School of Information Science and Technology
ShanghaiTech University

Kernels Methods and Support Vector Machines

Shuhao Xia

May 4, 2020

Agenda

Kernels Methods

- Understanding Kernels
- Constructing Kernels
- Other forms of Kernel Functions

Support Vector Machines (SVM)

- Maximizing Margin Classifiers
- SVM Optimization
- Soft Margin

Intuition

Memory-based method

- Keep and use training samples,
- Measure the similarity of training and test samples,
- Fast to 'train' but slow at making predictions for test samples

E.g., Nearest neighbors.

Kernels Function

Now we need a metric to measure such a similarity. Typically, we use inner product, and *kernels function* is therefore defined as

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}),$$

where $\phi(\cdot)$ is a fixed nonlinear *feature space* mapping.

Kernels Function

Mercer condition

$k(\cdot)$ is a valid kernel if and only if the Gram matrix $\mathbf{K} \triangleq [k(\mathbf{x}_n, \mathbf{x}_m)]$ satisfies the following conditions,

- $\mathbf{K}$ is symmetric,
- $\mathbf{K}$ is postive semi-definite, e.g., $\mathbf{a}^\top \mathbf{K}\mathbf{a}$ $\forall \mathbf{a}$.

Mercer condition is often used to test whether a function is a valid kernel without explicit $\phi(\mathbf{X})$.

Remark

- Conceptually, to apply kernel function, we
  1. find $\phi(\cdot)$ and calculate $\phi(\mathbf{x})$,
  2. measure the similarity by some kernel $k(\mathbf{x}, \mathbf{x}')$.

  However, $\mathbf{x}$ is typically mapped to a high-dimensional space by $\phi(\cdot)$ so that the computational complexity for inner product is huge.
- In practice, we can therefore work directly in terms of kernels, which allows us implicitly to use feature spaces of high, even infinite dimensionality.

Techniques for constructing

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$, $k_2(\mathbf{x}, \mathbf{x}')$, the following new kernels will also valid:

$$k(\mathbf{x}, \mathbf{x}') = ck_1(\mathbf{x}, \mathbf{x}') \tag{1}$$
$$k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}') \tag{2}$$
$$k(\mathbf{x}, \mathbf{x}') = q(k_1(\mathbf{x}, \mathbf{x}')) \tag{3}$$
$$k(\mathbf{x}, \mathbf{x}') = \exp(k_1(\mathbf{x}, \mathbf{x}')) \tag{4}$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \tag{5}$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}') \tag{6}$$
$$k(\mathbf{x}, \mathbf{x}') = k_3(\phi(\mathbf{x}), \phi(\mathbf{x}')) \tag{7}$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}' \tag{8}$$

where $c > 0$ is a constant, $f(\cdot)$ is any function, $q(\cdot)$ is a polynomial with nonnegative coefficients, $k_3(\cdot)$ is a valid kernel and $\mathbf{A}$ is a symmetric positive semidefinite matrix.

Example: Gaussian Kernel

Gaussian Kernel is defined as follows,

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\|\mathbf{x} - \mathbf{x}'\|^2 / 2\sigma^2\right).$$

Proof.

Expanding the square term,

$$\|\mathbf{x} - \mathbf{x}'\|^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x} + (\mathbf{x}')^{\mathrm{T}}\mathbf{x}' - 2\mathbf{x}^{\mathrm{T}}\mathbf{x}',$$

to give

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\mathbf{x}^{\mathrm{T}}\mathbf{x}/2\sigma^2\right)\exp\left(\mathbf{x}^{\mathrm{T}}\mathbf{x}'/\sigma^2\right)\exp\left(-(\mathbf{x}')^{\mathrm{T}}\mathbf{x}'/2\sigma^2\right),$$

and then making use of (2) and (4), together with the validity of the linear kernel $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^{\mathrm{T}}\mathbf{x}'$. □

Other Forms

- *stationary* kernels $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$, invariant to translations in input space,
- *homogeneous* kernels $k(\mathbf{x}, \mathbf{x}') = k(\|\mathbf{x} - \mathbf{x}'\|)$, depend only on the magnitude of the distance.

Intuition

Consider two-class classification problem using linear model of the form

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b. \tag{9}$$

If the training data set is linearly separable,

- ≥ 1 *hyperplanes* to correctly classify all the training samples,
- PLA can efficiently find one of them dependent on initialization.

However, how to find the best one of these hyperplanes?

Margin

Definition

The smallest distance between the decision boundary and any of the samples.

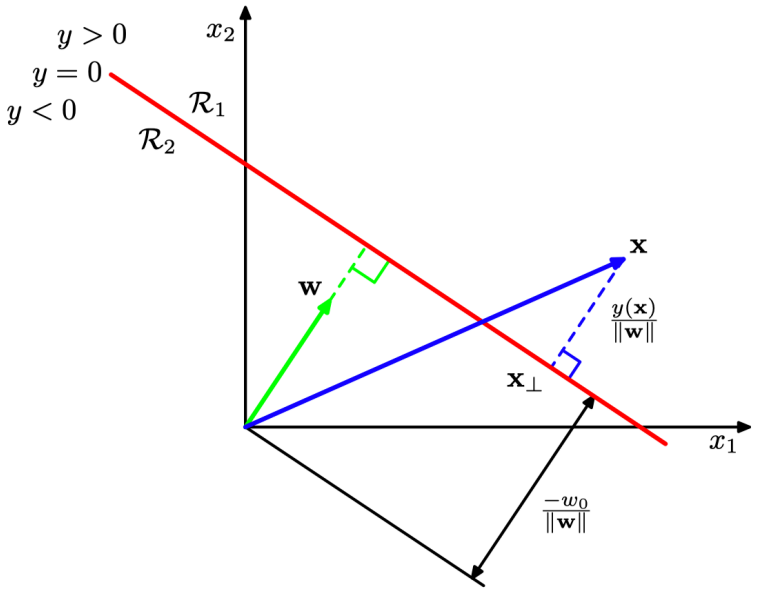

Figure: Linear Model

Maximizing Margin

For all correct solutions, we have $t_n y(x_n) > 0 \ \forall n$. As shown in the figure, the distance of a point $\mathbf{x}_n$ to the decision surface is given by

$$\frac{t_n y\left(\mathbf{x}_n\right)}{\|\mathbf{w}\|} = \frac{t_n\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b\right)}{\|\mathbf{w}\|}.$$

The margin is given by the smallest distance, and then we want to maximize the margin. The whole optimization problem can be written as

$$\arg\max_{\mathbf{w},b}\left\{\frac{1}{\|\mathbf{w}\|}\min_n\left[t_n\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b\right)\right]\right\}.$$

Canonical Representation

Setting

$$t_n \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_n + b\right) = 1$$

for the point that is closest to the surface. Therefore, all samples will satisfy the constraints

$$t_n \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_n + b\right) \geqslant 1, \quad n = 1, \ldots, N.$$

The original problem can be converted to the following optimization problem

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s. t.} \quad & t_n \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_n + b\right) \geqslant 1, \quad n = 1, \ldots, N. \end{aligned}$$

SVM Optimization

Introducing Lagrange multipliers $a_n \geq 0$, the Lagrangian function is given by

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \left\{ t_n \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_n + b \right) - 1 \right\}.$$

Hence, we want to optimize the Lagrangian function as follows

$$\min_{\mathbf{w}, b} \max_{\mathbf{a}} L(\mathbf{w}, b, \mathbf{a})$$

$$\text{s. t. } a_n \geq 0$$

Using strong duality, the problem becomes

$$\max_{\mathbf{a}} \min_{\mathbf{w}, b} L(\mathbf{w}, b, \mathbf{a})$$

$$\text{s. t. } a_n \geq 0$$

Dual Representation

Setting the derivatives of $L(\mathbf{w}, b, \mathbf{a})$ with respect to $\mathbf{w}, b$ to zero, we obtain the following two conditions

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

$$0 = \sum_{n=1}^{N} a_n t_n$$

Eliminating $\mathbf{w}$ and b from $L(\mathbf{w}, b, \mathbf{a})$, we arrive the dual form

$$\max_{\mathbf{a}} \widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

$$\text{s.t. } a_n \geqslant 0, \quad n = 1, \ldots, N$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

K.K.T Condition

The K.K.T condition can be derived as follows

$$\begin{aligned} a_n &\geqslant 0 \\ t_n y\left(\mathbf{x}_n\right) - 1 &\geqslant 0 \\ a_n\left\{t_n y\left(\mathbf{x}_n\right) - 1\right\} &= 0 \end{aligned}$$

which can be interpreted as

- $a_n = 0, t_n y(\mathbf{x}_n) - 1 \geq 0$, n-th sample plays no role for predicting,
- $a_n \geq 0, t_n y(\mathbf{x}_n) - 1 = 0$, n-th sample is called support vectors.

Soft Margin

In practice, the training samples are almost not linear separable. Hence, we may not want exact separation, which can also lead to poor generalization.

To tolerate "wrong side", we introduce *slack variables* $\xi_n \geq 0$, so the SVM optimization problem can be written as

$$\begin{aligned} \min_{\mathbf{w}} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n \\ \text{s.t. } & 1 - t_n y(\mathbf{x}_n) - \xi_n \leq 0, \forall n \\ & \xi_n \geq 0 \end{aligned}$$

- $C \to \infty \implies \xi_n \to 0$
- $C < \infty \implies \xi_n > 0$

Soft Margin

The corresponding Lagrangian is given by

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N} a_n \{t_n y(\mathbf{x}_n) - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n,$$

and K.K.T condition is given by

$$\begin{aligned} a_n &\geqslant 0 \\ t_n y(\mathbf{x}_n) - 1 + \xi_n &\geqslant 0 \\ a_n(t_n y(\mathbf{x}_n) - 1 + \xi_n) &= 0 \\ \mu_n &\geqslant 0 \\ \xi_n &\geqslant 0 \\ \mu_n\xi_n &= 0 \end{aligned}.$$

The optimization problem is

$$\max_{\mathbf{a},\boldsymbol{\mu}} \min_{\mathbf{w},b,\boldsymbol{\xi}} L(\mathbf{w}, b, \mathbf{a}, \boldsymbol{\mu})$$

$$\text{s.t. } a_n \geq 0, \mu_n \geq 0, \forall n.$$

Soft Margin

The dual form then is written as

$$\min_{\mathbf{a},\mu}\max_{\mathbf{w},b,\xi} L(\mathbf{w}, b, \mathbf{a}, \mu)$$

$$\text{s.t. } a_n \geq 0, \mu_n \geq 0, \forall n.$$

After optimizing $\mathbf{w}, b, \xi_n$, we arrive at

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0$$

$$\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n = C - \mu_n$$

Dual Lagrangian

$$\widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k\left(\mathbf{x}_n, \mathbf{x}_m\right).$$

Soft Margin

The optimization problem

$$\max_{\mathbf{a}} \widetilde{L}(\mathbf{a})$$
$$\text{s.t. } a_n \geqslant 0, \quad n = 1, \ldots, N$$
$$\sum_{n=1}^{N} a_n t_n = 0$$
$$C - a_n - \mu_n = 0.$$

Summary

Pros

- Rigorous mathematical theory, strong interpretability
- Able to solve the high-dimensional problem with kernels methods
- Computational complexity depends on support vectors

Cons

- Long training time, e.g. SMO algorithm
- Large space if using kernels methods
- Predicting time depends on the number of support machines