Lec 7. Kernel Methods

1 Motivation: Why Linear Models Are Not Enough

We begin with linear models of the form

$$f(x) = w^\top x,$$

which form the foundation of regression and classification methods such as linear regression, logistic regression, and linear discriminant analysis.

Linear models are attractive because they are simple, interpretable, and lead to convex optimization problems with well-understood theoretical properties. However, they suffer from an important limitation: the decision boundary is restricted to be linear in the input space.

In many real-world problems, the relationship between predictors and response is inherently nonlinear. In such cases, linear models can only perform well if the data are represented using an appropriate set of features.

1.1 Naive Fix: Feature Expansion

A natural idea is to introduce nonlinear features explicitly. For example, one might map

$$x \;\mapsto\; \phi(x) = (x_1, x_2, x_1^2, x_1x_2, x_2^2, \dots).$$

A linear model applied to $\phi(x)$ can represent nonlinear relationships in the original input space.

While this approach is conceptually simple, it has several drawbacks:

- The dimensionality of $\phi(x)$ grows rapidly.
- Explicit feature construction is ad hoc and problem-dependent.
- High-dimensional feature spaces increase the risk of overfitting.
- Computation and storage become expensive.

This motivates the following question:

Can we obtain nonlinear models without explicitly constructing nonlinear features?

2 Core Idea: Similarity Instead of Features

Rather than focusing on explicit feature representations, kernel methods take a geometric perspective. Instead of asking "**What features should we construct?**", we ask "**How similar are two inputs?**" This shift replaces explicit feature engineering with a notion of similarity that implicitly defines the geometry of the feature space.

2.1 Inner Products, Geometry, and Hilbert Spaces

An *inner product* is a function $\langle \cdot, \cdot \rangle$ that maps two vectors to a scalar. In $\mathbb{R}^p$, the standard inner product is the dot product,

$$\langle u, v \rangle = u^\top v.$$

Inner products induce a geometric structure on a vector space. In particular, they define:

- a norm: $\|u\| = \sqrt{\langle u, u \rangle}$,
- angles and similarity: $\cos \angle(u, v) = \frac{\langle u, v \rangle}{\|u\| \, \|v\|}$,
- distances: $\|u - v\|^2 = \langle u, u \rangle + \langle v, v \rangle - 2\langle u, v \rangle$.

Many learning algorithms depend on the data only through inner products. As a result, changing the inner product changes the geometry in which learning takes place.

A *Hilbert space* is a vector space equipped with an inner product such that the space is *complete* with respect to the norm induced by the inner product.

Completeness means that every Cauchy sequence in the space converges to an element within the space. This property is essential for optimization problems, as it guarantees that minimizers of well-posed objectives actually exist.

The notion of an inner product is not restricted to finite-dimensional vector spaces. Importantly, Hilbert spaces may be infinite-dimensional. For example, the space $L^2([0, 1])$ of square-integrable functions,

$$L^2([0, 1]) = \left\{ f : \int_0^1 f(x)^2 \, dx < \infty \right\},$$

equipped with the inner product

$$\langle f, g \rangle = \int_0^1 f(x) g(x) \, dx,$$

is an infinite-dimensional Hilbert space.

Thus, Hilbert spaces allow us to extend geometric concepts such as length, angles, and projections from finite-dimensional vectors to functions. This generalization is fundamental to kernel methods, which operate by mapping data into (possibly infinite-dimensional) Hilbert spaces and performing linear learning in that space.

2.2 Kernel Functions and the Kernel Trick

A *kernel function* is defined as

$$K(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}},$$

where ϕ is a feature map into a (possibly high- or infinite-dimensional) Hilbert space $\mathcal{H}$.

The key idea is that we do not need to compute $\phi(x)$ explicitly. If we can evaluate $K(x, x')$ directly, we can compute inner products in $\mathcal{H}$ without ever constructing the feature vectors themselves. This idea is known as the *kernel trick*.

Through the kernel trick, nonlinear structure is introduced implicitly while maintaining the computational structure of linear algorithms.

2.3 Examples of Kernel Functions

We now present several commonly used kernel functions.

2.3.1 Linear Kernel

The linear kernel is defined as

$$K(x, x') = x^\top x'.$$

This kernel corresponds to the identity feature map $\phi(x) = x$. The linear kernel serves as an important baseline and shows that kernel methods generalize classical linear models.

2.3.2 Polynomial Kernel

The polynomial kernel of degree d is defined as

$$K(x, x') = (x^\top x' + c)^d, \qquad c \geq 0.$$

This kernel implicitly corresponds to a feature map that includes all monomials of degree up to d. For example, when $d = 2$, the feature space contains quadratic terms such as $x_i x_j$.

Polynomial kernels allow models to capture interactions between features, but their expressiveness grows rapidly with d, which can lead to overfitting.

2.3.3 Gaussian (Radial Basis Function) Kernel

The Gaussian or radial basis function (RBF) kernel is defined as

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2} \right), \qquad \sigma > 0.$$

This kernel measures similarity based on Euclidean distance: points that are close together have large kernel values, while distant points have kernel values close to zero.

The Gaussian kernel corresponds to an infinite-dimensional feature space. The parameter σ controls the locality of the kernel:

- Small σ leads to highly localized, flexible functions.
- Large σ leads to smoother, more global functions.

2.3.4 Radial Basis Function Kernels

More generally, a radial basis function kernel takes the form

$$K(x, x') = \psi(\|x - x'\|),$$

where ψ is a nonnegative, decreasing function of the distance between x and x'. The Gaussian kernel is the most widely used example of this class.

Radial kernels emphasize locality and are particularly effective when the target function varies smoothly with the input.

2.3.5 Sigmoid Kernel

The sigmoid kernel is defined as

$$K(x, x') = \tanh(\kappa\, x^\top x' + c),$$

for parameters $\kappa > 0$ and c.

This kernel is inspired by neural networks and resembles the activation function of a single hidden layer perceptron. However, it does not satisfy the positive semidefiniteness condition for all parameter choices and must be used with care.

3 What Is a Kernel Method?

A learning algorithm is called a *kernel method* if it depends on the data only through inner products, which can be replaced by kernel evaluations. That is, wherever the algorithm uses

$$x_i^\top x_j,$$

we may substitute

$$K(x_i, x_j).$$

This substitution yields nonlinear learning algorithms whose optimization structure remains unchanged. The nonlinearity arises entirely from the geometry induced by the kernel.

3.1 Kernel Methods as Function-Space Regularization

Kernel methods can be viewed as regularized learning problems in a function space. Specifically, they solve optimization problems of the form

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

where $\mathcal{H}_K$ is the reproducing kernel Hilbert space (RKHS) associated with the kernel K, and $\|\cdot\|_{\mathcal{H}_K}$ is the RKHS norm.

This formulation generalizes ridge regression:

$$\min_{w} \sum_{i} L(y_i, w^\top x_i) + \lambda \|w\|_2^2,$$

by replacing parameter vectors with functions and Euclidean norms with RKHS norms.

3.2 The RKHS Norm

For functions of the form

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x),$$

the RKHS norm is given by

$$\|f\|_{\mathcal{H}_K}^2 = \sum_{i,j} \alpha_i \alpha_j K(x_i, x_j).$$

The RKHS norm measures the complexity of a function relative to the kernel. A small RKHS norm corresponds to a smooth, simple function, while a large norm corresponds to a more complex or highly varying function. Importantly, the notion of complexity depends on the choice of kernel.

4 Support Vector Machines as a Canonical Kernel Method

Support Vector Machines (SVMs) provide the most prominent example of kernel methods. In the linear case, SVMs solve

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{subject to } y_i(w^\top x_i + b) \geq 1.$$

The dual formulation of the SVM depends on the data only through inner products:

$$\max_{\alpha} \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^\top x_j.$$

Replacing $x_i^\top x_j$ with $K(x_i, x_j)$ yields a nonlinear classifier that is linear in feature space. The resulting decision boundary is nonlinear in the input space but corresponds to a linear separator in the RKHS.

Maximizing the margin in SVMs is equivalent to minimizing the RKHS norm of the decision function. Thus, SVMs implement function-space regularization in a geometrically interpretable way.

5 Other Kernel Methods

Beyond SVMs, several other learning algorithms fall within the kernel method framework:

- **Kernel Ridge Regression**: regression with squared loss and RKHS regularization.
- **Gaussian Processes**: a Bayesian interpretation of kernel-based function learning, where the kernel defines the covariance structure.
- **Kernel PCA**: an unsupervised kernel method for nonlinear dimensionality reduction.

5.1 Ridge Regression and Kernel Ridge Regression

We first recall ridge regression in the standard linear setting. Given training data $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^p$, ridge regression solves

$$\min_{w \in \mathbb{R}^p} \sum_{i=1}^n (y_i - w^\top x_i)^2 \; + \; \lambda \|w\|_2^2.$$

The ℓ_2 penalty controls the magnitude of the parameter vector w and prevents overfitting by discouraging overly complex linear models.

Kernel ridge regression generalizes this idea to nonlinear function spaces. Let $\mathcal{H}_K$ be the RKHS associated with kernel K. Kernel ridge regression solves

$$\min_{f \in \mathcal{H}_K} \sum_{i=1}^n (y_i - f(x_i))^2 \; + \; \lambda \|f\|_{\mathcal{H}_K}^2.$$

Here, the Euclidean norm $\|w\|_2$ is replaced by the RKHS norm $\|f\|_{\mathcal{H}_K}$, which measures the complexity of the function f relative to the kernel. When the kernel is linear, kernel ridge regression reduces exactly to ridge regression.

5.2 Principal Component Analysis and Kernel PCA

Principal Component Analysis (PCA) seeks directions in $\mathbb{R}^p$ along which the projected data have maximum variance. Given centered data $x_1, \ldots, x_n$, PCA solves

$$\max_{w \in \mathbb{R}^p} \sum_{i=1}^n (w^\top x_i)^2 \quad \text{subject to} \quad \|w\|_2^2 = 1.$$

The constraint $\|w\|_2^2 = 1$ prevents trivial solutions and ensures that the direction w captures meaningful variance.

Kernel PCA extends PCA to nonlinear feature spaces. Let $\phi(x)$ denote a feature map into an RKHS $\mathcal{H}_K$. Kernel PCA solves

$$\max_{f \in \mathcal{H}_K} \sum_{i=1}^{n} f(x_i)^2 \quad \text{subject to} \quad \|f\|_{\mathcal{H}_K}^2 = 1.$$

Thus, kernel PCA performs PCA in the feature space induced by the kernel. The unit RKHS norm constraint ensures that the extracted components are not arbitrarily scaled. When the kernel is linear, kernel PCA reduces to ordinary PCA.

6 Summary

Kernel methods provide a principled way to introduce nonlinearity into learning algorithms while retaining convex optimization and strong theoretical guarantees. They replace explicit feature construction with similarity and replace parameter regularization with function-space regularization. Support Vector Machines are the canonical example of this framework, but many other learning algorithms can be understood through the same lens.