Lab Assignment: 6

Objective: To implement Naive Bayes and apply on iris dataset.

Name: Aakash Verma

Reg. No.: 24-08-26

Course: M.Tech.(Cyber Security)

In [11]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

In [12]:
```
class NaiveBayes:
    def fit(self, X, y):
        self.classes, counts = np.unique(y, return_counts=True)
        self.priors = counts / len(y)
        self.means = {}
        self.variances = {}

        for label in self.classes:
            X_label = X[y == label]
            self.means[label] = X_label.mean(axis=0)
            self.variances[label] = X_label.var(axis=0)

    def predict(self, X):
        return np.array([self._predict_single(x) for x in X])

    def _predict_single(self, x):
        posteriors = []
        for label in self.classes:
            prior = np.log(self.priors[np.where(self.classes == label)[0][0]
            likelihood = -0.5 * np.sum(np.log(2 * np.pi * self.variances[lab
            likelihood -= 0.5 * np.sum(((x - self.means[label]) ** 2) / sel+
            posterior = prior + likelihood
            posteriors.append(posterior)
        return self.classes[np.argmax(posteriors)]
```

In [13]:
```
# Load Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
```

In [14]:
```
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran
```

In [15]:
```
# Initialize and fit the Naive Bayes classifier
nb = NaiveBayes()
nb.fit(X_train, y_train)
```

In [16]:
```
# Make predictions
predictions = nb.predict(X_test)
```

In [17]:
```
# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
print(f'Accuracy: {accuracy:.2f}')
```

```
Accuracy: 1.00
```

Conclusion:

1. Naive Bayes effectively classified Iris species based on conditional probabilities, achieving strong accuracy due to its reliance on the assumption of feature independence.

2. The implementation highlighted Naive Bayes' simplicity and computational efficiency, making it an excellent choice for quick and effective classification tasks in machine learning.

In [ ]:
```
```