Machine Learning I
Week 8: Kernels

Martin Felder, Christian Osendorfer

Technische Universität München

14 January 2010

Recap: Bayesian Linear Regression

Construct model from M functions $\phi(\mathbf{x})$:

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}) \tag{1}$$

Assume the usual Gaussian (conjugate!) prior

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0 = 0, \mathbf{S}_0 = \alpha^{-1}\mathbf{I})$$

For N training samples, this yields a Gaussian posterior solution with

$$\mathbf{m}_N = \beta \mathbf{S}_N \mathbf{\Phi}^{\mathsf{T}} \mathbf{T} \tag{2}$$

$$\mathbf{S}_N^{-1} = \alpha \mathbf{I} + \beta \mathbf{\Phi}^{\mathsf{T}} \mathbf{\Phi} \tag{3}$$

Re-formulating the MAP solution

But the mode of a Gaussian coincides with its maximum, therefore $\mathbf{w}_{\text{MAP}} = \mathbf{m}_N$.

Let's see what happens if we re-substitute the MAP solution into the defining equation:

$$y(\mathbf{x}, \mathbf{w}_{\text{MAP}}) = \mathbf{w}_{\text{MAP}}^{\mathsf{T}} \phi(\mathbf{x}) = \phi(\mathbf{x})^{\mathsf{T}} \mathbf{m}_N = \\ = \beta \phi(\mathbf{x})^{\mathsf{T}} \mathbf{S}_N \mathbf{\Phi}^{\mathsf{T}} \mathbf{T}$$

In other words, this is just a linear combination of our training points t_n:

$$y(\mathbf{x}, \mathbf{m}_N) = \sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) t_n$$

where

$$k(\mathbf{x}, \mathbf{x}_n) = \beta \phi(\mathbf{x})^{\mathsf{T}} \mathbf{S}_N \phi(\mathbf{x}_n)$$

is called (equivalent) kernel.

Equivalent Kernel

What is the interpretation of the kernel?

$$\text{cov}[y(\mathbf{x}), y(\mathbf{x}')] = \text{cov}[\boldsymbol{\phi}(\mathbf{x})^{\mathsf{T}}\mathbf{w}, \mathbf{w}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}')] = \tag{4}$$

$$= \boldsymbol{\phi}(\mathbf{x})^{\mathsf{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}) = \beta^{-1}k(\mathbf{x}, \mathbf{x}') \tag{5}$$

It can also be shown that

$$\sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) = 1,$$

although $k(\mathbf{x}, \mathbf{x}_n)$ can be negative.

Furthermore, since $\mathbf{S}_N$ is a covariance matrix, $\mathbf{S}_N^{\frac{1}{2}}$ must exist, and we can write

$$k(\mathbf{x}, \mathbf{x}') = \beta\boldsymbol{\phi}(\mathbf{x})^{\mathsf{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}') = \sqrt{\beta}\boldsymbol{\phi}(\mathbf{x})^{\mathsf{T}}\mathbf{S}_N^{\frac{\mathsf{T}}{2}}\mathbf{S}_N^{\frac{1}{2}}\boldsymbol{\phi}(\mathbf{x}')\sqrt{\beta} = \boldsymbol{\psi}(\mathbf{x})^{\mathsf{T}}\boldsymbol{\psi}(\mathbf{x}')$$

Dual Representation

Starting from a regularized quadratic error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n) - t_n\right)^2 + \frac{\lambda}{2}\mathbf{w}^{\mathsf{T}}\mathbf{w}, \tag{6}$$

we find the MAP solution by setting the gradient of $E(\mathbf{w})$ to zero. The solution for $\mathbf{w}$ can be written implicitly as

$$\mathbf{w} = -\frac{\lambda}{2}\sum_{n=1}^{N}\left(\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n) - t_n\right)\phi(\mathbf{x}) = \sum_{n=1}^{N} a_n\phi(\mathbf{x}_n) = \mathbf{\Phi}^{\mathsf{T}}\mathbf{a}. \tag{7}$$

When substituting this into Eqn. 6 and setting $\frac{\partial E(\mathbf{a})}{\partial \mathbf{a}}$ to zero, with the definition of the Gram matrix $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^{\mathsf{T}}$ we get

$$\mathbf{a} = (\mathbf{K} + \lambda \mathbf{I}_N)^{-1}\mathbf{T} \tag{8}$$

Dual Representation

Re-substituing this back into Eqn. 1 yields

$$y(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}) = \mathbf{a}^{\mathsf{T}}\mathbf{\Phi}\phi(\mathbf{x}) = \mathbf{k}(\mathbf{x})^{\mathsf{T}}(\mathbf{K} + \lambda\mathbf{I}_N)^{-1}\mathbf{T} \qquad (9)$$

where the vector $\mathbf{k}(\mathbf{x})$ has elements

$$k_n(\mathbf{x}) = k(\mathbf{x}_n, \mathbf{x}).$$

Expressing the solution in terms of $\mathbf{a}$ instead of $\mathbf{w}$ is called the dual formulation of the problem.

Properties:

- The whole solution is expressed in terms of kernel functions.
- The original formulation can be recovered by Eqn. 7.
- We have to invert an $N \times N$ matrix now, instead of an $M \times M$ matrix. Usually $N \gg M$ – but the kernels $k(\mathbf{x}, \mathbf{x}')$ allow us to use very large (even infinite) Ms implicitly!

A simple kernel

Assume $\mathbf{x} \in \mathbb{R}^2$ and we choose a kernel $k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T\mathbf{z})^2$

$\longrightarrow$ as is easy to check, this corresponds to a mapping function

$$\phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix} \in \mathbb{R}^3$$

Note: We are already saving a lot of multiplications here!

Neither this mapping nor the feature space dimension are uniquely defined. We might as well use

$$\phi(\mathbf{x}) = \frac{1}{\sqrt{2}} \begin{pmatrix} x_1^2 - x_2^2 \\ 2x_1x_2 \\ x_1^2 + x_2^2 \end{pmatrix} \quad \text{or even} \quad \phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ x_1x_2 \\ x_1x_2 \\ x_2^2 \end{pmatrix} \in \mathbb{R}^4$$

A simple kernel

Assume $\mathbf{x} \in \mathbb{R}^2$ and we choose a kernel $k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z})^2$
$\longrightarrow$ as is easy to check, this corresponds to a mapping function

$$\phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix} \in \mathbb{R}^3$$

Note: We are already saving a lot of multiplications here!

Neither this mapping nor the feature space dimension are uniquely defined. We might as well use

$$\phi(\mathbf{x}) = \frac{1}{\sqrt{2}} \begin{pmatrix} x_1^2 - x_2^2 \\ 2 x_1 x_2 \\ x_1^2 + x_2^2 \end{pmatrix} \quad \text{or even} \quad \phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_2 \\ x_2^2 \end{pmatrix} \in \mathbb{R}^4$$

What is a valid kernel?

- Is every $k(\mathbf{x}, \mathbf{z})$ feasible?
- Well, no – the feature space must be a Hilbert space.
- Mercer's condition ensures this:

  "A mapping $k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^T \phi(\mathbf{z})$ exists, iff for any $g(\mathbf{x})$ such that $\int g(\mathbf{x})^2 d\mathbf{x}$ is finite, then $\int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0$."

  $\longrightarrow$ Very inconvenient to prove!
- What happens if Mercer's condition is not fulfilled?
  - Often it may still work well – just the explanations given here won't hold anymore.
  - However it can happen that the optimization fails completely.

What is a valid kernel?

- Is every $k(\mathbf{x}, \mathbf{z})$ feasible?
- Well, no – the feature space must be a Hilbert space.
- Mercer's condition ensures this:

  "A mapping $k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^T \phi(\mathbf{z})$ exists, iff for any $g(\mathbf{x})$ such that $\int g(\mathbf{x})^2 d\mathbf{x}$ is finite, then $\int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0$."

  $\longrightarrow$ Very inconvenient to prove!
- What happens if Mercer's condition is not fulfilled?
  - Often it may still work well – just the explanations given here won't hold anymore.
  - However it can happen that the optimization fails completely.

What is a valid kernel?

- Is every $k(\mathbf{x}, \mathbf{z})$ feasible?
- Well, no – the feature space must be a Hilbert space.
- Mercer's condition ensures this:

  "A mapping $k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^T \phi(\mathbf{z})$ *exists, iff for any* $g(\mathbf{x})$ *such that* $\int g(\mathbf{x})^2 d\mathbf{x}$ *is finite, then* $\int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0$*."*

  $\longrightarrow$ Very inconvenient to prove!

- What happens if Mercer's condition is not fulfilled?
  - Often it may still work well – just the explanations given here won't hold anymore.
  - However it can happen that the optimization fails completely.

What is a valid kernel?

- Is every $k(\mathbf{x}, \mathbf{z})$ feasible?
- Well, no – the feature space must be a Hilbert space.
- Mercer's condition ensures this:

  "A mapping $k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^T \phi(\mathbf{z})$ *exists, iff for any* $g(\mathbf{x})$ *such that* $\int g(\mathbf{x})^2 d\mathbf{x}$ *is finite, then* $\int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0$*."*

  $\longrightarrow$ Very inconvenient to prove!
- What happens if Mercer's condition is not fulfilled?
  - Often it may still work well – just the explanations given here won't hold anymore.
  - However it can happen that the optimization fails completely.

What is a valid kernel?

- Is every $k(\mathbf{x}, \mathbf{z})$ feasible?
- Well, no – the feature space must be a Hilbert space.
- Mercer's condition ensures this:

  "A mapping $k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^T \phi(\mathbf{z})$ *exists, iff for any* $g(\mathbf{x})$ *such that* $\int g(\mathbf{x})^2 d\mathbf{x}$ *is finite, then* $\int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0$*."*

  $\longrightarrow$ Very inconvenient to prove!
- What happens if Mercer's condition is not fulfilled?
  - Often it may still work well – just the explanations given here won't hold anymore.
  - However it can happen that the optimization fails completely.

$\longrightarrow$ Often more convenient to start from a proven kernel and construct a new one using *kernel manipulation rules*.

Constructing Kernels

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following kernels are also valid:

- $k(\mathbf{x}, \mathbf{x}') = ck_1(\mathbf{x}, \mathbf{x}')$
- $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$
- $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$
- $k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}')$
- $k(\mathbf{x}, \mathbf{x}') = q(k_1(\mathbf{x}, \mathbf{x}'))$ and $k(\mathbf{x}, \mathbf{x}') = \exp(k_1(\mathbf{x}, \mathbf{x}'))$
- $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}\mathbf{A}\mathbf{x}'$
- $k(\mathbf{x}, \mathbf{x}') = k_3(\phi(\mathbf{x}), \phi(\mathbf{x}'))$

$f(\cdot)$ any function, $q(\cdot)$ polynomial with non-negative coefficients, $\phi(\cdot)$ function in $\mathbb{R}^M$, $k_3(\cdot, \cdot)$ valid kernel in $\mathbb{R}^M$, $\mathbf{A}$ symmetric positive definite matrix

Example Kernels

Define a range function as

$$r(\mathbf{x}) = \begin{cases} 1 & \text{if } \|\mathbf{x}\| \leq 1 \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \Delta\mathbf{x} \equiv \mathbf{x} - \mathbf{x}'.$$

Kernels sometimes found in the literature:

- Uniform: $k(\mathbf{x}, \mathbf{x}') = \frac{1}{2} r(\Delta\mathbf{x})$
- Triangle: $k(\mathbf{x}, \mathbf{x}') = (1 - \|\Delta\mathbf{x}\|) r(\Delta\mathbf{x})$
- Epanechnikov: $k(\mathbf{x}, \mathbf{x}') = \frac{3}{4}(1 - \|\Delta\mathbf{x}\|^2) r(\Delta\mathbf{x})$
- Quartic: $k(\mathbf{x}, \mathbf{x}') = \frac{15}{16}(1 - \|\Delta\mathbf{x}\|^2)^2 r(\Delta\mathbf{x})$
- Triweight: $k(\mathbf{x}, \mathbf{x}') = \frac{35}{32}(1 - \|\Delta\mathbf{x}\|^2)^3 r(\Delta\mathbf{x})$
- Gaussian: $k(\mathbf{x}, \mathbf{x}') = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\|\Delta\mathbf{x}\|^2\right)$
- Cosine: $k(\mathbf{x}, \mathbf{x}') = \frac{\pi}{4} \cos\left(\frac{\pi}{2}\Delta\mathbf{x}\right) r(\Delta\mathbf{x})$

Non-Vectorial Kernels

Note that Mercer's condition demands for a Hilbert space, not necessarily a vector space!

$\Longrightarrow$ Can define custom kernels for a broad class of data, like strings, graphs, text documents, sets, ...

Example: Given a fixed set $\mathcal{S}$, define a Hilbert space consisting of all possible subsets $\mathcal{A}$ of this set. A simple definition of a kernel in this space would be

$$k(\mathcal{A}_1, \mathcal{A}_2) = 2^{|\mathcal{A}_1 \cap \mathcal{A}_2|}.$$

Kernel Algorithms

Many algorithms can operate on kernels instead of non-transformed input vectors or basis functions, greatly increasing their modelling capabilities while keeping complexity in check. These include

- Support Vector Machines (SVMs; $\rightarrow$ next week!)
- Fisher's linear discriminant analysis
- principal components analysis (PCA)
- canonical correlation analysis
- ridge regression
- spectral clustering
- etc, etc.

Kernel application example: Defining implicit surfaces

Let $\theta(\mathbf{x})$ be a continuous scalar function with $\mathbf{x} \in \mathbb{R}^3$. An implicit surface is then defined by the set of points that satisfies

$$\theta(\mathbf{x}) = 0.$$

Application: Use kernels to reconstruct the surface of laser-scanned 3D objects. This method can easily deal with millions of 3D data points, and is resistant to outliers and noise.
The kernel used is of course specially designed for the task, taking surface normals into account, and uses regularization.