Sistemas de recomendación para la creación de listas de reproducción en Spotify

Javier Guillamón Pardo

April 29, 2019

1 Introducción

1.1 Repaso del reto RecSys Challenge 2018

Spotify es un servicio de streaming de música online con mas de 140 millones de usuarios activos y mas de 30 millones de canciones. Una de sus funciones más populares es la capacidad de crear listas de reproducción. Actualmente Spotify almacena mas de 2 billones de listas de reproducción.

El reto propuesto por RecSys Challenge 2018 se centra en la recomendación de música, especialmente en el el reto de la continuación automática de listas de reproducción. Proponiendo canciones apropiadas para añadir a una lista ya creada, un sistema de recomendación puede incrementar la participación del usuario haciendo más sencillo el proceso de creación de listas de reproducción, así como extendiendo la escucha más allá que el final de la lista de reproducción ya existente.

1.1.1 Información general

RecSys Challenge 2018 está organizado por Spotify, la Universidad de Massachusetts, Amherst y Johannes Kepler University, Linz. Para más información sobre los plazos seguidos por la organización del reto y para ver los mejores resultados seleccionados puede visitar su página principal en ACM RecSys Challenge page Buscar una forma mejor de comentar esta parte, no veo como meter esto sin pegarme un tiro en el pie a

1.1.2 Tarea

El objetivo del reto es desarrollar un sistema para la tarea de continuar listas de reproducción de forma automática. Dado un conjunto de datos de listas de reproducción el sistema creado debe generar una lista de canciones recomendadas que pueden añadirse a la lista de reproducción, de este modo *continuando* la lista de reproducción. La tarea se defina formalmente de la siguiente manera:
Entrada Una lista de reproducción creada por un usuario, representada por:

- meta-datos de la lista de reproducción (añadir información del README del dataset)
- K semilla de canciones: una lista de las K canciones en la lista de reproducción, donde K puede ser igual a 0, 1, 5, 10, 25 o 100

Resultado

- Una lista de 500 canciones recomendadas, ordenadas por relevancia en orden decreciente

El sistema debe ser capaz de trabajar con listas de reproducción para las cuales no se suministre ninguna semilla inicial. Para evaluar el desempeño de un sistema, las canciones recomendadas del resultado se comparan con el "conjunto de referencia" de la lista de reproducción original.

1.1.3 Dataset

Como parte de este reto, Spotify ha publicado *The Million Playlist Dataset* (MPD a partir de ahora). Se trata de un conjunto de 1.000.000 de listas de reproducción que han sido creadas por usuarios de Spotify, incluyen titulo de la lista de reproducción, listas de canciones y otros meta-datos explicados en mas profundidad en la siguiente sección Añadir sección

1.1.4 Dataset del reto y formato de entrega

Como parte del reto, RecSys ha publicado un dataset separado del MPD llamado *"test set"* que consiste en 10.000 listas de reproducción con información incompleta. Tiene muchos de los mismos campos de datos y sigue la misma estructura que MDP, pero las listas de reproducción solo incluyen K canciones.

Para cada lista de reproducción en *test set*, hay que entregar a Recsys una lista ordenada con 500 canciones, entregando solo las URIs de las canciones. EL formato del archivo entregado tiene que ser un csv comprimido en gzip (.csv.gz). El orden de las canciones recomendadas importa: las recomendaciones mas relevantes deben aparecer al principio de la lista. La entrega debe ser hecha con el siguiente formato separado por comas:

- Todos los campos tienen que separarse por comas. Es opcional tener espacios vacios antes y despues de las comas.
- Los comentarios están permitidos con un # al comienzo de una linea
- Se permiten lineas vacías, simplemente son ignoradas
- La primera línea no comentada/vacía debe empezar con"team_info", despues incluir el nombre del equipo, el canal al que se está participando (main o creative) e información de contacto

  ```
  team_info, my recsys team, main, my_team_email@gmail.com
  ```

- Por cada lista de reproducción tiene que haber una línea con el siguiente formato:

  ```
  pid, trackuri_1, trackuri_2, trackuri_3, ..., trackuri_499, trackuri_500
  ```

  con exactamente 500 canciones, donde *pid* es el id de la lista de reproducción y *trackuri_n* es el URI de una de las canciones de Spotify recomendadas para esta lista de reproducción.

