

**APRENDIZAJE NO SUPERVISADO**  
**MÁSTER EN INTELIGENCIA ARTIFICIAL**  
**UNIVERSIDAD INTERNACIONAL DE VALENCIA**

## TÉCNICAS DE CLUSTERING

Componentes del grupo:

**Coordinadora:** Carmen Raposo Jiménez  
**Secretario:** Víctor Dot Mariano  
**Revisor:** Ricardo Sánchez Pastor

7 Abril de 2019

# Índice general

<b>1. Conjuntos de Datos</b>	<b>1</b>
1.1. Agrupamiento Real Conocido . . . . .	1
1.2. Agrupamiento Real No Conocido . . . . .	1
<b>2. Métricas de evaluación</b>	<b>3</b>
2.1. Medidas de evaluación extrínsecas . . . . .	3
2.2. Medidas de evaluación intrínsecas . . . . .	4
<b>3. Algoritmos de clustering</b>	<b>5</b>
3.1. KMeans . . . . .	5
3.2. Aproximación aglomerativa . . . . .	6
3.3. Espectral . . . . .	8
3.4. DBSCAN . . . . .	8
3.5. Modelos de mixturas . . . . .	10
<b>4. Pruebas realizadas</b>	<b>12</b>
4.1. Dataset Iris . . . . .	12
4.1.1. KMeans . . . . .	13
4.1.2. Aproximación aglomerativa . . . . .	14
4.1.3. Espectral . . . . .	15
4.1.4. DBSCAN . . . . .	16
4.1.5. Modelos de mixturas . . . . .	17
4.1.6. Mejor algoritmo . . . . .	18
4.2. Dataset Wine . . . . .	19
4.2.1. KMeans . . . . .	19
4.2.2. Aproximación aglomerativa . . . . .	20
4.2.3. Espectral . . . . .	21
4.2.4. DBSCAN . . . . .	22
4.2.5. Modelos de mixturas . . . . .	23
4.2.6. Mejor algoritmo . . . . .	24

## Apartado 1

# Conjuntos de Datos

Se han escogido dos conjuntos de datos distintos para las dos casuísticas planteadas: uno conociendo la clasificación real de los datos y otro sin saberlo.

### 1.1. Agrupamiento Real Conocido

Como dataset para el problema cuyo agrupamiento real es conocido se ha seleccionado el ya cargado en las librerías de sklearn, **iris**, el cual categoriza distintos tipos de iris en tres especies. Este conjunto de datos tiene las siguiente características:

Número de instancias	150
Número de atributos	4
Descripción de atributos	sepal-length. Numérico. Longitud del sépalo. sepal-width. Numérico. Anchura del sépalo. petal-length. Numérico. Longitud del pétalo. petal-width. Numérico. Anchura del pétalo.
Tipo de datos de la clase	Categorico: iris-setosa, iris-versicolor, iris-virginica
Descripción de la clase	Nombre: class. Descripción: Especie de la planta.
Valores ausentes	0

De los cuatro atributos que posee este dataset, nuestro problema se centrará en los tres primeros, por lo que trabajaremos en un espacio de tres dimensiones para nuestros algoritmos.

### 1.2. Agrupamiento Real No Conocido

En este caso, se ha seleccionado otro dataset ya cargado en las librerías sklearn, **wine**, con sus datos de agrupación reales conocidos, aunque nosotros para nuestro problema los **obviaremos los datos de agrupación**, como si no existiesen.

Las características de este dataset son las siguientes:

<b>Número de instancias</b>	178
<b>Número de atributos</b>	13
<b>Descripción de atributos</b>	Alcohol. Numérico. Graduación del alcohol. Malic acid. Numérico. Concentración de ácido málico g/l Ash. Numérico. Materia inorgánica g/l Alkalinity of ash. Numérico. Alcalinidad de la materia orgánica. Magenium. Numérico. Magnesio mg/l Total phenols. Numérico. Total de fenoles Flavanoids. Numérico. Flavanoides. Nonflavanoid phenols. Numérico. Fenoles no flavonoides. Proanthocyanins. Numérico. Proantocianinas. Color intensity. Numérico. Intensidad del color. Hue. Numérico. Matiz. OD280/OD315. Numérico. Medida del contenido proteico. Proline. Numérico. Aminoácido.
<b>Tipo de datos de la clase</b>	Numérico: 1,2,3
<b>Descripción de la clase</b>	Nombre: class. Descripción: Tres zonas distintas de cultivo.
<b>Valores ausentes</b>	0

Como con el dataset anterior, se trabajarán sólo con tres atributos de los trece presentados, por lo que nos moveremos en un entorno tridimensional una vez más.

## Apartado 2

# Métricas de evaluación

Teniendo en cuenta la naturaleza del trabajo, necesitaremos definir dos conjuntos de métricas, unas para el problema con datos de agrupación real conocidos y otras para los no conocidos. De esta forma, las métricas quedan divididas en: **extrínsecas** e **intrínsecas**.

### 2.1. Medidas de evaluación extrínsecas

Las medidas escogidas para evaluar los algoritmos de agrupación de datos reales conocidos han sido tres:

- **Error** : Estudia el número medio de casos de errores que se han cometido, representándose por la siguiente fórmula:

$$E = 1 - \frac{1}{n} \max_{\sigma} \sum_{l=1}^{K'} n_{\sigma(l)l}$$

donde la función sigma asigna a cada clúster original un clúster predicho.

- **Pureza** : Se entiende como la precisión promedio, donde se calcula la precisión media de los clústeres predichos. Para ello, presentemos la siguiente fórmula:

$$Pu = \sum_{k=1}^K \frac{n_{k\cdot}}{n} \max_{l \in \{1, \dots, K'\}} P_{kl}$$

donde  $P_{kl}$  representa la precisión, es decir, la proporción de casos sobre un clúster predicho que pertenecen a la mismo clúster que al predicho.

- **F1** : A diferencia de las medidas anteriores esta representa el promedio ponderado de la media armónica de la precisión y el recall, siendo esta última medida equivalente a la precisión pero sobre un clúster real. Su expresión es la siguiente:

$$F1 = \sum_{l=1}^{K'} \frac{n_l}{n} \max_{k \in \{1, \dots, K\}} \left( \frac{2P_{kl}R_{lk}}{P_{kl} + R_{lk}} \right)$$

## 2.2. Medidas de evaluación intrínsecas

Para este caso se han escogido otras tres métricas, a diferencia de que estas pueden ser aplicadas para ambos problemas. Son las siguientes:

- **RMSSTD** : Es la raíz del cuadrado de la media de la desviación típica, de ahí las siglas en inglés RMSSTD, y su expresión matemática es:

$$RMSSTD = \sqrt{\frac{\sum_{k=1}^K \sum_{x_i \in C_k} x_i - c_k^2}{v \cdot \sum_{k=1}^K (|C_k| - 1)}}$$

Estudia la homogeneidad de los clústeres del agrupamiento sin tener en cuenta la distancia interclúster.

- **Ancho de silueta** : Esta métrica supone una diferencia normalizada de la distancia interclúster menos la distancia intraclúster, siendo su fórmula:

$$S = \frac{1}{K} \sum_{k=1}^K \frac{1}{|C_k|} \sum_{x_i \in C_k} \frac{b_k(x_i) - a_k(x_i)}{\max\{a_k(x_i), b_k(x_i)\}}$$

donde

$$a_k(x_i) = \frac{1}{|C_k| - 1} \sum_{x_{i'} \in C_k: x_{i'} \neq x_i} d(x_i, x_{i'}) \quad b_k(x_i) = \min_{h \neq k} \frac{1}{|C_h|} \sum_{x_{i'} \in C_h} d(x_i, x_{i'})$$

- **Índice Calinski-Harabasz** : Evalúa la bondad de un agrupamiento basado en la suma promedio de distancias interclúster e intraclúster al cuadrado, más concretamente de la siguiente forma:

$$iCH = \frac{(n - K) \sum_{k=1}^K |C_k| \cdot d(c_k, c)^2}{(K - 1) \sum_{k=1}^K \sum_{x_i \in C_k} d(x_i, c_k)^2}$$

## Apartado 3

# Algoritmos de clustering

En el ámbito del clustering, existen infinidad de técnicas y algoritmos de agrupamiento, pudiéndose hacer una amplia clasificación de los mismos en cinco grupos:

- Algoritmos de agrupamiento basados en particiones
- Agrupamiento jerárquico
- Agrupamiento espectral
- Agrupamiento basado en densidad
- Agrupamiento basado en modelos probabilísticos

En este trabajo se aplican cinco algoritmos de agrupación en total, perteneciendo cada uno a un grupo de los antes mencionados.

### 3.1. KMeans

K means es el algoritmo de clusterización más utilizado por su sencillez. Forma parte de los algoritmos basados en particiones. Este algoritmo es un método iterativo que utilizando un número K de clústeres como parámetro de entrada, posiciona K centroides de forma que maximiza la dispersión interclúster y minimiza la dispersión intraclúster (Ambos objetivos son equivalentes).

Dispersión intraclúster (minimizar):

$$I(C) = \frac{1}{2} \sum_{k=1}^K \sum_{i: C(x_i)=k} \sum_{i': C(x_{i'})=k} d(x_i, x_{i'})$$

Dispersión interclúster (maximizar):

$$O(C) = \frac{1}{2} \sum_{k=1}^K \sum_{i: C(x_i)=k} \sum_{i': C(x_{i'}) \neq k} d(x_i, x_{i'})$$

El pseudocódigo de este algoritmo es:

1. Elección K puntos del conjunto del dataset de forma aleatoria como centro del clúster.
2. Asignar a cada instancia del dataset al clúster más cercano
3. Para cada clúster k se recalcula el centro que minimiza la dispersión intracúster
4. Se repite el paso 2 y 3 hasta la convergencia

Este algoritmo se ha hecho muy popular debido a su sencillez y rapidez sin embargo tiene las siguientes desventajas:

- El número de clústeres es un parámetro
- Depende de los puntos de inicialización
- Es sensible a puntos outliers
- No funciona con variables discretas
- Tiene problemas para identificar clústeres convexos o de diferentes tamaños/ densidades

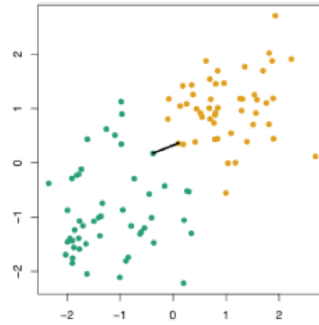
### 3.2. Aproximación aglomerativa

El algoritmo de aproximación aglomerativa es de tipo jerárquico, en el cual se empieza considerando cada una de las instancias del dataset como un clúster individual y de forma iterativa se van uniendo los pares de clústeres con menor disimilitud interclúster hasta terminar en un único clúster.

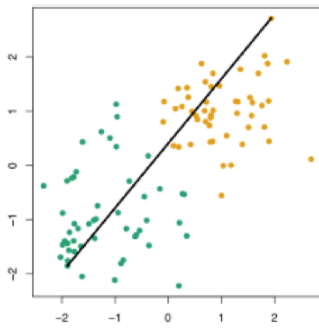
Para el cálculo de disimilitud interclúster se puede usar 3 tipos de criterios distintos:

Minimizar la disimilitud mínima interclúster: consiste en buscar la distancia mínima entre dos clústeres. Utilizar este criterio consigue clústeres alargados.

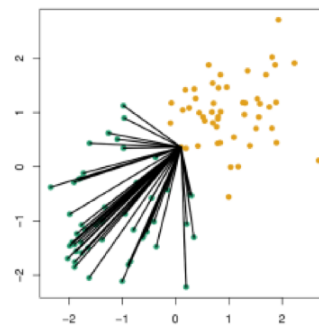




Minimizar la disimilitud máxima interclúster: este criterio utiliza la distancia entre los puntos más alejados de ambos clústeres. La elección de este criterio consigue clúster pequeños y densos.



Minimizar la disimilitud media interclúster.: utiliza la distancia media de todos los puntos de un clúster con otro clúster. El uso de este criterio consigue clústerws intermedios a los obtenidos con los dos criterios anteriores.



Este algoritmo es sencillo y funciona bastante bien con clúster de diferentes tamaños. Sin embargo, es un algoritmo lento y con problemas para distinguir entre clúster de diferentes densidades.

### 3.3. Espectral

El Algoritmo Espectral transforma el dataset a otro espacio vectorial que permita una separación de clústers más sencilla. Para ello lo primero que hace este algoritmo es crear un grafo y genera a partir de ella una matriz de adyacencias.

Esté grafo o matriz de adyacencias se puede construir de 3 formas distintas, ya sea conectando todos los nodos con todos, conectando solo aquellos nodos que estén a una distancia inferior de una distancia umbral o uniendo solo cada nodo con los knn nodos más cercanos. Siendo estas las dos últimas las más utilizadas debido a que suponen un menor coste de cálculo.

