Clojure — 1-Nearest Neighbor

L’algorisme del k Nearest Neightbors (kNN) o k veïns més propers és un dels algorismes bàsics de l’aprenentatge automàtic o del reconeixement de patrons. Aquest algorisme ens permet abordar problemes de classificació, que serà l'objectiu d'aquest projecte.

La classificació és una de les tasques de reconeixements de patrons en la que volem etiquetar un individu a partir de certes propietats que el caracteritzen; entenent com a individu una entitat de qualsevol tipus. A continuació es mostren dos exemples de conjunts de dades:

class	sepal-length	sepal-width	petal-length	petal-width
setosa	5.1	3.5	1.4	0.2
setosa	4.9	3.0	1.4	0.2
versicolor	6.1	2.9	4.7	1.4
versicolor	5.6	2.9	3.6	1.3
virginica	7.6	3.0	6.6	2.1
virginica	4.9	2.5	4.5	1.7

Font: *Iris* problem UCI repository (Frank & Asunción, 2010)

class	cap-shape	cap-color	gill-size	gill-color
poisonous	convex	brown	narrow	black
edible	convex	yellow	broad	black
edible	bell	white	broad	brown
poisonous	convex	white	narrow	brown
edible	convex	yellow	broad	brown
edible	bell	white	broad	brown
poisonous	convex	white	narrow	pink

Font: *Mushroom* problem from UCI repository (Frank & Asunción, 2010)

Els conjunts de dades per classificació tenen dos components: la matriu X (descripció dels exemples) i el vector y (objectiu de la classificació). L'objectiu del primer conjunt de dades és

classificar flors (*setosa*, *versicolor* o *virginica*) en funció de la llargària i amplada dels seus pètals i sèpals L'objectiu del segon és saber si un bolet és comestible o verinós en funció de les propietats del seu tronc i barret.

Teniu a la vostra disposició dos arxius iris.clj i mushroom.clj les taules de dades anteriors en format Clojure (vectors de maps). Als conjunts continguts en les taules els anomenarem conjunts d'entrenament.

La pregunta següent és: **com podem classificar els exemples següents utilitzant el 1NN?** On classificar té a veure amb escollir la seva etiqueta més previsible.

class	sep-len	sep-wid	pet-len	pet-wid
??	4.9	3.1	1.5	0.1

class	cap-shape	cap-color	gill-size	gill-color
??	convex	brown	narrow	black

El procés consisteix en buscar l'exemple del conjunt d'entrenament que més se li assembla i, un cop seleccionat, assignar al nou exemple l'etiqueta de l'exemple seleccionat. És a dir:

El primer pas consisteix a calcular la distància entre el nou exemple i tots els del conjunt d'entrenament:

- $i = argmin_i(distancia(X_i, T))$

En aquesta fórmula X_i correspon als diferents exemples del conjunt d'entrenament, T a l'exemple a classificar, $distancia$ a una funció de distància i $argmin$ ens retorna l'índex de l'exemple més proper.

El segon pas consisteix a calcular la classe:

- $h(T) = y_i$

Normalment anomenem h a la funció de classificació. Així, la etiqueta resultant serà l'etiqueta y de l'exemple i-èssim.

Fixeu-vos que la funció de distància és un paràmetre de l'algorisme. A continuació teniu les fórmules de 3 distàncies:

- Euclidea: $de(v, w) = \sqrt{\sum_{i=1}^{n}(v_i - w_i)^2}$
- Manhattan: $dm(v, w) = \sum_{i=1}^{n} |v_i - w_i|$

- Hamming: $dh(v, w) = \frac{\sum_{i=1}^{n} \delta(v_i, w_i)}{n}$ $\delta(a, b) = 1$, if $a \neq b$ $\delta(a, b) = 0$, if $a = b$.

Fixeu-vos que podem aplicar les 2 primeres distàncies per al conjunt de dades de les flors i l'última per al dels bolets.

Enunciat:

Definiu 3 funcions per calcular les distàncies: euclidea, manhattan i hamming de dos seqüències. Les funcions han de funcionar per seqüències de mides no conegudes.

1. ```
   (de [1 0] [1 1])  👉  1.0
   (de [0 0 0] [1 1 1])  👉  1.7320508075688772
   (dm [1 0] [1 1])  👉  1.0
   (dm [0 0 0] [1 1 1])  👉  3
   (dh [:a :b] [:b :b])  👉  1/2
   (dh [:a :a :a] [:b :b :b])  👉  1
   ```

2. Definiu una funció per implementar el 1NN tal i com es mostra en els jocs de proves. Els conjunts de dades són als arxius adjunts `iris.clj` i `mushroom.clj`.

En aquest problema **no** es pot utilitzar recursivitat ni iteracions explícites (*recur*).

Joc de proves:

- Iris:

  ```
  (use 'knn :reload-all)
  (use 'iris :reload-all)

  (def flor (knn de iris))

  (flor tst-iris)  👉  :setosa
  ```

- Mushroom:

  ```
  (use 'knn :reload-all)
  (use 'mushroom :reload-all)

  (def bolet (knn dh mushroom))

  (bolet tst-mushroom)  👉  :poisonous
  ```
  ```

Regles del joc:

- Heu de fer el treball en parelles.
- Heu de lliurar un arxiu `clj` amb el nom i cognoms dels dos membres del grup en un comentari a la part de dalt de l'arxiu.
- Lliureu només un dels membres del grup.