DDA5001 Machine Learning

Linear Classification III: Support Vector Machine (SVM)

Xiao Li

School of Data Science
The Chinese University of Hong Kong, Shenzhen

Recap: Overfitting

Overfitting: Smaller and smaller training error (small $\mathrm{Er}_{\mathrm{in}}$) but larger and larger $\mathrm{Er}_{\mathrm{out}}$).

Catalysts: Sample number n, noise, target complexity.

Validation: Estimate $\mathrm{Er}_{\mathrm{out}}$ to avoid overfitting:

$$\underbrace{\mathrm{Er}_{\mathrm{out}}(f)}_{\text{validation estimates this quantity}} \leq \mathrm{Er}_{\mathrm{in}}(f) + \text{overfit penalty.}$$

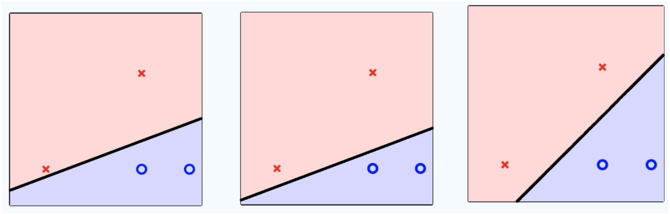
Regularization: Choose a simple $\hat{f}$ from the complex $\mathcal{H}$:

$$\min_{f \in \mathcal{H}} \mathrm{Er}_{\mathrm{in}}(f) + \Omega(f).$$

Overfitting helps by mitigating noise.

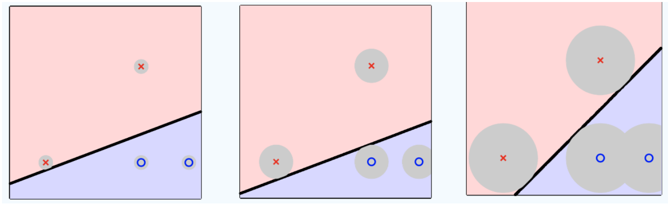
Support Vector Machine (SVM)

Perceptron: Which is The Optimal Classifier?



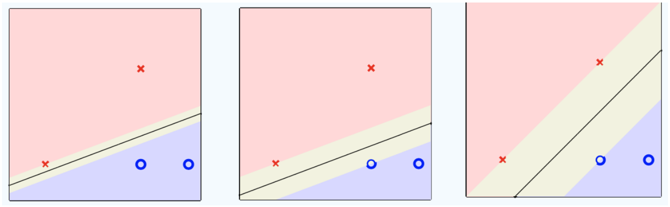
- Which classifier is possibly generated by perceptron? All the three.
- Apart from the above three lines, the perceptron learned model can actually have infinitely many possibilities once it classifies the training data.
- From VC analysis perspective, all the classifiers provide $\mathrm{Er}_{\mathrm{in}} = 0$, and hence we have the same $\mathrm{Er}_{\mathrm{out}}$-bound for every classifier. VC analysis is a worst case result for any classifier in $\mathcal{H}$, it does not count some other factors (we will specify in the next slide).
- Intuitively, the third classifier is the best.

Noise



- In practice, there are measurement errors – noise (stochastic). The noise effect is illustrated as shaded area around each point. The true data point can lie anywhere within this ‘region of uncertainty’ due to the measurement error.
- A classifier is robust to the noise if it can classify the true data points correctly.
- A classifier that can tolerate more noise, quantified as the radius of the shaded area, is more robust.
- In overfitting section, we know that noise is one of the main causes of overfitting. Regularization helps us combat noise and avoid overfitting. In our example, the rightmost classifier is more robust to noise without compromising $\mathrm{Er}_{\mathrm{in}}$. It is better ‘regularized’.

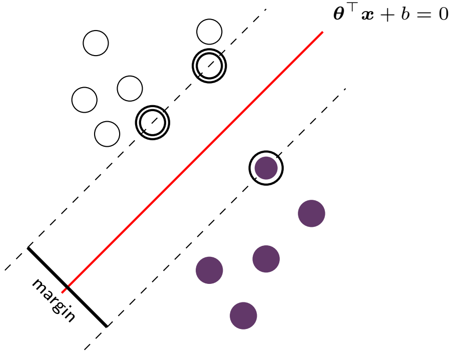
Quantify Noise Tolerance From The Classifier



- We can draw the separation region of the classifier that is formed by the area between two classes. The thickness reflects the amount of noise the classifier can tolerate. If any data point is perturbed by at most the thickness of each side, it will still be on the correct side of the classifier.
- The maximum thickness (noise tolerance) for a classifier is called its margin.

Maximal Margin Hyperplane

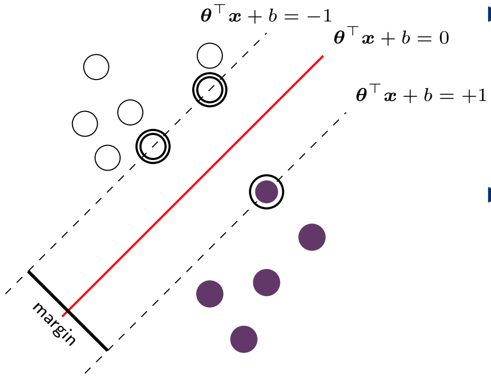
Support Vectors and Maximal Margin



- Select two parallel hyperplanes that separate the two classes and the distance between them is as large as possible.
- Support vectors determines these two parallel hyperplanes.
- Margin is the distance bounded by these two hyperplanes.
- The maximum-margin hyperplane lies halfway between them, we define it as

$$\boldsymbol{\theta}^\top \boldsymbol{x} + b = 0 \quad (\boldsymbol{\theta}, b \text{ are trainable parameters})$$

Expression of The Two Parallel Hyperplane


$\boldsymbol{\theta}^\top \boldsymbol{x} + b = -1$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = 0$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = +1$
margin

- Denote $\{\boldsymbol{x}^s\}$ as the support vectors.
- The two parallel hyperplanes pass the support vectors should have the expression

$$\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = \pm c.$$

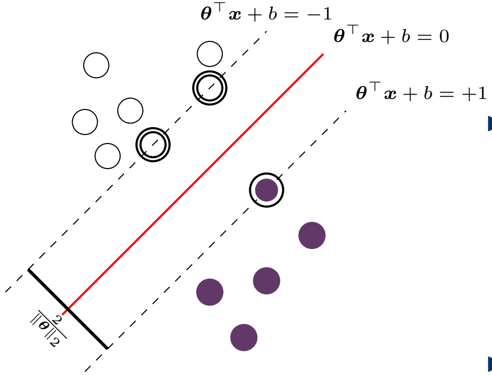
- We can apply normalization to $\boldsymbol{\theta}$ and b to get

$$\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = 1.$$

and

$$\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = -1.$$

Expression of The Margin



$\boldsymbol{\theta}^\top \boldsymbol{x} + b = -1$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = 0$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = +1$
$\frac{2}{\|\boldsymbol{\theta}\|_2}$

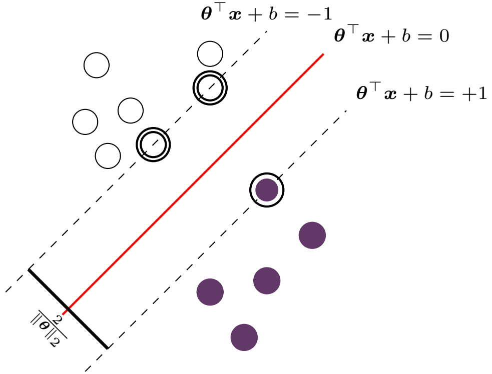
- Margin $= 2\times$ distance between $\boldsymbol{x}^s$ and the maximum-margin hyperplane.
- Margin also equals to the distance between two parallel hyperplanes: $\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = 1$ and $\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = -1$.
- Fact: The distance between two parallel hyperplanes, $\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = a_1$ and $\boldsymbol{\theta}^\top \boldsymbol{x}^s + b = a_2$, is given by

$$\frac{|a_1 - a_2|}{\|\boldsymbol{\theta}\|_2}.$$

- Thus, The margin:

$$\frac{2}{\|\boldsymbol{\theta}\|_2}$$

The Learning Problem



$\boldsymbol{\theta}^\top \boldsymbol{x} + b = -1$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = 0$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = +1$
$\frac{2}{\|\boldsymbol{\theta}\|_2}$

The principle: Maximize the margin

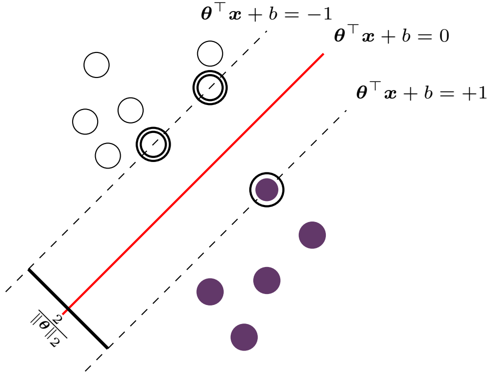
$$\underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\text{maximize}} \;\; \frac{2}{\|\boldsymbol{\theta}\|_2},$$

which is equivalent to

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\text{minimize}} \;\; \|\boldsymbol{\theta}\|_2^2.$$

Is it done?

The Learning Problem



$\boldsymbol{\theta}^\top \boldsymbol{x} + b = -1$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = 0$
$\boldsymbol{\theta}^\top \boldsymbol{x} + b = +1$
$\frac{2}{\|\boldsymbol{\theta}\|_2}$

We forget the constraint: $|\boldsymbol{\theta}^\top \boldsymbol{x}^s + b| = 1$. Since we often do not know which data point is a support vector as a prior, we need to use all the data points. Thus, we have

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^d, b\in\mathbb{R}}{\text{minimize}} \ \|\boldsymbol{\theta}\|_2^2$$
$$\text{subject to } |\boldsymbol{\theta}^\top \boldsymbol{x}_i + b| \geq 1, \quad i = 1, \ldots, n$$

Hard-margin SVM.

Note that $|\boldsymbol{\theta}^\top \boldsymbol{x}_i + b| = y_i(\boldsymbol{\theta}^\top \boldsymbol{x}_i + b)$.

We finally have the optimization formulation

$$\boxed{\begin{aligned} &\underset{\boldsymbol{\theta}\in\mathbb{R}^d, b\in\mathbb{R}}{\text{minimize}} \ \|\boldsymbol{\theta}\|_2^2 \\ &\text{subject to } \ y_i(\boldsymbol{\theta}^\top \boldsymbol{x}_i + b) \geq 1, \quad i = 1, \ldots, n \end{aligned}}$$

This is called hard-margin support vector machine (SVM).

- The word "hard-margin" means that all training data points must be correctly classified.
- This is ensured by the constraints, and can be regarded as a perceptron classifier.
- Thus, hard-margin SVM can be regarded as: Choose a classifier from all possible perceptron classifiers that has the largest margin.
- We motivated by that the most robust classifier is better 'regularized', which corresponds to the weight decay term in the objective function.

Optimization Method

$$\begin{aligned} &\underset{\boldsymbol{\theta}\in\mathbb{R}^d, b\in\mathbb{R}}{\text{minimize}} \ \|\boldsymbol{\theta}\|_2^2 \\ &\text{subject to } \ y_i(\boldsymbol{\theta}^\top \boldsymbol{x}_i + b) \geq 1, \quad i = 1, \dots, n \end{aligned}$$

- The learning problem is to minimize a (convex) quadratic function over linear constraints. In optimization society, this is a pretty much well developed area, called quadratic programming. There are plenty of solvers for quadratic programming, which are quite stable and fast. Thus, we can apply the well established solver for the hard-margin SVM. The details are out of the scope of the course.
- However, solving hard-margin SVM formulation is rare in practice ($\rightsquigarrow$later).

Is Max Margin Theoretically Better?

Regularization View and Modified VC Analysis

- As we said, we can relate the minimization of ℓ_2-norm to the ℓ_2-regularization, which in part explains it is better 'regularized' to tolerate noise.

More rigorously, we can have the modified VC analysis for max margin hyperplane.

- The VC dimension of a hypothesis set is the maximum number of points that can be shattered. For any (binary) linear classifier, we have shown that it is $d + 1$.
- Consider the hypothesis set $\mathcal{H}_\rho$ containing all hyperplanes of margin at least ρ. Our goal is to show that restricting the hypothesis set to max margin classifier can decrease the number of points that can be shattered, that is, $d_{\mathsf{VC}}(\rho) < d + 1$.

Modified VC Analysis

In general, we can prove the following result

Theorem: VC dimension of margin-ρ hyperplanes

Suppose the input space is the ball of radius R in $\mathbb{R}^d$, that is $\|\boldsymbol{x}\| \leq R$. Then,

$$d_{\mathsf{VC}}(\rho) \leq \lceil R^2/\rho^2 \rceil + 1.$$

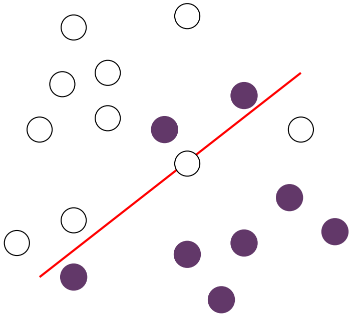
- Since its VC dimension is at most $d+1$. Thus, we can have

$$d_{\mathsf{VC}}(\rho) \leq \min\{\lceil R^2/\rho^2 \rceil, d\} + 1.$$

- The bound suggests that the margin ρ can be used to control the model complexity. Hence, seeking for a max margin (max ρ) can lead to better $\mathrm{Er}_{\mathrm{out}}$.

Soft-margin SVM

Non-linearly Separable Case



- A more practical case. For example, medium noise.
- Issue with hard-margin: some points will never satisfy the margin constraint.
- What will happen to the hard-margin SVM problem? Infeasible.
- How to solve this issue? Relaxation.

The Soft-margin SVM

We first introduce the Hinge loss:

$$h(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i) = \max(0, 1 - y_i(\boldsymbol{\theta}^\top \boldsymbol{x}_i + b))$$

- $h(\boldsymbol{\theta}) = 0$ if the point $\boldsymbol{x}_i$ lies in the correct side.
- For data on the wrong side, the function's value is proportional to the distance from the maximum-margin hyperplane.

The soft-margin SVM is to solve

$$\boxed{\underset{\boldsymbol{\theta} \in \mathbb{R}^d, b \in \mathbb{R}}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y_i(\boldsymbol{\theta}^\top \boldsymbol{x}_i + b)) + \lambda \|\boldsymbol{\theta}\|_2^2}$$

- $\lambda > 0$ determines the trade-off between increasing the margin and ensuring that $\boldsymbol{x}_i$ lies on the correct side of the margin.
- If the data is linearly separable, then we can choose a very small λ to let the soft-margin SVM works the same as hard-margin SVM.
- If data is not linearly separable, soft-margin SVM can still provide a meaningful classifier.

Subgradient method for Soft-margin SVM

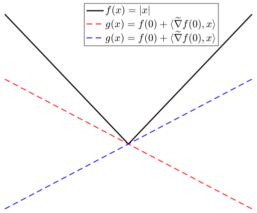
How to Solve The Soft-margin SVM?

- The hinge loss is not differentiable.
- One popular approach is to consider the dual formulation and then apply coordinate algorithm (in LibSVM). We consider another popular method, the subgradient method.

Subgradient: A generalization of gradient.

The subgradient of a convex $\mathcal{L}$ at $\boldsymbol{\theta}$ is defined as any element of the following subdifferential

$$\partial \mathcal{L}(\boldsymbol{\theta}) = \{\boldsymbol{s} \in \mathbb{R}^d : \mathcal{L}(\boldsymbol{w}) \geq \mathcal{L}(\boldsymbol{\theta}) + \boldsymbol{s}^\top (\boldsymbol{w} - \boldsymbol{\theta}), \quad \forall \boldsymbol{w} \in \mathbb{R}^d\}$$



Subgradient Method

Soft-margin SVM:

$$\boxed{\underset{\boldsymbol{\theta}\in\mathbb{R}^d, b\in\mathbb{R}}{\text{minimize}}\ \mathcal{L}(\boldsymbol{\theta}) := \frac{1}{n}\sum_{i=1}^{n}\max(0, 1 - y_i(\boldsymbol{\theta}^\top \boldsymbol{x}_i + b)) + \lambda\|\boldsymbol{\theta}\|_2^2}$$

Algorithmic design framework

$$\boxed{\boldsymbol{\theta}_{k+1} = \underset{\boldsymbol{\theta}\in\Theta}{\text{argmin}}\ \left\{ l_k(\boldsymbol{\theta}) = q_k(\boldsymbol{\theta}) + \frac{1}{2\mu_k}\|\boldsymbol{\theta} - \boldsymbol{\theta}_k\|_2^2 \right\}}$$

- When $q_k(\boldsymbol{\theta})$ is linear approximation using subgradient $\Longrightarrow$ **subgradient method**

Subgradient method

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \mu_k \boldsymbol{s}_k, \quad \forall \boldsymbol{s}_k \in \partial\mathcal{L}(\boldsymbol{\theta}_k)$$

Learning Rate in Subgradient Method

- Assume we are going to minimize

$$\min_{\theta} \ \mathcal{L}(\theta) = |\theta|$$

- The subdifferential of $|\boldsymbol{\theta}|$ is $\partial\mathcal{L}(\theta) = \operatorname{sign}(\theta)$.
- What will happen if we choose $\theta^0 = 0.05$ and $\mu_k = 0.1$? It will never converge.
- This is due to the fact that subgradient method is a non-descent method.

Solution:

- The learning rate in subgradient method is often chosen as diminishing / decaying.
- Decaying learning rate is a solution for ensuring convergence for non-descent method.

Two rule of thumb:

$$\mu_k = \frac{c}{k}. \quad \frac{c}{\sqrt{k}}, \quad \text{or} \quad \mu_k = c\rho^k \ (0 < \rho < 1), \quad \text{for some } c > 0.$$

Extension: SVM for Multi-class Classification

- SVM is intrinsically designed for binary classification. Unlike LR, its derivation cannot be generalized to multi-class case.
- The dominant approach for applying SVM to multi-class classification is to reduce the single multi-class problem into multiple binary classification problems.

Typical approaches include:

- One-verses-all: Classify one class from all other classes.
- One-versus-one: Classify every pair of classes.

$\rightsquigarrow$ Next lecture: Kernel method.