Centro de Investigación en Matemáticas (CIMAT). Unidad Monterrey

Tarea 3

Marcelo Alberto Sanchez Zaragoza

22 de marzo de 2021

1. Problema 1

Considera un modelo de mezclas de k distribuciones

$$f(x) = \sum_{k=1}^{K} w_k f_k(X)$$

Supón que tienes datos x_1, x_2,...,$x_n \sim f(x)$, y queremos ajustar el modelo de mezclas de Gaussianas (MMG) para usarlo como un soft-clustering.

1.1. Inciso a)

Para el primer inciso vamos a partir de lo siguiente:

$$p(X_i = x) = \sum_{k=1}^{K} P(Z_i = k) P(X_i = x | Z_i = k) = \sum_{k=1}^{K} w_k N(\mu_k, \Sigma_k)$$

Observamos que $Z_i \in 1, ..., K$ es la variable latente que nos ayuda a identificar el componente de la mezcla para X_i. Para los párametros desconocidos $\theta = \{\mu_1, ..., \mu_k, \Sigma_1, ..., \Sigma_k,, w_1, ..., w_k\}$, donde la función de densidad conjunta es:

$$L(\Theta, X_i, ..., X_k) = \prod_{k=1}^{n} \sum_{k=1}^{K} w_k N(x_i; \mu_k, \Sigma_k)$$

Ahora aplicando logaritmo tenemos lo siguiente:

$$l(\Theta) = \sum_{i=1}^{n} log\left(\sum_{k=1}^{K} w_k N(x_i; \mu_k, \Sigma_k)\right)$$

Observe que la suma de los componenete de K no permite que la función lofarítmica para que no aplique a las densidades normales, de modo que al diferenciar con respecto a μ_k e igualando a 0, entonces tenemos:

$$\sum_{i=1}^{n} \frac{w_k N(x_i; \mu_k, \Sigma_k)}{\sum_{k=1}^{K} w_k N(x_i; \mu_k, \Sigma_k)} \left(\frac{(x_i - \mu_k)}{\Sigma_k}\right)$$

Ahora si tomamos la distancia aposteriori de Z_i se tiene:

$$\begin{aligned} P(z_i = k|X_i) &= \frac{P(X_i|Z_i = k)P(Z_i = k)}{P(X_i)} \\ &= \frac{w_k N(\mu_k, \Sigma_k)}{\sum_{k=1}^{K} w_k N(\mu_k, \Sigma_k)} = \gamma(k) \end{aligned}$$

Expresando de mejor manera nuestra ecuación tenemos:

$$\sum_{i=1}^{n} \gamma(k) \frac{(x_i - \mu_k)}{\sum_k}$$

Debido a que la expresión del principo ya esta derivada respecto a μ_k solo

nos resta encontrar los puntos criticos igualando a cero, así:

$$\sum_{i=1}^{n} \gamma(k) \frac{x_i - \mu_k}{\sum_k} = 0$$
$$\sum_{i=1}^{n} \frac{\gamma(k)x_i - \gamma(k)\mu_k}{\sum_k} = 0$$

Donde finalmente tenemos: $\hat{\mu}_k = \frac{\sum_{i=1}^{n} \gamma(k)x_i}{\sum_{i=1}^{n} \gamma(k)}$
Podemos observar que $\hat{\mu}_k$ se podria ver como un promedio ponderado de los datos con los respectivos pesos, por lo que para $\hat{\Sigma}$ y w_k, vamos a tener:

$$\hat{\Sigma} = \frac{1}{N_k} \sum_{i=1}^{n} \gamma(k)(x_i - \mu_k)^2$$
$$\hat{w}_k = \frac{N_k}{n}$$

1.2. Inciso b)

Para desarrollar este inciso se emplearon datos que siguen una distribución Normal bivariada, para poder ilustrar mejor el algoritmo solicitado solo se tomaron dos muestras, es decir, nuestra mezcla de datos estara conformado por solo 2 grupos de datos. En la figura 1.1 se muestra como nuestros datos se encuentran mezclados, pero para hacer diferencia entre los grupos se le dio un color en especifico a cada uno.

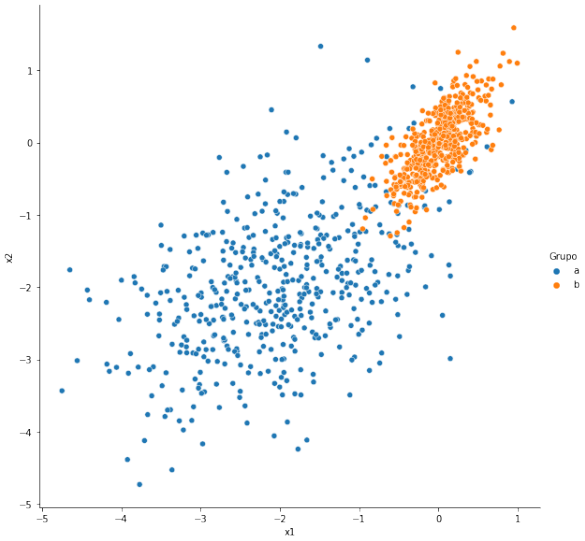

Figura 1.1: Distribución de los datos.

Observe que para el grupo 'a' los datos que se presentan están más cerca uno del otro y en el grupo 'b' se encuentran ligeramente más dispersos, esto con la intención de ilustrar mejor el algoritmo.
Los parametros que se ocuparon para cada conjunto se muestra a continuación: para el primer conjunto 'b', el color azul, tomamos una $\mu_1 = (-2,-2)^t$ y una matriz $\Sigma_1 = \begin{pmatrix} 1 & 0{,}5 \\ 0{,}5 & 1 \end{pmatrix}$, para nuestro segundo conjunto 'a' , color naranja, tomamos una $\mu_2 = (0,0)^t$ y una matriz $\Sigma_2 = \begin{pmatrix} 0{,}1 & 0{,}1 \\ 0{,}1 & 0{,}2 \end{pmatrix}$.

Una vez que tenemos nuestros datos, el algoritmo se implemento en un lenguaje de programación llamado $PYTHON$, que nos ayuda con los calculos ya que muchos de ellos son repetitivos y son demasiado extensos. El algoritmo no se va a describir en esta sección ya que solo nos interesa observa

los resultados de compararlo contra $fuzzy\ k-means$. También para implementar dicho algoritmo se necesito de dar valores iniciales para una matriz de medias, varianzas y pesos.
En la siguiente figura 1.2 se ilustra el resultado de implementar el algoritmo solicitado.

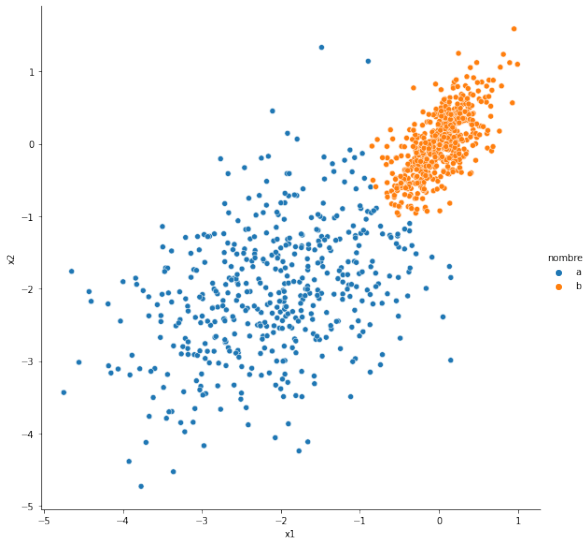

Figura 1.2: Resultado del algoritmo implementado.

En este caso observamos grupos muy parecidos a los que se presentaron en un inicio, son pocos los datos que se encuentran en un grupo equivocado. Para el caso de $fuzzy\ k-means$ tenemos lo siguiente:

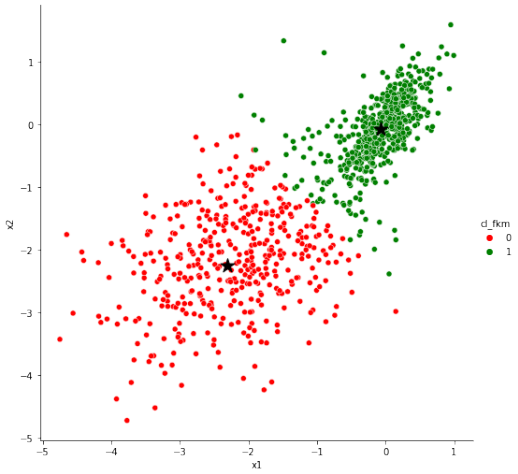

Figura 1.3: Resultado del algoritmo $fuzzy\ k-means$.

En la figura 1.3 se observa que nuestra mezcla de datos esta dividida, incluso podriamos poner una linea recta. En este caso se agrego el centro de cada grupo y puede que el centro coincida con la media que proporcionamos al principio pero la elección de los datos para grupo no es la adecuada.

1.2.1. CASO 1

Ahora vamos a variar un poco las matrices de medias y las matrices de varianzas y covarianzas para observar que sucede en algunos casos, si se desea repetir dichos resultados se anexa el código que se ocupo y donde lo único que se debe hacer es ingresar valores y ejecutar.
En la figura 1.4 se puede observar que nuestros datos siguen siendo conformados por dos grupos pero la varianza se modifico de forma que tenemos los datos más distribuidos y más mezclados.

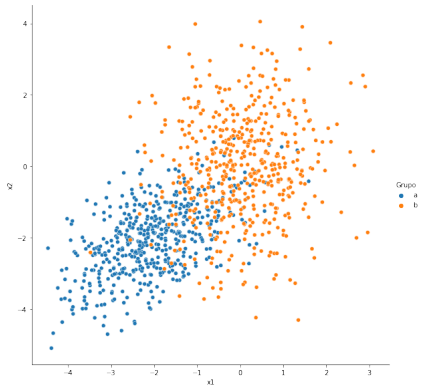

Figura 1.4: Datos con mayor varianza y centros cerca.

En la figura 1.5 muestra el resultado de la implementación del algoritmo MMG-EM, donde no agrupa nuestros datos como quisieramos.

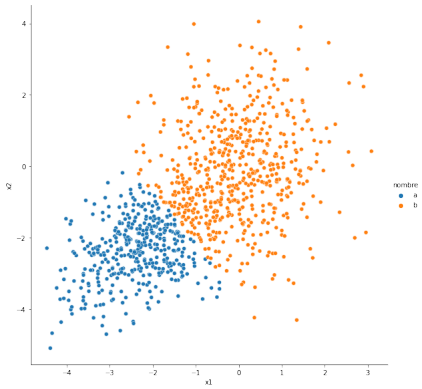

Figura 1.5: Resultado del algoritmo implementado.

Al hacer la comparación $fuzzy\ k-means$ notamos que tienen cierto parecido, en la figura 1.6 nos muestra que la selección de los datos hasta

cierto punto es buena.

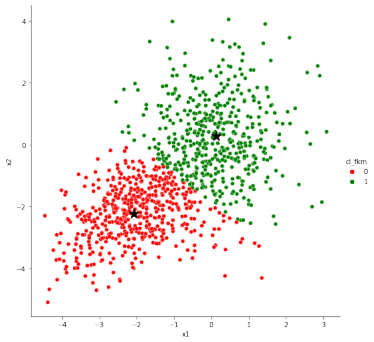

Figura 1.6: Resultado del algoritmo $fuzzy\ k-means$.

Los resultados de ambos métodos se podrian justificar por la forma que tomamos nuestros datos. Hay que tomar en cuenta que para muchos otros casos donde se tomen matrices de medias y varianzas, ambos no tendran el mismo resultado. Las matrices que se emplearon para este caso se presentan a continuación:

$$\mu_1 = \begin{pmatrix} -2 \\ -2 \end{pmatrix} \quad \Sigma_1 = \begin{pmatrix} 1 & 0{,}5 \\ 0{,}5 & 1 \end{pmatrix}$$
$$\mu_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Sigma_2 = \begin{pmatrix} 1 & 0{,}1 \\ 0{,}1 & 2 \end{pmatrix}$$

1.2.2. CASO 2

Para el ultimo ejemplo vamos a tomar valores de los vectores de medias alejados y observar que sucede.

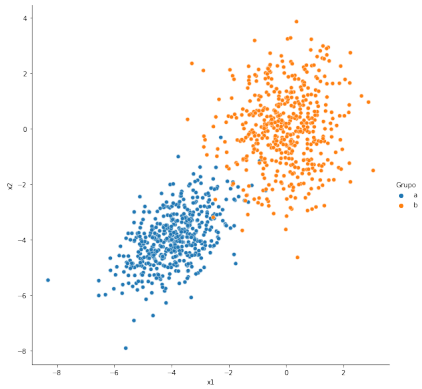

Figura 1.7: Datos con mayor varianza y centros alejados.

Con el algotimo obtenemos lo siguiente:

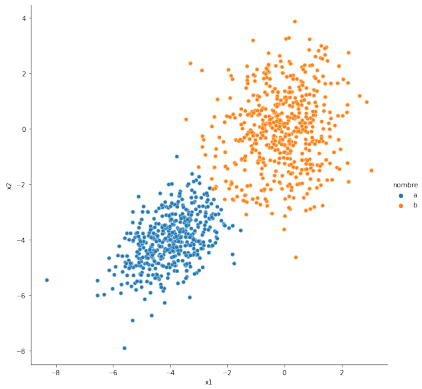

Figura 1.8: Resultado del algoritmo implementado.

Con $fuzzy\ k-means$ obtenemos:

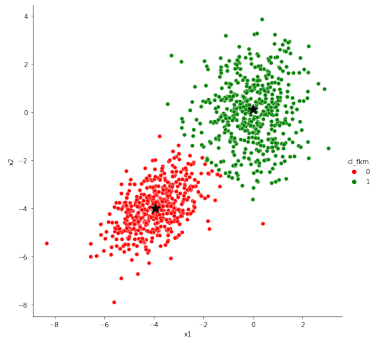

Figura 1.9: Resultado del algoritmo $fuzzy\ k-means$.

Las matrices que se emplearon para este caso se presentan a continuación:

$$\mu_1 = \begin{pmatrix} -4 \\ -4 \end{pmatrix} \quad \Sigma_1 = \begin{pmatrix} 1 & 0{,}5 \\ 0{,}5 & 1 \end{pmatrix}$$
$$\mu_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Sigma_2 = \begin{pmatrix} 1 & 0{,}1 \\ 0{,}1 & 2 \end{pmatrix}$$

En ambas graficas notamos una buena selección de los datos por lo que para ciertos casos ambos métodos funcionan bien pero para el caso que se mostro al principio de esta sección, si logra ilustra para que clase de mezclas podemos emplear el algoritmo MMG-EM que nos proporcionan ya que selecciona bien nuestros datos en determinados grupos, mientras que en $fuzzy\ k-means$ pareciera que nos divide el plano con una linea y con eso comienza la selección de datos para cada grupo.

1.3. Inciso c)

En la siguiente sección nos proponen considerar la matriz $\Sigma_k = \sigma^2 I$, además que cada grupo de datos tenga la misma matriz de varianzas y

covarianzas. De igual forma demostrar que cuando $\sigma \to 0$ el algoritmo MMG-EM agrupa de la misma forma que $k - means$.
Veamos un primer caso para los datos que nos solicitan.

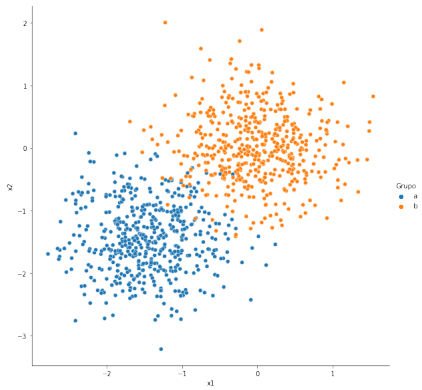

Figura 1.10: Datos con la misma matriz Σ.

En la figura 1.10 observamos que nuestros datos toman cierto comportamiento 'circular' en este caso nuestros centros están ligeramente lejanos, uno en (0, 0) y otro en (-1.5, 1.5). Lo que podemos observar que al hacer la matriz de varianzas y convarianzas($\Sigma_k = \sigma^2 I$), tener valores cada vez más cercanos a cero hace ese efecto 'circular' en nuestros datos.

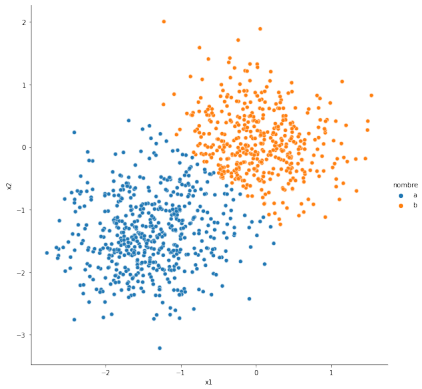

Figura 1.11: Resultado del algoritmo implementado MMG-EM.

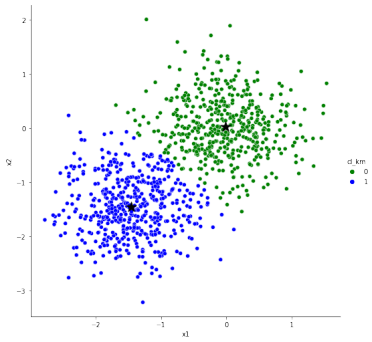

Figura 1.12: Resultado del algoritmo $k - means$.

En ambas graficas notamos cierta coincidencia en los resultados. Ahora vamos a utilizar los mismos centros pero haremos más pequeños los valores de la matriz Σ.
La figura 1.13 hace más evidente el comportamiento 'esferico' de nuestros datos.

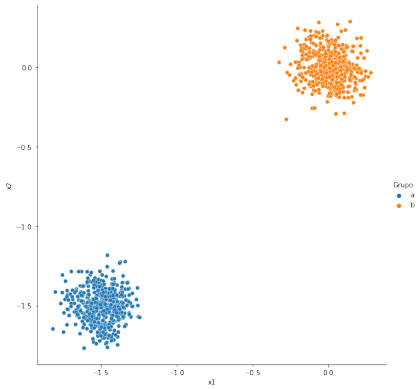

Figura 1.13: Datos con la misma matriz $\Sigma \to 0$.

En las siguientes figuras vamos a observar los resultados de ambos algoritmos.

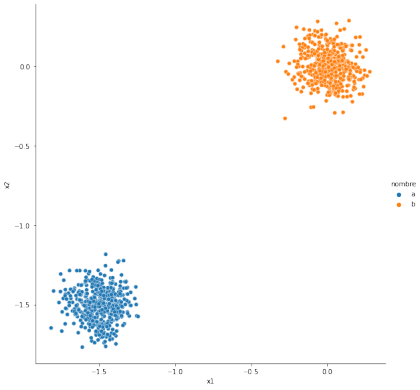

Figura 1.14: Resultado del algoritmo implementado MMG-EM.

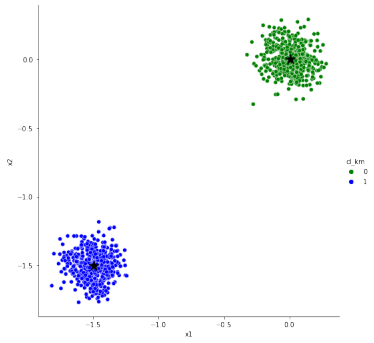

Figura 1.15: Resultado del algoritmo $k - means$.

Los resultados son buenos para ambos algoritmos pero recordemos que los centros están ligeramente alejados. Ahora solo con la intención de ilustrar otro caso vamos a tomar vectores de medias más cercanos y ver que sucede. Los vectores de medias los colocamos más cercanos $\mu_1 = (0{,}1, 0{,}1)^t$ y $\mu_2 = (0, 0)^t$.

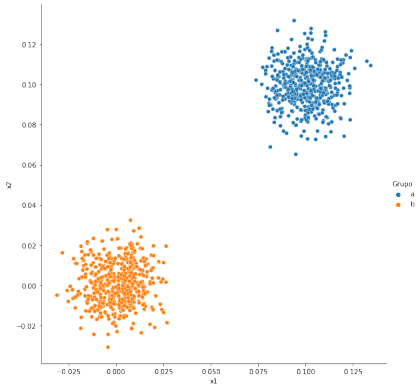

Figura 1.16: Datos con la misma matriz $\Sigma \to 0$.

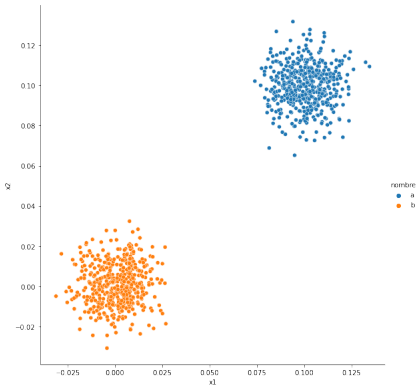

Figura 1.17: Resultado del algoritmo implementado MMG-EM.

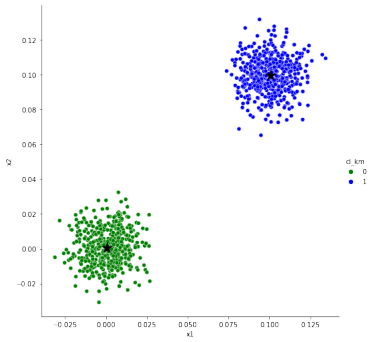

Figura 1.18: Resultado del algoritmo $k - means$.

Repitiendo el mismo proceso y conservando los centros los datos se van agrupando más y más de forma que la mezcla de los datos se divide poco a poco en nubes de datos circulares, esto nos ayuda mucho ya que teniendo este comportamiento ambos algoritmos nos dan resultados iguales y buenos.

2. Problema 2

En el siguiente problema nos apoyamos del paper Inderjit Dhillon, Yuqiang Guan and Brian Kulis: A Unified view of Kernel k-means, Spectral Clustering and Graph Cuts. UTCS Technical Report, 2005.
Con ayuda de este paper pudimos implementarlo en el lenguaje de programación *PYTHON*, como medida de apoyo seleccionamos nuestros datos siendo cautelosos, ya que nuestra intención es ilustrar el resultado final de dicho algoritmo. Si se desea consulta el algoritmo se anexa el paper.

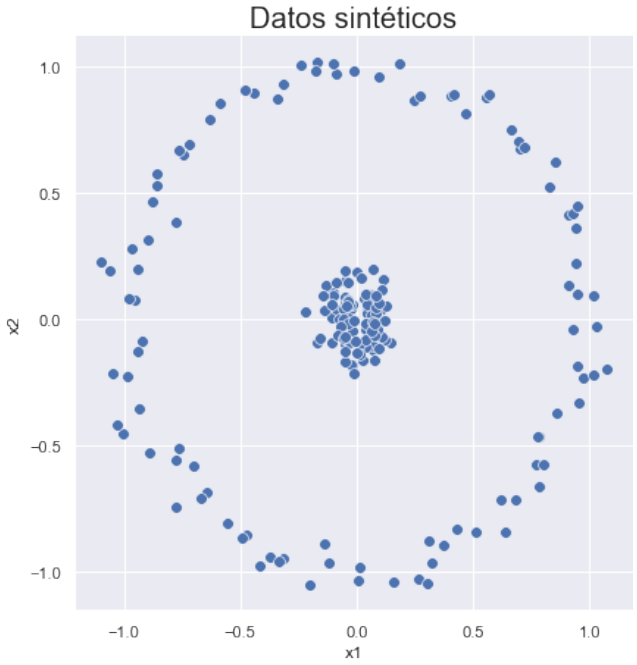

Figura 2.1: Distribución de los datos sinteticos.

Se observa que nuestros datos forman dos grupos e implementar otros métodos quiza no nos ayuden a separar de manera correcta nuestros datos. En la siguiente figura

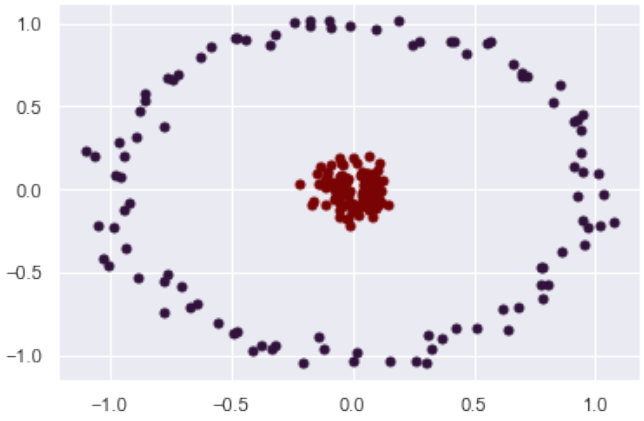

Figura 2.2: Algoritmo de kernel k-means.

En la figura 2.2 se ilustra como los datos están siendo seleccionados de manera correcta en dos grupos. Con fines de mostrar la eficiencia del método vamos a compararlo contra $k - means$ y $fuzzy - k - means$. Las ilustraciones las encontramos en la figura 2.3 y figura 2.4 respectivamente.

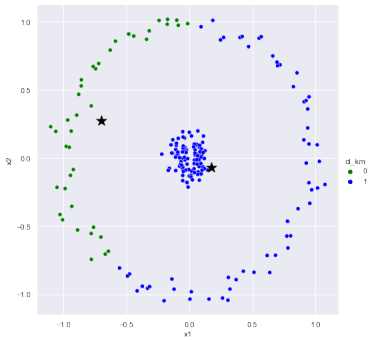

Figura 2.3: Algoritmo $k - means$.

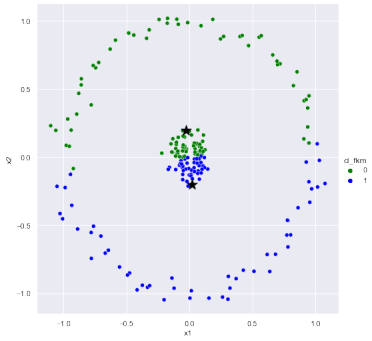

Figura 2.4: Algoritmo $fuzzy - k - means$.

Con ambos metódos observamos que nuestra selección de datos no es la mejor, debido a que parte nuestros datos como si hubiera una linea imaginaria y esto no es correcto ya que a simple vista vemos que hay dos grupos de datos pero estos métodos no funcionan muy bien.

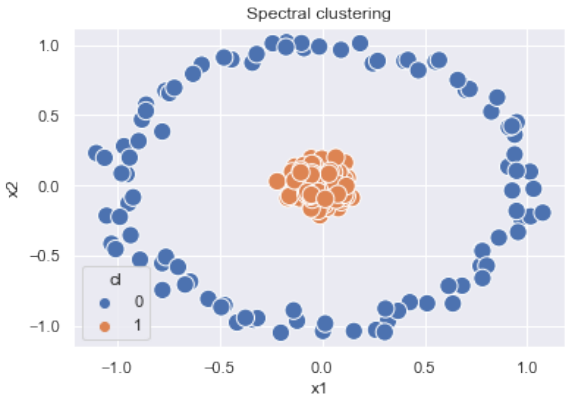

Figura 2.5: Algoritmo *Spectral clustering*.

En la figura 2.5 observamos un mejor comportamiento en la elección de los dos grupos de datos, comparado con los dos anteriores pues es claramente mejor.
Al implementar el algoritmo solicitado y con una minuciosa elección de los datos podemos encontrar una correcta elección de los dos grupos de datos y compararlos con los demás hace evidente la ventaja. Los parametros ocupados o tamaños de las muestras se podran observar en el código implementado, el cual se anexara para poder probar con distintos valores.

Al igual que en las partes anteriores vamos a mostrar otra estructura de datos y observar que método nos funciona mejor. En la figura 2.6 se muestra la forma que tienen nuestros datos sinteticos.

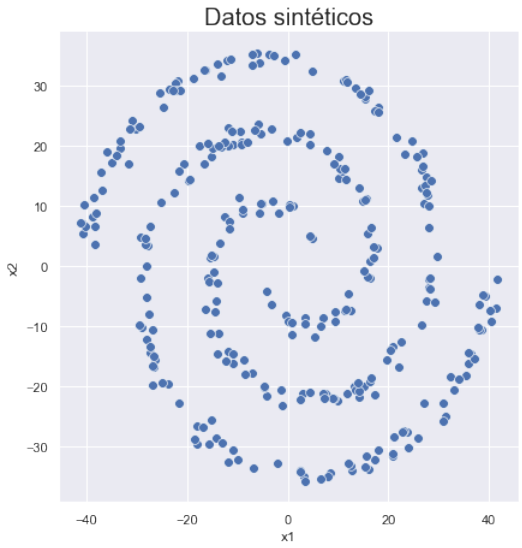

Figura 2.6: Distribución de los datos sinteticos.

En la figura 2.7 nos enseña el resultado de implementar el algoritmo basado en el paper.

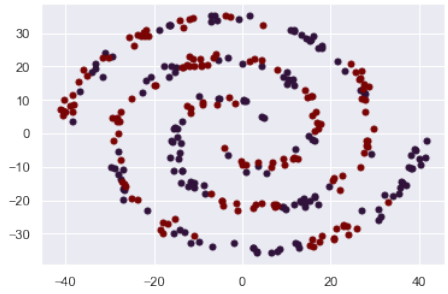

Figura 2.7: Algoritmo de kernel k-means.

En las siguientes figuras mostramos los resultados de $k-means$, $fuzzy\ k-means$ y *Spectral clustering*.

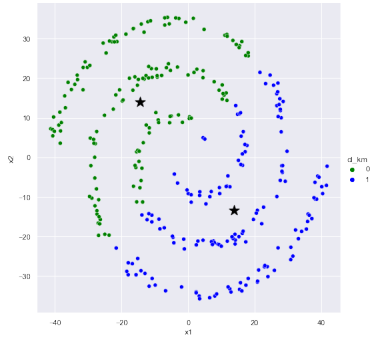

Figura 2.8: Algoritmo $k-means$.

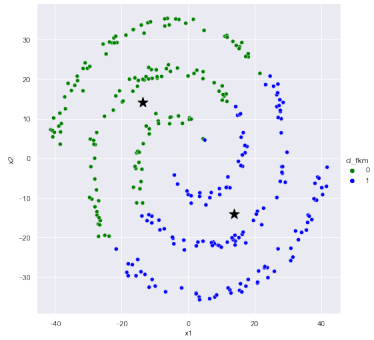

Figura 2.9: Algoritmo $fuzzy\ k-means$.

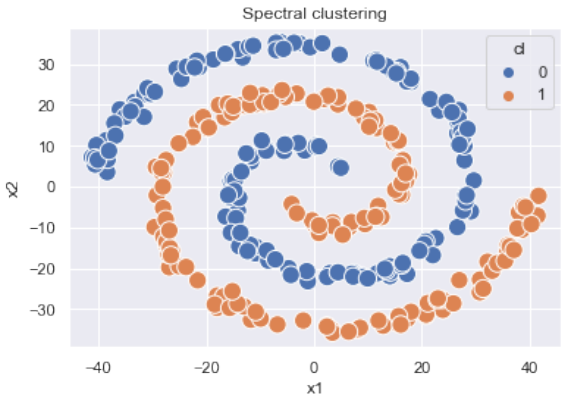

Figura 2.10: Algoritmo *Spectral clustering*.

A manera de conclusión de este pequeño ejemplo extra observamos que el método que estamos implementado depende en ocasiones del comportamiento de los datos, un factor importante que puede ser determinante son los parametros que ingresamos al momento de empezar el algoritmo. En este caso no nos ayudo a hacer los dos grupos de datos que queremos pero al final se observa que nos funciona mejor *Spectral clustering*.

3. Problema 3

En la siguiente sección se nos propocino un archivo $data fuits\_tarea.zip$ que contiene una serie de imagenes de distintas frutas, con tamaño de 100x100 pixeles. Se desea tratar de indentificar los dinstintos grupos de frutas.

Figura 3.1: Ejemplo: Imagen de una manzana.

Los incisos que debemos realizar son los siguientes:

a) Obtén una representación de las imágenes en el espacio RGB usando la mediana como medida de resumen en cada canal ¿Se logra identificar patrones interesantes en esta representación?

b) Realiza PCA y Kernel PCA con un kernel Gaussiano en los datos que obtuviste. ¿Se puede indentificar grupos interesantes o informativos de las imágenes en los primeros componentes principales?

c) Aplica K-means y Kernel K-means. Verifica si puedes identificar los diferentes grupoes de frutas.

d) Repite los incisos 3b y 3c usando el espacio HSV. Para incluir más información sobre cada dimensión, utiliza la información de los tres cuartiles centrales en cada una de ellas, de forma que tengas una representación en un espacio de tamaño d=9. ¿Notas alguna mejoría?

3.1. Inciso a)

Como ejemplo de una imagen en los canales RGB tenemos en la figura 3.2 una pequeña ilustración de lo que son estos canales, en algunos de ellos se presentan ciertas formas o sobras que para determinanda tarea nos pueden ayudar a reconocer ciertos patrones.

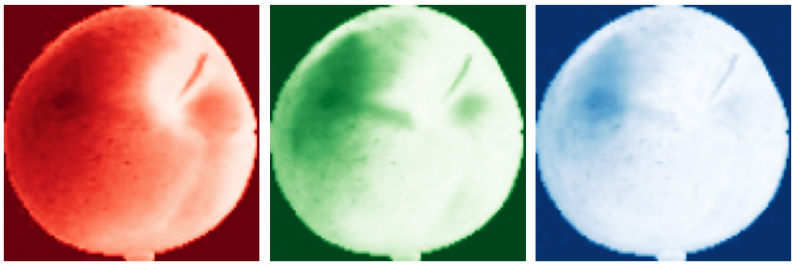

Figura 3.2: Ejemplo: Imagen de una manzana en canales RGB.

Ya que hemos ilustrado el canal en el cual vamos a trabajar, en este primer inciso se nos pide ocupar la mediana como medida de cada una de las imagenes, cabe mencionar que contamos con un total de 1300 imagenes por lo que para realizar dicha tarea nos apoyaremos del lenguaje de programación *Python*. En la figura 3.3 se muestra la impresión de los distintos tipos de frutas en un grafico de tres dimensiones, son tres dimensiones ya que estamos trabajando con el canal RGB.

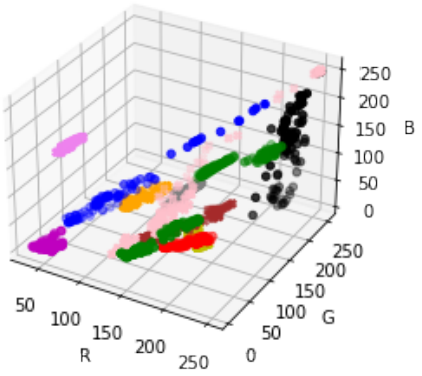

Figura 3.3: Representación en los canales RGB.

Como primera ilustración vemos que muchos de nuestros grupos de frutas se encuentran separados y otros se encuentran juntos, el mismo grafico no ayuda mucho debido a que se encuentra algo fijo por lo que vamos a mostrar una rotación del grafico.

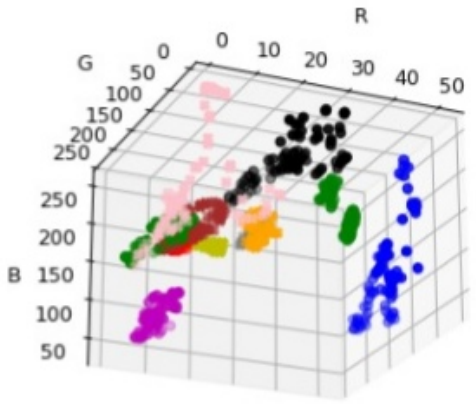

Figura 3.4: Representación en los canales RGB, rotación.

Antes de realizar la rotación si se logra observar cierta fruta alejada de todas las demás, en este caso estamos hablando del arándano que que en el grafico tiene un color rosa claro. El grupo conformado por las manzanas que ente caso contamos con manzanas rojas, golden y Gran-smith, se ve ligeramente distribuido y no en aglomerado en un solo punto como se quisiera pensar, llevan un color verde. Respecto a las demás furtas notamos ciertos grupos algo evidentes pero logran cruzarse un poco en algunos casos, con otros grupos y puede que sea algo complicado hacer la diferencia entre ellos.
En la figura 3.4 se omitieron los registros de la fruta de arandano para po-

der apreciar más las demás frutas, ya que ellos ya se encuentran separados. Hay que recordar que al realizar rotaciones del grafico ayudan a observar que desde cierta rotación los datos no se tocan o incluso se puden aislar pero dependeria mucho de la rotación que se ocupe.

3.2. Inciso b)

Realizamos PCA y kernel PCA con un kernel Gaussiano a las imágenes en el espacio RGB utilizando la mediana como resumen de información. En la figura 3.5 se muestran los datos proyectados en los dos primeros componentes principales, para esta representación se ve mejor la diferencia del grupo de imagenes de arandanos con las demás ya que esta se encuentra en la parte superior del grafico, totalmente aislada. Con las demás futas no sucede exactamente lo mismo pero si podemos decir que existe cierta diferencia entre ellas.

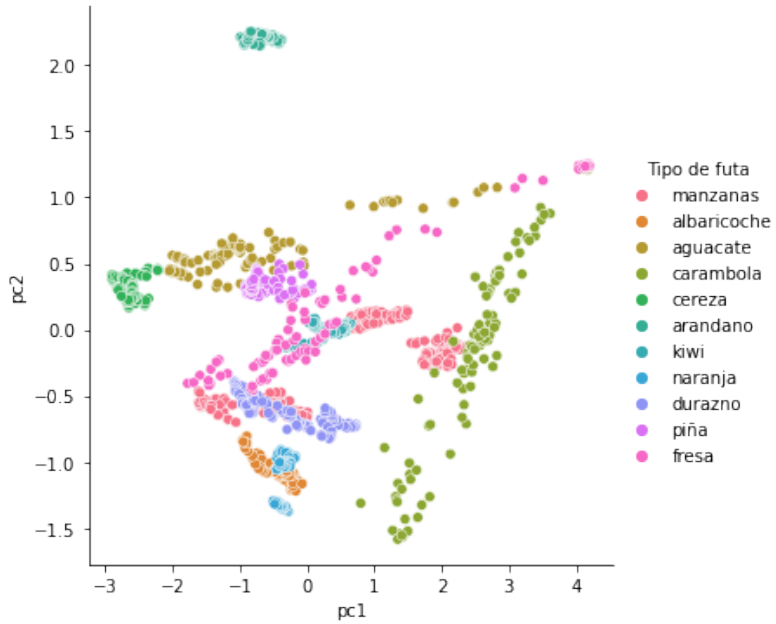

Figura 3.5: Representación de PCA en los dos primeros componentes principales.

Agregando al grafico que la figura 3.5 nos enseña es que grupos como los conformados por las cerezas si son identificados, este en la parte izquierda y muy juntos uno del otro. El grupo formado por las imagenes de carambola se encuentra ligeramente extendido pero no muy relacionado más que con el grupo de las manzanas, se puede atribuir a la coincidencia de cierto grupo de las manzanas debido al color, pero solo con ese grupo. Otro grupo muy aglomerado es el correspondiente a las piñas, se encuentra muy cerca de los aguacates pero podemos hacer enfasis en que sus imagenes solo se encuentran en una nube de puntos muy cerca uno del otro. Los demás grupos si se encuentran más combinados.

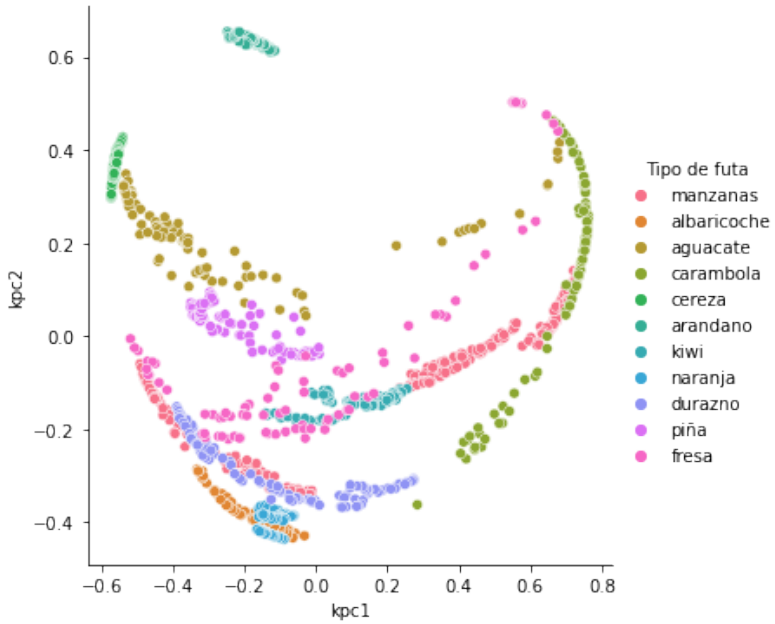

Figura 3.6: Representación con Kernel-PCA.

En la figura 3.6 nos muestra el resultado de ocupar Kernel-PCA con los primeros componentes principales, en este caso nos ayuda a 'extender' los distintos grupos, vemos que ciertos grupos se encuentran más cercanos y otros nos lo separa. Comparamdo las dos gráficas hay ciertas ventajas de uno y el otro pero como la intención es encontrar grupos o patrones para identificar las frutas, podemos decir que PCA ayuda ligeramente más a encontrar algunos grupos.

3.3. Inciso c)

Al mismo conjunto de datos de medianas de las imagenes en los canales de RGB les aplicamos K-means y Kernel K-means, en este caso tomamos 11 clusters como valor inicial.

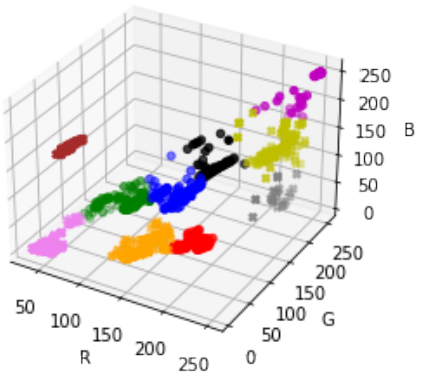

Figura 3.7: K-means.

En la figura 3.7 si prestamos atención podemos identificar ciertos grupos de frutas, comparando esta imagen con la primer representación que se hizo en el inciso a) se logran ver mejor los grupos, se puede observar que los puntos se juntaron más, es decir, se acercaron más entre ellos los puntos de un mismo tipo de fruta. Entonces si hay cierta mejoria y si es evidente, también hay que notar que los grupos no se encuentran muy combinados entre ellos.

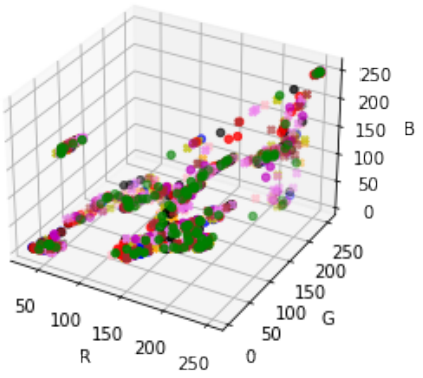

Figura 3.8: Kernel K-means.

En la figura 3.8 es complicado ahora encontrar estos patrones, ya que como vemos los distintos puntos se encuentran en varios lugares y se podría rotar el gráfico pero uno desearía que a la primer representación encontraramos una mejor vista de los grupo. Este grafico no ayuda mucho a identificar estos grupos.

3.4. Inciso d)

En este último inciso nos piden hacer los incisos b) y c) pero ocupando los canales de HSV. Antes de empezar a mostrar las graficas vamos a mostrar como es este nuevo canal.

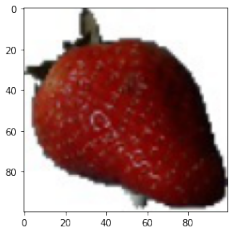

Figura 3.9: Ejemplo: Imagen de una fresa.

Como se menciono anteriormente, este canal también nos ayuda a recuperar cierta información que nos podría ayudar a hacer cierta diferencia entre las imagenes, en la 3.10 se muestra a manera de ejemplo como son estas caracteristicas que se rescatan.

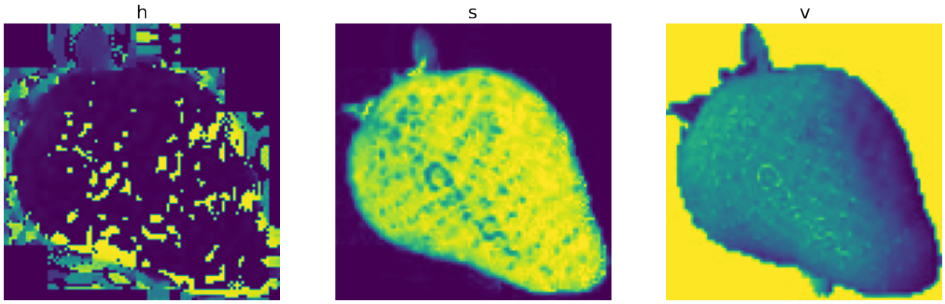

Figura 3.10: Representación en los canales HSV.

Ahora mostramos los resultados de realizar PCA y Kernel PCA. Se agrega la representación en tres dimensiones con fin ilustrativo.

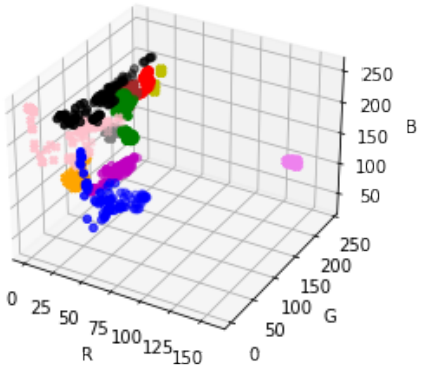

Figura 3.11: Representación en los canales HSV.

En la figura 3.12 se puede observar como ayuda a distinguir más algunos tipos de frutas, en este canal hace más evidente la diferencia que hay entre las imagenes de cerezas con las demás. Algunos grupos todavía se encuentran combinados y otros ligeramente movidos pero este canal si ayuda a distinguir grupos.

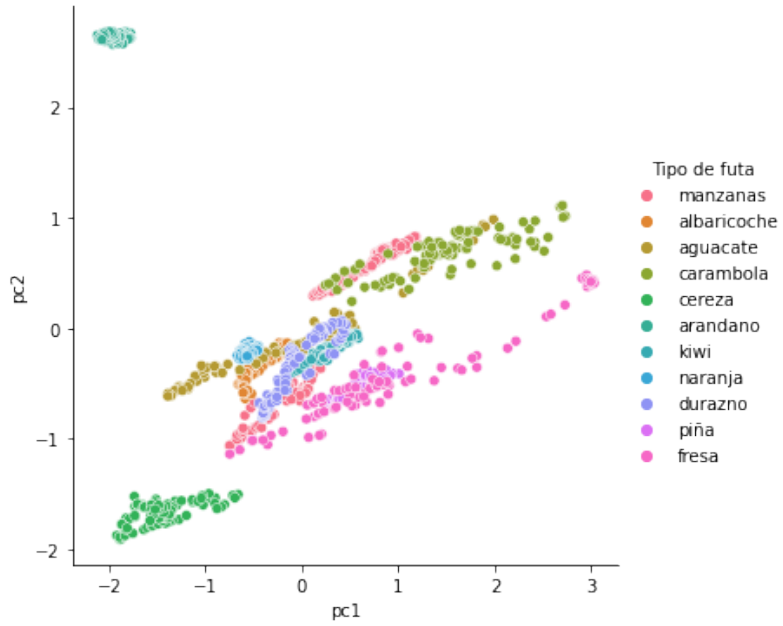

Figura 3.12: Representación de PCA en los dos primeros componentes principales.

En la figura 3.13 ayuda a diferenciar otros grupos, por ejemplo la carambola esta sobre una tipo curva pero preste atención a que no guarda mucha relación con los demás grupos, también rescata el grupo conformado por el arandano. el grupo de las manzas todavía se convina con las demás frutas esto puede ser atribuido a los colores de la fruta, forma o alguna otra caracteristica que hace confundir al modelo. Otra fruta que agrupa de forma adecuada es la piña, tiene uno que otro punto de otra fruta pero se logra diferenciar el grupo, al igual con la cereza, albaricoche, durazno y naranja, las demás ya se encuentran más combinadas.

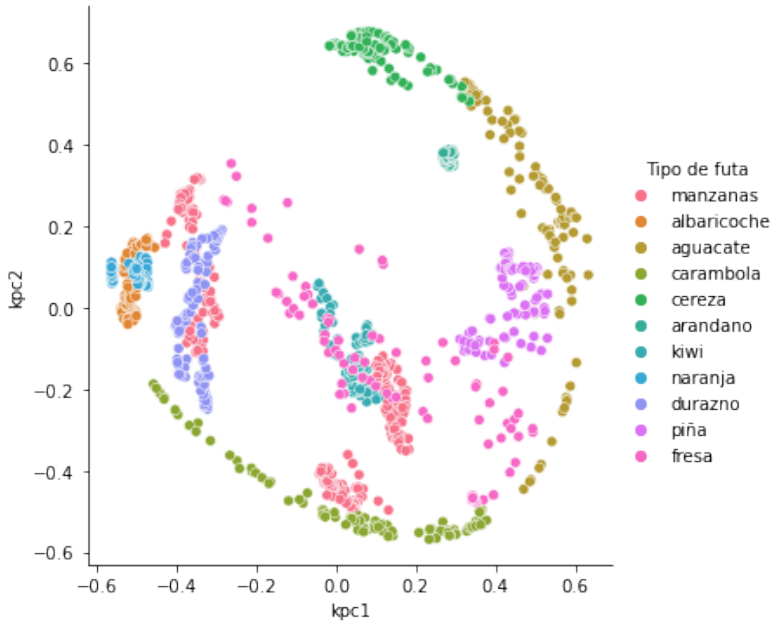

Figura 3.13: Representación con Kernel-PCA.

En la figura 3.14 nos muestra nuevamente un grafico en tres dimensiones y sucede lo mismo que con el canal RGB, ya que nos compacta más los grupos, en la figura 3.10 se ve más separados pero aqui los junta un poco pero hay que prestar atención que no los junta y revuelve tanta, solo compacta más los grupos.

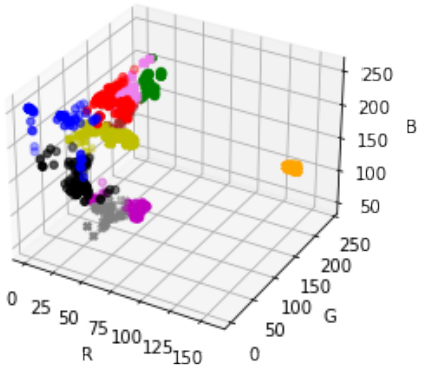

Figura 3.14: K-means.

En esta figura si es dificil encontrar un patrón principalmente por la forma del grafico y también porque mezcla los puntos de cada tipo de fruta.

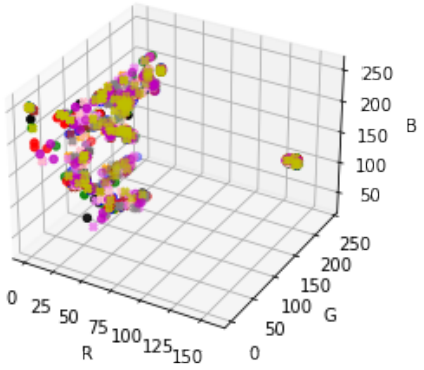

Figura 3.15: Kernel K-means.

A manera de conclusión de este ultimo ejercicio podemos decir que para distintos canales se observo muchos patrones, analizar PCA y los demás métodos en ambos nos ayuda para darnos un panorama de que canales se pueden tomar en cuenta para identificar cierto tipo de fruta y cuales ayudan para otros métodos. De igual forma para los grafico en tres dimensiones se recuerda que dependeria mucho de la rotación que se llegara a realizar para encontrar patrones o formas, ya que como estamos en un espacio de tres dimensiones el número de rotaciones puede ser basto. Todos los resultados expuestos se encuentran en un código que se anexa y donde se explica que se hizo en cada parte.