Gaussian Processes

Advanced Topics in High-Performance Computing

Faisal Qureshi
Gaussian Processes

These slides are based on Ch. 15 of the Kevin P. Murphy book titled, “Machine Learning: a Probabilistic Approach.”
Gaussian Processes

Consider the regression tasks. Given data \((x_i, y_i), i = 1, 2, \ldots, N\). We assume \(y_i = f(x_i)\) for some unknown function \(f\), possibly corrupted by noise. Our goal is to discover \(f\).

The optimal approach is to infer a distribution over functions given data, i.e., we need to find \(p(f|X, y)\).

Once we have \(p(f|X, y)\), we can use this to make predictions of the following form: “what is the \(y_*\) given a \(x_*\)” as follows

\[
p(y_*|x_*, X, y) = \int p(y_*|f, x_*)p(f|X, b y)df.
\]

Aside

To date, we have modelled \(p(\theta|D)\), we now endeavour to model \(p(f, D)\).
Guassian Process

- A GP defines a prior over functions.
- This prior can be converted to a posterior over functions once we have been presented with some data.
- Define a distribution over the function’s values at a finite, but arbitrary, set of points $x_1, x_2, \cdots, x_N$ to represent distribution over functions.
- A GP assumes that $p(f(x_1), f(x_2), \cdots, f(x_N))$ is jointly Gaussian with mean $\mu(x)$ and covariance $\Sigma(x)$ given by $\Sigma_{ij} = K(x_i, x_j)$. $K$ is a positive definite kernel function.
- The key idea is that if $x_i$ and $x_j$ are deemed by the kernel to be similar, then we expect the output of the function at these points to be similar too.
GP for regression

- Consider a prior on the regression function of GP

\[
f(x) \sim \text{GP}(m(x), \mathcal{K}(x, x')),
\]

where \( m(x) \) is the mean function and \( \mathcal{K}(x, x') \) is the covariance function. Specifically

\[
m(x) = \mathbb{E}[f(x)]
\]

and

\[
\mathcal{K}(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))]
\]

- This process defines a joint Gaussian

\[
p(f|X) = \mathcal{N}(f|\mu, K).
\]
Predictions using noise-free observations

- Given training data \( D = \{(x_i, f_i), i = 1 : N\} \), where \( f_i = f(x_i) \) is the noise-free observation of the function evaluated at \( x_i \).
- Given a test set \( X_* \) of size \( N_* \times D \), we want to predict the function outputs \( f_* \).
- By definition GP joint distribution has the following form

\[
\begin{pmatrix}
  f \\
  f_*
\end{pmatrix} \sim \mathcal{N}
\left( \begin{pmatrix}
  \mu \\
  \mu_*
\end{pmatrix},
\begin{pmatrix}
  K & K_* \\
  K^T_* & K_{**}
\end{pmatrix}
\right),
\]

where \( K = \mathcal{K}(X, X) \) is \( N \times N \), \( K_* = \mathcal{K}(X, X_*) \) is \( N \times N_* \) and \( K_{**} = \mathcal{K}(X_*, X_*) \) is \( N_* \times N_* \).

- We can get the posterior using the conditioning rules for Gaussians

\[
p(f_*|X_*, X, f) = \mathcal{N}(f_*|\mu_*, \Sigma_*)
\]

\[
\mu_* = \mu(X_*) + K^T_* K^{-1}(f - \mu(X))
\]

\[
\Sigma_* = K_{**} - K_* K^{-1} K_*
\]
Predictions using noisy observations

Consider the case where we observe a noisy version of the underlying functions, i.e., $y = f(x) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma_y^2)$.

The covariance of the observed noisy responses is

$$\text{cov}[y_p, y_q] = K(x_p, x_q) + \sigma_y^2 \delta_{pq},$$

where $\delta_{pq} = \mathbb{1}(p = q)$. So we can write

$$\text{cov}[y|K] = K + \sigma_y^2 \mathbf{I}_N \triangleq K_y.$$

This assumes that the noise terms were independently added to each observation.

The joint density of the observed data and the latent, noise-free function on the test points is given by

$$\begin{pmatrix} f \\ f_* \end{pmatrix} \sim \mathcal{N} \left( \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} K_y & K_* \\ K_*^T & K^{**} \end{pmatrix} \right),$$

where we assume a zero mean.
Predictions using noisy observations

- The posterior predictive density is

\[
p(f_* \mid X_*, X, f) = \mathcal{N}(f_* \mid \mu_*, \Sigma_*)
\]

\[
\mu_* = K_*^T K_y^{-1} f
\]

\[
\Sigma_* = K_{**} - K_*^T K_y^{-1} K_*
\]
Squared-Exponential Kernel

- We can use the squared-exponential kernel, aka Gaussian kernel or RBF kernel, which has the following form in 1D

\[ K(x, x') = \sigma_f^2 \exp \left( -\frac{1}{2l^2} (x - x')^2 \right), \]

where \( l \) controls the horizontal length scale over which the function varies, and \( \sigma_f^2 \) controls the vertical variation.
Effect of Kernel Parameters

- The performance of GPs depends upon the suitability of chosen kernel.
- For the squared-exponent kernel in the previous slide, $l$ and $\sigma_f^2$ control the horizontal and vertical scale respectively.
Estimating kernel parameters

- The squared-exponent kernel discussed above has two parameters: $l$ and $\sigma_f^2$.
- Exhaustive search over a discrete grid of values, with validation loss as objective.
  - Slow
  - This is how SVM (Support Vector Machines) kernels are sometimes tuned
- Maximize marginal likelihood

$$p(y|X) = \int p(y|f, X)p(f|X)df.$$  

- Recall $p(f|X) = \mathcal{N}(f|0, K_y)$ and $p(y|f) = \prod_i \mathcal{N}(y_i|f_i, \sigma_f^2)$, marginal likelihood is

$$\log p(y|X) = \log \mathcal{N}(y|0, K_y)$$

$$= -\frac{1}{2} y K_y^{-1} y - \frac{1}{2} \log |K_y| - \frac{N}{2} \log(2\pi)$$
Estimating kernel parameters

- Marginal likelihood:

\[
\log p(y|X) = -\frac{1}{2} y K_y^{-1} y - \frac{1}{2} \log |K_y| - \frac{N}{2} \log(2\pi)
\]

- The first term is a data fit term
- The second term is a model complexity term
- The last term is a constant

- Kernel parameters are often referred to as hyper parameters
  - \( \theta = [l, \sigma_f^2] \)

- Gradient descent: use

\[
\frac{\partial}{\partial \theta_j} \log p(y|X)
\]

to maximize \( \log p(y|X) \) with respect to hyper parameter \( \theta_j \). The actual form of \( \frac{\partial}{\partial \theta_j} \log p(y|X) \) will depend upon the form of the kernel and the hyper parameter with respect to which we are trying to maximize the kernel.
GP Posterior

3 function samples from the GP posterior
GP Regression
Computational concerns

For numerical stability, it is unwise to invert $K$ or $K_y$. Rather use the following algorithm for perform GP regression.

**Algorithm 15.1: GP regression**

1. $L = \text{cholesky}(K + \sigma^2_y I)$;
2. $\alpha = L^T \backslash (L \backslash y)$;
3. $E[f_*] = k_*^T \alpha$;
4. $v = L \backslash k_*$;
5. $\text{var}[f_*] = \kappa(x_*, x_*) - v^T v$;
6. $\log p(y|X) = -\frac{1}{2}y^T \alpha - \sum_i \log L_{ii} - \frac{N}{2} \log(2\pi)$

(See Section 15.2.5.)

GP using Python

- Check out
  