商品推薦系統

-協同過濾推薦系統

Lecture 2

講師 :周光宇博士

2018.08

協同過濾推薦

最突出的推薦方法

- 大型商業電子商務網站經常使用
- 很好理解，存在多種算法和變化
- 適用於許多領域（書籍，電影，DVD，..）

方法

- 使用“人群的智慧”來推薦物品

基本假設和想法

- 用戶給商品的評分（隱式或顯式）
- 過去曾經有類似喜好的顧客將來也會有類似的喜好

協同過濾推薦

輸入

- 用戶商品評分的矩陣

用戶商品評分

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1		3	3
User2	4		4	3	
User3	3	3	1		4
User4		5	5	2	1

解決方案

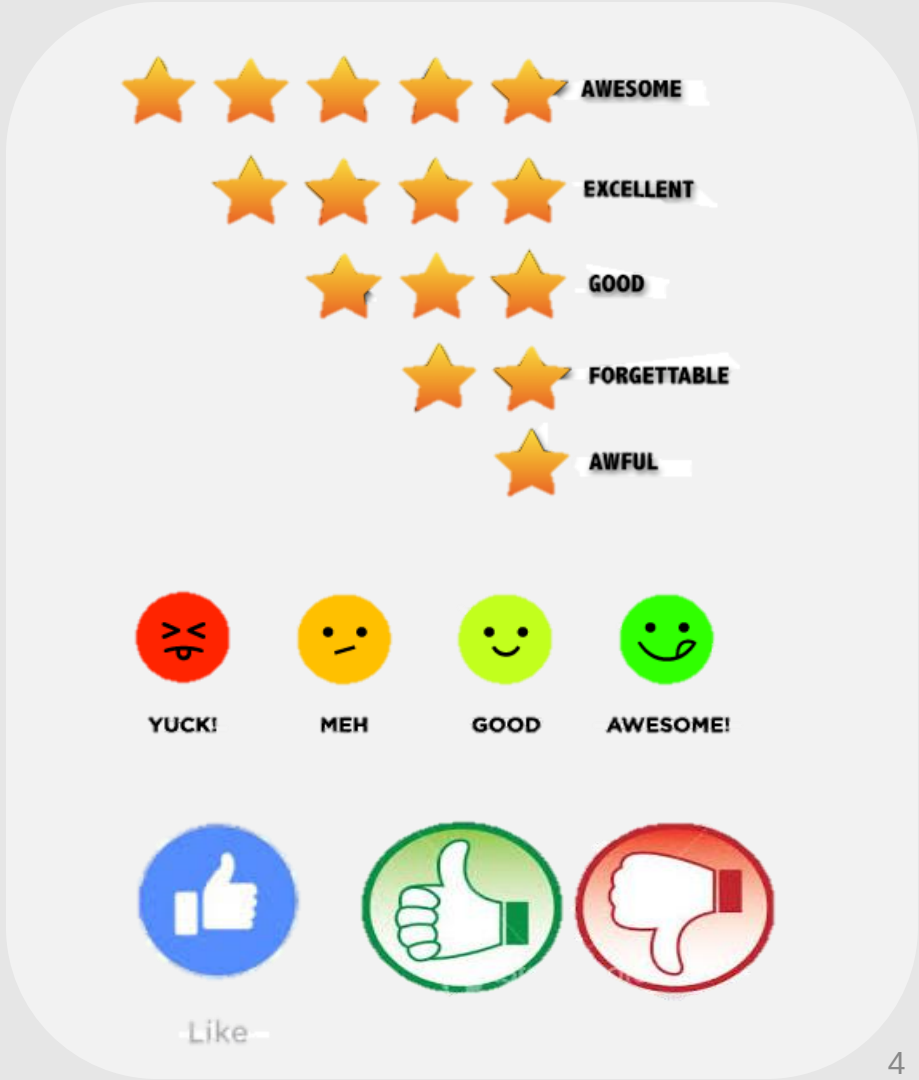
- 基於鄰里的推薦系統
- 基於模型的推薦系統

輸出類型

- 評分預測，計算當前用戶喜歡或不喜歡某個商品的程度
- 排序預測，推薦前N個(Top-N)用戶可能喜歡的商品

顯式評分類型

- Interval-Based
- Ordinal Ratings
- Binary Ratings
- Unary Ratings



商品的評分往往滿足被稱為長尾的特性。
只有一小部分的商品經常被評分。

順序評分矩陣

		Airplane	Matrix	50 First Dates	...	Lion King
		comedy	action	romance	...	cartoon
Joe	*27,M*	5	4	1		4
Karen	*53,F*	2		4		
...						
Ted	*25,M*	5	3			2
U_a	*48,M*	5	2	?	?	?

隱式評分(Implicit Ratings)

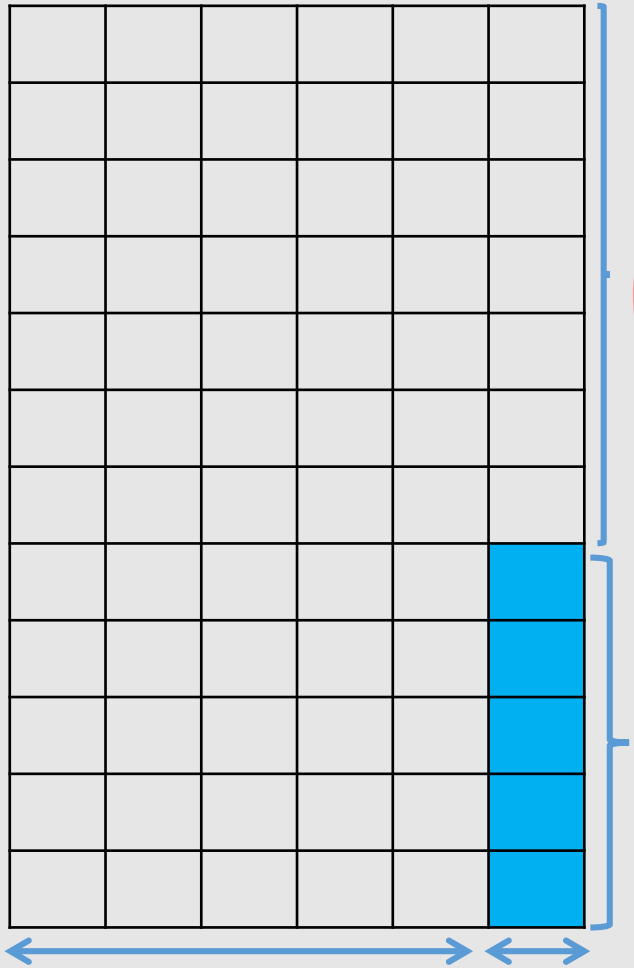
- 通常由線上商店或應用從顧客購買的行為蒐集
- 例如，當顧客購買物品時，許多推薦系統將該行為解釋為正面評價
- 隱式評分可以不斷的蒐集，不需要用戶端的額外參與
- 主要問題:
  - 不能確定用戶行為是否被正確地解釋
  - 例如，用戶可能不喜歡他或她已經購買的所有書籍；用戶也可能已經為別人買了一本書
- 有時會將「信賴水準(confidence level)」與「隱式評分」結合使用。當用戶購買特定商品的次數越多，我們可以更有信心用戶對於該商品的喜歡度越高。

一元評估矩陣(Unary Rating)

- 一元矩陣建立於用戶行為，亦稱作**隱式反饋矩陣 (Implicit Feedback Matrix)**

		Airplane	Matrix	50 First Dates	…	Lion King
		comedy	action	romance	…	cartoon
Joe	*27,M*	1	1	1		1
Karen	*53,F*	1		1		
…						
Ted	*25,M*	1	1			1
U_a	*48,M*	1	1	?	?	?

分類與協同過濾(Matrix Completion Problem)

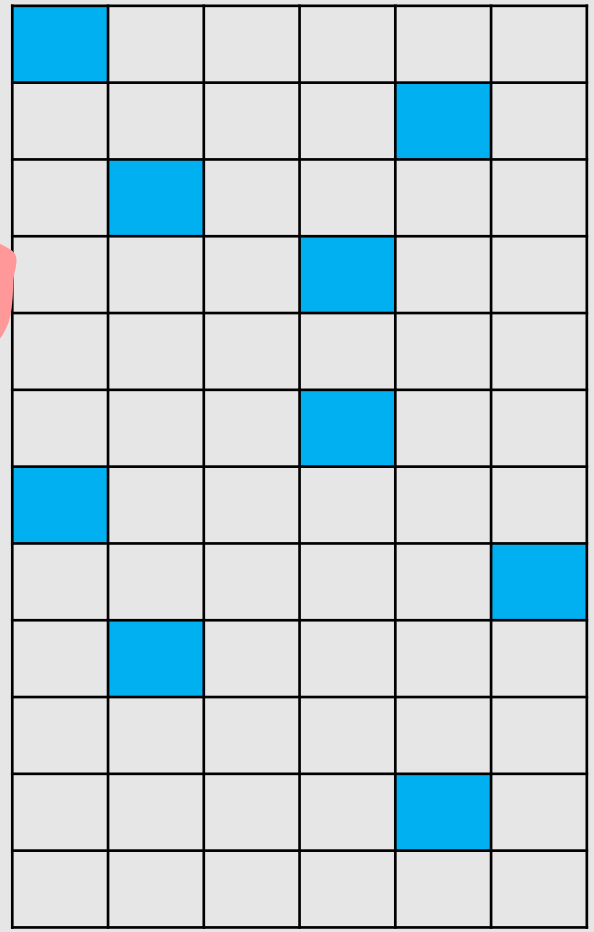


Training Rows

Test Rows

Independent Variables

Dependent Variables



Training and Test Rows 沒有分界

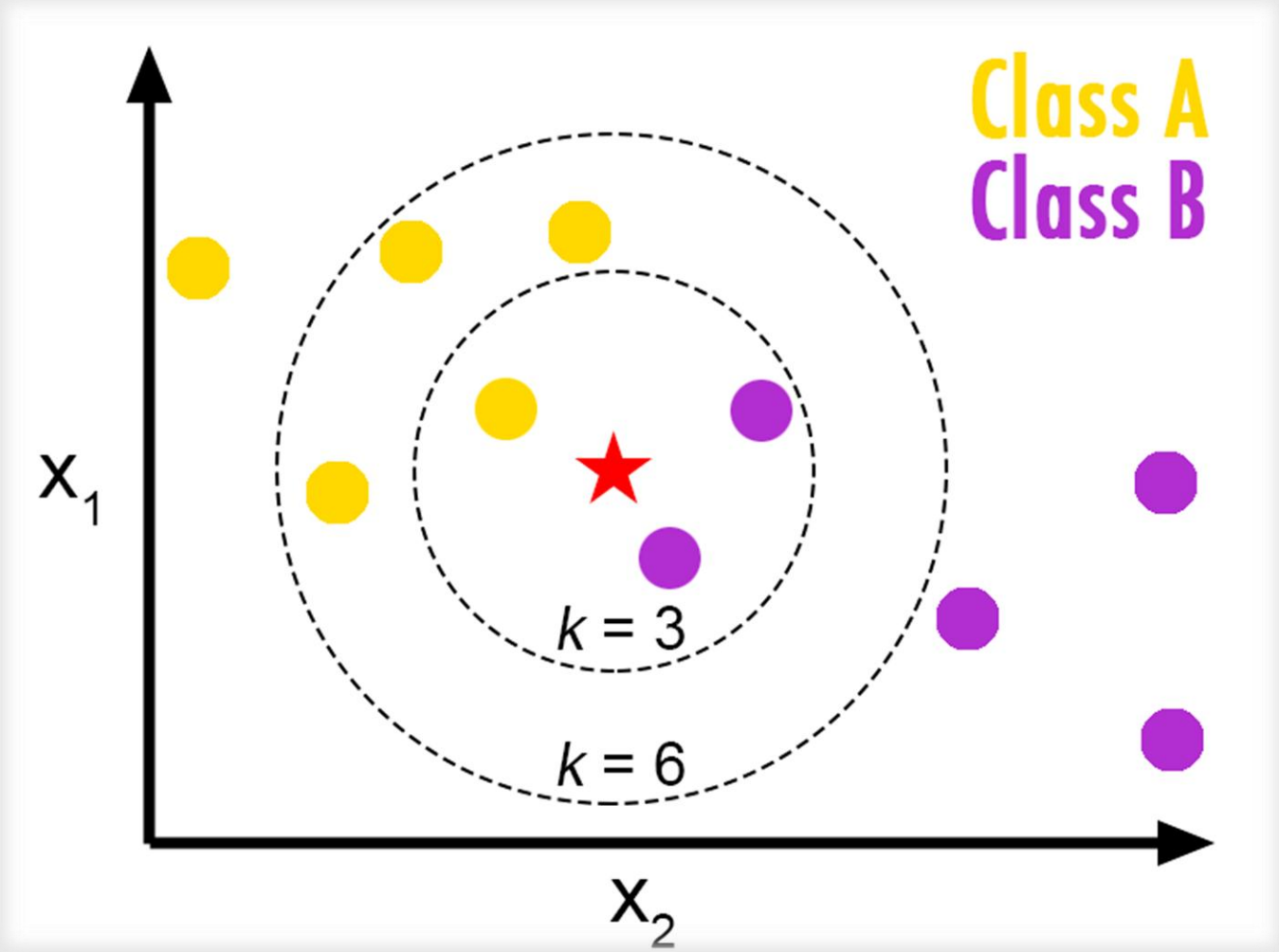
Independent and dependents variables沒有分界

