Phân loại văn bản
(Text Classification)

Nguyễn Mạnh Hiển
hiennm@tlu.edu.vn

Tác vụ phân loại

- Cho:
  - Mẫu dữ liệu $x \in X$, trong đó X là không gian mẫu.
  - Tập lớp $C = \{c_1, c_2, \ldots, c_n\}$.
- Dự đoán:
  - Lớp của x, tức là $c(x) \in C$, trong đó c(x) là hàm phân loại.

Học phân loại (learning to classify)

- ***Mẫu huấn luyện*** (training instance) là một mẫu x $\in$ X kèm theo lớp đúng c(x) đã biết của nó, ký hiệu là <x, c(x)>.
- Cho một ***tập huấn luyện*** (training set) D.
- Tìm hàm phân loại h(x) sao cho:

$$\forall < x, c(x) > \in D : h(x) = c(x)$$

- Trên thực tế, hàm h(x) tìm được có thể sai khác với c(x) trong một số trường hợp, tức là có ***sai số huấn luyện*** (training error).

  $\rightarrow$ Việc ***học/huấn luyện*** (tìm hàm h(x)) phải đảm bảo sai số huấn luyện nhỏ.

- Phải đánh giá độ chính xác phân loại của hàm h(x) tìm được trên một ***tập kiểm thử*** (test set) riêng biệt.

Ví dụ về tập dữ liệu huấn luyện

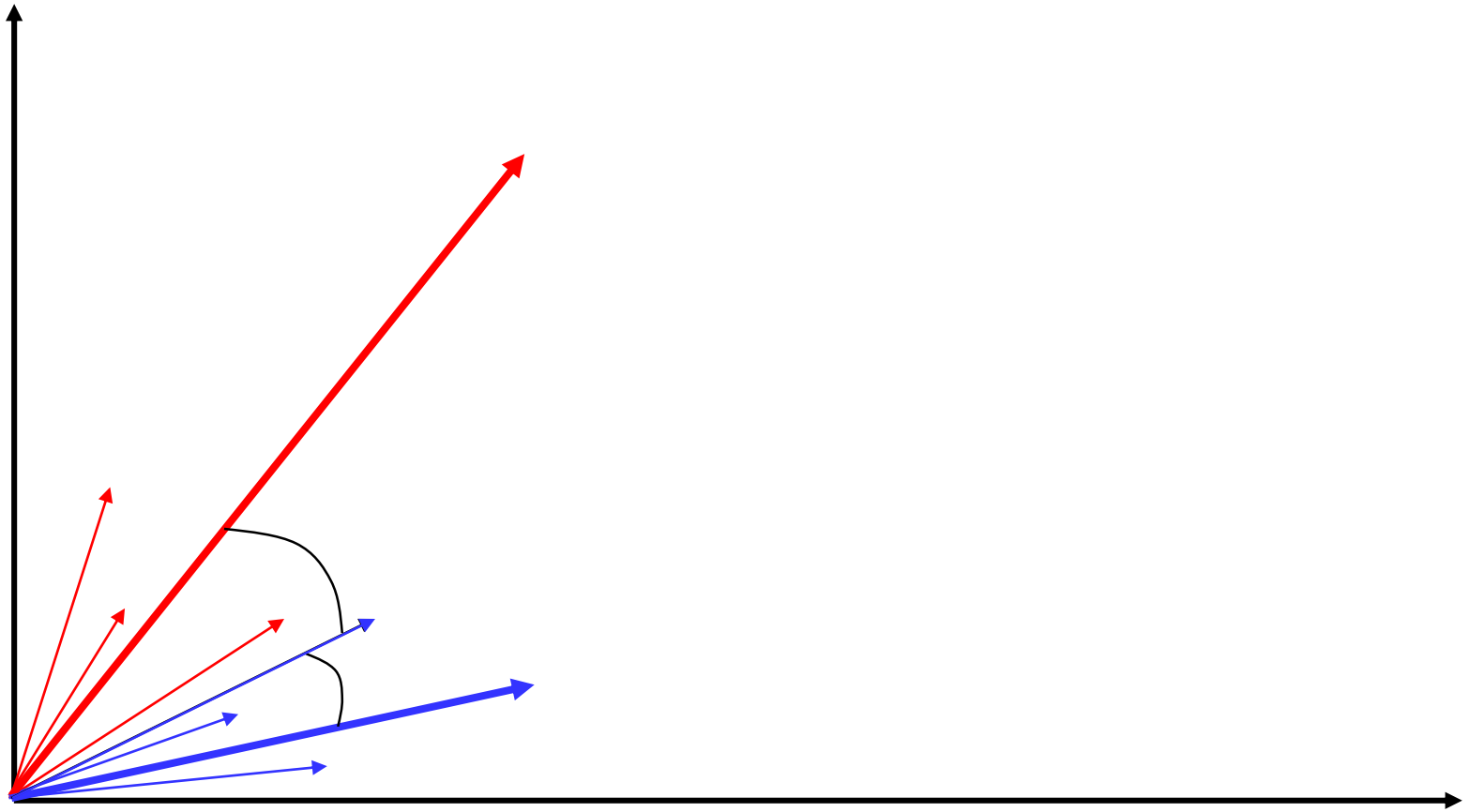
- Các thuộc tính mô tả các mẫu dữ liệu:
  - size $\in$ { small, medium, large }
  - color $\in$ { red, green, blue }
  - shape $\in$ { square, circle, triangle }
- C = { positive, negative }
- D:

Mẫu	size	color	shape	Lớp
1	small	red	circle	positive
2	large	red	circle	positive
3	small	red	triangle	negative
4	large	blue	circle	negative

Thuật toán phân loại văn bản Rocchio

- Dùng vector trọng số từ tf-idf để biểu diễn các văn bản (mỗi từ là một thuộc tính của văn bản).
- Đối với mỗi lớp văn bản, tính một vector ***nguyên mẫu*** (prototype) bằng cách cộng tất cả các vector biểu diễn các ***văn bản huấn luyện*** (training document) trong lớp văn bản đó.
  - Chú ý: Không cần tính vector trung bình vì góc giữa các véctơ không phụ thuộc chiều dài của chúng.
- Gán mỗi ***văn bản kiểm thử*** (test document) cho lớp văn bản có vector nguyên mẫu giống nó nhất theo độ đo tương tự côsin.

Minh họa thuật toán Rocchio



Thuật toán Rocchio (phần huấn luyện)

Giả sử tập lớp văn bản là $\{c_1, c_2, \ldots, c_n\}$

Với i chạy từ 1 đến n:

    Cho p_i = <0, 0, …, 0> // Khởi tạo vector nguyên mẫu

Với mỗi mẫu huấn luyện <x, c(x)> $\in$ D:

    Cho d là véctơ trọng số từ tf-idf của văn bản x

    Cho i là giá trị sao cho $c_i = c(x)$

    Cho $p_i = p_i + d$ // Cộng các vector trong lớp c_i

Thuật toán Rocchio (phần kiểm thử)

Cho văn bản kiểm thử x

Cho d là vector trọng số từ tf-idf của x

Cho m = –1 // Điểm số tương tự khởi đầu

Với i chạy từ 1 đến n:

    Cho s = CosSim(d, p_i) // Tính độ tương tự với nguyên mẫu

    Nếu s > m:

        Cho m = s

        Cho r = c_i // Cập nhật lớp có nguyên mẫu giống nhất

Trả về lớp r làm lớp dự đoán của x

Thuật toán láng giềng gần nhất (nearest neighbor)

- Chỉ cần lưu trữ lại các mẫu huấn luyện trong tập D.
- Với mẫu kiểm thử x:
  - Tính độ tương tự giữa x và tất cả các mẫu trong D.
  - Gán x cho lớp chứa mẫu giống nó nhất.
- Còn gọi là:
  - Học theo tình huống (case-based);
  - Học bằng cách ghi nhớ (memory-based);
  - Học lười (lazy learning).

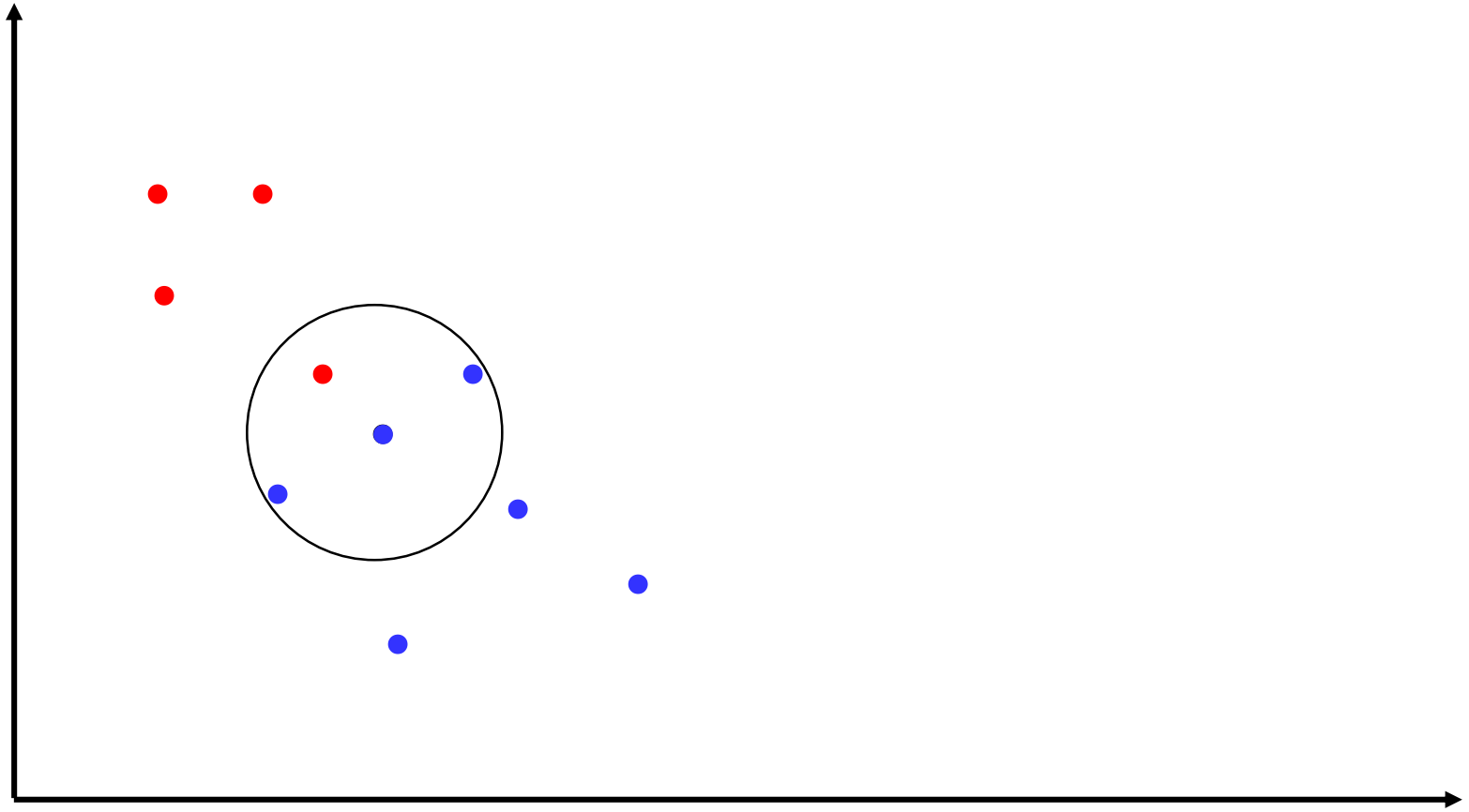
k láng giềng gần nhất (kNN)

- Chỉ dùng một mẫu gần nhất để dự đoán lớp sẽ dễ dẫn tới lỗi, vì có thể tồn tại:
  - mẫu không điển hình (nằm xa lớp của nó nhưng lại gần lớp khác);
  - nhiễu trong nhãn lớp của mẫu huấn luyện (gắn nhầm nhãn).
- Phương pháp khỏe mạnh hơn là tìm k mẫu giống nhất, và trả về lớp đa số trong k mẫu đó.
- Trong trường hợp chỉ có hai lớp, giá trị của k nên là số lẻ (ví dụ, 3 hoặc 5) để tránh số phiếu bằng nhau.

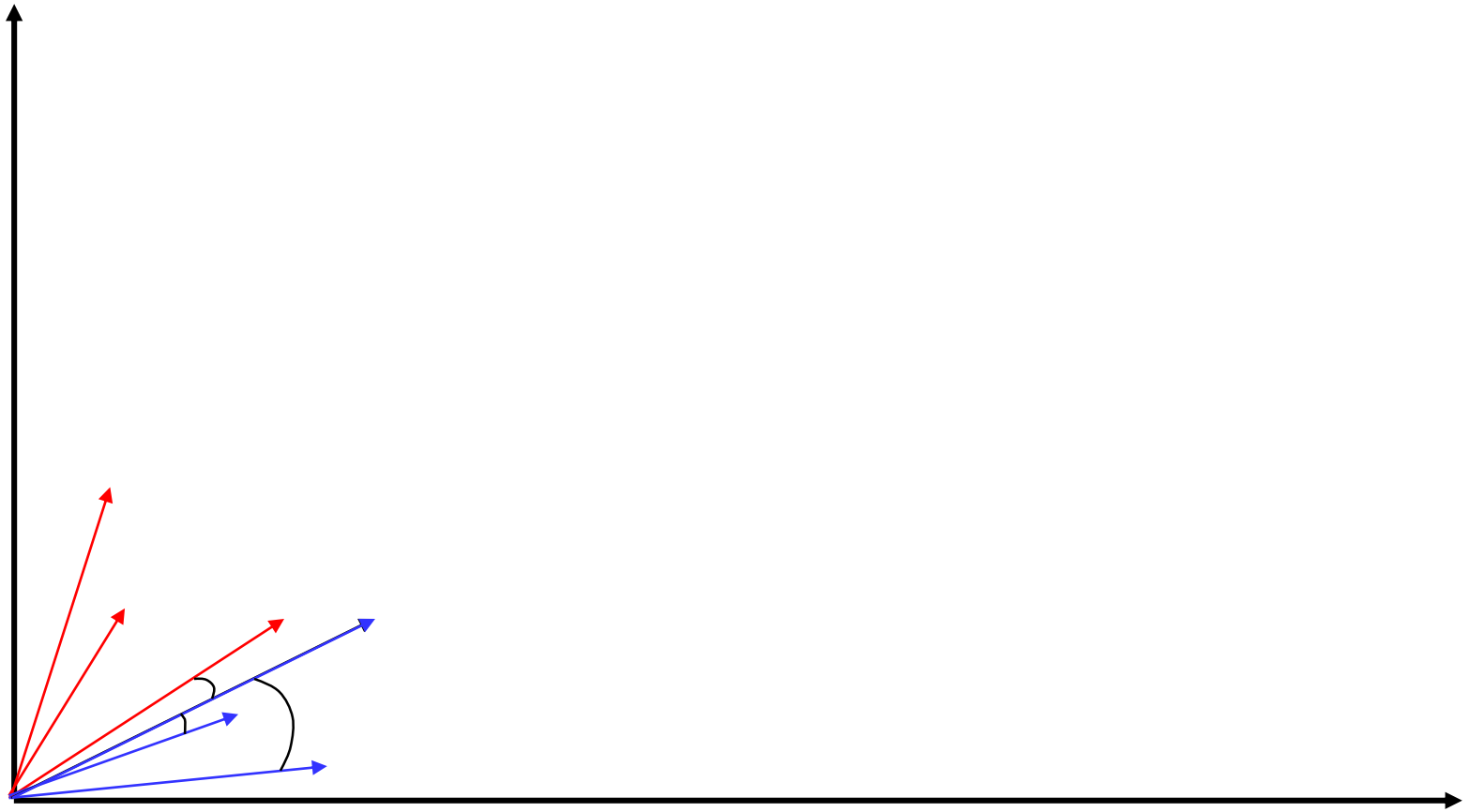
Độ đo tương tự

- Phương pháp láng giềng gần nhất phụ thuộc vào độ tương tự hoặc khoảng cách giữa các mẫu.
- Với không gian mẫu liên tục m chiều, đơn giản nhất là khoảng cách Ơclít.
- Với không gian mẫu nhị phân m chiều, đơn giản nhất là khoảng cách Hamming (số bit khác nhau).
- Với dữ liệu văn bản, độ tương tự côsin giữa các vector trọng số tf-idf thường hiệu quả nhất.

