Riconoscimento e Recupero dell'Informazione per Bioinformatica

LAB. 6 – K Nearest Neighbor (KNN)

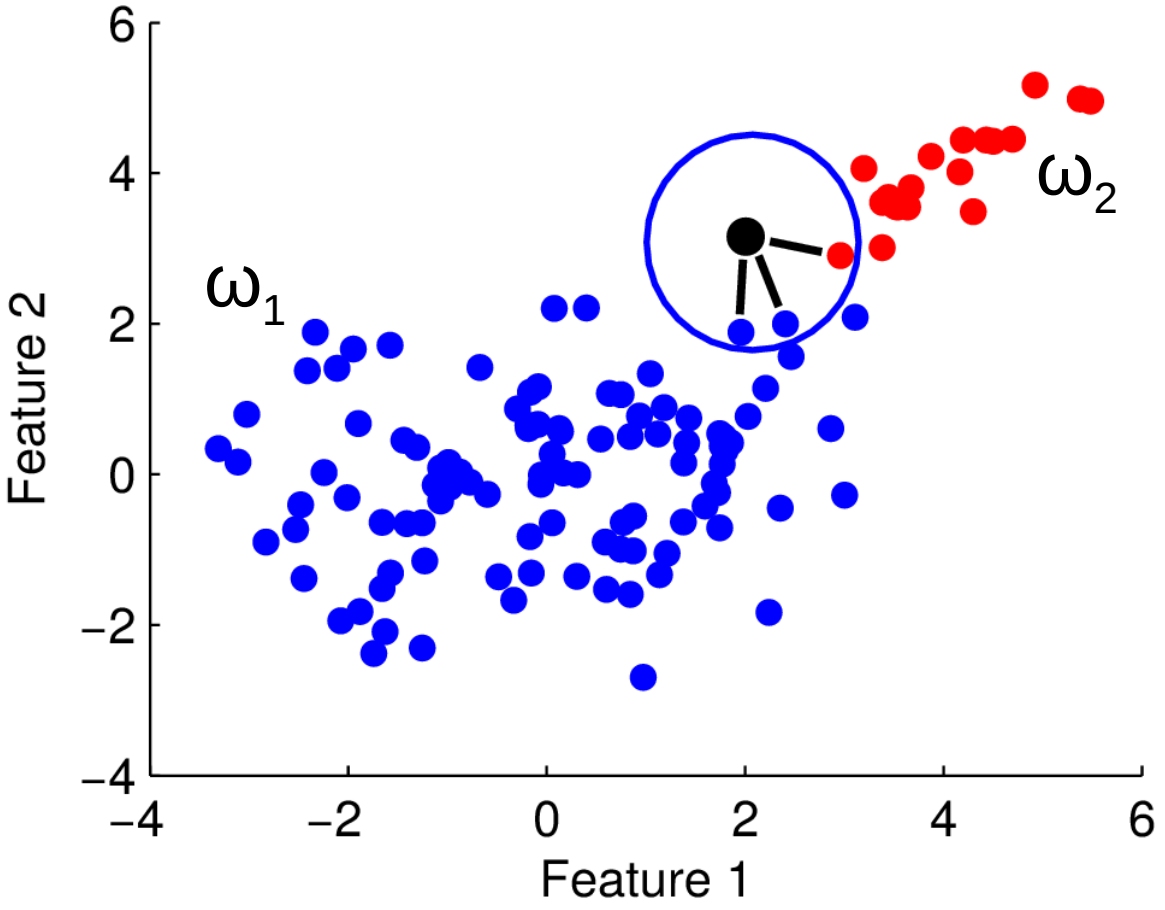
Pietro Lovato

Corso di Laurea in Bioinformatica
Dip. di Informatica – Università di Verona
A.A. 2016/2017

KNN

- Idea: dato un punto di test x_j^{TE} da classificare, considero i K punti di train più vicini a x_j^{TE} secondo una certa metrica

- Assegno a x_j^{TE} la classe più frequente fra questi K punti



KNN: in pratica

- Punto di partenza:
  - Insieme di dati di train
  - Etichette dei dati di train
  - Insieme di dati di test
  - Etichette dei dati di test (non verranno mai considerate se non nella validazione finale)

```
>> load dataset.mat
```

Come fare

- A priori, decido quanti vicini considerare K
- Dato un punto di test x_j^{TE},
  - Calcolo la distanza euclidea fra x_j^{TE} e tutti i punti di train x_i^{TR}
  - Ordino (in maniera crescente) le distanze e trovo gli indici dei K punti di train vicini
  - Controllo le etichette di questi K punti: devo trovare l'etichetta di classe più frequente

Come fare

- Devo trovare l'etichetta di classe più frequente. Una possibilità di realizzazione:

