



CSE: Faculty of Computer Science and Engineering

Thuyloi University

Hồi quy logistic (Logistic regression)

TS. Nguyễn Thị Kim Ngân

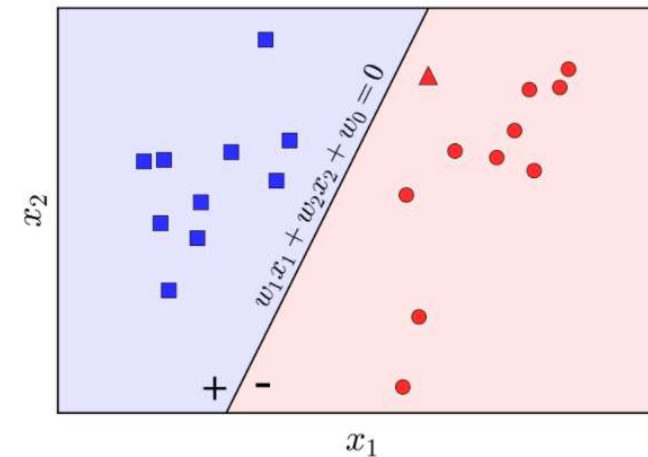
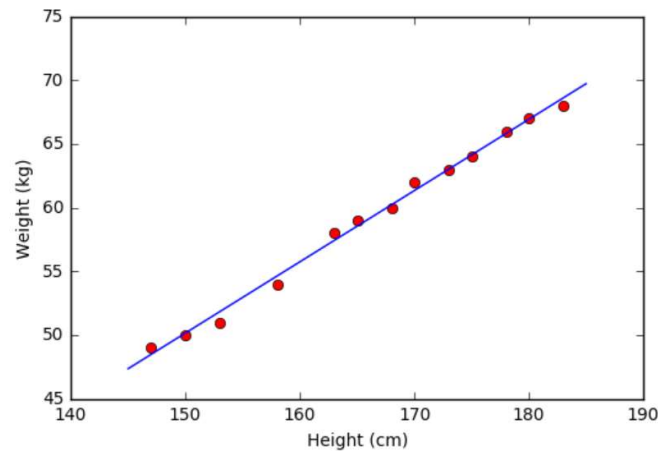
Giới thiệu

perceptron : tìm siêu phẳng
cart, id3: xây dựng cây quyết định

- Hai mô hình tuyến tính: Hồi quy tuyến tính (Linear Regression) và Thuật toán Perceptron (Perceptron Learning Algorithm, PLA) đều có chung một dạng:

$$y = f(\mathbf{w}^T \mathbf{x})$$

- trong đó $f(\mathbf{w}^T \mathbf{x})$ là hàm kích hoạt (*activation function*)
- Với linear regression thì $f(\mathbf{w}^T \mathbf{x}) = \mathbf{w}^T \mathbf{x}$, với PLA thì $f(\mathbf{w}^T \mathbf{x}) = \text{sgn}(\mathbf{w}^T \mathbf{x})$



$f(x) = \text{âm vô cùng} \rightarrow$ xác suất thuộc miền đỏ là cao

$f(x)=0 \rightarrow$ xác suất thuộc miền đỏ là thấp



Giới thiệu

- Hồi quy tuyến tính sử dụng tích vô hướng $\mathbf{w}^T \mathbf{x}$ để dự đoán output y . Phương pháp này phù hợp với bài toán mà output y là một số thực không bị chặn trên và dưới
- PLA, đầu ra chỉ nhận một trong hai giá trị 1 hoặc -1 , phù hợp với các bài toán *binary classification*.



Giới thiệu

- Trong phần này, ta sẽ giới thiệu mô hình có tên là *logistic regression*:
 - Đầu ra có thể được thể hiện dưới dạng xác suất (probability). Ví dụ: xác suất thi đỗ nếu biết thời gian ôn thi, xác suất ngày mai có mưa dựa trên những thông tin đo được trong ngày hôm nay,...
 - *Logistic regression*:
 - giống với **linear regression** ở khía cạnh **đầu ra là số thực**,
 - và giống với **PLA** ở việc đầu ra **bị chặn (trong đoạn $[0,1]$)**.
 - Mặc dù trong tên có chứa từ *regression*, logistic regression thường được sử dụng nhiều hơn cho các bài toán classification.

