Títol: **Generalització de mètodes de density-based clustering a dades mixtes**

Volum: **1**

Alumne: **Sheila Mollá Santiago**

Director/Ponent: **Karina Gibert Oliveras**

Departament: **EIO**

Data: **10 de Juny de 2014**

DADES DEL PROJECTE

Títol del Projecte: **Generalització de mètodes de density-based clustering a dades mixtes**

Nom de l'estudiant: ***Sheila Mollá Santiago***
Titulació: ***Enginyeria Informàtica (2003)***
Crèdits: ***37.5***
Director/Ponent: ***Karina Gibert Oliveras***
Departament: ***EIO***

MEMBRES DEL TRIBUNAL *(nom i signatura)*

President: ***Ramon Nonell i Torrent***

Vocal: ***Sebastià Xambó Descamps***

Secretari: ***Karina Gibert Oliveras***

QUALIFICACIÓ

Qualificació numèrica:
Qualificació descriptiva:

Data:

Universitat Politècnica de Catalunya (UPC)
BarcelonaTech

Facultat d'Informàtica de Barcelona (FIB)

Proyecto Fin de Carrera
Ingeniería Informática (2003)

Generalización de métodos de density-based clustering a datos mixtos

Autor:
Sheila Mollá Santiago
Director:
Karina Gibert Oliveras

Barcelona, 10 de junio de 2014

Agradecimientos

Agradezco a todos el soporte y el constante apoyo que me han brindado para poder alcanzar mi objetivo.

A Karina Gibert, gracias por orientarme en el desarrollo del proyecto, cualquier día y a cualquier hora.

A la institución que permitió realizar un estudio con datos reales, gracias por brindar los datos necesarios para poder utilizarlos en la ejecución de diferentes algoritmos: Instituto Guttmann (Hospital de Neurorehabilitación).

Por último, quisiera agradecer a todo el mundo que de una forma u otra ha estado implicado en el desarrollo de este trabajo, su paciencia y apoyo.

A mi madre, gracias por el cariño y apoyo incondicional que me has dado. Sin ti nada de esto hubiera sido posible y parte del mérito es tuyo.

A mi mejor amiga por su paciencia y saberme escuchar aunque muchas veces no supiera de que le estaba hablando.

Gracias a todos porque de forma directa o indirecta habéis contribuido a que haya llegado hasta aquí.

Índice general

Índice de figuras

Índice de tablas

Índice de Algoritmos

Capítulo 1

Introducción

El clustering de datos está en continuo desarrollo, y debido a los grandes volúmenes de datos almacenados, hacen falta métodos escalables. Por otro lado, el análisis de clusters se ha vuelto un tema altamente importante en los estudios de Data Mining y en la mayoría de estudios [KDnuggets 06], [Gibert & Sanchez 2011], [Gibert et al. 2010] aparece entre las técnicas más populares de este campo en aplicaciones reales. Las técnicas de análisis cluster han sido tradicionalmente utilizadas en muchas áreas, por ejemplo:

- Marketing: la búsqueda de grupos de clientes con un comportamiento similar dado una gran base de datos de los que contenemos sus propiedades y los registros de compra.
- Biología: clasificación de plantas y animales de los cuales hemos obtenido sus características.
- Bibliotecas: libro de pedidos.
- Seguro: la identificación de grupos de pólizas de seguros de automóviles con un costo promedio de reclamo alto. Identificación de fraudes.
- Urbanística: identificación de grupos de casas de acuerdo a su tipo, valor y ubicación geográfica.
- Estudios de terremotos: agrupar epicentros de terremotos observados para identificar zonas peligrosas.
- WWW: clasificación de documentos, la agrupación de datos de webs para descubrir grupos de patrones de acceso similares.

Estas técnicas tratan de clasificar en grupos las unidades de estudio descritas a través de un conjunto de variables.

Entre las muchas familias de métodos de clustering existentes, últimamente han ganado popularidad los métodos basados en densidades [Ester 96], que permiten identificar regiones que son densas o dispersas en el espacio y entonces encontrar las clases, y patrones entre sus atributos.

La aplicación de métodos de cluster a grandes bases de datos en general requiere: un mínimo conocimiento del dominio para determinar los parámetros de entrada, que se puedan identificar clusters con forma arbitraria y una alta eficiencia computacional en grandes bases de datos, tratamiento de datos heterogéneos, etc. A partir del problema abierto que se genera con el incremento de las dimensiones de

las bases de datos y la complejidad de las estructuras que se quieren reconocer, aparecen a finales de los 90 una familia nueva de métodos de clustering, son los métodos basados en densidades, especialmente diseñados para descubrir clusters de forma arbitraria. Entre los más conocidos está DBSCAN [Ester 96] que requiere dos parámetros de entrada y ayuda al usuario en la determinación del valor apropiado para ello.

Estos métodos construyen solamente clases densas y de cierto tamaño y a diferencia de la mayoría de métodos precedentes no producen particiones totales de los datos, sino que pueden dejar algunos objetos sin clasificar si están muy lejos de otros núcleos, según los parámetros especificados. Así los métodos basados en densidad pueden generar una clase de ruido con todos estos objetos que no se clasifican.

OPTICS [Ankerst 99] es otro método de la misma familia que corrige algunas limitaciones que se han observado en DBSCAN cuando la densidad de la nube de puntos es muy variable.

1.1. Objetivo

El objetivo principal de este Proyecto de Fin de Carrera es la realización de un estudio comparativo de las técnicas de clustering basadas en densidades dentro del campo de la minería de datos para posteriormente incorporar una implementación generalizada de 2 de estas técnicas en el software llamado **Java-KLASS** (del cual hablaremos posteriormente). Una generalización metodológica de la implementación que se propone es que se puedan utilizar métricas mixtas capaces de tratar matrices de datos heterogéneas, donde variables numéricas y categóricas coexistan describiendo a los individuos. Por tanto, habría que diseñar, implementar, integrar a **KLASS** y aplicar métodos de clustering basados en densidades y utilizar datos reales para el estudio de su comportamiento.

En primer lugar se realizará una introducción de las técnicas de clustering en Minería de Datos y se profundizará especialmente en la familia de métodos basados en densidades.

Se explicarán con detalle dos algoritmos muy conocidos dentro de esta tipología. Estos son: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) y OPTICS (Ordering Points To Identify the Clustering Structure).

El software **KLASS**, que ya realiza clustering jerárquico y contiene varias herramientas de ayuda a la interpretación de los clusters, es la plataforma sobre la que se va a intervenir para añadir dos nuevas funcionalidades: OPTICS y DBSCAN. Se implementarán ambos algoritmos.

Los requisitos deseados para dicha ampliación de **KLASS** son: la escalabilidad del algoritmo para poder manejar datos de gran dimensionalidad, la posibilidad de tratar con distintos tipos de atributos, la posibilidad de descubrir grupos con distinta forma, la capacidad de detectar ruido (noise) y la posibilidad de tener un conocimiento mínimo del entorno para determinar los parámetros de entrada.

Se desarrollará, también, una comparativa de estas técnicas de clustering para poder obtener conclusiones sobre su comportamiento con diferentes tipos de datos y con otras técnicas de cluster no basadas en densidades.

Un objetivo importante de este proyecto es que la integración de DBSCAN y OPTICS en **KLASS** permita la interacción fluida con todas las funcionalidades que **KLASS** ofrece de partida.

Para DBSCAN, además de dar el resultado del algoritmo, generará la variable de clase para poder añadirla a la matriz de datos y poder usar toda la potencia de **KLASS** en el análisis de las clases.

En el caso de OPTICS, el algoritmo dará como resultado el Reachability-Plot, que se utilizará para identificar los clusters y a partir de él se generará la variable de clase.

Usando ambas funcionalidades se hará un estudio con datos reales y una comparación de los resultados de ambos algoritmos con los algoritmos jerárquicos más clásicos, ya implementados.

1.2. Organización de la memoria

La organización del contenido de la memoria del presente Proyecto Fin de Carrera se ha llevado a cabo de la siguiente manera:

- Capítulo 1: Introducción.

  A lo largo de este capítulo se ofrece una visión global del proyecto, se habla de la motivación de realizar este proyecto con los algoritmos que se implementan en él y se detallan los objetivos a alcanzar y cómo llegar a la consecución de los mismos. También se describe la organización de la memoria.

- Capítulo 2: Estado del arte.

  El objetivo de este capítulo es el de presentar el estado del arte de los algoritmos que hemos implementado en este proyecto. Se hace una introducción breve de algunos de los algoritmos de clustering más comunes y se presentan un nuevo tipo de algoritmos de clustering basados en densidades que dan solución a algunos de los problemas que poseen los algoritmos de clustering clásicos. También nos ponemos en situación para hacer una comparación con otros Softwares disponibles.

  El capítulo está estructurado en 4 partes. La primera hace una introducción a las técnicas de clustering más comunes. La segunda habla de todas las métricas disponibles en nuestro software. La tercera habla de dos softwares que implementan los algoritmos que hemos introducido y los compara con el nuestro. Y la cuarta hace una presentación general del software sobre el cuál estamos desarrollando nuestros algoritmos.

- Capítulo 3: Métodos.

  Aquí se describen más detalladamente los algoritmos implementados (DBSCAN y OPTICS), se describe el pseudocódigo de cada uno de ellos y se hace una comparación entre los dos.

- Capítulo 4: Diseño e implementación.

  Se describen las clases que se han añadido al software **Java-KLASS**, se detalla la complejidad de los algoritmos implementados y se presenta el Diagrama de clases y de las estructuras nuevas añadidas al software.

- Capítulo 5: Experimentación y casos de estudio.

  Se presentan dos casos de estudio con diferentes bases de datos usando los dos algoritmos y otros que pertenecen a la familia de los métodos Jerárquicos. Se hace una comparativa entre los resultados de las distintas ejecuciones y se discuten los resultados obtenidos.

- Capítulo 6: Conclusiones.

En este capítulo se presentan las conclusiones finales consecuencia del trabajo realizado y se discuten los resultados obtenidos.

- Capítulo 7: Trabajo futuro.

  Se habla del trabajo futuro que se podría llevar a cabo a partir de lo realizado en este proyecto.

- Apéndice A: Conjunto de datos Iris.

  Presenta una tabla con las diferentes instancias del conjunto de datos de Iris.

- Apéndice B: Conjunto de datos Guttmann.

  Hace una descripción de los atributos de la base de datos del estudio neuropsicológico de Guttmann. También incorpora la descriptiva por grupos y la descriptiva 3D resultado de las diferentes ejecuciones.

Capítulo 2

Estado del arte

2.1. Introducción a las técnicas de clustering

El análisis de cluster se aplica cuando no conocemos a qué grupo pertenecen los datos y queremos encontrar dichos grupos.

Clustering es un proceso que agrupa los datos en clases o clusters, de forma que los datos de un mismo cluster son bastante parecidos y son diferentes de los de otro.

Cuando hacemos clusters podemos identificar regiones que son densas o dispersas en el espacio y entonces encontrar patrones entre sus atributos.

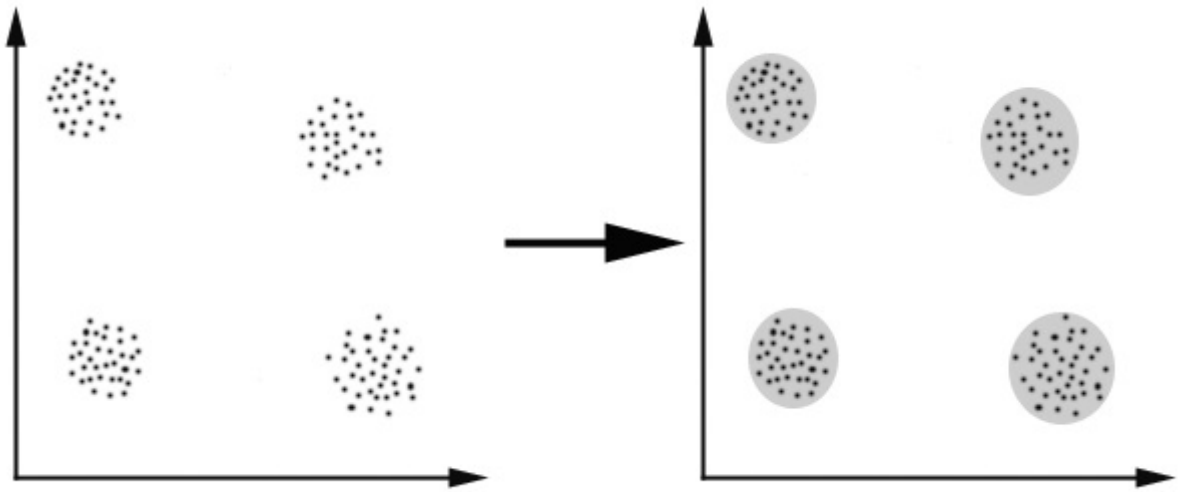


Figura 2.1: Ejemplo de clustering.

Para poder realizar esa agrupación de los datos, es necesario considerar una medida de distancia entre dos objetos. Existe un gran número de medidas de distancia: Euclídea, χ^2, Gower, etc. Más adelante hablaremos en profundidad de estas técnicas [Gibert & Nonell 2005 a].

El problema del clustering probablemente es el más estudiado dentro del Data Mining [KDnuggets 06]. Las técnicas de clustering se agrupan según su filosofía en distintas familias, entre las más conocidas hay: Clustering basado en particiones, Clustering jerárquico, Clustering basado en grid y Clustering basado en densidad.

En este proyecto nos centraremos en el clustering basado en densidad.

2.1.1. Clustering basado en particiones

Los algoritmos basados en particiones han sido durante mucho tiempo los más populares dentro de la minería de datos. Se asegura la división de la base de datos de n objetos, descritos por p variables, en k grupos, de manera que se minimize con un cierto criterio. En todos los métodos de particiones k es un parámetro de entrada del algoritmo y los datos se particionan en k clases, existan realmente o no. En general se optimiza con un criterio relacionado con el error cuadrático, es decir, la suma de los cuadrados de la distancia de cada objeto al centro de su cluster.

El esquema general es:

- Número de clusters k conocido.
- Cada cluster tiene asociado un centroide (centro geométrico del cluster).
- Los puntos se asignan al cluster cuyo centroide esté más cerca (utilizando cualquier métrica de distancia).
- Iterativamente, se van actualizando los centroides en función de las asignaciones de puntos a clusters, hasta que los centroides dejen de cambiar.

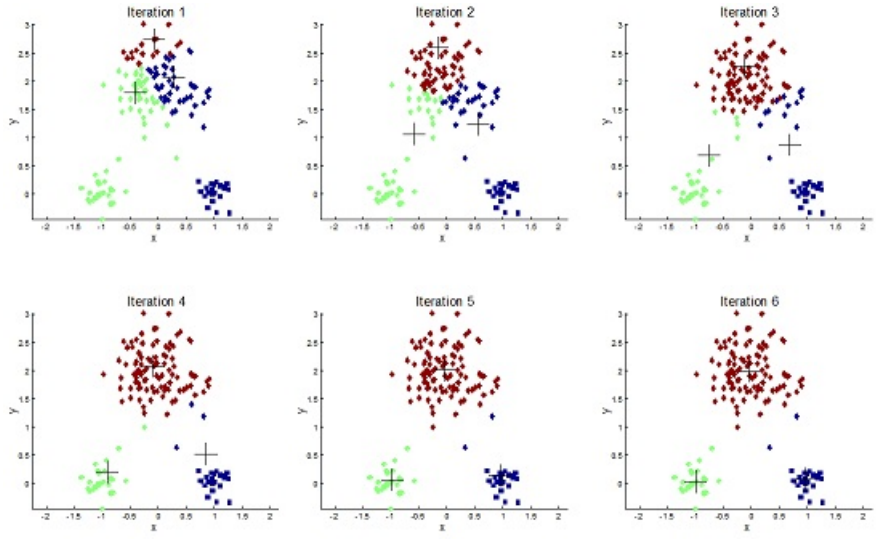


Figura 2.2: Algoritmo K-means.

Estos algoritmos operan iterativamente de acuerdo con el siguiente esquema, el algoritmo conocido como **K-means** [MacQueen 67]:

Inicialización

- Escoger k centroides aleatoriamente.
- Formar k grupos, asignando cada punto al centroide más cercano

Proceso iterativo

Mientras que los centroides cambien:

- Calcular las distancias de todos los puntos a los k centroides.

- Formar k grupos, asignando cada punto al centroide más cercano.
- Recalcular los nuevos centroides.

2.1.2. Clustering Jerárquico

Este tipo de algoritmos jerárquicos [Johnson 87] construyen una jerarquía de grupos anidados que se visualizan a través del *dendrograma*[1]. Este tipo de representación que podemos ver en la Figura 2.3 permite apreciar claramente las relaciones de agrupación entre los datos e incluso entre grupos de ellos. Observando las sucesivas subdivisiones podemos hacernos una idea sobre las posiciones relativas entre los objetos a clasificar y las clases que se han encontrado. Como consecuencia, el número de clases se determina a partir de la estructura del dendrograma.

Esta familia de métodos no requiere el número de clases como parámetro de entrada y se utiliza una matriz de distancias como criterio de clustering.

Según cómo se calcula la distancia del nuevo cluster al actual tenemos diferentes algoritmos: Single Link (SL), Average Link (AL) y Complete Link (CL).

- (SL): En cada paso se unen los dos grupos cuyos elementos más cercanos tienen la mínima distancia.
- (AL): En cada paso se unen los dos grupos tal que tienen la mínima distancia promedio entre sus puntos.
- (CL): En cada paso se unen los dos grupos tal que su unión tiene el diámetro mínimo o los dos grupos con la menor distancia máxima entre sus elementos.

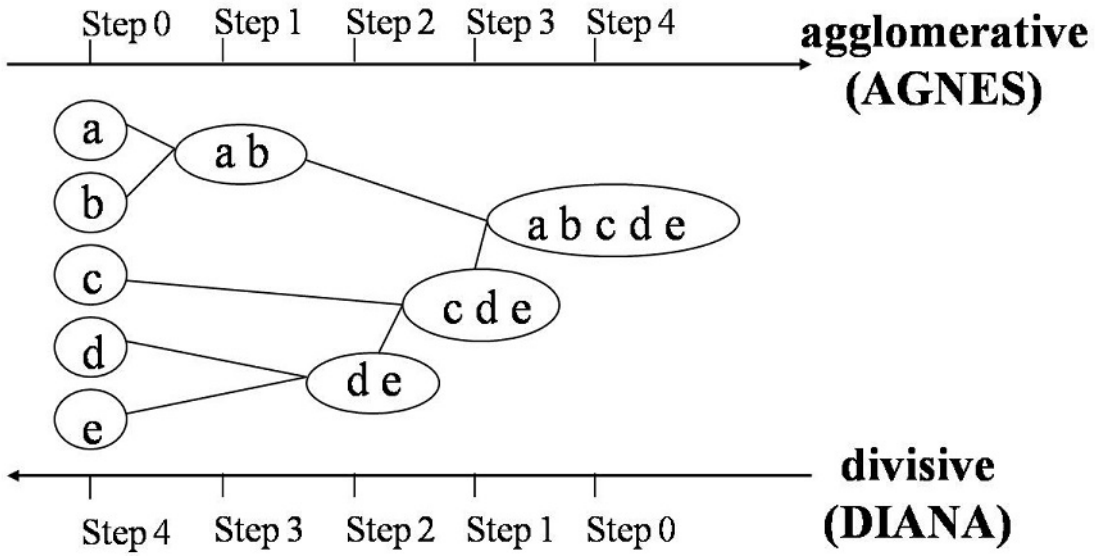


Figura 2.3: Ejemplo de árbol.

El árbol puede ser construido de dos formas: de abajo hacia arriba ("botton-up") o de arriba hacia abajo ("top-down").

[1]Un dendrograma es un tipo de representación gráfica o diagrama de datos en forma de árbol (Dendro=árbol) que organiza los datos en subcategorías que se van dividiendo en otros hasta llegar al nivel de detalle deseado (asemejándose a las ramas de un árbol que se van dividiendo en otras sucesivamente).

En el caso "botton-up", también llamado aglomerativo, se comienza con cada objeto formando un grupo por separado. Los objetos o grupos se combinan sucesivamente según el criterio de agregación elegido, hasta que todos los grupos se hayan unido en uno solo, o hasta que se cumpla alguna condición de terminación.

En el caso "top-down", también llamado divisivo, se comienza con todos los objetos en el mismo cluster, y a medida que se va iterando, se dividen los grupos en subconjuntos más pequeños según determinadas medidas, hasta que cada objeto esté en un cluster individual o hasta que se cumplan las condiciones de terminación.

La forma tradicional de representar la jerarquía de grupos no es la de la Figura 2.3 sino que en la práctica se usa otro tipo de representación. En la Figura 2.4 podemos ver dicha representación.

La ordenada del árbol, si se representa en vertical, (Figura 2.4) o la abscisa, si es horizontal, (Figura 2.3) está graduada y mide la heterogeneidad de las sucesivas clases.

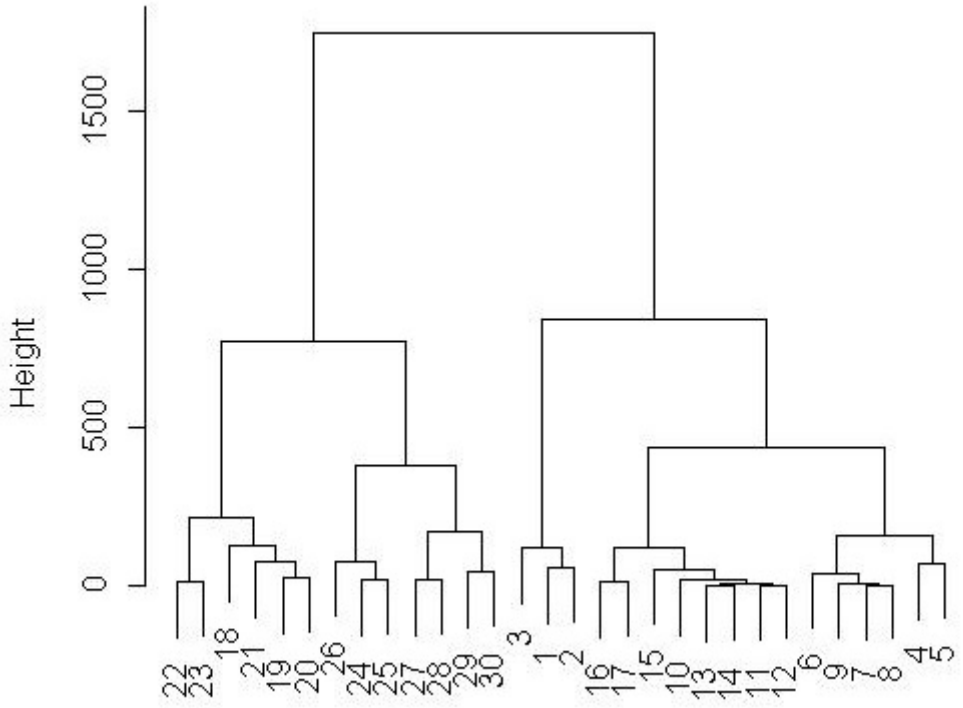


Figura 2.4: Ejemplo de Dendrograma.

Cada nivel muestra los grupos para ese nivel.

- Grupos individuales: Hoja.
- Raíz y nodos internos: Cluster.

Un grupo en el nivel i es la unión de sus clusters descendientes en el nivel $i + 1$.

Los pasos para realizar la clasificación jerárquica son:

1. Elegimos una medida de distancia entre los objetos a clasificar, que serán los clusters o clases iniciales.
2. Decidimos para cada caso qué datos unimos (en base al criterio de agregación).

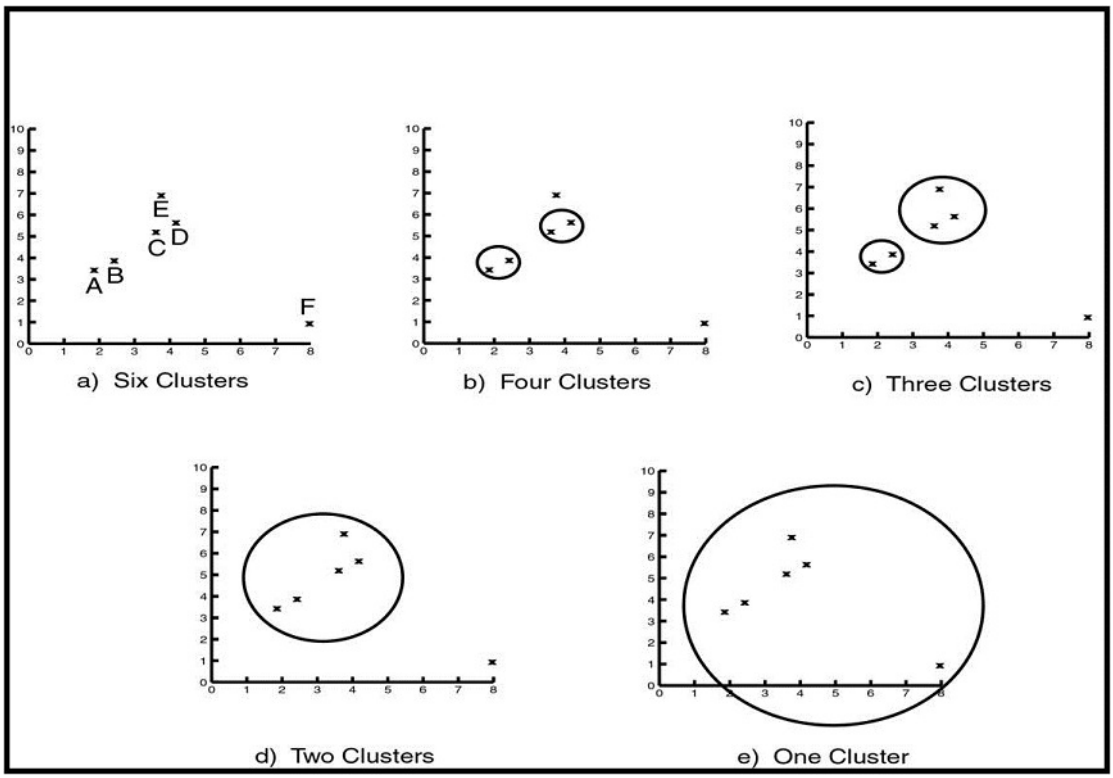


Figura 2.5: Niveles de clustering.

3. Juntamos estos dos clusters en uno nuevo que tenga al menos 2 objetos, de forma que el número de clusters decrece en una unidad.

4. Calculamos la distancia entre este nuevo cluster y el resto.

5. Volvemos al punto 2 hasta que todos los objetos estén en un único cluster.

AGNES (Agglomerative Nesting)

- Introducido por Kaufmann & Rousseeuw 1990.
- Implementado en paquetes de análisis estadísticos, por ejemplo, *Splus*.
- Utiliza el método de enlace único y la matriz de similitud/distancias.
- Combina los nodos que tienen la menor distancia.
- Procede de manera no-descendente.
- Termina cuando todos los nodos se han fusionado en un único grupo.

DIANA (Divisive Analysis)

- Introducido por Kaufmann & Rousseeuw 1990.
- Implementado en paquetes de análisis estadísticos, por ejemplo, *Splus*.
- Orden inverso de AGNES.
- Termina cuando cada nodo forma un cluster por si mismo.

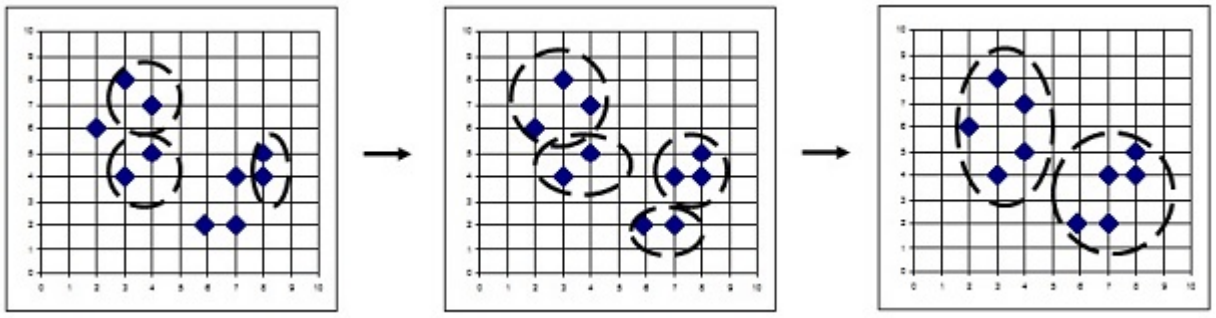


Figura 2.6: Algoritmo AGNES.

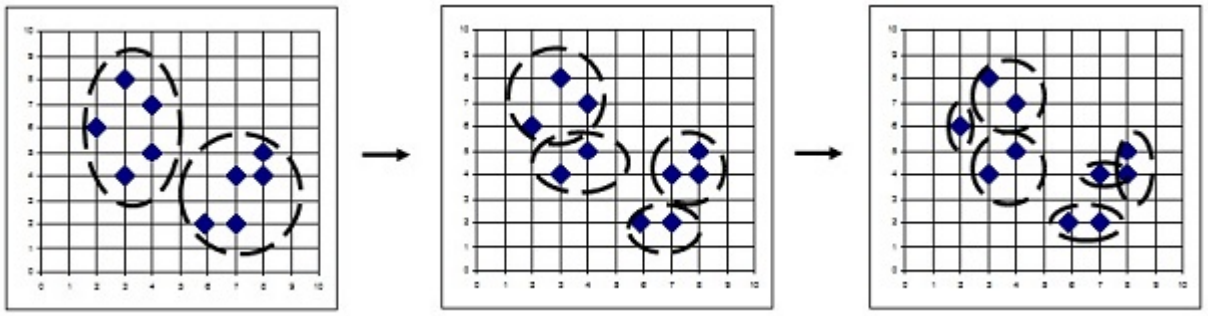


Figura 2.7: Algoritmo DIANA.

2.1.3. Clustering basado en grid

Este tipo de técnicas se basan en una división del espacio en un número finito de celdas que forman una estructura de rejilla en la que se lleva a cabo el clustering. Los objetos de cada una de las celdas son representados por un conjunto de atributos estadísticos de esa celda. El clustering se realiza utilizando la información estadística contenida en cada celda y no el de todos los datos. Dado que el tamaño de las rejillas es inferior al número de objetos, la velocidad aumenta de manera considerable. Para distribuciones de datos concentrados o irregulares, se necesita aumentar la granularidad de la rejilla para obtener un resultado que sea óptimo. Un ejemplo de este tipo de algoritmos es STING [Wang et al. 97], WaveCluster [Sheikholeslami et al. 98] y Clique [Agrawal 98].

2.1.4. Clustering basado en densidad

Los algoritmos basados en densidad tienen la capacidad de identificar grupos con regiones densas en el espacio, separadas por áreas de más baja densidad, donde eventualmente puede haber ruido. Estos algoritmos tienen por lo tanto, la intrínseca capacidad de detectar cúmulos de forma arbitraria y filtrar el ruido mediante la identificación de los valores atípicos. A partir de estas consideraciones se puede esperar una buena calidad del resultado del clustering, y de hecho los resultados experimentales confirman estas expectativas. Por otra parte, contrariamente a lo que se podría suponer, tales algoritmos son generalmente de buen rendimiento.

Las características de esta clase de algoritmos se explican en el capítulo de métodos (Capítulo 3). En particular, examinaremos DBSCAN [Ester 96], el primer y más conocido algoritmo basado en la densidad y OPTICS [Ankerst 99], su descendiente directo, diseñados especialmente para hacer frente a grandes conjuntos de datos, alta dimensionalidad y clases de diferente forma.

Estos métodos surgieron a finales de la década de los 90. Este tipo de algoritmos se basan en el concepto de densidad de un punto y miden el número de puntos alcanzables desde éste teniendo en cuenta un radio concreto y la densidad de la zona circundante.

2.2. Presentación de las métricas

El análisis numérico con matrices de datos heterogéneas requiere una especial atención ya que las distancias fueron originalmente concebidas para variables cuantitativas. Ya en [Anderberg 73], se proponen tres estrategias principales, extensamente discutidas en [Gibert & Nonell 2005 a] que son las siguientes:

- Particionar las variables según su tipo y analizar el tipo dominante [Lebart *et al.* 85].
- Convertir todas las variables a un único tipo, conservando la máxima información original;[2] este es un tema discutido por muchos autores [Anderberg 73] y [Gibert 94].
- Utilizar las medidas de compatibilidad que cubren cualquier combinación de tipos de variables; permite un clustering de matrices de datos heterogéneas sin transformar las propias variables. Las principales ventajas de esta propuesta donde:
  - se respeta la naturaleza de los datos originales.
  - no hay pérdida de información.
  - no hay que importar suposiciones previas arbitrarias que pueden desviar los resultados.
  - permite un estudio de todas las variables conjuntamente.
  - permite un análisis de las interacciones entre las variables de distinto tipo.

En [Gibert & Cortés 93], [Gibert 94] y [Gibert & Sonicki 99] se discute la última propuesta porqué es también nuestro enfoque. En el núcleo del clustering, las distancias entre individuos juegan un papel importante. Hay muchas métricas heterogéneas. En la literatura, se encuentran varias propuestas Gower 71 [Gower 71], Gowda & Diday 91 [Gowda & Diday 92], Gibert 91 [Gibert & Cortés 92, Gibert & Cortés 93, Gibert & Cortés 97], Ichino & Yaguchi 94 [Ichino & Yaguchi 94], Ralambondrainy 95 [Ralambondrainy 95].

Naturalmente, esto no es una revisión extensa, pero entre todas ellas, algunas han sido exitosamente implementadas en **KLASS**. Serán llamadas todas métricas mixtas por comodidad, aunque algunas son sólo coeficientes de disimilitud.

Además de la métrica mixta de Gibert, la cual se diseñó especialmente para **KLASS**, el software **KLASS** también incluye Gower 71, Gowda & Diday 91, Ichino & Yaguchi 94, Ralambondrainy 95 y otras métricas no mixtas.

En este capítulo se explicarán y se formularán las métricas utilizadas para el cálculo de distancias entre objetos.

Se han clasificado las métricas en función del tipo de variables a las que se pueden aplicar *(numéricas o categóricas)*.

[2] En Estadística, tradicionalmente, las variables simbólicas han sido convertidas en variables binarias; entonces, es conveniente hacer el clustering con la métrica χ^2 [Lebart *et al.* 85]. En la Inteligencia Artificial (IA), agrupar valores cuantitativos en un conjunto de cualitativos es más utilizado.

Las métricas implementadas en **Java-KLASS** son las siguientes:

- *Variables Numéricas*: Sólo habilitadas si todas las componentes de los vectores son numéricas. Actualmente se puede optar entre:
  - Euclídea: No admite datos faltantes en la matriz de datos.
    - No normalizada.
    - Normalizada por la Desviación Tipo.
    - Normalizada por Rango.

    Pueden ser calculadas al cuadrado y/o ponderadas.
  - Valor Absoluto: No admite datos faltantes en la matriz de datos.
    - No normalizada.
    - Normalizada por Rango.

    Pueden ser calculadas con objetos ponderados.
  - Minkovski: No admite datos faltantes en la matriz de datos.
    - No normalizada.
    - Normalizada por Rango.

    Dependen de un parámetro p que debe ser un valor entero >0.
- *variables categóricas*: Sólo habilitadas si todas las componentes de los vectores son categóricas. Actualmente se puede optar entre:
  - χ^2: Admite objetos ponderados y/o extendidos.
  - Hamming Generalizado.
- *Variables Numéricas y categóricas simultáneamente*: Habilitadas si tienen componentes de los vectores numéricas y categóricas simultáneamente. Actualmente se puede optar entre:
  - Mixta Gibert: No admite datos faltantes en las componentes numéricas de la matriz de datos [Gibert & Cortés 97]. Depende de 2 parámetros α y β y que tienen que ser valores entre 0 y 1 y deben sumar 1.
    - Valores automáticos de α y β.
    - Valores introducidos por el usuario de α y β.

    Pueden ser calculadas con objetos ponderados y admiten objetos extendidos.
  - Ralambondrainy: No admite datos faltantes en las componentes numéricas en la matriz de datos. Dos propuestas de normalización [Ralambondrainy 88].
    - Inercia.
    - Norma.

    Admiten objetos ponderados y/o extendidos.
  - Gower: Admite datos faltantes y objetos extendidos [Gower 71].

- Gowda - Diday: No admite datos faltantes en las componentes numéricas en la matriz de datos. Admite datos extendidos [Gowda & Diday 92].
- Ichino Yaguchi: No admite datos faltantes en las componentes numéricas de la matriz de datos [Ichino & Yaguchi 94].
  - Valores introducidos por el usuario de p entero > 0 y γ real en [0,0.5].

Representación de los resultados En general cuando se calcula la distancia entre todos los pares de objetos de un conjunto, los resultados se presentan como una matriz de doble entrada indexada en el conjunto de objetos de la forma siguiente [Gibert & Nonell 2005 a]:

$$\begin{array}{c} \\ i_1 \\ i_2 \\ \vdots \\ \vdots \\ i_n \end{array} \begin{array}{c} \begin{array}{ccccc} i_1 & i_2 & i_3 & \dots & i_n \end{array} \\ \begin{pmatrix} 0 & d(i_1,i_2) & d(i_1,i_3) & \dots & d(i_1,i_n) \\ d(i_2,i_1) & 0 & d(i_2,i_3) & \dots & d(i_2,i_n) \\ \vdots & \vdots & \vdots & d(i_j,i_k) & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d(i_n,i_1) & d(i_n,i_2) & d(i_n,i_3) & \dots & 0 \end{pmatrix} \end{array}$$

donde $d(i_j, i_k)$ es la distancia entre los objetos j y k y que será simétrica.

2.3. Distancias para variables numéricas

Dado un conjunto de n individuos I, $\{i_1, i_2, ..., i_n\}$, descritos por un conjunto de variables X_k, $k = 1, 2, ..., K$, donde todas las variables son numéricas **Java- KLASS** implementa las distancias que se presentan a continuación:

2.3.1. Distancia Euclídea

- Distancia euclídea original.

$$d(i,i') = \sqrt[2]{\sum_{k=1}^{K}(x_{ik} - x_{i'k})^2}$$

- Distancia euclídea normalizada:

$$d(i,i') = \sqrt[2]{\sum_{k=1}^{K}\frac{(x_{ik} - x_{i'k})^2}{s_k^2}} \tag{2.1}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define s_k^2 como:

$$s_k^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}$$

y

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{\mathrm{n}}$$

- Distancia euclídea normalizada por el rango:

$$d(i,i') = \sqrt[2]{\sum_{k=1}^{K}\frac{(x_{ik} - x_{i'k})^2}{{R_k}^2}}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define R_k como:

$$R_X = maxX - minX$$

donde max$X$= $\max_{i=1:n}(x_i)$ y min$X$ = $\min_{i=1:n}(x_i)$

Todas estas métricas pueden ser calculadas con objetos ponderados utilizando las siguientes expresiones:

- Distancia euclídea ponderada.

$$d(i,i') = \sqrt[2]{\sum_{k=1}^{K}(w_i x_{ik} - w_{i'} x_{i'k})^2}$$

donde wi es el peso del objeto $i \epsilon I$

- Distancia euclídea normalizada y ponderada:

$$d(i,i') = \sqrt[2]{\sum_{k=1}^{K}\frac{(w_i x_{ik} - w_{i'} x_{i'k})^2}{s_k^2}}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define s_k^2 como:

$$s_k^2 = \frac{\sum_{i=1}^{n}(w_i x_{ik} - \bar{x})^2}{(\sum_{i=1}^{n} w_i) - 1}$$

y

$$\overline{x} = \frac{\sum_{i=1}^{n} x_i * w_i}{\sum_{i=1}^{n} w_i}$$

- Distancia euclídea normalizada por el rango y ponderada:

$$d(i,i') = \sqrt[2]{\sum_{k=1}^{K}\frac{(w_i x_{ik} - w_{i'} x_{i'k})^2}{{R_k}^2}}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define R_k como:

$$R_X = maxX - minX$$

2.3.2. Distancia del Valor Absoluto

- Distancia del Valor Absoluto original.

$$d(i,i') = \sum_{k=1}^{K}|x_{ik} - x_{i'k}|$$

- Distancia del Valor Absoluto normalizada por el rango:

$$d(i,i') = \sum_{k=1}^{K}\frac{|x_{ik} - x_{i'k}|}{R_k}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define R_k como:

$$R_X = maxX - minX$$

Todas estas métricas pueden ser calculadas con objetos ponderados de la forma siguiente:

- Distancia del Valor Absoluto ponderada.

$$d(i,i') = \sum_{k=1}^{K} |w_i x_{ik} - w_{i'} x_{i'k}|$$

- Distancia del Valor Absoluto normalizada por el rango y ponderada:

$$d(i,i') = \sum_{k=1}^{K} \frac{|w_i x_{ik} - w_{i'} x_{i'k}|}{R_k}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define el R_k como:

$$R_X = maxX - minX$$

2.3.3. Distancia de Minkovski

Se trata de una familia de distancias indexada por el parámetro p, que indica el grado de la norma y raíz que se usa.

- Distancia de Minkovski original.

$$d_p(i,i') = \sqrt[p]{\sum_{k=1}^{K} |x_{ik} - x_{i'k}|^p}$$

$$p > 0$$

Así $d_p(i,i')$ coincide con la distancia en valor absoluto original si $p = 1$.

- Distancia de Minkovski normalizada por el rango:

$$d_p(i,i') = \sqrt[p]{\sum_{k=1}^{K} \left(\frac{|x_{ik} - x_{i'k}|}{R_k} \right)^p}$$

donde dadas n observaciones de una variable X, $(x_1 \dots x_n)$ se define el R_k como:

$$R_X = maxX - minX$$

Así $d_p(i,i')$ coincide con la distancia en valor absoluto normalizada por el rango si $p = 1$.

2.4. Distancias para variables categóricas

Dado un conjunto de n individuos I, $\{i_1, i_2, ..., i_n\}$, descritos por un conjunto de variables X_k, $k = 1, 2, ..., K$, donde todas las variables son categóricas **Java-KLASS** implementa las distancias que se presentan a continuación:

2.4.1. Distancia χ^2

- Distancia χ^2 original: Se define sobre una transformación de la matriz de datos cualitativos a su matriz en *forma disyuntiva completa* [Dillon 85] descomponiendo cada variable cualitativa $X_k, (k = 1 : K)$ en un paquete de variables binarias $\{Z_1^k, \dots, Z_{n_k}^k\}$, donde cada Z_j^k $(j = 1 : K)$ representa la modalidad $c_j^k \in \mathcal{D}_k$ d'X_k. Esta tabla disyuntiva completa de dimensiones $(n,\ \sum_{k=1}^{K} n_k)$ está formada, pues, por elementos

$$Z = \left(z_{ij}^k = \begin{cases} 1, & x_{ik} = c_j^k \text{ en la matriz original de datos brutos} \\ 0, & \text{de lo contrario} \end{cases} \right), \quad \begin{matrix} (i = 1 : n) \\ (k = 1 : K) \\ (j = 1 : n_k) \end{matrix}$$

Por ejemplo, la matriz

$$\begin{array}{c} \\ i_1 \\ \vdots \\ i_n \end{array} \begin{array}{c} \begin{array}{ccc} X_1 & \dots & X_K \end{array} \\ \begin{pmatrix} x_{11} & \dots & x_{1K} \\ \vdots & \vdots & \vdots \\ x_{n1} & \dots & x_{nK} \end{pmatrix} \end{array} \text{ amb } \begin{cases} x_{11} = c_2^1 \\ x_{1K} = c_2^K \\ \vdots \\ x_{n1} = c_{n_1}^1 \\ x_{nK} = c_{n_K}^n \end{cases}$$

corresponde a la tabla disyuntiva completa siguiente:

$$\begin{array}{c} \\ i_1 \\ \vdots \\ i_n \end{array} \begin{array}{c} \begin{array}{ccccccccc} Z_1^1 & & Z_2^1 & \dots & Z_{n_1}^1 & \dots & Z_1^K & \dots & Z_{n_K}^K \end{array} \\ \begin{pmatrix} z_{11}^1 = 0 & & z_{12}^1 = 1 & \dots & 0 & & 0 & 1 \dots & 0 \\ & & & & \vdots & & & & \\ 0 & & \dots & & 1 & & 0 & \dots & z_{i_n n_k}^k = 1 \end{pmatrix} \end{array}$$

Bajo esta representación, la distancia de χ^2 se suele utilizar para calcular distancias entre individuos [Benzécri 80]:

$$d^2(i,i') = \sum_{k=1}^{K} \sum_{j=1}^{n_k} \frac{\left(\frac{z_{ij}^k}{z_{i.}} - \frac{z_{i'j}^k}{z_{i'.}} \right)^2}{z_{.j}^k} \tag{2.2}$$

En esta expresión,

$$z_{i.} = \sum_{k=1}^{K} \sum_{j=1}^{n_k} z_{ij}^k, \ (i = 1:n),$$

y

$$z_{.j}^k = \sum_{i=1}^{n} z_{ij}^k = card\left(i : x_{ik} = c_j^k, \ (k = 1:K), (j = 1:n_k)\right)$$

se representa el número de individuos de la muestra de modalidad c_j^k

En [Gibert 94] se desarrolla una expresión equivalente directamente calculable sobre la matriz de datos original sin pasar por su forma disyuntiva completa que es:

$$\chi^2 = \sum_{k=1}^{K} d_k(i,i')$$

$$d_k(i,i') = \begin{cases} 0, & \text{si } x_{ik} = x_{i'k} \\ \frac{1}{I_{k^i}} + \frac{1}{I_{k^{i'}}}, & \text{de lo contrario} \end{cases}$$

donde:

- $I_{k^i} = card\ (\hat{i} : x_{\hat{i}k} = x_{ik})$, número de individuos de la muestra que, por la *k-ésima* variable, son de la misma modalidad que i.

- Distancia χ^2 ponderada:

$$\chi^2 = \sum_{k=1}^{K} d_k(i,i')$$

$$d_k(i,i') = \begin{cases} 0, & \text{si } x_{ik} = x_{i'k} \\ \frac{w_i}{I_{k^i}} + \frac{w_{i'}}{I_{k^{i'}}}, & \text{de lo contrario} \end{cases}$$

donde:

- w_i es el peso del objeto i.
- $I_{k^i} = \sum_{\forall \hat{i}: x_{\hat{i}k} = x_{ik}} w_i$, número de individuos de la muestra que, por la *k-ésima* variable, son de la misma modalidad que i.

- Distancia χ^2 generalizada a objetos extendidos: En [Gibert 94] y [Gibert & Cortés 97] se desarrolla la expresión que incorpora también el cálculo de distancias con objetos extendidos:

$$\chi^2 = \sum_{k=1}^{K} d_k(i, i') \tag{2.3}$$

donde:

$$d_k(i,i') = \begin{cases} 0, & \text{si } x_{ik} = x_{i'k} \\ \frac{1}{I_{k^i}} + \frac{1}{I_{k^{i'}}}, & \text{de lo contrario, si } i \text{ y } i' \text{ son compactas en } X_k \\ \frac{(f_i^{k_s}-1)^2}{I^{k_s}} + \sum_{j \neq s}^{n_k} \frac{(f_i^{k_j})^2}{I^{k_j}}, & \text{si } x_{ik} = c_S^k \text{ y } i \text{ es extedido en } X_k \\ \sum_{j=1}^{n_k} \frac{(f_i^{k_j} - f_{i'}^{k_j})^2}{I^{k_j}} & \text{por } i \text{ y } i' \text{ extendidos en } X_k \end{cases}$$

donde:

- I^{k_j} es el número de individuos de la muestra de valor c_j^k para la variable X_k.
- $I_{k^i} = card\ (\hat{i} : x_{\hat{i}k} = x_{ik})$, número de individuos de la muestra que, por la *k-ésima* variable, son de la misma modalidad que i.
- $(f_i^{k_1}, f_i^{k_2}, ..., f_i^{k_{n_k}})$ donde $f_i^{k_j}, j = 1, 2, ..., n_k$, es la proporción de objetos elementales de la clase representada por el objeto i que tienen valor c_j^k de la variable categórica X_k.

- Distancia χ^2 generalizada a objetos extendidos y ponderada:

$$\chi^2 = \sum_{k=1}^{K} d_k(i, i')$$

donde:

$$d_k(i,i') = \begin{cases} 0, & \text{si } x_{ik} = x_{i'k} \\ \frac{w_i}{I_{k^i}} + \frac{w_{i'}}{I_{k^{i'}}}, & \text{de lo contrario, si } i \text{ y } i' \text{ son compactas en } X_k \\ \frac{(w_i f_i^{k_s} - w_{i'})^2}{I^{k_s}} + \sum_{j \neq s}^{n_k} \frac{(w_i f_i^{k_j})^2}{I^{k_j}}, & \text{si } x_{ik} = c_S^k \text{ y } i \text{ es extendido en } X_k \\ \sum_{j=1}^{n_k} \frac{(w_i f_i^{k_j} - w_{i'} f_{i'}^{k_j})^2}{I^{k_j}} & \text{por } i \text{ y } i' \text{ extendidos en } X_k \end{cases}$$

donde:

- w_i es el peso del objeto i.
- I^{k_j} es la suma de pesos de los individuos de la muestra de valor c_j^k para la variable X_k.

- $I_{k^i} = \sum_{\forall \hat{i}: x_{\hat{i}k} = x_{ik}} w_{\hat{i}}$, número de individuos de la muestra que, por la *k-ésima* variable, son de la misma modalidad que i.
- $(f_i^{k_1}, f_i^{k_2}, ..., f_i^{k_{n_k}})$ donde $f_i^{k_j}, j = 1, 2, ..., n_k$, es la proporción de objetos elementales de la clase representada por el objeto cualquiera i sobre la categoría c_j^k de la variable categórica X_k.

2.4.2. Distancia de Hamming Generalizada

La distancia de Hamming original se define sobre cadenas de bits como el número de posiciones de la cadena con bits diferentes [Hamming 50]:

Aquí se implementa una idea similar que consiste en definir distancia 0 cuando hay igualdad y 1 cuando no hay:

$$d_h(i, i') = \sum_{k=1}^{K} d_k(i, i')$$

donde:

$$d_k(i, i') = \begin{cases} 0, & \text{si } x_{ik} = x_{i'k} \\ 1, & \text{si } x_{ik} \neq x_{i'k} \end{cases}$$

2.5. Distancias para variables numéricas y categóricas simultáneamente

Dado un conjunto de n individuos I, $\{i_1, i_2, ..., i_n\}$, descritos por un conjunto de variables X_k, $k = 1, 2, ..., K$, de las cuales n_ζ son variables numéricas y n_Q categóricas.

2.5.1. Distancia Mixta de Gibert

La métrica mixta [Gibert 91] se define como:

$$d^2_{(\alpha,\beta)}(i, i') = \alpha d^2_\zeta(i, i') + \beta d^2_Q(i, i') \tag{2.4}$$

donde:

- α, β son los pesos para balancear la influencia de los grupos de atributos numéricos y categóricos respectivamente y tienen que estar entre 0 y 1 y $\alpha + \beta = 1$.
- $d^2_\zeta(i, i')$ es la distancia euclídea normalizada por la desviación típica al cuadrado calculada con todas las variables numéricas, donde ζ es el conjunto de índices de todas las variables numéricas (ver la expresión 2.1).
- $d^2_Q(i, i')$ es la distancia de χ^2 generalizada a objetos extendidos calculada sobre toda la matriz de variables categóricas donde Q es el conjunto de índices de las variables cualitativas (ver la expresión 2.3).

La distancia Mixta de Gibert puede ser calculada con objetos ponderados si hacemos que $d^2_\zeta(i, i')$ y $d^2_Q(i, i')$ sean ambos calculados en forma ponderada.

En principio, α, β pueden tomar cualquier valor entre 0 y 1, pero se proponen valores para α y β iguales a [Gibert & Cortés 97]:

$$\alpha = \frac{n_\zeta}{d^2_{\zeta max^*}} \qquad \beta = \frac{n_Q}{d^2_{Q max^*}}$$

siendo n_ζ = número de variables numéricas,

n_Q = número de variables categóricas.

con $d^2_{\zeta max} = max_{i,i'}\{d^2_\zeta(i,i')\}$, $d^2_{Q max} = max_{i,i'}\{d^2_Q(i,i')\}$

2.5.2. Distancia de Ralambondrainy Generalizada

Ralambondrainy [Ralambondrainy 95] propone un método para medir distancias entre pares de individuos descritos por datos mixtos, que se formula como:

$$d^2(i,i') = \pi_1 d^2_{1/\sigma^2}(i,i') + \pi_2 d^2_{\chi^2}(i,i') \tag{2.5}$$

donde:

- $d_{1/\sigma^2}(i,i')$ es la distancia euclídea normalizada por la desviación típica al cuadrado calculada sobre todas las variables numéricas.
- originalmente [Ralambondrainy 95] define $d^2_{\chi^2}$ como la distancia de χ^2 definida sobre el paquete de variables binarias que se podrían desplegar sobre una variable categórica.

En [Ralambondrainy 88] se indica com calcular π_1 y π_2 que son los parámetros ponderadores de la influencia de las variables numéricas y categóricas. En este artículo se proponen dos formas de hacerlo:

- Estandarización por la inercia de cada grupo de variables

$$\pi_1 = \frac{1}{Card(Q)}$$

$$\pi_2 = \frac{1}{n_k - 1}$$

- Estandarización por la norma de los operadores

$$\pi_1 = \frac{1}{\sqrt{\sum\{\rho^2(X_k, X_{k'}) \| k, k' \in \mathcal{Q}\}}}$$

$$\pi_2 = \sqrt{n_k - 1}$$

- n_k es la suma de modalidades de todas las propiedades categóricas
- $\sum\{\rho^2(X_k, X_{k'}) \| k, k' \in \mathcal{Q}\}$ es la suma de las correlaciones de todas las variables numéricas.

Dejando de lado los valores de π_1 y π_2 es sorprendente el parecido entre Ralambondrainy (ver expresión 2.5) y la métrica Mixta (ver expresión 2.4). Observando los artículos originales, de hecho se trata de una misma familia de métricas, sólo que los valores propuestos para los índices α, β o Π_1, Π_2 tienen justificaciones muy diferentes, por lo tanto esta métrica también puede ser calculada con objetos extendidos si $d^2_{\chi^2}(i,i')$ se calcula de esta forma.

2.5.3. Distancia de Gower

El coeficiente de similitud de Gower es probablemente la propuesta más conocida para comparar objetos descritos por diferentes tipos de variables. Se probó en [Gower 71] que la definición del coeficiente de similitud de Gower al cuadrado tiene estructura métrica y que por tanto, es más que un coeficiente de similitud, es una distancia.

Del artículo [Gower 71] se ha extraído la formulación original del coeficiente de similitud de Gower. Su incorporación en **KLASS** requiere una adaptación que lo generalice a objetos extendidos.

Definición original

Dado un conjunto I de individuos $I = \{i_1, i_2, \ldots, i_n\}$, descritos por K variables $X_1, X_2, \ldots, X_K$ de diferentes tipos, siendo x_{ik} el valor que la variable $X_k (k = 1 : K)$ toma para el individuo $i \in I$, el coeficiente de similitud sugerido por Gower define la similitud entre el par de individuos (i, i') como:

$$s(y, i') = \frac{\sum_{k=1}^{K} w_k(y, i') s_k(y, i')}{\sum_{k=1}^{K} w_k(y, i')}$$

donde:

- K es el número de variables.
- $w_k(y, i')$ toma el valor de 1 o 0 indicando si la comparación de los individuos i, i' para la variable X_k, es o no es possible:

$$w_k(y, i') = \begin{cases} 0 & \text{si } (x_{ik} = \text{missing}) \text{ ó } (x_{i'k} = \text{missing}) \\ 0 & \text{si (la variable } X_k \text{ es binária) y } (x_{ik} = \text{false}) \text{ y } (x_{i'k} = \text{false}) \\ & \text{y queremos excluir la ausencia negativa de la variable } X_k \\ 1 & \text{de lo contrario} \end{cases}$$

- $s_k(y, i')$ indica el grado de similitud entre los individuos i, i' para la variable k-ésima. Este grado de similitud, que abarca tanto variables numéricas como categóricas, está definido en el intervalo $[0, 1]$ como:

$$s_k(y, i') = \begin{cases} 1 - \frac{|x_{ik} - x_{i'k}|}{R_k} & \text{si } (X_k \text{ es numérica}) \\ 1 & \text{si } (X_k \text{ es categórica) y } (x_{ik} = x_{i'k}) \\ 0 & \text{si } (X_k \text{ es categórica) y } (x_{ik} \neq x_{i'k}) \end{cases}$$

El hecho de que las unidades de medida de cada variable estén normalizadas por el rango de la variable X_k en lugar de la desviación estándar se debe a dos motivos, según Gower: el rango de la variable es más fácil de calcular y la desviación estándar tiene poco significado en el caso de trabajar con variables de tipos diversos.

2.5.4. Distancia de Gowda-Diday

En el proceso de búsqueda bibliográfica de métricas mixtas se han recogido dos artículos elaborados de forma conjunta por K. Chidananda Gowda y E. Diday [Gowda & Diday 91] y [Gowda & Diday 92]:

El primero presenta una nueva medida de disimilitud mixta formada por 3 componentes que los autores denominan *position* (posición), *span* (tramo, extenderse sobre, cruzar) y *content* (contenido).

En el segundo se definen las mismas medidas desde el punto de vista de la similitud.

La expresión implementada en **Java-KLASS** es la primera. En definitiva, mostramos disimilitud de Gowda y Diday adaptada para que pueda trabajar con objetos de tipo extendido. Sean i y i' dos objetos, descritos por K variables, entonces, la disimilitud entre dos objetos i, i' se define como:

$$D(i,i') = \sum_{k=1}^{K} D_k(i,i')$$

donde:

$$D_k(i,i') = \begin{cases} \frac{|x_{ik}-x_{i'k}|}{R_k} & \text{si } (X_k \text{ es numérica}) \\ 0 & \text{si } (X_k \text{ es categórica}), i,i' \text{ compactas y } x_{ik} = x_{i'k} \\ 1 & \text{si } (X_k \text{ es categórica}), i,i' \text{ compactas y } x_{ik} \neq x_{i'k} \\ \frac{2\cdot\left(1-f_{i'}^{k_o}\right)}{1+\sum_{s\neq o} f_{i'}^{k_s}} & \text{si } (X_k \text{ es categórica}),\ i \text{ compacta} : x_{ik} = c_s^k \text{ y } i' \text{ extès} \\ \frac{2\cdot\left(1-\sum_{j=1}^{n_k} Min(f_i^{k_j}, f_{i'}^{k_j})\right)}{\sum_{j=1}^{n_k} Max(f_i^{k_j}, f_{i'}^{k_j})} & \text{si } (X_k \text{ es categórica}),\ i,i' \text{ extendidos} \end{cases}$$

donde R_k es el rango de la variable X_k.

