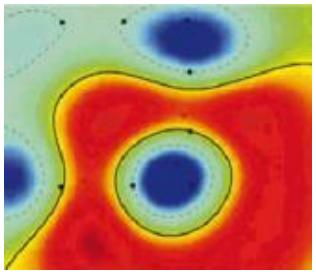
Kernel Methods

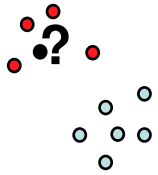
Introduction to SVMs, KPCA



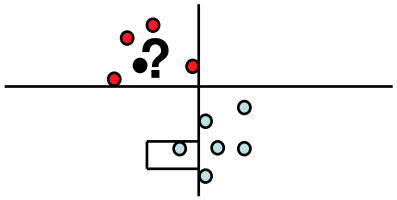
Lecture by Klaus-Robert Müller, TUB 2016

Machine Learning: some algorithms

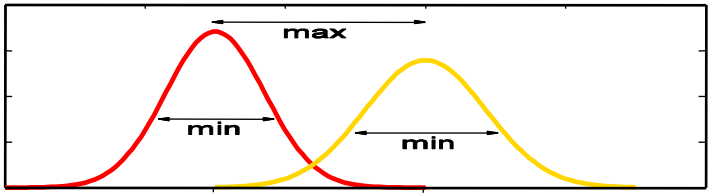
- k-nearest neighbor algorithm



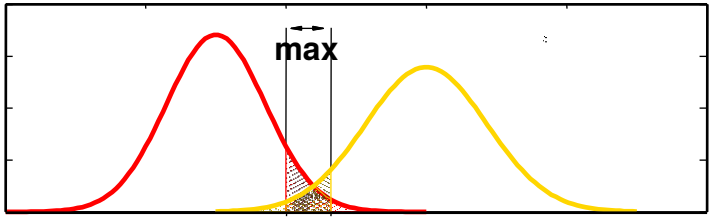
- decision tree



- Fisher discriminant



- Support Vector machine, LPM



Basic ideas in learning theory

Three scenarios: regression, classification & density estimation.
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

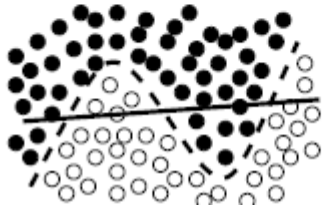
$$R_{emp}[f] = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} |f(\mathbf{x}_i) - y_i|^2$$

Basic ideas in learning theory II

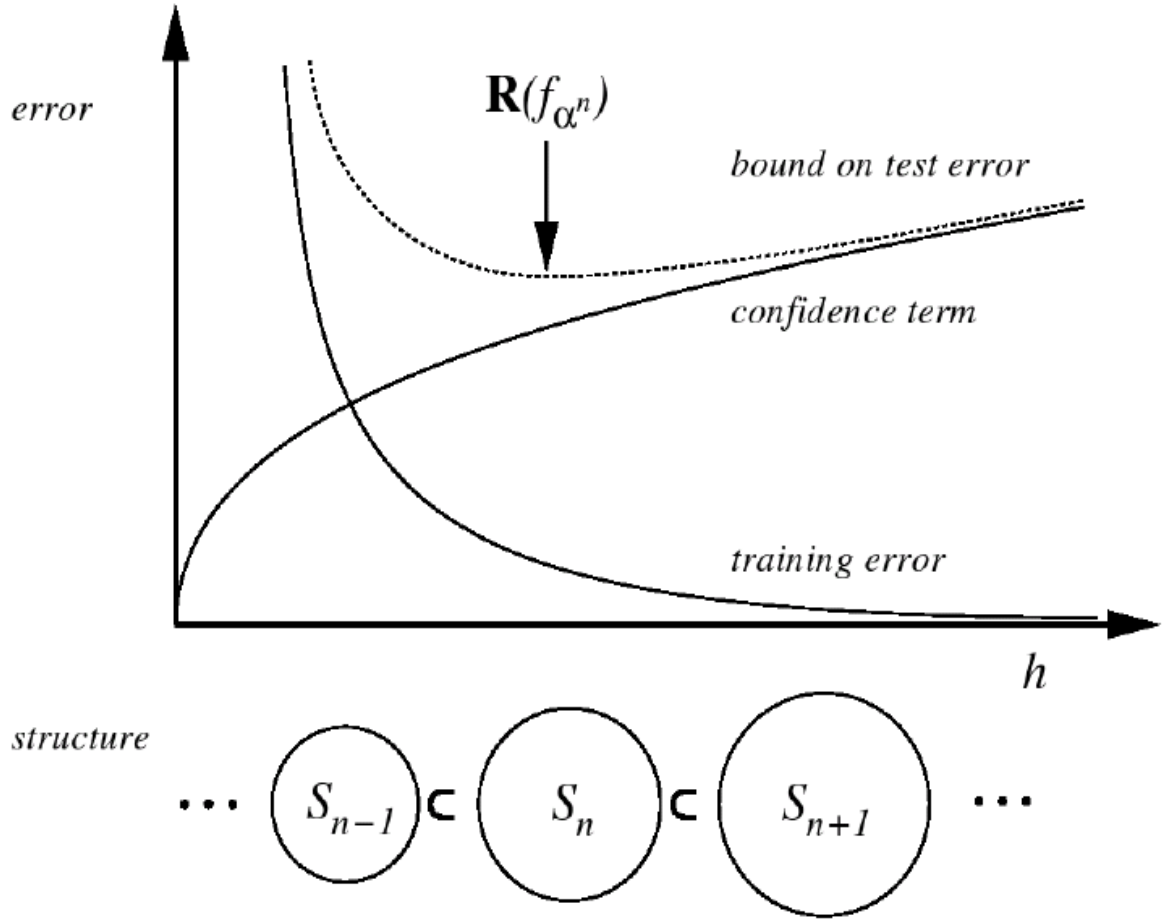
- Law of large numbers: $R_{emp}[f] \to R[f]$ as $N \to \infty$.
  "consistency" of ERM: for $N \to \infty$, ERM should lead to the same result as RM?
- No: *uniform* convergence needed (Vapnik) → VC theory.
  Thm. [classification] (Vapnik 95): with a probability of at least $1 - \eta$,

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

- Structural risk minimization (SRM): introduce structure on set of functions $\{f_\alpha\}$ & minimize RHS to get low risk! (Vapnik 95)
- d is VC dimension, measuring complexity of function class



Structural Risk Minimization: the picture



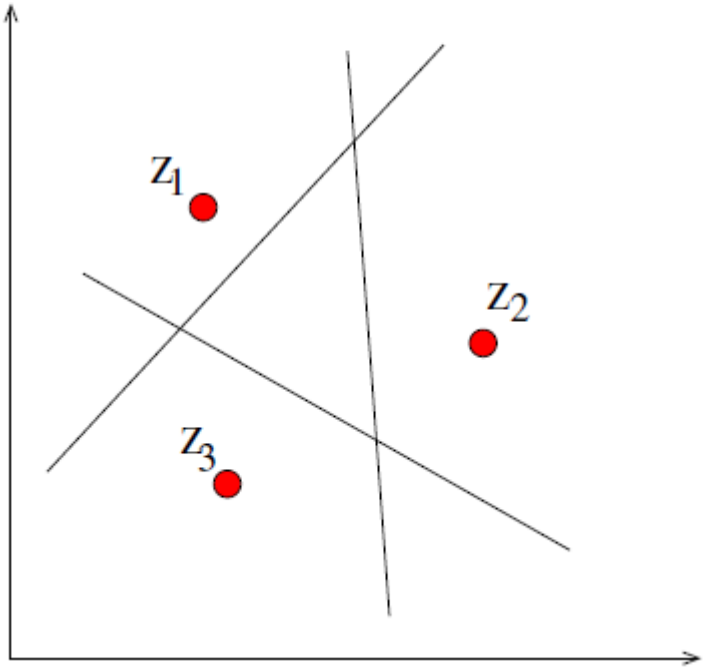
Learning f requires small training error *and* small complexity of the set $\{f_\alpha\}$.

VC Dimensions: an examples

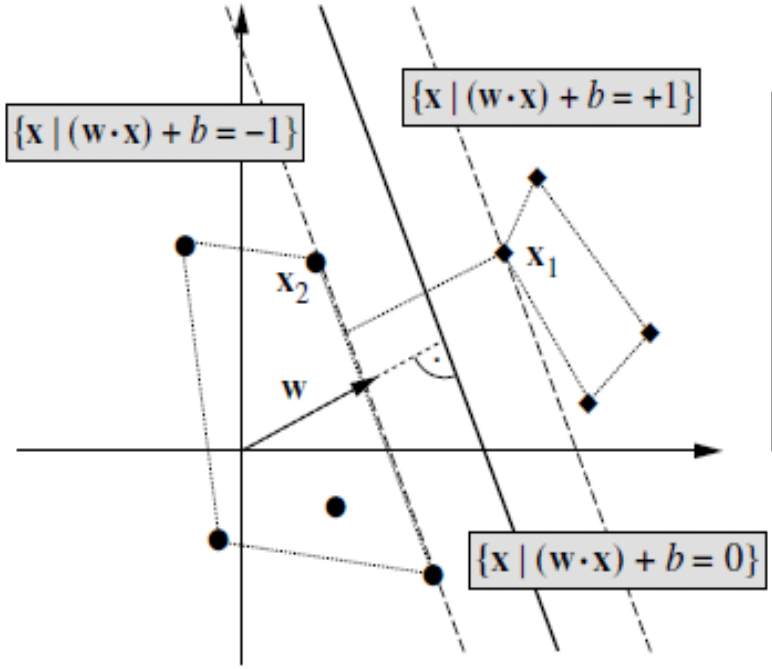
Half-spaces in $\mathbf{R}^2$:

$$f(x, y) = \text{sgn}(a + bx + cy), \quad \text{with parameters } a, b, c \in \mathbf{R}$$

- Clearly, we can shatter three non-collinear points.
- But we can never shatter four points.
- Hence the VC dimension is $d = 3$
- in n dimensions: VC dimension is $d = n + 1$



Linear Hyperplane Classifier



Note:

$$(\mathbf{w}\cdot\mathbf{x}_1) + b = +1$$
$$(\mathbf{w}\cdot\mathbf{x}_2) + b = -1$$
$$\Rightarrow \quad (\mathbf{w}\cdot(\mathbf{x}_1-\mathbf{x}_2)) = 2$$
$$\Rightarrow \left(\frac{\mathbf{w}}{\|\mathbf{w}\|}\cdot(\mathbf{x}_1-\mathbf{x}_2)\right) = \frac{2}{\|\mathbf{w}\|}$$

- hyperplane $y = \mathrm{sgn}\,(\mathbf{w}\cdot\mathbf{x}+b)$ in canonical form if $\min_{\mathbf{x}_i\in X} |\,(\mathbf{w}\cdot\mathbf{x}_i)+b| = 1.$, i.e. scaling freedom removed.
- larger margin $\sim 1/\|\mathbf{w}\|$ is giving better generalization $\rightarrow$ LMC!

VC Theory applied to hyperplane classifiers

- **Theorem (Vapnik 95)**: For hyperplanes in canonical form VC–dimension satisfying

$$d \leq \min\{[R^2 \|\mathbf{w}\|^2] + 1, n + 1\}.$$

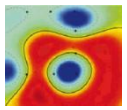
Here, R is the radius of the smallest sphere containing data.
Use d in SRM bound

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

- maximal margin = minimum $\|\mathbf{w}\|^2 \rightarrow$ good generalization, i.e. low risk, i.e. optimize

$$\min \|\mathbf{w}\|^2$$

- independent of the dimensionality of the space!



Feature Spaces & curse of dimensionality

The Support Vector (SV) approach: *preprocess* the data with

$$\begin{aligned} \Phi : \mathbf{R}^N &\rightarrow F \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}) \\ \text{where } N &\ll \dim(F). \end{aligned}$$

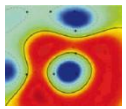
to get data $(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_N), y_N) \in F \times \mathbf{R}^M$ or $\{\pm 1\}$.

