ITF: **Information Technology Faculty**
Thuyloi University

Máy véc tơ hỗ trợ SVM

Trình bày: PGS.TS Nguyễn Hữu Quỳnh

Bài giảng được dựa trên giáo trình machine learning cơ bản

Giới thiệu

Khoảng cách từ một điểm tới một siêu mặt phẳng

- Trong không gian 2 chiều, khoảng cách từ một điểm có toạ độ (x_0, y_0) tới *đường thẳng* có phương trình $w_1x + w_2y + b = 0$ được xác định bởi:

$$\frac{|w_1x_0 + w_2y_0 + b|}{\sqrt{w_1^2 + w_2^2}}$$

- Trong không gian ba chiều, khoảng cách từ một điểm có toạ độ (x_0, y_0, z_0) tới một *mặt phẳng* có phương trình $w_1x + w_2y + w_3z + b = 0$ được xác định bởi:

$$\frac{|w_1x_0 + w_2y_0 + w_3z_0 + b|}{\sqrt{w_1^2 + w_2^2 + w_3^2}}$$

Giới thiệu

- Nếu bỏ dấu trị tuyệt đối ở tử số, ta biết được điểm đó nằm về phía nào của *mặt phẳng* đang xét:
  - Những điểm mang dấu dương nằm về cùng 1 phía
  - Những điểm mang dấu âm nằm về phía còn lại
  - Những điểm nằm trên *măt phẳng* làm cho tử số có giá trị bằng 0, tức khoảng cách bằng 0.
- Tổng quát lên không gian nhiều chiều: Khoảng cách từ một điểm (vector) có toạ độ x_0 tới *siêu mặt phẳng* (*hyperplane*) có phương trình $w^T x + b = 0$:

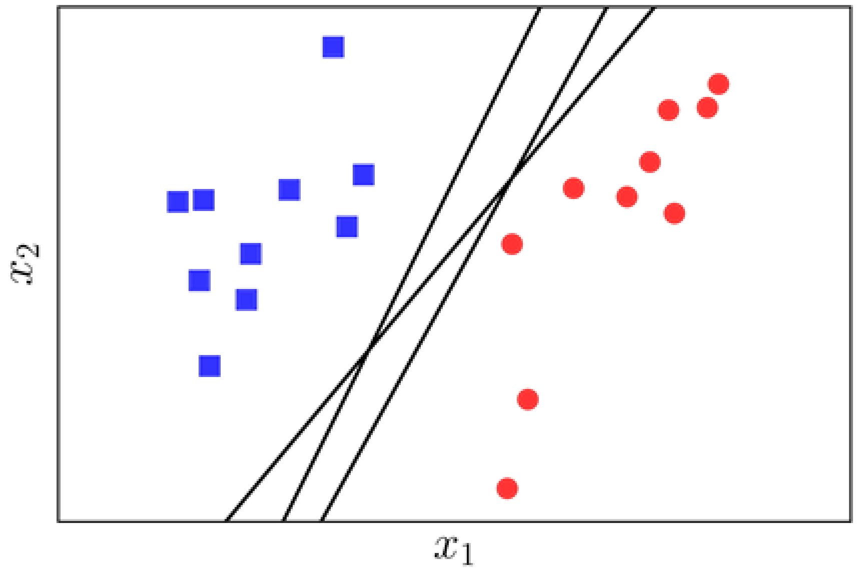
$$\frac{|\mathbf{w}^T \mathbf{x}_0 + b|}{||\mathbf{w}||_2}$$

Với $||\mathbf{w}||_2 = \sqrt{\sum_{i=1}^{d} w_i^2}$, d là số chiều của không gian

Giới thiệu

Nhắc lại bài toán phân chia hai classes

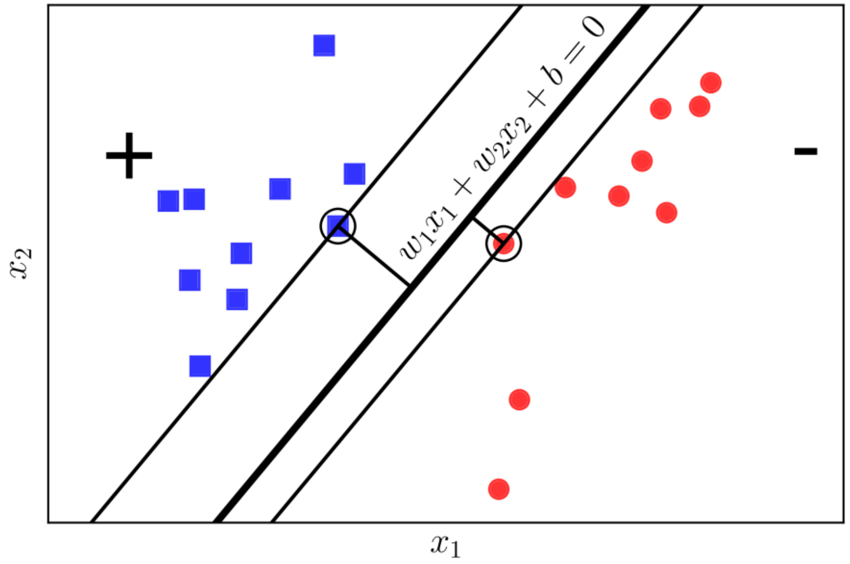
- Ta cùng quay lại với bài toán trong PLA: Hãy tìm một siêu mặt phẳng phân chia hai classes

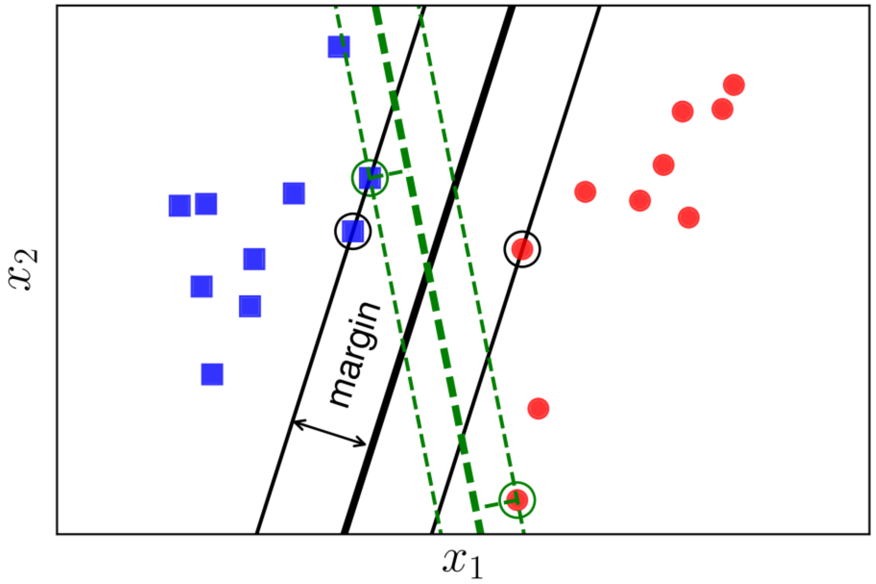


- Thuật toán PLA có thể làm được việc này nhưng nó có thể cho chúng ta vô số nghiệm

Giới thiệu

- Ta cần tìm một tiêu chuẩn để đo sự *công bằng* của hai class





Giới thiệu

- Ta cần một đường phân chia sao cho khoảng cách từ điểm gần nhất của mỗi class tới đường phân chia là như nhau, như thế thì mới *công bằng:*
  - Khoảng cách như nhau này được gọi là *margin* (*lề*).
  - *Margin* (*lề*) phải cực đại để mang lại hiệu ứng phân lớp tốt hơn vì *sự phân chia giữa hai classes là rạch ròi hơn*

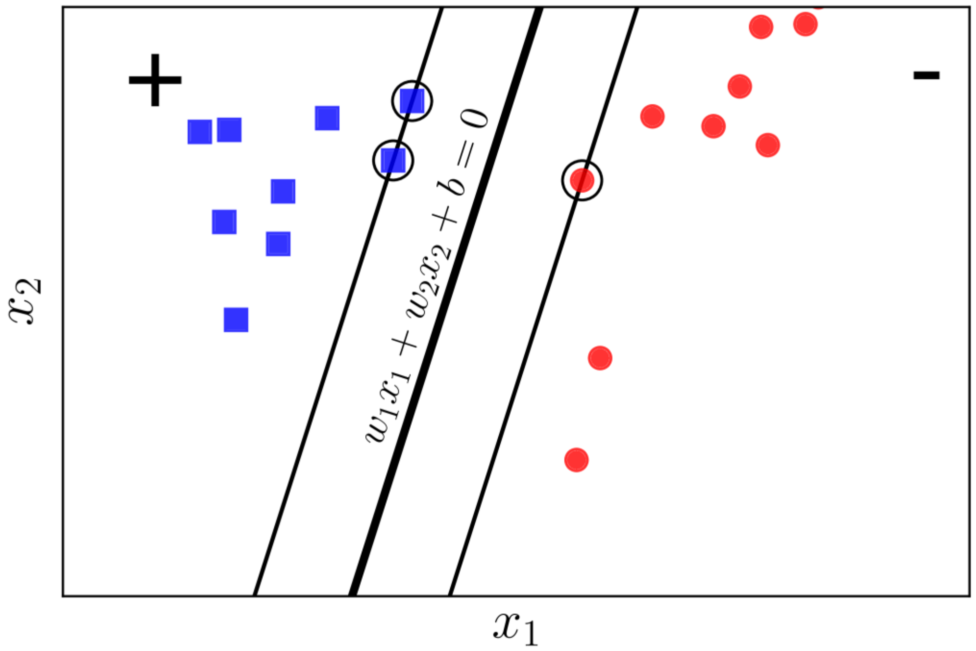
- Bài toán tối ưu trong *Support Vector Machine* (SVM) chính là bài toán đi tìm đường phân chia sao cho *margin* là lớn nhất. Đây cũng là lý do vì sao SVM còn được gọi là *Maximum Margin Classifier*.

Xây dựng bài toán tối ưu cho SVM

- Giả sử rằng các cặp dữ liệu của *training set* là $(x_1, y_1), (x_2, y_2), \dots, (xN, yN)$ với:
  - vector $x_i \in R^d$ thể hiện *đầu vào* của một điểm dữ liệu
  - y_i là *nhãn* của điểm dữ liệu đó
  - d là số chiều của dữ liệu
  - N là số điểm dữ liệu.
- Giả sử rằng *nhãn* của mỗi điểm dữ liệu được xác định bởi $y_i = 1$ (class 1) hoặc $y_i = -1$ (class 2) giống như trong PLA.

Xây dựng bài toán tối ưu cho SVM

- Ta cùng xét trường hợp trong không gian hai chiều



- Giả sử rằng các điểm vuông xanh thuộc class 1, các điểm tròn đỏ thuộc class -1 và mặt $w^Tx + b = w_1x_1 + w_2x_2 + b = 0$ là mặt phân chia giữa hai classes (Hình bên). Hơn nữa, class 1 nằm về *phía dương*, class -1 nằm về *phía âm* của mặt phân chia.

- Chú ý rằng ta cần đi tìm các hệ số w và b.

Xây dựng bài toán tối ưu cho SVM

- Ta quan sát thấy một điểm quan trọng sau đây: với cặp dữ liệu (xn,yn)(xn,yn) bất kỳ, khoảng cách từ điểm đó tới mặt phân chia là:

$$\frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||_2}$$

- Điều này có thể dễ nhận thấy vì theo giả sử ở trên, y_n luôn cùng dấu với *phía* của x_n. Từ đó suy ra y_n cùng dấu với $(w^T x_n + b)$, và tử số luôn là 1 số không âm.

- Với mặt phân chia như trên, *margin* được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó:

$$\text{margin} = \min_n \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||_2}$$

Xây dựng bài toán tối ưu cho SVM

- Bài toán tối ưu trong SVM chính là bài toán tìm w và bb sao cho *margin* này đạt giá trị lớn nhất:

$$(\mathbf{w}, b) = \arg\max_{\mathbf{w}, b} \left\{ \min_n \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||_2} \right\} = \arg\max_{\mathbf{w}, b} \left\{ \frac{1}{||\mathbf{w}||_2} \min_n y_n(\mathbf{w}^T\mathbf{x}_n + b) \right\}$$

- Việc giải trực tiếp bài toán này sẽ rất phức tạp, nhưng ta có cách để đưa nó về bài toán đơn giản hơn.

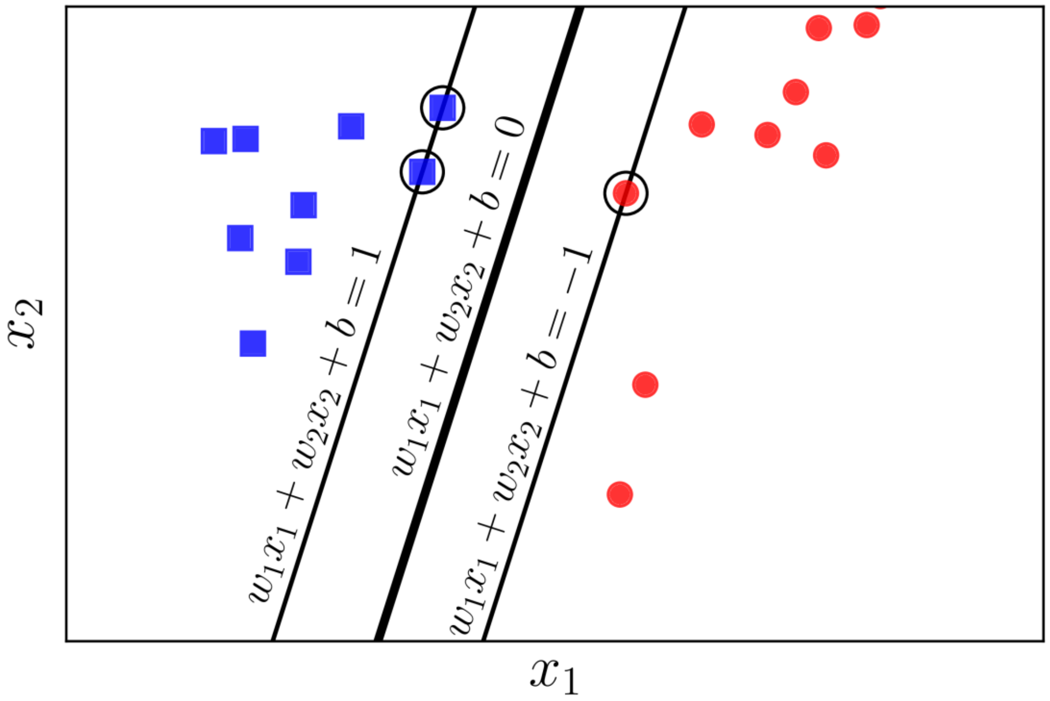
- Nhận xét quan trọng nhất là nếu ta thay vector hệ số w bởi kw và b bởi kb trong đó k là một hằng số dương thì mặt phân chia không thay đổi, tức khoảng cách từ từng điểm đến mặt phân chia không đổi, tức *margin* không đổi.

Xây dựng bài toán tối ưu cho SVM

- Dựa trên nhận xét, ta có thể giả sử:

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) = 1$$

với những điểm nằm gần mặt phân chia nhất như Hình:



Xây dựng bài toán tối ưu cho SVM

- Như vậy, với mọi n, ta có:

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$$

- Vậy bài toán tối ưu

$$(\mathbf{w}, b) = \arg\max_{\mathbf{w},b} \left\{ \min_n \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||_2} \right\} = \arg\max_{\mathbf{w},b} \left\{ \frac{1}{||\mathbf{w}||_2} \min_n y_n(\mathbf{w}^T\mathbf{x}_n + b) \right\}$$

Có thể đưa về bài toán tối ưu có ràng buộc sau đây:

$$(\mathbf{w}, b) = \arg\max_{\mathbf{w},b} \frac{1}{||\mathbf{w}||_2}$$

$$\text{subject to:} \quad y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n = 1, 2, \dots, N$$

Xây dựng bài toán tối ưu cho SVM

- Vậy bài toán tối ưu

$$(\mathbf{w}, b) = \arg\max_{\mathbf{w}, b} \frac{1}{||\mathbf{w}||_2}$$
$$\text{subject to:} \qquad y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n = 1, 2, \dots, N$$

là một bài toán lồi. Hơn nữa, nó là một Quadratic Programming.

- Từ đây có thể suy ra nghiệm cho SVM là *duy nhất*.

Tóm tắt

- Với bài toán binary classification mà 2 classes là *linearly separable*, có vô số các siêu mặt phẳng giúp phân biệt hai classes:
    - Với mỗi mặt phân cách, ta có một *classifier*
    - Khoảng cách gần nhất từ 1 điểm dữ liệu tới mặt phân cách ấy được gọi là *margin* của classifier đó.
- Support Vector Machine là bài toán đi tìm mặt phân cách sao cho *margin* tìm được là lớn nhất, đồng nghĩa với việc các điểm dữ liệu *an toàn nhất* so với mặt phân cách.
- Bài toán tối ưu trong SVM là một bài toán lồi với hàm mục tiêu là *stricly convex*, nghiệm của bài toán này là duy nhất. Hơn nữa, bài toán tối ưu đó là một Quadratic Programming (QP).

Tóm tắt

- Mặc dù có thể trực tiếp giải SVM qua bài toán tối ưu gốc này, thông thường người ta thường giải bài toán đối ngẫu.
- Bài toán đối ngẫu cũng là một QP nhưng nghiệm là *sparse* nên có những phương pháp giải hiệu quả hơn.
- Với các bài toán mà dữ liệu *gần linearly separable* hoặc *nonlinear separable*, có những cải tiền khác của SVM để thích nghi với dữ liệu đó.