Introduction to Kernel Methods

Lluís A. Belanche

belanche@lsi.upc.edu

Soft Computing Research Group
Computer Science School

Technical University of Catalonia Barcelona, Spain

Talk at the *Grup d'estudi de Machine Learning de Barcelona*

Barcelona, July 3 2014

What the talk is about

1. Introduction to kernel functions

2. Modelling with standard kernels; examples with the SVM

3. Non-standard kernels (non-numeric/non-vectorial data)

4. Other kernel-based modelling algorithms

5. Examples in real application domains

What the talk is NOT about

1. Reproducing Kernel Hilbert Spaces

2. The Representer theorem

3. Statistical Learning Theory

4. VC-dimension and friends

5. General kernel classes (invariances, universality, ...)

Introduction to kernel functions

Preliminaries

Suppose we have an i.i.d. sample of N *labelled* observations

$S = \{(\boldsymbol{x}_n, t_n)\}_{n=1,\ldots,N}$, where $\boldsymbol{x}_n \in \mathbb{R}^d, t_n \in \{0, 1\}$

In order to generalize to new inputs $\boldsymbol{x}^*$, we want to choose t^* s.t. $(\boldsymbol{x}^*, t^*)$ is in some sense similar to the training examples.

We need a **measure of similarity** for the inputs and the outputs:

$$s : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$$

$$s(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{x}' = \left\langle \boldsymbol{x}, \boldsymbol{x}' \right\rangle$$

where $\langle \cdot, \cdot \rangle$ denotes inner product in $\mathbb{R}^d$.

Introduction to kernel functions

Preliminaries

We extend it by means of a **mapping function**:

$$\phi : \mathcal{X} \to \mathcal{H}$$

as

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \right\rangle_{\mathcal{H}}$$

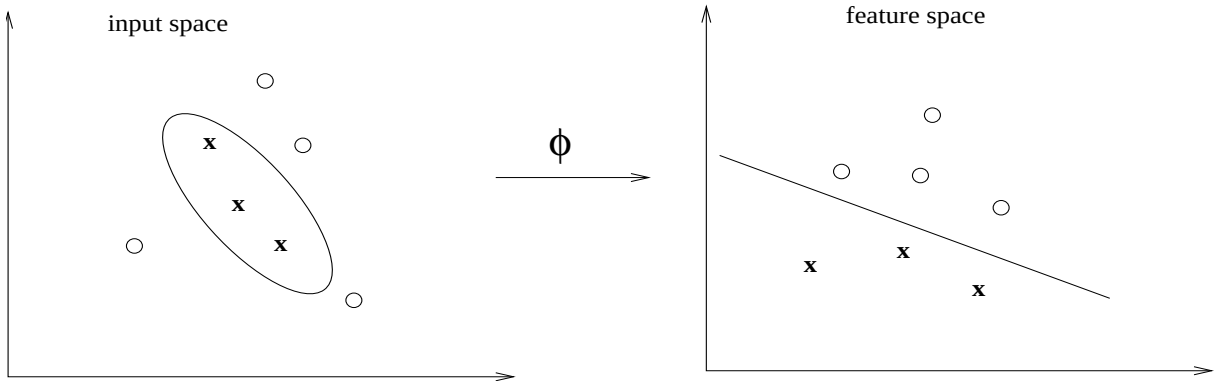
where $\mathcal{H}$ is a dot product space (aka the **feature space**).

Introduction to kernel functions

Preliminaries

A kernel function implicitly defines a map $\phi : \mathcal{X} \to \mathcal{H}$ from an input space of objects $\mathcal{X}$ into some Hilbert space $\mathcal{H}$ (the **feature space**).

The "kernel trick" consists in performing the mapping and the inner product simultaneously by defining its associated kernel function:



$$k(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \right\rangle_{\mathcal{H}}, \ \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X},$$

Introduction to kernel functions

Characterization of kernels

A symmetric function k is called *positive semi-definite* in $\mathcal{X}$ if:

for every $N \in \mathbb{N}$, and every choice $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in \mathcal{X}$,

the matrix $\mathbf{K} = (k_{ij})$, where $k_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is *positive semi-definite*.

Theorem 1 *k admits the existence of a map $\phi : \mathcal{X} \to \mathcal{H}$ s.t.*

$\mathcal{H}$ is a Hilbert space and $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle_{\mathcal{H}}$

if and only if k is a positive semi-definite symmetric function in $\mathcal{X}$.

Characterization of kernels

On positive semi-definiteness

There are many equivalent characterizations of the PSD property for real symmetric matrices. Here are some:

1. $A_{N\times N}$ is PSD if and only if all of its eigenvalues are non-negative.

2. $A_{N\times N}$ is PSD if and only if the determinants of all of its leading principal minors are non-negative.

3. $A_{N\times N}$ is PSD if and only if there is a PSD matrix B such that $BB^T = A$ (this matrix B is unique and called the square root of A).

4. $A_{N\times N}$ is PSD if and only if, $\forall \boldsymbol{c} \in \mathbb{R}^N,\ \boldsymbol{c}^T A \boldsymbol{c} \geq 0$.

Introduction to kernel functions

Back to the drawing board

$$k(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \right\rangle_{\mathcal{H}}$$

$\forall \boldsymbol{c} \in \mathbb{R}^N$,

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j k_{ij} = \sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j \left\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}} = \left\langle \sum_{i=1}^{N} c_i \phi(\boldsymbol{x}_i), \sum_{j=1}^{N} c_j \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}} \geq 0$$

1. Which holds for all choices of $\phi(\cdot)$

2. Generalizes dot product (think about the case $\phi(\boldsymbol{x}) = \boldsymbol{x}$)

Introduction to kernel functions

An example: the general linear kernel

If $A_{d\times d}$ is a PSD matrix, then the function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ given by $k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T A \boldsymbol{x}'$ is a kernel.

Proof. Since A is PSD we can write it in the form $A = BB^T$. For every $N \in \mathbb{N}$, and every choice $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in \mathbb{R}^d$, we form the matrix $K = (k_{ij})$, where $k_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T A \boldsymbol{x}_j$. Then for every $\boldsymbol{c} \in \mathbb{R}^N$:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j k_{ij} = \sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j \boldsymbol{x}_i^T A \boldsymbol{x}_j = \sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j (B^T \boldsymbol{x}_i)^T (B^T \boldsymbol{x}_j)$$

$$= \left\| \sum_{i=1}^{N} c_i (B^T \boldsymbol{x}_i) \right\|^2 \geq 0. \text{ Note that } \phi(\boldsymbol{z}) = B^T \boldsymbol{z}.$$

Introduction to kernel functions

Properties of kernels

Kernels share important properties with dot products:

1. Symmetry

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

2. Cauchy-Schwarz inequality

$$|k(\boldsymbol{x}, \boldsymbol{x}')| \leq \sqrt{k(\boldsymbol{x}, \boldsymbol{x})}\sqrt{k(\boldsymbol{x}', \boldsymbol{x}')}$$

3. Definiteness

$$k(\boldsymbol{x}, \boldsymbol{x}) = \|\phi(\boldsymbol{x})\|^2 \geq 0$$

Introduction to kernel functions

Construction of kernels

- Inner products: finite (sums), infinite countable (series) or infinite uncountable (integrals)
- Sums, products, direct sums and tensor products
- Multiplication by positive coefficient
- Limits of point-wise convergent sequences of kernels
- Composition with certain analytic functions
- Normalization

Introduction to kernel functions

Example 1: polynomial combinations

1. If k is a kernel and p is a polynomial of degree m with positive coefficients, then the function

$$k_p(\boldsymbol{x}, \boldsymbol{x}') = p(k(\boldsymbol{x}, \boldsymbol{x}'))$$

   is also a kernel.

2. The special case where k is linear and $p(z) = (az + b)^m, a > 0, b \geq 0$ leads to the so-called **polynomial kernel**

Introduction to kernel functions

Example 1: polynomial combinations

Consider the kernel family:

$$\Big\{k_i(\boldsymbol{x},\boldsymbol{x}') = \alpha_i\Big(\big\langle \boldsymbol{x},\boldsymbol{x}'\big\rangle + a_i\Big)^{\beta_i} \;/\; \beta_i \in \mathbb{N}, \alpha_i > 0, a_i \geq 0\Big\}$$

For any $q > 0 \in \mathbb{N}$,

$$\sum_{i=0}^{q} k_i(\boldsymbol{x},\boldsymbol{x}')$$

is a kernel.

Introduction to kernel functions

Example 1: polynomial combinations

Consider the particular case $a_i = 0, \beta_i = i$ and $\alpha_i = \frac{\alpha^i}{i!}$, for some real $\alpha > 0$, and take the limit $q \to \infty$.

The obtained series is convergent for all α and the resulting kernel is:

$$\sum_{i=0}^{\infty} \frac{\alpha^i}{i!} (\left\langle \boldsymbol{x}, \boldsymbol{x}' \right\rangle)^i = e^{\alpha \langle \boldsymbol{x}, \boldsymbol{x}' \rangle}$$

Assume that $x, x' \in \mathbb{R}$; then $\exp(\alpha x x') = \langle \phi(x), \phi(x') \rangle$ with $\phi(z) = \left(\sqrt{\frac{\alpha^i}{i!}} z^i \right)_{i=0}^{\infty}$, and therefore we have designed a feature space of infinite dimension!

Introduction to kernel functions

Example 2: user-defined kernel

Suppose we have designed a "kernel" on objects $\boldsymbol{x}, \boldsymbol{x}'$ in $\mathcal{X} = [-1, +1]^d$ from a set of descriptors $f_j : \mathcal{X} \to \mathbb{R}$ as the function:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{\sum_{i=1}^{\delta} f_i(\boldsymbol{x}) f_i(\boldsymbol{x}')}{\sqrt{\theta - \langle \boldsymbol{x}, \boldsymbol{x}' \rangle}} \tag{1}$$

where δ is the number of object descriptors and θ is a free parameter.

Under what conditions does (1) define a kernel?

Introduction to kernel functions

Example 2: user-defined kernel

Being an inner product, the numerator is immediately a kernel. Since the denominator must be positive, we conclude that $\theta > d$. Now define

$$f(z) = \frac{1}{\sqrt{\theta - z}}, \theta > d$$

Theorem 2 *Let k be a PSD kernel and $f(z) = \sum_{n=0}^{\infty} a_n z^n$ with $a_n \geq 0$ and radius of convergence R; if $|k(\cdot,\cdot)| < R$, then $f \circ k$ is a PSD kernel.*

We find $a_n = (2^n \theta^{(2n+1)/2})^{-1} \prod_{i=1}^{n} (2i-1)$, which is positive for all $\theta > 0$. Therefore both the denominator and numerator are kernels and so is their product, for $\theta > d$.

