Centro de Investigación en Matemáticas (CIMAT). Unidad Monterrey

Tarea 4

Hairo Ulises Miranda Belmonte

April 1, 2019

1 Problema

1. Implementa kernel k-means basandote en el artículo de Inderjit Dhillon, Yuqiang Guan and Brian Kulis: A Uni
ed view of Kernel k-means, Spectral Clustering and Graph Cuts. UTCS Technical Report, 2005. el cual está en la página del curso.

Escoge (o genera) algunos conjuntos de datos adecuados para verificar la eficiencia y ventajas del método. Compáralo con otros métodos para mostrar en qué casos es mejor su desempeño.

1.1 Solución

Se implementa Kernel K-means, con un kernel Gaussiano a datos no lineales, el código se encuentra en el archivo **"Tarea 5 Ejercicio 1"**. Se contrasta kernel k-means, respecto a clusters lineales como lo son; k-means, k-medoides y fuzzy k-means (los dos últimos con la librería **cluster**)y, técnicas que son lineales en el espacio de características para encontrar representaciones no lineales en el espacio original, como cluster espectral.

Se generan dos conjuntos de datos para verificar la eficiencia y ventajas de los métodos para distinto números de clusters. En la figura 1.1, se presentan los datos generados; en el gráfico "a", se toman 200 datos esféricos; en el "b", 100 datos en forma de espiral, ambos para probar con $k = 2$ (clases).

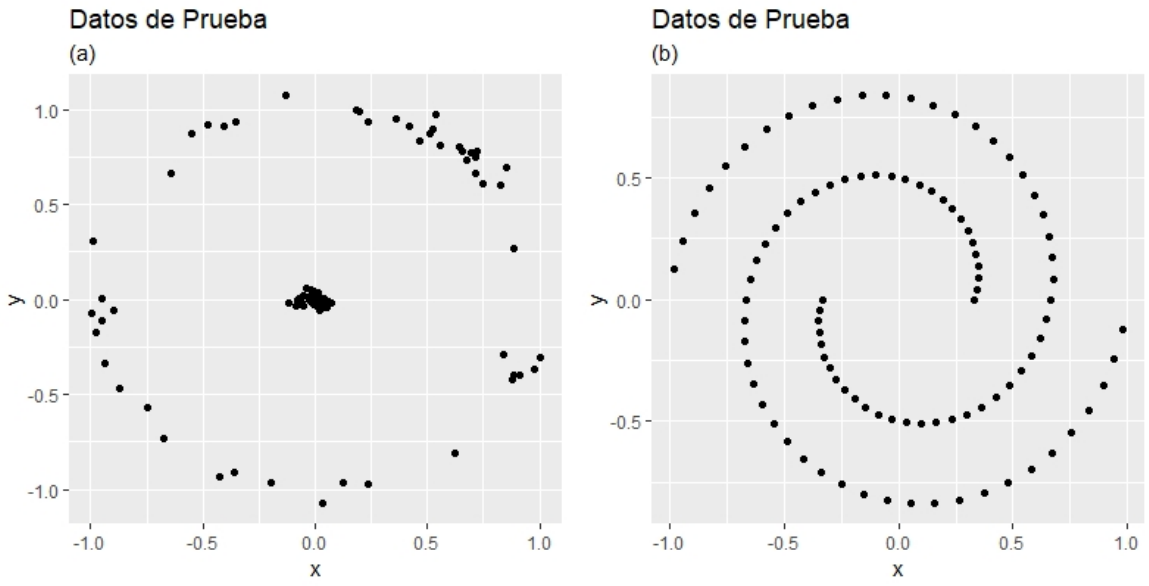


Figure 1.1: Datos de prueba

En la figura 1.2, se presentan los resultados para los conjuntos de datos. Se utiliza un kernel Gaussiano con parámetro $\sigma = 2$; se observa que para el conjunto de datos "a", el agrupamiento es correcto; sin embargo, para el conjunto de datos "b", kernel k-means no logra agrupar de forma correcta.

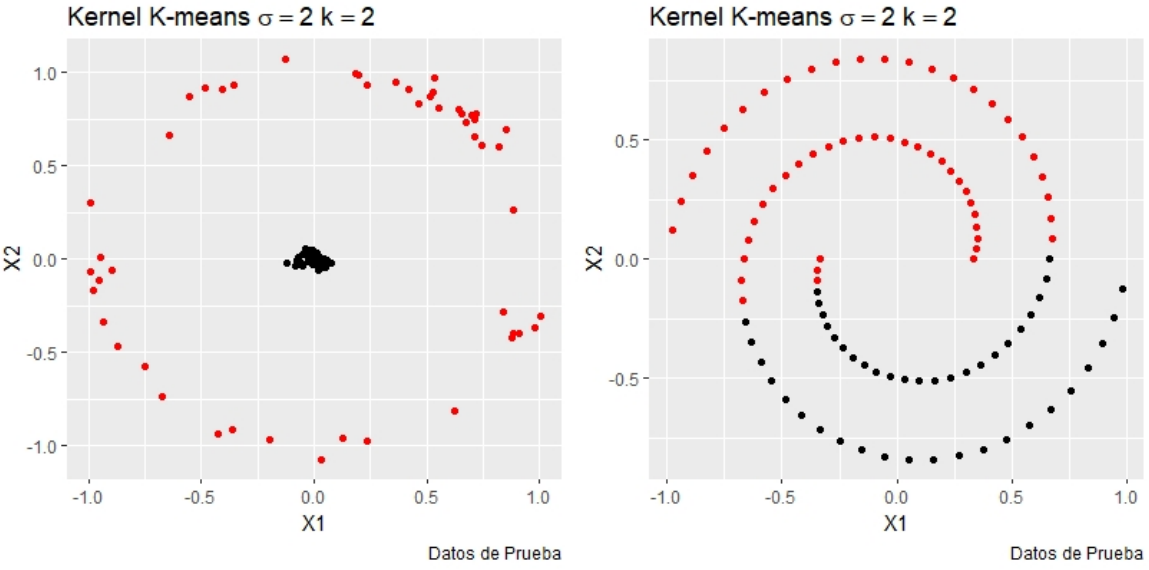


Figure 1.2: Agrupamiento figura (a)

Ahora, aplicamos técnicas distintas a kernel k-means para evaluar su desempeño. En la figura 1.3, se observa que para el conjunto de datos "a", k-means, k-medoides y fuzzy k-means, agrupan de forma incorrecta. Al realizar cluster espectral con $\sigma = 2$ (pequeño), no

se logra agrupar del todo bien; no obstante, con un sigma grande, se logra un agrupamiento adecuado.

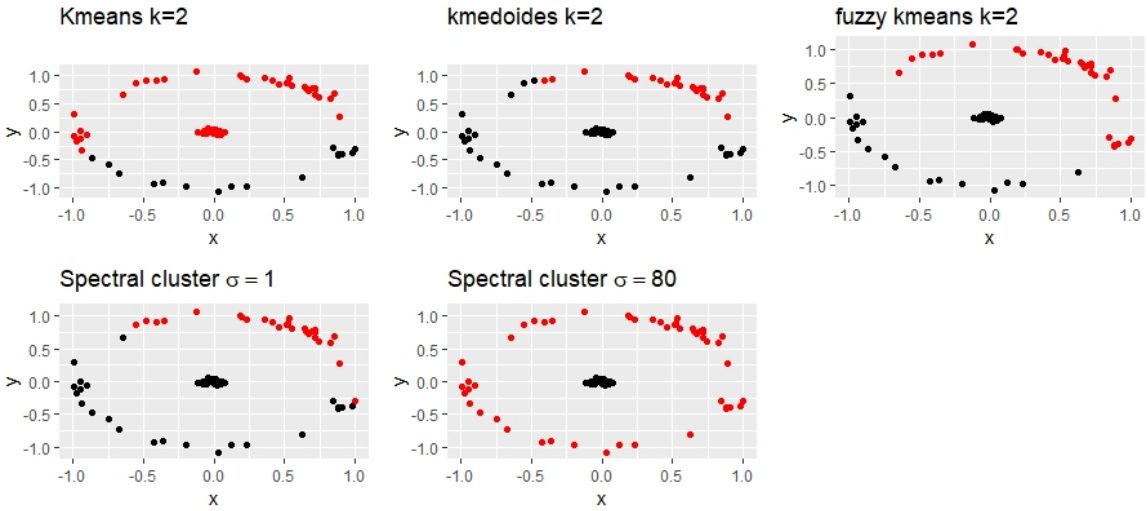


Figure 1.3: Agrupamiento figura (c)

La tabla 1.1, muestra la frecuencia de observaciones en cada grupo para cada técnica de la representación anterior.

Table 1.1: Frecuencia de observaciones en cada grupo

Clusters	$k = 1$	$k = 2$
K-means	32	168
K-medoides	152	48
fuzzy K-means	134	66
Cluster Espectral 1	146	54
Cluster Espectral 2	100	100
Kernel K-means	100	100

Cluster Espectral 1: $\sigma = 1$; Cluster Espectral 2: $\sigma = 80$; Kernel K-means 1: $\sigma = 2$

En la figura 1.4, se muestra el resultado del conjunto de datos "b". Podemos observar, que para todas la técnicas sólo el cluster espectral con $\sigma = 80$, y un kernel Gaussiano, realiza una agrupación incluso mejor que Kernel k-means.

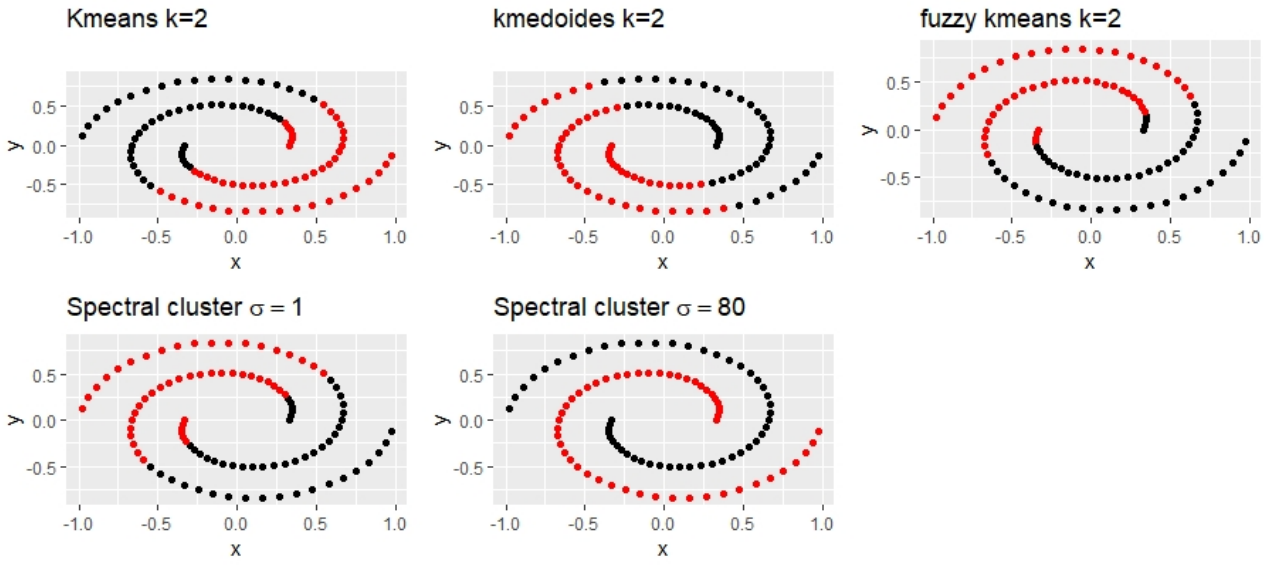


Figure 1.4: Agrupamiento figura (d)

La tabla 1.2, presenta la frecuencia de agrupación del gráfico de arriba. Como se observa, la mayoría de las técnicas agrupan las mismas cantidades de observaciones, pero como vimos en la figura anterior, de manera incorrecta.

Table 1.2: Frecuencia de observaciones en cada grupo

Clusters	$k = 1$	$k = 2$
K-means	50	50
K-medoides	50	50
fuzzy K-means	50	50
Cluster Espectral 1	50	50
Cluster Espectral 2	50	50
Kernel K-means	45	55

Cluster Espectral 1: $\sigma = 1$; Cluster Espectral 2: $\sigma = 80$; Kernel K-means 1: $\sigma = 1$; Kernel K-means 2: $\sigma = 80$

Se concluye que cluster espectral y kernel K-means, con un kernel Gaussiano y distintos niveles del parámetro, tienden a obtener resultados parecidos en representaciones no lineales en el espacio original de los datos.

2 Problema

2. Los datos en el archivo **datafruitstarea.zip** contienen imágenes pre procesadas de $100x100$ pixeles, que corresponden a diferentes tipos de frutas, tomadas en diferentes orientaciones y con diferentes características de forma y maduración. Supón que a una cadena de supermercados le interesa implementar un método automático de reconocimiento del tipo de fruta (y posiblemente su nivel de maduración) a través de las imágenes en color.

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

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

c) Aplica Kmeans y Kernel Kmeans. Verificar si puedes identificar los diferentes grupos de frutas.

d) Repite los incisos anteriores usando el espacio HSV (Hue, Saturation, Value). 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 mejora ?

2.1 Solución inciso "a"

Para este ejercicio se utiliza la librería **imager**, **rgl** y **grid**, que permite visualizar y manipular las imágenes. Con la librería **imager**, una imagen se visualiza en RGB, ejemplo la figura 2.1

Figure 2.1: Imagen de una manzana

La visualización de una imagen en cada canal de RGB se presenta en la figura 2.2.

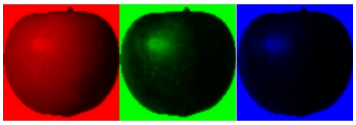


Figure 2.2: Imagen en canales R,G,B

La visualización en cada canal de la representación HSV se presenta en la figura 2.3.

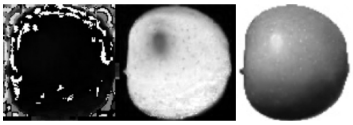
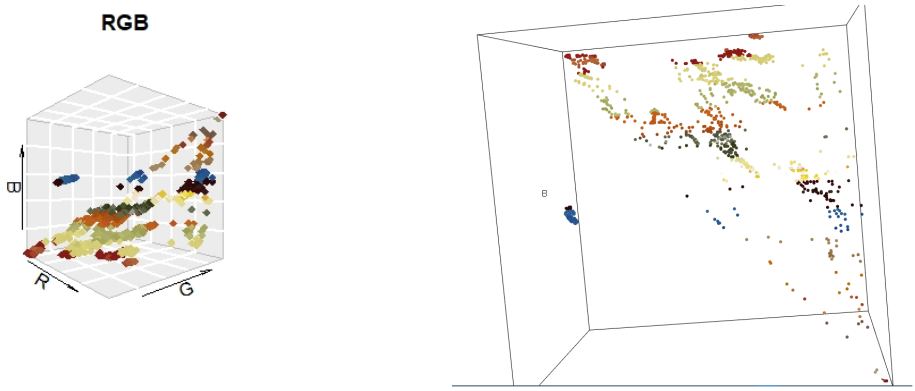


Figure 2.3: Imagen en canales H,S,V

Primero se pide representar las imágenes en el espacio RGB utilizando la mediana como resumen de valores en cada canal. En la figura 2.4a, se muestra de manera general la representación del espacio RGB; en la figura 2.4b, se rota la imagen con el fin de encontrar alguna patrón interesante.

En la figura 2.4b, se pueden ver algunos patrones, la separación del arándano (azul fuerte) hasta el extremo derecho con algunos puntos tintos de la cereza; las manzanas rojas se ubican en la parte superior del cubo, seguido de las manzanas golden (amarillo verdoso), y las manzana Gran -smith, después se identifica un tono naranja del durazno, seguido de un verde oscuro del aguacate y un tono amarillo de la carambula; asimismo, la cereza se mezcla con tonos azules del arándano (abajo del tono amarillo); después de esto, es difícil identificar las frutas restantes, piña, fresa, kiwi.

En algunas frutas (colores), se observa como la nitidez del color va disminuyendo, esas frutas representan aquellas que se encuentran pasando su etapa de maduración (se pudren).



(a)

(b) Imagen rotada del espacio RGB

Figure 2.4: Representación de las imágenes en el espacio RGB usando la mediana

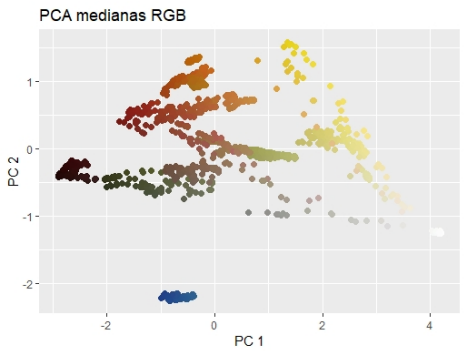
2.2 Solución inciso "b"

Se realiza PCA y Kernel PCA con un kernel Gaussiano en las imágenes en el espacio RGB utilizando la mediana como resumen de información.

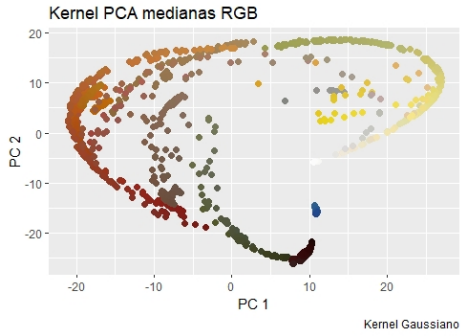
En la figura 2.2a, se observa la representación al realizar PCA, y proyectar las primeras dos componentes. Se logran identificar grupos de frutas por color, aquellas con colores oscuros en la parte izquierda; cerezas y aguacates, las manzanas rojas y las fresas, algunos duraznos y naranjas; los kiwis y las piñas en el centro de color café; en la parte de abajo, se separan los arándanos (azul oscuro); en la parte derecha del gráfico, las manzanas verdes, amarillas y la carambula.

En la figura 2.2b, se presentan las primeros dos componentes principales al utilizar Kernel PCA, con un kernel Gaussiano y $\sigma = 10$. Con esta representación se encuentran patrones más claros, por des-agregación de color, como los puntos rojos que pasan a naranja (manzana roja, fresas, duraznos y naranjas), después una linea café de los kiwis y piñas; en la parte inferior las cerezas en tinto, seguido de los aguacates; en la parte superior derecha las manzanas verdes, amarillas y la carambula.

Lo interesante de utilizar kernel PCA, es la forma en que va decolorando los puntos, en esta representación se puede ver la descomposición de la fruta, y como muchas frutas al pasar el proceso de maduración se confunden con aquellas cuyos colores se asemejan.



(a) Primeros dos componentes PCA

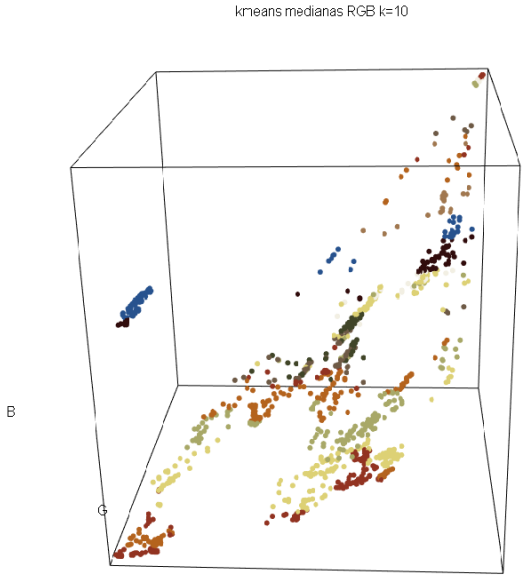


(b) Primeros dos componentes Kernel Gaussiano PCA $\sigma = 10$

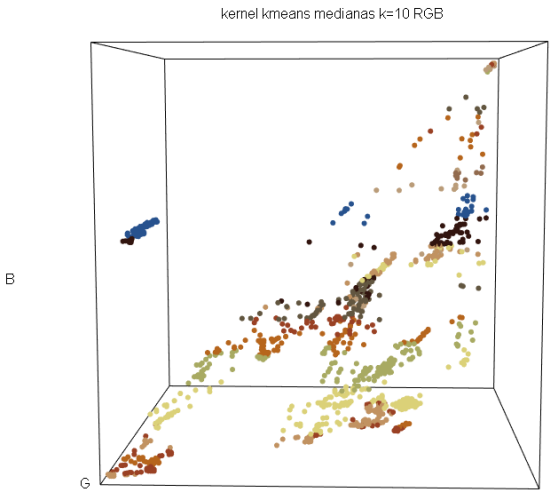
Figure 2.5: PCA y Kernel PCA en las medianas de las imagenes en el espacio RGB

2.3 Solución inciso "c"

Se le aplica a la mediana de los canales R,G,B, K-means y Kernel K-means, se utilizan 10 clusters como valor inicial, uno para cada fruta. En la figura 2.6a, se colorean los clusters depende donde el punto de la mediana se encuentre. No se observan patrones claros, se confunde al agrupar frutas naranjas con rojas, entre otras. En la figura 2.6b, se implementa Kernel k-menas con un kernel Gaussiano y $\sigma = 10$, mejora ligeramente el agrupamiento; algunos puntos como el café (kiwi, piña), los agrupa mejor. La tabla 2.1, presenta la frecuencia de agrupación de las observaciones en cada cluster, en el cual se observa gran diferencia entre métodos.



(a) K-means; $k = 10$



(b) kernel k-means; $k = 10$, $\sigma = 10$

Figure 2.6: K-means y Kernel K-means a las medianas en espacio RGB

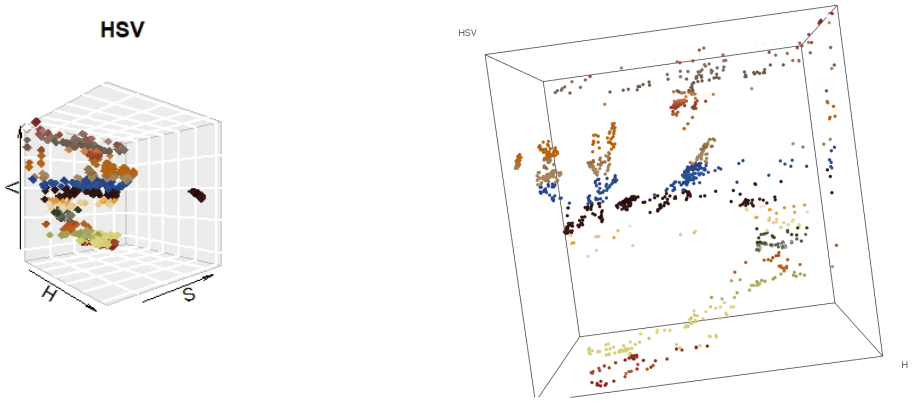
Clusters	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$	$k=10$
Kernel Kmeans	100	126	137	134	107	228	101	100	142	125
Kmeans	213	53	145	47	128	39	237	200	83	155

Table 2.1: Distribución de imágenes en clusters

2.4 Solución inciso "d"

Repetimos lo anterior usando el espacio HSV. Se utiliza la función **RGBtoHSV**, de la librería **imager** para pasar la imágenes de RGB a HVS y después calcular la mediana de cada canal. Esta función da los canales S y V en un rango de 0 a 1, pero el canal H (hue), se encuentra de 0 a 360[1], para esto se transforma el canal a una escala similar a los de S y V.

La figura 2.4, representa el espacio HSV con la mediana de cana imagen. Se encuentran patrones interesantes; por ejemplo, en la figura 2.4a, se observan bloques horizontales de colores que representan a cada fruta, y en la parte extrema. la cereza (color tinto) se separa en su totalidad. En la figura 2.4b, se tiene la rotación de la imagen 2.4a, donde se aprecia mejor lo ya mencionado; no obstante, bajo está representación no se puede decir mucho del proceso de maduración de las frutas.



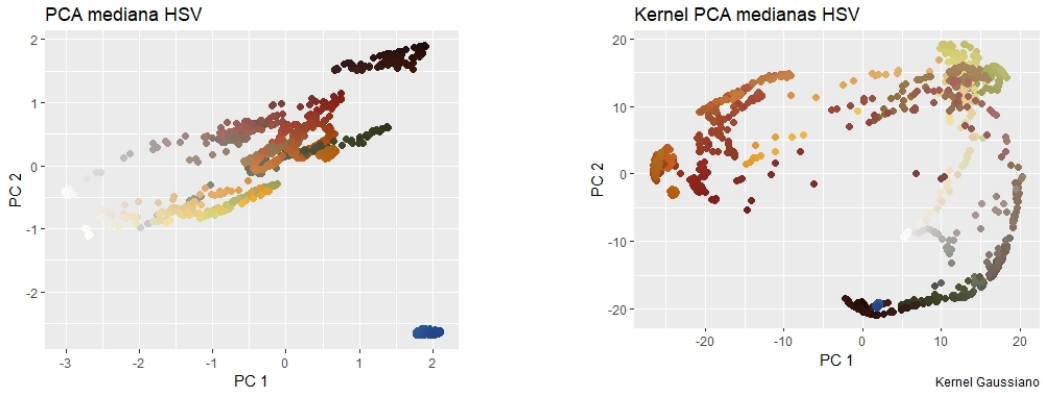
(a) (b) Imagen rotada del espacio HSV

Figure 2.7: Representación de las imágenes en el espacio HSV usando la mediana

Ahora, a la representación HSV, se le aplica PCA y Kernel PCA, este último con un kernel Gaussiano, y con $\sigma = 1$[2]. En la figura 2.8a, se observa que PCA realiza una buena representación, logra identificar grupos informativos, como las cerezas (color tinto), en la parte superior, y los arándanos; no obstante, en la figura 2.8b, al aplicar kernel PCA, se presentan cosas más interesantes, separa las frutas rojas, y naranjas, de los cafés, y verdes;

[1] Se divide entre 360 que es el máximo de esté canal.
[2] En el código se implementan varios valores de sigma, para no saturar el documento se seleccionó el que proporciona patrones más interesantes.

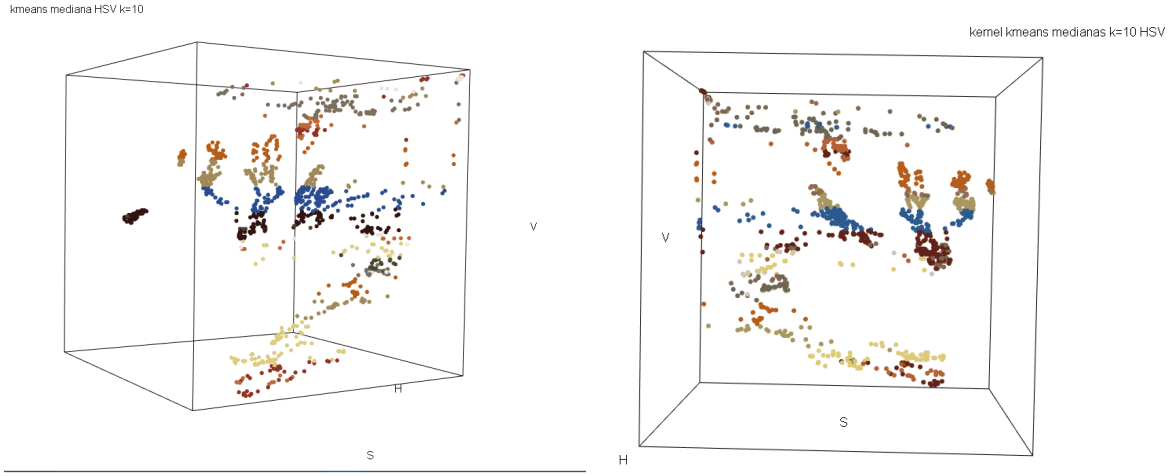
asimismo, los puntos extremos en cada escala de color se observan más nítidos, y tienden a descolorarse para darle paso a otra fruta; i.e., logra decir algo del proceso de maduración.



(a) Primeros dos componentes (b) Kernel PCA Gussiano $\sigma = 1$

Figure 2.8: PCA y Kernel PCA en las medianas de las imagenes en el espacio RGB

Al espacio HSV, se le aplica k-means y kernel k-means, este último con un kernel Gaussiano y $\sigma = 20$. En la figura 2.9, se presenta lo anterior, en el gráfico 2.9a , k-means, y en el 2.9b Kernel k-means. En general ambos separan adecuadamente la mayoría de las frutas por color; no obstante, si observamos la tabla 2.2, se presenta la distribución en la que los métodos clasifican en cada grupo. Bajo esta representación, y con los dos métodos, no se puede decir mucho sobre el proceso de maduración.



(a) K-means k=10 (b) kernel k-means; $\sigma = 20$

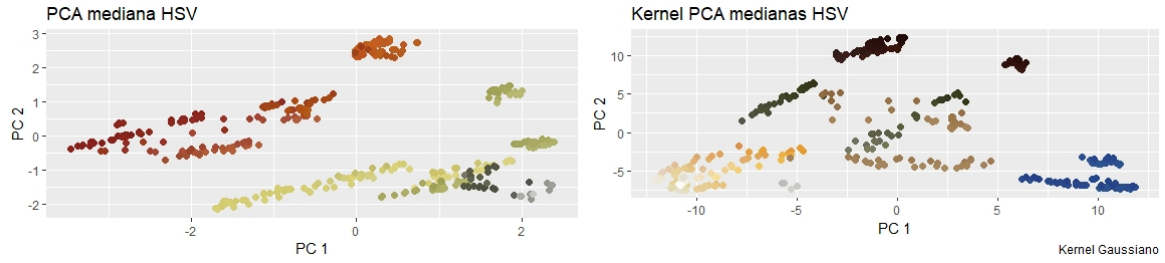
Figure 2.9: K-means y Kernel K-means a las medianas en espacio RGB

Clusters	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$	$k=10$
Kernel Kmeans	182	201	95	71	145	132	164	53	153	104
Kmeans	199	95	100	147	156	70	64	112	243	114

Table 2.2: Distribución de imágenes en clusters

Por último, se nos pide incluir más información sobre cada dimensión del espacio HSV, utilizando la información de los tres cuartiles centrales en cada canal. Lo que se hace es construir tres variables de cana canal H,S,V, con el porcentaje de medianas que represente cada cuartil, quedando con 9 dimensiones.

En la figura 2.10, se presenta la aplicación de PCA y kernel PCA a las nueve dimensiones, lo que se gráfica son los primeros dos componentes. En este caso los colores no representan las frutas, ya que hicimos una partición en cada canal e indexar los colores correspondientes de cada imagen no era factible. Lo que se puede observar es que con más información se separan cinco grupos, para el caso de PCA. Utilizando kernel PCA con un kernel Gaussiano y $\sigma = 25$, no se logra una separación muy adecuada en el centro de las observaciones, pero claramente se observan algunos grupos en los extremos



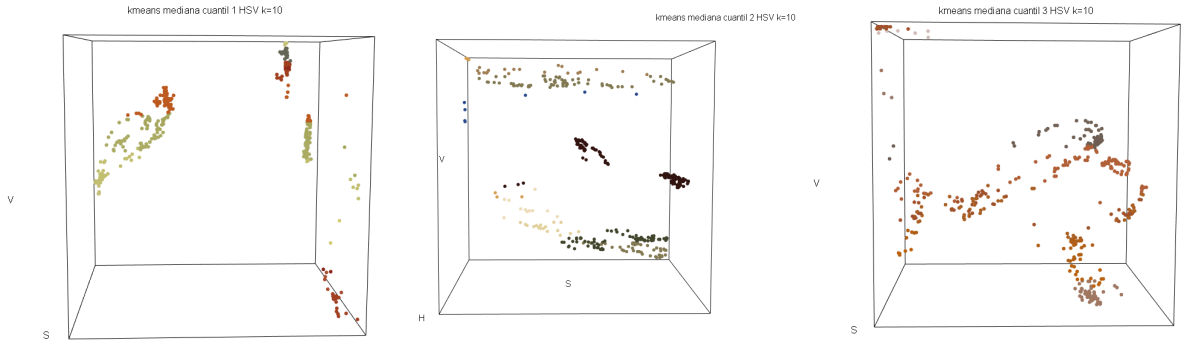
(a) PCA (b) kernel PCA; $\sigma = 25$

Figure 2.10: K-means y Kernel K-means a las medianas en espacio RGB dividido en tres cuantiles

Se le aplica k-means y kernel k-means a las nueve dimensiones, este último con un kernel Gaussiano y $\sigma = 25$. Para realizar la representación y respetar los colores de las observaciones, se presentan tres gráficos - figura 2.11 y 2.12- por separado de los cuartiles de los canales H, S, V[3].

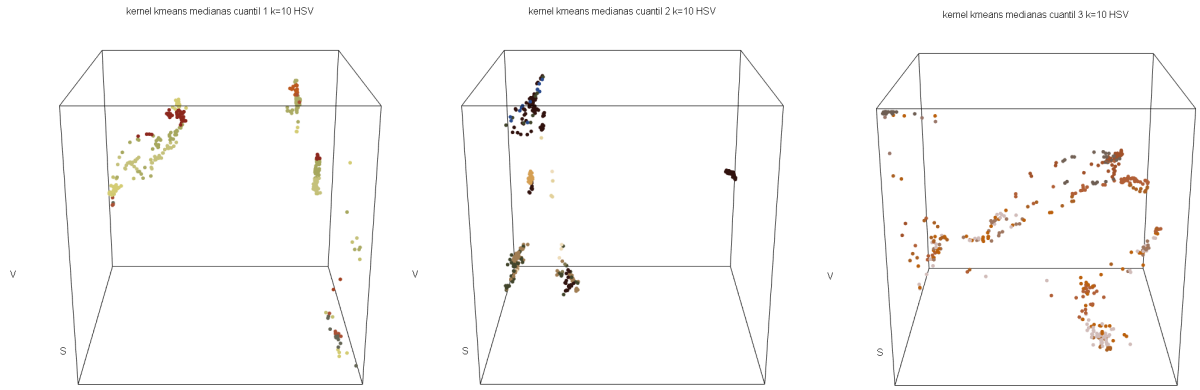
En la figura 2.11, se observa que k-means identifica las frutas en todos los casos; también se logra ver la decoloración de las frutas, en algunos casos. En la figura 2.12, con kernel k-means no se observan patrones interesantes como en el caso anterior.

[3]Nota: no se realiza por separado las técnicas de agrupamiento, solo la visualización se realiza por separado



(a) kmeans cuartil 1 HSV (b) kmeans cuartil 2 HSV (c) kmeans cuartil 3 HSV

Figure 2.11: K-means cuartiles del espacio HSV



(a) Kernel kmeans cuartil 1 HSV (b) Kernel cuartil 2 HSV (c) Kernel cuartil 3 HSV

Figure 2.12: Kernel K-means cuartiles del espacio HSV; kernel Gaussiano, $sigma = 25$, $k = 10$

En conclusión no se nota mejoras significativas, solamente con tres canales y utilizando PCA y kernel PCA, se puede extraer información sobre el tipo de fruta y su proceso de maduración.

3 Problema

3) Ejercicios de puntos extras. Sobre análisis de sentimientos.

a) Realiza PCA en la representación de los textos obtenida con bag of words usando los n términos más frecuentes (decide el valor de n). Realiza el preproceso que consideres necesario en los textos. ¿ Puedes identificar las críticas positivas y negativas en los componentes principales? Si la respuesta es no, explica las razones.

b) Redefine los datos de entrada de bag of words uniendo k documentos de cada categoría, por ejemplo $k = 5$. Repite el inciso anterior. ¿ Qué diferencias notaste? Describe tus hallazgos.

3.1 Solución inciso "a"

Para este ejercicio, se hace uso de la librería **tm** en R. Se utilizan sus funciones para extraer, y limpiar el *corpus* de textos. Se eliminan espacios en blanco, se remueven los números, las letras se homogenizan a minúsculas, se eliminan las *stopwords* y los *stems*, para todo documento.

Al procesar el corpus, se toman los 50 términos más frecuentes y se implementa PCA. En la figura 3.1, se tienen las dos primeras componentes principales; en la figura 3.1a, se realiza el PCA a la observaciones sin estandarizar; en la figura 3.1b, se estandarizan las observaciones previo a PCA. Con esta visualización no se puede identificar las críticas positivas y negativas.

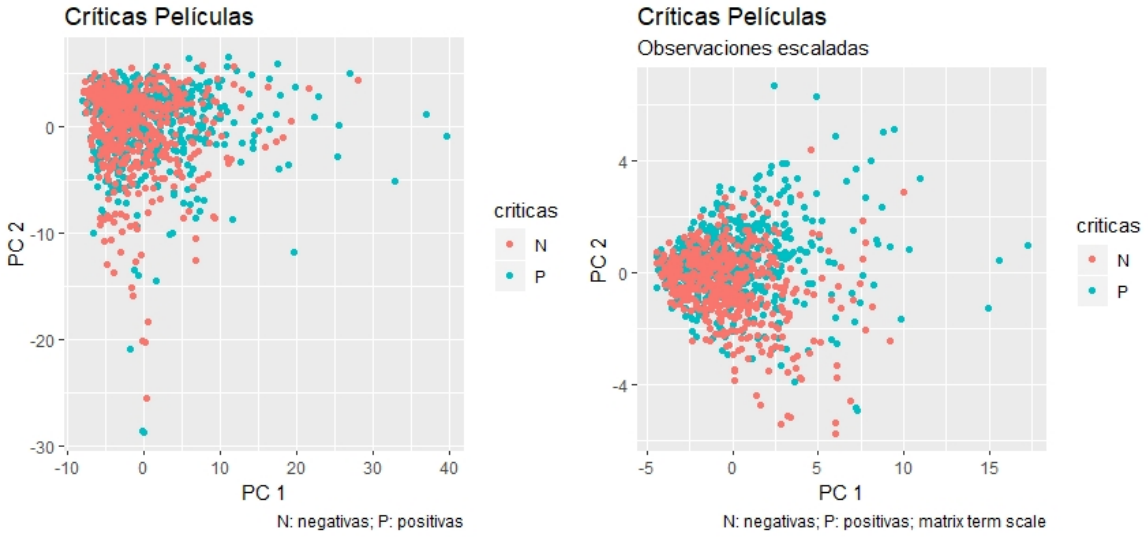


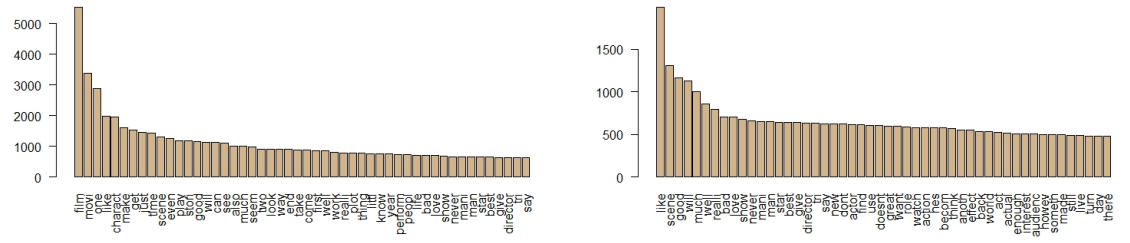
Figure 3.1: Primeras dos componentes de los 50 términos más frecuentes

Algunas de las razones, se deben al estar trabajando con matrices con alto nivel de dispersión[4];

[4]La matriz reportó 99% de dispersión, al remover términos dispersos con la función *removeSparseTerms*, nos queda una matriz con dispersión del 78%

también, tanto críticas negativas, como positivas, tienen términos neutrales que se repiten con mucha frecuencia, sesgando así el resultado.

Se retiran observaciones comunes con el fin de observar mejoras con PCA. En la figura 3.2a, están los 50 términos previos a ser removidos del corpus, en la figura 3.2b, se encuentran los 50 términos más frecuentes una vez removido aquellos comunes.



(a) Antes de retirar observaciones comunes (b) Después de retirar observaciones comunes

Figure 3.2: 50 términos más frecuentes

Se implementa con el nuevo *corpus* PCA, y se observa en la figura 3.3 las primeras dos componentes principales. En la figura 3.3a se encuentra PCA con observaciones no escaladas, y en la figura 3.3b, con las observaciones escaladas. Se puede observar que bajo la visualización de las observaciones estandarizadas, se tiene un ligero patrón donde críticas negativas (rosa) se posicionas por arriba de las críticas positivas (azul) que van hacía abajo.

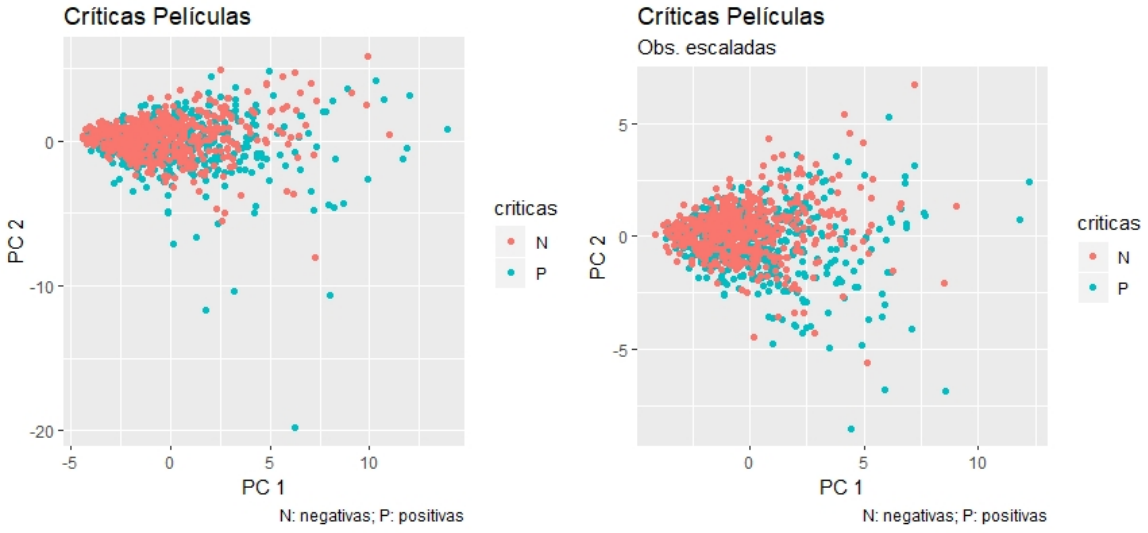


Figure 3.3: Primeras dos componentes de los 50 términos más frecuentes

Se concluye que no se pueden separar del todo las críticas negativas y positivas, ya que la

matriz es muy dispersa y existen términos para las dos críticas que se repiten y no aportan a ningun sentimiento.

3.2 Solución inciso "b"

Se redefinen los datos de entrada del *bag of words* uniendo 5 documentos de cada categoría. Se realiza el pre-proceso, similar al ejercicio anterior, posterior a eso se realiza PCA. En la figura 3.4, se tienen las dos primeras componentes principales, en la figura 3.4a, con los datos sin escalar, y en la figura 3.4b, con los datos escalados. A comparación del ejercicio anterior, se observan grupos que se forman con las críticas positivas y negativas.

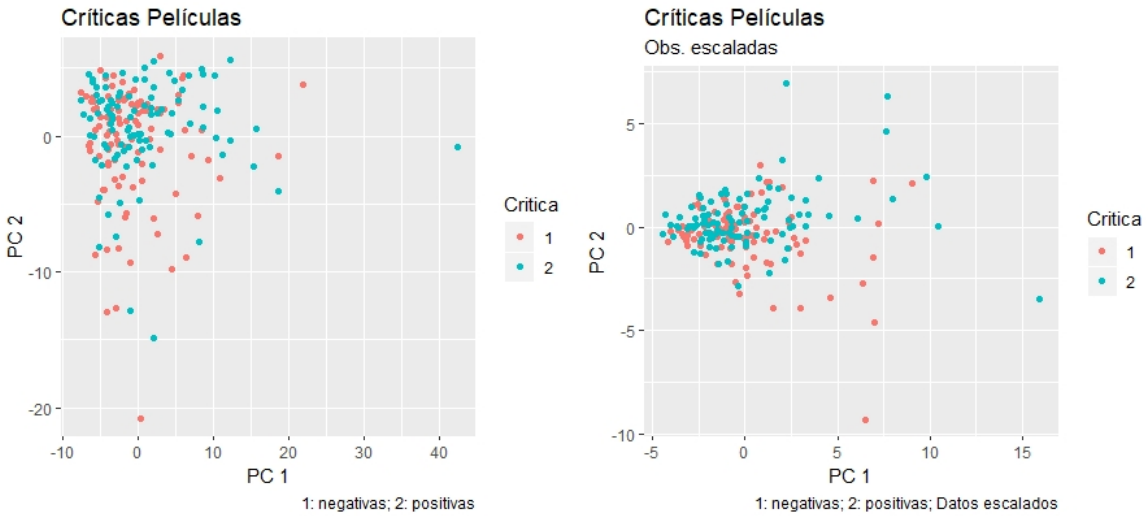


Figure 3.4: Primeras dos componentes de los 50 términos más frecuentes

Al igual que en el ejercicio pasado, se eliminan palabras comunes que sesgan la frecuencia de los términos. En la figura 3.5, se observan los 50 términos más frecuentes una vez eliminado términos comunes.

Figure 3.5: Términos más frecuentes

A estos 50 términos se le aplica PCA, y en la figura 3.6, se presentan las primeras dos componentes. En la figura 3.6a, las primeras componentes con observaciones sin escalar; en la figura 3.6b, las primeras dos componentes con las observaciones escaladas. Se puede ver, que al unir documentos y aplicar PCA, mejora la separación y la visualización de los grupos que realizan críticas positivas y negativas.

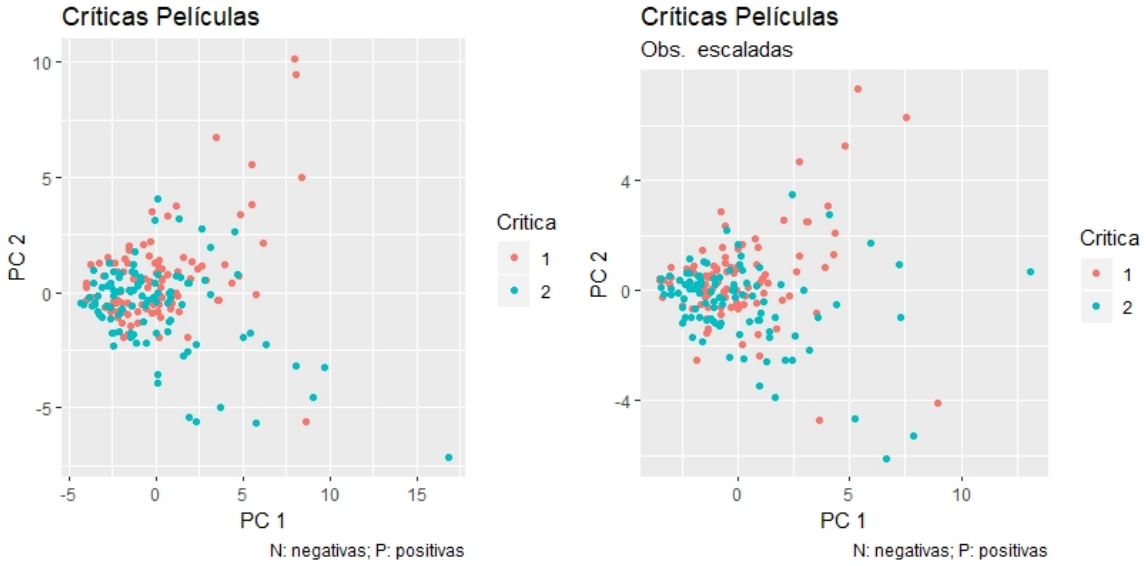


Figure 3.6: Primeras dos componentes de los 50 términos más frecuentes

4 Problema

4) Ejercicios de puntos extras. Sobre análisis de similaridad.

a) Realiza Kernel PCA con string kernels para analizar la similaridad de los textos **description x** y **description y**. Describe los patrones o grupos significativos que logres identificar en los primeros componentes principales. Prueba con diferentes tipos de string kernels y reporta cuál te proporciona el "mejor" resultado.

b) Considera los textos de **description x** como tu conjunto de entrenamiento y realiza Kernel PCA como en el inciso anterior. Selecciona algunos datos de la columna **description y** como datos de "prueba" y verifica qué texto del conjunto de entrenamiento es el más similar a cada uno usando la distancia mínima en el espacio de componentes principales. ¿ Coinciden con los del archivo original?

4.1 Solución inciso "a"

En este ejercicio cada archivo al contener más de 2 mil descripciones hace la visualización complicada; es por esto, que para cuestiones practicas, se trabaja con un subconjunto de observaciones, con un $n = 20$. Las observaciones son las primeras 20 del archivo **trin stock**, de las columnas **descripción x** y **descripción y**; se le realiza un pre-proceso al introducirlas en una lista y retirar los dígitos, espacios en blanco, **stop words**, **stems** y cambiando letras mayúsculas a minúsculas. Se le aplica kernel PCA con la función **kpca** utilizando distintos string kernel con la función **stringdot**, ambas de la librería **kernlab**. Una vez, aplicado kernel PCA se toman las dos componentes principales y se proyectan las observaciones.

En la figura 4.1, se muestran las dos primeras componentes al realizar kernel PCA con un string kernel *spectrum*, utilizando una longitud en la subcadena de tres, el color rosa representan las descripciones del texto de las acciones **y**, y el azul, las de **x**[5]. Se observan tres grupo; uno en la esquina inferior izquierda, cuyas descripciones son muy parecidas; otro en la esquina superior derecha, donde se confunden en la segunda palabra de las descripción; el tercer grupo se presenta en la esquina superior izquierda, donde se juntan varias descripciones con un buen agrupamiento - en varios casos - en las palabras que se parecen.

[5]Estos colores se respetan para todas las gráficas del ejercicio 4, inciso "a".

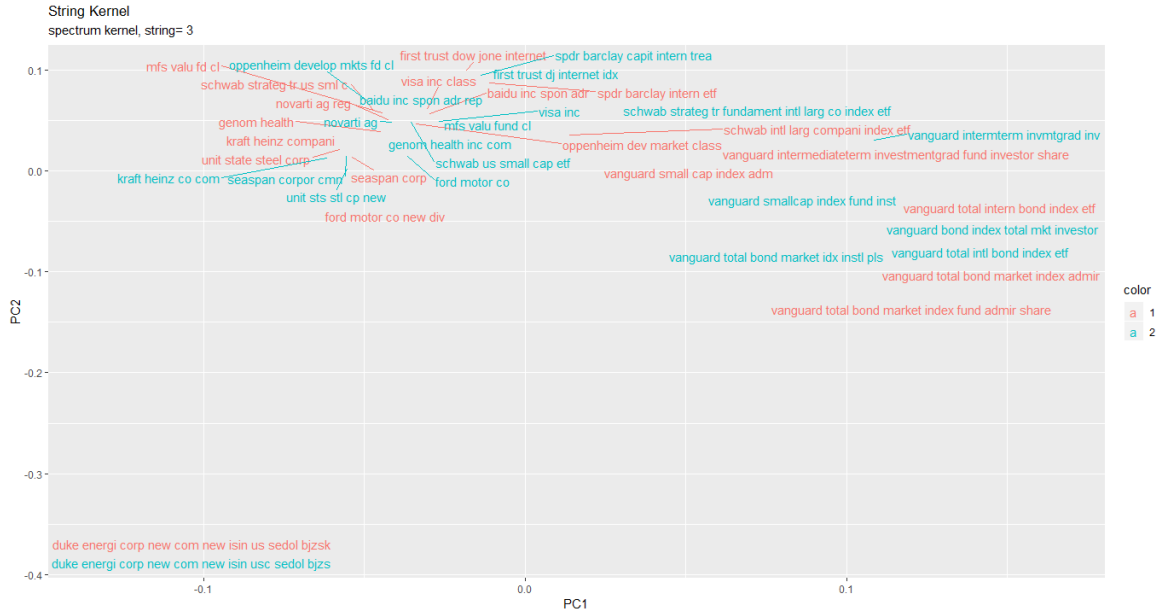


Figure 4.1: Primeras dos componentes; *spectrum* kernel con longitud de subcadena = 3

En la figura 4.2 se presentan las dos primeras componentes, al utilizar kernel PCA con un string kernel *bounrange*, y tamaño de subcadena 3. Se observa, un grupo en la parte superior, siendo el mismo que se diferencia con el sting kernel *spectrum*; asimismo, en la parte inferior izquierda, textos que inician con la misma palabra se confunden; en la parte inferior derecha, se acumulan descripciones, pero aquellas que tienden a estar juntas, se encuentran alejadas dentro del mismo grupo.

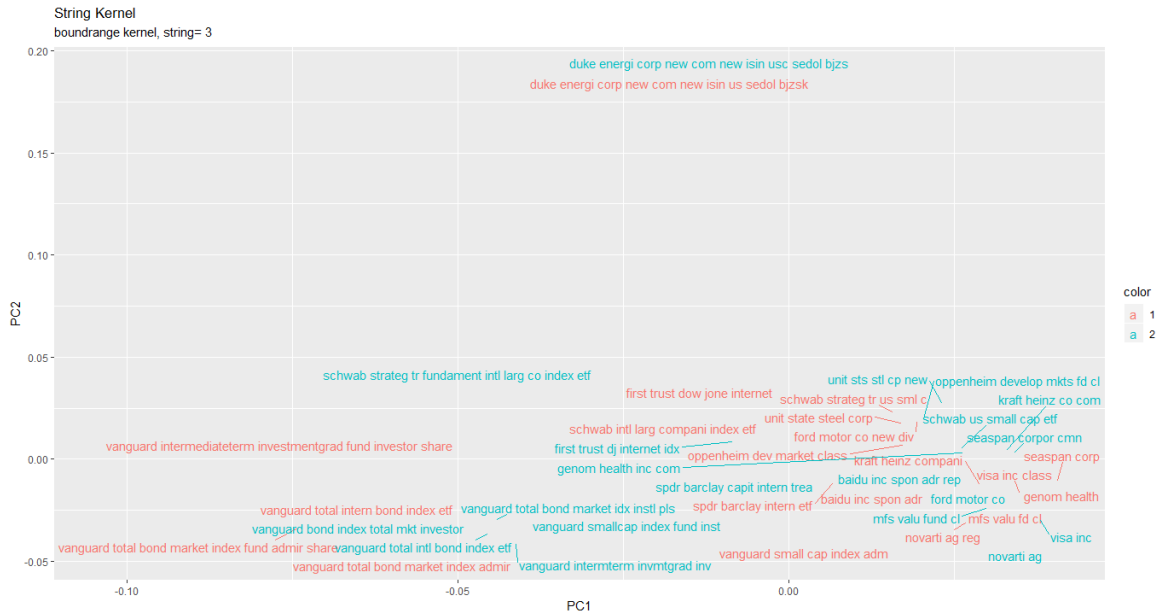


Figure 4.2: Primeras dos componentes; *boundrange* kernel con longitud de subcadena = 3

En la figura 4.3 se presentan las dos primeras componentes, al utilizar kernel PCA con un string kernel *constant*, y tamaño de subcadena 3. Con el el string kernel constante los resultados no son los mejores, ya que la mayoría de las veces agrupa descripciones que no son similares, y en grupos lejanos.

Figure 4.3: Primeras dos componentes; *constant* kernel con longitud de subcadena = 3

En la figura 4.4 se presentan las dos primeras componentes, al utilizar kernel PCA con un string kernel *exponential*, y tamaño de subcadena 3. En este caso, el string kernel exponencial, detecta el mismo grupo de la parte inferior izquierda; no obstante, en la parte superior agrupa cerca las descripciones similares, pero se equivoca con las descripciones que en su primera y segunda palabra inicien con la misma letra.

Figure 4.4: Primeras dos componentes; *exponential* kernel con longitud de subcadena = 3

Se sabe que si los tickets coinciden la descripciones serán las mismas, es por eso que se toman los tickets para buscar similaridad y mejorar la visualización. En este caso, se toma el mismo subconjunto que el anterior, solo que no se le aplica un pre-proceso, ya que estos "textos" son solo letras del abecedario. Al ser palabras con pocas letras, en todos los casos se utiliza en el string kernel una subcadena de uno[6].

En la figura 4.5, se presentan las primeras dos componentes con el string kernel **spectrum**. En muchos casos agrupa de forma correcta los tickets por letra inicial de palabra; no obstante, si existen tickets que inicien con la misma letra, tiende a confundir, haciendo que tickets distintos los agrupe como similares.

[6]Otra razón es que la función **stringdot** te pide un tamaño no mayor al número de letras que contiene el texto con menor palabras, un ejemplo, es el ticken con el texto "V", que es solo una palabra.

Figure 4.5: Primeras dos componentes; *spectrum* kernel con longitud de subcadena = 1

Como se observa en la figura 4.6, al utilizar un string kernel **boundrange**, sucede casi lo mismo que el espectral, con la única diferencia que los puntos se encuentras más cercanos.

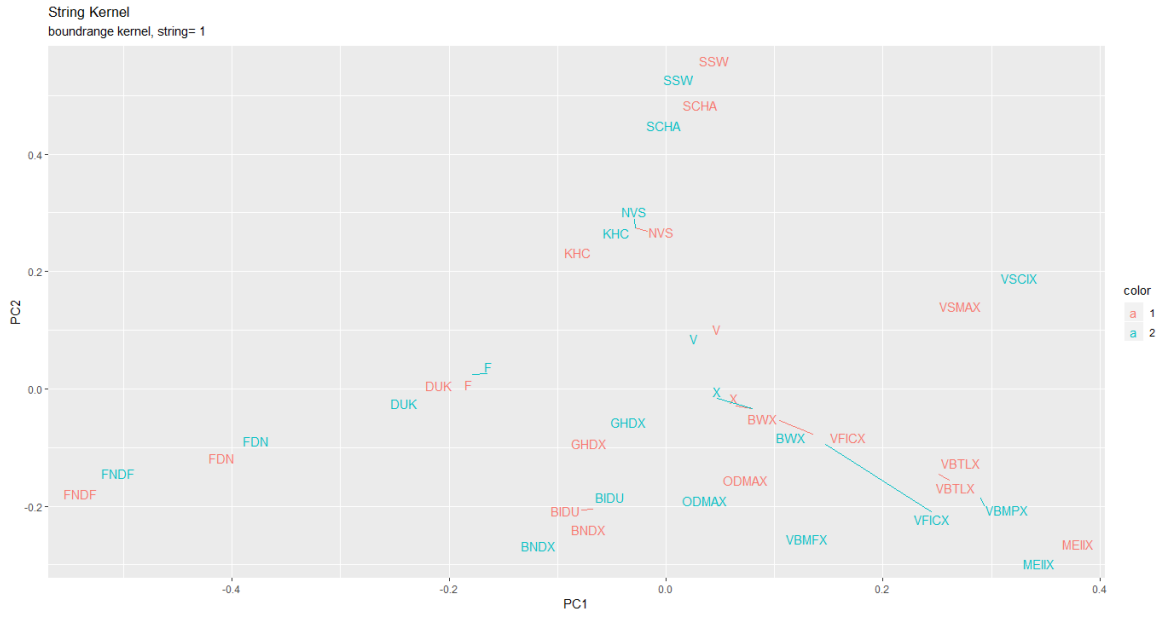


Figure 4.6: Primeras dos componentes; *boundrange* kernel con longitud de subcadena = 1

En la figura 4.7, se observa el caso al utilizar un string kernel **constant**; el cual realiza bien la agrupación de tickets similares, apartando aquellos que no son iguales, en una distancia adecuada de aquellos que inician con su misma letra, que es el caso del texto "VBMFX" de

color azul. Por otro lado, los grupos que forma ya no se distinguen tanto por letra inicial de cada palabra

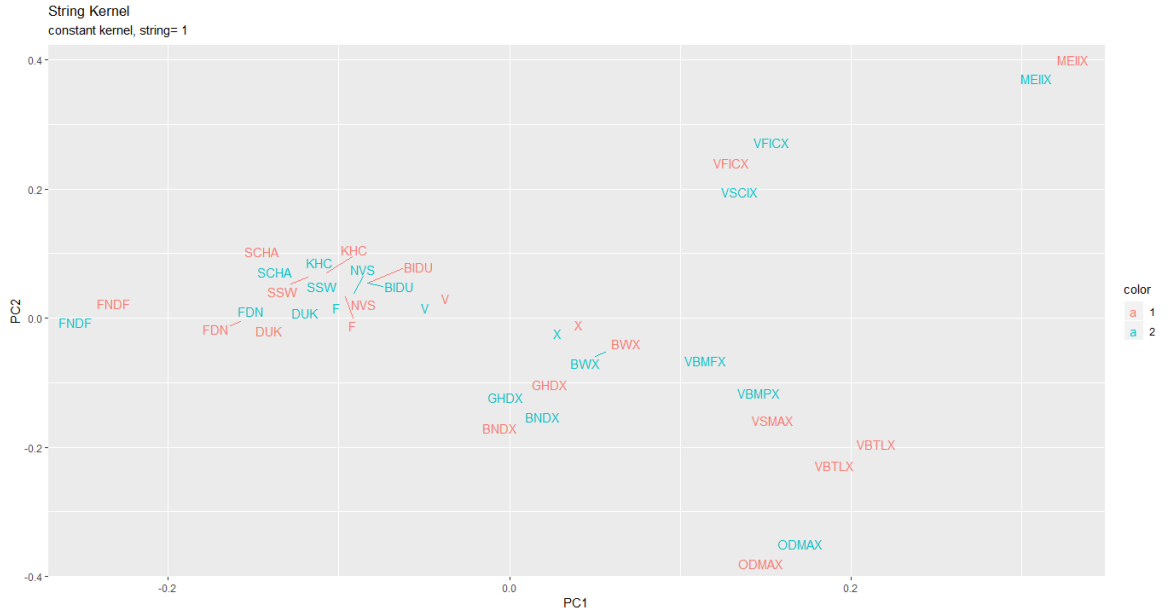


Figure 4.7: Primeras dos componentes; *constant* kernel con longitud de subcadena = 1

Por último, en la figura 4.8, se utiliza un string kernel exponencial, realizando mejor la agrupación de los tickets qie inician con la misma palabra, detectando tickets similares y a la vez separando aquellos que no lo son.

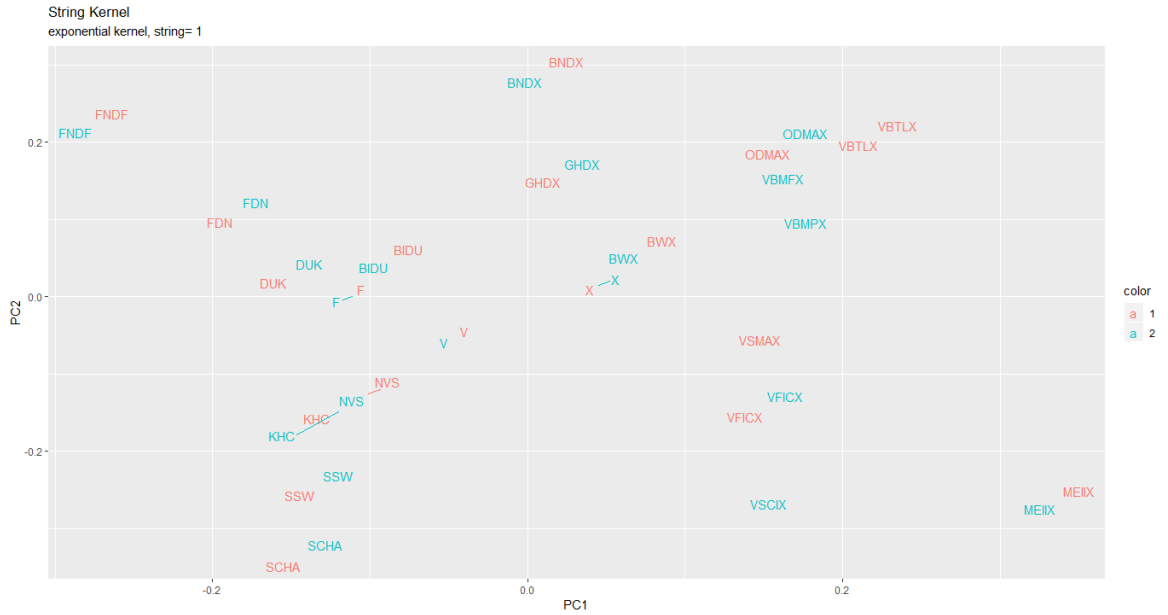


Figure 4.8: Primeras dos componentes; *exponential* kernel con longitud de subcadena = 1

En conclusión, observar los tickets nos lleva a mejores conclusiones para indicar si la descripción de la acción es la misma. Asimismo, un string kernel exponencial, diferencia mejor aquellas descripciones que no son las mismas, y a la vez, agrupa mejor a aquellas que si lo son.

4.2 Solución inciso "b"

Al igual que en el ejercicio anterior, se toma un subconjunto de descripciones para una mejor visualización y demostrar lo que se solicita. Se utilizan las **descripciones x**, se aplica al texto un pre-proceso previo para eliminar; espacios en blanco, números, símbolos, **stems**, **stop words**, y letras mayúsculas a minúsculas; después, se seleccionan las primeras diez descripciones, y se realiza Kernel PCA con distintos string kernels, con un tamaño de subcadena de ocho palabras.

Se debe recalcar que en el archivo **Tarea 4 Ejercicio 4**, se deja al usuario la oportunidad de incrementar el tamaño del subconjunto, con el fin de que pueda corroborar los resultados con otras descripciones.

Se proyectan descripciones del archivo **descripción y**, utilizando las componente obtenidas con Kernel PCA (texto en azul). El primer string kernel que se utiliza es el espectral, con subcadena de tamaño ocho, en la figura 4.9, se observan las proyecciones de las primeras cuatro descripciones de la base de prueba (texto en rojo), vemos en la parte izquierda que se reconocen la descripción similares; no obstante, en la parte inferior derecha, las descripciones al ser distintas, trata de localizarlas en la más parecida; comportamiento similar con las proyecciones en la parte superior derecha.

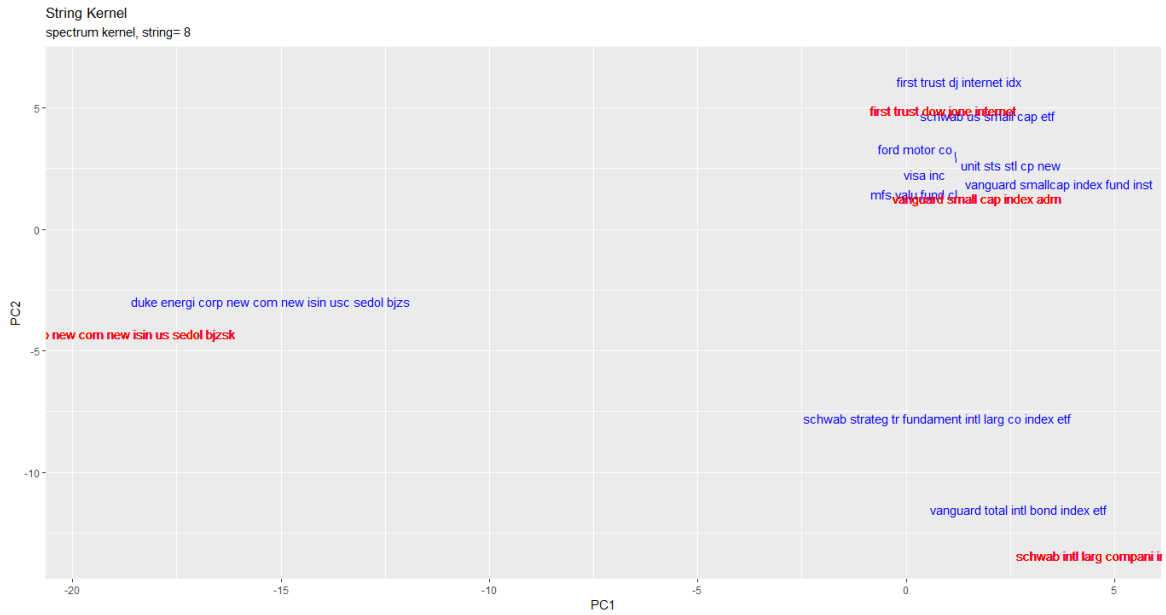


Figure 4.9: Proyección en primeras dos componentes; *spectrum* kernel con longitud de subcadena = 8

En la figura 4.10, se utiliza un string kernel exponencial, se observa que las proyecciones (texto en rojo) se posicionan no tan cerca respecto a la descripción que más se asimila; asimismo, cabe mencionar, que con el string kernel exponencial se tiende a equivocar más que el espectral.

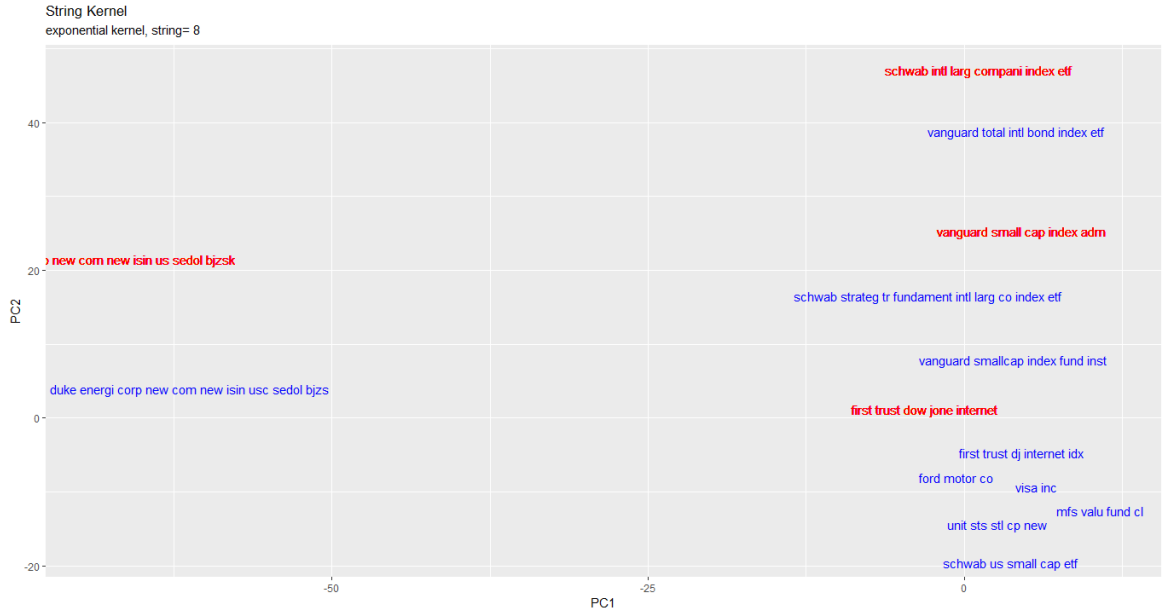


Figure 4.10: Proyección en primeras dos componentes; *exponential* kernel con longitud de subcadena = 8

El string kernel constante se aprecia en la figura 4.11, el desempeño es semejante que el exponencial, tiende a agrupar las proyecciones pero no tan cerca de la descripción que más se le parece.

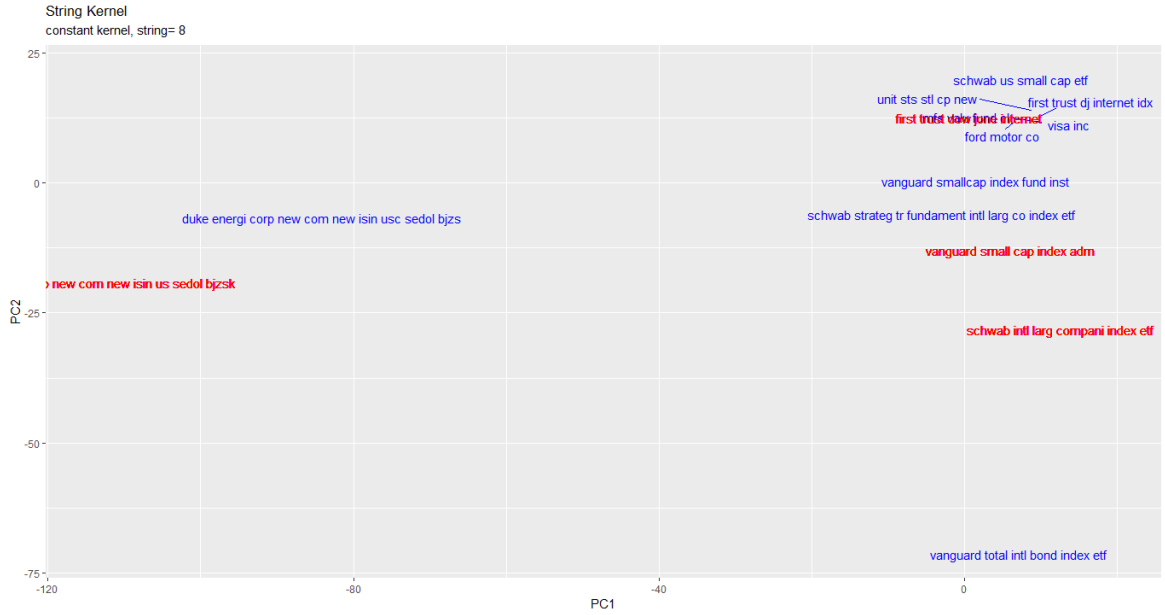


Figure 4.11: Proyección en primeras dos componentes; *constant* kernel con longitud de subcadena = 8

En la figura 4.12, se presentan los resultados con un string kernel **boundrange**, las proyecciones de los datos de prueba en algunos casos se localizan en el grupo correspondiente pero de forma alejada; vemos que la descripción **duke energi corp new isin us sedol bjzsk**, se encuentra cerca de la observación de entrenamiento; así como la descripción **first trust dow jones**; asimismo, **vanguard small cap index**, al ser descripción distinta al de la base de entrenamiento, tiende a buscar aquella descripción con mayor similaridad.

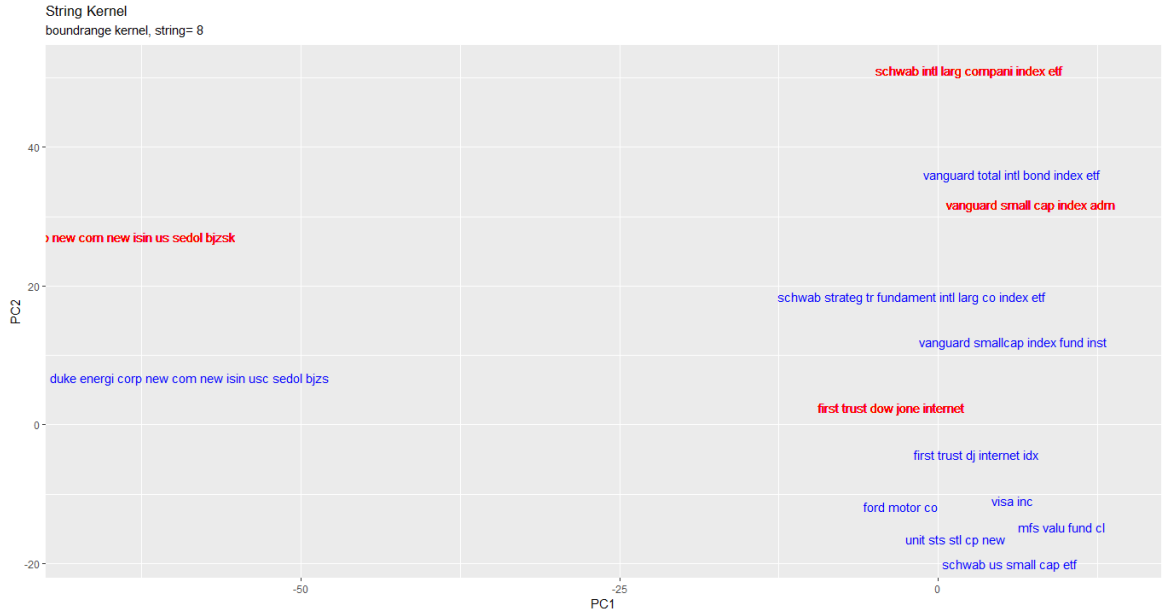


Figure 4.12: Proyección en primeras dos componentes; *boundrange* kernel con longitud de subcadena = 8

En conclusión, en este ejercicio con este subconjunto de datos, las proyecciones de la base de prueba sí coinciden con los archivos originales, y más si se utiliza un string kernel espectral.