Mathematics for Machine Learning

Classification with Support Vector Machines

May 24, 2025

Anastasiya Kuznetsova

Agenda

1. Refresh. Inner product

2. “Hard” SVM

3. Concept of the Margin

4. Refresh. Lagrange multiplier

5. Convex duality via Lagrange multiplier

6. Dual SVM

7. Kernels

Refresh. Inner product

Lengh of vector **u**,**v**

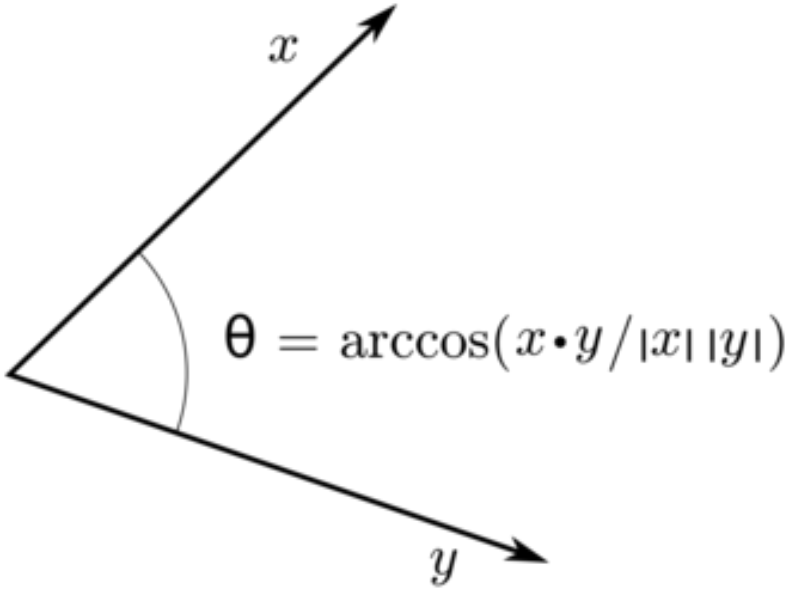
Symbol for inner product

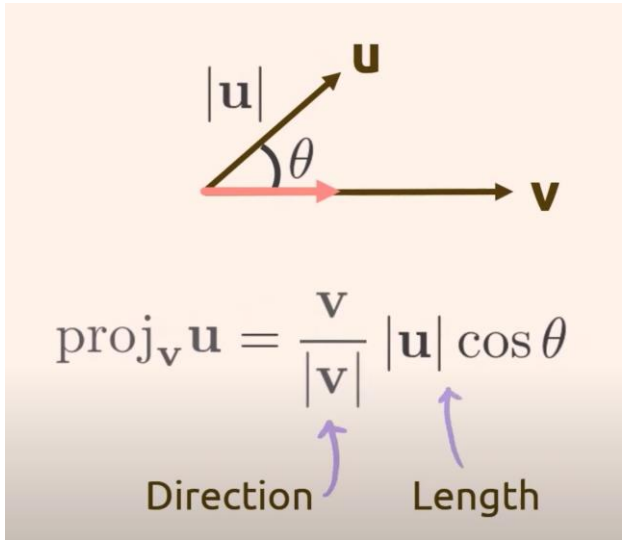
Angle between **u** and **v**

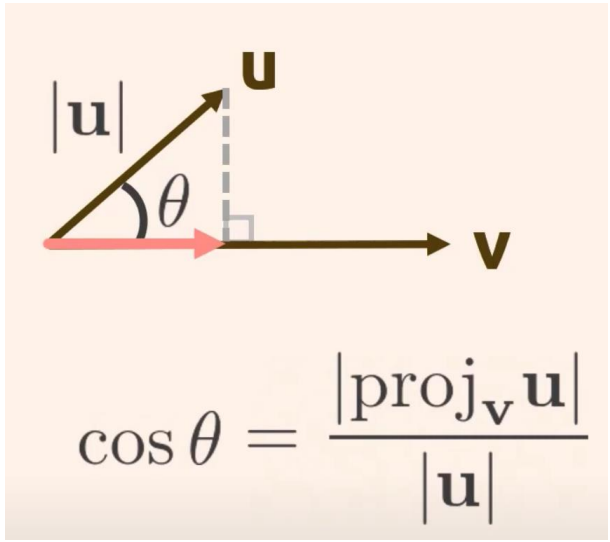
$$\mathbf{u} \bullet \mathbf{v} = |\mathbf{u}||\mathbf{v}|\cos(\theta) \quad (1)$$

$$= x_1 \times x_2 + y_1 \times y_2 \quad (2)$$

$$= \mathbf{u}\mathbf{v}^T \quad (3)$$







$$\text{proj}_{\mathbf{v}}\mathbf{u} = \frac{\mathbf{v}}{|\mathbf{v}|}|\mathbf{u}|\cos\theta$$

$$= \frac{\mathbf{v}}{|\mathbf{v}|}\frac{\mathbf{u}\cdot\mathbf{v}}{|\mathbf{u}||\mathbf{v}|}$$

$$= \frac{\mathbf{u}\cdot\mathbf{v}}{|\mathbf{v}|^2}\mathbf{v}$$

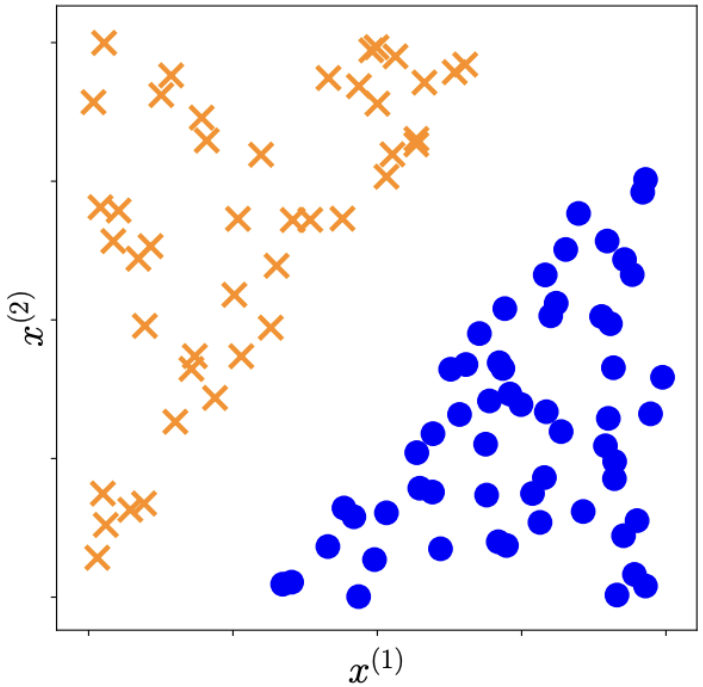
https://images.app.goo.gl/3TqWVA5qFgviLqWW6
https://www.youtube.com/watch?v=W37_2QEekt4

Classification with Support Vector Machines

$$f : \mathbb{R}^D \to \{+1, -1\}$$

$$\text{examples } \boldsymbol{x}_n \in \mathbb{R}^D$$

$$y_n \in \{+1, -1\}$$



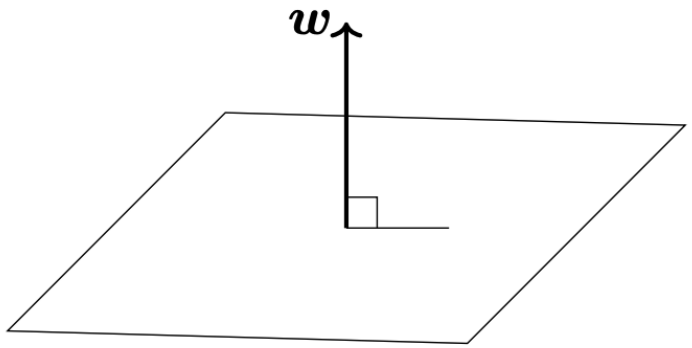
We imagine binary classification data, which can be separated by a hyperplane

Every example x_n (a vector of dimension 2) is a two-dimensional location ($x^{(1)}_n$ and $x^{(2)}_n$), and the corresponding binary label y_n

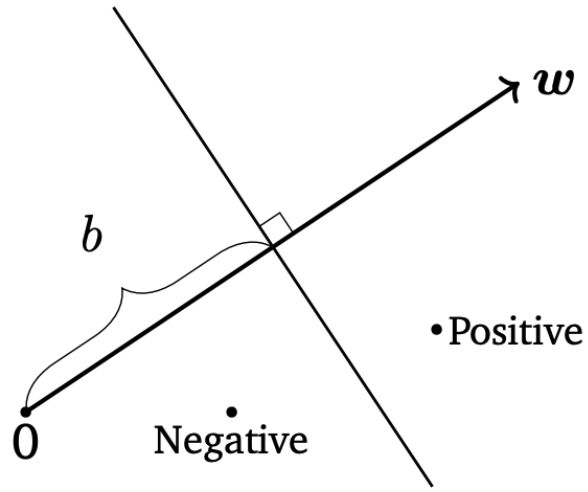
A hyperplane is an affine subspace of dimension D – 1 (if the corresponding vector space is of dimension D).

We formalize the idea of finding a linear separator of the two classes.

Separating Hyperplanes §12.1



(a) Separating hyperplane in 3D



(b) Projection of the setting in (a) onto a plane

Represent data in R^D

Find a hyperplane that divides the data into two partitions such that negative examples (labelled as -1) are on the opposite side of positive examples (labelled as +1)

Vector w is a vector normal to the hyperplane and b the intercept

$$f : \mathbb{R}^D \to \mathbb{R}$$

$$\boldsymbol{x} \mapsto f(\boldsymbol{x}) := \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$$

We define the hyperplane as

$$\{\boldsymbol{x} \in \mathbb{R}^D : f(\boldsymbol{x}) = 0\}$$

Separating Hyperplanes

Therefore, to classify a test example x_{test}, we calculate the value of the function $f(x_{test})$ and classify the example as +1 if $f(x_{test}) \geqslant 0$ and −1 otherwise.

When training the classifier, we want to ensure that the examples with positive labels are on the positive side of the hyperplane

$$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \geqslant 0 \quad \text{when} \quad y_n = +1$$

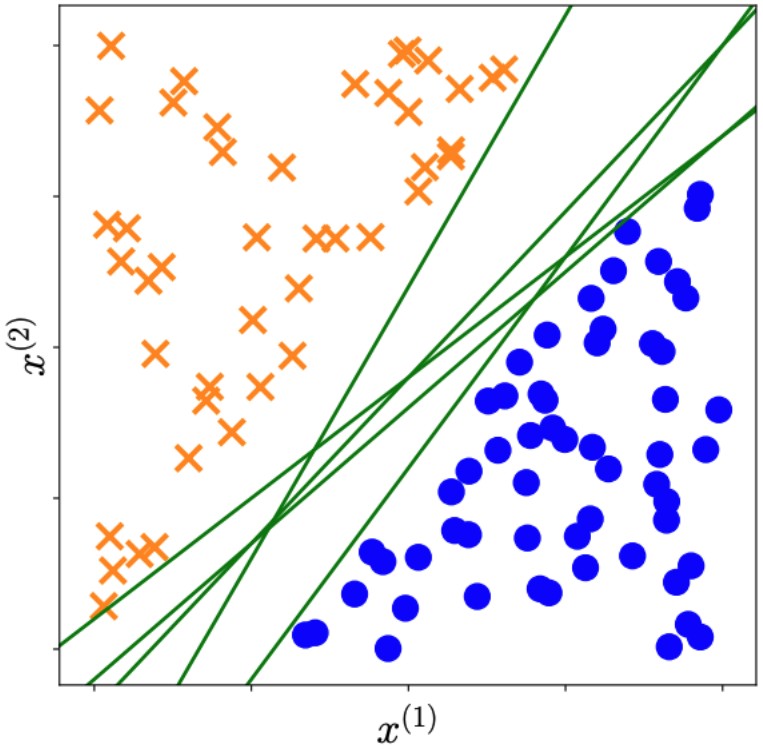
and the examples with negative labels are on the negative side, i.e.,

$$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b < 0 \quad \text{when} \quad y_n = -1\,.$$

In a single equation

$$y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant 0$$

Concept of the Margin §12.2.1



But we can create infinitely many separating hyperplanes,
How do we choose the best?

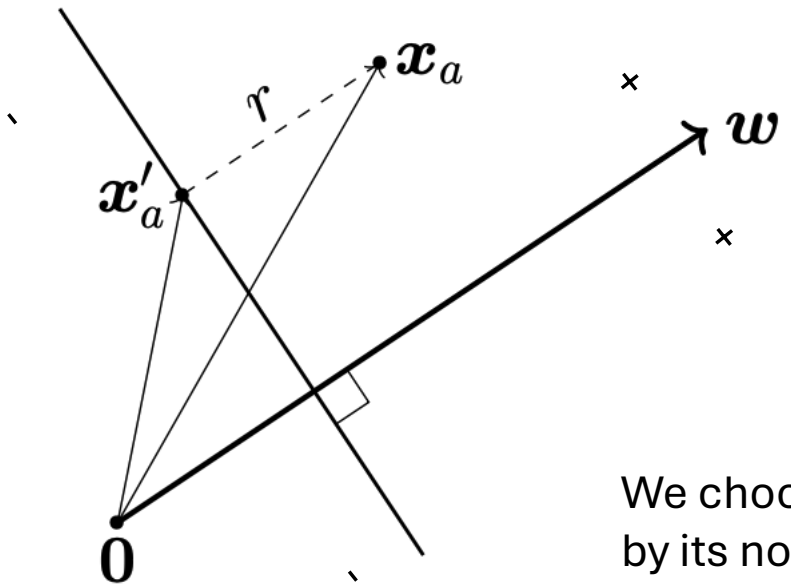
Choose the separating hyperplane that maximizes the margin between the positive and negative examples!

Margin is the distance of the separating hyperplane to the closest examples in the dataset, assuming that the dataset is linearly separable.

How can we calculate this distance?

Concept of the Margin

Consider a hyperplane $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b,$ and an example $\boldsymbol{x}_a$ on the positive side of the hyperplane $\langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b > 0.$



We need to compute distance $r > 0$ of $\boldsymbol{x}_a$ from the hyperplane

$\boldsymbol{x}'_a$ orthogonal projection of the $\boldsymbol{x}_a$ to the hyperplane

Since $\boldsymbol{w}$ is orthogonal to the hyperplane, we know that the distance r is just a scaling of this vector $\boldsymbol{w}$.

We choose to use a vector of unit length (its norm is 1) and obtain this by dividing $\boldsymbol{w}$ by its norm $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$

$$\boldsymbol{x}_a = \boldsymbol{x}'_a + r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\,.$$

If $\boldsymbol{x}_a$ is the closest point, r is the margin

Concept of the Margin

We would like the positive examples to be further than r from the hyperplane, and the negative examples to be further than r in a negative direction.

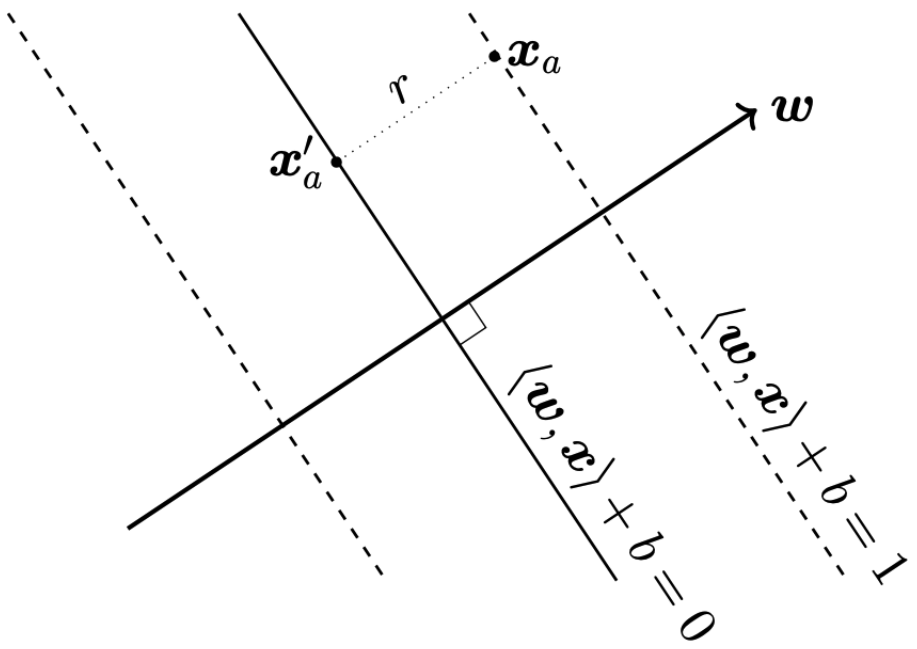
Since we are interested only in the direction, we add an assumption to our model that the parameter vector w is of unit length, i.e., ‖w‖ = 1, where $\|\boldsymbol{w}\| = \sqrt{\boldsymbol{w}^\top \boldsymbol{w}}$

Our objective $\max_{\boldsymbol{w},b,r} \underbrace{r}_{\text{margin}}$

$$\text{subject to} \quad \underbrace{y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant r}_{\text{data fitting}}, \underbrace{\|\boldsymbol{w}\| = 1}_{\text{normalization}}, \quad r > 0,$$

We want to maximize the margin r while ensuring that the data lies on the correct side of the hyperplane

Traditional Derivation of the Margin §12.2.2



$$x_a = x'_a + r\frac{w}{\|w\|}$$

Instead of $\|w\| = 1$ assumption we choose a scale for the data

We choose this scale such that the value of the predictor $\langle w, x \rangle + b$ is 1 at the closest example.

Since x'_a is the orthogonal projection of x_a onto hyperplane it lies exactly on the hyperplane

$$\langle w, x'_a \rangle + b = 0$$

$$\left\langle w, x_a - r\frac{w}{\|w\|} \right\rangle + b = 0$$

$$\underbrace{\langle w, x_a \rangle + b}_{1} - r\frac{\langle w, w \rangle}{\|w\|} = 0$$

1 by the assumption of the scale

Traditional Derivation of the Margin

1 - r $\frac{\|\boldsymbol{w}\|^2}{\|\boldsymbol{w}\|} = 0$

$$r = \frac{1}{\|\boldsymbol{w}\|}$$

We derived the distance r in terms of the normal vector w of the hyperplane

We want the positive and negative examples to be at least 1 away from the hyperplane

$$y_n(\langle \boldsymbol{w}, x_n \rangle + b) \geqslant 1 \,.$$

$$\max_{\boldsymbol{w},b} \quad \frac{1}{\|\boldsymbol{w}\|}$$
$$\text{subject to } y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant 1 \quad \text{for all} \quad n = 1, \ldots, N.$$

Traditional Derivation of the Margin

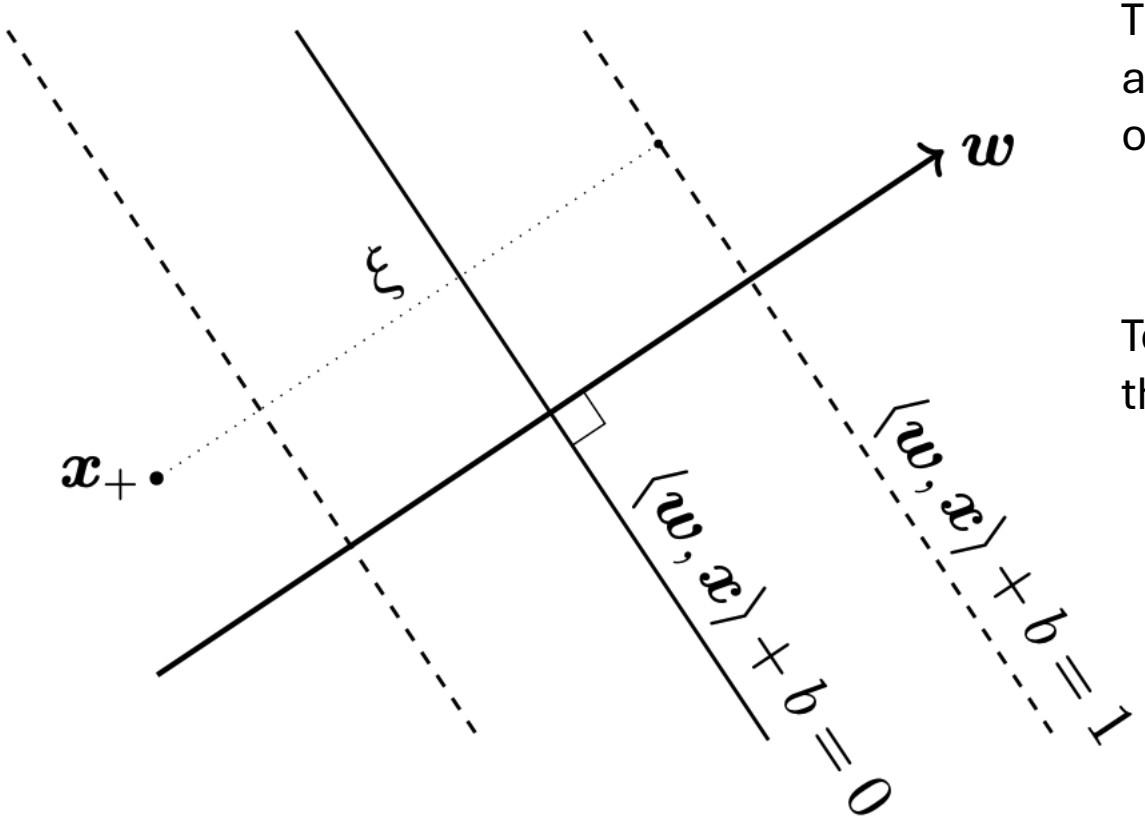
$$\max_{\boldsymbol{w},b} \quad \frac{1}{\|\boldsymbol{w}\|}$$
$$\text{subject to } y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant 1 \quad \text{for all} \quad n = 1, \ldots, N.$$

In real practice we **minimize** the squared norm

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2$$ ⟵ **hard margin SVM**
$$\text{subject to } y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant 1 \quad \text{for all} \quad n = 1, \ldots, N$$

Soft Margin SVM: Geometric View §12.2.4

In the case where data is not linearly separable, we may wish to allow some examples to fall within the margin region, or even to be on the wrong side of the hyperplane



The slack variable ξ_n to each example–label pair (x_n, y_n) that allows a particular example to be within the margin or even on the wrong side of the hyperplane

To encourage correct classification of the samples, we add the sum of the ξ_n values to the objective, weighted by *C*

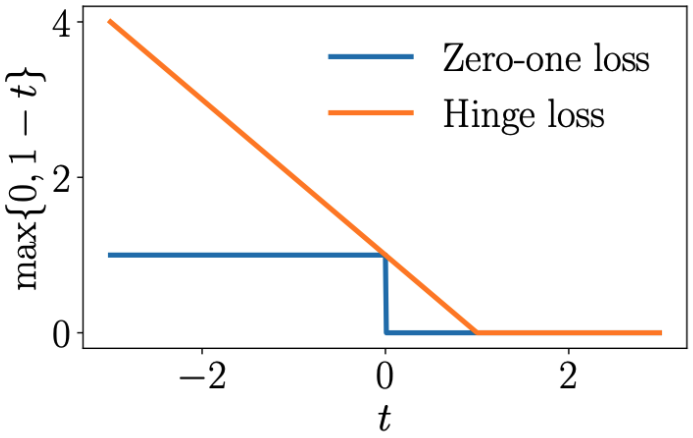
$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \\ \text{subject to} \quad & y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant 1 - \xi_n \\ & \xi_n \geqslant 0 \end{aligned}$$

Soft Margin SVM: Loss Function View §12.2.5

We choose the hyperplane as $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$

Hinge loss $\ell(t) = \max\{0, 1 - t\}$ where $t = yf(\boldsymbol{x}) = y(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)$

1. If f(x) is on the correct side of the hyperplane, and further than distance 1 → t ⩾ 1 ⟶ l=0
2. If f(x) is on the correct side of the hyperplane, but within the margin ⟶ 0<t<1 ⟶ l >0
3. If f(x) is on the opposite side of the hyperplane ⟶ t<0 ⟶ l >0
   l larger than in 2



Using hinge loss gives us the unconstrained optimization problem

$$\min_{\boldsymbol{w},b} \underbrace{\frac{1}{2}\|\boldsymbol{w}\|^2}_{\text{regularizer}} + \underbrace{C\sum_{n=1}^{N} \max\{0, 1 - y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b)\}}_{\text{error term}} .$$

Lagrange multiplier refresh

- The Lagrange multiplier technique lets you find the maximum or minimum of a multivariable function $f(x, y, \ldots)$ when there is some constraint on the input values you are allowed to use.

- This technique only applies to constraints that look something like this:

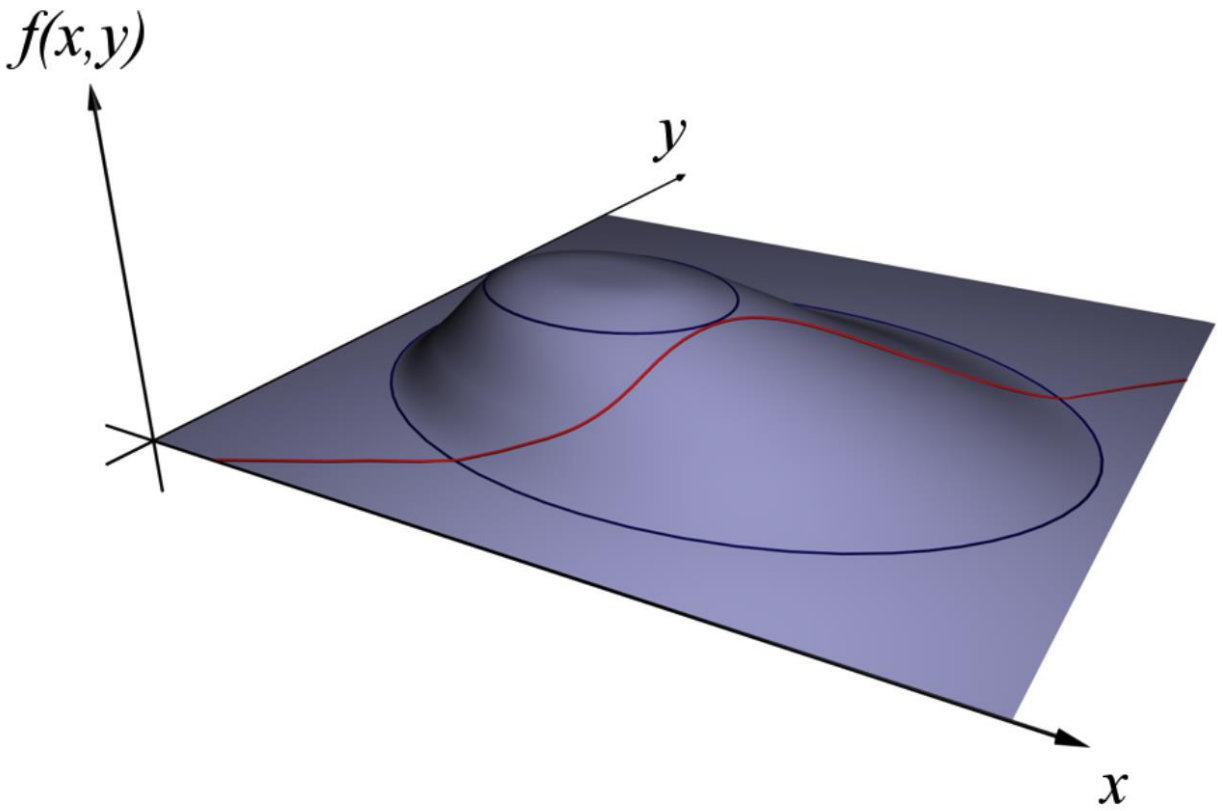
$$g(x, y, \ldots) = c$$

Here, g is another multivariable function with the same input space as f, and c is some constant. Picture

https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint

Lagrange multiplier refresh

For example, if the input space is two-dimensional, the graph of f with the line representing $g(x, y) = c$ projected onto it might look something like this:



https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint

Lagrange multiplier refresh

- **Step 1**: Introduce a new variable λ, and define a new function $\mathcal{L}$ as follows:

  $$\mathcal{L}(x, y, \ldots, \lambda) = f(x, y, \ldots) - \lambda(g(x, y, \ldots) - c)$$

  This function $\mathcal{L}$ is called the "Lagrangian", and the new variable λ is referred to as a "Lagrange multiplier"

- **Step 2**: Set the gradient of $\mathcal{L}$ equal to the zero vector.

  $$\nabla\mathcal{L}(x, y, \ldots, \lambda) = \mathbf{0} \quad \leftarrow \text{Zero vector}$$

  In other words, find the **critical points** of $\mathcal{L}$.

- **Step 3**: Consider each solution, which will look something like $(x_0, y_0, \ldots, \lambda_0)$. Plug each one into f. Or rather, first remove the λ_0 component, then plug it into f, since f does not have λ as an input. Whichever one gives the greatest (or smallest) value is the maximum (or minimum) point your are seeking.

https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint

Convex Duality via Lagrange Multipliers §12.3.1

Lagrange multiplier $\alpha_n \geqslant 0$ corresponding to the constraint $y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geqslant 1 - \xi_n$

Lagrange multiplier $\gamma_n \geqslant 0$ corresponding to the constraint of non-negativity of the slack variable $\xi_n \geqslant 0$

$$\mathfrak{L}(\boldsymbol{w}, b, \xi, \alpha, \gamma) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \qquad (12.34)$$

$$\underbrace{-\sum_{n=1}^{N}\alpha_n(y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) - 1 + \xi_n)}_{\text{constraint (12.26b)}} \underbrace{-\sum_{n=1}^{N}\gamma_n\xi_n}_{\text{constraint (12.26c)}} .$$

Differentiate the Lagrangian with respect to the three parameters

$$\frac{\partial \mathfrak{L}}{\partial \boldsymbol{w}} = \boldsymbol{w}^\top - \sum_{n=1}^{N}\alpha_n y_n \boldsymbol{x}_n{}^\top \qquad (12.35)$$

Convex Duality via Lagrange Multipliers

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

$$\frac{\partial \mathfrak{L}}{\partial \xi_n} = C - \alpha_n - \gamma_n \,.$$

By setting 12.35 to 0 $\boldsymbol{w} = \sum_{n=1}^{N} \alpha_n y_n \boldsymbol{x}_n$

Substituting the expression for w into the Lagrangian

$$\mathfrak{D}(\xi, \alpha, \gamma) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle - \sum_{i=1}^{N} y_i \alpha_i \left\langle \sum_{j=1}^{N} y_j \alpha_j \boldsymbol{x}_j, \boldsymbol{x}_i \right\rangle$$
$$+ C \sum_{i=1}^{N} \xi_i - b \sum_{i=1}^{N} y_i \alpha_i + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i \xi_i - \sum_{i=1}^{N} \gamma_i \xi_i \,. \tag{12.39}$$

Convex Duality via Lagrange Multipliers

Recall that inner products are symmetric and bilinear, then the terms in blue in 12.39 can be simplified

$$\mathfrak{D}(\xi, \alpha, \gamma) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \sum_{i=1}^{N} \alpha_i + \sum_{i=1}^{N} (C - \alpha_i - \gamma_i)\xi_i \,. \tag{12.40}$$

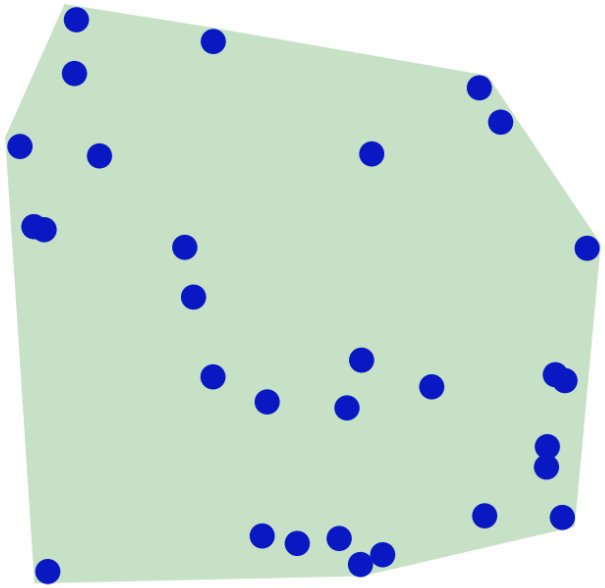
$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \sum_{i=1}^{N} \alpha_i \\
&\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0 \\
&0 \leqslant \alpha_i \leqslant C \quad \text{for all} \quad i = 1, \ldots, N \,.
\end{aligned}$$

Dual SVM: Convex Hull View §12.3.2

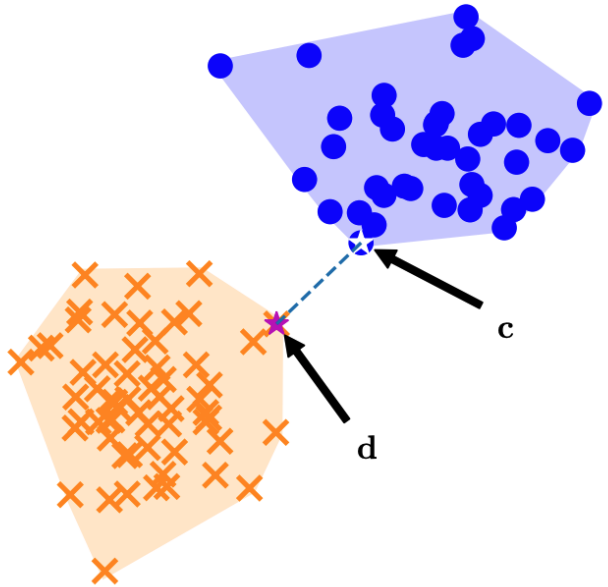
Consider the set of examples x_n with the same label. We would like to build a convex set that contains all the examples such that it is the smallest possible set. This is called the convex hull.

Building a convex convex hull can be done by introducing non-negative weights $\alpha n \geqslant 0$ corresponding to each example x_n

Then the convex hull can be described as the set $$\text{conv}(\boldsymbol{X}) = \left\{ \sum_{n=1}^{N} \alpha_n \boldsymbol{x}_n \right\} \text{ with } \sum_{n=1}^{N} \alpha_n = 1 \quad \text{and} \quad \alpha_n \geqslant 0, \quad (12.43)$$



(a) Convex hull.



(b) Convex hulls around positive (blue) and negative (orange) examples. The distance be-

We create convex hulls for positive and negative classes, respectively.

Dual SVM: Convex Hull View

We pick a point **c**, which is in the convex hull of the set of positive examples, and is closest to the negative class distribution.

We pick a point **d** in the convex hull of the set of negative examples and is closest to the positive class distribution;

We define a difference vector between d and c as $\boldsymbol{w} := \boldsymbol{c} - \boldsymbol{d}$.

Corresponding optimization problem $\arg\min_{\boldsymbol{w}} \|\boldsymbol{w}\| = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2$.

c can be expressed as a convex combination of the positive examples $\boldsymbol{c} = \sum_{n:y_n=+1} \alpha_n^+ \boldsymbol{x}_n$.

Similarly for the d $\boldsymbol{d} = \sum_{n:y_n=-1} \alpha_n^- \boldsymbol{x}_n$.

Objective $\min_{\boldsymbol{\alpha}} \frac{1}{2} \left\| \sum_{n:y_n=+1} \alpha_n^+ \boldsymbol{x}_n - \sum_{n:y_n=-1} \alpha_n^- \boldsymbol{x}_n \right\|^2$.

Dual SVM: Convex Hull View

Let α be the set of all coefficients, i.e., the concatenation of α+ and α−.

$$\sum_{n:y_n=+1} \alpha_n^+ = 1 \quad \text{and} \quad \sum_{n:y_n=-1} \alpha_n^- = 1\,.$$

This implies the constraint $\sum_{n=1}^{N} y_n \alpha_n = 0\,.$

Could be solved using Lagrangian!

But we won't solve it here ☺

Kernels §12.4

Recall the dual SVM formulation. The inner products are only between examples, not parameters. This means you can modify the representation of the data and only need to replace the inner product values.

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \sum_{i=1}^{N} \alpha_i$$

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0 \qquad (12.41)$$

$$0 \leqslant \alpha_i \leqslant C \quad \text{for all} \quad i = 1, \ldots, N\,.$$

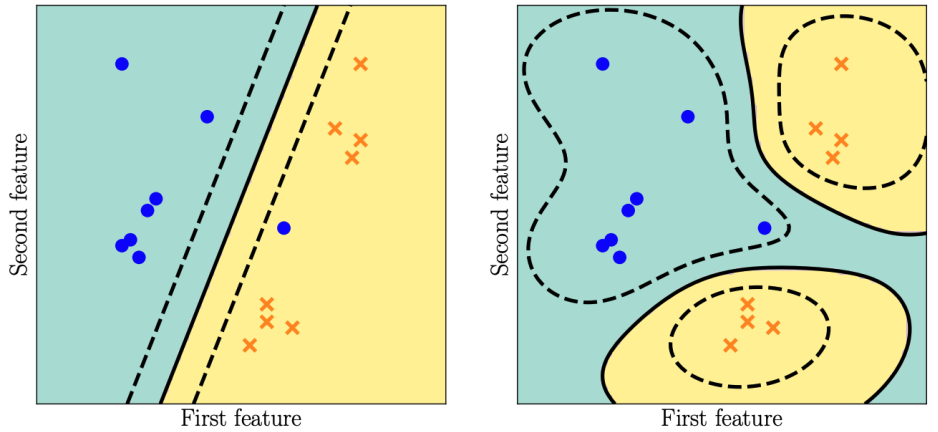
Since ϕ(x) could be a non-linear function, we can use the SVM (which assumes a linear classifier) for separating non-linear data.

We will define a similarity function *k*, between samples, without explicitly defining the non-linear feature map ϕ(x)

Kernels are by definition functions $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

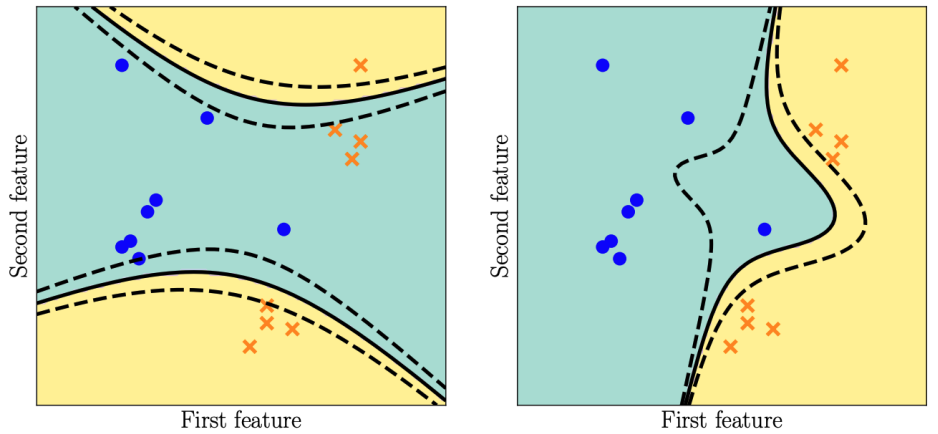
for which there is a Hilbert space $\mathcal{H}$ and $\phi : \mathcal{X} \to \mathcal{H}$

Is a feature map such that $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}$.



(a) SVM with linear kernel

(b) SVM with RBF kernel



(c) SVM with polynomial (degree 2) kernel

(d) SVM with polynomial (degree 3) kernel

Numerical Solution §12.5

The loss view of SVM is a convex unconstrained optimization problem
But the hinge loss (12.28) is not differentiable, so the subgradient approach is used to solve it

Both the primal and dual SVM result in a convex quadratic programming problem

- The primal SVM has optimization variables that have the size *D* of the dimension of samples
- The dual SVM has optimization variables that have the size *N* of the number of samples

The primal soft margin SVM in matrix form can be expressed as:

$$\begin{aligned}
&\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\begin{bmatrix}\boldsymbol{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{I}_D & \boldsymbol{0}_{D,N+1}\\ \boldsymbol{0}_{N+1,D} & \boldsymbol{0}_{N+1,N+1}\end{bmatrix}\begin{bmatrix}\boldsymbol{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix} + \begin{bmatrix}\boldsymbol{0}_{D+1,1} & C\boldsymbol{1}_{N,1}\end{bmatrix}^\top \begin{bmatrix}\boldsymbol{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix}\\
&\text{subject to} \quad \begin{bmatrix}-\boldsymbol{Y}\boldsymbol{X} & -\boldsymbol{y} & -\boldsymbol{I}_N\\ \boldsymbol{0}_{N,D+1} & & -\boldsymbol{I}_N\end{bmatrix}\begin{bmatrix}\boldsymbol{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix} \leqslant \begin{bmatrix}-\boldsymbol{1}_{N,1}\\ \boldsymbol{0}_{N,1}\end{bmatrix}.
\end{aligned} \tag{12.56}$$

The dual SVM can be written as:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{Y}\boldsymbol{K}\boldsymbol{Y}\boldsymbol{\alpha} - \boldsymbol{1}_{N,1}^\top \boldsymbol{\alpha}\\
&\text{subject to} \quad \begin{bmatrix}\boldsymbol{y}^\top\\ -\boldsymbol{y}^\top\\ -\boldsymbol{I}_N\\ \boldsymbol{I}_N\end{bmatrix}\boldsymbol{\alpha} \leqslant \begin{bmatrix}\boldsymbol{0}_{N+2,1}\\ C\boldsymbol{1}_{N,1}\end{bmatrix}.
\end{aligned} \tag{12.57}$$