02477 – Bayesian Machine Learning: Lecture 6

Michael Riis Andersen

Technical University of Denmark,
DTU Compute, Department of Applied Math and Computer Science

Outline

1. A bit more theory on covariance functions

2. Gaussian processes in practice

3. Gaussian process classification

4. A bit about neural networks for probabilistic modelling

A bit more theory on covariance functions

The big picture

1. We started with a Bayesian linear model

$$p(\boldsymbol{y}, \boldsymbol{w}) = p(\boldsymbol{y} \mid \boldsymbol{w}) p(\boldsymbol{w})$$

2. We introduced $\boldsymbol{f}$ into the model and marginalized over the weights $\boldsymbol{w}$

$$p(\boldsymbol{y}, \boldsymbol{f}) = \int p(\boldsymbol{y} \mid \boldsymbol{f}) p(\boldsymbol{f} \mid \boldsymbol{w}) p(\boldsymbol{w}) \mathrm{d}\boldsymbol{w} = p(\boldsymbol{y} \mid \boldsymbol{f}) p(\boldsymbol{f})$$

3. This gave us a prior for linear functions in function space $p(\boldsymbol{f})$,

$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f} \mid \mathbf{0}, \alpha^{-1} \boldsymbol{\Phi} \boldsymbol{\Phi}^{T}),$$

where the covariance function for $\boldsymbol{f}$ was given by

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{\alpha} \phi(\boldsymbol{x}_i)^{T} \phi(\boldsymbol{x}_j)$$

4. By changing the form of the covariance function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, we can model much more interesting functions

Notation and characterization

- We'll use the notation

$$f(\boldsymbol{x}) \sim \mathcal{GP}\left(m\left(\boldsymbol{x}\right), k\left(\boldsymbol{x}, \boldsymbol{x}'\right)\right)$$

- A Gaussian process can be considered as a prior distribution over functions $f : \mathcal{X} \to \mathbb{R}$ (the domain $\mathcal{X}$ is typically $\mathbb{R}^D$).

- A Gaussian process is completely characterized by its mean function $m\left(\boldsymbol{x}\right)$ and its covariance function $k\left(\boldsymbol{x}, \boldsymbol{x}'\right)$.

$$\begin{aligned} m\left(\boldsymbol{x}\right) &= \mathbb{E}\left[f\left(\boldsymbol{x}\right)\right] \\ k\left(\boldsymbol{x}, \boldsymbol{x}'\right) &= \mathbb{E}\left[\left(f(\boldsymbol{x}) - m\left(\boldsymbol{x}\right)\right)\left(f(\boldsymbol{x}') - m\left(\boldsymbol{x}'\right)\right)\right] \end{aligned}$$

- This means that $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$ are jointly Gaussian distributed with covariance $k\left(\boldsymbol{x}, \boldsymbol{x}'\right)$.

- Not all functions are valid covariance functions – more on that later.

Covariance functions

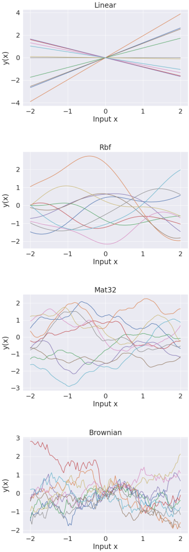
- A covariance function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ maps a pair of inputs $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}$ from some input space $\mathcal{X}$ to the real line $\mathbb{R}$
- Recall: the covariance / kernel matrix is given by

$$\boldsymbol{K}_{ij} = \mathrm{cov}\left(y(\boldsymbol{x}_i), y(\boldsymbol{x})_j\right) = k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$$

- Covariance functions must be symmetric & Positive Semi-Definite such that

$$\begin{aligned} \text{(Symmetric)} \quad & \boldsymbol{K} = \boldsymbol{K}^T \\ \text{(PSD)} \quad & \forall \boldsymbol{x} \neq 0 : \quad \boldsymbol{x}^T \boldsymbol{K} \boldsymbol{x} \geq 0 \end{aligned}$$

- Must hold for all possible data sets $\{\boldsymbol{x}_n\}_{n=1}^N \subset \mathcal{X}$ in the input space $\mathcal{X}$
- Covariance functions as prior information
- Stationary and isotropic covariance functions



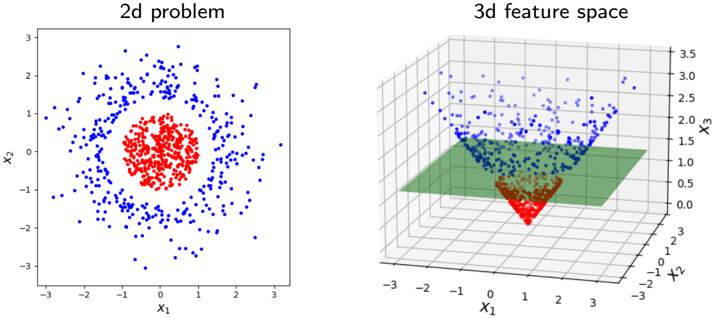
Digression: Feature expansions

- We derived the covariance function from a linear model $f(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$. The choice of feature expansion $\phi(\boldsymbol{x})$ can be crucial.

- **Example**: Binary classification problem in 2d, *not linear separable*

- Embedding $\boldsymbol{x}$ in *higher dimensional space* can make the problem *linear separable*, e.g.

$$\phi(\boldsymbol{x}) = \left[x_1, x_2, \sqrt{x_1^2 + x_2^2}\right]$$



2d problem

3d feature space

Kernels and feature spaces I

- The linear model $f(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$ with Gaussian priors $p(\boldsymbol{w}|\boldsymbol{0}, \alpha^{-1}\boldsymbol{I})$ yields

$$\boldsymbol{K}_{nm} = \mathrm{cov}(f(\boldsymbol{x}_n), f(\boldsymbol{x}_m)) = \alpha^{-1}\phi(\boldsymbol{x}_n)^T\phi(\boldsymbol{x}_m)$$

- Hence, changing the feature expansion $\phi(\boldsymbol{x})$ changes the covariance function.

- What about the other way around? If we adopt some covariance function, does it imply a specific feature expansion?

- Yes! (by *Mercer's theorem*).

Kernels and feature spaces II

- **Example**: Consider the following kernel for $\boldsymbol{x} = [x_1, x_2]^T \in \mathbb{R}^2$

$$
\begin{aligned}
k(\boldsymbol{x}, \boldsymbol{x}') &= (\boldsymbol{x}^T \boldsymbol{x}')^2 \\
&= (x_1 x_1' + x_2 x_2')^2 \\
&= x_1^2 (x_1')^2 + x_2^2 (x_2')^2 + 2 x_1 x_1' x_2 x_2' \\
&= \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2} x_1 x_2 \end{bmatrix}^T \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2} x_1 x_2 \end{bmatrix}
\end{aligned}
$$

- Hence, this kernel is equivalent to embedding the 2d point $\boldsymbol{x}$ in a 3D feature space, but we never explicitly construct the 3d feature representation

- How about the squared exponential kernel?

$$
k(\boldsymbol{x}, \boldsymbol{x}') = \kappa^2 \exp\left(- \frac{||\boldsymbol{x} - \boldsymbol{x}'||^2}{2\ell^2} \right)
$$

- One can show that the implicit feature space for the squared exponential kernel is *infinite dimensional*

Gaussian processes in practice

Gaussian processes in practice

- GPs requires careful implementation to make robust and to make it scale

- Frameworks for easy Gaussian process modelling
  1. GPy
  2. GPflow
  3. GPytorch
  4. GPJax
  5. BRMS (MC Stan)
  6. ...

- Approximations and computational tools for scaling GPs to millions of data points
  1. Exact inference
  2. Variational approximation and inducing points
  3. KISS (Kernel interpolation for structured Gaussian)
  4. Basis functions approximations
  5. ...

- Regression, classification, latent variable models...

Gaussian process classification

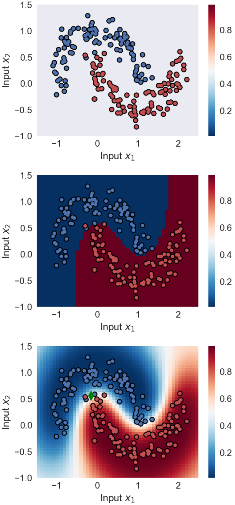
Why Gaussian processes for classification?

- Complex decision boundaries

  1. Non-linear boundary
  2. Can learn complexity of decision boundary from data

- Probabilistic classification

  1. How would you classify the green point?
  2. We want to model both epistemic and aleatoric uncertainty, while using highly flexible models



Gaussian processes for classification I

- Bayesian model for logistic regression

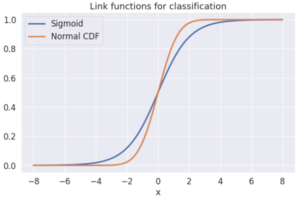
$$\begin{aligned} y_n &\sim \text{Ber}(\sigma(f(\boldsymbol{x}_n))) \\ f(\boldsymbol{x}_n) &= \phi(\boldsymbol{x_n})^T \boldsymbol{w} \\ \boldsymbol{w} &\sim \mathcal{N}\left(\boldsymbol{0}, \alpha^{-1}\boldsymbol{I}\right) \end{aligned}$$

- Gaussian process classification

$$\begin{aligned} y_n &\sim \text{Ber}(\sigma(f(\boldsymbol{x}_n))) \\ f &\sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}')) \end{aligned}$$

- Neural network classification

$$\begin{aligned} y_n &\sim \text{Ber}(\sigma(f(\boldsymbol{x}_n))) \\ f &= \text{NN}(\boldsymbol{x}_n|\boldsymbol{w}) \end{aligned}$$



- The function $\sigma : \mathbb{R} \rightarrow (0, 1)$ is called an *inverse link function*
  1. Sigmoid: $\sigma(x) = \frac{1}{1+\exp(-x)}$
  2. CDF of standard normal: $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(x|0, 1)\mathrm{d}x$

- Sigmoid can be more robust to outliers, but the CDF of the standard normal has appealing computational properties

Gaussian processes for classification II

- Likelihood for logistic regression (using $f_n = f(\boldsymbol{x}_n)$)

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{n=1}^{N} \sigma(f_n)^{y_n} (1 - \sigma(f_n))^{1-y_n}$$

- Our Gaussian process prior

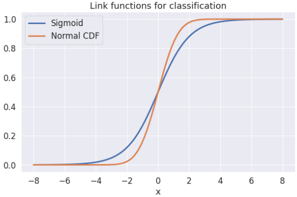
$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K})$$

- For regression we derived the predictive distribution analytically because everything was jointly Gaussian

- The predictive distribution for classification

$$p(y^* = 1|\boldsymbol{y}, \boldsymbol{x}^*) = \int p(y^* = 1|f^*)p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}f^*$$

- ... is again intractable, so we will use Laplace approximations in a 2 step-procedure
  1. Approximate $p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)$ using Laplace
  2. Compute $p(y^*|\boldsymbol{y}, \boldsymbol{x}^*)$



Gaussian processes for classification III

- The predictive distribution for classification

$$p(y^* = 1|\boldsymbol{y}, \boldsymbol{x}^*) = \int p(y^* = 1|f^*)p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}f^*$$

- Re-writing $p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)$

$$\begin{aligned} p(f^*|\boldsymbol{y}, \boldsymbol{x}^*) &= \int p(f^*, \boldsymbol{f}|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}\boldsymbol{f} && \text{(Sum rule)} \\ &= \int p(f^*|\boldsymbol{f}, \boldsymbol{y}, \boldsymbol{x}^*)p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}\boldsymbol{f} && \text{(Product rule)} \\ &= \int p(f^*|\boldsymbol{f}, \boldsymbol{x}^*)p(\boldsymbol{f}|\boldsymbol{y})\mathrm{d}\boldsymbol{f} && \text{(Conditional independence)} \\ &\approx \int p(f^*|\boldsymbol{f}, \boldsymbol{x}^*)q(\boldsymbol{f})\mathrm{d}\boldsymbol{f} && \text{(Laplace)} \end{aligned}$$

where $q(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{m}, \boldsymbol{S})$ is the Laplace approximation for $p(\boldsymbol{f}|\boldsymbol{y})$

- $\boldsymbol{m}_i$ is the approximate posterior mean for $f(\boldsymbol{x}_i)$ and similar for the variance $\boldsymbol{S}_{ii}$

Gaussian processes for classification IV

- We have

$$p(f^*|\boldsymbol{y}, \boldsymbol{x}^*) = \int p(f^*|\boldsymbol{f}, \boldsymbol{x}^*) q(\boldsymbol{f}) \mathrm{d}\boldsymbol{f}$$

- $p(f^*|\boldsymbol{f}, \boldsymbol{x}^*)$ is just a conditional Gaussian density from $p(f^*, \boldsymbol{f}|\boldsymbol{x}^*)$ (see Murphy1 p. 84 again)

$$p(f^*|\boldsymbol{f}, \boldsymbol{x}^*) = \mathcal{N}(\, f^* \,|\, \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{f} \,,\, k - \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{k} \,)$$

- Therefore, we can use the equations for linear Gaussian models again (see Murphy1 Section 3.3) to derive

$$\begin{aligned} p(f^*|\boldsymbol{y}, \boldsymbol{x}^*) &= \int \mathcal{N}(\, f^* \,|\, \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{f} \,,\, k - \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{k} \,) \mathcal{N}(\boldsymbol{f}|\boldsymbol{m}, \boldsymbol{S}) \mathrm{d}\boldsymbol{f} \\ &= \mathcal{N}(\, f^* \,| \mu_{f^*} \,,\, \sigma^2_{f^*} \,) \end{aligned}$$

  where

$$\begin{aligned} \mu_{f^*} &= \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{m} \\ \sigma^2_{f^*} &= k - \boldsymbol{k}^T \boldsymbol{K}^{-1} (\boldsymbol{K} - \boldsymbol{S}) \boldsymbol{K}^{-1} \boldsymbol{k} \end{aligned}$$

Gaussian processes for classification V: Making predictions

- Step 1: Compute the Laplace approximation

$$p(\boldsymbol{f}|\boldsymbol{y}) \approx q(\mathbf{f}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{m}, \boldsymbol{S})$$

- Step 2: Posterior distribution for latent function f^* using the Laplace approximation

$$p(f^*|\boldsymbol{y}, \boldsymbol{x}^*) = \mathcal{N}(\, f^* \,|\mu_{f*}\,,\, \sigma^2_{f*})$$

- Step 3: Several options for computing the predictive distribution for classification labels

$$p(y^* = 1|\boldsymbol{y}, \boldsymbol{x}^*) \approx \int p(y^* = 1|f^*)p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}f^* = \int \sigma(f^*)p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}f^*$$

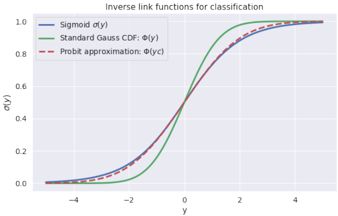
1. *Monte Carlo methods* (sampling)

$$p(y^* = 1|\boldsymbol{y}, \boldsymbol{x}^*) \approx \frac{1}{S}\sum_{i=1}^{S}\sigma\left(f^{(i)}\right) \qquad \text{for} \qquad f^{(i)} \sim \mathcal{N}(f|\mu_{f*}\,, \sigma^2_{f*})$$

2. *Probit approximation*

$$\sigma(f) \approx \Phi\left(f\sqrt{\frac{\pi}{8}}\right) \qquad \Rightarrow \qquad p(y^* = 1|\boldsymbol{y}, \boldsymbol{x}^*) \approx \Phi\left(\frac{\mu_{f*}}{\sqrt{\frac{8}{\pi} + \sigma^2_{f*}}}\right)$$

where Φ is the CDF of the standard normal



Gaussian processes for classification VI: Putting everything together

- Step 1: Construct Laplace approximation

$$p(\boldsymbol{f}|\boldsymbol{y}) \approx q(\mathbf{f}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{m}, \boldsymbol{S})$$

- Step 2: Posterior distribution for latent function f^*

$$p(f^*|\boldsymbol{y}, \boldsymbol{x}^*) = \mathcal{N}(f^*|\mu_{f*}, \sigma^2_{f*})$$

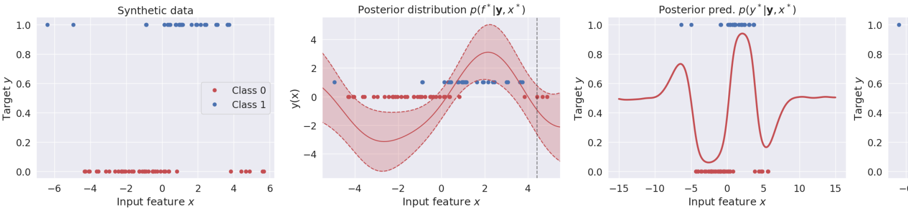
where

$$\mu_{f*} = \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{m}$$

$$\sigma^2_{f*} = k - \boldsymbol{k}^T \boldsymbol{K}^{-1}(\boldsymbol{K} - \boldsymbol{S})\boldsymbol{K}^{-1}\boldsymbol{k}$$

- Step 3: The predictive distribution for classification labels

$$p(y^* = 1|\boldsymbol{y}, \boldsymbol{x}^*) \approx \int p(y^* = 1|f^*)p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}d^* = \int \sigma(f^*)p(f^*|\boldsymbol{y}, \boldsymbol{x}^*)\mathrm{d}f^*$$



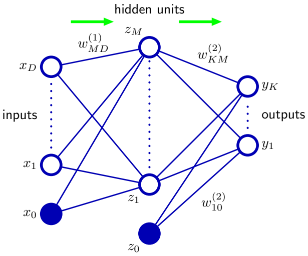
A bit about neural networks for probabilistic modelling

Neural Networks

- Sequence of linear (affine) and non-linear mappings

- Two-layer neural network (NN) with single output

$$\begin{aligned} \boldsymbol{z}_1 &= h_1(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b_1}) \\ \boldsymbol{z}_2 &= h_2(\boldsymbol{W}_2 \boldsymbol{z}_1 + \boldsymbol{b_2}) \\ f &= \boldsymbol{W}_3 \boldsymbol{z}_2 + \boldsymbol{b_3} \end{aligned}$$

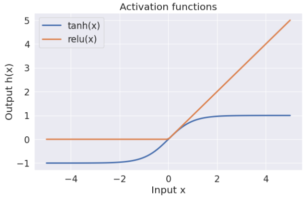


- From input to output (bias terms left out)

$$f(\boldsymbol{x}) = \boldsymbol{W}_3 h_2(\boldsymbol{W}_2 h_1(\boldsymbol{W}_1 \boldsymbol{x}))$$

- Our linear model with basis functions from week 2

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$$



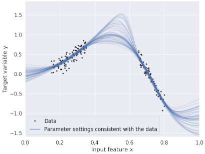
- Linear model with adaptive basis functions

$$\begin{aligned} z_2(\boldsymbol{x}) &= h_2(\boldsymbol{W}_2 h_1(\boldsymbol{W}_1 \boldsymbol{x})) = \boldsymbol{\Phi}_{\boldsymbol{W}}(\boldsymbol{x}) \\ f(\boldsymbol{x}) &= \boldsymbol{W}_3 \boldsymbol{z}_2 = \boldsymbol{W}_3 \boldsymbol{\Phi}_{\boldsymbol{W}}(\boldsymbol{x}) \end{aligned}$$

- NNs in probabilistic modelling

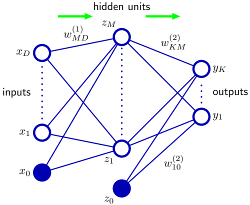
$$p(y_n|\boldsymbol{w}) = \mathcal{N}(y_n|f(\boldsymbol{x}|\boldsymbol{w}), \sigma^2) \qquad p(y_n|\boldsymbol{w}) = \text{Ber}(y_n|\sigma(f(\boldsymbol{x}|\boldsymbol{w})))$$

Bayesian Neural Networks



- Bayesian deep learning is a very active research area

- Why Bayesian neural networks?
  1. Ensemble methods often work well
  2. Uncertainty quantification
  3. Incorporating prior knowledge
  4. Less prone to overfitting
  5. Data efficiency
  6. Side step pathologies with maximum likelihood learning
  7. Active learning



- Posterior geometry of NNs are complicated!
  1. NNs are highly non-linear
  2. NNs have weight-space symmetries
  3. Often underdetermined by the data

- Bayesian inference in neural networks is generally a really *difficult problem*

$$\begin{aligned}\boldsymbol{z}_1 &= h_1(\boldsymbol{w}_1 \boldsymbol{x}) + \boldsymbol{b}_1\\ z_2 &= h_2(\boldsymbol{w}_2 \boldsymbol{z}_1) + \boldsymbol{b}_2\\ f &= \boldsymbol{w}_3 \boldsymbol{z}_2 + \boldsymbol{b}_3\end{aligned}$$

MAP estimators for probabilistic neural networks

- NNs for binary classification

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{n=1}^{N} \sigma(f_n)^{y_n} \left(1 - \sigma(f_n)\right)^{1-y_n}$$

$$f_n = \mathrm{NN}(\boldsymbol{x}_n|\boldsymbol{w})$$

- We can impose Gaussian priors on all the weights ($\boldsymbol{w}$ is vector containing all weights of the NN)

$$\boldsymbol{w} \sim \mathcal{N}(0, \alpha^{-1})$$

- The MAP (sometimes called to as *poor man's Bayes*) estimator is

$$\hat{\boldsymbol{w}}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{w}} \frac{p(\boldsymbol{y}|\boldsymbol{W})p(\boldsymbol{W})}{p(\boldsymbol{y})} = \arg\max_{\boldsymbol{W}} p(\boldsymbol{y}|\boldsymbol{W})p(\boldsymbol{W})$$

- We have

$$\ln p(\boldsymbol{y}|\boldsymbol{W})p(\boldsymbol{W}) = \sum_{n=1}^{N} \ln p(y_n|f(\boldsymbol{x}_n|\boldsymbol{w})) - \frac{\alpha}{2}\boldsymbol{w}^T\boldsymbol{w} + \text{const}$$

- For MAP estimation, Gaussian priors acts as ℓ_2-regularization

- Pros: easy and fast to implement, easy build complex models, Cons: no uncertainty and prone to overfitting

Questions: True or False

Spend 5 minutes on the DTU Learn quiz: "Lecture 6: Probabilistic neural networks."

Given a model with likelihood $p(\boldsymbol{y}|\boldsymbol{W})$ and supose we impose a flat prior on $\boldsymbol{w}$, i.e. $p(\boldsymbol{w}) \propto 1$, then ...

1. .... the maximum a posterior (MAP) solution is the same as the posterior mean. *True or false?*

2. ... the maximum likelihood solution and MAP (posterior mode) is the same. *True or false?*

3. ... the predictive distribution for MAP is the same as that for Bayesian inference? *True or False?*

For models with Gaussian priors

4 ... increasing α will cause MAP estimate of $\boldsymbol{w}$ to numerically larger. *True or False?*

5 ... increasing α will increase strength of regularization. *True or False?*