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

In [11]:
```
var=pd.read_csv('C://Users/Gopi/Desktop/heart.csv')
print(var.shape)
```

```
(303, 14)
```

In [12]:
```
var.isnull().mean().head()
```

Out[12]:
```
age         0.0
sex         0.0
cp          0.0
trestbps    0.0
chol        0.0
dtype: float64
```

In [13]:
```
var.isnull().values.any()
```

Out[13]:
```
False
```

In [16]:
```
y = var['target']
X = var.drop(['target'], axis = 1)
```

In [21]:
```
from sklearn.model_selection import train_test_split
X_train ,X_test ,y_train ,y_test =train_test_split(X,y,test_size=0.30)

print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(212, 13)
(91, 13)
(212,)
(91,)
```

KNN

In [28]:
```
# selecting the K value
import math
print(math.sqrt(len(y_test)))
```

```
9.539392014169456
```

In [30]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=9)

knn.fit(X_train,y_train)

y_pred=knn.predict(X_test)

from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_pred,y_test)
print(cm)
knnac=accuracy_score(y_pred,y_test)
print(knnac)
print()
from sklearn.model_selection import cross_val_score
k=cross_val_score(knn,X,y,cv=10)
print(k)
k.max()
```

```
[[16 13]
 [22 40]]
0.6153846153846154

[0.64516129 0.64516129 0.58064516 0.67741935 0.70967742 0.53333333
 0.7        0.76666667 0.5862069  0.72413793]
```

Out[30]:
```
0.7666666666666667
```

In [27]:
```
knn_scores = []
for k in range(1,21):
    knn_classifier = KNeighborsClassifier(n_neighbors = k)
    score=cross_val_score(knn_classifier,X,y,cv=10)
    knn_scores.append(score.max())
knn_scores=pd.DataFrame(knn_scores)
print(knn_scores.max())
```

```
0    0.870968
dtype: float64
```

NB

In [32]:
```
from sklearn.naive_bayes import GaussianNB
nb= GaussianNB()

nb.fit(X_train,y_train)

y_pred=nb.predict(X_test)

from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_pred,y_test)
print(cm)
nbac=accuracy_score(y_pred,y_test)
nbac

from sklearn.model_selection import cross_val_score
n=cross_val_score(nb,X,y,cv=10)
print()
print(n.max())
```

```
[[25  8]
 [13 45]]

0.8709677419354839
```

Decission Tree

In [34]:
```
from sklearn.tree import DecisionTreeClassifier
tree = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)

tree.fit(X_train,y_train)

y_pred=tree.predict(X_test)

from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_pred,y_test)
print(cm)
treeac=accuracy_score(y_pred,y_test)
print()
print(treeac)

from sklearn.model_selection import cross_val_score
tr=cross_val_score(tree,X,y,cv=10)
print()
print(tr)
tr.max()
```

```
[[27 11]
 [11 42]]

0.7582417582417582

[0.70967742 0.80645161 0.90322581 0.77419355 0.77419355 0.7
 0.73333333 0.76666667 0.72413793 0.75862069]
```

Out[34]:
```
0.9032258064516129
```

In [35]:
```
from sklearn.ensemble import RandomForestClassifier
forest = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)

forest.fit(X_train,y_train)

y_pred=forest.predict(X_test)

from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_pred,y_test)
print(cm)
print()
forestac=accuracy_score(y_pred,y_test)
print(forestac)

from sklearn.model_selection import cross_val_score
fo=cross_val_score(forest,X,y,cv=10)
print()
print(fo)
fo.max()
```

```
[[31 15]
 [ 7 38]]

0.7582417582417582

[0.83870968 0.70967742 0.87096774 0.90322581 0.96774194 0.8
 0.73333333 0.83333333 0.68965517 0.86206897]
```

Out[35]:
```
0.967741935483871
```

In [36]:
```
print('knn---------------',knnac)
print('NB----------------',nbac)
print('decision tree-----',treeac)
print('random forest-----',forestac)
```

```
knn--------------- 0.6153846153846154
NB---------------- 0.7692307692307693
decision tree----- 0.7582417582417582
random forest----- 0.7582417582417582
```

In [41]:
```
# After applying the cross_val_scores
print('knn---------------',knn_scores.max())
print('NB----------------',n.max())
print('decision tree-----',tr.max())
print('random forest-----',fo.max())
```

```
knn--------------- 0    0.870968
dtype: float64
NB---------------- 0.8709677419354839
decision tree----- 0.9032258064516129
random forest----- 0.967741935483871
```