UNIVERSIDAD DE GUADALAJARA
CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS

Análisis de algoritmos
Mtro. Jorge Ernesto Lopez Arce Delgado

Act. 03 Análisis de Clustering con TMAP en base de datos

Integrantes:
Juan Pablo Solis Regin - 220468416
Axxel Gael Hernandez Macias-219801705

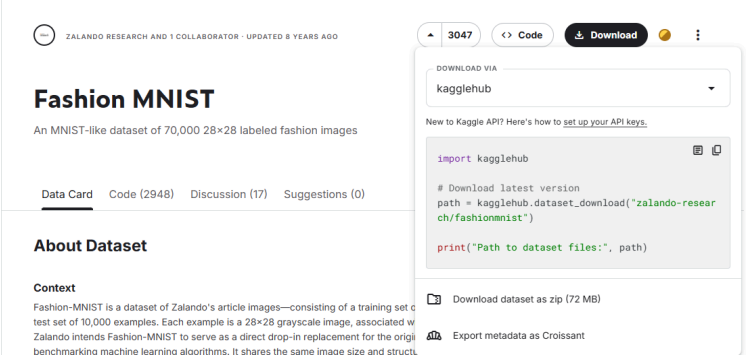
Introducción: La reducción de dimensionalidad es un método que se utiliza para representar un conjunto de datos utilizando un menor número de características (dimensiones) sin perder las propiedades significativas de los datos originales, lo que se traduce en eliminar características irrelevantes o redundantes para así crear un modelo con un número menor de variables. La importancia de esta se basa en lo siguiente:

- **Manejo de la "Maldición de la Dimensionalidad" (Curse of Dimentionality):** En espacios de alta dimensión, los datos se vuelven extremadamente dispersos. Esto significa que la distancia entre dos puntos cualesquiera tiende a ser casi la misma. Al reducir la dimensión, se recupera el significado de las distancias y se permite que los algoritmos de clustering identifiquen agrupaciones significativas.
- **Mejora de la Eficiencia Computacional:** Los algoritmos de clustering escalan mal con el número de dimensiones. Al reducir el número de variables, se disminuye drásticamente el tiempo de cálculo y los requisitos de memoria del algoritmo.
- **Visualización y Comprensión:** Es imposible visualizar datos con más de tres dimensiones, por lo que la reducción a 2 o 3 dimensiones permite graficar los datos y ver visualmente la separación de los *clusters* y la calidad del agrupamiento.

TMAP es una técnica de visualización y reducción de dimensionalidad basada en el concepto de gráficos de vecindad y teoría de grafos, similar a UMAP. Su principal valor en el análisis de clusters es su capacidad para preservar la estructura global y local de los datos de manera eficiente.

Objetivo: Por medio de esta actividad, se desea implementar una reducción de dimensionalidad de un algoritmo preparado acerca de prendas de ropa dispersas en un árbol, utilizando distintas librerías y documentación para centrarnos en el cluster que deseamos analizar, y hacer la identificación de sus respectivos sub clusters. A través de esto, lograremos separar los distintos datos de nuestro cluster elegido, y comprender cómo funciona más a fondo el algoritmo que lo compone.

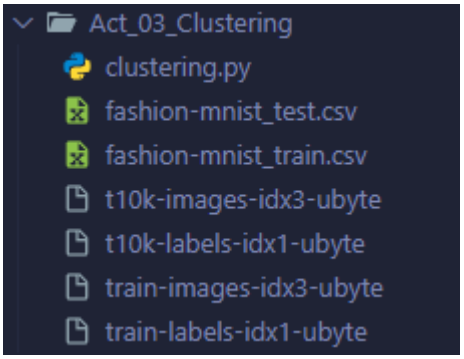
Desarrollo: Para empezar el desarrollo de esta actividad, necesitaremos primero descargar nuestra base de datos que vamos a utilizar, en este caso usaremos Fashion-MNIST desde Kaggle. Una vez que tengamos descargados los archivos necesarios, cargaremos el Archivo CSV en un DataFrame de Pandas en nuestro entorno.



Un archivo CSV (del inglés, Comma Separated Values) es cualquier archivo de texto en el cual los caracteres están separados por comas, haciendo una especie de tabla en filas y columnas, donde las columnas quedan definidas por cada punto y coma (;) mientras que las filas se definen mediante una línea adicional en el texto. Estos archivos sirven para manejar una gran cantidad de datos en formato de tabla, sin que ello conlleve un sobrecoste computacional adicional.

Después de cargar nuestro archivo CSV, deberemos separar las características (x) de las etiquetas (y).
Una vez realizado eso, seleccionamos un cluster de interés para hacer sus subclusters correspondientes. En nuestro caso decidimos seleccionar el cluster de camisas. Después, empezamos a seleccionar y filtrar las instancias correspondientes.
Para la implementación y creación de nuestro código hacemos lo siguiente:

1.- Importar los archivos de la base de datos a una carpeta nueva donde tendremos tambien nuestro codigo



2.- Desarrollamos el código, importando las librerías correspondientes. En nuestro caso decidimos utilizar umap en vez de tmpa debido a fallos con la ejecución del código con tmap.

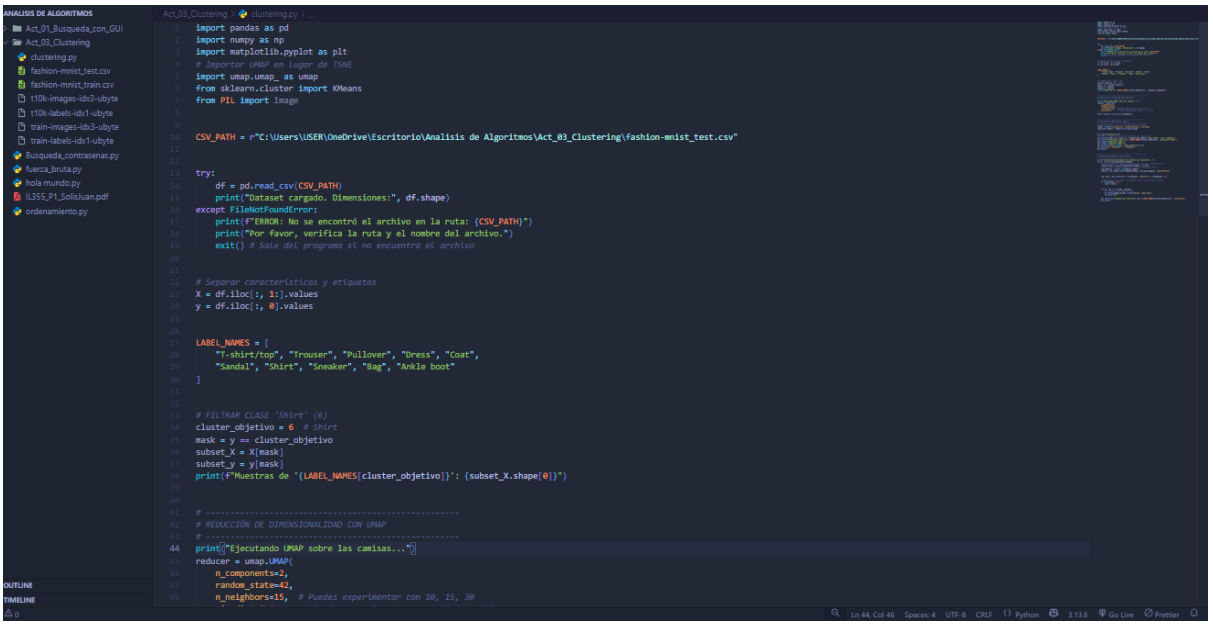
```python
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Importar UMAP en lugar de TSNE

import umap.umap_ as umap

from sklearn.cluster import KMeans
```

```
from PIL import Image


CSV_PATH = r"C:\Users\USER\OneDrive\Escritorio\Analisis de
Algoritmos\Act_03_Clustering\fashion-mnist_test.csv"


try:
    df = pd.read_csv(CSV_PATH)
    print("Dataset cargado. Dimensiones:", df.shape)
except FileNotFoundError:
    print(f"ERROR: No se encontró el archivo en la ruta:
{CSV_PATH}")
    print("Por favor, verifica la ruta y el nombre del archivo.")
    exit() # Sale del programa si no encuentra el archivo


# Separar características y etiquetas
X = df.iloc[:, 1:].values
y = df.iloc[:, 0].values


LABEL_NAMES = [
    "T-shirt/top", "Trouser", "Pullover", "Dress", "Coat",
    "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot"
]


# FILTRAR CLASE 'Shirt' (6)
```

```
cluster_objetivo = 6  # Shirt

mask = y == cluster_objetivo

subset_X = X[mask]

subset_y = y[mask]

print(f"Muestras de '{LABEL_NAMES[cluster_objetivo]}':
{subset_X.shape[0]}")



# ------------------------------------------------

# REDUCCIÓN DE DIMENSIONALIDAD CON UMAP

# ------------------------------------------------

print("Ejecutando UMAP sobre las camisas...")

# UMAP suele requerir menos parámetros de ajuste y es más rápido
que t-SNE

# n_neighbors y min_dist son parámetros clave para UMAP

# n_neighbors: Equilibra la estructura local vs. global (valores
más grandes = más global)

# min_dist: Controla cuán apretados están los puntos (valores más
bajos = puntos más juntos)

reducer = umap.UMAP(

    n_components=2,

    random_state=42,

    n_neighbors=15,  # Puedes experimentar con 10, 15, 30

    min_dist=0.1      # Puedes experimentar con 0.0, 0.1, 0.5

)

X_2d = reducer.fit_transform(subset_X)
```

```
# ----------------------------------------------------

# DETECCIÓN DE SUBCLUSTERS (KMeans)

# ----------------------------------------------------

# Aplicamos KMeans sobre la representación 2D de UMAP

kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)

subcluster_labels = kmeans.fit_predict(X_2d)



plt.figure(figsize=(7,6))

# Usamos un 'scatter plot' para visualizar los subclusters

plt.scatter(X_2d[:,0], X_2d[:,1], c=subcluster_labels,
cmap='tab10', s=15, alpha=0.7)

plt.title(f"Subclusters dentro de
'{LABEL_NAMES[cluster_objetivo]}' (proyección UMAP)")

plt.xlabel("Componente UMAP 1")

plt.ylabel("Componente UMAP 2")

plt.colorbar(ticks=range(3), label='Subcluster ID')

plt.grid(True, linestyle='--', alpha=0.5)

plt.show() #



# ----------------------------------------------------

# VISUALIZAR EJEMPLOS DE SUBCLUSTERS

# ----------------------------------------------------

print("\nVisualizando ejemplos de imágenes por subcluster...")
```

```
for sc in np.unique(subcluster_labels):

    # Encontrar hasta 5 índices aleatorios para una mejor
representación

    indices_todos = np.where(subcluster_labels == sc)[0]

    # Selecciona 5 índices aleatorios (o menos si hay menos de 5)

    num_ejemplos = min(5, len(indices_todos))

    indices = np.random.choice(indices_todos, size=num_ejemplos,
replace=False)



    fig, axes = plt.subplots(1, len(indices), figsize=(2 *
len(indices), 2))



    # Asegurarse de que 'axes' es iterable incluso si solo hay un
índice

    if len(indices) == 1:

        axes = [axes]



    for ax, idx in zip(axes, indices):

        # La imagen es de 28x28 píxeles

        ax.imshow(subset_X[idx].reshape(28,28), cmap='gray')

        ax.axis('off')



    plt.suptitle(f"Ejemplos del Subcluster {sc}
({LABEL_NAMES[cluster_objetivo]})", fontsize=14)

    plt.show()
```

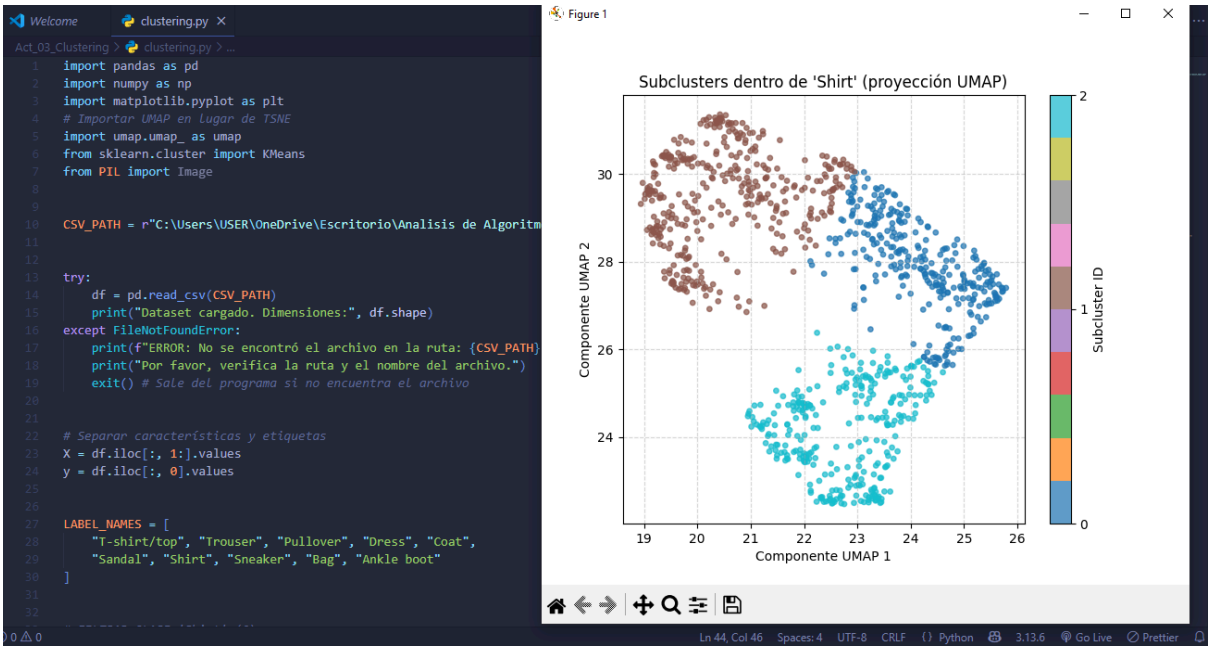


```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# Importar UMAP en lugar de TSNE
import umap.umap_ as umap
from sklearn.cluster import KMeans
from PIL import Image


CSV_PATH = r"C:\Users\USER\OneDrive\Escritorio\Analisis de Algoritmos\Act_03_Clustering\fashion-mnist_test.csv"


try:
    df = pd.read_csv(CSV_PATH)
    print("Dataset cargado. Dimensiones:", df.shape)
except FileNotFoundError:
    print(f"ERROR: No se encontró el archivo en la ruta: {CSV_PATH}")
    print("Por favor, verifica la ruta y el nombre del archivo.")
    exit() # Sale del programa si no encuentra el archivo


# Separar características y etiquetas
X = df.iloc[:, 1:].values
y = df.iloc[:, 0].values


LABEL_NAMES = [
    "T-shirt/top", "Trouser", "Pullover", "Dress", "Coat",
    "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot"
]


# FILTRAR CLASE 'Shirt' (6)
cluster_objetivo = 6  # Shirt
mask = y == cluster_objetivo
subset_X = X[mask]
subset_y = y[mask]
print(f"Muestras de '{LABEL_NAMES[cluster_objetivo]}': {subset_X.shape[0]}")


# ----------------------------------------------------
# REDUCCIÓN DE DIMENSIONALIDAD CON UMAP
# ----------------------------------------------------
print("Ejecutando UMAP sobre las camisas...")
reducer = umap.UMAP(
    n_components=2,
    random_state=42,
    n_neighbors=15,  # Puedes experimentar con 10, 30
```

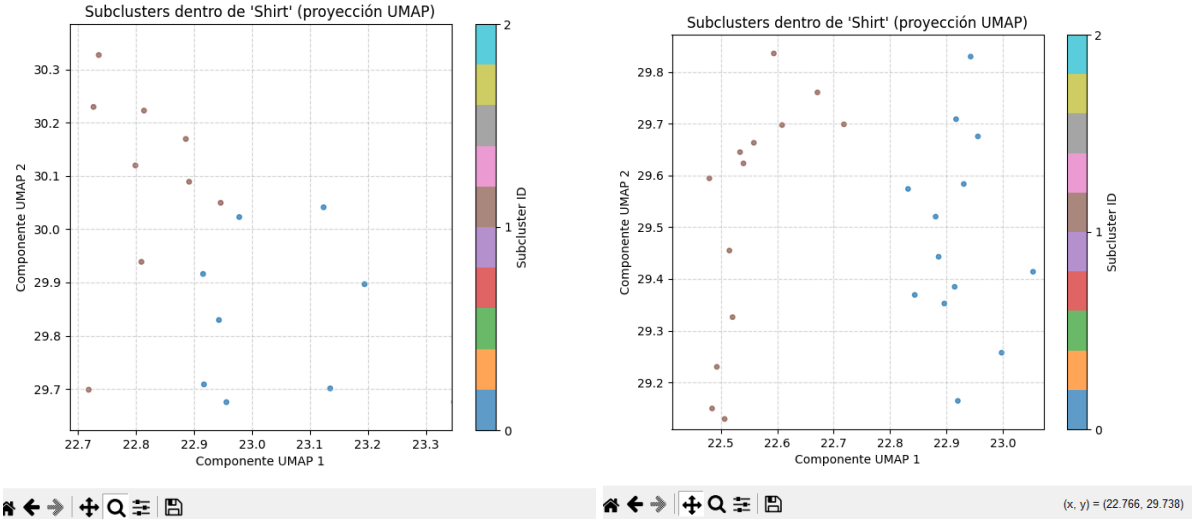

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# Importar UMAP en lugar de TSNE
import umap.umap_ as umap
from sklearn.cluster import KMeans
from PIL import Image


CSV_PATH = r"C:\Users\USER\OneDrive\Escritorio\Analisis de Algoritmos\Act_03_Clustering\fashion-mnist_test.csv"
```

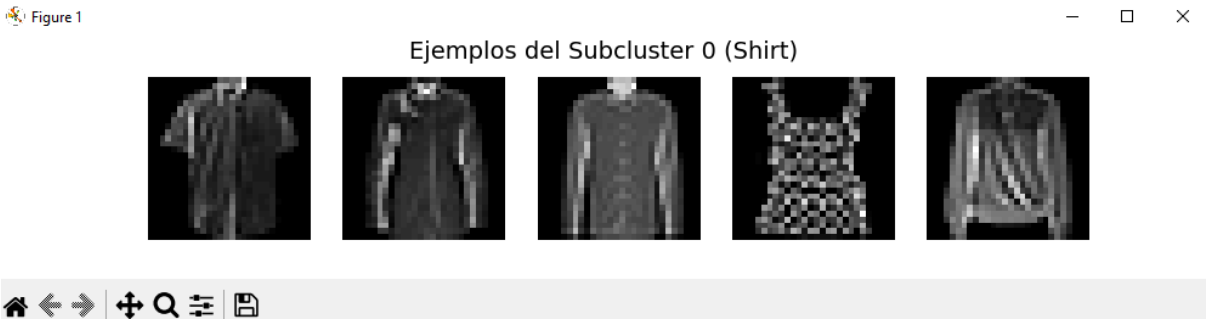

Usamos las librerías correspondientes al igual que definir la ruta de acceso para nuestro archivo CSV

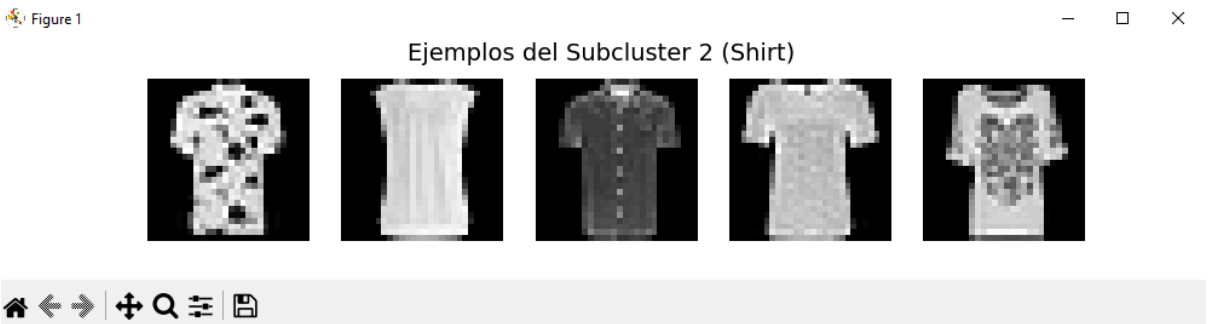
3.- Ejecutar el código para visualizar nuestro mapa.



A través de las herramientas que se encuentran en nuestro entorno grafico podemos modificar y visualizar mejor los subclusters, desde hacer zoom a un area en especifico, como movernos por el mapa







Como se puede observar, se visualizan algunos ejemplos de nuestros subclusters con algunas de las imágenes que contienen

Conclusión:
El uso de un algoritmo de mapeo nos permitió el poder hacer ramificaciones y búsqueda más completa y visualizar de mejor manera los datos de nuestro archivo CSV, haciéndolo más eficiente y práctico. También, nos permite entender más acerca del uso del clustering en algoritmos para desarrollar búsquedas o tareas complejas, que con otros algoritmos como los de fuerza bruta podrían ser más tardados o confusos de entender. En cambio, con una ayuda visual del mapa, nos permite conocer más sobre futuros usos de este algoritmo a nuestras prácticas. Una práctica muy interesante que nos deja muchos aprendizajes nuevos.

Fuentes de consulta:

- Ph.D, J. M., & Kavlakoglu, E. (2024, January 5). *Reducción de la dimensionalidad*. Ibm.com. https://www.ibm.com/mx-es/think/topics/dimensionality-reduction
- Pykes, K. (2024, February 21). *Tutorial pandas read_csv(): importación de datos*. Datacamp.com; DataCamp. https://www.datacamp.com/es/tutorial/pandas-read-csv
- *¿Qué es un archivo CSV y para qué sirve? [Octubre 2021]*. (n.d.). GEEKNETIC. https://www.geeknetic.es/Archivo-CSV/que-es-y-para-que-sirve
- *What is a UMAP?* (n.d.). Allen Institute. https://alleninstitute.org/resource/what-is-a-umap/
- *Fashion MNIST*. (n.d.). Www.kaggle.com. https://www.kaggle.com/datasets/zalando-research/fashionmnist
- Probst, D. (2025). *tmap - Visualize big high-dimensional data*. Gdb.tools. https://tmap.gdb.tools/
- Coenen, A., & Pearce, A. (n.d.). *Understanding UMAP*. Pair-Code.github.io. https://pair-code.github.io/understanding-umap/
- *La biblioteca estándar de Python*. (2025). Python Documentation. https://docs.python.org/es/3.13/library/index.html