Yale University

Margin Classification

S&DS 265 · Introductory Machine Learning · Lecture 9

Zhuoran Yang

Fall 2026

AI usage: figures were generated, and these slides polished, with the help of AI tools.

Outline

Learning Objectives

In this lecture you will learn:

1. Why geometric margin measures distance and robustness for a linear classifier;
2. How canonical scaling turns maximum margin into a convex optimization problem;
3. How slack variables, hinge loss, and regularization handle overlapping classes;
4. How feature maps and kernels produce nonlinear boundaries from linear scores;
5. Why the fitted coefficients lie in the span of the data, so a kernel replaces the feature map.

Outline

Motivating Example: Choosing a Boundary

A classifier predicts whether a patient has angiographic heart disease from measured risk factors. More than one linear boundary can classify the same training observations correctly.

If training error cannot distinguish those boundaries, which one should we trust near the decision boundary?

Example and data: ISLP Chapter 9; course calculations use the ISLP `Heart` data.

The Heart Data

age	rest BP	cholesterol	max HR	oldpeak	sex	disease
63	145	233	150	2.3	male	No
67	160	286	108	1.5	male	Yes
67	120	229	129	2.6	male	Yes
⋮	⋮	⋮	⋮	⋮	⋮	⋮
38	138	175	173	0.0	male	No

Source: The `Heart` data, ISLP Chapter 9; available at `statlearning.com`.

303 patients

- 13 clinical features
- Binary response
- 6 incomplete rows

Each row is one patient. Resting blood pressure is in mm Hg, cholesterol is in mg/dL, and maximum heart rate is in beats per minute; `disease` is the binary target. Six incomplete rows are omitted when fitting.

Reading One Row

age	rest BP	cholesterol	max HR	oldpeak	sex	disease
63	145	233	150	2.3	male	No

This patient was 63 years old, had resting blood pressure 145, cholesterol 233, and no recorded angiographic heart disease. The full feature vector also contains categorical clinical measurements.

Problem Setup

For training observations (x_i, y_i),

$$x_i \in \mathbb{R}^d, \qquad y_i \in \{-1, +1\}.$$

A linear score and classifier are

$$f_{w,b}(x) = w^\top x + b, \qquad \hat{y}(x) = \operatorname{sign}\{f_{w,b}(x)\}.$$

The decision boundary is the hyperplane

$$\mathcal{H} = \{x : w^\top x + b = 0\}.$$

Numerical features are standardized because Euclidean distance and the margin depend on scale; categorical features are converted to indicator columns before fitting.

The Linear Score Model Class

We begin with the finite-dimensional class

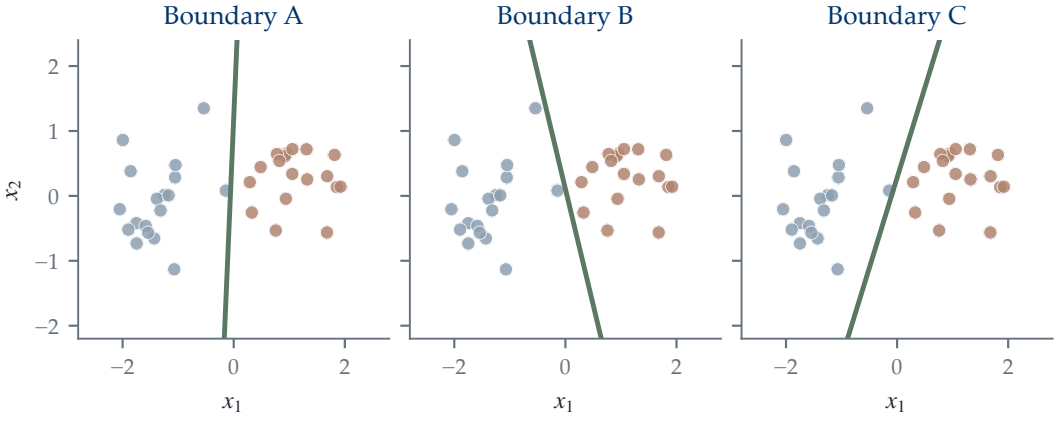
$$\mathcal{F}_{\text{lin}} = \{x \mapsto w^\top x + b : w \in \mathbb{R}^d,\ b \in \mathbb{R}\}.$$

Every member supplies a score, boundary, and classifier:

$$f(x) = w^\top x + b, \qquad \mathcal{H}_f = \{x : f(x) = 0\}, \qquad h_f(x) = \operatorname{sign}\{f(x)\}.$$

Restricting to this class shares one coefficient vector across all patients and makes estimation tractable, but it assumes the classes can be separated reasonably well by a hyperplane in the chosen feature coordinates.

Many Hyperplanes Can Separate the Same Data



Zero training error does not identify one separating boundary. We need a criterion for choosing among them.

Course illustration following the construction in ISLP Chapter 9.

Distance to a Hyperplane Is Scale-Free

derive

The sign of $w^\top x + b$ determines the predicted class. But scaling by any $c > 0$ leaves the boundary and predictions unchanged:

$$\operatorname{sign}\{c(w^\top x + b)\} = \operatorname{sign}\{w^\top x + b\}.$$

The shortest displacement to the boundary is parallel to its normal w. Write $\Delta = -aw$ and require $w^\top(x + \Delta) + b = 0$. Then

$$a = \frac{w^\top x + b}{\|w\|_2^2}, \qquad \boxed{\|\Delta\|_2 = \frac{|w^\top x + b|}{\|w\|_2}}.$$

Dividing by $\|w\|_2$ removes the arbitrary score scale.

The Signed Geometric Margin

For a labeled observation (x_i, y_i), define

$$\boxed{\gamma_i(w, b) = \frac{y_i(w^\top x_i + b)}{\|w\|_2}}.$$

- $\gamma_i > 0$: correctly classified;
- $\gamma_i = 0$: on the decision boundary;
- $\gamma_i < 0$: misclassified;
- Larger positive γ_i: farther from the boundary on the correct side.

The minimum margin is $\gamma_{\min}(w, b) = \min_i \gamma_i(w, b)$.

Margin Is a Robustness Radius

For a correctly classified (x_i, y_i), find the smallest perturbation that reaches the boundary:

$$\min_{\Delta:\, w^\top(x_i+\Delta)+b=0} \|\Delta\|_2 = \frac{|w^\top x_i + b|}{\|w\|_2} = \gamma_i(w, b).$$

The margin is exactly the feature perturbation required to erase this observation's current classification. Maximizing the minimum margin therefore chooses the separating boundary with the largest such radius around its nearest training observations.

This is a geometric motivation, not a guarantee against every kind of distribution shift or measurement error.

Maximum Margin Chooses the Widest Separation

Among separating hyperplanes, choose the one whose closest training observations are farthest away.

Course illustration following ISLP Chapter 9.

Outline

Canonical Scaling Fixes the Score Ambiguity

derive

The scale-free maximum-margin objective is

$$\max_{w,b} \min_i \frac{y_i(w^\top x_i + b)}{\|w\|_2}.$$

Multiplying (w, b) by any positive constant changes neither the ratio nor the boundary. Use this freedom to set the smallest **functional margin** to one. Then

$$\min_i y_i(w^\top x_i + b) = 1 \quad \Longrightarrow \quad \min_i \frac{y_i(w^\top x_i + b)}{\|w\|_2} = \frac{1}{\|w\|_2}.$$

Thus maximizing geometric margin is equivalent to minimizing $\|w\|_2$; the normalization changes no boundary or prediction.

Hard-Margin Support-Vector Classifier

The normalized problem is

$$\begin{array}{ll} \min\limits_{w,b} & \frac{1}{2}\|w\|_2^2 \\ \text{subject to} & y_i(w^\top x_i + b) \geq 1, \qquad i = 1, \dots, n. \end{array}$$

The two margin planes are $w^\top x + b = +1$ and $w^\top x + b = -1$. Their distance is

$$\frac{2}{\|w\|_2}.$$

This is a convex quadratic problem with linear constraints.

Support Vectors Determine the Boundary

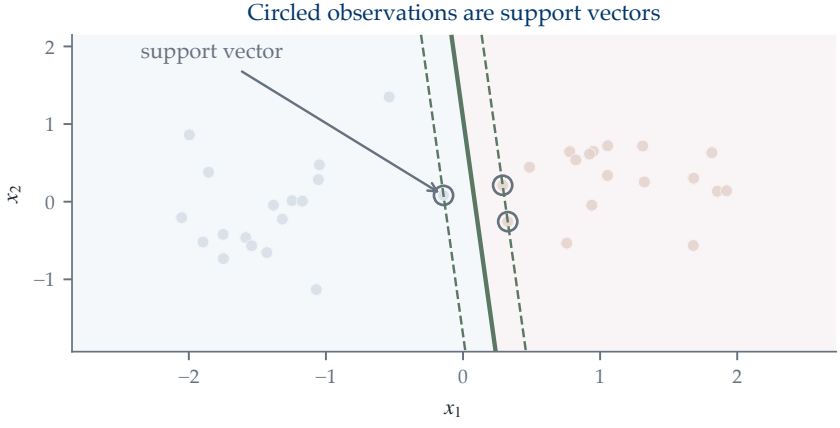
At the optimum, observations satisfying

$$y_i(\hat{w}^\top x_i + \hat{b}) = 1$$

lie on a margin plane and are called **support vectors**.

- Moving a point far outside the margin does not change the solution until it reaches the margin.
- Moving a support vector can rotate or translate the boundary.
- The fitted classifier is therefore controlled locally by observations nearest the boundary.

Only the Closest Observations Support the Margin



Circled observations touch the margin. They constrain the selected hyperplane; distant observations do not.

Course calculation on a declared two-dimensional dataset.

A Single Observation Can Collapse the Hard Margin

Hard-margin classification requires

$$y_i(w^\top x_i + b) > 0 \qquad \text{for every } i.$$

One overlapping or mislabeled observation can make the constraints infeasible. Even when the sample remains separable, one noisy observation can force a narrow, unstable margin.

The next model should tolerate violations and charge for them, rather than requiring perfect separation.

Outline

Slack Variables Measure Margin Violations

Introduce $\xi_i \geq 0$ and relax the constraints:

$$y_i(w^\top x_i + b) \geq 1 - \xi_i.$$

value of ξ_i	interpretation
$\xi_i = 0$	on or outside the correct margin
$0 < \xi_i < 1$	correct side, inside the margin
$\xi_i \geq 1$	on or across the decision boundary

The Soft-Margin Objective

The support-vector classifier solves

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(w^\top x_i + b) \geq 1 - \xi_i, \qquad \xi_i \geq 0. \end{aligned}$$

- Small C: wider margin, more tolerance for violations;
- Large C: violations are expensive, so the fit follows training data more closely.

Select C using validation or cross-validation, as in Lecture 4.

Eliminating Slack Variables Gives Hinge Loss

For fixed (w, b), the smallest feasible slack is

$$\xi_i = \max\{0, 1 - y_i(w^\top x_i + b)\}.$$

Substitution yields the regularized empirical-risk problem

$$\min_{w,b} \frac{1}{2}\|w\|_2^2 + C \sum_{i=1}^{n} \underbrace{\max\{0, 1 - y_i(w^\top x_i + b)\}}_{\text{hinge loss}}.$$

The model uses only a score and margin; unlike logistic regression, it does not by itself estimate a probability.

Recall: Logistic Loss Was Also a Function of the Margin

In Lecture 8 the per-observation loss collapsed to a function of the margin $m_i = y_i(w^\top x_i + b)$:

$$\ell_i^{\text{logistic}} = \log(1 + e^{-m_i}).$$

We have just derived a different one:

$$\ell_i^{\text{hinge}} = \max\{0,\, 1 - m_i\}.$$

Recall: Logistic Loss Was Also a Function of the Margin

In Lecture 8 the per-observation loss collapsed to a function of the margin $m_i = y_i(w^\top x_i + b)$:

$$\ell_i^{\text{logistic}} = \log(1 + e^{-m_i}).$$

We have just derived a different one:

$$\ell_i^{\text{hinge}} = \max\{0, 1 - m_i\}.$$

Both say the same kind of thing — be right, and be right by a margin — and differ only in how they charge for it.

Recall: Logistic Loss Was Also a Function of the Margin

In Lecture 8 the per-observation loss collapsed to a function of the margin $m_i = y_i(w^\top x_i + b)$:

$$\ell_i^{\text{logistic}} = \log(1 + e^{-m_i}).$$

We have just derived a different one:

$$\ell_i^{\text{hinge}} = \max\{0,\, 1 - m_i\}.$$

Both say the same kind of thing — be right, and be right by a margin — and differ only in how they charge for it.

So both methods minimize the same shape of objective, and only the function L changes.

One Template, Two Losses

Divide the soft-margin objective by nC, which does not move its minimizer, and both methods read

$$\min_{w,b} \frac{1}{n}\sum_{i=1}^{n} L(m_i) + \frac{\lambda}{2}\|w\|_2^2, \qquad m_i = y_i(w^\top x_i + b)$$

One Template, Two Losses

Divide the soft-margin objective by nC, which does not move its minimizer, and both methods read

$$\min_{w,b} \frac{1}{n}\sum_{i=1}^{n} L(m_i) + \frac{\lambda}{2}\|w\|_2^2, \qquad m_i = y_i(w^\top x_i + b)$$

- $L(m) = \log(1 + e^{-m})$: **logistic regression**;
- $L(m) = \max\{0, 1 - m\}$: the **support vector machine**, with $\lambda = 1/(nC)$;
- $L(m) = \mathbf{1}\{m < 0\}$ would be the error count itself — not convex, and not usable.

Software scales C differently, so check the convention before carrying a numerical value between packages.

What This View Buys

- Logistic regression began with a probability model: assume the log-odds is linear, then maximize likelihood. The loss was a consequence.
- The SVM begins with geometry: separate the classes with the widest gap. It assumes no distribution, and returns a score rather than a probability.
- Yet both land on the same template. A margin loss builds a classifier that needs no probabilistic model at all — convexity and a penalty are enough.

What This View Buys

- Logistic regression began with a probability model: assume the log-odds is linear, then maximize likelihood. The loss was a consequence.
- The SVM begins with geometry: separate the classes with the widest gap. It assumes no distribution, and returns a score rather than a probability.
- Yet both land on the same template. A margin loss builds a classifier that needs no probabilistic model at all — convexity and a penalty are enough.

The price is that an SVM score is not a probability. If deployment needs one, it must be calibrated afterwards.

Which Observations Move the Fit

The hinge is piecewise linear in the margin, so its slope is obvious on each piece:

$$L(m) = \max\{0, 1 - m\}, \qquad \frac{dL}{dm} = \begin{cases} -1, & m < 1 \quad \text{(inside the margin, or wrong)}, \\ 0, & m > 1 \quad \text{(safely correct)}. \end{cases}$$

Which Observations Move the Fit

The hinge is piecewise linear in the margin, so its slope is obvious on each piece:

$$L(m) = \max\{0, 1 - m\}, \qquad \frac{dL}{dm} = \begin{cases} -1, & m < 1 \quad \text{(inside the margin, or wrong)}, \\ 0, & m > 1 \quad \text{(safely correct)}. \end{cases}$$

A gradient step therefore uses only the rows with $m_i < 1$:

$$w \;\longleftarrow\; w - \eta\Big(\lambda w - \frac{1}{n} \sum_{i:\, m_i < 1} y_i x_i\Big).$$

Which Observations Move the Fit

The hinge is piecewise linear in the margin, so its slope is obvious on each piece:

$$L(m) = \max\{0, 1 - m\}, \qquad \frac{dL}{dm} = \begin{cases} -1, & m < 1 \quad \text{(inside the margin, or wrong)}, \\ 0, & m > 1 \quad \text{(safely correct)}. \end{cases}$$

A gradient step therefore uses only the rows with $m_i < 1$:

$$w \;\longleftarrow\; w - \eta\Big(\lambda w - \frac{1}{n} \sum_{i:\, m_i < 1} y_i x_i\Big).$$

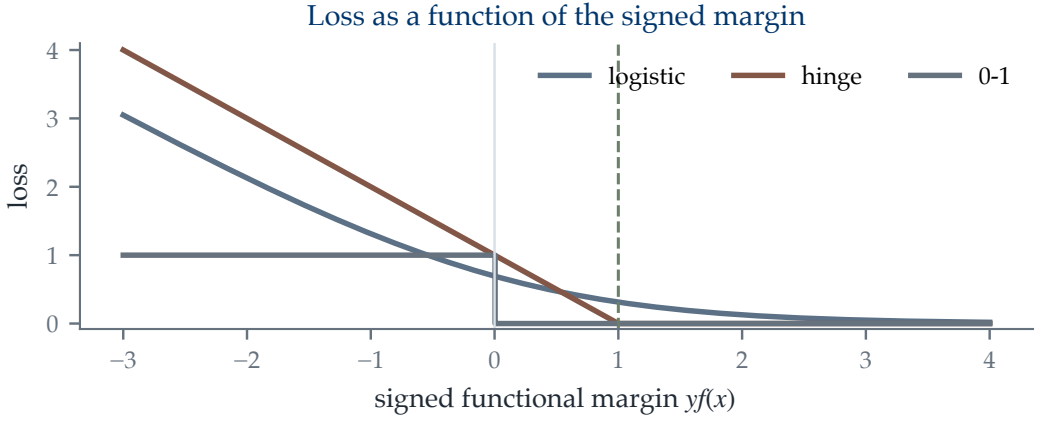
Correctly classified points beyond the margin contribute nothing. That is the support-vector fact again, now visible in the update.

The SVM Fitting Workflow

1. Encode labels as $-1, +1$ and standardize numerical features;
2. Choose a linear score or a feature map/kernel;
3. Minimize hinge empirical risk plus a coefficient penalty;
4. Choose C and any kernel parameters by validation or cross-validation;
5. Evaluate class-specific errors on held-out data; calibrate the fitted score separately if deployment needs probabilities.

The model class chooses possible boundaries, hinge loss chooses how violations count, regularization chooses stability, and the optimizer estimates the coefficients.

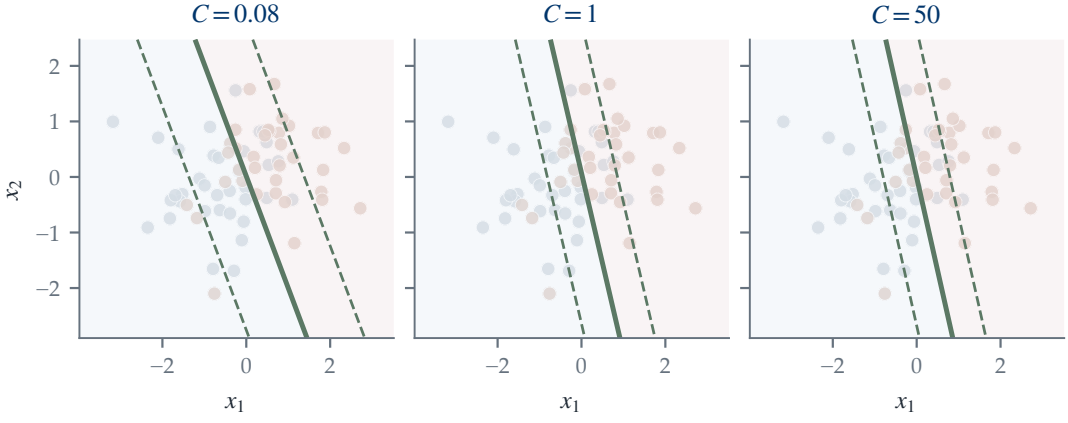
Logistic and Hinge Loss Reward Different Margins



Logistic loss decreases smoothly for every positive margin. Hinge loss is exactly zero once the functional margin reaches one.

Course calculation.

The Regularization Parameter Controls Margin Violations



A smaller C accepts more violations to obtain a wider, less sample-sensitive margin; a larger C fits the training sample more tightly.

Course calculation using linear support-vector classifiers.

Logistic Regression and SVM Separate Modeling Goals

	logistic regression	support-vector classifier
fitted object	conditional probability	decision score and margin
data loss	logistic / cross-entropy	hinge
far correct points	retain diminishing influence	zero hinge loss
direct output	$p(y = 1 \mid x)$	$\operatorname{sign} f(x)$

Both can use a linear score, regularization, and fixed nonlinear features.

Outline

A Linear Classifier Can Be Nonlinear in the Raw Input

Choose a feature map $\phi : \mathbb{R}^d \to \mathbb{R}^m$ and score

$$f_{w,b}(x) = w^\top \phi(x) + b.$$

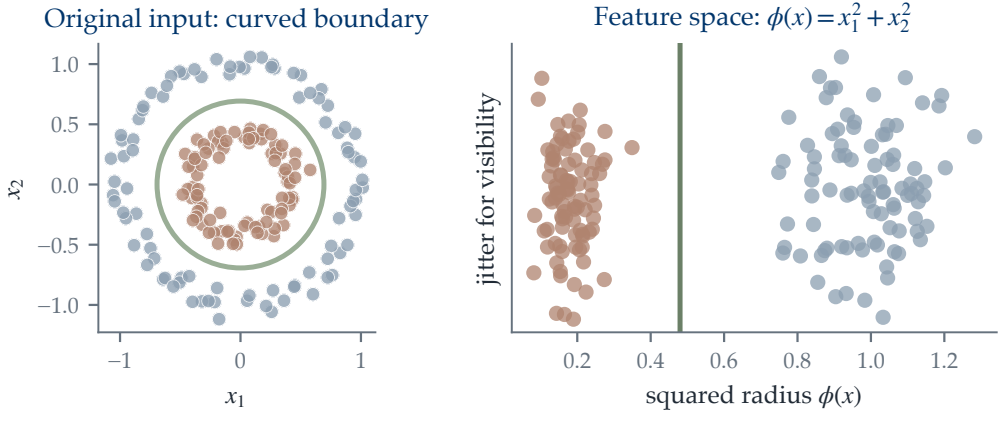
Logistic regression uses

$$p(y = 1 \mid x) = \sigma\{w^\top \phi(x) + b\},$$

while an SVM uses the same score inside a hinge-loss objective.

The boundary is a hyperplane in feature space but can curve in the original input space.

A Feature Map Can Make a Curved Boundary Linear



The feature map adds squared radius. A plane in the enlarged feature space corresponds to a circular boundary in the original plane.

Feature-expansion story follows ISLP Chapter 9.

Writing the Features Down Gets Expensive

Suppose ϕ collects every monomial of degree at most q. Then

$$m = \binom{d+q}{q},$$

and the `Heart` data has $d = 13$ features:

degree q	2	3	5
features m	105	560	8,568

Writing the Features Down Gets Expensive

Suppose ϕ collects every monomial of degree at most q. Then

$$m = \binom{d+q}{q},$$

and the `Heart` data has $d = 13$ features:

degree q	2	3	5
features m	105	560	8,568

We want the flexibility of a large ϕ without ever storing a vector of length m — and for some maps m is infinite.

The Fitted w Lies in the Span of the Data

derive

Split any candidate w along the subspace spanned by the training features, $\mathcal{S} = \text{span}\{\phi(x_1), \dots, \phi(x_n)\}$:

$$w = u + v, \qquad u \in \mathcal{S}, \quad v \perp \mathcal{S}.$$

The Fitted w Lies in the Span of the Data

derive

Split any candidate w along the subspace spanned by the training features, $\mathcal{S} = \text{span}\{\phi(x_1), \dots, \phi(x_n)\}$:

$$w = u + v, \qquad u \in \mathcal{S}, \quad v \perp \mathcal{S}.$$

The part v is invisible to the data, since $v^\top \phi(x_i) = 0$ gives

$$w^\top \phi(x_i) + b = u^\top \phi(x_i) + b \quad \Longrightarrow \quad \text{every margin } m_i \text{ is unchanged,}$$

The Fitted w Lies in the Span of the Data

derive

Split any candidate w along the subspace spanned by the training features, $\mathcal{S} = \text{span}\{\phi(x_1), \dots, \phi(x_n)\}$:

$$w = u + v, \qquad u \in \mathcal{S}, \quad v \perp \mathcal{S}.$$

The part v is invisible to the data, since $v^\top \phi(x_i) = 0$ gives

$$w^\top \phi(x_i) + b = u^\top \phi(x_i) + b \quad \Longrightarrow \quad \text{every margin } m_i \text{ is unchanged,}$$

but it is not invisible to the penalty: $\|w\|_2^2 = \|u\|_2^2 + \|v\|_2^2$. Dropping v leaves every loss term alone and lowers the penalty, so the minimizer has $v = 0$:

The Fitted w Lies in the Span of the Data

derive

Split any candidate w along the subspace spanned by the training features, $\mathcal{S} = \text{span}\{\phi(x_1), \dots, \phi(x_n)\}$:

$$w = u + v, \qquad u \in \mathcal{S}, \quad v \perp \mathcal{S}.$$

The part v is invisible to the data, since $v^\top \phi(x_i) = 0$ gives

$$w^\top \phi(x_i) + b = u^\top \phi(x_i) + b \quad \Longrightarrow \quad \text{every margin } m_i \text{ is unchanged,}$$

but it is not invisible to the penalty: $\|w\|_2^2 = \|u\|_2^2 + \|v\|_2^2$. Dropping v leaves every loss term alone and lowers the penalty, so the minimizer has $v = 0$:

it suffices to search over $w \in \mathcal{S}$

The Coefficients Are Weights on Training Rows

Therefore the fitted coefficient vector is a combination of the training features,

$$\widehat{w} = \sum_{i=1}^{n} \alpha_i \, \phi(x_i)$$

for some $\alpha \in \mathbb{R}^n$: the m unknowns in w collapse to n unknowns in α, whatever m was.

The subgradient step says the same thing concretely: starting from $w = 0$, each update adds multiples of $y_i \phi(x_i)$ for rows with $m_i < 1$ and shrinks what is already there, so w never leaves the span.

The Feature Map Appears Only in Inner Products

Substitute $\widehat{w} = \sum_i \alpha_i \phi(x_i)$ into the score:

$$\widehat{w}^\top \phi(x) + b = \sum_{i=1}^{n} \alpha_i \underbrace{\phi(x_i)^\top \phi(x)}_{\text{an inner product}} + b.$$

The Feature Map Appears Only in Inner Products

Substitute $\widehat{w} = \sum_i \alpha_i \phi(x_i)$ into the score:

$$\widehat{w}^\top \phi(x) + b = \sum_{i=1}^{n} \alpha_i \underbrace{\phi(x_i)^\top \phi(x)}_{\text{an inner product}} + b.$$

The coordinates of ϕ have vanished from the calculation. What remains is one number per pair of inputs, so define

$$\boxed{K(x, x') = \phi(x)^\top \phi(x')}$$

The Feature Map Appears Only in Inner Products

Substitute $\widehat{w} = \sum_i \alpha_i \phi(x_i)$ into the score:

$$\widehat{w}^\top \phi(x) + b = \sum_{i=1}^{n} \alpha_i \underbrace{\phi(x_i)^\top \phi(x)}_{\text{an inner product}} + b.$$

The coordinates of ϕ have vanished from the calculation. What remains is one number per pair of inputs, so define

$$\boxed{K(x, x') = \phi(x)^\top \phi(x')}$$

Supply K directly and the feature map need never be built — it may even be infinite-dimensional.

The Same Problem, Written in α

Collect the kernel values in the **Gram matrix** $\mathbf{K} \in \mathbb{R}^{n \times n}$, $\mathbf{K}_{ij} = K(x_i, x_j)$. Both pieces of the objective are then expressible in α alone:

$$m_i = y_i\{(\mathbf{K}\alpha)_i + b\}, \qquad \|w\|_2^2 = \alpha^\top \mathbf{K}\alpha.$$

The Same Problem, Written in α

Collect the kernel values in the **Gram matrix** $\mathbf{K} \in \mathbb{R}^{n \times n}$, $\mathbf{K}_{ij} = K(x_i, x_j)$. Both pieces of the objective are then expressible in α alone:

$$m_i = y_i\{(\mathbf{K}\alpha)_i + b\}, \qquad \|w\|_2^2 = \alpha^\top \mathbf{K}\alpha.$$

So the regularized problem becomes

$$\min_{\alpha \in \mathbb{R}^n,\, b} \; \frac{1}{n} \sum_{i=1}^{n} L(y_i\{(\mathbf{K}\alpha)_i + b\}) + \frac{\lambda}{2}\alpha^\top \mathbf{K}\alpha$$

Still convex, because $\mathbf{K}$ is positive semidefinite. Nothing about the method changed — only the variables we solve for.

Fitting and Predicting with a Kernel

1. Compute the Gram matrix $\mathbf{K}_{ij} = K(x_i, x_j)$ once;
2. Minimize the objective above over α and b;
3. Predict at a new x with $\hat{y}(x) = \text{sign}\{\sum_i \hat{\alpha}_i K(x_i, x) + b\}$.

Fitting and Predicting with a Kernel

1. Compute the Gram matrix $\mathbf{K}_{ij} = K(x_i, x_j)$ once;
2. Minimize the objective above over α and b;
3. Predict at a new x with $\hat{y}(x) = \text{sign}\{\sum_i \hat{\alpha}_i K(x_i, x) + b\}$.

- Under hinge loss most $\hat{\alpha}_i$ are zero: only support vectors carry weight, so prediction costs one kernel evaluation per support vector.
- Training cost now grows with n, not with m. Kernels trade feature dimension for sample size.

Not the Kernel of Lecture 6

Same word, different job. Compare carefully.

	kernel smoothing (L07)	kernel method (here)
what K does	weights nearby responses	computes an inner product
K is applied to	one query at a time	pairs of training inputs
the fitted rule	a local average	one global linear rule in ϕ
K must be	a decaying weight	an inner product of some ϕ

Not the Kernel of Lecture 6

Same word, different job. Compare carefully.

	kernel smoothing (L07)	kernel method (here)
what K does	weights nearby responses	computes an inner product
K is applied to	one query at a time	pairs of training inputs
the fitted rule	a local average	one global linear rule in ϕ
K must be	a decaying weight	an inner product of some ϕ

In Lecture 6, K said how much to count a neighbour. Here it says how similar two inputs are in a feature space we never build.

Three Kernels

- **Linear.** $K(x, x') = x^\top x'$, which is $\phi(x) = x$: the original linear classifier.
- **Polynomial.** $K(x, x') = (1 + x^\top x')^q$. Expanding the power shows ϕ contains every product of up to q coordinates — all interactions, never listed.
- **Radial basis (RBF).** $K(x, x') = e^{-\gamma \|x - x'\|_2^2}$, which is 1 when $x = x'$ and decays with distance: a similarity. Its ϕ is infinite-dimensional, which is exactly why we do not form it.

γ sets how quickly similarity decays, so it plays the role the bandwidth h played in Lecture 6 and is chosen the same way, by cross-validation.

Multiclass Strategies: Native and Reduced

Native K-class probability model

Softmax fits K scores jointly:

$$p_k(x) = \frac{e^{s_k(x)}}{\sum_j e^{s_j(x)}}.$$

Cross-entropy trains all class scores together.

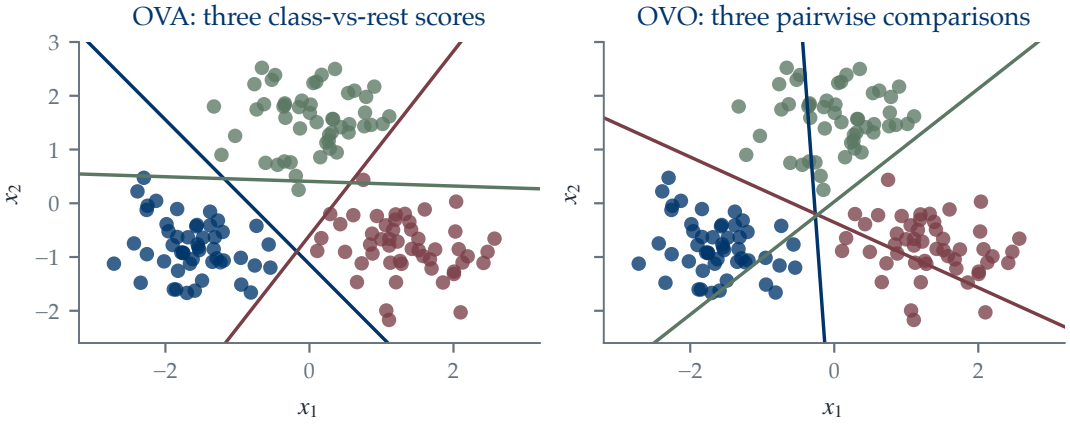
Reduction to binary classifiers

OVA: fit K class-versus-rest rules.

OVO: fit $K(K-1)/2$ pairwise rules and vote.

SVM software commonly uses OVO or OVA; multiclass logistic regression usually uses the native softmax model.

OVA and OVO Decompose a Multiclass Problem



OVA asks whether each class defeats the rest; OVO asks which class wins each pairwise comparison.

Course calculation on a declared three-class dataset.

Summary: Margin Classification

$$\gamma_i = \frac{y_i(w^\top x_i + b)}{\|w\|_2}, \qquad \min_{w,b} \frac{1}{2}\|w\|_2^2 + C\sum_i \max\{0, 1 - y_i(w^\top x_i + b)\}.$$

- Maximum margin chooses among separating hyperplanes; support vectors are the observations that constrain the fit.
- Hinge and logistic loss are two choices of L in one regularized margin objective.
- The fitted coefficients lie in the span of the training features, so the whole method can be written with a kernel and never forms ϕ.

Next Lecture

Linear, logistic, and margin methods build one global score from all training observations. Their geometry depends on a selected raw or constructed feature representation.

Lecture 10 — Decision Trees, Random Forests, and Gradient Boosting partitions feature space through recursive questions, then combines unstable trees into stronger ensemble predictors.

Reading for this lecture: ISLP Chapter 9.