Modelos Jerárquicos

Nicolás Kossacoff

Octubre 2024

1. Introducción

Un **modelo jerárquico** sobre un conjunto de datos $D = \{x_1, \ldots, x_n\}$ consiste en una sucesión de particiones anidadas, $\{\mathcal{C}_0, \ldots, \mathcal{C}_{n-1}\}$. Con ellas vamos a definir una jerarquía entre los clusters y representarlos gráficamente a través de una estructura similar a un árbol, llamada dendograma.

Definition. *Sean $\mathcal{C}$ y $\mathcal{B}$ dos particiones del conjunto de datos, D. Decimos que $\mathcal{B}$ se encuentra anidada en $\mathcal{C}$ si cumple que, para todo $B \in \mathcal{B}$, existe un $C \in \mathcal{C}$ tal que $B \subset C$. Es decir, que si una partición se encuentra anidada dentro de otra, los sub-conjuntos de la partición anidada están incluidos en los sub-conjuntos de la otra partición.*

1.1. Métodos

Hay dos tipos de métodos jerárquicos:

- **Aglomerativos (AGNES).** Comienza asumiendo que cada una de las observaciones es un cluster. En cada una de las iteraciones va combinando los dos nodos más similares en uno solo, hasta que finalmente el conjunto de datos D es considerado un nodo.
- **Divisivos (DIANA).** Comienzan con todo el conjunto de datos como un único cluster. En cada iteración va dividiendo los clusters existentes en cluster más pequeños, hasta que finalmente cada observación es considerada un cluster con una única observación.

2. Algoritmos Aglomerativos

2.1. Medidas de disimilaridad

Antes ajustar un algoritmo aglomerativo, es necesario contar con una **medida de disimilaridad** que nos permita ir agrupando las observaciones que son similares

entre sí[1].

Hay muchas maneras de medir la disimilaridad intra-grupos:

- **Single Linkage.** Calcula la disimilaridad entre las observaciones en el cluster G y las observaciones en el cluster H, y utiliza la **mínima** disimilaridad como la disimilaridad entre ambos clusters. Esta medida de disimilaridad suele devolver clusters más grandes y largos.

  Matemáticamente se puede definir como:

  $$D_{SL}(G, H) = \min_{i \in G, j \in H} d_{ij}$$

  donde $d_{ij} = d(x_i, x_j)$ es la dismilaridad entre la observación x_i y la observación x_j.

- **Complete Linkage.** Calcula la disimilaridad entre las observaciones en el cluster G y las observaciones en el cluster H, y utiliza la **máxima** disimilaridad como la disimilaridad entre ambos clusters. Esta medida de disimilaridad suele devolver clusters más compactos.

  Matemáticamente se puede definir como:

  $$D_{CL}(G, H) = \max_{i \in G, j \in H} d_{ij}$$

- **Average Linkage.** Calcula la disimilaridad entre las observaciones en el cluster G y las observaciones en el cluster H, y utiliza la disimilaridad **promedio** como la disimilaridad entre ambos clusters.

  Matemáticamente se puede definir como:

  $$D_{AL}(G, H) = \frac{1}{N_G} \frac{1}{N_H} \sum_{i \in G} \sum_{j \in H} d_{ij}$$

2.2. Algoritmo

Un resumen de este algoritmo se detalla a continuación:

1. Comienza con la partición en la que cada una de las observaciones es considerada un cluster. En ese momento se calcula la **matriz de disimilaridad,** que contiene las disimilaridad entre observaciones y se fija el nivel igual a $L(0) = 0$.

2. En este paso se buscan dos clusters, $\{(r), (s)\}$, con la menor disimilaridad posible (i.e., los clusters más cercanos).

   $$d(r, s) = \min d(r, s)$$

[1] Tener en cuenta que diferentes medidas de disimilaridad nos devuelven modelos jerárquicos distintos.

3. Una vez encontrados los dos clusters más cercanos, los juntamos en un único cluster y fijamos el nivel a $L(m) = d(r, s)$, donde m es el número de la iteración.

4. En este paso se actualiza la matriz de disimilaridad. Primero se borran las columnas y filas correspondientes a los clusters que fueron unificados, (r) y (s), y luego se agrega una columna y una fila para el nuevo cluster, (r, s). Para calcular la disimilaridad entre el nuevo cluster y los anteriores, utilizamos alguna de las disimilaridades definidas en la Sección 2.1.

   Existe una fórmula general para la actualización de la matríz:

$$d(k, (r, s)) = \alpha_s d(k, s) + \alpha_r d(k, r) + \beta d(r, s) + \gamma |d(k, r) - d(k, s)|$$

5. Si llegamos al punto en el que tenemos un único cluster frenamos, sino volvemos al paso (2).

2.3. Debilidad

El problema que tiene este algoritmo es que, para comenzar, necesita tener la matriz de disimilitud completa. Si nuestro conjunto de datos es muy grande, entonces el costo computacional incrementa exponencialmente.

3. Algoritmos Divisivos

A diferencia de los algoritmos vistos en la Sección 2, estos algoritmos comienzan con el conjunto de datos completo, D, y lo van segmentando hasta que cada observación sea un cluster.

3.1. Algoritmo

El paso a paso de este algoritmo se detalla a continuación:

1. En cada paso, se divide un cluster R en dos nuevos sub-conjuntos, A y B, donde $A = R$ y $B = \emptyset$.

2. Para cada observación $a \in A$, calculamos la disimilaridad promedio entre esa observación y el resto de las observaciones:

$$d(a, A - a) = \frac{1}{|A| - 1} \sum_{s \in A, s \neq a} d(s, a) \tag{1}$$

   donde $|A|$ es la cantidad de elementos en el sub-conjunto A. Luego, la observación a^* que maximiza la Ecuación (1), es decir, la observación que en promedio más se aleja del resto, es re-alocada siguiendo la regla de actualización:

$$A_{(t+1)} = A_{(t)} - a^*$$
$$B_{(t+1)} = B_{(t)} \cup a^*$$

3. Una vez asignada la primer observación al conjunto B, tenemos que fijarnos si el resto de las observaciones deben ser re-alocadas o no. Para eso necesitamos saber si es conveniente seguir pasando observaciones de A hacia B o si debemos frenar. Calculamos:

$$diss(a) = d(a, A - a) - d(a, B) = \frac{1}{|A| - 1} \sum_{s \in A, s \neq a} d(s, a) - \frac{1}{|B|} \sum_{h \in B} d(s, h) \quad (2)$$

   Tomemos a la observación a^{**}. Si $diss(a^{**}) > 0$, entonces la observación a^{**} es más parecida a las observaciones en B que en A, por lo que debemos re-alocarla según la regla antes vista. Si $diss(a^{**}) < 0$, entonces hay que frenar con la re-alocación.

4. Si el número de clusters finales ya se encuentra establecido, antes de continuar debemos definir cual de los dos clusters vamos a seguir particionando. Para eso, calculamos el diámetro de cada cluster:

$$diam(R) = \max_{a \in R, b \in R} d(a, b)$$

   El cluster que tenga mayor diámetro, es decir, el menos compacto de los dos, es el que tenemos que seguir particionando.