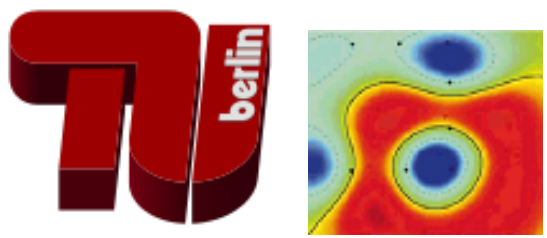
Lecture 10: Learning Theory and Kernels

Machine Learning 1



TU Berlin – WiSe 2016/2017

Occam's Razor and Prediction Strength

Occam's Razor in a ML context (2nd interpretation)

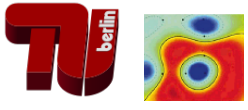
> *"Given two model with the same training-set error the simpler one should be preferred because it is likely to have lower generalization error."*

Question: What does "simple" mean?

Falsifiability/prediction strength (S. Hawking, after K. Popper)

> *"[a good model] must accurately describe a large class of observations on the basis of a model that contains only a few arbitrary elements, and it must make definite predictions about the results of future observations."*

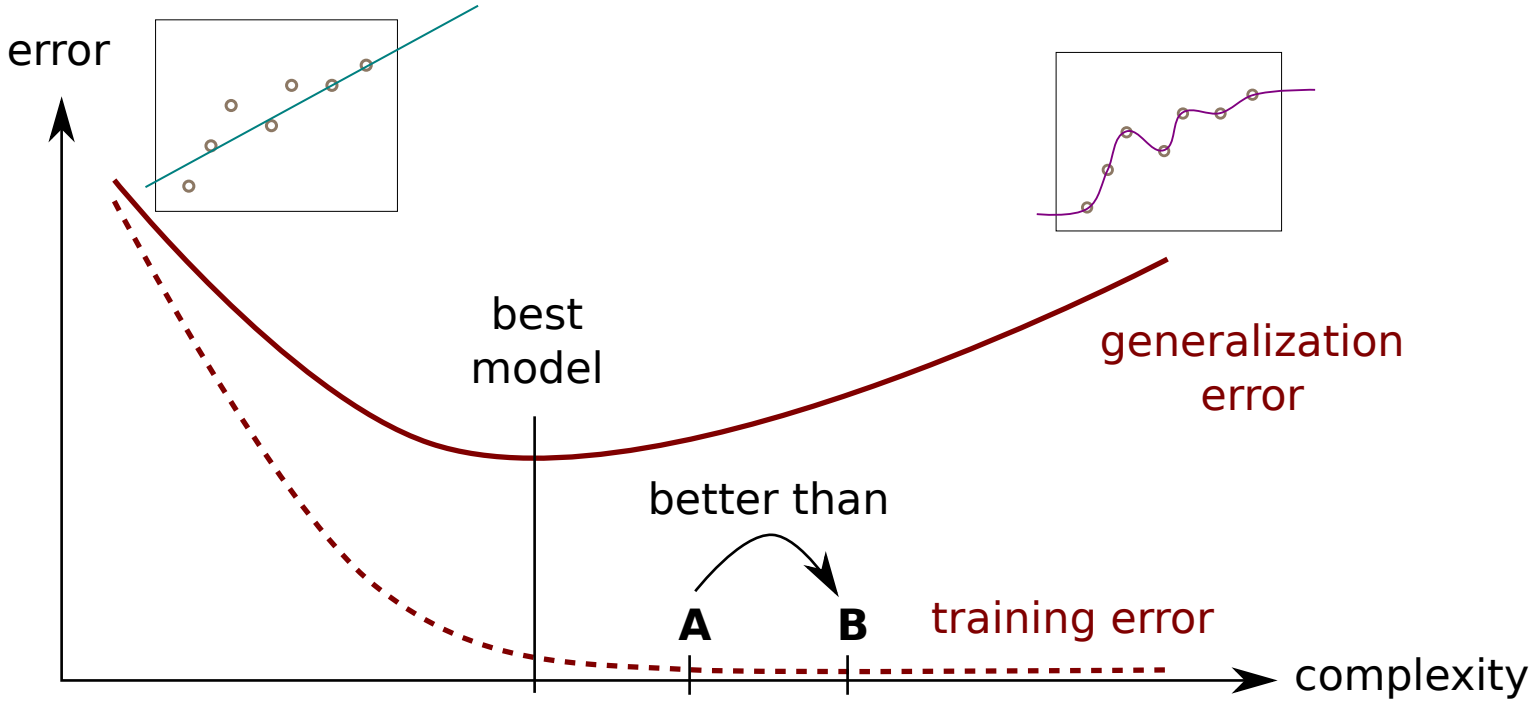
means: The model with lowest generalization error is preferable.



From Occam's Razor to Prediction Strength

"Given two model with the same training-set error the simpler one should be preferred because it is likely to have lower generalization error."

"[a good model] must accurately describe a large class of observations on the basis of a model that contains only a few arbitrary elements, and it must make definite predictions about the results of future observations."



Learning Theory

Learn f from examples

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \in \mathbb{R}^n \times \mathbb{R}^m \text{ or } \{\pm 1\}, \quad \text{generated from } P(\mathbf{x}, y),$$

such that expected number of errors on test set (drawn from $P(\mathbf{x}, y)$),

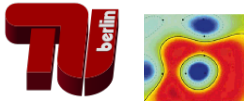
$$R[f] = \int \frac{1}{2} |f(\mathbf{x}) - y)|^2 \, dP(\mathbf{x}, y),$$

is minimal *(Risk Minimization (RM))*.

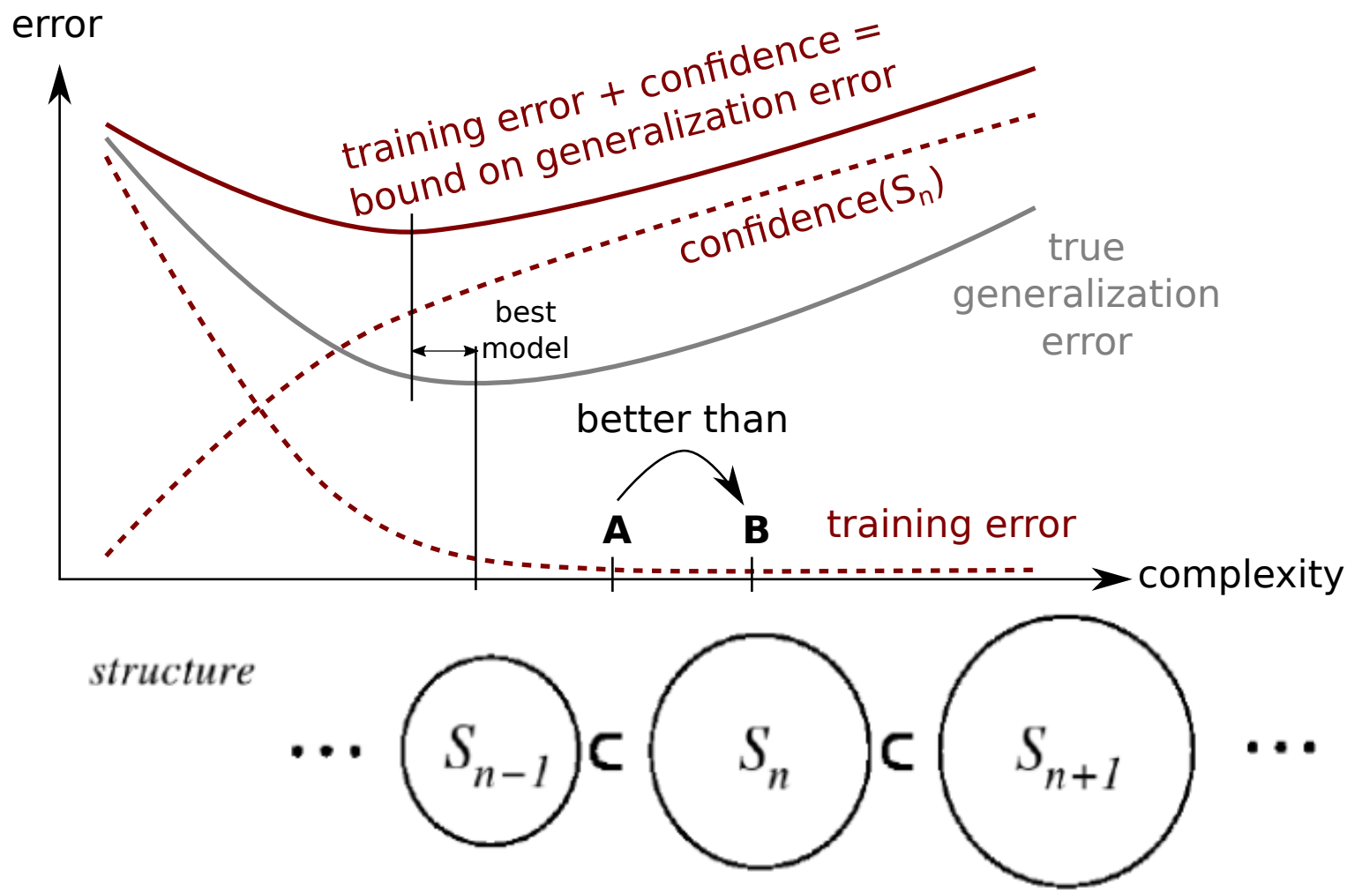
Problem: P is unknown. $\longrightarrow$ need an *induction principle*.

Empirical risk minimization (ERM): replace the average over $P(\mathbf{x}, y)$ by an average over the training sample, i.e. minimize the training error

$$R_{emp}[f] = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} |f(\mathbf{x}_i) - y_i|^2$$



Learning Theory: Structural Risk Minimization



Are there structures of functions $(S_n)_n$ that have such confidence score, and can be computed?

VC-Dimension

The VC-dimension is the *maximum* number of data points that the function class can *always* shatter (i.e. classify in *any* possible ways).

Formally:

- Shatter coefficient

$$S_{\mathcal{F}}(x_1, \ldots, x_n) = |\{f(x_1), \ldots, f(x_n)\}; f \in \mathcal{F}|$$

- Growth function

$$S_{\mathcal{F}}(n) = \max_{x_1,\ldots,x_n} S_{\mathcal{F}}(x_1, \ldots, x_n)$$

- VC-dimension

$$\mathrm{VC}_{\mathcal{F}} = \max\{n : S_{\mathcal{F}}(n) = 2^n\}$$

VC-Dimension (cont.)

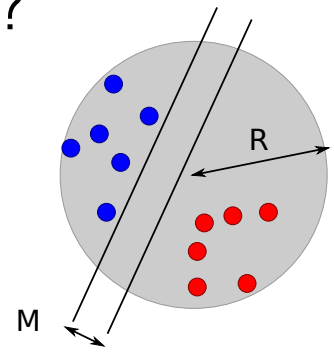
Intuitive definition:
The VC-dimension is the *maximum* number of data points that the function class can *always* shatter (i.e. classify in *any* possible ways).

Examples:

- VC-dimension of $f : \mathbb{R} \to \{-1, 1\},\ f(x) = \text{sign}(\sin(\alpha x))$?
  Answer: ∞

- VC-dimension of $f : \mathbb{R}^d \to \{-1, 1\},\ f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^\top \boldsymbol{x} + b)$?
  Answer: $d + 1$ (also related to the number of parameters).

- VC-dimension of $f : \mathbb{R}^d \to \{-1, 1\},\ f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^\top \boldsymbol{x} + b)$, where all data points are in a minimum enclosing sphere of radius R, and classified with some margin M ?
  Answer: $\min\left\{d + 1\,,\, 4\frac{R^2}{M^2} + 1\right\}$.
  $\Rightarrow$ Does not only depend on input dimension.
  $\Rightarrow$ Large margin lowers model complexity.



Bound on Generalization Error

Generalization bound [Vapnik]:

Let h denote the VC-dimension of $\mathcal{F}$. The true risk $R[f]$ (with $f \in \mathcal{F}$) is upper-bounded as:

$$R[f] \leq R_{\text{emp}}[f] + \sqrt{\frac{h\left(\log \frac{2N}{h} + 1\right) - \log(\eta/4)}{N}}$$

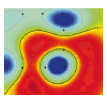
with probability $1 - \eta$.

Interpretation:

- Error increases with the VC-dimension h (for h small enough).
- Error decreases with the number of samples N.

