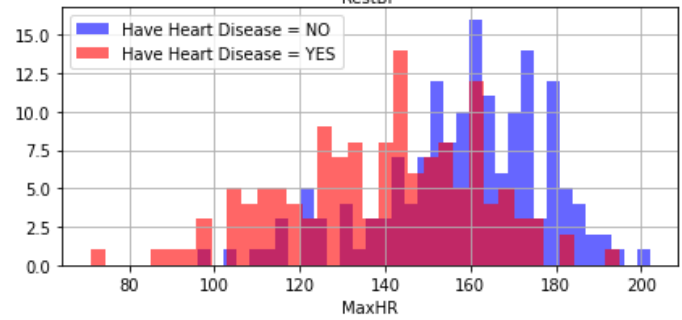
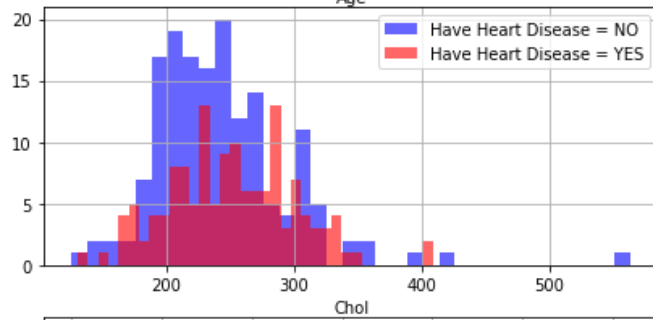
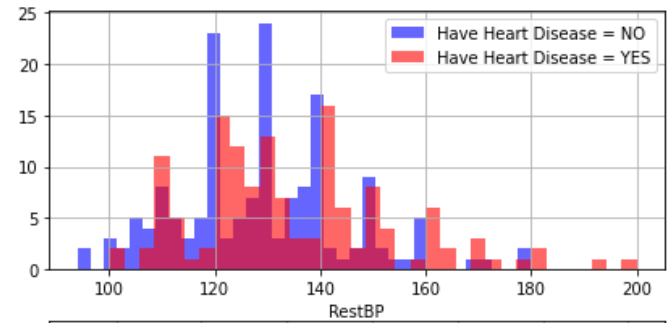
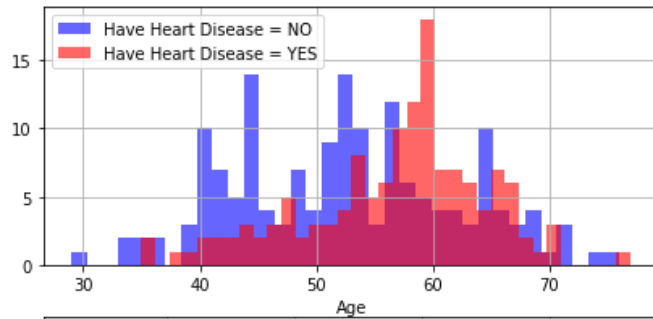
No	161
Yes	138

