Chapter 4 Classification

we cannot compare the normal logistic regression

Linear regression --> Least square
Logistic regression --> maximum likelihood

Logistic Regression is a specific type of Generalized Linear Model (Often Abbreviated GLM). Generalized Linear Models are a generalization of the concepts and abilities of regular Linear Models.

4. Classification

Main goal
Predict a **qualitative** (categorical) response variable Y instead of a quantitative one.

Three classic examples used throughout the chapter:

1. Emergency room diagnosis: stroke, drug overdose, epileptic seizure?
2. Online banking: is this transaction fraudulent?
3. DNA sequence: is this mutation disease-causing or not?

4.1 An Overview of Classification

Key idea
Classification assigns an observation to one of a set of discrete categories (classes).
We still have training data (x_i, y_i) pairs, but now y_i is a label (e.g., Yes/No, stroke/overdose/seizure, default/no default).

Most widely used methods covered in this chapter:

- Logistic regression
- Linear Discriminant Analysis (LDA)
- Quadratic Discriminant Analysis (QDA)
- Naïve Bayes
- K-nearest neighbors (mentioned briefly)

4.2 Why Not Linear Regression?

Core problem
Linear regression assumes Y is continuous and unbounded. When Y is binary (0/1) or multi-class, it produces nonsense predictions (probabilities < 0 or > 1).

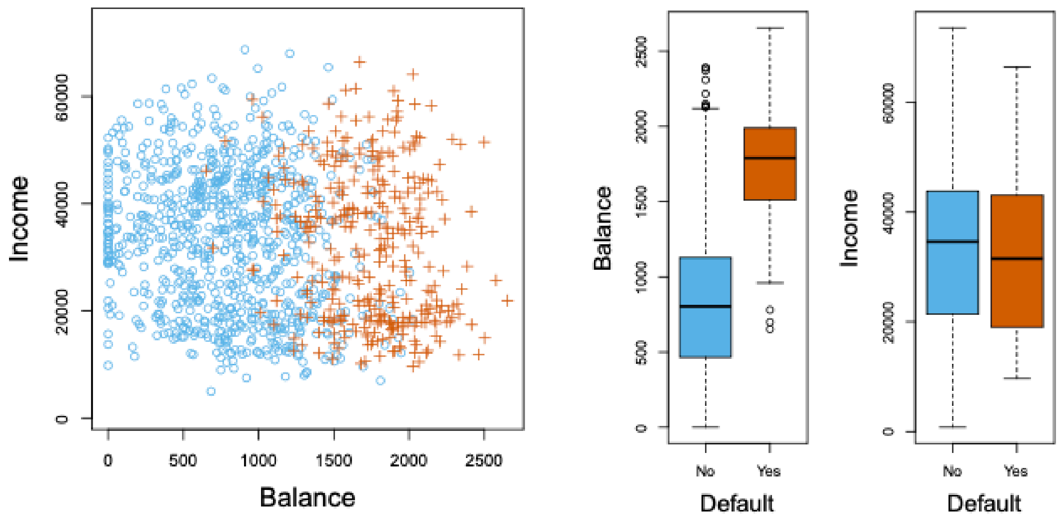


Figure 4.1

- Left: annual income vs monthly credit card balance, colored by default status
- Center & right: boxplots showing defaulters tend to have much higher balances

