User-Based CF II

IIC 3633 - Sistemas Recomendadores

Denis Parra
Profesor Asistente, DCC, PUC CHile

TOC administrativo

1. Presentación de Ivania
2. Mailing list (La creará Ivania)
   iic3633-2016-2@googlegroups.com
3. Blogs
   No olviden enviar hasta este domingo 7 su URL a indonoso@uc.cl
4. Cálculo de Nota Final
   30% Tarea 1 + 30% (Lecturas, Presentación) + 40% Proyecto Final

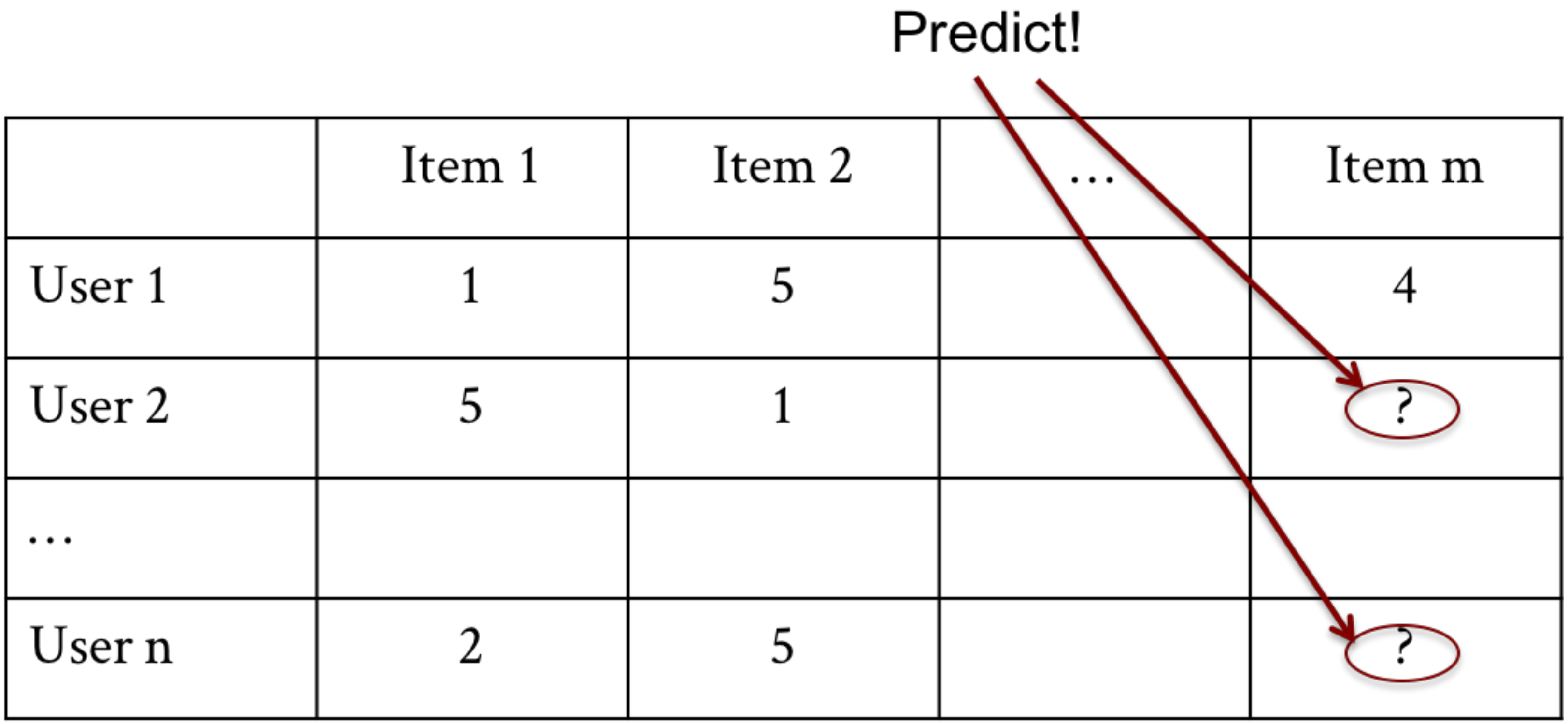
TOC

En esta clase

1. Resumen Clase anterior
2. Por qué otra versión de CF?
3. Filtrado Colaborativo Basado Items (Sarwar et al. 2010)
4. Referencias

Resumen última clase

Recommender Systems aim to help a user or a group of users in a system to select items from a crowded item or information space.



Predict!

	Item 1	Item 2	…	Item m
User 1	1	5		4
User 2	5	1		?
…				
User n	2	5		?

Resumen última clase

Problema de recomendación como predicción de ratings: evaluación

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\hat{r}_{ui} - r_{ui})^2}{n}}$$

$$MSE = \frac{\sum_{i=1}^{n}(\hat{r}_{ui} - r_{ui})^2}{n}$$

$$MAE = \frac{\sum_{i=1}^{n}|\hat{r}_{ui} - r_{ui}|}{n}$$

Resumen última clase II

- Ranking no personalizado: Varias opciones. Si consideramos que los ítems a rankear tienen valoraciones positivas y negativas, el ranking ideal debería considerar la proporción de positivas y la cantidad de muestras consideradas: una opción es el límite inferior del Intervalo de Confianza del Wilson Score, para un parámetro Bernoulli.

$$\left(\hat{p} + \frac{z^2_{\alpha/2}}{2n} \pm z_{\alpha/2}\sqrt{[\hat{p}(1-\hat{p}) + z^2_{\alpha/2}/4n]/n}\right)/(1+z^2_{\alpha/2}/n).$$

- Filtrado Colaborativo (Basado en el usuario): Buscamos los K usuarios más parecidos a nuestro "active" o "center" user (K-NN). Luego, hacemos predicción de items que los vecinos han consumido, pero que el "active user" no ha consumido aún.

$$Similaridad(u,v) = w(u,v), v \in K$$

$$\hat{p}_{u,i} = \bar{r}_u + \alpha \sum_{v \in N(u)} w(u,v)(r_{v,i} - \bar{r}_v)$$

Pros y Contras del Filtrado Colaborativo User-Based (KNN)

- PROS:
  - Fácil de implementar
  - Independiente del contexto
  - Comparado con otras técnicas, como basado en contenidos, más precisa
- CONS:
  - Sparsity
  - Cold-start
  - New Item

¿Por qué otra versión de Filtrado Colaborativo?

Balance entre Escalabilidad y Exactitud

- **Exactitud**: Mientras más vecinos K consideramos (bajo cierto umbral), mejor debería ser mi clasificación (Lathia et al. 2008)
- **Escalabilidad**: Pero mientras más usuarios n existen en el sistema, mayor es el costo de encontrar los K vecinos más cercanos, ya que K-NN es $O(dnk)$. Considerando un sitio con millones de usuarios, calcular las recomendaciones usando este método $memory - based$ se hace poco sustentable.

Más aún, hay que lidiar con otros problemas

- **Dispersión (Sparsity)**: La baja densidad de los datos hace que el Filtrado Colaborativo basado en el usuario sufra de "Cold-start" (usuarios con pocos ratings o historial de acciones) y también del "new item problem" (items nuevos que nadie los ha consumido)

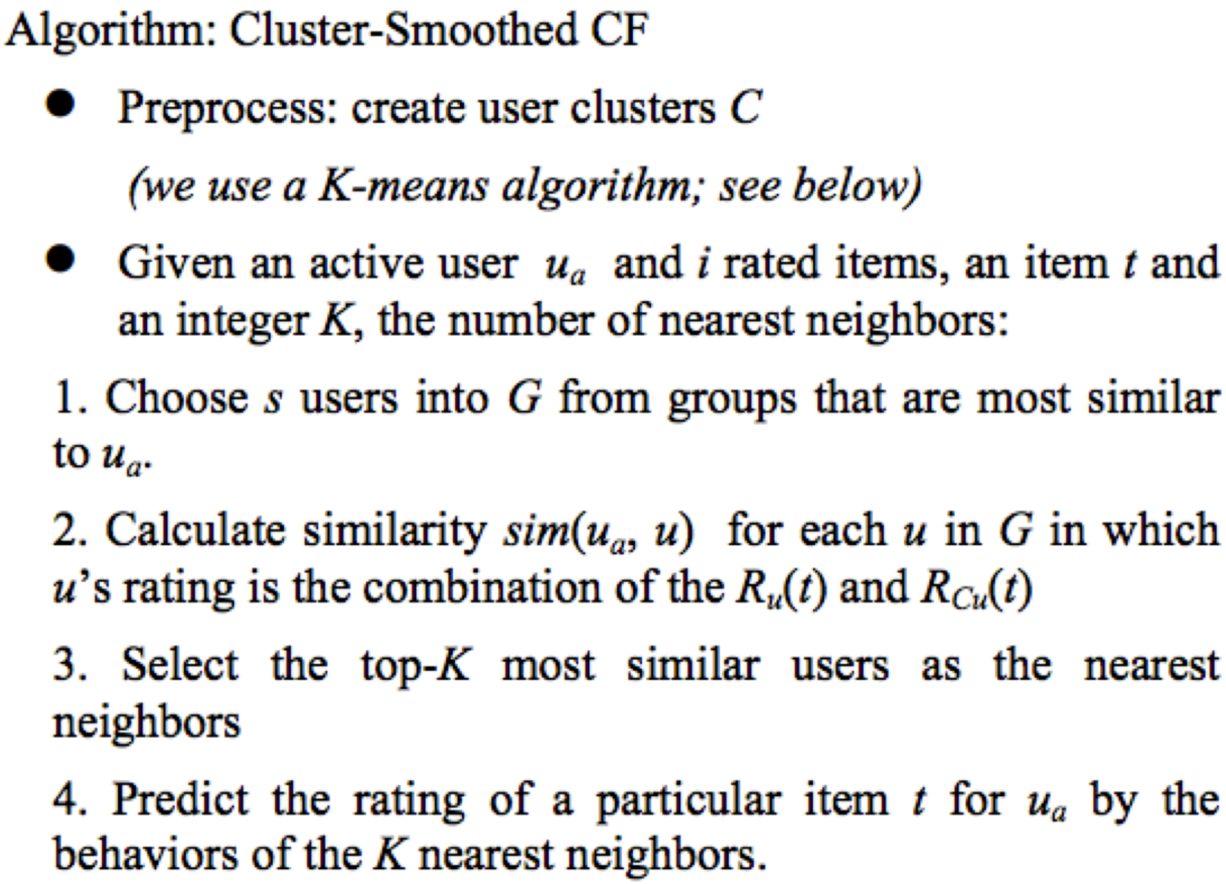
Opciones

- **Model-based methods**: Redes Bayesianas (ideales en casos en que las preferencias del usuario no cambian tan a menudo), Reducción de dimensionalidad (estado del arte, pero tiene algunos costos de implementación, especialmente en "tunear" los parámetros)
- **Clustering**, aunque tienen como efecto producir recomendaciones "no tan personalizadas" y, disminuir la exactitud de las predicciones en algunos casos (Breese et al. 1998)
- **Graph-based methods**: Horting, Random Walks, Spread of activation. Son menos precisos, pero contribuyen a dar mayor diversidad a las recomendaciones
- **Item-base recommendation**: Revisar user-based (precisión + simpleza) y escalarlo :-)

Alternativa UB-CF con Clustering

- Ungar, L. H., & Foster, D. P. (1998). Clustering methods for collaborative filtering. In AAAI workshop on recommendation systems. ~ EM.
- O'Connor, M., & Herlocker, J. (1999). Clustering items for collaborative filtering. In Proceedings of the ACM SIGIR workshop on recommender systems. ~ Hierarchical.
- Xue, G. R., Lin, C., Yang, Q., Xi, W., Zeng, H. J., Yu, Y., & Chen, Z. (2005). Scalable collaborative filtering using cluster-based smoothing. SIGIR ~ K-means.

Alternativa UB-CF con Clustering



Algorithm: Cluster-Smoothed CF

- Preprocess: create user clusters C

  (we use a K-means algorithm; see below)

- Given an active user u_a and i rated items, an item t and an integer K, the number of nearest neighbors:

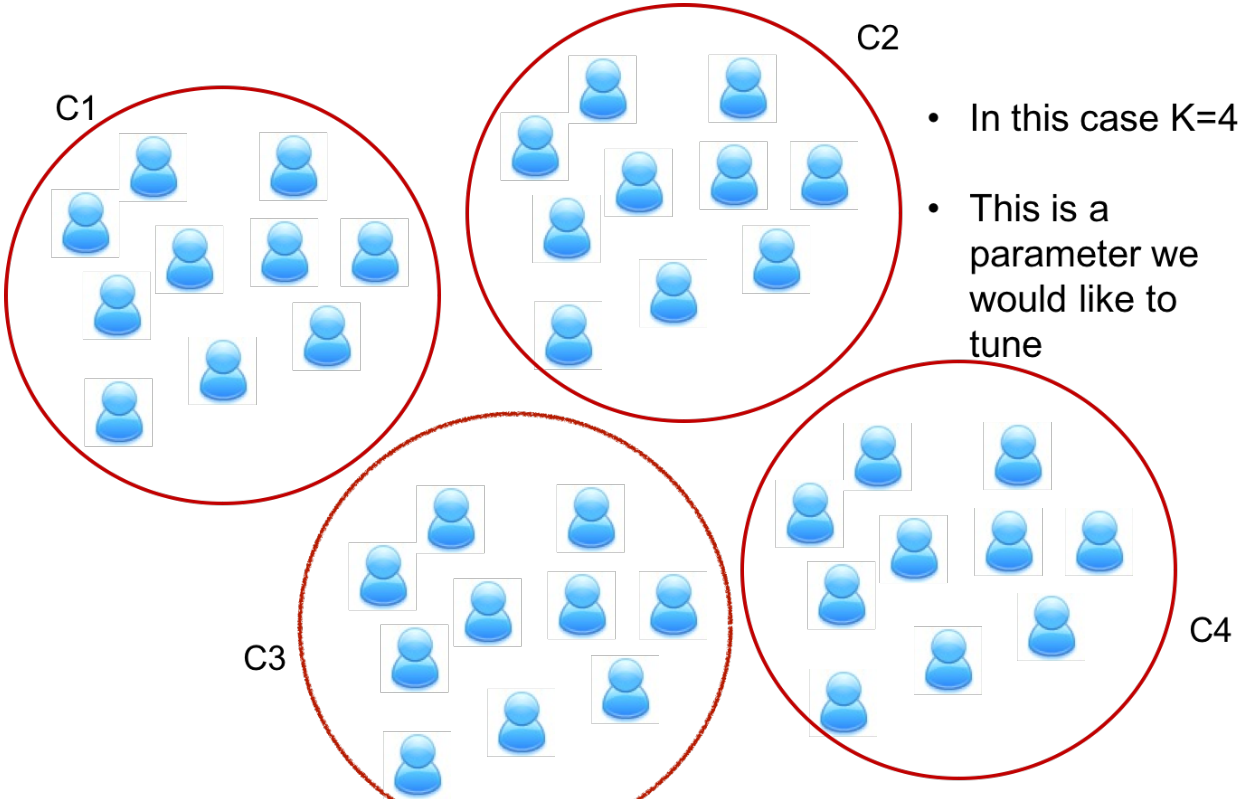
1. Choose s users into G from groups that are most similar to u_a.

2. Calculate similarity $sim(u_a, u)$ for each u in G in which u's rating is the combination of the $R_u(t)$ and $R_{Cu}(t)$

3. Select the top-K most similar users as the nearest neighbors

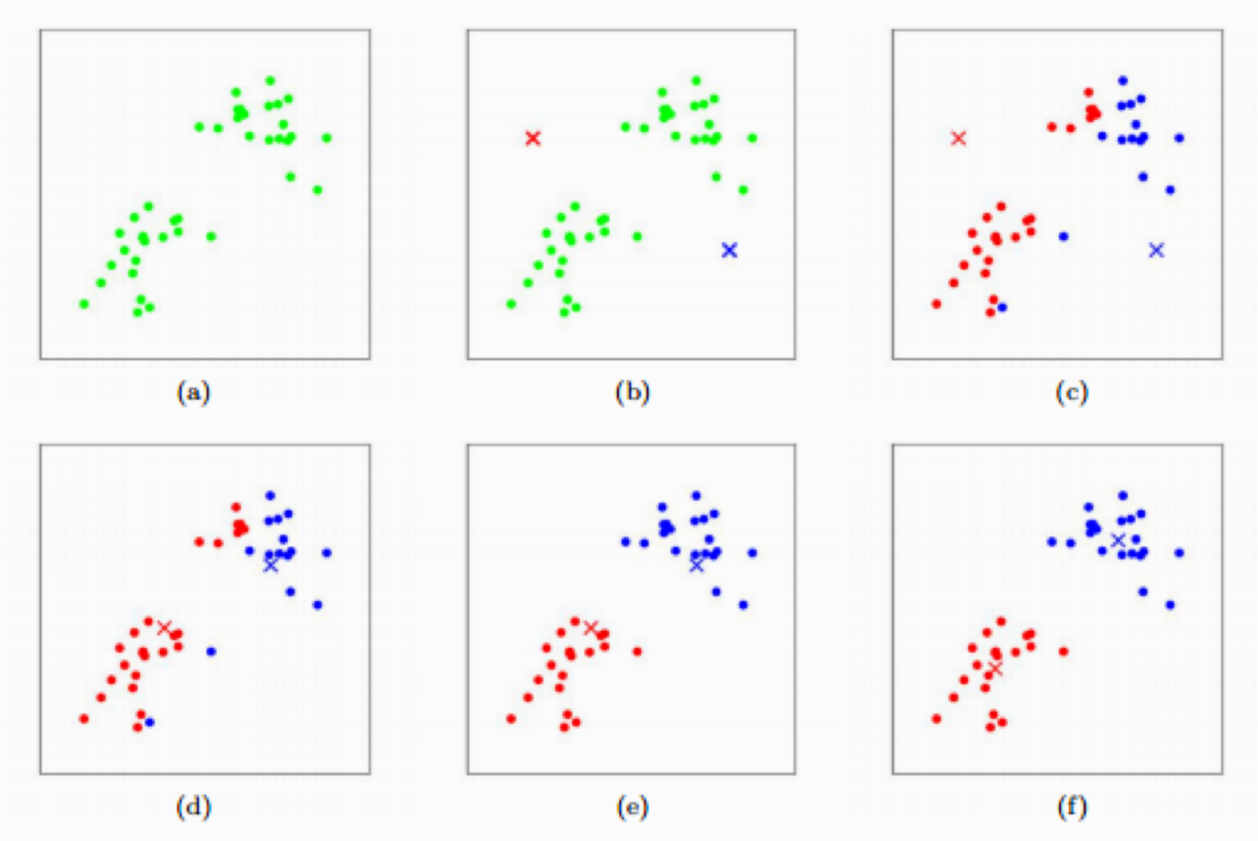
4. Predict the rating of a particular item t for u_a by the behaviors of the K nearest neighbors.

Paso 1: K-means



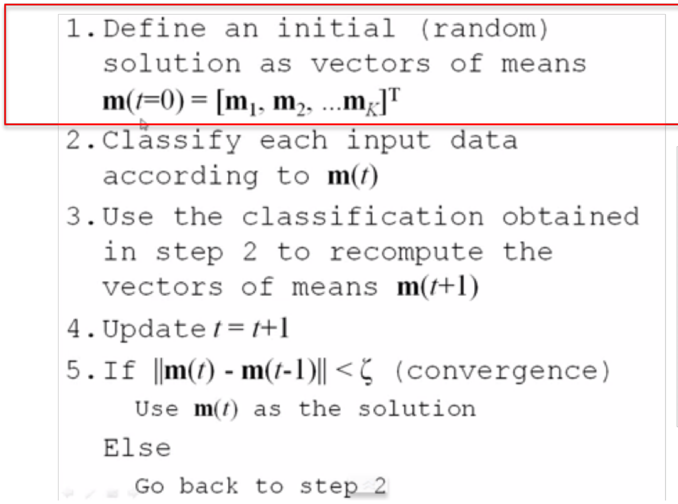
- In this case K=4
- This is a parameter we would like to tune

K-means I

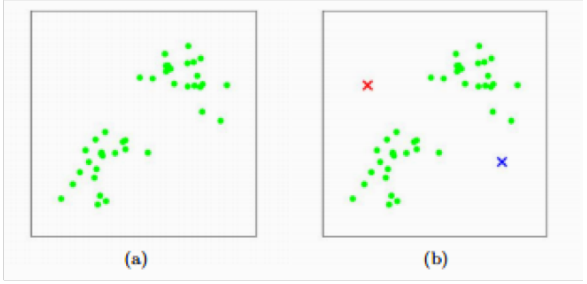


13/28

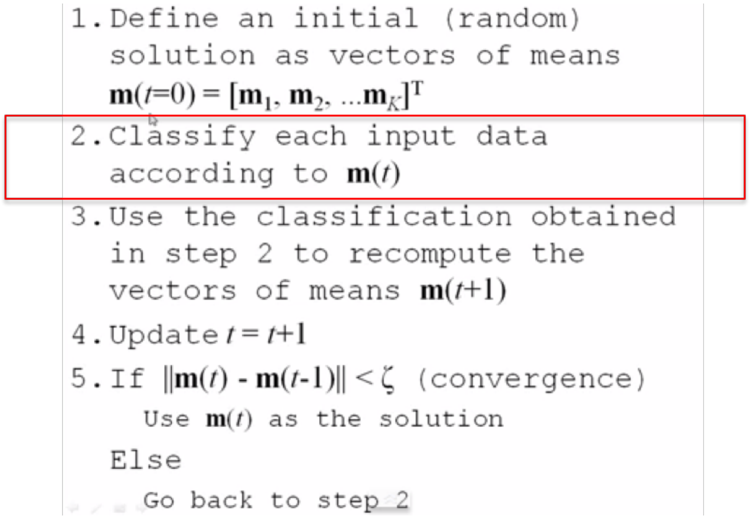
K-means II



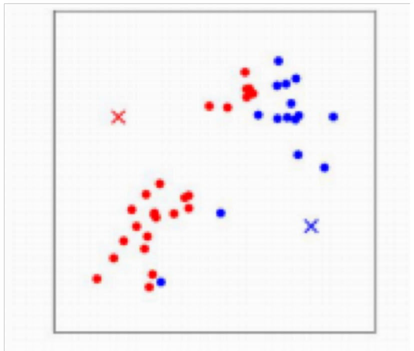
```
1. Define an initial (random)
   solution as vectors of means
   $\mathbf{m}(t=0) = [\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_K]^T$
2. Classify each input data
   according to $\mathbf{m}(t)$
3. Use the classification obtained
   in step 2 to recompute the
   vectors of means $\mathbf{m}(t+1)$
4. Update $t = t+1$
5. If $\|\mathbf{m}(t) - \mathbf{m}(t-1)\| < \zeta$ (convergence)
      Use $\mathbf{m}(t)$ as the solution
   Else
      Go back to step 2
```



K-means III

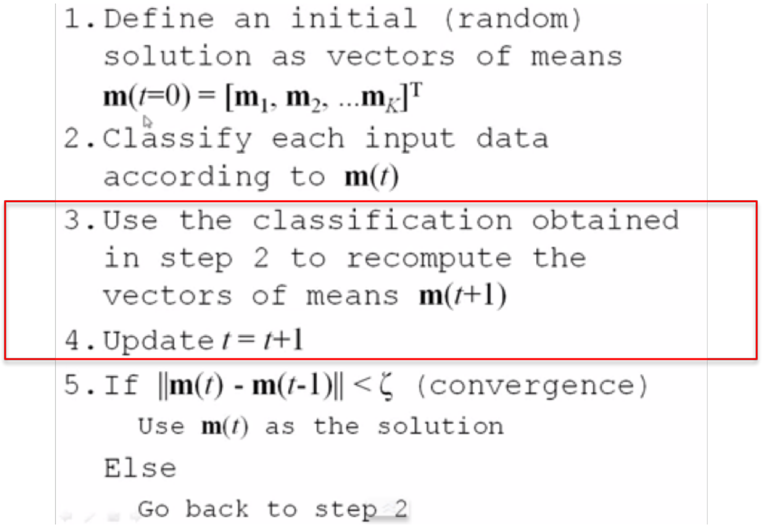


```
1. Define an initial (random)
   solution as vectors of means
   $\mathbf{m}(t=0) = [\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_K]^T$
2. Classify each input data
   according to $\mathbf{m}(t)$
3. Use the classification obtained
   in step 2 to recompute the
   vectors of means $\mathbf{m}(t+1)$
4. Update $t = t+1$
5. If $\|\mathbf{m}(t) - \mathbf{m}(t-1)\| < \zeta$ (convergence)
      Use $\mathbf{m}(t)$ as the solution
   Else
      Go back to step 2
```

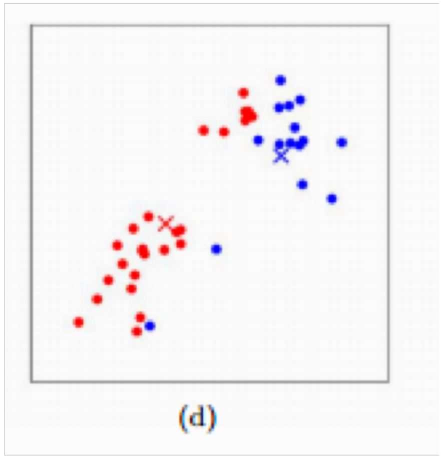


(c)

K-means IV

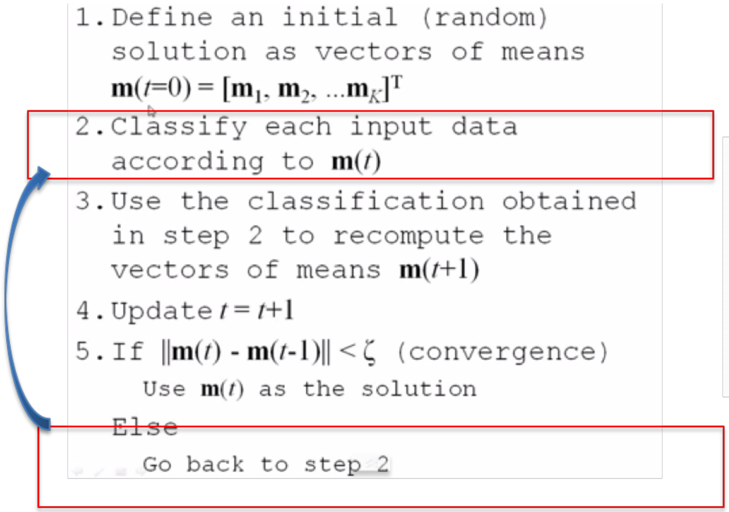


```
1. Define an initial (random)
   solution as vectors of means
   $\mathbf{m}(t=0) = [\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_K]^T$
2. Classify each input data
   according to $\mathbf{m}(t)$
3. Use the classification obtained
   in step 2 to recompute the
   vectors of means $\mathbf{m}(t+1)$
4. Update $t = t+1$
5. If $\|\mathbf{m}(t) - \mathbf{m}(t-1)\| < \zeta$ (convergence)
      Use $\mathbf{m}(t)$ as the solution
   Else
      Go back to step 2
```

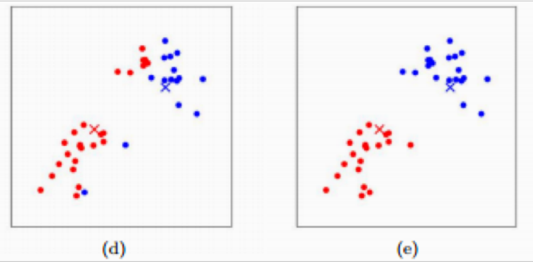


(d)

K-means V

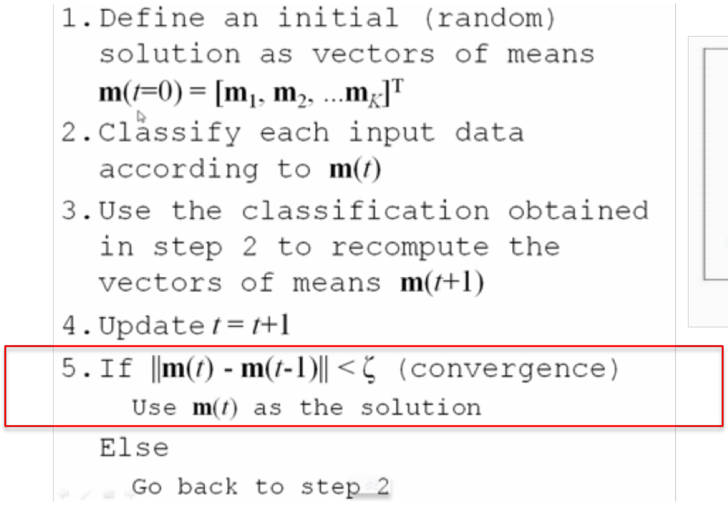


```
1. Define an initial (random)
   solution as vectors of means
   $\mathbf{m}(t=0) = [\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_K]^T$
2. Classify each input data
   according to $\mathbf{m}(t)$
3. Use the classification obtained
   in step 2 to recompute the
   vectors of means $\mathbf{m}(t+1)$
4. Update $t = t+1$
5. If $\|\mathbf{m}(t) - \mathbf{m}(t-1)\| < \zeta$ (convergence)
     Use $\mathbf{m}(t)$ as the solution
   Else
     Go back to step 2
```

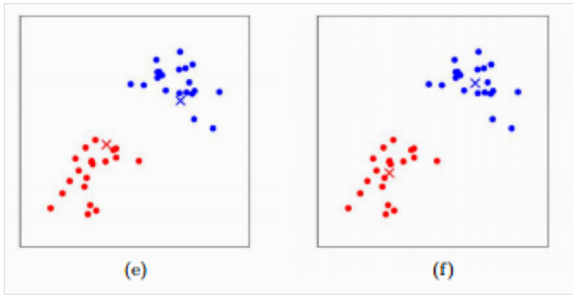


(d) (e)

K-means VI



```
1. Define an initial (random)
   solution as vectors of means
   $\mathbf{m}(t=0) = [\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_K]^T$
2. Classify each input data
   according to $\mathbf{m}(t)$
3. Use the classification obtained
   in step 2 to recompute the
   vectors of means $\mathbf{m}(t+1)$
4. Update $t = t+1$
5. If $\|\mathbf{m}(t) - \mathbf{m}(t-1)\| < \zeta$ (convergence)
     Use $\mathbf{m}(t)$ as the solution
   Else
     Go back to step 2
```



(e) (f)

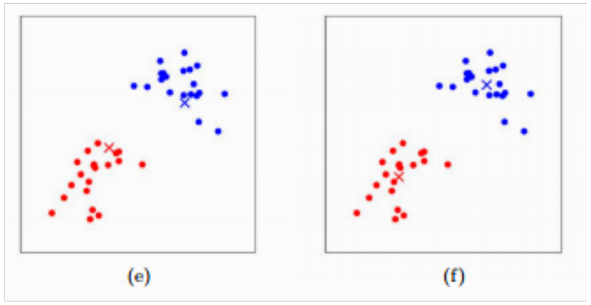
K-means VII

- Given the users' set

$$U = \{u_1, u_{2,} u_3, \dots u_n\}$$

- Where each user is represented by the items consumed

$$u_i = \{t_1, t_{2,} t_3, \dots t_m\}$$


(e) (f)

- We will cluster them into K groups using K-means

$$C = \{C_u^1, C_u^2, \dots, C_u^k\}$$

- Using the similarity function

$$sim_{u,u'} = \frac{\sum_{t \in T(u) \wedge T(u')} (R_u(t) - \overline{R_u}) \cdot (R_{u'}(t) - \overline{R_{u'}})}{\sqrt{\sum_{t \in T(u) \wedge T(u')} (R_u(t) - \overline{R_u})^2} \sqrt{\sum_{t \in T(u) \wedge T(u')} (R_{u'}(t) - \overline{R_{u'}})^2}}$$

Data Smoothing

- Muchos Usuarios tienen pocos ratings
- Si sabemos que el usuario pertenece a cierto cluster, podemos llenar tupla $R_u(t)$

$$R_u(t) = \begin{cases} R_u(t) & \text{if user } u \text{ rate the item } t \\ \hat{R}_u(t) & \text{else} \end{cases}$$

- Donde

$$\hat{R}_u(t) = \overline{R_u} + \Delta R_{C_u}(t)$$

$$\Delta R_{C_u}(t) = \sum_{u' \in C_u(t)} (R_{u'}(t) - \overline{R_{u'}}) / \, | \, C_u(t) \, |$$

y C_u: cluster al que pertenece usuario u.

Pre-Selección de Vecindario

- Podemos comparar vecinos considerando sólo los clusters más promisorios:

$$sim_{u_a,C} = \frac{\sum_{t\in T(u_a)\wedge T(C)} \Delta R_C(t)\cdot(R_{u_a}(t)-\overline{R_{u_a}})}{\sqrt{\sum_{t\in T(u_a)\wedge T(C)}(\Delta R_C(t))^2}\sqrt{\sum_{t\in T(u_a)\wedge T(C)}(R_{u_a}(t)-\overline{R_{u_a}})^2}}$$

- Considerando

$$\Delta R_{C_u}(t) = \sum_{u'\in C_u(t)} (R_{u'}(t)-\overline{R_{u'}})/\,|\,C_u(t)\,|$$

Selección de Vecinos

- Después de la pre-selección, recalculamos similaridad considerando rating original y rating del grupo, usando un factor de balance w_{ut}

$$w_{ut} = \begin{cases} 1-\lambda & \text{if user } u \text{ rate the item } t \\ \lambda & \text{else} \end{cases}$$

- Luego calculamos los K usuario más cercanos

$$sim_{u_a,u} = \frac{\sum_{t\in T(u_a)} w_{ut}\cdot(R_u(t)-\overline{R_u})\cdot(R_{u_a}(t)-\overline{R_{u_a}})}{\sqrt{\sum_{t\in T(u_a)} w_{ut}^{2}\cdot(R_u(t)-\overline{R_u})^{2}}\sqrt{\sum_{t\in T(u_a)}(R_{u_a}(t)-\overline{R_{u_a}})^{2}}}$$

Finalmente, predicción

$$R_{u_a}(t) = \overline{R_{u_a}} + \frac{\sum_{i=1}^{K} w_{ut} \cdot sim_{u_a,u} \cdot (R_u(t) - \overline{R_u})}{\sum_{i=1}^{K} w_{ut} \cdot sim_{u_a,u}}$$

Resultados I

ALGORITMOS

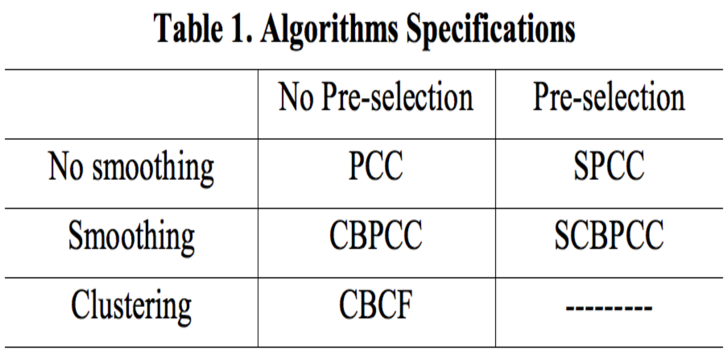


Table 1. Algorithms Specifications

	No Pre-selection	Pre-selection
No smoothing	PCC	SPCC
Smoothing	CBPCC	SCBPCC
Clustering	CBCF	---------

DATASET

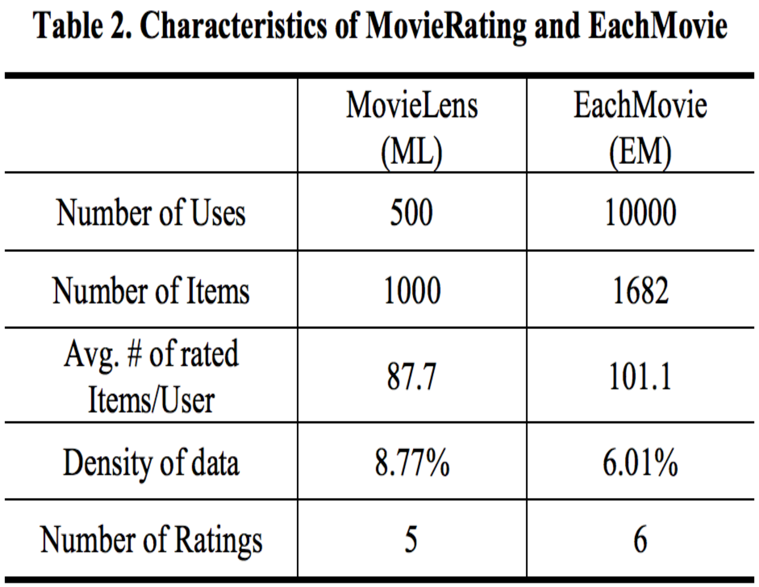


Table 2. Characteristics of MovieRating and EachMovie

	MovieLens (ML)	EachMovie (EM)
Number of Uses	500	10000
Number of Items	1000	1682
Avg. # of rated Items/User	87.7	101.1
Density of data	8.77%	6.01%
Number of Ratings	5	6

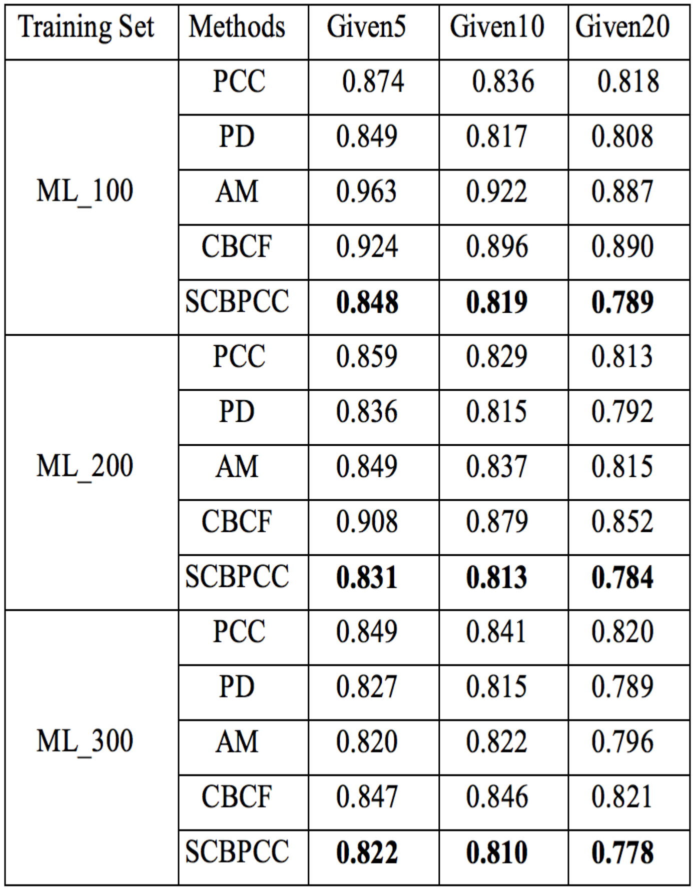
$$MAE = \frac{\sum_{u \in T} | R_u(t_j) - \tilde{R_u(t_j)} |}{|T|}$$

Evaluación:

Resultados II

MAE

Table 3. MAE on MovieLens for different algorithms. (A small value means a better performance)



Training Set	Methods	Given5	Given10	Given20
ML_100	PCC	0.874	0.836	0.818
	PD	0.849	0.817	0.808
	AM	0.963	0.922	0.887
	CBCF	0.924	0.896	0.890
	SCBPCC	**0.848**	**0.819**	**0.789**
ML_200	PCC	0.859	0.829	0.813
	PD	0.836	0.815	0.792
	AM	0.849	0.837	0.815
	CBCF	0.908	0.879	0.852
	SCBPCC	**0.831**	**0.813**	**0.784**
ML_300	PCC	0.849	0.841	0.820
	PD	0.827	0.815	0.789
	AM	0.820	0.822	0.796
	CBCF	0.847	0.846	0.821
	SCBPCC	**0.822**	**0.810**	**0.778**

PARÁMETROS

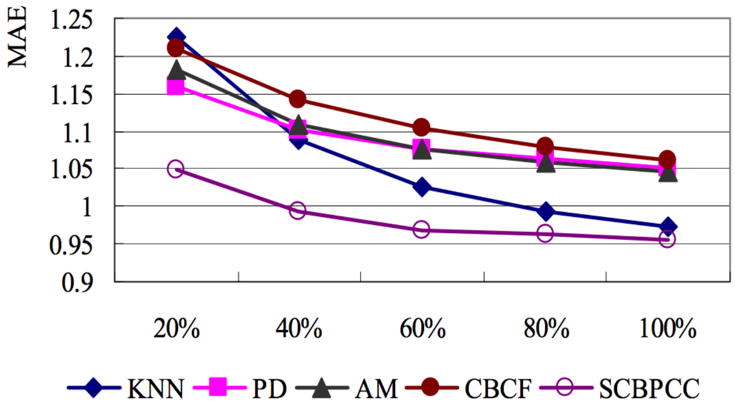
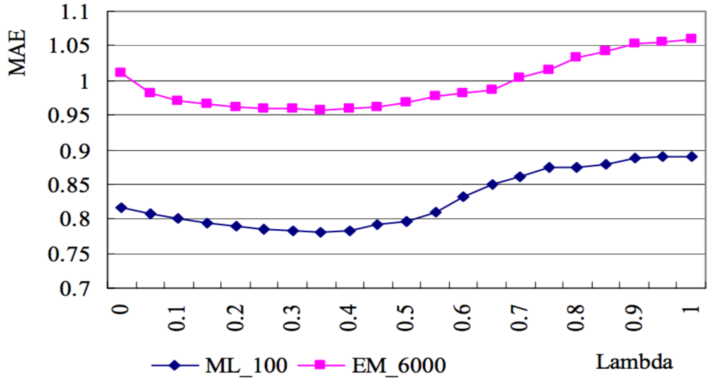


Figure 2. MAE on different density of EachMovie data (A small value means a better performance)



Proxima Clase

- Item-based Collaborative Filtering

Referencias

- 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.
- Lathia, N., Hailes, S., & Capra, L. (2008, March). The effect of correlation coefficients on communities of recommenders. In Proceedings of the 2008 ACM symposium on Applied computing (pp. 2000-2005). ACM.
- Breese, J. S., Heckerman, D., & Kadie, C. (1998, July). Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence (pp. 43-52). Morgan Kaufmann Publishers Inc.
- Das, A. S., Datar, M., Garg, A., & Rajaram, S. (2007, May). Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web (pp. 271-280). ACM.
- James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. 2010. The YouTube video recommendation system. In Proceedings of the fourth ACM conference on Recommender systems (RecSys '10).

Referencias Adicionales

- Neal Lathia, Stephen Hailes, and Licia Capra. 2008. The effect of correlation coefficients on communities of recommenders. In Proceedings of the 2008 ACM symposium on Applied computing (SAC '08). ACM, New York, NY, USA, 2000-2005
- Gui-Rong Xue, Chenxi Lin, Qiang Yang, WenSi Xi, Hua-Jun Zeng, Yong Yu, and Zheng Chen. 2005. Scalable collaborative filtering using cluster-based smoothing. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '05).
- O'Connor, M., & Herlocker, J. (1999, August). Clustering items for collaborative filtering. In Proceedings of the ACM SIGIR workshop on recommender systems (Vol. 128). UC Berkeley.
- Ungar, L. H., & Foster, D. P. (1998, July). Clustering methods for collaborative filtering. In AAAI workshop on recommendation systems (Vol. 1, pp. 114-129).
- Xavier Amatriain, Josep M. Pujol, and Nuria Oliver . 2009. I Like It... I Like It Not: Evaluating User Ratings Noise in Recommender Systems. In Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization: formerly UM and AH (UMAP '09),
- Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53.