Learn $\tilde{f}$ to construct $f = \tilde{f} \circ \Phi$

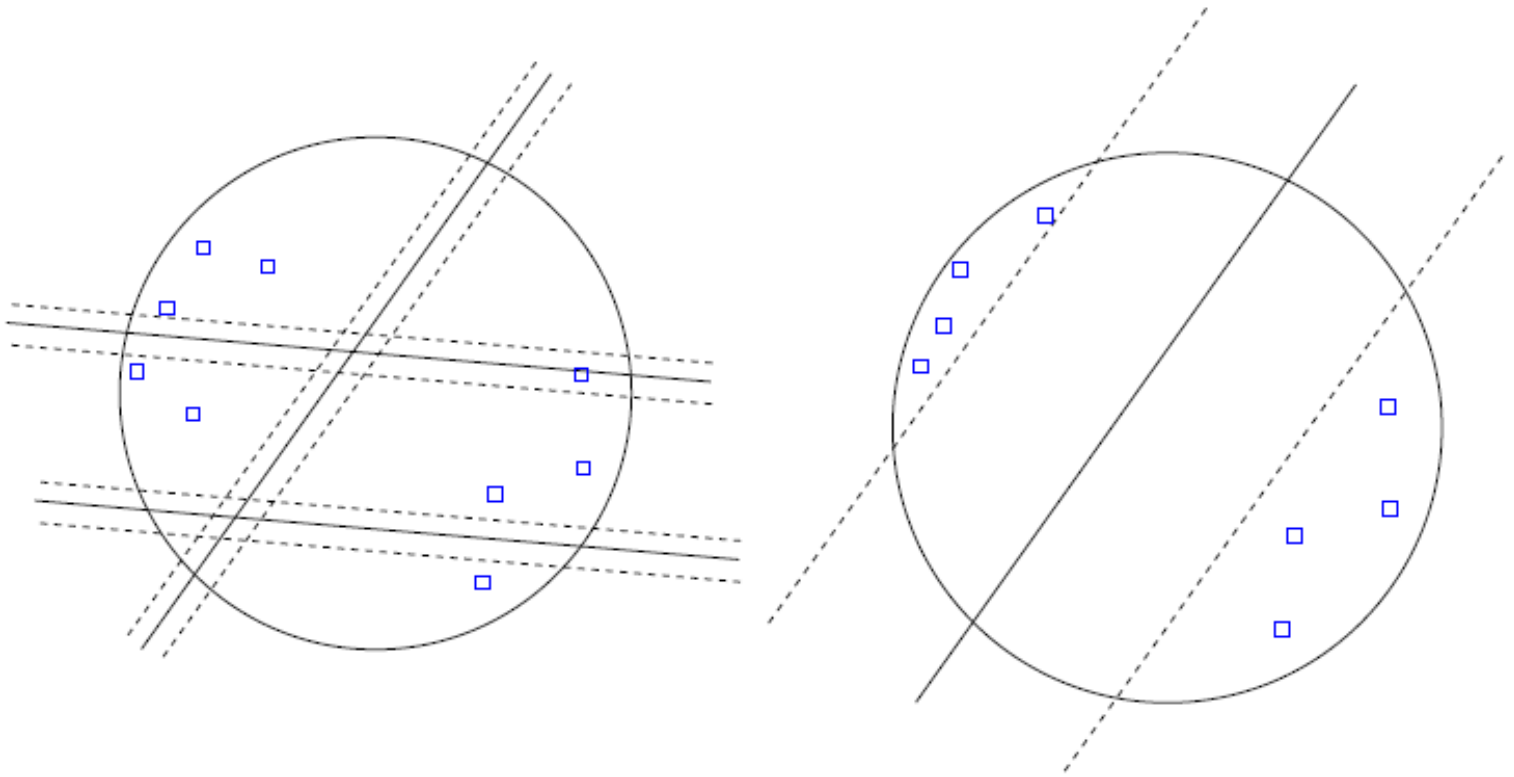
- classical statistics: **harder**, as the data are high-dimensional
- SV-Learning: (in some cases) **simpler**:

If Φ is chosen such that $\{\tilde{f}\}$ allows small training error *and* has low complexity, then we can guarantee good generalization.

The *complexity* matters, not the *dimensionality* of the space.



Margin Distributions – large margin hyperplanes



Feature Spaces & curse of dimensionality

The Support Vector (SV) approach: *preprocess* the data with

$$\begin{aligned} \Phi : \mathbf{R}^N &\rightarrow F \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}) \\ \text{where } N &\ll \dim(F). \end{aligned}$$

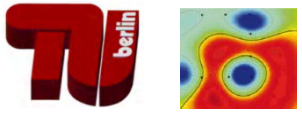
to get data $(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_N), y_N) \in F \times \mathbf{R}^M$ or $\{\pm 1\}$.

Learn $\tilde{f}$ to construct $f = \tilde{f} \circ \Phi$

- classical statistics: **harder**, as the data are high-dimensional
- SV-Learning: (in some cases) **simpler**:

If Φ is chosen such that $\{\tilde{f}\}$ allows small training error *and* has low complexity, then we can guarantee good generalization.

The *complexity* matters, not the *dimensionality* of the space.

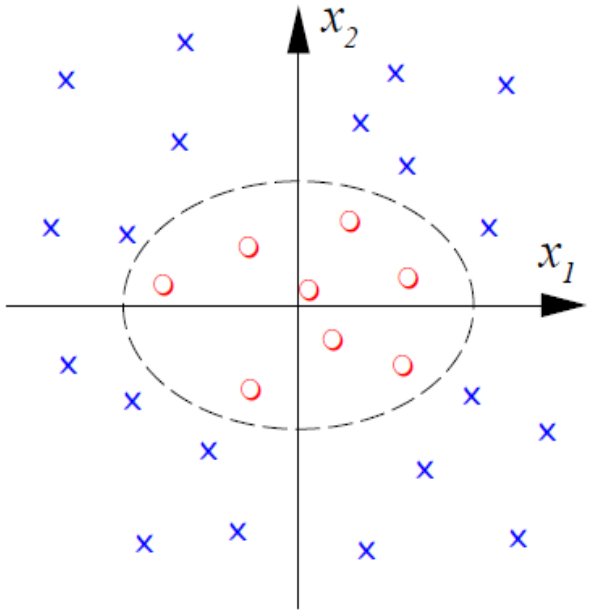


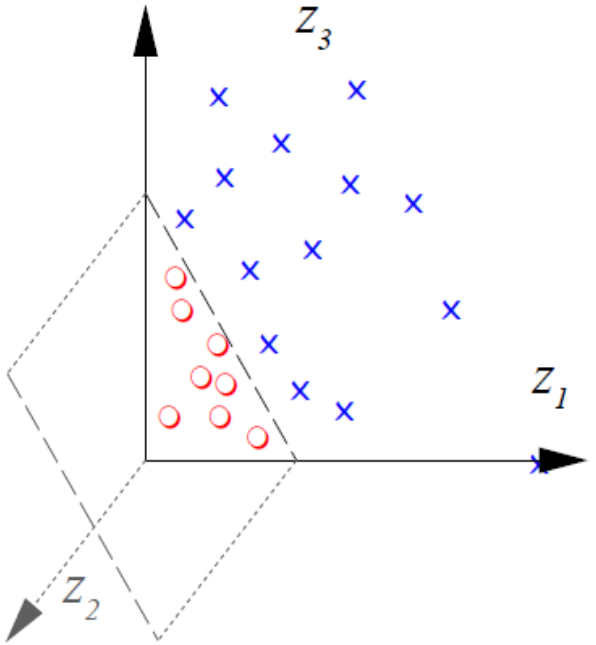
Nonlinear Algorithms in Feature Space

Example: all second order monomials

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

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



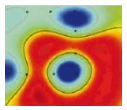


The kernel trick: an example

(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 $k(\mathbf{x}, \mathbf{y})$



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),

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

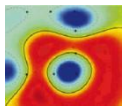
it can be expanded as

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

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 II

Examples of common kernels:

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

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)!

A RKHS representation of $\mathcal{F}$

$$\tilde{\Phi} : \mathbf{R}^N \longrightarrow \mathcal{H}, \qquad \mathbf{x} \mapsto k(\mathbf{x}, .)$$

Need a dot product $\langle ., . \rangle$ for $\mathcal{H}$ such that

$$\langle \tilde{\Phi}(\mathbf{x}), \tilde{\Phi}(\mathbf{y}) \rangle = k(\mathbf{x}, \mathbf{y}), \quad \text{i.e. require} \quad \langle k(\mathbf{x}, .), k(\mathbf{y}, .) \rangle = k(\mathbf{x}, \mathbf{y}).$$

For a Mercer kernel $k(\mathbf{x}, \mathbf{y}) = \sum_j \lambda_j \psi_j(\mathbf{x}) \psi_j(\mathbf{y})$, with $\lambda_i > 0$ for all i, and $(\psi_i \cdot \psi_j)_{L_2(\mathcal{C})} = \delta_{ij}$, this can be achieved by choosing $\langle ., . \rangle$ such that

$$\langle \psi_i, \psi_j \rangle = \delta_{ij} / \lambda_i.$$

$\mathcal{H}$, the closure of the space of all functions

$$f(\mathbf{x}) = \sum_i a_i k(\mathbf{x}, \mathbf{x}_i),$$

with dot product $\langle ., . \rangle$, is called reproducing kernel Hilbert space