Como se puede ver, la distancia de Gowda-Diday también corresponde como Gower, a la distancia en valor absoluto normalizada por el rango si X_k es numérica y Hamming si es categórica.

Definición original

La disimilitud total entre dos objetos i y i', $D(i,i')$, cada uno de ellos descritos por K variables de cualquier tipo, es la suma de cada una de las disimilitudes para cada variable X_k:

$$D(i,i') = \sum_{k=1}^{K} D_k(i_k, i'_k)$$

donde:

$$D_k(i,i') = D_p(i,i') + D_s(i,i') + D_c(i,i')$$

siendo D_p,D_s y D_c ϵ [0,1] y:

- D_p o *dissimilarity due to position*: indica las posiciones relativas de los valores de las variables en la recta real.
- D_s o *dissimilarity due to span*: mide los valores relativos de las variables sin considerar las partes que puedan tener en común.
- D_c o *dissimilarity due to content*: mide los valores comunes entre las variables.

Componente *Position* Mide las posiciones relativas de las variables en la recta real, sólo tiene sentido en variables numéricas y es igual a D_{kp}:

$$D_{kp} = \begin{cases} \frac{|x_{ik}-x_{i'k}|}{R_k} & \text{si } X_k \text{ es numérica} \\ 0 & \text{si } X_k \text{ es categórica} \end{cases}$$

donde R_k es el rango de la variable numérica X_k.

Antes de presentar las dos componentes de la disimilitud restantes debemos decir que en [Gowda & Diday 91] se trabaja con variables categóricas multievaluadas, es decir, un valor puede ser un subconjunto de modalidades de la variable.

Componente *Span* El componente D_s, mide el tamaño relativo de los objetos, independientemente de su intersección, se define como:

$$D_{ks} = \begin{cases} 0 & \text{si } X_k \text{ es numérica} \\ \frac{|la-lb|}{la+lb-int} & \text{si } X_k \text{ es categórica (multivaluada)} \end{cases}$$

siendo:

- la = #elementos en x_{ik}
- lb = #elementos en $x_{i'k}$
- int = #elementos comunes en x_{ik} y $x_{i'k}$
- $ls = la + lb - int$

Para considerar únicamente el tamaño relativo de los dos conjuntos de variables categóricas, el componente de la disimilitud *span* en variables categóricas multievaluadas vale 0 cuando los dos conjuntos presentan el mismo número de valores y vale 1 cuando como mínimo, uno de los dos conjuntos es el conjunto vacío, independientemente de la cardinalidad del otro conjunto.

Si i i', son objetos simbólicos, $la = lb = 1$ y el término $|la - lb|$ se hace nulo:

$$D_{ks}(x_{ik}, x_{i'k}) = \frac{|1-1|}{1+1-int} = \frac{0}{2-int}$$

hay que decir que el tamaño de la intersección siempre sería inferior o igual a 1, por tanto, el denominador es siempre diferente de 0 y no se produce ninguna indeterminación del tipo $0/0$.

Por tanto,

$$D_{ks} = \begin{cases} 0 & \text{si } X_k \text{ es numérica} \\ 0 & \text{si } X_k \text{ es categórica (no multievaluada)} \end{cases}$$

Componente *Content* El componente D_c, mide el tamaño de las partes comunes entre los objetos a comparar:

$$D_{kc} = \begin{cases} 0 & \text{si } X_k \text{ es numérica} \\ \frac{la+lb-2\cdot int}{la+lb-int} & \text{si } X_k \text{ es categórica} \end{cases}$$

Vale 0 cuando los dos conjuntos tengan exactamente los mismos elementos y vale 1 cuando se trate de conjuntos que no tengan ningún elemento en común, es decir, disjuntos.

Si i i', son objetos simbólicos, int, es:

$$int = \begin{cases} 1 & \text{si } x_{ik} = x_{i'k} \\ 0 & \text{si } x_{ik} \neq x_{i'k} \end{cases}$$

y haciendo $la = lb = 1$, D_{kc} se reduce a

$$D_{kc} = \begin{cases} 0 & \text{si } X_k \text{ es numérica} \\ 0 & \text{si } X_k \text{ es categórica y } x_{ik} = x_{i'k} \\ 1 & \text{si } X_k \text{ es categórica y } x_{ik} \neq x_{i'k} \end{cases}$$

2.5.5. Distancia de Ichino y Yaguchi

Es una distancia que proviene de la *distancia generalizada de Minkowski de orden p* donde $(p > 0)$, $d_p(i, i')$ como:

$$d_p(i, i') = \sqrt[p]{\sum_{k=1}^{K} \left(\frac{\phi(x_{ik}, x_{i'k})}{|U_k|} \right)^p}$$

donde $|U_k|$ es un normalizador que balancea las magnitudes de las variables refiriéndose todas al intervalo $[0, 1]$

$$|U_k| = \begin{cases} R_k & \text{si } X_k \text{ es cuantitativa continua} \\ n_k & \text{si } X_k \text{ es cuantitativa discreta, nominal, estructural} \end{cases}$$

donde R_k es el rango de la variable, y n_k es el número de valores incluidos en el dominio. En nuestro caso:

$$|U_k| = \begin{cases} R_k \text{ o rango de la variable} & \text{si } X_k \text{ es numérica} \\ n_k \text{ o número de valores incluidos en el dominio} & \text{si } X_k \text{ es categórica} \end{cases}$$

La función $\phi(x_{ik}, x_{i'k})$ mide la distancia de la variable k-ésima entre los individuos i y i', y se define como:

$$\phi(x_{ik}, x_{i'k}) = |x_{ik} \oplus x_{i'k}| - |x_{ik} \otimes x_{i'k}| + \gamma \cdot (2 \cdot |x_{ik} \otimes x_{i'k}| - |x_{ik}| - |x_{i'k}|) \tag{2.6}$$

con $\gamma \in [0, 0{,}5]$, on γ controla el efecto de la proximidad interna y externa (*inner-side nearness* y *outer-side nearness*) entre los valores x_{ik} i $x_{i'k}$ cuando los valores de las variables son intervalos. Los casos extremos son:

- Cuando $\gamma = 0$, la expresión de ϕ se reduce a

$$\phi(x_{ik}, x_{i'k}) = |x_{ik} \oplus x_{i'k}| - |x_{ik} \otimes x_{i'k}|$$

- Cuando $\gamma = 0{,}5$, la expresión de ϕ es

$$\phi(x_{ik}, x_{i'k}) = |x_{ik} \oplus x_{i'k}| - \frac{|x_{ik}| + |x_{i'k}|}{2}$$

Según los autores, hacer $\gamma = 0{,}5$ puede ser una de las opciones más sencillas en un caso general.

A continuación veremos cómo se calculan cada uno de los términos que aparecen en la expresión 2.6.

Para el caso de **Java-KLASS**, $|x_{ik}|$ se reduce a

$$|x_{ik}| = \begin{cases} 0 & \text{si } X_k \text{ es numérica} \\ 1 & \text{si } X_k \text{ es categórica} \end{cases}$$

Operador *JOIN CARTESIANA* El operador *join cartesiana*, $|x_{ik} \oplus x_{i'k}|$, se define para el caso de **Java-KLASS** como:

$$|x_{ik} \oplus x_{i'k}| = \begin{cases} |x_{ik} - x_{i'k}| & \text{si } X_k \text{ es numérica} \\ 1 & \text{si } X_k \text{ es categórica y } x_{ik} = x_{i'k} \\ 2 & \text{si } X_k \text{ es categórica y } x_{ik} \neq x_{i'k} \end{cases}$$

Operador *MEET CARTESIANA* El operador *meet cartesiana*, $x_{ik} \otimes x_{i'k}$, se define en **JavaKLASS** como:

$$|x_{ik} \otimes x_{i'k}| = x_{ik} \cap x_{i'k} = \begin{cases} 1 & \text{si } x_{ik} = x_{i'k} \\ 0 & \text{si } x_{ik} \neq x_{i'k} \end{cases}$$

Propiedades Lema: Para cada componente $x_{ik}, x_{i'k} \in \Lambda(U^{(d)})$, la función $\phi(x_{ik}, x_{i'k})$ satisface todos los axiomas de métrica. La demostración se encuentra en [Ichino & Yaguchi 94].

La función $d_p(x_{ik}, x_{i'k})$ también satisface todos los axiomas de métrica y se puede demostrar utilizando la desigualdad triangular y el lema anterior.

2.6. Sofware disponible

Hay muchos sistemas que se encargan de implementar este tipo de técnicas de clustering basadas en densidad. Concretamente, *Weka* [WEKA Documentation] y *ELKI* [ELKI] implementan los algoritmos que se han desarrollado para este proyecto.

2.6.1. ELKI

ELKI (for Environment for DeveLoping KDD-Applications Supported by Index-Structures) es una plataforma software de libre distribución orientada al Descubrimiento de Conocimiento en Bases de Datos (KDD, "minería de datos") desarrollada para su uso en la investigación y la docencia por la unidad de investigación de los sistemas de bases de datos del profesor Hans-Peter Kriegel en la Universidad Ludwig Maximilian de Munich, Alemania. Su objetivo es permitir el desarrollo y evaluación de algoritmos avanzados de minería de datos y su interacción con las estructuras de índices de base de datos.

El framework *ELKI* está escrito en Java y construido en torno a una arquitectura modular. Los algoritmos más conocidos incluidos actualmente pertenecen al clustering, detección de outliers e índices de base de datos. Un concepto clave de ELKI es permitir la combinación de algoritmos arbitrarios, tipos de datos, funciones de distancia y evaluar estas combinaciones. Cuando se desarrollan nuevos algoritmos o estructuras de índices, los componentes existentes pueden ser reutilizados y combinados.

El proyecto universitario se ha desarrollado para su uso en la enseñanza y la investigación. El código fuente está escrito con el propósito de la extensibilidad, la legibilidad y la reutilización, pero no está optimizado para un rendimiento amplio. La aplicación de los algoritmos requiere el conocimiento acerca de su uso y estudio de la documentación. Está dirigido a estudiantes, investigadores e ingenieros de software.

Los módulos de visualización utilizan SVG para la salida de gráficos escalables y Apache Batik para la representación de la interfaz de usuario, así como la exportación en PostScript y PDF para facilitar su inclusión en publicaciones científicas en LaTeX.

Comparándolo con el módulo que hemos implementado, ELKI tiene la posibilidad de seleccionar muchas métricas:

- EuclideanDistanceFunction
- ManhattanDistanceFunction

Figura 2.8: Captura ELKI.

- LPNormDistanceFunction
- MaximumDistanceFunction
- MinimumDistanceFunction
- ArcCosineDistanceFunction
- ...

Vemos que por ejemplo no tiene la posibilidad de usar métricas mixtas como en nuestro software, sino que asume que el clustering se realiza en un espacio cuantitativo como la mayoría de los paquetes disponibles en el mercado.

ELKI también da la posibilidad de representar los datos según la distancia escogida, pero los presenta haciendo una combinación de todos sus atributos en un espacio 2D, cosa que a veces puede dar dificultades a la hora de interpretar los puntos. En nuestro proyecto disponemos de buenas herramientas de interpretación de clases ya existentes en **KLASS** y además aportamos una herramienta adicional, que desarrolla otro tipo de representación, la 3D. Con esta visualización no solo podemos escoger qué atributos numéricos queremos representar sobre los ejes X, Y, Z y qué variable categórica queremos usar para identificar de que tipo son, sino que también damos la posibilidad de que el usuario interactúe con el sistema y él mismo pueda acercar/alejar o rotar la vista como le convenga, analizando las oclusiones.

No obstante, queremos remarcar las limitaciones que *ELKI* presenta para bases de datos de tamaño moderado.

2.6.2. ELKI

Weka (Waikato Environment for Knowledge Analysis - Entorno para Análisis del Conocimiento de la Universidad de Waikato) [WEKA] es una plataforma de software para aprendizaje automático y minería de datos escrito en Java y desarrollado en la Universidad de Waikato. *Weka* es un software libre distribuido bajo licencia GNU-GPL.

El paquete *Weka* contiene una colección de herramientas de visualización y algoritmos de aprendizaje automático para análisis de datos y modelado predictivo, unidos a una interfaz gráfica de usuario para acceder fácilmente a sus funcionalidades. La versión original de Weka fue un front-end en TCL/TK para modelar algoritmos implementados en otros lenguajes de programación, además de unas utilidades para preprocesamiento de datos desarrolladas en C para hacer experimentos de aprendizaje automático. Esta versión original se diseñó inicialmente como herramienta para analizar datos procedentes del dominio de la agricultura, pero la versión más reciente basada en Java (WEKA 3), que empezó a desarrollarse en 1997, se utiliza en muchas y muy diferentes áreas, en particular con finalidades docentes y de investigación.

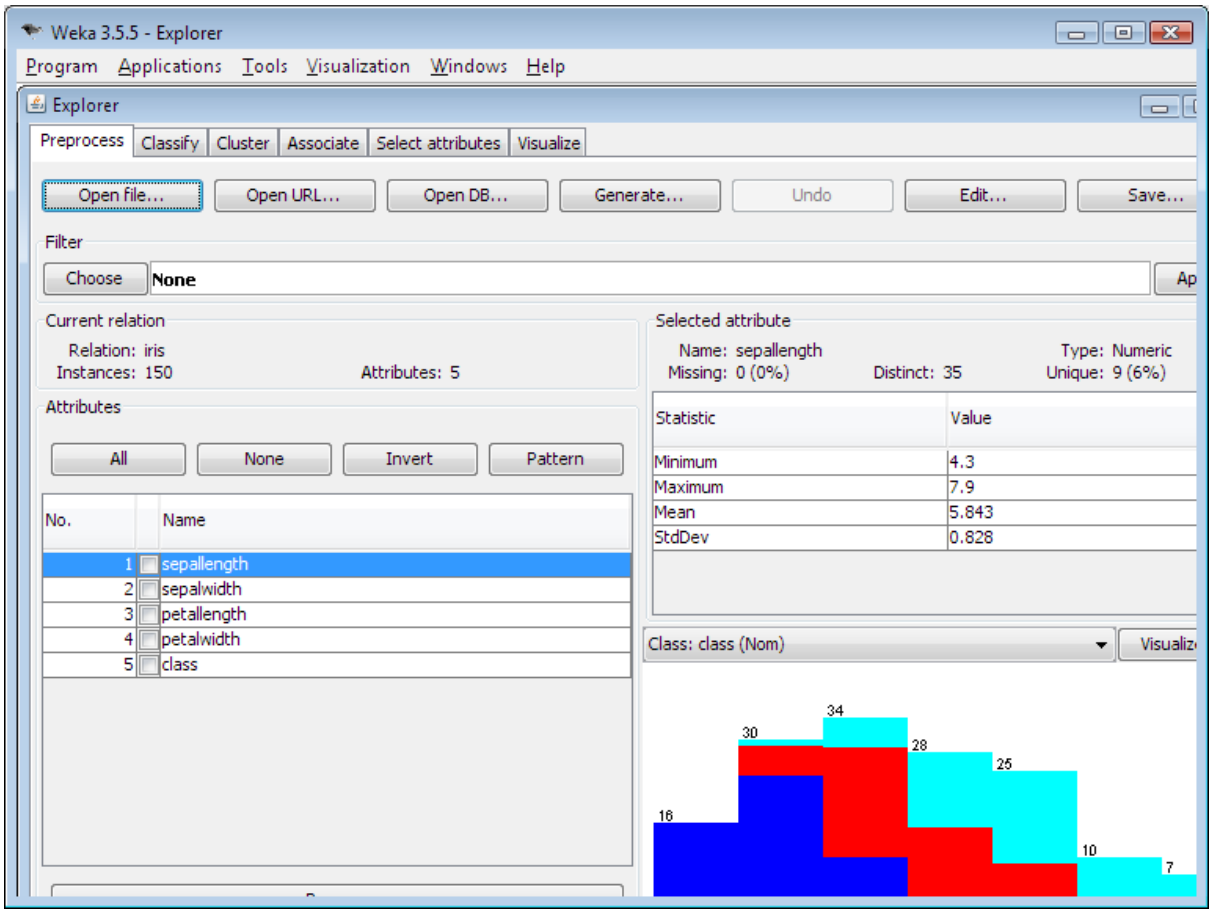


Figura 2.9: Captura Weka.

Respecto a las distancias métricas que se pueden utilizar, *Weka* solamente da la posibilidad de seleccionar la distancia Euclídea o la distancia de Manhattan. Esto hace que tengamos una restricción muy grande a la hora de manejar diferentes bases de datos. En cuanto a la representación de los datos, *Weka* da la posibilidad de visualizarlos en 2D por lo que, igual que pasaba con *ELKI*, da al usuario menos perspectiva de visualización.

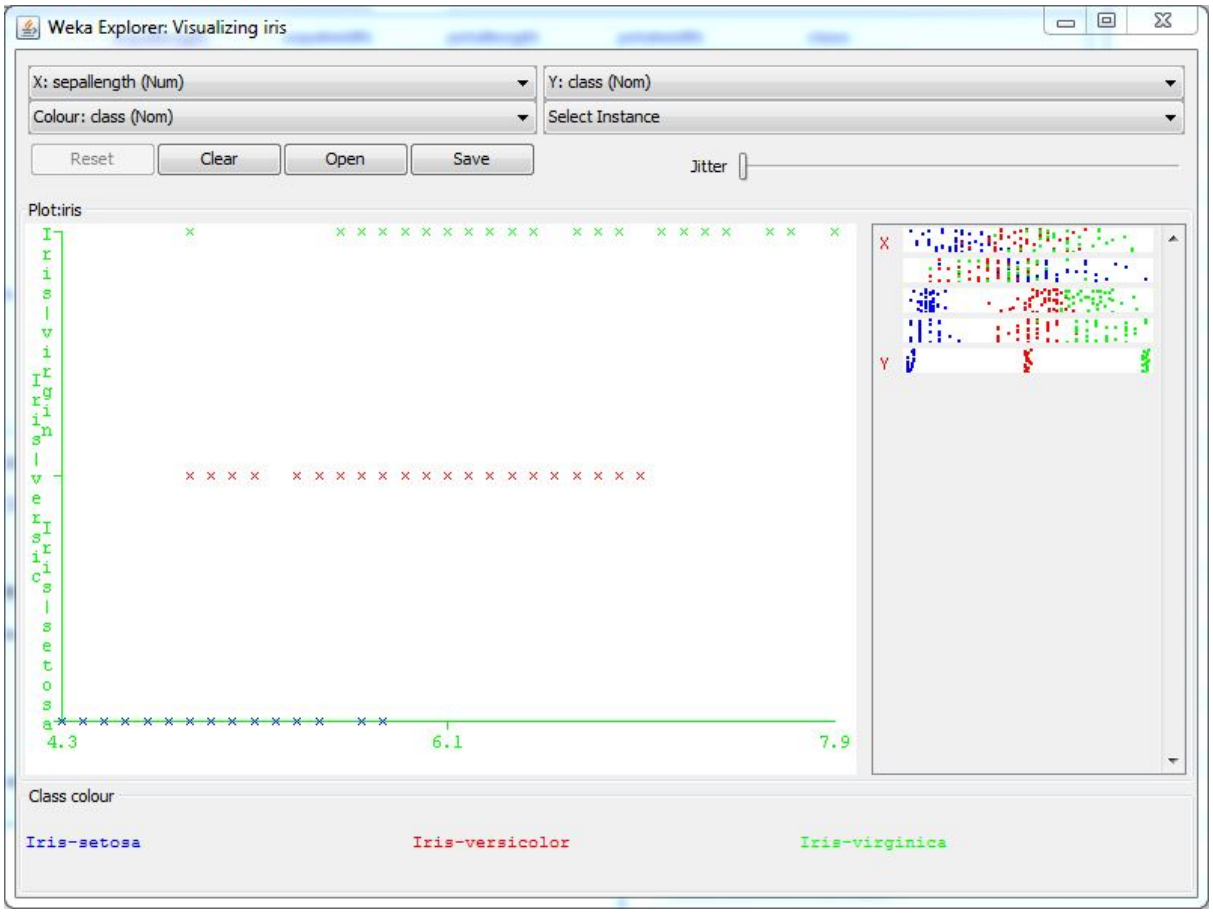


Figura 2.10: Captura Weka Visualizing.

Por último *Weka* tampoco está orientado a grandes bases de datos y aunque se ha posicionado con fuerza en el escenario de los paquetes de minería de datos presenta serias limitaciones para analizar bases de datos de tamaño considerable.

Por tanto la principal ventaja que aporta nuestro software respecto a otros que hay en el mercado es el uso de distintas métricas mixtas, la posibilidad de hacer una representación de los datos en 3D en la qué el usuario puede interactuar y la escalabilidad de la implementación que permite tratar bases de datos de cierta envergadura (se han realizado pruebas con 10.000 datos).

2.7. Presentación general de KLASS

Este proyecto se enmarca dentro de una línea de trabajo y estudio más general que trata la combinación de métodos estadísticos y de la Inteligencia Artificial para el Descubrimiento de Conocimiento *(Knowledge Discovery y Data Mining)* en dominios reales de estructura difícil y no muy conocidos, llamados dominios poco estructurados, impulsado por Karina Gibert del Departamento de Estadística e Investigación Operativa de la UPC desde los 90 y donde el uso indistinto de información numérica, categórica y conocimiento declarativo es una práctica habitual y que se ajusta bien a la realidad de los dominios poco estructurados.

En concreto lo que este proyecto aporta a **Java-KLASS** es la implementación de un nuevo módulo de clasificación con métodos basados en densidad que, entre otras cosas, contribuye a mejorar el reconocimiento de clusters.

2.7.1. Antecedentes

Fruto del trabajo y estudio sobre la clasificación de dominios poco estructurados surgió la primera versión de **KLASS**, un sistema orientado a la clasificación automática de dominios poco estructurados implementado en Lisp sobre el sistema operativo Solaris (UNIX) de SUN. Este paquete ha ido evolucionando de forma continuada desde la aparición de la primera versión que formaba parte de la tesis doctoral [Gibert 94] de Karina Gibert, y ha sido objeto de otros PFC, tanto de la Diplomatura de Estadística como de las Ingenierías en Informática de la UPC y de la UIB.

Después de todo el proceso de crecimiento y más de una década de utilización de **KLASS** surgió la necesidad de que **KLASS** pudiera funcionar sobre diferentes sistemas operativos por diversas razones, como son:

- Las dificultades para mantener la licencia LISP en la UPC.
- El elevado coste de mantenimiento de las máquinas Unix, y la actual política de utilización de PCs en la UPC y en general fuera de la UPC.
- La colaboración con equipos externos hace que se tenga que poder enviar el programa y que funcione en cualquier PC, sin la necesidad de que los investigadores que quieran utilizarlo como usuarios tengan una licencia de LISP.
- La incomodidad de tener que enviar todo el código fuente, si queremos que un usuario externo utilice **KLASS**, en el supuesto que disponga de LISP, ya que LISP es un lenguaje interpretado y no se pueden generar ejecutables.

Así pues, se planteó hacer una nueva versión de **KLASS** en Java, **Java-KLASS**, y de esta manera conseguir un sistema más portable.

Actualmente **KLASS** es un paquete que se está utilizando en diferentes entornos y por este motivo uno de los objetivos principales que se marcaron era que la nueva versión debía mantener la funcionalidad actual que ofrece el sistema como mínimo.

Por otra parte debido a la magnitud de la última versión LISP de **KLASS** (KLASS + tiene casi 28.000 líneas de código más las casi 10.000 líneas de Columbus y las casi 3.000 líneas de CIADEC) se decidió que el nuevo sistema **Java-KLASS** se debería desarrollar por partes.

Actualmente **Java-KLASS** [Gibert & Nonell 2008] y [Gibert & Nonell 2005 b] incluye 3MB de código Java y ofrece funcionalidades para realizar:

- Representación de matrices de datos.
- Gestión de bases de conocimiento.
- Estadística descriptiva extensa de los datos.
- Clasificación automática.
- Razonamiento automático.
- Interoperabilidad de métodos.
- Análisis dinámico [Gibert et al. 2010].
- Interpretación de las clases [Gibert et al. 2012] [Gibert et al. 2013], gráficamente [Gibert & Conti 2014] [Gibert et al. 2008] y conceptualmente [Gibert 2014].

A continuación se presenta la cronología de todas las versions de **KLASS** que se han desarrollado.

2.7.2. Cronología de KLASS

- Feb. 1991 **KLASS v0**. Tesina Karina Gibert. "KLASS. Estudi d'un sistema d'ajuda al tractament estadístic de grans bases de dades". Clasifica matrices de datos heterogéneas con la distancia mixta. [Gibert 91]
- Nov. 1994 **KLASS v1**. Tesis Karina Gibert. "L'ús de la informació simbòlica en l'automatització del tractament estadístic de dominis poc estructurats". És una ampliación de **KLASS v0**. Incorpora la clasificación basada en reglas. [Gibert 94]
- Jul. 1996 **KLASS v1.1**. PFC Xavier Castillejo. Incorpora a **KLASS.v1** una interfaz de ventanas independiente con un sistema que facilita el uso de KLASS desde SUN y desde PC a usuarios que desconocen Lisp y UNIX. Denominaremos **xcn.KLASS** al núcleo Lisp de esta nueva versión y **xcn.i** a la interfaz C. [Castillejo 1996]
- Oct. 1997 **jj.KLASS**. PFC Juan José Marquez y Juan Carlos Martín. Incorpora a la versión **KLASS.v1** nuevas opciones para el tratamiento de missings, la posibilidad de trabajar con objetos ponderados e implementa un test no paramétrico de comparación de clasificaciones [Márquez 97].
- Set. 1999 **KLASS v1.2**. PFC Xavier Tubau (versión β). Incorpora a la versión **xcn.KLASS** el módulo de comparación de clasificaciones de **jj.KLASS**, la métrica mixta de Ralambondrainy [Ralambondrainy 95] [Ralambondrainy 88] y prepara la formulación de tres más para su posterior implementación. Denominaremos **xt.KLASS** al núcleo Lisp de esta nueva versión y **xt.i** a la interfaz C asociada. [Tubau 1999]
- 1999-2000 **KLASS+ v1**. PFC Sílvia Bayona. Fusión definitiva de la versión **xt.KLASS** y **jj.KLASS**. Incorpora además un módulo nuevo de análisis descriptiva de datos, así como de las clases resultantes, reorientando KLASS hacia un propósito más general y menos especializado. Denominaremos **sbh.KLASS** al núcleo Lisp de esta nueva versión y **sbh.i** a la interfaz C asociada. [Bayona 2000]

- 2000-2002 **KLASS+ v2**. PFC Josep Oliveras. Agrega a **sbh.KLASS** las métricas mixtas pendientes (Gower [Gower 71] [Gower 66] [Gower 67], Gowda-Diday [Gowda & Diday 92] [Gowda & Diday 91] e Ichino-Yaguchi [Ichino & Yaguchi 94] [Ichino & Yaguchi 89]). Denominaremos **joc.KLASS** a esta nueva versión. [Oliveras 2002]

- 2000-2003 **jr.KLASS+**. Tesis doctoral Jorge Rodas. Integra **KLASS+ v.2** y **Columbus** [Rodas 2003].

- 2000-2003 Investigación Anna Salvador y Fernando Vázquez. Desarrollo de **CIADEC** [Gibert & Salvador 2000] [Vázquez & Gibert 2002].

- 2002-2003 **Java-KLASS v0**. PFC Mª del Mar Colillas. Versión Java del módulo de análisis descriptiva e integración con **CIADEC** y **Columbus**.

  - 2003-2005 **Java-KLASS v0.22**. Colaboración con Mar Colillas. Ampliación del módulo de análisis descriptiva e introducción de herramientas de gestión de datos *(definición de ordenaciones en los informes, posibilidad de varias matrices de objetos en el sistema simultaneamente, cambio de matriz activa).*

  - 2005-2006 **Java-KLASS v1.0**. Colaboración con Mar Colillas. Incluye la lectura y visualización de dendrogramas aislados, así como la generación de particiones a partir de ellos.

- 2006-2007 **Java-KLASS v2.0**. PFC Jose Ignacio Mateos. Ampliación de **Java-KLASS** con un módulo de cálculo de distancias para diferentes tipos de matrices de datos, incluyendo las que combinan información cualitativa y cuantitativa, tratamiento de missings y creación de submatrices.

- 2006-2007 **Java-KLASS v3.0**. PFC Roberto Tuda. Incluye un módulo de clasificación automática para métodos jerárquicos, utilizando todas las distancias implementadas en la v2.0 y una opción para estudiar agregaciones de objetos paso a paso. Se crea la opción de poder seleccionar el directorio de trabajo por defecto. Se le agrega la opción de añadir y grabar objetos con peso.

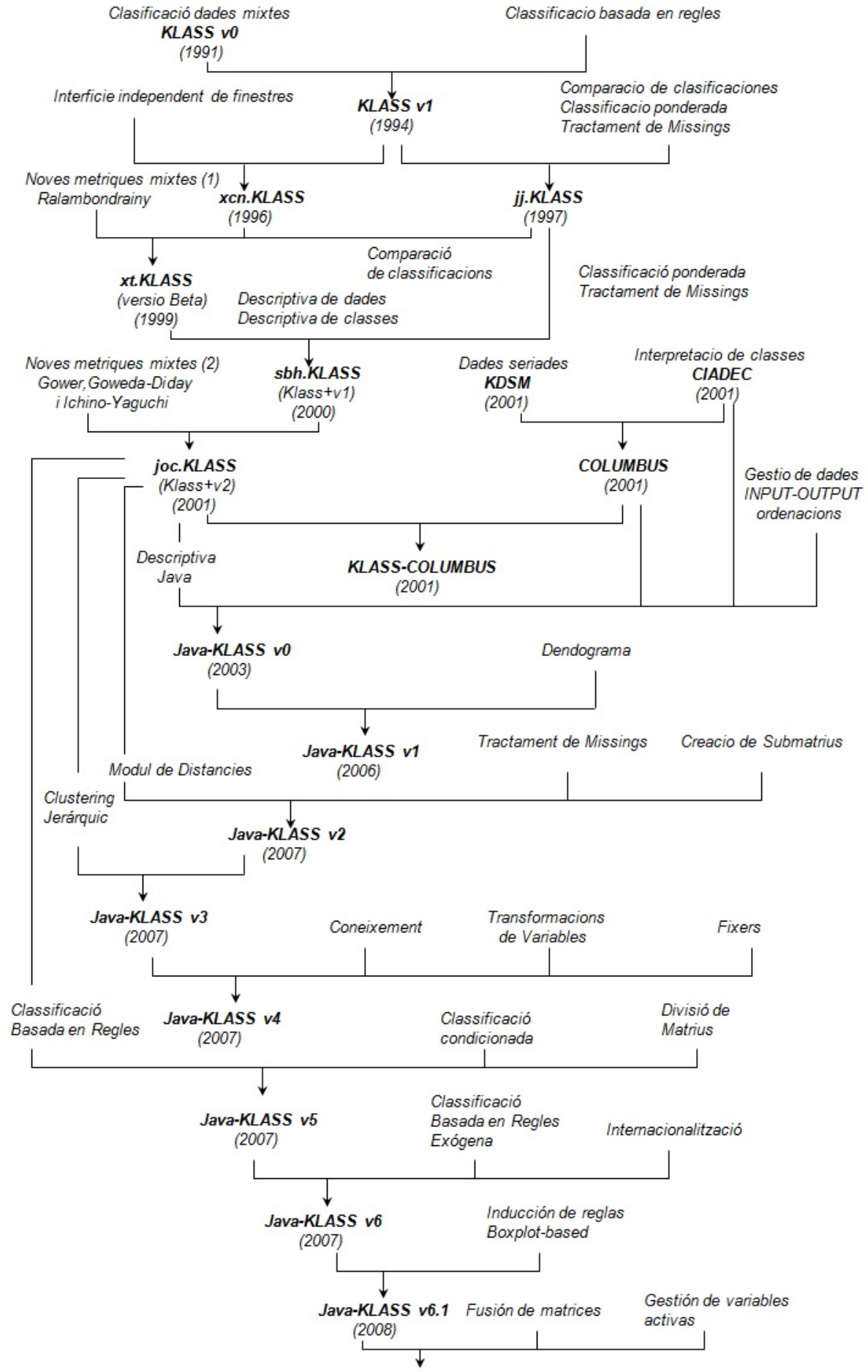
- 2006-2007 **Java-KLASS v4.0**. PFC Laia Riera Guerra. Introducción, gestión y evaluación de Bases de Conocimiento. Ampliación de **Java-KLASS** con un módulo de transformación de variables que permite discretizaciones, recodificaciones y cálculos aritméticos con variables numéricas. Por último, esta versión incluye la definición de submatrices via filtros lógicos sobre los objetos, la edición de metainformación de las variables de la matriz, eliminación de variables e importación de ficheros en formato .dat estándar.

- 2007 **Java-KLASS v5.0**. PFC Andreu Raya. Incluye la clasificación condicionada, la clasificación basada en reglas y funcionalidades de división de la base de datos y de gestión de árboles de clasificación *(o dendrogramas)* asociados a las diferentes matrices de datos.

- 2007 **Java-KLASS v6.0**. Trabajo Investigación Tutelada Alejandro García. Clasificación basada en reglas exógena. Internacionalización y Localización a tres idiomas (Catalán, Inglés, Castellano). Fusión de matrices.

- 2008 **Java-KLASS v6.4**. Trabajo de master Alfons Bosch Sansa, Patricia García Giménez, Ismael Sayyad Hernando. Boxplot-based discretization, Boxplot-based Induction rules.

- 2008. Tesis doctoral Alejandra Perez. Caracterización por condicionamientos sucesivos, metodología que induce automáticamente conceptos asociados a las clases descubiertas.
- 2008. Tesis doctoral Gustavo Rodriguez. Clasificación basada en reglas por estados que permite análisis de sistemas dinámicos.
- 2008: **Java-KLASS v7.0**: TrT Alejandro García Rudolph. Fusió de matrius i gestió de variables actives.
- 2009: **Java-KLASS v8.**: Tesi de master d'Ester Lozano. Criteris Best Local Concept and no close world assumption del CCEC. PT Alejandro García Rudolph. Classificació basada en regles per estats.
- 2010: **Java-KLASS v8.1**: Practica SISPD. Narcis Maragall. Boxplot Based Induction Rules.
- 2012: **Java-KLASS v8.6**: Practica SISPD. Pau. metodología CCEC.
- 2012: **Java-KLASS v9**: Practica SISPD. Marco Villegas. Criteris CCEC.
- 2013 **Java-KLASS v10**: Practica SISPD. Emili Boronat. Traffic Light Panel.
- 2014: **Java-KLASS v11**: Proyecto de Fin de Carrera Ingeniería Informática FIB. Sheila Mollá. DBSCAN, OPTICS, 3D Visualization.
- 2014: **Java-KLASS v12**: Practica SISPD. Jonathan Moreno. Optimización de expresiones lógicas.


Clasificació dades mixtes
KLASS v0
(1991)
Classificacio basada en regles
Interficie independent de finestres
KLASS v1
(1994)
Comparacio de clasificaciones
Classificacio ponderada
Tractament de Missings
Noves metriques mixtes (1)
Ralambondrainy
xcn.KLASS
(1996)
jj.KLASS
(1997)
Comparació
de classificacions
xt.KLASS
(versio Beta)
(1999)
Descriptiva de dades
Descriptiva de classes
Classificació ponderada
Tractament de Missings
Noves metriques mixtes (2)
Gower, Goweda-Diday
i Ichino-Yaguchi
sbh.KLASS
(Klass+v1)
(2000)
Dades seriades
KDSM
(2001)
Interpretacio de classes
CIADEC
(2001)
joc.KLASS
(Klass+v2)
(2001)
COLUMBUS
(2001)
Gestio de dades
INPUT-OUTPUT
ordenacions
Descriptiva
Java
KLASS-COLUMBUS
(2001)
Java-KLASS v0
(2003)
Dendograma
Java-KLASS v1
(2006)
Tractament de Missings
Creacio de Submatrius
Modul de Distancies
Clustering
Jerárquic
Java-KLASS v2
(2007)
Java-KLASS v3
(2007)
Coneixement
Transformacions
de Variables
Fixers
Classificació
Basada en Regles
Java-KLASS v4
(2007)
Classificació
condicionada
Divisió de
Matrius
Java-KLASS v5
(2007)
Classificació
Basada en Regles
Exógena
Internacionalització
Java-KLASS v6
(2007)
Inducción de reglas
Boxplot-based
Java-KLASS v6.1
(2008)
Fusión de matrices
Gestión de variables
activas

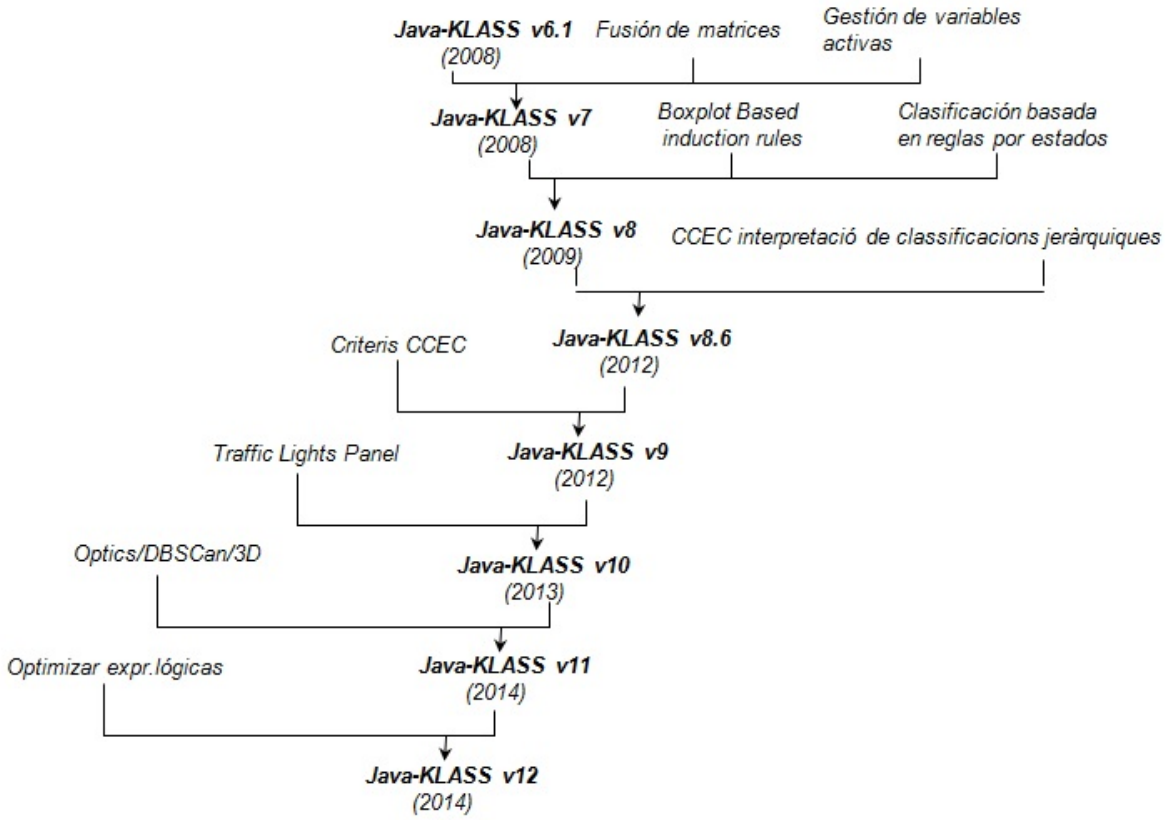


Figura 2.11: Cronología de **KLASS**

Capítulo 3

Métodos

En este apartado se hará una introducción de los dos algoritmos implementados en este proyecto.

3.1. DBSCAN

Como ya hemos dicho este tipo de algoritmos se basan en el concepto de densidad de la vecindad de un punto y miden el número de puntos alcanzables desde éste teniendo en cuenta un radio concreto.

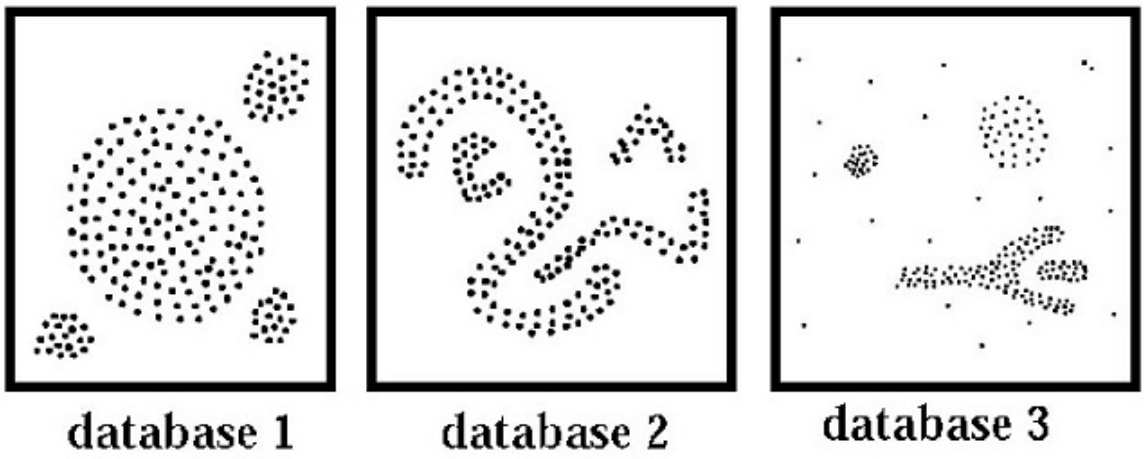


Figura 3.1: Ejemplo bases de datos.

Al mirar los conjuntos de las diferentes muestras de puntos representados en la Figura 3.1, se pueden detectar fácilmente y sin ambigüedades clusters de puntos y puntos de ruido que no pertenecen a ninguno de esos clusters.

La razón principal por la que reconocemos los clusters es que dentro de cada grupo hay una densidad típica de puntos que es considerablemente mayor que en el exterior del cluster. Además, la densidad dentro de las áreas de ruido es más baja que la densidad en cualquiera de los clusters.

En lo que sigue, trataremos de formalizar esta noción intuitiva de "clusters" y "ruido" en una base de datos D de algún espacio k-dimensional S. La idea clave es que para cada punto de un cluster, la vecindad de un radio dado (ε) tiene que contener un mínimo número de puntos (ν). Es decir, la densidad de los vecinos tiene que exceder un umbral.

La vecindad de un punto se determina por la elección de una función de distancia para dos puntos p y q (denotado por $dist(p,q)$) y el radio (ε). La generalización del método que se propone funciona con cualquier función de distancia de manera que se pueda elegir la más apropiada para cada aplicación dada. Para la presentación del método y a título estrictamente conceptual, todos los ejemplos que se mostrarán a continuación, serán en el espacio 2D utilizando la distancia Euclídea.

Definición 1. (ε-vecindad de un punto) La ε-vecindad de un punto p, denotado por N_ε(p), se define por:

$$N_\varepsilon(p) = \{q \in D | dist(p,q) \leq \varepsilon\}$$

Partiendo de la idea de que un cluster contiene una zona con alta densidad de puntos, un enfoque ingenuo podría requerir para cada punto en un cluster, que haya al menos un número mínimo (ν) de puntos en la ε-vecindad de ese punto. Sin embargo, este enfoque falla porque hay dos tipos de puntos en un cluster, los puntos dentro del cluster (*puntos nucleares o core points*) y los puntos de la frontera del cluster (*puntos fronterizos o border points*). En general, la ε-vecindad de un punto de la frontera contiene significativamente menos puntos que la ε-vecindad de un punto *core*. Por lo tanto, tendríamos que establecer el número mínimo de puntos a un valor relativamente bajo con el fin de incluir a todos los puntos que pertenecen al mismo cluster. Este valor, sin embargo, no será característico para el cluster correspondiente - en particular por la presencia de ruido. Por lo tanto, se requiere que para cada punto p en un cluster C, haya un punto q en C tal que p esté dentro de la ε-vecindad de q y N_ε(q) contenga al menos ν puntos.

Definición 2. (directly density-reachable) Dados el radio ε y el número mínimo de puntos ν, se dice que un punto p es *densamente alcanzable de manera directa* desde un punto q si:

1. $p \in N_\varepsilon$(q), esto es, p pertenece a la región de vecindad de q, y
2. $|N_\varepsilon(q)| \geq \nu$ (q está en zona densa, condición de punto *core*).

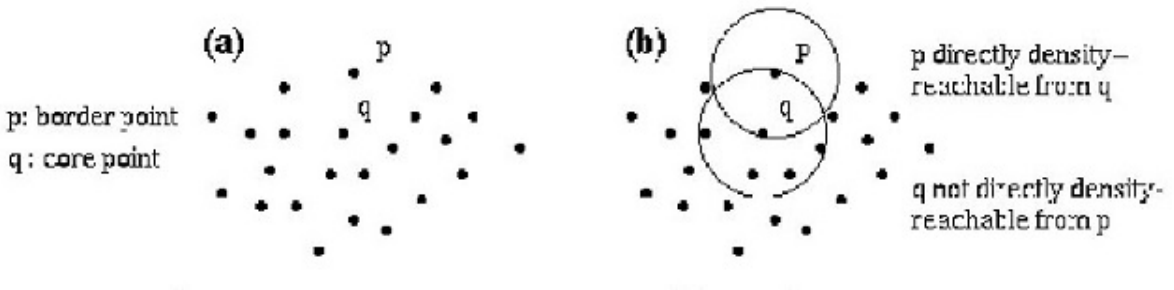


Figura 3.2: Puntos core y puntos fronterizos.

Obviamente, la relación *densamente alcanzable de manera directa* es simétrica para puntos de *core*. En general, sin embargo, no lo es entre un punto de *core* y un punto fronterizo. La Figura 3.2 (b) muestra el caso asimétrico.

Por ello se define una nueva relación de simetría.

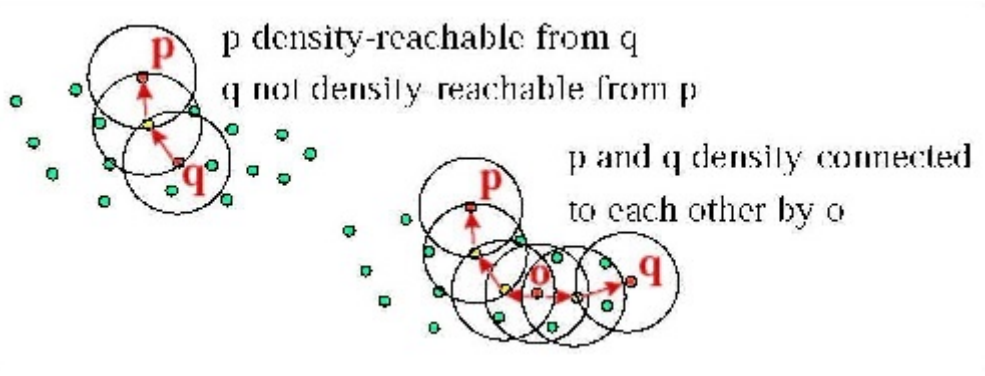


Figura 3.3: Density-reachable y density-connected.

Definición 3. (density-reachable) Dados el radio ε y el número mínimo de puntos ν, un punto p es *densamente alcanzable* desde un punto q si hay una cadena de puntos $p_1, ..., p_n, p_1 = q, p_n = p$ tal que p_{i+1} es *densamente alcanzable de manera directa* desde p_i.

Densamente alcanzable es una extensión canónica de *densamente alcanzable de manera directa*. Esta relación es transitiva, pero no es simétrica. La Figura 3.3 representa las relaciones de una muestra de puntos y, en particular, el caso asimétrico. Aunque no es simétrica en general, es obvio que *densamente alcanzable* es simétrica para los puntos de *core*.

Dos puntos fronterizos del mismo cluster C, posiblemente, no son *densamente alcanzables* desde uno al otro debido a que la condición de punto *core* podría no mantenerse ente los dos. Sin embargo, debe haber un punto nuclear en C de la que ambos puntos fronterizos de C son *densamente alcanzables*. Por lo tanto, se introduce la noción de *densamente conectado* que cubre esta relación de puntos fronterizos.

Definición 4. (density-connected) Dados el radio ε y el número mínimo de puntos ν, un punto p está *densamente conectado* a un punto q si hay un punto o de tal manera que ambos, p y q son *densamente alcanzables* desde o.

Densamente conectado es una relación simétrica. Para los puntos *densamente alcanzables*, la relación *densamente conectado* es también reflexiva.

Ahora, estamos en condiciones de definir nuestra noción basada en la densidad de un cluster. Intuitivamente, un cluster se define como un conjunto de puntos *densamente conectados* que es máximo con respecto a *densamente alcanzable*. El ruido se define en relación a un conjunto dado de clusters. El ruido es simplemente el conjunto de puntos en D que no pertenecen a ninguno de esos clusters.

Definición 5. (cluster) Dados el radio ε, el número mínimo de puntos ν y sea un conjunto de datos D, se dice que un grupo C es un subconjunto no vacío de D que satisface las siguientes condiciones:

1. $\forall p, q$: si $p \in C$ y q es *densamente alcanzable* desde p, dados ε y ν, entonces $q \in C$. (Maximitud)
2. $\forall p, q \in C$: q está *densamente conectado* a p dados ε y ν. (Conectividad)

Definición 6. (noise) Sean $C_1, ..., C_K$ los grupos del conjunto de datos D dados los parámetros ε_i y ν_i, $i = 1, ..., K$.

A continuación se define el *ruido* como el conjunto de puntos en la base de datos D que no pertenecen a ningún grupo C_i, es decir, el $ruido = \{p \in D | \forall i : p \notin C_i\}$

Hay que tener en cuenta que un grupo C, dados ε y ν, contiene al menos ν puntos debido a las siguientes razones. Puesto que C contiene al menos un punto p, p debe ser *densamente conectado* a sí mismo vía otro punto o (que puede ser igual a p). Por lo tanto, al menos o tiene que satisfacer la condición de punto *core* y, en consecuencia, la ε-vecindad de o contiene al menos ν puntos.

Los siguientes lemas son importantes para la validación de nuestro algoritmo de clustering. Intuitivamente, se establece lo siguiente. Dados los parámetros ε y ν, podemos descubrir un cluster en un enfoque de dos pasos. En primer lugar, elegir un punto arbitrario de la base de datos que satisfaga la condición de punto de *core* como semilla. En segundo lugar, recuperar todos los puntos que son *densamente alcanzables* desde la semilla obteniendo así el cluster que lo contiene.

Lema 1. Sea p un punto en D y $|N_\varepsilon(p)| \geq \nu$. Entonces el conjunto $O = \{o | o \in D$ y o es *densamente alcanzable* desde $p\}$ es un cluster dados ε y ν.

No es obvio que un cluster C esté únicamente determinado por cualquiera de sus puntos *core*. Sin embargo, cada punto en C es *densamente alcanzable* desde cualquier punto *core* de C y, por tanto, un grupo C contiene exactamente los puntos que son *densamente alcanzables* desde un *core* arbitrario de C.

Lema 2. Sea C un grupo y p cualquier punto de C con $|N_\varepsilon(p)| \geq \nu$. Entonces C es igual al conjunto $O = \{o | o$ es *densamente alcanzable* desde p, dados ε y $\nu\}$.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise), es un algoritmo basado en la densidad de agrupamiento espacial de aplicaciones con ruido. Es un algoritmo de clustering propuesto por Martin Ester, Hans-Peter Kriegel, Jörg Sander y Xiaowei Xu (1996) [Ester 96].

Este algoritmo, encuentra un número de clusters a partir de la distribución de densidad estimada de los nodos correspondientes. La idea es hacer crecer un cluster siempre y cuando la densidad en el entorno del objeto sea suficientemente alta (esto se determina a partir de un umbral de densidad mínimo (ν) que se configura como un parámetro de entrada del algoritmo).

Lo ideal sería conocer los parámetros ε y ν adecuados de cada cluster y al menos un punto del respectivo cluster. Entonces, podríamos recuperar todos los puntos que son *densamente alcanzables* desde ese punto, utilizando los parámetros correctos. Pero no hay una manera fácil de esta información por adelantado para todos los clusters de la base de datos. Sin embargo, hay una heurística simple y eficaz (presentada en la sección 3.1.2) para determinar los parámetros ε y ν del cluster menos denso en la base de datos. Por lo tanto, **DBSCAN** utiliza los valores globales de ε y ν, es decir, los mismos valores para todos los grupos. Los parámetros del cluster menos denso son buenos candidatos para estos valores de los parámetros globales, especificando la densidad más baja que no se considera ruido.

Este método permite la detección de clusters que tengan formas arbitrarias y filtrar grupos que están separados por regiones de baja densidad de objetos (ruido). A diferencia de otras familias de métodos de clustering, estos métodos no producen particiones completas, y pueden dejar elementos de las zonas menos densas sin clasificar, fuera de toda clase.

Algunas de las aplicaciones de DBSCAN realizadas con éxito son: la creación de perfiles de usuarios en Internet mediante la agrupación de sesiones Web, o el agrupamiento de bases de datos de imágenes en histogramas en color facilitando la búsqueda de imágenes similares, etc.

Los parámetros fundamentales son ε (radio máximo de la vecindad) y ν (número de puntos mínimo requerido para formar un cluster).

El algoritmo empieza con la elección de un punto de partida arbitrario que no ha sido visitado aún. Se recuperan todos los puntos *densamente alcanzables* desde este punto con respecto a ε y ν. Si contiene el número suficiente de puntos, se inicia un cluster, en caso contrario se etiqueta como ruido.

Si un punto se encuentra en la parte densa de un cluster, su ε-vecindad es también parte de ese cluster. Por tanto se suman todos los puntos que se encuentran dentro de la ε-vecindad, y de igual forma sus ε-vecindades cuando ellos también son densos.

El proceso continúa hasta que el cluster *densamente alcanzable* se ha encontrado completamente. De nuevo, se recupera un punto no visitado y se procesa, para el descubrimiento de otro cluster o ruido.

A continuación, presentamos una versión básica del **DBSCAN** omitiendo los detalles del tipo de datos y generación de información adicional acerca de los clusters:

3.1.1. Pseudocódigo

Algorithm 1 DBSCAN (SetOfPoints, ε, ν)

```
// SetOfPoints is UNCLASSIFIED
ClusterId := nextId(NOISE)
for i = 1 to SetOfPoints.size do
    Point := SetOfPoints.get(i)
    if Point.ClId = UNCLASSIFIED then
        if ExpandCluster(SetOfPoints, Point, ClusterId, ε, ν) then
            ClusterId := nextId(ClusterId)
        end if
    end if
end for
```

SetOfPoints es o bien toda la base de datos o un cluster descubierto en una ejecución anterior. ε y ν son los parámetros de densidad global determinados, ya sea manualmente o de acuerdo con las heurísticas que se presentan en la sección 3.1.2. La función *SetOfPoints.get(i)* devuelve el elemento i-ésimo de *SetOfPoints*. La función más importante utilizada por **DBSCAN** es *ExpandCluster* que se presenta a continuación:

Algorithm 2 ExpandCluster (SetOfPoints, Point, ClId, ε, ν):Boolean

```
function ExpandCluster(SetOfPoints, Point, ClId, ε, ν)
    seeds := SetOfPoints.regionQuery(Point, ε)
    if seeds.size < ν then // no core point
        SetOfPoints.changeClId(Point, NOISE)
        return False
    else// all points in seeds are density-reachable from Point
        SetOfPoints.changeClIds(seeds, ClId)
        seeds.delete(Point)
        while seeds <> Empty do
            currentP := seeds.first()
            result := SetOfPoints.regionQuery(currentP, ε)
            if result.size ≥ ν then
                for i = 1 to result.size do
                    resultP := result.get(i)
                    if resultP.ClId IN {UNCLASSIFIED, NOISE} then
                        if resultP.ClId = UNCLASSIFIED then
                            seeds.append(resultP)
                        end if
                        SetOfPoints.changeClId(resultP, ClId)
                    end if// UNCLASSIFIED or NOISE
                end for
            end if// result.size ≥ ν
            seeds.delete(currentP)
        end while// seeds <> Empty
        return True
    end if
end function
```

Una llamada a *SetOfPoints.regionQuery(Point, ε)* devuelve la ε-vecindad de *Point* en *SetOfPoints* como una lista de puntos. La función *regionQuery* puede ser llamada de manera eficiente por los métodos de acceso espaciales tales como los árboles R* [Beckmann et al. 1990], que se supone para están disponibles para el procesamiento eficiente de varios tipos de consultas espaciales [Brinkhoff et al. 1994]. La altura de un árbol R* es $O(\log n)$ para una base de datos de n puntos en el peor caso y una consulta a una región pequeña tiene que atravesar sólo un número limitado de trayectorias en el árbol R*. Dado que se espera que las ε-vecindades sean pequeñas en comparación con el tamaño de todo el espacio de los datos, la complejidad media del tiempo de ejecución de una sola consulta a *regionQuery* es $O(\log n)$. Para cada uno de los n puntos de la base de datos, como máximo se consulta una vez a *regionQuery*. Así, la complejidad media del tiempo de ejecución de DBSCAN es $O(n \cdot \log n)$.

El *ClId* (clusterId) de los puntos que se han marcado como NOISE (ruido) puede ser cambiado más tarde, si son *densamente alcanzables* desde algún otro punto de la base de datos. Esto sucede por los puntos frontera de un cluster. Esos puntos no se añaden a la lista *seeds* porque ya sabemos que un punto con un *ClId* de NOISE no es un punto *core*. Añadiendo esos puntos a *seeds* se harían nuevas consultas de zonas adicionales que no producirían nuevas respuestas.

Si dos clusters C_1 y C_2 están muy cerca uno del otro, podría suceder que algún punto p perteneciera a ambos, C_1 y C_2. Entonces p debe ser un punto fronterizo en ambos grupos porque de lo contrario C_1 sería igual a C_2 ya que utilizamos parámetros globales. En este caso, el punto p se asigna al primer cluster descubierto. Con excepción de estas situaciones, el resultado de DBSCAN es independiente del orden en el que los puntos de la base de datos son visitados por el Lema 2.

