

IBM - DATA SCIENCE - MACHINE LEARNING WITH PYTHON

RECOMMENDER SYSTEMS

Intro to Recommender Systems

Por más de que los gustos de las personas varíen, podemos encontrar similitudes o grupos entre los gustos de estas. *Las personas tienden a gustar de cosas en las mismas categorías o que compartan las mismas características.*

Por ejemplo, si hace poco compraste un libro de ML en python y te gusto, es muy probable que también te guste leer algo de data viz. Además, las personas tienden a tener gustos similares a las personas cercanas en sus vidas. Los RSists tienen estas cosas en cuenta para tratar de descubrir que más podría llegar a gustarle al usuario.

What are recommender systems?

Recommender systems capture the pattern of peoples' behavior and use it to **predict what else they might want or like.**



Applications

- What to buy?
 - E-commerce, books, movies, beer, shoes
- Where to eat?
- Which job to apply to?
- Who you should be friends with?
 - LinkedIn, Facebook, ...
- Personalize your experience on the web
 - News platforms, news personalization



Dentro de las ventajas de usar RS's tenemos..

Advantages of recommender systems

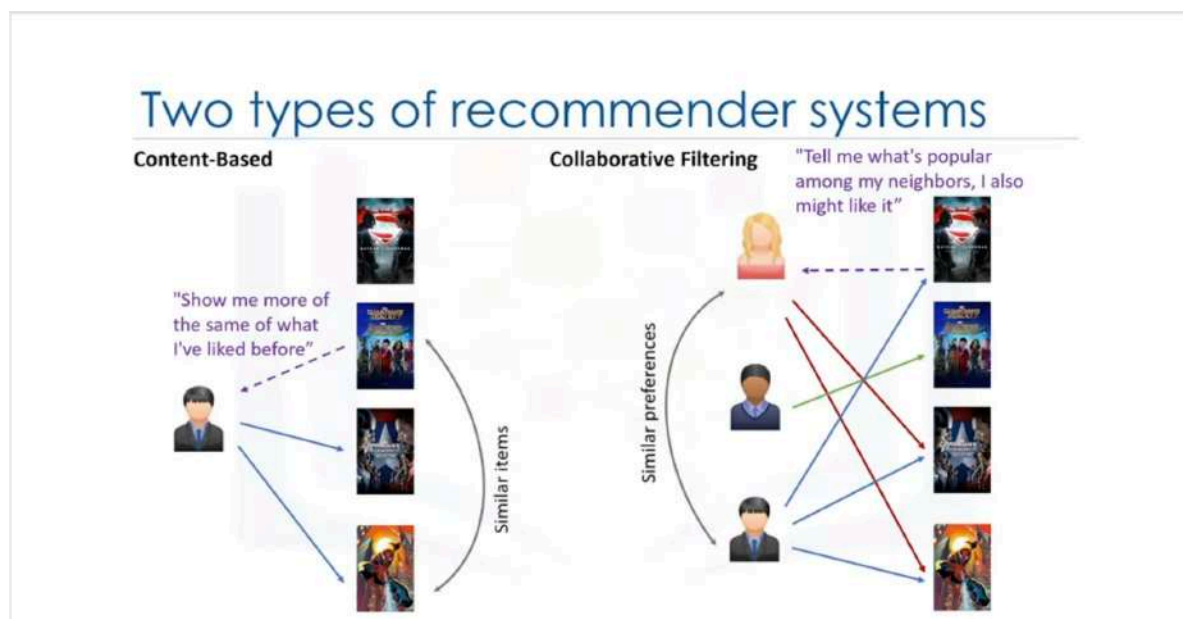
- Broader exposure
- Possibility of continual usage or purchase of products
- Provides better experience

Por lo general hay dos principales tipos de Rs's:

Content based y collaborative filtering.

La principal diferencia se encuentra en el tipo de acción q haga un usuario.. por ejemplo, lo principal en un content based es *mostrame mas de lo que estuve viendo que ya me gusto*. Estos intentan adivinar que es lo que le gusta (aspectos) de un ítem al usuario, para después poder hacer recomendaciones sobre otros ítems que los compartan.

Collaborative filtering se basa en el statement de *che cántame que le gusta mis vecinos porque capaz que a mi también me gustaría consumir de eso ee*. Asume que al usuario x le va a gustar lo que le gusta al y y al z.

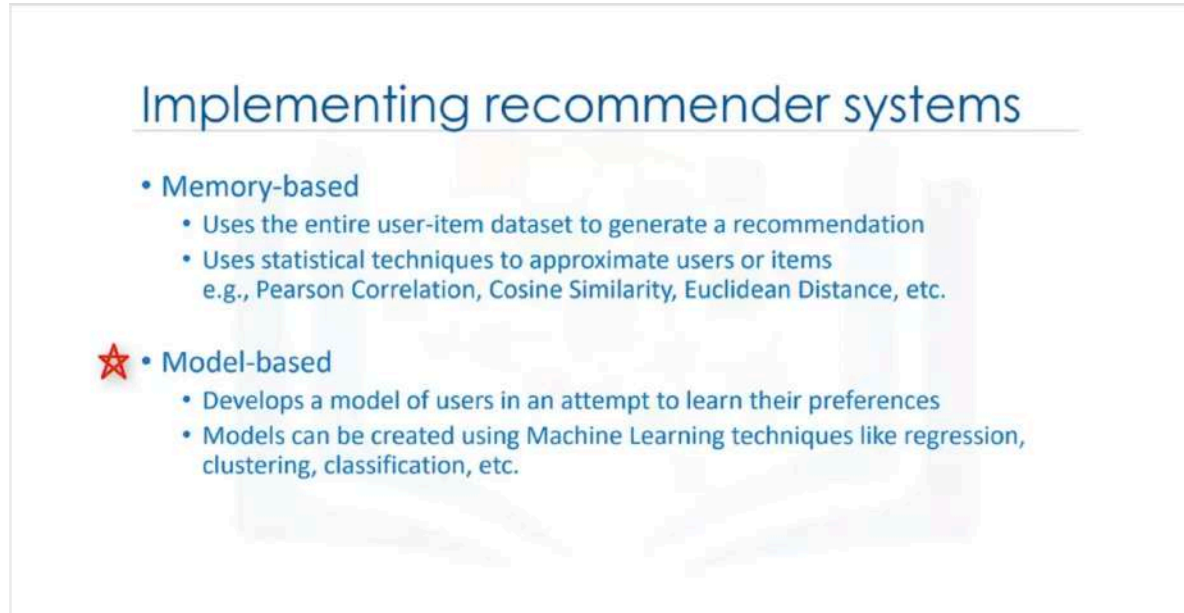


También hay sistemas que mezclan varios tipos.

En cuanto a implementarlos, hay dos tipos. Memory based y model based. En memory, usamos el user item dataset entero para generar la

recommendation. Usa técnicas estadísticas para aproximar usuarios o ítems.

En model based approaches, un modelo de usuario es desarrollado para tratar de aprender sus preferencias.



Implementing recommender systems

- **Memory-based**
 - Uses the entire user-item dataset to generate a recommendation
 - Uses statistical techniques to approximate users or items
e.g., Pearson Correlation, Cosine Similarity, Euclidean Distance, etc.
- ★ • **Model-based**
 - Develops a model of users in an attempt to learn their preferences
 - Models can be created using Machine Learning techniques like regression, clustering, classification, etc.

CONTENT BASED - RECOMMENDER SYSTEMS

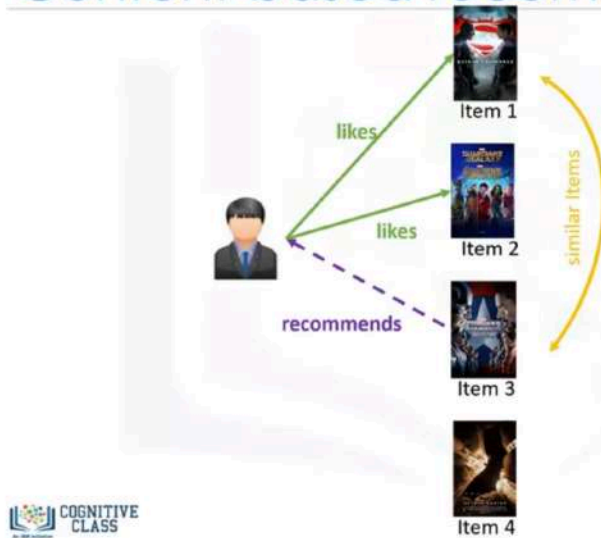


**Content-Based
Recommender Systems**

Saeed Aghabozorgi

Como dijimos, estos intentan recomendarle contenido a un usuario basados en sus perfiles. Se basa en los ratings del usuario sobre x contenido. Por ejemplo cantidad de views o likes sobre un ítem. El proceso de recomendación esta dado por la similaridad entre los ítems. Esto se mide según la similaridad del contenido de los ítems, como por ejemplo la categoría, tag, genero, etc.

Content-based recommender systems










Ejemplo, cual de los tres de abajo recomendamos si el usuario interactuó de dicha manera con los 3 primeros?

Content-based recommender systems




Lo primero que hacemos es formar un vector con los ratings de las pelis que ya vio. Lo llamamos **input user ratings**.

Weighing the genres

			Comedy	Adventure	Super Hero	Sci-Fi
	2		0	1	1	0
	10		1	1	1	1
	8		1	0	1	0





Input User Ratings

Movies Matrix




 COGNITIVE CLASS

4

Weighing the genres




	
	2
	10
	8


Input User Ratings

	Comedy	Adventure	Super Hero	Sci-Fi
	0	1	1	0
	1	1	1	1
	1	0	1	0

Movies Matrix

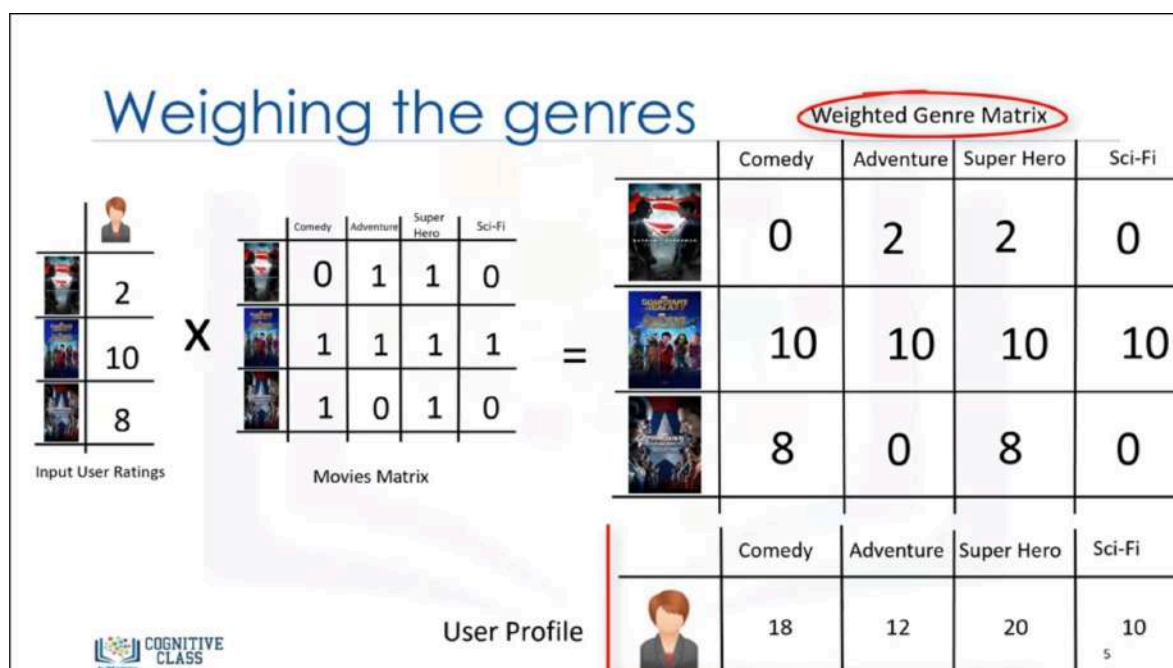
Weighted Genre Matrix

	Comedy	Adventure	Super Hero	Sci-Fi
	0	2	2	0
	10	10	10	10
	8	0	8	0

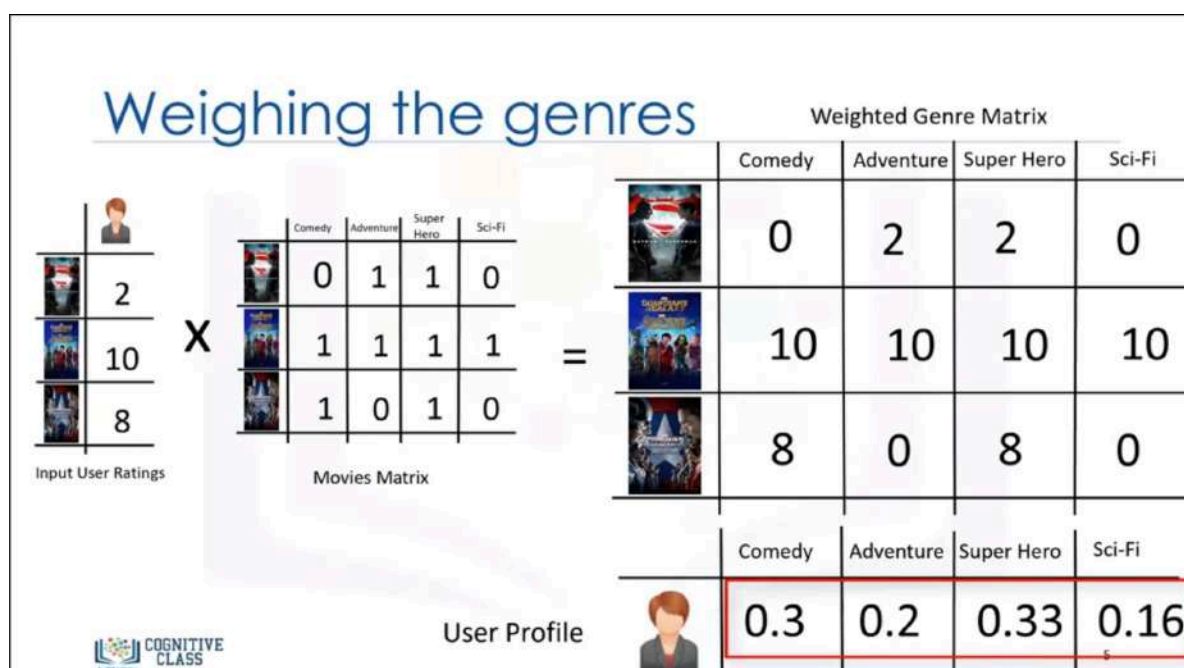


COGNITIVE CLASS

5



Normalizando para encontrar el user profile:



Ahora bien, podemos armar una matriz tambien con las posibilidades de recomendación que teníamos para nuestro usuario:


Candidate movies for recommendation



	Comedy	Adventure	Super Hero	Sci-Fi
	1	1	0	1
	0	0	1	0
	1	0	1	0

Para esto, solamente tenemos que multiplicar la matriz del user profile con la de las películas candidatas:

Finding the recommendation

	Comedy	Adventure	Super Hero	Sci-Fi
	0.3	0.2	0.33	0.16

User Profile



	Comedy	Adventure	Super Hero	Sci-Fi
	1	1	0	1
	0	0	1	0
	1	0	1	0

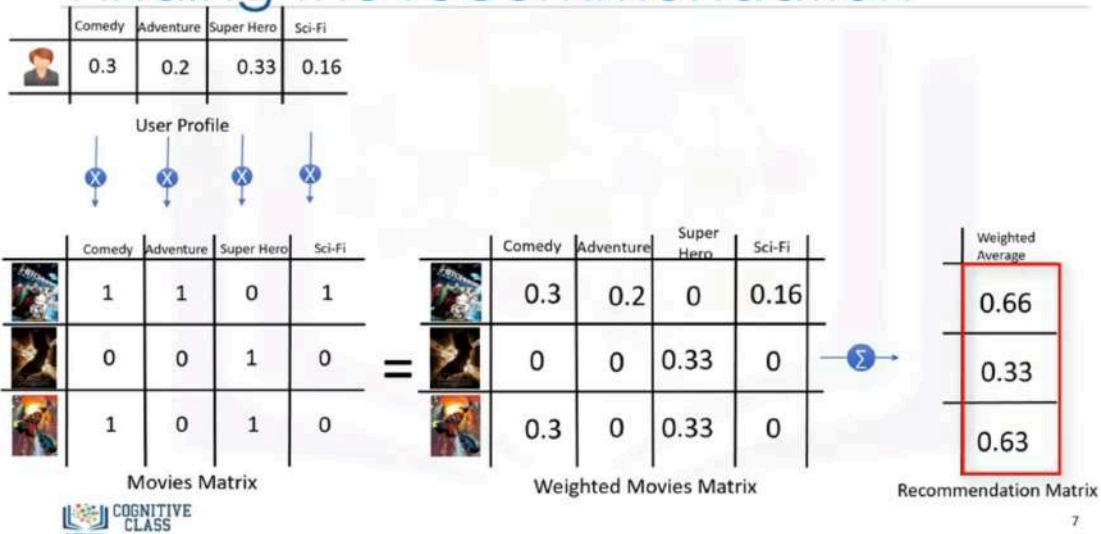
Movies Matrix

Finding the recommendation



Esto nos muestra el peso de cada genero con respecto al profile de nuestro usuario, entonces obtenemos:

Finding the recommendation



Con este vector, podemos ordenar según este peso las películas para recomendar :)

Content-based recommender systems



Si bien este modelo es muy eficiente, a veces no funciona. Por ejemplo si hay una película de drama y el usuario nunca vio una de estas y por lo tanto no lo categorizo, no tenemos manera de recomendarlo je

Content-based recommender systems



Este problema se puede resolver con otros tipos de RS como por ejemplo collaborative filtering :)

COLLABORATIVE FILTERING - RECOMMENDER SYSTEMS

Collaborative Filtering

Se basa en asumir que existen relaciones entre los productos y los intereses de los usuarios. Muchos RS usan cf para encontrar estas relaciones y para poder una recomendación acertada de un producto. Tiene básicamente dos approaches, User based y Item Based.

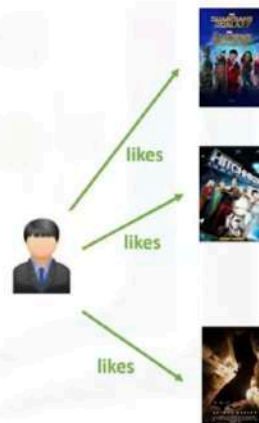
Collaborative filtering

- **User-based collaborative filtering**

- Based on users' neighborhood

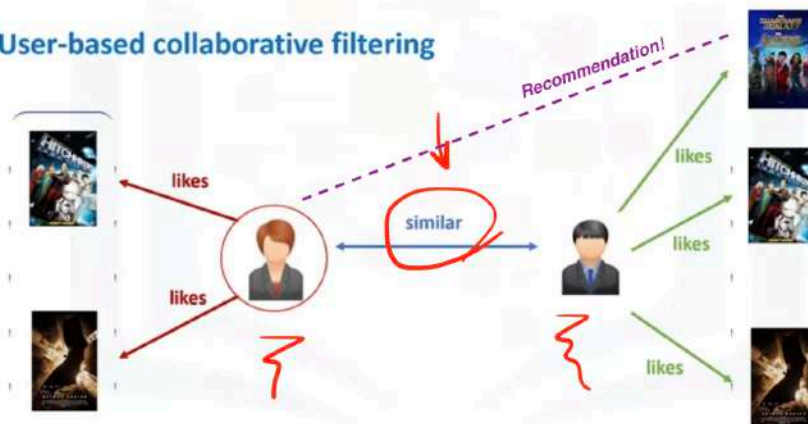
- ★ • **Item-based collaborative filtering**

- Based on items' similarity



Collaborative filtering

• User-based collaborative filtering



Bien, como funciona el algoritmo? Asumamos que tenemos una matriz como esta, que muestra los ratings de 4 usuarios distintos para 5 películas distintas:

User ratings matrix

	9	6	8	4	
	2	10	6		8
	5	9		10	7
Active user		10	7	8	

Ratings Matrix

Asumamos tambien que nuestro usuario activo ha visto y rateado 3 de estas 5 pelis.








Active user					
	?	10	7	8	?

Cual de esas dos que faltan le recomendamos?

El primer paso sera descubrir que tan similar es el usuario activo con los demas usuarios. Como lo hacemos? Hay varias técnicas vectoriales y estadísticas, como por ejemplo medidas de distancia o similitud.

Para calcular el nivel de similitud entre los usuarios, usamos las 3 películas que *todos* los usuarios ya tienen rateadas:

Learning the similarity weights

					
	9	6	8	4	
	2	10	6		8
	5	9		10	7
	?	10	7	8	?

Ratings Matrix

Podemos así calcular similitud o proximidad del usuario activo con los demás basado en esto:

Learning the similarity weights

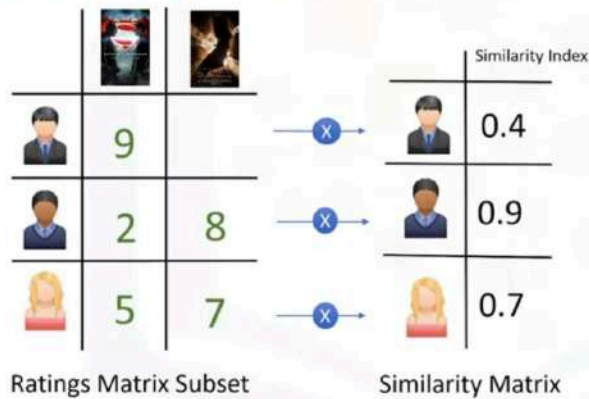
					
	9	6	8	4	
	2	10	6		8
	5	9		10	7
	?	10	7	8	?

Ratings Matrix



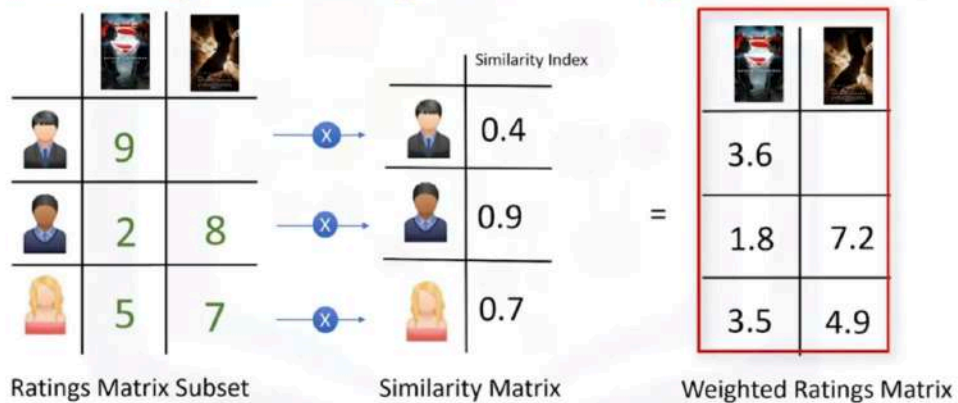
El siguiente paso del algoritmo es crear una matriz de ratings. Además, ya tenemos la similarity Matrix calculada previamente. Con esto podemos estimar los ratings de nuestro usuario de las películas restantes:

Creating the weighted ratings matrix



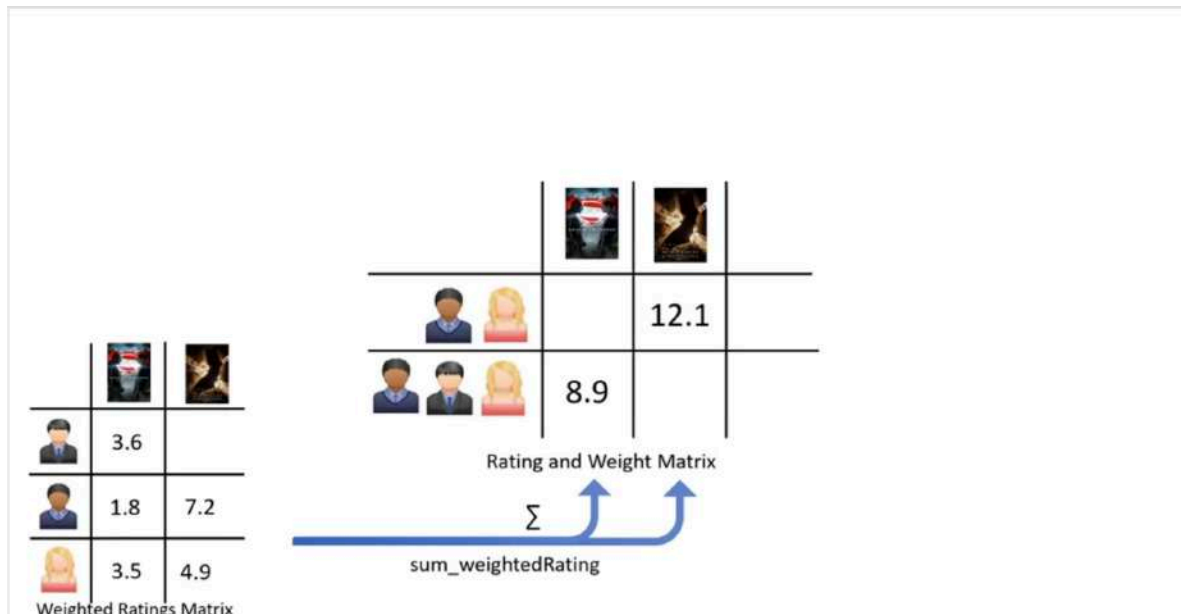
Multiplicando, obtenemos, segun el similarity índice, que tanto le van a gustar las pelis que faltan a nuestro wachin:

Creating the weighted ratings matrix

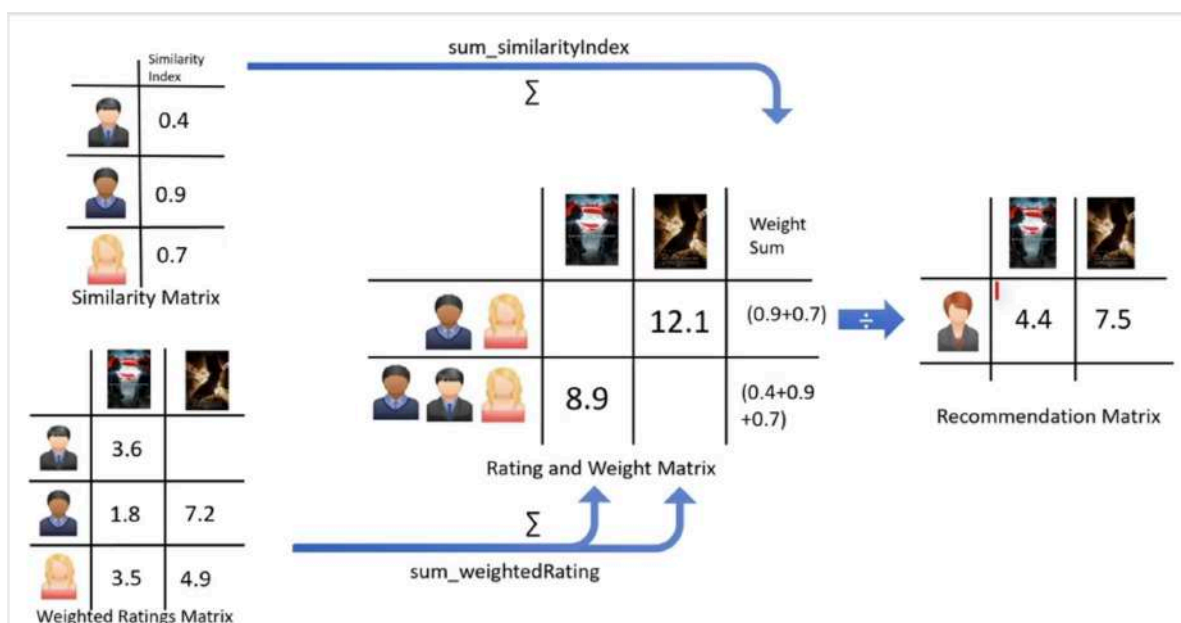


Le da mas peso al comportamiento de los usuarios que sean mas parecidos al usuario activo.

Teniendo esto, podemos crear la recommendation Matrix sumando todos los ratings:



De todas formas, como 3 usuarios ratearon la primer película pero solo dos reatearon la segunda, tenemos que normalizar los ratings values, lo hacemos dividiendo por la suma de los similarity indexes que les correspondan:

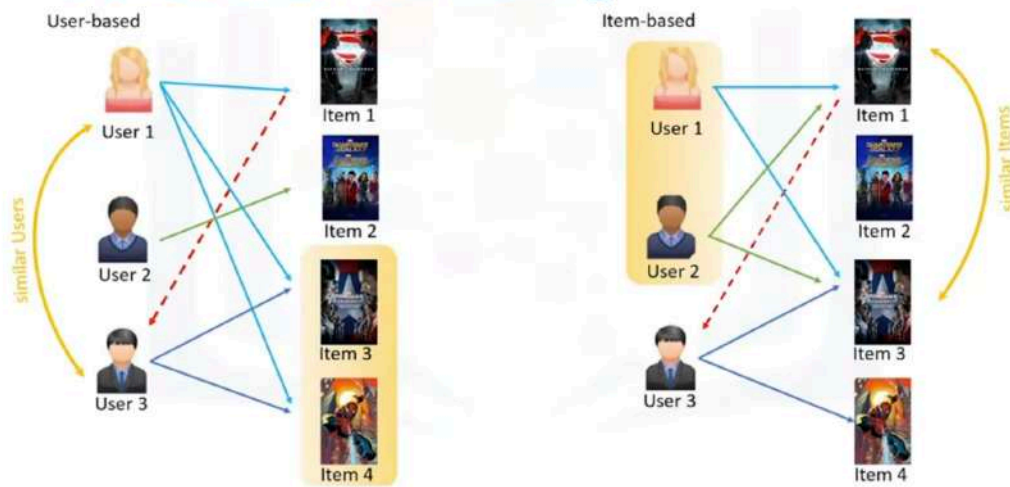


El resultado es el rating potencial que nuestro usuario le daría a las pelis faltantes.

Ahora bien, cuál es la diferencia con el CF ítem based?

En usuarios, la recomendación se basa en usuarios vecinos con los cuales el activo comparte alguna preferencia. En el ítem based, se toman como vecinos los ítems para recomendar.

Collaborative filtering



Si bien CF es re potente, tiene también sus desafíos:

Challenges of collaborative filtering

- **Data Sparsity**
 - Users in general rate only a limited number of items
- **Cold start**
 - Difficulty in recommendation to new users or new items
- **Scalability**
 - Increase in number of users or items

Dice que hay un par de soluciones para estas cosas, pero estan out of scope para nuestro curso :(