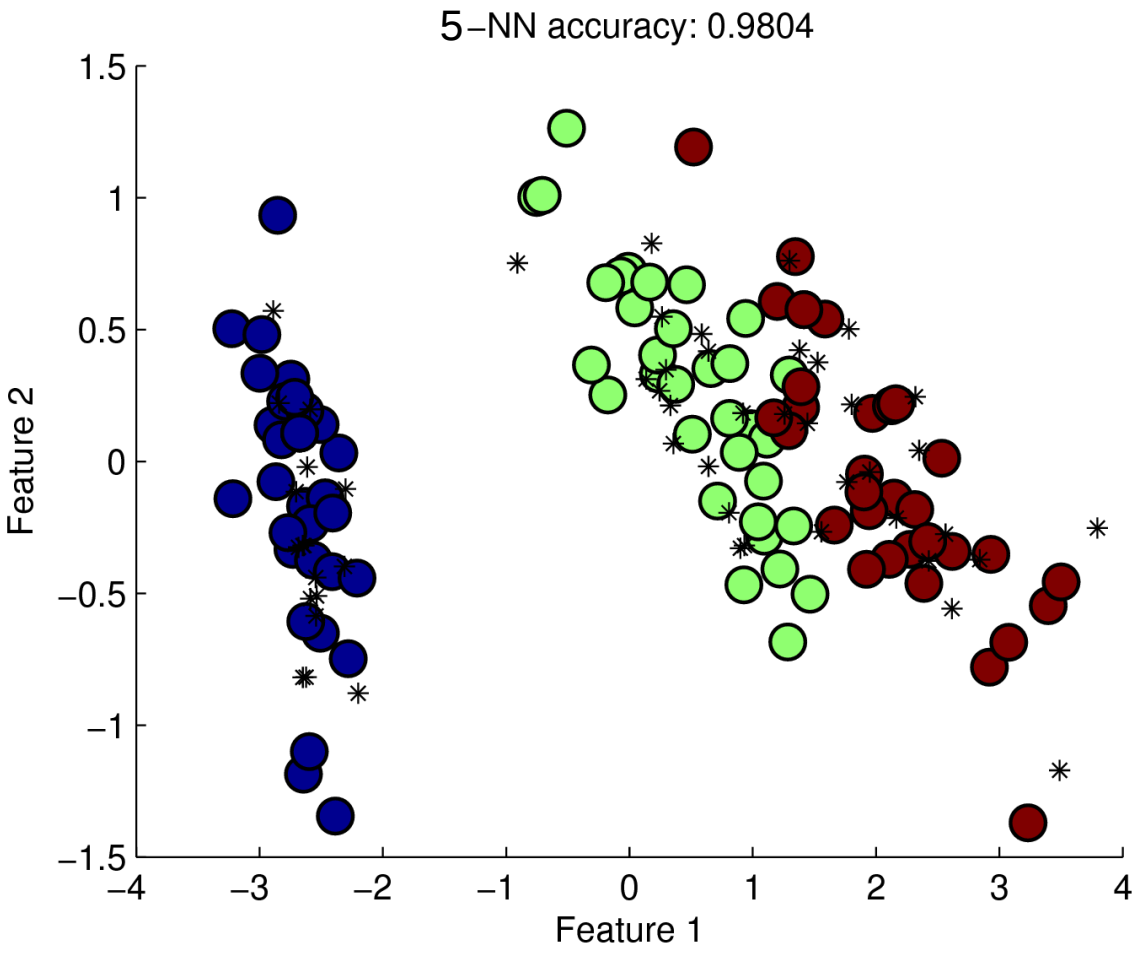
  1) Dato il vettore di etichette dei K punti vicini (il vettore sarà lungo K), il valore più frequente è detto moda

```
>> help mode
```

- Alla fine, assegno x_j^{TE} alla classe più frequente (risultato del comando `mode`)

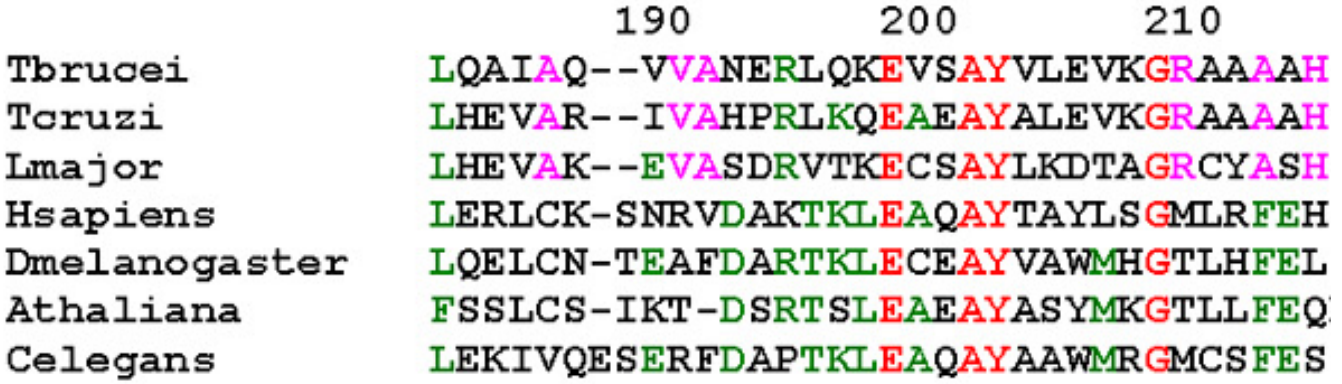
Esercizio 1

- Implementare in Matlab un sistema di classificazione attraverso KNN (utilizzare la distanza euclidea)
- Provare con K=5



Esercizio 2

- Uno dei vantaggi del KNN è che funziona anche per dati non vettoriali
- Esempio di dati non vettoriali: sequenze aminoacidiche




```
>> load sequences.mat
```

Esercizio 2

- Nel dataset sono presenti sequenze appartenenti alla superfamiglia delle Immunoglobuline G (IgG), con etichetta +1, e sequenze esterne alla superfamiglia, con etichetta -1

- Si vogliono classificare le IgG utilizzando il KNN

Esercizio 2

- Da tener presente:
  - Le sequenze sono salvate in un array di **celle**:

```
>> sequenza1 = data_train{1}
```

  - La similarità fra due sequenze S1 e S2 si può calcolare con il comando swalign:

```
>> sim = swalign(S1,S2)
```

  - N.B.: è una similarità, non una distanza!
    - Più alta la similarità, più le sequenze sono vicine
    - Attenzione a quando si ordinano le similarità per trovare i vicini