

COLE.VN
connecting knowledge

Chủ đề:
Học giám sát (*Supervised Learning*)

Mục đích buổi học

- Học viên được tiếp cận các khái niệm, kỹ thuật và mô hình học máy cơ bản và phổ dụng với bài toán học giám sát (Supervised Learning)
 - Bài toán hồi quy (Regression)
 - Bài toán phân loại (Classification)
- Học viên được tiếp cận với các ứng dụng bài toán học giám sát với nhiều ví dụ minh họa, dữ liệu từ các bài toán thực tế.
 - Dự đoán/ dự báo giá cả
 - Phân loại thông tin

Nội dung chính

Học giám sát (Supervised Learning)

Classification

Regression

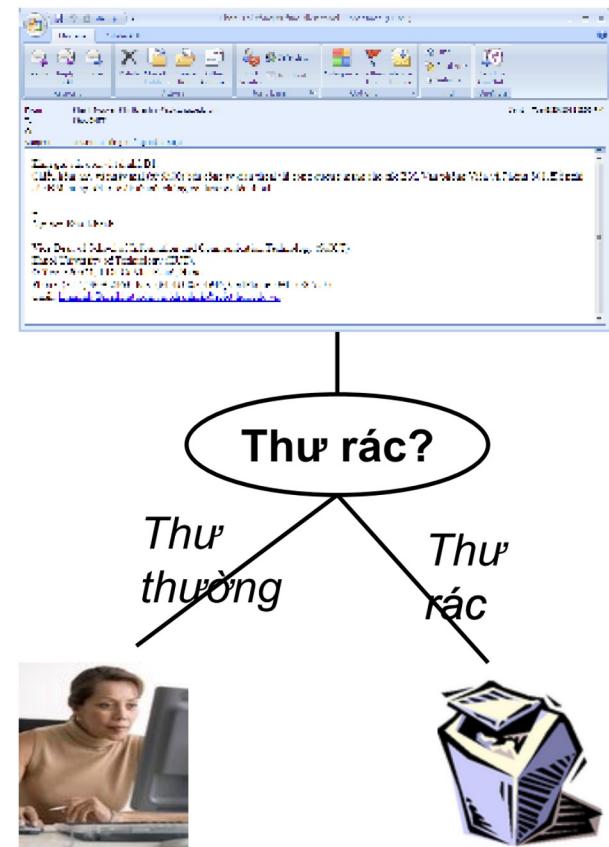
Tổng quan về học máy (Machine Learning)

- Một thuật toán machine learning là một thuật toán có khả năng **học** từ dữ liệu.
- Định nghĩa **học** (chương trình máy tính)
 - “*Một chương trình máy tính được gọi là học từ kinh nghiệm E để hoàn thành nhiệm vụ T, với hiệu quả được đo bằng phép đánh giá P, nếu hiệu quả của nó khi thực hiện nhiệm vụ T, khi được đánh giá bởi P, cải thiện theo kinh nghiệm E.*” - Tom Mitchell 1997
- Như vậy một bài toán học máy có thể biểu diễn bằng 1 bộ (T, P, E)
 - **T**: một công việc (nhiệm vụ)
 - **P**: tiêu chí đánh giá hiệu năng
 - **E**: kinh nghiệm

Tổng quan về học máy (Machine Learning) - Ví dụ

- **Lọc thư rác (email spam filtering)**

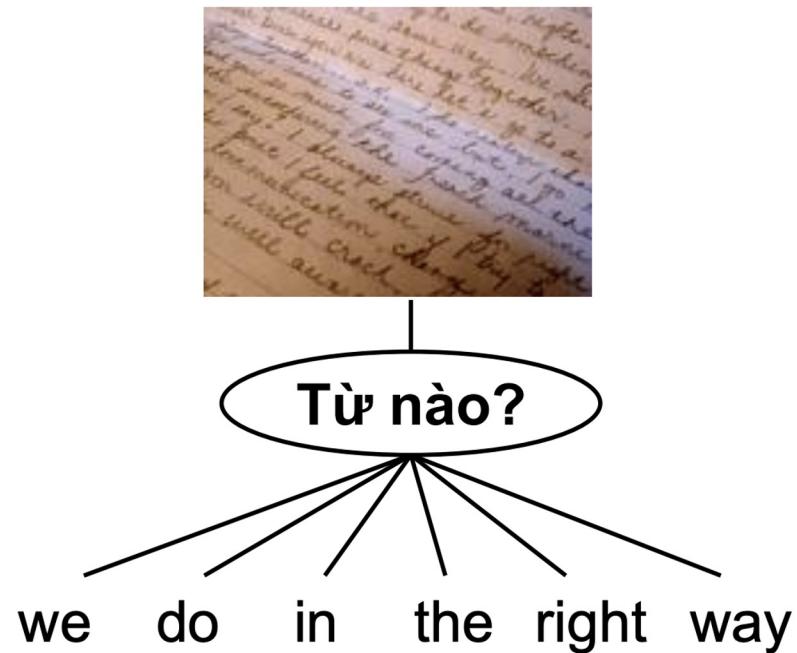
- T : Dự đoán (để lọc) những thư điện tử nào là thư rác (spam email)
- P : số lượng thư điện tử gửi đến được phân loại chính xác
- E : Một tập các thư điện tử (emails) mẫu, mỗi thư điện tử được biểu diễn bằng một tập thuộc tính (vd: tập từ khóa) và nhãn lớp (thư thường/thư rác) tương ứng



Tổng quan về học máy (Machine Learning) - Ví dụ

- **Nhận dạng chữ viết tay**

- **T**: Nhận dạng và phân loại các từ trong các ảnh chữ viết
- **P**: Tỷ lệ (%) các từ được nhận dạng và phân loại đúng
- **E**: Một tập các ảnh chữ viết, trong đó mỗi ảnh được gắn với một định danh của một từ



Tổng quan về học máy (Machine Learning) - Ví dụ

- **Gán nhãn ảnh**

- **T**: đưa ra một vài mô tả ý nghĩa của 1 bức ảnh
- **P**: ?
- **E**: Một tập các bức ảnh, trong đó mỗi ảnh đã được gán một tập các từ mô tả ý nghĩa của chúng



FISH WATER OCEAN
TREE CORAL



PEOPLE MARKET PATTERN
TEXTILE DISPLAY



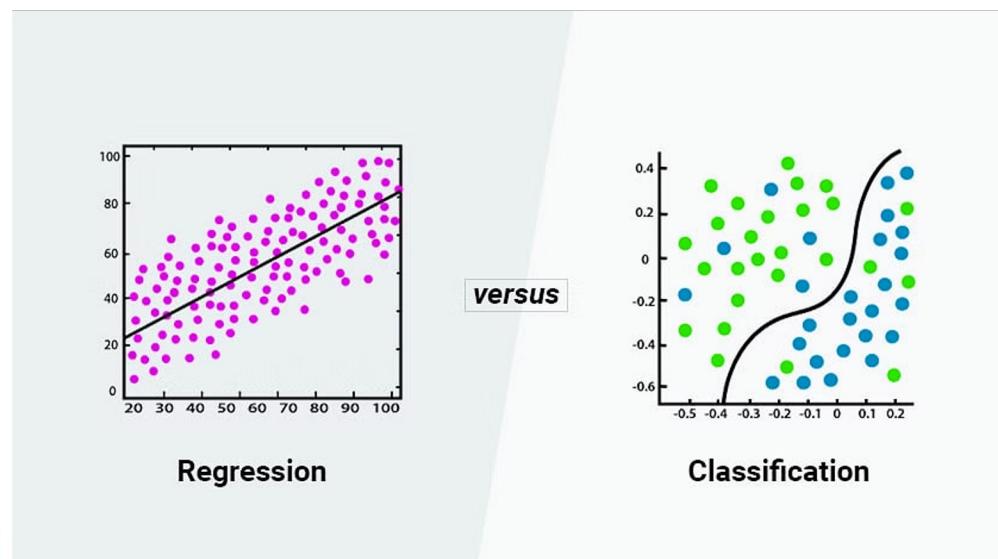
BIRDS NEST TREE
BRANCH LEAVES

Phân loại các bài toán học máy

- **Học giám sát (Supervised Learning):**
 - Dữ liệu đầu vào đã được gán nhãn cho các nhiệm vụ cụ thể
 - Dự đoán đầu ra của một hoặc nhiều dữ liệu mới dựa trên các cặp (đầu vào, đầu ra) đã biết từ trước.
- **Học không giám sát (Unsupervised Learning):**
 - Dữ liệu đầu vào không được gán nhãn
 - Tìm kiếm thông tin, các đặc trưng của bộ dữ liệu
- **Học tăng cường (Reinforcement Learning):**
 - Nghiên cứu cách thức một *agent* trong một môi trường nên chọn thực hiện các hành động nào để cực đại hóa một khoản thưởng (*reward*) nào đó về lâu dài.
 - Không có cặp dữ liệu vào/kết quả đúng, các hành động gần tối ưu cũng không được đánh giá đúng sai tường minh

Khái niệm chung học giám sát (Supervised Learning)

- Học có giám sát là một kĩ thuật của ngành học máy để xây dựng một hàm từ dữ liệu huấn luyện. Dữ liệu huấn luyện bao gồm các cặp gồm đối tượng đầu vào, và đầu ra mong muốn.
 - Đầu ra của hàm dự đoán là một giá trị liên tục
=> **Bài toán hồi quy**
 - Dự đoán một nhãn phân loại cho một đối tượng đầu vào
=> **Bài toán phân loại**



Khái niệm chung học giám sát (Supervised Learning)

Ví dụ cho bài toán phân loại

- Phân loại thư rác:
 - Xem thư nhận được là thư rác hay không
 - Nhãn: thư rác hoặc thư thông thường

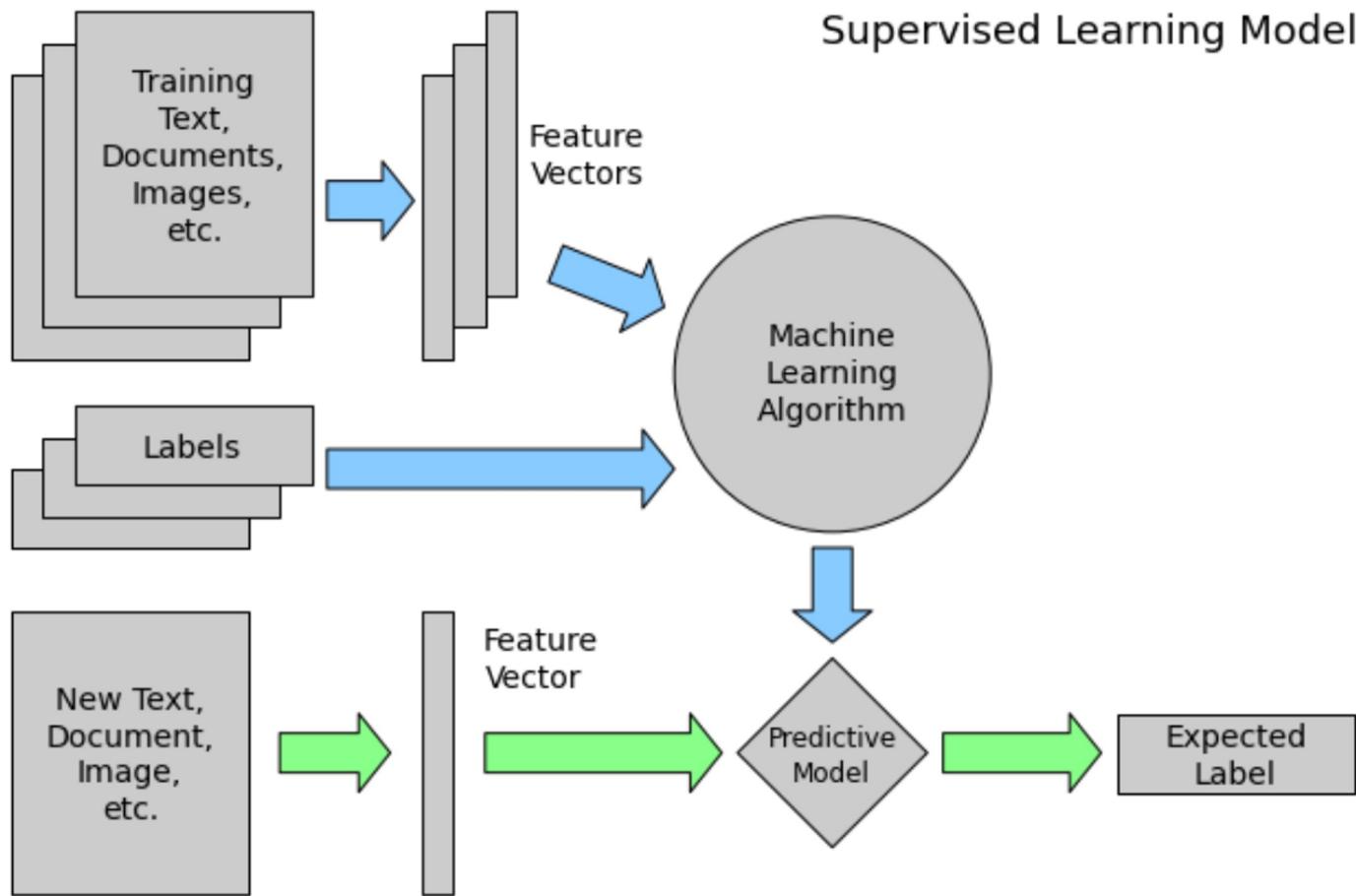
Ví dụ hồi quy

- Dự đoán giá xe:
 - Đoán giá xe dựa trên tập hợp các đặc trưng: số cây số đã đi, tuổi xe, hãng xe, đã gặp tai nạn chưa...
 - Nhãn: thông tin đặc trưng của các mẫu xe khác nhau cùng giá của chúng

Có thể áp dụng thuật toán hồi quy cho phân loại

- Hồi quy logistic: giá trị tương ứng là xác suất mẫu thuộc 1 lớp nhất định

Quy trình học giám sát

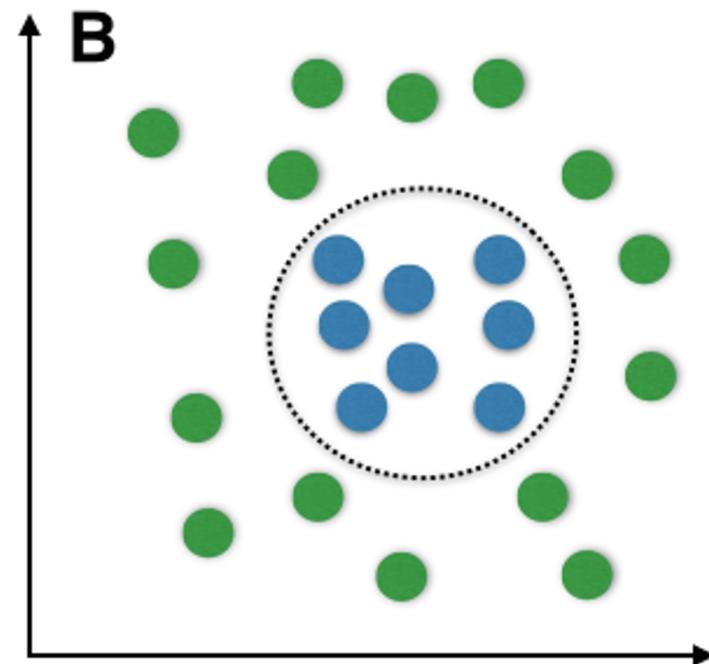
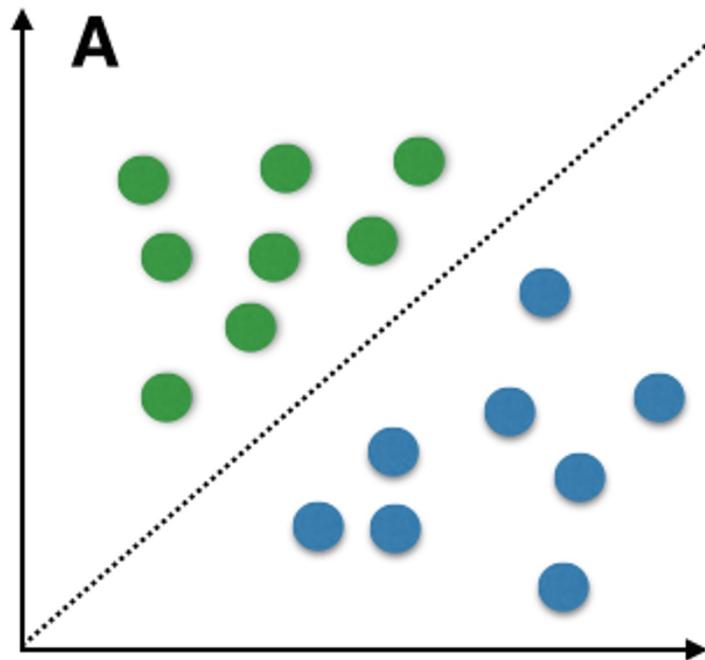


Classification

Supervised Classification

Ý tưởng:

Đi tìm ranh giới phân cách (decision boundary) giữa các lớp dữ liệu

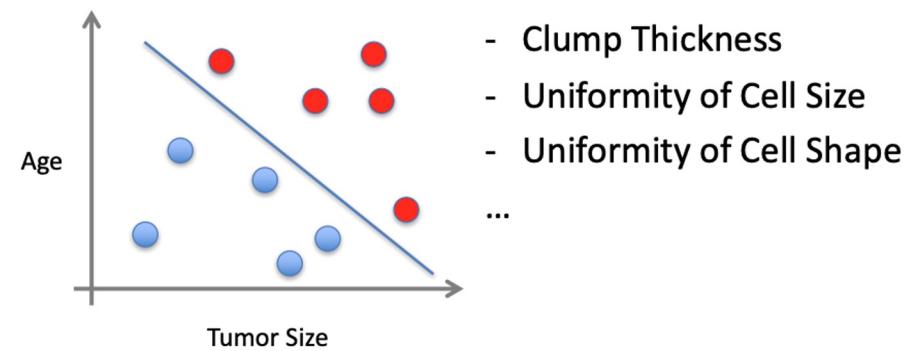
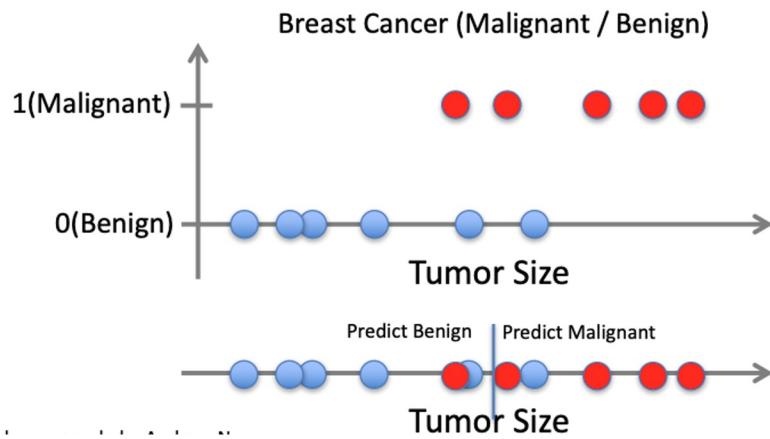


Supervised Classification

Bài toán:

Cho các cặp mẫu quan sát và đầu ra tương ứng (x_1, y_1),
(x_2, y_2), ..., (x_n, y_n). Cần tìm hàm xấp xỉ
 $f(x)$ để dự đoán giá trị đầu ra y khi cho dữ liệu đầu vào x

- Đầu ra - y : dạng nhãn (category)
- Đầu vào - x : Có thể là dữ liệu nhiều chiều (multi-dimensions)
 - Mỗi chiều tương ứng với một thuộc tính (attribute)



Supervised Classification - Ví dụ

Phân lớp hoa hoa Diên vĩ (Iris)

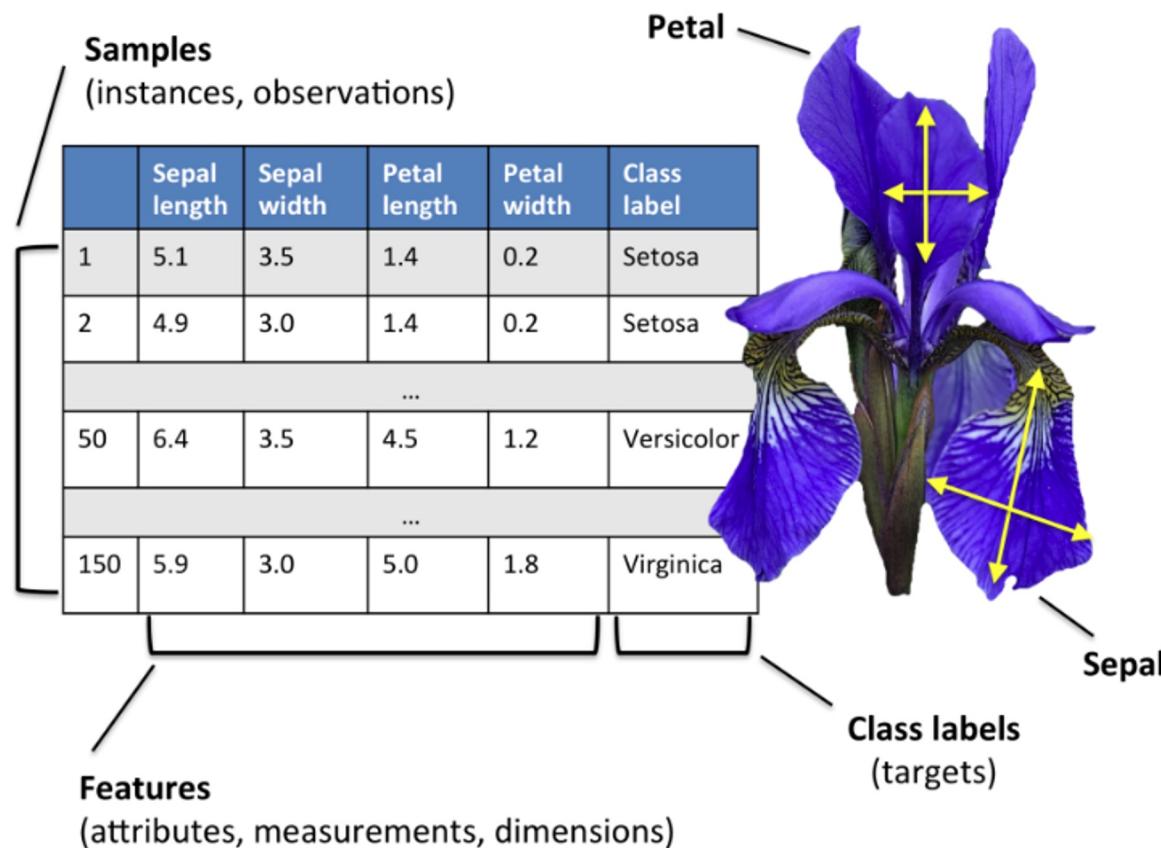
- 3 loài hoa, mỗi loài gồm 50 cá thể:
 - Iris setosa; Iris virginica, Iris versicolor
- 4 đặc trưng:
 - Độ dài của lá
 - Độ rộng của lá
 - Độ dài của cánh hoa
 - Độ rộng của cánh hoa



Xây dựng chương trình tự động phân lớp hoa

- Cho một bông hoa được biểu diễn bởi 4 đặc trưng
- Đưa bông hoa đó vào 1 trong 3 lớp

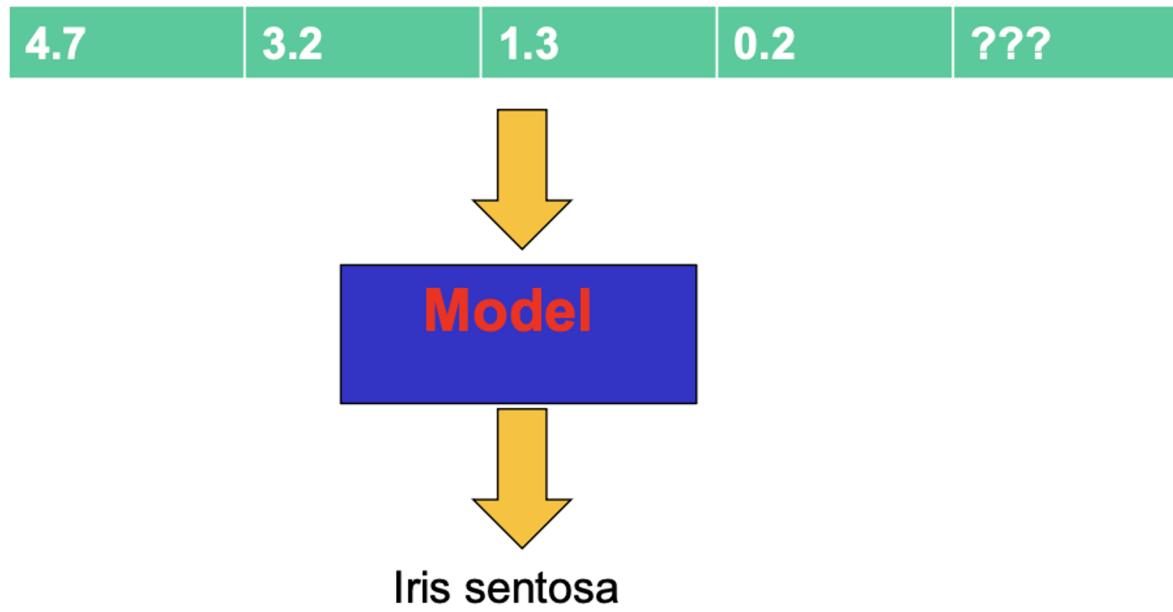
Supervised Classification - Ví dụ



Tập dữ liệu IRIS

Supervised Classification - Ví dụ

Kiểm tra:



Dựa vào các thông tin thuộc tính của loài hoa, mô hình được huấn luyện để đưa ra kết quả phân loại.

Các thuật toán phân loại (classification algorithms)

- **Naïve Bayes classifier**
- **k-Nearest Neighbors**
- **Decision Tree**
- **Random Forest**
- **Support vector machine**
- ...

Naïve Bayes classifier

- Là các phương pháp học phân lớp có giám sát và dựa trên xác suất
- Dựa trên một mô hình (hàm) xác suất
- Việc phân loại dựa trên các giá trị xác suất của các khả năng xảy ra của các giả thiết
- Là một trong các phương pháp học máy thường được sử dụng trong các bài toán thực tế
- Dựa trên định lý Bayes (Bayes theorem)

Định lý Bayes

$$P(h | D) = \frac{P(D | h).P(h)}{P(D)}$$

- $P(h)$: Xác suất trước (tiên nghiệm) của giả thiết h
 - $P(D)$: Xác suất trước (tiên nghiệm) của việc quan sát được dữ liệu D
 - $P(D|h)$: Xác suất (có điều kiện) của việc quan sát được dữ liệu D , nếu biết giả thiết h là đúng. (likelihood)
 - $P(h|D)$: Xác suất (hậu nghiệm) của giả thiết h là đúng, nếu quan sát được dữ liệu D
- Nhiều phương pháp phân loại dựa trên xác suất sẽ sử dụng xác suất hậu nghiệm (posterior probability) này!

Định lý Bayes - Ví dụ

Giả sử chúng ta có tập dữ liệu sau (dự đoán 1 người có chơi tennis)?

Ngày	Ngoài trời	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
N1	Nắng	Nóng	Cao	Yếu	Không
N2	Nắng	Nóng	Cao	Mạnh	Không
N3	Âm u	Nóng	Cao	Yếu	Có
N4	Mưa	Bình thường	Cao	Yếu	Có
N5	Mưa	Mát mẻ	Bình thường	Yếu	Có
N6	Mưa	Mát mẻ	Bình thường	Mạnh	Không
N7	Âm u	Mát mẻ	Bình thường	Mạnh	Có
N8	Nắng	Bình thường	Cao	Yếu	Không
N9	Nắng	Mát mẻ	Bình thường	Yếu	Có
N10	Mưa	Bình thường	Bình thường	Yếu	Có
N11	Nắng	Bình thường	Bình thường	Mạnh	Có
N12	Âm u	Bình thường	Cao	Mạnh	Có

Định lý Bayes - Ví dụ

- Dữ liệu **D**. *Ngoài trời là nắng và Gió là mạnh*
- Giả thiết (phân loại) **h**. Anh ta chơi tennis
- Xác suất trước **P(h)**. Xác suất rằng anh ta chơi tennis (bất kể *Ngoài trời* như thế nào và *Gió* ra sao)
- Xác suất trước **P(D)**. Xác suất rằng *Ngoài trời là nắng và Gió là mạnh*
- **P(D|h)**. Xác suất *Ngoài trời là nắng và Gió là mạnh*, nếu biết rằng anh ta chơi tennis
- **P(h|D)**. Xác suất anh ta chơi tennis, nếu biết rằng *Ngoài trời là nắng và Gió là mạnh*

Naïve Bayes classifier

- Biểu diễn bài toán phân loại (classification problem)
 - Một tập học **D_train**, trong đó mỗi ví dụ học x được biểu diễn là một vectơ n chiều: (x_1, x_2, \dots, x_n)
 - Một tập xác định các nhãn lớp: $C=\{c_1, c_2, \dots, c_m\}$
 - Với một ví dụ (mới) z, thì z sẽ được phân vào lớp nào?
- Mục tiêu: Xác định phân lớp có thể (phù hợp) nhất đối với z

$$c_{MAP} = \arg \max_{c_i \in C} P(c_i | z)$$

$$c_{MAP} = \arg \max_{c_i \in C} P(c_i | z_1, z_2, \dots, z_n)$$

$$c_{MAP} = \arg \max_{c_i \in C} \frac{P(z_1, z_2, \dots, z_n | c_i).P(c_i)}{P(z_1, z_2, \dots, z_n)} \quad (\text{bởi định lý Bayes})$$

Naïve Bayes classifier

- Để tìm được phân lớp có thể nhất đối với \mathbf{z} ...

$$c_{MAP} = \arg \max_{c_i \in C} P(z_1, z_2, \dots, z_n | c_i) \cdot P(c_i) \quad (P(z_1, z_2, \dots, z_n) \text{ là} \\ \text{như nhau với các lớp})$$

- Giả thuyết (assumption) trong phương pháp phân loại Naïve Bayes: Các thuộc tính là độc lập có điều kiện (conditionally independent) đối với các lớp

$$P(z_1, z_2, \dots, z_n | c_i) = \prod_{j=1}^n P(z_j | c_i)$$

- Phân loại Naïve Bayes tìm phân lớp có thể nhất đối với \mathbf{z}

$$c_{NB} = \arg \max_{c_i \in C} P(c_i) \cdot \prod_{j=1}^n P(z_j | c_i)$$

Naïve Bayes classifier

- Giai đoạn học (training phase), sử dụng một tập học
 - Đối với mỗi phân lớp có thể (mỗi nhãn lớp) $c_i \in C$
 - Tính giá trị xác suất tiên nghiệm: $P(c_i)$
 - Đối với mỗi giá trị thuộc tính x_j , tính giá trị xác suất xảy ra của giá trị thuộc tính đó đối với một phân lớp c_z : $P(x_j | c_i)$
- Giai đoạn phân lớp (classification phase), đối với một ví dụ mới
 - Đối với mỗi phân lớp $c_i \in C$, tính giá trị của biểu thức:

$$P(c_i) \cdot \prod_{j=1}^n P(x_j | c_i)$$

- Xác định phân lớp của z là lớp có thể nhất c^*

$$c^* = \arg \max_{c_i \in C} P(c_i) \cdot \prod_{j=1}^n P(x_j | c_i)$$

Naïve Bayes classifier - Ví dụ

Một sinh viên trẻ với thu nhập trung bình và mức đánh giá tín dụng bình thường sẽ mua một cái máy tính?

Rec. ID	Age	Income	Student	Credit_Rating	Buy_Computer
1	Young	High	No	Fair	No
2	Young	High	No	Excellent	No
3	Medium	High	No	Fair	Yes
4	Old	Medium	No	Fair	Yes
5	Old	Low	Yes	Fair	Yes
6	Old	Low	Yes	Excellent	No
7	Medium	Low	Yes	Excellent	Yes
8	Young	Medium	No	Fair	No
9	Young	Low	Yes	Fair	Yes
10	Old	Medium	Yes	Fair	Yes
11	Young	Medium	Yes	Excellent	Yes
12	Medium	Medium	No	Excellent	Yes
13	Medium	High	Yes	Fair	Yes
14	Old	Medium	No	Excellent	No

Naïve Bayes classifier - Ví dụ

Biểu diễn bài toán phân loại

- $z = (\text{Age}=\text{Young}, \text{Income}=\text{Medium}, \text{Student}=\text{True}, \text{Credit_Rating}=\text{Fair})$
- Có 2 phân lớp có thể:
 - c_1 ("Mua máy tính")
 - c_2 ("Không mua máy tính")
- Tính giá trị xác suất trước cho mỗi phân lớp
 - $P(c_1) = 9/14$
 - $P(c_2) = 5/14$
- Tính giá trị xác suất của mỗi giá trị thuộc tính đối với mỗi phân lớp
 - $P(\text{Age}=\text{Young}|c_1) = 2/9$
 - $P(\text{Age}=\text{Young}|c_2) = 3/5$
 - $P(\text{Income}=\text{Medium}|c_1) = 4/9$
 - $P(\text{Income}=\text{Medium}|c_2) = 2/5$
 - $P(\text{Student}=\text{Yes}|c_1) = 6/9$
 - $P(\text{Student}=\text{Yes}|c_2) = 1/5$
 - $P(\text{Credit_Rating}=\text{Fair}|c_1) = 6/9$
 - $P(\text{Credit_Rating}=\text{Fair}|c_2) = 2/5$

Naïve Bayes classifier - Ví dụ

Tính toán xác suất có thể xảy ra (likelihood) của ví dụ z đối với mỗi phân lớp

- Đối với phân lớp c_1

$$P(z|c_1) = P(\text{Age}=\text{Young}|c_1).P(\text{Income}=\text{Medium}|c_1).P(\text{Student}=\text{Yes}|c_1).$$

$$P(\text{Credit_Rating}=\text{Fair}|c_1) = (2/9).(4/9).(6/9).(6/9) = 0.044$$

- Đối với phân lớp c_2

$$P(z|c_2) = P(\text{Age}=\text{Young}|c_2).P(\text{Income}=\text{Medium}|c_2).P(\text{Student}=\text{Yes}|c_2).$$

$$P(\text{Credit_Rating}=\text{Fair}|c_2) = (3/5).(2/5).(1/5).(2/5) = 0.019$$

Xác định phân lớp có thể nhất (the most probable class)

- Đối với phân lớp c_1

$$P(c_1).P(z|c_1) = (9/14).(0.044) = 0.028$$

- Đối với phân lớp c_2

$$P(c_2).P(z|c_2) = (5/14).(0.019) = 0.007$$

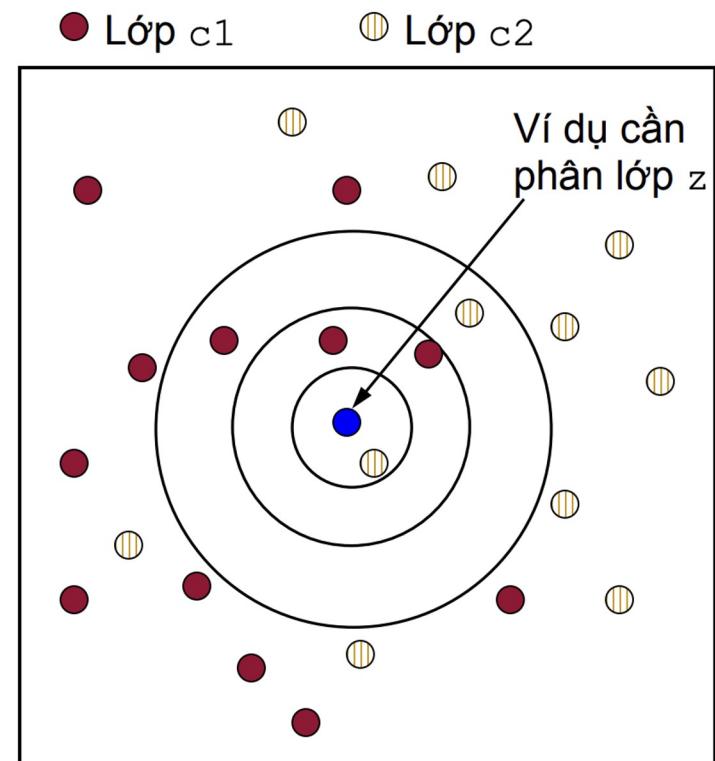
→ Kết luận: Anh ta (z) sẽ mua một máy tính!

k-Nearest Neighbors

- 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)
- Hai thành phần chính:
 - Độ đo tương đồng (similarity measure/distance) giữa các đối tượng.
 - Các hàng xóm sẽ dùng vào việc phán đoán

k-Nearest Neighbors

- 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

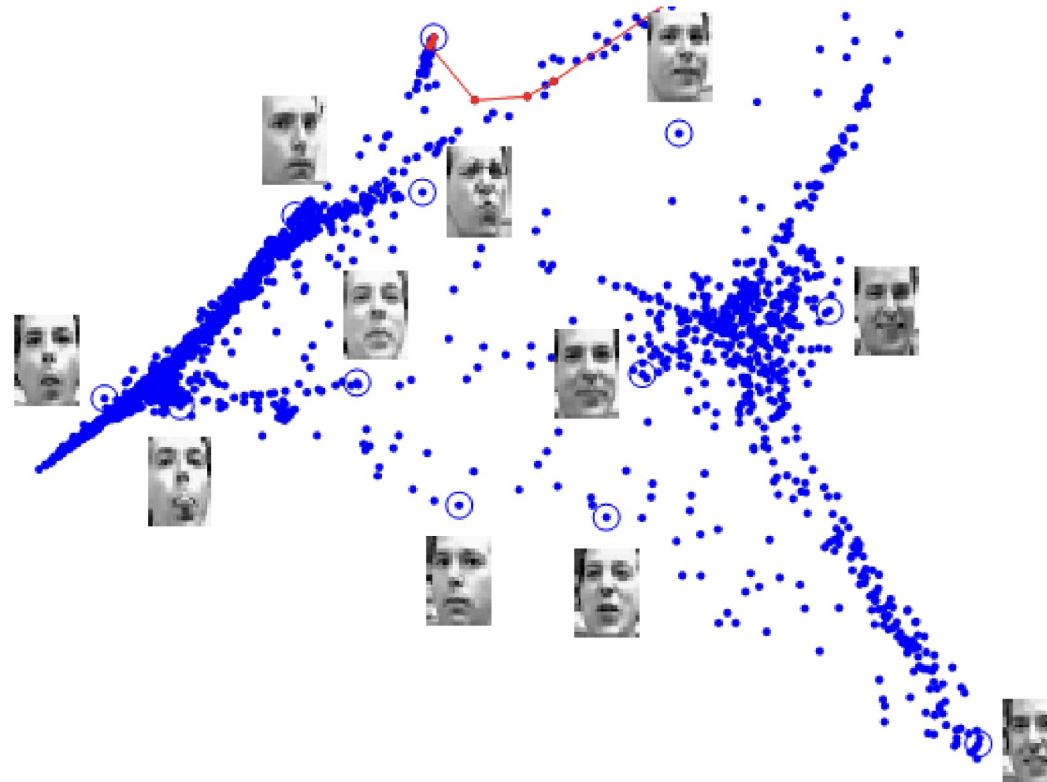


k-Nearest Neighbors

- Mỗi ví dụ học \mathbf{x} được biểu diễn bởi 2 thành phần:
 - Mô tả của ví dụ: $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_d)$, trong đó $\mathbf{x}_i \in \mathbb{R}$
 - Nhãn lớp $c \in \mathbf{C}$, với \mathbf{C} là tập nhãn cho trước
- Giai đoạn học
 - Đơn giản là lưu lại các ví dụ học trong tập học: \mathcal{D}
- Giai đoạn phân lớp: Để phân lớp cho một ví dụ (mới) \mathbf{z}
 - Với mỗi ví dụ học $\mathbf{x} \in \mathcal{D}$, tính khoảng cách giữa \mathbf{x} và \mathbf{z}
 - Xác định tập $NB(\mathbf{z})$ - các láng giềng gần nhất của \mathbf{z}
 - Gồm k ví dụ học trong \mathcal{D} gần nhất với \mathbf{z} tính theo một hàm khoảng cách d
 - Phân \mathbf{z} vào lớp chiếm số đông (the majority class) trong số các lớp ví dụ trong $NB(\mathbf{z})$

k-Nearest Neighbors - Các vấn đề cốt lõi

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



k-Nearest Neighbors - Các vấn đề cốt lõi

- Về lý thuyết thì 1-NN cũng có thể là một trong số các phương pháp tối ưu.
- 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/nhiễu 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

- 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 ($x_i \in \mathbb{R}$)
 - 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 hình học (Geometry distance functions)

- Hàm Minkowski (p-norm):

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

- Hàm Manhattan (p = 1):

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

- Hàm Euclide (p = 2):

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

Hàm tính khoảng cách hình học (Geometry distance functions)

- Có thể chuẩn hóa miền giá trị (đưa về cùng một khoảng giá trị)
- **Các thuộc tính khác nhau** có thể (nên) có **mức độ ảnh hưởng khác nhau** đối với giá trị khoảng cách
 - 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)
- Các láng giềng khác nhau có ảnh hưởng khác nhau đối với việc phân lớp/dự đoán cho z
 - Có thể 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!

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

Decision Tree (Cây quyết định)

Bài toán: quyết định có đợi 1 bàn ở quán ăn không, dựa trên các thông tin sau:

1. **Lựa chọn khác:** có quán ăn nào khác gần đó không?
2. **Quán rượu:** có khu vực phục vụ đồ uống gần đó không?
3. **Fri/Sat:** hôm nay là thứ sáu hay thứ bảy?
4. **Đói:** chúng ta đã đói chưa?
5. **Khách hàng:** số khách trong quán (không có, vài người, đầy)
6. **Giá cả:** khoảng giá (\$,\$\$,\$\$\$)
7. **Mưa:** ngoài trời có mưa không?
8. **Đặt chỗ:** chúng ta đã đặt trước chưa?
9. **Loại:** loại quán ăn (Pháp, Ý, Thái, quán ăn nhanh)
10. **Thời gian đợi:** 0-10, 10-30, 30-60, >60

Decision Tree (Cây quyết định)

- Các mẫu được miêu tả dưới dạng các giá trị thuộc tính (logic, rời rạc, liên tục)
- Ví dụ, tình huống khi đợi 1 bàn ăn

Example	Attributes										Target Wait
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
X_1	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X_2	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X_6	F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	\$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	\$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	\$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	\$	F	F	Burger	30–60	T

Các loại (lớp) của mẫu là khẳng định (T) hoặc phủ định (F)

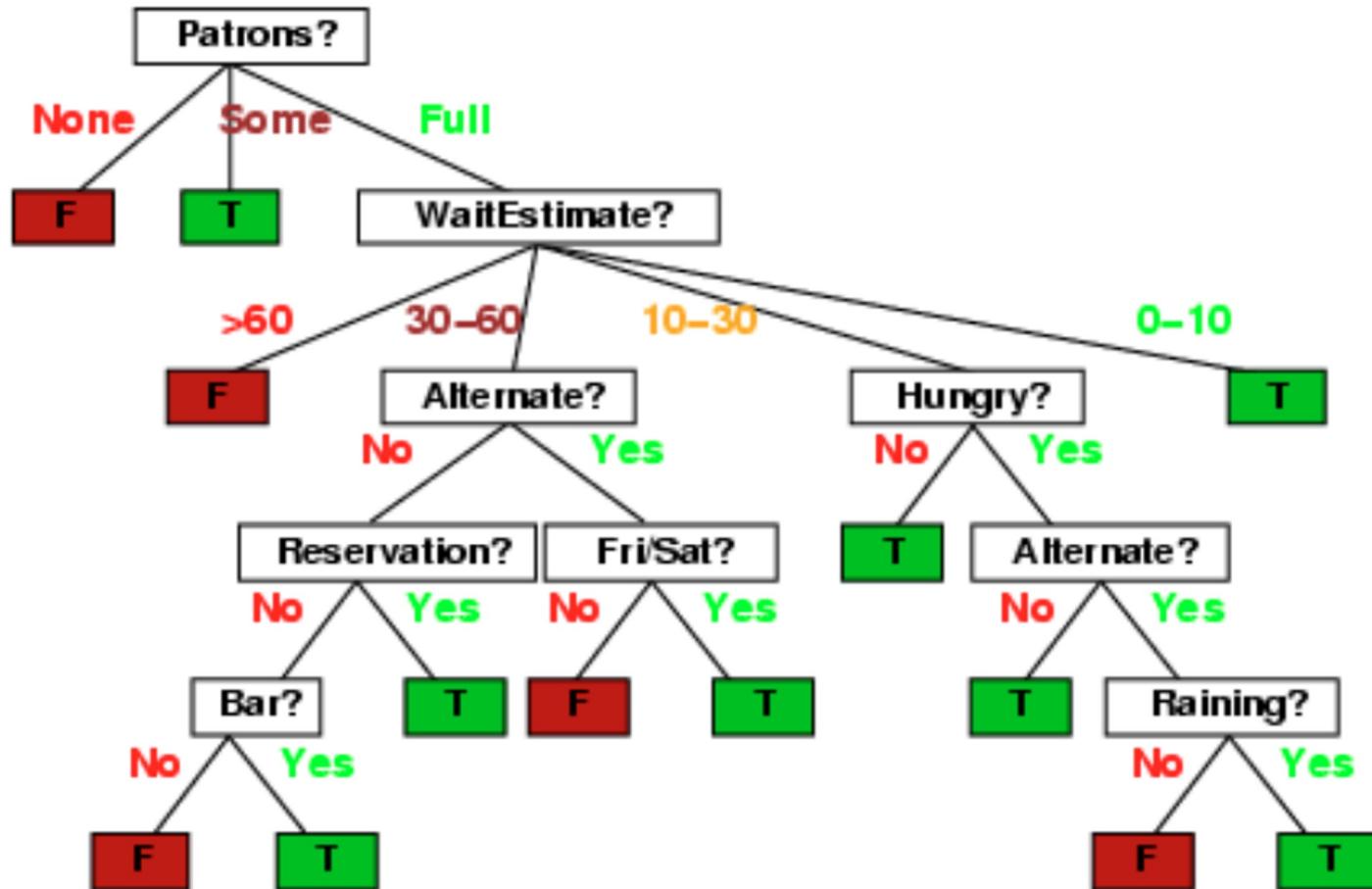
Decision Tree (Cây quyết định)

Attributes										Target Wait
Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
T	F	F	T	Full	\$	F	F	Thai	30–60	F
F	T	F	F	Some	\$	F	F	Burger	0–10	T
T	F	T	T	Full	\$	F	F	Thai	10–30	T
T	F	T	F	Full	\$\$\$	F	T	French	>60	F
F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
F	T	F	F	None	\$	T	F	Burger	0–10	F
F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
F	T	T	F	Full	\$	T	F	Burger	>60	F
T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
F	F	F	F	None	\$	F	F	Thai	0–10	F
T	T	T	T	Full	\$	F	F	Burger	30–60	T

Patrons, WaitEstimates, Alternative, Hungry, Rain

Decision Tree (Cây quyết định)

Cây quyết định là cách biểu diễn các giả thiết.



Decision Tree (Cây quyết định)

Không gian giả thiết:

Khi có n thuộc tính Boolean, số lượng các cây quyết định là?

= số các hàm Boolean

= số các giá trị khác nhau trong bảng ví dụ mẫu với 2^n hàng

= $2^{\wedge} 2^n$

Ví dụ, với 6 thuộc tính Boolean, có

18,446,744,073,709,551,616 cây

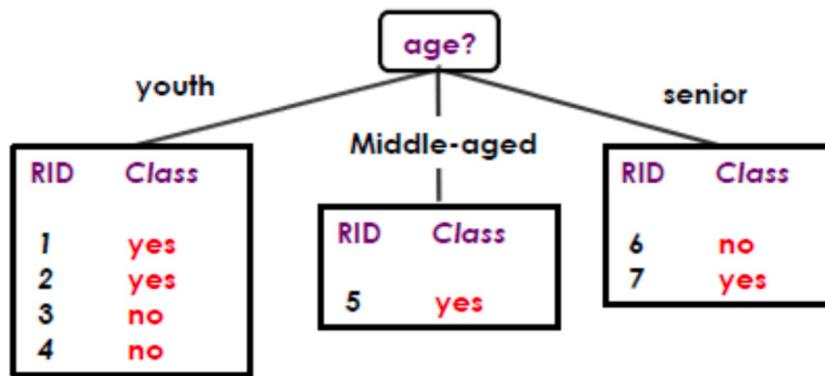
Decision Tree (Cây quyết định) - ID3

- ID3 (Iterative Dichotomiser 3) thực hiện tìm kiếm tham lam trên không gian các cây quyết định (do Ross Quinlan đề xuất năm 1986).
- Giả thuyết mỗi quan sát x được biểu diễn bởi d thuộc tính
 - $x = (x_1, x_2, \dots, x_d)$
 - Mỗi x_i là một thuộc tính rời rạc (discrete) hoặc định danh (categorical)
- Mỗi quan sát trong tập học có một nhãn lớp tương ứng.
- Xây dựng (học) một cây quyết định một cách quy nạp, theo chiến lược top-down, bắt đầu từ nút gốc ứng với tất cả tập học.

Decision Tree (Cây quyết định) - ID3

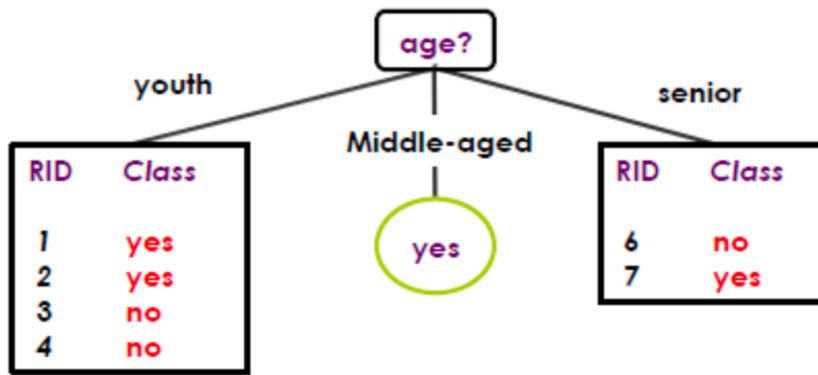
- Ở mỗi nút tiếp theo, chọn thuộc tính kiểm tra (là thuộc tính có khả năng phân loại tốt nhất đối với các ví dụ học gắn với nút đó).
- Tạo mới một cây con của nút hiện tại cho mỗi giá trị của thuộc tính kiểm tra, và tập học sẽ được tách ra (thành các tập con) tương ứng với cây con vừa tạo.
- Quá trình phát triển cây quyết định sẽ tiếp tục cho đến khi:
 - Tất cả các mẫu hoặc thuộc tính đã được sử dụng và biểu quyết theo số đông được sử dụng để gán node lá
 - Tất cả các mẫu thuộc cùng một lớp
- Chú ý: Mỗi thuộc tính chỉ được phép xuất hiện tối đa 1 lần đối với bất kỳ một đường đi nào trong cây.

ID3 - Ví dụ



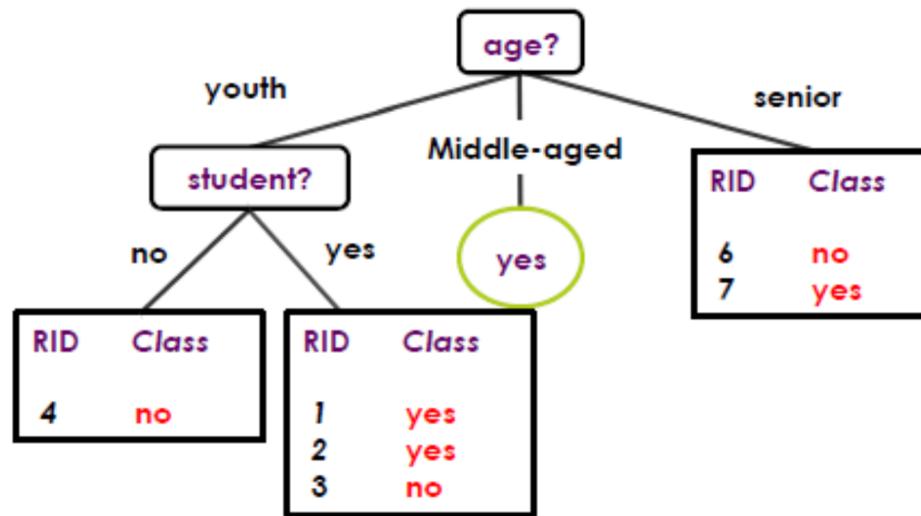
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

ID3 - Ví dụ



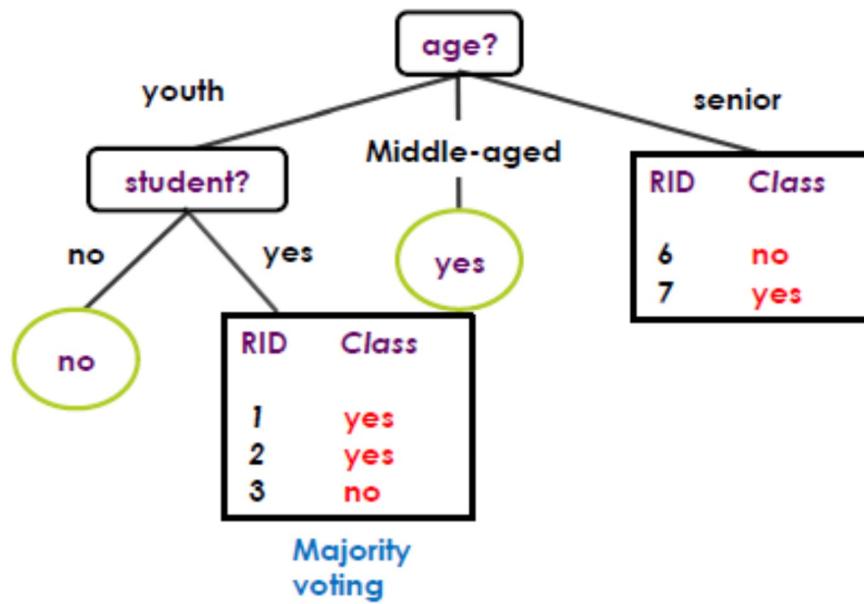
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

ID3 - Ví dụ



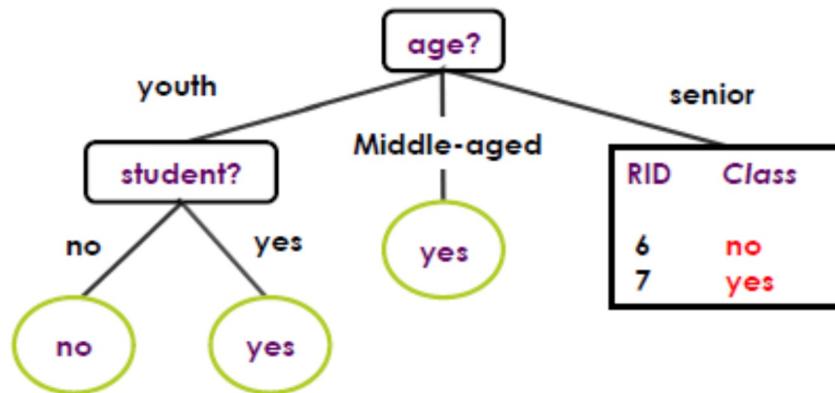
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

ID3 - Ví dụ



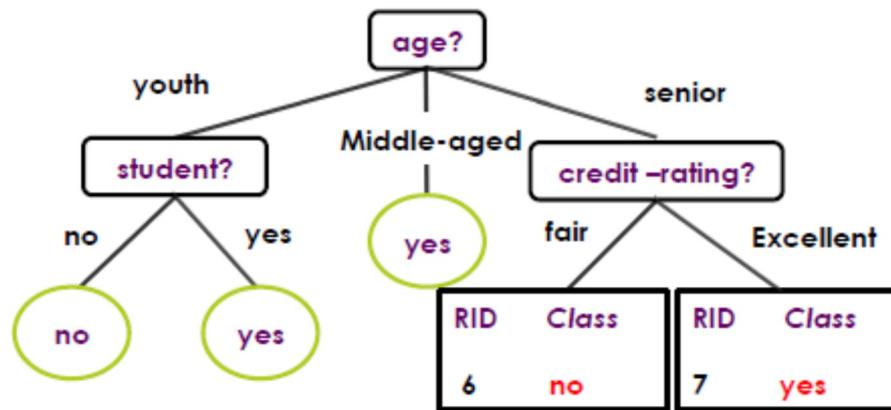
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

ID3 - Ví dụ



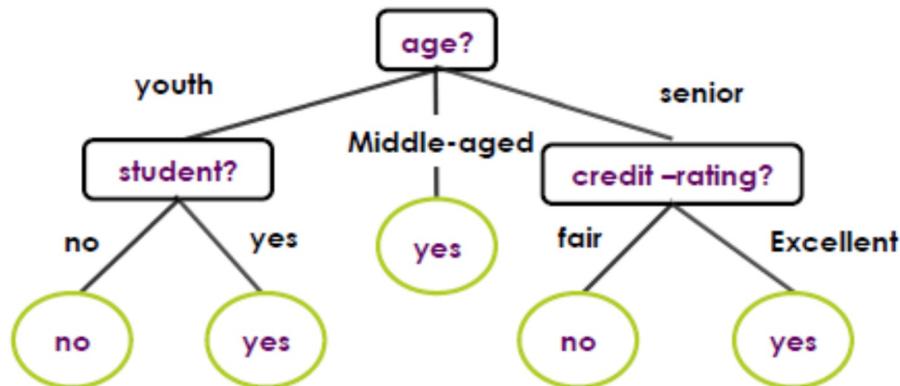
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

ID3 - Ví dụ



RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

ID3 - Ví dụ

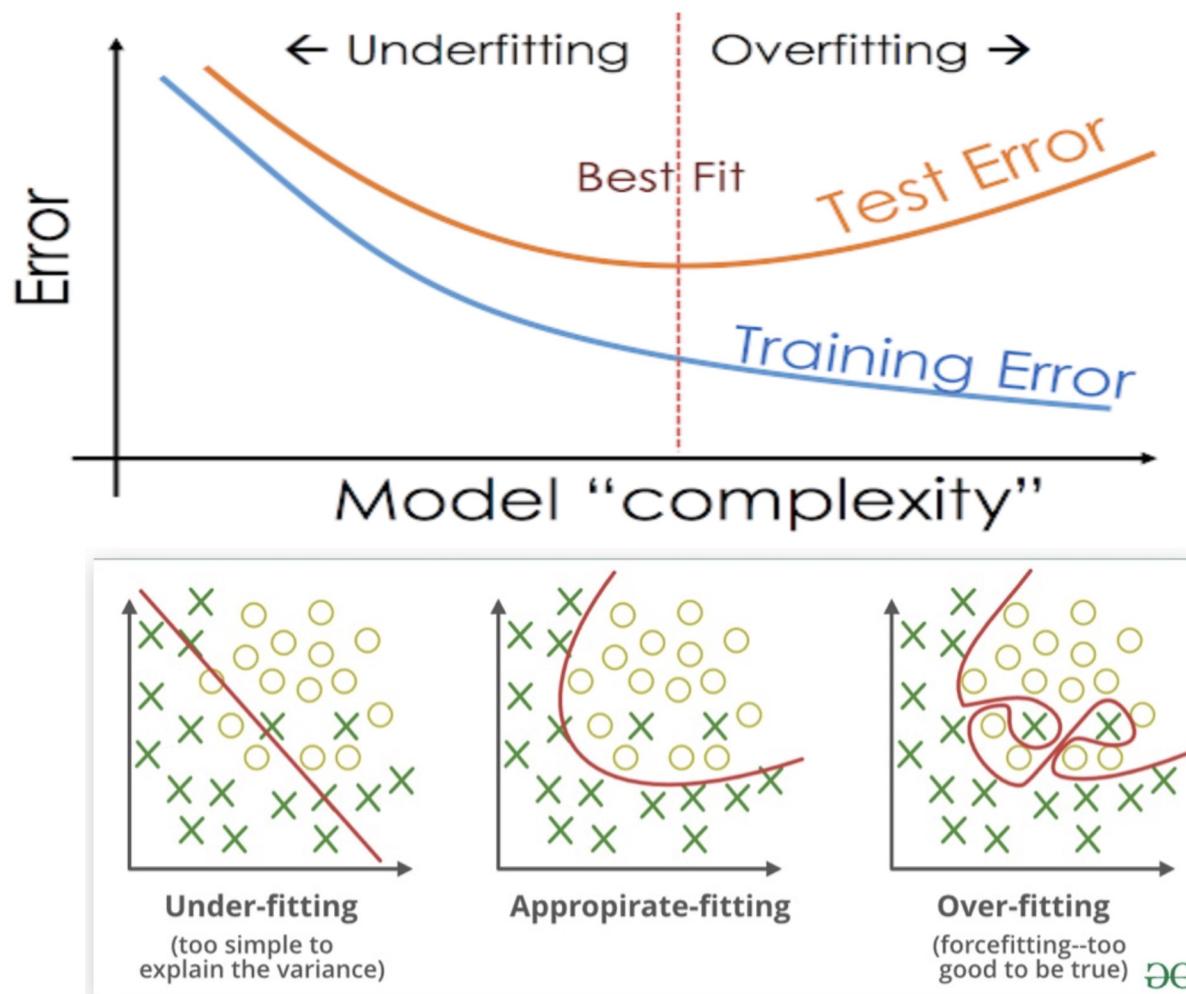


RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	yes
7	senior	yes	excellent	no

Overfitting

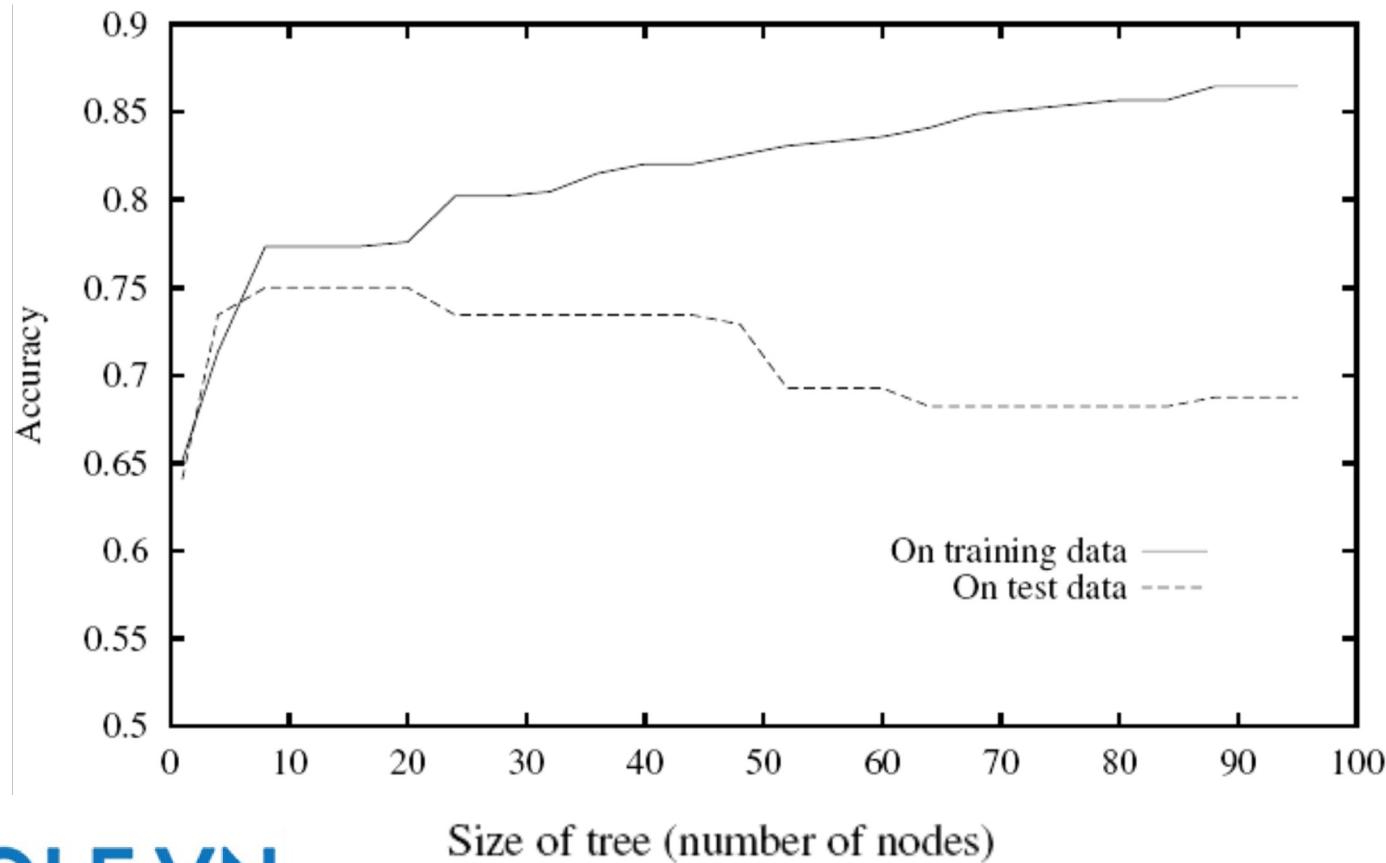
- Hàm h được gọi là overfitting nếu tồn tại hàm g mà:
 - g có thể tồi hơn h đối với tập huấn luyện
 - nhưng g tốt hơn h đối với dữ liệu tương lai.
- Model học thuộc dữ liệu trên tập huấn luyện
 - không có tính khái quát hóa
 - không hoàn thành tốt khi đem vào validate/test
- Vài nguyên nhân gây ra Overfitting:
 - Hàm h quá phức tạp
 - Lỗi (nhiều) trong tập huấn luyện (do quá trình thu thập/xây dựng tập dữ liệu)
 - Số lượng các ví dụ học quá nhỏ, không đại diện cho toàn bộ tập (phân bố) của các ví dụ của bài toán học

Overfitting



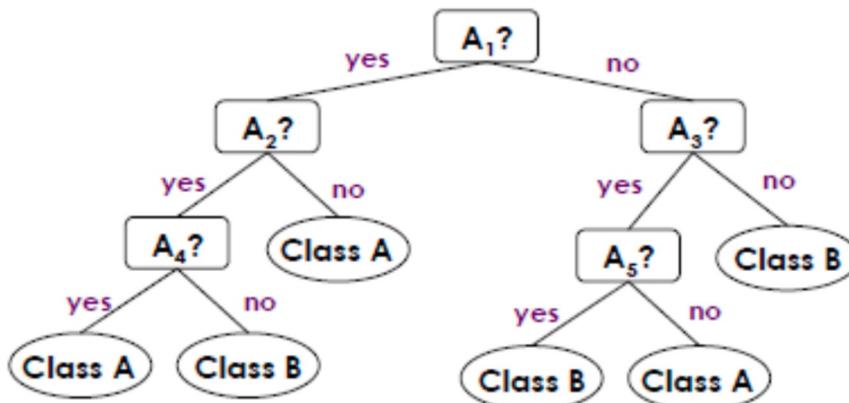
Overfitting

- Khi tăng cỡ lớn của một Cây quyết định thì chất lượng phán đoán của nó có thể giảm dần, mặc dù độ chính xác trên tập huấn luyện tăng dần

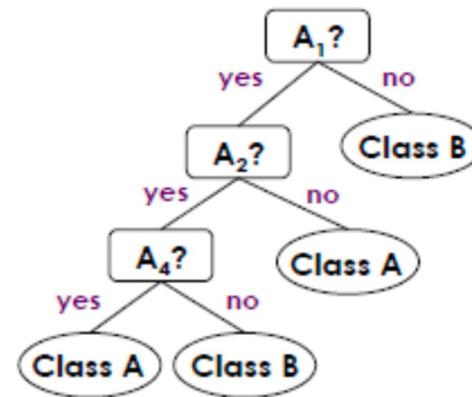


Overfitting

- Nhiều nhánh của cây quyết định phản ánh sự bất thường trong dữ liệu học (có thể do nhiễu hoặc giá trị ngoại lai)
- Độ chính xác thấp đối với các mẫu mới (kiểm tra)
- Giải pháp: Cắt tỉa (Pruning)
 - Loại bỏ những nhánh ít tin cậy



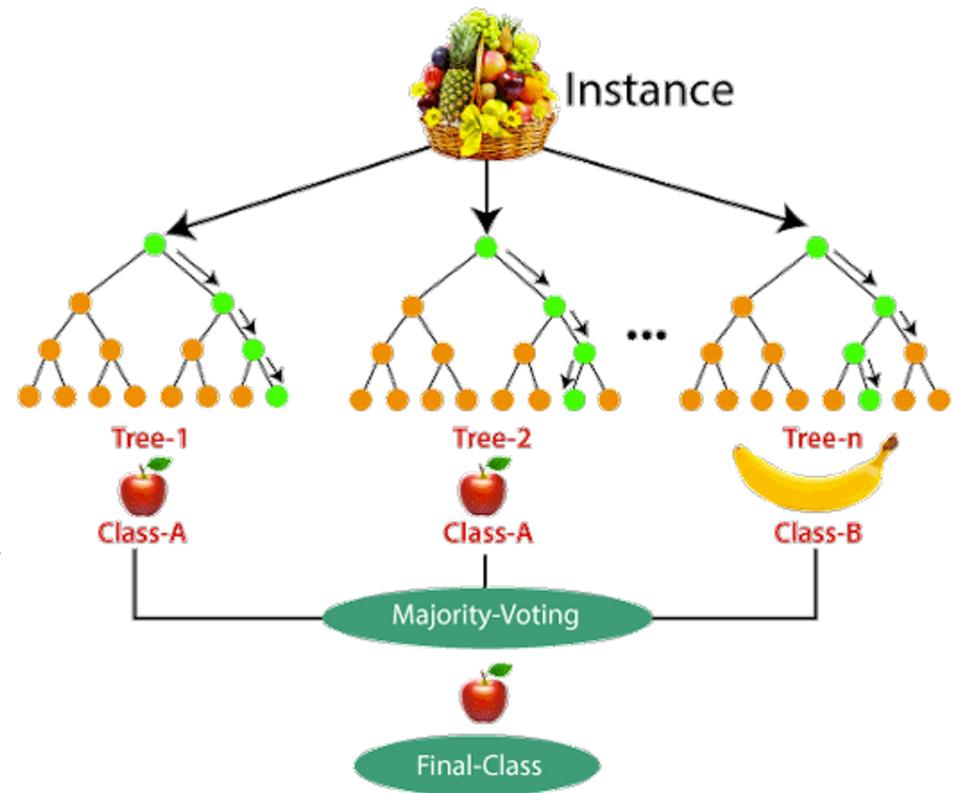
Before Pruning



After Pruning

Random Forest

- Mục tiêu: tránh overfitting
- Ý tưởng: chia thành nhiều decision tree, dữ liệu ở mỗi cây được lấy random từ tập training, kết quả sẽ được lấy trung bình hoặc chọn từ đa số để phục vụ cho phân loại và hồi quy.
- Đặc điểm quan trọng:
 - Từng cây chỉ có số lượng feature nhất định, và không cây nào giống nhau
 - Mỗi cây được tạo tách biệt và có thể được tính toán song song => giảm thời gian tính toán
 - Tính ổn định cao do kết quả dựa trên đa số/trung bình



So sánh Decision Tree và Random Forest

Decision Tree

- Dễ bị overfit nếu không bị kiểm soát
- Tính toán nhanh hơn
- Khi một bộ dữ liệu với các thuộc tính được đưa vào decision tree, nó sẽ hình thành một số quy tắc/hàm để tiến hành dự đoán

Random Forest

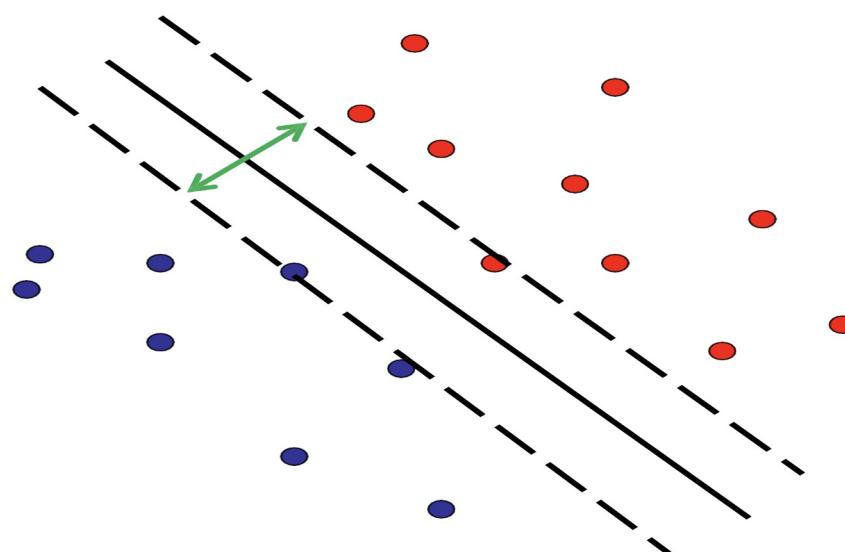
- Được tạo nên từ các tập hợp con của dữ liệu, kết quả cuối lấy từ trung bình, hoặc xét đa số nên giải quyết được overfit
- Tính toán chậm
- Random forest chọn ngẫu nhiên, dựng các decision tree và lấy kết quả trung bình. Nó không sử dụng bất cứ bộ công thức nào

SVM - Support vector machine

- Máy vectơ hỗ trợ (Support vector machine - SVM)
 - V. Vapnik và các đồng nghiệp của ông vào những năm 1970s ở Nga
 - Trở nên hoàn chỉnh và phổ biến vào những năm 1990s
- SVM là một phương pháp **phân lớp tuyến tính** (linear classifier), với mục đích xác định một siêu phẳng (hyperplane) để phân tách **hai lớp** của dữ liệu.
 - Ví dụ: lớp có nhãn dương (positive) và lớp có nhãn âm (negative)
- **Các hàm nhân (kernel functions)**, cũng được gọi là các hàm biến đổi (transformation functions), được dùng cho các trường hợp phân lớp phi tuyến

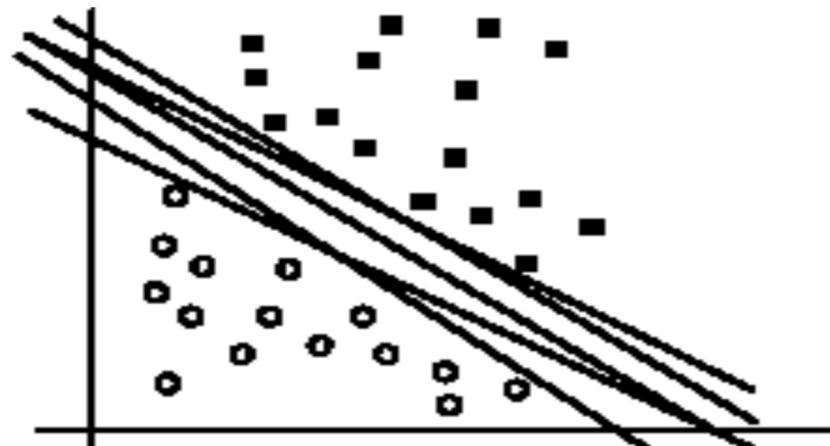
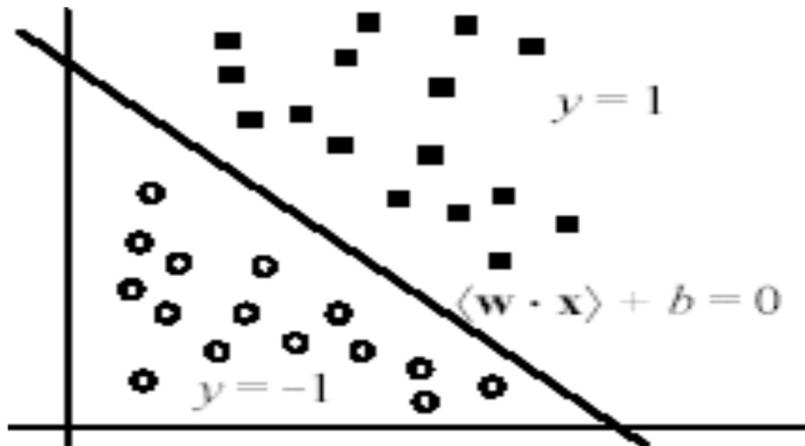
SVM - Support vector machine

- SVM là một trong những thuật toán phân lớp phổ biến và hiệu quả. Ý tưởng khá đơn giản, và để sử dụng ta cần một chút kiến thức về tối ưu.
- Giả sử rằng có 2 lớp dữ liệu mô tả bởi các điểm (feature vector) trong không gian nhiều chiều, và tồn tại *siêu phẳng* phân chia chính xác 2 lớp đó.



SVM - Support vector machine

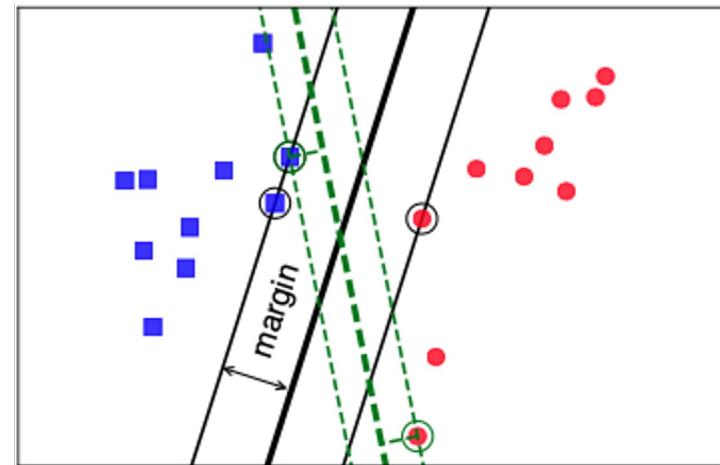
- Siêu phẳng phân tách các quan sát thuộc lớp dương và các quan sát thuộc lớp âm: $\mathbf{w} \cdot \mathbf{x} + b = 0$
- Còn được gọi là **ranh giới (bè mặt) quyết định**
- Tồn tại nhiều siêu phẳng phân tách. Chọn cái nào?



SVM - Support vector machine

- SVM giải quyết câu hỏi này dựa bằng cách xây dựng định nghĩa về lề và đưa ra thuật toán tối ưu nó.
- Lề (margin) của một lớp được định nghĩa là khoảng cách từ các điểm gần nhất của lớp đó tới mặt phân chia.

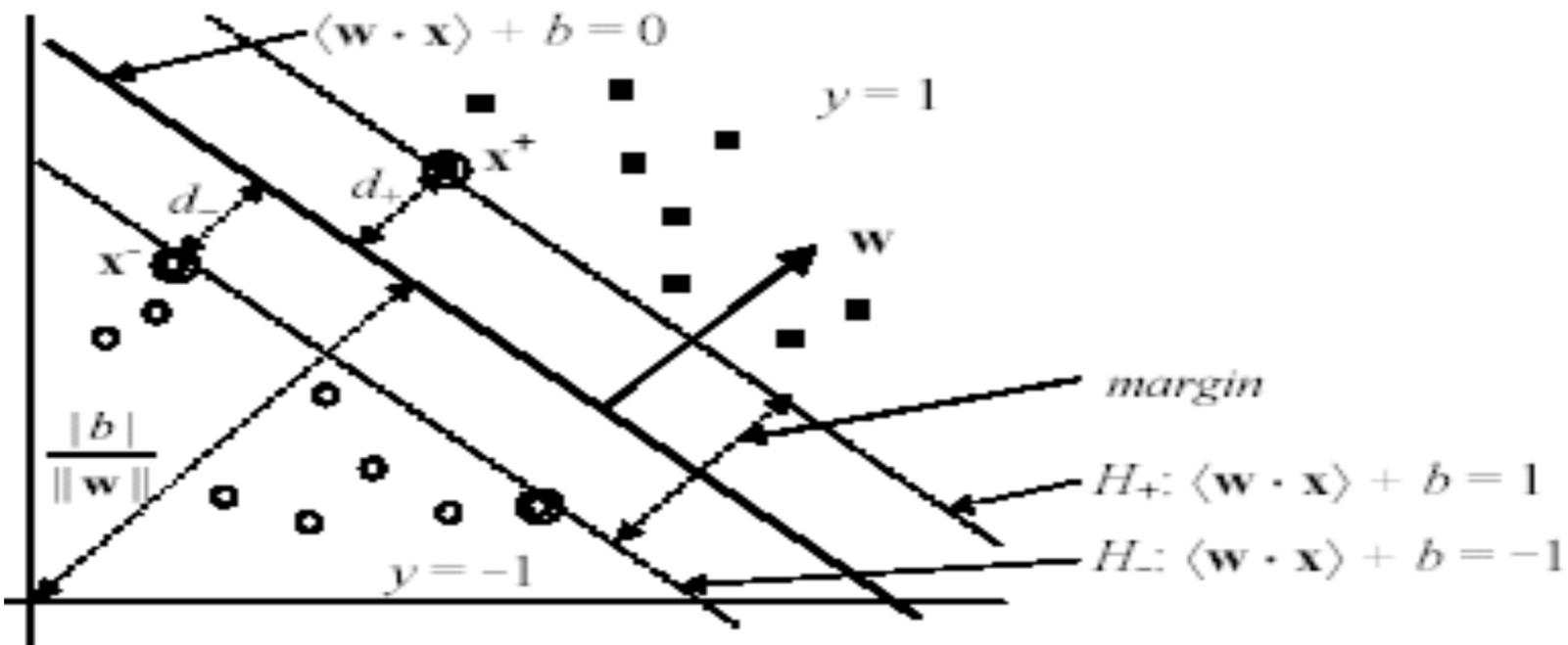
$$\frac{|w_1x_{10} + w_2x_{20} + \dots + w_dx_{d0} + b|}{\sqrt{w_1^2 + w_2^2 + \dots + w_d^2}} = \frac{|\mathbf{w}^T \mathbf{x}_0 + b|}{\|\mathbf{w}\|_2}$$



- Trong đó:
 - $\|\mathbf{w}\|$ là độ dài của \mathbf{w} : $\|\mathbf{w}\| = \sqrt{\langle \mathbf{w} \cdot \mathbf{w} \rangle} = \sqrt{w_1^2 + w_2^2 + \dots + w_d^2}$
 - $\mathbf{w} = [w_1, w_2, \dots, w_d]^T$
 - $\mathbf{x}_0 = [x_{10}, x_{20}, \dots, x_{d0}]^T$

SVM - Support vector machine

- SVM lựa chọn mặt siêu phẳng phân tách có **lề (margin) lớn nhất**
- Lý thuyết học máy đã chỉ ra rằng một mặt siêu phẳng phân tách như thế sẽ tối thiểu hóa giới hạn lỗi (phân lớp) mắc phải (so với mọi siêu phẳng khác)



SVM - Support vector machine

- Tính toán d_+ : khoảng cách từ \mathbf{x}^+ đến ($\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$)
 - Áp dụng các biểu thức [Eq.3-4]:

$$d_+ = \frac{|\langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b|}{\|\mathbf{w}\|} = \frac{|1|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$$

-
- Tính toán d_- : khoảng cách từ \mathbf{x}^- đến ($\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$)
 - Áp dụng các biểu thức [Eq.3-4]:

$$d_- = \frac{|\langle \mathbf{w} \cdot \mathbf{x}^- \rangle + b|}{\|\mathbf{w}\|} = \frac{|-1|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$$

- Tính toán mức lề

$$\text{margin} = d_+ + d_- = \frac{2}{\|\mathbf{w}\|}$$

SVM - Support vector machine

- Học SVM tương đương với giải quyết bài toán tối ưu bậc 2 sau đây
- Tìm w và b sao cho: $margin = \frac{2}{\|w\|}$ đạt cực đại
- Với điều kiện:

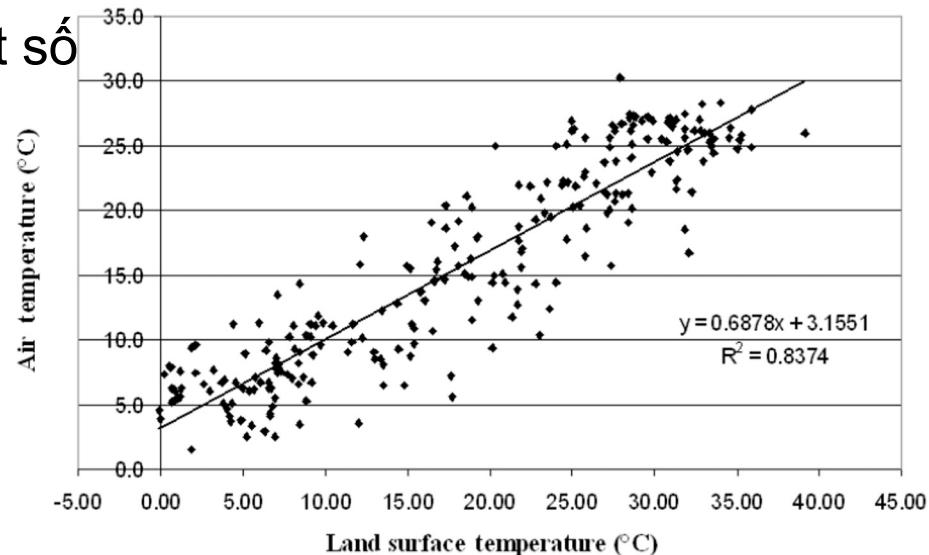
$$\begin{cases} \langle w \cdot x_i \rangle + b \geq 1, & \text{if } y_i = 1 \\ \langle w \cdot x_i \rangle + b \leq -1, & \text{if } y_i = -1 \end{cases}$$

với mọi ví dụ huấn luyện x_i ($i=1..r$)

Regression

Regression

- Classification: dự đoán loại (category)
- Regression: dự đoán giá trị số
 - Nhiệt độ trong tuần tới sẽ như thế nào?
 - Giá nhà đất khu vực nội thành là bao nhiêu?
 - Tương tự như classification
 - Đầu vào cho các mô hình này có thể là cả số và phân loại
 - Đầu ra dự đoán là một số



Hồi quy tuyến tính (linear regression)

- **Bài toán hồi quy:** cần học một hàm $y = f(x)$ từ một tập học cho trước $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_M, y_M)\}$ trong đó $y_i \approx f(x_i)$ với mọi i .
 - Mỗi quan sát được biểu diễn bằng một véctơ n chiều, chẳng hạn $x_i = (x_{i1}, x_{i2}, \dots, x_{in})^T$
 - Mỗi chiều biểu diễn một thuộc tính (attribute/feature)
- **Mô hình tuyến tính:** nếu giả thuyết hàm $y = f(x)$ là hàm có dạng tuyến tính

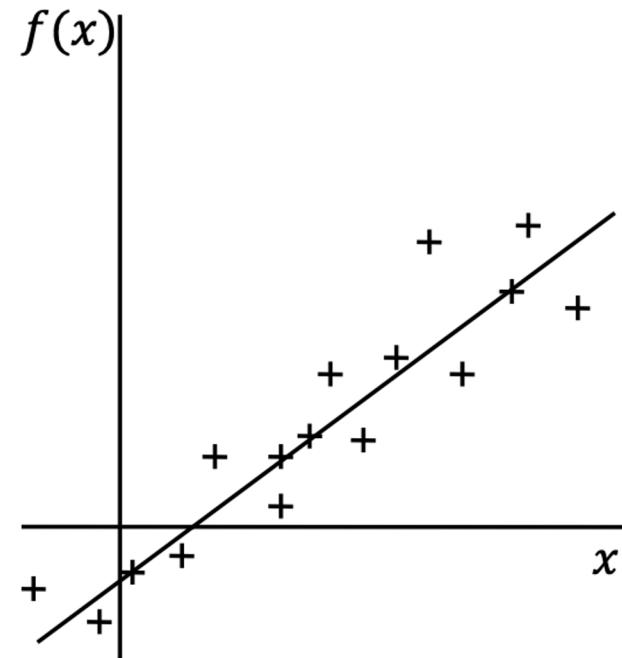
$$f(x) = w_0 + w_1x_1 + \dots + w_nx_n$$

- Học một hàm hồi quy tuyến tính thì tương đương với việc học véctơ trọng số $w = (w_0, w_1, \dots, w_n)^T$

Hồi quy tuyến tính (linear regression)

- Hàm tuyến tính $f(x)$ nào phù hợp?

0.13	-0.91
1.02	-0.17
3.17	1.61
-2.76	-3.31
1.44	0.18
5.28	3.36
-1.74	-2.46
7.93	5.56
...	...



Ví dụ: $f(x) = -1.02 + 0.83x$

Hồi quy tuyến tính (linear regression)

- Đối với mỗi quan sát $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$
 - Giá trị đầu ra mong muốn c_x
(Không biết trước đối với các quan sát trong tương lai)
 - Giá trị phán đoán (bởi hệ thống)
- Phán đoán cho quan sát tương lai $\mathbf{z} = (z_1, z_2, \dots, z_n)^T$
 - Cần dự đoán giá trị đầu ra, bằng cách áp dụng hàm mục tiêu đã học được f :

$$f(\mathbf{z}) = w_0 + w_1 z_1 + \dots + w_n z_n$$

Hồi quy tuyến tính (linear regression)

Mục tiêu học: học một hàm f^* sao cho khả năng phán đoán trong tương lai là tốt nhất.

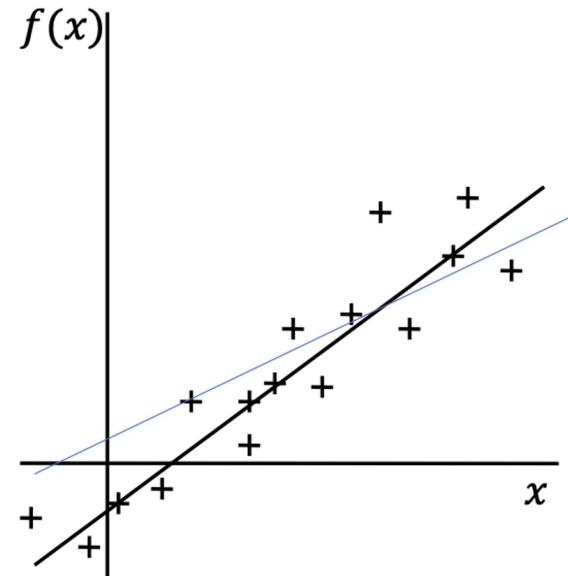
- Tức là sai số $|c_z - f(z)|$ là nhỏ nhất cho các quan sát tương lai z .
- Khả năng *tổng quát hóa* (generalization) là tốt nhất.

Vấn đề: Có vô hạn hàm tuyến tính!!

- Làm sao để học? Quy tắc nào?

Dùng một tiêu chuẩn để đánh giá.

- Tiêu chuẩn thường dùng là *hàm mất mát* (generalization error, loss function, ...)



Hồi quy tuyến tính (linear regression) - Hàm mất mát

- Định nghĩa hàm mất mát E
 - Lỗi (error/loss) phán đoán cho quan sát $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$

$$r(\mathbf{x}) = [c_x - f^*(\mathbf{x})]^2 = (c_x - w_0 - w_1x_1 - \dots - w_nx_n)^2$$

- Lỗi của hệ thống trên toàn bộ không gian của \mathbf{x} :

$$E = E_x[r(\mathbf{x})] = E_x[c_x - f^*(\mathbf{x})]^2$$

- Mục tiêu học là tìm hàm f^* mà E là nhỏ nhất:

$$f^* = \arg \min_{f \in H} E_x [r(\mathbf{x})]$$

- Trong đó H là không gian của hàm f .
- **Nhưng**: trong quá trình học ta không thể làm việc được với bài toán này.

Hồi quy tuyến tính (linear regression) - Hàm mất mát

- Ta chỉ quan sát được một tập $\mathbf{D} = \{(x_1, y_1), (x_2, y_2), \dots, (x_M, y_M)\}$. Cần học hàm f từ \mathbf{D} .
- *Lỗi thực nghiệm* (empirical loss; residual sum of squares)

$$RSS(f) = \sum_{i=1}^M (y_i - f(x_i))^2 = \sum_{i=1}^M (y_i - w_0 - w_1 x_{i1} - \dots - w_n x_{in})^2$$

- RSS/M là một xấp xỉ của $E_x[r(\mathbf{x})]$ trên tập học \mathbf{D}
- **Lỗi tổng quát hóa** (generalization error) của hàm f là
$$\left| \frac{1}{M} RSS(f) - E_x[r(\mathbf{x})] \right|$$

Bình phương tối thiểu (Ordinary Least Squares - OLS)

- Cho trước D, ta đi tìm hàm f mà có RSS nhỏ nhất.

$$f^* = \arg \min_{f \in H} RSS(f)$$

$$\Leftrightarrow \mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_{i=1}^M (y_i - w_0 - w_1 x_{i1} - \cdots - w_n x_{in})^2 \quad (1)$$

Đây được gọi là bình phương tối thiểu (least squares).

- Tìm nghiệm \mathbf{w}^* bằng cách lấy đạo hàm của RSS và giải phương trình $RSS' = 0$. Thu được:

$$\mathbf{w}^* = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y}$$

Trong đó A là ma trận dữ liệu cỡ $M \times (n+1)$ mà hàng thứ i là

$\mathbf{A}_i = (1, x_{i1}, x_{i2}, \dots, x_{in})$; \mathbf{B}^{-1} là ma trận nghịch đảo; $\mathbf{y} = (y_1, y_2, \dots, y_M)$.

Chú ý: giả thuyết $\mathbf{A}^T \mathbf{A}$ tồn tại nghịch đảo.

Bình phương tối thiểu (Ordinary Least Squares - OLS)

- Input: $\mathbf{D} = \{(x_1, y_1), (x_2, y_2), \dots, (x_M, y_M)\}$
- Output: \mathbf{w}^*
- Học \mathbf{w}^* bằng cách tính:

$$\mathbf{w}^* = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y}$$

Trong đó:

- \mathbf{A} là ma trận dữ liệu cỡ $M \times (n+1)$ mà hàng thứ i là $\mathbf{A}_i = (1, x_{i1}, x_{i2}, \dots, x_{in})$;
- \mathbf{B}^{-1} là ma trận nghịch đảo
- $\mathbf{y} = (y_1, y_2, \dots, y_M)$

Chú ý: giả thuyết $\mathbf{A}^T \mathbf{A}$ tồn tại nghịch đảo.

- Phán đoán cho quan sát mới \mathbf{x} :

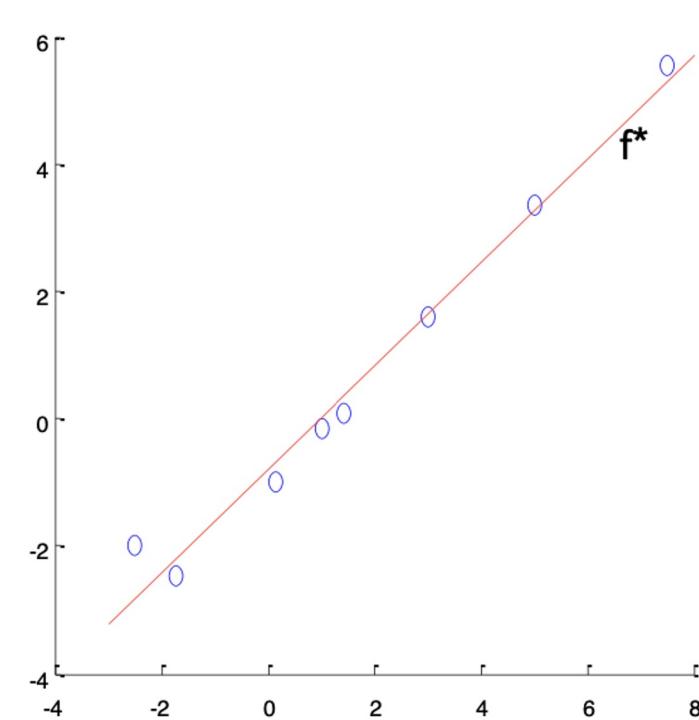
$$y_x = w_0^* + w_1^* x_1 + \dots + w_n^* x_n$$

Bình phương tối thiểu - Ví dụ

Kết quả học bằng bình phương tối thiểu

x	y
0.13	-1
1.02	-0.17
3	1.61
-2.5	-2
1.44	0.1
5	3.36
-1.74	-2.46
7.5	5.56

$$f^*(x) = 0.81x - 0.78$$



Bình phương tối thiểu - Nhược điểm

- Nếu $\mathbf{A}^T \mathbf{A}$ không tồn tại nghịch đảo thì không học được.
 - Nếu các thuộc tính (cột của A) có phụ thuộc lẫn nhau.
- Độ phức tạp tính toán lớn do phải tính ma trận nghịch đảo.
→ Không làm việc được nếu số chiều n lớn.
- Khả năng overfitting cao vì việc học hàm f chỉ quan tâm tối thiểu lỗi đối với tập học đang có.

Hồi quy Ridge

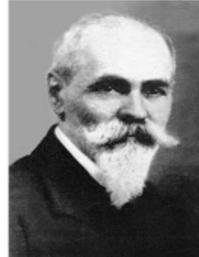
- Cho trước $\mathbf{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_M, y_M)\}$, ta đi giải bài toán:

$$f^* = \arg \min_{f \in H} RSS(f) + \lambda \|\mathbf{w}\|_2^2$$
$$\Leftrightarrow \mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_{i=1}^M (y_i - \mathbf{A}_i \mathbf{w})^2 + \lambda \sum_{j=0}^n w_j^2$$

- Trong đó $\mathbf{A}_i = (1, x_{i1}, x_{i2}, \dots, x_{in})$ được tạo ra từ x_i ; λ là một hằng số phạt ($\lambda > 0$).



Tikhonov,
smoothing an ill-
posed problem



Zaremba, model
complexity
minimization



Bayes: priors
over parameters



Andrew Ng: need no
maths, but it prevents
overfitting!

Hồi quy Ridge

- Tương đương với việc giải bài toán sau:

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_{i=1}^M (y_i - \mathbf{A}_i \mathbf{w})^2$$

sao cho $\sum_{j=0}^n w_j^2 \leq t$

t là một hằng số nào đó.

- Đại lượng hiệu chỉnh (phạt) $\lambda \|\mathbf{w}\|_2^2$
 - Có vai trò hạn chế độ lớn của \mathbf{w}^* (hạn chế không gian hàm f).
 - Đánh đổi chất lượng của hàm f đối với tập học \mathbf{D} , để có khả năng phán đoán tốt hơn với quan sát tương lai.

Hồi quy Ridge

- Tìm nghiệm \mathbf{w}^* bằng cách lấy đạo hàm của RSS và giải phương trình $\text{RSS}' = 0$. Thu được:

$$\mathbf{w}^* = (\mathbf{A}^T \mathbf{A} + \lambda \mathbf{I}_{n+1})^{-1} \mathbf{A}^T \mathbf{y}$$

- Trong đó \mathbf{A} là ma trận dữ liệu cỡ $M \times (n+1)$ mà hàng thứ i là $(1, x_{i1}, x_{i2}, \dots, x_{in})$; $\mathbf{y} = (y_1, y_2, \dots, y_M)^T$; \mathbf{I}_{n+1} là ma trận đơn vị cỡ $n+1$.
- So sánh với phương pháp bình phương tối thiểu:
 - Tránh được trường hợp ma trận dữ liệu suy biến. Hồi quy Ridge luôn làm việc được.
 - Khả năng overfitting thường ít hơn.
 - Lỗi trên tập học có thể nhiều hơn.
- Chú ý: chất lượng của phương pháp phụ thuộc rất nhiều vào sự lựa chọn của tham số λ .*

Hồi quy Ridge: thuật toán

- Input: $\mathbf{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_M, y_M)\}$, hằng số $\lambda > 0$
- Output: \mathbf{w}^*
- Học \mathbf{w}^* bằng cách tính:

$$\mathbf{w}^* = (\mathbf{A}^T \mathbf{A} + \lambda \mathbf{I}_{n+1})^{-1} \mathbf{A}^T \mathbf{y}$$

- Trong đó \mathbf{A} là ma trận dữ liệu cỡ $M \times (n+1)$ mà hàng thứ i là $\mathbf{A}_i = (1, x_{i1}, x_{i2}, \dots, x_{in})$; \mathbf{B}^{-1} là ma trận nghịch đảo; $\mathbf{y} = (y_1, y_2, \dots, y_M)^T$.
 - Phán đoán cho quan sát mới \mathbf{x} :
- $$y_x = w_0^* + w_1^* x_1 + \dots + w_n^* x_n$$
- **Chú ý:** để tránh vài ảnh hưởng xấu từ độ lớn của y , ta nên loại bỏ thành phần w_0 trong đại lượng phạt ở công thức (2). Khi đó nghiệm \mathbf{w}^* sẽ thay đổi một chút.

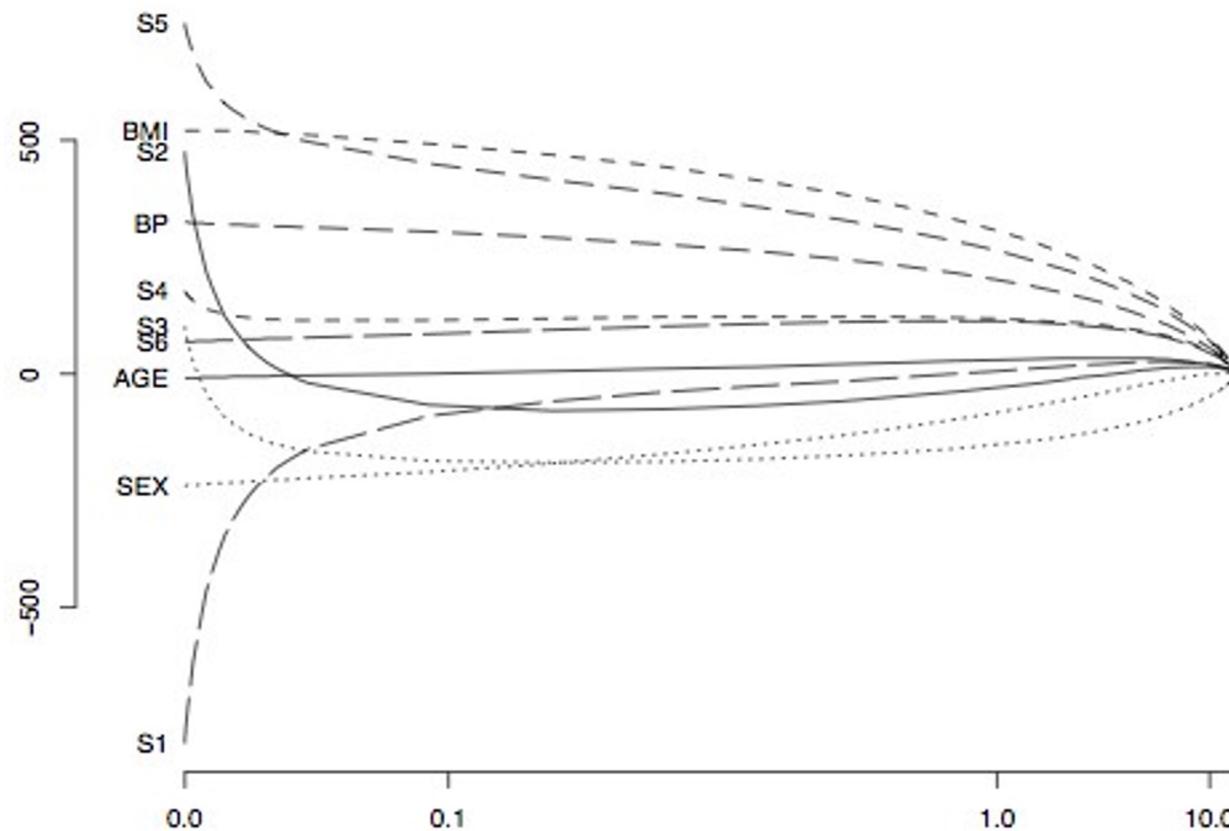
Hồi quy Ridge

- Xét tập dữ liệu Prostate gồm 67 quan sát dùng để học, và 31 quan sát dùng để kiểm thử. Dữ liệu gồm 8 thuộc tính.

w	Least squares	Ridge
0	2.465	2.452
lcavol	0.680	0.420
lweight	0.263	0.238
age	-0.141	-0.046
lbph	0.210	0.162
svi	0.305	0.227
lcp	-0.288	0.000
gleason	-0.021	0.040
pgg45	0.267	0.133
Test RSS	0.521	0.492

Hồi quy Ridge

- $\mathbf{W}^* = (w_0, S1, S2, S3, S4, S5, S6, AGE, SEX, BMI, BP)$ thay đổi khi cho λ thay đổi.



Hồi quy Lasso

- Hồi quy Ridge sử dụng chuẩn L² cho đại lượng hiệu chỉnh:

$$w^* = \arg \min_{\mathbf{w}} \sum_{i=1}^M (y_i - \mathbf{A}_i \mathbf{w})^2, \text{ sao cho } \sum_{j=0}^n w_j^2 \leq t$$

- Thay L² bằng L¹ thì ta sẽ thu được phương pháp LASSO:

$$w^* = \arg \min_{\mathbf{w}} \sum_{i=1}^M (y_i - \mathbf{A}_i \mathbf{w})^2$$

sao cho $\sum_{j=0}^n w_j^2 \leq t$

- Hoặc có thể viết lại:

$$w^* = \arg \min_{\mathbf{w}} \sum_{i=1}^M (y_i - \mathbf{A}_i \mathbf{w})^2 + \lambda \|\mathbf{w}\|_1$$

- Hàm mục tiêu của bài toán là không trơn. Do đó việc giải nó có thể khó hơn hồi quy Ridge.

Hồi quy Lasso: đại lượng hiệu chỉnh

- Các kiểu hiệu chỉnh khác nhau sẽ tạo ra các miền khác nhau cho w .
- LASSO thường tạo ra nghiệm thừa, tức là nhiều thành phần của w có giá trị là 0.
 - Vì thế LASSO thực hiện đồng thời việc hạn chế và lựa chọn đặc trưng

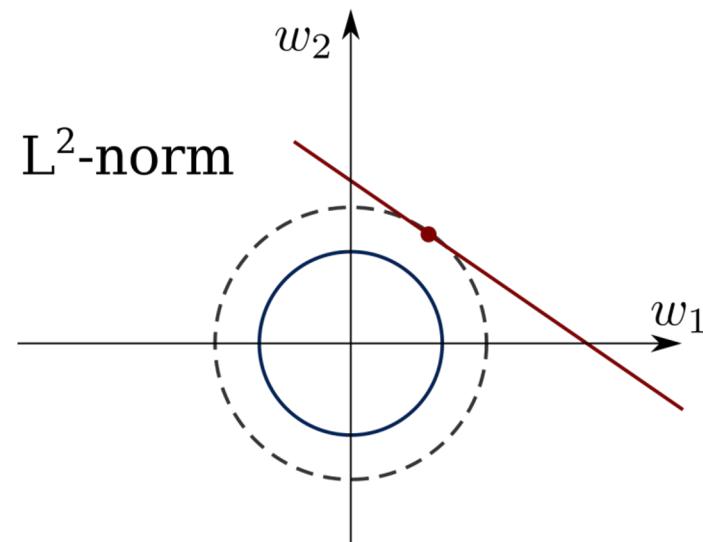
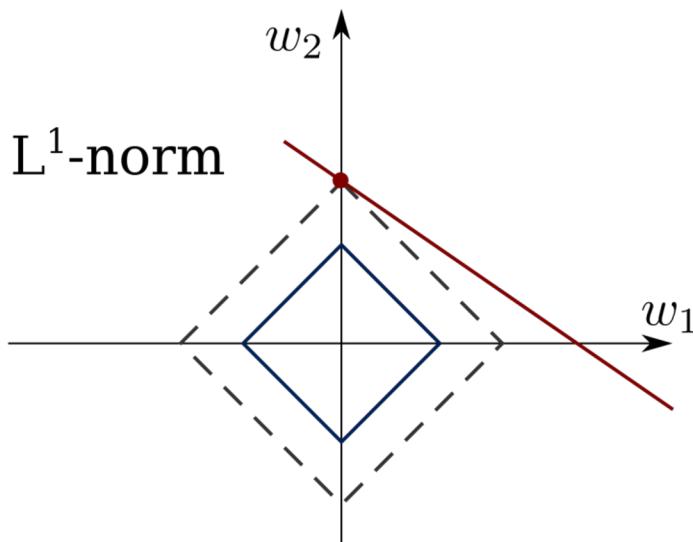


Figure by Nicoguaro - Own work, CC BY 4.0,
<https://commons.wikimedia.org/w/index.php?curid=5825896>

OLS, Ridge, LASSO

- Xét tập dữ liệu Prostate gồm 67 quan sát dùng để học, và 31 quan sát dùng để kiểm thử. Dữ liệu gồm 8 thuộc tính.

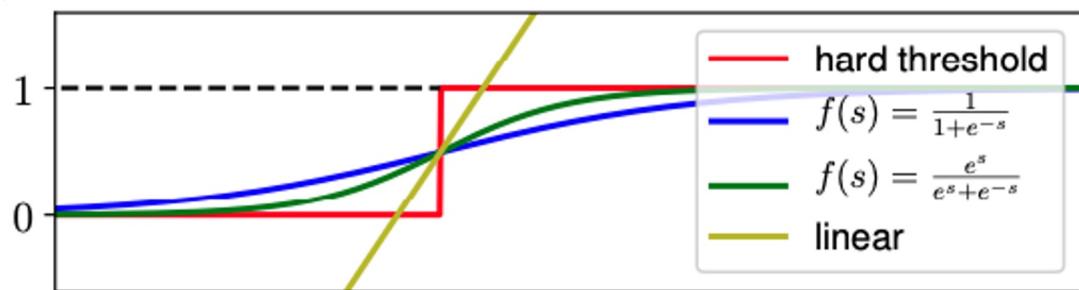
w	Ordinary Least Squares	Ridge	LASSO
0	2.465	2.452	2.468
lcavol	0.680	0.420	0.533
lweight	0.263	0.238	0.169
age	-0.141	-0.046	
lbph	0.210	0.162	0.002
svi	0.305	0.227	0.094
lcp	-0.288	0.000	
gleason	-0.021	0.040	
pgg45	0.267	0.133	
Test RSS	0.521	0.492	0.479

Một số trọng số là 0
→ Chúng có thể không quan trọng

Logistic Regression

- Nếu Linear Regression xác định một giá trị liên tục như nhiệt độ, kích thước thì Logistic regression xác định rằng một việc là **True** hay **False**, **Pass** hay **Fail**, **1** hay **0**

- Sử dụng các hàm kích hoạt đặc biệt



- Tính chất quan trọng của hàm:
 - Là các hàm số liên tục nhận giá trị thực trong khoảng $(0, 1)$.
 - Nếu coi điểm có tung độ 0.5 là ngưỡng, các điểm càng xa ngưỡng về phía bên trái có giá trị càng gần không, còn điểm càng xa ngưỡng về phía bên phải thì càng gần 1 .
 - Có đạo hàm ở mọi nơi, vì vậy có thể được lợi trong việc tối ưu
- Ta có thể thấy đường màu lục và màu lam phù hợp với tính chất trên

Hàm sigmoid

- Trong số các hàm có tính chất trên, có hàm sigmoid được sử dụng nhiều nhất, vì nó chăn trong khoảng (0, 1).

$$f(s) = \frac{1}{1 + e^{-s}} \triangleq \sigma(s)$$

- Ngoài ra, hàm *tanh* cũng hay được sử dụng:

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}.$$

- Hàm số này nhận giá trị trong khoảng (-1, 1)

Tổng kết buổi học

- Các thuật toán học giám sát cơ bản
 - Bài toán phân loại
 - Naive Bayes
 - kNN
 - Decision Tree
 - Random Forest
 - SVM
 - Bài toán hồi quy
 - Linear Regression
 - Ridge Regression
 - Lasso Regression
- Ý tưởng, cách tiếp cận, ứng dụng cho từng dạng đầu ra

THANK YOU !

COLE.VN
Connecting knowledge



www.cole.vn