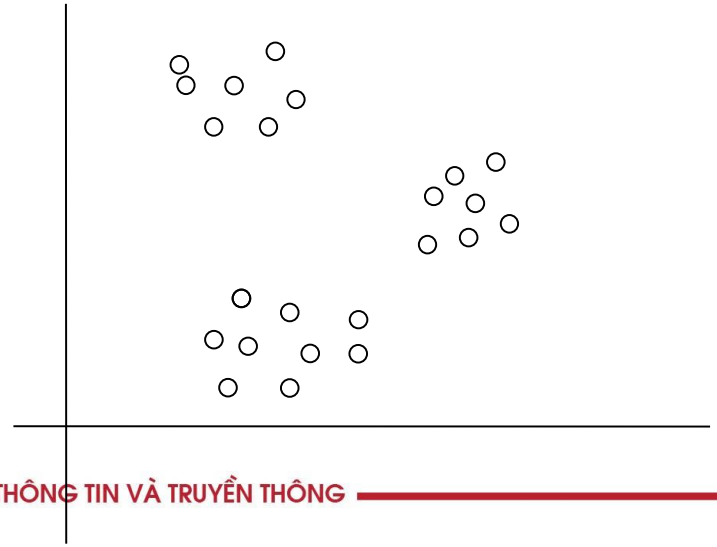
BÀI 2: HỌC MÁY (TIẾP)

Nội dung

1. Các khái niệm cơ bản
2. Thuật toán *k-means*
3. Biểu diễn cụm
4. Phân cụm phân cấp
5. Hàm khoảng cách
6. Chuẩn hóa dữ liệu
7. Xử lý nhiều loại thuộc tính
8. Phương pháp đánh giá
9. Khám phá các lỗ và vùng dữ liệu
10. Học LU
11. Học PU

1. Các k/n cơ bản

- Phân cụm là quá trình tổ chức các phần tử DL thành các nhóm trong đó các thành viên có tính chất tương tự nhau. Mỗi cụm bao gồm các phần tử DL tương tự nhau và khác biệt so với các phần tử DL thuộc các nhóm khác
- Ứng dụng: phân cụm nhóm khách hàng dựa theo sở thích để thiết kế chiến lược marketing; phân cụm khách hàng dựa theo chỉ số cơ thể để bố trí sản xuất quần áo; phân cụm bài báo để tổng hợp tin tức; ...



2. Thuật toán *k-means*

Algorithm k-means(*k*, *D*)
1 chọn k điểm DL làm centroid (trung tâm của cụm)
2 **repeat**
3 **for** mỗi điểm DL ***x*** ∈ *D* **do**
4 tính khoảng cách từ ***x*** tới mỗi centroid;
5 gán ***x*** cho centroid gần nhất // một centroid đại diện cho một cụm
6 **endfor**
7 tính toán lại các centroid dựa trên các cụm hiện tại
8 **until** the stopping criterion is met

Thuật toan K-means (tiếp)

Điều kiện hội tụ:

1. Số điểm DL được gán lại nhỏ hơn một ngưỡng
2. Số centroid bị thay đổi nhỏ hơn một ngưỡng
3. Tổng bình phương lỗi nhỏ hơn một ngưỡng

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} dist(\mathbf{x}, \mathbf{m}_j)^2$$

trong đó:

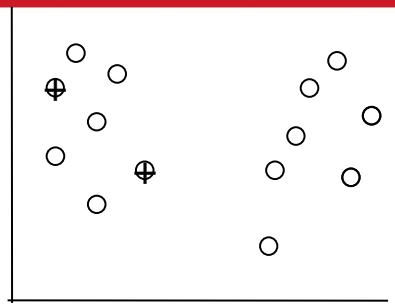
- k là số lượng cụm
- C_j là cụm thứ j
- $\boldsymbol{m}_j$ là centroid của C_j (véc-tơ trung bình của các điểm DL thuộc C_j)
- dist($\boldsymbol{x}$, $\boldsymbol{m}_j$) là khoảng cách giữa $\boldsymbol{x}$ và $\boldsymbol{m}_j$

$$\mathbf{m}_j = \frac{1}{|C_j|} \sum_{\mathbf{x}_i \in C_j} \mathbf{x}_i$$

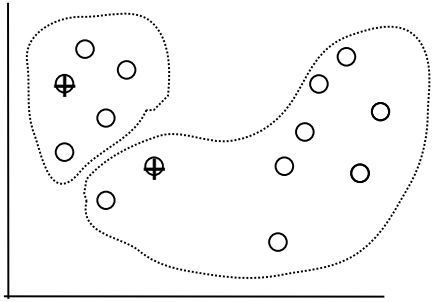
$$dist(\mathbf{x}_i, \mathbf{m}_j) = \| \mathbf{x}_i - \mathbf{m}_j \|$$

$$= \sqrt{(x_{i1} - m_{j1})^2 + (x_{i2} - m_{j2})^2 + \ldots + (x_{ir} - m_{jr})^2}$$

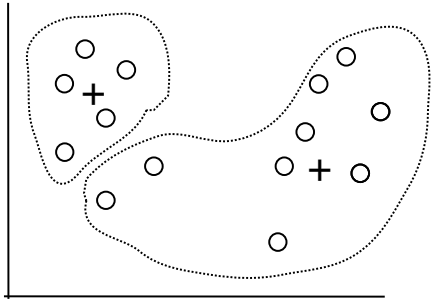
(A) Lựa chọn ngẫu nhiên k centroid



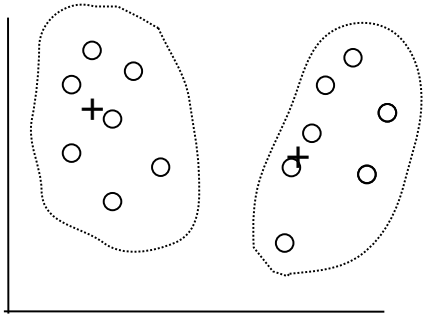
Vòng lặp 1:
(B) Gán cụm



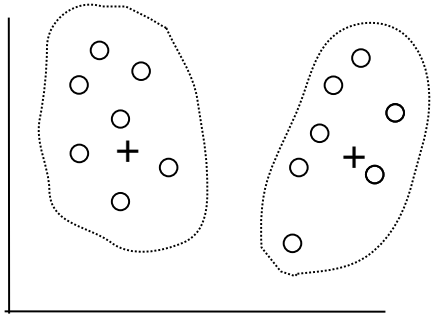
(C) Tính lại centroid



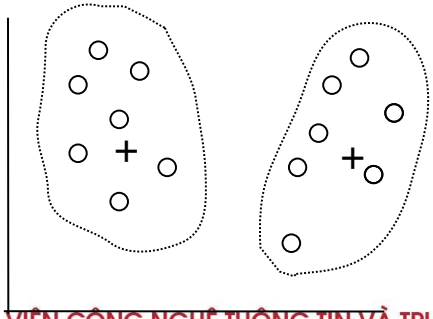
Vòng lặp 2:
(D) Gán cụm



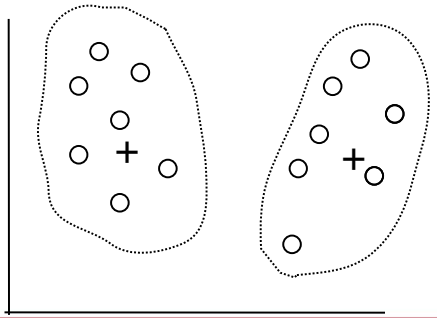
(E) Tính lại centroid



Vòng lặp 3:
(F) Gán cụm



(G) Tính lại centroid



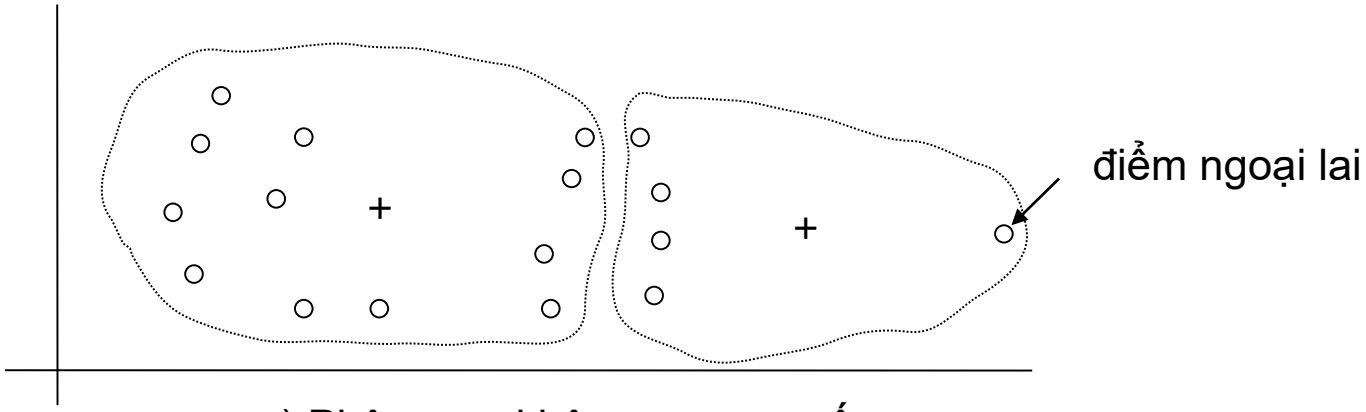
Thuật toán K-Means (tiếp)

Algorithm disk-k-means(*k*, *D*)
1 Chọn *k* điểm DL làm centroid $\boldsymbol{m}_j$, $j = 1, ..., k$;
2 **repeat**
3 khởi tạo $s_j \leftarrow 0$, $j = 1, ..., k$; // 0 là véc-tơ với các thành phần bằng 0
4 khởi tạo $n_j \leftarrow 0$, $j = 1, ..., k$; // n_j là số điểm trong cụm j
5 **for** mỗi điểm DL $\boldsymbol{x} \in D$ **do**
6 $j \leftarrow$ argmin dist($\boldsymbol{x}$,$\boldsymbol{m}_j$);
7 gán $\boldsymbol{x}$ cho cụm j;
8 $s_j \leftarrow s_j + x$;
9 $n_j \leftarrow n_j + 1$;
10 **endfor**
11 $\boldsymbol{m}_j \leftarrow s_j / n_j$, $j = 1, ..., k$;
12 **until** đ/k dừng thỏa mãn

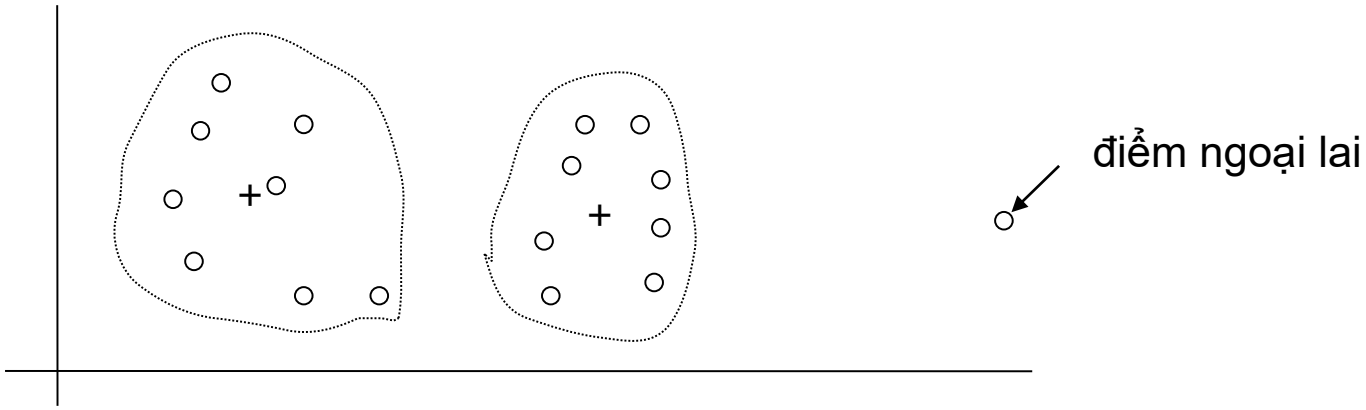
Thuật toán K-Means (tiếp)

- O(*tkn*) trong đó *t* là số vòng lặp, *k* là số cụm, *n* là số ví dụ trong DL huấn luyện
- Chỉ áp dụng cho DL tồn tại mean, đối với DL rời rạc, áp dụng thuật toán *k-modes*
- Giá trị *k* cho trước
- Nhạy cảm với các điểm DL ngoại lai (outlier) (các điểm nằm xa các điểm còn lại trong tập DL)
- Nhạy cảm với việc khởi tạo (thường tiến đến cực trị địa phương)
- Không phù hợp với các cụm có dạng siêu cầu

Thuật toán K-Means (tiếp)

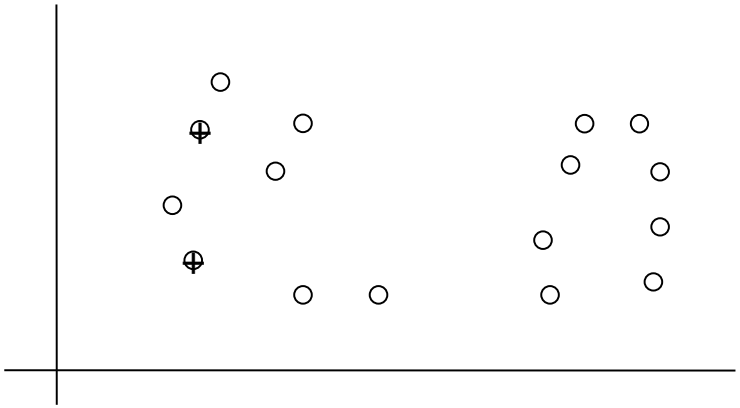


a) Phân cụm không mong muốn

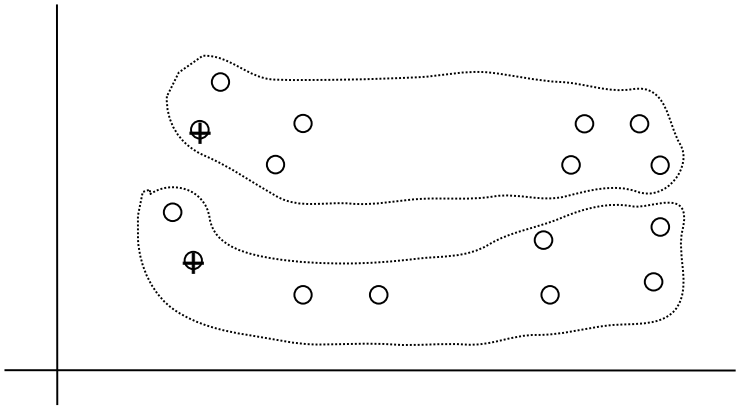


a) Phân cụm lý tưởng

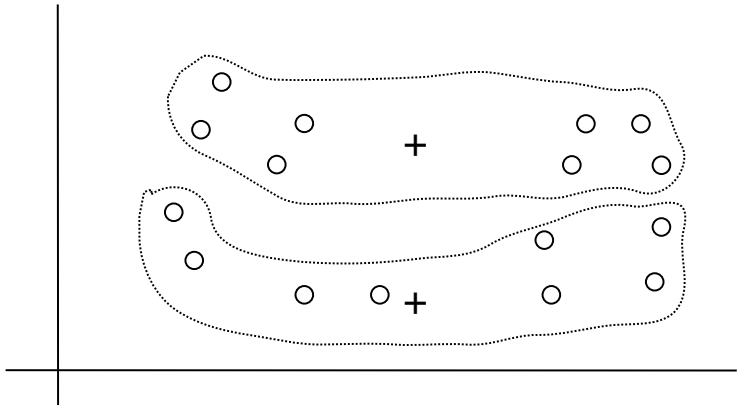
Thuật toán K-Means (tiếp)



(A) Khởi tạo ngẫu nhiên

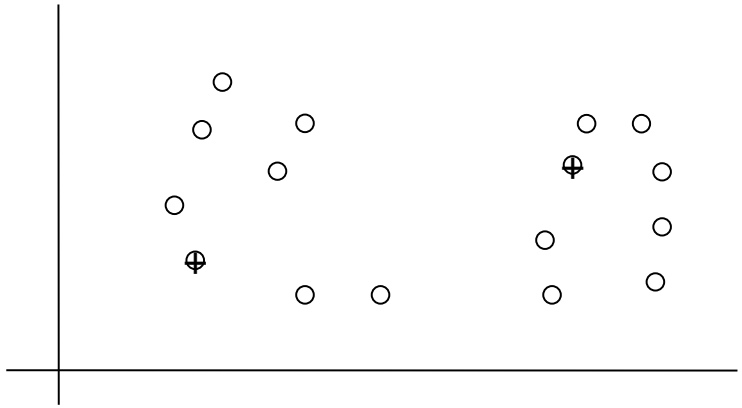


(B) Vòng lặp 1

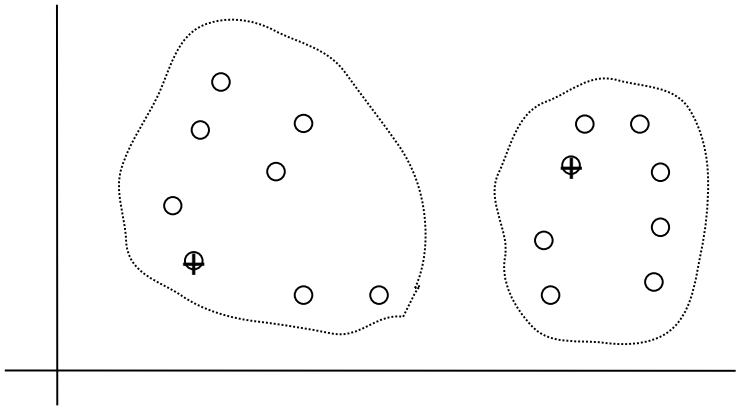


(C) Vòng lặp 2

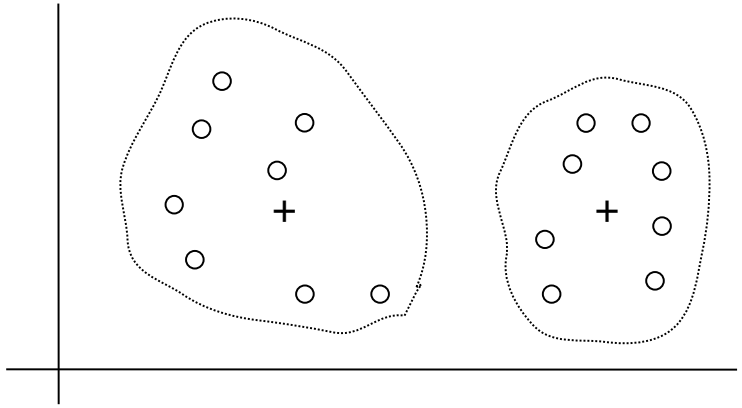
Thuật toán K-Means (tiếp)



(A) Khởi tạo ngẫu nhiên

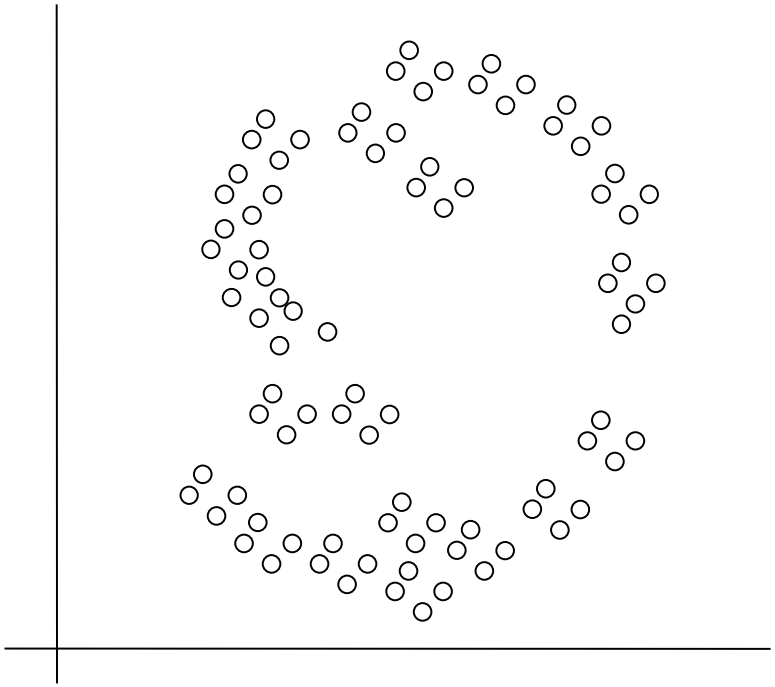


(B) Vòng lặp 1

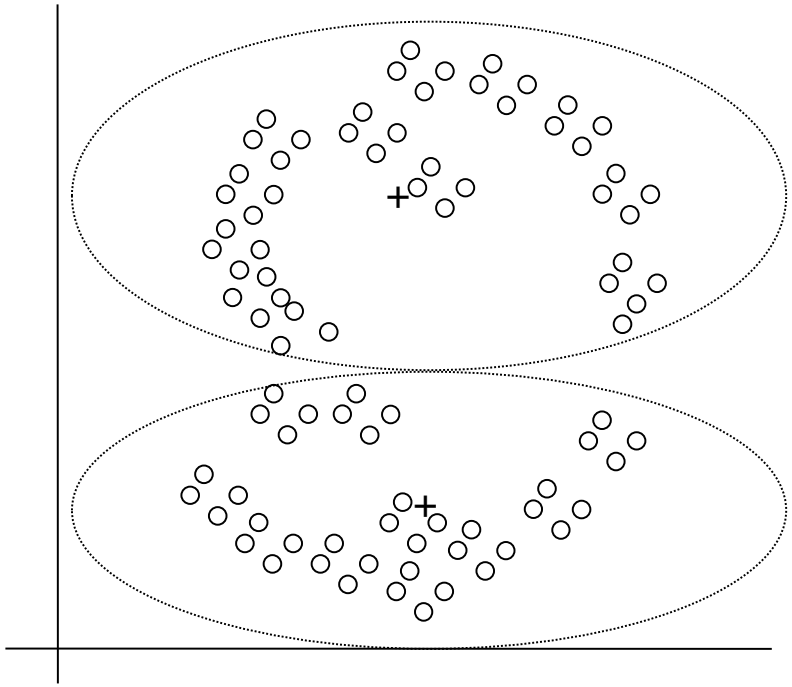


(B) Vòng lặp 2

Thuật toán K-Means (tiếp)



(A) Hai cụm siêu cầu tự nhiên



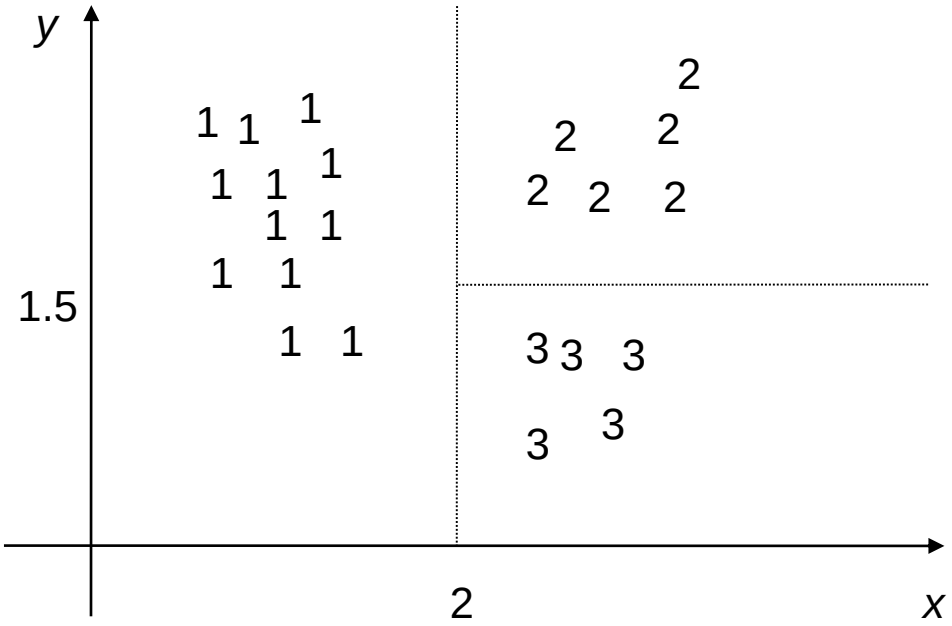
(A) Kết quả của *k-means* (k = 2)

3. Biểu diễn cụm

- Các cách biểu diễn phổ biến:
  - Dựa trên centroid: Phù hợp với cụm dạng ê-líp hoặc dạng cầu
  - Dựa trên mô hình phân loại: G/s mỗi cụm ứng với một lớp với các thành viên của cụm có nhãn lớp tương ứng
  - Dựa trên các giá trị phổ biến trong cụm: Phù hợp với giá trị rời rạc, bao gồm văn bản

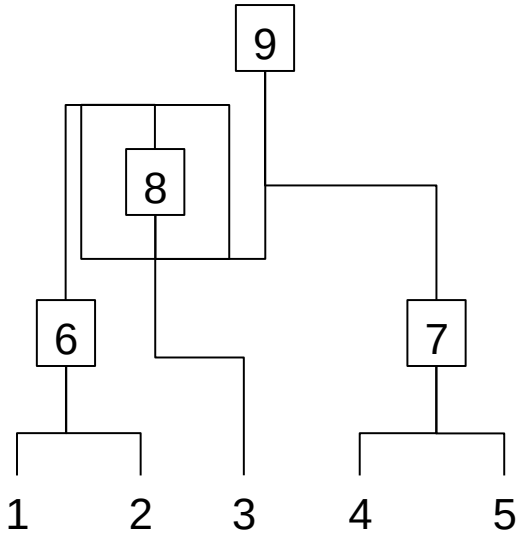
Biểu diễn cụm (tiếp)

x ≥ 2 → cụm 1
X >2, y > 1.5 → cụm 2
X > 2, y ≤ 1.5 → cụm 3



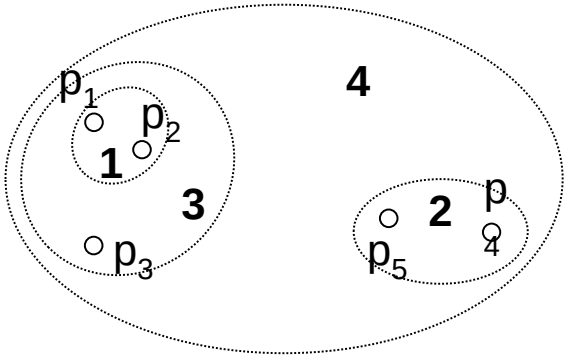
4. Phân cụm phân cấp

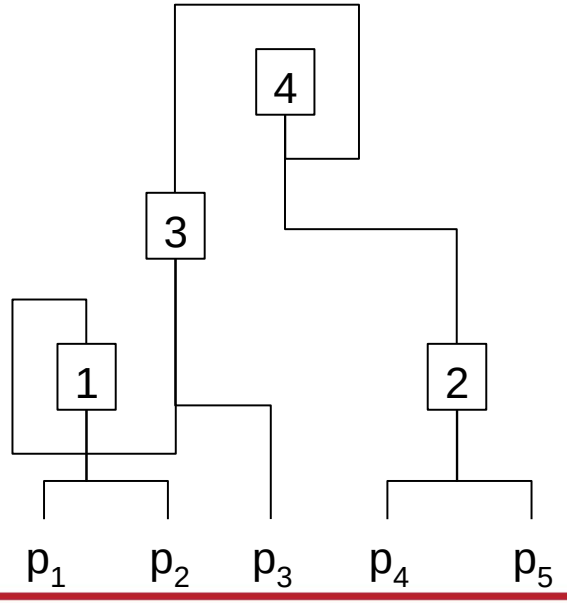
- DL được chia thành một chuỗi các cụm lồng nhau theo cấu trúc cây (dendogram)
- Lá của cây là các điểm DL, gốc chứa một cụm duy nhất, các nút trung gian chứa các nút cụm con
- Phân cụm từ dưới lên: Một cặp cụm gần nhất tại mỗi mức được gộp lại ở mức tiếp theo. Quá trình lặp lại cho tới khi chỉ còn một cụm
- Phân cụm từ trên xuống: Một cụm ban đầu chứa toàn bộ DL. Cụm này được chia thành các cụm con. Một cụm con được chia một cách đệ quy tới khi chỉ còn một phần tử



Phân cụm phân cấp (tiếp)

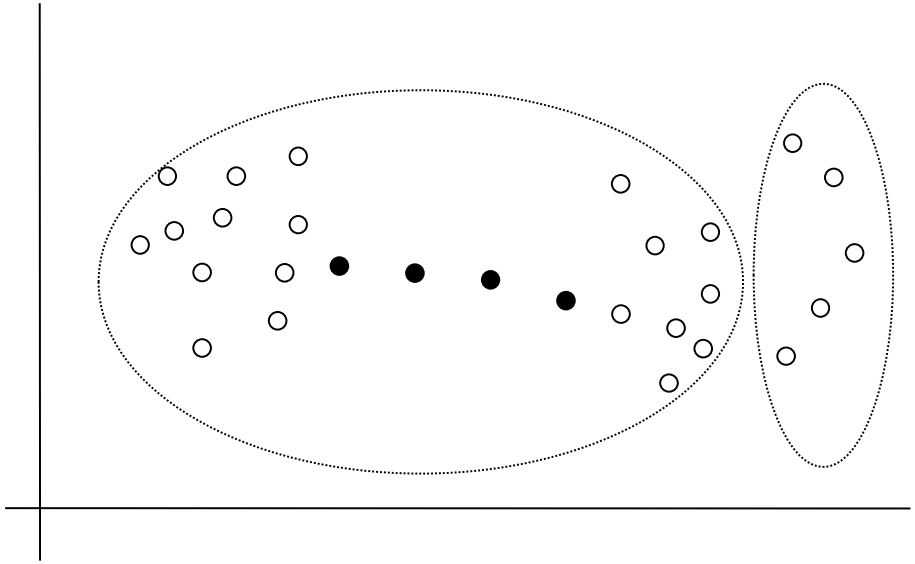
Algorithm Agglomerative(*D*)
1 Coi mỗi điểm DL trong *D* là một cụm,
2 Tính toán các cặp khoảng cách của $x_1, x_2, ..., x_n \in D$;
3 **repeat**
4 tìm hai cụm gần nhau nhất;
5 kết hợp hai cụm thành cụm mới *c*;
6 tính toán khoảng cách từ *c* tới các cụm khác;
7 **until** chỉ còn lại một cụm



