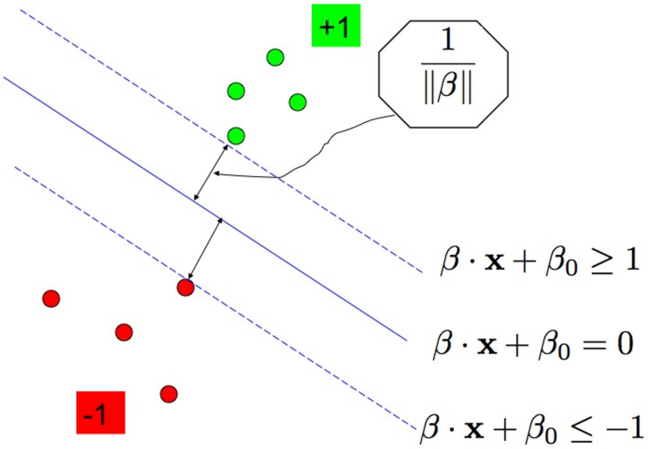
Linear SVM (separable case)

First consider the scenario where the two classes of points are separable.


+1
$\frac{1}{\|\beta\|}$
$\beta \cdot \mathbf{x} + \beta_0 \geq 1$
$\beta \cdot \mathbf{x} + \beta_0 = 0$
$\beta \cdot \mathbf{x} + \beta_0 \leq -1$
-1

It's desirable to have the width (called margin) between the two dashed lines to be large, i.e., have the buffer-zone between the two classes as large as possible. We can formulate this max-margin problem as follows:

$$\begin{aligned} \min_{\boldsymbol{\beta}, \beta_0} \quad & \frac{1}{2}\|\boldsymbol{\beta}\|^2 \\ \text{subject to} \quad & y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + \beta_0) - 1 \geq 0, \end{aligned} \tag{1}$$

where $\boldsymbol{\beta} \cdot \mathbf{x}_i = \boldsymbol{\beta}^t \mathbf{x}_i$ denotes the (Euclidian) inner product between two vectors. The constraints are imposed to make sure that the points are on the correct side of the dashed lines, i.e.,

$$\begin{aligned} \boldsymbol{\beta} \cdot \mathbf{x}_i + \beta_0 \geq +1 \quad & \text{for } y_i = +1, \\ \boldsymbol{\beta} \cdot \mathbf{x}_i + \beta_0 \leq -1 \quad & \text{for } y_i = -1. \end{aligned}$$

If $\mathbf{x}_i$ is on one of the dashed lines, i.e., $y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + \beta_0) - 1 = 0$, then we say that this i-th constraint (or i-th point) is *active*. If $y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + \beta_0) - 1 > 0$, we say it's *inactive*. Apparently, there should be at least two points, one from each class, which are active (otherwise we can improve the margin).

This type of constrained optimization (1), known as the *convex programming problem*, has been well-studied in the literature. We actually solve its twin-sister, the so-called dual problem,

$$\begin{aligned} \max_{\lambda_1,\ldots,\lambda_n} \quad & \sum \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ \text{subject to} \quad & \sum \lambda_i y_i = 0, \;\; \lambda_i \geq 0, \end{aligned} \tag{2}$$

where the variables are $\lambda_1, \ldots, \lambda_n$ and λ_i is associated with the ith observation.

Here are the connections between these two optimization problems, where $(\hat{\beta}_0, \hat{\boldsymbol{\beta}})$ denote the solution for the *primal problem* (1), and $(\lambda_1, \ldots, \lambda_n)$ the solution for the *dual problem* (2).

- Complementarity condition:

  $$\lambda_i \{ y_i (\mathbf{x}_i \cdot \hat{\boldsymbol{\beta}} + \hat{\beta}_0) - 1 \} = 0.$$

  That is, if a point is inactive (i.e, not on the dashed line), the corresponding λ_i must equal 0. Those points for which $\lambda_i > 0$ are called "support vectors".

- We can obtain an estimate of the slope $\boldsymbol{\beta}$ after solving the dual problem based on the following equality

  $$\hat{\boldsymbol{\beta}} = \sum \lambda_i y_i \mathbf{x}_i = \sum_{i \in N_s} \lambda_i y_i \mathbf{x}_i,$$

  where N_s denotes the set of support vectors. So .

- Pick any support vector, due to the complementarity condition, it must satisfy the equality $y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + \beta_0) - 1 = 0$, so we can solve for $\hat{\beta}_0$. Of course it is numerically safer to obtain an estimate of β_0 from each support vector, and then take the mean of all such values.

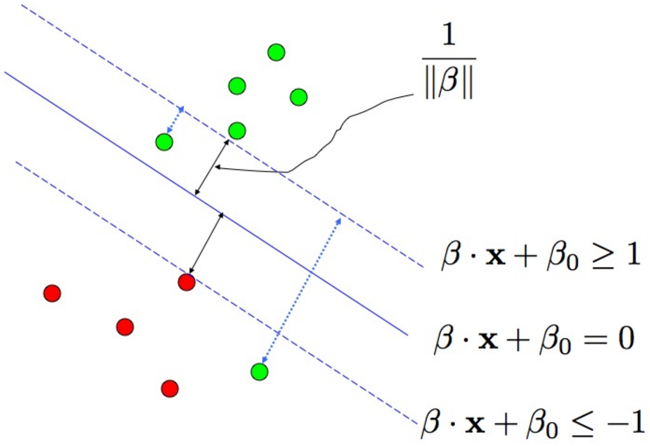
- For any new observation $\mathbf{x}_*$, the prediction is

  $$\text{sign}(\sum_{i \in N_s} \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x}_* + \hat{\beta}_0).$$

  Note that the classifier only depends on the value, $(y_i, \mathbf{x}_i)$, of the support vectors, so it's a *sparse* solution. Also, the classifier is *robust* in the sense that if we move an inactive point around in the correct region (i.e., don't cross the dashed line), then the estimated classifier stays the same since the inactive points, as long as they stay inactive (therefore their $\lambda_i = 0$), do not affect the classifier.

Linear SVM (non-separable case)

What if the two classes of points are not separable?


$\frac{1}{\|\beta\|}$
$\beta \cdot \mathbf{x} + \beta_0 \geq 1$
$\beta \cdot \mathbf{x} + \beta_0 = 0$
$\beta \cdot \mathbf{x} + \beta_0 \leq -1$

Then we introduce a slack variable ξ_i for each sample, and formulate the max-margin problem as follows

$$\begin{aligned} \min_{\boldsymbol{\beta},\beta_0} \quad & \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \gamma \sum \xi_i \\ \text{subject to} \quad & y_i(\mathbf{x}_i \cdot \boldsymbol{\beta} + \beta_0) - 1 + \xi_i \geq 0, \\ & \xi_i \geq 0. \end{aligned} \tag{3}$$

Note that $\xi_i > 0$ only for samples that are on the wrong side of the dashed line, and ξ_i is automatically (by the optimization) set to be 0 for samples that are on the correct side of the dashed line.

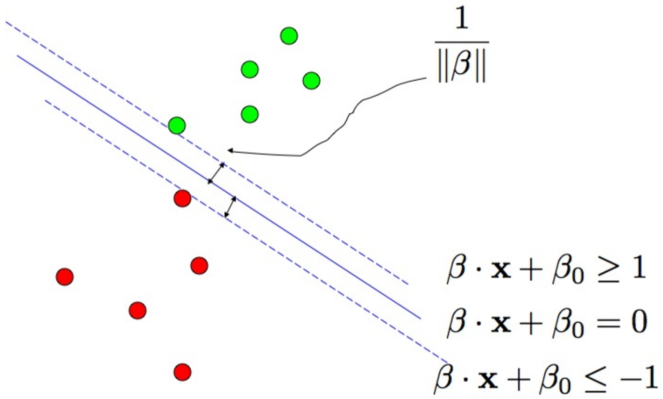
The optimization (3) reflects the trade-off between the margin $2/\|\boldsymbol{\beta}\|^2$ (your gain) and the sum of the positive slack variables (the price you need to pay), and γ is a tuning parameter which can be treated as given at this moment and is often selected by cross-validation in practice.

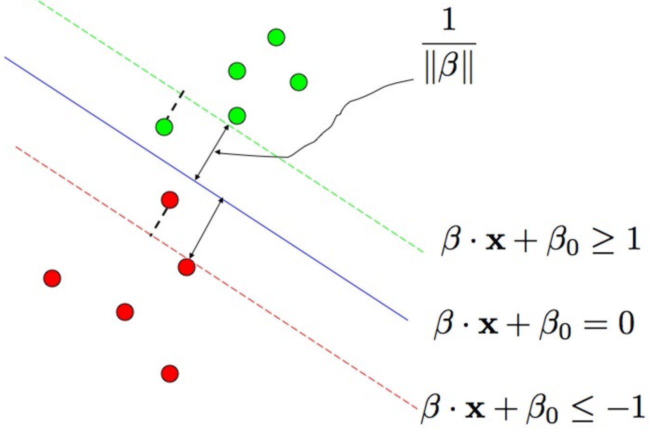
Similarly to the separable case, we solve the dual problem

$$\begin{aligned} \max_{\lambda_1,\ldots,\lambda_n} \quad & \sum \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ \text{subject to} \quad & \sum \lambda_i y_i = 0, \ 0 \leq \lambda_i \leq \gamma. \end{aligned}$$

The connections between the primal and the dual problems are similar to the ones for the separable case. Especially, the inactive points (i.e., the ones on the correct side of the dashed line) have their $\lambda_i = 0$. So after solving for λ_i's (only a hand of them will be non-zero), we can obtain a sparse classifier, which only depends on a subset of the data points (i.e., the support vectors).

Although this new optimization (3) is proposed for non-separable cases, we can use it on separable cases too: depending on the magnitude of γ, we might be willing to pay some price to have points cross the dashed line, if it will lead to a big gain of the margin.


$\frac{1}{\|\beta\|}$
$\beta \cdot \mathbf{x} + \beta_0 \geq 1$
$\beta \cdot \mathbf{x} + \beta_0 = 0$
$\beta \cdot \mathbf{x} + \beta_0 \leq -1$


$\frac{1}{\|\beta\|}$
$\beta \cdot \mathbf{x} + \beta_0 \geq 1$
$\beta \cdot \mathbf{x} + \beta_0 = 0$
$\beta \cdot \mathbf{x} + \beta_0 \leq -1$

Non-linear SVM

In linear SVMs, the decision boundaries are linear (i.e., hyper-planes in the feature space). To obtain more flexible classifiers that have nonlinear decision boundaries, we can consider this approach: first embed data points into a large feature space

$$\Phi : \mathcal{X} \to \mathcal{F}, \quad \Phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots), \tag{4}$$

then operate the linear SVM in the feature space $\mathcal{F}$; since a linear function of $\Phi(\mathbf{x})$ may be a non-linear function of $\mathbf{x}$, we can then obtain nonlinear classifiers.

Kernel trick. When operating the linear SVM in the feature space, the only quantity we need to evaluate is the inner product

$$K_\Phi(\mathbf{x}_i, \mathbf{x}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}_i) \rangle. \tag{5}$$

Note that I did not write the inner product as $\Phi(\mathbf{x})^t \Phi(\mathbf{x}_i)$ since the feature space $\mathcal{F}$ might be infinite dimensional (not a finite dimensional Euclidean space). For example, the mapped feature $\Phi(\mathbf{x})$ could be a function, and then $\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}_i) \rangle$ is just whatever inner product between two functions from the feature space $\mathcal{F}$ (apparently, we have assumed that $\mathcal{F}$ is a Hilbert space).

- Recall the linear SVM solves

$$\begin{aligned} \max_{\lambda_i} \quad & \sum \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ \text{subject to} \quad & \sum \lambda_i y_i = 0, \ 0 \le \lambda_i \le \gamma. \end{aligned}$$

  for $\lambda_1, \dots, \lambda_n$, and then the prediction at a new point $\mathbf{x}^*$ is given by

$$\begin{aligned} & \text{sign}(f(\mathbf{x}^*)) \\ = \ & \text{sign}\Big(\sum_{i \in N_s} \lambda_i y_i (\mathbf{x}_i \cdot \mathbf{x}^*) + \hat{\beta}_0 \Big) \end{aligned}$$

- Now the non-linear SVM solves

$$\begin{aligned} \max_{\lambda_i} \quad & \sum \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \sum \lambda_i y_i = 0, \ 0 \le \lambda_i \le \gamma. \end{aligned}$$

for $\lambda_1, \ldots, \lambda_n$, and then the prediction at a new point $\mathbf{x}^*$ is given by

$$\begin{aligned} & \text{sign}(f(\mathbf{x}^*)) \\ = & \text{sign}\Big(\sum_{i \in N_s} \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}^*) + \hat{\beta}_0\Big) \end{aligned}$$

The bivariate function K in (5) is often referred to as the *reproducing kernel* (r.k.) function or simply the *kernel* function[1]. We can view $K(\mathbf{x}, \mathbf{z})$ as a similarity measure between $\mathbf{x}$ and $\mathbf{z}$, which generalizes the ordinary Euclidean inner product between $\mathbf{x}$ and $\mathbf{z}$. Popular kernels (see p434 of textbook) include

- dth degree polynomial

$$K(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x} \cdot \mathbf{z})^d,$$

- Radial basis (the feature space is of infinite-dimension)

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\|\mathbf{x} - \mathbf{z}\|^2 / c).$$

SVM as a penalization method

Let $f(x) = \mathbf{x} \cdot \boldsymbol{\beta} + \beta_0$ and $y_i \in \{-1, 1\}$. Then

$$\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^n [1 - y_i f(\mathbf{x}_i)]_+ + \nu \|\boldsymbol{\beta}\|^2 \tag{6}$$

has the same solution as the linear SVM (3), when the tuning parameter ν is properly chose (which will depend on γ in (3)). So SVM is a special case of the following Loss + Penalty framework

$$\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^n L(y_i, f(\mathbf{x}_i)) + \nu \|\boldsymbol{\beta}\|^2.$$

The loss used in SVM is called the *hinge loss*,

$$L(y, f(\mathbf{x})) = [1 - y f(\mathbf{x})]_+.$$

[1] It can be shown that a kernel function $K(\mathbf{x}, \mathbf{z})$ must be symmetric and semi-positive definite.

Other popular loss functions for classification problems are (negative) Log-likelihood loss,

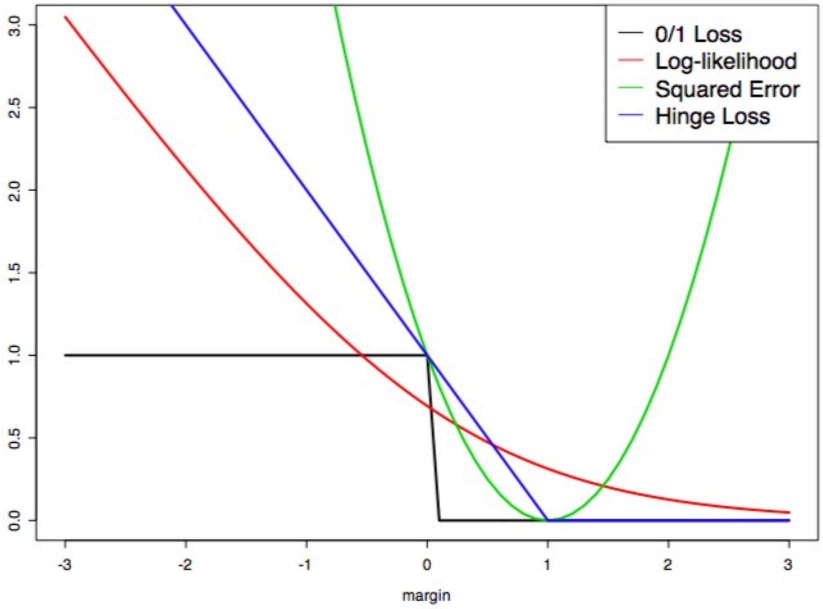
$$L(y, f(\mathbf{x})) = \log\left(1 + e^{-yf(\mathbf{x})}\right),$$

the squared error loss,

$$L(y, f(\mathbf{x})) = (y - f(x))^2 = (1 - yf(\mathbf{x}))^2,$$

and the 0/1 loss

$$L(y, f(\mathbf{x}) = \mathbf{1}\{yf(\mathbf{x}) \geq 0\}.$$



A nonlinear SVM (associated with a kernel function K) solves

$$\min_f \sum_{i=1}^{n} [1 - y_i f(\mathbf{x}_i)]_+ + \nu \|f\|^2_{\mathcal{H}_K}, \tag{7}$$

where f (usually nonlinear) belongs to a Hilbert space ${\mathcal{H}_K}^2$, which is determined by the kernel function K, and $\|f\|^2_{\mathcal{H}_K}$ denotes the corresponding norm. Although the function space $\mathcal{H}_K$ is very large (possibly infinite dimensional), a beautiful result, known as the *representer theorem*, shows that

[2] Known as the reproducing kernel Hilbert space (RKHS)

the minimizer of (7) is always finite dimensional (with maximal dim $= n$) and takes the following form

$$\begin{aligned} & \text{argmin}_{f\in\mathcal{H}_K}\Big[\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \nu\|f\|^2_{\mathcal{H}_K}\Big] \\ = \quad & \alpha_1 K(\mathbf{x}, \mathbf{x}_1) + \cdots + \alpha_n K(\mathbf{x}, \mathbf{x}_n). \end{aligned}$$

Note that this is true for any loss function, so we replace the hinge loss in (7) by $L(y, f(\mathbf{x}))$.

For example, suppose we use squared error loss, the optimization (7) becomes

$$\|\mathbf{y} - \mathbf{K}_{n\times n}\boldsymbol{\alpha}\|^2 + \nu\boldsymbol{\alpha}\mathbf{K}_{n\times n}\boldsymbol{\alpha},$$

where $\mathbf{K}_{n\times n}$ is a $n \times n$ matrix with the (i, j)th entry equal to $K(\mathbf{x}_i, \mathbf{x}_j)$, and we have used the result that

$$\|\alpha_1 K(\cdot, \mathbf{x}_1) + \cdots + \alpha_n K(\cdot, \mathbf{x}_n)\|^2_{\mathcal{H}_K} = \boldsymbol{\alpha}^t\mathbf{K}\boldsymbol{\alpha}.$$

In other words, what a nonlinear SVM does is to first create a set of n new predictors $K(\cdot, \mathbf{x}_i)$, each of which measures the similarity to the i-th sample, and then solve for a linear function of these n predictors in the loss + penalty framework where the penalty is a ridge-type L_2 penalty.

> *Wait*, we have learned that ridge penalty won't lead to a sparse solution, but why the solution from SVM is sparse?
>
> Yes, not like the L_1 penalty, the L_2 penalty on $\boldsymbol{\alpha}$ won't lead to a sparse solution. The penalty in SVM isn't L_1, but the hinge loss function is like L_1, and it is the hinge loss that leads to the sparse solution of $\boldsymbol{\alpha}$.