Where Deep Learning Losses Come From

Likelihood, cross-entropy, and regularization from probability

Prof. Nipun Batra

ES 667 · Deep Learning · IIT Gandhinagar

Three familiar losses, one probabilistic origin

You have already met these losses:

$$\mathcal{L}_{\text{reg}} = \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2 \quad \text{(regression)}$$

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{n}\sum_i [y_i \log p_i + (1 - y_i)\log(1 - p_i)] \quad \text{(binary classification)}$$

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{data}} + \lambda\|\boldsymbol{\theta}\|_2^2 \quad \text{(regularized)}$$

Objective. Derive each term from an explicit likelihood or prior.

Outline

1. **Predictive distributions** — what the network says about Y given x
2. **Likelihood** — how observed data score candidate parameters
3. **Negative log-likelihood** — why products become trainable sums
4. **Output losses** — Gaussian $\rightarrow$ MSE; Bernoulli/categorical $\rightarrow$ cross-entropy
5. **Priors and MAP** — why regularizers belong in the same objective

Probability primer

Discrete distributions assign probability to outcomes

V VISUAL

The **support** $\mathcal{S}_Y$ is the set of values that Y can take.

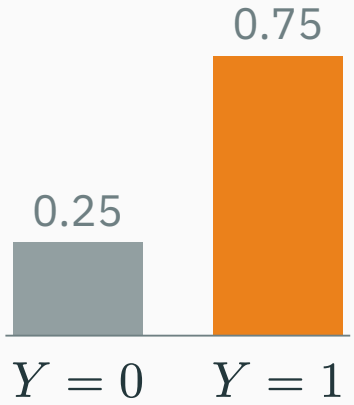
Probability mass function (PMF)

$$p_{Y(y)} = P(Y = y), \quad y \in \mathcal{S}_Y$$

$$p_{Y(y)} \geq 0, \quad \sum_{y \in \mathcal{S}_Y} p_{Y(y)} = 1$$

Examples. Bernoulli: $\{0, 1\}$ · categorical: $\{1, ..., K\}$ · Poisson: $\{0, 1, 2, ...\}$

Bernoulli(0.75) PMF



each bar is an outcome probability

Exact values can have nonzero probability: here $P(Y = 1) = 0.75$.

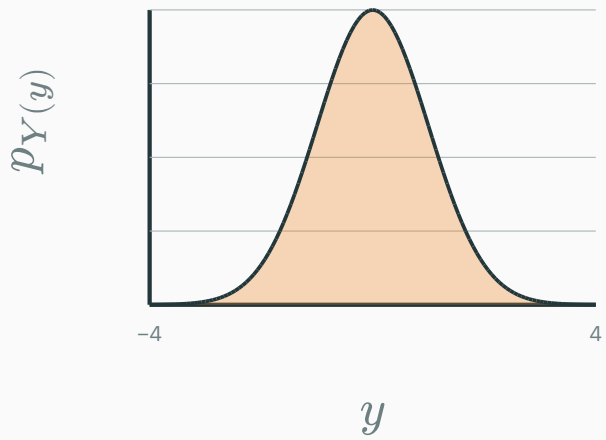
Continuous distributions assign density across a support

V VISUAL

The **support** may be bounded or unbounded:

$\mathcal{S}_Y = [a, b]$

for a Uniform variable; $\mathcal{S}_Y = \mathbb{R}$ for a Gaussian.



Probability density function (PDF)

$$p_{Y(y)} \geq 0, \quad \int_{\mathcal{S}_Y} p_{Y(y)} \, \mathrm{d}y = 1$$

the total area under a PDF is 1

For continuous Y, probability comes from area under the density.

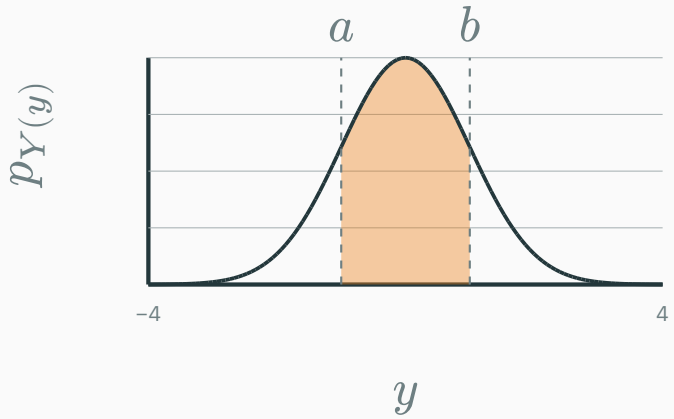
Density is not probability

V VISUAL

For a continuous Y, probability comes from **area**:

$$P(a \leq Y \leq b) = \int_a^b p_{Y(y)} \, \mathrm{d}y$$

$p_{Y(y)} \neq P(Y = y)$ — a density can **exceed 1**; only *integrals* are probabilities.



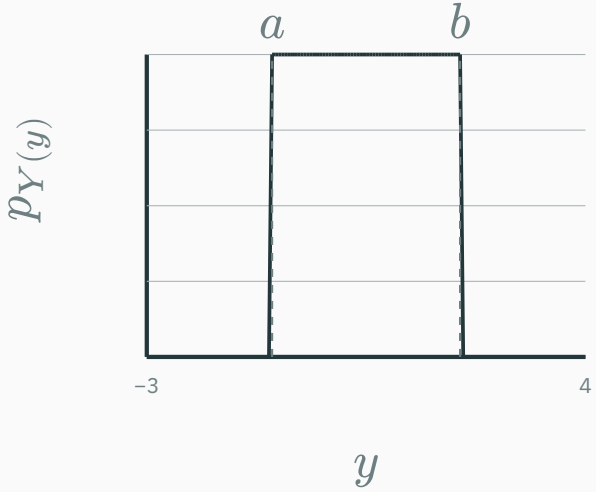
the shaded area is $P(a \leq Y \leq b)$

Uniform distribution: support matters

V VISUAL

$$Y \sim \mathcal{U}(a,b) \quad p_{Y(y)} = \begin{cases} 1/(b-a) & a \le y \le b \\ 0 & \text{otherwise} \end{cases}$$

The support matters: outside $[a,b]$, the density is exactly 0.



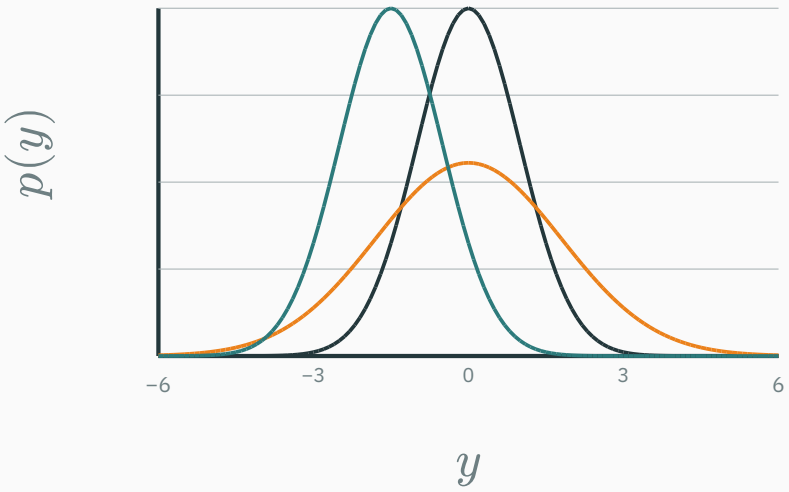
Gaussian distribution

V VISUAL

$$Y \sim \mathcal{N}(\mu, \sigma^2) \qquad p(y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)$$

μ is the mean; $\sigma > 0$ is the standard deviation (scale); σ^2 is the variance.

$\mu = 0, \sigma = 1$ $\mu = 0, \sigma = 1.8$ $\mu = -1.5, \sigma = 1$



Only two knobs: **location** μ and **scale** σ.

From probability to likelihood

Read the model notation one symbol at a time

Q QUESTION

$$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x})$$

$\boldsymbol{X}$ / $\boldsymbol{x}$	$\boldsymbol{X}$ is the random input vector; $\boldsymbol{x}$ is one observed input
Y / y	Y is the random target; y is one possible or observed value
$\boldsymbol{\theta}$	the **parameter vector** — the model's learned weights and biases
$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x})$	the probability (discrete Y) or density (continuous Y) assigned to y, **given** $\boldsymbol{x}$

Read aloud: "model θ's probability or density for y, given x."

A point prediction summarizes a distribution

V VISUAL

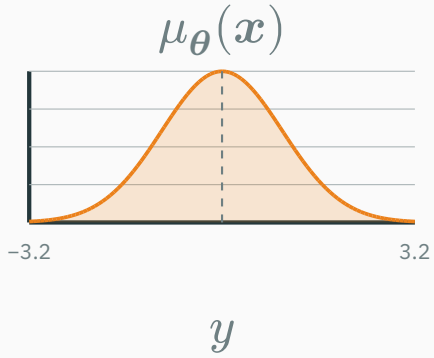
REGRESSION · $Y \in \mathbb{R}$

Point prediction: mean of $Y \mid \boldsymbol{X} = \boldsymbol{x}$

$$\hat{y}(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{\theta}}[Y \mid \boldsymbol{X} = \boldsymbol{x}] = \mu_{\boldsymbol{\theta}}(\boldsymbol{x})$$

Full predictive distribution

$$Y \mid \boldsymbol{X} = \boldsymbol{x} \sim \mathcal{N}\big(\mu_{\boldsymbol{\theta}}(\boldsymbol{x}), \sigma^2_{\boldsymbol{\theta}}(\boldsymbol{x})\big)$$



BINARY CLASSIFICATION · $Y \in \{0, 1\}$

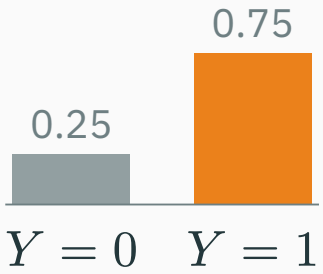
Point prediction: most probable outcome (mode)

$$\hat{y}(\boldsymbol{x}) = \mathbf{1}[p_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq 0.5]$$

Full predictive distribution

$$Y \mid \boldsymbol{X} = \boldsymbol{x} \sim \text{Bernoulli}(p_{\boldsymbol{\theta}}(\boldsymbol{x}))$$

example: $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0.75$



Neural network: $x \to$ distribution parameters. Prediction: $\hat{y}$ = mean or mode.

Prediction fixes the model and varies the outcome

V VISUAL

Example: $Y \in \{0, 1\}$. Fix the input $x^\star$ and one parameter setting θ_A.

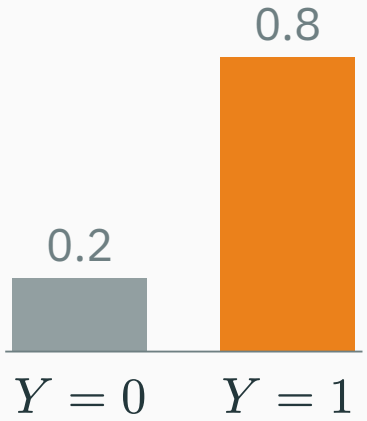
$$p_{\theta_A}(Y = 0 \mid x^\star) = 0.20$$

$$p_{\theta_A}(Y = 1 \mid x^\star) = 0.80$$

$$0.20 + 0.80 = 1$$

We vary the possible value of Y while keeping θ_A and $x^\star$ fixed.

predictive distribution



$$\hat{y}(x^\star) = \arg\max_y p_{\theta_A}(y \mid x^\star) = 1$$

One observed label gives one likelihood score

D DERIVATION

Now observe $y^\star = 1$ for the input $x^\star$, so $\mathcal{D} = \{(x^\star, 1)\}$.

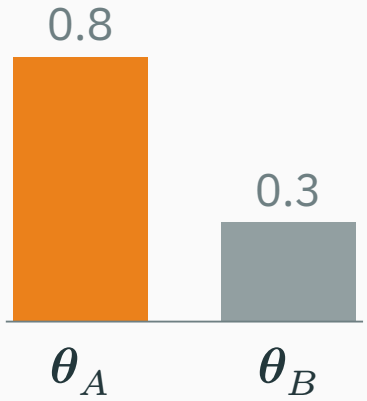
$$L(\theta; \mathcal{D}) = p_\theta(y^\star = 1 \mid x^\star)$$

Candidate A: $L(\theta_A; \mathcal{D}) = 0.80$
Candidate B: $L(\theta_B; \mathcal{D}) = 0.30$

$$L(\theta_A; \mathcal{D}) > L(\theta_B; \mathcal{D})$$

θ_A makes the observed label $y^\star = 1$ less surprising.

likelihood of the fixed observation



Likelihood is a **score as a function of θ**; the scores need not sum to 1 over parameter settings.

One continuous observation gives a likelihood too

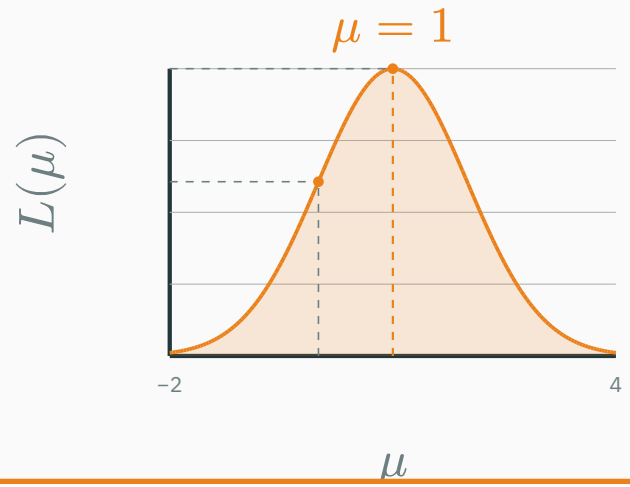
D DERIVATION

Suppose $Y \mid \mu \sim \mathcal{N}(\mu, 1)$ and we observe the fixed value $y^\star = 1$.

$$L(\mu; y^\star = 1) = \left[p_{Y(y \mid \mu)}\right]_{y=1} = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(1-\mu)^2}{2}\right).$$

Candidate $\mu = 1$: $L(1) \approx 0.399$
Candidate $\mu = 0$: $L(0) \approx 0.242$

$\mu = 1$ centres the Gaussian at $y^\star = 1$ and gives the largest **density**.



For continuous Y, this is a density value—not $P(Y = 1)$. Fix the observation, then compare parameter values.

A dataset needs a joint likelihood

Q QUESTION

For two observations, the likelihood starts as a **joint** probability or density:

$$L(\boldsymbol{\theta}; \mathcal{D}) = p_{\boldsymbol{\theta}}(y_1, y_2 \mid \boldsymbol{x}_1, \boldsymbol{x}_2).$$

We may write a product only after making an assumption about how the observations relate.

Question: when is $p(y_1, y_2 \mid \boldsymbol{\theta}) = p(y_1 \mid \boldsymbol{\theta}) p(y_2 \mid \boldsymbol{\theta})$ valid?

The product is a consequence of **conditional independence**; it is not automatic.

Independent means “no extra information”

D DERIVATION

After fixing θ, observing one outcome does not change the distribution of another:

$$p(y_2 \mid y_1, \boldsymbol{\theta}) = p(y_2 \mid \boldsymbol{\theta}).$$

The chain rule then becomes

$$p(y_1, y_2 \mid \boldsymbol{\theta}) = p(y_1 \mid \boldsymbol{\theta}) p(y_2 \mid y_1, \boldsymbol{\theta}) = p(y_1 \mid \boldsymbol{\theta}) p(y_2 \mid \boldsymbol{\theta}).$$

Coin-flip example. If $P(H \mid \theta) = \theta$ and the flips are independent, then $P(H, H \mid \theta) = \theta^2$.

Independence is what licenses multiplication.

Identically distributed means “the same sampling rule”

D DERIVATION

Coin flips

$Y_i \mid \theta \sim \text{Bernoulli}(\theta)$ for every i.
Each flip uses the same bias θ.

Supervised learning

The pairs $(\boldsymbol{X}_i, Y_i)$ come from the same population, and one shared $\boldsymbol{\theta}$ defines $p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i)$.

“Independent” and “identically distributed” are two separate claims; one can hold without the other.

i.i.d. = independent observations + one common data-generating rule.

Independence and identical distribution can fail separately

D DERIVATION

Not independent: daily weather

Tomorrow's weather depends on today's. A first-order Markov model writes

$$p(y_{1:T} \mid \theta) = p(y_1 \mid \theta) \prod_{t=2}^{T} p(y_t \mid y_{t-1}, \theta).$$

The factors are **conditional on the previous day**, not independent marginals.

Independent but not identical: two sensors

Two sensors measure the same temperature μ independently, but have different noise levels:

$$Y_i \mid \mu \sim \mathcal{N}(\mu, \sigma_i^2), \quad \sigma_1 \neq \sigma_2.$$

The likelihood is a product, but its factors use different σ_i.

i.i.d. is a useful special case—not a universal law.

Worked example: dependence changes the product

D DERIVATION

Let R_1 mean “rain today” and R_2 mean “rain tomorrow”. Suppose

$$P(R_1) = 0.4, \quad P(R_2|R_1) = 0.8, \quad P(R_2|\neg R_1) = 0.2.$$

1. Use the conditional rule

$$P(R_1, R_2) = P(R_1)P(R_2|R_1)$$
$$= 0.4 \times 0.8 = 0.32.$$

Both days are rainy with probability 0.32.

2. Test independence

$$P(R_2) = 0.4 \times 0.8 + 0.6 \times 0.2 = 0.44.$$

If the days were independent,

$$P(R_1)P(R_2) = 0.4 \times 0.44 = 0.176,$$

not 0.32.

Dependence does not forbid multiplication: use a conditional probability.

Worked example: independent does not mean identical

D DERIVATION

Given smoke S, two alarms act independently, but their sensitivities differ:

$$P(A_1 = 1|S) = 0.9, \quad P(A_2 = 1|S) = 0.6.$$

Independence

Both alarms ring with probability

$$P(A_1 = 1, A_2 = 1|S) = 0.9 \times 0.6 = 0.54.$$

The product is valid.

Not identically distributed

Alarm 1 detects smoke 90% of the time; Alarm 2 detects it 60% of the time.

$$0.9 \neq 0.6$$

We cannot reuse one common factor for both alarms.

Independent says “multiply”; identical says “reuse the same sampling rule”.

IID turns the joint likelihood into per-example factors

D DERIVATION

For i.i.d. supervised examples, treat the observed inputs as fixed. Conditional independence gives

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{n} p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i).$$

The **factorization** comes from independence; the shared factor form comes from the common model $p_{\boldsymbol{\theta}}$.

Next we compute this product, then introduce a transform that makes it easier to optimize.

This is the bridge from one-example likelihood to a dataset loss.

Work two tiny Bernoulli datasets

D DERIVATION

Compare a fair coin $\theta = 0.5$ with a head-biased coin $\theta = 0.8$.

Observed data	$L(0.5)$	$L(0.8)$	Which explains it better?
H, T	$0.5(0.5) = 0.25$	$0.8(0.2) = 0.16$	fair coin
H, H	$0.5^2 = 0.25$	$0.8^2 = 0.64$	head-biased coin

Same candidate models, different fixed data. Likelihood ranks candidates by how well they explain the observations actually seen.

Multiply one Bernoulli term per independent flip.

Two Gaussian measurements: multiply densities

D DERIVATION

Let $Y_i \mid \mu \sim \mathcal{N}(\mu, 1)$ independently, and observe $y_1 = 1$, $y_2 = 2$.

Candidate $\mu = 1.5$

$p(y_1|\mu) \approx 0.352$

$p(y_2|\mu) \approx 0.352$

$L(1.5) \approx 0.352 \times 0.352 = 0.124$

Candidate $\mu = 0$

$p(y_1|\mu) \approx 0.242$

$p(y_2|\mu) \approx 0.054$

$L(0) \approx 0.242 \times 0.054 = 0.013$

For continuous data we multiply densities; $\mu = 1.5$ gives the larger likelihood.

Maximum likelihood estimation

$$\hat{\boldsymbol{\theta}}_{\text{MLE}} = \arg\max_{\boldsymbol{\theta}} p(\mathcal{D} \mid \boldsymbol{\theta})$$

Maximum likelihood estimation (MLE). Choose the parameter values that assign the highest likelihood to the observed dataset.

Companion notebook: distributions, MLE, and MAP in PyTorch

I INTERACTIVE

INTERACTIVE

↗ colab.research.google.com/github/nipunbatra/dl-teaching/blob/master/notebooks/L01/00_likelihood_iid_worked_examples.ipynb

Explore `torch.distributions`, `log_prob`, sampling, i.i.d. factorization, coin-toss MLE, Gaussian regression, Bernoulli classification, and MAP with Gaussian, Laplace, and Student-t priors.

INTERACTIVE

↗ colab.research.google.com/github/nipunbatra/dl-teaching/blob/master/notebooks/L01/03_robust_linear_regression.ipynb

Short companion: fit the same line with Gaussian/MSE, Laplace/MAE, and Student-t losses; then see which observations control each estimate.

Coin flips: a likelihood you can compute

D DERIVATION

Observed: $H, H, T, T, T, H, H, T, T, T$
Hence $n_H = 4$ and $n_T = 6$.

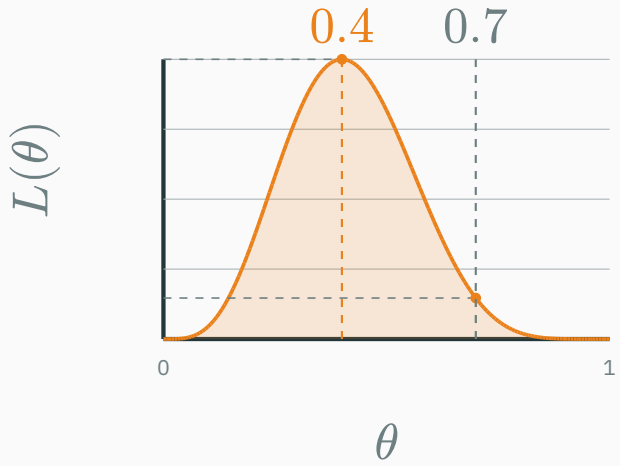
Model: $p(H \mid \theta) = \theta$, $p(T \mid \theta) = 1 - \theta$.

Because the flips are i.i.d.,

$$L(\theta) = \prod_i p(x_i \mid \theta)$$

$$= \theta^{n_H}(1-\theta)^{n_T}$$

$$= \theta^4(1-\theta)^6$$



$$L(0.4) \approx 0.00119$$
$$L(0.7) \approx 0.000175$$

about $6.8\times$ larger at $\theta = 0.4$

The likelihood peaks at $\theta = 0.4$ — the observed head fraction $\frac{4}{10}$.

Coin flips: calculate the log-likelihood

D DERIVATION

The **log-likelihood** is the logarithm of the likelihood — not a different model:

$$\ell(\theta) := \log L(\theta) = \log\left[\theta^4(1-\theta)^6\right] = 4\log\theta + 6\log(1-\theta)$$

Candidate $\theta = 0.4$

$L(0.4) \approx 0.00119$

$\ell(0.4) = 4\log 0.4 + 6\log 0.6 \approx -6.73$

Candidate $\theta = 0.7$

$L(0.7) \approx 0.000175$

$\ell(0.7) = 4\log 0.7 + 6\log 0.3 \approx -8.65$

Likelihoods below 1 have negative logs; among them, -6.73 is the larger value.

The observed sequence has higher log-likelihood at $\theta = 0.4$.

Why logs? The maximizing parameter does not change

D DERIVATION

The logarithm is **strictly increasing** on positive numbers:

$$a > b > 0 \quad \Longrightarrow \quad \log a > \log b$$

Therefore it preserves the ordering of all positive likelihood values:

$$L(0.4) > L(0.7)$$

apply the increasing function $\log$

$$\log L(0.4) > \log L(0.7)$$

$$\arg\max_{\theta} L(\theta) = \arg\max_{\theta} \log L(\theta)$$

Both likelihood and log-likelihood rank $\theta = 0.4$ above $\theta = 0.7$.

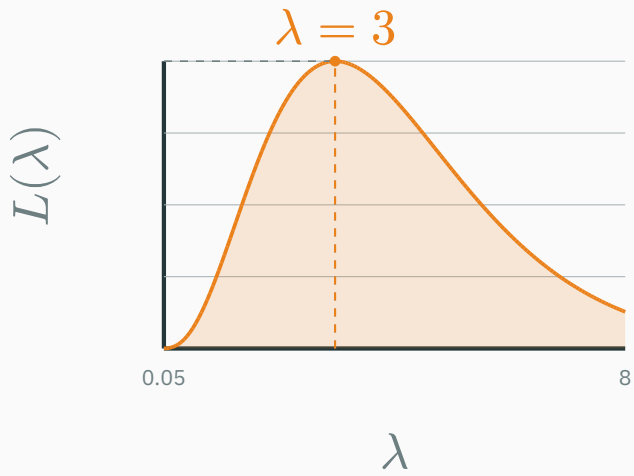
Poisson example: logging preserves the peak

V VISUAL

Suppose $Y \sim \text{Poisson}(\lambda)$ and the observed count is $y = 3$.

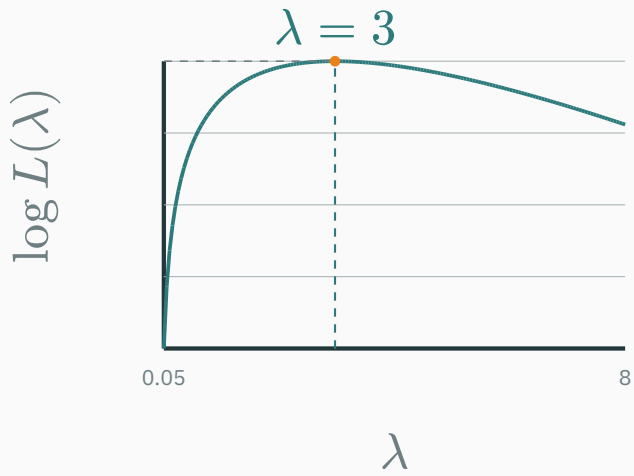
LIKELIHOOD

$$L(\lambda) = e^{-\lambda}\frac{\lambda^3}{3!}$$



LOG-LIKELIHOOD

$$\log L(\lambda) = 3\log\lambda - \lambda - \log 3!$$



Both curves peak at $\hat{\lambda}_{\text{MLE}} = 3$: the log changes the vertical scale, not the maximizing parameter.

Why logs? Products become stable sums

D DERIVATION

For independent observations,

$$L(\boldsymbol{\theta}) = \prod_i p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) \quad \Longrightarrow \quad \log L(\boldsymbol{\theta}) = \sum_i \log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i)$$

Worked numerical example. Suppose 200 observed outcomes each receive probability 0.01.

Direct product	$L = 0.01^{200} = 10^{-400} < 4.94 \times 10^{-324}$ → underflows to 0
Log space	$\log L = 200 \log(0.01) \approx -921.03$ → finite and representable

Float64 range: $\pm 1.80 \times 10^{308}$ · smallest positive normal 2.23×10^{-308} · smallest subnormal 4.94×10^{-324}.

Logs make optimization easier and prevent numerical underflow.

Coin flips: solve for the stationary point

D DERIVATION

For $L(\theta) = \theta^{n_H}(1-\theta)^{n_T}$, the log-likelihood is

$$\log L(\theta) = n_H \log \theta + n_T \log(1-\theta)$$

Set the derivative to zero:

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \log L(\theta) = \frac{n_H}{\theta} - \frac{n_T}{1-\theta} = 0$$

$$\Longrightarrow \theta^\star = \frac{n_H}{n_H + n_T} = \frac{4}{10} = 0.4$$

$\theta^\star = 0.4$ **is a candidate optimum. Next, check the curvature.**

The second derivative confirms a maximum

D DERIVATION

Differentiate the log-likelihood again:

$$\frac{\mathrm{d}^2}{\mathrm{d}\theta^2} \log L(\theta) = -\frac{n_H}{\theta^2} - \frac{n_T}{(1-\theta)^2}$$

For $n_H, n_T > 0$ and $0 < \theta < 1$, both terms are negative:

$$\frac{\mathrm{d}^2}{\mathrm{d}\theta^2} \log L(\theta) < 0$$

At the stationary point $\theta^\star = 0.4$,

$$-\frac{4}{(\quad)^2} - \frac{6}{(\quad)^2} = -25 - 16.67 \approx -41.67 < 0$$

The second derivative is negative everywhere, so the curve bends downward (is strictly concave). Its one stationary point is the unique global maximum: $\hat{\theta}_{\mathrm{MLE}} = 0.4$.

Maximizing likelihood = minimizing NLL

D DERIVATION

$$\arg\max_{\boldsymbol{\theta}} \prod_i p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) = \arg\max_{\boldsymbol{\theta}} \sum_i \log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i)$$

$$= \arg\min_{\boldsymbol{\theta}} \left(-\sum_i \log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) \right)$$

$$\ell_{i(\boldsymbol{\theta})} = -\log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i)$$

Empirical risk = average training loss

D DERIVATION

In ML, **risk** means expected loss. Because the data-generating distribution is unknown, we estimate it from the observed training set.

Empirical risk = average loss on $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$.

Per example	$\ell_{i(\boldsymbol{\theta})} = -\log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i)$
Training set	$\hat{R}(\boldsymbol{\theta}) = \frac{1}{n}\sum_i \ell_{i(\boldsymbol{\theta})} = -\frac{1}{n}\log p(\mathcal{D} \mid \boldsymbol{\theta})$

The final equality uses conditional independence: $p(\mathcal{D}|\boldsymbol{\theta}) = \prod_i p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)$.

Minimizing empirical NLL is maximum likelihood; $\frac{1}{n}$ only rescales the objective.

Worked example: support becomes a hard loss

D DERIVATION

Recall the Uniform observation model:

$$p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) = \begin{cases} 1/(b-a) & a \le y_i \le b \\ 0 & \text{otherwise} \end{cases}$$

If an observed target lies outside its support, then

$$p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) = 0$$

Its contribution to the negative log-likelihood is

$$\ell_{i(\boldsymbol{\theta})} = -\log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) = -\log 0 = +\infty$$

Zero likelihood becomes an infinite training penalty.

Checkpoint: uniform support

Q QUESTION

For a Uniform(a, b) observation model, what is the NLL if the target lies outside $[a, b]$?

A. 0

B. A fixed finite penalty

C. $-\log 0 = +\infty$

D. It depends only on σ

Answer: uniform support

A ANSWER

Correct: C — $-\log 0 = +\infty$

Why: The model assigns zero density outside its support, so the observation is impossible under the assumption.

Regression: why MSE?

Regression as signal plus noise

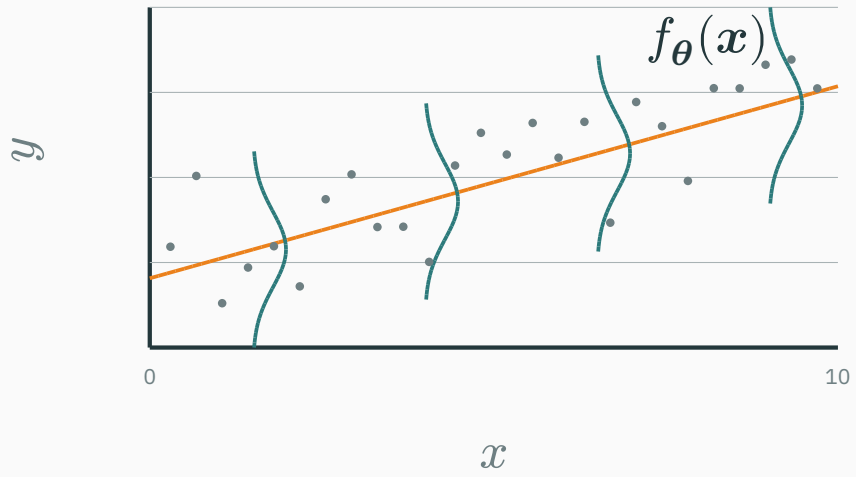
$$y_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

Therefore the conditional output distribution is

$$Y_i \mid \boldsymbol{X}_i = \boldsymbol{x}_i \sim \mathcal{N}(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \sigma^2)$$

What the assumption means visually

V VISUAL



Gaussian noise lives in the **output direction**, conditional on x — a vertical bell at every input.

Gaussian likelihood

D DERIVATION

$$p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right)^2}{2\sigma^2} \right)$$

Negative log of a single example:

$$-\log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i) = \frac{\left(y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right)^2}{2\sigma^2} + C$$

MSE is Gaussian MLE

Sum over the dataset:

$$-\log p(\mathcal{D} \mid \boldsymbol{\theta}) = \frac{1}{2\sigma^2} \sum_i \left(y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right)^2 + C$$

With σ fixed, the minimizer is

$$\hat{\boldsymbol{\theta}}_{\text{MLE}} = \arg\min_{\boldsymbol{\theta}} \sum_i \left(y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right)^2$$

MSE = Gaussian NLL, up to constants.

Linear regression makes f_θ explicit

D DERIVATION

For the one-dimensional picture,

$$\hat{y}_i = \theta_0 + \theta_1 x_i$$

For d features, prepend a constant 1 so the intercept is part of the same parameter vector:

$$\tilde{\boldsymbol{x}}_i = (1, \boldsymbol{x}_i), \quad \boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_d)$$

Then the same notation used throughout the lecture becomes

$$\hat{y}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \boldsymbol{\theta}^\top \tilde{\boldsymbol{x}}_i$$

θ_0 is the intercept; $\theta_1, \ldots, \theta_d$ are the slopes.

Gaussian MLE becomes ordinary least squares

D DERIVATION

Substitute the linear mean into the observation model:

$$Y_i \mid \boldsymbol{X}_i = \boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\theta}^\top \tilde{\boldsymbol{x}}_i, \sigma^2)$$

The vertical residual of point i is

$$r_i = y_i - \hat{y}_i = y_i - \boldsymbol{\theta}^\top \tilde{\boldsymbol{x}}_i$$

With σ fixed, maximum likelihood therefore gives

$$\hat{\boldsymbol{\theta}}_{\mathrm{MLE}} = \arg\min_{\boldsymbol{\theta}} \sum_i \underbrace{r_i^2}_{\text{squared vertical error}}$$

Ordinary least squares is Gaussian maximum likelihood for a linear mean.

The normal equations solve linear regression

D DERIVATION

Stack the augmented inputs into $\tilde{\boldsymbol{X}} = [\mathbf{1} \quad \boldsymbol{X}]$. Then

$$\hat{\boldsymbol{y}} = \tilde{\boldsymbol{X}}\boldsymbol{\theta}, \quad J(\boldsymbol{\theta}) = \left\|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{\theta}\right\|_2^2$$

The gradient $\nabla_{\boldsymbol{\theta}} J$ collects one partial derivative per parameter. Set it to zero:

$$\nabla_{\boldsymbol{\theta}} J = 2\tilde{\boldsymbol{X}}^\top\left(\tilde{\boldsymbol{X}}\boldsymbol{\theta} - \boldsymbol{y}\right) = \mathbf{0}$$

This gives the **normal equations**:

$$\tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}}\hat{\boldsymbol{\theta}} = \tilde{\boldsymbol{X}}^\top \boldsymbol{y}$$

If $\tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}}$ is invertible,

$$\hat{\boldsymbol{\theta}} = \left(\tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}}\right)^{-1} \tilde{\boldsymbol{X}}^\top \boldsymbol{y}$$

Linear regression has a closed form; neural networks keep the MSE objective but use iterative optimization.

Why earlier ML courses could ignore σ^2

Q QUESTION

σ^2 is the observation-noise variance: the expected squared spread of targets around the mean prediction.

For n Gaussian observations, write $r_i = y_i - \mu_i$. The dataset NLL is

$$\mathcal{L}(\boldsymbol{\theta}; \sigma^2) = \frac{1}{2\sigma^2} \sum_i r_i^2 + \frac{n}{2} \log \sigma^2 + C.$$

If one global σ^2 is fixed, then $\frac{n}{2} \log \sigma^2$ is constant and $\frac{1}{2\sigma^2}$ is only a positive scale factor:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \sigma^2) = \arg\min_{\boldsymbol{\theta}} \sum_i r_i^2.$$

For point predictions, fixed-variance Gaussian MLE gives the same fitted model as MSE.

How is σ^2 chosen in practice?

D DERIVATION

One global noise level.

Fit the mean model, then estimate the residual spread:

$$\hat{\sigma}^2_{\text{MLE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\mu}_i)^2.$$

Use this when a residual plot has roughly the same vertical width across inputs.

Input-dependent noise.

Let the network predict both

$$(\mu_i, s_i) = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i),$$

$$s_i = \log \sigma_i^2, \quad \sigma_i^2 = e^{s_i} > 0.$$

Use this when residual spread changes with the input—for example, a funnel shape.

Fit both cases with Gaussian NLL; validate interval coverage: the fraction of targets contained by intervals claiming a stated probability.

Homoscedastic vs input-dependent uncertainty

V VISUAL

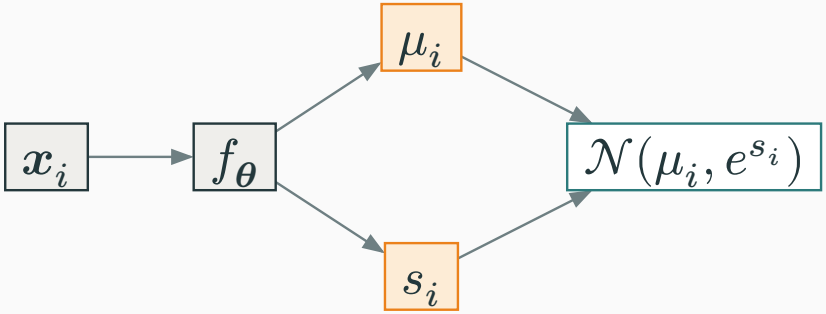
HOMOSCEDASTIC

Network output: μ_i

Shared variance: σ^2

same uncertainty at every x_i

HETEROSCEDASTIC



Network outputs: μ_i and s_i

$$s_i = \log \sigma_i^2, \quad \sigma_i^2 = e^{s_i} > 0$$

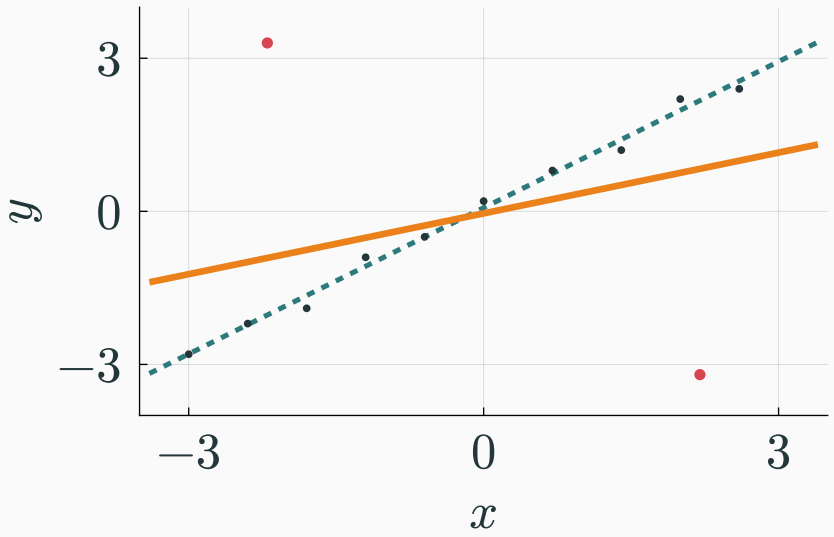
uncertainty changes with x_i

Predict log-variance so every real s_i maps to a positive variance e^{s_i}.

A few unusual observations can dominate a fitted line

Q QUESTION

SYNTHETIC · COMPUTED · exact 12-point teaching case; reproduced in the executed robust-notebook appendix



fit to the ten typical points MSE fit after adding two outliers

Ten typical observations lie near one linear trend.

Two unusual observations have large vertical errors.

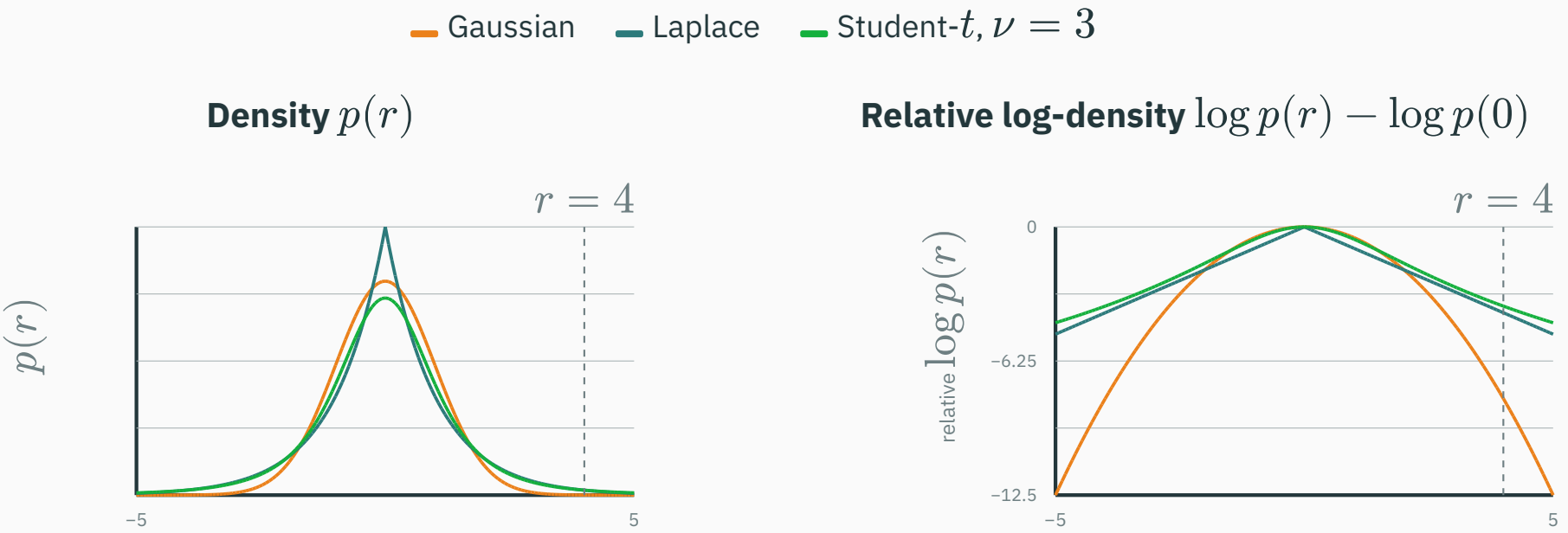
The squared-error fit becomes much flatter. How can two points change the fitted line so strongly?

Robust regression asks for a noise model that allows occasional large residuals without letting them dominate the fit.

Tails are easier to compare on a log scale

V VISUAL

For residual $r = y - \hat{y}$, compare unit-scale Gaussian, Laplace, and Student-t; $\nu = 3$ is its degrees of freedom.



Smaller ν means heavier Student-t tails. Heavy tails make an occasional large residual less surprising.

A large residual is less surprising under heavy tails

D DERIVATION

Evaluate the three densities at the same residual, $r = 4$, using unit scale parameters.

Gaussian

density ≈ 0.000134

The model finds this residual extremely surprising.

Laplace

density ≈ 0.00916

More tail density makes the same residual less surprising.

Student-t, $\nu = 3$

density ≈ 0.00916

Heavy tails explicitly allow occasional extreme errors.

Laplace and Student-t assign about $68\times$ the Gaussian density here.

The observation is not discarded; the assumed noise process simply says that such an error can occur.

One observation contributes one gradient signal

D DERIVATION

For observation i, the model predicts $\hat{y}_i$ and leaves residual $r_i = y_i - \hat{y}_i$.

Loss value ℓ_i

says how incompatible this observation is with the current prediction.

Gradient magnitude $|\ell_{i'}(r_i)|$

says how strongly this observation scales the next parameter update.

$$\nabla_{\boldsymbol{\theta}} \ell_i = -\ell_{i'}(r_i) \nabla_{\boldsymbol{\theta}} \hat{y}_i$$

The sign gives the correction direction; $|\ell_{i'}(r_i)|$ gives the size of this observation's gradient factor.

Residual density → NLL → parameter gradient

D DERIVATION

For one observation, $\hat{y}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ and $r_i = y_i - \hat{y}_i$.

$$\ell(r) = -\log p_{R(r)} \quad \rightarrow \quad \psi(r) = \mathrm{d}\frac{\ell}{\mathrm{d}r} = -\frac{1}{p_{R(r)}}\,\mathrm{d}\frac{p_{R(r)}}{\mathrm{d}r}$$

$$\boxed{\nabla_{\boldsymbol{\theta}} \ell_i = \psi(r_i)\nabla_{\boldsymbol{\theta}} r_i = -\psi(r_i)\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_i).}$$

chain rule: residual-loss derivative × residual-to-model derivative

Linear example: $f_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x \quad \rightarrow \quad \nabla_{\boldsymbol{\theta}} \ell_i = -\psi(r_i)(1, x_i)^\top.$

The density chooses $\psi(r)$; the model supplies $\nabla_\theta f$. Their product is the observation's parameter gradient.

Gaussian and Laplace residuals: the full chain

D DERIVATION

Noise	Residual density $p_{R(r)}$	NLL $\ell(r)$	$\psi(r) = \ell'(r)$
Gaussian	$\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{r^2}{2\sigma^2}}$	$\frac{r^2}{2\sigma^2} + C$	$\frac{r}{\sigma^2}$
Laplace	$\frac{1}{2s_L} e^{-\frac{\lvert r\rvert}{s_L}}$	$\frac{\lvert r\rvert}{s_L} + C$	$\frac{\operatorname{sign}(r)}{s_L}$

$$\nabla_{\boldsymbol{\theta}} \ell_i = -\psi(r_i) \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

At $r = 0$, $|r|$ has no single derivative; optimization uses any slope in $\left[-\frac{1}{s_L}, \frac{1}{s_L}\right]$, called a **subgradient**. Away from zero, the gradient magnitude is $\frac{1}{s_L}$.

Gaussian influence grows with $|r|$; Laplace influence has constant magnitude.

Student-t residual: the full chain

D DERIVATION

For degrees of freedom ν and scale s:

ν controls tail heaviness; $s > 0$ sets residual scale; $c_{\nu,s} > 0$ normalizes the density to integrate to 1.

density $p_{R(r)} = c_{\nu,s}\left(1 + \frac{r^2}{\nu s^2}\right)^{-\frac{\nu+1}{2}}$

NLL $\ell(r) = \frac{\nu+1}{2}\log\left(1 + \frac{r^2}{\nu s^2}\right) + C$

r derivative $\psi(r) = \mathrm{d}\frac{\ell}{\mathrm{d}r} = (\nu+1)\frac{r}{\nu s^2 + r^2}$

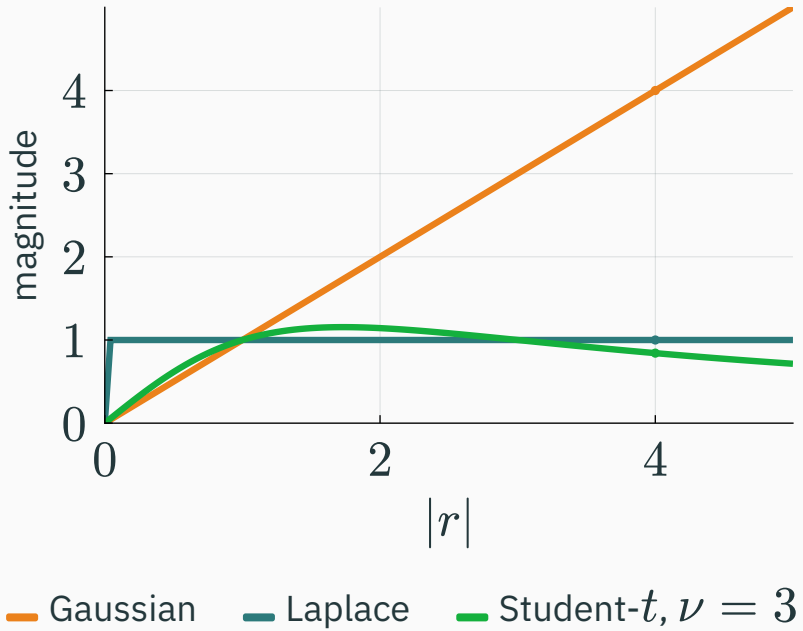
θ gradient $\nabla_{\boldsymbol{\theta}}\ell_i = -(\nu+1)\frac{r_i}{\nu s^2 + r_i^2}\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$

Near zero, $\psi(r) \approx \left(\frac{\nu+1}{\nu s^2}\right)r$; for $|r|$ very large, $\psi(r) \approx \frac{\nu+1}{r} \to 0$.

Student-t treats small residuals approximately quadratically, but an extreme residual's gradient contribution eventually decreases.

Large residuals create different gradient magnitudes

V VISUAL



At $r = 4$, the gradient factor is:

Gaussian

$$|\ell'(4)| = 4$$

Laplace

$$|\ell'(4)| = 1$$

Student-t

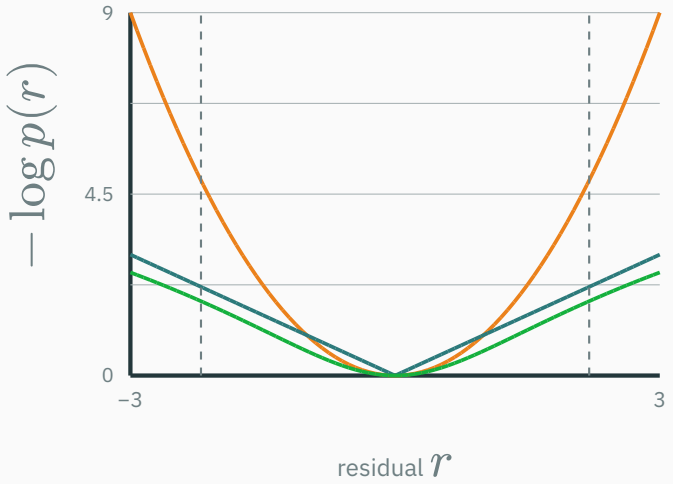
$$|\ell'(4)| = \tfrac{16}{19} \approx 0.84$$

Robust fitting keeps every observation, while limiting the gradient contribution of an extreme residual.

The noise model determines how errors are penalized

V VISUAL

Noise model	Negative log-likelihood
Gaussian	r^2
Laplace	$\lvert r \rvert$
Uniform	0 inside; $+\infty$ outside
Student-t	$\log\left(1 + \frac{r^2}{\nu}\right)$



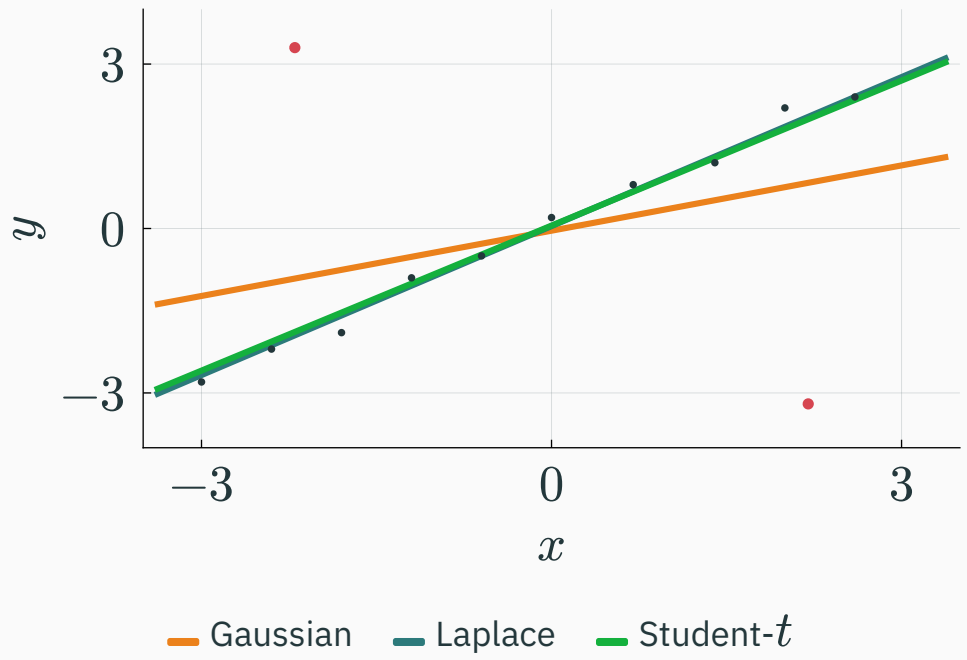
$r = y - \hat{y}$ · swatches match plot

At $r = 0$, prediction equals target. Farther horizontally = larger error; higher vertically = larger NLL penalty.

Gaussian penalizes large errors most; Student-t is more robust. The dashed Uniform boundaries mark an infinite penalty outside the allowed interval.

Fit the same data under three noise models

V VISUAL



Learned $\hat{y} = \theta_0 + \theta_1 x$

model	fitted (θ_0, θ_1)
Gaussian	$(-0.040, 0.397)$
Laplace	$(0.044, 0.906)$
Student-t	$(0.053, 0.882)$

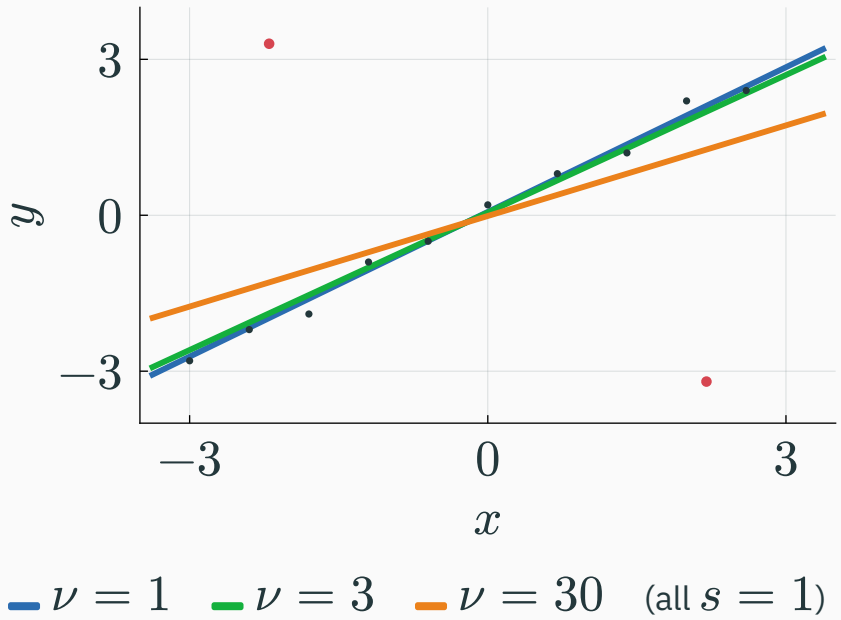
$$\sigma = 1 \cdot s_L = 1 \cdot (\nu, s) = (3, 1)$$

All twelve observations are retained. The models differ only in the residual likelihood optimized.

The Gaussian fit is pulled flat by the two red points; Laplace and Student-t stay close to the trend supported by the ten typical points.

Student-t hyperparameters control robustness

V VISUAL



Learned coefficients

ν	s	fitted (θ_0, θ_1)
1	1	$(0.064, 0.928)$
3	0.5	$(0.065, 0.934)$
3	1	$(0.053, 0.882)$
3	1.5	$(0.036, 0.807)$
30	1	$(-0.014, 0.581)$

Smaller ν gives heavier tails. Larger s delays the transition from quadratic to tail-robust behaviour.

For unregularized Gaussian and Laplace fits, changing the fixed scale only rescales the objective and leaves (θ_0, θ_1) unchanged; Student-t scale changes relative residual weights.

ν and s are modelling choices: cross-validation or domain knowledge should choose how quickly an observation is treated as a tail event.

Interactive: likelihood → loss

I INTERACTIVE

INTERACTIVE

↗ nipunbatra.github.io/interactive-articles/likelihood-to-loss

likelihood-to-loss.html — three synchronized panels: the **noise density** $p(r)$, the **loss** $-\log p(r)$, and **data with a fitted line**.

Controls: Gaussian / Laplace / Uniform / Student-t · noise scale · slope & intercept · **add an outlier** · per-point contributions.

Ask: which model reacts most to an outlier? why does Uniform give an infinite barrier? which loss is robust?

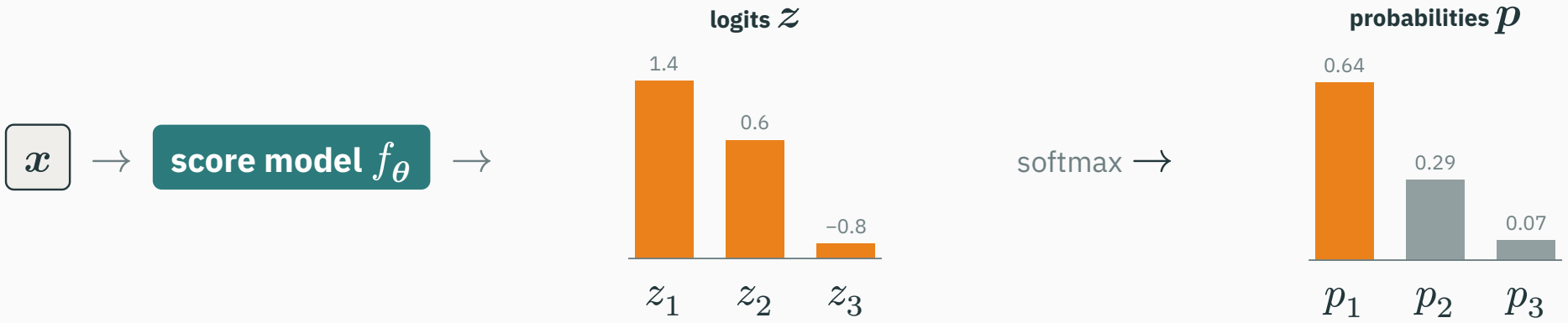
Classification: cross-entropy

Classification should output probabilities

For K classes:

$$p_{\boldsymbol{\theta}}(Y = k \mid \boldsymbol{x}) \geq 0, \qquad \sum_{k=1}^{K} p_{\boldsymbol{\theta}}(Y = k \mid \boldsymbol{x}) = 1$$

A network first produces unconstrained class scores, called **logits**; sigmoid or softmax converts them to probabilities. We derive both below.



$f_{\boldsymbol{\theta}}$ may be linear or nonlinear; the probability constraints come from the output transformation.

Binary classification: Bernoulli model

$$Y \mid \boldsymbol{X} = \boldsymbol{x} \sim \text{Bernoulli}(p_{\boldsymbol{\theta}}(\boldsymbol{x}))$$

Compact one-line form: Write $p = p_{\boldsymbol{\theta}}(\boldsymbol{x})$. Then

$$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) = p^y (1-p)^{1-y}$$

$y = 1 \Rightarrow p(y) = p$

$y = 0 \Rightarrow p(y) = 1 - p$

Bernoulli NLL gives BCE

D DERIVATION

$$-\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) = -\log\big(p^{y}(1-p)^{1-y}\big)$$

$$= -y\log p - (1-y)\log(1-p)$$

Binary cross-entropy = Bernoulli NLL.

Check. True label $y = 1$: if $p = 0.8$ the loss is $-\log 0.8 \approx 0.22$; if $p = 0.2$ it is $-\log 0.2 \approx 1.61$.

Why BCE matches a binary label better than MSE

For one observed example, let the target be $y = 1$ and the model predict

$$p = P(Y = 1|\boldsymbol{x}) = 0.8.$$

Squared error

$$\ell_{\text{MSE}} = (y - p)^2 = (1 - 0.8)^2 = 0.04.$$

Its usual likelihood interpretation is Gaussian:

$$Y|\boldsymbol{x} \sim \mathcal{N}(0.8, \sigma^2).$$

Permits all real Y—not just labels.

Binary cross-entropy

$$\ell_{\text{BCE}} = -\log p = -\log 0.8 \approx 0.223.$$

Its likelihood is Bernoulli:

$$Y|\boldsymbol{x} \sim \text{Bernoulli}(0.8).$$

Permits only $Y = 0$ or $Y = 1$.

MSE is possible; BCE is the negative log-likelihood matched to a binary observation.

Why use logits?

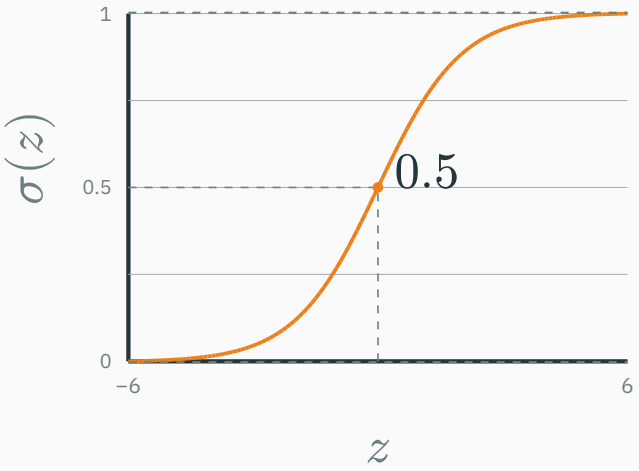
V VISUAL

The network emits an unbounded score $z = f_{\theta}(x) \in \mathbb{R}$; the **sigmoid** maps it to a probability:

$$p = \sigma(z) = \frac{1}{1 + e^{-z}}$$

$z \to -\infty : p \to 0$
$z = 0 : p = 0.5$
$z \to +\infty : p \to 1$

In practice sigmoid + BCE are fused for numerical stability.



Logistic regression = Bernoulli MLE

D DERIVATION

Use an augmented input $\tilde{\boldsymbol{x}}_i$ so $\boldsymbol{\theta}$ contains both weights and the intercept:

$$p_i = \sigma(\boldsymbol{\theta}^\top \tilde{\boldsymbol{x}}_i), \quad Y_i \mid \boldsymbol{X}_i = \boldsymbol{x}_i \sim \text{Bernoulli}(p_i)$$

. Likelihood of the whole labelled dataset (i.i.d.):

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} p_i^{y_i} (1 - p_i)^{1 - y_i}$$

Its negative log — the objective we minimise:

$$J(\boldsymbol{\theta}) = -\sum_{i=1}^{n} [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$

Logistic regression is Bernoulli maximum likelihood; its training objective is **binary cross-entropy**.

Multi-class model and softmax

V VISUAL

CATEGORICAL OBSERVATION

$$Y \mid \boldsymbol{X} = \boldsymbol{x} \sim \text{Categorical}(p_1, \ldots, p_K)$$

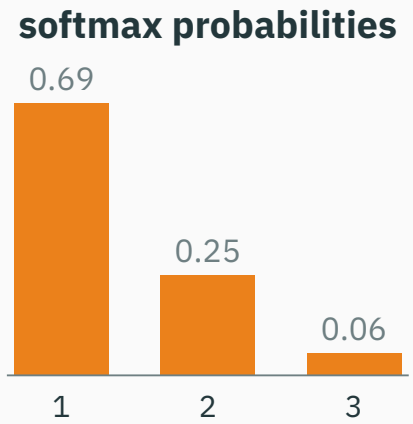
A **one-hot** target has one 1 at the true class and 0 everywhere else.

Example: class 2 of 3 $\rightarrow \boldsymbol{y} = (0, 1, 0)$

$$p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{k=1}^{K} p_k^{y_k}$$

NETWORK PARAMETERIZATION

$$\boldsymbol{z} = f_{\boldsymbol{\theta}}(\boldsymbol{x}), \quad p_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$$



Softmax guarantees $p_k > 0$ and $\sum_k p_k = 1$, as a categorical model requires.

Categorical NLL gives cross-entropy

D DERIVATION

$$-\log p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x}) = -\log \prod_k p_k^{y_k} = -\sum_k y_k \log p_k$$

Because $\boldsymbol{y}$ is one-hot, only the true class survives:

$$\mathcal{L} = -\log p_{\text{true class}}$$

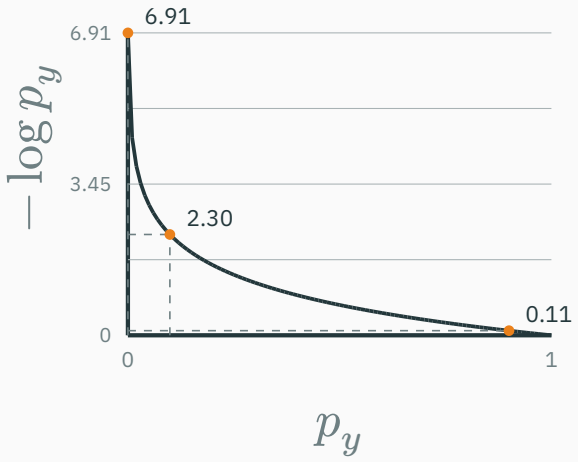
Confident mistakes are expensive

V VISUAL

$$\ell(p_y) = -\log p_y$$

Read off the loss at three confidences for the **true** class:

$$p_y = 0.9 \Rightarrow \ell \approx 0.105$$
$$p_y = 0.1 \Rightarrow \ell \approx 2.303$$
$$p_y = 0.001 \Rightarrow \ell \approx 6.908$$



The negative log makes a confident wrong prediction a large loss.

Checkpoint: confident mistakes

Q QUESTION

Why does cross-entropy penalize a confidently wrong prediction strongly?

A. It makes all class probabilities equal

B. The true-class probability is near zero, so $-\log p_y$ is large

C. It adds an L_2 penalty to the logits

D. The gradient is always zero

Answer: confident mistakes

A ANSWER

Correct: B — The true-class probability is near zero

Why: The negative log grows without bound as p_y approaches zero, matching the fact that the model declared the observed class nearly impossible.

Bayes’ rule for learning

Bayes' rule: from events to model parameters

V VISUAL

PROBABILITY LECTURE

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

A — event or hypothesis to infer
B — observed event or information

Examples: disease given a test; urn given a sampled colour.

MACHINE / DEEP LEARNING

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})}$$

$p(\boldsymbol{\theta})$ — prior before seeing the dataset
$p(\mathcal{D}|\boldsymbol{\theta})$ — likelihood of the fixed dataset
$p(\mathcal{D})$ — evidence (normalizing constant)
$p(\boldsymbol{\theta}|\mathcal{D})$ — posterior after seeing the dataset

Question: which parameter settings remain plausible after seeing the data?

Replace A by θ and B by $\mathcal{D}$: the algebra is unchanged; the interpretation changes.

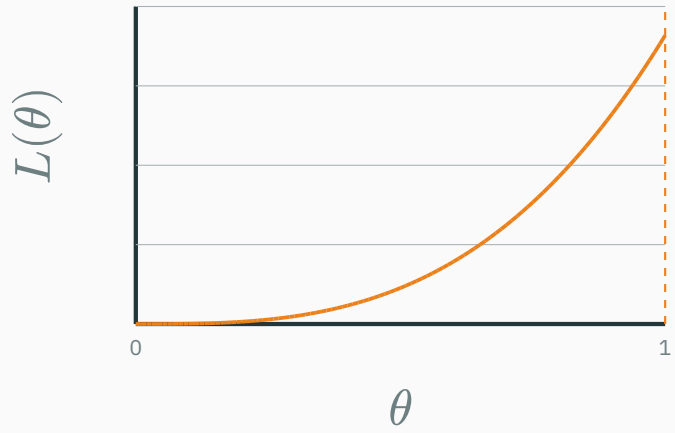
Three heads do not prove $\theta = 1$

Q QUESTION

Let $\theta = P(H)$ and suppose the first three flips are $\mathcal{D} = (H, H, H)$.

H · H · H

$$L(\theta) = \theta^3$$



$$L(\theta; \mathcal{D}) = p(\mathcal{D} \mid \theta) = \theta^3$$

$$\hat{\theta}_{\text{MLE}} = \arg\max_{0 \le \theta \le 1} \theta^3 = 1$$

Did we learn that the coin **always** lands heads —or did a merely head-biased coin produce a lucky run?

MLE identifies the best-fitting θ; it does not make $\theta = 1$ certain.

A prior tempers the conclusion from three flips

V VISUAL

Choose a smooth prior on $\theta \in [0, 1]$, highest near the fair-coin value 0.5.

Prior

$$p(\theta) = 6\theta(1-\theta)$$

Likelihood of H, H, H

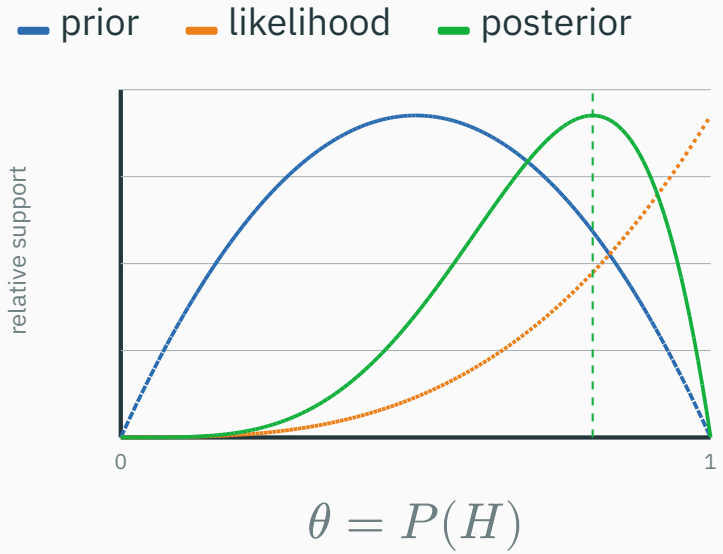
$$p(\mathcal{D} \mid \theta) = \theta^3$$

Posterior

$$p(\theta \mid \mathcal{D}) \propto \theta^3 p(\theta) \propto \theta^4(1-\theta)$$

Likelihood peak (MLE): $\theta = 1$

Posterior peak: $\theta = \frac{4}{5} = 0.8$



With little data, the prior matters; as observations accumulate, the likelihood increasingly dominates.

Step 1: fixed data give fixed likelihoods

D DERIVATION

Suppose we consider only two possible coins.

Fair coin F

$$P(H|F) = 0.5$$

Heads-biased coin B

$$P(H|B) = 0.9$$

Observed data: H · H · H

$$P(H, H, H|F) = 0.5^3 = 0.125 \qquad P(H, H, H|B) = 0.9^3 = 0.729$$

$$\text{likelihood ratio} = \frac{0.729}{0.125} \approx 5.83$$

HHH favours B over F by a factor of 5.83. These likelihoods do not depend on the prior.

Step 2: update a prior that favours the fair coin

D DERIVATION

Before seeing the flips, suppose our starting belief is $P(F) = 0.90$ and $P(B) = 0.10$.

Hypothesis	Prior	Likelihood	Prior $\times$ likelihood
Fair F	0.90	0.125	$0.90 \times 0.125 = 0.1125$
Biased B	0.10	0.729	$0.10 \times 0.729 = 0.0729$

$w_F = 0.1125$ and $w_B = 0.0729$ are **weights**, not probabilities yet.

Bayes first scores each hypothesis by prior $\times$ likelihood. The two scores need not add to 1.

Step 3: normalize the weights into probabilities

D DERIVATION

The normalizing constant is the sum of the two weights:

$$P(\mathcal{D}) = w_F + w_B = 0.1125 + 0.0729 = 0.1854.$$

$P(\mathcal{D})$ is the **evidence** (also called marginal likelihood); it makes posterior probabilities sum to 1.

Fair coin

$$P(F|\mathcal{D}) = \frac{0.1125}{0.1854} \approx 0.607$$

Heads-biased coin

$$P(B|\mathcal{D}) = \frac{0.0729}{0.1854} \approx 0.393$$

The normalized values now add to 1: $0.607 + 0.393 = 1$. They are the **posterior** probabilities—our beliefs after observing HHH.

HHH raises belief in B from 0.10 to 0.393, but it does not overturn the strong fair-coin prior.

Step 4: start instead from a biased-coin prior

D DERIVATION

Now use the **same hypotheses and the same HHH**, but start from $P(F) = 0.10$ and $P(B) = 0.90$.

Hypothesis	Prior	Likelihood	Weight
Fair F	0.10	0.125	0.0125
Biased B	0.90	0.729	0.6561

$$P(\mathcal{D}) = 0.0125 + 0.6561 = 0.6686$$

$$P(F|\mathcal{D}) \approx 0.019, \quad P(B|\mathcal{D}) \approx 0.981$$

The data again favours B by the same factor 5.83; this time the prior already favoured B.

What changed—and what stayed fixed?

V VISUAL

Starting belief	Prior $P(B)$	Posterior $P(B\|H,H,H)$
Probably fair	0.10	0.393
Probably biased	0.90	0.981

Fixed: the observations and likelihoods. In both analyses, HHH multiplies the odds for B by 5.83.

Changed: the prior. Different starting beliefs therefore lead to different posterior probabilities.

Odds for B versus F mean $\frac{P(B)}{P(F)}$.

posterior odds $=$ prior odds $\times$ likelihood ratio

Bayes does not erase the prior; it updates it with the same observations. More data makes the likelihood increasingly dominant.

Maximum a posteriori (MAP) estimation

D DERIVATION

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathcal{D}) = \arg\max_{\boldsymbol{\theta}} p(\mathcal{D} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})$$

The evidence $p(\mathcal{D})$ is constant with respect to $\boldsymbol{\theta}$, so it does not change the maximizing parameter. Take negative logs:

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} = \arg\min_{\boldsymbol{\theta}} \left(\underbrace{-\log p(\mathcal{D} \mid \boldsymbol{\theta})}_{\text{NLL}} + \underbrace{-\log p(\boldsymbol{\theta})}_{\text{negative log-prior}} \right)$$

MAP objective: NLL + negative log-prior. In ML language: data loss + regularization.

Priors, MAP and regularization

Every learner has an inductive bias

V VISUAL

When several predictors fit the training data, the learner still needs a preference for **unseen inputs**. That preference is its **inductive bias**.

Model	Built-in preference $\rightarrow$ consequence
Linear regression	responses vary linearly with features $\rightarrow$ line or plane extrapolation
k-NN	nearby inputs tend to have similar outputs $\rightarrow$ local prediction
Decision tree	a few axis-aligned rules partition the space $\rightarrow$ piecewise-constant regions
CNN	useful local patterns repeat across positions $\rightarrow$ shared filters recognize translations

A prior is one explicit inductive bias; the model class, features, architecture, and optimization add others.

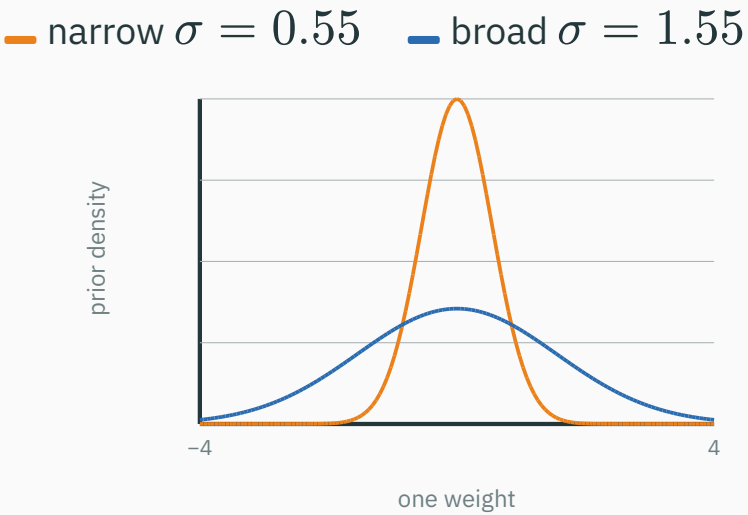
A neural-network prior ranks plausible parameter vectors

V VISUAL

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$$

For a neural network, $\boldsymbol{\theta}$ contains all weights and biases. Before using the current dataset, a prior states which settings are more plausible.

It can prefer small or sparse weights (**sparse** = many weights at or near zero), smooth functions, and correlated parameters.



narrow = stronger preference near 0

A prior makes part of the model's inductive bias explicit as a probability distribution.

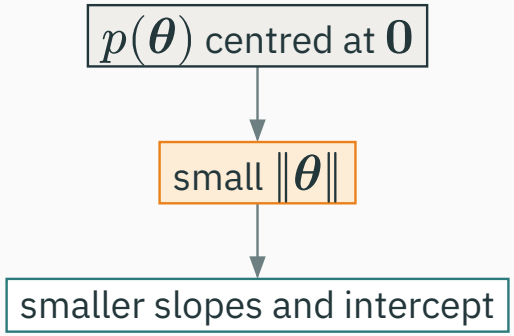
Parameter priors have visible consequences

V VISUAL

Example: a zero-centred prior makes parameter values near $\boldsymbol{\theta} = \mathbf{0}$ more plausible before seeing data.

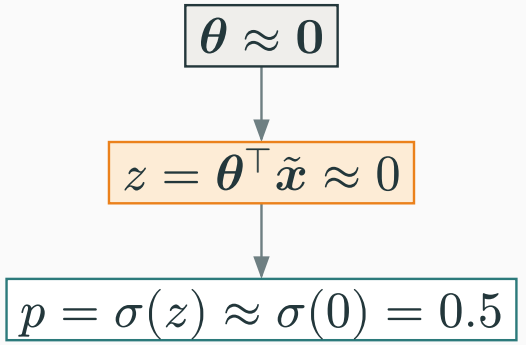
REGRESSION

$$Y_i \mid \boldsymbol{X}_i = \boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\theta}^\top \tilde{\boldsymbol{x}}_i, \sigma^2)$$



BINARY CLASSIFICATION

$$p_{\boldsymbol{\theta}}(Y = 1 \mid \boldsymbol{x}) = \sigma(\boldsymbol{\theta}^\top \tilde{\boldsymbol{x}})$$



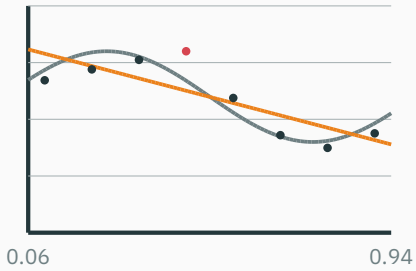
This conclusion depends on the prior: a prior centred elsewhere can prefer probabilities away from 0.5.

Model complexity: underfitting to overfitting

Fit the same eight observations by least squares; only the polynomial degree changes.

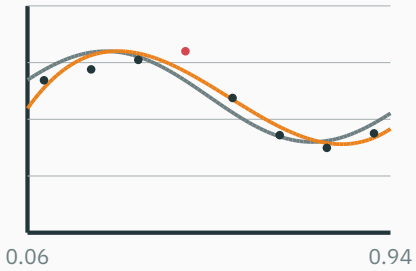
dashed = underlying trend · orange = fitted polynomial · red dot = one noisy observation

degree 1



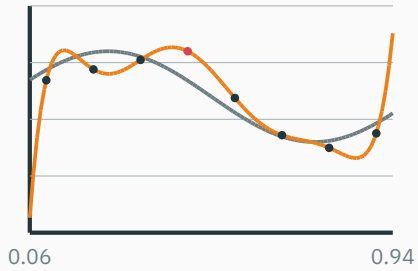
underfit

degree 3



captures the main trend

degree 7



overfit

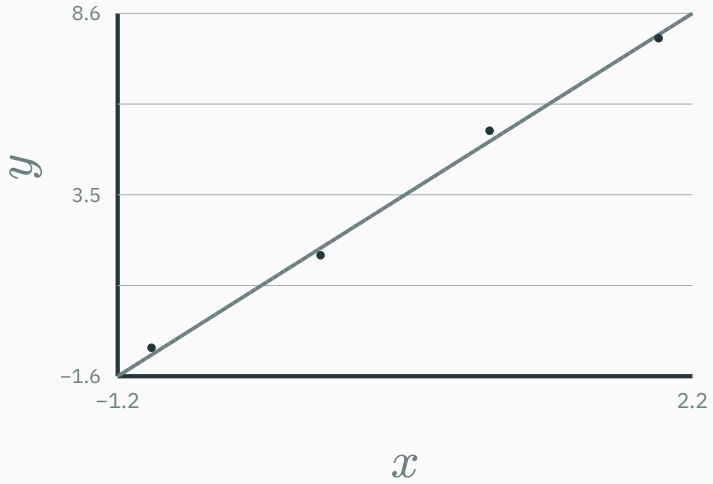
Higher degree lowers training error, but degree 7 chases the noisy observation and becomes unstable. Priors and regularization can prefer smoother solutions.

Start with four noisy points from a straight line

V VISUAL

The data were generated from $f_{\text{true}(x)} = 2 + 3x$ with small measurement noise.

x	$f_{\text{true}(x)}$	observed y
-1	-1	-0.8
0	2	1.8
1	5	5.3
2	8	7.9



grey line = generating rule · dots = observed data

The learner sees only the four observations and must infer the intercept b and slope w.

The likelihood peaks near the generating line

D DERIVATION

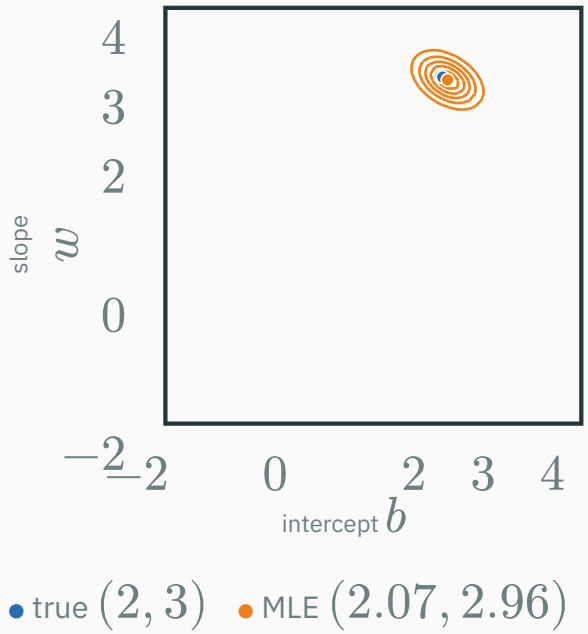
Model each observation as

$$Y_i | x_i, b, w \sim \mathcal{N}(b + w x_i, 0.5^2).$$

The data score each candidate line

$$L(b, w) \propto \exp\left(-\tfrac{1}{2(0.5)^2} \textstyle\sum_i (y_i - b - w x_i)^2\right).$$

candidate (b, w)	squared error
$(2, 3)$	0.180
$(0, 1)$	56.580
MLE $(2.07, 2.96)$	0.162



The observed data make intercepts near 2 and slopes near 3 most plausible.

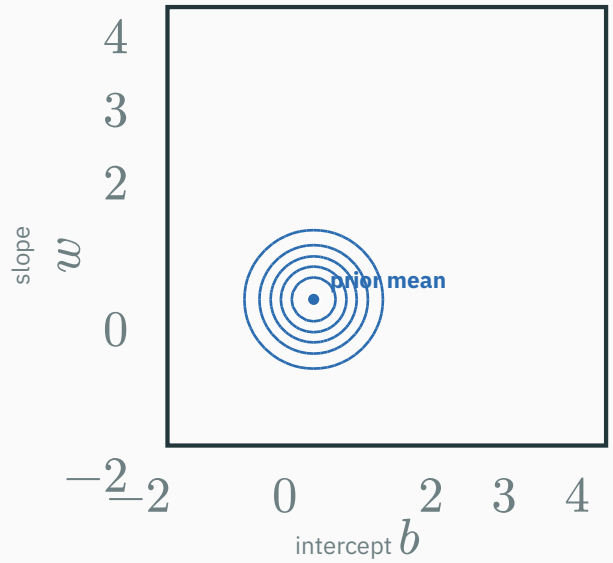
Before data, the prior prefers parameters near zero

V VISUAL

Choose a simple Gaussian prior over the two regression parameters:

$$\boldsymbol{\theta} = (b, w) \sim \mathcal{N}(\mathbf{0}, \tau^2 \boldsymbol{I}), \quad \text{here } \tau = 0.5.$$

τ is the prior standard deviation; $\boldsymbol{I}$ makes b and w independent with equal scale.



What does it express?

$$-\log p(b, w) = \tfrac{1}{2\tau^2}(b^2 + w^2) + C = 2(b^2 + w^2) + C \quad \text{when } \tau = 0.5.$$

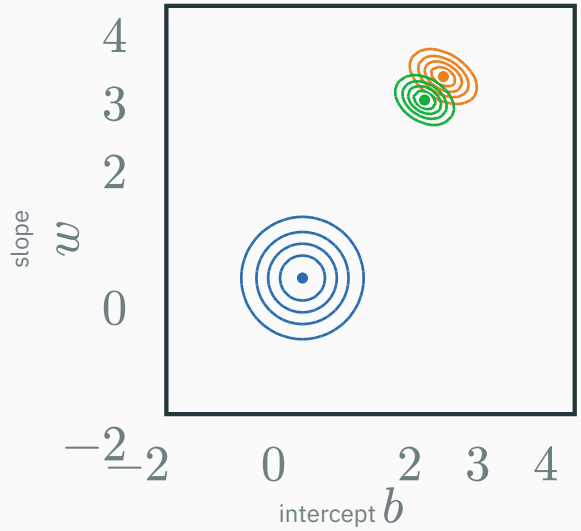
Before seeing this dataset:

- $b \approx 0$ and $w \approx 0$: small-intercept, flatter lines
- farther from $(0, 0)$: less prior density

The likelihood says “fit these data”; the prior says “prefer smaller coefficients”.

Bayes combines the regression likelihood and prior

$$p(b, w \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid b, w)\, p(b, w)}{p(\mathcal{D})}$$



Contours: prior likelihood posterior

MLE $(2.07, 2.96)$ posterior peak $(1.79, 2.62)$ prior mean $(0, 0)$

The narrower prior visibly pulls the posterior peak away from the MLE and toward zero.

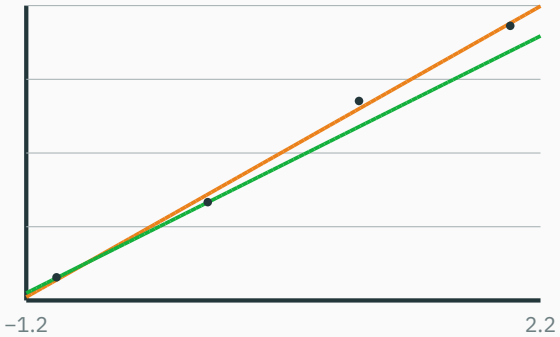
The prior changes the fitted line

V VISUAL

MLE, dashed MAP, solid

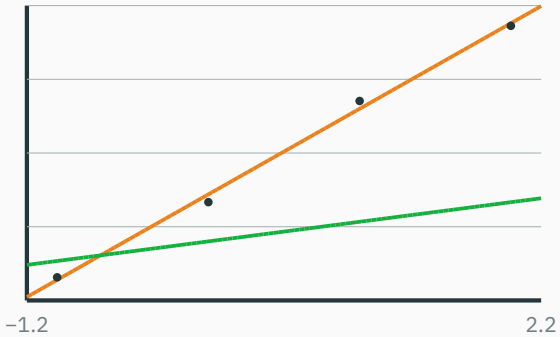
Moderate prior: $\tau = 0.5$

MAP: $\hat{y} = 1.79 + 2.62x$



Much stronger prior: $\tau = 0.1$

MAP: $\hat{y} = 0.44 + 0.68x$



$$(b, w): \quad (2.07, 2.96)_{\text{MLE}} \rightarrow (1.79, 2.62)_{\text{moderate}} \rightarrow (0.44, 0.68)_{\text{strong}}$$

Reducing τ puts more prior mass near the origin and moves both MAP coefficients much closer to zero.

A Gaussian prior is L_2 regularization

D DERIVATION

Assume the parameters are centred at zero before seeing the data:

$$\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \tau^2 \boldsymbol{I}).$$

$$\underbrace{-\log p(\mathcal{D} \mid \boldsymbol{\theta})}_{\text{NLL}} + \underbrace{-\log p(\boldsymbol{\theta})}_{\text{negative log-prior}}$$

$$-\log p(\boldsymbol{\theta}) = \frac{1}{2\tau^2}\|\boldsymbol{\theta}\|_2^2 + C$$

$\|\boldsymbol{\theta}\|_2^2 = \sum_j \theta_j^2$ is the sum of squared coefficients.

$$\Rightarrow \quad \mathcal{L}_{\text{MAP}} = \text{NLL} + \underbrace{\frac{1}{2\tau^2}}_{\lambda} \|\boldsymbol{\theta}\|_2^2 + C$$

$\lambda = \frac{1}{2\tau^2}$: smaller $\tau \to$ larger $\lambda \to$ stronger shrinkage toward 0.

Gaussian MAP gives ridge shrinkage, step by step

D DERIVATION

For one coefficient, let $z = \hat{\theta}_{\mathrm{MLE}}$ be the likelihood centre and s its width:

$$p(\mathcal{D}|\theta) \propto \exp\!\left(-\tfrac{(\theta-z)^2}{2s^2}\right), \quad p(\theta) \propto \exp\!\left(-\tfrac{\theta^2}{2\tau^2}\right).$$

$$J_2(\theta) = \underbrace{\tfrac{(\theta-z)^2}{2s^2}}_{\text{data NLL}} + \underbrace{\tfrac{\theta^2}{2\tau^2}}_{\text{Gaussian prior NLL}}.$$

$$\mathrm{d}\frac{J_2}{\mathrm{d}\theta} = \frac{\theta - z}{s^2} + \frac{\theta}{\tau^2} = 0$$

$$\tau^2(\theta - z) + s^2\theta = 0 \quad \rightarrow \quad (\tau^2 + s^2)\theta = \tau^2 z$$

$$\hat{\theta}_{\mathrm{MAP}} = \alpha z, \quad \alpha = \tfrac{\tau^2}{\tau^2+s^2} \in (0,1).$$

If $z = 0.6$ and $s = \tau$, then $\alpha = \frac{1}{2}$ and MAP $= 0.3$. Ridge reaches exactly zero only when $z = 0$.

Prior width controls how far MAP moves

V VISUAL

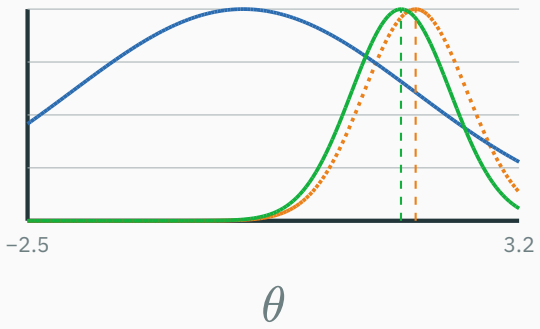
Same likelihood: $L(\theta) \propto \mathcal{N}\left(\theta; \hat{\theta}_{\text{MLE}} = 2, s^2\right)$ with $s = 0.6$; prior: $p(\theta) = \mathcal{N}(0, \tau^2)$.

$$\hat{\theta}_{\text{MAP}} = \frac{\tau^2}{\tau^2 + s^2}\hat{\theta}_{\text{MLE}}$$ (a shrinkage factor between 0 and 1)

prior likelihood posterior

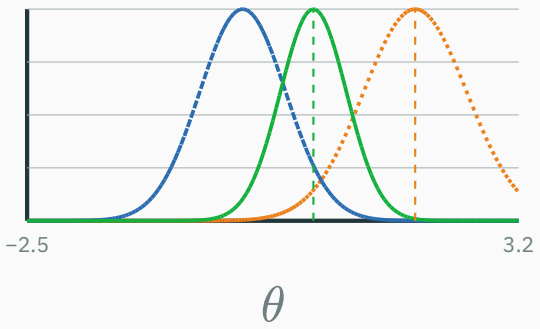
Broad prior: $\tau = 2, \lambda = 0.125$

$\text{MLE} = 2 \rightarrow \text{MAP} \approx 1.83$



Narrow prior: $\tau = 0.5, \lambda = 2$

$\text{MLE} = 2 \rightarrow \text{MAP} \approx 0.82$



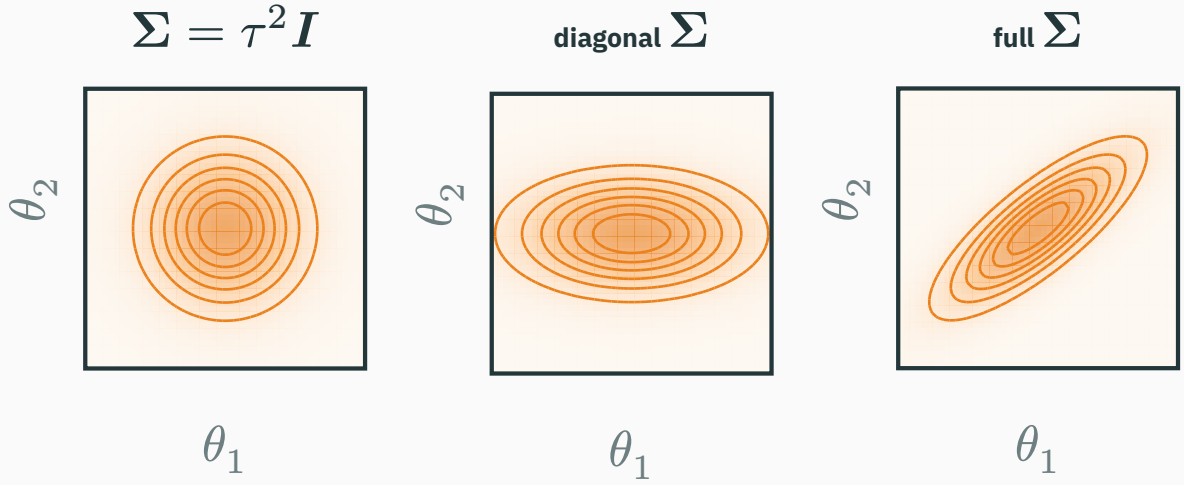
The narrower prior gives the larger λ and pulls MAP farther from the MLE toward zero.

A multivariate Gaussian prior has geometry

V VISUAL

$$\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad -\log p(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}) + C$$

$\boldsymbol{\Sigma}$ is the covariance matrix; $\boldsymbol{\Sigma}^{-1}$ is the precision matrix, which determines how strongly each direction is penalized.



Axes: θ_1 and θ_2 are two coordinates of the parameter vector $\boldsymbol{\theta}$.

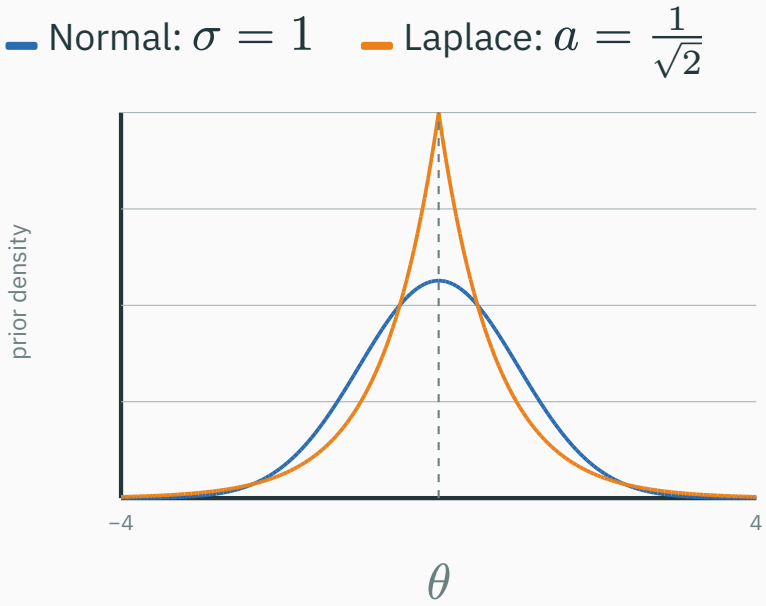
Colour = prior density $p(\boldsymbol{\theta})$: pale outer regions are less plausible; darker centres are more plausible.

Covariance says **which directions in parameter space are plausible.**

Laplace is sharper at zero than a Normal prior

V VISUAL

Compare zero-centred priors with the **same variance**, 1:



Normal prior

- smooth around $\theta = 0$
- lighter tails

Laplace prior

- a sharp peak at $\theta = 0$
- heavier tails

The Laplace peak creates a corner in the negative log-prior. Next we derive why that corner can set coefficients exactly to zero.

Laplace MAP gives an L_1 objective

D DERIVATION

Use the scalar likelihood centred at $z = \hat{\theta}_{\mathrm{MLE}}$ and a zero-centred Laplace prior with scale a:

$$p(\mathcal{D}|\theta) \propto e^{-\frac{(\theta-z)^2}{2s^2}}, \quad p(\theta) = \frac{1}{2a} e^{-\frac{|\theta|}{a}}.$$

$$-\log p(\theta) = \frac{|\theta|}{a} + \log(2a).$$

$$J_1(\theta) = \frac{(\theta - z)^2}{2s^2} + \frac{|\theta|}{a} + C.$$

For vectors, $\|\boldsymbol{\theta}\|_1 = \sum_j |\theta_j|$. Multiply by s^2; the minimizer does not change:

$$\tilde{J}_1(\theta) = \tfrac{1}{2}(\theta - z)^2 + \lambda|\theta|, \quad \lambda = \tfrac{s^2}{a}.$$

The Laplace negative log-prior is proportional to $|\theta|$, so MAP becomes an L_1-regularized optimization problem.

Soft thresholding follows from three cases

D DERIVATION

Differentiate $\tilde{J}_1(\theta) = \frac{1}{2}(\theta - z)^2 + \lambda|\theta|$ on either side of the corner.

Case 1: $\theta > 0$

$\tilde{J}_{1'} = \theta - z + \lambda$

$0 = \theta - z + \lambda$

$\theta = z - \lambda$, valid if $z > \lambda$

Case 2: $\theta < 0$

$\tilde{J}_{1'} = \theta - z - \lambda$

$0 = \theta - z - \lambda$

$\theta = z + \lambda$, valid if $z < -\lambda$

Case 3: $\theta = 0$

$\partial|\theta||_0 = [-1, 1]$

$0 \in -z + \lambda[-1, 1]$

valid iff $|z| \leq \lambda$

$$z > \lambda \quad \hat{\theta}_{\text{MAP}} = z - \lambda \qquad |z| \leq \lambda \quad \hat{\theta}_{\text{MAP}} = 0 \qquad z < -\lambda \quad \hat{\theta}_{\text{MAP}} = z + \lambda$$

$$\hat{\theta}_{\text{MAP}} = \text{sign}(z)\max(|z| - \lambda, 0).$$

Values inside the threshold band $[-\lambda, \lambda]$ map exactly to zero; values outside are pulled toward zero by λ.

Ridge shrinks; L_1 can threshold to exactly zero

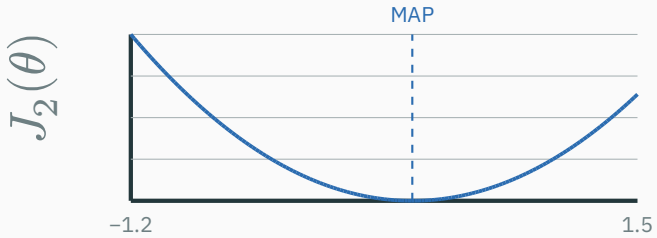
D DERIVATION

For one coefficient, let the data term be $D(\theta) = \frac{1}{2}(\theta - 0.6)^2$, centred at $z = 0.6$.

Normal prior $\rightarrow L_2$

$$J_2(\theta) = D(\theta) + \tfrac{1}{2}\theta^2$$

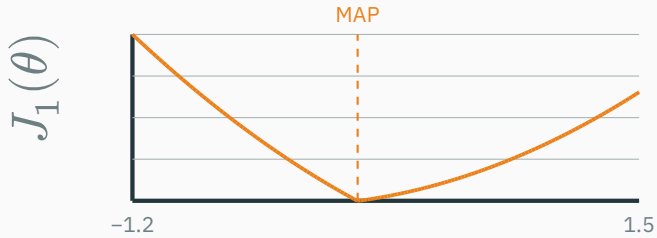
$$\hat{\theta}_{\text{MAP}} = \tfrac{z}{1+1} = 0.3$$



Laplace prior $\rightarrow L_1$

$$J_1(\theta) = D(\theta) + |\theta|$$

$$\hat{\theta}_{\text{MAP}} = \text{sign}(z)\max(|z| - 1, 0) = 0$$

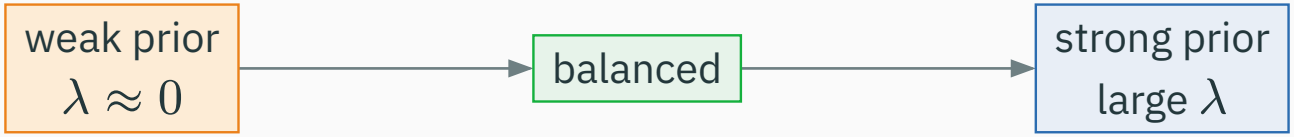


When the quadratic data term separates by coefficient—as with orthogonal features—L_1 applies this threshold coordinate by coordinate and can produce sparsity: many exact zeros.

Prior strength controls underfitting and overfitting

V VISUAL

For the summed-NLL convention used above, $\lambda = \frac{1}{2\tau^2}$:



DATA FIT DOMINATES

flexible model
sensitive to this sample · overfit

LIKELIHOOD + PRIOR

enough flexibility
better generalization

PRIOR DOMINATES

weights forced small
too rigid · underfit

Regularization trades flexibility for stability; more is not always better.

Interactive: likelihood, prior, MAP

I INTERACTIVE

INTERACTIVE

↗ nipunbatra.github.io/interactive-articles/map-regularization

map-regularization.html — a two-parameter linear model so contours are visible. Shows **likelihood**, **prior**, and **posterior** contours with the **MLE**, **MAP**, and zero points.
Controls: amount of data · noise · prior variance · Gaussian vs Laplace · prior correlation.

Ask: more data? narrower prior? why does MAP → MLE as data grows? a correlated prior?

Full Bayes and deep learning

MAP is still one parameter vector

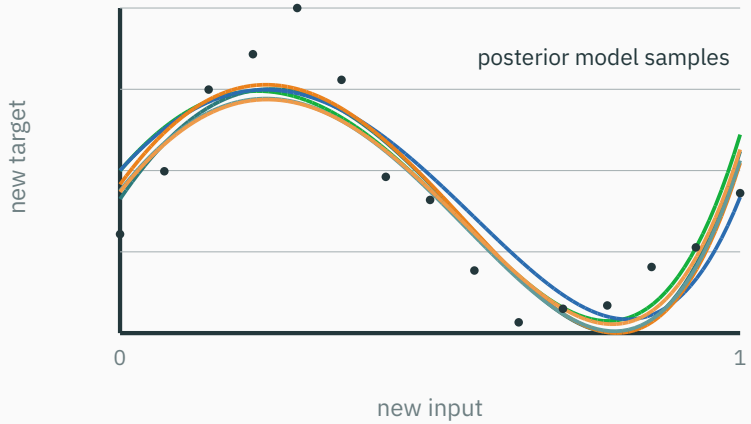
MLE $\hat{\boldsymbol{\theta}}_{\mathrm{MLE}}$ — one point

MAP $\hat{\boldsymbol{\theta}}_{\mathrm{MAP}}$ — one point

Full Bayes keeps the whole posterior $p(\boldsymbol{\theta} \mid \mathcal{D})$ — a **distribution**, not a point.

Posterior predictive distribution

$$p(y^\star \mid \boldsymbol{x}^\star, \mathcal{D}) = \int p_{\boldsymbol{\theta}}(y^\star \mid \boldsymbol{x}^\star)\, p(\boldsymbol{\theta} \mid \mathcal{D})\, \mathrm{d}\boldsymbol{\theta}$$



Bayesian prediction **averages over plausible models** — and reports uncertainty.

Why DL usually uses point estimates

A modern network has millions to billions of parameters, so integrating $p(\boldsymbol{\theta} \mid \mathcal{D})$ is hard. Common practical choices:

- MLE · MAP
- deep **ensembles**
- **dropout**-based approximations
- **variational inference**: fit a tractable approximate posterior

The standard supervised DL objective

B is the current minibatch; $|B|$ is its number of examples.

$$\min_{\boldsymbol{\theta}} \underbrace{\frac{1}{|B|} \sum_{i \in B} -\log p_{\boldsymbol{\theta}}(y_i \mid \boldsymbol{x}_i)}_{\text{likelihood / data fit}} + \underbrace{\lambda \|\boldsymbol{\theta}\|_2^2}_{\text{prior / regularization}}$$

With an averaged NLL, λ absorbs the dataset or minibatch scaling convention.

probabilistic model + neural network + gradient optimization

DL doesn't discard probabilistic ML — it **scales** it with expressive functions and gradient descent.

Retrieval summary

V VISUAL

Gaussian likelihood	$\Rightarrow$ **MSE**
Categorical likelihood	$\Rightarrow$ **cross-entropy**
Gaussian prior	$\Rightarrow L_2$
Laplace prior	$\Rightarrow L_1$
MLE + prior	$\Rightarrow$ **MAP**

Every supervised objective begins with two assumptions

V VISUAL

MODEL

observation model

$$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x})$$

parameter prior

$$p(\boldsymbol{\theta})$$

$\rightarrow$

TRAIN

$$\mathcal{L}_{\mathrm{MAP}}(\boldsymbol{\theta}) = \underbrace{\mathrm{NLL}(\mathcal{D}; \boldsymbol{\theta})}_{\text{data term}} + \underbrace{\Omega(\boldsymbol{\theta})}_{\text{parameter term}}$$

$$\Omega(\boldsymbol{\theta}) = -\log p(\boldsymbol{\theta})$$

$$\hat{\boldsymbol{\theta}}_{\mathrm{MAP}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{MAP}}$$

$\rightarrow$

PREDICT

plug-in: fix $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$

$$p_{\hat{\boldsymbol{\theta}}}(y^\star \mid \boldsymbol{x}^\star)$$

or average over the posterior

$$p(y^\star \mid \boldsymbol{x}^\star, \mathcal{D})$$

Loss $\leftrightarrow$ likelihood; regularizer $\leftrightarrow$ prior; prediction $\leftrightarrow$ plug-in or posterior predictive.

Next: from linear models to neural networks — **neurons, nonlinearities, and multilayer perceptrons.**