3.1.2. Determinación de los parámetros ε y ν

En esta sección, se presenta la heurística para determinar los parámetros ε y ν de los algoritmos DBSCAN y OPTICS. En realidad corresponden al radio (ε) y masa crítica (ν) del cluster menos denso en la base de datos. Esta heurística se basa en la siguiente observación [Ester 96]. Sea d la distancia de un punto p a su k-ésimo vecino más cercano, la d-vecindad de p contiene exactamente $k+1$ puntos para casi todos los puntos p. La d-vecindad de p contiene más de $k+1$ puntos sólo si varios puntos tienen exactamente la misma distancia d hasta llegar a p, que es bastante improbable. Además, cambiando k para un punto en un cluster no da lugar a grandes cambios de d. Esto sólo ocurre si los k-ésimos vecinos más cercanos de p para $k = 1, 2, 3, ...$ se encuentran aproximadamente en una línea recta, que, en general, no es cierto para un punto en un cluster.

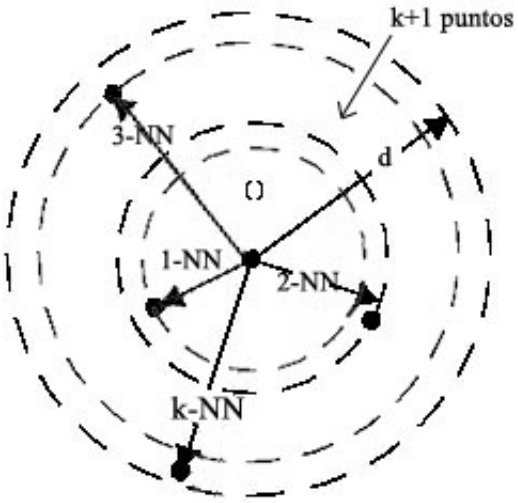


Figura 3.4: K-NN.

Para un k dado definimos una función $k-dist$ desde la base de datos D de los números reales, mapeando cada punto a la distancia de su vecindad más cercana de orden k.

$k-dist(i) = d(i, k-KNN(i))$, siendo $k-NN(i)$ el vecino más cercano de i.

Al ordenar los puntos de la base de datos en orden descendente de sus valores $k-dist$, el gráfico de esta función da algunas pistas acerca de la distribución de la densidad en la base de datos. Llamamos a este gráfico el *gráfico sorted k-dist*. Si elegimos un punto arbitrario p, establecemos el parámetro ε a $k-dist$(p) y establecemos el parámetro ν a k, todos los puntos con un valor igual o menor que $k-dist$ serán puntos *core*. Si pudiéramos encontrar un *punto umbral* con el valor $k-dist$ máximo en el cluster menos denso de D tendríamos los valores deseados de los parámetros. El punto de umbral es el primer punto del "valle" del *gráfico sorted k-dist* (ver Figura 3.5). Ya que desde este valle hacia la izquierda es dónde las distancias a los $k-NN$ se empiezan a hacer mucho mayores, estos puntos corresponden a ruido y se debe considerar un ε cerca del valle. El resto de puntos (a la derecha del umbral) se asignan a algún cluster.

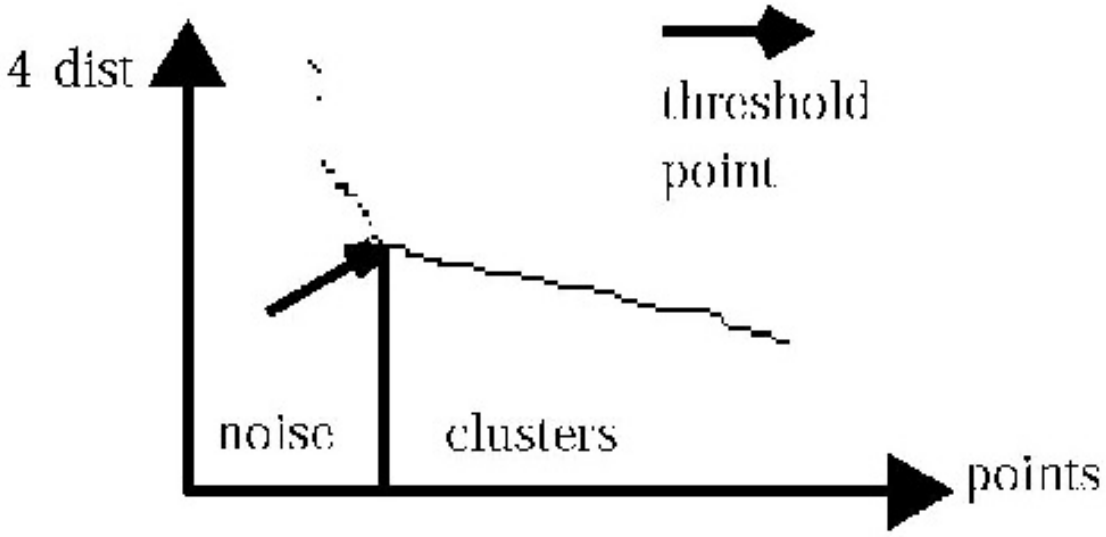


Figura 3.5: sorted 4-dist graph for sample database 3.

En general, es muy difícil detectar el primer "valle" automáticamente, pero es relativamente sencillo para un usuario ver este valle en una representación gráfica.

DBSCAN necesita dos parámetros, ε y ν. Sin embargo, los experimentos realizados por [Ester 96] indican que los gráficos $k-dist$ para $k > 4$ no difieren significativamente de la gráfica $4-dist$ y, además, necesitan considerablemente más cálculos. Por lo tanto, eliminamos el parámetro ν estableciéndolo en 4 para toda la base de datos (para los datos de 2 dimensiones). Se propone el siguiente enfoque interactivo para la determinación del parámetro ε de DBSCAN:

- El sistema calcula y muestra el gráfico $4-dist$ para la base de datos.
- Si el usuario puede estimar el porcentaje de ruido, este porcentaje se introduce y el sistema propone el punto umbral de la misma.
- El usuario acepta el umbral o propone otro punto. El valor $4-dist$ del punto umbral se usa como valor de ε para DBSCAN.

3.1.3. Ventajas del método DBSCAN

- No requiere especificar a priori el número de clusters en los datos, como en K-means.
- Puede encontrar grupos de topología arbitraria. Se puede incluso encontrar un cluster completamente rodeado por (pero no conectado a) un grupo diferente. Debido al parámetro ν, se reduce el llamado efecto de encadenamiento o chaining (diferentes grupos conectados por una fina línea de puntos mezclados en una sola clase).
- Tiene la capacidad de identificar la presencia de ruido.
- Requiere sólo dos parámetros y es sobre todo insensible al orden de los puntos en la base de datos. (Sin embargo, los puntos que se encuentran en el borde de dos clusters distintos, podrían cambiar la pertenencia al cluster si se cambia el orden de visita de los puntos).

- Está diseñado para su uso con bases de datos que pueden acelerar las consultas a *regionQuery*, por ejemplo, usando un árbol R*.

3.1.4. Desventajas del método DBSCAN

- La calidad de DBSCAN depende de la medida de distancia utilizada en la función *regionQuery*. La distancia métrica más común usada es la distancia Euclídea ordinaria, útil en matrices de datos numéricos exclusivamente. Especialmente para los datos de alta dimensión, este indicador es casi inútil debido a la llamada "maldición de la dimensionalidad" (curse of dimensionality), por lo que es difícil encontrar un valor adecuado para ε. Este efecto, sin embargo, también está presente en cualquier otro algoritmo basado en la distancia Euclídea. En este proyecto, se introducirá un parámetro al algoritmo que permita cambiar la función de distancia según la naturaleza de los datos.

- DBSCAN no puede agrupar bien conjuntos de datos con grandes diferencias en las densidades en distintas zonas de la nube de puntos, ya que la combinación $\nu-\varepsilon$ no se puede escoger adecuadamente para todos los grupos.

3.2. OPTICS

Un inconveniente de DBSCAN es que está optimizado para generar clusters de la misma densidad. Si los objetos forman parte de clusters de diferentes densidades puede tener dificultades para localizarlos.

Para solventar estos problemas se creó el algoritmo OPTICS (Ordering Points To Identify the Clustering Structure) propuesto por Ankerst 1999 [Ankerst 99].

OPTICS realiza un ordenamiento de los objetos representándolos de forma que emerja la estructura en clusters de la nube de puntos, según la densidad. El resultado de OPTICS es una representación gráfica conocida como Reachability Plot que da información sobre la estructura intrínseca de la base de datos y permite observar a posteriori cuantas clases hay.

Los objetos se ordenan de manera que los puntos más próximos espacialmente se convierten en vecinos.

Un cluster será un conjunto de objetos ubicados en una zona de alta densidad (que en un radio ε alrededor de un objeto haya al menos ν objetos).

Un conjunto *densamente conectado* (en el mismo sentido que se usaba en DBSCAN) reúne un número suficiente de objetos (ν) cercanos según la distancia ε y es la base para construir un cluster.

OPTICS se basa en un principio que se cumple siempre, que es que dado ν y $\varepsilon_2 < \varepsilon_1$, un cluster construido con ν objetos y radio ε_1 contiene todos los clusters que se pueden construir con estos elementos y con los parámetros ν y ε_2. Este hecho se ilustra en la Figura 3.6, donde C_1 y C_2, son clusters basados en densidad con respecto a ε_2 y C es un cluster basado en densidad con respecto a ε_1 conteniendo completamente C_1 y C_2.

En consecuencia, podríamos ampliar el algoritmo DBSCAN de tal manera que varios parámetros de distancia (ε) se procesen al mismo tiempo, es decir, los clusters con respecto a diferentes densidades se construyan de forma simultánea. Para producir un resultado consistente, sin embargo, deberíamos darlo en el orden específico en que se procesan los objetos cuando se expande un cluster. Siempre tenemos que

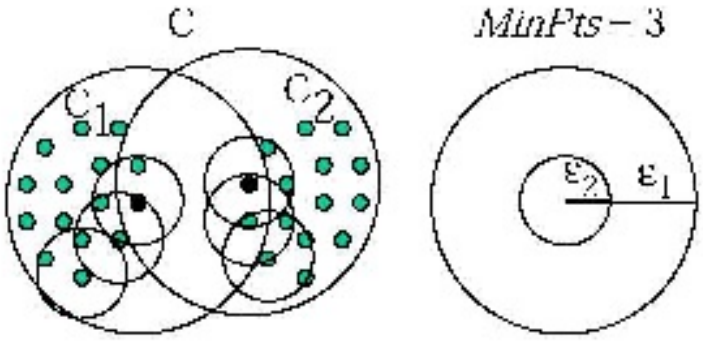


Figura 3.6: Ilustración de clusters basados en densidad anidados.

seleccionar un objeto que es *densamente alcanzable* con respecto al valor más bajo de ε para garantizar que los clusters con los valores más pequeños de ε se obtienen primero.

OPTICS funciona en principio como una extensión de DBSCAN para un número infinito de parámetros ε_i de distancia para los cuales, el parámetro ε, indicado por el usuario, actúa como cota superior de las vecindades a considerar (es decir, $0 \leq \varepsilon_i \leq \varepsilon$). La única diferencia es que nosotros no asignamos un número de miembros a cada cluster. En lugar de ello, almacenamos el orden en que los objetos se procesan, la información que se utilizaría para formar los clusters y se representa el Reachability Plot. En realidad hacen falta sólo dos valores para cada objeto: la *core-distance* y una *reachability-distance*, introducidas en las siguientes definiciones.

Definición 7. (core-distance de un objeto p) Sea p un objeto de una base de datos D, sea ε un valor de distancia, sea N_ε(p) la ε-vecindad de p, sea ν un número natural y sea ν-distance(p) la distancia desde p a sus ν' vecinos. Entonces, $core - distance_{\varepsilon,\nu}$ de p se define como:

$$core - distance_{\varepsilon,\nu}(p) = \begin{cases} \text{UNDEFINED}, & \text{if } |N_\varepsilon(p)| < \nu \\ \nu\text{-distance(p)}, & \text{otherwise} \end{cases}$$

La *core-distance* de un objeto p es simplemente la mínima distancia ε' que da una vecindad de p con ν o más elementos. Es decir, p cumple la condición de *core* con la ε'-vecindad que es una circunferencia de radio menor a ε y $|N'_\varepsilon(p)| \leq |N_\varepsilon(p)|$.

El parámetro ε se convierte para OPTICS en una cota superior del radio utilizado para analizar si una vecindad es zona densa o no. De lo contrario, la *core-distance* es UNDEFINED (no está definida).

Definición 8. (reachability-distance de un objeto p con respecto a o) Sean p y o objetos de una base de datos D, sea N_ε(o), la ε-vecindad de o, y sea ν un número natural. Entonces, la *reachability-distance* de p con respecto a o se define como:

$$reachability - distance_{\varepsilon,\nu}(p,o) = \begin{cases} \text{UNDEFINED}, & \text{if } |N_\varepsilon(o)| < \nu \\ \max(\text{core-distance(o)}, \text{distance(o, p)}), & \text{otherwise} \end{cases}$$

Intuitivamente, la *reachability-distance* de un objeto p con respecto a otro objeto o es la mínima distancia ε' que hace que la ε'-vecindad de o tenga ν objetos, es decir, que garantiza que p es *densamente alcanzable de manera directa* desde o, o lo que es lo mismo, que o es un objeto *core*. En este caso, la *reachability-distance* no puede ser más pequeña que la *core-distance* de o porque para distancias más

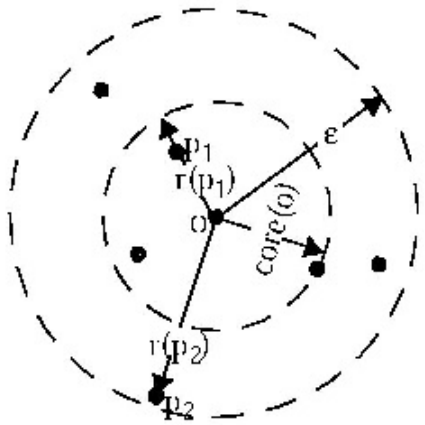


Figura 3.7: Core-distance(o), reachabilities-distances r(p_1,o),r(p_2,o) para $\nu = 4$.

pequeñas no hay ningún objeto *densamente alcanzable de manera directa* desde *o*. De lo contrario, si *o* no es un objeto *core*, incluso con la distancia ε, la *reachability-distance* de *p* con respecto a *o* es indefinida.

La *reachability-distance* de un objeto *p* depende del objeto *core* con respecto al cuál se calcula. La Figura 3.7 ilustra los conceptos *core-distance* y *reachability-distance*.

3.2.1. Pseudocódigo

Nuestro algoritmo OPTICS crea un Reachability Plot conteniendo un cierto ordenamiento de los datos, además almacena la *core-distance* y la *reachability-distance* adecuada para cada objeto. Veremos que esta información es suficiente para extraer todos los clusters con respecto a cualquier distancia ε' que es menor que la distancia de generación ε.

Algorithm 3 OPTICS (SetOfObjects, ε, ν, OrderedFile)

```
OrderedFile.open()
for i = 1 to SetOfObjects.size do
    Object := SetOfObjects.get(i)
    if NOT Object.Processed then
        ExpandClusterOrder(SetOfObjects, Object, ε, ν, OrderedFile)
    end if
end for
OrderedFile.close()
```

En el bucle principal del algoritmo, abrimos un archivo *OrderedFile* para escribir y cerramos al terminar el bucle. Cada objeto de la base de datos *SetOfObjects* se pasa al método *ExpandClusterOrder* si el objeto no se ha procesado todavía.

El pseudo-código para el procedimiento es *ExpandClusterOrder*. El método *ExpandClusterOrder* primero recupera la ε-vecindad del objeto pasado desde el bucle principal, se establece su *reachability-distance* a UNDEFINED y se determina su *core-distance*. A continuación, el objeto se escribe en el fichero *OrderedFile*. La condición IF comprueba si el objeto es *core* dada la distancia ε y si no lo es, el control se devuelve al método principal que selecciona el siguiente objeto no procesado de la base de datos. De lo contrario, si el objeto es un *core* con una distancia $\leq \varepsilon$, iterativamente recogemos la *reachability-distance* de los objetos *densamente alcanzables de manera directamente* dados ε y ν. Los

objetos que son *densamente alcanzables de manera directamente* desde un objeto *core* actual se insertan en la lista *OrderSeeds* para expandir seguir expandiendo el cluster. Los objetos contenidos en *OrderSeeds* están ordenados por su *reachability-distance* al objeto *core* más cercano del que son *densamente alcanzables de manera directa*. En cada paso del bucle WHILE, se selecciona un objeto *currentObject* que tiene la *reachability-distance* más pequeña en la lista *OrderSeeds* por el método *OrderSeeds:next()*. Se obtienen la ε-vecindad de este objeto y su *distancia-core*. Entonces, simplemente se escribe el objeto en el archivo *OrderedFile* con su *core-distance* y su actual *reachability-distance*. Si *currentObject* es *core*, se pueden insertar los candidatos en la lista *OrderSeeds*.

Algorithm 4 ExpandClusterOrder(SetOfObjects, Object, ε, ν, OrderedFile)

```
procedure ExpandClusterOrder(SetOfObjects, Object, ε, ν, OrderedFile)
    neighbors := SetOfObjects.neighbors(Object, ε)
    Object.Processed := TRUE
    Object.reachability-distance := UNDEFINED
    Object.setCoreDistance(neighbors, ε, ν)
    OrderedFile.write(Object)
    if Object.core-distance <> UNDEFINED then
        OrderSeeds.update(neighbors, Object)
        while NOT OrderSeeds.empty() do
            currentObject := OrderSeeds.next()
            neighbors := SetOfObjects.neighbors(currentObject, ε)
            currentObject.Processed := TRUE
            currentObject.setCoreDistance(neighbors, ε, ν)
            OrderedFile.write(currentObject)
            if currentObject.core-distance <> UNDEFINED then
                OrderSeeds.update(neighbors, currentObject)
            end if
        end while
    end if
end procedure
```

La inserción en la lista *OrderSeeds* y el manejo de las *reachability-distances* se hace en el método *OrderSeeds::update(neighbors, CenterObject)*. La *reachability-distance* de cada objeto en el conjunto neighbors es determinada con respecto a *CenterObject*. Los objetos que aún no se encuentran en la cola de prioridad *OrderSeeds* se insertan con su *reachability-distance*. Los objetos que ya están en la cola se mueven hacia la parte superior de la cola si su nueva *reachability-distance* es más pequeña que la que tenían anteriormente.

Debido a su equivalencia estructural con el algoritmo DBSCAN, el tiempo de ejecución de OPTICS es casi el mismo que el tiempo de ejecución para DBSCAN. Se han probado diferentes conjuntos de datos y diferentes ajustes de los parámetros. Simplemente resultó que el tiempo de ejecución de OPTICS era casi constante, 1.6 veces el tiempo de ejecución de DBSCAN (un 60 % más). Esto no es sorprendente ya que el tiempo de ejecución para OPTICS así como para DBSCAN es fuertemente dominado por el tiempo de ejecución de las consultas a la ε-vecindad que deben ser realizadas para cada objeto en la base de datos, es decir, el tiempo de ejecución para ambos algoritmos es $O(n \cdot$tiempo de ejecución de un consulta a ε-vecindad$)$.

Para recuperar la ε-vecindad de un objeto o, se hace una consulta con el centro o y el radio ε. Si no se soporta ningún tipo de estructura indexada, para responder a una consulta de estas regiones, se hace una exploración de toda la base de datos. En este caso, el tiempo de ejecución de OPTICS sería $O(n^2)$.

Algorithm 5 OrderSeeds::update(neighbors, CenterObject)

```
procedure OrderSeeds::update(neighbors, CenterObject)
    c-dist := CenterObject.core-distance
    for all Object from neighbors do
        if NOT Object.Processed then
            new-r-dist := max(c-dist, CenterObject.dist(Object))
            if Object.reachability-distance = UNDEFINED then
                Object.reachability-distance := new-r-dist
                insert(Object, new-r-dist)
            else//Object already in OrderSeeds
                if new-r-dist < Object.reachability-distance then
                    Object.reachability-distance := new-r-dist
                    decrease(Object, new-r-dist)
                end if
            end if
        end if
    end for
end procedure
```

Si se utiliza un árbol basado en índice espacial, el tiempo de ejecución se reduce a $O(n \cdot \log n)$ ya que las consultas a *regionQuery* se hacen de forma eficiente por los métodos de acceso espaciales, como la árboles R* [Beckmann 90]. La altura de una estructura de este tipo basada en árboles es $O(\log n)$ para una base de datos de n objetos en el peor de los casos y, por lo menos en espacios de dimensiones bajas, una consulta con una región "pequeña" tiene que atravesar solo un número limitado de caminos. Por otra parte, si tenemos un acceso directo a la ε-vecindad, por ejemplo, si se organizan los objetos en una rejilla, el tiempo de ejecución se reduce aún más a $O(n)$, ya que en una cuadrícula la complejidad de una sola consulta a la vecindad es $O(1)$.

Después de haber generado el Reachability Plot de una base de datos con respecto a la ε y ν, podemos extraer los clusters de esta ordenación con respecto a ν y una distancia $\varepsilon' \leq \varepsilon$ "escaneando" el reachability y asignando objetos a clusters en función de la *reachability-distance* y la *core-distance*.

Algorithm 6 ExtractDBSCAN-Clustering(ClusterOrderedObjs, ε', ν)

```
procedure ExtractDBSCAN-Clustering(ClusterOrderedObjs, ε', ν)
    // Precondition: ε' ≤ generating dist ε for ClusterOrderedObjs
    ClusterId := NOISE
    for i = 1 to ClusterOrderedObjs.size do
        Object := ClusterOrderedObjs.get(i)
        if Object.reachability-distance > ε' then
            // UNDEFINED > ε
            if Object.core-distance ≤ ε' then
                ClusterId := nextId(ClusterId)
                Object.clusterId := ClusterId
            else
                Object.clusterId := NOISE
            end if
        else// Object.reachability-distance ≤ ε'
            Object.clusterId := ClusterId
        end if
    end for
end procedure
```

En primer lugar, se comprueba si la *reachability-distance* del objeto actual es mayor que la distancia ε'. En este caso, el objeto no es *densamente alcanzable* con respecto a ε' y ν desde cualquiera de los

objetos que se encuentran antes que el objeto actual en el Reachability Plot. Esto es obvio, porque si *Object* es *densamente alcanzable* con respecto a ε' y ν desde un objeto que le precede en la ordenación, le habría sido asignada una *reachability-distance* de como máximo ε'. Por lo tanto, si la *reachability-distance* es mayor que ε', nos fijamos en la *core-distance* de *Objetos* e iniciamos un nuevo cluster si el objeto es *core* con respecto a ε' y ν; de lo contrario, el objeto se asigna a NOISE (hay que tener en cuenta que la *reachability-distance* del primer objeto en el Reachability Plot siempre es UNDEFINED y que asumimos que UNDEFINED es mayor que cualquier distancia definida). Si la *reachability-distance* del objeto actual es menor que ε', podemos simplemente asignar este objeto al cluster actual porque entonces es *densamente alcanzable* con respecto a ε' y ν de un objeto *core* anterior del Reachability Plot.

El clustering creado a partir de un conjunto de datos ordenado establecido por ExtractDBSCAN-Clustering es casi indistingible de un clustering creado por DBSCAN. Sólo algunos objetos fronterizos se pueden perder cuando se extraen con ExtractDBSCAN-Clustering si fueron procesados por el algoritmo OPTICS antes de que un objeto *core* del correspondiente cluster hubiera sido encontrado.

Sin embargo, la fracción de estos objetos fronterizos es tan pequeña que se puede omitir un postprocesado (es decir, volver a asignar los objetos a un cluster) sin mucha pérdida de información.

Extraer diferentes clusters basados en la densidad del Reachability Plot de un conjunto de datos no es el uso previsto del algoritmo OPTICS. Que una extracción es posible, sólo demuestra que el Reachability Plot de un conjunto de datos contiene en realidad la información acerca de la estructura de la agrupación intrínseca de ese conjunto de datos (hasta la distancia de generación ε). Esta información se puede analizar mucho más eficazmente mediante el uso de otras técnicas que se presentan en la siguiente sección.

3.2.2. Identificación de la estructura del cluster

El algoritmo OPTICS genera el Reachability Plot como una ordenación de los puntos, los valores de *reachability-distance* y los valores *core*. Sin embargo, para los análisis y técnicas iterativas siguientes, sólo son necesarios el ordenamiento y los valores de reachability. Para simplificar la notación, se especifica formalmente:

Definición 9. (resultados del algoritmo OPTICS) Sea DB una base de datos que contiene n puntos. El algoritmo OPTICS genera un ordenamiento de los puntos $o : \{1..n\} \rightarrow DB$ y los valores de reachability correspondientes $r : \{1..n\} \rightarrow R_{\geq o}$.

Las técnicas visuales presentadas a continuación se dividen en dos categorías principales. En primer lugar, los métodos para obtener una visión general de los datos. Estos son útiles para obtener una comprensión a alto nivel de la manera en que se estructuran los datos. Es importante ver la mayoría o incluso la totalidad de los datos a la vez, haciendo visualizaciones de pixels orientadas al método de elección. En segundo lugar, una vez que la estructura general se entiende el usuario está interesado en centrar la vista a los subconjuntos más interesantes. En la vista detallada correspondiente, se analizan clusters individuales (pequeños o grandes) y se examinan sus relaciones. Aquí es importante mostrar la máxima cantidad de información para que se pueda entender fácilmente. Por lo tanto, presentamos diferentes técnicas para estas dos tareas.

Debido a que la técnica detallada es una representación gráfica directa del clustering ordenado, lo

presentamos primero y luego continuamos con la técnica a alto nivel.

3.2.3. Reachability Plots y parámetros

La estructura de densidades de la base de datos se puede representar con el Reachability Plot y entender gráficamente. En principio, se puede ver la estructura del cluster de un conjunto de datos si los valores de *reachability-distance* r se representan para cada objeto en el cluster de ordenación o.

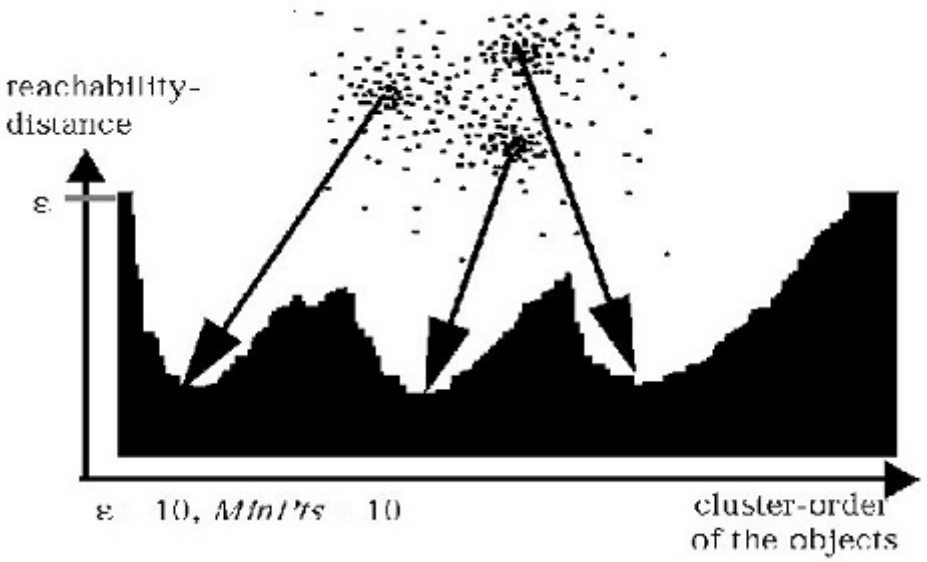


Figura 3.8: Ilustración del cluster de ordenación.

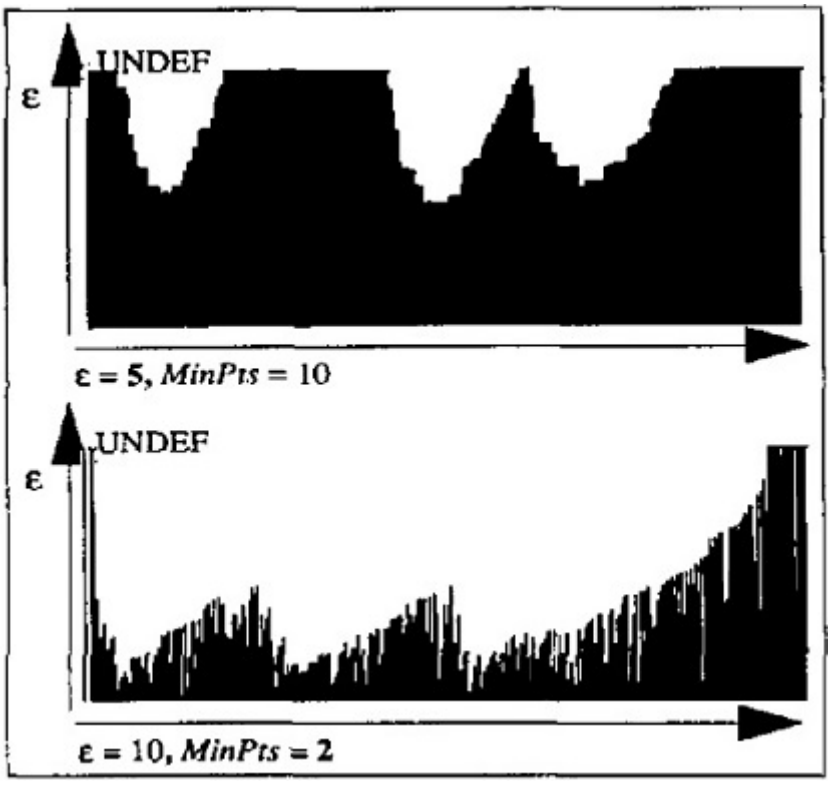


Figura 3.9: Efectos ajustar el parámetro cluster de ordenación.

Una ventaja adicional en comparación con otros métodos de clustering es que el *reachability-plot* es

bastante insensible a los parámetros de entrada del método (ε y ν). En términos generales, los valores tienen que ser lo suficientemente "grandes" para producir un buen resultado. Los valores concretos no son cruciales porque hay un amplio rango de valores posibles para el que siempre podemos ver la estructura del clustering de un conjunto de datos cuando se mira en el correspondiente *reachability-plot*. En realidad el Reachability Plot es robusto a pequeños cambios en los parámetros del algoritmo (ε y ν) que son los mimos que DBSCAN. La Figura 3.9 muestra los efectos de diferentes configuraciones del parámetro en el *reachability-plot* para el mismo conjunto de datos usado en al Figura 3.8. En el primer gráfico se utiliza una distancia ε más pequeña, para el segundo gráfico establecemos ν al valor más pequeño posible. A pesar de que estas gráficas tienen un aspecto diferente al de la representada en la Figura 3.8, la estructura general de clustering de los datos también se reconoce.

La distancia generadora ε influye en el número de niveles de clustering. Cuanto más pequeño elegimos el valor ε, más objetos pueden tener una *reachability-distance* INDEFINIDA. Por lo tanto, es posible que no se vean clusters de menor densidad, es decir, los clusters donde los objetos *core* son objetos *core* sólo para distancias mayores que ε.

El valor óptimo ε es el valor más pequeño de modo que el clustering de la base de datos con respecto a ε y ν consta de un sólo cluster que contiene casi todos los puntos de la base de datos. Entonces, la información de todos los niveles de clustering está contenida en el *reachability-plot*.

3.2.4. Extracción de los clusters a partir del Reachability Plot

A partir del Reachability Plot se pueden determinar visualmente los clusters que se van a conseguir. El Reachability Plot es un gráfico 2D, con el orden de los puntos procesados por OPTICS en el eje X y las *reachability-distances* de cada uno en el eje Y. Ya que los puntos que pertenecen a un cluster tienen una *reachability-distance* baja a su vecino más cercano, los clusters aparecen como valles en el Reachability Plot. Cuanto más profundo es el valle, más denso es el cluster.

La Figura 3.10 ilustra este concepto. En su parte superior izquierda, se muestra un ejemplo de un conjunto de datos. La parte superior derecha visualiza el árbol de expansión producido por OPTICS, y la parte inferior muestra el Reachability Plot calculado. Los colores en este gráfico son las etiquetas no calculadas por el algoritmo; pero es fácilmente visible cómo los valles del gráfico corresponden a los clusters en el conjunto de datos anterior. Los puntos amarillos en esta imagen se consideran ruido. Ellos no suelen ser asignados a clusters.

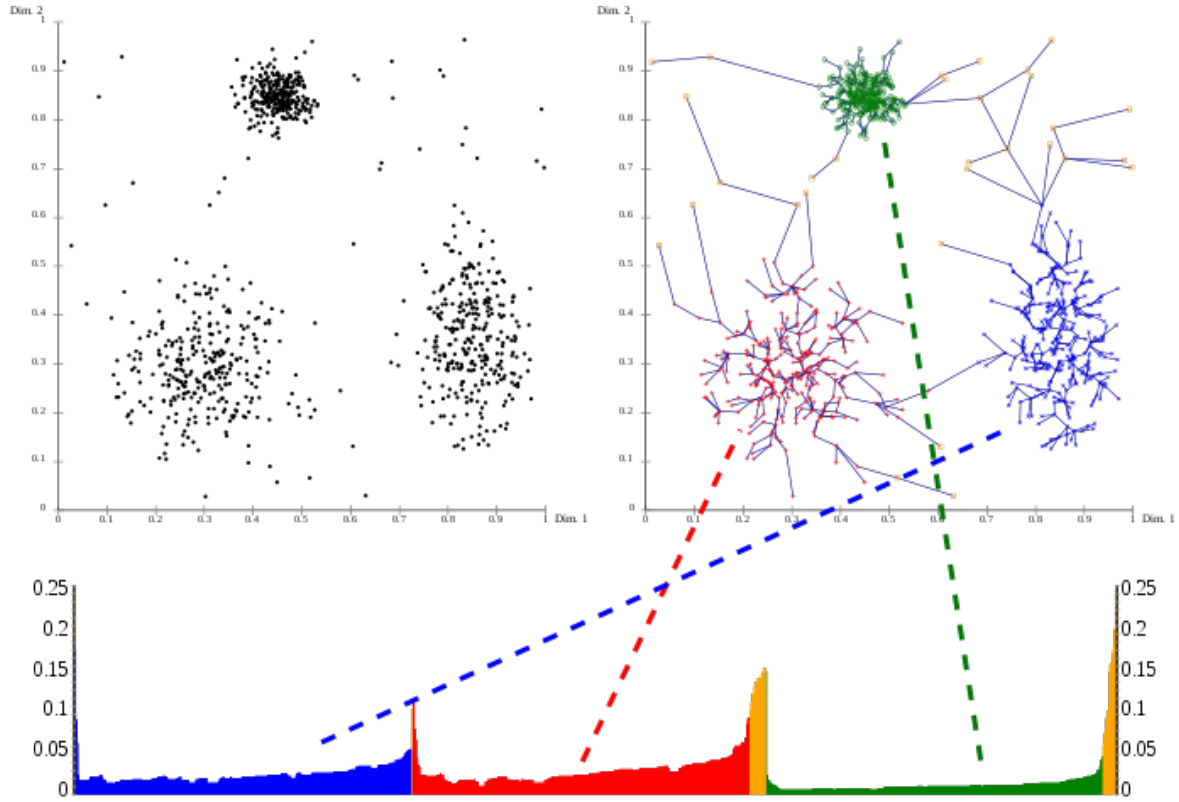


Figura 3.10: Extrayendo los clusters.

La extracción de clusters de este gráfico se puede realizar manualmente mediante la selección de un rango en el eje X después de inspeccionarlo de forma visual, mediante la selección de un umbral en el eje Y (el resultado será entonces similar al clustering resultante del DBSCAN con los mismos parámetros ε y ν; aquí un valor de 0.1 puede dar buenos resultados).

3.2.5. Ventajas del método OPTICS

La principal ventaja, además de las ya mencionadas en el apartado de DBSCAN, con otros algoritmos de clustering propuestos, es que no nos limitamos a un parámetro de configuración global. El Reachability Plot contiene información que es equivalente a los clusters correspondientes a una amplia gama de ajustes de parámetros y por lo tanto es una base versátil para el análisis automático e interactivo del cluster.

Capítulo 4

Diseño e implementación

Está sección habla en profundidad del diseño de estos algoritmos de clustering basados en densidad para **KLASS**.

Como ya se ha definido en la sección 1.1, en este proyecto se han implementado nuevas funcionalidades dentro de una aplicación existente, con el propósito de ampliarla.

4.1. Estructura de KLASS

Con el fin de entender los detalles de la siguiente sección, es necesario introducir la estructura de **KLASS** des del punto de vista de la implementación. **KLASS** es una aplicación Java que se ha implementado siguiendo una estructura de capas para separar la interfaz gráfica de los principales métodos y los datos de los objetos. **KLASS** se compone de las siguientes capas:

- jklass.iu: Esta capa contiene las clases relacionadas con la interfaz gráfica de usuario. **KLASS** tiene una interfaz de ventanas, y cada una de estas ventanas se implementa en una clase. Estas clases sólo puede llamar a los métodos de la parte del kernel para ejecutar las acciones.
- jklass.nucli: Esta capa es parte del núcleo de **KLASS**. Contiene todos los métodos que realmente ejecutan las acciones requeridas por el usuario. Las clases importantes para este proyecto son:
  - GestorMatriu: Representa una matriz de datos y contiene todos los métodos necesarios para su manejo. También contiene información que puede querer ser recuperada.
  - GestorKlass: **KLASS** es una aplicación donde puede existir más de una matriz de datos al mismo tiempo. Esta clase permite la gestión de las diferentes matrices llamando a los métodos de la instancia correspondiente, y que permite el acceso a toda la funcionalidad proporcionada por el kernel de **KLASS**.
  - GestorClassificacio: Es el módulo más importante de nuestro proyecto. Desde este módulo se llaman a las funciones necesarias para realizar la clasificación usando diferentes métricas y métodos. Entre ellos están los métodos implementados en este proyecto: DBSCAN y OPTICS.
- jklass.util: Este paquete contiene clases para la gestión de las opciones del sistema, parámetros de configuración y llamadas al sistema operativo.

Para la implementación de estas nuevas funcionalidades, se han añadido nuevas clases a la interfaz de usuario y nuevas clases al paquete *nucli*.

- jklass.iu:

  - DlgOpcDBSCAN: Esta clase simplemente abre un diálogo con las opciones disponibles para ejecutar el algoritmo DBSCAN, y una vez el usuario escoge las opciones deseadas y clica en *d'acord*, estos valores se guardan temporalmente hasta que se hace la clasificación o se realiza otra acción.

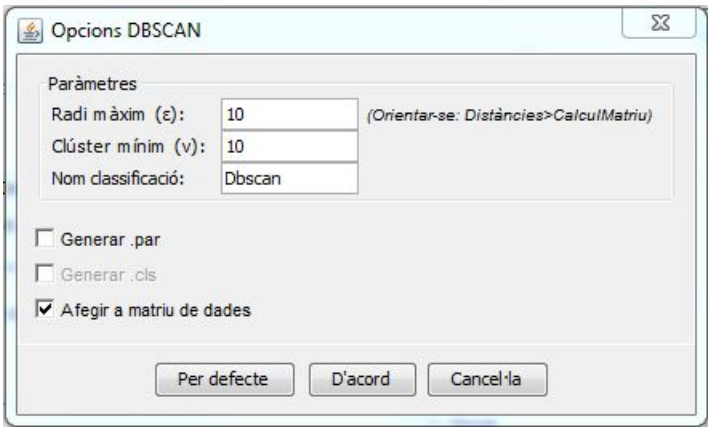


Figura 4.1: Captura diálogo opciones DBSCAN.

  - DlgOpcOPTICS: Abre un diálogo con las opciones disponibles para ejecutar el algoritmo OPTICS, de nuevo, una vez el usuario escoge las opciones deseadas y clica en *d'acord*, estos valores se guardan temporalmente hasta que se hace la clasificación o se realiza otra acción.

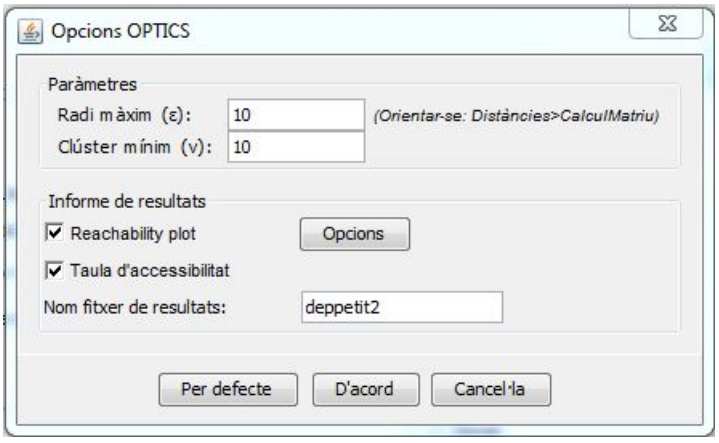


Figura 4.2: Captura diálogo opciones OPTICS.

  - DlgOpcReachabilityPlot: Abre un diálogo con las opciones disponibles para visualizar el Reachability Plot que genera OPTICS.

  - DlgCanviReach: Abre un diálogo con los Reachabilities guardados actualmente en el sistema para dar la posibilidad al usuario de elegir cuál será el Reachability seleccionado en el sistema.

  - Panel3D: Esta clase da la posibilidad de seleccionar las variables numéricas que se quieren representar en los ejes X, Y y Z y la variable categórica que se quiere representar para cada uno de los datos de la matriz. Una vez escogidas estas variables y clicando en *d'acord* se abre un *frame* con la visualización del 3D.

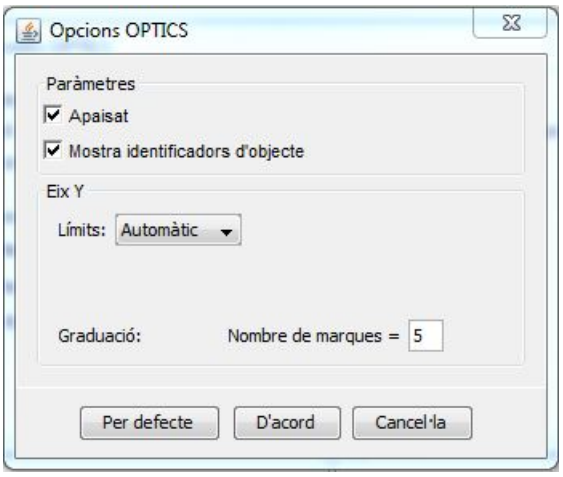


Figura 4.3: Captura diálogo opciones Reachability Plot.

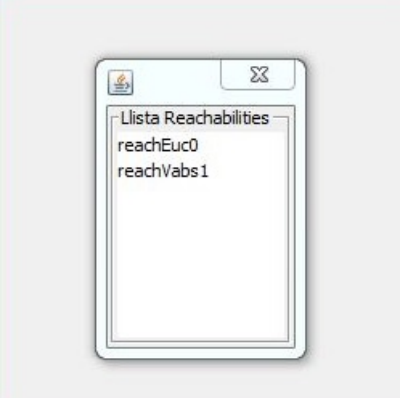


Figura 4.4: Captura diálogo lista Reachabilities.

- PanelTallaReachability: Este panel solo esta disponible una vez se ha generado por lo menos un Reachability Plot, es decir, hay en el sistema un gráfico generado con OPTICS el cuál podemos cortar para poder realizar una clasificación.

  Elegimos el nivel del corte (entre los valores indicados), un nombre para la clasificación, un prefijo para la clase, que ficheros se quieren generar (*.cls* o *.par*), etc.

  Una vez se escogen las opciones para cortar el Reachability Plot, se puede añadir la clasificación a la matriz de datos como en el caso DBSCAN.

- Window3D: Esta clase extiende de un frame para poder visualizar el gráfico 3D que ya hemos comentado en la clase Panel3D. Este frame permite además de visualizar las variables sobre los ejes de coordenadas, también permite hacer zoom de la vista y rotar los ejes para dar la posibilidad al usuario de tener diferentes perspectivas del gráfico.

  Los datos están representados con puntos o cruces de varios colores para diferenciar unos de otros. Datos con una variable categórica distinta serán representados con distintos colores.

- jklass.nucli:

  - Calculs3D: Esta clase es la que se encarga de a partir de una matriz de datos, hacer todos los cálculos necesarios para posteriormente poder representarlos sobre los ejes de coordenadas. Guardamos toda esta información dentro de esta clase y la consultamos cuando es necesaria.

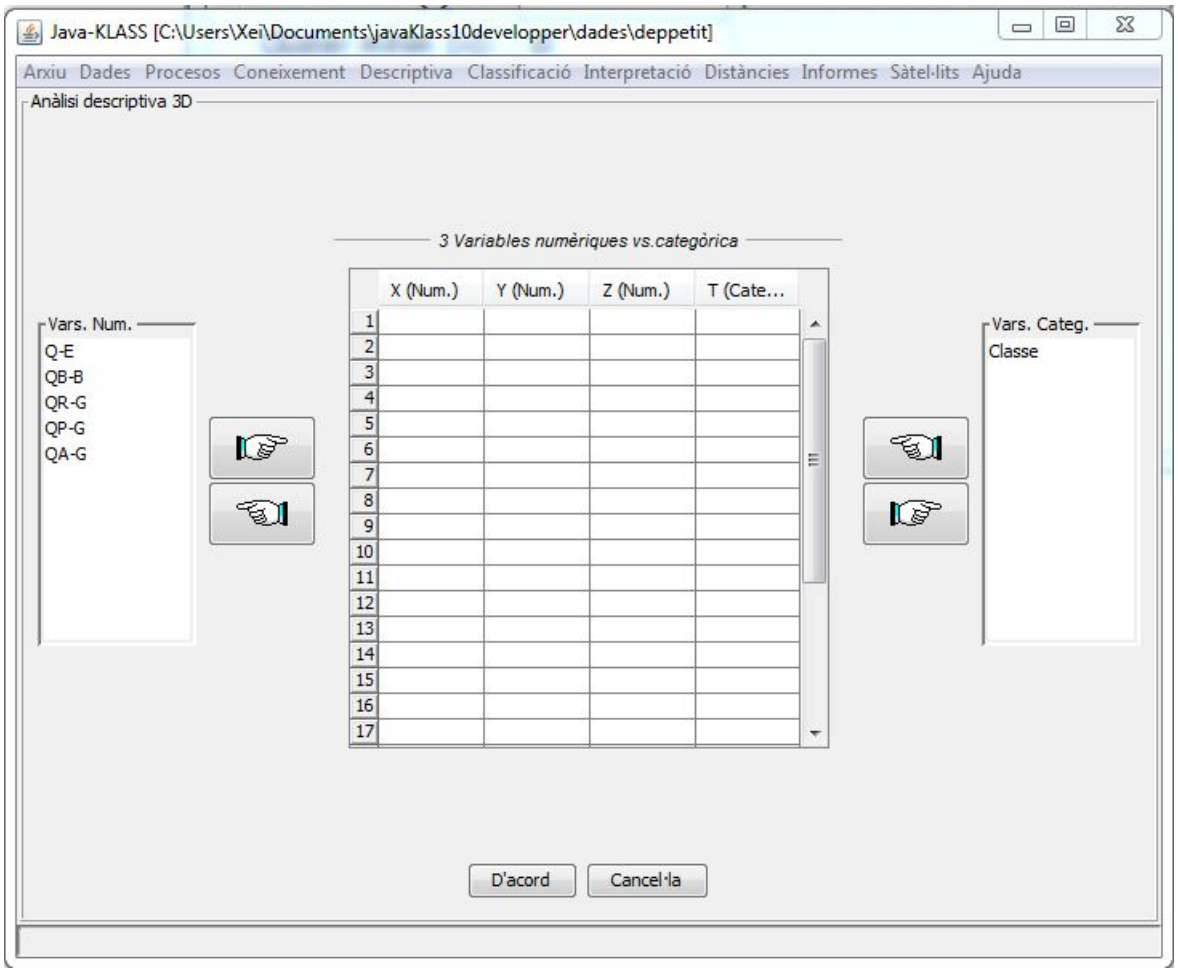


Figura 4.5: Captura Panel3D.

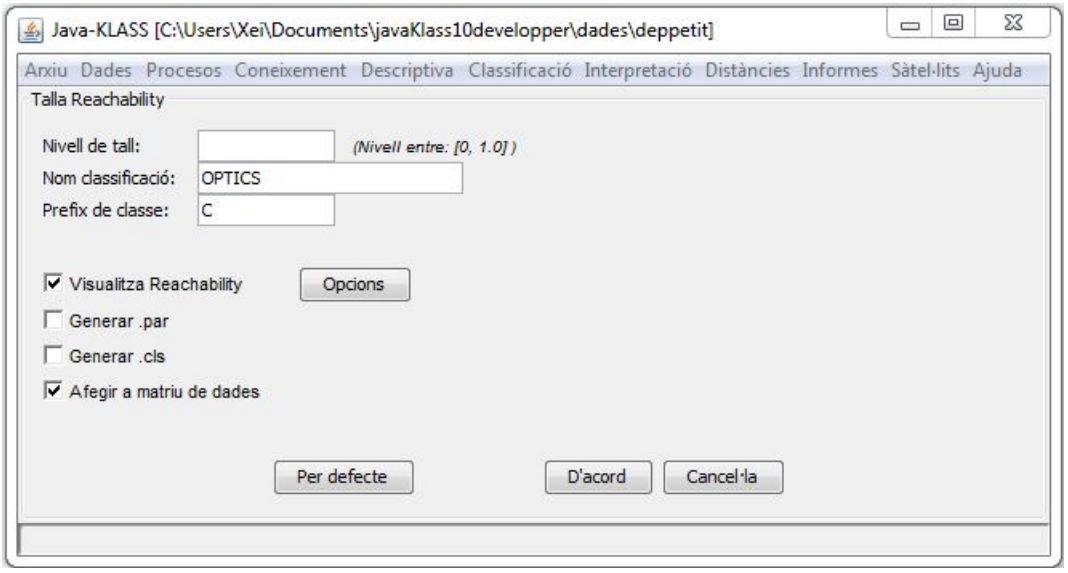


Figura 4.6: Captura PanelTallaReachability.

- DensityBasedClustering: Esta es la clase más importante del proyecto, ya que es la que implementa los dos algoritmos de clustering de los que hemos hablado. Es una subclase de *GestorClasificacion*. Se ha decidido que fuera una clase propia, y no añadir los métodos y estructuras directamente a la clase padre, para así tener el concepto de clustering basado en

densidad más separado y fuera más fácil su manipulación.

- ReachabilityPlot: Se encarga de dibujar el Reachability Plot en formato LaTeX, pasándole como parámetros la *reachability-distance*, los identificadores de los datos y todas las variables a partir de las cuales se han calculado las *reachability-distances* para poder visualizar toda esta información.
- Reachability: Esta clase ha sido creada expresamente para guardar toda esa información necesaria para dibujar el Reachability Plot y así poder mantener en el sistema los que se han creado previamente con esa matriz de datos. Posteriormente se puede escoger el Reachability Plot deseado desde el menú.

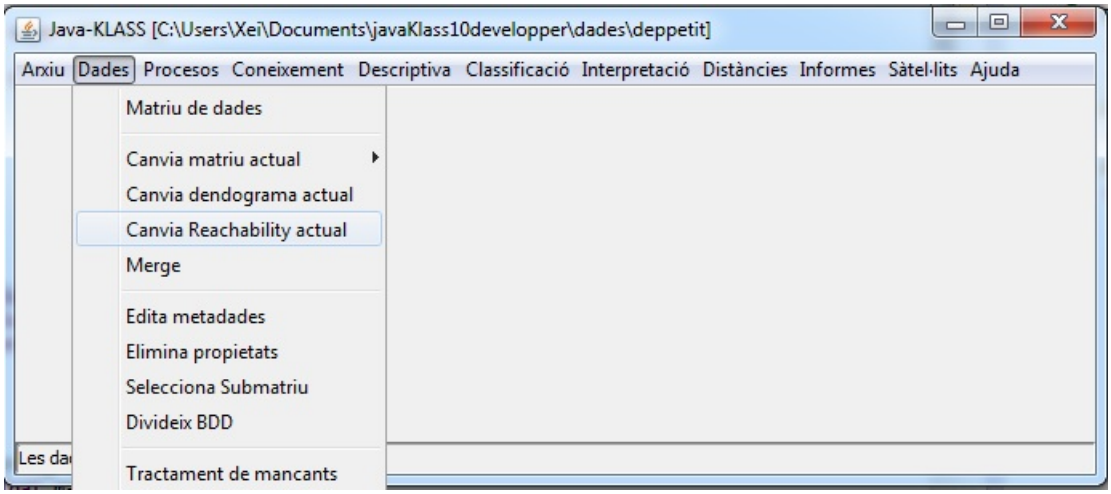


Figura 4.7: Captura cambia Reachability.

- ReachabilityTable: Se encarga de dibujar la tabla con la información de las *reachability-distances* en formato LaTeX, pasándole como parámetros las *reachability-distances* y los identificadores de los datos.

Además de añadir nuevas clases, se han añadido nuevos métodos a las clases ya existentes. Se mencionan los más relevantes:

- En GestorClasificacion
  - densityBasedClusteringResultatOPTICS: Encargada de generar la instancia de la subclase DensityBasedClustering y llamar al método que ejecuta el algoritmo OPTICS.
  - densityBasedClusteringResultatDBSCAN: Encargada de generar la instancia de la subclase DensityBasedClustering y llamar al método que ejecuta el algoritmo DBSCAN.
  - tallaReachability: Se encarga de llamar al método encargado de extraer los clusters una vez hemos ejecutado el algoritmo de OPTICS.
- En GeneradorTex
  - escriureParametres: Simplemente escribe los parámetros utilizados para ejecutar OPTICS en el fichero TeX.

- generarLtxReachability: Llama a la clase *ReachabilityPlot* previamente mencionada para generar el fichero LaTeX.

- En GestorMatriu

  - añadirClasificacion: Añade la clasificación previamente calculada a la matriz de datos.
  - ferReachability: Llama a la función que genera el Reachability Plot que actualmente está seleccionado en la matriz de datos.
  - obtenirCalculs3D: Obtiene los cálculos previos para realizar la visualización 3D.
  - agregarReachability: Añade un Reachability a la lista de reachabilities del gestorMatriu.
  - modificaReachActual: Modifica el Reachability que actualmente está seleccionado en el sistema.
  - obtenirLlistaNomReach: Devuelve la lista con los nombres de todos los Reachabilities que hay en el sistema.
  - generaParticio: Genera el fichero *.par* a partir de las clases después de hacer DBSCAN o cortar el Reachability.
  - densityBasedClusteringMatriuOPTICS: Crea una instancia del *GestorClasificacion* y llama al método encargado de hacer OPTICS.
  - densityBasedClusteringMatriuOPTICS: Crea una instancia del *GestorClasificacion* y llama al método encargado de hacer DBSCAN.
  - setClusters: Guarda el resultado de los clusters obtenidos con su identificador y el número de clases al hacer DBSCAN.
  - tallaReachability: Llama al método *tallaReachability* de *GestorClasificacion*.

- En GestorKlass

  - generaParticio: Llamará al método de *GestorMatriu* que generará el fichero *.par* con el nombre indicado.
  - obtenirIdReachability: Obtiene el identificador para el Reachability Plot a partir de la métrica y el método usado.
  - modificaReachActual: modifica el Reachability actual del sistema, llamando al método de la clase *GestorMatriu*.
  - densityBasedClusteringOPTICS: Llamará al método OPTICS con los datos de la matriz actual.
  - visualitzaReachabilityPlot: Llamará al método, que hay dentro de *GestorMatriu*, que generará el fichero LaTeX para visualizar el Reachability Plot deseado.
  - densityBasedClusteringDBSCAN: Llamará al método DBSCAN con los datos de la matriz actual.

A continuación se muestran los diagramas de *Interfaz de Usuario* y de *Núcleo* que intervienen en este proyecto. Las cajas que no contienen métodos son las que ya estaban en el sistema. La clase *GestorClasificacion* sin embargo, contiene métodos aunque no se ha creado para este proyecto. Esto es porqué es una de las clases más importantes que intervienen en él y se han señalado los métodos que se han creado en este proyecto.

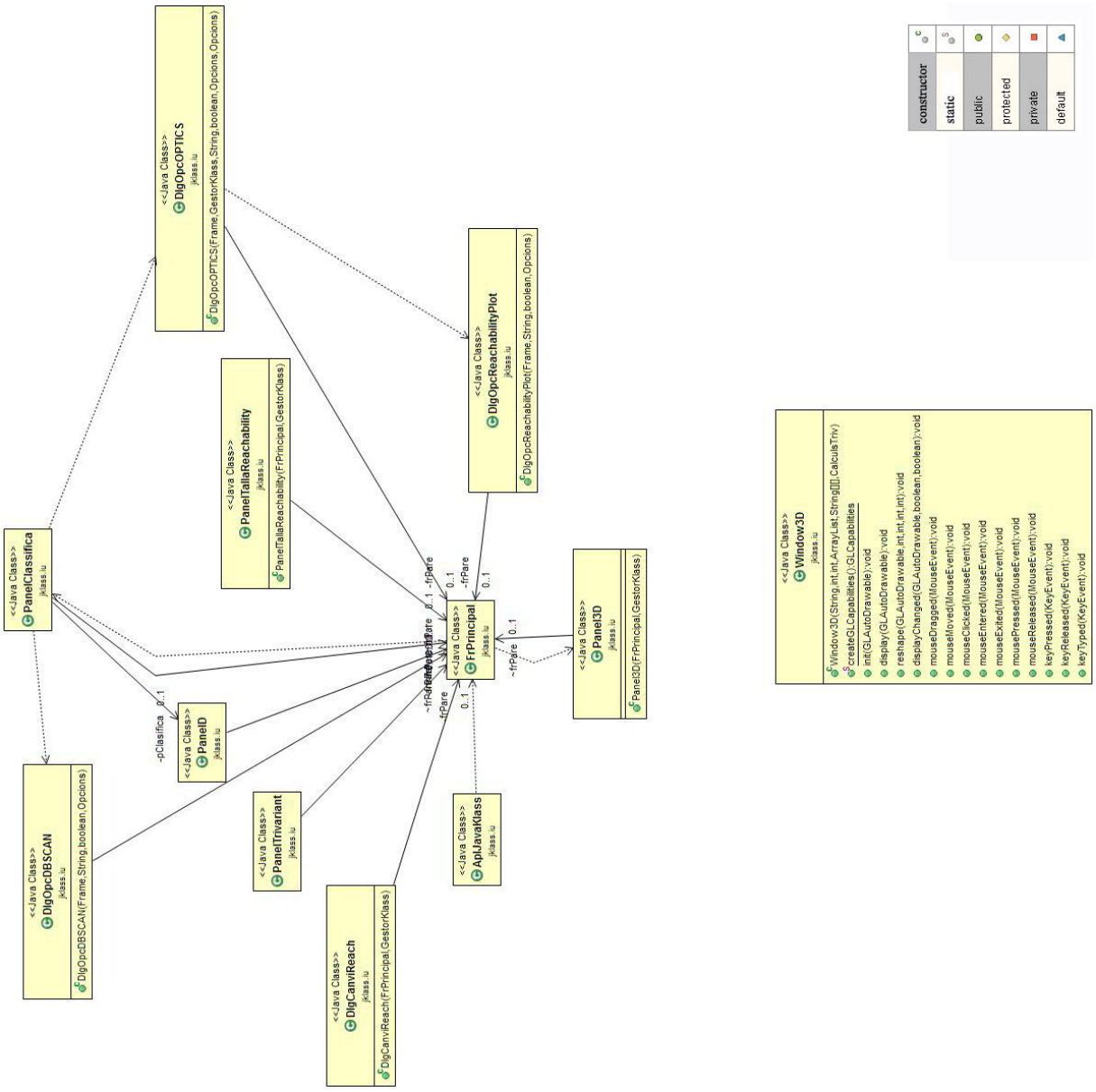


Figura 4.8: Diagrama de clases Interfaz de Usuario.

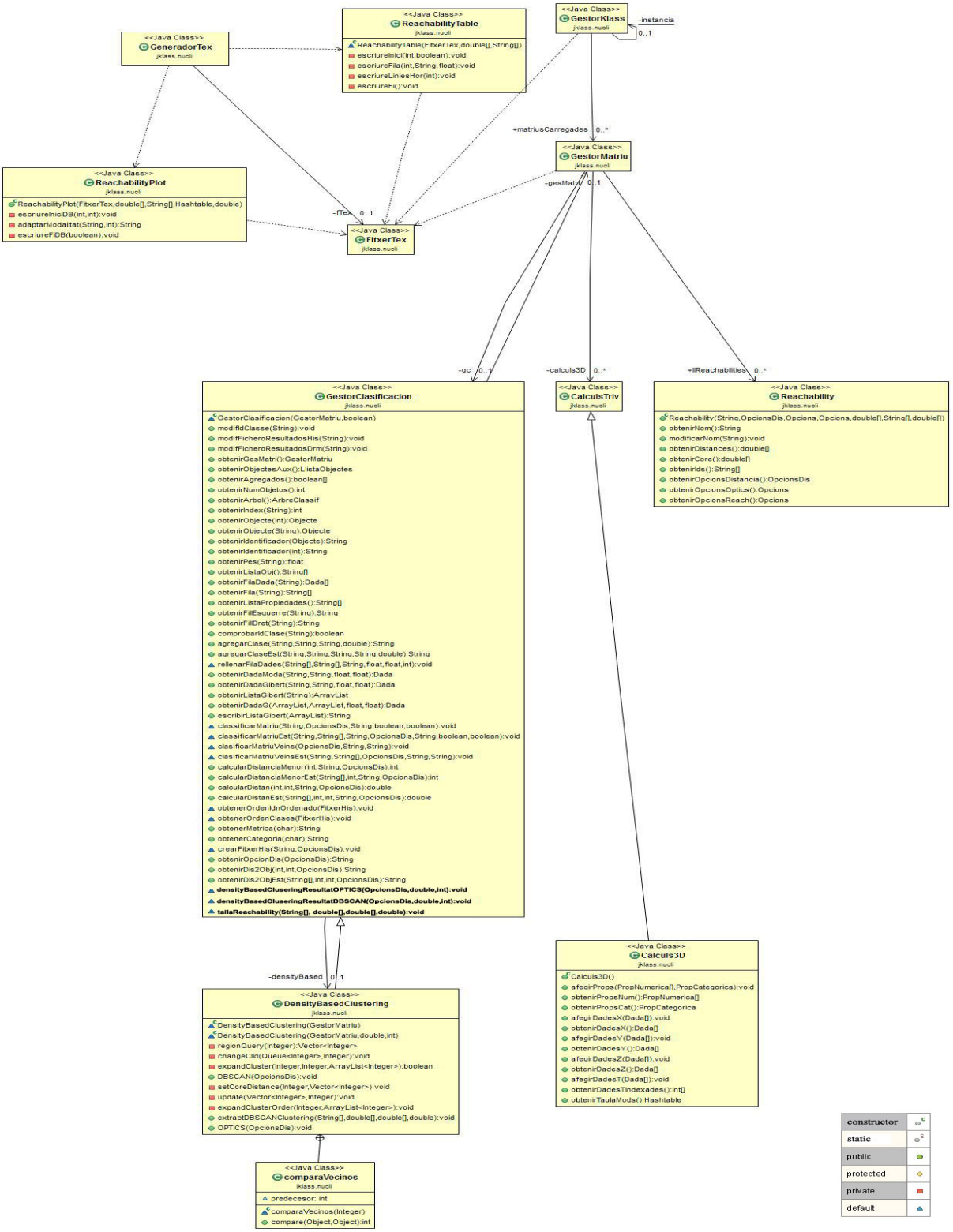


Figura 4.9: Diagrama de clases Núcleo.

Ahora veremos otro tipo de diagrama para dar una idea de algunas estructuras que se han creado para el manejo de información.

Es el caso de la estructura Reachability: Esta estructura ha sido creada para poder guardar toda la información necesaria para dibujar un Reachability Plot y así poder guardar varios Reachabilities asociados a una matriz de datos e irlos recuperando.

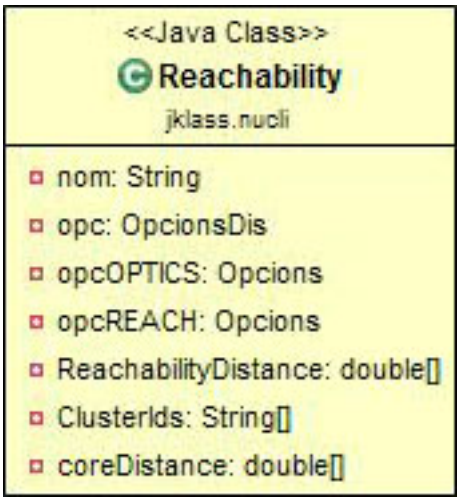


Figura 4.10: Estructura Reachability.

- nom: Es el nombre que le hemos asignado al Reacability Plot para poder distinguir varios en la matriz de datos.
- opc: Son las opciones de la distancia métrica que se han seleccionado para crearlo.
- opcOPTICS: Son las opciones escogías por el usuario para ejecutar OPTICS (ε, ν, etc).
- opcREACH: Opciones seleccionadas para la representación del Reachability (apaisado, mostrar identificadores, etc).
- ReachabilityDistance: Guarda la reachability-distance de cada uno de los objetos.
- ClusterIds: Identificadores de cada uno de los objetos.
- coreDistance: Guarda la distancia core de los objetos a representar.

También hay otra estructura a comentar, Calculs3D. Esta estructura además de hacer los cálculos para poder representar los datos en la perspectiva 3D, también guarda toda esta información para posteriormente poder recuperarla a la hora de dibujar el 3D.

- propsN: Guarda las propiedades numéricas usando una estructura ya existente en el software para guardar datos numéricos.
- propC: Guarda datos relacionados con la propiedad categórica usando otra estructura del sistema.
- indexModsC: Contiene las propiedades categóricas indexadas para posteriormente poder identificarlas con un color o forma distintos.
- dadesX: Contiene la situación de cada uno de los objetos sobre el eje de las X.

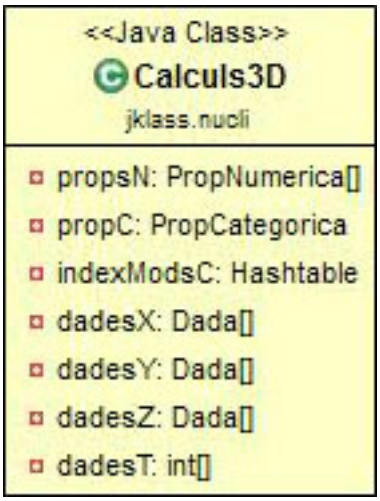


Figura 4.11: Estructura Calculs3D.

- dadesY: Contiene la situación de cada uno de los objetos sobre el eje de las Y.
- dadesZ: Contiene la situación de cada uno de los objetos sobre el eje de las Z.
- dadesT: Guarda a qué propiedad categórica pertenecen cada uno de los datos.

4.2. Complejidad de los algoritmos

4.2.1. DBSCAN

DBSCAN visita cada punto de la base de datos, y puede que varias veces. La complejidad del tiempo se rige sobre todo por las llamadas a *regionQuery* para recuperar la región de vecinos. Al inicio de cada ejecución de DBSCAN hacemos una llamada para obtener la matriz de distancias en un tiempo $O(n^2)$ y ya no tenemos que volver a consultarla. El algoritmo ejecuta una consulta a *regionQuery* por cada punto no visitado, y como ya disponemos de la matriz de distancias, solo tendremos que ir consultando esa matriz para obtener los vecinos del punto que estamos visitando. El tiempo para obtener los vecinos de un punto (*regionQuery*) es entonces como mucho $O(n)$. Y el coste del algoritmo sería el tiempo de visitar cada punto de la base de datos y por cada uno consultar sus vecinos, es decir, $O(n^2)$.

4.2.2. OPTICS

Como en el caso anterior, OPTICS visita cada punto de la base de datos y hace una consulta a *regionQuery* ($O(n)$). Además de consultar los vecinos, también actualizamos la *reachability-distance* de cada vecino si no ha sido visitado y el punto que se esta visitando tiene una distancia *core* definida, por tanto, el tiempo de ejecución global es el mismo que con DBSCAN aunque en media es algo superior.

Como se ha dicho en el apartado 3.2, los autores del paper de OPTICS, Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, Jörg Sander [Ankerst 99] hablan de un factor de desaceleración constante real de 1.6 en comparación con DBSCAN. El valor de ε podría influir en el coste del algoritmo, un valor demasiado grande puede aumentar el coste de la consulta a *regionQuery*.

Si escogemos una $\varepsilon > max - dist(p, q)$ posiblemente obtendremos una complejidad cuadrática, ya que cada consulta a *regionQuery* devolverá todo el conjunto de datos. Por tanto la variable ε debe elegirse adecuadamente para cada conjunto de datos.

4.2.3. Tabla comparativa

En la Tabla 4.1 vemos una comparativa de los tiempos de los algoritmos DBSCAN y OPTICS con bases de datos de distintos tamaños. Como ya hemos hablado anteriormente en media OPTICS se comporta peor que DBSCAN en cuanto a tiempo de ejecución, pero en general los tiempos son bastante parecidos.

Método \ Datos	150	1.000	10.000
DBSCAN	21.1 (milisegundos)	9.146 (segundos)	13.23 (minutos)
OPTICS	24.2 (")	9.102 (")	13.45 (")

Tabla 4.1: Comparación del tiempo de ejecución de DBSCAN y OPTICS.

Capítulo 5

Experimentación y casos de estudio

En este capítulo veremos algunos casos de estudio para ver cómo se comportan los algoritmos implementados y compararemos los resultados que dan cada uno.

5.1. Caso 1: Iris

El primer caso que se ha estudiado ha sido sobre el conjunto de datos *Iris*. Los datos de flores Iris o conjunto de datos Iris de Fisher es un conjunto de datos multivariante introducido por Sir Ronald Fisher 1936 [Fisher 36] como un ejemplo de análisis discriminante. A veces a los datos se les llama conjunto de datos Iris de Anderson porque Edgar Anderson [Anderson 35] [Anderson 36] recogió los datos para cuantificar la variación morfológica de las flores Iris de tres especies relacionadas. Dos de las tres especies fueron recolectadas en la Península Gaspé "todas de la misma pastura, y recogidas en el mismo día y se miden al mismo tiempo por la misma persona con el mismo instrumento".

El conjunto de datos consta de 50 flores de cada una de las tres especies de Iris (*Iris setosa*, *Iris virginica* e *Iris versicolor*). A partir de cada muestra, se miden cuatro características.

- longitud pétalo (centímetros).
- anchura pétalo (")
- longitud sépalo (")
- anchura sépalo (")

Sobre la base de la combinación de estas cuatro características, Fisher desarrolló un modelo lineal discriminante para distinguir las especies entre sí.

Basado en el modelo discriminante lineal de Fisher, este conjunto de datos (ver apéndice A) se convirtió en un caso de prueba típica para muchas técnicas de clasificación de *machine learning*, tales cómo *support vector machines*.

El uso de este conjunto en el análisis de clusters de datos sin embargo es poco común, ya que el conjunto de datos sólo contiene dos grupos de los 3 diferenciables de forma bastante obvia. Uno de los grupos contiene *Iris setosa*, mientras que el otro grupo contiene *Iris virginica* e *Iris versicolor* y no es linealmente separable. Por ello, sin la información sobre especies que Fisher proporciona es muy difícil que un algoritmo de clustering clásico reconozca bien las especies. Esto hace que los datos den un buen ejemplo para explicar la diferencia entre las técnicas supervisadas y no supervisadas en la minería de

(a) Iris setosa (b) Iris virginica (c) Iris versicolor

Figura 5.1: Especies de Iris

datos: el modelo discriminante lineal de Fisher sólo puede obtenerse cuando se utiliza la información sobre la especie. En general, las etiquetas de clase no van a coincidir necesariamente con los clusters [Farber 2010].

De cara a comparar los resultados que vayamos obteniendo tenemos una clara ventaja con este conjunto de datos, ya que tenemos como referencia la variable especie que se podrá utilizar para validar el resultado del cluster y evaluar su calidad. Los datos que conforman el dataset se encuentran listados en la Tabla A.1 del apéndice A.

La distribución de las especies la podemos ver si desde **KLASS** hacemos una Tabla de frecuencias (Tabla 5.1):

Taula de freqüències				
Modalitats	**Freq. absol.**	**Freq. acum.**	**Freq. relat.**	**Freq. rel. acum.**
Iris-versicolor	50	50	0.3333	0.3333
Iris-virginica	50	100	0.3333	0.6667
Iris-setosa	50	150	0.3333	1
dades mancants	0	N = 150	0	

Tabla 5.1: Tabla de frecuencias de las especies.

También podemos mostrar la descriptiva univariante de las demás variables (Tabla 5.2 y Figura 5.2):

Estadístics sumaris	sepalLength	sepalWidth	petalLength	petalWidth
Mitjana	5.8433	3.7587	1.1987	3.054
Mediana	5.8	4.35	1.3	3
Primer quartil (Q1)	5.1	1.6	0.3	2.8
Tercer quartil (Q3)	6.4	5.1	1.8	3.3
Mínim	4.3	1	0.1	2
Màxim	7.9	6.9	2.5	4.4
Quasi-desviació típica	0.8281	1.7644	0.7632	0.4336
Coeficient de variació	0.1412	0.4679	0.6345	0.1415
Nombre d'objectes	150			
Nombre de dades mancants	0			
Nombre d'observacions útils	150			

Tabla 5.2: Estadísticos sumarios de variables de Iris

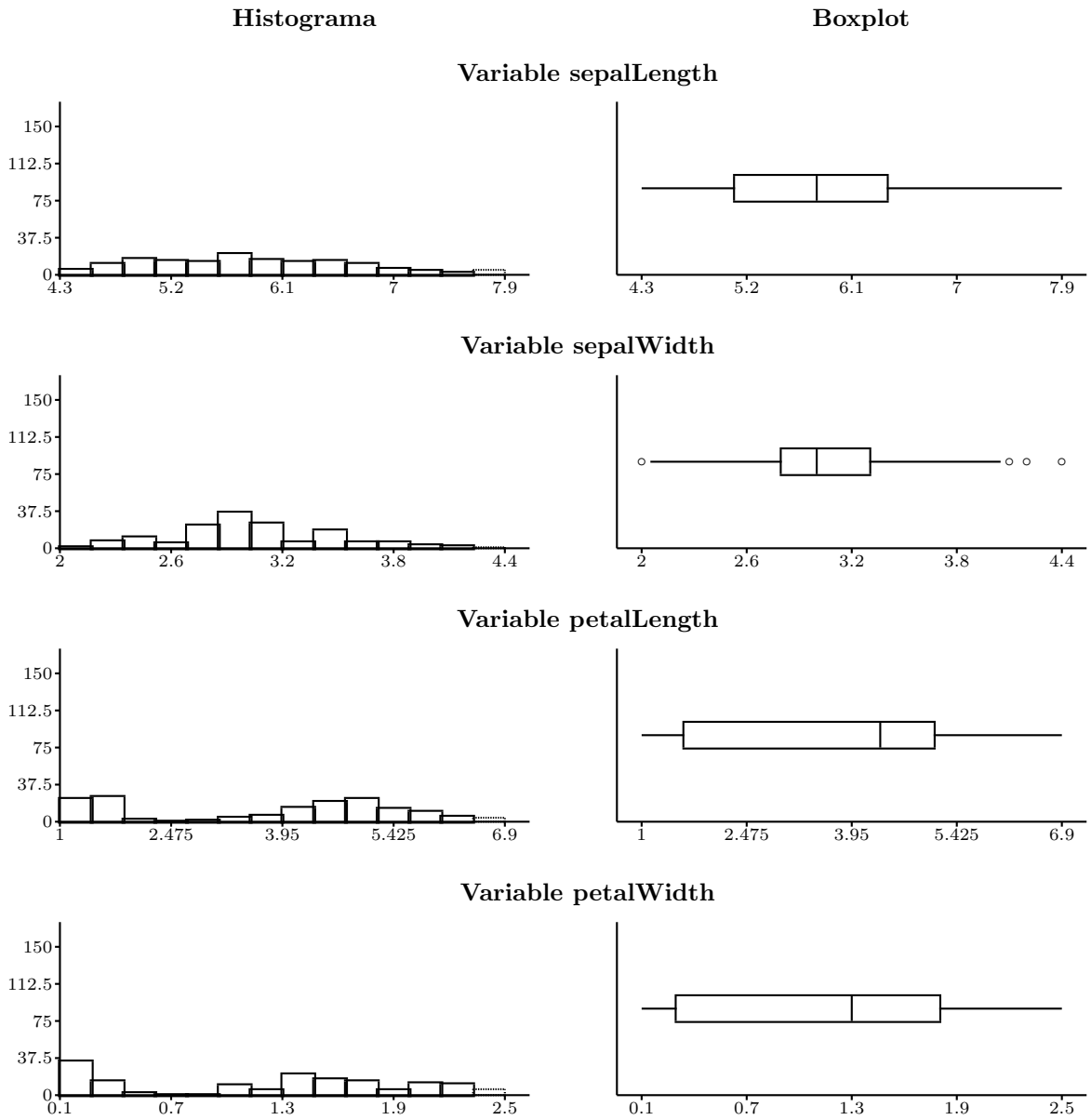


Figura 5.2: Histogramas y boxplots de variables numéricas de Iris

La distribución de las variables condicionada a la especie se ve en la Figura 5.3:

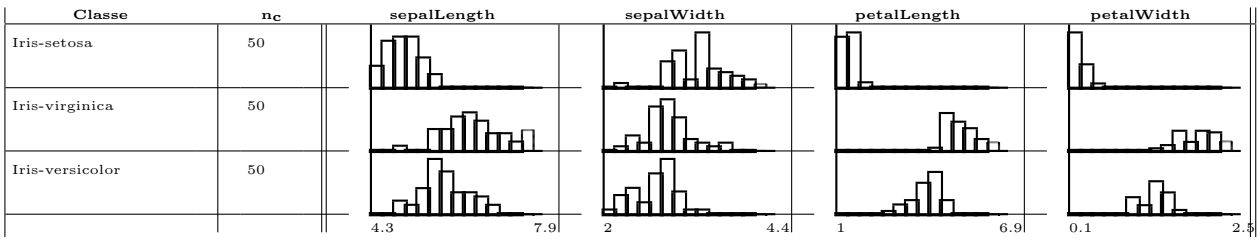


Figura 5.3: Descriptiva de la variable Species

A partir del CPG pareciera que las dimensiones de los pétalos debieran tener mayor poder explicativo. Vemos cómo *Iris-setosa* se diferencia de las otras especies en que tiene los pétalos más cortos y estrechos. También vemos que tiene los sépalos más cortos y anchos que las otras dos y por tanto cómo decíamos

en la introducción de este caso, se presume una clara separación entre la especie *I.setosa* y las otras dos, pero quizás no tan clara entre *I.virginica* e *I.versicolor*.

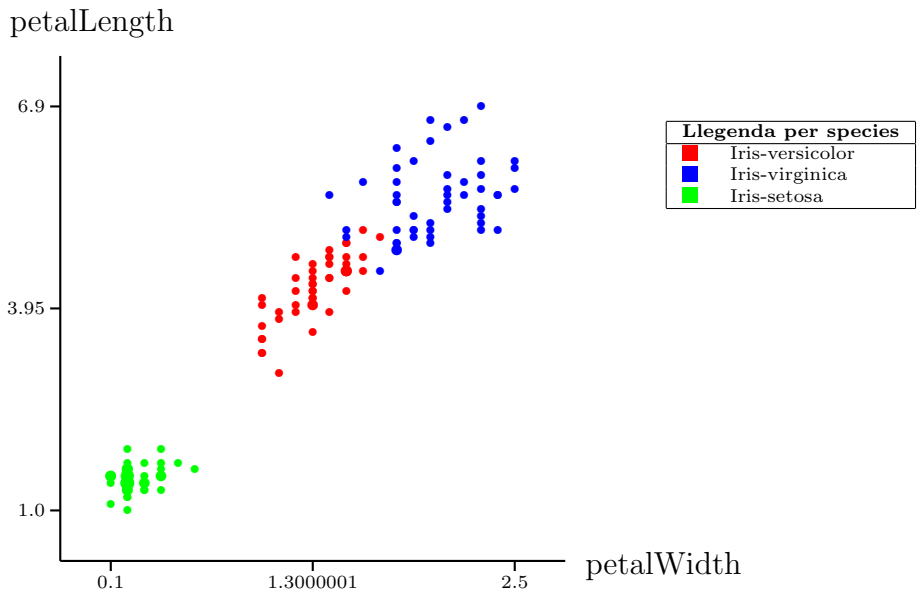


Figura 5.4: Letterplot de las variables petalWidth y petalLength vs. Species

La Figura 5.4 muestra un plot de las medidas del sépalo condicionado a la especie donde, de acuerdo con lo explicado en la presentación de esta base de datos, se observa cómo existe una relación global entre la longitud y la anchura de los pétalos; para todo el género y todas las especies, los datos se alinean alrededor de una única recta que representa homogeneidad en la proporción entre ancho y largo de los pétalos para todo el género Iris. La especie *I.setosa* tiene valores mucho menores que las demás, configurando pétalos más pequeños. Las flores de esta especie forman una clase muy compacta claramente diferenciada de las demás y linealmente separable del resto de especies. Sin embargo, para las otras 2 especies, existe una clara tendencia de la *I.versicolor* a tomar valores intermedios y de la *I.virginica* a tomar valores mayores, pero la frontera entre estas dos especies ya es más borrosa y no seria posible hallar una separación lineal entre estas dos.

5.1.1. Ejecución con Vecinos recíprocos

Ahora vamos a ver qué pasa si intentamos clasificar este conjunto de datos con *Vecinos recíprocos*, con el criterio de Ward y con métrica Euclídea (Figura 5.5). Para clasificar se usan sólo las variables petalLength, petalWidth, sepalLength y sepalWidth, remarcamos que no se ha utilizado la variable de especie.

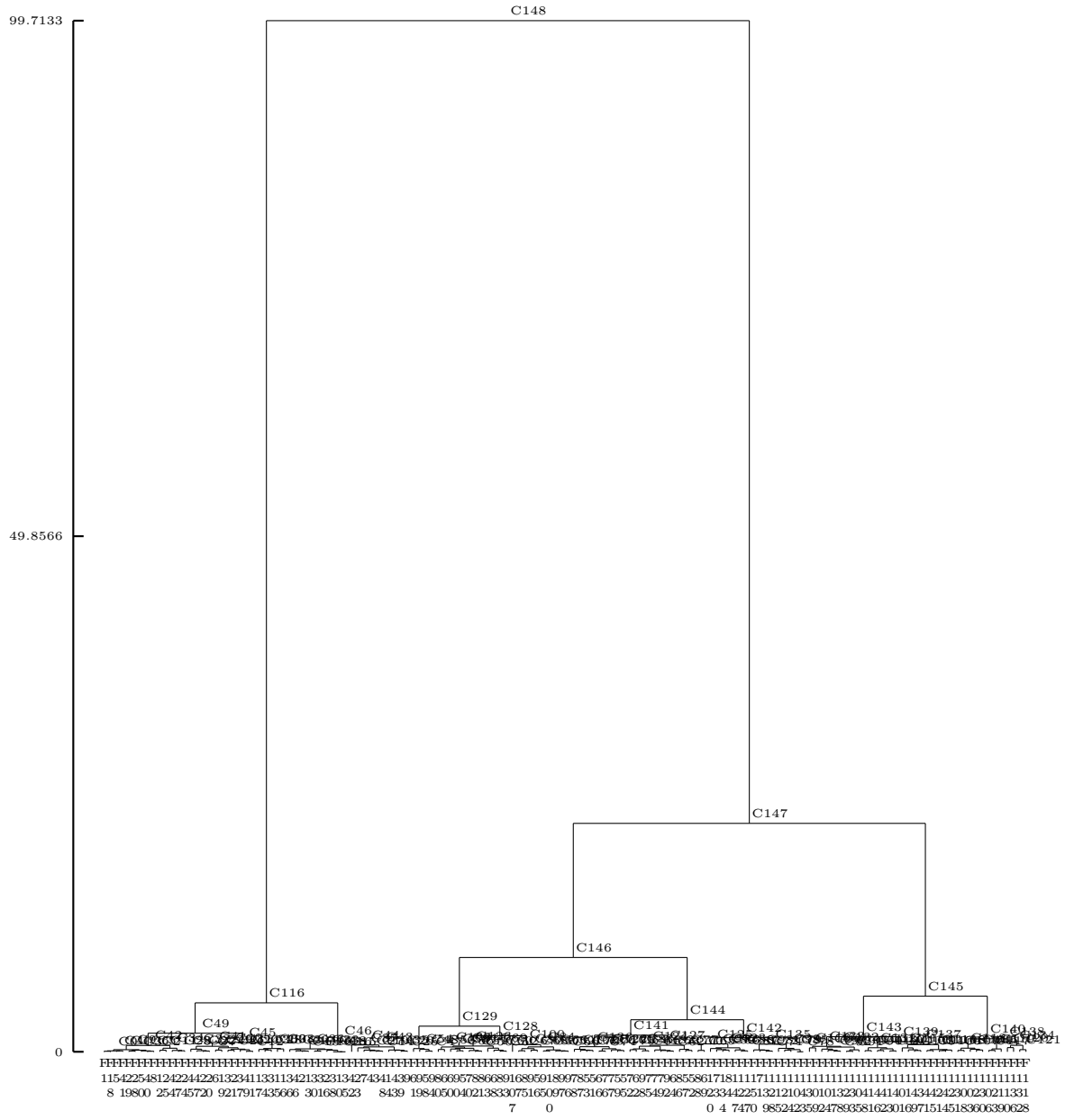


Figura 5.5: Árbol general de clasificación con vecinos recíprocos para Iris.

El resultado del dendrograma de la Figura 5.5 sugiere un corte en 2 ó 3 clases. Es más probable que el corte en 2 separe las *I.setosa* de las demás. Como en este caso particular conocemos la existencia de 3 especies de flores en la base de datos, realizamos el corte en 3.

El panel de clases (Figura: 5.6) visualiza la distribución condicionada a las clases. Pareciera que esta clasificación identifica una clase de flores con pétalos más pequeños y sépalos más cortos y anchos (C116), otra con pétalos intermedios, sépalos más estrechos y de longitud intermedia (C146) y una tercera con pétalos más grandes y sépalos más largos y estrechos (C145).

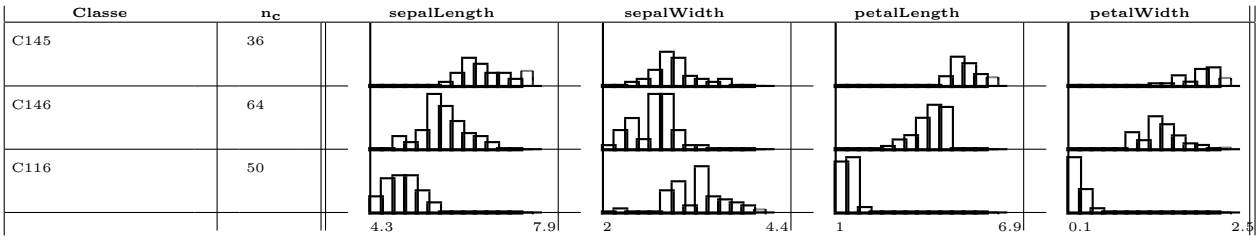


Figura 5.6: Descriptiva de clases de la ejecución Vecinos recíprocos para Iris.

	CLASSE	**C116**	**C146**	**C145**
VARIABLE	N = 150	$n_c = 50$	$n_c = 64$	$n_c = 36$
sepalLength	$\bar{\mathbf{X}}$	5.006	5.9297	6.8528
	S	0.3525	0.4856	0.5074
	min	4.3	4.9	6.1
	max	5.8	7	7.9
	N*	0	0	0
	Me	5	5.9	6.7
sepalWidth	$\bar{\mathbf{X}}$	3.418	2.7578	3.075
	S	0.381	0.2964	0.298
	min	2.3	2	2.5
	max	4.4	3.4	3.8
	N*	0	0	0
	Me	3.4	2.8	3
petalLength	$\bar{\mathbf{X}}$	1.464	4.4109	5.7861
	S	0.1735	0.5103	0.463
	min	1	3	5.1
	max	1.9	5.1	6.9
	N*	0	0	0
	Me	1.5	4.5	5.7
petalWidth	$\bar{\mathbf{X}}$	0.244	1.4391	2.0972
	S	0.1072	0.2947	0.2624
	min	0.1	1	1.4
	max	0.6	2.4	2.5
	N*	0	0	0
	Me	0.2	1.4	2.1

Tabla 5.3: Descriptiva por grupos ejecución vecinos recíprocos para Iris.

En la Tabla 5.3 vemos cómo hay una clara diferencia entre la longitud y anchura de los pétalos de la clase C116 con las otras clases, por tanto podemos pensar que este método ha identificado perfectamente el conjunto de datos de *Iris-setosa*. También podemos observar cómo la longitud y anchura de los pétalos y sépalos de la clase C145 es ligeramente superior que en las otras clases.
Realizamos una comparación cruzando esta variable de clase con la especie (Figura 5.4).

Species \ VRE	**C116**	**C146**	**C145**	**útils**	**mancants**
Iris-versicolor	0	50	0	50	0
Iris-virginica	0	14	36	50	0
Iris-setosa	50	0	0	50	0
útils	50	64	36	150	
mancants	0	0	0		0

Tabla 5.4: Tabla de contingencia Species/ejecución Vecinos recíprocos para Iris.

Vemos cómo al parecer el clustering distribuye los datos en 3 clases de tamaño equitativo. C116 reconoce perfectamente *Iris-setosa*, C145 contiene únicamente elementos de la especie *Iris-virginica* pero no los incluye todos. Existen 14 flores *Iris-virginica* de la frontera que el método de clustering confunde con *Iris-versicolor*.

5.1.2. Ejecución con Clasificación condicionada

Vamos a usar ahora el método de *Clasificación condicionada*, con el criterio de Ward, métrica Euclídea y tomando como variable de condición la especie (Figura 5.7), ya que en este caso excepcionalmente disponemos de esta variable.

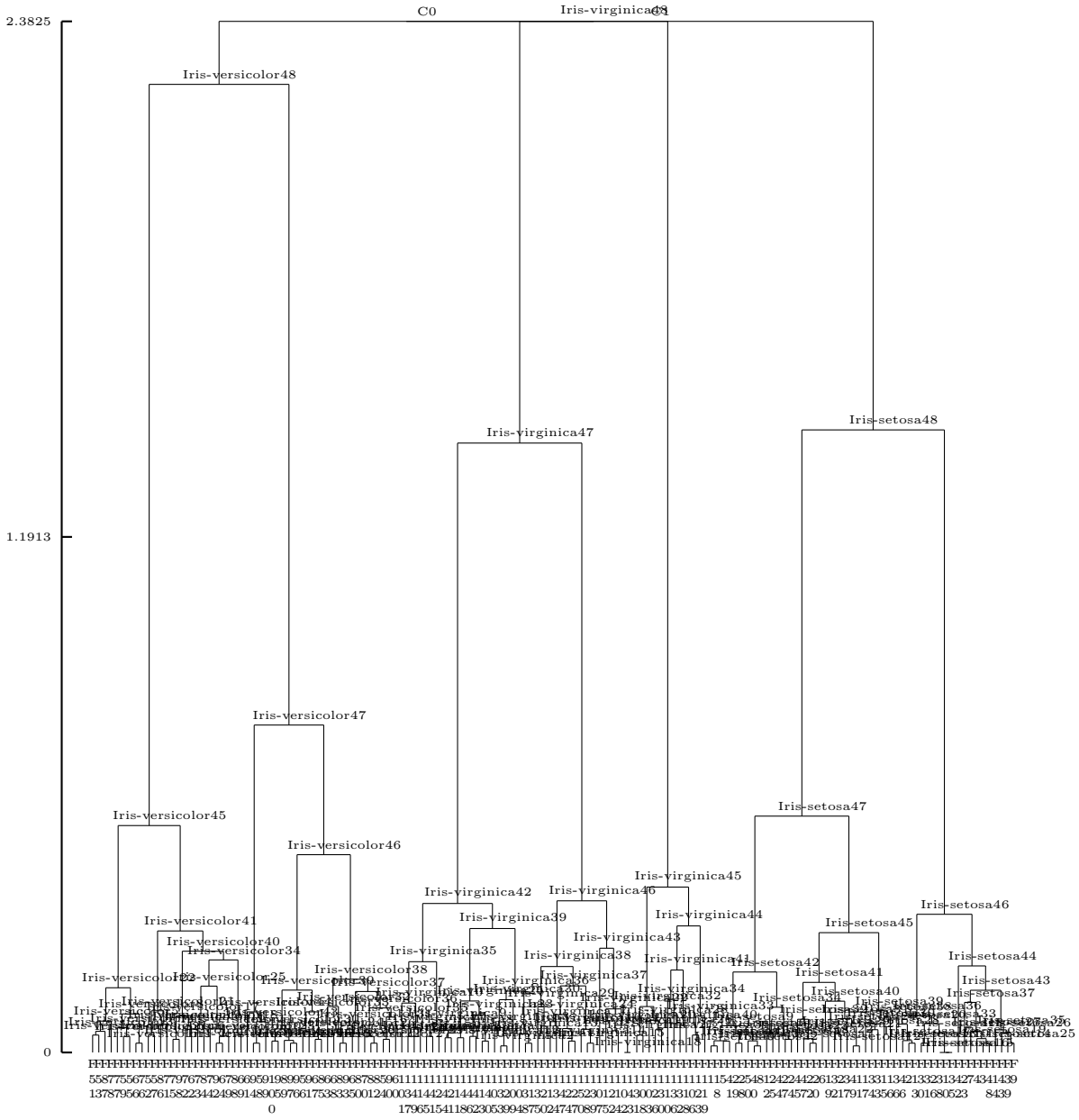


Figura 5.7: Árbol general de clasificación con clasificación condicionada para Iris.

El resultado del dendrograma de la Figura 5.7 sugiere un corte en 5 clases.

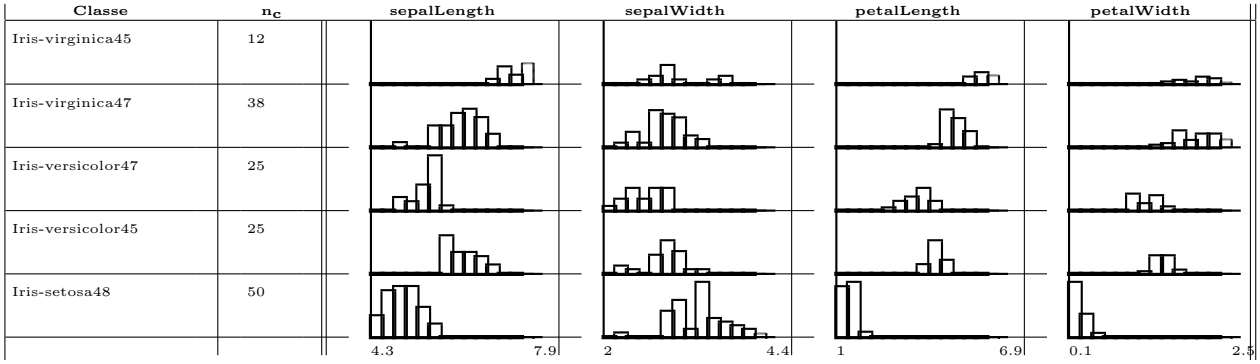


Figura 5.8: Descriptiva de clases de la ejecución Clasificación condicionada para Iris.

En la Figura 5.8 vemos cómo se subdivide la especie *I.virginica* en dos clases con anchuras parecidas de sépalos y pétalos pero una de ellas tiene sépalos más largos. Con la especie *I.versicolor* pasa algo similar, subdivide la especie en dos, una con pétalos y sépalos intermedios y otra con pétalos y sépalos más cortos y pétalos más estrechos.

	CLASSE	**Iris-setosa48**	**Iris-versicolor45**	**Iris-versicolor47**	**Iris-virginica47**	**Iris-virginica45**
VARIABLE	N = 150	$n_c = 50$	$n_c = 25$	$n_c = 25$	$n_c = 38$	$n_c = 12$
sepalLength	$\bar{\mathbf{X}}$	5.006	6.352	5.52	6.3079	7.475
	S	0.3525	0.3242	0.2799	0.422	0.2701
	min	4.3	5.9	4.9	4.9	7.1
	max	5.8	7	6	6.9	7.9
	N*	0	0	0	0	0
	Me	5	6.3	5.6	6.35	7.5
sepalWidth	$\bar{\mathbf{X}}$	3.418	2.92	2.62	2.9263	3.125
	S	0.381	0.2828	0.2723	0.2844	0.398
	min	2.3	2.2	2	2.2	2.6
	max	4.4	3.4	3	3.4	3.8
	N*	0	0	0	0	0
	Me	3.4	2.9	2.6	3	3
petalLength	$\bar{\mathbf{X}}$	1.464	4.592	3.928	5.3158	6.3
	S	0.1735	0.2564	0.3943	0.356	0.3568
	min	1	4	3	4.5	5.8
	max	1.9	5.1	4.5	6	6.9
	N*	0	0	0	0	0
	Me	1.5	4.6	4	5.3	6.2
petalWidth	$\bar{\mathbf{X}}$	0.244	1.456	1.196	2.0184	2.05
	S	0.1072	0.1387	0.1594	0.2837	0.2541
	min	0.1	1.2	1	1.4	1.6
	max	0.6	1.8	1.5	2.5	2.5
	N*	0	0	0	0	0
	Me	0.2	1.5	1.2	2	2.05

Tabla 5.5: Descriptiva por grupos ejecución clasificación condicionada para Iris.

En la Tabla 5.5 vemos cómo de nuevo la longitud y anchura de los pétalos de una de las clases (I.setosa48) es muy diferente de las otras. Este método también identifica perfectamente los elementos de la clase *Iris-setosa*. En cambio hay una subdivisión de las otras 2 especies en 2 subclases cada una, donde parece ser que la longitud del sépalo es la que crearía mayores diferencias entre subespecies. En realidad la confusión se produciría entre *versicolor* y *virginica* que contiene las flores de la frontera entre ambas clases.

Species \ CCondicionada	**setosa48**	**versicolor45**	**versicolor47**	**virginica47**	**virginica45**	**útils**	**mancants**
Iris-versicolor	0	25	25	0	0	50	0
Iris-virginica	0	0	0	38	12	50	0
Iris-setosa	50	0	0	0	0	50	0
útils	50	25	25	38	12	150	
mancants	0	0	0	0	0		0

Tabla 5.6: Tabla de contingencia Species/ejecución Clasificación condicionada.

En la Tabla 5.6 vemos que si utilizamos la variable de clase como condición este algoritmo clasifica perfectamente los datos en una de las clases (I.setosa48) que se identifica con la *I.setosa*, pero luego

reparte en 4 clases más el resto de datos. La especie *Iris-versicolor* la reparte en dos clases, la especie *Iris-virginica* también y la que clasifica perfectamente es *Iris-setosa*. Por tanto este método ya no se equivoca asignando flores de la especie *virginica* a la *versicolor* por el hecho de tratar 2 subespecies dentro de éstas, pero hay que tener en cuenta que esto corresponde a un enfoque supervisado donde la variable de clase se conoce a priori y entra en el modelo.

5.1.3. Ejecución con DBSCAN

Ahora ejecutaremos el algoritmo DBSCAN con los datos mencionados utilizando la distancia Euclídea y diferentes valores de ε y ν para ver la calidad de la clasificación. Para determinar los valores de ε y ν seguiremos las indicaciones que se dan en la sección 3.1.2. La Figura 5.9 muestra el gráfico 4-dist donde podemos ver los valles.

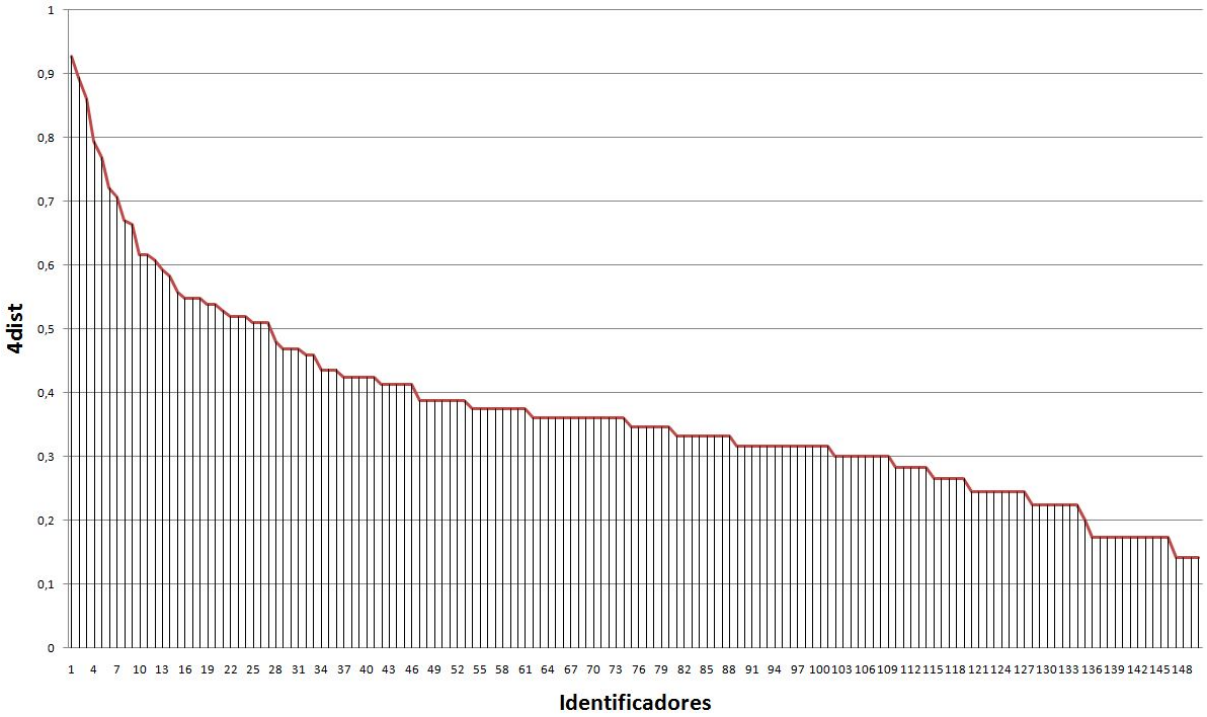


Figura 5.9: Gráfica 4-dist para Iris.

Podemos intuir que uno de los valles más claros se encuentra alrededor de 0.55 de distancia por tanto elegimos $\varepsilon = 0{,}55$ y $\nu = 4$.

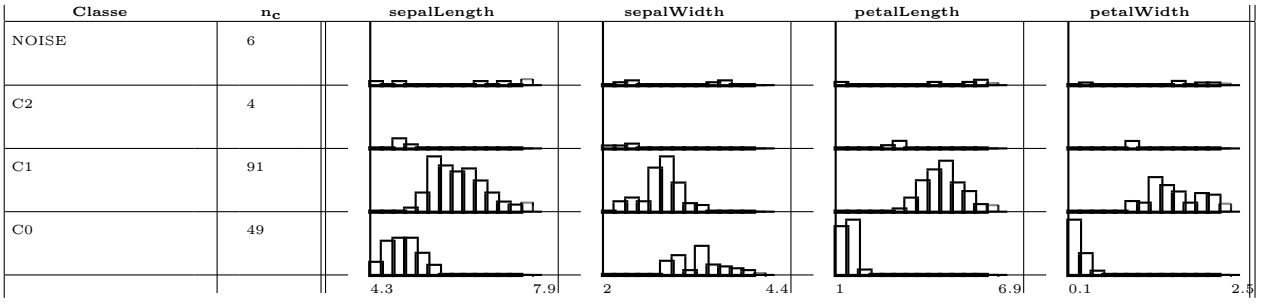


Figura 5.10: Descriptiva de clases de la ejecución DBSCAN ($\varepsilon = 0{,}55$ y $\nu = 4$).

En la Figura 5.10, vemos cómo DBSCAN diferencia la clase (C0) de las demás por la forma de sus

pétalos, el resto de flores las clasifica en una clase con valores intermedios para los pétalos y los sépalos (C1) y luego considera dos subgrupos de flores más. Uno de los grupos considera que son valores de ruido (NOISE) y el otro (con valores de sépalos más cortos y estrechos) considera que pertenecen a otra clase (C2).

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 150	$n_c = 49$	$n_c = 91$	$n_c = 4$	$n_c = 6$
sepalLength	$\bar{\mathbf{X}}$	5.0163	6.2835	5	6.4833
	S	0.3484	0.5714	0.0817	1.4483
	min	4.3	5.2	4.9	4.5
	max	5.8	7.7	5.1	7.9
	N*	0	0	0	0
	Me	5	6.3	5	6.95
sepalWidth	$\bar{\mathbf{X}}$	3.4408	2.8769	2.3	3.0833
	S	0.3488	0.2785	0.216	0.7195
	min	2.9	2.2	2	2.3
	max	4.4	3.4	2.5	3.8
	N*	0	0	0	0
	Me	3.4	2.9	2.35	3.05
petalLength	$\bar{\mathbf{X}}$	1.4673	4.9231	3.275	5.1333
	S	0.1737	0.7367	0.2062	2.0265
	min	1	3.5	3	1.3
	max	1.9	6.9	3.5	6.7
	N*	0	0	0	0
	Me	1.5	4.9	3.3	5.95
petalWidth	$\bar{\mathbf{X}}$	0.2429	1.6846	1.025	1.75
	S	0.108	0.4093	0.05	0.7662
	min	0.1	1	1	0.3
	max	0.6	2.5	1.1	2.5
	N*	0	0	0	0
	Me	0.2	1.6	1	1.9

Tabla 5.7: Descriptiva por grupos ejecución DBSCAN ($\varepsilon = 0{,}55$ y $\nu = 4$).

En la Tabla 5.7 vemos cómo este método también diferencia una clase de las otras por la longitud y anchura de sus pétalos (C0), de nuevo esta clase sería la especie *Iris-setosa*.

Species \ DBSCAN	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
Iris-versicolor	0	46	4	0	50	0
Iris-virginica	0	45	0	5	50	0
Iris-setosa	49	0	0	1	50	0
útils	49	91	4	6	150	
mancants	0	0	0	0		0

Tabla 5.8: Tabla de contingencia Species/ejecución DBSCAN ($\varepsilon = 0{,}55$ y $\nu = 4$).

Como podemos ver en la Tabla 5.8 los elementos de la clase *Iris-setosa* los clasifica casi todos en la misma clase y los del resto de especies en una clase mayoritaria. Como ya sabíamos con anterioridad este conjunto datos tiene 1 clase muy diferenciada de las otras dos que son más difíciles de separar.

Vemos ahora qué pasa si usamos otro valor de ε, por ejemplo: $\varepsilon = 0{,}42$. También aumentamos el valor de ν para que no haga clases sólo con 4 elementos. Elegimos $\nu = 5$.

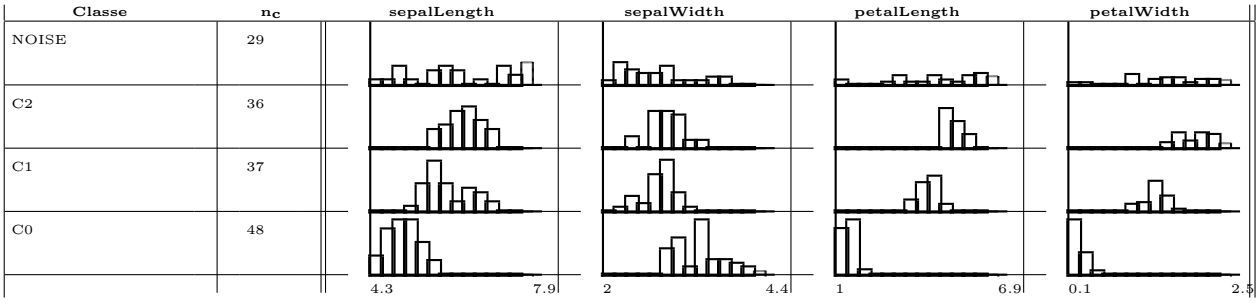


Figura 5.11: Descriptiva de clases de la ejecución DBSCAN ($\varepsilon = 0,42$ y $\nu = 5$).

En la Figura 5.11 vemos ahora una mejor clasificación de los datos ya que clasifica los elementos en 3 clases diferenciadas, pero obtenemos una mayor presencia de ruido. Los elementos de la clase C0 son flores con pétalos más cortos y estrechos, que coincide con las características de *I.setosa*. Los elementos de las clases C1 y C2 tienen sépalos con valores intermedios, sin embargo, se diferencian en que una (C2) tiene los pétalos más largos y anchos que la otra (C1).

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 150	$n_c = 48$	$n_c = 37$	$n_c = 36$	$n_c = 29$
sepalLength	$\bar{\mathbf{X}}$	5.025	6.0216	6.3667	6.3207
	S	0.3467	0.4928	0.3711	1.0884
	min	4.3	5.2	5.6	4.5
	max	5.8	7	7.1	7.9
	N*	0	0	0	0
	Me	5	5.9	6.4	6.2
sepalWidth	$\bar{\mathbf{X}}$	3.4375	2.8459	2.9639	2.7966
	S	0.3517	0.2364	0.2451	0.5074
	min	2.9	2.3	2.5	2
	max	4.4	3.3	3.4	3.8
	N*	0	0	0	0
	Me	3.4	2.9	3	2.8
petalLength	$\bar{\mathbf{X}}$	1.4771	4.3405	5.2972	4.8828
	S	0.1614	0.3312	0.3342	1.587
	min	1.1	3.7	4.8	1
	max	1.9	5	5.9	6.9
	N*	0	0	0	0
	Me	1.5	4.4	5.25	5.1
petalWidth	$\bar{\mathbf{X}}$	0.2437	1.3405	2.0111	1.5897
	S	0.109	0.1572	0.2561	0.6073
	min	0.1	1	1.5	0.2
	max	0.6	1.7	2.5	2.5
	N*	0	0	0	0
	Me	0.2	1.3	2	1.6

Tabla 5.9: Descriptiva por grupos ejecución DBSCAN ($\varepsilon = 0,42$ y $\nu = 5$).

En la Tabla 5.9 vemos cómo sigue identificando una clase (C0) por la anchura y longitud de sus pétalos. Podemos pensar que estaría identificando los elementos de *I.setosa*. Luego diferencia dos clases con una media de longitud y anchura de pétalos ligeramente diferente y por último elementos que no ha

podido clasificar correctamente los introduce en NOISE.

Species \ DBSCAN	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
Iris-versicolor	0	37	3	10	50	0
Iris-virginica	0	0	33	17	50	0
Iris-setosa	48	0	0	2	50	0
útils	48	37	36	29	150	
mancants	0	0	0	0		0

Tabla 5.10: Tabla de contingencia Species/ejecución DBSCAN ($\varepsilon = 0{,}42$ y $\nu = 5$).

En la Tabla 5.10 vemos cómo ahora además de introducir en C0 únicamente flores *I.setosa*, introduce en C1 flores que sólo pertenecen a *I.versicolor*. La clase C2 la conforman flores del tipo *I.virginica* mayoritariamente, aunque también introduce por error elementos que son de la especie *I.versicolor*, si bien son muy pocos. El resto de elementos no los sabe clasificar y los introduce en el conjunto RUIDO.

VRE \ DBSCAN	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C116	48	0	0	2	50	0
C146	0	37	14	13	64	0
C145	0	0	22	14	36	0
útils	48	37	36	29	150	
mancants	0	0	0	0		0

Tabla 5.11: Tabla de contingencia VRE/ejecución DBSCAN ($\varepsilon = 0{,}42$ y $\nu = 5$).

En la Tabla 5.11 vemos una comparación de la clasificación que hacen los métodos de vecinos recíprocos y DBSCAN ($\varepsilon = 0{,}42$, $\nu = 5$). Observamos cómo la clasificación de los elementos de *I.setosa* es idéntica en ambos casos. Sin embargo, los 14 elementos de *I.virginica* que vecinos recíprocos confundía con *I.versicolor*, DBSCAN los está introduciendo en la clase correcta, aunque otros que deberían ir en *I.virginica* los considera ruido, por tanto no se equivoca clasificándolos, simplemente no los clasifica. Podemos decir que DBSCAN se equivoca menos a la hora de clasificar las flores, pero por contra tenemos un número considerable de flores sin clasificar.

CCondicionada \ DBSCAN	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
Iris-setosa48	48	0	0	2	50	0
Iris-versicolor45	0	19	3	3	25	0
Iris-versicolor47	0	18	0	7	25	0
Iris-virginica47	0	0	32	6	38	0
Iris-virginica45	0	0	1	11	12	0
útils	48	37	36	29	150	
mancants	0	0	0	0		0

Tabla 5.12: Tabla de contingencia CC/ejecución DBSCAN ($\varepsilon = 0{,}42$ y $\nu = 5$).

En la Tabla 5.12 hacemos una comparativa de Clasificación condicionada y DBSCAN ($\varepsilon = 0{,}42$, $\nu = 5$). Ahora vemos cómo además de coincidir en la identificación de la especie *I.setosa*, estos dos métodos discrepan en 3 elementos. DBSCAN considera que hay 3 elementos de la clase Iris-versicolor45 que deberían ir en otra y se equivoca. Además DBSCAN no subdivide los elementos de *I.versicolor* e *I.virginica* en dos subclases pero volvemos a tener la presencia de ruido que hace que tengamos elementos

sin clasificar, la mayoría de los cuáles corresponden precisamente a las Iris-virginica45 de la Clasificación condicionada que era una de las clases de confusión.

A medida que vamos disminuyendo ε (mínima distancia del cluster) se van separando los datos en más clases y no hacemos que la clasificación se acerque más a la realidad sino todo lo contrario.

5.1.4. Ejecución con OPTICS

Vamos a ver ahora de qué forma clasifica los datos OPTICS para los mismos valores de ε y ν:

Para $\varepsilon = 0{,}55$ y $\nu = 4$, el Reachability Plot generado quedaría como el que se ve en la Figura 5.12:

Clustering OPTICS

Radi màxim: 0.55
Cluster mínim: 4
Eix y: Automàtic
Mètrica: Euclídea *(Paràmetres:* No normalitzar; Distàncies no ponderades; Distàncies no al quadrat)

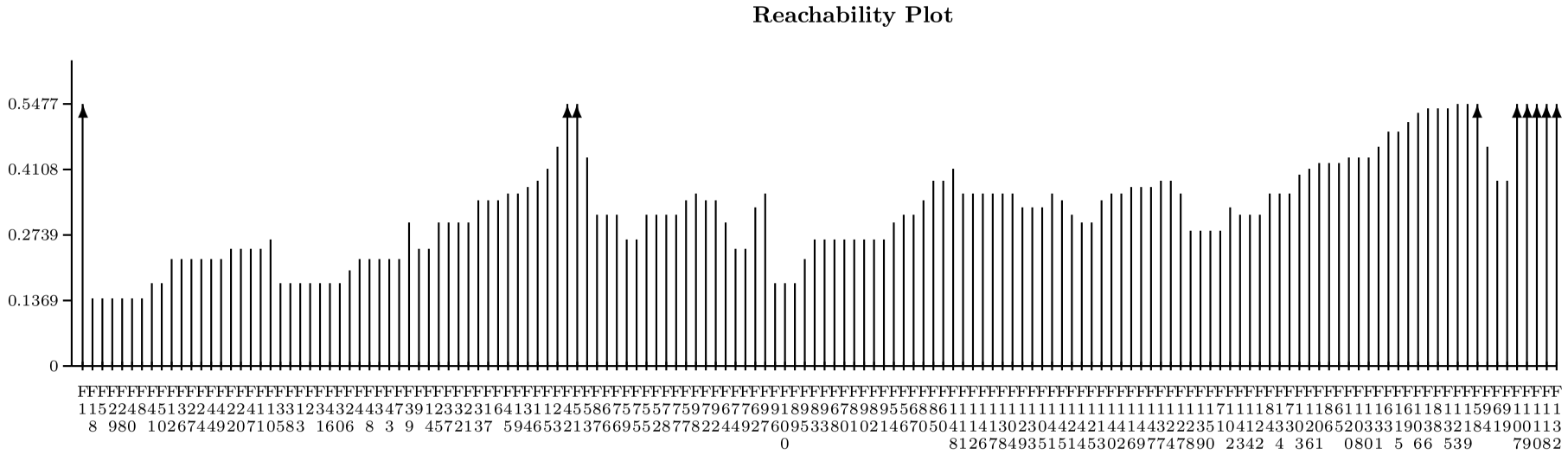


Figura 5.12: Reachability Plot para $\varepsilon = 0{,}55$ y $\nu = 4$ con los datos de Iris.

Ahora deberíamos cortar a una altura concreta para que entonces extraiga los clusters, cortamos en 0.41 ya que es un valor que nos hace pensar que encontrará 3 clases. En Reachability de la Figura 5.13 muestra el corte.

Clustering OPTICS

Radi màxim: 0.55
Cluster mínim: 4
Eix y: Automàtic
Mètrica: Euclídea *(Paràmetres:* No normalitzar; Distàncies no ponderades; Distàncies no al quadrat)

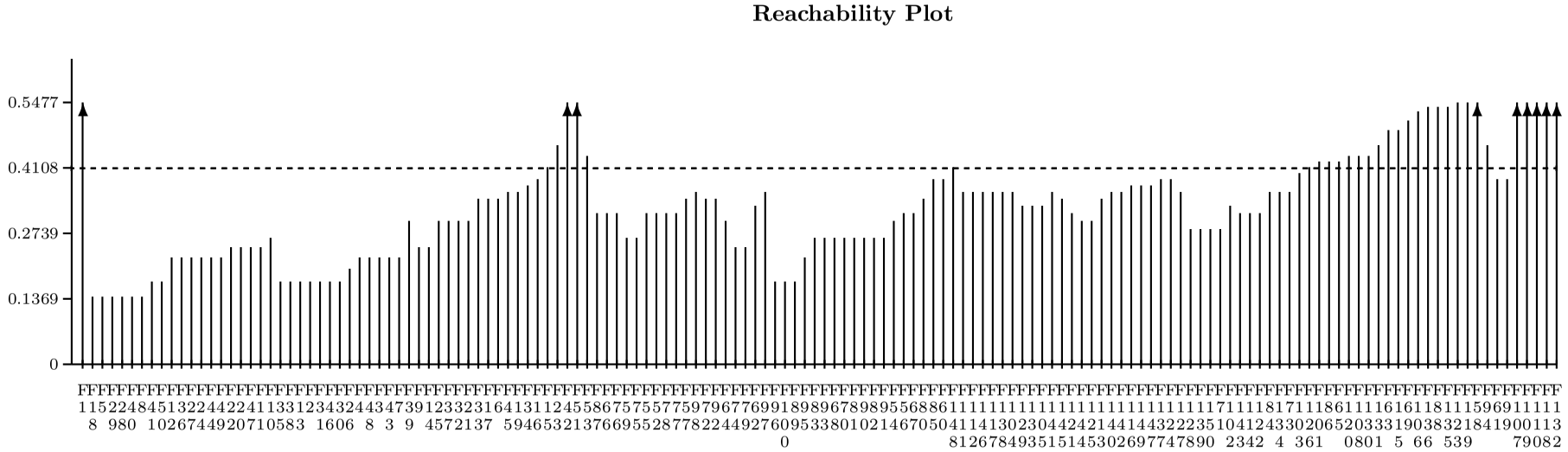


Figura 5.13: Reachability Plot cortado en $\varepsilon' = 0{,}41$ para $\varepsilon = 0{,}55$ y $\nu = 4$ con los datos de Iris.

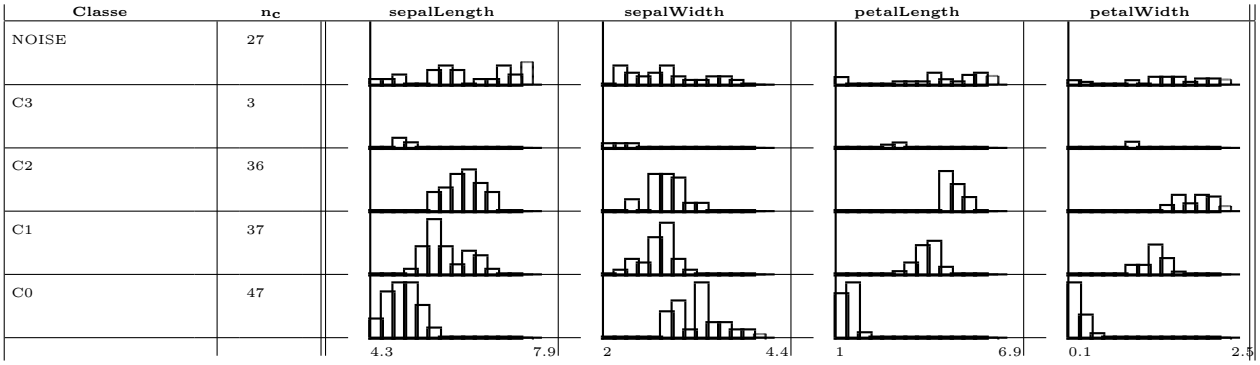


Figura 5.14: Descriptiva de clases de la ejecución OPTICS ($\varepsilon = 0{,}55$ y $\nu = 4$).

En la Figura 5.14 vemos cómo se clasifican los elementos en 4 clases diferentes, aunque tenemos muchos elementos que se consideran ruido. Los elementos de la clase C0 son flores con pétalos más cortos y estrechos, que coinciden con las características de *I.setosa*. Los elementos de las clases C1 y C2 tienen sépalos con valores intermedios, sin embargo, se diferencian en que una (C2) tiene los pétalos más largos y anchos que la otra (C1). En la clase C3 introduce flores que tienen los sépalos y los pétalos más cortos y estrechos y es una clase con muy pocos elementos (3). Los elementos de NOISE son flores de muchas características diferentes, que OPTICS no ha sabido clasificar.

La Tabla 5.13 da detalles estadísticos sobre la distribución condicionada a las clases.

	CLASSE	**C0**	**C1**	**C2**	**C3**	**NOISE**
VARIABLE	N = 150	$n_c = 47$	$n_c = 37$	$n_c = 36$	$n_c = 3$	$n_c = 27$
sepalLength	$\bar{\mathbf{X}}$	5.0085	5.9865	6.3667	5.0333	6.4926
	S	0.3309	0.4668	0.3711	0.0577	1.0329
	min	4.3	5.2	5.6	5	4.5
	max	5.7	6.9	7.1	5.1	7.9
	N*	0	0	0	0	0
	Me	5	5.8	6.4	5	6.3
sepalWidth	$\bar{\mathbf{X}}$	3.4255	2.8297	2.9639	2.2667	2.9222
	S	0.3454	0.232	0.2451	0.2517	0.533
	min	2.9	2.3	2.5	2	2.2
	max	4.4	3.3	3.4	2.5	4
	N*	0	0	0	0	0
	Me	3.4	2.9	3	2.3	2.9
petalLength	$\bar{\mathbf{X}}$	1.483	4.3081	5.2972	3.2667	4.9704
	S	0.1579	0.3531	0.3342	0.2517	1.6875
	min	1.1	3.5	4.8	3	1
	max	1.9	5	5.9	3.5	6.9
	N*	0	0	0	0	0
	Me	1.5	4.3	5.25	3.3	5.6
petalWidth	$\bar{\mathbf{X}}$	0.2447	1.3297	2.0111	1.0333	1.6148
	S	0.11	0.1664	0.2561	0.0577	0.6503
	min	0.1	1	1.5	1	0.2
	max	0.6	1.7	2.5	1.1	2.5
	N*	0	0	0	0	0
	Me	0.2	1.3	2	1	1.7

Tabla 5.13: Descriptiva por grupos ejecución OPTICS ($\varepsilon = 0{,}55$ y $\nu = 4$).

En la Tabla 5.14 vemos cómo nuevamente *I.setosa* se reconoce bien (C0), introduce en C1 y C3 flores

Species \ OPTICS	**C0**	**C1**	**C2**	**C3**	**NOISE**	**útils**	**mancants**
Iris-versicolor	0	37	3	3	7	50	0
Iris-virginica	0	0	33	0	17	50	0
Iris-setosa	47	0	0	0	3	50	0
útils	47	37	36	3	27	150	
mancants	0	0	0	0	0		0

Tabla 5.14: Tabla de contingencia Species/ejecución OPTICS ($\varepsilon = 0{,}55$ y $\nu = 4$).

que sólo pertenecen a *I.versicolor*, teniendo C3 los sépalos más pequeños. La clase C2 la conforman flores del tipo *I.virginica* mayoritáriamente, aunque también introduce por error un pequeño grupo de elementos que son de la especie *I.versicolor*. El resto de elementos no los sabe clasificar y los introduce en el conjunto RUIDO.

VRE \ OPTICS	**C0**	**C1**	**C2**	**C3**	**NOISE**	**útils**	**mancants**
C116	47	0	0	0	3	50	0
C146	0	37	14	3	10	64	0
C145	0	0	22	0	14	36	0
útils	47	37	36	3	27	150	
mancants	0	0	0	0	0		0

Tabla 5.15: Tabla de contingencia VRE/ejecución OPTICS ($\varepsilon = 0{,}55$ y $\nu = 4$).

En la Tabla 5.15 vemos una comparación de la clasificación que hacen los métodos de vecinos recíprocos y OPTICS ($\varepsilon = 0{,}55$, $\nu = 4$). Observamos cómo la clasificación de los elementos de *I.setosa* es idéntica en ambos casos, sin embargo, los elementos de *I.virginica* que vecinos recíprocos clasifica en *I.versicolor*, OPTICS los está introduciendo en el conjunto RUIDO, por tanto no se equivoca clasificándolos, simplemente no los clasifica. Podemos decir que OPTICS se equivoca menos a la hora de clasificar (igual que pasaba con DBSCAN), pero por contra tenemos un número considerable de flores sin clasificar.

CC \ OPTICS	**C0**	**C1**	**C2**	**C3**	**NOISE**	**útils**	**mancants**
Iris-setosa48	47	0	0	0	3	50	0
Iris-versicolor45	0	18	3	0	4	25	0
Iris-versicolor47	0	19	0	3	3	25	0
Iris-virginica47	0	0	32	0	6	38	0
Iris-virginica45	0	0	1	0	11	12	0
útils	47	37	36	3	27	150	
mancants	0	0	0	0	0		0

Tabla 5.16: Tabla de contingencia CC/ejecución OPTICS ($\varepsilon = 0{,}55$ y $\nu = 4$).

En la Tabla 5.16 hacemos una comparativa de Clasificación condicionada y OPTICS ($\varepsilon = 0{,}55$, $\nu = 4$). Ahora vemos cómo además de coincidir en la identificación de la especie *I.setosa*, estos dos métodos discrepan en 3 elementos. OPTICS considera que hay 3 elementos de la clase Iris-versicolor45 que deberían ir en otra (C2) y también considera que hay 3 elementos de la clase Iris-versicolor47 que deberían ir en una clase diferente (C3) y en ambos casos de equivoca. Además OPTICS no subdivide los elementos de *I.versicolor* e *I.virginica* en dos subclases pero volvemos a tener la presencia de ruido que hace que tengamos elementos sin clasificar.

En la Tabla 5.17 observamos cómo DBSCAN y OPTICS clasifican de manera diferente para los

DBSCAN \ OPTICS	**C0**	**C1**	**C2**	**C3**	**NOISE**	**útils**	**mancants**
C0	47	0	0	0	2	49	0
C1	0	37	36	0	18	91	0
C2	0	0	0	3	1	4	0
NOISE	0	0	0	0	6	6	0
útils	47	37	36	3	27	150	
mancants	0	0	0	0	0		0

Tabla 5.17: Tabla de contingencia DBSCAN/ejecución OPTICS($\varepsilon = 0{,}55$ y $\nu = 4$).

mismos parámetros de ε y ν. OPTICS subdivide los elementos de la clase (C1) de DBSCAN en dos clases distintas, cosa que hace correctamente porque son flores de diferentes especies, pero introduce más elementos considerados ruido y tenemos flores que no conseguimos clasificar.

Vemos ahora cómo OPTICS ($\varepsilon = 0{,}55$, $\nu = 4$) clasifica mejor que DBSCAN ($\varepsilon = 0{,}55$, $\nu = 4$). Esto pasa porque OPTICS además da la posibilidad de elegir el nivel de corte del Reachability Plot y esto nos da la posibilidad de en mayor o menor medida elegir cuántas clases queremos construir a posteriori.

Ejecutamos OPTICS ahora con los valores $\varepsilon = 0{,}42$ y $\nu = 5$, el Reachability Plot generado quedaría como el que se ve en la Figura 5.15:

Clustering OPTICS

Radi màxim: 0.42
Cluster mínim: 5
Eix y: Automàtic
Mètrica: Euclídea *(Paràmetres:* No normalitzar; Distàncies no ponderades; Distàncies no al quadrat)

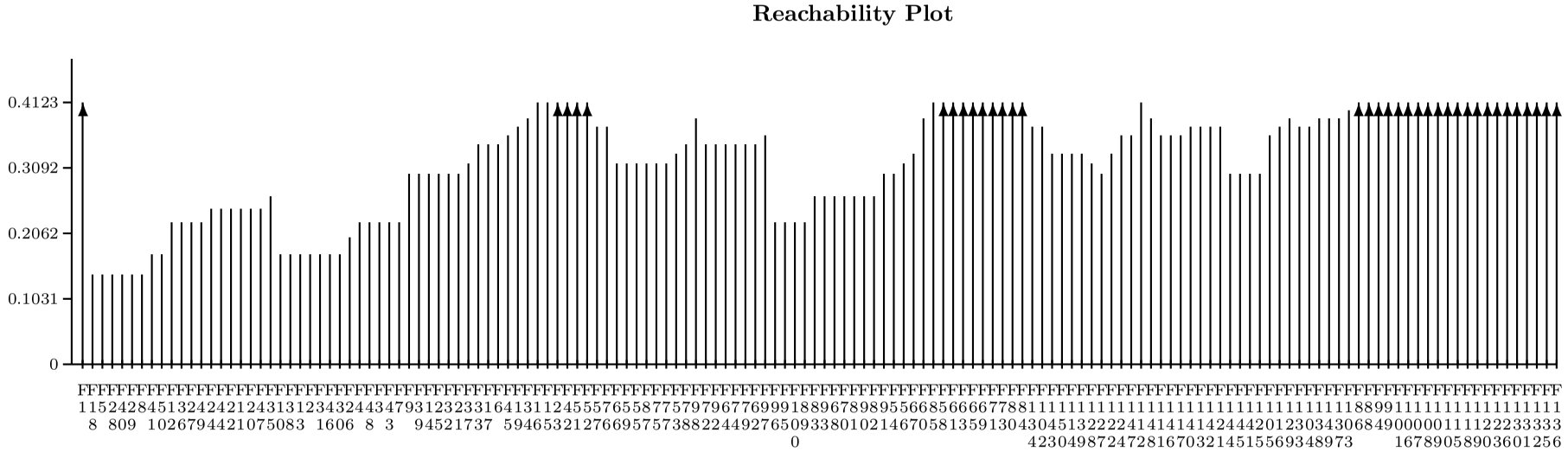


Figura 5.15: Reachability Plot para $\varepsilon = 0{,}42$ y $\nu = 5$ con los datos de Iris.

Ahora cortamos en el máximo valor posible para obtener el mayor número de clases: 0.4123. En la Figura 5.16 mostramos el corte realizado.

Clustering OPTICS

Radi màxim: 0.42
Cluster mínim: 5
Eix y: Automàtic
Mètrica: Euclídea *(Paràmetres:* No normalitzar; Distàncies no ponderades; Distàncies no al quadrat)

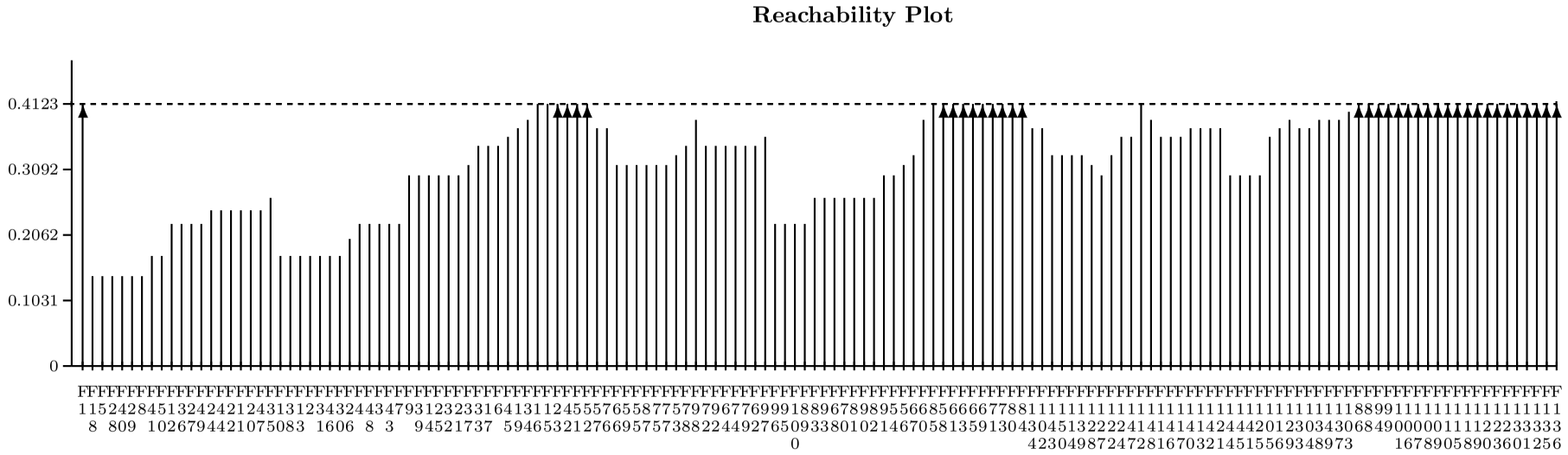


Figura 5.16: Reachability Plot cortado en $\varepsilon' = 0{,}4123$ para $\varepsilon = 0{,}42$ y $\nu = 5$ con los datos de Iris.

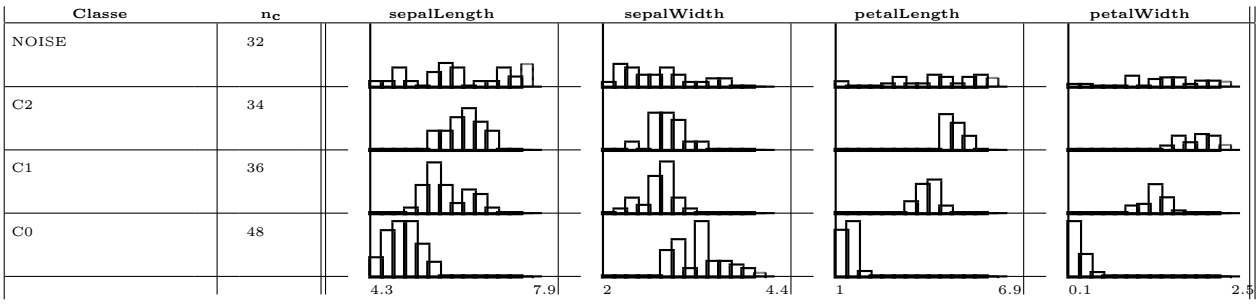


Figura 5.17: Descriptiva de clases de la ejecución OPTICS ($\varepsilon = 0{,}42$ y $\nu = 5$) para Iris.

En la Figura 5.17 vemos ahora una mejor clasificación de los datos ya que clasifica los elementos en 3 clases diferentes, pero obtenemos una mayor presencia de ruido. Los elementos de la clase C0 son flores con pétalos más cortos y estrechos, que coincide con las características de *I.setosa*. Los elementos de las clases C1 y C2 tienen sépalos con valores intermedios, sin embargo, se diferencian en que una (C2) tiene los pétalos más largos y anchos que la otra (C1).

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 150	$n_c = 48$	$n_c = 36$	$n_c = 34$	$n_c = 32$
sepalLength	$\bar{\mathbf{X}}$	5.025	5.9944	6.3824	6.3281
	S	0.3467	0.4708	0.3729	1.0443
	min	4.3	5.2	5.6	4.5
	max	5.8	6.9	7.1	7.9
	N*	0	0	0	0
	Me	5	5.85	6.4	6.25
sepalWidth	$\bar{\mathbf{X}}$	3.4375	2.8361	2.9706	2.8125
	S	0.3517	0.232	0.2355	0.4956
	min	2.9	2.3	2.5	2
	max	4.4	3.3	3.4	3.8
	N*	0	0	0	0
	Me	3.4	2.9	3	2.8
petalLength	$\bar{\mathbf{X}}$	1.4771	4.3306	5.3235	4.875
	S	0.1614	0.3302	0.3248	1.5087
	min	1.1	3.7	4.8	1
	max	1.9	5	5.9	6.9
	N*	0	0	0	0
	Me	1.5	4.35	5.3	4.95
petalWidth	$\bar{\mathbf{X}}$	0.2437	1.3389	2.0324	1.5875
	S	0.109	0.1591	0.2446	0.5796
	min	0.1	1	1.5	0.2
	max	0.6	1.7	2.5	2.5
	N*	0	0	0	0
	Me	0.2	1.3	2	1.6

Tabla 5.18: Descriptiva por grupos ejecución OPTICS ($\varepsilon = 0{,}42$ y $\nu = 5$) con los datos de Iris.

La Tabla 5.18 da detalles estadísticos sobre la distribución condicionada a las clases.

En la Tabla 5.19 vemos cómo ahora además de introducir en C0 únicamente flores *I.setosa*, introduce en C1 flores que sólo pertenecen a *I.versicolor*. La clase C2 la conforman flores del tipo *I.virginica* mayoritáriamente, aunque también introduce por error elementos que son de la especie *I.versicolor*. El resto de elementos no los sabe clasificar y los introduce en el conjunto RUIDO.

Species \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
Iris-versicolor	0	36	1	13	50	0
Iris-virginica	0	0	33	17	50	0
Iris-setosa	48	0	0	2	50	0
útils	48	36	34	32	150	
mancants	0	0	0	0		0

Tabla 5.19: Tabla de contingencia Species/ejecución OPTICS ($\varepsilon = 0{,}42$ y $\nu = 5$).

VRE \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C116	48	0	0	2	50	0
C146	0	36	12	16	64	0
C145	0	0	22	14	36	0
útils	48	36	34	32	150	
mancants	0	0	0	0		0

Tabla 5.20: Tabla de contingencia VRE/ejecución OPTICS ($\varepsilon = 0{,}42$ y $\nu = 5$).

En la Tabla 5.20 vemos una comparación de la clasificación que hacen los métodos de vecinos recíprocos y OPTICS ($\varepsilon = 0{,}42$, $\nu = 5$). Observamos cómo la clasificación de los elementos de *I.setosa* es idéntica en ambos casos, sin embargo, algunos elementos de *I.virginica* que vecinos recíprocos clasifica en *I.versicolor*, OPTICS los está introduciendo en la clase correcta y otros en RUIDO, por tanto no se equivoca clasificándolos, no los clasifica. Podemos decir que OPTICS se equivoca menos a la hora de clasificar las flores, pero por contra tenemos flores sin clasificar.

CC \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
Iris-setosa48	48	0	0	2	50	0
Iris-versicolor45	0	18	1	6	25	0
Iris-versicolor47	0	18	0	7	25	0
Iris-virginica47	0	0	32	6	38	0
Iris-virginica45	0	0	1	11	12	0
útils	48	36	34	32	150	
mancants	0	0	0	0		0

Tabla 5.21: Tabla de contingencia CC/ejecución OPTICS ($\varepsilon = 0{,}42$ y $\nu = 5$).

En la Tabla 5.21 hacemos una comparativa de Clasificación condicionada y OPTICS ($\varepsilon = 0{,}42$, $\nu = 5$). Ahora vemos cómo además de coincidir en la identificación de la especie *I.setosa*, estos dos métodos discrepan en 1 elemento. OPTICS considera que hay 1 elemento de la clase Iris-versicolor45 que debería ir en otra (C2) y se equivoca. Además OPTICS no subdivide los elementos de *I.versicolor* e *I.virginica* en dos subclases pero volvemos a tener la presencia de ruido que hace que tengamos elementos sin clasificar.

En esta última Tabla 5.22 hacemos una comparativa de la clasificación que hacen los dos métodos implementados en este proyecto con los mismos valores de ε y ν. En este caso vemos cómo elementos que DBSCAN ($\varepsilon = 0{,}42$, $\nu = 5$) clasificaba erróneamente en *I.virginica* (C2) elementos que eran de *I.versicolor*, OPTICS ($\varepsilon = 0{,}42$, $\nu = 5$) los clasifica correctamente. También hay 1 elemento que DBSCAN clasifica correctamente en *I.versicolor* (C1) que OPTICS lo considera ruido, pero en general OPTICS se equivoca menos que DBSCAN.

DBSCAN \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C0	48	0	0	0	48	0
C1	0	36	0	1	37	0
C2	0	0	34	2	36	0
NOISE	0	0	0	29	29	0
útils	48	36	34	32	150	
mancants	0	0	0	0		0

Tabla 5.22: Tabla de contingencia DBSCAN/ejecución OPTICS ambos con ($\varepsilon = 0{,}42$ y $\nu = 5$).

5.1.5. Comparación de las ejecuciones

Método	$\frac{Err}{N}$	$\frac{Err+Noise}{N}$	$\frac{Err}{Clasif}$	$\frac{Noise}{N}$
VRE	0.09	0.09	0.09	0
CC	0	0	0	0
DBSCAN ($\varepsilon = 0{,}55$, $\nu = 4$)	0.307	0.347	0.319	0.04
DBSCAN ($\varepsilon = 0{,}42$, $\nu = 5$)	0.02	0.213	0.0248	0.193
OPTICS ($\varepsilon = 0{,}55$, $\nu = 4$)	0.02	0.2	0.024	0.18
OPTICS ($\varepsilon = 0{,}42$, $\nu = 5$)	0.007	0.22	0.008	0.213

Tabla 5.23: Comparación de las ejecuciones entre todos los métodos.

En la Tabla 5.23 podemos ver un resumen de los resultados con todos los métodos que hemos ejecutado. La primera columna corresponde a los datos que ha clasificado erróneamente del total de la base de datos. La segunda, al mismo cálculo pero ahora teniendo en cuenta los que ha deja sin clasificar. La tercera a los errores que se han cometido pero del total de clasificados. Y por último, la cuarta corresponde al porcentaje de datos que ha asignado a ruido.

Podemos decir en conclusión que OPTICS tiene ventaja respecto de los demás métodos ya que se equivoca menos a la hora de clasificar. Pero tanto DBSCAN como OPTICS incorporan la noción de ruido y por tanto dejan elementos sin clasificar y no los asignan a ninguna clase, cosa que los métodos Jerárquicos que hemos ejecutado no hacen. Por tanto el porcentaje total de decisiones correctas con los métodos de DBSCAN y OPTICS es peor si tenemos en cuenta los datos que asigna a ruido y la performance global empeora.

5.1.6. Comparación del tiempo de ejecución

Como ya he hablado en la sección 4.2.2 el tiempo de ejecución de OPTICS es casi constante, 1.6 veces el tiempo de ejecución de DBSCAN. Vamos a ver si lo hace para estos datos. Realizamos 10 ejecuciones con estos datos y hacemos un promedio:

Como podemos ver en la Tabla 5.24, el tiempo de ejecución de OPTICS es similar a DBSCAN y también ganamos en calidad de clasificación. Sin embargo, Vecinos recíprocos tarda 4 veces más que DBSCAN aunque hace una clasificación que se parece más a la real. El método de clasificación condicionada aunque tarda más que DBSCAN y OPTICS, es más rápido que Vecinos recíprocos.

Tabla 5.24: Tiempo ejecución caso Iris

Algoritmo	**Tiempo (milisegundos)**
DBSCAN	24.1
OPTICS	29
VRE	86
CC	45.9

5.2. Caso 2: Guttmann

5.2.1. Contexto y presentación de los datos

Este estudio se ubica en el marco de un proyecto de colaboración entre la Dra. K.Gibert y el Institut Guttmann-Hospital de Neurorehabilitació, cuyo principal objetivo es comprender por qué pacientes con una lesión neurológica parecida registran distintas respuestas a un mismo tratamiento neurorehabilitador. Para ello se realizó un ejercicio de clustering que permitió identificar 5 perfiles de pacientes con distintos patrones de respuesta al tratamiento. Se utilizan datos provenientes de la evaluación neuropsicológica de 47 pacientes con daño cerebral adquirido de origen traumático [traumatismo craneoencefálico (TCE)] que han recibido tratamiento rehabilitador en el Institut Guttmann-Hospital de Neurorehabilitación.

Se ha evaluado el estado neuropsicológico de cada paciente al inicio del tratamiento y al final, con lo que se ha podido cuantificar la diferencia entre el estado del paciente antes y después del tratamiento, supuestamente imputable a los efectos del mismo.

La evaluación del estado del paciente se realiza a través de distintos tests que miden diferentes funciones neuropsicológicas. Para la elaboración de este informe se han contemplado 4 de estas funciones:

- Atención
- Memoria Verbal y Aprendizaje
- Funciones ejecutivas
- Lenguaje

Los tests que se han utilizado para evaluar cada uno de estas funciones son las siguientes:

- Atención
  - Subtest dígitos directos del Test Barcelona
  - TAS (Test de Atención Sostenida)
  - Stroop (Palabra-Color)
  - TMT A
- Memoria Verbal y Aprendizaje
  - Subtest dígitos inversos del Test Barcelona
  - Subtest de memoria verbal del test Barcelona
  - Subtest de aprendizaje verbal del test Barcelona

- Funciones ejecutivas
  - WCST (Wisconsin Card Sorting Test)
  - TMT B
  - Stroop (Interferencia)
- Lenguaje
  - Subtest repetición del Test Barcelona
  - Subtest denominación del Test Barcelona
  - Subtest comprensión verbal (palabras) del Test Barcelona
  - Subtest comprensión verbal (órdenes) del Test Barcelona

En el Anexo B.1 se halla una breve descripción de estos tests así cómo los nombres de las variables que en la base de datos recogen las puntuaciones de cada uno.

Finalmente se tiene información relativa a la gravedad del TCE que ha sufrido el paciente, lo que a posteriori permitirá relacionar el perfil de recuperación con las características de la lesión. La escala de Coma de Glasgow (GCS, Glasgow Coma Scale [Glasgow 2001]) se usa para medir el nivel de consciencia de un paciente con TCE y en consecuencia soporta la severidad de la lesión. Tiene en cuenta la apertura ocular, la respuesta verbal y la motora. Una persona en estado normal de consciencia presentaría una puntuación de 15 en esta escala. La mínima puntuación es 3 y se interpreta según el siguiente baremo:

- TCE Leve .... 13-15 puntos
- TCE Moderado.. 9-12 puntos
- TCE Grave.... 8 Puntos o menos

En este caso no tenemos como referencia ninguna variable de clase sino que ésta es la incognita del problema, identificar grupos de pacientes que presentan un patrón de respuesta al tratamiento. Por tanto clasificamos a ciegas. En el anexo B.4 tenemos una descriptiva univariante de las variables.

5.2.2. Ejecución con Clasificación condicionada

Ahora vamos a ver que pasa si intentamos clasificar este conjunto de datos con un método jerárquico, con el método de Ward y con métrica Euclídea. El resultado es el de la Figura 5.18.

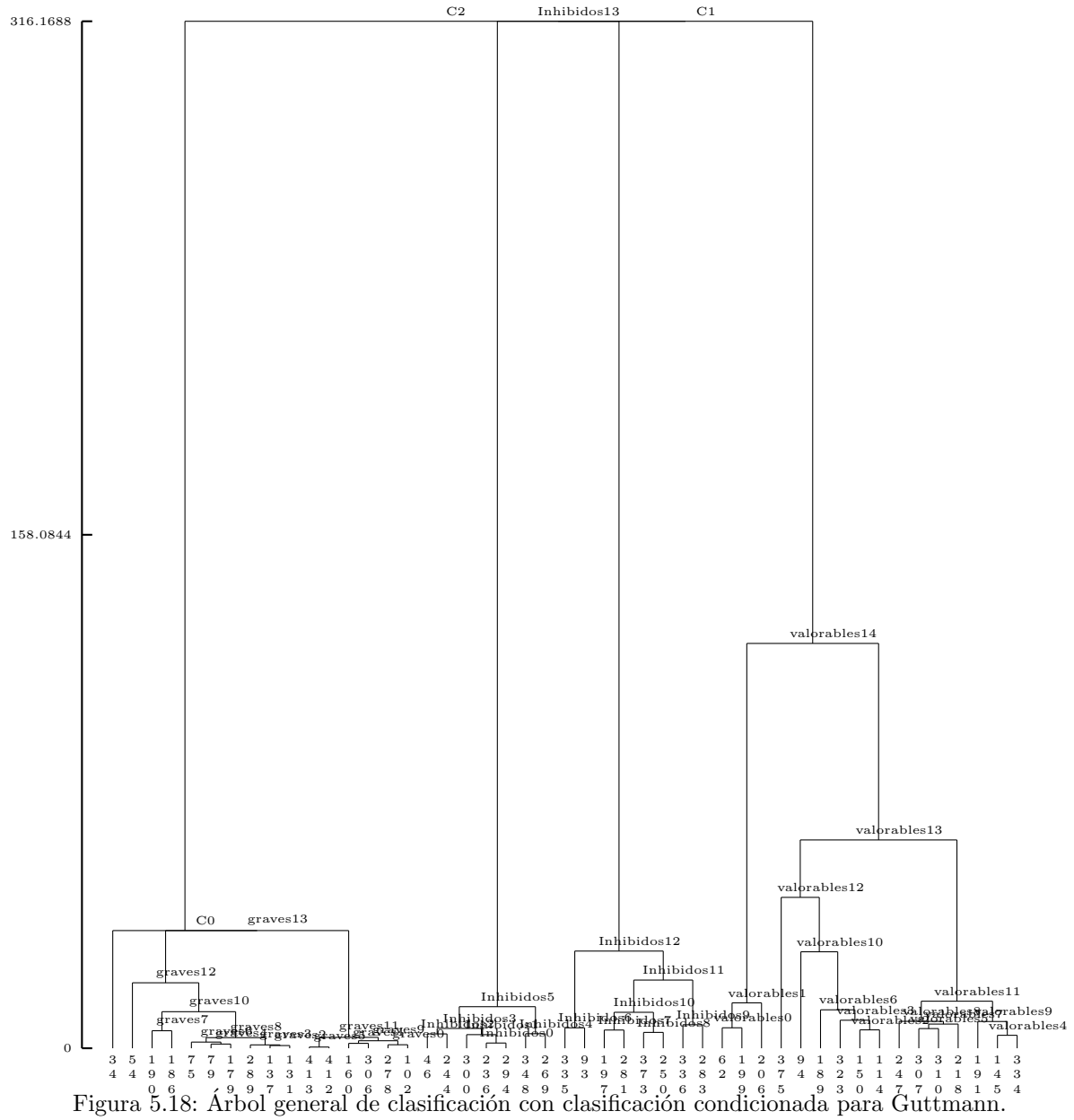


Figura 5.18: Árbol general de clasificación con clasificación condicionada para Guttmann.

El resultado del dendrograma de la Figura 5.18 sugiere un corte en 5 clases. En el anexo B.5 se hallan las estadísticas básicas por clase.

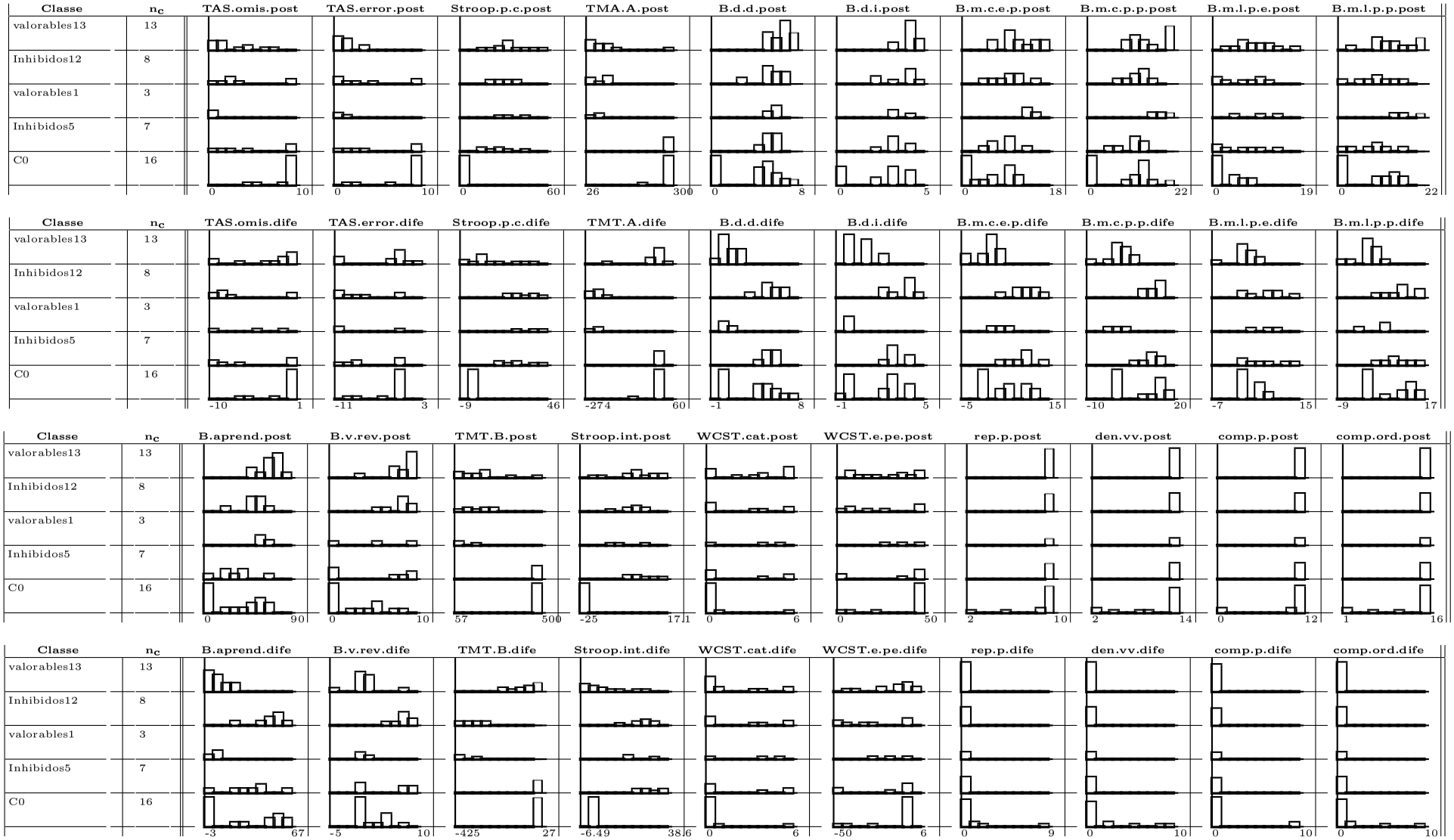


Figura 5.19: Descriptiva de clases de la ejecución Clasificación condicionada para Guttmann.

Vemos también la descriptiva de clases (Figura 5.19):

En la Figura 5.19 se muestra un panel de clases dónde las medidas después del tratamiento y las diferencias antes y después del tratamiento se puede analizar en vertical.

Vemos cómo por ejemplo este método ha asignado a la clase *valorables13* pacientes que al inicio del tratamiento no conseguían pasar los test y al final sí. En la clase *Inhibidos12* hay pacientes que al inicio pasaban casi todos los test y al final acababan pasándolos todos. En la clase *valorables1* introduce pacientes que pasaban los test de *Atención* y *Funciones ejecutivas* pero de *Memoria* no, sin embargo, al final del tratamiento los pasaba todos. En *Inhibidos5* hay pacientes que hacían el test de Atención de Stroop pero no el de TMT A y al final pasan todos menos el de Stroop de Funciones ejecutivas. Y por último en la clase *C0* hay pacientes con una situación muy grave que no pasan ningún test al inicio ni al final.

En las Figuras 5.20 y 5.21 se encuentra la descriptiva 3D de las variables que más discriminan a la hora de hacer la clasificación.

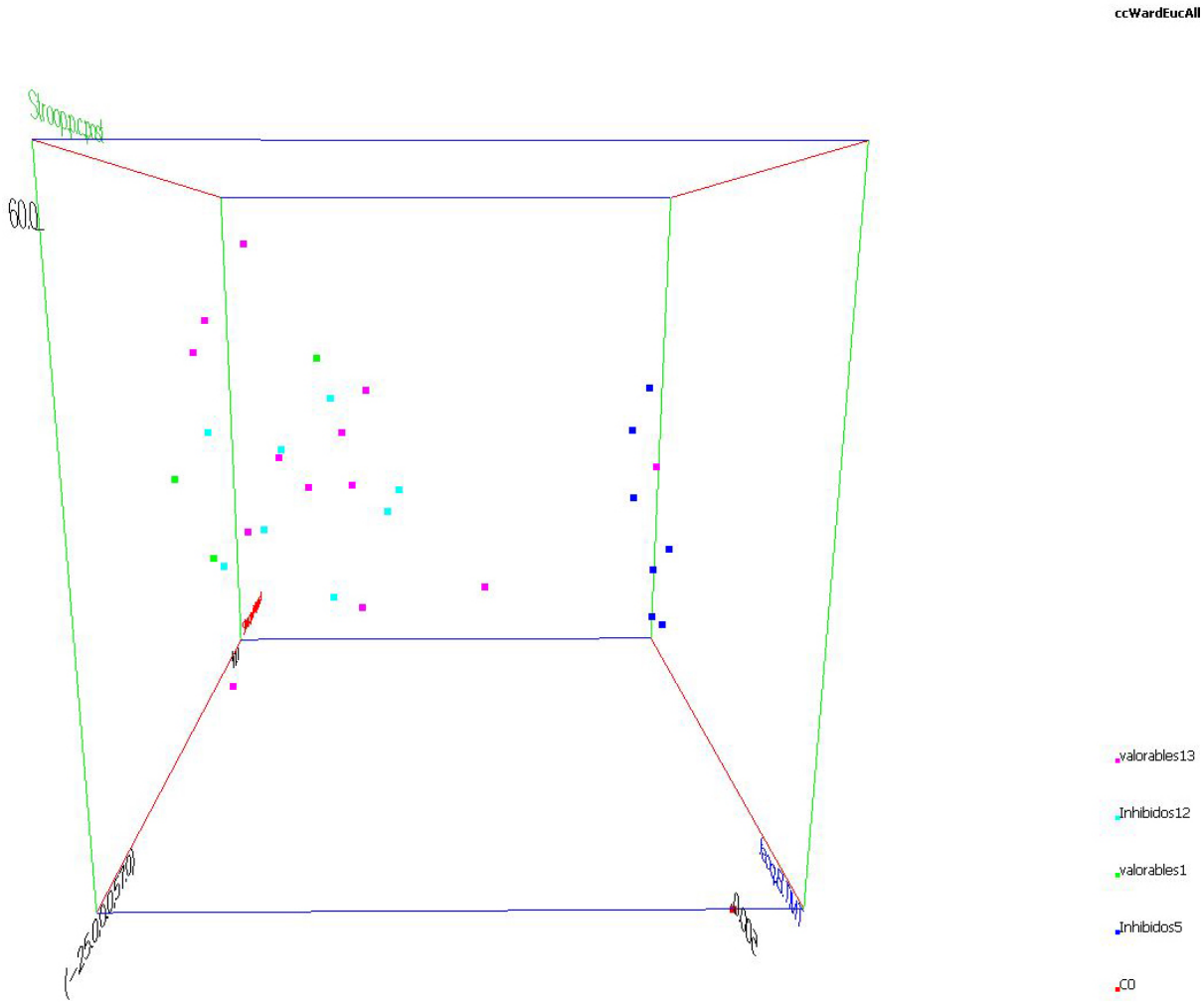


Figura 5.20: Visualización 3D ejecución CC de variables Post.

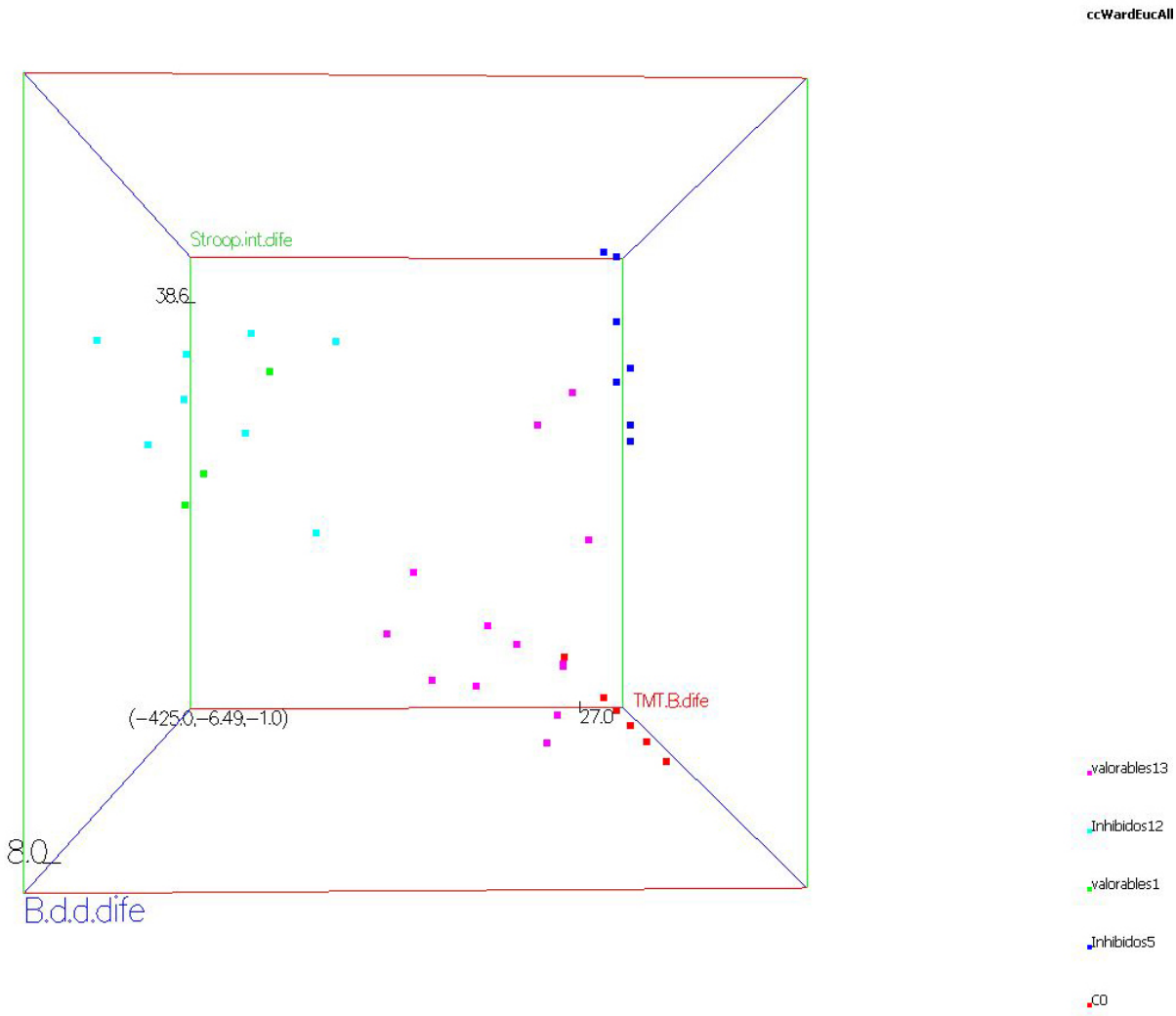


Figura 5.21: Visualización 3D ejecución CC de variables Dife.

5.2.3. Ejecución con DBSCAN

Ejecutaremos el algoritmo DBSCAN con los datos mencionados utilizando la métrica Euclídea.

Elegimos $\nu = 2$ ya que la matriz de datos no es muy grande. Representamos el gráfico 2-dist (Figura: 5.22) para orientarnos acerca de los valores de ε.

Vemos que hay un valle en 82 aproximadamente, así que elegimos $\varepsilon = 82$. Vamos al menú y accedemos al *Panel Clasifica* y desde allí elegimos el método DBSCAN con parámetros $\varepsilon = 82$ y $\nu = 2$.

El algoritmo identifica 6 clases, de las cuáles una es NOISE (ruido) conteniendo el 6 % de los datos.

En el anexo B.6 se hallan las estadísticas básicas por clase.

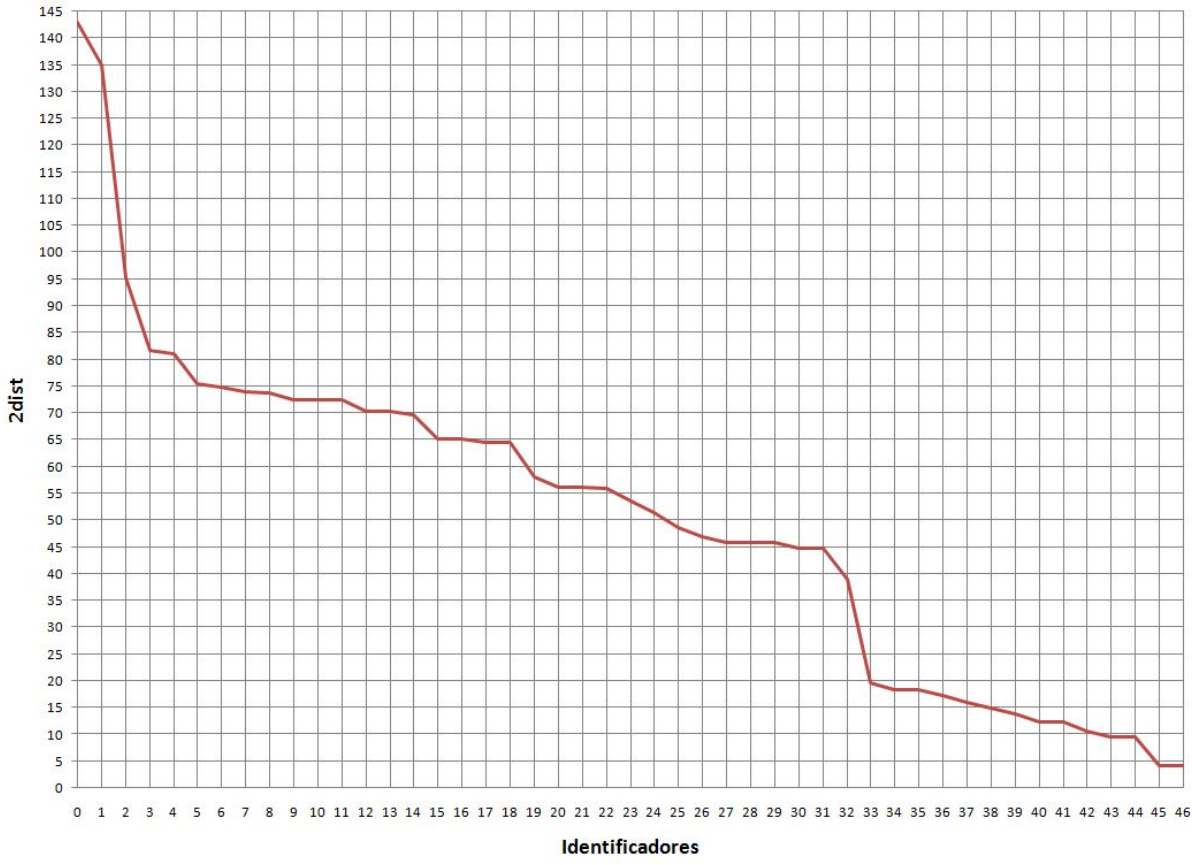


Figura 5.22: Gráfica 2-dist para Gutmmann.

Figura 5.23: Descriptiva de clases de la ejecución DBSCAN $\varepsilon = 82$ $\nu = 2$ para Guttmann

En la Figura 5.23 vemos cómo en la clase (C0) ha introducido pacientes que no hacían las pruebas de Atención y Funciones ejecutivas ni al inicio ni al final del tratamiento. En cuanto a Memoria algunos de esos pacientes perdían capacidad para realizar las pruebas.

En las clases (C1 y C4) ha introducido pacientes que al final hacen casi todas las pruebas pero al inicio C4 no las hacía y C1 hacía las de Atención, Memoria y las Funciones ejecutivas TMT B y Stroop.

Los de la clase (C3) hacen las pruebas de Atención TAS y Stroop y las Funciones ejecutivas WCST al inicio y al final de manera moderada. En general no mejoran mucho por que ya logran pasar algunas pruebas.

Los de la clase (C2) hacen las pruebas de Atención Stroop de manera moderada aunque en general mejoran. Y las Funciones ejecutivas las hacen mal al inicio y bien al final del tratamiento.

Comparamos estos resultados con los obtenidos con el método anterior:

CC \ DBSCAN	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**	**útils**	**mancants**
C0	15	0	0	0	0	1	16	0
Inhibidos5	7	0	0	0	0	0	7	0
valorables1	0	3	0	0	0	0	3	0
Inhibidos12	0	6	2	0	0	0	8	0
valorables13	1	0	0	3	7	2	13	0
útils	23	9	2	3	7	3	47	
mancants	0	0	0	0	0	0		0

Tabla 5.25: Tabla de contingencia CC/DBSCAN ($\varepsilon = 82$ y $\nu = 2$) para Guttmann.

En la Tabla 5.25 vemos cómo hay un conjunto de datos mayoritario (C0) que correspondería a los pacientes en estado grave. Del resto de clases no podemos decir gran cosa, esta clasificación de los datos no se parece a la qué hace el método Jerárquico.

En las Figuras 5.24 y 5.25 se encuentra la descriptiva 3D de las variables que aparentemente más discriminan a la hora de hacer la clasificación.

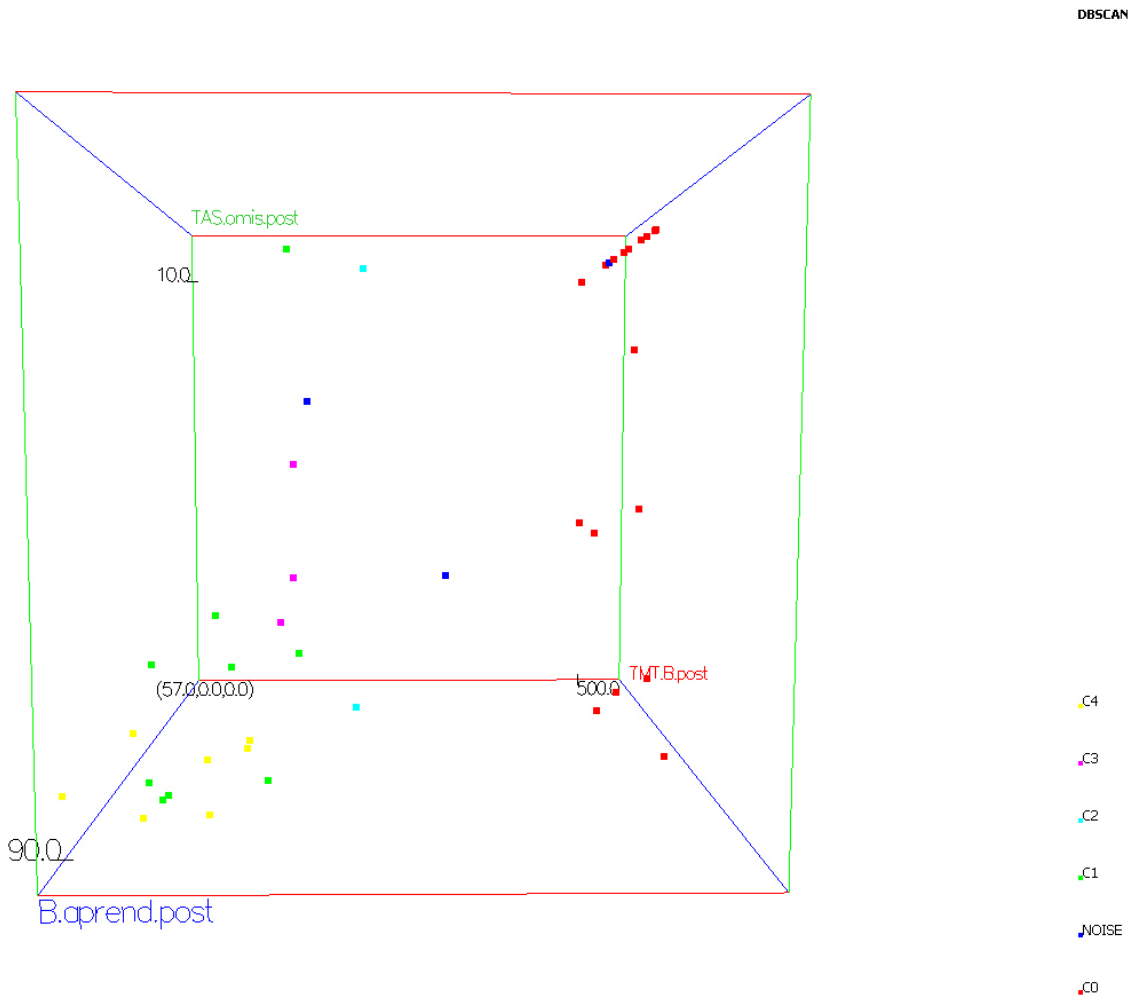


Figura 5.24: Visualización 3D ejecución DBSCAN de variables Post.

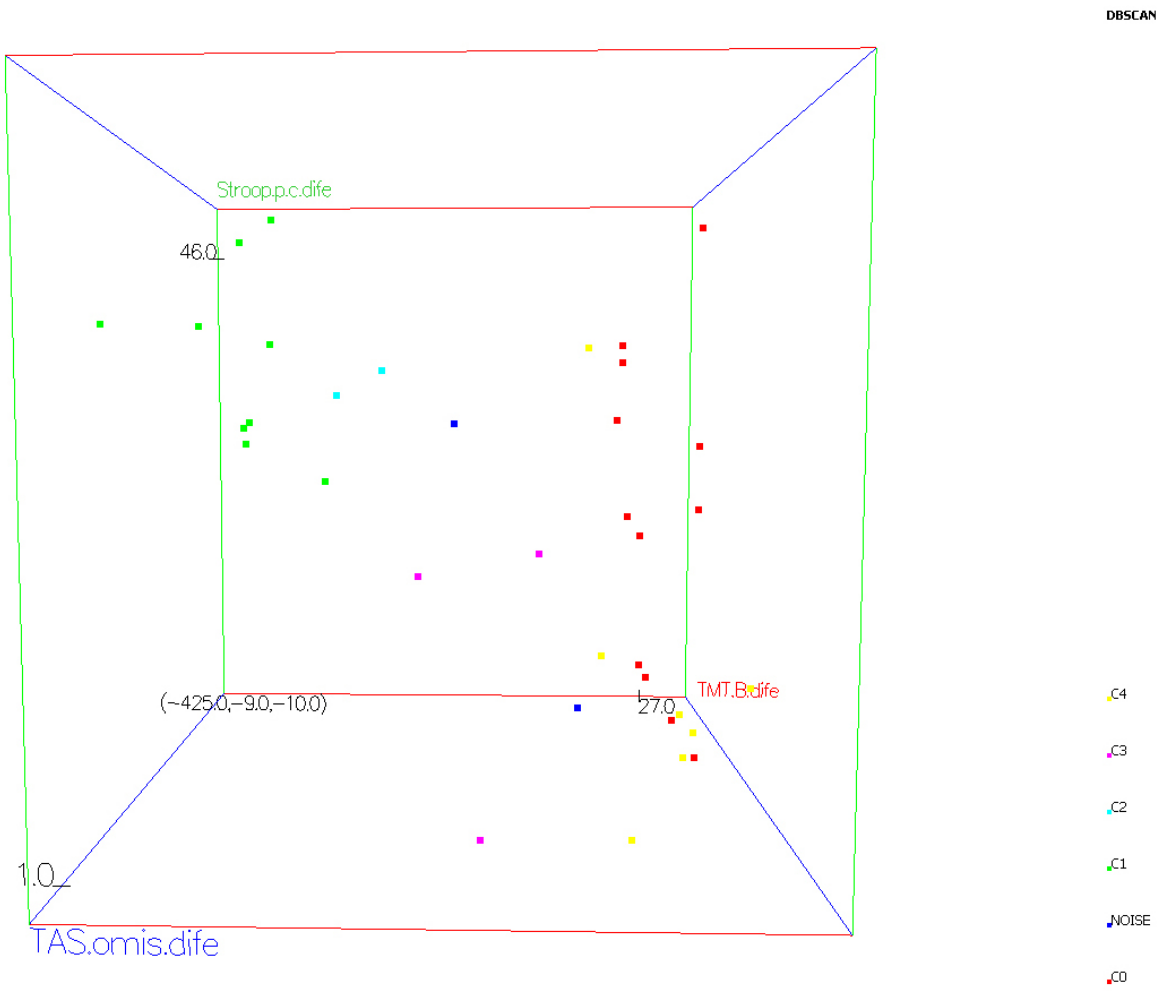


Figura 5.25: Visualización 3D ejecución DBSCAN de variables Dife.

5.2.4. Ejecución con OPTICS

Vamos a ver qué pasa si para los mismos valores de ε ejecutamos OPTICS (Figura: 5.26).

Clustering OPTICS

Radi màxim: 82
Cluster mínim: 2
Eix y: Automàtic
Mètrica: Euclídea *(Paràmetres:* No normalitzar; Distàncies no ponderades; Distàncies no al quadrat)

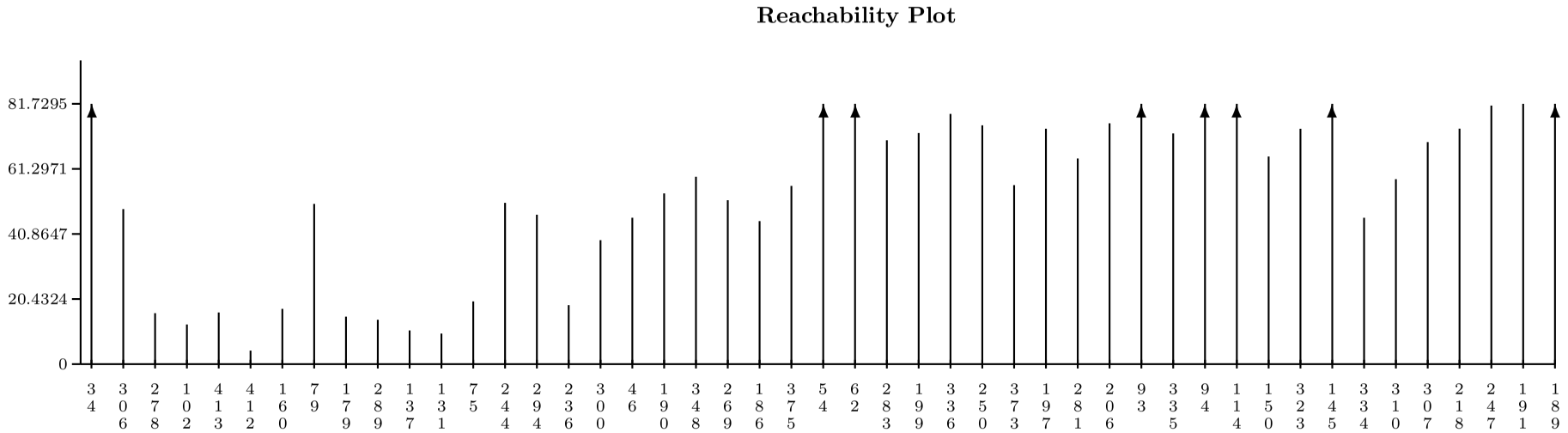


Figura 5.26: Reachability Plot para $\varepsilon = 82$ y $\nu = 2$ con los datos de Guttmann.

Clustering OPTICS

Radi màxim: 82
Cluster mínim: 2
Eix y: Automàtic
Mètrica: Euclídea *(Paràmetres:* No normalitzar; Distàncies no ponderades; Distàncies no al quadrat)

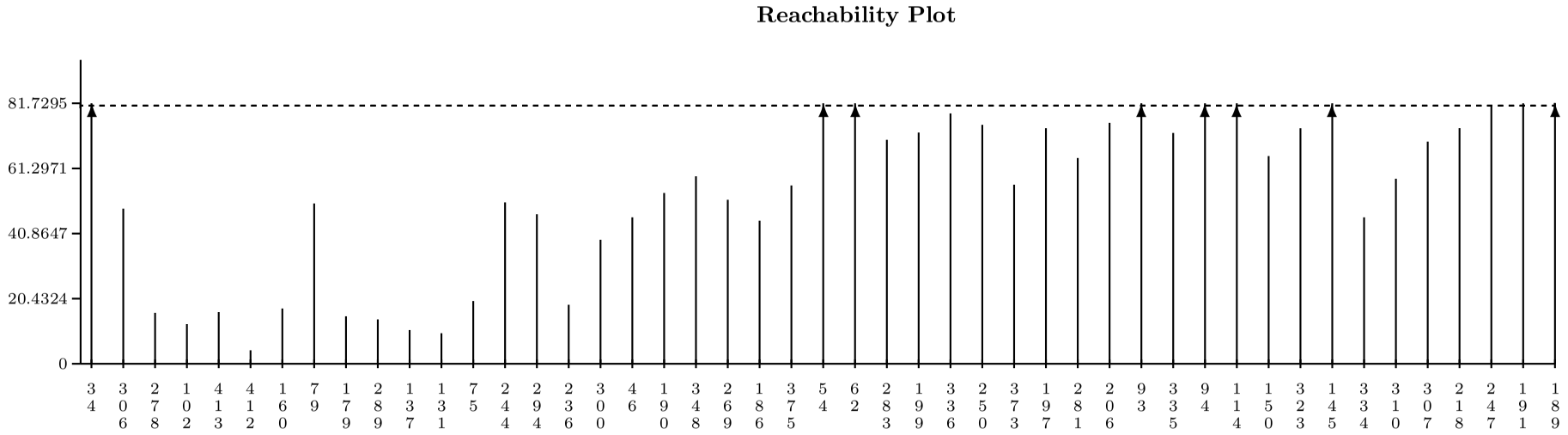


Figura 5.27: Reachability Plot cortado en $\varepsilon' = 81$ para $\varepsilon = 82$ y $\nu = 2$ con los datos de Guttmann.

Cortamos en 81 porque nos hace pensar que obtendremos un mayor número de clases. La Figura 5.27 muestra el corte realizado.

En el anexo B.7 se hallan las estadísticas básicas por clase.

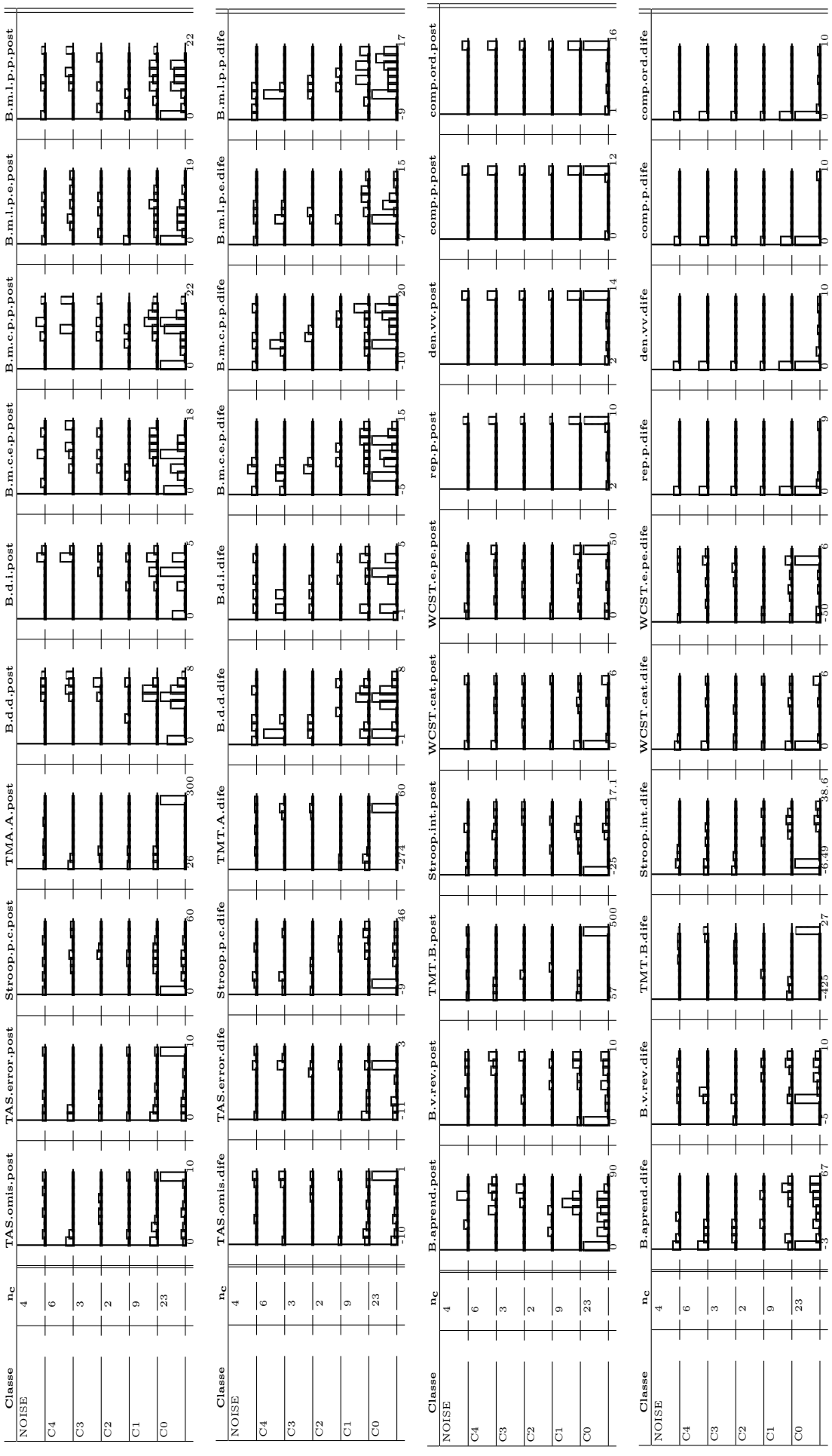


Figura 5.28: Descriptiva de clases de la ejecución OPTICS $\varepsilon = 82$ $\nu = 2$

En la Figura 5.28 vemos cómo el algoritmo introduce en C0 los pacientes resistentes al tratamiento.

En la clase (C1) pacientes que mejoran un poco la Memoria y la Atención. En cuanto a las Funciones ejecutivas las hacen más o menos bien al inicio y al final del tratamiento, aunque las de WCST no las hacen.

Y por último en la clase (C2) incluye pacientes que mejoran mucho la Atención y la Memoria. Y podemos decir que las Funciones ejecutivas las hacen mejor al final que al inicio del tratamiento.

Comparamos con los dos casos anteriores:

CC \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C0	15	0	0	1	16	0
Inhibidos5	7	0	0	0	7	0
valorables1	0	0	0	3	3	0
Inhibidos12	0	2	0	6	8	0
valorables13	1	0	3	9	13	0
útils	23	2	3	19	47	
mancants	0	0	0	0		0

Tabla 5.26: Tabla de contingencia CC/OPTICS ($\varepsilon = 82$ $\nu = 2$) para Guttmann.

En la 5.26 vemos cómo en el caso anterior hay un conjunto de datos mayoritario (C0) que correspondería a los pacientes en estado grave. Del resto de clases no podemos decir gran cosa, esta clasificación de los datos tampoco se parece a la qué hace el método Jerárquico.

DBSCAN \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C0	23	0	0	0	23	0
C1	0	2	0	7	9	0
C2	0	0	0	2	2	0
C3	0	0	0	3	3	0
C4	0	0	3	4	7	0
NOISE	0	0	0	3	3	0
útils	23	2	3	19	47	
mancants	0	0	0	0		0

Tabla 5.27: Tabla de contingencia DBSCAN/OPTICS ambos con ($\varepsilon = 82$ $\nu = 2$) para Guttmann.

Como vemos en la Tabla 5.27 en este caso tanto con DBSCAN como con OPTICS obtenemos la misma clasificación, aunque hay elementos que OPTICS no los clasifica y los introduce en el conjunto ruido.

Hay que decir que estos dos métodos son muy sensibles a los parámetros de entrada y por eso de su elección depende el resultado de la clasificación.

En las Figuras 5.29 y 5.30 se encuentra la descriptiva 3D de las variables que más discriminan a la hora de hacer la clasificación.

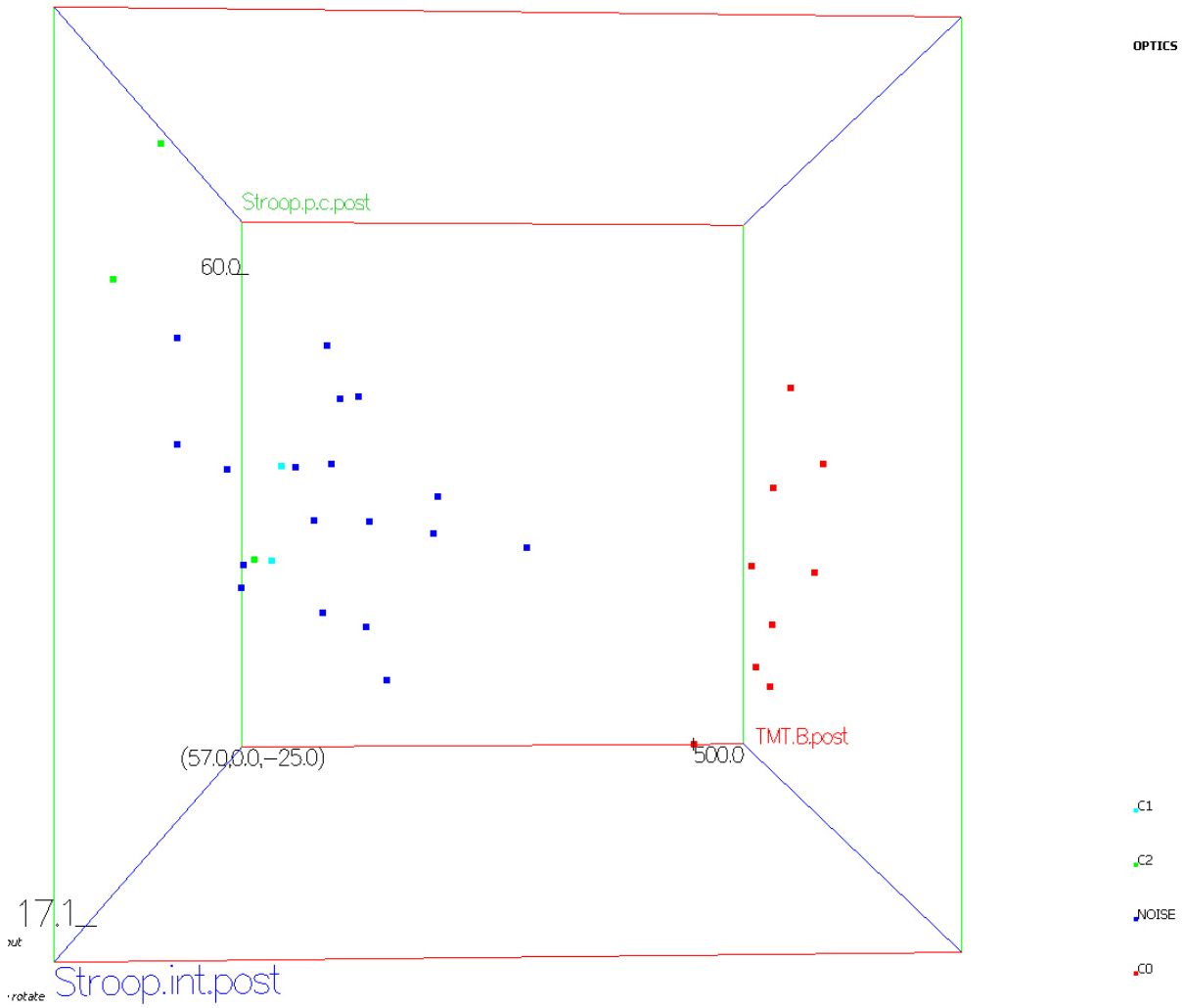


Figura 5.29: Visualización 3D ejecución OPTICS de variables Post.

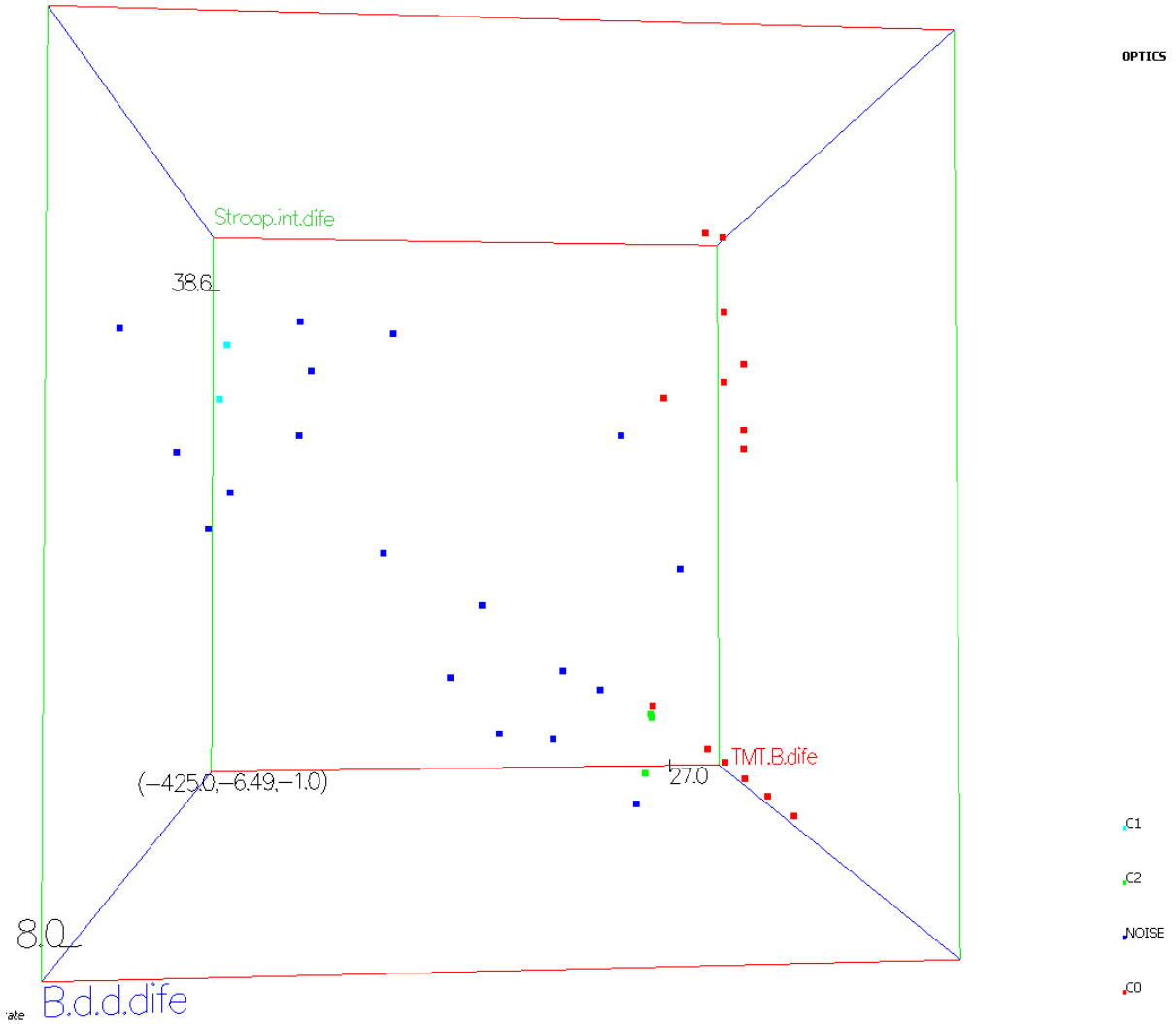


Figura 5.30: Visualización 3D ejecución OPTICS de variables Dife.

5.2.5. Comparación del tiempo de ejecución

Realizamos 10 ejecuciones con estos datos y hacemos un promedio:

Tabla 5.28: Tiempo ejecución caso Gutmman

Algoritmo	**Tiempo (milisegundos)**
DBSCAN	29.1
OPTICS	32.7
CC	28

Vemos cómo con los datos anteriores DBSCAN y OPTICS en promedio se comportan de manera similar y el método de clasificación condicionada con estos datos donde el número de variables es mayor que en el caso anterior pero la cantidad de datos es menor, se comporta de mejor manera y además clasifica los datos en 5 clases haciendo una buena clasificación de los pacientes.

5.3. Ejecución con métrica mixta

Aunque en el paper [Gibert & Sonicki 2002] habla de que discretizar variables numéricas a priori para convertirlas en categóricas con datos reales es una mala práctica, a modo académico vamos a discretizar algunas variables para poder obtener otras que sean categóricas y así poder usar estos métodos de clustering con métricas mixtas, ilustrando la capacidad de la implementación propuesta para DBSCAN y OPTICS de poder trabajar con métricas de datos heterogéneos.

5.3.1. Ejecución con DBSCAN

Elegimos $\nu = 4$ porque cómo hemos dicho anteriormente hay pocos datos. Representamos el gráfico 4-dist (Figura: 5.31) para orientarnos acerca de los valores de ε.

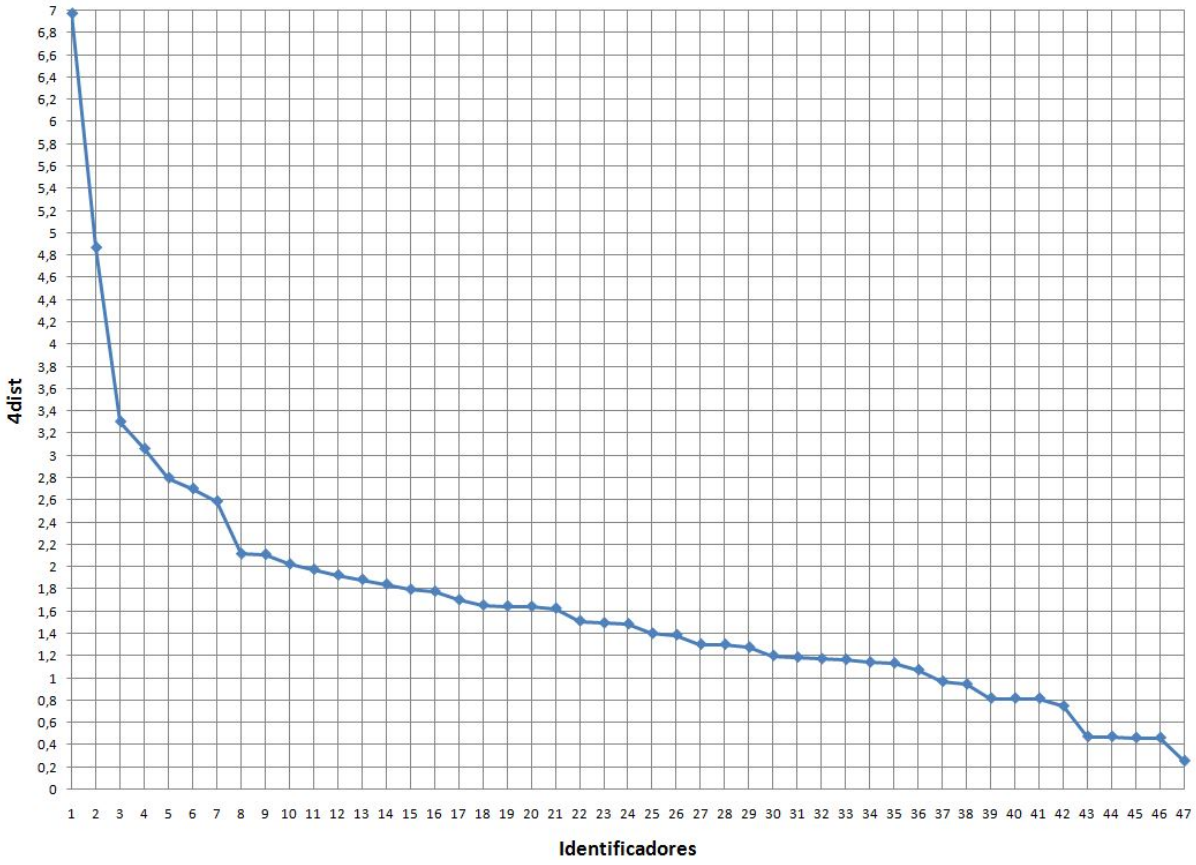


Figura 5.31: Gráfica 4-dist para Gutmmann con datos Mixtos.

Elegimos uno de los valles más claros $\varepsilon = 1{,}5$. En el anexo B.8 se hallan las estadísticas básicas por clase.

El algoritmo en este caso obtiene 4 clases, de las cuales una es NOISE con el 34 % de los datos.

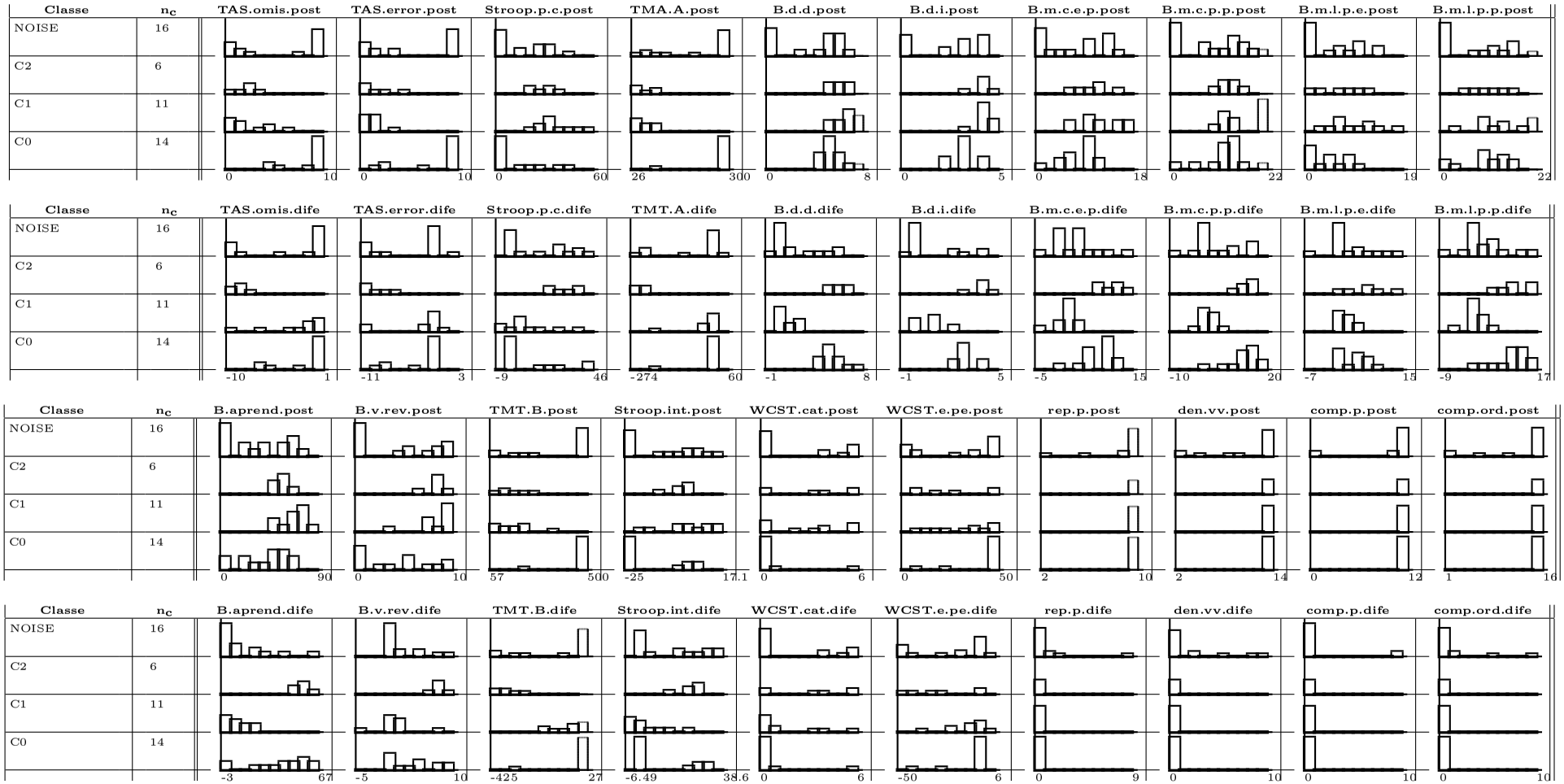


Figura 5.32: Descriptiva de clases de la ejecución DBSCAN ($\varepsilon = 1{,}5$ y $\nu = 4$) condicionada para Guttmann con dato Mixtos.

En la Figura 5.32, vemos cómo DBSCAN, igual que en los casos anteriores, introduce en C0 los pacientes en estado grave que son resistentes al tratamiento. En C1 pacientes que han mejorado con el tratamiento y en C2 pacientes que ya hacían algunas de las pruebas.

Comparamos estos resultados con los obtenidos con el método anterior:

CC \ DBSCAN	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C0	9	0	0	7	16	0
Inhibidos5	4	0	0	3	7	0
valorables1	0	0	0	3	3	0
Inhibidos12	1	0	6	1	8	0
valorables13	0	11	0	2	13	0
útils	14	11	6	16	47	
mancants	0	0	0	0		0

Tabla 5.29: Tabla de contingencia CC/DBSCAN ($\varepsilon = 1{,}5$ y $\nu = 4$) para Guttmann con datos Mixtos.

En la Tabla 5.29 vemos cómo en este caso la clasificación que hace DBSCAN introduce en C0 las clases que el método Jerárquico identifica como C0, Inhibidos5 y algún elemento de Inhibidos12. En C1 introduce pacientes de la clase valorables13 del método Jerárquico y en C2 los de la clase Inhibidos12. En este caso la clasificación que hace OPTICS se parece un poco más a la que hace el método Jerárquico pero introduce varios datos (34 %) en el conjunto NOISE y obtiene 3 clases en lugar de 5.

En las Figuras 5.33 y 5.34 se encuentra la descriptiva 3D de las variables que más discriminan.

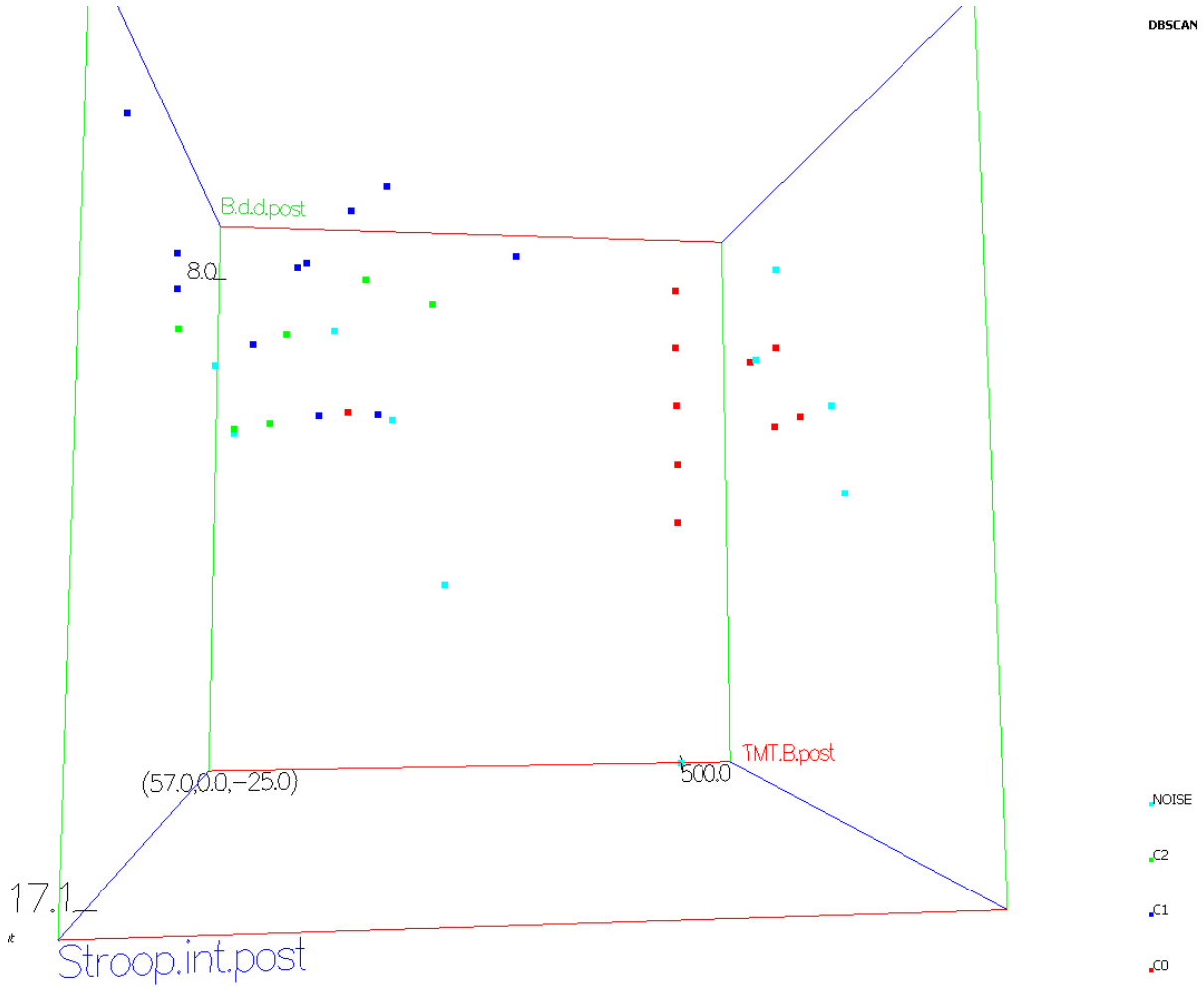


Figura 5.33: Visualización 3D ejecución DBSCAN con datos mixtos de variables Post.

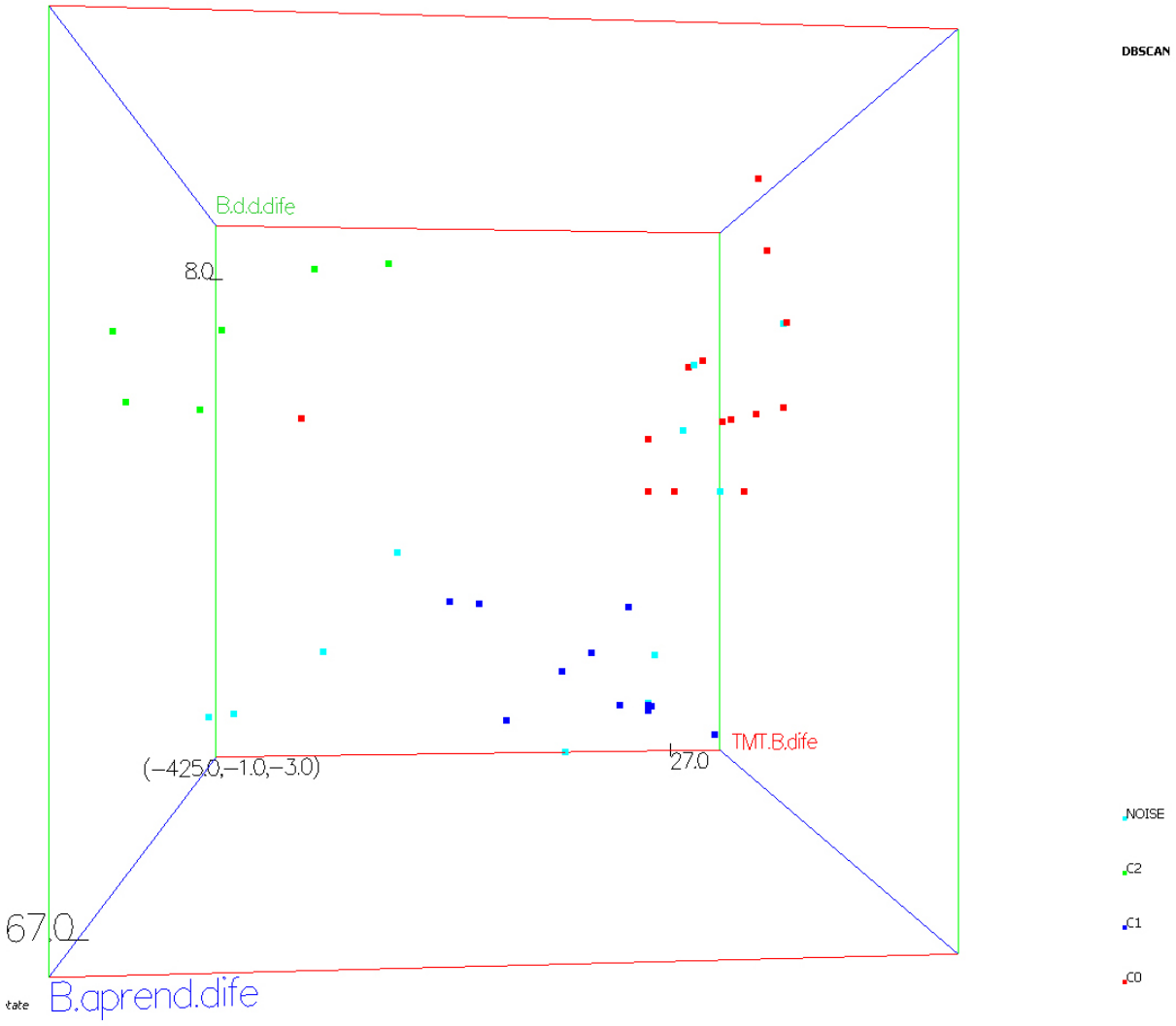


Figura 5.34: Visualización 3D ejecución DBSCAN con datos mixtos de variables Dife.

5.3.2. Ejecución con OPTICS

Vamos a ver que pasa si para los mismos valores de ε ejecutamos OPTICS (Figura: 5.35).

Cortamos en 1.49 para obtener el mayor número de clases posible. La Figura 5.36 muestra el corte realizado.

Clustering OPTICS

Radi màxim: 1.5
Cluster mínim: 4
Eix y: Automàtic
Mètrica: Mixta Gibert *(Paràmetres:* α: 0.04931097; β: 0.950689; Distàncies no ponderades)

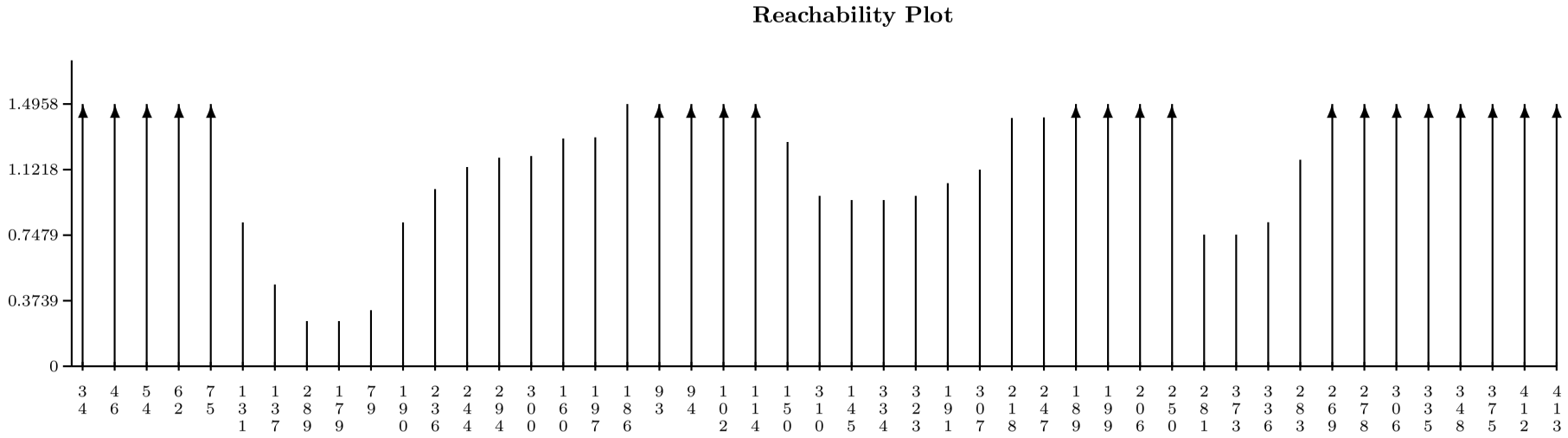


Figura 5.35: Reachability Plot para $\varepsilon = 1{,}5$ y $\nu = 4$ con los datos de Guttmann para datos Mixtos.

Clustering OPTICS

Radi màxim: 1.5
Cluster mínim: 4
Eix y: Automàtic
Mètrica: Mixta Gibert *(Paràmetres:* α: 0.04931097; β: 0.950689; Distàncies no ponderades)

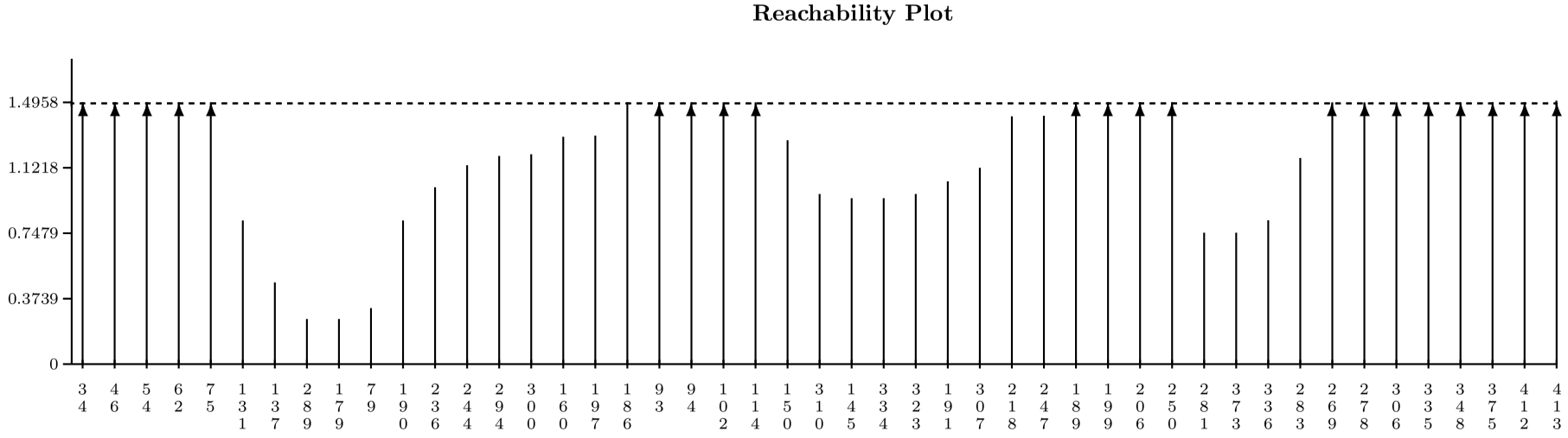


Figura 5.36: Reachability Plot cortado en $\varepsilon' = 1{,}49$ para $\varepsilon = 1{,}5$ y $\nu = 4$ con los datos de Guttmann para datos Mixtos.

El algoritmo al igual que DBSCAN, obtiene 4 clases, de las cuales una es NOISE con el 38 % de los datos.

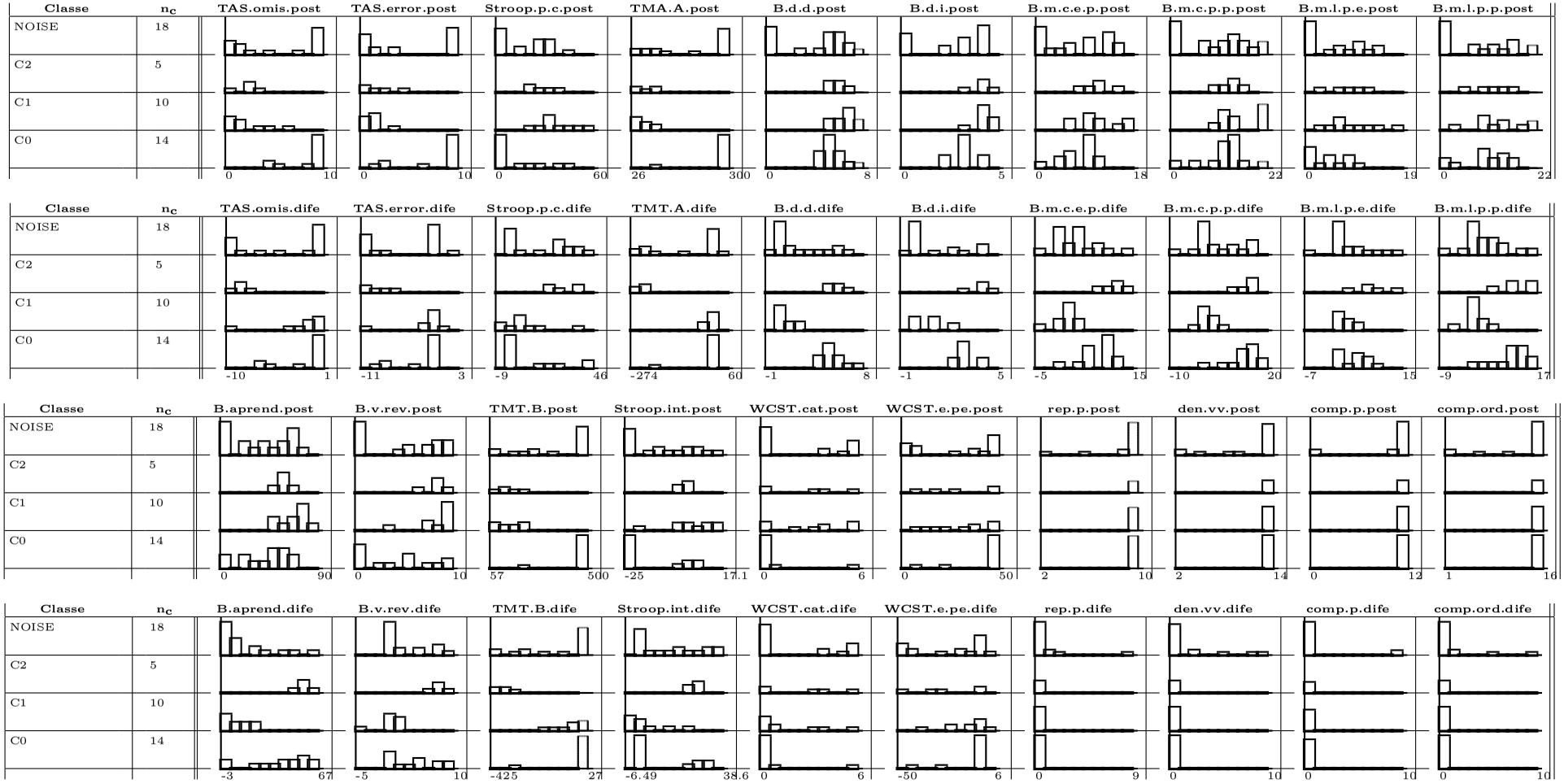


Figura 5.37: Descriptiva de clases de la ejecución OPTICS ($\varepsilon = 1{,}5$ y $\nu = 4$) condicionada para Guttmann con dato Mixtos.

En la Figura 5.37 vemos cómo OPTICS también introduce en C0 los pacientes en estado grave que son resistentes al tratamiento. En C1 pacientes que han mejorado con el tratamiento y en C2 pacientes que ya hacían algunas de las pruebas. Es decir hace una clasificación parecida a la qué hace DBSCAN.

En el anexo B.9 se hallan las estadísticas básicas por clase.

Comparamos estos resultados con los obtenidos con los métodos anteriores:

CC \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C0	9	0	0	7	16	0
Inhibidos5	4	0	0	3	7	0
valorables1	0	0	0	3	3	0
Inhibidos12	1	0	5	2	8	0
valorables13	0	10	0	3	13	0
útils	14	10	5	18	47	
mancants	0	0	0	0		0

Tabla 5.30: Tabla de contingencia CC/OPTICS ($\varepsilon = 1{,}5$ y $\nu = 4$) para Guttmann con datos Mixtos.

En la Tabla 5.30 vemos cómo OPTICS hace el mismo tipo de clasificación que hace DBSCAN si lo comparamos con el método Jerárquico.

DBSCAN \ OPTICS	**C0**	**C1**	**C2**	**NOISE**	**útils**	**mancants**
C0	14	0	0	0	14	0
C1	0	10	0	1	11	0
C2	0	0	5	1	6	0
NOISE	0	0	0	16	16	0
útils	14	10	5	18	47	
mancants	0	0	0	0		0

Tabla 5.31: Tabla de contingencia DBSCAN/OPTICS ($\varepsilon = 1{,}5$ y $\nu = 4$) para Guttmann con datos Mixtos.

En la Tabla 5.31 podemos comprobar como ambos algoritmos coinciden a la hora de clasificar. En este caso OPTICS introduce dos elementos más en el conjunto NOISE y por tanto obtenemos más elementos sin clasificar (38 %).

En las Figuras 5.38 y 5.39 se encuentra la descriptiva 3D de las variables que más discriminan.

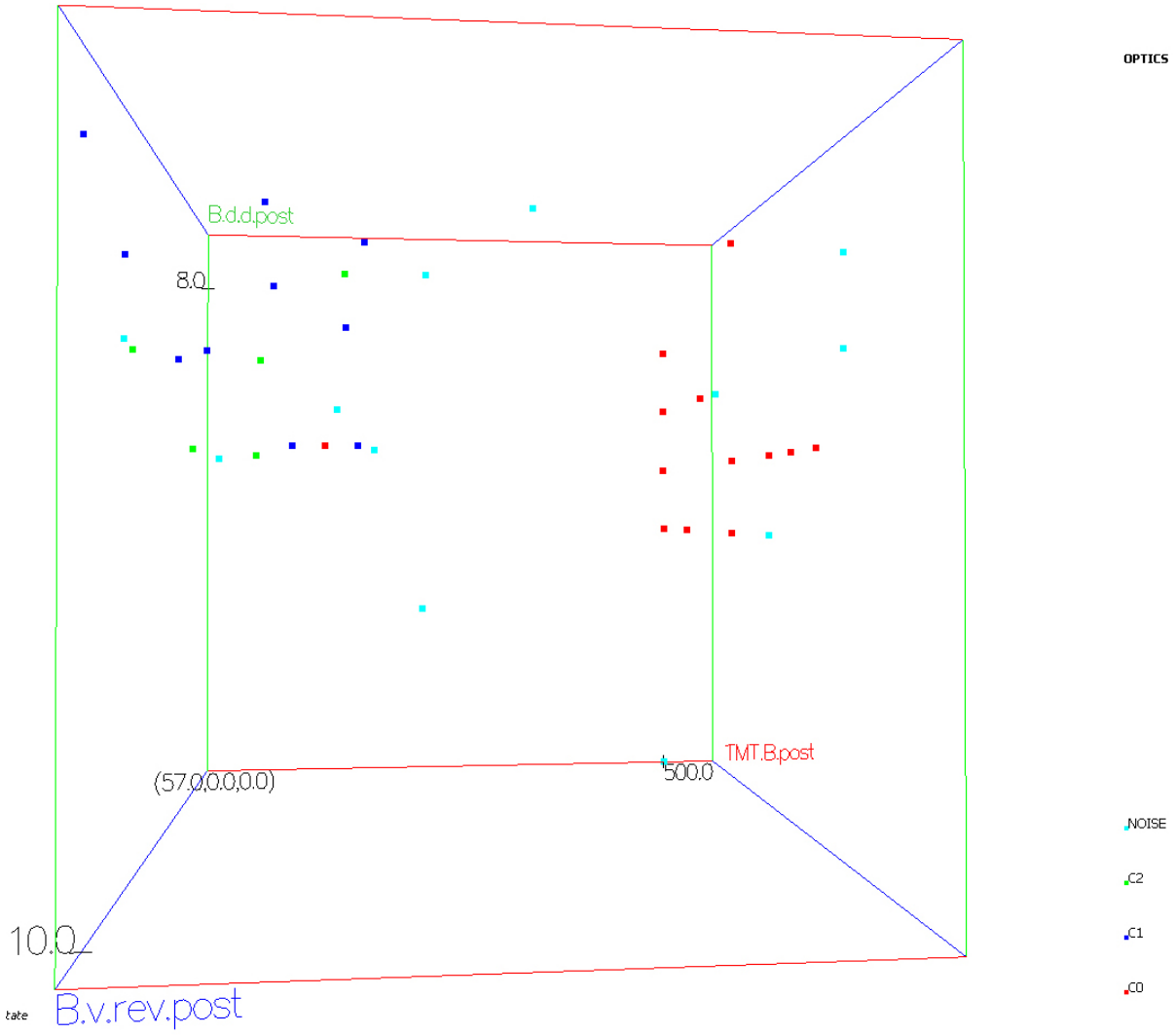


Figura 5.38: Visualización 3D ejecución OPTICS con datos mixtos de variables Post.

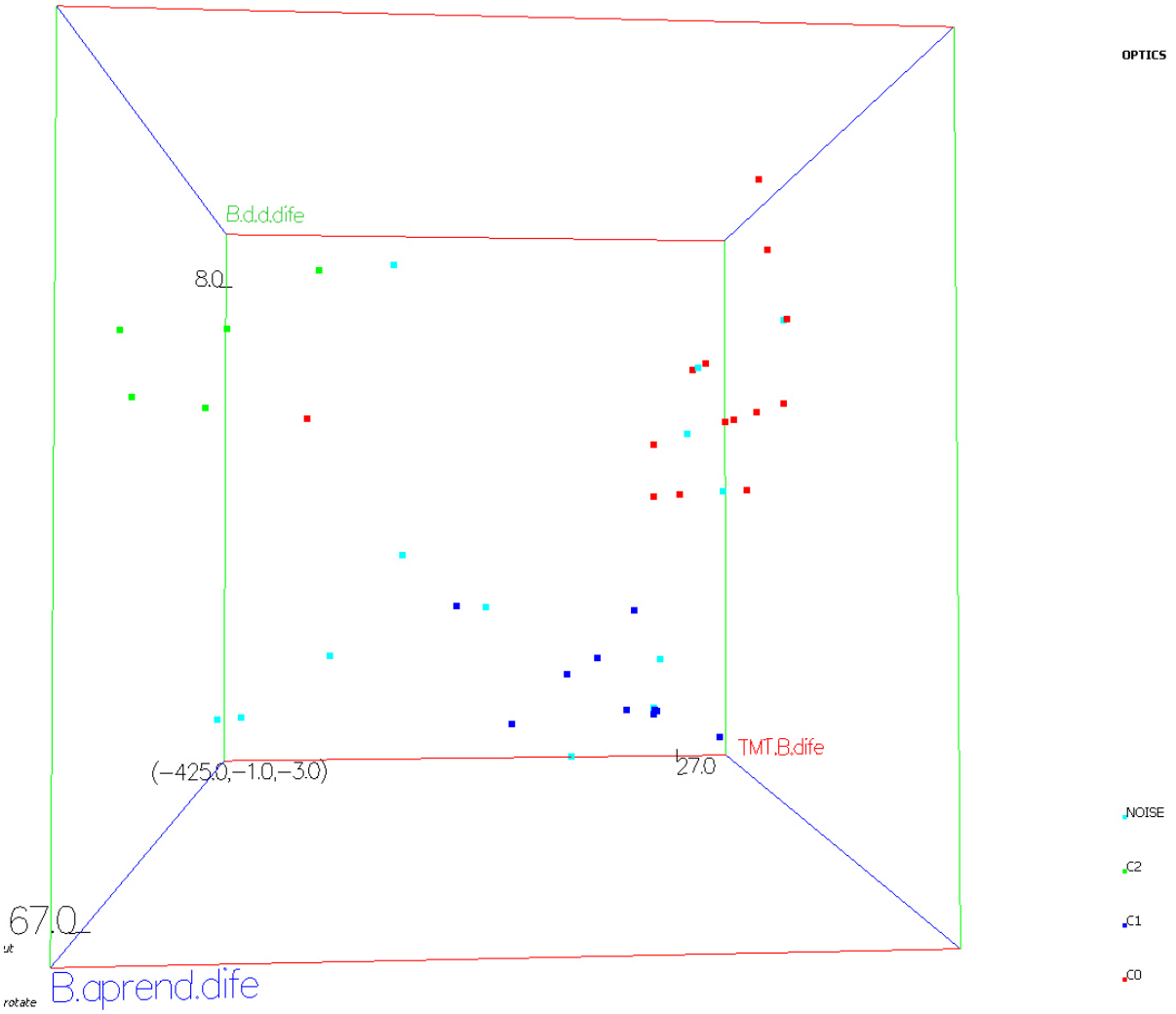


Figura 5.39: Visualización 3D ejecución OPTICS con datos mixtos de variables Dife.

