REGRESSION II

Machine Learning for Autonomous Robots

Bilal Wehbe
DFKI Bremen

November 30, 2023 – Bremen, Deutschland

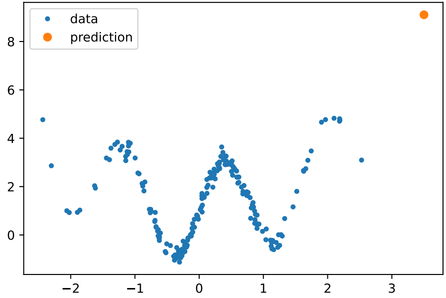
University of Bremen

German Research Center for Artificial Intelligence

Let's recap Gaussian Process Regression

Why Gaussian Process?

- In machine learning we are dealing with data to learn hypotheses
- Data is **measured** from the real-world:
  - Real-world data is unbalanced
  - measurement = noise
- So far the models we've seen do not model the uncertainty at their output.
- Lack of information is modeled using probability distributions at the output of a ML model



Types of Uncertainty

Aleatoric Uncertainty

- Uncertainty that is inherent to the data, e.g., sensor noise.
- Cannot be reduced by adding more data

Epistemic Uncertainty

- Uncertainty produced by the model, e.g., biased hypothesis, lack of data, etc.
- Can be reduced by adding more training data or tuning the model

Gaussian Process

Definition

Let T be a set. $(X_t : t \in T)$ is called a Gaussian process if and only if for every finite set of indices $t_1 \dots t_n \in S$ the set $(X_{t_1} \dots X_{t_n})$ is a multivariate Gaussian.

Reminder

- **one-dimensional:** Gaussian curve with mean μ and variance σ^2
- **multi-dimensional (finite):** multivariate Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix Σ
- **infinite-dimensional extension:** Gaussian Process with mean function $\boldsymbol{\mu}(\boldsymbol{t})$ and covariance function $\boldsymbol{k}(\boldsymbol{t}, \boldsymbol{t}')$

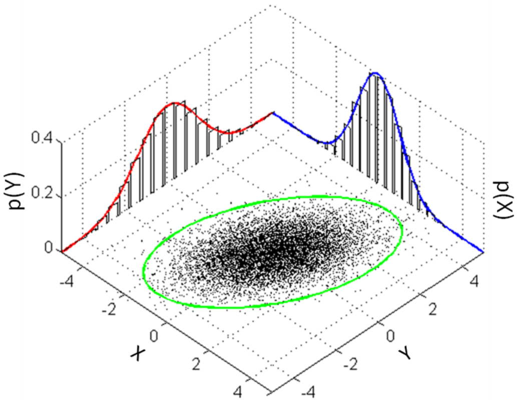
Conditional Distributions

Given two random vectors $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ taken from a probability distribution given as

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{ and } \boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$$

The conditional distribution of $\boldsymbol{X}_1$ given known values for $\boldsymbol{X}_2 = x_2$ is a multivariate normal with $(p(\boldsymbol{X}_1 | \boldsymbol{X}_2 = x_2))$

$$\text{mean vector} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$$
$$\text{cov matrix} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$



https://en.wikipedia.org/wiki/Marginal_distribution#/media/File:MultivariateNormal.png

with $p(x) = \int_y p(x,y)dy$

Gaussian Process

Definition

A Gaussian process can be fully specified it term of its mean function $\mu(\boldsymbol{x})$ and covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ given as

$$f(\boldsymbol{x}) \sim GP(\mu(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}'))$$

Note

The covariance function is a positive definite kernel $\kappa(\boldsymbol{x}, \boldsymbol{x}')$. The mean and covariance functions can be written as

$$\begin{aligned} \mu(\boldsymbol{x}) &= \mathbb{E}[f(\boldsymbol{x})] \\ \kappa(\boldsymbol{x}, \boldsymbol{x}') &= \mathbb{E}[(f(\boldsymbol{x}) - \mu(\boldsymbol{x}))(f(\boldsymbol{x}') - \mu(\boldsymbol{x}'))] \end{aligned}$$

