Support Vector Machines

Juho Lee
Grad school of AI
Samsung AI-Expert Course

Before we start

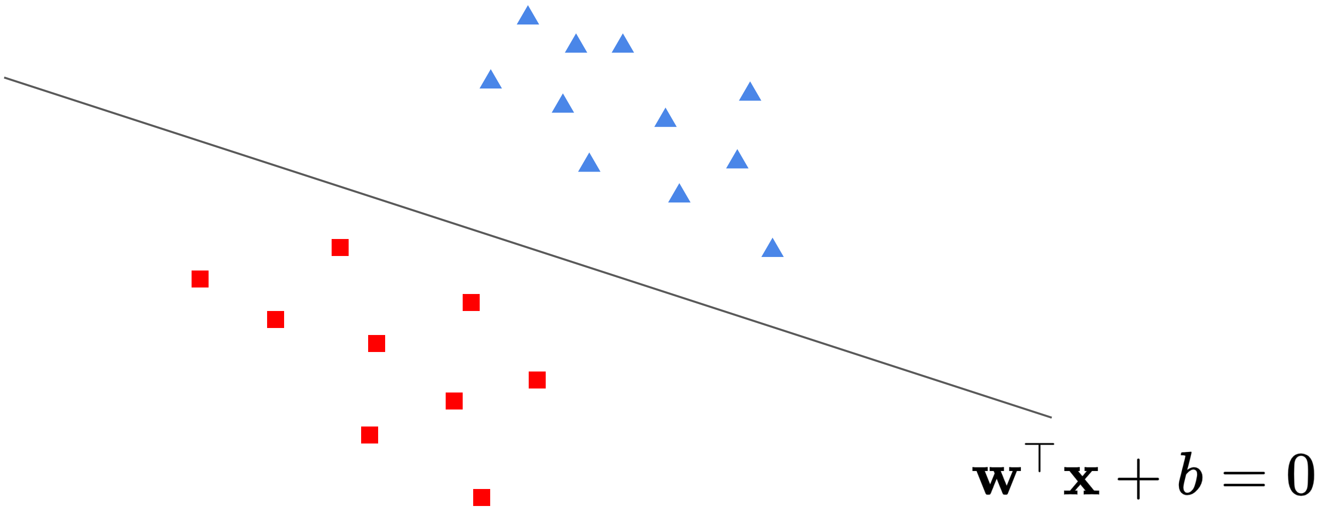
Clone the repository

`https://github.com/juho-lee/samsung_AI_expert`

You will find the update slide and codes.

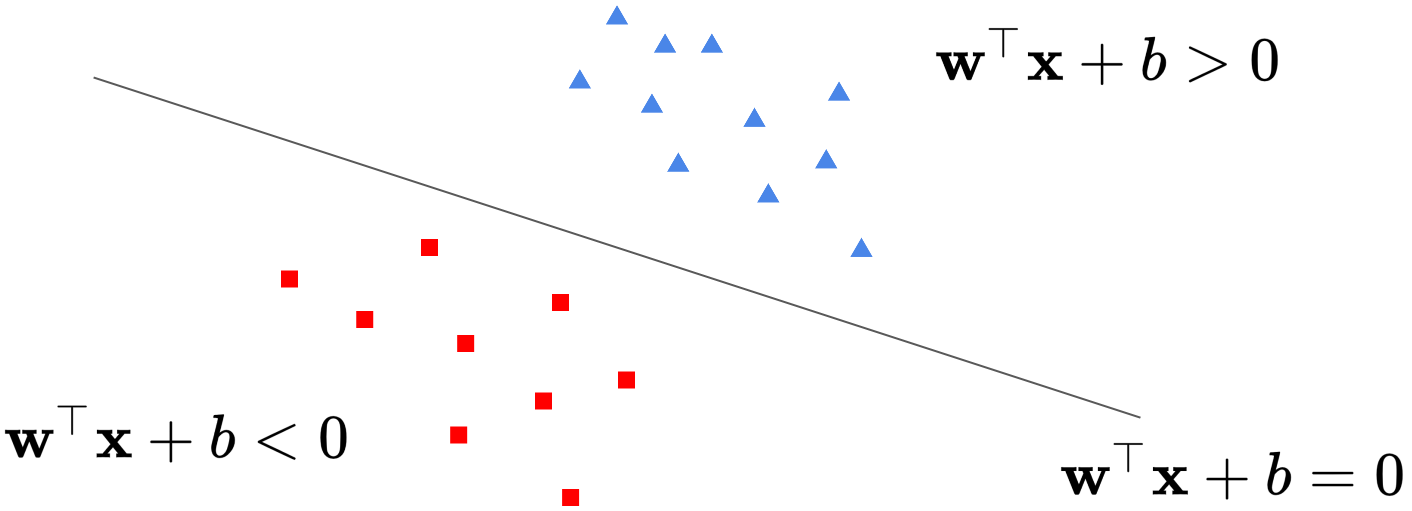
Maximum margin principle

Binary classification



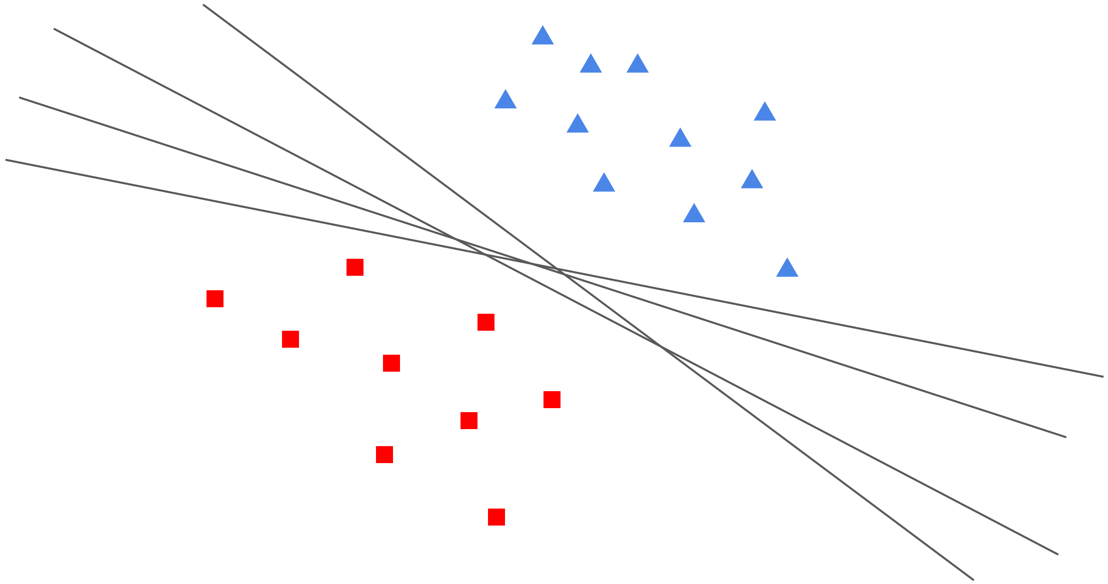
Binary classification

$$f(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^\top \mathbf{x} + b)$$



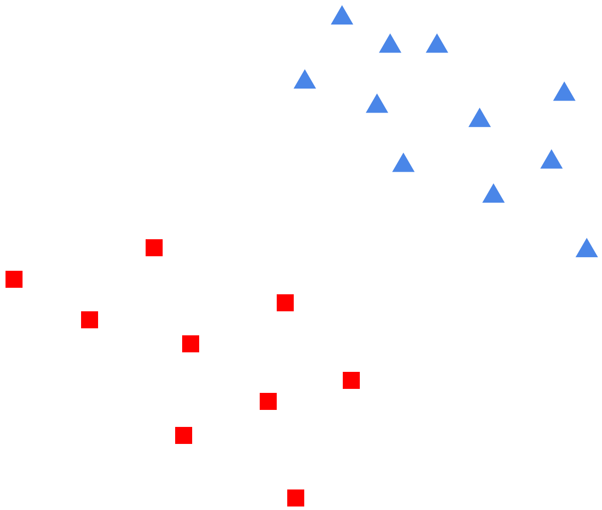
Binary classification

What makes a good hyperplane?



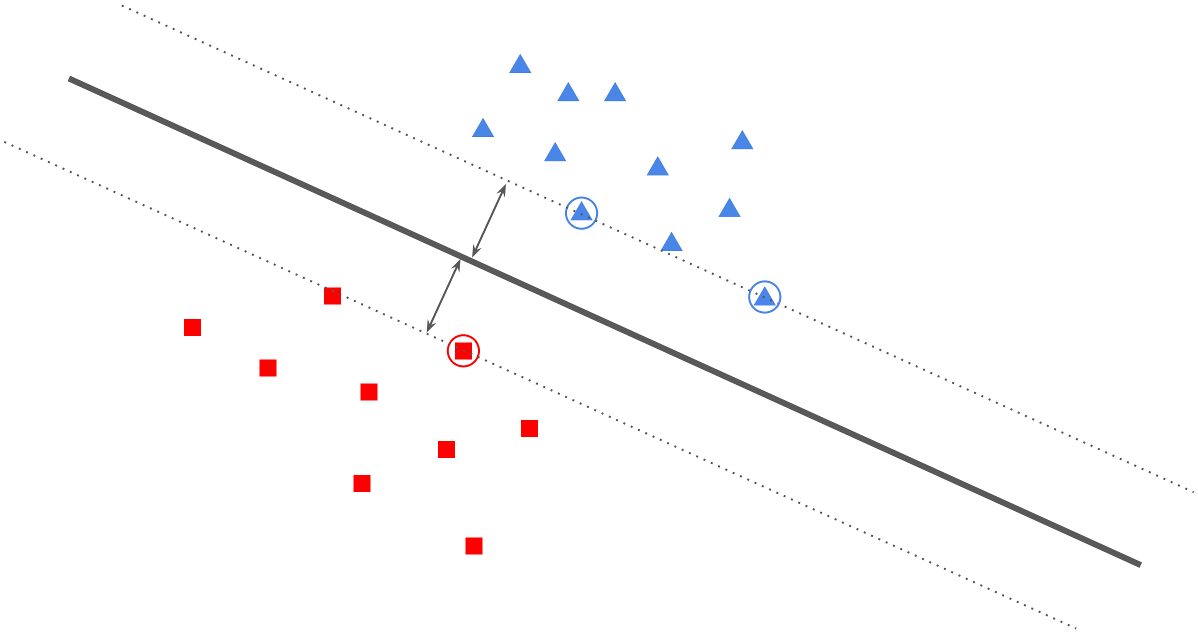
Binary classification

What makes a good hyperplane?



