Universidad Nacional
Autónoma de México

Inteligencia Artificial

Clustering Jerárquico-Particional

Grupo 3

Nombre:
Barreiro Valdez Alejandro

Práctica 7

Profesor: Dr. Guillermo Gilberto Molero Castillo

31 de marzo de 2022

Introducción

Para esta práctica se generarán dos modelos de clustering, uno particional y uno jerárquico, a partir de los datos que se propongan. A partir de cada uno de los modelos se generarán interpretaciones de las agrupaciones generadas y se analizarán los resultados que se generaron. Con esta práctica se reforzará el conocimiento del clustering y se podrá hacer uso de datos que proponga el alumno.

Objetivos

A partir de un conjunto de datos de características físicas de las ondas que tienen diferentes canciones de diferentes géneros, obtener grupos de canciones con características similares que determinen el género, a través de clustering jerárquico y particional.

Desarrollo

Se importaron las bibliotecas necesarias para el desarrollo de la práctica.

```
import pandas as pd                 # Para la manipulación y análisis de datos
import numpy as np                  # Para crear vectores y matrices n dimensionales
import matplotlib.pyplot as plt     # Para la generación de gráficas a partir de los datos
import seaborn as sns               # Para la visualización de datos basado en matplotlib
%matplotlib inline
```

Se leen los datos a utilizar. Se tienen mil datos y 28 columnas de variables.

```
Gmusica = pd.read_csv('musica.csv')
Gmusica
```

	filename	chroma_stft	rmse	spectral_centroid	spectral_bandwidth	rolloff	zero_crossing_rate	mfcc1	mfcc2	mfcc3	...
0	blues.00000.wav	0.349943	0.130225	1784.420446	2002.650192	3806.485316	0.083066	-113.596748	121.557297	-19.158825	...
1	blues.00001.wav	0.340983	0.095918	1529.835316	2038.617579	3548.820207	0.056044	-207.556793	124.006721	8.930560	...
2	blues.00002.wav	0.363603	0.175573	1552.481958	1747.165985	3040.514948	0.076301	-90.754387	140.459900	-29.109968	...
3	blues.00003.wav	0.404779	0.141191	1070.119953	1596.333948	2185.028454	0.033309	-199.431152	150.099213	5.647593	...
4	blues.00004.wav	0.308590	0.091563	1835.494603	1748.362448	3580.945013	0.101500	-160.266037	126.198807	-35.605450	...
...	...	...	...	...	...	...	...	...	...	...	...
995	rock.00095.wav	0.351991	0.079469	2008.581132	2106.617024	4254.215942	0.089267	-153.632309	109.857262	-23.085709	...
996	rock.00096.wav	0.398653	0.076452	2006.051164	2068.327905	4147.374921	0.097659	-142.424210	116.219780	-32.177074	...
997	rock.00097.wav	0.432103	0.081617	2077.190361	1926.989678	4030.767293	0.121824	-125.031311	115.194977	-47.993507	...
998	rock.00098.wav	0.362349	0.083888	1398.672358	1818.148469	3014.740104	0.048731	-224.972305	123.656891	-9.754534	...
999	rock.00099.wav	0.358195	0.054461	1609.442919	1797.065098	3246.280370	0.076290	-235.189331	123.888542	-22.536184	...

1000 rows × 28 columns

Se agrupan a los datos por género para obtener cuántas canciones de cada género se tienen en el conjunto de datos. Se tienen 100 canciones de 10 géneros distintos.

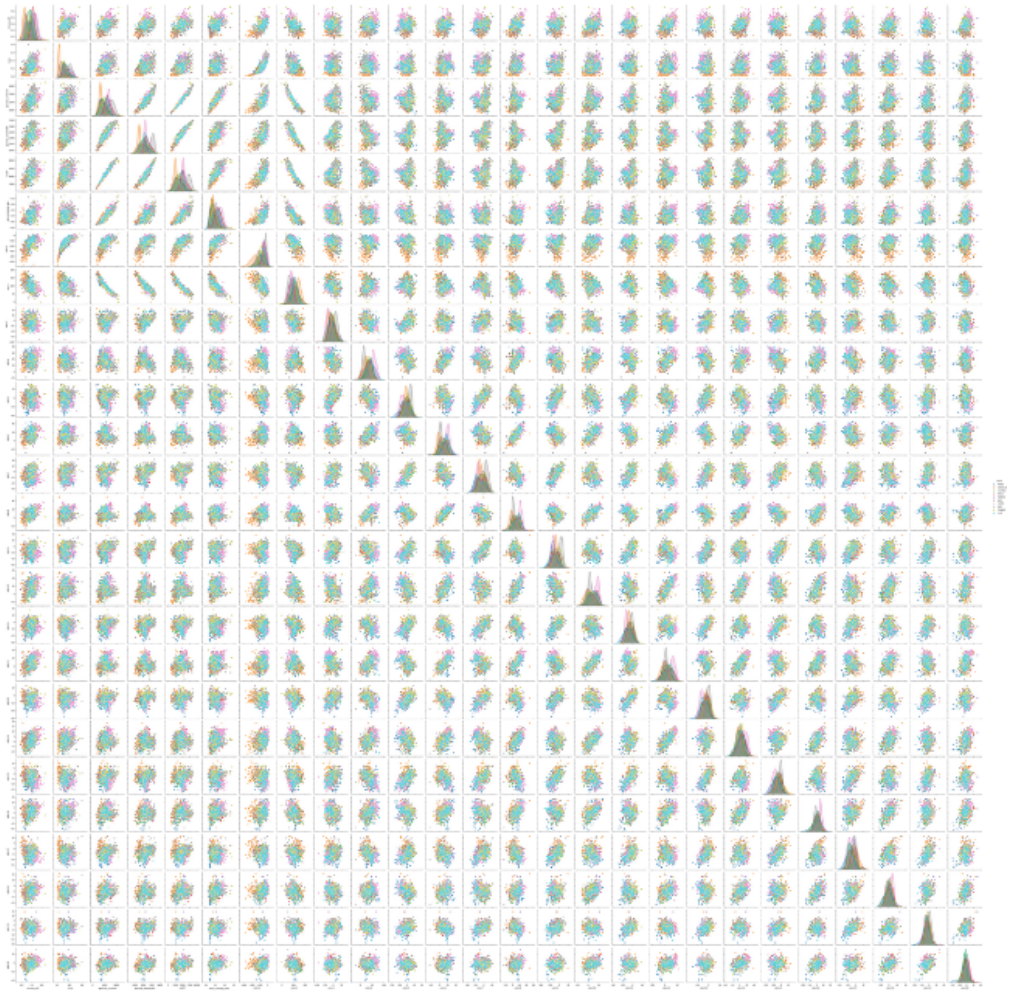
Además, se verificó si existían datos nulos para poder eliminar dichas filas, pero no existe ningún dato nulo.

```
print(Gmusica.groupby('label').size())
```

```
label
blues        100
classical    100
country      100
disco        100
hiphop       100
jazz         100
metal        100
pop          100
reggae       100
rock         100
dtype: int64
```

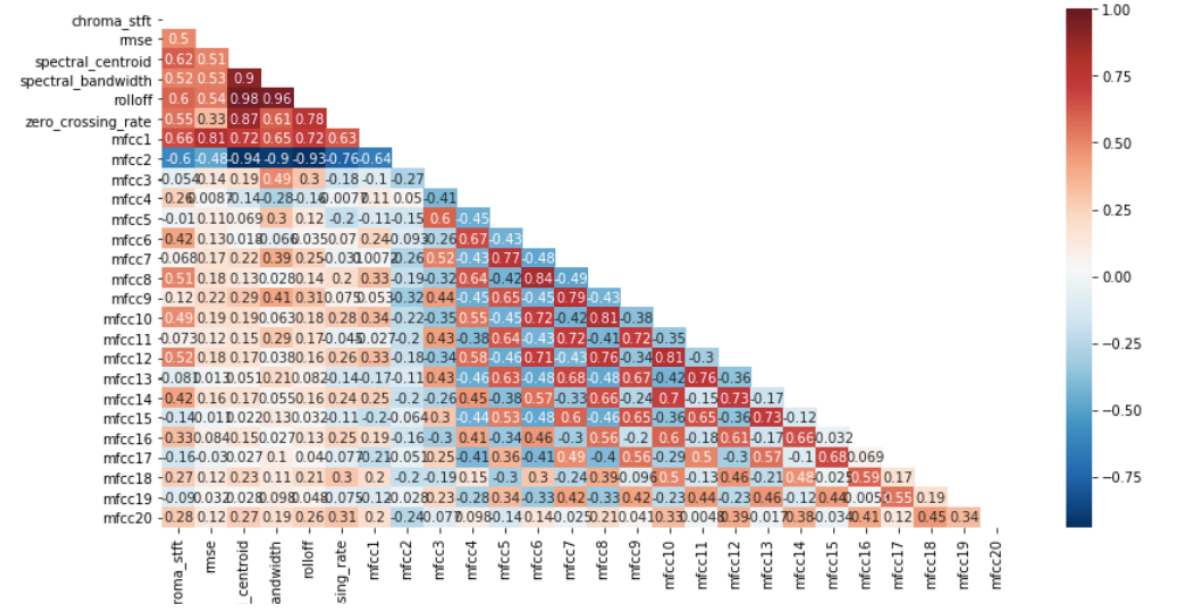
Se generó una evaluación visual de la correlación entre las diferentes variables de los datos pero al ser muchas variables esta evaluación visual no fue tan útil. Se utilizará el valor de las correlaciones para generar una reducción de dimensionalidad.

```
sns.pairplot(Gmusica, hue='label')
plt.show()
```



Se generó la matriz de correlaciones de los datos utilizando el método de Pearson y a partir de ella se generó un mapa de calor donde se puede observar qué relaciones tienen alta correlación entre ellas. Con este pasó se generó la selección de variables eliminando aquellas que tienen alta correlación entre ellas.

```
plt.figure(figsize=(14,7))
MatrizInf = np.triu(CorrMusica)
sns.heatmap(CorrMusica, cmap='RdBu_r', annot=True, mask=MatrizInf)
plt.show()
```



Las variables seleccionadas fueron: chroma_stft, rmse, spectral_bandwidth, rolloff, mfcc3 al mfcc5, mfcc7 y mfcc9 al mfcc20. El DataFrame resultante es el siguiente:

```
MatrizVariables = np.array(Gmusica[['chroma_stft', 'rmse', 'spectral_bandwidth', 'rolloff', 'mfcc3','mfcc4', 'mfcc5'
pd.DataFrame(MatrizVariables)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.349943	0.130225	2002.650192	3806.485316	-19.158825	42.351028	-6.376458	-13.697912	-12.285267	10.980492	-8.324325	8.810669	-3.667368
1	0.340983	0.095918	2038.617579	3548.820207	8.930560	35.874683	2.916037	-8.554704	-10.103617	11.903744	-5.560388	5.376803	-2.239120
2	0.363603	0.175573	1747.165985	3040.514948	-29.109968	31.689013	-13.987036	-13.649585	-11.780589	9.706443	-13.123111	5.789265	-8.905224
3	0.404779	0.141191	1596.333948	2185.028454	5.647593	26.871927	1.754462	-4.830883	-0.757742	8.149012	-3.196314	6.087677	-2.476421
4	0.308590	0.091563	1748.362448	3580.945013	-35.605450	22.153301	-32.489269	-23.357929	-11.805832	1.206805	-13.083821	-2.806384	-6.934123

Se estandarizaron los datos para que ninguna tuviera mayor valor que otros. Esta matriz estandarizada será utilizada para los dos métodos de clustering. La matriz de variables estandarizadas es la siguiente.

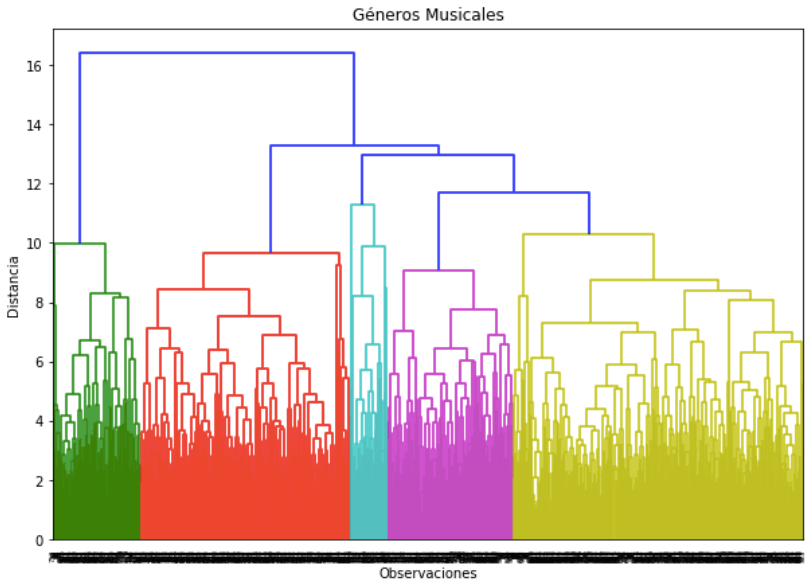
```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
estandarizar = StandardScaler()                          # Se instancia el objeto StandardScaler o MinMaxScale
MEstandarizada = estandarizar.fit_transform(MatrizVariables)   # Sescalan los datos
pd.DataFrame(MEstandarizada)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	
0	-0.351748	-0.010723	-0.456037	-0.486166	-0.472090	0.363653	-0.428098	-0.862290	-0.638827	0.409659	-0.337931	0.646276	0.183186	0.792921	-(
1	-0.461466	-0.533266	-0.387668	-0.649869	0.823297	-0.025115	0.332557	-0.344691	-0.375344	0.526026	0.067600	0.134824	0.414751	0.486404	-(
2	-0.184484	0.680012	-0.941680	-0.972811	-0.931003	-0.276376	-1.051076	-0.857427	-0.577876	0.249079	-1.042019	0.196257	-0.666035	-0.572254	-
3	0.319725	0.156317	-1.228392	-1.516328	0.671898	-0.565540	0.237474	0.030065	0.753379	0.052781	0.414461	0.240704	0.376277	-0.570291	(
4	-0.858124	-0.599604	-0.939405	-0.629459	-1.230552	-0.848794	-2.565611	-1.834449	-0.580925	-0.822213	-1.036254	-1.084009	-0.346457	-1.865428	-
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
995	-0.326676	-0.783817	-0.258409	-0.201709	-0.653184	1.365514	-0.519290	-1.142400	-1.837342	0.967008	-0.933029	1.173558	-1.217452	0.798770	-
996	0.244707	-0.829779	-0.331192	-0.269588	-1.072447	0.770165	-0.591722	-1.326594	-1.824224	0.386954	-1.021387	1.428926	-1.880314	0.473547	-(
997	0.654318	-0.751107	-0.599858	-0.343673	-1.801848	0.992966	-1.009652	-0.733559	-1.372710	0.300362	-1.471299	0.895204	-2.132789	-0.113928	-
998	-0.199838	-0.716514	-0.806751	-0.989186	-0.038396	1.218706	0.948719	-0.515322	-0.150098	0.638514	-0.437174	0.478329	-0.855171	-0.406761	-(
999	-0.250702	-1.164739	-0.846828	-0.842082	-0.627842	0.997827	-0.023738	-0.194236	-0.697159	1.131102	0.452001	0.833364	-1.657366	0.265414	-(

1000 rows × 20 columns

Se creó el clustering jerárquico para saber el número de agrupaciones que utilizará este algoritmo.

```
#Se importan las bibliotecas de clustering jerárquico para crear el árbol
import scipy.cluster.hierarchy as shc
from sklearn.cluster import AgglomerativeClustering
plt.figure(figsize=(10, 7))
plt.title("Géneros Musicales")
plt.xlabel('Observaciones')
plt.ylabel('Distancia')
Arbol = shc.dendrogram(shc.linkage(MEstandarizada, method='complete', metric='euclidean'))
#plt.axhline(y=7, color='orange', linestyle='--')
#Probar con otras medciones de distancia (euclidean, chebyshev, cityblock)
```



A partir de este modelo se obtuvo que se tenían 5 grupos y se crearon las etiquetas para estos cinco grupos. Dichas etiquetas se añadieron al conjunto de datos

originales con el nombre de la columna de clusterH. El conjunto de datos resultante es el siguiente.

```
Gmusica['clusterH'] = MJerarquico.labels_
Gmusica
```

ssing_rate	mfcc1	mfcc2	mfcc3	...	mfcc13	mfcc14	mfcc15	mfcc16	mfcc17	mfcc18	mfcc19	mfcc20	label	clusterH
0.083066	-113.596748	121.557297	-19.158825	...	-3.667368	5.751691	-5.162763	0.750948	-1.691938	-0.409953	-2.300209	1.219929	blues	1
0.056044	-207.556793	124.006721	8.930560	...	-2.239120	4.216963	-6.012273	0.936110	-0.716537	0.293876	-0.287431	0.531573	blues	1
0.076301	-90.754387	140.459900	-29.109968	...	-8.905224	-1.083720	-9.218359	2.455806	-7.726901	-1.815723	-3.433434	-2.226821	blues	1
0.033309	-199.431152	150.099213	5.647593	...	-2.476421	-1.073890	-2.874778	0.780977	-3.316932	0.637982	-0.619690	-3.408233	blues	4
0.101500	-160.266037	126.198807	-35.605450	...	-6.934123	-7.558618	-9.173553	-4.512165	-5.453538	-0.924161	-4.409333	-11.703781	blues	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0.089267	-153.632309	109.857262	-23.085709	...	-12.306271	5.780973	-10.279924	1.791489	-13.304210	2.473193	-6.717574	-1.189238	rock	1
0.097659	-142.424210	116.219780	-32.177074	...	-16.394691	4.152589	-4.350760	3.736455	-10.845638	1.875218	-7.459579	-2.802677	rock	1
0.121824	-125.031311	115.194977	-47.993507	...	-17.951916	1.211113	-11.534864	1.844774	-12.847901	3.447425	-12.594178	-2.107002	rock	1
0.048731	-224.972305	123.656891	-9.754534	...	-10.071786	-0.255098	-5.276486	-2.816289	-4.416438	1.558265	-5.043121	-3.585596	rock	4
0.076290	-235.189331	123.888542	-22.536184	...	-15.019588	3.110468	-7.127754	1.784515	-7.070405	0.023643	-2.022034	1.158525	rock	1

Se contó la cantidad de elementos que hay en cada uno de los clústeres para el tamaño de cada clúster.

```
#Cantidad de elementos en los clusters
Gmusica.groupby(['clusterH'])['clusterH'].count()
```

```
clusterH
0     50
1    386
2    280
3    117
4    167
Name: clusterH, dtype: int64
```

Para la interpretación de cada uno de los datos se contó de qué genero realmente pertenece para saber qué género sí se está agrupando bien y cuáles no.

```
Datos del cluster #0:
label
classical    32
country       3
hiphop        4
jazz          8
pop           1
reggae        2
Name: label, dtype: int64
Datos del cluster #1:
label
blues        54
classical     1
country      37
disco        52
hiphop       38
jazz         27
metal        97
reggae       27
rock         53
Name: label, dtype: int64
Datos del cluster #2:
label
country    21
disco      32
hiphop     53
jazz       18
metal       3
pop        86
reggae     45
rock       22
Name: label, dtype: int64
Datos del cluster #3:
label
blues        18
classical     9
country      33
disco        10
hiphop        2
jazz         12
pop           9
reggae       11
rock         13
Name: label, dtype: int64
Datos del cluster #4:
label
blues        28
classical    58
country       6
disco         6
hiphop        3
jazz         35
pop           4
reggae       15
rock         12
Name: label, dtype: int64
```

Para el clúster 0 se tiene un grupo de canciones donde la gran mayoría son de clásica. Algunas canciones de jazz se metieron a esta agrupación, pero en general es una de las mejores agrupaciones, aunque tiene pocos datos. Debe de ser algún sub-género de clásica que sea muy distinguible.

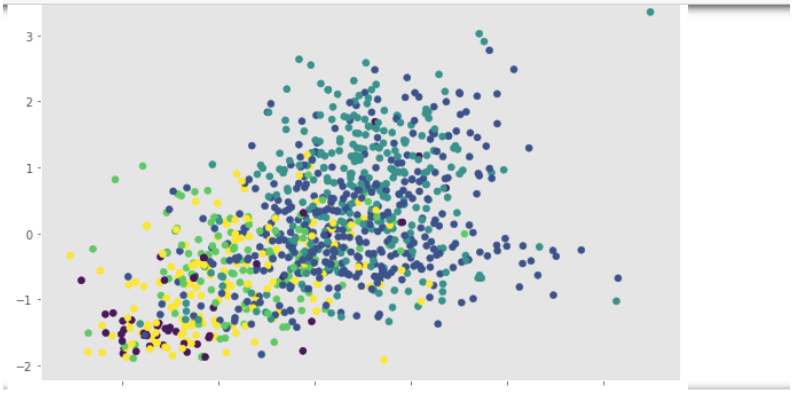
Para el clúster 1 se tiene un grupo de canciones donde se agrupó a la gran mayoría de las canciones. Esta agrupación tiene la peculiaridad de que contiene casi todas las canciones de metal. Esto puede ser un indicador de que esta agrupación capta canciones con mucho ruido y mucho movimiento. Prueba de esto es que también incluye la mitad del disco, la mitad del rock y la mitad del blues.

Para el clúster 2 se tiene un grupo de canciones donde la mayoría son de pop. Este grupo también cuenta con muchas canciones, pero es de los que mejores clasificó los géneros. Cuenta con casi todas las canciones de pop, la mitad de hip-hop y la mitad de reggae.

Para el clúster 3 se tiene un grupo de canciones donde la música country. En general, el country fue un género que se encontró en varias agrupaciones y esto se puede deber a que es muy parecido a otros géneros y es muy variado en su sonido. Para el clúster 4 se tiene un grupo de canciones donde la mayoría son de clásica y de jazz. Este grupo es muy similar al primero y esto podría ser porque representa otro subgénero de la música clásica. El jazz es un género que va ligado con la música clásica por lo que tiene sentido que se encuentre en dos agrupaciones junto a ella.

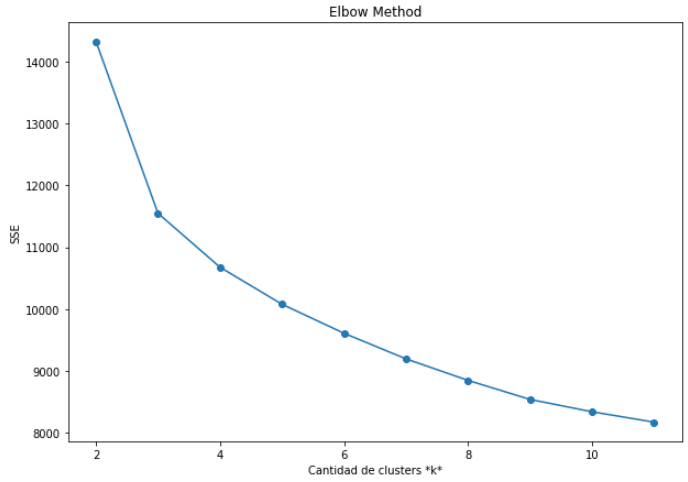
Se generó una gráfica donde se etiquetan los datos con colores. De esta manera no se ve ninguna relación entre las agrupaciones porque el modelo toma en cuenta más dimensiones.

```
plt.figure(figsize=(10, 7))
plt.scatter(MEstandarizada[:,0], MEstandarizada[:,1], c=MJerarquico.labels_)
plt.grid()
plt.show()
```



Para el clustering particional se importan las bibliotecas necesarias para generar el modelo utilizando sklearn. Posteriormente, se calcula el SSE para cada uno de los posibles valores de *k* para graficarlo y generar el método del codo. La gráfica generada es la siguiente.

```
#Se grafica SSE en función de k
plt.figure(figsize=(10, 7))
plt.plot(range(2, 12), SSE, marker='o')
plt.xlabel('Cantidad de clusters *k*')
plt.ylabel('SSE')
plt.title('Elbow Method')
plt.show()
```

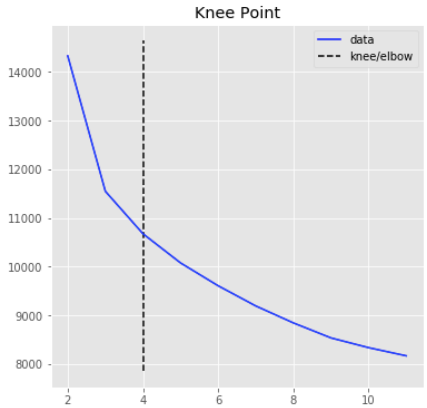


Utilizando la biblioteca como en prácticas anteriores se encontró el punto de inflexión para determinar el número de clústeres. Para esta variación de la agrupación se tienen 4 clústeres.

```
from kneed import KneeLocator
kl = KneeLocator(range(2, 12), SSE, curve="convex", direction="decreasing")
kl.elbow
```

```
4
```

```
plt.style.use('ggplot')
kl.plot_knee()
```



Se crean las etiquetas del modelo y se les adjuntan a los datos originales. La tabla resultante es la siguiente.

```
Gmusica['clusterP'] = MParticional.labels_
Gmusica
```

rossing_rate	mfcc1	mfcc2	mfcc3	...	mfcc14	mfcc15	mfcc16	mfcc17	mfcc18	mfcc19	mfcc20	label	clusterH	clusterP
0.083066	-113.596748	121.557297	-19.158825	...	5.751691	-5.162763	0.750948	-1.691938	-0.409953	-2.300209	1.219929	blues	1	1
0.056044	-207.556793	124.006721	8.930560	...	4.216963	-6.012273	0.936110	-0.716537	0.293876	-0.287431	0.531573	blues	1	0
0.076301	-90.754387	140.459900	-29.109968	...	-1.083720	-9.218359	2.455806	-7.726901	-1.815723	-3.433434	-2.226821	blues	1	1
0.033309	-199.431152	150.099213	5.647593	...	-1.073890	-2.874778	0.780977	-3.316932	0.637982	-0.619690	-3.408233	blues	4	0
0.101500	-160.266037	126.198807	-35.605450	...	-7.558618	-9.173553	-4.512165	-5.453538	-0.924161	-4.409333	-11.703781	blues	3	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0.089267	-153.632309	109.857262	-23.085709	...	5.780973	-10.279924	1.791489	-13.304210	2.473193	-6.717574	-1.189238	rock	1	1
0.097659	-142.424210	116.219780	-32.177074	...	4.152589	-4.350760	3.736455	-10.845638	1.875218	-7.459579	-2.802677	rock	1	1
0.121824	-125.031311	115.194977	-47.993507	...	1.211113	-11.534864	1.844774	-12.847901	3.447425	-12.594178	-2.107002	rock	1	1
0.048731	-224.972305	123.656891	-9.754534	...	-0.255098	-5.276486	-2.816289	-4.416438	1.558265	-5.043121	-3.585596	rock	4	3
0.076290	-235.189331	123.888542	-22.536184	...	3.110468	-7.127754	1.784515	-7.070405	0.023643	-2.022034	1.158525	rock	1	1

Se cuentan las canciones que pertenecen a cada cluster.

```
Gmusica.groupby(['clusterP'])['clusterP'].count()
```

```
clusterP
0    180
1    356
2    303
3    161
Name: clusterP, dtype: int64
```

Para el clustering particional también se genera una interpretación a partir de el número de canciones que realmente pertenecen a cierto género. Los resultados para cada agrupación son los siguientes.

```
Datos del cluster #0:
label
blues        26
classical    71
country      16
disco         1
hiphop        2
jazz         34
metal         1
pop           2
reggae       24
rock          3
Name: label, dtype: int64
Datos del cluster #1:
label
blues        47
country      29
disco        53
hiphop       40
jazz         17
metal        96
reggae       21
rock         53
Name: label, dtype: int64
Datos del cluster #2:
label
country      20
disco        41
hiphop       55
jazz         21
metal         3
pop          93
reggae       43
rock         27
Name: label, dtype: int64
Datos del cluster #3:
label
blues        27
classical    29
country      35
disco         5
hiphop        3
jazz         28
pop           5
reggae       12
rock         17
Name: label, dtype: int64
```

Para el clúster 0 se tiene un grupo de canciones donde se agrupan a casi todas las canciones de música clásica. Esto es una mejora respecto al modelo anterior donde

se tenían dos agrupaciones para la música clásica. En esta agrupación también se encuentra el jazz y otro género que no debería estar relacionado que es el reggae.
Para el clúster 1 se tiene un grupo de canciones que se parece mucho al segundo clúster del modelo anterior. Es la agrupación con más datos y contiene a casi todas las canciones de metal del conjunto de datos. Además, contiene a otras canciones de otros géneros que también llegan a ser ruidosos y enérgicos como el rock, el blues, el hip-hop y el disco.
Para el clúster 2 se tiene un grupo de canciones donde el pop es el género dominante ya que casi todas las canciones del género se encuentran en esta agrupación. Es el segundo clúster con más datos y también contiene canciones de hip-hop, disco y reggae.
Para el clúster 3 se tiene un grupo de canciones donde se tiene la menor cantidad de canciones y no se tiene un género que tengan muchas más canciones que otro. Los géneros que se encuentran en esta agrupación son blues, clásica, country y jazz.

Conclusiones

Con estos dos modelos se generaron agrupaciones que permitieron clasificar a diversas canciones por sus características físicas. A partir de dichas clasificaciones se pudo observar cómo existe cierta relación para algunos géneros como el pop, el metal y la música clásica, pero para los demás no existía una agrupación o similitud muy grande. Los resultados fueron buenos y se generaron buenos modelos con mucho sentido. Si se hubieran generado 10 clústeres o muchos más se tendrían modelos que caen en el sobre ajuste, se busca un modelo general que tenga sentido en lo que haga. Aunque ambos modelos tuvieron resultados similares se tuvo uno con menor cantidad de agrupaciones. Gracias a esta práctica se reforzaron ambos algoritmos, aquello que se necesita para llevarlos a cabo y se pudo utilizar un algoritmo de aprendizaje automático para generar resultados a partir de datos propuestos por el alumno.