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

import warnings
warnings.filterwarnings('ignore')

dataset=pd.read_csv(r"D:\DS_NIT\Machine Learning\Classifications\Social_Network_Ads.csv")

x=dataset.iloc[:,[2,3]].values
y=dataset.iloc[:,-1].values

#Splitting the dataset into training and test set
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)

#Feature scaling
from sklearn.preprocessing import MinMaxScaler
sc=MinMaxScaler()
#from sklearn.preprocessing import Normalizer
#sc=Normalizer()
x_train=sc.fit_transform(x_train)
x_test=sc.transform(x_test)


#Training the naive bayes modele on the training set
from sklearn.naive_bayes import MultinomialNB
classifier=MultinomialNB()
classifier.fit(x_train,y_train)

#predict the test set results
y_pred=classifier.predict(x_test)

#Making the confusion matrix
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,y_pred)
print(cm)

from sklearn.metrics import accuracy_score
ac=accuracy_score(y_test,y_pred)
print(ac)

bias=classifier.score(x_train,y_train)
print(bias)

variance=classifier.score(x_test,y_test)
print(variance)

from sklearn.metrics import classification_report
cr=classification_report(y_test,y_pred)
print(cr)

#bernoulli Naive Bayes
from sklearn.naive_bayes import BernoulliNB
bernouli_cl=BernoulliNB()
bernouli_cl.fit(x_train,y_train)

bernouli_ypred=bernouli_cl.predict(x_test)
#Confusion matrix
bern_cm=confusion_matrix(y_test,bernouli_ypred)
print(bern_cm)

#accuracy
bern_ac=accuracy_score(y_test,bernouli_ypred)
print(bern_ac)
```

```
#bias
bern_bias=bernouli_cl.score(x_train,y_train)
print(bern_bias)

#variance
bern_var=bernouli_cl.score(x_test,y_test)
print(bern_var)

from sklearn.metrics import classification_report
bern_cr=classification_report(y_test,bernouli_ypred)
print(bern_cr)

from sklearn.naive_bayes import GaussianNB
gausian_cl=GaussianNB()
gausian_cl.fit(x_train,y_train)

gausian_ypred=gausian_cl.predict(x_test)

#accuracy
gausian_ac=accuracy_score(y_test,gausian_ypred)
print(gausian_ac)

#bias
gausian_bias=gausian_cl.score(x_train,y_train)
gausian_bias()

#variance
gausian_variance=gausian_cl.score(x_test,y_test)
print(gausian_variance)

#classification report
gausian_cr=classification_report(y_test,gausian_ypred)
print(gausian_cr)
```

without scaling			
	accuracy	bias	variance
	0.5625	0.67	0.56
with scaling-Normalizer			
	0.72	0.62	0.72
with scaling-StandardScaler			
MultinomialNB does not work with negative values. Since StandardScaler can produce negative values			
with scaling-MinMaxScaler			
	0.72	0.62	0.72

Multinominal				
	precision	**recall**	**f1-score**	**support**
0	0.72	1	0.94	58
1	0	0	0.83	22
accuracy			0.72	80
macro avg	0.36	0.5	0.42	80
weighted_avg	0.53	0.72	0.61	80

Bernouli				
	precision	recall	f1-score	support
0	0.72	1	0.84	58
1	0	0	0.83	22
accuracy			0.72	80
macro avg	0.36	0.5	0.42	80
weighted_avg	0.53	0.72	0.61	80

Gaussian				
	precision	recall	f1-score	support
0	0.93	0.95	0.94	58
1	0.86	0.82	0.84	22
accuracy			0.91	80
macro avg	0.89	0.88	0.89	80
weighted_avg	0.91	0.91	0.91	80