Linear Classification and Support Vector Machine and Stochastic Gradient Descent and Multi-class Classification

Prof. Mingkui Tan

SCUT Machine Intelligence Laboratory (SMIL)

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 Stochastic Gradient Descent

5 Mini-Batch Stochastic Gradient Descent

6 Multi-class Classification

7 Dual Problem For SVM

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 Stochastic Gradient Descent

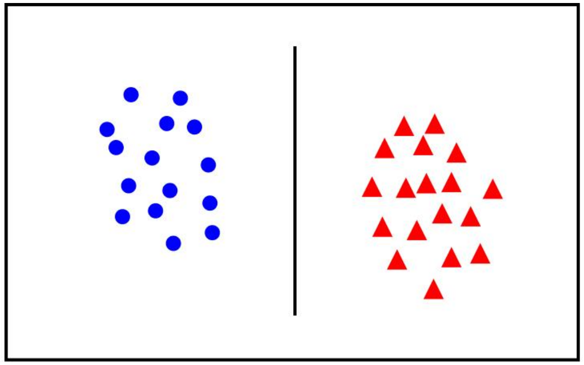
5 Mini-Batch Stochastic Gradient Descent

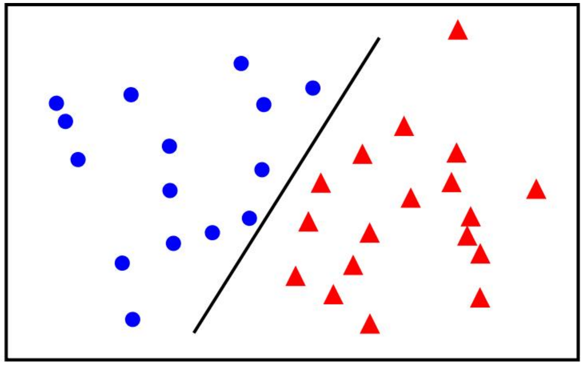
6 Multi-class Classification

7 Dual Problem For SVM

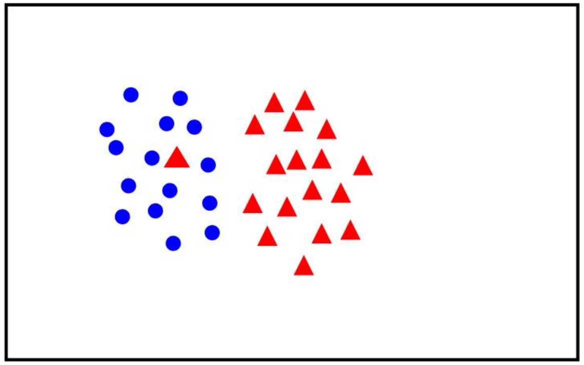
Linear Separability

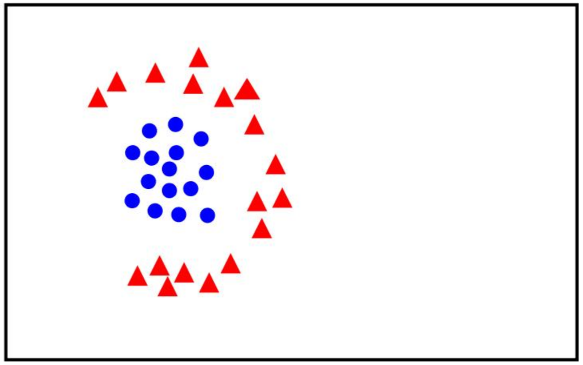
Linearly separable





Not linearly separable



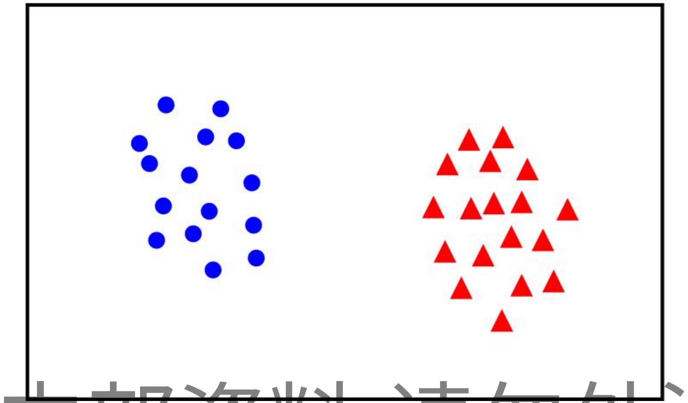


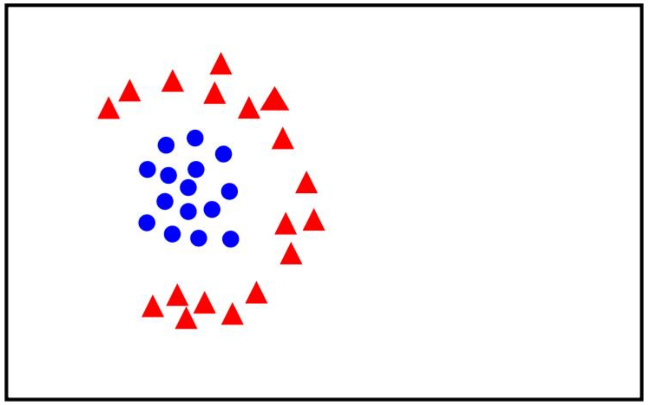
Binary Classification

- Given training data $\{(\mathbf{x}_i,\ y_i)\}_{i=1}^{n}$ for i = 1…n, with $\mathbf{x}_i \in \mathbb{R}^m$ and $y_i \in \{-1,\ 1\}$, learn a classifier $f(x)$ such that

$$f(\mathbf{x}_i)\begin{cases} \geq 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$

i.e. $y_i f(\mathbf{x}_i) > 0$ for a correct classification

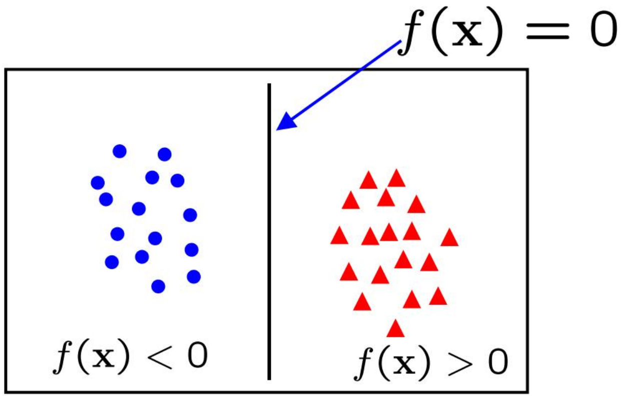




Linear Classifiers: 2D Example

A linear classifier has the form:

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$$

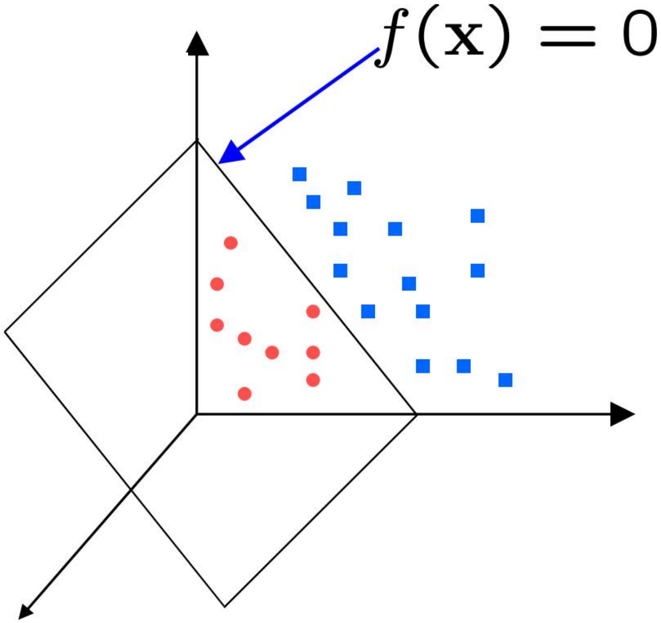


- In 2-D, the discriminant is a line
- **w** is the normal to the line, and *b* is the bias
- **w** is known as the weight vector

Linear Classifiers: 3D Example

A linear classifier has the form:

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$$



- In 3-D the discriminant is a plane, and in m-D it is a hyperplane

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

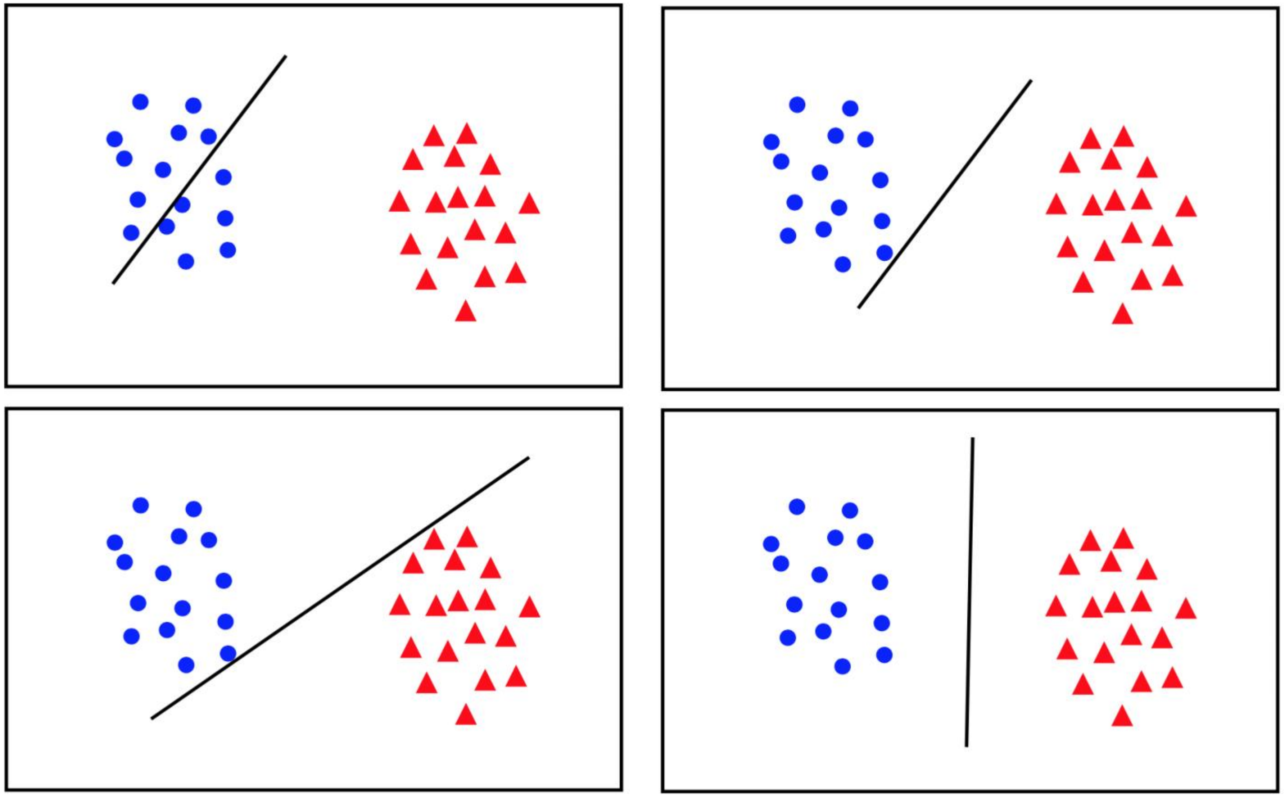
4 Stochastic Gradient Descent

5 Mini-Batch Stochastic Gradient Descent

6 Multi-class Classification

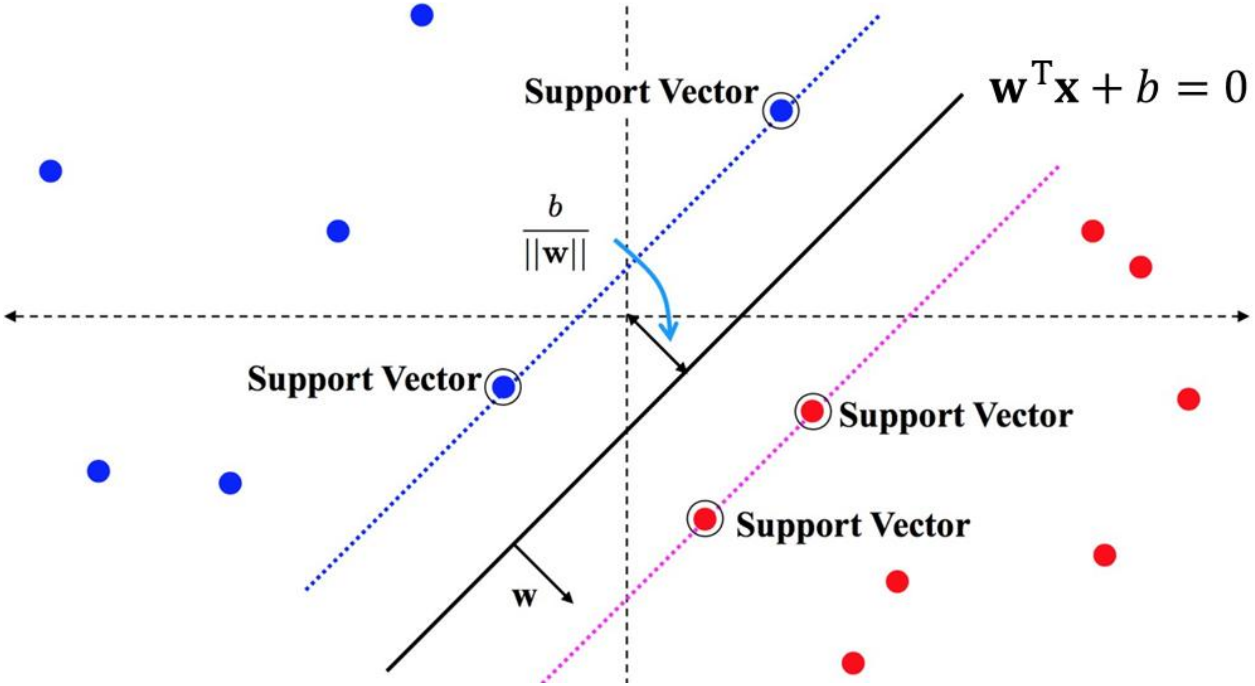
7 Dual Problem For SVM

What’s a Good Decision Boundary?



- Maximum margin solution: most stable under perturbations of the inputs

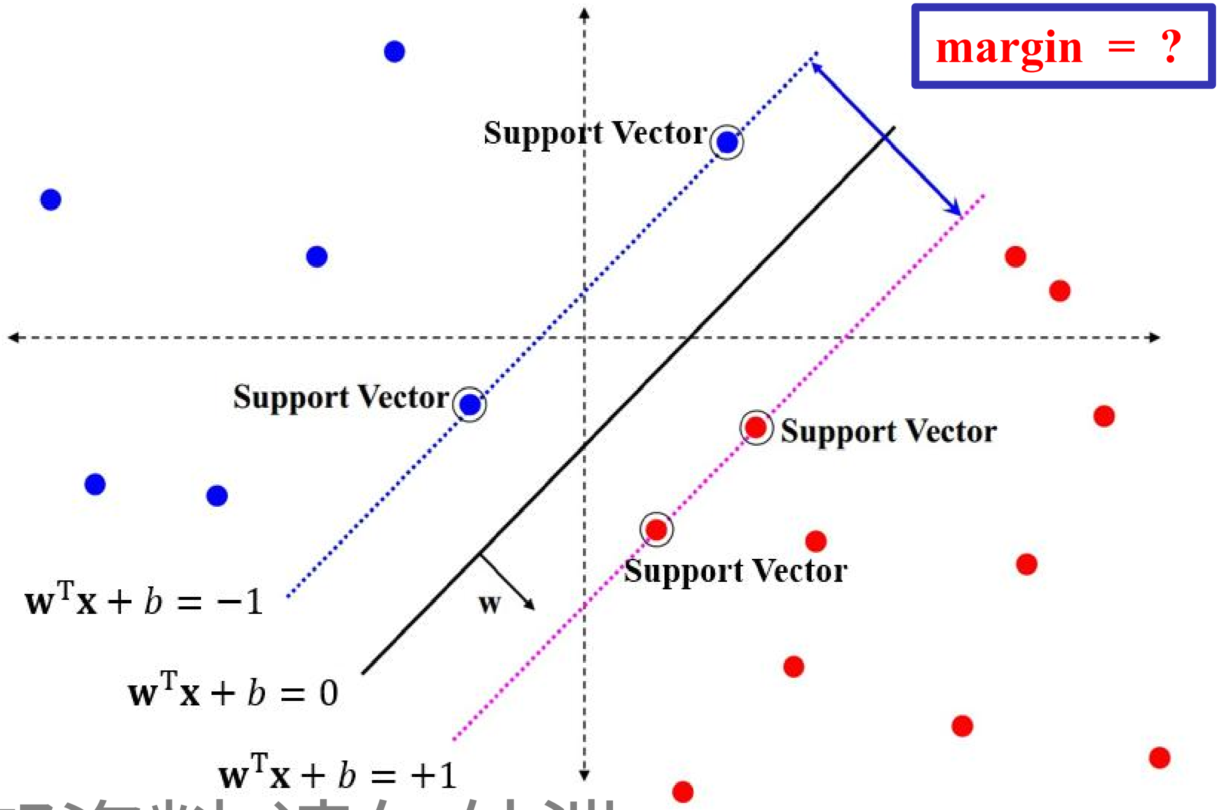
Max-margin Methods



- Select two parallel hyperplanes that separate the two classes of data and let the distance between them as large as possible
- The region bounded by these two hyperplanes is called the "margin"

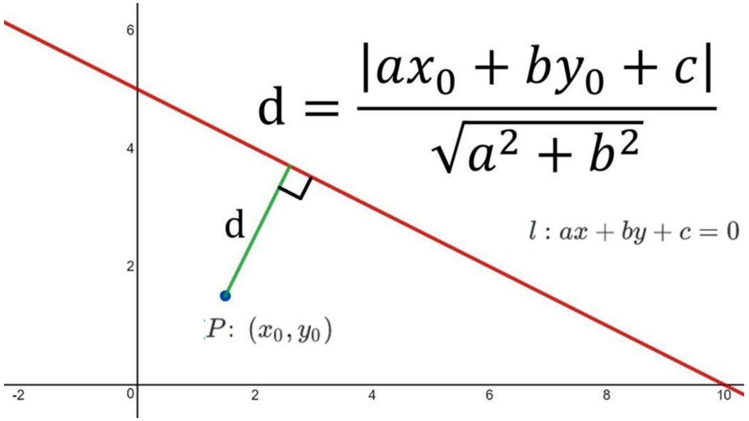
SVM-sketch Derivation

- Choose normalization such that $\mathbf{w}^{\mathrm{T}}\mathbf{x}_{+} + b =+ 1$ and $\mathbf{w}^{\mathrm{T}}\mathbf{x}_{-} + b =- 1$ for the positive and negative support vectors respectively
- Then how to calculate the margin ?



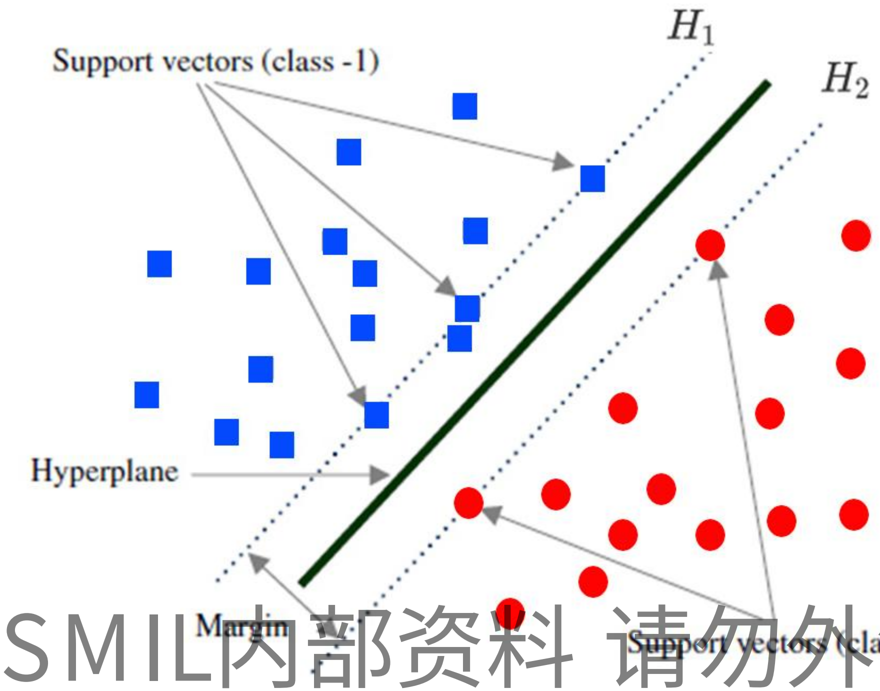
margin = ?
Support Vector
Support Vector
Support Vector
Support Vector
$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = -1$
w
$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$
$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = +1$

Geometric Margin



$$d = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

- Distance from point to line in **two-dimensional plane**

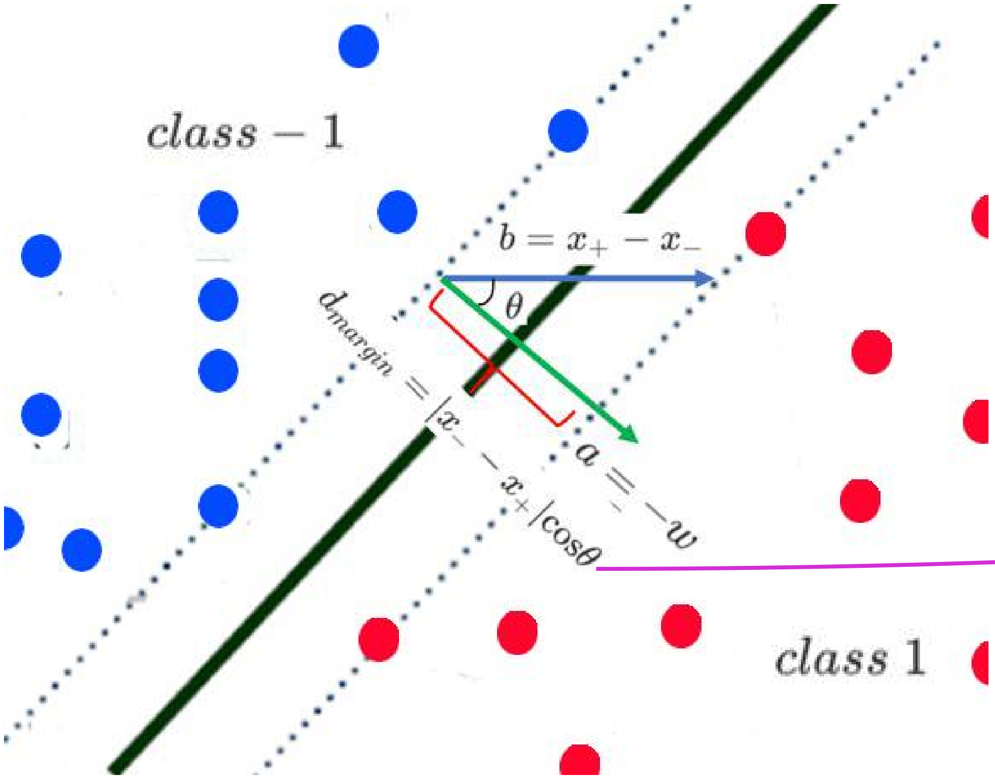


- Distance from point to line in **high-dimensional space**

$$\frac{|\mathbf{w}^{\mathbf{T}}\mathbf{x} + \mathbf{b}|}{||\mathbf{w}||},$$

$$\text{where } ||\mathrm{w}|| = \sqrt{\mathrm{w}_1^2 + \cdots + \mathrm{w}_n^2}$$

Functional Margin



- Hyperplane H0: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_+ + b = 0$
- Hyperplane H1: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_+ + b$ += 1
- Hyperplane H2: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_- + b =- 1$

$$\boxed{d_{margin} = |\mathbf{x}_+ - \mathbf{x}_-|cos\theta}$$

H1−H2 = 1-(-1) = $\mathbf{w}^{\mathrm{T}}(\mathbf{x}_+ - \mathbf{x}_-)$ =$|\mathbf{w}||\mathbf{x}_+ - \mathbf{x}_-|cos\theta$ =$||\mathbf{w}||d_{margin}$

$$\boxed{\boldsymbol{d_{margin} = \frac{2}{||w||}}}$$

Basic Support Vector Machine

- SVM can be considered as an optimization problem to maximize margin.

- So the problem can be formulated as an optimization:

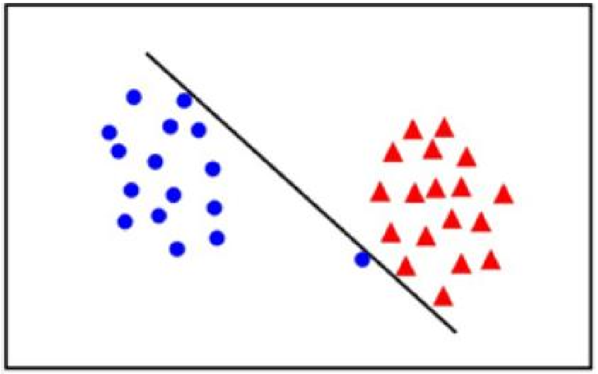
$$\max_{w,b} \frac{2}{||\mathbf{w}||} \quad s.t. \quad \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \begin{cases} \geq 1 & y_i += 1 \\ \leq -1 & y_i =- 1 \end{cases}$$

- Or equivalently:

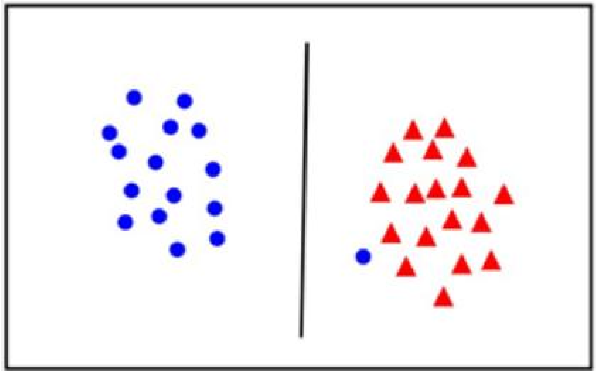
$$\min_{\mathbf{w},b} \frac{||\mathbf{w}||^2}{2}$$

$$s.t. \quad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1, \quad i = 1,2,3,\ldots,n$$

Linear Separability Again: What is the Outlier?



The point can be linearly separated but the margin is very narrow

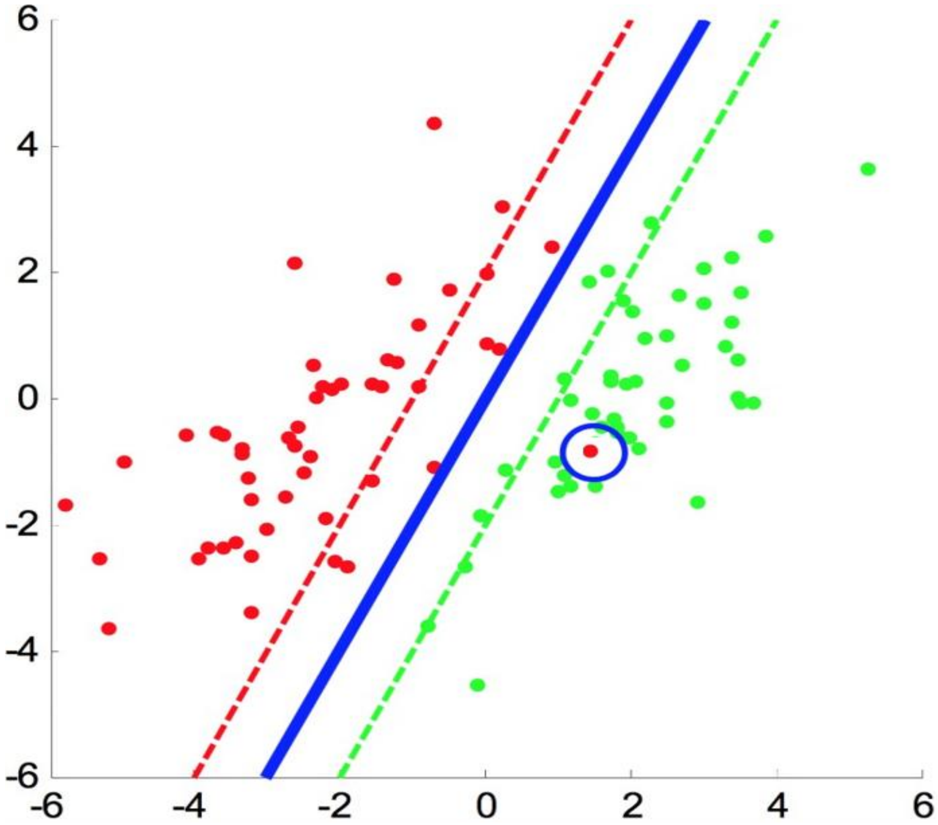


The large margin solution is better, even though one sample point cannot satisfy the constraint

- In general, there exists a trade-off between the margin and the number of mistakes on the training data

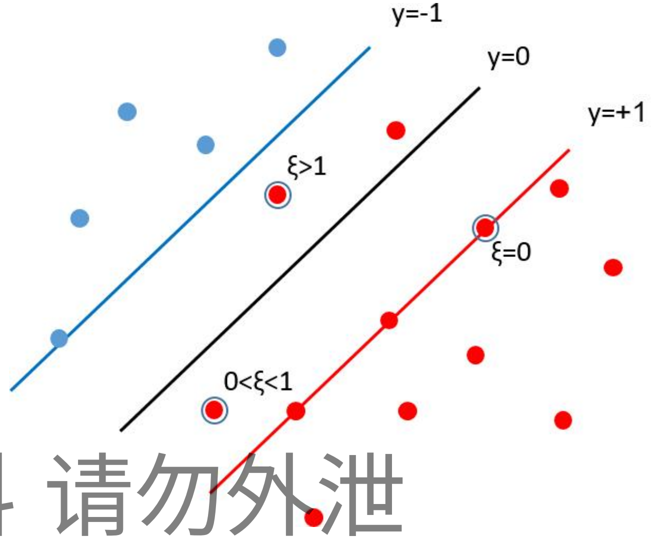
Linear Separability Again: What is the Outlier?

- Moreover, training data may not be linearly separable!



Relaxed Formulation

- Introduce slack variable $\xi_i \geq 0$ for each i, which represents how much example i is on the wrong side of margin boundary
  - $\xi_i = 0$ means the basic formulation of SVM works well
  - $0 < \xi_i < 1$ means it is correctly classified, but with a smaller margin than $\frac{2}{||\mathbf{w}||}$
  - $\xi_i \geq 1$ means it is incorrectly classified



Soft Margin Formulation

- The previous optimization problem:

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

$$s.t.\ \ y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1, \qquad i = 1,2,3,\ldots,n$$

- Now, the optimization problem becomes:

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

$$s.t. \qquad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i,\ \ \forall\xi_i \geq 0,\ \ i = 1,2,\ldots,n$$

where C is a hyperparameter:

-small C makes constraints easy to be ignored

-large C makes constraints hard to be ignored

Soft Margin Formulation

- As can be seen from above:

$$\xi_i \geq 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b),\ \forall \xi_i \geq 0,\ i = 1, 2, \ldots, n$$

- What does ξ_i value?

  - if $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1$, then $\xi_i = 0$ (Correctly classified)
  - if $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 1$, then $\xi_i \geq 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)$

  Recall that we should minimize $\frac{C}{n}\sum_{i=1}^{n}\xi_i$, so we have:

  if $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 1$, then $\xi_i = 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)$

- So we can rewrite ξ_i as:

$$\xi_i = \max\left(0, 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)\right)$$

Hinge Loss

- Hinge loss:

$$\xi_i = \max\left(0,\ 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)\right)$$

- The optimization problem becomes:

$$\min_{\mathbf{w},b} \frac{||\mathbf{w}||^2}{2} + \frac{C}{n}\sum_{i=1}^{n} \max(0,1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b))$$

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 Stochastic Gradient Descent

5 Mini-Batch Stochastic Gradient Descent

6 Multi-class Classification

7 Dual Problem For SVM

Gradient Descent

- Gradient descent minimum optimization problem:

$$\min_{\mathbf{w},b} L(\mathbf{w},b) = \frac{||\mathbf{w}||^2}{2} + \frac{C}{n}\sum_{i=1}^{n} \max\left(0,\ 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)\right)$$

- To minimize a loss function $L(\mathbf{w},b)$, use the iterative update:

$$\mathbf{w} = \mathbf{w} - \eta\nabla_{\mathbf{w}}L(\mathbf{w},b)$$

$$b = b - \eta\nabla_{b}L(\mathbf{w},b)$$

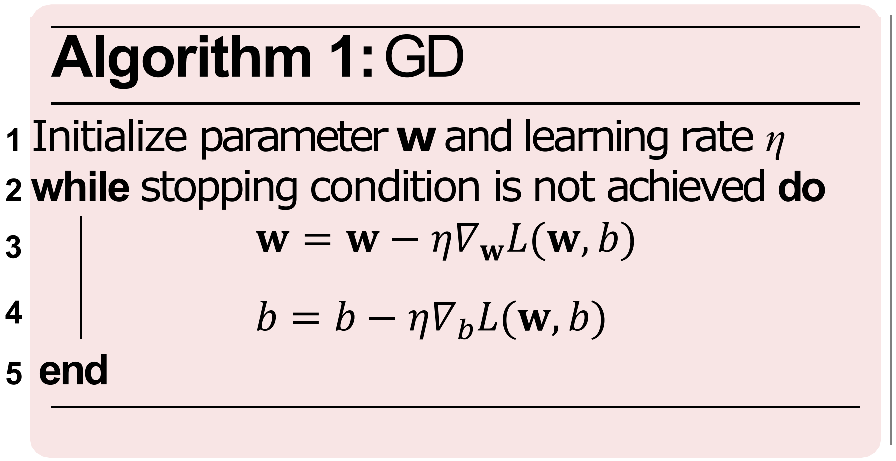
where η is the learning rate

Gradient Descent

- Let $g_{\mathbf{w}}(\mathbf{x}_i) = \frac{\partial \xi_i}{\partial \mathbf{w}}, \; g_b(\mathbf{x}_i) = \frac{\partial \xi_i}{\partial b}$

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b) = \mathbf{w} + \frac{C}{n} \sum_{i=1}^{n} g_{\mathbf{w}}(\mathbf{x}_i)$$

$$\nabla_b L(\mathbf{w}, b) = \frac{C}{n} \sum_{i=1}^{n} g_b(\mathbf{x}_i)$$



Algorithm 1: GD

```
Initialize parameter **w** and learning rate $\eta$
**while** stopping condition is not achieved **do**
    $\mathbf{w} = \mathbf{w} - \eta \nabla_{\mathbf{w}} L(\mathbf{w}, b)$
    $b = b - \eta \nabla_b L(\mathbf{w}, b)$
**end**
```

Gradient Computation

- Gradient descent minimum optimization problem:

$$\min_{\mathbf{w},b} L(\mathbf{w},b) = \frac{||\mathbf{W}||^2}{2} + \frac{C}{n}\sum_{i=1}^{n} \max(0,\ 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b))$$

- Gradient computation :

$$\nabla L = \begin{bmatrix} \nabla_{\mathbf{w}} L(\mathbf{w},b) \\ \nabla_b L(\mathbf{w},b) \end{bmatrix}$$

Gradient Computation

- The hinge loss is $\xi_i = \max\left(0,\ 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)\right)$
- Let $g_{\mathbf{w}}(\mathbf{x}_i) = \frac{\partial \xi_i}{\partial \mathbf{w}}$
- If $1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0$:

$$g_{\mathbf{w}}(\mathbf{x}_i) = \frac{\partial)-y_i)\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b((}{\partial \mathbf{w}}$$

$$=- \frac{\partial(y_i\mathbf{w}^{\mathrm{T}}\mathbf{x}_i)}{\partial \mathbf{w}}$$

$$=- y_i\mathbf{x}_i$$

- If $1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) > 0$:

$$g_{\mathbf{w}}(\mathbf{x}_i) = 0$$

Gradient Computation

- So we have:

$$g_{\mathbf{w}}(\mathbf{x}_i) = \begin{cases} -y_i\mathbf{x}_i & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0 \\ 0 & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 0 \end{cases}$$

- Let $g_b(\mathbf{x}_i) = \frac{\partial \xi_i}{\partial b}$

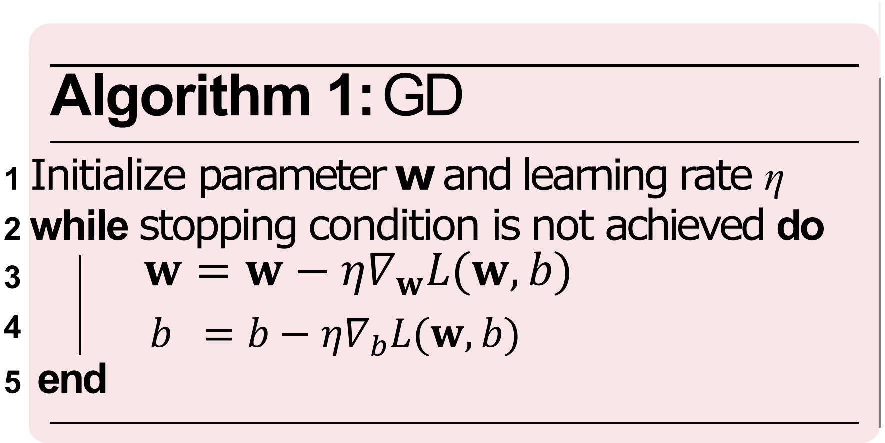
$$g_b(\mathbf{x}_i) = \begin{cases} -y_i & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0 \\ 0 & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 0 \end{cases}$$

Gradient Descent

- Gradient descent is a batch algorithm that uses all examples

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b) = \mathbf{w} + \frac{C}{n} \sum_{i=1}^{n} g_{\mathbf{w}}(\mathbf{x}_i)$$

$$\nabla_{b} L(\mathbf{w}, b) = \frac{C}{n} \sum_{i=1}^{n} g_{b}(\mathbf{x}_i)$$



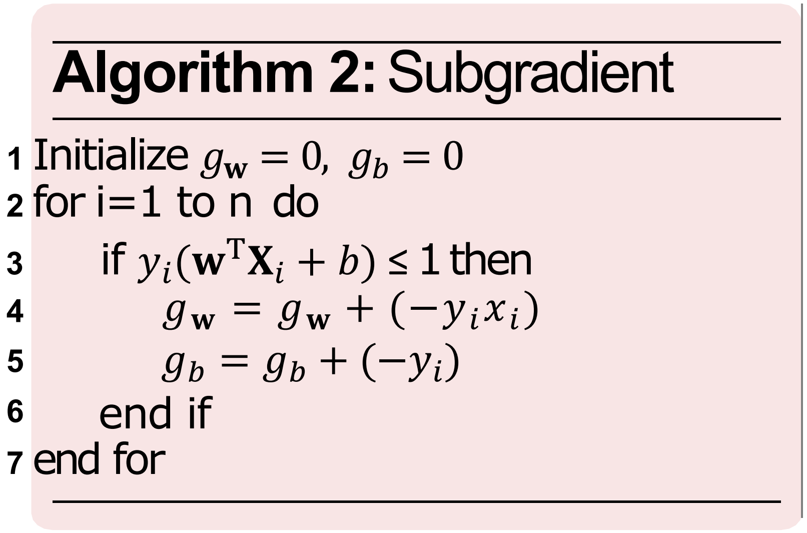
```
Algorithm 1: GD
Initialize parameter w and learning rate $\eta$
while stopping condition is not achieved do
    $\mathbf{w} = \mathbf{w} - \eta \nabla_{\mathbf{w}} L(\mathbf{w}, b)$
    $b \;= b - \eta \nabla_{b} L(\mathbf{w}, b)$
end
```

Gradient Descent

- The algorithm for calculating the subgradient is as follows:

$$g_{\mathbf{w}}(\mathbf{x}_i) = \begin{cases} -y_i\mathbf{x}_i & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0 \\ 0 & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 0 \end{cases}$$

$$g_b(\mathbf{x}_i) = \begin{cases} -y_i & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0 \\ 0 & 1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) < 0 \end{cases}$$



Algorithm 2: Subgradient

```
Initialize $g_{\mathbf{w}} = 0,\ g_b = 0$
for i=1 to n do
    if $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \leq 1$ then
        $g_{\mathbf{w}} = g_{\mathbf{w}} + (-y_i x_i)$
        $g_b = g_b + (-y_i)$
    end if
end for
```

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 **Stochastic Gradient Descent**

5 Mini-Batch Stochastic Gradient Descent

6 Multi-class Classification

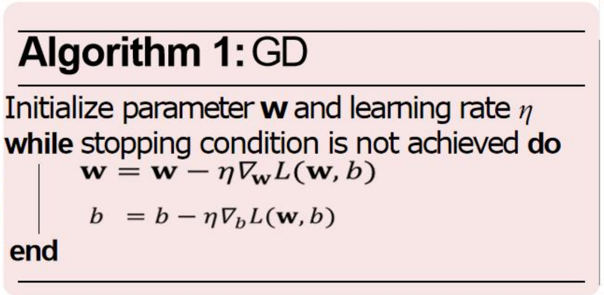
7 Dual Problem For SVM

Stochastic Optimization Motivation

- Information is redundant amongst samples
- Sufficient samples mean we can afford noisy updates
- Never-ending stream means we should not wait for all data
- Tracking non-stationary data means the target is changing

Stochastic Gradient Descent

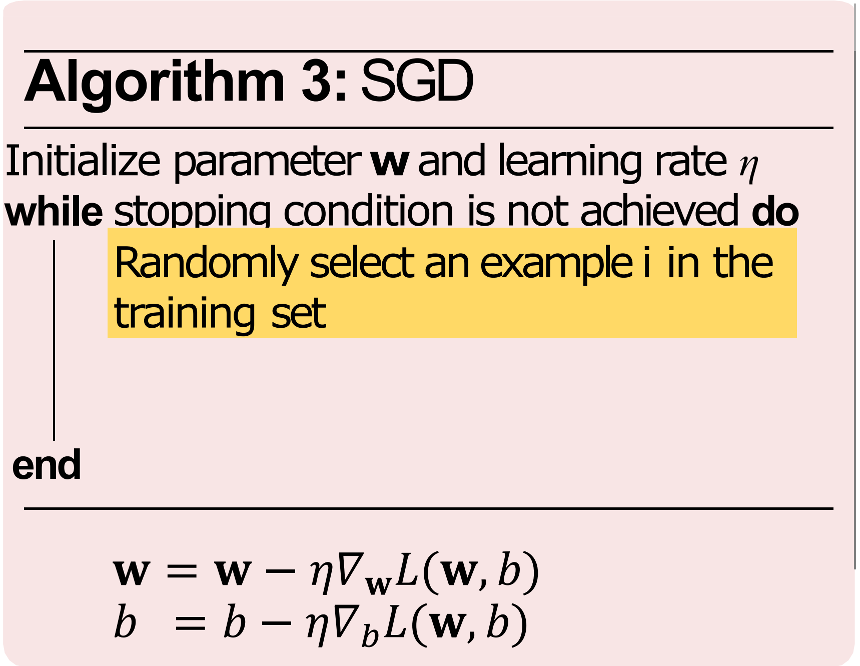
- Stochastic Gradient Descent works similar as GD that uses the loss of a single training sample to approximate the average loss



```
Algorithm 1: GD
Initialize parameter w and learning rate $\eta$
while stopping condition is not achieved do
    $\mathbf{w} = \mathbf{w} - \eta \nabla_{\mathbf{w}} L(\mathbf{w}, b)$
    $b \ \ = b - \eta \nabla_b L(\mathbf{w}, b)$
end
```

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b) = \mathbf{w} + \frac{C}{n} \sum_{i=1}^{n} g_{\mathbf{w}}(\mathbf{x}_i)$$

$$\nabla_b L(\mathbf{w}, b) = \frac{C}{n} \sum_{i=1}^{n} g_b(\mathbf{x}_i)$$



```
Algorithm 3: SGD
Initialize parameter w and learning rate $\eta$
while stopping condition is not achieved do
    Randomly select an example i in the training set
    $\mathbf{w} = \mathbf{w} - \eta \nabla_{\mathbf{w}} L(\mathbf{w}, b)$
    $b \ \ = b - \eta \nabla_b L(\mathbf{w}, b)$
end
```

n —> 1

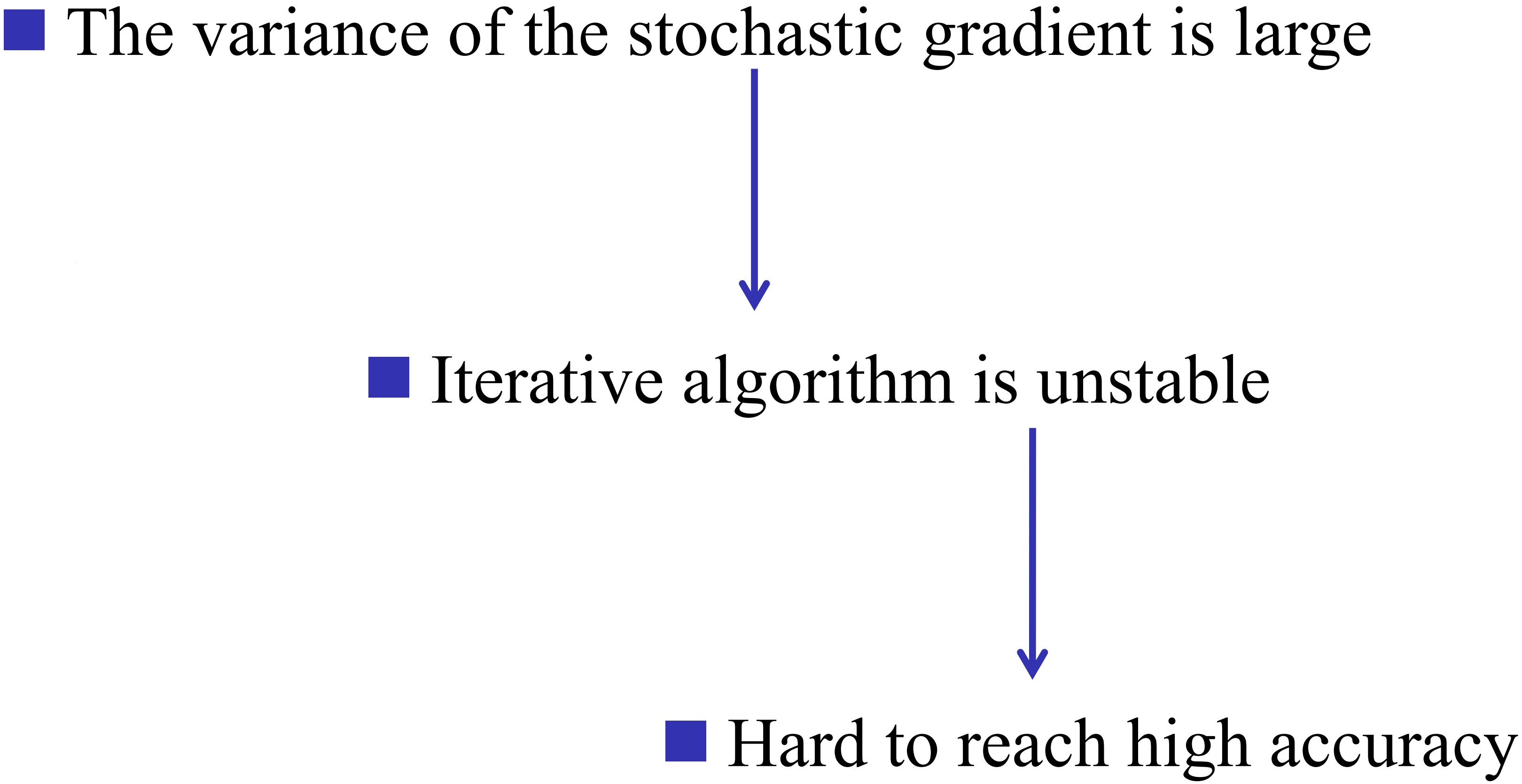
$$\nabla_{\mathbf{w}} L(\mathbf{w}, b) = \mathbf{w} + C g_{\mathbf{w}}(\mathbf{x}_i)$$

$$\nabla_b L(\mathbf{w}, b) = C g_b(\mathbf{x}_i)$$

Benefits of SGD

- Gradient is easy to calculate (instantaneous)
- Less prone to local minima
- Small memory to use
- Get to a reasonable solution quickly
- Can be used for more complex models

Limitation of SGD



- The variance of the stochastic gradient is large
- Iterative algorithm is unstable
- Hard to reach high accuracy

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 Stochastic Gradient Descent

5 **Mini-Batch Stochastic Gradient Descent**

6 Multi-class Classification

7 Dual Problem For SVM

Minibatch Stochastic Gradient Descent

- Like the single random sample, the full gradient is approximated via an unbiased noisy estimate
- Rather than use a single point, randomly choose a subset S_k ,and optimization problem is as follows:

$$\min_{\mathbf{w},b} L := \frac{||\mathbf{w}||^2}{2} + \frac{C}{|S_k|} \sum_{i \in S_k} \max(0,\ 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$$

- The size of S_k, denoted by $|S_k|$, is much smaller than the original data size n, and $|S_k|$ can be $2^3,\ 2^4$ ...

Minibatch Stochastic Gradient Descent

- MSGD works identically to SGD, except that we use more than one training example to make each estimate of the gradient

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b) = \mathbf{w} + \frac{C}{|S_k|} \sum_{i \in S_k} g_{\mathbf{w}}(X_i)$$

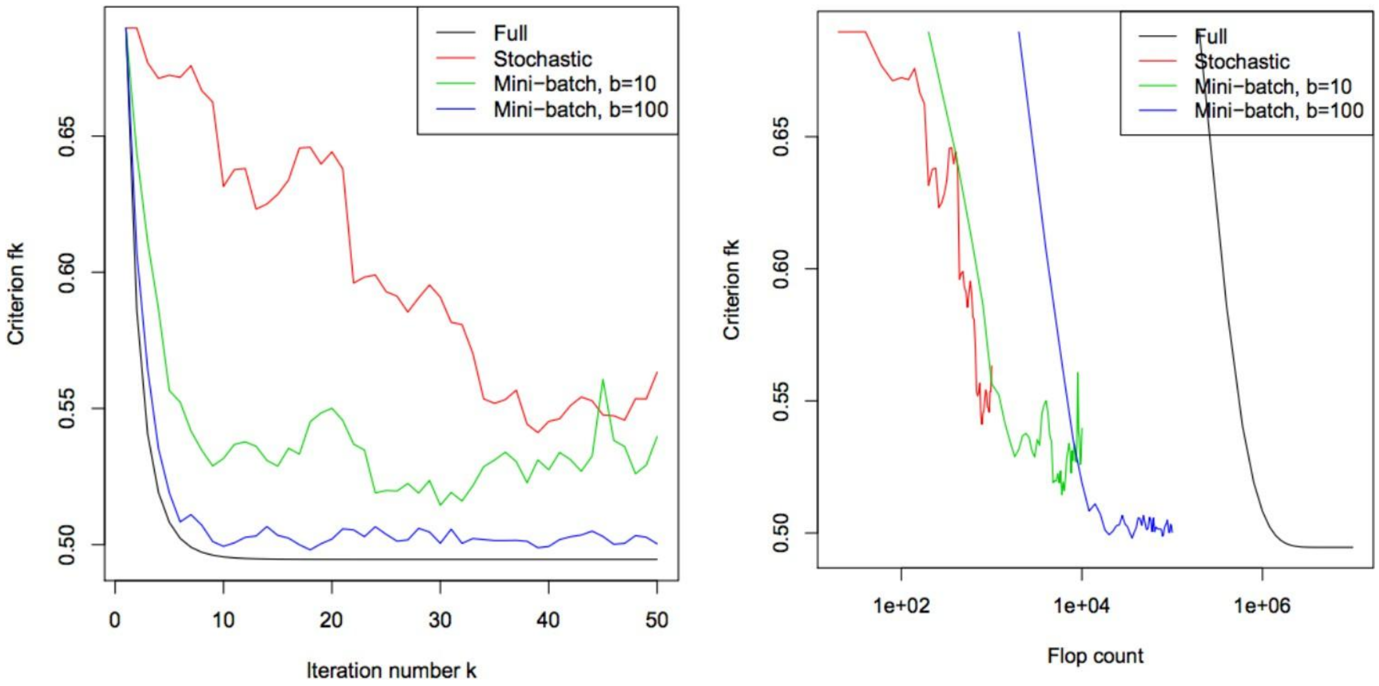
$$\nabla_b L(\mathbf{w}, b) = \frac{C}{|S_k|} \sum\nolimits_{i \in S_k} g_b(X_i)$$

Algorithm 4: MSGD

```
Initialize parameter **w** and learning rate $\eta$
**while** stopping condition is not achieved **do**
    Randomly select $|S_k|$ examples in the training set
    $\mathbf{w} = \mathbf{w} - \eta \nabla_{\mathbf{w}} L(\mathbf{w}, b)$
    $b \;= b - \eta \nabla_b L(\mathbf{w}, b)$
**end**
```

Example

- Larger mini-batch size takes more computation time but less iterations



Importance of Learning Rate

- Learning rate has a large impact on convergence

  Too small → too slow

  Too large → oscillate and may even diverge

- Should learning rate be fixed or adaptive?

- Is convergence necessary?

  Non-stationary: convergence may not be required

  Stationary: learning rate should decrease with time

SGD Recommendations

Randomly shuffle training examples

- Although theory says you should randomly pick examples, it is easier to pass through your training set sequentially
- Shuffling before each iteration eliminates the effect of order

Monitor both training cost and validation errors

- Set aside samples for validation set
- Compute the objective on the training set and validation set (expensive but better than overfitting or wasting computation)

SGD Recommendations

Check gradient using finite differences

- Incorrect computation can yield erratic and slow algorithm
- Verify your code by slightly perturbing the parameter and inspecting differences between the two gradients

Leverage sparsity of the training examples

- For very high-dimensional vectors with few non-zero coefficients, you only need to update the weight corresponding to non-zero pattern in **x**

SGD Recommendations

Experiment with the learning rates using small sample of training set

- SGD convergence learning rates are independent from sample size
- Use traditional optimization algorithms as a reference point

Use learning rates of the form $\eta_t = \eta_{t-1}(1 + \eta_{t-1}\lambda t)^{-1}$

- Allow you to start from reasonable learning rates determined by testing on a small sample
- Work well in most situations if the initial point is slightly smaller than best value observed in training sample

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 Stochastic Gradient Descent

5 Mini-Batch Stochastic Gradient Descent

6 Multi-class Classification

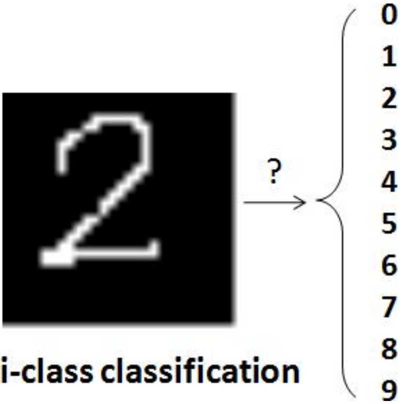
7 Dual Problem For SVM

Three Common Classification Problems

- Binary classification
- Multi-class classification
- Multi-label classification

Is it the number 2?(Y/N)

Binary classification



Multi-class classification

Multi-label classification

Multi-class Classification

Multi-class classification is the common classification problem, which classifies instances into one of the more than two classes.

Dataset

- MNIST
- Cifar-10 and Cifar-100
- ImageNet
- …

Multi-class Classification

Three general strategies：

- Convert to binary classification
- Extend from binary classification
- Hierarchical classification

Convert to Binary Classification

The strategies reduce the problem of multi-class classification to multiple binary classification problems.

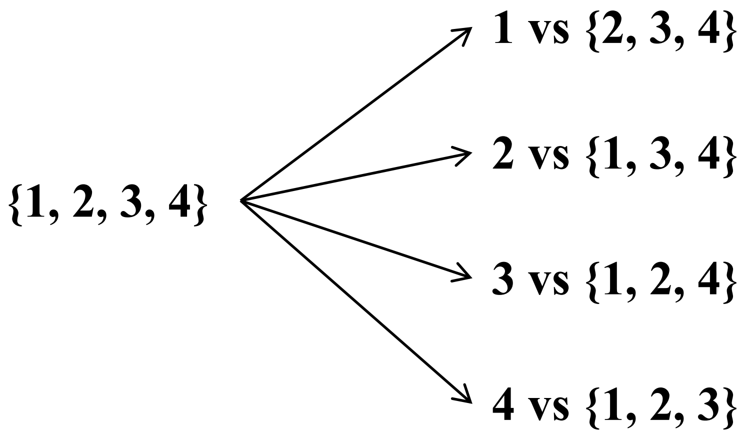
- One-vs.-rest
- One-vs.-one
- Decision Directed Acyclic Graph, DDAG

One-vs.-rest Method

Training:

- For each class

  Train a binary classifier with the samples of that class as positive samples and others as negatives



{1, 2, 3, 4}

1 vs {2, 3, 4}

2 vs {1, 3, 4}

3 vs {1, 2, 4}

4 vs {1, 2, 3}

One-vs.-rest

One-vs.-rest Method

Prediction:

- For each binary classifier
  - Produce a real-valued confidence score
- Predict the label with the highest confidence score

$$\widehat{y} = \arg\max_{k \in \{1 \ldots k\}} f_k(x)$$

One-vs.-rest Method

Advantage vs Disadvantages

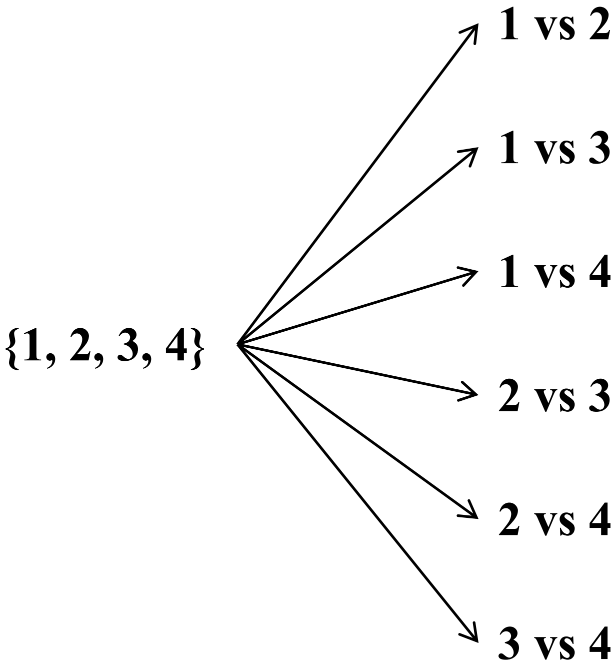
- Advantage
  - Train K binary classifiers for a K-way multi-class problem
- Disadvantages
  - The distributions of the binary classifications are unbalanced (The set of negatives is much larger than the set of positives)
  - The scale of the confidence values may differ between the binary classifiers

One-vs.-one Method

Training:

- For each pair of classes

  Train a binary classifier to discriminate between them.



{1, 2, 3, 4}

1 vs 2

1 vs 3

1 vs 4

2 vs 3

2 vs 4

3 vs 4

One-vs.-one

One-vs.-one Method

Prediction:

- For each binary classifier
  - Contrast the two categories and do a voting

    A=B=C=D=0

    A vs B-classifier: if A win, A=A+1; otherwise, B=B+1;

    A vs C-classifier: if A win, A=A+1; otherwise, C=C+1;

    …

    C vs D-classifier: if C win, C=C+1; otherwise, D=D+1;
- Predict the label with the maximum number of votes wins

$$\hat{y} = \max(A, B, C, D)$$

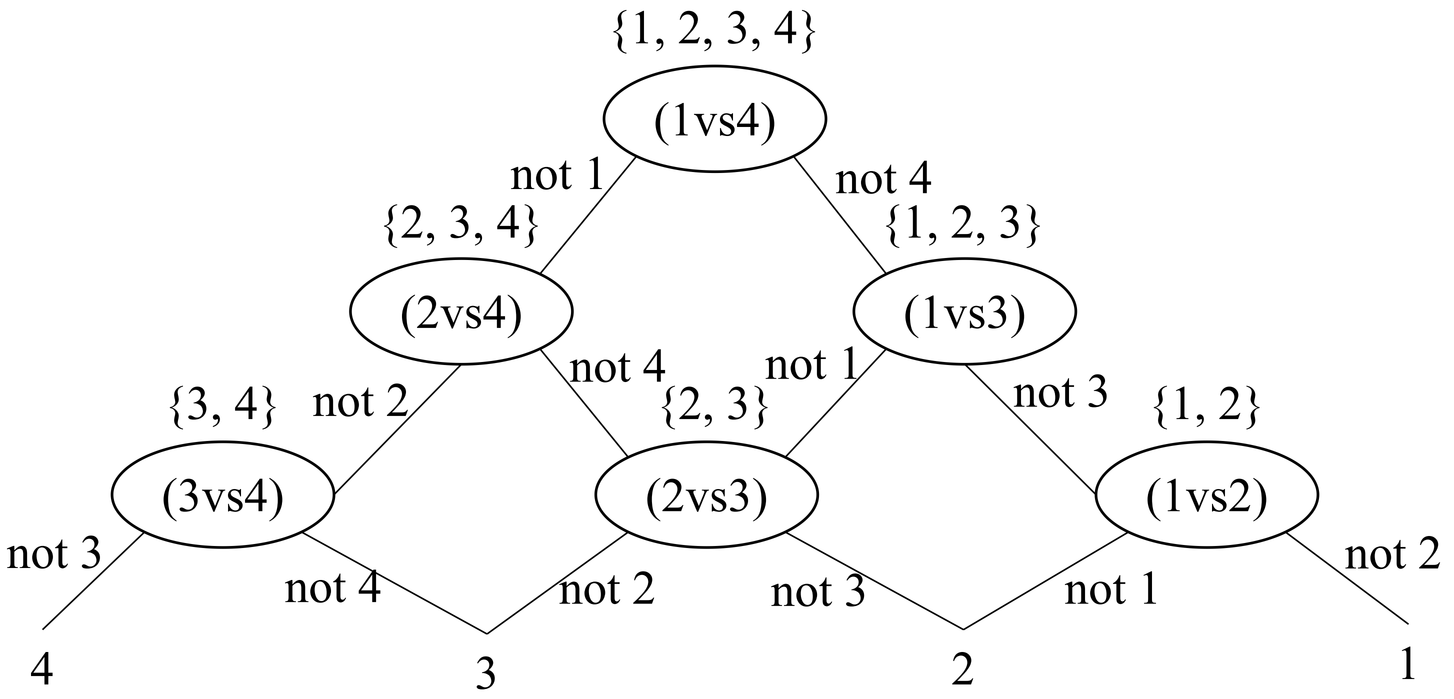
One-vs.-one Method

Advantage vs Disadvantages

- Advantage
  - The distributions of the binary classification are balanced
- Disadvantages
  - Train $K(K-1)/2$ binary classifiers for a K−� � � multi-class problem, which has high computed complexity
  - Suffer from ambiguities when receive the same number of votes

Decision Directed Acyclic Graph

- Compared to One-vs.-one method, it uses a rooted binary directed acyclic graph which has internal nodes and leaves



Decision Directed Acyclic Graph

Decision Directed Acyclic Graph

Predict:

- Start at the root node
- Before reaching a leaf node:
  - Evaluate the binary decision function
  - Move to either left or right depending on the output value

Compared to One-vs.-one method

- The correlation between each of the binary classifications brings the cost of the prediction down

Extend from Binary Classification

- The strategy extends the existing binary classifiers to solve multi-class classification problems
  - K-nearest neighbors

K-nearest Neighbors

- Calculate the distances

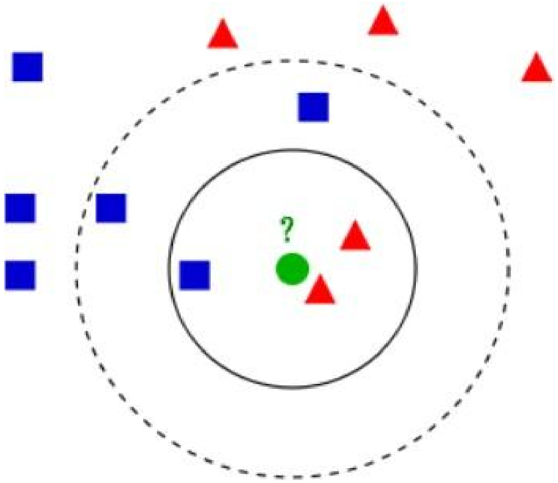
  Calculate the distances between the test object and each object in the training set

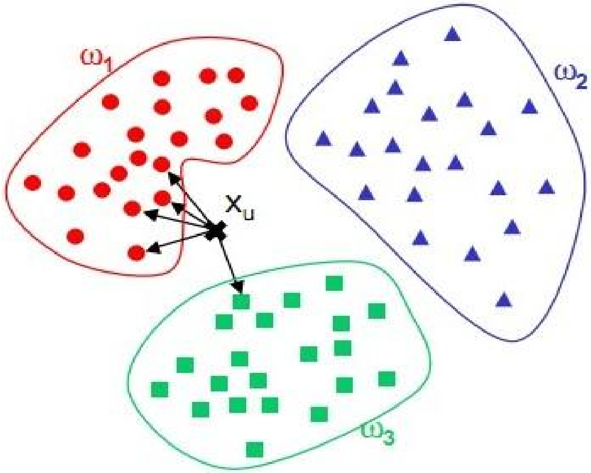
- Find the neighbors

  Get the K-nearest training objects as neighbors

- Vote on labels

  Classify the test object based on the most frequent class of the neighbors





K-nearest Neighbors

Advantages vs Disadvantages

- Advantages
  - The method is a non-parametric classification algorithm
  - The algorithm can naturally handle binary and multi-class classification
- Disadvantages
  - The computational and memory requirements are high
  - Finding good representations and distance measures between objects is hard

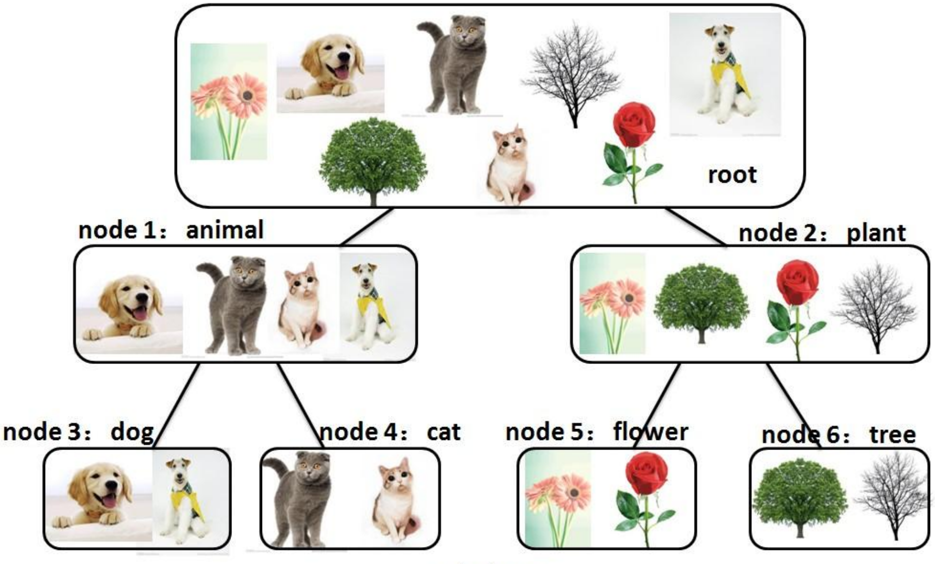
Hierarchical Classification

- The strategy tackles the multi-class classification problem by dividing the output space into a tree
  - Label tree

Label Tree

Train:

- Before the leaf nodes contain only a single class
  - Each parent node is divided into a number of clusters, one for each child node
- At each node, a simple classifier is trained to discriminate between the different child class clusters



Label tree

Label Tree

Predict:

- Start from the root node
  - Travel to a leaf node which is associated with a label

Advantage vs Disadvantage

- Advantage
  - The tree method brings the cost of the prediction down
- Disadvantage
  - Depended on a good clustering method.

Q & A?

Contents

1 Linear Classification

2 Support Vector Machine

3 Gradient Descent

4 Stochastic Gradient Descent

5 Mini-Batch Stochastic Gradient Descent

6 Multi-class Classification

7 Dual Problem For SVM

Dual Problem for SVM

- An optimization problem can be considered in two ways, primal problem and dual problem
  - for primal problem of basic SVM:

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

$$s.t. \qquad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1, \qquad\qquad i = 1,2,\dots,n$$

  - its Lagrange function is:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)\right) \qquad (1)$$

Dual Problem for SVM

- its Lagrange "dual function" is:

$$\boldsymbol{D}(\boldsymbol{\alpha}) = \boldsymbol{inf}\, L(w, b, \alpha)$$

Dual function gives the lower bound of the optimal value of primal problem

- dual problem: the best lower bound dual function can get

$$\max_{\alpha} D(\alpha)$$

Dual Problem for SVM

- setting partial derivative of D with respect to w equal to 0:

$$\nabla_{w} D = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0$$

$$\rightarrow \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \qquad (2)$$

- setting partial derivative of $\mathcal{L}$ with respect to b equal to 0:

$$\nabla_{b} D = \sum_{i=1}^{n} \alpha_i y_i = 0 \qquad (3)$$

Dual Problem for SVM

- Put (2), (3) into (1), we can get:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)\right)$$

$$= \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \mathbf{w}^{\mathrm{T}}\alpha_i y_i \mathbf{x}_i - \sum_{i=1}^{n} \alpha_i y_i$$

$$= \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \sum_{i=1}^{n} \alpha_i \quad - \mathbf{w}^{\mathrm{T}}\mathbf{w} - 0$$

$$= \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\left[\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right]^{\mathrm{T}} \left[\sum_{j=1}^{n} \alpha_i y_j \mathbf{x}_i\right]$$

$$= \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^{\mathrm{T}} X_j$$

Dual Problem for SVM

- Last, we can get the dual problem:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j$$

$$s.t. \qquad \sum_{i=1}^{n} \alpha_i y_i = 0,$$

$$\alpha_i \geq 0, \qquad i = 1,2,\ldots,n$$

Thank You