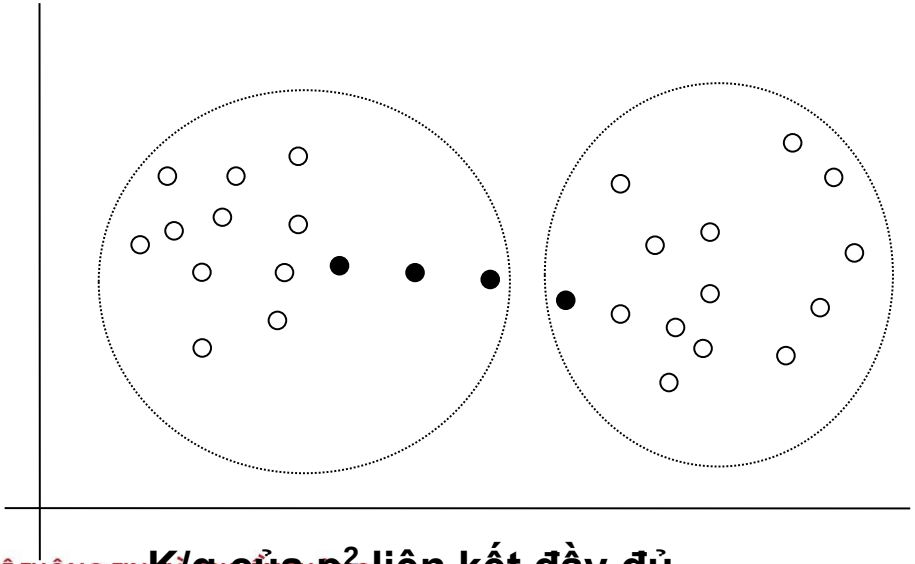


Phân cụm phân cấp (tiếp)

- P^2 liên kết đơn:
  - K/c giữa hai cụm là k/c ngắn nhất giữa hai điểm DL của mỗi cụm
  - Phù hợp với các cụm không có dạng ê-líp
  - Có thể gây ra hiệu ứng chuỗi do nhiễu trong DL
  - Độ phức tạp tính toán: $O(n^2)$
- P^2 liên kết đầy đủ:
  - K/c giữa hai cụm là k/c lớn nhất giữa hai điểm DL của mỗi cụm
  - Không gặp hiệu ứng chuỗi nhưng nhạy cảm với điểm DL ngoại lai
  - Độ phức tạp tính toán $O(n^2 \log n)$
- P^2 liên kết trung bình:
  - K/c giữa hai cụm là k/c trung bình giữa hai điểm DL của mỗi cụm
  - Độ phức tạp tính toán: $O(n^2 \log n)$



Hiệu ứng chuỗi của p^2 liên kết đơn



K/q của p^2 liên kết đầy đủ

Phân cụm phân cấp (tiếp)

- Ngoài ra còn có các p^2 khác, vd:
  - K/c giữa hai cụm là k/c giữa hai centroid của chúng
  - P^2 *Ward*: K/c giữa hai cụm là độ tăng tổng bình phương lỗi từ hai cụm đó tới cụm mới (nếu) được kết hợp
- Ưu điểm: Phân cụm phân cấp cho phép tạo ra số lượng cụm tùy theo mức trên cây
- Nhược điểm: Phân cụm phân cấp có chi phí tính toán và lưu trữ cao

5. Hàm khoảng cách
5.1 Thuộc tính liên tục

Minkowski($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = $(|x_{i1}-x_{j1}|^h+|x_{i2}-x_{j2}|^h+...+|x_{ir}-x_{jr}|^h)^{\frac{1}{h}}$

Euclidean($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = $\sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+...+(x_{ir}-x_{jr})^2}$.

Manhattan($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = $|x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+...+|x_{ir}-x_{jr}|$

Weighted_Euclidean($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = $\sqrt{w_1(x_{i1}-x_{j1})^2+w_2(x_{i2}-x_{j2})^2+...+w_r(x_{ir}-x_{jr})^2}$.

Squared_Euclidean($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = $(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+...+(x_{ir}-x_{jr})^2$

Chebychev($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = $\max(|x_{i1}-x_{j1}|,|x_{i2}-x_{j2}|,...,|x_{ir}-x_{jr}|)$

5.2 Thuộc tính nhị phân & rời rạc

		Điểm DL x_j		
		1	0	
	1	a	b	a + b
Điểm DL x_i	0	c	d	c + d
		a +c	b + d	a + b + c + d

Ma trận nhập nhằng của hai điểm DL chỉ chứa thuộc tính nhị phân

Thuộc tính đối xứng: Hai giá trị nhị phân có tầm quan trọng như nhau

$$\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{b + c}{a + b + c + d}$$

VD:

$\boldsymbol{x}_1$	1	1	1	0	1	0	0
$\boldsymbol{x}_2$	0	1	1	0	0	1	0

$$\text{dist}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{2 + 1}{2 + 2 + 1 + 2} = 3/7 = 0.429$$

Thuộc tính nhị phân & rời rạc (tiếp)

Thuộc tính bất đối xứng: Hai giá trị nhị phân có tầm quan trọng khác nhau

$$\text{Jaccard}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{b + c}{a + b + c}$$

Thuộc tính rời rạc tổng quát:

$$\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{r - q}{r}$$

trong đó - r: số lượng thuộc tính
- q: số lượng giá trị khớp nhau giữa $\boldsymbol{x}_i$ và $\boldsymbol{x}_j$

6. Chuẩn hóa DL

VD:
Thuộc tính 1 có giá trị trong khoảng [0, 1], thuộc tính 2 có giá trị trong khoảng [0, 1000]
$\boldsymbol{x}_i$: (0.1, 20), $\boldsymbol{x}_j$: (0.9. 720)

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(0.9-0.1)^2 + (720-20)^2} = 700.000457$$

Sau khi chuẩn hóa thuộc tính 2 về khoảng [0, 1]
$\boldsymbol{X}_i$: (0.1, 0.02), $\boldsymbol{x}_j$: (0.9, 0.72) → dist($\boldsymbol{x}_i$, $\boldsymbol{x}_j$) = 1.063

Thuộc tính liên tục tuyến tính:

$$\text{rg}(x_{if}) = \frac{x_{ij} - \min(f)}{\max(f) - \min(f)}$$

$$\sigma_f = \sqrt{\frac{\sum_{i=1}^{n}(x_{if} - \mu_f)^2}{n-1}}, \quad \mu_f = \frac{1}{n}\sum_{i=1}^{n} x_{if}, \qquad z(x_{if}) = \frac{x_{ij} - \mu_f}{\sigma_f}$$

Chuẩn hóa DL (tiếp)

- Thuộc tính liên tục dạng mũ: lo-ga-rít hóa

VD: Ae^{Bt}

- Thuộc tính rời rạc không có trật tự (vd: các loại hoa quả): Có thể chuyển về dạng nhị phân
- Thuộc tính rời rạc có trật tự (vd: lứa tuổi): chuẩn hóa tương tự thuộc tính liên tục tuyến tính

7. Xử lý nhiều loại thuộc tính

- DL chứa nhiều loại thuộc tính: nhị phân đối xứng, nhị phân bất đối xứng, liên tục tuyến tính, liên tục phi tuyến, rời rạc, rời rạc có trật tự
- Chuyển đổi tất cả về loại thuộc tính phổ biến nhất (vd liên tục tuyến tính)
- Tính k/c trên từng thuộc tính và tổng hợp lại

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{f=1}^{r} \delta_{ij}^f d_{ij}^f}{\sum_{f=1}^{r} \delta_{ij}^f}.$$

trong đó $\boldsymbol{r}$ là số thuộc tính trong DL, d_{ij}^f là k/c giữa $\boldsymbol{x}_i$ và $\boldsymbol{x}_j$ tính theo thuộc tính f, $\delta_{ij}^f = 1$ nếu thuộc tính f tồn tại ở cả $\boldsymbol{x}_i$ và $\boldsymbol{x}_j$ và $\delta_{ij}^f = 0$ nếu ngược lại

8. P^2 đánh giá

- Dựa trên người dùng: Dựa trên đánh giá của một nhóm các chuyên gia. Đánh giá cuối cùng là trung bình của cả nhóm. Phù hợp với một số loại DL (văn bản)
- Dựa trên khả năng phân loại: Sử dụng DL được phân loại. Mỗi lớp tương ứng với một cụm. Sử dụng các độ đo phân loại

$$entropy(D_i) = -\sum_{j=1}^{k} \Pr_i(c_j) \log_2 \Pr_i(c_j).$$

$$entropy_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times entropy(D_i)$$

$$purity(D_i) = \max_i(\Pr_i(c_j))$$

$$purity_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times purity(D_i).$$

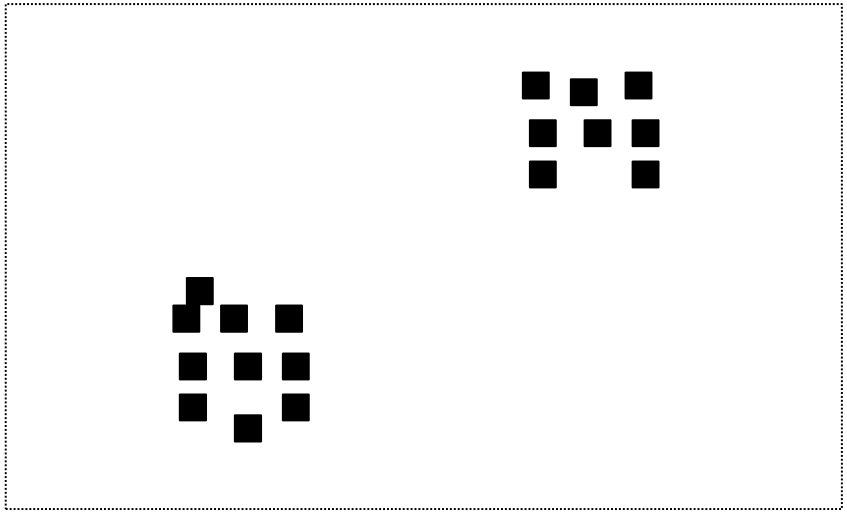
P2 đánh giá (tiếp)

- Độ nén: Thể hiện mức độ tập trung của các điểm DL trong một cụm xung quanh centroid (vd tổng bình phương lỗi)
- Độ cô lập: Thể hiện mức độ phân tách của các cụm thông qua k/c giữa các centroid
- Đánh giá gián tiếp: Phân cụm được sử dụng làm tác vụ trung gian → đánh giá kĩ thuật phân cụm thông qua đánh giá tác vụ cuối. Vd: phân cụm người dùng được áp dụng trong hệ gợi ý (sản phẩm)

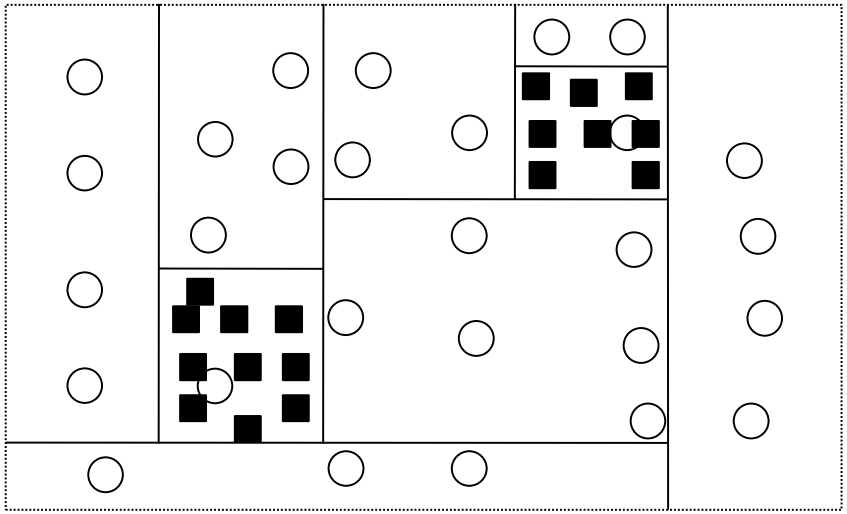
9. Khám phá lỗ và vùng DL

- DL tồn tại các vùng tập trung điểm DL và các vùng không chứa hoặc chứa ít DL (các lỗ)
- Việc khám phá các lỗ DL có vai trò quan trọng trong một số ứng dụng
- Bài toán phân loại:

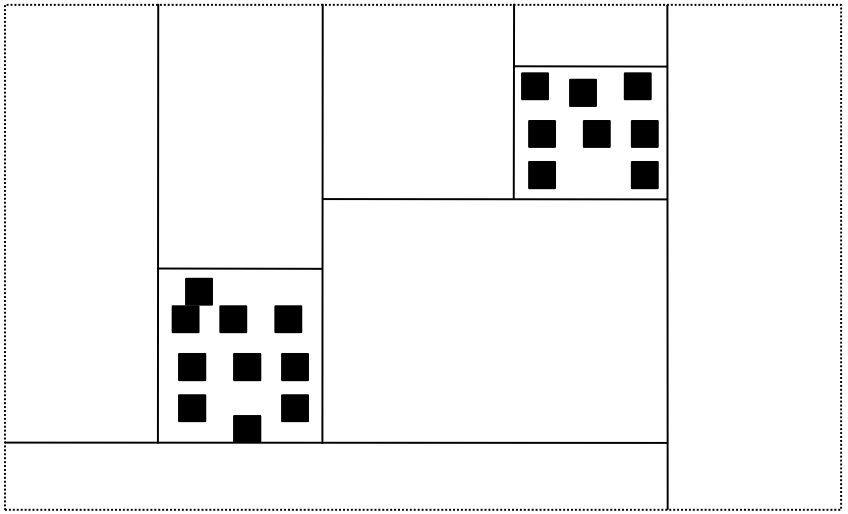
1. G/s các điểm DL đã có có nhãn Y. Thêm các điểm DL mới một cách ngẫu nhiên và gán nhãn N
2. Sử dụng một kĩ thuật phân loại (vd cây quyết định) để phân loại DL
3. Thu được mô hình phân loại với nhãn quan tâm N



(A) Không gian DL gốc



(B) Phân vùng với các điểm DL bổ sung



(C) Phân vùng trên DL gốc

10. Học LU

- Học có giám sát cho độ chính xác cao nhưng đòi hỏi nhiều DL có nhãn. Gán nhãn DL là công việc thủ công, đòi hỏi nhiều thời gian và công sức. Trong các ứng dụng như phân loại văn bản web, các nhãn DL liên tục thay đổi.
- Học LU (labeled and unlabeled examples) xây dựng bộ phân loại trên một lượng nhỏ DL có nhãn và sử dụng một lượng lớn DL không có nhãn để cải thiện bộ phân loại
- Vd: Bộ phân loại văn bản sử dụng 'bài tập' làm đặc trưng để phân loại các văn bản thuộc chủ đề *giáo dục*. Dựa trên DL không có nhãn, có thể phát hiện ra 'bài tập' thường cùng x/h với 'bài giảng', từ đó bổ sung 'bài giảng' làm đặc trưng cho bộ phân loại

10.1 Thuật toán EM

- EM (Expectation Maximization) là thuật toán lặp để cực đại hóa ước lượng khả năng đối với DL khuyết thiếu. Bước kỳ vọng làm đầy DL khuyết thiếu dựa trên ước lượng của tham số hiện tại. Bước cực đại hóa ước lượng lại tham số với mục tiêu cực đại hóa khả năng.
- EM + mô hình phân loại:

1) DL có nhãn *L* được dùng để xây dựng bộ phân loại *f*

2) Sử dụng *f* để phân loại DL chưa có nhãn *U*

3) Cập nhật lại f dựa trên *L* và *U*; quay lại 2), lặp tới khi hội tụ

Thuật toán EM (tiếp)

Algorithm EM(L, U)
1 Học bộ phân loại NB *f* từ tập DL có nhãn *L*
2 **repeat**
// Bước E
3 **for** mỗi ví dụ d_i trong U **do**
4 Dùng *f* để tính Pr($c_j|d_i$)
5 **endfor**
// Bước M
6 học bộ phân loại *f* từ *L* và *U* (tính Pr(c_j) và Pr($w_t|c_j$)
7 **until** the classifier parameters stabilize

Thuật toán EM (tiếp)

Cho D_o gồm các ví dụ có nhãn, D_u gồm các ví dụ không có nhãn
Hàm log likelihood có dạng:

$$\log \Pr(D_o;\Theta) = \log \sum_{D_u} \Pr(D_o, D_u;\Theta)$$

Thay vì cực đại hóa log likelihood, ta cực đại hóa kì vọng của log likelihood đầy đủ:

$$\sum_{D_u} E(D_u \mid D_o;\Theta^{T-1}) \log \Pr(D_o, D_u;\Theta)$$

Xác suất điều kiện của một văn bản biết lớp của nó:

$$\Pr(d_i \mid c_j;\Theta) = \Pr(\mid d_i \mid) \mid d_i \mid! \prod_{t=1}^{\mid V \mid} \frac{\Pr(w_t \mid c_j;\Theta)^{N_{ti}}}{N_{ti}!}$$

G/s các văn bản được sinh ra độc lập, hàm likelihood có thể viết dưới dạng:

$$\prod_{i=1}^{\mid D \mid} \Pr(d_i \mid c_{(i)};\Theta) \Pr(c_{(i)};\Theta) = \prod_{i=1}^{\mid D \mid} \Pr(\mid d_i \mid) \mid d_i \mid! \prod_{t=1}^{\mid V \mid} \frac{\Pr(w_t \mid c_{(i)};\Theta)^{N_{ti}}}{N_{ti}!} \Pr(c_{(i)};\Theta)$$

Thuật toán EM (tiếp)

Hàm log likelihood có dạng:

$$\sum_{i=1}^{|D|}\sum_{t=1}^{|V|} N_{ti} \log \Pr(w_t \mid c_{(i)}; \Theta) + \sum_{i=1}^{|D|} \log \Pr(c_{(i)}; \Theta) + \phi$$

Sử dụng biến indicator h_{ki}, h_{ki} = 1 khi văn bản i có nhãn k, hàm log likelihood có dạng:

$$\sum_{i=1}^{|D|}\sum_{t=1}^{|V|}\sum_{k=1}^{|C|} h_{ik} N_{ti} \log \Pr(w_t \mid c_k; \Theta) + \sum_{i=1}^{|D|}\sum_{k=1}^{|C|} h_{ik} \log \Pr(c_k; \Theta) + \phi$$

Kì vọng của log likelihood đầy đủ:

$$\sum_{i=1}^{|D|}\sum_{t=1}^{|V|}\sum_{k=1}^{|C|} \Pr(c_k \mid d_i; \Theta^{T-1}) N_{ti} \log \Pr(w_t \mid c_k; \Theta)$$

$$+ \sum_{i=1}^{|D|}\sum_{k=1}^{|C|} \Pr(c_k \mid d_i; \Theta^{T-1}) \log \Pr(c_k; \Theta) + \phi$$

Thuật toán EM (tiếp)

Lagrangian:

$$\sum_{i=1}^{|D|}\sum_{t=1}^{|V|}\sum_{k=1}^{|C|}\Pr(c_k \mid d_i;\Theta^{T-1})N_{ti}\log\Pr(w_t \mid c_k;\Theta)$$

$$+\sum_{i=1}^{|D|}\sum_{k=1}^{|C|}\Pr(c_k \mid d_i;\Theta^{T-1})\log\Pr(c_k;\Theta)$$

$$+\lambda\left(1-\sum_{k=1}^{|C|}\Pr(c_k;\Theta)\right)+\sum_{t=1}^{|V|}\sum_{k=1}^{|C|}\lambda_{tk}\left(1-\sum_{t=1}^{|V|}\Pr(w_t \mid c_k;\Theta)\right)+\phi$$

Lấy đạo hàm theo λ, ta có

$$\sum_{k=1}^{|C|}\Pr(c_k;\Theta)=1$$

Lấy đạo hàm theo $\Pr(c_k;\Theta)$, ta có

$$\sum_{i=1}^{|D|}\Pr(c_k \mid d_i;\Theta^{T-1})=\lambda\Pr(c_k;\Theta)$$ với $k = 1, 2, \ldots, |C|$

Tính tổng theo k, ta có

$$\lambda=\sum_{i=1}^{|D|}\sum_{k=1}^{|C|}\Pr(c_k \mid d_i;\Theta^{T-1})=|D|$$

Thuật toán EM (tiếp)

