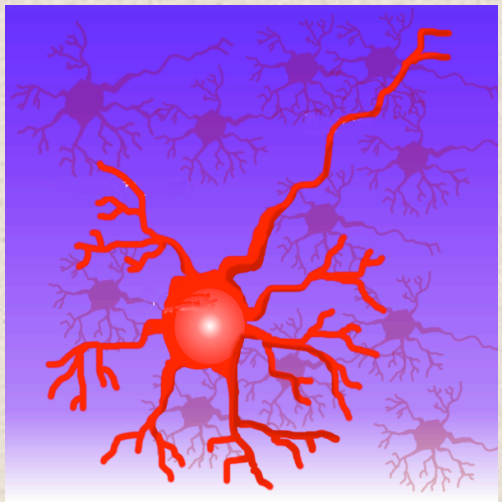
Corso di intelligenza artificiale 2

# Text Categorization

## metodologia di apprendimento mediante approcci statistici e reti neuronali artificiali



Claudio Biancalana

# Contatti

**Claudio Biancalana**

**claudio.biancalana@dia.uniroma3.it**

**www.dia.uniroma3.it/~biancal/**

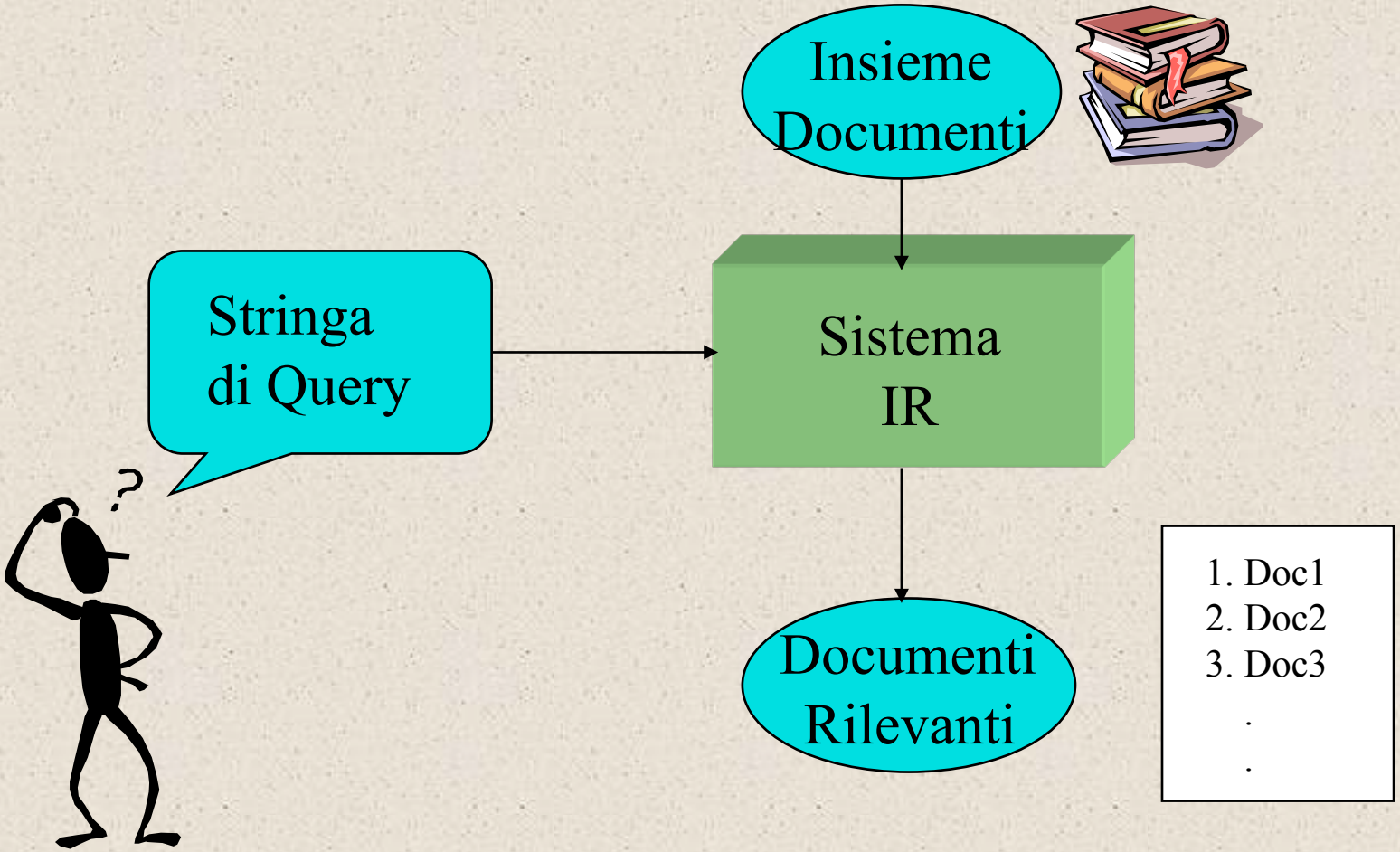
# Information Retrieval (IR)

- Inidicizzazione e ricerca di documenti testuali
- Ricerca di pagine sul web
- Obiettivo principale è la ricerca efficace di documenti rilevanti
- Obiettivo secondario è la ricerca efficiente tra un inisieme vasto di documenti

