Naive Bayes Classification — A Tutorial

Naive Bayes is a family of simple, fast, and surprisingly effective probabilistic classifiers. This tutorial starts from the basic idea and then works through the deeper properties that explain *why* it behaves the way it does — from its hidden connection to linear models, through smoothing and numerical stability, to the estimation theory underneath it.

1. The core idea

We have a data point described by features $\mathbf{x} = (x_1, \ldots, x_d)$, and we want to assign it to one of several classes C. Bayes' theorem lets us turn the question around:

$$P(C \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid C)\, P(C)}{P(\mathbf{x})}.$$

Since $P(\mathbf{x})$ is the same for every class, choosing the most probable class means maximizing the numerator:

$$\hat{C} = \arg\max_C \; P(C)\, P(\mathbf{x} \mid C).$$

The hard part is $P(\mathbf{x} \mid C)$ — the joint distribution over all features. Naive Bayes makes one bold simplifying assumption to make it tractable: **the features are conditionally independent given the class.** That turns the joint into a product:

$$\hat{C} = \arg\max_C \; P(C) \prod_{j=1}^{d} P(x_j \mid C).$$

Each factor $P(x_j \mid C)$ is easy to estimate by counting. That single assumption is the whole trick.

2. Why "naive," and does the assumption hurt?

Key idea: the *class-conditional independence* assumption, and the difference between **estimating probabilities** and **choosing the winning class**.

The name comes from the assumption itself: real features are rarely independent, so treating them as if they were is "naive." Yet the classifier often works very well anyway. The reason is subtle and worth understanding:

Classification only needs the **arg-max to be correct** — we just need the true class to score highest. It does *not* need the estimated probability to be accurate. When features are correlated, the independence assumption distorts the *magnitude* of $\hat{P}(C \mid \mathbf{x})$, but it frequently preserves the *ranking* of the classes. So Naive Bayes can be a poor probability estimator and still be an excellent classifier. This robustness — being optimal for classification even when the independence assumption is badly violated — was analyzed by Domingos and Pazzani (1997).

3. Naive Bayes is secretly a linear classifier

Key idea: the log of the posterior ratio is a straight-line (affine) function of the features.
Take two classes C_1 and C_0 and look at the log of their posterior ratio:

$$\log \frac{P(C_1 \mid \mathbf{x})}{P(C_0 \mid \mathbf{x})} = \log \frac{P(C_1)}{P(C_0)} + \sum_{j=1}^{d} \log \frac{P(x_j \mid C_1)}{P(x_j \mid C_0)}.$$

For categorical features (written as one-hot indicators), each term $\log \frac{P(x_j|C_1)}{P(x_j|C_0)}$ is just a constant multiplied by an indicator. So the whole expression has the form

$$\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}) + b,$$

and the decision boundary $\{\mathbf{x} : \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}) + b = 0\}$ is a hyperplane. Naive Bayes, logistic regression, and shared-covariance LDA all produce **linear** decision boundaries — what separates them is how they *estimate* the weights, not the shape of the boundary they can draw.

4. Gaussian Naive Bayes and the shape of the boundary

Key idea: plug a Gaussian into the log-ratio and see which terms survive.
For continuous features we model each as a normal distribution, $P(x_j \mid C) = \mathcal{N}(x_j; \mu_{jC}, \sigma^2_{jC})$, so

$$\log P(x_j \mid C) = -\tfrac{1}{2}\log(2\pi\sigma^2_{jC}) - \frac{(x_j - \mu_{jC})^2}{2\sigma^2_{jC}}.$$

Two cases:
Different variance per class ($\sigma_{jC_1} \neq \sigma_{jC_0}$): the x_j^2 terms do **not** cancel, so the boundary is **quadratic** (a diagonal-covariance special case of quadratic discriminant analysis).
Shared variance ($\sigma_{jC_1} = \sigma_{jC_0}$): the x_j^2 terms cancel, leaving a **linear** boundary (this is the link to linear discriminant analysis, and to Section 3).
So Gaussian Naive Bayes is a *curved* classifier in general; it becomes linear only when the variances are shared — and that same condition is what makes it coincide with logistic regression, as we see next.

5. Relationship to logistic regression

Key idea: Naive Bayes and logistic regression are a *generative–discriminative pair*.
Starting from the shared-variance Gaussian case above, the log-ratio is linear, $\log \frac{P(C_1|\mathbf{x})}{P(C_0|\mathbf{x})} = \mathbf{w}^\top \mathbf{x} + b$, which rearranges to

$$P(C_1 \mid \mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}} = \sigma(\mathbf{w}^\top \mathbf{x} + b).$$

That is *exactly* the logistic-regression model. The two methods share the same functional form but differ in philosophy:

Naive Bayes (generative): models how the data is generated — it estimates $P(\mathbf{x} \mid C)$ and $P(C)$ separately by counting. It can have higher error if its assumptions are wrong, but it converges to that error with very little data.
Logistic regression (discriminative): fits the weights directly to predict the class as well as possible. It usually reaches lower error in the limit, but needs more data to get there.
Ng and Jordan (2001) made this precise: the generative method approaches its asymptotic error faster (needing roughly on the order of $\log d$ examples, for d features), while the discriminative method reaches a lower asymptotic error but needs on the order of d examples. Same form, opposite trade-off — Naive Bayes tends to win with little data, logistic regression with plenty.

6. Laplace (add-one) smoothing, derived

Key idea: the zero-frequency problem, and smoothing as a Bayesian estimate.
If some feature value never appears with a class in the training data, its estimated probability is 0. Because the classifier multiplies all the factors together, that single zero wipes out the entire product — one unseen value can veto a whole class. This happens because the plain frequency estimate (the maximum-likelihood estimate) sits right at the edge of what's possible.
The fix has a clean justification. Place a symmetric Dirichlet prior $\boldsymbol{\theta} \sim \mathrm{Dirichlet}(\alpha, \ldots, \alpha)$ on the K category probabilities. With observed counts $n_1, \ldots, n_K$ (total N), the posterior is $\mathrm{Dirichlet}(n_1 + \alpha, \ldots, n_K + \alpha)$, and its mean is

$$\hat{\theta}_k = \frac{n_k + \alpha}{N + \alpha K}.$$

Setting $\alpha = 1$ gives **Laplace (add-one) smoothing**; a general α gives **Lidstone smoothing**. So smoothing is not an arbitrary patch — it is principled shrinkage toward a uniform prior, and the raw frequency estimate is simply the $\alpha \to 0$ limit.

7. Why we compute in log-space

Key idea: avoiding numerical underflow, and turning products into sums.
Multiplying many small probabilities together quickly produces a number too small for a computer to represent (underflow). Taking logarithms turns the product into a sum,

$$\log P(C) + \sum_j \log P(x_j \mid C),$$

which is numerically stable, and because $\log$ is increasing, the class with the largest log-score is the same as the class with the largest score. To convert log-scores back into normalized probabilities safely, use the **log-sum-exp** trick, subtracting the maximum first:

$$\log \sum_i e^{a_i} = a^* + \log \sum_i e^{a_i - a^*}, \qquad a^* = \max_i a_i.$$

8. Is Naive Bayes the "Bayes-optimal" classifier?

Key idea: don't confuse the ideal classifier with this particular approximation.

The **Bayes-optimal classifier** is the theoretical best: it uses the *true* posterior distribution and achieves the minimum possible error. Naive Bayes uses an *approximation* of that posterior, built on the independence assumption. The two coincide **only if** class-conditional independence actually holds in the real data. In general it does not, so Naive Bayes is a biased approximation of the ideal — sharing the word "Bayes" does not make it optimal.

9. What correlated or duplicated features do

Key idea: the same evidence gets counted twice.

Suppose two features are identical (or highly correlated). Because the model multiplies a separate factor for each feature, that shared piece of evidence is counted **twice**, as though it were two independent observations. The result is an over-confident posterior — probabilities pushed unrealistically close to 0 or 1 — and if the correlation is strong enough, the inflated evidence can even change the predicted class. This is why Naive Bayes probability estimates are often poorly calibrated even when its classifications are correct, echoing the point from Section 2: the winning class often survives, the probability value does not.

10. Handling a missing feature value

Key idea: under independence, you can simply drop the missing factor.

If a feature x_m is missing at prediction time, we marginalize (sum) over its possible values:

$$\sum_{x_m} P(C) \prod_j P(x_j \mid C) = P(C) \Big[\prod_{j \neq m} P(x_j \mid C) \Big] \underbrace{\sum_{x_m} P(x_m \mid C)}_{=1}.$$

Because the missing feature's probabilities sum to 1, its factor disappears cleanly. In practice you just leave that feature out of the product — no imputation required.

11. Estimating the parameters: MLE vs MAP

Key idea: frequencies versus frequencies-with-a-prior.

Maximum likelihood (MLE): the plain frequency, $\hat{P}(x_j = v \mid C) = \dfrac{\text{count}(x_j = v,\, C)}{\text{count}(C)}$.

Maximum a posteriori (MAP): the smoothed estimate with a Dirichlet prior, $\dfrac{\text{count} + \alpha}{N + \alpha K}$.

MLE is the special case $\alpha \to 0$. Adding a prior buys robustness in low-count situations at the cost of a little bias — the same smoothing idea from Section 6, viewed as parameter estimation.

12. A fully worked example

We use the classic *buys_computer* dataset (14 training records):

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no

Goal: classify $X = (\text{age} = \text{youth},\ \text{income} = \text{medium},\ \text{student} = \text{yes},\ \text{credit} = \text{fair})$.
There are 9 "yes" records and 5 "no," so the priors are $P(\text{yes}) = 9/14$ and $P(\text{no}) = 5/14$. Counting each feature value within each class:

feature value	$P(\cdot \mid \text{yes})$	$P(\cdot \mid \text{no})$
age = youth	2/9	3/5
income = medium	4/9	2/5
student = yes	6/9	1/5
credit = fair	6/9	2/5

Multiply the prior by the four likelihoods for each class:

$$P(X \mid \text{yes})\,P(\text{yes}) = \tfrac{9}{14} \cdot \tfrac{2}{9} \cdot \tfrac{4}{9} \cdot \tfrac{6}{9} \cdot \tfrac{6}{9} = \tfrac{16}{567} \approx 0.02822,$$

$$P(X \mid \text{no})\,P(\text{no}) = \tfrac{5}{14}\cdot\tfrac{3}{5}\cdot\tfrac{2}{5}\cdot\tfrac{1}{5}\cdot\tfrac{2}{5} = \tfrac{6}{875} \approx 0.006857.$$

The "yes" score is larger, so we **predict buys_computer = yes.** Normalizing gives the actual probability:

$$P(\text{yes} \mid X) = \frac{0.02822}{0.02822 + 0.006857} \approx 0.805.$$

A special case: if the value income = medium had never appeared with class "no," then $P(\text{medium} \mid \text{no}) = 0$ would zero out the entire "no" score regardless of the other evidence. This is exactly the situation Laplace smoothing fixes — for the 3 income categories we would use $\frac{0+1}{5+3}$ instead of $\frac{0}{5}$.

13. Cost-sensitive decisions: is 0.5 always the right threshold?

Key idea: the best threshold depends on the cost of each kind of mistake.

By default we predict the class with probability above 0.5. But if the two kinds of error carry different costs (say, a missed fraud is far worse than a false alarm), the optimal rule changes. With λ_{ij} the cost of predicting i when the truth is j, we should predict C_1 when

$$\frac{P(C_1 \mid \mathbf{x})}{P(C_0 \mid \mathbf{x})} > \frac{\lambda_{10} - \lambda_{00}}{\lambda_{01} - \lambda_{11}}.$$

Equal costs recover the familiar 0.5 threshold; unequal costs shift it. This cleanly separates the **model** (which produces probabilities) from the **decision rule** (which turns them into actions).

14. The unifying view: the exponential family

Key idea: for a broad class of distributions, the Naive Bayes posterior is a softmax of a linear function.

Many common distributions — Bernoulli, categorical, Gaussian, Poisson — belong to the **exponential family**, which can be written

$$P(x_j \mid C) = h(x_j)\exp\!\big(\eta_{jC}^{\top} T(x_j) - A(\eta_{jC})\big).$$

Substituting into the log-posterior gives

$$\log P(C \mid \mathbf{x}) = \text{const} + \log P(C) + \sum_j \big[\eta_{jC}^{\top} T(x_j) - A(\eta_{jC})\big],$$

which is **linear in the sufficient statistics** $T(x_j)$. For multiple classes this makes the posterior a **softmax of a linear function.** This single fact unifies the earlier sections: the categorical case (Section 3), the shared-variance Gaussian case (Section 4), and the logistic-regression connection (Section 5) are all instances of the same underlying result.

Summary

Naive Bayes rests on one assumption — features are independent given the class — which makes it fast and data-efficient. Beneath that simplicity sits a rich structure: in log-space it is a linear (or, for Gaussians with

differing variances, quadratic) classifier; it forms a generative counterpart to logistic regression; its smoothing rule is a Bayesian estimate under a Dirichlet prior; and for exponential-family distributions its posterior is a softmax of a linear function. Understanding these connections explains both why such a simple method performs so well and where its limitations — poor probability calibration, sensitivity to correlated features — come from.