Facultad de Ciencias, UNAM
Reconocimiento de Patrones y Aprendizaje Automatizado
GTZAN Dataset - Music Genre Classification

Rubí Rojas Tania Michelle

05 de junio de 2021

1. Introducción

Hoy en día, la música es algo indispensable para nosotros. Utilizamos plataformas como `Spotify`, `Youtube` o `Apple Music` para escuchar canciones de nuestro agrado, además de que podemos encontrar listas de reproducción adecuadas a nuestras preferencias o algunas otras que contienen canciones de acuerdo a su género musical. Éstas últimas son generadas automáticamente por cada plataforma, en lugar de tener a alguien más que las clasifique por ellos. Pero, ¿cómo lo hacen? ¿cómo logran automatizar este proceso? A lo largo de este trabajo lograremos ver una forma para lograr esto.

2. Objetivo

`"Dada una canción, queremos saber a qué genero musical pertenece"`

3. Métodos

3.1. Conjunto de Datos

El conjunto de noticias que utilizaremos se obtuvo del sitio web *Kaggle*

`https://www.kaggle.com/andradaolteanu/gtzan-dataset-music-genre-classification`

el cual contiene dos archivos y dos carpetas:

1. **features_30_sec.csv**

   Contiene 1000 entradas (10 por cada género) con 60 atributos que son características basadas en el timbre, las cuales miden la similitud entre canciones y proporcionan información semántica sobre la señal. Entre estos atributos se encuentran:

   - **filename**: el nombre del archivo.
   - **length**: el tamaño del archivo.
   - **spectral_bandwidth**: el ancho de banda espectral.
   - **spectral_centroid**: el centroide espectral.
   - **rms**: el nivel promedio de una onda en el espectro.
   - **label**: el género musical al cual pertenece.

   Las características brindadas en este archivo corresponden a canciones cuya duración es de 30 segundos.

2. **features_3_sec.csv**

   Mismos atributos que el archivo anterior, con la diferencia de que corresponden a canciones cuya duración es de 3 segundos (por lo que incrementa 10 veces el número de entradas).

3. **genres_original**

   Corresponden a los audios de las canciones utilizadas en el conjunto de datos.

4. **images_original**

   Corresponden al espectograma de Mel de cada una de las canciones utilizadas en el conjunto de datos.

3.2. Preprocesamiento de Datos

Tenemos 10 géneros musicales: blues, clásica, country, disco, hiphop, jazz, metal, pop, reggae, rock.
Los datos contenidos en el archivo **features_3_sec.csv** se ven de la siguiente forma:

filename	length	chroma_stft_mean	chroma_stft_var	rms_mean	rms_var	spectral_centroid_mean	spectral_centroid_var	spectral_bandwidth_mean
blues.00000.wav	661794	0.350088	0.088757	0.130228	0.002827	1784.165850	1.297741e+05	2002.449060
blues.00001.wav	661794	0.340914	0.094980	0.095948	0.002373	1530.176679	3.758501e+05	2039.036516
blues.00002.wav	661794	0.363637	0.085275	0.175570	0.002746	1552.811865	1.564676e+05	1747.702312
blues.00003.wav	661794	0.404785	0.093999	0.141093	0.006346	1070.106615	1.843559e+05	1596.412872
blues.00004.wav	661794	0.308526	0.087841	0.091529	0.002303	1835.004266	3.433999e+05	1748.172116
blues.00005.wav	661794	0.302456	0.087532	0.103494	0.003981	1831.993940	1.030482e+06	1729.653287
blues.00006.wav	661794	0.291328	0.093981	0.141874	0.008803	1459.366472	4.378594e+05	1389.009131
blues.00007.wav	661794	0.307955	0.092903	0.131822	0.005531	1451.667066	4.495682e+05	1577.270941
blues.00008.wav	661794	0.408879	0.086512	0.142416	0.001507	1719.368948	1.632828e+05	2031.740381
blues.00009.wav	661794	0.273950	0.092316	0.081314	0.004347	1817.150863	2.982361e+05	1973.773306

Figura 1: Dataset con las primeras 10 entradas

Dividimos el conjunto de datos en dos:

- X: corresponde a los primeros 59 atributos (excepto el atributo `filename`).
- y: corresponde al atributo 60, el cual es la etiqueta del género musical.

