上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Lecture 11: Support Vector Machines

Tao Huang

John Hopcroft Center, School of Computer Science, Shanghai Jiao Tong University

https://taohuang.info/cs3317

https://oc.sjtu.edu.cn/courses/89538

1. Margin & Support Vectors

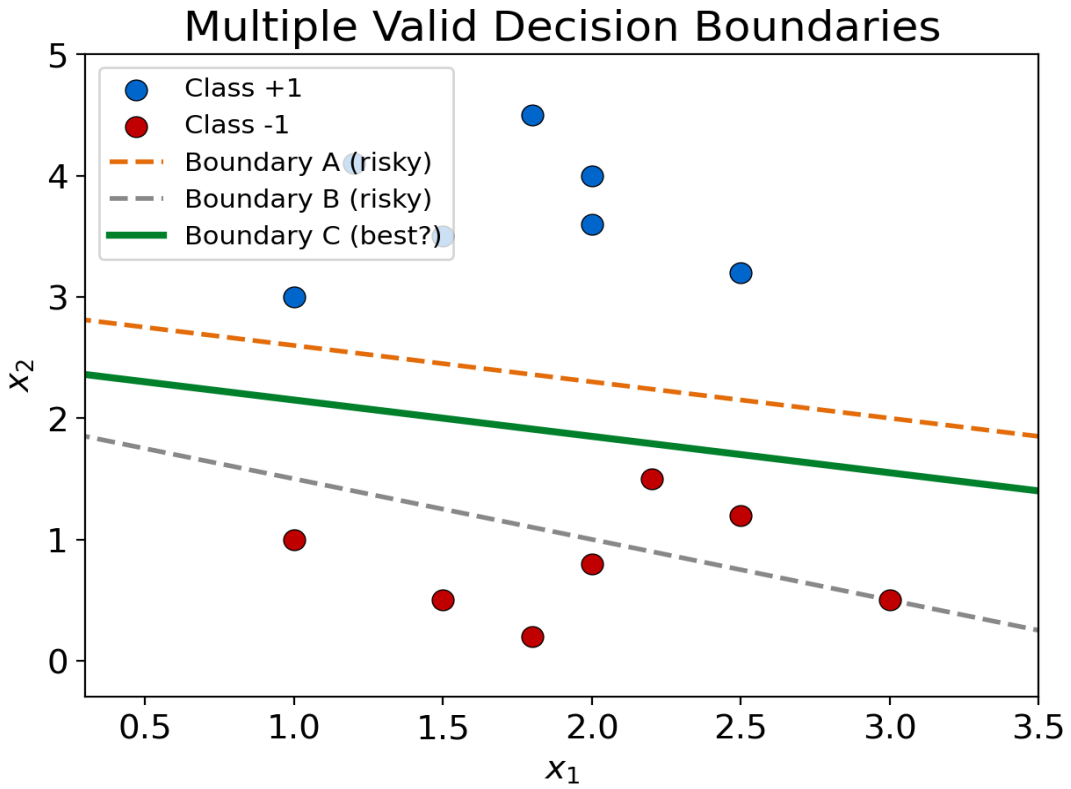
Recap: Linear Classification

Recall from L9: linear classifier

$$f(x) = \text{sign}(w \cdot \phi(x))$$

- Decision boundary: $w \cdot \phi(x) = 0$
- We learned how to find w via logistic regression + gradient descent

- **Question: Is the boundary we found actually the best one?**



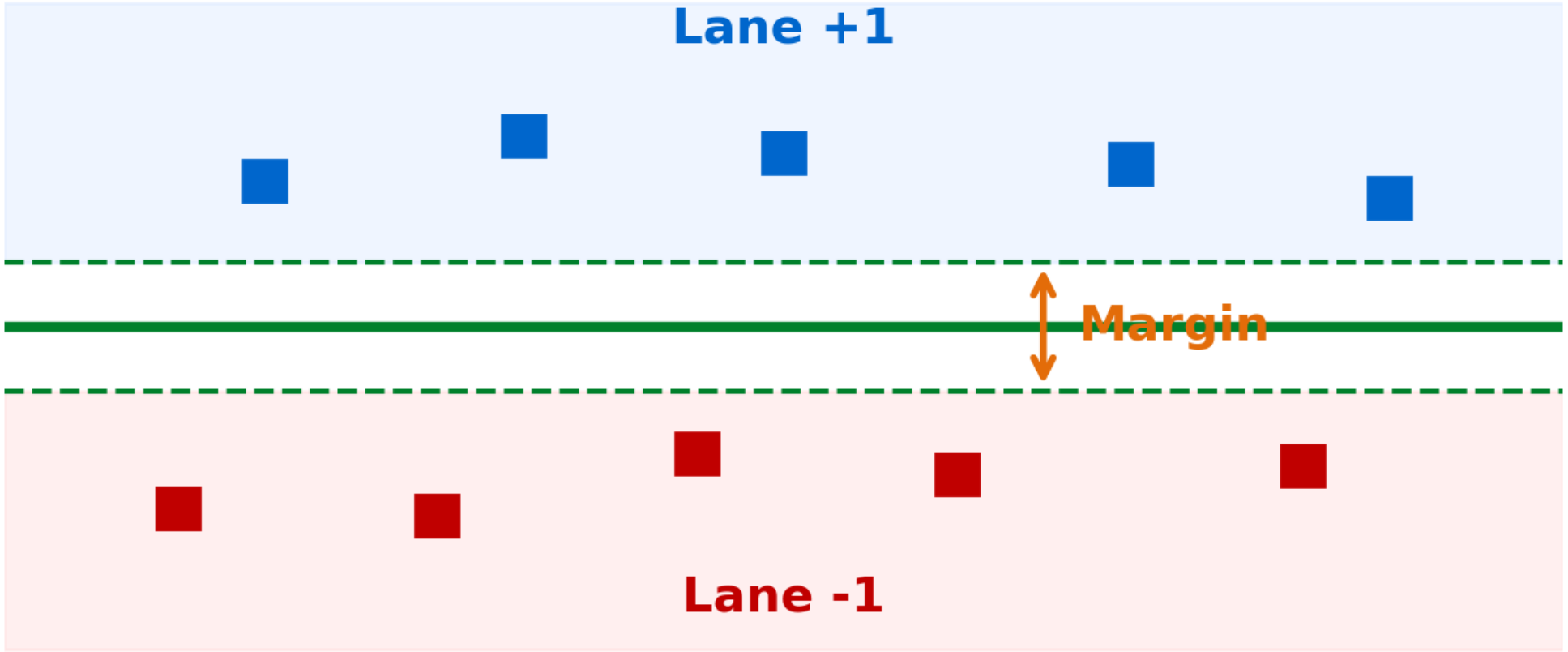
Many Valid Boundaries Exist

- Consider a linearly separable 2D dataset:
- All three lines below separate the data with 100% training accuracy:
  - Line A: passes very close to the + class
  - Line B: passes very close to the – class
  - Line C: stays far from both classes

- **Same accuracy, very different boundaries.**
- **Question: Which boundary would you trust on new data? Why?**

Analogy: The Safest Lane Divider

SVM = Finding the Safest Lane Divider



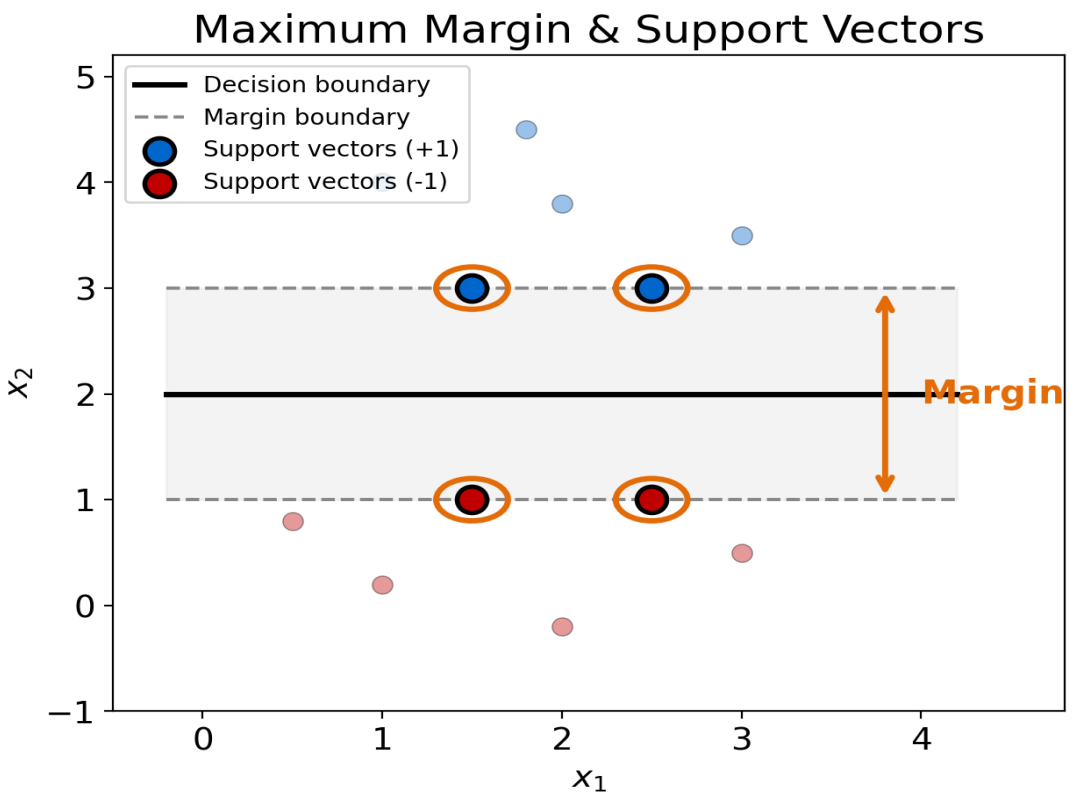
The Margin

Definition

- The margin is the distance between the decision boundary and the nearest data point from either class.

Why maximize margin?

- Wider margin → more room for error on new data
- Better generalization (recall overfitting from L8!)
- Robust to small perturbations in input



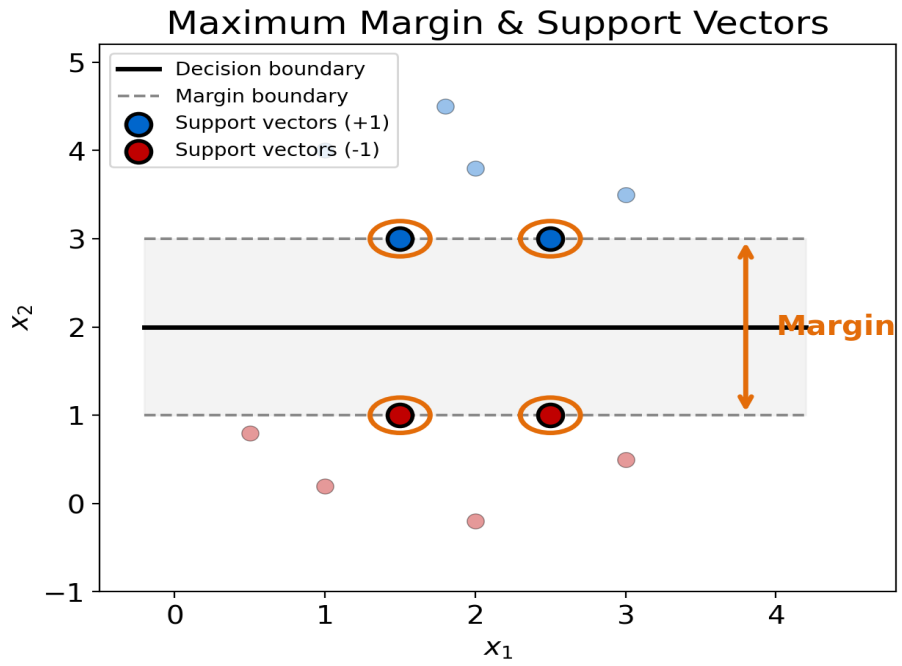
Support Vectors

The nearest points that define the margin are called support vectors.

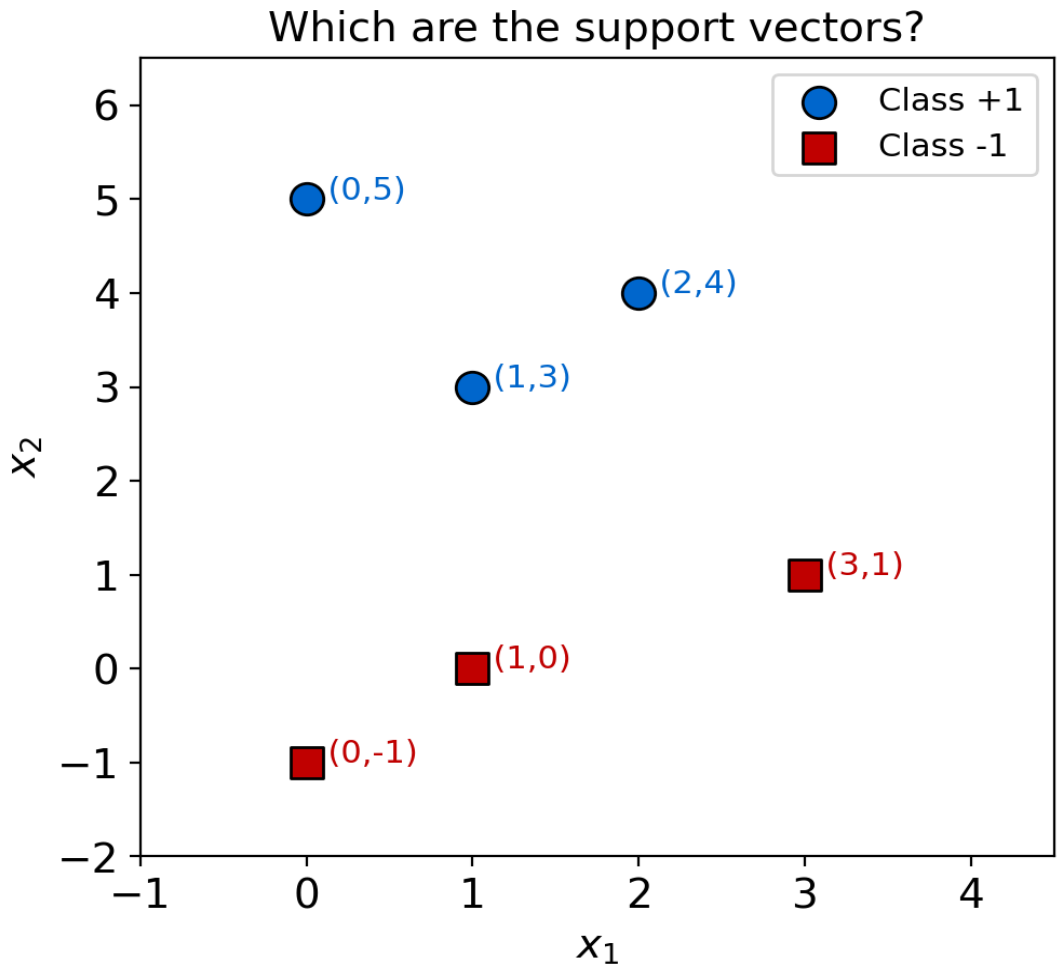
Key Insight:

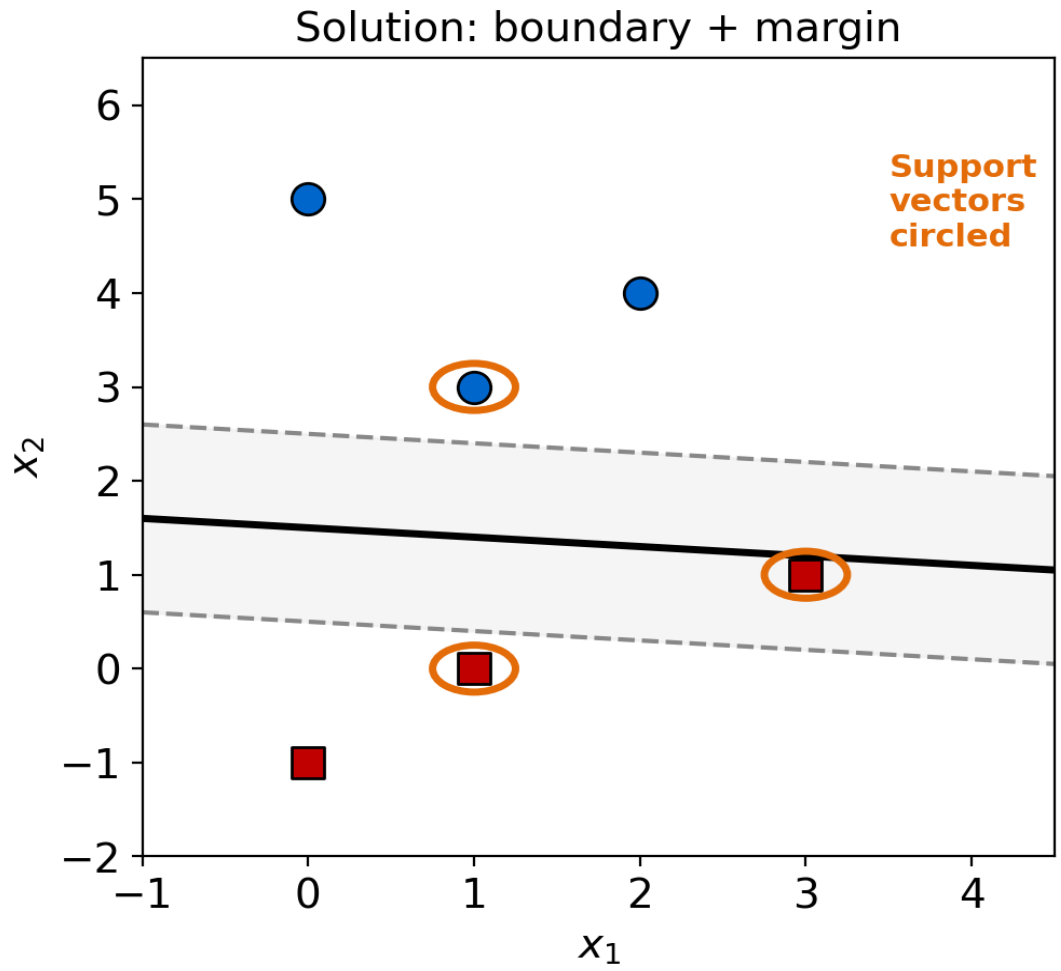
- Only support vectors determine the boundary
- Remove any non-support-vector point → boundary unchanged
- Remove a support vector → boundary changes!

- Think of it as: the boundary is propped up by a few critical examples.
- **This gives SVM its name: Support Vector Machine.**



Think: Identify the Support Vectors





Functional Margin

Setup

- Binary classification: $y \in \{+1, -1\}$
- Linear model: $f(x) = w^T x + b$

Functional Margin of point (x_i, y_i):

$$\hat{y}_i = y_i(w^T x_i + b)$$

- Correct classification ↔ functional margin > 0
- Larger value → more confident prediction

Problem: we can scale w and b to inflate this arbitrarily!

Geometric Margin

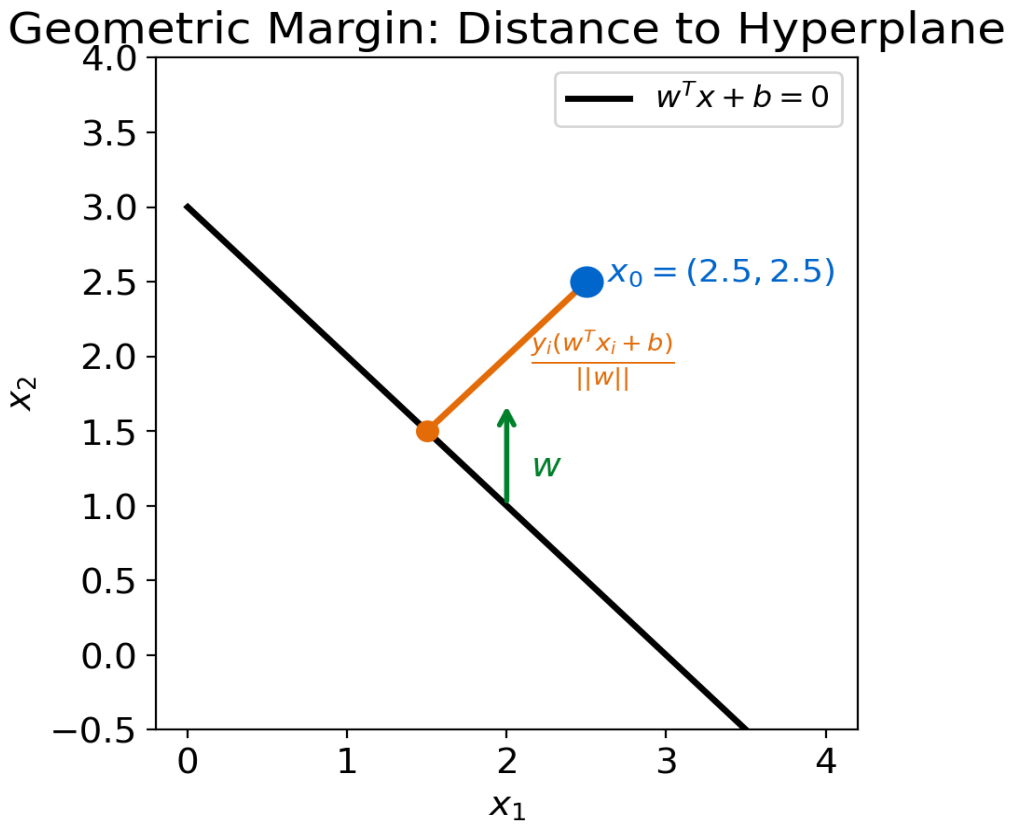
Normalize by ||w|| to get the true distance:

- Geometric margin $= y_i(w^T x_i + b)/\|w\|$
- This is the perpendicular distance from point to hyperplane.

The margin of the classifier:

= minimum geometric margin over all training points

- Now scaling w doesn't change the margin!



🧠 Quick Check: Compute the Margin

Given: w = [1, 1], b = −3, point x = [2, 2], label y = +1

Step 1: Functional margin

- $y(w^T x + b) = 1 \times (1 \times 2 + 1 \times 2 - 3) = 1$

Step 2: Geometric margin

- $1 / ||w|| = 1 / \sqrt{2} \approx 0.707$

Now try: w = [2, 2], b = −6 (same boundary!)

- Functional margin: 2 (doubled!)
- Geometric margin: $2 / \sqrt{8} = 1/\sqrt{2} \approx 0.707$ (same!)

→ Geometric margin is scale-invariant. That's why we use it.

Maximum Margin Objective

Goal: maximize the minimum geometric margin

- By convention, fix functional margin of support vectors = 1.
- Then margin = 1 / ||w||.

Equivalent formulation:

- minimize $(1/2)||w||^2$
- subject to $y_i(w^T x_i + b) \geq 1, \forall i$

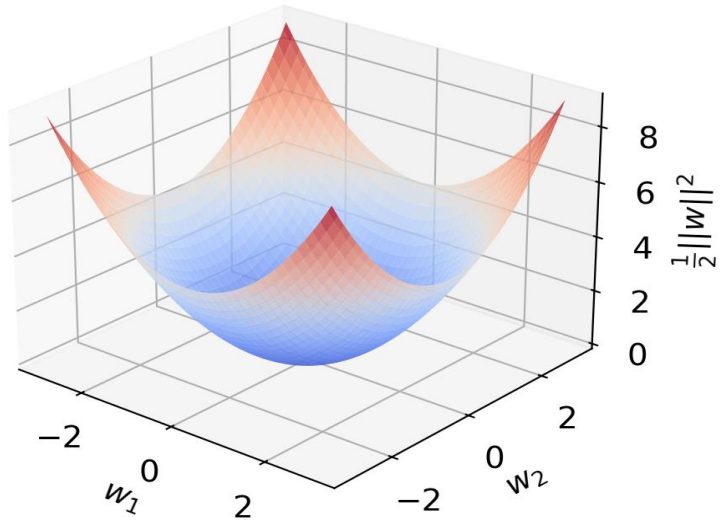
- This is a convex quadratic program → guaranteed global optimum!

Why Minimize $\|w\|^2$?

$$\max \text{ margin} = \max\left(\frac{1}{\|w\|}\right) = \min \|w\| = \min \frac{1}{2}\|w\|^2$$

- Squaring makes it smooth and differentiable
- The 1/2 factor gives clean derivatives

Convex Objective: Single Global Minimum



Connection to L10:

- We are still doing convex optimization — just with constraints now.
- The loss landscape is a bowl: one global minimum.

Lagrangian Formulation

Lagrange multipliers $a_i \geq 0$ convert constraints into penalties:

$$\max_{\alpha \geq 0} \min_{w,b} L(w, b, \alpha) \qquad L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i \left[y_i (w^\top x_i + b) - 1 \right]$$

Setting derivatives to zero:

- $\frac{\partial L}{\partial w} = 0 \quad \rightarrow \quad w = \sum_i \alpha_i y_i x_i$
- $\frac{\partial L}{\partial b} = 0 \quad \rightarrow \quad \sum_i \alpha_i y_i = 0$

Key: w is a linear combination of training points!

The Dual Problem

Substituting back gives the dual:

- maximize $\sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j \, y_i y_j \, x_i^\top x_j$
- subject to $\alpha_i \geq 0, \quad \sum_i \alpha_i y_i = 0$

Key Observations:

- Depends only on dot products $x_i^\top x_j$
- Most $a_i = 0$ at the solution → sparsity!
- Points with $a_i > 0$ are support vectors

The dot product observation leads to the kernel trick (later).

Interpreting the Solution

Prediction for new point x:

$$f(x) = \text{sign}\left(\sum_i \alpha_i y_i \, x_i^\top x + b\right)$$

- Points with $\alpha_i > 0$: support vectors (they define w)
- Points with $\alpha_i = 0$: irrelevant to the model

The SVM model is sparse:

- only a handful of training examples affect the decision.

Compare: logistic regression uses ALL training points.

Example: SVM from Scratch

Dataset (4 points in 2D):

- (+1): (1, 3), (2, 3)
- (−1): (1, 1), (2, 1)

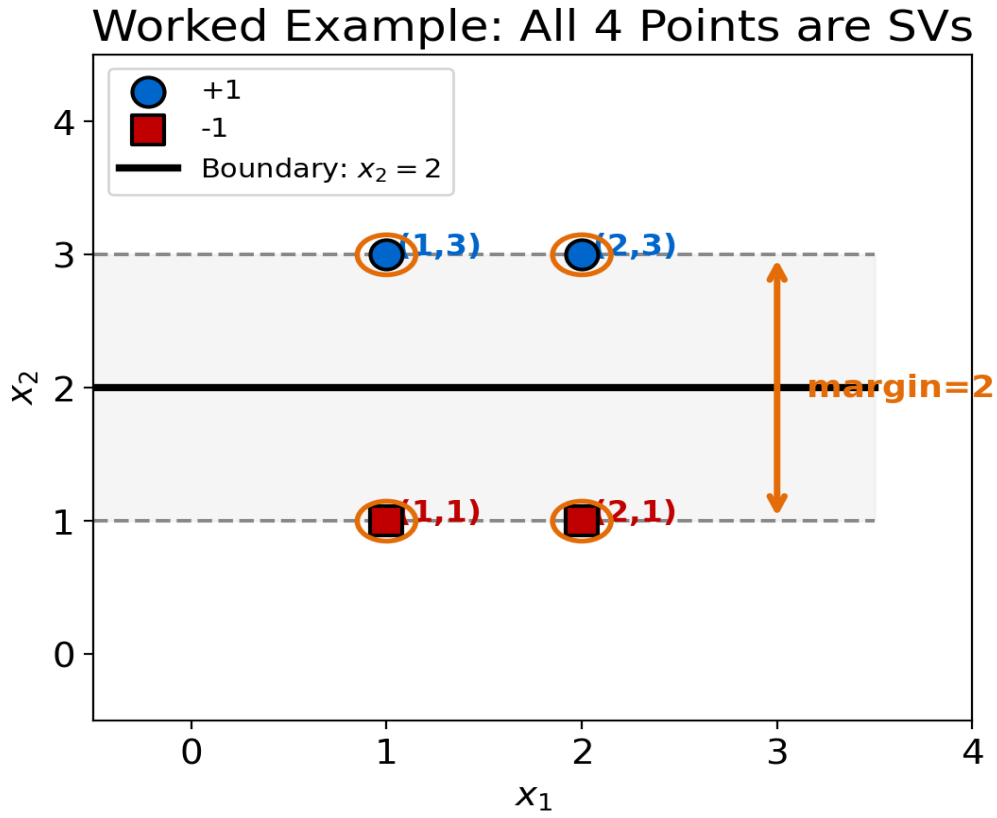
Step 1: Support vectors by inspection:

- Closest points: (1,3), (2,3) from +1 and (1,1), (2,1) from −1

Step 2: The optimal boundary is y = 2

w = [0, 1], b = −2

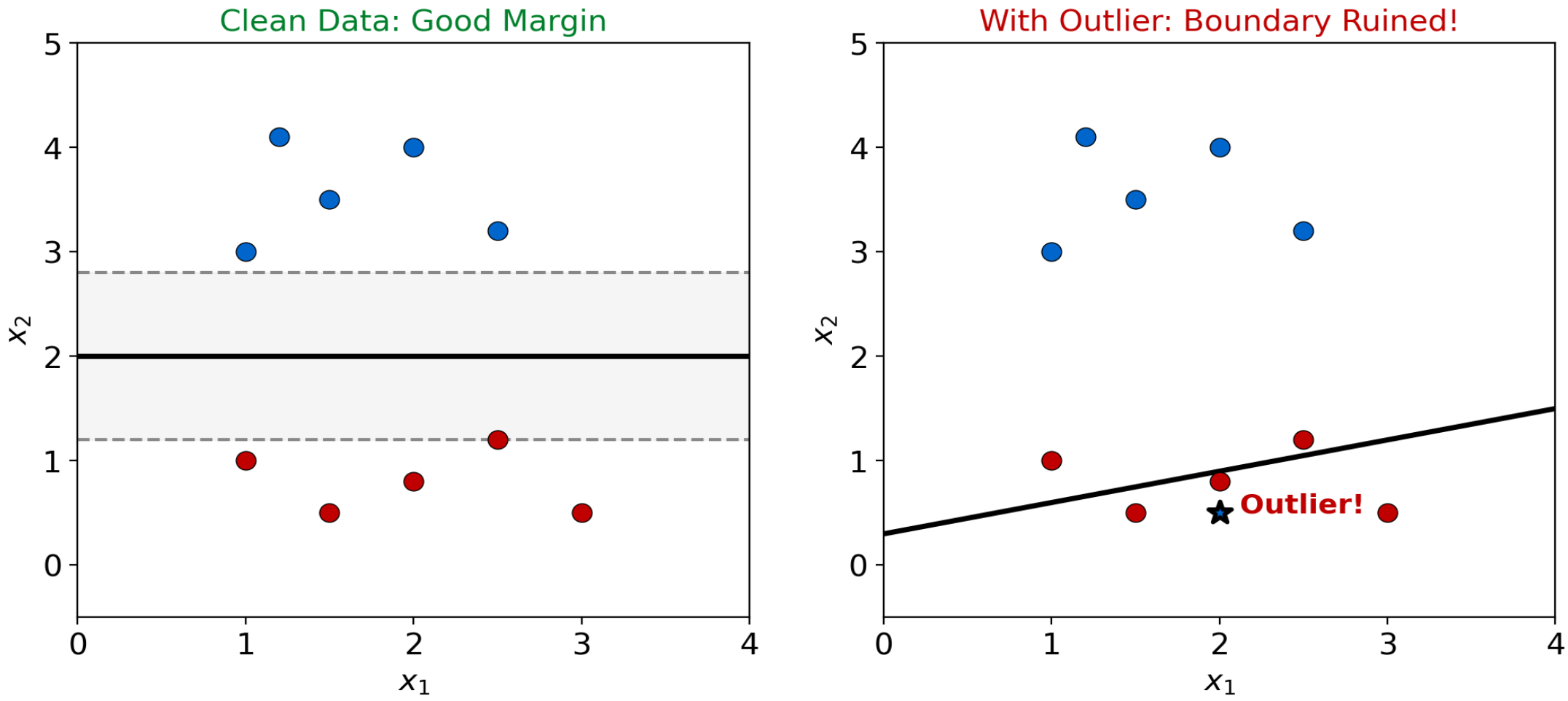
Step 3: Margin = 2/||w|| = 2/1 = 2



Step 4: Verify: $y_i(w^T x_i + b) = 1 \times (3 - 2) = 1 \geq 1$ ✓

2. Soft Margin & Hinge Loss

What if Data Isn't Separable?



🧠 Think: The Outlier Problem

- Imagine a perfectly separable dataset.
- Now add ONE outlier from class +1 deep in the −1 region.

What happens to the hard-margin SVM?

- Option A: The margin shrinks dramatically to fit the outlier
- Option B: No separating hyperplane exists at all

- Either way, it‘s bad. Should the model perfectly classify every training point?
- **Answer: No! Allow some mistakes to get a better boundary.**

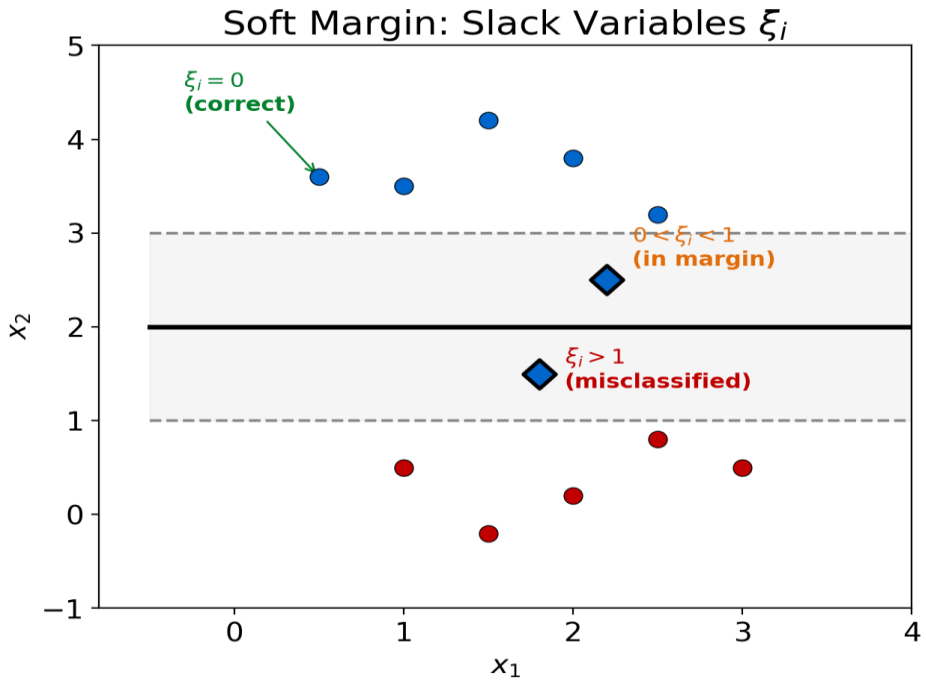
Soft Margin SVM

Introduce slack variables $\xi_i \geq 0$:

- New constraints: $y_i(w^T x_i + b) \geq 1 - \xi_i$

Interpretation of ξ_i:

- $\xi_i = 0$ correctly classified, outside margin
- $0 < \xi_i < 1$ correct side, but inside margin
- $\xi_i \geq 1$ misclassified!



ξ_i measures how much each point violates the margin.

Soft Margin Objective

Formulation:

- minimize $\frac{1}{2}\|w\|^2 + C\sum_i \xi_i$
- subject to $y_i(w^\top x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$

C controls the trade-off:

- Large C = strict teacher
  → penalizes errors heavily → narrow margin
- Small C = lenient teacher
  → allows errors → wider margin

Experiment: Effect of C

Before I show you — predict what happens!

- **C = 0.01 (very lenient)**

→ Wide margin, many violations allowed (underfitting?)

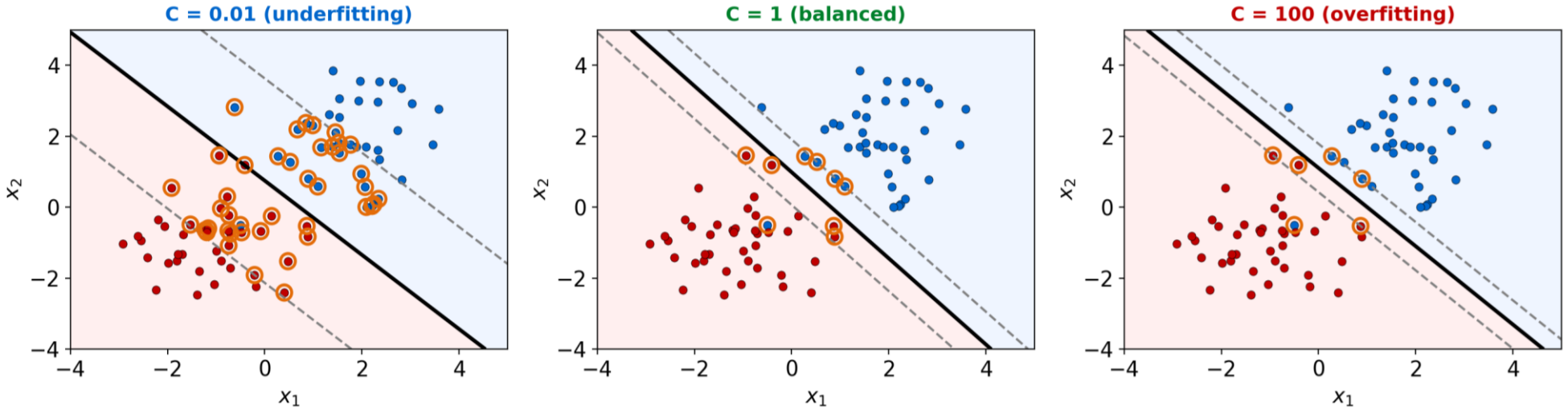
- **C = 1 (balanced)**

→ Good trade-off between margin width and accuracy

- **C = 100 (very strict)**

→ Narrow margin, almost no violations (overfitting?)

- **If C → ∞: hard margin SVM. If C → 0: ignores data entirely.**



Hinge Loss: The SVM Loss Function

Rewrite the soft margin objective:

$$\min \frac{1}{2}\|w\|^2 + C\sum_i \max\left(0,\ 1 - y_i(w^\top x_i + b)\right)$$

- **The $\max\left(0,\ 1 - y_i f(x_i)\right)$ term is the hinge loss.**

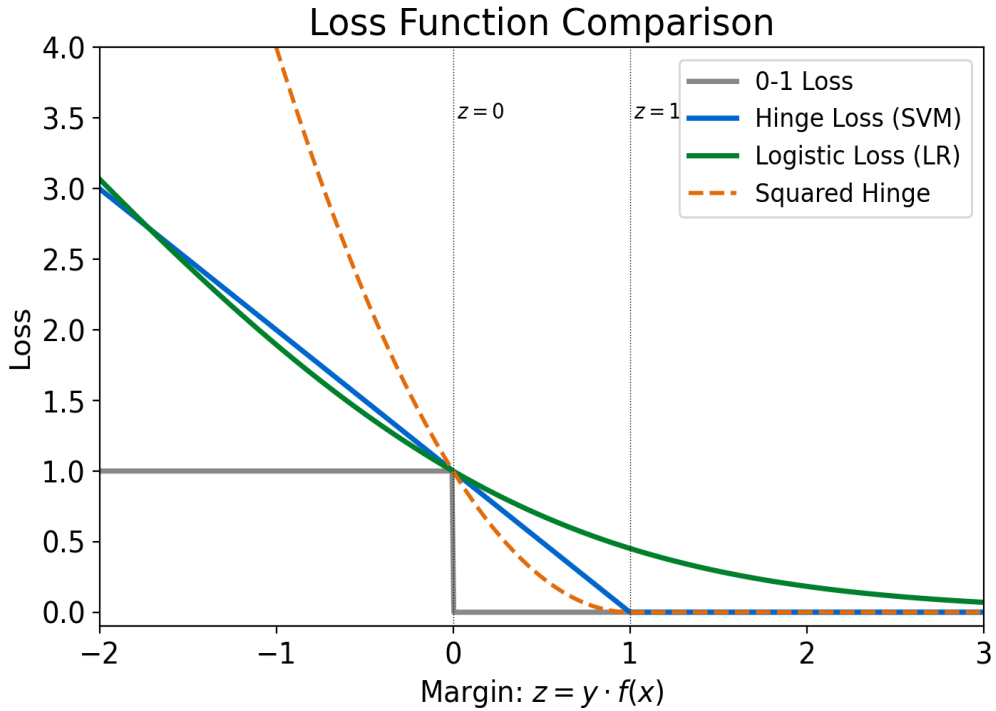
Properties:

- Zero loss when margin ≥ 1 (point is safely classified)
- Linear penalty for violations
- Convex (but not differentiable at margin = 1)

Loss Function Comparison

All losses plotted as functions of margin $z = y \cdot f(x)$:

- 0-1 Loss (L8): ideal but not differentiable
- Hinge Loss (SVM): convex, not smooth at $z = 1$
- Cross-Entropy (L9): smooth, always differentiable
- Squared Loss (L9): for regression



Hinge and logistic loss are both convex surrogates for 0-1 loss.

- SVM: $(1/2)||w||^2 + C \times$ hinge loss
- LR: cross-entropy loss

Same linear model, different loss functions!

SVM vs Logistic Regression

- **Both use the same linear model: $w^T\phi(x)$**
- Loss: SVM = hinge | LR = cross-entropy
- Output: SVM = class label | LR = probability
- Sparsity: SVM uses only support vectors | LR uses all data

- In practice, they often give very similar boundaries.

When to use which?

- Need probabilities? → Logistic Regression
- Need hard decisions + kernel trick? → SVM

3. The Kernel Trick

Beyond Linear: Nonlinear Data

- Linear SVM can only draw straight/flat boundaries.

- Many real problems need curves:
  - Concentric circles (inner class vs outer class)
  - XOR pattern
  - Spiral data

  A straight line fails completely on these.

- **Can we use feature engineering (L8) here?**

Feature Mapping to Higher Dimensions

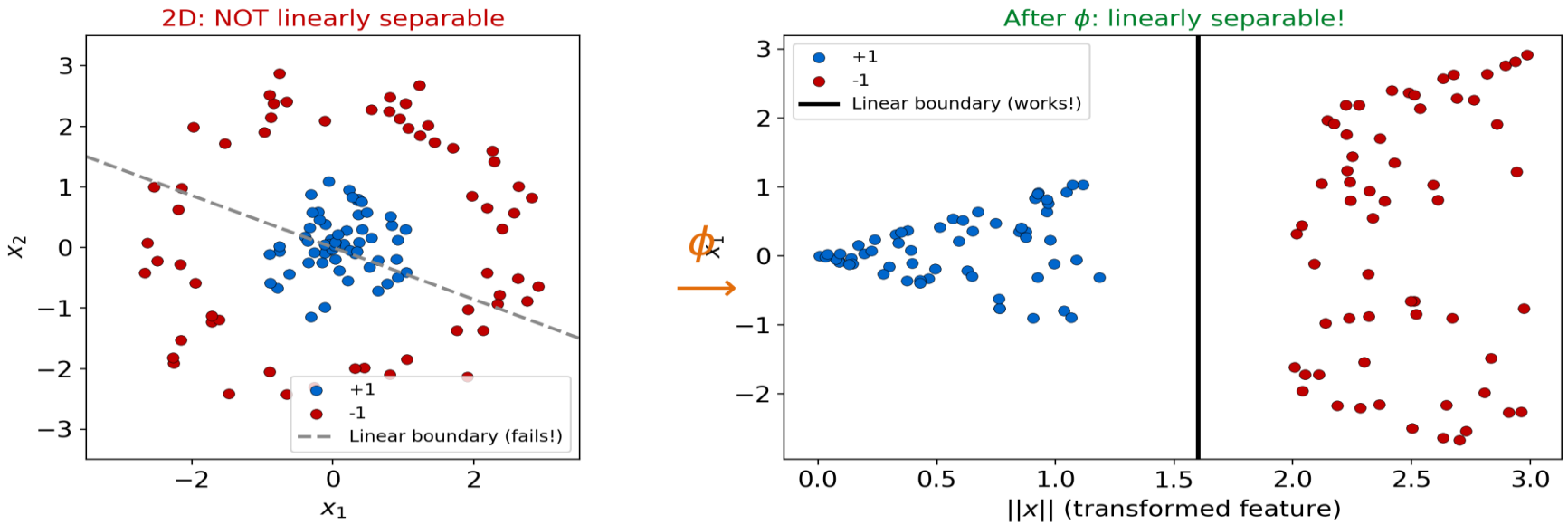
Idea from L8: map x → ϕ(x) into higher dimensions!

Example: $x = [x_1, x_2] \quad \rightarrow \quad \phi(x) = [x_1^2, \sqrt{2}\, x_1 x_2, x_2^2]$

- In this new 3D space, the data may become linearly separable!
- Then apply standard linear SVM on ϕ(x).

Problem: ϕ(x) can be very high-dimensional → expensive to compute

Can we avoid computing ϕ explicitly?



The Kernel Trick

- **Recall: the dual SVM uses only dot products** $\phi(x_i)^\top \phi(x_j)$
- A kernel function computes this directly:

$$K(x_i, x_j) = \phi(x_i)^\top \phi(x_j)$$

- **We never need to compute ϕ explicitly!**

Analogy:

- Like computing the area of a rectangle without separately measuring length and width.

Example: Polynomial Kernel

- **Let** $x = [x_1, x_2], \quad z = [z_1, z_2]$

  $K(x, z) = (x^\top z)^2 = (x_1 z_1 + x_2 z_2)^2$

- **Expand:**

  $K(x, z) = x_1^2 z_1^2 + 2 x_1 x_2 z_1 z_2 + x_2^2 z_2^2$

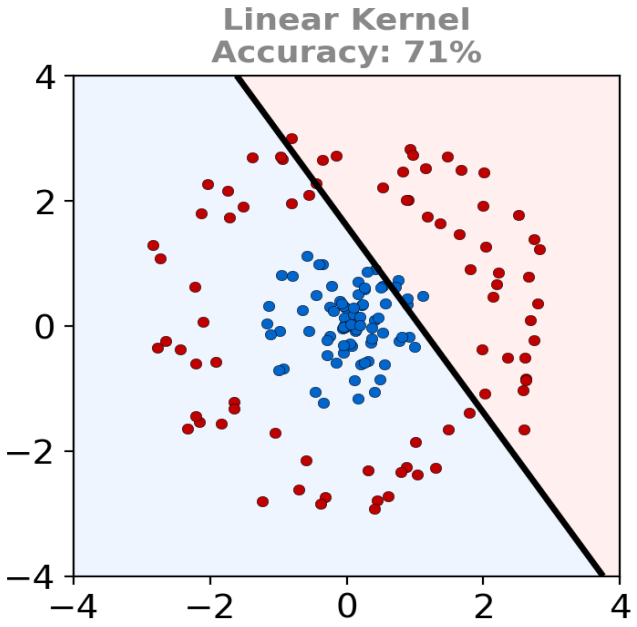
- **This equals $\phi(x)^T \phi(z)$ where:**

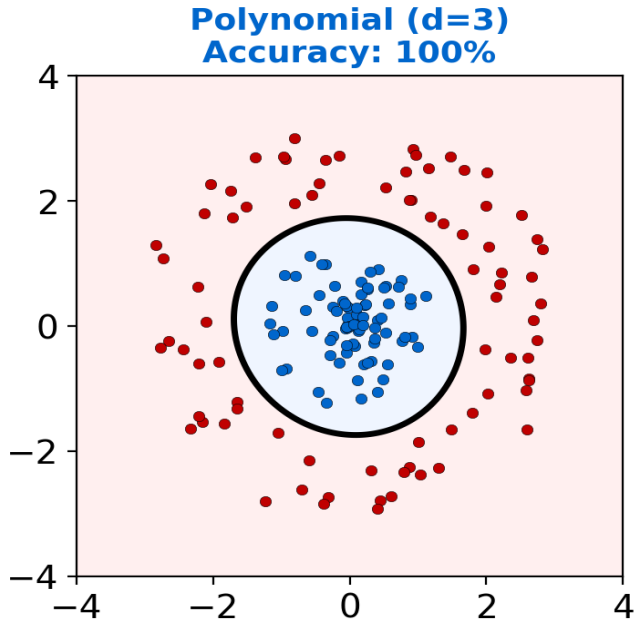
  $\phi(x) = [x_1^2, \ \sqrt{2}\, x_1 x_2, \ x_2^2]$

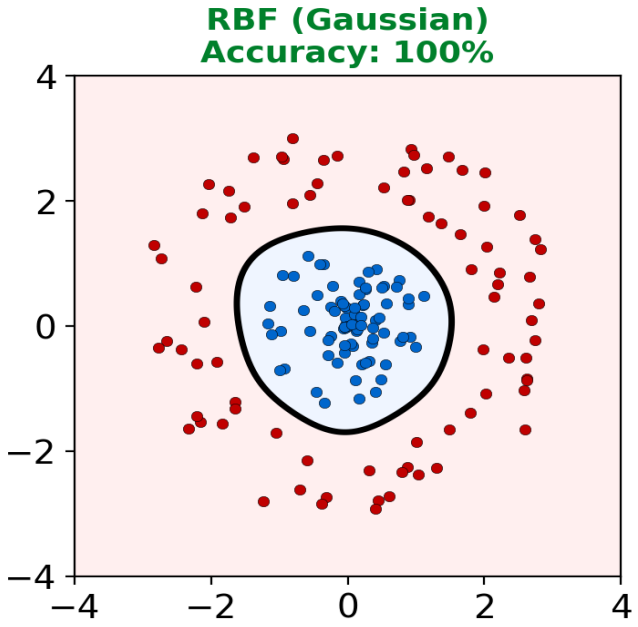
- **A 3D dot product computed with a single 2D operation!**
- No explicit mapping needed!

Common Kernel Functions

- **Linear:** $K(x, z) = x^\top z$
  - Standard linear SVM
- **Polynomial:** $K(x, z) = (x^\top z + c)^d$
  - Maps to degree-d feature combinations
- **RBF / Gaussian:** $K(x, z) = \exp\left(-\gamma \|x - z\|^2\right)$
  - Maps to infinite-dimensional space!
  - Most widely used kernel in practice
  - γ (gamma) controls how local each support vector's influence is.

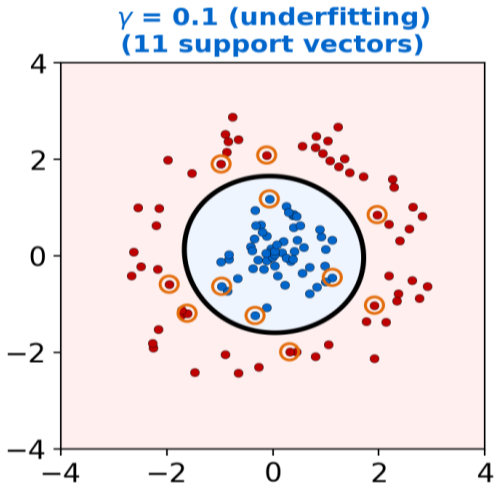


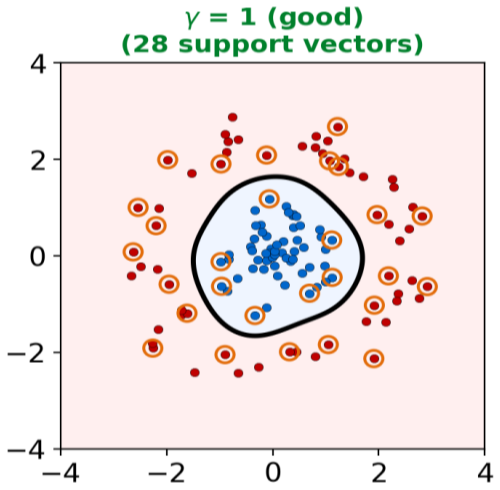


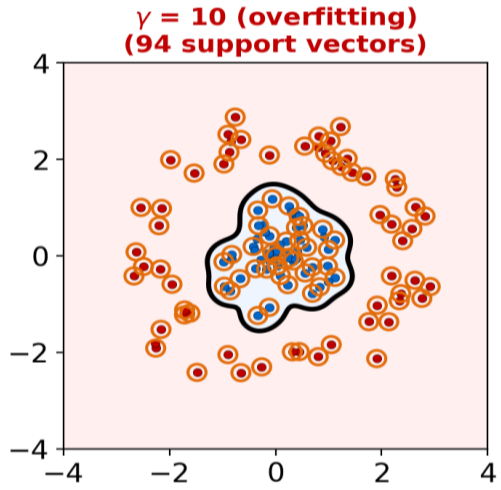


Experiment: Effect of γ in RBF Kernel

- **Small γ (e.g., 0.1)**
  - Each support vector influences a wide area → Smooth, simple boundary (underfitting risk)
- **Medium γ (e.g., 1)**
  - Balanced influence → good boundary
- **Large γ (e.g., 10)**
  - Each support vector only affects nearby points → Wiggly, complex boundary (overfitting risk)
- **Another bias-variance tradeoff! (Just like C)**
  - Which γ overfits? Which underfits? Discuss!

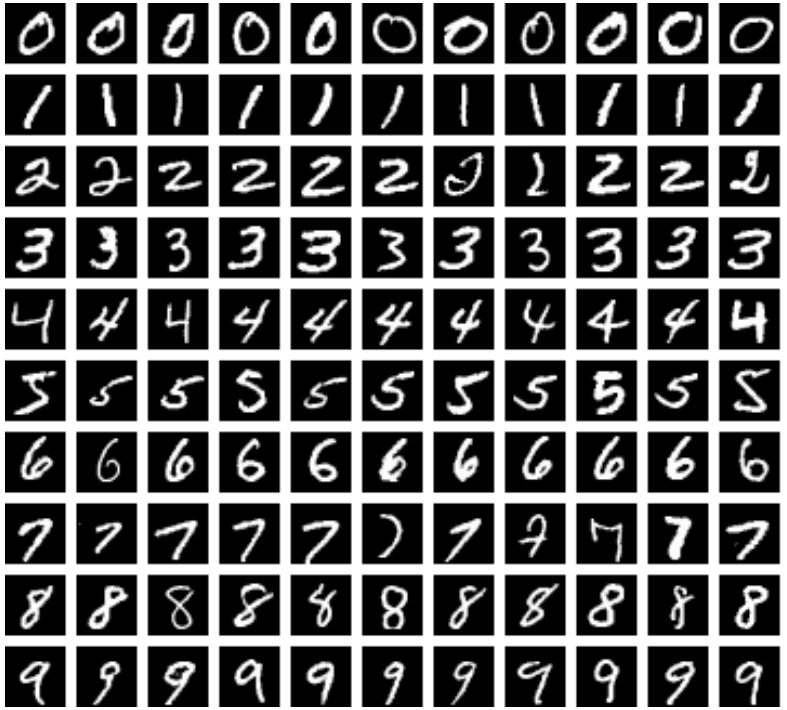






Case Study: Handwritten Digit Recognition

Classic ML benchmark: classify digits 0–9



MNIST dataset

Setup:

- Each image: 28 × 28 = 784-dimensional vector
- 10 classes (digits 0 through 9)

Results (late 1990s):

- SVM + RBF kernel: ~98.6% accuracy
- State of the art at the time!
- With just a kernel and support vectors — no deep learning.
- This was one of the applications that made SVMs famous.

Summary & Next Lecture

5 Key Concepts:

1. Margin — the gap between boundary and nearest points
2. Support Vectors — the critical points that define the boundary
3. Soft Margin (C) — allows misclassification for robustness
4. Hinge Loss — convex surrogate for 0-1 loss
5. Kernel Trick — nonlinear boundaries without explicit mapping

Next Lecture: Decision Trees & Ensemble Methods

- Can many weak learners beat one strong learner?