```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

df=pd.read_csv('/content/drive/MyDrive/Heart.csv')

df.head()
```

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

```
#Find Shape of Data
df.shape
```

```
(303, 15)
```

```
#Find data type of each column
df.dtypes
```

```
Unnamed: 0      int64
Age             int64
Sex             int64
ChestPain      object
RestBP          int64
Chol            int64
Fbs             int64
RestECG         int64
MaxHR           int64
ExAng           int64
Oldpeak       float64
Slope           int64
Ca            float64
Thal           object
AHD            object
dtype: object
```

```
#Find Mean age of patients
mean1 = df['Age'].mean()
print ('Mean Age: ' + str(mean1))
```

```
Mean Age: 54.43894389438944
```

```
#Finding out Zero's
(df==0).sum()
```

```
Unnamed: 0      0
Age             0
Sex            97
```

```
ChestPain       0
RestBP          0
Chol            0
Fbs           258
RestECG       151
MaxHR           0
ExAng         204
Oldpeak        99
Slope           0
Ca            176
Thal            0
AHD             0
dtype: int64
```

```
#Find Missing Values
df.isna().sum()
```

```
Unnamed: 0    0
Age           0
Sex           0
ChestPain     0
RestBP        0
Chol          0
Fbs           0
RestECG       0
MaxHR         0
ExAng         0
Oldpeak       0
Slope         0
Ca            4
Thal          2
AHD           0
dtype: int64
```

```
from sklearn.model_selection import train_test_split

train,test=train_test_split(df[["Age","Sex","ChestPain","RestBP","Chol"]], test_size=0.2)
print("Training Data:",train)
print("Testing Data:",test)
```

```
Training Data:      Age  Sex     ChestPain  RestBP  Chol
83    68    1    nonanginal     180   274
108   61    1  asymptomatic     120   260
186   42    1    nonanginal     120   240
273   71    0  asymptomatic     112   149
141   59    1       typical     170   288
..   ...  ...           ...     ...   ...
272   46    1  asymptomatic     140   311
239   42    1    nontypical     120   295
68    59    1  asymptomatic     170   326
283   35    1    nontypical     122   192
152   67    0    nonanginal     115   564

[242 rows x 5 columns]
Testing Data:      Age  Sex     ChestPain  RestBP  Chol
116   58    1    nonanginal     140   211
125   45    0    nontypical     130   234
190   50    1    nonanginal     129   196
291   55    0    nontypical     132   342
211   38    1       typical     120   231
..   ...  ...           ...     ...   ...
212   41    1    nonanginal     130   214
```

```
131   51    1    nonanginal      94   227
37    57    1  asymptomatic     150   276
134   43    0    nonanginal     122   213
285   58    1  asymptomatic     114   318

[61 rows x 5 columns]
```

```
df[(df['AHD'] == 'Yes')].min()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuis
  """Entry point for launching an IPython kernel.
Unnamed: 0              2
Age                    35
Sex                     0
ChestPain     asymptomatic
RestBP                100
Chol                  131
Fbs                     0
RestECG                 0
MaxHR                  71
ExAng                   0
Oldpeak               0.0
Slope                   1
Ca                    0.0
AHD                   Yes
dtype: object
```

```
df[(df['AHD'] == 'Yes')].max()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuis
  """Entry point for launching an IPython kernel.
Unnamed: 0        302
Age                77
Sex                 1
ChestPain     typical
RestBP            200
Chol              409
Fbs                 1
RestECG             2
MaxHR             195
ExAng               1
Oldpeak           6.2
Slope               3
Ca                3.0
AHD               Yes
dtype: object
```

```
from sklearn import metrics
from sklearn.metrics import confusion_matrix


y_true = [2, 0, 2, 2, 0, 1]
y_pred = [0, 0, 2, 2, 0, 2]
confusion_matrix(y_true, y_pred)
print(metrics.classification_report(y_true, y_pred,labels=[0,1,2]))
```

```
                precision    recall  f1-score   support

           0       0.67      1.00      0.80         2
           1       0.00      0.00      0.00         1
           2       0.67      0.67      0.67         3

    accuracy                           0.67         6
   macro avg       0.44      0.56      0.49         6
weighted avg       0.56      0.67      0.60         6

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetric
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetric
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: UndefinedMetric
  _warn_prf(average, modifier, msg_start, len(result))
```

```
from sklearn.metrics import *

predicted_positive=[0]*400 +[1]*100;
actual_pos=[1]*50+[0]*450;
print("Confusion Matrix\n",confusion_matrix(actual_pos, predicted_positive))
print("Accuracy Score \n",accuracy_score(actual_pos,predicted_positive))
print("Precision Score \n",precision_score(actual_pos,predicted_positive))
print("Recall Score \n",recall_score(actual_pos, predicted_positive,average=None))
print("f1 Score \n",f1_score(actual_pos, predicted_positive, average=None))
```

```
Confusion Matrix
 [[350 100]
 [ 50   0]]
Accuracy Score
 0.7
Precision Score
 0.0
Recall Score
 [0.77777778 0.        ]
f1 Score
 [0.82352941 0.        ]
```