IIC 2433 Minería de Datos

https://github.com/marcelomendoza/IIC2433

Cierre de la clase 3 – kNN y LOF

Vecinos cercanos:

¿Cómo se particiona un espacio de representación en dos subespacios al usar Ball-trees?

¿Cuál es la limitación que asumimos al usar un kd-tree?

¿Si uso embeddings, por ejemplo PCA, es mejor usar un kd-tree o un ball-tree?

Local Outlier Factor:

Si un dato es un outlier ¿Su LRD es mayor o menor que la de sus vecinos densos?

¿Cuál es el parámetro que se ajusta para calcular LOF?

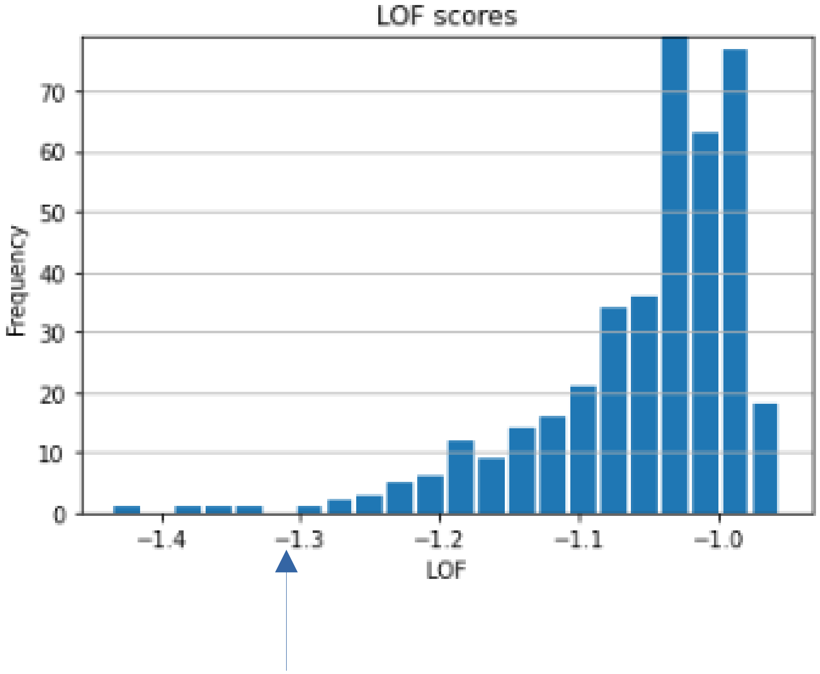
Cierre de la clase 3 – Actividad formativa

	precision	recall	f1-score	support
0	0.33	1.00	0.50	1
1	1.00	1.00	1.00	2
2	1.00	1.00	1.00	1
3	0.50	0.33	0.40	3
4	0.00	0.00	0.00	0
5	1.00	1.00	1.00	3
6	1.00	1.00	1.00	2
7	0.00	0.00	0.00	1
8	0.60	1.00	0.75	3
9	1.00	1.00	1.00	1
10	1.00	1.00	1.00	1
11	1.00	1.00	1.00	1
12	0.50	0.50	0.50	2
13	1.00	1.00	1.00	4
14	0.33	1.00	0.50	1
15	0.00	0.00	0.00	1
16	0.00	0.00	0.00	3
17	0.80	1.00	0.89	4
19	1.00	1.00	1.00	2
20	0.50	0.67	0.57	3
21	1.00	0.75	0.86	4
22	0.50	1.00	0.67	1
23	1.00	1.00	1.00	2
24	1.00	1.00	1.00	1
25	1.00	0.50	0.67	2
26	1.00	1.00	1.00	1
27	1.00	1.00	1.00	3
28	1.00	0.67	0.80	3
29	0.75	1.00	0.86	3
30	1.00	1.00	1.00	3
31	1.00	0.67	0.80	3
32	1.00	1.00	1.00	2
33	1.00	1.00	1.00	2
34	1.00	1.00	1.00	1
35	1.00	1.00	1.00	1
36	1.00	1.00	1.00	3
37	1.00	1.00	1.00	1
38	0.67	1.00	0.80	2
39	0.00	0.00	0.00	3
accuracy			0.80	80
macro avg	0.76	0.80	0.76	80
weighted avg	0.78	0.80	0.78	80

¿Por qué son tan dispares los resultados?

Cierre de la clase 3 – Actividad formativa

Donde cortamos (umbral) depende de donde se distribuyen os datos normales (inliers)



gap

- ISOLATION FOREST -

Isolation Forest

Objetivo del método:

Detectar outliers (datos anómalos) sin construir un perfil de datos inliers, es decir, que no dependa fuertemente de la caracterización de los datos de la distribución base.

Idea:

Operacionalizar el concepto ‘**isolation**’, sobre la base de una complejidad lineal al número de ejemplos del datasets, lo cual lo haría más escalable que LOF.

Propiedades de las anomalías que son usadas por el método:

i) Las anomalías son minoritarias.
ii) Tienen valores en sus atributos que son muy distintos que los de los *inliers*.

¿Cómo lo abordamos?

Usar árboles para aislar anomalías, dejándolas cerca de la raíz del árbol. Los *inliers* tenderán a quedar más cerca de las hojas del árbol. Estos árboles se denominan **isolation trees** (iTrees).

Isolation Forest

Isolation Trees:

Las anomalías serán aquellas instancias del dataset que tienen largos promedios de caminosa la raíz (average path length) más cortos que el resto.

Para robustecer el método, el average path length se calcula sobre varios iTrees, lo cual le da el nombre al método (isolation forest).

Hiperparámetros:

1) Número de iTrees que vamos a construir.
2) Tamaños de las muestras (sub-sampling) usado para construir cada iTree.

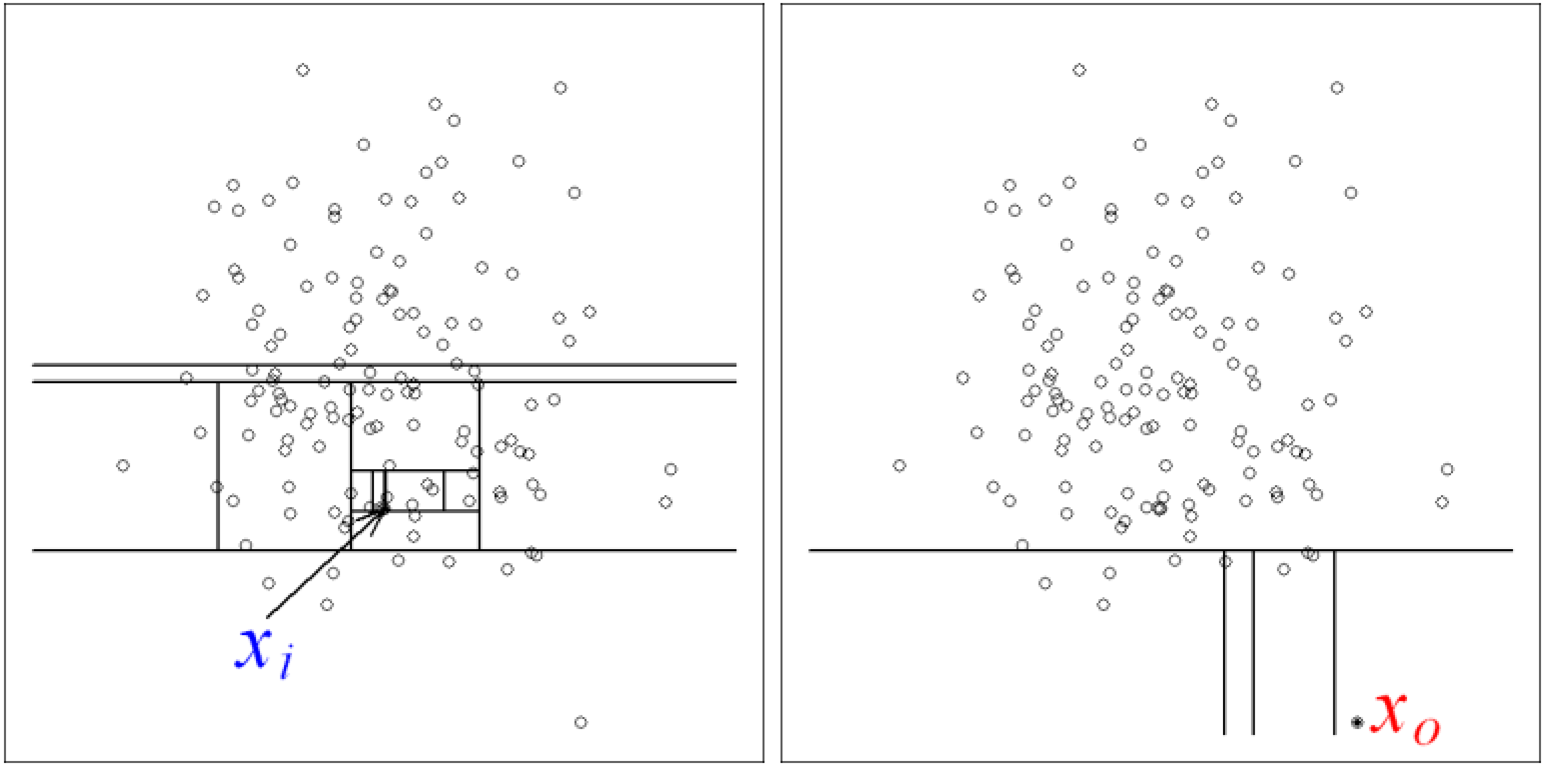
Propiedades:

