FE540 금융공학 인공지능 및 기계학습

Gaussian Processes

Kee-Eung Kim
Department of Computer Science
KAIST

Linear Regression Revisited

- Consider Linear Regression with basis functions $\boldsymbol{\phi}(\mathbf{x})$:

$$y(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x})$$
$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

- Matrix notation of training input points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ and their function values $y_n = y(\mathbf{x}_n)$ using design matrix $\Phi_{nk} = \phi_k(\mathbf{x}_n)$

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{w}$$

- Since $\mathbf{w}$ is Gaussian, $\mathbf{y}$ is Gaussian with:

$$E[\mathbf{y}] = \boldsymbol{\Phi} E[\mathbf{w}] = 0$$
$$Cov[\mathbf{y}] = E[\mathbf{y}\mathbf{y}^\top] = \boldsymbol{\Phi}^\top E[\mathbf{w}\mathbf{w}^\top]\boldsymbol{\Phi} = \frac{1}{\alpha}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top \equiv \mathbf{K}$$

  - K is the "Kernel" matrix: $K_{nm} = k(\mathbf{x}_n, \mathbf{x}_m) = \dfrac{1}{\alpha}\boldsymbol{\phi}(\mathbf{x}_n)^\top \boldsymbol{\phi}(\mathbf{x}_m)$

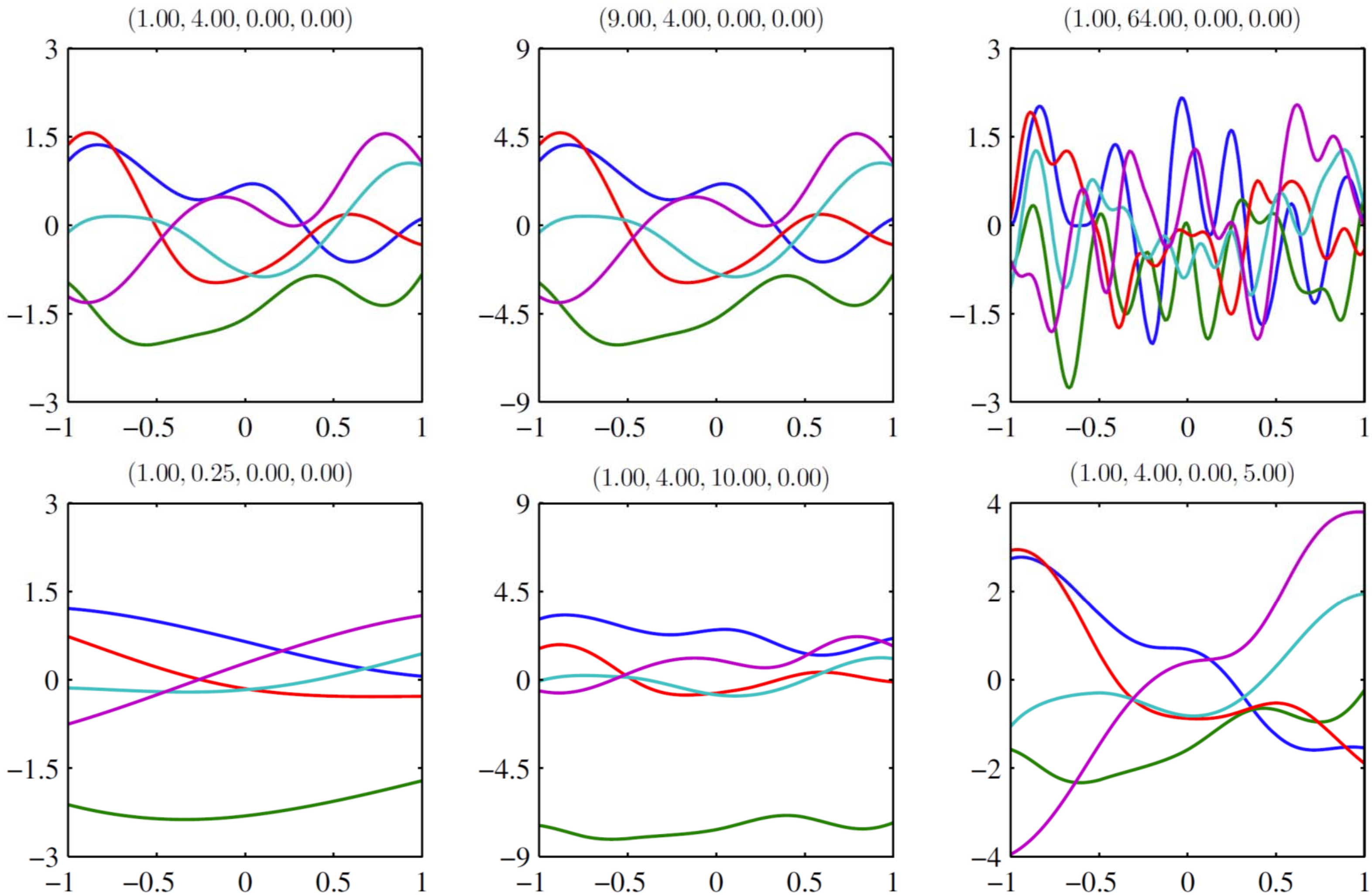
Gaussian Processes

- GP is defined as a probability distribution over functions $y(\mathbf{x})$ such that the set of function values evaluated at input points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ is jointly Gaussian

- Instead of specifying the basis functions $\phi(\mathbf{x})$, we directly specify the Kernel matrix, e.g.
  - Inner-Product:
  $$k_{IP}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$$
  - Squared Exponential:
  $$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp(-\tfrac{1}{2}\|\mathbf{x} - \mathbf{x}'\|^2/\ell^2)$$
  - Automatic Relevance Determination:
  $$k_{ARD}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp(-\tfrac{1}{2}(x_d - x'_d)^2/\ell_d^2)$$
  - Mix and match:
  $$k(\mathbf{x}, \mathbf{x}') = \alpha k_1(\mathbf{x}, \mathbf{x}') + \beta k_2(\mathbf{x}, \mathbf{x}')$$

Samples from GP

$$k(\mathbf{x}, \mathbf{x}') = \theta_0 \exp\left[-\frac{\theta_1}{2}\|\mathbf{x} - \mathbf{x}'\|^2\right] + \theta_2 + \theta_3 \mathbf{x}^\top \mathbf{x}'$$



GP Regression

- From the definition of a GP

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$$

- Assume Gaussian noise in the target values

$$p(t_n|y_n) = \mathcal{N}(t_n|y_n, \beta^{-1})$$
$$p(\mathbf{t}|\mathbf{y}) = \mathcal{N}(\mathbf{t}|\mathbf{y}, \beta^{-1}\mathbf{I}_N)$$

- Marginal distribution of $\mathbf{t}$:

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{y})p(\mathbf{y})d\mathbf{y} = \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C}) \qquad \mathbf{C} = \mathbf{K} + \beta^{-1}\mathbf{I}_N$$

- Given test input point $\mathbf{x}_{N+1}$, dist. of $\mathbf{t}_{N+1} = [t_1, \ldots, t_N, t_{N+1}]$

$$p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^\top & c \end{bmatrix}$$

$$k_n = k(\mathbf{x}_n, \mathbf{x}_{N+1})$$
$$c = k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \beta^{-1}$$

GP Regression

- Given test input point $\mathbf{x}_{N+1}$, dist. of $\mathbf{t}_{N+1} = [t_1, \ldots, t_N, t_{N+1}]$

$$p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^\top & c \end{bmatrix}$$

$$k_n = k(\mathbf{x}_n, \mathbf{x}_{N+1})$$

$$c = k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \beta^{-1}$$

- Prediction of t_{N+1} at the input point $\mathbf{x}_{N+1}$ is done by computing the conditional distribution $p(t_{N+1}|\mathbf{t}_N)$

$$p(t_{N+1}|\mathbf{t}_N) = \mathcal{N}(t_{N+1}|m(\mathbf{x}_{N+1}), \sigma^2(\mathbf{x}_{N+1}))$$

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^\top C_N^{-1}\mathbf{t}$$

$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^\top C_N^{-1}\mathbf{k}$$

  - Computational complexity: $O(N^3)$ for training, $O(N^2)$ for test

Hyperparameter Learning

- Tuning hyperparameters is essential for GP
  - Maximize log likelihood (MLE)
  - Maximize log posterior using prior $p(\boldsymbol{\theta})$: a little harder

- Log likelihood of training data:

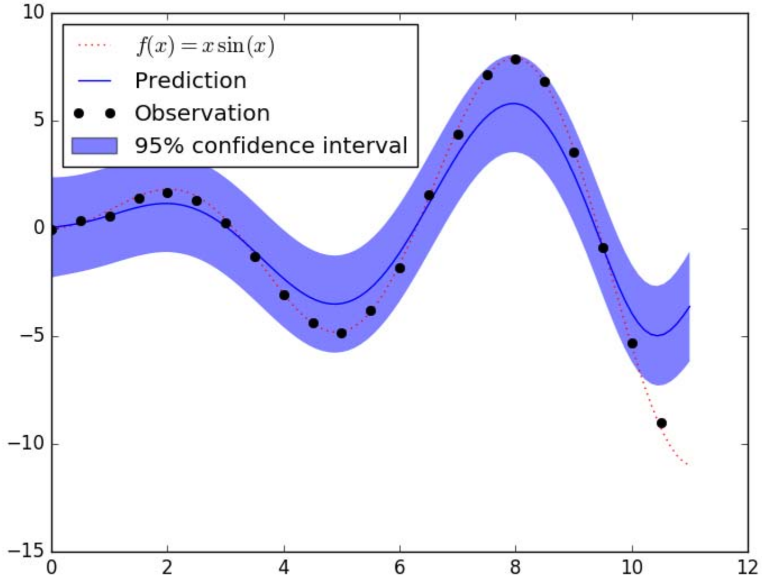
$$\log p(\mathbf{t}|\boldsymbol{\theta}) = \log \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C}) = -\frac{1}{2}\log|C_N| - \frac{1}{2}\mathbf{t}^\top C_N^{-1}\mathbf{t} - \frac{N}{2}\log 2\pi$$

- Gradient:

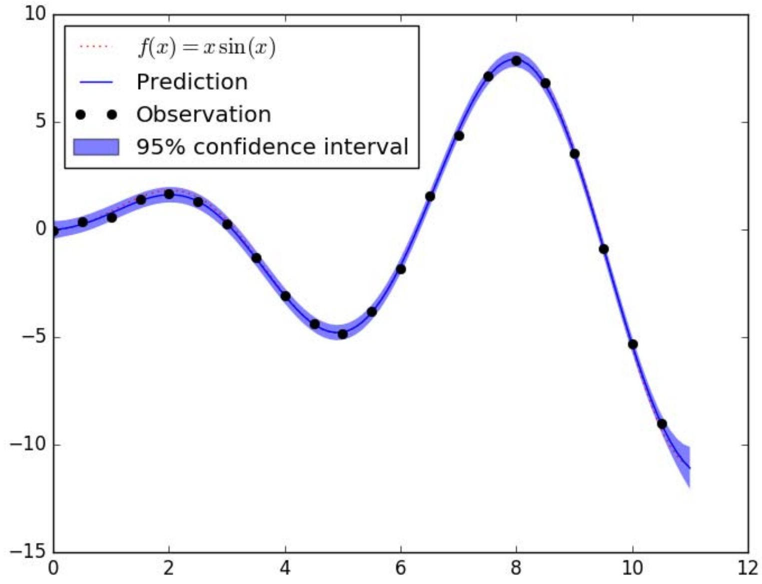
$$\frac{\partial}{\partial\theta}\log p(\mathbf{t}|\boldsymbol{\theta}) = -\frac{1}{2}\mathrm{tr}\left(C_N^{-1}\frac{\partial C_N}{\partial\theta}\right) + \frac{1}{2}\mathbf{t}^\top C_N^{-1}\frac{\partial C_N}{\partial\theta}C_N^{-1}\mathbf{t}$$

GP Regression: Example

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|^2/\ell^2)$$



Before hyperparameter learning



After hyperparameter learning

GP Classification

- Assume binary classification problem: $t \in \{0, 1\}$

- Define a GP over activation function $a(\mathbf{x})$ and then transform it via sigmoid $y = \sigma(a)$, so that we have Bernoulli distribution over the target variable

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

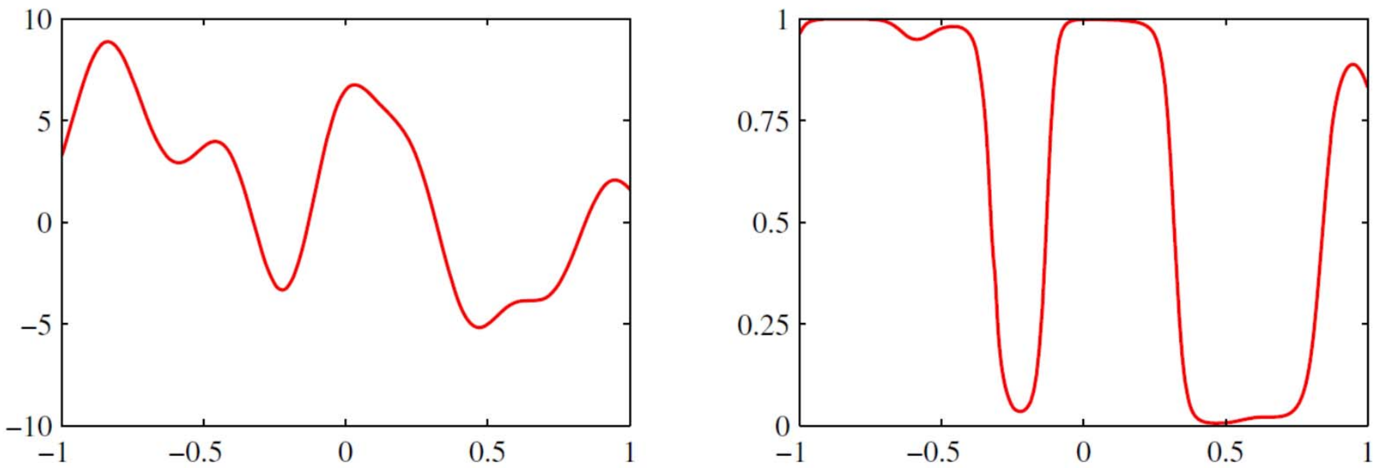


Figure 6.11 The left plot shows a sample from a Gaussian process prior over functions $a(\mathbf{x})$, and the right plot shows the result of transforming this sample using a logistic sigmoid function.

GP Classification

- Summary of main equations

$$p(\mathbf{a}_{N+1}) = \mathcal{N}(\mathbf{a}_{N+1}|\mathbf{0}, C_{N+1})$$

$$C_{nm} = k(\mathbf{x}_n, \mathbf{x}_m) + \nu\delta_{nm}$$

$$p(\mathbf{t}_{N+1}|\mathbf{a}_{N+1}) = \prod_n \sigma(a_n)^{t_n}(1-\sigma(a_n))^{1-t_n}$$

- Predictive distribution

$$p(t_{N+1}=1|\mathbf{t}_N) = \int p(t_{N+1}=1|a_{N+1})p(a_{N+1}|\mathbf{t}_N)da_{N+1}$$

$$= \int \sigma(a_{N+1})p(a_{N+1}|\mathbf{t}_N)da_{N+1}$$

- No analytical solution!
  - Sampling
  - Analytical approximation

GP Classification: Laplace Approx.

- Want: $p(t_{N+1}=1|\mathbf{t}_N) = \int \sigma(a_{N+1}) p(a_{N+1}|\mathbf{t}_N) da_{N+1}$
  - Approximate $p(a_{N+1}|\mathbf{t}_N)$ as a Gaussian, "Convolving" with the sigmoid will be dealt later

$$\begin{aligned} p(a_{N+1}|\mathbf{t}_N) &= \int p(a_{N+1}, \mathbf{a}_N|\mathbf{t}_N) d\mathbf{a}_N \\ &= \frac{1}{p(\mathbf{t}_N)} \int p(a_{N+1}, \mathbf{a}_N) p(\mathbf{t}_N|a_{N+1}, \mathbf{a}_N) d\mathbf{a}_N \\ &= \frac{1}{p(\mathbf{t}_N)} \int p(a_{N+1}|\mathbf{a}_N) p(\mathbf{a}_N) p(\mathbf{t}_N|\mathbf{a}_N) d\mathbf{a}_N \\ &= \int p(a_{N+1}|\mathbf{a}_N) p(\mathbf{a}_N|\mathbf{t}_N) d\mathbf{a}_N \end{aligned}$$

  - From GP prior: $p(a_{N+1}|\mathbf{a}_N) = \mathcal{N}(a_{N+1}|\mathbf{k}^\top C_N^{-1}\mathbf{a}_N, c - \mathbf{k}^\top C_N^{-1}\mathbf{k})$
  - Laplace approximation of $p(\mathbf{a}_N|\mathbf{t}_N)$: Gaussian around the mode

GP Classification: Laplace Approx.

- Laplace approximation of $p(\mathbf{a}_N|\mathbf{t}_N)$
  - Reminder:

$$p(\mathbf{a}_N) = \mathcal{N}(\mathbf{a}_N|\mathbf{0}, C_N)$$

$$p(\mathbf{t}_N|\mathbf{a}_N) = \prod_n \sigma(a_n)^{t_n}(1-\sigma(a_n))^{1-t_n} = \prod_n e^{a_n t_n}\sigma(-a_n)$$

  - Thus,

$$\log p(\mathbf{a}_N|\mathbf{t}_N) = \log p(\mathbf{a}_N) + \log p(\mathbf{t}_N|\mathbf{a}_N) + C$$

$$= -\frac{1}{2}\mathbf{a}_N^\top C_N^{-1}\mathbf{a}_N - \frac{1}{2}\log|C_N| + \mathbf{t}_N^\top \mathbf{a}_N - \sum_n \log(1+e^{a_n}) + C$$

$$\nabla \log p(\mathbf{a}_N|\mathbf{t}_N) = \mathbf{t}_N - \boldsymbol{\sigma}_N - C_N^{-1}\mathbf{a}_N$$

  - Gradient ascent or Newton's method:

$$\nabla^2 \log p(\mathbf{a}_N|\mathbf{t}_N) = -\mathbf{W}_N - C_N^{-1}$$

$$\mathbf{W}_N = \mathrm{diag}(\sigma(a_1)(1-\sigma(a_1)), \ldots \sigma(a_N)(1-\sigma(a_N)))$$

GP Classification: Laplace Approx.

□ From gradient and hessian of log posterior

$$\nabla \log p(\mathbf{a}_N|\mathbf{t}_N) = \mathbf{t}_N - \boldsymbol{\sigma}_N - C_N^{-1}\mathbf{a}_N$$

$$\nabla^2 \log p(\mathbf{a}_N|\mathbf{t}_N) = -\mathbf{W}_N - C_N^{-1}$$

$$\mathbf{W}_N = \mathrm{diag}(\sigma(a_1)(1-\sigma(a_1)), \ldots \sigma(a_N)(1-\sigma(a_N))$$

- The extreme point $\mathbf{a}_N^*$ can be found efficiently, and use it to obtain the Gaussian

$$p(\mathbf{a}_N|\mathbf{t}_N) \approx q(\mathbf{a}_N) = \mathcal{N}(\mathbf{a}_N|\mathbf{a}_N^*, (\mathbf{W}_N^* + C_N^{-1})^{-1})$$

□ To summarize: $p(a_{N+1}|\mathbf{t}_N) = \int p(a_{N+1}|\mathbf{a}_N)p(\mathbf{a}_N|\mathbf{t}_N)d\mathbf{a}_N$

$$\approx \int p(a_{N+1}|\mathbf{a}_N)q(\mathbf{a}_N)d\mathbf{a}_N$$

$$E[a_{N+1}|\mathbf{t}_N] = \mathbf{k}^\top(\mathbf{t}_N - \boldsymbol{\sigma}_N)$$

$$Var[a_{N+1}|\mathbf{t}_N] = c - \mathbf{k}^\top(\mathbf{W}_N^{-1} + C_N)^{-1}\mathbf{k}$$

KAIST
Korea Advanced Institute of Science and Technology
한국과학기술원

GP Classification: Laplace Approx.

