Machine Learning 2

2. Gaussian Process Models (GPs)

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

			A. Advanced Supervised Learning
Fri.	24.4.	(1)	A.1 Generalized Linear Models
Fri.	1.5.	—	— *Labour Day* —
Fri.	8.5.	(2)	A.2 Gaussian Processes
Fri.	15.5.	(3)	A.3 Advanced Support Vector Machines
			B. Ensembles
Fri.	22.5.	(4)	B.1 Stacking
			& B.2 Boosting
Fri.	29.5.	(5)	B.3 Mixtures of Experts
Fri.	5.6.	—	— *Pentecoste Break* —
			C. Sparse Models
Fri.	12.6.	(6)	C.1 Homotopy and Least Angle Regression
Fri.	19.6.	(7)	C.2 Proximal Gradients
Fri.	26.6.	(8)	C.3 Laplace Priors
Fri.	3.7.	(9)	C.4 Automatic Relevance Determination
			D. Complex Predictors
Fri.	10.7.	(10)	D.1 Latent Dirichlet Allocation (LDA)
Fri.	17.7.	(11)	Q & A

Outline

Outline

Gaussian Process Model

Gaussian Processes describe

- the **vector** $y := (y_1, \ldots, y_N)^T$ **of all targets**
- as a sample from a **normal distribution**
- where targets of different instances are **correlated by a kernel** Σ:
- and thus depend on the **matrix** X **of all predictors**:

$$y \mid X \sim \mathcal{N}(y \mid \mu(X), \Sigma(X))$$

with

$$\begin{aligned} \mu(X)_n &:= m(x_n) \\ \Sigma(X)_{n,m} &:= k(x_n, x_m), \quad n, m \in \{1, \ldots, N\} \end{aligned}$$

with a **kernel function** k and **mean function** m (often $m = 0$).

Kernels

The kernel k measures how much targets y, y' correlate given their predictors x, x'.

- $k(x, x')$ is larger the more similar x, x' are
- esp. $k(x, x) \geq k(x, x') \ \forall x, x'$

Example: **squared exponential kernel** / **Gaussian kernel**

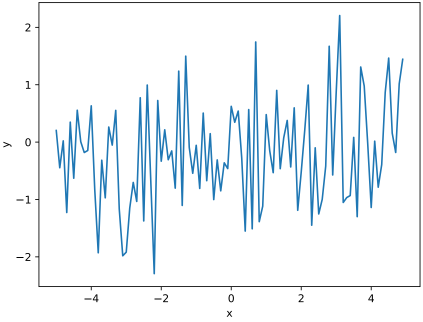
$$k(x, x') := \sigma_f^2 \, e^{-\frac{1}{2\ell^2}||x-x'||^2}$$

with **kernel (hyper)parameters**

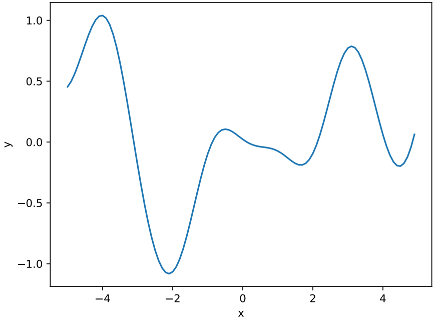
ℓ	horizontal length scale (x)
σ_f^2	vertical variation (y)

GPs as Prior on Functions

identity kernel

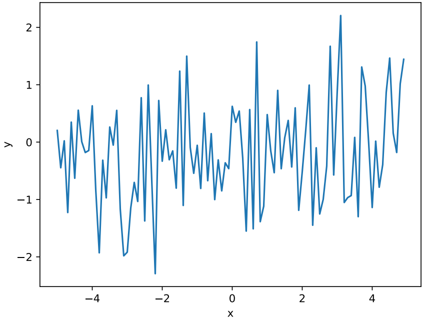


squared exponential kernel

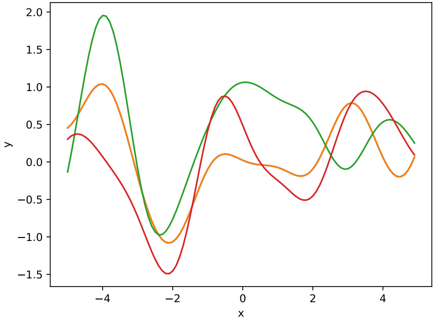


GPs as Prior on Functions

identity kernel



squared exponential kernel



Outline

Conditional Distributions of Multivariate Normals

Let y_A, y_B be jointly Gaussian

$$y := \begin{pmatrix} y_A \\ y_B \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} y_A \\ y_B \end{pmatrix} \middle| \begin{pmatrix} \mu_A \\ \mu_B \end{pmatrix}, \begin{pmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{pmatrix} \right)$$

then the **conditional distribution** is

$$p(y_B \mid y_A) = \mathcal{N}(y_B \mid \mu_{B|A}, \Sigma_{B|A})$$

with

$$\mu_{B|A} := \mu_B + \Sigma_{BA}\Sigma_{AA}^{-1}(y_A - \mu_A)$$
$$\Sigma_{B|A} := \Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB}$$

Predictions w/o Noise

Let y, X be the training data,
X_* be the test data and
y_* be the test targets to predict.

$$\begin{pmatrix} y \\ y_* \end{pmatrix} \mid X, X_* \sim \mathcal{N}(\begin{pmatrix} y \\ y_* \end{pmatrix} \mid \begin{pmatrix} \mu \\ \mu_* \end{pmatrix}, \begin{pmatrix} \Sigma & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{pmatrix})$$

with

$$\mu := m(X), \quad \mu_* := m(X_*)$$
$$\Sigma := k(X, X), \quad \Sigma_* := k(X, X_*), \quad \Sigma_{**} := k(X_*, X_*)$$

Then

$$p(y_* \mid y) = \mathcal{N}(y_* \mid \tilde{\mu}_*, \tilde{\Sigma}_*)$$

$$\tilde{\mu}_* := \mu_* + \Sigma_*^T \Sigma^{-1}(y - \mu)$$
$$\tilde{\Sigma}_* := \Sigma_{**} - \Sigma_*^T \Sigma^{-1} \Sigma_*$$

Example w/o Noise

fig/03-gaussian-process-models/f{Murphy2012-fig15.[illegible]

Without noise the data is interpolated.

[?, fig. 15.2]

Predictions with Noise

No noise:

$$\Sigma := K$$

With noise:

$$\Sigma := K + \sigma_y^2 I$$

Then as before

$$p(y_* \mid y) = \mathcal{N}(y_* \mid \tilde{\mu}_*, \tilde{\Sigma}_*)$$

now with

$$\begin{aligned}\tilde{\mu}_* &:= \mu_* + K_*^T (K + \sigma_y^2 I)^{-1} (y - \mu) \\ \tilde{\Sigma}_* &:= K_{**} + \sigma_y^2 I - K_*^T (K + \sigma_y^2 I)^{-1} K_*\end{aligned}$$

where

$$K := k(X, X), \quad K_* := k(X, X_*), \quad K_{**} := k(X_*, X_*)$$

Predictions with Noise, Zero Means

$$p(y_* \mid y) = \mathcal{N}(y_* \mid \tilde{\mu}_*, \tilde{\Sigma}_*)$$

with

$$\begin{aligned}\tilde{\mu}_* :=& \mu_* + K_*^T (K + \sigma_y^2 I)^{-1} (y - \mu) \\ \tilde{\Sigma}_* :=& K_{**} + \sigma_y^2 I - K_*^T (K + \sigma_y^2 I)^{-1} K_*\end{aligned}$$