- Isolation Forest no usa distancia o métricas basadas en densidad.
- Isolation Forest requiere de un uso más bajo de memoria que el de LOF.
- Isolation Forest escala mejor en alta dimensionalidad que LOF.

Isolation Forest

Isolation: Separar una instancia del resto de las instancias del dataset (dejar al dato solo).

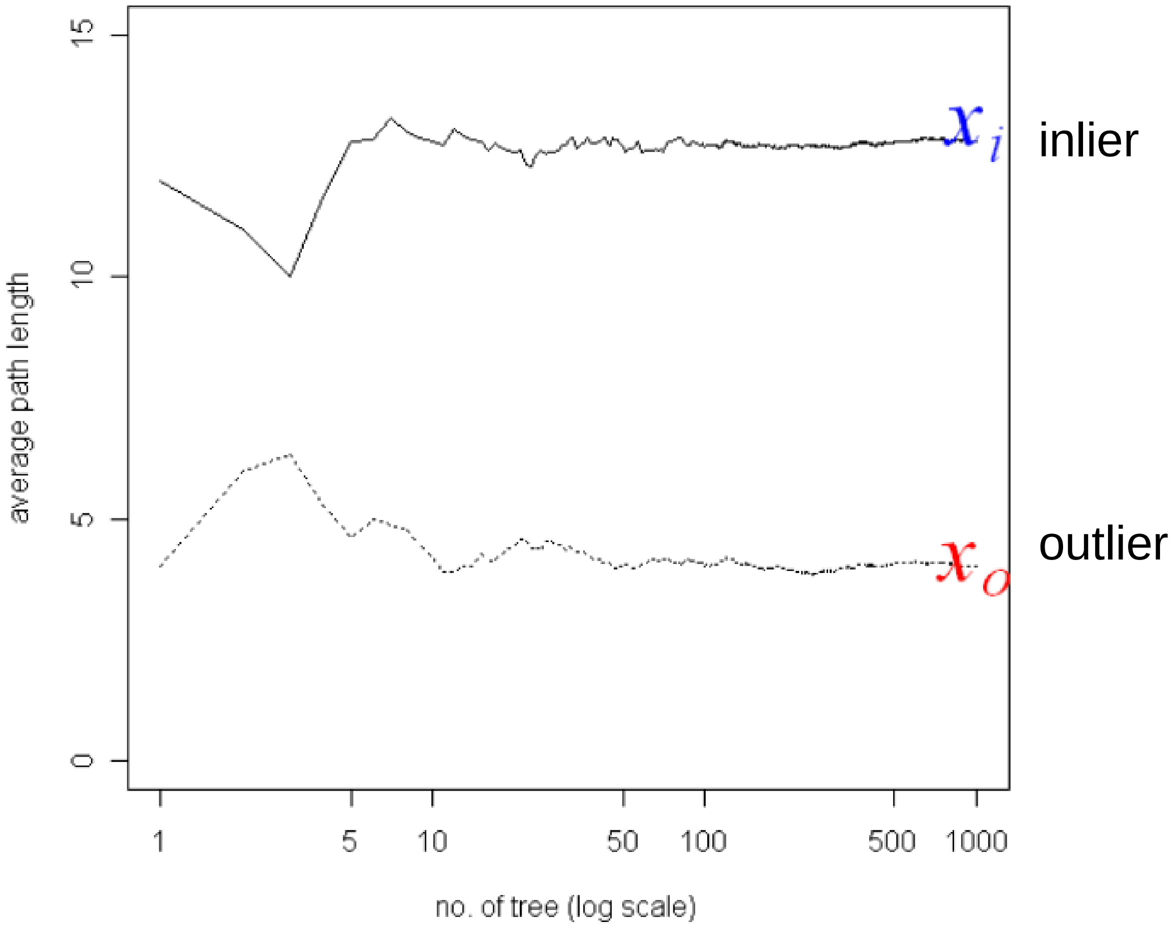
La base del método: iTrees (random trees). Usar una partición aleatorizada sobre una submuestra del dataset. Un inlier requerirá más particiones para ser aislado, mientras que un outlier podría ser aislado en base a unas pocas particiones.



(A) inlier (B) outlier

Isolation Forest

Largo promedio de los caminos (avg path length): un outlier, requerir menos particiones para ser aislado. Entonces, dado que queda más cerca de la raíz del árbol de particiones (random tree), sus caminos a la raíz son más cortos que los de los inliers.



Isolation Forest

Para construir un iTree, se muestrean n instancias del dataset, $X = \{x_1 , \ldots , x_n\}$ y se divide *X* recursivamente escogiendo un atributo *q* al azar y un valor de split *p* (ej., mediana), de manera que *X* se divide en dos subconjuntos. El algoritmo termina cuando los *X* generados cumplen que:

i) $|X| = 1$,
ii) Ya usamos todos los atributos para hacer la partición, ó
iii) Todas las instancias en *X* tienen el mismo valor de *q*.

Un iTree es un arbol binario construido usando este método.

Si asumimos que todas las instancias de *X* son distintas, el iTree tendrá *n* datos aislados. Como es binario, habrá $n-1$ nodos internos (revisar resultado de discretas). Por tanto, el árbol en total tiene $2n-1$ nodos. En consecuencia, el método requiere memoria lineal a *n*.

Isolation Forest

Algorithm 1 : $iForest(X, t, \psi)$

Inputs: X - input data, t - number of trees, ψ - sub-sampling size

Output: a set of t *iTrees*

Initialize $Forest$

set height limit $l = ceiling(\log_2 \psi)$

for $i = 1$ to t **do**

  $X' \leftarrow sample(X, \psi)$

  $Forest \leftarrow Forest \cup iTree(X', 0, l)$

end for

return $Forest$

Algorithm 2 : $iTree(X, e, l)$

Inputs: X - input data, e - current tree height, l - height limit

Output: an iTree

if $e \geq l$ or $|X| \leq 1$ **then**

  return *exNode*$\{Size \leftarrow |X|\}$

else

  let Q be a list of attributes in X

  randomly select an attribute $q \in Q$

  randomly select a split point p from max and min values of attribute q in X

  $X_l \leftarrow filter(X, q < p)$

  $X_r \leftarrow filter(X, q \geq p)$

  return *inNode*$\{Left \leftarrow iTree(X_l, e+1, l),$

  $Right \leftarrow iTree(X_r, e+1, l),$

  $SplitAtt \leftarrow q,$

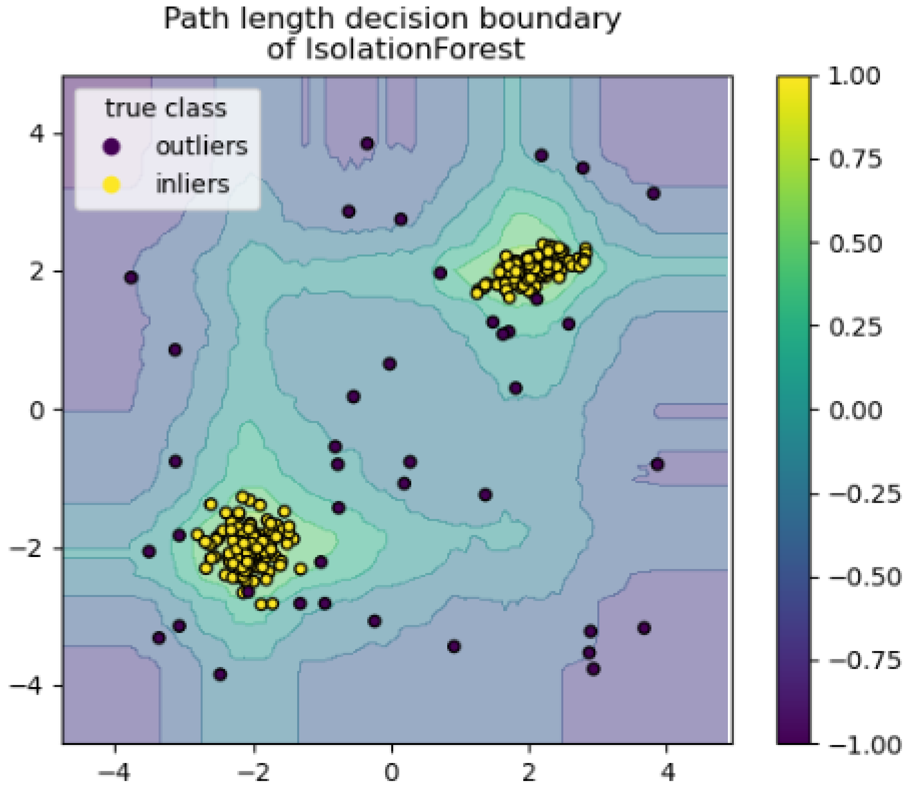
  $SplitValue \leftarrow p\}$

end if

Una vez que hemos construido el iForest, podemos construir un ranking de instancias en base a el valor esperado del camino hacia la raíz sobre Forest. Empíricamente se fija *l* por $log_2(n)$.

Isolation Forest

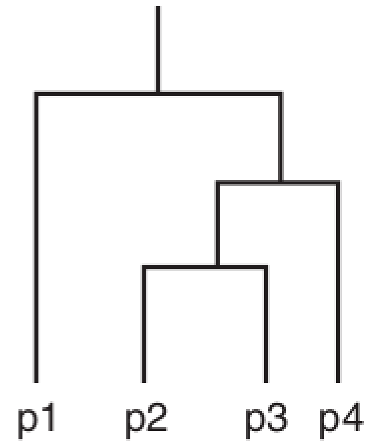
Los avg path lenghts se estandarizan [-1,1]. El resultado del algoritmo muestra algo como lo siguiente:

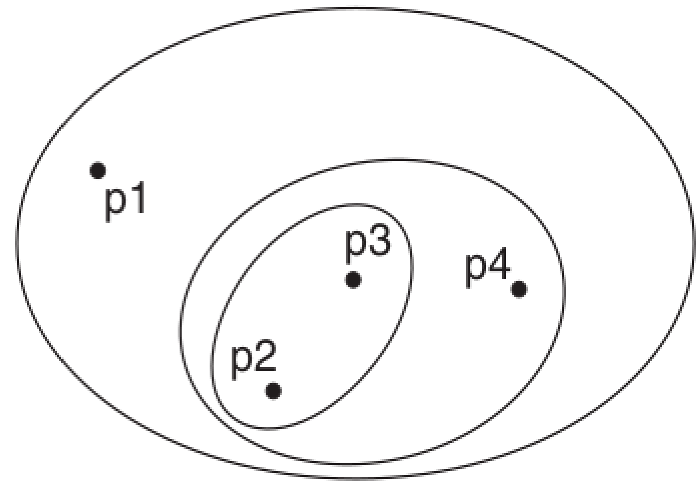


- HAC -

Clustering Jerárquico

Idea:



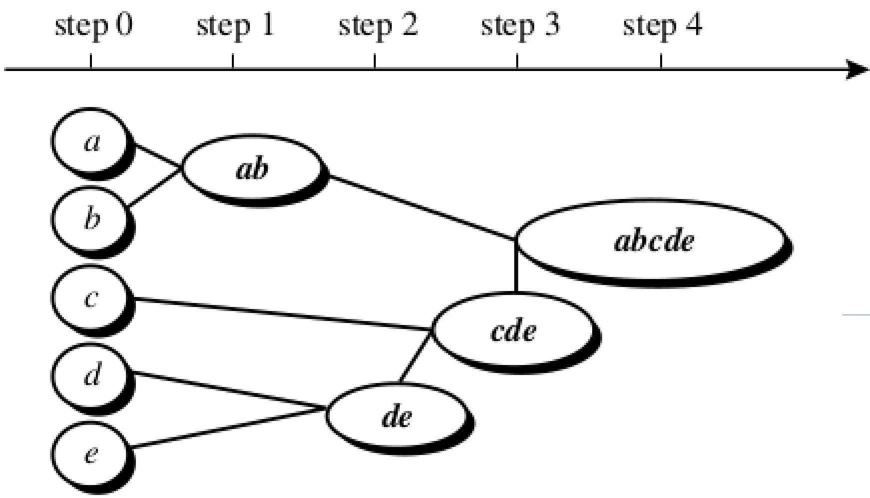


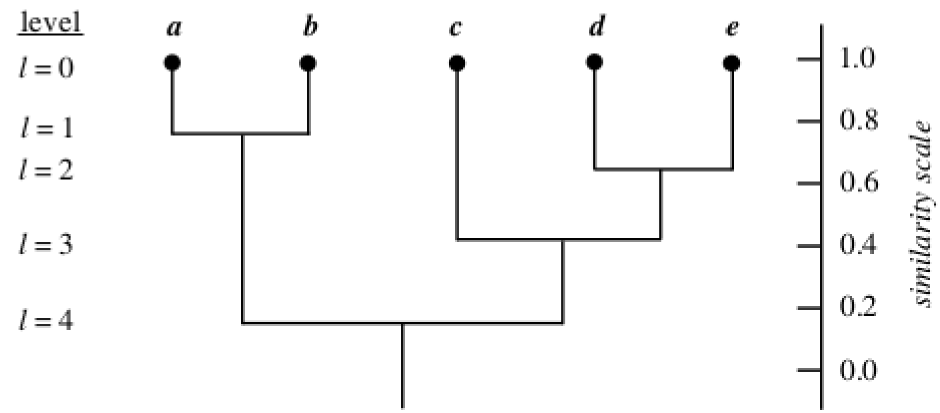
Algorithm Basic agglomerative hierarchical clustering algorithm.

