Aprendizaje automático

Practica 7: Agrupamiento

Universidad de Zaragoza, curso 2022/2023

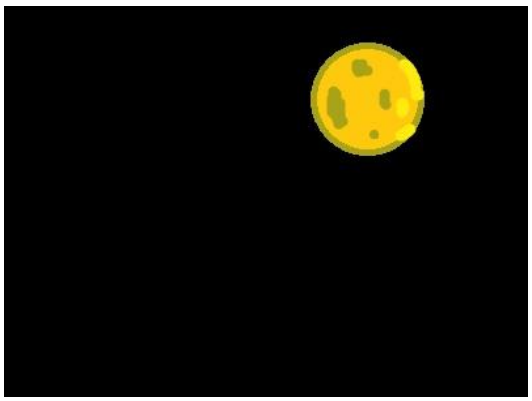
Juan Eizaguerri Serrano 816079

Implementación del algoritmo K-means

Con el objetivo de aplicarlo al problema de la compresión de imágenes, se implementa el algoritmo K-means, que dado un conjunto de datos X de n dimensiones, agrupa estos datos en K clusters. Una vez terminada la ejecución del algoritmo, cada dato pertenece al cluster con media μ_k más cercana.

En este problema los datos representan los píxeles de la imagen, siendo el tamaño de estos *ancho_img * alto_img*, es decir, el número de píxeles. Además, cada dato tendrá 3 atributos para representar el valor de los canales RGB de ese píxel de la imágen.

En primer lugar, se inicializan los K clusters en datos aleatorios. Esto podría causar problemas en imágenes como la de la siguiente figura con un color uniforme y un detalle en otros colores que ocupa poca parte de la imágen.



La mayoría de los colores se centran en la pequeña luna, sin embargo, la gran mayoría de los píxeles son negros, por lo que es mucho más probable que los clusters se inicien con media en el color negro, esto provoca que se den mínimos locales durante la ejecución del algoritmo y no se generen los clusters adecuados para la imágen.

Para solucionar este problema, al elegir los valores aleatorios hay que asegurar que su valor es único para no repetir píxeles con el mismo color.

Una vez iniciados los centroides, se entra en el bucle de iteraciones del algoritmo K-means:

- Para cada dato, asignarle el cluster más cercano (c_i).
- Para cada cluster, actualizar su centroide μ_k a la media de los datos que pertenecen a dicho cluster.

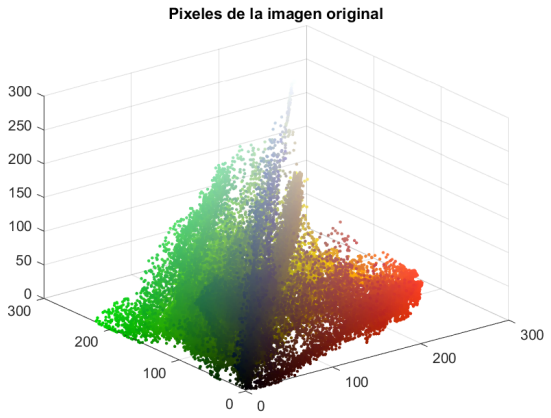
Esto se repite hasta que c_i no cambia para ningún dato i.

K-means en cuantificación de imágenes

El objetivo es agrupar los colores en K clústers y asignar a cada píxel de la imágen el valor del centroide del clúster al que pertenece.

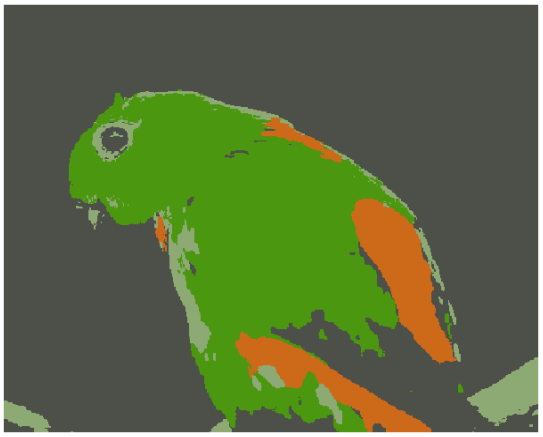
Se cargan los datos de la imágen en una matriz de m filas y 3 columnas, donde m es el número de píxeles de la imágen y las columnas almacenan los valores RGB.

Se puede representar el espacio de colores de la imágen.

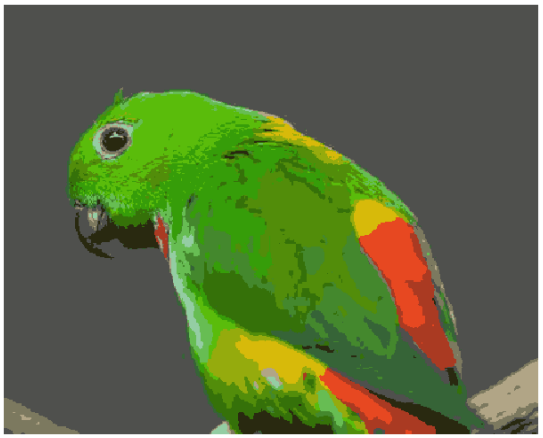


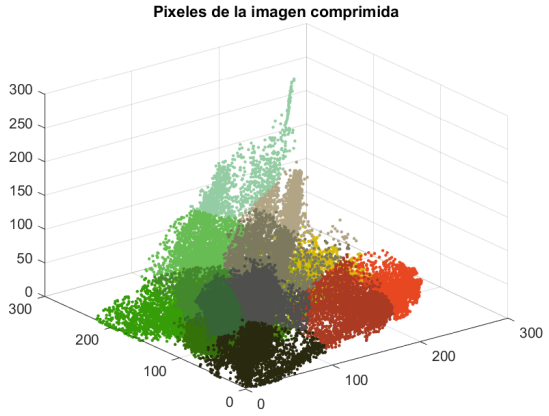
Se utiliza K-means para agrupar los colores de la imágen y asignar un clúster a cada píxel de la imágen. Una vez hecho eso, se puede reconstruir la imágen para comprobar el error de reconstrucción. También se visualiza el clúster al cual se asigna cada color de la imágen. Estos son los resultados con distintos valores de K.

K=4

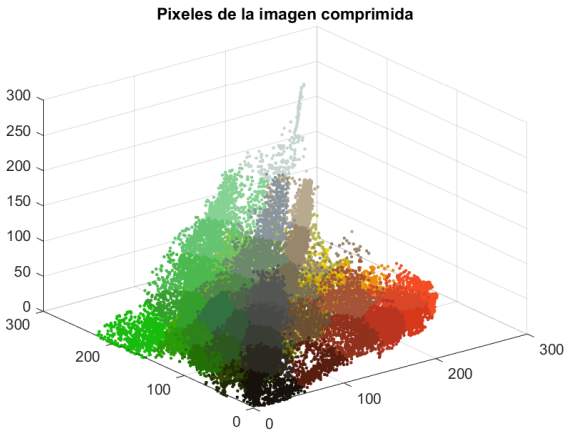


K=16





K=64

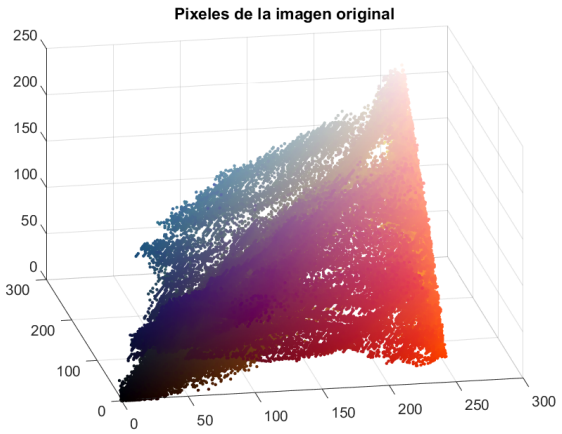


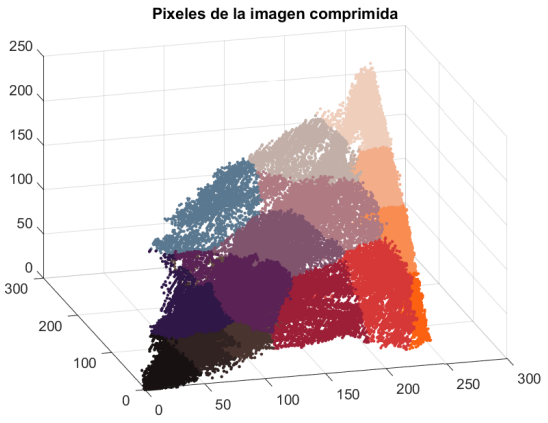
Se observa que para mayores valores de K, al crearse más agrupamientos, los valores de los píxeles tienen más probabilidad de aproximarse a un clúster. Esto significa que la imágen se parece más a la original.

Se puede medir el error con la función de distorsión, que calcula la media de las distancias cuadradas del valor real al valor asignado para cada píxel de la imágen. Representa el error de reconstrucción de los datos.

	Función de distorsión (J)
Loro (K = 4)	1655.53
Loro (K = 16)	525.16
Loro (K = 64)	136.87

El valor ideal de K depende de la imágen utilizada y la calidad del resultado que se quiere obtener. Una imágen que utiliza muchos colores distintos necesitará más agrupamientos que una que utilice menos si se quiere mantener un error bajo. A continuación se muestra una imágen con un espacio de colores más complicado de agrupar y el resultado obtenido al aplicar K-means con K=16.

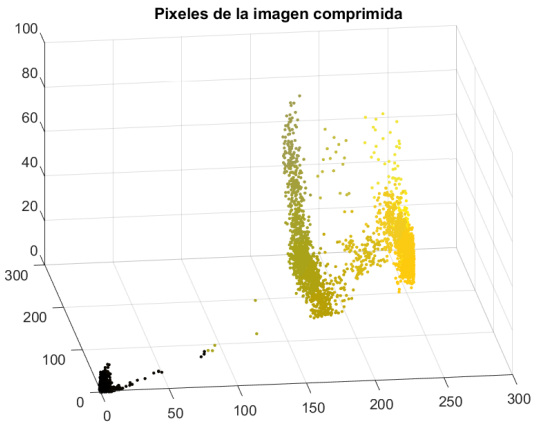




	Función de distorsión (J)
Paisaje(K = 16)	673.77

Se observa que utilizando el mismo valor de K, el resultado en este caso es mucho peor que para las pruebas anteriores. Visiblemente la imágen pierde mucha calidad, especialmente en el degradado del cielo, y la función de distorsión es mucho mayor.

Para la imágen de la luna del apartado anterior, 16 clústers son más que suficientes para representar los colores de la imágen.



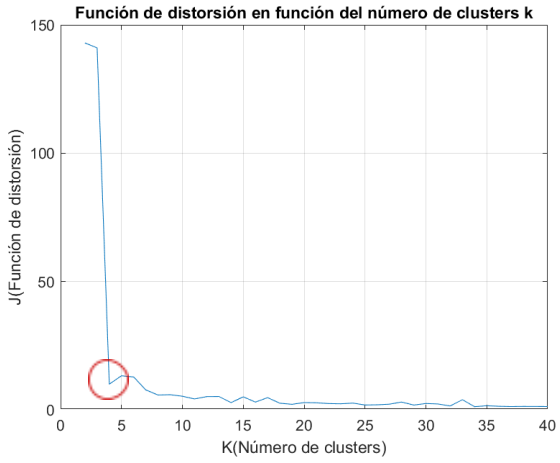
	Función de distorsión (J)
Luna(K = 16)	3.80

Como es de esperar, apenas se pierde calidad. Cuatro clústers hubiesen sido suficientes para representar la imágen.

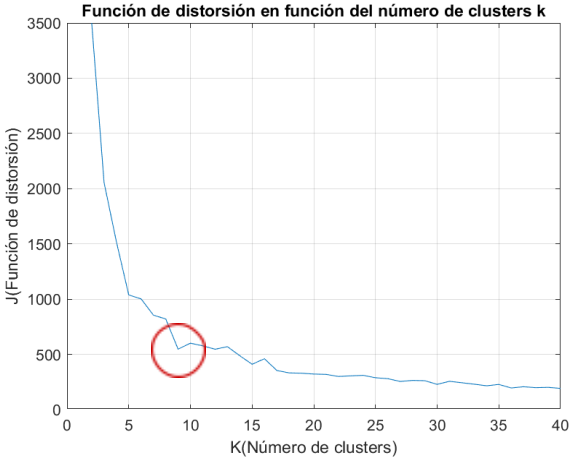
A mayor número de clústers la imágen siempre tendrá mayor calidad y mejor función de distorsión, pero disminuye la compresión.

Si el objetivo es minimizar el error de distorsión manteniendo el número de Clústers más bajo posible para aumentar la ganancia de compresión, se puede utilizar el método del codo. Para aplicarlo, se muestra la función de distorsión en función del parámetro K y se busca el “codo” de la curva que genera. Se han probado valores de K entre 0 y 40 para las 3 imágenes utilizadas en el desarrollo de esta práctica.

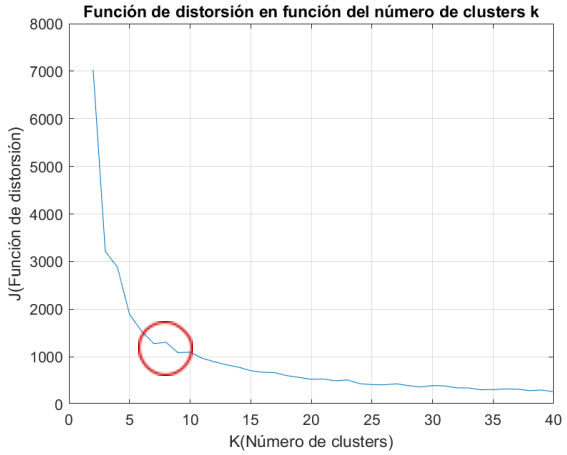
Luna



Loro



Paisaje



Para la imágen de la luna, el codo es muy visible y a partir de K = 4 la distorsión apenas disminuye. Para las imágenes del loro y el paisaje, la curva es menos pronunciada, pero se puede encontrar un codo en torno a K=8 y K=7, respectivamente.

Conclusiones

En el desarrollo de esta práctica se ha demostrado que es posible utilizar el algoritmo K-means para cuantificar imágenes, reduciendo su espacio de colores al del parámetro K (Número de clusters). Se ha utilizado este método con distintas imágenes evaluando sus resultados mostrando los nuevos espacios de color tras el agrupamiento, reconstruyéndo las imágenes y calculando la función de distorsión.

También se ha tratado el problema de buscar un valor óptimo del parámetro K mediante el método del codo sobre la función de distorsión.

A modo de resumen, se muestran los resultados tomados a lo largo de la práctica.

Imágen(K)	**Función de distorsión (J)**
Loro (K = 4)	1655.53
Loro (K = 16)	525.16
Loro (K = 64)	136.87
Paisaje(K = 16)	673.77
Luna(K = 16)	3.80
Loro(K = 8)	815.39
Luna(K = 4)	140.40
Paisaje(K = 7)	1275.23