Capítulo 6

Conclusiones

En este proyecto se ha profundizado en la clasificación con métodos de clustering basados en densidades y en el conocimiento del software **KLASS**. Se han diseñado e implementado dos métodos de esta familia (DBSCAN y OPTICS) que incorporan una parametrización de la métrica que permite utilizar todas las que ya están implementadas en **KLASS** (entre ellas la mixta). Para ello además de implementar estos dos algoritmos, hemos intervenido en el software para:

- Crear de pantallas para la elección de los métodos y parámetros.
- Dar la posibilidad de generar dos ficheros que exportan las clasificaciones realizadas (*.par* y *.cls*) para su posterior recuperación.
- Dar la posibilidad de añadir esa clasificación a la matriz de datos directamente.
- En el caso de OPTICS, crear de pantallas para elegir las opciones de visualización del Reachability-Plot y la Tabla de accesibilidad.
- Crear las funciones necesarias para generar, guardar y poder recuperar el Reachability-Plot a partir de la ejecución de OPTICS.
- Añadir una nueva funcionalidad para cortar el Reachability-Plot a la altura deseada por el usuario y la dar la posibilidad de elegir como se quiere ver el corte y generar la clasificación resultante.
- Añadir una nueva funcionalidad para poder hacer visualizaciones 3D, dando la posibilidad de elegir 3 variables numéricas a representar sobre los ejes X, Y, Z y una variable categórica para identificar la variable de clase. También se da la posibilidad de que el usuario rote y aleje/acerque la visualización como desee.

Se han aplicado estos nuevos métodos en 2 casos de estudio, uno con datos académicos y otro con datos reales. Además en el caso del estudio con datos reales se ha hecho una modificación de los datos para generar otro caso y poder aplicar los métodos usando métricas mixtas. Los resultados de estos experimentos demuestran que DBSCAN y OPTICS hacen una peor clasificación de los datos si tenemos en cuenta los que se dejan sin clasificar, al menos con los valores de ε y ν que hemos probado, pero se equivoca menos en los que clasifica. Además, los experimentos han demostrado que tanto DBSCAN como OPTICS superan en términos de eficiencia cuando los datos empiezan a crecer.

El problema que tanto DBSCAN como OPTICS tienen es la determinación de los parámetros de entrada. Es responsabilidad del usuario esta elección y esto influye en el resultado de los experimentos. Como ya hemos hablado en el capítulo 5, los algoritmos implementados aportan la noción de RUIDO que puede que en ocasiones no nos interese. Esta noción de ruido hace que los algoritmos se equivoquen menos al asignar un elemento a un clase, pero dejan más elementos sin clasificar y por tanto no nos aportan información para ciertos objetos. Hay ocasiones en las que según los datos que estemos manejando, sí nos interese dejar elementos sin clasificar. Por ejemplo, a la hora de dar un tratamiento a un paciente que no sabemos qué tipo de enfermedad tiene y darle un tratamiento erróneo puede hacer empeorar el estado de éste. Por tanto, recomendaríamos el uso de estos métodos semideterministas cuando fuera preferible no actuar a actuar erróneamente.

En cuanto a la ventaja que tiene OPTICS frente a DBSCAN, es que calcula una ordenación de los objetos que se analiza con posterioridad para determinar el número de clases de la base de datos y este enfoque del algoritmo confiere flexibilidad al rol que juega el parámetro ε en el algoritmo. Como en DBSCAN, este algoritmo es más eficaz y eficiente que otros métodos de clustering conocidos. Además OPTICS hace una mejor clasificación y nos da la posibilidad de tener una representación de la estructura de los clusters para posteriormente poder extraerlos, dándole al usuario la posibilidad de escoger a qué altura quiere cortar la representación, de forma parecida a como se hace con el dendrograma de los métodos jerárquicos. Sin embargo ambos métodos escalan mejor que éste último.

En este proyecto también hemos aportado al software el desarrollo de una nueva forma de representar los datos en 3 dimensiones según sus atributos y poder visualizar a la clase que pertenecen de una manera fácil. Esta nueva visualización incorporada nos aporta muchas ventajas, sobre todo al usuario del sistema. El usuario podrá decidir en todo momento desde que ángulo ve los datos y desde que distancia. Además podrá decidir que 3 variables quiere visualizar sobre los ejes en tiempo real, incluso hacer varias visualizaciones a la vez. Otros softwares dan la posibilidad de hacer visualizaciones pero no hasta este punto cosa que hace que nuestro sistema vaya un poco más allá.

Capítulo 7

Trabajo futuro

La propuesta que se hace en este trabajo permite seguir investigando sobre los métodos de clustering basados en densidad. En particular hay cinco puntos fundamentales en los cuales enfocarse:

- Cuándo funcionan mejor los algoritmos de DBSCAN y OPTICS.
  - Realizar un estudio detallado sobre como elegir correctamente los parámetros de entrada de los algoritmos (DBSCAN y OPTICS) para una mejor clasificación de los datos.
  - Estudiar con qué tipo de datos funcionan mejor los métodos basados en densidad implementados respecto de otros métodos de clustering.
- Adaptación del algoritmo para clustering de grandes bases de datos.
  - Realizar el clustering de bases de datos muy grandes (millones de registros). Ya existen algunos algoritmos que se han enfocado en el tema. A veces necesitamos hacer clustering de gran cantidad de datos y esto puede ser bastante costoso a nivel tiempo, sobre todo a la hora de leer los datos.

    El objetivo, entonces, es cómo acceder a los datos, sin que eso ocupe demasiado tiempo.

    El plan en nuestro caso, sería investigar qué estructuras podemos usar para guardar los datos y qué procedimientos serían útiles para adaptar los algoritmos internos (OPTICS y DBSCAN) y reducir su coste.
- Desarrollo de herramientas de visualización que faciliten el análisis de los resultados obtenidos.
  - En los últimos tiempos muchos paquetes de software han sido creados para facilitar la tarea de visualización de componentes gráficos, cualquier aplicación que necesite mostrar resultados de una forma más intuitiva y amigable.

    La mayoría de estos paquetes hacen abstracción de objetos y utilización de colores reduciendo la cantidad de elementos a mostrar, simplificando el análisis y brindando un detalle más exhaustivo de información.

    En este proyecto se ha intentado facilitar esa tarea de visualización de los resultados, consiguiendo muy buenos resultados que benefician al usuario.

    Sería útil estudiar qué tipo de mejoras de la visualización se podrían añadir, o incorporar otros métodos de visualización más intuitivos para garantizar en la medida de lo posible en análisis de los resultados.

- Desarrollo de alguna herramienta para la generación y visualización del gráfico k-dist para ayudar al usuario en la determinación de los parámetros de entrada de los algoritmos DBSCAN y OPTICS.
- Adaptación de los algoritmos implementados para la identificación de clases en bases de datos espaciales [Ester 96].

Bibliografía

[Agrawal 98] Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In Proceedings of the ACM SIGMOD Conference on Management of Data, Seattle, Washington (pp. 94-105).

[Anderberg 73] Anderberg, M. R., "Cluster analysis for applications". New York:Academic Press Inc., 1973.

[Anderson 35] Anderson, E. (1935). The Irises of the Gaspe Peninsula. Bulletin of the American Iris Society 59:2-5. Philadelphia, PA, USA.

[Anderson 36] Anderson, E. (1936). The Species Problem in Iris. Annals of the Missouri Botanical Garden, 23, 457-509.

[Ankerst 99] Ankerst, M., Breunig, M. M., Kriegel, H., and Sander, J. (1999). OPTICS: Ordering Points To Identify the Clustering Structure (pp. 49-60). ACM Press.

[Bayona 2000] Bayona, S. (2000, July). Descriptiva de dades y de classes.

[Beckmann 90] Beckmann N., Kriegel H.-P, Schneider R., Seeger B.: "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles", Proc. ACM SIGMOD Int. Conf. on Management of Data, Atlantic City, NJ, ACM Press, New York, 1990, pp. 322-331.

[Beckmann et al. 1990] Beckman, M. E. (ed.) (1990) Journal of Phonetics 18,3, Theme issue: Phonetic Representation.

[Benzécri 80] Benzécri, J. P. (1980). L'analyse des données. Paris: Dunod.

[Brinkhoff et al. 1994] Brinkhoff, T., Kriegel, H.-P., Schneider, R., and Seeger, B. (1994). Multi-Step Processing of Spatial Joins, 23(2), 197-208.

[Castillejo 1996] Castillejo, X. (1996, July). Un entorn de treball per a Klass.

[Digby & Gower 81] Digby, P.G.N. and Gower, J.C. (1981), "Ordination Between- and Within-Groups Applied to Soil Classification", in Down-to-Earth Statistics: Solutions Looking for Geological Problems, Syracuse University Geological Contributions, 8, 63-75.

[Dillon 85] Dillon, W. R., and Goldstein, M. (1984). Multivariate analysis. Methods and applications. Wiley.

[ELKI] http://elki.dbs.ifi.lmu.de/

[Ester 96] Ester, M., Kriegel, H., S, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise (pp. 226-231). AAAI Press.

[Farber 2010] Färber, I., Günnemann, S., Kriegel, H., Kröger, P., Müller, E., Schubert, E., Zimek, A. (n.d.). On Using Class-Labels in Evaluation of Clusterings.

[Fisher 36] Fisher, R.: 1936, The use of multiple measurements in taxonomic problems, in Ann. Eurgenics, Vol. 7, Part II, pp. 179-188.

[Gibert 91] Gibert, K. (1991). "**KLASS**: Estudi d'un sistema d'ajuda al tractament estadístic de grans bases de dades: Tesi de llicenciatura". Dep. LSI, UPC.

[Gibert & Cortés 92] Gibert, K., and Cortés, U. (1992). KLASS: Una herramienta estadística para la creación de prototipos en ISD. In IBERAMIA-92. (pp. 483-497). México.

[Gibert & Cortés 93] Gibert, K. and U. Cortés. "Combining a knowledge based system with a clustering method for an inductive construction of models". In 4th AISTATS, 1993. Fl, USA.

[Gibert 94] Gibert, K. (1994). "L'ús de la informació simbòlica en l'automatització del tractament estadístic de dominis poc estructurats". Ph. D. Thesis. EIO Dep., UPC, Barcelona, Spain. ISBN: 84-689-0579-8.

[Gibert & Cortés 97] Gibert, K., and Cortés, U. (1997). Weighing quantitative and qualitative variables in clustering methods. Mathware and Soft Computing, 4(3), 251-266.

[Gibert & Sonicki 99] Gibert, K., and Sonicki, Z. (1999). Classification Based on Rules and Thyroids Dysfunctions. Applied Stochastic Models in Business and Industry, 15(4), 319-324.

[Gibert & Salvador 2000] Gibert, K., & Salvador, A. (2000). Aproximación difusa a la identificación de situaciones características en el tratamiento de aguas. In X ESTYLF (pp. 497-502).

[Gibert & Sonicki 2002] Gibert, K., and Sonicki, Z. (2002). Impact of Data encoding and Thyroids dysfunctions. Studies in Health Technology and Informatics 90: 494-498.

[Gibert & Nonell 2005 a] Gibert, K., Nonell, R., Velarde, J. M., and Colillas, M. M. (2005). Knowledge Discovery with clustering: impact of metrics and reporting phase by using KLASS. Neural Network World, 15(4), 319-326.

[Gibert & Nonell 2005 b] Descriptive statistics with KLASS. Supporting LaTeX documents ellaboration. In Procs 3rd World Conf on Computational Statistics and Data Analysis pp 90 Limassol (Cyprus)

[Gibert & Nonell 2008] Pre and post-processing in KLASS. Proc. of the iEMSs IVth Int'l Congress of Environmental Modeling and Software (DM-TES'08 Workshop), vol III: 1965-1966.

[Gibert et al. 2008] K. Gibert, A. García-Rudolph, G. Rodríguez-Silva (Dic 2008). The role of KDD Support-Interpretation tools in the conceptualization of medical profiles: An application to neurorehabilitation. 178-182. Acta Informatica Medica, 16(4), Avicena, 0353-8109, Sarajevo.

[Gibert et al. 2010] Luis Salvador-Carulla, José Alberto Salinas, Manuel Martín, Mont-serrat Grané, Karina Gibert, Miquel Roca, Antonio Bulbena (Nov. 2010). A preliminary taxonomy and a standard knowledge base for mental-health system indicators in Spain, 1-28. International journal of mental health systems, 4(29).

[Gibert & Sanchez 2011] K. Gibert, M. Sànchez-Marrè (March 2011). Outcomes from the iEMSs Data Mining in the Environmental Sciences Workshop Series (pp. 983-985). Environmental Modelling and Software, 26, 712-723.

[Gibert et al. 2012] K. Gibert, D. Conti, D. Vrecko (jun 2012). Assisting the end-user in the interpretation of profiles for decision support. An application to wastewater treatment plants,931-944.Environmental Engineering and Management Journal.11(5), U. Iasi, 1582-9596, Romania. http://omicron.ch.tuiasi.ro/EEMJ/pdfs/vol11/no5/6_764_Gibert_11.pdf

[Gibert et al. 2013] Karina Gibert, Gustavo Rodríguez-Silva, Roberta Annicchiarico (feb 2013). Post-processing: bridging the gap between modelling and effective decision-support. The Profile Assessment Grid in Human Behaviour, 1633-1639. Mathematical and Computer Modelling,57(7-8),Elsevier, 0895-7177, Amsterdam.

[Gibert & Conti 2014] K. Gibert, D. Conti (May 2014). On the understanding of profiles by means of post-processing techniques: An application to financial assets. International Journal of computer mathematics. Taylor and Francis. ISSN 0921-7126

[Gibert 2014] Gibert K (2014). Automatic generation of classes interpretation as a bridge between clustering and decision making,154-182. International Journal of Multicriteria Decision Making,4(2),Inderscience,online: 2040-1078, http://www.inderscience.com/info/ingeneral/forthcoming.php?jcode=ijmcdm

[Gibert et al. 2010] K. Gibert, G. Rodríguez-Silva, I. Rodríguez-Roda (2010). Knowledge Discovery with Clustering based on rules by States: A water treatment application, 712-723. Environmental Modelling and Software, 25, 712-723, The Netherlands. ISSN 1364-8152.

[Glasgow 2001] Scale, G. C. (2001). Glasgow coma scale.

[Gowda & Diday 91] Diday, E., and Gowda, K. C. (1991). Symbolic clustering using a new dissimilarity measure. Pattern Recognition, 24(6), 567-578.

[Gowda & Diday 92] Gowda, K. C., Diday, E. (1992). "Symbolic Clustering Using a New Dissimilarity Measure'. In IEEE Transactions on Systems Management and Cybernetics (pp. 368-378).

[Gower 66] Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. Biometrika, 53, 315-328.

[Gower 67] Gower, J. C. (1967). A comparison of some methods of cluster analysis. Biometrics, december.

[Gower 71] Gower, J. C. "A General Coefficient of Similarity and Some of Its Properties". Biometrics, 27, pag. 857-871.

[Hamming 50] R. W. Hamming, Bell System Tech. J. 26, No. 2, lJ.i7 (April 1950).

[Ichino & Yaguchi 89] Ichino, M., & Yaguchi, H. (1989). Generalized Minkowski metrics for mixed features. Trans. IEICE Japó, J72-A(2), 398-405.

[Ichino & Yaguchi 94] Ichino, M., Yaguchi, H. (1994). "Generalized Minkowski metrics for mixed feature-type data analysis". In IEEE Transactions on Systems, Man, and Cybernetics (pp. 698-708).

[Johnson 87] Johnson, S. C.: 1987, Hierarchical clustering schemes, in Psychometrika, Vol. 2, pp. 241-254.

[KDnuggets 06] http://www.kdnuggets.com/polls/2006/data_mining_methods.htm. Data Mining Methods (Apr 2006).

[Lebart *et al.* 85] Lebart, L., Morineau, A., Fenelon, J. P. "Tratamiento estadístico de datos: métodos y programas". Barcelona: Marcombo, 1985.

[MacQueen 67] MacQueen, J. B.: 1967, Some methods for classification and analysis of multivariate observations, in Proc. of the 5th Berkeley Symposium on Math. Statistics and Probability, pp. 281-297, University of California Press.

[Márquez 97] Márquez, J., & Martín, J. C. (1997, October). La clasificación automática en las ciencias de la salud.

[Oliveras 2002] Oliveras Castellà, Josep. (2002). Dades heterogènies amb classificació automàtica. Implementació i comparativa de mètriques mixtes.

[Ralambondrainy 88] Ralambondrainy, H. (1988). A clustering method for nominal data and mixture of numerical and nominal data. Clasification and Related Methods of Data Analysis. H.H.Bock, Elsevier Science Publishers, B.V. (North-Holland).

[Ralambondrainy 95] Ralambondrainy, H. (1995). "A conceptual version of the K-means algorithm". In Pattern Recognition Letters (pp. 1147-1157).

[Rodas 2003] Rodas Osollo, J. E. (2003, April). Knowledge Discovery in repeated and very short serial measures with a blocking factor. Universitat Politecnica de Catalunya.

[Sheikholeslami et al. 98] Sheikholeslami, G. , Chatterjee, S. and Zhang, A. 1998. Wave Cluster:A multi-resolution clustering approach for very large spatial databases. In Proc. 1997 Int. Conf. Very Large Data Bases (VLDB'97).

[Tubau 1999] Tubau, X. (1999, October). Sobre el comportament de les mètriques mixtes en algorismes de Clustering.

[Vázquez & Gibert 2002] Vázquez, F., & Gibert, K. (2002). Robustness of class prediction depending on reference partition in ill-structured domains. In XVIII Iberoamerican Conference on Artificial Intelligence, Workshop de Minería de Datos y Aprendizaje (IBERAMIA2002) (pp. 13-22). Sevilla.

[Wang et al. 97] Wang, W., Yang, J., and Muntz, R. (1997). STING: A statistical information grid approach to spatial data mining (pp. 186-195). Morgan Kaufmann.

[WEKA] http://www.cs.waikato.ac.nz/ml/weka/

[WEKA Documentation] http://www.cs.waikato.ac.nz/ml/weka/documentation.html

Apéndice A

Conjunto de datos Iris

Tabla A.1: Instancias en el dataset Iris

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	I. setosa
4.9	3.0	1.4	0.2	I. setosa
4.7	3.2	1.3	0.2	I. setosa
4.6	3.1	1.5	0.2	I. setosa
5.0	3.6	1.4	0.2	I. setosa
5.4	3.9	1.7	0.4	I. setosa
4.6	3.4	1.4	0.3	I. setosa
5.0	3.4	1.5	0.2	I. setosa
4.4	2.9	1.4	0.2	I. setosa
4.9	3.1	1.5	0.1	I. setosa
5.4	3.7	1.5	0.2	I. setosa
4.8	3.4	1.6	0.2	I. setosa
4.8	3.0	1.4	0.1	I. setosa
4.3	3.0	1.1	0.1	I. setosa
5.8	4.0	1.2	0.2	I. setosa
5.7	4.4	1.5	0.4	I. setosa
5.4	3.9	1.3	0.4	I. setosa
5.1	3.5	1.4	0.3	I. setosa
5.7	3.8	1.7	0.3	I. setosa
5.1	3.8	1.5	0.3	I. setosa
5.4	3.4	1.7	0.2	I. setosa
5.1	3.7	1.5	0.4	I. setosa
4.6	3.6	1.0	0.2	I. setosa
5.1	3.3	1.7	0.5	I. setosa
4.8	3.4	1.9	0.2	I. setosa
5.0	3.0	1.6	0.2	I. setosa
5.0	3.4	1.6	0.4	I. setosa
5.2	3.5	1.5	0.2	I. setosa
5.2	3.4	1.4	0.2	I. setosa
4.7	3.2	1.6	0.2	I. setosa
4.8	3.1	1.6	0.2	I. setosa
5.4	3.4	1.5	0.4	I. setosa
5.2	4.1	1.5	0.1	I. setosa
5.5	4.2	1.4	0.2	I. setosa
4.9	3.1	1.5	0.2	I. setosa
5.0	3.2	1.2	0.2	I. setosa
5.5	3.5	1.3	0.2	I. setosa
4.9	3.6	1.4	0.1	I. setosa
4.4	3.0	1.3	0.2	I. setosa
5.1	3.4	1.5	0.2	I. setosa
5.0	3.5	1.3	0.3	I. setosa
4.5	2.3	1.3	0.3	I. setosa

(continuación)

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
4.4	3.2	1.3	0.2	I. setosa
5.0	3.5	1.6	0.6	I. setosa
5.1	3.8	1.9	0.4	I. setosa
4.8	3.0	1.4	0.3	I. setosa
5.1	3.8	1.6	0.2	I. setosa
4.6	3.2	1.4	0.2	I. setosa
5.3	3.7	1.5	0.2	I. setosa
5.0	3.3	1.4	0.2	I. setosa
7.0	3.2	4.7	1.4	I. versicolor
6.4	3.2	4.5	1.5	I. versicolor
6.9	3.1	4.9	1.5	I. versicolor
5.5	2.3	4.0	1.3	I. versicolor
6.5	2.8	4.6	1.5	I. versicolor
5.7	2.8	4.5	1.3	I. versicolor
6.3	3.3	4.7	1.6	I. versicolor
4.9	2.4	3.3	1.0	I. versicolor
6.6	2.9	4.6	1.3	I. versicolor
5.2	2.7	3.9	1.4	I. versicolor
5.0	2.0	3.5	1.0	I. versicolor
5.9	3.0	4.2	1.5	I. versicolor
6.0	2.2	4.0	1.0	I. versicolor
6.1	2.9	4.7	1.4	I. versicolor
5.6	2.9	3.6	1.3	I. versicolor
6.7	3.1	4.4	1.4	I. versicolor
5.6	3.0	4.5	1.5	I. versicolor
5.8	2.7	4.1	1.0	I. versicolor
6.2	2.2	4.5	1.5	I. versicolor
5.6	2.5	3.9	1.1	I. versicolor
5.9	3.2	4.8	1.8	I. versicolor
6.1	2.8	4.0	1.3	I. versicolor
6.3	2.5	4.9	1.5	I. versicolor
6.1	2.8	4.7	1.2	I. versicolor
6.4	2.9	4.3	1.3	I. versicolor
6.6	3.0	4.4	1.4	I. versicolor
6.8	2.8	4.8	1.4	I. versicolor
6.7	3.0	5.0	1.7	I. versicolor
6.0	2.9	4.5	1.5	I. versicolor
5.7	2.6	3.5	1.0	I. versicolor
5.5	2.4	3.8	1.1	I. versicolor
5.5	2.4	3.7	1.0	I. versicolor
5.8	2.7	3.9	1.2	I. versicolor
6.0	2.7	5.1	1.6	I. versicolor
5.4	3.0	4.5	1.5	I. versicolor
6.0	3.4	4.5	1.6	I. versicolor
6.7	3.1	4.7	1.5	I. versicolor
6.3	2.3	4.4	1.3	I. versicolor
5.6	3.0	4.1	1.3	I. versicolor
5.5	2.5	4.0	1.3	I. versicolor
5.5	2.6	4.4	1.2	I. versicolor
6.1	3.0	4.6	1.4	I. versicolor
5.8	2.6	4.0	1.2	I. versicolor
5.0	2.3	3.3	1.0	I. versicolor
5.6	2.7	4.2	1.3	I. versicolor
5.7	3.0	4.2	1.2	I. versicolor
5.7	2.9	4.2	1.3	I. versicolor
6.2	2.9	4.3	1.3	I. versicolor
5.1	2.5	3.0	1.1	I. versicolor
5.7	2.8	4.1	1.3	I. versicolor

(continuación)

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
6.3	3.3	6.0	2.5	I. virginica
5.8	2.7	5.1	1.9	I. virginica
7.1	3.0	5.9	2.1	I. virginica
6.3	2.9	5.6	1.8	I. virginica
6.5	3.0	5.8	2.2	I. virginica
7.6	3.0	6.6	2.1	I. virginica
4.9	2.5	4.5	1.7	I. virginica
7.3	2.9	6.3	1.8	I. virginica
6.7	2.5	5.8	1.8	I. virginica
7.2	3.6	6.1	2.5	I. virginica
6.5	3.2	5.1	2.0	I. virginica
6.4	2.7	5.3	1.9	I. virginica
6.8	3.0	5.5	2.1	I. virginica
5.7	2.5	5.0	2.0	I. virginica
5.8	2.8	5.1	2.4	I. virginica
6.4	3.2	5.3	2.3	I. virginica
6.5	3.0	5.5	1.8	I. virginica
7.7	3.8	6.7	2.2	I. virginica
7.7	2.6	6.9	2.3	I. virginica
6.0	2.2	5.0	1.5	I. virginica
6.9	3.2	5.7	2.3	I. virginica
5.6	2.8	4.9	2.0	I. virginica
7.7	2.8	6.7	2.0	I. virginica
6.3	2.7	4.9	1.8	I. virginica
6.7	3.3	5.7	2.1	I. virginica
7.2	3.2	6.0	1.8	I. virginica
6.2	2.8	4.8	1.8	I. virginica
6.1	3.0	4.9	1.8	I. virginica
6.4	2.8	5.6	2.1	I. virginica
7.2	3.0	5.8	1.6	I. virginica
7.4	2.8	6.1	1.9	I. virginica
7.9	3.8	6.4	2.0	I. virginica
6.4	2.8	5.6	2.2	I. virginica
6.3	2.8	5.1	1.5	I. virginica
6.1	2.6	5.6	1.4	I. virginica
7.7	3.0	6.1	2.3	I. virginica
6.3	3.4	5.6	2.4	I. virginica
6.4	3.1	5.5	1.8	I. virginica
6.0	3.0	4.8	1.8	I. virginica
6.9	3.1	5.4	2.1	I. virginica
6.7	3.1	5.6	2.4	I. virginica
6.9	3.1	5.1	2.3	I. virginica
5.8	2.7	5.1	1.9	I. virginica
6.8	3.2	5.9	2.3	I. virginica
6.7	3.3	5.7	2.5	I. virginica
6.7	3.0	5.2	2.3	I. virginica
6.3	2.5	5.0	1.9	I. virginica
6.5	3.0	5.2	2.0	I. virginica
6.2	3.4	5.4	2.3	I. virginica
5.9	3.0	5.1	1.8	I. virginica

Apéndice B

Conjunto de datos Guttmann

B.1. Tests utilizados en la evaluación del estado neuropsicológico

Test de atención sostenida (TAS) -TAS.omis.pre, TAS.omis.post, TAS.error.pre, TAS.error,post

La tarea consiste en responder de una manera consistente cada vez que aparece un estímulo determinado entre un conjunto de estímulos diferentes presentados al azar, durante un periodo de tiempo relativamente largo (cinco minutos). Rango de 0 (mejor) a 10 (peor). Se miden omisiones y errores.

Test de Stroop -Stroop.p.c.pre, Stroop.p.c.post, Stroop.int.pre, Stroop.int.post-

Este test está formado por tres condiciones: en la primera (palabra) el sujeto ha de leer nombre de colores (azul, verde y rojo) escritos en tinta de color negra. En la segunda condición (color), ha de denominar los colores de estímulos neutros (XXX) impresos en los colores anteriormente comentados. En la tercera prueba (palabra-color) el sujeto ha de inhibir la respuesta de lectura de la palabra y denominar el color de la tinta en que está escrita. La interferencia se interpola entre las puntuaciones obtenidas en las tres condiciones. En palabra-color (pc) se contabiliza el número de palabras.

Trail making test (TMT) parte A y B -TMT.A.pre, TMA.A.post, TMT.B.pre, TMT.B.post-

Este test consta de dos partes (parte A y parte B). En la parte A al sujeto se le da una hoja con varios números dispersos del 1 al 25 y él debe ir uniéndolos con líneas por orden de menor a mayor. En la parte B, además de números en la hoja hay letras, el sujeto debe unir el primer número, el 1, con la primera letra del abecedario, la A; después el 2 con la B, etc., hasta completar los 13 números y las letras desde la A hasta la L. Se contabilizan los segundos de ejecución.

Dígitos directos e inversos del Test Barcelona -B.d.d.pre, B.d.d.post, B.d.i.pre, B.d.i.post-

Mediante esta prueba se evalúa la capacidad del sujeto para evocar series de dígitos progresivamente más largas en el mismo orden (dígitos directos) y en orden inverso (dígitos inversos). Los dígitos directos son utilizados para valorar la capacidad atencional del sujeto, mientras los dígitos inversos para evaluar la memoria de trabajo. En los directos, el rango va de 0 (peor) a 9 (mejor) y en el inverso de 0 (peor) a 8 (mejor).

Subtest de memoria del test Barcelona -B.m.c.p.e.pre, B.m.c.e.p.post, B.m.c.p.p.pre, B.m.c.p.p.post, B.m.l.p.e.pre, B.m.l.p.e.post, B.m.l.p.p.pre, B.m.l.p.p.post-

Mediante este subtest se explora la memoria verbal del sujeto mediante el recuerdo que éste tiene de dos historias que le son leídas. Valora recuerdo libre (e) y recuerdo con ayudas (p) , tanto a corto (c) como a largo (l) plazo. Rango de 0 (peor) a 24 (mejor).

Aprendizaje verbal del test Barcelona -B.aprend.pre, B.aprend.post-

En este subtest del Test Barcelona se presentan al sujeto de forma verbal diez palabras; éste debe intentar recordar el máximo número de palabras posibles. El test consta de 10 ensayos. En esta prueba se valora la "curva de aprendizaje" del paciente. Rango de 0 (peor) a 100 (mejor).

Wisconsin card sorting test (WCST) -WCST.cat.pre, WCST.cat.post, WCST.e.pe.pre, WCST.e.pe.post-

En este test el sujeto debe ordenar una serie de cartas con dibujos por el color, la forma o el número de elementos que contenga cada una de ellas (categorías). El sujeto debe deducir como ordenarlas por nuestra respuesta, que será únicamente sí o no según si la ha ordenado de acuerdo a una de las categorías, pero no podrá recibir una explicación más compleja que le permita deducir cómo realizar el test. El rango en categorías va de 0 (peor) a 6 (mejor) y en errores perseverativos va de 0 (mejor) a 50 (peor). Se miden la cantidad de imágenes ordenadas y los errores perseverativos.

Subtest de lenguaje del test Barcelona:

Repetición -rep.p.pre, rep.p.post- Valora la capacidad de la persona en repetir palabras. Rango de 0 (peor) a 10 (mejor).

Denominación -den.vv.pre, den.vv.post- Valora la capacidad de la persona en denominar palabras. Rango de 0 (peor) a 14 (mejor).

Comprensión verbal (palabras) -comp.p.pre, comp.p.post- Valora la capacidad de la persona para comprender palabras aisladas. Rango de 0 (peor) a 12 (mejor).

Comprensión verbal (órdenes) -comp.ord.pre, comp.ord.post- Valora la capacidad de la persona para comprender órdenes verbales (simples, semicomplejas y complejas). Rango de 0 (peor) a 16 (mejor).

B.2. Evaluación de las características de la lesión

Escala de Glasgow (GCS, Glasgow Coma Scale) es una escala que se usa para medir el nivel de conciencia de un paciente con traumatismo craneoencefálico.

Para determinarlo se utilizan como indicadores la apertura ocular, la respuesta verbal y la respuesta motora.

Apertura ocular

- Espontánea: 4
- Al estímulo verbal (al pedírselo): 3
- Al recibir un estimulo doloroso: 2
- No responde: 1

Respuesta verbal

- Orientado: 5
- Confuso: 4
- Palabras: 3
- Sonidos incomprensibles: 2
- No responde: 1

Respuesta motora

- Cumple órdenes: 6
- Localiza el estímulo doloroso: 5
- Retira ante el estímulo doloroso: 4
- Respuesta en flexión (postura de decorticación): 3
- Respuesta en extensión (postura de descerebración): 2
- No responde: 1

Los valores de los tres indicadores se suman y dan el resultado en la escala de Glasgow. El nivel normal es 15 (4 + 5 + 6) que corresponde a un individuo sano. El valor mínimo es 3 (1 + 1 + 1).

La puntuación obtenida es empleada para determinar estado clínico del paciente, pronóstico, indicaciones terapéuticas y realizar un seguimiento del estado neurológico. Cuando se emplea en un paciente con trauma craneoencefálico (TCE) se puede clasificar como:

- TCE Leve .... 13-15 puntos
- TCE Moderado.. 9-12 puntos
- TCE Severo.... 8 Puntos o menos

B.3. Listado de tests neuropsicológicos

Tabla B.1: Listado de tests neuropsicológicos

Nombre	Descripción	Valores	Mecánica	Función
TAS.omis	TAS Test Atención Sostenida Número de omisiones	0 a 10	No presiona barra espaciadora cuando se presenta el estímulo target	Atención

(continuación)

Nombre	Descripción	Valores	Mecánica	Función
TAS.error	TAS Test Atención Sostenida Número de errores	0 a 12	Presiona barra espaciadora cuando se presenta estímulo diferente al target	Atención
Stroop.p.c.	Stroop palabra color Número de palabras	0 a 300	Se contabiliza el número de palabras donde el sujeto denomina el nombre de la tinta en que están impresas	Atención
TMT.A	Trial Making Test A tiempo en segundos	0 a 400	Trial Making Test A se contabiliza el tiempo en unir los numeros del 1 al 25 en orden ascendente	Atención
B.d.d.	Barcelona dígitos directos largo de la serie	0 a 9	Repetir series de dígitos progresivamente mas largas	Memoria
B.d.i	Barcelona dígitos inversos largo de la serie	0 a 8	Repetir series de dígitos progresivamente mas largas en el orden inverso al original	Memoria
B.m.c.p.e.	Barcelona memoria corto plazo espontáneo	0 a 24	Valora memoria verbal mediante el recuerdo de dos historias que le son leídas, se asigna un puntaje sin guiar al paciente	Memoria
B.m.c.p.p.	Barcelona memoria corto plazo pregunta	0 a 24	Valora memoria verbal mediante el recuerdo de dos historias que le son leídas, asignando un puntaje guiando al paciente con preguntas	Memoria
B.m.l.p.e.	Barcelona memoria largo plazo espontáneo	0 a 24	Valora memoria verbal mediante el recuerdo de dos historias que le son leídas, se asigna un puntaje sin guiar al paciente	Memoria
B.m.l.p.p.	Barcelona memoria largo plazo pregunta	0 a 24	Valora memoria verbal mediante el recuerdo de dos historias que le son leídas, asignando un puntaje guiando al paciente con preguntas	Memoria
B. aprend	Barcelona aprendizaje	0 a 100	Valora memoria verbal mediante el recuerdo de 10 palabras que le son leídas 10 veces	Memoria
B.v.	Barcelona			
TMT.B	Trial Making Test B tiempo en segundos	0 a 800	Trial Making Test A se contabiliza el tiempo en unir los numeros y letras del 1 al 25 en orden ascendente	Funciones Ejecutivas

(continuación)

Nombre	Descripción	Valores	Mecánica	Función
Stroop.int	Stroop Interferencia	-25.0 a 25.0	P= lee nombres de colores escritos en tinta negra C= lee nombres de colores escritos en tinta de colores PC=denomina el color de la tinta P*C/(P+C)=PC' Interferencia= PC-PC'	Funciones Ejecutivas
WCST.cat	Wisconsin categorías	0 a 6	Ordenar imágenes por color, forma o número de elementos	Funciones Ejecutivas
WCST.e.pe	Wisconsin errores perseverativos	0 a 50	Ordena de manera incorrecta, se le da la pista (que siempre es SI o NO) y vuelve a ordenarlo de manera incorrecta	Funciones Ejecutivas
Rep.p	Repetir palabras	0 a 10	Valora la capacidad de la persona de repetir palabras	Lenguaje
Den.vv	Denominar palabras	0 a 14	Valora la capacidad de la persona de denominar palabras	Lenguaje
Comp.p	Comprender palabras	0 a 12	Valora la capacidad de la persona para comprender palabras aisladas	Lenguaje
Comp.ord	Comprender ordenes	0 a 16	Valora la capacidad de la persona para comprender ordenes verbales	Lenguaje

B.4. Descriptiva univariante

Estadístics sumaris	TAS.omis.post	TAS.omis.dife	TAS.error.post	TAS.error.dife	Stroop.p.c.post
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	5.1489	-3.0213	4.5532	-3.4043	20.6383
Mediana	4	-1	2	0	25
Primer quartil (Q1)	1	-7	0	-9	0
Tercer quartil (Q3)	10	0	10	0	35
Mínim	0	-10	0	-11	0
Màxim	10	1	10	3	60
Quasi-desviació típica	4.2528	3.8757	4.4858	4.4996	17.3912
Coeficient de variació	0.8171	-1.2691	0.9747	-1.3076	0.8337

Tabla B.2: Estadísticos sumarios de variables de Guttmann 1

Estadístics sumaris	Stroop.p.c.dife	TMA.A.post	TMT.A.dife	B.d.d.post	B.d.d.dife
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	13.9149	184.2128	-60.4255	5.1702	3.0638
Mediana	7	210	0	5	4
Primer quartil (Q1)	0	66	-193	5	0
Tercer quartil (Q3)	26	300	0	7	5
Mínim	-9	26	-274	0	-1
Màxim	46	300	60	8	8
Quasi-desviació típica	15.6522	117.7831	103.4069	2.1399	2.7059
Coeficient de variació	1.1128	0.6325	-1.693	0.4095	0.8737

Tabla B.3: Estadísticos sumarios de variables de Guttmann 2

Estadístics sumaris	B.d.i.post	B.d.i.dife	B.m.c.e.p.post	B.m.c.e.p.dife	B.m.c.p.p.post
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	3.1489	1.8936	8.8085	4.8511	12.4255
Mediana	3	2	9	4	14
Primer quartil (Q1)	3	0	6	1	9
Tercer quartil (Q3)	4	3	12	9	16
Mínim	0	-1	0	-5	0
Màxim	5	5	18	15	22
Quasi-desviació típica	1.351	1.6449	4.8483	4.6903	6.1952
Coeficient de variació	0.4244	0.8594	0.5445	0.9565	0.4933

Tabla B.4: Estadísticos sumarios de variables de Guttmann 3

Estadístics sumaris	B.m.c.p.p.dife	B.m.l.p.e.post	B.m.l.p.e.dife	B.m.l.p.p.post	B.m.l.p.p.dife
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	6.766	6.1064	3.6596	10.4255	5.4894
Mediana	5	6	2	11	4
Primer quartil (Q1)	0	0	0	6	0
Tercer quartil (Q3)	14	10	7	15	11
Mínim	-10	0	-7	0	-9
Màxim	20	19	15	22	17
Quasi-desviació típica	7.0563	5.1298	4.3101	6.358	6.4298
Coeficient de variació	1.0318	0.8311	1.1652	0.6033	1.1588

Tabla B.5: Estadísticos sumarios de variables de Guttmann 4

Estadístics sumaris	B.aprend.post	B.aprend.dife	B.v.rev.post	B.v.rev.dife	TMT.B.post
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	46.8298	25.1489	5.4255	2.8723	333.766
Mediana	55	20	7	1	500
Primer quartil (Q1)	29	2	0	0	150
Tercer quartil (Q3)	65	50	9	7	500
Mínim	0	-3	0	-5	57
Màxim	90	67	10	10	500
Quasi-desviació típica	25.2538	24.0326	3.7285	3.6749	178.2489
Coeficient de variació	0.5335	0.9454	0.6799	1.2657	0.5283

Tabla B.6: Estadísticos sumarios de variables de Guttmann 5

Estadístics sumaris	TMT.B.dife	Stroop.int.post	Stroop.int.dife	WCST.cat.post	WCST.cat.dife
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	-97.6596	-7.1238	11.5085	2	1.5319
Mediana	0	-2.89	2.6	0	0
Primer quartil (Q1)	-197	-25	0	0	0
Tercer quartil (Q3)	0	4.36	25.29	4	4
Mínim	-425	-25	-6.49	0	0
Màxim	27	17.1	38.6	6	6
Quasi-desviació típica	148.0682	14.6277	13.852	2.5538	2.3392
Coeficient de variació	-1.5	-2.0314	1.1908	1.2632	1.5106

Tabla B.7: Estadísticos sumarios de variables de Guttmann 6

Estadístics sumaris	WCST.e.pe.post	WCST.e.pe.dife	rep.p.post	rep.p.dife	den.vv.post
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	34.3191	-12.5106	9.7021	0.2766	13.1277
Mediana	50	0	10	0	14
Primer quartil (Q1)	19	-25	10	0	14
Tercer quartil (Q3)	50	0	10	0	14
Mínim	0	-50	2	0	2
Màxim	50	6	10	9	14
Quasi-desviació típica	18.7838	18.0878	1.3008	1.3465	2.7632
Coeficient de variació	0.5415	-1.4303	0.1326	4.8161	0.2082

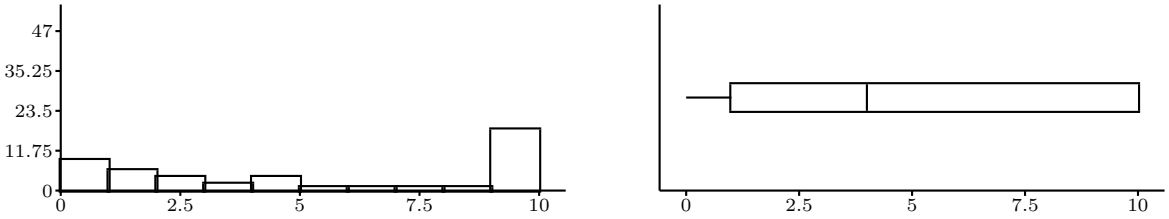
Tabla B.8: Estadísticos sumarios de variables de Guttmann 7

Estadístics sumaris	den.vv.dife	comp.p.post	comp.p.dife	comp.ord.post	comp.ord.dife
Nombre d'objectes	47				
Nombre de dades mancants	0				
Nombre d'observacions útils	47				
Mitjana	0.5745	11.4043	0.4255	15.0213	0.3617
Mediana	0	12	0	16	0
Primer quartil (Q1)	0	12	0	16	0
Tercer quartil (Q3)	0	12	0	16	0
Mínim	0	0	0	1	0
Màxim	10	12	10	16	10
Quasi-desviació típica	1.9863	2.4642	2.0403	3.4293	1.6209
Coeficient de variació	3.4207	0.2138	4.7434	0.2259	4.4333

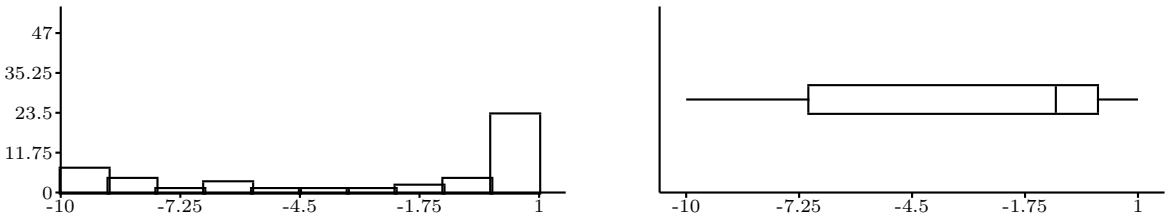
Tabla B.9: Estadísticos sumarios de variables de Guttmann 8

Histograma **Boxplot**

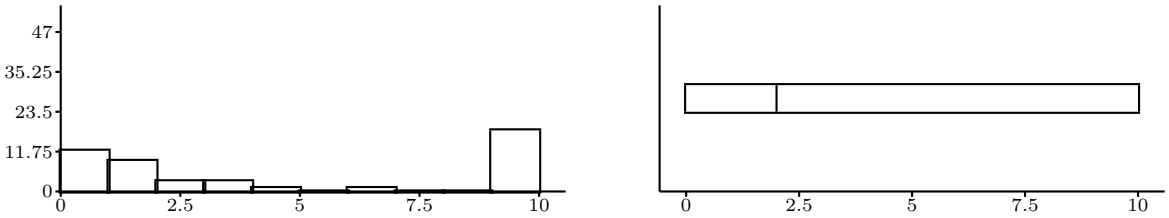
Variable TAS.omis.post



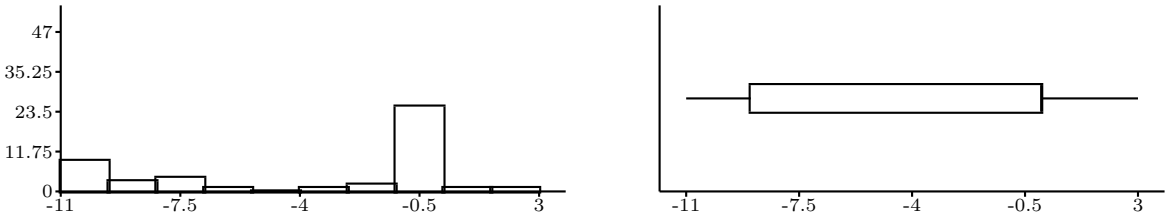
Variable TAS.omis.dife



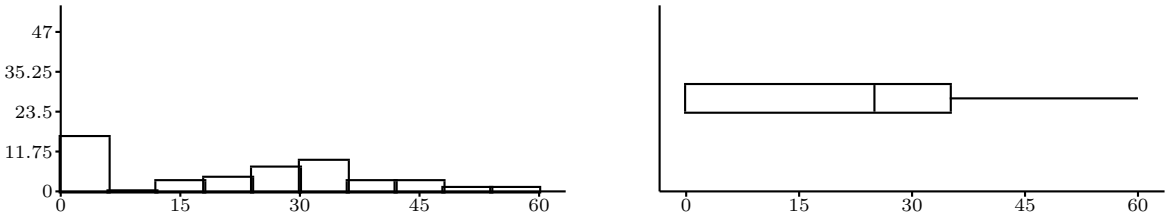
Variable TAS.error.post



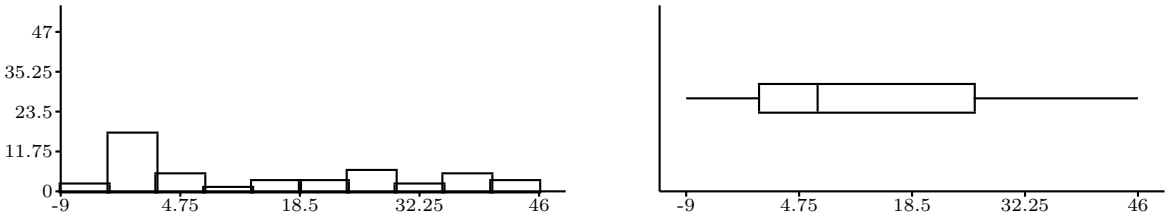
Variable TAS.error.dife



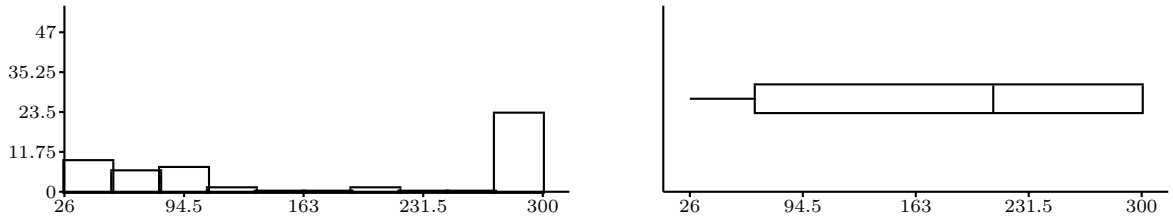
Variable Stroop.p.c.post



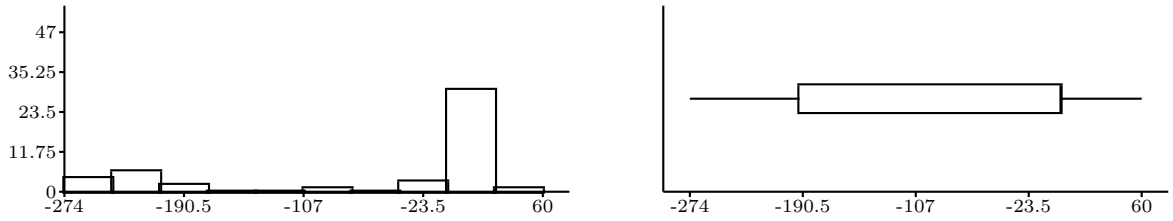
Variable Stroop.p.c.dife



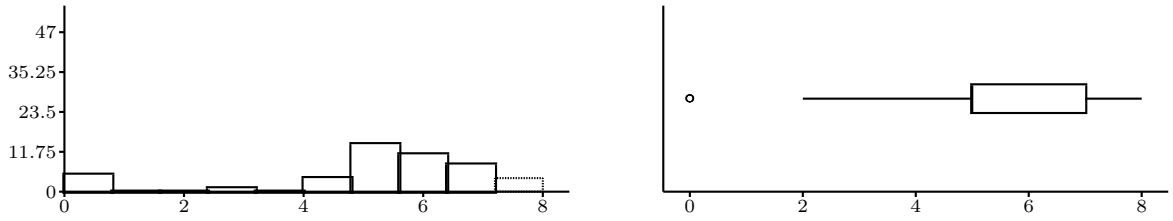
Variable TMA.A.post



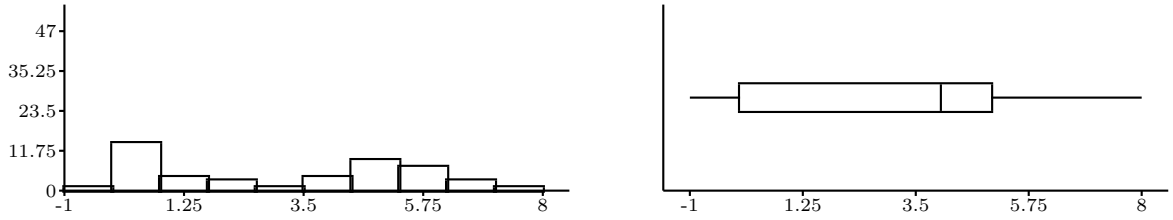
Variable TMT.A.dife



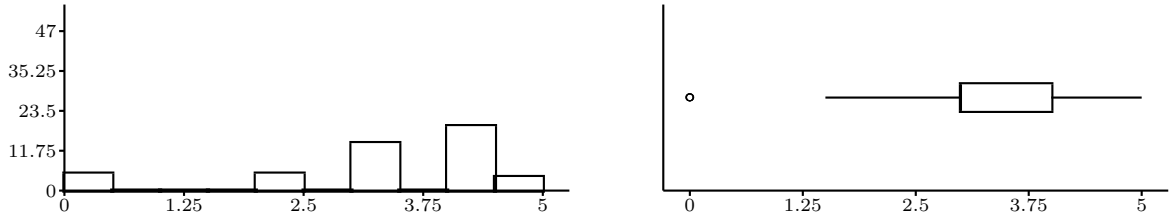
Variable B.d.d.post



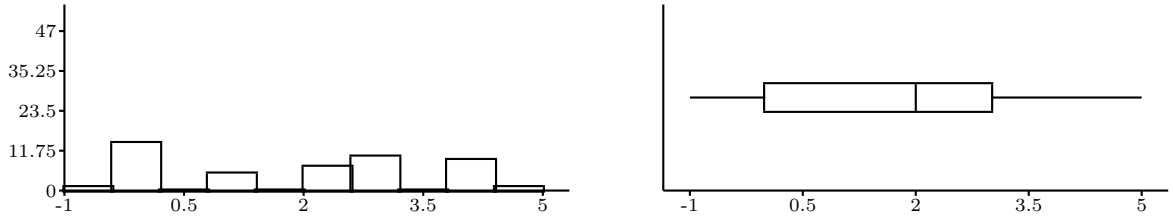
Variable B.d.d.dife



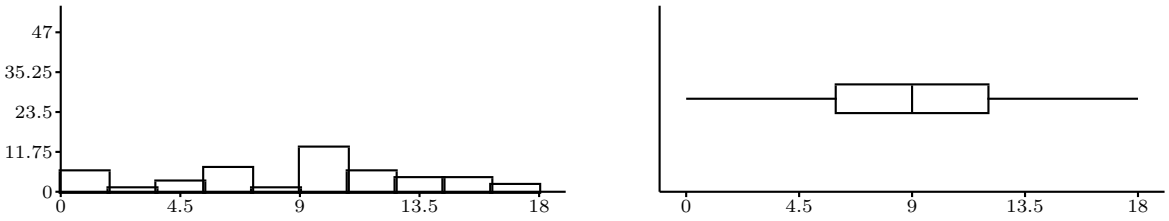
Variable B.d.i.post



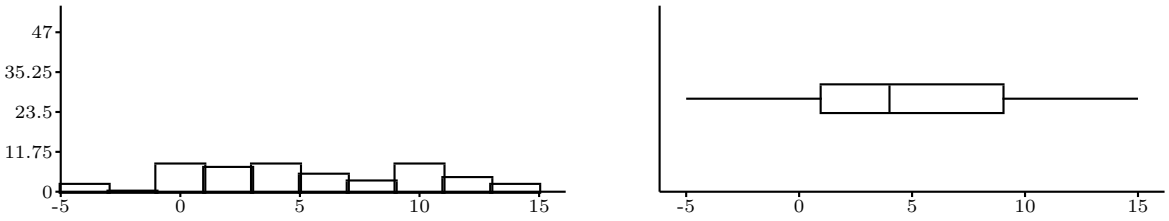
Variable B.d.i.dife



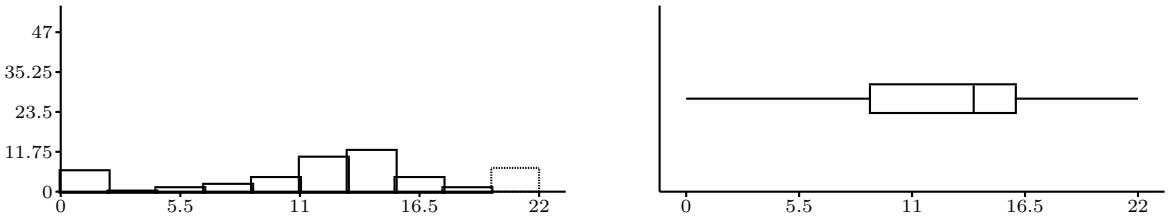
Variable B.m.c.e.p.post



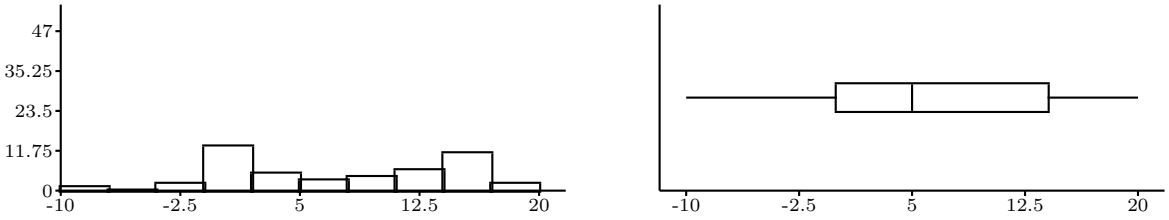
Variable B.m.c.e.p.dife



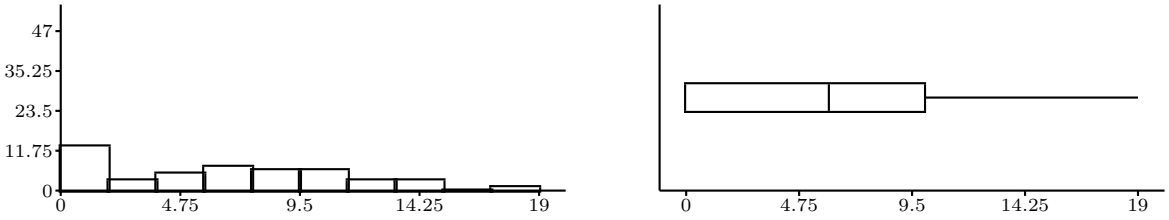
Variable B.m.c.p.p.post



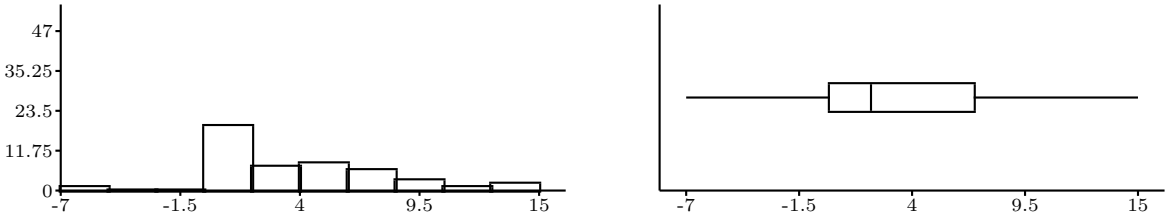
Variable B.m.c.p.p.dife



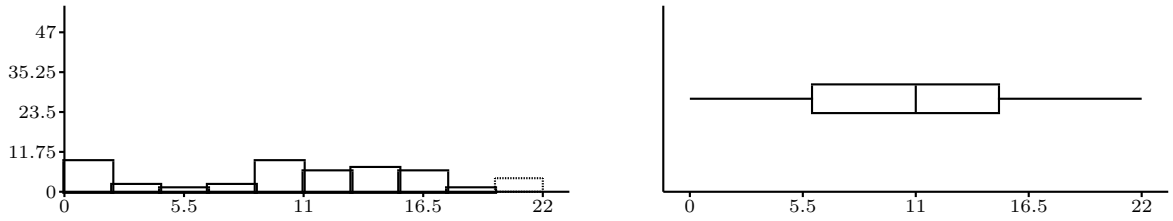
Variable B.m.l.p.e.post



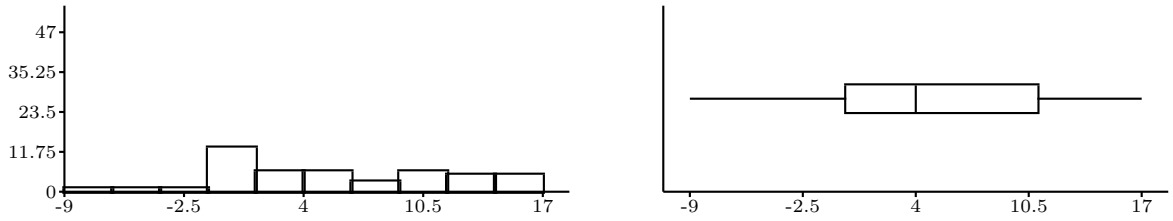
Variable B.m.l.p.e.dife



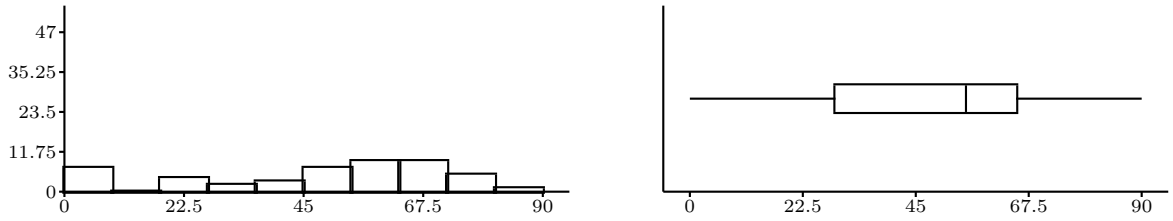
Variable B.m.l.p.p.post



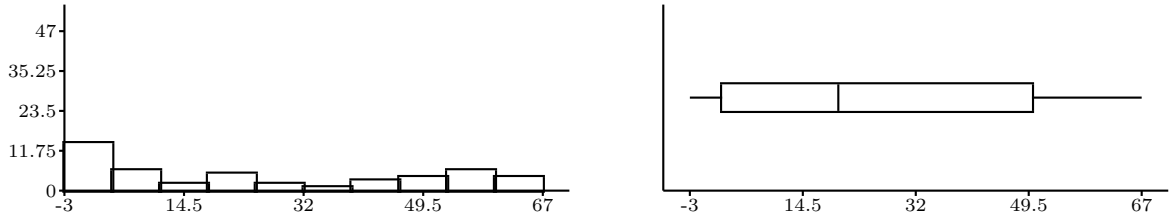
Variable B.m.l.p.p.dife



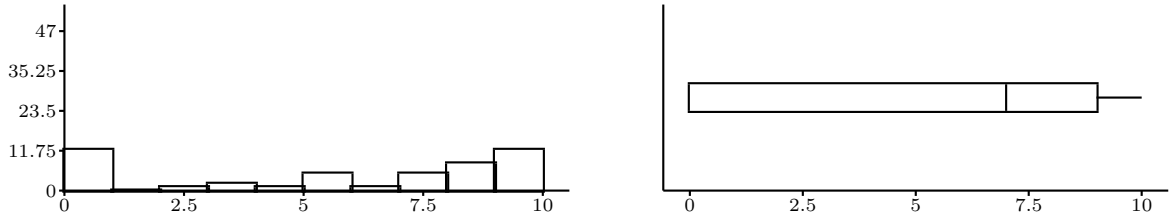
Variable B.aprend.post



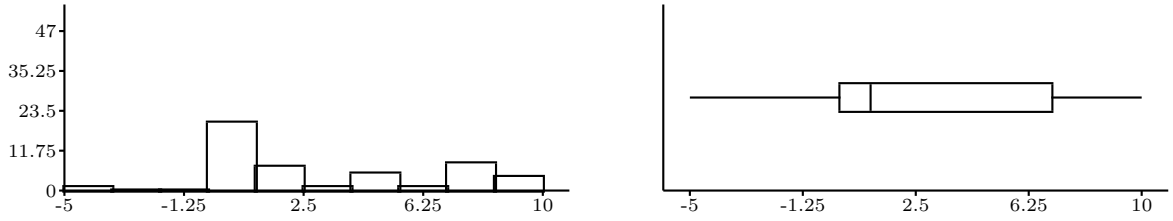
Variable B.aprend.dife



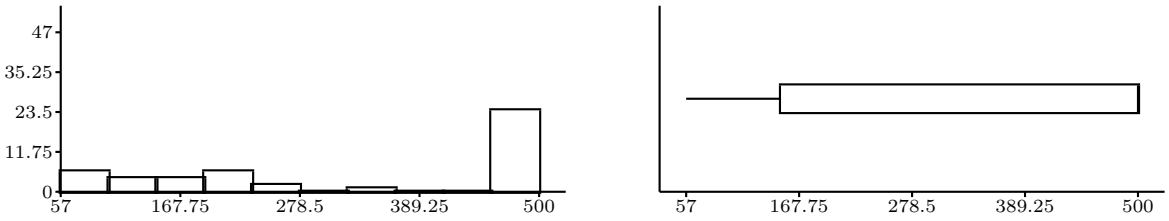
Variable B.v.rev.post



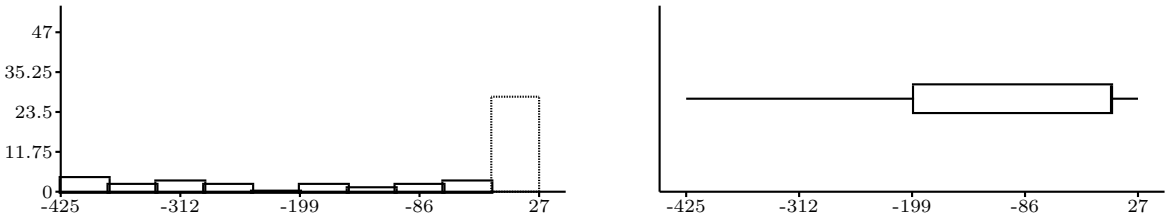
Variable B.v.rev.dife



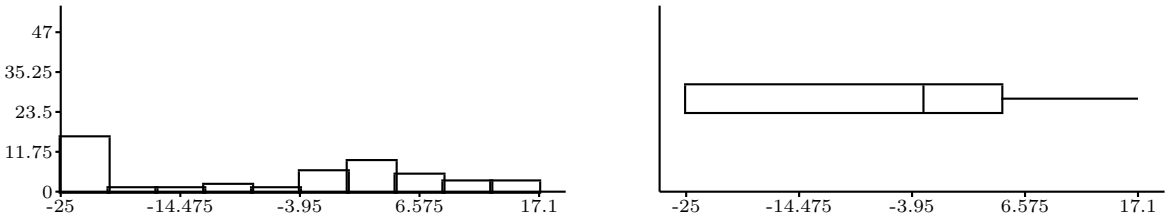
Variable TMT.B.post



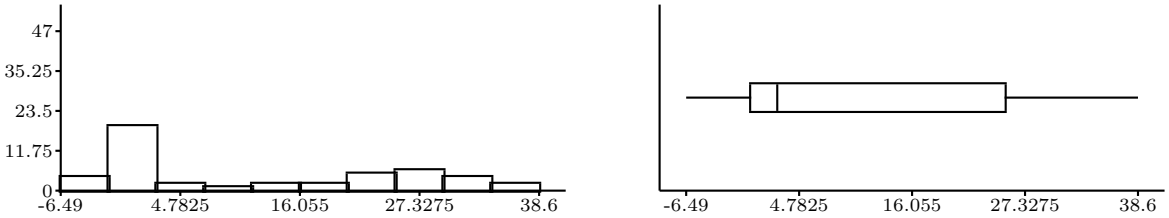
Variable TMT.B.dife



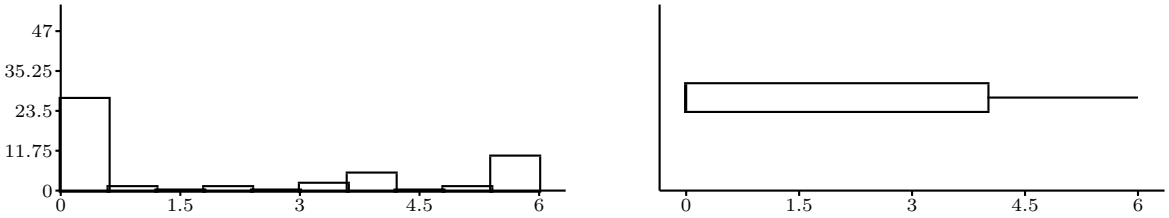
Variable Stroop.int.post



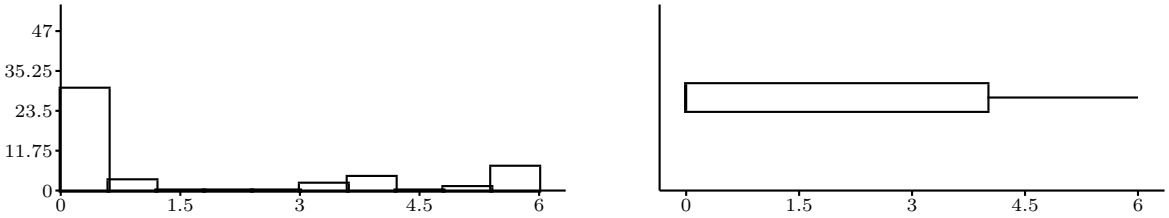
Variable Stroop.int.dife



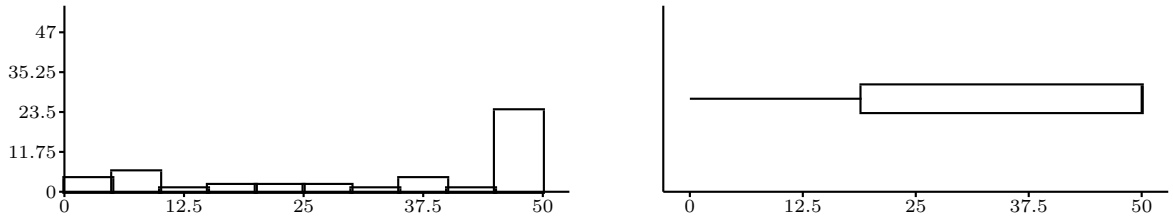
Variable WCST.cat.post



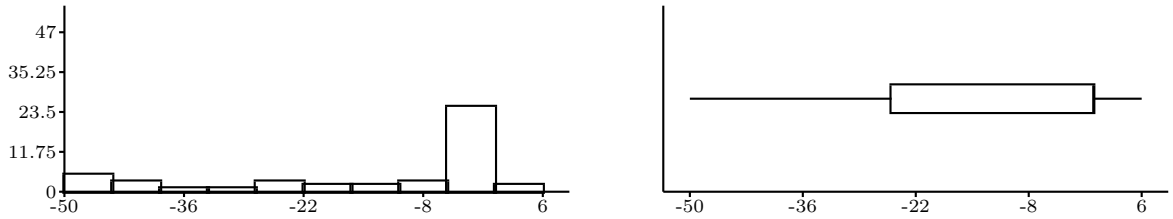
Variable WCST.cat.dife



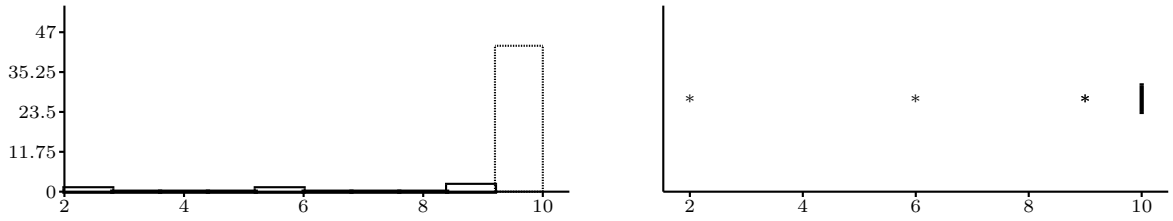
Variable WCST.e.pe.post



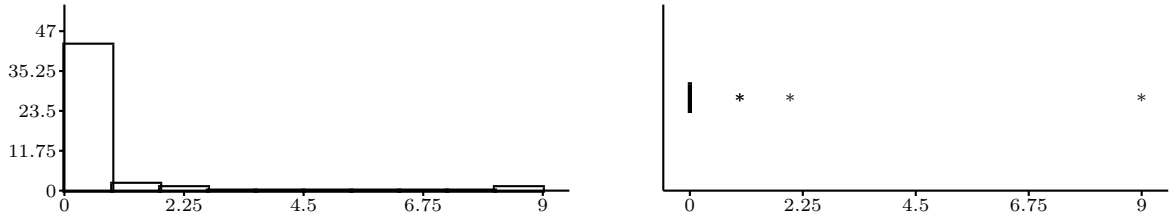
Variable WCST.e.pe.dife



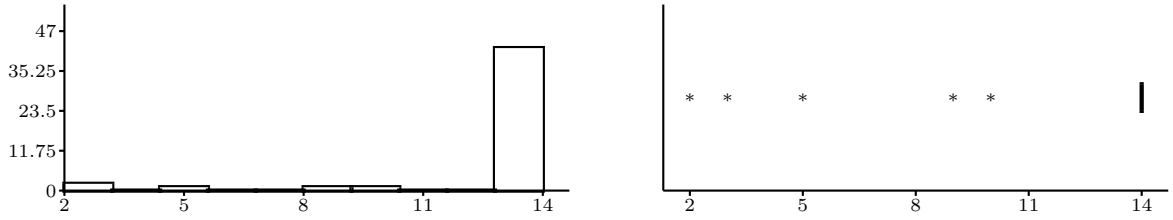
Variable rep.p.post



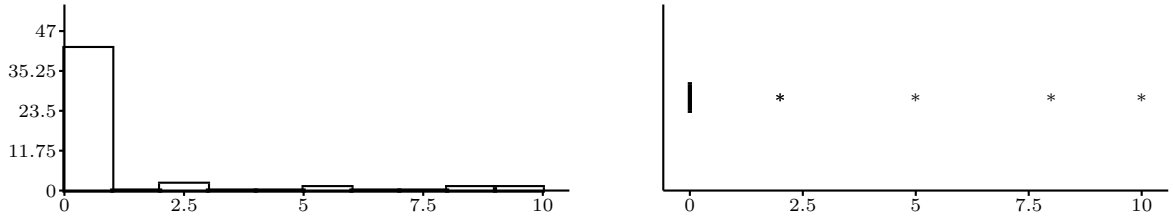
Variable rep.p.dife



Variable den.vv.post



Variable den.vv.dife



Variable comp.p.post

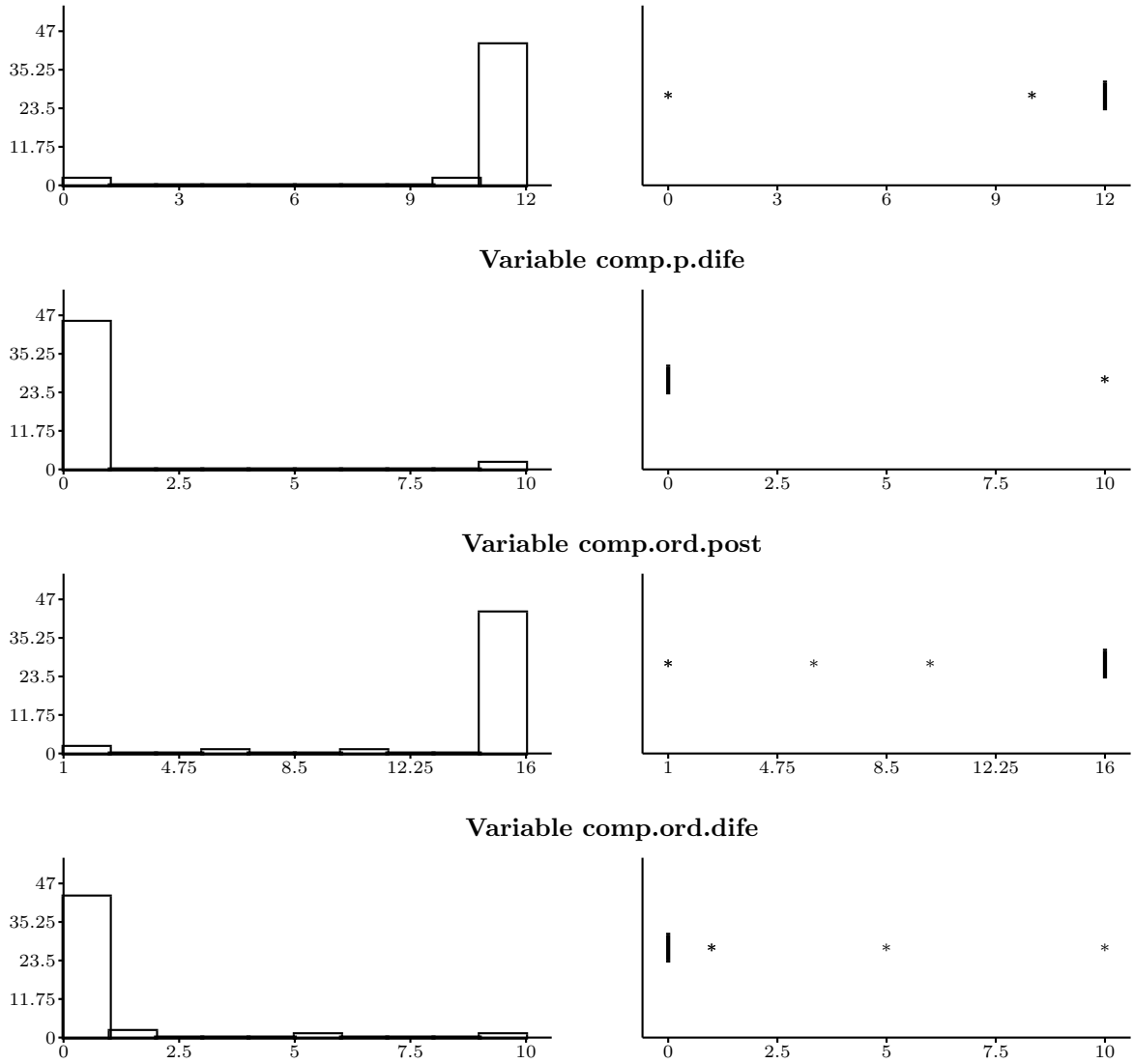


Tabla B.10: Histogramas y boxplots de variables de Guttmann.

B.5. Descriptiva por grupos con un método jerárquico

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
TAS.omis.post	$\bar{\mathbf{X}}$	9.1875	5.2857	0	3.75	2.1538
	S	1.905	4.5722	0	3.9551	2.4099
	N*	0	0	0	0	0
TAS.omis.dife	$\bar{\mathbf{X}}$	-0.8125	-4.7143	-5	-6.25	-2.3846
	S	1.905	4.5722	4.5826	3.9551	3.6638
	N*	0	0	0	0	0
TAS.error.post	$\bar{\mathbf{X}}$	8.6875	5.1429	0.3333	3.375	0.8462
	S	2.9826	4.6342	0.5774	4.3074	1.0682
	N*	0	0	0	0	0
TAS.error.dife	$\bar{\mathbf{X}}$	-1.3125	-4.8571	-7	-6.625	-2.3846
	S	2.9826	4.6342	6.0828	4.3074	4.5192
	N*	0	0	0	0	0
Stroop.p.c.post	$\bar{\mathbf{X}}$	0	25.2857	35.3333	30.25	34.2308
	S	0	10.291	10.504	7.421	12.6369
	N*	0	0	0	0	0
Stroop.p.c.dife	$\bar{\mathbf{X}}$	0	25.2857	35.3333	30.25	9.9231
	S	0	10.291	10.504	7.421	13.8652
	N*	0	0	0	0	0

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
TMA.A.post	$\bar{\mathbf{X}}$	294.375	300	58.3333	67.625	87.0769
	S	22.5	0	22.0303	26.4572	69.3607
	N*	0	0	0	0	0
TMT.A.dife	$\bar{\mathbf{X}}$	-5.625	0	-241.6667	-232.375	-12.7692
	S	22.5	0	22.0303	26.4572	57.8808
	N*	0	0	0	0	0
B.d.d.post	$\bar{\mathbf{X}}$	3.6875	5.2857	5.6667	5.5	6.6154
	S	2.7741	0.7559	0.5774	1.3093	1.1209
	N*	0	0	0	0	0
B.d.d.dife	$\bar{\mathbf{X}}$	3.375	5.2857	0.3333	5.5	0.6154
	S	2.8954	0.7559	0.5774	1.3093	0.9608
	N*	0	0	0	0	0
B.d.i.post	$\bar{\mathbf{X}}$	2.1875	3.1429	3.3333	3.5	4.0769
	S	1.6419	0.6901	0.5774	1.069	0.6405
	N*	0	0	0	0	0
B.d.i.dife	$\bar{\mathbf{X}}$	1.875	3.1429	0	3.5	0.6923
	S	1.7464	0.6901	0	1.069	0.7511
	N*	0	0	0	0	0

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
B.m.c.e.p.post	$\bar{\mathbf{X}}$	5	8.2857	14.6667	9.5	12
	S	4.5461	2.9277	1.1547	3.4226	3.6742
	N*	0	0	0	0	0
B.m.c.e.p.dife	$\bar{\mathbf{X}}$	4.6875	8.2857	3	9.5	0.7692
	S	4.423	2.9277	2	3.4226	2.8034
	N*	0	0	0	0	0
B.m.c.p.p.post	$\bar{\mathbf{X}}$	8.9375	10.7143	19	12.875	15.8462
	S	7.4518	3.4017	2.6458	2.9001	4.9303
	N*	0	0	0	0	0
B.m.c.p.p.dife	$\bar{\mathbf{X}}$	8.25	10.7143	0.3333	12.875	0.5385
	S	7.6551	3.4017	3.0551	2.9001	4.0128
	N*	0	0	0	0	0
B.m.l.p.e.post	$\bar{\mathbf{X}}$	2.5625	6.1429	9.6667	6.25	9.5385
	S	3.1826	5.64	5.1316	4.9785	4.5573
	N*	0	0	0	0	0
B.m.l.p.e.dife	$\bar{\mathbf{X}}$	2.5625	6.1429	6	6.25	1.5385
	S	3.1826	5.64	3.6056	4.9785	3.1785
	N*	0	0	0	0	0

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
B.m.l.p.p.post	$\bar{\mathbf{X}}$	7.3125	8.8571	17	9.625	14.0769
	S	6.9062	5.1455	3.6056	4.6272	5.2355
	N*	0	0	0	0	0
B.m.l.p.p.dife	$\bar{\mathbf{X}}$	7	8.8571	2.6667	9.625	-0.0769
	S	6.9378	5.1455	4.1633	4.6272	3.6162
	N*	0	0	0	0	0
B.aprend.post	$\bar{\mathbf{X}}$	30.3125	32.7143	64	49.75	69
	S	26.966	19.7629	4.3589	13.4881	11.804
	N*	0	0	0	0	0
B.aprend.dife	$\bar{\mathbf{X}}$	27.6875	32.7143	3.6667	49.75	7.7692
	S	27.7842	19.7629	4.1633	13.4881	7.5294
	N*	0	0	0	0	0
B.v.rev.post	$\bar{\mathbf{X}}$	2.4375	4.8571	5	7.625	8.1538
	S	2.8512	4.6342	5	1.4079	1.9081
	N*	0	0	0	0	0
B.v.rev.dife	$\bar{\mathbf{X}}$	1.9375	4.8571	0.6667	7.625	0.5385
	S	2.4891	4.6342	1.1547	1.4079	2.5038
	N*	0	0	0	0	0

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
TMT.B.post	$\bar{\mathbf{X}}$	500	500	120	169.375	190.1538
	S	0	0	52.915	66.4077	118.3553
	N*	0	0	0	0	0
TMT.B.dife	$\bar{\mathbf{X}}$	0	0	-380	-330.625	-61.9231
	S	0	0	52.915	66.4077	72.3124
	N*	0	0	0	0	0
Stroop.int.post	$\bar{\mathbf{X}}$	-25	4.4671	-3.15	0.5	3.0277
	S	0	6.5632	7.0741	5.6442	10.6285
	N*	0	0	0	0	0
Stroop.int.dife	$\bar{\mathbf{X}}$	0	29.4671	21.85	25.5	5.0062
	S	0	6.5632	7.0741	5.6442	10.9222
	N*	0	0	0	0	0
WCST.cat.post	$\bar{\mathbf{X}}$	0.4375	2.2857	3	2.375	3.3077
	S	1.5042	2.9277	2.6458	2.7223	2.6263
	N*	0	0	0	0	0
WCST.cat.dife	$\bar{\mathbf{X}}$	0.4375	2.2857	3	2.375	1.6154
	S	1.5042	2.9277	2.6458	2.7223	2.3288
	N*	0	0	0	0	0

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
WCST.e.pe.post	$\bar{\mathbf{X}}$	42.5	34.8571	37.6667	25.625	28.5385
	S	16.7571	21.6905	12.0139	21.6922	16.9981
	N*	0	0	0	0	0
WCST.e.pe.dife	$\bar{\mathbf{X}}$	-7.5	-15.1429	-12.3333	-24.375	-10
	S	16.7571	21.6905	12.0139	21.6922	15.2042
	N*	0	0	0	0	0
rep.p.post	$\bar{\mathbf{X}}$	9.125	10	10	10	10
	S	2.1564	0	0	0	0
	N*	0	0	0	0	0
rep.p.dife	$\bar{\mathbf{X}}$	0.8125	0	0	0	0
	S	2.2574	0	0	0	0
	N*	0	0	0	0	0
den.vv.post	$\bar{\mathbf{X}}$	11.4375	14	14	14	14
	S	4.3354	0	0	0	0
	N*	0	0	0	0	0
den.vv.dife	$\bar{\mathbf{X}}$	1.6875	0	0	0	0
	S	3.1774	0	0	0	0
	N*	0	0	0	0	0

	CLASSE	**C0**	**Inhibidos5**	**valorables1**	**Inhibidos12**	**valorables13**
VARIABLE	N = 47	$n_c = 16$	$n_c = 7$	$n_c = 3$	$n_c = 8$	$n_c = 13$
comp.p.post	$\bar{\mathbf{X}}$	10.25	12	12	12	12
	S	4.0579	0	0	0	0
	N*	0	0	0	0	0
comp.p.dife	$\bar{\mathbf{X}}$	1.25	0	0	0	0
	S	3.4157	0	0	0	0
	N*	0	0	0	0	0
comp.ord.post	$\bar{\mathbf{X}}$	13.125	16	16	16	16
	S	5.5	0	0	0	0
	N*	0	0	0	0	0
comp.ord.dife	$\bar{\mathbf{X}}$	1.0625	0	0	0	0
	S	2.6949	0	0	0	0
	N*	0	0	0	0	0

Tabla B.11: Descriptiva por grupos ejecución Clasificación condicionada para Guttmann.

B.6. Descriptiva por grupos con DBSCAN ($\varepsilon = 82$, $\nu = 2$)

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
TAS.omis.post	$\bar{\mathbf{X}}$	7.6087	2.1111	5.5	4.3333	0.4286	7
	S	3.6648	3.1798	6.364	1.5275	0.5345	3
	N*	0	0	0	0	0	0
TAS.omis.dife	$\bar{\mathbf{X}}$	-2.3913	-6.2222	-4.5	-1.3333	-1.7143	-2
	S	3.6648	3.7006	6.364	2.0817	3.7289	3.4641
	N*	0	0	0	0	0	0
TAS.error.post	$\bar{\mathbf{X}}$	7.1739	2	5	1.3333	0.5714	4.3333
	S	4.1082	3.2787	7.0711	1.5275	0.5345	5.1316
	N*	0	0	0	0	0	0
TAS.error.dife	$\bar{\mathbf{X}}$	-2.8261	-7.1111	-5	-1.3333	-1.4286	-2.3333
	S	4.1082	4.285	7.0711	1.1547	3.8235	6.8069
	N*	0	0	0	0	0	0
Stroop.p.c.post	$\bar{\mathbf{X}}$	9.1304	31.8889	30.5	35	38	13.6667
	S	13.9522	9.0477	2.1213	5	14.8661	13.0512
	N*	0	0	0	0	0	0
Stroop.p.c.dife	$\bar{\mathbf{X}}$	9.1304	31.8889	30.5	7.3333	6.2857	10
	S	13.9522	9.0477	2.1213	10.7858	13.7564	14
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
TMA.A.post	$\bar{\mathbf{X}}$	300	64	70	83.6667	50.5714	145.6667
	S	0	26.0336	24.0416	18.8768	11.8161	56.0922
	N*	0	0	0	0	0	0
TMT.A.dife	$\bar{\mathbf{X}}$	0	-236	-230	-5.3333	-2.4286	-74.3333
	S	0	26.0336	24.0416	13.3167	12.8563	127.2255
	N*	0	0	0	0	0	0
B.d.d.post	$\bar{\mathbf{X}}$	4.2174	5.6667	5	6.3333	6.7143	6.3333
	S	2.4855	0.7071	2.8284	1.1547	1.1127	1.5275
	N*	0	0	0	0	0	0
B.d.d.dife	$\bar{\mathbf{X}}$	3.7391	3.8889	5	1	0.4286	2.3333
	S	2.6149	2.7588	2.8284	1	0.7868	3.5119
	N*	0	0	0	0	0	0
B.d.i.post	$\bar{\mathbf{X}}$	2.4348	3.5556	3	4	4.2857	4
	S	1.4405	0.8819	1.4142	1	0.488	0
	N*	0	0	0	0	0	0
B.d.i.dife	$\bar{\mathbf{X}}$	2.087	2.4444	3	1	0.7143	1.6667
	S	1.6213	2.0069	1.4142	1	0.7559	2.0817
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
B.m.c.e.p.post	$\bar{\mathbf{X}}$	6.4783	12.1111	5.5	11	12.1429	9
	S	4.5912	2.8916	2.1213	4.5826	3.9761	6
	N*	0	0	0	0	0	0
B.m.c.e.p.dife	$\bar{\mathbf{X}}$	5.8261	8.2222	5.5	2.3333	0.4286	-0.3333
	S	4.2816	4.5491	2.1213	0.5774	2.6367	4.1633
	N*	0	0	0	0	0	0
B.m.c.p.p.post	$\bar{\mathbf{X}}$	9.6087	15.4444	10.5	14.3333	16.4286	15
	S	6.5833	3.5746	3.5355	5.5076	4.7559	6.5574
	N*	0	0	0	0	0	0
B.m.c.p.p.dife	$\bar{\mathbf{X}}$	8.6087	9.2222	10.5	3.6667	-0.1429	2
	S	6.6246	7.12	3.5355	1.5275	1.9518	12
	N*	0	0	0	0	0	0
B.m.l.p.e.post	$\bar{\mathbf{X}}$	4.1304	8.7778	0	7.6667	9.7143	7.3333
	S	4.4752	3.9616	0	5.1316	5.3452	6.6583
	N*	0	0	0	0	0	0
B.m.l.p.e.dife	$\bar{\mathbf{X}}$	3.6957	7.5556	0	2	2.5714	-1.6667
	S	4.258	3.6439	0	1	2.5071	4.7258
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
B.m.l.p.p.post	$\bar{\mathbf{X}}$	8.5217	13.2222	4.5	10.6667	14.7143	10.3333
	S	6.388	4.4378	3.5355	7.0238	4.8206	10.0167
	N*	0	0	0	0	0	0
B.m.l.p.p.dife	$\bar{\mathbf{X}}$	7.6522	8.4444	4.5	2.6667	-0.5714	-2.3333
	S	6.2929	5.615	3.5355	1.5275	2.5728	5.8595
	N*	0	0	0	0	0	0
B.aprend.post	$\bar{\mathbf{X}}$	33.0435	57.6667	35.5	71.3333	66.8571	56.3333
	S	26.2115	8.1394	20.5061	9.0185	15.3994	23.6714
	N*	0	0	0	0	0	0
B.aprend.dife	$\bar{\mathbf{X}}$	28.1739	37.5556	35.5	12.6667	7.5714	11.3333
	S	25.7676	26.2541	20.5061	7.0238	8.0178	16.2583
	N*	0	0	0	0	0	0
B.v.rev.post	$\bar{\mathbf{X}}$	3.4348	7	6.5	7.3333	8.4286	6.3333
	S	3.8356	3.0414	2.1213	3.7859	1.1339	2.0817
	N*	0	0	0	0	0	0
B.v.rev.dife	$\bar{\mathbf{X}}$	2.6522	5.5556	6.5	-1.6667	0.5714	4
	S	3.4982	3.8115	2.1213	2.8868	0.5345	3
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
TMT.B.post	$\bar{\mathbf{X}}$	500	134.1111	254	206	114.8571	350
	S	0	47.7845	1.4142	11.5326	42.6944	141.0674
	N*	0	0	0	0	0	0
TMT.B.dife	$\bar{\mathbf{X}}$	0	-365.8889	-246	-148.6667	-15.5714	-83.3333
	S	0	47.7845	1.4142	47.5219	28.2185	85.049
	N*	0	0	0	0	0	0
Stroop.int.post	$\bar{\mathbf{X}}$	-14.8274	-0.2167	-1.75	11.5933	2.0557	-12.5033
	S	14.6533	5.15	11.6673	6.1228	11.0188	13.914
	N*	0	0	0	0	0	0
Stroop.int.dife	$\bar{\mathbf{X}}$	10.1726	24.7833	23.25	-0.11	3.5871	4.2
	S	14.6533	5.15	11.6673	3.6344	11.3046	5.1884
	N*	0	0	0	0	0	0
WCST.cat.post	$\bar{\mathbf{X}}$	1.2609	2.4444	3	4	2.7143	2
	S	2.3783	2.4552	4.2426	2	2.7516	3.4641
	N*	0	0	0	0	0	0
WCST.cat.dife	$\bar{\mathbf{X}}$	1.2609	2.4444	3	1.3333	1.5714	0
	S	2.3783	2.4552	4.2426	1.5275	2.4398	0
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
WCST.e.pe.post	$\bar{\mathbf{X}}$	40.4348	34.6667	3	25	34	17.3333
	S	17.6734	16.7332	4.2426	9.5394	17.9815	22.6789
	N*	0	0	0	0	0	0
WCST.e.pe.dife	$\bar{\mathbf{X}}$	-9.5652	-15.3333	-47	-10.3333	-7.1429	-18.3333
	S	17.6734	16.7332	4.2426	5.7735	14.9491	27.7909
	N*	0	0	0	0	0	0
rep.p.post	$\bar{\mathbf{X}}$	9.3913	10	10	10	10	10
	S	1.8275	0	0	0	0	0
	N*	0	0	0	0	0	0
rep.p.dife	$\bar{\mathbf{X}}$	0.5652	0	0	0	0	0
	S	1.9028	0	0	0	0	0
	N*	0	0	0	0	0	0
den.vv.post	$\bar{\mathbf{X}}$	12.2174	14	14	14	14	14
	S	3.7774	0	0	0	0	0
	N*	0	0	0	0	0	0
den.vv.dife	$\bar{\mathbf{X}}$	1.1739	0	0	0	0	0
	S	2.7411	0	0	0	0	0
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 7$	$n_c = 3$
comp.p.post	$\bar{\mathbf{X}}$	10.7826	12	12	12	12	12
	S	3.4504	0	0	0	0	0
	N*	0	0	0	0	0	0
comp.p.dife	$\bar{\mathbf{X}}$	0.8696	0	0	0	0	0
	S	2.881	0	0	0	0	0
	N*	0	0	0	0	0	0
comp.ord.post	$\bar{\mathbf{X}}$	14	16	16	16	16	16
	S	4.7386	0	0	0	0	0
	N*	0	0	0	0	0	0
comp.ord.dife	$\bar{\mathbf{X}}$	0.7391	0	0	0	0	0
	S	2.2807	0	0	0	0	0
	N*	0	0	0	0	0	0

Tabla B.12: Descriptiva por grupos ejecución DBSCAN ($\varepsilon = 82$, $\nu = 2$) para Guttmann.

B.7. Descriptiva por grupos con OPTICS ($\varepsilon = 82$, $\nu = 2$)

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
TAS.omis.post	$\bar{\mathbf{X}}$	7.6087	2.1111	5.5	4.3333	0.3333	5.5
	S	3.6648	3.1798	6.364	1.5275	0.5164	3.873
	N*	0	0	0	0	0	0
TAS.omis.dife	$\bar{\mathbf{X}}$	-2.3913	-6.2222	-4.5	-1.3333	-1.8333	-1.75
	S	3.6648	3.7006	6.364	2.0817	4.0702	2.8723
	N*	0	0	0	0	0	0
TAS.error.post	$\bar{\mathbf{X}}$	7.1739	2	5	1.3333	0.5	3.5
	S	4.1082	3.2787	7.0711	1.5275	0.5477	4.5092
	N*	0	0	0	0	0	0
TAS.error.dife	$\bar{\mathbf{X}}$	-2.8261	-7.1111	-5	-1.3333	-1.5	-2
	S	4.1082	4.285	7.0711	1.1547	4.1833	5.5976
	N*	0	0	0	0	0	0
Stroop.p.c.post	$\bar{\mathbf{X}}$	9.1304	31.8889	30.5	35	36.5	22
	S	13.9522	9.0477	2.1213	5	15.6939	19.7821
	N*	0	0	0	0	0	0
Stroop.p.c.dife	$\bar{\mathbf{X}}$	9.1304	31.8889	30.5	7.3333	6.1667	9.25
	S	13.9522	9.0477	2.1213	10.7858	15.0654	11.5289
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
TMA.A.post	$\bar{\mathbf{X}}$	300	64	70	83.6667	51.6667	120.25
	S	0	26.0336	24.0416	18.8768	12.5486	68.4221
	N*	0	0	0	0	0	0
TMT.A.dife	$\bar{\mathbf{X}}$	0	-236	-230	-5.3333	-2.8333	-55.75
	S	0	26.0336	24.0416	13.3167	14.0345	110.3279
	N*	0	0	0	0	0	0
B.d.d.post	$\bar{\mathbf{X}}$	4.2174	5.6667	5	6.3333	6.6667	6.5
	S	2.4855	0.7071	2.8284	1.1547	1.2111	1.291
	N*	0	0	0	0	0	0
B.d.d.dife	$\bar{\mathbf{X}}$	3.7391	3.8889	5	1	0.3333	2
	S	2.6149	2.7588	2.8284	1	0.8165	2.9439
	N*	0	0	0	0	0	0
B.d.i.post	$\bar{\mathbf{X}}$	2.4348	3.5556	3	4	4.1667	4.25
	S	1.4405	0.8819	1.4142	1	0.4082	0.5
	N*	0	0	0	0	0	0
B.d.i.dife	$\bar{\mathbf{X}}$	2.087	2.4444	3	1	0.5	1.75
	S	1.6213	2.0069	1.4142	1	0.5477	1.7078
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
B.m.c.e.p.post	$\bar{\mathbf{X}}$	6.4783	12.1111	5.5	11	12.5	9.25
	S	4.5912	2.8916	2.1213	4.5826	4.2308	4.9244
	N*	0	0	0	0	0	0
B.m.c.e.p.dife	$\bar{\mathbf{X}}$	5.8261	8.2222	5.5	2.3333	0.1667	0.25
	S	4.2816	4.5491	2.1213	0.5774	2.7869	3.594
	N*	0	0	0	0	0	0
B.m.c.p.p.post	$\bar{\mathbf{X}}$	9.6087	15.4444	10.5	14.3333	16.6667	15
	S	6.5833	3.5746	3.5355	5.5076	5.164	5.3541
	N*	0	0	0	0	0	0
B.m.c.p.p.dife	$\bar{\mathbf{X}}$	8.6087	9.2222	10.5	3.6667	-0.1667	1.5
	S	6.6246	7.12	3.5355	1.5275	2.137	9.8489
	N*	0	0	0	0	0	0
B.m.l.p.e.post	$\bar{\mathbf{X}}$	4.1304	8.7778	0	7.6667	10.3333	7
	S	4.4752	3.9616	0	5.1316	5.5737	5.4772
	N*	0	0	0	0	0	0
B.m.l.p.e.dife	$\bar{\mathbf{X}}$	3.6957	7.5556	0	2	2	0.25
	S	4.258	3.6439	0	1	2.1909	5.4391
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
B.m.l.p.p.post	$\bar{\mathbf{X}}$	8.5217	13.2222	4.5	10.6667	15.6667	10
	S	6.388	4.4378	3.5355	7.0238	4.5019	8.2057
	N*	0	0	0	0	0	0
B.m.l.p.p.dife	$\bar{\mathbf{X}}$	7.6522	8.4444	4.5	2.6667	0.3333	-3.25
	S	6.2929	5.615	3.5355	1.5275	1.0328	5.1235
	N*	0	0	0	0	0	0
B.aprend.post	$\bar{\mathbf{X}}$	33.0435	57.6667	35.5	71.3333	67.3333	58.25
	S	26.2115	8.1394	20.5061	9.0185	16.8127	19.7041
	N*	0	0	0	0	0	0
B.aprend.dife	$\bar{\mathbf{X}}$	28.1739	37.5556	35.5	12.6667	8.3333	9.25
	S	25.7676	26.2541	20.5061	7.0238	8.501	13.9134
	N*	0	0	0	0	0	0
B.v.rev.post	$\bar{\mathbf{X}}$	3.4348	7	6.5	7.3333	8.3333	7
	S	3.8356	3.0414	2.1213	3.7859	1.2111	2.1602
	N*	0	0	0	0	0	0
B.v.rev.dife	$\bar{\mathbf{X}}$	2.6522	5.5556	6.5	-1.6667	0.6667	3
	S	3.4982	3.8115	2.1213	2.8868	0.5164	3.1623
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
TMT.B.post	$\bar{\mathbf{X}}$	500	134.1111	254	206	122.3333	280
	S	0	47.7845	1.4142	11.5326	41.4472	181.2917
	N*	0	0	0	0	0	0
TMT.B.dife	$\bar{\mathbf{X}}$	0	-365.8889	-246	-148.6667	-8.3333	-77.25
	S	0	47.7845	1.4142	47.5219	22.7039	70.5
	N*	0	0	0	0	0	0
Stroop.int.post	$\bar{\mathbf{X}}$	-14.8274	-0.2167	-1.75	11.5933	1.9717	-8.7375
	S	14.6533	5.15	11.6673	6.1228	12.068	13.6306
	N*	0	0	0	0	0	0
Stroop.int.dife	$\bar{\mathbf{X}}$	10.1726	24.7833	23.25	-0.11	3.8167	3.7025
	S	14.6533	5.15	11.6673	3.6344	12.3657	4.3516
	N*	0	0	0	0	0	0
WCST.cat.post	$\bar{\mathbf{X}}$	1.2609	2.4444	3	4	2.1667	3
	S	2.3783	2.4552	4.2426	2	2.5626	3.4641
	N*	0	0	0	0	0	0
WCST.cat.dife	$\bar{\mathbf{X}}$	1.2609	2.4444	3	1.3333	1.6667	0.25
	S	2.3783	2.4552	4.2426	1.5275	2.6583	0.5
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
WCST.e.pe.post	$\bar{\mathbf{X}}$	40.4348	34.6667	3	25	38.5	14.75
	S	17.6734	16.7332	4.2426	9.5394	14.7614	19.2246
	N*	0	0	0	0	0	0
WCST.e.pe.dife	$\bar{\mathbf{X}}$	-9.5652	-15.3333	-47	-10.3333	-8	-14.25
	S	17.6734	16.7332	4.2426	5.7735	16.1864	24.116
	N*	0	0	0	0	0	0
rep.p.post	$\bar{\mathbf{X}}$	9.3913	10	10	10	10	10
	S	1.8275	0	0	0	0	0
	N*	0	0	0	0	0	0
rep.p.dife	$\bar{\mathbf{X}}$	0.5652	0	0	0	0	0
	S	1.9028	0	0	0	0	0
	N*	0	0	0	0	0	0
den.vv.post	$\bar{\mathbf{X}}$	12.2174	14	14	14	14	14
	S	3.7774	0	0	0	0	0
	N*	0	0	0	0	0	0
den.vv.dife	$\bar{\mathbf{X}}$	1.1739	0	0	0	0	0
	S	2.7411	0	0	0	0	0
	N*	0	0	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**C3**	**C4**	**NOISE**
VARIABLE	N = 47	$n_c = 23$	$n_c = 9$	$n_c = 2$	$n_c = 3$	$n_c = 6$	$n_c = 4$
comp.p.post	$\bar{\mathbf{X}}$	10.7826	12	12	12	12	12
	S	3.4504	0	0	0	0	0
	N*	0	0	0	0	0	0
comp.p.dife	$\bar{\mathbf{X}}$	0.8696	0	0	0	0	0
	S	2.881	0	0	0	0	0
	N*	0	0	0	0	0	0
comp.ord.post	$\bar{\mathbf{X}}$	14	16	16	16	16	16
	S	4.7386	0	0	0	0	0
	N*	0	0	0	0	0	0
comp.ord.dife	$\bar{\mathbf{X}}$	0.7391	0	0	0	0	0
	S	2.2807	0	0	0	0	0
	N*	0	0	0	0	0	0

Tabla B.13: Descriptiva por grupos ejecución OPTICS ($\varepsilon = 82$, $\nu = 2$) para Guttmann.

B.8. Descriptiva por grupos con DBSCAN ($\varepsilon = 1{,}5$, $\nu = 4$) datos Mixtos

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
TAS.omis.post	$\bar{\mathbf{X}}$	8.6429	1.8182	1.6667	5.6875
	S	2.4054	2.0889	1.0328	4.743
	N*	0	0	0	0
TAS.omis.dife	$\bar{\mathbf{X}}$	-1.3571	-2	-8.3333	-3.1875
	S	2.4054	3.3166	1.0328	4.3239
	N*	0	0	0	0
TAS.error.post	$\bar{\mathbf{X}}$	7.9286	0.7273	1.1667	5.5
	S	3.5619	0.9045	1.6021	4.7329
	N*	0	0	0	0
TAS.error.dife	$\bar{\mathbf{X}}$	-2.0714	-2.1818	-8.8333	-3.375
	S	3.5619	3.9703	1.6021	5.0183
	N*	0	0	0	0
Stroop.p.c.post	$\bar{\mathbf{X}}$	10.1429	36.0909	28.6667	16.1875
	S	15.7522	12.2756	7.1181	16.4407
	N*	0	0	0	0
Stroop.p.c.dife	$\bar{\mathbf{X}}$	10.1429	8.3636	28.6667	15.5
	S	15.7522	13.0864	7.1181	16.7212
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
TMA.A.post	$\bar{\mathbf{X}}$	286.0714	64.7273	58.1667	224.5
	S	52.1159	24.1168	22.7633	106.7221
	N*	0	0	0	0
TMT.A.dife	$\bar{\mathbf{X}}$	-13.9286	-20.5455	-241.8333	-60.5
	S	52.1159	58.377	22.7633	107.9148
	N*	0	0	0	0
B.d.d.post	$\bar{\mathbf{X}}$	5.3571	6.7273	6	3.625
	S	1.1507	1.1037	0.8944	2.6802
	N*	0	0	0	0
B.d.d.dife	$\bar{\mathbf{X}}$	5.3571	0.7273	6	1.5625
	S	1.1507	0.9045	0.8944	2.3936
	N*	0	0	0	0
B.d.i.post	$\bar{\mathbf{X}}$	3	4.1818	4	2.25
	S	0.6794	0.603	0.6325	1.6931
	N*	0	0	0	0
B.d.i.dife	$\bar{\mathbf{X}}$	3	0.8182	4	0.875
	S	0.6794	0.7508	0.6325	1.5864
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
B.m.c.e.p.post	$\bar{\mathbf{X}}$	7.5714	12.0909	10.5	7
	S	3.1796	3.8589	3.0166	6.0332
	N*	0	0	0	0
B.m.c.e.p.dife	$\bar{\mathbf{X}}$	7.5714	1	10.5	3
	S	3.1796	2.2361	3.0166	4.2583
	N*	0	0	0	0
B.m.c.p.p.post	$\bar{\mathbf{X}}$	12.0714	16.3636	13.5	9.625
	S	4.9219	4.9045	2.51	7.6322
	N*	0	0	0	0
B.m.c.p.p.dife	$\bar{\mathbf{X}}$	12.0714	1.0909	13.5	3.5
	S	4.9219	2.4271	2.51	6.802
	N*	0	0	0	0
B.m.l.p.e.post	$\bar{\mathbf{X}}$	4	9.4545	6.6667	5.4375
	S	3.6162	4.9672	4.8854	5.6565
	N*	0	0	0	0
B.m.l.p.e.dife	$\bar{\mathbf{X}}$	4	2.3636	6.6667	3.125
	S	3.6162	2.0136	4.8854	5.4513
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
B.m.l.p.p.post	$\bar{\mathbf{X}}$	10.2143	14.0909	10.8333	7.9375
	S	5.041	5.5759	4.0702	7.6895
	N*	0	0	0	0
B.m.l.p.p.dife	$\bar{\mathbf{X}}$	10.2143	0.5455	10.8333	2.75
	S	5.041	2.6216	4.0702	6.0388
	N*	0	0	0	0
B.aprend.post	$\bar{\mathbf{X}}$	40	68.3636	55.3333	34.8125
	S	21.7645	12.7693	6.8896	29.2261
	N*	0	0	0	0
B.aprend.dife	$\bar{\mathbf{X}}$	40	9	55.3333	11.9375
	S	21.7645	7.3485	6.8896	19.0034
	N*	0	0	0	0
B.v.rev.post	$\bar{\mathbf{X}}$	3.7857	8	7.8333	4.1875
	S	3.534	2	0.9832	4.1828
	N*	0	0	0	0
B.v.rev.dife	$\bar{\mathbf{X}}$	3.7857	0.5455	7.8333	1.8125
	S	3.534	2.7336	0.9832	3.0815
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
TMT.B.post	$\bar{\mathbf{X}}$	477.8571	159.2727	151.6667	395.9375
	S	82.851	77.9167	65.1634	164.1059
	N*	0	0	0	0
TMT.B.dife	$\bar{\mathbf{X}}$	-22.1429	-65.9091	-348.3333	-91.5625
	S	82.851	76.4283	65.1634	157.2866
	N*	0	0	0	0
Stroop.int.post	$\bar{\mathbf{X}}$	-15.1136	3.1064	-1.375	-9.3219
	S	13.9378	11.6409	5.2602	15.2821
	N*	0	0	0	0
Stroop.int.dife	$\bar{\mathbf{X}}$	9.8864	3.1618	23.625	14.1225
	S	13.9378	9.3452	5.2602	15.2664
	N*	0	0	0	0
WCST.cat.post	$\bar{\mathbf{X}}$	0.5	2.8182	3.1667	2.3125
	S	1.6053	2.562	2.7142	2.7741
	N*	0	0	0	0
WCST.cat.dife	$\bar{\mathbf{X}}$	0.5	1.3636	3.1667	1.9375
	S	1.6053	2.0627	2.7142	2.645
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
WCST.e.pe.post	$\bar{\mathbf{X}}$	45	32.3636	25.8333	29.5
	S	13.1909	15.5452	20.213	21.7041
	N*	0	0	0	0
WCST.e.pe.dife	$\bar{\mathbf{X}}$	-5	-8	-24.1667	-17.8125
	S	13.1909	11.9583	20.213	21.6586
	N*	0	0	0	0
rep.p.post	$\bar{\mathbf{X}}$	10	10	10	9.125
	S	0	0	0	2.1564
	N*	0	0	0	0
rep.p.dife	$\bar{\mathbf{X}}$	0	0	0	0.8125
	S	0	0	0	2.2574
	N*	0	0	0	0
den.vv.post	$\bar{\mathbf{X}}$	14	14	14	11.4375
	S	0	0	0	4.3354
	N*	0	0	0	0
den.vv.dife	$\bar{\mathbf{X}}$	0	0	0	1.6875
	S	0	0	0	3.1774
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 11$	$n_c = 6$	$n_c = 16$
comp.p.post	$\bar{\mathbf{X}}$	12	12	12	10.25
	S	0	0	0	4.0579
	N*	0	0	0	0
comp.p.dife	$\bar{\mathbf{X}}$	0	0	0	1.25
	S	0	0	0	3.4157
	N*	0	0	0	0
comp.ord.post	$\bar{\mathbf{X}}$	16	16	16	13.125
	S	0	0	0	5.5
	N*	0	0	0	0
comp.ord.dife	$\bar{\mathbf{X}}$	0	0	0	1.0625
	S	0	0	0	2.6949
	N*	0	0	0	0

Tabla B.14: Descriptiva por grupos ejecución DBSCAN (ε = 1,5, ν = 4) para Guttmann con datos Mixtos.

B.9. Descriptiva por grupos con OPTICS ($\varepsilon = 1{,}5$, $\nu = 4$) datos Mixtos

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
TAS.omis.post	$\bar{\mathbf{X}}$	8.6429	1.6	1.8	5.3333
	S	2.4054	2.0656	1.0954	4.6018
	N*	0	0	0	0
TAS.omis.dife	$\bar{\mathbf{X}}$	-1.3571	-1.6	-8.2	-3.6667
	S	2.4054	3.2042	1.0954	4.325
	N*	0	0	0	0
TAS.error.post	$\bar{\mathbf{X}}$	7.9286	0.8	1.4	4.8889
	S	3.5619	0.9189	1.6733	4.7883
	N*	0	0	0	0
TAS.error.dife	$\bar{\mathbf{X}}$	-2.0714	-1.4	-8.6	-4.1111
	S	3.5619	3.1693	1.6733	5.1779
	N*	0	0	0	0
Stroop.p.c.post	$\bar{\mathbf{X}}$	10.1429	37.1	28	17.6111
	S	15.7522	12.4495	7.746	16.0226
	N*	0	0	0	0
Stroop.p.c.dife	$\bar{\mathbf{X}}$	10.1429	6.6	28	17
	S	15.7522	12.3396	7.746	16.3347
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
TMA.A.post	$\bar{\mathbf{X}}$	286.0714	60.5	59.2	208.4444
	S	52.1159	20.6841	25.2923	110.9908
	N*	0	0	0	0
TMT.A.dife	$\bar{\mathbf{X}}$	-13.9286	-3.3	-240.8	-78.2222
	S	52.1159	12.3112	25.2923	114.1127
	N*	0	0	0	0
B.d.d.post	$\bar{\mathbf{X}}$	5.3571	6.6	5.8	4.0556
	S	1.1507	1.075	0.8367	2.8174
	N*	0	0	0	0
B.d.d.dife	$\bar{\mathbf{X}}$	5.3571	0.6	5.8	1.8889
	S	1.1507	0.8433	0.8367	2.587
	N*	0	0	0	0
B.d.i.post	$\bar{\mathbf{X}}$	3	4.2	4	2.4444
	S	0.6794	0.6325	0.7071	1.6881
	N*	0	0	0	0
B.d.i.dife	$\bar{\mathbf{X}}$	3	0.8	4	1.0556
	S	0.6794	0.7888	0.7071	1.6618
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
B.m.c.e.p.post	$\bar{\mathbf{X}}$	7.5714	11.8	11.2	7.4444
	S	3.1796	3.9384	2.7749	5.9727
	N*	0	0	0	0
B.m.c.e.p.dife	$\bar{\mathbf{X}}$	7.5714	1	11.2	3.1111
	S	3.1796	2.357	2.7749	4.1429
	N*	0	0	0	0
B.m.c.p.p.post	$\bar{\mathbf{X}}$	12.0714	15.8	13.6	10.5
	S	4.9219	4.7796	2.7928	7.763
	N*	0	0	0	0
B.m.c.p.p.dife	$\bar{\mathbf{X}}$	12.0714	1	13.6	3.9444
	S	4.9219	2.5386	2.7928	6.7864
	N*	0	0	0	0
B.m.l.p.e.post	$\bar{\mathbf{X}}$	4	9.1	8	5.5556
	S	3.6162	5.087	4.062	5.7724
	N*	0	0	0	0
B.m.l.p.e.dife	$\bar{\mathbf{X}}$	4	2.4	8	2.8889
	S	3.6162	2.1187	4.062	5.1779
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
B.m.l.p.p.post	$\bar{\mathbf{X}}$	10.2143	13.5	11.6	8.5556
	S	5.041	5.5025	4.0373	7.7704
	N*	0	0	0	0
B.m.l.p.p.dife	$\bar{\mathbf{X}}$	10.2143	0.4	11.6	2.9444
	S	5.041	2.7162	4.0373	5.7647
	N*	0	0	0	0
B.aprend.post	$\bar{\mathbf{X}}$	40	68.2	56.4	37.6111
	S	21.7645	13.4478	7.1274	28.8406
	N*	0	0	0	0
B.aprend.dife	$\bar{\mathbf{X}}$	40	9.1	56.4	13.8333
	S	21.7645	7.7381	7.1274	20.0243
	N*	0	0	0	0
B.v.rev.post	$\bar{\mathbf{X}}$	3.7857	8.1	7.8	4.5556
	S	3.534	2.079	1.0954	4.0761
	N*	0	0	0	0
B.v.rev.dife	$\bar{\mathbf{X}}$	3.7857	-0.1	7.8	2.4444
	S	3.534	1.792	1.0954	3.4338
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
TMT.B.post	$\bar{\mathbf{X}}$	477.8571	142.2	131.4	384.3333
	S	82.851	56.4187	47.1943	158.3585
	N*	0	0	0	0
TMT.B.dife	$\bar{\mathbf{X}}$	-22.1429	-55.5	-368.6	-104.5556
	S	82.851	71.8753	47.1943	153.0783
	N*	0	0	0	0
Stroop.int.post	$\bar{\mathbf{X}}$	-15.1136	4.917	0.35	-9.675
	S	13.9378	10.5118	3.5028	14.4173
	N*	0	0	0	0
Stroop.int.dife	$\bar{\mathbf{X}}$	9.8864	2.478	25.35	13.9422
	S	13.9378	9.5562	3.5028	14.3755
	N*	0	0	0	0
WCST.cat.post	$\bar{\mathbf{X}}$	0.5	3.1	2.6	2.3889
	S	1.6053	2.5144	2.6077	2.8105
	N*	0	0	0	0
WCST.cat.dife	$\bar{\mathbf{X}}$	0.5	1.5	2.6	2.0556
	S	1.6053	2.1213	2.6077	2.711
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
WCST.e.pe.post	$\bar{\mathbf{X}}$	45	31.3	29.8	28.9444
	S	13.1909	15.9586	19.8167	21.4132
	N*	0	0	0	0
WCST.e.pe.dife	$\bar{\mathbf{X}}$	-5	-8.1	-20.2	-18.6667
	S	13.1909	12.6003	19.8167	21.4558
	N*	0	0	0	0
rep.p.post	$\bar{\mathbf{X}}$	10	10	10	9.2222
	S	0	0	0	2.0452
	N*	0	0	0	0
rep.p.dife	$\bar{\mathbf{X}}$	0	0	0	0.7222
	S	0	0	0	2.1367
	N*	0	0	0	0
den.vv.post	$\bar{\mathbf{X}}$	14	14	14	11.7222
	S	0	0	0	4.1559
	N*	0	0	0	0
den.vv.dife	$\bar{\mathbf{X}}$	0	0	0	1.5
	S	0	0	0	3.0341
	N*	0	0	0	0

	CLASSE	**C0**	**C1**	**C2**	**NOISE**
VARIABLE	N = 47	$n_c = 14$	$n_c = 10$	$n_c = 5$	$n_c = 18$
comp.p.post	$\bar{\mathbf{X}}$	12	12	12	10.4444
	S	0	0	0	3.8535
	N*	0	0	0	0
comp.p.dife	$\bar{\mathbf{X}}$	0	0	0	1.1111
	S	0	0	0	3.2338
	N*	0	0	0	0
comp.ord.post	$\bar{\mathbf{X}}$	16	16	16	13.4444
	S	0	0	0	5.2493
	N*	0	0	0	0
comp.ord.dife	$\bar{\mathbf{X}}$	0	0	0	0.9444
	S	0	0	0	2.5546
	N*	0	0	0	0

Tabla B.15: Descriptiva por grupos ejecución OPTICS ($\varepsilon = 1{,}5$, $\nu = 4$) para Guttmann con datos Mixtos.