Introduction to kernel functions

Normalization

If k is a kernel, then so is:

$$k_n(\boldsymbol{x}, \boldsymbol{x}') = \frac{k(\boldsymbol{x}, \boldsymbol{x}')}{\sqrt{k(\boldsymbol{x}, \boldsymbol{x})}\sqrt{k(\boldsymbol{x}', \boldsymbol{x}')}}$$

Moreover,

$$|k_n(\boldsymbol{x}, \boldsymbol{x}')| \leq 1 \text{ and } k_n(\boldsymbol{x}, \boldsymbol{x}) = 1.$$

Introduction to kernel functions

The polynomial kernel

- Suppose we take $k(\boldsymbol{u}, \boldsymbol{v}) = \langle \boldsymbol{u}, \boldsymbol{v} \rangle^D, \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d$ (a simple choice).
- What is the underlying mapping ϕ here?

  $\Longrightarrow$ Answer: this choice of kernel corresponds to a map ϕ leading into the space spanned by all products of exactly D dimensions of $\mathbb{R}^d$.
- Let us take, for instance, $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^2$, and take $D = 2$:

$$k(\boldsymbol{u}, \boldsymbol{v}) = \langle \boldsymbol{u}, \boldsymbol{v} \rangle^2 = \left[\left\langle \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \right\rangle \right]^2$$

$$= (u_1 v_1 + u_2 v_2)^2 = (u_1 v_1)^2 + 2 u_1 v_1 u_2 v_2 + (u_2 v_2)^2$$

$$= u_1^2 v_1^2 + (\sqrt{2} u_1 u_2)(\sqrt{2} v_1 v_2) + u_2^2 v_2^2$$

$$= \left\langle \begin{pmatrix} u_1^2 \\ \sqrt{2} u_1 u_2 \\ u_2^2 \end{pmatrix}, \begin{pmatrix} v_1^2 \\ \sqrt{2} v_1 v_2 \\ v_2^2 \end{pmatrix} \right\rangle = \langle \phi(\boldsymbol{u}), \phi(\boldsymbol{v}) \rangle$$

- Therefore, $\phi : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ with $\phi(\boldsymbol{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^T$

Introduction to kernel functions

The Gaussian kernel

We say that a kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is:

Translation invariant if it has the form $k(\boldsymbol{x}, \boldsymbol{x}') = T(\boldsymbol{x} - \boldsymbol{x}')$, where $T : \mathbb{R}^d \to \mathbb{R}$ is a differentiable function.

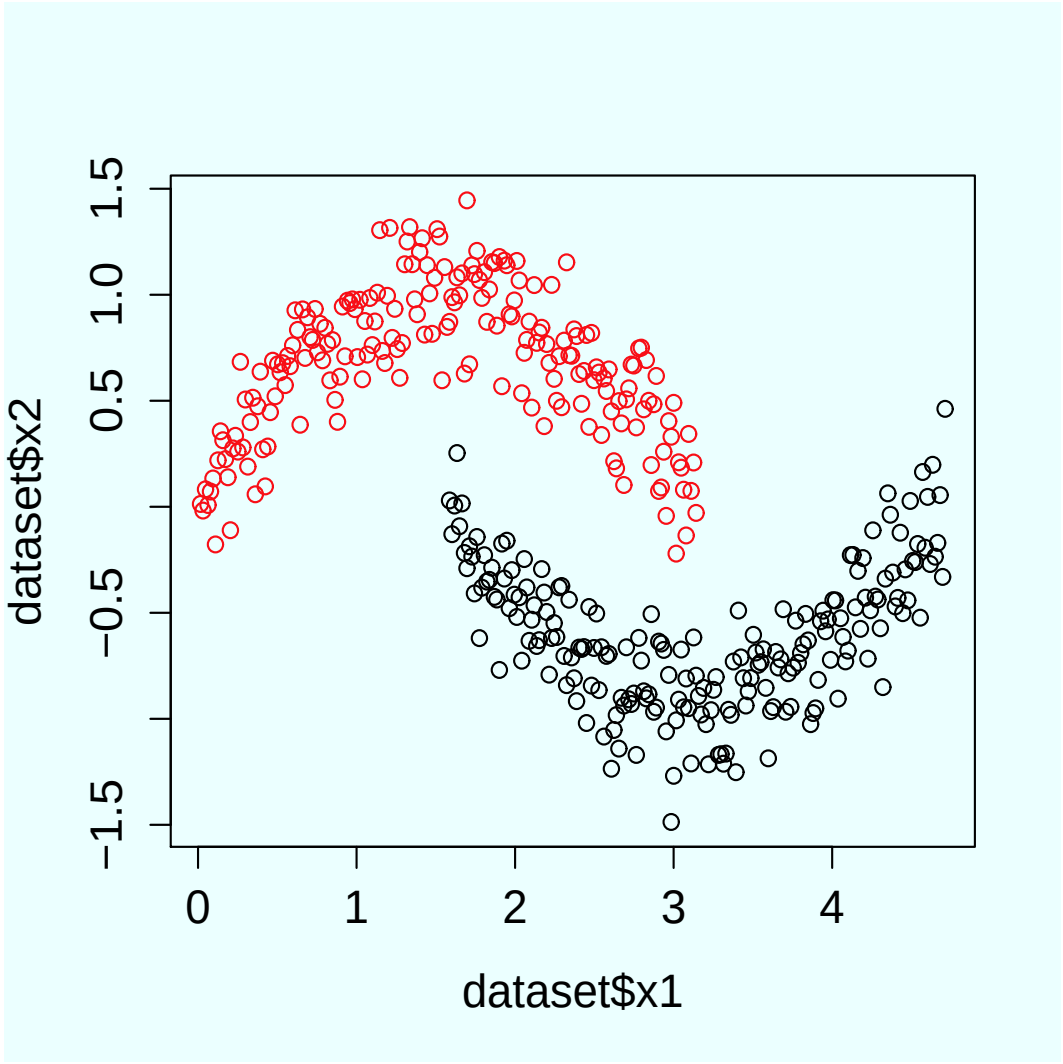
Radial if it has the form $k(\boldsymbol{x}, \boldsymbol{x}') = t(\|\boldsymbol{x} - \boldsymbol{x}'\|)$, where $t : [0, \infty) \to [0, \infty)$ is a differentiable function. Radial kernels fulfill $k(\boldsymbol{x}, \boldsymbol{x}) = t(0)$.

Consider the function $t(z) = \exp(-\gamma z^2), \gamma > 0$. The resulting radial kernel is known as the **Gaussian kernel**:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|^2)$$

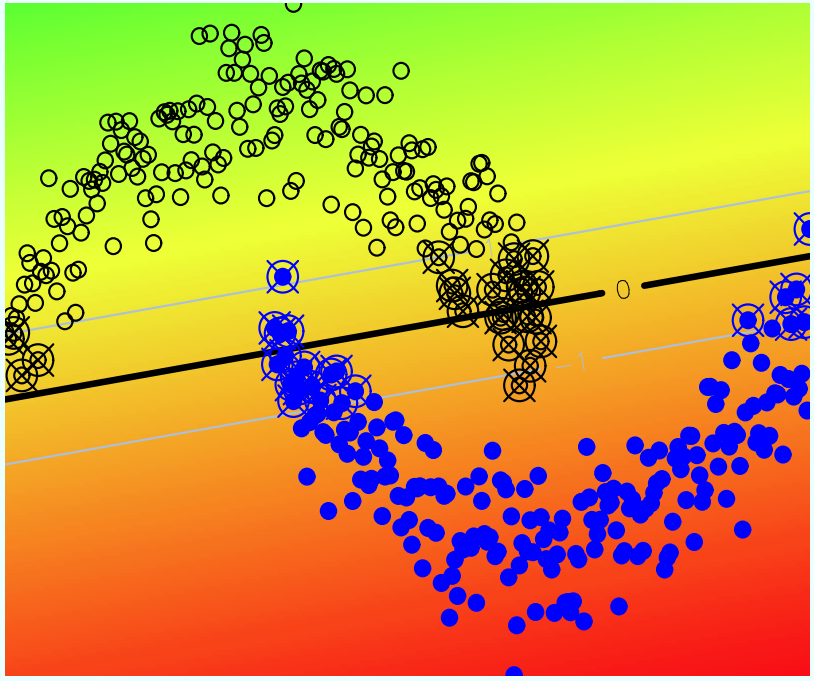
Note that some people call it the RBF kernel *par excellence*!

Modelling with the SVM using standard kernels



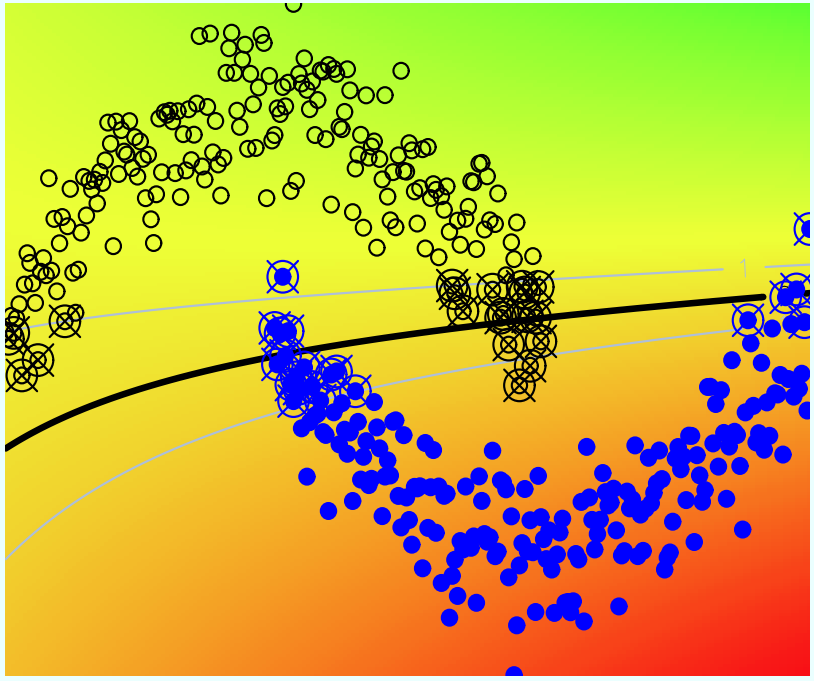
2D artificial data (2 sinusoids), size $N = 200$ per class

Modelling with the SVM using standard kernels



Linear kernel ($C = 1$), $k(\boldsymbol{x}, \boldsymbol{x}') = 0{,}5\,\langle \boldsymbol{x}, \boldsymbol{x}' \rangle$, 46 SVs

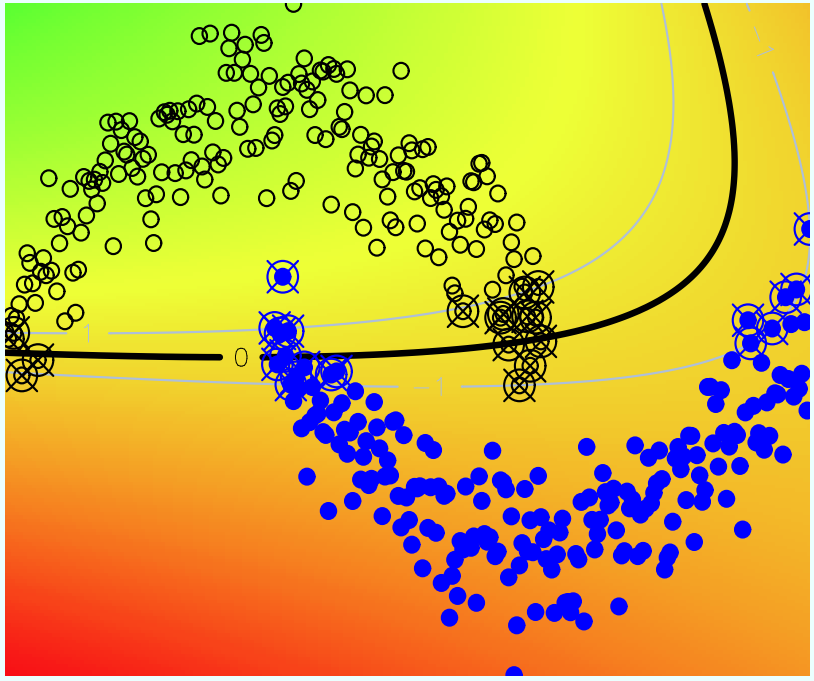
Modelling with the SVM using standard kernels



poly.2

Quadratic kernel ($C = 1$), $k(\boldsymbol{x}, \boldsymbol{x}') = (0{,}5\,\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + 1)^2$, 41 SVs

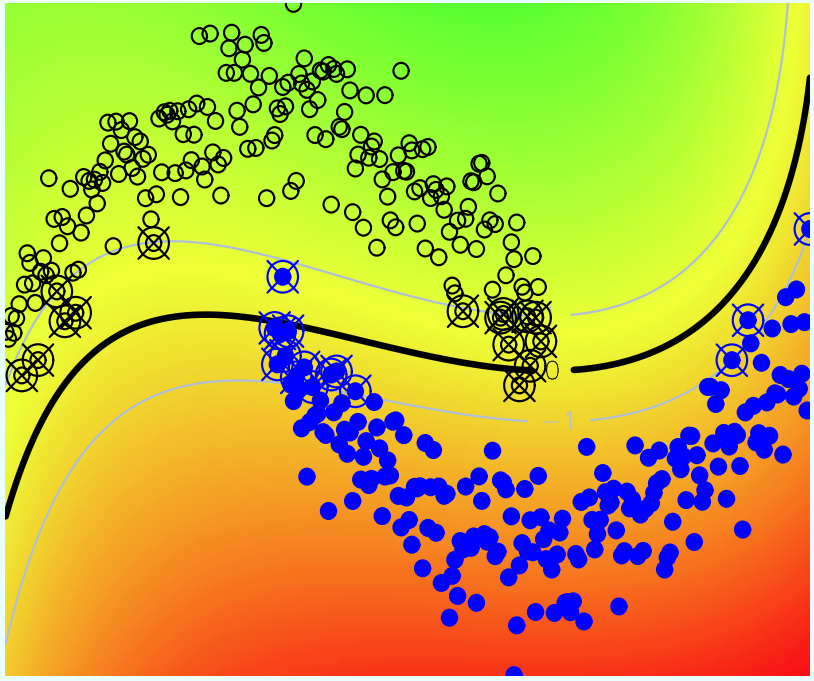
Modelling with the SVM using standard kernels



poly.2

Quadratic kernel ($C = 50$), $k(\boldsymbol{x}, \boldsymbol{x}') = (0{,}5\,\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + 1)^2$, 32 SVs

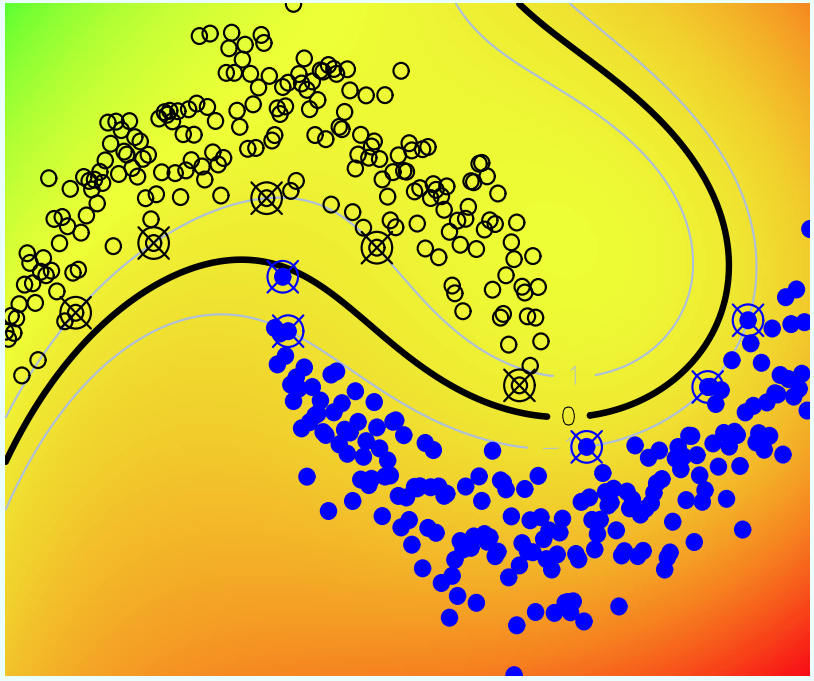
Modelling with the SVM using standard kernels



poly.3

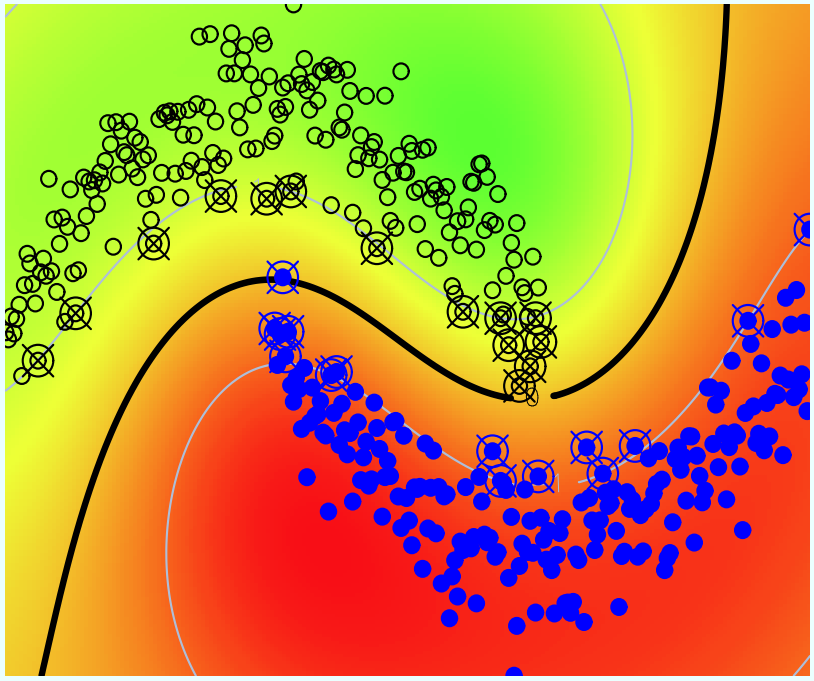
Cubic kernel ($C = 1$), $k(\boldsymbol{x}, \boldsymbol{x}') = (0{,}5\,\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + 1)^3$, 30 SVs

Modelling with the SVM using standard kernels



Cubic kernel ($C = 50$), $k(\boldsymbol{x}, \boldsymbol{x}') = (0{,}5 \langle \boldsymbol{x}, \boldsymbol{x}' \rangle + 1)^3$, 10 SVs

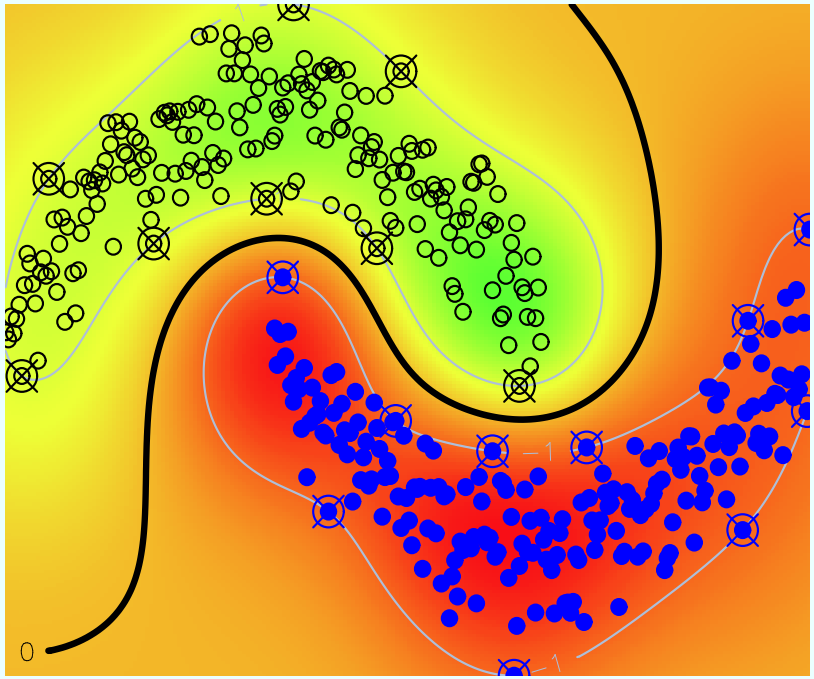
Modelling with the SVM using standard kernels



RBF

RBF Gaussian kernel ($C = 1$), $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-0{,}5\|\boldsymbol{x} - \boldsymbol{x}'\|^2)$, 29 SVs

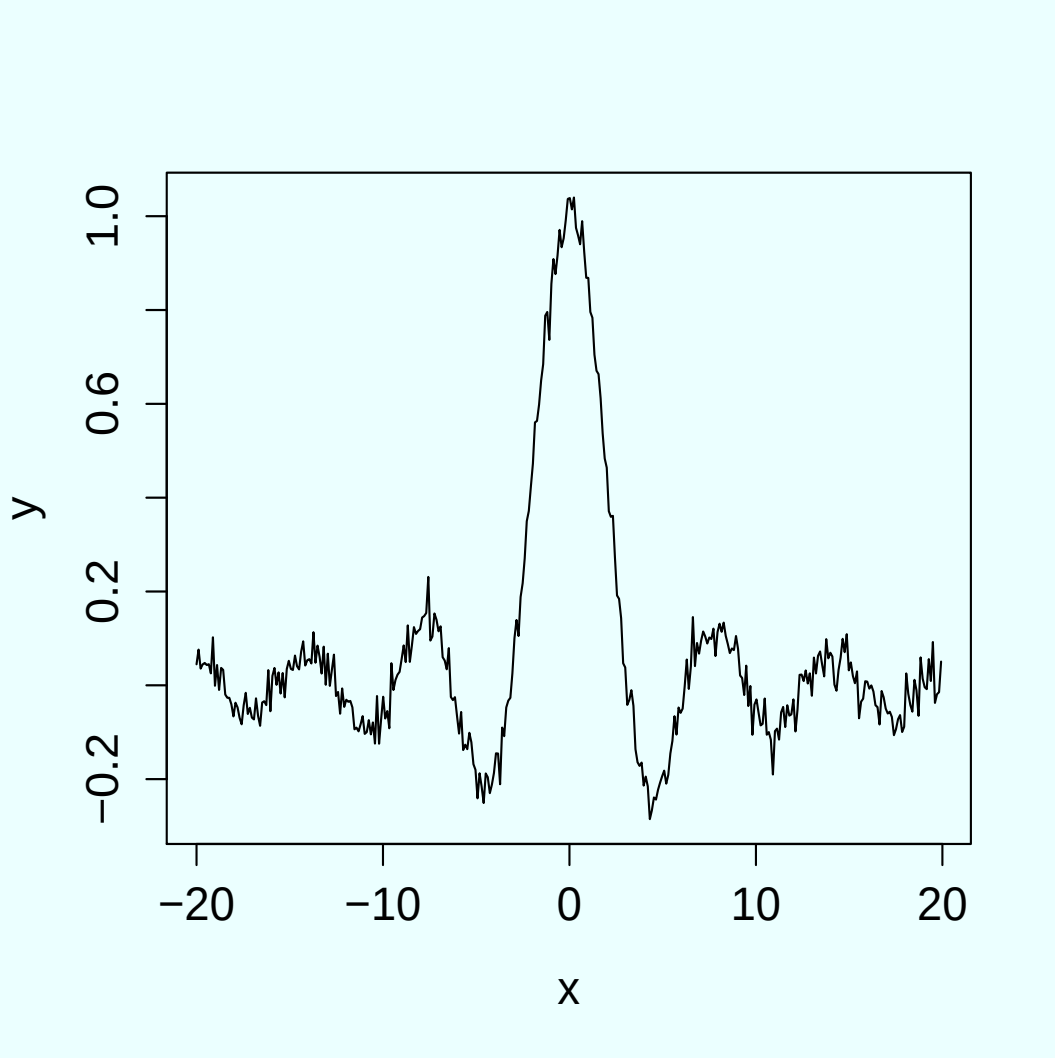
Modelling with the SVM using standard kernels



RBF

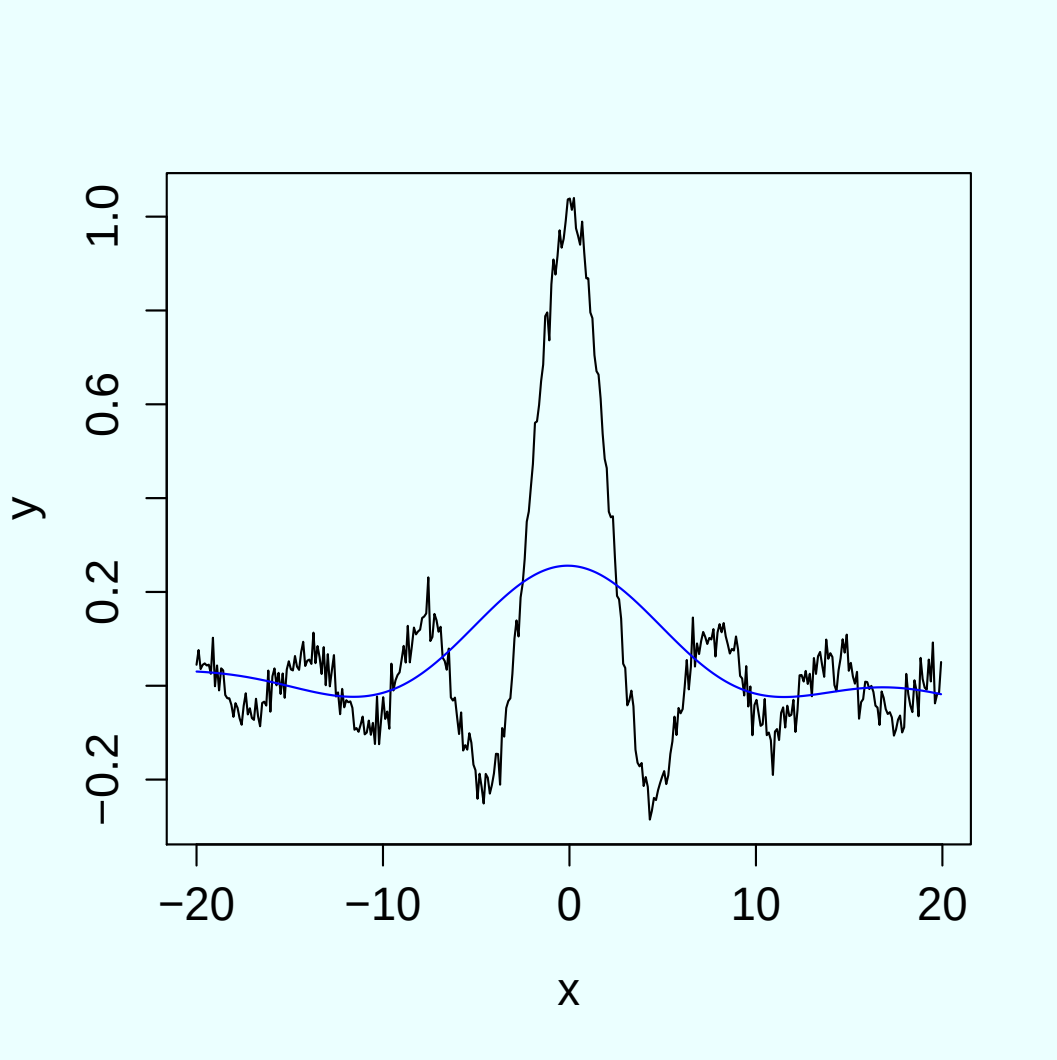
RBF Gaussian kernel ($C = 50$), $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-2\|\boldsymbol{x} - \boldsymbol{x}'\|^2)$, 18 SVs

Modelling with the SVM using standard kernels



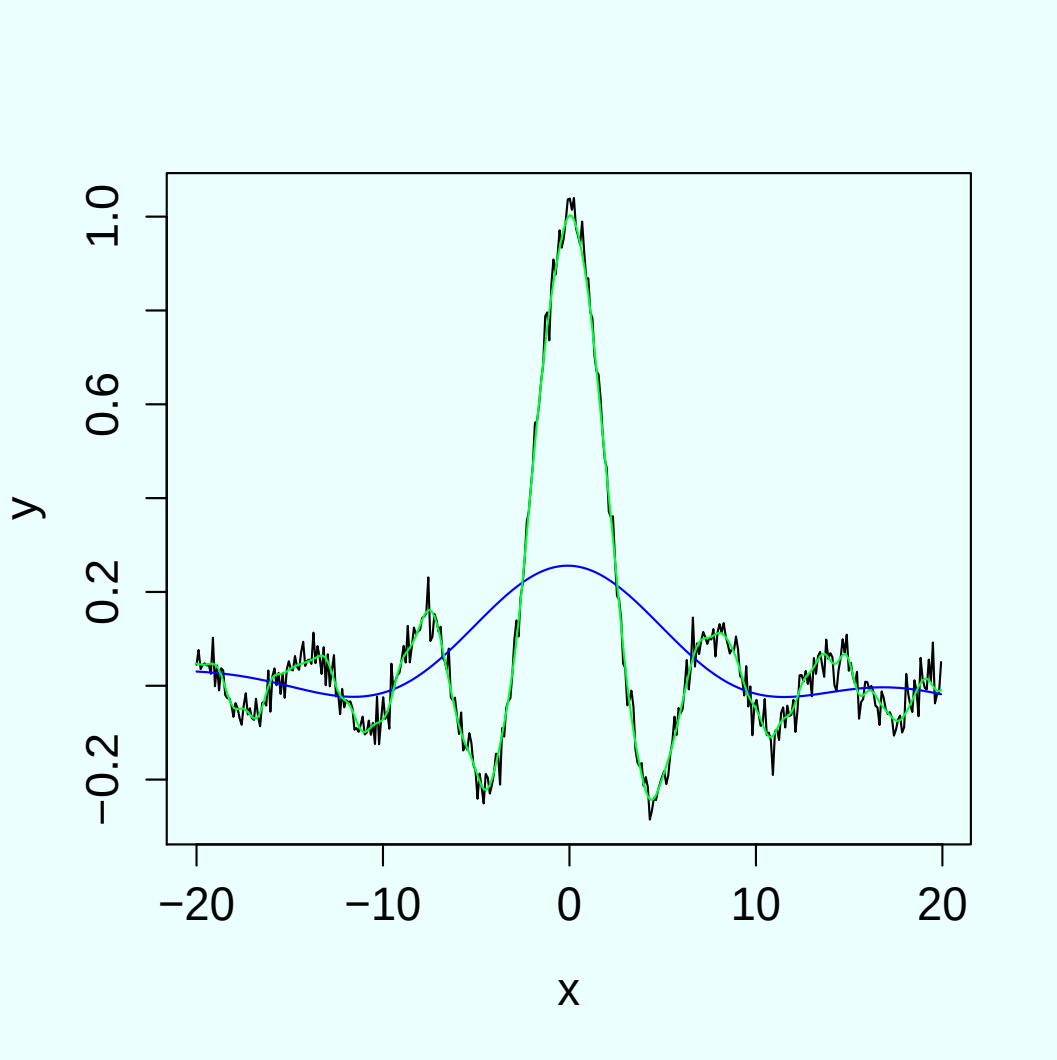
A really nice looking function: $\frac{\sin x}{x}$+N(0,0.03^2) in $[-20 : 20]$

Modelling with the SVM using standard kernels



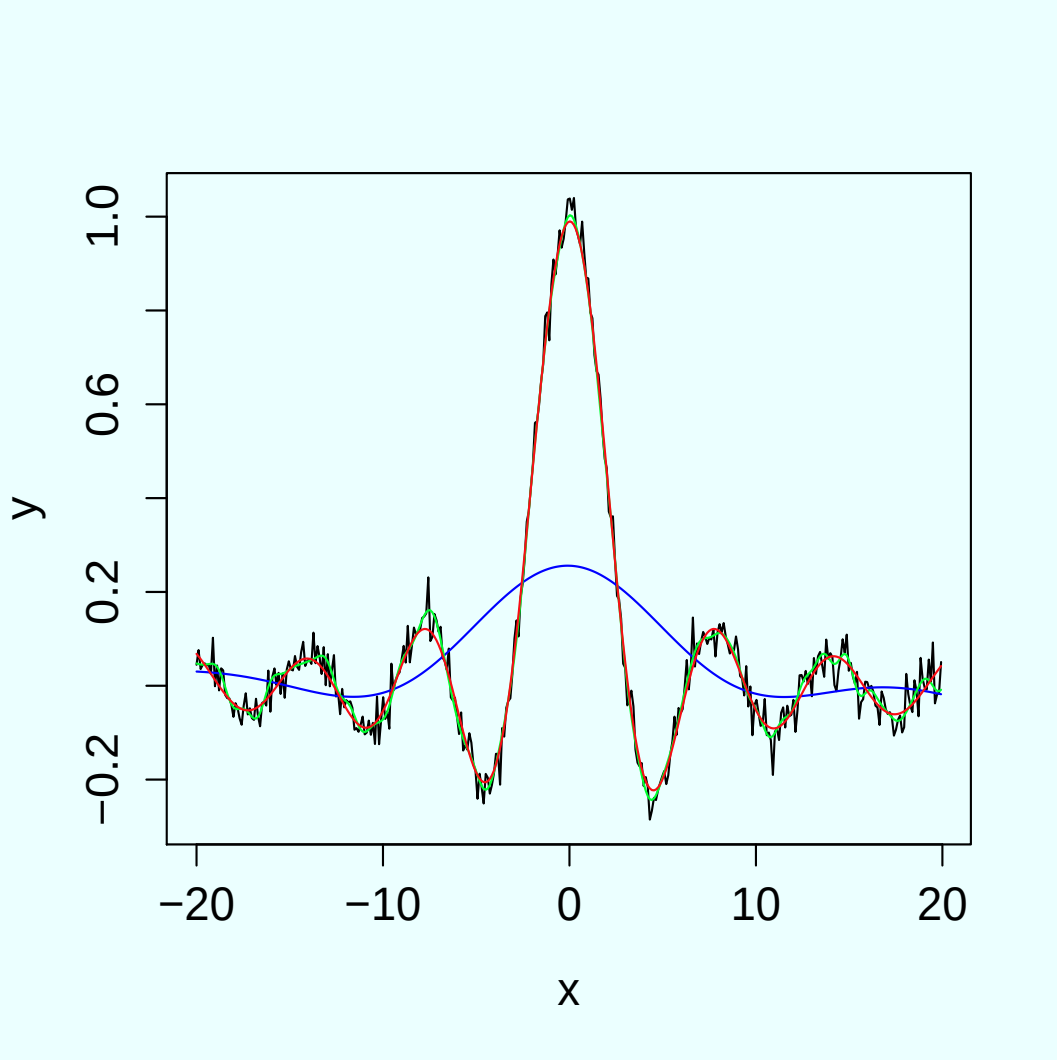
With this choice of the 'epsilon' and 'gamma' parameters, the SVM underfits the data (blue line)

Modelling with the SVM using standard kernels



With this choice of the 'epsilon' and 'gamma' parameters, the SVM overfits the data (green line)

Modelling with the SVM using standard kernels



With this choice of the 'epsilon' and 'gamma' parameters, the SVM has a very decent fit (red line)

Non-standard kernels

- Kernels on sets (bitstrings)
- Graph kernels
- Generative kernels
- Convolution kernels
- Tree kernels
- String kernels (text)

... and many others (functional data, categorical data, ...)

Non-standard kernels

Set kernels

Consider a feature space with one feature for every subset $A \subseteq \{1, \cdots, d\}$ of the input features. For $\boldsymbol{x} \in \mathbb{R}^d$, the feature A is given by $\phi_A(\boldsymbol{x}) = \prod_{i \in A} x_i$.

The *all-subsets* kernel is defined by the mapping

$$\phi : \boldsymbol{x} \to (\phi_A(\boldsymbol{x}))_{A \subseteq \{1, \cdots, d\}}$$

and then

$$k(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \right\rangle = \sum_{A \subseteq \{1, \cdots, d\}} \phi_A(\boldsymbol{x}) \phi_A(\boldsymbol{x}')$$

$$= \sum_{A \subseteq \{1, \cdots, d\}} \prod_{i \in A} x_i x_i' = \prod_{i=1}^{d} (1 + x_i x_i')$$

Non-standard kernels

The Jensen-Shannon divergence kernel

Generative kernels are adequate when a statistical model for the data is available:

- The Jensen-Shannon divergence gives a measure of whether two samples are drawn from the same source distribution

- It can be interpreted as the average distance between each probability distribution and the average distribution

$$k(\mathcal{P}, \mathcal{P}') = \exp\left(-\gamma\left\{H\left(\frac{\mathcal{P}+\mathcal{P}'}{2}\right) - \frac{1}{2}\left(H(\mathcal{P}) + H(\mathcal{P}')\right)\right\}\right),\ \gamma > 0$$

where $\mathcal{P}, \mathcal{P}'$ are two probability distributions with support in $\mathcal{X}$ and H is Shannon's entropy.

Non-standard kernels

Bitstrings/Sets

Theorem 3 *Let* $\boldsymbol{x}, \boldsymbol{x}' \in \{0,1\}^d$*. The* Jaccard Coefficient*, the fraction of* $1-1$ *matches, is a valid kernel.*

Proof. For every $N \in \mathbb{N}$, and every choice $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in \{0,1\}^d$, we form the matrix $K = (k_{ij})$, where $k_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{d} \sum_{k=1}^{d} x_{ik} x_{jk} = \frac{1}{d} \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle$

Then we have, for any $\boldsymbol{c} \in \mathbb{R}^N$:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j \frac{1}{d} \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle = \frac{1}{d} \left\langle \sum_{i=1}^{N} c_i \boldsymbol{x}_i, \sum_{j=1}^{N} c_j \boldsymbol{x}_j \right\rangle = \frac{1}{d} \left\| \sum_{i=1}^{N} c_i \boldsymbol{x}_i \right\|^2 \geq 0$$

Non-standard kernels

The Spectrum (aka n-Gram) kernel

- Let Σ be a finite alphabet
- An n-Gram is a block of n adjacent characters from alphabet Σ
- The number of distinct n-Grams in a text $\boldsymbol{x}$ is $\leq |\Sigma|^n$

Define $k(\boldsymbol{x}, \boldsymbol{x}') = \sum_{s \in \Sigma^n} |s \in \boldsymbol{x}| \cdot |s \in \boldsymbol{x}'|$

Example: Protein **aababc** with alphabet $\Sigma = \{a, b, c\}$, $n = 2$:

aa	ab	ac	ba	bb	bc	...
1	2	0	1	0	1	...

Non-standard kernels

The Spectrum (aka n-Gram) kernel

- Notice that $\phi(\boldsymbol{x}) = \Big(|s \in \boldsymbol{x}|\Big)_{s \in \Sigma^n}$

- While the feature space is large (even for fairly small n), the feature vectors are sparse; it has been shown that this kernel can be computed in $O(|\boldsymbol{x}| + |\boldsymbol{x}'|)$ time and memory

- Given $k \geq 1$, the k-spectrum of an input sequence $\boldsymbol{x}$ is the set of all the contiguous subsequences of length k that $\boldsymbol{x}$ contains

Non-standard kernels

Kernels from graphs

Consider a graph $G = (V, E)$, where the set of vertices V are the data points and E is the set of edges.

The idea is to compute a (base) similarity matrix $B_{|V|\times|V|}$ whose entries are the weights of the edges and consider $B^2 = BB^T$.

Examples:

- protein-protein interactions
- people-to-people interactions

If the graph G is unweighted then the (i, j) element of B^2 is the number of common friends between data points i and j (or the number of paths of length 2 between i and j) and it can be thought of as a measure of their similarity.

Non-standard kernels

Kernels from graphs

Note:

1. The entries of B may be real-valued numbers (similarities)
2. Higher powers of B measure higher order similarities
3. Only the even powers are guaranteed to be psd

Consider, for a given $\lambda \in (0, 1)$:

$$\sum_{k=0}^{\infty} \frac{1}{k!} \lambda^k B^k = \exp(\lambda B)$$

If $B = U \Lambda U^T$ is the spectral decomposition of B, then $B^2 = (U \Lambda U^T)(U \Lambda U^T) = U \Lambda^2 U^T$.

In general, we have $B^k = U \Lambda^k U^T$ and therefore

$$K = \exp(\lambda B) = U \exp(\lambda \Lambda) U^T$$

is an example of a *diffusion* kernel.

Other kernel-based modelling algorithms

- SVMs for classification, regression and novely detection
- Relevance Vector Machine and Gaussian Processes
- Kernel Nearest Neighbours
- Clustering (Spectral Clustering, Kernel k-means)
- Kernel Discriminant Analysis (KFDA)
- Kernel Principal Components Analysis (KPCA)
- Kernel Canonical Correlation Analysis (KCCA)
- Kernel Independent Component Analysis (KICA)
- Kernel Regression (Linear and Logistic)

... and many others

Other kernel-based modelling algorithms

Further characterization of kernels

A symmetric function k is called *conditionally positive semi-definite* in $\mathcal{X}$ if for every $N \in \mathbb{N}$, and every choice $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in \mathcal{X}$, the matrix $\mathbf{K} = (k_{ij})$, where $k_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is *conditionally positive semi-definite*.

A real symmetric matrix $A_{N \times N}$ is CPSD if and only if, for every $\boldsymbol{c} \in \mathbb{R}^N$ such that $\sum_{i=1}^{N} \boldsymbol{c}_i = 0,\ \boldsymbol{c}^T A \boldsymbol{c} \geq 0$.

It turns out that it suffices for a kernel to be CPSD! Since the class of CPSD kernels is larger than that of PSD kernels, a larger set of learning algorithms are prone to kernelization.

Other kernel-based modelling algorithms

What is "kernelizing"?

- If an algorithm is **distance-based**, the idea is:

  1. substitute $||\boldsymbol{x}_i - \boldsymbol{x}_j||$ by the feature space counterpart $||\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)||_{\mathcal{H}}$

  2. replace it by $\sqrt{-k(\boldsymbol{x}_i, \boldsymbol{x}_j)}$, where k is any CPSD kernel

- If an algorithm is **inner product-based**, the idea is:

  1. substitute $\left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle$ by the feature space counterpart $\left\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}}$

  2. replace it by $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where k is any PSD kernel.

Other kernel-based modelling algorithms

First example: nearest neighbours

kNN classifies new examples by finding the k closest examples in the sample and taking a majority vote

$$k(\boldsymbol{x},\boldsymbol{x})+k(\boldsymbol{x}',\boldsymbol{x}')-2k(\boldsymbol{x},\boldsymbol{x}') = \langle\phi(\boldsymbol{x}),\phi(\boldsymbol{x})\rangle+\left\langle\phi(\boldsymbol{x}'),\phi(\boldsymbol{x}')\right\rangle-2\left\langle\phi(\boldsymbol{x}),\phi(\boldsymbol{x}')\right\rangle$$

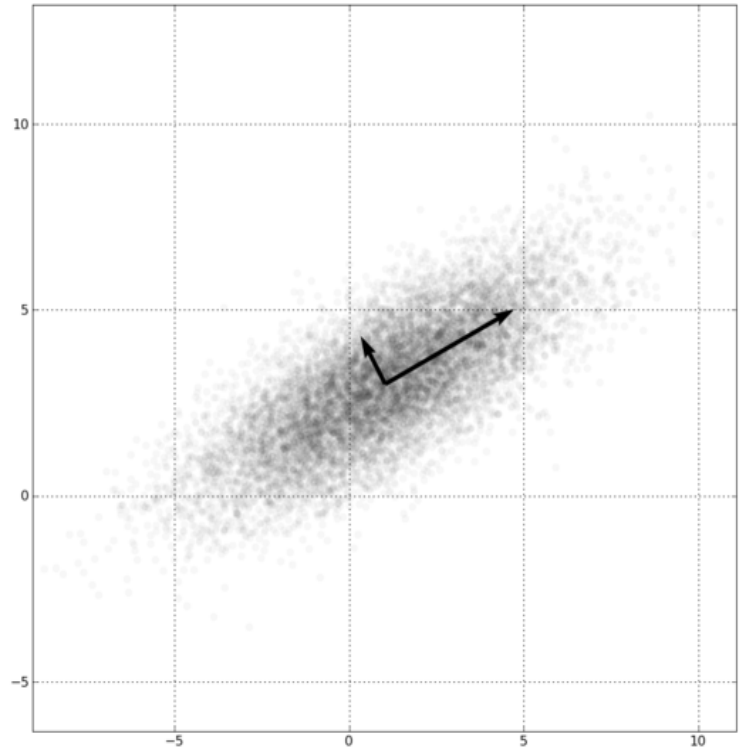
$$= \|\phi(\boldsymbol{x})\|^2 + \|\phi(\boldsymbol{x}')\|^2 - 2\left\langle\phi(\boldsymbol{x}),\phi(\boldsymbol{x}')\right\rangle = \|\phi(\boldsymbol{x}) - \phi(\boldsymbol{x}')\|^2 = d^2_{\mathcal{H}}(\boldsymbol{x},\boldsymbol{x}')$$

- This distance in *feature space* can be calculated using 3 calls to the kernel function (or 1 if k is normalized), for k PSD

- Note that $\sqrt{-k(\boldsymbol{x},\boldsymbol{x}')}$ is also a distance, for k CPSD

Other kernel-based modelling algorithms

Second example: kernel PCA



(Source: Wikipedia)

Other kernel-based modelling algorithms

Second example: kernel PCA

- We are given a data set $X = \{\boldsymbol{x}_n\}, \boldsymbol{x}_n \in \mathbb{R}^d$ for $n = 1, ..., N$ which is centered around the origin, i.e. $\sum_{i=1}^{N} \boldsymbol{x}_n = 0$.

- The sample covariance matrix of the data is $C = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T$.

- The goal of PCA is to replace the original axes with the principal components (PC) of the data

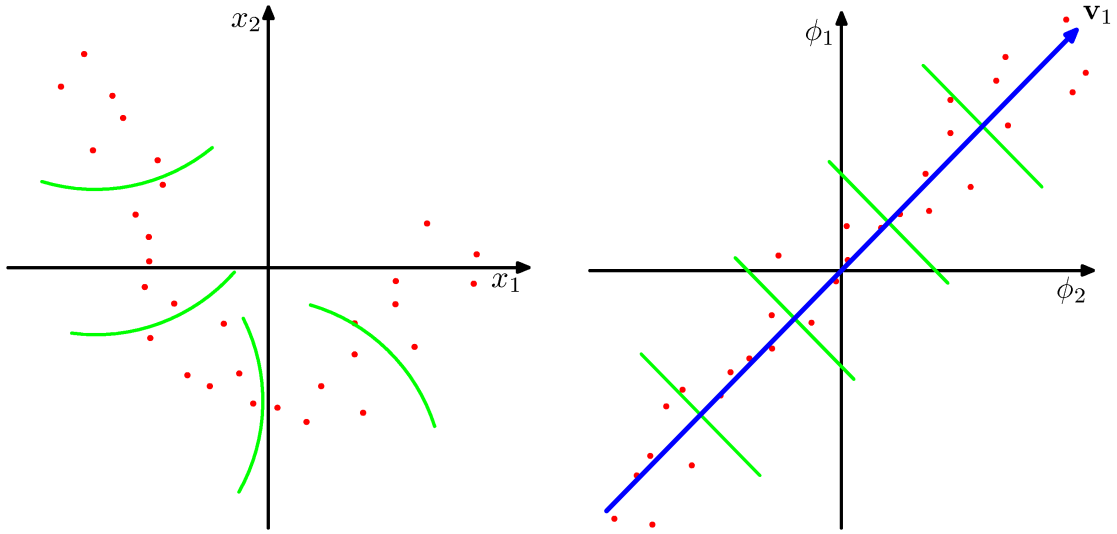
Other kernel-based modelling algorithms

Second example: kernel PCA

- The PCs are the d orthogonal vectors (seen as projection directions) leading to the maximum variance of projected data
- These vectors correspond to the d (orthogonal) eigenvectors of C ordered with decreasing eigenvalue $\lambda_1 > \lambda_2 \ldots > \lambda_d \geq 0$
- If we could define our principal components to be arbitrary manifolds we could get higher variance and better separability
- If we first map our data into a higher dimensional space where the data falls neatly onto some hyperplane or where the data is separable we can perform Standard PCA in that higher dimensional space

Other kernel-based modelling algorithms

Second example: kernel PCA



Input space $\boldsymbol{x}$ projected into feature space $\phi(\boldsymbol{x})$

Left: Green lines indicate non-linear projections onto the first PC

Right: Green lines indicate linear projections onto the first PC

(from *Pattern Recognition and Machine Learning*, C. M. Bishop, Springer, 2006)

Other kernel-based modelling algorithms

Second example: kernel PCA

Kernel PCA allows us to perform PCA in this higher dimension using the kernel trick, doing all our calculations in a lower dimension.

Recall the idea of mapping input data into some Hilbert space (called the *feature space*) via a non-linear mapping $\phi : \mathbb{R}^d \to \mathcal{H}$.

The new sample covariance matrix C of the data is defined as

$$C = \frac{1}{N} \sum_{n=1}^{N} \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^T$$

Other kernel-based modelling algorithms

Second example: kernel PCA

1. We are given a data set of d-dimensional vectors $X = \{\boldsymbol{x}_n\}$ for $n = 1, ..., N$ which we center around the origin, as $\boldsymbol{x}_n \leftarrow \boldsymbol{x}_n - \frac{1}{N} \sum_{m=1}^{N} \boldsymbol{x}_m$.

2. Compute the sample covariance matrix of the data as $C = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T$.

3. Compute the eigenvalues/vectors $(\lambda_n, \boldsymbol{v}_n)$ of C; pick a number $l < d$.

4. Let $\hat{\boldsymbol{x}}_n = V \boldsymbol{x}_n$, where $V = [\boldsymbol{v}_1^T; \ldots; \boldsymbol{v}_l^T]$.

5. The result is the l-dimensional data sample $\hat{X} = \{\hat{\boldsymbol{x}}_n\}$ for $n = 1, ..., N$

Other kernel-based modelling algorithms

Second example: kernel PCA

Centered data is required to perform an effective PCA.

Although the $\{\boldsymbol{x}_n\}$ were centered data, the $\{\phi(\boldsymbol{x}_n)\}$ are *not* guaranteed to be centered in the feature space!

We have to 'center' the kernel matrix before doing Kernel PCA:

$$\mathbf{K} \leftarrow \mathbf{K} - \frac{1}{N}\mathbf{1}\mathbf{K} - \frac{1}{N}\mathbf{K}\mathbf{1} + \frac{1}{N^2}\mathbf{1}\mathbf{K}\mathbf{1}$$

where $\mathbf{1}$ is a $N \times N$ matrix of ones

Other kernel-based modelling algorithms

Second example: kernel PCA

1. Same as step 1. for PCA.

2. Choose a kernel function k. Compute the kernel matrix of the data as $\mathbf{K} = (k_{nm})$, where $k_{nm} = k(\boldsymbol{x}_n, \boldsymbol{x}_m)$.

3. Center the kernel matrix, as: $\mathbf{K} \leftarrow \mathbf{K} - \frac{1}{N}\mathbf{1}\mathbf{K} - \frac{1}{N}\mathbf{K}\mathbf{1} + \frac{1}{N^2}\mathbf{1}\mathbf{K}\mathbf{1}$

4. Compute the eigenvalues/vectors $(\lambda_n, \boldsymbol{v}_n)$ of $\mathbf{K}$; choose a number $l < N$. Set $\boldsymbol{\alpha}_j = \boldsymbol{v}_j / \sqrt{\lambda_j}, j = 1, \ldots, l$.

5. Let $\widehat{\boldsymbol{x}}_n = \left(\sum_{m=1}^{N} (\boldsymbol{\alpha}_j)_m k(\boldsymbol{x}_n, \boldsymbol{x}_m) \right)_{j=1}^{l}$. The result is the l-dimensional data sample $\widehat{X} = \{\widehat{\boldsymbol{x}}_n\}$ for $n = 1, ..., N$

Examples in real application domains

Text Visualization

Example: the famous Reuters news articles dataset

- All articles with no Topic annotations are dropped
- The text of each article is converted to lowercase, whitespace is normalized to single-spaces
- Only the first term from the Topic annotation list is retained (some articles have several topics assigned)
- The resulting dataset is a list of pairs (Topic, News content)
- We willl only use two topics for analysis: Crude Oil and Grain-related news
- The resulting dataset contains 875 news items on crude oil and grain
- The goal is to create a classifier for the news articles

Examples in real application domains

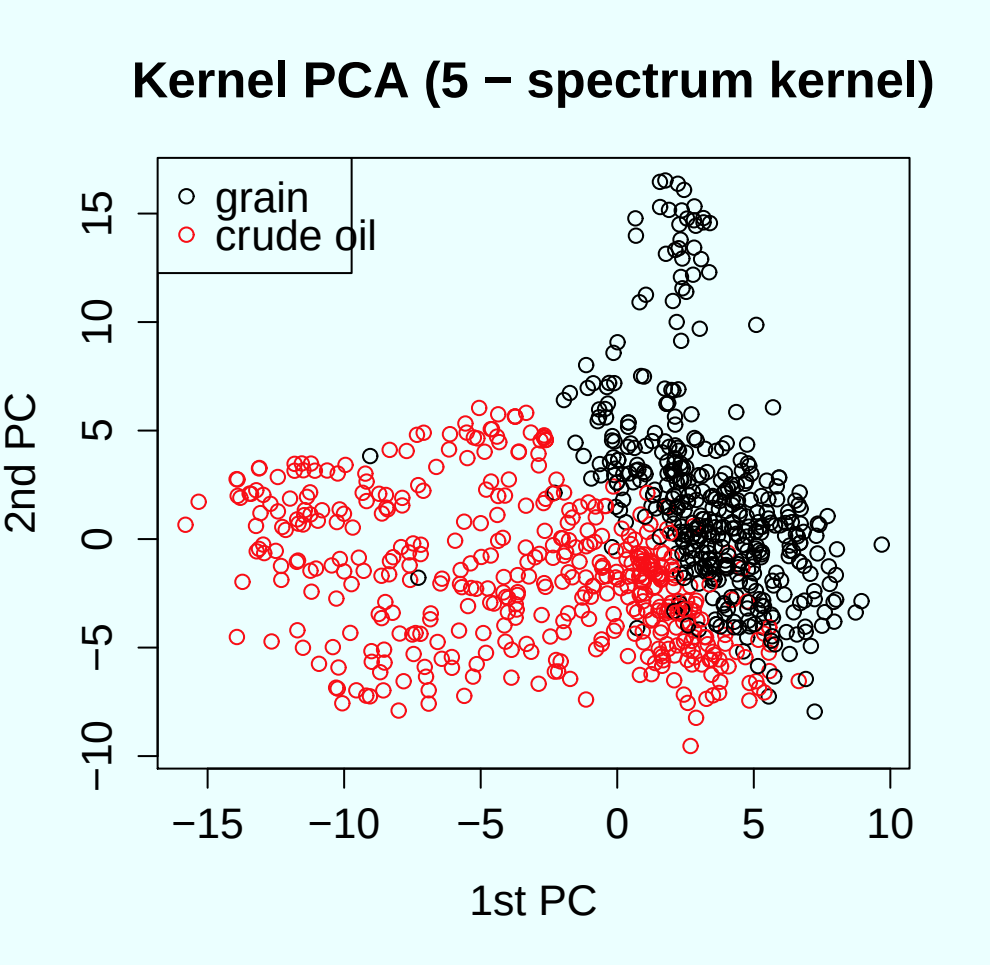
Text Visualization

- An example of a text about **grain**: "u.s. grain carloadings totaled 26,108 cars in the week ended february 21, down 2.2 pct from the previous week but 22.8 pct above the corresponding week a year ago, the association of american railroads reported. grain mill product loadings in the week totalled 11,382 cars, down 1.8 pct from the previous week but 7.6 pct above the same week a year earlier, the association said. reuter"

- An example of a text about **crude oil**: "diamond shamrock corp said that effective today it had cut its contract prices for crude oil by 1.50 dlrs a barrel. the reduction brings its posted price for west texas intermediate to 16.00 dlrs a barrel, the copany said. "the price reduction today was made in the light of falling oil product prices and a weak crude oil market,.[a] company spokeswoman said. diamond is the latest in a line of u.s. oil companies that have cut its contract, or posted, prices over the last two days citing weak oil markets. reuter"

Examples in real application domains

Text Visualization



Examples in real application domains

Handling missing values in microbiology

- Modern modelling problems are difficult for a number of reasons, including the challenge of dealing with a significant amount of missing information

- Missing values almost always represent a serious problem because they force to preprocess the dataset and a good deal of effort is normally put in this part of the modelling

- In order to process such datasets with kernel methods, an imputation procedure is then deeemed a necessary but demanding step.

Examples in real application domains

Handling missing values in microbiology

- The study of fecal source pollution in waterbodies is a major problem in ensuring the welfare of human populations

- **Microbial source tracking** methods attempt to identify the source of contamination, allowing for improved risk analysis and better water management

- The available dataset includes 148 observations about 10 chemical, microbial, and eukaryotic markers of fecal pollution in water

- All variables (except the class variable) are binary, i.e., they signal the presence or absence of a particular marker

Examples in real application domains

Handling missing values in microbiology

Origin	HF183	HF134	CF128	Humito	Pomito	Bomito	ADO	DEN
Human :50	0 :68	0 :81	0 :104	0 :35	0 :83	0 :78	0 :56	0 :80
Cow :26	1 :40	1 :26	1 : 5	1 :79	1 :32	1 :32	1 :59	1 :34
Poultry:31	$\mathcal{X}$:31	$\mathcal{X}$:32	$\mathcal{X}$:30	$\mathcal{X}$:25	$\mathcal{X}$:24	$\mathcal{X}$:29	$\mathcal{X}$:24	$\mathcal{X}$:25
Pig :32								

Summary (counts) table for the full dataset. The first column is the target class. The symbol $\mathcal{X}$ denotes a missing value.

The percentage of missing values is around 19.8 %, and all the predictive variables have percentages between 17 % and 23 %

Examples in real application domains

Handling missing values in microbiology

Theorem 4 *Let the symbol $\mathcal{X}$ denote a missing element, for which only equality is defined. Let $k : X \times X \to \mathbb{R}$ be a symmetric kernel in X and P a probability mass function (PMF) in X. Then the function $k^{\mathcal{X}}(x, y)$ given by*

$$k^{\mathcal{X}}(x,y) = \begin{cases} k(x,y), & \text{if } x, y \neq \mathcal{X} \ ; \\ g(x) = \sum\limits_{y' \in X} P(y')k(x,y'), & \text{if } x \neq \mathcal{X} \text{ and } y = \mathcal{X}; \\ g(y) = \sum\limits_{x' \in X} P(x')k(x',y), & \text{if } x = \mathcal{X} \text{ and } y \neq \mathcal{X}; \\ G = \sum\limits_{x' \in X} P(x') \sum\limits_{y' \in X} P(y')k(x',y'), & \text{if } x = y = \mathcal{X} \end{cases}$$

is a kernel in $X \cup \{\mathcal{X}\}$.

Examples in real application domains

Handling missing values in microbiology

For the particular case of binary variables $x, y \in \{v_1, v_2\}$, a convenient approach is to define the kernel:

$$k_{0/1}(x, y) = \mathbb{I}_{\{x=y\}} \tag{2}$$

where

$$\mathbb{I}_{\{z\}} = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{if } z \text{ is false} \end{cases}$$

Examples in real application domains

Handling missing values in microbiology

Consider now $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^d$. When we apply the Theorem to this kernel, we obtain an extended multivariate kernel:

$$\mathcal{K}_1(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{d}\sum_{i=1}^{d} \begin{cases} 1 & \text{if } x_i = y_i = 1 \ ; \\ P_i(x_i), & \text{if } x_i \neq \mathcal{X} \text{ and } y_i = \mathcal{X}; \\ P_i(y_i), & \text{if } x_i = \mathcal{X} \text{ and } y_i \neq \mathcal{X}; \\ (P_i(0))^2 + (P_i(1))^2, & \text{if } x_i = y_i = \mathcal{X}; \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

This kernel is a generalization of the classical *simple matching coefficient*, proposed by Sokal and Michener for numerical taxonomy

Examples in real application domains

Handling missing values in microbiology

Theorem 5 *Let the symbol $\mathcal{X}$ denote a missing element, for which only equality is defined. Let $k : X \times X \to \mathbb{R}$ be a symmetric kernel in $X = \{0, 1\}^d$. Let $c(\boldsymbol{x})$ be the set of completions of $\boldsymbol{x}$. Given two vectors $\boldsymbol{x}, \boldsymbol{y} \in X$, the function*

$$\mathcal{K}_2(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{|c(\boldsymbol{x})||c(\boldsymbol{y})|} \sum_{\boldsymbol{x}' \in c(\boldsymbol{x})} \sum_{\boldsymbol{y}' \in c(\boldsymbol{y})} k(\boldsymbol{x}', \boldsymbol{y}') \tag{4}$$

is a kernel in $X \cup \{\mathcal{X}\}$.

Examples in real application domains

Handling missing values in microbiology

			10x10cv for each class			
Approach	C	10x10cv	Human	Cow	Poultry	Swine
1KE	2.0	79.3	95.4	64.5	75.2	69.4
2KE	1.6	78.2	92.6	62.8	71.8	74.2
1MI	1.0	79.9	92.7	66.4	69.4	80.2
2MI	1.0	79.0	94.5	57.5	70.8	78.8

Mean 10x10cv accuracies for the four approaches to handle missing values. Also shown are best cost parameter C and detailed class performance.

(*joint work with G. Nebot, T. Aluja and V. Kobayashi*)

Examples in real application domains

Classification of DNA sequences

- DNA sequences of promoter and non-promoters.
- A *promoter* is a region of DNA that initiates or facilitates transcription of a particular gene.
- The dataset consists of 106 observations and 57 categorical variables describing the DNA sequence, represented as the nucleotide at each position: [A] adenine, [C] cytosine, [G] guanine, [T] thymine.
- The response is a binary class: "+" for a promoter gene and "−" for a non-promoter gene.

Examples in real application domains

Classification of DNA sequences

- A *categorical* variable takes discrete and unordered values, which are commonly known as *modalities*.

- Some symbolic values are naturally ordered; in many cases no order should be defined (the only meaningful relation is $=$ / $\neq$).

The basic similarity measure (which is a valid kernel) for these variables is the *overlap*. Let x_{ik}, x_{jk} be the modalities taken by two examples $\boldsymbol{x}_i, \boldsymbol{x}_j$, then $s(x_{ik}, x_{jk}) = \mathbb{I}_{\{x_{ik}=x_{jk}\}}$, where

$$\mathbb{I}_{\{z\}} = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{if } z \text{ is false} \end{cases}$$

Examples in real application domains

Classification of DNA sequences

The similarity between two multivariate categorical vectors is then proportional to the number of variables in which they match.

Definition 6 (Overlap/Dirac kernel)

$$k_0(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{d} \sum_{i=1}^{d} \mathbb{I}_{\{x_{ik}=x_{jk}\}}$$

Another kernel that can be used is the kernel:

Definition 7 (Gaussian Radial Basis Function kernel)

$$k_{RBF}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\gamma||\boldsymbol{x} - \boldsymbol{x}'||^2\right) \ , \gamma > 0$$

In order to use this kernel, categorical variables with m modalities are coded using a binary expansion representation.

Examples in real application domains

Classification of DNA sequences

Definition 8 (Univariate kernel $k_1^{(U)}$)

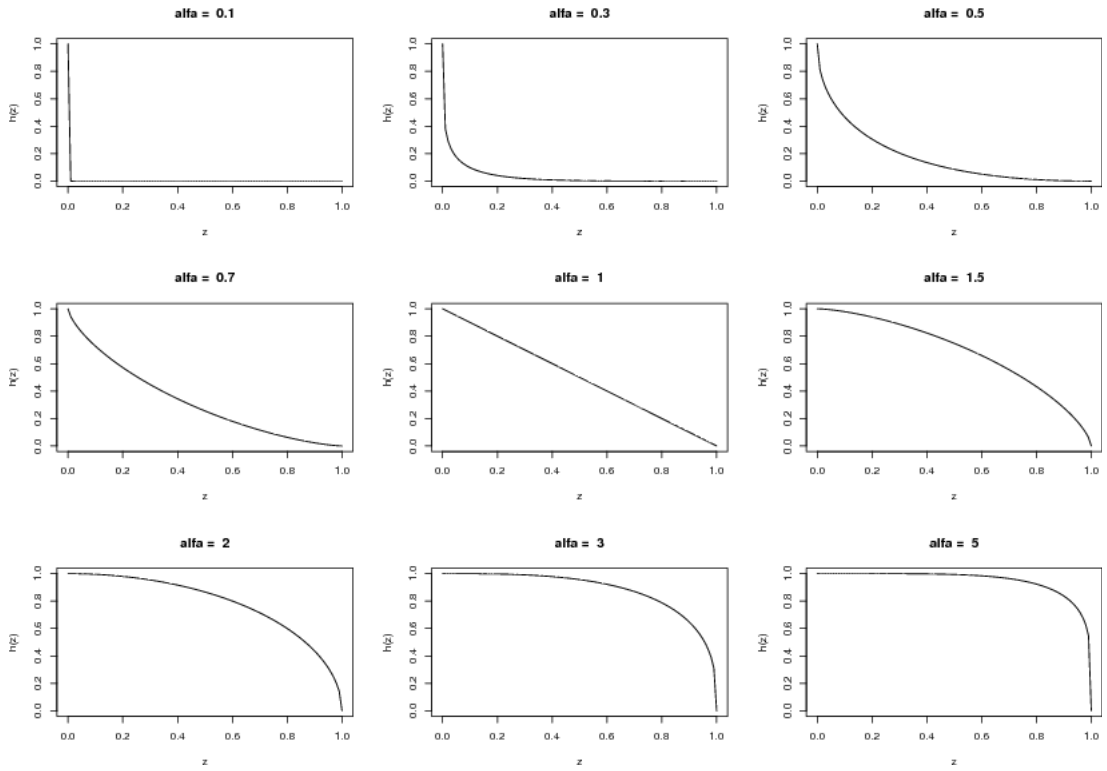
$$k_1^{(U)}(z_i, z_j) = \begin{cases} h_\alpha(P_Z(z_i)) & \text{if } z_i = z_j \\ 0 & \text{if } z_i \neq z_j \end{cases}$$

Definition 9 (Inverting function)

$$h_\alpha(z) = (1 - z^\alpha)^{1/\alpha},\ \alpha > 0$$

Examples in real application domains

Classification of DNA sequences



The family of inverting functions $h_\alpha(z)$ for different values of α.

Examples in real application domains

Classification of DNA sequences

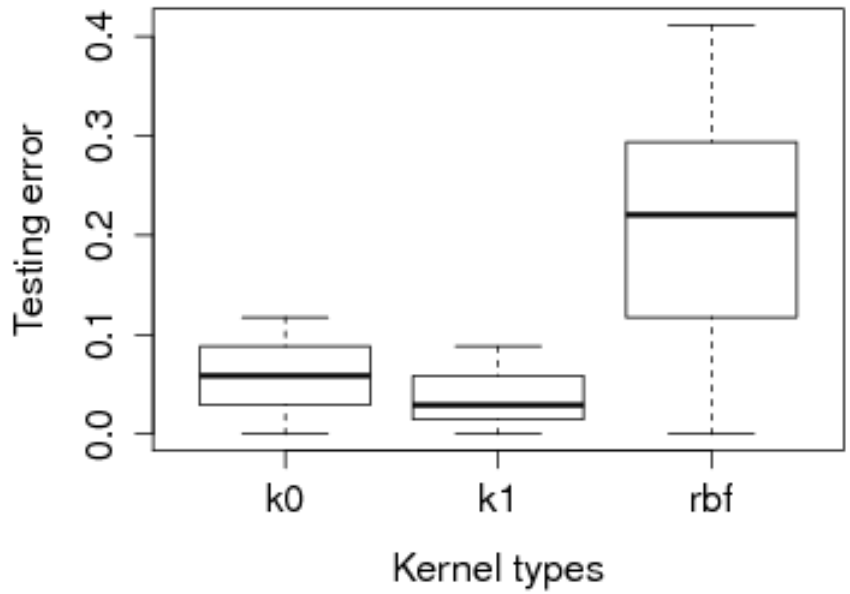
Definition 10 (Multivariate kernel k_1)

$$k_1(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{\gamma}{d}\sum_{i=1}^{d} k_1^{(U)}(x_{ik}, x_{jk})\right),\ \gamma > 0$$

Theorem 11 *The kernel matrix generated by the multivariate kernel k_1 is positive semi-definite (PSD).*

Examples in real application domains

Classification of DNA sequences



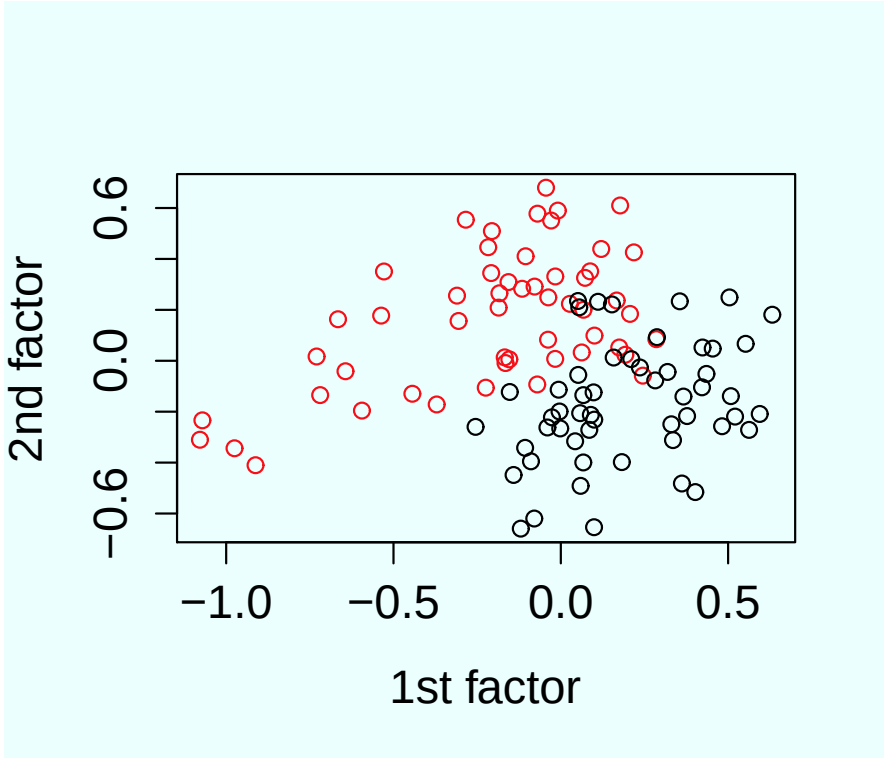
Test error distributions on the PromoterGene problem.

(*joint work with M. Villegas*)

Examples in real application domains

Classification of DNA sequences

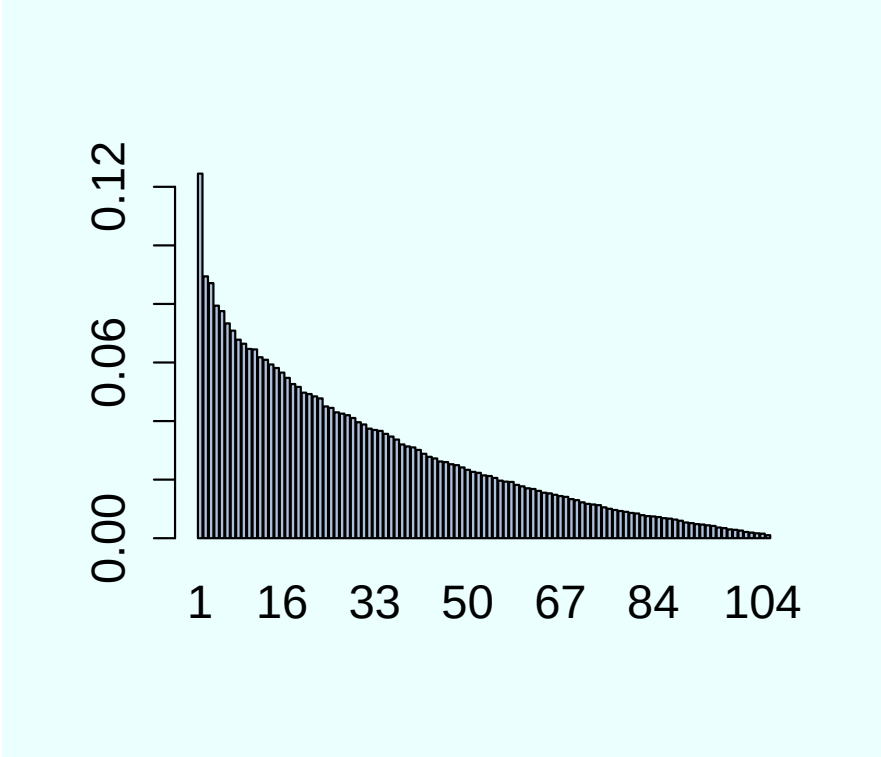
Want more ... ? Since the data is categorical, we perform a Multiple Correspondence Analysis (the analog of PCA for categorical data)



Projection of our data in the first two factors.

Examples in real application domains

Classification of DNA sequences



Histogram of eigenvalues.

Examples in real application domains

Classification of DNA sequences

- Choose the RBF kernel with automatic adjustment of the variance
- $C = 5$
- Gives a prediction (test) error of 14.3 %

A personal view

Pros of SVMs over Neural Networks

1. No local minima; few parameters to set
2. Automatic selection of position and number of hidden units
3. Capable of operating on any input space
4. Built-in regularization
5. Workable generalization bounds, via VC dimension theory
6. Better interpretability (to some extent)

A personal view

Cons of SVMs over Neural Networks

1. Hidden neurons must be centered at the training examples

2. Only one hidden layer

3. Needs similarities to be PSD

4. Some kernels are computationally costly

A personal view

Conclusions

1. Importance of designing kernels that do not constitute explicit inner products between objects, and therefore fully exploit the kernel trick

2. Possibility of learning the kernel function (or the kernel matrix) from the training data

3. Possibility of combining many different kernels and learn their coefficients from the training data (MKL)

4. Need theoretical analyses on the implications of the kernel choice for the success of kernel-based methods