In [35]:
```
import warnings
import pandas as pd
import numpy as np
warnings.filterwarnings("ignore")
```

In [36]:
```
df1= pd.read_csv("heart.csv")
```

In [37]:
```
df1.head()
```

Out[37]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

In [38]:
```
print(len(df1))
```

```
1025
```

In [39]:
```
df1.describe
```

Out[39]:
```
<bound method NDFrame.describe of       age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak
\
0      52    1   0       125   212    0        1      168      0      1.0
1      53    1   0       140   203    1        0      155      1      3.1
2      70    1   0       145   174    0        1      125      1      2.6
3      61    1   0       148   203    0        1      161      0      0.0
4      62    0   0       138   294    1        1      106      0      1.9
...   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
1020   59    1   1       140   221    0        1      164      1      0.0
1021   60    1   0       125   258    0        0      141      1      2.8
1022   47    1   0       110   275    0        0      118      1      1.0
1023   50    0   0       110   254    0        0      159      0      0.0
1024   54    1   0       120   188    0        1      113      0      1.4

      slope  ca  thal  target
0         2   2     3       0
1         0   0     3       0
2         0   0     3       0
3         2   1     3       0
4         1   3     2       0
...     ...  ..   ...     ...
1020      2   0     2       1
1021      1   1     3       0
1022      1   1     2       0
1023      2   0     2       1
1024      1   1     3       0

[1025 rows x 14 columns]>
```

In [40]:
```
df1.isnull().sum()
```

Out[40]:
```
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
```

In [41]:
```
cat_val=[]
cont_val=[]
for column in df1.columns:
    if df1[column].nunique() <=10:
        cat_val.append(column)
    else:
        cont_val.append(column)
```

In [42]:
```
cat_val
```

Out[42]:
```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

In [43]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
df1[cont_val]=sc.fit_transform(df1[cont_val])
```

In [44]:
```
df1.head()
```

Out[44]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	-0.268437	1	0	-0.377636	-0.659332	0	1	0.821321	0	-0.060888	2	2	3	0
1	-0.158157	1	0	0.479107	-0.833861	1	0	0.255968	1	1.727137	0	0	3	0
2	1.716595	1	0	0.764688	-1.396233	0	1	-1.048692	1	1.301417	0	0	3	0
3	0.724079	1	0	0.936037	-0.833861	0	1	0.516900	0	-0.912329	2	1	3	0
4	0.834359	0	0	0.364875	0.930822	1	1	-1.874977	0	0.705408	1	3	2	0

In [64]:
```
X = df1.drop(columns=[ 'target'])
y = df1['target']
```

In [65]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 1)
```

In [66]:
```
## Roandfom forest tree
```

In [67]:
```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier( random_state = 0)
classifier.fit(X_train, y_train)
```

Out[67]:
```
RandomForestClassifier(random_state=0)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [68]:
```
y_pred = classifier.predict(X_test)
```

In [69]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[109   0]
 [  0  96]]
```

Out[69]:
```
1.0
```

In [70]:
```
## Decision tree
```

In [71]:
```
from sklearn.tree import DecisionTreeClassifier
classifier_1 = DecisionTreeClassifier()
classifier_1.fit(X_train, y_train)
```

Out[71]:
```
DecisionTreeClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [72]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
y_pred2 = classifier_1.predict(X_test)
cm = confusion_matrix(y_test, y_pred2)
print(cm)

accuracy_score(y_test, y_pred2)
```

```
[[109   0]
 [  0  96]]
```

Out[72]:
```
1.0
```

In [73]:
```
##
```

In [74]:
```
from sklearn.linear_model import LogisticRegression
classifier2 = LogisticRegression(random_state = 0)
classifier2.fit(X_train, y_train)
```

Out[74]:
```
LogisticRegression(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [75]:
```
y_pred3= classifier2.predict(X_test)
```

In [76]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred3)
print(cm)
accuracy_score(y_test, y_pred3)
```

```
[[80 29]
 [10 86]]
```

Out[76]:
```
0.8097560975609757
```

In [77]:
```
## KNN
```

In [78]:
```
from sklearn.neighbors import KNeighborsClassifier
classifier3 = KNeighborsClassifier(n_neighbors = 5)
classifier3.fit(X_train, y_train)
```

Out[78]:
```
KNeighborsClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [79]:
```
y_pred_4 = classifier3.predict(X_test)
```

In [83]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred_4)
print(cm)
accuracy_score(y_test, y_pred_4)
```

```
[[87 22]
 [14 82]]
```

Out[83]:
```
0.824390243902439
```

In [ ]:
```
```

In [101]:
```
import pickle
```

In [102]:
```
pickle.dump(classifier, open("model1.pkl", "wb"))
```

In [ ]:
```
```