In the previous video, we saw the implementation of GaussianNB in Sklearn. GaussianNB is used with continuos data, whereas in the previous video, labelled(categorical) data has been used.

We converted the Iris dataset into labelled(categorical) data, and used it with GaussianNB. This is why we got poor result (0.90 precision) in the previous video.

The correct implementation is given as follows:

In [1]:
```
from sklearn import naive_bayes, datasets
from sklearn import model_selection
iris = datasets.load_iris()
X = iris.data
Y = iris.target
```

In [2]:
```
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y, test_size=0.25, random_state=
0)
```

In [3]:
```
gnb = naive_bayes.GaussianNB() # GAUSSIAN NAIVE BAYES CLASSIFIER
gnb.fit(X_train, Y_train)
```

Out[3]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

In [4]:
```
y_pred = gnb.predict(X_test)
```

In [5]:
```
# Various metrics for understanding how well the model has performed.
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

print("Classification Report")
print(classification_report(Y_test,y_pred))
print("Confusion Matrix")
print(confusion_matrix(Y_test,y_pred))
print()
print("Accuracy Score")
print(accuracy_score(Y_test,y_pred) * 100, "%", sep="")
```

```
Classification Report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        16
           2       1.00      1.00      1.00         9

    accuracy                           1.00        38
   macro avg       1.00      1.00      1.00        38
weighted avg       1.00      1.00      1.00        38

Confusion Matrix
[[13  0  0]
 [ 0 16  0]
 [ 0  0  9]]

Accuracy Score
100.0%
```

It is seen that we achieve **100% accuracy**.