

In [1]:

```

import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import random
import matplotlib.pyplot
%matplotlib inline

#model validation
from sklearn.metrics import classification_report, confusion_matrix, precision_score, f1_score
from sklearn import metrics

# cross validation
from sklearn.model_selection import StratifiedKFold

from sklearn.naive_bayes import GaussianNB

```

In [2]:

```

from sklearn.model_selection import train_test_split

```

In [3]:

```

#import the heart disease dataset
df = pd.read_csv("heart_statlog.csv")
df.head()

```

Out[3]:

	age	sex	chest	resting_blood_pressure	serum_cholesterol	fasting_blood_sugar	resting_ele
0	70	1	4	130	322	0	
1	67	0	3	115	564	0	
2	57	1	2	124	261	0	
3	64	1	4	128	263	0	
4	74	0	2	120	269	0	

In [4]:

```

from sklearn import preprocessing
le = preprocessing.LabelEncoder()
df['class'] = le.fit_transform(df["class"])

```

In [5]:

```

# segregating dataset into features i.e., X and target variables i.e., y
X = df.drop(['class'],axis=1)
y = df['class']

```

In [6]:

```
#splitting the model into training and testing set
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.2, shuffle
```

In [7]:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train[['age', 'resting_blood_pressure', 'serum_cholesterol', 'maximum_heart_rate_achieved', '
X_train.head()
```

Out[7]:

	age	sex	chest	resting_blood_pressure	serum_cholesterol	fasting_blood_sugar	resi
<b>197</b>	0.465116	0	3	0.10	0.302405	0	
<b>174</b>	0.000000	1	1	0.18	0.192440	0	
<b>173</b>	0.790698	0	3	0.20	0.292096	0	
<b>249</b>	0.604651	1	4	0.45	0.536082	0	
<b>207</b>	0.558140	1	3	0.05	0.391753	0	

In [8]:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_test[['age', 'resting_blood_pressure', 'serum_cholesterol', 'maximum_heart_rate_achieved', 'o
X_test.head()
```

Out[8]:

	age	sex	chest	resting_blood_pressure	serum_cholesterol	fasting_blood_sugar	resi
<b>260</b>	0.617021	0	3	0.309524	0.460241	0	
<b>194</b>	0.404255	1	3	0.357143	0.255422	1	
<b>112</b>	0.659574	0	4	0.761905	0.375904	0	
<b>14</b>	0.595745	0	4	0.404762	0.371084	0	
<b>206</b>	0.659574	0	3	0.095238	0.407229	0	

In [9]:

```

from sklearn import model_selection
from sklearn.model_selection import cross_val_score

# function initializing baseline machine learning models
def GetBasedModel():
    basedModels = []
    basedModels.append(('NB' , GaussianNB()))
    return basedModels

# function for performing 10-fold cross validation of all the baseline models
def BasedLine2(X_train, y_train,models):
    # Test options and evaluation metric
    num_folds = 10
    scoring = 'accuracy'
    seed = 7
    results = []
    names = []
    for name, model in models:
        kfold = model_selection.KFold(n_splits=10, random_state=None)
        cv_results = model_selection.cross_val_score(model, X_train, y_train, cv=kfold, sco
        results.append(cv_results)
        names.append(name)
        msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
        print(msg)

    return results,msg

```

In [10]:

```

models = GetBasedModel()
names,results = BasedLine2(X_train, y_train,models)

```

NB: 0.831818 (0.104351)

In [11]:

```

NB = GaussianNB()
NB.fit(X_train,y_train)
y_pred_NB = NB.predict(X_test)

```

In [12]:

```

CM=confusion_matrix(y_test,y_pred_NB)

TN = CM[0][0]
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
specificity = TN/(TN+FP)
sensitivity = TP/(TP+FN)
acc= accuracy_score(y_test, y_pred_NB)
prec = precision_score(y_test, y_pred_NB)
rec = recall_score(y_test, y_pred_NB)
f1 = f1_score(y_test, y_pred_NB)

model_results =pd.DataFrame([[ 'NB',acc, prec,rec,f1,sensitivity,specificity]],
                             columns = [ 'Model', 'Accuracy', 'Precision', 'Recall', 'f1 score', 'Sensitivity'

model_results

```

Out[12]:

	Model	Accuracy	Precision	Recall	f1 score	Sensitivity	Specificity
0	NB	0.814815	0.733333	0.916667	0.814815	0.916667	0.733333

In [13]:

```

pip install sklearn-genetic

```

Note: you may need to restart the kernel to use updated packages.

'C:\Users\Nupur' is not recognized as an internal or external command, operable program or batch file.

In [14]:

```
from genetic_selection import GeneticSelectionCV

from sklearn.neighbors import KNeighborsClassifier
#build the model with your preferred hyperparameters.
NB = GaussianNB()

# create the GeneticSelection search with the different parameters available.
NB = GeneticSelectionCV(NB,
                        cv=10,
                        scoring="accuracy",
                        max_features=13,
                        n_population=270,
                        crossover_proba=0.5,
                        mutation_proba=0.2,
                        n_generations=50,
                        crossover_independent_proba=0.5,
                        mutation_independent_proba=0.05,
                        n_gen_no_change=10,
                        n_jobs=-1)

# fit the GA search to our data.
NB = NB.fit(X_train, y_train)

# print the results.
print(NB.support_)
```

```
[False True True False True False False True True True False True
 True]
```

In [15]:

```
y_pred_NB = NB.predict(X_test)
print("Accuracy score after genetic algorithm is= "+str(accuracy_score(y_test,y_pred_NB)))
```

```
Accuracy score after genetic algorithm is= 0.8518518518518519
```

In [16]:

```
CM=confusion_matrix(y_test,y_pred_NB)

TN = CM[0][0]
FN = CM[1][0]
TP = CM[1][1]
FP = CM[0][1]
specificity = TN/(TN+FP)
sensitivity = TP/(TP+FN)
acc= accuracy_score(y_test, y_pred_NB)
prec = precision_score(y_test, y_pred_NB)
rec = recall_score(y_test, y_pred_NB)
f1 = f1_score(y_test, y_pred_NB)

model_results =pd.DataFrame([[ 'NB',acc, prec,rec,f1,sensitivity,specificity]],
                             columns = [ 'Model', 'Accuracy','Precision','Recall','f1 score','Sensitivity'

model_results
```

Out[16]:

	Model	Accuracy	Precision	Recall	f1 score	Sensitivity	Specificity
0	NB	0.851852	0.833333	0.833333	0.833333	0.833333	0.866667