In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:

```
data=pd.read_csv("heart.csv")
```

In [4]:

```
data
```

Out[4]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

1025 rows × 14 columns

In [5]:

```
data.columns
```

Out[5]:

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [6]:

```
data.shape
```

Out[6]:

```
(1025, 14)
```

In [9]:

```
data.isnull().sum()
```

Out[9]:

```
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 [10]:

```
x=data.drop(["target"],axis=1)
y=data["target"]
```

In [11]:

```
x
```

Out[11]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3

1025 rows × 13 columns

In [12]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(x,y,random_state=42,test_size=0.2)
```

In [18]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
```

model=DecisionTreeClassifier() model.fit(X_train,Y_train) prediction=model.predict()

In [19]:

```
model=DecisionTreeClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

In [22]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[22]:

```
0.9853658536585366
```

In [24]:

```
model=RandomForestClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

In [25]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[25]:

```
0.9853658536585366
```

In [26]:

```
model=KNeighborsClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228:
FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the defaul
t behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, thi
s behavior will change: the default value of `keepdims` will become False, the `axis`
over which the statistic is taken will be eliminated, and the value None will no long
er be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [27]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[27]:

```
0.7317073170731707
```

In [ ]:

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
data=pd.read_csv("heart.csv")
```

In [4]:
```
data
```

Out[4]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

1025 rows × 14 columns

In [5]:
```
data.columns
```

Out[5]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [6]:
```
data.shape
```

Out[6]:
```
(1025, 14)
```

In [9]:
```
data.isnull().sum()
```

Out[9]:

```
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 [10]:

```
x=data.drop(["target"],axis=1)
y=data["target"]
```

In [11]:

```
x
```

Out[11]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3

1025 rows × 13 columns

In [12]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(x,y,random_state=42,test_size=0.2)
```

In [18]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
```

model=DecisionTreeClassifier() model.fit(X_train,Y_train) prediction=model.predict()

In [19]:

```
model=DecisionTreeClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

In [22]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[22]:

```
0.9853658536585366
```

In [24]:

```
model=RandomForestClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

In [25]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[25]:

```
0.9853658536585366
```

In [26]:

```
model=KNeighborsClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228:
FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the defaul
t behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, thi
s behavior will change: the default value of `keepdims` will become False, the `axis`
over which the statistic is taken will be eliminated, and the value None will no long
er be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [27]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[27]:

```
0.7317073170731707
```

In [ ]:

```

```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
data=pd.read_csv("heart.csv")
```

In [4]:
```
data
```

Out[4]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

1025 rows × 14 columns

In [5]:
```
data.columns
```

Out[5]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [6]:
```
data.shape
```

Out[6]:
```
(1025, 14)
```

In [9]:
```
data.isnull().sum()
```

Out[9]:

```
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 [10]:

```
x=data.drop(["target"],axis=1)
y=data["target"]
```

In [11]:

```
x
```

Out[11]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3

1025 rows × 13 columns

In [12]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(x,y,random_state=42,test_size=0.2)
```

In [18]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
```

model=DecisionTreeClassifier() model.fit(X_train,Y_train) prediction=model.predict()

In [19]:
```
model=DecisionTreeClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

In [22]:
```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[22]:
```
0.9853658536585366
```

In [24]:
```
model=RandomForestClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

In [25]:
```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[25]:
```
0.9853658536585366
```

In [26]:
```
model=KNeighborsClassifier()
model.fit(X_train,Y_train)
prediction=model.predict(X_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228:
FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the defaul
t behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, thi
s behavior will change: the default value of `keepdims` will become False, the `axis`
over which the statistic is taken will be eliminated, and the value None will no long
er be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [27]:
```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test,prediction)
accuracy
```

Out[27]:
```
0.7317073170731707
```

In [ ]: