Bayesian Linear Regression & Gaussian Processes script

Marco Milanta & Anastasiia Makarova

October 2021

1 Linear regression with feature map

The goal of the script is to show the relation between Bayesian linear regression (BLR) and Gaussian processes regression (GPR). Consider an unknown function $f : \mathcal{X} \to \mathbb{R}$ defined over some domain $\mathcal{X}$, and a dataset $\{(x_1, y_1), \dots, (x_m, y_m)\}$ of function evaluations $y = f(x) + \epsilon$ perturbed by some noise ϵ, $x \in \mathcal{X}, y$.

In BLR, we assume f to be linear x, particularly, $y = x^T\mathbf{w} + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. The weights $\mathbf{w}$ are unknown and a priori assumed to be normally distributed $p(\mathbf{w}) = \mathcal{N}(0, \sigma_p^2\mathbf{I}_d)$. Here, we consider a more sophisticated model, assuming f to be linear in the feature map $\phi : \mathcal{X} \to \mathbb{R}^d$, i.e.:

$$y = \phi(x)^T\mathbf{w} + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma_n^2).$$

Alternatively, in GPR, we assume prior directly over the function f, i.e., $f \sim \mathcal{N}(\mu, k)$, where

$$\begin{aligned} \mu(x) &= 0 \quad \forall x \in \mathcal{X} \\ k(x, x') &= \sigma_p^2\phi(x)^T\phi(x') \quad \forall x, x' \in \mathcal{X}. \end{aligned}$$

Task: We want to make a prediction $y(x^*)$ at some new point $x^* \in \mathcal{X}$ as well as measure the uncertainty in the prediction. To this end, we derive the posterior over $y(x^*)$ in both cases, showing the advantages and weaknesses of each method.

1.1 Bayesian linear regression

Here we derive the posterior for Bayesian linear regression in features space.

(a) We first rewrite the problem in the vector form as $y_{1:m} = \mathbf{\Phi}\mathbf{w} + \epsilon, \epsilon \sim \mathcal{N}(0, \mathbf{I}_m\sigma_n^2)$, where

$$\mathbf{\Phi} := \begin{bmatrix} \phi(x_1)^T \\ \phi(x_2)^T \\ \vdots \\ \phi(x_m)^T \end{bmatrix} \in \mathbb{R}^{m\times d}.$$

(b) The posterior distribution over $\mathbf{w}$ can be rewritten via Bayes rule as follows:

$$p(\mathbf{w} \mid y_{1:m}) = \frac{p(\mathbf{w}, y_{1:m})}{p(y_{1:m})} = \frac{1}{p(y_{1:m})}p(y_{1:m} \mid \mathbf{w})p(\mathbf{w}).$$

Further derivation will rely in the following two intuition:

1. The quantity $\frac{1}{p(y_{1:m})}$ (or of any quantity which doesn't depend on $\mathbf{w}$) can be ignored since $p(\mathbf{w} \mid y_{1:m})$ must integrate to 1 and, therefore, a multiplicative factor that doesn't depend on $\mathbf{w}$ can be recomputed at the end. To show this better, let's say we find

   $$p(\mathbf{w} \mid y_{1:m}) = \frac{1}{Z}\tilde{p}(\mathbf{w}) \qquad \text{where } Z \text{ is unknown.}$$

   Then, by imposing that it integrates to 1 we get

   $$\begin{aligned} 1 &= \int_{\mathbb{R}^d} \frac{1}{Z}\tilde{p}(\mathbf{w})d\mathbf{w} \\ 1 &= \frac{1}{Z}\int_{\mathbb{R}^d} \tilde{p}(\mathbf{w})d\mathbf{w} \qquad \text{since } Z \text{ doesn't depend on } \mathbf{w} \\ Z &= \int_{\mathbb{R}^d} \tilde{p}(\mathbf{w})d\mathbf{w}. \end{aligned}$$

2. Both $p(\mathbf{w})$ and $p(y_{1:m} \mid \mathbf{w})$ represent Gaussian distributions, where the latter is due to $y_{1:m} \mid \mathbf{w} \sim \mathcal{N}(\mathbf{\Phi w}, \mathbf{I}_m\sigma_m^2)$ since

   $$y_{1:m} = \underbrace{\mathbf{\Phi w}}_{\text{fixed if we condition on } \mathbf{w}} + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \mathbf{I}_m\sigma_\varepsilon^2).$$

Then, once we understand consideration 1 and 2, we can compute:

$$\begin{aligned} p(\mathbf{w} \mid y_{1:m}) &= \frac{1}{Z}p(y_{1:m} \mid \mathbf{w})p(\mathbf{w}) \\ &= \frac{1}{Z} \underbrace{\frac{1}{\sqrt{(2\pi)^m|\sigma_n^2\mathbf{I}_m|}}}_{\text{doesn't depend on } \mathbf{w} \to = Z'} \exp\left(-\frac{1}{2}(y_{1:m} - \mathbf{\Phi w})^T \underbrace{(\sigma_n^2\mathbf{I}_m)^{-1}}_{\frac{1}{\sigma_n^2}}(y_{1:m} - \mathbf{\Phi w})\right) \cdot \\ &\cdot \underbrace{\frac{1}{\sqrt{(2\pi)^d|\sigma_p^2\mathbf{I}_d|}}}_{\text{doesn't depend on } \mathbf{w} \to = Z''} \exp\left(-\frac{1}{2}\mathbf{w}^T \underbrace{(\sigma_p^2\mathbf{I}_d)^{-1}}_{\frac{1}{\sigma_p^2}} \mathbf{w}\right) \\ &= \frac{Z'Z''}{Z}\exp\left(-\frac{1}{2}\left(\frac{1}{\sigma_n^2}\left\|y_{1:m}\right\|^2 + \frac{1}{\sigma_n^2}\mathbf{w}^T\mathbf{\Phi}^T\mathbf{\Phi w} + \frac{1}{\sigma_p^2}\mathbf{w}^T\mathbf{w} - \frac{1}{\sigma_n^2}y_{1:m}^T\mathbf{\Phi w} - \frac{1}{\sigma_n^2}\mathbf{w}^T\mathbf{\Phi}^T y_{1:m}\right)\right) \\ &= \frac{Z'Z''}{Z}\exp\left(-\frac{1}{2}\left(\mathbf{w}^T \underbrace{\left(\frac{1}{\sigma_n^2}\mathbf{\Phi}^T\mathbf{\Phi} + \frac{1}{\sigma_p^2}\mathbf{I}_d\right)}_{:=\bar{\Sigma}^{-1}} \mathbf{w} - \underbrace{\frac{1}{\sigma_n^2}y_{1:m}^T\mathbf{\Phi}}_{:=\nu^T}\mathbf{w} - \mathbf{w}^T \underbrace{\frac{1}{\sigma_n^2}\mathbf{\Phi}^T y_{1:m}}_{:=\nu} + \frac{1}{\sigma_n^2}\left\|y_{1:m}\right\|^2\right)\right) \cdot \\ &\cdot \underbrace{\exp^{\left(-\frac{1}{2}\frac{1}{\sigma_n^2}\left\|y_{1:m}\right\|^2\right)}}_{\text{doesn't depend on } \mathbf{w} \to = Z'''} \\ &= \frac{Z'Z''Z'''}{Z}\exp\left(-\frac{1}{2}\left(\mathbf{w}^T\bar{\Sigma}^{-1}\mathbf{w} - \nu^T\mathbf{w} - \mathbf{w}^T\nu\right)\right). \end{aligned}$$

We later clarify how we define $\bar{\Sigma}^{-1}$, and now let us shown how to get to a standard posterior via completing the square in the vector case. Particularly, we want to find $\bar{\mu}$ and δ such that:

$$(\mathbf{w}-\bar{\mu})^T\bar{\Sigma}^{-1}(\mathbf{w}-\bar{\mu})+\delta=\mathbf{w}^T\bar{\Sigma}^{-1}\mathbf{w}-\nu^T\mathbf{w}-\mathbf{w}^T\nu.$$

We leave as an easy challenge to show that $\bar{\mu}=\bar{\Sigma}\nu$ and $\delta=-\bar{\mu}^T\bar{\Sigma}^{-1}\bar{\mu}$. Then,

$$\begin{aligned}p(\mathbf{w}\mid y_{1:m}) &= \frac{Z'Z''Z'''}{Z}\exp\left(-\frac{1}{2}(\mathbf{w}-\bar{\mu})^T\bar{\Sigma}^{-1}(\mathbf{w}-\bar{\mu})-\frac{1}{2}\delta\right)\\ &= \frac{Z'Z''Z'''}{Z}\exp\left(-\frac{1}{2}(\mathbf{w}-\bar{\mu})^T\bar{\Sigma}^{-1}(\mathbf{w}-\bar{\mu})\right)\qquad\underbrace{\exp\left(-\frac{1}{2}\delta\right)}_{\text{doesn't depend on }\mathbf{w}\rightarrow=Z''''}\\ &= \frac{Z'Z''Z'''Z''''}{Z}\exp\left(-\frac{1}{2}(\mathbf{w}-\bar{\mu})^T\bar{\Sigma}^{-1}(\mathbf{w}-\bar{\mu})\right).\end{aligned}$$

Finally, we have a term resembling PDF of Gaussian distribution with mean $\bar{\mu}$ and covariance matrix $\bar{\Sigma}$, and the constant factors Zs. This yields:

$$\mathbf{w}\mid y_{1:m}\sim\mathcal{N}(\bar{\mu},\bar{\Sigma}).$$

(c) Finally, for the prediction $y^*=\phi(x^*)^T\mathbf{w}+\epsilon^*$ we get:

$$y^*\mid y_{1:m}\sim\mathcal{N}(\phi(x^*)^T\mu,\phi(x^*)^T\bar{\Sigma}\phi(x^*)+\sigma_n^2).$$

This follows by two simple facts:

- $\mathbb{E}[Ax]=A\mathbb{E}[x]$
- $cov[Ax]=Acov[x]A^T$

Finally, we can unfold $\bar{\mu}$ and $\bar{\Sigma}$

$$y^*\mid y_{1:m}\sim\mathcal{N}\left(\phi(x^*)^T\frac{1}{\sigma_n^2}\left(\frac{1}{\sigma_n^2}\mathbf{\Phi}^T\mathbf{\Phi}+\frac{1}{\sigma_p^2}\mathbf{I}_d\right)^{-1}\mathbf{\Phi}^Ty_{1:m},\phi(x^*)^T\left(\frac{1}{\sigma_n^2}\mathbf{\Phi}^T\mathbf{\Phi}+\frac{1}{\sigma_p^2}\mathbf{I}_d\right)^{-1}\phi(x^*)+\sigma_n^2\right).$$

2 GP approach

Here we look at the problem in a different way. What we did before was in two stages: first we find the posterior distribution on $\mathbf{w}$, then we look for the prediction in x^*. In the GP context we look directly at the distribution of the prediction y^*, we completely elude $\mathbf{w}$. Later on, we will see that this can bring to great advantages.

(a) Combining data and prediction in a single vector: The strategy is to put y^* in a vector with $y_{1:m}$, and then use the conditioning. This is nice, since we have very good formulas for conditioning Gaussian vectors. We write

$$\begin{bmatrix}y^*\\ y_{1:m}\end{bmatrix}=\begin{bmatrix}\phi(x^*)\\ \mathbf{\Phi}\end{bmatrix}\mathbf{w}+\begin{bmatrix}0\\ \mathbf{I}_m\end{bmatrix}\epsilon.$$

Therefore, since it's all Gaussian:

$$\begin{aligned}\begin{bmatrix} y^* \\ y_{1:m}\end{bmatrix} &\sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}\phi(x^*)^T \\ \boldsymbol{\Phi}\end{bmatrix}\sigma_p^2\mathbf{I}_d\begin{bmatrix}\phi(x^*) & \boldsymbol{\Phi}^T\end{bmatrix} + \begin{bmatrix}0\\ \mathbf{I}_m\end{bmatrix}\sigma_\varepsilon^2\mathbf{I}_m\begin{bmatrix}0 & \mathbf{I}_m\end{bmatrix}\right)\\ &= \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}\phi(x^*)^T\phi(x^*)\sigma_p^2 & \sigma_p^2\phi(x^*)^T\boldsymbol{\Phi}^T \\ \sigma_p^2\boldsymbol{\Phi}\phi(x^*) & \sigma_p^2\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2\mathbf{I}_m\end{bmatrix}\right).\end{aligned}$$

Here we use just used the formula $cov[Ax] = Acov(c)A^T$.

(b) Moving to the kernel: The next big step is to get rid of ϕ. To do so, we remember the definition of the kernel $k(x, x') := \sigma_p^2\phi(x)^T\phi(x)$. We now see that this is very reasonable, since, we can use k to get rid of the direct dependency from ϕ. Practically, what we show here, is that we can rewrite the matrix $\begin{bmatrix}\phi(x^*)^T\phi(x^*)\sigma_p^2 & \sigma_p^2\phi(x^*)^T\boldsymbol{\Phi}^T \\ \sigma_p^2\boldsymbol{\Phi}\phi(x^*) & \sigma_p^2\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2\mathbf{I}_m\end{bmatrix}$ can be written only by using the kernel:

- For the top-left corner:
$$\sigma_p^2\phi(x)^T\phi(x') = k(x, x')$$
- For the top-right and bottom-left:
$$\sigma_p^2\boldsymbol{\Phi}\phi(x^*) = \sigma_p^2\begin{bmatrix}\phi(x_1)\\ \vdots \\ \phi(x_n)\end{bmatrix}\phi(x^*) = \begin{bmatrix}\sigma_p^2\phi(x_1)^T\phi(x^*)\\ \vdots \\ \sigma_p^2\phi(x_v)^T\phi(x^*)\end{bmatrix} = \begin{bmatrix}k(x_1, x^*)\\ \vdots \\ k(x_n, x^*)\end{bmatrix} := \mathbf{k}_{Ax^*}$$
- For the bottom-right:
$$\begin{aligned}\sigma_p^2\boldsymbol{\Phi}^T\boldsymbol{\Phi} &= \sigma_p^2\begin{bmatrix}\phi(x_1)^T\phi(x)\\ \vdots \\ \phi(x_v)^T\phi(x)\end{bmatrix}\begin{bmatrix}\phi(x_1) & \cdots & \phi(x_n)\end{bmatrix}\\ &= \begin{bmatrix}\sigma_p^2\phi(x_1)^T\phi(x_1) & \cdots & \sigma_p^2\phi(x_1)^T\phi(x_n)\\ \vdots & \ddots & \vdots \\ \sigma_p^2\phi(x_n)^T\phi(x_1) & \cdots & \sigma_p^2\phi(x_n)^T\phi(x_n)\end{bmatrix}\\ &= \begin{bmatrix}k(x_1, x_1) & \cdots & k(x_1, x_n)\\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n)\end{bmatrix} := \mathbf{K}_{AA}\end{aligned}$$