Normalizamos nuestro conjunto X usando `StandardScaler`. Ésta clase estándariza los datos eliminando la media y escalando los datos de forma que su varianza sea igual a 1. Ahora bien, la variable de salida es un valor de `string`, por lo que debemos convertirlos en valores enteros entre 0 y 9. Esto lo podemos lograr usando la clase `LabelEncoder`, pues ésta modelará la codificación requerida y creará una nueva variable de salida. De esta forma obtenemos que:

- $0 \rightarrow$ blues $\quad 1 \rightarrow$ classical
- $2 \rightarrow$ country $\quad 3 \rightarrow$ disco
- $4 \rightarrow$ hiphop $\quad 5 \rightarrow$ jazz
- $6 \rightarrow$ metal $\quad 7 \rightarrow$ pop
- $8 \rightarrow$ reggae $\quad 9 \rightarrow$ rock

Finalmente, dividimos nuestro conjunto de datos con un split del $80 - 20$ para entrenamiento y prueba.

3.3. Plan de Acción

Para darle una solución a nuestro objetivo, utilizaremos dos *modelos* de clasificación diferentes: SVC (Support Vector Classifier) y PCA (principal Component Analysis) con SVC.

3.3.1. SVC (Support Vector Classifier)

3.3.1.1. Implementación

Creamos nuestro modelo SVC (usando un kernel lineal) y lo entrenamos. Luego, ya podemos comenzar a realizar nuestras predicciones.

```
1         svclassifier = SVC (kernel = 'linear')
2         svclassifier . fit ( X_train, y_train)
3         y_pred = svclassifier.predict(X_test)
4
```

donde obtenemos un 98 % de precisión en el conjunto de entrenamiento y 76 % en el de prueba.

3.3.2. PCA (principal Component Analysis) con SVC

3.3.2.1. Obtención del número de componentes

Usaremos PCA para obtener la lista de atributos que tienen mayor varianza (mayor poder explicativo). Éstos serán las componentes principales.

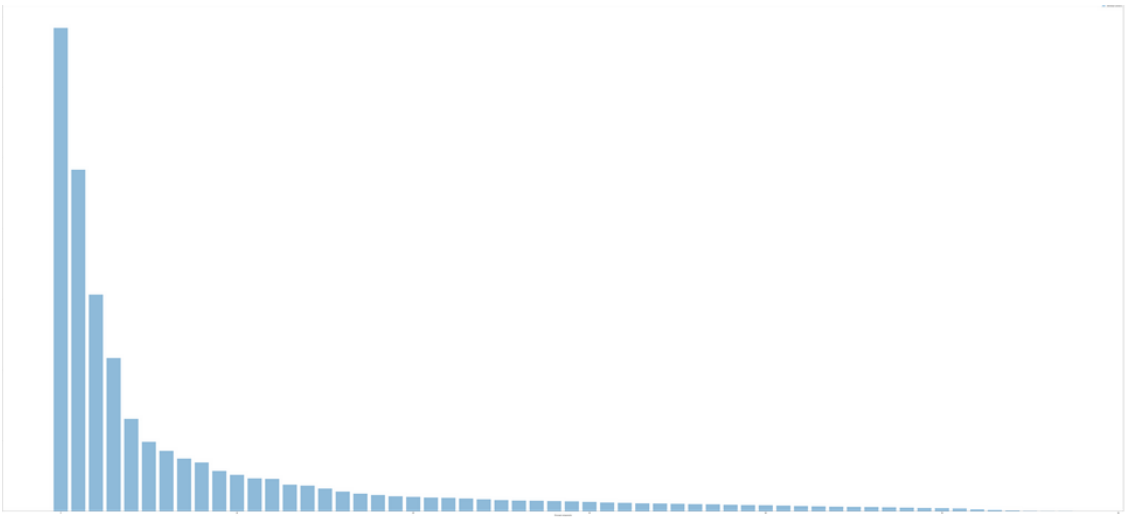

Figura 2: Variance ratio vs Principal components

Con esta gráfica podemos observar que con un número de componentes entre 40 y 45 se explica la mayor varianza, pero para tener un número en concreto realizamos además una gráfica

`Cumulative variance vs The number of components needed to explain variance`

la cual se encuentra en el notebook `proyecto-pca`, pero no se incluyó por ser demasiado grande. Gracias a esta gráfica sabemos que con 43 componentes obtenemos una varianza explicada del 98 %.

3.3.2.2. PCA/SVC

Creamos un modelo SVC exactamente igual que en la sección anterior, sólo que los datos de X serán aquellos obtenidos en PCA.

Así, con este modelo obtenemos una precisión de 96 % en el conjunto de entrenamiento y 74 % en el conjunto de prueba.

4. Resultados

Partiéndo del hecho de que las precisiones obtenidas fueron

- Para SVC: 76 %
- Para PCA/SVC: 74 %

compararemos los reportes de clasificación de ambos algoritmos.

4.1. Reportes de clasificación

Los reportes de clasificación muestran las principales métricas de clasificación, incluidas la precisión y la recuperación, la puntuación (media armónica de la precisión y la recuperación) y el soporte (número de observaciones de esa clase en el conjunto de entrenamiento).

	precision	recall	f1-score	support
0	0.65	0.68	0.67	19
1	0.94	0.89	0.92	19
2	0.60	0.86	0.71	21
3	0.67	0.71	0.69	17
4	0.75	0.95	0.84	19
5	0.76	0.65	0.70	20
6	0.96	0.85	0.90	26
7	0.85	0.77	0.81	22
8	0.89	0.53	0.67	15
9	0.67	0.64	0.65	22
accuracy			0.76	200
macro avg	0.77	0.75	0.75	200
weighted avg	0.78	0.76	0.76	200

(a) SVC

	precision	recall	f1-score	support
0	0.67	0.82	0.73	22
1	1.00	0.93	0.97	15
2	0.64	0.70	0.67	20
3	0.56	0.62	0.59	16
4	0.68	0.59	0.63	22
5	0.82	0.93	0.87	15
6	0.82	0.78	0.80	23
7	0.86	0.75	0.80	24
8	0.68	0.62	0.65	21
9	0.81	0.77	0.79	22
accuracy			0.74	200
macro avg	0.75	0.75	0.75	200
weighted avg	0.75	0.74	0.75	200

(b) PCA/SVC

Figura 3: Reportes de clasificación

Podemos notar que algunos géneros los clasificó mejor un modelo que otro.

En particular, PCA/SVC logró un mayor *balance* en cuanto a las precisiones al momento de realizar la clasificación.

4.2. Predicciones

Ahora bien, veamos cómo funciona cada uno de nuestros modelos al momento de realizar la clasificación.

	Actual	Predicted
0	4	4
1	9	2
2	5	5
3	2	2
4	2	0
5	3	3
6	1	1
7	3	9
8	8	4
9	9	3
10	9	9
11	5	0
12	8	7
13	7	2
14	5	1

(a) SVC

	Actual	Predicted
0	7	3
1	7	7
2	8	8
3	2	2
4	5	5
5	2	2
6	4	4
7	9	2
8	7	7
9	6	6
10	3	3
11	0	0
12	0	5
13	6	6
14	4	4

(b) PCA/SVC

Para mayor simplicidad, sólo mostramos los prirmeros 15 resultados de las predicciones. Como observamos, en esta primera muestra, ambos se comportan más o menos igual con respecto a las predicciones realizadas.

5. Conclusiones

Ambos modelos de clasificación me parecen bastante buenos. La precisión que se obtuvo en cada uno de ellos es más o menos la que esperaba, y la verdad estoy muy satisfecha con éstos. Me agradó el hecho de que PCA funcionara bien en este conjunto de datos, además de que al quitarle entre 25 y 15 componentes ($95 - 98$ de varianza explicada), logramos obtener resultados considerablemente buenos al momento de la clasificación. Al final me quedé con 43 componentes, pues así obtenía un valor de precisión que me agradaba más.

Mi principal problema al momento de elaborar el proyecto fue que tuve errores al momento de usar PCA. En un inicio creía que lo estaba haciéndo bien (tenía 20 componentes principales), pero después de investigar un poco más, encontré que estaba calculando mal el número de componentes, y después de realizar las correcciones pertinentes, logré que PCA funcionara adecuadamente (y esto se vio reflejado muchísimo en la precisión del nuevo modelo, pues pasar de 20 a 43 componentes es un gran cambio).

Respecto al proyecto en general, siento que la elección del tema me ayudó mucho a comprender y reforzar algunos otros aspectos. Un `dataset` de música me amenizó la realización del proyecto, pues era algo que me llamaba mucho la atención y la clasificación de música es algo del diario en las plataformas digitales. Creo que el paso siguiente es realizar un sistema recomendador de música utilizándo redes neuronales, el cual es un proyecto que haré en mi tiempo libre.

Referencias

[1] SVC lineal
https://unipython.com/svc-lineal-machine-learning/

[2] Implementing SVM and Kernel SVM with Python's Scikit-Learn
https://stackabuse.com/implementing-svm-and-kernel-svm-with-pythons-scikit-learn/

[3] Dimensionality Reduction in Python with Scikit-Learn
https://stackabuse.com/dimensionality-reduction-in-python-with-scikit-learn/