Recuperación de información y recomendaciones en la web

Curso 2021

Sistema de recomendaciones de Spotify

Grupo 12

Martin Vigliarolo - 5.065.182-9
Alejandro Silva - 4.939.755-7
Mauricio Xavier - 4.965.678-9

Indice

Introducción

Sabido es que en los últimos años han surgido una serie de productos y servicios que se han convertido en moneda corriente para todos los usuarios de internet en la actualidad, y si se habla de sectores específicos, los servicios para la reproducción de contenidos multimedia vía streaming se han colocado en las primeras posiciones respecto a popularidad. Si hablamos ahora de reproducción de música por streaming, sin lugar a dudas la plataforma Spotify es referencia absoluta. Esto es porque dicho servicio permite a cualquier usuario de todo el mundo acceder en cualquier momento a prácticamente cualquier canción que desee escuchar, de manera rápida y sencilla. La plataforma además de resolver la administración de millones de datos, incluyendo canciones, artistas, álbumes, playlists y varios tipos más, cuenta con una característica que de hecho la hace destacarse sobre su competencia, y esta es su asombroso sistema de recomendaciones, el cual es implementado a través diversos algoritmos de machine learning, los cuales son entrenados constantemente para desempeñar no solo cada vez un mejor trabajo, sino mantener a los usuarios de la plataforma enganchados a la misma, recomendando constantemente canciones nuevas de su interés, usando criterios como géneros y los gustos de sus amigos.

En el presente informe se presenta inicialmente el contexto del problema que se quiere abordar y la solución que se quiere llevar a cabo. Luego se especifica como se ha desarrollado la solución propuesta. Finalmente se tiene una sección de conclusiones donde se reflexiona sobre las lecciones aprendidas.

Problema a abordar

En este trabajo se intenta como principal objetivo recrear algunos de los principales algoritmos que conforman el sistema que es utilizado por la plataforma de música Spotify para recomendar música a sus usuarios.

Ese sistema se denomina BaRT (Bandits for Recommendations as Treatments) y fue desarrollado especialmente por Spotify, el cual presenta un motor de recomendaciones basado en diferentes estrategias, pero dentro de los cuales se destacan un modelo de recomendaciones basado en contenido y otro con técnicas de filtrado colaborativo.

Enfoque de la solución

Dado que el sistema de recomendación de Spotify es sumamente amplio y potente, su motor de recomendación está basado en distintas estrategias entre las que destacan modelos de filtrado colaborativo que analizan el comportamiento de los usuarios de forma individual y global y un modelo basado en contenido que permite sugerir a los usuarios nuevas canciones en función de su similitud con el contenido (canciones, artistas, playlists, etc) juzgadas previamente por ellos.

A raíz de ello, el enfoque de la presente solución se centró en abordar cada uno de los casos por separado. En especial, se elaboró un sistema de recomendaciones basado en contenido, en el lenguaje de programación Python, compuesto por diferentes partes, desde la depuración y limpieza de datos obtenidos de una fuente externa para tareas de experimentación, normalización y vectorización como parte de algunas de las diversas tareas de pre-procesamiento de los datos, y otras funcionalidades, con el fin de implementar una función final que sea capaz de generar recomendaciones personalizadas tomando como entrada una playlist creada por el propio usuario y retornando un conjunto de canciones consideradas similares, y por ende, potencialmente del agrado del usuario.

Además, respecto al caso del filtrado colaborativo, la realización es simple ya que las herramientas provistas por las librerías seleccionadas se encargan de la lógica. Se analizó la cercanía entre los usuarios asignando pesos a las canciones más escuchadas por los mismos, para de esta forma ver si había coincidencia con los otros en alguno de sus temas más escuchados o en los géneros de estas canciones.

Solución propuesta

Modelo de recomendación basado en contenido

Para esta estrategia se buscó como objetivo principal obtener un conjunto final de canciones recomendadas a través de distintas funcionalidades implementadas que nos permitieran determinar similaridad entre canciones según sus respectivos géneros asociados.

Para esto se utilizó un notebook de Jupyter como entorno de ejecución para Python sobre el se realizaron distintos experimentos hasta construir paso a paso nuestra solución final, dividiendo la misma en diferentes bloques y funcionalidades, pudiendo observar resultados parciales e ir ajustando detalles de modo de alcanzar resultados satisfactorios.

Principales librerías Python utilizadas

Pandas y **NumPy**: para la mayor parte de manipulación y análisis de datos.

Sklearn: Como principal herramienta para las tareas de machine learning del tipo análisis predictivo de los datos.

Spotipy: Librería liviana para trabajar sobre la API de Spotify, conectándonos a la misma a través de esta y obteniendo distintos datos musicales de la plataforma y datos relacionados a usuarios específicos.

Preparación de los datos

Como primera tarea para llevar a cabo una solución se buscó un conjunto de datos de Spotify suficientemente amplio que sea de utilidad. A través de la plataforma Kaggle en la que se encuentran miles de datasets de uso público para fines de estudios científicos de datos y aprendizaje automático, se encontró un dataset de Spotify que contenía distinta metadata relacionada a diferentes características sobre miles de canciones contenidas en la plataforma.

En particular, del dataset hallado se tomaron dos archivos .csv, ya que cada uno de los sets pertenecientes contenían datos según diferentes enfoques, seleccionando así por un lado un conjunto de datos a nivel de canciones y distintas características asociadas, y por otro un conjunto de datos a nivel de artistas y los respectivos géneros relacionados.

Datos a nivel de canciones:
En este dataset se presentan datos que para un total de 174389 canciones se incluye por cada una: nombre, artistas correspondientes de la misma, un valor alfanumérico a modo de identificador de cada canción, duración, año y fecha de lanzamiento y otros valores numéricos describiendo información sobre ciertas características técnicas sonoras de las canciones como volumen, vivacidad, energía, instrumentalidad, bailabilidad, acústica, entre otras, y además se incluye un valor relacionado a la popularidad.

data.csv

```
#Dataset de canciones y distintos datos relacionados
spotify_data = pd.read_csv('data.csv')
spotify_data.tail(10)
```

	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness
174379	0.79500	['Alessia Cara']	0.429	144720	0.211	0	45XnLMuqf3vRfskEAMUeCH	0.000000
174380	0.04840	['Stephan F', 'YA-YA']	0.693	177148	0.826	0	1Cbf6PLWsL4s51eFepXx6L	0.000012
174381	0.79500	['Alessia Cara']	0.429	144720	0.211	0	4pPFI9jsgulh3wC7Otoyy8	0.000000
174382	0.14100	['BigBankCarti', 'Keyvo400']	0.544	215014	0.407	1	3ASGdyWXeXsXtOIWtm0tv4	0.000000
174383	0.79500	['Alessia Cara']	0.429	144720	0.211	0	52YtxLVUyvtiGPxwwxayHZ	0.000000
174384	0.00917	['DJ Combo', 'Sander-7', 'Tony T']	0.792	147615	0.866	0	46LhBf6TvYjZU2SMvGZAbn	0.000060
174385	0.79500	['Alessia Cara']	0.429	144720	0.211	0	7tue2Wemjd0FZzRtDrQFZd	0.000000
174386	0.80600	['Roger Fly']	0.671	218147	0.589	0	48Qj61hOdYmUCFJbpQ29Ob	0.920000
174387	0.92000	['Taylor Swift']	0.462	244000	0.240	1	1gcyHQpBQ1lfXGdhZmWrHP	0.000000
174388	0.23900	['Roger Fly']	0.677	197710	0.460	0	57tgYkWQTNHVFEt6xDKKZj	0.891000

key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo	valence	year
4	0.196	-11.665	1	A Little More	0	2021-01-22	0.0360	94.710	0.228	2021
1	0.231	-2.669	1	Only Tonight - Radio Edit	0	2020-12-25	0.0762	126.049	0.361	2020
4	0.196	-11.665	1	A Little More	0	2021-01-22	0.0360	94.710	0.228	2021
4	0.253	-12.745	0	LayUp	0	2020-12-31	0.2330	129.750	0.490	2020
4	0.196	-11.665	1	A Little More	0	2021-01-22	0.0360	94.710	0.228	2021
6	0.178	-5.089	0	The One	0	2020-12-25	0.0356	125.972	0.186	2020
4	0.196	-11.665	1	A Little More	0	2021-01-22	0.0360	94.710	0.228	2021
4	0.113	-12.393	0	Together	0	2020-12-09	0.0282	108.058	0.714	2020
0	0.113	-12.077	1	champagne problems	69	2021-01-07	0.0377	171.319	0.320	2021
7	0.215	-12.237	1	Improvisations	0	2020-12-09	0.0258	112.208	0.747	2020

Datos a nivel de artistas y géneros relacionados

Como se puede observar en el conjunto de datos anterior no hay ninguna información relacionada al género de las canciones que pudiera ser útil para el fin de este trabajo. Sin embargo, el dataset utilizado cuenta con otro conjunto de datos que sí contiene datos relacionados a este aspecto. En el archivo data_w_genre.csv se encuentran datos clasificados según varias de las mismas variables del conjunto de datos anterior a nivel de canciones pero con el agregado de contar con una columna “genre” que contiene un conjunto de diferentes géneros asociados a cada artista con canciones incluidas en el dataset.

data_w_genre.csv

```
#Dataset de artistas y generos correspondientesS
data_w_genre = pd.read_csv('data_w_genres.csv')
data_w_genre.sample(10)
```

	artists	acousticness	danceability	duration_ms	energy	...	popularity	key	mode	count	genres
22594	Planxty	0.815000	0.593500	272093.000000	0.430500	...	33.000000	7	1	4	['british folk', 'celtic', 'irish folk', 'uill...
10064	Frédéric Chopin	0.990084	0.339743	252552.787759	0.106923	...	5.351431	1	1	1013	['classical', 'early romantic era', 'polish cl...
22916	Puzzle	0.011700	0.420000	163515.000000	0.544000	...	60.000000	7	1	2	['indie garage rock']
10109	Future Breeze	0.006830	0.663000	221160.000000	0.925000	...	61.000000	9	1	2	['acid trance', 'bubble trance', 'dream trance...
24433	Roy Jones Jr.	0.020665	0.618500	269866.500000	0.745500	...	64.500000	1	1	2	[]
7990	Dutch Swing College Band	0.670000	0.443667	122490.000000	0.704000	...	14.666667	0	1	6	['big band', 'dixieland', 'new orleans jazz', ...
9077	Evan Rachel Wood	0.724000	0.321000	260907.000000	0.429000	...	74.000000	6	1	2	['hollywood']

Para trabajar con los datos en cuestión se realizaron ciertas operaciones sobre ambos datasets para facilitar la manipulación de los mismos aplicando distintas expresiones regulares que permitieran extraer “limpiamente” los datos deseados del tipo artistas y géneros en un mismo formato.

Un inconveniente que se descubrió en el conjunto de datos utilizado fue que se hallaron varias canciones repetidas y por ende con id’s y características asociadas con valores diferentes, como se puede observar en el siguiente ejemplo:

```
spotify_data[spotify_data['name']=='Hello']
```

artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	...	name	popularity	release_date	s
['Lionel Richie']	0.487	251107	0.246	0	0mHyWYXmmCB9iQyK18m3FQ	0.000000	9	0.0983	...	Hello	57	1983-01-01	
['Adele']	0.578	295502	0.430	0	62PaSfnXSMyLshYJrlTuL3	0.000000	5	0.0854	...	Hello	71	2015-11-20	
['Lionel Richie']	0.563	248573	0.262	0	1b16zlZIdL2LIMfDiANwIk	0.000000	9	0.1940	...	Hello	42	1983	
['Ice Cube', 'Dr. Dre', 'MC Ren']	0.903	232533	0.610	1	60uesDF4UyLUs61FgiDkdp	0.000000	10	0.3890	...	Hello	58	2000-02-29	
['Oasis']	0.318	203187	0.952	0	4qYlBtzkmby4r1N7etPnUv	0.015200	9	0.1670	...	Hello	55	1995-10-02	
['Evanescence']	0.416	220360	0.178	0	0aYUqsvZG7bAslrUkd9Z0g	0.000138	11	0.3420	...	Hello	58	2003-03-04	
['KAROL G', 'Ozuna']	0.802	194933	0.839	0	2776pPgCa5KrLVUqNBtjfJ	0.000052	7	0.1170	...	Hello	0	2021-01-22	
['KAROL G', 'Ozuna']	0.802	194933	0.839	0	2IjhKzXJAPBmRN0QlkxKHG	0.000052	7	0.1170	...	Hello	0	2021-01-22	
['KAROL G', 'Ozuna']	0.802	194933	0.839	0	36YVsPZh2byKbodjUDyjTG	0.000052	7	0.1170	...	Hello	0	2021-01-22	

Para esto se decidió como primer paso agregar un nuevo identificador de tipo “artists_song” concatenando en un string los valores correspondientes a los artistas y nombre de las canciones y en consecuencia de ese nuevo id eliminar aquellas canciones duplicadas tomando como criterio de pertencia dejar en el dataset la

canción con fecha de lanzamiento más reciente. En la creación del nuevo id en caso de que hubiera más de un artista en una canción se optó por crear diferentes identificadores por cada artista considerándolo útil dado que cada artista está asociado a diferentes géneros.

artists	...	id	...	release_date	...	year	...	artists_song
['Poe']	...	1UT2yVLkkJyoLVtGbPgHpi	...	1995-09-14	...	1995	...	PoeHello
['Oasis']	...	4qYlBtzkmby4r1N7etPnUv	...	1995-10-02	...	1995	...	OasisHello
['OMFG']	...	6BAnxKyld909yo6Pk1DO3r	...	2014-12-03	...	2014	...	OMFGHello
['Lionel Richie']	...	0mHyWYXmmCB9iQyK18m3FQ	...	1983-01-01	...	1983	...	Lionel RichieHello
['KAROL G', 'Ozuna']	...	2776pPgCa5KrLVUqNBtjfJ	...	2021-01-22	...	2021	...	KAROL GHello
['J. Cole']	...	30Chv2Smlry70YwtmtaKnj	...	2014-12-09	...	2014	...	J. ColeHello
['Ice Cube', 'Dr. Dre', 'MC Ren']	...	60uesDF4UyLUs61FgiDkdp	...	2000-02-29	...	2000	...	Ice CubeHello
['Evanescence']	...	0aYUqsvZG7bAslrUkd9Z0g	...	2003-03-04	...	2003	...	EvanescenceHello
['Adele']	...	62PaSfnXSMyLshYJrlTuL3	...	2015-11-20	...	2015	...	AdeleHello

Por último, como paso final en la preparación de los datos se realizó una depuración de los pasos anteriores en los que se fueron agregando nuevas columnas de artistas y géneros según fue necesario, y sobre todo se logró unificar en un solo conjunto de datos las canciones a partir de sus identificadores y sus respectivos géneros, información de gran utilidad que no se tenía en un principio directamente en alguno de los archivos del dataset usado.

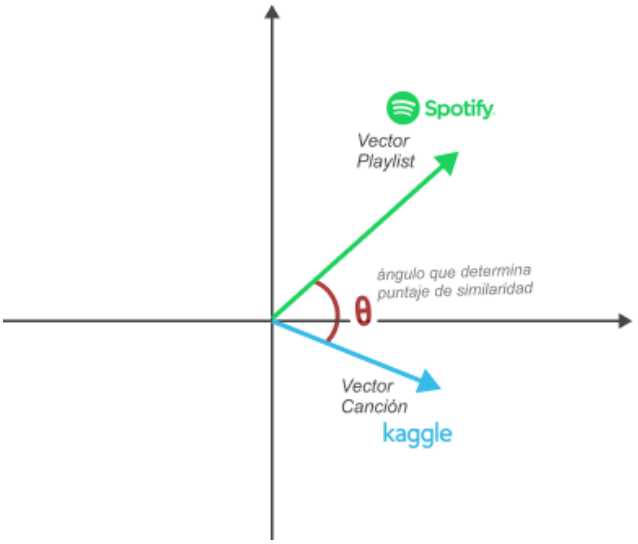
	id	genres_upd	song_genres_list
0	000G1xMMuwxNHmwVsBdtj1	[[candy_pop, dance_rock, new_wave, new_wave_po...	[rock, dance_rock, power_pop, permanent_wave, ...
1	000Npgk5e2SgwGaIsN3ztv	[[classic_bollywood, classic_pakistani_pop, fi...	[sufi, ghazal, filmi, classic_pakistani_pop, c...
2	000ZxLGm7jDlWCHtcXSeBe	[[boogie-woogie, piano_blues, stride]]	[boogie-woogie, stride, piano_blues]
3	000jBcNljWTnyjB4YO7ojf	[[bomba_y_plena]]	[bomba_y_plena]
4	000mGrJNc2GAgQdMESdgEc	[[classical, french_romanticism, late_romantic...	[classical, french_romanticism, late_romantic_...

Implementación algoritmo de recomendaciones

Una vez asegurada la información necesaria respecto a las canciones, artistas y sus géneros se procedió a implementar las distintas funcionalidades y componentes que formaron el algoritmo de recomendación deseado.

La idea principal del algoritmo en cuestión pasa por desarrollar un sistema de puntuación que permita asignar un puntaje a canciones en base a diferentes características de las mismas según los pesos que hayan sido definidos para cada una de las características dadas. Una vez determinados dichos puntajes de las canciones, dada una playlist extraída de la cuenta de un usuario específico creada por el mismo, en función de dichos puntajes de cada una de las canciones contenidas en la playlist se buscó resumir todos esos valores en un vector final para la playlist sumando los valores de todas las canciones correspondientes a un mismo tipo de característica y obteniendo una coordenada por cada apartado.

Tras establecer un mecanismo de puntuación sobre las canciones, en base a playlists creadas por usuarios y sus respectivos vectores generados, se creó una función que retorna efectivamente una playlist de canciones recomendadas que considerara similares.



Para determinar dicha similaridad se aplicó el algoritmo de **Similitud de Coseno**, calculando un puntaje en función del ángulo entre el vector de la playlist del usuario y cada una de las demás canciones del dataset que no se encuentran en la playlist. Este cálculo retorna 1 como resultado si ambos vectores evaluados están dirigidos a un mismo lugar y por ende su ángulo es 0°, lo cual representaría la mayor similaridad posible, mientras que por contrapartida cuanto más alejados estén dichos vectores mayor será el ángulo obtenido, menor el puntaje obtenido y así una menor similaridad. Por lo tanto, evidentemente se incluirán en nuestra playlist final recomendada aquellas canciones cuya distancia sea la menor posible en relación a la playlist generada según las preferencias del usuario.

Pre-procesamiento

Antes de comenzar a implementar el algoritmo principal de recomendaciones se realizaron tareas de pre-procesamiento de los datos con el objetivo de ensamblar adecuadamente nuestro sistema de recomendación final.

En primer lugar, teniendo en cuenta que muchos de los valores contenidos en las diferentes columnas del dataset oscilaban en un rango amplio y distintas escalas según cada característica, se procedió entonces a normalizar estos valores de tipo float para contar con dichos datos dentro de magnitudes similares. Como por ejemplo, para la columna de popularidad, dada la media de estos valores, se los convirtió a una escala del 0 al 20 según porcentajes calculados.

Además, con el objetivo de optimizar ciertos datos para el posterior procesamiento a través de un algoritmo de aprendizaje automático, se definió una función que dado un conjunto de datos y una columna seleccionada del mismo aplica la estrategia de One-Hot Encoding, método que permite convertir los datos correspondientes a nombres de, en este caso géneros y las diversas características de las canciones, a nuevas columnas que representen dichas características como categorías asignándoles valores numéricos donde cada valor corresponde a un vector binario, incluyendo 0’ o 1’ según a los distintos registros del dataset según contenga o no dicha categoría.

Vectorización
Todos los pasos anteriores de pre-procesamiento tendrán su utilidad en la definición de nuestra función para el proceso de vectorización de la información del dataset que consideremos necesaria. Más precisamente, se aplicó el método TF-IDF Vectorizer de la librería de Scikit-Learn, donde el algoritmo de TF-IDF determina la relevancia de una palabra o frase dentro de una colección de datos, transformando así dicho texto en una representación numérica la cual será de gran utilidad en la toma de decisiones en base a este tipo de valores. Según esta medida, el valor tf-idf aumenta proporcionalmente al número de veces que una palabra aparece en el conjunto de datos, pero es compensada por la frecuencia de la palabra en la colección del conjunto de datos, lo cual permite manejar el hecho de que algunas palabras son generalmente más comunes que otras.

Esto último será un factor crucial para nuestro algoritmo en relación a la aplicación de dicha vectorización sobre los valores correspondientes a los géneros, pues, considerando que contamos con un amplio abanico de diferentes géneros en nuestro conjunto de datos, resulta razonable que a ciertos géneros más genéricos como “pop” o “rock”, que si bien muchisimas veces, no se les de tanta relevancia como a otros géneros no tan comunes como “classic_latvian_pop” o “rock_urbano_mexicano” por ejemplo, que si bien no cuentan con gran cantidad de ocurrencias resultan mucho más específicos por lo que podrían conducir a recomendaciones más personalizadas.

Finalmente, tras la normalización de algunos valores, la utilización del One-Hot Encoding y la aplicación del vectorizador TF-IDF, obtenemos una función que nos retorna un conjunto de datos con todos los pesos o puntajes asignados según los diferentes géneros, características técnicas, popularidad y año para todas las canciones del dataset. A continuación una muestra de esto:

genre|british_folk	...	genre|celtic_punk	genre|celtic_rock	...	genre|classic_bollywood	...	genre|classic_uk_pop	...	genre|folk_rock	...	genre|zydeco
0.000000	...	0.0	0.000000	...	0.000000	...	0.000000	...	0.000000	...	0.0
0.000000	...	0.0	0.000000	...	0.000000	...	0.000000	...	0.000000	...	0.0
0.000000	...	0.0	0.000000	...	0.000000	...	0.000000	...	0.000000	...	0.0
0.590353	...	0.0	0.604885	...	0.000000	...	0.441204	...	0.301557	...	0.0
0.000000	...	0.0	0.000000	...	0.474161	...	0.000000	...	0.000000	...	0.0

acousticness	danceability	...	tempo	valence	pop|0	pop|1	pop|2	...	pop|20	year|1920	year|1921	...	year|2021	id
0.191566	0.143522	...	0.096311	0.1344	0.15	0.00	0.0	...	0.0	0.0	0.0	...	0.0	1ceK5XrTBAoX5K32skksSx
0.148795	0.106680	...	0.092455	0.0966	0.00	0.00	0.0	...	0.0	0.0	0.0	...	0.0	1grsf0up9GXHPrq5uiXZOx
0.152008	0.144130	...	0.055778	0.1068	0.15	0.00	0.0	...	0.0	0.0	0.0	...	0.0	3waWRbWlxXtBja0SW8IlkC
0.110040	0.138259	...	0.101904	0.1300	0.00	0.15	0.0	...	0.0	0.0	0.0	...	0.0	4AzC5QUV6AuWZfoWyXcdkR
0.199598	0.069838	...	0.056108	0.1366	0.15	0.00	0.0	...	0.0	0.0	0.0	...	0.0	0mQBIvf5bxkUq7LIJeHXah

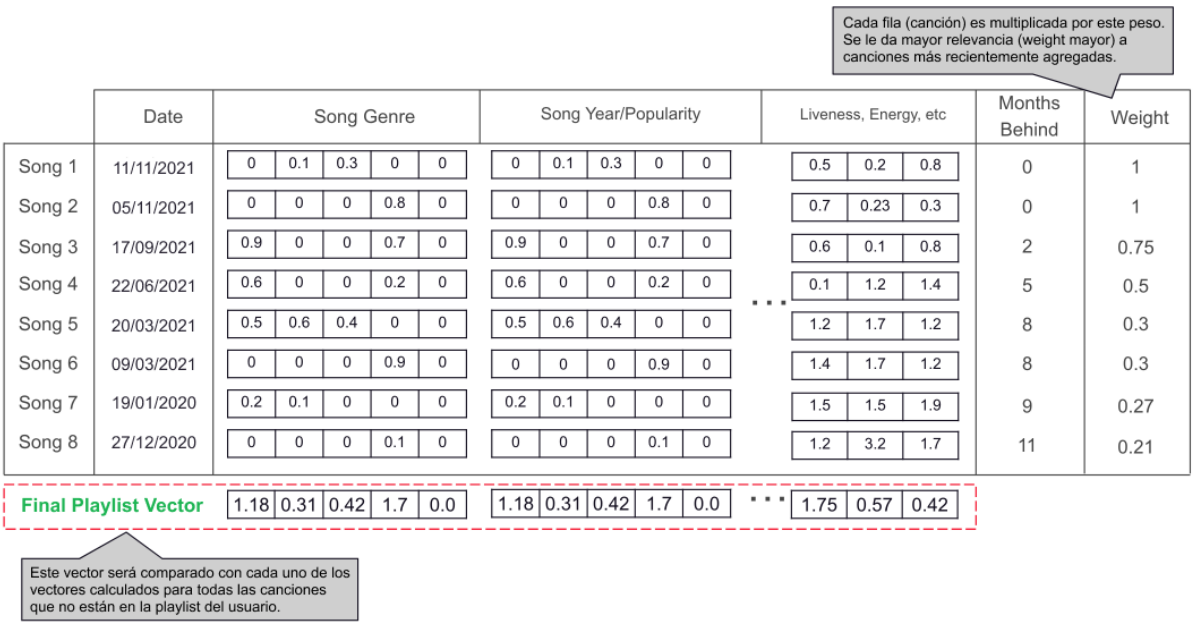
Generación del vector playlist:
Tras contar con prácticamente todos los requerimientos necesarios ya implementados para nuestro algoritmo de recomendaciones, resta definir el sistema de puntuaciones sobre las canciones extraídas de las playlists del usuario en cuestión

que utilizaremos para aplicar la medida de similitud de coseno que permita obtener finalmente un conjunto de canciones recomendadas.

Para crear dicho vector se planteó el siguiente sistema de puntuación a modo de matriz donde: cada canción extraída de la playlist está representada en cada fila y cada una de sus columnas corresponden a los valores asignados según cada una de las medidas ponderadas para cada una de las diferentes características (géneros, popularidad, año, aspectos técnicos) que le fueron asignadas en el paso anterior según el método TF-IDF como observamos en la captura anterior del conjunto de datos obtenido tras la vectorización de los textos y la aplicación de one-hot encoding y normalización.

Aparte de los puntajes asignados según las características mencionadas antes, otro factor clave que se optó por tener en cuenta para nuestro sistema de puntuación fue la fecha en que el usuario agregó una canción a la playlist. La razón de esto fue que en ocasiones un usuario crea una playlist en un momento determinado y en el correr del tiempo va sumando nuevas canciones a la misma según sus preferencias en ese periodo de tiempo determinado, de modo que posiblemente las canciones que fueron agregadas hace mucho más tiempo tal vez actualmente no reflejen fielmente las preferencias o gustos del usuario de la misma forma que lo hacía tiempo atrás. Por esto, se decidió agregar una nueva columna
a la matriz del sistema de puntuación donde en base a cuantos meses atrás fue agregada cada canción se le asigne un peso tal que mientras más reciente sea cada canción en la playlist mayor sea ese valor (1 si fue agregada hace menos de un mes y tendiendo a 0 mientras mayor sea la cantidad de meses) y por ende una mayor incidencia en el puntaje del vector final, pues este peso se utilizará para realizar el producto con cada uno de los demás valores de la matriz.

El resultado final tendrá la forma de un vector donde la cantidad de coordenadas estará determinada por la cantidad de características generadas por tf-idf, obteniéndose el valor final de cada coordenada a través de la suma (según diferentes columnas) de cada uno de los puntajes asignados a las canciones de la playlist.



Generación de recomendaciones
Finalmente, luego de numerosos pasos previos solamente quedaba por combinar todas esas funcionalidades en una función final que tome como entrada una playlist seleccionada de la cuenta de Spotify de un usuario dado y obteniendo finalmente como salida una playlist personalizada de canciones -del largo deseado- como recomendación en base a las preferencias musicales y su respectiva actividad en la plataforma por parte un usuario dado.

Resultados obtenidos

Playlists del usuario en Spotify extraídas a través de la librería Spotipy:

```
# Playlists del usuario
id_name
```

```
{'Test': '3b3pLfbj8p8ATIyM14WpPi',
 'BluesJazzSoul': '4RgwtQGN3qiORBAJY8YmAg',
 'Playlist1': '1599AuhrUlMvNC3XWM4NrF',
 'allTimeClassics': '2co6vXpW30Rk4eb4K3PGNG',
 'ARG': '7ybhLQL70w6KQPyT6wSRYK',
 'New Playlist': '1WC5jUNoWr0Tj697qm2LlR',
 'Spanish': '1uAwFUFfEVsuzUX8VYh8Nl',
 'Daily Mix 3': '37i9dQZF1E38q3cgbIjQBV',
 '4K': '0c26eG6bjeuJsBf7dmlrmE',
 'Random': '4i4p9lg8O3wvsAaOYiIM6I'}
```

Prueba 1:

ENTRADA (Playlist seleccionada):
Para este ejemplo de ejecución de nuestro algoritmo se escogió la playlist llamada “BluesJazzSoul”, la cual por su nombre podemos tener un primer adelanto respecto a qué géneros encontraremos en la misma.

A continuación se listan en orden descendente según la fecha en que fue agregada cada una de las canciones de la playlist.

```
# Canciones contenidas en la playlist seleccionada de Spotify y en el dataset
playlist_user
```

	artist	name	id	...	date_added
16	James Brown	That's Life	2ypFEmLB5q5j2l34bFxF8Y	...	2021-10-26 18:07:08+00:00
15	James Brown & The Famous Flames	It's A Man's, Man's, Man's World	3SQ9Hb9rfpJ02AWfaOPhpy	...	2021-10-26 18:03:53+00:00
14	Herbie Hancock	Cantaloupe Island - Remastered	0sCeNwt8xRCMR4NhKpMyBe	...	2021-10-26 17:57:15+00:00
13	B.B. King	The Thrill Is Gone	3cg0dJfrQB66Qf2YthPb6G	...	2021-10-26 17:41:22+00:00
12	B.B. King	Lucille	4ZSJs1cqeincEi2KjUGmZC	...	2021-10-26 17:36:18+00:00
11	Luther Allison	Bad News Is Coming	1TRIYxcVJYPki5AX2hZzZT	...	2021-10-26 17:33:04+00:00
9	Aretha Franklin	I Say a Little Prayer	7haFcQaoTBr2qY6G0r4JSH	...	2021-10-21 17:14:40+00:00
8	James Brown	People Get Up And Drive Your Funky Soul - Remix	5U3i59kbTLrxo46TU1FRnF	...	2021-10-21 17:14:34+00:00
7	Kool & The Gang	Get Down On It - Single Version	4yKZACkuudvfd600H2dQie	...	2021-10-21 17:14:14+00:00
5	Nina Simone	My Baby Just Cares for Me - 2013 Remastered Ve...	6VTbbVjKOC2qWagIDbkJrC	...	2021-10-21 17:13:44+00:00
2	Cream	Crossroads - Live	1sxtxIKhRiQwuGpUwHoEHq	...	2021-10-21 17:12:29+00:00
1	Albert King	Born Under A Bad Sign - Mono Mix	3ocm1Cf1Dk1ODrdBdybh82	...	2021-10-21 17:12:22+00:00
0	B.B. King	To Know You Is To Love You	4c0Mgs559lG4sPiEC3XcxY	...	2021-10-21 17:12:09+00:00

OBS: *Cabe destacar que ciertas canciones que se encuentran en Spotify podrían no estar incluidas en nuestro dataset por ello tratamos de armar playlists con canciones que si estuvieran.*

SALIDA (Playlist recomendada):
A partir de la playlist elegida, el dataset utilizado y las distintas funcionalidades que se fueron implementado invocamos a nuestra función generadora de playlist recomendada y obtuvimos el siguiente conjunto de canciones ordenadas de mayor a menor según similaridad:

```
playlist_recomendada.head(10)
```

	artists	...	id	...	name	popularity	release_date	...	song_genres_list	popularity_scaled	sim
148523	['B.B. King', 'Bobby "Blue" Bland']	...	3BNcwgLb29DG60BBWfEWQE	...	That's The Way Love Is - Live At Western Recor...	19	1974-01-01	...	[soul, classic_rock, traditional_blues, soul_b...	3	0.671634
8588	['B.B. King']	...	5iiBzeQs7NzeAHZaJFUQH4	...	Come By Here	7	1958-01-01	...	[classic_rock, traditional_blues, soul_blues, ...	1	0.668864
8602	['B.B. King', 'Bobby "Blue" Bland']	...	3m6FB5phiTsCEvFeDwlBay	...	3 O'Clock Blues - Live At Western Recorders St...	21	1974-01-01	...	[soul, classic_rock, traditional_blues, soul_b...	4	0.668673
8534	['B.B. King']	...	3mrrWFfuesBi5fswdz1Xxf	...	So Excited	30	1969-12-05	...	[classic_rock, traditional_blues, soul_blues, ...	6	0.664266

...

148565	['B.B. King', 'Bobby "Blue" Bland']	...	2AgBtTjP0OGJH5GoZWjpay	...	I'm Sorry - Live At Western Recorders Studio1/...	18	1974-01-01	...	[soul, classic_rock, traditional_blues, soul_b...	3	0.664173
148513	['B.B. King']	...	2c2SLDFaxkpcJG4t3YElvM	...	Treat Me Right	7	1958-01-01	...	[classic_rock, traditional_blues, soul_blues, ...	1	0.663045
148601	['B.B. King', "D'Angelo"]	...	3hCIZ8sgJm5MUh7ulehPGX	...	Ain't Nobody Home	42	1997-11-04	...	[classic_rock, traditional_blues, soul_blues, ...	8	0.662369
148564	['B.B. King']	...	0U4HvBg1x5k4XMBKI1cfds	...	I've Got Papers On You, Baby	8	1958-01-01	...	[classic_rock, traditional_blues, soul_blues, ...	1	0.662277
148520	['B.B. King']	...	3I7PM0ArfQPsLWpk4auNGx	...	The Woman I Love	9	1958-01-01	...	[classic_rock, traditional_blues, soul_blues, ...	1	0.662175
148509	['B.B. King']	...	5TPFCtAbbcSRooduH3cyHo	...	We Can't Make It Right	8	1958-01-01	...	[classic_rock, traditional_blues, soul_blues, ...	1	0.660961

Como podemos observar hubo un artista, y por ende sus respectivos géneros, que influyó notoriamente en las recomendaciones obtenidas, siendo este B.B King, uno de los principales exponentes del blues en la historia de la música.

De todos modos, si se toma una muestra diferente de la salida podemos corroborar que también aparecen otros artistas relacionados con dichos géneros como Muddy Waters, Eric Clapton, Albert King, por ejemplo.

```
playlist_recomendada.tail(10)
```

	artists	id	name	popularity	release_date	song_genres_list	popularity_scaled	sim
60599	['Muddy Waters']	4MWQ0HRkPlWq70BKxeY1S1	Walkin' Thru The Park	29	1966-01-01	[soul, chicago_blues, classic_rock, traditiona...	5	0.617200
148536	['B.B. King']	0vUiPygLO2QnZpliFWNh3M	Servant's Prayer	7	1959-01-01	[classic_rock, traditional_blues, soul_blues, ...	1	0.617027
148496	['B.B. King']	0XM1Tk2fwpDtrnwklhEWXX	You Know I Go For You	12	1957-01-01	[classic_rock, traditional_blues, soul_blues, ...	2	0.616594
148490	['B.B. King']	3DbVBS8JojEGfuLOZRGgWX	You're Still My Woman	26	1970-01-01	[classic_rock, traditional_blues, soul_blues, ...	5	0.614808
148546	['B.B. King', 'The Rolling Stones']	1HK0gVdcEwTUaJzCTNhjA3	Paying The Cost To Be The Boss	43	1997-11-04	[album_rock, classic_rock, rock, traditional_b...	8	0.614220
148531	['B.B. King']	6F76ic7c6au3QxG6jaso7N	Sweet Little Angel - Live At The Regal Theater...	38	1965	[classic_rock, traditional_blues, soul_blues, ...	7	0.614065
60656	['Muddy Waters']	65VzqBb5ogD2xIFD8jetMG	Got My Mojo Working	50	1997-05-20	[soul, chicago_blues, classic_rock, traditiona...	10	0.613988
60619	['Muddy Waters']	2cb3fhkmr3pZO9CYsXfa4M	My John The Conqueror Root	21	1964-04-05	[soul, chicago_blues, classic_rock, traditiona...	4	0.613886

...

148539	['B.B. King', 'Eric Clapton']	1bursm5ZnNHQS55a23h7kR	Rock Me Baby	45	1997-11-04	[album_rock, soft_rock, classic_rock, rock, tr...	9	0.613756
154832	['Albert King']	0eHclX6LLS9t2bfDLZ6Vhe	Kansas City	24	1967	[classic_rock, traditional_blues, soul_blues, ...	4	0.613718

Prueba 2:

ENTRADA (Playlist seleccionada):
Para este ejemplo se eligió una playlist creada por el usuario con canciones pertenecientes a grandes bandas o artistas referentes del rock argentino como Luis Alberto Spinetta (Pescado Rabioso), Gustavo Cerati (Soda Stereo) y Charly García, Fito Paez, por ejemplo.

```
# Canciones contenidas en la playlist seleccionada de Spotify y en el dataset
playlist_user
```

	artist	name	id	url	date_added
8	Fito Paez	11 Y 6	2PkTjm1QAJCZbJ9MqC4RqA	https://i.scdn.co/image/ab67616d00001e02921417...	2021-11-15 18:17:42+00:00
6	Pescado Rabioso	Bajan	6hmYi0E6EBEmDeztQHaH0C	https://i.scdn.co/image/ab67616d00001e0250db5a...	2021-11-15 18:16:56+00:00
3	Charly García	No Voy en Tren	5MV4J9OdmVGz63b1gnezp6	https://i.scdn.co/image/ab67616d00001e0242542c...	2021-11-15 18:16:31+00:00
0	Soda Stereo	Un Millón De Años Luz - Remasterizado 2007	0nJrTGcXdJWkKz02UMQdzO	https://i.scdn.co/image/ab67616d00001e02c451ed...	2021-11-15 18:15:56+00:00

SALIDA (Playlist recomendada):

	artists	id	name	song_genres_list	popularity_scaled	similarity
115571	['Fito Paez']	0fAkoHmERNn6PuRqfK7411	Polaroid De Locura Ordinaria	[rock_nacional, rock_en_espanol, argentine_roc...	11	0.815942
135167	['Charly García']	2MXqrO1RBfek6RoijghYYp	Promesas Sobre El Bidet	[rock_nacional, rock_en_espanol, post-punk_arg...	11	0.827196
63656	['Miguel Mateos']	1l0mEM93oZMERzBmOCuiHe	Obsesión	[rock_nacional, rock_en_espanol, argentine_roc...	12	0.807303
38002	['Serú Girán']	5bXhx2vO2N4ZwUi0QFn4HU	Seminare	[rock_nacional, argentine_indie, rock_en_espan...	12	0.745652
31241	['Sui Generis']	47im9FzRfqAS2KYvw3NhSw	Confesiones de Invierno	[rock_nacional, rock_en_espanol, nueva_cancion...	10	0.726916
153696	['Almendra']	6GPkBUXMrGSkwK3TrrHHXN	Muchacha (Ojos de Papel)	[rock_en_espanol, argentine_rock, latin_rock, ...	11	0.726241
70951	['Luis Alberto Spinetta']	3T5SksrIFxunlotQ0lmSwv	Cancion para los Días de la Vida	[rock_nacional, argentine_indie, rock_en_espan...	9	0.767160
115576	['Fito Paez']	4Ft15s9T7pcG7O85VbdxVl	Al lado del camino	[rock_nacional, rock_en_espanol, argentine_roc...	12	0.798859
34920	['Soda Stereo']	65DBZofl0b79kfHTcWWDuU	Trátame Suavemente - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	14	0.724609
34965	['Soda Stereo']	6WB3ovHYXvhXfXQnAeE5qj	Danza Rota - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	9	0.779122

...

34958	['Soda Stereo']	61HI37BKbYcSeIJ3Sd8oU0	El Rito - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	11	0.732990
34949	['Soda Stereo']	6ogBARwwlfsmmgyxWeEzQF	Estoy Azulado - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	9	0.775282
34940	['Soda Stereo']	3JCz3RYG6psLPHrW5ZF1wg	Mundo De Quimeras - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	8	0.725073
34935	['Soda Stereo']	71awpJoi5bqGMBrTkHDDoL	Persiana Americana - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	13	0.727298
34933	['Soda Stereo']	2JD91cpdSGoNh9K50GmByg	Por Qué No Puedo Ser Del Jet Set? - Remasteriz...	[rock_en_espanol, post-punk_argentina, argenti...	10	0.730189
38006	['Serú Girán']	0HOkeup7WSNOaOVQKbDNp	Mientras Miro Las Nuevas Olas - Remastered	[rock_nacional, argentine_indie, rock_en_espan...	8	0.749446
101191	['Gustavo Cerati']	3ac2SnHk4KYjP2Au4ZL7eu	Lisa	[latin_rock, rock_en_espanol, argentine_rock, ...	10	0.790687
51469	['Pescado Rabioso']	5K6fJf9I8WTsscngGert0b	Las Habladurias del Mundo	[rock_nacional, argentine_indie, rock_en_espan...	9	0.808791
51468	['Pescado Rabioso']	79vy4Ma2UIfzXxJx0SXLGL	Por	[rock_nacional, argentine_indie, rock_en_espan...	8	0.795811
135166	['Charly García']	7mnf5omGMUwSfpwZrt5AZ0	Raros Peinados Nuevos	[rock_nacional, rock_en_espanol, post-punk_arg...	10	0.824068

En este caso, se aplicó nuevamente la función generadora y a través de la función sample de la librería Pandas, recuperamos un subconjunto aleatorio de 20 canciones dentro del total de canciones recomendadas por dicha función.

A diferencia de la prueba 1 con la otra playlist, en esta oportunidad los resultados obtenidos fueron sumamente satisfactorios teniendo en cuenta algunos de los siguientes factores:

- En primer lugar, en esta ocasión el conjunto de canciones recomendadas si contó con una gran variedad de diversos artistas o bandas que a simple vista, por conocimiento popular, podrían considerarse similares, dado que prácticamente en su totalidad son nombres muy populares en la historia del rock argentino.
- Además, una observación no menor que vale la pena destacar es que por ejemplo, en nuestra playlist tomada como entrada, incluimos una canción del famoso grupo musical de los 70’ llamado Pescado Rabioso liderado por Luis Alberto Spinetta, artista del cual no incluimos ninguna canción como solista y sin embargo, pese a no estar explícitamente su nombre en el campo de artista, la función nos recomendó canciones suyas, tanto como solista como de sus otras bandas, como lo fue Almendra por ejemplo; mismo caso con Soda Stereo, banda de la cual agregamos una canción y obtuvimos una canción solista recomendada de su líder, Gustavo Cerati. Algo similar pero en el caso inverso sucedió con el artista Charly García, del cual incluimos una canción

suya como solista y el algoritmo incluyó canciones de dos de sus grandes bandas como lo fueron Sui Generes y Seru Giran.

```
playlist_recomendada.head(2)
```

	artists	id	name	song_genres_list	popularity_scaled	similarity
53191	['Patricio Rey y sus Redonditos de Ricota']	5WeBnrDPyLhxruxVryHCkn	La Bestia Pop	[rock_nacional, rock_en_espanol, post-punk_arg...	11	0.882541
115573	['Fito Paez']	0XeDFZzlpnhUT4OAXsFjfx	Giros	[rock_nacional, rock_en_espanol, argentine_roc...	9	0.860158

...

```
playlist_recomendada.tail(2)
```

	artists	id	name	song_genres_list	popularity_scaled	similarity
34922	['Soda Stereo']	5ZDMWyxvoEEi5ofWUjf0NL	Tele-Ka - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	9	0.724308
34934	['Soda Stereo']	1tooZ43bHR3bSVIDNwEZRM	Pic Nic En El 4° B - Remasterizado 2007	[rock_en_espanol, post-punk_argentina, argenti...	9	0.724283

- Otro dato a destacar fue que se logró un mayor grado de similaridad para esta prueba, pues las 40 canciones recomendadas por el algoritmo obtuvieron un puntaje de similaridad en un rango entre aproximadamente 0.72 y 0.88, valores que, recordando que según la medida de similaridad del coseno, mientras más se aproximen a 1, menor es la distancia entre los vectores calculados y por ende, mayor es la similaridad lograda. Por ello, dada su cercanía a 1, y en comparación a la primera prueba donde los niveles de similitud fueron menores, los resultados fueron ampliamente satisfactorios.

Modelo de recomendación de filtrado colaborativo

Como se mencionó previamente, Spotify además del modelo de recomendación basado en contenido incorpora un modelo de recomendación por filtrado colaborativo para apoyarse en los usuarios más allá del que es recomendado en cuestión para obtener mejores recomendaciones.

¿Qué es un modelo de recomendación por filtrado colaborativo?

Es un modelo en el cual se recopila información de los gustos y preferencias de contenido de los usuarios para poder, en base a esa información, recomendar contenido contemplando el consumido por usuarios con gustos y preferencias similares. Muchas plataformas utilizan este modelo de recomendaciones, tales como Amazon y Netflix entre otras. La diferencia con estas plataformas es que Spotify no cuenta con un sistema de puntuación de canciones o artistas, por lo que surge la pregunta de cómo es que hace la plataforma para deducir los gustos musicales de los usuarios, y la respuesta es que lo hace mediante el análisis de la frecuencia en la cual un usuario escucha ciertas canciones.

¿Cómo se piensa este tipo de filtrado?

La clave para las recomendaciones colaborativas es la capacidad de poder moldear los gustos de un cierto usuario, para poder tener una representación tangible para poder comparar con la representación de otro usuario. Es por ello que se recurre a métodos de abstracción de los datos del sistema para poder expresarlos de una manera simple y efectiva, y poder así realizar las comparaciones de manera simple y efectiva con algoritmos lo menos costosos posibles dado que de por sí, estos procesos necesitan mucho procesamiento dados los grandes volúmenes de datos con los que se trabaja.

¿Cómo se realiza este tipo de filtrado?

El filtrado de este tipo se puede realizar utilizando vectores binarios. Esta es la técnica utilizada por Spotify y es la que se describe a continuación:

Cada usuario puede ser representado mediante el uso de un vector binario compuesto por ceros y unos. Estos ceros y unos representan las diversas canciones que contiene la plataforma, siendo cero el indicador de que dicha canción no fue escuchada por el usuario y uno el indicador de que esa canción efectivamente fue escuchada por el usuario. Ya con esta información se puede conseguir un perfil de usuario que moldee sus gustos musicales de una manera simple y efectiva, pero aún resta más por analizar para poder mejorar la fidelidad del modelado de estos gustos personales, por ejemplo un aspecto clave es el de la frecuencia con la que dichas canciones se escuchan. Es decir, no es lo mismo una canción que el usuario escuchó por última vez hace dos años que una que el usuario escuchó ayer, y para ello se incorpora un sistema de multiplicación de estos vectores por constantes que reducen

el valor de estos unos en el vector para quitarle "peso" a las canciones que ha escuchado hace más tiempo.

Una vez tomadas en cuenta estas consideraciones ya se obtiene un buen modelo que representa los gustos musicales de un usuario, y con él se puede tomar el de otro usuario de la plataforma y compararlos para ver qué tan cercanos son.

¿Cómo se implementó este algoritmo?

En cuanto a implementación, la realización es simple ya que el modelo como fue mencionado es sencillo y las herramientas provistas por las librerías seleccionadas se encargan de la lógica más pesada detrás de los conceptos requeridos para llevar a la práctica el algoritmo.

Se comienza por tomar los datos recabados de los usuarios mediante la lectura de un archivo .csv que contiene una fila conteniendo el nombre de las canciones, la siguiente conteniendo el género correspondiente, y finalmente tres filas más, una por cada sujeto de estudio que en este caso fue cada uno de los tres participantes de este proyecto en cuestión. La fila correspondiente a cada usuario contiene un valor asignado a cada canción. Ese valor es el puesto que la canción posee en un ranking de las canciones más escuchadas por el usuario en un período determinado de tiempo, que a modo de ejemplo se tomó el período de cuatro semanas y las veinte canciones más escuchadas en orden descendiente.

Una vez el código tiene acceso a esos datos, pasa analizar para el usuario al cual se le quiere hacer una recomendación la distancia euclidiana a cada uno de los demás usuarios que se estudian. Esta distancia es la seleccionada dado que por el modelo utilizado los usuarios, tenemos a cada usuario como un vector de valores y dicha ecuación es la idónea para encontrar una distancia entre dos vectores que potencialmente pueden tener muchas dimensiones.

La ecuación se describe en la siguiente imagen, y es provista en el código por la librería **Scipy**.

The Euclidean distance measure given in Equation 2.1 is generalized by the **Minkowski** distance metric shown in Equation 2.2,

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{k=1}^{n} |x_k - y_k|^r \right)^{1/r}, \tag{2.2}$$

where r is a parameter. The following are the three most common examples of Minkowski distances.

- $r = 1$. City block (Manhattan, taxicab, L_1 norm) distance. A common example is the **Hamming distance**, which is the number of bits that are different between two objects that have only binary attributes, i.e., between two binary vectors.
- $r = 2$. Euclidean distance (L_2 norm).
- $r = \infty$. Supremum (L_{max} or L_∞ norm) distance. This is the maximum difference between any attribute of the objects. More formally, the L_∞ distance is defined by Equation 2.3

$$d(\mathbf{x}, \mathbf{y}) = \lim_{r \to \infty} \left(\sum_{k=1}^{n} |x_k - y_k|^r \right)^{1/r}. \tag{2.3}$$

Durante el proceso de calcular la distancia para cada otro usuario del sistema que resulte interesante evaluar, se guarda la distancia mínima en cada caso y al final se registra cuál fue el usuario con la mínima distancia resultante en total. Es de éste usuario que se toman sus canciones más escuchadas, se restan las que el usuario a recomendar ya tiene en sus más escuchadas, y le son recomendadas las restantes.

Es así cómo se implementó, de manera simple, un algoritmo sólido de recomendaciones por filtrado colaborativo.

Trabajo a Futuro

Respecto al algoritmo de recomendaciones basado en contenido, dada la situación con la que nos encontramos en la primera prueba que fue realizada tomando como entrada la playlist *BluesJazzSoul*, donde si bien la cual contenía varias canciones de diferentes artistas cuyos géneros se encontraban relacionados efectivamente con los del nombre de la playlist, el algoritmo retorna un conjunto de recomendaciones donde predomina fuertemente un artista en particular, apareciendo en la gran mayoría de las canciones sugeridas. Pese a que en la segunda prueba realizada el comportamiento fue totalmente distinto y satisfactorio, obteniendo recomendaciones muy diversas y con altos grados de similaridad, no resulta demasiado raro este comportamiento planteado como problemático, pues es de esperar que ante canciones de un artista específico se obtengan como canciones más similares, otras canciones del mismo artista. Sin embargo, si nos detenemos a observar cómo funciona en la práctica Spotify, comúnmente sugiere sí, otras canciones del mismo artista, pero por supuesto también lo hace de manera mixta agregando canciones de otros. Por ello, se nos ocurre como una posible incorporación o modificación del algoritmo, una funcionalidad de filtrado que permita obtener playlists más diversas, limitando el número de canciones a agregar de un mismo artista, recorriendo probablemente un conjunto mayor de canciones sugeridas, en orden según sus respectivos puntajes de similaridad.

Otro aspecto a tener en cuenta es que los datasets con los que se realizó el trabajo se encuentran desactualizados, lo cual fue un limitante a la hora de crear playlists para realizar posteriores pruebas observando que muchas de las canciones que se agregaban no se encontraran en el dataset. Dada la inmediatez y dinamismo de la industria musical, donde miles de canciones son lanzadas y agregadas constantemente a la plataforma de Spotify, de hecho según datos oficiales cuenta con alrededor de 70 millones de canciones, resultaría imposible poder contar con un dataset fijo y actualizado, sin embargo una buena opción sería trabajar como punto de partida con un dataset dado, pero implementar una función que permita extraer canciones del catálogo de Spotify que no se encuentren el dataset y almacenarlas en estructuras dinámicas haciendo uso de potentes librerías. Esto sin dudas ayudaría a lograr recomendaciones cada vez más precisas aumentando considerablemente las opciones a la hora de recomendar canciones.

Por el lado de las recomendaciones por filtrado colaborativo, se identifica que el algoritmo puede ser mejorado tomando en cuenta a la hora de calcular la distancia entre dos usuarios, los géneros de las canciones. Es decir, no sólo calcular la distancia por canciones en particular escuchadas por cada usuario, sino que darle un peso total a todas esas canciones categoralizándolas dentro de sus respectivos géneros para evitar escenarios donde dos usuarios no tengan mucho en común en cuanto a canciones escuchadas, pero dado que tienen la mínima distancia se compartirán canciones aún cuando puede haber otro usuario con ninguna canción en común, pero con una notoria similaridad de géneros escuchados y por lo que ese usuario resultaría más apropiado para basar las recomendaciones.

Otra posibilidad de trabajo a futuro, a pesar de que seguramente se aleje bastante del algoritmo originalmente planteado, sería abordar la problemática de recomendaciones de podcasts. Los mismos han adquirido una popularidad increíble en los últimos años y al día de hoy la plataforma cuenta con más de 3 millones de podcasts. Si bien muchas de las características y funcionalidades de nuestro algoritmo no aplicarían correctamente al caso, muchas otras sí, y por ejemplo, haciendo una analogía entre géneros de canciones con temáticas de podcasts, artistas como creadores de podcasts, y otras características a definir, posiblemente se podría diseñar un sistema de puntuación como el que realizamos para determinar similitudes y generar recomendaciones relacionadas a este tipo de contenido.

Conclusiones

Como usuarios de una plataforma como Spotify que forma parte de nuestra cotidianidad, el desafío propuesto de investigar cómo funciona su potente y exitoso sistema de recomendaciones, el cual lo hace nada más ni nada menos que destacarse frente a su competencia, hizo que resultare una tarea muy interesante de abordar. Las conclusiones que se desprenden por parte de los creadores de este informe son que la base del éxito del algoritmo son la conjunción de dos algoritmos de recomendación potentes ya que por sí solos los mismos no dan los mejores resultados esperables para un usuario que desea descubrir música, y esto se considera es por las siguientes razones:

1. Al recomendar canciones únicamente por contenido, se termina o bien recomendando los mismos artistas que el usuario ya conoce y escucha, o se termina profundizando principalmente en los mismos géneros. En la segunda opción la gran carencia es que dentro de un género pueden haber múltiples artistas enormemente diferentes pero que encajen bajo los mismos parámetros que describen su música y finalmente el usuario no disfrute de su música y la recomendación obtenida.
2. Al recomendar canciones únicamente por filtrado colaborativo se puede caer en recomendaciones que dejen al usuario desconcertado por la diferencia de estilo con sus preferencias musicales y poco satisfecho con los resultados. Esto es porque un usuario puede escuchar en un 80% la misma música que el usuario a ser recomendado, pero que el restante 20% sea música que no sea para nada del agrado del otro. Incluso puede darse el caso en un sistema pequeño que ningún usuario comparta realmente gustos con él, y que aún así el algoritmo encuentre a dos con la distancia mínima el uno del otro y termine recomendando música para nada acorde a lo esperado.

Es por estos dos puntos que se concluye que la clave del éxito reside en un uso en conjunto de ambos procesos y que el resultado final se componga por una intersección del set de recomendaciones por contenido junto con el set de recomendaciones por filtrado colaborativo. Esto implica que ambos deberán ser lo más grandes posibles y finalmente esto deriva en un aspecto clave que es que estos

algoritmos deben resultar sumamente óptimos para que la experiencia de los usuarios sea placentera y fluida.

Resulta interesante ver cómo esta investigación deja abierta la posibilidad a muchas formas de mejorar el proceso de recomendación, como se mencionó en la sección de trabajo a futuro, y también cómo estos algoritmos conforman una componente clave de los sistemas de uso masivo al punto de ser quienes marcan la diferencia entre si un usuario elige una plataforma u la otra para escuchar su música, ver sus películas, descubrir nuevos libros, y consumir cualquier tipo de entretenimiento. Se vio el trabajo pesado que llevan de fondo consigo, y cómo los conceptos de Recuperación de Información y Recomendaciones en la web merecen su estudio como rama a parte dado que es un campo en auge y que puede y debe ser mejorado para satisfacer las necesidades de un mercado que cada vez accede a más y más plataformas que lo necesitan.

Referencias

1. https://numpy.org/
2. https://scikit-learn.org/stable/
3. https://keras.io/
4. https://developer.spotify.com/
5. https://spotipy.readthedocs.io/en/2.19.0/
6. https://www.kaggle.com/jsongunsw/spotify-datasets
7. https://www.kaggle.com/mrmorj/dataset-of-songs-in-spotify
8. https://towardsdatascience.com/how-to-build-an-amazing-music-recommendation-system-4cce2719a572
9. Building A Spotify Recommendation Engine With Python - YouTube
10. https://towardsdatascience.com/a-simple-song-recommender-system-in-python-tutorial-3e4c111198d6
11. http://www.math.utah.edu/~gustafso/s2018/2270/projects-2018/submittedprojects/sorenNelson/Spotify%27s%20Collaborative%20Filtering.pdf
12. Imagen sobre ecuación de distancia euclidiana tomada del libro Introduction to Data Mining de Michael Steinbach, Pang-Ning Tan, y Vipin Kumar.