Learning

Trí tuệ nhân tạo

HK1, 2022 - 2023

Nội dung

Tại sao tác tử cần học?

- Môi trường chưa biết
  - Ví dụ: Robot thiết kế để di chuyển trong một mê cung, khi chuyển sang một mê cung mới thì nó phải học lại cấu trúc của mê cung đó
- Môi trường thay đổi qua thời gian
  - Ví dụ: Một tác tử thiết kế dự đoán thị trường chứng khoán học lại khi có các thay đổi về điều kiện xung quanh
- Không có ý tưởng để lập trình một giải pháp
  - Nhận dạng gương mặt của thành viên gia đình

Thành phần học

Thiết kế một thành phần học ảnh hưởng bởi:

- Thành phần gì được cải thiện
- Tri thức trước (prior knowledge) mà tác tử đã có
- Biểu diễn (representation) được sử dụng
- Phản hồi (feedback) sẵn có để học các thành phần
  - Học giám sát (Supervised learning)
  - Học không giám sát (Unsupervised learning)
  - Học tăng cường (Reinforcement learning)

Giới thiệu về máy học

- Một quá trình nhờ đó một hệ thống **cải thiện hiệu suất** (hiệu quả hoạt động) của nó *[Simon, 1983]*
- Một quá trình mà một chương trình máy tính cải thiện hiệu suất của nó trong một công việc **thông qua kinh nghiệm** *[Mitchell, 1997]*
- Việc lập trình các máy tính để tối ưu hóa một tiêu chí hiệu suất dựa trên các dữ liệu **ví dụ** hoặc **kinh nghiệm** trong quá khứ *[Alpaydin, 2004]*

Giới thiệu về máy học

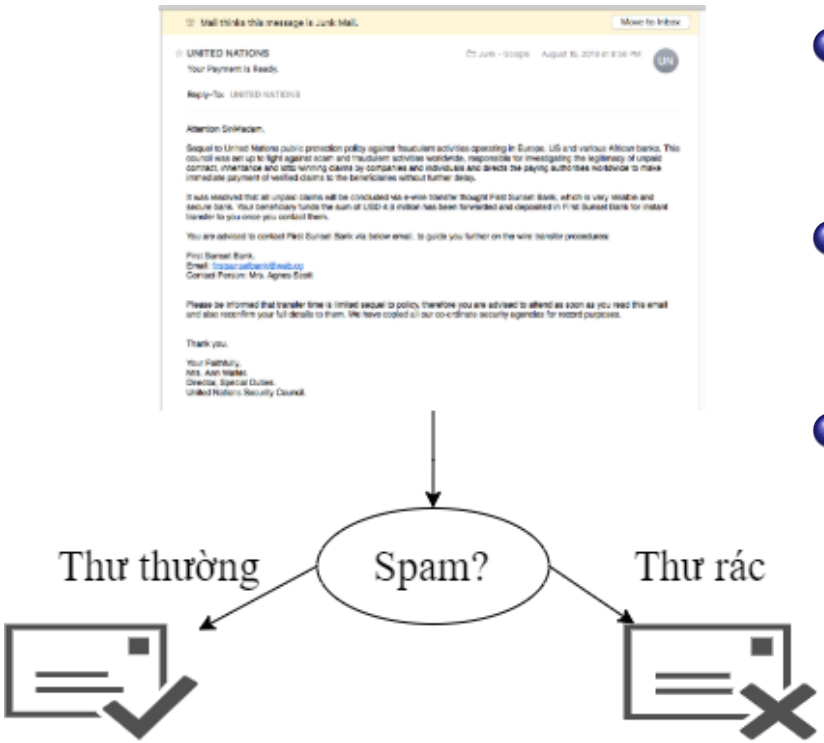
Biểu diễn một bài toán máy học

Machine learning = Cải thiện hiệu quả một công việc thông qua kinh nghiệm

- Một công việc (nhiệm vụ - Task) **T**
- Đối với các tiêu chí đánh giá hiệu suất (Performance) **P**
- Thông qua (sử dụng) kinh nghiệm (Experience) **E**

Ví dụ bài toán máy học

Lọc thư rác - Email spam filtering



- ***T***: Dự đoán (để lọc) những thư điện tử nào là thư rác (spam email)
- ***P***: tỷ lệ % các thư điện tử gửi đến được phân loại chính xác
- ***E***: Một tập các thư điện tử (emails) mẫu, mỗi thư điện tử được biểu diễn bằng một tập thuộc tính (ví dụ: tập từ khóa) và nhãn lớp (thư thường/thư rác) tương ứng

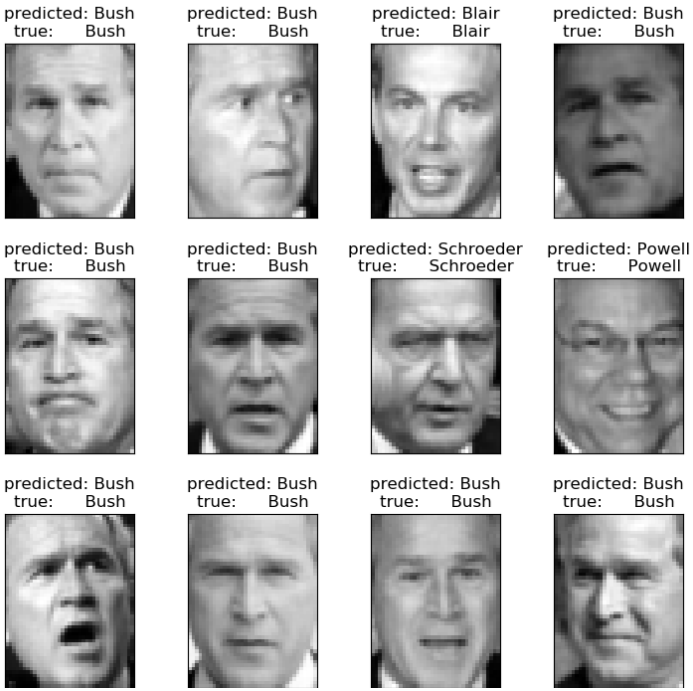
Ví dụ bài toán máy học

Phân loại các trang web theo các chủ đề

- ***T***: Phân loại các trang web theo các chủ đề đã định trước
- ***P***: tỷ lệ % các trang web được phân loại chính xác
- ***E***: Một tập các trang web, trong đó mỗi trang web gắn với một chủ đề

Ví dụ bài toán máy học

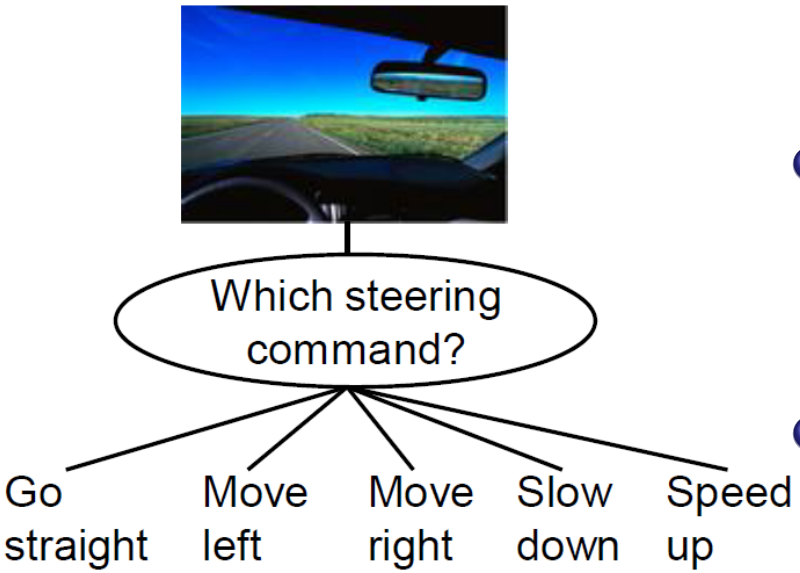
Nhận dạng gương mặt



- ***T***: Dự đoán hình gương mặt thuộc về người nào trong nhóm người đã xác định trước
- ***P***: tỷ lệ % nhận dạng chính xác
- ***E***: Một tập hình ảnh, mỗi hình ảnh được gắn với định dạng một người tương ứng

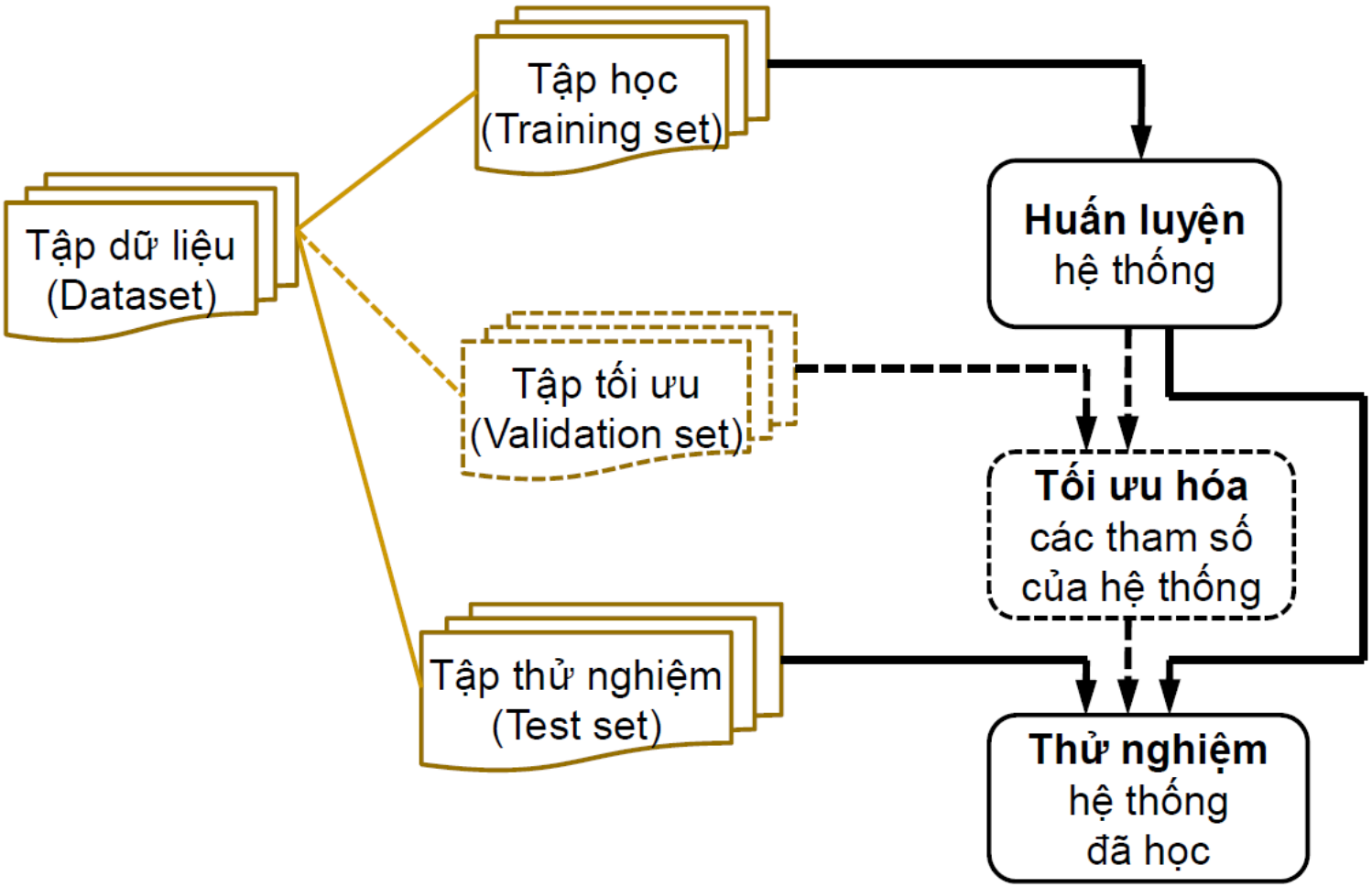
Ví dụ bài toán máy học

Bài toán robot lái xe tự động



- $\boldsymbol{T}$: Robot (được trang bị các camera quan sát) lái xe tự động trên đường
- $\boldsymbol{P}$: khoảng cách trung bình mà robot có thể lái xe tự động trước khi xảy ra lỗi (tai nạn)
- $\boldsymbol{E}$: Một tập các ví dụ được ghi lại khi quan sát một người lái xe trên đường, trong đó mỗi ví dụ gồm một chuỗi các ảnh và các lệnh điều khiển xe

Quá trình học



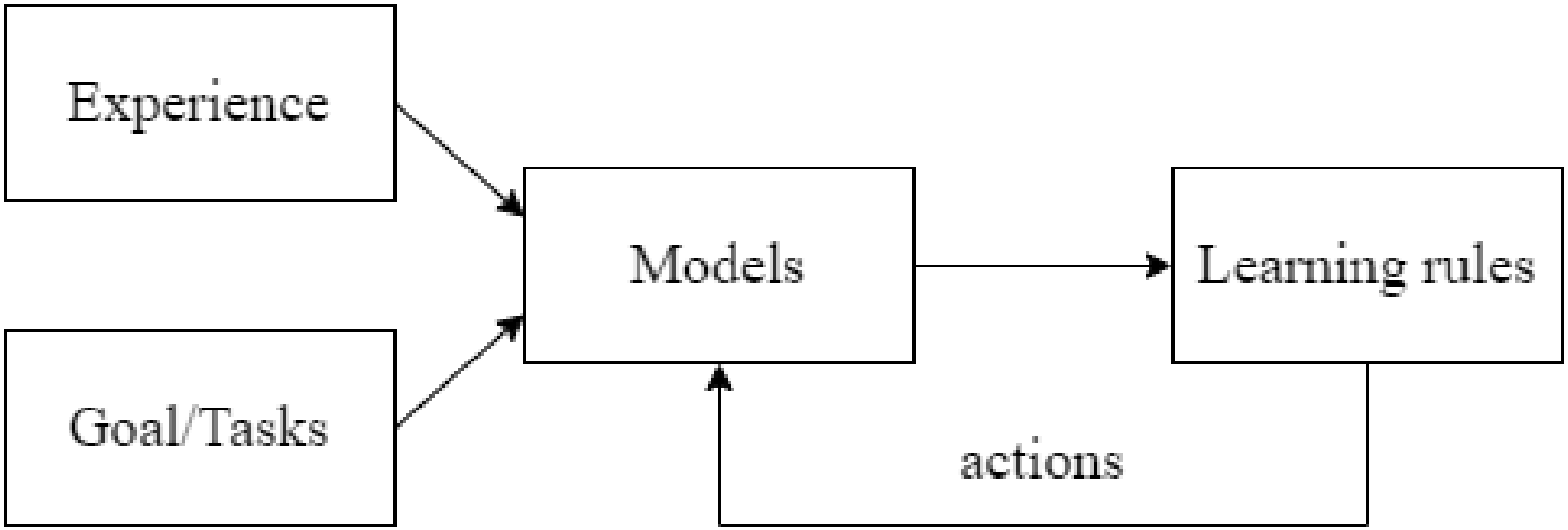
Bài toán học máy - Các thành phần chính

Một hệ thống máy học bao gồm các thành phần

- Mục đích (goal): xác định các nhiệm vụ (task) để thực hiện mục tiêu
- Mô hình (model): hàm toán học để ánh xạ kinh nghiệm - hành động hay còn được gọi là *hàm mục tiêu (giả thiết, khái niệm)*
- Luật học (Learning rules): cập nhật các tham số (parameter) của mô hình thông qua kinh nghiệm
- Kinh nghiệm (Experience): tập các dữ liệu học

Bài toán máy học - Các thành phần chính

Một hệ thống máy học bao gồm các thành phần



Thuật ngữ

- Các *nhiệm vụ* được mô tả thông qua việc xử lý một điểm dữ liệu đầu vào
- Các điểm dữ liệu (example/instance) thường được đưa về dạng tập hợp các con số, các đặc trưng/thuộc tính (feature/attribute)
- Những điểm dữ liệu được biểu diễn dưới dạng ma trận hoặc mảng nhiều chiều
  - Các điểm dữ liệu được biểu diễn dưới dạng mảng một chiều, được gọi là *vector đặc trưng (feature vector)*
  - Ký hiệu: $x \in \mathbb{R}^d$ trong đó d là số đặc trưng

Thuật ngữ

- Kinh nghiệm: bộ dự liệu được sử dụng để xây dựng mô hình. Bộ dữ liệu thường được chia:
  - tập huấn luyện và tập kiểm tra
  - tập huấn luyện, tập xác thực và tập kiểm tra
- *Tập huấn luyện (training set)*: gồm các điểm dữ liệu được sử dụng trực tiếp trong việc xây dựng mô hình
- *Tập kiểm tra (test set)*: dùng để đánh giá hiệu quả mô hình; không được sử dụng trong quá trình xây dựng mô hình
- *Tập xác thực (validation set)*: được sử dụng trong việc lựa chọn các siêu tham số mô hình

Các bài toán cơ bản trong Máy học

- Phân loại (classification)
- Hồi quy (regression)
- Phân cụm (clustering)
- Hoàn thiện dữ liệu (data completion)
- ...

Các bài toán cơ bản trong Học máy

Phân loại (classification)

- Bài toán yêu cầu xác định *lớp/nhãn (class/label)* của một điểm dữ liệu trong số C nhãn khác nhau
- Cặp *(dữ liệu, nhãn)*, (x, y) với y nhận một trong C giá trị trong tập đích $\mathcal{Y}$
- Xây dựng mô hình: tìm hàm ánh xạ f từ một điểm dữ liệu x vào một phần tử $y \in \mathcal{Y}$

$$y = f(\mathrm{x}) : \mathbb{R}^d \to \mathcal{Y}$$

Các bài toán cơ bản trong Học máy

Hồi quy (regression)

- Tập giá trị $\mathcal{Y}$ gồm các giá trị thực
- Xây dựng mô hình

$$f : \mathbb{R}^d \to \mathbb{R}$$

- Ví dụ: Ước lượng giá một căn nhà rộng x_1 m^2, có x_2 phòng ngủ và cách trung tâm thành x_3 km
- Có thể mở rộng bài toán dự đoán nhiều đầu ra cùng một lúc

$$f : \mathbb{R}^d \to \mathbb{R}^m$$

Các bài toán cơ bản trong máy học

Phân cụm (clustering)

- Chia dữ liệu $\mathcal{X}$ thành các cụm nhỏ dựa trên sự liên quan giữa các dữ liệu trong mỗi cụm
- Dữ liệu huấn luyện không có nhãn, mô hình tự phân chia dữ liệu thành các cụm khác nhau
- Ví dụ: Phân cụm khách hàng dựa trên hành vi mua hàng

Các bài toán cơ bản trong máy học

Hoàn thiện dữ liêu - data completion

- Bài toán dự đoán các trường dư liệu còn thiếu.
- Nhiệm vụ: dựa trên mối tương quan giữa các điểm dữ liệu để dự đoán những giá trị còn thiếu

Một số bài toán khác:

- Xếp hạng (ranking)
- Thu thập thông tin (information retrieval)
- Giảm chiều dữ liệu (dimentionality reduction)
- ...

Phân nhóm các thuật toán Học máy

Dựa trên tính chất dữ liệu, có thể được phân thành hai nhóm chính

- Học có giám sát (supervised learning)
- Học không giám sát (unsupervised learning)

Ngoài ra, hai nhóm thuật toán khác

- Học bán giám sát (semi-supervised learning)
- Học củng cố (reinforcement learning)

Phân nhóm các thuật toán máy học

Học có giám sát (supervised learning)

- Việc xây dựng mô hình dự đoán mối quan hệ giữa đầu vào x và đầu ra y được thực hiện dựa trên các cặp (đầu vào, đầu ra) $\{(\mathrm{x}_1, y_1), (\mathrm{x}_2, y_2), \ldots, (\mathrm{x}_N, y_N)\}$ đã biết trong tập huấn luyện
- Bài toán phân loại và hồi quy thuộc nhóm thuật toán này
- Huấn luyện: xây dựng hàm f sao cho với mọi $i = 1, 2, \ldots, N$ thì $f(\mathrm{x}_i)$ gần với y_i nhất
- Với điểm dữ liệu x ngoài tập huấn luyện, đầu ra dự đoán $f(\mathrm{x})$ gần với đầu ra thực y

Ví dụ

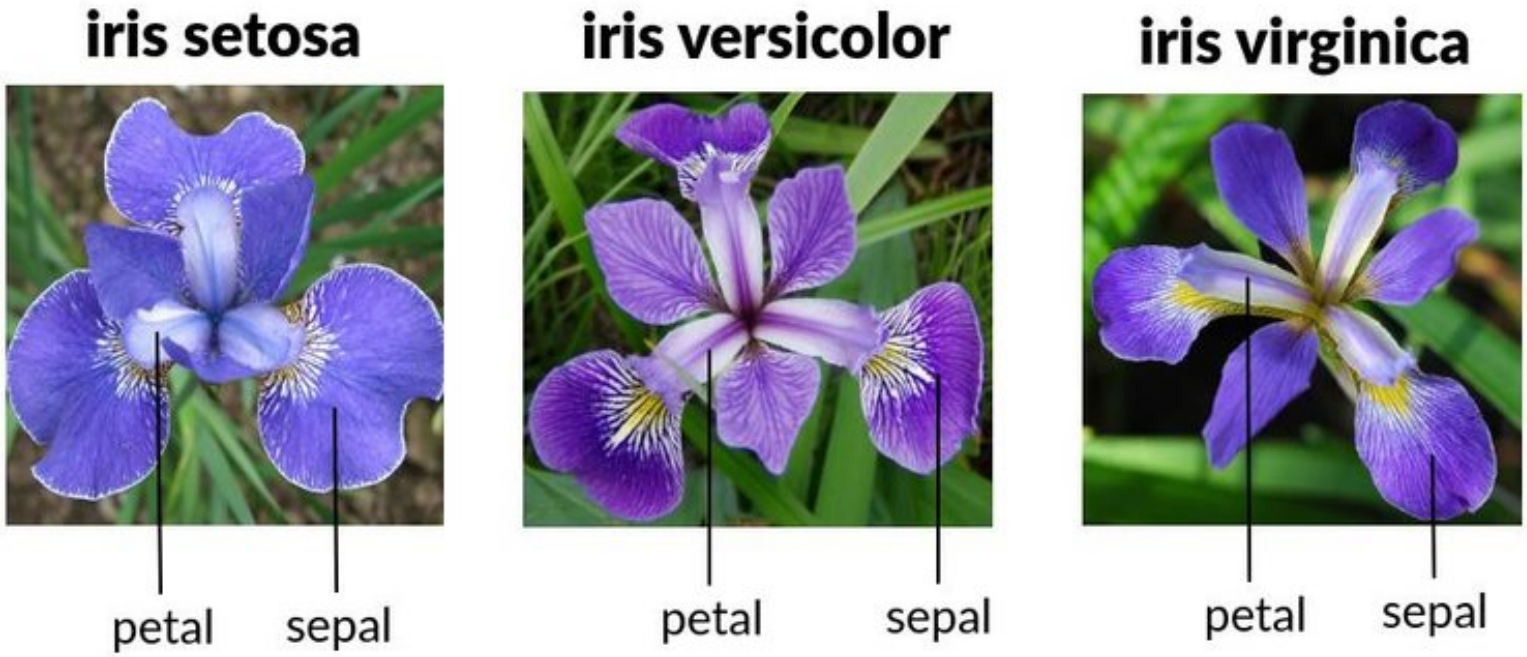
IRIS dataset

Iris Versicolor

Iris Setosa

Iris Virginica

Ví dụ


iris setosa
petal
sepal
iris versicolor
petal
sepal
iris virginica
petal
sepal

Ví dụ

sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	class
6.3	2.8	5.1	1.5	Iris-virginica
4.6	3.6	1	0.2	Iris-setosa
6	2.2	4	1	Iris-versicolor
6.4	3.2	5.3	2.3	Iris-virginica
5.6	2.8	4.9	2	Iris-virginica
5.4	3	4.5	1.5	Iris-versicolor
5.4	3.7	1.5	0.2	Iris-setosa
4.9	2.5	4.5	1.7	Iris-virginica
6.9	3.1	5.1	2.3	Iris-virginica
4.6	3.1	1.5	0.2	Iris-setosa
6.2	2.8	4.8	1.8	Iris-virginica
5	3	1.6	0.2	Iris-setosa
6.1	2.8	4	1.3	Iris-versicolor
6.3	2.5	5	1.9	Iris-virginica
6.8	3.2	5.9	2.3	Iris-virginica
5.1	3.8	1.6	0.2	Iris-setosa
6.4	2.9	4.3	1.3	Iris-versicolor

Ví dụ

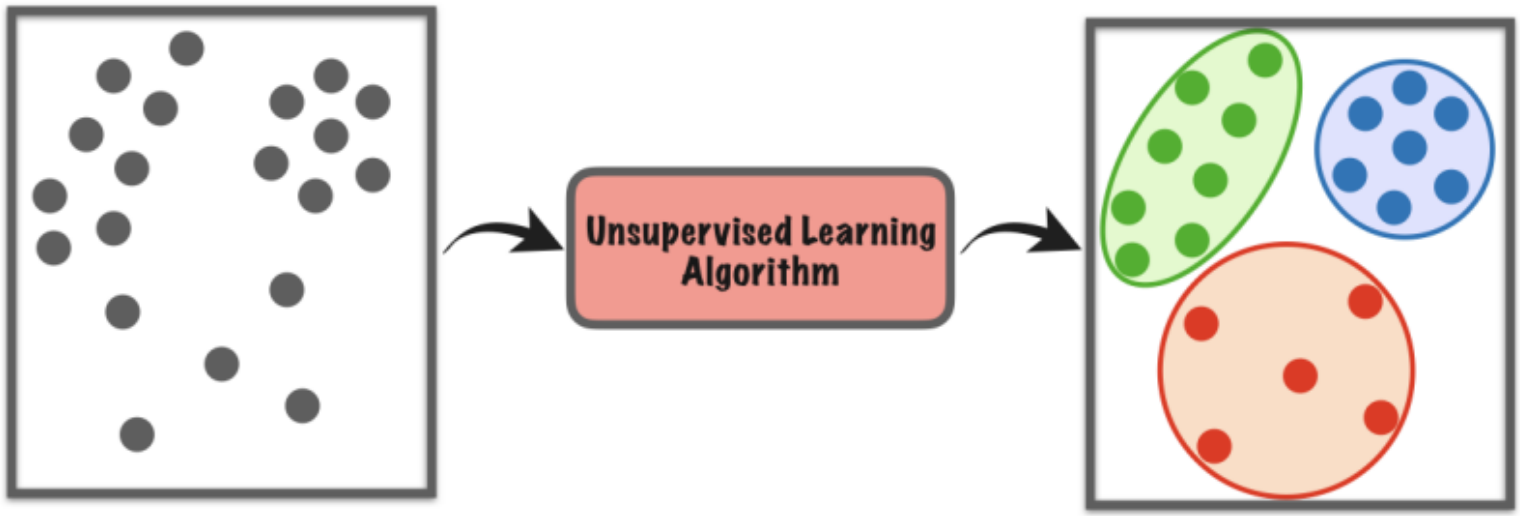
	youtube	facebook	newspaper	sales
1	276.12	45.36	83.04	26.52
2	53.40	47.16	54.12	12.48
3	20.64	55.08	83.16	11.16
4	181.80	49.56	70.20	22.20
5	216.96	12.96	70.08	15.48
6	10.44	58.68	90.00	8.64
7	69.00	39.36	28.20	14.16
8	144.24	23.52	13.92	15.84
9	10.32	2.52	1.20	5.76
10	239.76	3.12	25.44	12.72
11	79.32	6.96	29.04	10.32
12	257.64	28.80	4.80	20.88
13	28.56	42.12	79.08	11.04
14	117.00	9.12	8.64	11.64
15	244.92	39.48	55.20	22.80
16	234.48	57.24	63.48	26.88
17	81.36	43.92	136.80	15.00
18	337.68	47.52	66.96	29.28

Phân nhóm các thuật toán máy học

Học không giám sát (unsupervised learning)

- Dữ liệu huấn luyện chỉ gồm dữ liệu đầu vào x, không có đầu ra tương ứng
- Có thể không dự đoán đầu ra nhưng trích xuất những thông tin quan trọng dựa trên mối liên quan giữa các điểm dữ liệu

Ví dụ



Phân nhóm các thuật toán máy học

Học bán giám sát (semi-supervised learning)

- Dữ liệu bao gồm các cặp (đầu vào, đầu ra) và dữ liệu chỉ có đầu vào
- Ví dụ: thuật toán nhận dạng vật thể cho xe tự lái

  Một lượng lớn video thu được từ camera xe hơi; một lượng nhỏ trong video huấn luyện được nhận dạng cụ thể

Phân nhóm các thuật toán máy học

Học tăng cường (reinforcement learning)

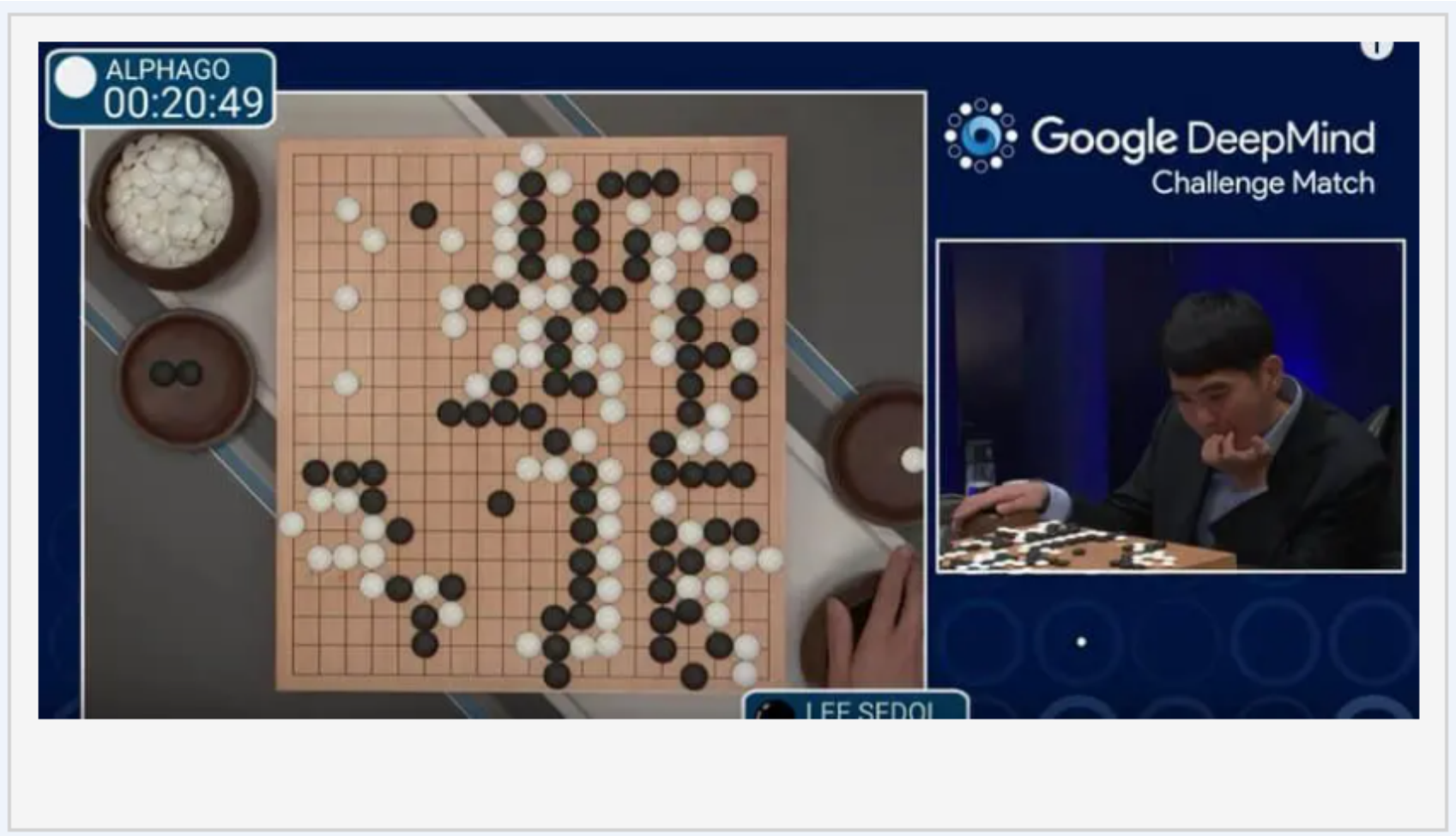
- Không yêu cầu dữ liệu huấn luyện, mô hình học cách ra quyết định bằng cách giao tiếp với môi trường xung quanh
- Ví dụ: AlphaGo Zero
- Dựa trên đại lượng gọi là *điểm thưởng (reward)*. Mô hình cần tìm ra một thuật toán tối đa điểm thưởng qua nhiều lần chơi khác nhau

Phân nhóm các thuật toán máy học

Học tăng cường (reinforcement learning)

- Không yêu cầu dữ liệu huấn luyện, mô hình học cách ra quyết định bằng cách giao tiếp với môi trường xung quanh
- Ví dụ: AlphaGo Zero
- Dựa trên đại lượng gọi là *điểm thưởng (reward)*. Mô hình cần tìm ra một thuật toán tối đa điểm thưởng qua nhiều lần chơi khác nhau

Ví dụ

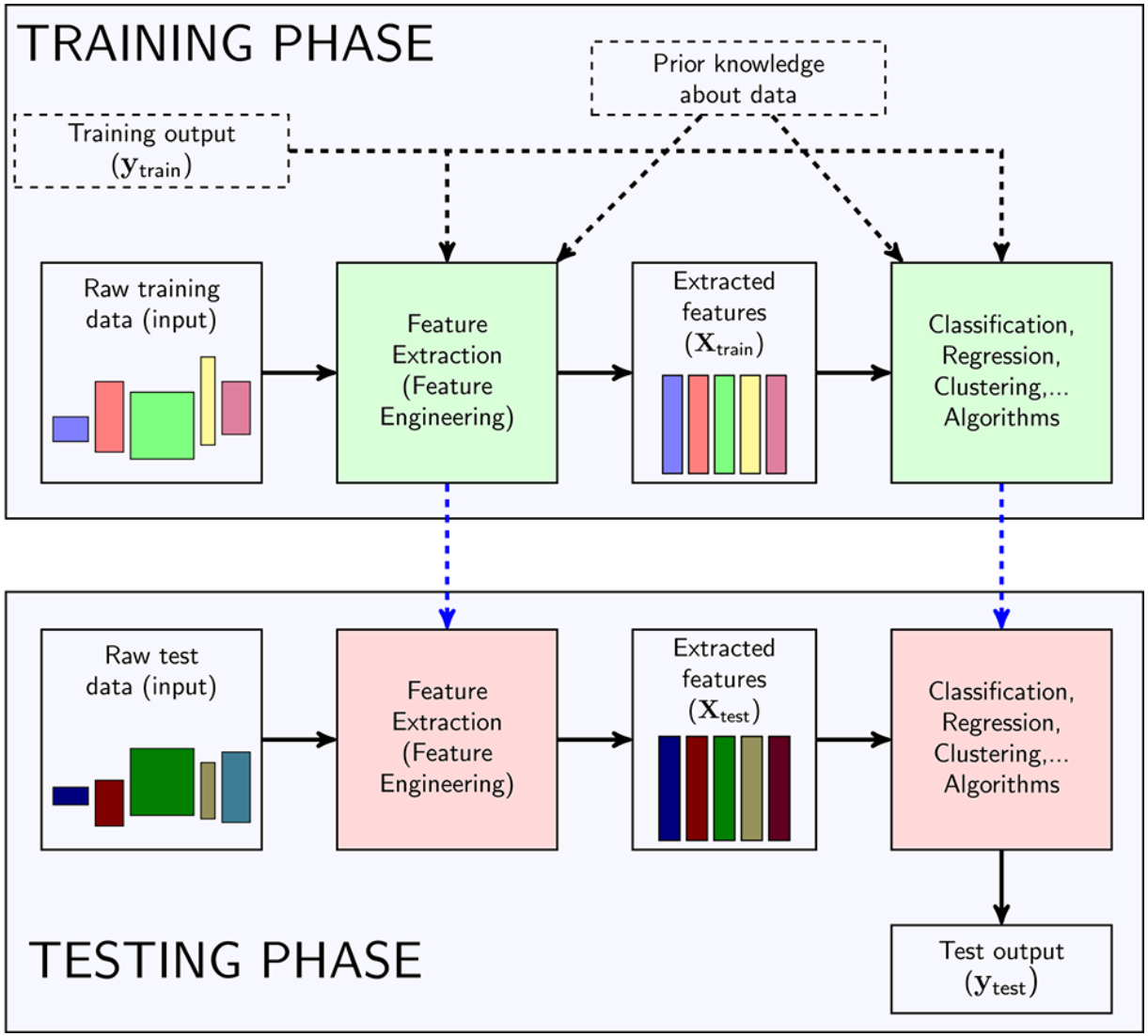


Hàm mất mát và tham số mô hình

- Mô hình được mô tả bởi bộ các *tham số mô hình (model parameter)*
- Thuật toán học máy: tìm các tham số mô hình tối ưu cho mỗi bài toán, hay tìm các tham số sao cho các phép đánh giá đạt kết quả tốt nhất
- Quan hệ giữa một phép đánh giá và tham số mô hình được mmoo tả thông qua *hàm mất mát* (loss function hay cost function)
  - $\rightarrow$ Tìm tham số mô hình sao cho phép đánh giá trả về kết quả tốt nhất hay tối thiểu hàm mất mát
  - $\rightarrow$ Giải bài toán tối ưu (quá trình *học* (learning) của máy)

$$\theta^* = \arg\min_{\theta} \mathcal{L}(\theta)$$

Mô hình chung



Giới thiệu bài toán hồi quy tuyến tính (Linear regression)

- Bài toán ví dụ:
  - Mỗi căn nhà rộng x_1 m^2, có x_2 phòng ngủ, cách trung tâm thành phố x_3 km sẽ có một giá bán nhất định
  - Giả sử có số liệu thống kê của > 1000 căn nhà
  - có một căn nhà mô tả $x = [100, 4, 12]$ thì dự đoán giá nhà đó bao nhiêu
  - Hàm dự đoán: $\hat{y} = f(x) : \mathbb{R}^n \to \mathbb{R}$
    với $x = [x_1, x_2, x_3]$
  - Giả sử: $f(x) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$
  - Bài toán đi tìm các hệ số w_0, w_1, w_2, w_3

Phân tích bài toán

- Nếu đặt $\mathbf{w} = [w_0, w_1, w_2, w_3]^T$: hệ số cần tối ưu
- Vector đầu vào: $\bar{\mathbf{x}} = [1, x_1, x_2, x_3]$
- Giá trị dự đoán tính bằng: $\hat{y} = \bar{\mathbf{x}}\,\mathbf{w} \approx y$
- Sai số giữa giá trị thực tế và dự đoán:

$$\frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - \bar{\mathbf{x}}\,\mathbf{w})^2$$

Hàm mất mát

- Giả sử có N điểm dữ liệu quan sát được. Chúng ta muốn tổng sai số là nhỏ nhất:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \bar{\mathbf{x}}_i\,\mathbf{w})^2$$

- Tương đương với tìm $\mathbf{w}$ để hàm mất mát $\mathcal{L}(\mathbf{w})$ đạt giá trị nhỏ nhất $=>$ $\mathbf{w}$ là *điểm tối ưu*

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{w})$$

Nghiệm cho bài toán

- Phổ biến là giải phương trình đạo hàm (gradient) bằng 0

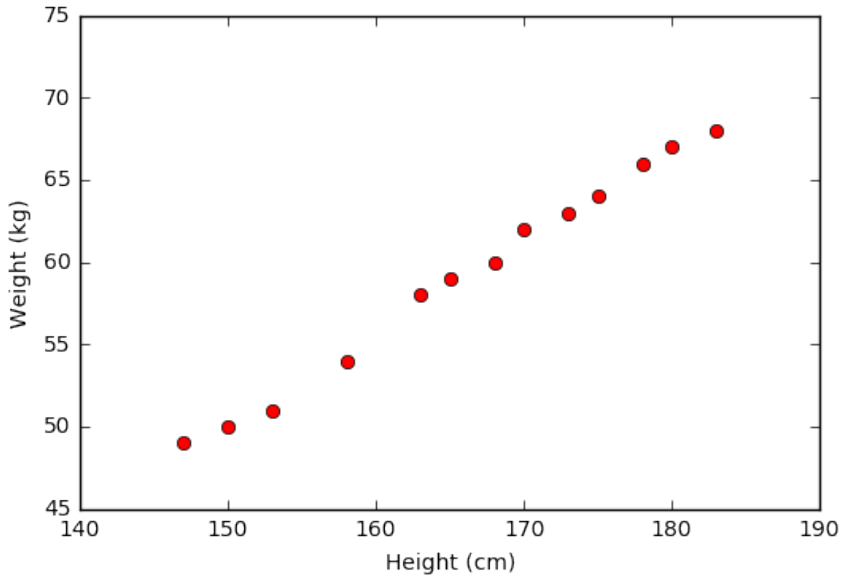
$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} = \bar{\mathbf{X}}^T(\bar{\mathbf{X}}\mathbf{w} - \mathbf{y})$$

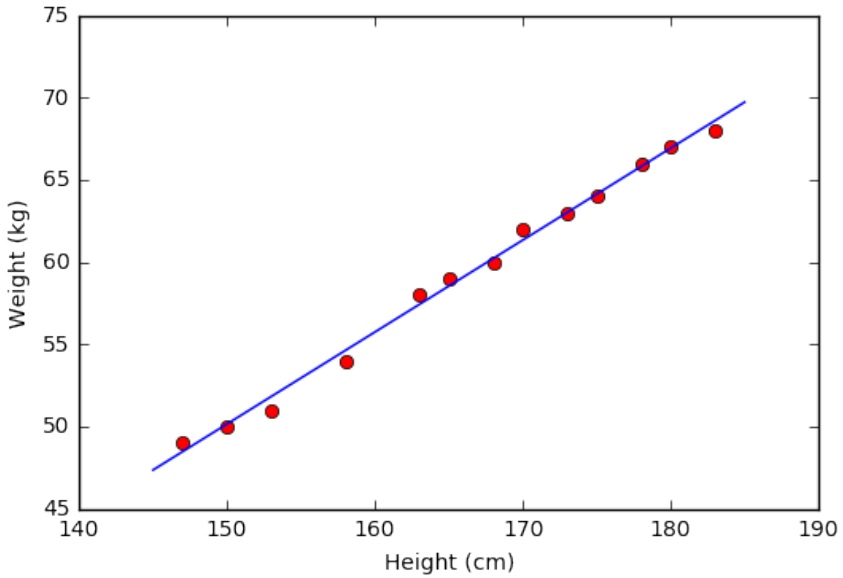
với

$$\bar{\mathbf{X}} = [\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \ldots, \bar{\mathbf{x}}_N]$$

$$\mathbf{y} = [y_1, y2, \ldots, y_N]$$

Ví dụ



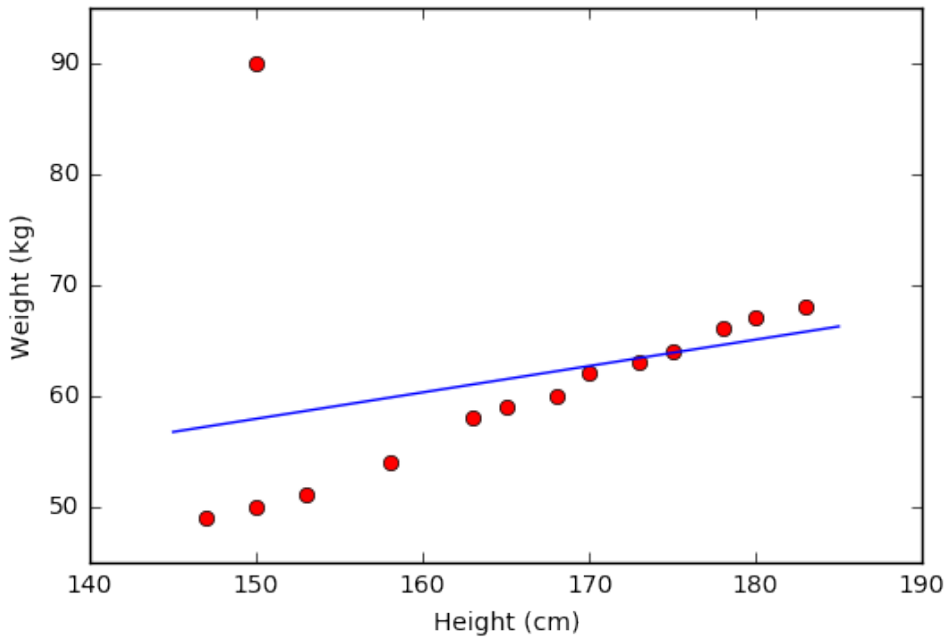


- Trên thực tế các mô hình chỉ cần tuyến tính theo **w**. Ví dụ như:

$$f(x) = w_1 x_1 + w_2 x_2^2 + w_3 x_1^3 + w_5 x_1 x_2 + w_0$$

Hạn chế

- Nhạy cảm với nhiễu (sensitive to noise)



- Không biểu diễn được mô hình phức tạp

K lân cận

K lân cận (K-nearest neighbor hay KNN):

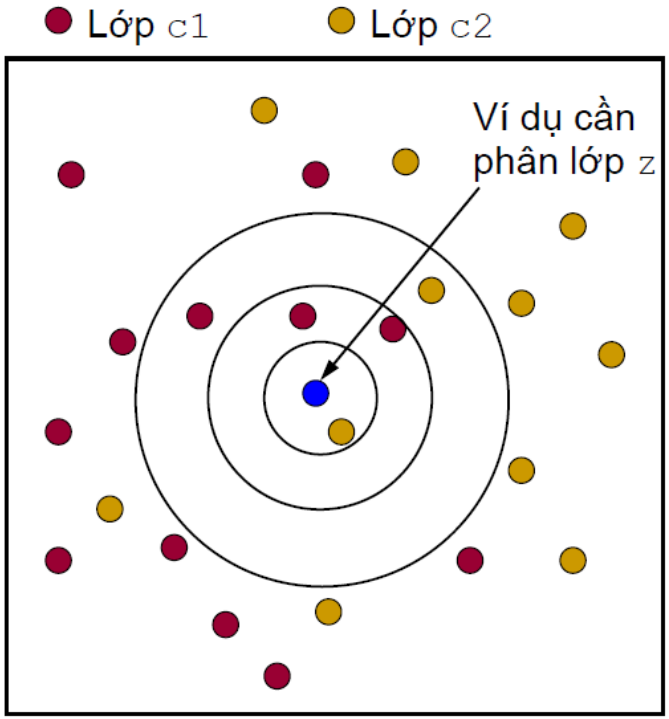
- một trong những thuật toán học có giám sát đơn giản nhất
- ghi nhớ một cách máy móc toàn bộ dữ liệu đó
- tính toán được thực hiện tại pha kiểm tra
- có thể được áp dụng vào các bài toán phân loại và hồi quy
- thuật toán *lazy learning*, *instance-based*, hoặc *memory-based learning*

K lân cận

Ý tưởng của phương pháp học dựa trên láng giềng gần nhất:

- Với một bộ dữ liệu học
  - Lưu lại các ví dụ học
  - Không cần xây dựng một mô hình (mô tả) rõ ràng và tổng quát của hàm mục tiêu cần học
- Đối với một điểm dữ liệu cần phân loại/dự đoán
  - Kiểm tra (xét) quan hệ giữa điểm đó với các điểm học để gán giá trị của hàm mục tiêu (một nhãn lớp hoặc một giá trị thực)

Ví dụ: Phân lớp dựa trên k-NN



- Nếu chỉ xét 1 láng giềng gần nhất

  $\rightarrow z$ được phân vào lớp c
- Nếu xét 3 láng giềng gần nhất

  $\rightarrow z$ được phân vào lớp c
- Nếu xét 5 láng giềng gần nhất

  $\rightarrow z$ được phân vào lớp c

Bài toán phân lớp k-NN

- Mỗi điểm dữ liệu được biểu diễn bởi 1 vector đặc trưng $x \in \mathbb{R}^m$ và nhãn $y \in \mathcal{Y}$
- Giai đoạn huấn luyện: các ví dụ học được lưu lại trong tập học $\mathcal{D}^{\text{train}} = \{x_i | i = \{1, \ldots, N\}\}$
- Giai đoạn kiểm tra/dự đoán: để phân lớp cho một ví dụ (mới) z
  - Với mỗi ví dụ học $x \in \mathcal{D}^{\text{train}}$, tính khoảng cách giữa x và z
  - Xác định tập $NB(z)$ gồm các láng giềng gần nhất của z Gồm k ví dụ học trong $\mathcal{D}^{\text{train}}$ gần nhất với z tính theo một hàm khoảng cách d
  - Phân z vào lớp chiếm số đông trong số các lớp của các ví dụ học trong $NB(z)$

Bài toán hồi quy k-NN

- Mỗi điểm dữ liệu được biểu diễn bởi 1 vector đặc trưng $x \in \mathbb{R}^m$ và nhãn $y \in \mathbb{R}$
- Giai đoạn huấn luyện: các ví dụ học được lưu lại trong tập học $\mathcal{D}^{\text{train}} = \{x_i | i = \{1, \ldots, N\}\}$
- Giai đoạn kiểm tra/dự đoán: để xác định giá trị cho một ví dụ (mới) z
  - Với mỗi ví dụ học $x \in \mathcal{D}^{\text{train}}$, tính khoảng cách giữa x và z
  - Xác định tập $NB(z)$ gồm các láng giềng gần nhất của z Gồm k ví dụ học trong $\mathcal{D}^{\text{train}}$ gần nhất với z tính theo một hàm khoảng cách d
  - Giá trị đầu ra dự đoán đối với z:

$$\hat{y} = \frac{1}{k} \sum_{x \in NB(z)} y$$

Số láng giềng

- Dựa trên một láng giềng gần nhất thường *không chính xác*
  - Các ví dụ bất thường, không điển hình
  - Do lỗi trong quá trình xây dựng dữ liệu
- Thường xét $k > 1$ các điểm dữ liệu gần nhất với điểm cần dự đoán
- k thường được chọn là một số lẻ

Hàm tính khoảng cách

- Hàm tính khoảng cách d
  - Đóng vai trò rất quan trọng trong phương pháp học dựa trên láng giềng gần nhất
  - Thường được xác định trước và không thay đổi trong suốt quá trình học và dự đoán
- Hàm khoảng cách d
  - *Các hàm khoảng cách hình học*: thuộc tính đầu vào là kiểu số thực $x \in \mathbb{R}^m$
  - *Hàm khoảng cách Hamming*: thuộc tính đầu vào là kiểu nhị phân $x \in \{0, 1\}^m$
  - *Hàm tính độ tương tự Cosine*: thường sử dụng cho các bài toán phân lớp văn bản trong đó x gồm các giá trị trọng số *TF/IDF* của các từ khóa

Hàm tính khoảng cách

Hàm tính khoảng cách hình học (Geometry distance functions)

- Hàm Manhattan:

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

- Hàm Euclid:

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

- Hàm Minkowski (p-norm)

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

Hàm tính khoảng cách

- Hàm Hamming:
  - Đầu vào nhị phân, ví dụ $x = \{1, 0, 1, 0\}$

$$d = \sum_{i}^{m} \text{diff}(x_i, z_i) \qquad \text{diff}(x_i, z_i) = \begin{cases} 1, & \text{nếu } x_i \neq z_i \\ 0, & \text{nếu } x_i = z_i \end{cases}$$

- Hàm tính độ tương tự Cosine

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

Ví dụ

```
knn1 = KNeighborsClassifier(p=1)
knn1.fit(X_train,y_train)
preds1 = knn1.predict(X_test)

knn2 = KNeighborsClassifier(p=2)
knn2.fit(X_train,y_train)
preds2 = knn2.predict(X_test)

knn3 = KNeighborsClassifier(p=3)
knn3.fit(X_train,y_train)
preds3 = knn3.predict(X_test)

print("Do chinh xac: {}, {}, {}".format(accuracy_score(y_test, preds1),
                                        accuracy_score(y_test, preds2),
                                        accuracy_score(y_test, preds3)) )
```

Kết quả

```
Accuracy: 0.9749652294853964, 0.9847009735744089, 0.9874826147426982
```

Chuẩn hóa miền giá trị thuộc tính

Lấy hàm Euclid như hàm ví dụ:

- Giả sử mỗi điểm dữ liệu được biểu diễn bằng 03 đặc trưng: *Age*, *Income* (mỗi tháng) và *Height* (đo theo mét)
  - $x = (Age = 20, Income = 12000, Height = 1.68)$
  - $y = (Age = 40, Imcome = 1300, Height = 1.75)$
- Khoảng cách giữa x và z:
  - $d(x, z) = \sqrt{(20 - 40)^2 + (12000 - 1300)^2 + (1.68 - 1.75)^2}$
  - Giá trị phụ thuộc chủ yếu bởi sự khác biệt của *Income*
- Cần chuẩn hóa miền giá trị (đưa về cùng một khoảng giá trị)

Trọng số của các thuộc tính

Lấy hàm Euclid như hàm ví dụ:

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

- Các thuộc tính có cùng ảnh hưởng đối với giá trị
- Các thuộc tính khác nhau có thể có mức độ ảnh hưởng khác nhau đối với giá trị khoảng cách
  $\rightarrow$ Phải tích hợp các giá trị trọng số của các thuộc tính trong hàm tính khoảng cách

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

với w_i là trọng số của đặc trưng i

Trọng số của các thuộc tính

Xác định các giá trị trọng số như thế nào?

- Dựa trên các tri thức cụ thể của bài toán
- Bằng một quá trình tối ưu hóa các giá trị trọng số

Khoảng cách của các láng giềng

- Xét tập $NB(z)$ gồm k điểm dữ liệu gần nhất với điểm cần dự đoán z
  - Mỗi điểm có khoảng cách khác nhau đến z
  - Các láng giềng có ảnh hưởng như nhau đối với việc 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 cho các láng giềng gần hơn

Khoảng cách của các láng giềng

```
model1 = KNeighborsClassifier(weights='uniform')
model1.fit(X_train,y_train)
preds1 = model1.predict(X_test)
model2 = KNeighborsClassifier(weights='distance')
model2.fit(X_train,y_train)
preds2 = model2.predict(X_test)
print("Accuracy: {}, {}".format(accuracy_score(y_test, preds1),
                                accuracy_score(y_test, preds2)))
```

Kết quả

```
Accuracy: 0.9791376912378303, 0.9819193324061196
```

Ưu, nhược điểm của k-NN

Ưu điểm:

- Không cần bước học (hệ thống chỉ đơn giản là lưu lại các ví dụ học)
- Hoạt động tốt đối với các bài toán có số lớp khá lớn
- Phương pháp học k-NN ($k \gg 1$) có thể làm việc được cả với dữ liệu lỗi
- Việc dự đoán kết quả của dữ liệu mới rất đơn giản

Nhược điểm:

- Phải xác định hàm tính khoảng cách phù hợp
- Chi phí tính toán (về thời gian và bộ nhớ) tại thời điểm đánh giá/dự đoán
- Có thể dự đoán sai, do các thuộc tính không liên quan (irrelevant attributes)
- Rất nhạy cảm với nhiễu khi k nhỏ