Lecture 4: Maximal Margin Methods for Classification

Readings: ESL (Ch. 4, Ch. 12.1–12.3), ISL (Ch. 9), Bach (Ch. 4.1); code

Topics: hard-margin support vector classifiers (SVCs), soft-margin support vector classifiers (SVCs), the kernel trick and support vector machines (SVMs), SVMs in practice, appendix

1 Introduction

1.1 Direct Approaches: Construct Separating Hyperplanes

Recall that we consider a two-class problem with labels $y_i \in \{-1, 1\}$, and linear classifiers are of the form[1]

$$\hat{G}(x) = \text{sign}(x^T\beta + \beta_0)$$

The decision boundary is a hyperplane defined by $\{x : x^T\beta + \beta_0 = 0\}$ (see Exercise 1 for a refresher).

Definition 1.1 (Linear Separability)**.** A dataset $\{(x_i, y_i)\}_{i=1}^N$ with $y_i \in \{-1, 1\}$ is called linearly separable if there exists at least one hyperplane (β, β_0) that correctly classifies all input samples, i.e., such that

$$y_i(x_i^T\beta + \beta_0) > 0 \quad \text{for all } i.$$

When the data are separable, there are infinitely many such hyperplanes. How do we find one? And which one should we choose?

- The **Perceptron Learning Algorithm** finds *a* separating hyperplane, if one exists (last lecture).
- The **Optimal Separating Hyperplane** (Support Vector Classifier) finds the *best* one by maximizing the margin (this lecture).

2 Hard-Margin Support Vector Classifiers (SVCs)

2.1 Maximizing the Margin: The Separable Case

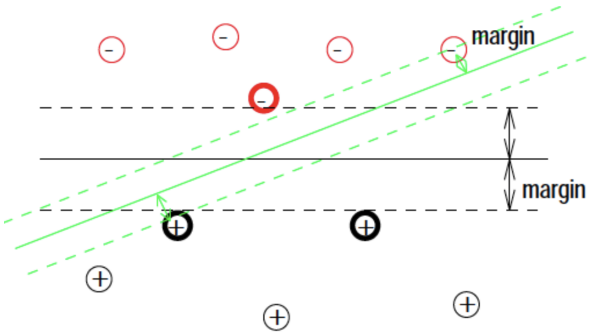
The optimal separating hyperplane (due to Vapnik) maximizes the **margin** (its distance to the closest point from either class):

$$M := \min_{i=1,\ldots,N} \frac{|\beta^T x_i + \beta_0|}{\|\beta\|}.$$

Why should we maximize the margin? The margin reflects the amount of noise the separator can tolerate, giving some amount of margin of safety to measurement errors. Larger margin

[1] We take the definition that $\text{sign}(x) = 1$ if $x > 0$ and $= -1$ if $x < 0$.

provides more robustness against noise (so that we can be more confident about the predictions) and tends to lead to better generalization on new data.



The black separating plane is better than the green one, because it has larger margin.

2.2 Finding the Fattest Separating Hyperplane

Maximizing the Margin (Separable Case)

We solve the constrained optimization problem:

$$\max_{\beta,\beta_0} \left\{ \min_{i=1,\ldots,N} \frac{|\beta^T x_i + \beta_0|}{\|\beta\|} \right\} \quad \text{subject to} \quad y_i \left(\beta^T x_i + \beta_0\right) > 0, \ \forall i.$$

This is equivalent (see blackboard[2]) to minimizing $\|\beta\|^2$ subject to scaled constraints:

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 \quad \text{subject to} \quad y_i(\beta^T x_i + \beta_0) \geq 1, \ \forall i. \tag{1}$$

This is a **convex optimization problem** (quadratic criterion with linear inequality constraints[3]).

The margin is then given by

$$M = 1/\|\beta\|$$

(also called thickness).

3 Soft-Margin Support Vector Classifiers (SVCs)

3.1 How About the Non-Separable Case?

Maximizing the Margin (Non-Separable Case)

To handle overlapping classes, we maximize M while allowing some points to be on the wrong side of the margin. Introduce the slack variables $\xi_i \geq 0$:

$$y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i, \quad \forall i,$$

[2] ⚠ In Equation (1), it is $\|\beta\|^2$ instead of the seemingly natural choice of $\|\tilde{\beta}\|^2$, where $\tilde{\beta} = (\beta, \beta_0)$.
[3] See Ch. 7.3 in `https://mml-book.github.io/` for a refresher.

and modify the optimization problem to:

$$\min_{\beta,\beta_0,\xi} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i \quad \text{s.t.} \quad \xi_i \geq 0,\ y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i\ \forall i. \tag{2}$$

- The parameter C is a cost parameter that controls the trade-off between maximizing the margin and the total error $\sum \xi_i$ (misclassifications occur when $\xi_i > 1$).
- As $C \to \infty$, it approaches the hard-margin case.
- See Exercise 2 for a problem equivalent to Equation (2).

3.2 Visualization

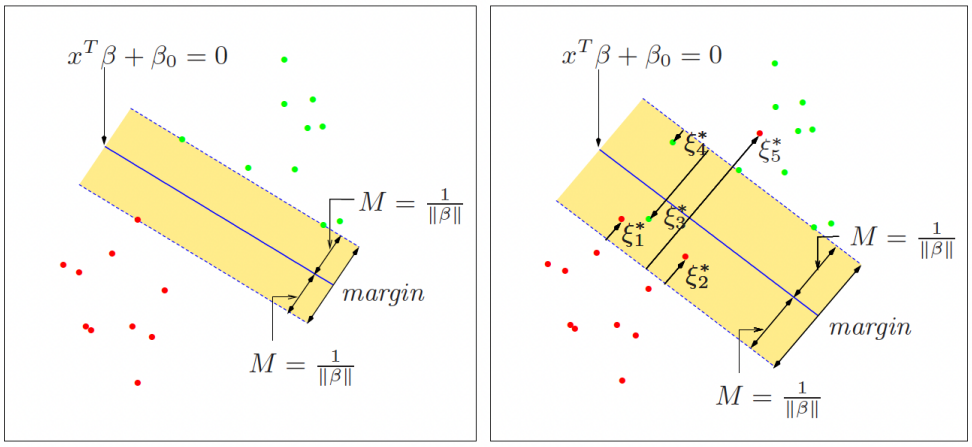


FIGURE 12.1. *Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width $2M = 2/\|\beta\|$. The right panel shows the nonseparable (overlap) case. The points labeled ξ_j^* are on the wrong side of their margin by an amount $\xi_j^* = M\xi_j$; points on the correct side have $\xi_j^* = 0$. The margin is maximized subject to a total budget $\sum \xi_i \leq$ constant. Hence $\sum \xi_j^*$ is the total distance of points on the wrong side of their margin.*

*The "margin" labeled in the plot is the margin width, which is twice our M.

3.3 The Lagrange Dual Problem

The soft-margin primal problem Equation (2) is still a convex optimization problem and admits a quadratic programming solution. We can solve it by moving to its dual form (see Appendix 1–2), which is often computationally simpler[4].

The Wolfe Dual Problem

Maximize

$$L_D(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N}\alpha_i\alpha_k y_i y_k\, x_i^\top x_k \tag{3}$$

[4]Typically solved with coordinate-descent algorithms such as sequential minimal optimization (SMO).

subject to

$$0 \leq \alpha_i \leq C, \ \forall i = 1, \ldots, N, \qquad \sum_{i=1}^{N} \alpha_i y_i = 0.$$

KKT conditions: for all $i = 1, \ldots, N$,

- *Stationarity:*

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i, \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \quad C - \alpha_i - \mu_i = 0.$$

- *Primal feasibility:*

$$y_i(\beta^\top x_i + \beta_0) \geq 1 - \xi_i, \qquad \xi_i \geq 0.$$

- *Dual feasibility:*

$$\alpha_i \geq 0, \qquad \mu_i \geq 0.$$

- *Complementary slackness:*

$$\alpha_i \big[y_i(\beta^\top x_i + \beta_0) - (1 - \xi_i) \big] = 0, \qquad \mu_i \xi_i = 0.$$

3.4 The Solution and Support Vectors

- The solution for the weight vector β is recovered from the optimal $\hat{\alpha}_i$ values:

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i.$$

- The coefficients $\hat{\alpha}_i$ from the dual solution determine the hyperplane. Points x_i for which $\hat{\alpha}_i > 0$ are called **support vectors**[5]. The KKT conditions reveal the role of each data point:
  - **Non-Support Vector ($\hat{\alpha}_i = 0$):**
    The point is correctly classified with a margin to spare ($y_i f(x_i) > 1$).
  - **Margin Support Vector ($0 < \hat{\alpha}_i < C$):**
    The point lies exactly on the margin ($y_i f(x_i) = 1$), and $\hat{\xi}_i = 0$.
  - **Bounded Support Vector ($\hat{\alpha}_i = C$):**
    The point violates the margin ($y_i f(x_i) \leq 1$). It lies inside the margin or is misclassified ($\hat{\xi}_i > 0$).
- Any of these margin points ($\hat{\alpha}_i > 0, \hat{\xi}_i = 0$) can be used to solve for β_0, i.e., by solving

$$\beta^T x_i + \beta_0 = 1$$

  for any margin support vector x_i. Typically we use an average of all the solutions for numerical stability.
- The final decision function is:

$$\hat{G}(x) = \text{sign}\left(\sum_{i=1}^{N} \hat{\alpha}_i y_i x^T x_i + \hat{\beta}_0 \right) \tag{4}$$

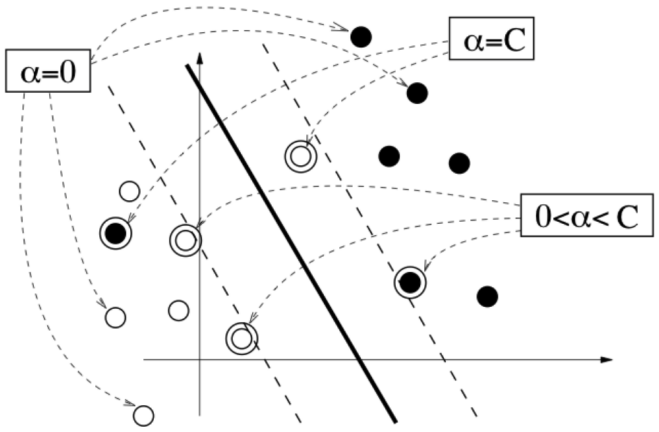
  (derive a formula for $\hat{\beta}_0$). We call the model a **support vector classifier (SVC)**.

[5]Thus the optimal hyperplane is determined entirely by a subset of training points near the decision boundary, making the solution sparse and focused on the most informative samples.

