In [1]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import sklearn
from sklearn import linear_model
```

In [2]:
```
df = pd.read_csv("heart.csv")
df.head()
```

Out[2]:

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

In [3]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

df = pd.read_csv("heart.csv")
features = df.drop('age', axis=1)
target = df['target']
X_train, X_test, y_train, y_test = train_test_split(features, target,
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=
rf_classifier.fit(X_train, y_train)
print("Prediction:", rf_classifier.predict(X_test.head(1)))
```

```
Prediction: [0]
```

In [4]:
```
#Random forest
```

In [8]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

df = pd.read_csv("heart.csv")
x = df.drop('target', axis=1)
y= df['target']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.
kavya = RandomForestClassifier(n_estimators=100, criterion='entropy',
kavya.fit(x_train, y_train)
score=kavya.score(x_test, y_test)
print(score)
```

```
0.9853658536585366
```

In [6]:
```
# Naive Bayes
```

In [9]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
df = pd.read_csv("heart.csv")
x = df.drop('target', axis=1)
y= df['target']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.
kavya = GaussianNB()
kavya.fit(x_train, y_train)
score=kavya.score(x_test, y_test)
print(score)
```

```
0.8
```

In [10]:
```
#SVM
```

In [11]:
```
from sklearn.svm import SVC
df = pd.read_csv("heart.csv")
x = df.drop('target', axis=1)
y= df['target']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.
kavya =SVC()
kavya.fit(x_train, y_train)
score=kavya.score(x_test, y_test)
print(score)
```

```
0.6829268292682927
```

In [12]:
```
#Decision Tree
```

In [13]:
```
from sklearn.tree import DecisionTreeClassifier
df = pd.read_csv("heart.csv")
x = df.drop('target', axis=1)
y= df['target']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.
kavya =DecisionTreeClassifier()
kavya.fit(x_train, y_train)
score=kavya.score(x_test, y_test)
print(score)
```

```
0.9853658536585366
```

In [14]:
```
#KNN
```

In [15]:
```
from sklearn.neighbors import KNeighborsClassifier
df = pd.read_csv("heart.csv")
x = df.drop('target', axis=1)
y= df['target']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.
kavya =KNeighborsClassifier()
kavya.fit(x_train, y_train)
score=kavya.score(x_test, y_test)
print(score)
```

```
0.7317073170731707
```

In [16]:
```
#Logistic Regression
```

In [17]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
log_reg = LogisticRegression(solver='saga', max_iter=1000)
log_reg.fit(x_train_scaled, y_train)
y_pred = log_reg.predict(x_test_scaled)
score = log_reg.score(x_test_scaled, y_test)
print("Accuracy Score:",score)
```

```
Accuracy Score: 0.7951219512195122
```

In [ ]: