Expected Utility Explanation

W

This is a copy of a conversation between ChatGPT & Anonymous.

Uploaded an image
I don't undestand why it is written as an expectation of u(y, 0) \geq expecration U(y, 1). Can you please explain

In a nutshell, the inequality

$$\backslash\mathrm{E}\big[U(y^*,0)\big] \;\geq\; \backslash\mathrm{E}\big[U(y^*,1)\big]$$

is simply the statement “pick the action $a \in \{0,1\}$ that has the higher *expected* utility,” where the expectation is taken over the *unknown* true label y^*. Since you don’t know whether $y^* = 0$ (“no cancer”) or $y^* = 1$ (“cancer”) at the time you make your prediction, you weight each possible true outcome by its probability and compare the two actions:

1. **Expected utility of predicting “no cancer”** ($a = 0$):

$$\backslash\mathrm{E}\big[U(y^*,0)\big] = P(y^* = 0 \mid x)\,U_{00} \;+\; P(y^* = 1 \mid x)\,U_{10} = (1-p)\,U_{00} \;+\; p\,U_{10},$$

where

- $U_{00} = U(y^* = 0,\, \hat{y} = 0)$ is the utility of correctly predicting no cancer,
- $U_{10} = U(y^* = 1,\, \hat{y} = 0)$ is the utility (usually a big penalty) of missing a real cancer.

2. **Expected utility of predicting “cancer”** ($a = 1$):

$$\backslash\mathrm{E}\big[U(y^*,1)\big] = (1-p)\,U_{01} \;+\; p\,U_{11},$$

where

- $U_{01} = U(y^* = 0,\, \hat{y} = 1)$ is the utility (false-alarm cost) of a healthy patient flagged as cancer,
- $U_{11} = U(y^* = 1,\, \hat{y} = 1)$ is the utility of correctly catching a real cancer.

You “dare” to predict $\hat{y} = 0$ (i.e.\ no-cancer) exactly when

$$(1-p)\,U_{00} \;+\; p\,U_{10} \;\geq\; (1-p)\,U_{01} \;+\; p\,U_{11}\,,$$

because that means the *average* payoff of saying “no cancer” outweighs the average payoff of saying “cancer.” Rearranging that inequality gives you the threshold

$$p \;\leq\; \frac{U_{01} - U_{00}}{(U_{10} - U_{11}) + (U_{01} - U_{00})},$$

and plugging in your numerical utilities (e.g.\ $-1, -10, -100$, etc.) yields the specific cutoff $p \approx 0.099$, or equivalently $1-p > 0.901$. In plain English:

> **“If you’re more than 90.1% sure there’s *no* cancer (i.e.\ $P(y^*=0 \mid x) > 0.901$), then maximize your expected utility by predicting ‘no cancer.’”**

That’s why it’s written in terms of expectations: you never know the true y^* at decision time, so you compare the *expected* payoffs of each action under your posterior belief about y^*.

Uploaded an image
How do I write it all up?

Here’s a clean way you could write the whole thing up in your notes or in a paper. I’ve broken it into the usual pieces—setup, expected-utility calculations, decision rule, threshold derivation, and numerical plug-in—so that it reads as a self-contained argument.

---

1. Setup

- We have a binary outcome $y^* \in \{0,1\}$ (“no cancer” vs. “cancer”) and two possible predictions (actions) $\hat{y} \in \{0,1\}$.
- The utility of predicting $\hat{y} = a$ when the true label is y^* is

$$U_{y^*a} \;=\; U\bigl(y^*,\,a\bigr).$$

- Our model gives us a posterior probability

$$p \;=\; P\bigl(y^*=1 \mid x\bigr), \quad 1-p \;=\; P\bigl(y^*=0 \mid x\bigr).$$

---

2. Expected utilities

Since you don’t know y^* at decision time, you choose the action a with the higher *expected* utility under your posterior.

1. **Predict “no cancer”** $(a=0)$:

$$\E\bigl[U(y^*,0)\bigr] = P(y^*=0 \mid x)\,U_{00} \;+\; P(y^*=1 \mid x)\,U_{10} = (1-p)\,U_{00} \;+\; p\,U_{10}.$$

2. **Predict “cancer”** $(a=1)$:

$$\E\bigl[U(y^*,1)\bigr] = (1-p)\,U_{01} \;+\; p\,U_{11}.$$

---

3. Decision rule

Choose the action a whose expected utility is larger. In particular, you "dare" to predict $\hat{y} = 0$ exactly when

$$\backslash\mathrm{E}\big[U(y^*,0)\big] \;\ge\; \backslash\mathrm{E}\big[U(y^*,1)\big],$$

i.e.

$$(1-p)\,U_{00} + p\,U_{10} \;\ge\; (1-p)\,U_{01} + p\,U_{11}.$$

---

4. Solving for the probability threshold

Rearrange the inequality to isolate p:

1. Expand both sides:

$$(1-p)\,U_{00} + p\,U_{10} \;\ge\; (1-p)\,U_{01} + p\,U_{11}.$$

2. Collect the terms in p on one side:

$$(1-p)\,U_{00} - (1-p)\,U_{01} \;\ge\; p\,U_{11} - p\,U_{10},$$

   or

$$(1-p)\,(U_{00} - U_{01}) \;\ge\; p\,(U_{11} - U_{10}).$$

3. Divide through by the positive quantity $\big[(U_{10} - U_{11}) + (U_{01} - U_{00})\big]$ (you check signs so the inequality stays the same), to get

$$p \;\le\; \frac{U_{01} - U_{00}}{(U_{10} - U_{11}) + (U_{01} - U_{00})}.$$

Thus the critical threshold is

$$p_{\text{thresh}} \;=\; \frac{U_{01} - U_{00}}{(U_{10} - U_{11}) + (U_{01} - U_{00})} \quad\Longrightarrow\quad \begin{cases} \text{predict "no cancer"} & \text{if } p \le p_{\text{thresh}}, \\ \text{predict "cancer"} & \text{if } p > p_{\text{thresh}}. \end{cases}$$

---

5. Numerical example

Suppose your utility matrix is (for instance)

	$\hat{y} = 0$ **("no cancer")**	$\hat{y} = 1$ **("cancer")**
$y^* = 0$	$U_{00} = -1$	$U_{01} = -10$
$y^* = 1$	$U_{10} = -100$	$U_{11} = -1$

Then

$$p_{\text{thresh}} = \frac{U_{01} - U_{00}}{(U_{10} - U_{11}) + (U_{01} - U_{00})} = \frac{-10-(-1)}{(-100-(-1)) + (-10-(-1))} = \frac{-9}{(-99)+(-9)} = \frac{-9}{-108}$$

Equivalently,

$$1 - p_{\text{thresh}} \approx 0.9167,$$

so you only predict “no cancer” if you’re at least 91.67% sure there really is *no* cancer.

Putting it all in one paragraph

> **Decision rule:**
> Given a posterior $p = P(y^* = 1 \mid x)$, compute
>
> $$\E[U(y^*, 0)] = (1-p)U_{00} + p\,U_{10}, \qquad \E[U(y^*, 1)] = (1-p)U_{01} + p\,U_{11}.$$
>
> Predict “no cancer” if and only if
>
> $$(1-p)\,U_{00} + p\,U_{10} \;\ge\; (1-p)\,U_{01} + p\,U_{11},$$
>
> which rearranges to
>
> $$p \;\le\; \frac{U_{01} - U_{00}}{(U_{10} - U_{11}) + (U_{01} - U_{00})}.$$
>
> Plugging in $U_{00} = -1$, $U_{01} = -10$, $U_{10} = -100$, $U_{11} = -1$ yields a threshold $p \approx 0.0833$, i.e.\ you only predict “no cancer” when $P(\text{cancer} \mid x) \le 0.0833$ (equivalently $P(\text{no cancer} \mid x) \ge 0.9167$).

Feel free to adapt the notation and the level of wordiness to suit your audience, but this structure will give you a crisp, complete derivation.

How do I know what term to multiply the parts of the utility matrix with p and 1 - p?

Whenever you compute an expected utility, you’re just doing

$$\E\big[U(y^*, a)\big] \;=\; \sum_{y^* \in \{0,1\}} P\big(y^* \mid x\big)\; U\big(y^*, a\big).$$

Here your posterior gives

$$P(y^* = 1 \mid x) = p, \quad P(y^* = 0 \mid x) = 1 - p.$$

So:

- **If you choose action** $a = 0$ (“predict no cancer”), then

  $$\E[U(y^*, 0)] = P(y^* = 0)\,U_{00} \;+\; P(y^* = 1)\,U_{10} = (1-p)\,U_{00} \;+\; p\,U_{10}.$$

  You multiply U_{00} by $(1-p)$ because U_{00} is the utility when $y^* = 0$, which happens with probability $1-p$. You multiply U_{10} by p because U_{10} is the utility when $y^* = 1$, which happens with probability p.

- **If you choose action** $a = 1$ (“predict cancer”), then

$$\backslash\mathrm{E}[U(y^*, 1)] = P(y^* = 0)\,U_{01} \;+\; P(y^* = 1)\,U_{11} = (1-p)\,U_{01} \;+\; p\,U_{11}.$$

Again, U_{01} is the payoff when the true label is 0 (prob.\ $1-p$), and U_{11} when it's 1 (prob.\ p).

In general:

1. Look at the first index of U_{ij}:
    - If it's 0, weight by $P(y^* = 0) = 1 - p$.
    - If it's 1, weight by $P(y^* = 1) = p$.
2. Sum up those weighted terms to get $\backslash\mathrm{E}[U(y^*, a)]$.

That's exactly why the "0-row" utilities all get $(1-p)$ in front of them, and the "1-row" utilities get p.

Uploaded an image
How would I isolate p?

Start from the key inequality:

$$(1-p)\,U_{00} \;+\; p\,U_{10} \;\geq\; (1-p)\,U_{01} \;+\; p\,U_{11}.$$

---

1. Expand both sides

$$U_{00} \;-\; p\,U_{00} \;+\; p\,U_{10} \;\geq\; U_{01} \;-\; p\,U_{01} \;+\; p\,U_{11}.$$

---

2. Collect the "constant" terms on one side, the p-terms on the other

Bring the non-p terms to the left and all the p-terms to the right:

$$U_{00} - U_{01} \;\geq\; p\,U_{00} - p\,U_{10} \;+\; p\,U_{01} - p\,U_{11}.$$

Factor p on the right:

$$U_{00} - U_{01} \;\geq\; p\,\bigl[U_{00} - U_{10} + U_{01} - U_{11}\bigr].$$

---

3. Solve for p

Define

$$D \;=\; U_{00} - U_{10} + U_{01} - U_{11}.$$

Then the inequality is

$$U_{00} - U_{01} \;\geq\; p\,D \quad\Longrightarrow\quad p \;\leq\; \frac{U_{00} - U_{01}}{D}.$$

---

4. Put it in the “standard” form

Notice

$$\frac{U_{00}-U_{01}}{D}=\frac{-(U_{01}-U_{00})}{-(\,U_{10}-U_{11}+U_{01}-U_{00}\,)}=\frac{U_{01}-U_{00}}{U_{10}-U_{11}+U_{01}-U_{00}}.$$

So you can equivalently write

$$\boxed{p\;\le\;\frac{U_{01}-U_{00}}{U_{10}-U_{11}+U_{01}-U_{00}}}$$

as the cutoff for predicting “no cancer.”

In words: when you expand, you get a linear inequality in p; you group all of the p–terms on one side, factor out p, and then divide by the remaining coefficient to isolate p.

Uploaded an image
What would be the utility and threshold for this?

For this matrix

$$U=\begin{array}{c|cc} & \hat{y}=0 & \hat{y}=1 \\ \hline y=0 & 2 & 1 \\ y=1 & 1 & 2 \end{array}$$

and posterior $p=P(y=1\mid x)=0.129$, the two expected utilities are

$$\begin{aligned} \backslash\mathrm{E}\big[U(y,0)\big] &= (1-p)\,U_{00}+p\,U_{10}=(1-0.129)\cdot 2\;+\;0.129\cdot 1=1.742+0.129=1.871,\\ \backslash\mathrm{E}\big[U(y,1)\big] &= (1-p)\,U_{01}+p\,U_{11}=(1-0.129)\cdot 1\;+\;0.129\cdot 2=0.871+0.258=1.129. \end{aligned}$$

Since $1.871>1.129$, you would predict $\hat{y}=0$.

Decision threshold

In general you predict $\hat{y}=0$ exactly when

$$(1-p)\,U_{00}+p\,U_{10}\;\ge\;(1-p)\,U_{01}+p\,U_{11}\implies p\;\le\;\frac{U_{01}-U_{00}}{(U_{10}-U_{11})+(U_{01}-U_{00})}.$$

Here

$$U_{01}-U_{00}=1-2=-1,\quad U_{10}-U_{11}=1-2=-1,$$

so

$$p_{\text{thresh}}=\frac{-1}{(-1)+(-1)}=\frac{-1}{-2}=0.5.$$

Thus for any $p \le 0.5$ you pick $\hat{y} = 0$ (and for $p > 0.5$ you pick $\hat{y} = 1$).

What does it mean?

It simply means that with this particular utility matrix your optimal rule is the usual “choose the more likely class”—i.e. predict $\hat{y} = 0$ whenever $P(y = 1 \mid x) \le 0.5$, and $\hat{y} = 1$ whenever $P(y = 1 \mid x) > 0.5$. Concretely:

- **Threshold** $p_{\text{thresh}} = 0.5$ arises because the gain from a correct decision over a wrong one is the same for both classes (you get $+2$ instead of $+1$ in either the $y = 0$ or the $y = 1$ case), so there is no asymmetry to shift the cutoff away from 0.5.
- In your example $p = 0.129 < 0.5$, so you predict $\hat{y} = 0$.

In other words, when the utilities penalize false-positives and false-negatives equally (and reward true-positives and true-negatives equally), maximizing expected utility just reduces to the MAP rule of picking the class with posterior probability above 50 %.

How do I understand the expected utility?

Think of the expected utility of a decision as the “long-run average payoff” you’d get if you made that same decision over and over again in worlds where y is 0 with probability $1 - p$ and 1 with probability p. Formally, for each action $\hat{y} = a$ you list out:

- What you get (the utility) if $y = 0$: that’s U_{0a}, which happens with chance $1 - p$.
- What you get if $y = 1$: that’s U_{1a}, which happens with chance p.

Then you weight each utility by its probability and add:

$$\E\bigl[U(y,a)\bigr] \;=\; (1-p)\,U_{0a} \;+\; p\,U_{1a}.$$

A concrete illustration

Take your example with

$$U = \begin{array}{c|cc} & \hat{y}=0 & \hat{y}=1 \\ \hline y=0 & 2 & 1 \\ y=1 & 1 & 2 \end{array}$$

and $p = 0.129$.

- **If you predict** $\hat{y} = 0$, you get:
  - 2 points when $y = 0$ (prob.\ $1 - p = 0.871$),

    - 1 point when $y = 1$ (prob.\ $p = 0.129$).

  So

$$\E[U(y, 0)] \;=\; 0.871 \cdot 2 \;+\; 0.129 \cdot 1 \;=\; 1.742 \;+\; 0.129 \;=\; 1.871.$$

- **If you predict** $\hat{y} = 1$, you get:
    - 1 point when $y = 0$ (prob.\ 0.871),
    - 2 points when $y = 1$ (prob.\ 0.129).

  So

$$\E[U(y, 1)] \;=\; 0.871 \cdot 1 \;+\; 0.129 \cdot 2 \;=\; 0.871 \;+\; 0.258 \;=\; 1.129.$$

Since $1.871 > 1.129$, predicting $\hat{y} = 0$ gives the higher *average* payoff, so that's the decision you choose.

Why this matters

- You **can't** know the true y in advance, so you hedge by averaging over your uncertainty.
- The expected utility tells you, "*if* I make this choice now, *on average* how well will I do?" You pick the action with the larger average.
- In large repeated use (or in a single high-stakes decision), this is the rigorous way to translate your beliefs (p) and your preferences/preferences (the U matrix) into one crisp rule.

Uploaded an image
I am confused about how would I do this. Could you please give me a hint?

Here's the one-sentence hint:

> **Use the posterior samples of your model parameters to approximate**
>
> $$p(y^* = k \mid D, x^*) \;=\; \int p(y^* = k \mid \theta, x^*)\, p(\theta \mid D)\, d\theta \;\approx\; \frac{1}{S} \sum_{s=1}^{S} p\big(y^* = k \mid \theta^{(s)},\, x^*\big),$$
>
> or equivalently,
> draw one $y^{*(s)}$ from $p(y^* | \theta^{(s)}, x^*)$ for each posterior draw $\theta^{(s)}$ and then count the fraction of times you get k.

In other words:

1. **Loop over your S posterior draws $\theta^{(1)}, \ldots, \theta^{(S)}$.**
2. For each $\theta^{(s)}$, compute the predictive probability vector
   $\big\{p(y^* = 1 \mid \theta^{(s)}, x^*),\; p(y^* = 2 \mid \theta^{(s)}, x^*),\; p(y^* = 3 \mid \theta^{(s)}, x^*)\big\}$.
3. Average those S vectors component-wise to get your Monte Carlo estimate of $p(y^* = k \mid D, x^*)$.

Or equivalently, if it's easier, for each $\theta^{(s)}$ draw one $y^{*(s)}$ from that categorical distribution and then just tally up how often you get 1, 2, 3. That frequency converges to the table you're after.

But what is the mean?

The "mean" here is nothing fancier than the ordinary arithmetic average you take over your Monte Carlo draws. Concretely, if you have S posterior draws $\theta^{(1)}, \ldots, \theta^{(S)}$, then for each draw you either

1. compute the predictive probability vector

$$\big(p(y^* = 1 \mid \theta^{(s)}, x^*),\ p(y^* = 2 \mid \theta^{(s)}, x^*),\ p(y^* = 3 \mid \theta^{(s)}, x^*)\big),$$

   or

2. sample one draw $y^{*(s)} \sim \mathrm{Categorical}\big(p(y^* \mid \theta^{(s)}, x^*)\big)$.

Then your Monte Carlo estimator for
$p(y^* = k \mid D, x^*)$
is just the **mean** of those S numbers:

- **If you're averaging probabilities**:

$$\hat{p}(y^* = k) = \frac{1}{S}\sum_{s=1}^{S} p\big(y^* = k \mid \theta^{(s)}, x^*\big).$$

- **If you're tallying sampled labels**:

$$\hat{p}(y^* = k) = \frac{1}{S}\sum_{s=1}^{S} \mathbf{1}\{y^{*(s)} = k\}.$$

That $\frac{1}{S}\sum$ is exactly the "mean" of your Monte Carlo draws.

I have no idea how would I do this

Let's make it completely concrete with a tiny "template" in (Python-style) pseudocode. You can adapt it to whatever language or framework you're using.

---

1. **Assume you have**
   - S posterior draws $\{\theta^{(s)}\}_{s=1}^{S}$,
   - a function

```python
def predict_probs(theta, x_star):
    # returns a length-K vector [p1, p2, …, pK] = p(y*=k | theta, x_star)
    ...
```

   - the new input x^*.

2. **Build an array to hold your Monte Carlo estimates**

```python
import numpy as np

S = len(theta_samples)
K = …               # number of classes, e.g. 3
probs = np.zeros((S, K))
```

3. **Fill it with the predictive probabilities**

```python
for s in range(S):
    probs[s, :] = predict_probs(theta_samples[s], x_star)
```

4. **Take the “mean” across your Monte Carlo draws**

```python
# This computes, for each k,  (1/S) * sum_{s=1}^S probs[s, k]
p_hat = probs.mean(axis=0)
```

5. **Read off your table**

```python
# p_hat is now a length-K vector whose entries are
# p_hat[k] ≈ p(y* = k+1 | D, x*)   (if you zero-indexed)
```

What does each line do?

- **`probs[s, :] = …`**
  computes the predictive distribution under the single parameter draw $\theta^{(s)}$.
- **`probs.mean(axis=0)`**
  takes the ordinary arithmetic average of those S probability-vectors, component by component.
- The result **`p_hat[k]`** is exactly

$$\hat{p}(y^* = k \mid D, x^*) = \frac{1}{S} \sum_{s=1}^{S} p\big(y^* = k \mid \theta^{(s)}, x^*\big).$$

That’s all there is to the “mean” in Monte Carlo: you just average the per-draw predictive probabilities.

Uploaded an image
What is this equation?

That symbol

$$\|x - x'\|_2$$

denotes the **ℓ_2 norm** (a.k.a. the Euclidean norm) of the vector difference $x - x'$. Concretely, if

$$x = (x_1, x_2, \ldots, x_n), \quad x' = (x'_1, x'_2, \ldots, x'_n),$$

then

$$\|x - x'\|_2 \; = \; \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \cdots + (x_n - x'_n)^2},$$

which you can think of as the ordinary straight-line (Euclidean) distance between the two points x and x' .

How do I represnt it in code?

Here's how you'd write the Euclidean-norm $\|x - x'\|_2$ in a few common flavours of Python:

---

1. Using NumPy

python

```python
import numpy as np

x      = np.array([x1, x2, ..., xn])
x_prime = np.array([x1p, x2p, ..., xnp])

# Option A: np.linalg.norm
dist = np.linalg.norm(x - x_prime)         # default is ord=2

# Option B: explicit
diff = x - x_prime
dist = np.sqrt(np.dot(diff, diff))
```

---

2. Pure-Python (no dependencies)

python

```python
import math

x       = [x1, x2, ..., xn]
x_prime = [x1p, x2p, ..., xnp]

dist = math.sqrt(
    sum((xi - xpi)**2 for xi, xpi in zip(x, x_prime))
)
```

---

3. With `scipy` convenience

```
python

from scipy.spatial.distance import euclidean

dist = euclidean(x, x_prime)
```

In every case, **`dist`** will be

$$\|x - x'\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - x'_i)^2}\,.$$

Uploaded an image
How would I do the following?

For any zero-mean Gaussian process

$$f(x) \sim \backslash\mathrm{GP}\big(0,\ k_2(x, x')\big),$$

the prior variance at the point x is just

$$\backslash\mathrm{Var}\big[f(x)\big] \;=\; k_2(x, x)\,,$$

because

$$\backslash\mathrm{Cov}\big(f(x), f(x')\big) = k_2(x, x') \quad\Longrightarrow\quad \backslash\mathrm{Var}\big[f(x)\big] = k_2(x, x)\,.$$

Plugging in your kernel from eq. (8)

If in your notes eq. (8) is, say, the squared-exponential kernel

$$k_2(x, x') = \sigma^2 \exp\Big(-\tfrac{(x - x')^2}{2\ell^2}\Big),$$

then simply set $x' = x$:

$$\backslash\mathrm{Var}\big[f(x)\big] = k_2(x, x) = \sigma^2 \exp\Big(-\tfrac{(x - x)^2}{2\ell^2}\Big) = \sigma^2 \exp(0) = \sigma^2.$$

So the **analytical prior variance** is constant σ^2.

If your eq (8) is some other stationary kernel, you do exactly the same thing—substitute $x' = x$ into k_2 and simplify. That gives you the closed-form expression for $\backslash\mathrm{Var}[f(x)]$.

How do you find prior density?

Whenever you have

$$f \sim \backslash\mathrm{GP}(0,\ k_2(x,x'))$$

and you want the **prior density** of the function values at one or more inputs, you just write down the corresponding Gaussian density:

1. At a single point x

The prior on $f(x)$ is

$$f(x)\ \sim\ \mathbb{N}\big(0,\ k_2(x,x)\big).$$

Hence its density is

$$p\big(f(x)\big) = \frac{1}{\sqrt{2\pi\, k_2(x,x)}}\ \exp\Bigl(-\frac{[\,f(x)\,]^2}{2\, k_2(x,x)}\Bigr).$$

2. At a finite collection of inputs $\{x_i\}_{i=1}^n$

Let $\mathbf{f} = [f(x_1),\dots,f(x_n)]^\top$ and form the $n \times n$ covariance matrix

$$K_{ij} = k_2(x_i,x_j).$$

Then

$$\mathbf{f}\ \sim\ \mathbb{N}\big(\mathbf{0},\ K\big),$$

and its joint prior density is the usual multivariate-Gaussian formula:

$$p(\mathbf{f}) = \frac{1}{(2\pi)^{n/2}\,|K|^{1/2}}\ \exp\Bigl(-\tfrac{1}{2}\,\mathbf{f}^\top\, K^{-1}\,\mathbf{f}\Bigr).$$

That’s all there is to “finding the prior density” of a Gaussian-process: pick your input locations, build the covariance K using your kernel k_2, and plug into the Gaussian pdf.

How is it for normal distribution

For a one-dimensional Gaussian (normal) random variable

$$X \sim \mathbb{N}(\mu,\sigma^2),$$

the **density** (pdf) is

$$p(x) = \frac{1}{\sqrt{2\pi\,\sigma^2}}\ \exp\Bigl(-\frac{(x-\mu)^2}{2\,\sigma^2}\Bigr).$$

For a d-dimensional multivariate normal

$$\mathbf{X} \sim \mathbb{N}(\boldsymbol{\mu}, \Sigma),$$

the **joint density** is

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}\,|\Sigma|^{1/2}}\,\exp\Bigl(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\top}\,\Sigma^{-1}\,(\mathbf{x}-\boldsymbol{\mu})\Bigr).$$

In the GP-prior context:

- **At one input** x, $f(x) \sim \mathbb{N}(0,\, k_2(x,x))$ so you use the one-dimensional formula with $\mu = 0$ and $\sigma^2 = k_2(x,x)$.
- **At** n **inputs** $\{x_i\}$, $\mathbf{f} = [f(x_1), \ldots, f(x_n)] \sim \mathbb{N}(\mathbf{0}, K)$ with $K_{ij} = k_2(x_i, x_j)$, so you use the multivariate formula above with $\boldsymbol{\mu} = \mathbf{0}$ and $\Sigma = K$.