基於鄰里vs基於模型

- 基於鄰里的方法可以看作是k−最近鄰 (k-nearest neighbour) 分類器的推廣。 這些方法是基於實例的方法。
- 在基於模型的方法中，使用數據建立機器學習模型，並將訓練與預測階段分開。
- 幾乎所有傳統的機器學習方法，如決策樹、貝葉斯(Bayesian)分類器、回歸、支持向量機和神經網絡都可以推廣到協同過濾模型。 這是因為傳統的分類和回歸問題是矩陣完成問題(Matrix Completion Problem P.8)的特例。

基於鄰里的協同過濾

K-最近鄰算法



Source: Medium.com

基於鄰里的協同過濾

- 協同過濾算法的概念與K-最近鄰算法相似
- 也被稱為基於內存(memory)的算法
- 算法基於以下事實:

  相似用戶顯示相似的評分行為模式,
  類似的商品獲得類似的評分。

- 有兩種主要類型:

  基於用戶鄰里的協同過濾
  基於商品鄰里的協同過濾

- 算法的結果可以用兩種方法之一來表示:

  預測用戶對商品的評分
  推薦top-k商品或top-k用戶

基於用戶鄰里的模型

- 該方法是找出與目標用戶相似的用戶。然後，使用他們的評分來預測目標用戶的評分。
- 為了找目標用戶的鄰里用戶，我們必須計算目標用戶與所有其他用戶的相似度。因此，需要一個計算相似度的函數。
- 因為不同的用戶可能具有不同的評分模式，相似度計算比較不直接。一種常用的方法是Pearson相關係數。
- 目標用戶的 Top-K 鄰居是透過Pearson係數來識別的。 由於目標用戶的鄰居不太可能購買每件商品，因此，每件商品已經給了評分的鄰居數量不同。所以針對每個要預測評分的商品,分別選擇最相似的k個對該商品已經有評分的用戶。
- 目標用戶的預測評分由這些對該商品已經評分的用戶的評分來計算加權平均值。

計算用戶相似度 – Pearson相關係數

- 該方法計算兩個用戶(u和v)對相同商品評分之間的統計相關性（Pearson's）以決定其相似度。

$$\text{Sim}(u, v) = \frac{\sum_{i \in I_u \cap I_v}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I_u \cap I_v}(r_{v,i} - \bar{r}_v)^2}}$$

Where $I_u$denote the set of item indices that user u has specified ratings and $\bar{r}_u$ **is the average rating for user u for items that both users rated**

- 實證分析顯示，對於基於用戶的推薦系統 - 至少對於最好推薦領域的研究 - Pearson相關係數優於其他計算用戶相似度的方法

要選擇多少個鄰居?

- 為每個商品的評分預測選擇鄰居是基於預定的「鄰居數目」或「相似度門檻」 。
- 相似度較低的用戶會引入過多的雜訊(noise)； 因此， 應該排除在外。
- 預定的「相似度門檻」與應用領域相關， 因此需要分析相關數據再來決定。
- 一般來說， 「鄰居數目」在20-50範圍內的值是許多應用領域的合理起始數目。

用戶-用戶相似度計算

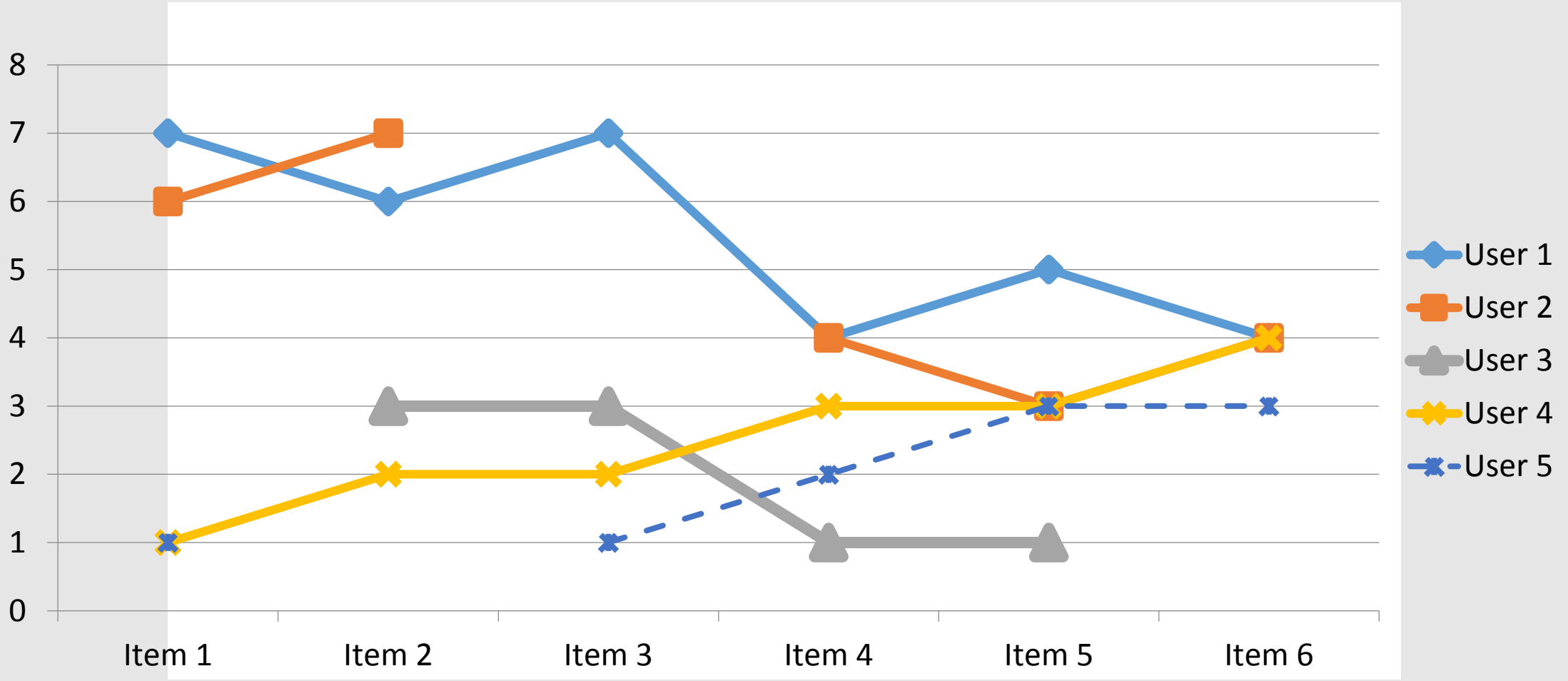
	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Mean Rating	Pearson (I, 3)	Cosine (I, 3)
User 1	7	6	7	4	5	4	5.5	0.894	0.956
User 2	6	7	?	4	3	4	4.8	0.939	0.981
User 3	?	3	3	1	1	?	2	1.0	1.0
User 4	1	2	2	3	3	4	2.5	-1.0	0.789
User 5	1	?	1	2	3	3	2	-0.817	0.645

$$\text{Pearson(u, v)} = \frac{\sum_{i \in I_u \cap I_v}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I_u \cap I_v}(r_{v,i} - \bar{r}_v)^2}}$$

$$\text{Cosine (u, v)} = \frac{\sum_{i \in I_u \cap I_v}(r_{u,i})(r_{v,i})}{\sqrt{\sum_{i \in I_u \cap I_v}(r_{u,i})^2}\sqrt{\sum_{i \in I_u \cap I_v}(r_{v,i})^2}}$$

Where I_u denote the set of item indices that user u has specified ratings

用戶-用戶相似度


8
7
6
5
4
3
2
1
0
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
User 1
User 2
User 3
User 4
User 5

基於用戶鄰里的模型 – 預測

- 透過Pearson係數找出Top-K個鄰居之後，通常把與目標用戶u具有非常低或負相關性的用戶過濾掉。
- 讓我們把結果的相似用戶索引集表示為 $P_u(j)$,j是我們想要預測用戶u評分的商品。
- **基於用戶的鄰里模型給用戶u對商品j的預測評分是:**

$$\hat{r}_{uj} = \bar{r}_u + \frac{\sum_{v \in P_u(j)} Sim(u,v) \cdot (r_{vj} - \bar{r}_v)}{\sum_{v \in P_u(j)} |Sim(u,v)|}$$

基於用戶鄰里的模型 – 預測例子

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Mean Rating	Pearson (I, 3)
User 1	7	6	7	4	5	4	5.5	0.894
User 2	6	7	?	4	3	4	4.8	0.939
User 3	?	3	3	1	1	?	2	1.0
User 4	1	2	2	3	3	4	2.5	-1.0
User 5	1	?	1	2	3	3	2	-0.817

$$\hat{r}_{uj} = \bar{r}_u + \frac{\sum_{v \in P_u(j)} Sim(u,v) \cdot (r_{vj} - \bar{r}_v)}{\sum_{v \in P_u(j)} |Sim(u,v)|}$$

$$\hat{r}_{31} = 2 + \frac{0.894 \cdot (7 - 5.5) + 0.939 \cdot (6 - 4.8)}{(0.894 + 0.939)} = 3.35$$

$$\hat{r}_{36} = 2 + \frac{0.894 \cdot (4 - 5.5) + 0.939 \cdot (4 - 4.8)}{(0.894 + 0.939)} = 0.86$$

基於用戶鄰里協同過濾的一些問題

- 基於用戶鄰里的協同過濾儘管有效，但隨著用戶群的增長，其可擴展性問題也隨之而來。
- 每當用戶對商品有新的評分，他們與其他用戶的相似性將隨著他們新的評分向量而可能改變
- 用戶的相似鄰居不僅取決於他們的評分，而且還取決於其他用戶的評分，所以他們的相似鄰居可以由於系統中任何用戶提供的新評分而改變；因此，提前找到類似的用戶是複雜的。

基於商品鄰里的模型

- 該方法是找出用戶目標商品的相似商品以計算目標商品的預測評級。
- 需要定義兩個商品 a 和 b 評分向量之間的相似度函數 Sim（a，b）。計算相似度 Sim（a，b）的一種方法是是餘弦(Cosine)相似度。它比其他基於商品的相似度計算更準確。
- 餘弦相似度是根據每對商品的平均中心評分(用戶 mean-centered rating) s_{uj} 來計算。 計算商品 a和 b 之間的餘弦相似度，如下一張投影片所示。
- Top-K的鄰居商品是以通過目標用戶已經評分商品的平均中心餘弦相似度來識別的。
- 目標用戶的目標商品預測評級是用戶的這些相似商品原始評級的加權平均值。

計算商品相似度 – 餘弦(cosine)相似係數

- 在商品與商品相似過濾中產生更好的結果
- 評級被視為n維空間中的向量
- 調整後的餘弦相似度:
  - 將平均用戶評分考慮在內，調整原來的評分
  - U: 對商品 “a” 和 “b” 都進行評分的用戶組

$$\mathrm{Sim}(a,b) = \frac{\sum_{u\in U}(r_{u,a}-\overline{r_u})(r_{u,b}-\overline{r_u})}{\sqrt{\sum_{u\in U}(r_{u,a}-\overline{r_u})^2}\sqrt{\sum_{u\in U}(r_{u,b}-\overline{r_u})^2}}$$

這和Pearson的相似性是一樣的嗎？ 如果不是，為什麼？

基於商品鄰里的模型 – 預測

- 用戶對目標商品的評分可以用相似商品相似度的加權平均分數來預測
- 然後挑選具有最高預測分數的候選商品做來生成Top-K推薦
- 在為用戶 u 收集了與目標商品相似的一組商品 S 之後, $p_{u,i}$ 可以預測如下:

$$p_{u,i} = \frac{\sum_{j \in S} sim(i,j) r_{u,j}}{\sum_{j \in S} |sim(i,j)|} \text{ where sim(i, j) > 0}$$

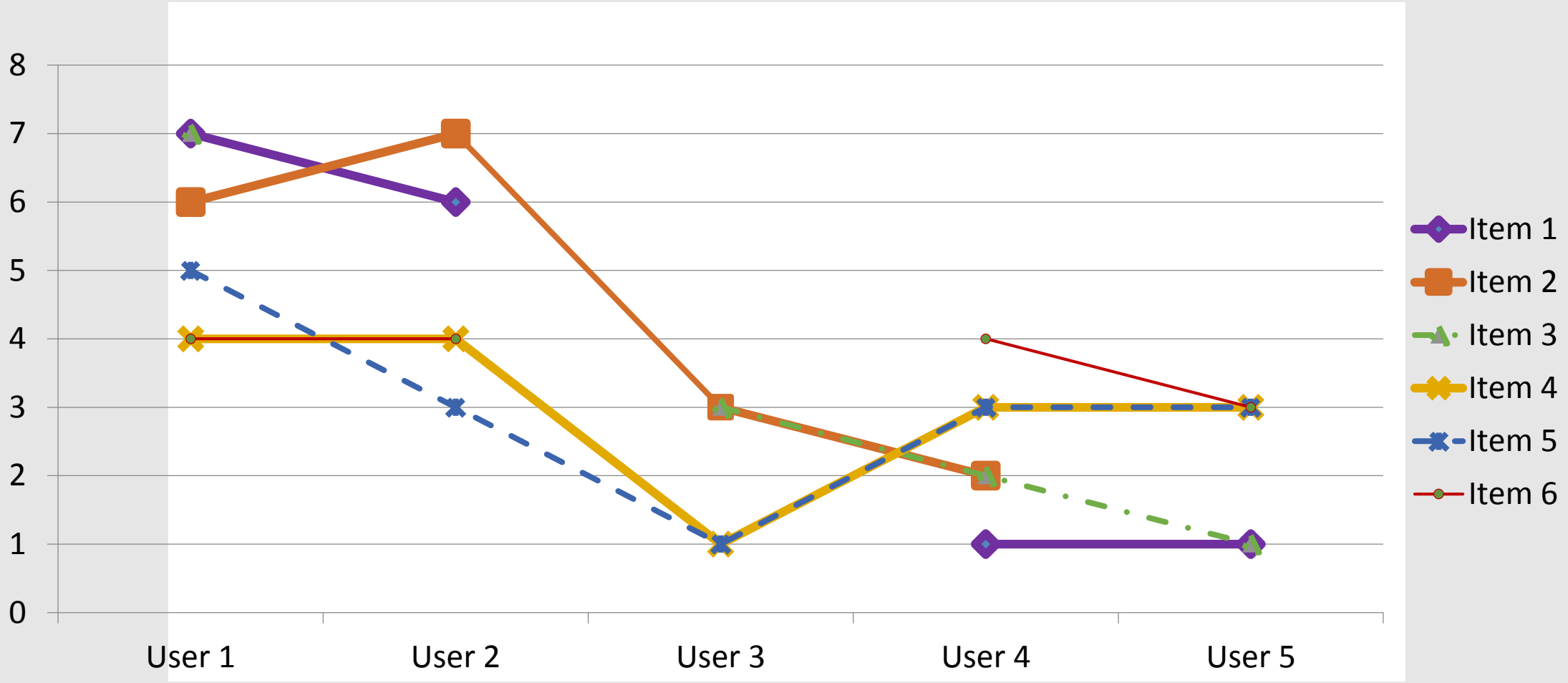
其中 $r_{u,j}$ 是原始評級 (不是以平均中心的評級)

基於商品鄰里的模型

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Mean Rating
User 1	7	6	7	4	5	4	5.5
User 2	6	7	?	4	3	4	4.8
User 3	?	3	3	1	1	?	2
User 4	1	2	2	3	3	4	2.5
User 5	1	?	1	2	3	3	2

我們用相同的用戶商品矩陣

商品-商品相似度


8
7
6
5
4
3
2
1
0
User 1
User 2
User 3
User 4
User 5
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6

基於平均中心餘弦相似度的例子

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	1.5	0.5	1.5	-1.5	-0.5	-1.5
User 2	1.2	2.2	?	-0.8	-1.8	-0.8
User 3	?	1	1	-1	-1	?
User 4	-1.5	-0.5	-0.5	0.5	0.5	1.5
User 5	-1	?	-1	0	1	1
Item-Item Cosine (1, j)	1	0.735	0.912	-0.848	-0.813	-0.990
Item-Item Cosine (6, j)	-0.990	-0.622	-0.912	0.829	0.730	1

$$\text{Adjusted Cosine (1, 3)} = \frac{1.5 \cdot 1.5 + (-1.5).(-0.5) + (-1).(-1)}{\sqrt{1.5^2 + (-1.5)^2 + (-1.5)^2} \cdot \sqrt{1.5^2 + (-0.5)^2 + (-1)^2}} = 0.912$$

商品2和商品3與商品1相似，商品4和商品5與商品6相似

基於商品鄰里的模型 - 預測例子

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	7	6	7	4	5	4
User 2	6	7	?	4	3	4
User 3	?	3	3	1	1	?
User 4	1	2	2	3	3	4
User 5	1	?	1	2	3	3
Item-Item Cosine (1, j)	1	0.735	0.912	-0.848	-0.813	-0.990
Item-Item Cosine (6, j)	-0.990	-0.622	-0.912	0.829	0.730	1

$$p_{u,i} = \frac{\sum_{j \in S} sim(i,j) r_{u,j}}{\sum_{j \in S} |s(i,j)|}$$

$$\hat{r}_{31} = \frac{0.735 \cdot 3 + 0.912 \cdot 3}{(0.735 + 0.912)} = 3$$

$$\hat{r}_{36} = \frac{0.829 \cdot 1 + 0.730 \cdot 1}{(0.829 + 0.730)} = 1$$

基於用戶和基於商品方法的比較

	基於商品	基於用戶
預測的準確性	通常更好。 因為使用自己的評級。	
多樣性，新穎性和驚喜性 Diversity, Novelty, and Serendipity		通常更好。 因為使用其他用戶的評分。
預測的解釋	通常可以提供具體的解釋。	
隨著評分變化的穩定性	更穩定。 因為通常用戶比商品多得多，因此，每個商品有更多的評分。	

基於鄰里的方法的優點和缺點

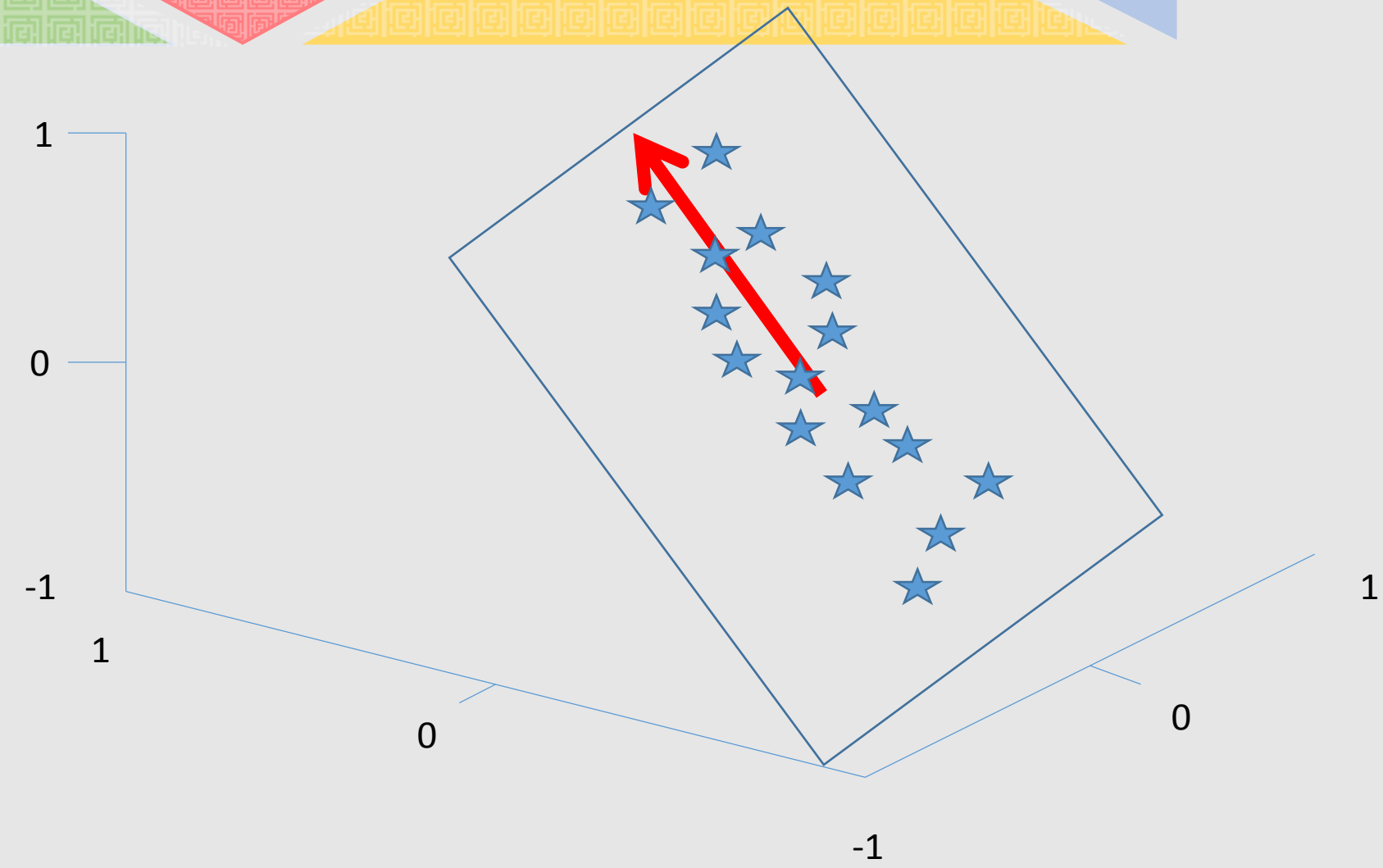
優點	缺點
簡單和直覺的方法	離線階段在大規模設置中有時可能不切實際
易於實行和調試	由於評分矩陣的稀疏性而覆蓋的範圍有限
易於解釋為什麼一個商品被推薦	冷啟動問題
可以解釋基於商品方法的推薦	
對增加新的商品和用戶表現相對穩定	

基於模型的協同過濾

概述

- 基於模型的推薦系統通常比基於鄰域的方法有許多優點:
  - **儲存空間效率:** 通常學習模型的大小比原始評分矩陣小得多。
  - **訓練速度和預測速度**: 訓練模型的構建要快得多，可以使用簡潔和概括的模型來高效地進行預測。
  - **避免過度配適 (overfitting):** 總結方法通常可以幫助避免過度配適; 也可以使用正則化(regularization) 方法來使這些模型更健全。
- 一般來說，一些最準確的方法都是使用基於模型的技術，尤其是潛在因子模型(Latent Factor Model)。

降維



矩陣分解(Matrix Factorization)例子

	NERO	JULIUS CAESAR	CLEOPATRA	SLEEPLESS IN SEATLE	PTRTTY WOWAN	CASABLANCA
1	1	1	1	0	0	0
2	1	1	1	0	0	0
3	1	1	1	0	0	0
4	1	1	1	1	1	1
5	-1	-1	-1	1	1	1
6	-1	-1	1	1	1	1
7	-1	-1	-1	1	1	1

≈

	HISTORY	ROMANCE
1	1	0
2	1	0
3	1	0
4	1	1
5	-1	1
6	-1	1
7	-1	1

X

	NERO	JULIUS CAESAR	CLEOPATRA	SLEEPLESS IN SEATLE	PTRTTY WOWAN	CASABLANCA
HISTORY	1	1	1	0	0	0
ROMANCE	0	0	1	1	1	1

殘差矩陣(Residual Matrix)

	NERO	JULIUS CAESAR	CLEOPATRA	SLEEPLESS IN SEATLE	PTRTTY WOWAN	CASABLANCA
HISTORY	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
Both	0	0	-1	0	0	0
ROMANCE	0	0	-1	0	0	0
	0	0	-1	0	0	0
	0	0	-1	0	0	0

Matrix Factorization Example

1	1			-1	
1		1			
	1	1		-1	
1	1		1		1
-1		-1		1	
	-1	1			1
-1			1	1	

≈

X

Matrix Factorization Example

1	1	1		0	0	
2	1		1		0	0
3		1	1	0		0
4	1	1		1		1
5	-1		-1		1	
6		-1	1		1	1
7	-1			1	1	

≈

1	0
1	0
1	0
1	1
-1	1
-1	1
-1	1

X

1	1	1	0	0	0
0	0	1	1	1	1

Collaborative Filtering Tool Packages

Surprise 使用非常強的算法，如奇異值分解（**Singular Value Decomposition** SVD）來最小化RMSE（均方根誤差）並提供很好的建議。

Install the Surprise library using Anaconda:

$ conda install -c conda-forge scikit-surprise

Surprise provides various ready-to-use prediction algorithms such as baseline algorithms, neighborhood methods, matrix factorization-based algorithms, and many others. Also, various similarity measures (cosine, MSD, Pearson…) are built-in.

補充  時間

基線(Baseline) 預測法

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	7	6	7	4	5	4
User 2	6	7	?	4	3	4
User 3	?	3	3	1	1	?
User 4	1	2	2	3	3	4
User 5	1	?	1	2	3	3

如果我們還不知道怎麼找類似的用戶或商品，
你會如何預測用戶3的商品1和6的評級？

基線(Baseline) 預測法

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	7	6	7	4	5	4
User 2	6	7	?	4	3	4
User 3	?	3	3	1	1	?
User 4	1	2	2	3	3	4
User 5	1	?	1	2	3	3

- GLOBAL AVERAGE RATING = 3.9
- USER 3'S AVERAGE RATING:
  (3+3+1+1)/4 = 2
- ITEM 1'S AVERAGE RATING:
  3.75; ITEM 6'S AVERAGE RATING = 3.75
- ITEM 1'S RATING DIFFERENTIAL: ((7-5.5)+(6-4.8)+(1-2.5)+(1-2))/4 = 0.05
- ITEM 6'S RATING DIFFERENTIAL: ((4-5.5)+(4-4.8)+(4-2.5)+(3-2))/4 = 0.05
- USER 3'S BASELINE PREDICTION FOR ITEM 1 = 2 + 0.05 = 2.05
- USER 3'S BASELINE PREDICTION FOR ITEM 6 = 2 + 0.05 = 2.05

基線(Baseline) 預測法

- 可以用來與個性化的預測比較
- 對於為新用戶提供預測也是有用的
- 最簡單的基準是預測系統中所有評級的平均評分（全球均值）(3.9)
- 可以透過預測該用戶(2.0)或該商品的平均評分來加強 (3.75)
- 也可以透過將用戶平均值與特定商品的用戶平均評分的平均偏差相結合來進一步增強基線 (2.05)

基於商品最近鄰居的協同過濾

- Alice和一些其他用戶評分的數據庫：

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

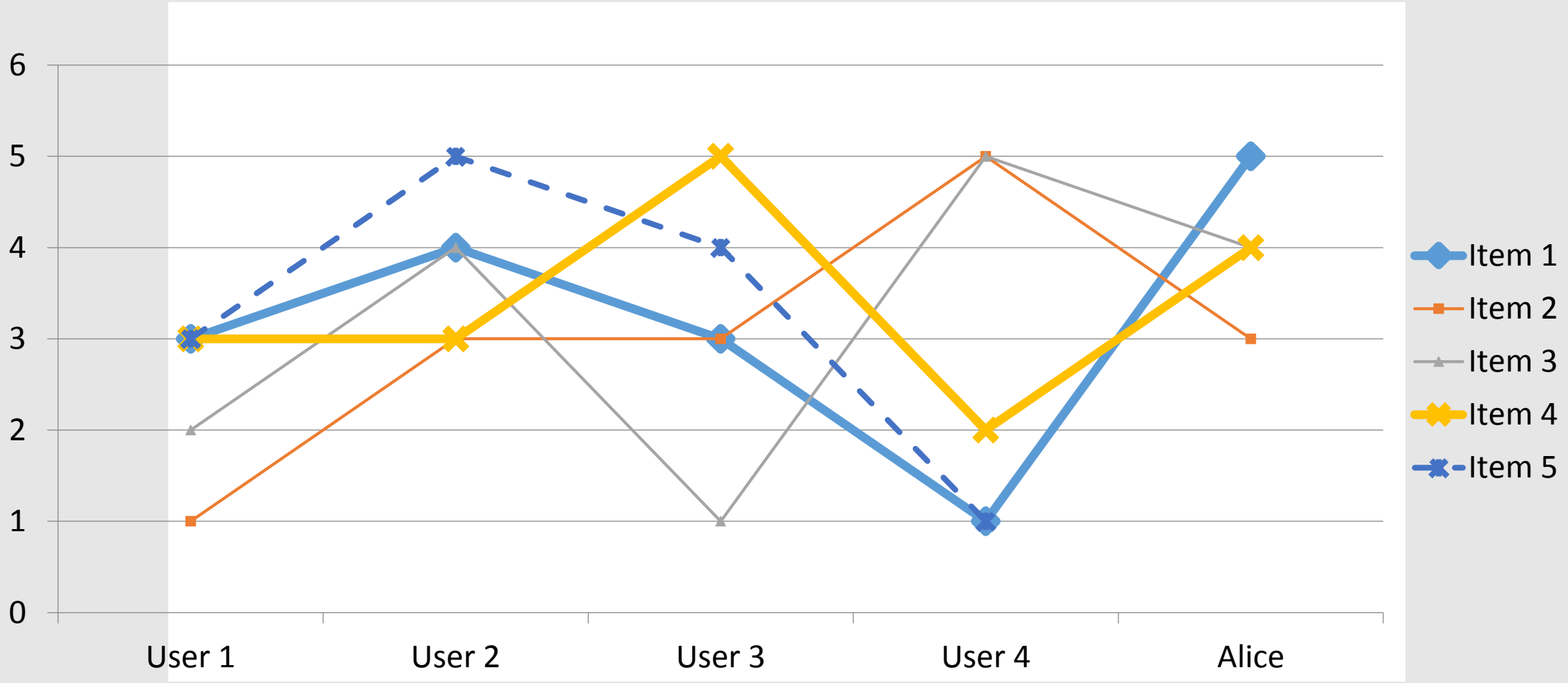
- 基於商品最近鄰居的協同過濾，是否應該把商品5推薦給Alice?

  請提供以下內容：
  - 商品5最近的鄰居
  - 基於商品的最近鄰居(一個和兩個鄰居)協同過濾，愛麗絲商品5的預測評分

基於商品最近鄰居的協同過濾

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

用戶商品評級可視化(基於商品)



基於平均中心餘弦的相似度

- $S(I5, I1) = \frac{0.6*0.6+0.2*1.2+(-0.2)*0.80+(-1.8)*(-1.8)}{\sqrt{0.6^2+0.2^2+(-0.2)^2+(-1.8)^2}*\sqrt{0.6^2+1.2^2+0.8^2+(-1.8)^2}} = 0.80$

- $S(I5, I4) = \frac{0.6*0.6+(-0.8)*1.2+2.8*0.80+(-0.8)*(-1.8)}{\sqrt{0.6^2+(-0.8)^2+2.8^2+(-0.8)^2}*\sqrt{0.6^2+1.2^2+0.8^2+(-1.8)^2}} = 0.42$

	Item1	Item2	Item3	Item4	Item5
Alice	1.00	-1.00	0.00	0.00	?
User1	0.60	-1.40	-0.40	0.60	0.60
User2	0.20	-0.80	0.20	-0.80	1.20
User3	-0.20	-0.20	-2.20	2.80	0.80
User4	-1.80	2.20	2.20	-0.80	-1.80

預測

- $p_{A5} = \frac{5*0.8+4*0.42}{(0.8+0.42)} = 4.66$ (2個鄰居)
- $p_{A5} = \frac{5*0.8}{0.8} = 5.0$ (1個鄰居)
- User-Based Prediction: 4.87

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

可以向Alice解釋評分的原因

潛在因子模型(Latent Factor Models)

- The basic idea is to exploit the fact that **significant portions of the rows and columns of data matrices are highly correlated** 高度相關; therefore, the data has built-in redundancies and the **matrix is often approximated quite well by a low-rank matrix**. 因此,這個矩陣通常能夠用低秩矩陣很接近地代表。
- The **low-rank approximation can be determined even with a small subset of the entries** in the original matrix and it often provides a robust estimation of the missing entries.即使使用原始矩陣中一小部分的值，也可以決定近似的低秩矩陣

潛在因子模型(Latent Factor Models)

- The **latent factor models are considered to be state-of-the–art** in recommender systems. These models leverage well-known **dimensionality reduction methods** to fill in the missing entries. 這些模型利用眾所周知的降維方法來填補缺失的值。
- Since the basic assumption for the collaborative filtering is that the **user-item ratings are highly correlated**, latent factor models have become the state-of-the-art in collaborative filtering and the ability to estimate latent factors with missing data is the key to the success of the latent factor approach.協作過濾的用戶產品評分高度相關, 因此能夠用潛在因子模型

潛在因子模型的低秩直覺

- 考慮評分矩陣R中的所有的值都被填滿的簡單情況。
- 秩(Rank)為 k << min {m，n}的任何 m×n 矩陣R可以用以下rank-k因子的乘積表示：

  R = UV^T 其中U是一個 m×k 矩陣，V是一個 n×k 矩陣
- 當矩陣R具有大於k的秩時，通常可以近似地表示為秩-k因子的乘積：

  R ≈ UV^T
- 這個近似矩陣的誤差等於||R - UV^T||^2
- 我們用一個7個用戶和電影的例子來說明矩陣分解及其殘差矩陣(residual matrix)。

基本矩陣分解原則
(Basic Matrix Factorization Principles)

- R $\approx UV^T$ where each column of U (or V) is referred to as **a latent vector** and each row of U (or V) is referred to as a **latent factor**
- U contains entries corresponding to the affinity of user i towards the **k concepts** in the ratings matrix while V contains entries corresponding to the affinity of the item j toward these k concepts.
- Therefore, **each rating r_{ij} can be approximately expressed as a dot product of the ith user factor and jth item factor**: $r_{ij} \approx \vec{u}_i \cdot \vec{v}_j$ where $\vec{u}_i = (u_{i1}, \ldots, u_{ik})$ and $\vec{v}_j = (v_{j1}, \ldots., v_{jk})$

基本矩陣分解原則
(Basic Matrix Factorization Principles)

- An intuitive way to express the above equation is as follows:

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

$$=$$

$\sum_{s=1}^{k}$(affinity of user i towards concepts) · (affinity of the item j toward these k concepts)

- The key differences among various matrix factorization methods arise in terms of the constraints imposed on U and V and the nature of the objective function

無約束矩陣分解 (Unconstrained Matrix Factorization)

- 為了找出完全填滿的用戶產品評分矩陣R的因子矩陣U和V，可以製定一個關於矩陣U和V的優化問題:

  $$Minimize\ J = \frac{1}{2}\|R - UV^T\|^2 \quad (Frobenius\ norm)$$

  $$Subject\ to:\ No\ constraints\ on\ U\ and\ V$$

- 然而，對於一個典型的協同過濾問題，評分矩陣只是部分填滿的，目標函數只能根據觀察到的值來重寫。
- 令在R中所有填值的用戶產品對（I，j）的集合由S表示$where\ i \in \{1, \ldots, m\} and\ j \in \{1, \ldots, n\}$;
- 因此，集合S被定義如下: S={(I, j): r_{ij}有填值}
- 如果我們可以將不完全填滿的矩陣R分解為近似的乘積UV^T of fully specified matrices $U = [u_{is}]_{mxk}$ and $V = [v_{js}]_{nxk}$,那麼R中的所有值都可以被預測: $\hat{r}_{ij} = \sum_{s=1}^{k} u_{is}v_{js}$

無約束矩陣分解 (Unconstrained Matrix Factorization)

- 因此，優化問題可以被重寫為:

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

Subject to: No constraints on U and V

- 這可以通過梯度下降方法來解決。 因此，需要計算目標函數的偏導數(partial differentiation) w.r.t. u_{is} and v_{js}.
- $\frac{\partial J}{\partial u_{is}} = \sum_{j:(i,j)\in S}(r_{ij} - \sum_{s=1}^{k} u_{is}v_{js})(-v_{js})\ \forall i \in \{1,\dots,m\},$
- $s \in \{1,\ ,k\} = \sum_{j:(i,j)\in S}(e_{ij}\ )(-v_{js})\ \forall i \in \{1,\dots,m\}, s \in \{1,\ ,k\}$
- $\frac{\partial J}{\partial v_{js}} = \sum_{j:(i,j)\in S}(r_{ij} - \sum_{s=1}^{k} u_{is}v_{js})(-u_{is})\ \forall j \in \{1,\dots,n\},$
- $s \in \{1,\ ,k\} = \sum_{j:(i,j)\in S}(e_{ij}\ )(-u_{is})\ \forall j \in \{1,\dots,n\}, s \in \{1,\ ,k\}$
- 然後，我們可以使用梯度下降算法來導出U和V.
- 在將不完全矩陣R分解為U和V的近似乘積之後，第（I，j）個缺失值, $\hat{r}_{ij}$,可以預測如下:

$$\hat{r}_{ij} = \sum_{s=1}^{k} u_{is} \cdot v_{js}$$