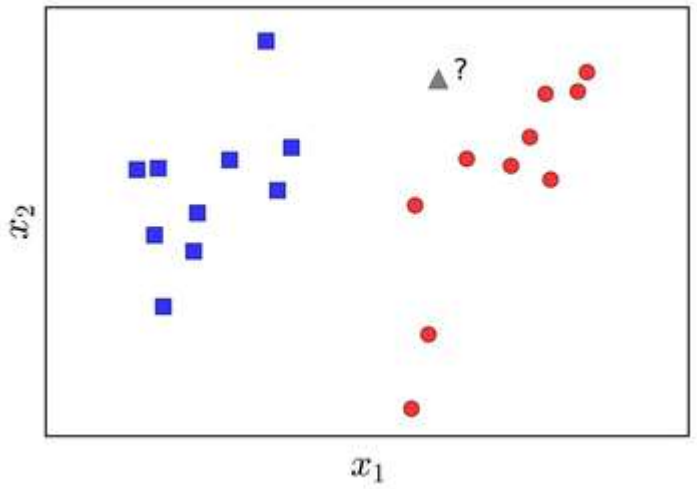
CSE: Faculty of Computer Science and Engineering
Thuyloi University

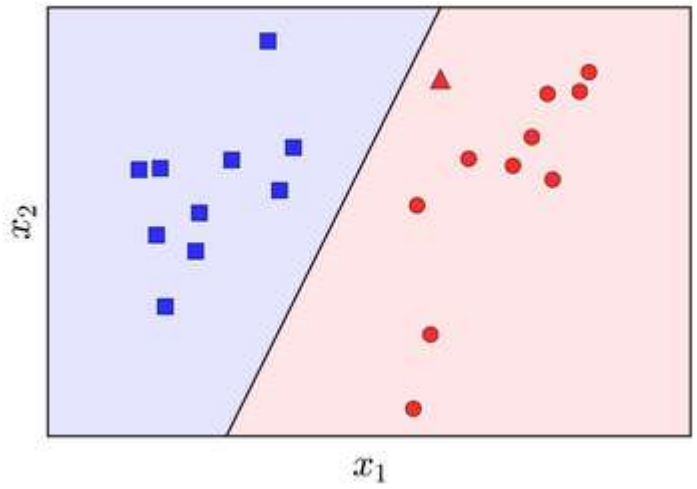
Perceptron Learning Algorithm

Trình bày: PGS.TS Nguyễn Hữu Quỳnh

Giới thiệu

- Perceptron là một thuật toán Classification cho trường hợp đơn giản nhất:
  - chỉ có hai class (*binary classification*)
  - cũng chỉ hoạt động được trong một trường hợp rất cụ thể
- Giả sử chúng ta có hai tập hợp dữ liệu đã được gán nhãn





Bài toán Perceptron

- *Cho hai class được gán nhãn, hãy tìm một đường phẳng sao cho:*
  - *toàn bộ các điểm thuộc class 1 nằm về 1 phía,*
  - *toàn bộ các điểm thuộc class 2 nằm về phía còn lại của đường phẳng đó.*
- Nếu tồn tại một đường phẳng phân chia hai class thì ta gọi hai class đó là *linearly separable*.
- Các thuật toán classification tạo ra các boundary là các đường phẳng được gọi chung là Linear Classifier.

Thuật toán Perceptron (PLA)

- ý tưởng cơ bản của PLA:
  - xuất phát từ một nghiệm dự đoán nào đó,
  - qua mỗi vòng lặp, nghiệm sẽ được cập nhật tới một ví trí tốt hơn
  - cập nhật dựa trên việc giảm giá trị của một hàm mất mát nào đó.
- G/S ma trận chứa các điểm dữ liệu:
  - $\mathrm{X} = [\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_N] \in \mathbb{R}^{d \times N}$
  - Mỗi cột $\mathrm{x}_i \in \mathbb{R}^{d \times 1}$ là một điểm dữ liệu trong không gian d chiều
- G/S các nhãn của mỗi điểm dữ liệu được lưu trong một véc tơ hàng
  - $\mathrm{y} = [y_1, y_2, \dots, y_N] \in \mathbb{R}^{1 \times N}$
  - Với $y_i = 1$ nếu x_i thuộc class 1 và $y_i = -1$ nếu x_i thuộc class 2

