

Recomendador Slope One

IIC 3633 - Sistemas Recomendadores

Denis Parra

Profesor Asistente, DCC, PUC Chile

TOC

En esta clase

1. Resumen últimas clases
2. Control de lectura próxima semana y Tarea
3. Recomendador Slope One
4. Ejemplo pequeño

Resumen últimas clases

- **Ranking no personalizado:** Ordenar items considerando el porcentaje de valoraciones positivas y la cantidad total de valoraciones.
- **Filtrado Colaborativo basado en el usuario:** buscar K usuarios más parecidos, luego predecir rating de items no consumidos por el active user.
- **Filtrado Colaborativo basado en items:** pre-calculan directamente similaridad entre items co-rated, saltándose búsqueda de K-vecindario.

Control de Lectura Próxima semana

Preguntas tipo

- Describir la idea y fundamento del método o algoritmo
- Escribir las fórmulas para cálculo de similaridad y la predicción
- Calcular un ejemplo pequeño (traer calculadora)
- Identificar pros y contras de cada método
- Efecto de diferentes parámetros

Lecturas a Evaluar

- How not to sort by Average Rating, [Evan Miller Blog](#)
- Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994, October). GroupLens: an open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM conference on Computer supported cooperative work (pp. 175-186). ACM.
- Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web (pp. 285-295). ACM.
- Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999, August). An algorithmic framework for performing collaborative filtering. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (pp. 230-237).
- Lemire, D., & Maclachlan, A. (2005, April). Slope One Predictors for Online Rating-Based Collaborative Filtering. In SDM (Vol. 5, pp. 1-5).

Tarea 1

- Será anunciada el martes 19 de Agosto (Próxima semana)
- Considerando un dataset que nosotros les entregaremos, deben implementar los recomendadores vistos hasta ahora y elegir el mejor. Con ese "mejor recomendador" realizarán dos tareas sobre un **Hold Out dataset**: predecir ratings y generar una lista topN de recomendaciones.

PREDICCIÓN

```
user, item, rating
1 user, item, rating
2 1, 3, ?
3 1, 5, ?
4 1, 7, ?
5 10, 12, ?
6 10, 17, ?
7 10, 23, ?
8 .....
```

Line 8, Column 5 Tab Size: 4 Plain Text

RECOMENDACIÓN TOP-N

```
user, item (ranked)
1 user, item (ranked)
2 1, 7
3 1, 2
4 1, 87
5 10, 9
6 10, 54
7 10, 4
8 ...
9
```

Line 7, Column 6 Tab Size: 4 Plain Text

Recomendador de Pendiente Uno

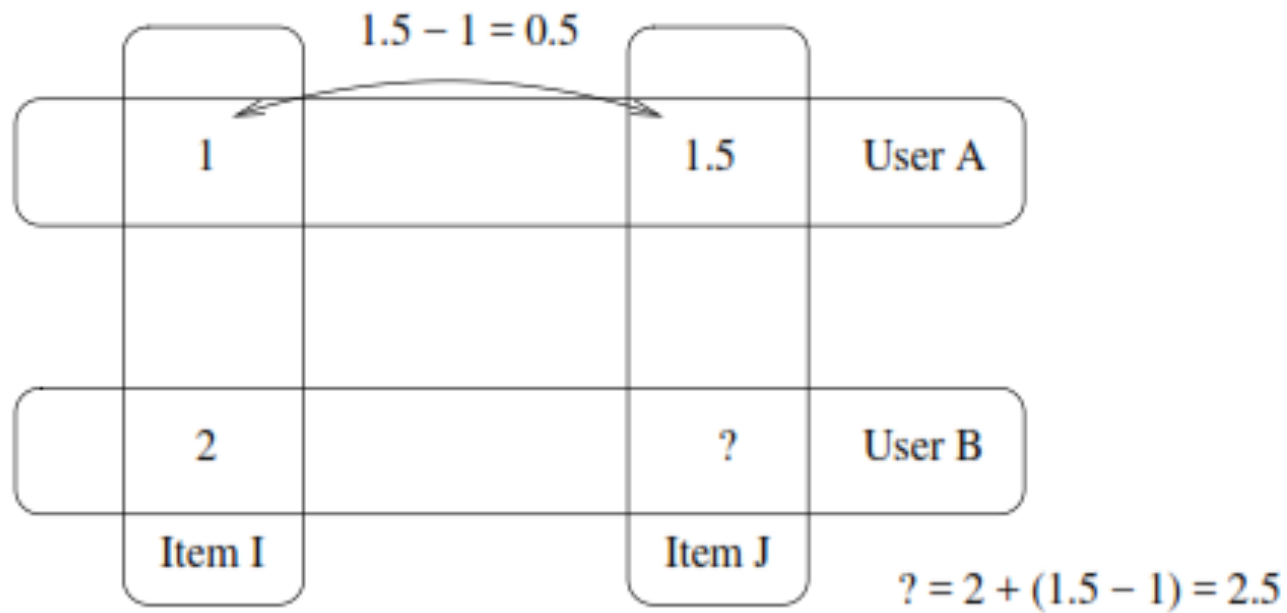
Lemire, D., & Maclachlan, A. (2005, April). **Slope One Predictors for Online Rating-Based Collaborative Filtering**. In SDM (Vol. 5, pp. 1-5).

Los autores se enfocan en 5 objetivos:

1. Fácil de Implementar y mantener
2. Actualizable en línea: nuevos ratings deberían cambiar las predicciones rápidamente.
3. Eficiente al momento de consulta: costo principal debería llevarlo el almacenamiento.
4. Funciona con poco feedback del usuario
5. Razonablemente preciso, dentro de ciertos rangos en los que una pequeña ganancia en exactitud no signifique un gran sacrificio de simplicidad y escalabilidad.

Idea Principal

Si observamos las **diferencias** de ratings entre pares de items, podemos hacer una predicción.



Fórmula principal y Ponderación

- Ecuación Lineal de Pendiente uno

$$f(x) = x + b$$

- Si tenemos dos vectores con ratings v_i y w_i donde $i = 1..n$, buscamos el mejor predictor de la forma $f(x) = x + b$ para predecir w usando v . Luego, tratamos de minimizar la expresión

$$\sum_{i=1}^n (v_i + b - w_i)^2$$

- Derivando con respecto a b e igualando la expresión a 0, nos queda

$$b = \frac{\sum_i (v_i - w_i)}{n}$$

Implementación Normal y Ponderaciones

- Implementación Normal

$$dev_{j,i} = \sum_{u \in S_{j,i}(\chi)} \frac{u_j - u_i}{card(S_{j,i}(\chi))}, \text{ desviación promedio entre dos items } i, j$$

$$P(u)_j = \frac{1}{card(R_j)} \sum_{i \in R_j} (dev_{j,i} + u_i), \text{ predicción en base a varios items } i$$

$$P(u)_j^{S1} = \bar{u} + \frac{1}{card(R_j)} \sum_{i \in R_j} dev_{j,i}, \text{ aproximación cuando } R_j S(u)$$

- Métodos de Ponderación
 - Weighted
 - Bipolar

Métodos de Ponderación

- **Weighted Slope One:** Algunos pares de items han sido evaluados por muchos usuarios, otros no tanto

$$P(u)_j^{wS1} = \frac{\sum_{i \in S(u) - \{j\}} (dev_{j,i} + u_i) c_{j,i}}{\sum_{i \in S(u) - \{j\}} c_{j,i}}, \text{ donde } c_{i,j} = \text{card}(S_{j,i}(\chi))$$

- **Bi-Polar:** los items evaluados positivamente son considerados distintos a los evaluados negativamente.

$$dev_{j,i}^{like} = \sum_{u \in S_{j,i}^{like}(\chi)} \frac{u_j - u_i}{\text{card}(S_{j,i}^{like}(\chi))}, \text{ desviacion media entre dos items liked}$$

$$p_{j,i}^{like} = dev_{j,i}^{like} + u_i, \text{ y de modo similar, } p_{j,i}^{dislike} = dev_{j,i}^{dislike} + u_i$$

Combinando ambos, like y dislike, la predicción queda:

$$P(u)_j^{bpS1} = \frac{\sum_{i \in S(u)^{like} - \{j\}} p_{j,i}^{like} c_{j,i}^{like} + \sum_{i \in S(u)^{dislike} - \{j\}} p_{j,i}^{dislike} c_{j,i}^{dislike}}{\sum_{i \in S(u)^{like} - \{j\}} c_{j,i}^{like} + \sum_{i \in S(u)^{dislike} - \{j\}} c_{j,i}^{dislike}}$$

Resultados en paper de Lemire (2005)

Scheme	EachMovie	Movielens
BI-POLAR SLOPE ONE	0.194	0.188
WEIGHTED SLOPE ONE	0.198	0.188
SLOPE ONE	0.200	0.188
BIAS FROM MEAN	0.203	0.191
ADJUSTED COSINE ITEM-BASED	0.209	0.198
PER USER AVERAGE	0.231	0.208
PEARSON	0.194	0.190

Table 1: All Schemes Compared: All But One Mean Average Error Rates for the EachMovie and Movielens data sets, lower is better.

Ejercicio :-)

(A pedido de A.P.)

- Usando Weighted Slope One, calcule qué rating le daría U3 a Spiderman*:

USER	HARRY POTTER	BATMAN	SPIDERMAN
U1	5	3	4
U2	?	2	4
U3	4	2	?

- Usando Weighted Slope One, calcule qué rating le daría U3 a Spiderman*:

1. Desviación promedio entre Spiderman y Batman (en la resta, primero Spiderman)

$$dev_{j,i} = \sum_{u \in S_{j,i}(\chi)} \frac{u_j - u_i}{card(S_{j,i}(\chi))}, j: \text{Spiderman}, i: \text{Batman}$$

2. Desviación promedio Spider Man y Harry Potter (en la resta, primero Spiderman)

$$dev_{j,i} = \sum_{u \in S_{j,i}(\chi)} \frac{u_j - u_i}{card(S_{j,i}(\chi))}, j: \text{Spiderman}, i: \text{Harry Potter}$$

3. Predicción: Usando el rating que el usuario U3 le dio a Batman y a Harry Potter, calcular rating de U3 a Spiderman

$$P(u)_j^{wS1} = \frac{\sum_{i \in S(u) - \{j\}} (dev_{j,i} + u_i) c_{j,i}}{\sum_{i \in S(u) - \{j\}} c_{j,i}}, \text{ donde } c_{i,j} = card(S_{j,i}(\chi))$$

Usando Weighted Slope One, calcule qué rating le daría U3 a Spiderman*:

- Desviación promedio entre Spiderman y Batman (en la resta, primero Spiderman)

$$[(4 - 3) + (4 - 2)]/2 = 1,5$$

- Desviación promedio SpiderMan y Harry Potter (en la resta, primero Spiderman)

$$[(4 - 5)]/1 = -1$$

- **Predicción:** Usando el rating que el usuario U3 le dio a Batman y a Harry Potter, calcular rating de U3 a Spiderman

- Usando Batman (rating the U3 a Batman + desviación)

$$r = 2 + 1,5 = 3,5$$

- Usando Harry Potter (rating the U3 a Harry Potter + desviación)

$$r = 4 - 1 = 3$$

- Considerando ambas predicciones y la cantidad de muestras usadas en cada uno

$$(3,5 * 2 + 3 * 1)/(2 + 1) = 3,33$$

Referencias

- Lemire, D., & Maclachlan, A. (2005, April). Slope One Predictors for Online Rating-Based Collaborative Filtering. In SDM (Vol. 5, pp. 1-5).
- *Ejercicio sacado de <http://www.slideshare.net/irecsys/slope-one-recommender-on-hadoop-15199798>