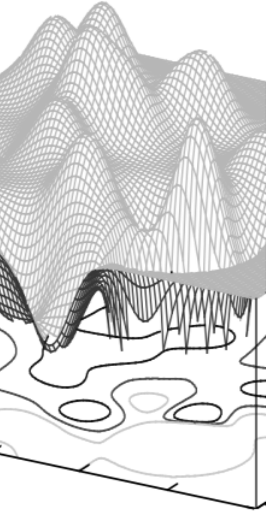


Machine Learning

Probabilistic Machine Learning

learning as inference, Bayesian Kernel Ridge regression = Gaussian Processes, Bayesian Kernel Logistic Regression = GP classification, Bayesian Neural Networks

Marc Toussaint
University of Stuttgart

Summer 2019

Learning as Inference

- The parameteric view

$$P(\beta|\text{Data}) = \frac{P(\text{Data}|\beta)\; P(\beta)}{P(\text{Data})}$$

- The function space view

$$P(f|\text{Data}) = \frac{P(\text{Data}|f)\; P(f)}{P(\text{Data})}$$

- Today:
  - Bayesian (Kernel) Ridge Regression $\leftrightarrow$ Gaussian Process (GP)
  - Bayesian (Kernel) Logistic Regression $\leftrightarrow$ GP classification
  - Bayesian Neural Networks (briefly)

- Beyond learning about specific Bayesian learning methods:

  Understand relations between

$$\text{loss/error} \quad \leftrightarrow \quad \text{neg-log likelihood}$$

$$\text{regularization} \quad \leftrightarrow \quad \text{neg-log prior}$$

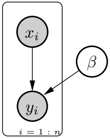
$$\text{cost (reg.+loss)} \quad \leftrightarrow \quad \text{neg-log posterior}$$

Gaussian Process = Bayesian (Kernel) Ridge Regression

Ridge regression as Bayesian inference

- We have random variables $X_{1:n}, Y_{1:n}, \beta$

- We observe data $D = \{(x_i, y_i)\}_{i=1}^n$ and want to compute $P(\beta \mid D)$



- Let's assume:
  $P(X)$ is arbitrary
  $P(\beta)$ is Gaussian: $\beta \sim \mathcal{N}(0, \frac{\sigma^2}{\lambda}) \propto e^{-\frac{\lambda}{2\sigma^2} \|\beta\|^2}$
  $P(Y \mid X, \beta)$ is Gaussian: $y = x^\top \beta + \epsilon \;\; , \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$

Ridge regression as Bayesian inference

- Bayes' Theorem:

$$P(\beta \mid D) = \frac{P(D \mid \beta)\ P(\beta)}{P(D)}$$

$$P(\beta \mid x_{1:n}, y_{1:n}) = \frac{\prod_{i=1}^{n} P(y_i \mid \beta, x_i)\ P(\beta)}{Z}$$

$P(D \mid \beta)$ is a *product* of independent likelihoods for each observation (x_i, y_i)

Ridge regression as Bayesian inference

- Bayes' Theorem:

$$P(\beta \mid D) = \frac{P(D \mid \beta)\ P(\beta)}{P(D)}$$

$$P(\beta \mid x_{1:n}, y_{1:n}) = \frac{\prod_{i=1}^n P(y_i \mid \beta, x_i)\ P(\beta)}{Z}$$

$P(D \mid \beta)$ is a *product* of independent likelihoods for each observation (x_i, y_i)

Using the Gaussian expressions:

$$P(\beta \mid D) = \frac{1}{Z'} \prod_{i=1}^n e^{-\frac{1}{2\sigma^2}(y_i - x_i^\top \beta)^2}\ e^{-\frac{\lambda}{2\sigma^2}\|\beta\|^2}$$

Ridge regression as Bayesian inference

- Bayes' Theorem:

$$P(\beta \mid D) = \frac{P(D \mid \beta)\ P(\beta)}{P(D)}$$

$$P(\beta \mid x_{1:n}, y_{1:n}) = \frac{\prod_{i=1}^n P(y_i \mid \beta, x_i)\ P(\beta)}{Z}$$

$P(D \mid \beta)$ is a *product* of independent likelihoods for each observation (x_i, y_i)

Using the Gaussian expressions:

$$P(\beta \mid D) = \frac{1}{Z'} \prod_{i=1}^n e^{-\frac{1}{2\sigma^2}(y_i - x_i^\top \beta)^2}\ e^{-\frac{\lambda}{2\sigma^2}\|\beta\|^2}$$

$$-\log P(\beta \mid D) = \frac{1}{2\sigma^2}\Big[\sum_{i=1}^n (y_i - x_i^\top \beta)^2 + \lambda\|\beta\|^2\Big] + \log Z'$$

$$\boxed{-\log P(\beta \mid D) \propto L^{\text{ridge}}(\beta)}$$

1st insight: The *neg-log posterior* $P(\beta \mid D)$ is proportional to the cost function $L^{\text{ridge}}(\beta)$!

Ridge regression as Bayesian inference

- Let us compute $P(\beta \,|\, D)$ explicitly:

$$
\begin{aligned}
P(\beta \,|\, D) &= \frac{1}{Z'} \prod_{i=1}^n e^{-\frac{1}{2\sigma^2}\ (y_i - x_i^\top \beta)^2}\ e^{-\frac{\lambda}{2\sigma^2}\, \|\beta\|^2} \\
&= \frac{1}{Z'}\ e^{-\frac{1}{2\sigma^2}\ \sum_i (y_i - x_i^\top \beta)^2}\ e^{-\frac{\lambda}{2\sigma^2}\, \|\beta\|^2} \\
&= \frac{1}{Z'}\ e^{-\frac{1}{2\sigma^2}\, [(y - X\beta)^\top (y - X\beta) + \lambda \beta^\top \beta]} \\
&= \frac{1}{Z'}\ e^{-\frac{1}{2}\, [\frac{1}{\sigma^2}\, y^\top y + \frac{1}{\sigma^2}\, \beta^\top (X^\top X + \lambda \mathbf{I}) \beta - \frac{2}{\sigma^2}\, \beta^\top X^\top y]} \\
&= \mathcal{N}(\beta \,|\, \hat\beta, \Sigma)
\end{aligned}
$$

This is a Gaussian with covariance and mean

$$\Sigma = \sigma^2\ (X^\top X + \lambda \mathbf{I})^{\text{-}1}\ , \quad \hat\beta = \tfrac{1}{\sigma^2}\ \Sigma X^\top y = (X^\top X + \lambda \mathbf{I})^{\text{-}1} X^\top y$$

- **2nd insight:** The mean $\hat\beta$ is exactly the classical $\operatorname{argmin}_\beta L^{\text{ridge}}(\beta)$.
- **3rd insight:** The Bayesian approach not only gives a mean/optimal $\hat\beta$, but also a variance Σ of that estimate. **(Cp. slide 02:13!)**

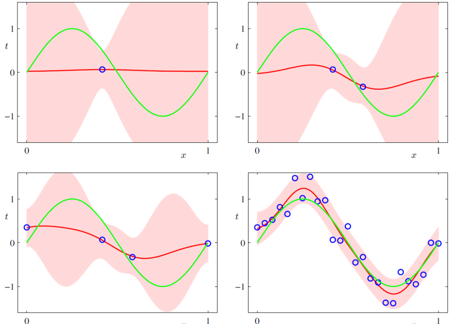
Predicting with an uncertain β

- Suppose we want to make a prediction at x. We can compute the **predictive distribution** over a new observation y^* at x^*:

$$\begin{aligned} P(y^* \mid x^*, D) &= \int_\beta P(y^* \mid x^*, \beta)\ P(\beta \mid D)\ d\beta \\ &= \int_\beta \mathcal{N}(y^* \mid \phi(x^*)^\top \beta, \sigma^2)\ \mathcal{N}(\beta \mid \hat\beta, \Sigma)\ d\beta \\ &= \mathcal{N}(y^* \mid \phi(x^*)^\top \hat\beta,\ \sigma^2 + \phi(x^*)^\top \Sigma \phi(x^*)) \end{aligned}$$

Note, for $f(x) = \phi(x)^\top \beta$, we have $P(f(x) \mid D) = \mathcal{N}(f(x) \mid \phi(x)^\top \hat\beta,\ \phi(x)^\top \Sigma \phi(x))$ without the σ^2

- So, y^* is Gaussian distributed around the mean prediction $\phi(x^*)^\top \hat\beta$:



Wrapup of Bayesian Ridge regression

- **1st insight:** The *neg-log posterior* $P(\beta \mid D)$ is equal to the cost function $L^{\text{ridge}}(\beta)$.

  This is a very very common relation: optimization costs correspond to neg-log probabilities; probabilities correspond to exp-neg costs.

- **2nd insight:** The mean $\hat\beta$ is exactly the classical $\text{argmin}_\beta\, L^{\text{ridge}}(\beta)$

  More generally, the most likely parameter $\text{argmax}_\beta\, P(\beta|D)$ is also the least-cost parameter $\text{argmin}_\beta\, L(\beta)$. In the Gaussian case, most-likely β is also the mean.

- **3rd insight:** The Bayesian inference approach not only gives a mean/optimal $\hat\beta$, but also a variance Σ of that estimate

  This is a core benefit of the Bayesian view: It naturally provides a probability distribution over predictions (*"error bars"*), not only a single prediction.

Kernel Bayesian Ridge Regression

- As in the classical case, we can consider arbitrary features $\phi(x)$
- .. or directly use a kernel $k(x, x')$:

$$
\begin{aligned}
P(f(x) \mid D) &= \mathcal{N}(f(x) \mid \phi(x)^\top \hat\beta,\ \phi(x)^\top \Sigma \phi(x)) \\
\phi(x)^\top \hat\beta &= \phi(x)^\top X^\top (XX^\top + \lambda \mathbf{I})^{\text{-}1} y \\
&= \kappa(x)(K + \lambda \mathbf{I})^{\text{-}1} y \\
\phi(x)^\top \Sigma \phi(x) &= \phi(x)^\top \sigma^2\ (X^\top X + \lambda \mathbf{I})^{\text{-}1} \phi(x) \\
&= \frac{\sigma^2}{\lambda} \phi(x)^\top \phi(x) - \frac{\sigma^2}{\lambda} \phi(x)^\top X^\top (XX^\top + \lambda \mathbf{I}_k)^{\text{-}1} X \phi(x) \\
&= \frac{\sigma^2}{\lambda} k(x, x) - \frac{\sigma^2}{\lambda} \kappa(x)(K + \lambda \mathbf{I}_n)^{\text{-}1} \kappa(x)^\top
\end{aligned}
$$

3rd line: As on slide 05:2
2nd to last line: Woodbury identity $(A + UBV)^{\text{-}1} = A^{\text{-}1} - A^{\text{-}1} U (B^{\text{-}1} + V A^{\text{-}1} U)^{\text{-}1} V A^{\text{-}1}$
with $A = \lambda \mathbf{I}$

- In standard conventions $\lambda = \sigma^2$, i.e. $P(\beta) = \mathcal{N}(\beta | 0, 1)$
  – Regularization: scale the covariance function (or features)

Gaussian Processes

are equivalent to Kernelized Bayesian Ridge Regression
(see also Welling: "Kernel Ridge Regression" Lecture Notes; Rasmussen & Williams sections 2.1 & 6.2; Bishop sections 3.3.3 & 6)

- But it is insightful to introduce them again from the "function space view": GPs define a probability distribution over functions; they are the infinite dimensional generalization of Gaussian vectors

Gaussian Processes – function prior

- The function space view

$$P(f|D) = \frac{P(D|f)\; P(f)}{P(D)}$$

- A Gaussian Processes **prior** $P(f)$ defines a probability distribution over functions:
  - A function is an infinite dimensional thing – how could we define a Gaussian distribution over functions?
  - For every finite set $\{x_1, .., x_M\}$, the function values $f(x_1), .., f(x_M)$ are Gaussian distributed with mean and covariance

$$\begin{aligned} \mathsf{E}\{f(x_i)\} &= \mu(x_i) \qquad \text{(often zero)} \\ \mathsf{cov}\{f(x_i), f(x_j)\} &= k(x_i, x_j) \end{aligned}$$

    Here, $k(\cdot,\cdot)$ is called **covariance function**

- Second, for Gaussian Processes we typically have a Gaussian **data likelihood** $P(D|f)$, namely

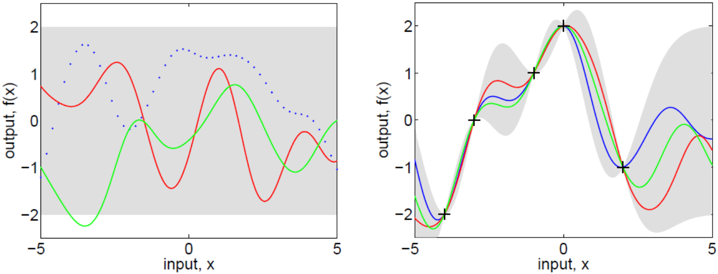
$$P(y \,|\, x, f) = \mathcal{N}(y \,|\, f(x), \sigma^2)$$

Gaussian Processes – function posterior

- The **posterior** $P(f|D)$ is also a Gaussian Process, with the following mean of $f(x)$, covariance of $f(x)$ and $f(x')$: (based on slide 10 (with $\lambda = \sigma^2$))

$$\mathsf{E}\{f(x) \,|\, D\} = \kappa(x)(K + \lambda \mathbf{I})^{\text{-}1} y \; + \; \mu(x)$$
$$\mathsf{cov}\{f(x), f(x') \,|\, D\} = k(x,x') - \kappa(x')(K + \lambda \mathbf{I}_n)^{\text{-}1} \kappa(x')^\top$$

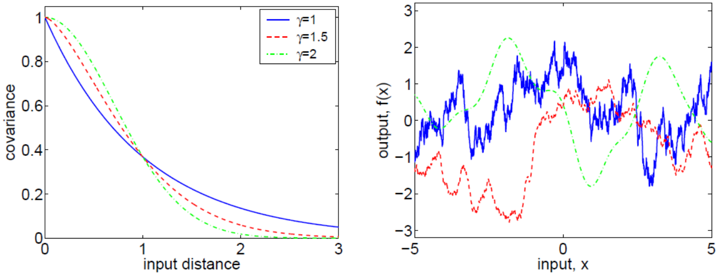
Gaussian Processes



(a), prior (b), posterior

(from Rasmussen & Williams)

GP: different covariance functions

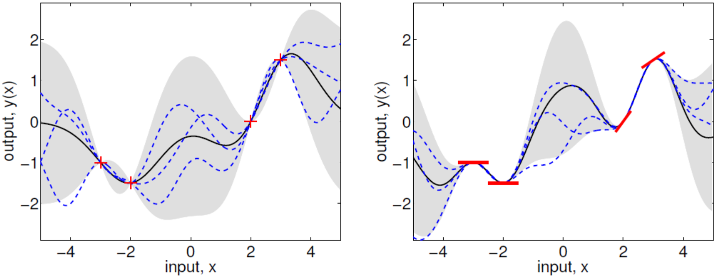


(from Rasmussen & Williams)

- These are examples from the γ-exponential covariance function

$$k(x, x') = \exp\{-|(x - x')/l|^{\gamma}\}$$

GP: derivative observations



(from Rasmussen & Williams)

- Bayesian Kernel Ridge Regression = Gaussian Process

- GPs have become a standard regression method
- If exact GP is not efficient enough, many approximations exist, e.g. sparse and pseudo-input GPs

GP classification = Bayesian (Kernel) Logistic Regression

Bayesian Logistic Regression (binary case)

- f now defines a discriminative function:

$$
\begin{aligned}
P(X) &= \text{arbitrary} \\
P(\beta) &= \mathcal{N}(\beta|0, \frac{2}{\lambda}) \propto \exp\{-\lambda\|\beta\|^2\} \\
P(Y\!=\!1 \mid X, \beta) &= \sigma(\beta^\top \phi(x))
\end{aligned}
$$

- Recall

$$L^{\text{logistic}}(\beta) = -\sum_{i=1}^{n} \log p(y_i \mid x_i) + \lambda\|\beta\|^2$$

- Again, the parameter posterior is

$$P(\beta|D) \propto P(D \mid \beta)\; P(\beta) \propto \exp\{-L^{\text{logistic}}(\beta)\}$$

Bayesian Logistic Regression

- Use **Laplace approximation** (2nd order Taylor for L) at $\beta^* = \operatorname{argmin}_\beta L(\beta)$:

$$L(\beta) \approx L(\beta^*) + \bar\beta^\top \nabla + \frac{1}{2}\bar\beta^\top H \bar\beta \,, \quad \bar\beta = \beta - \beta^*$$

  At β^* the gradient $\nabla = 0$ and $L(\beta^*) = \text{const}$. Therefore

$$\tilde P(\beta|D) \propto \exp\{-\frac{1}{2}\bar\beta^\top H \bar\beta\}$$
$$\Rightarrow \; P(\beta|D) \approx \mathcal{N}(\beta|\beta^*, H^{\text{-}1})$$

- Then the predictive distribution of the *discriminative function* is also Gaussian!

$$\begin{aligned} P(f(x) \,|\, D) &= \textstyle\int_\beta P(f(x) \,|\, \beta) \; P(\beta \,|\, D) \; d\beta \\ &\approx \textstyle\int_\beta \mathcal{N}(f(x) \,|\, \phi(x)^\top\beta, 0) \; \mathcal{N}(\beta \,|\, \beta^*, H^{\text{-}1}) \; d\beta \\ &= \mathcal{N}(f(x) \,|\, \phi(x)^\top\beta^*, \phi(x)^\top H^{\text{-}1}\phi(x)) \;\; =: \; \mathcal{N}(f(x) \,|\, f^*, s^2) \end{aligned}$$

- The predictive distribution over the label $y \in \{0,1\}$:

$$\begin{aligned} P(y(x){=}1 \,|\, D) &= \textstyle\int_{f(x)} \sigma(f(x)) \; P(f(x)|D) \; df \\ &\approx \sigma((1+s^2\pi/8)^{\text{-}\frac{1}{2}} f^*) \end{aligned}$$

  which uses a probit approximation of the convolution.
  $\rightarrow$ The variance s^2 pushes the predictive class probabilities towards 0.5.

Kernelized Bayesian Logistic Regression

- As with Kernel Logistic Regression, the MAP discriminative function f^* can be found iterating the Newton method $\leftrightarrow$ iterating GP estimation on a *re-weighted* data set.
- The rest is as above.

Kernel Bayesian Logistic Regression

is equivalent to Gaussian Process Classification

- GP classification became a standard classification method, if the prediction needs to be a meaningful probability that takes the *model uncertainty* into account.

Bayesian Neural Networks

Bayesian Neural Networks

- Simple ways to get uncertainty estimates:
  - Train ensembles of networks (e.g. bootstrap ensembles)
  - Treat the output layer fully probabilistic (treat the trained NN body as feature vector $\phi(x)$, and apply Bayesian Ridge/Logistic Regression on top of that)

- Ways to treat NNs inherently Bayesian:
  - Infinite single-layer NN $\rightarrow$ GP (classical work in 80/90ies)
  - Putting priors over weights ("Bayesian NNs", Neil, MacKay, 90ies)
  - Dropout (much more recent, see papers below)

- Read

  Gal & Ghahramani: *Dropout as a bayesian approximation: Representing model uncertainty in deep learning* (ICML'16)

  Damianou & Lawrence: *Deep gaussian processes* (AIS 2013)

Dropout in NNs as Deep GPs

- Deep GPs are essentially a a chaining of Gaussian Processes
  - The mapping from each layer to the next is a GP
  - Each GP could have a different prior (kernel)

- Dropout in NNs
  - Dropout leads to randomized prediction
  - One can estimate the mean prediction from T dropout samples (MC estimate)
  - Or one can estimate the mean prediction by averaging the weights of the network (“standard dropout”)
  - Equally one can MC estimate the variance from samples
  - Gal & Ghahramani show, that a Dropout NN is a Deep GP (with very special kernel), and the “correct” predictive variance is this MC estimate plus $\frac{pl^2}{2n\lambda}$ (kernel length scale l, regularization λ, dropout prob p, and n data points)

No Free Lunch

- Averaged over *all* problem instances, any algorithm performs equally. (E.g. equal to random.)
  - "there is no one model that works best for every problem"

  Igel & Toussaint: *On Classes of Functions for which No Free Lunch Results Hold* (Information Processing Letters 2003)

- Rigorous formulations formalize this "average over *all* problem instances". E.g. by assuming a uniform prior over problems
  - In black-box optimization, a uniform distribution over underlying objective functions $f(x)$
  - In machine learning, a uniform distribution over the hiddern true function $f(x)$

  ... and NLF always considers *non-repeating queries*.

- But what does *uniform distribution over functions mean?*

No Free Lunch

- Averaged over *all* problem instances, any algorithm performs equally. (E.g. equal to random.)
  - "there is no one model that works best for every problem"

  Igel & Toussaint: *On Classes of Functions for which No Free Lunch Results Hold* (Information Processing Letters 2003)

- Rigorous formulations formalize this "average over *all* problem instances". E.g. by assuming a uniform prior over problems
  - In black-box optimization, a uniform distribution over underlying objective functions $f(x)$
  - In machine learning, a uniform distribution over the hiddern true function $f(x)$

  ... and NLF always considers *non-repeating queries*.

- But what does *uniform distribution over functions mean?*

- NLF is trivial: when any previous query yields NO information at all about the results of future queries, anything is exactly as good as random guessing

Conclusions

- Probabilistic inference is a very powerful concept!
  – Inferring about the world given data
  – Learning, decision making, reasoning can view viewed as forms of (probabilistic) inference

- We introduced Bayes' Theorem as the fundamental form of probabilistic inference

- Marrying Bayes with (Kernel) Ridge (Logisic) regression yields
  – Gaussian Processes
  – Gaussian Process classification

- We can estimate uncertainty also for NNs
  – Dropout
  – Probabilistic weights and variational approximations; Deep GPs

- No Free Lunch for ML!