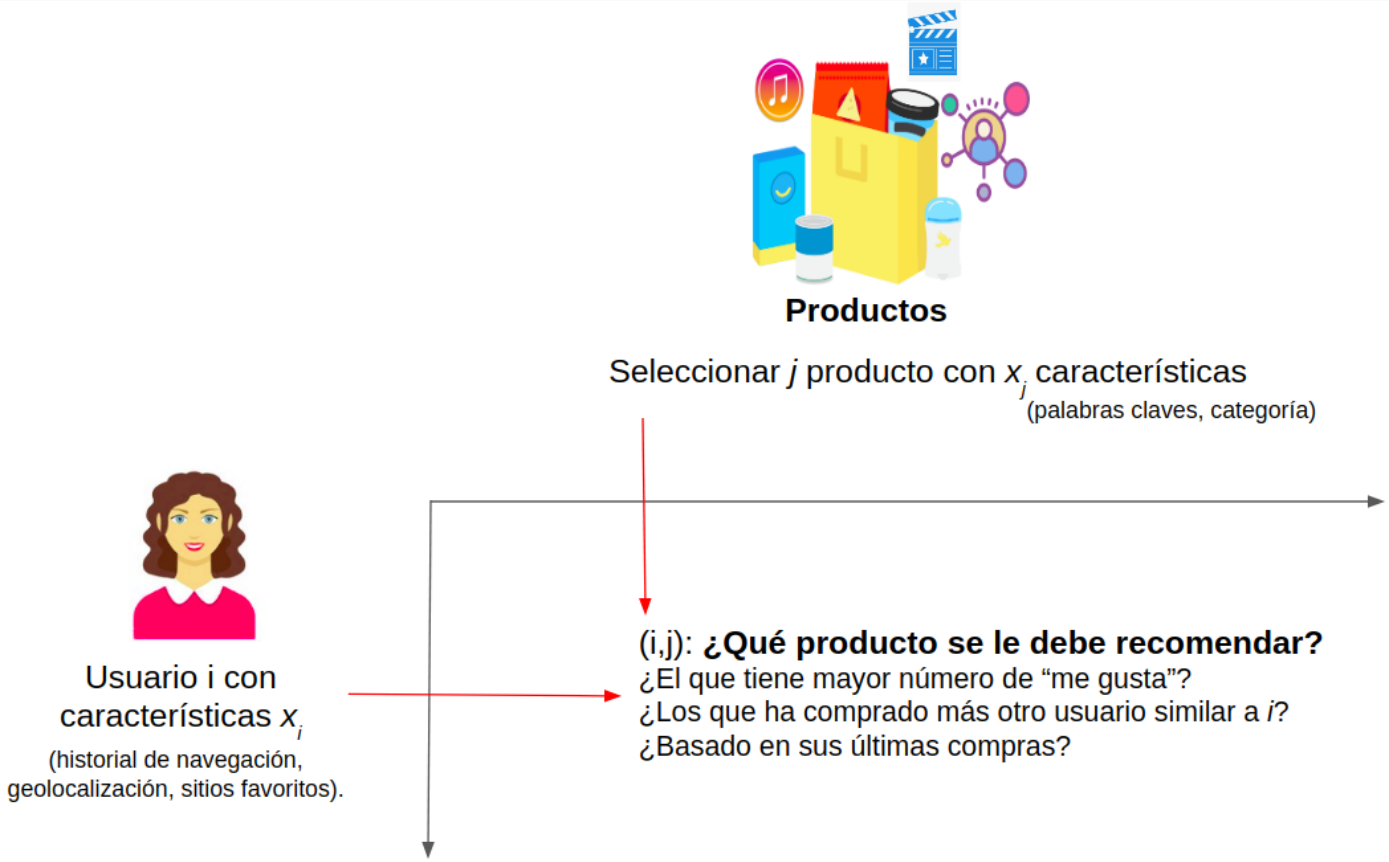
Unidad 4: Sistemas de recomendación

Modelado de filtrado colaborativo y descomposición de matrices

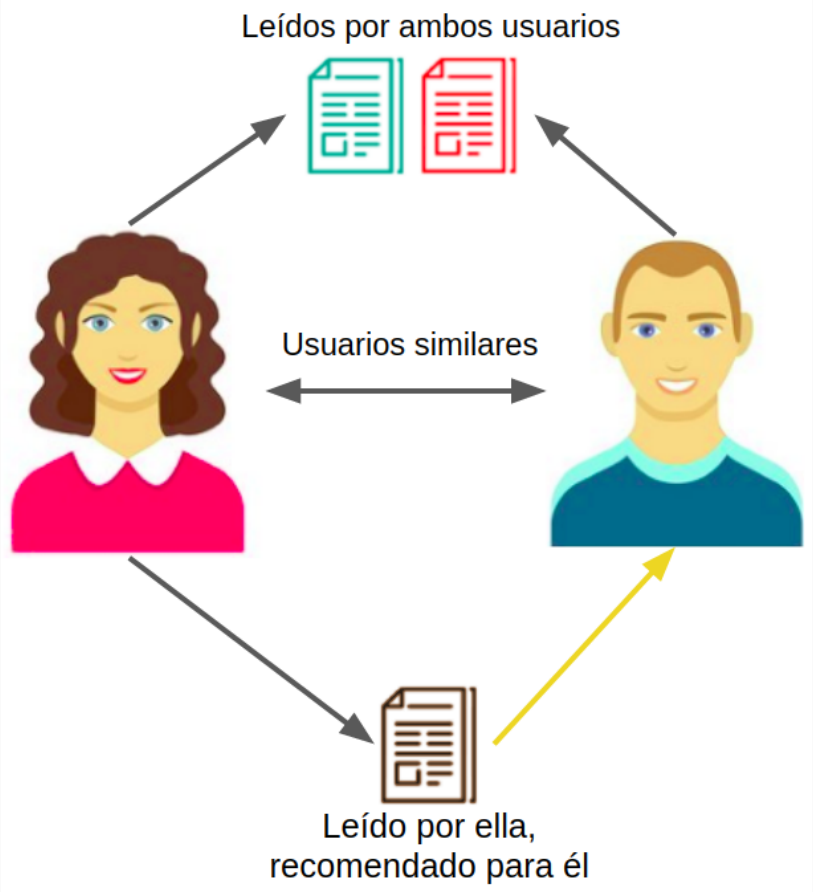
Blanca Vázquez y Gibran Fuentes-Pineda

24 de octubre de 2022

Generalidades



Modelo de filtrado colaborativo



La similitud entre usuarios puede ser usada para inferir las calificaciones no observadas.

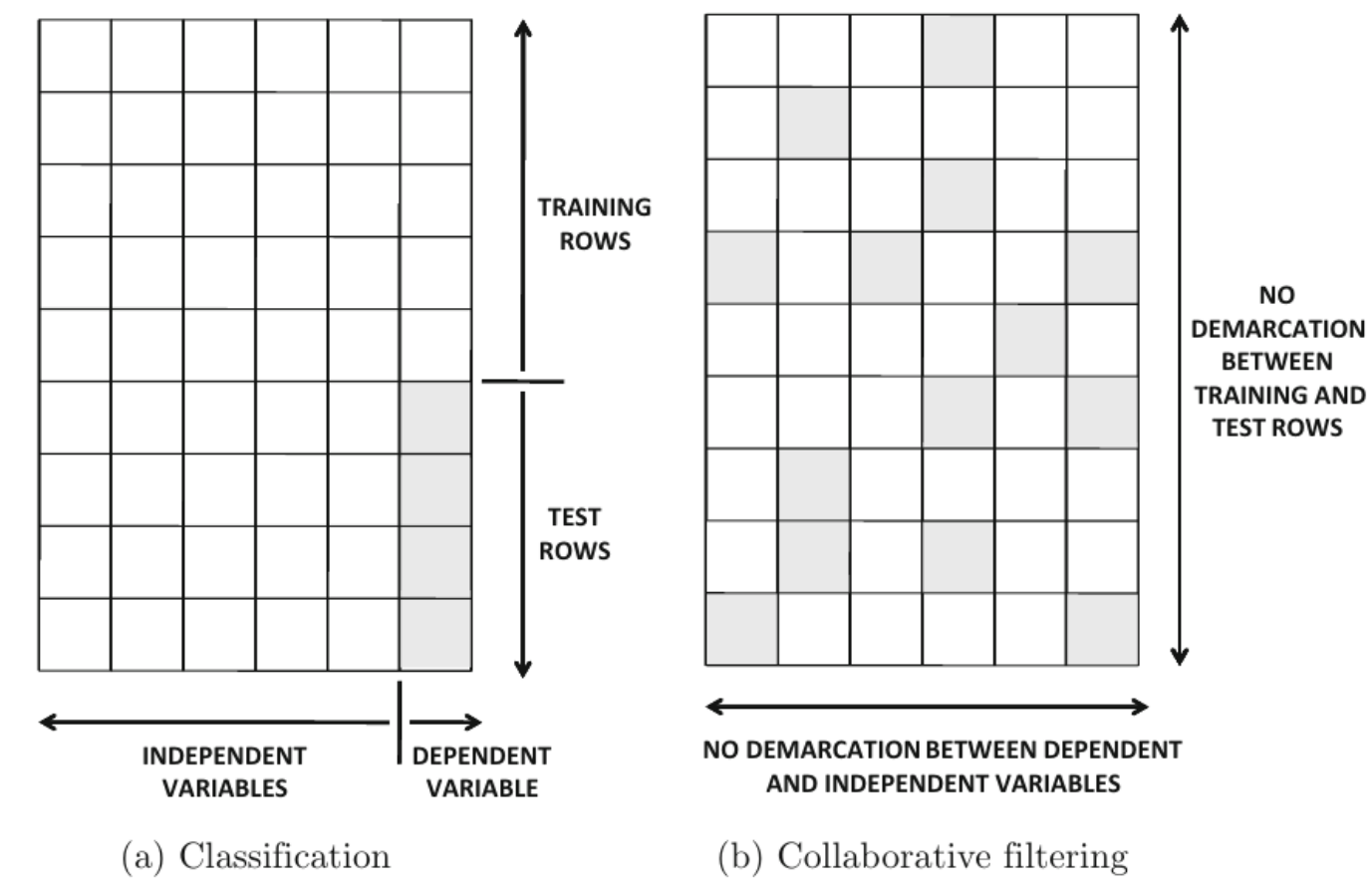
Modelos de filtrado colaborativo

- Emplean las calificaciones obtenidas por los usuarios para la generación de las recomendaciones.
- El principal reto de este modelo es la dispersión de las matrices (valores observados y no observados).

	Usuario 1	Usuario 2	....	Usuario n
P_1	4	?	?	1
P_2	?	3	?	?
...	?	1	?	?
P_n	2	?	?	3

Filtrado colaborativo

Puede ser visto como una generalización del problema de clasificación y regresión.



Comparando el problema de clasificación tradicional con el filtrado colaborativo. Imagen tomada de Aggarwal, 2016.

Problema de clasificación

Clasificación	Filtrado
-Separación entre variables independientes y dependiente (clase).	- Cada columna puede ser independiente o dependiente (dependerá sobre lo que se quiere predecir)
- Clara separación entre datos de de entrenamiento y de prueba.	- En el mejor de los casos, los valores observados (serán los de entrenamiento) y los no observados (serán los de prueba).
- Las columnas representan características y las filas instancias.	- Debido a la naturaleza de los datos, se generan los modelos basados en productos o en usuarios.

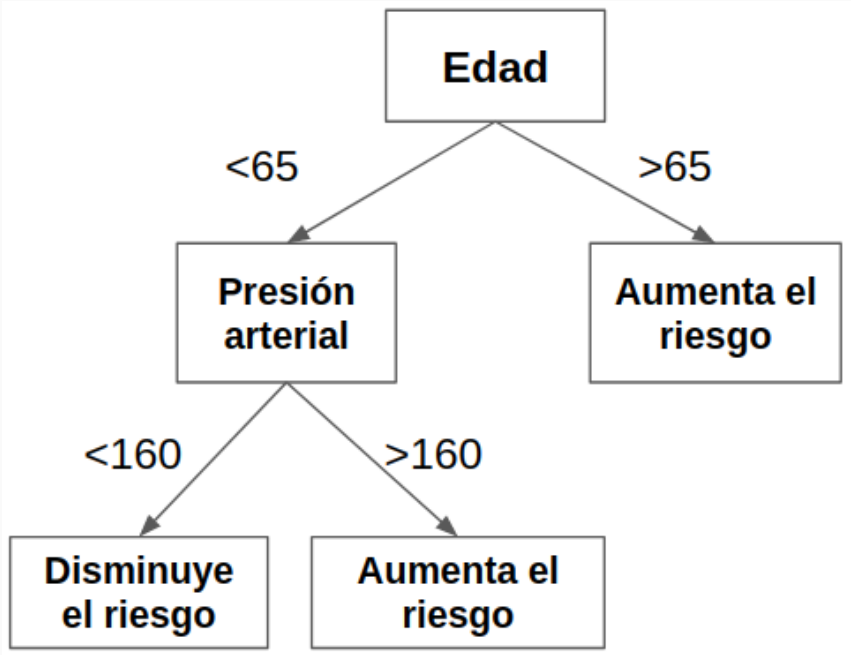
Problema de clasificación

Debido a la generalidad del filtrado colaborativo, diferentes métodos de aprendizaje de máquinas son usados en la generación de recomendaciones:

- Métodos basados en ensambles (árboles de decisión y regresión).
- Naive Bayes
- Redes neuronales
- Modelos de factor latente

Árboles de decisión y regresión

Son una partición jerárquica del espacio de datos a partir de un conjunto de criterios de decisión en las variables independientes.



Retos en la construcción de árboles de decisión

Los principales desafíos en la extensión de los árboles de decisión en el filtrado colaborativo son:

- Los datos no están separados en forma de variables independientes y clase.
- La matriz de puntuaciones es mayormente dispersa y con datos ausentes.
- Dado que no es claro, las variables independientes y dependientes, ¿qué variable debe predecir el árbol?

Retos en la construcción de árboles de decisión

¿Qué variable debe predecir el árbol?

- Se construyen árboles de decisión separados para predecir el *rating* por cada producto.
- Supongamos, una matriz de *mxn* (*m* usuarios y *n* productos), el *rating* del producto que se quiera predecir será la variable **dependiente** y el resto **independiente**.
- El número de árboles a construir será igual al número de productos de la matriz.

Ausencia de datos en los árboles de decisión

Sin embargo, uno de los principales retos de los árboles de decisión en el filtrado colaborativo es: **la ausencia de datos.**

Para atender este problema:

- Se usan métodos de reducción de dimensionalidad
- Posteriormente, se calcula la matriz de covarianza usando los vectores propios.
- La representación reducida se usa para la construcción de los árboles, como si fuera un problema de clasificación.

Filtrado colaborativo basado en reglas

Se aborda la generación de recomendaciones basado en **reglas de asociación.**

Item ⇒ / Customer ⇓	*Bread*	*Butter*	*Milk*	*Fish*	*Beef*	*Ham*
Jack	1	1	1	0	0	0
Mary	0	1	1	0	1	0
Jane	1	1	0	0	0	0
Sayani	1	1	1	1	1	1
John	0	0	0	1	0	1
Tom	0	0	0	1	1	1
Peter	0	1	0	1	1	0

$$\{Butter, Milk\} \Rightarrow \{Bread\}$$

Imagen tomada de Aggarwal, 2016.

Aprovechando las reglas de asociación en el filtrado colaborativo

Cuando el número de productos es pequeño, cada valor de la combinación producto - calificación puede ser tratado como un **pseudo-producto**.

Ejemplo:

- {Producto = leche, calificación = 'me gusta'} $\Rightarrow$ {Producto = 'pan', calificación = 'me gusta'}

Las reglas de asociación son construidas en términos de pseudo-productos.

Filtrado colaborativo basado en reglas

Ejemplo de reglas basadas en **pseudo-productos:**

- {Producto = mantequilla, calificación = ‘me gusta’} AND {Producto = leche, calificación = ‘me gusta’} $\Rightarrow$ {Producto = ’pan’, calificación = ‘me gusta’}

Ejemplo de reglas basadas en **pseudo-usuarios:**

- {Usuario = Pepe, calificación = ‘me gusta’} $\Rightarrow$ {Usuario = ’Paco’, calificación = ‘no me gusta’} AND {Usuario = ’Lola’, calificación = ‘me gusta’}

Recordemos

- El objetivo de los sistemas de recomendación es **proporcionar un conjunto de sugerencias útiles a un usuario final.**
- Para generar las recomendaciones, frecuentemente se usa la **similaridad** entre productos, usuarios o ambos.

Introducción a los modelos de factor latente

- Usan los métodos de reducción de dimensionalidad para llenar los valores ausentes en la matriz de *ratings*.
- La idea básica es explotar que filas y columnas en una matriz están altamente correlacionadas (matriz de bajo rango).
- Objetivo: completar los valores ausentes usando la matriz de bajo rango.

Intuición

Los modelos de factor latente intentan explicar los *ratings* de los usuarios a partir de inferir patrones de productos y usuarios.

Características de la factorización de matrices

Se busca identificar patrones entre usuarios y productos, para lograrlo se calculan los vectores latentes (a través de factorización de matrices).

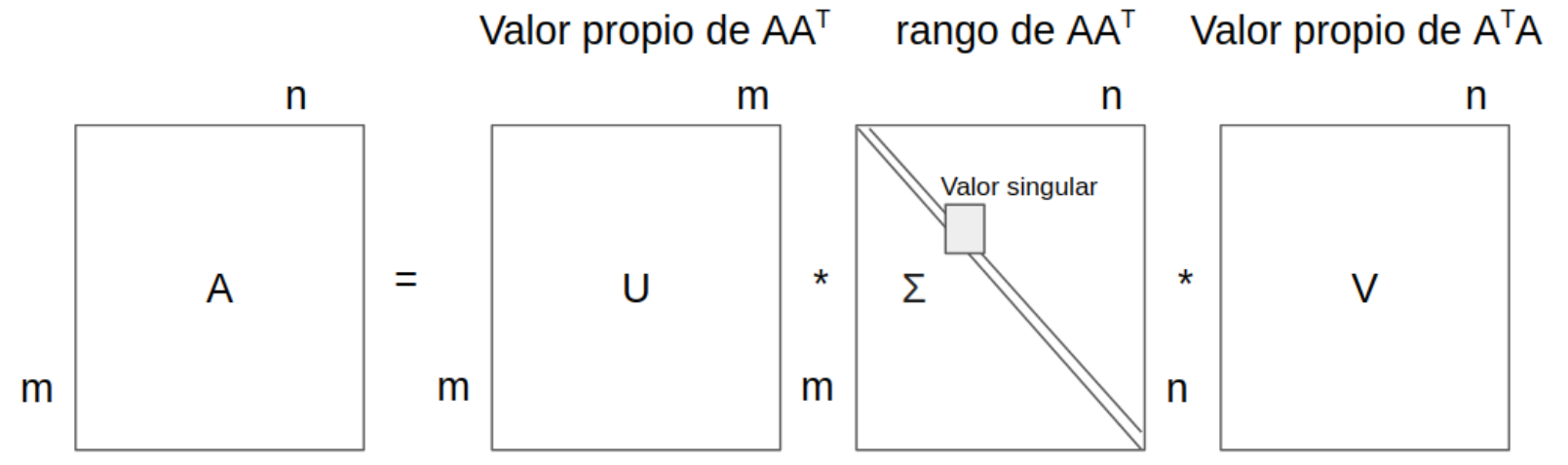
- Tiene buena escalabilidad
- Se obtienen predicciones más precisas
- Flexibilidad

¿Cómo generar los vectores latentes?

- Usar descomposición de valores singulares (SVD)
- Análisis de componentes principales (PCA)

Descomposición de valores singulares

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



Desventaja de SVD

- La SVD generalmente se emplea en matrices sin datos ausentes.
- Enfoques previos, usaban el promedio de los *ratings* de los usuarios y productos para llenar las matrices.

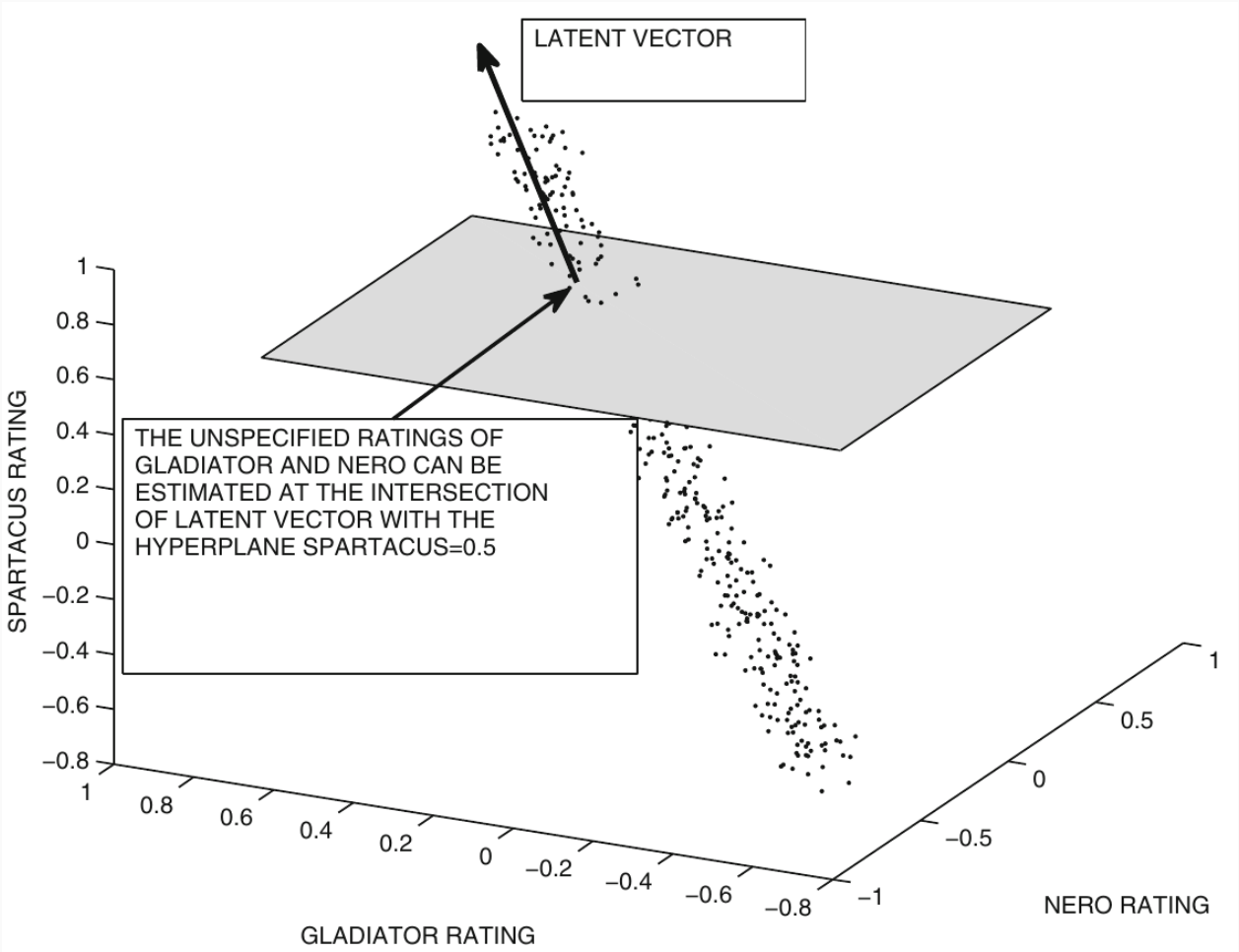
Vectores latentes

- La clave de los modelos de factor latente es la habilidad para estimar los vectores latentes con datos ausentes.
- Recordemos que partimos del supuesto que los elementos en la matriz están altamente correlacionados, por lo tanto, estas redundancias ayudan a reconstruir los valores ausentes.

Ejemplo

- Supongamos que tenemos una matriz con 3 películas, en la cual los 3 productos están positivamente correlacionados.
- Películas: Nerón, Gladiador y Espartaco.
- Los *ratings* son valores continuos [-1,1].
- Si los *ratings* están positivamente correlacionados, entonces la gráfica de dispersión tridimensional de las 3 películas sería una línea unidimensional.

Ejemplo



Aprovechar las redundancias basadas en la correlación para estimar los datos faltantes para un usuario cuya única calificación especificada es un valor de 0.5 para la película Spartacus.
Imagen tomada de Aggarwal, 2016.

Intuición

- La intuición geométrica vista en el ejemplo anterior es útil, cuando los vectores latentes son mutuamente ortogonales.
- Sin embargo, eso no siempre sucede.

Factorización de matrices

- ¿Cómo calcular los vectores latentes cuando no son mutuamente ortogonales?
- La **factorización** es una manera general de aproximación de una matriz cuando es propensa a la reducción de dimensionalidad (correlaciones entre columnas o filas).

Principios básicos

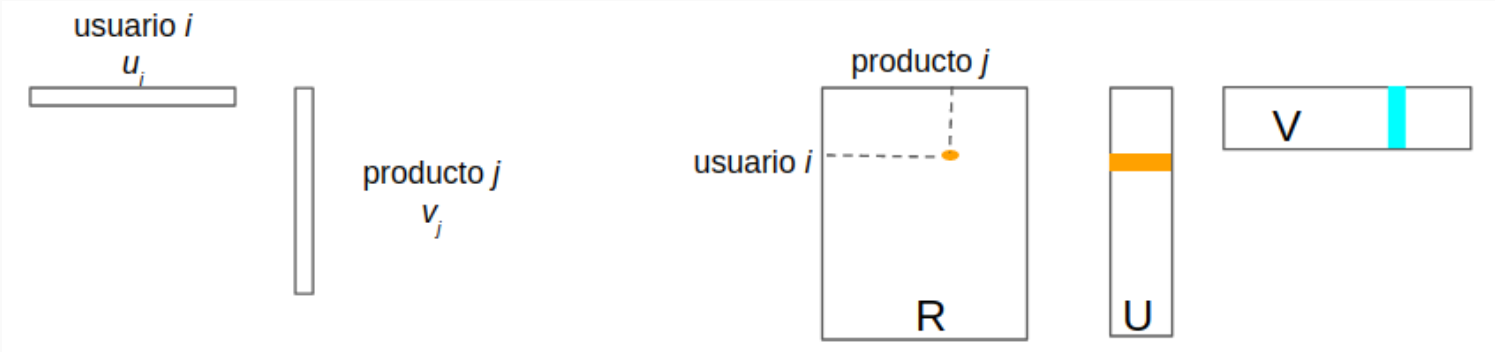
Dada una matriz de *ratings* R de mxn, esta matriz puede ser factorizada en 2 matrices:

- Matriz U: mxk
- Matriz V: nxk

$$R \approx UV^T$$

- Cada columna de U o V se refiere como vector latente o componente latente
- Cada fila de U o V se refiere como factor latente.

Cálculo de los vectores latentes



$$R \approx UV^T$$

donde r_{ij} es la calificación que el usuario i le da al producto j y es del tamaño mxn, k es el rango, u_i es el vector del usuario i y v_j es el vector del producto j.

Factorización de matrices

$$R \approx UV^T$$

- Cada fila $\bar{u}_i$ en la matriz U se denomina **factor de usuario**.
- Cada fila $\bar{v}_i$de la matriz V se denomina **factor de producto**.
- Objetivo final: modelar cada producto y usuario como un vector de factores.

Por lo tanto, para predecir el *rating* r_{ij} en R puede ser expresado como:

$$r_{ij} \approx \bar{u}_i . \bar{v}_j$$

Factorización de matrices

Dado que los factores latentes $\bar{u}_i = (u_{i1}, ..., u_{ik})$ y $\bar{v}_i = (v_{j1}, ..., v_{jk})$ pueden verse como las afinidades de los usuarios por k conceptos diferentes, la ecuación:

$$r_{ij} \approx \bar{u}_i . \bar{v}_j$$

Puede ser expresada como:

$$r_{ij} \approx \sum_{s=1}^{k} u_{is} . v_{js}$$

$= \sum_{s=1}^{k}$ (afinidad del usuario i para el concepto s) * (afinidad del producto j para el concepto s)

Ejemplo

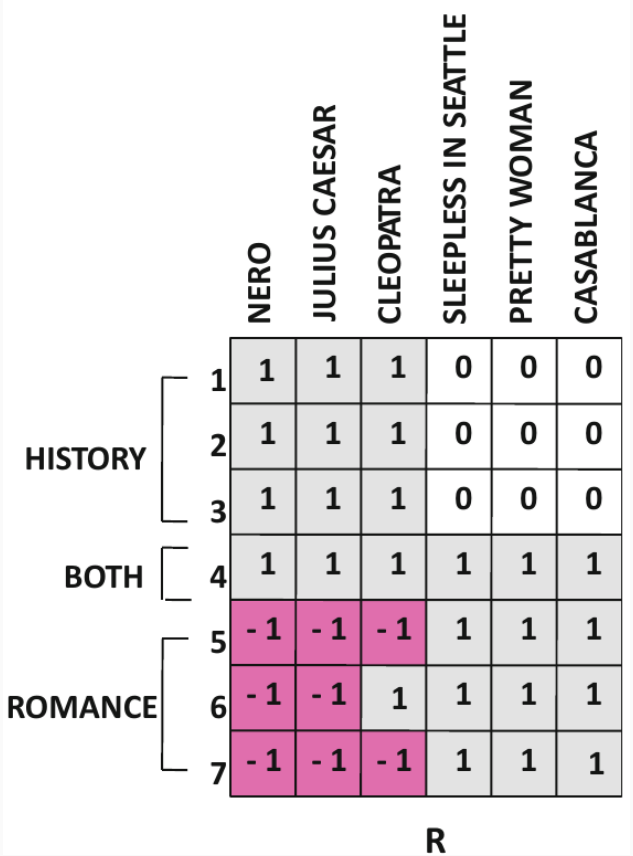


Imagen tomada de Aggarwal, 2016.

Ejemplo

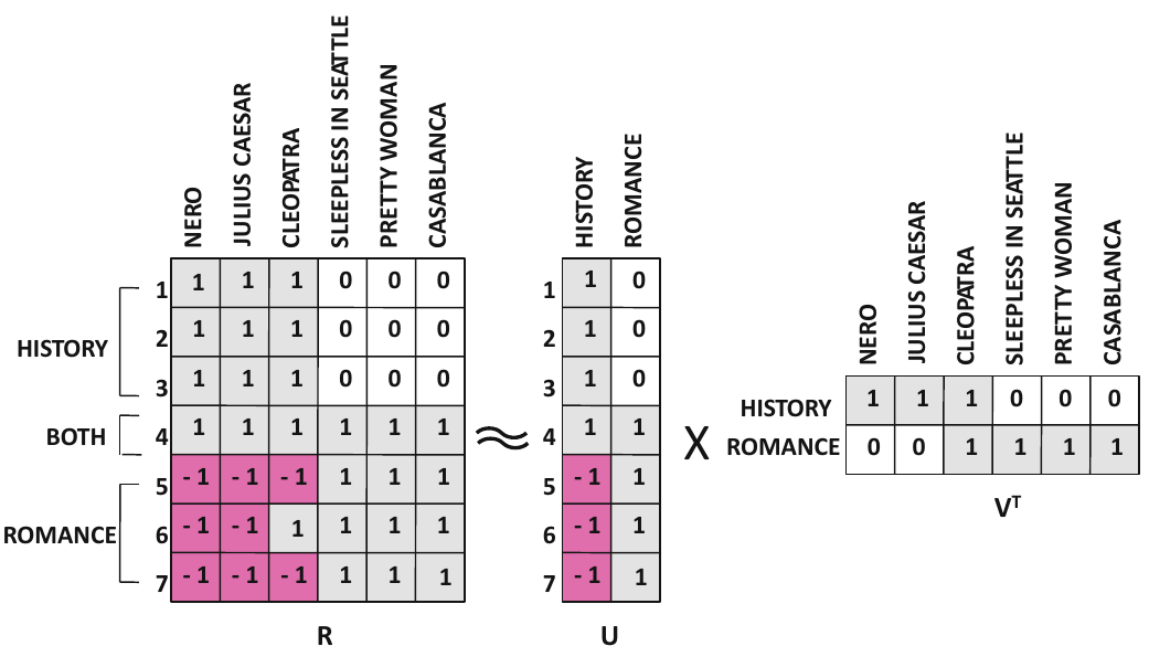
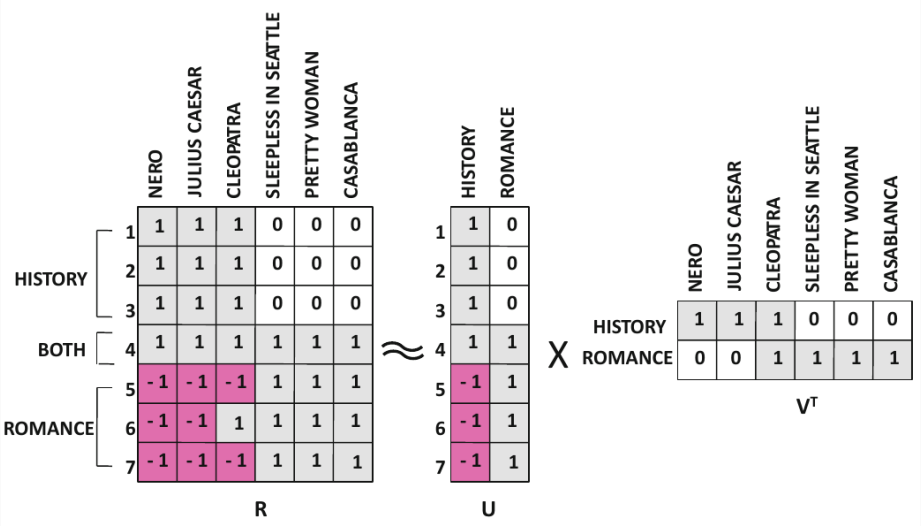


Imagen tomada de Aggarwal, 2016.

Factorización de matrices

Los conceptos se refieren a los géneros romántico e histórico.



Por lo tanto, la ecuación

$$r_{ij} \approx \sum_{s=1}^{k} u_{is}.v_{js}$$

Puede interpretarse como:
$r_{ij} \approx$ (afinidad del usuario *i* para *historia*) * (afinidad del producto *j* para *historia*) + (afinidad del usuario *i* para *romance*) * (afinidad del producto *j* para *romance*).

Importante:

- La noción de concepto (ej., romance e historia) frecuentemente no tienen una interpretación semántica.
- Un vector latente comúnmente es más parecido a esto:

$$\begin{vmatrix} 1 \\ -2.5 \\ 3 \\ -0.2 \end{vmatrix}$$

- Lo importante es: un vector latente describe los patrones de correlación en la matriz R.

Principales diferencias entre métodos

Existen muchos métodos de factorización de matrices, las principales diferentes son:

- Las restricciones de las matriz U y V
  - Ortogonalidad o no negatividad de los vectores latentes.
- La naturaleza de la función objetivo
  - Minimizar la norma de Frobenius o maximizar la estimación de probabilidad.

Calculando las matrices U y V

Como se ha mencionado previamente, cualquier matriz R puede descomponerse en 2 matrices (U,V), tal que UV^T puede aproximar la matriz original de R (sin valores ausentes).

¿Cómo obtener las matrices U y V que aproximen a R?

Se puede formular como un problema de optimización:

$$minimizar \quad J = \frac{1}{2}||R - UV^T||^2$$

Esta función objetivo se puede ver como una **función de pérdida cuadrática**. Asimismo, métodos del descenso del gradiente pueden dar una solución óptima para esta factorización.

¿Cómo obtener las matrices U y V que aproximen a R con datos ausentes?

Recordemos: las matrices de *ratings* se caracterizan tener **¡datos ausentes!**, es decir, solo un conjunto de entradas tiene valores observados.

- Por lo tanto, la función objetivo anterior, no podría ser aplicada
  - No es posible calcular la norma de Frobenius con entradas faltantes.

¿Cómo obtener las matrices U y V que aproximen a R con datos ausentes?

Solución: la función objetivo debe re-escribirse usando solo las entradas observadas para aprender U y V.

- *Nice part:* una vez aprendidos los factores latentes U y V es posible reconstruir la matriz completa usando UV^T.

¿Cómo obtener las matrices U y V que aproximen a R con datos ausentes?

- Supongamos todos los pares observados (i, j) en R los cuales denotaremos como S.
- Sea $i \in (1, ..., m)$ el índice de un usuario y $j \in (1, ..., n)$ el índice de un producto.

Por lo tanto, el conjunto S de pares observados (usuario-producto) se define:

$$S = \{(i, j) : r_{ij} \quad par \quad observado\}$$

¿Cómo obtener las matrices U y V que aproximen a R con datos ausentes?

Una vez calculadas las matrices U y V usando las entradas observadas, es posible predecir las entradas faltantes:

$$\hat{r_{ij}} = \sum_{s=1}^{k} u_{is}.v_{js}$$

La diferencias entre las entradas observadas y las entradas predichas de una entrada específica (i, j) es:

$$e_{ij} = (r_{ij} - \hat{r_{ij}}) = r_{ij} - \sum_{s=1}^{k} u_{is}.v_{js}$$

Actualización de la función objetivo

Dada una matriz R con entradas faltantes, la función objetivo se calcula sobre las entradas observadas:

$$minimizar \quad J = \frac{1}{2} \sum_{(i,j) \in S} e_{ij}^2 = \frac{1}{2} \sum_{(i,j) \in S} (r_{ij} - \sum_{s=1}^{k} u_{is}.v_{js})^2$$

Nota: la función objetivo suma el error únicamente sobre las entradas observadas.

Actualización de la función objetivo

Cabe señalar que cada uno de los términos:

$$(r_{ij} - \sum_{s=1}^{k} u_{is}.v_{js})^2$$

Es el error cuadrado e_{ij}^2 entre la entrada observada y la entrada predicha (i, j).

Recuerda, u_{is} y v_{js} son las variables desconocidas las cuales necesitan ser aprendidas para minimizar la función objetivo.

Factorización de matrices

- El algoritmo del descenso del gradiente es el método más usado para minimizar la función objetivo.
  - Fácil de implementar
  - El tiempo de ejecución es relativamente rápido