Machine Learning

Gaussian Processes

Wojciech Czech

January 15, 2024

Institute of Computer Science

Table of contents

Learning hyperparameters

Fitting Gaussian process kernel

- Create parametrized kernel function by addition or multiplication of basic kernels to reflect different characteristics of the data:
  a. Long term smooth change
  b. Seasonality
  c. Short term irregularities
  d. Medium term irregularities
  e. Precision of white noise
  f. Length scale of correlations
- Learn hyperparameters of a kernel function by maximizing log likelihood function

Parametrized kernel functions

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_0 \exp\left(-\frac{\theta_1}{2}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2\right) + \theta_2 + \theta_3 \boldsymbol{x}_i^T \boldsymbol{x}_j \tag{1}$$

$$\begin{aligned} k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_0 \exp\left(-\frac{\theta_1}{2}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2\right) + \\ \theta_2 \exp\left(-\frac{2}{\theta_3}\sin^2\left(\pi\frac{\|\boldsymbol{x}_a - \boldsymbol{x}_b\|}{\theta_4}\right)\right) \exp\left(-\frac{\theta_5}{2}\left\|\boldsymbol{x}_a - \boldsymbol{x}_b\right\|^2\right) + \\ \theta_6\left(1 + \frac{\left\|\boldsymbol{x}_a - \boldsymbol{x}_b\right\|^2}{2\theta_7\theta_8}\right)^{-\theta_7} \end{aligned} \tag{2}$$

Log Likelihood function

$$t(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \tag{3}$$

$$p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{t}|\boldsymbol{\mu}, \Sigma_{\boldsymbol{\theta}}) = \mathcal{N}(\boldsymbol{t}|m(\boldsymbol{X}), \boldsymbol{C}_{\boldsymbol{\theta}}) \tag{4}$$

$$m(\boldsymbol{X}) = \boldsymbol{0} \tag{5}$$

$$\boldsymbol{\theta}_{ML} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \{p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta})\} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \{-\ln p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta})\} \tag{6}$$

$$\ln p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2} \ln |\boldsymbol{C}_{\boldsymbol{\theta}}| - \frac{1}{2} \boldsymbol{t}^T \boldsymbol{C}_{\boldsymbol{\theta}}^{-1} \boldsymbol{t} - \frac{n}{2} \ln (2\pi) \tag{7}$$

Maximum likelihood solution

$$\frac{\partial}{\partial \theta_i} \ln p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2} Tr\left(\boldsymbol{C}_{\boldsymbol{\theta}}^{-1} \frac{\partial \boldsymbol{C}_{\boldsymbol{\theta}}}{\partial \theta_i}\right) + \frac{1}{2} \boldsymbol{t}^T \boldsymbol{C}_{\boldsymbol{\theta}}^{-1} \frac{\partial \boldsymbol{C}_{\boldsymbol{\theta}}}{\partial \theta_i} \boldsymbol{C}_{\boldsymbol{\theta}}^{-1} \boldsymbol{t} \tag{8}$$

$$\frac{\partial}{\partial \theta_i} \ln p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta}) = 0 \tag{9}$$

$\ln p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta})$ is non-convex and can have multiple local maxima

Practical approach

Fit the parameters $\boldsymbol{\theta}$ by maximizing likelihood $p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta})$ of the Gaussian process distribution based on observed data $(\boldsymbol{X}, \boldsymbol{t})$.

1. Sum all kernels to single parametrized kernel containing all signal characteristics
2. Divide observed data into mini-batches
3. Define log likelihood function
4. Use *Adam Optimizer*[3] to minimize negative log likelihood (run optimization on all batches)
5. Use fitted kernel, mean function and observed data to make posterior predictions on unobserved data

Adam optimizer

- Stochastic gradient descent method based on adaptive estimation of first-order and second-order moments (**ADA**ptive **M**oment estimation)
- `tf.keras.optimizers.Adam`
- State-of-the-art optimization method well-suited to large parameter spaces
- Adam combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems [2]
- Adam is relatively easy to configure where the default configuration parameters do well on most problems [2]

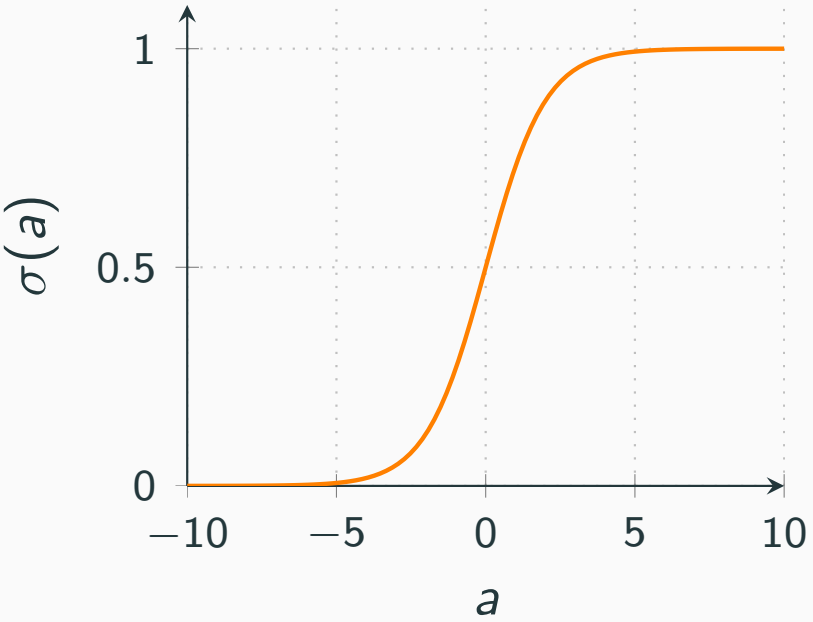
Gaussian process for classification

Probabilistic two-class classification problem

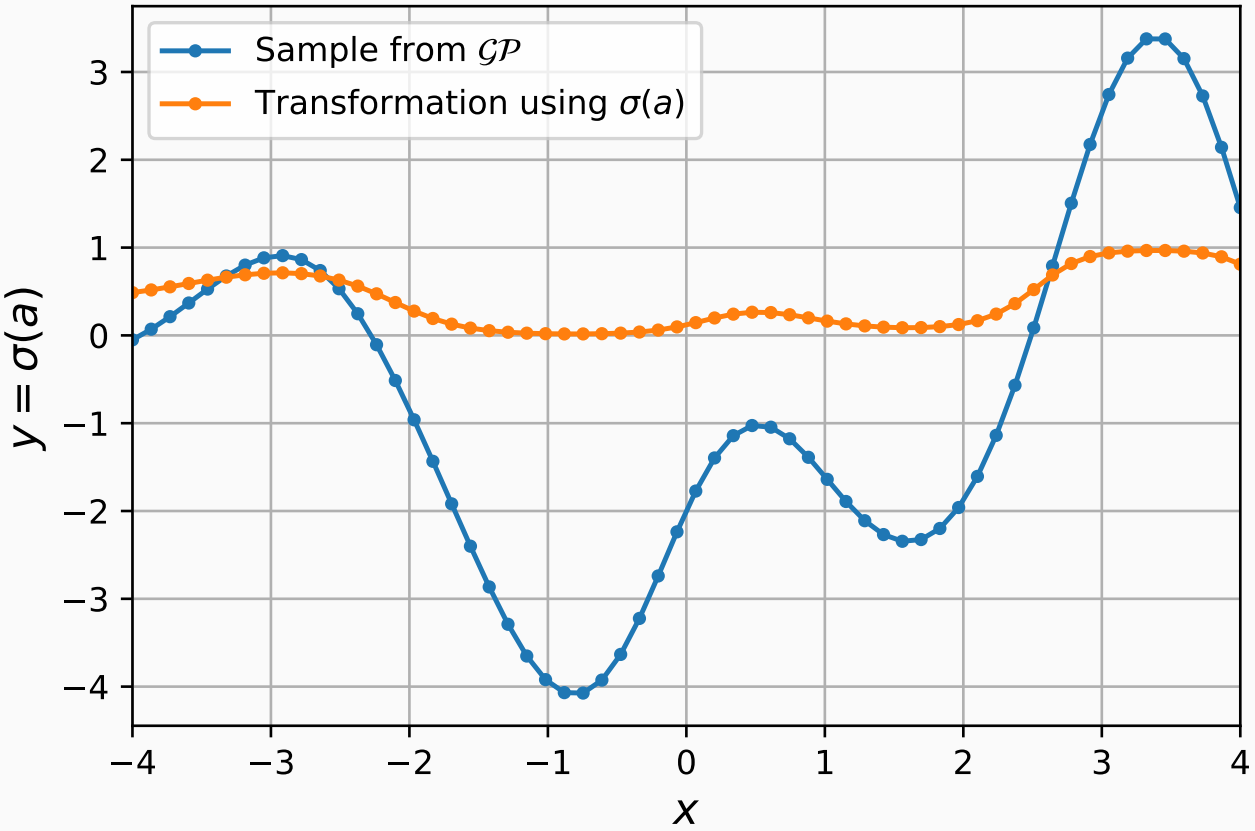
- Target: $t \in \{0, 1\}$, which follows Bernoulli distribution
- Define Gaussian process over function $a(\boldsymbol{x})$, then transform function using logistic sigmoid $y = \sigma(a)$, where:

$$\sigma(a) = \frac{1}{1 + \exp(-a)} \qquad (10)$$

- We obtain non-Gaussian stochastic process over functions $y(\boldsymbol{x})$, $y \in (0, 1)$



Logistic sigmoid transformation



$$p(t|a) = \sigma(a)^t(1 - \sigma(a))^{(1-t)} \tag{11}$$

Classification problem

- Training data
  a) Target values: $\boldsymbol{t}_n = [t_1, \ldots, t_n]^T$, $t_i \in \{0, 1\}$
  b) Input values: $\boldsymbol{X}_n = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$
- Test data: $\boldsymbol{x}_{n+1}$ with target variable t_{n+1}
- Determine predictive distribution $p(t_{n+1}|\boldsymbol{t}_n, \boldsymbol{X}_n, \boldsymbol{x}_{n+1})$
- Gaussian process prior over vector $\boldsymbol{a}_{n+1} = [a(\boldsymbol{x}_1), \ldots, a(\boldsymbol{x}_{n+1})]$

$$p(\boldsymbol{a}_{n+1}) = \mathcal{N}(\boldsymbol{a}_{n+1}|\boldsymbol{0}, \boldsymbol{C}_{n+1}) \tag{12}$$

$\boldsymbol{C}_{n+1} \in \mathbb{R}^{(n+1)\times(n+1)}$ - covariance matrix

$$[\boldsymbol{C}_{n+1}]_{i,j} = C(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \nu\delta_{ij} \tag{13}$$

Classification problem - predictive distribution

- Kernel function depends on vector of parameters $\boldsymbol{\theta}$
- Non-Gaussian stochastic process over $\boldsymbol{t}_{n+1}$
- Conditioning on training data (predictive distribution):

$$p(t_{n+1} = 1|\boldsymbol{t}_n) = \int p(t_{n+1} = 1|a_{n+1})p(a_{n+1}|\boldsymbol{t}_n)da_{n+1} \tag{14}$$

$$p(t_{n+1} = 1|a_{n+1}) = \sigma(a_{n+1}) \tag{15}$$

- Predictive distribution cannot be derived analytically

Laplace approximation

Laplace approximation

Laplace approximation

Gaussian approximation to a probability density defined over a set of continuous variables (see [1]).

$$p(z) = \frac{1}{Z} f(z) \tag{16}$$

$$Z = \int f(z) dz \tag{17}$$

Single continuous variable

Find Gaussian approximation $q(z)$, which is centered on a mode of the distribution $p(z)$.

Mode of $p(z)$

- Find a point z_0, so that $f'(z_0) = 0$
- Taylor expansion of $u(z) = \ln f(z)$

$$u(z) \simeq u(z_0) + \underbrace{u'(z_0)(z - z_0)}_{0} + u''(z_0)(z - z_0)^2 \tag{18}$$

$$f(z) \simeq f(z_0) \exp\left\{-\frac{A}{2}(z - z_0)^2\right\} \tag{19}$$

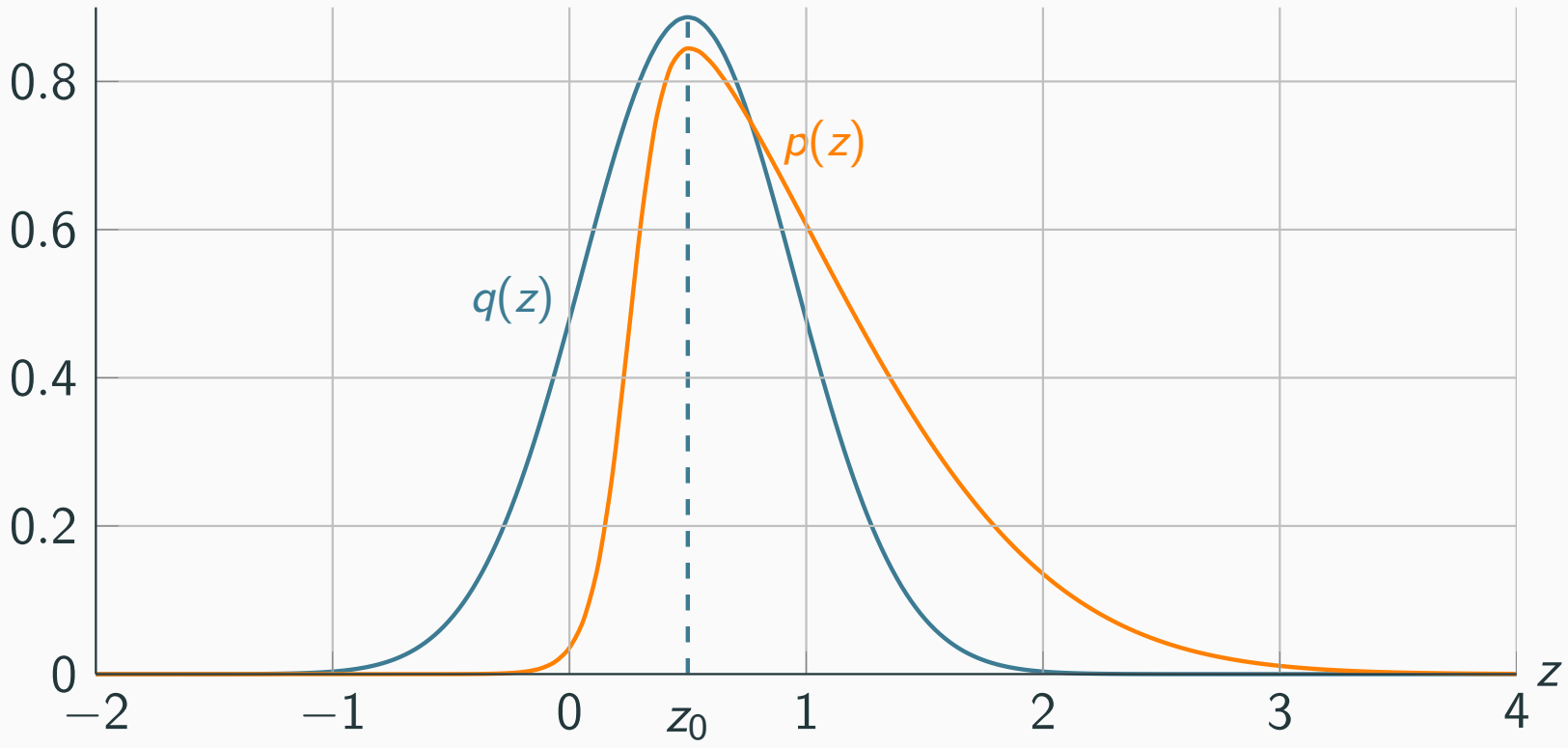
$$A = -u''(z_0) \tag{20}$$

- Normalized distribution $q(z)$

$$q(z) = \left(\frac{A}{2\pi}\right)^{1/2} \exp\left\{-\frac{A}{2}(z - z_0)^2\right\} \tag{21}$$

- Gaussian approximation will only be well defined if $A > 0$ (maximum)

Example 1: Visualization of the Laplace approximation



Multidimensional Laplace approximation

$$p(\boldsymbol{z}) = \frac{1}{Z} f(\boldsymbol{z}) \tag{22}$$

$$Z = \int f(\boldsymbol{z}) d\boldsymbol{z} \tag{23}$$

- Find a point $\boldsymbol{z}_0$, so that $\nabla f(\boldsymbol{z}_0) = \boldsymbol{0}$
- Taylor expansion of $u(\boldsymbol{z}) = \ln f(\boldsymbol{z})$

$$u(\boldsymbol{z}) \simeq u(\boldsymbol{z}_0) - \frac{1}{2}(\boldsymbol{z} - \boldsymbol{z}_0)^T \boldsymbol{A} (\boldsymbol{z} - \boldsymbol{z}_0) \tag{24}$$

- Hessian matrix $\boldsymbol{A} \in \mathbb{R}^{m \times m}$

$$\boldsymbol{A} = -\nabla^2 u(\boldsymbol{z}) \Big|_{\boldsymbol{z}=\boldsymbol{z}_0} = -\Delta \ln f(\boldsymbol{z}) \Big|_{\boldsymbol{z}=\boldsymbol{z}_0} \tag{25}$$

Multidimensional Laplace approximation - taking exponential

$$f(\boldsymbol{z}) \simeq f(\boldsymbol{z}_0) \exp\left\{-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}_0)^T \boldsymbol{A}(\boldsymbol{z}-\boldsymbol{z}_0)\right\} \tag{26}$$

- Normalized distribution $q(\boldsymbol{z})$

$$q(\boldsymbol{z}) = \frac{|\boldsymbol{A}|^{1/2}}{(2\pi)^{m/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}_0)^T \boldsymbol{A}(\boldsymbol{z}-\boldsymbol{z}_0)\right\} \tag{27}$$

- Gaussian approximation will be only well defined if $\boldsymbol{A}$ is positive definite (stationary point $\boldsymbol{z_0}$ must be local maximum)

Laplace approximation - summary

1. Find mode $\boldsymbol{z}_0$ (optimization algorithm)
2. Evaluate Hessian matrix at $\boldsymbol{z}_0$
3. In case of multi-modal distribution $p(\boldsymbol{z})$ obtain multiple Laplace approximations
4. Laplace approximation works well when the number of data points is large (central limit theorem)
5. Approximation is local - distribution at specific value of variable

Laplace approximation for Gaussian process classification

Predictive distribution for classification

$$p(t_{n+1} = 1|\boldsymbol{t}_n) = \int \sigma(a_{n+1}) p(a_{n+1}|\boldsymbol{t}_n) da_{n+1} \tag{28}$$

- Approximation of the convolution of a logistic sigmoid with a Gaussian distribution [1]:

$$\int \sigma(a) \mathcal{N}(a|\mu, \sigma^2) da \simeq \sigma(\kappa(\sigma^2)\mu) \tag{29}$$

$$\kappa(\sigma^2) = (1 + \pi\sigma^2/8)^{-1/2} \tag{30}$$

- Gaussian approximation of posterior distribution $p(a_{n+1}|\boldsymbol{t}_n)$ needed

Posterior distribution and Bayes' theorem

$$
\begin{aligned}
p(a_{n+1}|\boldsymbol{t}_n) &= \int p(a_{n+1}, \boldsymbol{a}_n|\boldsymbol{t}_n) d\boldsymbol{a}_n \\
&= \frac{1}{p(\boldsymbol{t}_n)} \int p(a_{n+1}, \boldsymbol{a}_n) p(\boldsymbol{t}_n|a_{n+1}, \boldsymbol{a}_n) d\boldsymbol{a}_n \\
&= \frac{1}{p(\boldsymbol{t}_n)} \int p(a_{n+1}|\boldsymbol{a}_n) p(\boldsymbol{a}_n) p(\boldsymbol{t}_n|\boldsymbol{a}_n) d\boldsymbol{a}_n \\
&= \int \underbrace{p(a_{n+1}|\boldsymbol{a}_n)}_{\mathcal{GP}\,\text{regression}} \underbrace{p(\boldsymbol{a}_n|\boldsymbol{t}_n)}_{\Delta\,\text{approximation}} d\boldsymbol{a}_n
\end{aligned}
\tag{31}
$$

Gaussian process regression

$$p(a_{n+1}|\boldsymbol{a}_n) = \mathcal{N}(a_{n+1}|\boldsymbol{k}^T\boldsymbol{C}_n^{-1}\boldsymbol{a}_n, c - \boldsymbol{k}^T\boldsymbol{C}_n^{-1}\boldsymbol{k}_n) \tag{32}$$

Laplace approximation of posterior distribution $p(\boldsymbol{a}_n|\boldsymbol{t}_n)$

- Prior $p(\boldsymbol{a}_n)$ is given by zero-mean Gaussian process with covariance matrix $\boldsymbol{C}_n$
- Assumption about independence of training data points gives:

$$p(\boldsymbol{a}_n|\boldsymbol{t}_n) = \prod_{i=1}^{n} \sigma(a_i)^{t_i}(1-\sigma(a_i))^{1-t_i} = \prod_{i=1}^{n} e^{a_i t_i}\sigma(-a_i) \tag{33}$$

- Taylor expansion of $\ln p(\boldsymbol{a}_n|\boldsymbol{t}_n)$:

$$\begin{aligned} \Psi(\boldsymbol{a}_n) &= \ln p(\boldsymbol{a}_n) + \ln p(\boldsymbol{a}_n|\boldsymbol{t}_n) \\ &\simeq -\frac{1}{2}\boldsymbol{a}_n^T\boldsymbol{C}_n^{-1}\boldsymbol{a}_n - \frac{n}{2}\ln(2\pi) - \frac{1}{2}|\boldsymbol{C}_n| + \boldsymbol{t}_n^T\boldsymbol{a}_n \\ &\quad - \sum_{i=1}^{n} \ln\left(1+e^{a_i}\right) + const. \end{aligned} \tag{34}$$

Laplace approximation of posterior distribution $p(\boldsymbol{a}_n|\boldsymbol{t}_n)$

- Find mode of posterior distribution $p(\boldsymbol{a}_n|\boldsymbol{t}_n)$
- Gradient of $\Psi(\boldsymbol{a}_n)$:

$$\nabla\Psi(\boldsymbol{a}_n) = \boldsymbol{t}_n - \boldsymbol{\sigma}_n - \boldsymbol{C}_n^{-1}\boldsymbol{a}_n \tag{35}$$

$$\boldsymbol{\sigma} = [\sigma(a(\boldsymbol{x}_1)), \ldots, \sigma(a(\boldsymbol{x}_n))] = [\sigma(a_1), \ldots, \sigma(a_n)]$$

- Laplacian of $\Psi(\boldsymbol{a}_n)$:

$$\nabla^2\Psi(\boldsymbol{a}_n) = -\boldsymbol{W}_n - \boldsymbol{C}_n^{-1} \tag{36}$$

$$\boldsymbol{W}_n = diag(\sigma(a_i)(1 - \sigma(a_i))$$

- Hessian matrix $\boldsymbol{A} = -\nabla^2\Psi(\boldsymbol{a}_n)$ is positive definite - posterior distribution has a single global maximum

Newton-Raphson solution for mode

$$\boldsymbol{a}_n^\star = \boldsymbol{C}_n(\boldsymbol{t}_n - \boldsymbol{\sigma}_n) \tag{37}$$

- Gaussian approximation to the posterior $p(\boldsymbol{a}_n|\boldsymbol{t}_n)$ is given by:

$$q(\boldsymbol{a}_n) = \mathcal{N}(\boldsymbol{a}_n|\boldsymbol{a}_n^\star, \boldsymbol{A}^{-1}) \tag{38}$$

Gaussian process classification - summary

Goal: estimate predictive distribution:

$$p(t_{n+1} = 1|\boldsymbol{t}_n) = \int \sigma(a_{n+1}) p(a_{n+1}|\boldsymbol{t}_n) da_{n+1} \tag{39}$$

1. Resolve Gaussian process regression:

$$p(a_{n+1}|\boldsymbol{a}_n) = \mathcal{N}(a_{n+1}|\boldsymbol{k}^T \boldsymbol{C}_n^{-1} \boldsymbol{a}_n, c - \boldsymbol{k}^T \boldsymbol{C}_n^{-1} \boldsymbol{k}_n) \tag{40}$$

2. Find Laplace approximation of posterior distribution $p(\boldsymbol{a}_n|\boldsymbol{t}_n)$

$$q(\boldsymbol{a}_n) = \mathcal{N}(\boldsymbol{a}_n|\boldsymbol{a}_n^\star, \boldsymbol{A}^{-1}) \tag{41}$$

3. Use Bayes' theorem (Eq. 31) and equations Eq. 32, Eq. 38 to calculate:

$$p(a_{n+1}|\boldsymbol{t}_n) = \int p(a_{n+1}|\boldsymbol{a}_n) p(\boldsymbol{a}_n|\boldsymbol{t}_n) d\boldsymbol{a}_n \tag{42}$$

4. Use equations Eq. 29 and Eq. 30 to calculate convolution given by Eq. 39 and get predictive distribution

Gaussian process classification - kernel parameters

- Determine kernel function parameters $\boldsymbol{\theta}$ using maximum likelihood approach for $p(\boldsymbol{t}_n|\boldsymbol{\theta})$

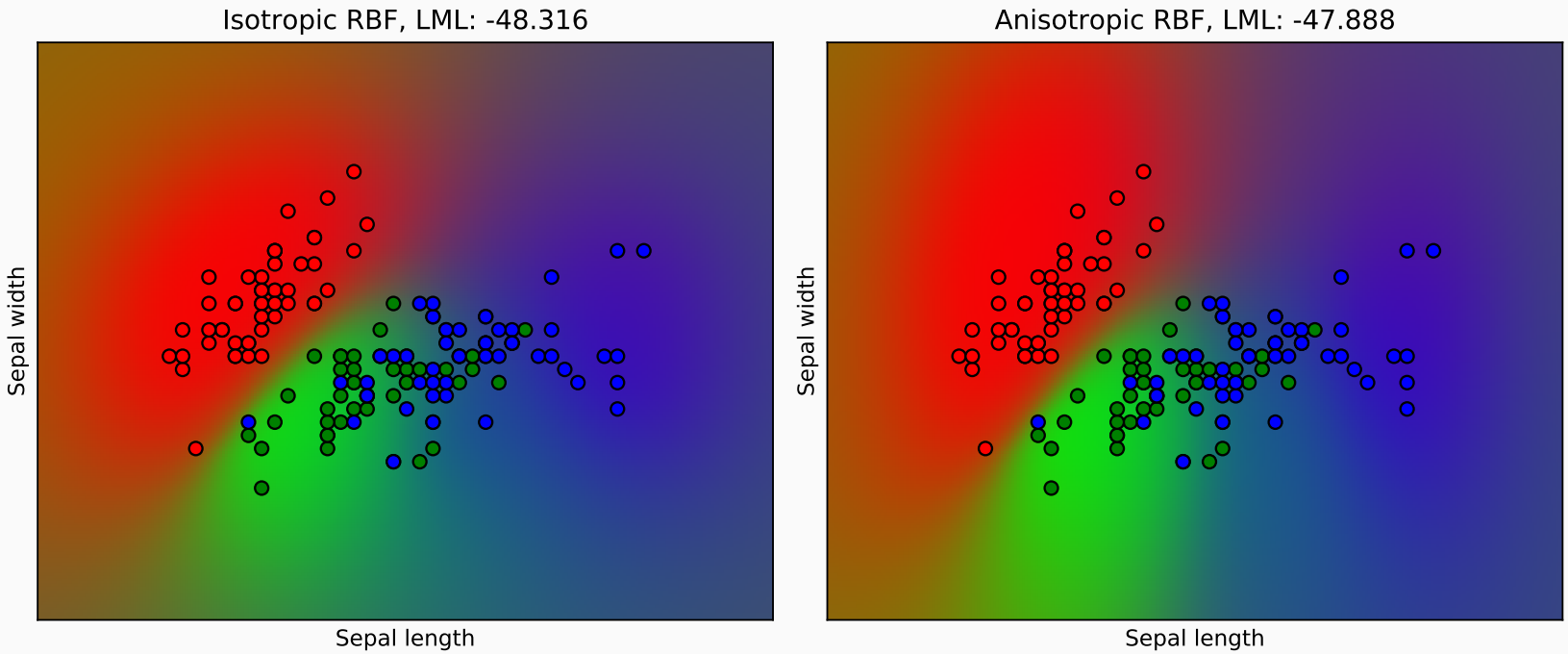
$$p(\boldsymbol{t}_n|\boldsymbol{\theta}) = \int p(\boldsymbol{t}_n|\boldsymbol{a}_n)p(\boldsymbol{a}_n|\boldsymbol{\theta})d\boldsymbol{a}_n \tag{43}$$

- Laplacian approximation can be used again
- Approximation of log likelihood function

$$\ln p(\boldsymbol{t}_n|\boldsymbol{\theta}) = \Psi(\boldsymbol{a}_n^\star) - \frac{1}{2}\ln|\boldsymbol{W}_n + \boldsymbol{C}_n^{-1}| + \frac{n}{2}\ln 2\pi \tag{44}$$

- Use standard nonlinear optimization algorithms to determine a value for $\boldsymbol{\theta}$

Example 2: Iris classification using Gaussian process



Example 2: Configuration

- Gaussian process classification (GPC) based on Laplace approximation (Scikit Learn)
- Two types of kernels tested: Isotropic RBF and Anisotropic RBF
- Optimal kernel parameters θ are selected during fitting
- Maximal log-likelihood value reported
- Probabilities represented using RGB heatmaps

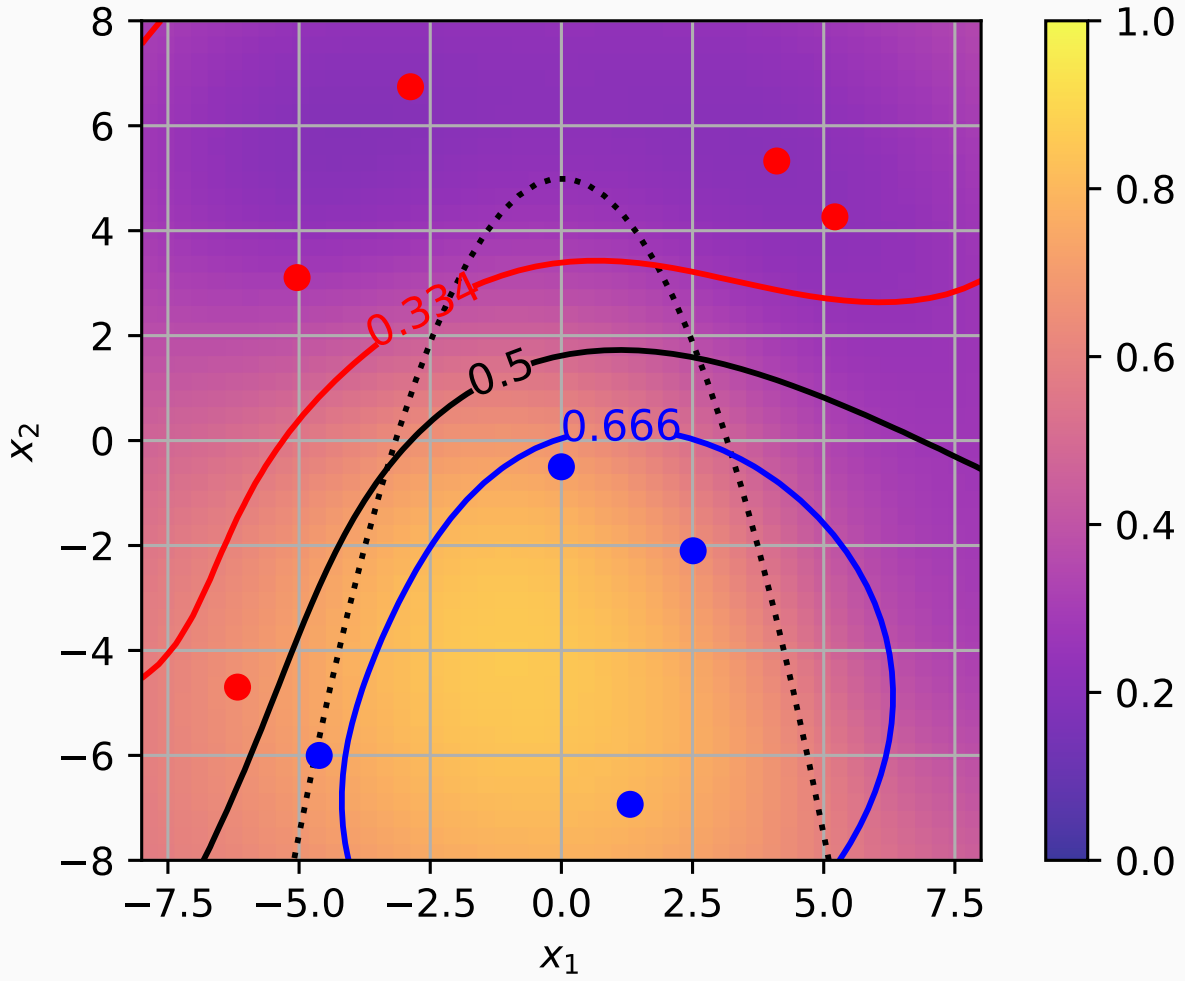
Anisotropic RBF kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{x}')\right) \tag{45}$$

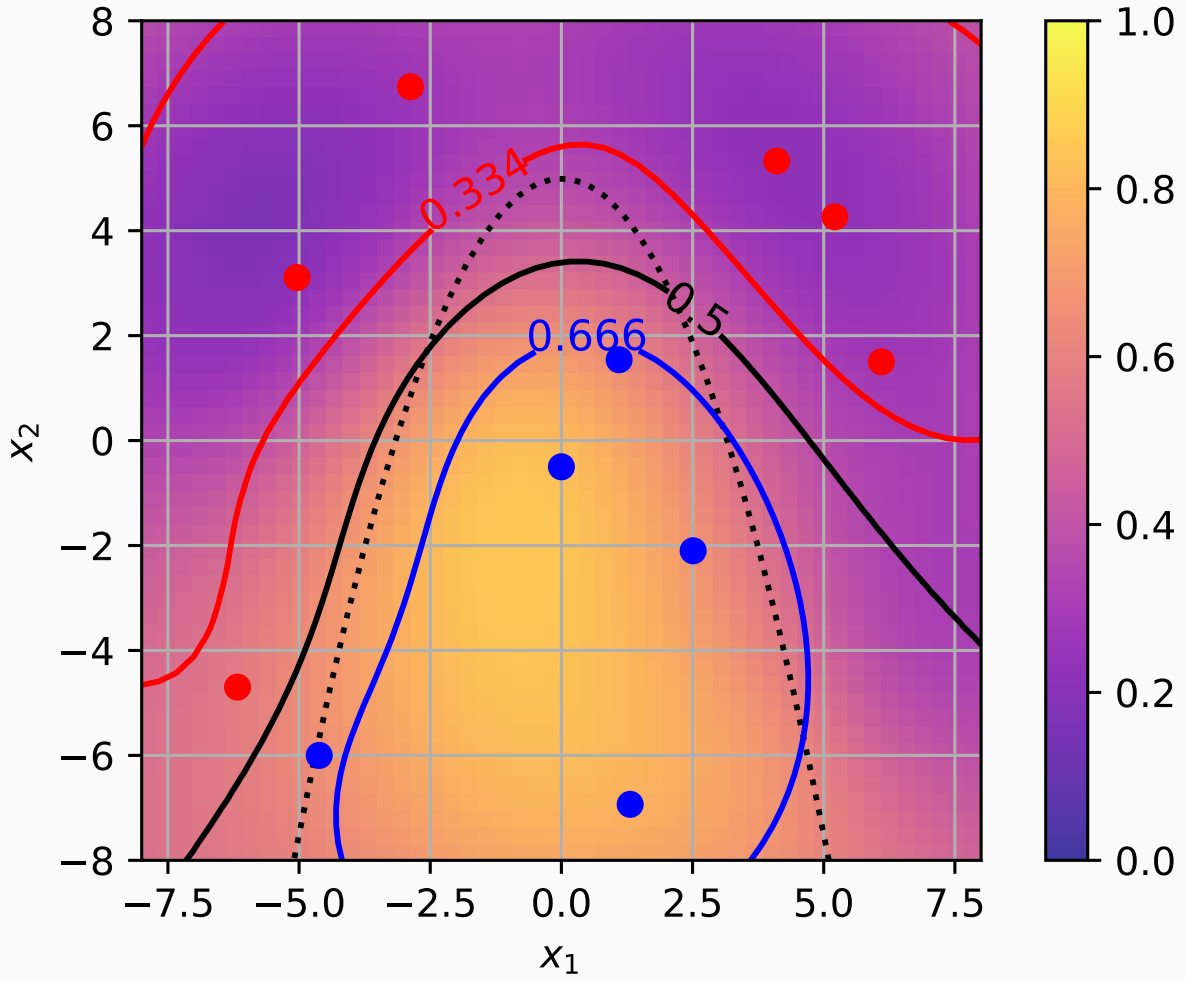
$$\boldsymbol{M} = diag(\boldsymbol{\theta}) \tag{46}$$

$$\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^{m \times 1}$$
$$\boldsymbol{\theta} \in \mathbb{R}^{m \times 1}$$
$$\boldsymbol{M} \in \mathbb{R}^{m \times m}$$

Example 3: RBF kernel iso-probability lines



Example 3: RBF kernel iso-probability lines



Automatic relevance determination

ARD basics

- Construct special kernel function with parameter for each input variable
- Optimize parameters on training set using maximum likelihood
- Get information about relative importance of each dimension in training set
- Detect input variables that have little effect on the predictive distribution (discard them)

ARD kernels

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_0 \exp\left\{-\frac{1}{2}\sum_{k=1}^{D}\eta_k(x_{ik} - x_{jk})^2\right\} \tag{47}$$

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_0 \exp\left\{-\frac{1}{2}\sum_{k=1}^{D}\eta_k(x_{ik} - x_{jk})^2\right\} + \theta_1 + \theta_2\sum_{k=1}^{D}x_{ik}x_{jk} \tag{48}$$

Gaussian processes - summary

Advantages

- Parameter tuning within the framework (easier to select hyperparameters than in case of linear regression or neural networks)
- Predictions are probabilistic (confidence intervals)
- Flexibility in selecting prior (kernel)
- Essentially non-parametric model (black box)
- Convenient probabilistic framework well suited for many tasks

Disadvantages

- Expensive computationally (optimization of parameter space)
- $\mathcal{O}(n^3)$
- Gaussian processes are not sparse - they use the whole samples/features information to build prediction model
- Gaussian processes lose efficiency in high dimensional spaces ($> 10^5$)

References


C. M. Bishop.
***Pattern Recognition and Machine Learning*.**
Springer, 2006.


J. Brownlee.
***Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*, Accessed January, 2022.**
Available at `https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/`.


D. P. Kingma and J. Ba.
Adam: A method for stochastic optimization.
arXiv preprint arXiv:1412.6980, 2014.