In [1]:
```
import os
os.getcwd()
```

Out[1]:
```
'C:\\Users\\omnic\\ML Practice\\LAB1'
```

In [2]:
```
import pandas as pd
```

In [3]:
```
# import dataset
df = pd.read_csv('Heart.csv')
```

In [4]:
```
df.head()
```

Out[4]:

	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

In [5]:
```
# shape of dataset ie no of row , columns
df.shape
```

Out[5]:
```
(303, 15)
```

In [6]:
```
# missing values in table
df.isnull()
```

Out[6]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slop
0	False	False	False	False	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	...	...	
298	False	False	False	False	False	False	False	False	False	False	False	Fals
299	False	False	False	False	False	False	False	False	False	False	False	Fals
300	False	False	False	False	False	False	False	False	False	False	False	Fals
301	False	False	False	False	False	False	False	False	False	False	False	Fals
302	False	False	False	False	False	False	False	False	False	False	False	Fals

303 rows × 15 columns

In [7]:
```
# missing values in summary shows null values
df.isnull().sum()
```

Out[7]:
```
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
```

In [8]:
```
# shows not null values
df.count()
```

Out[8]:
```
Unnamed: 0    303
Age           303
Sex           303
ChestPain     303
RestBP        303
Chol          303
Fbs           303
RestECG       303
MaxHR         303
ExAng         303
Oldpeak       303
Slope         303
Ca            299
Thal          301
AHD           303
dtype: int64
```

In [9]:
```
# data types of columns
df.dtypes
```

Out[9]:
```
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
```

In [10]:
```
# find zeros
df == 0
```

Out[10]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slop
0	False	False	False	False	False	False	False	False	False	True	False	Fals
1	False	False	False	False	False	False	True	False	False	False	False	Fals
2	False	False	False	False	False	False	True	False	False	False	False	Fals
3	False	False	False	False	False	False	True	True	False	True	False	Fals
4	False	False	True	False	False	False	True	False	False	True	False	Fals
...	...	...	...	...	...	...	...	...	...	...	...	
298	False	False	False	False	False	False	True	True	False	True	False	Fals
299	False	False	False	False	False	False	False	True	False	True	False	Fals
300	False	False	False	False	False	False	True	True	False	False	False	Fals
301	False	False	True	False	False	False	True	False	False	True	True	Fals
302	False	False	False	False	False	False	True	True	False	True	True	Fals

303 rows × 15 columns

In [11]:
```
# highlight zeros
df[df == 0]
```

Out[11]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.0	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN	0.0	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	0.0	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	0.0	0.0	NaN	0.0	NaN	NaN
4	NaN	NaN	0.0	NaN	NaN	NaN	0.0	NaN	NaN	0.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
298	NaN	NaN	NaN	NaN	NaN	NaN	0.0	0.0	NaN	0.0	NaN	NaN
299	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.0	NaN	0.0	NaN	NaN
300	NaN	NaN	NaN	NaN	NaN	NaN	0.0	0.0	NaN	NaN	NaN	NaN
301	NaN	NaN	0.0	NaN	NaN	NaN	0.0	NaN	NaN	0.0	0.0	NaN
302	NaN	NaN	NaN	NaN	NaN	NaN	0.0	0.0	NaN	0.0	0.0	NaN

303 rows × 15 columns

In [12]:
```
# count of zeros
df[df == 0].count()
```

Out[12]:
```
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
```

In [13]:
```
# column names only
df.columns
```

Out[13]:
```
Index(['Unnamed: 0', 'Age', 'Sex', 'ChestPain', 'RestBP', 'Chol', 'Fbs',
       'RestECG', 'MaxHR', 'ExAng', 'Oldpeak', 'Slope', 'Ca', 'Thal', 'AHD'],
      dtype='object')
```

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 15 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  303 non-null    int64  
 1   Age         303 non-null    int64  
 2   Sex         303 non-null    int64  
 3   ChestPain   303 non-null    object 
 4   RestBP      303 non-null    int64  
 5   Chol        303 non-null    int64  
 6   Fbs         303 non-null    int64  
 7   RestECG     303 non-null    int64  
 8   MaxHR       303 non-null    int64  
 9   ExAng       303 non-null    int64  
 10  Oldpeak     303 non-null    float64
 11  Slope       303 non-null    int64  
 12  Ca          299 non-null    float64
 13  Thal        301 non-null    object 
 14  AHD         303 non-null    object 
dtypes: float64(2), int64(10), object(3)
memory usage: 35.6+ KB
```

In [15]:
```
df.describe()
```

Out[15]:

	Unnamed: 0	Age	Sex	RestBP	Chol	Fbs	RestECG	MaxHR
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	152.000000	54.438944	0.679868	131.689769	246.693069	0.148515	0.990099	149.607261
std	87.612784	9.038662	0.467299	17.599748	51.776918	0.356198	0.994971	22.875003
min	1.000000	29.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000
25%	76.500000	48.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000
50%	152.000000	56.000000	1.000000	130.000000	241.000000	0.000000	1.000000	153.000000
75%	227.500000	61.000000	1.000000	140.000000	275.000000	0.000000	2.000000	166.000000
max	303.000000	77.000000	1.000000	200.000000	564.000000	1.000000	2.000000	202.000000

In [17]:
```
df.Age.mean()
```

Out[17]:
```
54.43894389438944
```

In [19]:
```
newdf = df[['Age', 'Sex', 'ChestPain', 'RestBP', 'Chol']]
```

In [20]:
```
newdf
```

Out[20]:

	Age	Sex	ChestPain	RestBP	Chol
0	63	1	typical	145	233
1	67	1	asymptomatic	160	286
2	67	1	asymptomatic	120	229
3	37	1	nonanginal	130	250
4	41	0	nontypical	130	204
...	...	...	...	...	...
298	45	1	typical	110	264
299	68	1	asymptomatic	144	193
300	57	1	asymptomatic	130	131
301	57	0	nontypical	130	236
302	38	1	nonanginal	138	175

303 rows × 5 columns

In [21]:
```
# cross validation
from sklearn.model_selection import train_test_split
```

In [22]:
```
train, test = train_test_split(df, random_state = 0, test_size = 0.25)
```

In [23]:

```
train.shape
```

Out[23]:

```
(227, 15)
```

In [24]:

```
test.shape
```

Out[24]:

```
(76, 15)
```

In [25]:

```
import numpy as np
```

In [26]:

```
actual = list(np.ones(45)) + list(np.zeros(450)) + list(np.ones(5))
np.array(actual)
```

Out[26]:

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 1., 1., 1., 1., 1.])
```

In [27]:

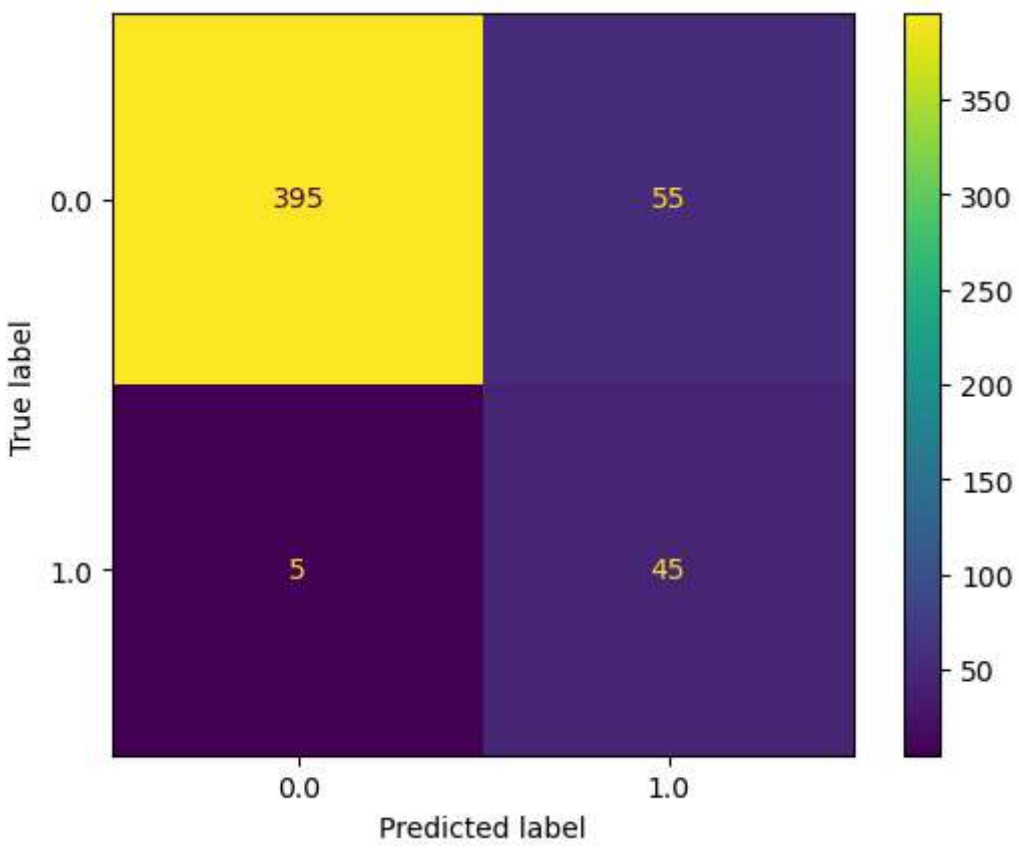
```
predicted = list(np.ones(100)) + list(np.zeros(400))
np.array(predicted)
```

Out[27]:
```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0.])
```

In [28]:
```python
from sklearn.metrics import ConfusionMatrixDisplay
```

In [29]:
```python
ConfusionMatrixDisplay.from_predictions(actual, predicted)
```

Out[29]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x22b452b06d0>
```



In [30]:
```
from sklearn.metrics import classification_report
```

In [32]:
```
print(classification_report(actual, predicted))
```

```
              precision    recall  f1-score   support

         0.0       0.99      0.88      0.93       450
         1.0       0.45      0.90      0.60        50

    accuracy                           0.88       500
   macro avg       0.72      0.89      0.76       500
weighted avg       0.93      0.88      0.90       500
```

In [33]:
```
from sklearn.metrics import accuracy_score
accuracy_score(actual, predicted)
```

Out[33]:
```
0.88
```

In [ ]: