DigitalHouse > Coding School

DATA SCIENCE

MÓDULO XX

Sistemas de Recomendación

Objetivos

1. Definición del problema y abordaje
2. Arquitectura
3. Memory-Based Collaborative Filtering
4. Model-Based Collaborative Filtering
5. Caso de estudio: Netflix

Introducción

Los sistemas de recomendación tienen por objetivo sugerir a un usuario un determinado bien, servicio u otro usuario.

Los sistemas de recomendación (SR) se aplican en contextos donde un **usuario** por sí mismo no puede evaluar correctamente qué consumir (qué se puede consumir es un "**item**").

En este contexto, existen sugerencias no personalizadas y sugerencias personalizadas.

Las no personalizadas suelen ser más sencillas de hacer y podemos encontrar en este grupo a, por ejemplo, las siguientes reglas de negocio:

- "Las comedias más buscadas"
- "Los items más populares este mes"
- "Programas estadounidenses basados en libros" (tomado de Netflix)

En el caso de las sugerencias personalizadas, nuestro interés es **rankear** los items intentando que a mayor ranking mayor sea la preferencia de un usuario por el mismo.

Para ello, los SR capturan información sobre las preferencias de los usuarios, esto puede ser **explícito** o **implícito**:

- Explícito: este es el caso en que un usuario puntúa un producto, sea con estrellas, con números, con una "manito" arriba o abajo, o un "me gusta" o "no me gusta", etc.
- Implícito: en este caso se infieren las preferencias a partir de las acciones, por ejemplo, si un usuario realizó un clic en un determinado producto, vio un capítulo de una serie, vio los detalles de una descripción, etc, entonces infiero que tiene cierto interés en el producto.

Los sistemas de recomendación involucran varios pasos además del modelo de recomendaciones en sí. Se presenta una posible esquematización:

1) **Recolección de los datos:** esto involucra tomar información sobre las preferencias, sobre las características de los usuarios y sobre los items.
2) **Modelo de recomendaciones:** aquí se encuentra el modelo que genera la personalización del contenido.

3) **Post procesamiento de la recomendación:** en este punto se suelen incluir ciertas reglas de negocio que permiten mejorar las sugerencias evitando sugerencias inútiles.

En E-commerce: no recomendar un producto si no se cuenta con el stock o si es de otro país.

En streaming: no sugerir una película que ya fue vista o que ya no es parte del contenido de la página.

En redes sociales: evitar sugerir perfiles que no se pueden ver públicamente o que es la misma persona pero con otro perfil.

4) **Módulos online:** una vez que se cuenta con la recomendación hay que poder entregarla eficientemente a los usuarios cuando éstos las requieren.

Generalmente, ésto involucra distintos servicios que en real time deben poder responder las necesidades de las plataformas.

5) **Interfaz de usuario:** finalmente las recomendaciones se entregan en una interfaz gráfica, el front-end de una app.

Posteriormente nos detendremos más en este aspecto.

Clasificación

Los sistemas de recomendación tradicionalmente se agrupan en:

- Basados en popularidad: en este caso no hay una gran personalización.
- Collaborative filtering (filtro colaborativo)
- Content-based filtering (filtro basado en contenido)
- Híbridos, combinando diversos enfoques

Además, existe una gran cantidad de enfoques más recientes que emplean técnicas de deep learning.

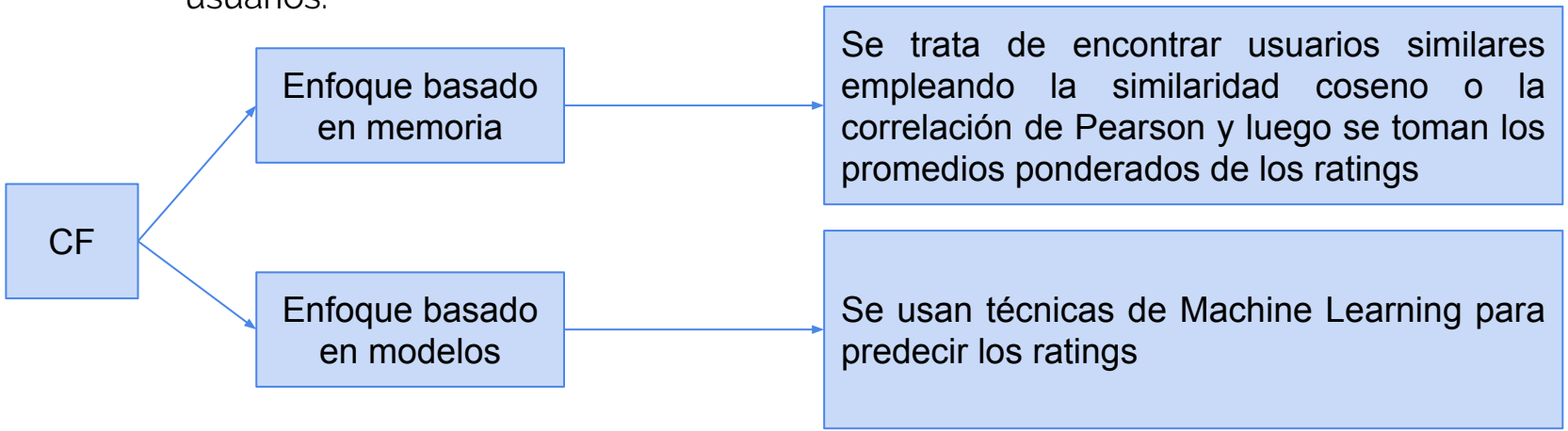
Los sistemas de recomendación tradicionalmente se agrupan en:

- Basados en popularidad: en este caso no hay una gran personalización.
- **Collaborative filtering (filtro colaborativo)**
- Content-based filtering (filtro basado en contenido)
- Híbridos, combinando diversos enfoques

Además, existe una gran cantidad de enfoques más recientes que emplean técnicas de deep learning.

El filtro colaborativo se basa en las siguientes ideas:

1) Los usuarios dan ratings al catálogo de items (de manera explícita o implícita).
2) Los patrones en los datos me pueden ayudar a predecir los ratings porque:
   a) Hay clientes con preferencias similares
   b) Hay características latentes en los items que influencian los ratings de los usuarios.



Memory-based CF

Enfoque basado en memoria: podemos distinguir dos tipos de filtros:

- **User-item filtering:** se toma a un usuario, se encuentran usuarios similares y se recomiendan items que a esos usuarios similares les gustaron. En este caso el input es un usuario y el output es un item.

  Típicamente, se explican como "A usuarios que son similares a vos también les gustó…".
- **Item-item filtering:** se toma un item, se encuentran usuarios que les haya gustado ese item, y se buscan otros items que a esos usuarios (o usuarios similares) les haya gustado. Aquí el input es un item y el output es un item.

  En general se explican como "A usuarios que les gustó … también les gustó … ".

Vamos a entender cómo se calcula cada uno y por qué el item-item filtering (o item-based filtering) es preferido por sobre el user-item filtering (o user-based filtering).

Nuestro problema es, dada la siguiente matriz, predecir los valores faltantes, rankear las mejores sugerencias y sugerir los K items con mayor ranking para alguno de los usuarios.

	The Name of the Wind	The Lord of the Rings	Introduction to Statistical Learning	Brave New World	A Study in Scarlet
Nico	5	4	4	?	?
Leo	?	3	5	5	5
Fran	4	5	5	?	3

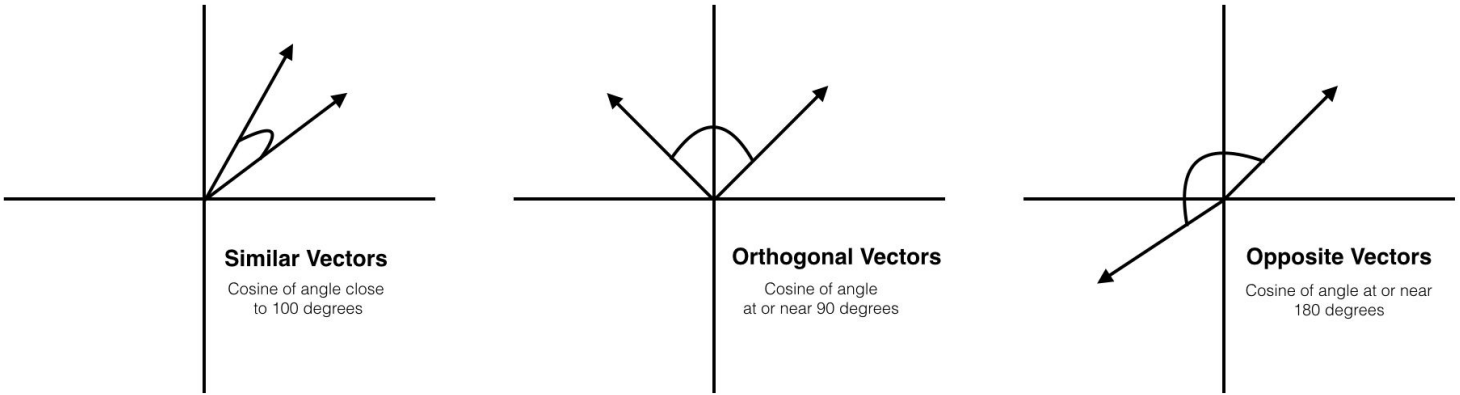
User-item matrix

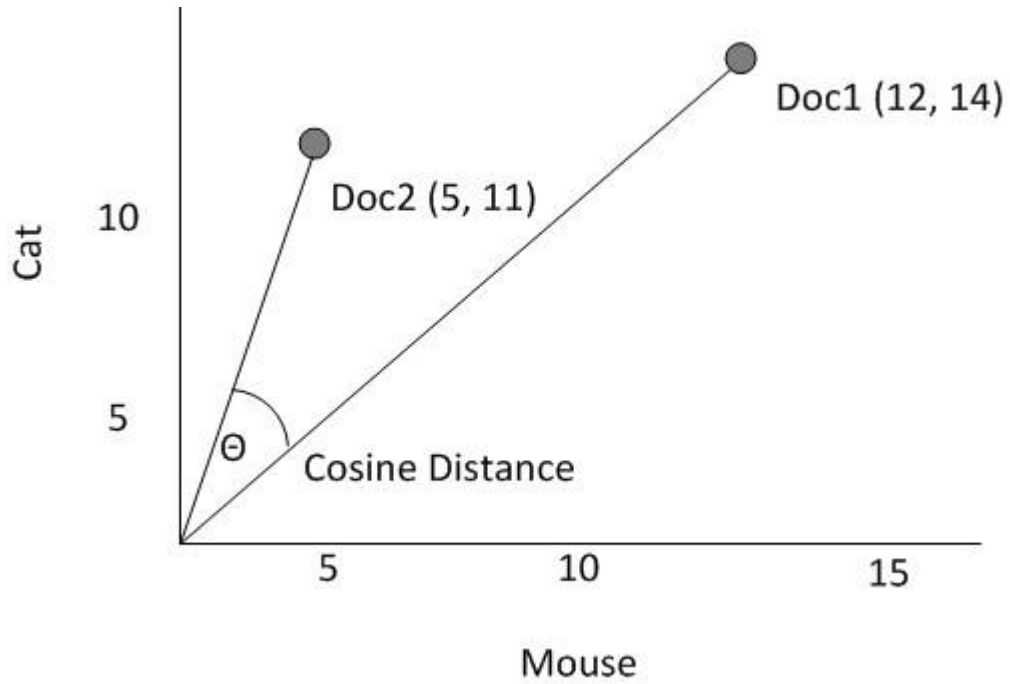
Antes de ver cómo realizar una recomendación, va a ser necesario entender la similitud coseno. A nosotros nos va a importar saber qué tan diferentes son los usuarios o qué tan diferentes son los items, para ello necesitamos una medida de distancia.

Para calcular la distancia entre los usuarios (o entre los items) podemos pensarlos como vectores en el espacio.

Por ejemplo, podemos querer comparar un usuario que puntuó (A=5, B=4, C=5), con un usuario que puntuó (A=5, B=3, C=2).

Sin embargo, **no nos interesa tomar las cantidades en consideración** sino la proporción. Es decir, no nos importa la **magnitud** de los vectores, sino su dirección y sentido. Por este motivo, vamos a emplear la similitud coseno.





En este caso el documento 1 cuenta con casi la misma cantidad de palabras “cat” y “mouse”, por eso vemos que se puede representar como una recta similar a una de 45˚. Por otro lado, el documento 2 tiene más de dos veces “cat” que “mouse”, lo cual da lugar a una recta más inclinada.

La similitud coseno entonces capta el ángulo entre ambos vectores

La similitud coseno se calcula de la siguiente manera:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Como se vé, se requiere sumar los componentes de cada uno de los vectores para la posición i, y dividirlo por el producto de los módulos de los vectores.

Ahora bien, veamos cuáles son los pasos a seguir para poder realizar un filtro User-item (o user-based)

En este caso los pasos a seguir son:

1- Calcular la similaridad entre los usuarios

2- Predecir los ratings de los items que el usuario X no puntuó

3- Seleccionar K mejores películas

1- Calcular la similaridad entre los usuarios

Como vimos antes, vamos a emplear la similaridad coseno. Calculemos cuánto nos da la distancia entre Nico y Fran. Recordemos que :

Los ratings de Nico fueron (5, 4, 4, ?, ?) y los de Fran (4, 5, 5, ?, 3). Los casos que en alguno de los dos tiene un nulo no los podemos incluir, entonces y, siguiendo la fórmula vista, la similitud se calcula como:

```
(5*4 + 4*5 + 4*5)/(np.sqrt(5**2+4**2+4**2)*np.sqrt(4**2+5**2+5**2)) = 0.978
```

Así, calculamos las similitudes de Nico contra todos los demás.

2- Predecir los ratings

Una vez que calculamos las similitudes de un usuario contra los demás necesitamos generar la predicción. Para ello lo que hacemos es un promedio de las puntuaciones de las personas que puntuaron ese item ponderado por la similitud. Luego, multiplicamos por un factor de normalización. Dado un usuario c y un item s, la predicción para c de s será:

$$r_{c,s} = k \sum sim(c, c') . r_{c',s}, c' \epsilon C$$

$$k = 1 / \sum |sim(c, c')|, c' \epsilon C$$

2- Predecir los ratings

¿Cómo lo aplicamos en nuestro caso?

Supongamos que queremos predecir A Study in Scarlet (Study) in para Nico, lo que tenemos que hacer es entonces:

R (Nico, Study) = K x [sim(Nico, Leo) x R(Leo, Study) + sim(Nico, Fran) x R(Fran, Study)]

Con K = 1/ [|sim(Nico, Leo)| + |sim(Nico, Fran)|]

Si realizamos las cuentas nos da

R (Nico, Study) = K x [0.97x5 + 0.98 x 3] con K = 1/ [0,97 + 0,98]

R (Nico, Study) = 0.51 x 7,79 = 4

3- Elegir los k mejores libros

En este caso tenemos pocos libros, pero si tuviéramos muchos items deberíamos elegir los k mejores y mostrar solamente eso.

Por otra parte, los pasos a seguir para poder realizar un filtro Item-item (o item-based)

En este caso los pasos a seguir son:

1- Transponer la matriz user-item

2- Calcular la similitud entre los items y construir la matriz de similitudes

3- Predecir

4- Seleccionar K mejores películas

1- Transponer la matriz

	Nico	Leo	Fran
The Name of the Wind	5	?	4
The Lord of the Rings	4	3	5
Introduction to Statistical Learning	4	5	5
Brave New World	?	5	?
A Study in Scarlet	?	5	3

2- Calcular matriz de similitudes

	The Name of the Wind	The Lord of the Rings	Introduction to Statistical Learning	Brave New World	A Study in Scarlet
The Name of the Wind					
The Lord of the Rings	0.975				
Introduction to Statistical Learning	0.975	0.975			
Brave New World	?	1	1		
A Study in Scarlet	1	0,88	0,97	1	

3- Predecir los ratings

Una vez que calculamos las similitudes entre items podemos aplicar la misma fórmula que antes pero pensando el problema al revés.

Volvamos al caso de predecir A Study in Scarlet para Nico. Ahora en vez de hacerlo partiendo de los ratings de otras personas para Study lo vamos a hacer tomando los ratings de los items que Nico sí puntuó y viendo las similitudes entre estos items y Study. Entonces sería:

R (Nico, Study) = K x [sim(Name, Study) x R(Nico, Name) + sim(Lord, Study) x R(Nico, Lord) + sim(Statistical, Study) x R(Statistical, Lord)] con K = 1/ [|sim(Name, Study)| + |sim(Lord, Study)|+|sim(Statistical, Study)|]

R(Nico, Study) = K x [1x5 + 0,88 x 4 + 0,97 x 4] con K = 1 / [1 + 0,88 + 0,97]

R(Nico, Study) = 4,35

3- Elegir los k mejores libros

Como en el user-based filtering tendremos que elegir los k mejores items.

¿Qué ventajas tiene uno sobre otro?

La principal ventaja de IBCF contra UBCF es que en el primer caso se puede computar la matriz de similitudes de items de manera offline y tener precalculado ese paso. Esto permite luego generar predicciones online de manera mucho más rápida que en el caso de UBCF.

No es conveniente realizar este proceso de cómputo de similitudes offline en el caso de los usuarios porque:

1- La cantidad de usuarios suele ser mucho mayor que la de items. Computar ésto haría al proceso muy costo.

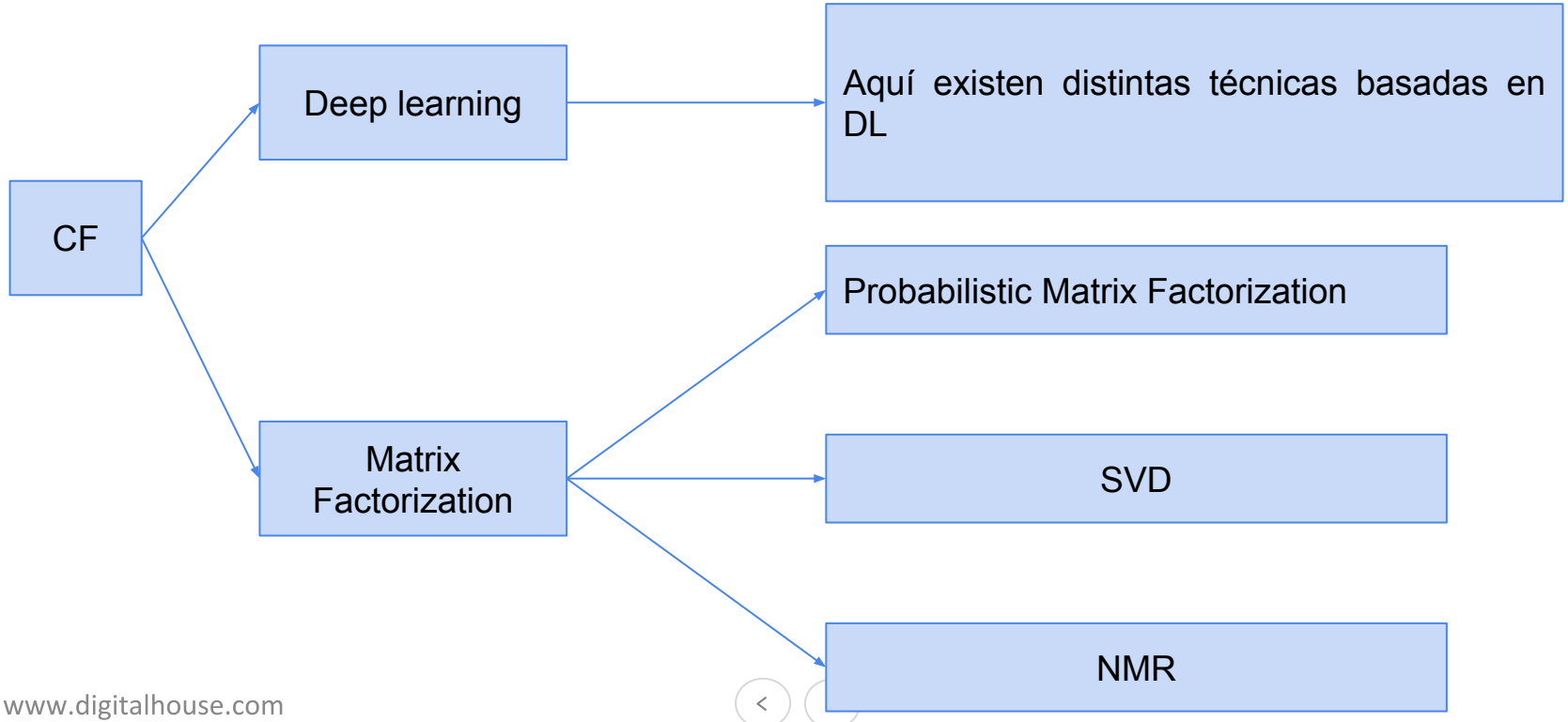
2- Cuando un usuario puntúa un item, sus similitudes con otros usuarios se ven drásticamente afectadas, ya que un usuario suele puntuar pocos items. Por otra parte, esto afecta poco a la matriz de items a items, ya que cada item es probable que haya recibido puntuaciones de diversos usuarios.

Práctica Guiada

Model-based CF

Clasificación

Esta clasificación no intenta ser exhaustiva sino dar un pantallazo de las principales técnicas



SVD

SVD (Singular Value Descomposition) es un método de factorización algebraica que tiene diversas aplicaciones, una de las aplicaciones más famosas es para SR.

Dado una matriz A se puede demostrar que:

$$A = U\Sigma V^T$$

Donde U y V son ortogonales y Sigma es una matriz diagonal de valores singulares. Los valores singulares de A equivalen a la raiz de los autovalores de A:

$$\sigma_i = \sqrt{\lambda_i}$$

SVD

$\sum$ por ser una matriz diagonal opera como escalando una de las matrices vecinas, por este motivo vamos a suponer que la juntamos con alguna de ellas.

Entonces, ahora podemos reescribir nuestro problema como:

$$R = P.Q$$

- Cada fila de P es específica al usuario u
- Cada columna de Q es específica al item i

SVD

Resolver el problema anterior es imposible si la matriz es incompleta, lo cual es el caso normal ya que los usuarios nunca puntúan todos los items. Por este motivo se replantea el problema de la siguiente manera:

$$\hat{r}_{ui} = \mu + b_i + b_u + p_u q_i$$

Donde r es la predicción del ranking, para el usuario u y el item i. μ es un parámetro constante de sesgo, b_i es un sesgo para el item i, b_u es un sesgo para el usuario u. $p_u q_i$ es la factorización de la matriz original.

SVD

Para encontrar los parámetros se emplea descenso por el gradiente y se minimiza la suma de errores cuadráticos. Además, en el problema se agrega un parámetro de regularización. Sólo se toman en consideración los errores de casos completos, los casos nulos son omitidos para el ajuste de lo parámetros. Entonces el problema a resolver es:

$$Min \sum (r_{ui} - \mu - b_i - b_u - p_u q_i)^2 + \lambda(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2)$$

SVD

Para encontrar los parámetros se emplea descenso por el gradiente y se minimiza la suma de errores cuadráticos. Además, en el problema se agrega un parámetro de regularización. Sólo se toman en consideración los errores de casos completos, los casos nulos son omitidos para el ajuste de lo parámetros. Entonces el problema a resolver es:

$$Min \sum (r_{ui} - \mu - b_i - b_u - p_u q_i)^2 + \lambda(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2)$$

$\hat{r}_{ui}$

hiperparámetro de regularización

Cold start

¿Qué significa?

Cold start es el problema de no saber qué recomendarle a un usuario la primera vez o de contar con un item nuevo que aún no ha sido puntuado.

Para resolver esto existen distintas estrategias:

- Recomendaciones basadas en el contenido. En este caso se intenta buscar productos con las mismas características, calculando la similitud entre los productos. Cómo medir esta similitud va a variar de acuerdo al problema, en los documentos se puede hacer calculando la similitud coseno entre los vectores de palabras.
- Pedirle al usuario que nos dé cierta información sobre gustos.
- Dar una sugerencia basada en la popularidad de los items en la plataforma.
- Usar la información de contexto: ¿desde dónde se conecta, cuál es su género, etc?

Métricas de evaluación

En SR se emplean métricas vistas previamente como accuracy, mae, rmse, etc. Por otra parte, existen métricas específicas entre las que vale la pena mencionar a FCP (fraction of concordant pairs o fracción de pares concordantes).

FCP es la proporción de pares de items que fueron correctamente rankeados, es decir, que son concordantes, sobre el total de pares. Esto es, cuando para un usuario u, un item i y un item j se cumple que: $\hat{r}_{ui} > \hat{r}_{uj}$ y que $r_{ui} > r_{uj}$

Análisis de Netflix

Ingreso

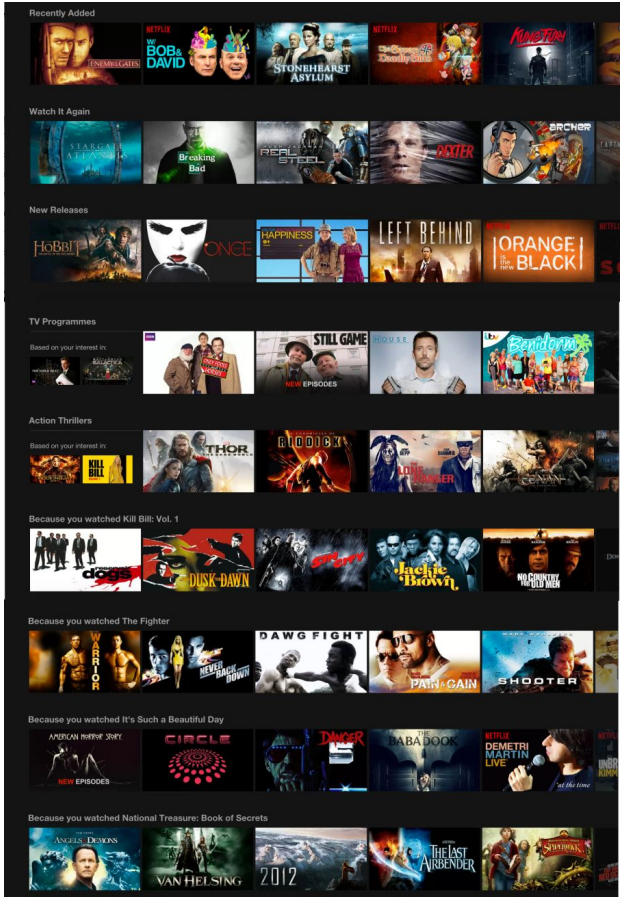
Es importante entender que no sólo el modelo del sistema de recomendación es relevante para el usuario sino todo el contexto, es decir, la UX completa. Por este motivo nos detendremos brevemente a entender la plataforma de Netflix.

En primer lugar, la plataforma nos requiere identificar qué perfil vamos a usar. Esto va a permitirle a la plataforma cargar los datos históricos del usuario.

Pantalla principal

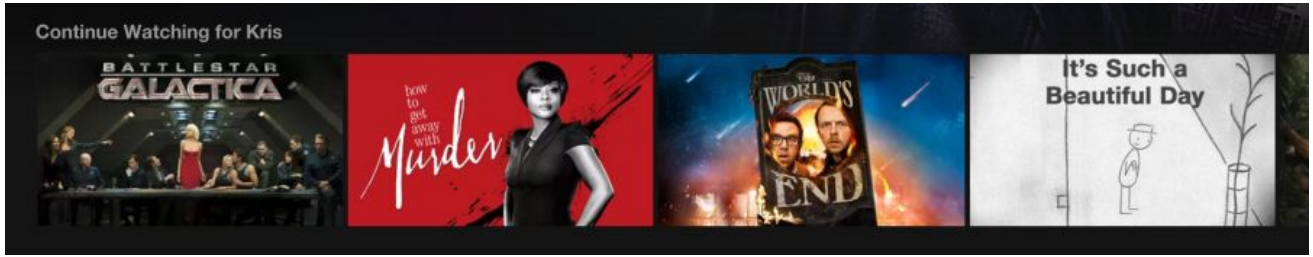
En esta pantalla es donde Netflix nos sugiere qué producciones podrían ser de nuestro interés.

Aquí es donde se mezclan las recomendaciones personalizadas que surgen de los modelos y las que provienen de reglas de negocios, así como otras recomendaciones no personalizadas.

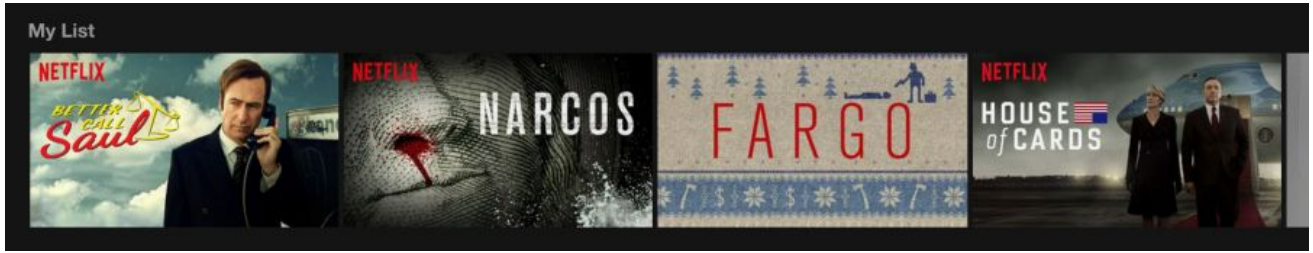


Sugerencias sin modelos

Sugerencia de seguir viendo algo que el usuario ya vio



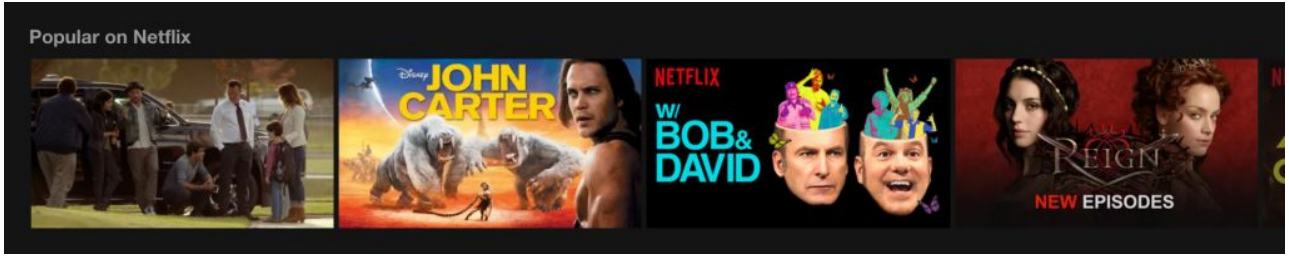
Lista que el usuario mismo genera



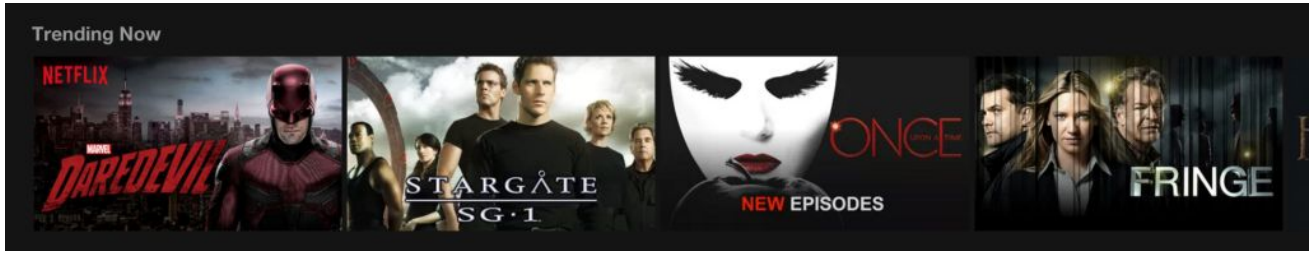
Basados en popularidad:

Si bien parecen genéricos, están personalizados

Popular



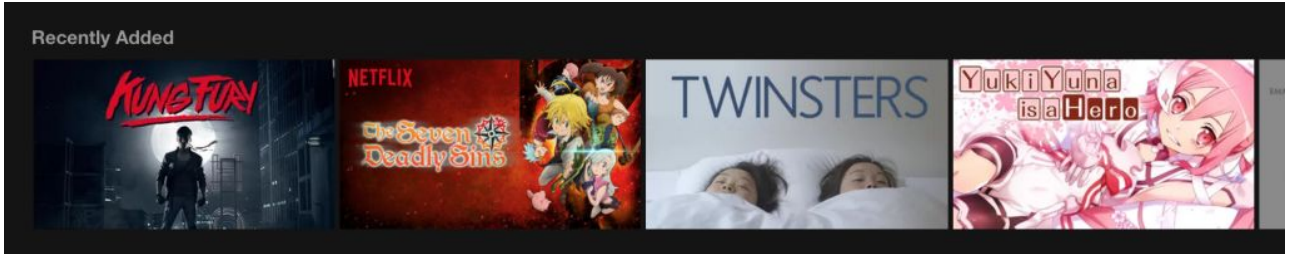
Tendencias



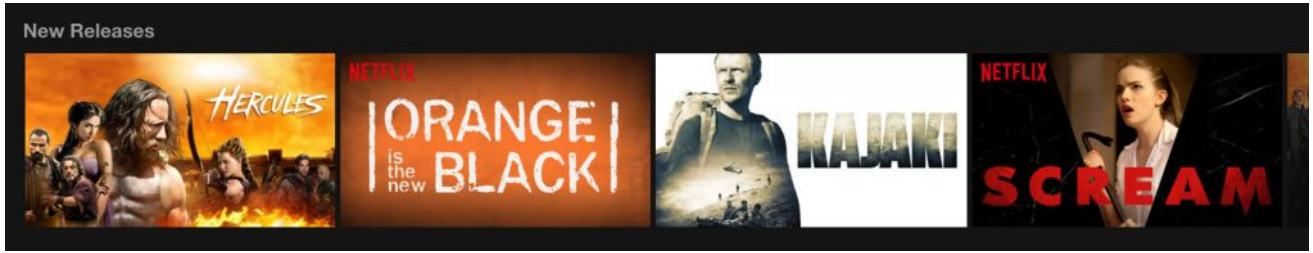
Según recencia

También están personalizados

Recientemente agregados

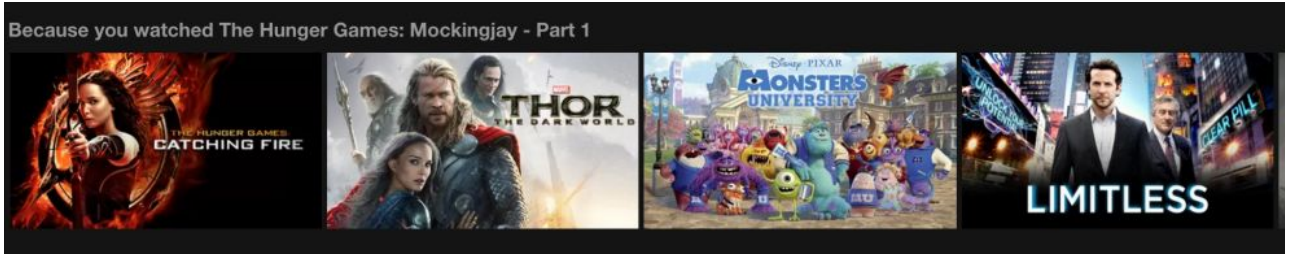


Nuevos lanzamientos



Otros

Porque viste...



Netflix permite buscar en géneros y subgéneros y allí ordena las producciones basándose en algún motor de recomendación