1.1.5 Notas importantes sobre la entrega

- Las canciones provistas como parte del set del reto no serán incluidas en la entrega
- La entrega no tendrá canciones duplicadas
- La entrega tendrá exactamente 500 canciones, después de quitar las canciones duplicadas
- Cualquier entrega que incumpla alguna de las reglas será rechazada por el sistema de puntuación Un ejemplo de entrega se encuentra en el anexo Añadir referencia, para poder ver como es la entrega esperada.

1.1.6 Métricas

Las entregas son evaluadas usando las siguientes métricas. Todas las métricas serán evaluadas al nivel de canción (las canciones exactas deben coincidir) y al nivel de artista (cualquier canción de ese artista se considera acierto).A continuación, nos referimos al conjunto de canciones objetivo como G, y nos referimos a la lista de canciones recomendadas como R. Nos referimos al tamaño de una lista como $|*|$,y usamos desde:hasta subíndices para indexar una lista.

1. R-precision R-precision es el número de canciones relevantes conseguidas dividido por el número de canciones relevantes conocidas

$$\text{R-precision} = \frac{|G \cap R_{1:|G|}|}{|G|}$$

   Esta métrica es una media de todas las listas de reproducción en el dataset del reto. Premia el número total de canciones relevantes conseguidas, sin importar su orden.

2. Normalized discounted comulative gain (NDCG) Discounted cumulative gain (DCG) mide la calidad de la clasificación de las canciones recomendadas, aumentando cuando canciones relevantes están más altas en la lista. Normalized DCG (NDCG) se determina calculando el DCG y dividiéndolo por el DCG ideal en el que las canciones recomendadas están perfectamente clasificadas.

$$DCG = rel_1 + \sum_{i=2}^{|R|} \frac{rel_i}{log_2(i+1)}$$

   El DCG ideal o IDCG es, en nuestro caso, igual a :

$$IDCG = 1 + \sum_{i=2}^{|G|} \frac{1}{log_2(i+1)}$$

Si el tamaño del conjunto de la intersección de G y R, es vacío, entonces DCG es igual a 0.

La métrica de NDCG se calcula entonces como:

$$NDCG = \frac{DCG}{IDCG}$$

3. Clics en Canciones Recomendadas Canciones Recomendadas es una características de Spotify. Dado un conjunto de canciones de una lista de reproducción, se recomienda 10 canciones a añadir a la lista. La lista puede ser refrescada para añadir 10 canciones mas. Clics en canciones recomendadas es el número de recargas necesarias antes de que una canción relevante sea encontrada. Se calcula con:

$$\text{clics} = \left\lfloor \frac{\arg\min_i\{R_i : R_i \in G|\} - 1}{10} \right\rfloor$$

Si no se puede calcular (en el caso de que no haya ninguna canción relevante en R), devuelve 51, que es el numero máximo de clics posibles mas uno.

1.2 Dataset de entrenamiento

El dataset usado para entrenar el modelo es "THe Million Playlist Dataset", MPD a partir de ahora. Consiste en 1.000.000 de listas de reproducción creadas por usuarios de Spotify. El dataset ha sido distribuido por Spotify para los participantes académicos del reto RecSys 2018.

1.2.1 Qué es MPD

MPD consta de un millón de listas de reproducción generadas por usuarios. Estas listas de reproducción fueron creadas entre enero de 2010 y octubre de 2017. Cada lista de reproducción contiene un titulo, una lista de canciones, información sobre ediciones y más información miscelánea que veremos en mas profundidad en Añadir referencia.

1.2.2 Descripción detallada

MPD consiste en 1000 archivos partidos. Estos archivos siguen la siguiente regla de nomenclatura: mpd.slice.*ID_LSITA_INICIAL-ID_LISTA_FINAL* Por ejemplo, las primearas 1000 listas de reproducción están en el archivo mpd.slice.0-999.json y las últimas 1000 están en el archivo mpd.slice.999000-999999.json Cada archivo es un diccionario JSON con dos campos, *info* y *playlists*

- info El campo *info* es un diccionario que contiene información general sobre una porción del MPD
  - **slice** - rango de listas de reproducción que abarca está porción en concreto

  - **version** - versión actual del MPD
  - **generated_on** - maca de tiempo indicando cuando el archivo fue generado

- playlists lista que contiene 1000 listas de reproducción. Cada lista es un diccionario que contiene los siguientes campos:
  - **pid** - integer - id de la lista de reproducción dentro del MPD, valor entre 0 y 999999
  - **name** - string - nombre de la lista de reproducción
  - **description** - string opcional - si existe es la descripción dada a la lista de reproducción. La mayoría de las listas no tienen descripción
  - **modified_at** - seconds - marca de tiempo (en segundos desde el epoch) en la que la lista fue editada por última vez. Los tiempos se redondean a medianoche GMT del día en el que la lista fue actualizada por última vez
  - **num_artists** - número total de artistas únicos de las canciones de la lista
  - **num_albums** - número total de álbumes únicos de las canciones de la lista
  - **num_tracks** - número de canciones en la lista
  - **num_followers** - número de seguidores de la lista en el momento en el que MPD fue creado. El número no incluye al creador
  - **num_edits** - número de sesiones de edición separadas. Canciones añadidas dentro de una ventana de 2 horas se consideran añadidas en una misma sesión
  - **duration_ms** - milliseconds - duración total de todas las canciones en millisegundos
  - **collaborative** - boolean - si es True la lista es collaborative, múltiples usuarios pueden contribuir a la lista
  - **tracks** - lista con la información de cada una de las canciones de la lista de reproducción. Cada elemento de la lista es un diccionario con los siguientes campos:
    * **track_name** - nombre de la canción
    * **track_uri** - URI de la canción en Spotify
    * **album_name** - nombre del álbum de la canción
    * **album_uri** - URI del álbum en Spotify
    * **artis_name** - nombre del artista principal de la canción
    * **artis_uri** - URI del artista principal en Spotify
    * **duration_ms** - duración de la canción en milisegundos
    * **pos** - posición de la canción en la lista, empezando desde 0

1.2.3 Cómo fue creado el dataset

El dataset de MPD fue creado cogiendo muestras del conjunto de billones de listas de reproducción que los usuarios de Spotify han ido creando sobre los años. Listas de reproducción escogidas aleatoriamente que cumplen los siguientes criterios:

- Creada por un usuario residente en Estados Unidos que tenga al menos 13 años de edad
- En el momento de crear el MPD era una lista pública
- Contiene al menos 5 canciones
- Contiene 250 o menos canciones
- Contiene al menos 3 artistas diferentes
- Contiene al menos 2 álbumes diferentes
- No contiene canciones locales (aquellas que tiene el usuario, que no pertenecen a Spotify)
- Tienen al menos un seguidor, sin contar al creador
- Creada después del 1 de enero de 2010 y antes del 1 de diciembre de 2017
- No tiene un titulo ofensivo
- No tiene un titulo orientado a adultos si el creador de la lista es era menor de 18 años en el momento de la creación

  Adicionalmente, algunas lista de reproducción han sido modificadas:

- Las descripciones de listas de reproducción potencialmente ofensivas han sido eliminadas
- Las canciones añadidas desde el 1 de noviembre de 2017 en adelante han sido eliminadas

Las listas son escogidas aleatoriamente, en la mayoría de los casos, pero en algunas listas se han añadido "paper tracks", entradas intencionalmente erróneas, para poder identificar si el dataset esta siendo usado fuera de las condiciones especificadas.

1.2.4 Datos demográficos

Datos demográficos de los usuarios que han contribuido a la generación de MPD

- **Genero**
  - Masculino: 45%
  - Femenino: 54%

  - No especificado: 0.5%
  - No binario: 0.5%

- **Edad**
  - 18-24: 43%
  - 25-34: 31%
  - 35-44: 9%
  - 45-54: 4%
  - 55+: 3%
  - otros: 10%

- **Nacionalidad**
  - Estados Unidos: 100%

1.3 Dataset del reto

El dataset consta de 10.000 listas de reproducción incompletas.

1.3.1 Formato

Un único diccionario JSON con tres campos:

- **date** - fecha en la que el dataset fué generado. Debería ser "2018-01-16 08:47:28.198015"
- **version** - versión del dataset. Debería ser "v1"
- **playlists** - una lista de 10.000 listas de reproducción incompletas. Cada elemento de esta lista contiene los siguientes campos:
  - **pid** - identificador de la lista de reproducción
  - **name** - (opcional) nombre de la lista de reproducción. En algunos casos el nombre no se entrega
  - **num_holdouts** - número de canciones que han sido omitidas de la lista de reproducción
  - **tracks** - canciones que contiene la lista de reproducción, puede ser una lista vacía. Cada elemento de la lista contiene los siguientes campos:
    - **pos** - posición de la canción dentro de la lista de reproducción, empezando a contar desde 0
    - **track_name** - nombre de la canción
    - **track_uri** - URI de la canción en Spotify
    - **artist_name** - nombre del artista principal de la canción

    - **artist_uri** - URI del artista principal en Spotify
    - **album_name** - nombre del álbum al que pertenece la canción
    - **album_uri** - URI del álbum al que pertenece la canción en Spotify
    - **duration_ms** - duración de la canción en milisegundos
  - **num_samples** - número de canciones incluidas en la lista de reproducción
  - **num_tracks** - número total de canciones que tiene la lista de reproducción completa

    Hay que destacar que len(tracks) es igual a num_samples y que num_tracks es igual a num_samples más num_holdouts

1.3.2 Categorías del reto

Las 10.000 listas de reproducción se reparten en 10 categorías diferentes, con 1.000 listas de reproducción por categoría

1. Predecir canciones para una lista de reproducción dando sólo el nombre de la lista
2. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y la primera canción
3. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y las 5 primeras canciones
4. predecir canciones para una lista de reproducción dando sólo las 5 primeras canciones
5. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y las 10 primeras canciones
6. predecir canciones para una lista de reproducción dando sólo las 10 primeras canciones
7. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y las 25 primeras canciones
8. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y 25 canciones aleatorias
9. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y las 100 primeras canciones
10. predecir canciones para una lista de reproducción dando sólo el nombre de la lista y 100 canciones aleatorias

1.3.3 Cómo el dataset del reto fue contruido

Las listas de reproducción en el dataset del reto han sido seleccionadas siguiendo el mismo criterio usado para la selección del Dataset del Millón de Playlists, MPD Añadir referencia. Adicionalmente, las listas de reproducción en el dataset del reto siguen las siguientes limitaciones:

- Todas las canciones en el dataset del reto aparecen en el MPD
- Todas las canciones ocultas, dentro del holdout, aparecen en el MPD

1.3.4 Ejemplo de entrega

Ejemplo en: Añadir referencia

Este ejemplo muestra el formato esperado, debe seguir las siguientes reglas:

- Todos los campos deben estar separados por comas. Se acepta que haya espacios antes y después de las comas, pero es opcional
- Se permiten comentarios con "#" al inicio de una linea
- Se permiten lineas vacías, simplemente son ignoradas
- La primera línea no comentada/vacía debe empezar con "team_info" y después incluir el nombre del equipo, el modo en el que está participando (main o creative) y la información de contacto
- Por cada lista de reproducción debe haber una línea con exactamente 500 canciones de la siguiente forma: pid, trackuri_1, trackuri_2, ..., trackuri_499, trackuri_500
- Las canciones dadas para realizar el reto, de una lista de reproducción en particular, no deben incluirse en la entrega de esa lista de reproducción.
- La entrega para una lista de reproducción en partícula no debe tener canciones duplicadas
- La entrega para una lista de reproducción en particular debe tener exactamente 500 canciones
- Cualquier entrega que no cumpla estas normas no podrá ser puntuada

2 Desarrollo

2.1 Filtrado colaborativo

Usando solo los datos proporcionados en el MPD podemos sacar 2 conjuntos principales: Canciones y Listas de reproducción.

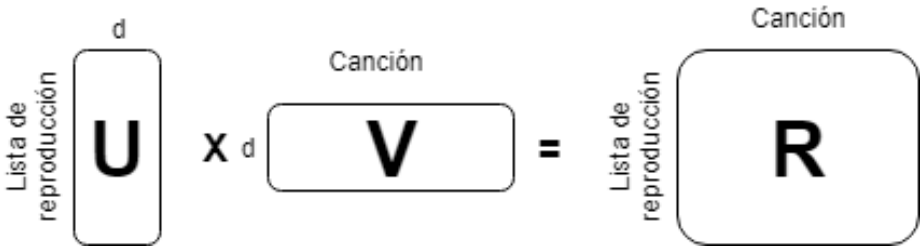

Figure 1: Usuario X lista de reproduccion

Al juntar estos dos conjuntos de datos conseguimos una matriz *UserxItem* en la que las filas representan las listas de reproducción y cada columna representa una canción. Esto nos deja una matriz cuya principal característica es que los datos están muy dispersos. Cave destacar que el significado de una matriz dispersa no se aplica de la misma forma a una matriz *UserxItem*, el significado matemático implica que el valor sea 0, en nuestro caso ese 0 significa un campo no definido o inexistente, implica que una canción no pertenece a una lista de reproducción. Por lo que una matriz dispersa no implica que haya muchos 0, implica que hay muchos campos vacíos. Llegamos a una matriz dispersa ya que cada *User*, lista de reproducción, sólo puede tener como máximo 250 canciones, teniendo en cuenta que hay 2.262.292 canciones en todo el MPD por cada lista de reproducción como mucho se usa un 0.01% del espacio.

¿Cual es el objetivo del filtrado colaborativo?

El objetivo principal es hacer recomendaciones usando una matriz dispersa valiéndose de las similitudes entre sus usuarios. La matriz tiene que ser dispersa, de no serlo no hay nada que recomendar, ya que el conocimiento sería perfecto, si todas las listas tienen todas las canciones no se podría realizar recomendaciones, es necesario que haya información oculta, si la información recibida en forma de matriz no es completa significa que el algoritmo va a poder actuar. Entonces el objetivo principal del filtrado colaborativo es calcular la probabilidad de que una canción pueda pertenecer a una lista de reproducción dada.

Todo esto está por definir, solo es una primera aproximación sobre las ideas a contar

El filtrado colaborativo asume que a los usuarios les gustan cosas similares a lo que ya les gusta, y similares a lo que le gusta a persona con gustos similares. Aplicando esto a nuestro problema, el filtrado colaborativo asume que las listas

de reproducción asimilan con más probabilidad canciones que estén en listas de reproducción parecidas.

Por ende, el filtrado colaborativo es esencial para poder crear un algoritmo de recomendación. En este trabajo vamos a tratar varios de estos algoritmos para primero ver cómo es su desempeño en nuestro problema y a partir de ahí poder desarrollar nuestro propio algoritmo buscando una mejora, ya sea en la precisión de las recomendaciones o en el tiempo de ejecución.

El primer algoritmo que hemos implementado ha sido **"user-user"**. Este algoritmo se caracteriza por buscar las relaciones directamente desde los usuarios, listas de reproducción en nuestro caso. Examina las listas de reproducción y calcula la similitud entre ellas, con jaccard, y escoge las canciones que mas se repitan entre las listas con mayor puntuación.

La segunda implementación que hemos realizado ha sido **"item-item"**. Este algoritmo se caracteriza por buscar las relaciones directamente desde los items, en nuestro caso las canciones. Examina las canciones y busca en que listas de reproducción están, calcula la similitud entre ellas y escoge las canciones que tengan mayor puntuación.

2.2 user-user

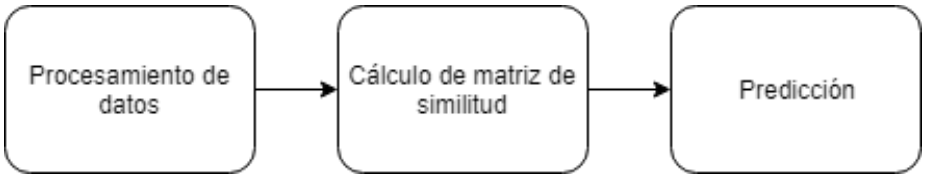

Figure 2: Flujo de ejecución user-user

Para implementar el algoritmo de filtrado colaborativo *user-user* hemos seguido el siguiente flujo de ejecución:

1. Procesamiento de datos: Para poder operar bien los datos tenemos que filtrar la información para quedarnos sólo con aquello que nos sea útil. El algoritmo recibe los datos, en formato JSON, sacados del MPD. Los datos escogidos son: el id de cada lista de reproducción (*pid*) y la URI de cada canción (*track_uri*). estos datos se almacenan en una matriz UxV, siendo U la cantidad de ids y V la cantidad de canciones únicas. Según van apareciendo URIs nuevas se les asigna un identificador, para no perder el valor de esta URI creamos un diccionario que mantiene la relación entre el nuevo identificador y la URI. Cada fila representa una lista de reproducción, y su contenido son los identificadores de las canciones que

pertenecen a cada lista. Para facilitar el acceso a estos datos y no tener que procesar los JSON en cada ejecución del algoritmo guardamos esta matriz en un archivo CSV, para ello primero lo transformamos en un DataFrama apoyándonos en la librería de Python Pandas. Este método consta de dos bucles *for* para poder recorrer todo el fichero, la complejidad temporal es de $O(n^2)$, en el caso de querer procesar todos los datos del MPD puede llegar a tardar mucho tiempo Hacer medidas, una tabla o algo