Thuật toán Perceptron (PLA)

- G/S, tại một thời điểm, ta tìm được đường boundary là đường phẳng có phương trình:

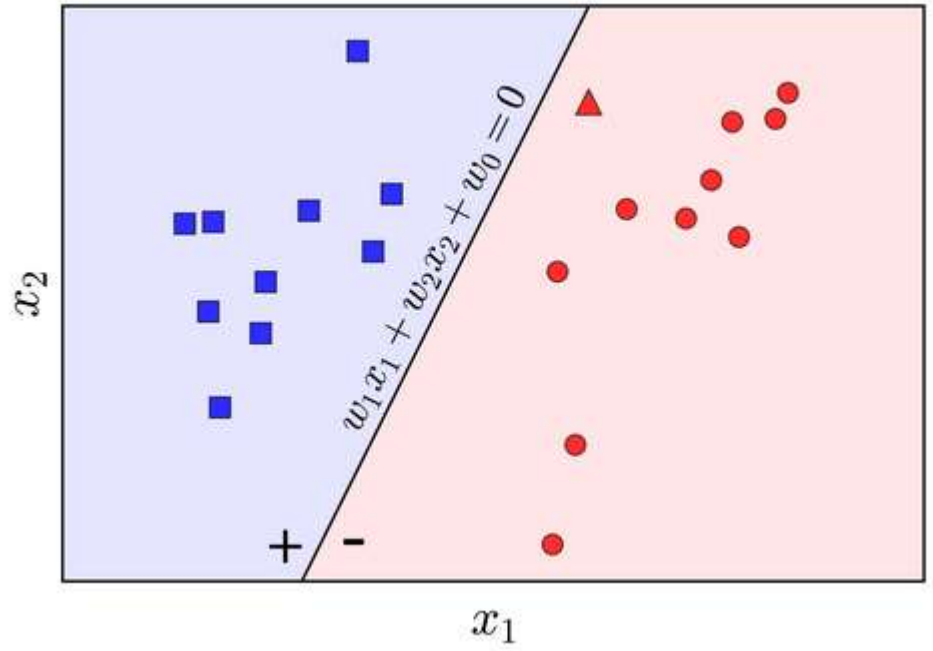
$$f_{\mathrm{w}}(\mathrm{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + w_0 = 0$$

Hay

$$\mathrm{w}^T \mathrm{x} = 0$$

Thuật toán Perceptron (PLA)

- Để đơn giả, ta làm việc với trường hợp mỗi điểm dữ liệu chỉ có hai chiều (d=2)
- G/S đường thẳng $w_1 x_1 + w_2 x_2 + w_0 = 0$ là nghiệm cần tìm như hình:



Thuật toán Perceptron (PLA)

- Như vậy, các điểm nằm về cùng một phía so với đường thẳng làm cho $f_{\mathrm{w}}(\mathrm{x})$ mang cùng dấu
- Các dấu này tương ứng với nhãn y của mỗi lớp
- Vậy, nếu tìm được w (nghiệm của bài toán Perceptron) và một điểm x chưa có nhãn, ta có thể xác định class của nó bởi:

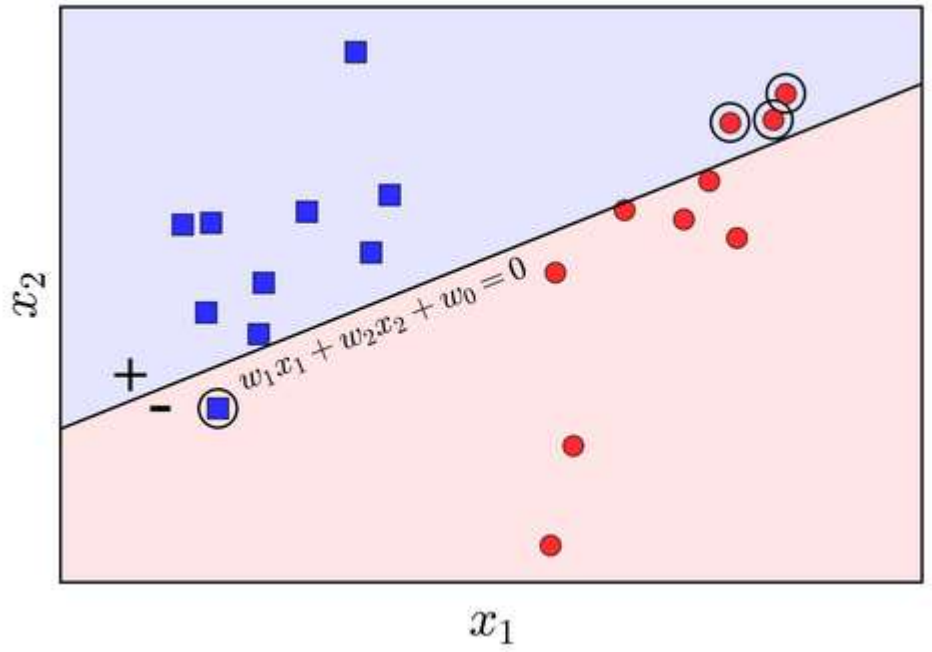
$$label(\mathrm{x}) = 1 \; n\acute{e}u \; \mathrm{w}^T\mathrm{x} \geq 0, ng\text{ược } l\text{ại } -1$$

Hay:

$$label(\mathrm{x}) = sgn(\mathrm{w}^T\mathrm{x})$$

Xây dựng hàm mất mát

- Tiếp theo, ta cần xây dựng hàm mất mát với tham số w bất kỳ.
- Vẫn trong không gian hai chiều, giả sử đường thẳng $w_1x_1 + w_2x_2 + w_0 = 0$ được cho như:



Xây dựng hàm mất mát

- Điều chúng ta mong muốn là không có điểm nào bị misclassified.
- Hàm mất mát đơn giản nhất chúng ta nghĩ đến là hàm *đếm* số lượng các điểm bị misclassied và tìm cách tối thiểu hàm số này:

$$J_1(\mathrm{w}) = \sum_{\mathrm{x}_i \in M} (-y_i sgn(\mathrm{w}^T \mathrm{x}_i))$$

- Hạn chế quan trọng: hàm số này là rời rạc, không tính được đạo hàm theo w nên rất khó tôi ưu.

Xây dựng hàm mất mát

- Xét hàm mất mát:

$$J(\mathrm{w}) = \sum_{\mathrm{x}_i \in M} (-y_i \ \mathrm{w}^T \mathrm{x}_i)$$

  - Khi một x_i (bị phân lớp sai) nằm càng xa boundary thì giá trị $-y_i \ \mathrm{w}^T \mathrm{x}_i$ càng lớn
  - Giá trị nhỏ nhất của hàm mất mát này bằng 0 nếu không có điểm nào bị phân lớp sai
  - Hàm này trừng phạt nặng những điểm lấn sâu sang lãnh thổ của lớp khác

Xây dựng hàm mất mát

- Tại một thời điểm, nếu ta chỉ quan tâm đến điểm bị phân lớp sai, hàm $J(\mathrm{w})$:
  - tính được đạo hàm
  - Ta có thể sử dụng giảm Gradient để tìm w
- Với một điểm x_ibị phân lớp sai, hàm mất mát trở thành:

$$J(\mathrm{w}, \mathrm{x}_i, y_i) = -y_i \ \mathrm{w}^T \mathrm{x}_i$$

- Đạo hàm:

$$\nabla_{\mathrm{w}} J(\mathrm{w}, \mathrm{x}_i, y_i) = -y_i \ \mathrm{x}_i$$

Xây dựng hàm mất mát

- Qui tắc cập nhật:

$$\mathrm{w} = \mathrm{w} + \eta y_i \, \mathrm{x}_i$$

- Ta có một quan sát nhỏ

$$\mathrm{w}_{t+1}^T \mathrm{x}_i = (\mathrm{w}_t + y_i \, \mathrm{x}_i)^T \mathrm{x}_i$$
$$= \mathrm{w}_t^T \, \mathrm{x}_i + y_i \|\mathrm{x}_i\|_2^2$$

- Nếu $y_i = 1$:
  - vì x_i bị phân lớp sai nên $\mathrm{w}_t^T \mathrm{x}_i < 0$
  - vì $y_i = 1$ nên $y_i \|\mathrm{x}_i\|_2^2 = \|\mathrm{x}_i\|_2^2 \geq 1$

do đó, $\mathrm{w}_{t+1}^T \mathrm{x}_i > \mathrm{w}_t^T \mathrm{x}_i$

w_{t+1} tiến về phía làm cho x_i được phân lớp đúng

Xây dựng hàm mất mát

- Đến đây, cảm nhận của chúng ta với thuật toán này là:
  - cứ chọn đường boundary đi.
  - Xét từng điểm một, nếu điểm đó bị misclassified thì tiến đường boundary về phía làm cho điểm đó được classifed đúng.
- Có thể thấy rằng, khi di chuyển đường boundary này:
  - các điểm trước đó được classified đúng có thể lại bị misclassified.
  - Mặc dù vậy, ta sẽ tìm được đường phẳng phân chia hai lớp, miễn là hai lớp đó là linearly separable.

Thuật toán Perceptron

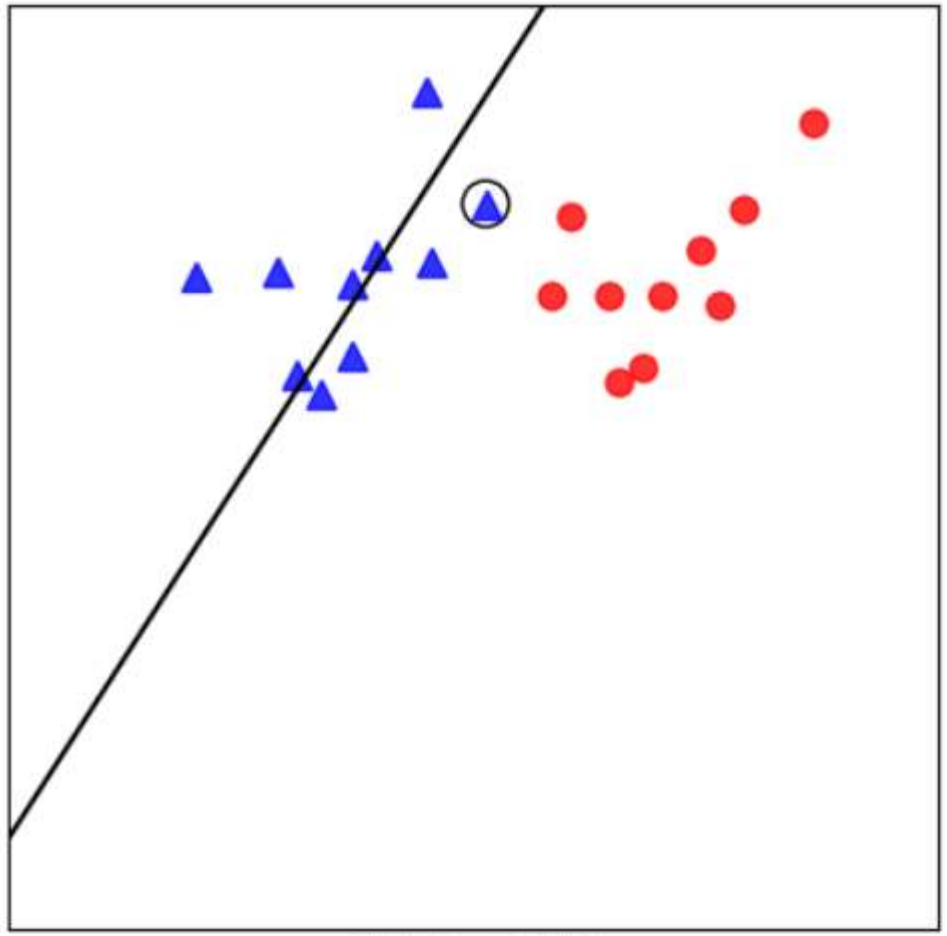
1. Chọn ngẫu nhiên một vector hệ số $\mathbf{w}$ với các phần tử gần 0.
2. Duyệt ngẫu nhiên qua từng điểm dữ liệu $\mathbf{x}_i$:
   - Nếu $\mathbf{x}_i$ được phân lớp đúng, tức $\mathrm{sgn}(\mathbf{w}^T\mathbf{x}_i) = y_i$, chúng ta không cần làm gì.
   - Nếu $\mathbf{x}_i$ bị misclassifed, cập nhật $\mathbf{w}$ theo công thức:

$$\mathbf{w} = \mathbf{w} + y_i\mathbf{x}_i$$

3. Kiểm tra xem có bao nhiêu điểm bị misclassifed. Nếu không còn điểm nào, dừng thuật toán. Nếu còn, quay lại bước 2.

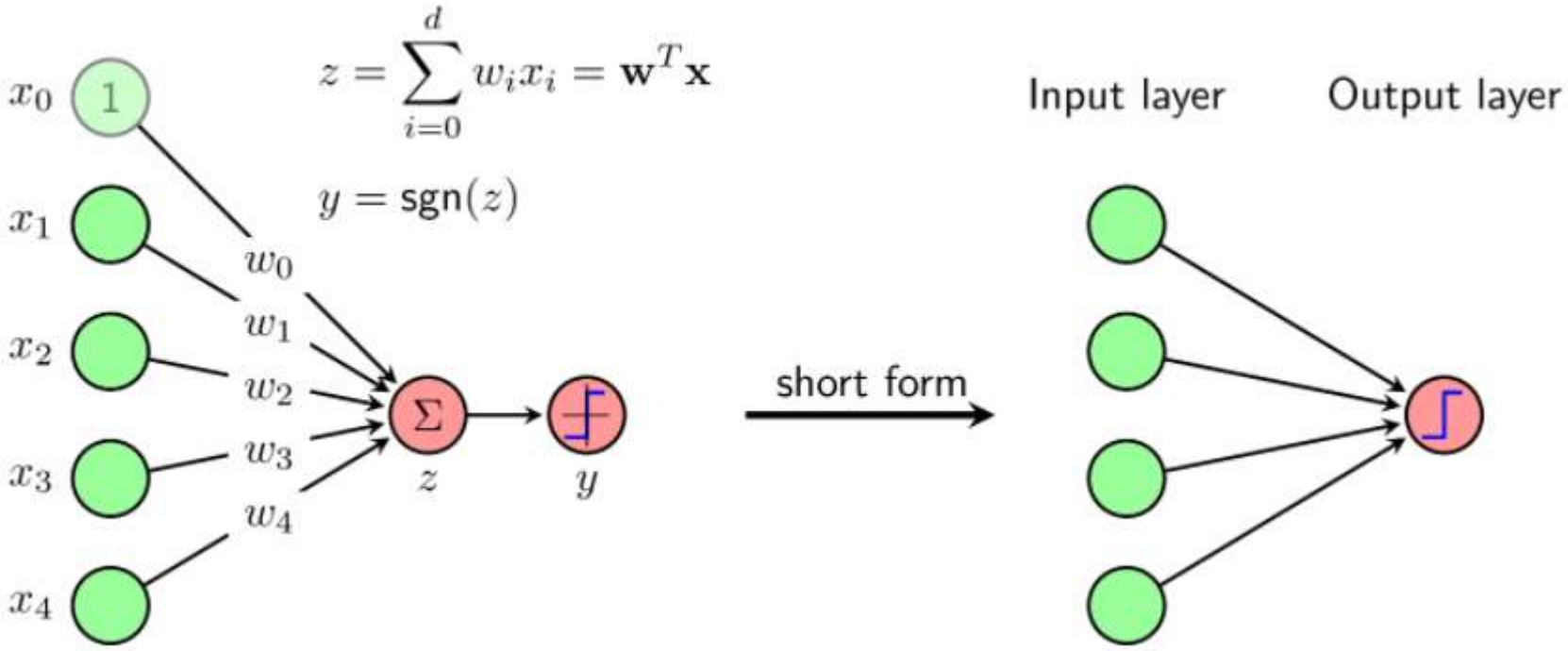
Ví dụ trên Python

- Ta sẽ tạo hai nhóm dữ liệu
  - mỗi nhóm có 10 điểm,
  - mỗi điểm dữ liệu có hai chiều



Mô hình Neural Network đầu tiên

- Hàm số xác định class của Perceptron label(x)=sgn($w^T x$) có thể được mô tả như hình vẽ (được gọi là network) dưới đây:

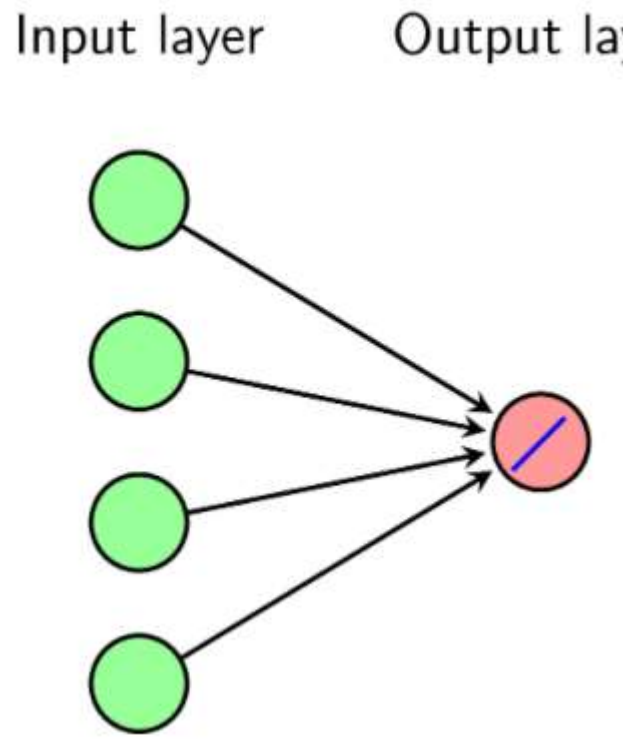


Mô hình Neural Network đầu tiên

- Trong thuật toán PLA: ta phải tìm các weights trên các mũi tên sao cho với mỗi x_i ở tập các điểm dữ liệu đã biết được đặt ở Input layer, output của network này trùng với nhãn y_i tương ứng
- Hàm số y=sgn(z) còn được gọi là *activation function*. Đây chính là dạng đơn giản nhất của Neural Network.

Mô hình Neural Network đầu tiên

- Nếu ta thay *activation function* bởi y=zy=z, ta sẽ có Neural Network mô tả thuật toán Linear Regression



Mô hình Neural Network đầu tiên

- Mô hình perceptron ở trên khá giống với một node nhỏ của dây thân kinh sinh học