In [1]:

```
# import library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
```

In [2]:

```
#baca file
data = pd.read_csv('../spesifikasi/tubes2_HeartDisease_train.csv')
data.rename(columns = {'Column1':'Age',
                               'Column2':'Sex',
                               'Column3':'Chest-Pain Type',
                               'Column4':'Resting Blood Pressure',
                               'Column5':'Serum Cholestrol',
                               'Column6':'Fasting Blood Sugar',
                               'Column7':'Resting ECG',
                               'Column8':'Max Heart Rate Achieved',
                               'Column9':'Exercise Induced Angina',
                               'Column10':'ST Depression Induced',
                               'Column11':'Peak Exercise ST',
                               'Column12':'Number of Major Vessels',
                               'Column13':'Thal',
                               'Column14':'Diagnose'
                              }, inplace = True)

# handle missing value
data.replace({'?' : None, 'None' : None}, inplace=True)
data['Resting Blood Pressure'].fillna(value = data['Resting Blood Pressure'].median(),inplace=True)
data['Serum Cholestrol'].fillna(value = data['Serum Cholestrol'].median(),inplace=True)
data['Fasting Blood Sugar'].fillna(value = data['Fasting Blood Sugar'].mode()[0],inplace=True)
data['Resting ECG'].fillna(value = data['Resting ECG'].mode()[0],inplace=True)
data['Max Heart Rate Achieved'].fillna(value = data['Max Heart Rate Achieved'].median(),inplace=True)
data['Exercise Induced Angina'].fillna(value = data['Exercise Induced Angina'].mode()[0],inplace=True)
data['ST Depression Induced'].fillna(value = data['ST Depression Induced'].median(),inplace=True)
data['Peak Exercise ST'].fillna(value = data['Peak Exercise ST'].mode()[0],inplace=True)
data['Number of Major Vessels'].fillna(value = data['Number of Major Vessels'].mode()[0],inplace=True)
data['Thal'].fillna(value = data['Thal'].mode()[0],inplace=True)

def classify_column(key, min_range, data, change='int64'):
    data[key] = data[key].astype(change)
    data[key] = data[key] / min_range
    data[key] = data[key].astype('int64')

#make all column categorical
classify_column('Age', 5, data);
classify_column('Resting Blood Pressure', 10, data)
classify_column('Serum Cholestrol', 20, data)
classify_column('Max Heart Rate Achieved', 15, data)
classify_column('ST Depression Induced', 5, data, 'float64')
data
```

Out[2]:

	Age	Sex	Chest-Pain Type	Resting Blood Pressure	Serum Cholestrol	Fasting Blood Sugar	Resting ECG	Max Heart Rate Achieved	Exercise Induced Angina	ST Depression Induced
0	10	1	4	12	10	0	0	9	0	0
1	11	1	4	15	10	0	0	7	1	0
2	10	0	3	13	15	1	0	11	0	0
3	9	0	3	12	9	0	0	8	0	0
4	10	1	4	12	0	0	1	10	1	0
5	12	0	4	13	15	0	0	8	0	0
6	12	1	4	13	15	0	0	9	1	0
7	11	1	2	13	12	0	0	7	0	0
8	8	1	2	15	13	0	0	9	0	0
9	10	1	3	12	12	0	2	9	0	0
10	11	1	4	11	0	0	0	9	0	0
11	9	1	3	11	7	0	2	8	0	1
12	12	1	1	14	10	1	1	6	0	0
13	11	0	2	13	17	0	0	11	0	2
14	12	1	4	13	10	0	1	9	1	0
15	13	1	4	13	18	0	0	9	0	0
16	12	1	3	16	0	0	0	4	1	0
17	11	1	4	15	13	0	2	7	1	1
18	8	1	1	12	14	0	1	10	0	0
19	13	1	4	12	12	1	0	10	0	0
20	11	1	4	11	0	0	0	6	0	0
21	9	0	2	11	8	0	0	9	0	0
22	10	1	3	13	8	0	0	10	0	0
23	10	1	4	12	12	0	0	7	1	0
24	10	0	3	12	14	0	2	10	0	1
25	10	1	3	15	8	1	1	10	0	0
26	12	0	4	18	16	0	0	10	1	0
27	10	1	4	10	11	1	0	9	0	0
28	8	1	4	10	0	0	1	7	0	0
29	11	1	4	14	10	0	0	8	1	0
...	...	...	...	...	...	...	...	...	...	...
749	9	0	3	13	10	0	1	9	0	0
750	6	1	2	11	11	0	0	12	0	0
751	11	1	2	15	11	0	2	10	0	0

	Age	Sex	Chest-Pain Type	Resting Blood Pressure	Serum Cholestrol	Fasting Blood Sugar	Resting ECG	Max Heart Rate Achieved	Exercise Induced Angina	ST Depression Induced
752	14	1	4	15	15	0	0	7	1	0
753	11	1	3	15	10	1	0	10	0	3
754	10	1	4	13	0	0	2	9	1	0
755	7	1	2	12	9	0	0	11	0	0
756	6	0	2	13	8	0	0	12	0	0
757	11	1	3	13	11	0	1	9	0	0
758	10	0	3	11	10	0	0	10	0	3
759	12	1	4	13	0	0	0	5	1	0
760	7	1	3	11	12	0	2	11	0	0
761	11	1	4	15	11	0	1	8	1	3
762	10	1	2	12	0	0	0	6	0	0
763	10	1	3	12	9	0	0	10	0	0
764	10	0	2	12	11	1	0	9	0	0
765	11	1	4	12	0	0	1	6	1	0
766	11	1	4	12	15	0	2	11	0	0
767	8	1	4	13	12	1	2	9	1	0
768	11	1	4	12	0	0	0	7	0	0
769	14	1	4	16	11	1	2	8	0	0
770	10	1	4	16	12	0	1	5	1	0
771	12	1	3	13	14	0	0	9	0	0
772	10	1	4	11	0	0	0	8	1	0
773	13	1	3	18	13	1	2	10	1	3
774	12	0	4	14	13	0	2	10	0	7
775	12	1	2	13	0	0	0	9	0	0
776	10	1	1	12	10	0	2	8	1	2
777	12	1	4	13	9	0	0	9	0	0
778	11	1	3	13	12	1	1	9	0	0

779 rows × 14 columns

In [3]:

```
#column 1- 13 and 14
X = data.drop(['Diagnose'], axis=1)
y = data['Diagnose']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, rand
om_state=4126)

#train
gnb = GaussianNB()
model = gnb.fit(X_train, y_train)

predicted = model.predict(X_test)
gnb_acc = accuracy_score(y_test, predicted) * 100
gnb_prec = precision_score(y_test, predicted, average='macro') * 100
gnb_rec = recall_score(y_test, predicted, average='macro') * 100
cnf_matrix_gnb = confusion_matrix(y_test, predicted)

print('y_test')
print('Akurasi Naive Bayes =', gnb_acc, '%')
print('Presisi Naive Bayes =', gnb_prec, '%')
print('Recall Naive Bayes =', gnb_rec, '%', '\n')
print('Confusion Matrix')
print(cnf_matrix_gnb)

print()
print()

predicted = model.predict(X_train)
gnb_acc = accuracy_score(y_train, predicted) * 100
gnb_prec = precision_score(y_train, predicted, average='macro') * 100
gnb_rec = recall_score(y_train, predicted, average='macro') * 100
cnf_matrix_gnb = confusion_matrix(y_train, predicted)

print('y_test')
print('Akurasi Naive Bayes =', gnb_acc, '%')
print('Presisi Naive Bayes =', gnb_prec, '%')
print('Recall Naive Bayes =', gnb_rec, '%', '\n')
print('Confusion Matrix')
print(cnf_matrix_gnb)
```

```
y_test
Akurasi Naive Bayes = 63.07692307692307 %
Presisi Naive Bayes = 41.54836427939876 %
Recall Naive Bayes = 40.84188188832151 %

Confusion Matrix
[[77  7  1  0  0]
 [10 37  5 11  3]
 [ 1  6  4  7  1]
 [ 2  7  4  4  1]
 [ 1  1  1  3  1]]


y_test
Akurasi Naive Bayes = 57.1917808219178 %
Presisi Naive Bayes = 43.15058313681003 %
Recall Naive Bayes = 43.02733656022619 %

Confusion Matrix
[[206  41   7   6   4]
 [ 46  85  13  15   0]
 [  7  28  20  12   6]
 [  7  30  11  18   6]
 [  1   2   4   4   5]]
```

In [4]:

```
from sklearn.externals import joblib

#save model
joblib.dump(gnb, 'gnb_model.joblib')
```

Out[4]:

```
['gnb_model.joblib']
```

In [5]:

```
#load model
loaded_gnb = joblib.load('gnb_model.joblib')

#baca file test
data_test = pd.read_csv('../spesifikasi/tubes2_HeartDisease_test.csv')
data_test.rename(columns = {'Column1':'Age',
                            'Column2':'Sex',
                            'Column3':'Chest-Pain Type',
                            'Column4':'Resting Blood Pressure',
                            'Column5':'Serum Cholestrol',
                            'Column6':'Fasting Blood Sugar',
                            'Column7':'Resting ECG',
                            'Column8':'Max Heart Rate Achieved',
                            'Column9':'Exercise Induced Angina',
                            'Column10':'ST Depression Induced',
                            'Column11':'Peak Exercise ST',
                            'Column12':'Number of Major Vessels',
                            'Column13':'Thal',
                            'Column14':'Diagnose'
                           }, inplace = True)

# handle missing value
data_test.replace({'?' : None, 'None' : None}, inplace=True)
data_test['Resting Blood Pressure'].fillna(value = data_test['Resting Blood Pr
essure'].median(),inplace=True)
data_test['Serum Cholestrol'].fillna(value = data_test['Serum Cholestrol'].med
ian(),inplace=True)
data_test['Fasting Blood Sugar'].fillna(value = data_test['Fasting Blood Suga
r'].mode()[0],inplace=True)
data_test['Resting ECG'].fillna(value = data_test['Resting ECG'].mode()[0],inp
lace=True)
data_test['Max Heart Rate Achieved'].fillna(value = data_test['Max Heart Rate
 Achieved'].median(),inplace=True)
data_test['Exercise Induced Angina'].fillna(value = data_test['Exercise Induce
d Angina'].mode()[0],inplace=True)
data_test['ST Depression Induced'].fillna(value = data_test['ST Depression Ind
uced'].median(),inplace=True)
data_test['Peak Exercise ST'].fillna(value = data_test['Peak Exercise ST'].mod
e()[0],inplace=True)
data_test['Number of Major Vessels'].fillna(value = data_test['Number of Major
 Vessels'].mode()[0],inplace=True)
data_test['Thal'].fillna(value = data_test['Thal'].mode()[0],inplace=True)

#make all column categorical
classify_column('Age', 5, data_test);
classify_column('Resting Blood Pressure', 10, data_test)
classify_column('Serum Cholestrol', 20, data_test)
classify_column('Max Heart Rate Achieved', 15, data_test)
classify_column('ST Depression Induced', 5, data_test, 'float64')
data_test
```

Out[5]:

	Age	Sex	Chest-Pain Type	Resting Blood Pressure	Serum Cholestrol	Fasting Blood Sugar	Resting ECG	Max Heart Rate Achieved	Exercise Induced Angina	ST Depression Induced
0	12	1	2	16	13	1	1	10	0	0
1	12	1	4	14	10	0	0	10	0	0
2	10	1	4	13	12	0	0	6	1	0
3	9	1	4	12	13	0	0	7	0	0
4	11	0	1	13	15	0	0	6	0	0
5	11	1	3	13	11	0	1	8	0	0
6	10	1	4	14	14	0	0	8	1	8
7	12	1	4	13	8	0	1	8	1	0
8	9	0	3	16	9	0	0	10	0	0
9	13	0	3	11	28	0	2	10	0	3
10	8	1	4	11	0	0	0	9	1	0
11	11	1	4	14	12	0	0	6	1	0
12	10	1	2	13	11	0	0	10	0	0
13	13	1	4	11	12	0	2	10	0	1
14	10	1	2	17	10	0	1	7	0	0
15	11	1	4	16	14	1	2	8	0	0
16	11	1	4	12	0	1	1	9	1	0
17	5	1	2	12	12	0	0	10	0	0
18	10	1	3	13	16	0	0	8	0	0
19	11	1	4	10	11	0	0	10	0	0
20	15	0	3	14	9	0	1	7	0	2
21	13	1	4	13	0	0	0	8	1	0
22	9	1	4	11	10	0	0	9	0	0
23	10	1	4	13	10	1	0	7	1	0
24	11	1	4	11	0	0	1	5	0	0
25	10	0	3	12	10	0	2	7	0	0
26	7	1	4	11	0	0	0	8	1	0
27	12	1	4	16	11	1	0	7	1	0
28	13	1	4	11	10	0	2	8	1	0
29	11	1	3	10	0	0	0	9	0	0
...	...	...	...	...	...	...	...	...	...	...
111	9	1	1	13	0	0	1	9	0	0
112	10	1	3	13	9	1	2	10	0	2
113	8	0	4	10	13	0	2	8	0	1

	Age	Sex	Chest-Pain Type	Resting Blood Pressure	Serum Cholestrol	Fasting Blood Sugar	Resting ECG	Max Heart Rate Achieved	Exercise Induced Angina	ST Depression Induced
114	7	0	4	14	8	0	0	10	0	0
115	10	1	4	14	10	1	2	10	1	6
116	8	1	2	12	9	0	0	10	0	0
117	13	1	3	11	10	1	2	6	1	0
118	10	1	1	12	8	0	0	9	0	0
119	12	1	3	15	12	1	0	9	1	0
120	13	1	4	13	10	1	1	8	0	0
121	10	1	4	14	17	0	2	8	1	0
122	8	1	2	12	13	0	0	11	0	0
123	8	1	3	10	12	0	0	5	1	0
124	14	1	3	12	10	0	0	6	1	0
125	10	1	2	12	9	0	0	9	0	0
126	9	1	3	12	9	0	0	9	0	0
127	12	1	4	15	10	1	0	7	1	0
128	11	1	4	13	19	1	2	8	0	0
129	13	1	3	12	0	0	1	8	0	0
130	9	0	2	11	10	0	0	10	0	0
131	10	0	4	13	15	0	0	9	1	2
132	12	1	4	15	0	0	1	5	0	0
133	12	1	4	15	0	0	1	10	0	0
134	13	1	4	17	13	1	0	7	1	0
135	11	1	4	16	8	1	2	6	0	0
136	12	1	2	13	8	0	1	8	0	0
137	8	1	3	16	7	0	0	9	0	0
138	13	1	1	13	12	0	0	8	0	0
139	10	1	4	13	9	0	0	9	0	0
140	7	1	2	12	10	0	1	9	0	0

141 rows × 13 columns

In [6]:

```
result = loaded_gnb.predict(data_test)
result
```

Out[6]:

```
array([1, 2, 1, 1, 0, 1, 4, 1, 0, 1, 1, 1, 0, 2, 1, 3, 2, 0, 1, 1, 0, 2,
       0, 2, 2, 0, 1, 3, 2, 1, 0, 1, 2, 0, 1, 0, 0, 1, 0, 0, 1, 0, 4, 1,
       2, 1, 2, 2, 1, 1, 1, 1, 2, 2, 1, 1, 1, 3, 1, 0, 1, 0, 3, 4, 0, 0,
       1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 4, 1, 2, 0, 3, 1, 0,
       3, 1, 3, 0, 0, 0, 0, 1, 0, 1, 1, 0, 3, 0, 1, 0, 2, 1, 0, 1, 3, 3,
       0, 1, 2, 0, 0, 4, 0, 3, 0, 1, 3, 2, 0, 1, 1, 0, 2, 3, 3, 2, 0, 1,
       3, 2, 3, 3, 1, 1, 0, 1, 0], dtype=int64)
```