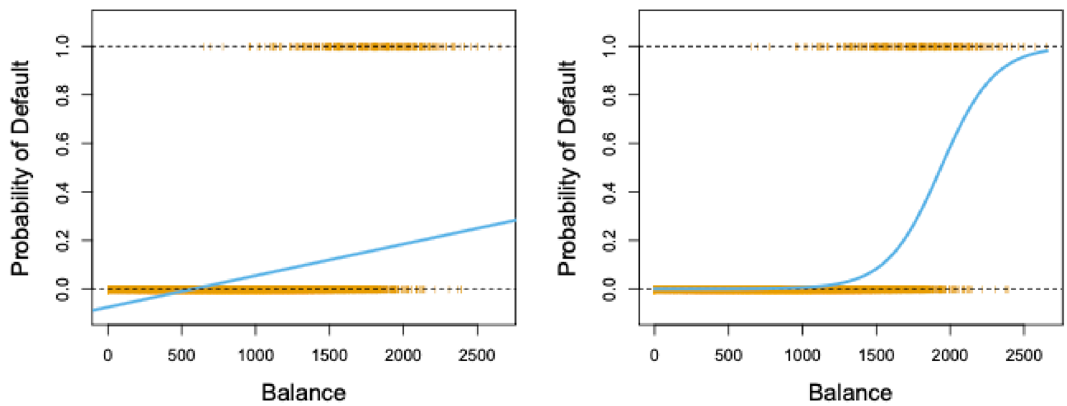


Figure 4.2 (most important figure in 4.2)

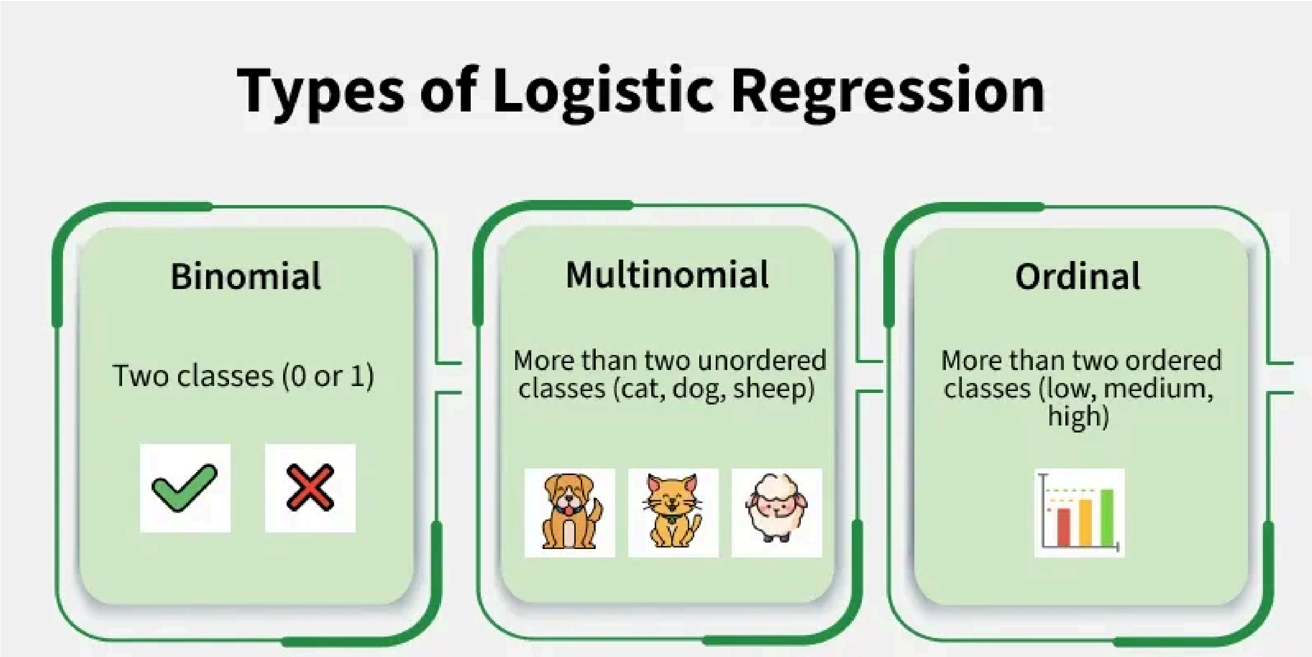
- Left panel: linear regression fit → line goes below 0 and above 1
- Right panel: logistic regression fit → nice S-shaped curve, stays in [0,1]

Takeaway

Linear regression is inappropriate for classification because:

- Predicted values can fall outside [0,1]
- The relationship between predictors and probability is usually not linear

4.3 Logistic Regression



4.3.1 The Logistic Model

Core equation

Instead of modeling Pr(Y=1|X) directly, model the **log-odds** linearly:

$$\log\left(p(X) / (1 - p(X)) \right) = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p$$

Solving for probability gives the **logistic (sigmoid) function**:

$$p(X) = e^{(\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p)} / (1 + e^{(\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p)})$$
$$= 1 / (1 + e^{-(\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p)})$$

Figure 4.2 (right panel) shows this S-shape perfectly on the Default data using balance.

4.3.2 Estimating the Regression Coefficients

We use **maximum likelihood** (not least squares) to find $\hat{\beta}$.

	Coefficient	Std. error	z-statistic	p-value
`Intercept`	−10.6513	0.3612	−29.5	<0.0001
`balance`	0.0055	0.0002	24.9	<0.0001

Table 4.1 (logistic regression with only balance)

- Intercept very negative → low baseline default risk
- $\hat{\beta}$ balance ≈ 0.00555 → very strong positive effect

	Coefficient	Std. error	z-statistic	p-value
`Intercept`	−3.5041	0.0707	−49.55	<0.0001
`student[Yes]`	0.4049	0.1150	3.52	0.0004

Table 4.2 (logistic regression with only student status)

- Students appear to have slightly higher default probability

	Coefficient	Std. error	z-statistic	p-value
`Intercept`	−10.8690	0.4923	−22.08	<0.0001
`balance`	0.0057	0.0002	24.74	<0.0001
`income`	0.0030	0.0082	0.37	0.7115
`student[Yes]`	−0.6468	0.2362	−2.74	0.0062

Table 4.3 (multiple logistic regression: balance + income + student)

- Now student coefficient is **negative** → students are actually less likely to default when balance and income are controlled for

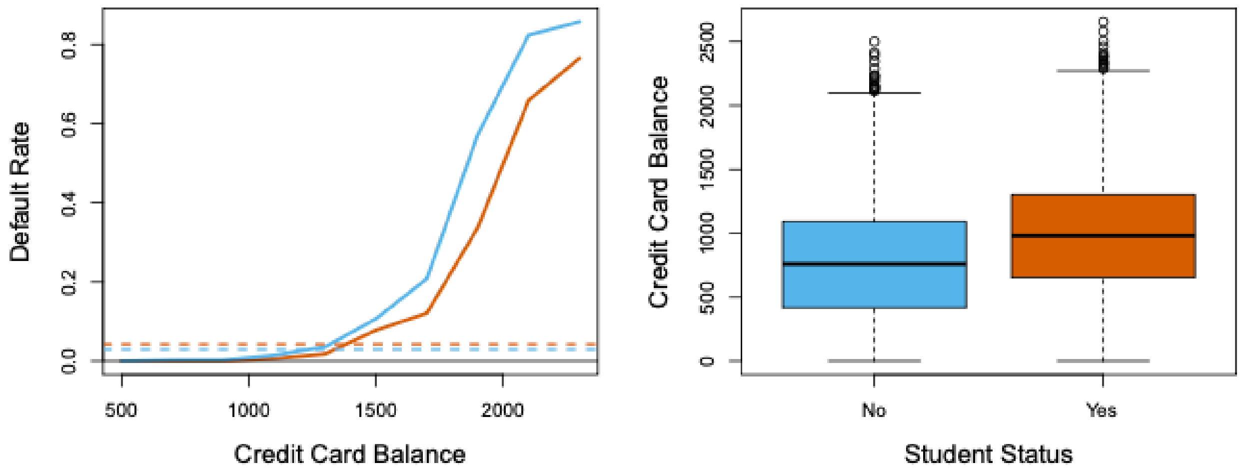


Figure 4.3 (classic confounding illustration)

- Left: default rate vs balance → separate lines for students (orange) and non-students (blue)
- Right: **box plot** → students have much higher balances on average
  → **Confounding**: students borrow more, so look riskier overall, but are safer at the same balance level.