Una vez calculada la matriz de adyacencia se transforma los datos para obtener una representación alternativa de los datos. Para ello primero se calcula la matriz Laplaciana,

$$L = D - W$$

Siendo W la matriz del grafo obtenido en el paso anterior y D la matriz diagonal donde  $D_{i,i} = \sum W_{i,j}$ .

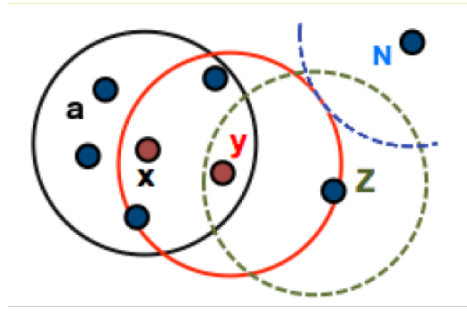
Está matriz laplaciana se descompone en vectores propios y se ordenan por su valor de forma ascendente. Se seleccionan los primeros K vectores propios. Cada uno de los vectores propios será una componente del nuevo espacio vectorial en el cual se puede descomponer cada uno de las instancias del dataset.

Una vez transformado el espacio, se puede utilizar cualquier método de clusterización ( K means por defecto) para dividir el data set en clústeres.

Este algoritmo tiene una gran base matemática que permite poder trabajar con clúster de diversas formas y tamaños y trabaja bien con clústeres convexos. Sin embargo, el número de clústeres es un parámetro de entrada.

### 3.4. DBSCAN

El DBSCAN es un algoritmo basado en densidad que requiere de una serie definiciones que explicaremos a continuación para generar los deferentes clústeres.



Punto nuclear ( $y$ ). Es aquel punto del dataset que tiene un número mínimo de vecinos  $M$  a una distancia inferior  $\epsilon$ .

Un punto directamente denso-alcanzable ( $z$  para  $y$ ) es aquel punto que está a una distancia inferior  $\epsilon$  de un punto nuclear.

Un punto ruido ( $N$ ) es aquel punto que no es directamente denso-alcanzable por ningún punto nuclear.

Un punto denso-alcanzable( $a$  para  $y$ ) es aquel punto que es alcanzable por un punto nuclear pasando por otros puntos nucleares que son directamente alcanzables por el punto nuclear inicial.

Puntos denso conectados ( $z$  y  $a$ ) son todos los puntos que están conectados entre sí a través de puntos nucleares directamente denso-conectados.

Un clúster se formaría a partir de todos los puntos denso-conectados.

El pseudocódigo de este algoritmo sería el siguiente:

1. Para todos los puntos:
  - a) Si el punto ya está asignado pasar a siguiente punto.
  - b) Calcular vecindario  $V$  del punto.
  - c) Si  $V \geq M$  asignar  $c$  como ruido y pasar al siguiente punto.
  - d) Crear un clúster nuevo  $C$  y asignarle el punto.
  - e) Para todos los puntos de  $V$ .
    - 1) Si el punto está asignado como ruido asignarlo al clúster  $C$  y pasar al siguiente punto de  $V$ .
    - 2) Si el punto ya está en otro clúster pasar al siguiente punto.
    - 3) Asignar el punto al clúster  $C$ .
    - 4) Calcular el vecindario  $V'$  del punto.
    - 5) Si  $V > M$  unir  $V$  y  $V'$ .

Este algoritmo tiene la gran ventaja de que no es necesario especificar el número de clúster, funciona perfectamente con clúster de diferentes tamaños y formas, y se puede utilizar con diferentes medidas de distancias. Sin embargo, tiene una definición compleja, le cuesta trabajar con clústeres de diferentes densidades y tiene dos parámetros que ajustar

### 3.5. Modelos de mixturas

Este algoritmo forma parte de los algoritmos de clústerización basados en agrupamientos de probabilidad. En el cual se asume que el dataset ha sido generado por un modelo probabilístico de mixturas gaussianas. Este algoritmo intenta encontrar los parámetros del modelo generador que hubiese generado el dataset.

Para empezar se asume que cada clúster tiene su propia distribución de probabilidad gaussiana, y la probabilidad de un punto es la suma de la probabilidad de las distribuciones gaussianas ( de la mixtura).

Partiendo de aquí, hay que encontrar los parámetros de dicha mixtura. Estos parámetros son:

$Z_{ik}$ : probabilidad de que el punto  $i$  pertenezca al clúster  $K$ .

$$\hat{z}_{ik} = \hat{\pi}_k \cdot \mathcal{N}(\mathbf{x}_i | \hat{\mu}_k, \hat{\Sigma}_k) / \left( \sum_{k'=1}^K \hat{\pi}_{k'} \cdot \mathcal{N}(\mathbf{x}_i | \hat{\mu}_{k'}, \hat{\Sigma}_{k'}) \right)$$

$\pi_k$ : La probabilidad de que un punto genérico pertenezca al clúster  $K$ .

$$\hat{\pi}_k = \sum_{i=1}^n \hat{z}_{ik} / n$$

$\mu_k$ : el valor medio del clúster  $K$ .

$$\hat{\mu}_k = \left( \sum_{i=1}^n \hat{z}_{ik} \cdot \mathbf{x}_i \right) / \left( \sum_{i=1}^n \hat{z}_{ik} \right)$$

$\sigma_k$ : la covarianza del clúster  $K$ .

$$\hat{\Sigma}_k = \left( \sum_{i=1}^n \hat{z}_{ik} (\mathbf{x}_i - \hat{\mu}_k)(\mathbf{x}_i - \hat{\mu}_k)^T \right) / \left( \sum_{i=1}^n \hat{z}_{ik} \right)$$

Para encontrar estos parámetros se utiliza el algoritmo de aprendizaje Esperanza-Maximización (E-M). Este algoritmo es necesario ya que se utiliza como parámetros  $\pi$ ,  $\mu$  y  $\sigma$  y estos 3 parámetros utilizan a  $z$  como variable).

En el paso E de este algoritmo se calcula  $z$  a partir de unos valores de  $\pi$ ,  $\mu$  y  $\sigma$  iniciales.

En el paso M se calcula los valores de  $\pi$ ,  $\mu$  y  $\sigma$  con las  $z$  calculadas en el paso E.

Se repite el paso E y M hasta que se converge.

Una vez ha convergido tenemos la probabilidad de que cada punto pertenezca a cada uno de los clústeres. Y para terminar se posiciona a cada punto en el clúster que más probabilidad tenga.

Este algoritmo funciona muy bien con clústeres Elípticos y de diferente densidad.

Se podría decir que el algoritmo K-means es el caso particular para el que todas las componentes tienen la misma covarianza y los clúster son del mismo tamaño

## Apartado 4

# Pruebas realizadas

Se ha llevado a cabo sobre ambos problemas la iteración de los cinco algoritmos mencionados anteriormente, estableciendo dos partes:

- ¿Cuál es la mejor configuración para cada algoritmo?
- Partiendo de la mejor configuración de cada algoritmo, ¿cuál es el mejor algoritmo para cada problema?

Sabiendo la estructura de las pruebas realizadas, podemos pasar a la exposición de los resultados y conclusiones obtenidos.

### 4.1. Dataset Iris

El Dataset Iris se corresponde con el problema cuyos datos de agrupamiento real son conocidos. A continuación, su representación visual:

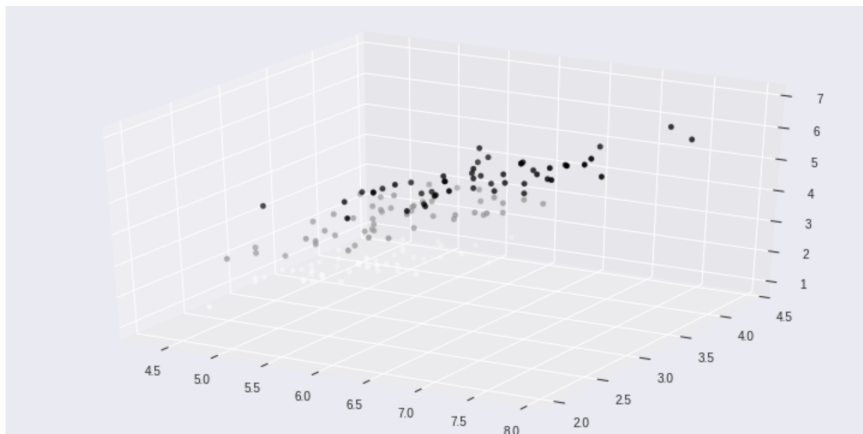


Figura 4.1: Dataset Iris

Como se ha expuesto antes, lo primero es conocer cuál es la mejor configuración posible entre todas las escogidas para cada algoritmo, por lo que veamos uno por uno los resultados obtenidos.

#### 4.1.1. KMeans

Este algoritmo sólo depende de un parámetro,  $K$ , por lo que las distintas configuraciones se basan en distintos valores para este parámetro.

Los distintos valores escogidos han sido  $\{2, 3, 4, 5\}$  centros, siendo la mejor elección del valor para cada métrica la mostrada en la imagen 4.2.

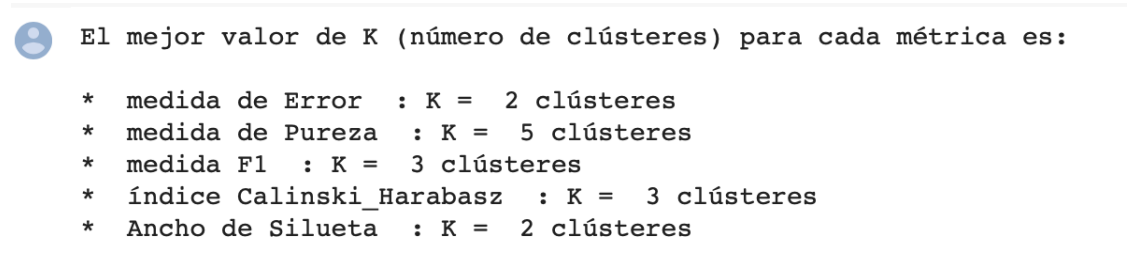


Figura 4.2: Mejor valor del parámetro  $K$  para cada medida de evaluación

Como se puede apreciar, el mejor valor del parámetro se halla entre 2 y 3 clústeres, pero optaremos por establecer como **mejor configuración** para este algoritmos el valor de **3 clústeres** ya que es lo más aproximado a la agrupación real del dataset, como queda reflejado en la figura 4.1.

Los valores de las métricas para esta elección son:

- **Error:** 0.12
- **Pureza:** 0.8799999999999999
- **F1:** 0.8792270531400966
- **Ancho de silueta:** 0.5498955810221876
- **Índice Calinski-Harabasz:** 556.1703613698259

y un ejemplo de solución que se obtiene es el siguiente:

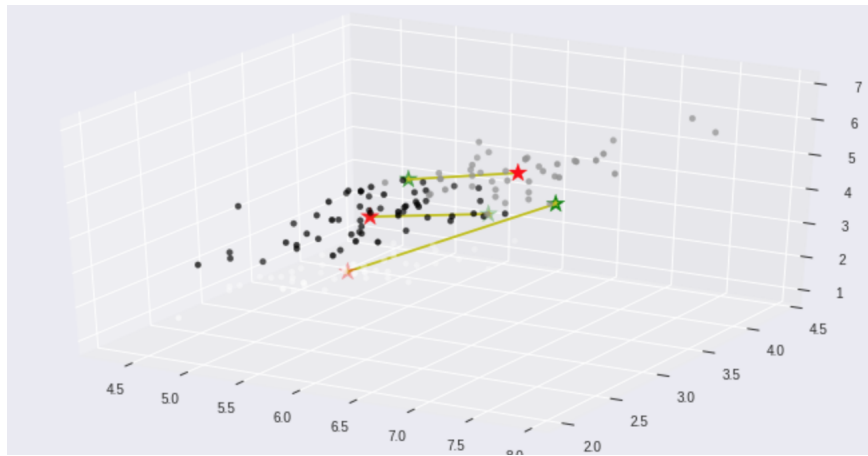


Figura 4.3: Resolución del KNN para 3 clústeres

#### 4.1.2. Aproximación aglomerativa

El algoritmo jerárquico de aproximación aglomerativa depende de dos parámetros:

- $K$ : Número de clústeres del agrupamiento.
- Función de disimilitud: Puede ser mínima, máxima o media.

Así que para averiguar la mejor configuración de este algoritmo iteramos el parámetro  $K$  sobre valores del 2 al 6 y sobre los tres tipos de disimilitudes hasta encontrar la combinación óptima.

Las combinaciones de parámetros que optimizan cada métrica de evaluación son:

```

mejor medida error:
k=2 disimilitud: minima
mejor medida pureza:
k=3 disimilitud: maxima
mejor medida f1:
k=3 disimilitud: maxima
mejor medida Silhouette:
k=2 disimilitud: minima
mejor medida calinski_harabaz:
k=4 disimilitud: media

```

Figura 4.4: Mejor valor del parámetro  $K$  y de la disimilitud para cada medida de evaluación



Como podemos observar, los mejores valores para el parámetro  $K$  son 2 y 3, quedándonos con 3 ya que se aproxima más a la agrupación real conocida. Por otra parte, la función de disimilitud que acompaña al parámetro  $K$  cuando toma valor 3 es la disimilitud máxima, por lo que consideraremos que la **configuración óptima** es  $K = 3$  y la función de **disimilitud máxima**.

Los valores de las métricas de esta configuración son:

- **Error:** 0.23333333333333328
- **Pureza:** 0.8466666666666666
- **F1:** 0.7959199473541594
- **Ancho de silueta:** 0.5136379163325477
- **Índice Calinski-Harabasz:** 510.38321714086834

Siendo el dendograma correspondiente a dicha configuración:

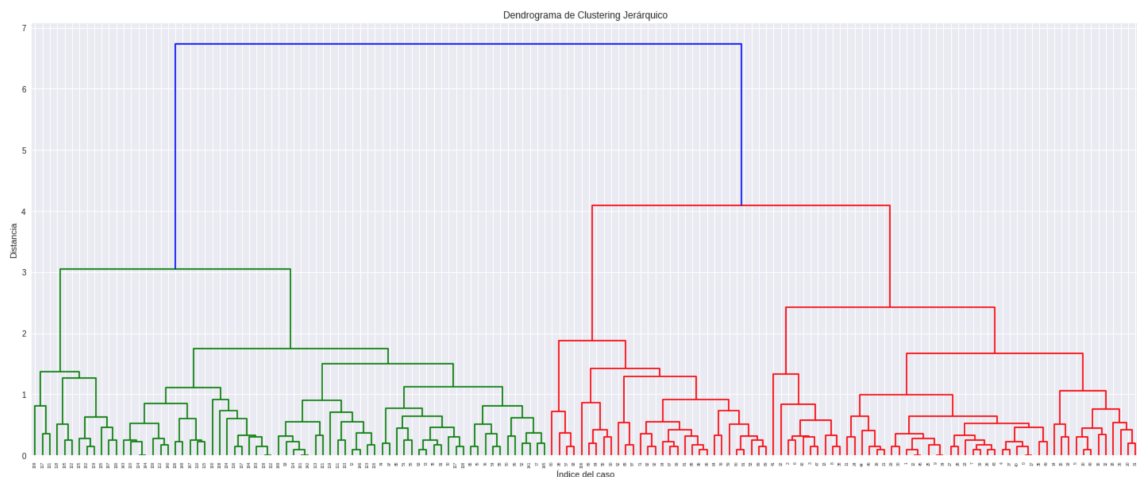


Figura 4.5: Dendograma del Jerárquico Aglomerativo para 3 clústeres y disimilitud máxima

#### 4.1.3. Espectral

El algoritmo Espectral depende tan solo de dos parámetros:

- $K$ : Número de clústeres
- $knn$ : Número de vecinos más cercanos

Se ha fijado  $K$  en 3, ya que sabemos que es la agrupación real de los datos, y se ha iterado sobre distintos valores de  $knn$  para obtener el valor óptimo de la configuración de este algoritmo. Se ha iterado los valores del parámetro  $knn$  desde el 1 al 15, obteniendo que la **mejor configuración** es para **kkn=4**.

Los valores de las métricas obtenidos para esta configuración son:

- **Error:** 0.09999999999999998
- **Pureza:** 0.8999999999999999
- **F1:** 0.8982809480215644
- **Ancho de silueta:** 0.5498051264683734
- **Índice Calinski-Harabasz:** 545.4377870197994

Visualmente, la agrupación queda de la siguiente forma:

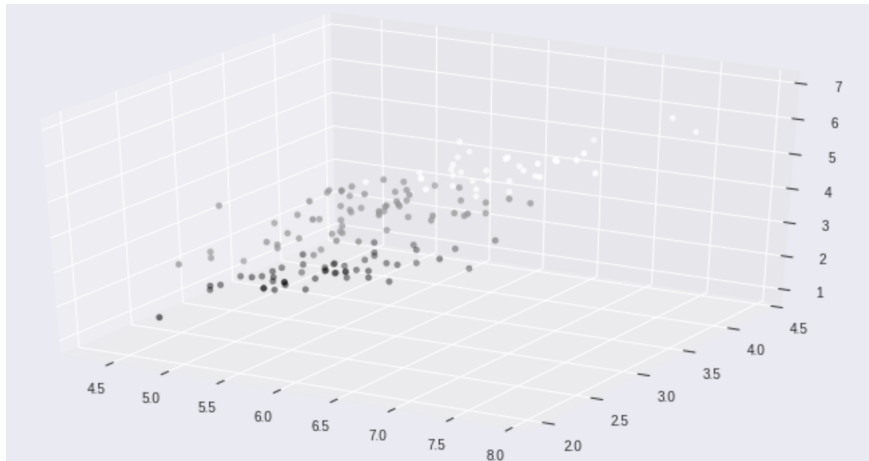


Figura 4.6: Resolución de Spectral para 3 clústeres con 4 vecinos más cercanos

#### 4.1.4. DBSCAN

Este algoritmo depende de dos parámetros como se explicó anteriormente:

- $\epsilon$ : Acota el radio del vecindario a tener en cuenta
- $M$ : Número de vecinos situados dentro de un vecindario necesarios para que un punto se considere nuclear.

En este caso, se ha iterado sobre  $\epsilon$  tomando valores desde 0.3 a 0.8 con un paso de 0.05 y sobre  $M$  tomando valores desde 3 hasta 20.

De esta forma, evaluando una vez más los resultados de las métricas, se obtiene que la mejor configuración es para  $\epsilon = 0,65$  y  $M = 3$ , siendo los resultados obtenidos:

- **Error:** 0.013333333333333308
- **Pureza:** 0.6799999999999999
- **F1:** 0.7747747747747749
- **Ancho de silueta:** 0.5316465076207373
- **Índice Calinski-Harabasz:** 277.81765872523886

Visualmente el agrupamiento obtenido es el siguiente:

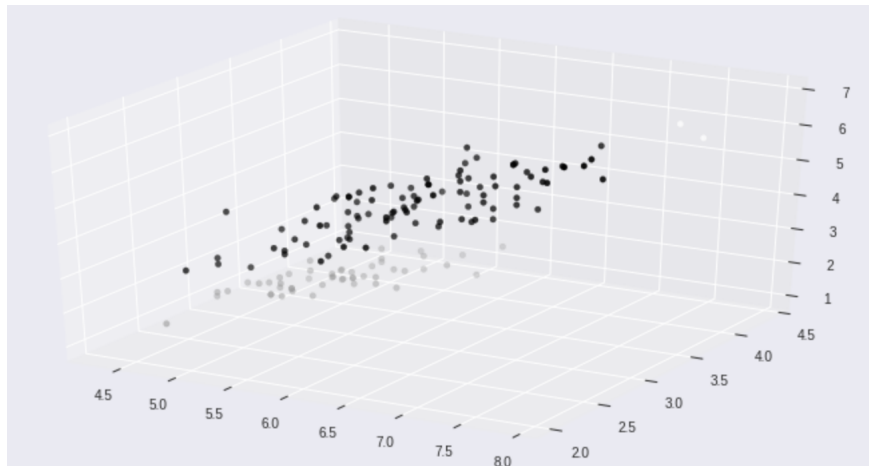


Figura 4.7: Resolución de DBSCAN para  $\epsilon = 0,65$  y  $M = 3$

#### 4.1.5. Modelos de mixturas

Por último el algoritmo de Mixtura Gaussiana donde, al igual que en el algoritmo KNN, sólo variamos el valor de un parámetro,  $K$ , en función del número de clústeres que deseamos predecir.

Los valores escogidos para este parámetro son  $\{2, 3, 4, 5\}$  centros, y el mejor valor de dicho parámetro para cada medida de evaluación es:

 El mejor valor de  $K$  (número de clústeres) para cada métrica es:

- \* medida de Error :  $K = 2$  clústeres
- \* medida de Pureza :  $K = 3$  clústeres
- \* medida F1 :  $K = 3$  clústeres
- \* índice Calinski\_Harabasz :  $K = 2$  clústeres
- \* Ancho de Silueta :  $K = 2$  clústeres

Figura 4.8: Mejor valor del parámetro  $K$  para cada medida de evaluación

Para el algoritmo de mixturas Gaussianas parece evidente que la mejor configuración es la de 2 clústeres siendo los valores de las métricas para estas características las siguientes:

- **Error:** 0.0
- **Pureza:** 0.6666666666666666
- **F1:** 0.7777777777777778
- **Ancho de silueta:** 0.6885194944858716
- **Índice Calinski-Harabasz:** 496.72125954418374

Siendo un ejemplo de solución para el algoritmo de mixturas Gaussianas con dos clústeres el mostrado a continuación:

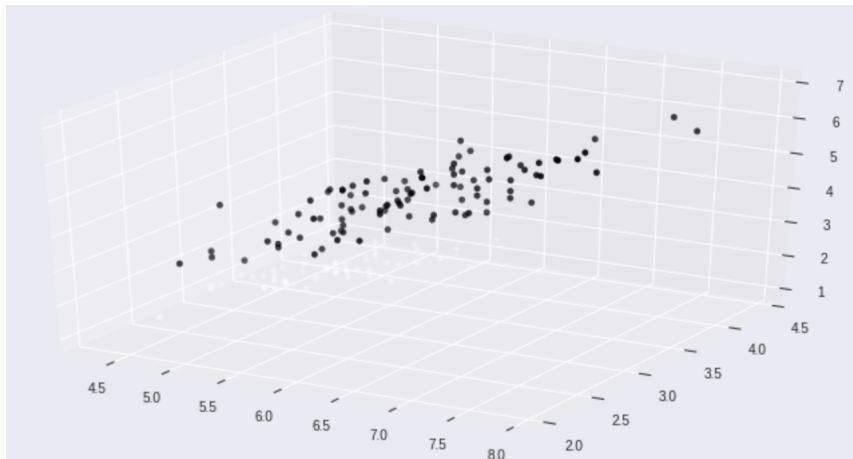


Figura 4.9: Resolución del algoritmos de Mixtura para 2 clústeres

#### 4.1.6. Mejor algoritmo

Partiendo de las mejores configuraciones de cada algoritmo, pasamos a comparar las métricas de cada uno de ellos para decidir cuál es el mejor para el agrupamiento de nuestro dataset:

```

Algoritmos con mejor métrica:

Medida error:
Mixturas medida error

Medida pureza:
Spectral pureza agrupamiento

Medida F1:
DBScan F1

Medida Silhouette:
Kmean silueta

Medida calinski harabaz:
DBScan calinski_harabaz

```

Figura 4.10: Mejores algoritmos para cada medida de evaluación

Como se puede observar en la comparativa, parece que el **mejor algoritmo** es el **DBSCAN** con  $\epsilon = 0,65$  y  $M = 3$ , siendo los resultados de las métricas:

- **Error:** 0.013333333333333308
- **Pureza:** 0.6799999999999999
- **F1:** 0.7747747747747749
- **Ancho de silueta:** 0.5316465076207373
- **Índice Calinski-Harabasz:** 277.81765872523886

## 4.2. Dataset Wine

Por otra parte, tenemos el Dataset Wine, usado para nuestro problema de datos de agrupamiento real no conocidos que, como se explicó en la sección de Conjunto de Datos, aunque sí que se posean estos datos, se van a obviar y proceder como si no existiesen.

Visualmente, el Dataset Wine con los tres atributos que se van a usar quedaría de la siguiente forma:

### 4.2.1. KMeans

Este algoritmo sólo depende de un parámetro,  $K$ , por lo que las distintas configuraciones se basan en distintos valores para este parámetro.

Los distintos valores escogidos han sido  $\{2, 3, 4, 5\}$  centros, siendo la mejor elección del valor para cada métrica la mostrada en la imagen 4.12.

Como podemos observar en este caso, parece que no hay una clara mejor opción, pero si analizamos más en detalle los resultados de las métricas se aprecia que la diferencia para la métrica de ancho de silueta es ínfima entre 2 y 3 clústeres, por lo que asumiremos

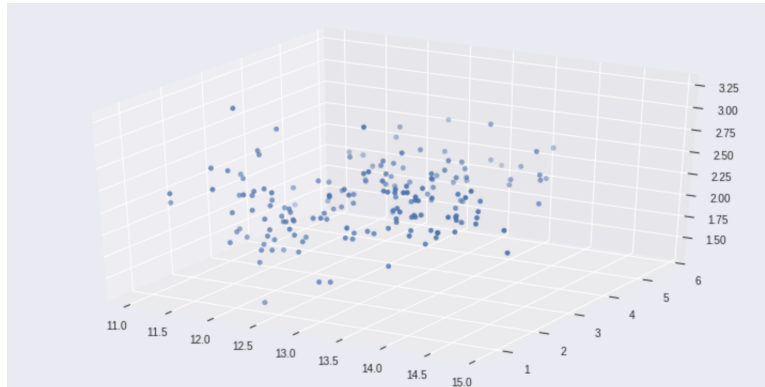



Figura 4.11: Dataset Wine

 El mejor valor de  $K$  (número de clústeres) para cada métrica es:

- \* índice Calinski\_Harabasz :  $K = 3$  clústeres
- \* Ancho de Silueta :  $K = 2$  clústeres
- \* RMSSTD :  $K = 5$  clústeres

Figura 4.12: Mejor valor del parámetro  $K$  para cada medida de evaluación

como **mejor configuración 3 clústeres**.

Los valores de las métricas para esta elección son:

- **Índice Calinski-Harabasz:** 198.05949075445147
- **Ancho de silueta:** 0.45082007409391917
- **RMSSTD:** 0.4630056453425246

y un ejemplo de solución que se obtiene es el siguiente:

#### 4.2.2. Aproximación aglomerativa

El algoritmo jerárquico de aproximación aglomerativa depende de dos parámetros:

- $K$ : Número de clústeres del agrupamiento.
- Función de disimilitud: Puede ser mínima, máxima o media.

Así que para averiguar la mejor configuración de este algoritmo iteramos el parámetro  $K$  sobre valores del 2 al 6 y sobre los tres tipos de disimilitudes hasta encontrar la combinación óptima.

La configuración óptima que se obtiene es que  $K = 2$  y disimilitud media, con valores de las métricas:

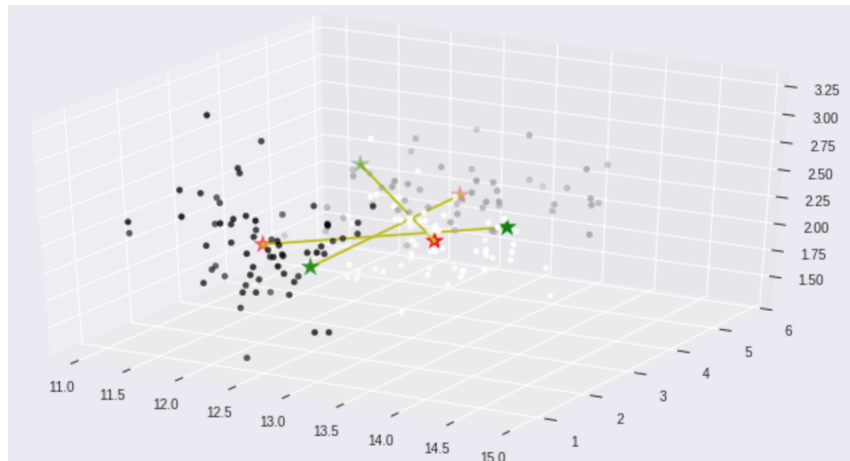


Figura 4.13: Resolución del KNN para 3 clústeres

- **Índice Calinski-Harabasz:** 169.97406757446487
- **Ancho de silueta:** 0.42837079589062194

Siendo el dendograma correspondiente a dicha configuración:

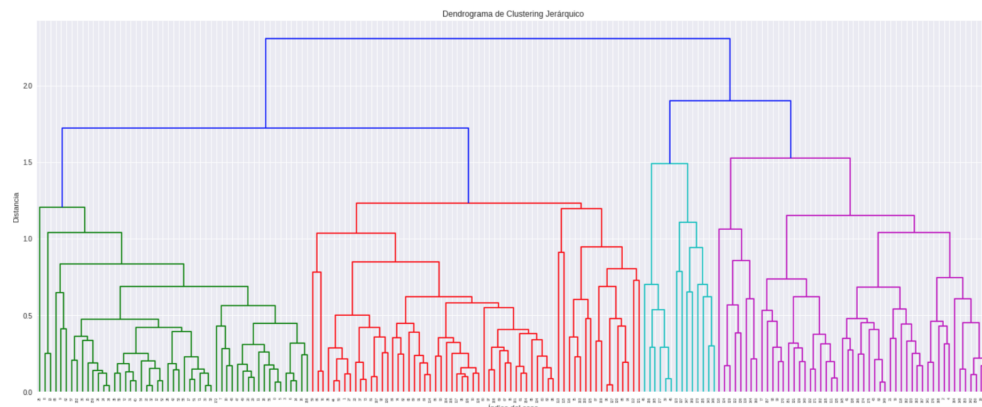


Figura 4.14: Dendograma del Jerárquico Aglomerativo para 2 clústeres y disimilitud media

### 4.2.3. Espectral

El algoritmo Espectral depende tan solo de dos parámetros:

- $K$ : Número de clústeres
- $knn$ : Número de vecinos más cercanos

Se ha fijado  $K$  en 3 y se ha iterado sobre distintos valores de  $knn$  para obtener el valor óptimo de la configuración de este algoritmo. Se ha iterado los valores del parámetro  $knn$  desde el 1 al 15, obteniendo que la **mejor configuración** es para **kkn=5**.

Los valores de las métricas obtenidos para esta configuración son:

- **Índice Calinski-Harabasz:** 178.5325543114175
- **Ancho de silueta:** 0.4416382929002338

Visualmente, la agrupación queda de la siguiente forma:

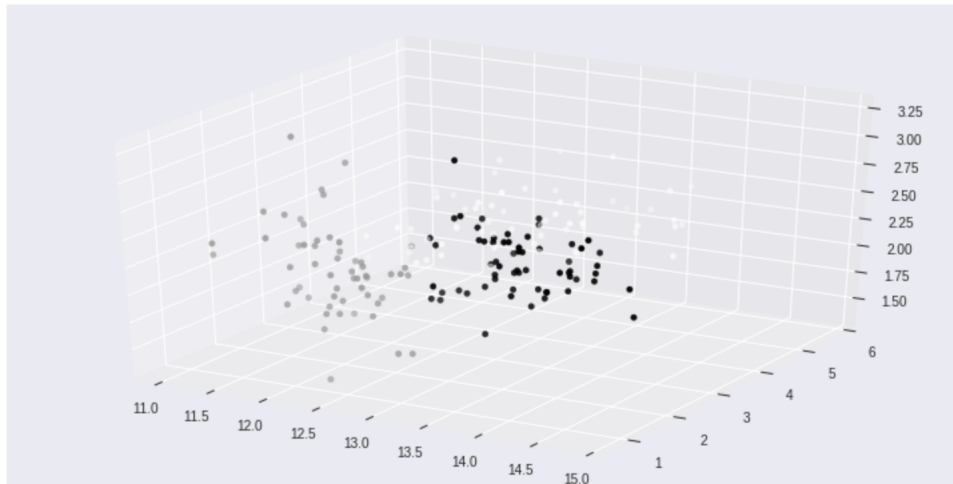


Figura 4.15: Resolución de Spectral para 3 clústeres con 5 vecinos más cercanos

#### 4.2.4. DBSCAN

Este algoritmo depende de dos parámetros como se explicó anteriormente:

- $\epsilon$ : Acota el radio del vecindario a tener en cuenta
- $M$ : Número de vecinos situados dentro de un vecindario necesarios para que un punto se considere nuclear.

En este caso, se ha iterado sobre  $\epsilon$  tomando valores desde 0.3 a 0.8 con un paso de 0.05 y sobre  $M$  tomando valores desde 3 hasta 20.

De esta forma, evaluando una vez más los resultados de las métricas, se obtiene que la mejor configuración es para  $\epsilon = 0,65$  y  $M = 3$ , siendo los resultados obtenidos:

- **Error:** 0.013333333333333308
- **Pureza:** 0.6799999999999999



- **F1:** 0.7747747747747749
- **Ancho de silueta:** 0.5316465076207373
- **Índice Calinski-Harabasz:** 277.81765872523886

Visualmente el agrupamiento obtenido es el siguiente:

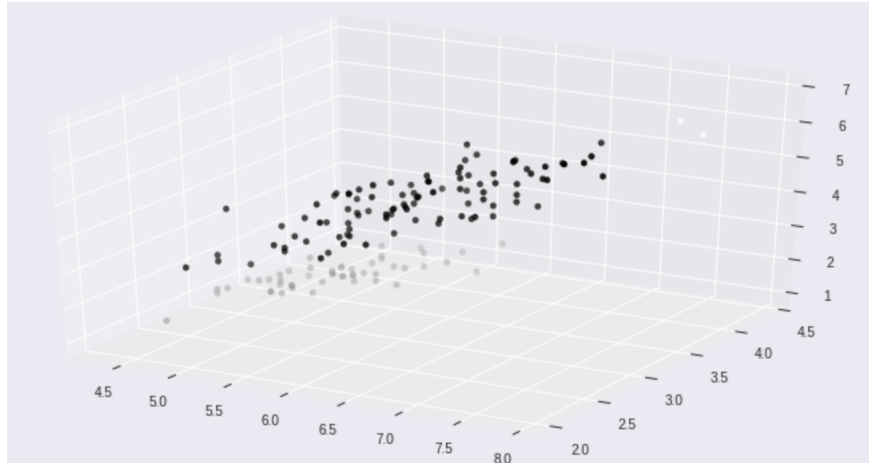


Figura 4.16: Resolución de DBSCAN para  $\epsilon = 0,65$  y  $M = 3$

#### 4.2.5. Modelos de mixturas

Por último el algoritmo de Mixtura Gaussiana donde, al igual que en el algoritmo KNN, sólo variamos el valor de un parámetro,  $K$ , en función del número de clústeres que deseamos predecir.

Los valores escogidos para este parámetro son  $\{2, 3, 4, 5\}$  centros, y el mejor valor de dicho parámetro para cada medida de evaluación es:

- 👤 El mejor valor de  $K$  (número de clústeres) para cada métrica es:
- \* índice Calinski\_Harabasz :  $K = 4$  clústeres
  - \* Ancho de Silueta :  $K = 3$  clústeres

Figura 4.17: Mejor valor del parámetro  $K$  para cada medida de evaluación

En este caso no se puede aplicar la medida RMSSTD ya que este algoritmo no devuelve los centros de los clústeres predichos, por lo que observando tan solo las dos métricas obtenidas no parece quedar muy claro cual es la mejor opción. Por ello, mirando más detalladamente los resultados de las métricas para estos dos valores, la diferencia entre los valores del índice de Calinski-Harabasz es mucho más insignificante que la diferencia entre los puntajes del ancho de silueta.

Es por eso, por lo que asumiremos como **mejor configuración** de este algoritmo la opción de **3 clústeres**, con valores de las métricas:

- **Índice Calinski-Harabasz:** 152.83652626950627
- **Ancho de silueta:** 0.41297491420107685

Siendo un ejemplo de solución para el algoritmo de mixturas Gaussianas con tres clústeres el mostrado a continuación:

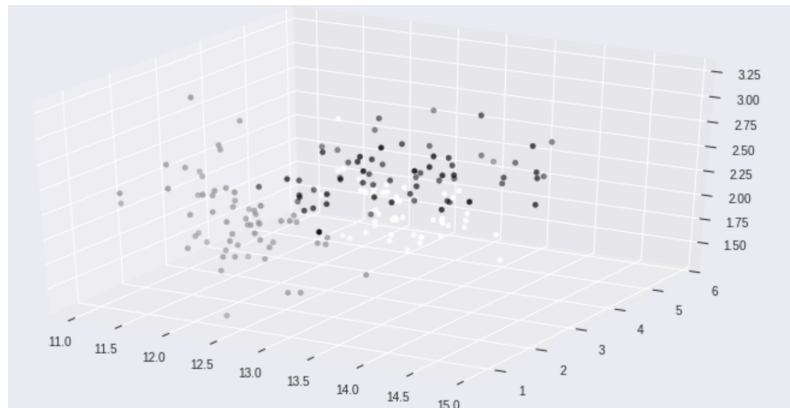


Figura 4.18: Resolución del algoritmos de Mixtura para 3 clústeres

#### 4.2.6. Mejor algoritmo

Partiendo de las mejores configuraciones de cada algoritmo, pasamos a comparar las métricas de cada uno de ellos para decidir cuál es el mejor para el agrupamiento de nuestro dataset:

```
Algoritmos con mejor métrica:

Mejor medida Silhouette:
Kmean Silhouette

Mejor medida calinski_harabaz:
DBScan calinski_harabaz_score k=2

Mejor medida RMSSTD:
Kmean RMSSTD
```

Figura 4.19: Mejores algoritmos para cada medida de evaluación

Como se puede observar en la comparativa, parece que el **algoritmo con mejores resultados** para este problema es el **K-Means** con  $K = 3$ , siendo los valores de las métricas:

- **Índice Calinski-Harabasz:** 198.059491
- **Ancho de silueta:** 0.450820
- **RMSSTD:** 0.463006