Name: AHD, dtype: int64

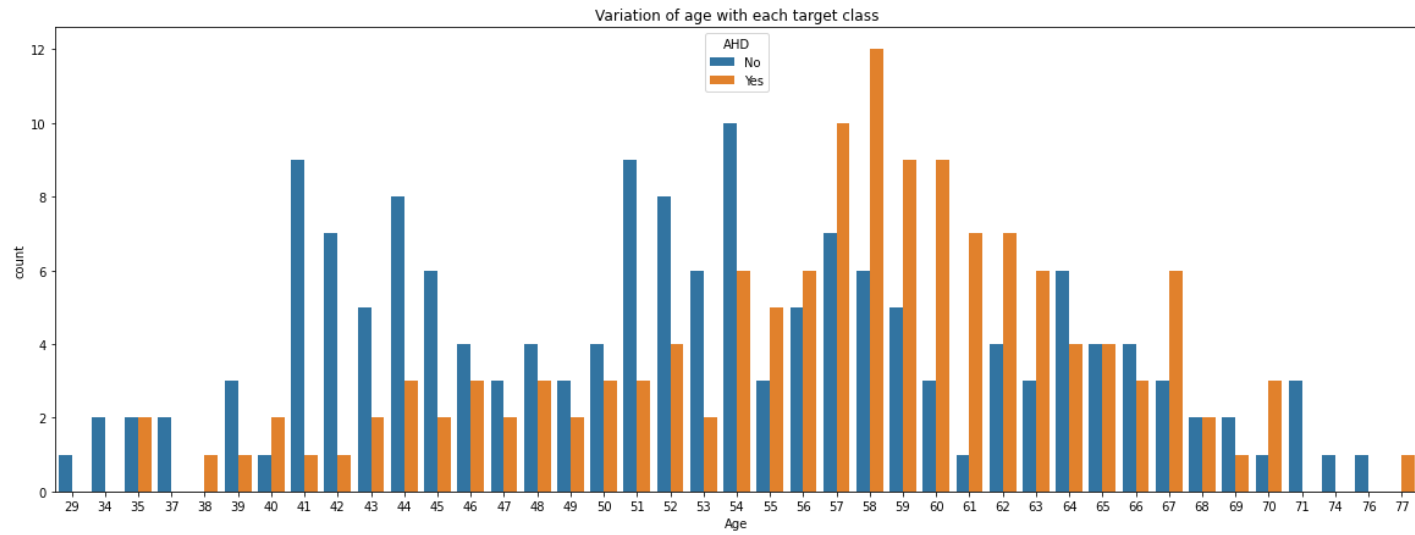
**Graph for number of patients having heart disease**



## Relation of various attributes with the target

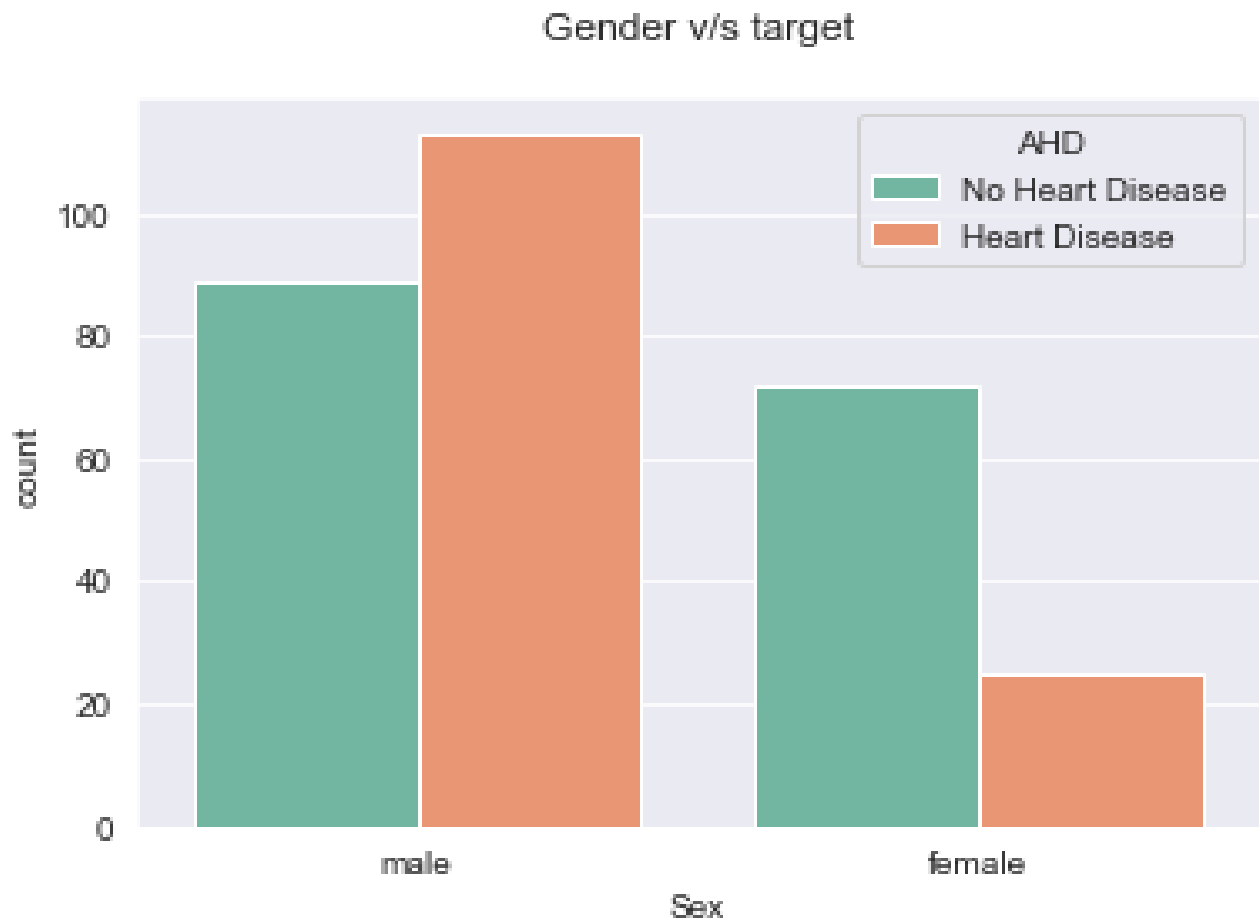


## Variation of age with each target class



**According to this dataset, Males are more susceptible to get Heart Disease than Females.**

**Proof:**



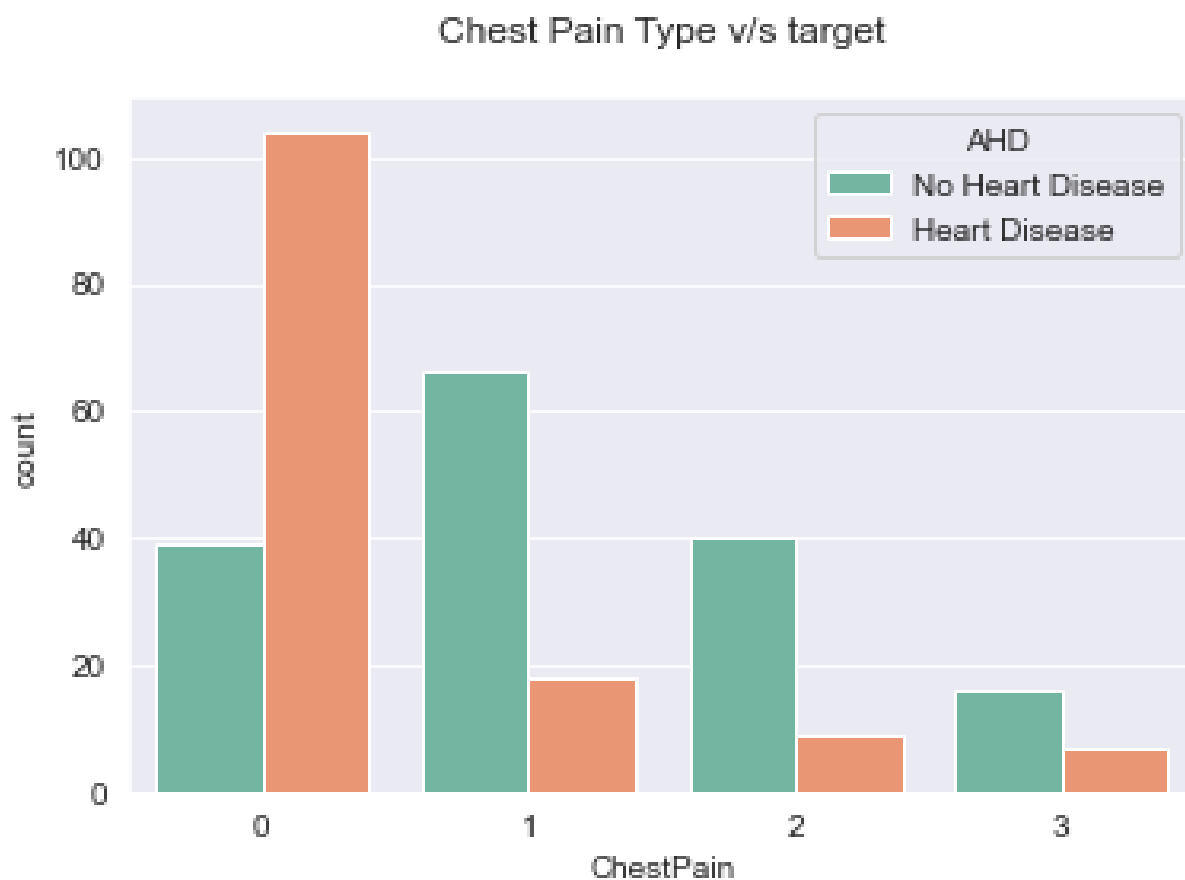


There are four types of chest pain, typical, non typical, asymptomatic and nonanginal.

Most of the Heart Disease patients are found to have asymptomatic chest pain

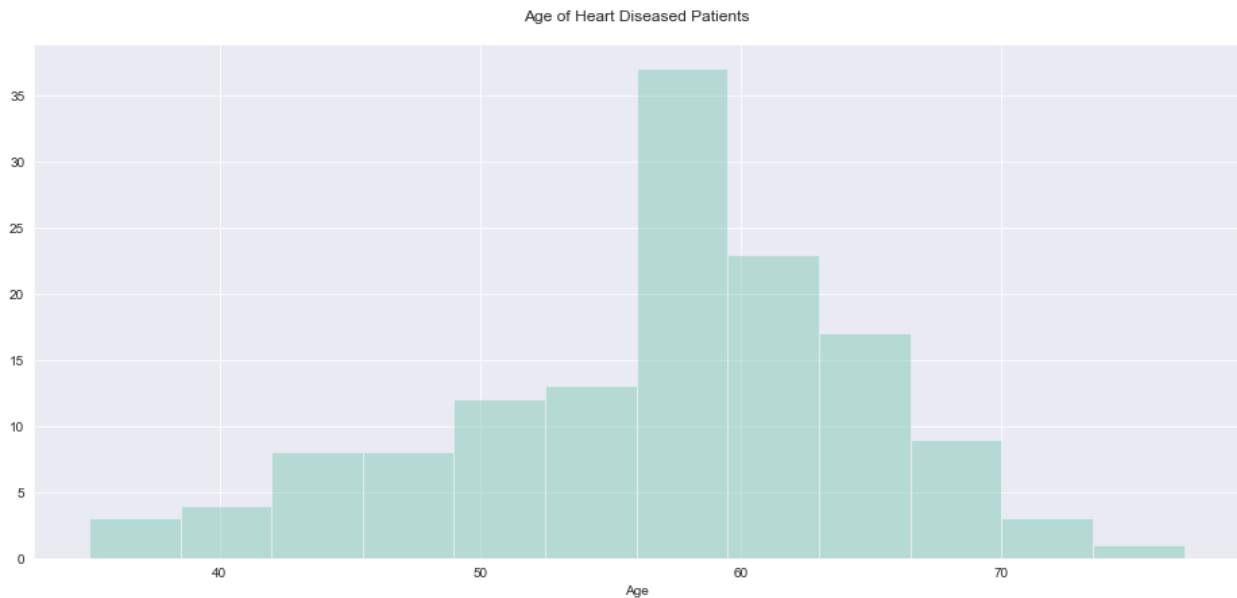
- ◆ Value 1: typical
- ◆ Value 2: non typical
- ◆ Value 3: asymptomatic
- ◆ Value 4: nonanginal

Proof:



**Heart Disease is very common in the seniors which is composed of age group 60 and above and common among adults which belong to the age group of 41 to 60. But it's rare among the age group of 19 to 40 and very rare among the age group of 0 to 18.**

**Proof:**



## 5. Prepare Data for Modeling

```
In [32]: 1 features = data.drop(columns='AHD', axis=1)
          2 target = data['AHD']
```

## 6. Splitting the data

```
In [35]: 1 from sklearn.model_selection import train_test_split
          2 X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.25, random_state=45)

In [36]: 1 print(f'features.shape: {features.shape}, X_train.shape: {X_train.shape}, X_test.shape: {X_test.shape}')
```

features.shape: (299, 13), X\_train.shape: (224, 13), X\_test.shape: (75, 13)

## 7. Fitting into Model

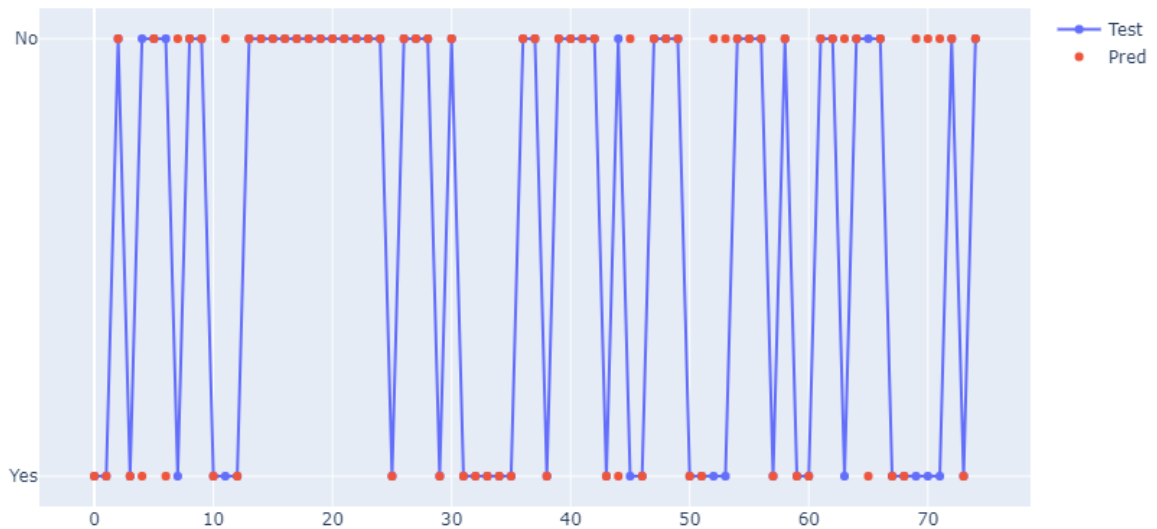
```
In [37]: 1 from sklearn.linear_model import LogisticRegression
          2 model = LogisticRegression()

In [38]: 1 # training the LogisticRegression model with Training data
          2 model.fit(X_train, y_train)
```

LogisticRegression()

## 8. Prediction and plotting 'predicted and actual' graph

```
In [39]: 1 y_pred = model.predict(X_test)
2 actual = []
3 predction = []
4 for i, j in zip(y_test, y_pred):
5     actual.append(i)
6     predction.append(j)
7 dic = {
8     'Actual': actual,
9     'Prediction': predction
10 }
11 result = pd.DataFrame(dic)
12 #####
13 import plotly.graph_objects as go
14 fig = go.Figure()
15 fig.add_trace(go.Scatter(x=np.arange(0,len(y_test)), y=y_test,
16                         mode='markers+lines',
17                         name='Test'))
18 fig.add_trace(go.Scatter(x=np.arange(0,len(y_test)), y=y_pred,
19                         mode='markers',
20                         name='Pred'))
```



- The red dots represent the predicted values that are either 0 or 1 and the blue line & dot represents the actual value of that particular patient.
- In the places where the red dot and blue dot do not overlap are the wrong predictions and where the both dots overlap those are the right predicted values.

## 9. Model Evaluation

```
In [40]: 1 from sklearn.metrics import accuracy_score

In [41]: 1 # accuracy on training data
          2 X_train_prediction = model.predict(X_train)
          3 training_data_accuracy = accuracy_score(X_train_prediction, y_train)

In [42]: 1 print('Accuracy on Training data : ', training_data_accuracy)

Accuracy on Training data : 0.8705357142857143
```

**Accuracy on Training data : 0.8705357142857143**

```
In [43]: 1 # accuracy on test data
          2 X_test_prediction = model.predict(X_test)
          3 test_data_accuracy = accuracy_score(X_test_prediction, y_test)

In [44]: 1 print('Accuracy on Testing data : ', test_data_accuracy)

Accuracy on Testing data : 0.8266666666666667
```

**Accuracy on Testing data : 0.8266666666666667**

```
In [45]: 1 test_score = accuracy_score(y_test, model.predict(X_test)) * 100
          2 train_score = accuracy_score(y_train, model.predict(X_train)) * 100
          3
          4 results_df = pd.DataFrame(data=[["Logistic Regression", train_score, test_score]],
          5                             columns=['Model', 'Training Accuracy %', 'Testing Accuracy %'])
          6 results_df
```

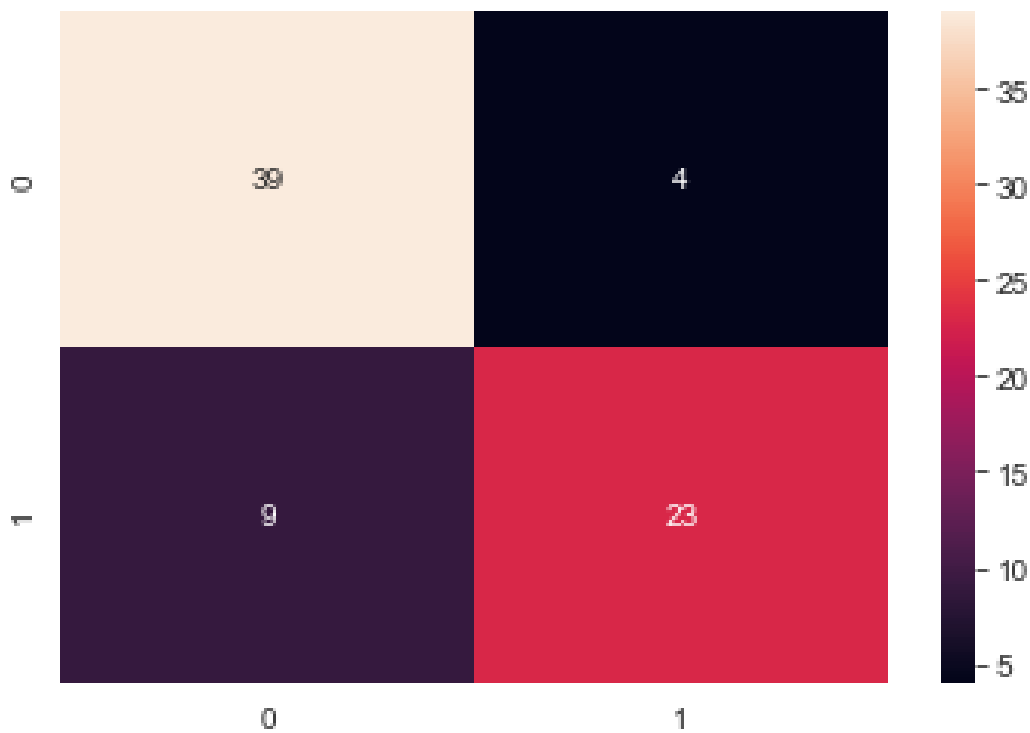
	Model	Training Accuracy %	Testing Accuracy %
0	Logistic Regression	87.053571	82.666667

## Classification report:

```
In [46]: 1 from sklearn.metrics import classification_report
          2 print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
No	0.81	0.91	0.86	43
Yes	0.85	0.72	0.78	32
accuracy			0.83	75
macro avg	0.83	0.81	0.82	75
weighted avg	0.83	0.83	0.82	75

## Confusion Matrix:



# **Conclusion**

Heart Disease is one of the major concerns for society today.

It is difficult to manually determine the odds of getting heart disease based on risk factors. However, machine learning techniques are useful to predict the output from existing data.

Thus, we have successfully completed our project by choosing Logistic Regression.



# References

- <https://towardsdatascience.com/heart-disease-prediction-73468d630cfc>
- <https://towardsdatascience.com/heart-disease-uci-diagnosis-prediction-b1943ee835a7>