Gaussian Process

Definition

Given an i.i.d. dataset $\mathcal{D} = (\boldsymbol{x}, \boldsymbol{y}) = \{(x_i, y_i), i = 1, \cdots, n\}$ we can define a joint Gaussian for all features using the parameters $\boldsymbol{\mu} = [\mu(x_1), \cdots, \mu(x_n)]$ and $\boldsymbol{K} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Note

$\mu(\cdot)$ can be chosen freely. $\boldsymbol{K}$ is the Gram matrix. Thus the joint Gaussian can be written as

$$p(f|\boldsymbol{X}) = \mathcal{N}(f|\boldsymbol{\mu}, \boldsymbol{K})$$

On account of the training data, the goal is to calculate a posterior in order to predict unseen test samples $(\boldsymbol{x}_*, \boldsymbol{y}_*)$.

Definition

Let $\boldsymbol{y}$ denote the set of training targets y_i. Thus we can the joint distribution as

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{m} \\ \boldsymbol{m}_* \end{pmatrix}, \begin{pmatrix} \boldsymbol{K} & \boldsymbol{K}_* \\ \boldsymbol{K}_*^T & \boldsymbol{K}_{**} \end{pmatrix}\right)$$

where covariance matrices $\boldsymbol{K} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{K}_* = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_{*j})$, $\boldsymbol{K}_{**} = \kappa(\boldsymbol{x}_{*i}, \boldsymbol{x}_{*j})$.

Gaussian Process

We can write the posterior using the rules of conditional Gaussians as

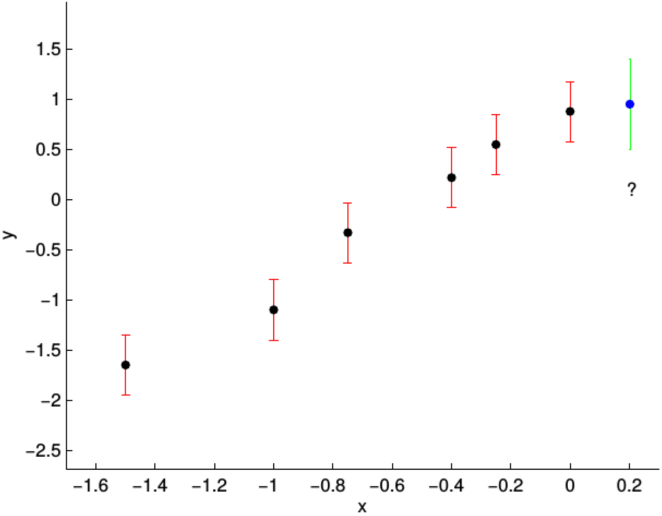
$$
\begin{aligned}
p(\boldsymbol{y}_*|\boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*) \\
\boldsymbol{\mu}_* &= \mu(\boldsymbol{X}_*) + \boldsymbol{K}_*^T \boldsymbol{K}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}) \\
\boldsymbol{\Sigma}_* &= \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}^{-1} \boldsymbol{K}_*
\end{aligned}
$$

where $\boldsymbol{\mu}_*$ and $\boldsymbol{\Sigma}_*$ are the predicted mean and covariance of the test samples.
Derivation can be found in Chapter 4.3 of

Murphy, K. P. Machine learning: a probabilistic perspective. 2012

Gaussian Process Regression

Our given data looks like:



Gaussian Process Regression *Getting the Kernel*

- Assumption: The GP's mean is 0 everywhere.

- Assumption: The GP's mean is 0 everywhere.
- The data is only described by the covariance function, which is a **kernel**.

- Assumption: The GP's mean is 0 everywhere.
- The data is only described by the covariance function, which is a **kernel**.
- We use:

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

  as kernel.

- Assumption: The GP's mean is 0 everywhere.
- The data is only described by the covariance function, which is a **kernel**.
- We use:

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

  as kernel.
- We model measurement noise with a Gaussian.

$$y = f(x) + \mathcal{N}(0, \sigma_n^2)$$

Gaussian Process Regression *Getting the Kernel*

- Let's fold the noise into the Kernel

$$k(x, x') = \sigma_f^2 \exp\left[\frac{-(x - x')^2}{2l^2}\right] + \sigma_n^2 \delta(x, x')$$

- The Kronecker $\delta(x_i, x_j) = 1$ if $i = j$, else 0

- Given the Kernel:

$$k(x, x') = \sigma_f^2 \exp\left[\frac{-(x - x')^2}{2l^2}\right] + \sigma_n^2 \delta(x, x')$$

- And the six obersavations at $x \in \{-1.5, -1, -0.75, -0.4, -0.25, 0\}$
- $\sigma_n = 0.3$, $l = 1$ and $\sigma_f = 1.27$

Calculate two elements on the diagonal and a few off diagonal elements. What do you notice?

