UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Sistemas de recomendación de música para grupos

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Juan Carlos Carrillo Moraga

Tutor: Laura Sebastiá Tarín

Curso 2020-2021

Resum

Els sistemes de recomanació s'utilitzen cada vegada més en diversos àmbits com el cinema, els viatges, la música, etc. Un sistema de recomanació per a grups és un sistema que recomana articles a un grup d'usuaris de manera col·lectiva. Per a això, es consideren les preferències de cadascun dels membres del grup per a oferir una recomanació global per a tots ells. Aquests sistemes, a més de les preferències dels usuaris, es fa ús d'aspectes socials i de comportament dels membres del grup per a generar recomanacions de grup que augmenten la qualitat dels continguts recomanats. Aquest treball se centra en el domini de la música per a oferir recomanacions d'una llista de cançons per a un grup d'usuaris.

Paraules clau: Sistemes de recomanació de grups, Sistemes de recomanació de música, Sistemes de recomanació, Sistemes de la informació, Recuperació de la informació

Resumen

Los sistemas de recomendación se utilizan cada vez más en diversos ámbitos como el cine, los viajes, la música, etc. Un sistema de recomendación para grupos es un sistema que recomienda artículos a un grupo de usuarios de forma colectiva. Para ello, se consideran las preferencias de cada uno de los miembros del grupo para ofrecer una recomendación global para todos ellos. Estos sistemas, además de las preferencias de los usuarios, se hace uso de aspectos sociales y de comportamiento de los miembros del grupo para generar recomendaciones de grupo que aumenten la calidad de los contenidos recomendados. Este trabajo se centra en el dominio de la música para ofrecer recomendaciones de una lista de canciones para un grupo de usuarios.

Palabras clave: Sistemas de recomendación de grupos, Sistemas de recomendación de música,Sistemas de recomendación, Sistemas de información, Recuperación de la información

Abstract

Recommender systems are increasingly used in a variety of fields such as film, travel, music, etc. A group recommender system is a system that recommends items to a group of users collectively. To do so, the preferences of each of the group members are taken into account to provide a global recommendation for all of them. These systems, in addition to user preferences, make use of social and behavioural aspects of the group members to generate group recommendations that increase the quality of the recommended content. This work focuses on the domain of music to provide recommendations of a playlist for a group of users.

Key words: Group recommendation systems, Music Recommender Systems,Recommender systems, Information systems, Information Retrieval

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

Los sistemas de recomendación resuelven el problema de la sobrecarga de información y ayudan a los usuarios a elegir entre las decisiones del día a día. Recogen las preferencias de los usuarios y generan recomendaciones adecuadas al gusto del usuario. La información sobre las preferencias de los usuarios puede adquirirse mediante valoraciones o *feedback* explícitas o implícitas.

- ***Feedback* implícito:** Las preferencias del usuario son extrapoladas en base a sus acciones como clics, búsquedas y compras. Por ejemplo, *Spotify* utiliza como métrica de éxito implícita que, si has estado durante más de 30 segundos escuchando una canción, esta es de tu agrado. Se encuentran en abundancia, pero resulta más difícil discernir cuando una valoración es negativa.

- ***Feedback* explícito:** El usuario especifica sus preferencias mediante acciones como reaccionar ante un artículo o calificarlo. Por ejemplo, en el sistema recomendador *MovieLens* puedes calificar una película con una clasificación entre una (si no te ha gustado la película) y cinco estrellas (si te ha encantado la película). Puedes especificar tus preferencias tanto positivas (lo que te gusta dando una buena valoración) como negativa (indicando lo que no te ha gustado). La cantidad de datos que se puede recopilar es menor comparado con el *feedback* implícito.

Muchos sistemas de recomendación tradicionales se han concentrado sólo en modelos de un solo usuario. Pero en la vida real, hay muchas situaciones en las que interactuamos sobre todo con grupos, como ver una película con la familia, cenar con los compañeros de trabajo, planear un viaje con los amigos, etc. Por lo que la recomendación de grupos es un problema también importante que abordar. En este trabajo veremos cómo implementar los sistemas recomendadores para grupo basándonos en los sistemas recomendadores para un solo usuario.

Los problemas que surgen al recomendar artículos a grupos, se pueden organizar en cuatro subtareas:

1. Adquirir información sobre la preferencia de los usuarios.
2. Generar recomendaciones.
3. Explicar las recomendaciones.
4. Ayudar a los miembros del grupo a tomar una decisión final.

En este trabajo nos focalizaremos en la subtarea de la generación de las recomendaciones.

El desarrollo de los sistemas de recomendación es un esfuerzo multidisciplinar en el que participan profesionales de diferentes campos, como la inteligencia artificial, la interacción persona-computador, la minería de datos, la estadística, los sistemas de apoyo a la decisión, el marketing, el comportamiento del consumidor y la psicología [41]. Por lo que en este trabajo abarcaremos diferentes campos de la computación.

1.1 Motivación

La motivación de este trabajo ha sido combinar mi pasión por la música y la informática. Actualmente los servicios de música en *streaming* se enfocan al uso de manera individual, existen pocas aplicaciones comerciales que tengan en cuenta a un grupo para escuchar música. En este trabajo trataremos de implementar esta funcionalidad con los servicios de la aplicación *Spotify*.

1.2 Objetivos

El presente trabajo está orientado a problemas de recomendación de grupos. Teniendo cierto conocimiento con respecto a los gustos y preferencias de uno o varios usuarios, el objetivo final será obtener una recomendación de agrado para el grupo.

Para ello los objetivos que tendremos que alcanzar serán:

- Conocer los principales enfoques que se utilizan a la hora de desarrollar un sistema recomendador.
- Estudiar cómo se puede desarrollar un sistema recomendador para grupos basándonos en los sistemas recomendadores para un individuo.
- Conocer los principales enfoques que se utilizan a la hora de desarrollar un sistema recomendador grupal.
- Para obtener las preferencias de los usuarios (miembros del grupo) necesitamos saber qué canciones escucha un usuario diariamente o qué canciones le gustan más. Para ello deberemos aprender a usar la libreria *Spotipy* la cual nos permitirá realizar peticiones a la API-REST de *Spotify* que nos facilitará las preferencias de los miembros del grupo.
- Un sistema recomendador necesita de datos para conocer las preferencias de sus usuarios, por lo que además deberemos aprender los fundamentos sobre la ciencia de datos. Aprender a utilizar algunas librerías como *pandas* o *sklearn* para el preprocesado de datos para estandarizarlos.
- Explotar el dataset disponible en *Kaggle*, *Spotify Dataset 1922-2021, 600k Tracks*, el cual contiene las características del audio de alrededor de 600000 canciones, para obtener las características de los artículos (canciones) a recomendar.
- Profundizar nuestra habilidad con Python.
- Desarrollar y evaluar nuestro sistema recomendador.

1.3 Estructura de la memoria

En esta sección presentaremos la organización que hemos seguido en este TFG. En primer lugar, desarrollaremos los fundamentos necesarios para comprender los sistemas de recomendación en el capítulo 2 donde:

1. Estudiaremos la recomendación para usuarios individuales
2. Presentaremos escenarios de recomendación relacionados con el filtrado colaborativo y el filtrado basado en el contenido para usuarios individuales
3. Conoceremos las diferentes estrategias de agregación que permiten utilizar la recomendación para usuarios individuales y adaptarlos al contexto de la recomendación grupal.
4. Estudiaremos la recomendación para grupos.
5. Presentaremos escenarios de recomendación relacionados con el filtrado colaborativo y el filtrado basado en el contenido para grupos
6. Discerniremos las diferentes formas de evaluar los sistemas recomendadores de grupo.

En el capítulo 3 profundizaremos en el estado del arte de los problemas que acontecen a la recomendación de música como tambíen las aplicaciones existentes en el ámbito de la recomendación de música para grupos.

A continuación, en el capítulo 4 diseñaremos y evaluaremos nuestro propio sistema de recomendación grupal.

Finalizaremos con unas conclusiones sobre el trabajo, limitaciones que hemos detectado y trabajo futuro en el capítulo 5.

CAPÍTULO 2
Fundamentos teóricos

En este capítulo revisaremos los principipales enfoques que existen a la hora de implementar un sistema de recomendación. Veremos la relación entre los sistemas recomendadores y cómo dependiendo de los factores que se tienen en cuenta a la hora de tomar una decisión, es más recomendable usar un sistema u otro. Por último, analizaremos cómo estos sistemas clásicos de recomendación, donde no se tiene en cuenta el contexto grupal, pueden ser adaptados para generar recomendaciones para un grupo

2.1 Sistemas recomendadores para usuarios individuales.

Los sistemas de recomendación son herramientas importantes que ayudan a los usuarios a conocer opciones o elementos de interés para personalizar la experiencia del usuario. Tenemos contacto con estos poderosos sistemas de recomendación a diario. Cuando disfrutamos de un video en *Youtube* o dejamos que Spotify haga una mezcla de artistas para una playlist estamos aportando elementos de personalización para que estos sistemas construyan sus recomendaciones.

A continuación, descubriremos más en profundidad qué son los sistemas de recomendación, cómo funcionan, sus principales tipos y exploraremos algunos ejemplos.

2.1.1. ¿Qué es un sistema de recomendación?

Un sistema de recomendación es una herramienta que establece un conjunto de criterios y valoraciones sobre los datos de los usuarios para realizar predicciones sobre recomendaciones de elementos que puedan ser de utilidad o valor para el usuario. Estos sistemas seleccionan datos proporcionados por el usuario de forma directa o indirecta, y procede a analizar y procesar información del historial del usuario para transformar estos datos en conocimiento de recomendación.

Los sistemas de recomendación en la actualidad tienen un nivel de eficiencia alto ya que pueden asociar elementos de nuestros perfiles de consumo como el historial de compras, selección de contenidos e inclusive nuestras horas de actividad, para realizar las recomendaciones. A continuación explicaremos los dos principales enfoques que se utilizan para desarrollar un sistema de recomendación, el basado en contenido y el filtrado colaborativo.

2.1.2. Recomendación basado en contenido

Este enfoque se basa en la suposición de intereses personales monótonos. Es decir, un usuario interesado en política no suele cambiar su ideología de un día para otro [7]. El filtrado basado en el contenido se basa en (a) un conjunto de usuarios y (b) un conjunto de categorías (o palabras clave) que se han asignado a (o extraído de) el conjunto de artículos. Compara el contenido de los artículos ya consumidos con los nuevos para encontrar los artículos que son similares a los ya consumidos (valorados positivamente) por el usuario. [8]

Pongamos un ejemplo para dilucidar todo. Supongamos que hemos obtenido los intereses de nuestros usuarios mediante nuestra plataforma ya sea por *feedback* implícito o explícito y los hemos guardado en una matriz.

Categorías	u1	u2	u3	u4	ua
Playa	X	X	X	X	-
Ciudad	X	X	X	X	X
Naturaleza	X	X	X	-	X
Entretenimiento	X	X	X	X	X

Si estamos interesados en conocer una recomendación de artículos para el usuario u_a , tenemos que buscar los artículos más parecidos a los que ya se han consumido.¿Cómo podemos cuantificar matemáticamente los artículos más parecidos? Bien hay diferentes maneras de realizarlo pero en nuestro ejemplo hemos utilizado el Coeficiente de Sorensen-Dice cuya fórmula adaptada a nuestro ejemplo es:

$$similaridad(u_a, item) = \frac{2 * |categorias(u_a) \cap categorias(item)|}{|categorias(u_a)| + |categorias(item)|} \tag{2.1}$$

Si aplicamos esta fórmula con todo el corpus de items de nuestro sistema y nos quedamos con aquellos que superan un umbral o bien con el elemento más similar habremos obtenido nuestras recomendaciones basadas en contenido. En nuestro ejemplo el ítem a recomendar seria t7 (Ciudad del Cabo) como podemos apreciar en la tabla a continuación.

Item	Rating(u_a)	Nombre	Categorías				Similaridad(u_a, t_i)
			Playa	Ciudad	Naturaleza	Entretenimiento	
t1		Viena	-	x	-	x	$\frac{4}{5}$
t2		Yellowstone	-	-	x	-	$\frac{1}{2}$
t3		Nueva York	x	x	-	x	$\frac{2}{3}$
t4	4.0	Montañas Azules	-	-	x	-	-
t5		Londres	-	x	-	x	$\frac{4}{5}$
t6	4.0	Beijing	-	x	-	x	-
t7		Ciudad del Cabo	x	x	x	x	$\frac{6}{7}$ ✓
t8		Yosemite	-	-	x	-	$\frac{1}{2}$
t9		Pittsburgh	-	x	-	x	-
user u_a				x	x	x	

Tabla 2.1: Tabla de características de articulos y su similaridad para el usuario ua.

2.1.3. Filtrado colaborativo

El filtrado colaborativo se basa en la idea del "boca a boca", donde las opiniones de familiares y amigos desempeñan un papel importante a la hora de tomar una decisión.

[4] Este tratará de recomendar elementos a usuarios con preferencias similares. Un algoritmo muy usado para lograr esta tarea es el *KNN* o de vecinos más cercanos. Las preferencias de los vecinos más cercanos (usuarios con gusto similar) se utilizan para extrapolar las valoraciones futuras de los artículos del usuario al cual queremos hacer una recomendación.

Pasemos ahora a ejemplificar el filtrado colaborativo para aclarar todos los conceptos subyacentes. Supongamos que hemos obtenido ya una serie de valoraciones de nuestro corpus de items (en nuestro caso ciudades visitadas) para cada uno de los usuarios.

Item	Nombre	u1	u2	u3	u4	ua
t1	Viena	5.0			4.0	
t2	Yellowstone	4.0				
t3	Nueva York		3.0	4.0	3.0	
t4	Montañas Azules		5.0	5.0		4.0
t5	Londres			3.0		
t6	Beijing		4.5	4.0		4.0
t7	Ciudad del Cabo	4.0				
t8	Yosemite		2.0			
t9	Paris				3.0	
t10	Pittsburgh				5.0	3.0
	Media	4.33	3.625	4.0	3.75	3.67

Si queremos encontrar dentro de nuestra red de usuarios, personas con gusto similar podemos realizarlo de muchas maneras diferentes, en nuestro caso, usaremos el coeficiente de correlación de Pearson donde TD_c es el conjunto de elementos que han sido valorados por ambos usuarios (u_a y u_x), r_{x,t_i} es la valoración del usuario x para el elemento t_i , y $\bar{r_x}$ es la valoración media del usuario x. La fórmula adaptada a nuestro caso sería:

$$similaridad(u_a, u_x) = \frac{\sum_{t_i \in I} (r_{a,t_i} - \bar{r_a}) x (r_{x,t_i} - \bar{r_x})}{\sqrt{\sum_{t_i \in I} (r_{a,t_i} - \bar{r_a})^2} * \sqrt{\sum_{t_i \in I} (r_{x,t_i} - \bar{r_x})^2}} \tag{2.2}$$

Si calculamos los coeficientes de correlación de Pearson para el usuario u_a el resultado es:

	u1	u2	u3	u4
u_a	-	0.97	0.70	-

Si el número de elementos en común es inferior a 2, asumimos que no hay similitud entre los dos usuarios, este sería el caso de u_1 y u_4. u_2 seria el usuario mas parecido a u_a.

Una vez que tenemos nuestros candidatos con gustos similares deberemos obtener los elementos a recomendar. Para ello utilizaremos el algoritmo de vecinos más cercanos. La fórmula sería: (En la formula NN *nearest neighbors*, vecinos más cercanos serían los calculados en el apartado anterior)

$$prediccion(u_a, t) = \hat{r}(u_a, t) = \bar{r_a} + \frac{\sum_{u_j \in NN} similaridad(u_a, u_j) * (r_{j,t} - \bar{r_j})}{\sum_{u_j \in NN} similaridad(u_a, u_j)} \tag{2.3}$$

Como resultado obtenemos:

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
$u2a$	-	-	3.0	5.0	-	4.5	-	2.0	-	-
u_3	-	-	4.0	5.0	3.0	4.0	-	-	-	-
u_a	-	-	-	4.0	-	4.0	-	-	-	3.0
Prediccion para u_a	-	-	3.30 ✓	-	2.66	-	-	2.04	-	-

Los destinos de viaje calificados por los vecinos más cercanos pero no valorados por u_a son t3, t5 y t8 . Debido a las predicciones calculadas, el ítem t3 estaría mejor clasificado que los ítems t5 y t8 en una lista de recomendación.

Un reto a la hora de determinar la similitud entre usuarios es la dispersión de la matriz de valoración. Los usuarios suelen proporcionar valoraciones sólo para un subconjunto muy pequeño de los elementos totales. Por ejemplo, una persona que utilice el sistema recomendador de películas MovieLens, solo valorará una ínfima parte de todas las películas disponibles puesto que no las habrá visto todas.

2.1.4. Filtrado basado en contenido vs Filtrado colaborativo

Ahora desarrollaremos las ventajas del filtrado basado en contenido respecto al colaborativo. Es necesario comentar que no hay un recomendador mejor a otro sino que dependiendo de la situación es más favorable utilizar uno u otro. [43]

1. **Independencia de los Usuarios:** Los recomendadores basados en contenido explotan únicamente las clasificaciones proporcionadas por el usuario activo para crear su propio perfil. En cambio, los métodos de filtrado colaborativo necesitan clasificaciones de otros usuarios para encontrar los "vecinos más cercanos" del usuario activo. Por ende, solo se recomendarán los artículos que más gustan a los vecinos del usuario activo.

2. **Transparencia:** Las explicaciones sobre cómo funciona el sistema de recomendación pueden proporcionarse al enumerar explícitamente las características de contenido o las descripciones que causaron que un elemento se produzca en la lista de recomendaciones. Esas características son indicadores a consultar para decidir si confiar en una recomendación. Por el contrario, los sistemas colaborativos son cajas negras ya que la única explicación para una recomendación de elemento es que a los usuarios desconocidos con gustos similares les gusta ese artículo.

3. **Ítem Nuevo:** los recomendadores basados en contenido son capaces de recomendar artículos que aún no han sido calificados por ningún usuario. Como consecuencia, no sufren el problema del primer evaluador.

Sin embargo tiene varias deficiencias como son:[43]

1. **Análisis de Contenido Limitado:** las técnicas basadas en el contenido tienen un límite natural en el número y el tipo de funciones asociadas, ya sean automáticas o manuales, con los objetos que recomiendan. El conocimiento de dominio a menudo se necesita, por ejemplo, para recomendaciones de películas, el sistema necesita conocer a los actores y directores, y algunas veces, también se necesitan ontologías de dominio. Ningún sistema de recomendación basado en el contenido puede proporcionar sugerencias adecuadas si el contenido analizado no contiene suficiente información para discriminar los elementos que le gustan a los usuarios de los que el usuario no les gustan.

2. **Sobrespecialización:** los recomendadores basados en contenido no tienen un método inherente para encontrar algo inesperado. El sistema sugiere elementos cuyas puntuaciones son altas cuando se comparan con el perfil del usuario, por lo tanto, se le recomendarán al usuario elementos similares a los que ya se calificaron. Para dar un ejemplo, cuando un usuario solo ha calificado películas dirigidas por Stanley Kubrick, se le recomendará ese tipo de películas. Una técnica basada en el contenido "perfecta" raramente encontraría algo nuevo, lo que limitaría el rango de aplicaciones para las cuales sería útil.

Las ventajas del filtrado colaborativo respecto al basado en contenido son:

1. **Serendipia:** El recomendador puede ayudar a los usuarios a descubrir nuevos intereses. De forma aislada, el sistema puede no saber que el usuario está interesado en un artículo determinado, pero podría recomendarlo porque usuarios similares están interesados en ese artículo.

2. **Útil en la recomendación de artículos complejos:** Como música, películas, fotografías...

Las principales desventajas del filtrado colaborativo son:[42]

1. **El problema de arranque en frío:** cuando no hay información disponible sobre un artículo, no podemos predecir la valoración del mismo.

2. **Dispersión de los datos:** En la práctica, muchos sistemas de recomendación comerciales se basan en conjuntos de datos de gran tamaño. Como resultado, la matriz usuario elemento utilizada para el filtrado colaborativo podría ser muy grande y dispersa, lo que provoca desafíos en la realización de la recomendación.

3. **Escalabilidad:** Cuando el número de usuarios y elementos aumentan demasiado los algoritmos tradicionales FC sufren serios problemas de escalabilidad. Por ello es frecuente utilizar el clustering junto con esta técnica para solventar esta problemática.

4. ***Shilling attacks*:** En un sistema de recomendación donde todo el mundo puede evaluar, la gente puede dar un montón de opiniones positivas para sus propios artículos y clasificaciones negativas para sus competidores.

5. **Artículo Nuevo:** Para un nuevo elemento añadido, el sistema de filtrado colaborativo no sería capaz de recomendarlo hasta que es evaluado por un número sustancial de usuarios.

2.2 Filtrado colaborativo basado en modelos.

Existen dos maneras de realizar el filtrado colaborativo, hasta ahora solo nos hemos centrado en el que se conoce como filtrado colaborativo basado en memoria. Este se realiza principalmente utilizando la matriz de interacción usuario-artículo, y cómo un usuario reacciona a ella, es decir, se basa en la calificación que un usuario da a un artículo. En este enfoque (basado en memoria) no se aplican técnicas como la reducción de la dimensionalidad ni ajustes del modelo de aprendizaje automático como tal.

Es necesario comentar en este trabajo que, existen maneras de realizar el filtrado colaborativo que ayudan a solventar el problema de la escalabilidad que ocurre cuando

tenemos millones de usuarios en nuestra plataforma o millones de artículos que recomendar, en este caso el mejor modo de proceder sería utilizar el filtrado colaborativo basado en modelos. Aquí no es necesario recordar la matriz. A partir de la matriz, intentamos aprender cómo se comporta un usuario específico o un artículo. Comprimimos la gran matriz de interacción mediante la reducción de la dimensionalidad o el uso de algoritmos de agrupación (*clustering*). En este tipo, ajustamos modelos de aprendizaje automático e intentamos predecir cuántas valoraciones dará un usuario a un producto. Existen varios métodos:

- Algoritmos de clustering
- Algoritmo basado en la factorización de matrices
- Métodos de aprendizaje profundo

2.2.1. Algoritmos de *clustering* (agrupación).

Se utilizan para agrupar datos existentes de los que desconocemos sus características en común o queremos descubrirlas. Estos métodos intentan crear "puntos centrales" y jerarquías para diferenciar grupos y descubrir características comunes por cercanía. Para ello normalmente utilizan algoritmos simples de agrupación como *KNN*para encontrar los K vecinos más cercanos basándose en una métrica de similitud utilizada. Los más utilizados son:

- **K-Medias**
- **K-Medianas**
- ***Hierarchical Clustering***

2.2.2. Algoritmo basado en la factorización de matrices.

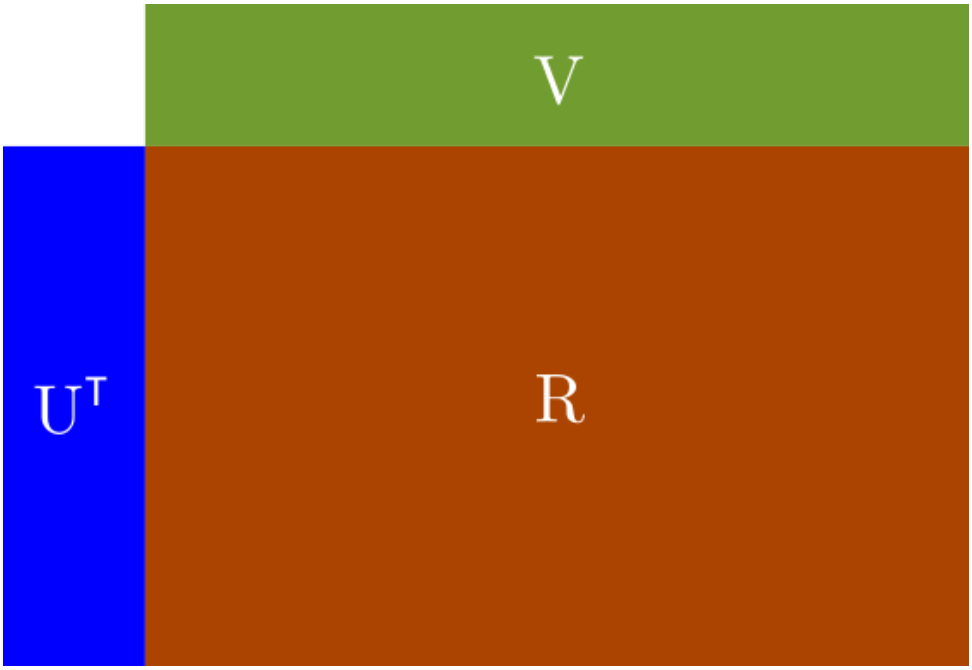
Al igual que cualquier número grande puede ser factorizado en números más pequeños, la tabla o matriz de interacción usuario-artículo también puede ser factorizada en dos matrices más pequeñas, y estas dos matrices también pueden ser utilizadas para generar de nuevo la matriz de interacción. La idea subyacente a la factorización matricial es representar a los usuarios y a los artículos en un espacio de menor dimensión.

Esta familia de métodos se hizo ampliamente conocida durante el reto del premio Netflix debido a su eficacia.

Algunos de los más utilizados y sencillos son:

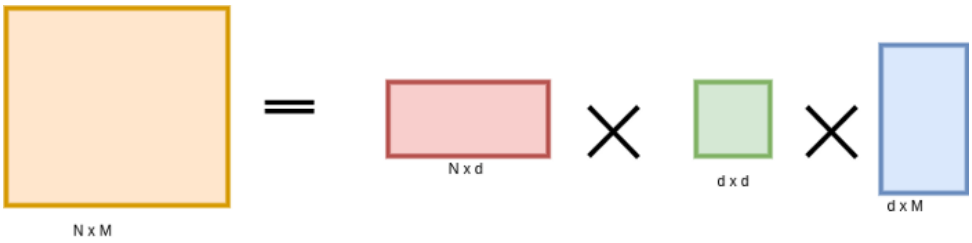
- **SVD**: Descomposición de valores singulares
- **PMF**: Factorización de la matriz de probabilidad
- **NMF**: Factorización de matrices no negativas

En el PMF la matriz se descompone de esta manera:

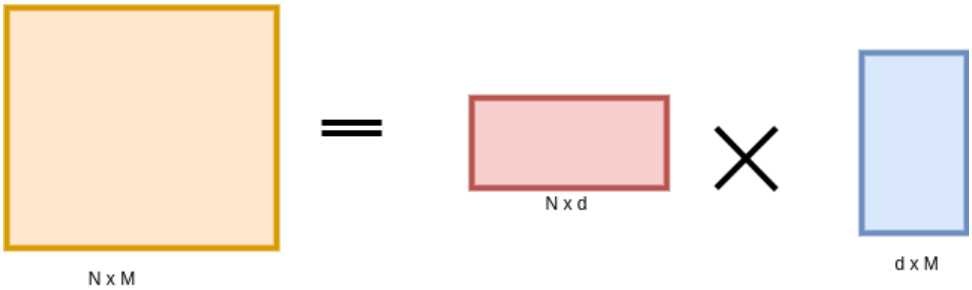


Donde U^T es una matriz NxD donde N es el número de usuarios registrados, y D es la valoración del usuario. V es una matriz DxM, donde M es el número de elementos a valorar. Por lo tanto, la matriz R de valoraciones NxM puede aproximarse mediante $R = U^T V$. Dado que U y V son matrices de bajo rango, el PMF también se conoce como un problema de factorización de matrices de bajo rango. Además, este rasgo particular de las matrices U y V hace que el PMF sea escalable incluso para conjuntos de datos que contienen millones de registros. PMF se basa en las ideas del aprendizaje bayesiano para la estimación de parámetros. En general, podemos decir que en la inferencia bayesiana, nuestro objetivo es encontrar una distribución posterior de los parámetros del modelo recurriendo a la regla de Bayes.

En el SVD la matriz se descompone de esta manera:



En el NMF la matriz se descompone de esta manera:



Donde N es el número de elementos, M es el número de usuarios y d es la dimensión o tamaño del vector de características.

2.2.3. Métodos de aprendizaje profundo.

La factorización matricial es la variante más utilizada del filtrado colaborativo. Utiliza como métrica de similitud entre otras, un producto interno (producto escalar) fijo de la matriz usuario-artículo para aprender de las interacciones usuario-artículo. De esta manera, podemos predecir artículos a usuarios basándonos en usuarios de gusto similar. El filtrado colaborativo basado en redes neuronales (NCF) sustituye el producto interno usuario-artículo por una arquitectura de red profunda. Puesto que con la factorización de matrices no se puede aprender de la interacción del usuario-artículo, el NCF trata de aprender de dichas interacciones con un perceptron multicapa.

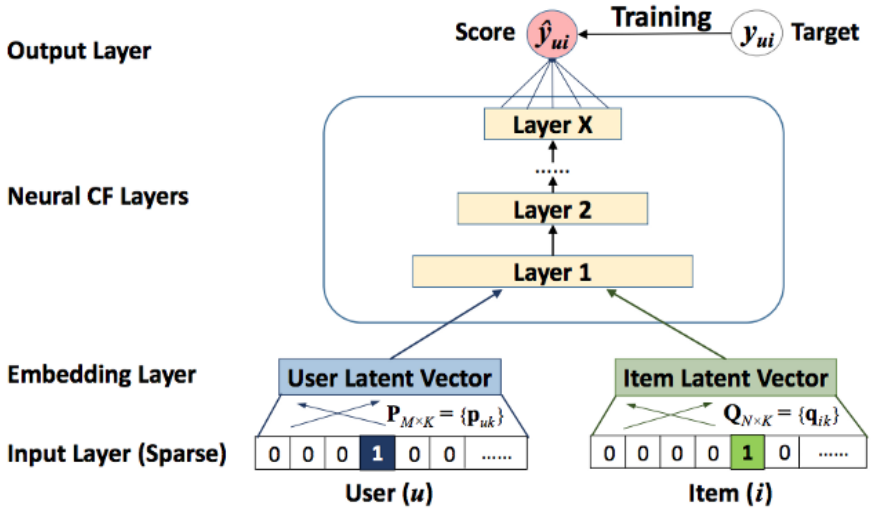


Figura 2.1: Arquitectura del sistema. [44]

La arquitectura típica de un sistema de recomendación basado en el filtrado colaborativo usando redes neuronales es el mostrado en la figura.

2.3 Estrategias para implementar los sistemas recomendadores para grupos.

En esta sección, trataremos de mostrar cómo se puede implementar la recomendación grupal sobre la base de paradigmas que hemos mencionado con anterioridad. En concreto, nos centramos en el filtrado colaborativo, el filtrado basado en el contenido, y la recomendación híbrida. A lo largo de este capítulo, diferenciamos entre (1) predicciones agregadas y (2) modelos agregados como estrategias básicas para agregar las preferencias de los miembros individuales del grupo.

2.3.1. Estrategias de agregación de preferencias.

Independientemente de la forma en que se adquieran las preferencias de los usuarios pertenecientes al grupo, una recomendación de grupo se determina agregando estas preferencias mediante dos maneras diferentes. En el primer caso, la etapa de recomendación

precede a la etapa de agregación, es decir, los artículos recomendados a los miembros individuales del grupo se agregan a una recomendación de grupo. En el segundo caso, la etapa de agregación precede a la de recomendación: los perfiles de grupo creados a partir de los perfiles de usuario individuales son la base para determinar una recomendación de grupo. Denotamos la primera estrategia de agregación como predicciones agregadas y la segunda como modelos agregados. Vease la figura 2.2 para una mejor comprensión.

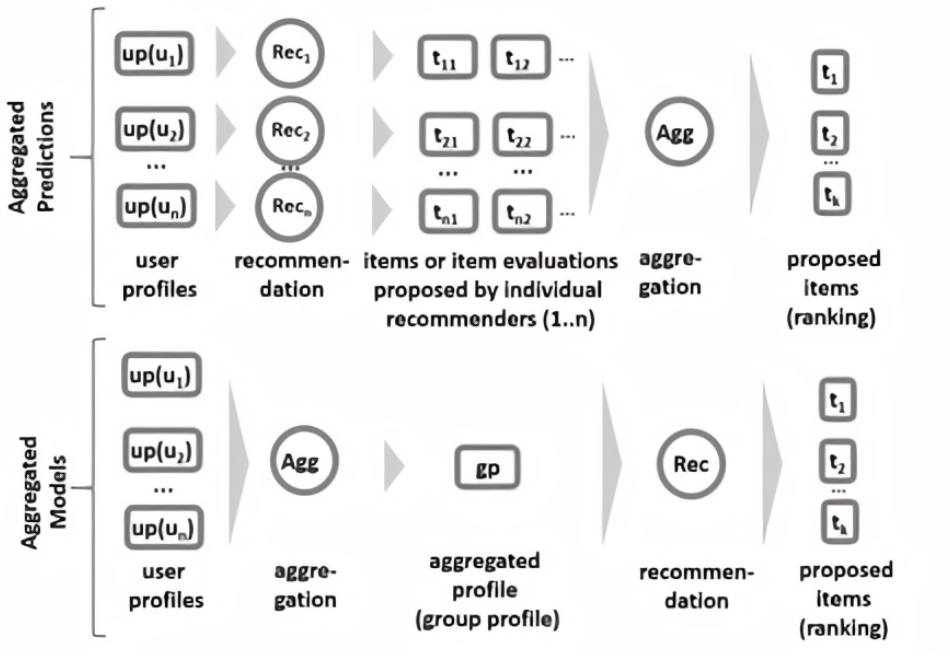


Figura 2.2: Estrategia de agregación de predicciones y de agregación del modelo. [40]

Agregación de predicciones

Dependiendo de la salida del sistema recomendador tenemos dos maneras de realizar la agregación de predicciones.

1. Obtener como salida elementos.

2. Obtener como salida una clasificación (o *Ranking*)

En primer lugar, se pueden fusionar las recomendaciones realizadas a los miembros individuales del grupo. Este enfoque puede utilizarse si se presenta un conjunto de soluciones candidatas y los miembros del grupo se encargan de seleccionar la recomendación final. Utilizando esta estrategia la salida del recomendador sería una lista de n elementos sin ordenar, puesto que se supone que el grupo decide de estos elementos cuál es mejor. Por lo tanto, los miembros del grupo desempeñan un papel importante en el proceso de toma de decisiones, ya que no se proporciona una clasificación entre las recomendaciones individuales propuestas. El esquema que seguiría este planteamiento sería el mostrado en la Figura 2.3 En segundo lugar, tendríamos como salida del sistema recomendador las predicciones (puntuación que obtiene el sistema sobre lo que estima que el usuario clasificaría) específicas de los miembros del grupo para los elementos candidatos. El resultado de este enfoque es una clasificación de los elementos recomendados o *ranking*. El esquema que seguiría este planteamiento sería el mostrado en la Figura 2.4

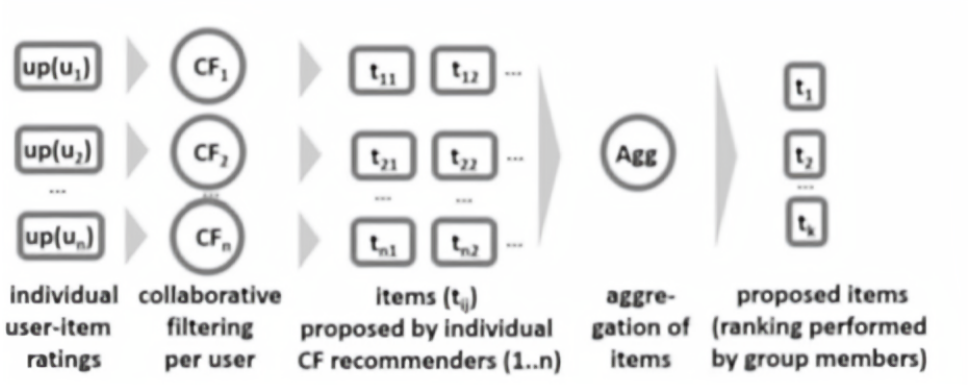


Figura 2.3: Estrategia de agregación de artículos predichos. [40]

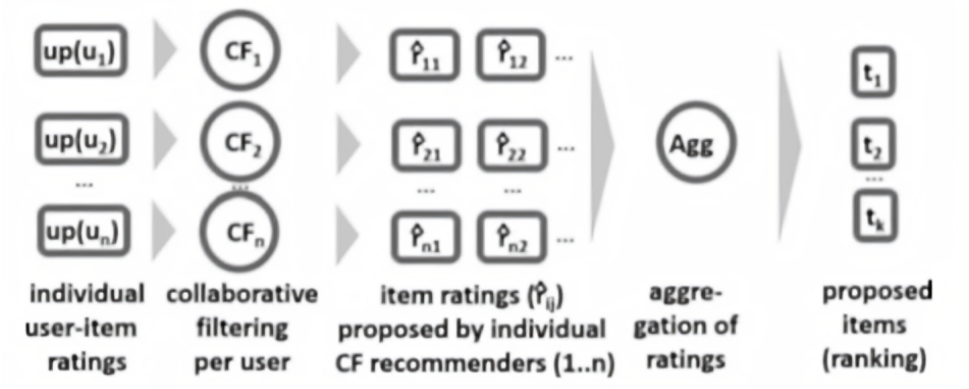


Figura 2.4: Estrategia de agregación de valoraciones. [40]

Agregación del modelo

En lugar de agregar recomendaciones para usuarios individuales, este enfoque construye un modelo de preferencias de grupo (perfil de grupo) que luego se utiliza para determinar las recomendaciones. Esto es especialmente útil en escenarios donde los miembros del grupo deben tener la oportunidad de analizar, negociar y adaptar las preferencias del grupo [38]. Otra ventaja es, que se puede aliviar la preocupación por la privacidad de los usuarios, ya que no hay necesidad específica de registrar y mantener los perfiles individuales de los usuarios.

2.3.2. Funciones de agregación de preferencias.

Un problema importante en todos los escenarios de recomendación de grupo mencionados es cómo adaptar a los miembros del grupo como conjunto, dada la información sobre las preferencias individuales [36]. Como no hay una forma óptima de agregar listas de recomendaciones, hay que utilizar una aproximación (a partir de ahora funciones de agregación) para llegar a una recomendación que tenga en cuenta, en la medida de lo posible, las preferencias individuales de los miembros del grupo. Como se menciona en [37], las funciones de agregación pueden clasificarse en:

- Basadas en la mayoría (M)
- Basadas en el consenso (C)
- Basadas en el límite (L)

En la tabla 2.2 podemos apreciar una visión general de los diferentes tipos de funciones de agregación tomadas de la teoría de la decisión social.

Las funciones de agregación basadas en la mayoría (M) llevan a cabo la agregación centrándose en los elementos más populares [18]. Ejemplos de funciones basadas en la mayoría son la votación por pluralidad (PLU) (el ganador es el elemento con el mayor número de votos), el recuento de Borda (BRC) (el ganador es el elemento con la mejor puntuación total de la clasificación), etc.

Las funciones basadas en el consenso (C) llevan a cabo la agregación teniendo en cuenta las preferencias de todos los miembros del grupo [18]. Algunos ejemplos son el *Additive Utilitarian* (el ganador es el elemento con el máximo de la suma de las evaluaciones individuales de los usuarios), el promedio (el ganador es el elemento con la media máxima de las evaluaciones individuales de los usuarios), etc.

Las funciones basadas en el límite (L) llevan a cabo la agregación teniendo en cuenta sólo un subconjunto de las preferencias de los usuarios [18]. Ejemplos de funciones límite son:

- **mínima miseria** (la recomendación elegida es aquella con la mayor puntuación de todas las puntuaciones más bajas dadas a los elementos del sistema. Cuando se utiliza esta función, pueden seleccionarse elementos que nadie odia pero tampoco le gustan a nadie especialmente)
- **el más satisfactorio** (el ganador es el artículo con la evaluación más alta de todas las evaluaciones individuales; pueden seleccionarse artículos que sólo gusten a unas pocas personas), etc.

2.4 Sistemas recomendadores para grupos.

A continuación, desarrollaremos cómo funcionan los sistemas de recomendación para grupos. Veremos cómo se puede implementar tanto el filtrado colaborativo como el basado en contenido para grupos. Para cada uno veremos cómo pueden implementarse las dos técnicas que hemos comentado en el punto anterior (agregación de predicciones y agregación de modelo).

2.4.1. Filtrado colaborativo para grupos.

Siguiendo la estrategia de agregación de predicciones.

Cuando se aplica la estrategia de agregación de valoraciones en combinación con el filtrado colaborativo, las valoraciones se determinan para los usuarios individuales y luego se agregan en una recomendación para el grupo (véase la figura 2.4). Siguiendo este enfoque, para cada miembro del grupo i y cada artículo j no valorado por este miembro del grupo, se determina una predicción de valoración $\hat{r_{ij}}$. Para simplificar, suponemos que los artículos t1...t10 de la tabla 2.3 no han sido todavía valorados por los miembros del grupo y son predicciones que han sido obtenidas por el recomendador colaborativo. Para ello además hemos supuesto que alguna variante del filtrado colaborativo ya se ha aplicado para predecir las valoraciones (por ejemplo, un enfoque de factorización matricial), para inferir las tablas de valoración de los elementos de los usuarios. A continuación, estas predicciones se agregan utilizando diferentes funciones de agregación. El resultado de la etapa de agregación es una clasificación de los elementos candidatos. En nuestro ejemplo, la mayoría de las funciones de agregación recomiendan el ítem t6. Para

Función de agregación	Descripción	Recomendación
Additive Utilitarian (ADD) [C]	Suma de las evaluaciones de cada ítem	$\underset{(t\in I)}{\text{argmax}}\ (\sum_{u\in G} eval(u,t))$
Promedio (AVG) [C]	Media de las evaluaciones de cada ítem	$\underset{(t\in I)}{\text{argmax}}(\frac{\sum_{u\in G} eval(u,t)}{\lvert G\rvert})$
Media con umbral (AVM) [C]	Media de las evaluaciones de cada ítem (si todas las evaluaciones están por encima de un umbral)	$\underset{(t\in I:\nexists u\in G \mid eval(u,t)<=threshold)}{\text{argmax}}(\frac{\sum_{u\in G} rating(u,t)}{\lvert G\rvert})$
Equidad (FAI) [C]	Clasificación de los artículos como si los individuos ($u \in G$) los eligieran uno tras otro	$\underset{(t\in I)}{\text{argmax}}(eval(u,t))$[in each iteration]
Multiplicativo (MUL) [C]	Mutiplicación de las evaluaciones de cada ítem.	$\underset{(t\in I)}{\text{argmax}}(\prod_{u\in G} eval(u,t))$
Votacion por aprobacion (APP) [M]	Numero de evaluaciones de cada item que superan un umbral.	$\underset{(t\in I)}{\text{argmax}}(\lvert \{u \in G : eval(u,t) >= umbral\} \rvert)$
Recuento de Borda (BRC) [M]	Suma de las puntuaciones de la clasificación	$\underset{(t\in I)}{\text{argmax}}(\prod_{u\in G} score(u,t))$
Regla de Copeland (COP) [M]	Número de victorias (v)-número de pérdidas (d) en la comparación de la evaluación por pares	$\underset{(t\in I)}{\text{argmax}}(\lvert v(t, I-\{t\})\rvert - \lvert d(t, I-\{t\})\rvert)$
Votación por pluralidad (PLU) [M]	El item con mayor #votos de $u \in G$	$\underset{(t\in I)}{\text{argmax}}(votings(t))$[para cada iteración]
Mínima miseria (LMS) [L]	La recomendación elegida es aquella con la mayor puntuación de todas las puntuaciones más bajas	$\underset{(t\in I)}{\text{argmax}}(mineval(t))$
Votación por mayoría (MAJ) [L]	El item con mejor evaluacion por mayoría es el recomendado.	$\underset{(t\in I)}{\text{argmax}}(majorityeval(t))$
Máxima satisfacción (MPL) [L]	El artículo con la evaluación más alta de todas las evaluaciones individuales	$\underset{(t\in I)}{\text{argmax}}(maxeval(t))$
Persona más respetada (MRP) [L]	Los ítem con mayor #votos se eligen uno tras otro.	$\underset{(t\in I)}{\text{argmax}}(votings(t))$[para cada iteración]

Tabla 2.2: Funciones de agregación para sistemas de recomendación de grupo donde argmax se asume que retorna el item a recomendar. Las letras entre corchetas M, C y L denotan las categorías de las funciones basado en la mayoría, basado en el consenso y basados en el límite; u representa un usuario (miembro del grupo), G un grupo, t un ítem e I es el corpus de Items (elementos).

calcular la función AVG es decir la media, para cada ítem cogemos las predicciones de cada usuario las sumamos y dividimos por el número de usuarios, aquel cuya media sea la más alta será el recomendado. Para el recuento de Borda o BRC simplemente sumamos la valoración de cada usuario, aquel artículo cuya suma sea la máxima sera recomendado. Para calcular la mínima miseria LMS para cada ítem nos quedamos con aquel cuya predicción sea la mínima, de este conjunto escogeremos aquel que mayor predicción tenga, ese sera el artículo a recomendar.

Una alternativa a la agregación de valoraciones es la agregación de artículos predichos donde los elementos determinados por los recomendadores individuales se agregan en una recomendación de grupo (véase la figura 2.3). Siguiendo este enfoque, los artículos con la mayor valoración prevista para un usuario específico se consideran parte de la recomendación.

Si queremos generar una recomendación compuesta por un máximo de 10 artículos, los dos artículos más valorados en la recomendación de cada miembro del grupo pueden ser incluidos en la recomendación del grupo. En el ejemplo de la tabla 2.3, t1 y t3 son los dos artículos más valorados por u1, t4 y t10 para el usuario u2....La unión de estas recomendaciones individuales representa la recomendación del grupo. De esta manera, los miembros del grupo se encargan de la clasificación de los artículos. Esta forma de construir una recomendación de grupo es similar a la idea de la función de agregación Equidad (véase la tabla 2.2).

Aparte de estos enfoques, las recomendaciones también pueden ser determinadas por la idea del *emsemble voting* donde cada función de agregación puede representar un voto.

Cuanto más votos tenga un artículo mayor será su relevancia. En nuestro ejemplo 2.3 t6 tiene el mayor número de votos por lo que sería el elemento con mayor relevancia.

Nombre Item	Descripción	Predicciones r_{ij}					Función de agregación		
		u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t1	Viena	5.0(9)	3.5(2)	1.0(0)	4.5(7)	5.0(9)	3.8	27	1.0
t2	Yellowstone	2.5(0)	4.0(4)	3.0(3)	2.0(0)	1.1(0)	2.5	7	1.1
t3	Nueva York	4.9(8)	3.8(3)	4.0(7)	3.3(4)	4.0(5)	4.0	27	3.3✓
t4	Montañas Azules	3.1(2)	5.0(9)	4.2(8)	2.4(1)	4.4(8)	3.8	28	2.4
t5	Londres	4.0(4)	4.3(7)	3.3(5)	4.1(6)	2.9(3)	3.7	25	2.9
t6	Beijing	4.5(6)	4.1(5)	5.0(9)	3.2(3)	4.2(6)	4.2✓	29✓	3.2
t7	Ciudad del Cabo	4.2(5)	4.2(6)	3.4(6)	3.1(2)	3.8(4)	3.7	23	3.1
t8	Yosemite	3.4(3)	2.6(0)	1.6(1)	5.0(9)	2.4(2)	3.0	15	1.6
t9	Paris	4.7(7)	3.1(1)	2.7(2)	3.6(5)	2.2(1)	3.3	16	2.2
t10	Pittsburgh	2.6(1)	4.5(8)	3.1(4)	4.6(8)	4.3(7)	3.8	28	2.6

Tabla 2.3: Predicciones del recomendador colaborativo y puntuación de las funciones de agregación.Entre paréntesis tenemos la valoración del usuario necesaria para calcular el recuento de Borda BRC.

Siguiendo la estrategia de agregación de modelo.

Cuando se utiliza este enfoque de agregación, las valoraciones de los usuarios individuales se agregan en un perfil de grupo gp (véase la figura 2.2). A partir del perfil de grupo (gp), el filtrado colaborativo determina una clasificación para cada artículo. En este enfoque, el grupo está representado por un perfil de grupo (gp) que incluye evaluaciones específicas de los artículos (*ratings*) derivadas a través de funciones de agregación que son aplicadas a las valoraciones de los artículos de cada miembro del grupo. A menudo, la agregación se basa en una función de media ponderada, pero pueden considerarse alternativas como las de la tabla 2.2. Siguiendo esta estrategia, se aplica el filtrado colaborativo para un perfil de grupo dado (gp), y se buscan perfiles de grupo similares (normalmente utilizando k vecinos más cercanos (*k-NN*)). Se recuperan estos vecinos o perfiles de grupo similares y se utilizan para determinar una recomendación. En nuestro ejemplo, el artículo t2 (Yellowstone) no es conocido por el grupo actual gp, pero ha recibido las valoraciones más altas de los grupos de "vecinos" más cercanos gx y gy (véase la tabla 2.4), lo que lo convierte en un candidato de recomendación para gp.

La similitud entre el perfil de grupo gp y otro perfil de grupo gx (el vecino más cercano) puede determinarse, por ejemplo, mediante el coeficiente de correlación de Pearson (comentados en el filtrado colaborativo para usuarios individuales). La fórmula 2.4 es una versión adaptada que determina la similitud entre el perfil de un grupo y los perfiles de otros grupos. En la fórmula, TD_c representa el conjunto de elementos que han sido calificados por ambos grupos (gp y gx), r_{gx,t_i} es la valoración del grupo gx para el artículo ti, y $\bar{r_{gx}}$ es la valoración media del grupo gx.

$$similaridad(gp, gx) = \frac{\sum_{t_i \in TD_c} (r_{gp,t_i} - \bar{r_{gp}}) \times (r_{gx,t_i} - \bar{r_{gx}})}{\sqrt{\sum_{t_i \in TD_c} (r_{gp,t_i} - \bar{r_{gp}})^2} \times \sqrt{\sum_{t_i \in TD_c} (r_{gx,t_i} - \bar{r_{gx}})^2}} \tag{2.4}$$

La información sobre grupos con un comportamiento similar (tienen valoraciones similares) comparado con el grupo gp es la base para predecir la valoración de un articulo

que todavía no ha sido valorado por los miembros del grupo gp (véase la formula 2.5)

$$prediccion(gp,t) = \hat{r}(gp,t) = r^{-}_{gp} + \frac{\sum_{gj \in NN} similaridad(gp,gj) \times (r_{gj,t} - r^{-}_{gj})}{\sum_{gj \in NN} similaridad(gp,gj)} \quad (2.5)$$

Item	Nombre	gp	$gx \in NN$	$gy \in NN$	Prediccion
t1	Viena	5.0	5.0	4	-
t2	Yellowstone	-	4.0	4.5	4.49 ✓
t3	Nueva York	4.0	3.0	3.5	-
t4	Montañas Azules	-	4.5	4	4.44
t5	Londres	4.0	3.9	3.5	-
t6	Beijing	-	3.5	3	3.44
t7	Ciudad del Cabo	-	4.7	3	3.99
t8	Yosemite	3.0	3.8	3.2	-
t9	Paris	4.0	3.9	2.9	-
t10	Pittsburgh	-	5.0	3.3	4.28
Media		4.0	4.13	3.5	-

Tabla 2.4: Filtrado colaborativo aplicado a perfiles de grupo.

2.4.2. Recomendación basada en contenido para grupos.

Siguiendo la estrategia de agregación de predicciones.

Cuando se utiliza esta estrategia de agregación, los recomendadores individuales basados en contenidos determinan la similitud entre (a) los artículos no consumidos por el usuario y (b) su perfil de usuario. Las similitudes de los artículos identificados se agregan y forman así la base de una recomendación de grupo (véase la figura 2.5).

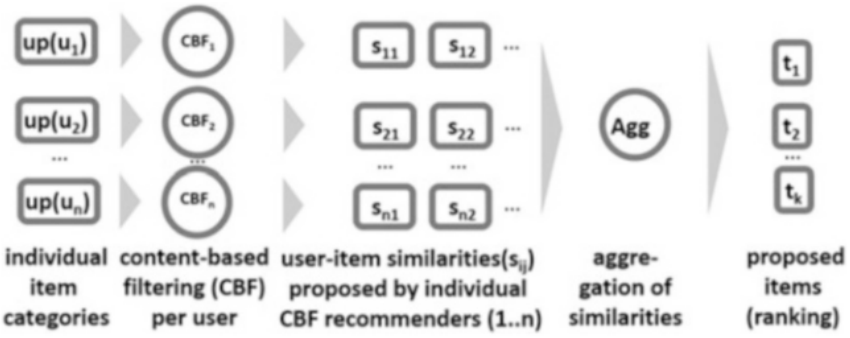


Figura 2.5: Filtrado basado en contenido para grupos usando predicciones agregadas. S_{ij} indica la similaridad entre un usuario i y un articulo j.

La tabla 2.5 muestra ejemplos de perfiles de los miembros del grupo u1...u5 Para cada uno de estos perfiles, se determina la similitud con los artículos incluidos en la tabla 2.1 (suponemos que estos artículos no han sido consumidos/evaluados por los miembros del grupo). Estos valores de similitud son la base para realizar una recomendación de grupo (véase la tabla 2.6).

Las similitudes usuario-artículo de la tabla 2.6 son calculadas por un recomendador basado en el contenido (la métrica de similitud que utilizamos es la misma que utilizamos

para calcular la recomendación individual, la fórmula de Sorensen-Dice 2.1). El cálculo se basa en las categorías incluidas en la Tabla 2.1, es decir: playa, ciudad, naturaleza y entretenimiento. Por ejemplo, $similaridad(u1, t2) = \frac{2 \times |categorias(u1) \cap categorias(t2)|}{|categorias(u1)| + |categorias(t2)|} = \frac{2}{3} = 0{,}66$

Categorías	Miembros del grupo				
	u1	u2	u3	u4	u5
Playa	X	X	X	X	X
Ciudad	-	X	-	X	-
Naturaleza	X	-	X	-	X
Entretenimiento	-	-	-	-	-

Tabla 2.5: Tabla de preferencias de los miembros del grupo.

Artículo	Nombre	Puntuaciones usuario-artículo (BRC)					Función de agregación		
		u1	u2	u3	u4	u5	AVG	BRC	LMS
t1	Viena	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t2	Yellowstone	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t3	Nueva York	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t4	Montañas Azules	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t5	Londres	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t6	Beijing	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t7	Ciudad del Cabo	0.66(7.5)	0.66(8.5)	0.66(7.5)	0.66(8.5)	0.66(7.5)	0.66 ✓	39.5 ✓	0.66 ✓
t8	Yosemite	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66/7.5)	0.4	24.5	0
t9	Paris	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t10	Pittsburgh	0(2.5)	0.66(8.5)	0(2.5)	0.66(8.5)	0(2.5)	0.26	24.5	0

Tabla 2.6: Similaridad usuario-artículo. Entre paréntesis tenemos la valoración del usuario necesaria para calcular el recuento de Borda BRC.

