Machine Learning for OR & FE

Kernel Methods (and the Kernel Trick)

Martin Haugh

Department of Industrial Engineering and Operations Research
Columbia University
Email: martin.b.haugh@gmail.com

Additional References: Christopher Bishop's *PRML*, Theodoridis' *Machine Learning: A Bayesian and Optimization Perspective*, Hastie and Efron's *Computer Age Statistical Inference*

Outline

The Separable Case

Two classes which (for now) are assumed to be linearly separable.

Training data $\mathbf{x}_1, \ldots, \mathbf{x}_n$ with corresponding targets, $t_1, \ldots, t_n$ with $t_i \in \{-1, 1\}$.

We consider a classification rule of the form of the form

$$\begin{aligned} h(\mathbf{x}) &= \text{sign}\left(\mathbf{w}^\top \mathbf{x} + b\right) \\ &= \text{sign}\left(y(\mathbf{x})\right) \end{aligned}$$

where $y(\mathbf{x}) := \mathbf{w}^\top \mathbf{x} + b$.

Can re-scale $(\mathbf{w}, b)$ without changing the decision boundary.

Therefore choose $(\mathbf{w}, b)$ so that training points closest to boundary satisfy $y(\mathbf{x}) = \pm 1$

- see left-hand component of Figure 7.1 from Bishop.

Let $\mathbf{x}_1$ be closest point from class with $t_1 = -1$ so that $\mathbf{w}^\top \mathbf{x}_1 + b = -1$.

And let $\mathbf{x}_2$ be closest point from class with $t_2 = 1$ so that $\mathbf{w}^\top \mathbf{x}_2 + b = 1$.

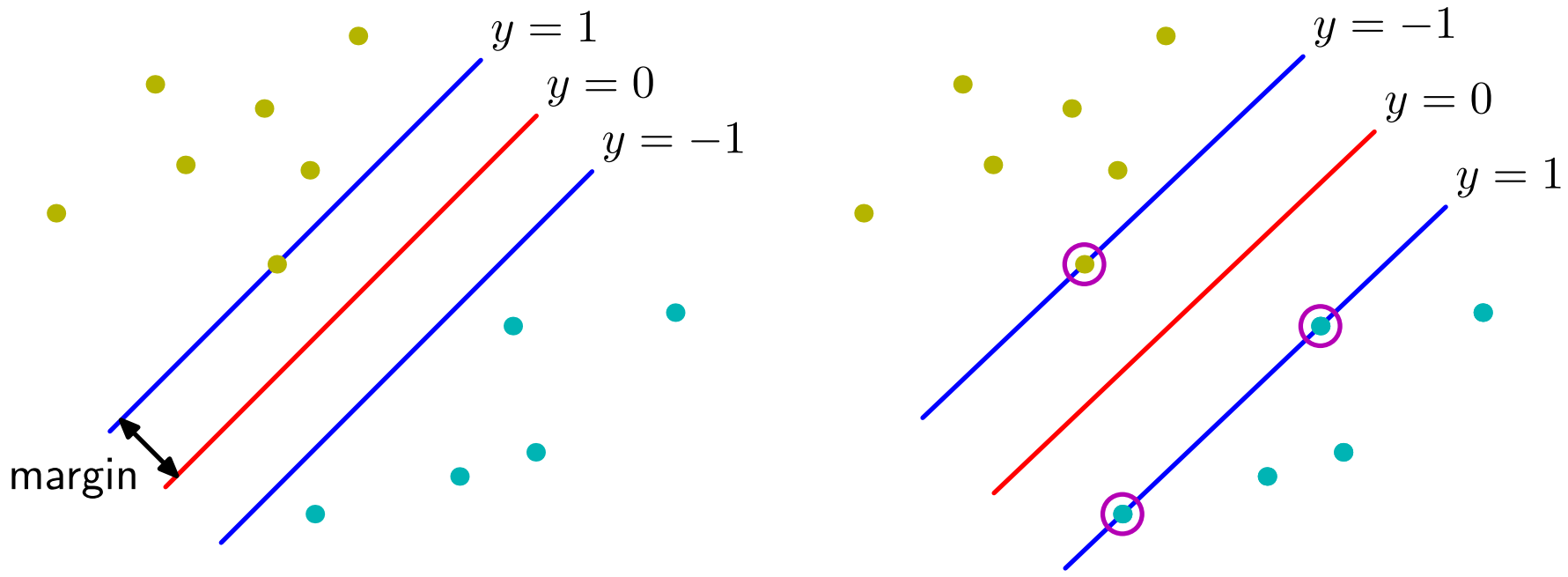


Figure 7.1 from Bishop: The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points, as shown on the left figure. Maximizing the margin leads to a particular choice of decision boundary, as shown on the right. The location of this boundary is determined by a subset of the data points, known as support vectors, which are indicated by the circles.

Geometry of Maximizing the Margin

Recall the perpendicular distance of a point $\mathbf{x}$ from the hyperplane, $\mathbf{w}^\top \mathbf{x} + b = 0$, is given by

$$|\mathbf{w}^\top \mathbf{x} + b| / ||\mathbf{w}||.$$

Therefore distance of closest points in each class to the classifier is $1/||\mathbf{w}||$.

An SVM seeks the maximum margin classifier that separates all the data

- seems like a good idea
- and can be justified by statistical learning theory.

Maximizing the margin, $1/||\mathbf{w}||$, is equivalent to minimizing $f(\mathbf{w}) := \frac{1}{2}\mathbf{w}^\top \mathbf{w}$.

Therefore obtain the following primal problem for the separable case:

$$\min_{\mathbf{w},b} \quad f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} \tag{1}$$

$$\text{subject to} \quad t_i \left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1, \quad i = 1, \ldots, n \tag{2}$$

Note that (2) ensures that all the training points are correctly classified.

The Primal Problem

The primal problem is a quadratic program with linear inequality constraints

- moreover it is convex and therefore has a unique minimum.

From the problem's geometry should be clear that only the points closest to the boundary are required to define the optimal hyperplane

- see right-hand component of Figure 7.1 from Bishop.
- these are called the support vectors
- and will see that the solution can be expressed using only these points.

The Non-Separable Case

In general the data will be non-separable so the primal problem of (1) and (2) will be infeasible.

Several ways to proceed: **e.g.** minimize the number of misclassified points, but this is NP-hard.

Instead we allow points to violate the margin constraints and penalize accordingly in the objective function. This yields the more general non-separable primal problem:

$$\min_{\mathbf{w},\boldsymbol{\xi},b} \ \frac{1}{2}\mathbf{w}^\top\mathbf{w} \ + \ C\sum_{i=1}^{n}\xi_i \tag{3}$$

$$\begin{aligned} \text{subject to} \quad t_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) &\geq 1-\xi_i, \quad i=1,\ldots,n \\ \xi_i &\geq 0, \quad i=1,\ldots,n \end{aligned} \tag{4}$$

- again a convex quadratic programming problem with linear constraints
- the penalty C usually chosen by cross-validation.

Aside: the Hinge Loss Function

The primal objective function of the SVM classifier may be written (why?) as

$$\begin{aligned} \text{Obj. Fun.} &= \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^{n}\xi_i \\ &\equiv \frac{1}{2C}||\mathbf{w}||^2 + \sum_{i=1}^{n} E_{sv}(t_i y_i) \end{aligned}$$

where

$$E_{sv}(t_i y_i) := [1 - t_i y_i]^+ \tag{5}$$

and where $y_i := y(\mathbf{x}_i)$.

The error function $E_{sv}(\cdot)$ is known as the hinge loss function.

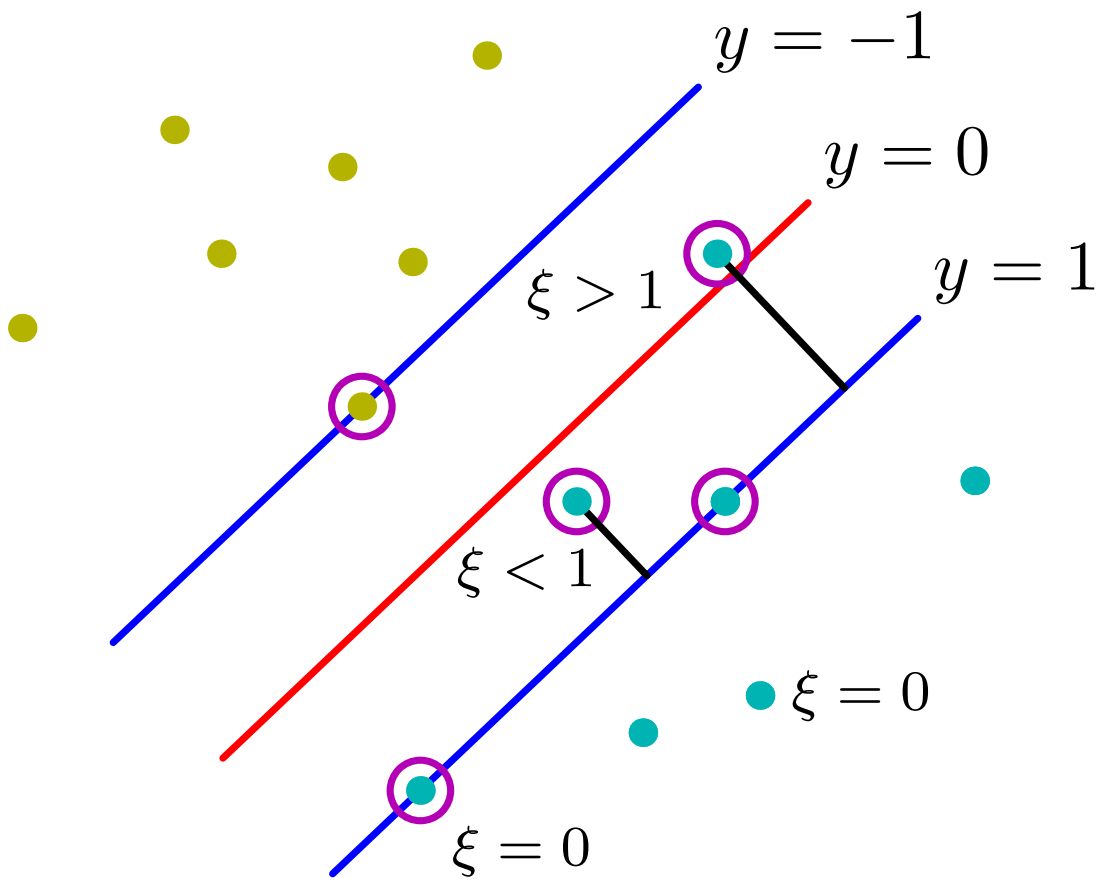


Figure 7.3 from Bishop: Illustration of the slack variables in $\xi_n \geq 0$. Data points with circles around them are support vectors.

Note that the slack variables allow points to be misclassified.

The Dual (of the Non-Separable Problem)

It's more convenient to work with the dual problem.

Because the primal problem is convex, the dual and primal have equal optimal objective functions.

Can be shown that the dual problem reduces to

$$\max_{\boldsymbol{\alpha}\geq 0,\, \boldsymbol{\lambda}\geq 0} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j \underbrace{\mathbf{x}_i^\top \mathbf{x}_j}_{\text{an inner product!}} \tag{6}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i t_i = 0$$

$$C - \alpha_i - \lambda_i = 0, \qquad i = 1, \ldots, n \tag{7}$$

where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)$ are Lagrange multipliers for the constraints (4)

- again a convex quadratic program with linear constraints
- the original dual plus the additional linear constraints of (7).

Can remove $\boldsymbol{\lambda}$ from the dual by replacing (7) with $\alpha_i \leq C$ for $i = 1, \ldots, n$.

The Kernel Trick

The kernel-trick very commonly used in regression, classification, PCA etc.

- Allows problem to be easily embedded in much higher dimensional spaces and often infinite dimensional spaces
- But without having to do an infinite amount of work.

Suppose instead of using $\mathbf{x} \in \mathbb{R}^m$ to describe the inputs we instead use a feature map, $\boldsymbol{\phi}(\mathbf{x})^\top \in \mathbb{R}^M$, often with $M >> m$.

Can now solve same dual problem as (6) except $\mathbf{x}_i^\top \mathbf{x}_j$ replaced with $\boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j)$.
Optimal b^* satisfies

$$b^* = t_j - \sum_{i=1}^{n} \alpha_i^* t_i \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j) \quad \text{for any } C > \alpha_j^* > 0 \tag{8}$$

and, for a new data-point $\mathbf{x}$, we predict

$$\text{sign}\left(\mathbf{w}^{*\top} \boldsymbol{\phi}(\mathbf{x}) + b^*\right) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i^* t_i \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}) + b^*\right). \tag{9}$$

The Gram Matrix

Define the Gram matrix, $\mathbf{K} = \boldsymbol{\phi}\boldsymbol{\phi}^\top$ to be the $n \times n$ matrix with

$$\mathbf{K}_{ij} := \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j) =: k(\mathbf{x}_i, \mathbf{x}_j) \tag{10}$$

For any set of points, $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the kernel matrix $\mathbf{K}$ is positive semi-definite so that $\mathbf{z}^\top \mathbf{K} \mathbf{z} \geq 0$ for all $\mathbf{z} \in \mathbb{R}^n$.