- Calculate the Gram matrix K for $x \in \{-1.5, -1, -0.75, -0.4, -0.25, 0\}$ and $\sigma_n = 0.3$, $l = 1$ and $\sigma_f = 1.27$
- Calculate K_* and K_{**}

Hint: use *cdist* from *scipy.spatial.distance*, or *pairwise_distances* from *sklearn.metrics.pairwise*

We obtain:

$$K = \begin{bmatrix} 1.70 & 1.42 & 1.21 & 0.87 & 0.72 & 0.51 \\ 1.42 & 1.70 & 1.56 & 1.34 & 1.21 & 0.97 \\ 1.21 & 1.56 & 1.70 & 1.51 & 1.42 & 1.21 \\ 0.87 & 1.34 & 1.51 & 1.70 & 1.59 & 1.48 \\ 0.72 & 1.21 & 1.42 & 1.59 & 1.70 & 1.56 \\ 0.51 & 0.97 & 1.21 & 1.48 & 1.56 & 1.70 \end{bmatrix}$$

And for our new estimate at x_* we need:

$$K_* = [k(x_*, x_1) \ldots k(x_*, x_n)]$$

and

$$K_{**} = [k(x_*, x_*)]$$

We obtain:

$$K = \begin{bmatrix} 1.70 & 1.42 & 1.21 & 0.87 & 0.72 & 0.51 \\ 1.42 & 1.70 & 1.56 & 1.34 & 1.21 & 0.97 \\ 1.21 & 1.56 & 1.70 & 1.51 & 1.42 & 1.21 \\ 0.87 & 1.34 & 1.51 & 1.70 & 1.59 & 1.48 \\ 0.72 & 1.21 & 1.42 & 1.59 & 1.70 & 1.56 \\ 0.51 & 0.97 & 1.21 & 1.48 & 1.56 & 1.70 \end{bmatrix}$$

And for our new estimate at x_* we need:

$$K_* = \begin{bmatrix} 0.38 & 0.79 & 1.03 & 1.35 & 1.46 & 1.58 \end{bmatrix}$$

and

$$K_{**} = \begin{bmatrix} 1.70 \end{bmatrix}$$

Assumption

For GPs we assume that our data is generated from a sample of a multivariate Gaussian distribution.

$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_*^\intercal \\ K_* & K_{**} \end{bmatrix}\right)$$

But we are interested in $p(y_*|y)$ which follows a Gaussian distribution:
$y_*|y \sim \mathcal{N}(K_*K^{-1}y, K_{**} - K_*K^{-1}K_*^\intercal)$

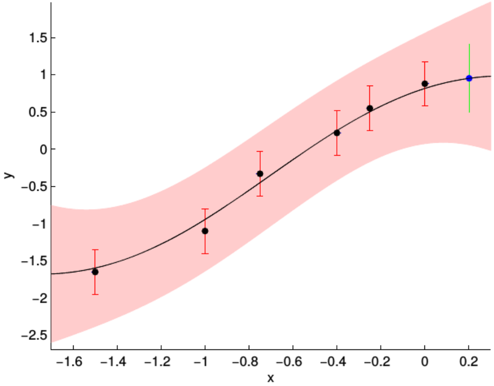
Calculate the mean and variance of y_* using the training samples

$$y = \begin{bmatrix} -1.65 & -1.1 & -0.35 & 0.2 & 0.52 & 0.85 \end{bmatrix}$$

.

Estimated data

For 1000 different values for $x*$ we get:



M.Ebden Gaussian Process for Regression: A Quick Introduction 2008

GPR in the Real World

- How did we choose $\sigma_n = 0.3$, $l = 1$ and $\sigma_f = 1.27$?

GPR in the Real World

- How did we choose $\sigma_n = 0.3$, $l = 1$ and $\sigma_f = 1.27$?
- The reliability of a model depends on the choice of its hyper-parameters, call them $\boldsymbol{\theta} = \{l, \sigma_n, \sigma_f\}$
- The maximum *a posteriori* estimate of $\boldsymbol{\theta}$ is when $p(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{y})$ is highest.

GPR in the Real World

- How did we choose $\sigma_n = 0.3$, $l = 1$ and $\sigma_f = 1.27$?
- The reliability of a model depends on the choice of its hyper-parameters, call them $\boldsymbol{\theta} = \{l, \sigma_n, \sigma_f\}$
- The maximum *a posteriori* estimate of $\boldsymbol{\theta}$ is when $p(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{y})$ is highest.
- According to Bayes' theorem this can be obtained by maximizing

$$\log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) = -\frac{1}{2} y^T \boldsymbol{K}^{-1} y - \frac{1}{2} \log|\boldsymbol{K}| - \frac{n}{2} \log 2\pi$$

GPR in the Real World

- How did we choose $\sigma_n = 0.3$, $l = 1$ and $\sigma_f = 1.27$?
- The reliability of a model depends on the choice of its hyper-parameters, call them $\boldsymbol{\theta} = \{l, \sigma_n, \sigma_f\}$
- The maximum *a posteriori* estimate of $\boldsymbol{\theta}$ is when $p(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{y})$ is highest.
- According to Bayes' theorem this can be obtained by maximizing

$$\log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) = -\frac{1}{2} y^T \boldsymbol{K}^{-1} y - \frac{1}{2} \log|\boldsymbol{K}| - \frac{n}{2} \log 2\pi$$

- The gradient is computed as

$$\frac{\partial}{\partial \theta_j} \log p(\boldsymbol{y}|\boldsymbol{X}) = \frac{1}{2} trace\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^T - \boldsymbol{K}^{-1}) \frac{\partial \boldsymbol{K}}{\partial \theta_j} \right), \qquad \boldsymbol{\alpha} = \boldsymbol{K}^{-1} y$$

Rasmussen, C. and C. Williams Gaussian Processes for Machine Learning. MIT Press 2006

Other Kernels

We can also a combination of Kernels:

Short-term and long-term dynamics:

$$k(x, x') = \sigma_{f_1}^2 \text{exp}\left[\frac{-(x - x')^2}{2l_1^2}\right] + \sigma_{f_2}^2 \text{exp}\left[\frac{-(x - x')^2}{2l_2^2}\right] + \sigma_n^2 \delta(x, x')$$

Periodicity:

$$k(x, x') = \sigma_f^2 \text{exp}\left[\frac{-(x - x')^2}{2l^2}\right] + \text{exp}\{-2\text{sin}^2\left[\nu\pi(x - x')\right]\} + \sigma_n^2 \delta(x, x')$$

Other Kernels

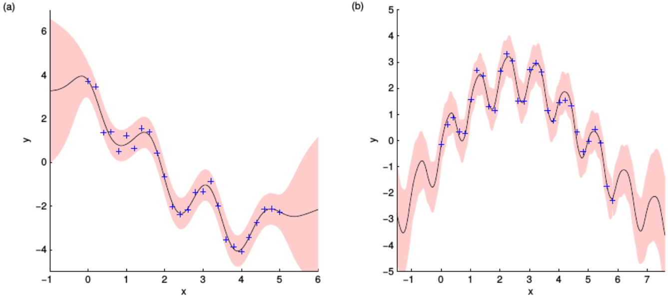
We can also a combination of Kernels:

Short-term and long-term dynamics:

$$k(x, x') = \sigma_{f_1}^2 \exp\left[\frac{-(x - x')^2}{2l_1^2}\right] + \sigma_{f_2}^2 \exp\left[\frac{-(x - x')^2}{2l_2^2}\right] + \sigma_n^2 \delta(x, x')$$

Periodicity:

$$k(x, x') = \sigma_f^2 \exp\left[\frac{-(x - x')^2}{2l^2}\right] + \exp\{-2\sin^2[\nu\pi(x - x')]\} + \sigma_n^2 \delta(x, x')$$



Further Readings/Tutorials

- https://scikit-learn.org/stable/modules/gaussian_process.html
- https://nbviewer.ipython.org/github/SheffieldML/notebook/blob/master/GPy/index.ipynb