```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

iris = load_iris()
X, y = iris.data, iris.target

class NaiveBayes:
    def fit(self, X, y):
        self._classes = np.unique(y)
        self._mean = np.array([X[y == c].mean(axis=0) for c in self._classes])
        self._var = np.array([X[y == c].var(axis=0) for c in self._classes])
        self._priors = np.array([X[y == c].shape[0] / len(y) for c in self._classes])

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

    def _predict(self, x):
        posteriors = [np.log(priors) + np.sum(np.log(self._pdf(idx, x))) for idx, priors in enumerate(self._priors)]
        return self._classes[np.argmax(posteriors)]

    def _pdf(self, idx, x):
        mean, var = self._mean[idx], self._var[idx]
        numerator = np.exp(- (x - mean) ** 2 / (2 * var))
        denominator = np.sqrt(2 * np.pi * var)
        return numerator / denominator

X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.3)
nb = NaiveBayes()
nb.fit(X_train, Y_train)
y_pred = nb.predict(X_test)
print('Accuracy: ' + str(np.mean(y_pred == Y_test)))
```

```
Accuracy: 0.9777777777777777
```