1: Compute the proximity matrix, if necessary.
2: **repeat**
3:     Merge the closest two clusters.
4:     Update the proximity matrix to reflect the proximity between the new cluster and the original clusters.
5: **until** Only one cluster remains.

Clustering Jerárquico

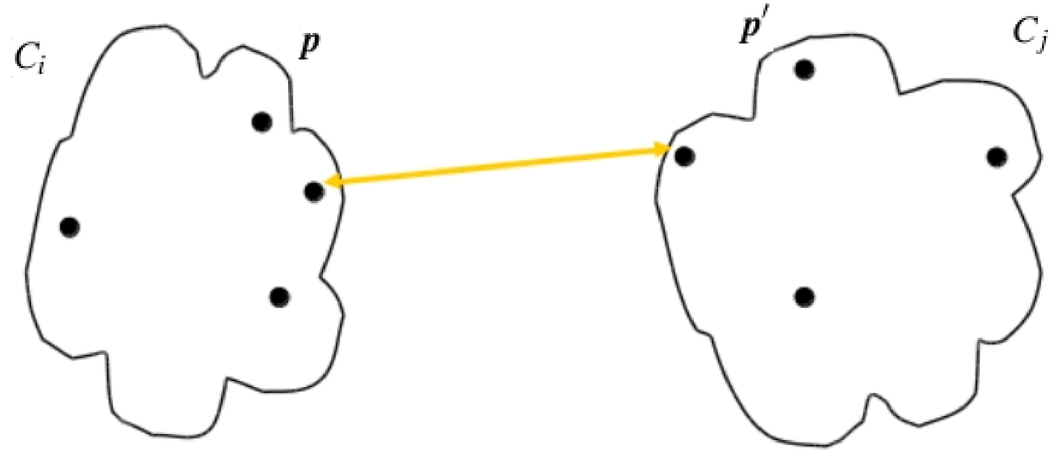
aglomerativo





Clustering Jerárquico Aglomerativo

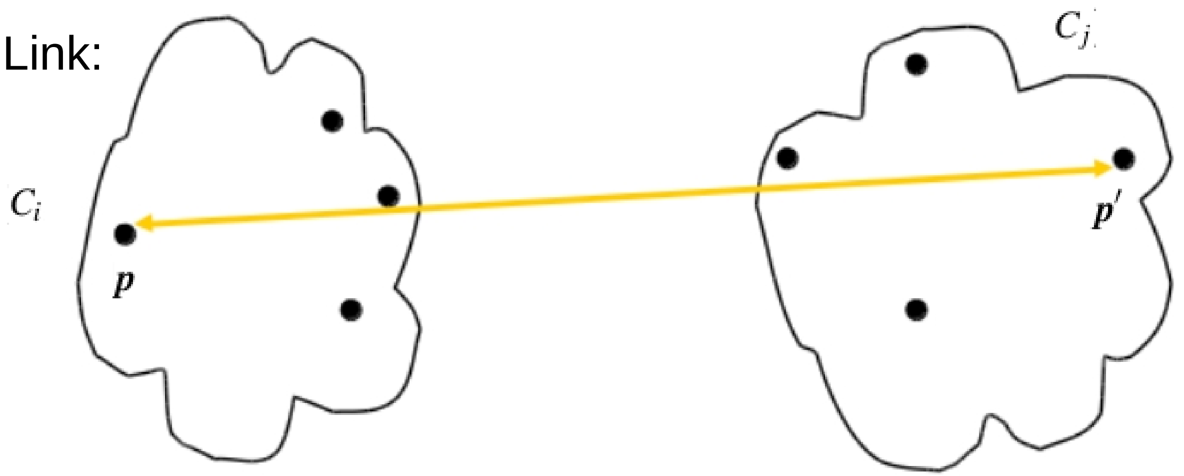
Single Link:



$$d_{min}(C_i, C_j) = \quad min_{\boldsymbol{p} \in C_i,\, \boldsymbol{p}' \in C_j} |\boldsymbol{p} - \boldsymbol{p}'|$$

