Kristian Stoyanov Belkin, Paul Daniel Pop i Raúl Jiménez Ayza

NIU: 1667799, 1607129 i 1688916

**Proyecto 2 Parte 3 ETIQUETATGE Inteligencia artificial**

Contenido

[Introducción 2](#_Toc372326377)

[Funciones de análisis cualitativo 3](#_Toc395352342)

[Retrieval\_by\_color: 3](#_Toc1215118155)

[Test 1 4](#_Toc1398083906)

[Test 2 4](#_Toc1794427276)

[Retrieval\_by\_shape: 5](#_Toc1723024038)

[Test 1 5](#_Toc952257845)

[Test 2 6](#_Toc1767001859)

[Retrieval\_combined: 7](#_Toc1281271898)

[Test 1 7](#_Toc253954595)

[Test 2 8](#_Toc1881052663)

[Conclusión 8](#_Toc1942139393)

[Funciones de análisis cuantitativo 9](#_Toc1097025487)

[Get\_shape\_accuracy: 9](#_Toc1992802112)

[Get\_color\_accuracy: 9](#_Toc1689294185)

[Kmean\_knn\_statistics: 10](#_Toc298867056)

[Mejoras o cambios K-means y KNN 12](#_Toc1588287487)

[Métodos de inicialización de centroides para K-means 12](#_Toc1623830430)

[K-means ++ 13](#_Toc2023194567)

[Inicialización Aleatoria 14](#_Toc1684372248)

[Pruebas de rendimiento 14](#_Toc1680771283)

[Test 1 14](#_Toc1743360878)

[Test 2 16](#_Toc1316850677)

[Test 3 18](#_Toc1553530961)

[Conclusión 21](#_Toc1364822925)

[Coeficiente de Fisher 22](#_Toc233762913)

[Find\_BestK: 30](#_Toc1975023940)

[Test 1 34](#_Toc1532470086)

[Test 2 34](#_Toc523487747)

[Test 3 35](#_Toc176003295)

[Test 4 35](#_Toc2130496886)

[Test 5 36](#_Toc843515158)

[Test 6 37](#_Toc1926056856)

[Conclusiones Find\_best\_K 39](#_Toc30493650)

[Conclusiones Globales 39](#_Toc1925865236)

# Introducción

En este segunda practica tenemos la tarea general de realizar un etiquetaje automático de imágenes de ropa, a través de nuestro código deberíamos poder asignar etiquetas a diferentes tipos de ropa (8 en total) con sus respectivos colores (11 en total).

Para realizar estas tareas utilizamos dos archivos para dos algoritmos llamados KNN (K-nearest neighbors o k vecinos más próximos) y K-means (K-means clustering o K-medias). A continuación, explicaremos un poco más en detalle ambos algoritmos:

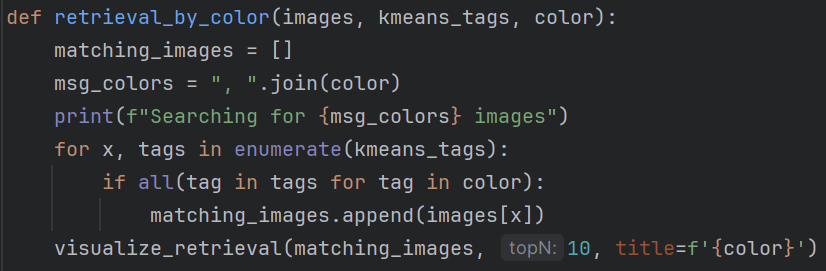
K-means es un método de clasificación no supervisada, que utilizamos para generar clústeres a partir de la información que tenemos de los colores, para poder agrupar los pixeles de cada imagen y así determinar los colores predominantes.

KNN a diferencia del K-means es un método de clasificación supervisado, que utilizamos para encontrar el tipo de ropa de cada imagen, lo que acaba haciendo es buscar K vecinos más cercanos y se asigna el tipo mirando por la mayoría de los vecinos más cercanos.

# Funciones de análisis cualitativo

## Retrieval\_by\_color:

La función retrieval\_by\_color es un buscador que permite encontrar una serie de imágenes a partir de los valores color introducidos.

*Código de la función*

Esta función recibe como parámetros:

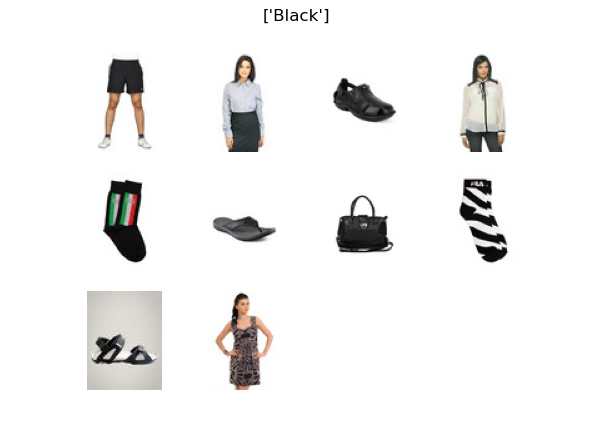
* Images: el set de imágenes sobre las cuales queremos buscar.
* Kmeans\_tags: las predicciones de color realizadas por el algoritmo K-means previamente sobre el set de imágenes.
* Color: el color de las imágenes que queremos encontrar.

Respecto al código, cada sub-lista dentro de la lista kmeans\_tags representa los colores predichos para una imagen. Para encontrar las imágenes que presentan el color que estamos buscando simplemente recorremos la lista de kmeans\_tags y miramos si algun color coincide con el color buscado, si es así nos guardamos esa imagen.

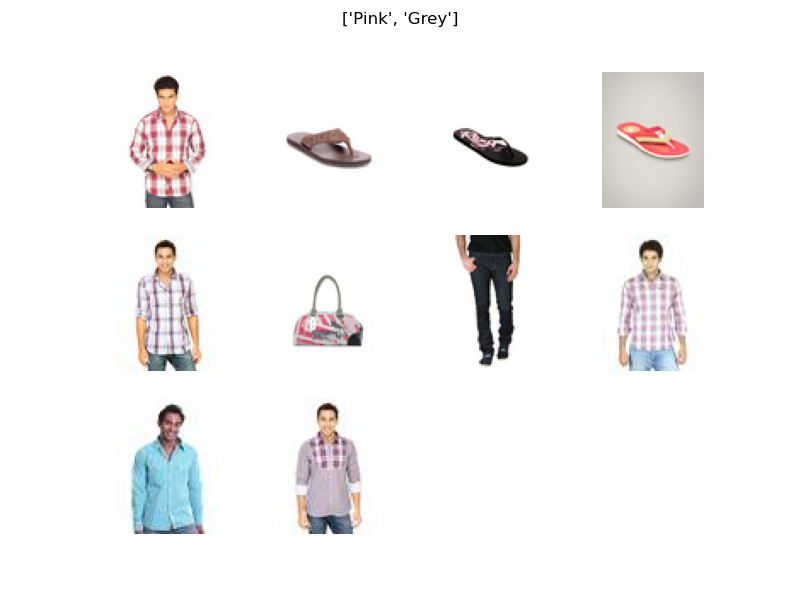
A continuación, exploraremos algunos de los resultados obtenidos al ejecutar la función con diferentes criterios de color. Para estas pruebas se ha usado 1000 imágenes del set de entrenamiento.

### Test 1

Buscando prendas negras:



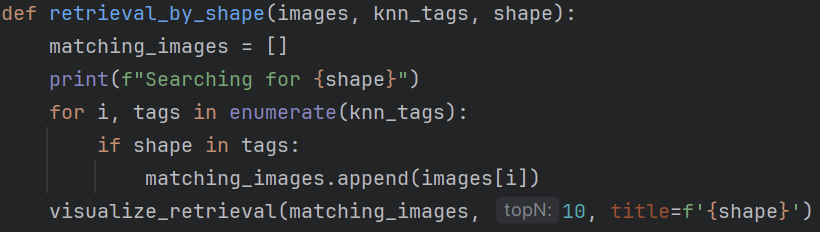
### Test 2

Buscando prendas rosas y grises:

Para todos estos test se ha conseguido una exactitud global del 96% usando el algoritmo de K-means. No obstante, hay que considerar que, para ciertas imágenes, colores como el negro, naranja, gris o blanco pueden estar presentes en el fondo de la imagen, en el cabello de la persona o en su tono de piel, por eso en algunas predicciones hay ciertos colores que no están presentes en la prenda.

## Retrieval\_by\_shape:

La función retrieval\_by\_shape es muy similar a la función vista anteriormente, pero en este caso hace la búsqueda por tipo de prenda en vez de por color.

*Código de la función*

Al igual que antes, la función recibe por parámetros las imágenes a evaluar, las predicciones del algoritmo KNN y el tipo de prenda que queremos encontrar.

A continuación están los resultados de haber ejecutado la función sobre el mismo conjunto de 1000 imágenes.

### Test 1

Buscando vaqueros:



### Test 2

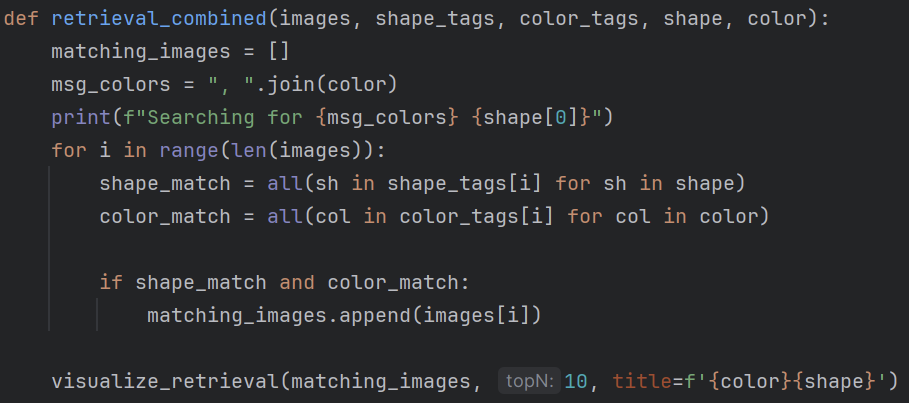
Buscando vestidos:



Para todos estos test se ha conseguido una exactitud global del 91% usando el algoritmo de KNN. En la primera prueba hubo algunos fallos, donde se clasificaron 2 prendas incorrectamente como vaqueros, cuando eran shorts pero en las otras pruebas las predicciones parecen bastante buenas. Aunque el porcentaje de exactitud haya sido menor comparado con la clasificación por color, evaluando los resultados visualmente parece ser que el KNN es más preciso.

## Retrieval\_combined:

La función retrieval\_combined es la combinación de las dos funciones vistas anteriormente, permite buscar por tipo de prenda y por color al mismo tiempo. Para este caso pasaremos directamente a las pruebas realizadas.



*Código de la función*

### Test 1

Buscando vestidos negros:



### Test 2

Buscando camisas blancas y naranjas



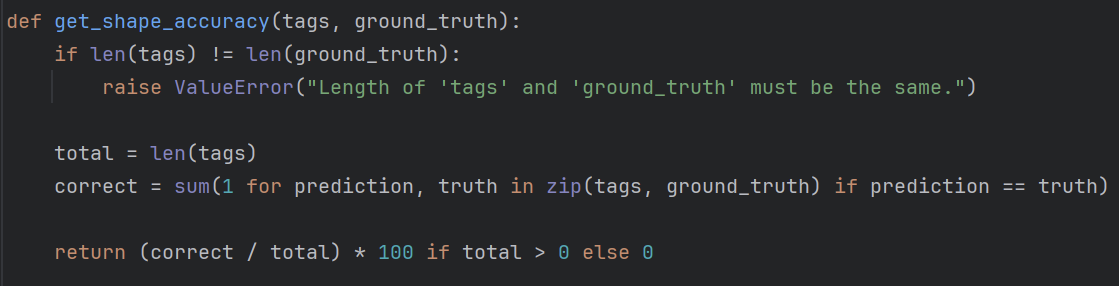
## Conclusión

Observando los resultados de estas pruebas cualitativas podemos llegar a la conclusión de que, aunque el porcentaje de éxito sea mayor en la clasificación por color, las predicciones de tipo de prenda parecen ser mucho más acertadas ya que no dependen del color de la imagen. Esto se puede ver al usar la función retrieval\_combined donde el tipo de prenda se predice bastante bien, pero los colores se confunden mucho con los colores de la piel de los modelos y con el color del fondo de las imágenes.

# Funciones de análisis cuantitativo

## Get\_shape\_accuracy:

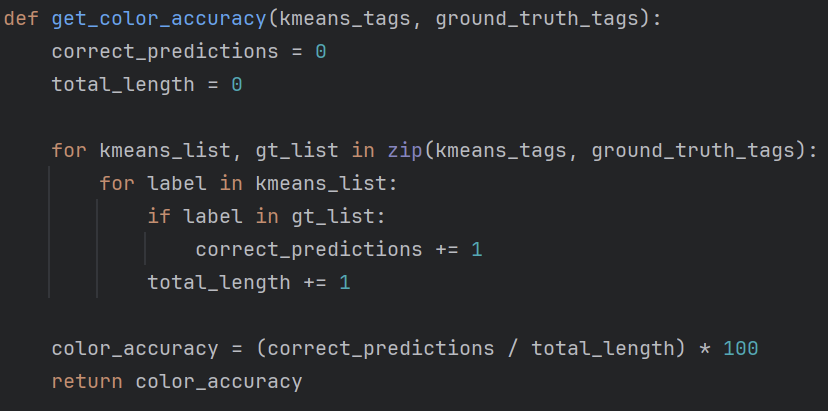
Esta función nos servirá para validar los tags que hemos podido obtener del algoritmo KNN.



Lo primero que hacemos en esta función es comprobar que las listas de los tags sean iguales en tamaño, en caso de no ser así provocamos un error. Guardamos en total la longitud para posteriormente sacar el porcentaje y procedemos a comprobar si las predicciones son correctas comparándolas con el “ground truth”, si la predicción es correcta incrementamos el número de predicciones correctas. Finalmente dividimos las predicciones correctas entre las totales y multiplicamos por 100 para dar un porcentaje.

## Get\_color\_accuracy:

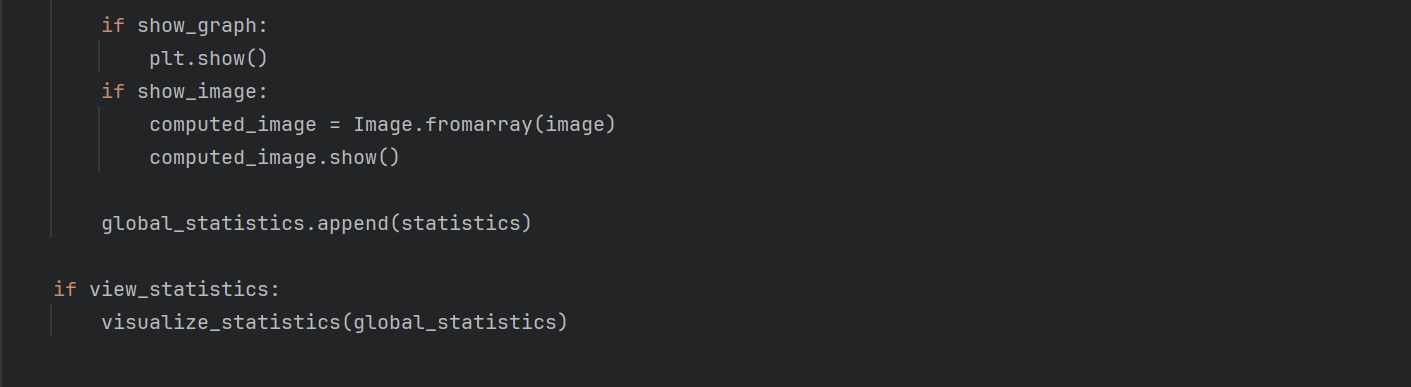
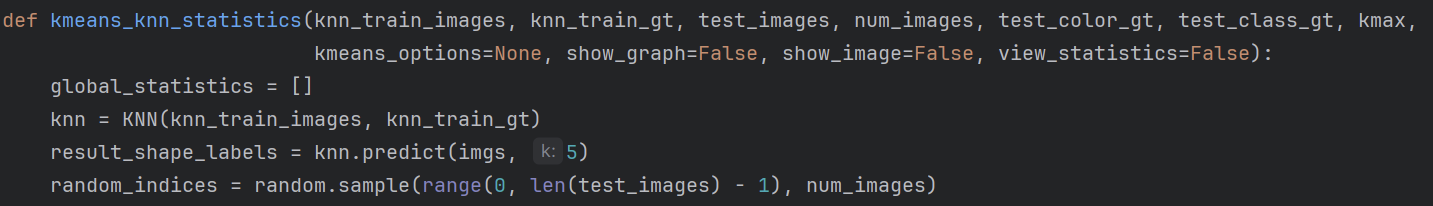
Muy parecida a la función anterior, ésta nos servirá para validar los tags que hemos obtenido del algoritmo K-means.



Al igual que antes, comparamos los valores predichos con el “ground truth” y en el caso de que coincidan aumentamos el número de predicciones correctas y calculamos el valor del accuracy.

Ambas funciones anteriores han sido utilizadas en las pruebas cualitativas para saber el porcentaje de aciertos de los algoritmos K-means y KNN. También serán utilizadas en la función cuantitativa que se explicará a continuación.

## Kmean\_knn\_statistics:



La función kmeans\_statistics ejecuta el algoritmo K-means y KNN en un conjunto de imágenes para evaluar y comparar su rendimiento según varias medidas. A continuación se describen los parámetros de entrada de la función y las diferentes medidas calculadas.

**Parámetros de entrada**:

* **Knn\_train\_images:** Conjunto de imágenes de entrenamiento para el algoritmo KNN.
* **Knn\_train\_gt:** Etiquetas de “ground truth” de las clases correspondientes a las imágenes de entrenamiento.
* **Test\_images:** Conjunto de imágenes que se desean clasificar.
* **Num\_images:** Número de imágenes a seleccionar aleatoriamente para el análisis.
* **Test\_color\_gt:** Etiquetas de “ground truth” de los colores para las imágenes a clasificar.
* **Test\_class\_gt:** Etiquetas de “ground truth” del tipo de prenda (clases) para las imágenes a clasificar.
* **Kmax:** Valor máximo de K (número de clusters) a considerar en el algoritmo K-means.
* **Kmeans\_options:** Diccionario con opciones adicionales para la inicialización y configuración del algoritmo K-means.
* **Show\_graph:** Booleano que indica si se deben mostrar gráficos en 3D de las nubes de puntos.
* **Show\_image:** Booleano que indica si se deben mostrar las imágenes procesadas.
* **View\_statistics:** Booleano que indica si se deben visualizar las estadísticas globales mediante gráficos.

**Estadísticas calculadas:**

* **K:** Número de clusters utilizado en K-means.
* **Init\_method:** Método de inicialización utilizado para K-means.
* **WCD:** Distancia dentro de la clase (Within Class Distance), que mide la compactación de los clusters.
* **Num\_iterations:** Número de iteraciones que el algoritmo K-means necesitó para converger.
* **Convergence\_time:** Tiempo total (en segundos) que tomó el algoritmo K-means para converger.
* **Found\_color:** Colores encontrados por K-means en la imagen.
* **Color\_gt:** Colores de “ground truth” de la imagen.
* **Color\_accuracy:** Precisión de los colores encontrados en comparación con los colores de "ground\_truth".
* **Found\_shape:** Tipo de prenda(clases) encontradas por el clasificador KNN.
* **Shape\_gt:** Clases de "ground\_truth” de la imagen.
* **Shape\_accuracy**: Precisión de las formas encontradas en comparación con las formas de "ground\_truth".

Esta función será utilizada en los próximos experimentos sobre las mejoras realizadas al algoritmo K-means.

# Mejoras o cambios K-means y KNN

Estas son las diferentes mejoras y pruebas que hemos planteado para nuestros algoritmos:

## Métodos de inicialización de centroides para K-means

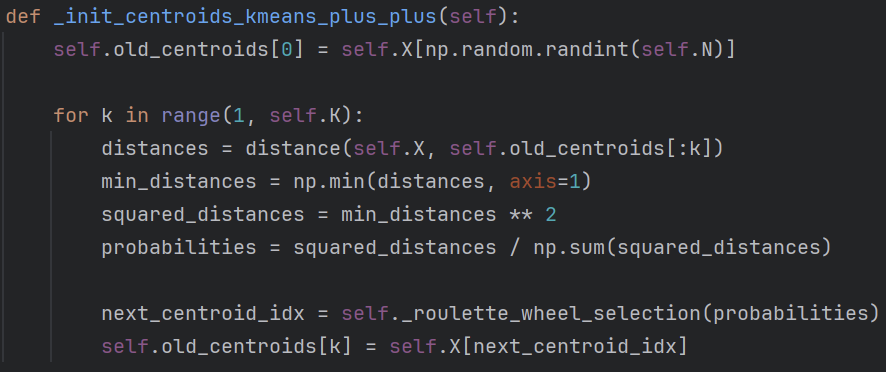
Uno de los principales problemas del K-means tradicional es su sensibilidad a la inicialización de los centroides. En nuestro caso el algoritmo requiere que los centroides iniciales se elijan de formas muy simples, lo que puede llevar a soluciones subóptimas y muy sensibles a la inicialización de dichos centroides. Además, en conjuntos de datos no uniformes o que representen clases muy diferentes en sus características, K-means estándar puede converger hacia óptimos locales en lugar de llegar a soluciones globales.

### K-means ++

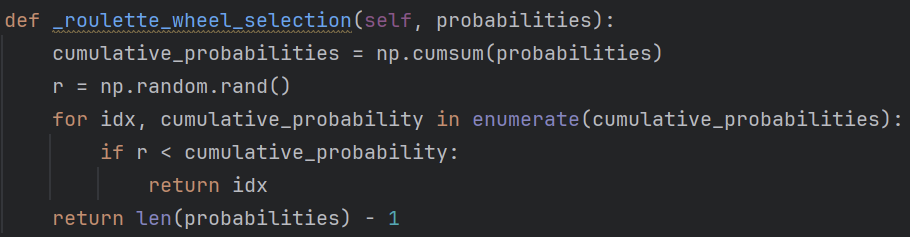
Por lo tanto, K-means++ es una mejora del algoritmo tradicional que aborda concretamente los problemas relacionados con la inicialización de los centroides. La clave del K-means++ es elegir los centroides de una manera más inteligente y efectiva para distribuirlos más uniformemente en el espacio de características.

La idea principal radica en seleccionar los centroides de acuerdo con un criterio basado en la distancia, para así inicializar los centros de inercia con más separación entre ellos. Para conseguir este resultado se pueden seguir estos pasos:

1. **Selección del primer centroide**: Se elige aleatoriamente un centroide.
2. **Encontrar candidatos para el próximo centroide**: Para cada punto de datos restante, se calcula la distancia al centroide más cercano seleccionado hasta el momento (distancia euclidiana por ejemplo). Después de calcular las distancias, se asigna a cada punto de datos una probabilidad de ser seleccionado como próximo centroide, donde la probabilidad de cada punto es proporcional al cuadrado de la distancia mínima encontrada en el paso anterior.
3. **Selección del siguiente centroide:** Se elige aleatoriamente según las probabilidades calculadas anteriormente. Así, los puntos más lejanos de los centroides ya seleccionados tendrán más probabilidad de ser seleccionados como el próximo centroide.
4. **Repetir hasta seleccionar todos los centroides**

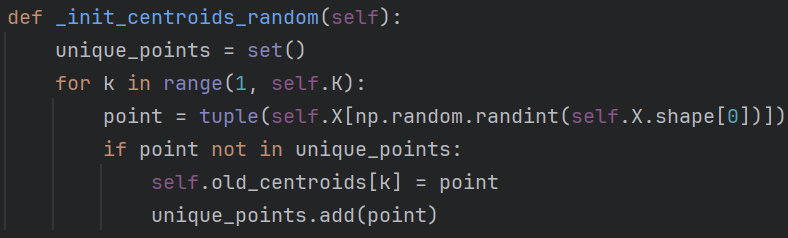
Este es el código para implementar el K-means++:

También vamos a necesitar esta función para elegir aleatoriamente el siguiente centroide:



### Inicialización Aleatoria

Otra manera más simple de inicializar los centroides es escoger puntos aleatorios, solo habrá que tener en cuenta no elegir puntos repetidos. Este es el código:



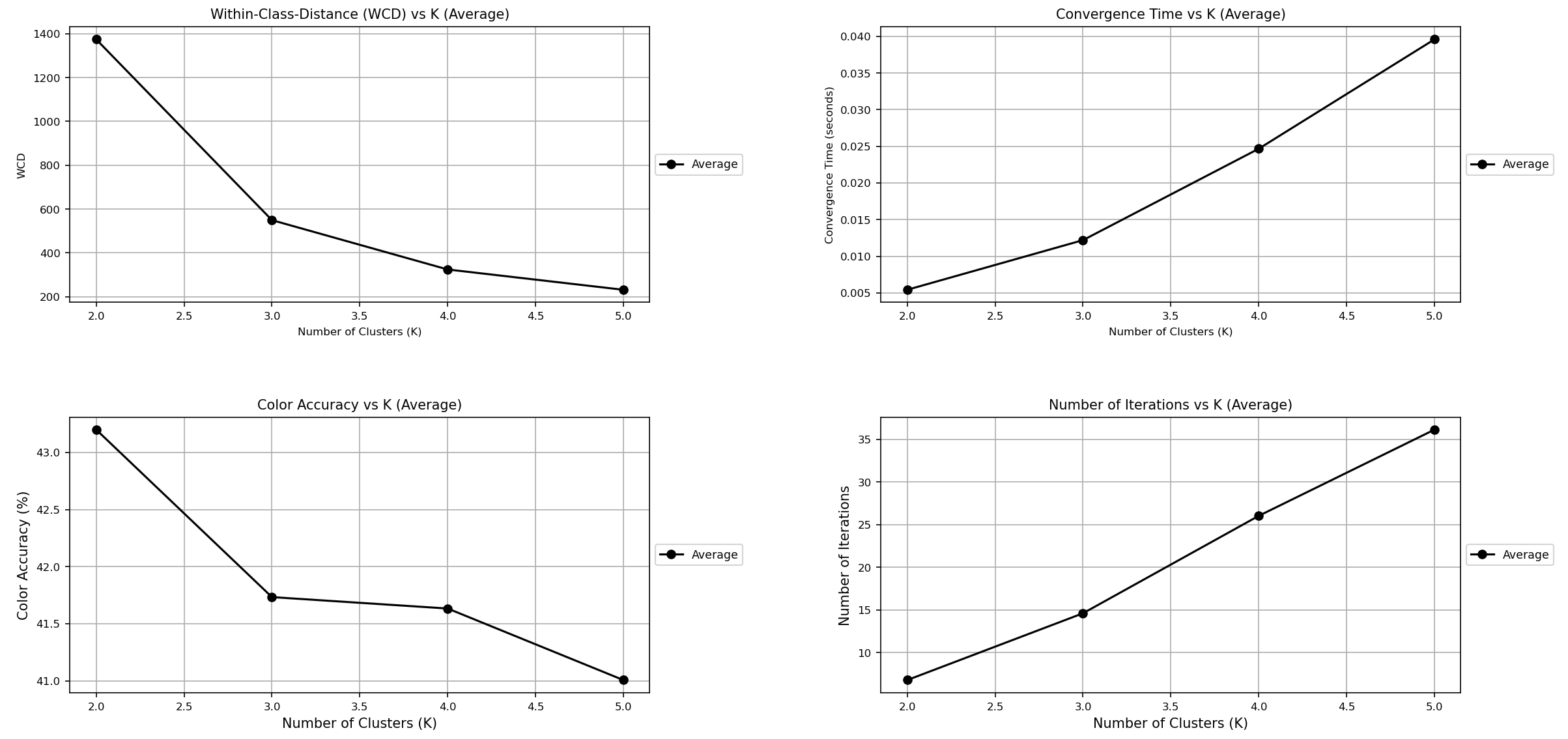
## Pruebas de rendimiento

A continuación vamos a realizar unas pruebas que van a utilizar los diferentes métodos de inicialización para el algoritmo K-means y diferentes sets de imágenes (imágenes sin modificar e imágenes recortadas) para poder evaluar y comparar el rendimiento de nuestro algoritmo con las mejoras.

### Test 1

En esta prueba vamos a comparar el rendimiento de los modos de inicialización first, random y k-means++ usando imágenes sin modificar y diferentes valores para la k. Vamos a utilizar 250 imágenes aleatorias del set de imágenes de test y vamos a ver varias medidas representadas en gráficas.

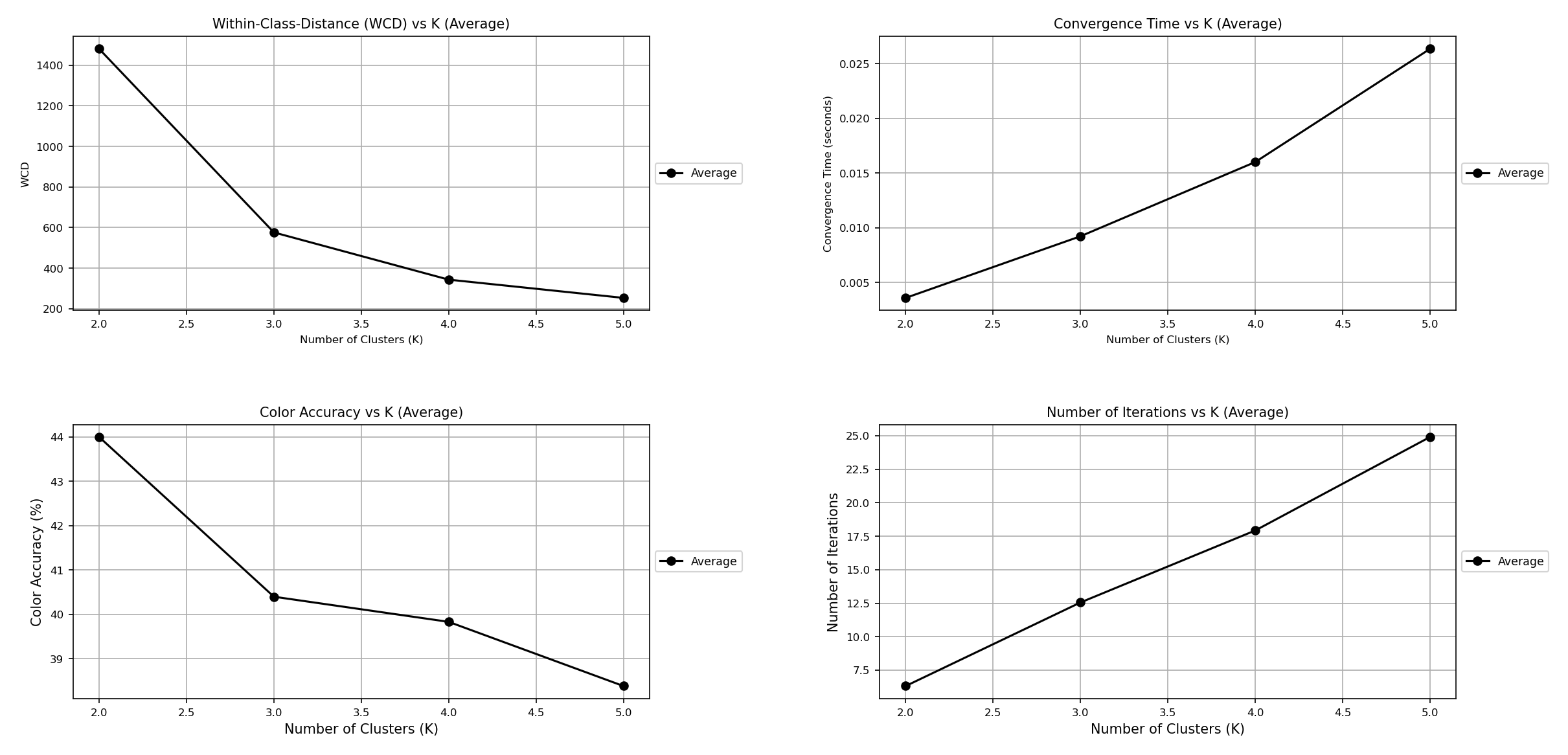
1. **Método First:**



Podemos observar como para todas las imágenes, a medida que la k aumenta, la media del WCD va disminuyendo y se establece en un rango de [200,400]. Por otro lado, el tiempo para converger y el número de iteraciones aumentan por cada valor de k adicional.

En cuanto a la tasa de exactitud de los colores predichos tenemos el máximo en k = 2, esto se puede deber a que las imágenes procesadas tienen 2 colores principales. Aumentar o disminuir la k en este caso provoca que el algoritmo K-means encuentre más o menos colores para dicha imagen, dando posibilidad a encontrar colores de más o no encontrar todos los colores correctos, afectando a la tasa de exactitud.

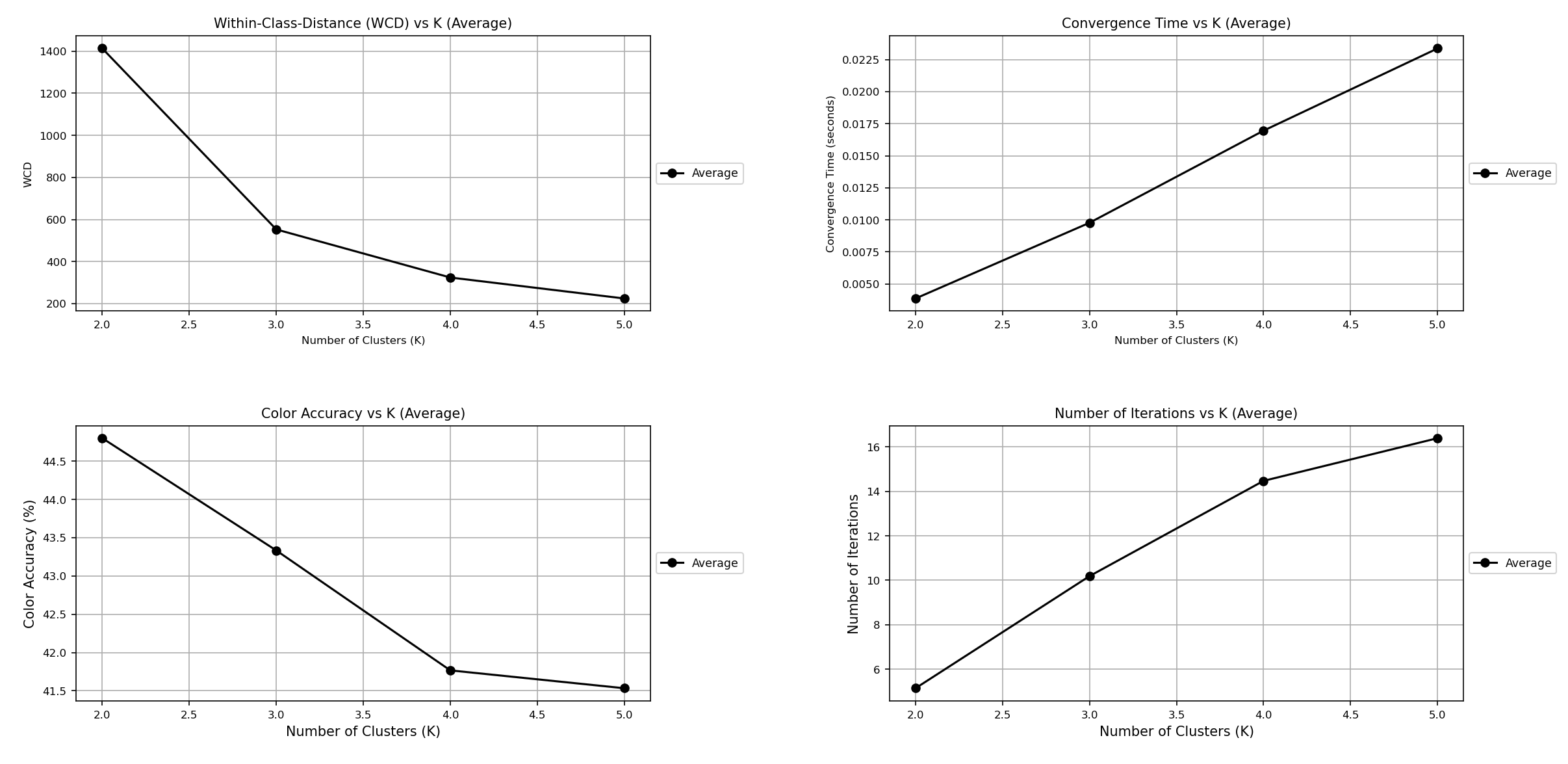
1. **Método Aleatorio:**



Podemos observar que usando la inicialización aleatoria la media de WCD y la tasa de exactitud es muy parecida a usar el método first. En cambio, para k = 5 el tiempo de convergencia ha disminuido 0.015 segundos y el número de iteraciones ha disminuido en 10 iteraciones.

Podemos observar que usando la inicialización aleatoria la media del número de iteraciones y el tiempo de convergencia disminuyen, pero la media de la tasa de exactitud es considerablemente peor. Esto puede ocurrir ya que los puntos se eligen de forma aleatoria o simplemente las imágenes aleatorias se han predicho peor en esta prueba.

1. **Método Kmeans++:**

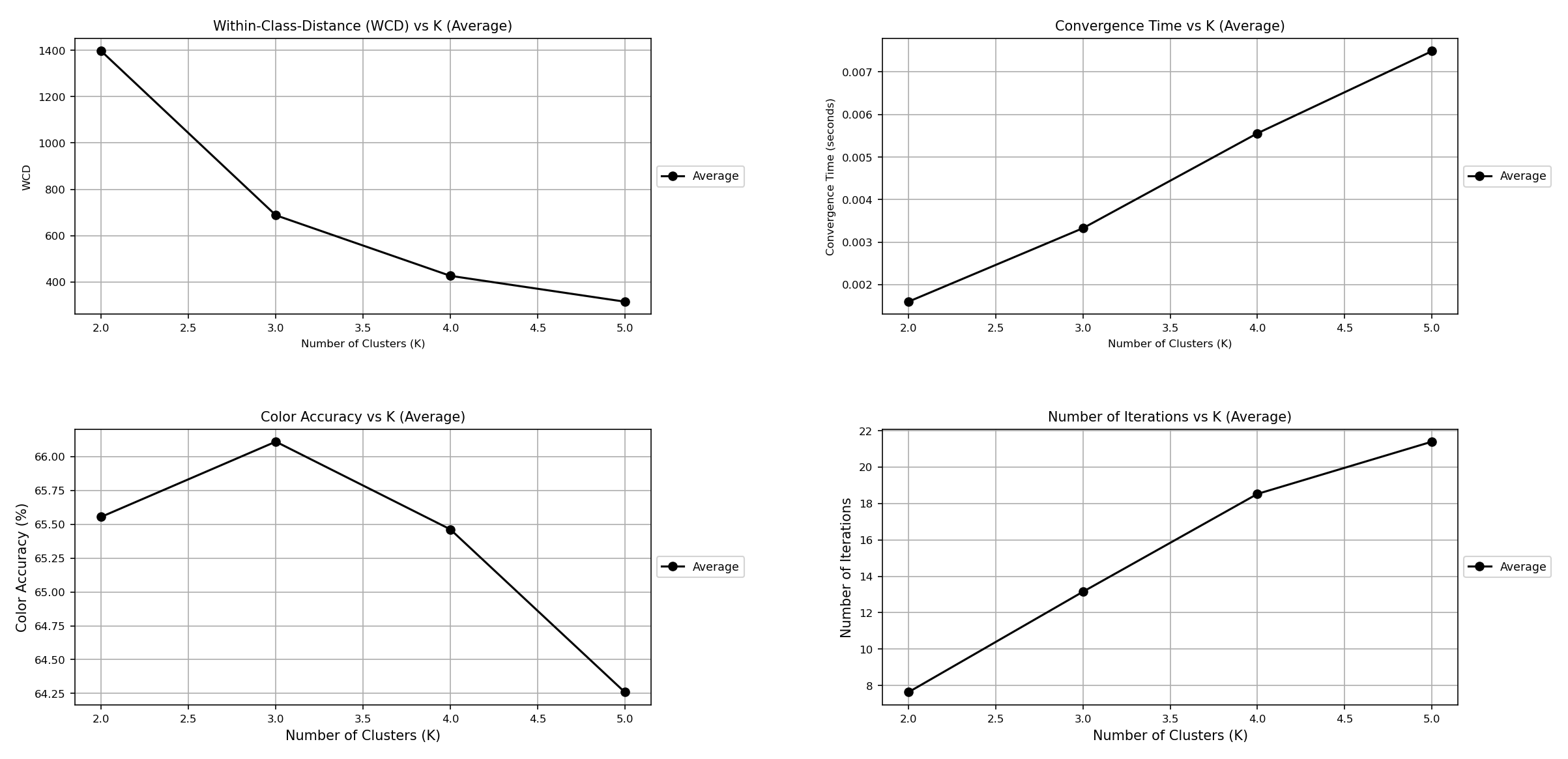


Para el Kmeans++ el WCD, el tiempo de convergencia y el número de iteraciones se mantiene muy parecido con el método aleatorio. Se puede notar una pequeña mejora en la tasa de exactitud respecto al método aleatorio, pero no es muy significativo.

### Test 2

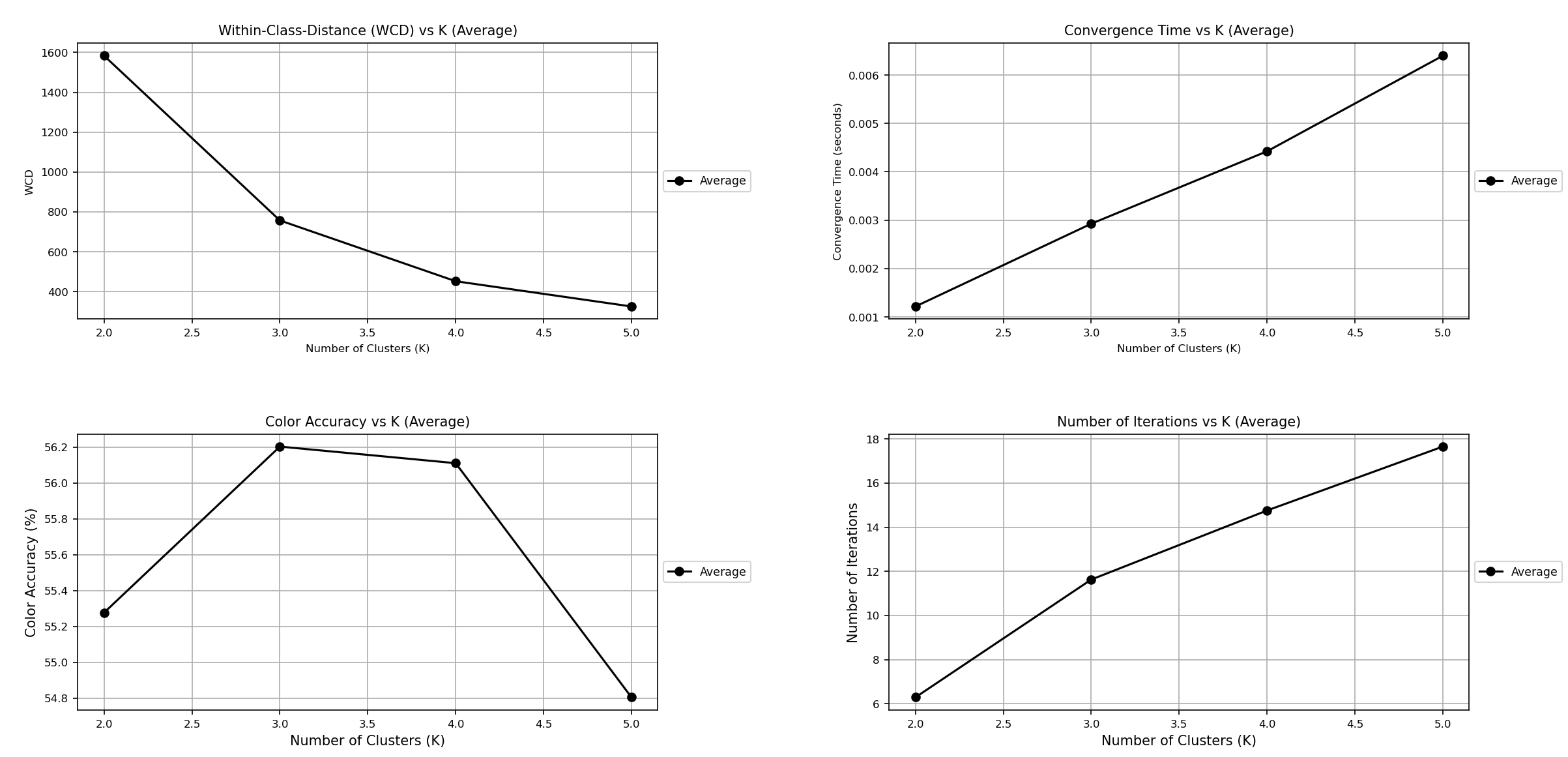
Para esta prueba vamos a hacer lo mismo que la prueba anterior pero ahora utilizando las imágenes recortadas para ver si supone una mejora más significativa. Solo tenemos 180 imágenes recortadas así que las pruebas se harán con 180 imágenes en vez de 250 como en el test anterior.

1. **Método First:**



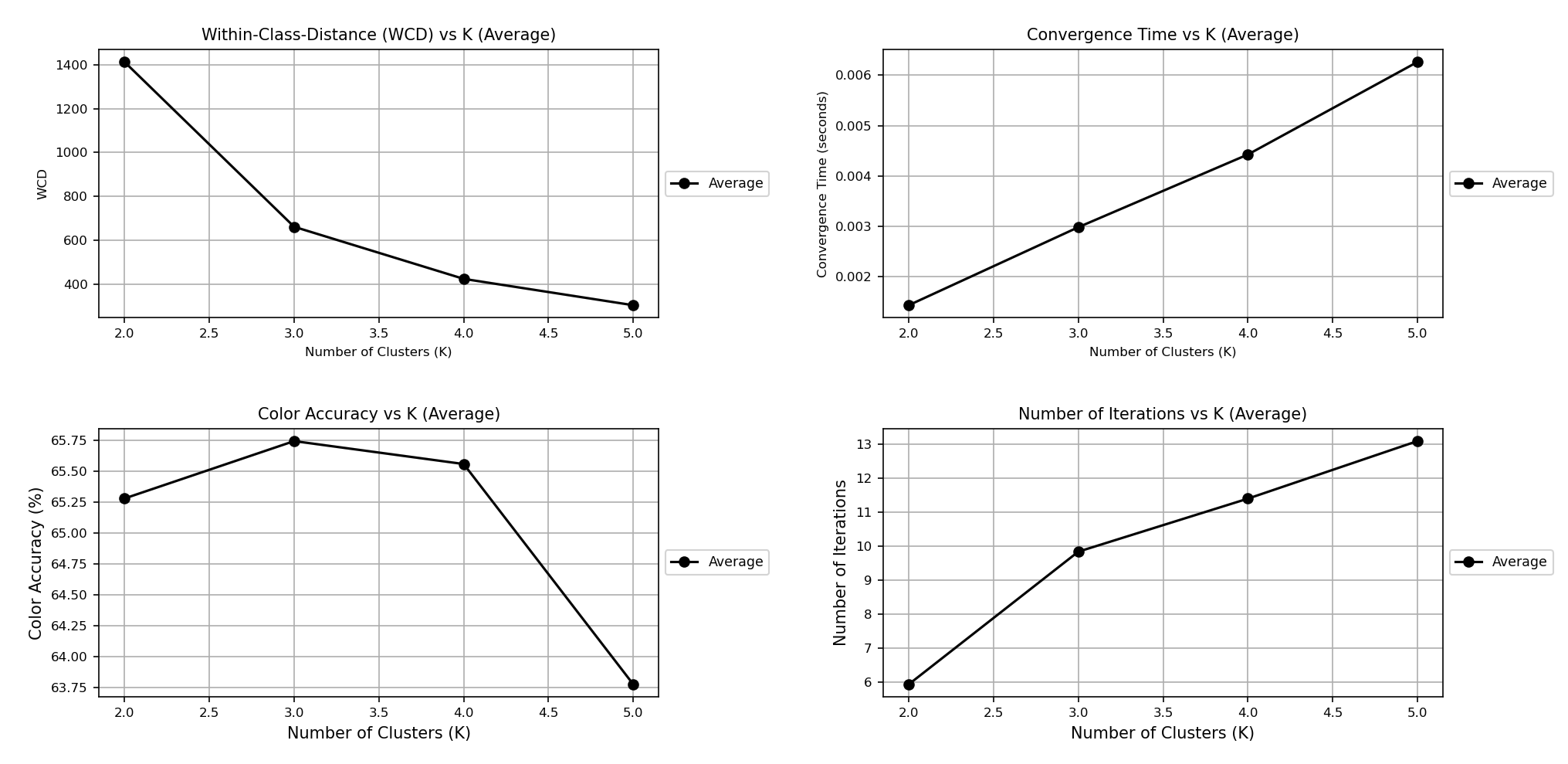
Aqui se observa que usando las imágenes recortadas la tasa de exactitud, el tiempo de convergencia y el número de iteraciones mejora considerablemente comparado con el uso de las imágenes sin modificar.

1. **Método Aleatorio:**



Al igual que en el método anterior, los resultados son mejores con las imágenes recortadas. Comparado con el método First, en este caso podemos ver que el tiempo de convergencia y el número de iteraciones es levemente mejor. En la exactitud de color sí que hay una disminución de rendimiento importante, pasando de un rango de exactitud de [64,66] a uno de [54,56].

1. **Método Kmeans++:**



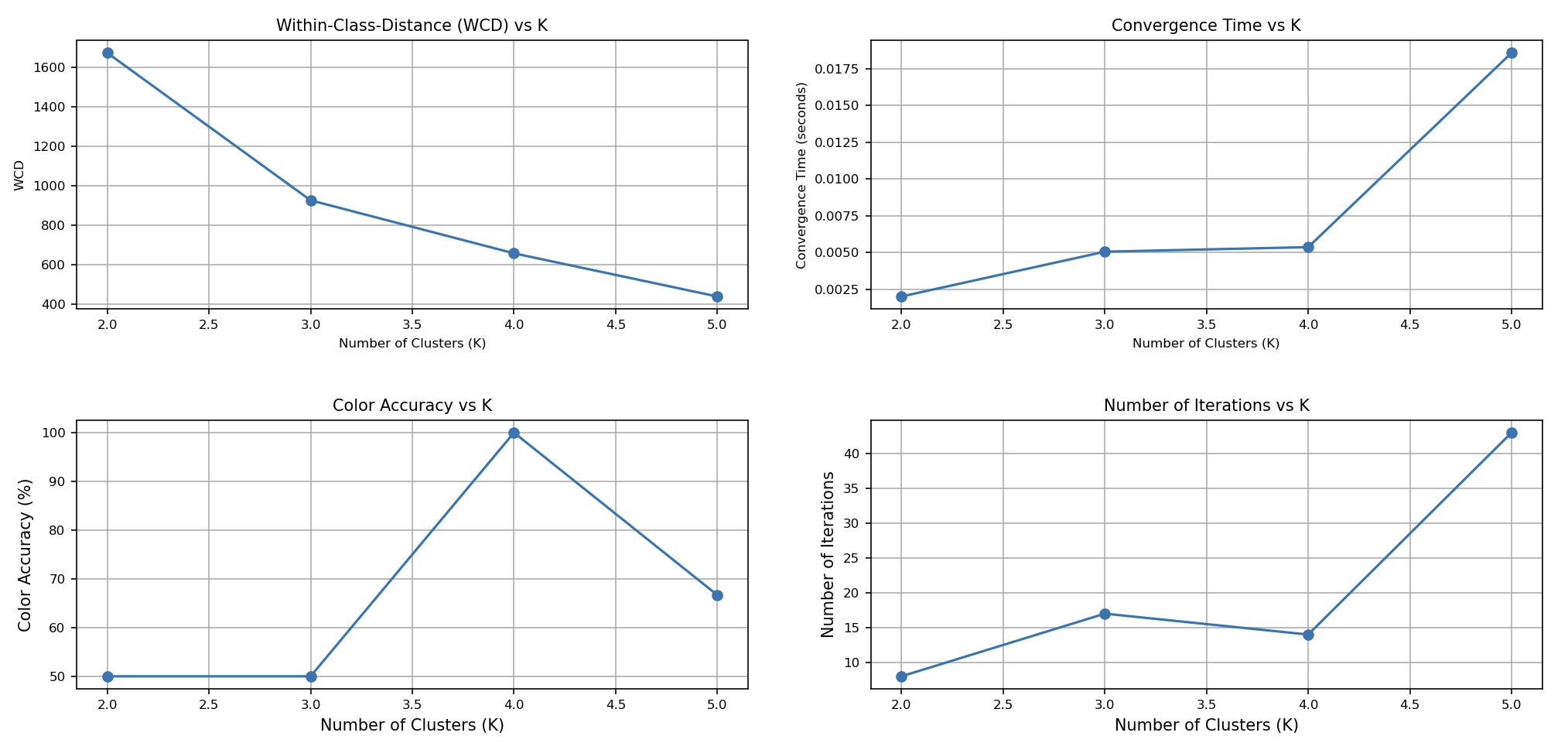
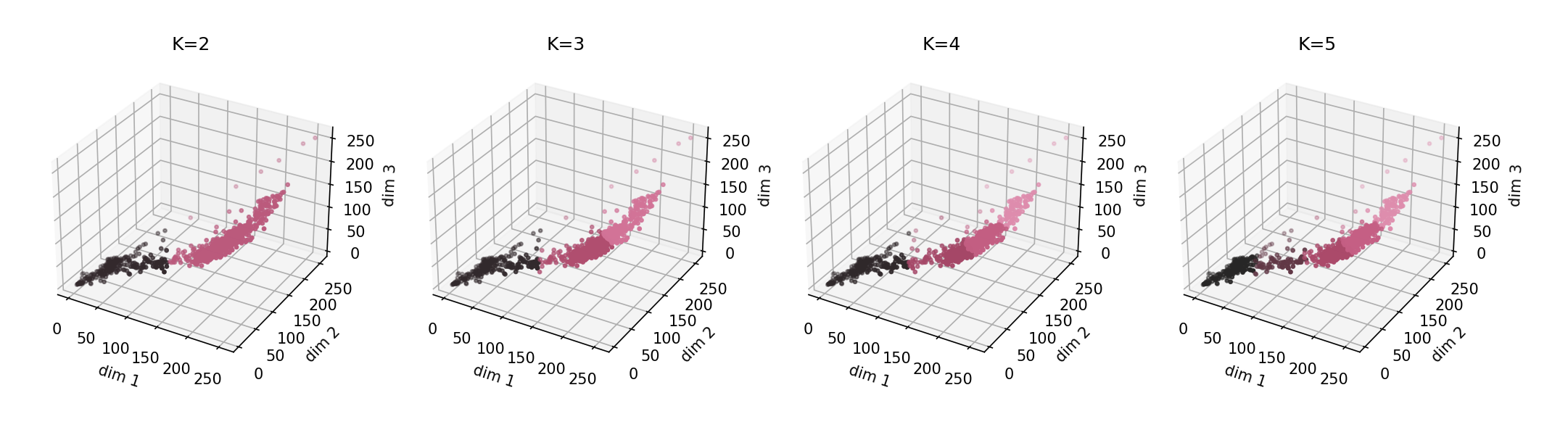
Para el caso de Kmeans++ con imágenes recortadas también se ve una mejora considerable comparada con usar imágenes sin modificar. Pero comparada con los métodos anteriores usando imágenes recortadas podemos ver una muy pequeña mejora en la tasa de exactitud, tiempo de convergencia y número de iteraciones, pero sigue sin ser significativo.

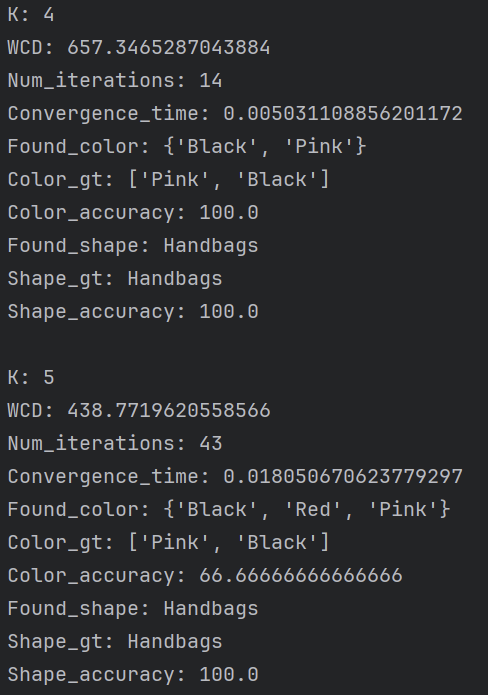
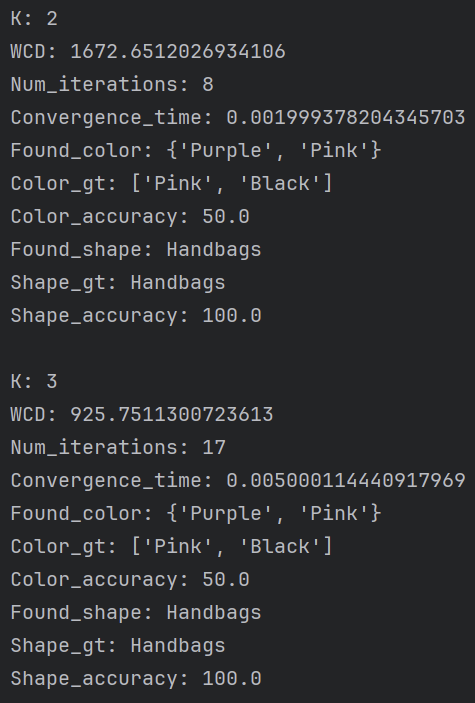
### Test 3

Para esta prueba vamos a seguir la misma filosofía de aplicar los tres métodos de inicialización que tenemos, pero esta vez lo haremos sobre solo una imagen recortada para ver si a pequeña escala podemos detectar algún cambio que sea significativo. Esta va a ser la imágen que vamos a evaluar:

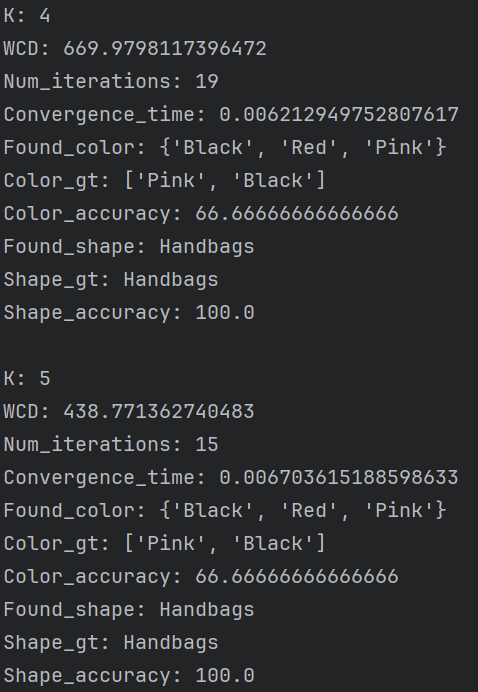
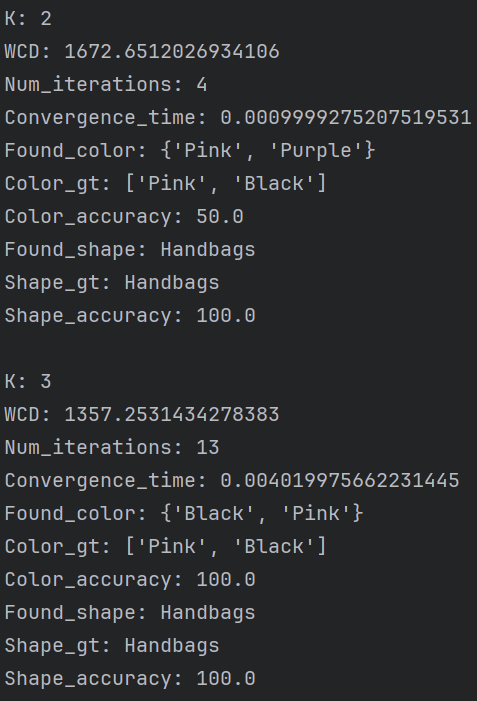
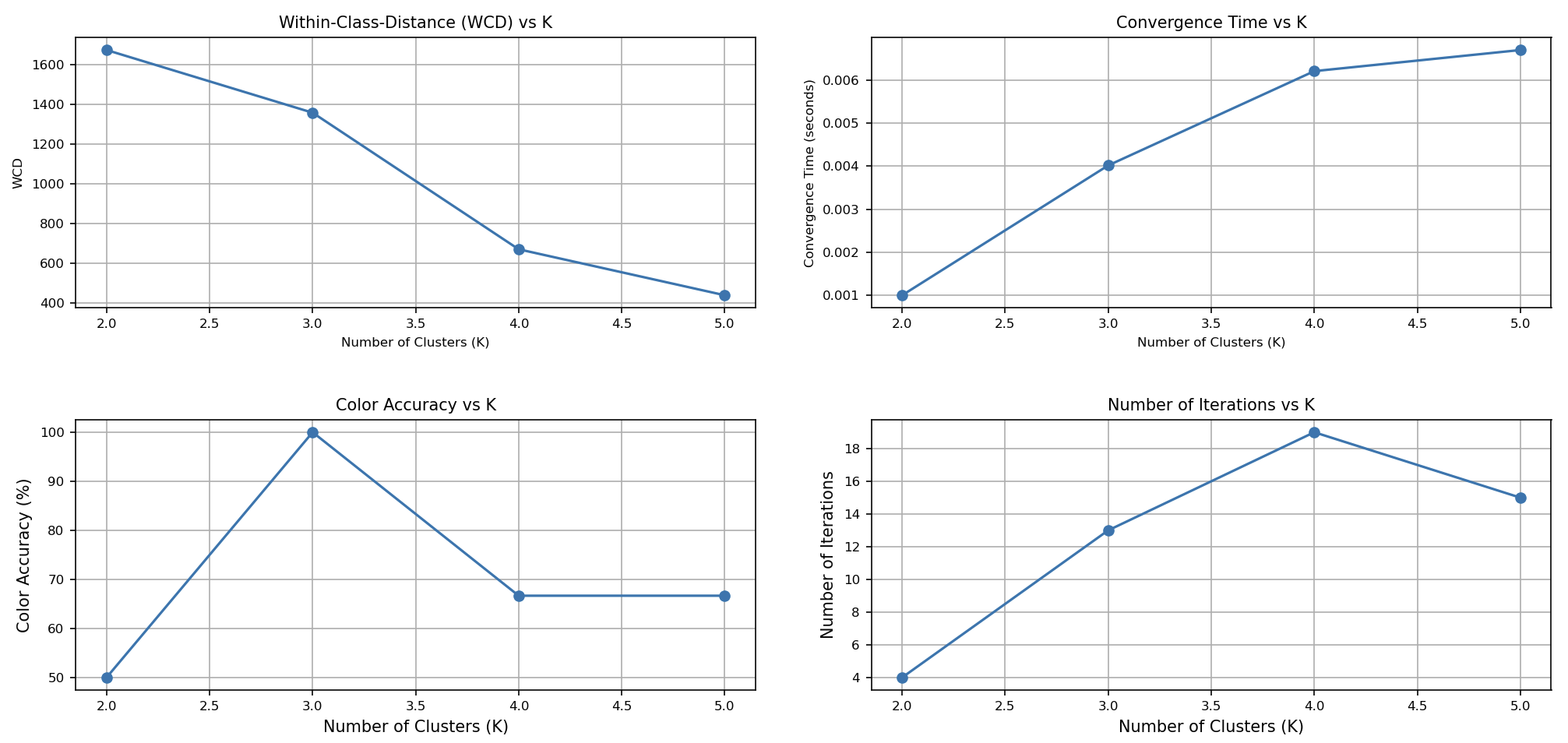
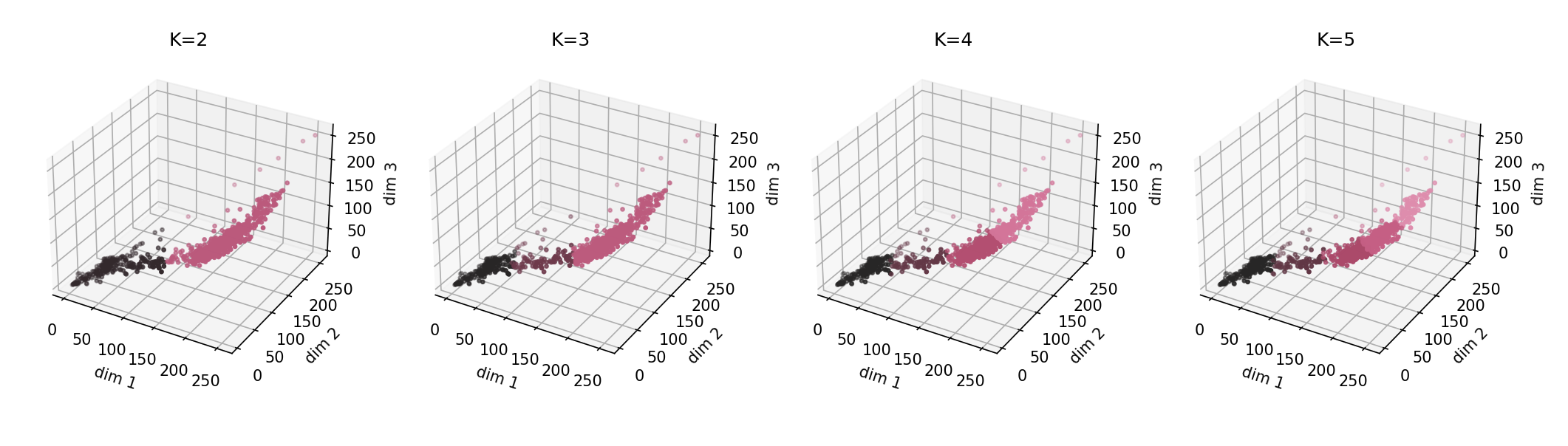


1. Método First:

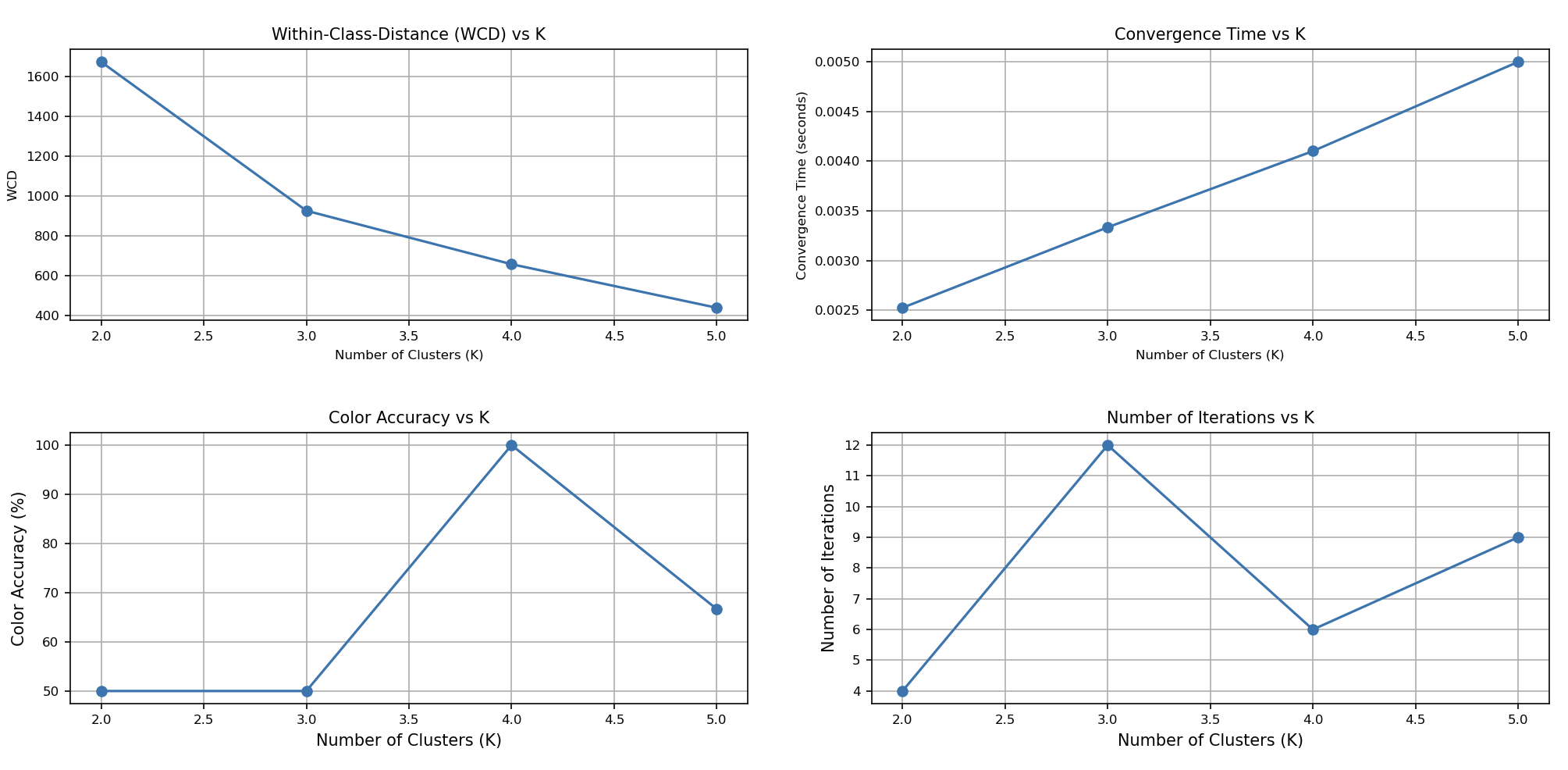
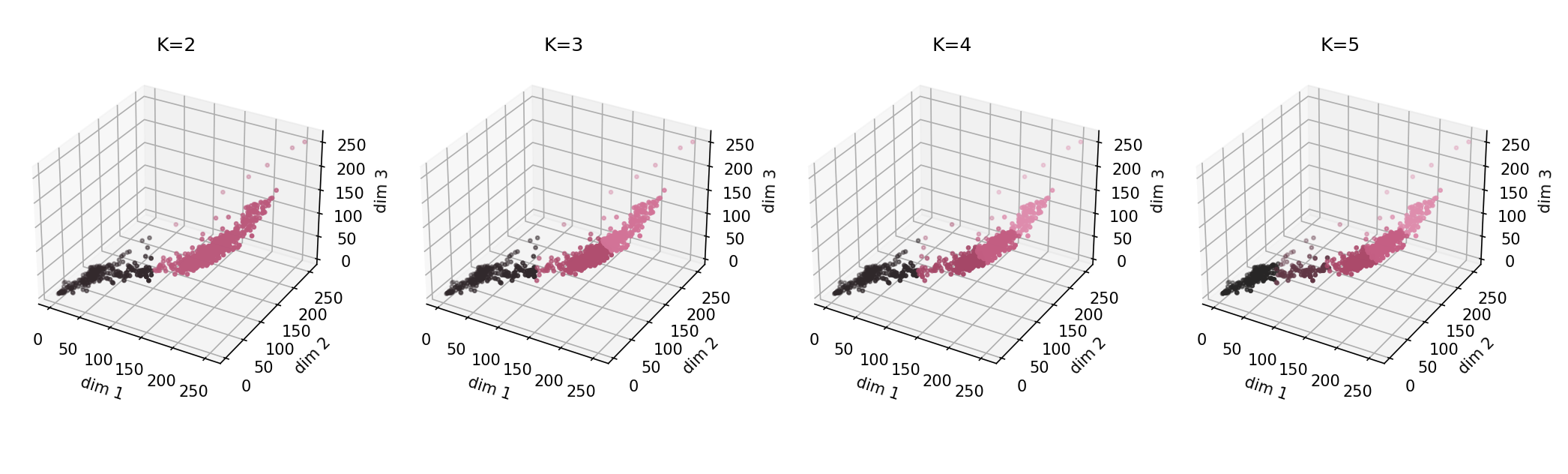


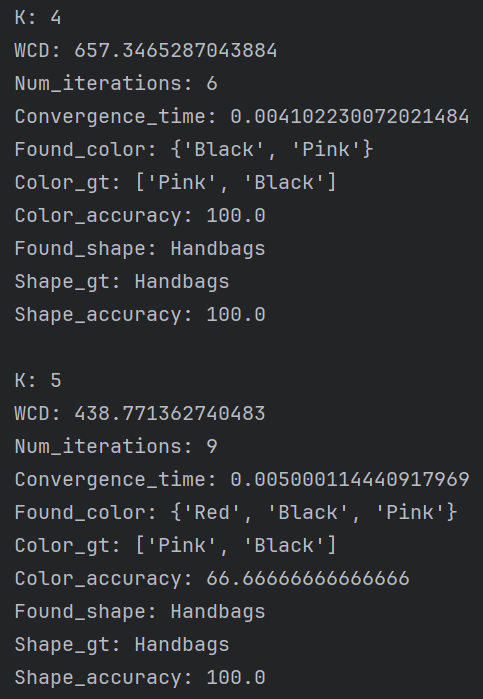
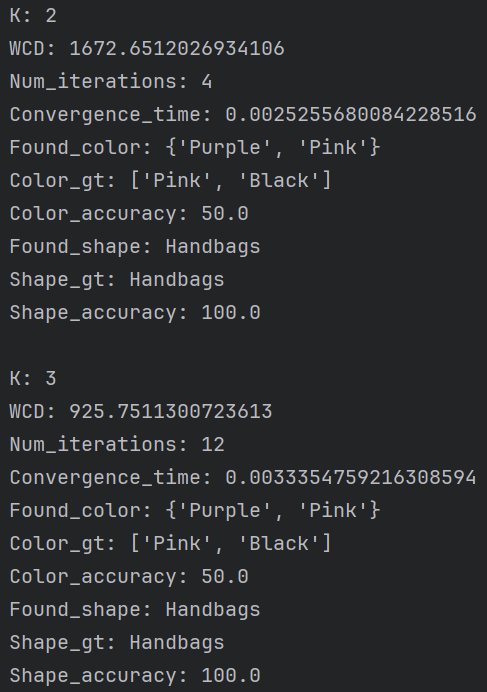


1. Método Aleatorio:



1. Método Kmeans++:





Viendo los resultados sobre esta imagen podemos concluir que el WCD y la tasa de exactitud son casi idénticas en todos los métodos de inicialización. Es en las otras dos medidas donde vemos mejoras más grandes. Para el método First tenemos 43 iteraciones en k = 5, con el método Aleatorio tenemos 15 iteraciones y con el Kmeans++ 9 iteraciones. La mejora en tiempo de convergencia aumenta de igual manera, ya que depende directamente del número de iteraciones.

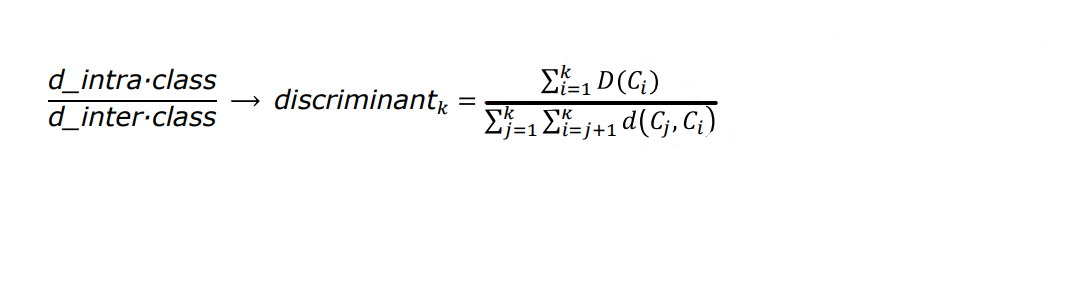
## Conclusión

El método Kmeans++ elije los centroides de manera más inteligente que los otros dos métodos, resultando en tiempos de convergencia y tasas de exactitud mucho mejores, tal como hemos visto en el Test 3. En los tests 1 y 2 la media de las mejoras sobres éstas medidas es más leve, aunque si fuésemos a realizar los mismos tests sobre un conjunto de imágenes mucho más masivo sí veríamos un cambio considerable.

No obstante, hay que tener en cuenta que gran parte de las mejoras se deben a utilizar imágenes recortadas, ya que al eliminar el color del fondo o eliminar al modelo y dejar solamente la prenda en la imagen, el K-means puede clasificar mejor los colores. Aun así, está claro que el Kmeans++ es la mejor opción de inicialización que podemos elegir para nuestro problema de clasificación de prendas.

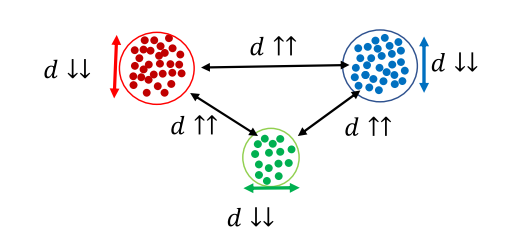
## Coeficiente de Fisher

La idea es mejorar la clasificación del K-means aplicando el coeficiente de Fisher, el cual utiliza tanto el Intra-class y el Inter-class, de cara a programar solo tendríamos que diseñar la Inter-class, ya que en la entrega anterior ya teníamos diseñada la Intra-class, y el coeficiente de Fisher que equivale a la siguiente fórmula que aplica las dos anteriores:

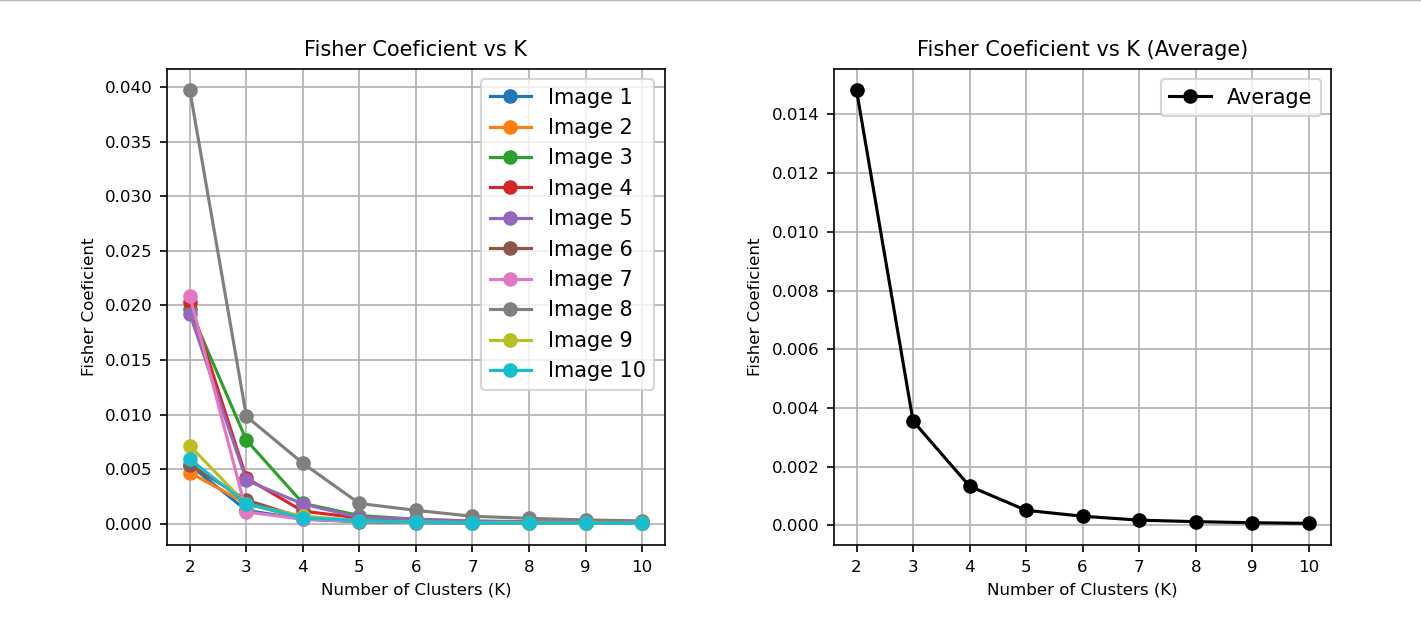


Esto mejorará la clasificación, ya que proporciona una mejor relación tanto dentro de la clase como fuera de ella, de ahí que utilice ambos discriminantes Intra e Inter.

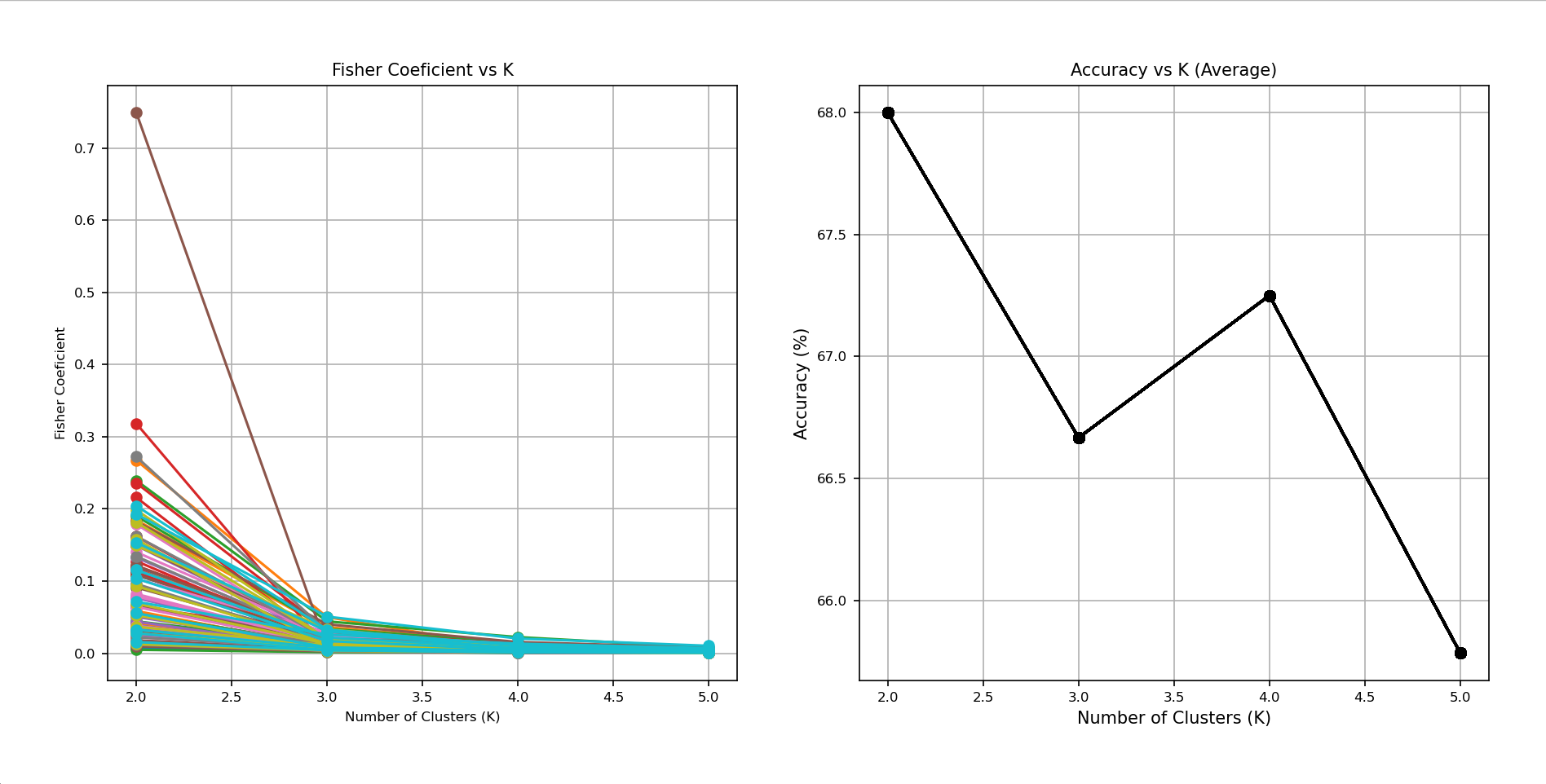
El objetivo es que el discriminante sea lo más pequeño posible, lo que indicaría que la distancia entre clases diferentes sea lo más alta posible, en cambio, la distancia dentro de las mismas clases sea la más baja posible, esto indicaría una buena distribución, como en el ejemplo siguiente, proporcionado del PDF expuesto en clase.



Resultados para 10 imágenes probando con hasta 10 K:

Esto nos indica que contra más K el coeficiente de Fisher será más pequeño lo que indica mejor clasificación a la hora de crear las diferentes clases, este ejemplo se basa en 10 imágenes, pero en el siguiente que trata de 100 imágenes también lo podemos observar.

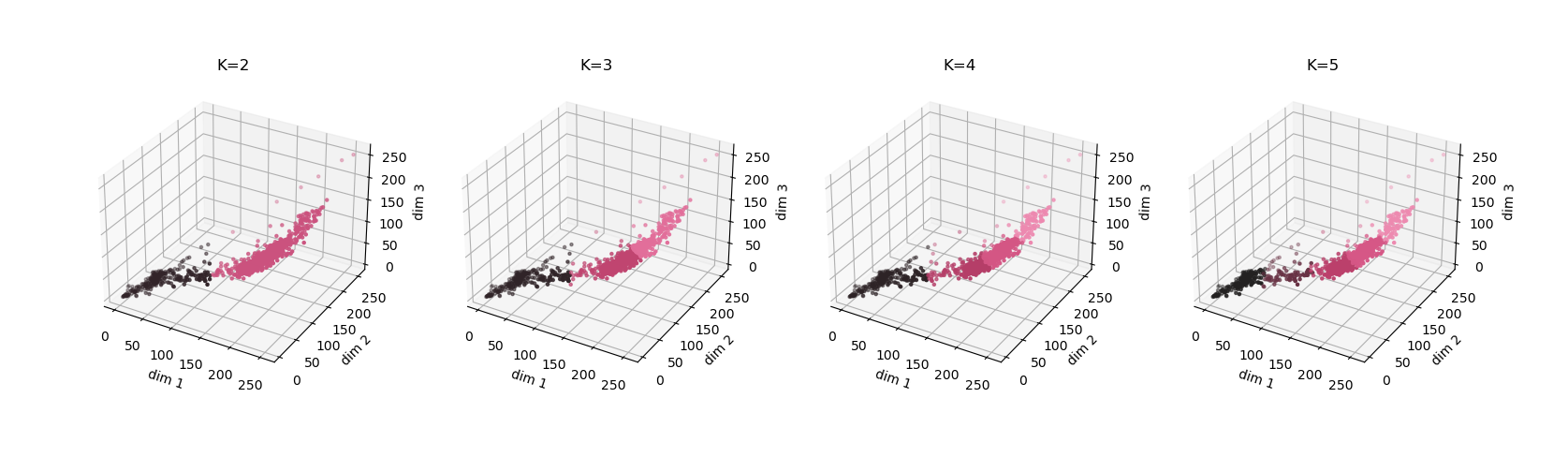
Resultados para 100 imágenes recortadas comparado con la media de acierto:

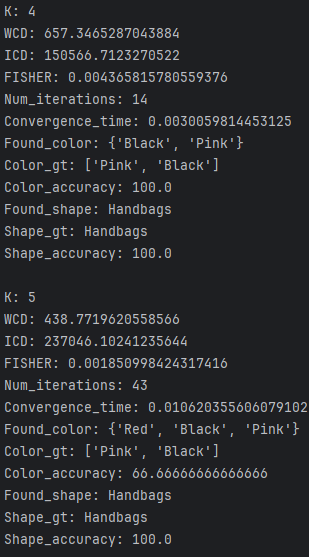
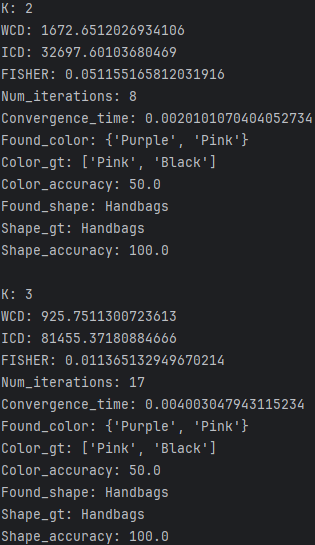


En las gráficas anteriores observamos la relación entre el acierto con la K y el coeficiente de Fisher, podemos ver que aunque el coeficiente Fisher mejora, el acierto es un poco relativo, esto también depende muy a gran parte con el ground\_truth, por eso podemos ver pico en K=2 y K=4, muy probablemente si ejecutásemos esto con hasta K=10 o más el acierto sería peor, pero realmente la clasificación de la imagen sería mejor, en este caso podríamos considerar que una K adecuada sería K=4.

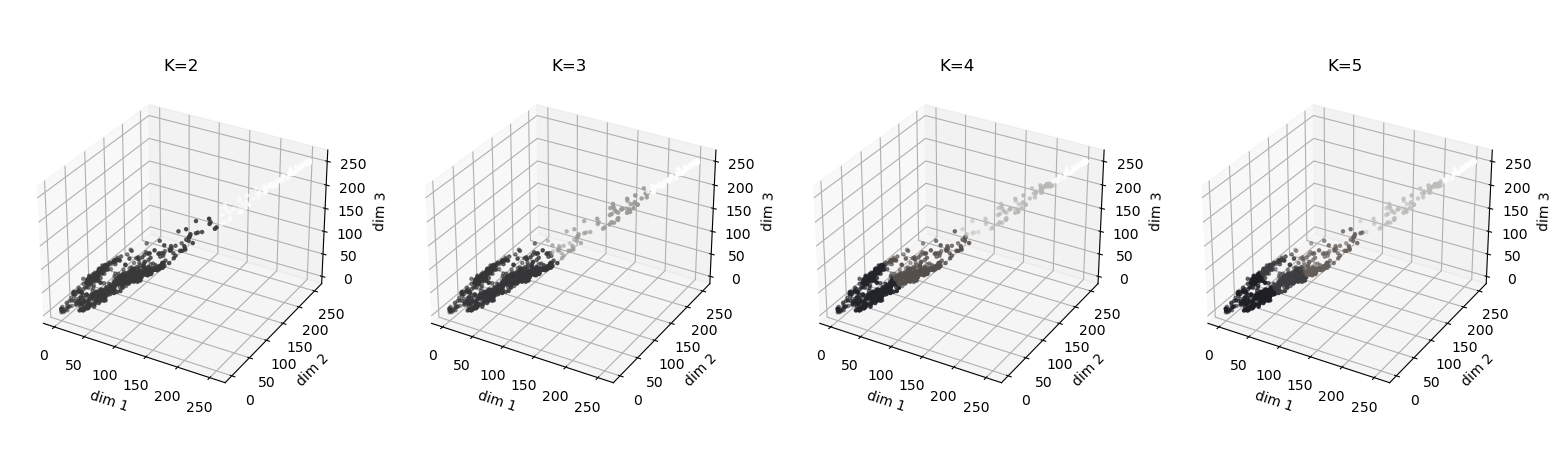
Visualización de 5 imagenes con su respectivo k y estadísticas (estas pruebas se ejecutan sobre las cropped image, para no obtener valores redundantes como el color blanco o el color de piel, etc):

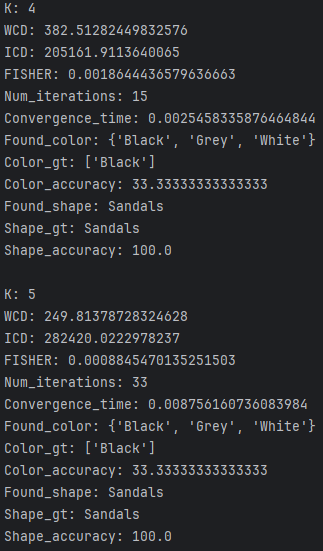
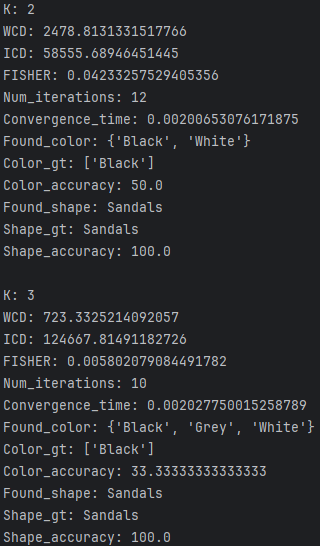
IMG1:



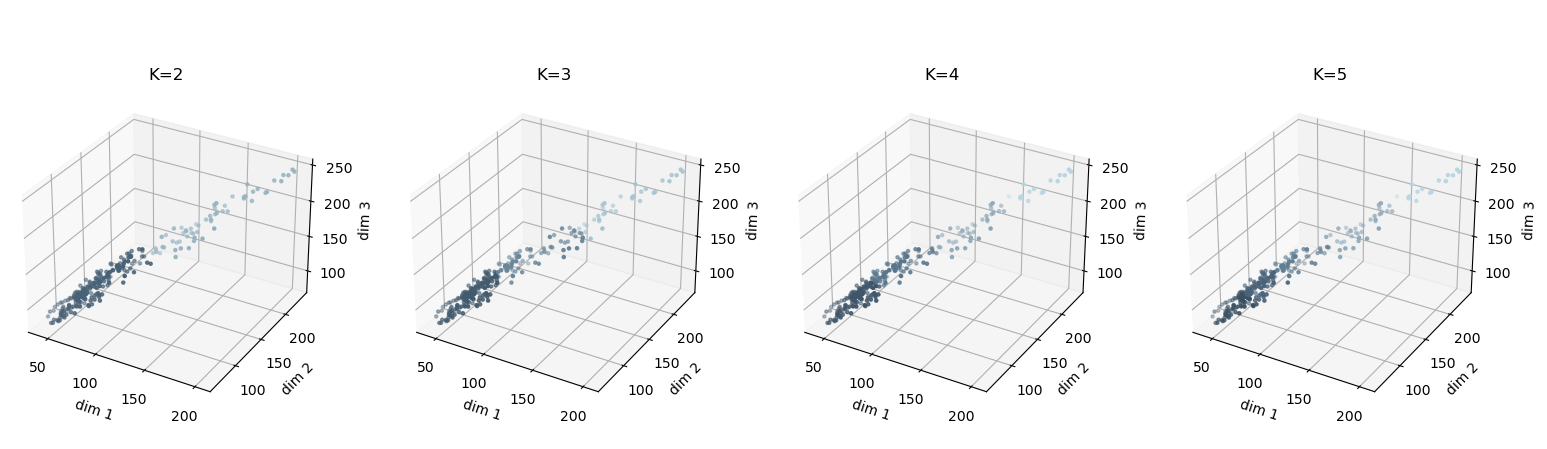


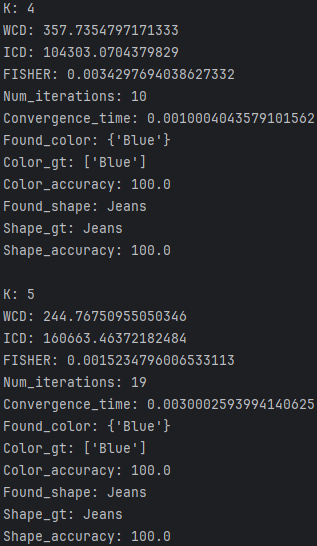
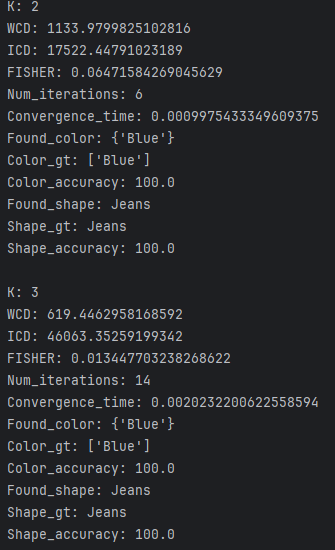
IMG2:



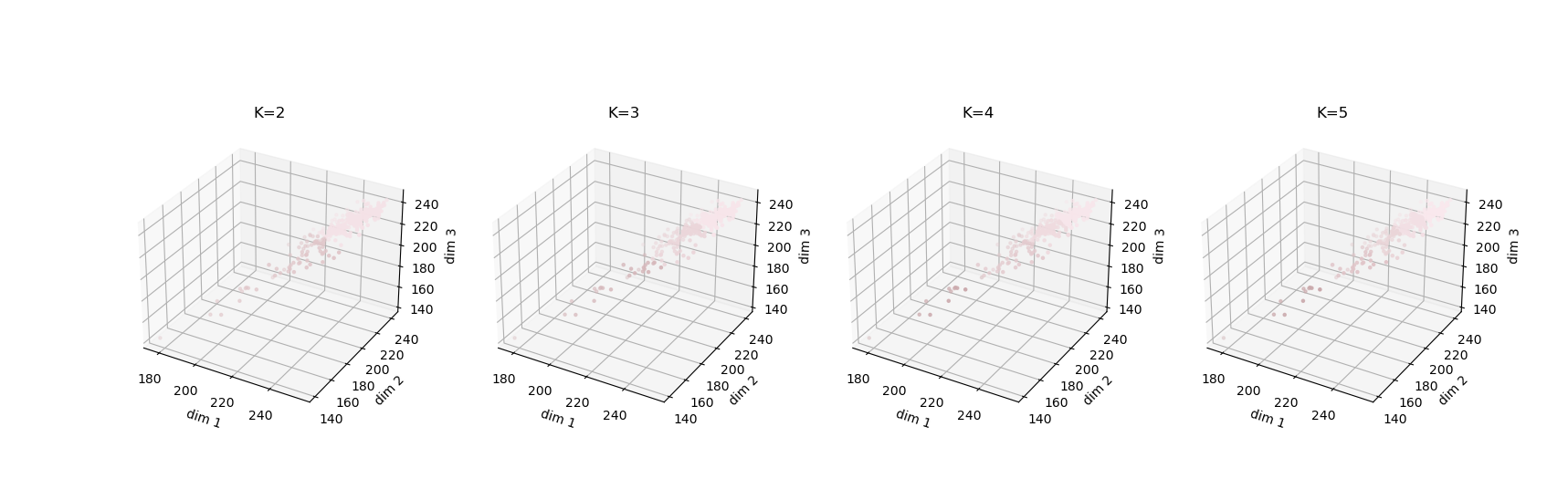


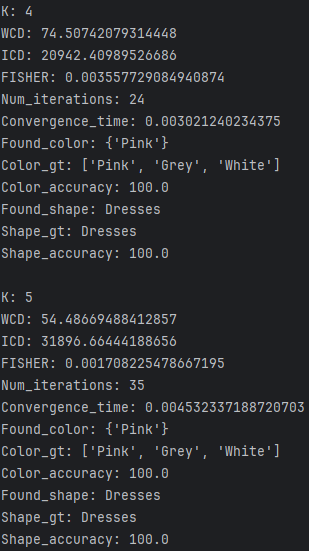
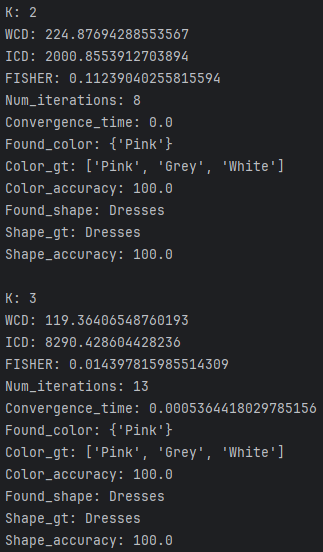
IMG3:



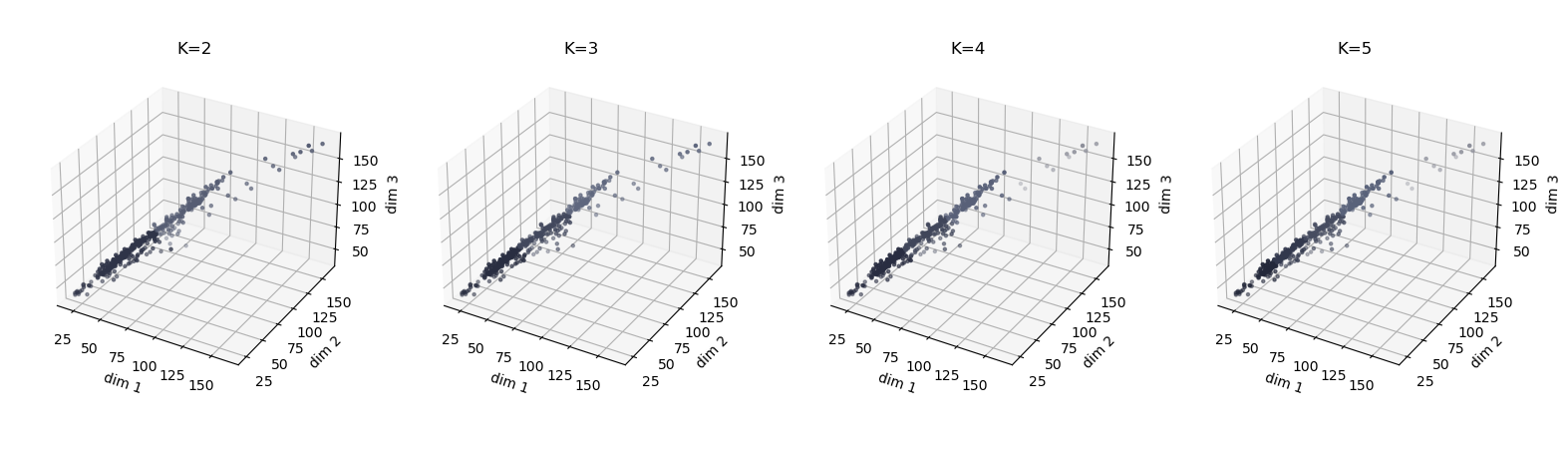


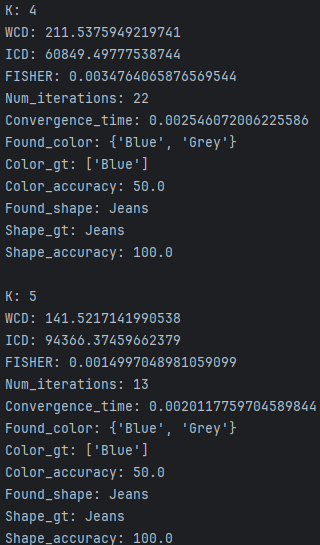
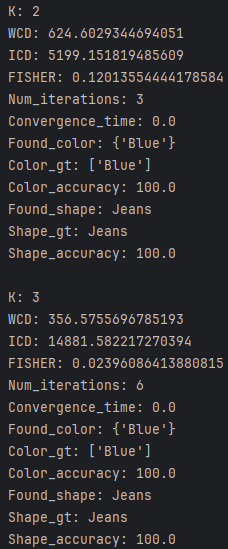
IMG4:





IMG5:





Después de las pruebas anteriores, vemos cómo con el incremento de K mejoramos el coeficiente de Fisher, vemos cómo el WCD disminuye y el ICD aumenta, esto es positivo respecto a lo que queremos. Si el ground truth fuese más complejo nuestro resultado sería mucho mejor contra mejor Fisher tengamos, en este caso tendríamos que mirar que K nos sería más útil, podríamos decir que K=4 o K=2 serían muy buenas opciones en el escenario actual.

## Find\_BestK:

En la primera parte utilizamos la siguiente formula con un límite del 20% para encontrar la mejor K:

Imagen que contiene Texto

Descripción generada automáticamente

En esta mejora buscaremos fórmulas alternativas o probar mejores límites para intentar conseguir una mejor K.

Para realizar los tests hemos decidido incrementar y decrementar el umbral del threshold de 5 en 5 para poder ajustarlo de forma más precisa. Partimos de un threshold de 20 e iremos probando con 3 imágenes, normales y recortadas. Nuestro max\_K sería de 10.

Originalmente pensamos que esto sería suficiente, pero con las 3 imagenes no cambio nada del resultado. Así que empezamos con pruebas mucho más grandes.

Para dejar una base de lo que tenemos sin modificar estos son los mejores resultados obtenidos con y sin las mejoras de random /kmeans++

\*Por mejores resultados consideramos los valores más altos en todos los aspectos, priorizando el color accuracy y el WCD \*:

Threshold 20% cropped:

|  |  |  |
| --- | --- | --- |
| Img\_01 | Img\_02 | Img\_03 |
| **Normal:**  K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.005997419357299805  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con kmeans++:**  K: 4  WCD: 657.3465287043884  Num\_iterations: 6  Convergence\_time: 0.004000186920166016  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con random:**  K: 3  WCD: 926.1602988679557  Num\_iterations: 16  Convergence\_time: 0.00599980354309082  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | **Normal:**  K: 10  WCD: 115.9004695955256  Num\_iterations: 25  Convergence\_time: 0.03552412986755371  Found\_color: {'Grey', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 33.33333333333333  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con kmeans++:**  K: 2  WCD: 2479.0276868194464  Num\_iterations: 6  Convergence\_time: 0.002002239227294922  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con random:**  K: 2  WCD: 2479.2066301140903  Num\_iterations: 10  Convergence\_time: 0.003999948501586914  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | **Normal:**  K: 9  WCD: 122.56903763401371  Num\_iterations: 15  Convergence\_time: 0.009000539779663086  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con kmeans++:**  K: 10  WCD: 86.19469524882537  Num\_iterations: 17  Convergence\_time: 0.016002655029296875  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con random:**  K: 8  WCD: 140.10997371445177  Num\_iterations: 41  Convergence\_time: 0.018003463745117188  Found\_color: {'Blue', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Threshold 20% un-cropped:

|  |  |  |
| --- | --- | --- |
| Img\_01 | Img\_02 | Img\_03 |
| **Normal:**  K: 2  WCD: 678.388323581342  Num\_iterations: 3  Convergence\_time: 0.003000497817993164  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con kmeans++:**  K: 2  WCD: 678.388323581342  Num\_iterations: 3  Convergence\_time: 0.003000497817993164  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con random:**  K: 2  WCD: 678.388323581342  Num\_iterations: 5  Convergence\_time: 0.0037398338317871094  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | **Normal:**  K: 2  WCD: 545.0106497914171  Num\_iterations: 4  Convergence\_time: 0.009015083312988281  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con Kmeans++:**  K: 2  WCD: 545.0106497914171  Num\_iterations: 3  Convergence\_time: 0.0040018558502197266  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con random:**  K: 2  WCD: 545.0106497914171  Num\_iterations: 4  Convergence\_time: 0.003995656967163086  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | **Normal:**  K: 4  WCD: 609.4479810053048  Num\_iterations: 25  Convergence\_time: 0.0480191707611084  Found\_color: {'Red', 'Orange', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con Kmeans++:**  K: 4  WCD: 609.4479810053048  Num\_iterations: 7  Convergence\_time: 0.013017416000366211  Found\_color: {'Orange', 'Red', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0  **Con Random:**  K: 3  WCD: 1154.8341365204988  Num\_iterations: 12  Convergence\_time: 0.015085697174072266  Found\_color: {'Red', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 33.33333333333333  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Para no complicar demasiado los resultados y ejecutar 10 iteraciones por cada opción kmeans++ y random haremos los test sin usar estas mejoras implementadas.

### Test 1

Threshold 15% cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.004997730255126953  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 2479.2066301140903  Num\_iterations: 9  Convergence\_time: 0.003998279571533203  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 117.52717960233952  Num\_iterations: 15  Convergence\_time: 0.008000373840332031  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Threshold 15% un-cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 2  WCD: 678.388323581342  Num\_iterations: 5  Convergence\_time: 0.013007402420043945  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 545.0106497914171  Num\_iterations: 4  Convergence\_time: 0.009999513626098633  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 4  WCD: 609.4479810053048  Num\_iterations: 25  Convergence\_time: 0.02802133560180664  Found\_color: {'Red', 'Orange', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

### Test 2

Threshold 10% cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.009003400802612305  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 2479.2066301140903  Num\_iterations: 9  Convergence\_time: 0.002996683120727539  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 117.52717960233952  Num\_iterations: 15  Convergence\_time: 0.008049249649047852  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Threshold 10% un-cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 2  WCD: 678.388323581342  Num\_iterations: 5  Convergence\_time: 0.008966207504272461  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 545.0106497914171  Num\_iterations: 4  Convergence\_time: 0.008865833282470703  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 4  WCD: 609.4479810053048  Num\_iterations: 25  Convergence\_time: 0.03654313087463379  Found\_color: {'Red', 'Orange', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

### Test 3

Threshold 25% cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.009999513626098633  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 2479.2066301140903  Num\_iterations: 9  Convergence\_time: 0.0030002593994140625  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 117.52717960233952  Num\_iterations: 15  Convergence\_time: 0.013999223709106445  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Threshold 25% un-cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 2  WCD: 678.388323581342  Num\_iterations: 5  Convergence\_time: 0.009000301361083984  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 545.0106497914171  Num\_iterations: 4  Convergence\_time: 0.00899505615234375  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 4  WCD: 609.4479810053048  Num\_iterations: 25  Convergence\_time: 0.027010679244995117  Found\_color: {'Red', 'Orange', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

### Test 4

Threshold 30% cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.00699615478515625  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 2479.2066301140903  Num\_iterations: 9  Convergence\_time: 0.004004001617431641  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 117.52717960233952  Num\_iterations: 15  Convergence\_time: 0.007998943328857422  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Threshold 30% un-cropped:

|  |  |  |
| --- | --- | --- |
| Img1 | Img2 | Img3 |
| K: 2  WCD: 678.388323581342  Num\_iterations: 5  Convergence\_time: 0.025196313858032227  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 545.0106497914171  Num\_iterations: 4  Convergence\_time: 0.008005142211914062  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 4  WCD: 609.4479810053048  Num\_iterations: 25  Convergence\_time: 0.026999473571777344  Found\_color: {'Red', 'Orange', 'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

### Test 5

Para la prueba 5 hemos decidido realizar los tests previos únicamente cropped con una mayor cantidad de imágenes

20:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Img1 | Img2 | Img3 | Img4 | Img5 |
| K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.007711172103881836  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 2479.2066301140903  Num\_iterations: 9  Convergence\_time: 0.0029964447021484375  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 117.52717960233952  Num\_iterations: 15  Convergence\_time: 0.007977962493896484  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 224.87694288553567  Num\_iterations: 8  Convergence\_time: 0.0019996166229248047  Found\_color: {'Pink'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 46.92199955837047  Num\_iterations: 22  Convergence\_time: 0.01502084732055664  Found\_color: {'Grey', 'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |
| Img6 | Img7 | Img8 | Img9 | Img10 |
| K: 7  WCD: 228.7659958831942  Num\_iterations: 48  Convergence\_time: 0.0379643440246582  Found\_color: {'Grey', 'White', 'Orange', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 6  WCD: 256.100895673273  Num\_iterations: 22  Convergence\_time: 0.01125192642211914  Found\_color: {'Pink'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 245.23956868706057  Num\_iterations: 4  Convergence\_time: 0.000997304916381836  Found\_color: {'Grey', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 25.59950827721661  Num\_iterations: 7  Convergence\_time: 0.0020067691802978516  Found\_color: {'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 8  WCD: 164.48339336004668  Num\_iterations: 22  Convergence\_time: 0.01703500747680664  Found\_color: {'Pink', 'Red', 'Blue', 'Grey'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

Treashold 10:

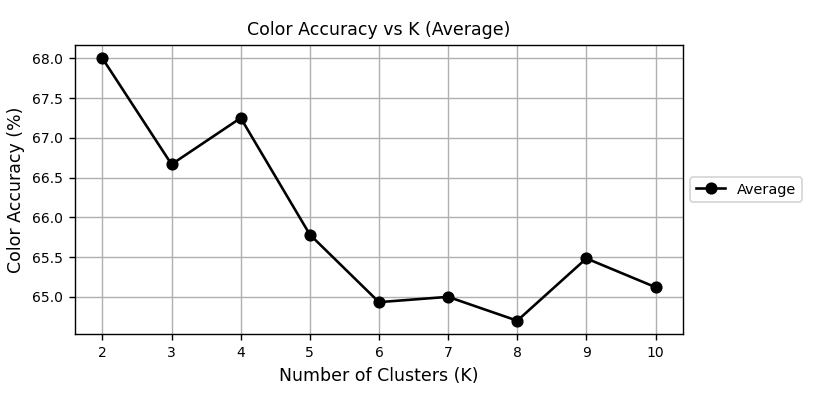
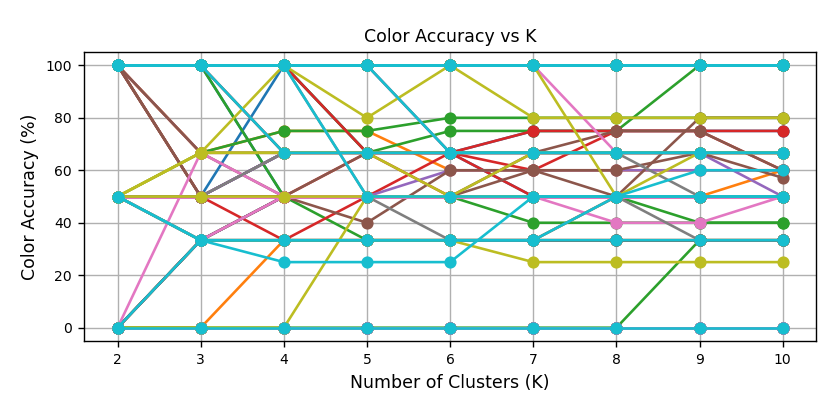
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Img1 | Img2 | Img3 | Img4 | Img5 |
| K: 4  WCD: 657.3465287043884  Num\_iterations: 14  Convergence\_time: 0.007711172103881836  Found\_color: {'Pink', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 2479.2066301140903  Num\_iterations: 9  Convergence\_time: 0.0029964447021484375  Found\_color: {'White', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 117.52717960233952  Num\_iterations: 15  Convergence\_time: 0.007977962493896484  Found\_color: {'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 224.87694288553567  Num\_iterations: 8  Convergence\_time: 0.0019996166229248047  Found\_color: {'Pink'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 10  WCD: 46.92199955837047  Num\_iterations: 22  Convergence\_time: 0.01502084732055664  Found\_color: {'Grey', 'Blue'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 0.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |
| Img6 | Img7 | Img8 | Img9 | Img10 |
| K: 7  WCD: 228.7659958831942  Num\_iterations: 48  Convergence\_time: 0.0379643440246582  Found\_color: {'Grey', 'White', 'Orange', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 6  WCD: 256.100895673273  Num\_iterations: 22  Convergence\_time: 0.01125192642211914  Found\_color: {'Pink'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 245.23956868706057  Num\_iterations: 4  Convergence\_time: 0.000997304916381836  Found\_color: {'Grey', 'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 50.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 2  WCD: 25.59950827721661  Num\_iterations: 7  Convergence\_time: 0.0020067691802978516  Found\_color: {'Black'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 100.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 | K: 8  WCD: 164.48339336004668  Num\_iterations: 22  Convergence\_time: 0.01703500747680664  Found\_color: {'Pink', 'Red', 'Blue', 'Grey'}  Color\_gt: ['Pink', 'Black']  Color\_accuracy: 25.0  Found\_shape: Handbags  Shape\_gt: Handbags  Shape\_accuracy: 100.0 |

### Test 6

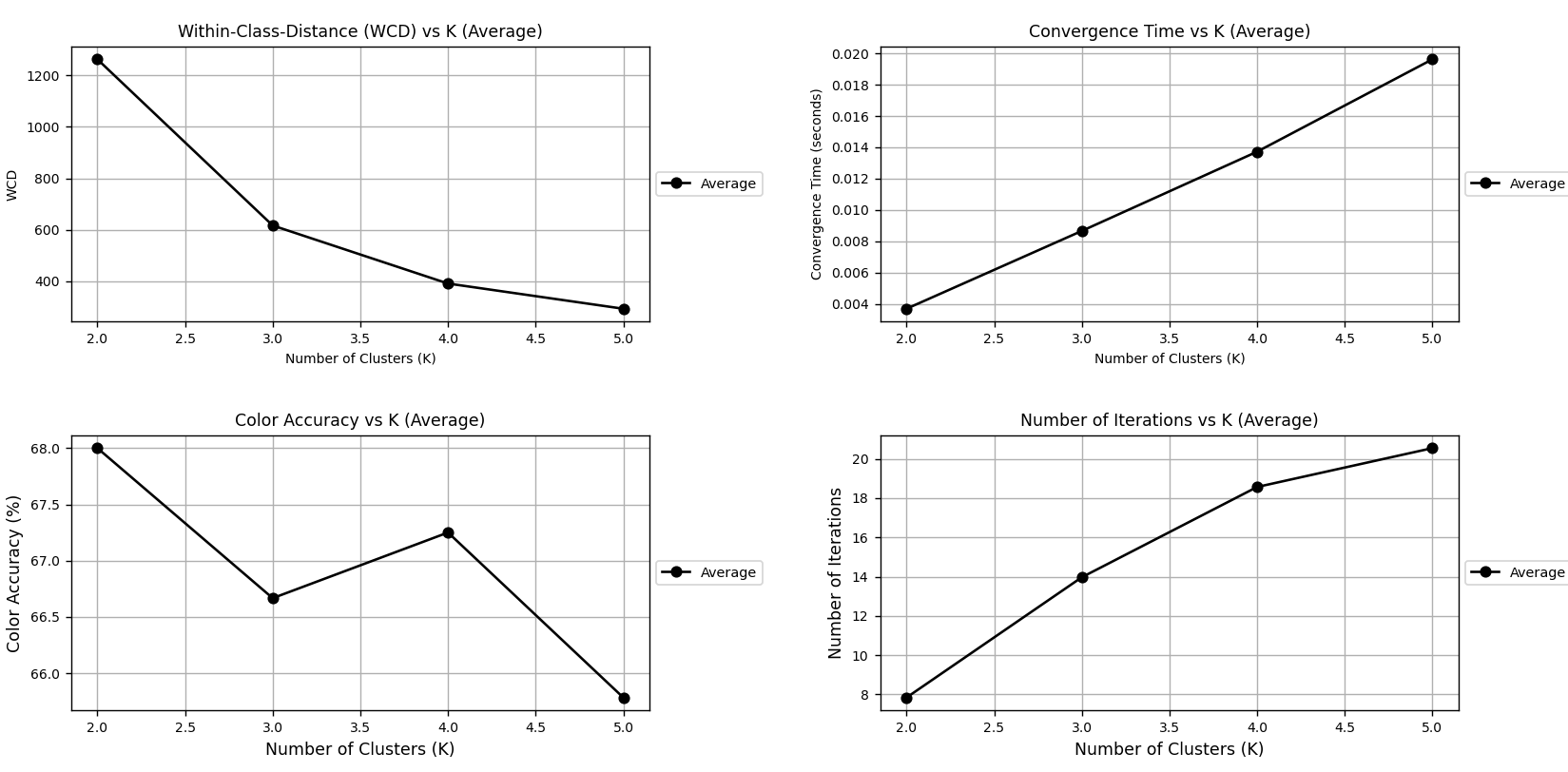
Finalmente hemos realizado una prueba con 100 imágenes sin recortar y hemos observado la estadística de cómo cambia el accuracy.

Threashold 20 / 15 / 10 / 25 / 30:

MaxK = 10



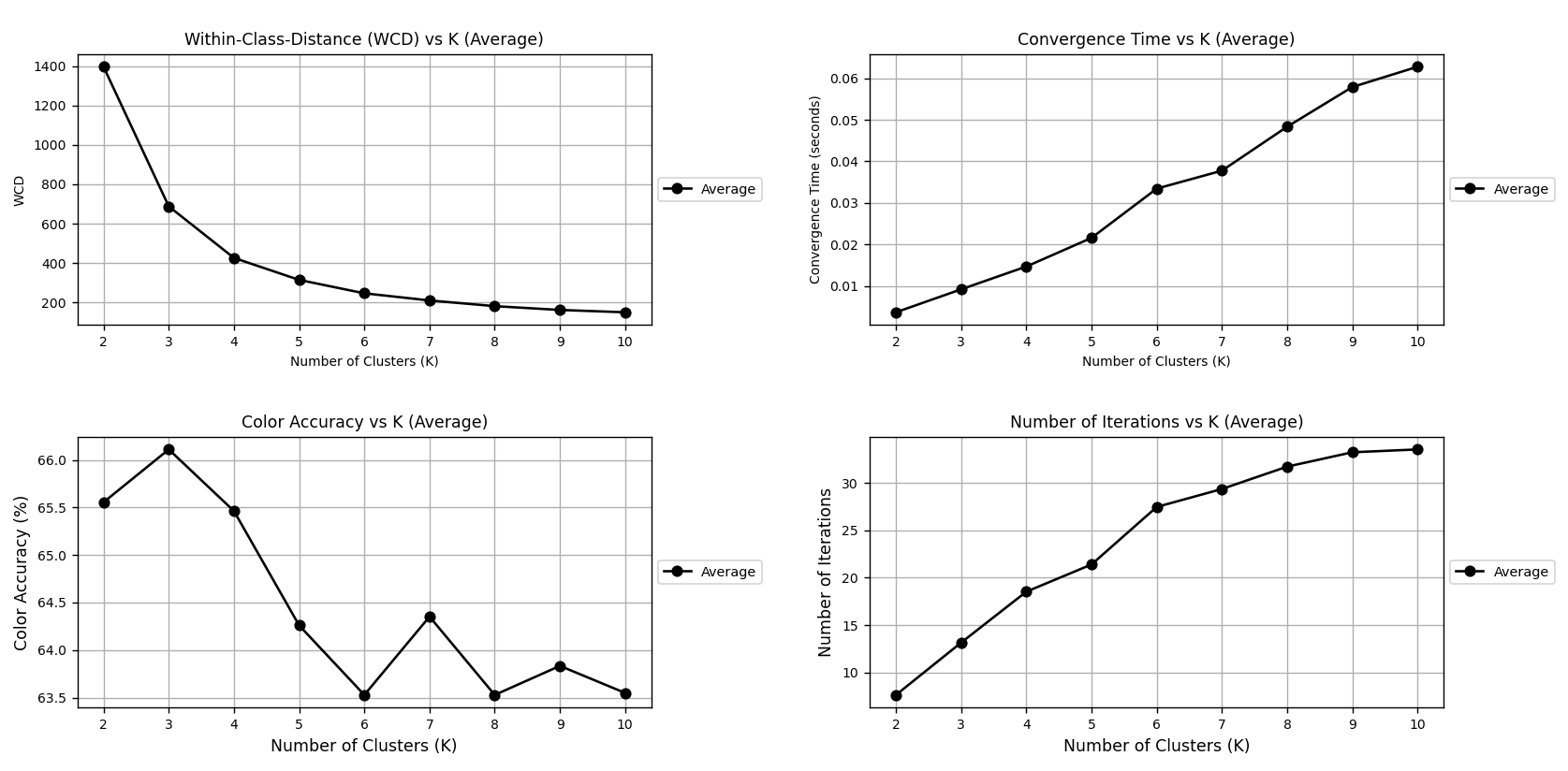
MaxK = 5



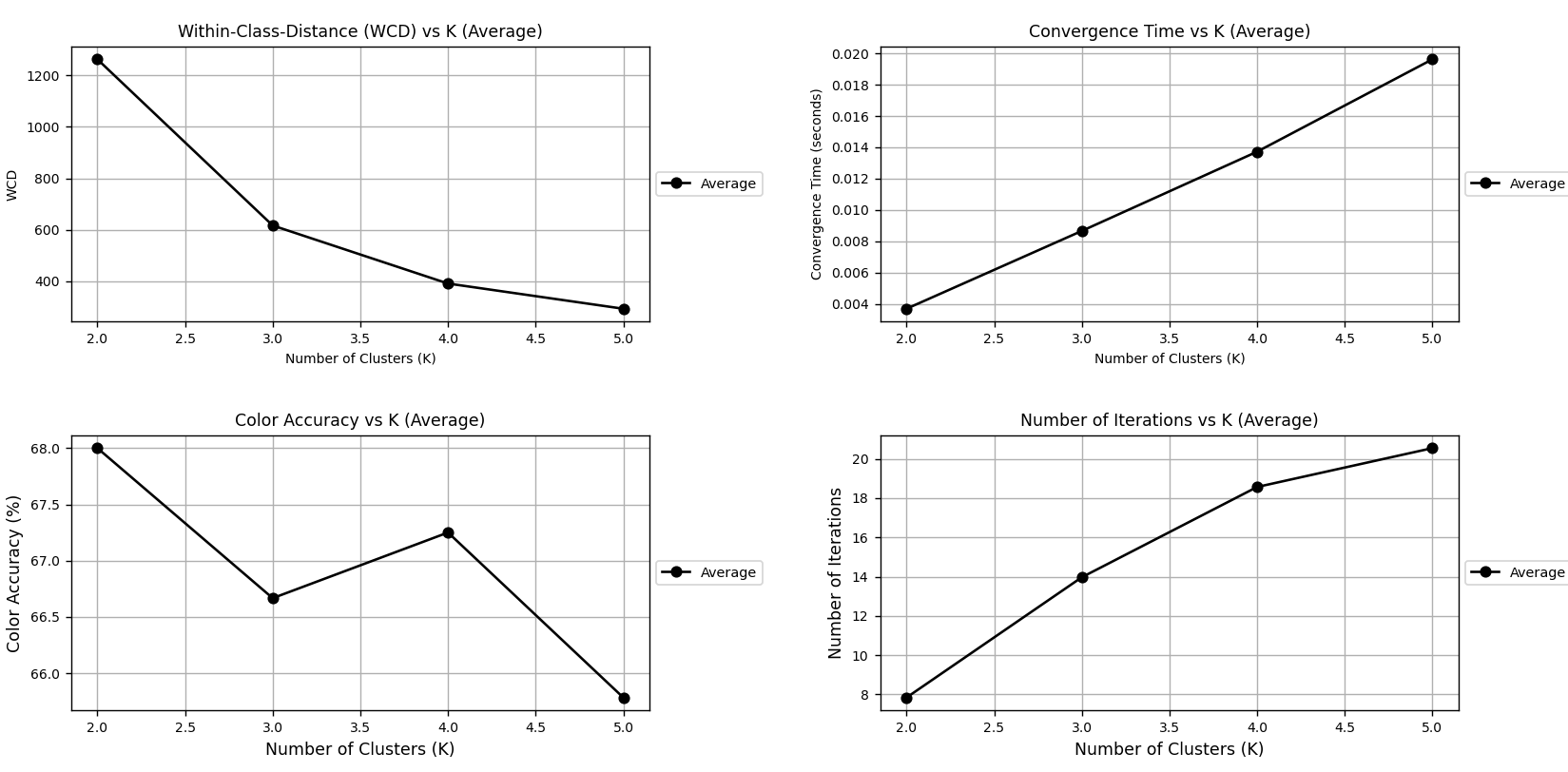
Con 1000 imágenes:

MaxK =10

Se mantiene igual que el accuracy de 100 imágenes.



MaxK = 5



### Conclusiones Find\_best\_K

Las conclusiones que he podido sacar de esta mejora son mixtas, por gran parte de lo realizado estaba convencido que había algún error en lo que estaba realizando ya que no veía ninguna mejora sustancial en lo que es el color accuracy y la funcionalidad del Kmeans. Si que hay cierta mejora ejecutando las opciones que proporcionan más aleatoriedad por así decirlo, pero por el resto no he notado cambios. La mayoría del tiempo se han repetido las estadísticas para los 5 valores del treashold.

También he podido apreciar que apartir del K > 4 realmente no merece la pena ejecutarlo ya que no beneficiaría el tiempo ni la funcionalidad del programa. Hasta en algunos casos al no tener tantos colores aparecen predicciones muy fuera de lo común.

Otras observaciones que tenemos es que a medida que va subiendo la K tenemos una WCD mucho mejor, pero comparando con los resultados no necesariamente es bueno ya que tener un WCD muy bajo a costa de tener K = 10 no nos aporta ningún beneficio, sólo inconvenientes y posibles fallos del accuracy.

# Conclusiones Globales

En conclusión, en esta práctica hemos intentado solucionar un problema de etiquetado automatizado de imágenes de prendas utilizando los algoritmos K-Means y KNN. Se ha utilizado el K-Means para encontrar los colores principales de las imágenes y el KNN se ha utilizado para determinar el tipo de prenda.

Para las funciones cualitativas hemos implementado todas las funciones que se nos ha sugerido, siendo estas el retrieve\_by\_color, retrieve\_by\_shape y retrieve\_combined. Hemos realizado varias pruebas para comprobar el correcto funcionamiento de las funciones. En estos tests hemos visto que el algoritmo de K-Means era el que más mejoras necesitaba, ya que en casos done la imagen tuviese fondo o el modelo estuviese presente fallaba mucho en la predicción de los colores de las prendas.

Para las funciones cuantitativas hemos desarrollado funciones para estudiar el porcentaje de acierto para las predicciones del K-Means y el KNN. Además, hemos implementado varias versiones de k\_means\_statistics para poder evaluar el K-Means y el KNN con diferentes valores para la k y poder procesar varias imágenes. También hemos desarrolado varias funciones auxiliares que hemos utilizado en k\_means\_statistics para crear gráficas que representen las diferentes medidas que hemos calculado al ejecutar los algoritmos varias iteraciones.

Respecto a las mejoras para el K-Means, hemos visto que las que más nos han ayudado para mejorar el rendimiento han sido los nuevos métodos de inicialización de centroides: Kmeans++ e inicialización aleatoria.

Para la mejora del Best\_K consideramos que lo mejor que se puede hacer es fijar el maxK alrededor del 4 o 5 ya que no beneficia un valor superior. Combiando con el resto de mejora como el random y kmeans++ mejora sustancialmente el tiempo y las iteraciones.

En resumen, esta práctica nos ha permitido adentrarnos un poco en la aplicación de la IA y a comprender mejor cómo son los procesos de análisis de rendimiento para poder encontrar mejoras para nuestros códigos.