



TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH
HO CHI MINH CITY UNIVERSITY OF FOREIGN LANGUAGES - INFORMATION TECHNOLOGY

Bài toán phân lớp

Biên soạn: ThS. Vũ Đình Ái (aivd@huflit.edu.vn)

Cập nhật: tháng 09/2023

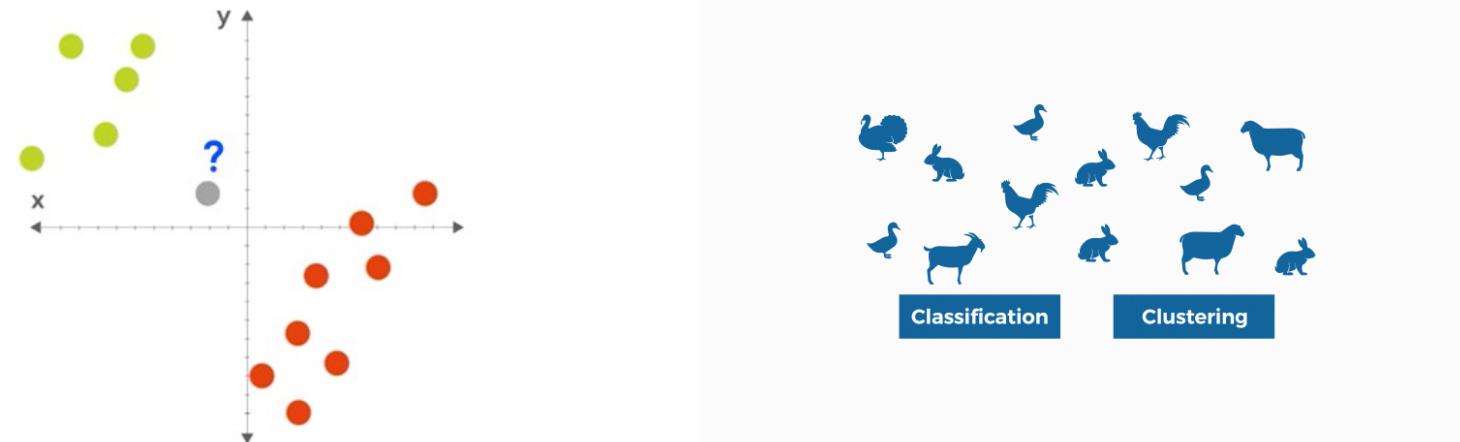
www.huflit.edu.vn

Nội dung

- Giới thiệu bài toán phân lớp
- Thuật toán logistic Regression
- Thuật toán KNN
- Thuật toán cây quyết định
- Thuật toán Naïve Bayes
- Support Vector Machine

Giới thiệu bài toán phân lớp

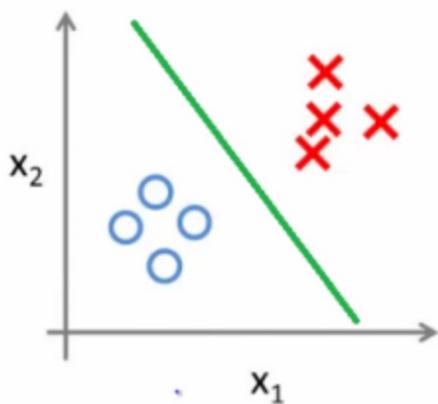
- **Bài toán phân lớp** là quá trình phân lớp 1 đối tượng dữ liệu vào 1 hay nhiều lớp đã cho trước nhờ 1 mô hình phân lớp (model).
 - Mô hình này được xây dựng dựa trên 1 tập dữ liệu được xây dựng trước đó có gán nhãn (hay còn gọi là tập huấn luyện).
 - Quá trình phân lớp là quá trình gán nhãn cho đối tượng dữ liệu.
- **Như vậy, nhiệm vụ của bài toán phân lớp là cần tìm 1 mô hình phân lớp để khi có dữ liệu mới thì có thể xác định được dữ liệu đó thuộc vào phân lớp nào.**



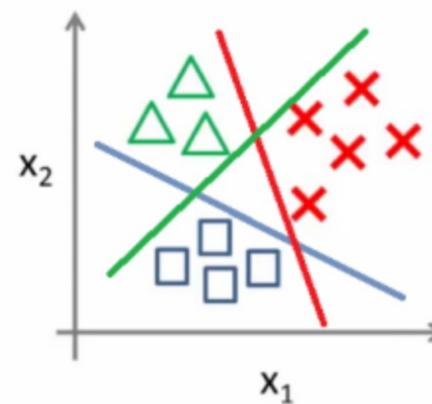
Giới thiệu bài toán phân lớp

- Có nhiều bài toán phân lớp dữ liệu như phân lớp nhị phân (binary classification), phân lớp đa lớp (multiclass classification),
 - **Bài toán phân lớp nhị phân** là bài toán gán nhãn dữ liệu cho đối tượng vào 1 trong 2 lớp khác nhau dựa vào việc dữ liệu đó có hay không có các đặc trưng (feature) của bộ phân lớp.
 - **Bài toán phân lớp đa lớp** là quá trình phân lớp dữ liệu với số lượng lớp lớn hơn 2. Như vậy với từng dữ liệu phải xem xét và phân lớp chúng vào những lớp khác nhau chứ không phải là 2 lớp như bài toán phân lớp nhị phân. Và thực chất bài toán **phân lớp nhị phân** là bài toán **đặt biệt** của **phân lớp đa lớp**.

Binary classification:



Multi-class classification:



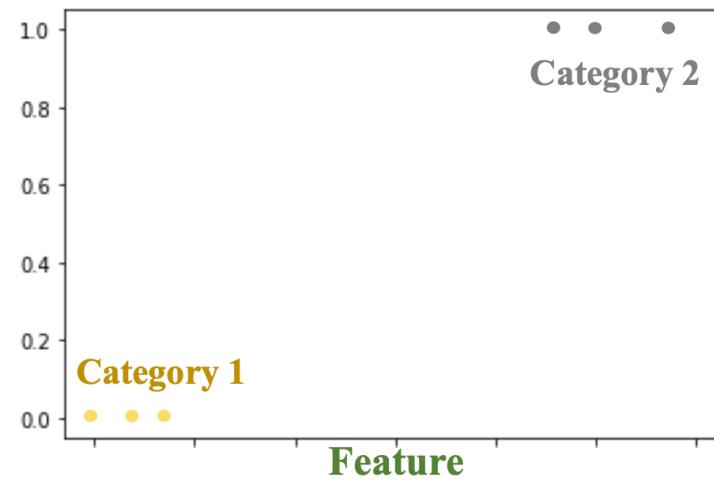
Thuật toán logistic Regression

- **Logistic Regression** cũng là thuật toán máy học thuộc nhóm học có giám sát (Supervised)
- Với bài toán sử dụng thuật toán **Linear Regression** thì đầu ra của bài toán là **dữ liệu liên tục** (dùng để dự đoán) thì bài toán sử dụng **Logistic Regression** với đầu ra là **dữ liệu rời rạc** (dùng để phân lớp/gán nhãn cho dữ liệu chưa được huấn luyện)
- Ví dụ :
 - Giả sử trong trường hợp chúng ta cần phân loại các loài hoa như sau

❖ Given a new kind of data

Feature	Label
Petal Length	Category
1.4	Flower A
1	Flower A
1.5	Flower A
3	Flower B
3.8	Flower B
4.1	Flower B

Plot data



Thuật toán logistic Regression

❖ Given a new kind of data

Feature	Label
Petal_Length	Category
1.4	Flower A
1	Flower A
1.5	Flower A
3	Flower B
3.8	Flower B
4.1	Flower B

Category 1
Category 2

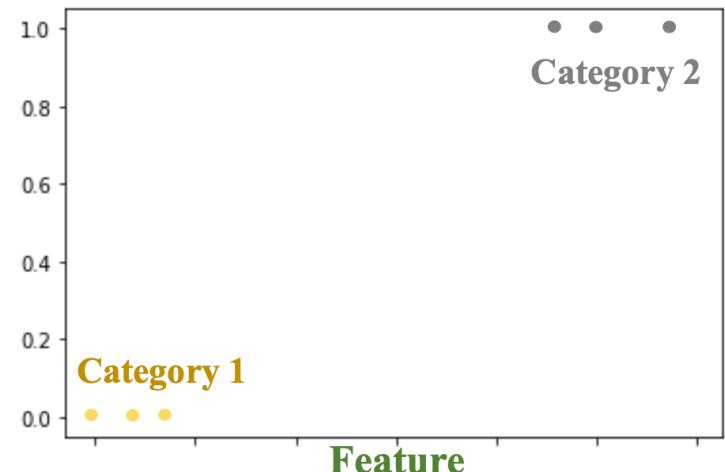
Assign numbers
to categories

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

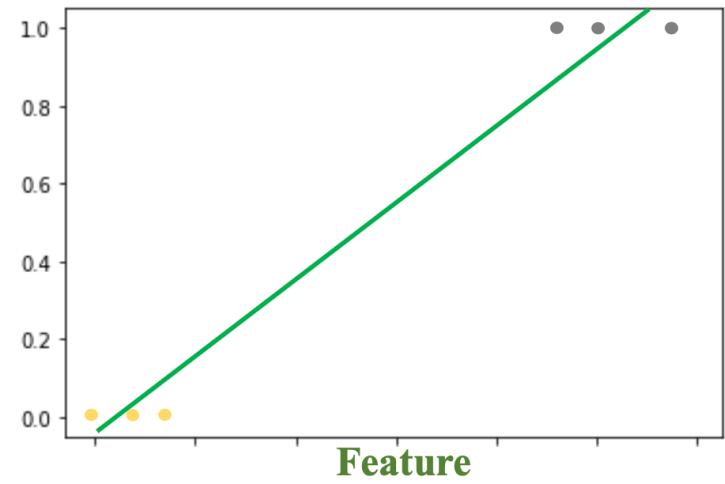
Category 1
Category 2

20

Plot data



A line is not suitable
for this data



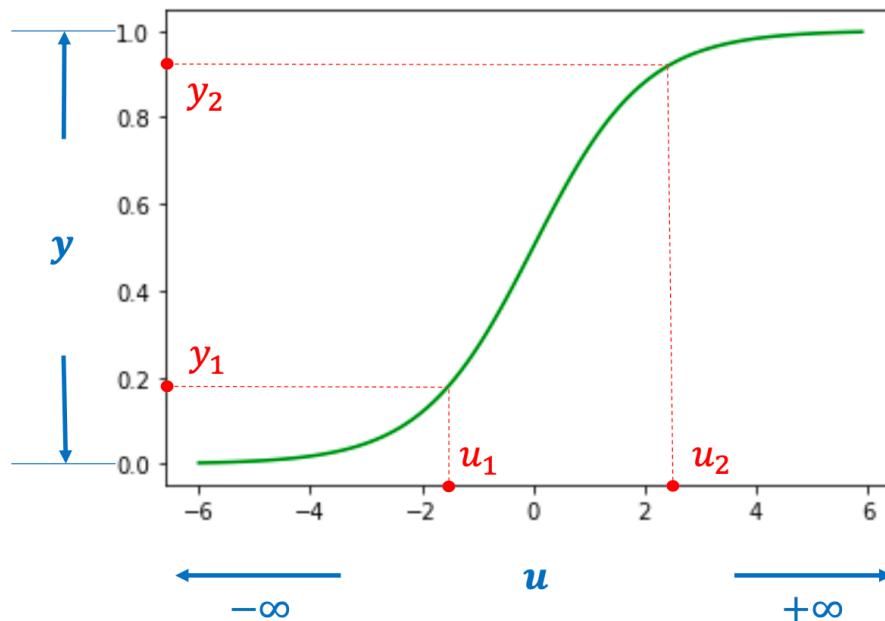
Thuật toán logistic Regression

Sigmoid function

$$y = \sigma(u) = \frac{1}{1 + e^{-u}}$$
$$u \in (-\infty, +\infty)$$
$$y \in (0, 1)$$

Property

$$\forall u_1, u_2 \in [a, b] \text{ và } u_1 \leq u_2 \rightarrow \sigma(u_1) \leq \sigma(u_2)$$



hàm `sigmoid` được sử dụng cho bài toán **Logistic Regression** vì nó bị chặn trong khoảng $[0, 1]$ - thoả mãn bài toán phân loại nhị phân:

Khi $h_{\theta}(x) > 0.5$ dự đoán $y = 1$

Khi $h_{\theta}(x) < 0.5$ dự đoán $y = 0$

Thuật toán logistic Regression

- Chúng ta mong muốn giá trị đầu ra: $0 \leq h_{\theta}(x) \leq 1$ Chính vì vậy mà chúng ta sử dụng một **Activation Function (hàm kích hoạt)** để thay đổi thành hàm phi tuyến.
- Khi đó hàm giả thuyết:

$$h_{\theta}(x) = g(\theta^T x) = g(z) = \frac{1}{1 - e^{-z}}$$

- Trong đó:
 - với $z = \theta_0 + \theta_1 x = \theta^T x$
 - **g: kí hiệu cho Activation Function**

Thuật toán logistic Regression

❖ Given a new kind of data

Feature	Label
Petal_Length	Category
1.4	Flower A
1	Flower A
1.5	Flower A
3	Flower B
3.8	Flower B
4.1	Flower B

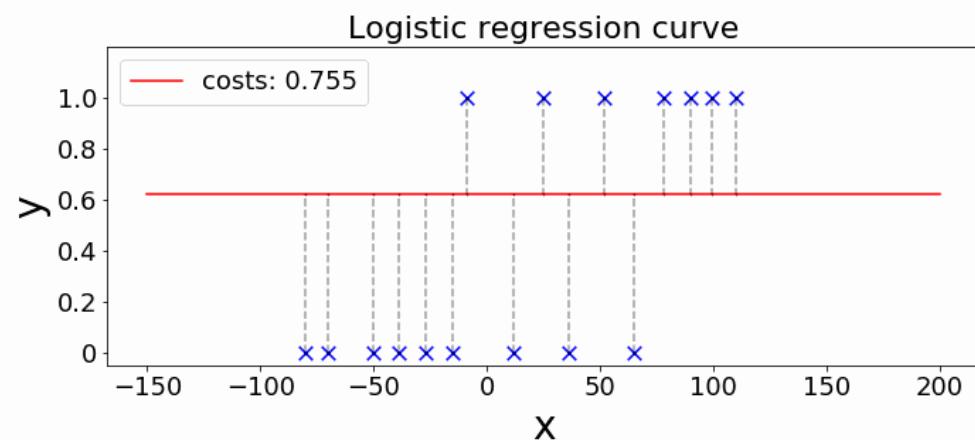
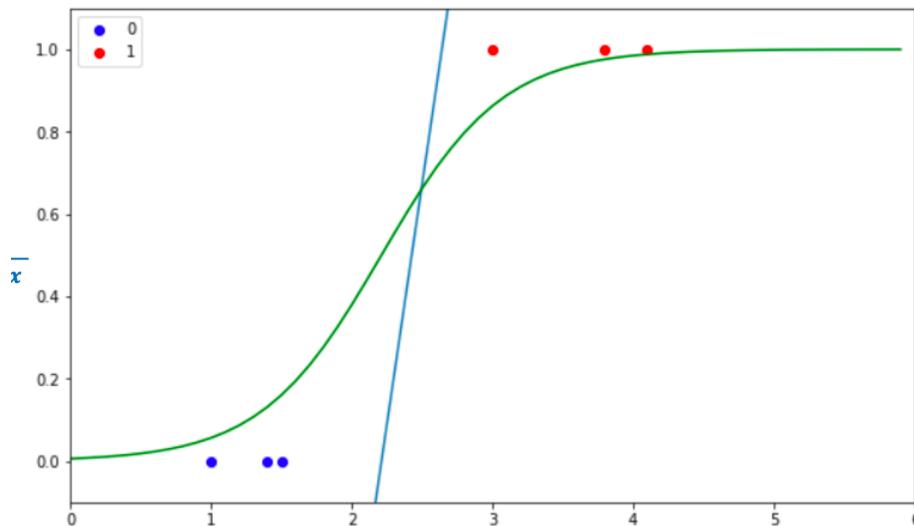
Category 1
Category 2

Assign numbers
to categories

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

Category 1
Category 2

20



Thuật toán logistic Regression

❖ Given a new kind of data

Feature	Label
Petal_Length	Category
1.4	Flower A
1	Flower A
1.5	Flower A
3	Flower B
3.8	Flower B
4.1	Flower B

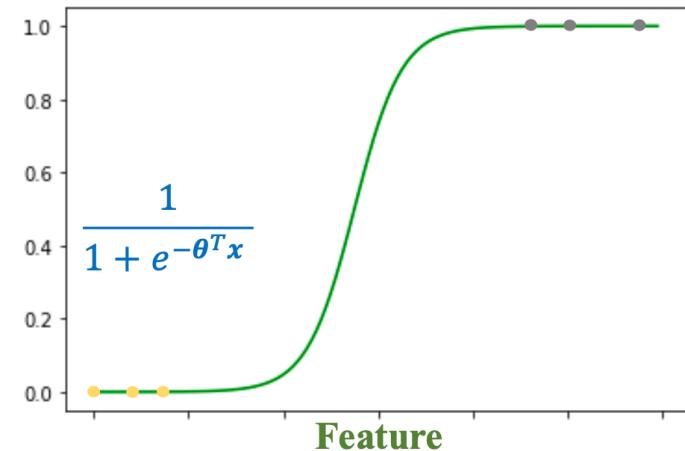
↓ Assign numbers to categories

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

20

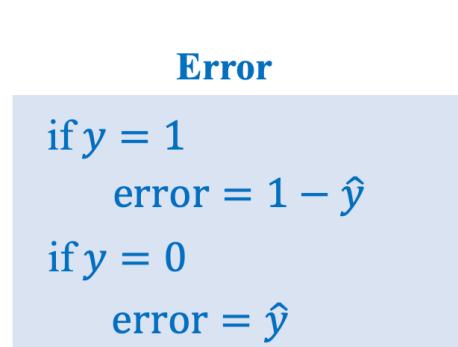
Sigmoid function could fit the data

$$z = \theta^T x$$
$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$
$$\hat{y} \in (0 \quad 1)$$



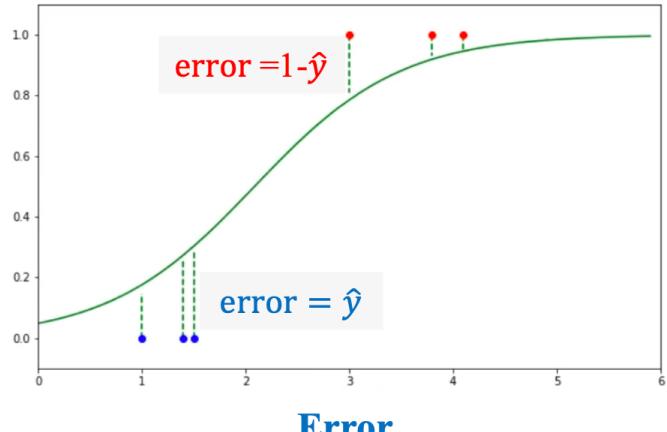
Error

$$\text{if } y = 1 \quad \text{error} = 1 - \hat{y}$$
$$\text{if } y = 0 \quad \text{error} = \hat{y}$$

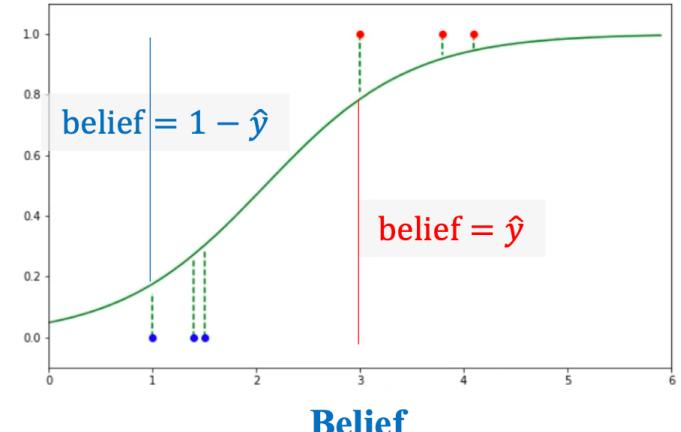


Thuật toán logistic Regression

❖ Construct loss



```
if  $y_i = 1$   
    error =  $1 - \hat{y}_i$   
if  $y_i = 0$   
    error =  $\hat{y}_i$ 
```



```
if  $y_i = 1$   
    belief =  $\hat{y}_i$   
if  $y_i = 0$   
    belief =  $1 - \hat{y}_i$ 
```

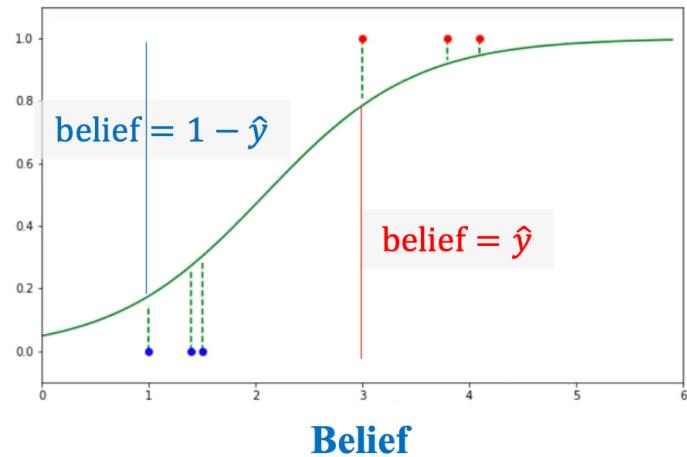
$$P = \hat{y}_i^{y_i} (1 - \hat{y}_i)^{1-y_i}$$

20

Minimize error ~ maximize belief ~ Minimize (-belief)

Thuật toán logistic Regression

❖ Construct loss



if $y_i = 1$
 belief = \hat{y}_i
if $y_i = 0$
 belief = $1 - \hat{y}_i$

$$P_i = \hat{y}_i^{y_i} (1 - \hat{y}_i)^{1-y_i}$$

20

$$\text{belief} = \prod_{i=1}^n P_i \quad \text{since iid}$$

$$\text{log_belief} = \sum_{i=1}^n \log P_i$$

$$\text{log_belief} = \sum_{i=1}^n [y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)]$$

$$\text{loss} = -\text{log_belief}$$

$$= - \sum_{i=1}^n [y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)]$$

$$L = \frac{1}{N} (-y^T \log(\hat{y}) - (1 - y^T) \log(1 - \hat{y}))$$

Binary cross-entropy

Thuật toán logistic Regression

▪ Hàm mất mát

$$L(\theta) = -\frac{1}{n} \left[\sum_{i=1}^n y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \right]$$

▪ Trong đó:

- y : giá trị thật (phân lớp)
- \hat{y} : giá trị dự đoán (phân lớp)
- θ : trọng số
- n : tổng số lượng dữ liệu đầu vào

Thuật toán logistic Regression

- Áp dụng Gradient Descent cho hàm mất mát để tối ưu cho kết quả thấp nhất.

$$z = \theta^T x$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

$$L = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

Model and Loss

$\frac{\partial L}{\partial \theta_i}$	$\frac{\partial L}{\partial \hat{y}}$	$\frac{\partial \hat{y}}{\partial z}$	$\frac{\partial z}{\partial \theta_i}$	Derivative
$= \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z} \frac{\partial z}{\partial \theta_i}$	$= -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}} = \frac{\hat{y}-y}{\hat{y}(1-\hat{y})}$	$= \hat{y}(1-\hat{y})$	$= x_i$	$= x_i(\hat{y}-y)$

- Cập nhật giá trị θ

$$\theta = \theta - \eta L'_{\theta}$$

- Trong đó:

- η : tốc độ học (LR)
- θ : trọng số

Thuật toán logistic Regression - SGD

1) Pick a sample (x, y) from training data

2) Compute output \hat{y}

$$z = \theta^T x$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

3) Compute loss

$$L(\theta) = (-y \log \hat{y} - (1-y) \log(1-\hat{y}))$$

4) Compute derivative

$$L'_{\theta} = x(\hat{y} - y)$$

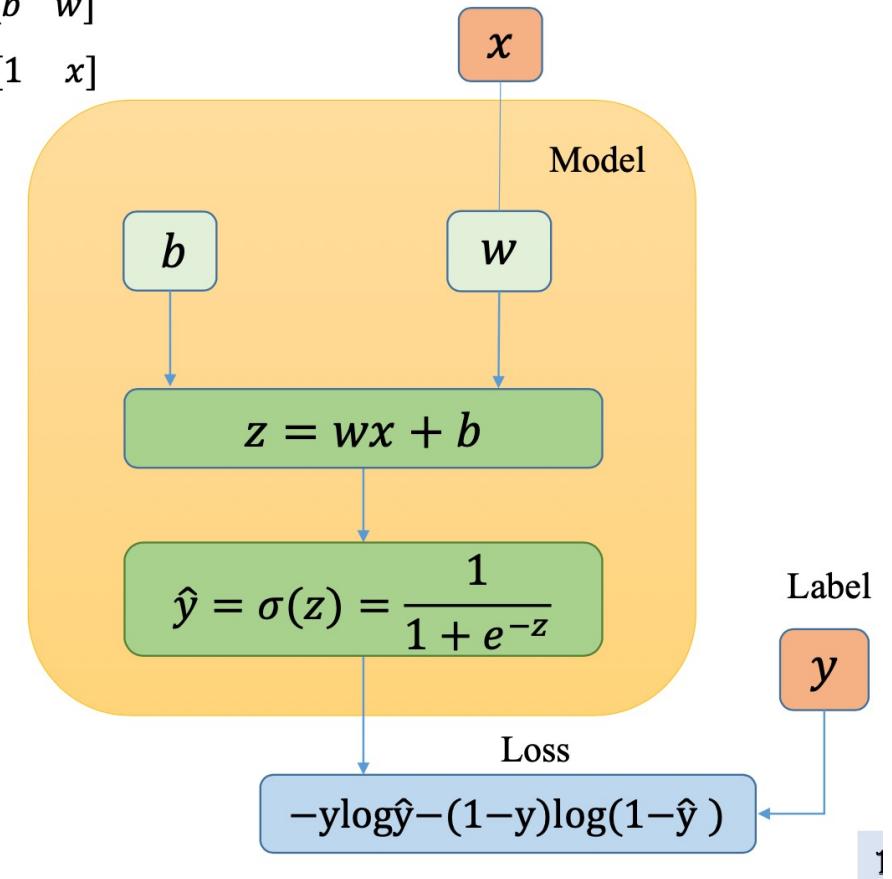
5) Update parameters

$$\theta = \theta - \eta L'_{\theta}$$

η is learning rate

$$\theta^T = [b \quad w]$$

$$x^T = [1 \quad x]$$



Thuật toán logistic Regression - SGD

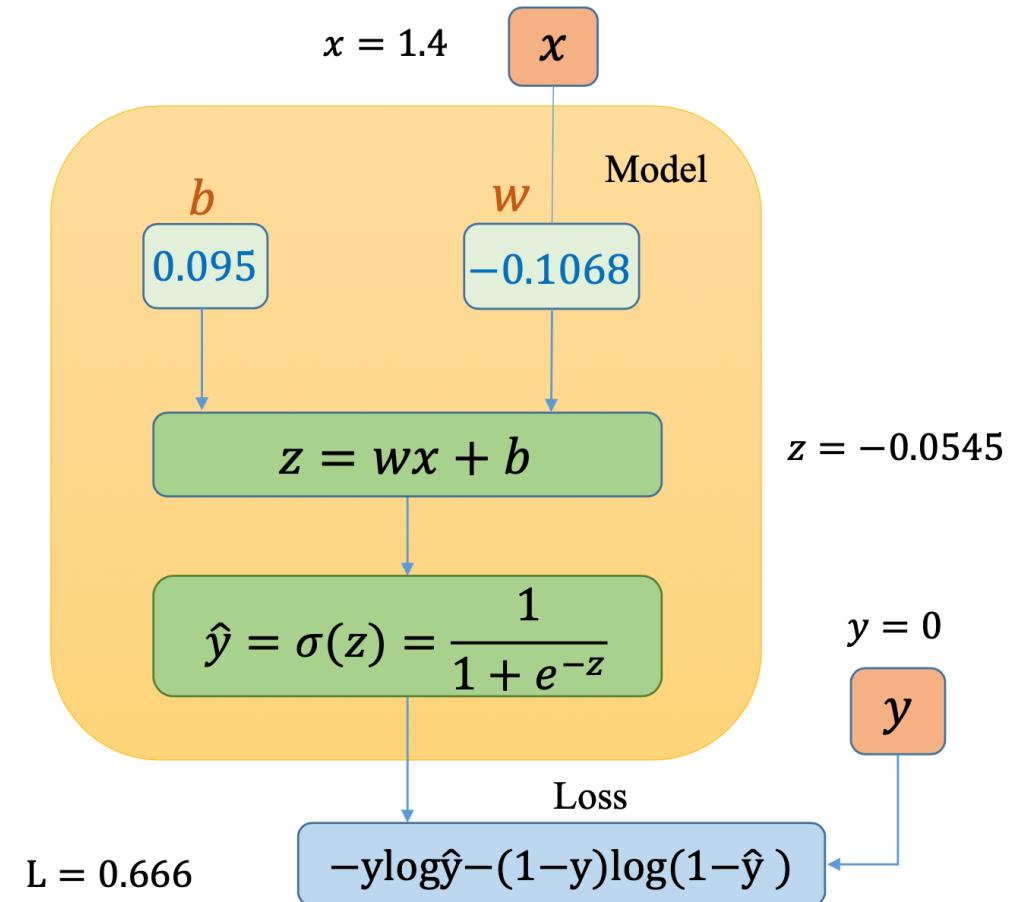
Dataset

Petal_Length	Label
1.4	0
1.5	0
3	1
4.1	1

$$\mathbf{x} = \begin{bmatrix} 1 \\ 1.4 \end{bmatrix} \quad \mathbf{y} = [0]$$

$$\hat{y} = 0.486$$

previous L = 0.6733



Thuật toán logistic Regression - SGD

■ Stochastic Gradient Descent

1) Pick a sample (x, y) from training data

2) Tính output \hat{y}

$$z = \theta^T x$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

3) Tính loss

$$L(\theta) = (-y \log \hat{y} - (1-y) \log(1-\hat{y}))$$

4) Tính đạo hàm

$$L'_{\theta} = x(\hat{y} - y)$$

5) Cập nhật tham số

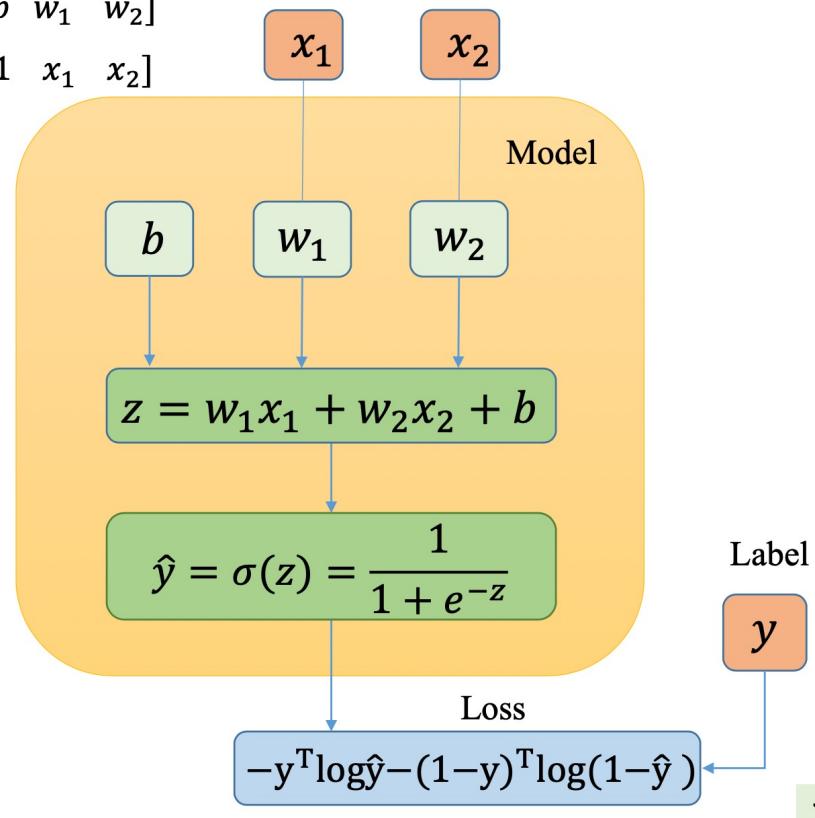
$$\theta = \theta - \eta L'_{\theta}$$

η is learning rate

ear 2020

$$\theta^T = [b \ w_1 \ w_2]$$

$$x^T = [1 \ x_1 \ x_2]$$



Thuật toán logistic Regression - MiniBatch

■ MiniBatch Gradient Descent

1) Pick m samples from training data

2) Tính output \hat{y}

$$z = \theta^T x$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

3) Tính loss

$$L(\theta) = \frac{1}{m} (-y^T \log \hat{y} - (1-y)^T \log(1-\hat{y}))$$

4) Tính đạo hàm

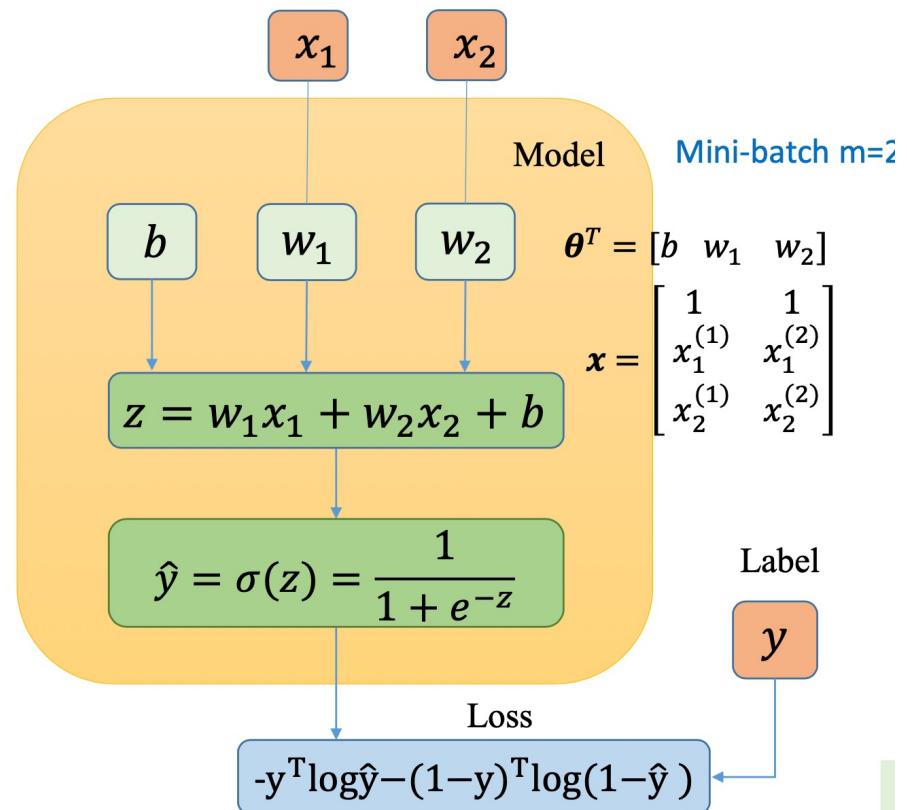
$$L'_{\theta} = \frac{1}{m} x^T (\hat{y} - y)$$

5) Cập nhật tham số

$$\theta = \theta - \eta L'_{\theta}$$

η is learning rate

v 2020



Thuật toán logistic Regression - Batch

▪ Batch Gradient Descent

1) Pick all the samples from training data

2) Tính output \hat{y}

$$z = \boldsymbol{\theta}^T \mathbf{x}$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

3) Tính loss (binary cross-entropy)

$$L(\boldsymbol{\theta}) = \frac{1}{N} (-\mathbf{y}^T \log \hat{y} - (1-\mathbf{y})^T \log(1-\hat{y}))$$

4) Tính đạo hàm

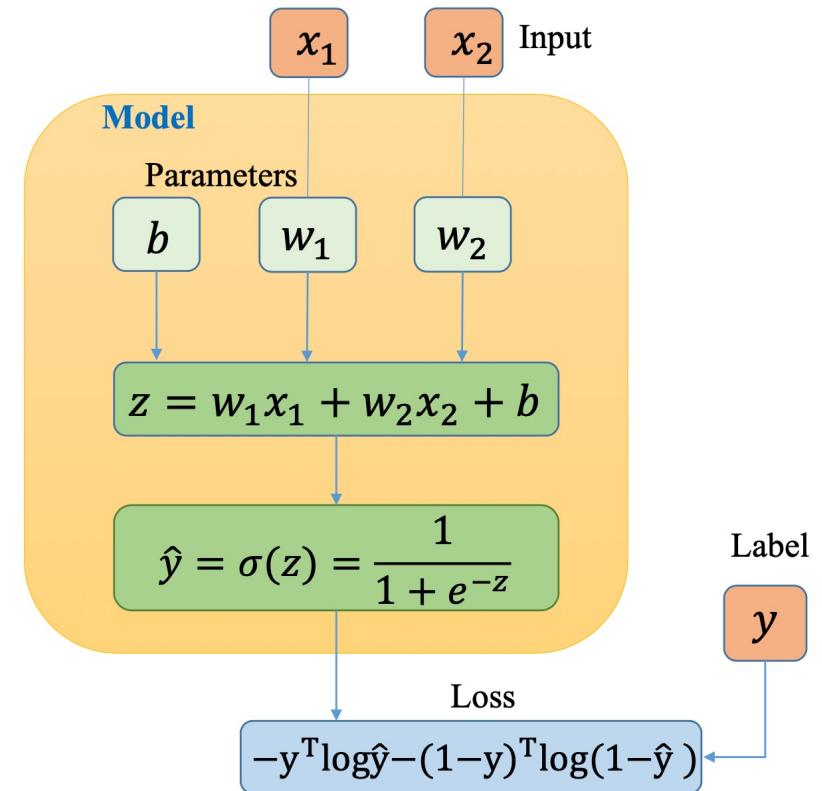
$$L'_{\boldsymbol{\theta}} = \frac{1}{N} \mathbf{x}^T (\hat{y} - \mathbf{y})$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

xx 2020

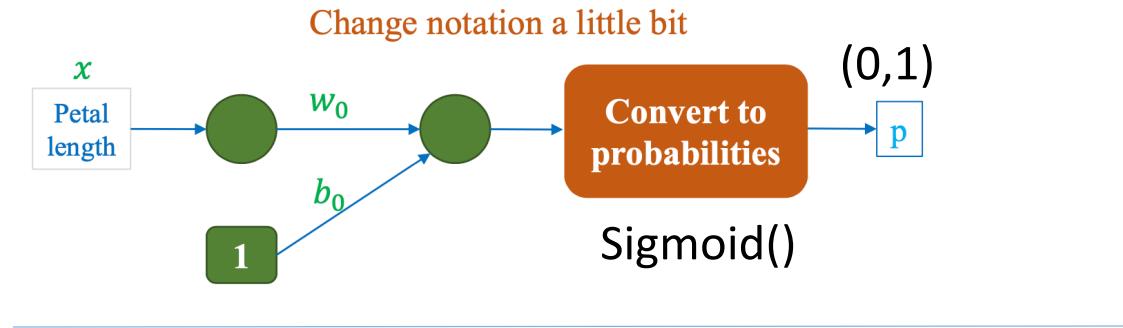


22

Softmax Regression - SGD

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

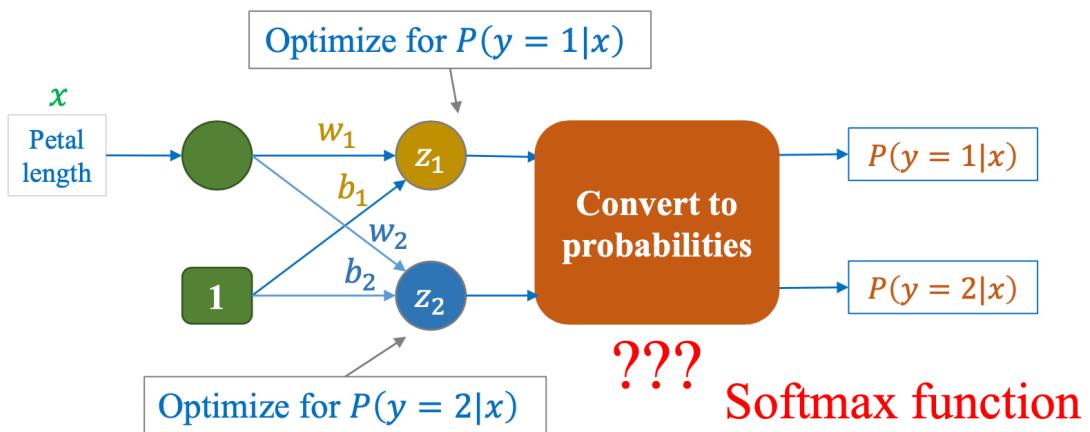
C₀ C₁



Feature	Label
Petal_Length	Label
1.4	1
1.3	1
1.5	1
4.5	2
4.1	2
4.6	2

* Indices is from 1

Explicitly output $P(y = 1|x)$ and $P(y = 2|x)$



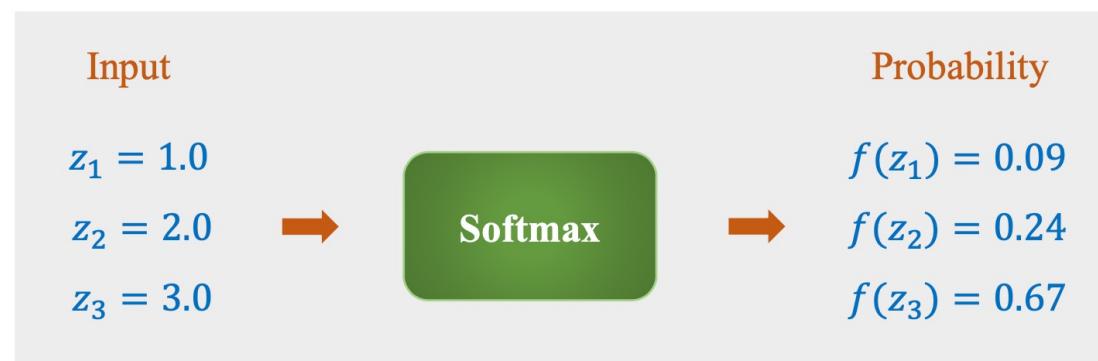
Softmax Regression - SGD

Softmax function

$$P_i = f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

$$\sum_i f(z_i) = 1$$



Softmax Regression - SGD

Feature	Label
Petal_Length	Label
1.4	1
1.3	1
1.5	1
4.5	2
4.1	2
4.6	2

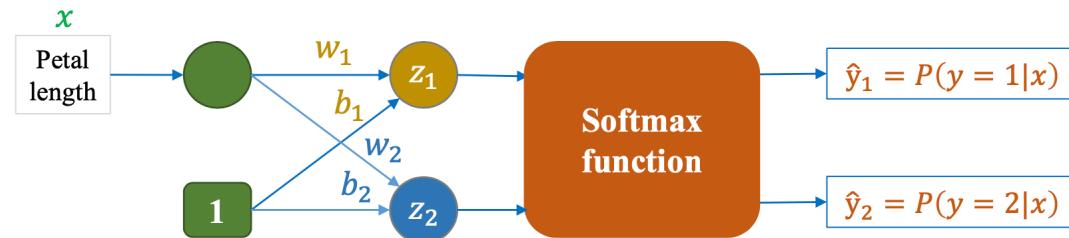
Softmax function

$$P_i = f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

$$\sum_i f(z_i) = 1$$

Explicitly output $P(y = 1|x)$ and $P(y = 0|x)$



How about loss function?

Index from 0

$$L(\theta) = -y \log \hat{y} - (1-y) \log(1-\hat{y})$$

Otherwise

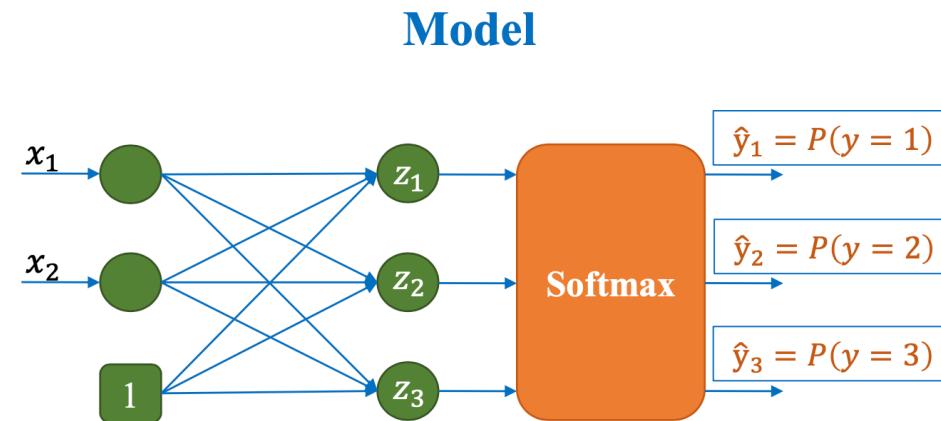
$$L(\theta) = -\delta(y, 1) \log \hat{y}_1 - \delta(y, 2) \log \hat{y}_2$$

$$\delta(i, j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Softmax Regression - SGD

Feature	Label
Petal_Length	Petal_Width
1.5	0.2
1.4	0.2
1.6	0.2
4.7	1.6
3.3	1.1
4.6	1.3
5.6	2.2
5.1	1.5
5.6	1.4

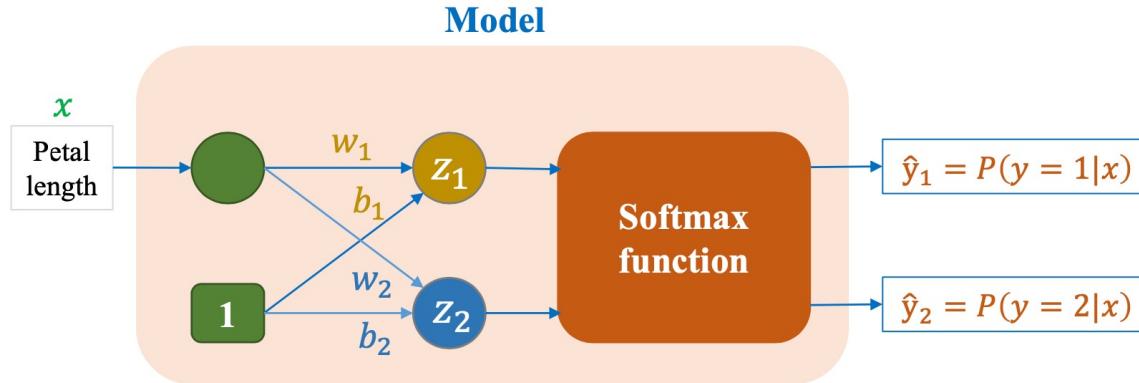
#class=3
#feature=2



Softmax Regression - SGD

❖ Simple illustration

Feature	Label
Petal_Length	Label
1.4	1
1.3	1
1.5	1
4.5	2
4.1	2
4.6	2



$$\delta(i,j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

$$z_1 = xw_1 + b_1$$

$$z_2 = xw_2 + b_2$$

$$\hat{y}_1 = \frac{e^{z_1}}{\sum_{j=1}^2 e^{z_j}}$$

$$\hat{y}_2 = \frac{e^{z_2}}{\sum_{j=1}^2 e^{z_j}}$$

$$\begin{aligned} L(\theta) &= -\delta(i,j)\log\hat{y}_1 - \delta(i,j)\log\hat{y}_2 \\ &= -\sum_{i=1}^2 \delta(i,y)\log\hat{y}_i \end{aligned}$$

Derivative

$$\frac{\partial \hat{y}_i}{\partial z_j} = \hat{y}_i (\delta(i,j) - \hat{y}_i)$$

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - \delta(i,y)$$

Softmax Regression - SGD

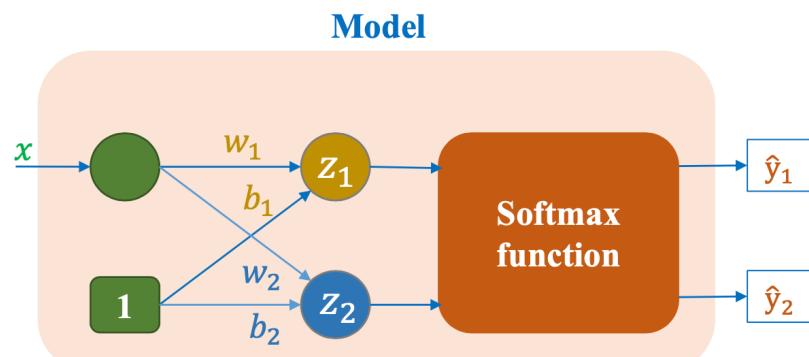
Feature	Label
Petal_Length	Label
1.4	1
1.3	1
1.5	1
4.5	2
4.1	2
4.6	2

* Label indices are from 1

$$\theta = \begin{bmatrix} b_1 & b_2 \\ w_1 & w_2 \end{bmatrix}$$

$$x = \begin{bmatrix} 1 \\ x \end{bmatrix}$$

Input with one example
 $(x, y) = (1.4, 1)$



Forward computation

$$z = \theta^T x$$

$$\hat{y} = \frac{e^z}{\sum_{j=1}^2 e^{z_j}}$$

Loss function

$$L(\theta) = - \sum_{i=1}^2 \delta(i, y) \log \hat{y}_i$$

$$\delta(i, j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Derivative

$$\frac{\partial \hat{y}_i}{\partial z_j} = \hat{y}_i (\delta(i, j) - \hat{y}_i)$$

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - \delta(i, y)$$

$$\frac{\partial L}{\partial w_i} = x (\hat{y}_i - \delta(i, y))$$

$$\frac{\partial L}{\partial b_i} = \hat{y}_i - \delta(i, y)$$

Softmax Regression - SGD

Petal_Length	Petal_Width	Label
1.5	0.2	1
1.4	0.2	1
1.6	0.2	1
4.7	1.6	2
3.3	1.1	2
4.6	1.3	2
5.6	2.2	3
5.1	1.5	3
5.6	1.4	3

#feature n=2

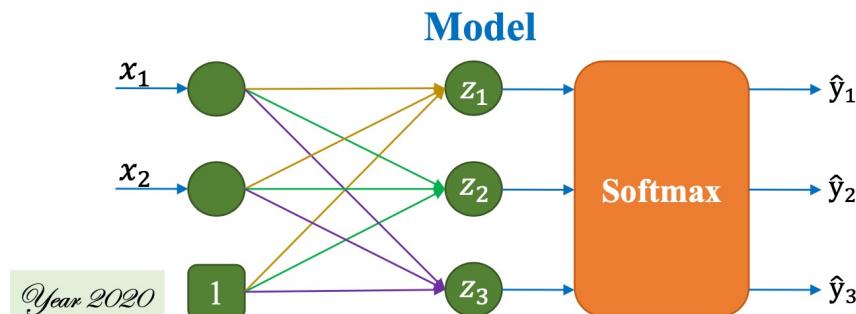
#example m=9

#class k=3

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_1 \ \boldsymbol{\theta}_2 \ \boldsymbol{\theta}_3]$$

$$= \begin{bmatrix} b_1 & b_2 & b_3 \\ w_{11} & w_{21} & w_{31} \\ w_{12} & w_{22} & w_{32} \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \quad x_0 = 1$$



1) Pick a sample (\mathbf{x}, y) from training data

2) Tính output $\hat{\mathbf{y}}$

$$\mathbf{z} = \boldsymbol{\theta}^T \mathbf{x}$$

$$\hat{\mathbf{y}} = \frac{e^{\mathbf{z}}}{\sum_{i=1}^k e^{z_i}}$$

3) Tính loss (cross-entropy)

$$L(\boldsymbol{\theta}) = - \sum_{i=1}^k \delta(i, y) \log \hat{y}_i$$

4) Tính đạo hàm

$$\frac{\partial L}{\partial \boldsymbol{\theta}_i} = \mathbf{x} (\hat{y}_i - \delta(i, y))$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

Softmax Regression - SGD

Update parameters

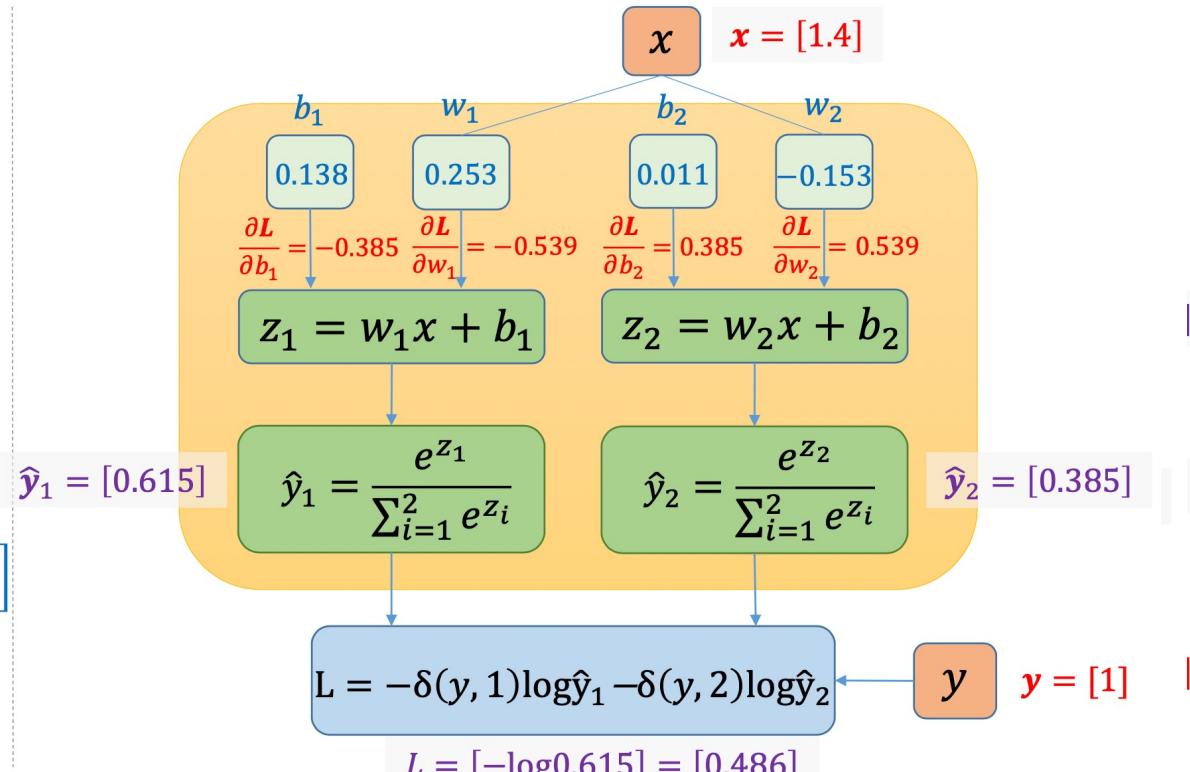
$$\theta = \theta - \eta L'_{\theta}$$

η is learning rate

$$\theta = \begin{bmatrix} b_1 & b_2 \\ w_1 & w_2 \end{bmatrix} \quad L'_{\theta} = \begin{bmatrix} \frac{\partial L}{\partial b_1} & \frac{\partial L}{\partial b_2} \\ \frac{\partial L}{\partial w_1} & \frac{\partial L}{\partial w_2} \end{bmatrix}$$

$$\begin{aligned} \theta &= \begin{bmatrix} 0.1 & 0.05 \\ 0.2 & -0.1 \end{bmatrix} - 0.1 \begin{bmatrix} -0.385 & 0.385 \\ -0.539 & 0.539 \end{bmatrix} \\ &= \begin{bmatrix} 0.138 & 0.011 \\ 0.253 & -0.153 \end{bmatrix} \end{aligned}$$

Year 2020



Thuật toán KNN – K nearest neighbor

- K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning.
- Khi training, thuật toán này không học một điều gì từ dữ liệu training (đây cũng là lý do thuật toán này được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Classification và Regression.
- KNN còn được gọi là một thuật toán Instance-based hay Memory-based learning.

Thuật toán KNN – K nearest neighbor

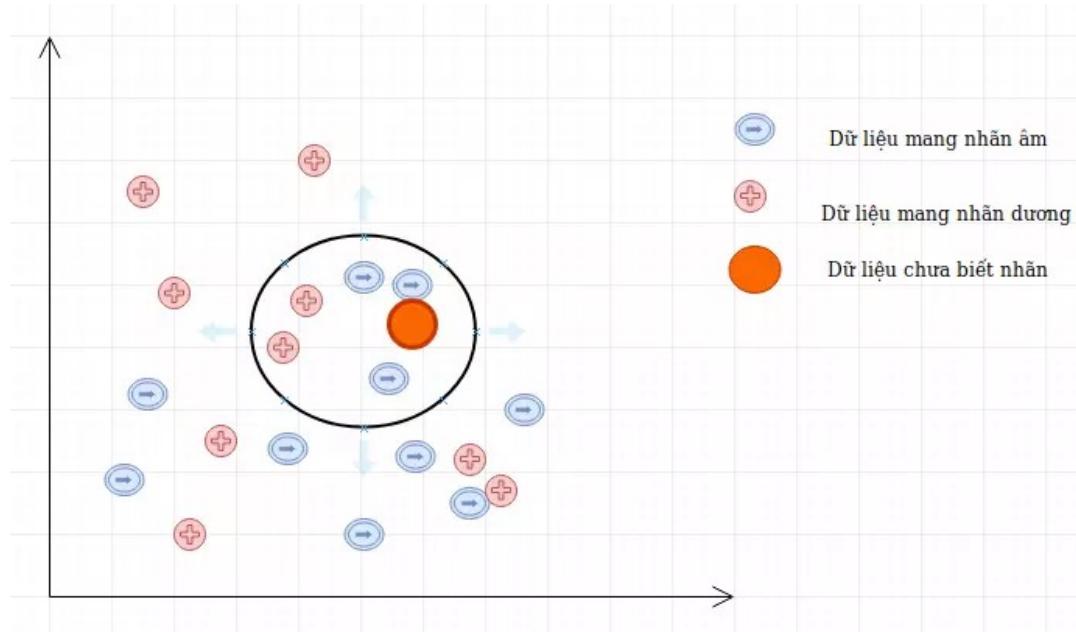
▪ Ứng dụng thực tế của KNN:

- Phân loại email rác: KNN có thể được sử dụng để phân loại email là rác hoặc không phải rác dựa trên các đặc trưng như từ khóa, độ dài và tần suất xuất hiện của từ trong email.
- Nhận dạng khuôn mặt: KNN có thể được áp dụng trong các hệ thống nhận dạng khuôn mặt để xác định xem một khuôn mặt mới có thuộc về người nào trong cơ sở dữ liệu đã biết hay không.
- Hệ thống gợi ý sản phẩm: KNN có thể được sử dụng trong hệ thống gợi ý sản phẩm để đề xuất các sản phẩm tương tự dựa trên sự tương đồng của các sản phẩm và lịch sử mua hàng của người dùng.

Thuật toán KNN – K nearest neighbor

■ Ý tưởng:

- **Thuật toán KNN** cho rằng những dữ liệu tương tự nhau sẽ tồn tại **gần nhau** trong một không gian, từ đó công việc của chúng ta là sẽ tìm k điểm gần với dữ liệu cần kiểm tra nhất.
- Việc tìm khoảng cách giữa 2 điểm cũng có nhiều công thức có thể sử dụng, tùy trường hợp mà chúng ta lựa chọn cho phù hợp.



Thuật toán KNN – K nearest neighbor

- Độ đo khoảng cách 2 điểm dữ liệu x và y với k thuộc tính

Distance functions

Euclidean

$$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

Manhattan

$$\sum_{i=1}^k |x_i - y_i|$$

Minkowski

$$\left(\sum_{i=1}^k (|x_i - y_i|)^q \right)^{1/q}$$

Thuật toán KNN – K nearest neighbor

- **Input :** D là tập các điểm dữ liệu đã được gắn nhãn và A là dữ liệu chưa được phân loại.
- **Output:** A đã được phân loại
- **Các bước thuật toán**
 - Bước 1: Đo khoảng cách (Euclidian, Manhattan, Minkowski, Minkowski hoặc Trọng số) từ dữ liệu mới A đến tất cả các dữ liệu khác đã được phân loại trong D.
 - Bước 2: Chọn K (K là tham số mà bạn định nghĩa) khoảng cách nhỏ nhất.
 - Bước 3: Kiểm tra danh sách các lớp có khoảng cách ngắn nhất và đếm số lượng của mỗi lớp xuất hiện.
 - Bước 4: Lấy đúng lớp (lớp xuất hiện nhiều lần nhất).
 - Bước 5: Lớp của dữ liệu mới là lớp mà đã nhận được ở bước 4. In kết quả

Thuật toán KNN – K nearest neighbor

- Ví dụ:

Tập dữ liệu D

	A	B	C
1	x	y	label
2	2	17	+
3	3	11	-
4	4	23	+
5	1	12	+
6	2	6	-
7	6	2	+
8	11	4	-
9	3	24	-
10	14	2	-
11	9	4	+
12	24	23	+
13	7	6	+
14	23	4	-
15	14	16	-
16	12	3	+

A ($x=3$, $y =9$) label ?

Thuật toán KNN – K nearest neighbor

- Bước 1 : Đo khoảng cách từ A đến các điểm tập dữ liệu D bằng độ đo Euclidian

	A	B	C	D	E
1	x	y	label		Distance
2	2	17	+		8.06
3	3	11	-		11.40
4	4	23	+		23.35
5	1	12	+		12.04
6	2	6	-		6.32
7	6	2	+		6.32
8	11	4	-		11.70
9	3	24	-		24.19
10	14	2	-		14.14
11	9	4	+		9.85
12	24	23	+		33.24
13	7	6	+		9.22
14	23	4	-		23.35
15	14	16	-		21.26
16	12	3	+		12.37
17					
18	3	9			

Thuật toán KNN – K nearest neighbor

- Bước 2: chọn $K=5$ (**)
- Bước 3:

	A	B	C	D	E	F
1	x	y	label		Distance	Rank
2	2	17	+		8.06	3
3	3	11	-		11.40	
4	4	23	+		23.35	
5	1	12	+		12.04	
6	2	6	-		6.32	1
7	6	2	+		6.32	2
8	11	4	-		11.70	
9	3	24	-		24.19	
10	14	2	-		14.14	
11	9	4	+		9.85	5
12	24	23	+		33.24	
13	7	6	+		9.22	4
14	23	4	-		23.35	
15	14	16	-		21.26	
16	12	3	+		12.37	
17						
18	3	9				

Thuật toán KNN – K nearest neighbor

- Bước 4: Lấy đúng lớp (lớp xuất hiện nhiều lần nhất).
 - Có 4 điểm thuộc lớp (+) và 1 điểm thuộc lớp (-)

	A	B	C	D	E	F
1	x	y	label			
2	2	17	+		8.06	3
3	3	11	-		11.40	
4	4	23	+		23.35	
5	1	12	+		12.04	
6	2	6	-		6.32	1
7	6	2	+		6.32	2
8	11	4	-		11.70	
9	3	24	-		24.19	
10	14	2	-		14.14	
11	9	4	+		9.85	5
12	24	23	+		33.24	
13	7	6	+		9.22	4
14	23	4	-		23.35	
15	14	16	-		21.26	
16	12	3	+		12.37	
17						
18	3	9				

- Bước 5: dự đoán A thuộc lớp (+)

Thuật toán KNN – K nearest neighbor

Height (in cms)	Weight (in kgs)	T Shirt Size
158	58	M
158	59	M
158	63	M
160	59	M
160	60	M
163	60	M
163	61	M
160	64	L
163	64	L
165	61	L
165	62	L
165	65	L
168	62	L
168	63	L
168	66	L
170	63	L
170	64	L
170	68	L

New customer named
'Monica' has height
161cm and weight 61kg ,
T Shirt Size ?

Thuật toán KNN – K nearest neighbor

- Cài đặt bằng python

Thuật toán cây quyết định (Decision Tree)

- Decision tree là một dạng công cụ giúp hỗ trợ cho việc đưa ra quyết định dựa trên mô hình cây (tree-like model) quyết định và hệ quả.
- Trong decision tree, chúng ta sẽ có các nút (node) của cây chứa các phép so sánh trên một thuộc tính (attribute) và kết quả của phép so sánh này sẽ dẫn đến các nhánh (branch) hệ quả khác nhau.
- Hệ quả cuối cùng của một nhánh (branch) được gọi là lá (leaf), nó sẽ biểu thị nhóm (class) tương ứng của cá thể (instance) trong bài toán phân loại (classification), hay một giá trị số trong bài toán hồi quy (regression).

Thuật toán cây quyết định (Decision Tree)

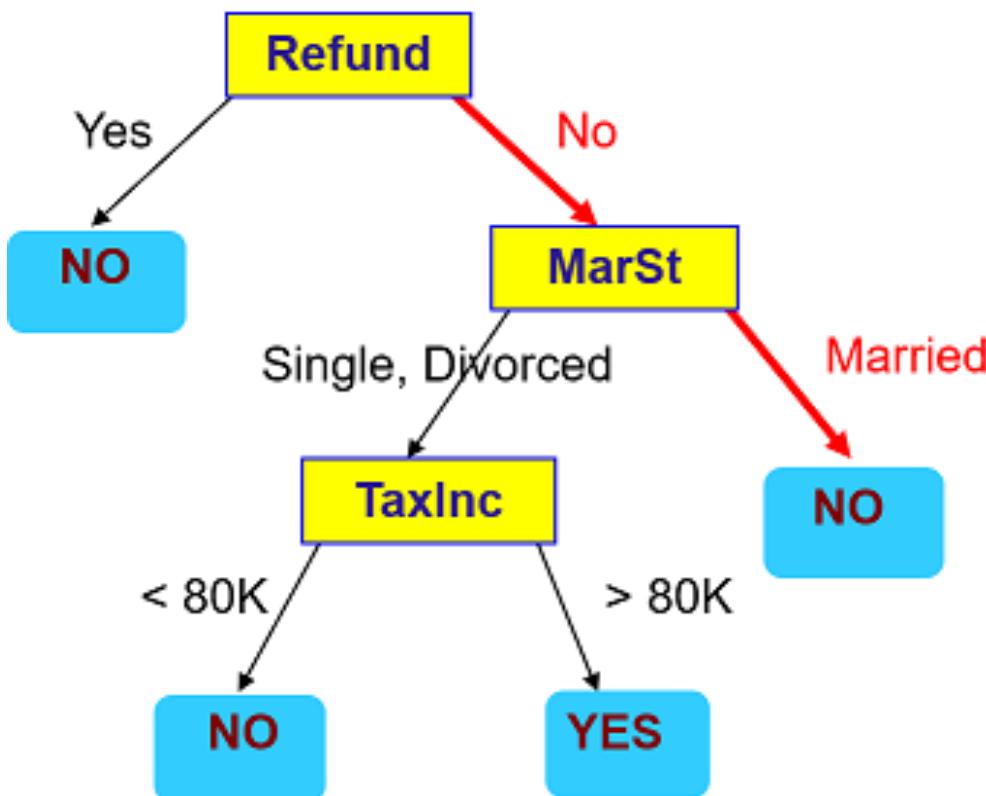
Xét ví dụ : tập huấn luyện (training set) của chúng ta như mô tả trong hình bên có 10 cá thể (instance) với

- 3 biến không phụ thuộc (independent variable) đó là hoàn tiền (refund), tình trạng hôn nhân (marital status), thu nhập chịu thuế (taxable income).
- Biến phụ thuộc (dependent variable) ở đây là có gian lận (cheat) thuế hay không

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Thuật toán cây quyết định (Decision Tree)

- Ứng từng giá trị của cá thể (instance) trên vào Decision tree đã xây dựng, chúng ta sẽ có được suy luận (deduction) rằng cá thể (instance) này không gian lận (cheat) thuế.



Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Thuật toán cây quyết định (Decision Tree)

- Có một số thuật toán để tạo một cây quyết định,
 - **CART** (Classification and Regression Trees) → dùng Gini Index(Classification) để kiểm tra.

$$Gini = 1 - \sum_{i=1}^C (p_i)^2$$

- **ID3** (Iterative Dichotomiser 3) → dùng Entropy function và Information gain để kiểm tra.

$$E(S) = \sum_{i=1}^c - p_i \log_2 p_i$$

Thuật toán cây quyết định (Decision Tree – ID3)

- ID3 là từ viết tắt của cụm từ **Iterative Dichotomiser 3**.
- ID3 là một thuật toán phân loại theo cách tiếp cận tham lam bằng cách chọn thuộc tính tốt nhất nhằm mang lại **Information Gain** (IG - lợi ích của thông tin) tối đa hoặc **Entropy** tối thiểu (entropy dùng để chỉ trạng thái ngẫu nhiên hoặc không có trật tự).
- Thuật toán **ID3** thực hiện các bước lần lượt như sau:
 - Tính **entropy** của tập dữ liệu
 - Đối với **mỗi thuộc tính/tính năng**: ID3 sẽ tính entropy cho tất cả giá trị phân loại và mức **IG** của tính năng đó
 - Tìm kiếm **thuộc tính/tính năng** để có được **IG tối đa**
 - **Lặp lại các bước thực hiện** cho đến khi đạt được **cây mong muốn**

Thuật toán cây quyết định (Decision Tree – ID3)

- Entropy là thuật ngữ thuộc Nhiệt động lực học, là thước đo của sự biến đổi, hỗn loạn hoặc ngẫu nhiên. Năm 1948, Shannon đã mở rộng khái niệm Entropy sang lĩnh vực nghiên cứu, thống kê với công thức như sau:
 - Với một phân phối xác suất của một biến rời rạc x có thể nhận n giá trị khác nhau x_1, x_2, \dots, x_n .
 - Giả sử rằng xác suất để x nhận các giá trị này là $p_i = p(x=x_i)$.
 - Ký hiệu phân phối này là $p=(p_1, p_2, \dots, p_n)$. Entropy của phân phối này được định nghĩa là:

$$H(p) = - \sum_{i=1}^n p_i \log_2(p_i)$$

Thuật toán cây quyết định (Decision Tree – ID3)

▪ Information Gain

$$G(x, S) = H(S) - H(x, S)$$

▪ Trong đó:

- $H(S)$ là Entropy tổng của toàn bộ tập data set S
- $H(f, S)$ là Entropy được tính trên thuộc tính f
- Do $H(S)$ là không đổi với mỗi tầng, ta chọn thuộc tính f có Entropy nhỏ nhất để thu được $\text{Gain}(S, f)$ lớn nhất.

Thuật toán cây quyết định (Decision Tree – ID3)

■ Ví dụ

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	no

Thuật toán cây quyết định (Decision Tree – ID3)

- Trong 14 giá trị đầu ra ở Bảng trên, có 5 giá trị bằng **no** và 9 giá trị bằng **yes**. Entroy tại **root node** của bài toán là:

$$H(S) = -\frac{5}{14} \log\left(\frac{5}{14}\right) - \frac{9}{14} \log\left(\frac{9}{14}\right) \approx 0.65$$

- Tiếp theo ta tính entropy cho các thuộc tính **outlook, temperature, humidity, wind**

Thuật toán cây quyết định (Decision Tree – ID3)

■ Thuộc tính **outlook**

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes

id	outlook	temperature	humidity	wind	play
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes

id	outlook	temperature	humidity	wind	play
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
10	rainy	mild	normal	weak	yes
14	rainy	mild	high	strong	no

$$H(\mathcal{S}_s) = -\frac{2}{5} \log\left(\frac{2}{5}\right) - \frac{3}{5} \log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(\mathcal{S}_o) = 0$$

$$H(\mathcal{S}_r) = -\frac{3}{5} \log\left(\frac{2}{5}\right) - \frac{3}{5} \log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(outlook, \mathcal{S}) = \frac{5}{14}H(\mathcal{S}_s) + \frac{4}{14}H(\mathcal{S}_o) + \frac{5}{14}H(\mathcal{S}_r) \approx 0.48$$

Thuật toán cây quyết định (Decision Tree – ID3)

■ Với **temperature**

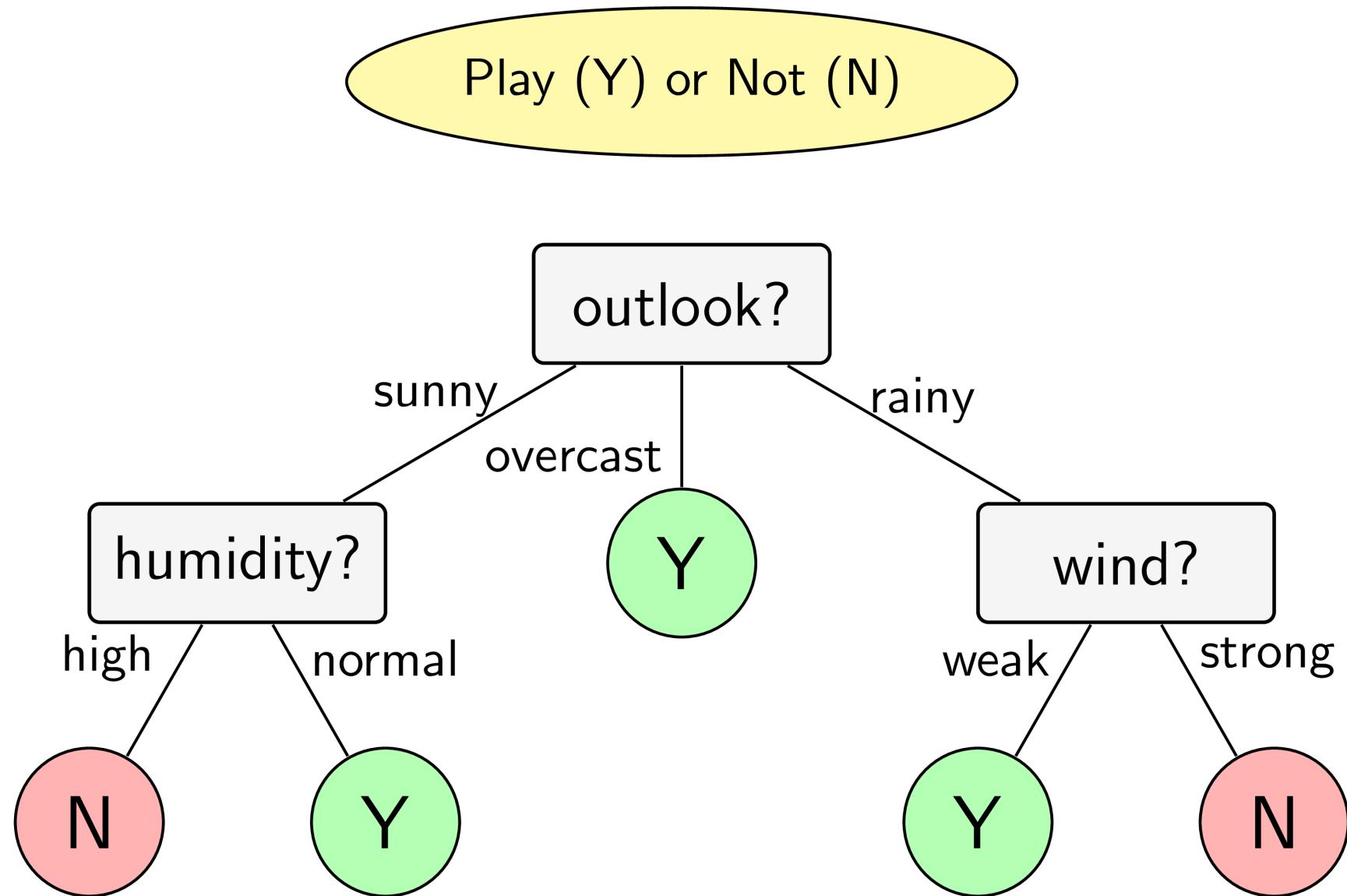
$$H(\mathcal{S}_h) = -\frac{2}{4} \log\left(\frac{2}{4}\right) - \frac{2}{4} \log\left(\frac{2}{4}\right) \approx 0.693$$
$$H(\mathcal{S}_m) = -\frac{4}{6} \log\left(\frac{4}{6}\right) - \frac{2}{6} \log\left(\frac{2}{6}\right) \approx 0.637$$
$$H(\mathcal{S}_c) = -\frac{3}{4} \log\left(\frac{3}{4}\right) - \frac{1}{4} \log\left(\frac{1}{4}\right) \approx 0.562$$
$$H(\text{temperature}, \mathcal{S}) = \frac{4}{14} H(\mathcal{S}_h) + \frac{6}{14} H(\mathcal{S}_m) + \frac{4}{14} H(\mathcal{S}_c) \approx 0.631$$

■ Với **humidity** và **wind**

$$H(\text{humidity}, \mathcal{S}) \approx 0.547, \quad H(\text{wind}, \mathcal{S}) \approx 0.618$$

■ Như vậy, thuộc tính cần chọn ở bước đầu tiên là **outlook** vì **H(outlook, S)** đạt giá trị nhỏ nhất (information gain là lớn nhất).

Thuật toán cây quyết định (Decision Tree – ID3)



Thuật toán cây quyết định (Decision Tree – ID3)

▪ Điều kiện dừng

- nếu node đó có entropy bằng 0, tức mọi điểm trong node đều thuộc một class.
- nếu node đó có số phần tử nhỏ hơn một ngưỡng nào đó. Trong trường hợp này, ta chấp nhận có một số điểm bị phân lớp sai để tránh overfitting. Class cho leaf node này có thể được xác định dựa trên class chiếm đa số trong node.
- nếu khoảng cách từ node đó đến root node đạt tới một giá trị nào đó. Việc hạn chế chiều sâu của tree này làm giảm độ phức tạp của tree và phần nào giúp tránh overfitting.
- nếu tổng số leaf node vượt quá một ngưỡng nào đó.
- nếu việc phân chia node đó không làm giảm entropy quá nhiều (information gain nhỏ hơn một ngưỡng nào đó).

Thuật toán cây quyết định (Decision Tree – ID3)

■ Bài tập:

STT	Vóc dáng	Quốc tịch	Gia cảnh	Nhóm
1	Nhỏ	Đức	Độc thân	A
2	Lớn	Pháp	Độc thân	A
3	Lớn	Đức	Độc thân	A
4	Nhỏ	Ý	Độc thân	B
5	Lớn	Đức	Có gia đình	B
6	Lớn	Ý	Độc thân	B
7	Lớn	Ý	Có gia đình	B
8	Nhỏ	Đức	Có gia đình	B
9	Nhỏ	Pháp	Có gia đình	?

Thuật toán cây quyết định (Decision Tree – ID3)

- Cài đặt bằng python

Thuật toán Random Forest

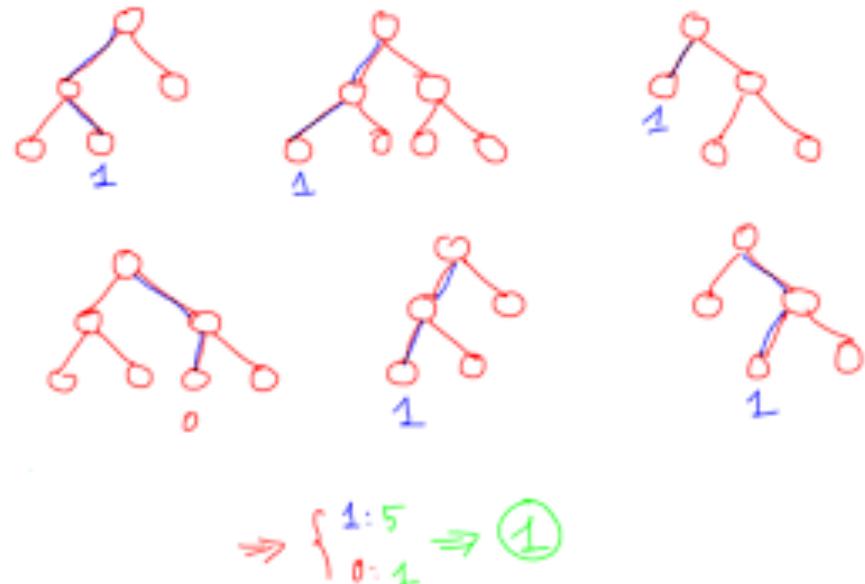
- Dù có độ chính xác khá cao nhưng cây quyết định tồn tại những hạn chế lớn đó là:
 - Dễ xảy ra **quá khớp** nếu số lượng các **đặc trưng để hỏi lớn**. Khi **độ sâu của cây quyết định không bị giới hạn** thì có thể tạo ra những node lá chỉ có một vài quan sát. Những kết luận **dự báo** từ chúng thường **chỉ đúng trên tập huấn luyện mà không đúng trên tập kiểm tra**.
 - Trong tình huống bộ dữ liệu **có số lượng biến lớn**. Một cây quyết định có độ sâu giới hạn (**để giảm thiểu quá khớp**) thường bỏ sót những biến quan trọng.
 - **Cây quyết định** chỉ tạo ra một kịch bản **dự báo duy nhất cho mỗi một quan sát** nên nếu **model có hiệu suất kém** thì **kết quả sẽ bị chêch**.

Thuật toán Random Forest

- Nếu như sức mạnh của một cây quyết định là yếu thì hợp sức của nhiều cây quyết định sẽ trở nên mạnh mẽ hơn .**Ý tưởng của sự hợp sức (ensemble learning) đã hình thành nên mô hình rừng cây (Random Forest).**
- **Mô hình khá nổi tiếng**, ở các cuộc thi trên kaggle mô hình *Random forest* thường được sử dụng và nhiều lần dành chiến thắng. Vì có độ chính xác cao, giảm thiểu hiện tượng **quá khớp** nên mô hình *rừng cây* được sử dụng rộng rãi trong cả hai lớp bài toán phân loại và dự báo của học có giám sát.

Thuật toán Random Forest

- Random Forest được huấn luyện dựa trên sự phối hợp giữa luật kết hợp (*ensembling*) và quá trình *lấy mẫu tái lập (bootstrapping)*.
- Thuật toán này tạo ra **nhiều cây quyết định** mà mỗi cây quyết định được huấn luyện dựa trên **nhiều mẫu con** khác nhau và kết quả dự báo là *bầu cử (voting)* từ toàn bộ những cây quyết định.



Thuật toán Random Forest



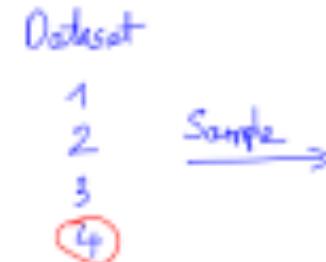
Output
3



Output
3
1

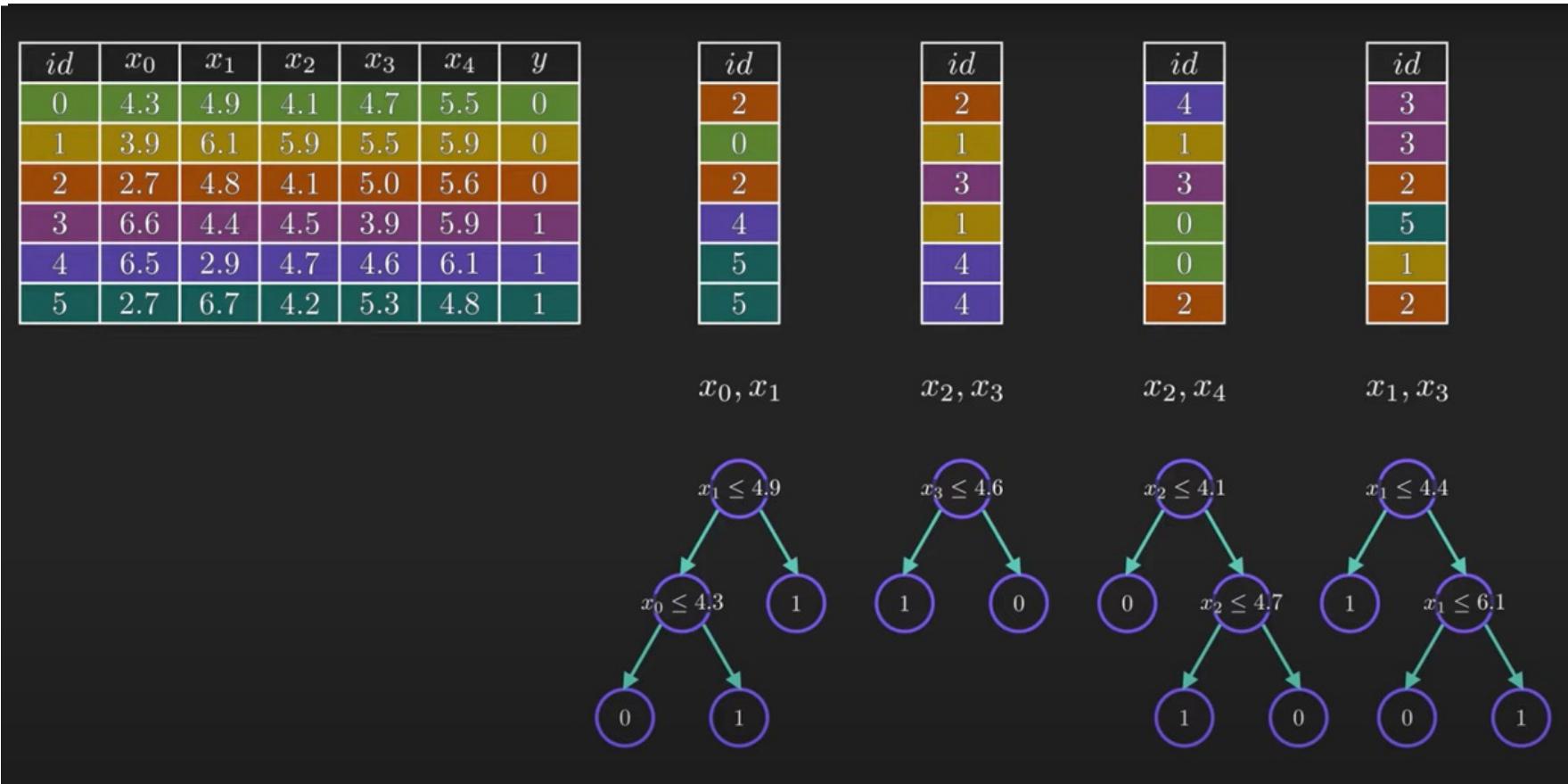


Output
3
1
3

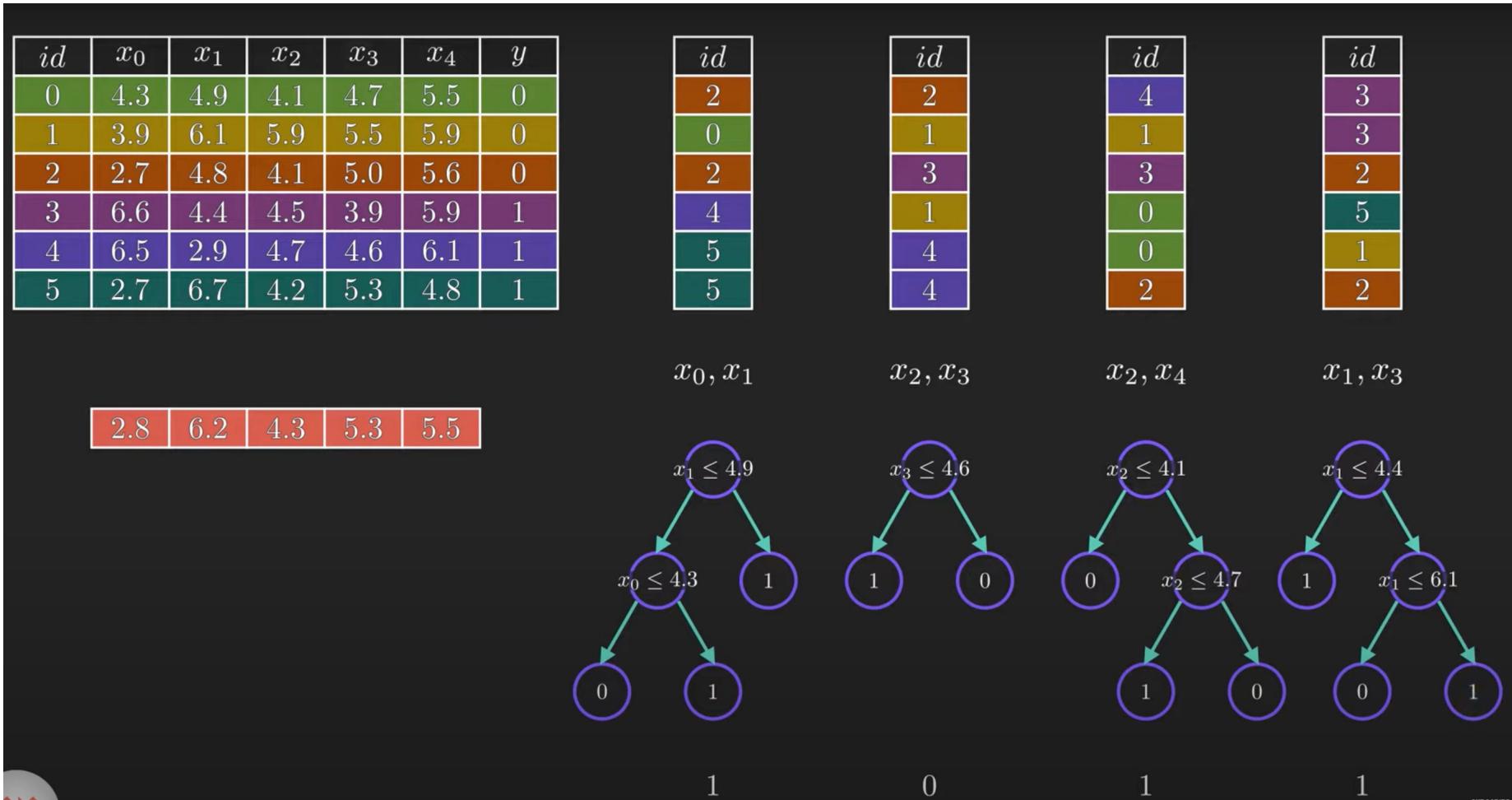


- Thuật toán gồm các bước sau:
 - Chọn các **mẫu ngẫu nhiên** từ **tập dữ liệu** đã cho.
 - Thiết lập **cây quyết định** cho từng mẫu và nhận kết quả dự đoán từ mỗi quyết định cây.
 - Hãy bỏ phiếu cho mỗi kết quả dự đoán.
 - Chọn kết quả được dự đoán nhiều nhất là **dự đoán cuối cùng**.

Thuật toán Random Forest



Thuật toán Random Forest



Thuật toán Naïve Bayes

- Đây là một kỹ thuật phân loại dựa trên Định lý Bayes với giả định về sự độc lập giữa các yếu tố dự đoán.
- Bộ phân loại Naive Bayes giả định rằng sự hiện diện của một đối tượng cụ thể trong một lớp không liên quan đến sự hiện diện của bất kỳ đối tượng địa lý nào khác.
- Ví dụ: một quả có thể được coi là táo nếu nó có màu đỏ, tròn và đường kính khoảng 3 inch. Ngay cả khi các đặc điểm này phụ thuộc vào nhau hoặc dựa trên sự tồn tại của các đặc điểm khác, tất cả các đặc tính này đều góp phần độc lập vào xác suất quả này là táo và đó là lý do tại sao nó được gọi là ‘Naive’.

Thuật toán Naïve Bayes

■ Đặc điểm của Naive Bayesian

- Thuật toán Naïve Bayes là một thuật toán học có giám sát, dựa trên **định lý Bayes** và được sử dụng để giải các bài toán phân loại.
- Nó chủ yếu được sử dụng trong *phân loại văn bản* bao gồm một tập dữ liệu đào tạo chiều cao.
- Naïve Bayes Classifier là một trong những thuật toán Phân loại đơn giản và hiệu quả nhất giúp xây dựng các mô hình học máy nhanh có thể đưa ra dự đoán nhanh chóng.
- **Nó là một bộ phân loại theo xác suất, có nghĩa là nó dự đoán trên cơ sở xác suất của một đối tượng .**
- Một số ví dụ phổ biến của Thuật toán Naïve Bayes là **lọc thư rác, phân tích tình cảm và phân loại các bài báo** .

Thuật toán Naïve Bayes

▪ Các loại mô hình Naïve Bayes:

- **Gaussian** : Mô hình Gaussian giả định rằng các đối tượng địa lý tuân theo phân phối chuẩn. Điều này có nghĩa là nếu các yếu tố dự đoán nhận các giá trị liên tục thay vì rời rạc, thì mô hình giả định rằng các giá trị này được lấy mẫu từ phân phối Gaussian.
- **Đa thức** : Bộ phân loại Naïve Bayes đa thức được sử dụng khi dữ liệu được phân phối đa thức. Nó chủ yếu được sử dụng cho các vấn đề phân loại tài liệu, nó có nghĩa là một tài liệu cụ thể thuộc về danh mục nào như Thể thao, Chính trị, giáo dục, v.v. Trình phân loại sử dụng tần suất từ cho các yếu tố dự đoán.
- **Bernoulli** : Bộ phân loại Bernoulli hoạt động tương tự như bộ phân loại Đa thức, nhưng các biến dự báo là các biến Booleans độc lập. Chẳng hạn như nếu một từ cụ thể có trong tài liệu hay không. Mô hình này cũng nổi tiếng với các nhiệm vụ phân loại tài liệu.

Thuật toán Naïve Bayes

- là một thuật toán phân lớp được mô hình hóa dựa trên định lý Bayes trong xác suất thống kê:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

- Trong đó:
 - $P(y|X)$ gọi là posterior probability: xác suất của mục tiêu y với điều kiện có đặc trưng X
 - $P(X|y)$ gọi là posterior probability: xác suất của đặc trưng X với điều kiện có mục tiêu y
 - $P(y)$ gọi là prior probability của mục tiêu y
 - $P(X)$ gọi là prior probability của đặc trưng X

Thuật toán Naïve Bayes

- Với X là vector các đặc trưng, có thể viết dưới dạng

$$X = (x_1, x_2, x_3, \dots, x_n)$$

- Khi đó, đẳng thức Bayes trở thành

$$P(y|x_1, x_2, x_3, \dots, x_n) = \frac{P(x_1|y)P(x_2|y) \dots P(x_n|y)}{P(X)}$$

- Trong mô hình Navie Bayes, có 2 giả thuyết được đặt ra :

- Các đặc trưng dựa vào mô hình là độc lập với nhau. Tức là sự thay đổi giá trị của một đặc trưng không ảnh hưởng đến các đặc trưng còn lại
- Các đặc trưng dựa vào mô hình có ảnh hưởng ngang nhau đối với đầu ra mục tiêu. Khi đó kết quả mục tiêu y để $P(y|X)$ đạt cực đại trở thành:

$$y = \operatorname{argmax}_y P(y) \prod_{i=1}^n P(x_i|y)$$

Thuật toán Naïve Bayes

- Thuật toán:

- Bước 1: Tính xác suất biến cố X với từng lớp
- Bước 2: Tính xác suất xảy ra biến cố X
- Bước 3: sử dụng phương trình Naive Bayesian để tính xác suất sau cho mỗi lớp. Lớp có xác suất cao nhất là kết quả của dự đoán.

Thuật toán Naïve Bayes - Bernoulli

- **Bộ phân loại Bernoulli (Bernoulli Naive Bayes)** là một phiên bản của thuật toán Naive Bayes, sử dụng phân phối Bernoulli trong đó mỗi **đặc trưng** là một **biến nhị phân**, đại diện cho việc **một yếu tố** nào đó có xuất hiện trong mẫu hay không.
- **Phân phối Bernoulli:** Phân phối Bernoulli mô tả các biến ngẫu nhiên có hai trạng thái (0 hoặc 1), tương ứng với hai khả năng "có" hoặc "không".

$$p(x) = p(X = x) = \begin{cases} p & \text{khi } x = 1 \\ 1 - p & \text{khi } x = 0 \end{cases}$$

- Trong đó
 - x là biến ngẫu nhiên
 - $p(x)$ là xác suất của x

Thuật toán Naïve Bayes - Bernoulli

Ví dụ

Studied	Result
No	Fail
No	Pass
Yes	Fail
Yes	Pass
Yes	Pass
No	Fail

if studies = yes
the result =Fail or Pass

$$p(A|B) = \frac{p(B|A) * p(A)}{p(B)}$$

Let C and X are random variables

$$p(c|x) = \frac{p(x|c) * p(c)}{p(x)}$$

$$p(C = c|X = x) = \frac{p(X = x|C = c) * p(C = c)}{p(X = x)}$$

Thuật toán Naïve Bayes - Bernoulli

$$p(C = c|X = x) = \frac{p(X = x|C = c) * p(C = c)}{p(X = x)}$$

$$p(C = c1|X = x) = ? \quad p(C = c2|X = x) = ?$$

$$p(res = pass | stud = yes) = \frac{p(stud = yes | res = pass) * p(res = pass)}{p(stud = yes)}$$

$$p(res = fail | stud = yes) = \frac{p(stud = yes | res = fail) * p(res = fail)}{p(stud = yes)}$$

Thuật toán Naïve Bayes - Bernoulli

Studied	Result
No	Fail
No	Pass
Yes	Fail
Yes	Pass
Yes	Pass
No	Fail

$$p(res = pass) = \frac{3}{6} = \frac{1}{2}$$

$$p(res = fail) = \frac{3}{6} = \frac{1}{2}$$

$$p(res = pass | stud = yes) = \frac{p(stud = yes | res = pass) * p(res = pass)}{p(stud = yes)}$$

$$p(res = fail | stud = yes) = \frac{p(stud = yes | res = fail) * p(res = fail)}{p(stud = yes)}$$

Thuật toán Naïve Bayes - Bernoulli

Studied	Result
No	Fail
No	Pass
Yes	Fail
Yes	Pass
Yes	Pass
No	Fail

$$p(res = pass) = \frac{3}{6} = 0.5$$

$$p(res = fail) = \frac{3}{6} = 0.5$$

$$p(stud = yes | res = pass) = \frac{2}{3}$$

$$p(stud = yes | res = fail) = \frac{1}{3}$$

$$p(res = pass | stud = yes) = \frac{p(stud = yes | res = pass) * p(res = pass)}{p(stud = yes)}$$

$$p(res = fail | stud = yes) = \frac{p(stud = yes | res = fail) * p(res = fail)}{p(stud = yes)}$$

Thuật toán Naïve Bayes - Bernoulli

Studied	Result
No	Fail
No	Pass
Yes	Fail
Yes	Pass
Yes	Pass
No	Fail

$$p(res = pass) = \frac{3}{6} = \frac{1}{2}$$

$$p(res = fail) = \frac{3}{6} = \frac{1}{2}$$

$$p(stud = yes | res = pass) = \frac{2}{3}$$

$$p(stud = yes | res = fail) = \frac{1}{3}$$

$$p(stud = yes)$$

$$= p(stud = yes | res = pass) * p(res = pass) + p(stud = yes | res = fail) * p(res = fail)$$

$$= \frac{2}{3} * \frac{1}{2} + \frac{1}{3} * \frac{1}{2} = \frac{1}{2}$$

$$p(res = pass | stud = yes) = \frac{p(stud = yes | res = pass) * p(res = pass)}{p(stud = yes)}$$

$$p(res = fail | stud = yes) = \frac{p(stud = yes | res = fail) * p(res = fail)}{p(stud = yes)}$$

Thuật toán Naïve Bayes - Bernoulli

Studied	Result
No	Fail
No	Pass
Yes	Fail
Yes	Pass
Yes	Pass
No	Fail
Yes	???

$$p(res = pass) = \frac{3}{6} = \frac{1}{2}$$

$$p(res = fail) = \frac{3}{6} = \frac{1}{2}$$

$$p(stud = yes) = \frac{1}{2}$$

$$p(stud = yes | res = pass) = \frac{2}{3}$$

$$p(stud = yes | res = fail) = \frac{1}{3}$$

$$p(res = pass | stud = yes) = \frac{p(stud = yes | res = pass) * p(res = pass)}{p(stud = yes)}$$

$$= \frac{2}{3} * \frac{1}{2} * \frac{2}{1} = \frac{2}{3}$$

$$p(res = fail | stud = yes) = \frac{p(stud = yes | res = fail) * p(res = fail)}{p(stud = yes)}$$

$$= \frac{1}{3} * \frac{1}{2} * \frac{2}{1} = \frac{1}{3}$$

Thuật toán Naïve Bayes - Bernoulli

- Xét ví dụ :

Color	Type	Origin	Stolen
Red	Sports	Domestic	Yes
Red	Sports	Domestic	No
Red	Sports	Domestic	Yes
Yellow	Sports	Domestic	No
Yellow	Sports	Imported	Yes
Yellow	SUV	Imported	No
Yellow	SUV	Imported	Yes
Yellow	SUV	Domestic	No
Red	SUV	Imported	No
Red	Sports	Imported	Yes

Color = Red
Type = SUV
Origin = Domestic
Stolen ?

Thuật toán Naïve Bayes - Bernoulli

- Bước 1:

- $P(\text{Yes}) = \frac{5}{10}$, $P(\text{No}) = \frac{5}{10}$

- Điều kiện: Yes

- $\checkmark P(\text{Red}|\text{Yes}) = \frac{3}{5}$, $P(\text{SUV } |\text{Yes}) = \frac{1}{4}$, $P(\text{Domestic}|\text{Yes}) = \frac{2}{5}$

- $\checkmark P(X|\text{Yes}) = P(\text{Red}|\text{Yes}) * P(\text{SUV } |\text{Yes}) * P(\text{Domestic}|\text{Yes}) = \frac{3}{5} * \frac{1}{4} * \frac{2}{5}$
 $= \frac{6}{100}$

- Điều kiện: No

- $\checkmark P(\text{Red}|\text{No}) = \frac{2}{5}$, $P(\text{SUV } |\text{No}) = \frac{3}{4}$, $P(\text{Domestic}|\text{No}) = \frac{3}{5}$

- $\checkmark P(X|\text{No}) = P(\text{Red}|\text{No}) * P(\text{SUV } |\text{No}) * P(\text{Domestic}|\text{No}) = \frac{18}{100}$

Thuật toán Naïve Bayes - Bernoulli

■ Bước 2:

- $P(X) = P(X|Yes) * P(Yes) + P(X|No) * P(No)$

$$= \frac{6}{100} * \frac{5}{10} + \frac{18}{100} * \frac{5}{10} = \frac{6}{100} * \frac{5}{10}$$

■ Bước 3:

- $P(Yes|X) = (P(X|Yes)*P(Yes)) / P(X) = (6/100 * 5/10) / (120/1000) = 3/12 = 1/4 = 0.25$
- $P(No|X) = (P(X|No)*P(No)) / P(X) = (18/100 * 5/10) / (120/1000) = 9/12 = 3/4 = 0.75$

■ => The car is not stolen

Thuật toán Naïve Bayes - Gaussian

- **Bộ phân loại Gaussian (Gaussian Naive Bayes)** là một biến thể của thuật toán Naive Bayes, đặc biệt phù hợp cho các bài toán phân loại khi các **đặc trưng liên tục và có phân phối chuẩn (Gaussian distribution)**.
- **Phân phối Gaussian** còn được gọi là phân phối chuẩn (**normal distribution**), có hình dạng đồ thị giống như hình chuông.
- Một phân phối chuẩn có dạng một đường cong hình chuông, được xác định bằng hai tham số:
 - **Giá trị trung bình (mean)**: Thể hiện giá trị trung tâm của phân phối.
 - **Phương sai (variance)**: Thể hiện độ phân tán của các giá trị quanh trung bình.

Thuật toán Naïve Bayes - Gaussian

- **Công thức tính xác suất của phân phối chuẩn:**
Công thức mật độ xác suất của phân phối Gaussian đối với một đặc trưng x được mô tả bởi:

$$P(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Trong đó:
 - μ : Giá trị trung bình.
 - σ^2 : Phương sai (variance).
 - x : Giá trị của đặc trưng.

Thuật toán Naïve Bayes - Gaussian

■ Ví dụ

Length	Category
1.4	0
1	0
1.3	0
1.9	0
2	0
3.8	1
4.1	1
3.9	1
4.2	1
3.4	1

One feature: Length (continuous)

Two classes: '0' and '1'

class=? when Length=3.0

Thuật toán Naïve Bayes - Gaussian

■ Áp dụng phân phối Gaussian

❖ Example: one feature

Length	Category
1.4	0
1	0
1.3	0
1.9	0
2	0
3.8	1
4.1	1
3.9	1
4.2	1
3.4	1

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$-\infty < x < \infty$$

μ : mean

σ^2 : variance

$$p(A|B) = \frac{p(B|A) * p(A)}{p(B)}$$

Let C and X are random variables

$$p(c|x) = \frac{p(x|c) * p(c)}{p(x)}$$

One feature: Length (continuous)

Two classes: '0' and '1'

class=? when Length=3.0

$$p(C = c|X = x) = \frac{p(X = x|C = c) * p(C = c)}{p(X = x)}$$

Thuật toán Naïve Bayes - Gaussian

❖ Example

Length	Category
1.4	0
1	0
1.3	0
1.9	0
2	0
3.8	1
4.1	1
3.9	1
4.2	1
3.4	1

$$\mu_0 = 1.52$$

$$\sigma_0^2 = 0.1416$$

$$\mu_1 = 3.88$$

$$\sigma_1^2 = 0.0776$$

$$f(x|0) = \frac{1}{\sigma_0 \sqrt{2\pi}} e^{-\frac{(x-\mu_0)^2}{2\sigma_0^2}}$$

$$f(x|1) = \frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}$$

Length	pdf(Length)	Category
1.4	1.007	0
1	0.408	0
1.3	0.893	0
1.9	0.636	0
2	0.469	0
3.8	1.374	1
4.1	1.048	1
3.9	1.428	1
4.2	0.74	1
3.4	0.324	1

$$p(Cat = 0 | Len = 3.0) = \frac{p(Len = 3.0 | Cat = 0) * p(Cat = 0)}{p(Len = 3.0)}$$

$$p(Cat = 1 | Len = 3.0) = \frac{p(Len = 3.0 | Cat = 1) * p(Cat = 1)}{p(Len = 3.0)}$$

Thuật toán Naïve Bayes - Gaussian

- Bước 1: Tính xác suất biến cố X theo từng lớp với phân phối Gaussian

❖ Example

Length	Category
1.4	0
1	0
1.3	0
1.9	0
2	0
3.8	1
4.1	1
3.9	1
4.2	1
3.4	1

$$\mu_0 = 1.52$$

$$\sigma_0^2 = 0.1416$$

$$\mu_1 = 3.88$$

$$\sigma_1^2 = 0.0776$$

$$f(x|0) = \frac{1}{\sigma_0 \sqrt{2\pi}} e^{-\frac{(x-\mu_0)^2}{2\sigma_0^2}}$$

$$f(x|1) = \frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}$$

$$p(Cat = 0) = 0.5$$

$$p(Len = 3.0 | Cat = 0) = \frac{1}{\sigma_0 \sqrt{2\pi}} e^{-\frac{(3.0-\mu_0)^2}{2\sigma_0^2}} = 0.0004638$$

$$p(Cat = 1) = 0.5$$

$$p(Len = 3.0 | Cat = 1) = \frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{(3.0-\mu_1)^2}{2\sigma_1^2}} = 0.0097495$$

Thuật toán Naïve Bayes - Gaussian

■ Bước 2: Tính xác suất biến cố X

Length	Category
1.4	0
1	0
1.3	0
1.9	0
2	0
3.8	1
4.1	1
3.9	1
4.2	1
3.4	1

$$p(Cat = 0) = 0.5$$

$$p(Cat = 1) = 0.5$$

$$p(Len = 3.0|Cat = 0) = 0.0004638$$

$$p(Len = 3.0|Cat = 1) = 0.0097495$$

$$p(Len = 3.0) = p(Len = 3.0|Cat = 0) * p(Cat = 0) + p(Len = 3.0|Cat = 1) * p(Cat = 1)$$

$$= 0.0004638 * 0.5 + 0.0097495 * 0.5 = 0.005106$$

Thuật toán Naïve Bayes - Gaussian

■ Bước 3:

❖ Example

Length	Category
1.4	0
1	0
1.3	0
1.9	0
2	0
3.8	1
4.1	1
3.9	1
4.2	1
3.4	1

$$p(Cat = 0) = 0.5$$

$$p(Cat = 1) = 0.5$$

$$p(Len = 3.0|Cat = 0) = 0.0004638$$

$$p(Len = 3.0|Cat = 1) = 0.0097495$$

$$p(Len = 3.0) = 0.005106$$

$$p(Cat = 0|Len = 3.0) = \frac{p(Len = 3.0|Cat = 0) * p(Cat = 0)}{p(Len = 3.0)} = \frac{0.0004638 * 0.5}{0.005106} = 0.0454$$

$$p(Cat = 1|Len = 3.0) = \frac{p(Len = 3.0|Cat = 1) * p(Cat = 1)}{p(Len = 3.0)} = \frac{0.0097495 * 0.5}{0.005106} = 0.9545$$

Thuật toán Naïve Bayes - Multinomial

- **Bộ phân loại Đa thức (Multinomial Naive Bayes)** là một biến thể của thuật toán Naive Bayes, được thiết kế đặc biệt cho dữ liệu rời rạc, thường là dữ liệu biểu thị số lần xuất hiện của một đặc trưng.
- Đây là một trong những mô hình phổ biến nhất trong các bài toán phân loại văn bản, nơi mỗi đặc trưng thường là một từ và giá trị của nó là tần suất xuất hiện từ đó trong văn bản.
- **Phân phối Đa thức:** Bộ phân loại này hoạt động tốt trên dữ liệu có bản chất rời rạc (discrete data), chẳng hạn như số lượng từ trong văn bản. Phân phối đa thức mô tả khả năng của các lần đếm khác nhau trong tập dữ liệu, ví dụ như số lần một từ xuất hiện trong một tài liệu.

Thuật toán Naïve Bayes - Multinomial

▪ Phương pháp Laplace smoothing:

- một kỹ thuật làm mịn dữ liệu dạng phân loại (categorical data), một giá trị nhỏ gọi là **pseudo-count** sẽ được thêm vào để làm thay đổi xác suất đầu ra.
- Kết quả là sẽ không còn xác suất nào bằng không nữa, đây là một cách điều chỉnh Naive Bayes, khi **pseudo-count (alpha)** bằng không có nghĩa là không làm mịn gì cả.

$$P(x_i|C) = \frac{x_i + \alpha}{N + \alpha d}$$

▪ Trong đó:

- x_i : là đặc trưng (biến ngẫu nhiên)
- C: là biến phân loại
- N: tổng số x_i thuộc C
- α : Tham số làm mịn Laplace (để tránh xác suất bằng 0).
- d: số lượng C

Thuật toán Naïve Bayes - Multinomial

- Ví dụ:

ID	Mây	Áp suất	Gió	Kết quả
1	Ít	Cao	Bắc	Không mưa
2	nhiều	Cao	Bắc	Mưa
3	ít	Thấp	Bắc	Không mưa
4	nhiều	Thấp	Bắc	Mưa
5	nhiều	Trung Bình	Bắc	Mưa
6	ít	Cao	Nam	Không mưa
7	nhiều	Cao	Nam	Mưa
8	nhiều	Thấp	Nam	Không mưa

X = Mây :ít, Áp suất: Thấp, Gió: Nam

Kết quả: mưa hay không mưa

Thuật toán Naïve Bayes - Multinomial

- Bước 1:

- $P(\text{Mưa}) = 4/8$, $P(\text{Không mưa})=4/8$
- **$P(\text{ít|Mưa}) = 0 \Rightarrow \text{Laplace smoothing}$**
 - ✓ $\alpha = 1$, $d = 2$ (Mưa , Không mưa) , $N = 4$
- **C = Mưa**
 - ✓ $P(\text{ít|Mưa}) =(0+1)/(4+2) = 1/6$
 - ✓ $P(\text{thấp |Mưa})= (1+1) /(4+3) = 2/7$
 - ✓ $P(\text{Nam|Mưa})= (1+1)/(4+2) = 2/6$
 - ✓ **$P(X|Mưa) = P(\text{ít|Mưa})*P(\text{thấp |Mưa})*P(\text{Nam|Mưa})$**
= 1/63

Thuật toán Naïve Bayes - Multinomial

- C = Không mưa

- ✓ $P(\text{ít} | \text{Không mưa}) = (3+1)/(4+2) = 4/6$

- ✓ $P(\text{thấp} | \text{Không Mưa}) = (2+1)/(4+3) = 3/7$

- ✓ $P(\text{Nam} | \text{Không Mưa}) = (2+1)/(4+2) = 3/6$

- ✓ $P(X|\text{Không mưa}) = P(\text{ít}|\text{Không mưa}) * P(\text{thấp}|\text{Không Mưa}) * P(\text{Nam}|\text{Không Mưa}) = 1/7$

- Bước 2:

- $P(X) = P(X|\text{Mưa}) * P(\text{Mưa}) + P(X|\text{Không mưa}) * P(\text{Không mưa}) = (1/63) * (4+1)/(8+2) + 1/7 * (4+1)/(8+2) = 13/28$

- Bước 3:

- $P(\text{Mưa}|X) = (P(X|\text{Mưa}) * P(\text{Mưa})) / P(X) = 0.85$

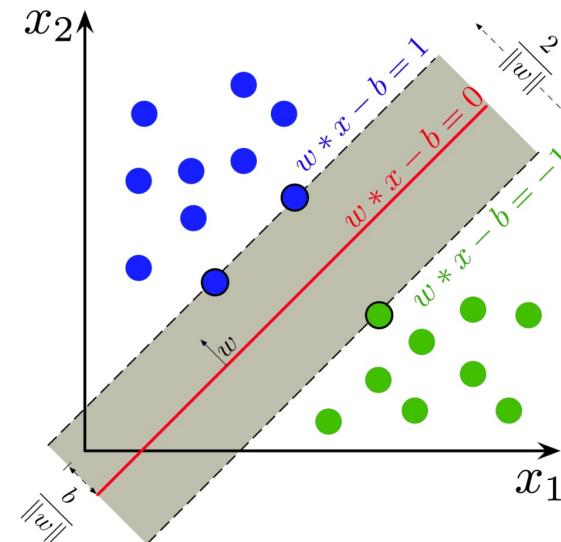
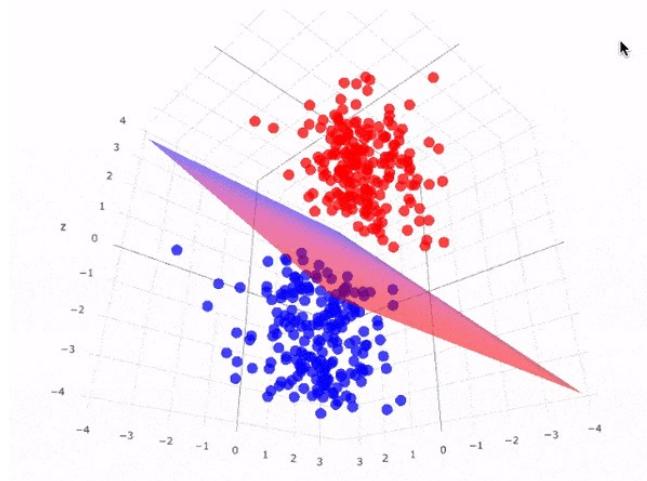
- $P(\text{Không mưa}|X) = (P(X|\text{Không mưa}) * P(\text{Không mưa})) / P(X) = 0.15$

Thuật toán Naïve Bayes - Gaussian

- Cài đặt python

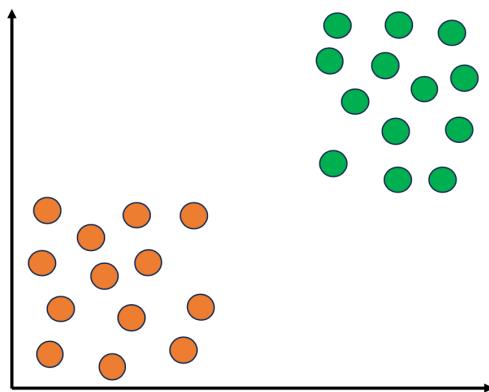
Support Vector Machine

- SVM là thuật toán quan trọng trong các thuật toán Học máy. Nó được sử dụng trong cả phân lớp lưỡng quy .
- Mục tiêu của SVM là tìm ra một siêu phẳng trong không gian N chiều (ứng với N đặc trưng) chia dữ liệu thành hai phần tương ứng với lớp của chúng

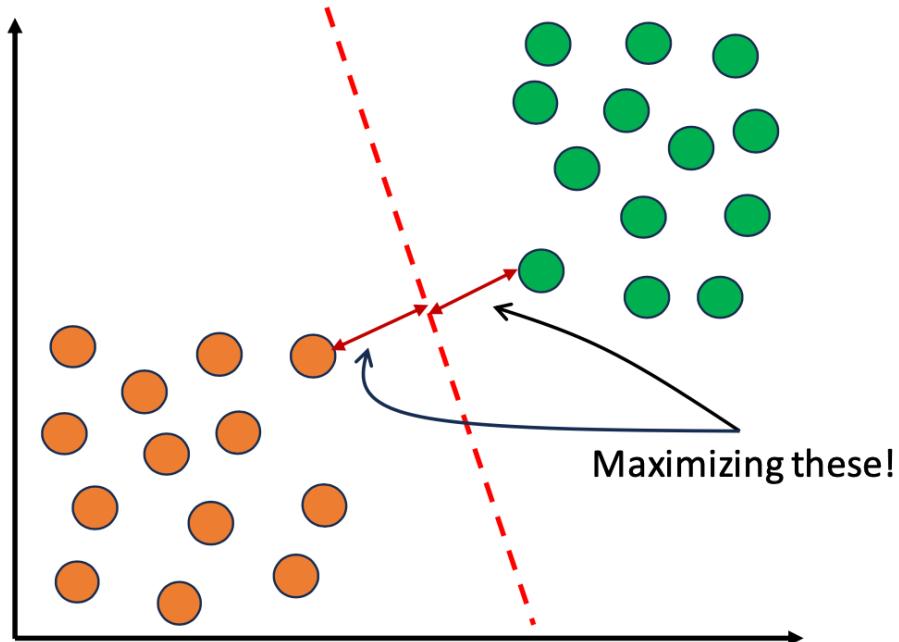


Support Vector Machine

- Giả sử chúng ta phải phân loại tập dữ liệu các lớp dương (màu xanh) nhãn là 1 và các dữ liệu lớp âm (màu cam) nhãn là -1 (tập dữ liệu có thể phân tách tuyến tính).



Support Vector Machine

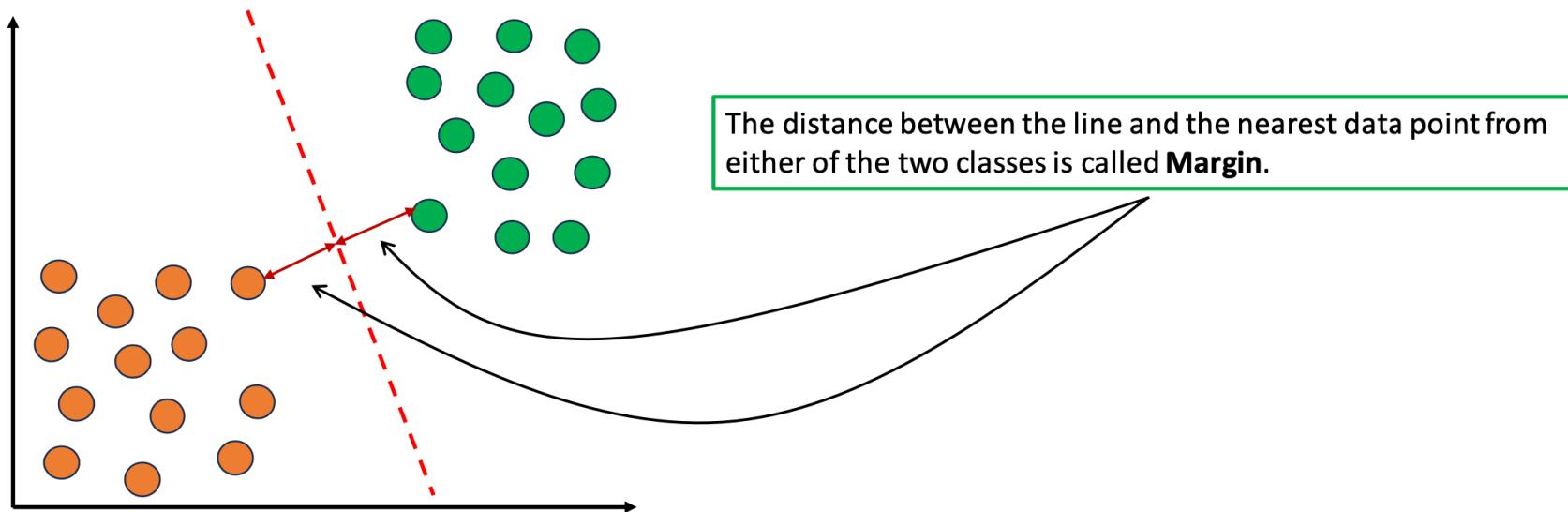


SVM idea: Find the line that best separates the data into classes while maximizing the distances between nearest points.

Support Vector Machine

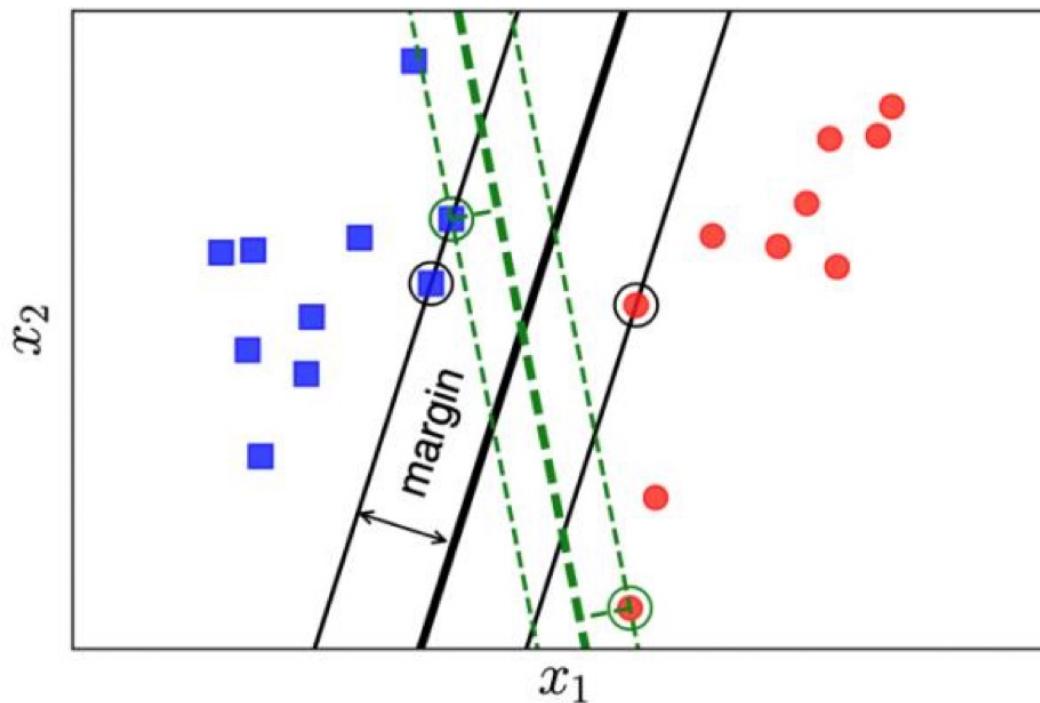
▪ Độ rộng của margin:

- Nếu ta định nghĩa độ thỏa mãn của một lớp tỉ lệ thuận với khoảng cách gần nhất từ một điểm của lớp đó tới đường/mặt phân chia, thì lớp tròn đỏ sẽ không thỏa mãn vì đường phân chia gần nó hơn lớp vuông xanh rất nhiều.
- Chúng ta cần một đường phân chia sao cho khoảng cách từ điểm gần nhất của mỗi lớp (các điểm được khoanh tròn) tới đường phân chia là như nhau. Khoảng cách như nhau này được gọi là margin.



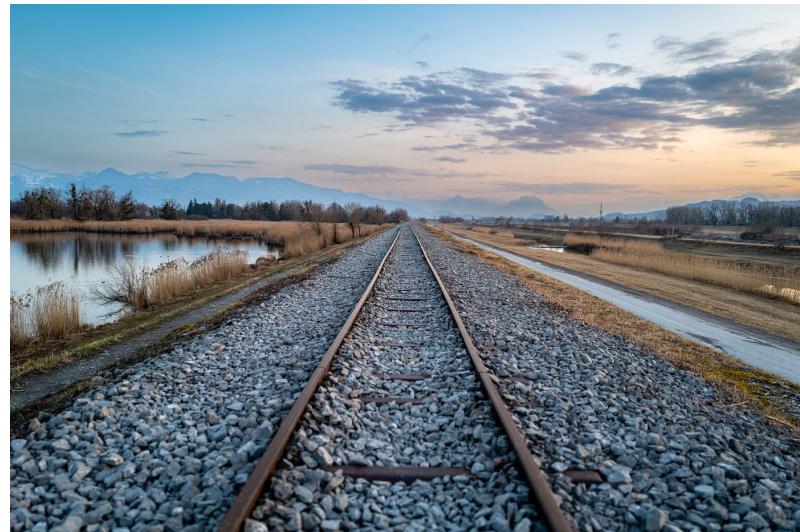
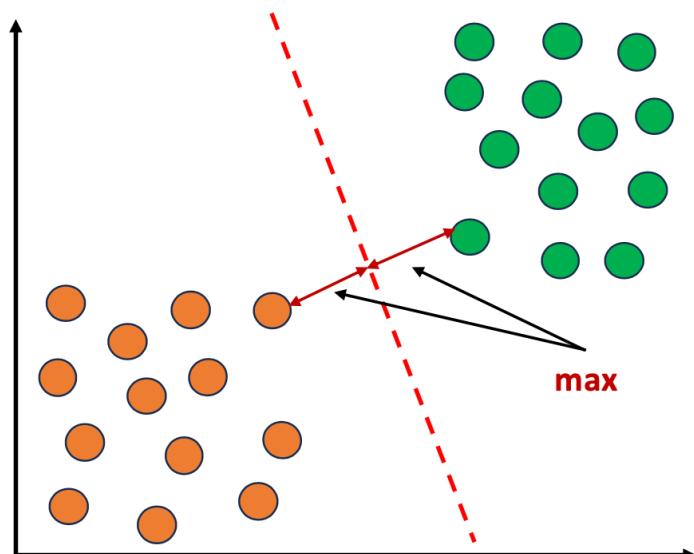
Support Vector Machine

- Việc margin rộng hơn sẽ mang lại hiệu quả phân lớp tốt hơn vì sự phân chia giữa hai lớp là rạch ròi hơn.
- Bài toán tối ưu trong SVM chính là bài toán đi tìm đường phân chia sao cho margin là lớn nhất.



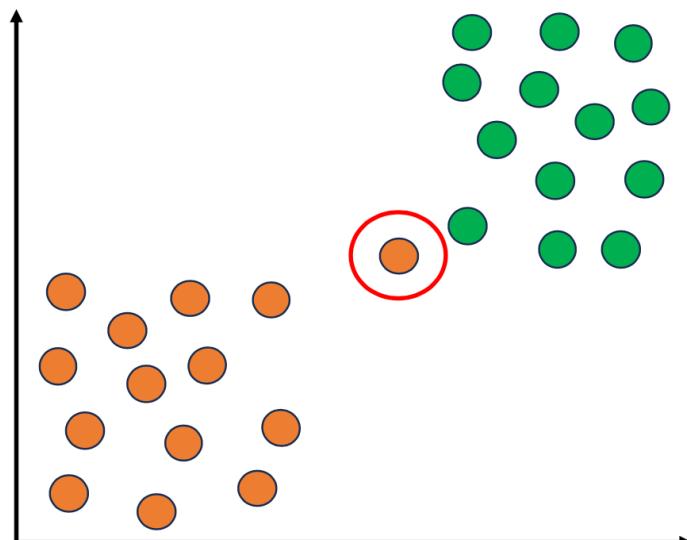
Support Vector Machine – Hard Margin

- Mục tiêu là tìm siêu phẳng có thể phân loại hoàn toàn dữ liệu thành hai lớp mà không có bất kỳ lỗi phân loại nào bằng cách là tìm ra biên lớn nhất có thể.

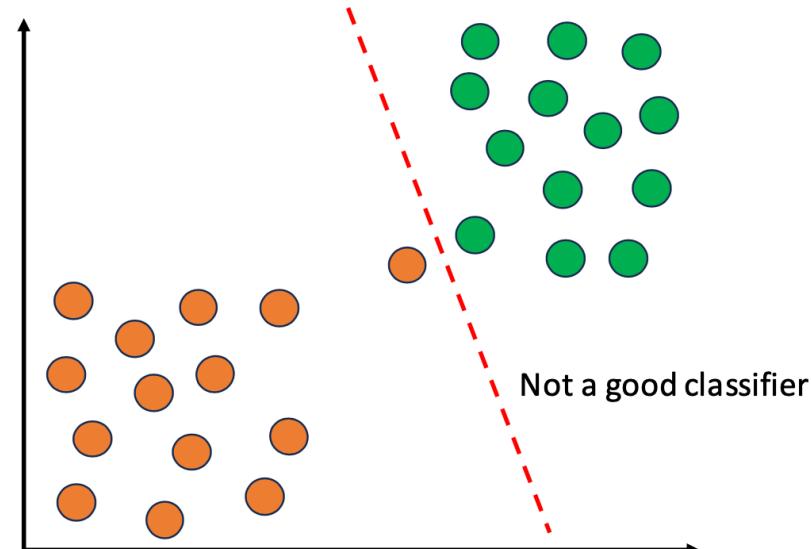


Support Vector Machine – Hard Margin

- Tuy nhiên, điều này không phải lúc nào cũng khả thi khi dữ liệu không thể phân tách tuyến tính hoặc chứa các điểm ngoại lệ..



However, assume we have an outlier

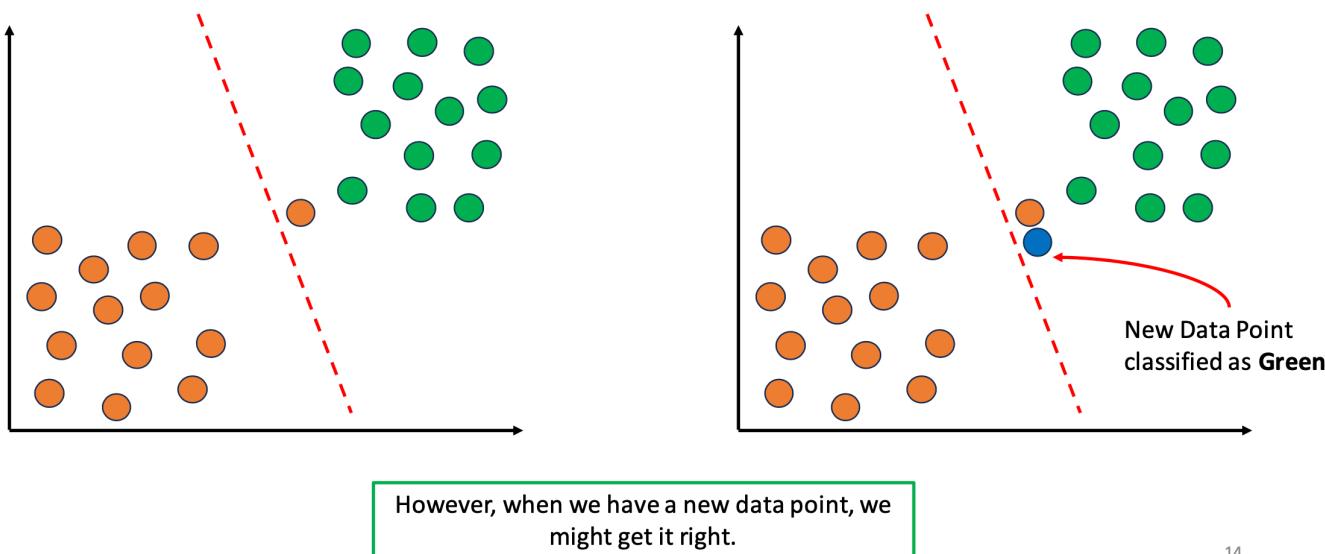


Using Hard Margin SVM, we might have a line like this.

12

Support Vector Machine – Soft Margin

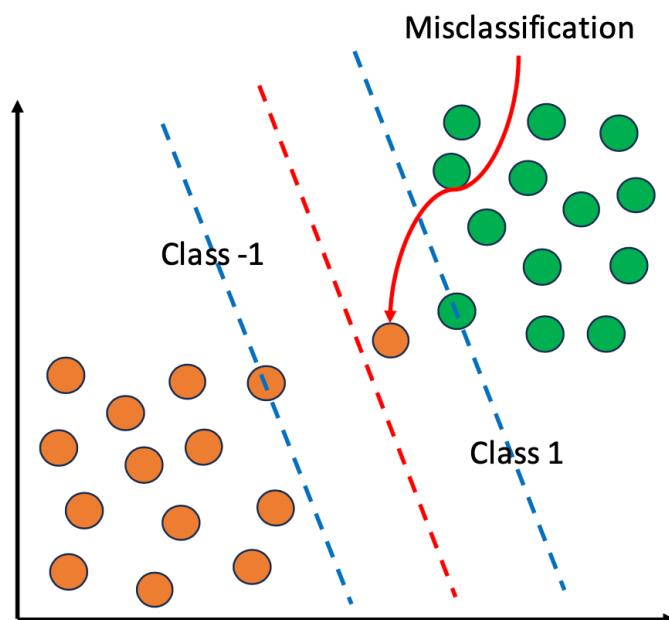
- **Soft Margin SVM** là một biến thể của SVM, sử dụng trong trường hợp dữ liệu được phân tách tuyến tính, khi chúng ta không thể vẽ được đường thẳng để phân loại các điểm dữ liệu.



14

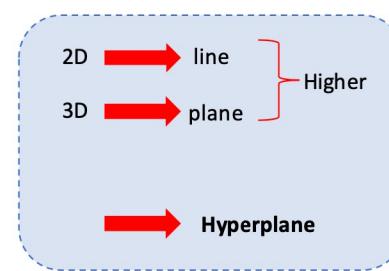
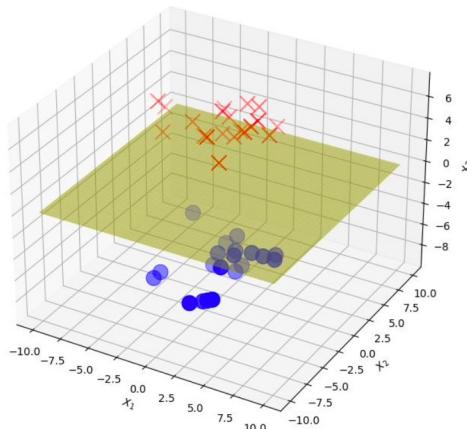
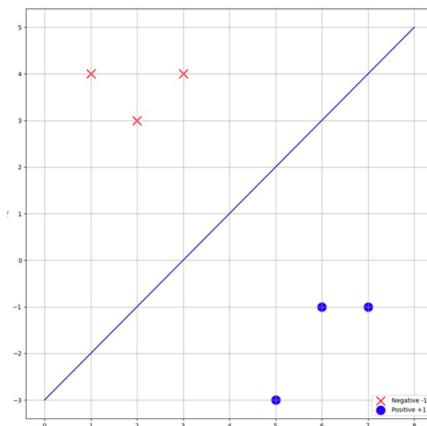
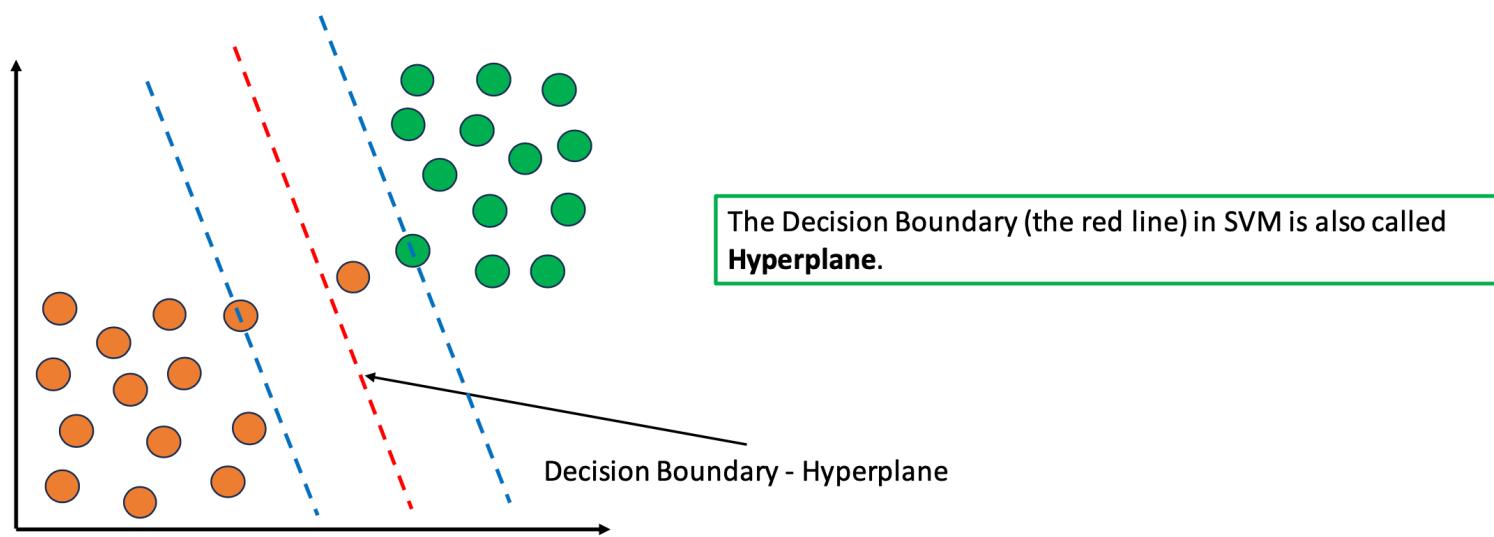
Support Vector Machine – Soft Margin

- Thuật toán này cho phép SVM mắc một số lỗi nhất định với mục tiêu giữ cho lề càng rộng càng tốt và các điểm khác vẫn được phân loại chính xác.
- **Soft Margin SVM** sẽ cân bằng giữa việc phân loại sai và tối đa hóa lề. Sự cho phép vi phạm một số điểm dữ liệu giúp giảm thiểu overfitting (quá khớp) và làm cho mô hình SVM linh hoạt hơn trong việc xử lý các tập dữ liệu có độ phức tạp cao hơn



To better have a sense of relation between data points and Soft Margin, we draw two parallel lines to the Decision Boundary on **Support Vectors**.

Support Vector Machine – Soft Margin



Support Vector Machine

- Cài đặt python

Một số phương pháp đánh giá

- Khi thực hiện bài toán phân loại, có 4 trường hợp của dự đoán có thể xảy ra:
 - True Positive (TP)**: đối tượng ở lớp Positive, mô hình phân đối tượng vào lớp Positive (dự đoán đúng)
 - True Negative (TN)**: đối tượng ở lớp Negative, mô hình phân đối tượng vào lớp Negative (dự đoán đúng)
 - False Positive (FP)**: đối tượng ở lớp Negative, mô hình phân đối tượng vào lớp Positive (dự đoán sai) – Type I Error
 - False Negative (FN)**: đối tượng ở lớp Positive, mô hình phân đối tượng vào lớp Negative (dự đoán sai) – Type II Error

		Predicted Class		
		Positive	Negative	
Actual Class	Positive	True Positive (TP) Type II Error	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP + FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN + FP)}$
		Precision $\frac{TP}{(TP + FP)}$	Negative Predictive Value $\frac{TN}{(TN + FN)}$	Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$

Một số phương pháp đánh giá

- **Accuracy** được định nghĩa là tỷ lệ phần trăm dự đoán đúng cho dữ liệu thử nghiệm. Nó có thể được tính toán bằng cách chia số lần dự đoán đúng cho tổng số lần dự đoán.

$$\text{Accuracy} = \frac{TP + TN}{\text{total sample}}$$

Một số phương pháp đánh giá

- **Precision** trả lời cho câu hỏi trong các trường hợp được dự báo là positive thì có bao nhiêu trường hợp là đúng ?
- precision càng cao thì mô hình của chúng ta càng tốt trong việc phân loại

$$\text{Precision} = \frac{TP}{\text{total predicted positive}}$$

- Precision thể hiện sự chuẩn xác của việc phát hiện các điểm Positive. Số này càng cao thì model nhận các điểm Positive càng chuẩn

Một số phương pháp đánh giá

- Recall(độ bao phủ) đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive. Công thức của recall như sau:

$$\text{Recall} = \frac{TP}{\text{total actual positive}}$$

- Recall thể hiện khả năng phát hiện tất cả các Positive, tỉ lệ này càng cao thì khả năng bỏ sót các điểm càng Positive càng thấp

Một số phương pháp đánh giá

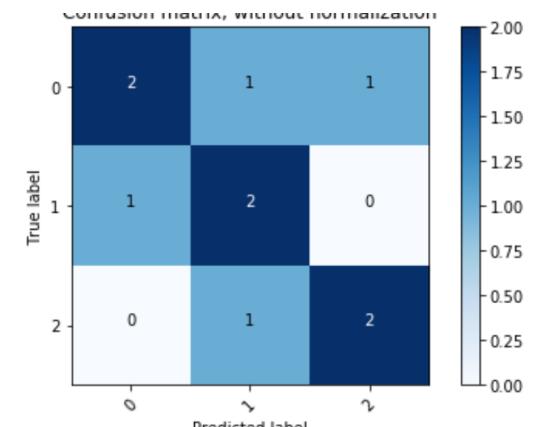
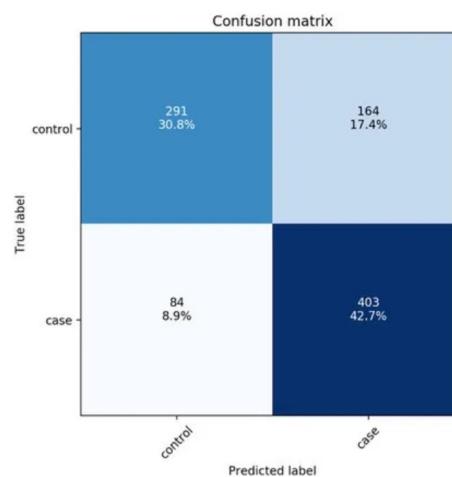
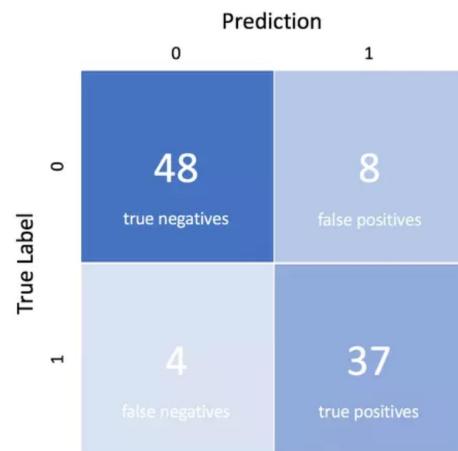
- **F1-Score** là trung bình điều hòa giữa precision và recall. Do đó nó đại diện hơn trong việc đánh giá độ chính xác trên đồng thời precision và recall.
- **F1-Score** càng lớn càng tốt

$$F_1 = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}} :$$

Một số phương pháp đánh giá

▪ Confusion Matrix :

- ma trận nhầm lẫn hay ma trận lỗi là một bộ cục bảng cụ thể cho phép hình dung hiệu suất của một thuật toán, một trong những kỹ thuật đo lường hiệu suất phổ biến nhất và được sử dụng rộng rãi cho các mô hình phân loại.
- chỉ ra được cụ thể mỗi loại được phân loại như thế nào, lớp nào được phân loại đúng nhiều nhất, và dữ liệu thuộc lớp nào thường bị phân loại nhầm vào lớp khác.



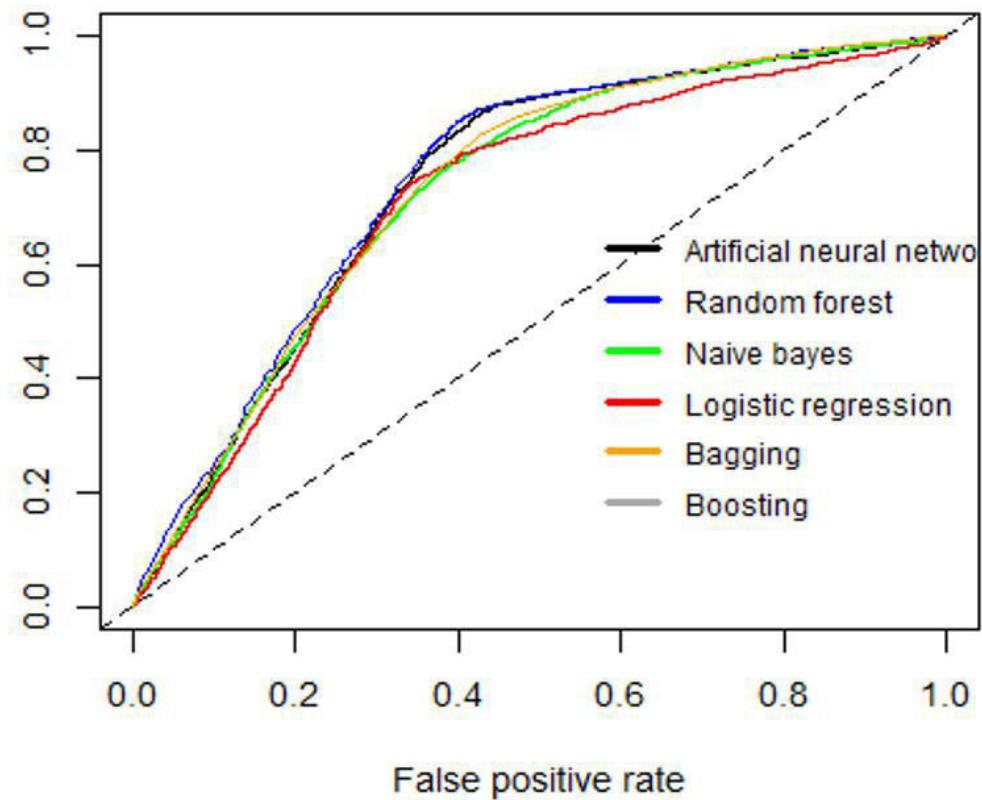
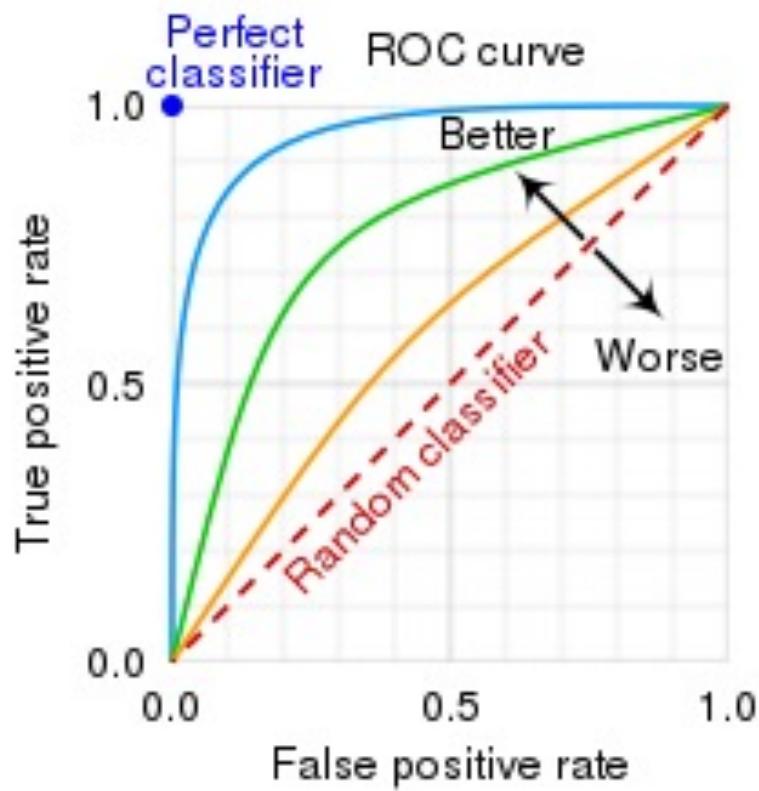
Một số phương pháp đánh giá

ROC (Receiver Operating Characteristics) là đường cong biểu diễn khả năng phân loại của một mô hình phân loại tại các ngưỡng threshold. Đường cong này dựa trên hai chỉ số :

- TPR (true positive rate): Chỉ số này chính bằng *độ phủ*. Một số tài liệu thống kê còn gọi chúng là *độ nhạy*(sensitivity).
- FPR (false positive rate): Tỷ lệ dự báo sai các trường hợp thực tế là *âm tính* thành *dương tính* trên tổng số các trường hợp thực tế là *âm tính*.
- ROC thể hiện sự tương quan giữa Precision và Recall khi thay đổi threshold

Một số phương pháp đánh giá

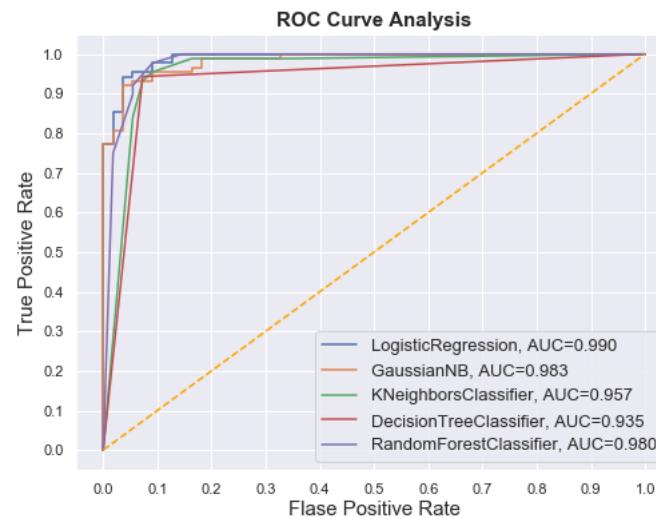
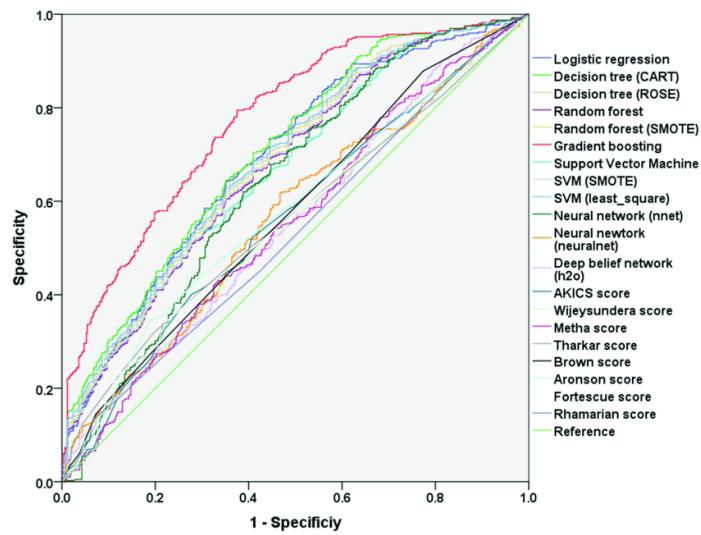
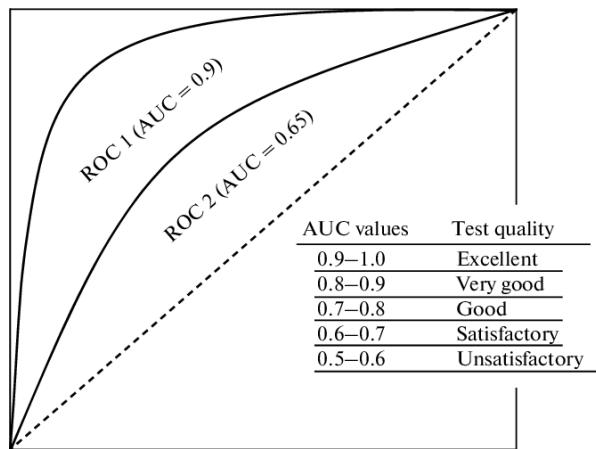
■ Ví dụ



Một số phương pháp đánh giá

- AUC là chỉ số được tính toán dựa trên đường cong ROC (receiving operating curve) nhằm **đánh giá khả năng phân loại** của mô hình tốt như thế nào ?
- Phần diện tích gạch chéo nằm dưới đường cong ROC và trên trực hoành là AUC (area under curve) có giá trị nằm trong khoảng [0, 1].
 - Khi diện tích này càng lớn thì đường cong ROC có xu hướng tiệm cận đường thẳng $y=1$ và khả năng phân loại của mô hình càng tốt.
 - Khi đường cong ROC nằm sát với đường chéo chính đi qua hai điểm $(0, 0)$ và $(1, 1)$, mô hình sẽ tương đương với một phân loại ngẫu nhiên.
 - Trường hợp đường cong ROC nằm dưới đường chéo chính thể hiện rằng mô hình có chất lượng kém và thậm chí thua xa một dự báo ngẫu nhiên.
- AUC : là phần nằm dưới ROC, vùng càng lớn thì model càng tốt

Một số phương pháp đánh giá



Q &A
