ME 491 Lecture 20

5.7 Support Vector Machines (SVM)

Core idea: optimize for hyperplanes to split data into distinct clusters.

Three constructs:

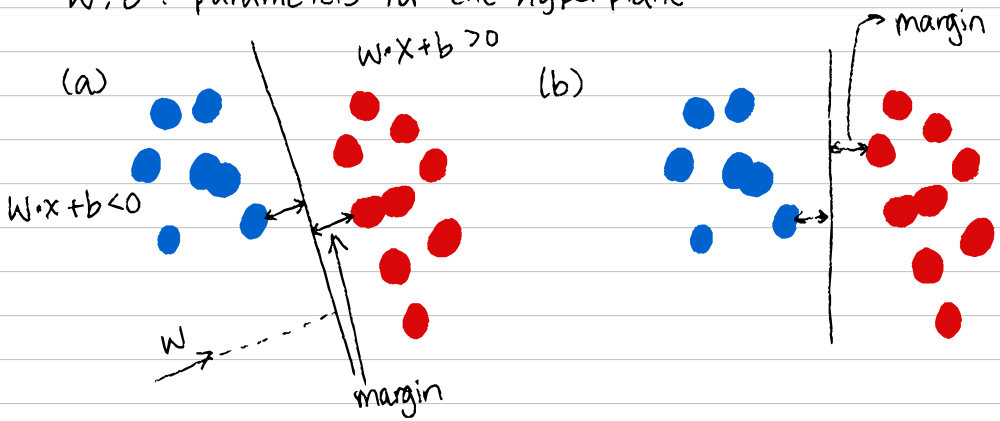
1. Linear SVM
2. Nonlinear SVM
3. Kernel methods for SVM.

* Linear SVM

Construct a hyperplane:

$$W \cdot X + b = 0$$

$\vec{W}, b$: parameters for the hyperplane



The goal for SVM is not only optimize a decision line that makes the fewest labeling errors, but also to optimize the largest margin between data.

After the hyperplane is constructed, we can classify test data using the following classifier.

$$y_j(W \cdot x_j + b) = \text{sign}(w \cdot x_j + b) = \begin{cases} +1 & \text{red ball} \\ -1 & \text{blue ball} \end{cases}$$

To determine $\vec{w}$ and b in a principled way, we need to formulate an optimization problem.

To construct the optimization objective function, we define a loss function:

$$\ell(y_j, \bar{y}_j) = \ell(y_j, \text{Sign}(w \cdot x_j + b)) = \begin{cases} 0 & \text{if } y_j = \text{sign}(w \cdot x_j + b) \\ +1 & \text{if } y_j \neq \text{sign}(w \cdot x_j + b) \end{cases}$$

To state more simply:

$$\ell(y_j, \bar{y}_j) = \begin{cases} 0 & \text{if data is correctly labeled} \\ +1 & \text{if data is incorrectly labeled} \end{cases}$$

The loss function right now only take into account of mislabeling, we also want to make the margin as large as possible.

We can then frame the linear SVM optimization problem as:

$$\underset{w,b}{\text{argmin}} \sum_{j=1}^{m} \ell(y_j, \bar{y}_j) + \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad \min_j |x_j \cdot w| = 1$$

The loss function here is discrete, so it is very hard to optimize. Most optimization algorithms are based on some version of gradient descent. So requires smooth objective functions.

A common formulation is:

$$\underset{w,b}{\text{argmin}} \sum_{j=1}^{m} H(y_j, \bar{y}_j) + \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad \min_j |x_j \cdot w| = 1$$

$$H(y_j, \bar{y}_j) = \max(0, 1 - y_j \cdot \bar{y}_j) \quad \text{Hinge loss function}$$

* Nonlinear SVM

To perform nonlinear SVM, we simply maps data to a nonlinear, higher dimensional space:

$$X \to \Phi(x) \qquad \Phi(x): \text{new observables of the data}$$

Hyperplane function:

$$f(x) = w \cdot \Phi(x) + b, \quad y_j \in \{\pm 1\} \text{ for each point } f(x_j)$$

- Simple example of nonlinear mapping (Fig. 5.23)

$$X = (x_1, x_2) \Rightarrow (z_1, z_2, z_3) := (x_1, x_2, x_1^2 + x_2^2)$$

The ability of SVM to embed in higher-dimensional nonlinear spaces makes it one of the most successful machine learning algorithms developed.

The underlying optimization algorithm remains unchanged with little modification to the labeling function

$$\tilde{y}_j = \text{Sign}(w \cdot \Phi(x_j) + b)$$

* Kernel Methods for SVM

Some time your data has so many features that computing the vector $\vec{w}$ is prohibitively expensive and may not even be represented explicitly in memory. We need to apply the **kernel trick** to solve the problem.

We represent $\vec{w}$ as

$$\vec{w} = \sum_{j=1}^{m} \alpha_j \Phi(x_j)$$

α_j: parameters that weight the different nonlinear observable functions

We can have a generalized hyperplane function:

$$f(x) = \sum_{j=1}^{m} \alpha_j \Phi(x_j) \cdot \Phi(x) + b$$

The kernel function is then defined as

$$K(x_j, x) = \Phi(x_j) \cdot \Phi(x)$$

We can formulate an optimization function

$$\underset{\alpha, b}{\operatorname{argmin}} \sum_{j=1}^{m} H(y_j, \bar{y}_j) + \frac{1}{2} \left\| \sum_{j=1}^{m} \alpha_j \Phi(x_j) \right\|^2 \text{ subject to } \min_j |x_j \cdot w| = 1$$

During the optimization process, we now optimize over α_j instead of the whole $\vec{w}$.

Two of the most commonly used kernel functions:

- radial basis function: $K(x_j, x) = \exp(-\gamma \|x_j - x\|^2)$
- polynomial kernel: $K(x_j, x) = (x_j \cdot x + 1)^N$

N: degree of polynomials to be considered
γ: width of Gaussian kernel measuring the distance between individual data points x_j and the classification line.