Remark:

- If the VC-dimension is infinite, the empirical error does not converge to $R[f]$ even for $N \to \infty$ (see $\sin(\alpha x)$ example).



From Linear to Nonlinear Models

Linear models of type $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$ are easy to regularize (e.g. dimensionality reduction, large-margin). However the class of linear models is too restrictive (cannot solve nonlinear problems).

Approach 1: Consider an arbitrary nonlinear function $f(\boldsymbol{x})$, (e.g. a quadratic model $f(\boldsymbol{x}) = \boldsymbol{x}^\top A \boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c$), and learn the parameters of this model from the data.

Approach 2: Apply a *fixed* nonlinear feature map $\Phi : \mathbb{R}^d \to \mathbb{R}^h$ to the data, and learn a *linear* model in the image domain:

$$f(\boldsymbol{x}) = \boldsymbol{\beta}^\top \Phi(\boldsymbol{x}) + b$$

Linear Model in Feature Space

Procedure:

1. Consider a *fixed* feature map:

$$\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^h$$
$$\boldsymbol{x} \mapsto \Phi(\boldsymbol{x})$$
$$\text{where} \quad d \ll h \quad \text{(typically)}$$

2. *Learn* $\boldsymbol{\beta} \in \mathbb{R}^h$ and $b \in \mathbb{R}$ such that the function

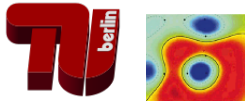
$$f(\boldsymbol{x}) = \boldsymbol{\beta}^\top \Phi(\boldsymbol{x}) + b$$

correctly predicts the data.

Example: *Feature map that computes all 2nd order monomials:*

$$(x_1, x_2) \mapsto (x_1^2, x_1 x_2, x_1 x_3, \ldots, x_d^2).$$

The resulting feature space has $d \cdot (d+1)/2$ *dimensions* ($\gg d$).

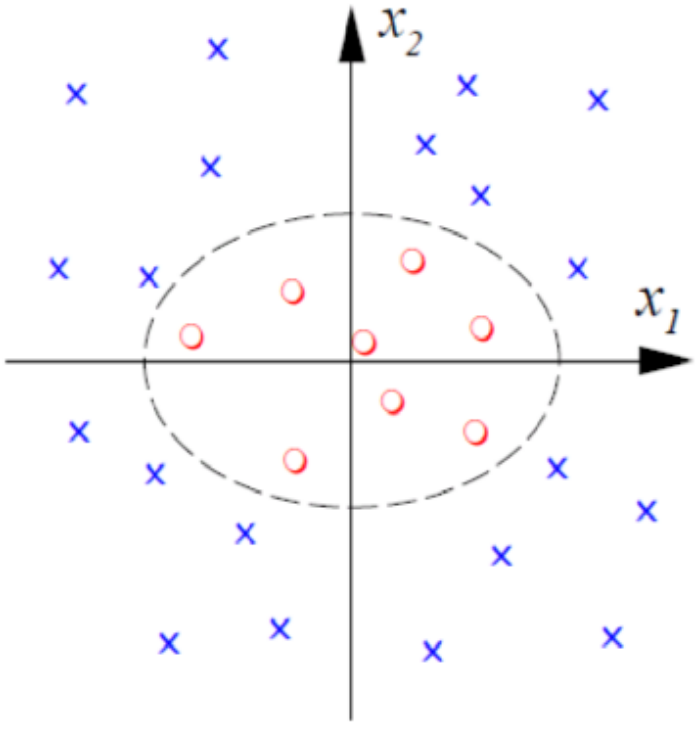


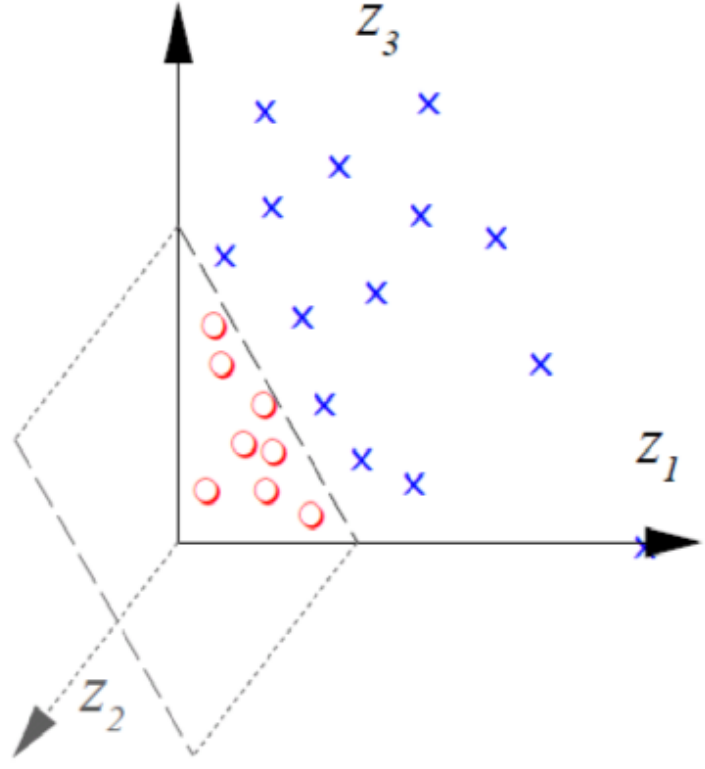
Linear Model in Feature Space

Example: all second order monomials

$$\Phi : \mathbf{R}^2 \rightarrow \mathbf{R}^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$



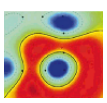


Linear Model in Feature Space

Remarks:

- Classical statistics: **harder** as the data become high-dimensional.
- VC-dimension viewpoint: **not harder** as we can control complexity via other means (e.g. by enforcing a large margin in the feature space).

$\Rightarrow$ **complexity matters, not dimensionality.**



From Feature Spaces to Kernels

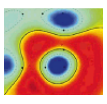
The kernel trick:

(cf. Boser, Guyon & Vapnik 1992)

$$\begin{aligned}(\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})) &= (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)(y_1^2, \sqrt{2}\, y_1 y_2, y_2^2)^\top \\ &= (\mathbf{x} \cdot \mathbf{y})^2 \\ &= : k(\mathbf{x}, \mathbf{y})\end{aligned}$$

- Scalar product in (**high dimensional**) feature space can be computed in $\mathbf{R}^2$!
- works only for Mercer Kernels $k(\mathbf{x}, \mathbf{y})$

Note: Next week, it will be shown that many algorithms can be made nonlinear (e.g. SVMs, ridge regression, PCA) and can be directly expressed in terms of kernel computations (and therefore, bypass the mapping to the high-dimensional feature space).



Kernology

[Mercer] If k is a continuous kernel of a positive integral operator on $L_2(\mathcal{D})$ (where $\mathcal{D}$ is some compact space),

$$\int f(\mathbf{x})k(\mathbf{x},\mathbf{y})f(\mathbf{y})\,d\mathbf{x}\,d\mathbf{y} \geq 0, \quad \text{for} \quad f \neq 0$$

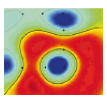
it can be expanded as

$$k(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N_F} \lambda_i \psi_i(\mathbf{x})\psi_i(\mathbf{y})$$

with $\lambda_i > 0$, and $N_F \in \mathbf{N}$ or $N_F = \infty$. In that case

$$\Phi(\mathbf{x}) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(\mathbf{x}) \\ \sqrt{\lambda_2}\psi_2(\mathbf{x}) \\ \vdots \end{pmatrix}$$

satisfies $(\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})) = k(\mathbf{x},\mathbf{y})$.

