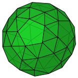
Linear Programming for Data Science

Lê Tiến Hợp, Trần Bảo Minh
Nguyễn Mai Anh Thư, Nguyễn Thị Bảo Tiên

PiMA 2024



Trình bày: Nhóm 2

Ngày 29 tháng 6 năm 2025

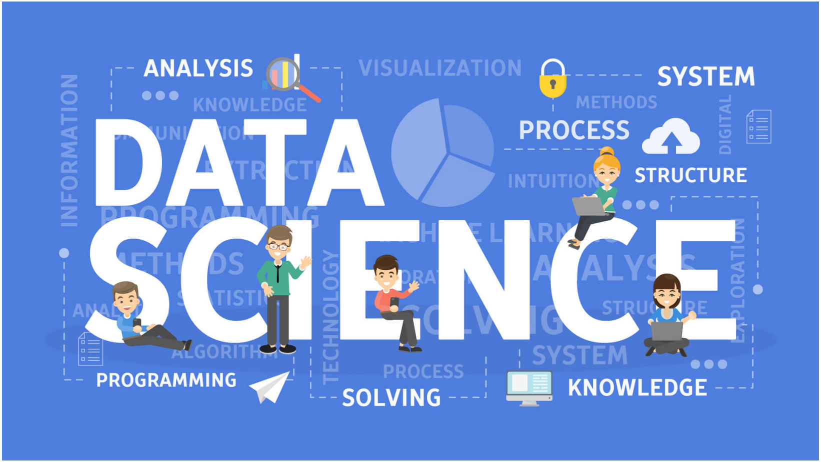
Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Data Science



Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

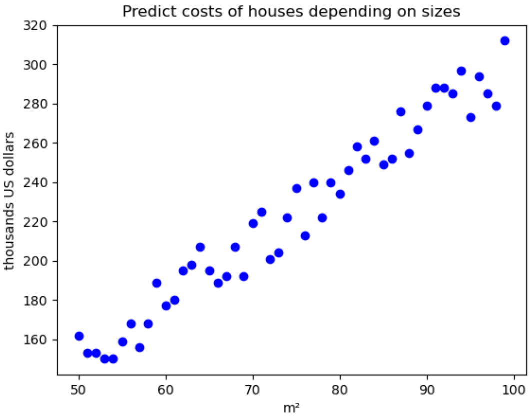
3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Bài toán ví dụ

Xét bài toán ước lượng giá nhà của một căn nhà có diện tích x m^2.

Bài toán ví dụ:

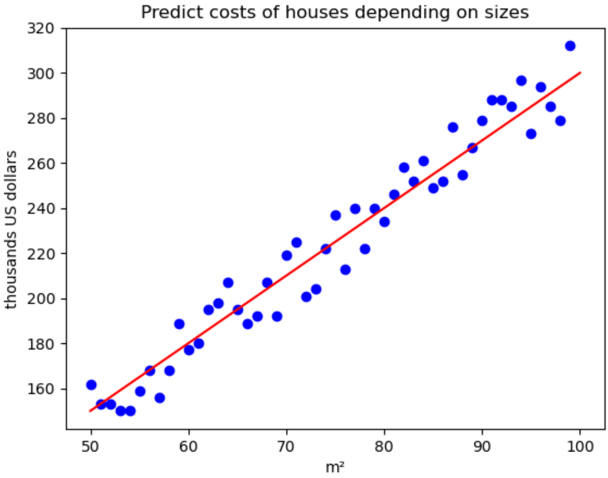
Giả sử ta đã thu thập được số liệu từ 100 căn nhà trong một thành phố.



Đặt vấn đề

Liệu rằng khi có một căn nhà mới xây có dữ liệu về thông tin như trên thì ta có thể dự đoán giá y của căn nhà đó không? Nếu có thì kết quả dự đoán giá nhà $\hat{y} = f(x)$ sẽ được biểu diễn như thế nào?

Nhận xét


Predict costs of houses depending on sizes
thousands US dollars
m²
320
300
280
260
240
220
200
180
160
50
60
70
80
90
100

Nhận xét

- Ta dự đoán đường thẳng biểu diễn mối quan hệ giữa giá nhà và diện tích là $y = w_1 x + w_0$.
- Từ đó, ta tìm cách tìm w_1 và w_0 để sai số so với các điểm dữ liệu là thấp nhất có thể.

Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Ta tổng quát Linear Regression với d biến đầu vào và có biến tự do:

$$y = w_0 + w_1 x_n + ... + w_d x_d = x^T w,$$

trong đó: $w = [w_0, w_1, ..., w_d]^T$ và $x = [1, x_1, x_2, ..., x_d]^T$.

Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Nhắc lại

Ta nhắc lại bài toán Linear Programming như sau:
Với
$A_{ub} \in \mathbb{R}^{m_{ub} \times n}, b_{ub} \in \mathbb{R}_{m_{ub}}, A_{eq} \in \mathbb{R}^{m_{eq} \times n}, b_{eq} \in \mathbb{R}^{m_{eq}}, c, l, u \in \mathbb{R}^n$

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ \text{s.t.} \quad & A_{ub} x \leq b_{ub} \\ & A_{eq} x = b_{eq} \\ & l \leq x \leq u \end{aligned}$$

L^1 Regression

Xét các bài toán với các cặp điểm $(x_i, y_i), i = 1, 2, \cdots, n$.
Ta xây dựng bài toán mất mát cho L^1-regression:

$$\mathcal{L}_1(w) = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - x_i^T w\right|,$$

trong đó: n là số điểm dữ liệu ta thu thập được, y_i là giá trị đầu ra của mỗi điểm, $x_i = [1, x_{i1}, x_{i2}, \cdots, x_{id}]^T$ với x_{ij} là đặc tính thứ j của dữ liệu thứ i.

L^1 Regression

Xét bài toán sau:

$$\begin{aligned} \text{minimize:} \quad & \textstyle\sum_i t_i \\ \text{subject to:} \quad & t_i - \left| y_i - \textstyle\sum_j w_j x_{ij} \right| = 0, i = 1, 2, ..., n \end{aligned}$$

L^1 Regression

Ta làm lỏng các ràng buộc của bài toán gốc ta được bài toán sau:

$$\begin{aligned} \text{minimize:} \quad & \sum_i t_i \\ \text{subject to:} \quad & \left| y_i - \sum_j w_j x_{ij} \right| \le t_i, i = 1, 2, ..., n \end{aligned}$$

Nhận xét: Khử giá trị tuyệt đối

Tại sao có thể chuyển từ bài toán gốc qua bài toán Linear Programming?

Nhận xét: Khử giá trị tuyệt đối

Tại sao có thể chuyển từ bài toán gốc qua bài toán Linear Programming?

- **(1)** Nghiệm tối ưu của bài toán Linear Programming là một nghiệm chấp nhận được của bài toán gốc.
- **(2)** Miền nghiệm của bài toán gốc là con của miền nghiệm của bài toán Linear Programming.

L^1 Regression

Từ **nhận xét (1) và (2)** ta nới lỏng và khử điều kiện giá trị tuyệt đối để chuyển về bài toán sau:

$$\begin{array}{rc} \text{minimize:} & \sum_i t_i \\ \text{subject to:} & -t_i \leq y_i - \sum_j w_j x_{ij} \leq t_i, i = 1, 2, ..., n \end{array}$$

L^1 Regression

Cụ thể, ta có thể biến đổi bài toán về dạng sau:

$$
\begin{aligned}
\text{minimize:} \quad & 0w_0 + 0w_1 + \cdots + 0w_d + t_1 + \cdots + t_n \\
\text{subject to:} \quad & w_0 + w_1x_1 + \cdots + w_dx_{nd} - t_1 \leq y_1 \\
& \vdots \\
& w_0 + w_1x_1 + \cdots + w_dx_{nd} - t_n \leq y_n \\
& -w_0 - w_1x_1 - \cdots - w_dx_{nd} - t_1 \leq -y_1 \\
& \vdots \\
& -w_0 - w_1x_1 - \cdots - w_dx_{nd} - t_n \leq -y_n
\end{aligned}
$$

Đến đây ta có thể bắt đầu giải bài toán L^1 Regression bằng Linear Programming thông qua thư viện scipy.

From scipy to linprog

linprog (**c**, **A_ub**=None, **b_ub**=None, **A_eq**=None, **b_eq**=None, **bounds**=(0, None), **methods**='highs', **callback**=None, **options**=None, $\mathbf{x}_0 = None$, **integrality** $= None$)

From scipy to linprog

$$\mathbf{c} = \begin{bmatrix} (0)_{1\times m} & | & (1)_{1\times n} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

$$\mathbf{b\_ub} = \begin{bmatrix} (y)_{1\times n} & | & -(y)_{1\times n} \end{bmatrix} = \begin{bmatrix} y_1 \cdots y_n - y_1 \cdots - y_n \end{bmatrix}$$

$$\mathbf{A\_ub} = \left[\begin{array}{c|c|c} (1)_{n\times 1} & X_{n\times d} & I_{n\times n} \\ \hline -(1)_{n\times 1} & -X_{n\times d} & -I_{n\times n} \end{array}\right]$$

$$= \begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} & 0 & 0 & \cdots & 1 \\ -1 & -x_{11} & \cdots & -x_{1d} & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & -x_{n1} & \cdots & -x_{nd} & 0 & 0 & \cdots & -1 \end{bmatrix}$$

$$\mathbf{bounds} = (\text{None}, \text{None})$$

L^2 *Regression*

Tương tự với L^1-regression, ta xây dựng hàm mất mát với L^2-regression như sau:

$$\mathcal{L}_2(w) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - x_i^T w\right)^2$$

Cách xử lí bài toán:

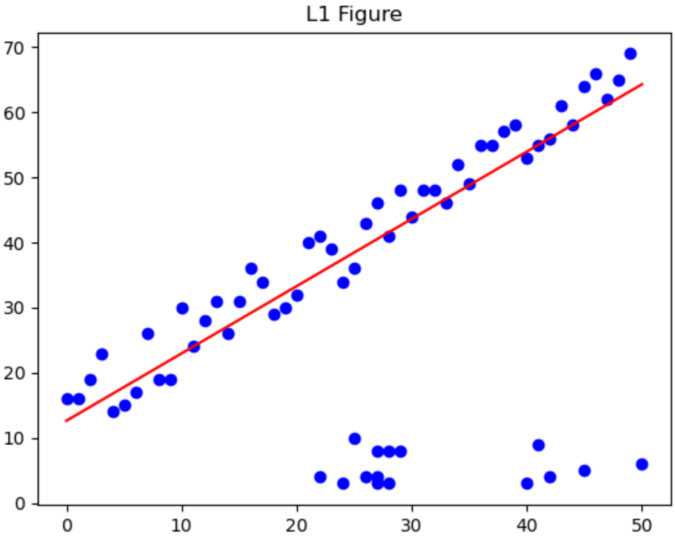
Trước khi giải bài toán trên, ta viết gọn hàm số dưới dạng ma trận, vector và norm:

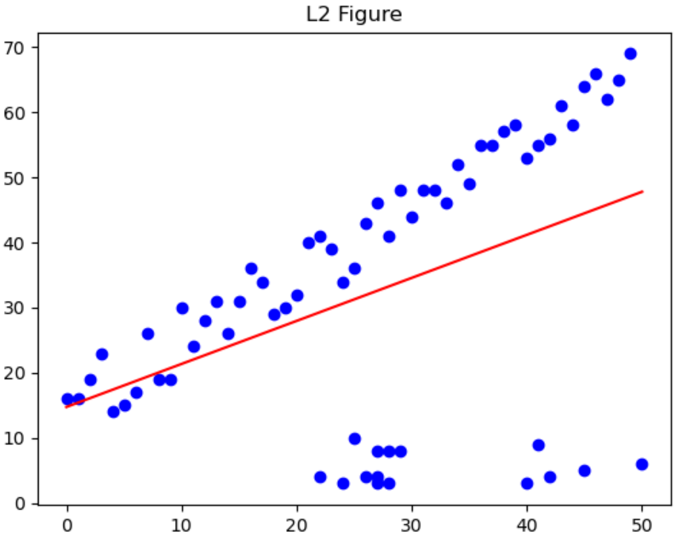
$$\mathcal{L}_2(w) = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i^T w)^2 = \frac{1}{n}\left\| \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} w \right\|^2 = \frac{1}{n}\|y - Xw\|^2$$

với $y = [y_1, y_2, \cdots, y_n]^T, X = [x_1, x_2, \cdots, x_n]^T$.

Cách xử lí bài toán:

$f(w) = \|y - Xw\|^2 = \langle Xw - y, Xw - y \rangle = (Xw - y)^T(Xw - y) = (X^T w^T - y^T)(Xw - y) = X^T w^T Xw - y^T Xw - w^T X^T y + y^T y = w^T(X^T X)xw - (2y^T X)w + y^T y.$
Do $y^T Xw = X^T w^T y$ nên $\nabla f(w) = 2(X^T X)w - 2X^T y$ và $\nabla^2 f(w) = 2(X^T X)$. Hơn nữa,
$h^T \nabla^2 f(w) h = 2[h^T(X^T X)h] = 2[(h^T X^T)(h)] = 2||Xh||^2 \geq 0$ $\forall h \in \mathbb{R}^N$ hay ma trận $\nabla^2 f(w)$ xác định dương.
Hơn nữa, $\text{rank}(X) = n$ nên $\text{rank}(X^T X) = n$, hay $X^T X$ khả nghịch. Vậy nghiệm $w^* = (X^T X)^{-1} X^T y$ của $\nabla f(x) = 0$ là cực tiểu toàn cục của hàm f. Đến đây ta tìm được giá trị tối ưu.


L1 Figure
70
60
50
40
30
20
10
0
0
10
20
30
40
50


L2 Figure
70
60
50
40
30
20
10
0
0
10
20
30
40
50

Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Polynomial Regression

Mô hình hồi quy đa thức có dạng:

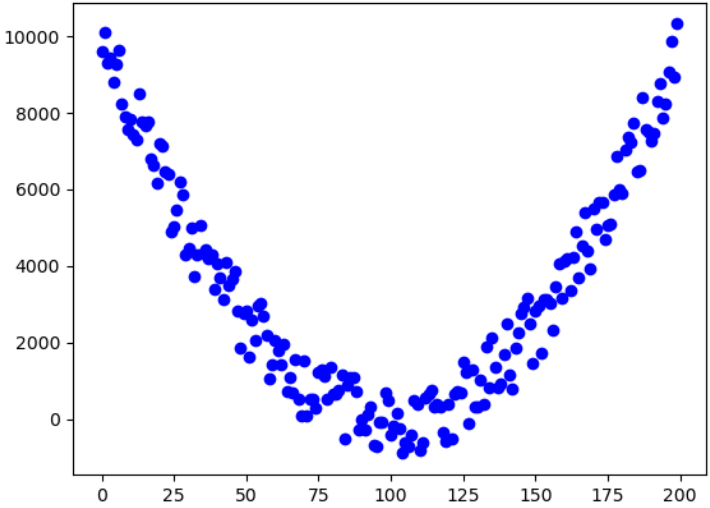
$$\begin{aligned}\hat{y} =& w_0 + w_1 x_i + w_2 {x_i}^2 + \cdots + w_m {x_i}^m, (i = 1, 2, \cdots, n) \\ =& w^T x \\ x =& \begin{bmatrix} 1 & x_1 & \cdots & x_n \end{bmatrix}\end{aligned}$$

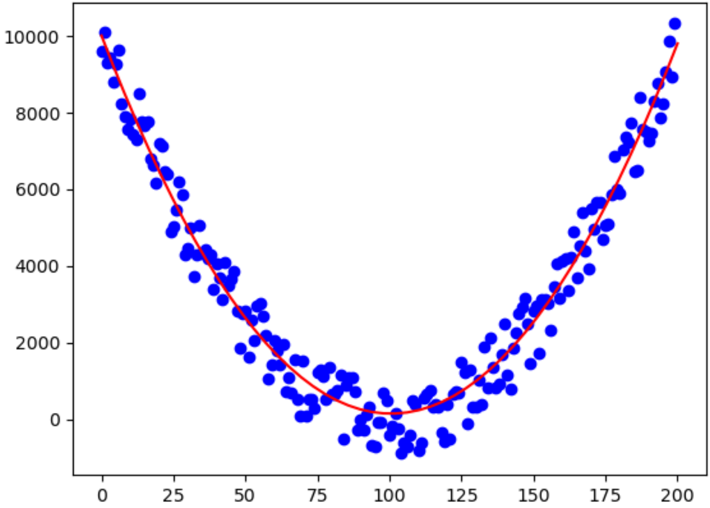
Dễ nhận thấy, (x_i, y_i) là dữ liệu đầu vào. Do đó, ta chỉ đang tìm w bằng việc tối ưu $|\hat{y} - w^T x|$.

Polynomial Regression

$$\mathbf{A\_ub} = \left[\begin{array}{c|c|c} (1)_{n\times 1} & \mathrm{X}_{n\times m} & \mathrm{I}_{n\times n} \\ \hline -(1)_{n\times 1} & -X_{n\times m} & -I_{n\times n} \end{array}\right]$$

$$= \begin{bmatrix} 1 & x_{11} & \cdots & (x_{1m})^m & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 1 & x_{n1} & \cdots & (x_{nm})^m & 0 & 0 & \cdots & 1 \\ -1 & -x_{11} & \cdots & -(x_{1m})^m & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \ddots & \vdots \\ -1 & -x_{n1} & \cdots & -(x_{nm})^m & 0 & 0 & \cdots & -1 \end{bmatrix}$$


10000
8000
6000
4000
2000
0
0
25
50
75
100
125
150
175
200


10000
8000
6000
4000
2000
0
0
25
50
75
100
125
150
175
200

Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Động lực thực tế

Phân lớp dữ liệu

Một số ứng dụng thực tế:

Động lực thực tế

Phân lớp dữ liệu

Một số ứng dụng thực tế:

- Nhận diện khuôn mặt
- Phân loại hình ảnh
- Tin sinh học
- Nhận diện kí tự viết tay
- ...

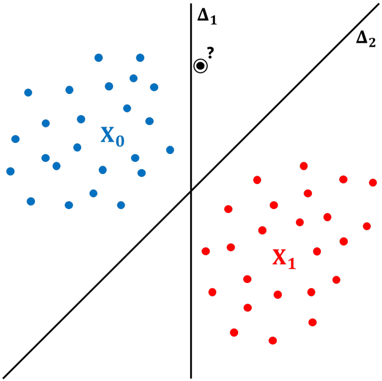
Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

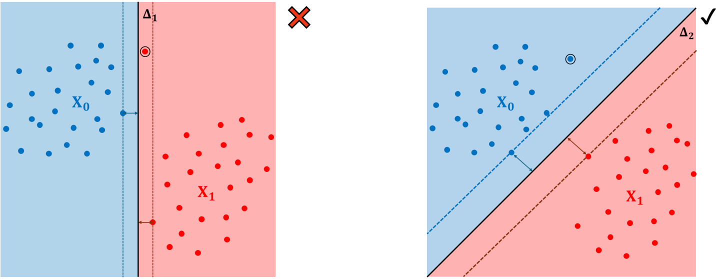
3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Ví dụ



Hình: Có nhiều đường phân chia 2 lớp X_0 và X_1

Đường phân chia nào thực sự tốt hơn ?



Khoảng cách từ các điểm gần nhất của một tập hợp tới mặt phân chia được gọi là **margin**. Để mô hình học cách nhận diện tốt nhất, ta cần tìm mặt phân chia 2 tập hợp dữ liệu tốt nhất, tức để 2 tập hợp dữ liệu cách xa nhau nhất.

Đường phân chia nào thực sự tốt hơn ?

Bài toán Support Vector Machine

Bài toán tối ưu trong SVM là bài toán đi tìm **mặt phân chia tốt nhất** (sao cho margin giữa hai lớp bằng nhau và lớn nhất).

Kiến thức cũ

Cho các cặp dữ liệu $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ với vector $x_i \in \mathbb{R}^d$ thể hiện đầu vào của một điểm dữ liệu và y_i là nhãn của dữ liệu đó.

Khoảng cách từ điểm đó tới mặt phân chia

$$\frac{\left|w^T x_i + b\right|}{\|w\|} = \frac{y_i(w^T x_i + b)}{\|w\|} \quad \forall i = 1, 2, ..., n$$

Mô hình hoá bài toán gốc

Bài toán Support Vector Machine: tìm w và b sao cho margin đạt max

$$
\begin{aligned}
(w, b) &= \arg\max_{w,b} \left\{ \min_i \frac{y_i(w^T x_i + b)}{\|w\|} \right\} \\
&= \arg\max_{w,b} \left\{ \frac{1}{\|w\|} \min_i y_i(w^T x_i + b) \right\}
\end{aligned}
$$

Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Nhận xét bài toán SVM

Gọi $(x_k, y_k) \in \mathbb{R}^d \times \mathbb{R}$ là điểm gần nhất của một lớp tới mặt phân chia. Giả sử $y_k(w^T x_k + b) = a$. Khi ta chia w và b cho a:

- Tử số $y_k(w^T x_k + b) = 1$
- Minimize $\|w\|$ tương tự với việc minimize $\frac{1}{a}\|w\|$
- Siêu phẳng phân chia không thay đổi

Nhận xét bài toán SVM

Vậy nên, không mất tính tổng quát:

$$y_k(w^T x_k + b) = 1$$

với những điểm nằm gần mặt phân chia nhất.

Như vậy, với mọi i ta luôn có:

$$y_i(w^T x_i + b) \geq 1$$

Mô hình hoá bài toán SVM

Bài toán tối ưu có thể viết lại dưới dạng:

$$\begin{aligned} \underset{w,b}{\text{maximize}} \quad & \frac{1}{\|w\|} \\ \text{subject to:} \quad & y_i(w^T x_i + b) \geq 1 \quad \forall i = 1, 2, ..., n \end{aligned}$$

Nói cách khác, ta phải chứng minh:

$$\begin{aligned} \underset{w,b}{\text{minimize}} \quad & \|w\| \\ \text{subject to:} \quad & y_i(w^T x_i + b) \geq 1 \quad \forall i = 1, 2, ..., n \end{aligned}$$

Mô hình hoá bài toán SVM

Để đưa bài toán về dạng Linear Programming, ta có thể sử dụng $L_1 - norm$ để đưa vector $\|w\|$ về dạng tổng của các phần tử $|w_1| + |w_2| + ... + |w_n|$.

Đặt $t_i = \|w_i\| \quad \forall n = 1, 2, ..., d$, bài toán được viết lại dưới dạng:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{d} t_j \\
\text{subject to:} \quad & y_n(w^T x_i + b) \geq 1 \quad && \forall i = 1, 2, ..., n \\
& t_j = |w_j| && \forall j = 1, 2, ..., d
\end{aligned}
$$

Mô hình hoá bài toán gốc

Có vẻ như ràng buộc $t_j = |w_j|$ sẽ gây không ít khó khăn trong quá trình giải. May mắn thay, **nhận xét (1)** đã giúp chúng ta chứng minh được rằng nghiệm tối ưu của bài toán trên cũng là nghiệm tối ưu của bài toán dưới đây:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{d} t_j \\
\text{subject to:} \quad & y_i(w^T x_i + b) \geq 1 \quad && \forall i = 1, 2, ..., n \\
& -t_i \leq w_j \leq t_j && \forall j = 1, 2, ..., d
\end{aligned}$$

Contents

1 Data science

2 Linear Regression
- Động lực
- Tổng quát hóa bài toán
- Các hướng xử lí bài toán Linear Regression
  - L^1 regression
  - L^2 regression
- Mở rộng
  - Polynomial Regression

3 Support Vector Machine
- Động lực
- Xây dựng bài toán tối ưu cho thuật toán Support Vector Machine
- Hướng tiếp cận trong Linear Programing
- Lập trình trong Python

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)
- n : số tấm ảnh trong dataset

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)
- n : số tấm ảnh trong dataset
- x : vector chứa

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)
- n : số tấm ảnh trong dataset
- x : vector chứa
  - $t = [t_1, t_2, ..., t_d]$

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)
- n : số tấm ảnh trong dataset
- x : vector chứa
  - $t = [t_1, t_2, ..., t_d]$
  - $w = [w_1, w_2, ..., w_d]$

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)
- n : số tấm ảnh trong dataset
- x : vector chứa
  - $t = [t_1, t_2, ..., t_d]$
  - $w = [w_1, w_2, ..., w_d]$
  - $w_0 =$ scalar

Lưu ý về kí hiệu

- d : số pixel trong 1 tấm ảnh ($d = 28x28 = 784$)
- n : số tấm ảnh trong dataset
- x : vector chứa
  - $t = [t_1, t_2, ..., t_d]$
  - $w = [w_1, w_2, ..., w_d]$
  - $w_0 = \text{scalar}$

$x = [t_1, t_2, ..., t_d, w_1, w_2, ..., w_d, w_0]$ (vector x có $2d + 1$ phần tử)

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Bài toán hiện có:

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Bài toán hiện có:

Minimize: $t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Bài toán hiện có:

Minimize: $t_1 + t_2 + ... + t_d +$
$0w_1 + 0w_2 + ...0w_d + 0w_0$

Subject to:

$-t_j \leq w_j \leq t_j,\ j = 1, 2, ..., d$

-$y_i(w^T x_i + w_0) \leq -1$,
$i = 1, 2, ..., n$

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Bài toán hiện có:

Minimize: $t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$

Subject to:

$-t_j \leq w_j \leq t_j$, $j = 1, 2, ..., d$

-$y_i(w^T x_i + w_0) \leq -1$, $i = 1, 2, ..., n$

Bài toán gốc:

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Bài toán hiện có:

Minimize: $t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$

Subject to:

$-t_j \leq w_j \leq t_j$, $j = 1, 2, ..., d$

-$y_i(w^T x_i + w_0) \leq -1$, $i = 1, 2, ..., n$

Bài toán gốc:

Minimize: $c^T x$

Đặt vấn đề

Làm sao để đưa về bài toán gốc và sử dụng hàm `linprog`?

Bài toán hiện có:

Minimize: $t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$

Subject to:

$-t_j \leq w_j \leq t_j$, $j = 1, 2, ..., d$

-$y_i(w^T x_i + w_0) \leq -1$, $i = 1, 2, ..., n$

Bài toán gốc:

Minimize: $c^T x$

Subject to:

$Ax \leq b$

`bounds[i][0]` $\leq x_i \leq$ `bounds[i][1]`

Mô hình hoá c^T, A và b

Hàm mục tiêu:
$c^Tx = t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$

Mô hình hoá c^T, A và b

Hàm mục tiêu:
$c^T x = t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$
$\longrightarrow c^T = [1, ..., 1, 0, ..., 0, 0]$ (d số 1 và $d + 1$ số 0)

Mô hình hoá c^T, A và b

Hàm mục tiêu:
$c^T x = t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$
$\longrightarrow c^T = [1, ..., 1, 0, ..., 0, 0]$ (d số 1 và $d + 1$ số 0)

Code: `c = np.concatenate((np.ones(d), np.zeros(d+1)))`

Mô hình hoá c^T, A và b

Hàm mục tiêu:
$c^T x = t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$
$\longrightarrow c^T = [1, ..., 1, 0, ..., 0, 0]$ (d số 1 và $d+1$ số 0)

Code: `c = np.concatenate((np.ones(d), np.zeros(d+1)))`

A = tham số của 3 biến t_j, w_j, w_0 trong constraints

Mô hình hoá c^T, A và b

Hàm mục tiêu:
$c^T x = t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$
$\longrightarrow c^T = [1, ..., 1, 0, ..., 0, 0]$ (d số 1 và $d + 1$ số 0)

Code: `c = np.concatenate((np.ones(d), np.zeros(d+1)))`

A = tham số của 3 biến t_j, w_j, w_0 trong constraints
Chia A ra A_1, A_2, A_3 tương ứng với 3 loại điều kiện:

Mô hình hoá c^T, A và b

Hàm mục tiêu:
$c^T x = t_1 + t_2 + ... + t_d + 0w_1 + 0w_2 + ...0w_d + 0w_0$
$\longrightarrow c^T = [1, ..., 1, 0, ..., 0, 0]$ (d số 1 và $d + 1$ số 0)

Code: `c = np.concatenate((np.ones(d), np.zeros(d+1)))`

A = tham số của 3 biến t_j, w_j, w_0 trong constraints
Chia A ra A_1, A_2, A_3 tương ứng với 3 loại điều kiện:

- $A_1 : -t_j - w_j \leq 0$
- $A_2 : -t_j + w_j \leq 0$
- $A_3 : \left(-\sum_{j=1}^{d} y_i x_{i,j} w_j\right) - y_i w_0 \leq -1$

Tóm tắt ma trận A và vector b

$$
\begin{array}{c}
\begin{array}{ccccccccc} 1 & & & d & d+1 & & & 2d & 2d+1 \end{array}\\
\left[\begin{array}{cccc|cccc|c} t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 \end{array}\right]
\end{array}
$$

$$
\begin{array}{c}
1\\ \\ \\ d\\ d+1\\ \\ \\ 2d\\ 2d+1\\ \\ \\ 2d+n
\end{array}
\left[\begin{array}{cccc|cccc|c}
-1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0\\
0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0\\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\
0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0\\
\hline
-1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0\\
0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0\\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\
0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0\\
\hline
0 & 0 & \cdots & 0 & -y_1x_{1,1} & -y_1x_{1,2} & \cdots & -y_1x_{1,d} & -y_1\\
0 & 0 & \cdots & 0 & -y_2x_{2,1} & -y_2x_{2,2} & \cdots & -y_2x_{2,d} & -y_2\\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\
0 & 0 & \cdots & 0 & -y_nx_{n,1} & -y_nx_{n,2} & \cdots & -y_nx_{n,d} & -y_n
\end{array}\right]
\left[\begin{array}{c}
0\\ 0\\ \vdots\\ 0\\ \hline 0\\ 0\\ \vdots\\ 0\\ \hline -y_1\\ -y_2\\ \vdots\\ -y_n
\end{array}\right]
$$

Tóm tắt ma trận A và vector b

$$A_1 : -t_j - w_j \leq 0$$

$$
\begin{array}{c|cccc|cccc|c|c}
 & 1 & & & d & d+1 & & & 2d & 2d+1 & \\
 & t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 & \\
\hline
1 & -1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0 & 0 \\
\hline
d+1 & -1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
2d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0 & 0 \\
\hline
2d+1 & 0 & 0 & \cdots & 0 & -y_1x_{1,1} & -y_1x_{1,2} & \cdots & -y_1x_{1,d} & -y_1 & -y_1 \\
 & 0 & 0 & \cdots & 0 & -y_2x_{2,1} & -y_2x_{2,2} & \cdots & -y_2x_{2,d} & -y_2 & -y_2 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
2d+n & 0 & 0 & \cdots & 0 & -y_nx_{n,1} & -y_nx_{n,2} & \cdots & -y_nx_{n,d} & -y_n & -y_n
\end{array}
$$

Tóm tắt ma trận A và vector b

$$A_1 : -t_j - w_j \leq 0$$

$$
\begin{array}{c|cccc|cccc|c|c|c}
 & 1 & & & d & d+1 & & & 2d & 2d+1 & & \\
 & t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 & & \\
\hline
1 & -1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 & & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 & & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\
d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0 & & 0 \\
\hline
d+1 & -1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 & & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\
2d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0 & & 0 \\
\hline
2d+1 & 0 & 0 & \cdots & 0 & -y_1x_{1,1} & -y_1x_{1,2} & \cdots & -y_1x_{1,d} & -y_1 & & -y_1 \\
 & 0 & 0 & \cdots & 0 & -y_2x_{2,1} & -y_2x_{2,2} & \cdots & -y_2x_{2,d} & -y_2 & & -y_2 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\
2d+n & 0 & 0 & \cdots & 0 & -y_nx_{n,1} & -y_nx_{n,2} & \cdots & -y_nx_{n,d} & -y_n & & -y_n
\end{array}
$$

Tóm tắt ma trận A và vector b

$$A_2 : -t_j + w_j \leq 0$$

$$
\begin{array}{cc}
 & \begin{array}{cccc:cccc:c} 1 & & & d & d+1 & & & 2d & 2d+1 \end{array} \\
 & \left[\begin{array}{cccc:cccc:c} t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 \end{array}\right] \\
\begin{array}{c} 1 \\ \\ \\ d \\ d+1 \\ \\ \\ 2d \\ 2d+1 \\ \\ \\ 2d+n \end{array} &
\left[\begin{array}{cccc:cccc:c}
-1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 \\
0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0 \\
\hdashline
-1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 \\
0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0 \\
\hdashline
0 & 0 & \cdots & 0 & -y_1x_{1,1} & -y_1x_{1,2} & \cdots & -y_1x_{1,d} & -y_1 \\
0 & 0 & \cdots & 0 & -y_2x_{2,1} & -y_2x_{2,2} & \cdots & -y_2x_{2,d} & -y_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 0 & -y_nx_{n,1} & -y_nx_{n,2} & \cdots & -y_nx_{n,d} & -y_n
\end{array}\right]
\left[\begin{array}{c}
0 \\ 0 \\ \vdots \\ 0 \\ \hdashline 0 \\ 0 \\ \vdots \\ 0 \\ \hdashline -y_1 \\ -y_2 \\ \vdots \\ -y_n
\end{array}\right]
\end{array}
$$

Tóm tắt ma trận A và vector b

$$A_2 : -t_j + w_j \leq 0$$

$$
\begin{array}{c c c}
 & \begin{array}{cccc|cccc|c} 1 & & & d & d+1 & & & 2d & 2d+1 \\ t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 \end{array} & \\
\begin{array}{c} 1 \\ \\ \\ d \\ d+1 \\ \\ \\ 2d \\ 2d+1 \\ \\ \\ 2d+n \end{array} &
\left[\begin{array}{cccc|cccc|c}
-1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 \\
0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0 \\
\hline
-1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 \\
0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0 \\
\hline
0 & 0 & \cdots & 0 & -y_1x_{1,1} & -y_1x_{1,2} & \cdots & -y_1x_{1,d} & -y_1 \\
0 & 0 & \cdots & 0 & -y_2x_{2,1} & -y_2x_{2,2} & \cdots & -y_2x_{2,d} & -y_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 0 & -y_nx_{n,1} & -y_nx_{n,2} & \cdots & -y_nx_{n,d} & -y_n
\end{array}\right]
&
\left[\begin{array}{c}
0 \\ 0 \\ \vdots \\ 0 \\ \hline 0 \\ 0 \\ \vdots \\ 0 \\ \hline -y_1 \\ -y_2 \\ \vdots \\ -y_n
\end{array}\right]
\end{array}
$$

Tóm tắt ma trận A và vector b

$$A_3 : \left(-\sum_{j=1}^{d} y_i x_{i,j} w_j \right) - y_i w_0 \leq -1$$

$$
\begin{array}{c|cccc|cccc|c|c}
 & 1 & & & d & d+1 & & & 2d & 2d+1 & \\
 & t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 & \\
\hline
1 & -1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0 & 0 \\
\hline
d+1 & -1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
2d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0 & 0 \\
\hline
2d+1 & 0 & 0 & \cdots & 0 & -y_1 x_{1,1} & -y_1 x_{1,2} & \cdots & -y_1 x_{1,d} & -y_1 & -y_1 \\
 & 0 & 0 & \cdots & 0 & -y_2 x_{2,1} & -y_2 x_{2,2} & \cdots & -y_2 x_{2,d} & -y_2 & -y_2 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
2d+n & 0 & 0 & \cdots & 0 & -y_n x_{n,1} & -y_n x_{n,2} & \cdots & -y_n x_{n,d} & -y_n & -y_n
\end{array}
$$

Tóm tắt ma trận A và vector b

$$A_3 : \left(-\sum_{j=1}^{d} y_i x_{i,j} w_j \right) - y_i w_0 \leq -1$$

$$
\begin{array}{c|cccc|cccc|c|c}
 & 1 & & & d & d+1 & & & 2d & 2d+1 & \\
 & [\; t_1 & t_2 & \cdots & t_d & w_1 & w_2 & \cdots & w_d & w_0 \;] & \\
\hline
1 & -1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 & 0 & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & -1 & \cdots & 0 & 0 & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 & 0 & 0 \\
\hline
d+1 & -1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
 & 0 & -1 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
2d & 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & 1 & 0 & 0 \\
\hline
2d+1 & 0 & 0 & \cdots & 0 & -y_1 x_{1,1} & -y_1 x_{1,2} & \cdots & -y_1 x_{1,d} & -y_1 & -y_1 \\
 & 0 & 0 & \cdots & 0 & -y_2 x_{2,1} & -y_2 x_{2,2} & \cdots & -y_2 x_{2,d} & -y_2 & -y_2 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
2d+n & 0 & 0 & \cdots & 0 & -y_n x_{n,1} & -y_n x_{n,2} & \cdots & -y_n x_{n,d} & -y_n & -y_n
\end{array}
$$

Tóm tắt ma trận A và vector b

$$
\begin{array}{c c c}
 & \begin{array}{ccc} \times d & \times d & \times 1 \\ \left[\ \mathbf{t} \right. & \mathbf{w} & \left. w_0\ \right] \end{array} & \\
\begin{array}{c} \times d \\ \times d \\ \times n \end{array} & \left[\begin{array}{c|c|c} -\mathbb{I}_d & -\mathbb{I}_d & (0)_d \\ \hline -\mathbb{I}_d & \mathbb{I}_d & (0)_d \\ \hline (0)_{n\times d} & \begin{matrix} -y_1\mathbf{x}_1^T \\ -y_2\mathbf{x}_2^T \\ \vdots \\ -y_n\mathbf{x}_n^T \end{matrix} & -\mathbf{y} \end{array}\right] & \left[\begin{array}{c} (0)_d \\ \hline (0)_d \\ \hline -\mathbf{y} \end{array}\right]
\end{array}
$$

Mô hình hoá A

```
A1 = np.concatenate(
(-np.identity(d), -np.identity(d), np.zeros((d, 1))),
axis=1)
```

Mô hình hoá A

```
A1 = np.concatenate(
(-np.identity(d), -np.identity(d), np.zeros((d, 1))),
axis=1)

A2 = np.concatenate(
(-np.identity(d),np.identity(d),np.zeros((d, 1))),
axis=1)
```

Mô hình hoá A

```
A1 = np.concatenate(
(-np.identity(d), -np.identity(d), np.zeros((d, 1))),
axis=1)

A2 = np.concatenate(
(-np.identity(d),np.identity(d),np.zeros((d, 1))),
axis=1)

A3 = np.concatenate((
    np.zeros((n, d)),
    -label * data,
    -label
), axis=1)
```

Mô hình hoá A

```
A1 = np.concatenate(
(-np.identity(d), -np.identity(d), np.zeros((d, 1))),
axis=1)

A2 = np.concatenate(
(-np.identity(d),np.identity(d),np.zeros((d, 1))),
axis=1)

A3 = np.concatenate((
    np.zeros((n, d)),
    -label * data,
    -label
), axis=1)

A = np.concatenate((A1, A2, A3), axis=0)
```

Mô hình hoá b

$b = [0, ..., 0, -1, ..., -1]$ ($2d$ số 0 và n số -1)

Mô hình hoá b

$b = [0, ..., 0, -1, ..., -1]$ ($2d$ số 0 và n số -1)

```
b = np.concatenate((
        np.zeros((2*d,1)),
        -np.ones((n,1))
    ), axis=0)
```

In kết quả

```
result
```

In kết quả

```
result = linprog(
```

In kết quả

```
result = linprog(
    c=c,
    A_ub=A,
    b_ub=b,
    bounds=(None, None)
)
```

In kết quả

```
result = linprog(
    c=c,
    A_ub=A,
    b_ub=b,
    bounds=(None, None)
)
print(f"Result = {result.x}")
```

In kết quả

```
result = linprog(
    c=c,
    A_ub=A,
    b_ub=b,
    bounds=(None, None)
)
print(f"Result = {result.x}")
```

Diễn giải kết quả (interpretation):

In kết quả

```
result = linprog(
    c=c,
    A_ub=A,
    b_ub=b,
    bounds=(None, None)
)
print(f"Result = {result.x}")
```

Diễn giải kết quả (interpretation):

- t = `result.x[0:d]`
- w = `result.x[d:2*d]`
- w_0 = `result.x[2*d]`

Kết thúc

Xin cảm ơn vì đã lắng nghe!