Universidad de Costa Rica

Escuela de Matemática

Métodos de Clustering en Python

CA-0305: Herramientas de Ciencia de Datos II

I-2024

Prof: Luis Alberto Potoy Juárez

Estudiante:

Sofía Bocker Brenes - C11102

1 Tema

Existen categorías y subcategorías de clustering, las categorías o tipos más conocidos son: clustering de centroides, densidad, jerárquico, distribución, entre otros.

El clustering con centroides es el más conocido y el que más se aplica de todos ya que es eficiente y relativamente simple, pero sensible a las condiciones iniciales y a los valores atípicos. Trabaja sobre la cercanía de los puntos de datos al valor central elegido. Es ideal para conjuntos de datos con clústers bien definidos y fácilmente separables. También es adecuado cuando el número de conglomerados se conoce o puede estimarse fácilmente. Sin embargo, no es la mejor opción para conjuntos de datos con grupos superpuestos o formas no uniformes (Verma, 2023).

El ejemplo más conocido es el clustering por medio de k-means. Este es un proceso iterativo de asignación de cada punto de datos a los grupos y, lentamente, los puntos de datos se agrupan en función de características similares. El objetivo es minimizar la suma de distancias entre los puntos de datos y el centroide del cluster, para identificar el grupo correcto al que debe pertenecer cada punto (Hamraz, 2023).

Con respecto al clustering por medio de densidad, como su nombre indica, considera la densidad por delante de la distancia. Además, no asignan valores atípicos a los clusters. Es ideal para conjuntos de datos con grupos superpuestos o de forma irregular. Estos métodos gestionan regiones de datos densas y escasas centrándose en la densidad local y pueden distinguir grupos con una variedad de morfologías (GeeksforGeeks, 2023).

En este caso, el más conocido es DBSCAN, el cual divide los datos en regiones densas de puntos que están separados por áreas menos densas. Define clusters como áreas de los datos donde hay muchos puntos cercanos entre sí, mientras que los puntos que están lejos de cualquier grupo se consideran valores atípicos o ruido (Yenigün, 2024).

Siguiendo con el clustering jerárquico, crea un árbol de clusters. Como es de esperar, se adapta bien a datos jerárquicos, como las taxonomías. Hay dos tipos principales de clustering jerárquico: aglomerativo (bottom-up) y divisoria (top-down) (Martínez, 2020).

Continuando con el ejemplo de clustering jerárquico, se tiene el método aglomerativo. Se utiliza para agrupar objetos en grupos según su similitud entre sí, de forma ascendente, donde cada punto de datos se asigna a su propio cluster. Luego esos cluster se unen. En cada iteración, se fusionan grupos similares hasta que todos los puntos de datos forman parte de un gran clus-

ter raíz (Martínez, 2020).

Por último, está el clustering basado en distribuciones estadísticas. A medida que aumenta la distancia desde el centro de la distribución, disminuye la probabilidad de que un punto pertenezca a la distribución, por lo que crea y agrupa puntos de datos según su probabilidad de pertenecer a la misma distribución (Developers, 2024).

En clustering por distribución está el modelo de mezcla gaussiana, el cual asume que los datos son generados por una mezcla de distribuciones gaussianas (normales), no necesita datos de forma circular para que funcione bien, a diferencia de k-means (Universidad Francisco de Vitoria, s.f.).

2 Idea principal del código

2.1 Librerías necesarias

Primeramente, se decidió aplicar cada una de las metodologías sobre la base reconocida *Iris*, perteneciente a la librería *sklearn.datasets*. Es debido a esto que el código está adaptado para recibir como objeto alguno de estos datasets: load_iris, load_digits, load_wine, load_breast_cancer, load_diabetes, load_linnerud (Scikit-Learn, s.f.).

```
from sklearn.datasets import load_iris, load_digits, load_wine,
load_breast_cancer, load_diabetes, load_linnerud
```

Hay cuatro clases, una para cada uno de los métodos que se van a implementar: kmeans, DBSCAN, aglomerativo y mezcla gaussiana. Para esto, se necesitan importar los métodos respectivos de la librería *sklearn.metrics* (Scikit-Learn, 2024b), (Scikit-Learn, 2024c), (Scikit-Learn, 2024d), (Scikit-Learn, 2024a).

```
from sklearn.cluster import KMeans
from sklearn.cluster import DBSCAN
from sklearn.cluster import AgglomerativeClustering
from sklearn.mixture import GaussianMixture
```

Hay un método cuya función es imprimir una tabla resumen del dataset que se va a clusterizar, por lo que es necesario el uso de la librería *pandas*.

```
import pandas as pd
```

Adicionalmente, se decidió usar el método *silhouette score* para poder estudiar la separación entre los grupos resultantes. Tiene un rango de $[-1, 1]$, entre más alto el valor, mejor es el clustering (Scikit-Learn, 2024e).

```
from sklearn.metrics import silhouette_score
```

Por último, para poder visualizar los clústeres resultantes para cada metodología, se requiere importar de las siguientes librerías:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

2.2 Métodos

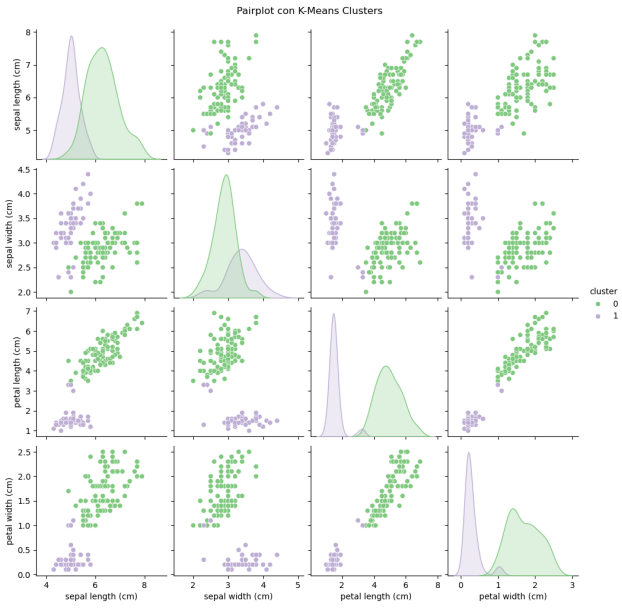
Cada clase rescibe un objeto: la base de datos extraída de *sklearn.datasets*, además, cada clase posee 5 métodos. El primer se encarga de devolver una tabla resumen de las estadísticas del dataste. El segundo encuentra los cústers y los agrega como una columna, de modo que devuelve un dataframe con el dataset y sus clústeres. El tercero utiliza el método de clusters para poder imprimir las visualizaciones correspondientes por medio de un pairplot. El cuarto itera sobre un rango de clústeres e imprime el *silhouette score* para cada número de clústeres, lo que permite distinguir (de acuerdo a esta métrica) cuál es el más apropiado. Finalmente, el último método imprime el *silhouette score* del clúster escogido y con el cual se aplicó la metodología.

Las clases fueron hechas en *Spyder*, por lo que se encuentran en un arhivo .py. El archivo min donde se realizaron todas las pruebas fue hecho en un notebook de *Jupyter*. En este, se imprimen los resultados al utilizar las cantidad de clústeres recomendados por el *silhouette score*.

Todo se encuentra en el siguiente repositorio de GitHub: click **aquí**

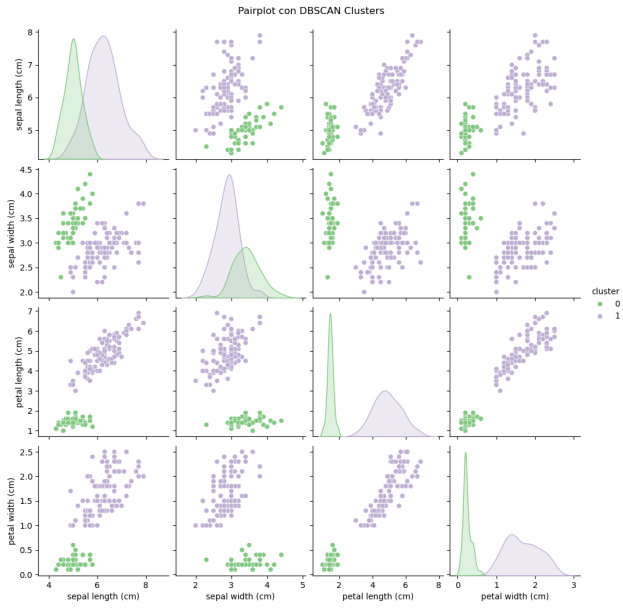
3 Resultados

Figura 1: Resultados K-Means



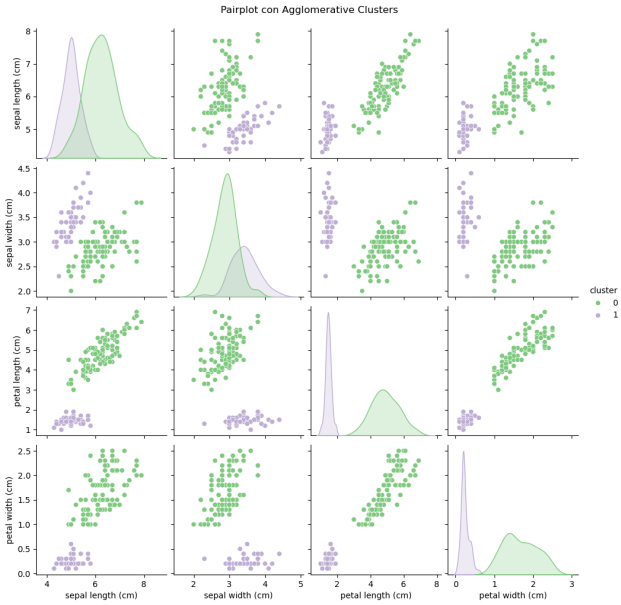
Fuente: Elaboración propia

Figura 2: Resultados DBSCAN



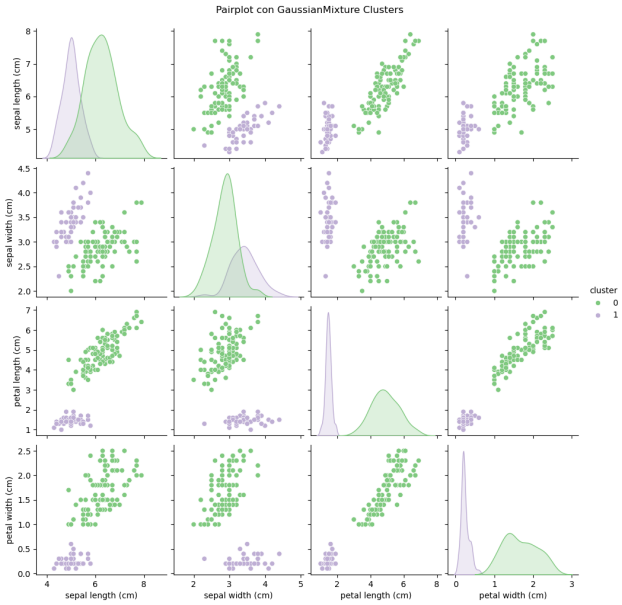
Fuente: Elaboración propia

Figura 3: Resultados Aglomerativo



Fuente: Elaboración propia

Figura 4: Resultados Gaussian Mixture



Fuente: Elaboración propia

4 Recomendaciones

- Se realizaron las pruebas en una base de datos que ya está predispuesta a comportarse ”bonito”, por lo que sería interesante utilizar métodos de clustering en una base desordenada, que se apega más a la realidad. Para eso, habría que pasar por una limpieza exhaustiva de los datos, sin embgargo, puede ser enriquecedor.
- Sólo se pudieron a prueba cuatro de los métodos más conocidos, lo mejor es estudiar acerca de otros métodos con el fin de saber mejor cúal es el que se adapta a las necesidades.
- Se utilizó el método *Silhouette Score* para poder determinar el mejor número de clústeres, sin embargo, no es recomendado basarse en un sólo método ya que existe margen de errores, además de que puede ser bastante rígido. Lo mejor sería utilizar más de una métrica, además de aplicar un criterio de experto.

Referencias

Developers, G. (2024). *Clustering algorithms.* Descargado de https://developers.google.com/machine-learning/clustering/clustering-algorithms

GeeksforGeeks. (2023). *Clustering in machine learning.* Descargado de https://www.geeksforgeeks.org/clustering-in-machine-learning/

Hamraz, M. (2023). *K-means clustering explained.* Descargado de https://medium.com/@musharrafhamraz02/k-means-clustering-explained-bc58ebf7396b

Martínez, E. (2020). *8 algoritmos de agrupación en clústeres en el aprendizaje automático que todos los científicos de datos deben conocer.* Descargado de https://www.freecodecamp.org/espanol/news/8-algoritmos-de-agrupacion-en-clusteres-en-el-aprendizaje-automatico-que-todos-los-cientificos-de-datos-deben-conocer/

Scikit-Learn. (s.f.). *Toy datasets in scikit-learn.* Descargado de https://scikit-learn.org/stable/datasets/toy_dataset.html

Scikit-Learn. (2024a). *Agglomerative clustering.* Descargado de https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

Scikit-Learn. (2024b). *Dbscan clustering.* Descargado de https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

Scikit-Learn. (2024c). *Gaussianmixture clustering.* Descargado de https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html

Scikit-Learn. (2024d). *K-means clustering.* Descargado de https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Scikit-Learn. (2024e). *K-means clustering and silhouette analysis.* Descargado de https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Universidad Francisco de Vitoria. (s.f.). *Clustering: ¿qué es?* Descargado de https://postgrado.ufv.es/clustering-que-es/

Verma, A. (2023). *Centroid-based clustering: A powerful machine learning technique for partitioning datasets.* Descargado de https://dev.to/anurag629/centroid-based-clustering-a-powerful-machine-learning-technique-for-partitioning-datasets-41im

Yenigün, O. (2024). *Dbscan clustering algorithm demystified.* Descargado de https://builtin.com/articles/dbscan