Lecture Notes on Support Vector Machine

Feng Li
fli@sdu.edu.cn
Shandong University, China

1 Hyperplane and Margin

In a n-dimensional space, a hyper plane is defined by

$$\omega^T x + b = 0 \tag{1}$$

where $\omega \in \mathbb{R}^n$ is an outward pointing normal vector, and b is a bias term. The n-dimensional space is separated into two half-spaces $H^+ = \{x \in \mathbb{R}^n \mid \omega^T x + b \geq 0\}$ and $H^- = \{x \in \mathbb{R}^n \mid \omega^T x + b < 0\}$ by the hyperplane, such that we can identify to which half-space a given point $x \in \mathbb{R}^n$ belongs according to $\text{sign}(\omega^T x + b)$

Now, given a set of training data $\{(x^{(i)}, y^{(i)})\}_{i=1,\cdots,m}$, we assume that $y^{(i)} = 1$ if $\omega^T x^{(i)} + b \geq 0$, while $y^{(i)} = 0$ otherwise. We define the *margin* between each of them and the hyperplane. In fact, for $(x^{(i)}, y^{(i)})$, margin $\gamma^{(i)}$ is the *signed* distance between $x^{(i)}$ and the hyperplane $\omega^T x + b = 0$, i.e.

$$w^T \left(x^{(i)} - \gamma^{(i)} \frac{w}{\|w\|} \right) + b = 0$$

and we thus have

$$\gamma^{(i)} = \left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} \tag{2}$$

as shown in Fig. 1. In particular, if $y^{(i)} = 1$, $\gamma^{(i)} \geq 0$; otherwise, $\gamma^{(i)} < 0$. We then remove the sign and introduce the concept of *geometric margin* that is unsigned as follows:

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} \right) \tag{3}$$

With respect to the whole training set, the margin is written as

$$\gamma = \min_i \gamma^{(i)} \tag{4}$$

2 Support Vector Machine

2.1 Formulation

The hyperplane actually serves as a decision boundary to differentiating positive labels from negative labels, and we make more confident decision if larger margin

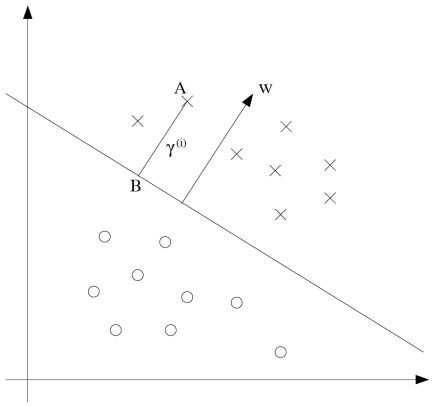


Figure 1: Margin and hyperplane.

is given. By leveraging different values of ω and b, we can construct a infinite number of hyperplanes, but which one is the best? The goal of *Supported Vector Machine* (SVM) is to maximize γ, and the SVM problem can be formulated as

$$\begin{aligned} &\max_{\gamma,\omega,b} \quad \gamma \\ &s.t. \quad y^{(i)}(\omega^T x^{(i)} + b) \geq \gamma \|\omega\|, \quad \forall i \end{aligned}$$

Note that, scaling ω and b (e.g., multiplying both ω and b by a constant) **does** not change the hyperplane, we can scaling (w, b) such that

$$\min_i \{y^{(i)}(\omega^T x^{(i)} + b)\} = 1,$$

such that the representation of the margin becomes $1/\|\omega\|$ according to Eq. (4). Then, the problem formulation can be rewritten as

$$\begin{aligned} &\max_{\omega,b} \quad 1/\|w\| \\ &s.t. \quad y^{(i)}(\omega^T x^{(i)} + b) \geq 1, \quad \forall i \end{aligned}$$

Since maximizing $1/\|\omega\|$ is equivalent to minimizing $\|\omega\|^2 = \omega^T\omega$, we further rewrite the problem formulation as follows

$$\min_{\omega,b} \quad \omega^T\omega \tag{5}$$

$$s.t. \quad y^{(i)}(\omega^T x^{(i)} + b) \geq 1, \quad \forall i \tag{6}$$

This is a *quadratic programming* (QP) problem, and can be solved by exiting generic QP solvers, e.g., interior point method, active set method, gradient projection method. Unfortunately, the existing generic QP solvers is of low efficiency, especially in face of a large training set.

2.2 Preliminary Knowledge of Convex Optimization

Before diving into optimizing the problem formulation of SVM to improve the efficiency of solving QP programming problem, we first introduce some preliminaries about convex optimization.

2.2.1 Optimization Problems and Lagrangian Duality

We now consider the following optimization problem

$$\min_{\omega} \quad f(\omega) \tag{7}$$
$$s.t. \quad g_i(\omega) \leq 0, i = 1, \cdots, k \tag{8}$$
$$h_j(\omega) = 0, j = 1, \cdots, l \tag{9}$$

where $\omega \in \mathcal{D}$ is variable with $\mathcal{D} \subseteq \mathbb{R}^n$ denotes the feasible domain defined by the constraints. $f(\omega)$ is so-called objective function. In the above optimization problem, we have k inequality constraints $g_i(\omega)$ and l equality constraints $h_j(\omega)$.

We construct the *Lagrangian* of the above optimization problem, i.e., $L : \mathbb{R}^n \times \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$ with $\mathbf{dom}L = \mathcal{D} \times \mathbb{R}^k \times \mathbb{R}^l$,

$$\mathcal{L}(\omega, \alpha, \beta) = f(\omega) + \sum_{i=1}^{k} \alpha_i g_i(\omega) + \sum_{j=1}^{l} \beta_j h_j(\omega) \tag{10}$$

In fact, $\mathcal{L}(\omega, \alpha, \beta)$ can be treated as a weighted sum of objective and constraint functions. α_i is *Lagrange multiplier* associated with $g_i(\omega) \leq 0$, while β_i is *Lagrange multiplier* associated with $h_i(\omega) = 0$

We then define its Lagrange dual function $\mathcal{G} : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$ as an infimum [1] of $\mathcal{L}$ with respect to ω, i.e.,

$$\begin{aligned} \mathcal{G}(\alpha, \beta) &= \inf_{\omega \in \mathcal{D}} \mathcal{L}(\omega, \alpha, \beta) \\ &= \inf_{\omega \in \mathcal{D}} \left(f(\omega) + \sum_{i=1}^{k} \alpha_i g_i(\omega) + \sum_{j=1}^{l} \beta_j h_j(\omega) \right) \end{aligned} \tag{11}$$

We observe that, i) the infimum is unconstrained (as supposed to the original constrained minimization problem); ii) $\mathcal{G}$ is an infimum of a set of affine functions and thus is a *concave* function regardless of the original problem; iii) $\mathcal{G}$ can be $-\infty$ for some α and β

Theorem 1. *Lower bounds property: If $\alpha \succeq 0$, then $\mathcal{G}(\alpha, \beta) \leq p^*$ where p^* is the optimal value of the original problem (7)∼(9).*

Proof. If $\widetilde{\omega}$ is feasible, then we have $g_i(\widetilde{\omega}) \leq 0$ for $\forall i = 1, \cdots, k$ and $h_j(\widetilde{\omega}) = 0$ for $\forall j = 1, \cdots, l$. Since $\alpha \succeq 0$, we have $f(\widetilde{\omega}) \geq \mathcal{L}(\widetilde{\omega}, \alpha, \beta)$ for all feasible $\widetilde{\omega}$'s. Because $\mathcal{L}(\tilde{\omega}, \alpha, \beta) \geq \inf_{\omega \in \mathcal{D}} L(\omega, \alpha, \beta)$, we have

$$f(\tilde{\omega}) \geq \mathcal{L}(\tilde{\omega}, \alpha, \beta) \geq \inf_{\omega \in \mathcal{D}} L(\omega, \alpha, \beta) = \mathcal{G}(\alpha, \beta)$$

for all feasible $\tilde{\omega}$. We now choose minimizer of $f(\tilde{\omega})$ over all feasible $\tilde{\omega}$'s to get $p^* \geq \mathcal{G}(\alpha, \beta)$. □

[1] In mathematics, the infimum (abbreviated **inf**; plural **infima**) of a subset S of a partially ordered set T is the greatest element in T that is less than or equal to all elements of S, if such an element exists. More details about infimum and its counterpart **suprema** can be found in `https://en.wikipedia.org/wiki/Infimum_and_supremum`.

We could try to find the best lower bound by maximizing $\mathcal{G}(\alpha, \beta)$. This is in fact the dual problem. We formally define the Lagrange dual problem as follows

$$\begin{array}{ll} \max_{\alpha,\beta} & \mathcal{G}(\alpha, \beta) \\ s.t. & \alpha \succeq 0, \quad \forall i = 1, \cdots, k \end{array}$$

We denote by d^* the optimal value of the above Lagrange dual problem. The *weak* duality $d^* \leq p^*$ always holds for all optimization problems, and can be used to find non-trivial lower bounds for difficult problems. We say the duality is *strong* if the equality holds, i.e., $d^* = p^*$. In this case, we can optimize the original problem by optimizing its dual problem.

2.2.2 Complementary Slackness

Let ω^* be a primal optimal point and (α^*, β^*) be a dual optimal point.

Theorem 2. *Complementary slackness: If strong duality holds, then*

$$\alpha_i^* g_i(\omega^*) = 0$$

for $\forall i = 1, 2, \cdots, k$

Proof.

$$\begin{aligned} f(\omega^*) &= \mathcal{G}(\alpha^*, \beta^*) \\ &= \inf_{\omega} \left(f(\omega) + \sum_{i=1}^{k} \alpha_i^* g_i(\omega) + \sum_{j=1}^{l} \beta_j^* h_j(\omega) \right) \\ &\leq f(\omega^*) + \sum_{i=1}^{k} \alpha_i^* g_i(\omega^*) + \sum_{j=1}^{l} \beta_j^* h_j(\omega^*) \\ &\leq f(\omega^*) \end{aligned}$$

The first line is due to the strong duality, and the second line is the definition of the dual function. The third line follows because the infimum of the Lagrangian over ω is less than or equal to its value at $\omega = \omega^*$. We have the fourth line since $\alpha_i^* \geq 0$ and $g_i(\omega^*) \leq 0$ holds for $\forall i = 1, \cdots, k$, and $h_j(\omega^*) = 0$ for $j = 1, \cdots, l$. We conclude that the last two inequalities hold with equality, such that we have $\sum_{i=1}^{k} \alpha_i^* g_i(\omega^*) = 0$. Since each term, i.e., $\alpha_i^* g_i(\omega^*)$, is nonpositive, we thus conclude $\alpha_i^* g_i(\omega^*) = 0$ for $\forall i = 1, 2, \cdots, k$. □

Another observation is that, since the inequality in the third line holds with equality, ω^* actually minimizes $\mathcal{L}(\omega, \alpha^*, \beta^*)$ over ω.

2.2.3 Karush-Kuhn-Tucker Conditions

We now introduce *Karush-Kuhn-Tucker* (KKT) conditions. We assume that the objective function and the inequality constraint functions are differentiable. Again, let ω^* and (α^*, β^*) be any primal and dual optimal points, and we suppose the strong duality holds. Since ω^* is a minimizer of $\mathcal{L}(\omega, \alpha^*, \beta^*)$ over ω, it follows that its gradient must vanish at ω^*, i.e.,

$$\nabla f(\omega^*) + \sum_{i=1}^{k} \alpha_i^* \nabla g_i(\omega^*) + \sum_{j=1}^{l} \beta_j^* \nabla h_j(\omega^*) = 0 \tag{12}$$

We thus summarize the KKT conditions as follows:

$$g_i(\omega^*) \leq 0,\ \forall i = 1, \cdots, k \tag{13}$$
$$h_j(\omega^*) = 0,\ \forall j = 1, \cdots, l \tag{14}$$
$$\alpha_i^* \geq 0,\ \forall i = 1, \cdots, k \tag{15}$$
$$\alpha_i^* g_i(\omega^*) = 0,\ \forall i = 1, \cdots, k \tag{16}$$
$$\nabla f(\omega^*) + \sum_{i=1}^{k} \alpha_i^* \nabla g_i(\omega^*) + \sum_{j=1}^{l} \beta_j^* \nabla h_j(\omega^*) = 0 \tag{17}$$

Remarks: *For any optimization problem with differentiable objective and constraint functions for which strong duality obtains, any pair of primal and dual optimal points must satisfy the KKT conditions.*

2.2.4 Convex Optimization Problems

An optimization problem is *convex*, if both objective function $f(\omega)$ and inequality constraints $g_i(\omega)$ $(i = 1, \cdots, k)$ are convex, and the equality constraints $h_j(\omega)$ are affine functions, which are denoted by $A\omega - b = 0$ where A is a $l \times n$ matrix. Therefore, a convex optimization problem can be represented by

$$\min_{\omega} \quad f(w) \tag{18}$$
$$s.t. \quad g_i(w) \leq 0, i = 1, \cdots, k \tag{19}$$
$$Aw - b = 0 \tag{20}$$

Although strong duality does not (in general) hold, but we usually (but not always) have strong duality for convex optimization problems. There are many results that establish conditions on the problem, beyond convexity, under which strong duality holds. These conditions are called *constraint qualifications*. One simple constraint qualification is *Slater's condition*.

Theorem 3. *Slater's condition: Strong duality holds for a convex problem*

$$\begin{aligned} \min_{\omega} \quad & f(w) \\ s.t. \quad & g_i(w) \leq 0, i = 1, \cdots, k \\ & Aw - b = 0 \end{aligned}$$

if it is strictly feasible, i.e.,

$$\exists \omega \in \mathbf{int}\mathcal{D} : g_i(\omega) < 0, i = 1, \cdots, m, Aw = b$$

Detailed proof of the above theorem can be found in `http://www.ifp.illinois.edu/~angelia/L8_strongdthms.pdf`

For convex optimization problem, the KKT conditions are also sufficient for the points to be primal and dual optimal. In particular, suppose $\widetilde{\omega}$, $\widetilde{\alpha}$, and $\widetilde{\beta}$

are any points satisfying the following KKT conditions

$$g_i(\widetilde{\omega}) \leq 0, \ \forall i = 1, \cdots, k \tag{21}$$
$$h_j(\widetilde{\omega}) = 0, \ \forall j = 1, \cdots, l \tag{22}$$
$$\widetilde{\alpha}_i \geq 0, \ \forall i = 1, \cdots, k \tag{23}$$
$$\widetilde{\alpha}_i g_i(\widetilde{\omega}) = 0, \ \forall i = 1, \cdots, k \tag{24}$$
$$\nabla f(\widetilde{\omega}) + \sum_{i=1}^{k} \widetilde{\alpha}_i \nabla g_i(\widetilde{\omega}) + \sum_{j=1}^{l} \widetilde{\beta}_j \nabla h_j(\widetilde{\omega}) = 0 \tag{25}$$

then they are primal and dual optimal with strong duality holding.

3 Duality of SVM

We now re-visit our problem formulation of SVM. The (primal) SVM problem is given

$$P: \quad \min_{\omega, b} \quad \frac{1}{2}\|\omega\|^2 \tag{26}$$
$$s.t. \quad y^{(i)}(\omega^T x^{(i)} + b) \geq 1, \quad \forall i \tag{27}$$

where we introduce a constant $1/2$ so as to simplify our later derivations.

Theorem 4. *The dual optimization problem of the primal SVM problem P can be formulated as*

$$D: \quad \max_{\alpha} \quad \mathcal{G}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j (x^{(i)})^T x^{(j)} \tag{28}$$
$$s.t. \quad \alpha_i \geq 0 \quad \forall i \tag{29}$$
$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \tag{30}$$

Proof. We calculate the Lagrange dual function $\mathcal{G}(\alpha)$ by taking the infimum of $\mathcal{L}(\omega, b, \alpha)$ over ω and b.

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y^{(i)}(w^T x^{(i)} + b) - 1) \tag{31}$$

where $\alpha_i \geq 0$ is the Lagrangian multiplier for the i-th inequality constraint.Therefore, we calculate the gradient of $\mathcal{L}(\omega, b, \alpha)$ with respect to ω, and let the gradient be zero

$$\bigtriangledown_{\omega} \mathcal{L}(\omega, b, \alpha) = w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0$$

and we thus have

$$\omega = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \tag{32}$$

Similarly,

$$\frac{\partial}{\partial b}\mathcal{L}(\omega, b, \alpha) = \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \tag{33}$$

In another word, the above two equations are necessary to calculating $\inf_{\omega,b} \mathcal{L}(\omega, b, \alpha)$ over ω and b. Substituting (32) and (33) into (31) gives us

$$\begin{aligned}
& \frac{1}{2}\|\omega\|^2 - \sum_{i=1}^{m} \alpha_i [y^{(i)}(\omega^T x^{(i)} + b) - 1] \\
= \; & \frac{1}{2}\omega^T\omega - \sum_{i=1}^{m} \alpha_i y^{(i)} \omega^T x^{(i)} - \sum_{i=1}^{m} \alpha_i y^{(i)} b + \sum_{i=1}^{m} \alpha_i \\
= \; & \frac{1}{2}\omega^T \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} - \omega^T \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} - \sum_{i=1}^{m} \alpha_i y^{(i)} b + \sum_{i=1}^{m} \alpha_i \\
= \; & -\frac{1}{2}\omega^T \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} - \sum_{i=1}^{m} \alpha_i y^{(i)} b + \sum_{i=1}^{m} \alpha_i \\
= \; & -\frac{1}{2}\left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}\right)^T \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} - b\sum_{i=1}^{m} \alpha_i y^{(i)} + \sum_{i=1}^{m} \alpha_i \\
= \; & -\frac{1}{2}\sum_{i=1}^{m} \alpha_i y^{(i)} (x^{(i)})^T \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} - b\sum_{i=1}^{m} \alpha_i y^{(i)} + \sum_{i=1}^{m} \alpha_i \\
= \; & -\frac{1}{2}\sum_{i=1,j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)} - b\sum_{i=1}^{m} \alpha_i y^{(i)} + \sum_{i=1}^{m} \alpha_i \\
= \; & \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1,j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)} - b\sum_{i=1}^{m} \alpha_i y^{(i)} \\
= \; & \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1,j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)}
\end{aligned}$$

which completes our proof. □

It is a convex optimization problem respecting Slater's condition; therefore, the strong duality ($p^* = d^*$) holds and optimal solutions of ω, α and β satisfy the KKT conditions (as well as complementary slackness). We can use several off-the-shelf solvers (e.g., quadprog (MATLAB), CVXOPT, CPLEX, IPOPT, etc.) exist to solve such a QP problem.

Let α^* be the optimal value of α for the dual SVM problem. We can use Eq. (32) to calculate the optimal value of ω, i.e., ω^*. The question is, being aware of ω^*, how to calculating the optimal value of b, i.e., b^*? Recall that, due to the complementary slackness,

$$\alpha_i^*(y^{(i)}({\omega^*}^T x^{(i)} + b^*) - 1) = 0$$

for $\forall i = 1, \cdots, k$, we have

$$y^{(i)}({\omega^*}^T x^{(i)} + b^*) = 1$$

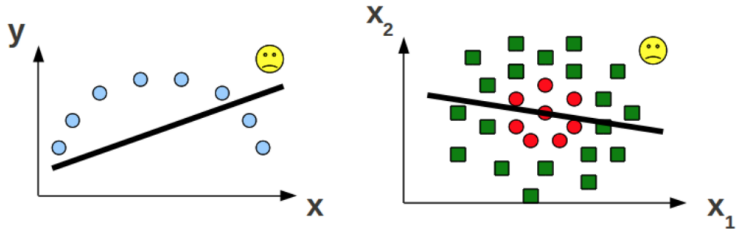


Figure 2: Non-linear data v.s. linear classifier

for $\{i : \alpha_i^* > 0\}$. As $y^{(i)} \in \{-1, 1\}$, we have

$$b^* = y^{(i)} - \omega^{*T} x^{(i)}$$

for $\forall i$ such that $\alpha_i^* > 0$. For robustness, we calculated the optimal value for b by taking the average

$$b^* = \frac{\sum_{i:\alpha_i^*>0}(y^{(i)} - \omega^{*T} x^{(i)})}{\sum_{i=1}^{m} \mathbf{1}(\alpha_i^* > 0)}$$

In fact, most α_i's in the solution are zero (sparse solution). According to completment slackness, for the optimal α_i^*'s,

$$\alpha_i^*[1 - y^{(i)}(\omega^{*T} x^{(i)} + b^*)] = 0$$

α_i^* is non-zero only if $x^{(i)}$ lies on the one of the two margin boundaries. i.e., for which $y^{(i)}(\omega^{*T} x^{(i)} + b) = 1$. These data samples are called **support vector**, i.e., support vectors "support" the margin boundaries, as shown in Fig. **??**. We can redefine ω by

$$w = \sum_{s \in \mathcal{S}} \alpha_s y^{(s)} x^{(s)}$$

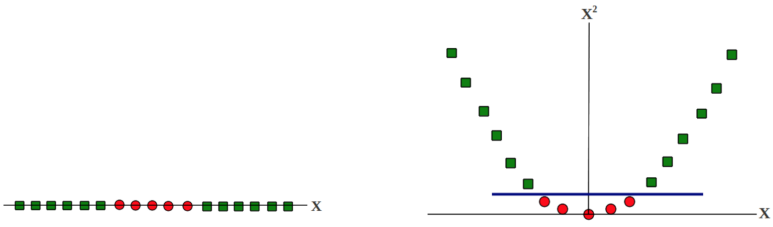
where $\mathcal{S}$ denotes the set of the indices of the support vectors

4 Kernel based SVM

By far, one of our assumption is that the training data can be separated linearly. Nevertheless, Linear models (e.g., linear regression, linear SVM etc.) cannot reflect the nonlinear pattern in the data, as demonstrated in Fig. 4.

The basic idea of kernel method is to make linear model work in nonlinear settings by introducing kernel functions. In particular, by mapping data to higher dimensions where it exhibits linear patterns, we can employ the linear model in the new input space. Mapping is equivalent to changing the feature representation

We take the following binary classification problem for example. As shown in Fig. 3 (a), Each sample is represented by a single feature x, and no linear separator exists for this data. We map each data sample by $x \to \{x, x^2\}$, such that each sample now has two features ("derived" from the old representation).

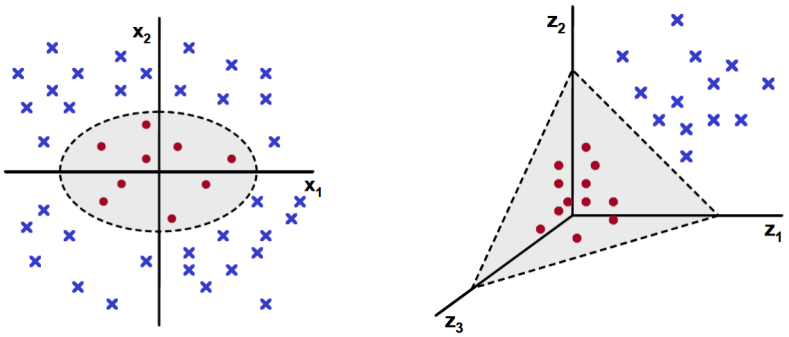


(a) (b)

Figure 3: Feature mapping for 1-dimensional feature space.

As shown in Fig. 3 (b), data become linearly separable in the new representation

Another example is given in Fig. 4. Each sample is defined by $x = \{x_1, x_2\}$, and there is no linear separator exists for this data. We apply the mapping $x = \{x_1, x_2\} \rightarrow z = \{x_1^2, \sqrt{2}x_1x_2, x_2^2\}$, such that the data become linearly separable in the output 3D space.



(a) (b)

Figure 4: Feature mapping for 2-dimensional feature space.

We now consider the follow quadratic feature mapping ϕ for a sample $x = \{x_1, \cdots, x_n\}$

$$\phi : x \rightarrow \{x_1^2, x_2^2, \cdots, x_n^2, x_1x_2, x_1x_2, \cdots, x_1x_n, \cdots, x_{n-1}x_n\}$$

where each new feature uses a pair of the original features. It can be observed that, feature mapping usually leads to the number of features blow up, such that i) computing the mapping itself can be inefficient, especially when the new space is very high dimensional; ii) storing and using these mappings in later computations can be expensive (e.g., we may have to compute inner products in a very high dimensional space); iii) using the mapped representation could be inefficient too. Fortunately, kernels help us avoid both these issues! With the help of kernels, the mapping does not have to be explicitly computed, and computations with the mapped features remains efficient.

Let's assume we are given a function K (kernel) that takes as inputs x and z

$$\begin{aligned} K(x,z) &= (x^T z)^2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 x_2 z_1 z_2 \\ &= (x_1^2, \sqrt{2}x_1 x_2, x_2^2)^T (z_1^2, \sqrt{2}z_1 z_2, z_2^2) \end{aligned}$$

The above function K implicitly defines a mapping ϕ to a higher dimension space

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

Now, simply defining the kernel a certain way gives a higher dimension mapping ϕ. The mapping does not have to be explicitly computed, while computations with the mapped features remain efficient. Moreover, the kernel $K(x,z)$ also computes the dot product $\phi(x)^T \phi(z)$

Formally speaking, each kernel K has an associated feature mapping ϕ, which takes input $x \in \mathcal{X}$ (input space) and maps it to $\mathcal{F}$ (feature space). $\mathcal{F}$ needs to be a vector space with a dot product defined upon it, and is so-called a *Hilbert space*. In another word, kernel $K(x,z) = \phi(x)^T \phi(z)$ takes two inputs and gives their similarity in $\mathcal{F}$ space

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

The problem is, can just any function be used as a kernel function? The answer is no, and kernel function must satisfy *Mercer's Condition*. To introduce Mercer's condition, we need to define the quadratically integrable (or square integrable) function concept. A function $q : \mathbb{R}^n \to \mathbb{R}$ is square integrable if

$$\int_{-\infty}^{\infty} q^2(x)dx < \infty$$

A function $K(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ satisfies Mercer's condition if for any square integrable function $q(x)$, the following inequality is always true:

$$\int\int q(x)K(x,z)q(z)dxdz \geq 0$$

Let K_1 and K_2 be two kernel functions then the followings are as well:

- Direct sum: $K(x,z) = K_1(x,z) + K_2(x,z)$
- Scalar product: $K(x,z) = \alpha K_1(x,z)$
- Direct product: $K(x,z) = K_1(x,z)K_2(x,z)$
- Kernels can also be constructed by composing these rules

In the context of SVM, Mercer's condition translates to another way to check whether K is a valid kernel (i.e., meets Mercer's condition or not). The kernel function K also defines the Kernel Matrix over the data (also denoted by K). Given m samples $\{x^{(1)}, x^{(2)}, \cdots, x^{(m)}\}$, the (i,j)-th entry of K is defined as

$$K_{i,j} = K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)})$$

If the matrix K is positive semi-definite, function $K(\cdot,\cdot)$ is a valid kernel function.

Follows are some commonly used kernels:

- Linear (trivial) Kernal:
$$K(x,z) = x^T z$$
- Quadratic Kernel
$$K(x,z) = (x^T z)^2 \quad or \quad (1+x^T z)^2$$
- Polynomial Kernel (of degree d)
$$K(x,z) = (x^T z)^d \quad or \quad (1+x^T z)^d$$
- Gaussian Kernel
$$K(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$$
- Sigmoid Kernel
$$K(x,z) = \tanh(\alpha x^T + c)$$

Overall, kernel $K(x,z)$ represents a dot product in some high dimensional feature space $\mathcal{F}$
$$K(x,z) = (x^T z)^2 \quad or \quad (1+x^T z)^2$$

Any learning algorithm in which examples only appear as dot products $({x^{(i)}}^T x^{(j)})$ can be kernelized (i.e., non-linearlized), by replacing the ${x^{(i)}}^T x^{(j)}$ terms by $\phi(x^{(i)})^T\phi(x^{(j)}) = K(x^{(i)}, x^{(j)})$. Actually, most learning algorithms are like that, such as SVM, linear regression, etc. Many of the unsupervised learning algorithms too can be kernelized (e.g., K-means clustering, Principal Component Analysis, etc.)

Recall that, the dual problem of SVM can be formulated as

$$\begin{aligned} \max_\alpha \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i,j=1}^m y^{(i)}y^{(j)}\alpha_i\alpha_j < x^{(i)}, x^{(j)} > \\ s.t. \quad & \alpha_i \geq 0 \ \ (\forall i), \quad \sum_{i=1}^m \alpha_i y^{(i)} = 0 \end{aligned}$$

Replacing $< x^{(i)}, x^{(j)} >$ by $\phi(x^{(i)})^T\phi(x^{(j)}) = K(x^{(i)}, x^{(j)}) = K_{ij}$ gives us

$$\begin{aligned} \max_\alpha \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i,j=1}^m y^{(i)}y^{(j)}\alpha_i\alpha_j K_{i,j} \\ s.t. \quad & \alpha_i \geq 0 \ \ (\forall i), \quad \sum_{i=1}^m \alpha_i y^{(i)} = 0 \end{aligned}$$

SVM now learns a linear separator in the kernel defined feature space $\mathcal{F}$, and this corresponds to a non-linear separator in the original space $\mathcal{X}$

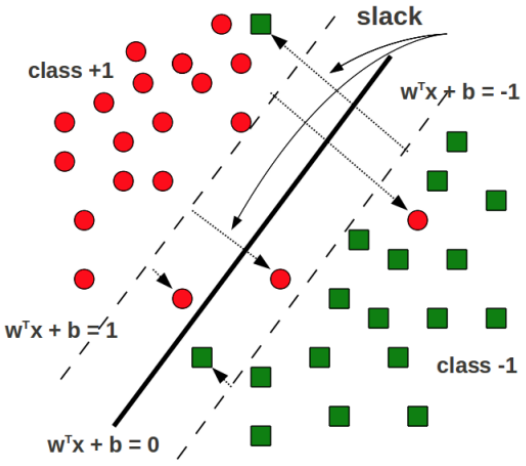


Figure 5: Regularized (Soft-Margin) SVM

Prediction can be made by the SVM without kernel by

$$y = \text{sign}(\omega^T x) = \text{sign}\left(\sum_{s\in\mathcal{S}} \alpha_s y^{(s)} {x^{(s)}}^T x\right)$$

where we assume assume $b = 0$ [2] We replacing each example with its feature mapped representation ($x \to \phi(x)$)

$$y = \text{sign}\left(\sum_{s\in\mathcal{S}} \alpha_s y^{(s)} {x^{(s)}}^T x\right) = \text{sign}\left(\sum_{s\in\mathcal{S}} \alpha_s y^{(s)} K(x^{(s)}, x)\right)$$

Kernelized SVM needs the support vectors at the test time (except when you can write $\phi(x)$ as an explicit, reasonably-sized vector). In the unkernelized version $\omega = \sum_{s\in\mathcal{S}} \alpha_s y^{(s)} x^{(s)}$ can be computed and stored as a $n \times 1$ vector, so the support vectors need not be stored.

5 Regularized SVM

We now introduce regularization to SVM. The regularized SVM is also called *Soft-Margin SVM*. In the regularized SVM, we allow some training examples to be misclassified, such that some training examples to fall within the margin region, as shown in Fig. 5. For the linearly separable case, the constraints are

$$y^{(i)}(\omega^T x^{(i)} + b) \geq 1$$

for $\forall i = 1, \cdots, m$, while in the non-separable case, we relax the above constraints as:

$$y^{(i)}(\omega^T x^{(i)} + b) \geq 1 - \xi_i$$

[2] This can be done by shifting the hyper plane (as well as the test data) such that the plane just passing through the original point (i.e., $b = 0$).

for $\forall i = 1, \cdots, m$, where ξ_i is called *slack variable*.

In the non-separable case, we allow misclassified training examples, but we would like their number to be minimized, by minimizing the sum of the slack variables $\sum_i \xi_i$. We reformulating the SVM problem by introducing slack variables ξ_i

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \tag{34}$$

$$s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, \quad \forall i = 1, \cdots, m \tag{35}$$

$$xi_i \geq 0, \quad \forall i = 1, \cdots, m \tag{36}$$

The parameter C controls the relative weighting between the following two goals: i) although small C implies that $\|\omega\|^2/2$ dominates such that large margins are preferred, this allows a potential large number of misclassified training examples; ii) large C means $C\sum_{i=1}^{m}\xi_i$ dominates $\Rightarrow$ such that the number of misclassified examples is decreased at the expense of having a small margin.

The Lagrangian can be defined by

$$\mathcal{L}(w, b, \xi, \alpha, r) = \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i[y^{(i)}(w^T x^{(i)} + b) - 1 + \xi_i] - \sum_{i=1}^{m} r_i\xi_i$$

and according to KKT conditions, we have

- $\bigtriangledown_w \mathcal{L}(w^*, b^*, \xi^*, \alpha^*, r^*) = 0 \quad \Rightarrow \quad w^* = \sum_{i=1}^{m}\alpha_i^* y^{(i)} x^{(i)}$
- $\bigtriangledown_b \mathcal{L}(w^*, b^*, \xi^*, \alpha^*, r^*) = 0 \quad \Rightarrow \quad \sum_{i=1}^{m}\alpha_i^* y^{(i)} = 0$
- $\bigtriangledown_{\xi_i} \mathcal{L}(w^*, b^*, \xi^*, \alpha^*, r^*) = 0 \quad \Rightarrow \quad \alpha_i^* + r_i^* = C$, for $\forall i$
- $\alpha_i^*, r_i^*, \xi_i^* \geq 0$, for $\forall i$
- $y^{(i)}({w^*}^T x^{(i)} + b^*) + \xi_i^* - 1 \geq 0$, for $\forall i$
- $\alpha_i^*(y^{(i)}({w^*}^T x^{(i)} + b^*) + \xi_i^* - 1) = 0$, for $\forall i$
- $r_i^*\xi_i^* = 0$, for $\forall i$

We then formulated the corresponding dual problem as

$$\begin{aligned}
\max_{\alpha} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j < x^{(i)}, x^{(j)} > \\
s.t. \quad & 0 \leq \alpha_i \leq C, \quad \forall i = 1, \cdots, m \\
& \sum_{i=1}^{m}\alpha_i y^{(i)} = 0
\end{aligned}$$

We can use existing QP solvers to address the above optimization problem

Let α_i^* be the optimal values of α. We now show how to calculate the optimal values of ω and b. According to KKT conditions, ω^* can be calculated by

$$\omega^* = \sum_{i=1}^{m}\alpha_i^* y^{(i)} x^{(i)}$$

Since $\alpha_i^* + r_i^* = C$, for $\forall i$, we have

$$r_i^* = C - \alpha_i^*, \ \forall i$$

Also, considering $r_i^* \xi_i^* = 0$, for $\forall i$, we have

$$(C - \alpha_i^*)\xi_i^* = 0, \ \forall i$$

For $\forall i$ such that $\alpha_i^* \neq C$, we have $\xi_i^* = 0$, and thus

$$\alpha_i^*(y^{(i)}({\omega^*}^T x^{(i)} + b^*) - 1) = 0$$

for those i's. For $\forall i$ such that $0 < \alpha_i^* < C$, we have

$$y^{(i)}({\omega^*}^T x^{(i)} + b^*) = 1$$

Hence,

$${\omega^*}^T x^{(i)} + b^* = y^{(i)}$$

for $\{i : 0 < \alpha_i^* < C\}$. We finally calculate b^* as

$$b = \frac{\sum_{i:0<\alpha_i^*<C}(y^{(i)} - {\omega^*}^T x^{(i)})}{\sum_{i=1}^{m} \mathbf{1}(0 < \alpha_i^* < C)}$$

6 Sequential Minimal Optimization Algorithm