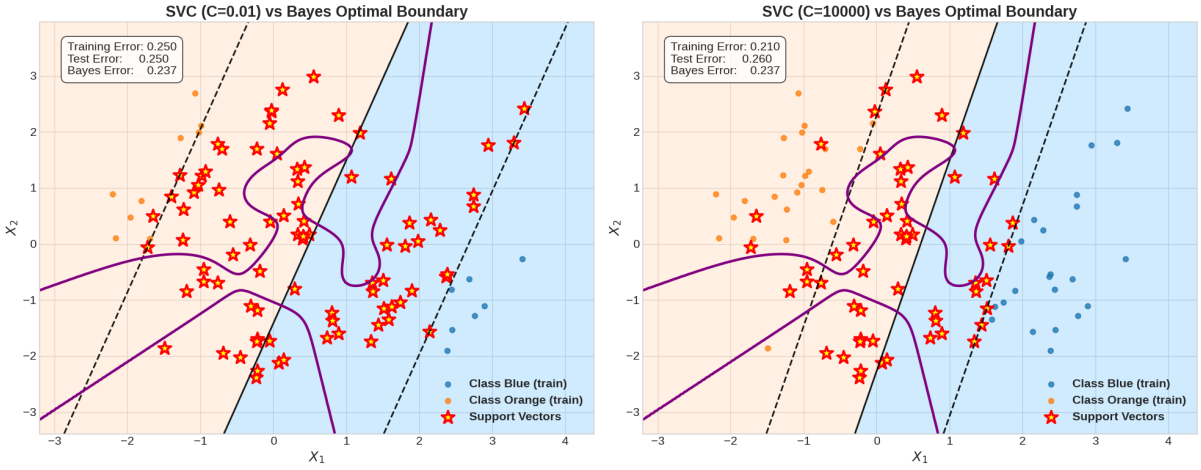
3.5 Visualization in 2D

Below is the separating hyperplane learned by an SVC to separate the black (-1) and white $(+1)$ circles. Shown are their corresponding Lagrange multipliers α.

Each point correctly classified with large confidence ($y_i f(x_i) > 1$) has a null multiplier. Other points are called support vectors. They can be on the boundary, in which case the multiplier satisfies $0 \leq \alpha \leq C$, or on the wrong side of the boundary, in which case $\alpha = C$.



3.6 SVC: The Role of the Cost Parameter



- **Low C – High Regularization:**
  - Creates a **wide margin** by tolerating misclassifications.
- **High C – Low Regularization:**
  - Creates a **narrow margin** to minimize training errors.

In practice, the optimal value of C is estimated by cross-validation.

See Exercise 3 for a fun exploration.

3.7 Optimal Separating Hyperplanes: Hard vs. Soft Margin

1. The Hard-Margin Classifier

- **Assumption:** The training data is perfectly, linearly separable.
- **Objective:** Find the hyperplane with the maximum possible margin. No violation of the margin condition is allowed.
- **Formulation:**

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to:

$$y_i(x_i^T\beta + \beta_0) \geq 1, \ \forall i.$$

- **Limitation:** Extremely sensitive to outliers and will fail if the data are not separable.

2. The Soft-Margin Classifier

- **Assumption:** The data may not be perfectly separable.
- **Objective:** Find a balance between a large margin and a low rate of margin violations.
- **Formulation:**

$$\min_{\beta,\beta_0,\xi} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i$$

subject to:

$$y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \ \xi_i \geq 0.$$

- **Robustness:** The cost parameter C controls the trade-off, making the classifier robust to outliers.

3.8 Perceptron vs. SVCs: Robustness to Noise

- **Data:** Generate a small, linearly separable two-class dataset.
- **Step 1:** Fit three models on clean data:

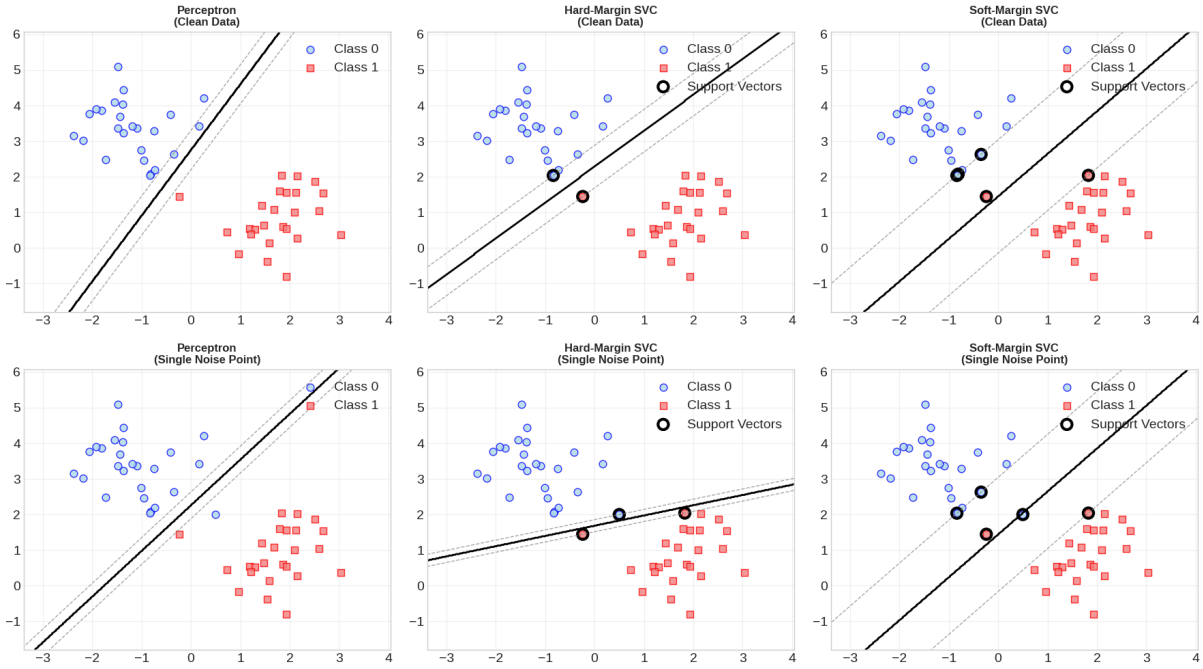
  (1) Perceptron (finds any separating hyperplane),

  (2) Hard-Margin SVC ($C = 10^6$, maximizes margin),

  (3) Soft-Margin SVC ($C = 1.0$, allows some misclassifications).

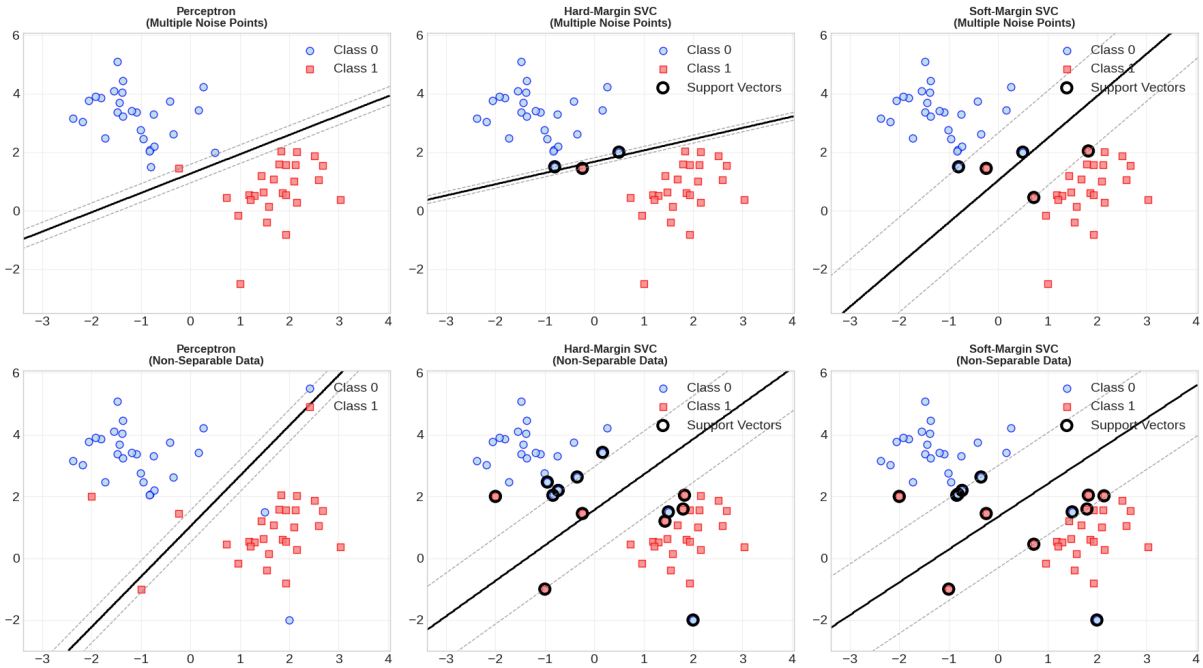
- **Step 2:** Test robustness under three noise scenarios:
  - Add a single noisy data point near the decision boundary
  - Add multiple noisy outlier points in different regions
  - Create non-separable data by adding points with conflicting class labels
- **Step 3:** Re-fit all models on each noisy dataset and compare:
  - Decision boundary stability and changes
  - Support vector identification (for SVCs)
  - Margin visualization and robustness
  - Classification accuracy on each scenario

3.9 Empirical Results



The soft-margin classifiers (particularly SVMs; see Lecture 5) are still often used in certain applied domains (e.g., bioinformatics, text classification, small/medium-scale tabular problems).

3.10 Empirical Results (continued)



⚠ For hard-margin SVC, all support vectors are on the margin boundary. For non-separable data, the optimization problem has no solution. What we did is simulating a soft-margin SVC with a large C (not a true hard-margin SVC). In the fifth plot, you are seeing a soft-margin SVC under pressure. The fact that it produces extra support vectors is the visual proof that a perfect hard-margin solution is impossible there.

3.11 A Unifying View: Loss + Penalty

The classification and regression models we have seen so far can be understood through a single, powerful framework.

The "Loss + Penalty" Framework: find the function f that minimizes

$$\frac{1}{N}\sum_{i=1}^{N}\underbrace{L(y_i, f(x_i))}_{\text{Loss Function}} + \lambda \underbrace{J(f)}_{\text{Penalty Term}} . \tag{5}$$

Model	**Loss Function** $L(y, f(x))$	**Penalty** $J(f)$
Ridge Regression	$(y - f(x))^2$	$\|\beta\|_2^2$
Lasso Regression	$(y - f(x))^2$	$\|\beta\|_1$
Logistic Regression[6]	$\log(1 + e^{-yf(x)})$	$\|\beta\|_2^2$ (Ridge) or $\|\beta\|_1$ (Lasso)
SVC	$[1 - yf(x)]_+$ (Hinge Loss)	$\|\beta\|_2^2$

So far, we have restricted f to the space of linear functions. In future lectures, we will study this framework for f belonging to other function spaces.

Thinking of using squared loss for classification? See Exercise 4.

4 The Kernel Trick and Support Vector Machines (SVMs)

4.1 SVCs (Linear Decision Boundaries): An Observation

The soft-margin SVC finds an optimal linear boundary by solving:

$$\min_{\beta,\beta_0,\xi} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i$$

subject to

$$y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0.$$

The Dual Problem

The solution is found by solving the Wolfe dual problem for coefficients α_i:

$$\max_{\alpha} \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N}\alpha_i\alpha_k y_i y_k \langle x_i, x_k\rangle$$

subject to

$$0 \leq \alpha_i \leq C \qquad \text{and} \qquad \sum_{i=1}^{N}\alpha_i y_i = 0.$$

The decision function is

$$f(x) = \text{sign}\left(\sum_{i=1}^{N}\alpha_i y_i \langle x, x_i\rangle + \beta_0\right).$$

Notice that both the optimization and the solution depend only on **inner products** $\langle x_i, x_k\rangle$.

4.2 From Linear to Nonlinear Decision Boundaries

⚠ Real-world decision boundaries are rarely linear.

In principle, we can construct a linear boundary with SVCs in a higher dimensional, nonlinearly transformed version of the feature space, leading to a non-linear boundary in the original space.

However, we pay a price to get this sophisticated boundary in terms of a tendency to overfit...

Building on our inner product observation earlier, how can we extend the SVCs to handle non-linear boundaries while reducing overfitting (or at least better ways to navigate the bias-variance tradeoff)?

4.3 The Kernel Trick (More on This in Lecture 6)

💡 Transform the inputs x with a suitable feature map $h : \mathbb{R}^d \to \mathcal{H}$ and find a trick to efficiently run SVCs in very high dimensional spaces (e.g., $\mathcal{H} = \mathbb{R}^D$ for $D > d$) or even infinite-dimensional spaces.

The Inner Products Observation

With the nonlinear features in place, the dual problem and the final decision function depend on the features only through inner products $\langle h(x_i), h(x_k)\rangle$.

- **Dual Objective:**

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle h(x_i), h(x_k)\rangle.$$

- **Decision Function:**

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i)\rangle + \beta_0.$$

Definition 4.1 (The Kernel Trick)**.** Instead of explicitly computing the features $h(x)$ in a very high-dimensional space, we only need to specify a **kernel function**

$$k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R},$$

which computes the inner product in feature space, while operating directly on the original inputs:

$$k(x, x') = \langle h(x), h(x')\rangle. \tag{6}$$

4.4 Popular Kernel Functions and Efficiency of the Kernel Trick

- **p-th Degree Polynomial:**

$$k(x, x') = (1 + \langle x, x'\rangle)^p$$

  - Feature space is $\mathbb{R}^D$, with

$$D = \binom{d+p}{p}$$

(huge for large d, p).

- Example: $d = 1000$, $p = 3 \implies D \approx 1.67 \times 10^8$.
- Kernel trick: compute in $O(d)$ time instead of $O(D)$.

- **Radial Basis Function (RBF) Kernel:**

$$k(x, x') = \exp(-\gamma \|x - x'\|^2).$$

This corresponds to an *infinite-dimensional* feature space (more details later in Lecture 6). The parameter γ controls the width of the kernel.

- **Neural Network Kernel:**

$$k(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2).$$

This kernel is related to two-layer neural networks.

⚠ Strictly speaking, the kernel $k(x, x')$ must be a *positive definite* function (i.e., the Gram matrix K, with entries $K_{ij} = k(x_i, x_j)$, is symmetric and PSD); see Lecture 6. For certain choices of (κ_1, κ_2), the tanh kernel is positive definite, but not universally.

4.5 The Support Vector Machine (SVM)

Definition 4.2 (The Support Vector Machine)**.** The SVM is the non-linear classifier obtained by replacing all inner products $\langle x_i, x_k \rangle$ in the SVC with a chosen kernel function $k(x_i, x_k)$.

SVM Dual Problem[a]

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k k(x_i, x_k)$$

subject to

$$0 \leq \alpha_i \leq C \qquad \text{and} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0.$$

[a] It is this dual view of the SVM that allows us to use the kernel trick! :)

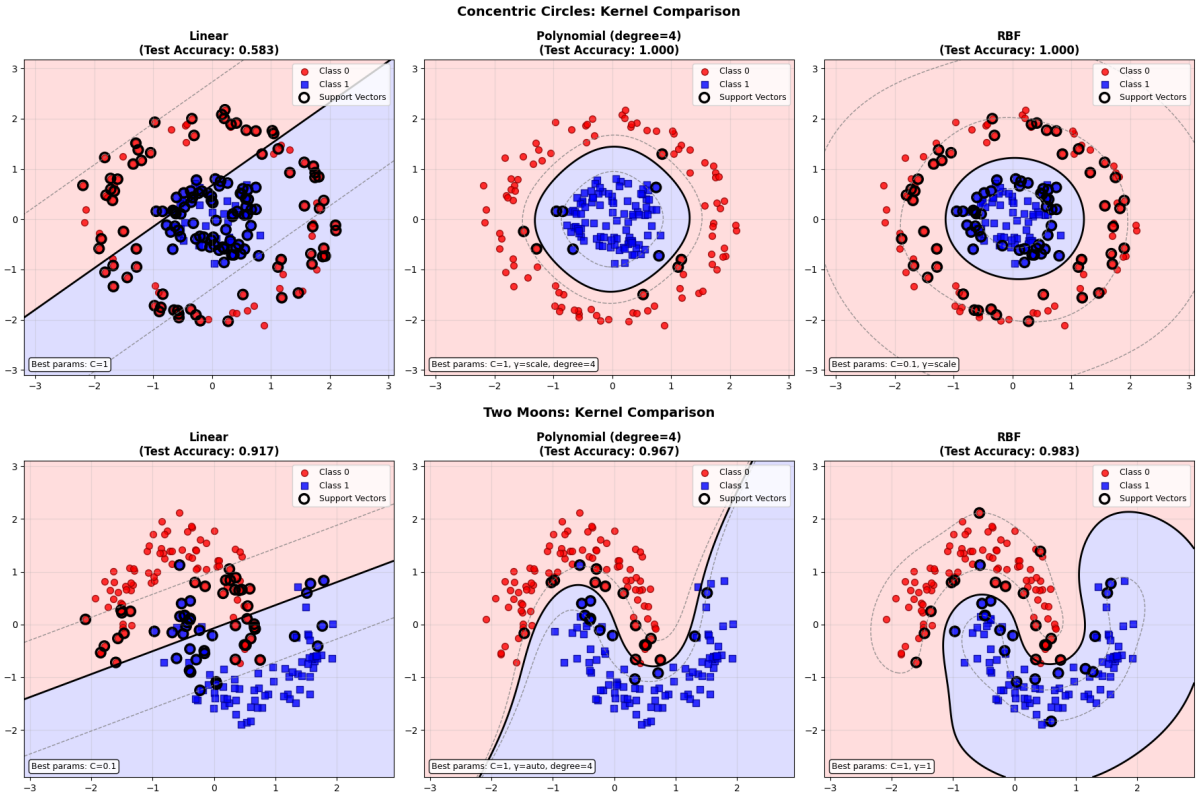
The final classifier is a non-linear function of x:

$$\hat{f}(x) = \text{sign}\left(\sum_{i=1}^{N} \hat{\alpha}_i y_i k(x, x_i) + \hat{\beta}_0 \right), \tag{7}$$

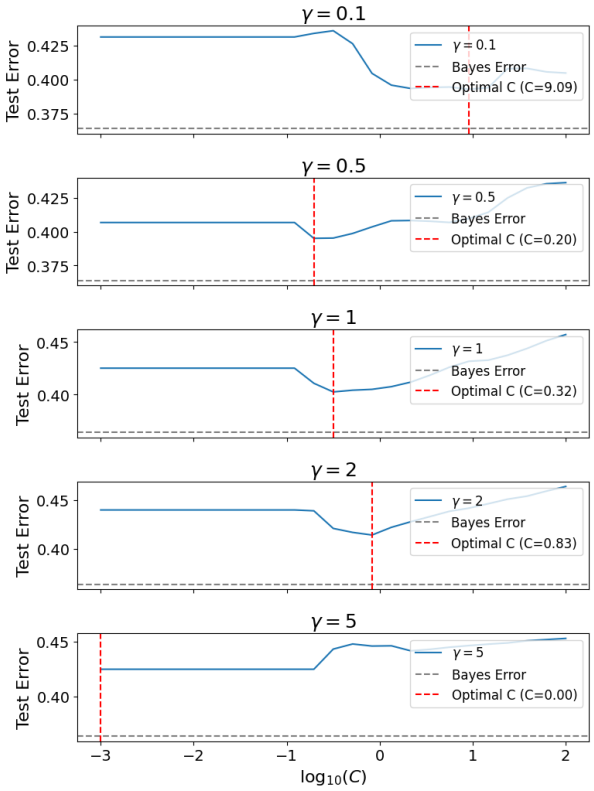
where the sum is only over the support vectors (observations for which $\hat{\alpha}_i > 0$).

5 SVMs in Practice

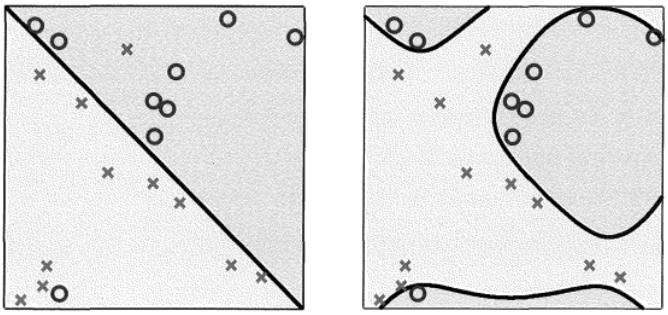
5.1 Example: SVMs and the Choice of Kernels



5.2 Example: SVMs with RBF Kernel of Different Widths



5.3 Should We Always Use Nonlinear Kernel Functions?



(a) Linear fit (b) 4th order polynomial fit

The training samples are not linearly separable. A fourth order polynomial separates all the points (training error is zero) but a line separates the data after omitting a few points (outliers).

How about misclassification error on test samples? Bias-variance tradeoff in play.

5.4 Summary: LDA vs. Logistic vs. SVC ($\mathcal{Y} = \{-1, +1\}$)

Behavior of classifiers when the classes are further apart:

- LDA: Boundary roughly at midpoint; probabilities extreme.
- Logistic Regression: Confidence increases smoothly; probabilities approach 0/1.
- SVC: Margin increases; only support vectors matter; wide margin if separable.

Aspect	**LDA**	**Logistic Regression**	**Soft-Margin SVC (Hinge)**
Approach	Generative	Probabilistic/Discriminative	Direct (nice geometric view[7])
Loss	NLL of a Gaussian model	Logistic (smooth, convex)	Hinge (piecewise linear, non-smooth)
Optimization	Closed-form	Newton or SGD	Quadratic programming (SMO)
Complexity	Extremely fast	Scales well with N	Moderate to high
Probabilities?	Yes (Bayes' rule)	Yes (sigmoid)	No (needs calibration)
Margin	None	Implicit "soft" margin	Explicit maximum margin
Data assumptions	Gaussian with equal covariance	None	None
Regularization	Implicit (shrinkage possible)	Explicit (ℓ_1, ℓ_2)	Explicit ($\|\beta\|^2$, C)
Best for	Small data, Gaussian-like, very fast	Large-scale, probabilities, interpretability	Medium data, robust margin, high-D sparse

We can apply kernel tricks to enhance the performance of all three methods! We will formalize the notion of kernels and kernel trick, and unify all these under the neat framework of **kernel methods** in Lecture 6. But before that we will look at more examples of nonlinear transformations in next lecture.

