IMPORTING LIBRARIES

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

READING DATASET

In [2]:
```
df=pd.read_csv(r"D:\Projects\Project\HEART DISEASE PREDICTION\heart.csv")
```

In [3]:
```
df
```

Out[3]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

In [4]:
```
#getting the shape of the data
df.shape
```

Out[4]:
```
(303, 14)
```

In [6]:
```
#getting type of data for each column
df.dtypes
```

Out[6]:
```
age           int64
sex           int64
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

In [7]:
```
df.head()
```

Out[7]:

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

DIVIDING DATA INTO DEPENDENT AND INDEPENDENT VARIABLE

In [9]:
```
#independent variable
x=df.drop("target",axis=1)
```

In [10]:
```
x
```

Out[10]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2

303 rows × 13 columns

In [11]:
```
#dependent variable(target variable)
y=df["target"]
```

In [12]:
```
y
```

Out[12]:
```
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
```

TRAIN TEST MODULE

In [13]:
```
#creating training data and test data for x and y variable
from sklearn.model_selection import train_test_split
```

In [14]:
```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [15]:
```
xtrain
```

Out[15]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
74	43	0	2	122	213	0	1	165	0	0.2	1	0	2
153	66	0	2	146	278	0	0	152	0	0.0	1	1	2
64	58	1	2	140	211	1	0	165	0	0.0	2	0	2
296	63	0	0	124	197	0	1	136	1	0.0	1	0	2
287	57	1	1	154	232	0	0	164	0	0.0	2	1	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
251	43	1	0	132	247	1	0	143	1	0.1	1	4	3
192	54	1	0	120	188	0	1	113	0	1.4	1	1	3
117	56	1	3	120	193	0	0	162	0	1.9	1	0	3

47	47	1	2	138	257	0	0	156	0	0.0	2	0	2
172	58	1	1	120	284	0	0	160	0	1.8	1	0	2

242 rows × 13 columns

In [16]:
```
ytrain
```

Out[16]:
```
74     1
153    1
64     1
296    0
287    0
      ..
251    0
192    0
117    1
47     1
172    0
Name: target, Length: 242, dtype: int64
```

In [17]:
```
xtest
```

Out[17]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
225	70	1	0	145	174	0	1	125	1	2.6	0	0	3
152	64	1	3	170	227	0	0	155	0	0.6	1	0	3
228	59	1	3	170	288	0	0	159	0	0.2	1	0	3
201	60	1	0	125	258	0	0	141	1	2.8	1	1	3
52	62	1	2	130	231	0	1	146	0	1.8	1	3	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
146	44	0	2	118	242	0	1	149	0	0.3	1	1	2
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2
26	59	1	2	150	212	1	1	157	0	1.6	2	0	2
108	50	0	1	120	244	0	1	162	0	1.1	2	0	2
89	58	0	0	100	248	0	0	122	0	1.0	1	0	2

61 rows × 13 columns

In [18]:
```
ytest
```

Out[18]:
```
225    0
152    1
228    0
201    0
52     1
      ..
146    1
302    0
26     1
108    1
89     1
Name: target, Length: 61, dtype: int64
```

FEATURE SCALING

In [19]:
```
#standardize the independent features present in the data in a fixed range.
from sklearn.preprocessing import StandardScaler
```

In [20]:
```
sc=StandardScaler()
```

In [23]:
```
xtrain=sc.fit_transform(xtrain)
xtest=sc.fit_transform(xtest)
```

In [24]:
```
xtrain
```

Out[24]:
```
array([[-1.32773282, -1.43641607,  0.98584243, ..., -0.66169316,
```

```
         -0.70710678, -0.46472917],
        [ 1.24903178, -1.43641607,  0.98584243, ..., -0.66169316,
          0.26516504, -0.46472917],
        [ 0.35276583,  0.69617712,  0.98584243, ...,  0.95577901,
         -0.70710678, -0.46472917],
        ...,
        [ 0.12869935,  0.69617712,  1.94013791, ..., -0.66169316,
         -0.70710678,  1.14190596],
        [-0.87959984,  0.69617712,  0.98584243, ...,  0.95577901,
         -0.70710678, -0.46472917],
        [ 0.35276583,  0.69617712,  0.03154696, ..., -0.66169316,
         -0.70710678, -0.46472917]])
```

In [25]:
```
xtest
```

Out[25]:
```
array([[ 1.87528580e+00,  6.21581561e-01, -1.01006076e+00,
          7.38067738e-01, -1.57231830e+00, -4.16025147e-01,
          8.91132789e-01, -1.09707537e+00,  1.54560308e+00,
          1.62748286e+00, -2.26232796e+00, -7.45049451e-01,
          1.06436231e+00],
        [ 1.23443183e+00,  6.21581561e-01,  2.12283957e+00,
          2.04685065e+00, -3.86846038e-01, -4.16025147e-01,
         -1.12216722e+00,  2.78025949e-01, -6.46996639e-01,
         -3.40335861e-01, -5.99653193e-01, -7.45049451e-01,
          1.06436231e+00],
        [ 7.00386852e-01,  6.21581561e-01,  2.12283957e+00,
          2.04685065e+00,  9.77565437e-01, -4.16025147e-01,
         -1.12216722e+00,  4.61372791e-01, -6.46996639e-01,
         -7.33899606e-01, -5.99653193e-01, -7.45049451e-01,
          1.06436231e+00],
        [ 8.07195847e-01,  6.21581561e-01, -1.01006076e+00,
         -3.08958588e-01,  3.06543400e-01, -4.16025147e-01,
         -1.12216722e+00, -3.63687998e-01,  1.54560308e+00,
          1.82426474e+00, -5.99653193e-01,  2.64906471e-01,
          1.06436231e+00],
        [ 1.02081384e+00,  6.21581561e-01,  1.07853946e+00,
         -4.72020065e-02, -2.97376433e-01, -4.16025147e-01,
          8.91132789e-01, -1.34504446e-01, -6.46996639e-01,
          8.40355374e-01, -5.99653193e-01,  2.28481832e+00,
          1.06436231e+00],
        [-4.74512092e-01,  6.21581561e-01, -1.01006076e+00,
         -3.61309905e-01,  6.64421820e-01, -4.16025147e-01,
         -1.12216722e+00,  7.82229765e-01, -6.46996639e-01,
         -4.38726798e-01, -5.99653193e-01, -7.45049451e-01,
          1.06436231e+00],
        [-1.32898405e+00,  6.21581561e-01, -1.01006076e+00,
         -1.09422833e+00, -1.72889011e+00, -4.16025147e-01,
         -1.12216722e+00, -1.60127918e+00,  1.54560308e+00,
          1.03713725e+00, -5.99653193e-01, -7.45049451e-01,
          1.06436231e+00],
        [ 1.12762283e+00,  6.21581561e-01, -1.01006076e+00,
         -4.72020065e-02,  2.17073795e-01, -4.16025147e-01,
         -1.12216722e+00, -8.86677351e-02, -6.46996639e-01,
          4.46791629e-01, -5.99653193e-01,  2.64906471e-01,
          1.06436231e+00],
        [ 3.79959867e-01, -1.60879933e+00, -1.01006076e+00,
          3.61739014e+00,  9.77565437e-01,  2.40370085e+00,
         -1.12216722e+00, -7.30381683e-01,  1.54560308e+00,
          3.00495597e+00, -2.26232796e+00,  1.27486239e+00,
          1.06436231e+00],
        [ 1.12762283e+00,  6.21581561e-01, -1.01006076e+00,
         -4.72020065e-02,  1.91699629e+00,  2.40370085e+00,
         -1.12216722e+00, -7.76218393e-01,  1.54560308e+00,
          8.40355374e-01,  1.06302157e+00,  2.28481832e+00,
          1.06436231e+00],
        [ 4.86768862e-01,  6.21581561e-01,  1.07853946e+00,
          9.99824320e-01, -2.64595356e+00,  2.40370085e+00,
          8.91132789e-01,  1.10308674e+00, -6.46996639e-01,
         -7.33899606e-01,  1.06302157e+00,  2.64906471e-01,
          1.06436231e+00],
        [-3.67703097e-01, -1.60879933e+00, -1.01006076e+00,
         -4.72020065e-02,  5.52584814e-01, -4.16025147e-01,
          8.91132789e-01,  6.44719633e-01, -6.46996639e-01,
         -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
         -7.39140495e-01],
        [ 1.34124082e+00,  6.21581561e-01, -1.01006076e+00,
          2.14554575e-01,  2.17073795e-01, -4.16025147e-01,
         -1.12216722e+00, -1.00540195e+00, -6.46996639e-01,
          1.82426474e+00, -5.99653193e-01,  2.64906471e-01,
          1.06436231e+00],
        [-3.67703097e-01,  6.21581561e-01,  3.42393479e-02,
```

```
 -4.72020065e-02,  4.85482610e-01, -4.16025147e-01,
  8.91132789e-01,  1.01141332e+00, -6.46996639e-01,
 -3.40335861e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[-2.60894102e-01, -1.60879933e+00,  1.07853946e+00,
 -5.70715170e-01, -5.65785247e-01, -4.16025147e-01,
  8.91132789e-01,  4.15536081e-01, -6.46996639e-01,
  6.43573501e-01, -5.99653193e-01, -7.45049451e-01,
 -7.39140495e-01],
[-1.54085107e-01,  6.21581561e-01,  1.07853946e+00,
 -1.61774150e+00, -4.98683044e-01, -4.16025147e-01,
  8.91132789e-01, -2.72014577e-01,  1.54560308e+00,
  2.50009756e-01, -5.99653193e-01, -7.45049451e-01,
 -7.39140495e-01],
[-4.72761125e-02,  6.21581561e-01, -1.01006076e+00,
 -1.19893097e+00, -2.52641630e-01,  2.40370085e+00,
  8.91132789e-01, -8.86677351e-02, -6.46996639e-01,
 -8.32290543e-01,  1.06302157e+00,  2.28481832e+00,
  1.06436231e+00],
[-4.74512092e-01,  6.21581561e-01,  1.07853946e+00,
 -3.61309905e-01,  2.39441197e-01,  2.40370085e+00,
  8.91132789e-01,  1.19476016e+00, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00,  1.27486239e+00,
 -7.39140495e-01],
[ 1.02081384e+00,  6.21581561e-01, -1.01006076e+00,
 -5.70715170e-01,  5.07850011e-01, -4.16025147e-01,
  8.91132789e-01, -2.28882984e+00,  1.54560308e+00,
  8.40355374e-01, -5.99653193e-01,  1.27486239e+00,
  1.06436231e+00],
[-1.11536606e+00,  6.21581561e-01,  1.07853946e+00,
 -5.70715170e-01, -9.60698217e-02,  2.40370085e+00,
  8.91132789e-01,  2.06565766e+00, -6.46996639e-01,
 -1.43553989e-01, -2.26232796e+00, -7.45049451e-01,
  1.06436231e+00],
[-1.11536606e+00,  6.21581561e-01, -1.01006076e+00,
  4.76311157e-01, -4.09213439e-01, -4.16025147e-01,
  8.91132789e-01,  1.33227029e+00, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[-1.43579305e+00,  6.21581561e-01, -1.01006076e+00,
 -6.75417803e-01, -5.65785247e-01, -4.16025147e-01,
  8.91132789e-01, -4.09524709e-01, -6.46996639e-01,
  2.50009756e-01, -5.99653193e-01, -7.45049451e-01,
  1.06436231e+00],
[ 1.02081384e+00,  6.21581561e-01,  3.42393479e-02,
 -5.70715170e-01,  8.20993629e-01, -4.16025147e-01,
 -1.12216722e+00, -2.10548300e+00, -6.46996639e-01,
  4.46791629e-01, -5.99653193e-01,  2.64906471e-01,
  1.06436231e+00],
[ 1.66341877e-01,  6.21581561e-01, -1.01006076e+00,
 -1.09422833e+00, -8.56561463e-01, -4.16025147e-01,
 -1.12216722e+00, -1.87629945e+00,  1.54560308e+00,
 -9.30681479e-01, -5.99653193e-01,  2.64906471e-01,
 -7.39140495e-01],
[-1.22217506e+00,  6.21581561e-01,  3.42393479e-02,
 -1.09422833e+00, -2.07906828e-01, -4.16025147e-01,
  8.91132789e-01,  1.86352528e-01, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[ 1.02081384e+00, -1.60879933e+00,  1.07853946e+00,
 -4.72020065e-02,  4.18380407e-01, -4.16025147e-01,
  8.91132789e-01, -2.38050326e+00, -6.46996639e-01,
  2.50009756e-01, -5.99653193e-01,  2.64906471e-01,
  1.06436231e+00],
[-4.72761125e-02,  6.21581561e-01, -1.01006076e+00,
 -3.08958588e-01, -7.22357056e-01, -4.16025147e-01,
  8.91132789e-01,  8.73903186e-01, -6.46996639e-01,
  5.32278835e-02,  1.06302157e+00,  1.27486239e+00,
  1.06436231e+00],
[-6.88130082e-01,  6.21581561e-01, -1.01006076e+00,
 -5.70715170e-01,  1.05236789e-01, -4.16025147e-01,
 -1.12216722e+00, -2.26177867e-01, -6.46996639e-01,
 -1.43553989e-01,  1.06302157e+00, -7.45049451e-01,
  1.06436231e+00],
[-1.43579305e+00, -1.60879933e+00,  1.07853946e+00,
  3.71608524e-01, -5.43417846e-01, -4.16025147e-01,
  8.91132789e-01,  1.40515818e-01, -6.46996639e-01,
 -9.30681479e-01, -5.99653193e-01, -7.45049451e-01,
 -7.39140495e-01],
[-4.72761125e-02,  6.21581561e-01,  3.42393479e-02,
  1.62203259e-01, -9.68398469e-01, -4.16025147e-01,
  8.91132789e-01,  4.15536081e-01, -6.46996639e-01,
 -1.43553989e-01,  1.06302157e+00,  2.64906471e-01,
```

```
 -7.39140495e-01],
[ 8.07195847e-01,  6.21581561e-01,  1.07853946e+00,
  4.76311157e-01, -1.32627689e+00, -4.16025147e-01,
 -1.12216722e+00,  2.78025949e-01, -6.46996639e-01,
  2.02104661e+00, -5.99653193e-01, -7.45049451e-01,
 -7.39140495e-01],
[-7.94939077e-01,  6.21581561e-01, -1.01006076e+00,
  5.81013789e-01,  1.44728086e+00, -4.16025147e-01,
 -1.12216722e+00, -8.86677351e-02,  1.54560308e+00,
 -9.30681479e-01, -5.99653193e-01,  2.28481832e+00,
  1.06436231e+00],
[-1.22217506e+00, -1.60879933e+00,  3.42393479e-02,
 -2.56607272e-01,  1.38017866e+00, -4.16025147e-01,
  8.91132789e-01,  6.44719633e-01, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[-4.72761125e-02,  6.21581561e-01,  1.07853946e+00,
  2.15155328e+00, -1.01313327e+00,  2.40370085e+00,
  8.91132789e-01,  5.98882923e-01, -6.46996639e-01,
 -4.38726798e-01,  1.06302157e+00, -7.45049451e-01,
  1.06436231e+00],
[-1.86302903e+00,  6.21581561e-01,  3.42393479e-02,
 -4.66012537e-01, -1.16970508e+00, -4.16025147e-01,
  8.91132789e-01,  1.14892345e+00, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[ 7.00386852e-01, -1.60879933e+00, -1.01006076e+00,
  2.25625591e+00,  1.05236789e-01, -4.16025147e-01,
  8.91132789e-01, -2.72014577e-01,  1.54560308e+00,
 -9.30681479e-01, -5.99653193e-01, -7.45049451e-01,
 -7.39140495e-01],
[ 1.87528580e+00,  6.21581561e-01,  1.07853946e+00,
  1.52333748e+00,  5.52584814e-01, -4.16025147e-01,
  8.91132789e-01, -1.69295260e+00,  1.54560308e+00,
  1.92265567e+00, -5.99653193e-01,  2.64906471e-01,
  1.06436231e+00],
[-7.94939077e-01,  6.21581561e-01,  3.42393479e-02,
 -1.51904639e-01,  1.42491346e+00, -4.16025147e-01,
 -1.12216722e+00,  9.65576607e-01, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[ 3.79959867e-01,  6.21581561e-01, -1.01006076e+00,
 -3.08958588e-01,  1.05236789e-01,  2.40370085e+00,
 -1.12216722e+00, -2.26177867e-01,  1.54560308e+00,
  2.50009756e-01, -5.99653193e-01,  2.64906471e-01,
 -7.39140495e-01],
[ 1.02081384e+00, -1.60879933e+00, -1.01006076e+00,
  4.76311157e-01,  5.30217413e-01, -4.16025147e-01,
 -1.12216722e+00,  5.07209502e-01, -6.46996639e-01,
  2.61139223e+00, -2.26232796e+00,  1.27486239e+00,
 -7.39140495e-01],
[ 4.86768862e-01,  6.21581561e-01, -1.01006076e+00,
  4.76311157e-01, -1.16970508e+00, -4.16025147e-01,
  8.91132789e-01, -4.28310246e-02, -6.46996639e-01,
 -5.37117734e-01, -5.99653193e-01, -7.45049451e-01,
 -2.54264330e+00],
[ 4.86768862e-01, -1.60879933e+00, -1.01006076e+00,
 -1.51904639e-01,  1.31307646e+00, -4.16025147e-01,
 -1.12216722e+00,  4.61372791e-01, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00,  2.64906471e-01,
 -7.39140495e-01],
[-4.72761125e-02,  6.21581561e-01,  1.07853946e+00,
  3.71608524e-01, -4.76315642e-01, -4.16025147e-01,
  8.91132789e-01,  9.19739897e-01, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00,  3.29477424e+00,
 -7.39140495e-01],
[-5.81321087e-01,  6.21581561e-01, -1.01006076e+00,
 -1.09422833e+00,  6.86789221e-01, -4.16025147e-01,
 -1.12216722e+00, -1.41793234e+00,  1.54560308e+00,
  5.32278835e-02, -5.99653193e-01,  2.64906471e-01,
 -7.39140495e-01],
[-1.54085107e-01,  6.21581561e-01,  2.12283957e+00,
 -3.08958588e-01, -6.99989655e-01, -4.16025147e-01,
 -1.12216722e+00, -1.09707537e+00,  1.54560308e+00,
  4.46791629e-01,  1.06302157e+00,  2.64906471e-01,
 -7.39140495e-01],
[ 1.87528580e+00,  6.21581561e-01,  3.42393479e-02,
  1.31393222e+00,  1.57671845e-02, -4.16025147e-01,
 -1.12216722e+00, -2.72014577e-01, -6.46996639e-01,
 -9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 -7.39140495e-01],
[-1.22217506e+00,  6.21581561e-01,  1.07853946e+00,
 -4.72020065e-02, -6.77622253e-01, -4.16025147e-01,
```

```
-1.12216722e+00,  8.73903186e-01, -6.46996639e-01,
 1.03713725e+00, -5.99653193e-01, -7.45049451e-01,
-7.39140495e-01],
[-2.60894102e-01,  6.21581561e-01,  1.07853946e+00,
 4.76311157e-01, -2.52641630e-01, -4.16025147e-01,
 8.91132789e-01,  6.44719633e-01, -6.46996639e-01,
-3.40335861e-01, -5.99653193e-01,  2.64906471e-01,
 1.06436231e+00],
[ 3.79959867e-01, -1.60879933e+00, -1.01006076e+00,
 1.62203259e-01,  3.68402099e+00, -4.16025147e-01,
-1.12216722e+00,  4.88423965e-02,  1.54560308e+00,
 9.38746310e-01, -5.99653193e-01,  1.27486239e+00,
 1.06436231e+00],
[-9.01748072e-01,  6.21581561e-01,  3.42393479e-02,
-5.70715170e-01,  4.18380407e-01, -4.16025147e-01,
 8.91132789e-01,  1.10308674e+00, -6.46996639e-01,
-9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
 1.06436231e+00],
[-4.72761125e-02,  6.21581561e-01,  3.42393479e-02,
-5.70715170e-01,  1.80515928e+00, -4.16025147e-01,
 8.91132789e-01,  1.05725003e+00, -6.46996639e-01,
-7.33899606e-01,  1.06302157e+00, -7.45049451e-01,
-7.39140495e-01],
[ 2.30252177e+00, -1.60879933e+00,  3.42393479e-02,
-5.70715170e-01,  5.52584814e-01, -4.16025147e-01,
-1.12216722e+00, -1.28042221e+00,  1.54560308e+00,
-7.33899606e-01,  1.06302157e+00,  2.64906471e-01,
-7.39140495e-01],
[-1.22217506e+00, -1.60879933e+00,  1.07853946e+00,
-9.89525700e-01,  5.30217413e-01, -4.16025147e-01,
-1.12216722e+00,  1.05725003e+00,  1.54560308e+00,
-9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
-7.39140495e-01],
[-1.22217506e+00,  6.21581561e-01,  3.42393479e-02,
 2.14554575e-01, -9.23663667e-01, -4.16025147e-01,
 8.91132789e-01, -7.76218393e-01, -6.46996639e-01,
-9.30681479e-01, -5.99653193e-01, -7.45049451e-01,
-2.54264330e+00],
[-1.43579305e+00, -1.60879933e+00,  1.07853946e+00,
-1.93184939e+00, -1.01313327e+00, -4.16025147e-01,
 8.91132789e-01,  1.37810700e+00, -6.46996639e-01,
-9.30681479e-01,  1.06302157e+00, -7.45049451e-01,
-7.39140495e-01],
[-1.86302903e+00,  6.21581561e-01, -1.01006076e+00,
-5.70715170e-01, -1.03550067e+00, -4.16025147e-01,
 8.91132789e-01, -8.67891814e-01,  1.54560308e+00,
 6.43573501e-01, -5.99653193e-01, -7.45049451e-01,
 1.06436231e+00],
[-9.01748072e-01, -1.60879933e+00,  1.07853946e+00,
-6.75417803e-01, -5.13350192e-02, -4.16025147e-01,
 8.91132789e-01,  3.00568594e-03, -6.46996639e-01,
-6.35508670e-01, -5.99653193e-01,  2.64906471e-01,
-7.39140495e-01],
[ 4.86768862e-01, -1.60879933e+00,  3.42393479e-02,
-4.72020065e-02, -1.85539427e-01, -4.16025147e-01,
-1.12216722e+00,  1.14892345e+00, -6.46996639e-01,
-9.30681479e-01, -5.99653193e-01,  2.64906471e-01,
-7.39140495e-01],
[ 7.00386852e-01,  6.21581561e-01,  1.07853946e+00,
 9.99824320e-01, -7.22357056e-01,  2.40370085e+00,
 8.91132789e-01,  3.69699370e-01, -6.46996639e-01,
 6.43573501e-01,  1.06302157e+00, -7.45049451e-01,
-7.39140495e-01],
[-2.60894102e-01, -1.60879933e+00,  3.42393479e-02,
-5.70715170e-01, -6.60021675e-03, -4.16025147e-01,
 8.91132789e-01,  5.98882923e-01, -6.46996639e-01,
 1.51618820e-01,  1.06302157e+00, -7.45049451e-01,
-7.39140495e-01],
[ 5.93577857e-01, -1.60879933e+00, -1.01006076e+00,
-1.61774150e+00,  8.28693881e-02, -4.16025147e-01,
-1.12216722e+00, -1.23458550e+00, -6.46996639e-01,
 5.32278835e-02, -5.99653193e-01, -7.45049451e-01,
-7.39140495e-01]])
```

IMPLEMENTING KNN ALGORITHM

In [53]:
```
#impprting algorithm
from sklearn.neighbors import KNeighborsClassifier
#creating no of neighbors (k=5)
```

```
classifier = KNeighborsClassifier(n_neighbors=12)
#train the model
classifier.fit(xtrain, ytrain)
```

Out[53]:
```
KNeighborsClassifier(n_neighbors=12)
```

MAKING PREDICTIONS

In [54]:
```
y_pred=classifier.predict(xtest)
y_pred
```

Out[54]:
```
array([0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0,
       1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1], dtype=int64)
```

CREATING DATA FRAME

In [55]:
```
#creating data frame for comparing actual values with predicted values
data=pd.DataFrame({"Actual":ytest,"Predicted":y_pred})
```

In [56]:
```
data
```

Out[56]:

	Actual	Predicted
225	0	0
152	1	1
228	0	0
201	0	0
52	1	0
...	...	...
146	1	1
302	0	1
26	1	1
108	1	1
89	1	1

61 rows × 2 columns

MAKING OWN PREDICTIONS

In [57]:
```
#making predictions for our own input values
y_pred2=classifier.predict([[63,1,3,145,233,1,0,150,0,23,0,0,1]])
```

In [58]:
```
#getting target vvalue for above mentioned inputs
y_pred2
```

Out[58]:
```
array([1], dtype=int64)
```

EVALUATING THE ALGORITHM

In [59]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [60]:
```
#getting confusion matrix - since it is a classification problem
confusion_matrix(ytest,y_pred)
```

Out[60]:
```
array([[22,  5],
       [ 3, 31]], dtype=int64)
```

In [61]:
```
#getting accuracy of the model
accuracy_score(ytest,y_pred)
```

Out[61]: 0.8688524590163934

SELECTING K VALUE IN KNN

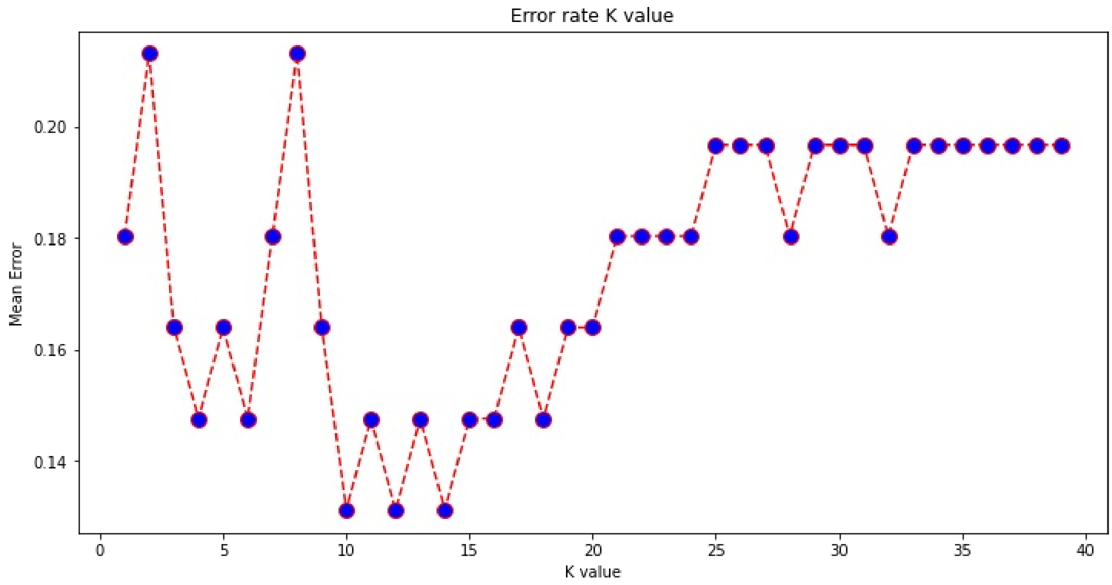
In [51]:
```
#calculating error rate with respect to the k value
error=[]
for i in range(1,40):
    knn=KNeighborsClassifier(n_neighbors=i)
    knn.fit(xtrain,ytrain)
    y_pred=knn.predict(xtest)
    error.append(np.mean(y_pred!=ytest))
print(error)
```

```
[0.18032786885245902, 0.21311475409836064, 0.16393442622950818, 0.14754098360655737, 0.16393442622950818, 0.14754
098360655737, 0.18032786885245902, 0.21311475409836064, 0.16393442622950818, 0.13114754098360656, 0.1475409836065
5737, 0.13114754098360656, 0.14754098360655737, 0.13114754098360656, 0.14754098360655737, 0.14754098360655737, 0.
16393442622950818, 0.14754098360655737, 0.16393442622950818, 0.16393442622950818, 0.18032786885245902, 0.18032786
885245902, 0.18032786885245902, 0.18032786885245902, 0.19672131147540983, 0.19672131147540983, 0.1967213114754098
3, 0.18032786885245902, 0.19672131147540983, 0.19672131147540983, 0.19672131147540983, 0.18032786885245902, 0.196
72131147540983, 0.19672131147540983, 0.19672131147540983, 0.19672131147540983, 0.19672131147540983, 0.19672131147
540983, 0.19672131147540983]
```

PLOTTING GRAPH BETWEEN K VALUE AND ERROR VALUE

In [52]:
```
plt.figure(figsize=(12,6))
plt.plot(range(1,40),error,color="red",linestyle="dashed",marker="o",markerfacecolor="blue",markersize=10)
plt.title("Error rate K value")
plt.xlabel("K value")
plt.ylabel("Mean Error")
```

Out[52]: Text(0, 0.5, 'Mean Error')



In [ ]:
```
#when the k value is 5 accuracy of the model is 83
#when the k values is 12 accuracy of the model is 86
#with the help of k value we can reduce the error and increase the accuracy.
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js