Definition. We say a function, $k(\mathbf{x}, \mathbf{x}')$, is a kernel if it corresponds to a scalar product, $\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ in some feature space, $\mathbb{R}^M$, possibly with $M = \infty$.

Theorem. A necessary and sufficient condition for a function, $k(\mathbf{x}, \mathbf{x}')$, to be a kernel is that the corresponding Gram matrix, $\mathbf{K}$, be positive semi-definite for all possible choices of $\mathbf{x}_1, \ldots, \mathbf{x}_n$.

Kernels

Key implication of theorem is possibility of implicitly defining a (possibly infinite-dimensional) feature map, $\boldsymbol{\phi}(\cdot)$, using a kernel function $k(\cdot, \cdot)$.

Note that $\boldsymbol{\phi}(\cdot)$ is not explicitly required to state the dual problem, nor is it required in (8) and (9)

- only $k(\cdot, \cdot)$ is required!
- a big advantage since far less work may be required to compute $k(\cdot, \cdot)$.

e.g. Let $m = 2$ and define $k(\mathbf{x}, \mathbf{x}') := (\mathbf{x}^\top \mathbf{x}')^2$. Easy to check that $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ where

$$\boldsymbol{\phi}(\mathbf{x}) := \left(x_1^2,\, \sqrt{2}x_1 x_2,\, x_2^2\right).$$

But calculating $k(\mathbf{x}, \mathbf{x}')$ requires $O(m)$ $(= \dim(\mathbf{x}))$ work whereas calculating $\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ requires $O(M) = O(m^2)$ work.

More generally, we could define $k(\mathbf{x}, \mathbf{x}') := (\mathbf{x}^\top \mathbf{x}' + c)^p$

- computing it will still be $O(m)$ but working with corresponding feature mapping will be $O(m^p)$.

Kernelizing the Dual

Can easily apply the kernel trick to obtain the following dual problem:

$$\max_{\boldsymbol{\alpha} \geq 0} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{11}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i t_i = 0 \tag{12}$$

$$\alpha_i \leq C, \quad i = 1, \ldots, n \tag{13}$$

Given an optimal solution, $\boldsymbol{\alpha}^*$, can recover the SVM classifier as:

$$b^* = t_j - \sum_{i=1}^{n} \alpha_i^* t_i k(\mathbf{x}_i, \mathbf{x}_j) \quad \text{for any } C > \alpha_j^* > 0$$

and, for a new data-point $\mathbf{x}$, the prediction

$$\text{sign}\left(\mathbf{w}^{*\top} \boldsymbol{\phi}(\mathbf{x}) + b^*\right) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i^* t_i k(\mathbf{x}_i, \mathbf{x}) + b^*\right). \tag{14}$$

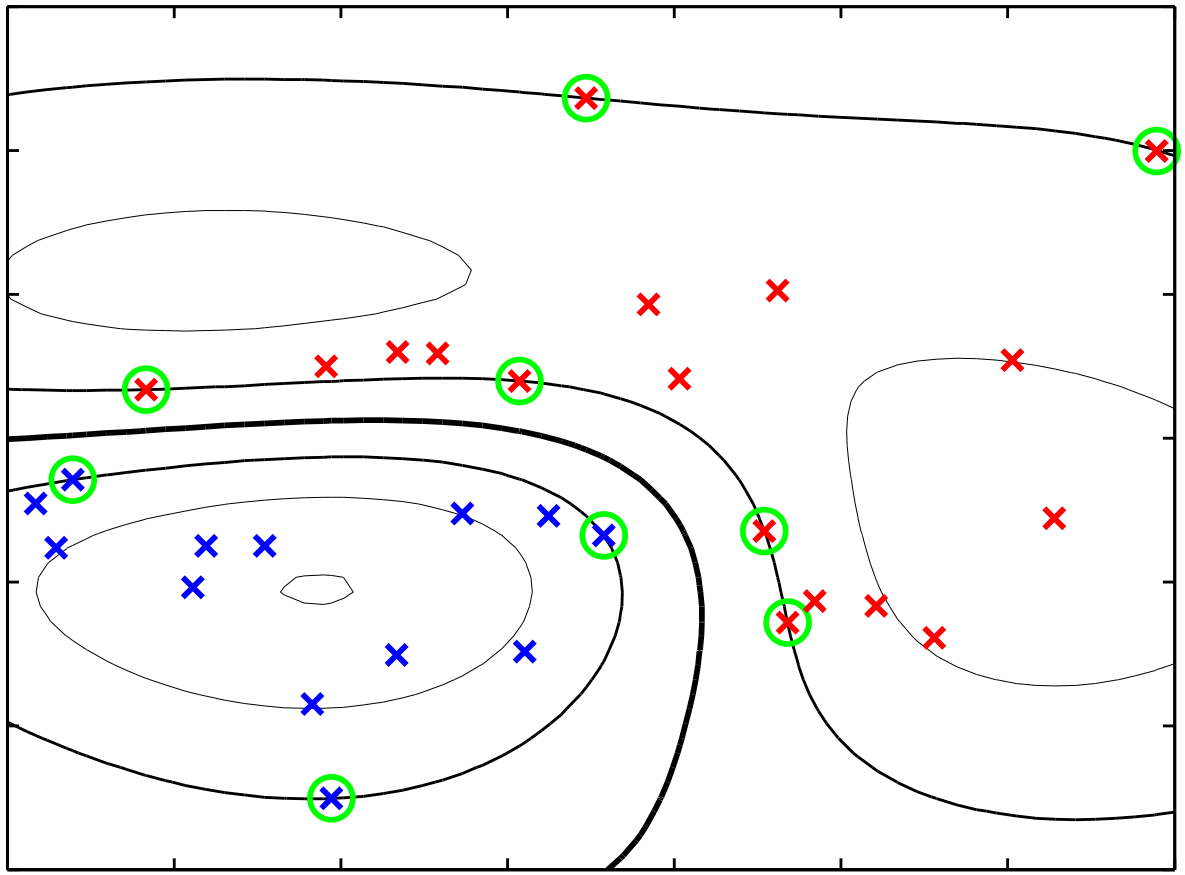
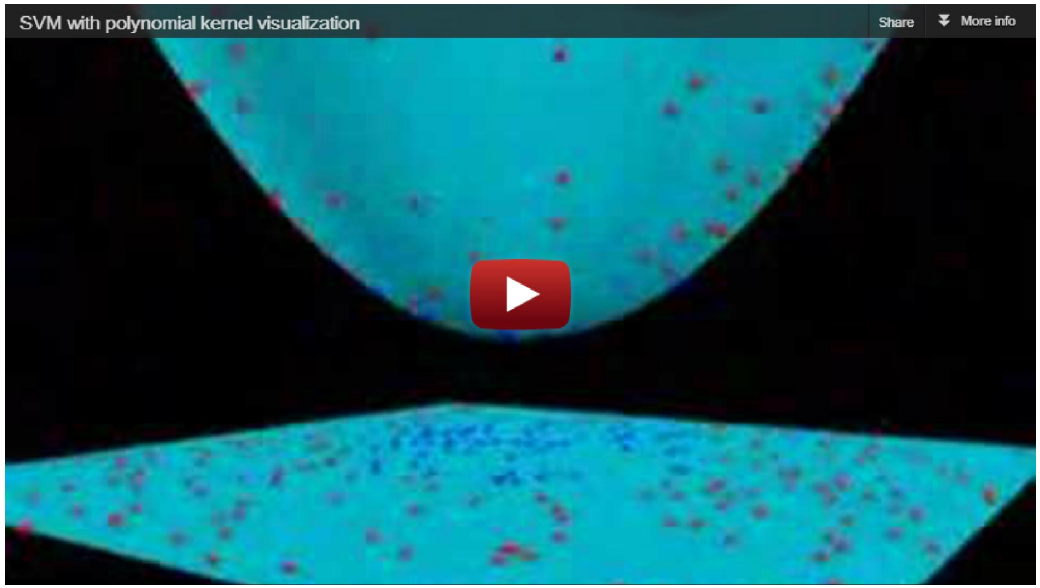


Figure 7.2 from Bishop: Example of synthetic data from two classes in two dimensions showing contours of constant $y(x)$ obtained from a support vector machine having a Gaussian kernel function. Also shown are the decision boundary, the margin boundaries, and the support vectors.

- It so happens here that the data is linearly separable in the Gaussian-kernel space but not in the original space. (In general the data will not be separable in either space.)



A Demo of SVM Classification with Polynomial Kernel by Udi Aharoni

Introduction to RKHS's*

Have seen kernel trick applied to SVMs but needed to work with dual problem.

Question: Can we apply the kernel trick to the primal problem, i.e. without having to formulate and work with the dual?

Question: And can we do this more generally, i.e. not just for SVMs?

Answer to both questions is "yes" but to see this must first introduce reproducing kernel Hilbert spaces (RKHS).

*This section on RKHS's draws from lecture notes by Lorenzo Rosasco.

The Regularization Problem

Goal of **regularization** is to:

1. Help control over-fitting the training data and thus ensure a better generalization error.
2. (Sometimes) ensure the learning optimization problem is well-posed.

If we use $\mathcal{H}$ to denote our hypothesis or function space then a general approach to regularization solves

$$\min_{f \in \mathcal{H}} \underbrace{\mathsf{Err}(f)}_{\text{empirical error}} + \lambda \underbrace{\mathsf{Pen}(f)}_{\text{penalty error}} \tag{15}$$

λ controls the tradeoff between the two terms and could be chosen via cross-validation.

The Regularization Problem

A more specific version of (15) is

$$\min_{f \in \mathcal{H}} \underbrace{\sum_{i=1}^{n} L(f(x_i), y_i)}_{\text{empirical error}} + \underbrace{\lambda \Omega \left(||f||^2_{\mathcal{H}} \right)}_{\text{penalty error}} \tag{16}$$

where:

- $n = \#$ of training data points.
- $||f||_{\mathcal{H}}$ denotes the norm of f in the function space $\mathcal{H}$.
- $L(f(x), y)$ is the loss function for predicting $f(x)$ when the true value is y.
- $\Omega : [0, +\infty) \to \mathbb{R}$ is an arbitrary strictly monotonic increasing function.

Note that a large number of standard supervised learning problems can be formulated as in (16).

Penalty term is used to control over-fitting and intuition tells us that it must therefore impose some form of **smoothing** on f.

Before proceeding, need to define some of the aforementioned terms ...

Some Functional Analysis

Let $\mathcal{F}$ be a function space whose elements are functions defined on X.

Definition. An inner product is a function $\langle \cdot, \cdot \rangle : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$ that satisfies for every $f, g \in \mathcal{F}$ and $\alpha_1, \alpha_2 \in \mathbb{R}$:

1. $\langle f, g \rangle = \langle g, f \rangle$. (symmetry)
2. $\langle \alpha_1 f_1 + \alpha_2 f_2, g \rangle = \alpha_1 \langle f_1, g \rangle + \alpha_2 \langle f_2, g \rangle$. (linearity)
3. $\langle f, f \rangle \geq 0$ for all $f \in \mathcal{F}$ and $\langle f, f \rangle = 0$ if and only if $f = 0$.

Definition. A norm is a non-negative function $|| \cdot || : \mathcal{F} \to \mathbb{R}$ such that for all $f, g \in \mathcal{F}$ and $\alpha \in \mathbb{R}$ we have:

1. $||f|| \geq 0$ and $||f|| = 0$ if and only if $f = 0$.
2. $||f + g|| \leq ||f|| + ||g||$.
3. $||\alpha f|| = |\alpha| \, ||f||$.

Given an inner product, $\langle \cdot, \cdot \rangle$, we can define a norm according to

$$||f|| := \sqrt{\langle f, f \rangle} \tag{17}$$

Hilbert Spaces

Definition. A complete metric space is a metric space where every Cauchy sequence converges.

Definition. A Hilbert space is a vector space equipped with an inner product $\langle \cdot, \cdot \rangle$. Moreover it is complete w.r.t. the norm defined by the inner product.

We will also only work with separable Hilbert spaces. By definition such a space has a countably dense subset so that is also has a countable orthonormal basis.

(Separable) Hilbert spaces are therefore (typically infinite-dimensional) generalizations of the usual Euclidean space $\mathbb{R}^d$

- very useful as we can apply mathematical tools of Euclidean space to them.

Examples of Function Spaces

e.g. Let $\mathcal{F}$ be the space, $C[a,b]$, of continuous function on the interval $[a,b]$ with the max norm

$$||f|| := \max_{a \leq x \leq b} |f(x)|$$

No inner product that induces this norm so $C[a,b]$ is not a Hilbert space.

e.g. Consider space, $L_2[a,b]$, of square-integrable functions on the interval $[a,b]$ with inner product defined by

$$\langle f, g \rangle := \int_a^b f(x)g(x)\,dx$$

with induced norm

$$||f|| = \int_a^b f(x)^2\,dx.$$

Can check that this is a complete space and therefore $L_2[a,b]$ is a Hilbert space.

The Problem with $L_2[a, b]$ as a Hypothesis Space

Let $f \in L_2[a, b]$ and consider a finite number of values $x_1, \ldots, x_n \in [a, b]$. Now define g as follows:

$$g(x) := \begin{cases} c, & x = x_i \text{ for any } i = 1, \ldots, n \\ f(x), & \text{otherwise.} \end{cases}$$

Then $g \in L_2[a, b]$ and $||f - g|| = 0$.

But this means taking $\mathcal{H} = L_2[a, b]$ in (16) will **not** work. Why?

And so any solution to (16) with $\mathcal{H} = L_2[a, b]$ (or $\mathcal{H} = L_2^d[a, b]$ more generally) will have **no predictive value**.

Must therefore refine our hypothesis space $\mathcal{H}$ so that we overcome this problem

- leads to reproducing kernel Hilbert spaces (RKHS's) – Hilbert spaces with the **reproducing kernel** property.

Reproducing Kernel Hilbert Spaces

Definition. An evaluation functional at the point $x \in X$ over the Hilbert space $\mathcal{H}$ is a linear functional $\mathcal{F}_x : \mathcal{H} \to \mathbb{R}$ that evaluates each function in $\mathcal{H}$ at the point x so that

$$\mathcal{F}_x[f] = f(x)$$

for all $f \in \mathcal{H}$.

Definition. A Hilbert space $\mathcal{H}$ is a reproducing kernel Hilbert space if the evaluation functionals are bounded, that is for all $x \in X$ there exists some $M > 0$ such that

$$|\mathcal{F}_x[f]| = |f(x)| \leq M||f||_{\mathcal{H}}$$

for all $f \in \mathcal{H}$.

Question: Explain why the evaluation functionals in the Hilbert space $L_2[a, b]$ are **not** bounded and so $L_2[a, b]$ is not a RKHS.

Can use the Riesz Representation Theorem to establish the following result.

Reproducing Kernel Hilbert Spaces

Theorem. If $\mathcal{H}$ is a RKHS then for each $x \in X$ there exists a function $k_x \in \mathcal{H}$ (called the **representer** of x) with the reproducing property

$$\mathcal{F}_x[f] = \langle k_x, f \rangle_{\mathcal{H}} = f(x) \tag{18}$$

for all $f \in \mathcal{H}$.

Since $k_{x'} \in \mathcal{H}$ for any $x' \in X$ we can take $f = k_{x'}$ in (18) to obtain

$$\begin{aligned} \langle k_x, k_{x'} \rangle_{\mathcal{H}} &= k_x(x') \\ &= k_{x'}(x). \quad \text{(Why?)} \end{aligned}$$

We now define the reproducing kernel of $\mathcal{H}$:

Definition. The reproducing kernel of the RKHS $\mathcal{H}$ is a function $k : X \times X \to \mathbb{R}$ defined by

$$k(x, x') := k_x(x'). \tag{19}$$

Reproducing Kernel Hilbert Spaces

We also have a more general definition of a reproducing kernel:

Definition. A function $k : X \times X \to \mathbb{R}$ is a reproducing kernel if it is symmetric, i.e. $k(x, x') = k(x', x)$ for all $x, x' \in X$, and positive-definite so that

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

for any $m \in \mathbb{N}$ and choice of $x'_1, \ldots, x'_m \in X$ and $c_1, \ldots, c_m \in \mathbb{R}$.

The two definitions are equivalent as established by the following theorem.

Theorem. A RKHS defines a corresponding reproducing kernel, i.e. a symmetric and positive-definite function $k(\cdot, \cdot)$. Conversely a reproducing kernel defines a unique RKHS.

The importance of this theorem is that it allows us to (implicitly) define a RKHS via a symmetric positive definite function k without having to derive k from the definition of the function space directly.

Proof of Theorem

A RKHS $\Rightarrow$ a Reproducing Kernel:

Define the function $k(\cdot, \cdot)$ as in (19). It is clearly symmetric and positive-definiteness follows because

$$\begin{aligned}
\sum_{i,j=1}^{m} c_i c_j k(x'_i, x'_j) &= \sum_{i,j=1}^{m} c_i c_j \langle k_{x'_i}, k_{x'_j} \rangle_{\mathcal{H}} \\
&= \sum_{i,j=1}^{m} \langle c_i k_{x'_i}, c_j k_{x'_j} \rangle_{\mathcal{H}} \\
&= \langle \sum_{i=1}^{m} c_i k_{x'_i}, \sum_{j=1}^{m} c_j k_{x'_j} \rangle_{\mathcal{H}} \\
&= || \sum_{i=1}^{m} c_i k_{x'_i} ||^2_{\mathcal{H}} \\
&\geq 0
\end{aligned}$$

as desired.

Proof of Theorem

A Reproducing Kernel $\Rightarrow$ a RKHS :

Consider the space $\mathcal{F}$ of functions **spanned** by $\{k_x \,|\, x \in X\}$ where $k(\cdot,\cdot)$ is the reproducing kernel (so $k(\cdot,\cdot)$ is symmetric and positive definite).

Let $f, g \in \mathcal{F}$ so there exists $s, r \in \mathbb{N}$, points $x_i', z_j' \in X$ and $\alpha_i, \beta_j \in \mathbb{R}$ s.t.

$$\begin{aligned} f(x) &= \sum_{i=1}^{s} \alpha_i k_{x_i'}(x) \\ g(x) &= \sum_{j=1}^{r} \beta_j k_{z_j'}(x). \end{aligned}$$

Define an inner product in this space according to

$$\langle f, g \rangle := \sum_{i=1}^{s} \sum_{j=1}^{r} \alpha_i \beta_j k(x_i', z_j'). \tag{20}$$

Must show inner product in (20) is indeed an inner product and then define $\mathcal{H} := \bar{\mathcal{F}}$, the completion of the function space $\mathcal{F}$. $\square$

Solving the Regularization Problem & Representer Theorem

Return now to original regularization problem:

$$\min_{f \in \mathcal{H}} \underbrace{\sum_{i=1}^{n} L(f(x_i), y_i)}_{\text{empirical error}} + \underbrace{\lambda \Omega\left(||f||_{\mathcal{H}}^2\right)}_{\text{penalty error}} \tag{21}$$

where $||f||_{\mathcal{H}}$ denotes the inner-product norm of f in the RKHS $\mathcal{H}$.

Note that typically we are given the reproducing kernel function $k(\cdot, \cdot)$ and don't bother to explicitly define $\mathcal{H}$ (which is the completion of the space spanned by $\{k_x \,|\, x \in X\}$ with inner product given by (20)).

Remark: Note that the x_i''s and z_j''s appearing in (20) are f- and g-dependent, respectively. They are **not** related to the data-points $x_1, \ldots, x_n$ of (21).

The problem in (21) is infinite-dimensional (in general) and so there may be computational issues in solving it.

But fortunately we have the representer theorem ...

The Representer Theorem

Theorem. Every solution $f^*(\cdot)$ to (21) admits a representation of the form

$$f^* = \sum_{j=1}^{n} c_j k_{x_j} \tag{22}$$

where $c_j \in \mathbb{R}$ for $j = 1, \ldots, n$.

The representer theorem is a remarkable result as it allows to reformulate the infinite dimensional problem of (21) to the finite dimensional problem of optimizing over $c_1, \ldots, c_n$.

(Sketch) Proof of the Representer Theorem

Consider the linear subspace of $\mathcal{H}$

$$\mathcal{H}_0 := \left\{ f \in \mathcal{H} \,:\, f = \sum_{j=1}^{n} c_j k_{x_j} \right\}$$

– the space spanned by the representers of the training data.

Let $\mathcal{H}_0^\perp$ be the linear subspace of $\mathcal{H}$ that is orthogonal to $\mathcal{H}_0$ so that

$$\mathcal{H}_0^\perp := \{ g \in \mathcal{H} \,:\, \langle g, f \rangle = 0 \text{ for all } f \in \mathcal{H}_0 \}\,.$$

Since $\mathcal{H}_0$ is finite-dimensional and hence closed we can write $\mathcal{H} = \mathcal{H}_0 \oplus \mathcal{H}_0^\perp$.

Any $f \in \mathcal{H}$ an therefore be uniquely decomposed as

$$f = f_0 + f_0^\perp$$

where $f_0 \in \mathcal{H}_0$ and $f_0^\perp \in \mathcal{H}_0^\perp$. Let $f_0 = \sum_{j=1}^{n} c_j k_{x_j}$ for some $c_1, \ldots, c_n$.

We now make two observations:

(Sketch) Proof of the Representer Theorem

1. By orthogonality we have

$$||f_0 + f_0^{\perp}||_{\mathcal{H}}^2 = ||f_0||_{\mathcal{H}}^2 + ||f_0^{\perp}||_{\mathcal{H}}^2$$

2. By the reproducing property and orthogonality we have

$$\begin{aligned} f(x_i) &= \langle f, k_{x_i} \rangle \\ &= \langle f_0 + f_0^{\perp}, k_{x_i} \rangle \\ &= \langle f_0, k_{x_i} \rangle \\ &= f_0(x_i). \end{aligned}$$

Can now rewrite the objective in (21) as

$$\sum_{i=1}^{n} L(f_0(x_i), y_i) + \lambda \Omega \left(||f_0||_{\mathcal{H}}^2 + ||f_0^{\perp}||_{\mathcal{H}}^2 \right) \tag{23}$$

and the result follows since Ω is strictly increasing by assumption. $\square$

The Representer Theorem

An optimal solution to (21) therefore takes the form given in (22) with corresponding objective function

$$\sum_{i=1}^{n} L\left(f(x_i), y_i\right) + \lambda\Omega\left(||f||_{\mathcal{H}}^2\right)$$

$$= \sum_{i=1}^{n} L\left(\sum_{j=1}^{n} c_j\, k(x_i, x_j), y_i\right) + \lambda\Omega\left(\sum_{i,j=1}^{n} c_i c_j\, k(x_i, x_j)\right)$$

$$= \sum_{i=1}^{n} L\left(\mathbf{c}^\top \mathbf{K}_{i\cdot}, y_i\right) + \lambda\Omega\left(\mathbf{c}^\top \mathbf{K}\mathbf{c}\right) \tag{24}$$

where $\mathbf{c} := (c_1 \ldots c_n)^\top$, $\mathbf{K}$ is the $n \times n$ Gram matrix with $(i,j)^{th}$ element $k(x_i, x_j)$ and $\mathbf{K}_{i\cdot}$ is the i^{th} row of $\mathbf{K}$.

Back to SVMs

Using the hinge loss formulation of SVMs in (5) and taking Ω to be the identity function, we can now use (24) to formulate a kernelized primal SVM:

$$\min_{\mathbf{c},b} \sum_{i=1}^{n} \left[1 - t_i(\mathbf{c}^\top \mathbf{K}_{i\cdot} + b)\right]^+ + \lambda \mathbf{c}^\top \mathbf{K} \mathbf{c} \tag{25}$$

Only difference between (25) and what (24) would imply is inclusion of offset b.

- Inclusion of b easily justified (and is a special case of the semiparametric representer theorem on next slide).
- Note b is not included in the regularization term in (25). Why?

Exercise: Can you relate the problem in (25) to our earlier primal formulation of the (non-separable) SVM problem with features $\phi(\mathbf{x}) = k_{\mathbf{x}}$?

Recently, the approach for very large problems has been to solve (25) directly using first-order methods and by smoothing the hinge loss function.

Remark: It should be clear from the RKHS formulation that the kernel trick does not work with $||\cdot||_1$ or Lasso-style regularization.

The Semiparametric Representer Theorem

Theorem. In addition to our earlier assumptions, assume also there exists a set of real-valued functions

$$\psi_m : X \to \mathbb{R}, \quad m = 1, \ldots, M$$

with the property that the $n \times M$ matrix with elements $\psi_m(x_n)$ has rank M. Then any solution $\tilde{f}$ to the learning problem

$$\min_{\substack{\tilde{f} = f + h,\, f \in \mathcal{H} \\ h \in \mathsf{Span}\{\psi_m,\, m = 1, \ldots, M\}}} \sum_{i=1}^{n} L(\tilde{f}(x_i), y_i) + \lambda \Omega\left(||f||_{\mathcal{H}}^2\right) \tag{26}$$

has the representation

$$\tilde{f} = \sum_{j=1}^{n} c_j k_{x_j} + \sum_{m=1}^{M} b_m \psi_m.$$

The semiparametric representer theorem can be very useful in applications where we know (typically from domain-specific knowledge) a good set of pre-selected basis functions, ψ_m for $m = 1, \ldots, M$, and we use the RKHS component f to smoothly fit that component of the output y that is not explained by the ψ_m's.

Ridge Regression

Recall that ridge regression solves

$$\hat{\boldsymbol{\beta}}^{\mathsf{R}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \frac{\lambda}{2} \sum_{j=1}^{p} \beta_j^2 \right\}$$

- shrinks regression coefficients towards 0 by imposing a penalty on their size
- λ a complexity parameter that controls the amount of shrinkage.

An equivalent formulation is

$$\hat{\boldsymbol{\beta}}^{\mathsf{R}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \right\} \tag{27}$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s$$

Standard (why?) to **scale and standardize** inputs before applying ridge regression.

Ridge Regression

Note β_0 is generally **not** shrunk so that procedure does not depend on origin chosen for Y.

To handle this and use matrix notation can split estimation into two steps:

1. Set $\hat{\beta}_0 = \bar{y} = \left(\sum_{i=1}^{n} y_i\right)/n$
2. Center the inputs so that $x_{ij} \to x_{ij} - \bar{x}_j$.
   Now estimate $\beta_1, \ldots, \beta_p$ using ridge regression without intercept and using the centered x_{ij}'s.

Dropping β_0 from $\boldsymbol{\beta}$, the ridge regression of step 2 therefore solves

$$\hat{\boldsymbol{\beta}}_{\mathsf{R}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \frac{\lambda}{2} \boldsymbol{\beta}^\top \boldsymbol{\beta} \right\}$$

which has solution

$$\hat{\boldsymbol{\beta}}_{\mathsf{R}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_m)^{-1} \mathbf{X}^\top \mathbf{y}. \tag{28}$$

Question: Can we apply the kernel trick to (28)?

Kernel Ridge Regression

Answer: Yes but a little work is required since (28) not (yet) in a form that only requires inner products. To fix this, we need the following matrix identity.

Matrix Identity: Let $\mathbf{P}$, $\mathbf{B}$ and $\mathbf{R}$ denote matrices of conformable sizes. Then

$$\left(\mathbf{P}^{-1} + \mathbf{B}^\top \mathbf{R}^{-1} \mathbf{B}\right)^{-1} \mathbf{B}^\top \mathbf{R}^{-1} = \mathbf{P}\mathbf{B}^\top \left(\mathbf{B}\mathbf{P}\mathbf{B}^\top + \mathbf{R}\right)^{-1}. \tag{29}$$

Take $\mathbf{P} = \lambda^{-1}\mathbf{I}_m$, $\mathbf{B} = \mathbf{X}$ and $\mathbf{R} = \mathbf{I}_n$ in (29) to obtain

$$\begin{aligned} \hat{\boldsymbol{\beta}}_{\mathsf{R}} &= \lambda^{-1}\mathbf{X}^\top \left(\lambda^{-1}\mathbf{X}\mathbf{X}^\top + \mathbf{I}_n\right)^{-1} \mathbf{y} \\ &= \mathbf{X}^\top \left(\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I}_n\right)^{-1} \mathbf{y}. \end{aligned} \tag{30}$$

Our prediction $\hat{y}(\mathbf{x}^{\mathsf{new}})$ at a new point, $\mathbf{x}^{\mathsf{new}}$, will therefore be

$$\hat{y}(\mathbf{x}^{\mathsf{new}}) = \hat{\boldsymbol{\beta}}_{\mathsf{R}}^\top \mathbf{x}^{\mathsf{new}} = \mathbf{y}^\top \left(\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I}_n\right)^{-1} \mathbf{X}\,\mathbf{x}^{\mathsf{new}} \tag{31}$$

Kernel Ridge Regression

Note that (31) is now amenable to the kernel trick as it only depends on $\mathbf{X}$ via inner products.

Therefore have the following more general version of (31)

$$\hat{\boldsymbol{\beta}}_{\mathsf{R}}^{\top}\mathbf{x}^{\mathsf{new}} \quad = \quad \mathbf{y}^{\top}\left(\mathbf{K}+\lambda\mathbf{I}_n\right)^{-1}\mathbf{k}(\mathbf{x}^{\mathsf{new}}) \tag{32}$$

where we recall $\mathbf{K} = \boldsymbol{\phi}\boldsymbol{\phi}^{\top}$ is the $n \times n$ Gram matrix with

$$\mathbf{K}_{ij} := \boldsymbol{\phi}(\mathbf{x}_i)^{\top}\boldsymbol{\phi}(\mathbf{x}_j) =: k(\mathbf{x}_i, \mathbf{x}_j)$$

and $\mathbf{k}(\mathbf{x}^{\mathsf{new}})$ is the $n \times 1$ vector with i^{th} element $k(\mathbf{x}_i, \mathbf{x}^{\mathsf{new}})$.

Exercise: Show that we can obtain (32) directly from (24) by defining the loss function L and the regularization function Ω appropriately.

Remark: The offset, β_0, can be handled as on Slide 37 or directly via the semi-parametric representer theorem.

Constructing New Kernels (Bishop)

We assume:

- $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$ are valid kernels.
- $c > 0$ is a constant.
- $f(\cdot)$ is any function.
- $q(\cdot)$ is a polynomial with nonnegative coefficients.
- $\boldsymbol{\phi}(\mathbf{x})$ is a function from $\mathbf{x}$ to $\mathbb{R}^M$.
- $k_3(\cdot, \cdot)$ is a valid kernel in $\mathbb{R}^M$.
- $\mathbf{A}$ is a symmetric positive semi-definite matrix.
- $\mathbf{x}_a$ and $\mathbf{x}_b$ are variables (not necessarily disjoint) with $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$.
- k_a and k_b are valid kernel functions over their respective spaces.

Constructing New Kernels (Bishop)

Then the following are all valid kernels:

$$
\begin{aligned}
k(x, x') &= c k_1(\mathbf{x}, \mathbf{x}') \\
k(x, x') &= f(\mathbf{x}) k_1(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') && (33) \\
k(x, x') &= q(k_1(\mathbf{x}, \mathbf{x}')) \\
k(x, x') &= \exp(k_1(\mathbf{x}, \mathbf{x}')) && (34) \\
k(x, x') &= k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \\
k(x, x') &= k_1(\mathbf{x}, \mathbf{x}') k_2(\mathbf{x}, \mathbf{x}') \\
k(x, x') &= k_3(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}')) \\
k(x, x') &= \mathbf{x}^\top \mathbf{A} \mathbf{x}' \\
k(x, x') &= k_a(\mathbf{x}_a, \mathbf{x}'_a) + k_b(\mathbf{x}_b, \mathbf{x}'_b) \\
k(x, x') &= k_a(\mathbf{x}_a, \mathbf{x}'_a) k_b(\mathbf{x}_b, \mathbf{x}'_b)
\end{aligned}
$$

The Gaussian kernel

The Gaussian kernel is given by:

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right) \tag{35}$$

It is a valid kernel because

$$\begin{aligned}\exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right) &= \exp\left(\frac{-\mathbf{x}^\top \mathbf{x}}{2\sigma^2}\right) \exp\left(\frac{\mathbf{x}^\top \mathbf{x}'}{\sigma^2}\right) \exp\left(\frac{-\mathbf{x}'^\top \mathbf{x}'}{2\sigma^2}\right) \\ &= f(\mathbf{x}) \exp\left(\frac{\mathbf{x}^\top \mathbf{x}'}{\sigma^2}\right) f(\mathbf{x}')\end{aligned}$$

and now we can apply (33) and (34).

Constructing Kernels for Other Objects

The kernel trick can be extended to inputs that are symbolic and not just vectors of real numbers.

Examples of such inputs are graphs, sets, strings, and text documents.

e.g. Consider a fixed set and define the space consisting of all possible subsets of this set. If A_1 and A_2 are two such subsets then let

$$k(A_1, A_2) := 2^{|A_1 \cap A_2|}$$

where $|A|$ denotes the number of subsets in A.

$k(\cdot, \cdot)$ is a valid kernel because it can be shown to correspond to an inner product in a feature space

- so we could easily use SVMs to classify these sets.

String Kernels

In many applications such as text mining and computational biology it's important to be able to quntify how similar two strings are.

e.g. Consider following two strings, x_1 and x_2, from Chap. 9 of Hastie and Efron:

IPTSALVKETLALLSTHRTLLIANETLRIPVPVHKNHQLCTEEIFQGIGTLESQTVQGGTV
ERLFKNLSLIKKYIDGQKKKCGEERRRVNQFLDYLQEFLGVMNTEWI

PHRRDLCSRSIWLARKIRSDLTALTESYVKHQGLWSELTEAERLQENLQAYRTFHVLLA
RLLEDQQVHFTPTEGDFHQAIHTLLLQVAAFAYQIEELMILLEYKIPRNEADGMLFEKK
LWGLKVLQELSQWTVRSIHDLRFISSHQTGIP

Each string represents a protein molecule with each character representing an amino acid which can be one of 20 different types.

Question: How similar are these strings?

To answer this, can treat each string x as a document consisting of letters from an alphabet of size 20.

Will define a feature vector $h^m(x)$ to consist of counts of all m-grams in the protein of length m.

String Kernels

e.g. If $m = 3$ then $20^3 = 8000$ possible m-grams so $h^3(x)$ a 8000×1 vector.

Let $h^m_{\text{STR}}(x) = \#$ times m-gram "STR" appears in x. Then have

$$\begin{aligned} h^3_{\text{LQE}}(x_1) &= 1 \\ h^3_{\text{LQE}}(x_2) &= 2. \end{aligned}$$

Often want to take m quite large in which case 20^m will be very large and then calculating inner products $\mathbf{K}_m(x_1, x_2) := \langle h^m(x_1), h^m(x_2) \rangle$ very expensive.

But $h^m(x)$ generally very sparse and calculating kernel $\mathbf{K}_m(x_1, x_2)$ can be done efficiently using tree methods without ever having to calculate feature vectors. □

More generally, let $\mathcal{S}^*$ denote set of all possible strings that can be constructed using symbols from a dictionary / alphabet $\mathcal{S}$.

Given strings $x, y \in \mathcal{S}^*$ can define the kernel

$$k(x, y) := \sum_{s \in \mathcal{S}^*} w_s h_s(x) h_s(y)$$

where $w_s \geq 0$ and $h_s(x) = \#$ times substring s appears in x.

Appendix: Mercer's Theorem

Mercer's Theorem provides another approach to developing theory of RKHS's.

Suppose again that we have a symmetric positive semi-definite (and continuous) kernel k as defined on slide 26.

Ignoring technical details, $k(x, z)$ can be represented via an orthornormal eigen expansion

$$k(x, z) = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(z) \tag{36}$$

with each $\lambda_i \geq 0$, $\sum_{i=1}^{\infty} \lambda_i < \infty$ and where the convergence in (36) is uniform and absolute so that

$$\lim_{n \to \infty} \sup_{x,z} |k(x, z) - \sum_{i=1}^{n} \lambda_i \phi_i(x) \phi_i(z)| = 0.$$

Appendix: Mercer's Theorem

Now define $\mathcal{H}_K$ to be the set of functions f of the form

$$f(x) = \sum_{i=1}^{\infty} c_i \phi_i(x).$$

with

$$||f||^2_{\mathcal{H}_K} := \sum_{i=1}^{\infty} \frac{c_i^2}{\lambda_i} < \infty. \tag{37}$$

Corresponding inner product is

$$\langle f, g \rangle := \sum_{i=1}^{\infty} (b_i c_i)/\lambda_i$$

where $g(x) := \sum_{i=1}^{\infty} b_i \phi_i(x)$.

Clear from (37) that c_i's must go to zero quickly if $f \in \mathcal{H}_K$. Why?

- so dividing by λ_i's in (37) amounts to imposing a smoothness condition on the our space $\mathcal{H}_K$.

Appendix: Mercer's Theorem

So the penalty term $\Omega\left(||f||^2_{\mathcal{H}}\right)$ in (21) acts like a roughness penalty as it penalizes functions with large coefficients on the eigen functions with small corresponding eigen values.

We can also show the **representer** property since

$$\begin{aligned}
\langle f(\cdot), k(\cdot, z)\rangle &= \langle \sum_{i=1}^{\infty} c_i \phi_i(\cdot), \sum_{i=1}^{\infty} \lambda_i \phi_i(z)\phi_i(\cdot)\rangle \\
&= \sum_{i=1}^{\infty} \frac{c_i \lambda_i \phi_i(z)}{\lambda_i} \\
&= \sum_{i=1}^{\infty} c_i \phi_i(z) \\
&= f(z)
\end{aligned}$$

as desired.