Mathematics for Machine Learning

— Classification with Support Vector Machines

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

Fall 2025

Credits for the resource

- The slides are based on the textbooks:
  - *Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong: Mathematics for Machine Learning. Cambridge University Press. 2020.*
  - *Arnold J. Insel, Lawrence E. Spence, Stephen H. Friedberg: Linear Algebra, 4th Edition. Prentice Hall. 2013.*
  - *Howard Anton, Chris Rorres, Anton Kaul: Elementary Linear Algebra, 12th Edition. Wiley. 2019.*
- We could partially refer to the monograph:
  Francesco Orabona: A Modern Introduction to Online Learning.
  https://arxiv.org/abs/1912.13213

Outline

Outline

Binary Classification

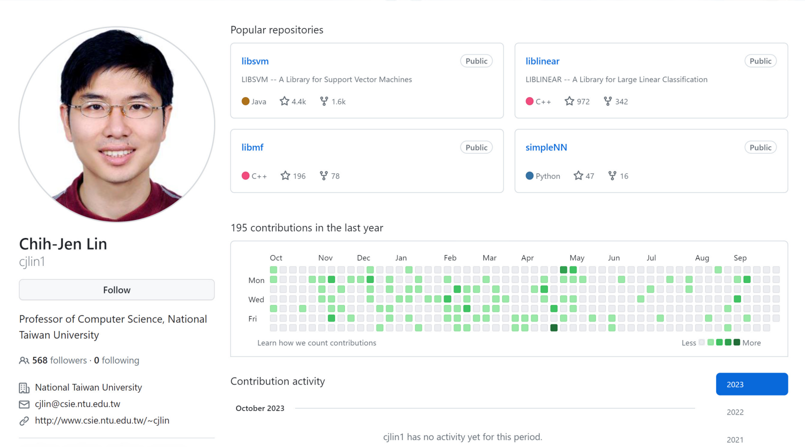
- **Focus:** predictors of the form:

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

- **Given:** a set of example-label pairs $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ as the training dataset.
- **Goal:** a model of parameters giving the smallest classification error.

The model: Hyperplane (an affine subspace of dimension $D - 1$).

Chih-Jen Lin's libsvm (https://github.com/cjlin1)



Purpose of Using SVM

- SVM allows for a geometric way of thinking (supervised learning).
- Resort to a variety of optimization tools.

Outline

Separating Hyperplanes

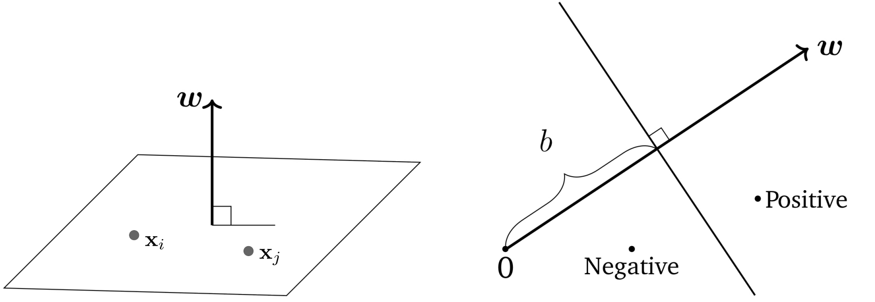
Separating Hyperplane

- Consider a function $f : \mathbb{R}^D \to \mathbb{R}$ such that

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b,$$

  parametrized by $\mathbf{w} \in \mathbb{R}^D$ and $b \in \mathbb{R}$.
- We define the hyperplane that separates the two classes in the binary classification problem as

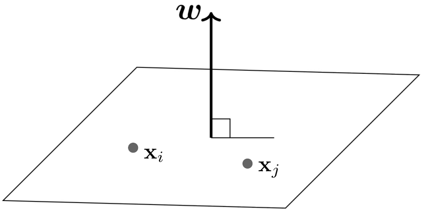
$$\{\mathbf{x} \in \mathbb{R}^D : f(\mathbf{x}) = 0\}.$$



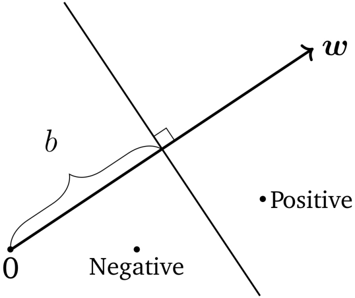
(a) Separating hyperplane in 3D

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

- **w**: a normal vector to the hyperplane (?)

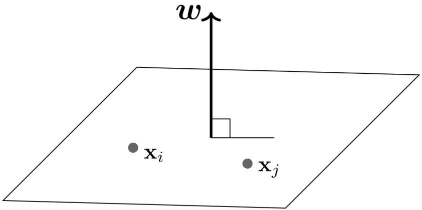


(a) Separating hyperplane in 3D

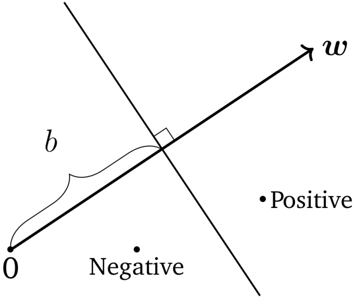


(b) Projection of the setting in (a) onto a plane

- $\mathbf{w}$: a normal vector to the hyperplane (?)
- $f(\mathbf{x}_i) = f(\mathbf{x}_j) = 0 \quad \& \quad \mathbf{w} \perp (\mathbf{x}_i - \mathbf{x}_j)$ (?)



(a) Separating hyperplane in 3D



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

- $\mathbf{w}$: a normal vector to the hyperplane (?)
- $f(\mathbf{x}_i) = f(\mathbf{x}_j) = 0 \quad \& \quad \mathbf{w} \perp (\mathbf{x}_i - \mathbf{x}_j)$ (?)
  - $f(\mathbf{x}_i) - f(\mathbf{x}_j) = \langle \mathbf{w}, \mathbf{x}_i \rangle + b - (\langle \mathbf{w}, \mathbf{x}_j \rangle + b) = \langle \mathbf{w}, \mathbf{x}_i - \mathbf{x}_j \rangle$

Classifier: Separating Hyperplanes

Ensure that the examples with positive labels are on the positive side of the hyperplane.

$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 0 \text{ when } y_i = +1.$$

Ensure that the examples with negative labels are on the negative side of the hyperplane.

$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b < 0 \text{ when } y_i = -1.$$

Classifier: Separating Hyperplanes

Ensure that the examples with positive labels are on the positive side of the hyperplane.

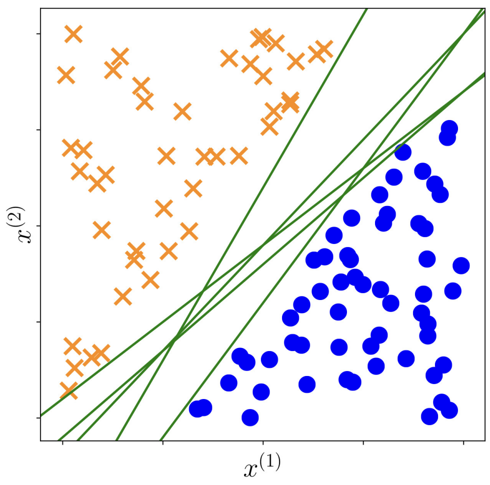
$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 0 \text{ when } y_i = +1.$$

Ensure that the examples with negative labels are on the negative side of the hyperplane.

$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b < 0 \text{ when } y_i = -1.$$

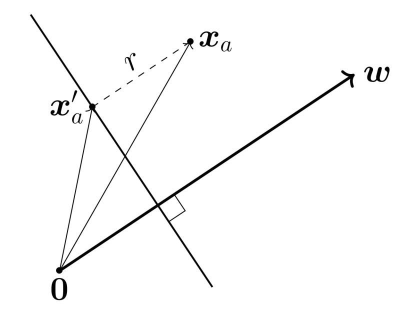
- These two conditions $\iff$ $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 0$.

Possible Separating Hyperplanes



Outline

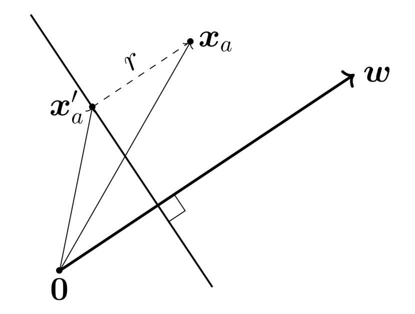
Concept of the Margin



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

- We can choose $\mathbf{w}$ of unit length: $\|\mathbf{w}\| = 1$ to simplify our discussion.
  - The Euclidean norm: $\|\mathbf{w}\| = \sqrt{\mathbf{w}^\top \mathbf{w}}$.

Concept of the Margin



$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq r.$$

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

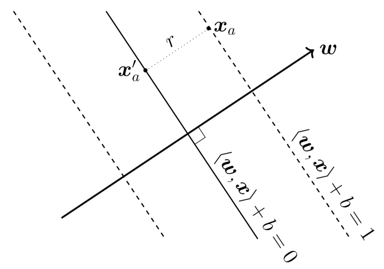
- We can choose $\mathbf{w}$ of unit length: $\|\mathbf{w}\| = 1$ to simplify our discussion.
    - The Euclidean norm: $\|\mathbf{w}\| = \sqrt{\mathbf{w}^\top \mathbf{w}}$.
- We choose $\mathbf{x}_a$ to be the point closest to the hyperplane, and the distance r is the margin.

One single constrained optimization problem

$$\max_{\mathbf{w},b,r} \underbrace{r}_{\text{margin}}$$

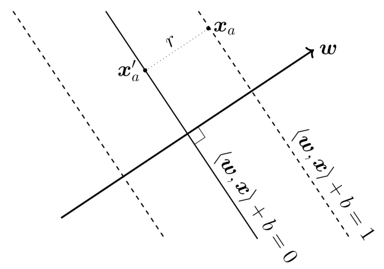
$$\text{subject to} \quad \underbrace{y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq r}_{\text{data fitting}}, \ \underbrace{\|\mathbf{w}\| = 1}_{\text{normalization}}, r > 0.$$

An alternative explanation



- Rescale the data such that $\langle \mathbf{w}, \mathbf{x} \rangle + b = 1$ at the closest example $\mathbf{x}$.

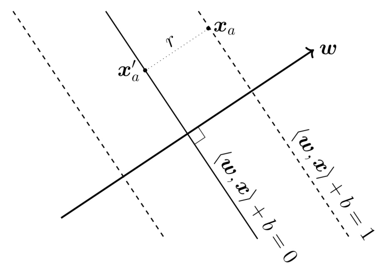
An alternative explanation



- Rescale the data such that $\langle \mathbf{w}, \mathbf{x} \rangle + b = 1$ at the closest example $\mathbf{x}$.
- $\mathbf{x}'_a$ is the orthogonal projection of $\mathbf{x}_a$ onto the hyperplane

$$\langle \mathbf{w}, \mathbf{x}'_a \rangle + b = 0.$$

An alternative explanation

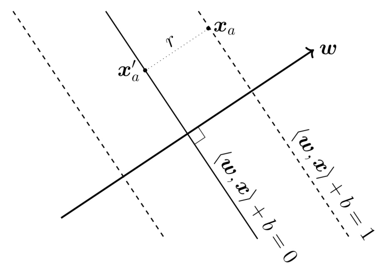


- Rescale the data such that $\langle \mathbf{w}, \mathbf{x} \rangle + b = 1$ at the closest example $\mathbf{x}$.
- $\mathbf{x}'_a$ is the orthogonal projection of $\mathbf{x}_a$ onto the hyperplane

$$\langle \mathbf{w}, \mathbf{x}'_a \rangle + b = 0.$$

$$\left\langle \mathbf{w}, \mathbf{x}_a - r\frac{\mathbf{w}}{\|\mathbf{w}\|} \right\rangle + b = 0.$$

An alternative explanation



- Rescale the data such that $\langle \mathbf{w}, \mathbf{x}\rangle + b = 1$ at the closest example $\mathbf{x}$.
- $\mathbf{x}'_a$ is the orthogonal projection of $\mathbf{x}_a$ onto the hyperplane

$$\langle \mathbf{w}, \mathbf{x}'_a\rangle + b = 0.$$

$$\left\langle \mathbf{w}, \mathbf{x}_a - r\frac{\mathbf{w}}{\|\mathbf{w}\|}\right\rangle + b = 0.$$

$$\langle \mathbf{w}, \mathbf{x}_a\rangle + b - r\frac{\langle \mathbf{w}, \mathbf{w}\rangle}{\|\mathbf{w}\|} = 0$$

$$\Rightarrow \quad r = \frac{1}{\|\mathbf{w}\|}.$$

Remark

We will show that setting the margin $r = \frac{1}{\|\mathbf{w}\|}$ to be 1 is equivalent to assuming $\|\mathbf{w}\| = 1$.

Combining the Two Conditions

$$\max_{\mathbf{w},b} \quad \frac{1}{\|\mathbf{w}\|}$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \;\; \text{for all} \;\; i = 1, \ldots, N.$$

Combining the Two Conditions

$$\max_{\mathbf{w},b} \quad \frac{1}{\|\mathbf{w}\|}$$
$$\text{subject to} \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \;\text{ for all }\; i = 1, \ldots, N.$$

Instead, we often do the minimization:

Hard Margin SVM

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{subject to} \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \;\text{ for all }\; i = 1, \ldots, N.$$

- "Hard": no violation of the margin condition is allowed.

Why We Can Set the Margin to 1? (1/3)

Recall the original setting:

$$\max_{\mathbf{w},b,r} \quad \underbrace{r}_{\text{margin}}$$

$$\text{subject to} \quad \underbrace{y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq r,}_{\text{data fitting}} \ \underbrace{\|\mathbf{w}\| = 1}_{\text{normalization}}, r > 0.$$

Reparametrize the equation with a new weight vector $\mathbf{w}'$:

$$\max_{\mathbf{w}',b,r} \quad r^2$$

$$\text{subject to} \quad y_i \left(\left\langle \frac{\mathbf{w}'}{\|\mathbf{w}'\|}, \mathbf{x}_i \right\rangle + b \right) \geq r, r > 0.$$

Why We Can Set the Margin to 1? (2/3)

Reparametrize the equation with a new weight vector $\mathbf{w}'$:

$$\max_{\mathbf{w},b,r} \quad r^2$$

$$\text{subject to} \quad y_i\left(\left\langle \frac{\mathbf{w}'}{\|\mathbf{w}'\|}, \mathbf{x}_i \right\rangle + b\right) \geq r, r > 0.$$

Divide the constraint by r:

$$\max_{\mathbf{w}',b,r} \quad r^2$$

$$\text{subject to} \quad y_i\left(\left\langle \frac{\mathbf{w}'}{\|\mathbf{w}'\|r}, \mathbf{x}_i \right\rangle + \frac{b}{r}\right) \geq 1, r > 0.$$

$\mathbf{w}'' = \mathbf{w}'/(\|\mathbf{w}'\|r)$, $b'' = b/r$.

Why We Can Set the Margin to 1? (2/3)

Reparametrize the equation with a new weight vector $\mathbf{w}'$:

$$\max_{\mathbf{w},b,r} \quad r^2$$

$$\text{subject to} \quad y_i \left(\left\langle \frac{\mathbf{w}'}{\|\mathbf{w}'\|}, \mathbf{x}_i \right\rangle + b \right) \geq r, r > 0.$$

Divide the constraint by r:

$$\max_{\mathbf{w}',b,r} \quad r^2$$

$$\text{subject to} \quad y_i \left(\left\langle \frac{\mathbf{w}'}{\|\mathbf{w}'\| r}, \mathbf{x}_i \right\rangle + \frac{b}{r} \right) \geq 1, r > 0.$$

$\mathbf{w}'' = \mathbf{w}'/(\|\mathbf{w}'\| r)$, $b'' = b/r$. So, $\|\mathbf{w}''\| = 1/r$.

Why We Can Set the Margin to 1? (3/3)

Finally,

$$\max_{\mathbf{w}'', b''} \quad \frac{1}{\|\mathbf{w}''\|^2}$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}'', \mathbf{x}_i \rangle + b'') \geq 1.$$

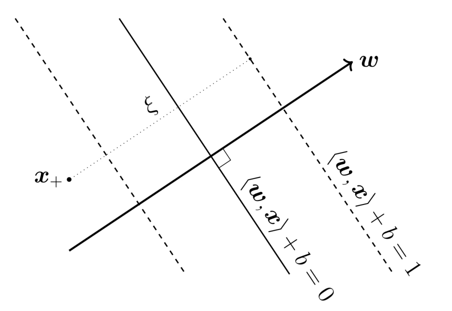
That is,

$$\min_{\mathbf{w}'', b''} \quad \frac{1}{2}\|\mathbf{w}''\|^2$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}'', \mathbf{x}_i \rangle + b'') \geq 1.$$

Outline

Soft Margin?



- When the data is NOT linearly separable, we wish to allow some examples to fall within the margin region.
- We subtract the value ξ_i from the margin, constraining ξ_i to be non-negative.
- Purpose: Encourage correct classification

Add ξ_i's to the objective, we get

The Soft Margin SVM

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0$$

for $i = 1, \ldots, N$.

C: regularization parameter. $\|\mathbf{w}\|^2$: the regularizer.

Outline

Primal SVM

- The primal SVM: the SVM in terms of variables $\mathbf{w}$ and b.
- The input $\mathbf{x} \in \mathbb{R}^D$ with D features, while $\mathbf{w}$ has the same dimension as $\mathbf{x}$.
  - The number of parameters grows linearly with the number of features.

Equivalent Optimization Problem: The Dual View

- We consider the dual problem: Dual Support SVM, which is independent of the number of features.

Equivalent Optimization Problem: The Dual View

- We consider the dual problem: Dual Support SVM, which is independent of the number of features.
- An additional advantage: Allow kernels to be applied easily.

Outline

Convex Duality

- We use $\alpha_i \geq 0$ and $\gamma_i \geq 0$ as the Lagrange multipliers.
  - α_i: w.r.t. the constraint that examples are correctly classified.
  - γ_i: w.r.t. the non-negativity constraint of the slack variable.

$$\begin{aligned}\mathfrak{L}(\mathbf{w}, b, \xi, \alpha, \gamma) \quad := \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\ & -\sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i\end{aligned}$$

Convex Duality

- We use $\alpha_i \geq 0$ and $\gamma_i \geq 0$ as the Lagrange multipliers.
  - α_i: w.r.t. the constraint that examples are correctly classified.
  - γ_i: w.r.t. the non-negativity constraint of the slack variable.

$$
\begin{aligned}
\mathfrak{L}(\mathbf{w}, b, \xi, \alpha, \gamma) \quad := \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\
& - \sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i
\end{aligned}
$$

- Then we derive the partial derivatives of $\mathfrak{L}$ w.r.t $\mathbf{w}$, b and ξ_i for all i.

Partial Derivatives of the Lagrangian

$$\mathfrak{L} = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i$$

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} &= \mathbf{w}^\top - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i^\top \\
\frac{\partial \mathfrak{L}}{\partial b} &= -\sum_{i=1}^{N}\alpha_i y_i \\
\frac{\partial \mathfrak{L}}{\partial \xi_i} &= C - \alpha_i - \gamma_i
\end{aligned}$$

Partial Derivatives of the Lagrangian

$$\mathfrak{L} = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i$$

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} &= \mathbf{w}^\top - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i^\top \\
\frac{\partial \mathfrak{L}}{\partial b} &= -\sum_{i=1}^{N}\alpha_i y_i \\
\frac{\partial \mathfrak{L}}{\partial \xi_i} &= C - \alpha_i - \gamma_i
\end{aligned}$$

- Maximizing the Lagrangian by setting $\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} = \mathbf{0}^\top$,

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i.$$

Partial Derivatives of the Lagrangian

$$\mathfrak{L} = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i$$

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} &= \mathbf{w}^\top - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i^\top \\
\frac{\partial \mathfrak{L}}{\partial b} &= -\sum_{i=1}^{N}\alpha_i y_i \\
\frac{\partial \mathfrak{L}}{\partial \xi_i} &= C - \alpha_i - \gamma_i
\end{aligned}$$

- Maximizing the Lagrangian by setting $\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} = \mathbf{0}^\top$,

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i.$$

- The optimal weight vector is a linear combination of the examples $\mathbf{x}_i$'s.

Partial Derivatives of the Lagrangian

$$\mathfrak{L} = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i$$

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} &= \mathbf{w}^\top - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i^\top \\
\frac{\partial \mathfrak{L}}{\partial b} &= -\sum_{i=1}^{N}\alpha_i y_i \\
\frac{\partial \mathfrak{L}}{\partial \xi_i} &= C - \alpha_i - \gamma_i
\end{aligned}$$

- Maximizing the Lagrangian by setting $\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} = \mathbf{0}^\top$,
$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i.$$
- The optimal weight vector is a linear combination of the examples $\mathbf{x}_i$'s.
- $\mathbf{x}_i$'s with $\alpha_i > 0$: support vectors.

Substituting the expression for **w** into the Lagrangian, we have

$$\begin{aligned}\mathfrak{D}(\xi,\alpha,\gamma) \quad &:= \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^{N} y_i \alpha_i \left\langle \sum_{j=1}^{N} y_j \alpha_j \mathbf{x}_j, \, \mathbf{x}_i \right\rangle \\ &\quad + C\sum_{i=1}^{N} \xi_i - b\sum_{i=1}^{N} y_i \alpha_i - \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i \xi_i - \sum_{i=1}^{N} \gamma_i \xi_i .\end{aligned}$$

Substituting the expression for **w** into the Lagrangian, we have

$$\begin{aligned}\mathfrak{D}(\xi, \alpha, \gamma) \quad := \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^{N} y_i \alpha_i \left\langle \sum_{j=1}^{N} y_j \alpha_j \mathbf{x}_j, \mathbf{x}_i \right\rangle \\ & + C\sum_{i=1}^{N} \xi_i - b\sum_{i=1}^{N} y_i \alpha_i - \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i \xi_i - \sum_{i=1}^{N} \gamma_i \xi_i.\end{aligned}$$

- No terms involving the primal variable **w**.

Partial Derivatives of the Lagrangian

$$\mathfrak{L} = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) - 1 + \xi_i) - \sum_{i=1}^{N}\gamma_i\xi_i$$

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial \mathbf{w}} &= \mathbf{w}^\top - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i^\top \\
\frac{\partial \mathfrak{L}}{\partial b} &= -\sum_{i=1}^{N}\alpha_i y_i \\
\frac{\partial \mathfrak{L}}{\partial \xi_i} &= C - \alpha_i - \gamma_i
\end{aligned}$$

- Maximizing the Lagrangian by setting $\frac{\partial \mathfrak{L}}{\partial b} = 0$,

$$\sum_{i=1}^{N}\alpha_i y_i = 0.$$

With terms simplified, we obtain the Lagrangian

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

Setting $\frac{\partial \mathfrak{L}}{\partial \xi_i} = 0$, we see that

$$C = \alpha_i + \gamma_i \;\Rightarrow\; \sum_{i=1}^{N} (C - \alpha_i - \gamma_i) \xi_i = 0.$$

With terms simplified, we obtain the Lagrangian

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

Setting $\frac{\partial \mathfrak{L}}{\partial \xi_i} = 0$, we see that

$$C = \alpha_i + \gamma_i \Rightarrow \sum_{i=1}^{N} (C - \alpha_i - \gamma_i)\xi_i = 0.$$

Since $\gamma_i \geq 0$, we have that $\alpha_i \leq C$.

The Dual SVM

The Dual SVM

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

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0,$$

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

- $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_N]^\top \in \mathbb{R}^N$: Lagrange multipliers.
- The set of inequality constraints: box constraints.

The Dual SVM

The Dual SVM

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

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0,$$

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

Efficient to implement numerically!

- $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_N]^\top \in \mathbb{R}^N$: Lagrange multipliers.
- The set of inequality constraints: box constraints.

From Dual to Primal

- Once we obtain $\boldsymbol{\alpha}$, we can recover $\mathbf{w}$.
    - Recall that $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.

From Dual to Primal

- Once we obtain $\boldsymbol{\alpha}$, we can recover $\mathbf{w}$.
  - Recall that $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.
- Call the optimal primal parameter $\mathbf{w}^*$.

From Dual to Primal

- Once we obtain $\boldsymbol{\alpha}$, we can recover $\mathbf{w}$.
  - Recall that $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.
- Call the optimal primal parameter $\mathbf{w}^*$.
- Consider an example $\mathbf{x}_i$ that lies exactly on the margin's boundary: $y_i(\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) = 1$, where y_i is either $+1$ or -1.

From Dual to Primal

- Once we obtain $\boldsymbol{\alpha}$, we can recover $\mathbf{w}$.
  - Recall that $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.
- Call the optimal primal parameter $\mathbf{w}^*$.
- Consider an example $\mathbf{x}_i$ that lies exactly on the margin's boundary: $y_i(\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) = 1$, where y_i is either $+1$ or -1.
  - $\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^* = y_i$.

From Dual to Primal

- Once we obtain $\boldsymbol{\alpha}$, we can recover $\mathbf{w}$.
  - Recall that $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.
- Call the optimal primal parameter $\mathbf{w}^*$.
- Consider an example $\mathbf{x}_i$ that lies exactly on the margin's boundary: $y_i(\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) = 1$, where y_i is either $+1$ or -1.
  - $\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^* = y_i$.
  - Hence, we can compute $b^* = y_i - \langle \mathbf{w}^*, \mathbf{x}_i \rangle$.

From Dual to Primal

- Once we obtain $\boldsymbol{\alpha}$, we can recover $\mathbf{w}$.
  - Recall that $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.
- Call the optimal primal parameter $\mathbf{w}^*$.
- Consider an example $\mathbf{x}_i$ that lies exactly on the margin's boundary: $y_i(\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) = 1$, where y_i is either $+1$ or -1.
  - $\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^* = y_i$.
  - Hence, we can compute $b^* = y_i - \langle \mathbf{w}^*, \mathbf{x}_i \rangle$.

Remark

- The primal SVM: # optimization variables: feature dimension D.
- The dual SVM: # optimization variables: the number N of examples.

The Dual SVM

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

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0,$$

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

- We can see the inner product occurs only between examples. No inner products between examples and parameters!
- **Kernel trick:** consider $\phi(\mathbf{x}_i)$ to represent $\mathbf{x}_i$ ($\phi : \mathcal{X} \to \mathcal{H}$).

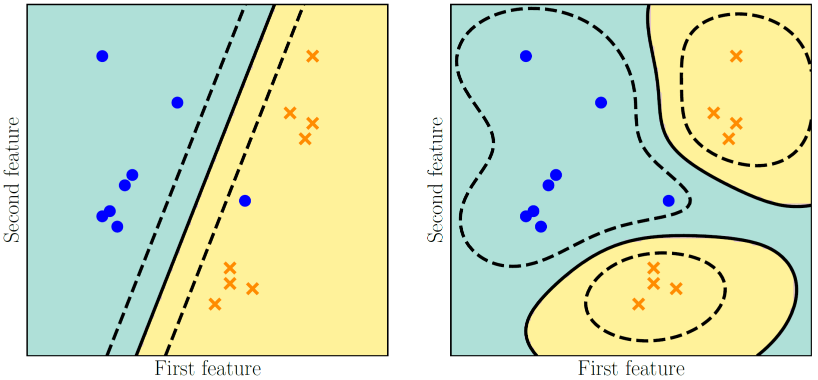
The Dual SVM

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

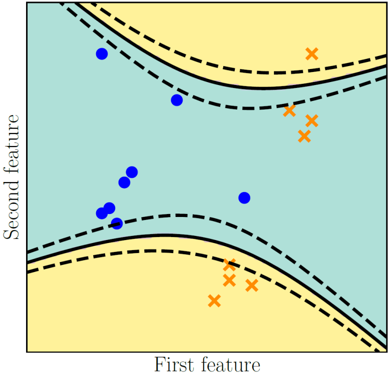
$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0,$$

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

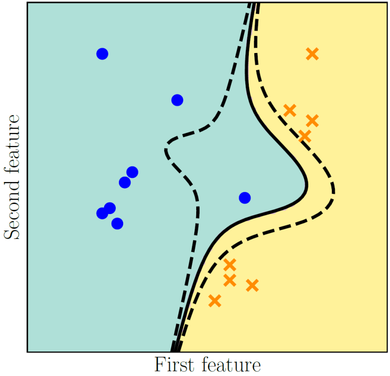
- We can see the inner product occurs only between examples. No inner products between examples and parameters!
- **Kernel trick:** consider $\phi(\mathbf{x}_i)$ to represent $\mathbf{x}_i$ ($\phi : \mathcal{X} \to \mathcal{H}$).
- Consider a similarity function $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}$ instead of defining $\phi(\cdot)$ and computing the resulting inner product.



(a) SVM with linear kernel (b) SVM with RBF kernel

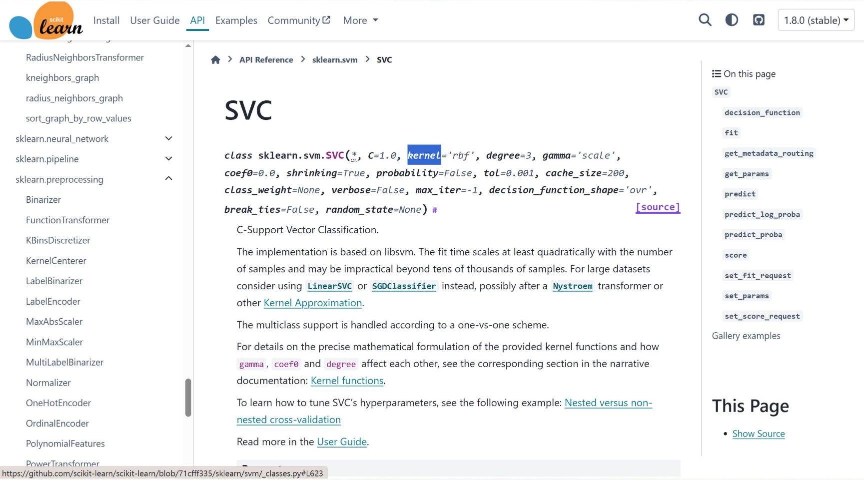


(c) SVM with polynomial (degree 2) kernel



(d) SVM with polynomial (degree 3) kernel

SVC — scikit-learn [link]



Install User Guide API Examples Community More 1.8.0 (stable)

RadiusNeighborsTransformer
kneighbors_graph
radius_neighbors_graph
sort_graph_by_row_values
sklearn.neural_network
sklearn.pipeline
sklearn.preprocessing
Binarizer
FunctionTransformer
KBinsDiscretizer
KernelCenterer
LabelBinarizer
LabelEncoder
MaxAbsScaler
MinMaxScaler
MultiLabelBinarizer
Normalizer
OneHotEncoder
OrdinalEncoder
PolynomialFeatures
PowerTransformer

API Reference > sklearn.svm > SVC

SVC

class sklearn.svm.**SVC**(*, *C=1.0*, *kernel='rbf'*, *degree=3*, *gamma='scale'*, *coef0=0.0*, *shrinking=True*, *probability=False*, *tol=0.001*, *cache_size=200*, *class_weight=None*, *verbose=False*, *max_iter=-1*, *decision_function_shape='ovr'*, *break_ties=False*, *random_state=None*) # [source]

C-Support Vector Classification.

The implementation is based on libsvm. The fit time scales at least quadratically with the number of samples and may be impractical beyond tens of thousands of samples. For large datasets consider using `LinearSVC` or `SGDClassifier` instead, possibly after a `Nystroem` transformer or other Kernel Approximation.

The multiclass support is handled according to a one-vs-one scheme.

For details on the precise mathematical formulation of the provided kernel functions and how `gamma`, `coef0` and `degree` affect each other, see the corresponding section in the narrative documentation: Kernel functions.

To learn how to tune SVC's hyperparameters, see the following example: Nested versus non-nested cross-validation

Read more in the User Guide.

On this page
SVC
decision_function
fit
get_metadata_routing
get_params
predict
predict_log_proba
predict_proba
score
set_fit_request
set_params
set_score_request
Gallery examples

This Page

- Show Source

https://github.com/scikit-learn/scikit-learn/blob/71cfff335/sklearn/svm/_classes.py#L623

SVC — scikit-learn [link]

Parameters:

C : ***float, default=1.0***

Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty. For an intuitive visualization of the effects of scaling the regularization parameter C, see Scaling the regularization parameter for SVCs.

kernel : ***{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'} or callable, default='rbf'***

Specifies the kernel type to be used in the algorithm. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape `(n_samples, n_samples)`. For an intuitive visualization of different kernel types see Plot classification boundaries with different SVM Kernels.

degree : ***int, default=3***

Degree of the polynomial kernel function ('poly'). Must be non-negative. Ignored by all other kernels.

gamma : ***{'scale', 'auto'} or float, default='scale'***

Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.

- if `gamma='scale'` (default) is passed then it uses 1 / (n_features * X.var()) as value of gamma,
- if 'auto', uses 1 / n_features
- if float, must be non-negative.

Outline

Revisit Soft SVM as an Example

The Soft Margin SVM

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i \\ \text{subject to} \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0 \end{aligned}$$

A revised form:

Revisit Soft SVM as an Example

The Soft Margin SVM

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i \\ \text{subject to} \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0 \end{aligned}$$

A revised form:

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i \\ \text{subject to} \quad & -y_i \mathbf{x}_i^\top \mathbf{w} - y_i b - \xi_i \leq -1, \\ & -\xi_i \leq 0 \end{aligned}$$

Concatenating the variables (Primal SVM)

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\begin{bmatrix}\mathbf{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix}^{\top}\begin{bmatrix}\boldsymbol{I}_D & \mathbf{0}_{D,N+1}\\ \mathbf{0}_{N+1,D} & \mathbf{0}_{N+1,N+1}\end{bmatrix}\begin{bmatrix}\mathbf{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix} + [\mathbf{0}_{D+1,1} \quad C\mathbf{1}_{N,1}]^{\top}\begin{bmatrix}\mathbf{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix}$$

$$\text{subject to } \begin{bmatrix}-\boldsymbol{YX} & -\mathbf{y} & -\boldsymbol{I}_N\\ \mathbf{0}_{N,D+1} & & -\boldsymbol{I}_N\end{bmatrix}\begin{bmatrix}\mathbf{w}\\ b\\ \boldsymbol{\xi}\end{bmatrix} \le \begin{bmatrix}-\mathbf{1}_{N,1}\\ \mathbf{0}_{N,1}\end{bmatrix}.$$

- $[\mathbf{w}^{\top}, b, \boldsymbol{\xi}^{\top}]^{\top} \in \mathbb{R}^{D+1+N}$.
- $\boldsymbol{I}_m \in \mathbb{R}^{m\times m}$: identity matrix.
- $\mathbf{0}_{m,n} \in \mathbb{R}^{m\times n}$: zeros of size $m \times n$, $\mathbf{1}_{m,n} \in \mathbb{R}^{m\times n}$: ones of size $m \times n$.
- $\mathbf{y} = [y_1, \cdots, y_N]^{\top}$
- $\boldsymbol{Y} = \text{diagonal}(\mathbf{y}) \in \mathbb{R}^{N\times N}$.
- $\boldsymbol{X} \in \mathbb{R}^{N\times D}$: concatenating all the examples.

Recall the Dual SVM

The Dual SVM

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

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0,$$

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

Concatenating the variables (Dual SVM)

K: kernel matrix for whch $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ (or simply $K_{ij} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$).

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{YKY}\boldsymbol{\alpha} - \mathbf{1}_{N,1}^\top \boldsymbol{\alpha}$$

$$\text{subject to} \quad \begin{bmatrix} \mathbf{y}^\top \\ -\mathbf{y}^\top \\ -\boldsymbol{I}_N \\ \boldsymbol{I}_N \end{bmatrix} \boldsymbol{\alpha} \leq \begin{bmatrix} \mathbf{0}_{N+2,1} \\ C\mathbf{1}_{N,1} \end{bmatrix}.$$

- Note that for equality constraints:

$$\boldsymbol{A}\mathbf{x} = \mathbf{b} \;\text{ is replaced by }\; \boldsymbol{A}\mathbf{x} \leq \mathbf{b} \text{ and } -\boldsymbol{A}\mathbf{x} \leq -\mathbf{b}.$$

Discussions