+R-Square and P-Value

There is no consensus way to calculate the r-square and p-value for logistic regression.

McFadden's Pseudo R-square

LL(fit) log likelihood fitted line

LL(Overall Probability)

And do it like the R-square in the Simple linear model

$$R^2 = \frac{\text{LL(overall probability) - LL(fit)}}{\text{LL(overall probability)}}$$

Chi-Squared value = 2(LL(Fit)-LL(Overall Probability))

4.3.3 Making Predictions

Once $\hat{\beta}$ are estimated, plug new X into the logistic formula to get predicted probability.

Example from Table 4.1:
balance = \$1,000 → predicted Pr(default) ≈ 0.00576 (very low)
balance = \$2,000 → predicted Pr(default) ≈ 0.0586 (much higher)

4.3.4 Multiple Logistic Regression

General form:

$\log(p(X)/(1-p(X))) = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p$

Same interpretation: each β_j is the change in log-odds per 1-unit change in X_j (holding others fixed).

4.3.5 Multinomial Logistic Regression

Extension to $K > 2$ classes (e.g., stroke / overdose / seizure).
One class is chosen as baseline; we model log-odds relative to that baseline for each of the other classes.

4.4 Generative Models for Classification

Big idea
Instead of directly modeling Pr(Y=k|X), model how X is generated in each class → then use Bayes rule.

$$Pr(Y = k|X = x) = Pr(X = x|Y = k) \times Pr(Y = k)/Pr(X = x)$$

Y=k given X
Three main methods:

Why discriminant Analysis?
-When classes are well-separated, the parameter estimate for the logistic regression model are

surprisingly unstable. so the linear discriminant analysis does not suffer from this problem.
-If n is small and the distribution of the predictors X is approximately normal in each of the classes, the linear discriminant model is again more stable than the logistic regression model.
-Linear discriminant analysis is popular when we have more than two response classes, because it also provides low-dimensional views of the data.

4.4.1 Linear Discriminant Analysis for p = 1

The Gaussian density

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma_k}\right)^2}$$

Where
Uk is the mean, and variance (in class k). we assume that variance K = normal variance

Assume each class is normal with same variance σ^2.

So plugging it into the Bayes Formula, we get the following

$$p_k(x) = \Pr(Y = k|X = x):$$

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma}\right)^2}}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu_l}{\sigma}\right)^2}}$$

To classify at the value $X = x$, we need to see which of the $p_k(x)$ is largest. Taking logs, and discarding terms that do not depend on k, we see that this is equivalent to assigning x to the class with the largest *discriminant score*:

$$\delta_k(x) = x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

Note that $\delta_k(x)$ is a *linear* function of x.

If there are $K = 2$ classes and $\pi_1 = \pi_2 = 0.5$, then one can see that the *decision boundary* is at

$$x = \frac{\mu_1 + \mu_2}{2}.$$

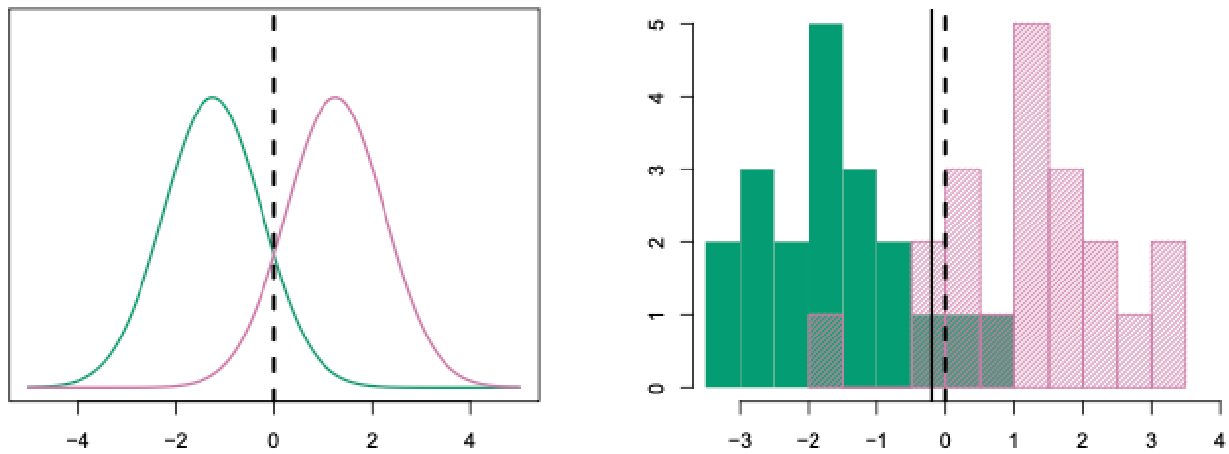


Figure 4.4

- Left: two overlapping normal densities + Bayes decision boundary (dashed vertical line)
- Right: 20 sampled points per class + histogram + LDA boundary (solid)

4.4.2 Linear Discriminant Analysis for p > 1

Assume multivariate normal with class-specific means μ_k but **common covariance matrix Σ** for all classes.

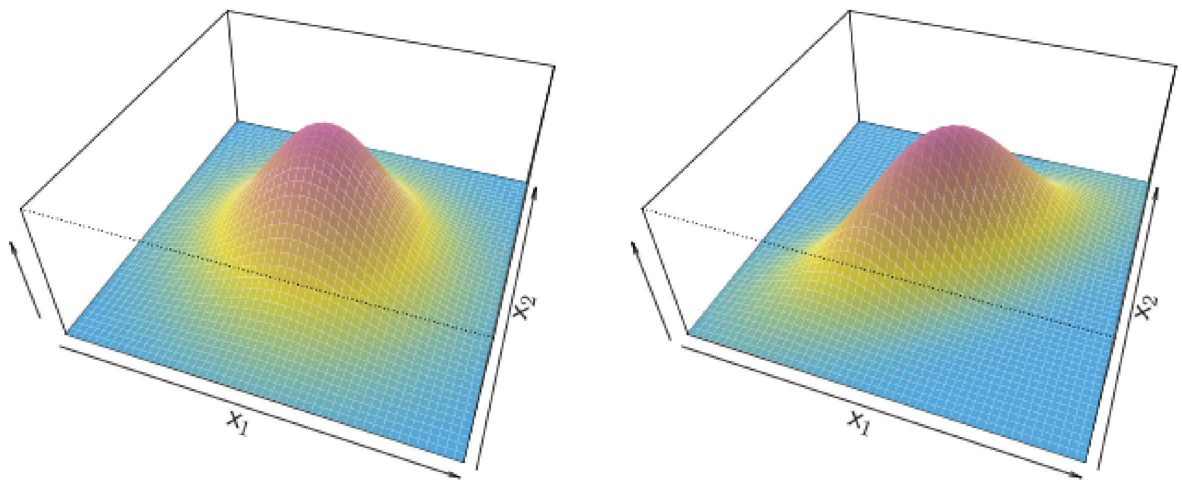


Figure 4.5

- Two multivariate Gaussians (p=2) with correlation 0.7 → elliptical contours

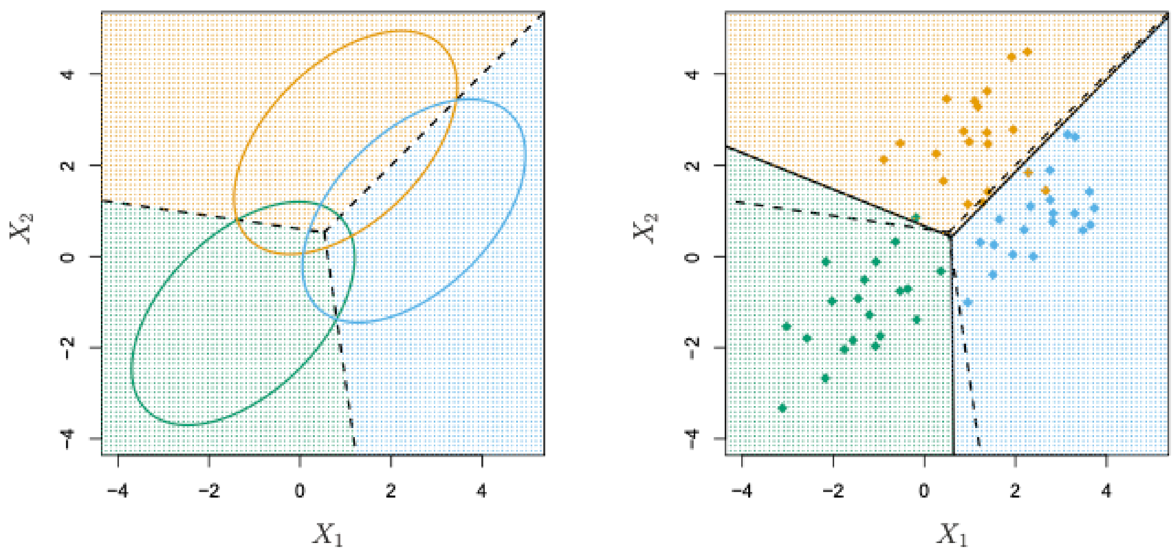


Figure 4.6 (very important)

- Left: three classes, 95% probability ellipses, Bayes boundaries (dashed), LDA boundaries (solid)
- Right: actual 20 points per class sampled → LDA boundaries very close to Bayes

Confusion matrix examples (Default data, 10,000 observations)

		True default status		
		No	Yes	Total
Predicted default status	No	9644	252	9896
	Yes	23	81	104
	Total	9667	333	10000

Table 4.4 (LDA with default threshold 0.5)

- Overall error ≈ 2.75%
- But misses 252 / 333 = 75.7% of actual defaulters → very low sensitivity

		True default status		
		No	Yes	Total
Predicted	No	9432	138	9570
default status	Yes	235	195	430
	Total	9667	333	10000

Table 4.5 (LDA with lower threshold ~20%)

- Predicts many more defaults (430 instead of 104)
- Catches more true defaulters but increases false positives

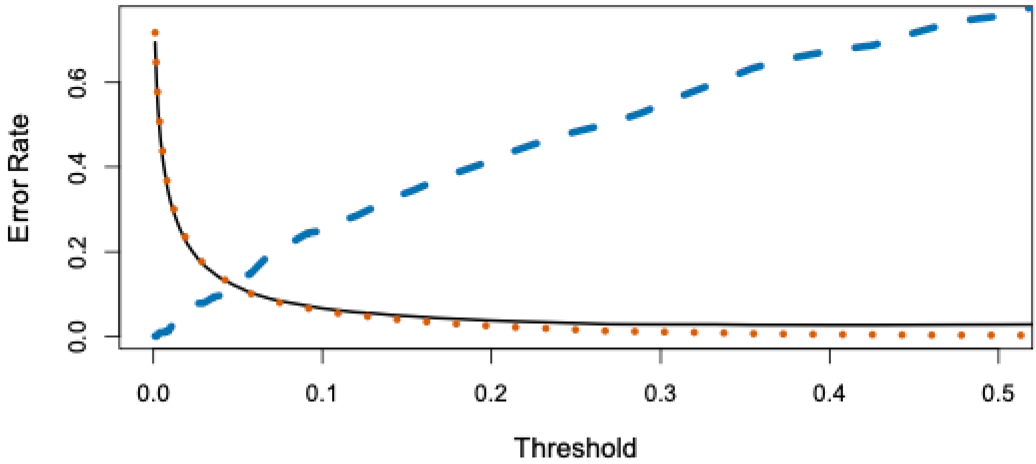


Figure 4.7

Error rates vs threshold:

- Black = overall error
- Blue dashed = error among true defaulters
- Orange dotted = error among true non-defaulters

→ Clear trade-off

4.4.3 Quadratic Discriminant Analysis

Same as LDA but **allows different covariance matrices Σ_k** for each class → decision boundaries become **quadratic**.

ROC Curve (Figure 4.8)

- x-axis = False Positive Rate (1 − specificity)
- y-axis = True Positive Rate (sensitivity / recall)
- AUC ≈ 0.95 for LDA on Default data → very good performance

Summary table – when to prefer which method

METHOD	ASSUMPTION	BOUNDARY SHAPE	BEST WHEN ...
Logistic Regression	linear in log-odds	linear	many predictors, no strong normality needed
LDA	multivariate normal + equal Σ	linear	classes roughly normal, similar spread
QDA	multivariate normal + different Σ_k	quadratic	clear difference in variance between classes
Naïve Bayes	features independent given class	usually linear	high-dimensional data (text, genomics), sparse

4.4.4 Naive bayes

Naive bayes is a ML classification algorithm that predicts the category of a data point using probability.
It assume that all features are independent of each other.
Ex: best use in spam filtering, document categorisation, sentiment analysis

Naive bayes use **Bayes's Theorem** to classify data based on the probabilities of different classes given the features of the data. It is used mostly in high-dimensional text classification.

Naive bayes's Characteristic:
-Very few number of parameters --> faster than other classification algorithm
-It assumes that each feature contributes to the predictions with no relation between each other

Why is it called Naive Bayes?
Because it assumes the presence of one feature does not affect other features.

Assumption of the Naive Bayes
-**Feature independence**: when we are trying to classify sth, each feature or piece of information in the data does not affect any other feature.
-**Continuos features are normally distributed**: If a feature is continuous, then it is assumed to be normally distributed within each class.
-**Discrete features have multinomial distributions**: If a feature is discrete, then it is assumed to have a multinomial distribution within each class.
-**Features are equally important:** all feature are assumed to contribute equally to the prediction of the class label.
-**No missing data**: the data should not contain any missing values.

Bayes's Theorem

$$P(y|X) = \frac{P(X|y) \cdot P(y)}{P(X)}$$

Where:

- $P(y|X)$: Posterior probability, probability of class y given features X
- $P(X|y)$: Likelihood, probability of features X given class y
- $P(y)$: Prior probability of class y
- $P(X)$: Marginal likelihood or evidence

The "naive" in Naive Bayes comes from the assumption that all features are independent given the class. That is:

$$P(x_1, x_2, ..., x_n|y) = P(x_1|y) \cdot P(x_2|y) \cdots P(x_n|y)$$

Thus, Bayes' theorem becomes:

$$P(y|x_1, ..., x_n) = \frac{P(y) \cdot \prod_{i=1}^{n} P(x_i|y)}{P(X)}$$

Since the denominator is constant for a given input, we can write:

$$P(y|x_1, ..., x_n) \propto P(y) \cdot \prod_{i=1}^{n} P(x_i|y)$$

We compute the posterior for each class y and choose the class with the highest probability:

$$\hat{y} = \arg\max_y P(y) \cdot \prod_{i=1}^{n} P(x_i|y)$$

This becomes our Naive Bayes classifier.

Naive Bayes for Continuous Features

For continuous features, we assume a Guassian Distribution

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Type of Naive Bayes Model

1. **Gaussian Naive Bayes**: Continuous values associated with each feature are assumed to be distributed according to a gaussian distribution.
2. **Multinomial naive Bayes**: when features represent the frequency of terms (word count) in a doc. Commonly applied in text classification
3. **Bernoulli Naive Bayes**: binary feature, where each feature indicates whether a word appears or not in a document. It is suited for scenarios where the presence or absence of terms is more relevant than their frequency. Both models are widely used in document classification tasks