CS-470: Machine Learning

Week 6 - Support Vector Machines

Instructor: Dr. Sajjad Hussain

Department of Electrical and Computer Engineering
SEECS, NUST

October 16th, 2025

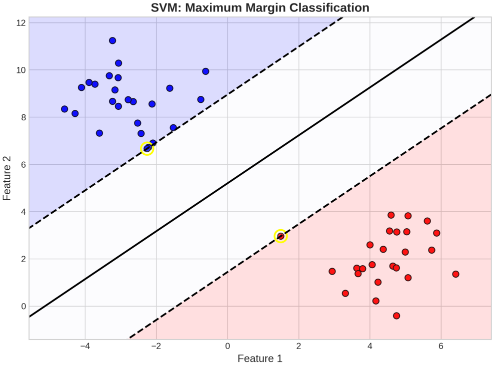
Learning Objectives

- Understand the intuition behind Support Vector Machines
- Learn how SVMs find the optimal decision boundary
- Differentiate between hard margin and soft margin SVMs
- Understand the kernel trick for non-linear data
- Learn about SVM applications in regression (SVR)
- Know how to choose appropriate SVM parameters

What is a Support Vector Machine (SVM)?

Support Vector Machines are like smart boundary-drawing tools that find the best possible line (or surface) to separate different classes of data.

Key Idea: Find the boundary that has the maximum "safety zone" (margin) on both sides.



Intuition: Why Maximize the Margin?

Think of the margin as a "safety buffer" around the decision boundary.

Larger margin means:

- More confident predictions (points are farther from the boundary)
- Better handling of slight data variations
- Improved performance on new, unseen data

Analogy: Driving in the middle of a wide road vs. driving on the edge - which is safer?

Why it matters:

- More robust to new, unseen data
- Less likely to be confused by random variations
- Often performs well with complex decision boundaries

Hard Margin SVM: Perfect World Scenario

When to use: When your data is perfectly separable with a straight line

The Goal: Find the line that:

- Correctly separates all points
- Has the maximum possible margin on both sides

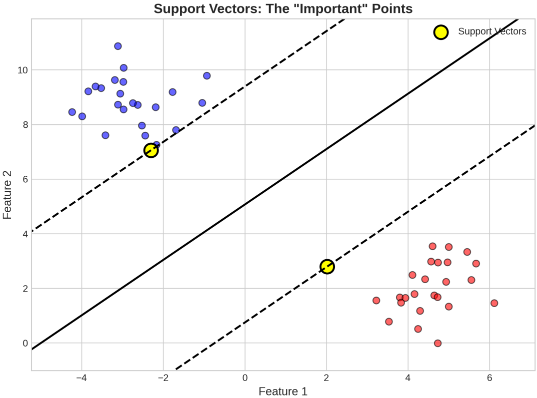
Mathematical View:

- We want to maximize the distance to the nearest points
- This turns out to be equivalent to minimizing $\|\mathbf{w}\|^2$
- All points must satisfy: $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$

Support Vectors: The Important Points

Support Vectors are the data points that actually define the boundary - they're like the "pillars" holding up the decision boundary. Only these points matter for the final decision boundary. If you remove all other points, you get the same boundary

Why they're useful: Makes the model efficient and interpretable.

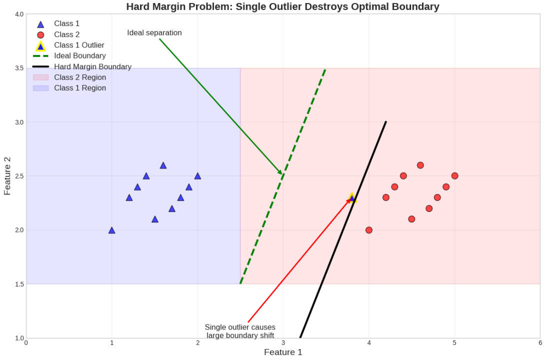


Limitations of Hard Margin SVM

Problem: Real-world data is rarely perfectly separable!

What goes wrong:

- A single outlier can drastically change the boundary
- Fails when classes overlap slightly



Soft Margin SVM: Handling Real-World Data

Idea: Allow some mistakes to get a more robust boundary.

How it works:

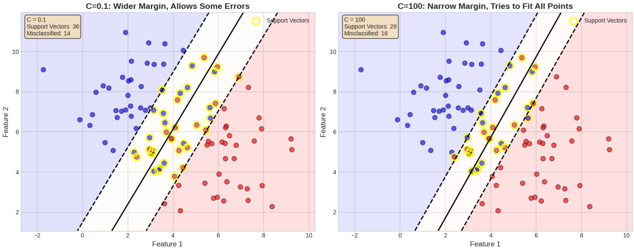
- We introduce "slack variables" that measure how much we violate the margin
- We balance between having a wide margin and allowing few errors
- The parameter C controls this trade-off

Analogy: Being strict vs. lenient when drawing boundaries between groups.

The C Parameter: Finding the Right Balance

What C controls: The trade-off between margin width and classification errors.

- **Large** C: Strict about errors, narrow margin (may overfit)
- **Small** C: Tolerant of errors, wide margin (may underfit)
- How to choose C: Use cross-validation and adjust based on performance.



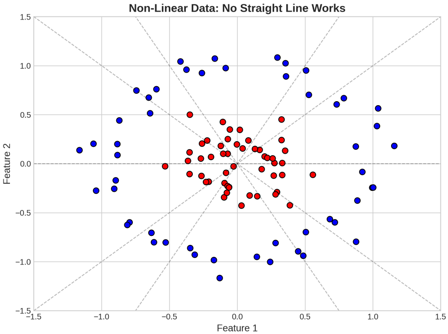
When Straight Lines Don't Work

Problem: Many real-world problems can't be separated with straight lines.

Examples:

- Circular patterns
- Complex, curved boundaries

Solution: Transform the data to a space where it becomes linearly separable.



Feature Mapping: Making Data Separable

Idea: Add new features that help separate the data.

Simple Example: 1D data that's not separable becomes separable when we add x^2 as a second feature.

Visualization: Imagine lifting points into a higher dimension where they can be separated by a flat plane.

Result: A curved boundary in the original space becomes a straight line in the new space.

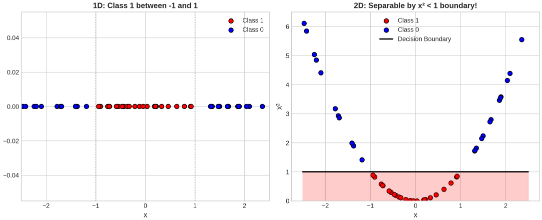
Example: From 1D to 2D for Better Separation

Scenario: Points along a line that can't be separated.

Transformation: Map x to (x, x^2) - now we have 2D data!

Result: In 2D space, we can draw a straight line that separates the classes perfectly.

Key Insight: Sometimes you need to look at data from a different "angle" to see the patterns.



The Kernel Trick: A Computational Shortcut

Problem: Explicit feature mapping can be computationally expensive.

Solution: Use kernel functions that give us the benefits of high-dimensional mapping without actually doing the transformation.

How it works: Kernels compute similarity between points in the high-dimensional space directly.

Popular Kernels:

- **Linear**: For simple, straight-line separation
- **Polynomial**: For curved, polynomial-shaped boundaries
- **RBF**: For complex, irregular boundaries

RBF Kernel: Measuring Similarity

Idea: Create features based on how similar each point is to "landmark" points.

Gaussian RBF Kernel:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right)$$

where:

- $\mathbf{x}, \mathbf{x}'$ are two data points
- $\|\mathbf{x} - \mathbf{x}'\|$ is the Euclidean distance between them
- γ controls the influence of each training example

How it works:

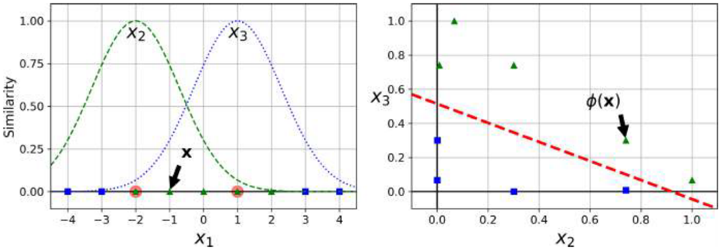
- Each landmark creates a "bump" of high similarity around it
- Points near landmarks get high values for those features
- The combination of these similarity features allows complex boundaries

RBF Example with Two Landmarks

Scenario: Using two strategic points as landmarks.
Transformation: Each point gets two similarity scores:

- Similarity to landmark 1
- Similarity to landmark 2

Result: In this 2D similarity space, we can now separate classes that were mixed in 1D.



Practical Considerations with RBF Kernel

Strengths:

- Very flexible - can handle complex boundaries
- Works well for many real-world problems

Challenges:

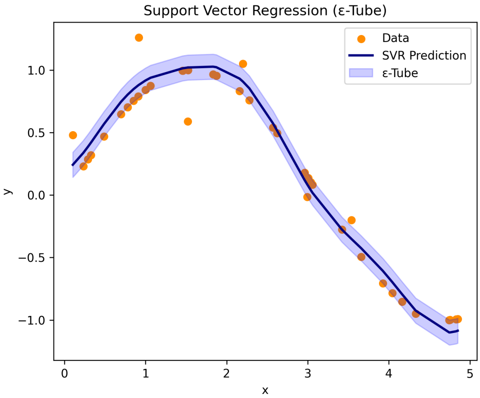
- Can be computationally expensive for large datasets
- Requires careful tuning of C and γ
- Risk of overfitting if parameters aren't chosen well

Tip: Always use cross-validation to choose the right parameters.

SVM for Regression: A Different Perspective

Idea: We want to predict continuous values while keeping the margin concept.

Key Concept: The ϵ-insensitive tube - we don't care about errors smaller than ϵ. This is more robust to outliers and focuses on getting the overall trend.



Understanding SVR Parameters

ϵ**:** Controls the width of the "don't care" zone

- Larger ϵ: More tolerant, simpler model
- Smaller ϵ: More precise, complex model

C**:** Same as in classification - balances fit vs. simplicity

When to use SVR:

- When you have outliers that might distort linear regression
- For non-linear relationships (with kernels)

Summary: Key Takeaways

- **Margin is key**: SVMs find boundaries with maximum safety zones
- **Flexibility matters**: Soft margin handles real-world imperfections
- **Parameters control behavior**: C balances margin vs. errors
- **Kernels add power**: Handle non-linear problems efficiently
- **Support vectors are efficient**: Only important points define the model
- **SVR extends the idea**: Apply margin concept to regression

Practical Advice:

- Start with linear SVM for simple problems
- Use RBF kernel for complex boundaries
- Always tune C (and γ for RBF) using cross-validation
- Consider computational cost for large datasets