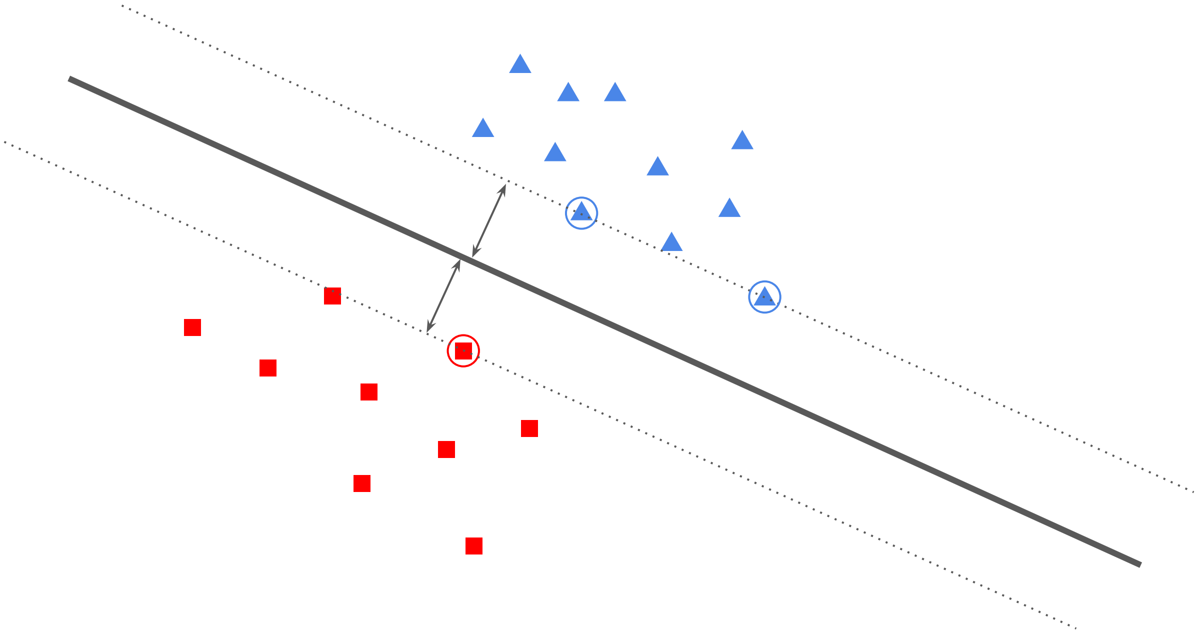
Binary classification

Margin: the distance between the hyperplane and the closest point



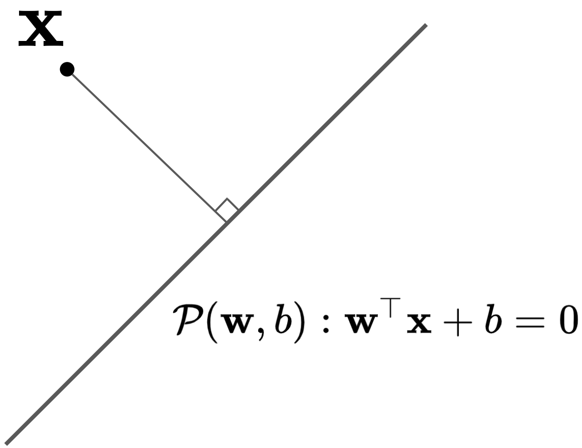
Binary classification

Support vector machine: find a hyperplane maximizing the margin.



Computing the margin

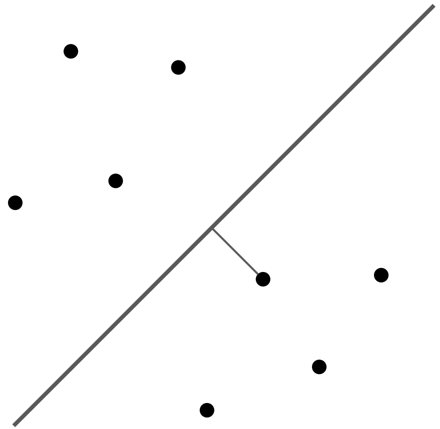
What is a distance between a point and a hyperplane?



$$\mathrm{dist}(\mathbf{x}, \mathcal{P}(\mathbf{w}, b)) = \frac{|\mathbf{w}^\top \mathbf{x} + b|}{||\mathbf{w}||}$$

Optimal hyperplane maximizing the margin

$$\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}, \quad y_i \in \{-1, 1\}$$



$$\mathcal{P} : \mathbf{w}^\top \mathbf{x} + b = 0$$

$$\gamma(\mathcal{D}) = \min_{i=1,...,n} \frac{y_i(\mathbf{w}^\top \mathbf{x}_i + b)}{\|\mathbf{w}\|}$$

$$\mathbf{w}^*, b^* = \underset{\mathbf{w},b}{\operatorname{argmax}}\ \gamma(\mathcal{D})$$

Observation: a hyperplane is invariant to the scaling

For an arbitrary constant k,

$$\mathcal{P}(\mathbf{w}, b) : \mathbf{w}^\top \mathbf{x} + b = 0.$$

$$\mathcal{P}(k\mathbf{w}, kb) : \mathbf{w}^\top \mathbf{x} + b = 0.$$

$$\mathrm{dist}(\mathbf{x}, \mathcal{P}(\mathbf{w}, b)) = \mathrm{dist}(\mathbf{x}, \mathcal{P}(k\mathbf{w}, kb)) = \frac{|\mathbf{w}^\top \mathbf{x} + b|}{\|\mathbf{w}\|}$$

SVM objective

Without loss of generality, one can scale such that

$$\gamma(\mathcal{D}) = \min_{i=1,...,n} \frac{y_i(\mathbf{w}^\top \mathbf{x}_i + b)}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}.$$

Then the optimization reduces to

$$\max_{\mathbf{w},b} \frac{1}{\|\mathbf{w}\|} \Longrightarrow \min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, n.$$

Lagrange duality

Lagrangian for constrained optimization problems

Consider a general constrained optimization problem.

$$\min_{\theta} \; f(\theta)$$

$$\begin{aligned} \text{s.t. } & g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, n \\ & h_j(\theta) = 0 \text{ for } j = 1, \ldots, m \end{aligned}$$

Lagrangian for constrained optimization problems

Define a **Lagrangian** as

$$\mathcal{L}(\theta, \alpha, \beta) = f(\theta) + \sum_{i=1}^{n} \alpha_i g_i(\theta) + \sum_{j=1}^{m} \beta_j h_j(\theta)$$

Then, the following is equivalent to the original problem (why?)

$$\min_{\theta} \ \max_{\alpha, \beta} \ \mathcal{L}(\theta, \alpha, \beta)$$

$$\text{s.t. } \alpha_i \geq 0 \text{ for } i = 1, \ldots, n,$$

We call this original problem as a “**primal**” problem.

Dual problem

Now define a “**dual**” problem where the order of min and max are switched.

$$\max_{\alpha,\beta} \min_{\theta} \mathcal{L}(\theta, \alpha, \beta)$$
$$\text{s.t. } \alpha_i \geq 0 \text{ for } i = 1, \ldots, n,$$

Why consider dual problem?

- The dual problem is **always convex**, regardless of the convexity of the primal.

Weak and strong duality

Weak duality: the solution of the primal is always bigger or equal to that of the dual (why?)

$$\max_{\alpha,\beta} \ \min_{\theta} \ \mathcal{L}(\theta, \alpha, \beta) \leq \min_{\theta} \ \max_{\alpha,\beta} \ \mathcal{L}(\theta, \alpha, \beta)$$

Strong duality: when the equality holds.

$$\max_{\alpha,\beta} \ \min_{\theta} \ \mathcal{L}(\theta, \alpha, \beta) = \min_{\theta} \ \max_{\alpha,\beta} \ \mathcal{L}(\theta, \alpha, \beta)$$

Karush-Kuhn-Tuchker (KKT) condition

When the strong duality holds, the following conditions are satisfied.

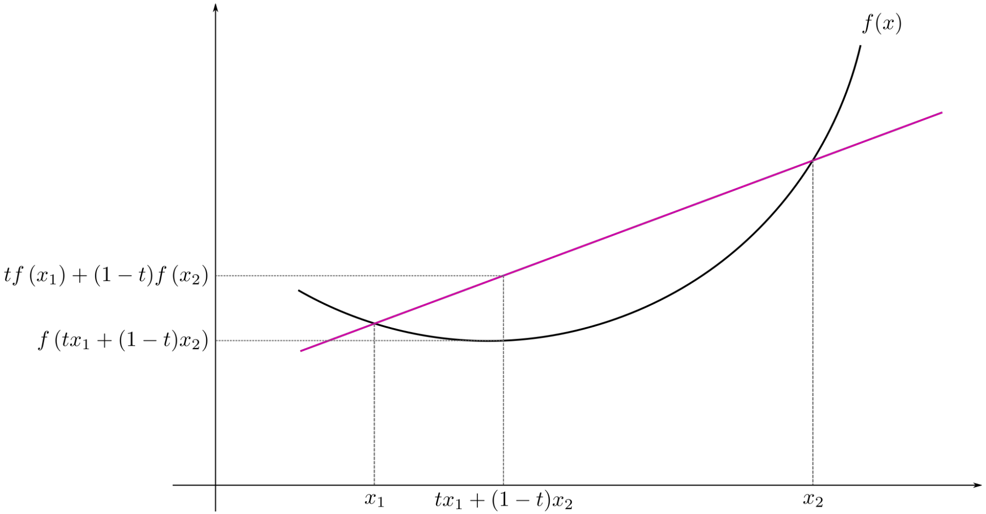
- Primal feasibility: $g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, n \quad h_j(\theta) = 0 \text{ for } j = 1, \ldots, m$
- Dual feasibility: $\alpha_i \geq 0 \text{ for } i = 1, \ldots, n.$
- Complementary slackness: $\alpha_i g_i(\theta) = 0 \text{ for } i = 1, \ldots, n.$
- Stationarity: $\nabla_\theta \mathcal{L}(\theta, \alpha, \beta) = 0.$

When does the strong duality holds?

Slator's condition

1. f **is convex.**
2. g_i **is affine for** $i = 1, \ldots, n.$
3. h_j **Is affine for** $j = 1, \ldots, m.$
4. **There exists** θ **such that**

$$\text{s. t. } g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, n$$
$$h_j(\theta) = 0 \text{ for } j = 1, \ldots, m$$



$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$$

Solving the dual of SVM problem

SVM objective

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

$$\text{s. t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, n$$

convex

$$f(\mathbf{w}, \mathbf{b}) = \frac{1}{2}\|\mathbf{w}\|^2.$$

affine

$$g_i(\mathbf{w}, b) = 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$$

Strong duality holds!

Dual of the SVM objective

Lagrangian:

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n} \alpha_i (1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b))$$

The dual problem is given as (prove it by yourself!)

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j x_i^\top x_j$$

$$\text{s.t. } \alpha_i \geq 0 \text{ for } i = 1, \ldots, n, \quad \sum_{i=1}^{n} \alpha_i y_i = 0.$$

Support vectors?

The optimal solution looks like

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

By the KKT condition,

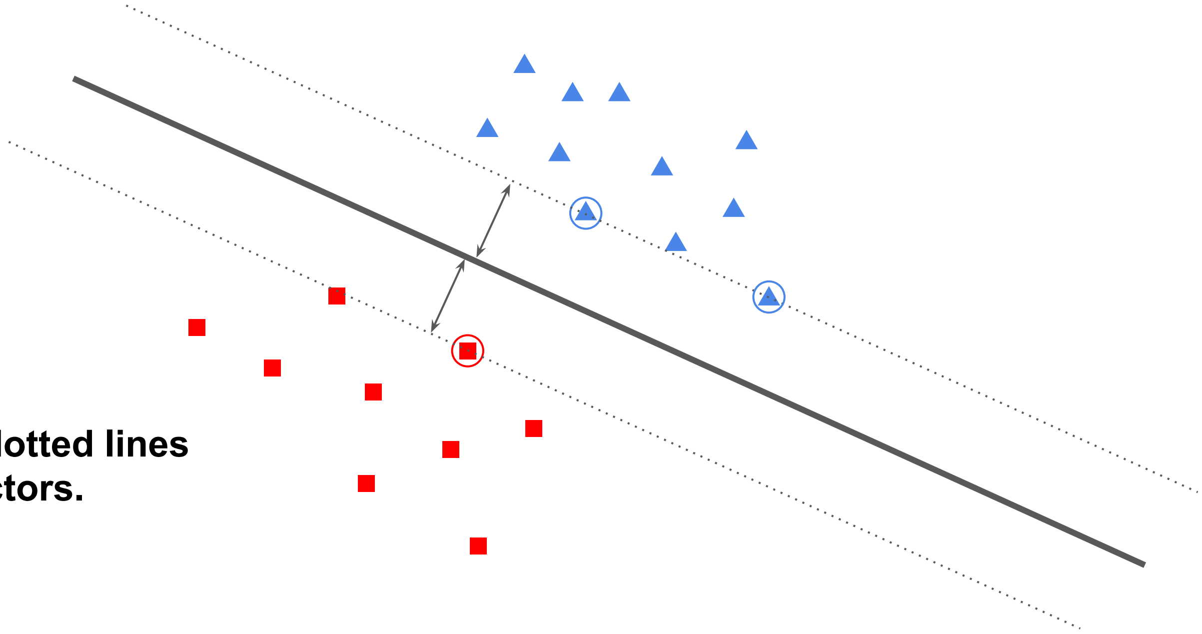
$$\alpha_i^\star \geq 0, \;\; \alpha_i^\star (1 - y_i((\mathbf{w}^\star)^\top \mathbf{x}_i + b)) = 0 \text{ for } i = 1, \ldots, n.$$
$$\alpha_i^\star > 0 \Rightarrow y_i((\mathbf{w}^\star)^\top \mathbf{x}_i + b) = 1.$$

I.e., only the points having smallest distance to the plane affect the decision.