Cập nhật $\Pr(c_j|\ \Theta)$:

$$\Pr(c_j;\Theta^T) = \frac{\sum_{i=1}^{|D|} \Pr(c_j \mid d_i;\Theta^{T-1})}{|D|}$$

Cập nhật $\Pr(w_t|\ c_j,\ \Theta)$:

$$\Pr(w_t \mid c_j;\Theta^T) = \frac{\sum_{i=1}^{|D|} N_{ti} \Pr(c_j \mid d_i;\Theta^{T-1})}{\sum_{s=1}^{|V|}\sum_{i=1}^{|D|} N_{si} \Pr(c_j \mid d_i;\Theta^{T-1})}$$

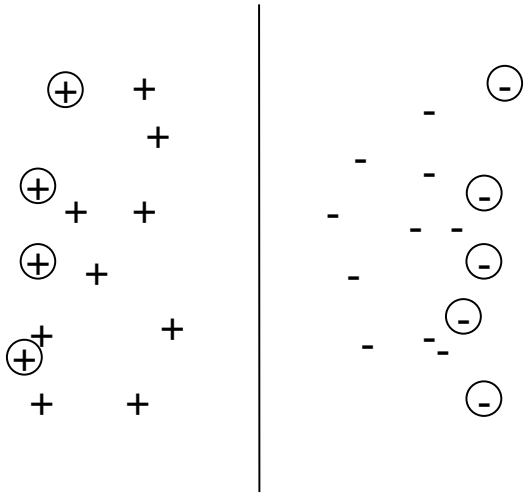
10.2 Đồng - huấn luyện

- Giả thiết tập các thuộc tính X có thể chia làm hai tập X_1 và X_2 để xây dựng hai bộ phân loại f_1 và f_2
- Giả thiết 1: Bộ phân loại xây dựng trên toàn bộ thuộc tính f có kết quả phân loại giống như f_1 và f_2
- Giả thiết 2: X_1 và X_2 độc lập với nhau đối với nhãn lớp

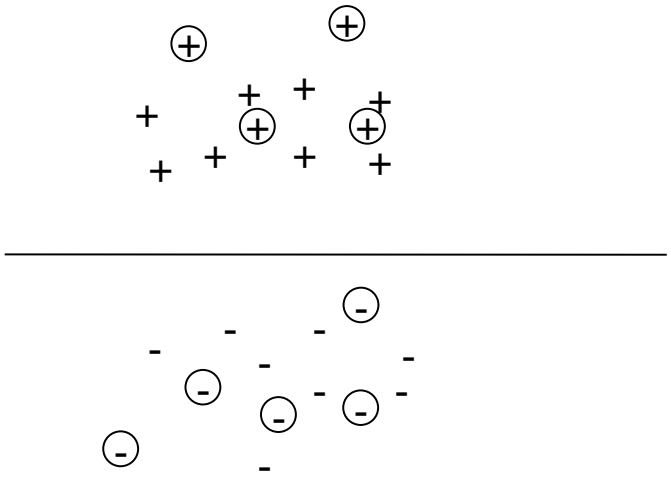
Đồng - huấn luyện (tiếp)

Algorithm co-training(L, U)
1 **repeat**
2 Xây dựng bộ phân loại f_1 sử dụng L dựa trên tập thuộc tính X_1
3 Xây dựng bộ phân loại f_2 sử dụng L dựa trên tập thuộc tính X_2
4 Dùng f_1 để phân loại các ví dụ trong U, với mỗi lớp c_i, lựa chọn n_i ví dụ mà f_1 tự tin nhất và thêm vào L.
5 Dùng f_2 để phân loại các ví dụ trong U, với mỗi lớp c_i, lựa chọn n_i ví dụ mà f_2 tự tin nhất và thêm vào L.
6 **until** U rỗng (hoặc sau một số vòng lặp nhất định)

Đồng - huấn luyện



(A) DL phân loại bởi f_1 trên U



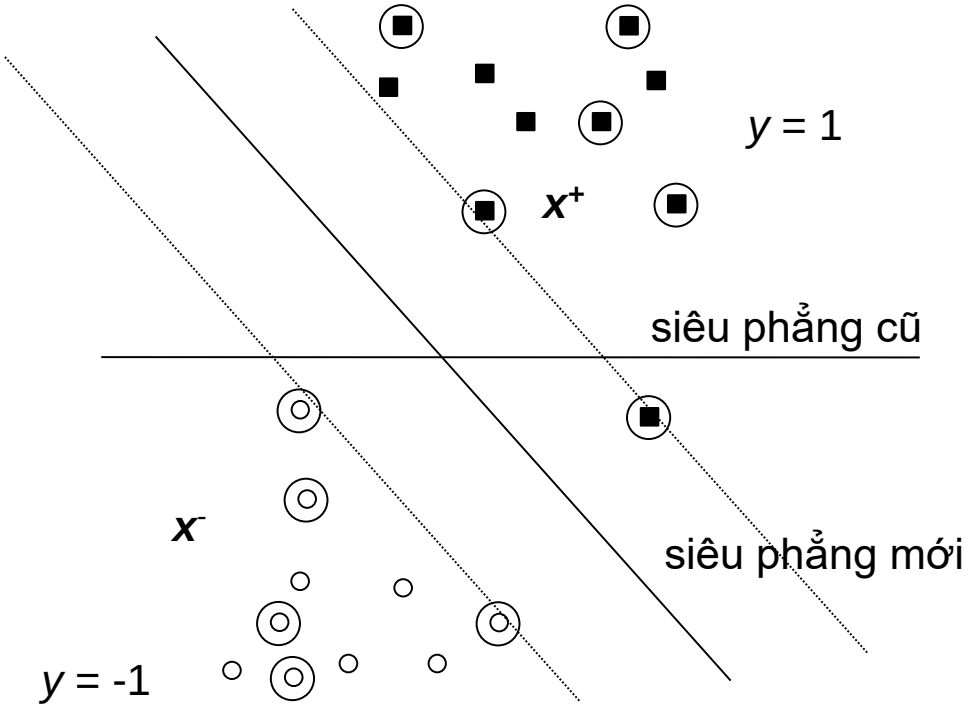
(B) DL bổ sung bởi f_1 đ/v f_2

10.3 Tự - huấn luyện

- Bộ phân loại *f* được xây dựng trên tập L
- Bộ phân loại *f* được dùng để phân loại các ví dụ trong tập U
- Các ví dụ có độ tự tin cao nhất được thêm vào tập L
- Quá trình lặp lại đến khi kết thúc (vd: U rỗng)

10.4 Suy diễn trên SVM

- Lựa chọn siêu phẳng nhằm cực đại hóa biên dựa trên các ví dụ không có nhãn



10.5 P^2 dựa trên đồ thị

- Từ *L* và *U*, xây dựng đồ thị gồm các đỉnh là các ví dụ, các cạnh có trọng số là độ tương đồng giữa các ví dụ.
  - Từ một đỉnh, chọn k hàng xóm lân cận nhất
  - Chọn độ tương đồng trên một ngưỡng tối thiểu
  - Sử dụng đồ thị kết nối đầy đủ với độ tương đồng theo hàm mũ
- Gán nhãn cho các ví dụ trong *U* dựa trên các ví dụ có nhãn trong *L* sao cho các đỉnh tương đồng nhau có cùng nhãn

P^2 dựa trên đồ thị (tiếp)

- Mincut: Các đỉnh có nhãn thuộc L được gán giá trị {0,1} tùy theo lớp positive hay negative; các cạnh được đánh trọng số w_{ij}; tìm cách gán nhãn cho các đỉnh chưa có nhãn thuộc U sao cho $\Sigma_{(i,j) \in E} w_{ij}|v_i - v_j|$ nhỏ nhất
- Tìm phép chia tập đỉnh V thành hai tập V_+ và V_- không giao nhau sao cho tổng trọng số các cạnh nối hai tập là nhỏ nhất. V_+ chứa các đỉnh có nhãn positive thuộc L và các đỉnh thuộc U; V_- chứa các đỉnh có nhãn negative thuộc L và các đỉnh thuộc U
- Thuật toán luồng cực đại với độ phức tạp tính toán $O(|V|^3)$

P^2 dựa trên đồ thị (tiếp)

- Trường Gaussian: Tối thiểu hóa $\Sigma_{(i,j) \in E} w_{ij}|v_{i-} v_j|^2$ với các định có giá trị thuộc [0, 1]
- Đồ thị phổ: Tối thiểu hóa cut(V_+, V_-) / ($|V_+||V_-|$) nhằm cân bằng số lượng đỉnh của hai tập do mincut thường có xu hướng chọn hai tập mất cân bằng

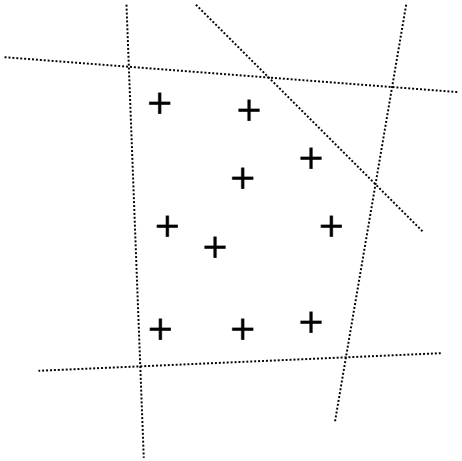
11. Học PU

- Trong một số ứng dụng người dùng chỉ quan tâm đến một lớp văn bản (positive) và không quan tâm đến các văn bản khác (negative)
- Cho tập P gồm các văn bản positive và tập U gồm các văn bản chưa có nhãn (bao gồm cả văn bản positive và negative), cần xây dựng một bộ phân loại cho phép xác định các văn bản positive

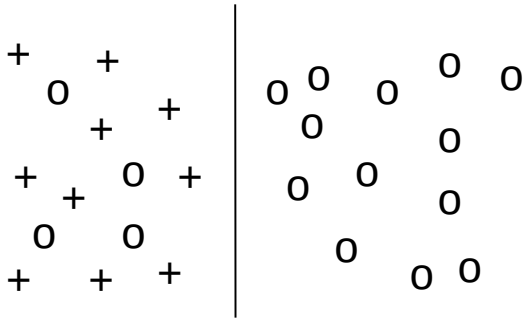
11.1 Ứng dụng của học PU

- VD: Cần xây dựng một CSDL gồm các công trình khoa học về lĩnh vực khai phá DL.
  - Đầu tiên, sử dụng các công trình thuộc hội nghị, tạp chí về khai phá DL làm các văn bản positive
  - Tiếp đó, tìm các công trình về khai phá DL trong các hội nghị và tạp chí về CSDL và trí tuệ nhân tạo
- Học với nhiều nguồn DL không có nhãn: vd, tìm các trang web về máy in
  - Tìm các trang positive từ trang amazon.com
  - Sử dụng học PU để tìm các trang positive ở các trang khác
- Học với DL negative không đáng tin cậy
- Mở rộng tập DL
- Covariate shift

11.2 Nền tảng lý thuyết



Phân loại chỉ dựa trên ví dụ positive



Phân loại dựa trên ví dụ positive và negative

Học PU (tiếp)

- $(\boldsymbol{x}_i, y_i)$ là các biến ngẫu nhiên được lấy từ phân phối xác suất $D_{(\boldsymbol{x}i, yi)}$, $y \in \{1, -1\}$ là biến ngẫu nhiên có điều kiện mà ta cần ước lược khi biết $\boldsymbol{x}$, $D_{\boldsymbol{x}|y=1}$ là phân phối điều kiện mà các ví dụ positive được sinh ra, $D_{\mathbf{x}}$ là phân phối biên sinh ra các ví dụ không có nhãn
- Mục tiêu là xây dựng một bộ phân loại f để phân loại văn bản positive và negative, bộ phân loại có xác suất sinh ra lỗi là nhỏ nhất $\Pr(f(\boldsymbol{x}) \neq y)$

Học PU (tiếp)

$\Pr(f(\boldsymbol{x}) \neq y) = \Pr(f(\boldsymbol{x}) = 1 \text{ và } y = -1) + \Pr(f(\boldsymbol{x}) = -1 \text{ và } y = 1)$

Ta có:

$\Pr(f(\boldsymbol{x}) = 1 \text{ và } y = -1) = \Pr(f(\boldsymbol{x}) = 1) - \Pr(f(\boldsymbol{x}) = 1 \text{ và } y = 1)$

$= \Pr(f(\boldsymbol{x}) = 1) - (\Pr(y = 1) - \Pr(f(\boldsymbol{x}) = -1 \text{ và } y = 1))$

Suy ra:

$\Pr(f(\boldsymbol{x}) \neq y) = \Pr(f(\boldsymbol{x}) = 1) – \Pr(y = 1) + 2\Pr(f(\boldsymbol{x}) = -1 | y = 1)\Pr(y = 1)$

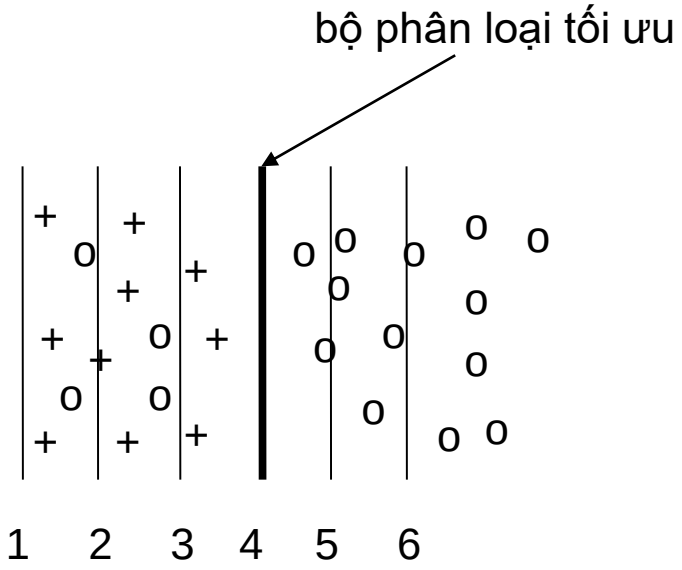
Do $\Pr(y = 1) = \text{const}$, tối thiểu hóa:

$\Pr(f(\boldsymbol{x}) = 1) + 2\Pr(f(\boldsymbol{x}) = -1 | y = 1)\Pr(y = 1)$

Nếu $\Pr(f(\boldsymbol{x}) = -1 | y = 1) <<$, xấp xỉ với tối thiểu hóa $\Pr(f(\boldsymbol{x}) = 1)$

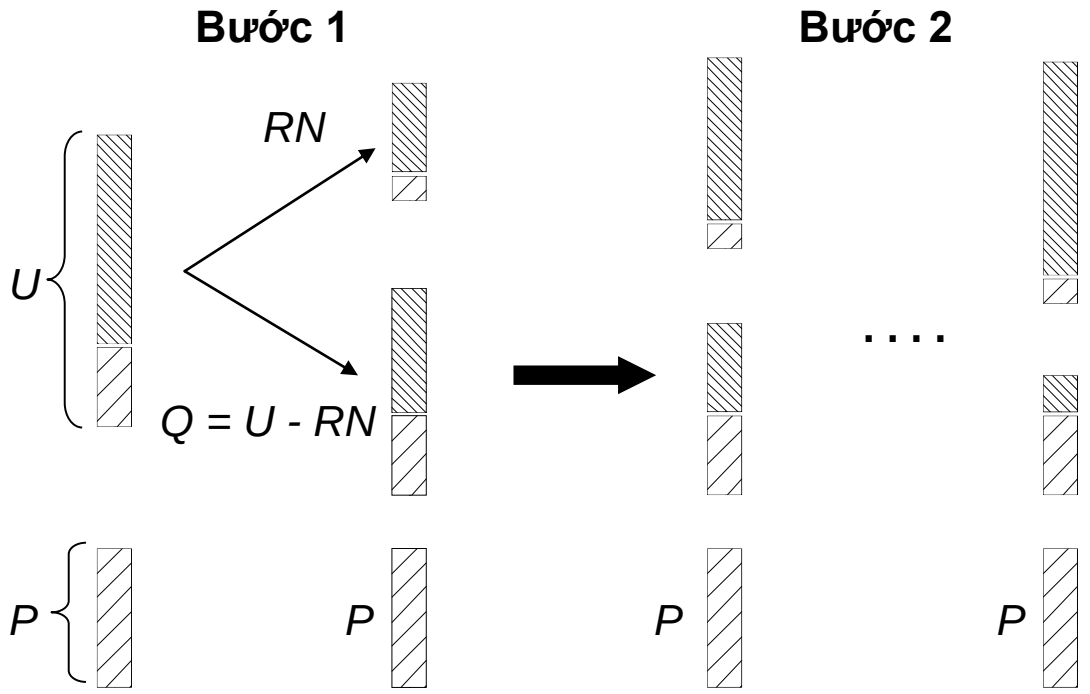
xấp xỉ với tối thiểu hóa $\Pr_U(f(\boldsymbol{x}) = 1)$ với đ/k $\Pr_P(f(\boldsymbol{x}) = 1) \geq r$ (recall)

Học PU (tiếp)



11.3 Xây dựng bộ phân loại: 2 - bước

- Bước 1: Xác định tập các văn bản negative đáng tin cậy *RN* từ *U*
- Bước 2: Xây dựng bộ phân loại từ P, RN, và U-RN



Bước 1

- Kỹ thuật do thám:

1. Chọn ngẫu nhiên một tập S từ P và đưa vào U (line 2-3). Các văn bản trong S cho phép tìm ra các văn bản positive trong U
2. Xây dựng bộ phân loại NB với P_S làm tập positive và U_s làm tập negative (line 3-7). Dùng bộ phân loại NB xác định các xác suất $\Pr(1|d)$ với các văn bản trong U_s
3. Sử dụng xác suất $\Pr(1|d)$ với các văn bản trong S để tìm ngưỡng t. Các văn bản có xác suất $\Pr(1|d) < t$ được đưa vào tập RN (line 10-14)

Bước 1 (tiếp)

Algorithm spy(P, U)
1 $RN \leftarrow \emptyset$;
2 $S \leftarrow$ sample(P, s%);
3 $U_s \leftarrow U \cup S$;
4 $P_s \leftarrow P - S$;
5 Gán mỗi văn bản trong P_s vào lớp 1;
6 Gãn mỗi văn bản trong U_s vào lớp -1;
7 NB(U_s, P_s); // xây dựng mô hình NB
8 phân loại các văn bản trong U_s sử dụng bộ phân loại NB;
9 Xác định ngưỡng xác suất t dựa trên S;
10 **for** mỗi văn bản $d \in U_s$ **do**
11 **if** xác suất Pr(1|d) < t **then**
12 $RN \leftarrow RN \cup \{d\}$;
13 **endif**
14 **endfor**

Bước 1 (tiếp)

- Kĩ thuật Cosine-Rocchio:
- Bước 1: Xác định tập negative tiềm năng PN
  - Biểu diễn các văn bản thành véc-tơ theo TF-IDF
  - Xây dựng véc-tơ đặc trưng $\boldsymbol{v}_P$ cho tập P. Tính độ tương đồng cosine giữa các văn bản trong P với $\boldsymbol{v}_P$ và sắp xếp theo thứ tự giảm dần. Một ngưỡng được xác định để lấy hết các văn bản positive ẩn trong U và lấy được tối thiểu các văn bản negative.
- Bước 2: Xây dựng bộ phân loại *Rocchio f* dựa trên P và PN. Các văn bản trong U được phân loại thành negative bởi f được đưa vào tập RN

Bước 1 (tiếp)

Algorithm CR(P, U)

1 $PN = \emptyset$; $RN = \emptyset$;

2 Biểu diễn mỗi văn bản $\boldsymbol{d} \in P$ và U thành véc-tơ theo TF-IDF;

3 $\mathbf{v}_P = \frac{1}{|P|}\sum_{\mathbf{d}\in P}\frac{\mathbf{d}}{\|\mathbf{d}\|}$;

4 Tính cos($\boldsymbol{v}_p$, $\boldsymbol{d}$) cho mỗi văn bản $\boldsymbol{d} \in P$;

5 Sắp xếp tất cả các văn bản $\boldsymbol{d} \in P$ dựa trên cos($\boldsymbol{v}_p$, $\boldsymbol{d}$) theo thứ tự giảm dần;

6 ω = cos($\boldsymbol{v}_p$, $\boldsymbol{d}$) với $\boldsymbol{d}$ được xếp ở vị trí (1- l)*|P|;

7 **for** mỗi văn bản $\boldsymbol{d} \in U$ **do**

8 **if** cos($\boldsymbol{v}_p$, $\boldsymbol{d}$) < ω **then**

9 $PN = PN \cup \{\boldsymbol{d}\}$

10 $\mathbf{c}_P = \frac{\alpha}{|P|}\sum_{\mathbf{d}\in P}\frac{\mathbf{d}}{\|\mathbf{d}\|} - \frac{\beta}{|PN|}\sum_{\mathbf{d}\in PN}\frac{\mathbf{d}}{\|\mathbf{d}\|}$;

11 $\mathbf{c}_{PN} = \frac{\alpha}{|PN|}\sum_{\mathbf{d}\in PN}\frac{\mathbf{d}}{\|\mathbf{d}\|} - \frac{\beta}{|P|}\sum_{\mathbf{d}\in P}\frac{\mathbf{d}}{\|\mathbf{d}\|}$;

12 **for** văn bản $\boldsymbol{d} \in U$ **do**

13 **if** cos($\boldsymbol{c}_{PN}$, $\boldsymbol{d}$) > cos($\boldsymbol{c}_P$, $\boldsymbol{d}$) **then**

14 $RN = RN \cup \{\boldsymbol{d}\}$

Bước 1 (tiếp)

- Kĩ thuật *1DNF*: Các từ trong P và U được thu thập để xây dựng từ vựng (line 1). Tập đặc trưng positive PF được xây dựng chứa các từ x/h phổ biến trong P hơn U (line 2-7). Các văn bản trong U không chứa thuộc tính nào trong PF được đưa vào tập RN (line 8-13)
- Kĩ thuật *NB*: Sử dụng một bộ phân loại *NB* để xây dựng tập RN từ U
- Kĩ thuật *Rocchio*: Sử dụng một bộ phân loại *Rocchio* để xây dựng tập RN từ U

Algorithm 1DNF(P, U)

1 Giả sử tập từ vựng $V = \{w_1, ..., w_n\}$, $w_i \in U \cup P$;
2 Khởi tạo tập thuộc tính positive $PF \leftarrow \emptyset$;
3 **for** mỗi $w_i \in V$ **do** // freq(w_i, P): số lần w_i xuất hiện trong P
4 **if** (freq(w_i, P) / |P| > freq(w_i, U) / |U|) **then**
5 $PF \leftarrow PF \cup \{w_i\}$;
6 **endif**
7 **endfor**
8 $RN \leftarrow U$;
9 **for** mỗi văn bản $d \in U$ **do**
10 **if** tồn tại w_j sao cho freq(w_j, d) > 0 and $w_j \in PF$ **then**
11 $RN \leftarrow RN - \{d\}$
12 **endif**
13 **endfor**

Algorithm NB(P, U)

1 Gán các văn bản trong P nhãn lớp 1;
2 Gán các văn bản trong U nhãn lớp -1;
3 Xây dựng bộ phân loại NB sử dụng P và U;
4 Sử dụng bộ phân loại để phân loại các văn bản trong U. Các văn bản được phân loại negative được đưa vào tập RN

Bước 2

- EM + NB: Bước expectation tính toán các xác suất nhãn của các văn bản trong U – RN. Bước maximization ước lượng lại tham số của bộ phân loại NB
- SVM: Tại mỗi bước lặp, bộ phân loại SVM *f* được xây dựng từ *P* và *RN*. Bộ phân loại f được sử dụng để phân loại các văn bản trong *Q*. Các văn bản được phân loại negative được loại khỏi *Q* và bổ sung vào *RN*. Quá trình lặp kết thúc khi không còn văn bản nào trong được phân loại negative

Bước 2 (tiếp)

Algorithm EM(*P*, *U*, *RN*)
1 Mỗi văn bản trong *P* được gán nhãn lớp 1;
2 Mỗi văn bản trong *RN* được gán nhãn lớp -1;
3 Học một bộ phân loại *NB f* từ *P* và *RN*
4 **repeat**
5 **for** mỗi văn bản d_i trong *U* - *RN* **do**
6 Sử dụng bộ phân loại *f* hiện tại để tính $\Pr(c_j | d_i)$
7 **endfor**
8 học bộ phân loại *NB* mới *f* từ *P*, *RN* và *U* - *RN* bằng cách tính $\Pr(c_j)$ và $\Pr(w_t | c_j)$
9 **until** các tham số của bộ phân loại ổn định

Bước 2 (tiếp)

Algorithm I-SVM(*P*, *RN*, *Q*)
1 Mỗi văn bản trong *P* được gán nhãn lớp 1;
2 Mỗi văn bản trong *RN* được gán nhãn lớp –1;
3 **loop**
4 Dùng *P* và *RN* để huấn luyện bộ phân loại SVM *f*;
5 Phân loại *Q* sử dụng *f*;
6 *W* là tập các văn bản trong *Q* được phân loại là negative;
7 **if** $W = \emptyset$ **then** exit-loop // hội tụ
8 **else** $Q \leftarrow Q - W$;
9 $RN \leftarrow RN \cup W$;
10 **endif**

11.4 Xây dựng bộ phân loại: SVM

- Cho tập DL huấn luyện $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2)\ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$ trong đó $\boldsymbol{x}_i$ là các véc-tơ và $y_i \in \{1, -1\}$. G/s k ví dụ đầu $\in P$ có nhãn positive ($y = 1$), các ví dụ sau $\in U$ được coi như có nhãn negative ($y = -1$)
- TH1: Tập P không chứa lỗi, theo lý thuyết khi số mẫu đủ lớn, có thể thu được bộ phân loại đủ tốt nếu tối thiểu hóa các văn bản không có nhãn được phân loại thành positive trong khi ràng buộc số ví dụ positive được phân loại tốt

Tối thiểu hóa: $\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\sum_{i=k}^{n} \xi_i$

Với ràng buộc: $<\boldsymbol{w} \cdot \boldsymbol{x}_i> + b \geq 1,\ i = 1, 2, \ldots, k\text{-}1$

$-1(<\boldsymbol{w} \cdot \boldsymbol{x}_i> + b) \geq 1 - \xi_i,\ i = k, k\text{+}1, \ldots n$

$\xi_i \geq 0,\ i = k, k\text{+}1, \ldots, n$

Xây dựng bộ phân loại SVM (tiếp)

- TH2: Tập P chứa ví dụ negative (do DL có nhiễu)

Tối thiểu hóa: $\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C_+ \sum_{i=1}^{k-1} \xi_i + C_- \sum_{i=k}^{n} \xi_i$

Với ràng buộc: $y_i(<\boldsymbol{w} \cdot \boldsymbol{x}_i> + b) \geq 1 - \xi_i,\ i = k, k+1, \ldots n$

$\xi_i \geq 0,\ i = k, k+1, \ldots, n$

trong đó C_+ và C_- là các trọng số của lỗi positive và negative tương ứng được xác định dựa trên tập validation theo độ đo

$$\frac{r^2}{\Pr(f(\mathbf{x}) = 1)}$$

Xây dựng bộ phân loại SVM (tiếp)

r = Pr(f(x) = 1| y = 1)

p = Pr(y = 1| f(**x**) = 1)

Ta có:

Pr(f(***x***) = 1| y = 1)Pr(y = 1) = Pr(y = 1| f(***x***) = 1) Pr(f(***x***) = 1)

Suy ra:

$$\frac{r}{Pr(f(\boldsymbol{x}) = 1)} = \frac{p}{Pr(y = 1)} \Rightarrow \frac{r^2}{Pr(f(\boldsymbol{x}) = 1)} = \frac{pr}{Pr(y = 1)}$$

có t/c tương tự F-score

11.5 Xây dựng bộ phân loại: Ước lượng xác suất

- Cho ví dụ $\boldsymbol{x}$ và nhãn $y \in \{1, -1\}$, cho $s = 1$ nếu $\boldsymbol{x}$ là ví dụ được gán nhãn và $s = 0$ nếu $\boldsymbol{x}$ không có nhãn, ta có $\Pr(s = 1 | \mathbf{x}, y = -1) = 0$
- Mục tiêu: Xây dựng hàm phân loại $f(\boldsymbol{x})$ sao cho $f(\boldsymbol{x})$ gần $\Pr(y = 1 | \boldsymbol{x})$ nhất
- Giả thuyết lựa chọn ngẫu nhiên hoàn toàn: các ví dụ positive được gán nhãn được lựa chọn hoàn toàn ngẫu nhiên từ các ví dụ positive $\Pr(s = 1 | \boldsymbol{x}, y = 1) = \Pr(s = 1 | y = 1) = c$
- Nếu dùng P và U để xây dựng hàm phân loại $g(\boldsymbol{x})$ thì $g(\boldsymbol{x}) = \Pr(s = 1 | \boldsymbol{x})$, khi đó $f(\boldsymbol{x}) = g(\boldsymbol{x}) / c$

Ước lượng xác suất (tiếp)

$g(\boldsymbol{x}) = \Pr(s = 1|\ \boldsymbol{x})$
$= \Pr(y = 1 \text{ và } s = 1|\ \boldsymbol{x})$
$= \Pr(y = 1|\ \boldsymbol{x})\Pr(s = 1|\ y = 1, \boldsymbol{x})$
$= \Pr(y = 1|\ \boldsymbol{x})\Pr(s = 1|\ y = 1)$
$= f(\boldsymbol{x})\Pr(s = 1|\ y = 1)$

Ước lượng c sử dụng tập validation V với V_P là tập các ví dụ positive trong V:

$$\hat{c} = \frac{1}{|V_P|}\sum_{x \in V_P} g(x)$$

$g(\boldsymbol{x}) = \Pr(s = 1|\ \boldsymbol{x})$
$= \Pr(s = 1|\ \boldsymbol{x}, y = 1)\Pr(y = 1|\ \boldsymbol{x}) + \Pr(s = 1|\ \boldsymbol{x}, y = -1)\Pr(y = -1|\ \boldsymbol{x})$
$= \Pr(s = 1|\ \boldsymbol{x}, y = 1) \times 1 + 0 \times 0$ do $\boldsymbol{x} \in V_P$
$= \Pr(s = 1|\ y = 1)$.

Chú ý: Nếu chỉ cần xếp hạng $\boldsymbol{x}$ theo xác suất thuộc lớp positive, có thể dùng trực tiếp g

ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!

soict.hust.edu.vn/ fb.com/groups/soict