6 Exercises

1. Consider the hyperplane

$$L = \{x \in \mathbb{R}^p : \beta^T x + \beta_0 = 0\},$$

where $\beta \in \mathbb{R}^p, \beta_0 \in \mathbb{R}$ are the parameters defining L.

(a) Show that the orthogonal distance from any point x_i to the hyperplane L is given by

$$\frac{|\beta^T x_i + \beta_0|}{\|\beta\|}.$$

(b) For a classification problem with labels $y_i \in \{-1, 1\}$, a point (x_i, y_i) is misclassified if $y_i(\beta^T x_i + \beta_0) \leq 0$. Show that for such a point, the distance from x_i to the hyperplane is

$$\frac{-y_i(\beta^T x_i + \beta_0)}{\|\beta\|}.$$

2. For $f(x) = x^T\beta + \beta_0$, consider the optimization problem:

$$\min_{\beta_0,\beta} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \frac{\lambda}{2}\|\beta\|^2, \tag{8}$$

where "+" indicates positive part.

(a) Show that the solution to the above problem (with $\lambda = 1/C$) is the same as that for the standard SVM formulation Equation (2).

(b) What are the Bayes predictor and Bayes error under the hinge loss?

3. Consider a dataset with three samples in $\mathbb{R}^2$, given by

$$x_1 = (0, 0), \qquad x_2 = (0, -1), \qquad x_3 = (-2, 0)$$

with corresponding labels $y_1 = y_2 = -1$, $y_3 = 1$.

(a) Determine the maximal margin hyperplane and its parameters (β, β_0), identify the support vectors, and calculate the margin width.

(b) Now add a point $x_4 = (3, 2)$ to class $+1$ and answer (a) to characterize the hard-margin SVC solution for this new 4-point dataset.

(c) For a soft-margin SVC with a small C, a possible solution is the original hyperplane from part (a). For this solution, calculate the slack variable ξ_4 required for the misclassified point x_4.

(d) The total cost for a soft-margin solution is

$$\frac{1}{2}\|\beta\|^2 + C\sum \xi_i.$$

Compare the solutions from (b) and (c), and explain how a sufficiently small C would lead the optimizer to prefer the solution in (c) over the one in (b).

4. Consider binary classification with labels $Y \in \{\pm 1\}$. A loss $L(y, f)$ is *margin-maximizing* if, whenever
$$\eta(x) := P(Y = 1|X = x) \neq \tfrac{1}{2},$$
there exists a conditional risk minimizer
$$f^*(x) \in \arg\min_f \{\eta(x)L(1, f) + (1 - \eta(x))L(-1, f)\}$$
such that (i) $\text{sign}(f^*(x)) = \text{sign}(2\eta(x) - 1)$ (correct classification) and (ii) $|f^*(x)| \geq 1$ (minimum confidence).

   Determine whether the loss functions in (a)–(c) below are margin-maximizing:

   (a) Squared Loss: $L(y, f) = (y - f)^2$,

   (b) Hinge Loss: $L(y, f) = [1 - yf]_+$,

   (c) Logistic Loss: $L(y, f) = \log(1 + e^{-yf})$.

   (d) A loss $L(y, f)$ is *calibration-prioritizing* if, whenever $\eta(x) \neq \frac{1}{2}$, the conditional risk minimizer $f^*(x)$ is a strictly monotonic function of $\eta(x)$ and satisfies $\text{sign}(f^*(x)) = \text{sign}(2\eta(x) - 1)$. For the losses in (a)–(c), determine whether they are calibration-prioritizing.

5. **Kernels and Linear Discriminant Analysis.** Solve Exercise 12.10 in ESL.

[Experimental] Let us use the same setup from the experimental exercise in Lecture 3. Generate a 2D synthetic dataset with two classes ($Y = 0$ and $Y = 1$), each following a multivariate Gaussian distribution:
$$X|Y = 0 \sim \mathcal{N}(\mu_0, \Sigma_0), \qquad X|Y = 1 \sim \mathcal{N}(\mu_1, \Sigma_1),$$
where
$$\mu_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \Sigma_0 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix},$$
$$\mu_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \quad \Sigma_1 = \begin{bmatrix} 1 & -0.3 \\ -0.3 & 1 \end{bmatrix}.$$
Generate 200 samples per class and then split the data into training (70%) and test sets (30%).

1. Implement a classification model using: SVC (with linear kernel) and SVMs using different kernel functions. Fit each model on the training data and compute the misclassification error on the test data.
2. Plot the decision boundaries of each model together with the test data points.
3. Find the optimal Bayes classifier for this synthetic setting and compare its misclassification error on the test set to the models above.
4. Discuss how the choice of kernel and the cost parameter C affect their performance relative to the Bayes classifier. Explore other settings such as when the ground truth class means are further apart.

A Appendix

A.1 Appendix 1: A Brief Review of Constrained Optimization

Minimizing a differentiable function $F : \mathbb{R}^m \to \mathbb{R}$ is often done by setting its gradient ∇F to zero to find stationary points. To check minimality:

- Use higher-order derivatives (e.g. Hessian test), or
- Use convexity: if F is convex, any stationary point is a global minimum.

Now we consider the equality constrained optimization problem:

$$\min_{z \in \mathbb{R}^m} F(z) \quad \text{subject to } G(z) = 0,$$

for some differentiable constraint function $G : \mathbb{R}^m \to \mathbb{R}$.

- At an optimal point $\bar{z}$, ∇F and ∇G must be parallel[8], i.e.:

  $$\nabla F(\bar{z}) + \mu \nabla G(\bar{z}) = 0,$$

  for some $\mu \neq 0$ (the Lagrange multiplier).

- Define the **Lagrangian**:

  $$L(z, \mu) = F(z) + \mu G(z),$$

  then $(\bar{z}, \mu)$ must be a stationary point of L.

A.2 Inequality Constraints

- What if we replace equality by inequality constraint:

  $$G(z) \leq 0?$$

  There are two cases:

  - Interior: $G(\bar{z}) < 0 \implies$ constraint inactive, $\nabla F(\bar{z}) = 0$.
  - Boundary: $G(\bar{z}) = 0 \implies$ equality constraint case.

- In the inequality case, we need ∇F and ∇G to point in different directions, otherwise we can move into the interior of the constraint set and decrease F in the process, contradicting the optimality of $\bar{z}$. So we need: (a) multiplier $\mu \geq 0$, and (b) complementary slackness:

  $$\mu G(\bar{z}) = 0.$$

- General problem:

  $$\min_{z \in \mathbb{R}^m} F(z) \quad \text{s.t. } G_j(z) \leq 0,\ j = 1, \ldots, n. \tag{9}$$

- Introduce Lagrange multipliers $\mu = (\mu_1, \ldots, \mu_n)$ with $\mu_j \geq 0$, and the Lagrangian:

  $$L(z, \mu) = F(z) + \sum_{j=1}^{n} \mu_j G_j(z).$$

[8] Otherwise, we can move along the curve $G(z) = 0$ (which is locally perpendicular to ∇G) to decrease the function value of F.

A.3 The KKT Conditions and Lagrangian Duality

Then, we can derive the following necessary conditions (known as the Karush-Kuhn-Tucker (KKT) conditions). They can be proven to be necessary for optimality under some technical assumptions[9] (constraint qualifications).

Theorem A.1 (KKT Conditions (Informal, Under Certain Assumptions))**.** *For each solution $\bar{z}$ of Equation* (9)*, there exists $\bar{\mu} \in \mathbb{R}^n$ which must satisfy:*

- ***Stationarity:*** $\nabla_z L(\bar{z}, \bar{\mu}) = 0$.
- ***Primal feasibility:*** $G_j(\bar{z}) \leq 0,\ j = 1, \ldots, n$.
- ***Dual feasibility:*** $\bar{\mu}_j \geq 0,\ j = 1, \ldots, n$.
- ***Complementary slackness:*** $\bar{\mu}_j G_j(\bar{z}) = 0,\ j = 1, \ldots, n$.

Moreover, if F and G_j are convex, then these conditions are also sufficient.

Finally, we may consider the dual of the problem Equation (9) as:

$$\max_{\mu \in \mathbb{R}^n} \tilde{F}(\mu), \text{ where } \tilde{F}(\mu) = \inf_{z \in \mathbb{R}^m} L(z, \mu),\ \mu \geq 0.$$

Under convexity and constraint qualification, **strong duality** holds: optimal primal and dual objective values coincide. A solution $(\bar{z}, \bar{\mu})$ to the KKT conditions solves both.

A.4 Appendix 2: Derivation of the Wolfe Dual Problem

We start with the primal problem and form the Lagrangian, which incorporates the constraints using Lagrange multipliers $\alpha_i, \mu_i \geq 0$ (dual feasibility). The Lagrange (primal) function is:

$$L_P = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i(x_i^T \beta + \beta_0) - 1 + \xi_i] - \sum_{i=1}^{N} \mu_i \xi_i. \tag{10}$$

We minimize L_P with respect to the primal variables β, β_0, ξ_i:

$$\begin{aligned}
\frac{\partial L_P}{\partial \beta} = \beta - \sum_{i=1}^{N} \alpha_i y_i x_i = 0 &\implies \beta = \sum_{i=1}^{N} \alpha_i y_i x_i, \\
\frac{\partial L_P}{\partial \beta_0} = -\sum_{i=1}^{N} \alpha_i y_i = 0 &\implies \sum_{i=1}^{N} \alpha_i y_i = 0, \\
\frac{\partial L_P}{\partial \xi_i} = C - \alpha_i - \mu_i = 0 &\implies \alpha_i = C - \mu_i.
\end{aligned}$$

Substitute these stationarity conditions back into L_P. The terms involving β_0 and ξ_i cancel out, and we replace β with its expression in terms of α_i. This gives us the Wolfe dual objective function:

$$\begin{aligned}
L_D &= \frac{1}{2}\left\|\sum_i \alpha_i y_i x_i\right\|^2 - \sum_i \alpha_i y_i x_i^T \left(\sum_k \alpha_k y_k x_k\right) + \sum_i \alpha_i \\
&= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k.
\end{aligned}$$

[9] See, e.g., Appendix B of `https://cs.nyu.edu/~mohri/mlbook/` for details.

The constraints $\alpha_i \geq 0$ and $\mu_i \geq 0$, combined with $\alpha_i = C - \mu_i$, imply that

$$0 \leq \alpha_i \leq C.$$

This completes the derivation of the dual problem.