### 1. Naïve Bayes Classifier

**A probabilistic classification algorithm based on Bayes' Theorem, assuming feature independence. In this practical, we use Gaussian Naïve Bayes, which assumes features follow a normal distribution.**

### 2. Bayes' Theorem

**The core formula behind Naïve Bayes:**

$$P(A|B)=P(B|A)\cdot P(A)P(B)P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}P(A|B)=P(B)P(B|A)\cdot P(A)$$

**This is used to calculate the probability of a class given the input features.**

### 3. Supervised Learning

**Naïve Bayes is a supervised learning technique because the model is trained on labeled data to predict output labels.**

### 4. Classification

**The task of predicting categorical labels (e.g., species names in Iris dataset). Naïve Bayes performs multi-class classification in this case.**

### 5. Iris Dataset

**A classic dataset containing 150 rows of flower measurements (sepal/petal length and width) and a label (Species). Commonly used for classification benchmarking.**

### 6. Data Preprocessing

- **Label Encoding: Converts text labels into numerical format.**
- **Train-Test Split: Splits dataset into training and testing sets for evaluation.**

### 7. Model Evaluation Metrics

- **Confusion Matrix: Matrix showing true/false positives/negatives.**
- **Accuracy, Precision, Recall, F1-Score: Key performance metrics.**

---

## 🔁 Related Topics

### 1. Gaussian Distribution

**Gaussian Naïve Bayes assumes each feature is normally distributed. Knowledge of mean, variance, and bell-curve behavior is important.**

### 2. Machine Learning Pipeline

**From loading data, preprocessing, model training, prediction, to evaluation. Understanding the full pipeline is essential in real-world ML projects.**

### 3. Overfitting vs Underfitting

**Overfitting happens when a model learns the training data too well, including noise Naïve Bayes, being simple, tends to generalize better and is less prone to overfitting.**

### 4. Scikit-learn (sklearn)

**Python ML library used here. Knowing how to use classes like `GaussianNB`, `train_test_split`, and `confusion_matrix` is key.**

### 5. Label Encoding vs One-Hot Encoding

**Here, `LabelEncoder` is used since class labels are categorical and need to be converted to numerical values. One-hot encoding is another technique used when multiple categorical features exist.**

### 6. Evaluation Metric Trade-offs

**Precision vs Recall trade-off, especially in imbalanced datasets. Choosing the right metric is context-dependent (e.g., spam detection prioritizes precision).**

### 7. Data Cleaning

**Checking `isnull().sum()` ensures there are no missing values, which could affect model training and evaluation.**