

# Nhập môn Học máy và Khai phá dữ liệu (IT3190)

**Nguyễn Nhật Quang**

*quang.nguyennhat@hust.edu.vn*

---

Trường Đại học Bách Khoa Hà Nội  
Viện Công nghệ thông tin và truyền thông  
Năm học 2021-2022

# Nội dung môn học:

- Giới thiệu về Học máy và Khai phá dữ liệu
- Tiền xử lý dữ liệu
- Đánh giá hiệu năng của hệ thống
- Hồi quy
- **Phân lớp**
  - **Bài toán phân lớp**
  - **Học dựa trên các láng giềng gần nhất (Nearest neighbors learning)**
- Phân cụm
- Phát hiện luật kết hợp

# Bài toán phân lớp

- Phân lớp (classification) thuộc nhóm bài toán học có giám sát (supervised learning)
- Mục tiêu của bài toán phân lớp là dự đoán một giá trị rời rạc (kiểu định danh)

$$f: X \rightarrow Y$$

trong đó,  $Y$  là tập hữu hạn các giá trị rời rạc (discrete values)

# Bài toán phân lớp: Đánh giá hiệu năng

$$Accuracy = \frac{1}{|D_{test}|} \sum_{x \in D_{test}} Identical(o(x), c(x));$$

$$Identical(a, b) = \begin{cases} 1, & \text{if } (a = b) \\ 0, & \text{if otherwise} \end{cases}$$

- $x$ : Một ví dụ trong tập thử nghiệm  $D_{test}$
- $o(x)$ : Phân lớp đưa ra bởi hệ thống đối với ví dụ  $x$
- $c(x)$ : Phân lớp thực sự (đúng) đối với ví dụ  $x$

# Ma trận nhầm lẫn (Confusion matrix)

- Còn được gọi là Contingency Table
- **Chỉ được sử dụng đối với bài toán phân lớp**
  - ▣ Không thể áp dụng cho bài toán hồi quy (dự đoán)
- **$TP_i$** : Số lượng các ví dụ thuộc lớp  $c_i$  được phân loại chính xác vào lớp  $c_i$
- **$FP_i$** : Số lượng các ví dụ không thuộc lớp  $c_i$  bị phân loại nhầm vào lớp  $c_i$
- **$TN_i$** : Số lượng các ví dụ không thuộc lớp  $c_i$  được phân loại chính xác
- **$FN_i$** : Số lượng các ví dụ thuộc lớp  $c_i$  bị phân loại nhầm (vào các lớp khác  $c_i$ )

Lớp $c_i$		Được phân lớp bởi hệ thống	
		Thuộc	Ko thuộc
Phân lớp thực sự (đúng)	Thuộc	<b><math>TP_i</math></b>	<b><math>FN_i</math></b>
	Ko thuộc	<b><math>FP_i</math></b>	<b><math>TN_i</math></b>

# Precision and Recall (1)

- Rất hay được sử dụng để đánh giá các hệ thống phân lớp văn bản (document classification)

- **Precision** đối với lớp  $c_i$ 
  - Tổng số các ví dụ thuộc lớp  $c_i$  được phân lớp chính xác chia cho tổng số các ví dụ được phân lớp vào lớp  $c_i$

$$Precision(c_i) = \frac{TP_i}{TP_i + FP_i}$$

- **Recall** đối với lớp  $c_i$ 
  - Tổng số các ví dụ thuộc lớp  $c_i$  được phân lớp chính xác chia cho tổng số các ví dụ thuộc lớp  $c_i$

$$Recall(c_i) = \frac{TP_i}{TP_i + FN_i}$$

# Precision and Recall (2)

- Làm thế nào để tính toán được giá trị Precision và Recall (một cách tổng thể) cho toàn bộ các lớp  $C=\{c_i\}$ ?

- Trung bình vi mô (Micro-averaging)

$$\text{Precision} = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FP_i)} \quad \text{Recall} = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FN_i)}$$

- Trung bình vĩ mô (Macro-averaging)

$$\text{Precision} = \frac{\sum_{i=1}^{|C|} \text{Precision}(c_i)}{|C|} \quad \text{Recall} = \frac{\sum_{i=1}^{|C|} \text{Recall}(c_i)}{|C|}$$

# $F_1$

- Tiêu chí đánh giá  $F_1$  là sự kết hợp của 2 tiêu chí đánh giá *Precision* và *Recall*

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

- $F_1$  là một **trung bình điều hòa (harmonic mean)** của các tiêu chí *Precision* và *Recall*
  - $F_1$  có xu hướng lấy giá trị gần với giá trị nào nhỏ hơn giữa 2 giá trị *Precision* và *Recall*
  - $F_1$  có giá trị lớn nếu cả 2 giá trị *Precision* và *Recall* đều lớn



# Top-k accuracy

- Suitable for ranking (i.e., learning to rank) problems
  - E.g., Ranked lists resulted by a search/retrieval engine

$$Accuracy = \frac{1}{|D_{test}|} \sum_{x \in D_{test}} InList(ol(x), c(x))$$

- $x$  is an example in the test set  $D_{test}$
- $ol(x)$  is the ranked list of class labels produced by the system for the example  $x$
- $c(x)$  is the true (expected/real) class label for the example  $x$
- $InList(ol(x), c(x))$  is equal to 1 if the class label  $c(x)$  appears in the ranked list  $ol(x)$ , and equal to 0 if otherwise

# Học dựa trên các láng giềng gần nhất

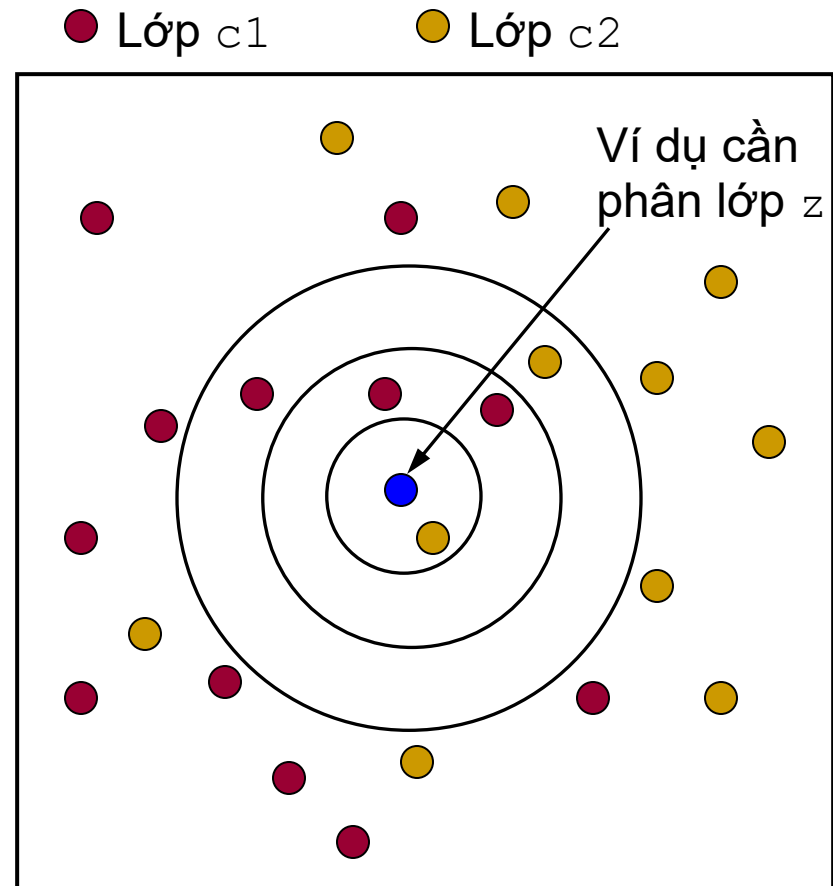
- Một số tên gọi khác của phương pháp học dựa trên các láng giềng gần nhất (Nearest neighbor learning)
  - Instance-based learning
  - Lazy learning
  - Memory-based learning
- Ý tưởng của phương pháp học dựa trên các láng giềng gần nhất
  - Với một tập các ví dụ học
    - (Đơn giản là) lưu lại các ví dụ học
    - Chưa xây dựng một mô hình (mô tả) rõ ràng và tổng quát của hàm mục tiêu cần học
  - Đối với một ví dụ cần phân loại/hồi quy
    - Xét quan hệ giữa ví dụ đó với các ví dụ học để gán giá trị của hàm mục tiêu (một nhãn lớp, hoặc một giá trị thực)

# Học dựa trên các láng giềng gần nhất

- Biểu diễn đầu vào của bài toán
  - Mỗi ví dụ  $x$  được biểu diễn là một vector  $n$  chiều trong không gian các vector  $X \in \mathbb{R}^n$
  - $x = (x_1, x_2, \dots, x_n)$ , trong đó  $x_i (\in \mathbb{R})$  là một số thực
- Có thể áp dụng được với cả 2 kiểu bài toán học
  - Bài toán *phân lớp (classification)*
    - Hàm mục tiêu có giá trị rời rạc (a discrete-valued target function)
    - Đầu ra của hệ thống là một trong số các giá trị rời rạc đã xác định trước (một trong các nhãn lớp)
  - Bài toán *hồi quy (regression)*
    - Hàm mục tiêu có giá trị liên tục (a continuous-valued target function)
    - Đầu ra của hệ thống là một giá trị số thực

# Ví dụ bài toán phân lớp

- Xét 1 láng giềng gần nhất  
→ Gán  $z$  vào lớp  $c_2$
- Xét 3 láng giềng gần nhất  
→ Gán  $z$  vào lớp  $c_1$
- Xét 5 láng giềng gần nhất  
→ Gán  $z$  vào lớp  $c_1$



# Giải thuật phân lớp k-NN

- Mỗi ví dụ học  $x$  được biểu diễn bởi 2 thành phần:
  - Mô tả của ví dụ:  $x = (x_1, x_2, \dots, x_n)$ , trong đó  $x_i \in \mathbb{R}$
  - Nhãn lớp:  $c \in C$ , với  $C$  là tập các nhãn lớp được xác định trước)
- Giai đoạn học
  - Đơn giản là lưu lại các ví dụ học trong tập học:  $D = \{x\}$
- Giai đoạn phân lớp: Để phân lớp cho một ví dụ (mới)  $z$ 
  - Với mỗi ví dụ học  $x \in D$ , tính khoảng cách giữa  $x$  và  $z$
  - Xác định tập  $NB(z)$  – các láng giềng gần nhất của  $z$ 
    - Gồm  $k$  ví dụ học trong  $D$  gần nhất với  $z$  tính theo một hàm khoảng cách  $d$
  - Phân  $z$  vào lớp chiếm số đông (the majority class) trong số các lớp của các ví dụ học trong  $NB(z)$

# Giải thuật hồi quy k-NN

- Mỗi ví dụ học  $x$  được biểu diễn bởi 2 thành phần:
  - Mô tả của ví dụ:  $x = (x_1, x_2, \dots, x_n)$ , trong đó  $x_i \in \mathbb{R}$
  - Giá trị đầu ra mong muốn:  $y_x \in \mathbb{R}$  (là một số thực)
- Giai đoạn học
  - Đơn giản là lưu lại các ví dụ học trong tập học  $D$
- Giai đoạn dự đoán: Để dự đoán giá trị đầu ra cho ví dụ  $z$ 
  - Đối với mỗi ví dụ học  $x \in D$ , tính khoảng cách giữa  $x$  và  $z$
  - Xác định tập  $NB(z)$  – các láng giềng gần nhất của  $z$ 
    - Gồm  $k$  ví dụ học trong  $D$  gần nhất với  $z$  tính theo một hàm khoảng cách  $d$
  - Dự đoán giá trị đầu ra đối với  $z$ :

$$y_z = \frac{1}{k} \sum_{x \in NB(z)} y_x$$

# Một hay nhiều láng giềng gần nhất?

- Việc phân lớp (hay dự đoán) chỉ dựa trên duy nhất một láng giềng gần nhất (là ví dụ học gần nhất với ví dụ cần phân lớp/dự đoán) thường *không* chính xác
  - Nếu ví dụ học này là một ví dụ bất thường, không điển hình (an outlier) – rất khác so với các ví dụ khác
  - Nếu ví dụ học này có nhãn lớp (giá trị đầu ra) sai – do lỗi trong quá trình thu thập (xây dựng) tập dữ liệu
- Thường xét  $k$  ( $>1$ ) các ví dụ học (các láng giềng) gần nhất với ví dụ cần phân lớp/dự đoán
- Đối với bài toán phân lớp có 2 lớp,  $k$  thường được chọn là một số lẻ, để tránh cân bằng về tỷ lệ các ví dụ giữa 2 lớp
  - Ví dụ:  $k=3, 5, 7, \dots$

# Hàm tính khoảng cách (1)

## ■ Hàm tính khoảng cách $d$

- Đóng vai trò rất quan trọng trong phương pháp học dựa trên các láng giềng gần nhất
- Thường được xác định trước, và không thay đổi trong suốt quá trình học và phân loại/dự đoán

## ■ Lựa chọn hàm khoảng cách $d$

- *Các hàm khoảng cách hình học*: Dành cho các bài toán có các thuộc tính đầu vào là kiểu số thực ( $x_i \in \mathbb{R}$ )
- *Hàm khoảng cách Hamming*: Dành cho các bài toán có các thuộc tính đầu vào là kiểu nhị phân ( $x_i \in \{0,1\}$ )
- *Hàm tính độ tương tự Cosine*: Dành cho các bài toán phân lớp văn bản ( $x_i$  là giá trị trọng số TF/IDF của từ khóa thứ  $i$ )



# Hàm tính khoảng cách (2)

## ■ Các hàm tính khoảng cách hình học (Geometry distance functions)

- Hàm Minkowski ( $p$ -norm):

$$d(x, z) = \left( \sum_{i=1}^n |x_i - z_i|^p \right)^{1/p}$$

- Hàm Manhattan ( $p=1$ ):

$$d(x, z) = \sum_{i=1}^n |x_i - z_i|$$

- Hàm Euclid ( $p=2$ ):

$$d(x, z) = \sqrt{\sum_{i=1}^n (x_i - z_i)^2}$$

- Hàm Chebyshev ( $p=\infty$ ):

$$d(x, z) = \lim_{p \rightarrow \infty} \left( \sum_{i=1}^n |x_i - z_i|^p \right)^{1/p} \\ = \max_i |x_i - z_i|$$

# Hàm tính khoảng cách (3)

## ■ Hàm khoảng cách Hamming

- Đối với các thuộc tính đầu vào là kiểu nhị phân ( $\{0,1\}$ )
- Ví dụ:  $x=(0,1,0,1,1)$

$$d(x, z) = \sum_{i=1}^n \text{Difference}(x_i, z_i)$$

$$\text{Difference}(a, b) = \begin{cases} 1, & \text{if } (a \neq b) \\ 0, & \text{if } (a = b) \end{cases}$$

## ■ Hàm tính độ tương tự Cosine

- Đối với đầu vào là một vector các giá trị trọng số (TF/IDF) của các từ khóa

$$\text{sim}(x, z) = \frac{x \cdot z}{\|x\| \|z\|} = \frac{\sum_{i=1}^n x_i z_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n z_i^2}}$$

# Chuẩn hóa miền giá trị thuộc tính

- Hàm tính khoảng cách Euclid: 
$$d(x, z) = \sqrt{\sum_{i=1}^n (x_i - z_i)^2}$$
- Giả sử mỗi ví dụ được biểu diễn bởi 3 thuộc tính: Age, Income (cho mỗi tháng), và Height (đo theo mét)
  - $x = (\text{Age}=20, \text{Income}=12000, \text{Height}=1.48)$
  - $z = (\text{Age}=40, \text{Income}=13000, \text{Height}=1.85)$
- Khoảng cách giữa  $x$  và  $z$ 
  - $d(x, z) = [(20-40)^2 + (12000-13000)^2 + (1.48-1.85)^2]^{1/2}$
  - Giá trị khoảng cách bị quyết định chủ yếu bởi giá trị khoảng cách (sự khác biệt) giữa 2 ví dụ đối với thuộc tính Income
    - Vì: Thuộc tính Income có miền giá trị rất lớn so với các thuộc tính khác
- Cần phải chuẩn hóa miền giá trị (đưa về cùng một khoảng giá trị)
  - Khoảng giá trị  $[0,1]$  thường được sử dụng
  - Đối với mỗi thuộc tính  $i$ :  $x_i = x_i / \max\_val(f_i)$

# Trọng số của các thuộc tính

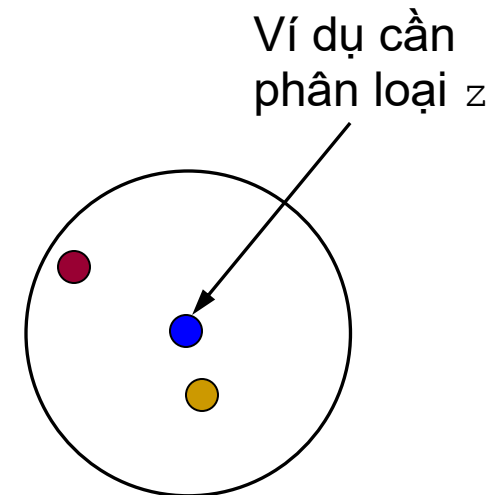
- Hàm khoảng cách Euclid:

$$d(x, z) = \sqrt{\sum_{i=1}^n (x_i - z_i)^2}$$

- Tất cả các thuộc tính có cùng (như nhau) ảnh hưởng đối với giá trị khoảng cách
- **Các thuộc tính khác nhau có thể (nên) có mức độ ảnh hưởng khác nhau** đối với giá trị khoảng cách
- Cần phải tích hợp (đưa vào) các giá trị trọng số của các thuộc tính trong hàm tính khoảng cách
$$d(x, z) = \sqrt{\sum_{i=1}^n w_i (x_i - z_i)^2}$$
  - $w_i$  là trọng số của thuộc tính  $i$ :
- Làm sao để xác định các giá trị trọng số của các thuộc tính?
  - Dựa trên các tri thức cụ thể của bài toán (vd: được chỉ định bởi các chuyên gia trong lĩnh vực của bài toán đang xét)
  - Bằng một quá trình tối ưu hóa các giá trị trọng số (vd: sử dụng một tập học để học một bộ các giá trị trọng số tối ưu)

# Khoảng cách của các láng giềng (1)

- Xét tập  $NB(z)$  – gồm  $k$  ví dụ học gần nhất với ví dụ cần phân lớp/dự đoán  $z$ 
  - Mỗi ví dụ (láng giềng gần nhất) này có khoảng cách khác nhau đến  $z$
  - Các láng giềng này có ảnh hưởng như nhau đối với việc phân lớp/dự đoán cho  $z$ ? → KHÔNG!
- Cần gán các mức độ ảnh hưởng (đóng góp) của mỗi láng giềng gần nhất tùy theo khoảng cách của nó đến  $z$ 
  - Mức độ ảnh hưởng cao hơn cho các láng giềng gần hơn!



# Khoảng cách của các láng giềng (2)

- Gọi  $v$  là hàm xác định trọng số theo khoảng cách

- Đối với một giá trị  $d(x, z)$  – khoảng cách giữa  $x$  và  $z$
- $v(x, z)$  tỷ lệ nghịch với  $d(x, z)$

- Đối với bài toán phân lớp: 
$$c(z) = \arg \max_{c_j \in C} \sum_{x \in NB(z)} v(x, z) \cdot \text{Identical}(c_j, c(x))$$

$$\text{Identical}(a, b) = \begin{cases} 1, & \text{if } (a = b) \\ 0, & \text{if } (a \neq b) \end{cases}$$

- Đối với bài toán dự đoán (hồi quy): 
$$f(z) = \frac{\sum_{x \in NB(z)} v(x, z) \cdot f(x)}{\sum_{x \in NB(z)} v(x, z)}$$

- Lựa chọn một hàm xác định trọng số theo khoảng cách:

$$v(x, z) = \frac{1}{\alpha + d(x, z)}$$

$$v(x, z) = \frac{1}{\alpha + [d(x, z)]^2}$$

$$v(x, z) = e^{-\frac{d(x, z)^2}{\sigma^2}}$$

# Lazy learning vs. Eager learning

- **Lazy learning.** Việc đánh giá hàm mục tiêu (target function) được hoãn lại cho đến khi xét ví dụ cần phân loại/dự đoán
  - Đánh giá (xấp xỉ) hàm mục tiêu *một cách cục bộ (locally)* và *riêng rẽ (diferrently)* cho mỗi ví dụ cần phân loại/dự đoán (*tại thời điểm phân loại/dự đoán của hệ thống*)
  - Tính toán nhiều lần các xấp xỉ *cục bộ* của hàm mục tiêu
  - Thường mất thời gian lâu hơn để đưa ra kết luận (phân lớp/dự đoán), và cần nhiều không gian nhớ hơn
  - Ví dụ: Nearest neighbor learner, Locally weighted regression
- **Eager learning.** Việc đánh giá hàm mục tiêu được hoàn thành trước khi xét đến bất kỳ ví dụ cần phân loại/dự đoán
  - Đánh giá (xấp xỉ) hàm mục tiêu *một cách tổng thể (globally)* đối với toàn bộ không gian các ví dụ (*tại thời điểm học của hệ thống*)
  - Tính toán một xấp xỉ *duy nhất (ở mức tổng thể)* của hàm mục tiêu
  - Ví dụ: Linear regression, Support vector machines, Neural networks, ...

# k-NN – Khi nào?

- Các ví dụ được biểu diễn trong không gian vectơ  $\mathbb{R}^n$
- Số lượng các thuộc tính (để biểu diễn ví dụ) là không nhiều
- Một tập học có kích thước lớn
- Các ưu điểm
  - Chi phí thấp cho quá trình huấn luyện (chỉ việc lưu lại các ví dụ học)
  - Hoạt động tốt với các bài toán phân loại gồm nhiều lớp
    - Không cần phải học  $n$  bộ phân loại cho  $n$  lớp
  - Phương pháp học k-NN ( $k \gg 1$ ) có khả năng xử lý nhiễu cao
    - Phân loại/dự đoán được thực hiện dựa trên  $k$  láng giềng gần nhất
- Các nhược điểm
  - Xác định giá trị cho tham số  $k$  (hoặc ngưỡng lựa chọn về sự khác biệt)
  - Phải lựa chọn hàm tính khoảng cách (sự khác biệt) thích hợp với bài toán
  - Chi phí tính toán (thời gian, bộ nhớ) cao tại thời điểm phân loại/dự đoán
  - Có thể cho kết quả kém/sai với các thuộc tính không liên quan