- The last step (convolution of Gaussian with sigmoid)

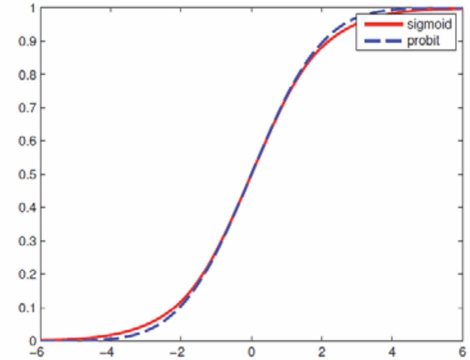
$$p(t_{N+1}=1|\mathbf{t}_N) = \int \sigma(a_{N+1}) p(a_{N+1}|\mathbf{t}_N) da_{N+1}$$
$$\approx \int \sigma(a_{N+1}) \mathcal{N}(a_{N+1}|\hat{\mu}, \hat{\sigma}^2) da_{N+1}$$

- Use $\displaystyle \int \Phi(\lambda a)\mathcal{N}(a|\mu,\sigma^2)da = \Phi\left(\frac{\mu}{(\lambda^{-2}+\sigma^2)^{1/2}}\right)$

  with approximation $\sigma(a) \approx \Phi(\lambda a), \lambda^2 = \pi/8$

  We obtain $\displaystyle \int \sigma(a)\mathcal{N}(a|\mu,\sigma^2)da \approx \sigma(\kappa(\sigma^2)\mu)$

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



- No, I won't show you the final formula

Hyperparameter Learning

- Likelihood:

$$p(\mathbf{t}_N|\boldsymbol{\theta}) = \int p(\mathbf{t}_N, \mathbf{a}_N|\boldsymbol{\theta})d\mathbf{a}_N$$

- Laplace approximation $p(\mathbf{x}) = \frac{1}{Z}f(\mathbf{x})$
  - $\log f(\mathbf{x}) \approx \log f(\mathbf{x}_0) - \frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^\top H(\mathbf{x}-\mathbf{x}_0)$

    $f(\mathbf{x}) \approx f(\mathbf{x}_0)\exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^\top H(\mathbf{x}-\mathbf{x}_0)\right]$

  - $Z = \int f(\mathbf{x})d\mathbf{x} \approx f(\mathbf{x}_0)\int \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^\top H(\mathbf{x}-\mathbf{x}_0)\right]d\mathbf{x}$

    $= f(\mathbf{x}_0)\dfrac{(2\pi)^{D/2}}{|H|^{1/2}}$

  - Observe that $Z = p(\mathbf{t}_N|\boldsymbol{\theta})$, $f(\mathbf{a}_N) = p(\mathbf{t}_N, \mathbf{a}_N|\boldsymbol{\theta})$, and $\mathbf{x}_0 = \mathbf{a}_N^*$

Hyperparameter Learning

- Laplace approximation of Log likelihood:

$$\log p(\mathbf{t}_N|\boldsymbol{\theta}) \approx \log p(\mathbf{t}_N, \mathbf{a}_N^*|\boldsymbol{\theta}) - \frac{1}{2}\log|\mathbf{W}_N + C_N^{-1}| + \frac{N}{2}\log(2\pi)$$

$$= \log p(\mathbf{t}_N|\mathbf{a}_N^*) + \log p(\mathbf{a}_N^*|\theta) - \frac{1}{2}\log|\mathbf{W}_N + C_N^{-1}| + C$$

$$= \mathbf{t}_N^\top \mathbf{a}_N^* - \textstyle\sum_n \log(1 + e^{a_n^*}) - \frac{1}{2}(\mathbf{a}_N^*)^\top C_N^{-1}\mathbf{a}_N^* - \frac{1}{2}\log|I + C_N\mathbf{W}_N| + C$$

- Calculation of gradients:
  - Gradient terms that directly involve $\boldsymbol{\theta}$, i.e. C_N^{-1}:

$$\frac{1}{2}(\mathbf{a}_N^*)^\top C_N^{-1}\frac{\partial C_N}{\partial \theta_j}C_N^{-1}\mathbf{a}_N^* - \frac{1}{2}tr\left[(I + C_N\mathbf{W}_N)^{-1}\mathbf{W}_N\frac{\partial C_N}{\partial \theta_j}\right]$$

