Lecture 6: Kernel Methods

Readings: Bach (Ch. 7), ESL (Ch. 5.8); code

Soon Hoe Lim

September 11, 2025

Outline

Connecting the Dots: The Big Picture

We've encountered several powerful, but seemingly distinct, ideas:

- In our study of SVMs, we saw a clever **"kernel trick"** that created complex non-linear boundaries by implicitly mapping data to a new feature space.
- In our lecture on non-linear modeling, we saw that **smoothing splines** find a flexible function by minimizing a combined "loss + smoothness penalty".
- Both the ridge regression and SVC also use a "loss + regularization penalty" to control complexity.

This raises some deep questions:

- Is the "kernel trick" a one-off gimmick just for SVMs, or is it a more general recipe?
- Is there a formal connection between a penalty on weights like $\|\beta\|^2$ and a penalty on a function's "wiggliness"?
- Can we build a **single, unified theory** that explains all of these?

In this lecture, we will reveal the beautiful and powerful theory of **kernel methods** that connects all these dots.

Not All Functions Are Valid Kernels

First, let's recall what kernels are. Given a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, when does there exist a feature mapping $\phi : \mathcal{X} \to \mathcal{H}$ such that $K(x, x') = \langle \phi(x), \phi(x') \rangle$?

Definition 1: Positive Semi-Definite (PSD) Kernel

A function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a **positive semi-definite kernel** if:

1. **Symmetry:** $K(x, x') = K(x', x)$ for all $x, x' \in \mathcal{X}$.
2. **Positive Semi-Definiteness:** For any finite set $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the Gram matrix $\mathbf{K}$ with entries $\mathbf{K}_{ij} = K(x_i, x_j)$ satisfies:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j K(x_i, x_j) \geq 0 \quad \text{for all } c_1, \ldots, c_n \in \mathbb{R}.$$

Equivalently: All eigenvalues of every Gram matrix $\mathbf{K}$ must be non-negative.

The Fundamental Kernel Theorem

Theorem 1: Mercer's Theorem (Finite Version)

A function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive semi-definite kernel if and only if there exists a Hilbert space $\mathcal{H}$ and a mapping $\phi : \mathcal{X} \to \mathcal{H}$ such that:

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

The PSD condition is both necessary and sufficient. It guarantees that the kernel trick is mathematically valid.

Proof.
See blackboard. $\square$

Kernel Verification: Examples

Verifying Polynomial Kernels

Claim: $K(x, x') = (x^T x')^d$ is PSD for any $d \geq 1$.
Proof: For any $c_1, \ldots, c_n$ and $x_1, \ldots, x_n$:

$$\sum_{i,j} c_i c_j K(x_i, x_j) = \sum_{i,j} c_i c_j (x_i^T x_j)^d \tag{1}$$

$$= \left(\sum_i c_i x_i^T \right)^d \left(\sum_j c_j x_j \right)^d \geq 0 \tag{2}$$

since it's a $2d$-th power of a real number.

Verifying Gaussian Kernels

Claim: $K(x, x') = \exp(-\|x - x'\|^2/(2\sigma^2))$ is PSD.
Proof idea:

- Expand: $\exp(-\|x - x'\|^2/(2\sigma^2)) = \exp(-\|x\|^2/(2\sigma^2)) \exp(x^T x'/\sigma^2) \exp(-\|x'\|^2/(2\sigma^2))$
- The middle term has Taylor series: $\exp(x^T x'/\sigma^2) = \sum_{k=0}^{\infty} \frac{(x^T x')^k}{\sigma^{2k} k!}$
- Each term $(x^T x')^k$ is PSD, positive coefficients preserve PSD property

Note: A more general proof for continuous, translation-invariant kernels relies on Bochner's Theorem, which states that such a function is PSD if and only if its Fourier transform is a non-negative measure (see Bach Ch. 7.3.3).

Operations That Preserve Kernels

Proposition 1: Kernel Closure Properties

If K_1, K_2 are PSD kernels, then the following are also PSD kernels:

1. **Positive scaling:** αK_1 for any $\alpha > 0$
2. **Sum:** $K_1 + K_2$
3. **Product:** $K_1 \cdot K_2$
4. **Pointwise limit:** $\lim_{n\to\infty} K_n$ (if it exists and is continuous)
5. **Composition with PSD function:** $f(K_1)$ where $f(t) = \sum_{n=0}^{\infty} a_n t^n$ with $a_n \geq 0$

Proof.
See blackboard. □

Building Complex Kernels:

- $K(x,x') = K_1(x,x') + K_2(x,x')$ combines different types of similarity
- $K(x,x') = \exp(K_1(x,x'))$ where K_1 is PSD creates "exponential" variants
- Polynomial kernels: $(x^T x' + c)^d$

⚠ $K(x,x') = \tanh(x^T x')$ is *not* always PSD, despite being used in practice!

From Kernels to Function Spaces

The eigendecomposition construction works for finite point sets, but we want to understand kernels as defining infinite-dimensional function spaces.

Definition 2: Reproducing Kernel Hilbert Space

Given a PSD kernel K on domain $\mathcal{X}$, the **Reproducing Kernel Hilbert Space** $\mathcal{H}_K$ is a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$ with two key properties:

1. **Kernel functions are in the space:** For each $x \in \mathcal{X}$, the function $K_x(\cdot) := K(\cdot, x)$ belongs to $\mathcal{H}_K$.
2. **Reproducing property:** For any $f \in \mathcal{H}_K$ and $x \in \mathcal{X}$:

$$f(x) = \langle f, K_x \rangle_{\mathcal{H}_K} = \langle f, K(\cdot, x) \rangle_{\mathcal{H}_K}.$$

The RKHS is the "natural" function space associated with kernel K. Functions in $\mathcal{H}_K$ can be evaluated at any point via inner product with the kernel.

The Moore-Aronszajn Theorem

Theorem 2: Existence and Uniqueness of RKHS

For every positive semi-definite kernel K on $\mathcal{X}$, there exists a unique Reproducing Kernel Hilbert Space $\mathcal{H}_K$ having K as its reproducing kernel.

Proof.
See blackboard. □

Proposition 2: Key Properties

- $K(x, y) = \langle K(\cdot, x), K(\cdot, y) \rangle_{\mathcal{H}_K}$.
- For $f = \sum_i \alpha_i K(\cdot, x_i)$: $\|f\|^2_{\mathcal{H}_K} = \sum_i \sum_j \alpha_i \alpha_j K(x_i, x_j)$.

Proof.
See blackboard. □

RKHS Examples

Linear Kernel: $K(x, x') = x^T x'$

$\mathcal{H}_K = \{f(x) = w^T x : w \in \mathbb{R}^p\}$ with $\|f\|^2_{\mathcal{H}_K} = \|w\|^2$
This recovers standard linear regression/classification.

Polynomial Kernel: $K(x, x') = (1 + x^T x')^d$

$\mathcal{H}_K$ consists of polynomials up to degree d with appropriate norm.

Gaussian Kernel: $K(x, x') = \exp(-\|x - x'\|^2/(2\sigma^2))$

$\mathcal{H}_K$ is infinite-dimensional and very rich:

- Contains smooth functions
- Universal approximation: dense in $C(\mathcal{X})$ for compact $\mathcal{X}$
- Functions decay appropriately at infinity

💡 Different kernels encode different notions of function complexity via their RKHS norms.

Recap: SVM as a Penalization Method

Definition 3: Hinge Loss + Penalty Formulation

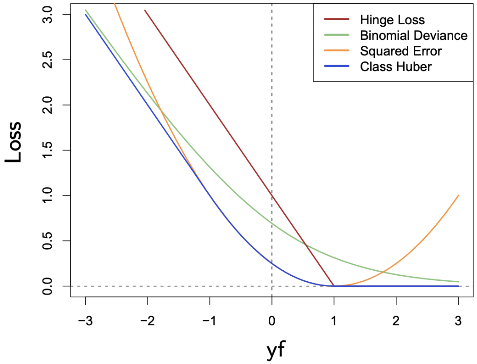
The SVM solution is equivalent to minimizing:

$$\min_{f \in \mathcal{H}_K} \left(\sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \lambda \|f\|^2_{\mathcal{H}_K} \right), \tag{3}$$

- $[1-u]_+ = \max(0, 1-u)$ is the **hinge loss function**.
- $\mathcal{H}_K$ is a Reproducing Kernel Hilbert Space defined by the kernel K.
- $\|f\|^2_{\mathcal{H}_K}$ is a penalty on the complexity (smoothness) of the function f.

Connection to Logistic Regression

This formulation is very similar to penalized logistic regression, which minimizes: $\sum_{i=1}^{N} \log(1 + e^{-y_i f(x_i)}) + \lambda \|f\|_{\mathcal{H}_K}^2$. The hinge loss and binomial deviance (logistic loss) are both convex surrogates for the 0-1 loss and produce similar classifiers.



The Key Result for Kernel Methods

Theorem 3: The Representer Theorem

Consider the regularized empirical risk minimization problem:

$$\min_{f \in \mathcal{H}_K} \left[\frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda \|f\|_{\mathcal{H}_K}^2 \right]$$

where L is any loss function, $\lambda > 0$, and $\mathcal{H}_K$ is the RKHS associated with PSD kernel K. Then the minimizer $\hat{f}$ has the finite representation: $\hat{f}(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i)$ for some coefficients $\alpha_1, \ldots, \alpha_N \in \mathbb{R}$.

Proof.
See blackboard. □

Remarkable: Even though we're optimizing over an infinite-dimensional function space, the solution lives in an n-dimensional subspace spanned by kernel evaluations at the training points!

Computational Implications

The infinite-dimensional optimization problem reduces to the finite-dimensional problem:

$$\min_{\alpha \in \mathbb{R}^n} \left[\sum_{i=1}^{N} L\left(y_i, \sum_{j=1}^{N} \alpha_j K(x_i, x_j) \right) + \lambda \sum_{i,j=1}^{N} \alpha_i \alpha_j K(x_i, x_j) \right],$$

which can be written as: $\min_{\alpha \in \mathbb{R}^N} \left[\sum_{i=1}^{N} L(y_i, (\mathbf{K}\alpha)_i) + \lambda \alpha^T \mathbf{K} \alpha \right]$.

- **Finite computation:** Only need to work with $N \times N$ Gram matrix $\mathbf{K}$
- **No explicit features:** Never compute $\phi(x)$, only kernel evaluations
- **General applicability:** Works for any loss function L
- **Scales with data:** Complexity is $O(N)$, not dimension of feature space

Kernel Ridge Regression

Linear Ridge Regression: $\min_w \|\mathbf{y} - \mathbf{X}w\|^2 + \lambda \|w\|^2$
Solution: $\hat{w} = (\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$
Prediction: $\hat{f}(x) = x^T \hat{w}$

The Representer Theorem allows us to substitute $f(x) = \sum_{i=1}^N \alpha_i K(x, x_i)$ into the general problem. For the squared loss, this leads to a convex optimization problem with a closed-form solution.

Definition 4: Kernel Ridge Regression

Apply the representer theorem with squared loss $L(y, f(x)) = (y - f(x))^2$:

- **Optimization problem:** $\min_{\alpha \in \mathbb{R}^N} \|\mathbf{y} - \mathbf{K}\alpha\|^2 + \lambda \alpha^T \mathbf{K}\alpha$
- **Solution:** $\hat{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{y}$
- **Prediction:** $\hat{f}(x) = \sum_{i=1}^N \hat{\alpha}_i K(x, x_i) = \mathbf{k}(x)^T (\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{y}$, where $\mathbf{k}(x) = (K(x, x_1), \ldots, K(x, x_N))^T$

Kernel Ridge Regression: Properties

Computational Complexity

- **Training:** $O(N^3)$ for matrix inversion (same as linear ridge with N features)
- **Prediction:** $O(N)$ per test point (need to evaluate N kernel functions)
- **Memory:** Store Gram matrix $\mathbf{K}$ (N^2 elements) and coefficients $\hat{\alpha}$ (N elements)

Relationship to Linear Ridge

When can we relate the two solutions?

- If $\mathbf{K} = \mathbf{X}\mathbf{X}^T$ (i.e., $K(x_i, x_j) = x_i^T x_j$), then:

$$\hat{\alpha} = (\mathbf{X}\mathbf{X}^T + \lambda \mathbf{I})^{-1}\mathbf{y}$$

- The linear solution is $\hat{w} = \mathbf{X}^T\hat{\alpha}$
- Both give same predictions: $\hat{f}(x) = x^T\hat{w} = \mathbf{k}(x)^T\hat{\alpha}$

Summary: The Power of the Kernel Viewpoint

- **Generality:** The kernel trick provides a powerful recipe for converting linear techniques that depend on inner products into non-linear methods capable of learning complex decision boundaries or regression functions.
- **Computational Efficiency:** By working with the Gram matrix $\mathbf{K}$, the complexity of the algorithm depends on the number of samples, not the (potentially infinite) dimension of the feature space.
- **Theoretical Foundation:** The theory of RKHS provides a rigorous mathematical framework for understanding why these methods work and for designing new kernel functions.
- **Unifying Framework:** The "Loss + Penalty in an RKHS" formulation, justified by the Representer Theorem, unifies many disparate-seeming methods (SVMs, Ridge Regression, Smoothing Splines) under a single elegant theory.

The Computational Bottleneck of Kernel Methods

The Representer Theorem is powerful, but it leads to a significant computational challenge for large datasets.

The Problem: The Gram Matrix

The solution for kernel methods like Kernel Ridge Regression or SVMs involves solving a linear system with the $N \times N$ Gram matrix $\mathbf{K}$.

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{y} \quad \text{(for Kernel Ridge Regression)}.$$

- Storage Cost: Storing the Gram matrix requires $O(N^2)$ memory.
- Computational Cost: Solving the linear system (e.g., via matrix inversion or Cholesky decomposition) costs $O(N^3)$ time. This is infeasible for large N.

Kernel methods in their standard form **do not scale well** with the number of samples N. We need approximation techniques.

Approximation via Random Fourier Features

Powerful way to approximate a kernel machine with a simple linear model, trading a controllable amount of accuracy for massive gain in compute efficiency.

Proposition 3: Bochner's Theorem

A continuous kernel $K(x, x')$ is positive semi-definite if and only if it is the Fourier transform of a non-negative measure. For a shift-invariant kernel $K(x, x') = K(x - x')$, this means:

$$K(x - x') = \int_{\mathbb{R}^d} p(\omega) e^{i\omega^T(x-x')} d\omega$$

where $p(\omega)$ is a non-negative probability density.

Proof.
See Bach Ch. 7.3.3. □

💡 We can approximate this integral with a Monte Carlo average. By sampling D frequencies ω_j from the density $p(\omega)$, we can construct an explicit, low-dimensional feature map that approximates the infinite-dimensional map.

The Random Fourier Features Algorithm

Random Fourier Features for Gaussian Kernels

```
1: Goal: Approximate the Gaussian kernel $K(x, x') = \exp(-\frac{\|x-x'\|^2}{2\sigma^2})$. Its Fourier transform is a Gaussian density.
2: Sampling: Draw $D$ random vectors $\omega_1, \ldots, \omega_D$ from $\mathcal{N}(0, \frac{1}{\sigma^2}\mathbf{I})$.
3: Feature Map: Create the new feature map $z : \mathbb{R}^d \to \mathbb{R}^D$: $z(x) = \frac{1}{\sqrt{D}} \begin{bmatrix} \cos(\omega_1^T x) \\ \sin(\omega_1^T x) \\ \vdots \\ \cos(\omega_D^T x) \\ \sin(\omega_D^T x) \end{bmatrix} \in \mathbb{R}^{2D}$.
4: Linear Model: Train a standard linear model (e.g., Ridge Regression or a linear SVC) on the new data $(z(x_i), y_i)$.
```

The inner product of these new features, $z(x)^T z(x')$, is an unbiased estimator of the true kernel value $K(x, x')$.

Theoretical Guarantees for Random Features

The performance of the linear model trained on random Fourier features converges to the performance of the full kernel machine.

Theorem 4: Approximation Guarantee (Informal)

Let $\hat{f}_D$ be the solution of Ridge Regression on D random Fourier features, and let $\hat{f}_{\mathcal{H}}$ be the solution of the full Kernel Ridge Regression. Then, for the squared loss case,

$$\mathbb{E}[R(\hat{f}_D)] - \mathbb{E}[R(\hat{f}_{\mathcal{H}})] = O\left(\frac{1}{\sqrt{D}}\right).$$

Proof.

See blackboard. □

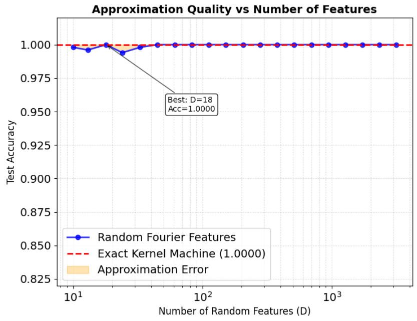
💡 A powerful result: by increasing D, we can get arbitrarily close to the performance of the exact kernel method. The computational cost is now only $O(ND^2 + D^3)$, which is much better than $O(N^3)$ if $D \ll N$.

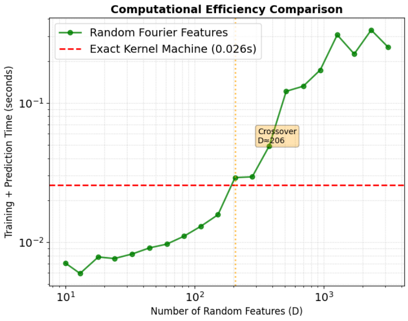
Example: Approximating a Kernel Machine

We can run a simulation to see how well the random features approximation works in practice.

- **Data:** A non-linearly separable "two moons" dataset with $N = 500$ points.
- **Models:**
  1. An exact Kernel Ridge Regression classifier (our gold standard).
  2. Several linear Ridge Regression classifiers trained on Random Fourier Features, with an increasing number of features D.
- **Analysis:** We will plot the test accuracy of the approximate models as a function of the number of random features D and compare it to the accuracy of the exact kernel machine.

Example: Random Features in Action





Random features can achieve nearly the same predictive power as a full kernel machine at a fraction of the computational cost, making kernel methods practical for large datasets!

💡 The seminal work by Rahimi and Recht on random features won the 2017 NIPS Test of Time Award:
`https://eecs.berkeley.edu/news/ben-recht-wins-nips-test-time-award/`

Exercises

Exercise 6

1. Nystrom approximation?
2. Show that if $k : \mathcal{X} \to \mathcal{X}$ is a positive-definite kernel, then so is the function $(x, x') \mapsto e^{k(x,x')}$.
3. (Mercer kernels) Solve Exercise 7.6 in Bach.
4. (Random feature expansion of Mercer kernels) Solve Exercise 7.10 in Bach.
5. Random features?