With $m = 0$:

$$p(y_* \mid y) = \mathcal{N}(y_* \mid \tilde{\mu}_*, \tilde{\Sigma}_*)$$

with

$$\begin{aligned}\tilde{\mu}_* :=& K_*^T (K + \sigma_y^2 I)^{-1} y \\ \tilde{\Sigma}_* :=& K_{**} + \sigma_y^2 I - K_*^T (K + \sigma_y^2 I)^{-1} K_*\end{aligned}$$

Prediction for a single instance

$$p(y_* \mid y) = \mathcal{N}(y_* \mid \tilde{\mu}_*, \tilde{\Sigma}_*)$$

with

$$\begin{aligned}\tilde{\mu}_* :=& K_*^T(K + \sigma_y^2 I)^{-1} y \\ \tilde{\Sigma}_* :=& K_{**} + \sigma_y^2 I - K_*^T(K + \sigma_y^2 I)^{-1} K_*\end{aligned}$$

Prediction $\hat{y}$ for a single instance x:

$$\hat{y}(x) := k_*^T(K + \sigma_y^2 I)^{-1} y = \sum_{n=1}^{N} \alpha_n k(x_n, x), \quad \alpha := (K + \sigma_y^2 I)^{-1} y$$

with

$$k_* := k(X, x)$$

But GPs can provide a joint inference for multiple instances.

Example with Noise

fig/03-gaussian-pro

fig/03-gaussian-process-models/{Murphy2012-fig15.3

$(\ell, \sigma_f, \sigma_y) = (1, 1, 0.1)$

$(\ell, \sigma_f, \sigma_y) = (0.3, 0.1?, 0.00005)$

[?, fig. 15.3]

Example with Noise

fig/03-gaussian-process-models/{Murphy2012-fig15.3

fig/03-gaussian-pro

$(\ell, \sigma_f, \sigma_y) = (1, 1, 0.1)$

$(\ell, \sigma_f, \sigma_y) = (3, 1.16, 0.89)$

[**?**, fig. 15.3]

Outline

Estimating Kernel Parameters

Either treating them as hyperparameters (grid search, random search) or maximize the marginal likelihood (empirical Bayes; grad. desc.).

Model:

$$p(y \mid X, \theta) = \mathcal{N}(y \mid 0, (K + \sigma_y^2 I)), \quad \theta := (\ell, \sigma_f^2, \sigma_y^2)$$

Negative log-likelihood:

$$\begin{aligned} L(\theta) &= -\log p(y \mid X, \theta) \\ &= \frac{1}{2} y^T (K + \sigma_y^2 I)^{-1} y + \frac{1}{2} \log \det(K + \sigma_y^2 I) + \frac{N}{2} \log(2\pi) \end{aligned}$$

Estimating Kernel Parameters

Negative log-likelihood ($\theta := (\ell, \sigma_f^2, \sigma_y^2)$):

$$L(\theta) = \frac{1}{2} y^T (K + \sigma_y^2 I)^{-1} y + \frac{1}{2} \log\det(K + \sigma_y^2 I) + \frac{N}{2} \log(2\pi)$$

Gradients:

$$\begin{aligned}\frac{\partial L}{\partial \theta_j} =& - \frac{1}{2} y^T (K + \sigma_y^2 I)^{-1} \frac{\partial (K + \sigma_y^2 I)}{\partial \theta_j} (K + \sigma_y^2 I)^{-1} y \\ & + \frac{1}{2} \text{tr}((K + \sigma_y^2 I)^{-1} \frac{\partial (K + \sigma_y^2 I)}{\partial \theta_j}) \\ =& - \frac{1}{2} \text{tr} \left((\alpha\alpha^T - (K + \sigma_y^2 I)^{-1}) \frac{\partial (K + \sigma_y^2 I)}{\partial \theta_j} \right), \quad \alpha := (K + \sigma_y^2 I)^{-1} y\end{aligned}$$

Note: $\partial(X^{-1}) = X^{-1}(\partial X)X^{-1}$, $\partial \det X = \frac{1}{\det X} \text{tr}\left((X^{-1})^T \partial X\right)$,
and $\text{tr}(aa^T B) = a^T B a$. $\theta_1 := \ell, \theta_2 := \sigma_f^2, \theta_3 := \sigma_y^2$.

Cholesky decompositon

How to solve $Ax = b$?

Matrix inversion: $x = A^{-1}b$ is problematic because

- Numerically unstable
- A^{-1} is dense, even if A is sparse

Better: LU-decomposition

$$Ax = b \xrightarrow{A=LU} \begin{array}{l} Lz = b \\ Ux = z \end{array}$$

- L and U lower/upper triangular
- if A symmetric pos.-definite, then (L, U) can be chosen s.t. $U = L^T$ (Cholesky-decomposition)

Local Minima for Kernel Parameters

fig/03-gaussian-process-models/{Murphy2012-fig15.5a}.pdf

fig/03-gaussian-process-models/{Murphy2012-fig15.5b}.pdf

fig/03-gaussian-process-models/{Murphy2012-fig15.5c}.pdf

- top: $(\ell, \sigma_y) \approx (10, 0.8)$
- left: $(\ell, \sigma_y) \approx (1, 0.1)$

[?, fig. 15.5]

Semi-parametric GPs

$$\begin{aligned} f(x) =& \beta^T \phi(x) + r(x) \\ r(X) \sim& \mathrm{GP}(r \mid 0, k(X, X)) \end{aligned}$$

Assuming

$$\beta \sim \mathcal{N}(\beta \mid b, B), \quad \text{e.g., } b := 0, B := \sigma_\beta^2 I$$

yields just another GP

$$f(X) \sim \mathrm{GP}(\phi(X)^T b, k(X, X) + \phi(X) B \phi(X)^T)$$

where

$$\phi(X) := (\phi(x_1), \ldots, \phi(x_N))^T$$

Outline

Model

$$p(y \mid x) := s(y\, f(x)), \quad y \in \{+1, -1\}, s := \text{logistic}$$
$$f \sim \text{GP}(f \mid 0, K(X, X))$$

- f: **latent score**

Inference

Two-step inference:

1. infer latent score variable:

$$p(f_* \mid X, y, x_*) = \int p(f_* \mid X, x_*, f)\, p(f \mid X, y)\, df$$

2. infer target:

$$\pi_* := p(y_* = +1 \mid X, y, x_*) = \int s(f_*)\, p(f_* \mid X, y, x_*)\, df_*$$

Non Gaussians are analytically intractable.

$\rightsquigarrow$ Gaussian approximation (**Laplace approximation**)

$\rightsquigarrow$ **Expectation Propagation (EP)**

$\rightsquigarrow$ further methods

Posterior

$$\begin{aligned}
p(f \mid X, y) &= \frac{p(y \mid f, X)\, p(f \mid X)}{p(y \mid X)} \propto p(y \mid f)\, p(f \mid X) \\
\ell(f) &= \log p(y \mid f) + \log p(f \mid X) \\
&= \log p(y \mid f) - \frac{1}{2} f^T K^{-1} f - \frac{1}{2} \log |K| - \frac{N}{2} \log 2\pi \\
\nabla \ell(f) &= \nabla \log p(y \mid f) - K^{-1} f \\
\nabla^2 \ell(f) &= \nabla^2 \log p(y \mid f) - K^{-1}
\end{aligned}$$

for logistic:

$$\begin{aligned}
\nabla \log p(y \mid f) &= y - \pi \\
\nabla^2 \log p(y \mid f) &= \operatorname{diag}(-\pi \circ (1 - \pi)) =: -W
\end{aligned}$$

at maximum:

$$\nabla \ell(f) = 0 \quad \Longrightarrow \quad f = K \nabla \log p(y \mid f)$$

Posterior

at maximum:

$$\nabla \ell(f) = 0 \quad \Longrightarrow \quad f = K \nabla \log p(y \mid f)$$

Use Newton to find a maximum:

$$\begin{aligned} f^{(t+1)} :=& f^{(t)} - (\nabla^2 \ell)^{-1} \nabla \ell \\ =& f^{(t)} + (K^{-1} + W^{(t)})^{-1} (\nabla \log p(y \mid f) - K^{-1} f^{(t)}) \\ =& (K^{-1} + W^{(t)})^{-1} (W^{(t)} f^{(t)} + \nabla \log p(y \mid f)) \end{aligned}$$

eventually yielding the maximum posterior $\hat{f}$.

Gaussian Approximation

$$p(f \mid X, y) \approx q(f \mid X, y) := \mathcal{N}(f \mid \hat{f}, (K^{-1} + W)^{-1})$$

using the Hessian as covariance matrix.

Predictions

exact mean

$$\begin{aligned} E_p(f_* \mid X, y, x_*) &= \int E(f_* \mid f, X, x_*)\, p(f \mid X, y) df \\ &= \int k(x_*)^T K^{-1} f\, p(f \mid X, y) df \\ &= k(x_*)^T K^{-1} E_p(f \mid X, y) \end{aligned}$$

approximated mean:

$$E_q(f_* \mid X, y, x_*) = k(x_*)^T K^{-1} \hat{f}$$

variance:

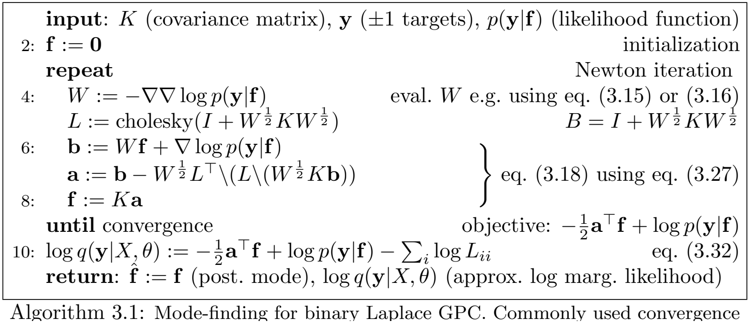
$$\mathrm{Var}_q(f_* \mid X, y, x_*) = k(x_*, x_*) - k_*^T (K + W^{-1})^{-1} k_*$$

predictions:

$$\bar{\pi}_* := E_q(\pi_* \mid X, y, x*) = \int s(f_*) q(f_* \mid X, y, x_*) df_*$$

solve integral via MCMC or
probit approximation (Murphy 8.4.4.2)

Algorithm (Step 1)

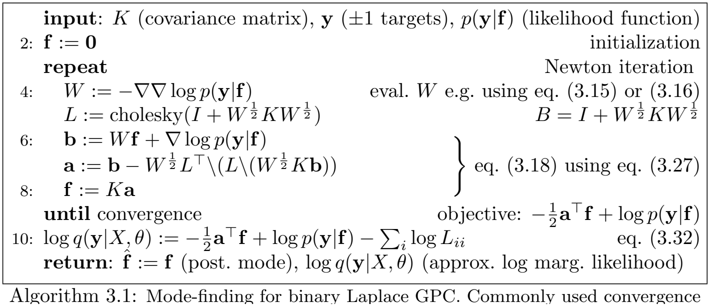


```
input: $K$ (covariance matrix), $\mathbf{y}$ ($\pm 1$ targets), $p(\mathbf{y}|\mathbf{f})$ (likelihood function)
$\mathbf{f} := \mathbf{0}$                                          initialization
repeat                                                   Newton iteration
    $W := -\nabla\nabla \log p(\mathbf{y}|\mathbf{f})$           eval. $W$ e.g. using eq. (3.15) or (3.16)
    $L := \text{cholesky}(I + W^{\frac{1}{2}} K W^{\frac{1}{2}})$                  $B = I + W^{\frac{1}{2}} K W^{\frac{1}{2}}$
    $\mathbf{b} := W\mathbf{f} + \nabla \log p(\mathbf{y}|\mathbf{f})$                  }
    $\mathbf{a} := \mathbf{b} - W^{\frac{1}{2}} L^\top \backslash (L \backslash (W^{\frac{1}{2}} K\mathbf{b}))$    } eq. (3.18) using eq. (3.27)
    $\mathbf{f} := K\mathbf{a}$                                      }
until convergence                 objective: $-\frac{1}{2}\mathbf{a}^\top\mathbf{f} + \log p(\mathbf{y}|\mathbf{f})$
$\log q(\mathbf{y}|X,\theta) := -\frac{1}{2}\mathbf{a}^\top\mathbf{f} + \log p(\mathbf{y}|\mathbf{f}) - \sum_i \log L_{ii}$        eq. (3.32)
return: $\hat{\mathbf{f}} := \mathbf{f}$ (post. mode), $\log q(\mathbf{y}|X,\theta)$ (approx. log marg. likelihood)
```

Algorithm 3.1: Mode-finding for binary Laplace GPC. Commonly used convergence

[Rasmussen/Williams 200

Algorithm (Step 1)



```
    input: $K$ (covariance matrix), $\mathbf{y}$ ($\pm 1$ targets), $p(\mathbf{y}|\mathbf{f})$ (likelihood function)
2:  $\mathbf{f} := \mathbf{0}$                                                    initialization
    repeat                                                                        Newton iteration
4:      $W := -\nabla\nabla \log p(\mathbf{y}|\mathbf{f})$             eval. $W$ e.g. using eq. (3.15) or (3.16)
        $L := \text{cholesky}(I + W^{\frac{1}{2}} K W^{\frac{1}{2}})$                 $B = I + W^{\frac{1}{2}} K W^{\frac{1}{2}}$
6:      $\mathbf{b} := W\mathbf{f} + \nabla \log p(\mathbf{y}|\mathbf{f})$        \
        $\mathbf{a} := \mathbf{b} - W^{\frac{1}{2}} L^\top \backslash (L \backslash (W^{\frac{1}{2}} K\mathbf{b}))$   } eq. (3.18) using eq. (3.27)
8:      $\mathbf{f} := K\mathbf{a}$                                               /
    until convergence                                   objective: $-\frac{1}{2}\mathbf{a}^\top\mathbf{f} + \log p(\mathbf{y}|\mathbf{f})$
10: $\log q(\mathbf{y}|X,\theta) := -\frac{1}{2}\mathbf{a}^\top\mathbf{f} + \log p(\mathbf{y}|\mathbf{f}) - \sum_i \log L_{ii}$          eq. (3.32)
    return: $\hat{\mathbf{f}} := \mathbf{f}$ (post. mode), $\log q(\mathbf{y}|X,\theta)$ (approx. log marg. likelihood)
```

Algorithm 3.1: Mode-finding for binary Laplace GPC. Commonly used convergence

Algorithm (Step 2)

```
   input: $\hat{\mathbf{f}}$ (mode), $X$ (inputs), $\mathbf{y}$ ($\pm 1$ targets), $k$ (covariance function),
                              $p(\mathbf{y}|\mathbf{f})$ (likelihood function), $\mathbf{x}_*$ test input
2: $W := -\nabla\nabla \log p(\mathbf{y}|\hat{\mathbf{f}})$
   $L := \text{cholesky}(I + W^{\frac{1}{2}} K W^{\frac{1}{2}})$                    $B = I + W^{\frac{1}{2}} K W^{\frac{1}{2}}$
4: $\bar{f}_* := \mathbf{k}(\mathbf{x}_*)^\top \nabla \log p(\mathbf{y}|\hat{\mathbf{f}})$                                 eq. (3.21)
   $\mathbf{v} := L \backslash \big(W^{\frac{1}{2}} \mathbf{k}(\mathbf{x}_*)\big)$                 $\Big\}$ eq. (3.24) using eq. (3.29)
6: $\mathbb{V}[f_*] := k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{v}^\top \mathbf{v}$
   $\bar{\pi}_* := \int \sigma(z) \mathcal{N}(z|\bar{f}_*, \mathbb{V}[f_*]) dz$                      eq. (3.25)
8: return: $\bar{\pi}_*$ (predictive class probability (for class 1))
```

Algorithm 3.2: Predictions for binary Laplace GPC. The posterior mode $\hat{\mathbf{f}}$ (which can be computed using Algorithm 3.1) is input. For multiple test inputs lines $4-7$ are applied to each test input. Computational complexity is $n^3/6$ operations once (line 3) plus n^2 operations per test case (line 5). The one-dimensional integral in line 7 can be done analytically for cumulative Gaussian likelihood, otherwise it is computed using an approximation or numerical quadrature.

MCMC

How to compute integrals of the form

$$\int_a^b h(x)p(x)dx$$

where p is a probability density on $[a, b]$.

$$\int_a^b h(x)p(x)dx = \mathbb{E}_p[h] \approx \frac{1}{N}\sum_{i=1}^{N} h(x_i) \qquad (1)$$

when x_i are sampled iid from p. (**Monte-Carlo**-integration)
Markov-Chain-Monte-Carlo: Clever sampling strategy of x_i

Approximation Methods for Large Datasets

See recent literature:

- Filippone, M. and Engler, R. 2015.
  Enabling scalable stochastic gradient-based inference for Gaussian processes by employing the Unbiased LInear System SolvEr (ULISSE), arXiv preprint arXiv:1501.05427. (2015).
- Dai, B., Xie, B., He, N., Liang, Y., Raj, A., Balcan, M.-F. and Song, L. 2014.
  Scalable Kernel Methods via Doubly Stochastic Gradients. arXiv:1407.5599 [cs, stat]. (Jul. 2014).
- Hensman, J., Fusi, N. and Lawrence, N.D. 2013.
  Gaussian processes for big data. arXiv preprint arXiv:1309.6835. (2013).

Summary

- **Gaussian processes** model continuous targets as jointly normally distributed.
  - correlated by covariance matrix depending on the predictors (**kernel**)
- **The squared exponential kernel** often is used as kernel.
  - having 2 kernel parameters: **horizontal length scale** and **vertical variation**
- **Noise variation** has to be added to the model
  — otherwise Gaussian processes interpolate the observed data.
- Kernel parameters can be learnt through gradient descent.
  - the objective is not convex, local minima need to be treated

Summary (2/2)

- For classification, Gaussian processes can be used to model
  - a **score function** f
  - that is mapped through the logistic function to probabilities π of target labels.
- The posterior is not Gaussian, but can be approximated by a Gaussian (**Laplace approximation**).
- Also the posterior predictive $E(\pi_* \mid x_*, X, y)$ cannot be computed analytically.
  - but it can be approximated by an integral over the (approximatly) normally distributed predictive score f_*
  - and thus be computed by MCMC.

Further Readings

- Rasmussen & Williams: Gaussian Processes for Machine Learning (free ebook!)
- See also [**?**, chapter 15].
- Conditioning Gaussians: [**?**, section 4.3].
- Derivatives of inverse of a matrix etc., see, e.g., *The Matrix Cookbook*, `http://www.mit.edu/~wingated/stuff_i_use/matrix_cookbook.pdf`

Some Matrix Derivatives

$$\partial(X^{-1}) = -X^{-1}(\partial X)X^{-1}$$
$$\partial(\log(|X|)) = \mathrm{tr}(X^{-1}\partial X)$$

Computing with traces:

$$\mathrm{tr}(aa^T B) = a^T Ba$$

References

Kevin P. Murphy.
Machine learning: a probabilistic perspective.
The MIT Press, 2012.