HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHÓA HỌC

HỌC MÁY CHO AN TOÀN THÔNG TIN

PHẦN 1: CƠ BẢN VỀ HỌC MÁY VÀ HỌC SÂU

Giảng viên: TS. Nguyễn Ngọc Điệp

E-mail: diepnguyenngoc@ptit.edu.vn

Đơn vị: Khoa An toàn thông tin, Học viện Công nghệ BCVT

Nội dung

- Các mô hình học máy truyền thống
  - Các mô hình học máy có giám sát
    - Cây quyết định
    - Naïve Bayes
    - kNN
    - Rừng ngẫu nhiên
    - SVM
    - Logistic Regression
  - Các mô hình học máy không giám sát
    - k-Means
    - DBSCAN

1.2. Các mô hình học máy truyền thống

- Các mô hình học máy có giám sát
  - **Cây quyết định**
  - Naïve Bayes
  - kNN
  - Rừng ngẫu nhiên
  - SVM
  - Logistic Regression

Dữ liệu huấn luyện

Ngày	Trời	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
D1	nắng	nóng	cao	yếu	không
D2	nắng	nóng	cao	mạnh	không
D3	u ám	nóng	cao	yếu	có
D4	mưa	trung bình	cao	yếu	có
D5	mưa	lạnh	bình thường	yếu	có
D6	mưa	lạnh	bình thường	mạnh	không
D7	u ám	lạnh	bình thường	mạnh	có
D8	nắng	trung bình	cao	yếu	không
D9	nắng	lạnh	bình thường	yếu	có
D10	mưa	trung bình	bình thường	yếu	có
D11	nắng	trung bình	bình thường	mạnh	có
D12	u ám	trung bình	cao	mạnh	có
D13	u ám	nóng	bình thường	yếu	có
D14	mưa	trung bình	cao	mạnh	không

thuộc tính

nhãn

mẫu

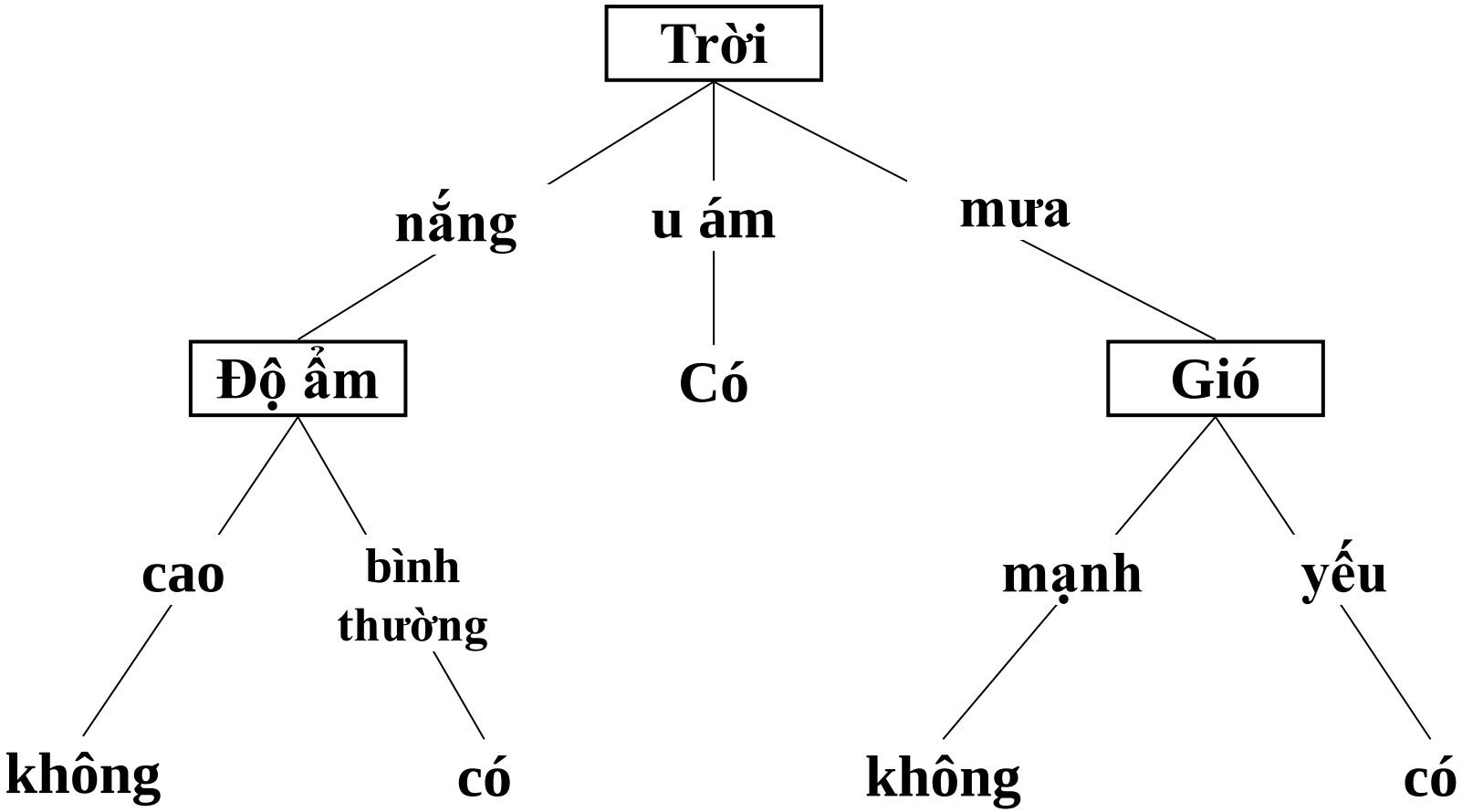
Ngày	Trời	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
D1	nắng	nóng	cao	yếu	không
D2	nắng	nóng	cao	mạnh	không
D3	u ám	nóng	cao	yếu	có
D4	mưa	trung bình	cao	yếu	có
D5	mưa	lạnh	bình thường	yếu	có
D6	mưa	lạnh	bình thường	mạnh	không
D7	u ám	lạnh	bình thường	mạnh	có
D8	nắng	trung bình	cao	yếu	không
D9	nắng	lạnh	bình thường	yếu	có
D10	mưa	trung bình	bình thường	yếu	có
D11	nắng	trung bình	bình thường	mạnh	có
D12	u ám	trung bình	cao	mạnh	có
D13	u ám	nóng	bình thường	yếu	có
D14	mưa	trung bình	cao	mạnh	không

Dữ liệu

- n mẫu huấn luyện, mỗi mẫu là một cặp $< \boldsymbol{x}, y >$
  - $\boldsymbol{x}$ là vector các thuộc tính
  - y là nhãn phân loại, $y \in C$ (tập các nhãn)

- Ví dụ mẫu D4
  - $\boldsymbol{x} = (mưa,\ trung\ bình, cao, yếu)$
  - $y = có$

Ví dụ cây quyết định



Cây quyết định là gì?

- Là mô hình phân loại có dạng cây
  - Mỗi nút trung gian (không phải lá) ứng với một phép kiểm tra thuộc tính, mỗi nhánh của nút ứng với một giá trị của thuộc tính tại nút đó
  - Mỗi nút lá ứng với một nhãn phân loại

- Quá trình phân loại thực hiện như sau
  - Mẫu phân loại đi từ gốc cây xuống dưới
  - Tại mỗi nút trung gian, thuộc tính tương ứng với nút được kiểm tra, tùy giá trị thuộc tính, mẫu được chuyển xuống nhánh tương ứng
  - Khi tới nút lá, mẫu được nhận nhãn phân loại của nút

Biểu diễn dưới dạng quy tắc

- Cây quyết định có thể biểu diễn tương đương dưới dạng các quy tắc logic
- Mỗi cây là tuyển của các quy tắc, mỗi quy tắc bao gồm các phép hội
- Ví dụ

(Trời = nắng ∧ Độ ẩm = bình_thường)
∨ (Trời = u_ám)
∨ (Trời = mưa ∧ Gió = yếu)

Học cây quyết định

- Cây quyết định được học (xây dựng) từ dữ liệu huấn luyện
- Với mỗi bộ dữ liệu có thể xây dựng nhiều cây quyết định
  - Chọn cây nào?
- Quá trình học là quá trình tìm kiếm cây quyết định phù hợp với dữ liệu huấn luyện
  - Cho phép phân loại đúng dữ liệu huấn luyện

Thuật toán ID3

- Xây dựng lần lượt các nút của cây bắt đầu từ gốc

- Thuật toán
  - **Khởi đầu**: nút hiện thời là nút gốc chứa toàn bộ tập dữ liệu huấn luyện
  - Tại nút hiện thời n, lựa chọn thuộc tính
    - Chưa được sử dụng ở nút tổ tiên
    - Cho phép phân chia tập dữ liệu hiện thời thành các tập con **một cách tốt nhất**
    - Với mỗi giá trị thuộc tính được chọn thêm một nút con bên dưới
    - Chia các ví dụ ở nút hiên thời về các nút con theo giá trị thuộc tính được chọn
  - **Lặp** (đệ quy) cho tới khi
    - Tất cả các thuộc tính đã được sử dụng ở các nút phía trên, hoặc
    - Tất cả ví dụ tại nút hiện thời có cùng nhãn phân loại
    - Nhãn của nút được lấy theo đa số nhãn của ví dụ tại nút hiện thời

Lưa chọn thuộc tính tại mỗi nút thế nào?

Tiêu chuẩn chọn thuộc tính của ID3

- Tại mỗi nút n
  - Tập (con) dữ liệu ứng với nút đó
  - Cần lựa chọn thuộc tính cho phép phân chia tập dữ liệu tốt nhất
- Tiêu chuẩn:
  - Dữ liệu sau khi phân chia càng đồng nhất càng tốt
  - Đo bằng độ tăng thông tin (Information Gain - IG)
  - **Chọn thuộc tính có độ tăng thông tin lớn nhất**
  - IG dựa trên entropy của tập (con) dữ liệu

Entropy

- Trường hợp tập dữ liệu S có 2 loại nhãn: đúng (+) hoặc sai (-)

$$\boldsymbol{Entropy(S) = -p_+ log_2 p_+ - p_- log_2 p_-}$$

p_+: % số mẫu đúng, p_-: % số mẫu sai

- Trường hợp tổng quát: có C loại nhãn

$$\boldsymbol{Entropy(S) = \sum_{i=1}^{C} -p_i log_2 p_i}$$

p_i : % ví dụ của S thuộc loại i

- Ví dụ

$$\boldsymbol{Entropy([9^+, 5^-]) = -(9/14) log_2(9/14) - (5/14) log_2(5/14) = 0.94}$$

Độ tăng thông tin IG

Với tập (con) mẫu S và thuộc tính A

$$IG(S,A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Trong đó:

values (*A*): tập các giá trị của A

S_v là tập con của S bao gồm các mẫu có giá trị của A bằng v

$|S|$ số phần tử của S

Ví dụ tính IG

❖ Tính $IG(S, Gió)$

$$values(Gió) = \{yếu, mạnh\}$$

$$S = [9+, 5-], H(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$$

$$S_{yếu} = [6+, 2-], \mathrm{H}(S_{yếu}) = -\frac{6}{8} log_2 \frac{6}{8} - \frac{2}{8} log_2 \frac{2}{8} = 0.811$$

$$S_{mạnh} = [3+, 3-], \mathrm{H}(S_{mạnh}) = -\frac{3}{6} log_2 \frac{3}{6} - \frac{3}{6} log_2 \frac{3}{6} = 1$$

$$IG(S, Gió) = H(S) - \frac{8}{14}\mathrm{H}(S_{yếu}) - \frac{6}{14}\mathrm{H}(S_{mạnh})$$

$$= 0.94 - \frac{8}{14} 0.811 - \frac{6}{14} 1$$

$$= 0.048$$

Các đặc điểm của ID3

- ID3 là thuật toán tìm kiếm cây quyết định phù hợp với dữ liệu huấn luyện
- Tìm kiếm theo kiểu tham lam, bắt đầu từ cây rỗng
- Hàm đánh giá là độ tăng thông tin
- ID3 có khuynh hướng (bias) lựa chọn cây đơn giản
  - Ít nút
  - Các thuộc tính có độ tăng thông tin lớn nằm gần gốc

Training error và Test error (1/2)

- Training error (lỗi huấn luyện)
  - Là lỗi đo được trên tập dữ liệu huấn luyện
  - Thường đo bằng sự sai khác giữa giá trị tính toán của mô hình và giá trị thực của dữ liệu huấn luyện
  - Trong quá trình học ta cố gắng làm giảm tới mức tối thiểu lỗi huấn luyện
- Test error (lỗi kiểm tra)
  - Là lỗi đo được trên tập dữ liệu kiểm tra
  - Là cái ta thực sự quan tâm

Làm sao ta có thể tác động tới hiệu quả của mô hình trên tập dữ liệu kiểm tra khi ta chỉ quan sát được tập dữ liệu huấn luyện?

Chống quá vừa bằng cách tỉa cây

- Chia dữ liệu thành hai phần
  - Huấn luyện
  - Kiểm tra
- Tạo cây đủ lớn trên dữ liệu huấn luyện
- Tính độ chính xác của cây trên tập kiểm tra
- Loại bỏ cây con sao cho kết quả trên dữ liệu kiểm tra được cải thiện nhất
- Lặp lại cho đến khi không còn cải thiện được kết quả nữa

Chống quá vừa dữ liệu bằng cách tỉa luật (C4.5)

- Biến đổi cây thành các luật
- Tỉa mỗi luật độc lập với các luật khác
  - Bỏ một số phần trong vế trái của luật
- Sắp xếp các luật sau khi tỉa theo mức độ chính xác của luật

Sử dụng thuộc tính có giá trị liên tục

- Tạo ra những thuộc tính rời rạc mới
- Ví dụ, với thuộc tính liên lục A, tạo ra thuộc tính rời rạc Ac như sau
  - $Ac = true$ nếu $A > c$
  - $Ac = false$ nếu $A \leq c$
- Xác định ngưỡng c thế nào?
  - Thường chọn sao cho Ac đem lại độ tăng thông tin lớn nhất
- Có thể chia thành nhiều khoảng với nhiều ngưỡng

Các độ đo khác

- Độ đo Information Gain (IG) ưu tiên thuộc tính có nhiều giá trị, ví dụ, thuộc tính ngày sẽ có độ tăng thông tin cao nhất

- Thông tin chia

$$SplitInformation(S,A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- Tiêu chuẩn đánh giá thuộc tính

$$GainRatio = \frac{InformationGain(S,A)}{SplitInformation(S,A)}$$

1.2. Các mô hình học máy truyền thống

- Các mô hình học máy có giám sát
  - Cây quyết định
  - **Naïve Bayes**
  - kNN
  - Rừng ngẫu nhiên
  - SVM
  - Logistic Regression

Phương pháp phân loại Bayes (1/2)

- Trong giai đoạn huấn luyện ta có một tập mẫu, mỗi mẫu được cho bởi cặp $< \boldsymbol{x}_i, y_i >$, trong đó
  - $\boldsymbol{x}_i$ là vector đặc trưng (thuộc tính)
  - y_i là nhãn phân loại,$y_i \in C$ (C là tập các nhãn)
- Sau khi huấn luyện xong, bộ phân loại cần dự đoán nhãn y cho mẫu mới $\boldsymbol{x} =< x_1, x_2, \dots, x_n >$

$$y = argmax_{c_j \in C} P(c_j | x_1, x_2, \dots, x_n)$$

- Sử dụng quy tắc Bayes

$$y = argmax_{c_j \in C} \frac{P(x_1, x_2, \dots, x_n | c_j) P(c_j)}{P(x_1, x_2, \dots, x_n)}$$

$$= argmax_{c_j \in C} P(x_1, x_2, \dots, x_n | c_j) P(c_j)$$

Phương pháp phân loại Bayes (2/2)

Tần xuất quan sát thấy nhãn c_j trên tập dữ liệu D: $\frac{count(c_j)}{|D|}$

$$y = argmax_{c_j \in C} P(x_1, x_2, \dots, x_n | c_j) P(c_j)$$

Sử dụng giả thiết về tính độc lập (**Đơn giản!!!**)

$$P(x_1, x_2, \dots, x_n | c_j) = P(x_1 | c_j) P(x_2 | c_j) \dots P(x_n | c_j)$$

Số lần xuất hiện x_i cùng với c_j chia cho số lần xuất hiện c_j: $\frac{count(x_i, c_j)}{count(c_j)}$

Ví dụ

- Xác định nhãn phân loại cho mẫu sau

$$< Trời = nắng, Nhiệt\ độ = trung\ bình, Độ\ ẩm = cao, Gió = mạnh >$$

$$y = argmax_{c \in \{có, không\}} P(Trời = nắng|c) P(Nhiệt\ độ = trung\ bình|c)$$
$$P(Độ\ ẩm = cao|c) P(Gió = mạnh|c) P(c)$$

1.2. Các mô hình học máy truyền thống

- Các mô hình học máy có giám sát
  - Cây quyết định
  - Naïve Bayes
  - **kNN**
  - Rừng ngẫu nhiên
  - SVM
  - Logistic Regression

Nguyên tắc chung

- kNN là một dạng học theo ví dụ (Instance-based learning)
- Không xây dựng mô hình
- Chỉ lưu lại các mẫu huấn luyện
- Xác định nhãn cho mẫu mới dựa trên những mẫu giống mẫu mới nhất
- Gọi là học lười (lazy learning)

Thuật toán kNN - k hàng xóm gần nhất

- k-nearest neighbors (k-NN)
- Chọn k mẫu giống mẫu cần phân loại nhất, gọi là k hàng xóm
- Gán nhãn phân loại cho mẫu chỉ sử dụng thông tin của k hàng xóm này
  - Ví dụ lấy theo đa số trong số k hàng xóm
- Chọn hàng xóm thế nào?

Tính khoảng cách

- Giả sử mẫu x có giá trị thuộc tính là $< a_1(x), a_2(x), \ldots, an(x) >$, thuộc tính là số thực
- Khoảng cách giữa hai mẫu x_i và x_j là khoảng cách Euclidean

$$d(x_i, x_j) = \sqrt{\sum_{l=1}^{n} (a_l(x_i) - a_l(x_j))^2}$$

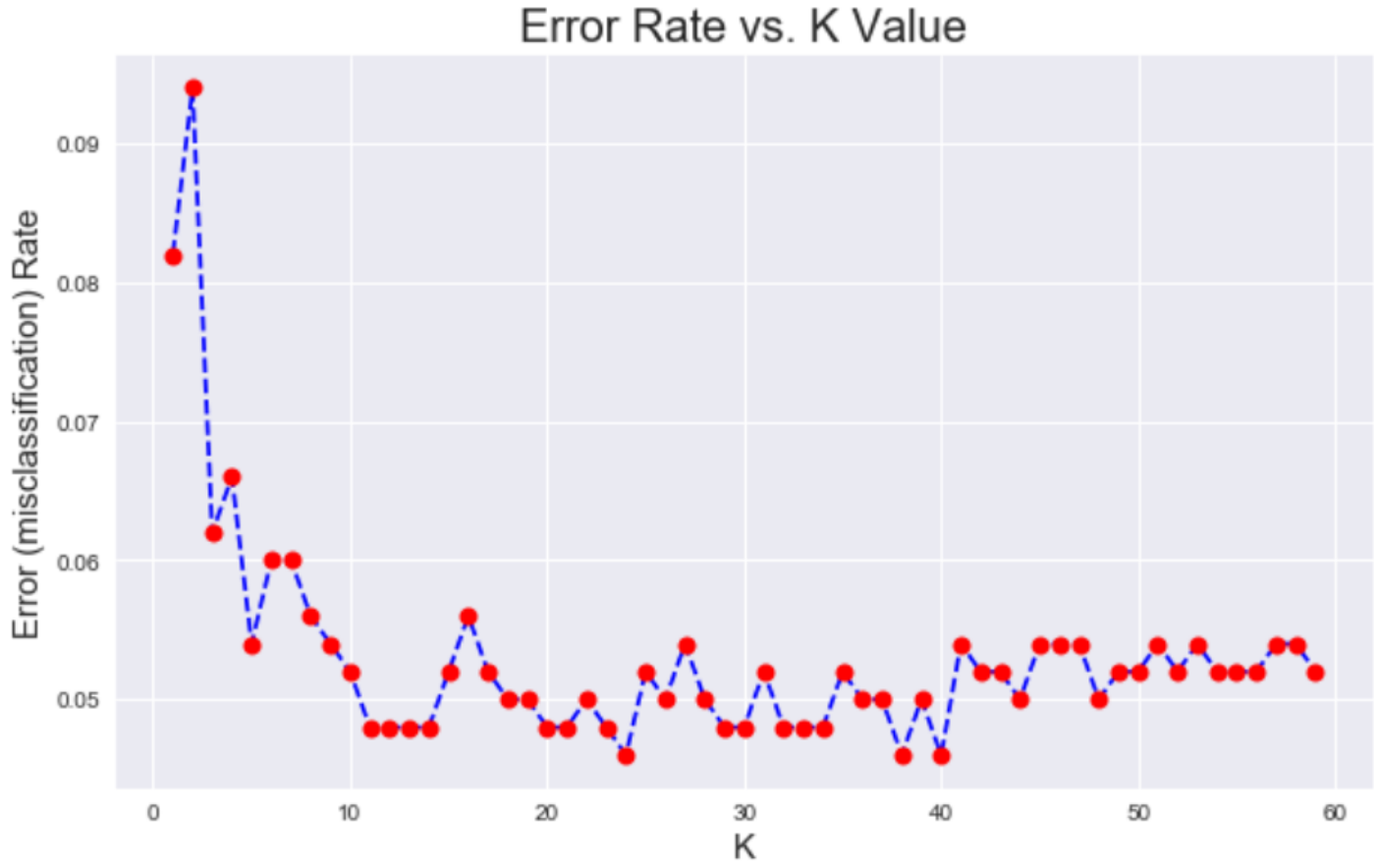
Câu hỏi

- Câu hỏi 1
  - Độ đo khoảng cách khác có được sử dụng không? Khi nào?

- Câu hỏi 2:
  - Làm sao biết k thế nào là phù hợp?

Câu hỏi

- Câu hỏi 2:
  - Làm sao biết k thế nào là phù hợp? Dựa vào pp khuỷu tay



Thuật toán k-NN

Giai đoạn học (huấn luyện)

Lưu các mẫu huấn luyện có dạng $<x, f(x)>$ vào cơ sở dữ liệu

Giai đoạn phân loại

Đầu vào: tham số k

Với mẫu x cần phân loại:

1. Tính khoảng cách $d(x, x_i)$ từ x tới tất cả mẫu x_i trong cơ sở dữ liệu
2. Tìm k mẫu có $d(x, x_i)$ nhỏ nhất, giả sử k mẫu đó là $x_1, x_2, \ldots, x_k$.
3. Xác định nhãn phân loại $f'(x)$ là nhãn chiếm đa số trong tập $\{ x_1, x_2, \ldots, x_k. \}$

k-NN trong scikit learn

```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *,
weights='uniform', algorithm='auto', leaf_size=30, p=2,
metric='minkowski', metric_params=None, n_jobs=None)
```
[source]

```python
from sklearn.neighbors import KNeighborsClassifier

# Khởi tạo bộ phân loại KNN với k láng giềng gần nhất
knn = KNeighborsClassifier(n_neighbors=5)

# Huấn luyện mô hình với dữ liệu huấn luyện
knn.fit(X_train, y_train)

# Dự đoán nhãn cho dữ liệu mới
y_pred = knn.predict(X_test)
```

Hiệu quả của k-NN

- Phụ thuộc vào k
- Cần chuẩn hóa dữ liệu để đảm bảo các thuộc tính có thang đo tương đương, giúp các độ đo khoảng cách hoạt động hiệu quả hơn
- Nên xem xét các độ đo khoảng cách khác nhau, tuy vào dữ liệu

1.2. Các mô hình học máy truyền thống

- Các mô hình học máy có giám sát
  - Cây quyết định
  - Naïve Bayes
  - kNN
  - **Rừng ngẫu nhiên**
  - SVM
  - Logistic Regression

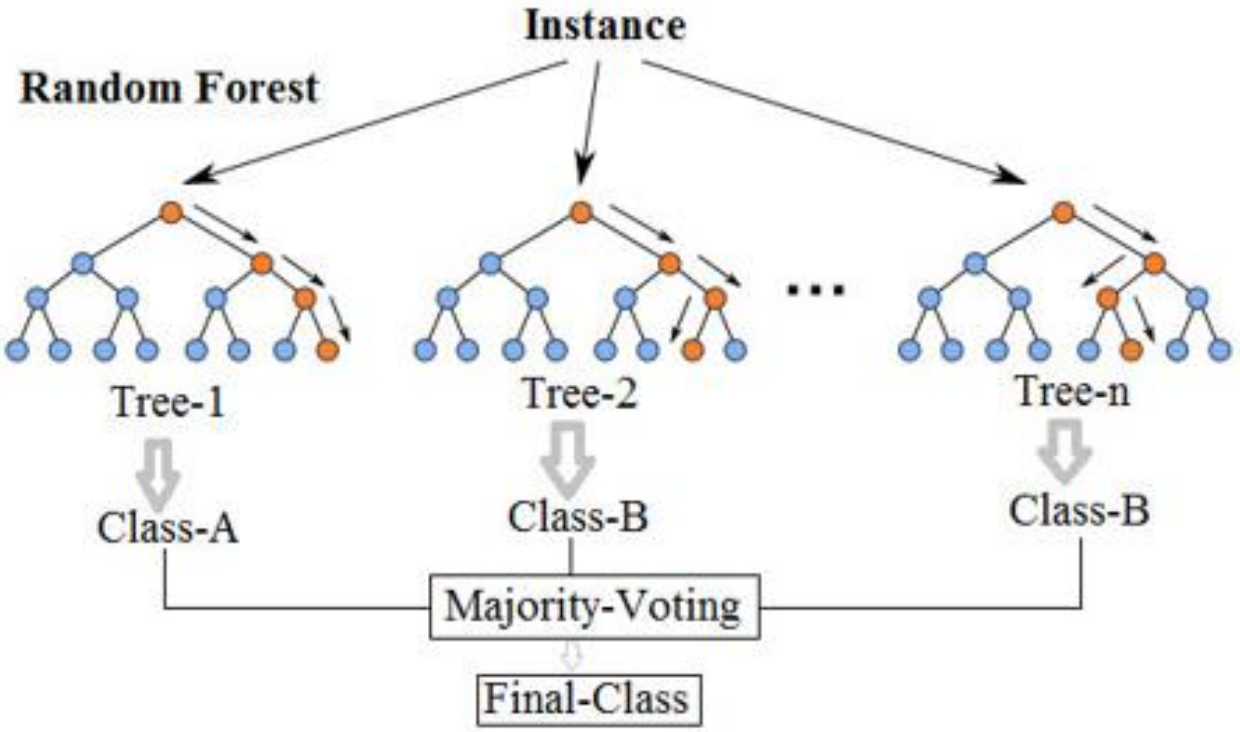
Thuật toán Rừng ngẫu nhiên

❖ Hạn chế của Cây quyết định

- Tính không ổn định do cực kỳ nhạy cảm với các nhiễu nhỏ trong dữ liệu ➔ một thay đổi nhỏ trong các ví dụ đào tạo có thể dẫn đến thay đổi lớn trong cấu trúc của Decision Tree (do cố gắng giảm thiểu entropy) ➔ ***Vấn đề Phương sai cao***
- ***Vấn đề quá khớp***
- Cách xử lý:
  - Cắt tỉa cây để xử lý quá khớp dữ liệu
  - Đưa thêm lớp ngẫu nhiên vào quá trình huấn luyện ➔ tạo ra Các bộ cây quyết đinh - các phiên bản hơi khác nhau của tập dữ liệu ➔ các dự đoán kết hợp của chúng không bị ảnh hưởng nhiều do phương sai cao
    - ➔ Đây chính là Rừng ngẫu nhiên

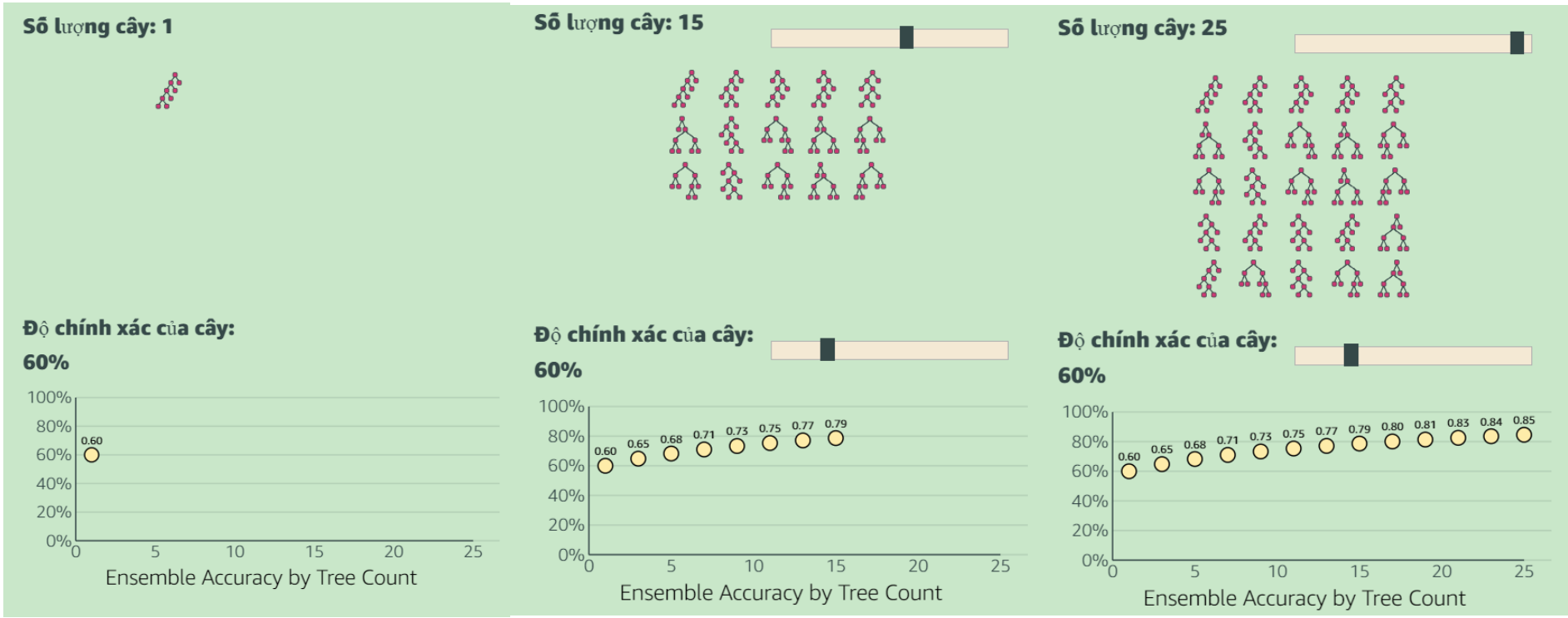
Thuật toán Rừng ngẫu nhiên

❖ Rừng ngẫu nhiên



Thuật toán Rừng ngẫu nhiên

❖ Rừng ngẫu nhiên



https://mlu-explain.github.io/random-forest/

Thuật toán Rừng ngẫu nhiên

❖ Rừng ngẫu nhiên:

- là một thuật toán học máy giám sát được sử dụng cho các nhiệm vụ phân loại và dự đoán. Nó hoạt động bằng cách kết hợp nhiều cây quyết định (decision tree) được xây dựng ngẫu nhiên để tạo ra một mô hình dự đoán mạnh mẽ và chính xác hơn so với các cây quyết định riêng lẻ.

❖ Ưu điểm:

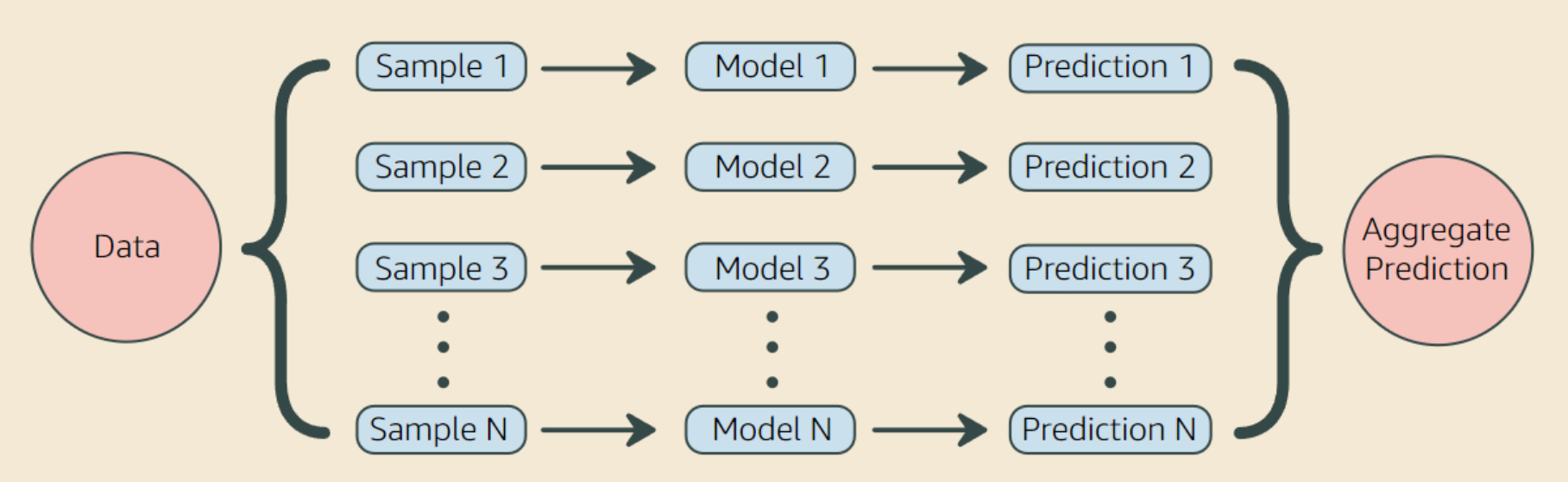
- Độ chính xác cao
- Khả năng chống nhiễu tốt
- Dễ sử dụng
- Có thể giải thích được

❖ Nhược điểm:

- Tốn thời gian đào tạo

Thuật toán Rừng ngẫu nhiên

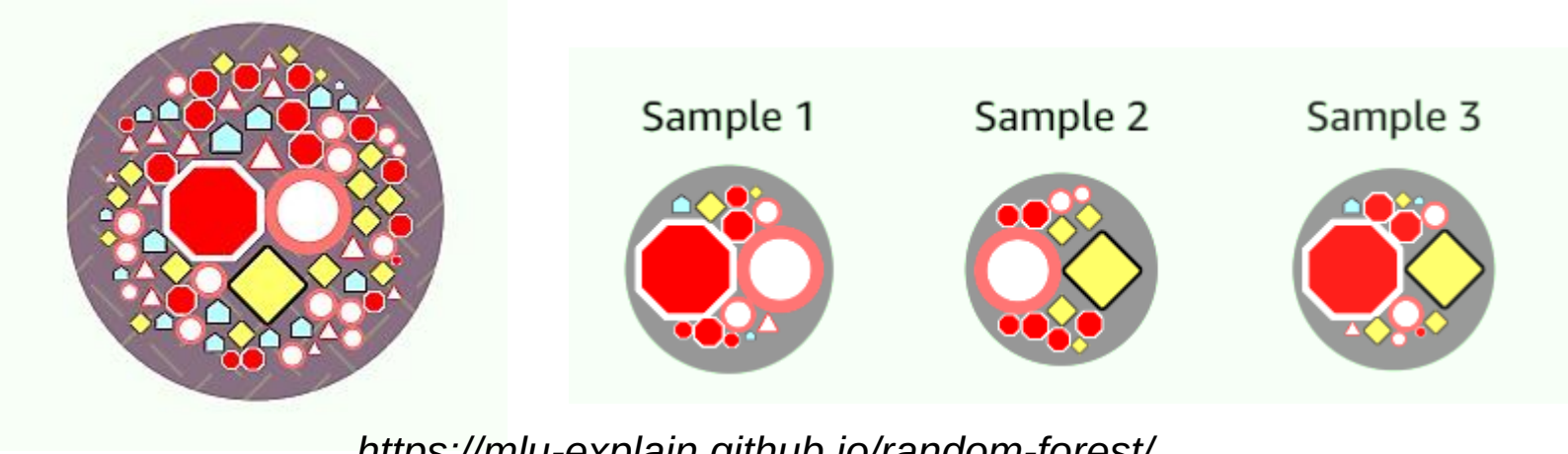
- Dựa trên học kết hợp
  - Mỗi mô hình là một cây quyết định



Thuật toán Rừng ngẫu nhiên

❖ Cách thức hoạt động:

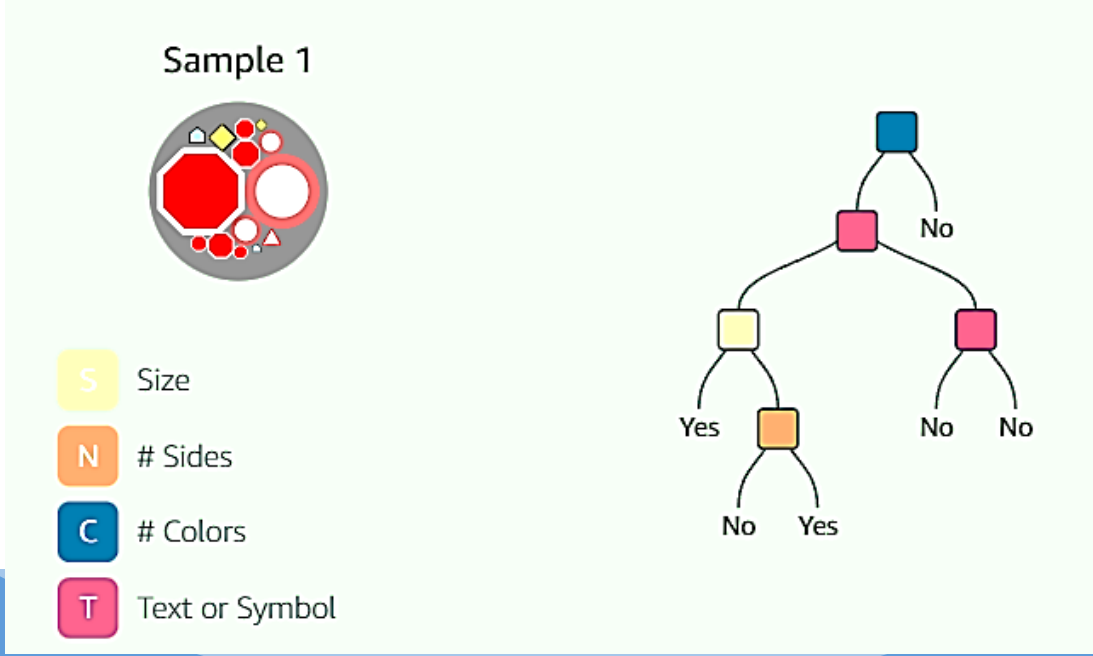
- Tạo tập dữ liệu:
  - Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kĩ thuật Bootstrapping (lấy mẫu ngẫu nhiên với bù đắp)
  - Khi lấy mẫu được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau.



https://mlu-explain.github.io/random-forest/

Thuật toán Rừng ngẫu nhiên

- Cách thức hoạt động:
  - Xây dựng cây quyết định:
    - Cho mỗi tập con, xây dựng một cây quyết định sử dụng một tập con ngẫu nhiên của các biến đặc trưng
    - Ví dụ: mỗi mẫu có 5 đặc trưng. Mẫu 1 chỉ chọn 4 đặc trưng để xây dựng cây.



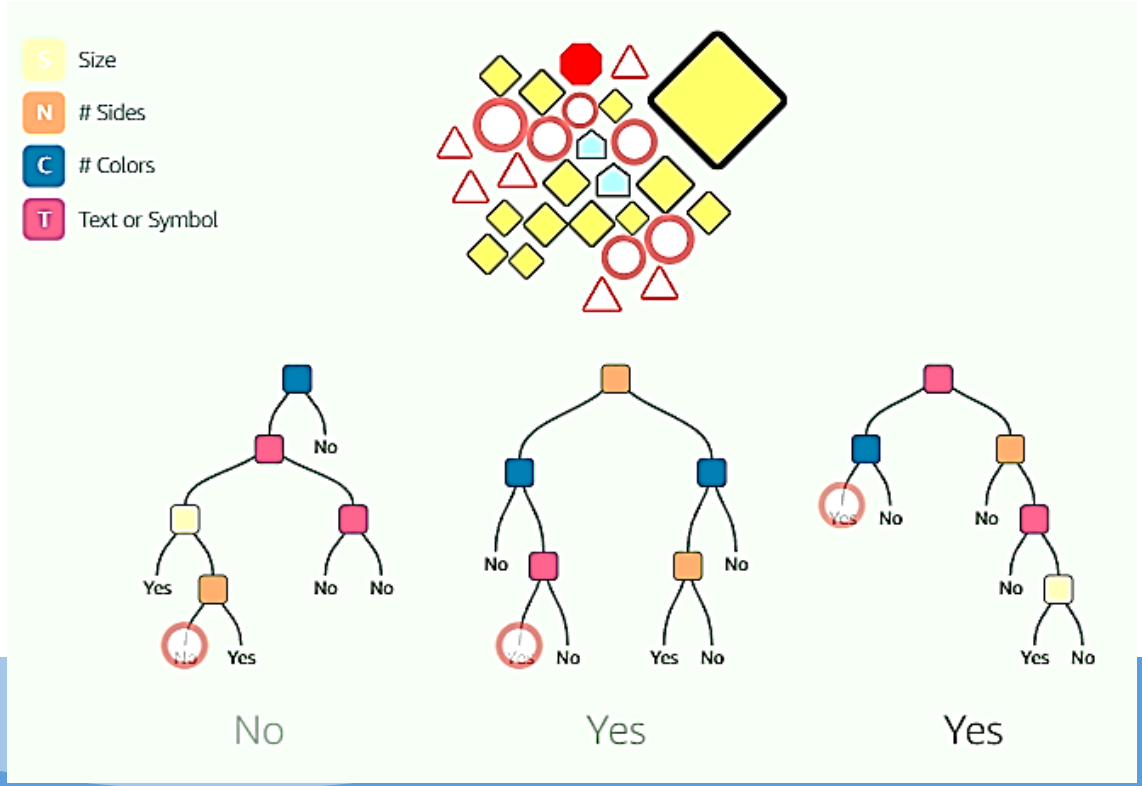
Thuật toán Rừng ngẫu nhiên

❖ Cách thức hoạt động:

- Dự đoán:
  - Khi có một điểm dữ liệu mới, mỗi cây quyết định trong rừng sẽ đưa ra dự đoán cho điểm dữ liệu đó. Dự đoán cuối cùng cho điểm dữ liệu là dự đoán của tất cả các cây quyết định.

Một biển báo tròn nhỏ có thể là biển báo giao thông không?
Hãy hỏi ba cái cây!

Theo đa số phiếu bầu, dự đoán là "Đồng ý"!



Rừng ngẫu nhiên trong scikit learn

```
from sklearn.ensemble import RandomForestClassifier

# Khởi tạo mô hình Random Forest với 100 cây quyết định
rf = RandomForestClassifier(n_estimators=100)

# Huấn luyện mô hình với dữ liệu huấn luyện
rf.fit(X_train, y_train)

# Dự đoán nhãn cho dữ liệu mới
y_pred = rf.predict(X_test)
```

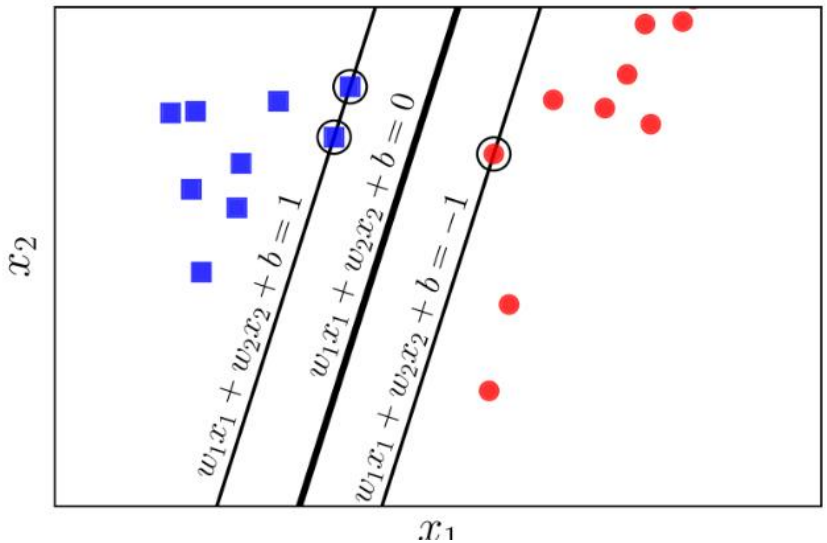
Ứng dụng trong giảm chiều

- Random Forest (RF) có thể được ứng dụng hiệu quả cho việc giảm chiều trong học máy, mang lại một số lợi ích như sau:
  - RF có thể xác định tầm quan trọng của từng đặc trưng/biến (thuộc tính) trong việc dự đoán. Dựa trên tầm quan trọng này, ta có thể loại bỏ các biến có độ ảnh hưởng thấp (ít liên quan đến nhãn dự đoán), giúp giảm thiểu kích thước dữ liệu và tăng hiệu quả của mô hình học máy.
  - RF có thể lựa chọn tập con đặc trưng phù hợp nhất cho bài toán.Thay vì sử dụng tất cả các đặc trưng ban đầu, RF có thể tự động chọn ra một tập con có khả năng dự đoán tốt nhất, giúp giảm nhiễu và tăng hiệu suất của mô hình.

1.2. Các mô hình học máy truyền thống

❖ Các mô hình học máy có giám sát

- Cây quyết định
- Naïve Bayes
- kNN
- Rừng ngẫu nhiên
- **SVM**
- Logistic Regression

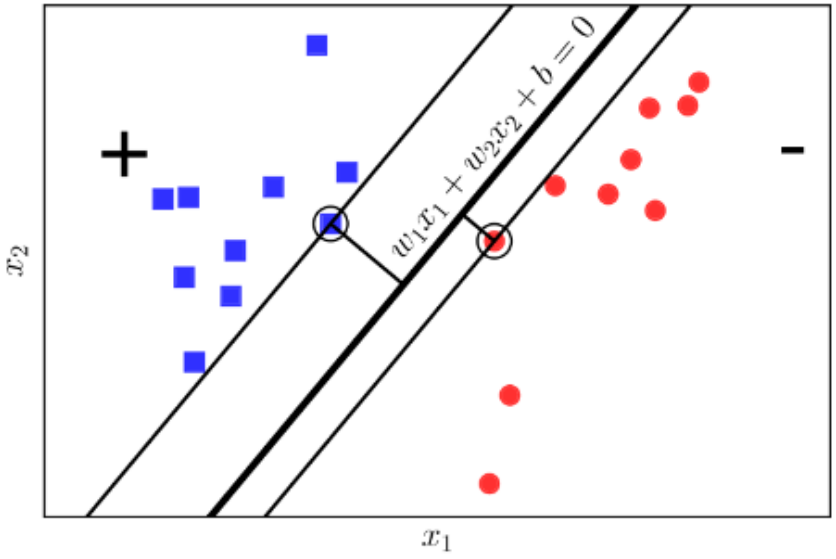


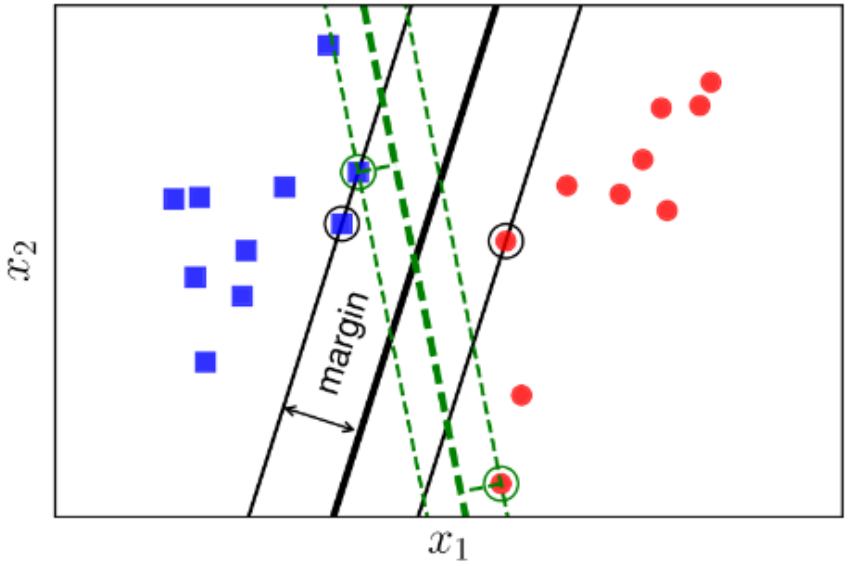
SVM (Support Vector Machine):

- SVM:
  - là một thuật toán học máy có giám sát được sử dụng cho các nhiệm vụ phân loại và hồi quy. Nó hoạt động bằng cách tìm kiếm một siêu phẳng (hyperplane) tối ưu trong không gian đa chiều nhằm phân chia các điểm dữ liệu thuộc các lớp khác nhau một cách hiệu quả nhất.
- Ưu điểm:
  - Độ chính xác cao
  - Khả năng chống nhiễu tốt
  - Có thể giải thích được
  - Hiệu quả: với tập dữ liệu có kích thước lớn.
- Nhược điểm:
  - Khó lựa chọn tham số
  - Bị ảnh hưởng bởi dữ liệu ngoại lệ

SVM (Support Vector Machine):

- Cách thức hoạt động: (tham khảo machinelearningcoban.com)
  - Biểu diễn dữ liệu
  - Tìm siêu phẳng: tìm kiếm một siêu phẳng trong không gian đa chiều có thể phân chia các điểm dữ liệu thuộc các lớp khác nhau một cách tối đa.
  - Phân loại



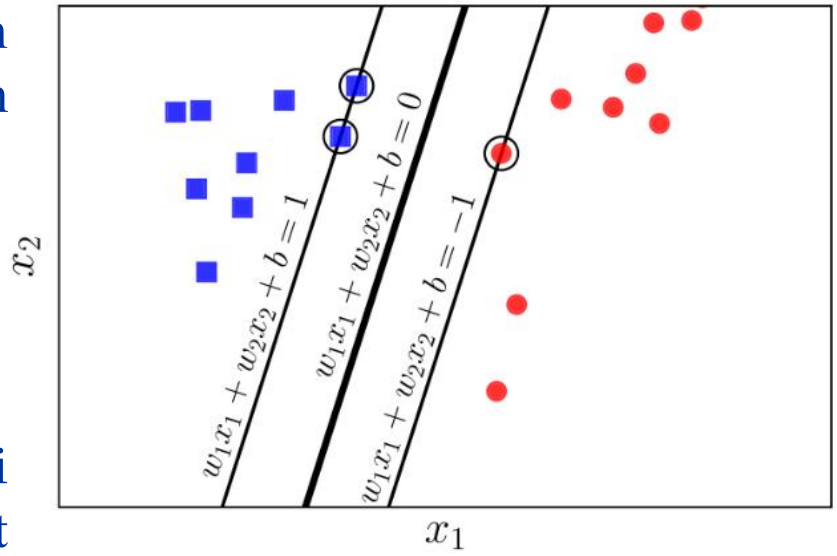


SVM (Support Vector Machine):

❖ Tìm siêu phẳng:

- margin được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó (bất kể điểm nào trong hai lớp)

$$\text{margin} = \min_n \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{||\mathbf{w}||_2}$$



- Bài toán tối ưu trong SVM chính là bài toán tìm **w** và b sao cho margin này đạt giá trị lớn nhất

$$(\mathbf{w}, b) = \arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||_2^2$$

$$\text{subject to:} \quad 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b) \leq 0, \forall n = 1, 2, \ldots, N$$

SVM (Support Vector Machine):

❖ Tìm siêu phẳng:

- Bài toán tối ưu trong SVM là một bài toán lồi với hàm mục tiêu là strictly convex, nghiệm của bài toán này là duy nhất. Hơn nữa, bài toán tối ưu đó là một bài toán Quy hoạch toàn phương (Quadratic Programming - QP).

- Mặc dù có thể trực tiếp giải SVM qua bài toán tối ưu gốc này, thông thường người ta thường giải bài toán đối ngẫu. Bài toán đối ngẫu cũng là một QP nhưng nghiệm là thưa (sparse) nên có những phương pháp giải hiệu quả hơn.

SVM trong scikit learn

- kernel (mặc định: 'rbf'):
  - Xác định loại hàm kernel được sử dụng để tính toán độ tương đồng giữa các điểm dữ liệu.
  - Các tùy chọn phổ biến bao gồm:
    - 'linear': Hàm kernel tuyến tính, phù hợp cho dữ liệu có mối quan hệ tuyến tính.
    - 'rbf': Hàm kernel Radial Basis Function (RBF), hiệu quả cho dữ liệu phi tuyến tính.
    - 'poly': Hàm kernel đa thức, có thể linh hoạt hơn nhưng cũng dễ bị quá phụ thuộc.
    - Lựa chọn kernel phù hợp phụ thuộc vào đặc điểm dữ liệu và bài toán cụ thể.
- C, gamma: các tham số điều chỉnh
- class_weight (mặc định: 'balanced'):
  - Xác định trọng số cho các lớp khác nhau.
  - Có thể hữu ích khi tập dữ liệu không cân bằng, với số lượng mẫu của một lớp ít hơn nhiều so với lớp khác.

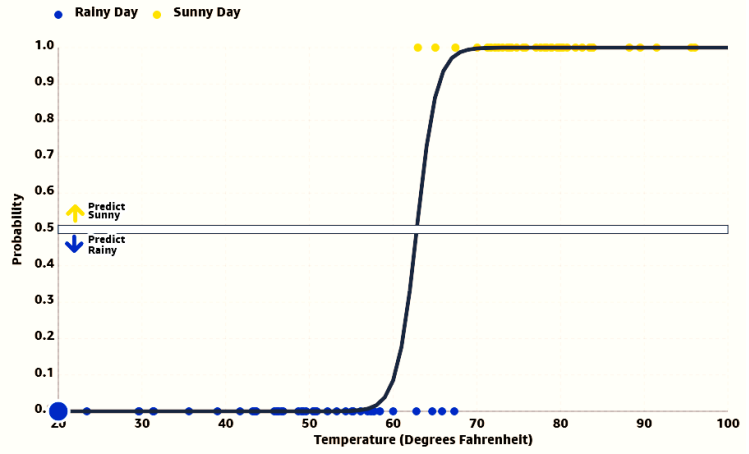
SVM trong scikit learn

- SVC dựa trên libsvm
- Nếu không chọn tham số, nhân phù hợp thì kết quả có thể kém.

```
from sklearn.svm import SVC

# Create an SVC classifier
clf = SVC()

# Fit the model to your training data
clf.fit(X_train, y_train)

# Make predictions on new data
y_pred = clf.predict(X_test)
```

1.2. Các mô hình học máy truyền thống

- Các mô hình học máy có giám sát
  - Cây quyết định
  - Naïve Bayes
  - kNN
  - Rừng ngẫu nhiên
  - SVM
  - **Logistic Regression (Hồi qui Logistic)**



Hồi quy Logistic

- Hồi quy Logistic (Logistic Regression) là một thuật toán học máy có giám sát được sử dụng cho các nhiệm vụ phân loại nhị phân. Nó mô hình hóa xác suất của một biến mục tiêu nhị phân (ví dụ: 0 hoặc 1, có hoặc không, v.v.) dựa trên một tập hợp các biến độc lập.
- Ưu điểm:
  - Dễ hiểu và triển khai
  - Hiệu quả
  - Có thể giải thích được
  - Chống nhiễu tốt
- Nhược điểm:
  - Giả định tuyến tính
  - Có thể gặp khó khăn với dữ liệu không cân bằng

Hồi quy Logistic

- Ứng dụng:
  - Thường sử dụng để phân lớp nhị phân, dù có thể mở rộng cho nhiều lớp
- Ý nghĩa:
  - Các mô hình phân loại đều tìm cách xác định đường biên phân chia tốt nhất các nhóm giữa liệu.
  - Hồi quy Logistic cũng tìm kiếm một đường biên phân chia như vậy để giải quyết bài toán phân loại nhị phân giữa hai nhóm 0 và 1

Hồi quy Logistic

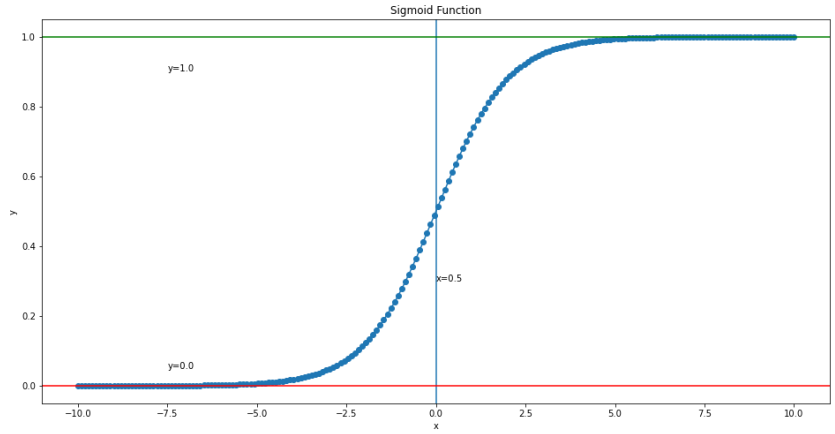
❖ Mô tả:

- Cho đầu vào $X=(x_1,x_2,...,x_k)$
- Đầu ra y (có giá trị 0 hoặc 1) có kết quả cụ thể nào đó với xác suất:

$$P(y=1|X) = \text{sigmoid}(z) = \frac{1}{1+e^{-z}}$$

- Với

$$z = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + ... + \hat{\beta}_k x_k$$

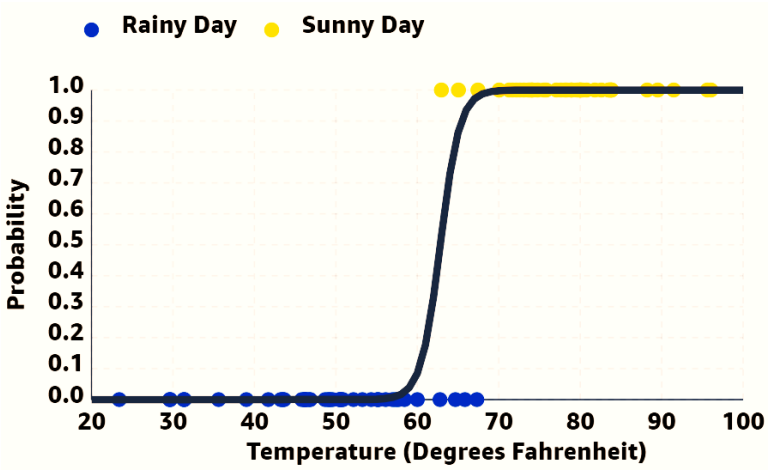


Hàm sigmoid có hình dạng là một đường cong chữ S và đơn điệu tăng, còn gọi là hàm Logistic - đại diện cho hồi qui Logistic .
Giá trị của hàm Sigmoid nằm trong khoảng [0,1]

Hồi quy Logistic

❖ Ví dụ:

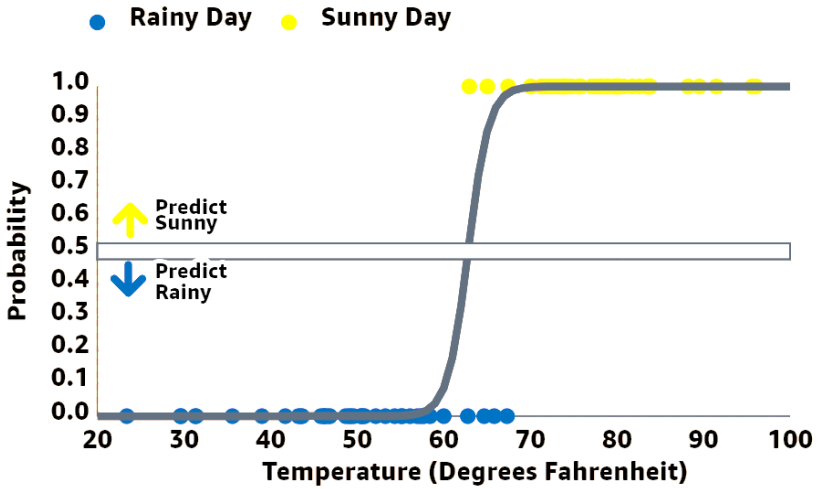
- Mục tiêu: dự đoán xem trời sẽ nắng hay mưa
- Bài toán phân lớp:
  - hai lớp: Ngày mưa và Ngày nắng.
  - Gán 0 cho Ngày mưa và 1 cho Ngày nắng.
  - Đặc trưng là giá trị liên tục: nhiệt độ, theo độ F.
  - Đối với mỗi ngày, vẽ biểu đồ giá trị này cùng với nhiệt độ tương ứng.

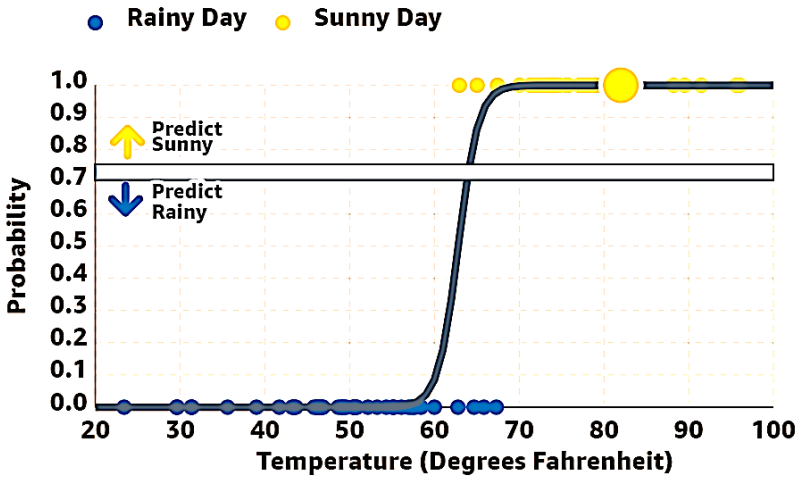


Hồi quy Logistic

❖ Ví dụ:

- Áp dụng hàm Logistic:
  - Giá trị của hàm này có thể được hiểu là xác suất do nằm từ 0 tới 1





- Ngưỡng điển hình là 0,5
- Đặt ngưỡng cao hơn nếu không thích mưa, cần thận trọng hơn. Ví dụ 0,7; nếu nhiệt độ nhỏ hơn 60 độ F ➔ dự đoán là trời mưa.

Hồi quy Logistic

❖ Đánh giá mô hình:

- Mục tiêu:
  - Tìm các tham số tối ưu hóa một hàm xác định mức độ hiệu quả của mô hình
  - Nói đơn giản, mục tiêu là đưa ra các dự đoán gần 1 khi kết quả là 1 và gần 0 khi kết quả là 0.
  - Trong học máy, hàm cần được tối ưu hóa được gọi là hàm mất mát hoặc hàm chi phí. Chúng ta sử dụng hàm mất mát để xác định mức độ phù hợp của mô hình với dữ liệu.
- Hàm Log-Loss, hoặc binary cross-entropy:

$$\text{Log-Loss} = \sum_{i=0}^{n} -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

Hay được dùng trong hồi quy logistic.
Để đánh giá độ sai lệch giữa giá trị dự đoán của mô hình (p_i) và giá trị thực tế (y_i) cho mỗi điểm dữ liệu.

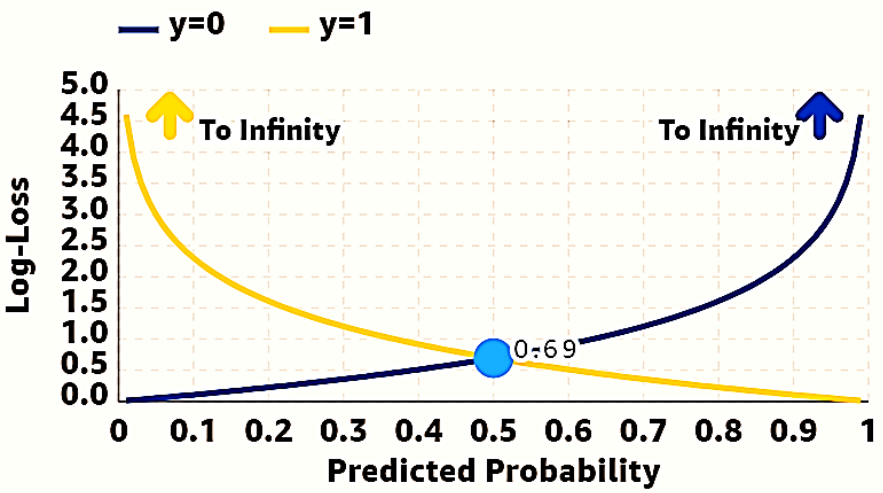
Hồi quy Logistic

❖ Đánh giá mô hình:

- Hàm Log-Loss, hoặc binary cross-entropy:

$$\text{Log-Loss} = \sum_{i=0}^{n} -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

- Phụ thuộc vào giá trị thực của y và xác xuất dự đoán
- Xác suất tiến đến gần giá trị thực y ➔ Log Loss giảm về 0
- Và ngược lại



Hồi quy Logistic

- Giảm thiểu mất mát:
  - Ước tính các hệ số $\hat{\beta}$
  - Sử dụng:
    - Gradient descent
    - Ước tính khả năng tối đa

- Nhớ lại mô tả bài toán:
  - Đầu ra y (có giá trị 0 hoặc 1) có kết quả cụ thể nào đó với xác suất:

$$P(y = 1|X) = \text{sigmoid}(z) = \frac{1}{1 + e^{-z}}$$

  - Với

$$z = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \ldots + \hat{\beta}_k x_k$$

Hồi quy Logistic

- Gradient descent:
  - Mục tiêu là giảm thiểu hàm chi phí Log-Loss trên tất cả các mẫu.
  - Mô tả:
    - Bắt đầu bằng chọn các giá trị tham số ban đầu
    - Cập nhật chúng theo từng bước bằng cách di chuyển chúng theo hướng làm giảm tổn thất
    - Ở mỗi lần lặp, giá trị tham số được cập nhật theo độ dốc, được chia tỷ lệ theo kích thước bước (hay còn gọi là tốc độ học).
    - Các tham số được cập nhật theo hướng ngược lại của độ dốc theo kích thước bước để cố gắng tìm các giá trị tham số làm giảm thiểu Log-Loss
    - Kết quả: cập nhật liên tục các hệ số của mô hình sao cho cuối cùng đạt đến giá trị nhỏ nhất của hàm lỗi và thu được đường cong sigmoid phù hợp với dữ liệu

Hồi quy Logistic

❖ Gradient descent:

- Ví dụ:
  - Mô hình: $$P(y=1|x) = \frac{1}{1+e^{-(-10+0.2x)}}$$
  - Với tham số weight $\hat{\beta}_1 = 0.2$
  - Và bias $\hat{\beta}_0 = -10$

$$\text{Log-Loss} = \sum_{i=0}^{n} -(y_i * \log(p_i) + (1-y_i) * \log(1-p_i))$$

Hồi quy Logistic

- Maximum Likelihood Estimation (Ước lượng khả năng tối đa):
  - Tìm mô hình tối đa hóa khả năng quan sát dữ liệu
  - Việc giảm thiểu Log-Loss tương đương với việc tối đa hóa Log-Likelihood
  - Do đó, mục tiêu là tìm các giá trị tham số tối đa hóa biểu thức sau:

$$\text{Log-Likelihood} = \sum_{i=0}^{n} (y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i))$$

  - Có thể thực hiện điều này bằng cách thực hiện phép vi phân hàm Log-Likelihood theo các tham số, đặt đạo hàm bằng 0 và giải phương trình để tìm ước tính của các tham số. **Hàm vi phân cho biết hướng gia tăng nhanh nhất của hàm tại mỗi điểm.**

Câu hỏi

- Hồi quy logistic áp dụng cho bài toán nhiều lớp như thế nào?

1.2. Các mô hình học máy truyền thống

- Các mô hình học máy không giám sát
  - Giới thiệu về phân cụm
  - k-Means
  - DBSCAN

Các mô hình học máy không giám sát

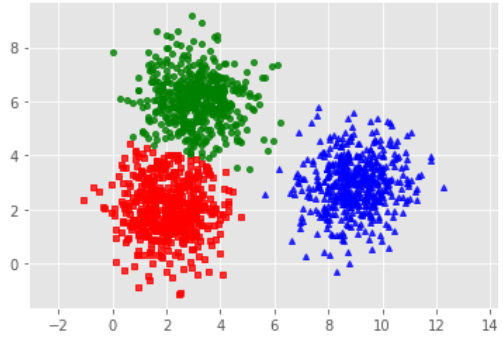
- Học có giám sát (Supervised learning)
  - Tập dữ liệu (dataset) bao gồm các mẫu, mà mỗi mẫu được gắn kèm với một nhãn lớp/giá trị đầu ra mong muốn
  - Mục đích là học (xấp xỉ) một giả thiết (vd: một phân lớp, một hàm mục tiêu,...) phù hợp với tập dữ liệu hiện có
  - Giả thiết học được (learned hypothesis) sau đó sẽ được dùng để phân lớp/dự đoán đối với các mẫu mới
- Học không có giám sát (Unsupervised learning)
  - Tập dữ liệu (dataset) bao gồm các mẫu, mà mỗi mẫu không có thông tin về nhãn lớp/giá trị đầu ra mong muốn
  - Mục đích là tìm ra (học) các cụm/các cấu trúc/các quan hệ tồn tại trong tập dữ liệu hiện có

Phân cụm

- Phân cụm/nhóm (Clustering) là phương pháp học không có giám sát được sử dụng phổ biến nhất
  - Tồn tại các phương pháp học không có giám sát khác, ví dụ: Lọc cộng tác (Collaborative filtering), Khai phá luật kết hợp (Association rule mining), …
- Học phân cụm
  - Đầu vào: một tập dữ liệu không có nhãn (các mẫu không có nhãn lớp/giá trị đầu ra mong muốn)
  - Đầu ra: các cụm (nhóm) của các mẫu
- Một cụm (cluster) là một tập các mẫu
  - Tương tự với nhau (theo một ý nghĩa, đánh giá nào đó)
  - Khác biệt với các mẫu thuộc các cụm khác

Phân cụm

- Phân cụm/nhóm (Clustering) là phương pháp học không có giám sát được sử dụng phổ biến nhất
  - Tồn tại các phương pháp học không có giám sát khác, ví dụ: Lọc cộng tác (Collaborative filtering), Khai phá luật kết hợp (Association rule mining), …
- Học phân cụm
  - Đầu vào: một tập dữ liệu không có nhãn (các mẫu không có nhãn lớp/giá trị đầu ra mong muốn)
  - Đầu ra: các cụm (nhóm) của các mẫu
- Một cụm (cluster) là một tập các mẫu
  - Tương tự với nhau (theo một ý nghĩa, đánh giá nào đó)
  - Khác biệt với các mẫu thuộc các cụm khác
  - Ví dụ: các mẫu được phân làm 3 cụm như hình

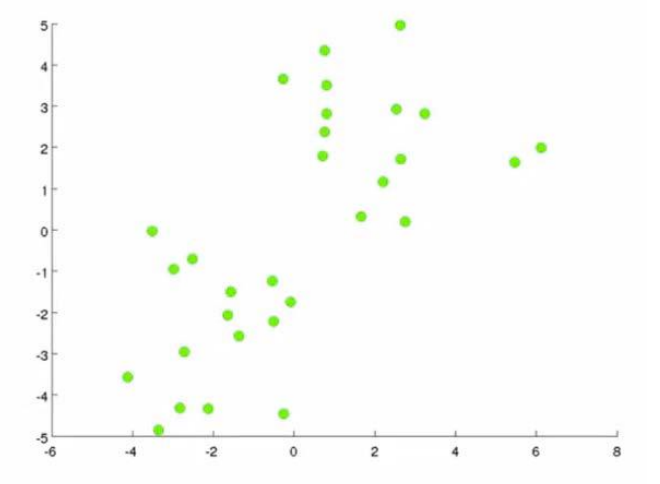


Phân cụm

- Các thành phần
  - Hàm tính khoảng cách (độ tương tự, độ khác biệt)
  - Giải thuật phân cụm
    - Dựa trên phân tách (Partition-based clustering)
    - Dựa trên tích tụ phân cấp (Hierarchical clustering)
    - Bản đồ tự tổ thức (Self-organizing map – SOM)
    - Các mô hình hỗn hợp (Mixture models)
    - …
  - Đánh giá chất lượng phân cụm
    - Khoảng cách/sự khác biệt giữa các cụm → Cần được cực đại hóa
    - Khoảng cách/sự khác biệt bên trong một cụm → Cần được cực tiểu hóa

k-Means

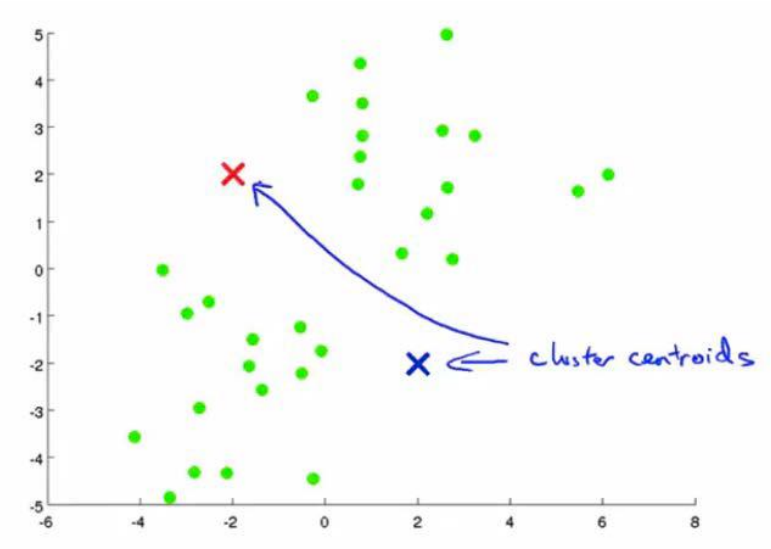
- Phân cụm k-Means:
  - Là phương pháp phổ biến nhất trong các phương pháp phân cụm dựa trên phân tách (partition-based clustering)
  - Tìm k cụm mô tả tốt nhất dữ liệu
  - Mỗi cụm (cluster) có một điểm trung tâm, được gọi là centroid
  - k (tổng số các cụm thu được) là một giá trị được xác định trước
    - Vd: được chỉ định bởi người thiết kế hệ thống phân cụm



Hình ảnh tham khảo trong Andrew Ng(Stanford Univ.), "Machine Learning"

k-Means

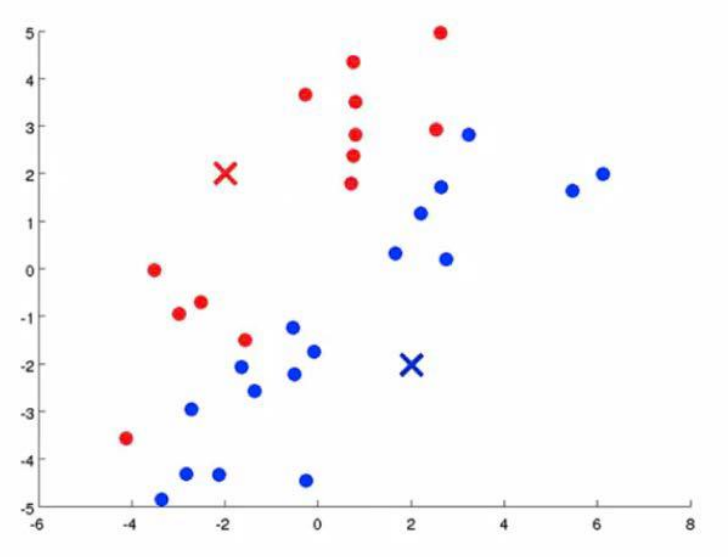
- Phân cụm k-Means:
  - Số cụm k = 2
    - Khởi tạo ngẫu nhiên các "centroid"

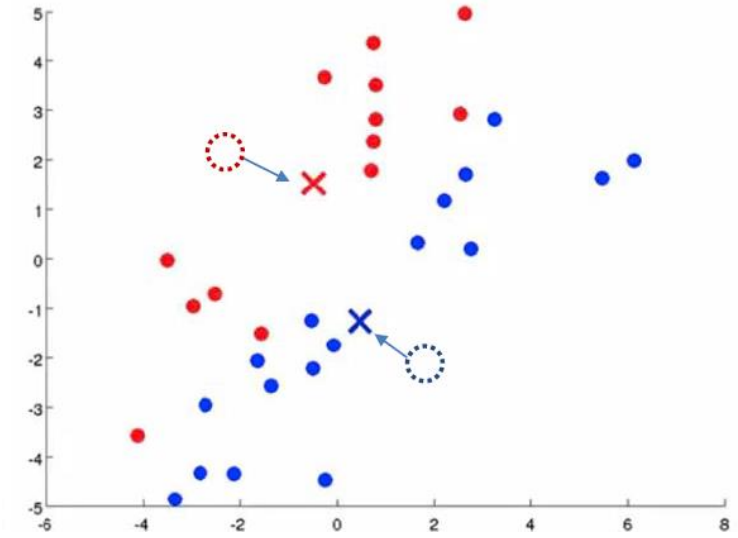


Slides from Andrew Ng(Stanford Univ.), "Machine Learning"

k-Means

- Số cụm k = 2:
  - Gán thành viên cụm
  - Cập nhật centroid cho cụm (trung bình các điểm dữ liệu trong mỗi cụm)

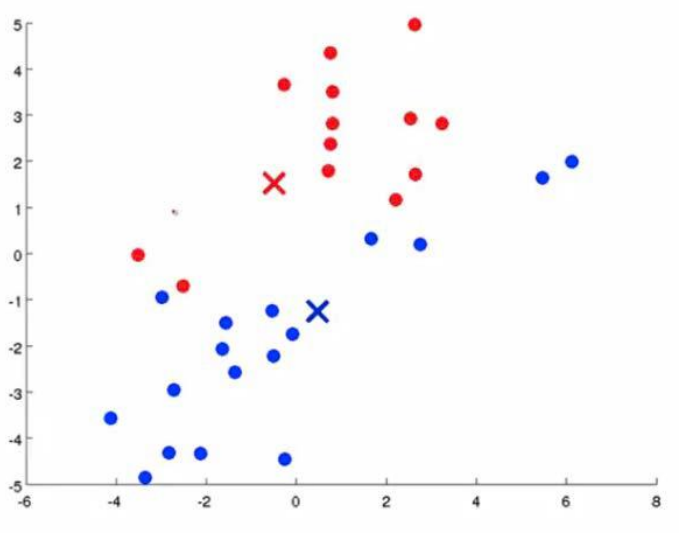




Slides from Andrew Ng(Stanford Univ.), “Machine Learning”

k-Means

- Số cụm k = 2:
  - Cập nhật các thành viên cụm
  - Lặp lại quá trình cho tới khi không có thành viên nào phải cập nhật nữa



Slides from Andrew Ng(Stanford Univ.), “Machine Learning”

k-Means

- Thuật toán: (hình bên)
- Điều kiện hội tụ
  - Không còn hoặc còn không đáng kể mẫu để gán vào cụm
  - Hoặc, không có thay đổi hoặc thay đổi không đáng kể về centroid của các cụm
  - Hoặc tổng lỗi phân cụm giảm không đáng kể

***k*-means**(D, k)

D: Tập ví dụ học

k: Số lượng cụm kết quả (thu được)

Lựa chọn ngẫu nhiên k ví dụ trong tập D để làm các điểm trung tâm ban đầu (initial centroids)

while not CONVERGENCE

  for each ví dụ x∈D

    Tính các khoảng cách từ x đến các điểm trung tâm (centroid)

    Gán x vào cụm có điểm trung tâm (centroid) gần x nhất

  end for

  for each cụm

    Tính (xác định) lại điểm trung tâm (centroid) dựa trên các ví dụ hiện thời đang thuộc vào cụm này

end while

return {k cụm kết quả}

$$Error = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} d(\mathbf{x}, \mathbf{m_i})^2$$

k-Means

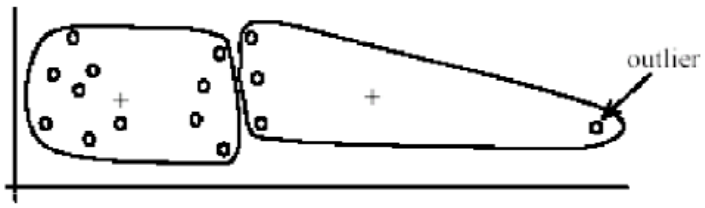
❖ Hàm khoảng cách:

- Ví dụ: Khoảng cách Euclide

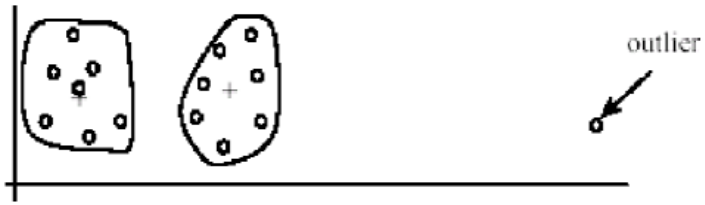
$$d(\mathbf{x}, \mathbf{m_i}) = \|\mathbf{x} - \mathbf{m_i}\| = \sqrt{(x_1 - m_{i1})^2 + (x_2 - m_{i2})^2 + \ldots + (x_n - m_{in})^2}$$

- (vectơ) $\boldsymbol{m_i}$ là điểm trung tâm (centroid) của cụm C_i
- $d(\boldsymbol{x}, \boldsymbol{m_i})$ là khoảng cách giữa ví dụ $\boldsymbol{x}$ và điểm trung tâm $\boldsymbol{m_i}$

k-Means



(A): Undesirable clusters



(B): Ideal clusters

[Liu, 2006]

- Ưu điểm:
  - Dễ hiểu và dễ cài đặt
  - Nếu số cụm ít và số bước lặp cho phân cụm nhỏ thì có độ phức tạp tuyến tính
  - Giải thuật phân cụm phổ biến nhất
- Nhược điểm
  - Cần xác định số cụm k trước
  - Nhạy cảm với các mẫu ngoại lai
    - Khác biệt với các mẫu khác, có thể do lỗi thu thập dữ liệu
  - Phụ thuộc chọn điểm trung tâm ban đầu

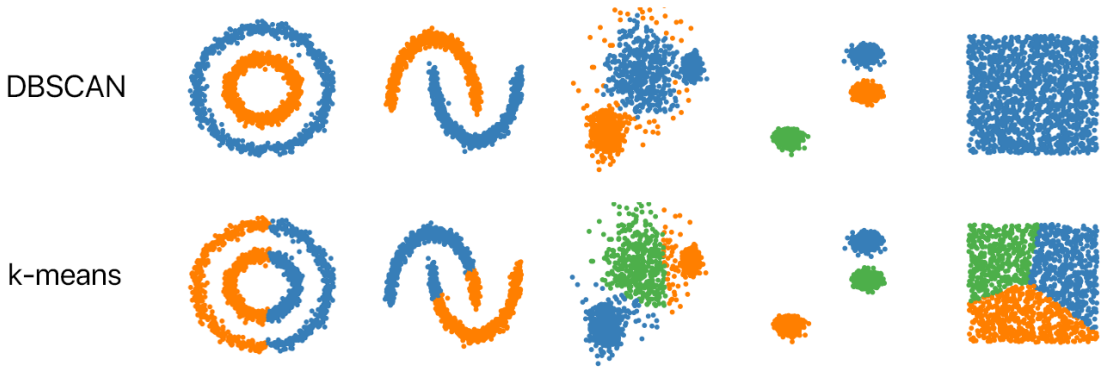
k-Means

❖ Tổng kết:

- Là giải thuật phổ biến nhất được dùng để giải quyết các bài toán phân cụm – do tính đơn giản và hiệu quả
- Các giải thuật phân cụm khác cũng có các nhược điểm riêng
- Về tổng quát, không có lý thuyết nào chứng minh rằng một giải thuật phân cụm khác hiệu quả hơn k-Means
- Một số giải thuật phân cụm có thể phù hợp hơn một số giải thuật khác đối với một số kiểu tập dữ liệu nhất định, hoặc đối với một số bài toán ứng dụng nhất định
- So sánh hiệu năng của các giải thuật phân cụm là một nhiệm vụ khó khăn
- Làm sao để biết được các cụm kết quả thu được là chính xác?

DBSCAN

- Hạn chế của phân cụm dựa trên đo khoảng cách
  - Khó phân biệt các cụm có hình dạng bất quy tắc
  - Khó xác định trước số lượng cụm
  - Xử lý các điểm ngoại lai



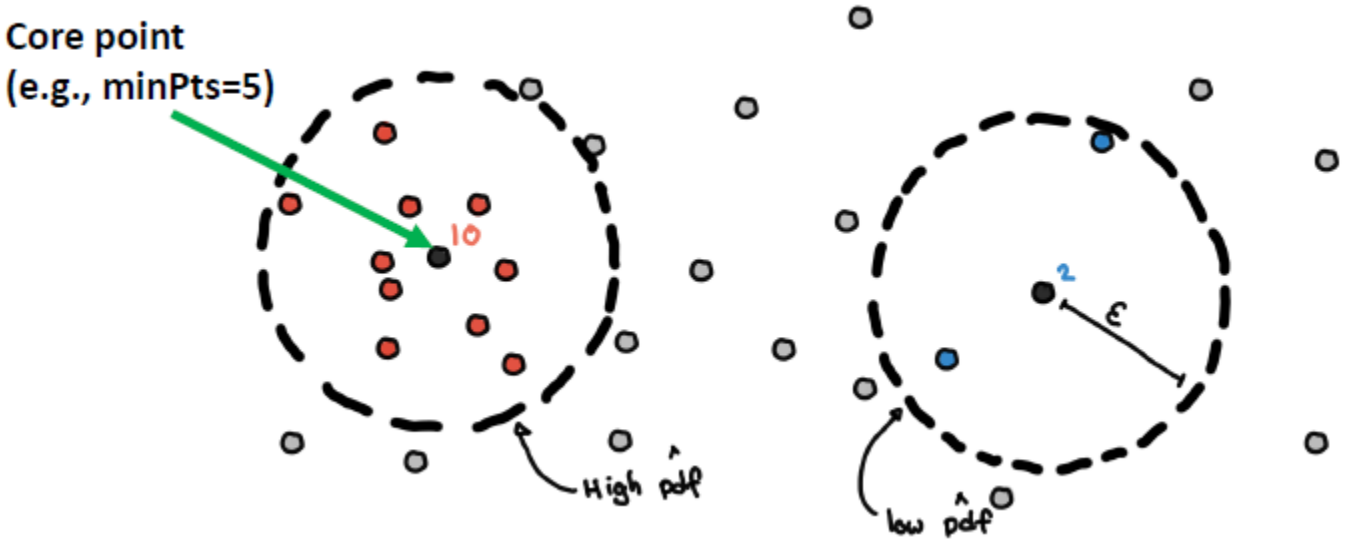
Tham khảo https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

DBSCAN

- DBSCAN (Density-Based Spatial Clustering of Applications with Noise):
  - phân cụm dữ liệu dựa trên mật độ không gian
- Ý tưởng:
  - một cụm trong không gian dữ liệu là một vùng liền kề có mật độ điểm cao, được ngăn cách với các cụm khác bằng các vùng liền kề có mật độ điểm thấp.
  - sử dụng bán kính vùng lân cận để đếm hàng xóm

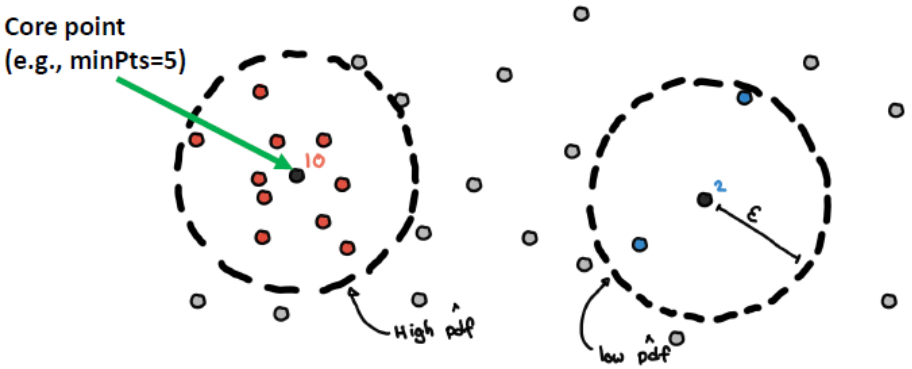
DBSCAN

- DBSCAN (Ester et al, 1996):
  - Hai tham số chính:
    - Eps (ε): Bán kính vùng lân cận.
    - MinPts (MinPoints): Số điểm tối thiểu trong một vùng lân cận để được coi là một cụm mật độ.



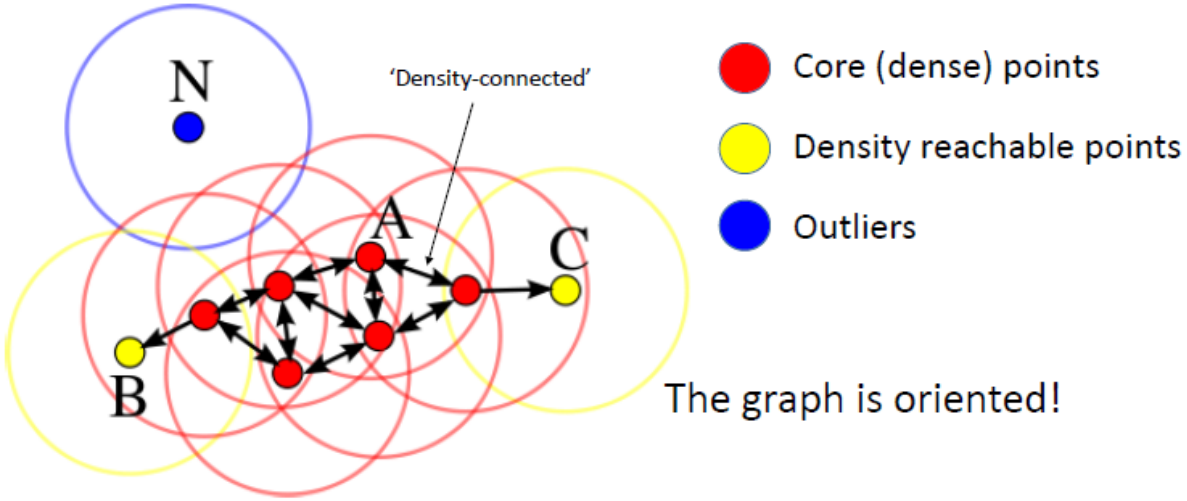
DBSCAN

- DBSCAN (Ester et al, 1996):
  - Thuật toán:
    - Xác định điểm lân cận: Cho mỗi điểm dữ liệu, tìm tất cả các điểm khác nằm trong phạm vi bán kính Eps.
    - Phân loại điểm:
      - Điểm lõi: Điểm có ít nhất MinPts điểm lân cận. Thuộc về một cụm mật độ.
      - Điểm biên: Điểm có ít hơn MinPts điểm lân cận nhưng nằm trong vùng lân cận của điểm lõi. Thuộc về cụm mật độ tương ứng.
      - Điểm nhiễu: Điểm không thuộc vào bất kỳ cụm mật độ nào.
    - Lặp lại: Tiếp tục quá trình cho đến khi tất cả các điểm được phân loại.



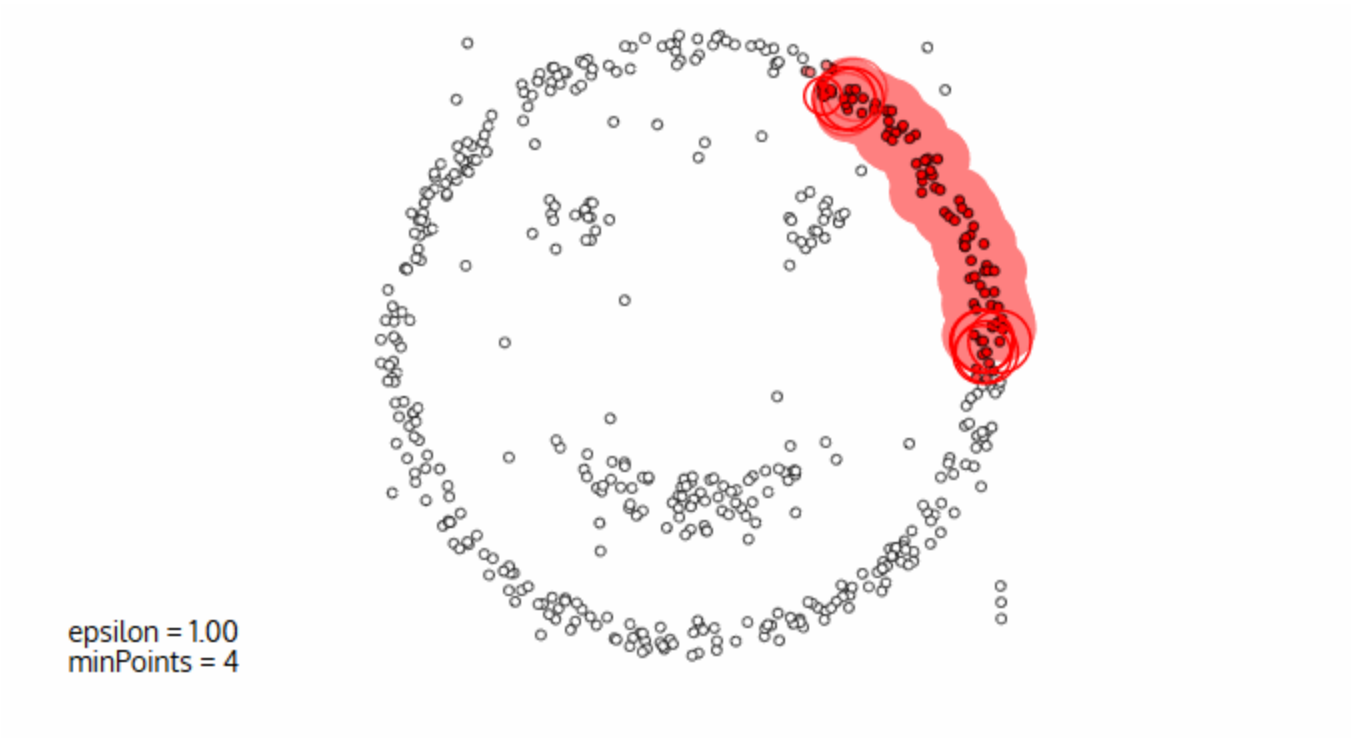
DBSCAN

- DBSCAN (Ester et al, 1996):
  - Kết quả thuật toán:
    - Xác định được các điểm lõi và điểm biên (dựa trên mật độ)
    - Tất cả các điểm trong cụm được kết nối lẫn nhau theo mật độ
    - Nếu một điểm có thể đến được từ một điểm trên cụm, thì nó sẽ thuộc cụm



DBSCAN

- Minh họa: Quá trình lan truyền để xác định các cụm của thuật toán DBSCAN.



Tham khảo từ digitalvidya blog

DBSCAN

- Tổng kết:
  - Ưu điểm:
    - Hiệu quả trong việc xác định các cụm có hình dạng và kích thước khác nhau.
    - Khả năng xử lý nhiễu tốt.
    - Không cần xác định trước số lượng cụm.
  - Nhược điểm:
    - Nhạy cảm với tham số Eps và MinPts.
    - Có thể tốn thời gian tính toán cho các tập dữ liệu lớn.
  - Một số chú ý khác:
    - Có độ phức tạp là O(n log n)
    - Các cụm khác nhau có thể có mật độ khác nhau, phù hợp với các trường hợp cụm thực tế thường có mật độ phân bố không đều.
    - Thư viện Scikit-learn hỗ trợ nhiều loại khoảng cách (Lp metrics) và các phương pháp tìm kiếm láng giềng hiệu quả.

Tổng kết nội dung

- Giới thiệu về các mô hình học máy truyền thống, gồm:
  - Các mô hình học máy có giám sát
    - Cây quyết định
    - Naïve Bayes
    - kNN
    - Rừng ngẫu nhiên
    - SVM
    - Logistic Regression
  - Các mô hình học máy không giám sát
    - k-Means
    - DBSCAN
- Mỗi mô hình có ưu và nhược điểm riêng. Phụ thuộc lớn vào dữ liệu đầu vào, tinh chỉnh tham số, …
- Các mô hình phổ biến và hiệu quả nhất: SVM, Rừng ngẫu nhiên, k-Means

Các câu hỏi và bài tập cuối chương

1. Thực hành làm việc với tập dữ liệu Iris trong scikit learn
2. Thực hành các mô hình đã học sử dụng scikit learn

- Với mỗi mô hình, thử nghiệm các tham số quan trọng của mô hình đã đề cập trên lớp và đánh giá ảnh hưởng của các tham số đó đối với hiệu năng của mô hình (theo độ đo độ chỉnh xác - accuracy).