In [2]:
```
%pip install -U scikit-learn==0.22.2
```

```
Collecting scikit-learn==0.22.2
  Downloading https://files.pythonhosted.org/packages/e1/7f/366dcba1ba076a88a50bea732dbc033c0c5bbf7876010e6edc67948579d5/scikit_learn-0.22.2-cp36-cp36m-manylinux1_x86_64.whl (7.1MB)
     |████████████████████████████████| 7.1MB 4.3MB/s
Requirement already satisfied, skipping upgrade: scipy>=0.17.0 in /usr/local/lib/python3.6/dist-packages (from scikit-learn==0.22.2) (1.4.1)
Requirement already satisfied, skipping upgrade: numpy>=1.11.0 in /usr/local/lib/python3.6/dist-packages (from scikit-learn==0.22.2) (1.18.5)
Requirement already satisfied, skipping upgrade: joblib>=0.11 in /usr/local/lib/python3.6/dist-packages (from scikit-learn==0.22.2) (0.16.0)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.22.2.post1
    Uninstalling scikit-learn-0.22.2.post1:
      Successfully uninstalled scikit-learn-0.22.2.post1
Successfully installed scikit-learn-0.22.2
```

In [31]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import keras
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network  import MLPClassifier
from sklearn.ensemble import RandomForestClassifier,BaggingClassifier,AdaBoostClassifier,VotingClassifier
from sklearn.svm import SVC
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
import joblib
from sklearn.model_selection import KFold
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score,cross_val_predict,KFold
import seaborn as sns
```

In [4]:
```
print('The scikit-learn version is {}.'.format(sklearn.__version__))
```

```
The scikit-learn version is 0.22.2.
```

In [ ]:
```
from google.colab import drive
```

```
drive.mount('/content/drive')list(data.sourcemmsi).count('2115')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%3aoob&response_type=code&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:
..........
Mounted at /content/drive
```

In [6]:
```
data=pd.read_csv("/content/Heart Disease Dataset.csv")
x=(data.drop("target",1).drop("oldpeak",1))
y=np.array(data["target"])
data
```

Out[6]:

	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 [7]:
```
train_x,test_x,train_y,test_y= sklearn.model_selection.train_test_split(x,y,test_size=0.3)
train_x
```

Out[7]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	slope	ca	thal
284	61	1	0	140	207	0	0	138	1	2	1	3
65	35	0	0	138	183	0	1	182	0	2	0	2
157	35	1	1	122	192	0	1	174	0	2	0	2
13	64	1	3	110	211	0	0	144	1	1	0	2
254	59	1	3	160	273	0	0	125	0	2	0	2
...	...	...	...	...	...	...	...	...	...	...	...	...
123	54	0	2	108	267	0	0	167	0	2	0	2
230	47	1	2	108	243	0	1	152	0	2	0	2
198	62	1	0	120	267	0	1	99	1	1	2	3
144	76	0	2	140	197	0	2	116	0	1	0	2
82	60	0	2	102	318	0	1	160	0	2	1	2

212 rows × 12 columns

In [8]:
```
modelRF=  RandomForestClassifier(random_state = 1,
                                 n_estimators = 750,
                                 max_depth = 15,
                                 min_samples_split = 5,  min_samples_leaf = 1)
modelRF.fit(train_x,train_y)
acc=modelRF.score(test_x,test_y)
acc
```

Out[8]: 0.8571428571428571

In [9]:
```
#selected..
modelDTB= BaggingClassifier(RandomForestClassifier(random_state = 1,
                                 n_estimators = 750,
                                 max_depth = 15,
                                 min_samples_split = 5,  min_samples_leaf = 1),n_estimators=10,max_features=1.0,max_
modelDTB.fit(train_x,train_y)
acc=modelDTB.score(test_x,test_y)
acc
```

Out[9]: 0.9010989010989011

In [10]:
```
modelDTA= AdaBoostClassifier(RandomForestClassifier(random_state = 1,
                                n_estimators = 750,
                                max_depth = 15,
                                min_samples_split = 5,  min_samples_leaf = 1),n_estimators=10,learning_rate=0.5)
modelDTA.fit(train_x,train_y)
acc=modelDTA.score(test_x,test_y)
acc
```

Out[10]: 0.8571428571428571

In [14]:
```
modelDTN= MLPClassifier(hidden_layer_sizes=128,activation="relu" )
modelDTN.fit(train_x,train_y)
acc=modelDTN.score(test_x,test_y)
acc
```

Out[14]: 0.8571428571428571

In [12]:
```
modelDTN=  BaggingClassifier(GaussianNB())
modelDTN.fit(train_x,train_y)
acc=modelDTN.score(test_x,test_y)
acc
```

Out[12]: 0.8571428571428571

In [16]:
```
# votion classifier
nn=MLPClassifier(hidden_layer_sizes=150,activation="relu")
ac=BaggingClassifier(DecisionTreeClassifier())
ba=BaggingClassifier(GaussianNB(),n_estimators=50,max_features=1.0,max_samples=0.5)
#rfc=RandomForestClassifier(n_estimators=1000)
#dt=DecisionTreeClassifier()
sv=SVC(C=1.0, kernel='rbf', degree=3, gamma='auto')
rfad=BaggingClassifier(RandomForestClassifier(random_state = 1,
                                n_estimators = 100,
                                max_depth = 15,
                                min_samples_split = 5,  min_samples_leaf = 1),n_estimators=10,max_features=1.0,max_
#rf=RandomForestClassifier(n_estimators=100)
lr=BaggingClassifier(LogisticRegression(max_iter=10000,penalty='l2'),n_estimators=50,max_features=1.0,max_samples=0.5
```

In [17]:
```
vc= VotingClassifier(estimators=[('nn',nn),('ba',ba),('sv',sv),('rfad',rfad),('lr',lr)], voting='hard')
```

In [20]:
```
vc.fit(train_x,train_y)
acc=vc.score(test_x,test_y)
joblib.dump(vc, "heart_disease_prediction.pkl")
print(acc)
```

```
0.9230769230769231
```

In [24]:
```
test=joblib.load('/content/heart_disease_prediction.pkl')
```

In [40]:
```
test.score(test_x,test_y)
```

Out[40]:
```
0.9120879120879121
```

In [39]:
```
k_fold = KFold(n_splits=10)
score=cross_val_score(vc, x, y, cv=k_fold, n_jobs=-1)
score.mean()
```

Out[39]:
```
0.7581720430107526
```