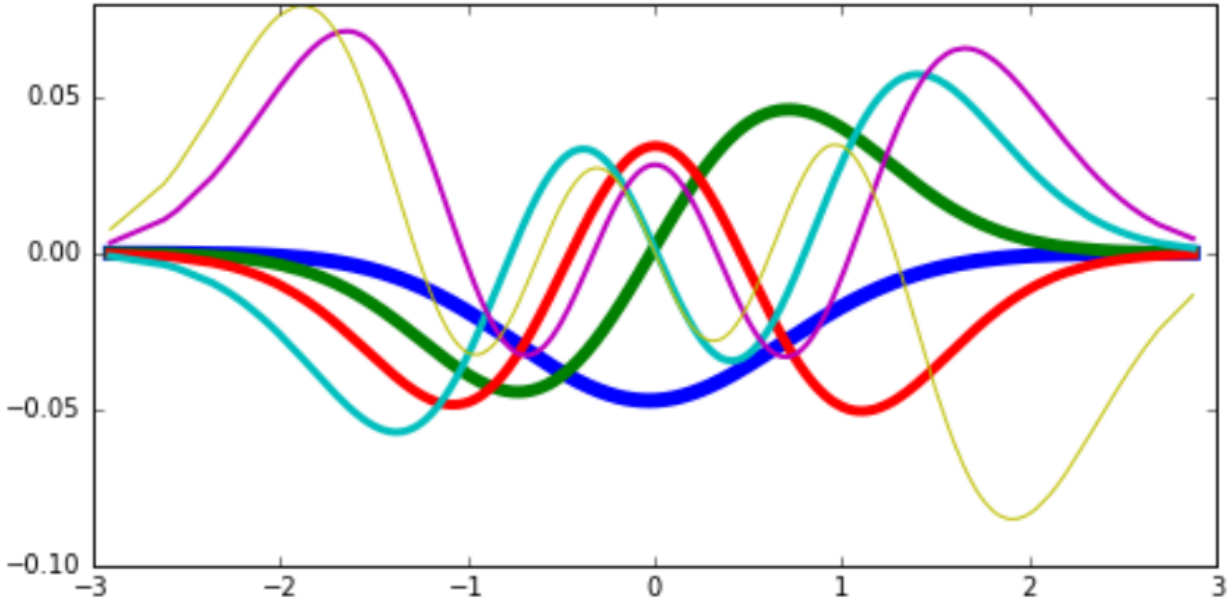


Kernology (cont.)

Computing $\psi_1(\boldsymbol{x}), \psi_2(\boldsymbol{x}), \ldots$ in practice.

```
# generate data and build kernel
X = numpy.sort(numpy.random.normal(0,1,[1000,1]),axis=0)
K = numpy.exp(-3*scipy.spatial.distance.cdist(X,X,'sqeuclidean'))

# compute and plot eigenbasis
U,W,V = numpy.linalg.svd(K,full_matrices=False)
for i in range(6): matplotlib.pyplot.plot(X[:,0],U[:,i],lw=6-i)
```



Kernology (cont.)

Examples of common kernels:

$$\begin{aligned}
\text{Polynomial} \quad & k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y}+c)^d \\
\sout{\text{Sigmoid} \quad k(\mathbf{x},\mathbf{y}) = \tanh(\kappa(\mathbf{x}\cdot\mathbf{y})+\Theta)} \\
\text{RBF} \quad & k(\mathbf{x},\mathbf{y}) = \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2/(2\,\sigma^2)\right) \\
\text{inverse multiquadric} \quad & k(\mathbf{x},\mathbf{y}) = \frac{1}{\sqrt{\|\mathbf{x}-\mathbf{y}\|^2+c^2}}
\end{aligned}$$

Note: kernels correspond to regularization operators (a la Tichonov) with regularization properties that can be conveniently expressed in Fourier space, e.g. Gaussian kernel corresponds to general smoothness assumption (Smola et al 98)!

Kernology (cont.)

In practice, to show that $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a (positive semi-definite) kernel, we can either [approach 1] show that:

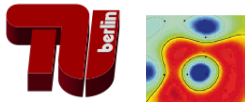
$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$$

for all collections of data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X}$ and coefficients $c_1, \ldots, c_N \in \mathbb{R}$.

or [approach 2] find a feature map $\Phi : \mathcal{X} \to \mathbb{R}^h$ such that

$$\Phi(\boldsymbol{x})^\top \Phi(\boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}')$$

for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$.



Final remarks

Cross-validation vs. VC dimension

- *Last week:* Since it was unclear how to measure complexity, we tried to directly predict the generalization error via cross-validation, and choose parameters to minimize this error.
- *This week:* Model complexity *can* be reliably measured (VC-dimension). It allows to do model selection (e.g. prioritize large-margin models) without having to consume data.

Nonlinear representation and kernels

- Linearly unsolvable problems can become solvable in high-dimensional feature spaces.
- High dimensionality is not a big problem, because other techniques (e.g. large margin) can be used to control capacity.