Teniendo la matriz de similitud usuario-artículos, las funciones de agregación pueden determinar una recomendación de grupo. Una alternativa a la agregación de similitudes sería agregar los artículos propuestos por los recomendadores individuales basados en el contenido. Si queremos generar una recomendación compuesta, por ejemplo, por un máximo de 5 artículos, el artículo mejor valorado de cada miembro del grupo se incluiría en la recomendación del grupo. En nuestro ejemplo de la Tabla 2.6, t2 se encuentra entre los artículos mejor valorados del usuario u1 (los otros tres están excluidos debido al límite de un artículo por usuario ya que queremos 5 recomendaciones), t7 puede ser seleccionado por el usuario u2, t4 para el usuario u3, t10 para el usuario u4 y t2 para el usuario u5. La recomendación del grupo sería $\{t2, t4, t7, t10\}$.

Siguiendo la estrategia de agregación de modelo

Cuando se utiliza esta estrategia, las categorías preferidas de los usuarios individuales se integran en un perfil de grupo gp. A partir de ahí, el filtrado basado en el contenido determina las recomendaciones calculando las similitudes entre gp y los artículos no consumidos por el grupo (véase la figura 2.5). En nuestro ejemplo (véase la tabla 2.5), el perfil de grupo derivado se representa por la unión de las categorías de los perfiles individuales de los usuarios. Se recomiendan los artículos que son similares a las categorías del perfil del grupo y que todavía no han sido consumidos por los miembros del grupo. En nuestro ejemplo, el perfil de grupo derivado gp incluye las categorías playa, ciudad y naturaleza.

La similitud entre el perfil de grupo gp y los artículos candidatos puede determinarse mediante la fórmula 2.6, que es una adaptación de la fórmula 2.1 a los conjuntos de grupos. Las similitudes entre gp y los elementos ti (tomados de nuestro conjunto de

elementos de ejemplo que se muestran en la Tabla 2.1) se determinan comparando las categorías playa, ciudad, naturaleza y ocio (véase la tabla 2.7).

$$similaridad(gp, item) = \frac{2 * |categorias(gp) \cap categorias(item)|}{|categorias(gp)| + |categorias(item)|} \tag{2.6}$$

Artículo	Nombre	Similaridad(gp,ti)
t1	Viena	$\frac{2}{5} = 0{,}4$
t2	Yellowstone	$\frac{2}{4} = 0{,}5$
t3	Nueva York	$\frac{2}{5} = 0{,}4$
t4	Montañas Azules	$\frac{2}{4} = 0{,}5$
t5	Londres	$\frac{2}{5} = 0{,}4$
t6	Beijing	$\frac{2}{5} = 0{,}4$
t7	Ciudad del Cabo	$\frac{6}{7} = 0{,}86$✓
t8	Yosemite	$\frac{2}{4} = 0{,}5$
t9	Paris	$\frac{2}{5} = 0{,}4$
t10	Pittsburgh	$\frac{2}{4} = 0{,}5$

Tabla 2.7: Similaridad articulo-perfil de grupo.

2.5 Evaluación del sistema recomendador

En los capítulos anteriores hemos aprendido a diseñar sistemas de recomendación de grupos, pero todavía no hemos desarrollado la cuestión sobre cómo evaluarlos. Las técnicas de evaluación de los sistemas de recomendación de grupo suelen ser las mismas o similares a las que se utilizan para los recomendadores de un solo usuario. En esta sección mostraremos cómo aplicar estas técnicas basándonos en ejemplos e introducimos enfoques de evaluación que son específicamente útiles en escenarios de recomendación de grupo.

La evaluación de un sistema de recomendación de grupos está intrínsecamente relacionado con la evaluación usada para los sistemas recomendadores de un solo usuario. [20] Distinguimos entre dos tipos de protocolos de evaluación:*offline* o *online*.

Evaluación *offline*: se basa en la idea de estimar la calidad de la predicción de un algoritmo usando datasets que incluyen evaluaciones *usuario x artículo (ratings)*. Estos dataset normalmente se dividen en set de entrenamiento y set de testing, con una proporción de los datos del 80 % para el de entrenamiento y un 20 % para el de test. Puesto que los *dataset* suelen derivarse de sistemas de recomendación para usuarios individuales, los *dataset* para grupos tienen que ser sintetizados para que sean aplicables a la evaluación de recomendadores de grupos.

Evaluación *online*: trata de evaluar los sistemas de recomendación mediante un método denominado *A/B texting* , en el que una parte de los usuarios son atendidos por el sistema de recomendación A y otra parte por el sistema de recomendación B. El sistema de recomendación que obtiene una mayor puntuación según una métrica elegida (por ejemplo, la tasa de clics) es elegido como mejor sistema de recomendación.

Independientemente del protocolo que utilicemos es frecuente también realizar un análisis cuantitativo *post hoc* para identificar diferencias entre interfaces, algoritmos y sistemas.

A continuación dividiremos las métricas de evaluación en torno a tres objetivos, (1) métricas de clasificación que evalúan hasta que punto un recomendador es capaz de fa-

cilitar artículos de relevancia para un usuario, (2) métricas de error que evalúan lo bien que un recomendador realiza predicciones (puntuación estimada sobre un artículo) y (3) métricas de *ranking* que evalúa qué tan bueno es el *ranking* que da como salida.

2.5.1. Métricas de clasificación

Podría decirse que, las métricas de clasificación más comúnmente usadas son *precision y recall*. Estas métricas suelen aplicarse en la evaluación *offline* donde los algoritmos de recomendación se entrenan utilizando una parte de los datos disponibles con fines de aprendizaje y luego se evalúan comparando las predicciones con una parte de los datos de test. A continuación, explicaremos brevemente el uso de las métricas de *precision y recall* aplicados en la recomendación de grupos.

Supongamos que tenemos una matriz donde tenemos guardadas las valoraciones de los articulos que han realizado los usuarios pertenecientes a los grupos y además las predicciones que el recomendador obtiene sobre los artículos.

Grupos	Miembros del grupo	Rating r(u_i, t_j)			Prediccion $\hat{r}(u_i, t_j)$		
		t1	t2	...	t1	t2	...
g1	u1	4.5	2.5	...	3.4	3.8	...
	u2	3.5	4.5	...	3.7	4.4	...
	u3	4.5	4.0	...	4.4	3.9	...
g2	u4	3.5	2.5	...	3.8	2.6	...
	u5	4.0	4.5	...	3.7	4.4	...
	u6	4.5	3.5	...	4.5	3.7	...
g3	u7	4.5	3.5	...	4.5	3.7	...
	u8	3.5	3.92.5	...	3.7	4.4	...
	u9	4.0	3.5	...	4.4	3.9	...
...	...	...	...	...	...		...

Tabla 2.8: Valoraciones (ratings) y predicciones para los artículos t1 y t2.

Estas valoraciones son individuales y para poder tener las métricas de grupo debemos obtener los *rating* grupales para los artículos por lo que obtenemos la media (por simplicidad en los cálculos) y obtenemos la siguiente tabla.

Grupos	Ratings $r(g_i, t_j)$		Predicciones $\hat{r}(g_i, t_j)$	
	t1	t2	t1	t2
g1	4.2	3.7	3.8	4.0
g2	4.0	3.5	4.0	3.6
g3	4.0	3.2	3.8	4.0

Tabla 2.9: Ejemplo de set de test: donde $r(g,t) = \frac{\sum_{u \in g} r(u,t)}{|g|}$ y $\hat{r}(g,t) = \frac{\sum_{u \in g} \hat{r}(u,t)}{|g|}$

Teniendo como base estas tablas vamos a proceder a aplicar las métricas de *precision y recall*. La *precision* es la fracción del número de elementos recomendados relevantes (verdaderos positivos) en relación con el número total de elementos recomendados. El *recall* es la fracción del número de elementos recomendados relevantes en relación con el número de todos los elementos relevantes. Ambas métricas suelen expresarse a un determinado nivel k, donde k es la longitud de la lista de elementos recomendados. Por ejemplo, la *precisión*@1 = 1 indica que se recomendó un elemento y que este se consideró

relevante. Si por ejemplo, la *precisión*@2 = 0.5 lo que nos indica es que de una lista de dos elementos recomendados sólo uno se considera una recomendación relevante (verdadero positivo).

La precisión de un recomendador de grupos que recomienda k artículos a un grupo g puede definirse de la siguiente manera.

$$precision@k(g) = \frac{|predichos_k(g) \cap relevantes(g)|}{k} \tag{2.7}$$

Y el *recall*:

$$recall@k(g) = \frac{|predichos_k(g) \cap relevantes(g)|}{relevantes(g)} \tag{2.8}$$

Donde $predichos_k(g)$ denota una lista de k artículos recomendados al grupo g y $relevantes(g)$ representa todos los artículos relevantes para g.

Por lo que si aplicamos las métricas con los datos anteriores obtendríamos:

Grupos	Predichos	Relevantes	precision@2	recall@2
g1	2	2	1.0	1.0
g2	2	1	0.5	1.0
g3	2	1	0.5	1.0
en general	6	4	0.67	1.0

En nuestro ejemplo, definimos un artículo como relevante si el rating del grupo es superior a 3.5 (umbral de relevancia). Por ejemplo, precision@2(g1) = 1 puesto que ambos artículos tienen un rating >3.5. La fila en la tabla de en general nos indica como es nuestro recomendador de bueno en su conjunto total. Puesto que 6 elementos han sido predichos en total pero tan solo 4 han sido relevantes obtenemos una precisión de nuestro recomendador de 0.67.

La *precision y el recall*pueden utilizarse tanto en las predicciones agregadas como en los modelos agregados basados en recomendadores de grupo, ya que ambas métricas de evaluación se aplican al resultado de la recomendación, es decir, son independientes del enfoque de agregación que hayamos utilizado. Lo mismo ocurre con la recomendación basada en el contenido, colaborativo, basado en críticas... En estos escenarios, el recomendador de grupo determina una evaluación global del artículo (similitud entre usuario y el artículo en el filtrado basado en el contenido) que se utiliza para estimar la relevancia del artículo. En consecuencia, se puede aplicar un umbral como el que hemos utilizado en nuestro ejemplo.

2.5.2. Métricas de error

Las métricas de error pueden utilizarse para medir el error cometido por un sistema de recomendación para predecir la valoración de un artículo (predicciones). El supuesto subyacente es que cuanto menor sea el error, mejor será el algoritmo evaluado. Un medio básico para medir los errores de predicción es el error medio absoluto (MAE).

$$MAE(g) = \frac{\sum_{r(g,t) \in R_g} |r(g,t) - \hat{r}(g,t)|}{|R_g|} \tag{2.9}$$

Donde R_g es el conjunto de valoraciones del grupo g que están en el conjunto de test.

Grupos	MAE
g1	$\frac{0,4+0,3}{2} = 0,35$
g2	$\frac{0,0+0,1}{2} = 0,05$
g3	$\frac{0,2+0,8}{2} = 0,5$
en general	0.3

Al igual que con la *precision y el recall*, el error medio absoluto (MAE) puede utilizarse en el contexto de las predicciones agregadas predicciones y modelos agregados basados en recomendadores de grupo. Teniendo una función que estime las valoraciones de los elementos de los usuarios, esta métrica también puede aplicarse en la recomendación basada en el contenido, entre otras.

2.5.3. Métricas de *ranking*

Las métricas dependientes del ranking no sólo tienen en cuenta la relevancia del elemento, sino también la posición del artículo en la lista de recomendaciones. Un ejemplo de este tipo de métrica es la ganancia acumulada descontada (DCG), que se basa en la idea de que los elementos que aparecen más abajo en la lista de una recomendación deben ser penalizados disminuyendo los valores de relevancia logarítmicamente.

$$DCG@k(g) = \sum_{i=1...k} \frac{2^{relevancia(t_i,g)} - 1}{log_2(1+i)} \tag{2.10}$$

Donde k denota el numero de artículos a recomendar y $relevancia(t_i, g)$ devuelve 1 si el artículo t_i (en la posición i) es relevante para el grupo g, 0 en cualquier otro caso.

Grupos	relevancia		DCG@k
	t1	t2	
g1	1	1	$\frac{1}{1} + \frac{1}{6} = 1,625$
g2	1	0	$\frac{1}{1} + \frac{0}{1,6} = 1$
g3	1	0	$\frac{1}{1} + \frac{0}{1,6} = 1$
En general			1.21

En nuestro ejemplo para aplicar la fórmula miraríamos en la tabla 2.9 y comprobaríamos si $r(g, t) > 3,5$ relevancia = 1, en cualquier otro caso, relevancia = 0.

Si la longitud de las listas de recomendación de los grupos varía, es decir, no hay una k fija que refleje el número de elementos recomendados debemos normalizar la DCG. Para ello estimamos una DCG "ideal" con la siguiente fórmula.

$$iDCG@k = \sum_{i=1...k} \frac{1}{log_2(1+i)} \tag{2.11}$$

El valor resultante es el utilizado para normalizar DCG.

$$nDCG@k(g) = \sum_{i=1...k} \frac{DCG@k(g)}{iDCG@k} \tag{2.12}$$

Cuando evaluamos secuencias de elementos recomendados, no solo la posición del artículo, sino también la posición del artículo respecto a la posición que el grupo hace sobre el artículo respecto a su relevancia. Con esto en mente, lo que podemos realizar es comparar la posición de los artículos en la secuencia con la secuencia elegida por un

grupo. Este tipo de evaluación puede realizarse, por ejemplo, usando el Coeficiente de correlación de rango de Kendall.

$$\tau(g) = \frac{|numerodeparesconcordantes| - |numerodeparesdiscordantes|}{k \times \frac{(k-1)}{2}} \tag{2.13}$$

2.5.4. Alcance y Serendipia.

En el contexto de los sistemas de recomendación, el alcance puede ser considerado desde diferentes puntos de vista. *Alcance de usuario* puede ser descrito como el número de usuarios para los cuales al menos puede ser realizada una recomendación. *Alcande de grupo* (GC) puede ser descrito como el número de grupos para los cuales al menos puede realizarse una recomendación grupal. Una recomendación grupal no podrá ser realizada en determinadas situaciones, por ejemplo, cuando el *rating* agregado esté por debajo de cierto umbral como ocurre en el filtrado colaborativo.

$$GC = \frac{|gruposconprediccion|}{|grupos|} \tag{2.14}$$

La serendipia en el contexto de los sistemas recomendadores puede ser definida como el hallazgo valioso para el usuario que se produce de manera inesperada. Una métrica para la serendipia sería la propuesta por M. Ge [22], mostrada a continuación, donde RS(g) es el conjunto de recomendaciones útiles para el grupo y PM(g) es un conjunto de recomendaciones generadas por un sistema "primitivo" (por ejemplo el basado en artículos populares).

$$SER(g) = \frac{|RS(g) - PM(g)|}{|RS(g)|} \tag{2.15}$$

2.5.5. Consenso y equidad.

El consenso como evaluación puede ser considerado como una métrica que evalúa hasta qué punto los miembros del grupo han llegado a un acuerdo con sus preferencias sobre los artículos. En el filtrado colaborativo podemos definir la métrica del consenso mediante la diferencia entre pares de (puntuaciones *ratings*) donde $r(u_i, t)$ seria la puntuación de un articulo t para el usuario u_i perteneciente al grupo g. r_{max} el máximo rating posible se utiliza como factor para la normalización.

$$consenso(g,t) = 1 - \frac{\sum\limits_{(u_i,u_j) \in g : i \neq j} |r(u_i,t) - r(u_j,t)|}{|g| \times (|g|-1)/2 \times r_{max}} \tag{2.16}$$

Puesto que los sistemas de recomendación de grupo implican los intereses de varias partes es interesante tener una métrica que cuantifique como de equitativo está siendo el sistema al recomendar. Si recomendamos artículos por filtrado colaborativo, la equidad puede ser definida como el conjunto de artículos que superan cierto umbral.

$$equidad(g,t) = \frac{|\bigcup\limits_{u \in g} : r(u,t) > umbral|}{|g|} \tag{2.17}$$

CAPÍTULO 3
Estado de la cuestión.

En este capítulo vamos a revisar algunas aplicaciones realizadas para sistemas recomendadores de musica para grupos. Después veremos dos de los principales desafios que hay actualmente en la recomendación de música, el problema del arranque en frío y la generacíon automática de listas de reproducción.

3.1 Particularidades de la recomendación de musica.

Schedl, M., Zamani, H., Chen, CW. entre otros [17] destacaron los principales aspectos que hacen de los sistemas de recomendacion de música (SRM) una tarea particular y que la distinguen de la recomendación de otros artículos, como películas, libros o productos.

Duración de los ítems: En la recomendación tradicional de películas, los ítems de interés tienen una duración típica de 90 minutos o más. En la recomendación de libros, el tiempo de consumo suele ser incluso mucho mayor. Por el contrario, la duración de los elementos musicales suele oscilar entre 3 y 5 minutos (excepto quizás para la música clásica). Por ello, los artículos musicales pueden considerarse más desechables.

Magnitud de los artículos: El tamaño de los catálogos comerciales de música más comunes es de decenas de millones de piezas musicales, mientras que los servicios de *streaming* de películas tienen que lidiar con catálogos mucho más pequeños, normalmente de miles a decenas de miles de películas y series. La escalabilidad es, por tanto, un problema mucho más importante en la recomendación de música que en la de películas.

Consumo secuencial: A diferencia de las películas, las piezas musicales suelen consumirse de forma secuencial, es decir, en una sesión de escucha o lista de reproducción. Esto plantea una serie de retos para un SRM, relacionados con la identificación de la disposición correcta de los elementos en una lista de recomendaciones.

Recomendación de elementos previamente recomendados: Recomendar la misma pieza musical de nuevo, en un momento posterior, puede ser apreciado por el usuario de un SRM, en contraste con un recomendador de películas o productos, donde las recomendaciones repetidas no suelen ser preferidas.

Comportamiento de consumo: La música suele consumirse de forma pasiva, en segundo plano. Aunque esto no es un problema en sí mismo, puede afectar a la obtención de preferencias. En particular, cuando se utiliza la retroalimentación implícita para inferir las preferencias del oyente, el hecho de que un oyente no esté prestando atención a la música (por lo tanto, por ejemplo, no se salta una canción) podría interpretarse erróneamente.

Intención y finalidad de la escucha: La música tiene diversas finalidades para las personas y, por tanto, determina su intención de escucharla. Esto debe tenerse en cuenta a la hora de crear un SRM. Schäfer entre otros [39] destilaron tres intenciones fundamentales a la hora de escuchar música: autoconciencia, relación social y regulación del estado de ánimo. La autoconciencia "ayuda a las personas a pensar en quiénes son, en quiénes les gustaría ser y en cómo trazar su propio camino". La relación social describe el uso de la música para sentirse cerca de los amigos y para expresar la identidad y los valores a los demás. La regulación del estado de ánimo tiene que ver con la gestión de las emociones, que es una cuestión fundamental para el bienestar de los seres humanos. De hecho, varios estudios han descubierto que la regulación del estado de ánimo y de las emociones es el propósito más importante por el que las personas escuchan música , por lo que a continuación analizaremos por separado el papel particular que desempeñan las emociones al escuchar música.

Emociones Se sabe que la música evoca emociones muy fuertes. Pero se trata de una relación mutua, ya que también las emociones de los usuarios afectan a las preferencias musicales. Debido a esta fuerte relación entre la música y las emociones, el problema de describir automáticamente la música en términos para describir emociones es un área de investigación activa, comúnmente denominada reconocimiento de emociones musicales (REM). Aunque el REM puede usarse para etiquetar la música por términos de emoción, Integrar esta información en el SRM es una tarea muy complicada, por tres razones. En primer lugar, los enfoques del REM suelen descuidar la distinción entre la emoción deseada (es decir, la emoción que el compositor, el autor de la canción tenía en mente cuando creó o interpretó la pieza), la emoción percibida (es decir, la emoción que se reconoce mientras se escucha) y la emoción inducida que siente el oyente. En segundo lugar, la preferencia por un determinado tipo de pieza musical cargada de emociones depende de si el usuario quiere mejorar o modular su estado de ánimo. En tercer lugar, los cambios emocionales suelen producirse dentro de la misma pieza musical, mientras que las etiquetas suelen extraerse para toda la pieza. Por lo tanto, para que la música y las etiquetas coincidan en términos de emociones, es necesario modelar la preferencia musical del oyente como una función dependiente del tiempo de sus experiencias emocionales, teniendo en cuenta también el propósito previsto (mejora o regulación del estado de ánimo). Se trata de una tarea muy difícil y que normalmente se descuida, por ello implementar un sistema recomendador consciente de las emociones sería una de las principales direcciones futuras en la investigación.

Contexto de escucha: Los aspectos situacionales o contextuales tienen una gran influencia en las preferencias musicales, el consumo y el comportamiento de interacción. Por ejemplo, es probable que un oyente cree una lista de reproducción diferente cuando se prepara para una cena romántica que cuando se prepara con los amigos para salir un viernes por la noche. Los tipos de contexto que se consideran con más frecuencia son la ubicación (por ejemplo, escuchar en el lugar de trabajo, en los desplazamientos al trabajo o mientras se descansa en casa) y el tiempo (que se suele clasificar, por ejemplo, en mañana, tarde y noche). Además, el contexto también puede estar relacionado con la actividad del oyente o el uso de diferentes dispositivos de escucha, por ejemplo, tapones en un *smartphone* frente a un equipo de música de alta fidelidad en casa, por nombrar algunos. Dado que la escucha de música es también una actividad muy social, investigar el contexto social de los oyentes es crucial para entender sus preferencias y su comportamiento. La importancia de tener en cuenta estos factores contextuales en la investigación se conoce como SRM contextuales como una dirección de investigación en tendencia.

3.2 Problema del arranque en frío.

Uno de los principales problemas de los sistemas recomendadores en general y de los sistemas de música en particular es el problema del arranque en frío, es decir cuando un nuevo usuario se registra en el sistema o se añade un artículo al catálogo, el sistema no tiene suficientes datos asociados a estos artículos/usuarios. En este caso, el sistema no puede recomendar correctamente los artículos existentes a un nuevo usuario (problema del nuevo usuario) o recomendar un nuevo artículo a los usuarios existentes (problema del nuevo artículo).

Otro subproblema del arranque en frío es el problema de la dispersión que se refiere al hecho de que el número de valoraciones dadas es mucho menor que el número de valoraciones posibles, hecho probable cuando el número de usuarios y artículos es grande. Una alta dispersión se traduce en una baja cobertura, ya que la mayoría de los usuarios tienden a calificar sólo una pequeña fracción de artículos. Esto tiene como consecuencia que las recomendaciones suelen ser poco fiables [24]. Los valores típicos de dispersión se acercan bastante al 100 % en la mayoría de los sistemas de recomendación del mundo real. En el ámbito de la música, éste es un problema especialmente importante, por ejemplo, el Dror G, entre otros [25], analizaron el conjunto de datos de *Yahoo*. En dicho documento, informan de una dispersión del 99,96 %. En comparación, el conjunto de datos de películas de *Netflix* tiene una dispersión de "sólo" el 98,82 %.

3.2.1. Soluciones actuales.

Ya se han propuesto varios enfoques para abordar el problema del arranque en frío en el ámbito de la música, sobre todo enfoques basados en el contenido, la hibridación, la recomendación entre dominios y el aprendizaje activo.

Los algoritmos de recomendación basados en el contenido no requieren valoraciones de usuarios distintos del usuario objetivo para obtener recomendaciones. Por tanto, siempre que se disponga de algunos datos sobre las preferencias del usuario, estas técnicas pueden utilizarse en escenarios de arranque en frío. Además, en el caso más grave, cuando se añade un nuevo artículo al catálogo, los métodos basados en el contenido permiten las recomendaciones, ya que pueden extraer características del nuevo elemento y utilizarlas para hacer recomendaciones. Cabe destacar que, aunque los sistemas de filtrado colaborativo tienen problemas de arranque en frío tanto para los nuevos usuarios como para los nuevos artículos, los sistemas basados en el contenido sólo tienen problemas de arranque en frío para los nuevos usuarios.

En cuanto al problema de un nuevo elemento, un enfoque estándar consiste en extraer una serie de características que definen las propiedades acústicas de la señal de audio y utilizar la recomendación basado en el contenido (basándonos en el perfil del usuario) para efectuar recomendaciones. La extracción de características de la canción suele hacerse de forma automática, pero también puede ser realizada manualmente por expertos musicales, como en el caso del Proyecto Genoma Musical de *Pandora*. [45]

Pandora utiliza hasta 450 características por canción, como "vocalista femenina agresiva", "coros prominentes", "letras abstractas", "uso de armonías inusuales"... Independientemente de si el proceso de extracción de características se realiza de forma automática o manual, este enfoque es ventajoso no sólo para abordar el problema de un nuevo elemento, sino también porque una representación precisa de las características puede ser utilizada para predecir los gustos e intereses de los usuarios.

Una técnica alternativa para abordar el problema de los nuevos elementos es la hibridación. En [23], los autores proponen un sistema de recomendación de música que

combina un sistema recomendador basado en el contenido y un recomendador de filtrado colaborativo basado en artículos. Para el sistema basado en el contenido, calcula las características acústicas, el timbre, el ritmo y el tono... El componente basado en el contenido ayuda al recomendador de filtrado colaborativo a la hora de abordar el problema del arranque en frío, ya que las características del primero se derivan automáticamente a través del análisis de audio.

Otra solución para el arranque en frío son las técnicas de recomendación entre dominios, cuyo objetivo es mejorar las recomendaciones en un dominio (en este caso la música) utilizando información sobre las preferencias del usuario en un dominio auxiliar.Por tanto, el conocimiento de las preferencias del usuario se obtiene de un dominio auxiliar al dominio de la música. El resultado es un modelo de usuario más completo y preciso. También es posible integrar información adicional sobre los (nuevos) usuarios, que no están directamente relacionados con la música, como su personalidad, para mejorar la estimación de las preferencias musicales del usuario. Varios estudios realizados sobre las características de la personalidad de los usuarios apoyan la conjetura de que puede ser útil explotar esta información en sistemas de recomendación musical [26][27].

Además de los enfoques mencionados, el aprendizaje activo ha mostrado resultados prometedores para tratar el problema del arranque en frío para un solo dominio [28] o en un escenario de recomendación entre dominios [29]. El aprendizaje activo aborda este problema en su origen, identificando y obteniendo datos que pueden representar las preferencias de los usuarios mejor que lo que ellos mismos proporcionan. Por lo tanto, un sistema de este tipo exige constantemente de una respuesta específica del usuario para poder funcionar correctamente.

3.3 Generación automática de listas de reproducción.

En su definición más genérica, una lista de reproducción es simplemente una secuencia de pistas destinadas a ser escuchadas juntas. La tarea de generación automática de listas de reproducción (APG) se refiere a la creación automática de estas secuencias de pistas. En este contexto, la ordenación de las canciones en una lista de reproducción para generar suele ser una de las características de la APG, que es una tarea muy compleja. Algunos autores han propuesto enfoques basados en cadenas de Markov para modelar las transiciones entre canciones en las listas de reproducción, por ejemplo, [30]. Las investigaciones recientes han encontrado pocas pruebas de que el orden exacto de las canciones sea realmente importante para los usuarios [31], mientras que el conjunto de canciones de una lista de reproducción [32] y las transiciones directas de canción a canción [33] sí son importantes.

Considerada como una variante de la APG, la tarea de continuación de la lista de reproducción (APC) consiste en añadir una o más pistas a una lista de reproducción de forma que se ajusten a las mismas características de la lista de reproducción original. Esto tiene ventajas tanto en la escucha como en la creación de listas de reproducción: los usuarios pueden disfrutar de la escucha de sesiones continuas más allá del final de una lista de reproducción de longitud finita, y, al mismo tiempo, les resulta más fácil crear listas de reproducción más largas y atractivas sin necesidad de estar muy familiarizados con la música.

Una gran parte de la tarea de APC consiste en inferir con precisión el objetivo de una lista de reproducción determinada. Esto es un reto no sólo por la amplia gama de propósitos que se persiguen, sino también por la diversidad de las características subyacentes que pueden ser necesarias para inferir esos propósitos. Un escenario de inicio extremo para esta tarea es cuando se crea una lista de reproducción con algunos metadatos (por

ejemplo, el título de una lista de reproducción), pero no se ha añadido ninguna canción a la lista de reproducción. Este problema se puede plantear como una tarea de recuperación de información ad hoc, en la que la tarea consiste en clasificar las canciones en respuesta a una consulta de metadatos proporcionada por el usuario.

Según un estudio realizado en 2016 por la *Music Business Association* [46], las listas de reproducción representaron el 31 % del tiempo de escucha de música entre los oyentes en EE. UU., más que los álbumes (22 %) de los oyentes en Estados Unidos, pero menos que las canciones sueltas (46 %). Otros estudios, como el de *MIDIA* [47], muestran que el 55 % de los suscriptores de servicios de música en *streaming* crean listas de reproducción, y algunos servicios de *streaming*, como *Spotify*, albergan actualmente más de 2.000 millones de listas de reproducción. En un estudio de 2017 realizado por Nielsen [48],se encontró que el 58 % de los usuarios en los Estados Unidos crean sus propias listas de reproducción, el 32 % las comparte con otros. Estudios como estos sugieren una creciente importancia de las listas de reproducción como modo de consumo de música, y como tal, el estudio de APG y APC nunca había sido tan relevante.

3.3.1. Soluciones Actuales.

La APG se ha estudiado desde que los servicios de música en *Streaming* pusieron a disposición de los usuarios enormes catálogos de música. Bonnin y Jannach ofrecen un estudio exhaustivo de este campo en [34]. En él, los autores definen la tarea del APG como la creación de una secuencia de pistas que cumplan con algunas "características objetivo" de una lista de reproducción, dado un "conocimiento de fondo" de las características del catálogo de pistas del que se extraen las pistas de la lista de reproducción. Los sistemas APG existentes abordan estos dos problemas de muchas maneras diferentes.

En los primeros enfoques [1] las características objetivo de la lista de reproducción se especifican como múltiples restricciones explícitas, que incluyen atributos musicales o metadatos como por ejemplo, el artista, el tempo o el estilo de música. En otros, las características objetivo son una única pista inicial [4] o una pista inicial y una final [1].

Otros enfoques crean una lista de reproducción circular que comprende todas las pistas de una determinada colección de música, de forma que las canciones consecutivas sean lo más parecidas posibles [12]. En otros trabajos, las listas de reproducción se crean basándose en el contexto del oyente [9].

Un enfoque común para construir el conocimiento de fondo del catálogo de música para la generación de listas de reproducción es utilizar técnicas de aprendizaje automático para extraer ese conocimiento de las listas de reproducción. En este caso, se asume que los creadores de estas listas de reproducción están aportando una información valiosa que indica qué pistas van juntas para crear una experiencia de escucha agradable. Algunos sistemas APG y APC propuestos se entrenan con listas de reproducción de fuentes como emisoras por ejemplo los servicios de *streaming* de música [5]. En el estudio de Pichl, entre otros, [5], se analizaron los nombres de las listas de reproducción en *Spotify* para crear clusters contextuales, que luego se utilizaron para mejorar las recomendaciones.

Un enfoque para abordar específicamente el orden de las canciones dentro de las listas de reproducción es el uso de modelos que se entrenan con listas de reproducción seleccionadas a mano. McFee y Lanckriet [6] representaron canciones mediante metadatos, familiaridad y características del contenido de audio, adoptando ideas del procesamiento del lenguaje natural, entrenaron varias cadenas de Markov para modelar las transiciones entre canciones.

3.4 Aplicaciones existentes.

MusicFX[15] es un sistema de recomendación de música para gimnasios. En sus orígenes se desarrolló para aplicarlo en el *Anderson Consulting Technology Park* con 600 miembros. La información sobre las preferencias de los usuarios se recogen cuando los usuarios se inscriben en el gimnasio rellenando un cuestionario. El cuestionario es respecto a géneros musicales como el rock, éxitos, country... donde para cada uno de estos géneros se debe hacer una valoración entre [-2...2] donde un -2 sería un me encanta y un 2 seria la detesto. MusicFX opera a nivel de género, por lo que no recomienda canciones específicas.

FLYTRAP[10] es una aplicación que utiliza la tecnología RFID (identificación por radiofrecuencia) para saber qué usuarios del sistema están en la misma sala y conformar así un grupo. El enfoque que se sigue para la recomendación es el basado en contenido combinado con la agregación de preferencias (mediante votación). El sistema aprovecha los conocimientos sobre las preferencias de los usuarios (por ejemplo, qué genero les gusta) y la relación entre diferentes evaluaciones entre las canciones (por ejemplo, qué tipo de transiciones entre canciones prefiere el usuario). En FLYTRAP las preferencias de los usuarios son recopiladas mediante un agente que está instalado localmente en el ordenador del usuario.

Jukola [14] es un ejemplo de sistema de recomendación musical en grupo con participación activa. Este sistema se instaló en un bar local de Bristol. Los asistentes al bar podían obtener acceso a un dispositivo tras registrarse para participar en el proceso de decisión sobre la música que se ponía en el café. La participación se produjo en forma de sistema de votación y cada dispositivo portátil podía emitir un voto antes de la selección de la siguiente pista. En la práctica, los dispositivos portátiles fueron utilizados por grupos establecidos, pero las decisiones se compartieron entre grupos aleatorios (es decir, todos los grupos participantes en el bar del café). El sistema demostró principalmente que el proceso de decisión en sí mismo puede ser una parte importante de la experiencia de los usuarios.

ADAPTIVERADIO [19] es una aplicación de recomendación de canciones basado en el contenido para grupos. Una característica singular de este sistema es que las preferencias de los usuarios que se tienen en cuenta son las negativas, es decir, las canciones que no le gustan. La idea subyacente es que a menudo es más fácil averiguar lo que no le gusta a un usuario que descubrir lo que le gusta. Si es necesario encontrar soluciones (recomendaciones) que satisfagan a todos los usuarios (que a todo el mundo le guste lo que escucha), este enfoque parece ser beneficioso. [**?**] La función de esta aplicación es reproducir canciones a un grupo y permitir que los miembros del grupo den su opinión sobre las canciones en forma de *dislikes*. Para un grupo dado, las canciones que no disgustan a uno de los miembros del grupo son candidatas a ser recomendadas. Por tanto, si a uno de los miembros del grupo no le gusta una canción esta se excluye y no volverá a reproducirse.Se utiliza una métrica de similitud básica que considera principalmente las canciones del mismo álbum como similares.

ADAPTIVE IN-VEHICLE MULTIMEDIA RECOMMENDER [13] es un sistema que recomienda elementos multimedia a un grupo de pasajeros. Los perfiles de usuario se intercambian a través de dispositivos utilizados durante el viaje. El sistema agrega las características relevantes de los perfiles individuales de los usuarios en un perfil central que se utiliza para determinar las recomendaciones (agregación del modelo como hemos visto en los fundamentos teóricos). En el contexto de las recomendaciones musicales las características relevantes podrían ser estilos musicales o nombres de artistas. A las características se les asigna un peso correspondiente que refleja la importancia de una

característica para todo el grupo. Aquellos elementos con mayor similitud global con el perfil de grupo tendrán la mayor probabilidad de ser recomendados. En este contexto no se utilizan funciones de agregación.

GROUPFUN[11] es una aplicación de recomendación de grupo implementada como un *plugin* de *Facebook* que recomienda listas de reproducción (o *playlists*) para eventos específicos, por ejemplo, cumpleaños. EN GROUPFUN, las listas de reproducción de los usuarios pueden ser agregadas y las recomendaciones (en este caso, enfocadas al evento) se determinan mediante una función de agregación llamada *suma ponderada probabilística* donde la probabilidad de que una canción se reproduzca se deriva de la popularidad global de una canción.

3.4.1. Análisis de los recomendadores de música de las aplicaciones.

Los ejemplos existentes de sistemas de recomendación de música en grupo se clasifican en la Tabla 3.1. La comparación muestra que existen tanto grupos con relaciones sociales significativas como los que no las tienen e incluso grupos al azar. Además, en situaciones en las que los sistemas de recomendación de música en grupo se encuentran en entornos compartidos, los sistemas tienen que hacer frente a la incertidumbre de quién está presente. Existen varias soluciones a este problema, como un sistema de inicio de sesión (por ejemplo, MusicFX), fichas y etiquetas (por ejemplo, Flytrap) y una mecánica probabilística. La mecánica probabilística puede utilizar predictores como la hora del día o utilizar predictores más sofisticados (por ejemplo, reconocimiento visual, reconocimiento de voz).

Aplicación	Tipo de grupo	Preferencias del usuario	Interacción de grupo	Recomienda	Agregación de
ADAPTIVERADIO	Establecido	Desarrolladas	Activo	Canción	Predicciones
FLYTRAP	Al azar	Desarrolladas	Pasivo	Canción	Predicciones
GROUPFUN	Ocasional	Conocidas	Pasivo	Secuencias	Modelo
ADAPTIVE IN-VEHICLE MULTIMEDIA RECOMMENDER	Ocasional	Desarrolladas	Activo	Canción	Predicciones
MusicFX	Al azar	Conocidas	Pasivo	Canción	Predicciones
Jukola	Al azar	Desarrolladas	Activo	Canción	Predicciones

Tabla 3.1: Comparación de las diferentes aplicaciones.

La mayoría de los sistemas desarrollan las preferencias del usuario a lo largo del tiempo haciendo uso de la retroalimentación explícita o implícita del usuario. Un reto inherente a los sistemas de recomendación en grupo es que la opinión de los usuarios puede estar influida por efectos de grupo como el contagio de las emociones y la conformidad. Por lo tanto, existe una mayor incertidumbre en la validez de la opinión del usuario obtenida y se debe tener cuidado de tener en cuenta los efectos de grupo en la interpretación de la opinión. Por ejemplo, las opiniones de los usuarios de Jukola se obtienen en un entorno social en el que un único dispositivo de obtención de opiniones es utilizado por grupos establecidos y la música es compartida por un grupo aleatorio mayor. Algunos usuarios de Jukola dijeron: ”Nos peleamos por lo que votamos”. [14] Esto demuestra que la retroalimentación de este contexto debe interpretarse como preferencias grupales de los grupos establecidos y no como preferencias individuales del usuario ya que éste puede estar en desacuerdo con la decisión del grupo. Por lo tanto, el *feedback* obtenido no debería utilizarse directamente para construir perfiles de usuario a largo plazo, sino como preferencias a corto plazo basadas en la sesión de los grupos establecidos.

A continuación, veremos cómo los enfoques de recomendación de grupos pueden diferenciarse en función de las siguientes características.

Estrategia de agregación de preferencias: En los sistemas de recomendación de grupo, existen dos estrategias básicas de agregación. En primer lugar, las recomendaciones se determinan para los miembros individuales del grupo y luego se agregan en una reco-

mendación de grupo. En segundo lugar, las preferencias de los usuarios individuales se agregan en un perfil de grupo que luego se utiliza para determinar una recomendación de grupo.

Algoritmo de recomendación: La lógica de recomendación de los recomendadores de grupo se basa en muchos casos en los recomendadores de usuarios individuales (filtrado colaborativo, filtrado basado en el contenido...) combinados con funciones de agregación seleccionadas de la teoría de la elección social. Estas funciones se analizarán a partir de los ejemplos del ámbito de los viajes presentados en el capítulo 2.

Preferencias conocidas de antemano: En algunos casos, las preferencias del usuario ya se conocen de sesiones de recomendación anteriores. En otros casos, las preferencias deben ser determinadas en un proceso posterior.

Consumo inmediato de artículos: Por un lado, una forma de recomendación podría ser que un grupo experimente directamente los artículos recomendados. Por ejemplo, consideremos las canciones que consumen los miembros de un gimnasio. Por otro lado, las recomendaciones a menudo son propuestas sin que los artículos se experimenten inmediatamente.

Grupo activo o pasivo Por un lado, los perfiles de grupo pueden generarse automáticamente si se conocen las preferencias de los miembros del grupo. Por otro lado, puede ser que no conozcamos de antemano las preferencias de los usuarios y cuanto más intensamente se discutan y negocien las preferencias para formar los modelos de grupo, mayor será el grado de actividad del grupo.

Número de artículos recomendados El resultado de un recomendador de grupo puede ser un solo artículo (por ejemplo, un restaurante para una cena o una película), pero también paquetes (por ejemplo, paquetes de viaje), secuencias (por ejemplo, canciones o planes de viaje).

Todas estas características se ilustran en la Tabla 3.2.

Características	Descripción
Estrategia de agregación utilizada	1) Determinación de los artículos/valoraciones para los miembros individuales del grupo, y posteriormente agregación de estos artículos/valoraciones para una recomendación de grupo, o (2) agregación de las preferencias de los miembros del grupo en un perfil de grupo, y posteriormente determinación de una recomendación para el grupo
Algoritmo de recomendación	Uno de los algoritmos de recomendación (es decir, colaborativo, basado en el contenido...)
¿Preferencias conocidas de antemano?	Por ejemplo, en el filtrado colaborativo, las valoraciones se conocen de antemano
¿Consumo inmediato de artículos recomendados?	Los recomendadores de grupo pueden recomendar (1) artículos que se consumirán en el futuro (por ejemplo, destinos de vacaciones como base para una decisión final tomada por (a) una persona responsable o (b) un grupo sobre la base de una discusión [35]), o (2) artículos de consumo inmediato (por ejemplo, canciones)
¿Grupo pasivo o activo?	Un grupo es pasivo si no influye activamente en la construcción de un perfil de grupo. Los grupos activos negocian el perfil del grupo.
Número de artículos a recomendar	Un recomendador de grupo puede centrarse en la recomendación de (1) un único elemento, como es el caso de los destinos de viaje, o (2) múltiples elementos representados, por ejemplo, como una secuencia (por ejemplo, elementos de televisión o una *playlist*)

Tabla 3.2: Características para clasificar un recomendador de grupos.[36][37]

CAPÍTULO 4
Desarrollo de un sistema de recomendación de música para grupos.

4.1 Sistema de recomendación desarrollado.

Antes de profundizar con el desarrollo en cuestión, aclararemos el sistema que vamos a implementar basándonos en la tabla 3.2. La estrategia de agregación que utilizaremos sera la 2) es decir determinaremos los artículos para los miembros del grupo y después haremos una selección de aquellas que sean mejor para el grupo. El algoritmo de recomendación que utilizaremos será el filtrado basado en contenido. Como preferencias de cada usuario, tomaremos aquellas canciones que el usuario ha añadido a la *playlist* para formar un perfil de usuario y también el perfil de grupo. Puesto que los artículos que recomendamos son canciones serán de consumo inmediato. Y, dado que no utilizaremos las estrategias de agregación de modelo no nos importa si es activo o pasivo el grupo. Sin embargo, puesto que no conocemos quién ha creado las listas de reproducción, no podemos recibir *feedback* por lo que tratamos con un grupo pasivo. El número de artículos a recomendar será de una secuencia de 10 canciones.

Los grupos pueden clasificarse en grupos homogéneos y grupos heterogéneos en función de su composición. Un grupo homogéneo está formado por miembros con intereses similares, mientras que los grupos heterogéneos están formados por miembros con intereses diversos. El problema con un grupo heterogéneo es que, como los miembros de un grupo tienen diferentes intereses, crear un consenso para satisfacer a todos los miembros del grupo es una tarea difícil. En este estudio, tratamos con *playlists* del dataset de *Spotify* [49] con un millón de *playlist* de las cuales haremos un filtrado que explicaremos más adelante. Por lo tanto, algunas serán de grupos homogéneos y otras serán de grupos heterogéneos. No podemos saberlo a priori.

4.2 Algoritmos de recomendación desarrollados.

Por un lado, disponemos de un dataset con 1M de *playlists* de las cuales formaremos los grupos y las canciones añadidas por cada usuario nos servirán para formar las preferencias de cada usuario. El conjunto de datos contiene 1.000.000 de listas de reproducción, incluidos los títulos de las listas y de las canciones, creadas por los usuarios en la plataforma *Spotify* entre enero de 2010 y octubre de 2017.

Cada lista de reproducción en el *dataset* contiene un título de lista de reproducción, la lista de canciones (incluyendo los IDs de las canciones y los metadatos), y otros campos de metadatos (última hora de edición, número de ediciones de la lista de reproducción, y más). Un ejemplo de este conjunto es el mostrado en la figura 4.1


```
{
        "name": "musical",
        "colaborative": "false",
        "pid": 5,
        "modified_at": 1493424000,
        "num_albums": 7,
        "num_tracks": 12,
        "num_followers": 1,
        "num_edits": 2,
        "duration_ms": 2657366,
        "num_artists": 6,
        "tracks": [
            {
                "pos": 0,
                "name_of_artist": "Degiheugi",
                "track_uri": "spotify:track:7vqa3sDmtEaVJ2gcvxtRID",
                "artist_uri": "spotify:artist:3V2paBXEoZIAhfZRJmo2jL",
                "track_name": "Finalement",
                "album_uri": "spotify:album:2KrRMJ9z7Xjoz1Az4O6UML",
                "duration_ms": 166264,
                "album_name": "Dancing Chords and Fireflies"
            } // omitimos pistas
        ],

    }
```


Figura 4.1: Contenido de una lista de reproducción que pertenece al set de datos de *Spotify* 1M *playlists*.

Por otro lado, para realizar nuestro sistema de recomendación también vamos a explotar el *dataset* de *Kaggle* [50], *Spotify Dataset 1922-2021, 600k Tracks*, el cual nos proporciona los vectores de características de alrededor de 600 mil canciones. Las características que contempla este *dataset* son las mostradas en la figura 4.2 o en la figura 4.3. Las características más relevantes son las:

- Relacionadas con el estado de ánimo: *Danceability, Valence, Energy, Tempo*
- Relacionadas con las propiedades de la canción: *Loudness, Speechiness, Instrumentalness*
- Relacionadas con el contexto: *Liveness, Acousticness*

Con este conjunto de datos desarrollaremos un sistema de recomendación basado en contenido. El filtrado basado en el contenido se basa en (a) un conjunto de usuarios y (b) un conjunto de categorías (o palabras clave) que se han asignado a (o extraído de) el

```
{
    "danceability": 0.735,
    "energy": 0.578,
    "key": 5,
    "loudness": -11.84,
    "mode": 0,
    "speechiness": 0.0461,
    "acousticness": 0.514,
    "instrumentalness": 0.0902,
    "liveness": 0.159,
    "valence": 0.624,
    "tempo": 98.002,
    "type": "audio_features",
    "id": "06AKEBrKUckW0KREUWRnvT",
    "uri": "spotify:track:06AKEBrKUckW0KREUWRnvT",
    "duration_ms": 255349,
    "time_signature": 4
}
```


Figura 4.2: Características de una canción perteneciente al set de datos de *Kaggle* .

conjunto de artículos. En nuestro caso, nuestras palabras clave serán las características mencionadas con anterioridad (*danceability, valence, liveness...*).

Nuestro sistema de recomendación se basa en los vectores de características de las canciones para encontrar canciones similares. Para ello utiliza la distancia coseno. Un ejemplo de un vector de características (en la imagen la primera canción del set de datos) podría ser el de la figura 4.3.

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
0	0.0594	1921	0.98200	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0	4BJqT0PrAfrxzMOxytFOIz	0.878000	10	0.6650	-20.096	1	Piano Concerto No. 3 in D Minor, Op. 30: III. ...	4	1921	0.0366	80.954

Figura 4.3: Vector de características.

En este trabajo hemos desarrollado cuatro maneras diferentes de recomendar canciones para un grupo. Para clarificar todos los conceptos que vamos a mencionar en esta sección, vamos a realizar un ejemplo sencillo donde tenemos cuatro canciones (figura 4.4), suponemos que tenemos un grupo de dos usuarios, el primer usuario ha añadido las dos primeras canciones a la playlist y el segundo las dos restantes.

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popularity	release_date	speechiness	tempo
0	0.0594	1921	0.98200	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0	4BJqT0PrAfrxzMOxytFOIz	0.878000	10	0.6650	-20.096	1	Piano Concerto No. 3 in D Minor, Op. 30: III. ...	4	1921	0.0366	80.954
1	0.9630	1921	0.73200	['Dennis Day']	0.819	180533	0.341	0	7xPhfUan2yNtyFG0cUWkt8	0.000000	7	0.1600	-12.441	1	Clancy Lowered the Boom	5	1921	0.4150	60.936
2	0.0394	1921	0.96100	['KHP Kridhamardawa Karaton Ngayogyakarta Hadi...	0.328	500062	0.166	0	1o6I8BglA6ylDMrIELygv1	0.913000	3	0.1010	-14.850	1	Gati Bali	5	1921	0.0339	110.339
3	0.1650	1921	0.96700	['Frank Parker']	0.275	210000	0.309	0	3ftBPsC5vPBKxYSee08FDH	0.000028	5	0.3810	-9.316	1	Danny Boy	3	1921	0.0354	100.109

Figura 4.4: Vector de características de las cuatro canciones.

Si disponemos en forma de matriz las características (solo tenemos en cuenta las numéricas) de las canciones de la playlist P tenemos:

$$P = \begin{bmatrix} 0,0594 & 1921 & 0,982 & \cdots & 1921 & 0,0366 & 80,954 \\ 0,963 & 1921 & 0,732 & \cdots & 1921 & 0,415 & 60,936 \\ 0,0394 & 1921 & 0,961 & \cdots & 1921 & 0,0339 & 110,339 \\ 0,165 & 1921 & 0,967 & \cdots & 1921 & 0,0354 & 100,109 \end{bmatrix}$$

Si disponemos en forma de matriz las características de las canciones que el usuario u1 y u2 ha añadido a la playlist P tenemos:

$$P_{u1} = \begin{bmatrix} 0,0594 & 1921 & 0,982 & \cdots & 1921 & 0,0366 & 80,954 \\ 0,963 & 1921 & 0,732 & \cdots & 1921 & 0,415 & 60,936 \end{bmatrix}$$

$$P_{u2} = \begin{bmatrix} 0,0394 & 1921 & 0,961 & \cdots & 1921 & 0,0339 & 110,339 \\ 0,165 & 1921 & 0,967 & \cdots & 1921 & 0,0354 & 100,109 \end{bmatrix}$$

4.2.1. *Recommend songs*

Este método obtiene un vector con la media de todas las características de las canciones de la *playlist* de los usuarios del grupo y lo utiliza para obtener las N canciones (vectores) más similares mediante la distancia coseno. Para realizar recomendaciones con este módulo necesitamos un perfil de grupo, que se obtiene realizando la media del vector de características de todas las canciones de la *playlist* Por lo tanto, esta forma de recomendar canciones utiliza como función de agregación la media.

$$mc^t = \frac{\sum_{c \in P} caract(c,t)}{|P|} \tag{4.1}$$

Donde mc^t es la media de características para la característica t *(danceability, instrumentalness...)*, P es el conjunto de canciones c de la *Playlist* y $caract(c,t)$ es una función que nos devuelve la característica de la canción c de tipo t. Realizamos mc^t para cada una de las características, obteniendo así vmc_G que es el vector media de todas las características de las canciones de la *playlist*, y por tanto, del grupo G.

Siguiendo con nuestro ejemplo, procedemos a calcular el vector de características (vmc_G). Realizamos la media teniendo en cuenta al grupo completo obteniendo vmc_G

$$vmc_G = \begin{bmatrix} 0,3067 & 1921 & 0,9105 & \cdots & 1921 & 0,130225 & 88,0845 \end{bmatrix}$$

Para obtener canciones similares a nuestro vector vmc_G realizamos la distancia coseno con nuestro corpus de todas las canciones C, comparando una a una y obteniendo aquellas cuya distancia sea menor.

$$recommend\_songs = \underset{(c \in C)}{\operatorname{argmin}}\ cosine\_dist(vmc_G, c) \tag{4.2}$$

Es decir, para la recomendación con *recommend_songs*, tal y como se especifica en la ecuación 4.2, calculamos la distancia coseno entre nuestro vector de características del grupo vmc_G y cada una de las canciones de nuestro corpus C, y se seleccionan aquellas N canciones con menor distancia coseno. En nuestro ejemplo, se realiza este cálculo con el dataset de *Kaggle* con las características de 600k canciones y nos quedamos con aquellas N=4 cuya distancia es menor, obteniendo como resultado las canciones de la figura 4.5.

	name	year	artists	id
3382	El Noom Yodaeb	1938	['Umm Kulthum']	12PDB2nGsnq2x5B7023T5y
77756	Symphony in Three Movements: I. [q = 160]	1947	['Igor Stravinsky', 'New York Philharmonic']	3Nzt6cp2YtFKd3jbHzoRhw
126284	Egmont, Op. 84: Overture - 1992 Remastered	1947	['Ludwig van Beethoven', 'Arturo Toscanini']	7cPUOY9KnS9uRTYg7Nn67f
156242	Symphony No. 41 in C Major, K. 551 "Jupiter": ...	1948	['Wolfgang Amadeus Mozart', 'Arturo Toscanini']	6j0yE4Lt30mw3pv1hEKSQr

Figura 4.5: Canciones recomendadas para el ejemplo con el método *recommend_songs*

4.2.2. *Recommend homogenize.*

Este método filtra las canciones de la *playlist* retirando aquellas cuyas características estén por debajo de la media. Después de haber realizado el filtrado se realiza *recommend_songs*.

En este caso se realiza un filtrado de la *playlist* del grupo P antes de generar las recomendaciones. El objetivo es "homogeneizar" las características de la *playlist*, es decir eliminar aquellas canciones cuyas características disten mucho de las características de la *playlist*. Esta decisión fue tomada para ver si obteníamos mejores resultados con un grupo con características similares, tratando de simular una *playlist* de un grupo homogéneo.

$$mc^{t}_{umbral} = \frac{\sum_{c \in P} caract(c,t)}{|P|} : caract(c,t) > umbral \tag{4.3}$$

De esta manera todas las canciones que están por debajo del umbral no se tienen en cuenta. Nosotros tomamos como umbral por defecto, la media de cada una de las características. Por lo tanto, esta forma de recomendar canciones utiliza como función de agregación la media con umbral. Realizamos mc^{t}_{umbral} para cada una de las características, obteniendo así $vmc_{G,umbral}$

$$recommend\_songs\_homogenize = \underset{(c \in C)}{\operatorname{argmin}}\ cosine\_dist(vmc_{G,umbral}, c) \tag{4.4}$$

En el caso de *recommend_homogenize* el filtrado por umbral solo nos elimina la primera canción. Y realiza *recommend_songs*, es decir obtiene vmc_G sin tener en cuenta la primera canción. Obtenemos como resultado las canciones de la figura 4.6.

	name	year	artists	id
127752	Sonata for Violin and Piano: I. Un poco allegro	1954	['Stefan Wolpe']	0MiykSM2Nm2hUDIqqRmq8P
60540	Yeh Teri Ankhiyan	1947	['Indrabadan Bhatt']	2ggvxAfQInh0mzymDknxBF
76718	Can't We Talk It Over	1939	['Jack Teagarden And His Orchestra']	7o5foa62sNJuvuLsKLamSQ
127725	Yeh Barkha Bahar	1954	['Lata Mangeshkar', 'Asha Bhosle']	0KB1rBcIzFGAl6IgRh7RCQ

Figura 4.6: Canciones recomendadas para el ejemplo con el método *recommend_homogenize*

4.2.3. *Recommend individually*

Este método obtiene por un lado, un vector por cada miembro del grupo con la media de todas las características de las canciones que ha añadido a la *playlist*. Después utiliza esos vectores para obtener canciones similares mediante la distancia coseno. La recomendación será la unión de estas canciones. Si por ejemplo queremos cuatro canciones y tenemos un grupo de dos personas, solo obtendremos dos canciones similares por miembro.