Clustering Jerárquico Aglomerativo

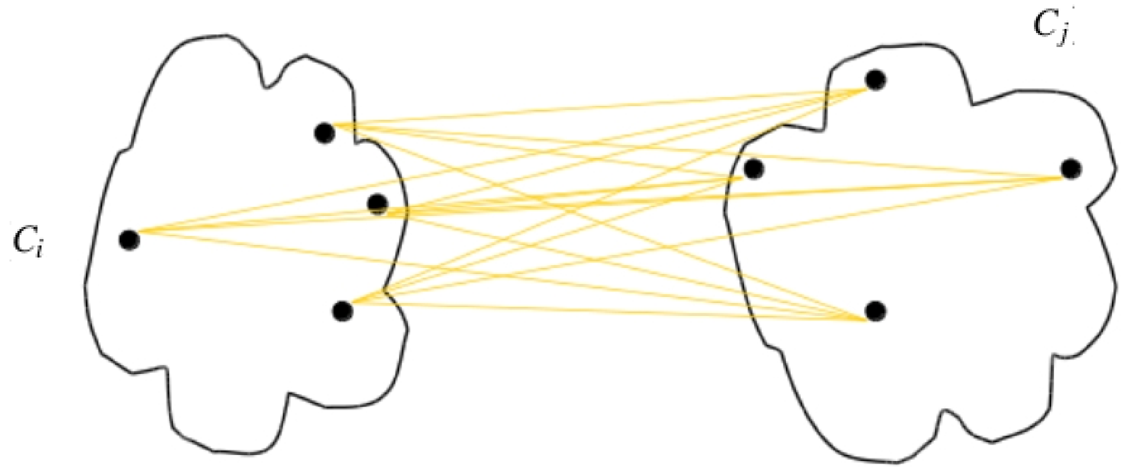
Complete Link:



$$d_{max}(C_i, C_j) = \quad max_{\boldsymbol{p} \in C_i, \boldsymbol{p}' \in C_j} |\boldsymbol{p} - \boldsymbol{p}'|$$

Clustering Jerárquico Aglomerativo

Average Link:



$$d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{\boldsymbol{p} \in C_i} \sum_{\boldsymbol{p}' \in C_j} |\boldsymbol{p} - \boldsymbol{p}'|$$

Clustering Jerárquico Aglomerativo

#datos de cada cluster

Método de Ward:

$$\Delta(A,B) = \sum_{i \in A \bigcup B} ||\overrightarrow{x_i} - \overrightarrow{m}_{A \bigcup B}||^2 - \sum_{i \in A} ||\overrightarrow{x_i} - \overrightarrow{m}_A||^2 - \sum_{i \in B} ||\overrightarrow{x_i} - \overrightarrow{m}_B||^2 = \frac{n_A n_B}{n_A + n_B} ||\overrightarrow{m}_A - \overrightarrow{m}_B||^2$$

Centroide del nuevo cluster

Clustering Jerárquico Aglomerativo

#datos de cada cluster

Método de Ward:

$$\Delta(A,B) = \sum_{i \in A \bigcup B} ||\overrightarrow{x_i} - \overrightarrow{m}_{A \bigcup B}||^2 - \sum_{i \in A} ||\overrightarrow{x_i} - \overrightarrow{m}_A||^2 - \sum_{i \in B} ||\overrightarrow{x_i} - \overrightarrow{m}_B||^2 = \frac{n_A n_B}{n_A + n_B} ||\overrightarrow{m}_A - \overrightarrow{m}_B||^2$$

Centroide del nuevo cluster

$$\sim \quad d_{mean}(C_i, C_j) = |\boldsymbol{m_i} - \boldsymbol{m_j}|$$