  - Gradient terms that are indirect through $\mathbf{a}_N^*$, i.e. $\mathbf{W}_N$:

$$-\frac{1}{2}\sum_n \frac{\partial \log|\mathbf{W}_N + C_N^{-1}|}{\partial a_n^*}\frac{\partial a_n^*}{\partial \theta_j}$$

$$= -\frac{1}{2}\sum_n [(I + C_N\mathbf{W}_N)^{-1}C_N]_{nn}\sigma_n^*(1 - \sigma_n^*)(1 - 2\sigma_n^*)\frac{\partial a_n^*}{\partial \theta_j}$$

$$\frac{\partial a_N^*}{\partial \theta_j} = (I + \mathbf{W}_N C_N)^{-1}\frac{\partial C_N}{\partial \theta_j}(\mathbf{t}_N - \boldsymbol{\sigma}_N)$$

KAIST
Korea Advanced Institute of Science and Technology
한국과학기술원

GP Classification: Example

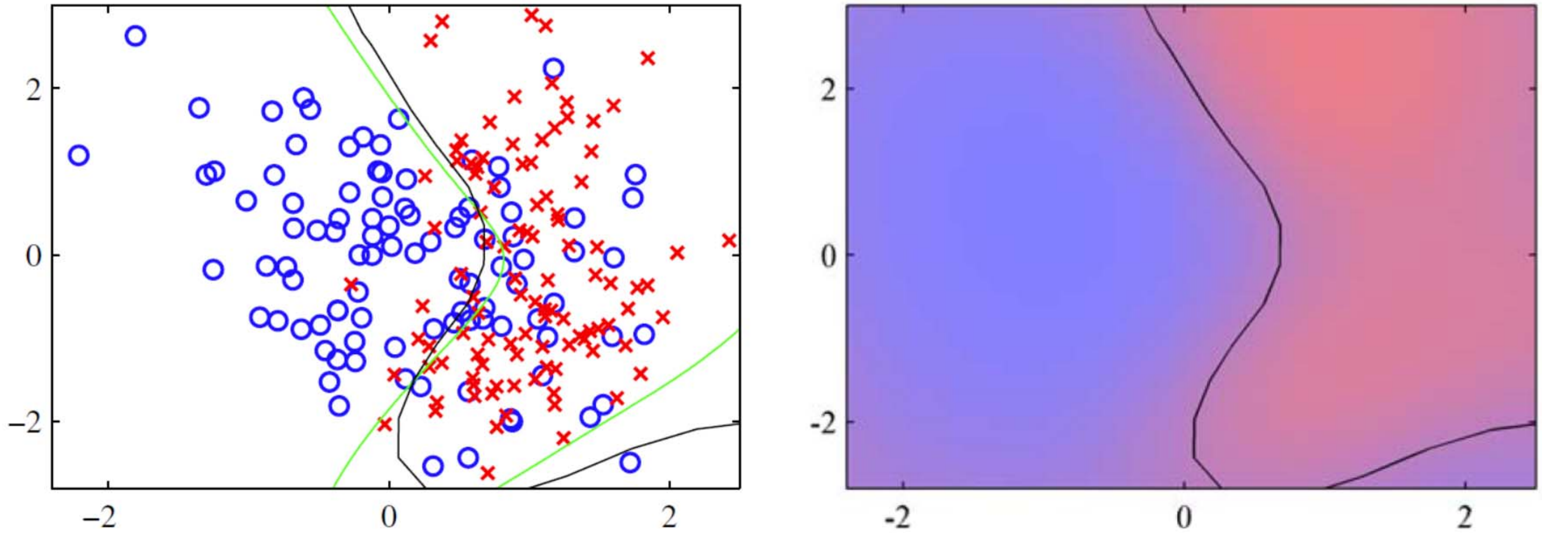


Figure 6.12 Illustration of the use of a Gaussian process for classification, showing the data on the left together with the optimal decision boundary from the true distribution in green, and the decision boundary from the Gaussian process classifier in black. On the right is the predicted posterior probability for the blue and red classes together with the Gaussian process decision boundary.