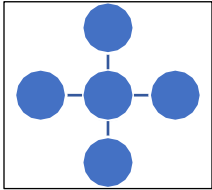


Nội dung môn học

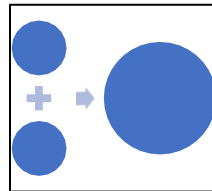
- Lecture 1: Giới thiệu về Học máy và khai phá dữ liệu
- Lecture 2: Thu thập và tiền xử lý dữ liệu
- Lecture 3: Hồi quy tuyến tính (Linear regression)
- Lecture 4+5: Phân cụm
- Lecture 6: Phân loại và Đánh giá hiệu năng
- **Lecture 7: dựa trên láng giềng gần nhất (KNN)**
- Lecture 8: Cây quyết định và Rừng ngẫu nhiên
- Lecture 9: Học dựa trên xác suất
- Lecture 10: Mạng nơron (Neural networks)
- Lecture 11: Máy vector hỗ trợ (SVM)
- Lecture 12: Khai phá tập mục thường xuyên và các luật kết hợp
- Lecture 13: Thảo luận ứng dụng học máy và khai phá dữ liệu trong thực tế

Các bạn phân loại thế nào?

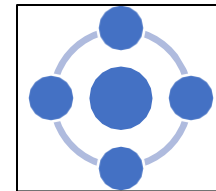
Class a



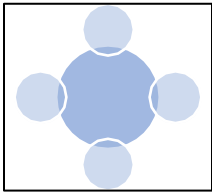
Class b



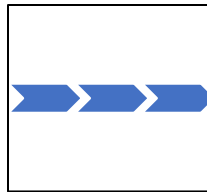
Class a



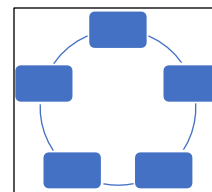
Class a



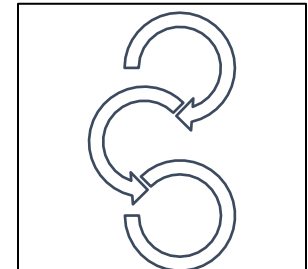
??



Class a



Class b



Học dựa trên các láng giềng gần nhất

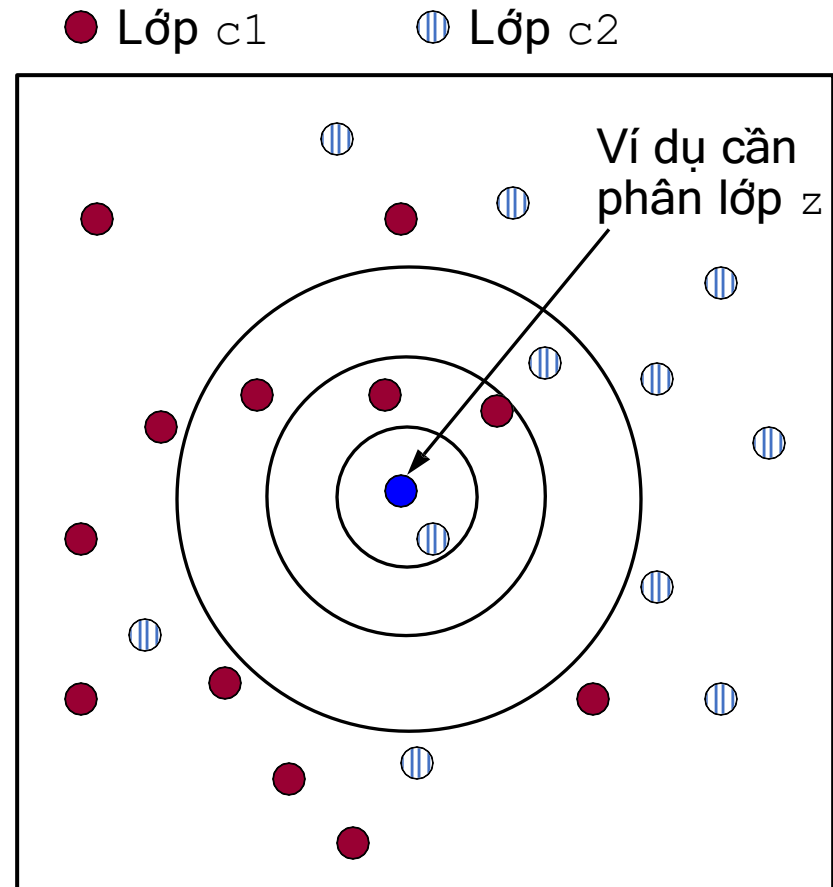
- ***K-nearest neighbors*** (k-NN) là một trong số các phương pháp phổ biến trong học máy. Vài tên gọi khác như:
 - Instance-based learning
 - Lazy learning
 - Memory-based learning
- Ý tưởng của phương pháp
 - Không xây dựng một mô hình (mô tả) rõ ràng cho hàm mục tiêu cần học.
 - Quá trình học chỉ lưu lại các dữ liệu huấn luyện.
 - Việc dự đoán cho một quan sát mới sẽ dựa vào các hàng xóm gần nhất trong tập học.
- Do đó k-NN là một phương pháp phi tham số (nonparametric methods)

k-NN

- Hai thành phần chính:
 - Độ đo tương đồng (similarity measure/distance) giữa các đối tượng.
 - Các hàng xóm sẽ dùng vào việc phán đoán.
- *Trong một số điều kiện thì k-NN có thể đạt mức lỗi tối ưu Bayes (mức lỗi mong muốn của bất kỳ phương pháp nào)*
[Gyuder and Hengartner, JMLR 2013]
 - Thậm chí khi chỉ dùng 1 hàng xóm gần nhất thì nó cũng có thể đạt đến mức lỗi tối ưu Bayes. [Kontorovich & Weiss, AISTATS 2015]

Ví dụ: bài toán phân lớp

- Xét 1 láng giềng gần nhất
→ Gán z vào lớp $c2$
- Xét 3 láng giềng gần nhất
→ Gán z vào lớp $c1$
- Xét 5 láng giềng gần nhất
→ Gán z vào lớp $c1$



Giải thuật k-NN cho phân lớp

- 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 $x_i \in R$
 - 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
- 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 trong $NB(z)$

Giải thuật k-NN cho hồi quy

- 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 $x_i \in R$
 - Giá trị đầu ra mong muốn: $y_x \in R$ (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 :

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

k-NN: Các vấn đề cốt lõi

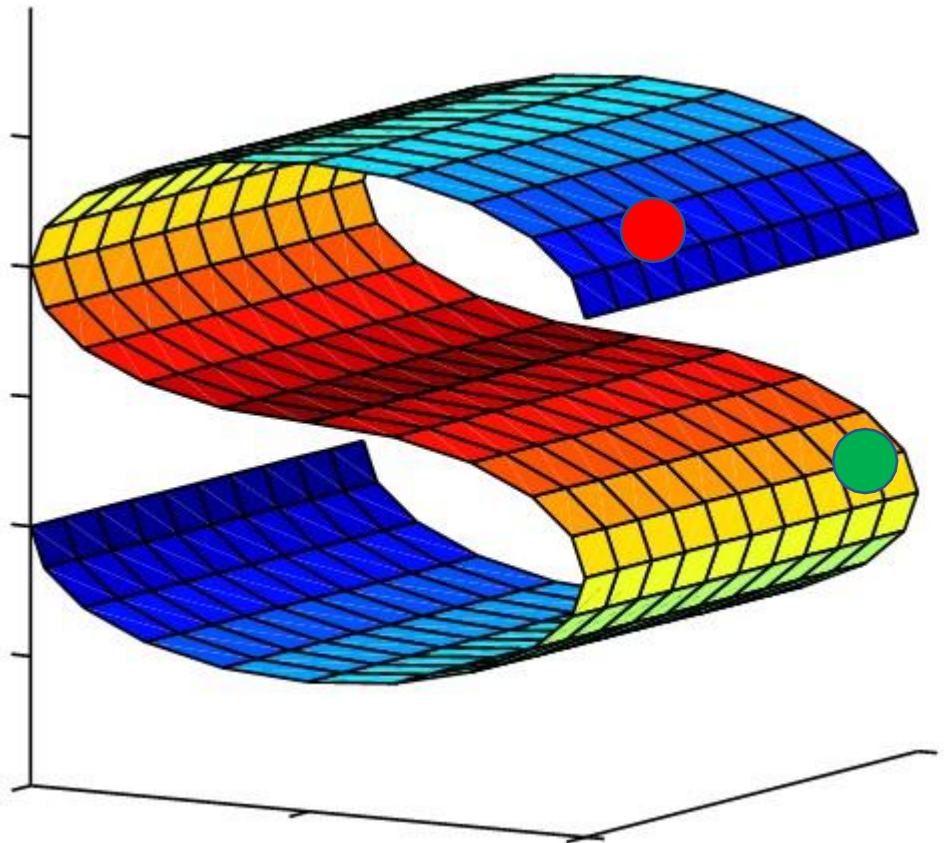


Suy
nghĩ
khác
nhau!

k-NN: Các vấn đề cốt lõi

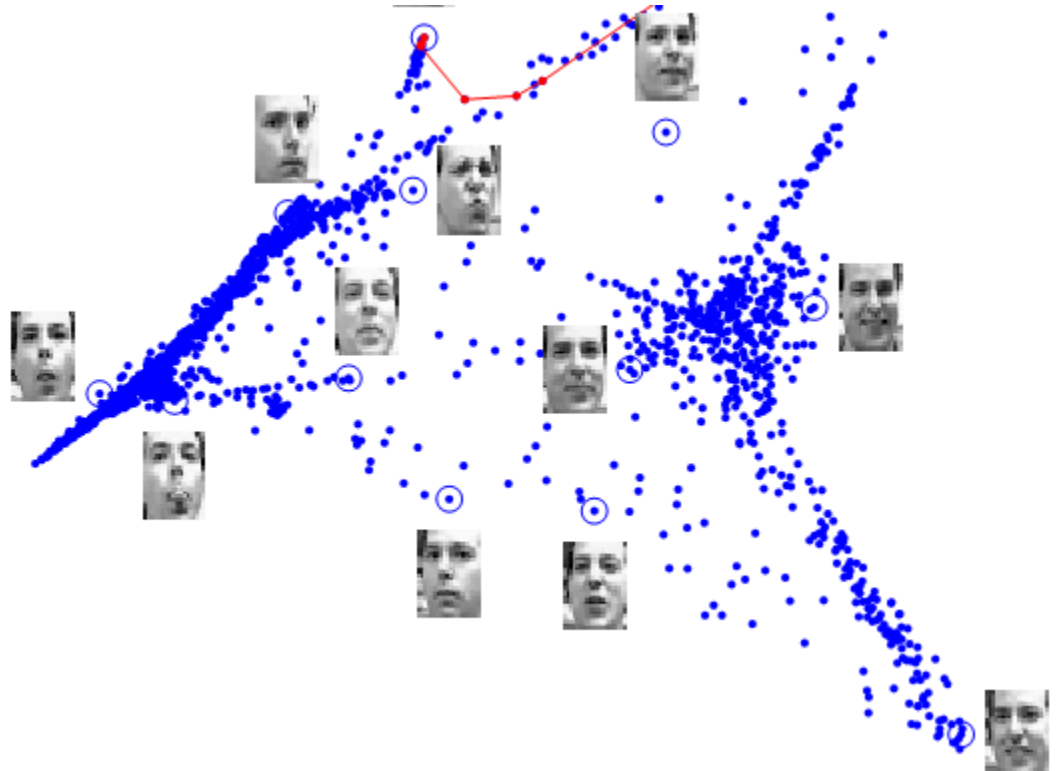
■ Hàm khoảng cách

- ❑ Mỗi hàm sẽ tương ứng với một cách nhìn về dữ liệu.
- ❑ Vô hạn hàm!!!
- ❑ Chọn hàm nào?



k-NN: Các vấn đề cốt lõi

- Chọn tập láng giềng $NB(z)$
 - Chọn bao nhiêu láng giềng?
 - Giới hạn chọn theo vùng?



k-NN: một hay nhiều láng giềng?

- Về lý thuyết thì 1-NN cũng có thể là một trong số các phương pháp tối ưu.
- k-NN là một phương pháp tối ưu Bayes nếu gặp một số điều kiện, chẳng hạn: y bị chặn, cỡ M của tập học lớn, hàm hồi quy liên tục, và

$$k \rightarrow \infty, (k/M) \rightarrow 0, (k/\log M) \rightarrow +\infty$$

- Trong thực tiễn ta nên lấy nhiều hàng xóm ($k > 1$) khi cần phân lớp/dự đoán, nhưng không quá nhiều. Lý do:
 - Tránh ảnh hưởng của lỗi/nhiều nếu chỉ dùng 1 hàng xóm.
 - Nếu quá nhiều hàng xóm thì sẽ phá vỡ cấu trúc tiềm ẩn trong dữ liệu.

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*: Có thể phù hợp với các bài toán có các thuộc tính đầu vào là kiểu số thực ($x_i \in R$)
- *Hàm khoảng cách Hamming*: Có thể phù hợp với các bài toán có các thuộc tính đầu vào là kiểu nhị phân ($x_i \in \{0,1\}$)

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 ($p = \infty$):

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

Hàm tính khoảng cách (3)

■ Hàm khoảng cách Hamming

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

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

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

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)
 - $x = (\text{Age}=20, \text{Income}=12000, \text{Height}=1.68)$
 - $z = (\text{Age}=40, \text{Income}=1300, \text{Height}=1.75)$
- Khoảng cách giữa x và z
 - $d(x, z) = [(20 - 40)^2 + (12000 - 1300)^2 + (1.68 - 1.75)^2]^{0.5}$
 - 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
- Có thể 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 : $x_i := x_i / \max(x_j)$

Trọng số của các thuộc tính

- Hàm khoảng cách Euclid:

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

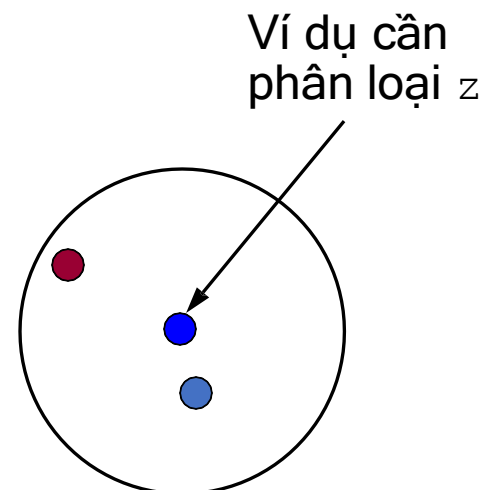
- Tất cả các thuộc tính có cùng (như nhau) ảnh hưởng đối với giá trị khoảng cách
- **Các thuộc tính khác nhau** có thể (nên) có **mức độ ảnh hưởng khác nhau** đối với giá trị khoảng cách
- Có thể phải tích hợp (đưa vào) các giá trị trọng số của các thuộc tính trong hàm tính khoảng cách

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

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

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

GT

1. Ý tưởng tổng quát

$d(x, z)$ là khoảng cách giữa hai điểm:

- x : điểm cần phân loại hoặc dự đoán
- z : một trong các láng giềng gần nhất

$v(x, z)$ là **trọng số** được gán cho láng giềng z , phụ thuộc vào khoảng cách $d(x, z)$.

- Láng giềng càng gần thì trọng số càng cao.
- Tức là:

$$v(x, z) \propto \frac{1}{d(x, z)} \text{ (nghịch đảo với khoảng cách)}$$

2. Giải quyết bài toán

♦ a. Đối với bài toán phân lớp (classification):

Công thức:

$$c(x) = \arg \max_{c_j \in C} \sum_{x_n \in NB(x)} v(x, x_n) \cdot \text{Identical}(c_j, c(x_n))$$

Giải thích:

- Ta tìm lớp c_j sao cho tổng trọng số của các láng giềng x_n thuộc lớp đó là lớn nhất.
- Hàm `Identical(a, b)`:

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

↓

=> Tức là chỉ cộng trọng số nếu điểm láng giềng thuộc lớp c_j .

♦ b. Đối với bài toán hồi quy (regression):

Công thức:

$$f(x) = \frac{\sum_{x_n \in NB(x)} v(x, x_n) \cdot f(x_n)}{\sum_{x_n \in NB(x)} v(x, x_n)}$$


Giải thích:

- $f(x_n)$: giá trị thực tại điểm láng giềng x_n .
- Kết quả là giá trị trung bình có trọng số theo khoảng cách — điểm gần có ảnh hưởng nhiều hơn.

- Nếu không dùng trọng số (truyền thống), thì:

$$f(z) = \frac{1}{K} \sum_{x_i \in NB(z)} f(x_i)$$

→ trung bình đơn giản giá trị các láng giềng.

- Nếu dùng trọng số, thì:
 - Các láng giềng **gần hơn sẽ có ảnh hưởng lớn hơn** trong việc xác định giá trị tại z .
 - Giúp **mềm hóa** mô hình, tránh nhiễu do các m xa gây ra.

Một số công thức trọng số phổ biến

3. Một số công thức trọng số phổ biến

Công thức trọng số

Diễn giải

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

Giảm tuyến tính với khoảng cách

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

Giảm nhanh hơn với khoảng cách lớn

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

Dạng Gaussian (mềm, giảm nhanh), phổ biến

Trong đó $\alpha > 0$, $\sigma > 0$ là các hệ số điều chỉnh mức độ ảnh hưởng của khoảng cách.

k-NN: Ưu nhược điểm

- 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 c bộ phân loại cho c lớp
- Về mặt lý thuyết thì k-NN có thể đạt khả năng phán đoán tối ưu khi gặp một số điều kiện.
- Rất Linh động trong việc chọn hàm khoảng cách.
 - Có thể dùng độ tương tự (similarity): cosine
 - Có thể dùng độ đo khác, chẳng hạn Kullback-Leibler divergence, Bregman divergence, ...

- Các nhược điểm

- 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

Tài liệu tham khảo

- A. Kontorovich and Weiss. A Bayes consistent 1-NN classifier. *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS). JMLR: W&CP volume 38*, 2015.
- A. Guyader, N. Hengartner. On the Mutual Nearest Neighbors Estimate in Regression. *Journal of Machine Learning Research* 14 (2013) 2361-2376.
- L. Gottlieb, A. Kontorovich, and P. Nisnevitch. Near-optimal sample compression for nearest neighbors. *Advances in Neural Information Processing Systems*, 2014.