Machine Learning (CE 40717)

Fall 2025

Ali Sharifi-Zarchi

CE Department
Sharif University of Technology

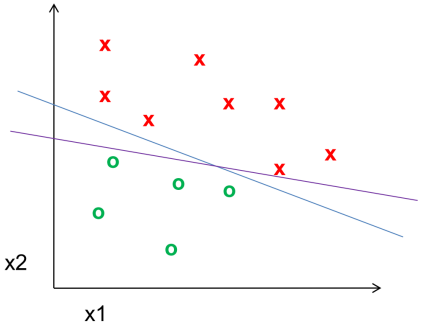
November 2, 2025

Support Vector Machines (SVMs)

- Developed in the 1990s by Vapnik and colleagues at Bell Labs based on statistical learning theory
- One of the most popular learning techniques until deep learning resurgence
- What is interesting about SVMs
  - **Generalization properties**, including achieving a margin and structural risk minimization
  - Extension to non-linear classifier via **kernels**
  - Dual form that shows how **linear classifiers can be seen as a weighted average of training examples**
  - Optimization via **stochastic gradient descent**, also used for neural networks

What is the best linear classifier?

- **Logistic regression**
  - Maximize expected likelihood of label given data
  - Every example contributes to loss
- **SVM**
  - Make all examples at least minimally confident
  - Base decision on a minimal set of examples



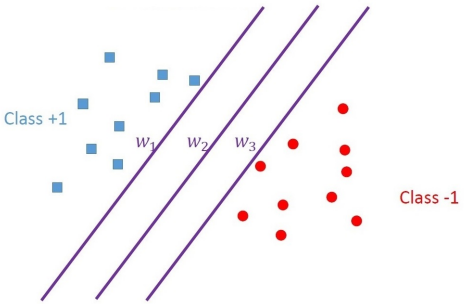
Multiple optimal solutions?

- Given training data $\{(x_i, y_i) : 1 \le i \le n\}$ i.i.d. from distribution D
- Hypothesis

$$y = \text{sign}(f_w(x)) = \text{sign}(w^T x)$$

  - $y = +1$ if $w^T x > 0$
  - $y = -1$ if $w^T x < 0$

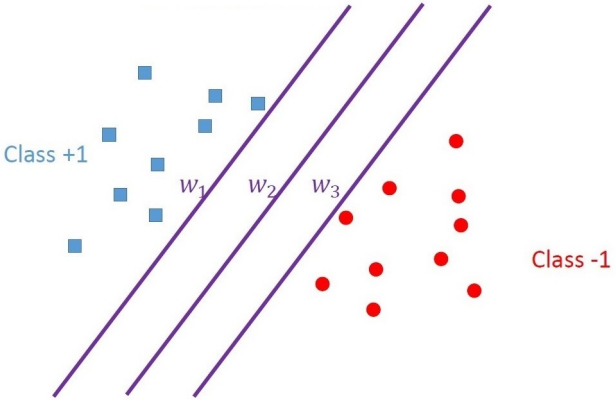
- Lets assume that we can optimize to find w



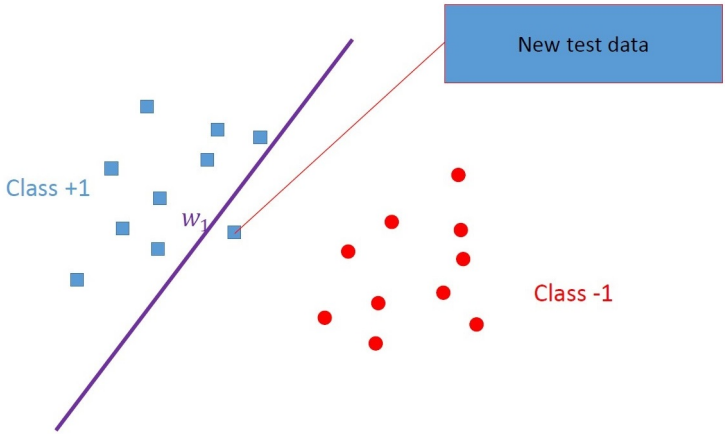
Same on empirical loss
Different on test/expected loss

What is better linear separation?

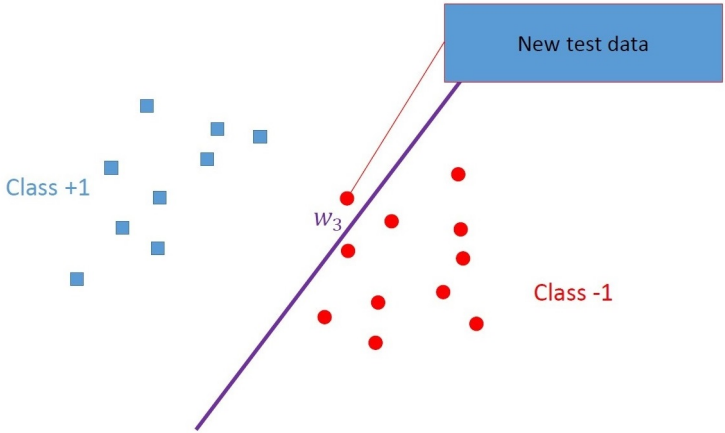
Which Separator Do You Pick?



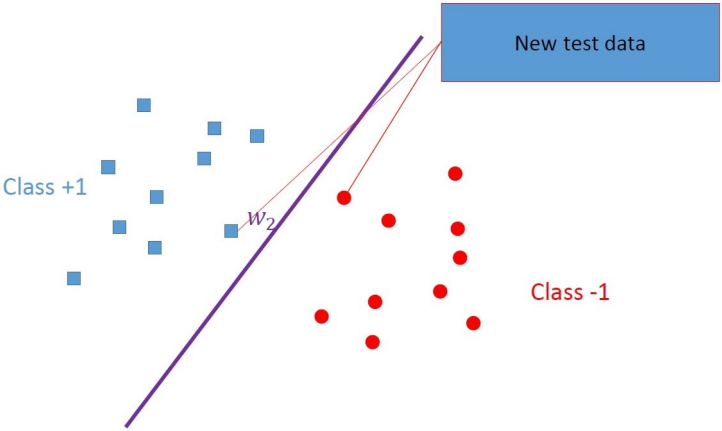
What is better linear separation?



What is better linear separation?

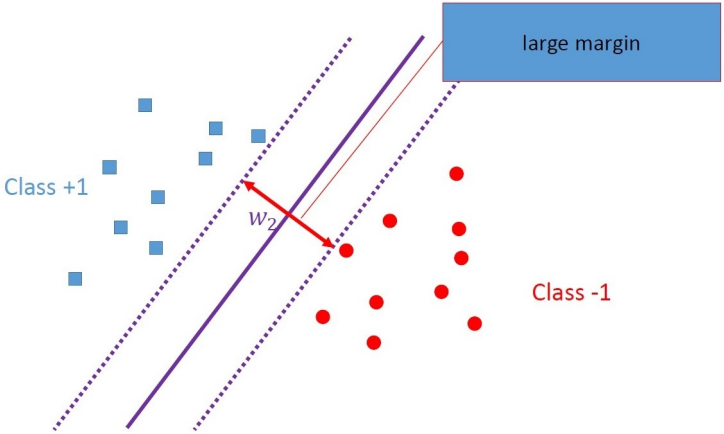


What is better linear separation?

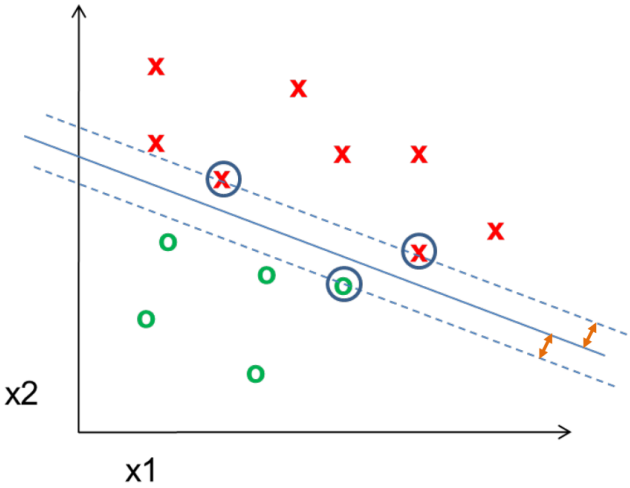


What is better linear separation?

Larger margin provides better generalization to unseen data.



SVM Terminology

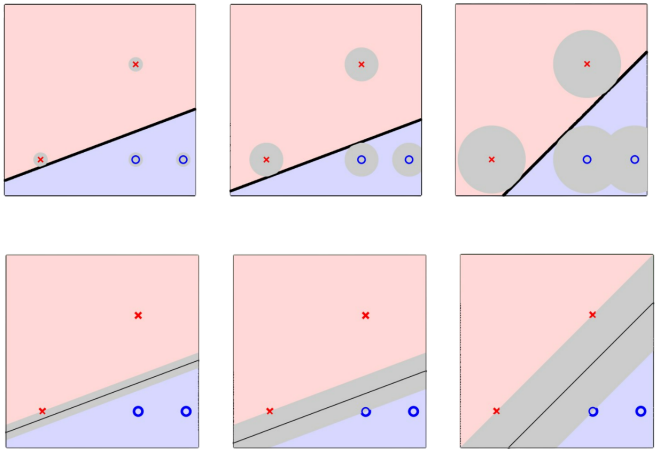


Support Vector: training observations whose distance to the hyperplane is equal to the margin

Margin:smallest distance between any training observation and the hyperplane.

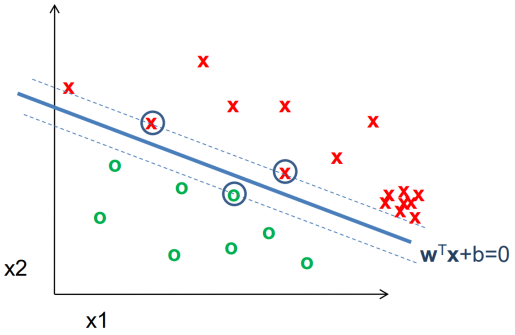
Fat Hyperplane

We call such hyperplanes fat



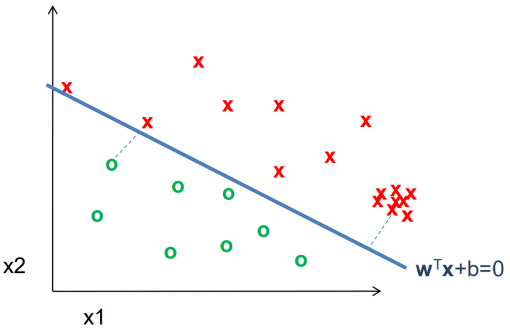
SVMs minimize $\mathbf{w}^T\mathbf{w}$ while preserving a margin of 1

Optimized SVM



Decision boundary depends only on support vectors (circled)

Optimized Linear Logistic Regression



Minimizes the sum of logistic error on all samples, so boundary should be further from dense regions

Why is it called a support vector?

- **Support**: maximal margin hyperplane only depends on these observations.
- **Vector**: points are vectors in p-dimensional space.

If support vectors are perturbed, then MM hyperplane will change.

If other training observations perturbed (provided not perturbed within margin distance of hyperplane), then MM hyperplane not affected.

Finding **w** with large margin

Let x_n be the nearest data point to the plane

$$w^\top x = 0.$$

How far is it? 2 preliminary technicalities:

1. Normalize w:

$$\left| w^\top x_n \right| = 1$$

2. Pull out w_0:

$$w = (w_1, \cdots, w_d)$$

apart from b.

The plane is now

$$w^\top x + b = 0$$

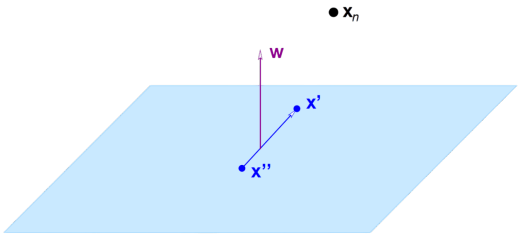
Computing the distance

The distance between $\mathbf{x}_n$ and the plane $\mathbf{w}^\top \mathbf{x} + b = 0$ where $|\mathbf{w}^\top \mathbf{x}_n + b| = 1$

The vector $\mathbf{w}$ is $\perp$ to the plane in the $\mathcal{X}$ space:

Take $\mathbf{x}'$ and $\mathbf{x}''$ on the plane

$$\mathbf{w}^\top \mathbf{x}' + b = 0 \quad \text{and} \quad \mathbf{w}^\top \mathbf{x}'' + b = 0 \Longrightarrow \quad \mathbf{w}^\top (\mathbf{x}' - \mathbf{x}'') = 0$$

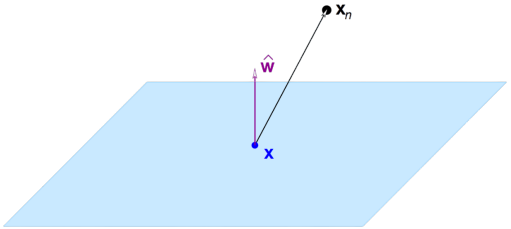


and the distance is ...

Distance between $\mathbf{x}_n$ and the plane: Take any point $\mathbf{x}$ on the plane Projection of $\mathbf{x}_n - \mathbf{x}$ on $\mathbf{w}$

$$\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \quad \Longrightarrow \quad \text{distance} = |\hat{\mathbf{w}}^\top (\mathbf{x}_n - \mathbf{x})|$$

$$\text{distance} = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\top \mathbf{x}_n - \mathbf{w}^\top \mathbf{x}| = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\top \mathbf{x}_n + b - \mathbf{w}^\top \mathbf{x} - b| = \frac{1}{\|\mathbf{w}\|}$$



The optimization problem

Maximize $\frac{1}{\|\mathbf{w}\|}$
subject to $\min_{n=1,2,\ldots,N} |\mathbf{w}^\top \mathbf{x}_n + b| = 1$

Notice: $|\mathbf{w}^\top \mathbf{x}_n + b| = y_n(\mathbf{w}^\top \mathbf{x}_n + b)$

Minimize $\frac{1}{2}\mathbf{w}^\top \mathbf{w}$
subject to $y_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 \quad \text{for} \quad n = 1, 2, \ldots, N$

$$\mathbf{w} \in \mathbb{R}^d, \quad b \in \mathbb{R}$$

Lagrange? inequality constraints $\Longrightarrow$ KKT

Lagrange formulation

Minimize $\mathscr{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} - \sum_{n=1}^{N} \alpha_n \big(y_n(\mathbf{w}^\top\mathbf{x}_n + b) - 1\big)$

w.r.t. $\mathbf{w}$ and b and maximize w.r.t. each $\alpha_n \geq 0$

$$\nabla_{\mathbf{w}}\mathscr{L} = \mathbf{w} - \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n = 0$$

$$\frac{\partial \mathscr{L}}{\partial b} = -\sum_{n=1}^{N} \alpha_n y_n = 0$$

Substituting

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n \quad \text{and} \quad \sum_{n=1}^{N} \alpha_n y_n = 0$$

in the Lagrangian $\mathscr{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} - \sum_{n=1}^{N} \alpha_n \big(y_n(\mathbf{w}^\top \mathbf{x}_n + b) - 1\big)$

we get $\mathscr{L}(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} y_n y_m \alpha_n \alpha_m \mathbf{x}_n^\top \mathbf{x}_m$

Maximize w.r.t. $\boldsymbol{\alpha}$ subject to $\alpha_n \geq 0$ for $n = 1, \ldots, N$ and $\sum_{n=1}^{N} \alpha_n y_n = 0$

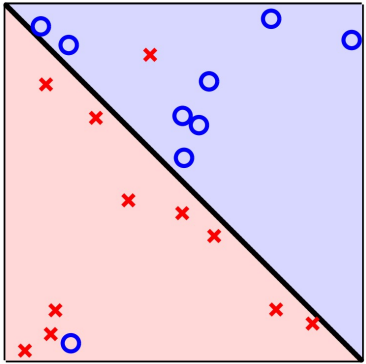
The solution quadratic programming

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\boldsymbol{\alpha}^\top \begin{bmatrix} y_1 y_1 \mathbf{x}_1^\top \mathbf{x}_1 & y_1 y_2 \mathbf{x}_1^\top \mathbf{x}_2 & \dots & y_1 y_N \mathbf{x}_1^\top \mathbf{x}_N \\ y_2 y_1 \mathbf{x}_2^\top \mathbf{x}_1 & y_2 y_2 \mathbf{x}_2^\top \mathbf{x}_2 & \dots & y_2 y_N \mathbf{x}_2^\top \mathbf{x}_N \\ \vdots & \vdots & \ddots & \vdots \\ y_N y_1 \mathbf{x}_N^\top \mathbf{x}_1 & y_N y_2 \mathbf{x}_N^\top \mathbf{x}_2 & \dots & y_N y_N \mathbf{x}_N^\top \mathbf{x}_N \end{bmatrix} \boldsymbol{\alpha} + (-\mathbf{1}^\top)\boldsymbol{\alpha}$$

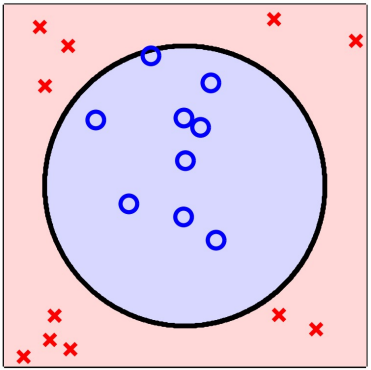
subject to

$$\mathbf{y}^\top \boldsymbol{\alpha} = 0 \quad \text{and} \quad \mathbf{0} \le \boldsymbol{\alpha} \le \infty$$

Non-Separable Data



Tolerate error



Nonlinear transform

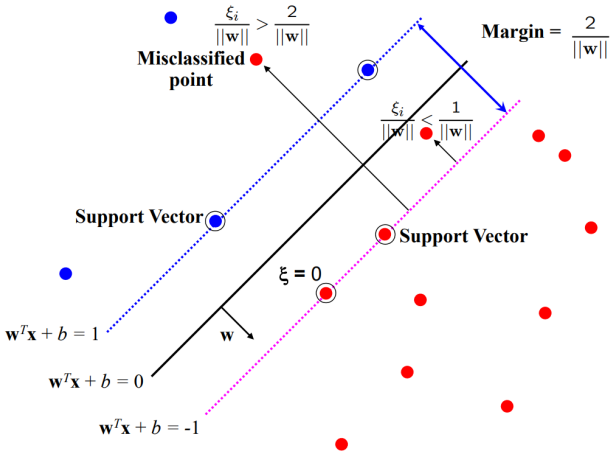
Introduce slack variables

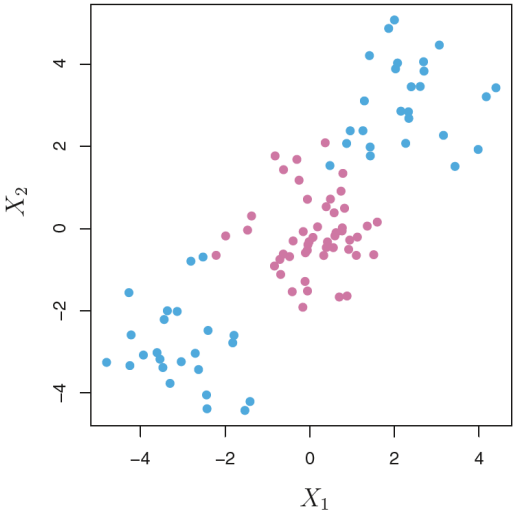
Margin violation:
$y_n(w^\top x_n + b) \geq 1$ fails

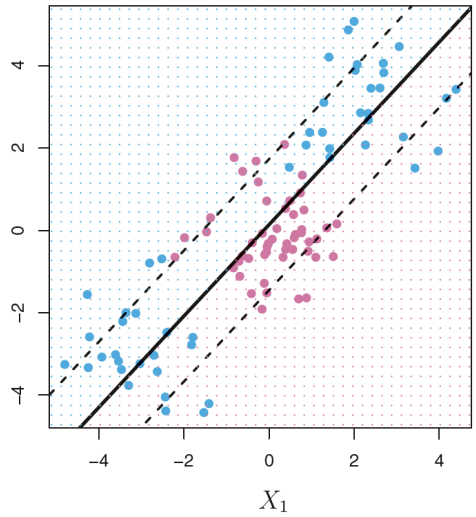
Quantify: $y_n(w^\top x_n + b) \geq 1 - \xi_n$,
$\xi_n \geq 0$

Total violation $= \sum_{n=1}^{N} \xi_n$

- for $0 < \xi \leq 1$ point is between margin and correct side of hyperplane. This is a **margin violation**
- for $\xi > 1$ point is **misclassified**




X_2
X_1
4
2
0
−2
−4
−4
−2
0
2
4


X_1
4
2
0
−2
−4
−4
−2
0
2
4

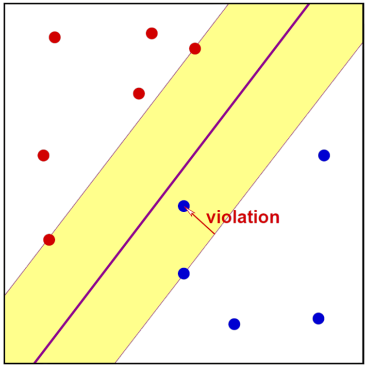
Soft Margin SVM

$$\text{minimize}_{b,w,\xi} \quad \frac{1}{2} w^\top w + C \sum_{n=1}^{N} \xi_n$$

$$\text{subject to:} \quad y_n(w^\top x_n + b) \geq 1 - \xi_n \quad \text{for } n = 1, \ldots, N$$

$$\xi_n \geq 0 \quad \text{for } n = 1, \ldots, N$$

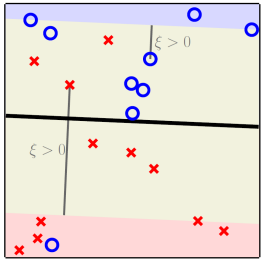
$$w \in \mathbb{R}^d, \quad b \in \mathbb{R}, \quad \xi \in \mathbb{R}^N$$



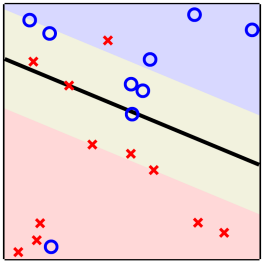
Soft Margin SVM

- Trades off soft in-sample error $\sum_{n=1}^{N} \xi_n$ with weight norm $\frac{1}{2} w^\top w$
- Every constraint can be satisfied if ξ_i is sufficiently large
- C is a **regularization** parameter:
  - small C allows constraints to be easily ignored → large margin
  - large C makes constraints hard to ignore → narrow margin
  - $C = \infty$ enforces all constraints: hard margin
- This is still a quadratic optimization problem and there is a unique minimum. Note, there is only one parameter, C.

Non-Separable Data



$C = 1$



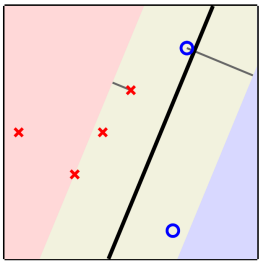
$C = 500$

$$\text{minimize}_{b,w,\xi} \quad \frac{1}{2} w^\top w + C \sum_{n=1}^{N} \xi_n$$

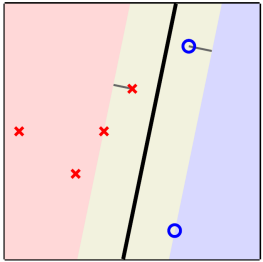
$$\text{subject to:} \quad y_n(w^\top x_n + b) \geq 1 - \xi_n \quad \text{for } n = 1, \ldots, N$$

$$\xi_n \geq 0 \quad \text{for } n = 1, \ldots, N$$

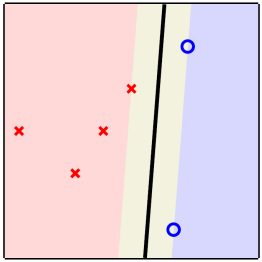
Soft Margin SVM With Separable Data



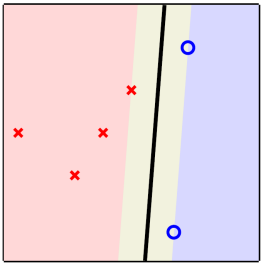
small C



medium C



large C



hard margin

$$\text{minimize}_{b,w,\xi} \quad \frac{1}{2} w^\top w + C \sum_{n=1}^{N} \xi_n$$

$$\text{subject to:} \quad y_n(w^\top x_n + b) \geq 1 - \xi_n \quad \text{for } n = 1, \ldots, N$$

$$\xi_n \geq 0 \quad \text{for } n = 1, \ldots, N$$

Choice of C is IMPORTANT

Lagrange formulation

$$\mathscr{L}(w, b, \xi, \alpha, \beta) = \frac{1}{2} w^\top w + C \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} \alpha_n \big(y_n (w^\top x_n + b) - 1 + \xi_n\big) - \sum_{n=1}^{N} \beta_n \xi_n$$

Minimize w.r.t. $\mathbf{w}$, $\mathbf{b}$, and ξ and maximize w.r.t. each $\alpha_n \geq 0$ and $\beta_n \geq 0$

$$\nabla_w \mathscr{L} = w - \sum_{n=1}^{N} \alpha_n y_n x_n = 0$$

$$\frac{\partial \mathscr{L}}{\partial b} = - \sum_{n=1}^{N} \alpha_n y_n = 0$$

$$\frac{\partial \mathscr{L}}{\partial \xi_n} = C - \alpha_n - \beta_n = 0$$

Soft-margin SVM: Dual problem

$$\max_{\boldsymbol{\alpha}} \left(\sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \mathbf{x}_n^\top \mathbf{x}_m \right)$$

Subject to

$$\sum_{n=1}^{N} \alpha_n y_n = 0, \quad 0 \le \alpha_n \le C, \quad n = 1, \dots, N$$

After solving the above quadratic problem, $\mathbf{w}$ is found as:

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$$

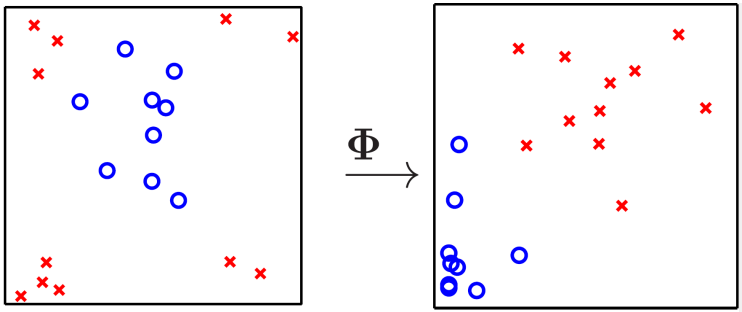
Not linearly separable data

Noisy data or overlapping classes
(we discussed about it: soft margin)

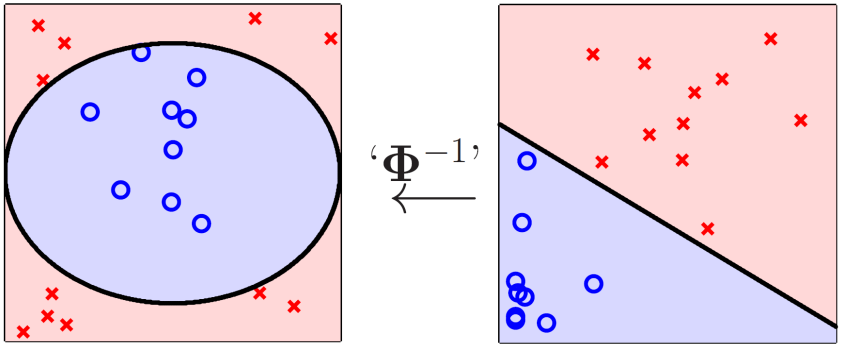
- Near linearly separable

Non-linear decision surface

- Transform to a new feature space



Mechanics of the Nonlinear Transform



Soft-margin SVM in a transformed space: Primal problem

- **Primal problem:**

$$\min_{\mathbf{w}, w_0} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N} \xi_n$$

subject to:

$$y_n\left(\mathbf{w}^T \phi(x_{(n)}) + w_0\right) \geq 1 - \xi_n, \quad n = 1, \ldots, N$$

$$\xi_n \geq 0$$

- $\mathbf{w} \in \mathbb{R}^m$: the weights that must be found.
- If $m \gg d$ (very high-dimensional feature space), then there are many more parameters to learn.

Soft-margin SVM in a transformed space: Dual problem

- **Optimization problem:**

$$\max_{\alpha} \left\{ \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \phi(x_n)^T \phi(x_m) \right\}$$

Subject to:

$$\sum_{n=1}^{N} \alpha_n y_n = 0$$

$$0 \leq \alpha_n \leq C \quad n = 1, \ldots, N$$

- If we have inner products $\phi(x_i)^T \phi(x_j)$, only $\boldsymbol{\alpha} = [\boldsymbol{\alpha_1}, \ldots, \boldsymbol{\alpha_N}]$ needs to be learned.
- Not necessary to learn m parameters as opposed to the primal problem.

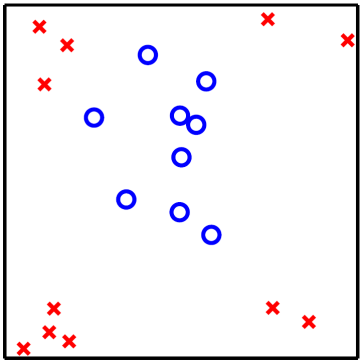
Classifying a new data

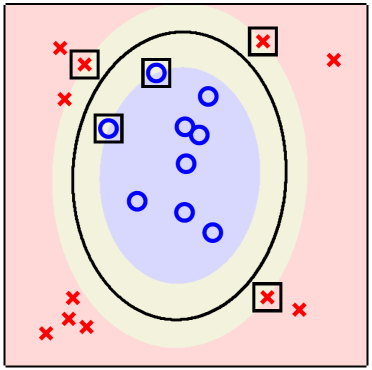
$$\hat{y} = sign(w_0 + \mathbf{w}^T \phi(x))$$

$$\text{where } \mathbf{w} = \sum_{\alpha_n > 0} \alpha_n y_n \phi(x_n)$$

$$\text{and } w_0 = y_s - \mathbf{w}^T \phi(x_s)$$

Not linearly separable data





Kernel SVM

- Learns linear decision boundary in a high-dimensional space without explicitly working on the mapped data.
- Let

$$\phi(x)^T \phi(x') = K(x, x') \quad \text{(kernel)}$$

- Example:

$$x = [x_1, x_2] \quad \text{and second-order } \phi:$$

$$\phi(x) = [1, x_1, x_2, x_1^2, x_2^2, x_1 x_2]$$

Then,

$$K(x, x') = 1 + x_1 x_1' + x_2 x_2' + x_1^2 x_1'^2 + x_2^2 x_2'^2 + x_1 x_1' x_2 x_2'$$

Kernel trick

- Compute $K(x, x')$ without transforming x and x'.
- Example: Consider

$$K(x, x') = (1 + x^T x')^2$$
$$= (1 + x_1 x'_1 + x_2 x'_2)^2$$
$$= 1 + 2x_1 x'_1 + 2x_2 x'_2 + x_1^2 x_1'^2 + x_2^2 x_2'^2 + 2x_1 x'_1 x_2 x'_2$$

This is an inner product in:

$$\phi(x) = [1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2]$$

$$\phi(x') = [1, \sqrt{2}x'_1, \sqrt{2}x'_2, x_1'^2, x_2'^2, \sqrt{2}x'_1 x'_2]$$

Polynomial kernel: Degree two

- We instead use

$$K(x, x') = (x^T x' + 1)^2$$

that corresponds to:

d-dimensional feature space:

$$x = [x_1, \ldots, x_d]^T$$

$$\phi(x) = [1, \sqrt{2}x_1, \ldots, \sqrt{2}x_d, x_1^2, \ldots, x_d^2, \sqrt{2}x_1x_2, \ldots, \sqrt{2}x_1x_d, \sqrt{2}x_2x_3, \ldots, \sqrt{2}x_{d-1}x_d]^T$$

Polynomial kernel

- This can similarly be generalized to d-dimensional x and ϕs are polynomials of order M:

$$K(x, x') = (1 + x^T x')^M$$
$$= (1 + x_1 x'_1 + x_2 x'_2 + \cdots + x_d x'_d)^M$$

- Example: SVM boundary for a polynomial kernel

$$w_0 + w^T \phi(x) = 0$$
$$\Rightarrow w_0 + \sum_{\alpha_i > 0} \alpha_i y_i \phi(x_i)^T \phi(x) = 0$$
$$\Rightarrow w_0 + \sum_{\alpha_i > 0} \alpha_i y_i k(x_i, x) = 0$$

$\Rightarrow w_0 + \sum_{\alpha_i > 0} \alpha_i y_i (1 + x_i^T x)^M = 0 \quad \rightarrow \quad$ Boundary is a polynomial of order M

Why kernel?

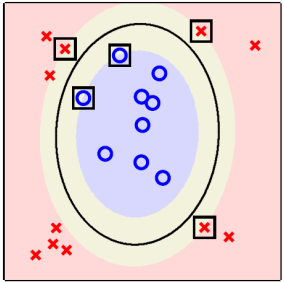
- Kernel functions K can indeed be efficiently computed, with a cost proportional to d (the dimensionality of the input) instead of m.
- **Example:** consider the second-order polynomial transform:

$$\phi(x) = [1, x_1, \ldots, x_d, x_1^2, x_1 x_2, \ldots, x_d x_d]^T \quad m = 1 + d + d^2$$

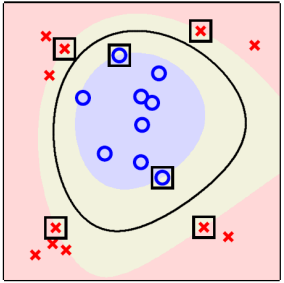
$$\phi(x)^T \phi(x') = 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d} \sum_{j=1}^{d} x_i x_j x_i' x_j' \quad O(m)$$

$$\phi(x)^T \phi(x') = 1 + (x^T x') + (x^T x')^2 \quad O(d)$$

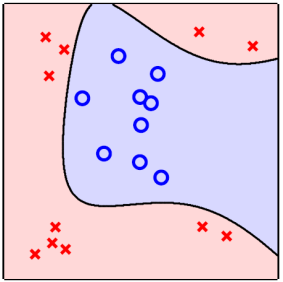
Nonlinear Transform and SVM



Φ_2+ SVM



Φ_3+ SVM



Φ_3+ pseudoinverse algorithm

- Φ_3 has almost 2× the parameters of Φ_2
- Φ_3-SVM does not display significant overfitting compared to Φ_3-regression
- support vectors did not double
- Can go to higher dimensions if support vectors stays small or margin stays large

Gaussian or RBF kernel

- If $K(x, x')$ is an inner product in some transformed space of x, it is good.
- $K(x, x') = \exp\left(-\frac{\|x-x'\|^2}{\gamma}\right)$
- Take one dimensional case with $\gamma = 1$:

$$K(x, x') = \exp\left(-(x-x')^2\right)$$

$$= \exp(-x^2)\exp(-x'^2)\exp(2xx')$$

$$= \exp(-x^2)\exp(-x'^2)\sum_{k=1}^{\infty}\frac{2^k x^k x'^k}{k!}$$

Some common kernel functions

- **Linear:** $k(x, x') = x^T x'$
- **Polynomial:** $k(x, x') = (x^T x' + 1)^M$
- **Gaussian:** $k(x, x') = \exp\left(-\frac{\|x-x'\|^2}{\gamma}\right)$
- **Sigmoid:** $k(x, x') = \tanh(a x^T x' + b)$

Kernel formulation of SVM

Optimization problem:

$$\max_{\boldsymbol{\alpha}} \left\{ \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m k(\mathbf{x}_n, \mathbf{x}_m) \right\}$$

$$\text{Subject to } \sum_{n=1}^{N} \alpha_n y_n = 0$$

$$0 \le \alpha_n \le C \quad n = 1, \dots, N$$

$$\mathbf{Q} = \begin{bmatrix} y_1 y_1 K(\mathbf{x}_1, \mathbf{x}_1) & \cdots & y_1 y_N K(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ y_N y_1 K(\mathbf{x}_N, \mathbf{x}_1) & \cdots & y_N y_N K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

Gaussian kernel

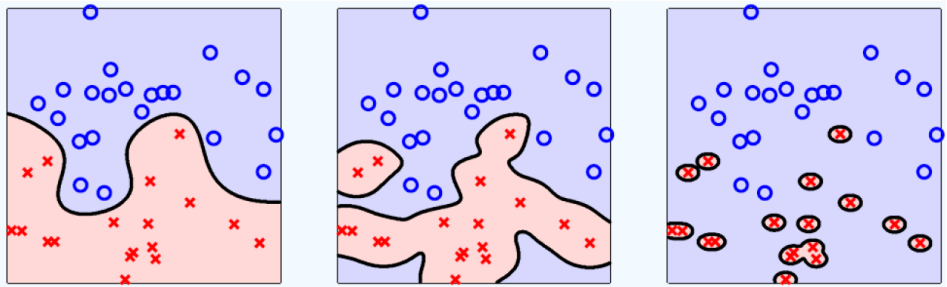
Example: SVM boundary for a Gaussian kernel

- Considers a Gaussian function around each data point.

$$w_0 + \sum_{\alpha_i > 0} \alpha_i y^{(i)} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}^{(i)}\|^2}{\sigma}\right) = 0$$

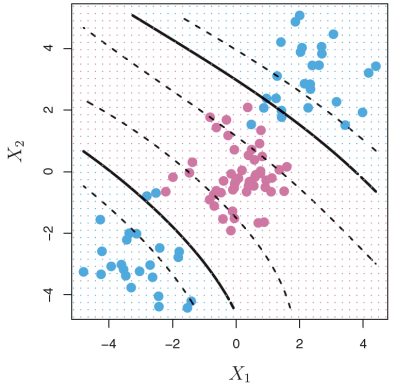
- SVM + Gaussian Kernel can classify any arbitrary training set
  - Training error is zero when $\sigma \to 0$
    - All samples become support vectors (likely overfitting)

Hard margin Example



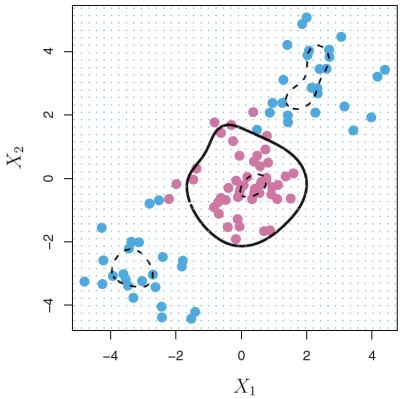
For narrow Gaussian (large σ), even the protection of a large margin cannot suppress overfitting.

SVM with a polynomial kernel of degree 3



$$K(x_i, x_i') = \left(1 + \sum_{j=1}^{p} x_{ij} x_{ij}'\right)^d$$

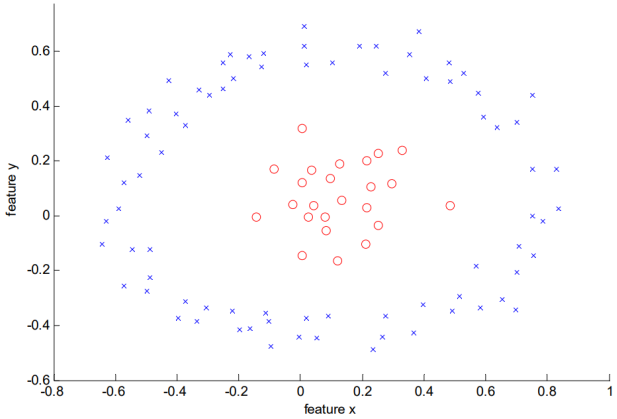
SVM with a radial kernel



$$K(x_i, x_i') = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{ij}')^2\right)$$

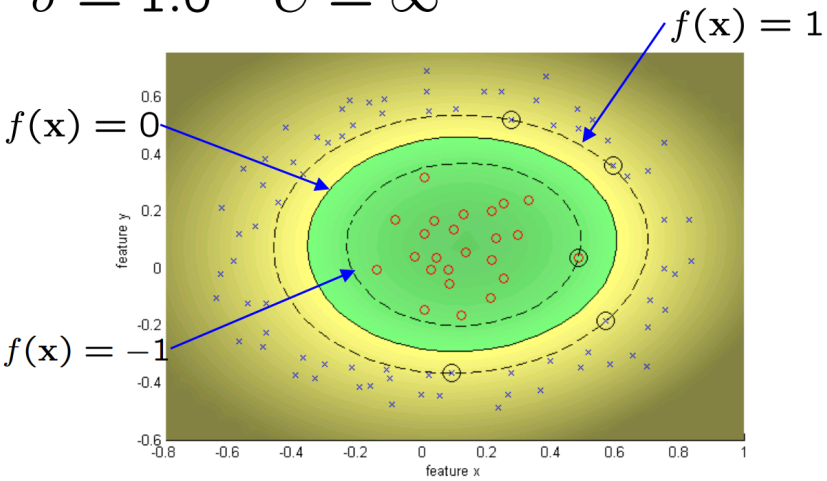
SVM Gaussian kernel: Example

$$f(\mathbf{x}) = w_0 + \sum_{\alpha_i > 0} \alpha_i y^{(i)} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}^{(i)}\|^2}{2\sigma^2}\right)$$



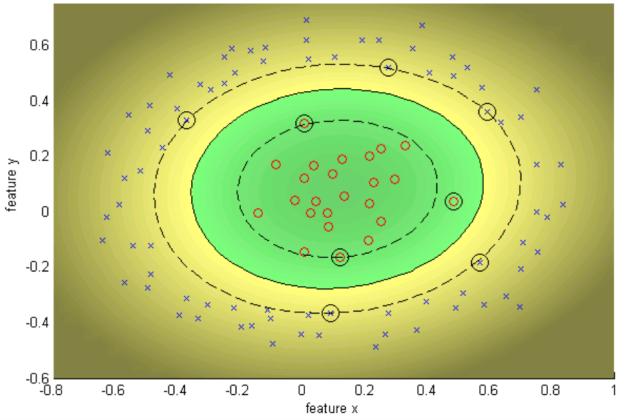
SVM Gaussian kernel: Example

$\sigma = 1.0 \quad C = \infty$



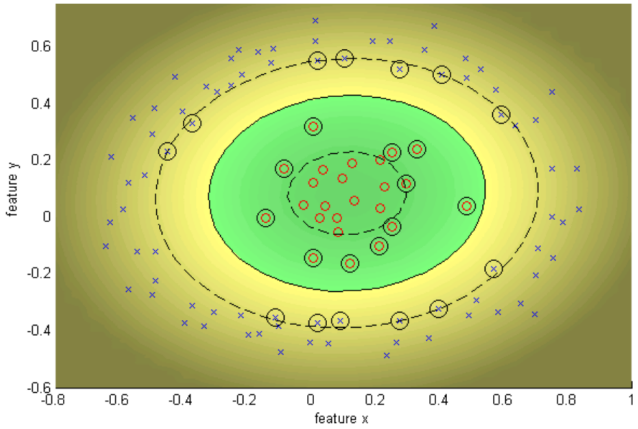
SVM Gaussian kernel: Example

$$\sigma = 1.0 \quad C = 100$$



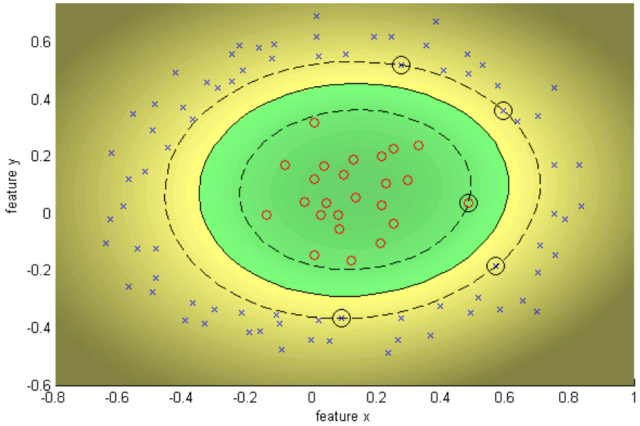
SVM Gaussian kernel: Example

$$\sigma = 1.0 \quad C = 10$$



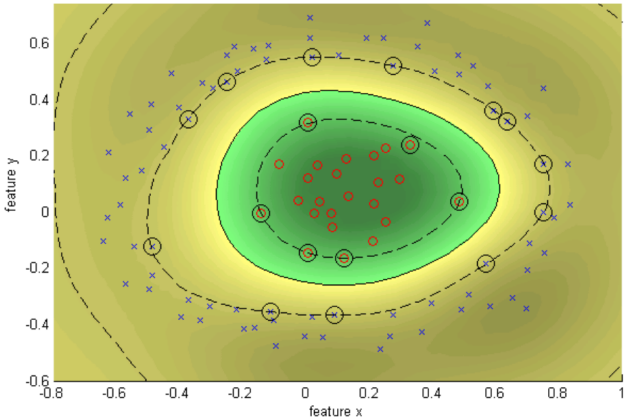
SVM Gaussian kernel: Example

$$\sigma = 1.0 \quad C = \infty$$



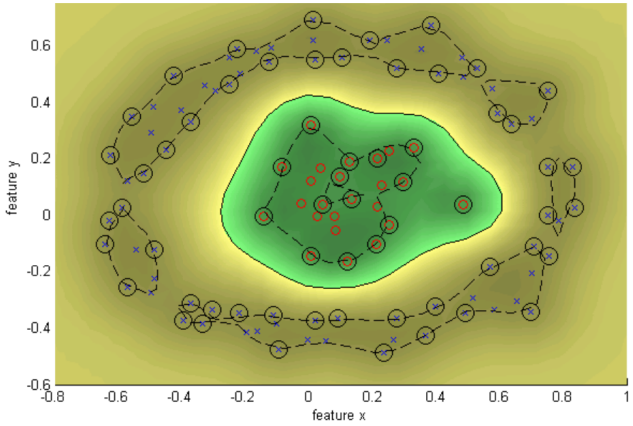
SVM Gaussian kernel: Example

$$\sigma = 0.25 \quad C = \infty$$



SVM Gaussian kernel: Example

$$\sigma = 0.1 \quad C = \infty$$



Some Common Tasks

- **Clustering**: Grouping data points into clusters based on similarity.
- **Dimensionality Reduction**: Reducing the number of features under consideration and keeping (perhaps approximately) the most informative features.
- **Anomaly Detection**: Identifying data points that deviate significantly from the norm (e.g., fraud detection).
- **Generative Modeling**: Learning the distribution of data to generate new, similar instances.

Contributions

- **This slide has been prepared thanks to:**
  - Mahdi Aghaei