Minh họa 3 láng giềng gần nhất dùng khoảng cách Ơclít



Minh họa 3 láng giềng gần nhất cho dữ liệu văn bản dùng độ đo côsin



k láng giềng gần nhất cho dữ liệu văn bản

Huấn luyện:

Với mỗi mẫu huấn luyện <x, c(x)> $\in$ D:

Tính vector trọng số tf-idf d_x cho văn bản x

Mẫu kiểm thử y:

Tính vector trọng số tf-idf d cho văn bản y

Với mỗi mẫu huấn luyện <x, c(x)> $\in$ D:

Tính $s_x = CosSim(d, d_x)$

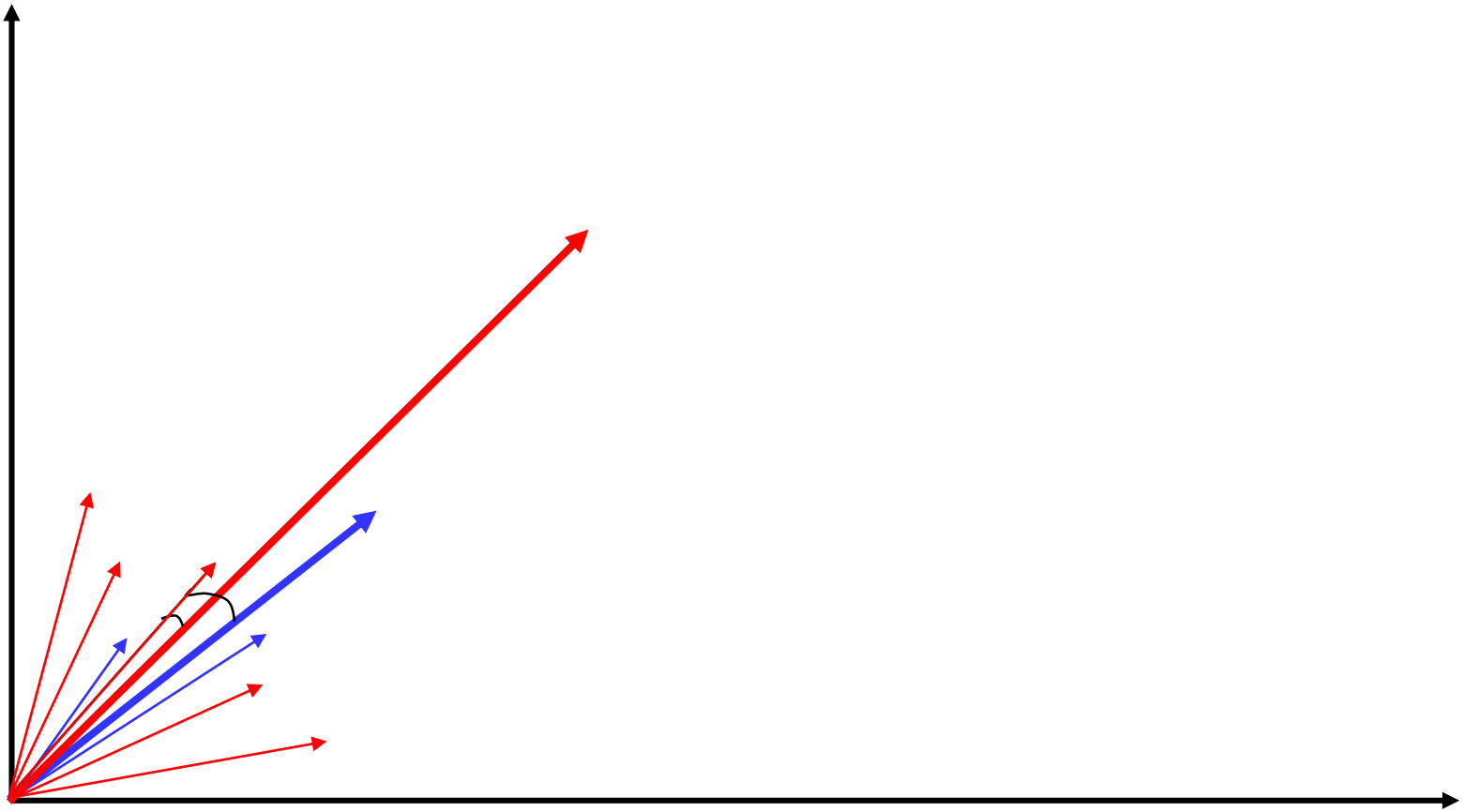
Sắp xếp các mẫu x trong D theo thứ tự giảm dần của s_x

Gọi N là tập k mẫu đầu tiên // Lấy k láng giềng gần nhất

Trả về lớp đa số trong tập N

Vấn đề với thuật toán Rocchio

Khi một lớp có nhiều cụm nằm rải rác (không tập trung một chỗ):



So sánh với 3 láng giềng gần nhất

Láng giềng gần nhất có thể giải quyết vấn đề đó tốt hơn: