

Tarea 5. Aprendizaje no supervisado

Salinas, Iván

10 de junio de 2024

1. Introducción

El Aprendizaje Automatizado (Machine Learning; ML) es una rama de la inteligencia artificial, en gran parte inspirada en el razonamiento humano, que comprende el aprendizaje a partir de experiencia (Sammut y Webb 2011)(Sammut y Webb 2011). El aprendizaje automático aborda, a su vez, una serie de problemáticas que tributan a problemas específicos, entre ellos: los problemas de clasificación, asociación, agrupamiento, y selección de rasgos. En el agrupamiento se parte de un conjunto de ejemplos el cual se desea organizar en grupos usualmente de acuerdo a una noción de similitud que generalmente es determinada por una función o métrica de distancia. La proximidad entre ejemplos determina la pertenencia o no a un grupo; por tanto, se estima que un elemento será más similar o tendrá mayores propiedades en común con los elementos de su grupo, que con respecto a los elementos de un grupo diferente [1].

Muchos de los métodos de aprendizaje automático dependen del cálculo de distancias para estimar la similitud entre dos ejemplos teniendo en cuenta la estructura de los datos. Este es el caso, por ejemplo, del algoritmo de los k vecinos más cercanos (k-Nearest Neighbor; kNN) para la comparación de las instancias entrantes con los datos conocidos (ejemplos de entrenamiento) o el k -medias (k-means) para calcular la distancia entre los objetos y su centro más cercano [1].

Muchos algoritmos no supervisados para la reducción de dimensionalidad realizan un aprendizaje no supervisado de métricas de distancias utilizando información de los propios datos o de la dimensión donde se encuentran representados. Este grupo de métodos se pueden clasificar en métodos no lineales y lineales. Los algoritmos de reducción no lineales consideran que cada uno de los datos de alta dimensionalidad puede ser descrito a través de una función compuesta por los parámetros más relevantes y los datos son vistos como extractos de una dimensión subyacente embebida en la dimensión original del espacio. El objetivo es embeber datos que originalmente se encuentran en una dimensión en otra dimensión reducida, al mismo tiempo que se preservan las características principales de los datos. Para cada espacio dimensional debe existir intrínsecamente un espacio reducido; y por tanto, es posible acceder a los datos reducidos a través de algoritmos que interpreten o preserven la naturaleza de los datos embebidos. Entre los métodos más utilizados de este tipo se encuentran ISOMAP,


GIA	Scan. D.N	CIBJO- IDC/HRD
 Colorless (Incoloro)	D	Blanco Excepcional+
	E	Blanco Excepcional
	F	Blanco Extra +
 Near colorless (casi Incoloro)	G	Blanco Extra
	H	Blanco
	I	Blanco Ligero
	J	Color

Figura 1: Clasificación de color.

el cual busca un sub-espacio que preserve mejor las distancias geodésicas entre dos puntos de datos y los métodos LLE y LE, que se enfocan en la preservación de las estructuras de las vecindades locales [1].

1.1. Objetivos

En este reporte se desea usar aprendizaje no supervisado spectral clustering partiendo desde lo visto en clase con k-medias para clasificar los datos que se han estado trabajando a lo largo del curso, además se añadirán los temas previamente trabajados como el procesamiento de los datos, su origen, la descripción de estos mismos y se llegara a una conclusión una vez obtenidos los resultados del análisis.

2. Descripción de los datos

2.1. Origen de los datos

Justificación y explicación de datos El conjunto de datos con el que se trabajara durante el curso sera un dataset con datos que se han juntado de diamantes y sus diferentes características, las cuales son:

Price: En el caso del precio de los diamantes, puede variar de forma continua, desde valores muy bajos hasta valores muy altos, sin límites específicos y con una infinidad de posibilidades entre ellos.

Carat: Son los quilates, el cual representa a 1 quilate = 0.2 gramos

Cut: Es la calidad de corte en el diamante que son Fair, Good, Very Good, Premium e Ideal. Se piensa que es importante por la importancia que puede tener un buen trabajo realizado por el artesano que corto el diamante.

Color: El color se clasifica desde la J a la D, lo cual nos indica que tan incoloro es el diamante, conforme más bajo sea su escala tiene un color cercano al amarillo. Esto se considera importante para el precio ya que nos indica la pureza del diamante (1).

Clarity: Es una medida de qué tan claro es el diamante (I1 (peor), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (mejor)). Se piensa que es importante ya que los diamantes al ser creados bajo altas temperaturas y presiones pueden llegar a

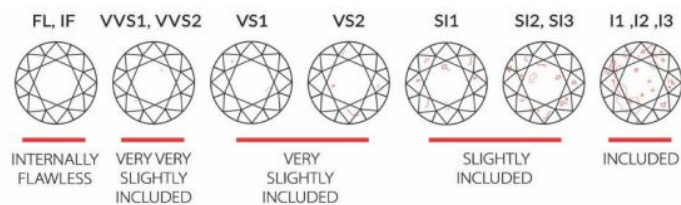


Figura 2: Clasificación de pureza.

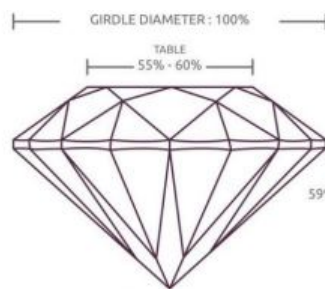


Figura 3: Proporción de table.

tener burbujas o imperfecciones dentro y eso se piensa que afecta el precio de este mismo (2).

Table: Ancho de la parte superior del diamante en relación con el punto más ancho (43-95). Se piensa que es importante en relación con el precio ya que es lo que le da la forma característica de diamante una buena proporción de ancho de mesa (3).

Luego tenemos las variables x y z las cuales representan el tamaño de las dimensiones del diamante.

X longitud en mm (0-10.74)

Y ancho en mm (0-58.9)

Z profundidad en mm (0-31.8)

La importancia de estas variables es vital ya que son las que nos describen las dimensiones del diamante mismo, por lo que un diamante con mayores dimensiones podría ser el que tenga mayor precio.

2.2. Preprocesamiento

Para tener una vista general de los datos tomamos una muestra de 5 observaciones aleatorias (1).

Ahora visualizamos las principales medidas descriptivas de nuestros datos con valores numéricos, con esto podemos notar que tenemos una muestra de 53,940 filas/observaciones, por otro lado, las variables x y z tienen valores de 0 lo cual nos da una idea de que hay datos los cuales hay que limpiar, ya que no

	carat	cut	color	clarity	depth	table	price	x	y	z
4339	1.50	Premium	H	I1	61.1	59.0	3599	7.37	7.26	4.47
15117	1.01	Premium	D	SI1	61.8	58.0	6075	6.42	6.37	3.95
25101	1.50	Very Good	D	VS2	63.8	55.0	13629	7.24	7.28	4.63
38478	0.42	Ideal	G	VVS2	62.1	57.0	1031	4.77	4.80	2.97
20351	1.35	Premium	H	VS1	60.5	60.0	8747	7.19	7.16	4.34

Cuadro 1: Muestra aleatoria de los datos.

existen diamantes bidimensionales o de una dimensión (2.2).

	carat	depth	table	price	x	y	z
count	53940	53940	53940	53940	53940	53940	53940
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43	43	326	0	0	0
25 %	0.400000	61	56	950	4.710000	4.720000	2.910000
50 %	0.700000	61.8	57	2401	5.700000	5.710000	3.530000
75 %	1.040000	62.5	59	5324.25	6.540000	6.540000	4.040000
max	5.010000	79	95	18823	10.74	58.9	31.8

Cuadro 2: Resumen estadístico de los datos

Se buscan cuantos datos no son nulos, de esta forma podemos notar que no hay datos nulos en los datos, por lo que ahora procederemos con buscar la cantidad de datos con valor 0 y los borramos. Como resultado de estas acciones terminamos borrando 35 filas que contenían 7 ceros en la variable y, 8 ceros en la variable x y 20 ceros en la variable z.

Variable	Cantidad de 0
carat	0
cut	0
color	0
clarity	0
depth	0
table	0
price	0
x	8
y	7
z	20

Cuadro 3: Cantidad de 0 por variable

Vamos a quitar los valores anormales que excedan nuestros máximos y mínimos, para hacer esto primero hacemos una copia de nuestro dataframe.

```
df_copy = df_copy[(df_copy["depth"] < 75) & (df_copy["depth"] > 45)]
```

Index	# Column	Non-Null Count	Dtype
0	53940	non-null carat	float64
1	53940	non-null cut	object
2	53940	non-null color	object
3	53940	non-null clarity	object
4	53940	non-null depth %	float64
5	53940	non-null table %	float64
6	53940	non-null price	int64
7	53940	non-null x (length)	float64
8	53940	non-null y (width)	float64
9	53940	non-null z (depth)	float64

Cuadro 4: Recuento de datos no nulos

```
df_copy = df_copy[(df_copy["table"] < 80) & (df_copy["table"] > 40)]
df_copy = df_copy[(df_copy["x"] < 30)]
df_copy = df_copy[(df_copy["y"] < 30)]
df_copy = df_copy[(df_copy["z"] < 30) & (df_copy["z"] > 2)]
df_copy.shape
```

Ahora hacemos una lista con nuestras variables categoricas para poder cambiar sus valores a números enteros, de esta forma sera mas fácil trabajar con los datos en un futuro, todos los valores serán desde la categoría peor calificada con valor 1 hasta la mejor que tendrá el valor numérico mas alto. Con esto terminamos la limpieza de datos y tenemos un dataframe de dimensiones (53907, 10).

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for col in object_cols:
    df_copy[col] = label_encoder.fit_transform(df_copy[col])
df_copy.head()
```

	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	2	1	3	61.5	55.0	326	3.95	3.98	2.43
2	0.21	3	1	2	59.8	61.0	326	3.89	3.84	2.31
3	0.23	1	1	4	56.9	65.0	327	4.05	4.07	2.31
4	0.29	3	5	5	62.4	58.0	334	4.20	4.23	2.63
5	0.31	1	6	3	63.3	58.0	335	4.34	4.35	2.75

Cuadro 5: Datos de diamantes con valores enteros en categorías.

2.3. Estadística descriptiva

Comenzaremos vizualisando las medidas descriptivas de nuestros datos que ya han sido limpiados en la seccion anterior.

	carat	cut	color	clarity	depth	table	price	x	y	z
count	53907	53907	53907	53907	53907	53907	53907	53907	53907	53907
mean	0.8	2.6	2.6	3.8	61.7	57.5	3930.6	5.7	5.7	3.5
std	0.5	1.0	1.7	1.7	1.4	2.2	3987.2	1.1	1.1	0.7
min	0.2	0.0	0.0	0.0	50.8	43.0	326.0	3.7	3.7	2.1
25 %	0.4	2.0	1.0	2.0	61.0	56.0	949.0	4.7	4.7	2.9
50 %	0.7	2.0	3.0	4.0	61.8	57.0	2401.0	5.7	5.7	3.5
75 %	1.0	3.0	4.0	5.0	62.5	59.0	5322.0	6.5	6.5	4.0
max	5.0	4.0	6.0	7.0	73.6	79.0	18823.0	10.7	10.5	7.0

Cuadro 6: Estadísticas descriptivas de los datos limpios.

Para una correcta visualización de los datos graficamos los histogramas de nuestras 10 variables y hacemos sus diagramas de dispersión y de violín.

3. Antecedentes

El clustering o agrupamiento es el proceso de particionar un conjunto de datos (u objetos) en un conjunto de subclases significativas llamadas grupos (clusters). Un grupo es una colección de objetos de datos que son similares a otros y así pueden ser tratados colectivamente como un grupo. El agrupamiento es una forma de clasificación no supervisada en la que, a diferencia de la supervisada, no se conocen las etiquetas de las clases (no hay clases predefinidas) y puede que tampoco se conozca el número de grupos. Un buen método de agrupamiento produce grupos de alta calidad en los cuales la similitud dentro del grupo es alta y la similitud entre las clases es baja. La medida de similitud se define usualmente por proximidad en un espacio multidimensional [2].

Como ejemplo de la gran influencia de métodos de aprendizaje automático nos podemos ir a campos como la medicina en donde los modelos de aprendizaje computacional se han aplicado con éxito tanto a problemas motivados por la práctica clínica, como el diagnóstico asistido por ordenador, como a problemas de análisis de datos de investigación médica básica. En los últimos años ha habido un gran auge en la investigación y desarrollo de modelos de aprendizaje computacional aplicados al diagnóstico médico de diversas enfermedades y condiciones médicas [3].

La figura 7 muestra el número de artículos publicados entre 1969 y 2014 sobre aplicaciones de técnicas de aprendizaje computacional al diagnóstico médico. Como se puede observar, hay una tendencia creciente, la cual se ha acelerado durante los últimos 10 años, alcanzando un volumen de cerca de 100 artículos sobre el tema por año. Los tipos de problemas de diagnóstico médico abordados con estas técnicas cubren prácticamente todas las especialidades de la medicina, algunos ejemplos incluyen: diagnóstico de glaucoma, identificación de enfermedades cardiovasculares, detección de la enfermedad de Alzheimer y detección del cáncer de próstata [3].

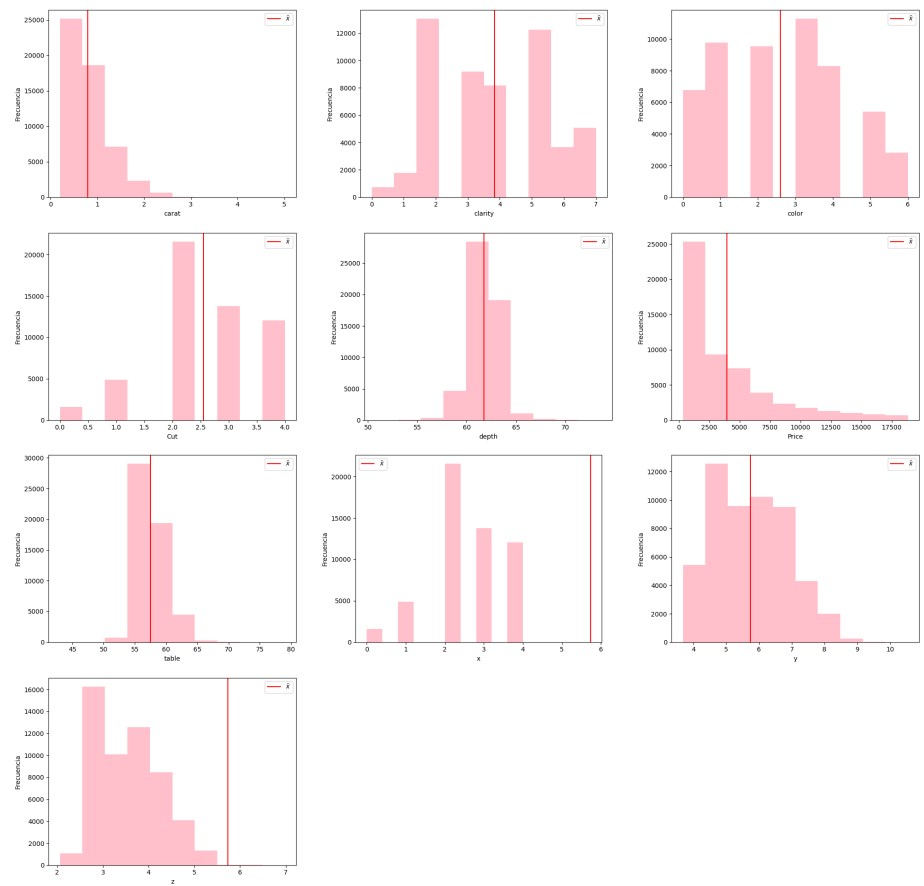


Figura 4: Histogramas

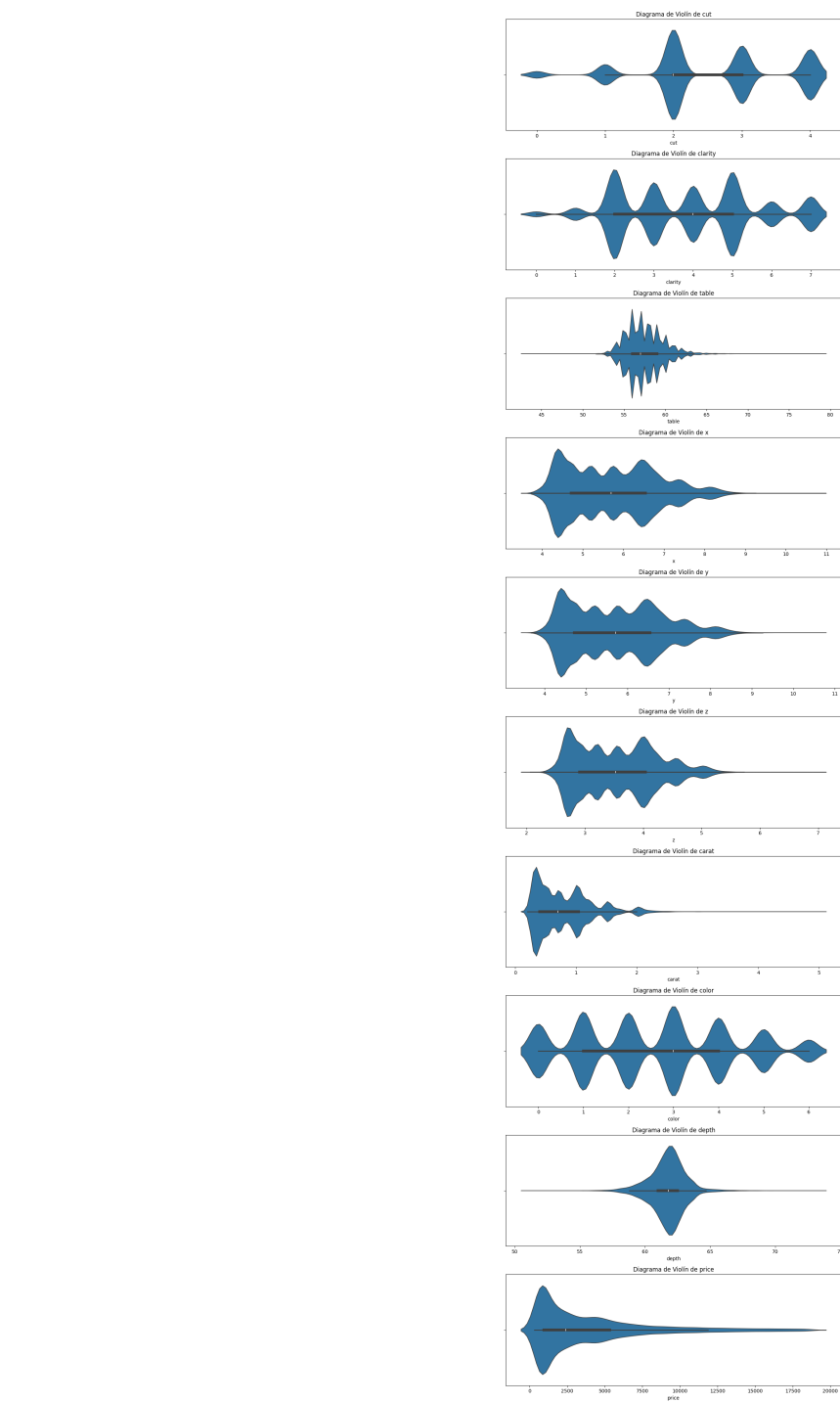


Figura 5: Diagramas de violín.

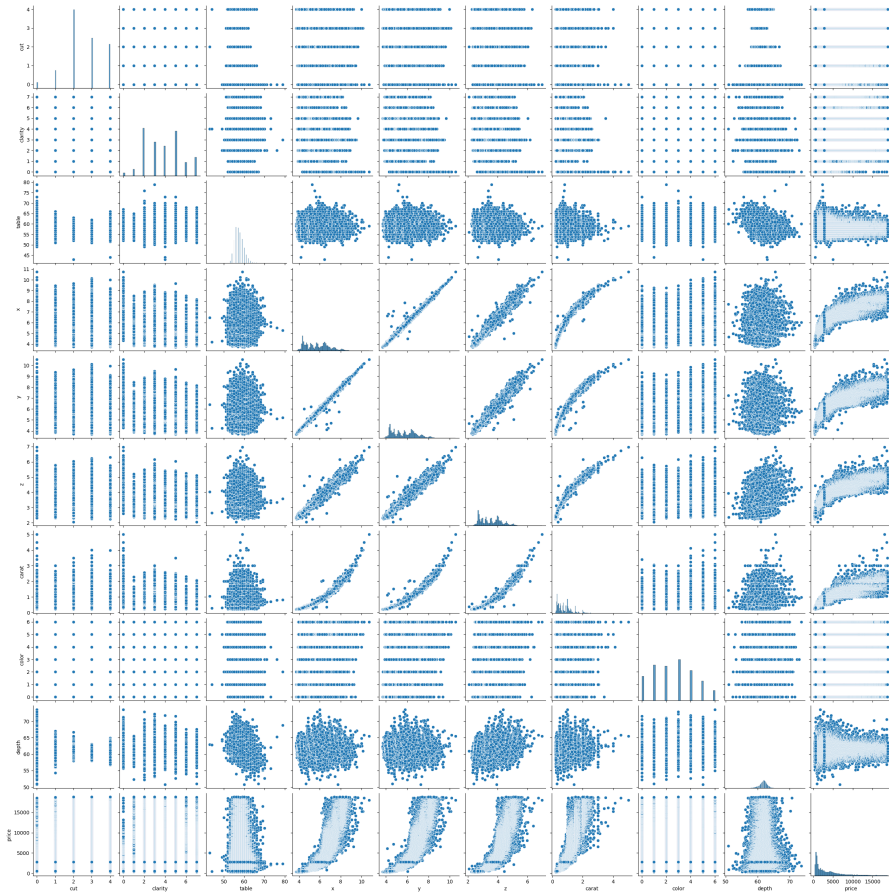


Figura 6: Diagramas de dispersión.

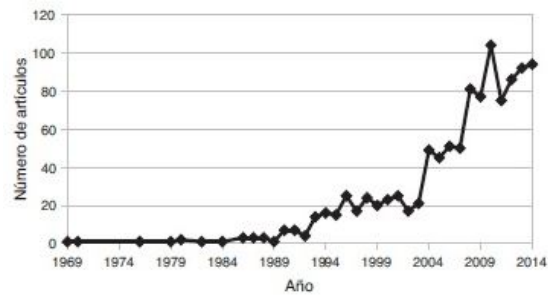


Figura 7: Aumento de artículos publicados con aprendizaje automático y diagnóstico médico [3]

4. Metodología

4.1. Método de k medias (clustering)

Esta técnica no supervisada consiste en generar $K \in N$ grupos para n elementos que incluyan a los n_k más cercanos (con base en cierta medida distancia, usualmente euclidiana) respecto a un centroide $c_k = (\bar{x}, \bar{y})$ tal que

$$\bar{x} = \frac{1}{n_k} \sum_{x_i \in S_k} x_i, \quad \bar{y} = \frac{1}{n_k} \sum_{y_i \in S_k} y_i \quad (1)$$

Formalmente, la distancia más cercana respecto a los centroides c_k se define como la minimización del error cuadrado para cada grupo:

$$SS_k = \sum_{i \in k} (x_i - \bar{x}_k)^2 + (y_i - \bar{y}_k)^2 \quad (2)$$

Este algoritmo es iterativo y tiene como función objetivo.

$$\min \sum_{i \in K} SS_k \quad (3)$$

4.2. Spectral Clustering

Spectral Clustering es un algoritmo de agrupamiento (clustering) que utiliza las propiedades del espectro (valores propios) de una matriz derivada de los datos para realizar la agrupación. A diferencia de otros métodos de clustering como k-means, que se basan en la minimización de una función objetivo específica (como la distancia dentro de los grupos), Spectral Clustering se basa en la teoría de grafos y la álgebra lineal [4].

4.2.1. Conceptos clave

- **Grafo de Similitud:** Los datos son representados como un grafo donde cada nodo representa un punto de datos y los bordes (con pesos) representan la similitud entre los puntos. La matriz de adyacencia A se construye a partir de este grafo, donde A_{ij} es la similitud entre los puntos de datos i y j .
- **Matriz Laplaciana:** Se define la matriz de grado D , una matriz diagonal donde D_{ii} es la suma de las similitudes del nodo i con todos los otros nodos. La matriz Laplaciana L se puede definir de varias formas, siendo una común $L = D - A$.
- **Descomposición en Valores Propios:** Se calcula la descomposición en valores propios de la matriz Laplaciana. Los vectores propios (eigenvectors) asociados con los menores valores propios (eigenvalues) son utilizados para reducir la dimensionalidad de los datos.

- **Agrupamiento:** Los datos son proyectados en el espacio de menor dimensión usando los vectores propios. Se aplica un algoritmo de clustering, como k-means, en este nuevo espacio de características para encontrar los grupos.

4.2.2. Ventajas de Spectral Clustering

- **Capacidad de Manejar Estructuras No Lineales:** Es eficaz en situaciones donde los clústeres no tienen una forma esférica, a diferencia de k-means.
- **Flexibilidad:** Puede ser adaptado para diferentes definiciones de similitud entre puntos de datos.
- **Relación con la Teoría de Grafos:** Proporciona una base teórica sólida para muchos problemas de agrupamiento.

4.2.3. Pasos en Spectral Clustering

1. Construcción del Grafo de Similitud:

- Seleccionar una medida de similitud adecuada (por ejemplo, distancia euclidiana, RBF kernel, k-nearest neighbors).
- Construir la matriz de similitud A .

2. Cálculo de la Matriz Laplaciana:

- Construir la matriz de grado D y la matriz Laplaciana L .

3. Descomposición en Valores Propios:

- Realizar la descomposición en valores propios de L y seleccionar los vectores propios correspondientes a los menores valores propios.

4. Agrupamiento en el Espacio Reducido:

- Utilizar los vectores propios seleccionados para proyectar los datos en un espacio de menor dimensión.
- Aplicar un algoritmo de clustering (como k-means) en este espacio para encontrar los clústeres.

5. Resultados

A continuación tenemos el código utilizado para el método de spectral clustering

```

import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import SpectralClustering
from sklearn.preprocessing import StandardScaler
from sklearn.manifold import TSNE

# Cargar los datos desde el archivo CSV
data = pd.read_csv("/content/diamantesCLEAN.csv")

# Extraer la variable "cut" y otras características relevantes
cut_data = data["cut"]
X = data.drop(columns=["cut"]) # Suponiendo que "cut" es la columna objetivo
y el resto son características

# Escalar los datos
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Aplicar Spectral Clustering
n_clusters = 5
spectral_clustering = SpectralClustering(n_clusters=n_clusters,
    affinity='nearest_neighbors', assign_labels='kmeans', random_state=42)
clusters = spectral_clustering.fit_predict(X_scaled)

# Agregar los resultados del clustering al DataFrame original
data['cluster'] = clusters

# Aplicar t-SNE para reducir la dimensionalidad a 2 componentes
tsne = TSNE(n_components=2, random_state=42)
X_tsne = tsne.fit_transform(X_scaled)

# Visualizar los resultados usando t-SNE
plt.figure(figsize=(10, 6))
plt.scatter(X_tsne[:, 0], X_tsne[:, 1], c=cut_data, cmap='gist_rainbow', s=10)
# Usar los valores reales de "cut" para colorear
plt.title('Visualización de t-SNE coloreada por la variable cut')
plt.xlabel('Componente 1')
plt.ylabel('Componente 2')
plt.colorbar(label='cut')
plt.savefig('tsne_colored_by_cut.png') # Guardar la figura
plt.show()

# Visualizar los resultados del clustering usando t-SNE
plt.figure(figsize=(10, 6))
plt.scatter(X_tsne[:, 0], X_tsne[:, 1], c=clusters, cmap='gist_rainbow', s=10)
# Usar los resultados del clustering para colorear

```

```
plt.title('Visualización de t-SNE después del clustering')
plt.xlabel('Componente 1')
plt.ylabel('Componente 2')
plt.colorbar(label='cluster')
plt.savefig('tsne_colored_by_clusters.png') # Guardar la figura
plt.show()
```

1. Carga y Preprocesamiento de Datos:

- a) Se carga el archivo CSV `diamante.csv`.
- b) Se extrae la columna `cut` y se eliminan del conjunto de datos las características que se usarán para el clustering.
- c) Los datos se escalan usando `StandardScaler` para normalizar las características.

2. Aplicación de Spectral Clustering:

- a) Se aplica el algoritmo de `Spectral Clustering` con `n_clusters=5` y se obtienen los clústeres.
- b) Se agregan los resultados del clustering al `DataFrame` original.

3. Aplicación de t-SNE:

- a) Se aplica `t-SNE` para reducir la dimensionalidad de los datos escalados a 2 componentes para facilitar la visualización.

4. Visualización:

- a) Se crea una gráfica t-SNE coloreada por la variable original `cut` figura 8.
- b) Se crea una gráfica t-SNE coloreada por los resultados del clustering figura 9.

6. Discusión

Viendo ambas representaciones, se puede deducir que las predicciones no han sido muy buenas. La figura de predicción 9 define 5 clusters más o menos separados. Sin embargo, nosotros conocemos las clases reales en la figura 8 y vemos que están muy mezcladas entre sí, sin ver clusters únicos ni definidos. Tal vez la única parte donde se pueden ver semejanzas sería en el centro de la figura donde está un poco revuelto y resalta más el color rojo.

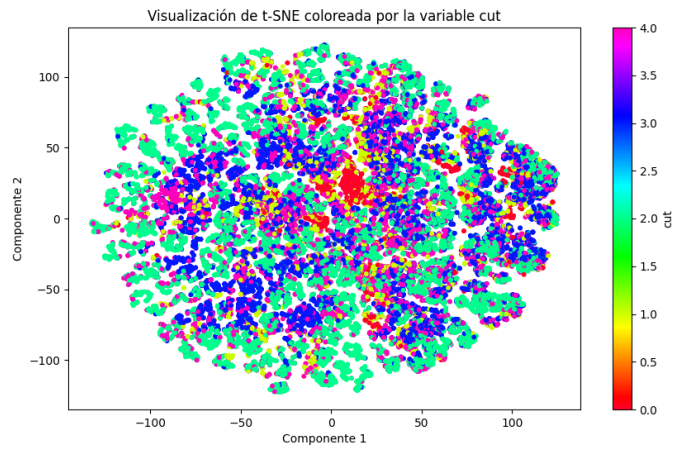


Figura 8: Gráfica t-SNE coloreada por la variable original cut

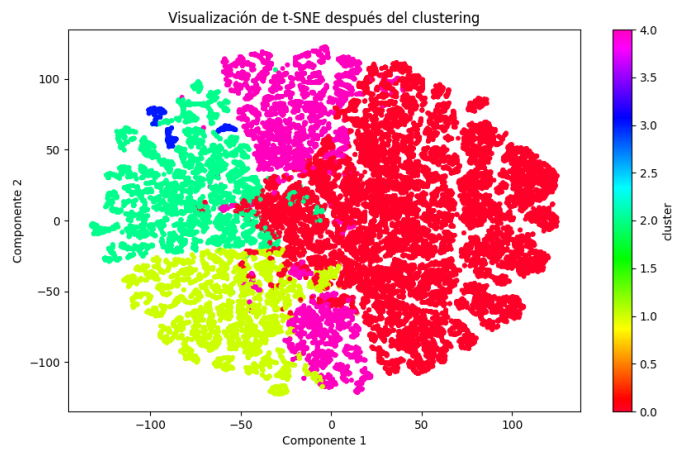


Figura 9: Gráfica t-SNE coloreada por los resultados del clustering.

Referencias

- [1] Isabel Cristina Pérez Verona and Leticia Arco García. Una revisión sobre aprendizaje no supervisado de métricas de distancia. *Revista Cubana de Ciencias Informáticas*, 10(4):43–67, 2016.
- [2] Jesús Cáceres Tello and Servicios Informáticos. Reconocimiento de patrones y el aprendizaje no supervisado. *Universidad de Alcalá, Madrid*, 2007.
- [3] Fabio A González. Modelos de aprendizaje computacional en reumatología. *Revista Colombiana de Reumatología*, 22(2):77–78, 2015.
- [4] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007.