统计机器学习

（小班研讨）

主讲教师： 刘峤

第4章 支持向量机与核方法

Support Vector Machines and Kernel Methods

感知机

感知机

- **概述**
  - 由美国学者Rosenblatt在1957年首次提出
  - 学习算法是Rosenblatt在1958年提出的
  - IEEE设立IEEE Frank Rosenblatt Award（2004）
    - https://www.ieee.org/about/awards/tfas/rosenblatt.html
  - 包含一个突触权值可调的神经元
  - 属于前向神经网络类型
  - 只能区分线性可分的模式

单层感知机

- 单层感知机：Perceptron Learning Algorithm (PLA)
  - 是具有单层处理单元的神经网络：模拟神经元接受环境信息，并由神经冲动（激活函数）进行信息传递
  - 用于求解输入空间的二分类问题：求解分类超平面，属于线性分类模型
  - 设输入向量 $\mathbf{x} \in R^n$，类别标记 $y \in \{-1, +1\}$，则感知机模型表示为：

$$f(\mathbf{x}) = sign(\mathbf{w} \cdot \mathbf{x} + b)$$

- 思考：$\mathbf{w} \cdot \mathbf{x} + b = 0$ 的几何意义是什么?
- 思考：$\mathbf{w}$ 的几何意义是什么?
- 思考：感知机与逻辑斯蒂回归的联系与区别在哪里?

单层感知机的求解

- 思考：若发生误分类的情况，误分类点到超平面的距离是?

$$d = -\frac{1}{\|\mathbf{w}\|_2} y^i (\mathbf{w} \cdot \mathbf{x}^i + b)$$

- 因此可以将损失函数定义为：

$$\mathcal{L}(\mathbf{w}, b) = -\sum_{\mathbf{x}^i \in D'} y^i (\mathbf{w} \cdot \mathbf{x}^i + b)$$

- 其中：D' 表示模型当前误分类的点的集合
- 思考：如何最小化该损失函数?

$$\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}, b) = -\sum_{\mathbf{x}^i \in D'} y^i \mathbf{x}^i \qquad \nabla_b \mathcal{L}(\mathbf{w}, b) = -\sum_{\mathbf{x}^i \in D'} y^i$$

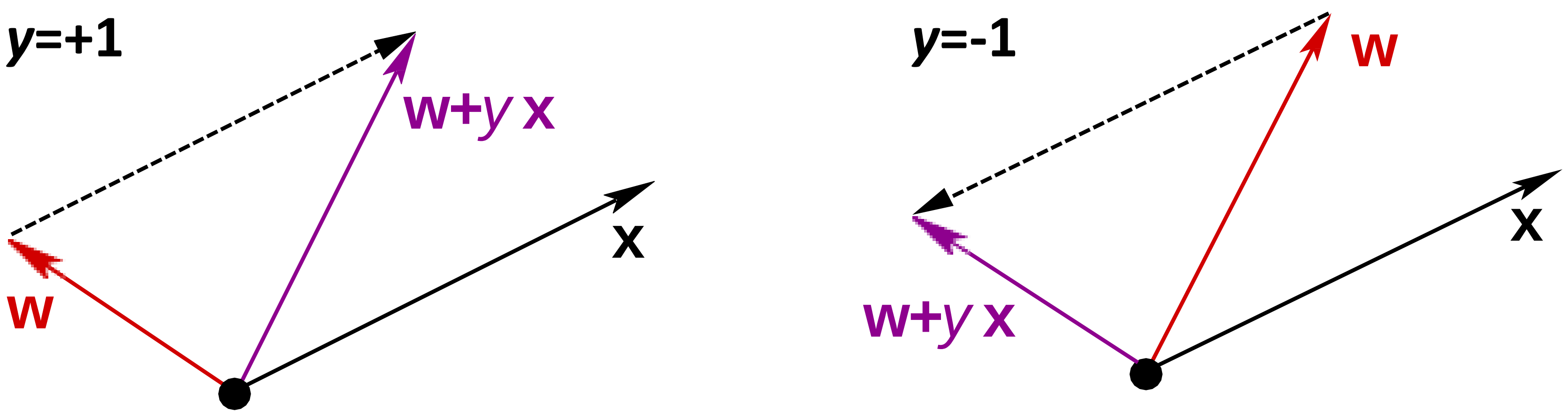
$$\mathbf{w} = \mathbf{w} + \eta \sum_{\mathbf{x}^i \in D'} y^i \mathbf{x}^i \qquad b = b + \eta \sum_{\mathbf{x}^i \in D'} y^i$$

Batch Gradient Descent

单层感知机的求解

- 随机梯度下降（Stochastic Gradient Descent)

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \eta y^i \mathbf{x}^i \qquad b_{t+1} = b_t + \eta y^i$$



Practical Implementation of PLA

- Start from some w_0 (say, 0), and 'correct' its mistakes on D

- For t = 0, 1, . . .

  1. find the next mistake of w_t called (x_t, y_t)

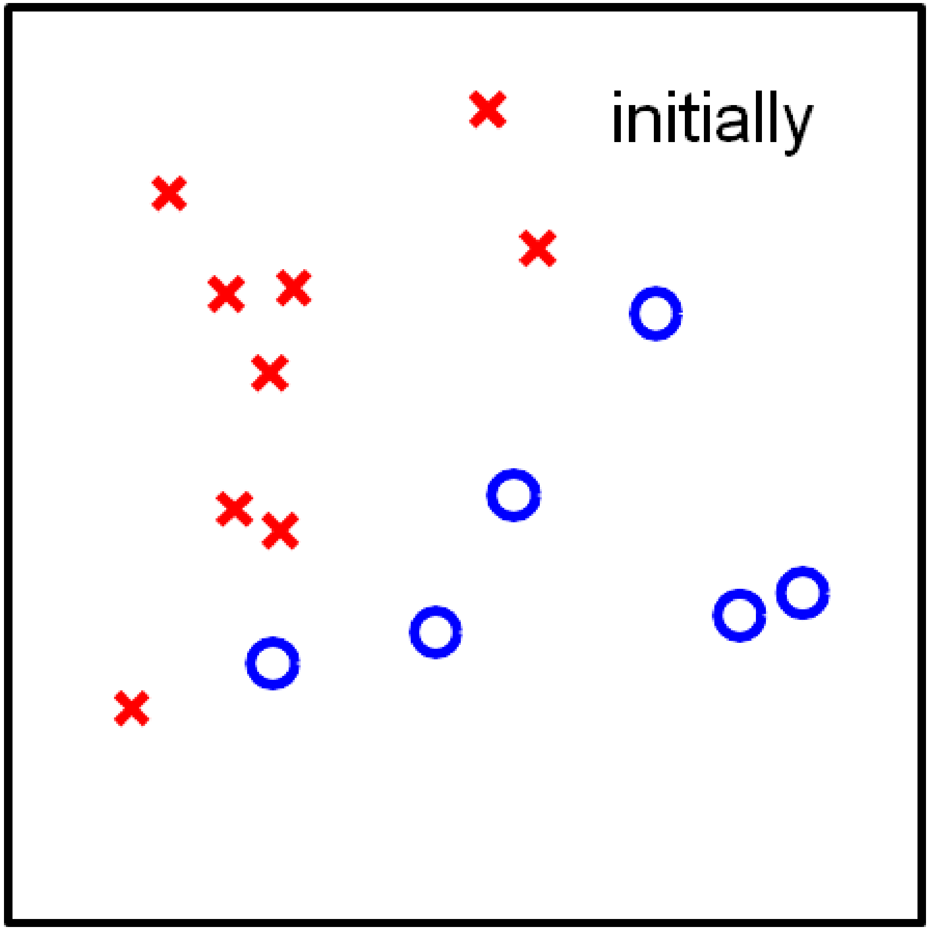
$$sign(\mathbf{w}_t \cdot \mathbf{x}_t + b) \neq y_t$$

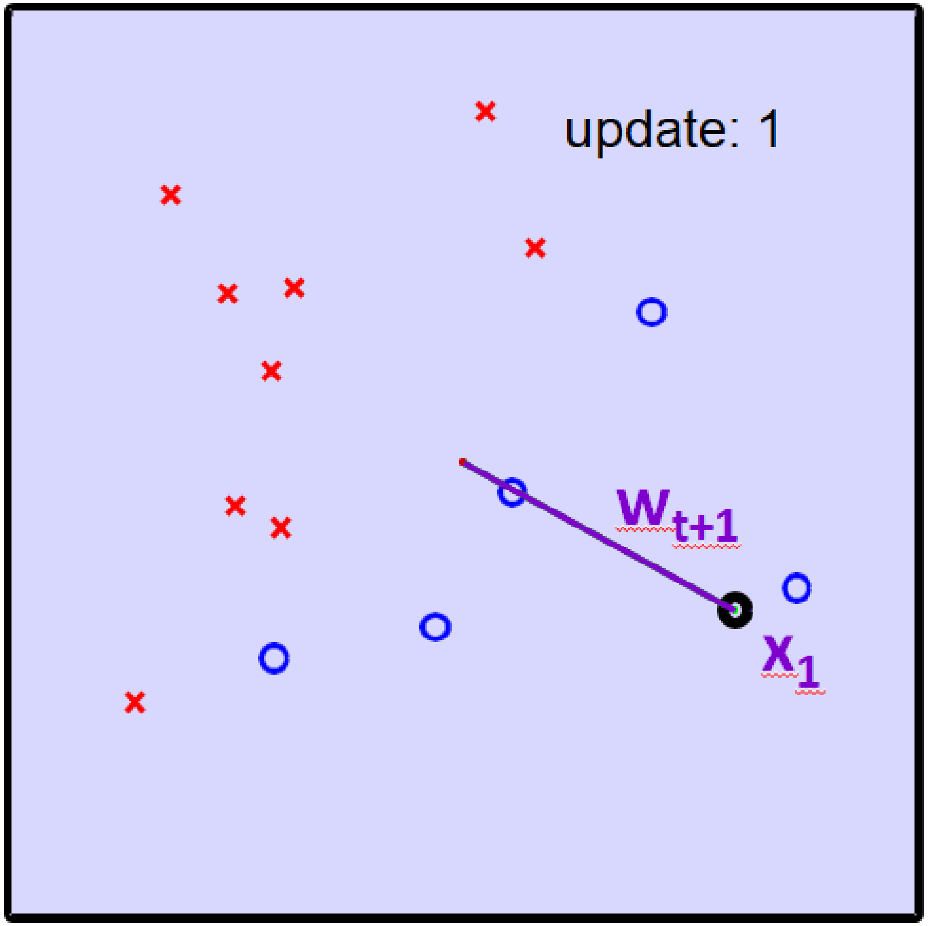
  2. correct the mistake by

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t$$

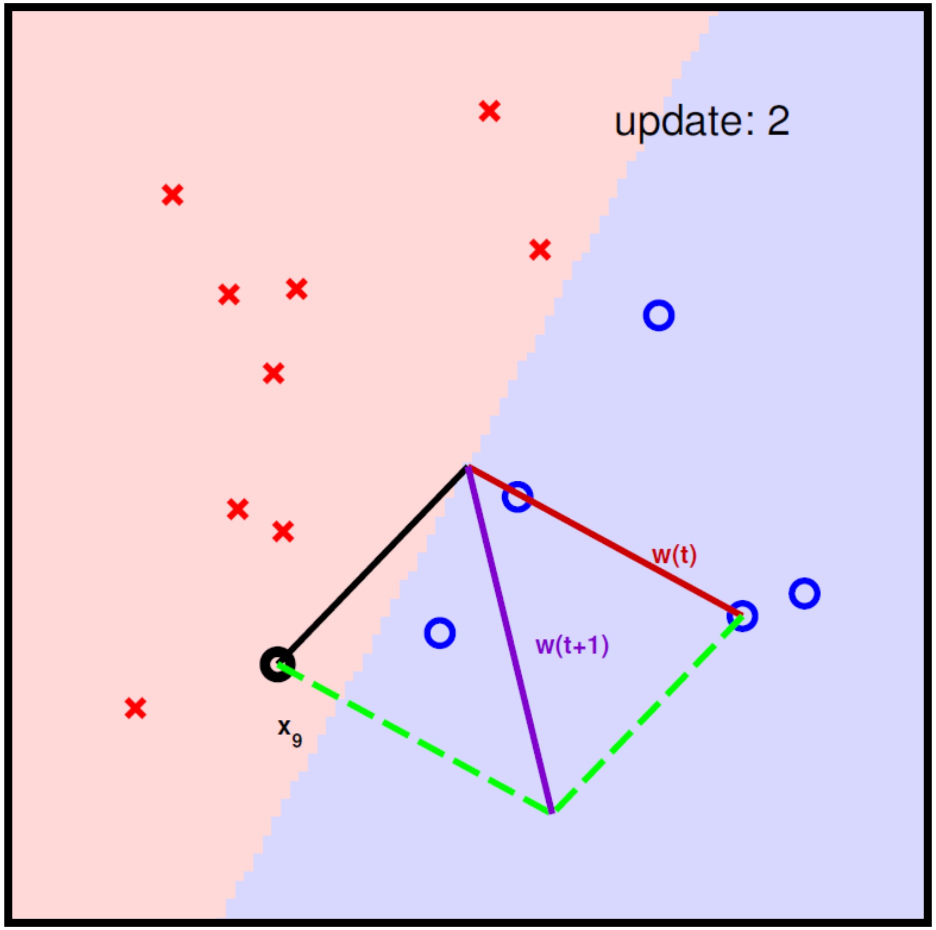
- … until a full cycle of not encountering mistakes

Seeing is Believing

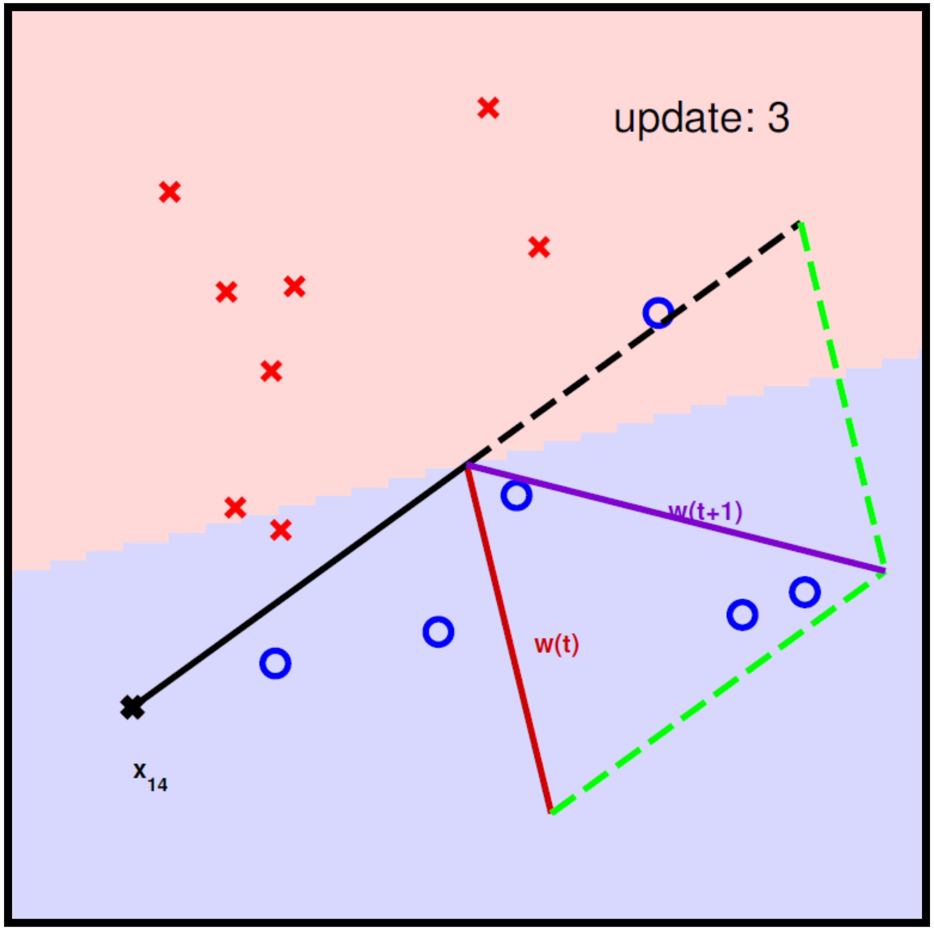




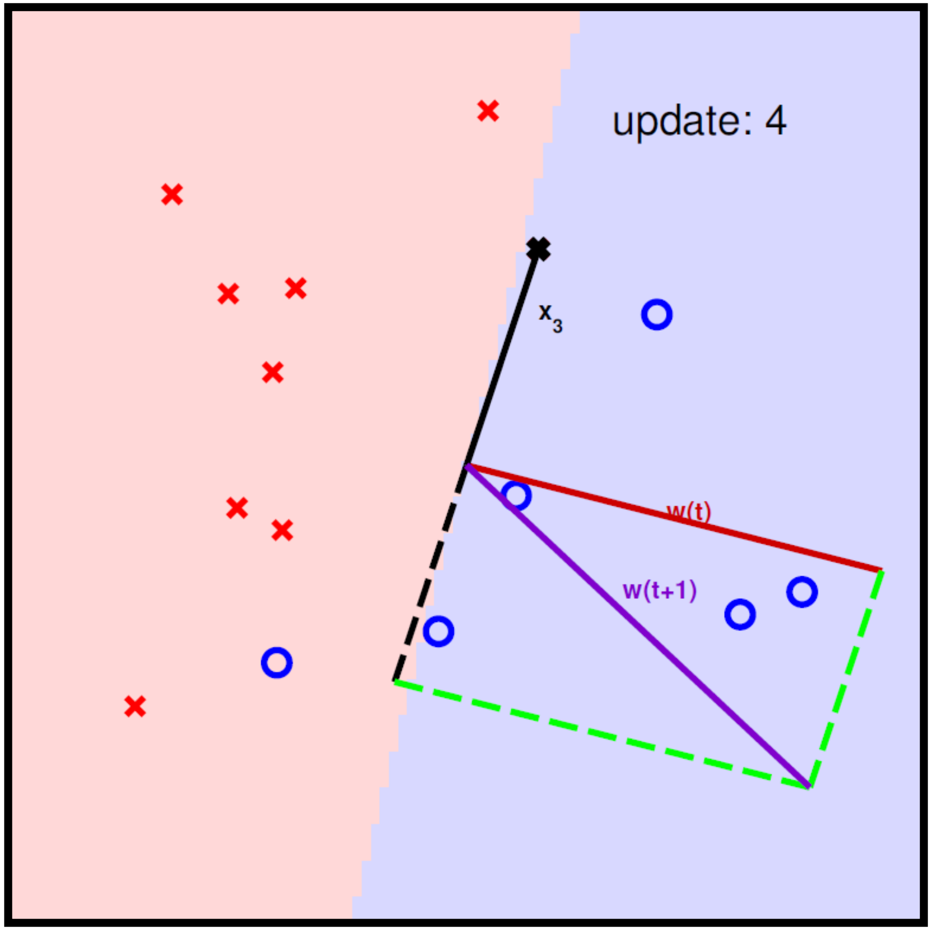
Seeing is Believing



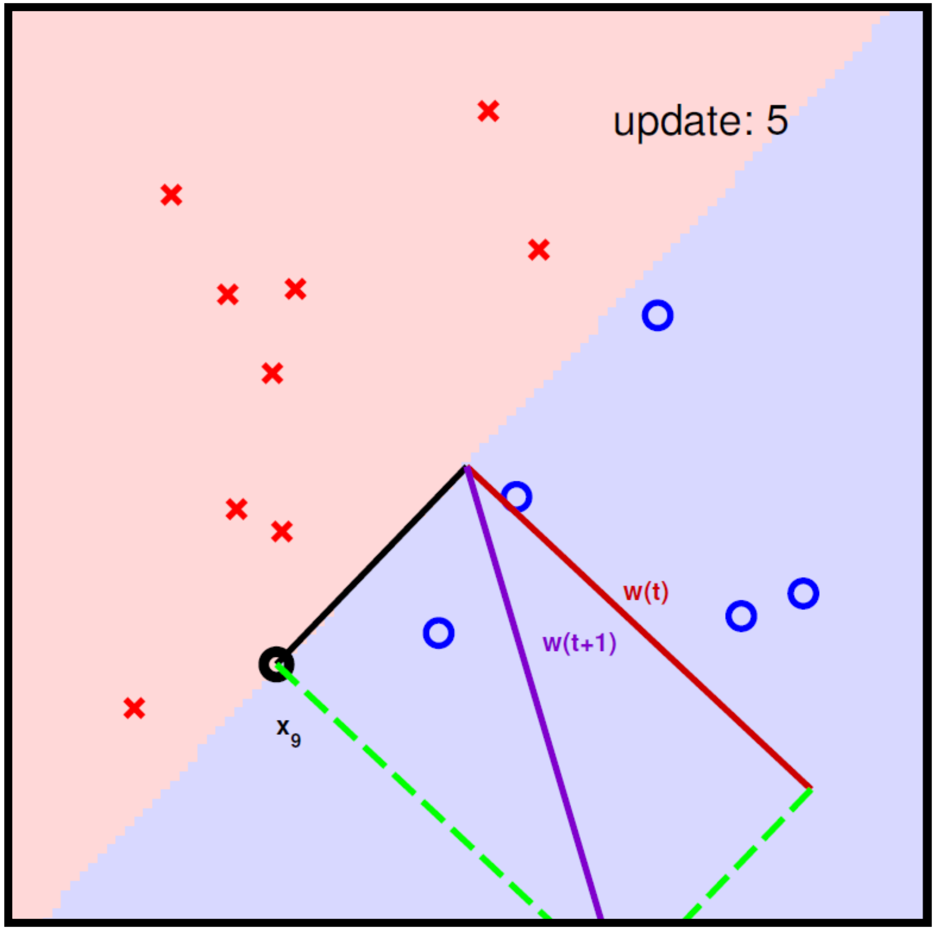
Seeing is Believing



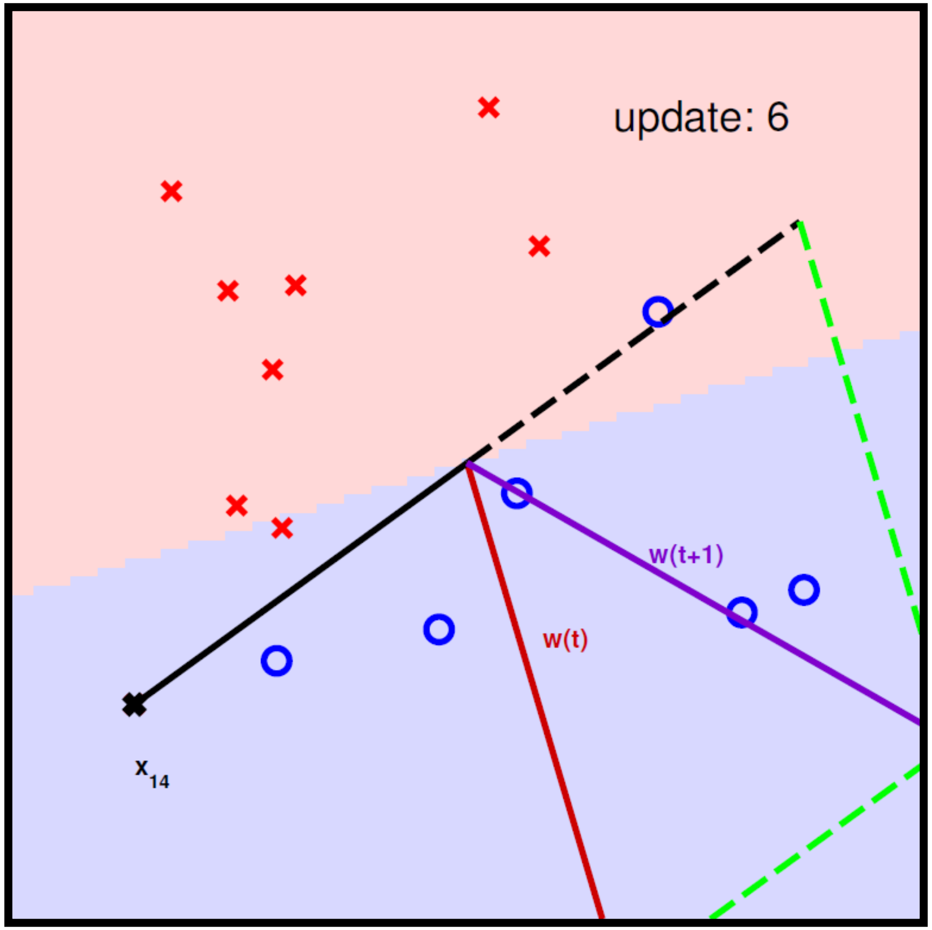
Seeing is Believing



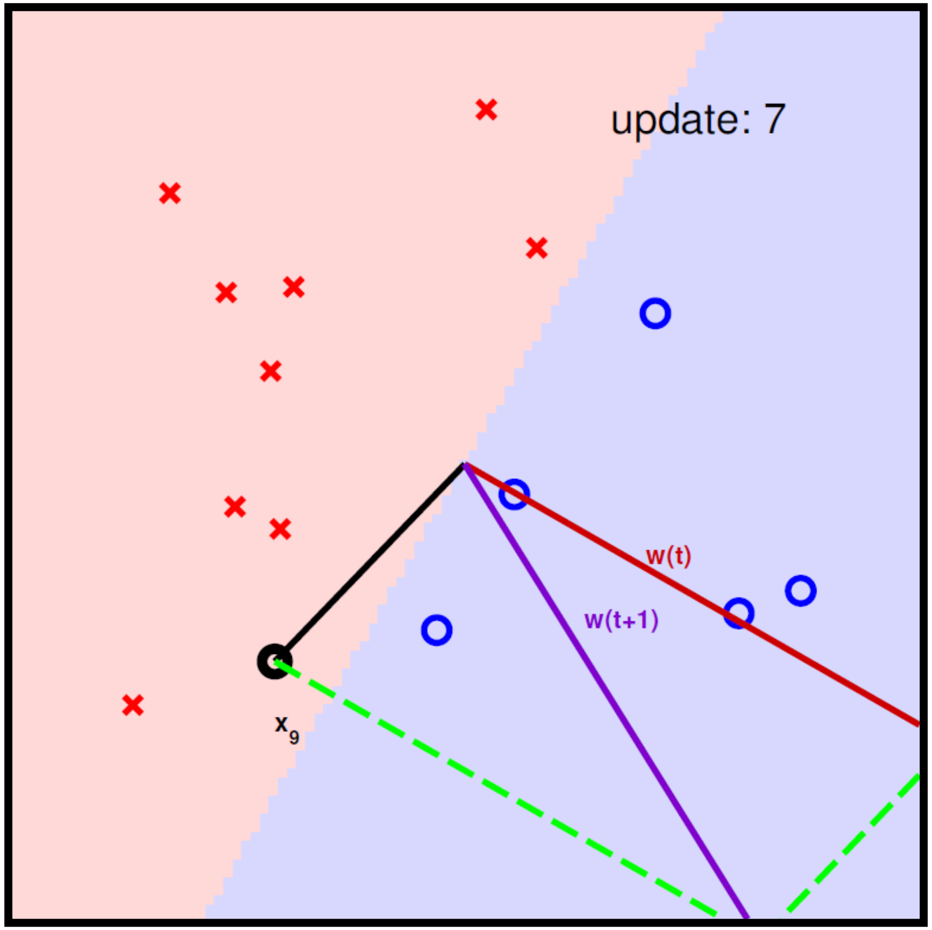
Seeing is Believing



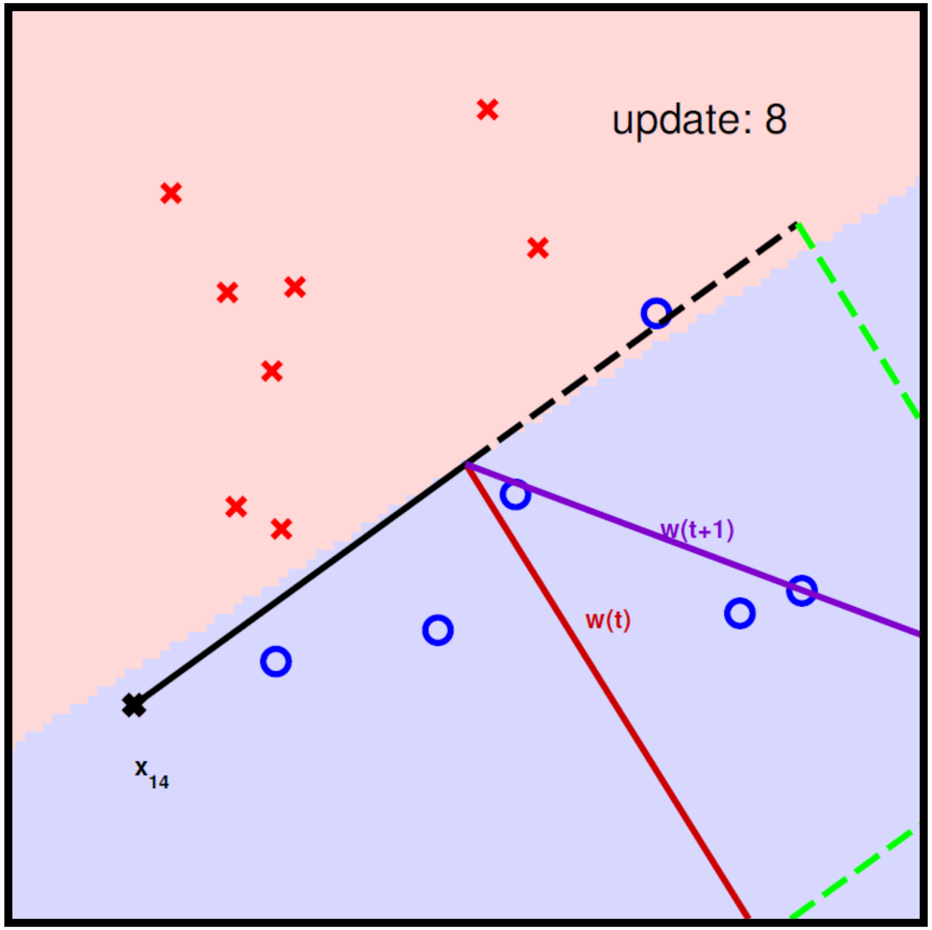
Seeing is Believing



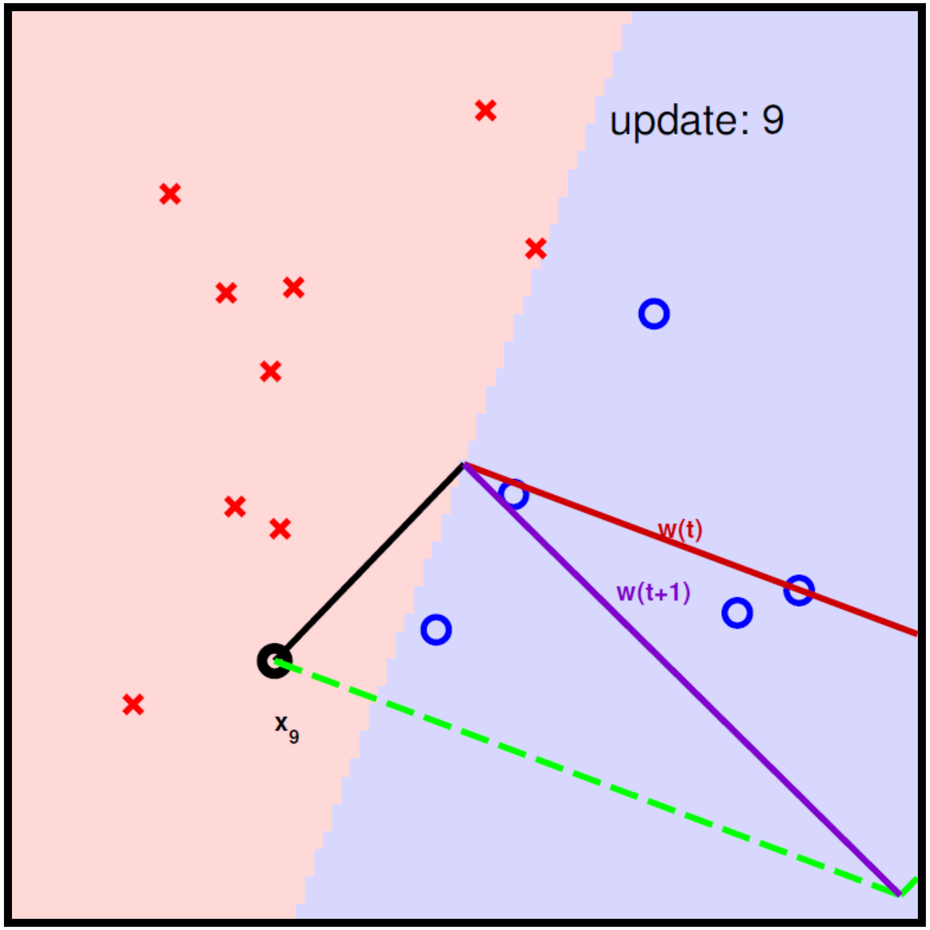
Seeing is Believing



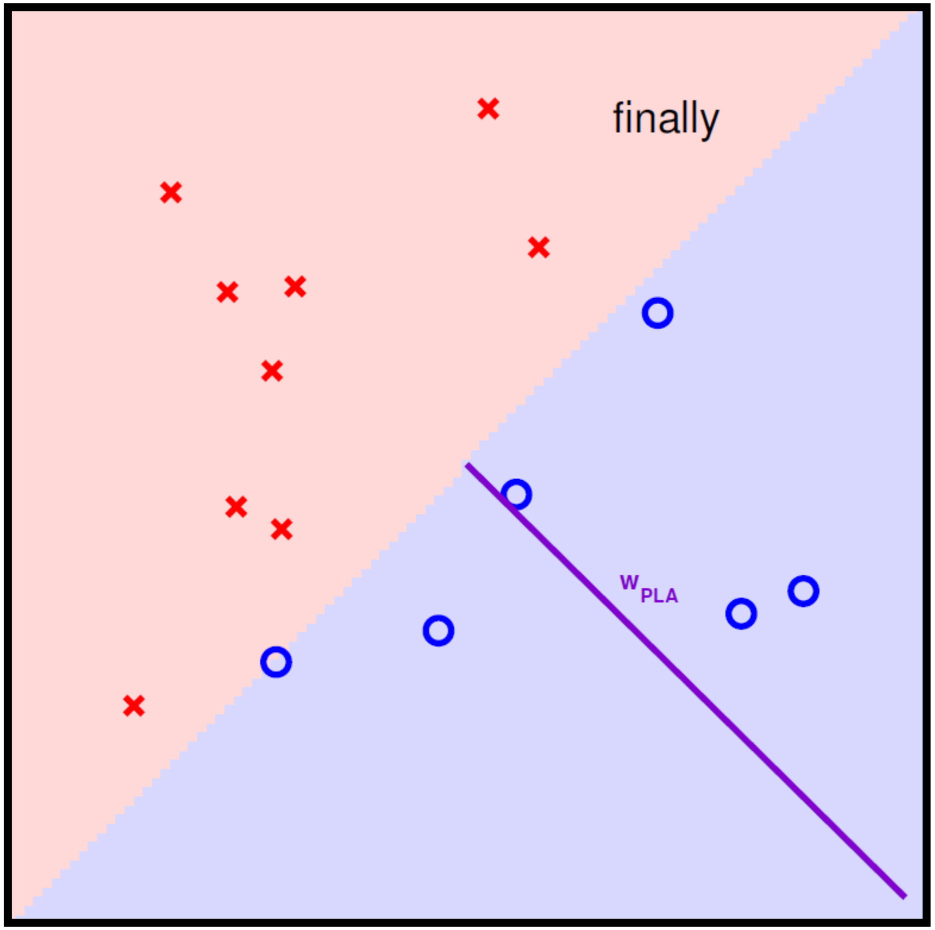
Seeing is Believing



Seeing is Believing



Seeing is Believing



感知机算法的对偶形式

- 对偶形式的基本思路是：将 w 和 b 表示为实例 x^i 和标记 y^i 的线性组合的形式，通过求解其系数而求得 w 和 b
- 不失一般性，假设初始值 w_0 和 b_0 均为0，对误分类点 (x^i,y^i) 采用：

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \eta y^i \mathbf{x}^i \qquad b_{t+1} = b_t + \eta y^i$$

- 逐步修改 w 和 b，设修改n次，则 w 和 b 关于 (x^i,y^i) 的增量分别为

$\alpha^i y^i \mathbf{x}^i$ 和 $\alpha^i y^i$，其中：$\alpha^i = n_i \eta$。由此 w 和 b 可以表示为：

$$\mathbf{w} = \sum_{i=1}^{N} \alpha^i y^i \mathbf{x}^i \qquad b = \sum_{i=1}^{N} \alpha^i y^i$$

- 显然：当 $\eta = 1$ 时，α^i 表示第i个实例点由于误分而更新的次数
- 下面对照原始形式来介绍感知机学习算法的对偶形式

感知机算法的对偶形式

- 将上述表达式代入感知机算法的表达式：

$$f(\mathbf{x}) = sign(\sum_{j=1}^{N} \alpha^j y^j \mathbf{x}^j \cdot \mathbf{x} + b)$$

- （1）令 $\alpha^i = 0$，$b = 0$
- （2）在训练集中选择输入数据：$(\mathbf{x}^i, y^i)$
- （3）参数更新，若：

$$y^i \left(\sum_{j=1}^{N} \alpha^j y^j \mathbf{x}^j \cdot \mathbf{x}^i + b \right) \leq 0$$

  则更新参数：$\alpha_{t+1}^i = \alpha_t^i + \eta$，$b_{t+1}^i = b_t^i + \eta y^i$

- （4）转至（2）直到没有误分类数据

感知机算法小结

- 感知机是二分类线性分类模型，是神经网络与支持向量机的基础
  - 与SVM相比，Perceptron仅考虑是否将所有训练数据正确分类
- 若数据线性可分，可以证明感知机在有限次迭代过程中收敛
  - Minsky在1969年证明了感知机无法解决许多基本问题（如异或问题）
  - 很难从样本数据集直接看出问题是否线性可分
  - 未能证明，一个感知机究竟需要经过多少步才能完成训练
  - 单个感知器不能对线性不可分问题实现两类分类
  - 单层感知器不能对线性不可分问题实现多类分类
- 感知机算法可以改写为对偶形式
  - 借助Gram矩阵可以大幅减少训练过程的计算量

拉格朗日对偶性

拉格朗日对偶性

- 在约束最优化问题中，常利用拉格朗日对偶性将原始问题转化为对偶问题，通过解对偶问题得到原始问题的解。
- 原始问题
  - 设：$f(x), c_i(x), h_j(x)$ 是定义在 $\mathbb{R}^n$ 上的连续可微函数
  - 考虑约束最优化问题：

Inequality constraints

$$\min_{x\in\mathbb{R}^n} f(x) \qquad \text{s.t.} \quad c_i(x) \le 0, \quad i = 1, 2, \ldots, k$$

Objective function

$$h_j(x) = 0, \quad j = 1, 2, \ldots, l$$

Equality constraints

- 称此约束最优化问题为原始最优化问题或原始问题

原始问题

- 引入广义拉格朗日函数（generalized Lagrange function）

$$\mathcal{L}(x,\alpha,\beta)=f(x)+\sum_{i=1}^{k}\alpha_i c_i(x)+\sum_{j=1}^{l}\beta_j h_j(x)$$

- 其中: $\alpha_i \geq 0, \beta_j$ 称为拉格朗日乘子。原始问题可表示为x的函数:

$$\theta_P(x)=\max_{\alpha,\beta:\alpha_i\geq 0}\mathcal{L}(x,\alpha,\beta)$$

- 其中：下标P表示原始问题。考虑某个x违反了原始的约束，即存在某个i或j，使得: $c_i(x)>0$ 或 $h_j(x)\neq 0$，则有:

$$\theta_P(x)=\max_{\alpha,\beta:\alpha_i\geq 0}\left[f(x)+\sum_{i=1}^{k}\alpha_i c_i(x)+\sum_{j=1}^{l}\beta_j h_j(x)\right]=+\infty$$

- 相反地，若x满足约束条件，则: $\theta_P(x)=\max_{\alpha,\beta:\alpha_i\geq 0} f(x)=f(x)$

原始问题

- 综上可知:

$$\theta_P(x) = \begin{cases} f(x), & x \text{ meet the original constraints} \\ +\infty, & \text{otherwise} \end{cases}$$

- 则如下极小化问题与原始最优化问题等价（即二者有相同的解）：

$$\min_x \theta_P(x) = \min_x \max_{\alpha,\beta:\alpha_i\geq 0} \mathcal{L}(x,\alpha,\beta)$$

- 上式称为广义拉格朗日函数的极小极大问题（Minimax）
- 方便起见，将原始问题的最优值称为原始问题的值

$$p^* = \min_x \theta_P(x)$$

对偶问题

- 已知

$$\min_x \theta_P(x) = \min_x \max_{\alpha,\beta:\alpha_i\geq 0} \mathcal{L}(x,\alpha,\beta)$$

- 定义

$$\theta_D(\alpha,\beta) = \min_x \mathcal{L}(x,\alpha,\beta)$$

- 再考虑极大化问题：

$$\max_{\alpha,\beta:\alpha_i\geq 0} \theta_D(\alpha,\beta) = \max_{\alpha,\beta:\alpha_i\geq 0} \min_x \mathcal{L}(x,\alpha,\beta)$$

- 该问题称为广义拉格朗日函数的极大极小问题

对偶问题

- 可以将广义拉格朗日函数的极大极小问题表示为约束最优化问题

$$\max_{\alpha,\beta}\theta_D(\alpha,\beta)=\max_{\alpha,\beta}\min_x\mathcal{L}(x,\alpha,\beta)$$

$$\text{s.t.}\quad \alpha_i\geq 0,\quad i=1,2,\ldots,k$$

- 称为原始问题的对偶问题。定义对偶问题的最优值：

$$d^*=\max_{\alpha,\beta:\alpha_i\geq 0}\theta_D(\alpha,\beta)$$

- 称为对偶问题的值。

原始问题与对偶问题的关系

- 定理1. 若原始问题和对偶问题都有最优值，则:

$$d^* = \max_{\alpha,\beta:\alpha_i \geq 0} \min_x \mathcal{L}(x,\alpha,\beta) \leq \min_x \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(x,\alpha,\beta) = p^*$$

- 证明：已知 $\theta_P(x) = \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(x,\alpha,\beta)$ $\theta_D(\alpha,\beta) = \min_x \mathcal{L}(x,\alpha,\beta)$

- 对任意的 α, β and x ，有:

$$\theta_D(\alpha,\beta) = \min_x \mathcal{L}(x,\alpha,\beta) \leq \mathcal{L}(x,\alpha,\beta) \leq \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(x,\alpha,\beta) = \theta_P(x)$$

- 即： $\theta_D(\alpha,\beta) \leq \theta_P(x)$

- 由于原始问题和对偶问题均有最优值，所以:

$$\max_{\alpha,\beta:\alpha_i \geq 0} \theta_D(\alpha,\beta) \leq \min_x \theta_P(x)$$

- 即： $d^* = \max_{\alpha,\beta:\alpha_i \geq 0} \min_x \mathcal{L}(x,\alpha,\beta) \leq \min_x \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(x,\alpha,\beta) = p^*$

原始问题与对偶问题的关系

- 推论1. 设 α^*, β^*, x^* 分别为原始问题和对偶问题的可行解，且：

$$d^* = p^*$$

- 则： α^*, β^*, x^* 分别为原始问题和对偶问题的最优解
- 解读：当原始问题和对偶问题的最优值相等时
  - 可以用求解对偶问题替代求解原始问题
  - 前提是对偶问题求解比直接求解原始问题简单
- 问题：什么情况下 $d^* = p^*$ ？
  - Karush–Kuhn–Tucker (KKT) conditions
  - KKT条件给出了判断 x^* 是否为最优解的必要条件

原始问题与对偶问题的关系

- 定理2. 考虑原始问题和对偶问题
- 假设：函数 $f(x), c_i(x)$ 是凸函数，$h_j(x)$ 是仿射函数
- 并假设：不等式约束 $c_i(x)$ 严格可行，即：$\exists x, \ \forall i, \ c_i(x) < 0$
- 则存在 x^*, α^*, β^*，使 x^* 是原始问题的解，α^*, β^* 是对偶问题的解
- 并且：$p^* = d^* = \mathcal{L}(x^*, \alpha^*, \beta^*)$

原始问题与对偶问题的关系

- 定理3. 考虑原始问题和对偶问题
- 假设：函数 $f(x), c_i(x)$ 是凸函数，$h_j(x)$ 是仿射函数
- 并假设：不等式约束 $c_i(x)$ 严格可行，即：$\exists x,\ \forall i,\ c_i(x) < 0$
- 则存在 x^*, α^*, β^*，使 x^* 是原始问题的解，α^*, β^* 是对偶问题的解的充要条件是 x^*, α^*, β^* 满足下面的KKT条件：

$$
\begin{cases}
\nabla_x \mathcal{L}(x^*, \alpha^*, \beta^*) = 0 \\
\alpha^* c_i(x^*) = 0, \quad i = 1, 2, \ldots, k \\
c_i(x^*) \leq 0, \quad i = 1, 2, \ldots, k \\
\alpha^* \geq 0, \quad i = 1, 2, \ldots, k \\
h_j(x^*) = 0, \quad j = 1, 2, \ldots, l
\end{cases}
$$

- 若：$\alpha^* \geq 0$
- 则：$c_i(x^*) = 0$
- 称为KKT的对偶互补条件

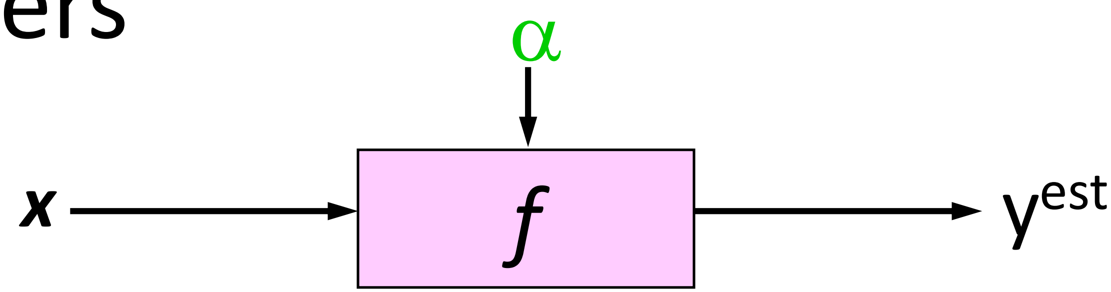
支持向量机

支持向量机 SVM

- SVM is a classifier derived from statistical learning theory by Vapnik and Chervonenkis
- SVMs are learning systems that

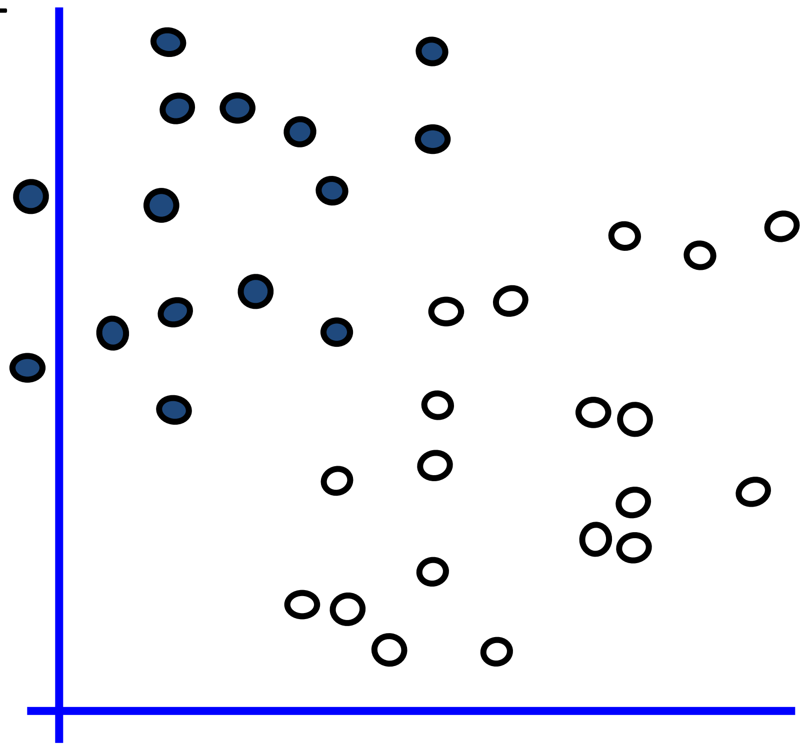
  - use a hyperplane of *linear functions*
  - in a high dimensional feature space — *Kernel function*
  - trained with a learning algorithm from optimization theory — *Lagrangian duality*
  - Implements a learning bias derived from statistical learning theory — *Generalisation*

Linear Classifiers



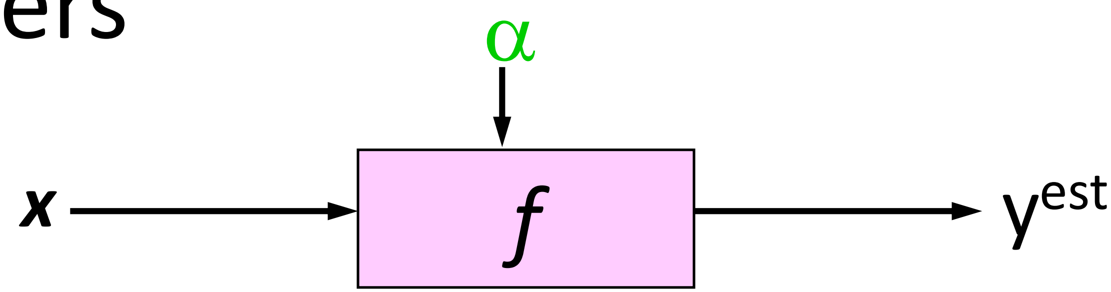
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- denotes +1
- denotes -1



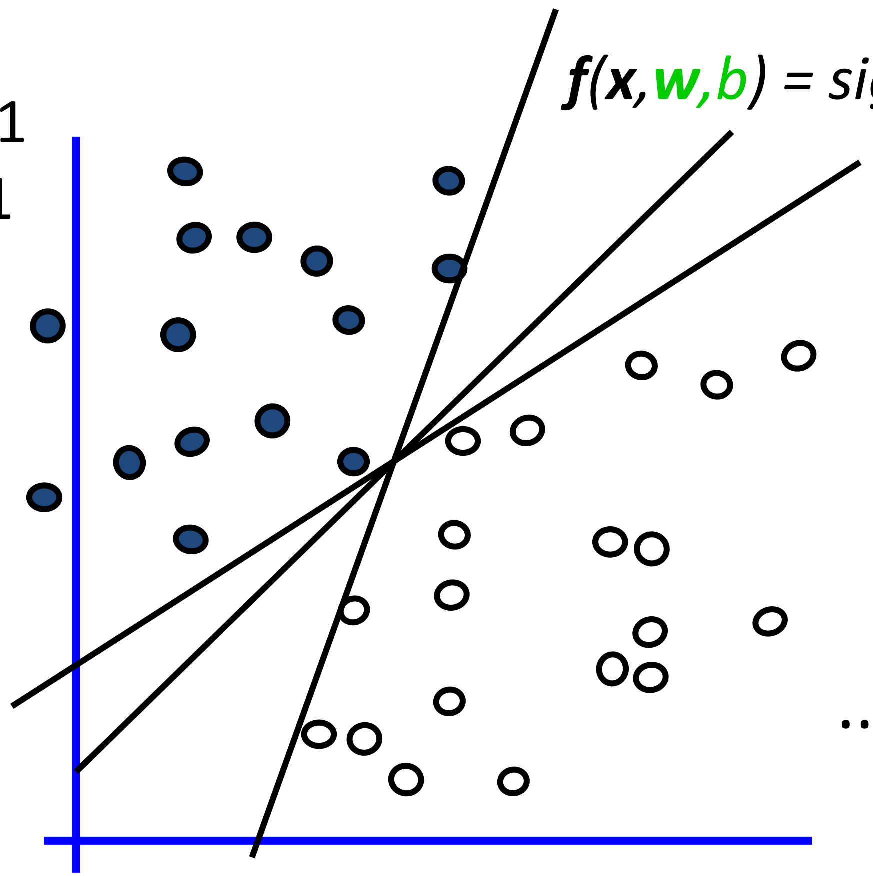
How would you classify this data?

Linear Classifiers



$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

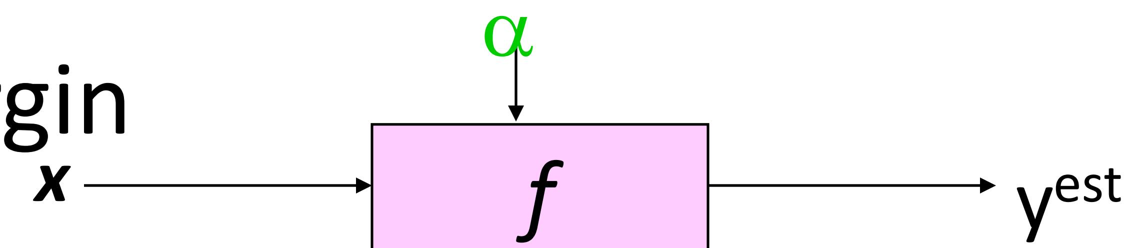
- denotes +1
- denotes -1



Any of these would be fine..

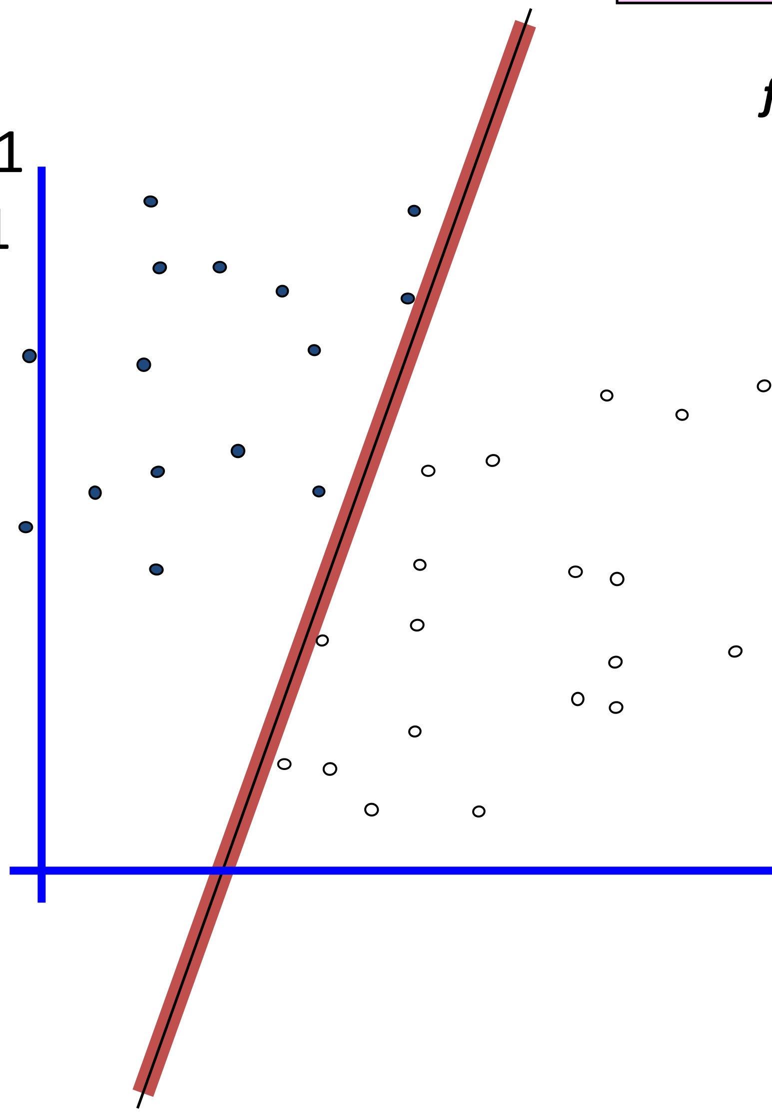
..but which is best?

Classifier Margin



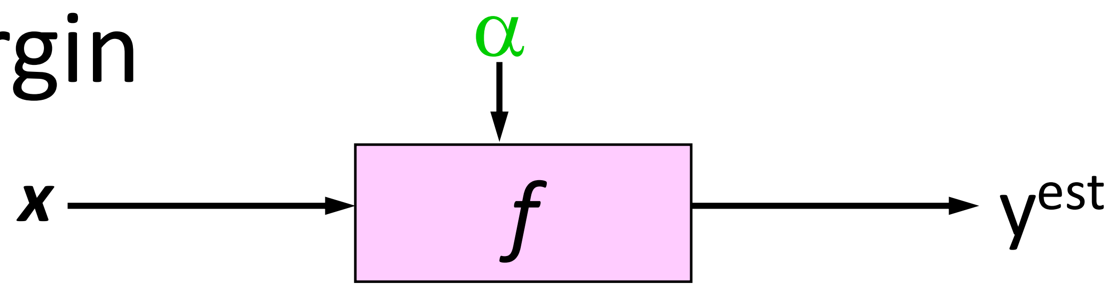
$f(\boldsymbol{x},\boldsymbol{w},b) = sign(\boldsymbol{w}.\ \boldsymbol{x} - b)$

- denotes +1
- ○ denotes -1

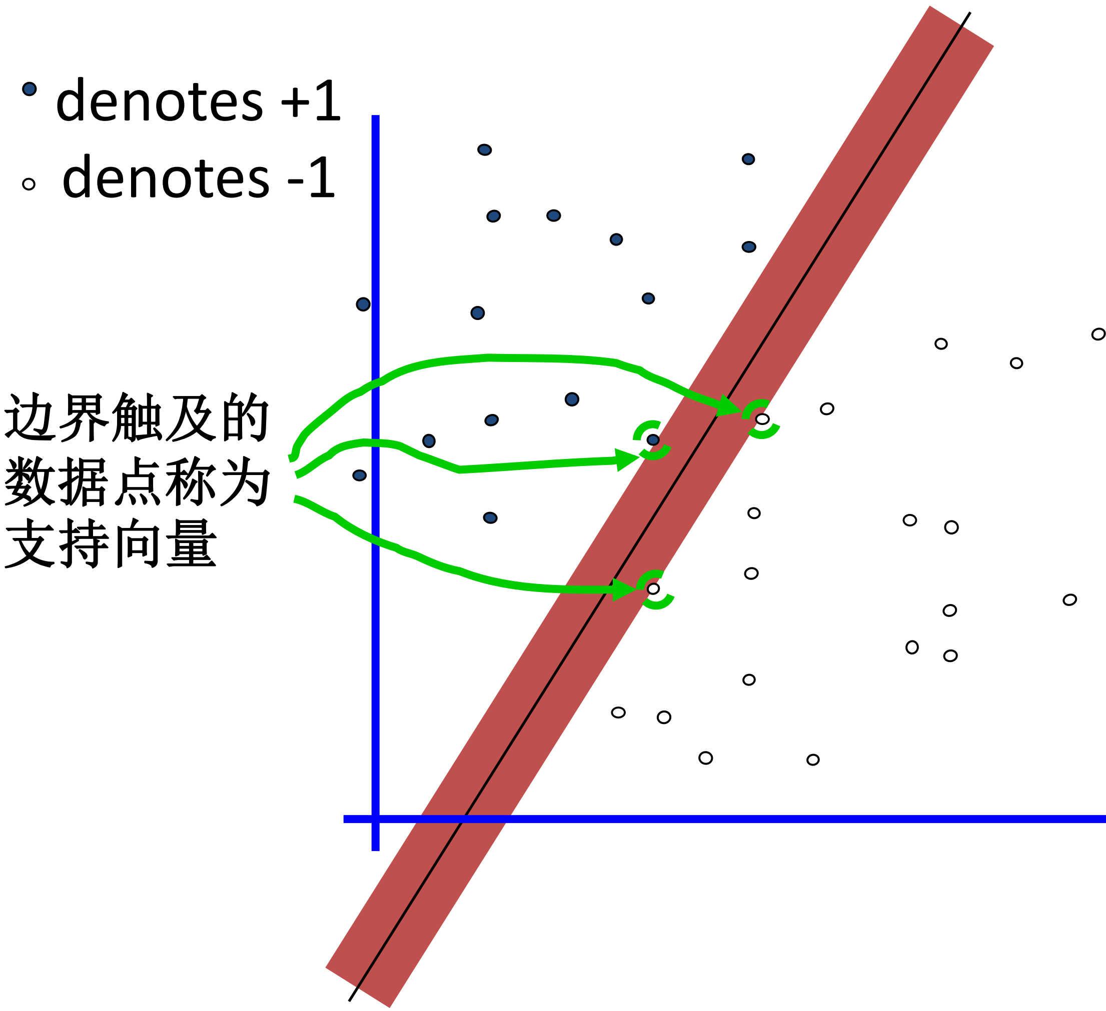


边界（**Margin**）定义为分界线的宽度：当分界线的宽度增长为恰好与数据点相接触时的宽度。

Maximum Margin



$$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$$



最大边界线性分类器（maximum margin linear classifier）是具有最大边界的线性分类器，这也是SVM的最简单形式，称为Linear SVM（LSVM）

电子科技大学

UeSTC

1956