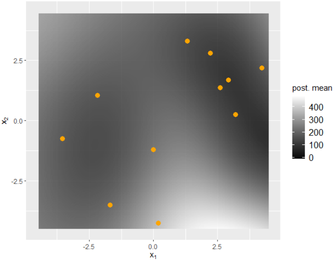
Introduction to Machine Learning

Gaussian Proccesses: Additional Material



Learning goals

- XXX
- XXX

NOTATION

In this chapter

- $(\mathbf{x}_*, y_*)$ denotes one single test observation, excluded from training
- $\mathbf{X}_* \in \mathbb{R}^{n_* \times p}$ contains a set of n_* test observations and
- $\mathbf{y}_* \in \mathbb{R}^{n_* \times p}$ the corresponding outcomes, excluded from training.

Noisy Gaussian Processes

NOISY GAUSSIAN PROCESS

In the above equations we implicitly assumed that we had access to the true function value $f(\mathbf{x})$. In many cases, we only have access to a noisy version thereof

$$y = f(\mathbf{x}) + \epsilon.$$

Assuming additive i.i.d. Gaussian noise, the covariance function becomes

$$\text{Cov}(y^{(i)}, y^{(j)}) = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) + \sigma_n^2 \delta_{ij}$$

where $\delta_{ij} = 1$ if $i = j$. In matrix notation, this becomes

$$\text{Cov}(\mathbf{y}) = \mathbf{K} + \sigma_n^2 \boldsymbol{I} =: \mathbf{K}_y.$$

The σ_n^2 is also called **nugget**.

GP VS. KERNELIZED RIDGE REGRESSION

The predictive function is then

$$\boldsymbol{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\bar{\boldsymbol{f}}_*, \mathrm{Cov}(\bar{\boldsymbol{f}}_*)).$$

with

- $\bar{\boldsymbol{f}}_* = \mathbf{K}_*^T \mathbf{K}_y^{-1} \mathbf{y}$ and
- $\mathrm{Cov}(\bar{\boldsymbol{f}}_*) = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}_y^{-1} \mathbf{K}_*$.

The predicted mean values at the training points $\bar{\boldsymbol{f}} = \boldsymbol{K}\mathbf{K}_y^{-1}\boldsymbol{y}$ are a **linear combination** of the $\boldsymbol{y}$ values.

Note: Predicting the posterior mean corresponds exactly to the predictions obtained by kernelized Ridge regression. However, a GP (as a Bayesian model) gives us much more information, namely a posterior distribution, whilst kernelized Ridge regression does not.

Bayesian Linear Regression as a GP

BAYESIAN LINEAR REGRESSION AS A GP

One example for a Gaussian process is the Bayesian linear regression model covered earlier. For $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \tau^2 \boldsymbol{I})$, the joint distribution of any set of function values

$$f(\mathbf{x}^{(i)}) = \boldsymbol{\theta}^T \mathbf{x}^{(i)} + \epsilon^{(i)}$$

is Gaussian.

The corresponding mean function is $m(\boldsymbol{x}) = \mathbf{0}$ and the covariance function is

$$\begin{aligned} \mathrm{Cov}(f(\boldsymbol{x}), f(\boldsymbol{x}')) &= \mathbb{E}[f(\boldsymbol{x}) f(\boldsymbol{x}')] - \underbrace{\mathbb{E}[f(\boldsymbol{x})]\mathbb{E}[f(\boldsymbol{x}']}_{=0} \\ &= \mathbb{E}[(\boldsymbol{\theta}^T \boldsymbol{x} + \epsilon^{(i)})^T (\boldsymbol{\theta}^T \boldsymbol{x}' + \epsilon^{(i)})] \\ &= \tau^2 \boldsymbol{x}^T \boldsymbol{x}' + \sigma^2 =: k(\boldsymbol{x}, \boldsymbol{x}'). \end{aligned}$$

FEATURE SPACES AND THE KERNEL TRICK

If one relaxes the linearity assumption by first projecting features into a higher dimensional feature space $\mathcal{Z}$ using a basis function $\phi : \mathcal{X} \rightarrow \mathcal{Z}$, the corresponding covariance function is

$$k(\boldsymbol{x}, \boldsymbol{x}') = \tau^2 \phi(\boldsymbol{x})^T \phi(\boldsymbol{x}') + \sigma^2.$$

To get arbitrarily complicated functions, we would have to handle high-dimensional feature vectors $\phi(\boldsymbol{x})$.

Fortunately, all we need to know are the inner products $\phi(\boldsymbol{x})^T \phi(\boldsymbol{x}')$ - the feature vector itself never occurs in calculations.

FEATURE SPACES AND THE KERNEL TRICK

If we can get the inner product directly **without** calculating the infinite feature vectors, we can infer an infinitely complicated model with a **finite amount** of computation. This idea is known as **kernel trick**.

A Gaussian process can be defined by either

- deriving the covariance function explicitly via inner products of evaluations of basis functions or
- choosing a positive definite kernel function (Mercer Kernel) directly, which corresponds - according to Mercer's theorem - to taking inner products in some (possibly infinite) feature space

SUMMARY: GAUSSIAN PROCESS REGRESSION

- Gaussian process regression is equivalent to **kernelized** Bayesian linear regression
- The covariance function describes the shape of the Gaussian process
- With the right choice of covariance function, remarkably flexible models can be built
- But: naive implementations of Gaussian process models scale poorly with large datasets as
  - the kernel matrix has to be inverted / factorized, which is $\mathcal{O}(n^3)$,
  - computing the kernel matrix uses $\mathcal{O}(n^2)$ memory - running out of memory places a hard limit on problem sizes
  - generating predictions is $\mathcal{O}(n)$ for the mean, but $\mathcal{O}(n^2)$ for the variance.

(...so we need special tricks)