Support vectors?

$\alpha_i^\star > 0.$

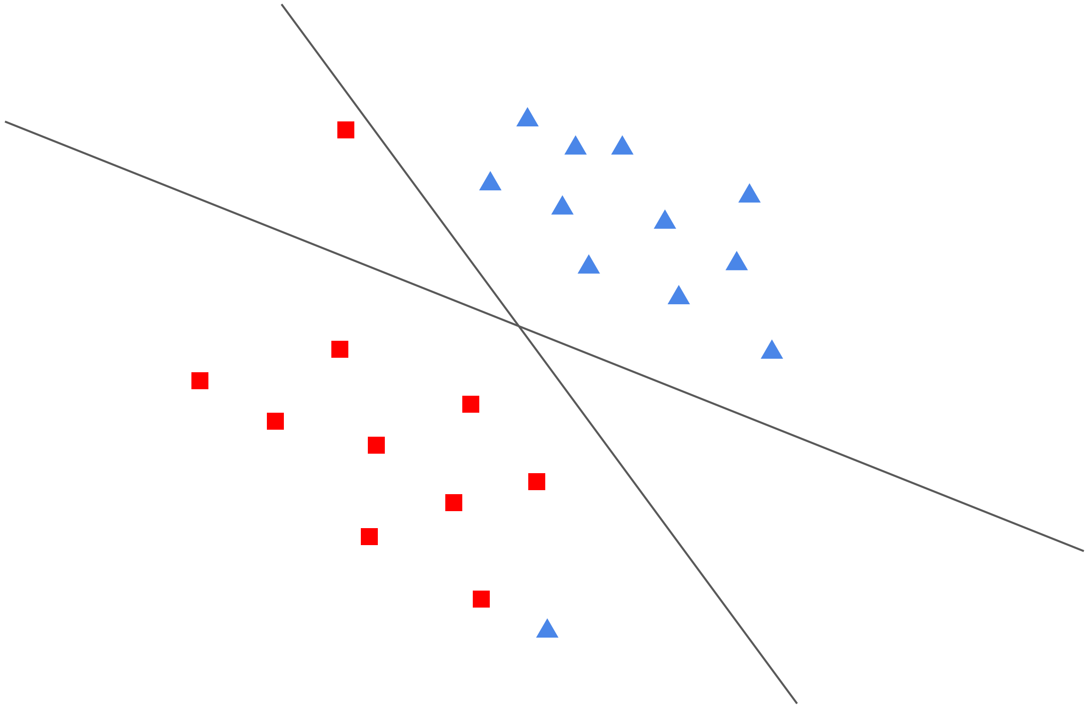
$y_i((\mathbf{w}^\star)^\top \mathbf{x}_i + b) = 1.$



The points on dotted lines are support vectors.

Soft-margin SVM for non-separable case

Many problems are actually non-separable



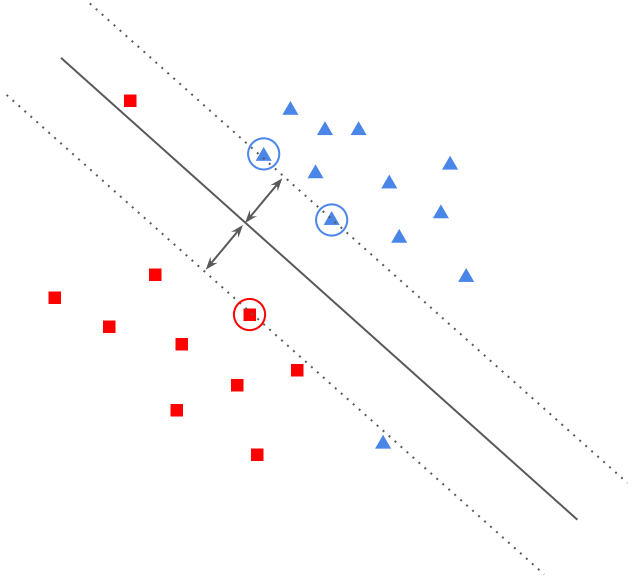
Cut me some slack

Introduce slack variables as

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

$$\text{s. t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0 \text{ for } i = 1, \ldots, n.$$



The dual of the soft-margin SVM

Prove it by yourself!

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^\top \mathbf{x}_j$$

$$\text{s. t. } 0 \leq \alpha_i \leq C \text{ for } i = 1, \ldots, n, \ \sum_{i=1}^{n} \alpha_i y_i = 0.$$

KKT condition for soft-margin SVM

Complementary slackness for optimal solutions $(\mathbf{w}^\star, b^\star, \xi^\star, \alpha^\star, \gamma^\star)$

$$\alpha_i^\star(1 - \xi_i - y_i((\mathbf{w}^\star)^\top \mathbf{x}_i + b)) = 0,$$
$$\gamma_i^\star \xi_i = 0 \text{ for } i = 1, \ldots, n.$$

Support vectors

$$\alpha_i^\star > 0 \implies y_i((\mathbf{w}^\star)^\top \mathbf{x}_i + b) = 1 - \xi_i^\star.$$

$$\Downarrow$$

$$\gamma_i > 0 \implies y_i((\mathbf{w}^\star)^\top \mathbf{x}_i + b) = 1.$$

Kernel trick

Feature mapping

We often want to work with a nonlinear feature mapping;

$$\phi(\mathbf{x}) = [x_1, x_1 x_2, x_1 x_2 x_3, \ldots,]^\top$$

$$\phi(\mathbf{x}) = [\sin x_1, \cos x_2, \sin x_3 \cos x_4, \ldots,]^\top$$

A SVM classifier with a feature map is

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \phi(\mathbf{x}) + b)$$

But what feature maps?

- Designing proper feature map for a given problem is often hard (this is before the deep learning era; we can learn those feature maps from data nowadays).
- Feature maps required to well-separate the data are often very high-dimensional, or even infinite dimensional, so hard to work with.
- Observation: directly working with the feature maps are hard, but working with the **inner-products** of them is more easy.

$$K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \phi(\mathbf{y}).$$

Kernels

- The function $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \phi(\mathbf{y})$ is called **kernel.**
- No discussion about its mathematical foundation today.
- Popular kernels

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right).$$

Gaussian kernel (a.k.a. RBF kernel)

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top \mathbf{y} + b)^d.$$

Polynomial kernel

SVM with nonlinear feature maps

The primal SVM objective: have to know the form of the feature map.

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

$$\text{s. t. } y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b) \geq 1 \text{ for } i = 1, \ldots, n$$

The dual objective: doesn't require the feature map, only the kernel.

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{s. t. } \alpha_i \geq 0 \text{ for } i = 1, \ldots, n, \quad \sum_{i=1}^{n} \alpha_i y_i = 0.$$

Kernel trick

- The optimization of problems requiring nonlinear feature maps can be made much easier by looking at the dual, because in many cases they only require the kernel.
- This also applies to many other machine learning techniques, e.g., kernel PCA, kernel LDA, ….

Extras

Solving the dual objective

- Quadratic programming solvers + some special care
- Sequential minimal optimization (SMO) algorithm
- If you are interested in implementing it by yourself, checkout
  - http://cs229.stanford.edu/notes/cs229-notes3.pdf
  - https://leon.bottou.org/publications/pdf/lin-2006.pdf
  - http://emilemathieu.fr/blog_svm.html

Multiclass SVM

- One vs. all classifier
    - Train one class vs the rest classes classifier for all classes.
    - Do classification.
    - Choose the class with highest confidence ($y_i(\mathbf{w}^\top \mathbf{x}_i + b)$).
- All pairs (one vs. one)
    - Train one vs one classifier for every pair of classes.
    - Do classification.
    - Pick the class who have won most.
- Advanced techniques
    - DAGSVM
    - Error-correcting output codes

Practical considerations

- The performance of SVM is sensitive to the hyperparameters.
  - The parameter σ^2 in RBF kernel.
  - The parameter C.
- The classes are often imbalanced - weighted SVM.

Coding practice