So Far…

- Our goal (supervised learning):
  - To learn a set of discriminant functions

- Bayesian framework
  - We could design an optimal classifier if we knew:
    - $P(\omega_i)$: priors and $P(x \mid \omega_i)$: class-conditional densities
    - Using training data to estimate $P(\omega_i)$ and $P(x \mid \omega_i)$

- Directly learning discriminant functions from the training data
  - We only know the form of the discriminant functions
  - Linear Methods for Regression

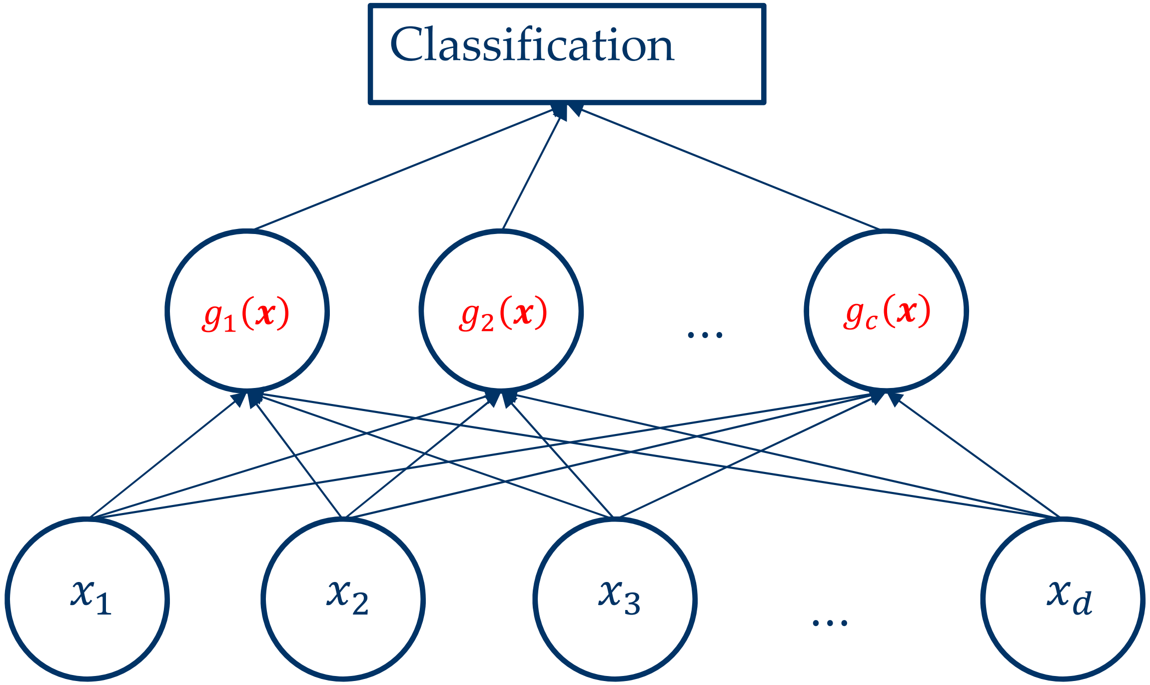
Linear Methods for Classification

Deng Cai (蔡登)

College of Computer Science
Zhejiang University

dengcai@gmail.com

Discriminant Functions and Classifiers



- Set of discriminant functions: $g_i(\boldsymbol{x}),\ i = 1, \cdots, c$

$$g(\boldsymbol{x}) = \left(XX^T + \lambda \boldsymbol{I}\right)^{-1} X\boldsymbol{y}$$

- Classifier assigns a feature vector $\boldsymbol{x}$ to class ω_i if:

$$g_i(\boldsymbol{x}) > g_j(\boldsymbol{x}), \qquad \forall j \neq i$$

Linear Regression of an Indicator Matrix

$$g(\boldsymbol{x}) = \left(XX^T + \lambda \boldsymbol{I}\right)^{-1} X\boldsymbol{y}$$

$$\mathbf{y} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 2 \\ \vdots \\ 2 \\ \vdots \\ c \\ \vdots \\ c \end{bmatrix} \qquad Y = \begin{bmatrix} 1 & 0 & & 0 \\ \vdots & \vdots & & \vdots \\ 1 & 0 & & 0 \\ 0 & 1 & & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & 1 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & 1 \end{bmatrix}$$

- One VS. Rest

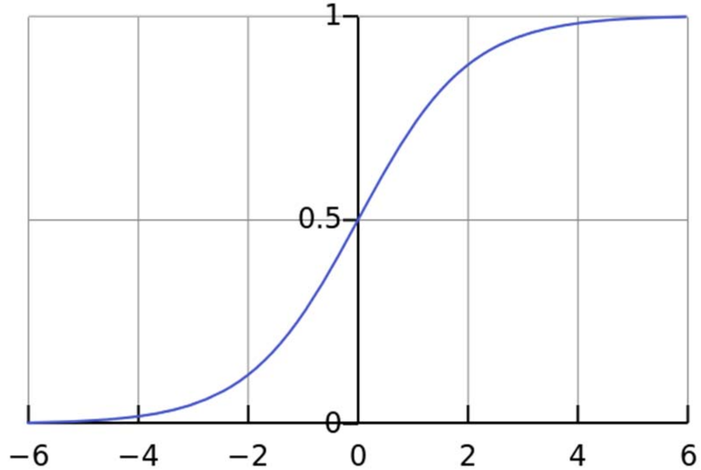
Sigmoid function (logistic function)

$$\sigma(t) = \frac{e^t}{1+e^t} = \frac{1}{1+e^{-t}}$$

- It is the cumulative distribution function (CDF) of the standard logistic distribution.
- While the input can have any value from $-\infty$ to $+\infty$, the output takes only values between 0 and 1, and hence is interpretable as probability

$$\sigma: \ \mathrm{R} \rightarrow (0,1)$$

- S-shaped



Logistic Regression

- **Logistic Regression (LR)** is a classification model used to describe the relationship between a **categorical** dependent variable and one or several independent variables by estimating **probabilities** using **sigmoid function**.

$$P(y_i = 1|\boldsymbol{x}_i, \boldsymbol{a}) = \sigma(\boldsymbol{a}^T\boldsymbol{x}_i) = \frac{1}{1 + e^{-\boldsymbol{a}^T\boldsymbol{x}_i}}$$

$$P(y_i = -1|\boldsymbol{x}_i, \boldsymbol{a}) = 1 - \sigma(\boldsymbol{a}^T\boldsymbol{x}_i) = 1 - \frac{1}{1 + e^{-\boldsymbol{a}^T\boldsymbol{x}_i}} = \frac{1}{1 + e^{\boldsymbol{a}^T\boldsymbol{x}_i}}$$

$$P(y_i = \pm 1|\boldsymbol{x}_i, \boldsymbol{a}) = \sigma(y_i\boldsymbol{a}^T\boldsymbol{x}_i) = \frac{1}{1 + e^{-y_i\boldsymbol{a}^T\boldsymbol{x}_i}}$$

Maximum Likelihood Estimation for Logistic Regression

$$P(y_i = \pm 1 | \boldsymbol{x}_i, \boldsymbol{a}) = \sigma(y_i \boldsymbol{a}^T \boldsymbol{x}_i) = \frac{1}{1 + e^{-y_i \boldsymbol{a}^T \boldsymbol{x}_i}}$$

$$P(D) = \prod_{i \in I} \sigma(y_i \boldsymbol{a}^T \boldsymbol{x}_i)$$

$$l(P(D)) = \sum_{i \in I} \log\left(\sigma(y_i \boldsymbol{a}^T \boldsymbol{x}_i)\right) = -\sum_{i \in I} \log\left(1 + e^{-y_i \boldsymbol{a}^T \boldsymbol{x}_i}\right)$$

- Logistic Regression:

$$E(\boldsymbol{a}) = \sum_{i \in I} \log\left(1 + e^{-y_i \boldsymbol{a}^T \boldsymbol{x}_i}\right)$$

  - $E(\boldsymbol{a})$: a convex function of $\boldsymbol{a}$?

Homework

Minimize a Differentiable Function

- Objective Function of Logistic Regression:

$$E(\boldsymbol{a}) = \sum_{i \in I} \log\left(1 + e^{-y_i \boldsymbol{a}^T \boldsymbol{x}_i}\right)$$

- Objective Function of Linear Regression:

$$E(\boldsymbol{a}) = \sum_{i \in I} \left(y_i - \boldsymbol{a}^T \boldsymbol{x}_i\right)^2$$

- Gradient Descent

Gradient Descent

Gradient Descent

- A first-order optimization algorithm.
- Can find a local minimum of a function
- One takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point.
- If instead one takes steps proportional to the positive of the gradient, one approaches a local maximum of that function;
- Another name: steepest descent

Gradient Descent

- If the multivariable function $J(\boldsymbol{w})$ is defined and differentiable in a neighborhood of a point $\boldsymbol{a}$, then $J(\boldsymbol{w})$ decreases fastest if one goes from $\boldsymbol{a}$ in the direction of the negative gradient of J at $\boldsymbol{a}$, $-\nabla J(\boldsymbol{a})$.
- If $\boldsymbol{b} = \boldsymbol{a} - \gamma \nabla J(\boldsymbol{a})$, for γ small enough, then $J(\boldsymbol{a}) \geq J(\boldsymbol{b})$.
- With this observation in mind, one starts with a guess $\boldsymbol{w}_0$ for a local minimum of J, and considers the sequence $\boldsymbol{w}_0, \boldsymbol{w}_1, \boldsymbol{w}_2, \cdots$ such that

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \gamma \nabla J(\boldsymbol{w}_n), n \geq 0$$

- We have

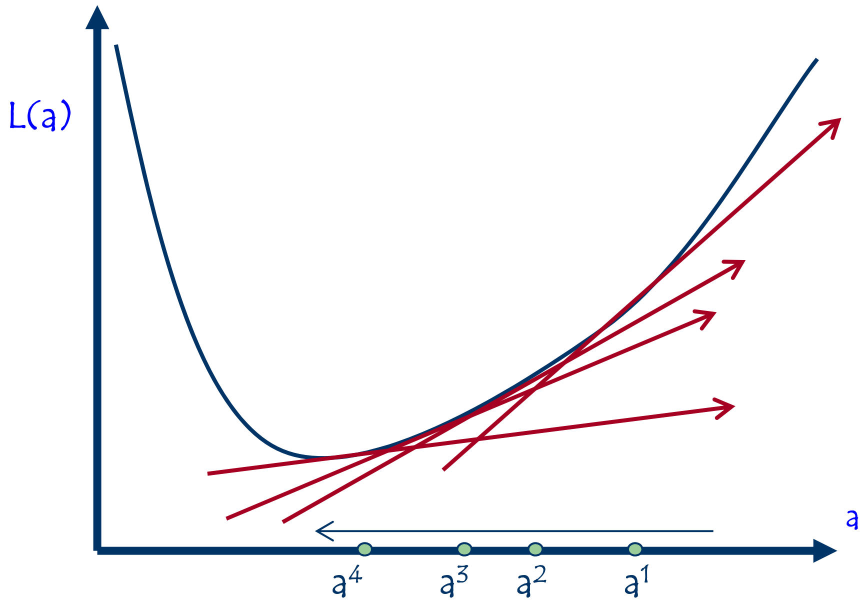
$$J(\boldsymbol{w}_0) \geq J(\boldsymbol{w}_1) \geq J(\boldsymbol{w}_2) \geq \cdots$$

- so hopefully the sequence $(\boldsymbol{w}_n)$ converges to the desired local minimum.
- Note that the value of the step size γ is allowed to change at every iteration.

Gradient Descent Algorithm

Algorithm 1 (Basic gradient descent)

<u>**begin**</u> <u>**initialize**</u> $\mathbf{a}$, criterion $\theta, \eta(\cdot), k = 0$
    <u>**do**</u> $k \leftarrow k + 1$
        $\mathbf{a} \leftarrow \mathbf{a} - \eta(k)\nabla J(\mathbf{a})$
    <u>**until**</u> $\eta(k)\nabla J(\mathbf{a}) < \theta$
<u>**return**</u> $\mathbf{a}$
<u>**end**</u>

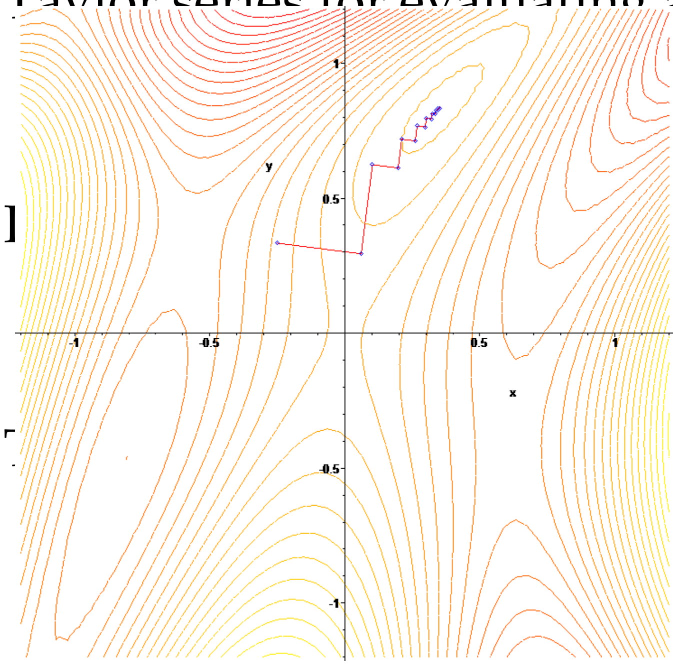


Minimize a Differentiable Function

- If the multivariable function $J(\boldsymbol{w})$ is defined and differentiable in a neighborhood of a point $\boldsymbol{a}$, then $J(\boldsymbol{w})$ decreases fastest if one goes from $\boldsymbol{a}$ in the direction of the negative gradient of J at $\boldsymbol{a}$, $-\nabla J(\boldsymbol{a})$.

- If $\boldsymbol{b} = \boldsymbol{a} - \gamma \nabla J(\boldsymbol{a})$, for γ small enough, then $J(\boldsymbol{a}) \geq J(\boldsymbol{b})$. Why?

- Taylor series for evaluating a function

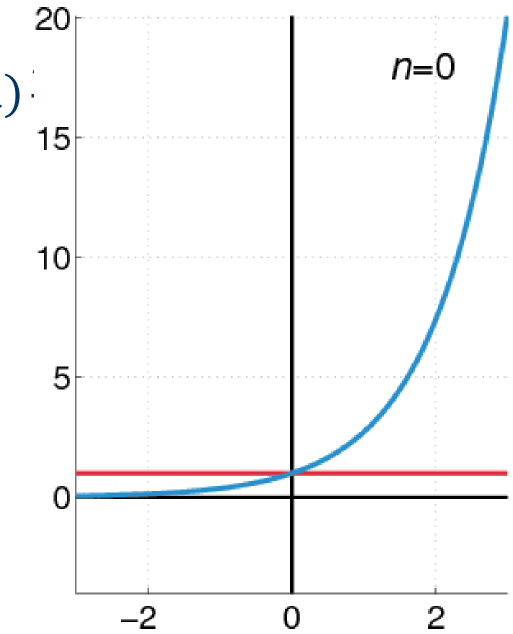


$$\ldots a + E''(a)\frac{\Delta a^2}{2!} + E'''(a) \ldots$$

-] … tion

a

- … Gradient Decent



Minimize a Differentiable Function

$$E(a+\Delta a) = E(a) + E'(a)\Delta a + E''(a)\frac{\Delta a^2}{2!} + E'''(a)\frac{\Delta a^3}{3!} + \cdots$$

- If we use a linear approximation, then Gradient Decent

$$\Delta a = -\eta E'(a)$$

- If we use a quadratic approximation, then
- Newton's Method

Choose Δa that $E'(a)\Delta a + E''(a)\frac{\Delta a^2}{2!}$ is minimum

$$E'(a) + E''(a)\Delta a = 0 \qquad \Delta a = -\frac{E'(a)}{E''(a)}$$

$$\Delta \boldsymbol{a} = -\eta[\mathbf{H}E(\boldsymbol{a})]^{-1}E'(\boldsymbol{a})$$

- Quasi-Newton
  - DFP, BFGS, L-BFGS, OWL-QN

Regularized Logistic Regression

$$E(\boldsymbol{a}) = \sum_{i \in I} \log\left(1 + e^{-y_i \boldsymbol{a}^T \boldsymbol{x}_i}\right) + \lambda \sum_{j=1}^{p} |a_j^2|$$

- L2-regularizer
- L1-regularizer (Sparse Logistic Regression)

Software

- LIBLINEAR
  - http://www.csie.ntu.edu.tw/~cjlin/liblinear/

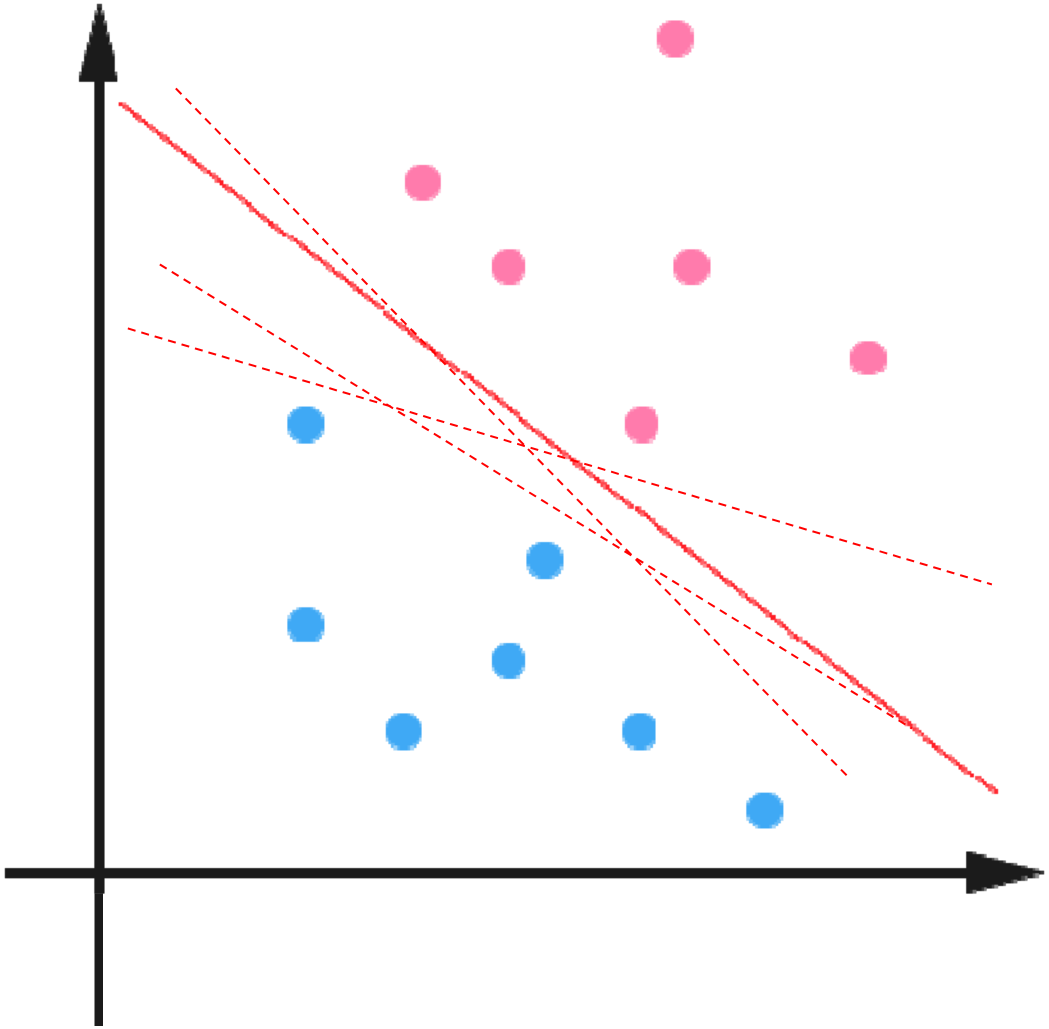
Support Vector Machine

Two-category Linearly Separable Case

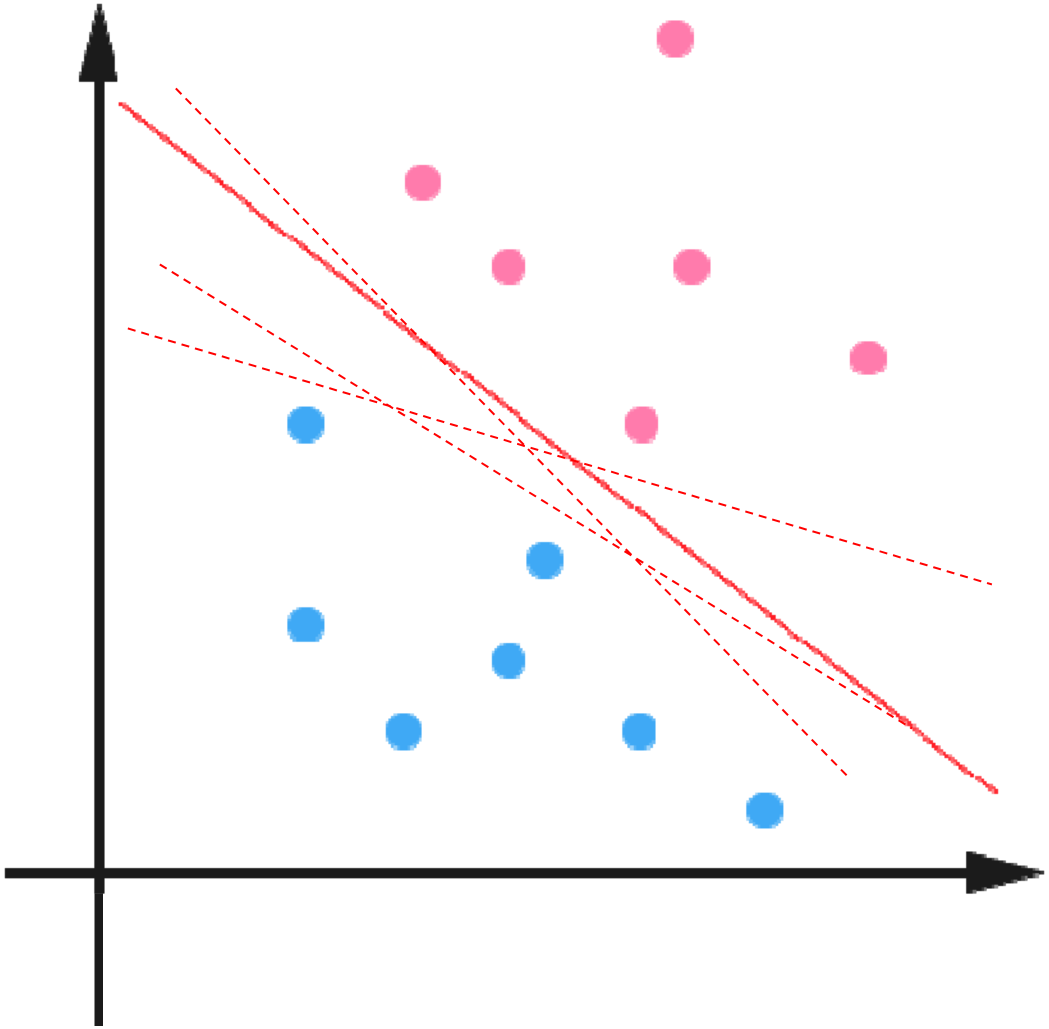
- If
  - $\boldsymbol{w}^T\boldsymbol{x} > 0$ for examples from the positive class.
  - $\boldsymbol{w}^T\boldsymbol{x} < 0$ for examples from the negative class.

- Such a weight vector $\boldsymbol{w}$ is called a *separating vector* or a *solution vector*
  - Does solution vector unique?

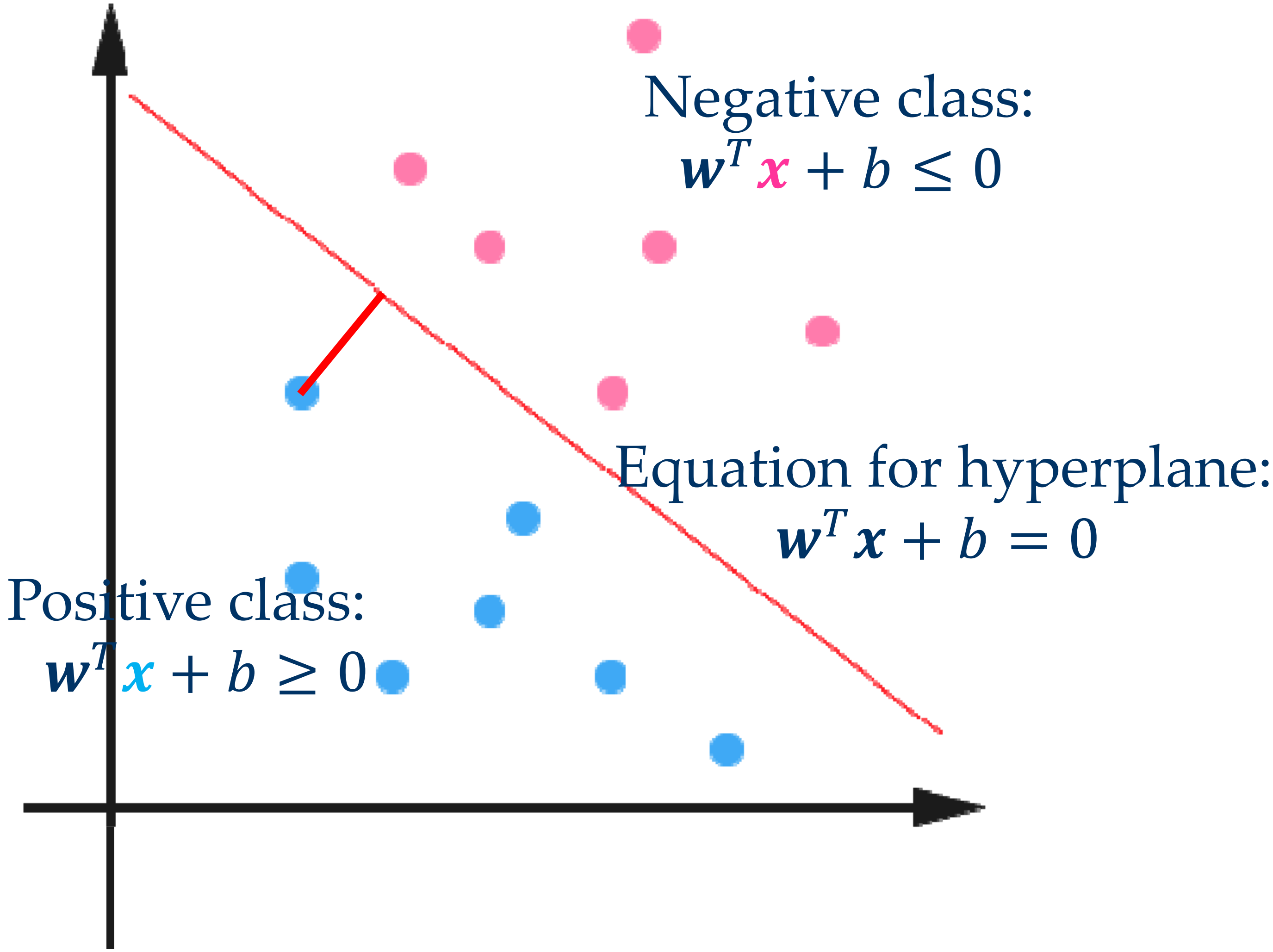
Non-uniqueness of hyperplane classifier



Which one is better?

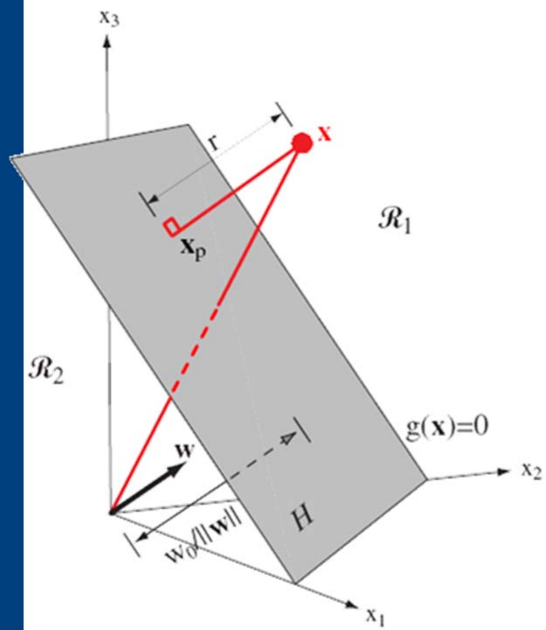


Binary Classification



Geometrical Margin

- Define γ as the distance from $\boldsymbol{x}$ to the hyperplane
  - Computation: let the projection of $\boldsymbol{x}$ into the hyperplane be $\boldsymbol{x}_0$, then we have



$$\boldsymbol{x} = \boldsymbol{x}_0 + y\gamma \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$$

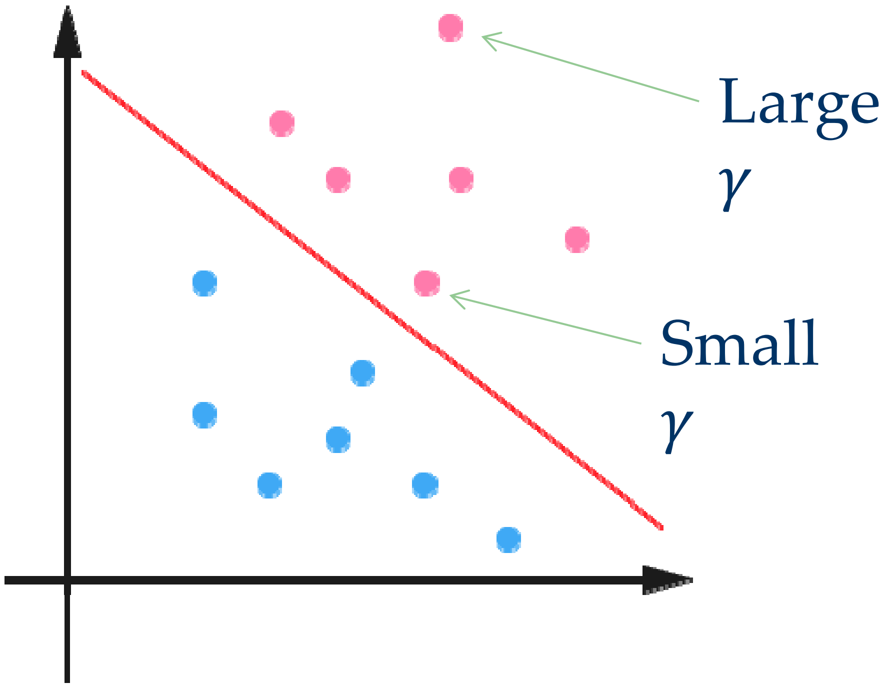
$$\boldsymbol{w}^T\left(\boldsymbol{x} - y\gamma \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\right) + b = 0$$

$$\gamma = y\frac{\boldsymbol{w}^T\boldsymbol{x} + b}{\|\boldsymbol{w}\|}$$

- γ: **geometrical margin**

Geometrical Margin

$$\gamma = y\frac{\boldsymbol{w}^T\boldsymbol{x} + b}{\|\boldsymbol{w}\|}$$



If the hyperplane moves a little, points with small γ will be affected, but points with large γ won't

Maximum Margin Classifier

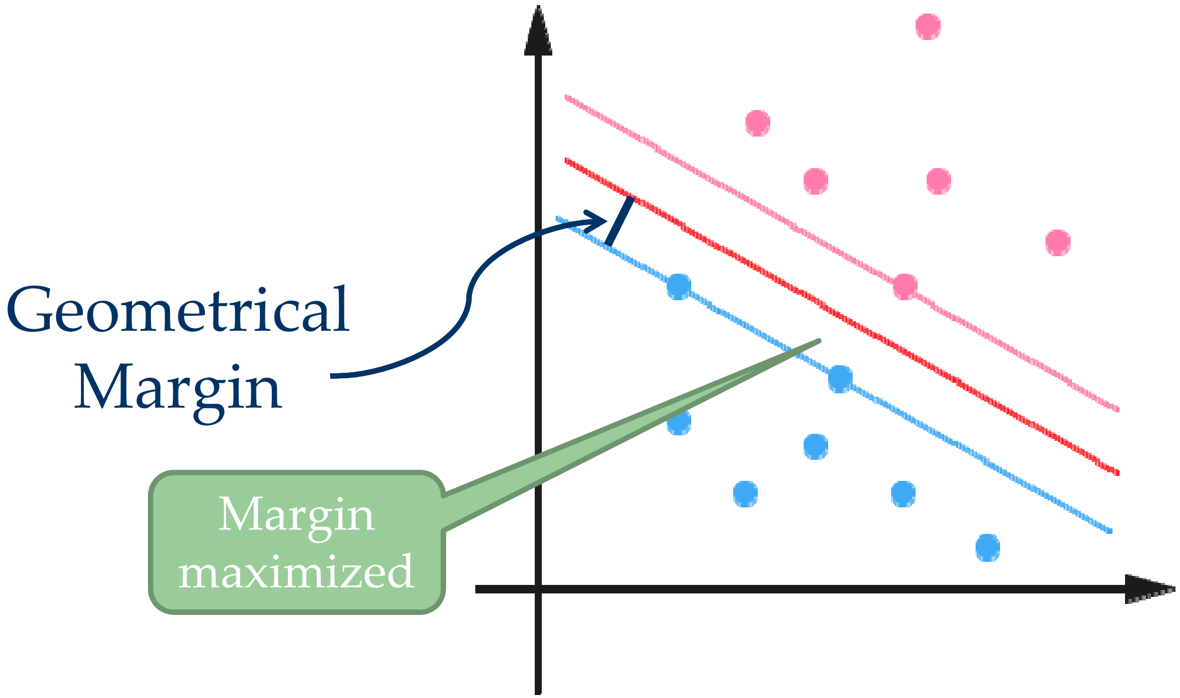
- Define the margin of a dataset be the minimum margin of each data point
- Maximum margin classifier tries to achieve the maximum possible margin for a given dataset
  - Thus maximize the *confidence* of classifying the dataset
- Goal: Find the hyperplane with the largest margin

Why Maximum Margin?

- Intuitively this feels safest
- If we've made a small error in the location of the boundary, this gives us least chance of causing misclassification
- There's some theory (using VC dimension) that is related to the proposition that this is a good thing.
- Empirically it works very, very well.

Maximum Margin Classifier

- Geometrical margin is a value uniquely determined by the position of the hyperplane
- If we scale $\boldsymbol{w}$, γ will not change as long as the hyperplane is kept fixed



Maximum Margin Classifier

$$\max_{\boldsymbol{w},b} \gamma = \max_{\boldsymbol{w},b} \frac{y(\boldsymbol{w}^T \boldsymbol{x} + b)}{\|\boldsymbol{w}\|}$$

$$s.t., \gamma_i \geq \gamma$$

We know $y(\boldsymbol{w}^T \boldsymbol{x} + b)$ can be made arbitrarily large without changing the hyperplane, so we simply fix it at $y(\boldsymbol{w}^T \boldsymbol{x} + b) = 1$

Maximum Margin Classifier

$$\max_{\mathbf{w},b} \frac{y(\mathbf{w}^T\mathbf{x}+b)}{\|\mathbf{w}\|} \Rightarrow \max_{\mathbf{w},b} \frac{1}{\|\mathbf{w}\|} \Rightarrow \min_{\mathbf{w},b} \|\mathbf{w}\|$$

$$s.t., \gamma_i = \frac{y_i(\mathbf{w}^T\mathbf{x}_i+b)}{\|\mathbf{w}\|} \geq \gamma$$

$$\Downarrow$$

$$y_i(\mathbf{w}^T\mathbf{x}_i+b) \geq \gamma\|\mathbf{w}\| = 1$$

$$\Downarrow$$

$$y_i(\mathbf{w}^T\mathbf{x}_i+b) \geq 1$$

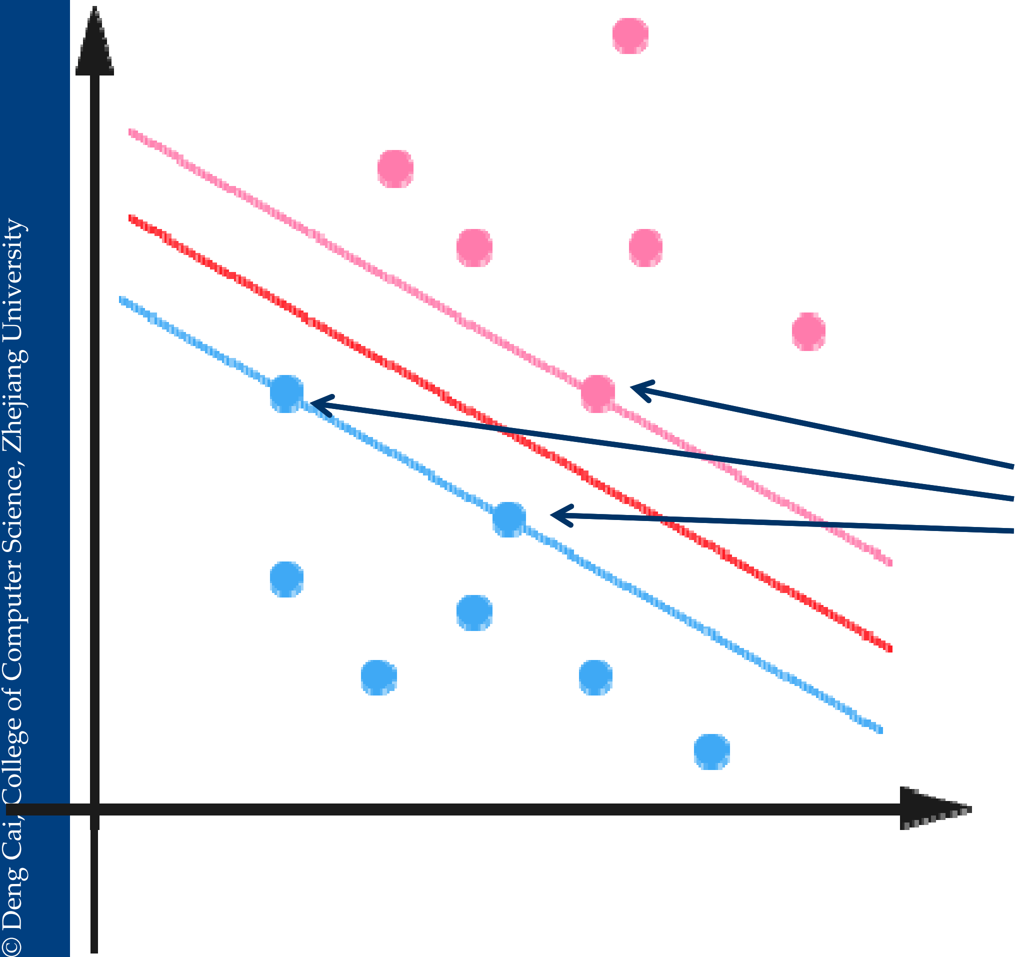
Maximum Margin Classifier

$$\min_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2$$

$$y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1$$

Square and a coefficient $\frac{1}{2}$ are added for the convenience of the derivation of optimization, and the minimizer of $\|\boldsymbol{w}\|$ and $\frac{1}{2}\|\boldsymbol{w}\|^2$ is obviously the same.

Support Vector Machine



Hyper plane of maximum margin is *support*ed by those points (vectors) on the margin. Those are called Support Vectors. Non-support vectors can move freely without affecting the position of the hyperplane as long as they don't exceed the margin.

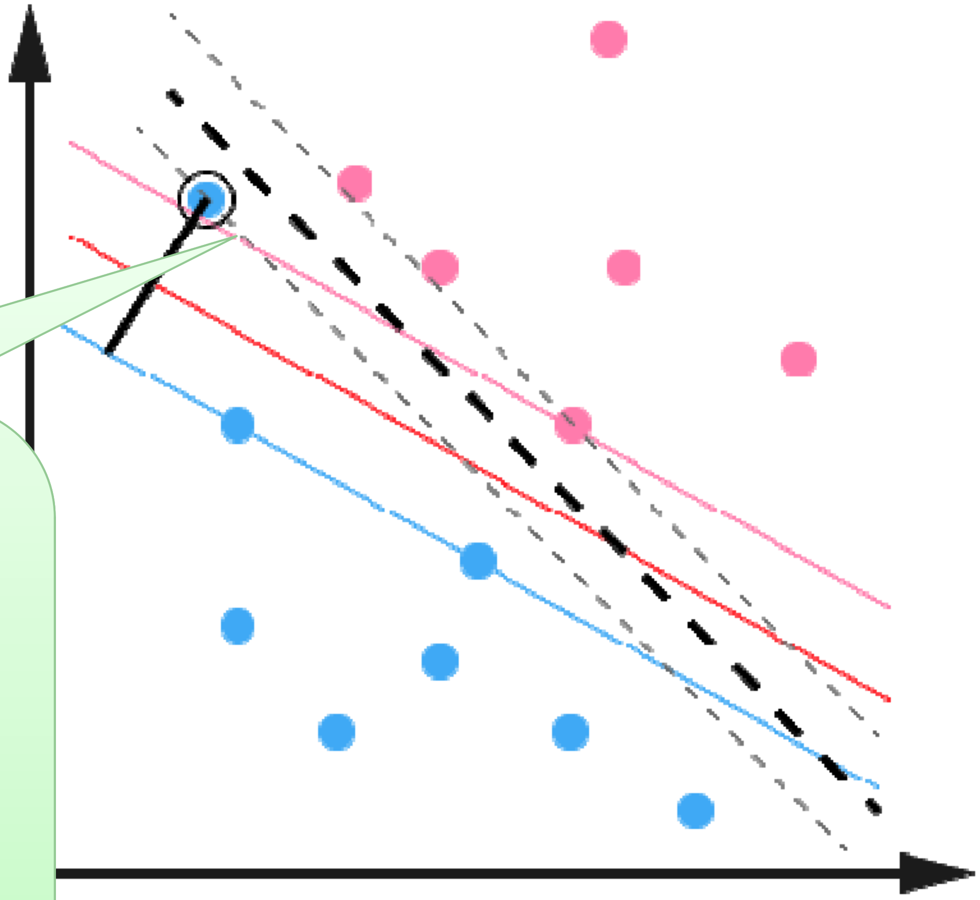
History of SVM

- SVM is related to statistical learning theory [3]
- SVM was first introduced in 1992 [1]
- SVM becomes popular because of its success in handwritten digit recognition
  - 1.1% test error rate for SVM. This is the same as the error rates of a carefully constructed neural network, LeNet 4.
    - See Section 5.11 in [2] or the discussion in [3] for details
- SVM is now regarded as an important example of "kernel methods", one of the key area in machine learning

[1] Bernhard E. Boser , Isabelle M. Guyon , Vladimir N. Vapnik, A Training Algorithm for Optimal Margin Classifiers. Proceedings of the Fifth Annual Workshop on Computational Learning Theory 5 144-152, Pittsburgh, 1992.
[2] L. Bottou *et al*. Comparison of classifier methods: a case study in handwritten digit recognition. Proceedings of the 12th IAPR International Conference on Pattern Recognition, vol. 2, pp. 77-82 1994.
[3] V. Vapnik. The Nature of Statistical Learning Theory. 2nd edition, Springer, 1999.

Weakness of the Original Model

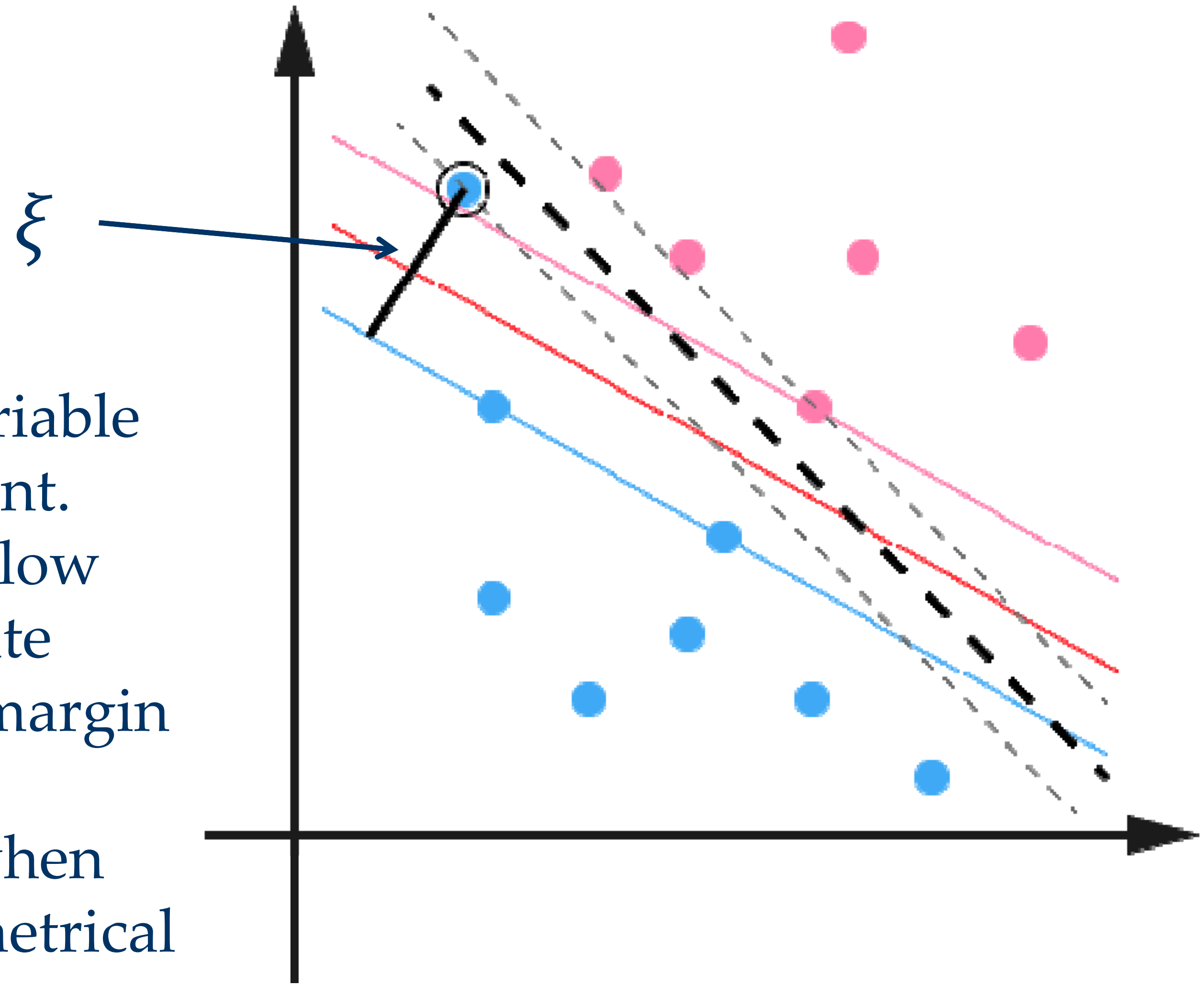
$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$y_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \geq 1$$



When an outlier appear, the optimal hyperplane may be pushed far away from its original/correct place. The resultant margin will also be smaller than before.

- Red Solid: the original hyperplane
- Dark dashed: the new hyperplane

Slack Variables



Assign a slack variable ξ to each data point. That means we allow the point to deviate from the correct margin by a distance of ξ (Actually $\|\boldsymbol{w}\|\xi$ when considering geometrical margin).

New Objective Function

Slack variables can't be arbitrarily large, we want to minimize the sum of all slack variables

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$y(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

New Objective Function

$$\min_{w,b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$y(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

- We would pay a cost of the objective function being increased by $C\ \xi_i$. The parameter C controls the relative weighting between the twin goals of making the $\|\boldsymbol{w}\|^2$ small (makes the margin large) and of ensuring that most examples have functional margin at least 1.

Software

Lots of SVM software:

- LibSVM (C++)
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- SVMLight (C)

Unconstrained Optimization Problem of SVM

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$y\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

$$\xi_i \geq 1 - y\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \qquad \xi_i = \max\left[1 - y\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right), 0\right]$$

$$\min_{\boldsymbol{w},b}\left\{\sum_{i=1}^{n} \underbrace{\max\left[1 - y\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right), 0\right]}_{\text{Loss function}} + \underbrace{\frac{1}{2C}\|\boldsymbol{w}\|^2}_{\text{Regularizer}}\right\}$$

$\ell(f) = \max[1 - yf, 0]$ Hinge loss

Linear regression: $E(\boldsymbol{a}) = \sum_{i\in I}\underbrace{\left(y_i - \boldsymbol{a}^T\boldsymbol{x}_i\right)^2}_{\text{Loss function}}$

$\ell(f) = (y - f)^2 = (1 - yf)^2$ Square loss

Logistic regression: $E(\boldsymbol{a}) = \sum_{i\in I}\underbrace{\log\left(1 + e^{-y_i\boldsymbol{a}^T\boldsymbol{x}_i}\right)}_{\text{Loss function}}$

$\ell(f) = \log\left(1 + e^{-yf}\right)$ Logistic loss

A General formulation of classifiers

$$\min_f \left\{ \sum_{i=1}^{n} \ell(f) + \lambda R(f) \right\}$$

Loss function

Regularizer

Square loss: $\ell(f) = (1 - yf)^2$ Ordinary regression

Logistic loss: $\ell(f) = \log\left(1 + e^{-yf}\right)$ Logistic regression

Hinge loss: $\ell(f) = \max[1 - yf, 0]$ SVM

L2-regularizer

L1-regularizer

Loss Function