Notice that everything only depends on ϕ only via the kernel! Then we can rewrite:

$$\begin{bmatrix} y^* \\ y_{1:m}\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}k(x^*, x^*) & \mathbf{k}_{Ax^*}^T \\ \mathbf{k}_{Ax^*} & \mathbf{K}_{AA} + \sigma_\varepsilon^2\mathbf{I}_m\end{bmatrix}\right)$$

(c) Making predictions: Now we can simply condition on data using the well known formula for Gaussian vectors. In this way we get directly the prediction

$$\begin{aligned}y^* \mid y_{1:m} &\sim \mathcal{N}\left(\tilde{\mu}, \tilde{\sigma}^2\right),\\ \tilde{\mu} &= \mathbf{k}_{Ax^*}^T(\mathbf{K}_{AA} + \sigma_\varepsilon^2\mathbf{I}_m)^{-1}y_{1:m},\\ \tilde{\sigma}^2 &= k(x^*, x^*) - \mathbf{k}_{Ax^*}^t(\mathbf{K}_{AA} + \sigma_\varepsilon^2\mathbf{I}_m)^{-1}\mathbf{k}_{Ax^*}.\end{aligned}$$

(d) Substitute back ϕ To compare it to the case before, we can substitute back ϕ. Now, we can use the well known formulas to condition $y^* \mid y_{1:m}$. What we get is

$$
\begin{aligned}
y^* \mid y_{1:m} &\sim \mathcal{N}(\tilde{\mu}, \tilde{\sigma}^2) \\
\tilde{\mu} &= \phi(x^*)^T \mathbf{\Phi}^T \left(\mathbf{\Phi}\mathbf{\Phi}^T + \frac{{\sigma_\varepsilon}^2}{\sigma_p^2} \mathbf{I}_m \right)^{-1} y_{1:m} \\
\tilde{\sigma}^2 &= \sigma_p^2 \left(\phi(x^*)^T \phi(x^*) - \phi(x^*)^T \mathbf{\Phi}^T \left(\mathbf{\Phi}\mathbf{\Phi}^T + \frac{\sigma_\varepsilon^2}{\sigma_p^2} \mathbf{I}_m \right)^{-1} \mathbf{\Phi}\phi(x^*) \right).
\end{aligned}
$$

3 Conclusion

Those two method yields of course the same result, but through completely different formulas. In the homework, for a specific kernel, you will show that the formulas are indeed equivalent. But how are those formula different? The dimension of the matrix we have to invert is drastically different:

- BLR: $(1/\sigma_n^2 \mathbf{\Phi}^T \mathbf{\Phi} + 1/\sigma_p^2 \mathbf{I}_d)^{-1} \in \mathbb{R}^{d \times d}$
- GP: $(\mathbf{K}_{AA} + \sigma_n^2 \mathbf{I}_m)^{-1} \in \mathbb{R}^{m \times m}$

In most of the cases we think that $d < n$, therefore the BLR seems to be much smarter. But, since the GP methods complexity doesn't scale with d, we can pick a feature map with d to be huge. In the limit, we can even take $d = \infty$. Even though this looks very For this to make sense we also need to make sure that

$$k(x, x') = \sum_{i=1}^{\infty} \phi(x)_i \phi(x')_i < \infty \quad \forall x, x' \in \mathcal{X}.$$

Such infinite dimensions kernels are very common! Gaussian, Laplace and Matérn are just some examples.

Random Fourier features: Now we can also understand the idea behind random Fourier features. Random Fourier features allow us to find a feature map $\mathbf{z} : \mathcal{X} \to \mathbb{R}^d$ such that:

$$k(x, x') \simeq \mathbf{z}^T(x)\mathbf{z}(x'),$$

where the precision of the inequality scales with d. Now, we can revert back the problem of GP regression to a problem of BLR by using $\mathbf{z}$. This is reasonable if m is huge. In this scenario, even if we take d to be quite large to achieve a good approximation, BLR is still much faster.