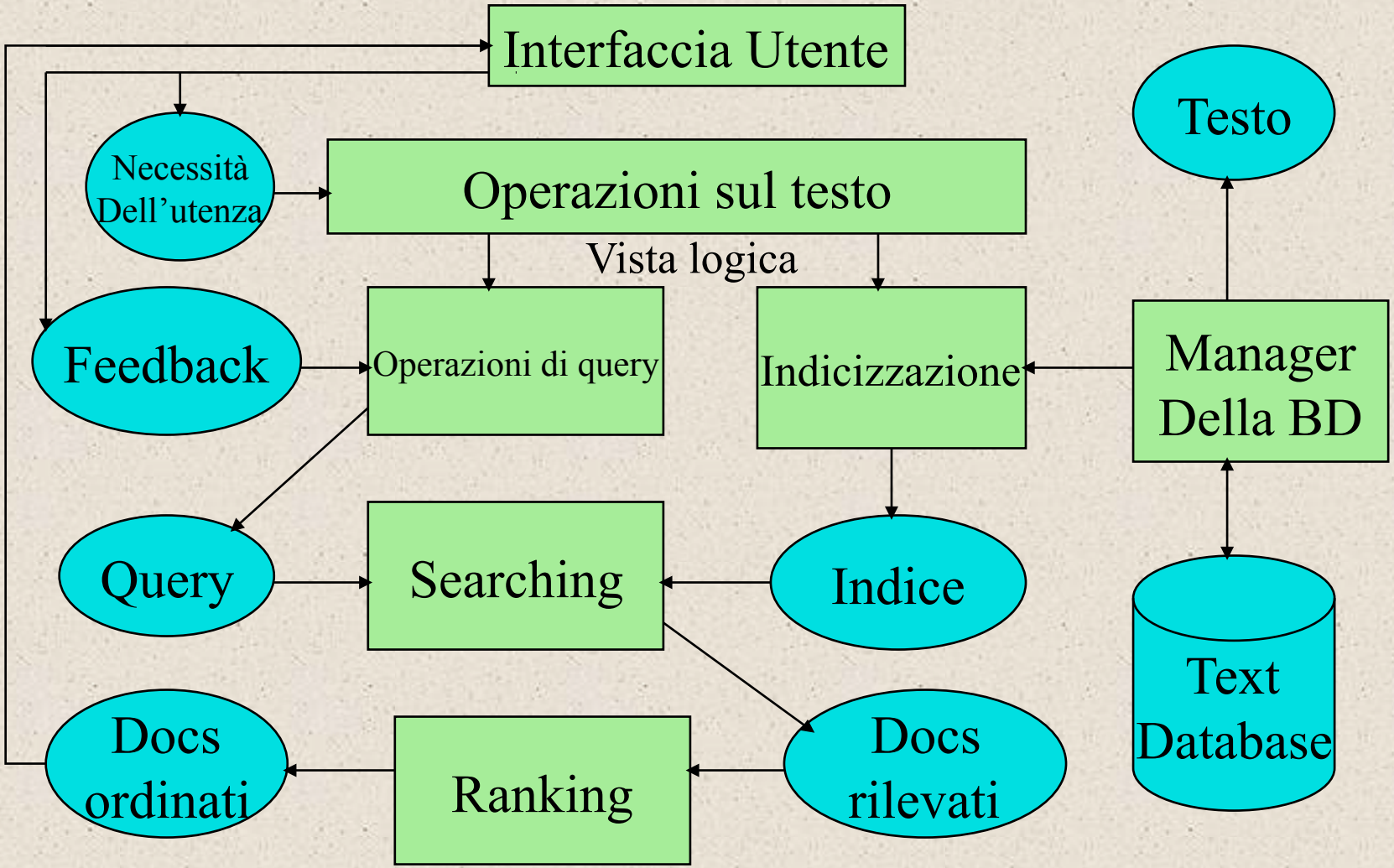
# Obiettivi IR

- Input:
  - Un insieme di documenti testuali scritti in linguaggio naturale.
  - Una stringa rappresentante una query.
- Output:
  - Un insieme di documenti ordinati in base alla loro rilevanza alla query.

# Sistema IR


Insieme
Documenti
Stringa
di Query
Sistema
IR
Documenti
Rilevanti
1. Doc1
2. Doc2
3. Doc3
.
.

# Architettura di un sistema IR



# Categorizzazione

- Input:
  - Una descrizione di una istanza, $x \in X$, dove X è l'*istanza linguaggio* o *spazio dell'istanza*.
  - Un numero fissato di categorie: $C = \{c_1, c_2, \ldots c_n\}$
- Output:
  - La categoria di $x$: $c(x) \in C$, dove $c(x)$ è una funzione di catgorizzazione che ha come dominio $X$ e come codominio $C$.

# Imparare per categorizzare

- Un esempio di apprendimento per una istanza $x \in X$, accoppiata con la sua categoria $c(x)$:

  $<x, c(x)>$ con una sconosciuta funzione di categorizzazione, $c$.
- Sia dato un insieme di esempi di apprendimento, $D$.
- Trovare una ipotizzata funzione di categorizzazione, $h(x)$, tale che:

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

*Consistenza*

# Category Learning Problem (esempio)

- Istanza di linguaggio: <grandezza, colore, forma>
  - grandezza ∈ {small, medium, large}
  - colore ∈ {rosso, blu, verde}
  - forma ∈ {quadrato, cerchio, triangolo}
- $C$ = {positivo, negativo}
- $D$:

| Esempio | Grand. | Colore | Forma | Categoria |
|---|---|---|---|---|
| 1 | small | rosso | cerchio | positivo |
| 2 | large | rosso | cerchio | positivo |
| 3 | small | rosso | triangolo | negativo |
| 4 | large | blu | cerchio | negativo |

# Text Categorization

- Assegnare documenti ad un insieme fissato di categorie.
- Applicazioni:
  - Pagine web
    - Recommending
    - Classificazioni (Yahoo!)
  - Messaggi su Newsgroup
    - Recommending
    - Filtraggio spam
  - Articoli
    - Giornali personalizzati
  - Email
    - Routing
    - Prioritizing
    - Folderizing
    - Filtraggio spam

# Algoritmi di apprendimento

- Lo sviluppo “manuale” di una funzione di text categorization è difficile.
- Algoritmi per l’apprendimento:
  - **Bayesian (naïve)**
  - Reti neuronali
  - **Relevance Feedback (Rocchio)**
  - Nearest Neighbor (case based)
  - **Support Vector Machines (SVM)**
  - Basato su regole

# Rappresentazione dei documenti

- Una collezione di n documenti può essere rappresentata nel modello come una matrice dei termini del documento.
- Un valore nella matrice corrisponde al "peso" di un termine nel documento; zero significa che il termine non è significativo nel documento o più semplicemente non è presente nel documento.

$$\begin{array}{c|cccc} & T_1 & T_2 & .... & T_t \\ D_1 & w_{11} & w_{21} & \dots & w_{t1} \\ D_2 & w_{12} & w_{22} & \dots & w_{t2} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ D_n & w_{1n} & w_{2n} & \dots & w_{tn} \end{array}$$

# TF*IDF

- Metodologia di rappresentazione dei pesi del vettore per il pattern recognition

TF = frequenza del termine nel documento

IDF = log (N/n)

N= numero totale di documenti nella collezione di training

n = numero di documenti che contengono il termine nella collezione di training

Per ogni termine abbiamo TF*IDF

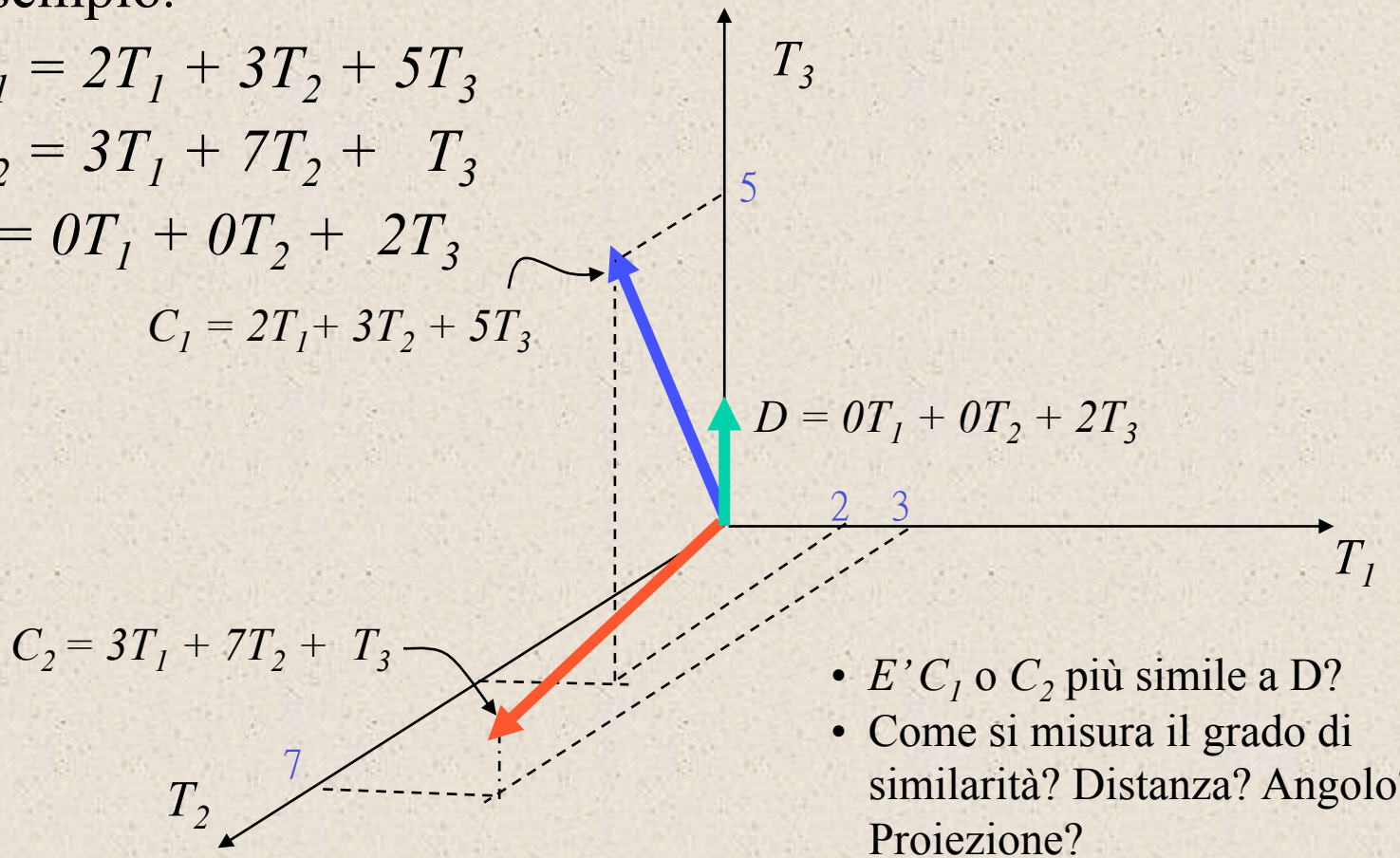
# Rappresentazione grafica

Esempio:

$C_1 = 2T_1 + 3T_2 + 5T_3$

$C_2 = 3T_1 + 7T_2 + T_3$

$D = 0T_1 + 0T_2 + 2T_3$




- *E' $C_1$* o $C_2$ più simile a D?
- Come si misura il grado di similarità? Distanza? Angolo? Proiezione?

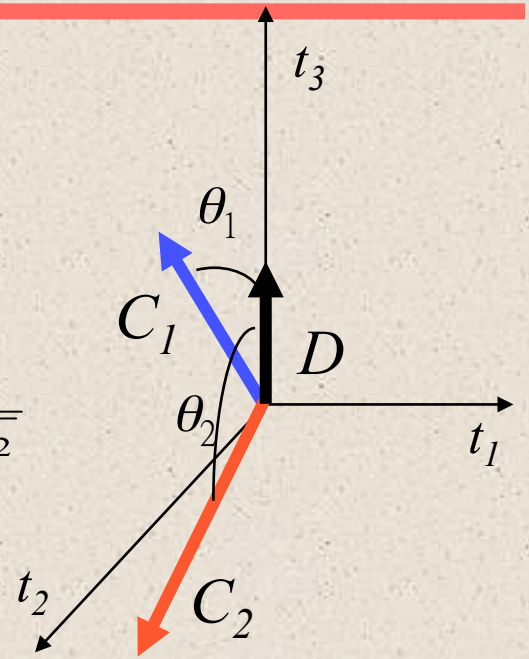
# Algoritmo rocchio

- Usiamo lo standard di indicizzazione TF/IDF per rapprensentare in forma vettoriale i documenti di testo (normalizzati secondo la frequenza massima di un termine)
- Per ogni categoria, viene elaborato un vettore “Prototipo” dalla somma dei vettori di training nella categoria
- Assegnamo il documento di test alla categoria col vettore “prototipo” più vicino mediante la regola di similarità del coseno.

# Regole di similarità del coseno

- Si misura il coseno dell'angolo fra due vettori…

$$\text{CosSim}(c_j, d) = \frac{c_j \cdot d}{|c_j| \cdot |d|} = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}$$




$C_1 = 2T_1 + 3T_2 + 5T_3$ $\quad \text{CosSim}(C_1, D) = 10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$

$C_2 = 3T_1 + 7T_2 + 1T_3$ $\quad \text{CosSim}(C_2, D) = 2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$

$D = 0T_1 + 0T_2 + 2T_3$

# Algoritmo Rocchio (apprendimento)

Sia l'insieme delle categorie $\{c_1, c_2, \ldots c_n\}$

For $i$ from 1 to $n$ let $\mathbf{p}_i = <0, 0, \ldots, 0>$ (*inizializzione*)

For each esempio di training $<x, c(x)> \in D$

Let $\mathbf{d}$ = vettore TF/IDF per il doc $x$

Let $i = j$: $(c_j = c(x))$

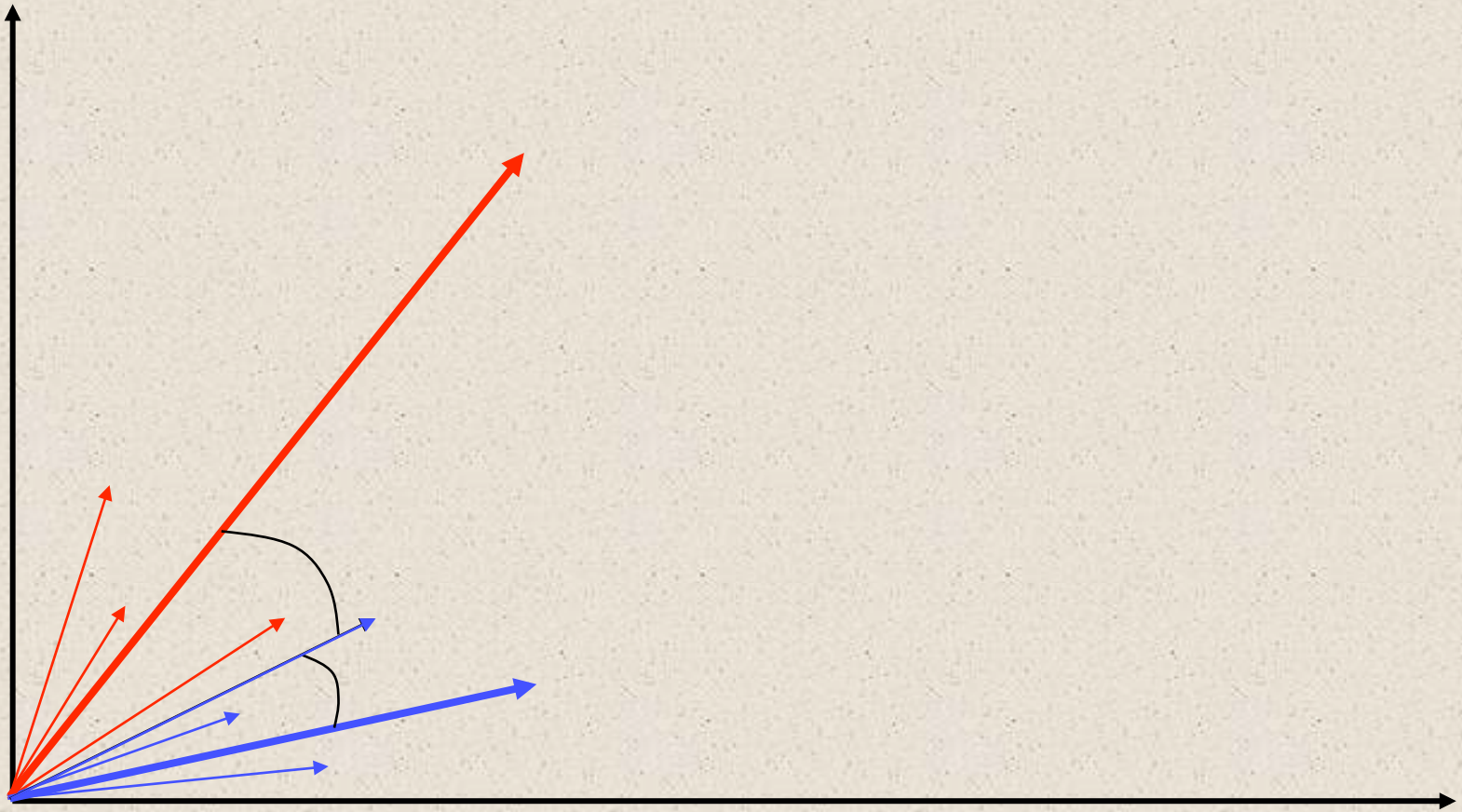
(*somma di tutti i vettori in $c_i$ per ottenere $\mathbf{p}_i$*)

Let $\mathbf{p}_i = \mathbf{p}_i + \mathbf{d}$

# Algoritmo Rocchio (Test)

Dato un documento di test $x$
Let $\mathbf{d}$ = vettore TF/IDF per $x$
Let $m = -2$ (*inizializzazione*)
For $i$ from 1 to $n$:
    (*calcola la similarità col vettore prototipo*)
    Let $s$ = cosSim($\mathbf{d}$, $\mathbf{p}_i$)
    if $s > m$
        let $m = s$
        let $r = c_i$ (*aggiorna il più simile*)
Return class $r$

# Text categorization con Rocchio



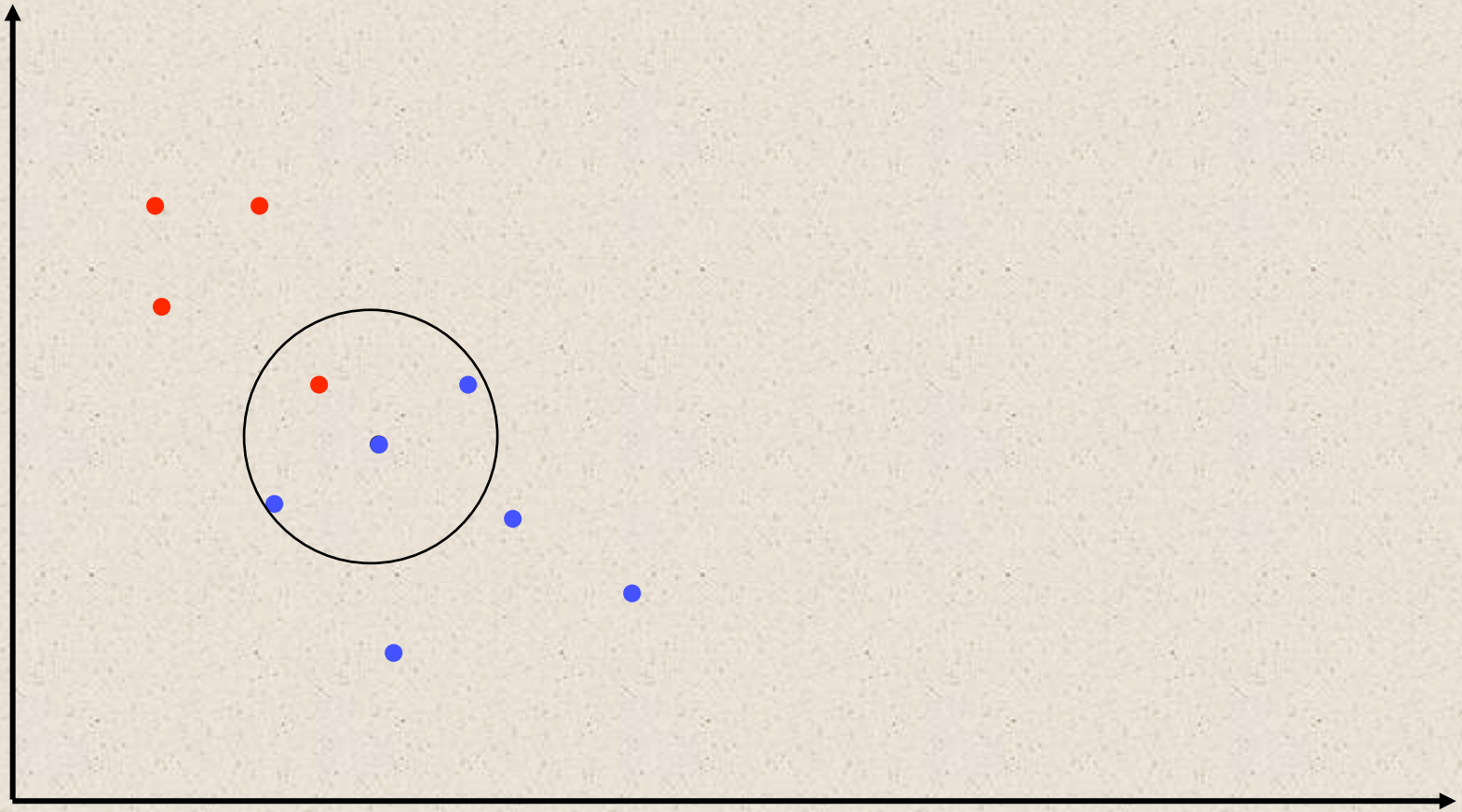
# Algoritmo di apprendimento Nearest-Neighbor

- L'apprendimento si riduce al modo di immagazzinare le rappresentazioni degli esempi di training in D.
- Test dell'istanza *x*:
  - Elabora la similarità tra x e tutti gli esempi in D.
  - Assegna ad *x* la categoria del più simile in *D*.
- Non si calcolano esplicitamente i prototipi delle categorie.
- Conosciuto anche sotto il nome di:
  - Case-based
  - Memory-based
  - Lazy learning

# Metrica per la similarità

- Nearest neighbor si basa su una metrica di similarità (o distanza)
- La più semplice per uno spazio contiunuo è la distanza euclidea.
- La più semplice per spazi d'istanza m-dimensionali binari è la distanza di Hamming
- Per i testi, la similarità basata sul coseno, per i vettori costruiti mediante indicizzazione TF-IDF, è tipicamente la più efficiente.

# 3 Nearest Neighbor Illustration (Euclidian Distance)



# K Nearest Neighbor per testi

**Training:**

For each each esempio di training $\langle x, c(x) \rangle \in D$

Calcola il corrispondente vettore TF-IDF, $\mathbf{d}_x$, per il doc $x$

**Test dell'istanza *y*:**

Calcola il vettore TF-IDF **d** per il doc y

For each $\langle x, c(x) \rangle \in D$

Let $s_x$ = cosSim(**d**, $\mathbf{d}_x$)

Ordina gli esempi, $x$, in $D$ al decrescere di $s_x$

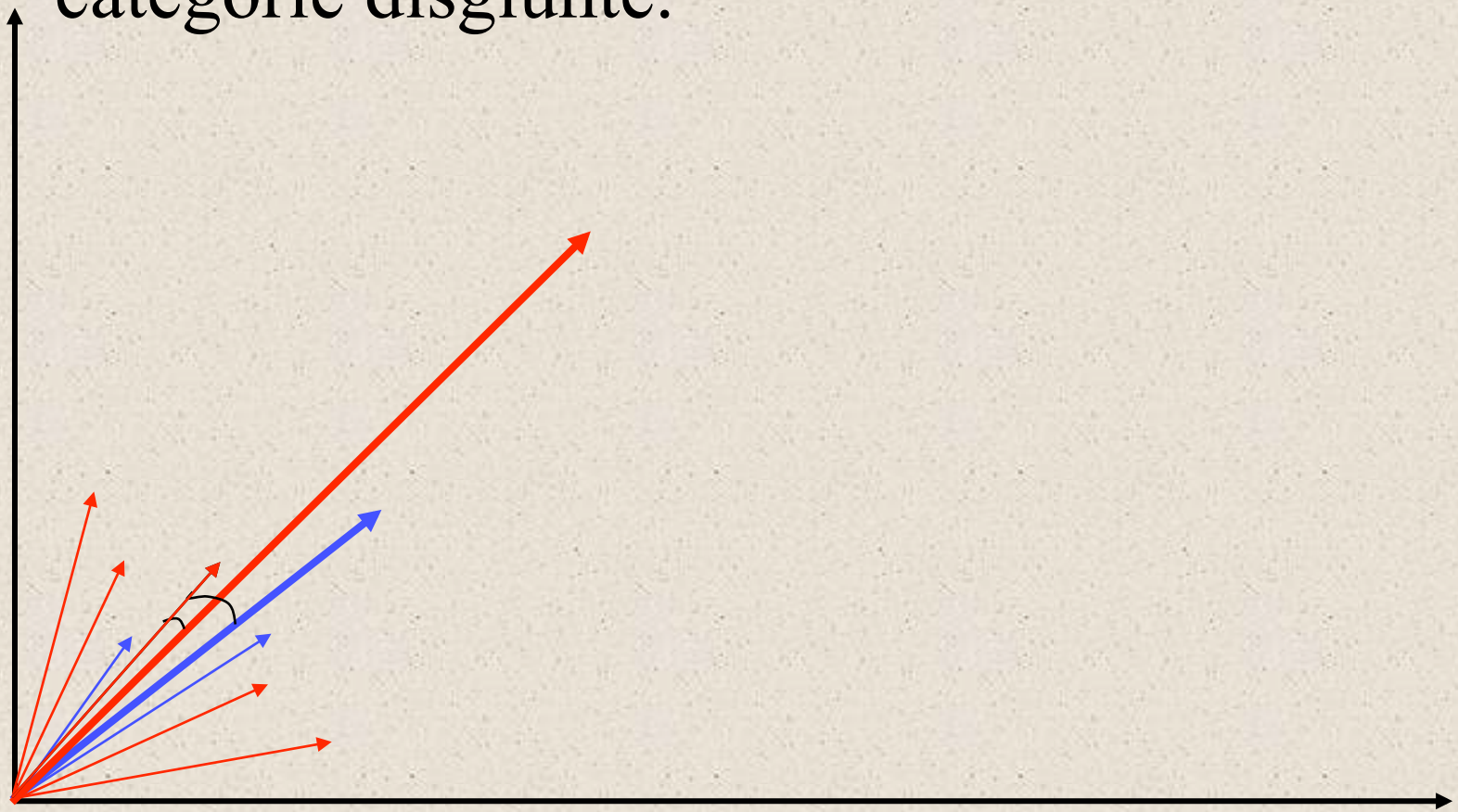
Let $N$ = I primi k esempi in D. (*ottiene così i vicini più simili*)

Return la classe con più esempi in N

# Nearest Neighbor per testi

# Rocchio: anomalia

- I prototipi possono avere problemi con categorie disgiunte.



# Nearest Neighbor

- Nearest Neighbor tende ad avere in tal caso un comportamento migliore.

# Metodi bayesiani

- Apprendere e classificare mediante approcci probabilistici
- Il teorema di bayes gioca un ruolo critico nell'apprendimento e classificazione.

# Categorizzazione Bayesiana

- Sia l'insieme delle categorie $\{c_1, c_2, \ldots c_n\}$
- Sia E una descrizione di un'istanza
- Determinare il grado di appartenenza di E per ogni $c_i$
- P($E$) può essere determinata solo se le categorie sono complete e disgiunte.

$$P(c_i \mid E) = \frac{P(c_i)P(E \mid c_i)}{P(E)}$$

$$P(E) = \sum_{i=1}^{n} P(c_i)P(E \mid c_i)$$

$$\sum_{i=1}^{n} P(c_i \mid E) = \sum_{i=1}^{n} \frac{P(c_i)P(E \mid c_i)}{P(E)} = 1$$

# Categorizzazione Bayesiana

- E' necessario sapere:
  - $P(c_i)$
  - $P(E \mid c_i)$
- $P(c_i)$ sono facilmente stimati dai dati.
  - se $n_i$ degli esempi in $D$ sono in $c_i$, allora $P(c_i) = n_i / |D|$
- Se si assume che le caratteristiche di una istanza siano indipendenti data la categoria $c_i$

$$P(E \mid c_i) = P(e_1 \wedge e_2 \wedge \cdots \wedge e_m \mid c_i) = \prod_{j=1}^{m} P(e_j \mid c_i)$$

# Esempio

- C = {allergia, raffreddore, salute}
- $e_1$ = starnuto; $e_2$ = tosse; $e_3$ = febbre
- E = {starnuto, tosse, ¬ febbre}

| **Prob** | **salute** | **raffreddore** | **allergia** |
|---|---|---|---|
| P($c_i$) | 0.9 | 0.05 | 0.05 |
| P(starnuto\|$c_i$) | 0.1 | 0.9 | 0.9 |
| P(tosse\|$c_i$) | 0.1 | 0.8 | 0.7 |
| P(febbre\|$c_i$) | 0.01 | 0.7 | 0.4 |

# Esempio (continua)

| **Probabilità** | **Salute** | **Rafferddore** | **Allergia** |
|---|---|---|---|
| P($c_i$) | 0.9 | 0.05 | 0.05 |
| P(starnuto \| $c_i$) | 0.1 | 0.9 | 0.9 |
| P(tosse \| $c_i$) | 0.1 | 0.8 | 0.7 |
| P(febbre \| $c_i$) | 0.01 | 0.7 | 0.4 |

E={starnuto, tosse, ¬ febbre}

P(salute | E) = (0.9)(0.1)(0.1)(0.99)/P(E)=0.0089/P(E)

P(raffreddore | E) = (0.05)(0.9)(0.8)(0.3)/P(E)=0.01/P(E)

P(allergia | E) = (0.05)(0.9)(0.7)(0.6)/P(E)=0.019/P(E)

Categforia più probabile: allergia

P(E) = 0.089 + 0.01 + 0.019 = 0.0379

P(salute | E) = 0.23

P(raffreddore | E) = 0.26

P(allergia | E) = 0.50

# Algoritmo di apprendimento
# Naïve Bayes

Let $V$ = il vocabolario di tutte le parole nei documenti in $D$

For each categoria $c_i \in C$

Let $D_i$ = il sottoinsieme di documenti in $D$ nella categoria $c_i$

$P(c_i) = |D_i| / |D|$

Let $T_i$ = la concatenazione di tutti i documenti in $D_i$

Let $n_i$ = il numero totale delle occorrenze in $T_i$

For each parola $w_j \in V$

Let $n_{ij}$ = il numero di occorrenze di $w_j$ in $T_i$

Let $P(w_i \mid c_i) = (n_{ij} + 1) / (n_i + |V|)$

# Algoritmo di test
# Naïve Bayes

Dato un documento di test $X$

Let $n$ = numero di occorrenze in $X$

Return categoria:

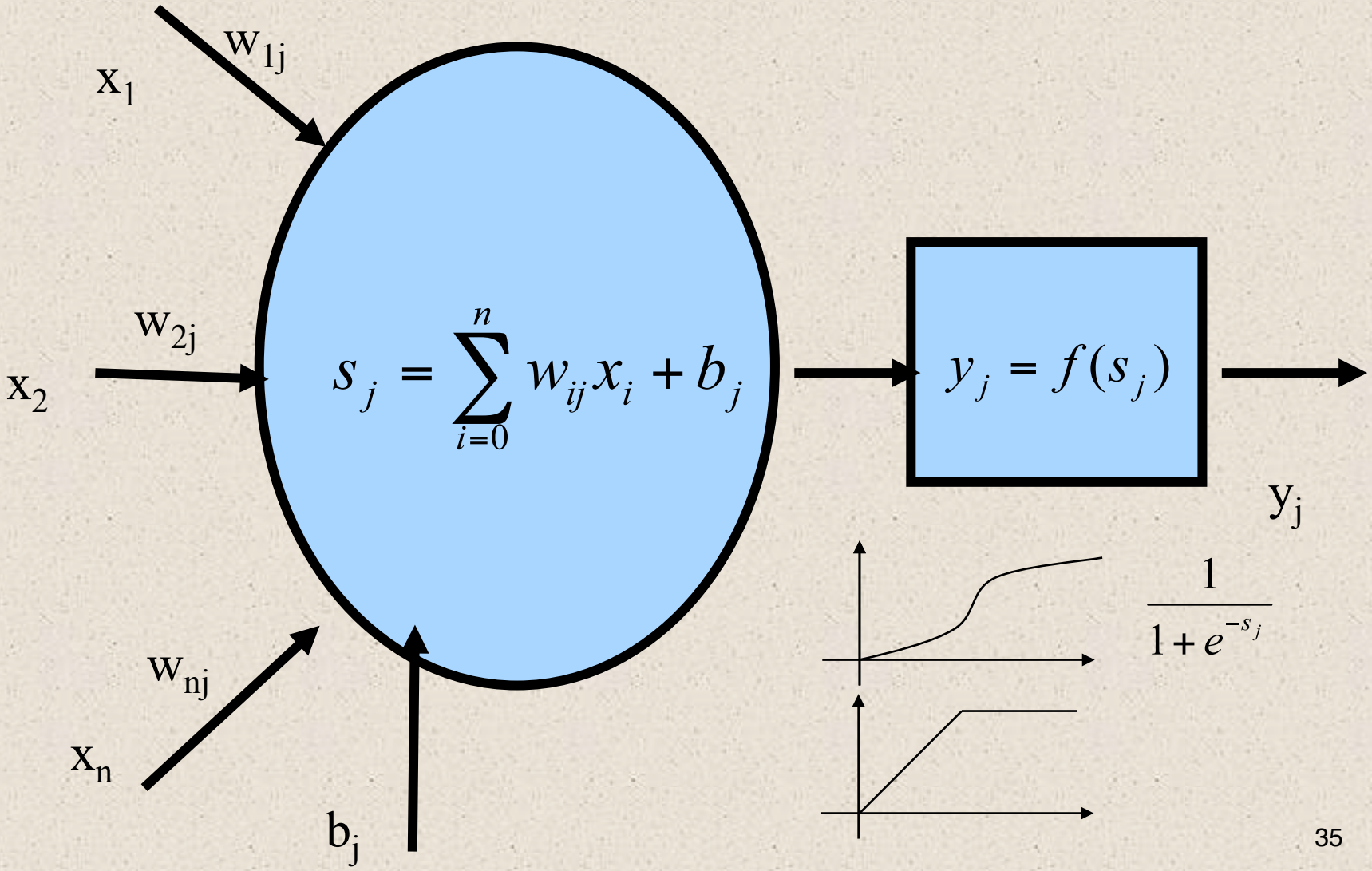
$$\underset{c_i \in C}{\operatorname{argmax}} P(c_i) \prod_{i=1}^{n} P(a_i \mid c_i)$$

dove $a_i$ l'occorrenza della parola nella *i-esima* posizione in $X$

# Reti neurali

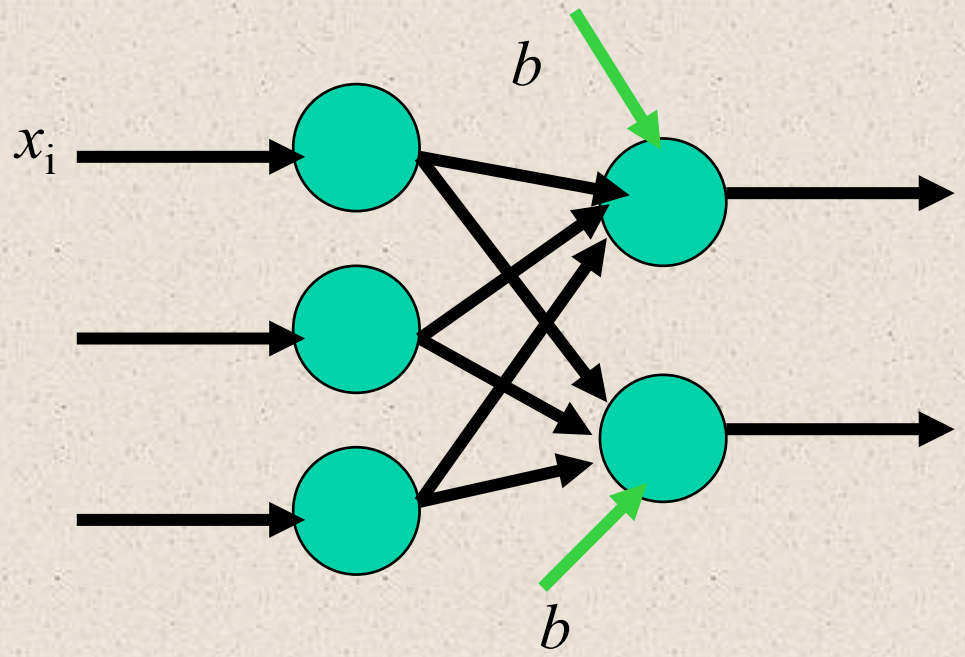
- Una rete neuronale consiste in un pool di semplici processi elementari che comunicano fra loro spedendosi segnali attraverso numerose connessioni pesate

# Neurone artificiale



# Percettrone

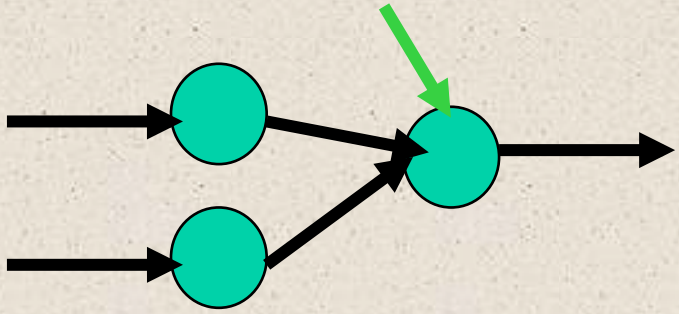
- Una rete a strato singolo consiste in uno o più neuroni di output, ognuno dei quali è connesso con un fattore peso $w_{ij}$ a tutti gli input $x_i$.



# Percettrone

- Nel caso più semplice abbiamo solo due input e un solo output. L'output del neurone è:

$$y = f\left(\sum_{i=1}^{2} w_i x_i + b\right)$$



- Supponiamo la seguente funzione di attivazione

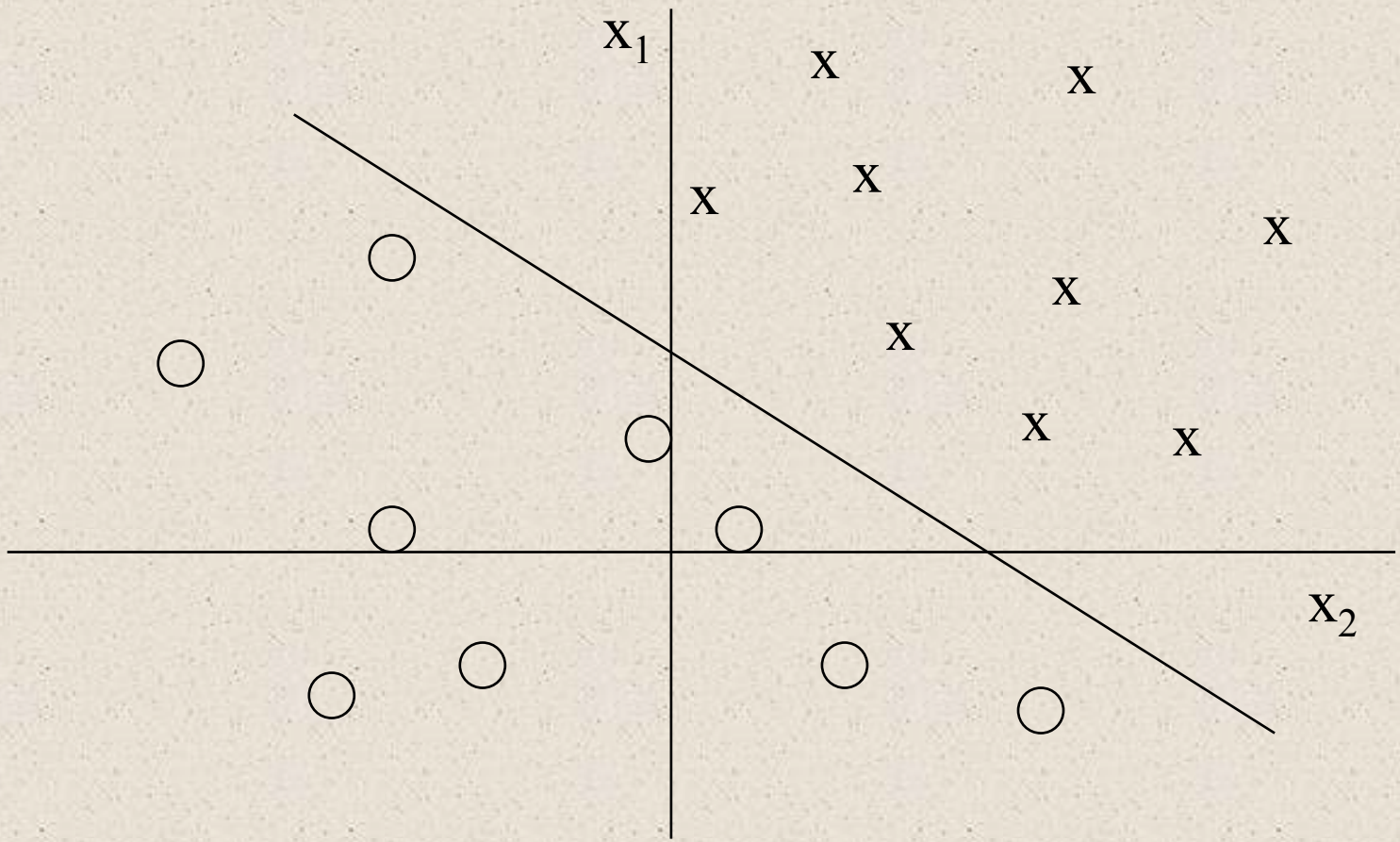
$$f = \begin{cases} 1 & if \quad s > 0 \\ -1 & if \quad s \leq 0 \end{cases}$$

# Percettrone

- In questa semplice rete (il neurone) può essere usato per separare gli input in due classi.
- La separazione nelle due classi è data da

$$w_1 x_1 + w_2 x_2 + b = 0$$

# Percettrone



# Learning

- I pesi della rete neurale sono modificati durante la fase di learning

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}$$

$$b_{ij}(t+1) = b_{ij}(t) + \Delta b_{ij}$$

# Learning

- Si parte con pesi casuali
- Dalla coppia di input *(**x**, d(**x**))*:
  se $y \neq d(x)$ allora modifica il peso mediante la formula:

$$\Delta w_{ij} = d(x) x_i$$

NB Regola del gradiente discendente

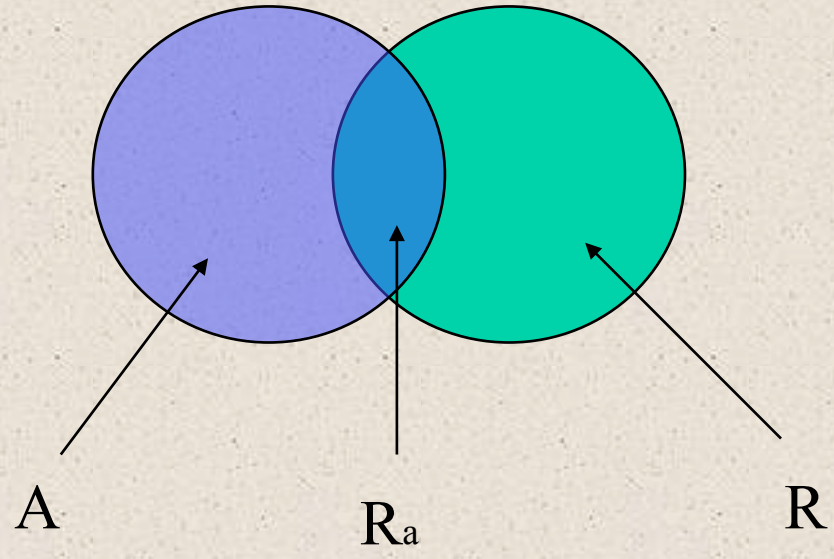
$$\Delta w_{ij} = -\gamma \frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i} \frac{\partial y_i}{\partial w_{ij}} \qquad \frac{\partial y_i}{\partial w_{ij}} = x_j \qquad \frac{\partial E}{\partial y_i} = -(d_i - y_i) = \delta_i \qquad \Delta w_{ij} = \gamma \delta_i x_j$$

# Categorizzazione e percettroni

- Un percettrone per ogni categoria
- Learning sui documenti di training della sua categoria
- Durante la fase di test, il percettrone fornisce un valore VERO/FALSO sull'appartenenza del vettore rappresentativo il documento alla categoria

# Valutare la categorizzazione

- Esistono due parametri accettati dalla comunità IR




RECALL: $|R_a|/|R|$ PRECISION: $|R_a|/|A|$

# Valutare la categorizzazione

- PRECISION
  - L'abilità nel restituire i documenti che sono più rilevanti.
- RECALL
  - L'abilità nel restituire tutti i documenti rilevanti dell'intero dominio.
- F-Measure

$$F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{R}+\frac{1}{P}}$$