Probabilistic Machine Learning
Lecture 13
Gaussian Process Classification

Philipp Hennig

08 June 2020

Eberhard Karls Universität Tübingen

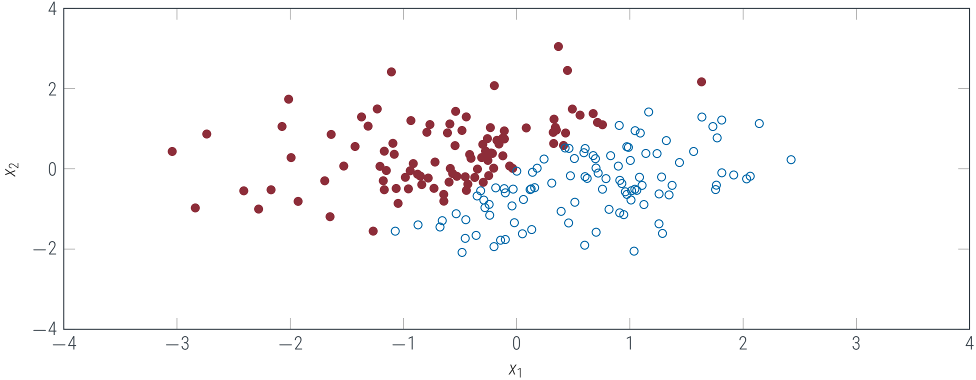
Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

#	date	content	Ex	#	date	content	Ex
1	20.04.	Introduction	1	14	09.06.	Generalized Linear Models	
2	21.04.	Reasoning under Uncertainty		15	15.06.	Exponential Families	8
3	27.04.	Continuous Variables	2	16	16.06.	Graphical Models	
4	28.04.	Monte Carlo		17	22.06.	Factor Graphs	9
5	04.05.	Markov Chain Monte Carlo	3	18	23.06.	The Sum-Product Algorithm	
6	05.05.	Gaussian Distributions		19	29.06.	Example: Topic Models	10
7	11.05.	Parametric Regression	4	20	30.06.	Mixture Models	
8	12.05.	Learning Representations		21	06.07.	EM	11
9	18.05.	Gaussian Processes	5	22	07.07.	Variational Inference	
10	19.05.	Understanding Kernels		23	13.07.	Topics	
12	26.05.	Gauss-Markov Models		25	20.07.	Example: Kernel Topic Models	
11	25.05.	An Example for GP Regression	6	24	14.07.	Example: Inferringevision	
13	08.06.	GP Classification	7	26	21.07.	Revision	

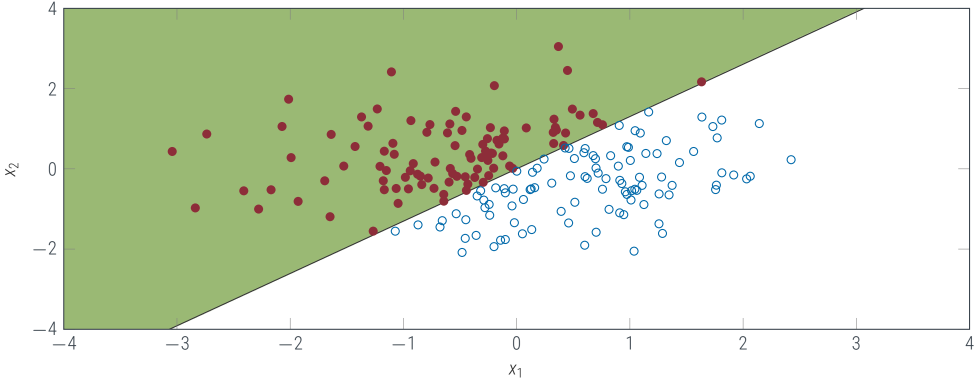
Classification Problems

a typography of supervised learning problems



Classification Problems

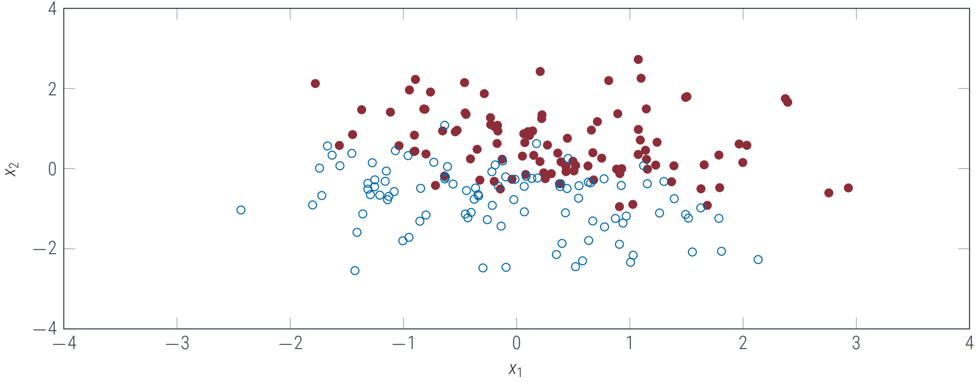
a typography of supervised learning problems



Classification Problems

a typography of supervised learning problems

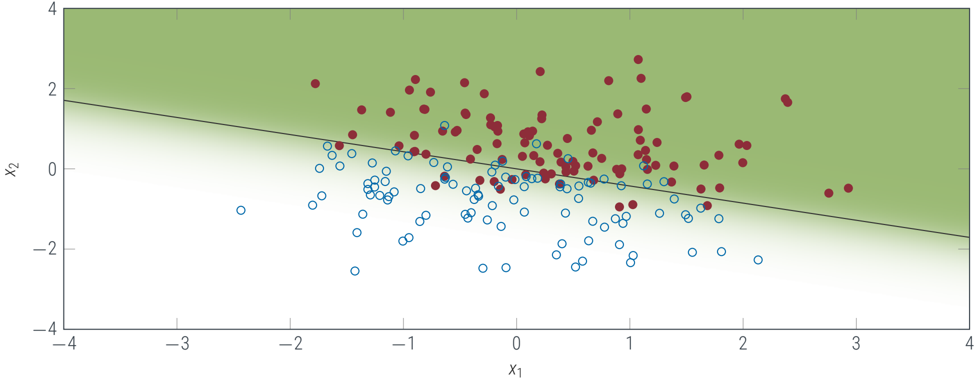
EBERHARD KARLS UNIVERSITÄT TÜBINGEN



Classification Problems

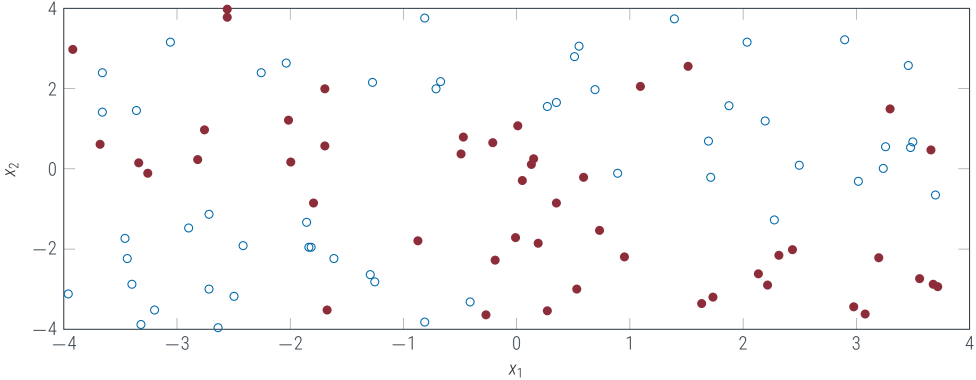
a typography of supervised learning problems

EBERHARD KARLS UNIVERSITÄT TÜBINGEN



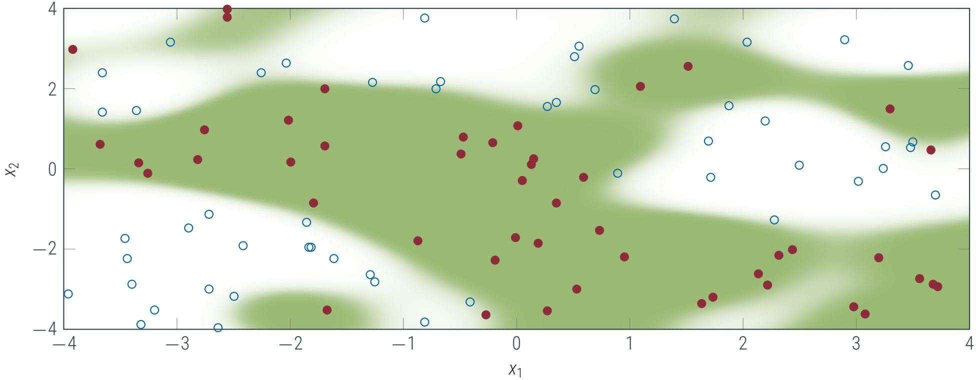
Classification Problems

a typography of supervised learning problems



Classification Problems

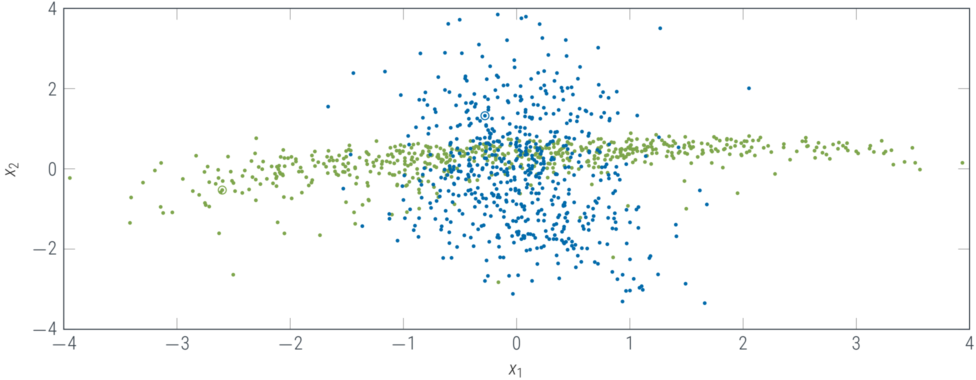
a typography of supervised learning problems



Classification Problems

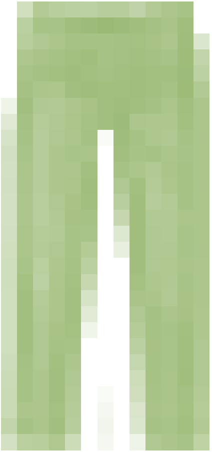
a typography of supervised learning problems

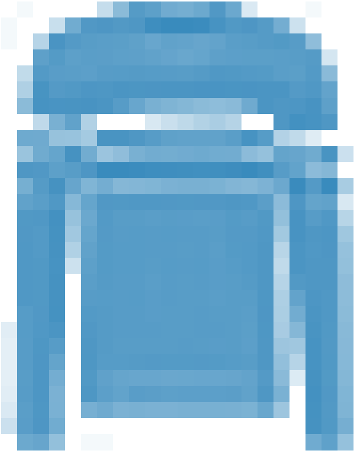
EBERHARD KARLS UNIVERSITÄT TÜBINGEN



Classification Problems

a typography of supervised learning problems





https://github.com/zalandoresearch/fashion-mnist

Classification vs. Regression

Two types of supervised learning problems

Regression:

Given supervised *data* (special case $d = 1$: univariate regression)

$$(X, Y) := (x_i, y_i)_{i=1,\dots,n} \text{ with } x_i \in \mathbb{X}, y_i \in \mathbb{R}^d$$

find function $f : \mathbb{X} \to \mathbb{R}^d$ such that f "models" $Y \approx f(X)$.

Classification:

Given supervised *data* (special case $d = 2$: binary classification)

$$(X, Y) := (x_i, c_i)_{i=1,\dots,n} \text{ with } x_i \in \mathbb{X}, c_i \in \{1, \dots, d\}$$

find probability $\pi : \mathbb{X} \to U^d$ ($U^d = \{p \in [0, 1]^d : \sum_{i=1}^d p_i = 1\}$) such that π "models" $y_i \sim \pi_{x_i}$.

Regression predicts a **function**, *classification* predicts a **probability**.

Until further notice, consider only discriminative binary classification:

$$y \in \{-1; +1\} \qquad x \rightharpoonup \pi(x) =: \pi_x \in [0, 1]$$

$$p(y \mid x) = \begin{cases} \pi(x) & \text{if } y = 1 \\ 1 - \pi(x) & \text{if } y = -1 \end{cases}$$

Until further notice, consider only discriminative binary classification:

$$y \in \{-1; +1\} \qquad x \rightharpoondown \pi(x) =: \pi_x \in [0, 1]$$

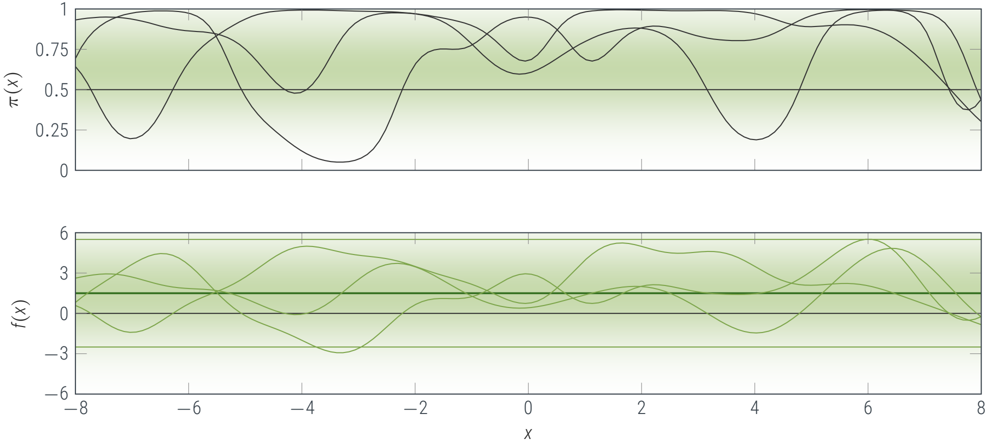
$$p(y \mid x) = \begin{cases} \pi(x) & \text{if } y = 1 \\ 1 - \pi(x) & \text{if } y = -1 \end{cases}$$

Discriminative learning phrased probabilistically:

- We would like to *learn* $\pi_x(y) = p(y \mid x)$
- This is *almost* like regression: $p(y \mid x) = \mathcal{N}(y; f_x, \sigma^2) = \pi_x(y)$
- only the *domain* is wrong: $y \in \{-1; 1\}$ vs. $y \in \mathbb{R}$.

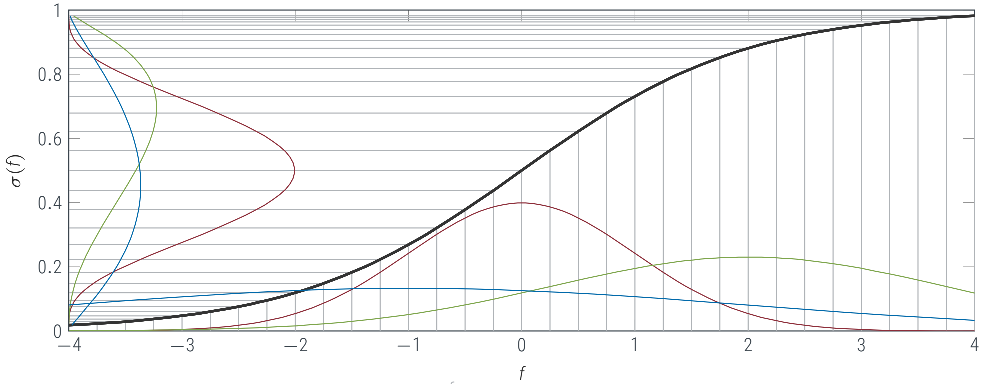
Let's not throw out Gauss just yet

Turning Gaussian process models into discrete likelihoods



Link Functions

sigmoids — the **logistic** link function



$$\pi_f = \sigma(f) = \frac{1}{1+\exp(-f)} = \int_{-\infty}^{f} \frac{1}{4}\operatorname{sech}^2\left(\frac{x}{2}\right)\,dx$$

$$\sigma(f) = 1 - \sigma(-f) \qquad f(\pi) = \ln \pi - \ln(1-\pi) \qquad \frac{d\pi}{df} = \pi(f)\cdot(1-\pi(f))$$

CODE

Generative Model for Logistic Regression.ipynb

$$p(f) = \mathcal{GP}(f; m, k)$$

$$p(y \mid f_x) = \sigma(y f_x) = \begin{cases} \sigma(f) & \text{if } y = 1 \\ 1 - \sigma(f) & \text{if } y = -1 \end{cases} \qquad \text{using } \sigma(x) = 1 - \sigma(-x).$$

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

$$p(f) = \mathcal{GP}(f; m, k)$$

$$p(y \mid f_x) = \sigma(y f_x) = \begin{cases} \sigma(f) & \text{if } y = 1 \\ 1 - \sigma(f) & \text{if } y = -1 \end{cases} \qquad \text{using } \sigma(x) = 1 - \sigma(-x).$$

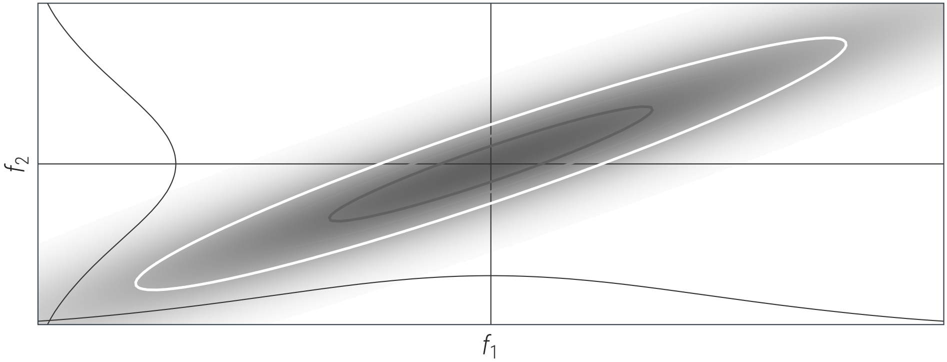
The problem: The posterior is not Gaussian!

$$p(f_X \mid Y) = \frac{p(Y \mid f_X) p(f_X)}{p(Y)} = \frac{\mathcal{N}(f_X; m, k) \prod_{i=1}^n \sigma(y_i f_{x_i})}{\int \mathcal{N}(f_X; m, k) \prod_{i=1}^n \sigma(y_i f_{x_i}) \, df_X}$$

$$\log p(f_X \mid Y) = -\frac{1}{2} f_X^\intercal k_{XX}^{-1} f_X + \sum_{i=1}^n \log \sigma(y_i f_{x_i}) + \text{const.}$$

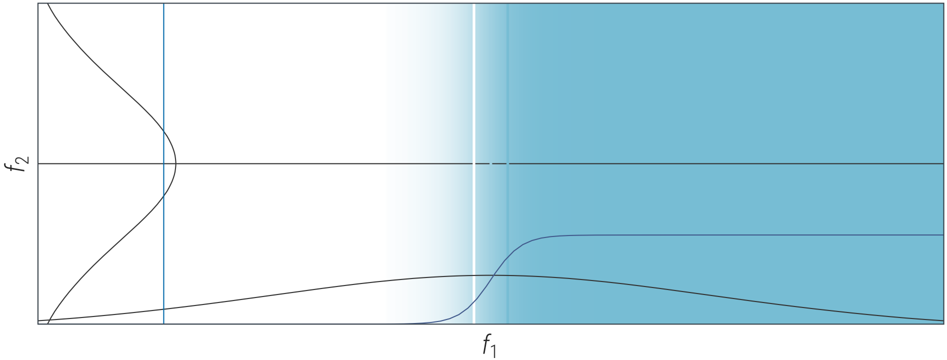
Logistic Regression is non-analytic

We'll have to break out the toolbox



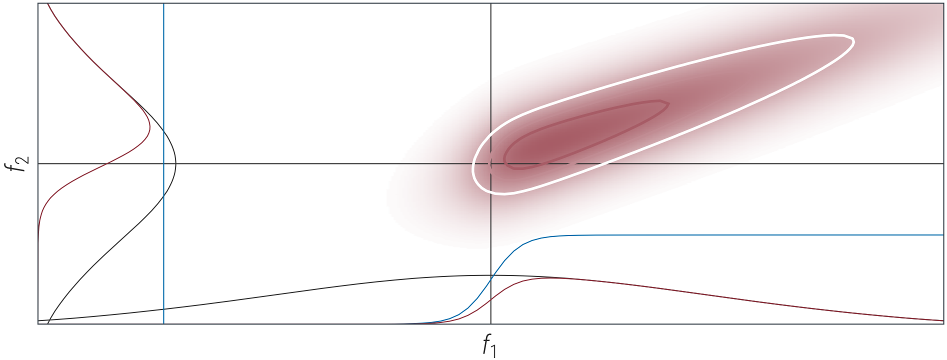
Logistic Regression is non-analytic

We'll have to break out the toolbox



Logistic Regression is non-analytic

We'll have to break out the toolbox



We do not always care about all details of the posterior, just “key aspects”.

- Remember that the Gaussian choice was also one of convenience.
- **Moments** of $p(f, y) = p(y \mid f)p(f)$ we may be interested in

$$\mathbb{E}_p(1) = \int p(y, f)\, df = \int 1 \cdot p(y, f)\, df = Z \quad \text{the } \textbf{evidence}$$

$$\mathbb{E}_{p(f|y)}(f) = \int f \cdot p(f \mid y)\, df = \frac{1}{Z} \int f \cdot p(f, y)\, df = \bar{f} \quad \text{the } \textbf{mean}$$

$$\mathbb{E}_{p(f|y)}(f^2) - \bar{f}^2 = \int f^2 \cdot p(f \mid y)\, df - \bar{f}^2 = \frac{1}{Z} \int f^2 \cdot p(f, y)\, df - \bar{f}^2 = \operatorname{var}(f) \quad \text{the } \textbf{variance}$$

Z for hyperparameter tuning — $\bar{f}$ as a point estimator — $\operatorname{var}(f)$ as an error estimator

Unfortunately, all these are usually intractable. But we can aim to approximate them.

The Toolbox

Framework:

$$\int p(x_1, x_2)\, dx_2 = p(x_1) \qquad p(x_1, x_2) = p(x_1 \mid x_2)p(x_2) \qquad p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)}$$

Modelling:

- Directed Graphical Models
- Gaussian Distributions
- Kernels
- Markov Chains
-
-

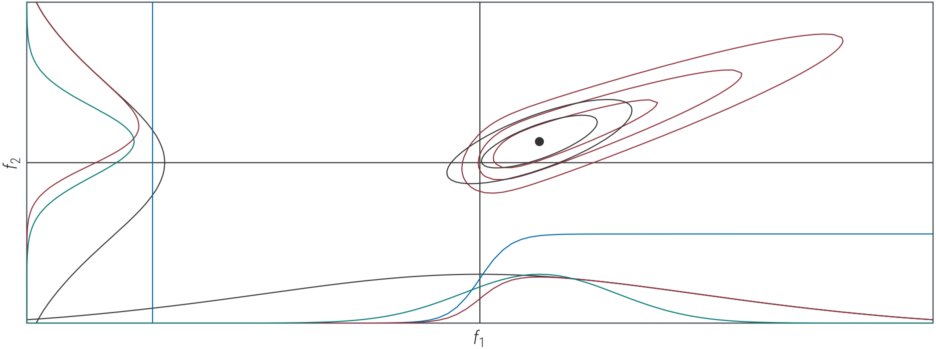
Computation:

- Monte Carlo
- Linear algebra / Gaussian inference
- maximum likelihood / MAP
- Laplace approximations
-

The Laplace Approximation

A local Gaussian approximation

Pierre Simon M. de Laplace, 1814



An idea as old as Probabilistic Inference

Théorie analytique des probabilités, 1814

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Pierre-Simon, marquis de Laplace (1749-1827)

364 THÉORIE ANALYTIQUE

l'intégrale du numérateur étant prise depuis $x = \theta$ jusqu'à $x = \theta'$, et celle du dénominateur étant prise depuis $x = 0$ jusqu'à $x = 1$;

La valeur de x la plus probable, est celle qui rend y un *maximum*. Nous la désignerons par a. Si aux limites de x, y est nul, alors chaque valeur de y a une valeur égale correspondante de l'autre côté du *maximum*.

Quand les valeurs de x, considérées indépendamment du résultat observé, ne sont pas également possibles; en nommant z la fonction de x qui exprime leur probabilité; il est facile de voir, par ce qui a été dit dans le premier chapitre de ce Livre, qu'en changeant dans la formule (1), y dans yz, on aura la probabilité que la valeur de x est comprise dans les limites $x = \theta$ et $x = \theta'$. Cela revient à supposer toutes les valeurs de x également possibles *à priori*, et à considérer le résultat observé, comme étant formé de deux résultats indépendans, dont les probabilités sont y et z. On peut donc ramener ainsi tous les cas à celui où l'on suppose *à priori*, avant l'événement, une égale possibilité aux différentes valeurs de x, et par cette raison, nous adopterons cette hypothèse dans ce qui va suivre.

Nous avons donné dans les n[os] 22 et suivans du premier Livre, les formules nécessaires pour déterminer par des approximations convergentes, les intégrales du numérateur et du dénominateur de la formule (1), lorsque les événemens simples dont se compose l'événement observé, sont répétés un très-grand nombre de fois; car alors y a pour facteurs, des fonctions de x élevées à de grandes puissances. Nous allons, au moyen de ces formules, déterminer la loi de probabilité des valeurs de x, à mesure qu'elles s'éloignent de la valeur a, la plus probable, ou qui rend y un *maximum*. Pour cela, reprenons la formule (c) du n° 27 du premier Livre,

$$\int y\,dx = Y.\left\{U + \frac{1}{2}.\frac{d^2U^3}{1.2.dx^2} + \frac{1.3}{2^2}.\frac{d^4.U^5}{1.2.3.4.dx^4} + \text{etc.}\right\}.\int dt.c^{-t^2}$$
$$+ \frac{Y}{2}.c^{-T^2}.\left\{\frac{d.U^2}{dx} - T.\frac{d^2.U^3}{1.2.dx^2} + (T^2+1).\frac{d^3.U^4}{1.2.3.dx^3} - \text{etc.}\right\}; \quad (2)$$
$$- \frac{Y}{2}.c^{-T'^2}.\left\{\frac{d.U^2}{dx} + T'.\frac{d^2.U^3}{1.2.dx^2} + (T'^2+1).\frac{d^3.U^4}{1.2.3.dx^3} + \text{etc.}\right\}$$

DES PROBABILITÉS. 365

v est égal à $\frac{x-a}{\sqrt{\log Y - \log y}}$, et U, $\frac{d.U^2}{dx}$, $\frac{d^2.U^3}{dx^2}$, etc. sont ce que deviennent v, $\frac{d.v^2}{dx}$, $\frac{d^2.v^3}{dx^2}$, etc., lorsqu'on y change après les différentiations, x en a, a étant la valeur de x qui rend y un *maximum*: T est égal à ce que devient la fonction $\sqrt{\log Y - \log y}$, lorsqu'on change x en $a - \theta$ dans y, et T' est ce que devient la même fonction, lorsqu'on y change x dans $a + \theta'$. L'expression précédente de $\int y\,dx$ donne la valeur de cette intégrale, dans les limites $x = a - \theta$ et $x = a + \theta'$; l'intégrale $\int dt.c^{-t^2}$ étant prise depuis $t = -T$ jusqu'à $t = T'$.

Le plus souvent, aux limites de l'intégrale $\int y\,dx$, étendue depuis $x = 0$ jusqu'à $x = 1$, y est nul; ou lorsque y n'est pas nul, il devient si petit à ces limites, qu'on peut le supposer nul. Alors, on peut faire à ces limites T et T' infinis, ce qui donne pour l'intégrale $\int y\,dx$, étendue depuis $x = 0$ jusqu'à $x = 1$,

$$\int y\,dx = Y.\left\{U + \frac{1}{2}.\frac{d^2.U^3}{1.2.dx^2} + \frac{1.3}{2^2}.\frac{d^4.U^5}{1.2.3.4.dx^4} + \text{etc.}\right\}.\sqrt{\pi};$$

ainsi la probabilité que la valeur de x est comprise dans les limites $x = a - \theta$ et $x = a + \theta'$, est égale à

$$\frac{\int dt.c^{-t^2}}{\sqrt{\pi}} + \frac{\left\{\begin{array}{l} \frac{1}{2}.c^{-T^2}.\left\{\frac{d.U^2}{dx} - T.\frac{d^2.U^3}{1.2.dx^2} + (T^2+1).\frac{d^3.U^4}{1.2.3.dx^3} - \text{etc.}\right\} \\ -\frac{1}{2}.c^{-T'^2}.\left\{\frac{d.U^2}{dx} + T'.\frac{d^2.U^3}{1.2.dx^2} + (T'^2+1).\frac{d^3.U^4}{1.2.3.dx^3} + \text{etc.}\right\} \end{array}\right\}}{\left\{U + \frac{1}{2}.\frac{d^2.U^3}{1.2.dx^2} + \frac{1.3}{2^2}.\frac{d^4.U^5}{1.2.3.4.dx^4} + \text{etc.}\right\}.\sqrt{\pi}}; \quad (3)$$

On voit par le n° 23 du premier Livre, que dans le cas où y a pour facteurs, des fonctions de x élevées à de grandes puissances de l'ordre $\frac{1}{\alpha}$, α étant une fraction extrêmement petite, alors U est le plus souvent de l'ordre $\sqrt{\alpha}$, ainsi que ses différences successives; U, $\frac{d.U^2}{dx}$, $\frac{d^2.U^3}{dx^2}$, etc. sont respectivement des ordres $\sqrt{\alpha}$, α, $\alpha^{\frac{3}{2}}$, etc.; d'où il suit que la convergence des séries de la formule (3), exige que T et T' ne soient pas d'un ordre supérieur à $\frac{1}{\sqrt{\alpha}}$.

The Laplace Approximation

formally

- Consider a probability distribution $p(\theta)$ (may be a posterior $p(\theta \mid D)$ or something else)
- find a (local) **maximum** of $p(\theta)$ or (equivalently) $\log p(\theta)$

$$\hat{\theta} = \arg\max \log p(\theta) \qquad \Rightarrow \qquad \nabla \log p(\hat{\theta}) = 0$$

- perform **second order Taylor expansion** around $\theta = \hat{\theta} + \delta$ in log space

$$\log p(\delta) = \log p(\hat{\theta}) + \frac{1}{2}\delta^{\intercal} \left(\underbrace{\nabla\nabla^{\intercal} \log p(\hat{\theta})}_{=:\Psi} \right) \delta + \mathcal{O}(\delta^3)$$

- define **the Laplace approximation** q **to** p

$$q(\theta) = \mathcal{N}(\theta; \hat{\theta}, -\Psi^{-1})$$

- Note that, if $p(\theta) = \mathcal{N}(\theta; m, \Sigma)$, then $p(\theta) = q(\theta)$

The Laplace Approximation for GP Classification

conceptual step (implementation details coming up)

[based on Rasmussen & Williams, 2006, §3.4]

- Find maximum posterior probability for **latent** f at **training points**

$$\hat{\mathbf{f}} = \arg\max \log p(\mathbf{f}_X \mid y)$$

- Assign approximate Gaussian posterior at training points

$$q(\mathbf{f}_X) = \mathcal{N}(\mathbf{f}_X; \hat{\mathbf{f}}, -(\nabla\nabla^\intercal \log p(\mathbf{f}_X \mid y)|_{\mathbf{f}_X = \hat{\mathbf{f}}})^{-1}) =: \mathcal{N}(\mathbf{f}_X; \hat{\mathbf{f}}, \hat{\Sigma})$$

- approximate posterior **predictions** at f_x for **latent function**

$$q(f_x \mid y) = \int p(f_x \mid \mathbf{f}_X) q(\mathbf{f}_X)\, d\mathbf{f}_X = \int \mathcal{N}(f_x; m_x + k_{xX} K_{XX}^{-1}(\mathbf{f}_X - m_X), k_{xx} - k_{xX} K_{XX}^{-1} k_{Xx}) q(\mathbf{f}_X)\, d\mathbf{f}_X$$

$$= \mathcal{N}(f_x;\; m_x + k_{xX} K_{XX}^{-1}(\hat{\mathbf{f}} - m_X),\; k_{xx} - k_{xX} K_{XX}^{-1} k_{Xx} + k_{xX} K_{XX}^{-1} \hat{\Sigma} K_{XX}^{-1} k_{Xx})$$

Compare with *exact* predictions

$$\mathbb{E}_{p(f_x, f_X \mid y)}(f_x) = \int (\mathbb{E}_{p(f_x \mid f_X)}(f_x)) p(f_X \mid y)\, df_X = m_x + k_{xX} K_{XX}^{-1}(\mathbb{E}_{p(f_X \mid y)}(f_X) - m_X) =: \bar{f}_x$$

Recall: $p(x) = \mathcal{N}(x; m, V), p(z \mid x) = \mathcal{N}(z; Ax, B) \;\Rightarrow\; p(z) = \int p(z \mid x) p(x)\, dx = \mathcal{N}(z; Am, AVA^\intercal + B)$.

The Laplace Approximation for GP Classification

conceptual step (implementation details coming up)

[based on Rasmussen & Williams, 2006, §3.4]

- Find maximum posterior probability for **latent** f at **training points**

$$\hat{\boldsymbol{f}} = \arg\max \log p(\boldsymbol{f}_X \mid y)$$

- Assign approximate Gaussian posterior at training points

$$q(\boldsymbol{f}_X) = \mathcal{N}(\boldsymbol{f}_X; \hat{\boldsymbol{f}}, -(\nabla\nabla^\intercal \log p(\boldsymbol{f}_X \mid y)|_{\boldsymbol{f}_X=\hat{\boldsymbol{f}}})^{-1}) =: \mathcal{N}(\boldsymbol{f}_X; \hat{\boldsymbol{f}}, \hat{\Sigma})$$

- approximate posterior **predictions** at f_x for **latent function**

$$q(f_x \mid y) = \int p(f_x \mid \boldsymbol{f}_X) q(\boldsymbol{f}_X)\, d\boldsymbol{f}_X = \int \mathcal{N}(f_x; m_x + k_{xX}K_{XX}^{-1}(\boldsymbol{f}_X - m_X), k_{xx} - k_{xX}K_{XX}^{-1}k_{Xx}) q(\boldsymbol{f}_X)\, d\boldsymbol{f}_X$$

$$= \mathcal{N}(f_x;\ m_x + k_{xX}K_{XX}^{-1}(\hat{\boldsymbol{f}} - m_X),\ k_{xx} - k_{xX}K_{XX}^{-1}k_{Xx} + k_{xX}K_{XX}^{-1}\hat{\Sigma}K_{XX}^{-1}k_{Xx})$$

Compare with *exact* predictions

$$\text{var}_{p(f_x, f_X \mid y)}(f_x) = \int (f_x - \bar{f}_x)^2\, dp(f_x \mid f_X)\, dp(f_X) = k_{xx} - k_{xX}K_{XX}^{-1}k_{Xx} + k_{xX}K_{XX}^{-1}\text{var}_{p(f_X \mid y)}(f_X)K_{XX}^{-1}k_{Xx})$$

Recall: $p(x) = \mathcal{N}(x; m, V), p(z \mid x) = \mathcal{N}(z; Ax, B) \Rightarrow p(z) = \int p(z \mid x)p(x)\, dx = \mathcal{N}(z; Am, AVA^\intercal + B).$

The Laplace Approximation for GP Classification

conceptual step (implementation details coming up)

[based on Rasmussen & Williams, 2006, §3.4]

- Find maximum posterior probability for **latent** f at **training points**

$$\hat{\boldsymbol{f}} = \arg\max \log p(\boldsymbol{f}_X \mid y)$$

- Assign approximate Gaussian posterior at training points

$$q(\boldsymbol{f}_X) = \mathcal{N}(\boldsymbol{f}_X; \hat{\boldsymbol{f}}, -(\nabla\nabla^\intercal \log p(\boldsymbol{f}_X \mid y)|_{\boldsymbol{f}_X = \hat{\boldsymbol{f}}})^{-1}) =: \mathcal{N}(\boldsymbol{f}_X; \hat{\boldsymbol{f}}, \hat{\Sigma})$$

- approximate posterior **predictions** at f_x for **latent function**

$$q(f_x \mid y) = \int p(f_x \mid \boldsymbol{f}_X) q(\boldsymbol{f}_X)\, d\boldsymbol{f}_X = \int \mathcal{N}(f_x; m_x + k_{xX}K_{XX}^{-1}(\boldsymbol{f}_X - m_X), k_{xx} - k_{xX}K_{XX}^{-1}k_{Xx}) q(\boldsymbol{f}_X)\, d\boldsymbol{f}_X$$

$$= \mathcal{N}(f_x;\ m_x + k_{xX}K_{XX}^{-1}(\hat{\boldsymbol{f}} - m_X),\ k_{xx} - k_{xX}K_{XX}^{-1}k_{Xx} + k_{xX}K_{XX}^{-1}\hat{\Sigma}K_{XX}^{-1}k_{Xx})$$

- compute predictions for **label probabilities**:

$$\mathbb{E}_{p(f|y)}[\pi_x] \approx \mathbb{E}_q[\pi_x] = \int \sigma(f_x) q(f_x \mid y)\, df_x \quad \text{or (not the same!)} \quad \hat{\pi}_x = \sigma(\mathbb{E}_q(f_x))$$

- the Laplace approximation is only very roughly motivated (see above)
- it can be **arbitrarily wrong**, since it is a **local** approximation
- but it is often among the most computationally efficient things to try
- for logistic regression, it tends to work relatively well, because
  - the log posterior is concave (see below)
  - the algebraic structure of the link function yields “almost” a Gaussian posterior (cf. picture above)

$$p(f) = \mathcal{GP}(f, m, k) \qquad p(\boldsymbol{y} \mid \boldsymbol{f}_X) = \prod_{i=1}^{n} \sigma(y_i f_{x_i}) \qquad \sigma(z) = \frac{1}{1 + e^{-x}}$$

$$\log p(\boldsymbol{f}_X \mid \boldsymbol{y}) = \log p(\boldsymbol{y} \mid \boldsymbol{f}_X) + \log p(\boldsymbol{f}_X) - \log p(\boldsymbol{y}) \quad \text{with} \quad \log \sigma(y_i f_{x_i}) = -\log(1 + e^{-y_i f_{x_i}})$$

$$= \sum_{i=1}^{n} \log \sigma(y_i f_{x_i}) - \frac{1}{2}(\boldsymbol{f}_X - \boldsymbol{m}_X)^\intercal K_{XX}^{-1}(\boldsymbol{f}_X - \boldsymbol{m}_X) + \text{const.}$$

$$\nabla \log p(\boldsymbol{f}_X \mid \boldsymbol{y}) = \sum_{i=1}^{n} \nabla \log \sigma(y_i f_{x_i}) - K_{XX}^{-1}(\boldsymbol{f}_X - \boldsymbol{m}_X) \quad \text{with} \quad \frac{\partial \log \sigma(y_i f_{x_i})}{\partial f_{x_j}} = \delta_{ij} \left(\frac{y_i + 1}{2} - \sigma(f_{x_i}) \right)$$

$$\nabla\nabla^\intercal \log p(\boldsymbol{f}_X \mid \boldsymbol{y}) = \sum_{i=1}^{n} \nabla\nabla^\intercal \log \sigma(y_i f_{x_i}) - K_{XX}^{-1} \quad \text{with} \quad \frac{\partial^2 \log \sigma(y_i f_{x_i})}{\partial f_{x_a} \partial f_{x_b}} = -\delta_{ia}\delta_{ib} \underbrace{\sigma(f_{x_i})(1 - \sigma(f_{x_i}))}_{=: w_i \text{ with } 0 < w_i < 1}$$

$$=: -\operatorname{diag} \boldsymbol{w} - K^{-1} = -(W + K^{-1}) \qquad \leftarrow \text{convex minimization / concave maximization!}$$

Implementing the Laplace Approximation I

Newton Optimization [Rasmussen & Williams, 2006, §3.4]

```
procedure OPTIMIZE(L(·), f_0)
    f ← f_0                                   // initialize
    while not converged do
        g ← ∇L(f)                             // compute gradient
        H ← (∇∇ᵀL(f))^{-1}                    // compute inverse Hessian
        Δ ← Hg                                // Newton step
        f ← f − Δ                             // perform step
        converged ← ‖Δ‖ < ε                   // check for convergence
    end while
    return f
end procedure
```

Implementing the Laplace Approximation II

Concrete Algorithm, preliminary form

[Rasmussen & Williams, 2006, §3.4]

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

procedure GP-LOGISTIC-TRAIN($K_{XX}, m_X, \boldsymbol{y}$)
    $f \leftarrow m_X$ // initialize
    while not converged **do**
        $r \leftarrow \frac{y+1}{2} - \sigma(f)$ // $= \nabla \log p(\boldsymbol{y} \mid f_X)$, gradient of log likelihood
        $W \leftarrow \mathrm{diag}(\sigma(f) \odot (1 - \sigma(f)))$ // $= -\nabla\nabla \log p(\boldsymbol{y} \mid f_X)$, Hessian of log likelihood
        $g \leftarrow r - K_{XX}^{-1}(f - m_X)$ // compute gradient
        $H \leftarrow -(W + K^{-1})^{-1}$ // compute inverse Hessian
        $\Delta \leftarrow Hg$ // Newton step
        $f \leftarrow f - \Delta$ // perform step
        converged $\leftarrow \|\Delta\| < \epsilon$ // check for convergence
    end while
    return f
end procedure

This can be numerically unstable as it (repeatedly) requires $(W + K^{-1})^{-1}$. For a numerically stable alternative, use $B := I + W^{1/2} K_{XX} W^{1/2}$ (cf. Rasmussen & Williams).

Computing Predictions

from the Laplace approximation

[Rasmussen & Williams, 2006, §3.4]

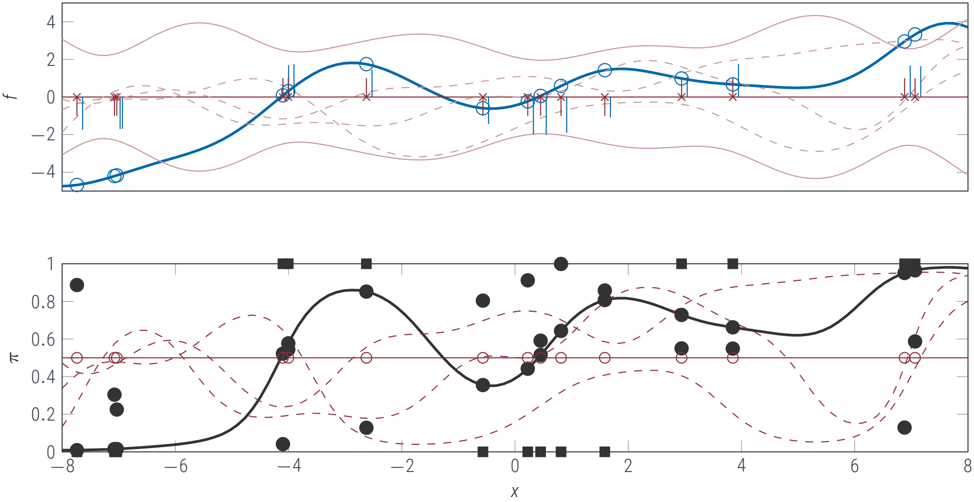
$$\log p(\boldsymbol{f}_X \mid \boldsymbol{y}) = \sum_{i=1}^{n} \log \sigma(y_i f_{x_i}) - \frac{1}{2}(\boldsymbol{f}_X - \boldsymbol{m}_X)^\intercal K_{XX}^{-1}(\boldsymbol{f}_X - \boldsymbol{m}_X) + \text{const.}$$

$$\nabla \log p(\boldsymbol{f}_X \mid \boldsymbol{y}) = \underbrace{\sum_{i=1}^{n} \nabla \log \sigma(y_i f_{x_i})}_{=:r} - K_{XX}^{-1}(\boldsymbol{f}_X - \boldsymbol{m}_X) \quad \text{with} \quad \frac{\partial \log \sigma(y_i f_{x_i})}{\partial f_{x_j}} = \delta_{ij}\left(\frac{y_i + 1}{2} - \sigma(f_{x_i})\right)$$

```
procedure GP-LOGISTIC-PREDICT(f̂, W, R, r, k, x)        // f̂, W, R = Cholesky(B), r handed over from training
    for i = 1, ..., LENGTH(x) do
        f̄_i ← k_{x_i X} r                              // mean prediction (note at minimum, 0 = ∇p(f_X | y) = r − K_XX^{-1}(f_X − m_X)).
        s ← R^{-1}(W^{1/2} k_{X x_i})                   // pre-computation allows this step in O(n^2)
        v ← k_{x_i x_i} − s^T s                         // v = cov(f_x)
        π̄_i ← ∫ σ(f_i) N(f_i, f̄_i, v) df_i             // predictive probability for class 1 is p(y | f̄) = ∫ p(y_x | f_x) p(f_x | f̄) df_x
    end for                                             // entire loop is O(n^2 m) for m test cases.
    return π̄_X
end procedure
```

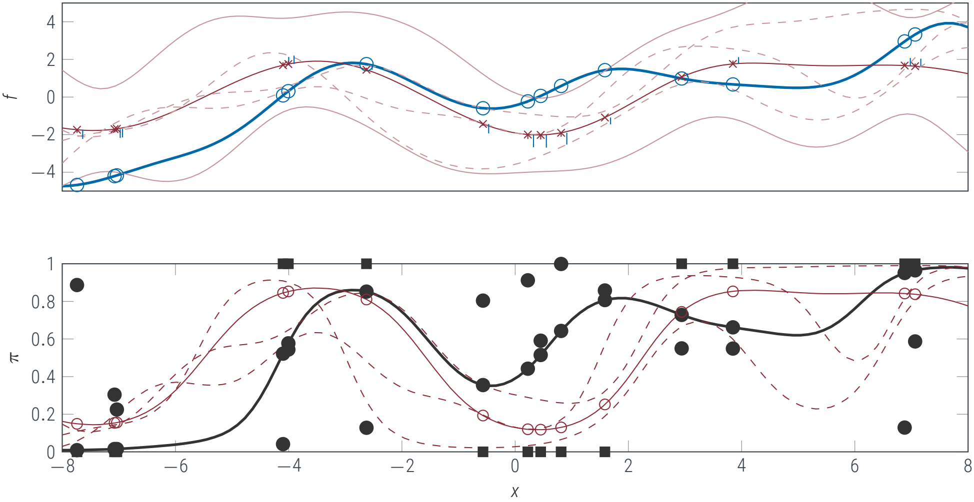
Pictorial View

Gaussian process logistic regression



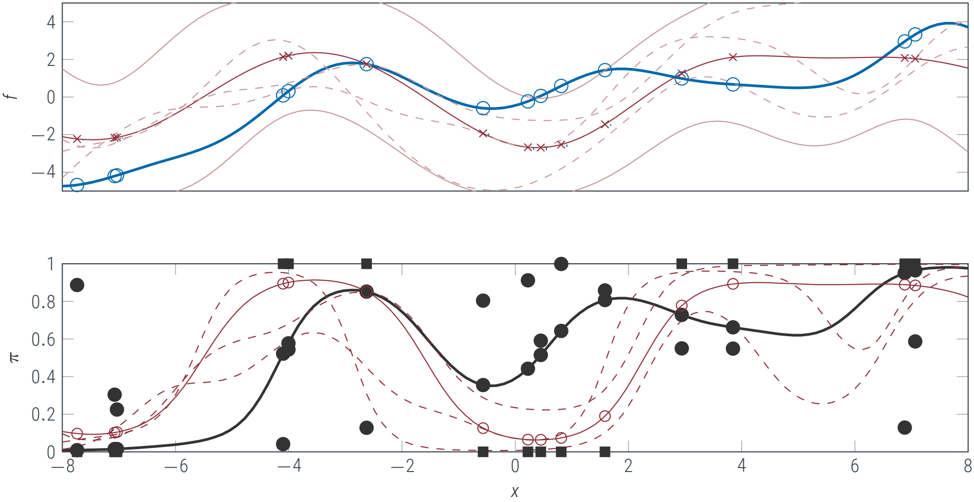
Pictorial View

Gaussian process logistic regression



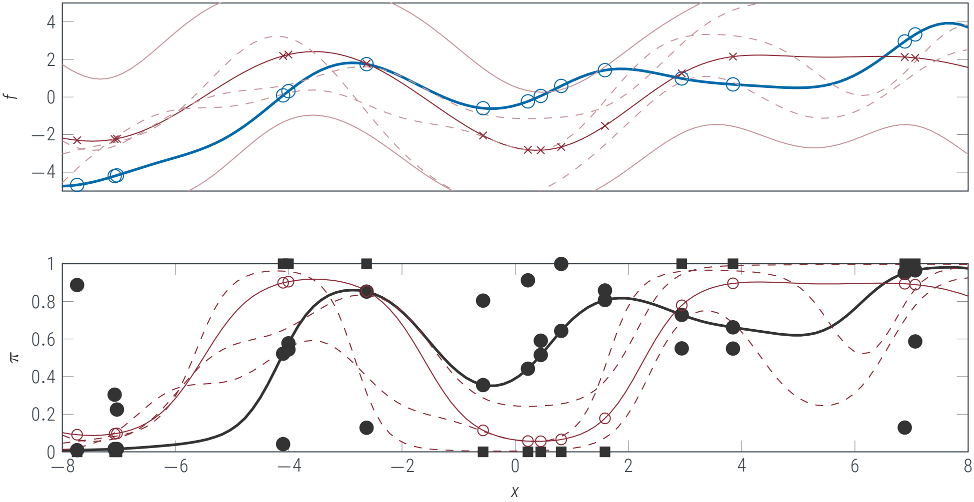
Pictorial View

Gaussian process logistic regression



Pictorial View

Gaussian process logistic regression



EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Gaussian Process Classification — (Probabilistic) Logistic Regression:

- Supervised classification phrased in a **discriminative** model with probabilistic interpretation
- model binary outputs as a **transformation** of a **latent function** with a Gaussian process prior
- due to **non-Gaussian likelihood**, the posterior is non-Gaussian; exact inference **intractable**
- **Laplace approximation:** Find MAP estimator, second order expansion for Gaussian approximation
- tune code for numerical stability, efficient computations
- Laplace approximation provides Gaussian posterior on training points, hence evidence, predictions