En este caso, para generar las recomendaciones obtenemos el vector media de características para cada usuario. Es decir, de la *playlist* solo seleccionamos las canciones del usuario sin tener en cuenta el resto de canciones.

$$mc_u^t = \frac{\sum_{c \in P_u} caract(c,t)}{|P_u|} \tag{4.5}$$

Donde mc_u^t es la media de características del usuario u para la característica t, P_u es el conjunto de canciones c que el usuario u ha añadido a la *Playlist*. Realizamos vmc_u^t para cada una de las características, obteniendo así vmc_u que es el vector media de todas las características de las canciones del usuario u.

Siguiendo con nuestro ejemplo, procedemos a calcular el vector de características (vmc). Realizamos la media teniendo en cuenta a cada usuario obteniendo vmc_{u1} para el usuario 1 y $vmcu2$ para el usuario 2.

$$vmc_{u1} = \begin{bmatrix} 0,5112 & 1921 & 0,857 & \cdots & 1921 & 0,2258 & 70,945 \end{bmatrix}$$

$$vmc_{u2} = \begin{bmatrix} 0,1022 & 1921 & 0,964 & \cdots & 1921 & 0,03465 & 105,224 \end{bmatrix}$$

Realizamos la distancia coseno con el corpus de canciones C y obtenemos aquellas canciones cuya distancia sea menor con nuestro vector vmc_u. Si queremos realizar una recomendación de cuatro canciones y tenemos dos usuarios solo recomendaremos dos para cada usuario.

$$recommend\_u = \underset{(c \in C)}{\operatorname{argmin}}\ cosine\_dist(vmc_u, c) \tag{4.6}$$

Una vez tenemos las canciones, procedemos a hacer la unión de las canciones para generar la recomendación final de grupo.

$$recommend\_individually = \cup_{u \in G} recommend_u \tag{4.7}$$

Para la recomendación con *recommend_individually* calculamos la distancia coseno como en la ecuación 4.6 para el vector vmc_{u1} para cada una de las canciones de nuestro corpus C y nos quedamos con aquellas 2 cuya distancia sea menor. Hacemos lo mismo para el vector vmc_{u2} y hacemos la union de las canciones obteniendo como resultado las canciones de la figura 4.7.

	name	year	artists	id
76177	Schubert: Symphony No. 8 in B Minor, D. 759, "...	1935	['Franz Schubert', 'Karl Böhm', 'Wiener Philha...	3eG5V71sCaXxiNjMok9DP9
142994	Symphony No. 9 in C Major, D. 944 "Great": IV....	1954	['Franz Schubert', 'Wiener Philharmoniker', 'W...	0U80cnafIuSgjEB3ejBAnf
173	Pas Pour Moi	1922	['Maurice Chevalier']	1QDTYyfndMN9SDtRKDv9XO
141580	Chhai Huyi Duniya Pe Abhi Raat Hai	1946	['Radharani']	7JEAAJ7h7ctZVkVDEoHXtM

Figura 4.7: Canciones recomendadas para el ejemplo con el método *recommend_individually*

4.2.4. *Recommend collaborative*

Este método de realizar las recomendaciones es muy similar al anterior, generamos las recomendaciones para cada usuario, obteniendo vmc_u para cada usuario y generando N canciones con $recommend_u$. Sin embargo de estas canciones recomendadas solo nos

quedaremos con aquellas cuyo vector de características sea más similar al vector de toda la playlist.

$$filter\_u = \underset{(\cup_{u \in G} recommend_u)}{\text{argmin}} \; cosine\_dist(vmc_G, c) \tag{4.8}$$

Al igual que en el caso anterior, procederemos a hacer la unión de las canciones para generar la recomendación final de grupo.

Para la recomendación con *recommend_collaborative* hacemos *recommend_individually* pero en lugar de solo recomendar dos canciones por usuario obtenemos diez canciones por usuario. De estas diez canciones calculamos la distancia coseno entre cada canción recomendada y vmc_G. De todas ellas seleccionamos las dos cuya distancia sea menor. Realizamos lo mismo para el segundo usuario y hacemos la unión obteniendo como resultado las canciones de la figura 4.8.

	name	year	artists	id
77756	Symphony in Three Movements: I. [q = 160]	1947	['Igor Stravinsky', 'New York Philharmonic']	3Nzt6cp2YtFKd3jbHzoRhw
76177	Schubert: Symphony No. 8 in B Minor, D. 759, "...	1935	['Franz Schubert', 'Karl Böhm', 'Wiener Philha...	3eG5V71sCaXxiNjMok9DP9
94112	Kunjan Ban Chhanri He Madho	1947	['M. S. Subbulakshmi']	4XQhtm3FoGRFl68BQrsKo8
77674	Jungle Mein Pariyon Ka Mela	1946	['Afsal', 'Gopal']	3on4moT6hGa67K34iwuUly

Figura 4.8: Canciones recomendadas para el ejemplo con el método *recommend_collaborative*

4.3 Descripción de la evaluación de nuestro SRG.

La investigación sobre sistemas de recomendación ha utilizado varios tipos de medidas para evaluar la calidad de las recomendaciones ofrecidas a los individuos, como la *precision*, el *recall* o el error medio absoluto (MAE). Sin embargo, hasta donde sabemos, no existe una medida ampliamente aceptada para evaluar los SRG. Por este motivo, hemos adaptado el MAE al contexto de las recomendaciones de grupo porque es el más utilizado y el más fácil de interpretar directamente cuando se utiliza para evaluar los SR.

En primer lugar, definimos el MAE^u, que da una medida de la desviación de la recomendación para el grupo con respecto a los valores estimados para un miembro del grupo por sí mismo. Dada una recomendación R^G de N ítems para un grupo G tal que $u \in G$, el error medio absoluto para el usuario u se define:

$$MAE^u = d(u, R^G) \tag{4.9}$$

donde $d(u, R^G)$ es la distancia coseno entre el vector de características de las canciones que ha añadido a la *playlist* el usuario u y el vector de características de las canciones recomendadas para el grupo G. Por lo tanto, el MAE^u indica lo adecuada que es la recomendación de grupo para el usuario u. Cuanto menor sea el MAE^u, más precisa es la recomendación de grupo para este usuario.

A diferencia de las recomendaciones individuales, cuando se trata de grupos, una cuestión muy importante es obtener recomendaciones que sean lo más satisfactorias posible para todos los miembros del grupo, es decir, evitar la miseria. Por lo tanto, nuestro interés es medir la satisfacción del grupo en su conjunto, es decir, la exactitud de la recomendación del grupo para todos sus miembros. Esto se consigue unificando el MAE^u de cada miembro del grupo en una única medida; concretamente, definimos el MAE^G como la media del MAE^u de todos los miembros del grupo. Por tanto, un MAE^G bajo indica que el grupo en su conjunto está muy satisfecho con la recomendación.

$$MAE^G = \frac{\sum_{i=1}^{|G|} d(u, R^G)}{|G|} \tag{4.10}$$

4.4 Conjunto de datos y generación de grupos.

Para llevar a cabo los experimentos, se generaron grupos de usuarios de distintos tamaños mediante el muestreo del conjunto de datos de *Spotify Million Playlist Dataset* [49], que contiene la información de un millón de *playlists* de usuarios de *Spotify*. En este experimento, solo contemplamos grupos aleatorios ya que estamos creando "grupos simulados" que son formados a partir de las *playlists* y dejamos la creación de grupos homogéneos (miembros con gustos similares) para un trabajo posterior. Ambos casos son comunes en la vida real. Los grupos homogéneos representan a personas con gustos comunes, como los grupos de amigos. En cambio, los grupos aleatorios representan a personas sin ninguna relación social, como personas al azar que toman el mismo autobús. Lo que realizamos es filtrar del conjunto de datos aquellas que fueran *playlists* colaborativas. El numero de usuarios para cada *playlist* que conformaría el numero de usuarios pertenecientes al grupo se tomó como el numero de usuarios que seguían la *playlist* más uno. Esto fue porque algunas *playlists* colaborativas solo tenían un seguidor. *Spotify* no ofrece información en este *dataset* del usuario que añadió la canción por lo que lo que realizamos una asignación circular de las canciones (si tenemos un grupo de dos personas y cuatro canciones, el primer usuario añadiría la primera canción y la tercera y el segundo usuario la segunda canción y la cuarta).

4.5 Análisis de los resultados obtenidos.

Para realizar este experimento analizamos un total de 3081 *playlists* y 3081 grupos "simulados" diferentes recomendando 10 canciones para cada método de los comentados con anterioridad. Los cálculos fueron realizados en un ordenador corriendo Ubuntu con un procesador Intel® Core™ i7-8750H CPU @ 2.20GHz × 6 núcleos.

	W	pval	equal_var
levene	1063.304866	0.0	False

Figura 4.9: Prueba de homocedasticidad de Levene.

Para ver si existen diferencias significativas entre cada método propuesto planteamos realizar un estudio estadístico de los datos. En concreto, tratamos de realizar un análisis de la varianza (ANOVA). Sin embargo, no pudimos realizarlo pues uno de los prerrequisitos de este análisis es que tiene que haber varianzas iguales entre todos los grupos. La homogeneidad de las varianzas puede comprobarse con la prueba de homocedasticidad de Levene realizada en la siguiente figura.

Para saber qué método funciona mejor planteamos realizar un conteo de valores mínimos ya que un MAE^G bajo indica que el grupo en su conjunto está muy satisfecho con la recomendación. Es decir, para cada *playlist* que hemos analizado nos quedaremos con el valor del método de recomendación que da un MAE mínimo. En total analizamos 3081 *playlists*, de las cuales 2030 tuvieron como mejor valor MAE el método de recomendación *recommend_individually*, 707 el método *recommend_collaborative*, 322 el método *recommend_songs* y 23 el método *recommend_homogenize*.

Procedemos a realizar este mismo conteo sin tener en cuenta el método *recommend_individually*. En este caso, de las 3081 *playlists* analizadas, 1895 tuvieron como mejor método de recomendación *recommend_collaborative*, 1144 *recommend_songs* y 43 *recommend_homogenize*. Este método (*recommend_collaborative*) parece más prometedor en el sentido de la re-

comendación de grupo pues hace una recomendación solo teniendo en cuenta al usuario pero solo selecciona aquellas que son más similares al perfil de grupo.

Apreciando la figura 4.10, donde vemos los resultados de las cuatro primeras *playlist* podemos ver como los resultados con el método *homogenize* no obtienen buenos resultados. Si dejamos de contemplar la recomendación con *recommend_homogenize* vemos que tres de las cuatro veces hemos obtenido buenos resultados con *recommend_individually*. Si además descartamos tanto *recommend_homogenize* como *recommend_individually* para ver si podemos extraer alguna conclusión más vemos que los resultados son similares, siendo un poco mejor el método *recommend_collaborative*. Habría que realizar un estudio posterior para ver cuáles son las características del grupo o de la *playlist* que hacen que un método funcione mejor que otro puesto que con los recursos que disponemos no es posible determinarlo con certeza.

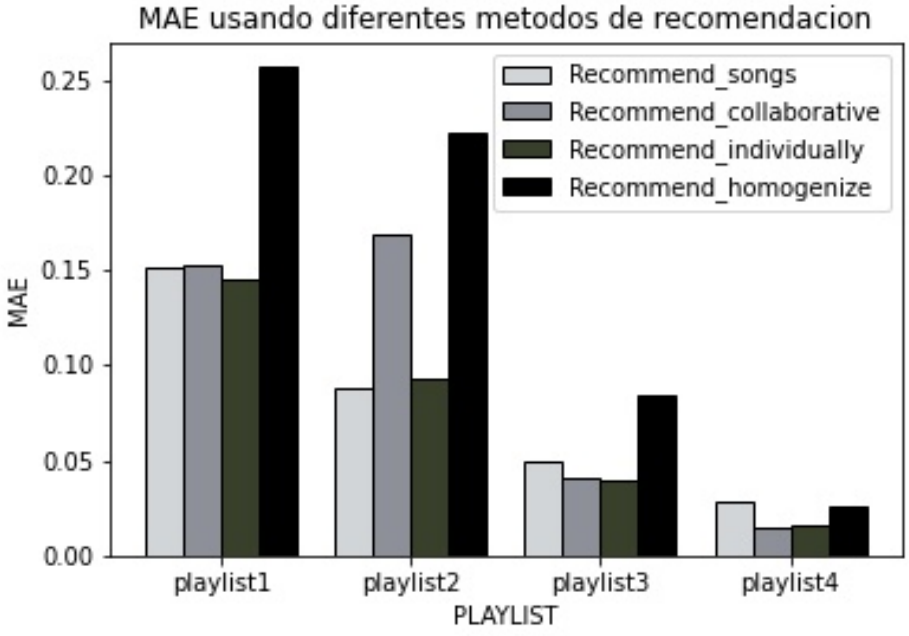


Figura 4.10: MAE usando los métodos de recomendación *songs, collaborative, individually, homogenize*.

Si representamos todos los valores MAE para cada método obtenemos el *boxplot* de la figura 4.11 donde podemos ver como en todos los casos disponemos de un número considerable de valores atípicos. Esto pensamos que puede ser debido a que en determinadas circunstancias que no sabemos cuyo trabajo se deja para un estudio posterior, obtenemos una recomendación que no satisface al grupo dando un valor MAE muy alto. Podemos ver también que tenemos una distribución asimétrica positiva, podemos apreciarlo en la tabla 4.12 que nos indica que nuestros datos se concentran "en la parte más baja" indicándonos que el funcionamiento en general de los métodos es bueno.

Cabe mencionar también la media de los valores MAE obtenidos con cada método, con *recommend_individidually* obtuvimos una media de 0.08784, con el método *recommend_collaborative* obtuvimos una media de 0.10012, con el método *recommend_songs* obtuvimos una media de 0.10242 y por último, con el método *recommend_homogenize* obtuvimos una media de 0.26248. Los valores son próximos a cero lo que nos indica que el grupo en su conjunto está satisfecho con la recomendación.

Por último, si representamos con un gráfico de lineas los valores MAE de las cien primeras *playlists* obtendríamos como resultado la gráfica de la figura 4.13. En ella po-

demos ver que todos los métodos exceptuando el de *recommend_homogenize* tienen un resultado similar. Si nos fijamos en la gráfica de la figura 4.14 podemos ver como en general, el método que mejor funciona es *recommend_individidually* (linea azul) seguido de *recommend_collaborative* (linea naranja) pues ambas lineas tienen los valores más bajos. Sin embargo no parece que hayan diferencias notables entre ambos métodos. Respecto al método *recommend_homogenize* vemos unos resultados con una valor MAE alto lo que nos hace dudar si se ha desarrollado correctamente este método.

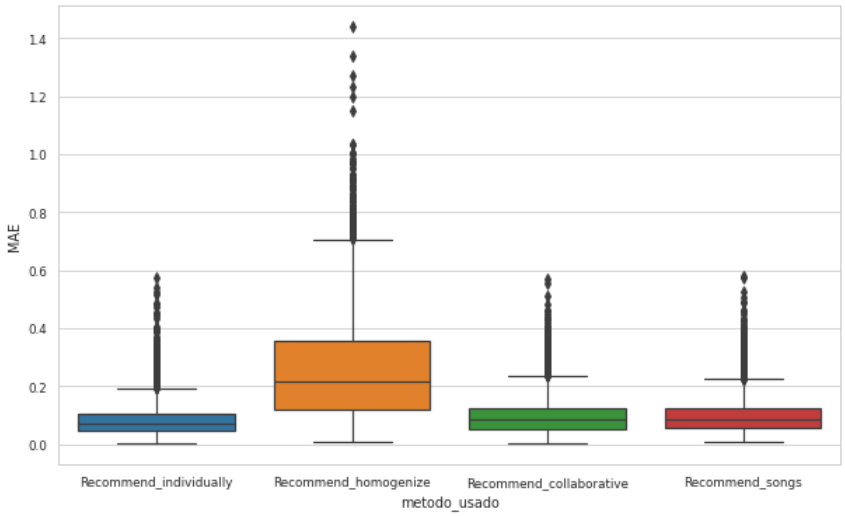
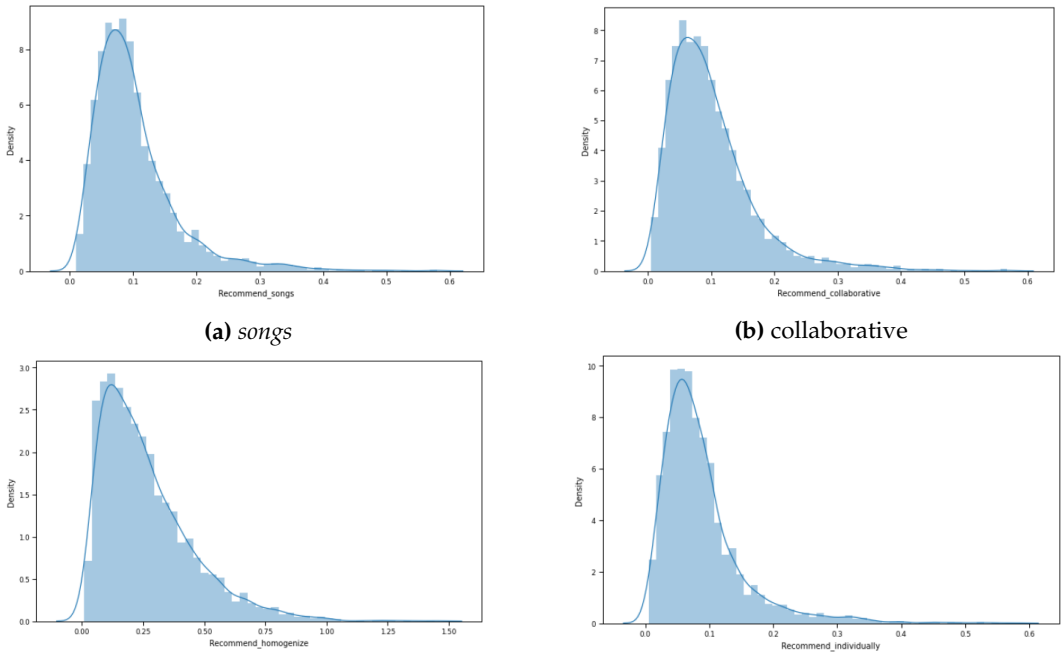


Figura 4.11: Diagrama de caja y bigotes para los métodos de recomendación *songs*, *collaborative*, *individually*, *homogenize*.



(a) *songs*

(b) collaborative

(c) *homogenize*

(d) *individually*

Figura 4.12: Gráfico de distribucion para los métodos de recomendación.

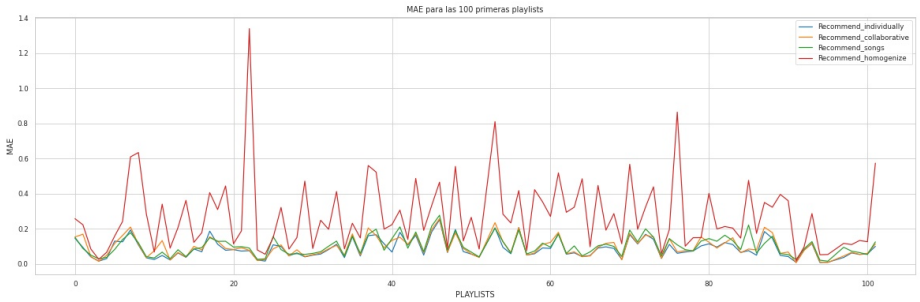


Figura 4.13: Gráfico de lineas de los valores MAE para los métodos de recomendación *songs, collaborative, individually, homogenize*.

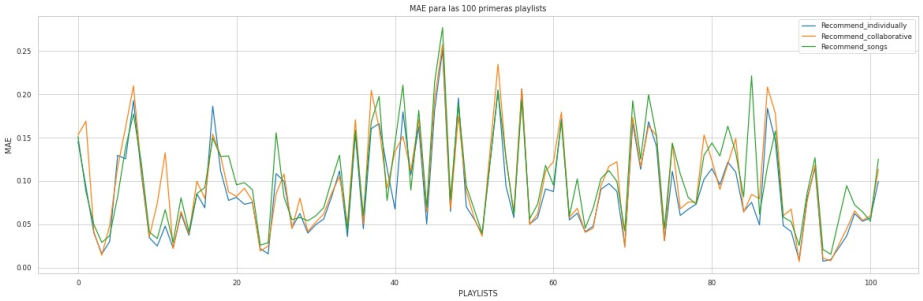


Figura 4.14: Gráfico de lineas de los valores MAE para los métodos de recomendación *songs, collaborative, individually*.

CAPÍTULO 5
Conclusiones, Limitaciones y Trabajo Futuro.

5.1 Conclusiones

En este trabajo, hemos introducido diferentes técnicas de recomendación de grupo que se basaban en los enfoques de recomendación para usuarios individuales presentados en la sección de Fundamentos Teóricos en la sección 2.3. Hemos mostrado cómo se pueden diseñar escenarios de recomendación de grupo relacionados con el filtrado colaborativo y el filtrado basado en el contenido. Y para ello, nos centramos en la discusión de las dos estrategias de agregación: (1) predicciones agregadas (artículos) y (2) modelos agregados. En (1), las recomendaciones se determinan para los miembros individuales del grupo y luego se agregan. En (2), se agregan las preferencias de los miembros del grupo y, a continuación, se determinan las recomendaciones a partir de la información contenida en el perfil de grupo integrado.

Centrándonos específicamente en los sistemas de recomendación de grupos, hemos ofrecido una visión general de las técnicas de evaluación. Hemos aprendido que la evaluación de los sistemas de recomendación de grupo a menudo puede llevarse a cabo empleando los enfoques de evaluación estándar de los sistemas de recomendación de un solo usuario. Queremos destacar que hay otras métricas que se pueden considerar al evaluar los sistemas de recomendación de grupo. Algunos ejemplos son la confianza (*trust*), la privacidad (*privacy*) y el rendimiento (*performance*). Es importante destacar que la utilidad de algunas métricas de evaluación también dependen del dominio del artículo. Por ejemplo, los sistemas de recomendación de música no sólo persiguen los objetivos de precisión, sino también otros criterios como por ejemplo medir la probabilidad de que vuelvas a escuchar una canción que hace tiempo que no escuchas pero que te gustó mucho en un determinado momento.

Puesto que realizar un estudio con personas reales estaba fuera del alcance de este trabajo tuvimos que realizar una simulacion basándonos en el *Spotify Million Playlist Dataset* [49]. Los enfoques de síntesis para generar grupos parecen una alternativa prometedora y 'barata' a los estudios con grupos reales. A menudo, se aplican enfoques de agrupación para derivar grupos a partir de conjuntos de datos de un solo usuario. La síntesis de grupos también puede basarse en el análisis de redes sociales en las que los lazos sociales pueden servir como indicador de la pertenencia a un grupo [16]. Sin embargo, hay que mencionar que estos enfoques se basan en simulaciones y no deberían sustituir a los estudios controlados. Para obtener una visión general de los conjuntos de datos relacionados con los sistemas de recomendación para un solo usuario y los marcos de software

existentes que pueden servir de base para el desarrollo de sistemas de recomendación de grupos, nos remitimos a Said entre otros. [21].

Esperamos que este trabajo haya contribuido a señalar los principales retos, a destacar las tendencias recientes y a identificar cuestiones de investigación interesantes en el ámbito de los sistemas de recomendación musical. Creemos que la investigación que aborde los retos y las tendencias comentadas allanará el camino para la próxima generación de sistemas de recomendación musical, y esperamos que se produzcan enfoques y sistemas interesantes e innovadores que mejoren la satisfacción y la experiencia del usuario, en lugar de limitarse a las medidas de precisión.

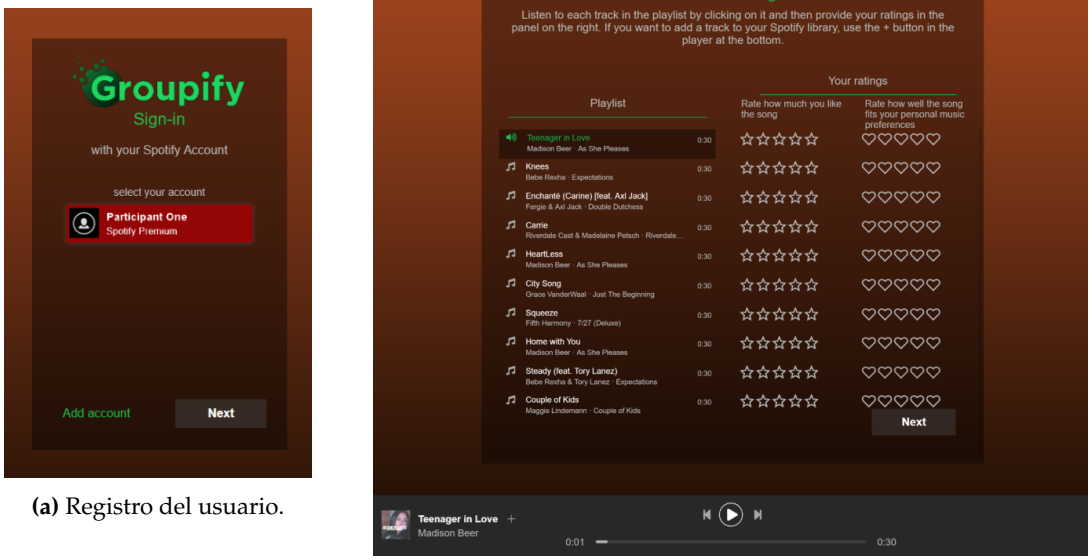
5.2 Limitaciones

A la hora de realizar este trabajo nos hemos encontrado con una serie de problemas que han perjudicado los resultados que hemos obtenido. Puesto que todo el experimento ha sido simulado, los resultados obtenidos no son tan ilustrativos como los que podemos obtener con un grupo de personas detrás que nos de *feedback* sobre los resultados que obtenemos. Además, no podemos obtener evaluaciones subjetivas sobre las recomendaciones que realizamos, por tanto, aunque nuestro recomendador tenga una buena métrica MAE puede que las recomendaciones no sean del agrado del grupo de usuarios. Las preferencias de cada usuario se obtenían solo teniendo en cuenta las canciones que añadía a la lista de reproducción. Esto es así puesto que no podemos acceder a las preferencias del usuario sin su previo consentimiento a través del API de Spotify. Por ese motivo, nuevamente perdemos en fidelidad al tratar con solo una parte del gusto del usuario.

5.3 Trabajo futuro

Como trabajo futuro, para cubrir todas estas limitaciones que nos han surgido proponemos realizar un estudio con un grupo de usuarios. Para ello es necesario el consentimiento de aquellos que intervengan en dicho experimento. Un ejemplo de una aplicación similar es Groupify, una aplicación realizada para hacer el seguimiento y obtener retroalimentación sobre las recomendaciones realizadas. [35]. Además, no hemos podido realizar las métricas que comentamos en la sección 2.5 debido a que nos faltaban datos por parte del usuario, así pues sería conveniente llevarlas a cabo junto con la aplicación que comentamos en un trabajo futuro.

Además, presentamos una perspectiva de lo que creemos que son las direcciones de investigación futuras más interesantes en los SRM. En concreto, hablamos de (1) los SRM 'psicológicamente inspirados' que tienen en cuenta en el proceso de recomendación factores como la emoción y la personalidad de los oyentes. Esto puede ser muy interesante en el ámbito musical pues si un usuario se encuentra animado no deseará escuchar canciones tristes. (2) los SRM conscientes del contexto, que modelan los aspectos contextuales y ambientales del proceso de consumo de música, infieren las necesidades e intenciones del oyente y, finalmente, integran estos modelos en el proceso de recomendación, y (3) los SRM conscientes de la cultura, que explotan el hecho de que el gusto musical depende en gran medida de los antecedentes culturales del oyente, donde la cultura puede definirse de múltiples maneras, incluyendo similitudes históricas, políticas, lingüísticas o religiosas.



(a) Registro del usuario.

(b) Retroalimentación del usuario

Figura 5.1: Groupify.

Bibliografía

[1] Alghoniemy M, Tewfik A A network flow model for playlist generation. Proceedings of the IEEE international conference on multimedia and expo (ICME), Tokyo, Japan

[2] Dara, S., Chowdary, C.R. & Kumar, C. A survey on group recommender systems. *J Intell Inf Syst 54, 271–295 (2020).* Consultar en: `https://doi.org/10.1007/s10844-018-0542-3`

[3] M. Ekstrand, J. Riedl, J. Konstan, . Collaborative filtering recommender systems. *Found. Trends Human-Comput. Interact. 4(2), 81–173 (2011)*

[4] Logan B Content-based playlist generation: exploratory experiments. (2002) Proceedings of the 3rd international symposium on music information retrieval (ISMIR), Paris, France

[5] Pichl M, Zangerle E, Specht G Towards a context-aware music recommendation approach: what is hidden in the playlist name? (2015) 2015 IEEE international conference on data mining workshop (ICDMW). IEEE, pp 1360–1365

[6] McFee B, Lanckriet G The natural language of playlists. Proceedings of the 12th international society for music information retrieval conference (ISMIR 2011), Miami, FL, USA

[7] M. Pazzani, D. Billsus, Mach. Learning and revising user profiles: the identification of interesting web sites. *Learn. 27(3), 313–331 (1997)*

[8] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, Recommender Systems – An Introduction *(Cambridge University Press, New York, 2010)*

[9] Schedl M, Breitschopf G, Ionescu B music genius: reggae at the beach, metal on a Friday night? (2014) Proceedings of the 4th ACM international conference on multimedia retrieval (ICMR), Glasgow, UK

[10] A. Crossen, J. Budzik, K. Hammond, FLYTRAP : intelligent group music recommendation *7th International Conference on Intelligent User Interfaces*, San Francisco, CA, USA (2002), pp. 184–185

[11] G. Popescu, P. Pu, What's the best music you have?: designing music recommendation for group enjoyment in groupfun *, in CHI '12 Extended Abstracts on Human Factors in Computing Systems*, Austin, TX, USA (2012), pp. 1673–1678

[12] Knees P, Pohle T, Schedl M, Widmer G Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. (2006) Proceedings of the 8th ACM SIGMM international workshop on multimedia information retrieval (MIR), Santa Barbara, CA, USA

[13] Y. Zhiwen, Z. Xingshe, Z. Daquing, An adaptive in-vehicle multimedia recommender for group users , *in 61st Vehicular Technology Conference*, Stockholm, Sweden (2005), pp. 1–5

[14] O'Hara, Kenton & Lipson, Matthew & Jansen, Marcel & Unger, Axel & Jeffries, Huw & Macer, Peter. Jukola: democratic music choice in a public space. (2004). 145-154. 10.1145/1013115.1013136.

[15] J. McCarthy, T. Anagnost, MusicFX: an arbiter of group preferences for computer supported collaborative workouts. *Conference on Computer Support Cooperative Work* , Seattle, WA, USA (1998), pp. 363–372

[16] A. Mislove, B. Viswanath, K. Gummadi, P. Druschel, You are who you know: inferring user profiles in online social networks 3rd ACM Conference on Web Search and Data Mining (WSDM'10), New York, 2010, pp. 251–260

[17] Schedl, M., Zamani, H., Chen, CW. et al. Current challenges and visions in music recommender systems research. *Int J Multimed Info Retr 7*, 95–116 (2018). `https://doi.org/10.1007/s13735-018-0154-2`

[18] Christophe Senot, Dimitre Kostadinov, Makram Bouzid, Jérôme Picault, and Armen Aghasaryan. Evaluation of group profiling strategies. *In Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Three (IJCAI'11). AAAI Press*, 2728–2733(2011). `https://doi.org/10.1007/s13735-018-0154-2`

[19] D. Chao, J. Balthorp, S. Forrest. Adaptive radio: achieving consensus using negative preferences *ACM SIGGROUP Conference on Supporting Group Work, Sanibel Island, FL, USA* , (2005), pp. 120–123

[20] J. Herlocker, J. Konstan, L. Terveen, J. Riedl, Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*22(1), 5–53 (2004)

[21] A. Said, D. Tikk, P. Cremonesi, Benchmarking a methodology for ensuring the relative quality of recommendation systems in software engineering (Springer, Berlin, 2014), pp. 275–300

[22] M. Ge, C. Delgado-Battenfeld, D. Jannach, Beyond accuracy: evaluating recommender systems by coverage and serendipity , *in 4th ACM Conference on Recommender Systems* Barcelona, 2010, pg 257–260

[23] Donaldson J A hybrid social-acoustic recommendation system for popular music. (2007) Proceedings of the ACM conference on recommender systems (RecSys), Minneapolis, MN, USA

[24] Kaminskas M, Ricci F Contextual music information retrieval and recommendation: state of the art and challenges. (2012) Comput Sci Rev 6(2):89–119

[25] Dror G, Koenigstein N, Koren Y, Weimer M The yahoo! music dataset and kdd-cup'11. (2011) Proceedings of the 2011 international conference on KDD Cup 2011, vol 18, pp 3–18. JMLR.org

[26] Ferwerda B, Graus M, Vall A, Tkalčič M, Schedl M The influence of users' personality traits on satisfaction and attractiveness of diversified recommendation lists. Proceedings of the 4th workshop on emotions and personality in personalized services (EMPIRE 2016), Boston, USA

[27] Hu R, Pu P A study on user perception of personality-based recommender systems. (2010) Bra PD, Kobsa A, Chin DN (eds) UMAP, Lecture Notes in Computer Science, vol 6075. Springer, pp 291–302

[28] Rashid AM, Karypis G, Riedl J Learning preferences of new users in recommender systems: an information theoretic approach. (2008) SIGKDD Explor Newsl 10:90–100. https://doi.org/10.1145/1540276.1540302

[29] Zhang Z, Jin X, Li L, Ding G, Yang Q Multi-domain active learning for recommendation. (2016) In: AAAI, pp 2358–2364

[30] Chen S, Moore JL, Turnbull D, Joachims T Playlist prediction via metric embedding. Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'12. ACM, New York, NY, USA, pp 714–722. https://doi.org/10.1145/2339530.2339643

[31] Tintarev N, Lofi C, Liem CC Sequences of diverse song recommendations: an exploratory study in a commercial system. Proceedings of the 25th conference on user modeling, adaptation and personalization, UMAP'17. ACM, New York, NY, USA, pp 391–392. https://doi.org/10.1145/3079628.3079633

[32] Vall A, Quadrana M, Schedl M, Widmer G, Cremonesi P The importance of song context in music playlists. Proceedings of the poster track of the 11th ACM conference on recommender systems (RecSys), Como, Italy

[33] Iman Kamehkhosh Dietmar Jannach GB How automated recommendations affect the playlist creation behavior of users. Joint proceedings of the 23rd ACM conference on intelligent user interfaces (ACM IUI 2018) workshops: intelligent music interfaces for listening and creation (MILC), Tokyo, Japan

[34] Bonnin G, Jannach D Automated generation of music playlists: survey and experiments. (2015) ACM Comput Surv

[35] Hadash, S. Evaluating a framework for sequential group music recommendations: a modular framework for dynamic fairness and coherence control (29 Mar 2019

[36] J. Masthoff. *Group recommender systems: combining individual models, in Recommender Systems Handbook* . Springer, pg. 677–702.

[37] J. Masthoff. *Group recommender systems: aggregation, satisfaction and group attributes, in Recommender Systems Handbook*. Springer, pg 743–776

[38] A. Jameson, B. Smyth. *Recommendation to groups, in The Adaptive Web*. Springer, pg 596–627

[39] Schäfer T, Sedlmeier P, Stdtler C, Huron D. *The psychological functions of music listening.*. Front Psychol 4(511):1–34

[40] Alexander Felfernig, Ludovico Boratto, Martin Stettinger, and Marko Tkali. *Group Recommender Systems: An Introduction (1st. ed.)*. 2018. Springer Publishing Company, Incorporated.

[41] F. Ricci, L. Rokach, and B. Shapira, *Recommender systems handbook*. Second Edition. Springer, 2015.

[42] Filtrado colaborativo, Consultar en https://es.wikipedia.org/wiki/Filtrado_colaborativo.

[43] Filtrado Basado en Contenido Consultar en https://recommendersys.wordpress.com/2017/11/05/filtrado-basado-en-contenido/

[44] Neural Collaborative Filtering Consultar en https://towardsdatascience.com/neural-collaborative-filtering-96cef1009401

[45] The Music Genome Project Consultar en https://www.pandora.com/about/mgp

[46] Playlists Overtake Albums in Listenership, Says LOOP Study Consultar en https://musicbiz.org/news/playlists-overtake-albums-listenership-says-loop-study/

[47] Announcing MIDiA's State Of The Streaming Nation 2 Report Consultar en https://www.midiaresearch.com/blog/announcing-midias-state-of-the-streaming-nation-2-report

[48] Music 360 – 2017 Highlights Consultar en https://www.nielsen.com/us/en/insights/report/2017/music-360-2017-highlights/

[49] Spotify Million Playlist Dataset Consultar en https://www.aicrowd.com/challenges/spotify-million-playlist-dataset-challenge

[50] Spotify Dataset 1922-2021, 600k tracks Consultar en https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks