Desafío - Dimensionalidad y agrupación (II)

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el `.zip` en el LMS.
- Desarrollo desafío:
    - El desafío se debe desarrollar de manera Individual.
    - Para la realización del desafío necesitarás apoyarte del archivo *Apoyo Desafío - Dimensionalidad y agrupación (II)*

Desafío 1: Preparación del ambiente de trabajo

Para este desafío trabajaremos de manera conjunta identificando la paleta de colores de carátulas de álbumes.

- Las imágenes se encuentran en una carpeta con el nombre `album_covers`.
- Cada imagen tiene la siguiente nomenclatura: `artista-nombre-del-album.jpg`.

El objetivo es generar un método que nos permita identificar la dominancia de una cantidad finita de colores.

Para importar imágenes y visualizarlas, vamos a importar las siguientes librerías:

- Comencemos por incluír las librerías clásicas: `pandas`, `numpy` y `matplotlib.pyplot`.
- `sklearn.cluster.KMeans`: para extraer los principales componentes de una matriz numérica.
- `skimage.io`: Para poder ingresar y leer imágenes.

Desafío 2: Importación de imágenes

- Comencemos por ingresar una imagen a nuestro ambiente de trabajo. Para ello ocuparemos `io.imread`. ¿Qué devuelve?
- Para visualizar la imágen en el notebook, ocupe `io.imshow`.

Desafío 3: Preprocesamiento de imágenes y KMeans

- Con la representación numérica de la imágen, vamos a extraer la altura, el ancho y la cantidad de canales mediante `shape`.
- Posteriormente redimensionaremos la imágen con reshape.
- Partamos por inicializar nuestro algoritmo `KMeans` con un `k=8`, ¿Qué significa esto?
- Vuelva a implementar el mismo algoritmo con `MiniBatchKMeans`. ¿Qué diferencia existe con `KMeans`?

Desafío 4: Extracción de valores

- Ahora extraemos las etiquetas predichas con `labels_`. Hasta el momento las etiquetas hacen referencia a cada centroide. Para imputar sentido en estos, debemos extraer los valores de los centroides.
- Para extraer los centroides (valores característicos), utilizamos el atributo `cluster_centers_`.
- Con las etiquetas, generamos un conteo de ocurrencia con `np.unique`. Para extraer el conteo, debemos implementar la opción `return_counts=True`.

Desafío 5: Conversión rgb a hex

- Con los centroides, vamos a convertirlos a formato hexadecimal. Vamos a generar una función y la pasaremos con `map` por cada centroide.

Desafío 6: Definición de base

Ahora generamos un DataFrame con las siguientes variables:

- El color `hex`.
- La ocurrencia del color en cada pixel `count`.
- El porcentaje de ocurrencia de cada color respecto a `cluster_centers_`.

Posteriormente ordenaremos los colores de forma descendente por el porcentaje de ocurrencia.

Desafío 7: Visualización

Genere un gráfico de barras donde presente el porcentaje de cada color. Las barras deben estar coloreadas con el color inferido.

Bonus point: Envuelva todo en una función