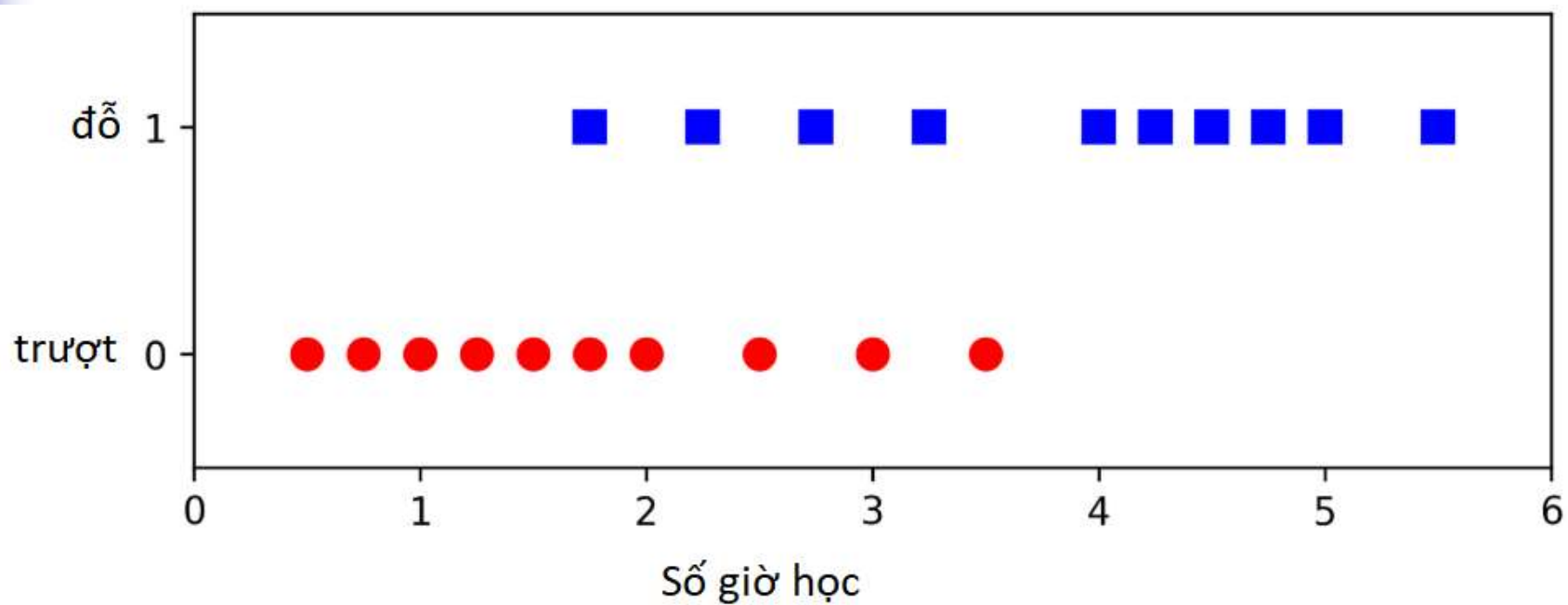


Giới thiệu

Ví dụ: Một nhóm 20 sinh viên dành thời gian trong khoảng từ 0 đến 6 giờ cho việc ôn thi và có kết quả như bảng. Hỏi rằng, em sinh viên ôn x giờ là đỗ hay trượt.
Tập huấn luyện

| Hours | Pass | Hours | Pass |
|-------|------|-------|------|
| .5 | 0 | 2.75 | 1 |
| .75 | 0 | 3 | 0 |
| 1 | 0 | 3.25 | 1 |
| 1.25 | 0 | 3.5 | 0 |
| 1.5 | 0 | 4 | 1 |
| 1.75 | 0 | 4.25 | 1 |
| 1.75 | 1 | 4.5 | 1 |
| 2 | 0 | 4.75 | 1 |
| 2.25 | 1 | 5 | 1 |
| 2.5 | 0 | 5.5 | 1 |

Giới thiệu



một mô hình *flexible* hơn.

ta cần



Mô hình Logistic Regression

Đầu ra dự đoán của:

- Linear Regression:

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

- PLA:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \mathbf{x})$$

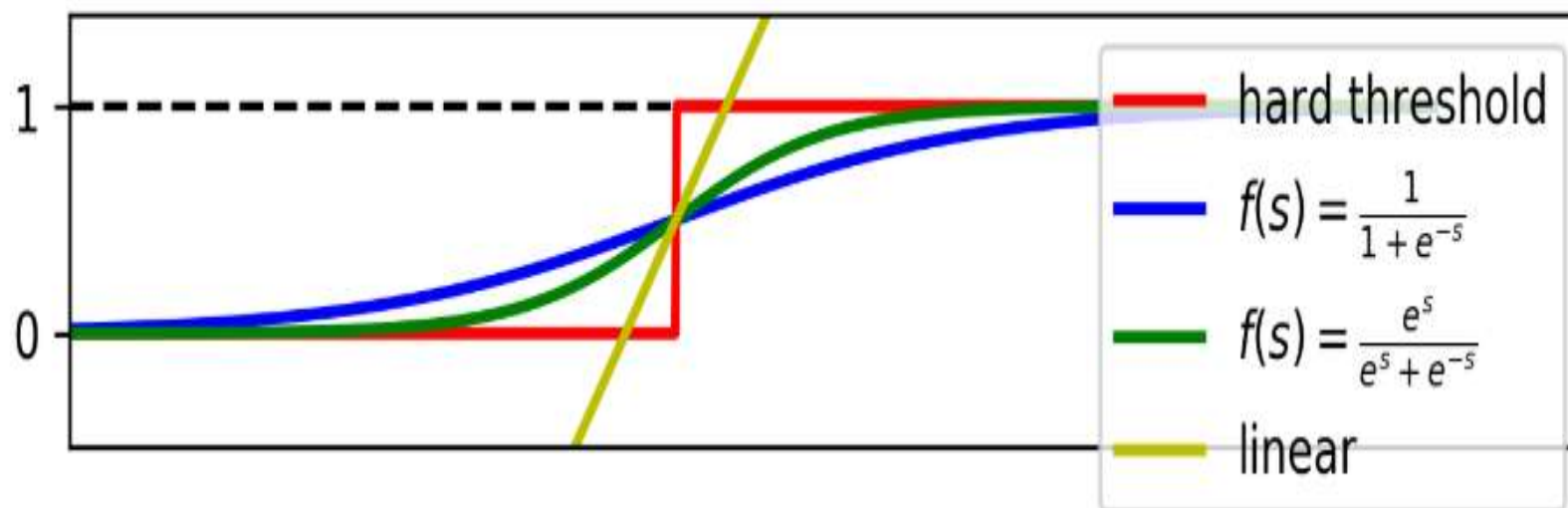
- Logistic regression

$$f(\mathbf{x}) = \theta(\mathbf{w}^T \mathbf{x})$$

tìm tenta sao cho () càng tiến về dương thì $f(x)=1$
ngc lại thì $f(x)=0$

Mô hình Logistic Regression

- Một số hàm kích hoạt cho mô hình tuyến tính:





Mô hình Logistic Regression

- Các đường xanh biển và xanh lá phù hợp với bài toán của chúng ta hơn vì:
 - Là hàm số liên tục nhận giá trị thực, bị chặn trong khoảng $(0,1)$.
 - Nếu coi điểm có tung độ là $\frac{1}{2}$ làm điểm phân chia thì các điểm càng xa điểm này về phía bên trái có giá trị càng gần 0. Ngược lại, các điểm càng xa điểm này về phía phải có giá trị càng gần 1. Điều này *khớp* với nhận xét rằng học càng nhiều thì xác suất đỗ càng cao và ngược lại.
 - *Mượt* (smooth) nên có đạo hàm mọi nơi, có thể được lợi trong việc tối ưu.



Mô hình Logistic Regression

- Sigmoid function

$$f(s) = \frac{1}{1 + e^{-s}} \triangleq \sigma(s)$$

- Hàm này được sử dụng nhiều nhất vì:

- nó bị chặn trong khoảng (0,1)

- $\lim_{s \rightarrow -\infty} \sigma(s) = 0; \lim_{s \rightarrow +\infty} \sigma(s) = 1$

- Có đạo hàm đơn giản

$$\sigma'(s) = \sigma(s)(1 - \sigma(s))$$



Hàm mất mát

Xây dựng hàm mất mát

- Với mô hình (các activation màu xanh biển và lá), ta có thể giả sử rằng xác suất để một điểm dữ liệu \mathbf{x} rơi vào class 1 là $f(\mathbf{w}^T \mathbf{x})$ và rơi vào class 0 là $1 - f(\mathbf{w}^T \mathbf{x})$.
- Với mô hình được giả sử này, với một điểm dữ liệu training (đã biết đầu ra y), ta có thể viết như sau:

xác suất của mẫu xi thuộc vào lớp 1

$$P(y_i = 1 | \mathbf{x}_i; \mathbf{w}) = f(\mathbf{w}^T \mathbf{x}_i)$$
$$P(y_i = 0 | \mathbf{x}_i; \mathbf{w}) = 1 - f(\mathbf{w}^T \mathbf{x}_i)$$

- Mục đích là tìm các hệ số \mathbf{w} sao cho $f(\mathbf{w}^T \mathbf{x}_i)$ càng gần với 1 càng tốt với các điểm dữ liệu thuộc class 1 và càng gần với 0 càng tốt với những điểm thuộc class 0.



Hàm mất mát

$$P(y_i = 1|\mathbf{x}_i; \mathbf{w}) = f(\mathbf{w}^T \mathbf{x}_i)$$

$$P(y_i = 0|\mathbf{x}_i; \mathbf{w}) = 1 - f(\mathbf{w}^T \mathbf{x}_i)$$

- Ký hiệu $z_i = f(\mathbf{w}^T \mathbf{x}_i)$ và viết gộp lại hai biểu thức

ta có: công thức tính xác suất để 1 mẫu xi đc gán nhãn là y_i

$$P(y_i|\mathbf{x}_i; \mathbf{w}) = z_i^{y_i} (1 - z_i)^{1-y_i}$$

y chỉ nhận 2 nhãn và 0 và 1

- Biểu thức ở dưới tương đương với hai biểu thức ở trên vì:
 - khi $y_i = 1$, phần thứ hai của vế phải sẽ triệt tiêu,
 - khi $y_i = 0$, phần thứ nhất sẽ bị triệt tiêu!
- Chúng ta muốn mô hình gần với dữ liệu đã cho nhất, tức xác suất $P(y_i|\mathbf{x}_i; \mathbf{w})$ đạt giá trị cao nhất.



Hàm mất mát

- Xét toàn bộ training set với $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbf{R}^{d \times N}$ và $y = [y_1, y_2, \dots, y_N]$, chúng ta cần tìm \mathbf{w} để $P(\mathbf{y}|\mathbf{X}; \mathbf{w})$ đạt giá trị lớn nhất hay
$$\mathbf{w} = \arg \max_{\mathbf{w}} P(\mathbf{y}|\mathbf{X}; \mathbf{w})$$
- Bài toán tìm tham số để mô hình gần với dữ liệu nhất trên đây có tên gọi chung là bài toán *maximum likelihood estimation* với hàm số phía sau argmax được gọi là *likelihood function*.



Hàm mất mát

- Giả sử các điểm dữ liệu được sinh ra một cách ngẫu nhiên độc lập với nhau (independent), ta có thể viết:

muốn có giá trị max

$$P(\mathbf{y}|\mathbf{X}; \mathbf{w}) = \prod_{i=1}^N P(y_i|\mathbf{x}_i; \mathbf{w})$$
$$= \prod_{i=1}^N z_i^{y_i} (1 - z_i)^{1-y_i}$$

- Trực tiếp tối ưu hàm số này theo \mathbf{w} khó



Hàm mất mát

$$\begin{aligned} P(\mathbf{y}|\mathbf{X}; \mathbf{w}) &= \prod_{i=1}^N P(y_i|\mathbf{x}_i; \mathbf{w}) \\ &= \prod_{i=1}^N z_i^{y_i} (1 - z_i)^{1-y_i} \end{aligned}$$

- Một phương pháp thường được sử dụng đó là lấy logarit tự nhiên (cơ số e) của *likelihood function* biến phép nhân thành phép cộng và để tránh việc số quá nhỏ.
- Sau đó lấy ngược dấu để được một hàm và coi nó là hàm mất mát. Lúc này bài toán tìm giá trị lớn nhất (maximum likelihood) trở thành bài toán tìm giá trị nhỏ nhất của hàm mất mát (hàm này còn được gọi là negative log likelihood):

$$\begin{aligned} J(\mathbf{w}) &= -\log P(\mathbf{y}|\mathbf{X}; \mathbf{w}) && \text{tìm } \mathbf{w} \text{ để hàm có giá trị nhỏ nhất} \\ &= -\sum_{i=1}^N (y_i \log z_i + (1 - y_i) \log(1 - z_i)) \end{aligned}$$



Dự đoán lớp của điểm x

- Sau khi có w , ta dự đoán nhận

$$\hat{y} = \text{sigmoid}(w_0 + w_1x_1 + w_2x_2 + \dots + w_dx_d)$$

- Với bài toán phân lớp:

Nếu $\hat{y} > 0.5$ thì x thuộc về lớp 1,
ngược lại, x thuộc về lớp 0

$$f(s) = \frac{1}{1 + e^{-s}} \triangleq \sigma(s)$$



Tối ưu hàm mất mát

- Chúng ta lại sử dụng phương pháp GD để tìm \mathbf{w}
- Công thức cập nhật (theo thuật toán GD) cho logistic regression là:

$$\mathbf{w} = \mathbf{w} + \eta(y_i - z_i)\mathbf{x}_i$$

sd mẫu bị gán nhãn sai để cập nhật \mathbf{w}



Ví dụ với Python

- Chav chương trình với dữ liệu huấn luyện:

| Hours | Pass | Hours | Pass |
|-------|------|-------|------|
| .5 | 0 | 2.75 | 1 |
| .75 | 0 | 3 | 0 |
| 1 | 0 | 3.25 | 1 |
| 1.25 | 0 | 3.5 | 0 |
| 1.5 | 0 | 4 | 1 |
| 1.75 | 0 | 4.25 | 1 |
| 1.75 | 1 | 4.5 | 1 |
| 2 | 0 | 4.75 | 1 |
| 2.25 | 1 | 5 | 1 |
| 2.5 | 0 | 5.5 | 1 |