6.7900 Fall 2024: Lecture Notes 17

1 Gaussian Processes 2

Recap: Last class we introduced Gaussian processes. As a reminder, we can define a Gaussian process as "a collection of random variables, any finite number of which have a joint Gaussian distribution" [Rasmussen and Williams 2006]. We write $f \sim \mathcal{GP}(m, k)$ where $m(x) = \mathbb{E}[f(x)]$ is the mean function and $k(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))]$ the covariance function or kernel. Common choices for these are $m(x) = 0$ and the squared exponential kernel:

$$k(x, x') = \sigma^2 \exp(-\frac{1}{2}\sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2})$$

where the signal variance σ^2 and lengthscales ℓ_d^2 are hyperparameters.

1.1 Fitting hyperparameters

In the last class, we saw how to fit the parameters given some hyperparameters. But how do we select such hyperparameters? Many techniques have been developed for this. A very common one is to use maximum likelihood estimation over them. Luckily most Gaussian processes software packages directly support this. In Figure 1 from 10 datapoints randomly sampled from a Gaussian process we fit using sklearn another Gaussian process with maximum likelihood estimation of the hyperparameters and see that the hyperparameters found are very close to the original ones. The code for this was:

```
signal_stddev_init = 1.0
kern_fit = signal_stddev_init**2*RBF(length_scale=1.0,length_scale_bounds=(0.01, 100))
gpfit = GaussianProcessRegressor(kernel=kern_fit,n_restarts_optimizer=10)
gpfit.fit(xobs,yobs)
```

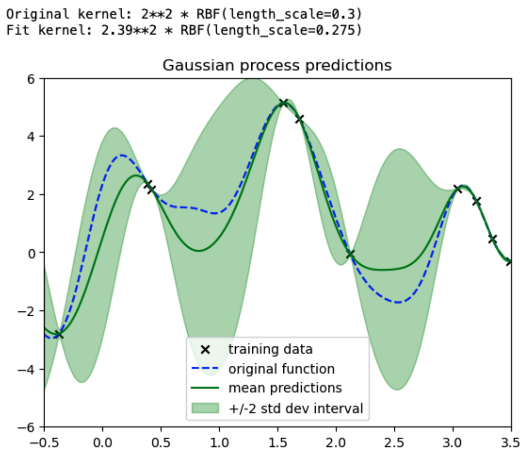

Figure 1: Fitting the hyperparameters of a Gaussian process.

1.2 Observation noise

So far we've been assuming that we observed $f(x)$ directly, but often the actual observation y has additional noise. For example, let's assume that we know that the noise is independent and Gaussian:

$$f \sim \mathcal{GP}(m,k),\ y(n) \sim f(x^{(n)}) + \epsilon^{(n)},\ \epsilon^{(n)} \overset{iid}{\sim} \mathcal{N}(0,\tau^2)$$

We observe $\{(x^{(n)}, y^{(n)})\}_{n=1}^N$ and want to learn the latent f. Because the sum of independent Gaussians is a Gaussian with means summed and covariances summed, the mean of $y^{(n)}$ is $m(x^{(n)})$ and the covariance $\text{Cov}(y^{(n)}, y^{(n')}) = k(x^{(n)}, x^{(n')}) + \tau^2 \mathbf{1}\{n = n'\}$. So our new Gaussian process fitting problem is:

We compare n not x because even for same x the noise is independent.

$$\begin{bmatrix} y^{(1:N)} \\ f(X') \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X,X) + \tau^2 I & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right)$$

Note, that while we are using y for the training data, at test time here we are assuming that we are aiming to still find the distribution of the underlying function not the noisy observations. Figures 2 and 3 show the output of fitting a Gaussian process on the noisy observations to obtain either the distribution of $f(x)$ or that of y.

1.3 Extrapolation

Extrapolation refers to making a prediction beyond the observed data. Typically since we have no data in these regions the behavior of the model largely depends on the intrinsic assumption inside of the model.

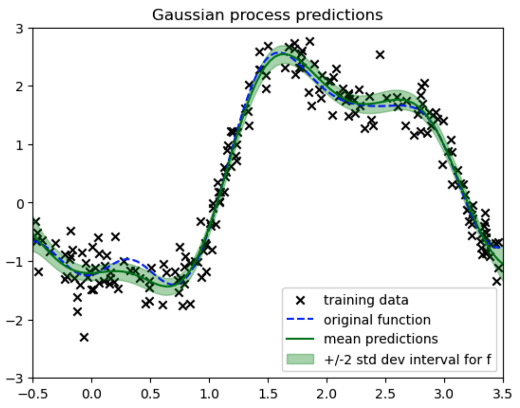

Figure 2: Fitting $f(x)$ from noisy y

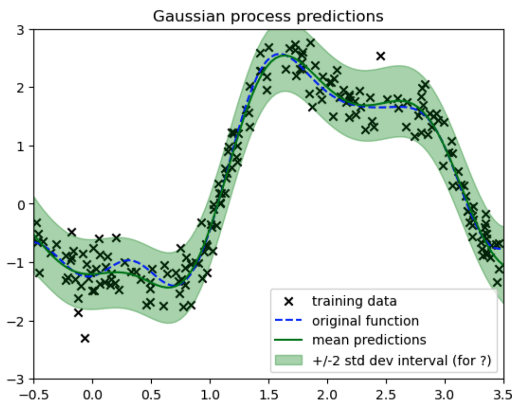

Figure 3: Fitting noisy y from noisy y

For example, when using GPs with a squared exponential kernel, for data points that are more than a handful of length scales from other data points will revert to prior behavior. See Figure 4 for an example of this happening with a population growth prediction model:

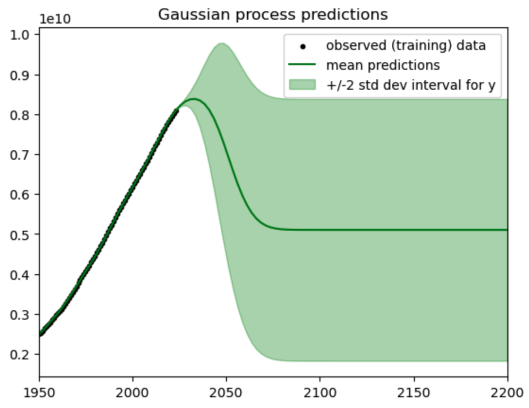

Figure 4: Fitting the population growth data as an example of Gaussian process extrapolation behavior.

Note, however, that often in machine learning we are in a different extrapolation behavior where we do not have a single dataset/time-series like in the human population growth example but instead have multiple datasets/time-series. Examples of this are vital sign records of many patients, purchase behavior for many books, and life cycles of many cells. Is this still extrapolation?

1.4 More than one input

Our illustrations have almost all been for one input so far, but in real life, it's typical to have more than one input. What could go wrong?

Regression in high dimensions is a fundamentally hard problem (without additional assumptions). One way to understand this is by looking at the distance between random points in high dimensions highlighted in Figure 5. When D=500 all the points are greater than 50 units apart!

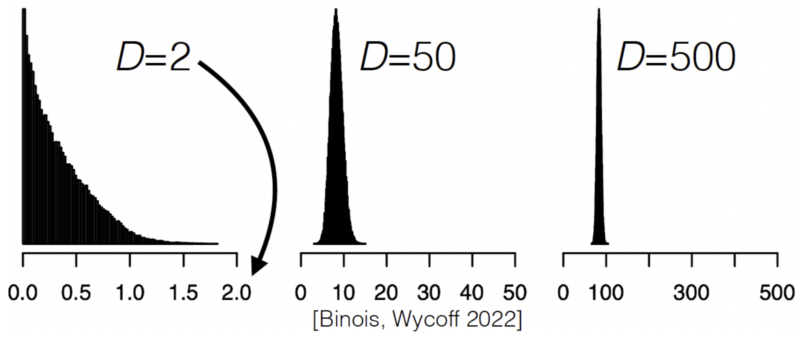

Figure 5: Histogram of squared inter-point distances between 10,000 uniformly randomly sampled points on $[0, 1]^D$.

If we use a Gaussian process either we make the lengthscales much larger than the scale of observations or otherwise for almost all points we would revert to the prior. In practice, this is a very active area of research where several improvements have been proposed.

1.5 Missing data

A very common real-life form of "noise" is missing data. This can be problematic because typical supervised learning algorithms require having a full feature matrix X and label vector Y. So how do we deal with the cases where some of our data is missing? It depends on *how* the data is missing.

Let $M_{nd} = 1$ if feature d in data point n is missing, else 0. Assume Y and X_{obs} are observed and X_{mis} is missing. We can formalize the types of "missingness" in three categories:

1. Missing Completely at Random (MCAR): where missingness doesn't depend on latent or observed X, Y. Formally: $p(M|X_{obs}, X_{mis}, Y) = p(M)$. An example of this is that some data entries were corrupted. This is very convenient from a modeling perspective but mostly unrealistic.

2. Missing at Random (MAR): where missingness depends only on observed (not latent) data. Formally: $p(M|X_{obs}, X_{mis}, Y) = p(M|X_{obs}, Y)$. An example could be assuming that whether we get a response to an income question depends entirely on the age that we fully observe. This is more realistic but still misses important cases.

3. Not Missing at Random (NMAR or MNAR): where missingness can depend on something besides the observed data. For example, the probability of missingness might be a direct function of the unobserved and unknown value (e.g. the likelihood of responding to the income question is a function of the income) or there are confounders (e.g. we don't collect data on education, but education determines the likelihood of income response). This is the most realistic assumption in many cases, but, in practice, we need other strong modeling assumptions to make practical progress.