Naive Bayes is a classification algorithm, which uses Bayes theorem of probability for prediction of unknown class. It uses probability to decide which class a test point belongs to. Naive Bayes is a purely statistical model. This algorithm is called Naive due to the assumption that the features/ attributes in the datasets are mutually independent.

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
```

```
data=pd.read_csv('/content/diabetesdata.csv')
```

```
data.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Out
0	6	148	72	35	0	33.6	0.627	50	
1	1	85	66	29	0	26.6	0.351	31	
2	8	183	64	0	0	23.3	0.672	32	
3	1	89	66	23	94	28.1	0.167	21	
4	0	137	40	35	168	43.1	2.288	33	

```
X=data.drop(columns='Outcome',axis=0)
y=data['Outcome']
```

```
scalar = StandardScaler()
```

```
scalar.fit(X)
```

```
    StandardScaler()
```

```
standardized_data = scalar.transform(X)
print(standardized_data)
```

```
    [[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
       1.4259954 ]
     [-0.84488505 -1.12339636 -0.16054575 ... -0.68442195 -0.36506078
      -0.19067191]
     [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
      -0.10558415]
     ...
     [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
      -0.27575966]
     [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
       1.17073215]
     [-0.84488505 -0.8730192   0.04624525 ... -0.20212881 -0.47378505
      -0.87137393]]
```

```
X = standardized_data
y = data['Outcome']
```

```
X_train,X_test,Y_train,Y_test = train_test_split(X,y,test_size=0.3,stratify=y,random_state= 2)
```

```
print(X.shape,X_train.shape,X_test.shape)
```

```
    (768, 8) (537, 8) (231, 8)
```

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train,Y_train)
y_pred =model.predict(X_test)
```

```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(Y_test,y_pred))
```

```
Accuracy: 0.7748917748917749
```

```
test_pred=model.predict(X_test)
```

```
print(metrics.classification_report(Y_test,test_pred))
print(metrics.confusion_matrix(Y_test,test_pred))
```

```
              precision    recall  f1-score   support

           0       0.79      0.89      0.84       150
           1       0.73      0.57      0.64        81

    accuracy                           0.77       231
   macro avg       0.76      0.73      0.74       231
weighted avg       0.77      0.77      0.77       231

[[133  17]
 [ 35  46]]
```

These metrics are calculated using True Positive/TP (person has diabetes and predicted diabetes) , True Negative/TN (person did not have diabetes and predicted non- diabetic), False Positive/FP (person did not have diabetes but predicted diabetes) and False Negative/FN (person had diabetes but predicted non-diabetic).