Importing the Dependencies

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
from sklearn.metrics import accuracy_score
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files heart.csv

- **heart.csv**(application/vnd.ms-excel) - 11328 bytes, last modified: 2/21/2022 - 100% done

```
Saving heart.csv to heart.csv
```

```
df=pd.read_csv('heart.csv')
```

```
#To print first five  rows of the data
```

```
df.head(5)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	th
0	63	1	3	145	233	1	0	150	0	2.3	0	0	
1	37	1	2	130	250	0	1	187	0	3.5	0	0	
2	41	0	1	130	204	0	0	172	0	1.4	2	0	
3	56	1	1	120	236	0	1	178	0	0.8	2	0	
4	57	0	0	120	354	0	1	163	1	0.6	2	0	

```
#To print last five rows of the data
```

```
df.tail()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca
298	57	0	0	140	241	0	1	123	1	0.2	1	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1

```
# number of rows and columns in the dataset
```

```
df.shape
```

```
(303, 14)
```

```
# Details about the dataset
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

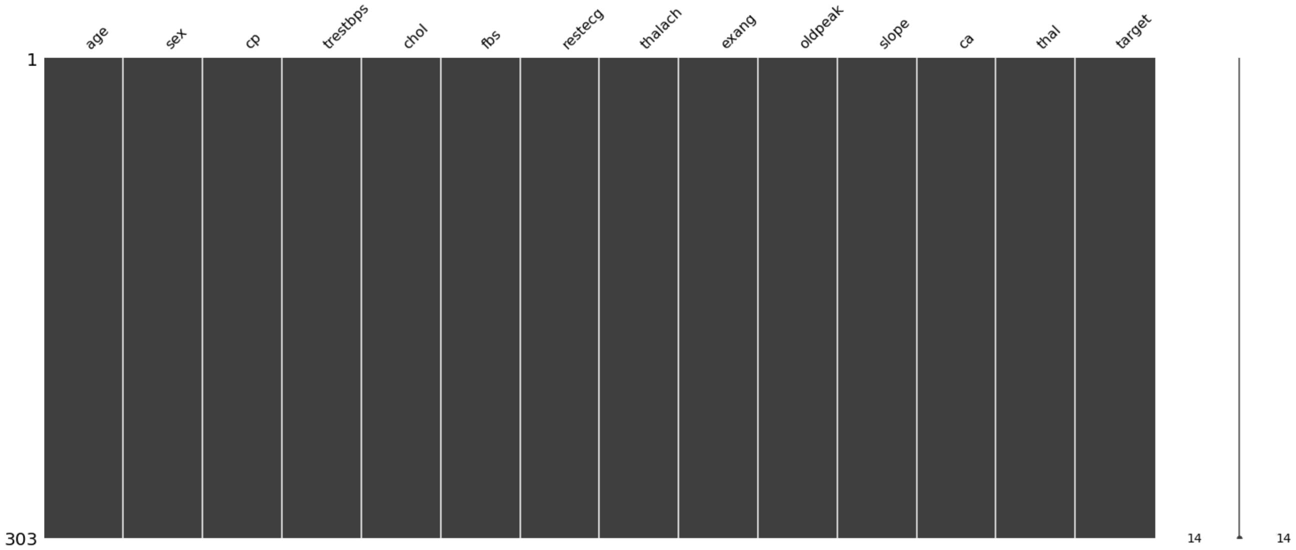
```
# Cross checking the missing values
```

```
df.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
```

```
     thal        0
     target      0
     dtype: int64
```

```
import missingno as msno
msno.matrix(df)
plt.show()
```



```
# statistical measures about the data
```

```
df.describe()
```

	age	sex	cp	trestbps	chol	fbs	reste
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.5280

```
#Distribution of Target Variable
```

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

```
1    165
0    138
Name: target, dtype: int64
```

```
#Splitting the data into Features and Labels
```

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

```
print(X)
```

```
     age  sex  cp  trestbps  chol  ...  exang  oldpeak  slope  ca  thal
0     63    1   3       145   233  ...      0      2.3      0   0     1
1     37    1   2       130   250  ...      0      3.5      0   0     2
2     41    0   1       130   204  ...      0      1.4      2   0     2
3     56    1   1       120   236  ...      0      0.8      2   0     2
4     57    0   0       120   354  ...      1      0.6      2   0     2
..   ...  ...  ..       ...   ...  ...    ...      ...    ...  ..   ...
298   57    0   0       140   241  ...      1      0.2      1   0     3
299   45    1   3       110   264  ...      0      1.2      1   0     3
300   68    1   0       144   193  ...      0      3.4      1   2     3
301   57    1   0       130   131  ...      1      1.2      1   1     3
302   57    0   1       130   236  ...      0      0.0      1   1     2

[303 rows x 13 columns]
```

```
print(Y)
```

```
0      1
1      1
2      1
3      1
4      1
      ..
298    0
299    0
300    0
301    0
302    0
Name: target, Length: 303, dtype: int64
```

```
#Separating  the Data into Training data & Test Data
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, rando
```

```
print(X.shape, X_train.shape, X_test.shape)
```

```
(303, 13) (242, 13) (61, 13)
```

```
# Model Training
```

```
#Naive Bayes Classifier
```

```
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
```

```
classifier = GaussianNB()
```

```
classifier.fit(X_train, Y_train)
```

```
GaussianNB()
```

```
#Model Evaluation
```

```
#Accuracy Score
```

```
# accuracy on training data
```

```
X_train_prediction = classifier.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

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

```
Accuracy on Training data :  0.8471074380165289
```

```
# accuracy on training data
```

```
X_test_prediction = classifier.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

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

```
Accuracy on Test data :  0.819672131147541
```

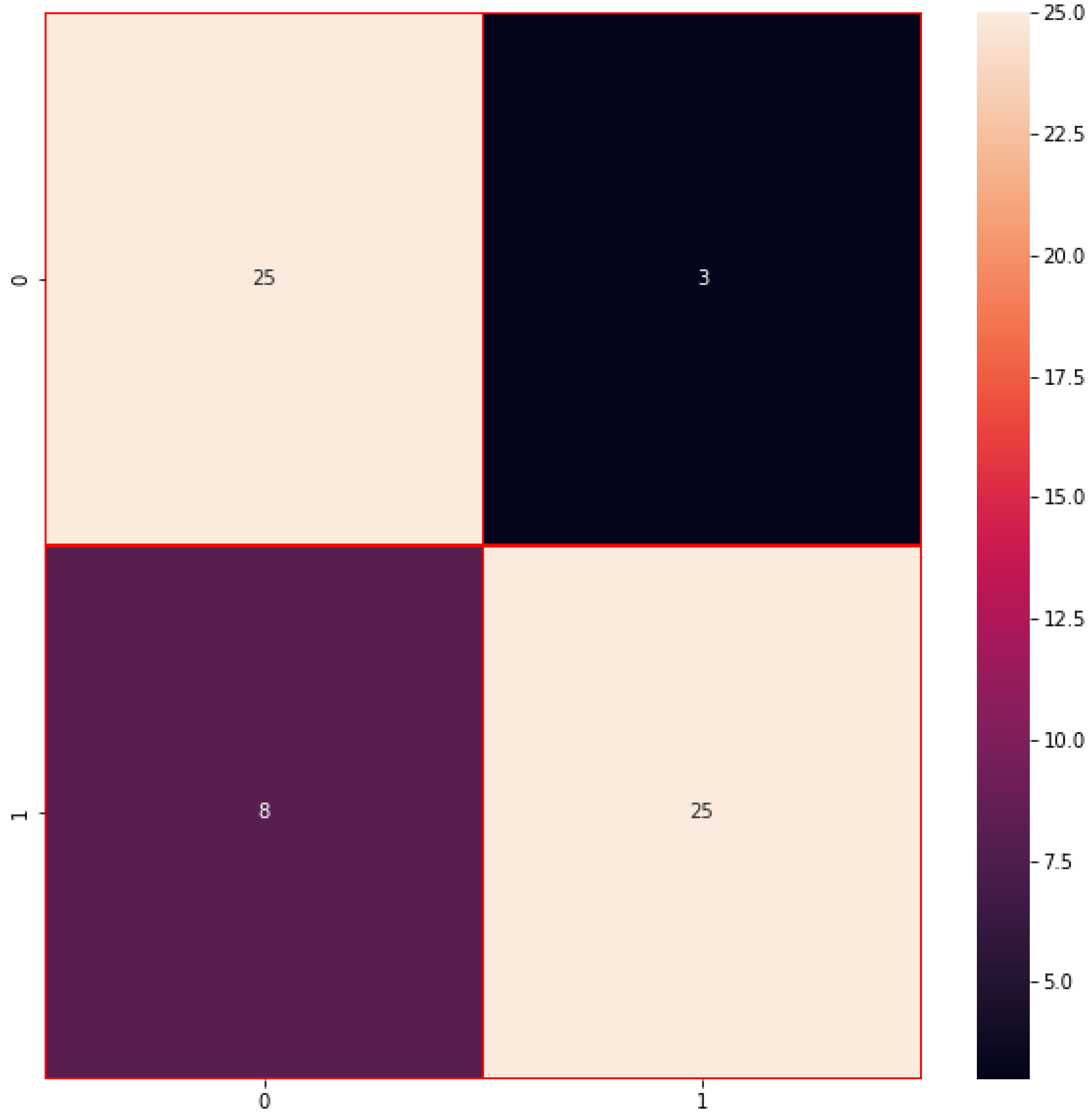
```
#Building a Predictive System


input_data = (56,1,2,130,256,1,0,142,1,0.6,1,1,1)


input_data = (43,1,0,115,303,0,1,181,0,1.2,1,0,2)


# 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 on instance


input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)


prediction = classifier.predict(input_data_reshaped)
print(prediction)
```

```
[0]
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not h
  "X does not have valid feature names, but"
```

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

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

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test, Y_pred)
f,ax = plt.subplots(figsize=(10, 10))
sns.heatmap(cm, annot=True, linewidths=0.5,linecolor="red", fmt= '.0f',ax=ax)
plt.show()
plt.savefig('ConfusionMatrix.png')
```



```
<Figure size 432x288 with 0 Axes>
```

✓ 0s completed at 8:14 PM