Collaborative Recommendation (Part B)

Topic 4B

Model-Based Approaches

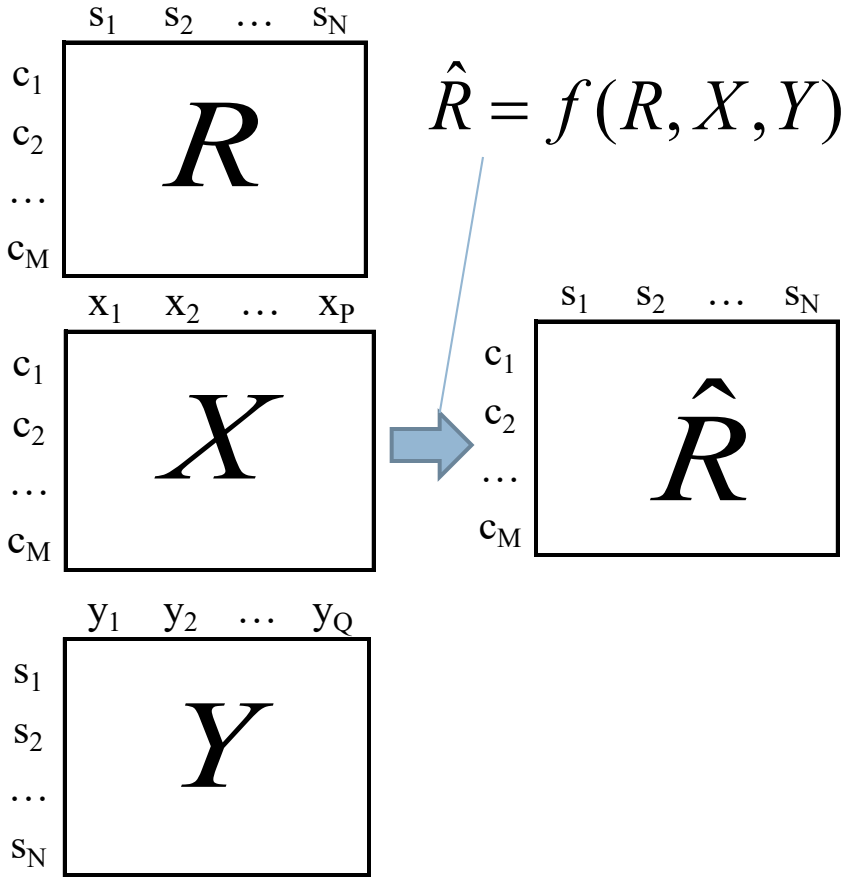
- Overall, collaborative filtering can be categorized into memory-based and model-based approaches.
- In model-based approaches, a model is created offline and then it is used to predict user ratings online.
  - Matrix factorization/latent factor models
  - Association rule mining
  - Probabilistic recommendation approaches
- Model-based approaches are more computationally complex than memory-based approaches, but they have several advantages.

Matrix Factorization/Latent Factor Models

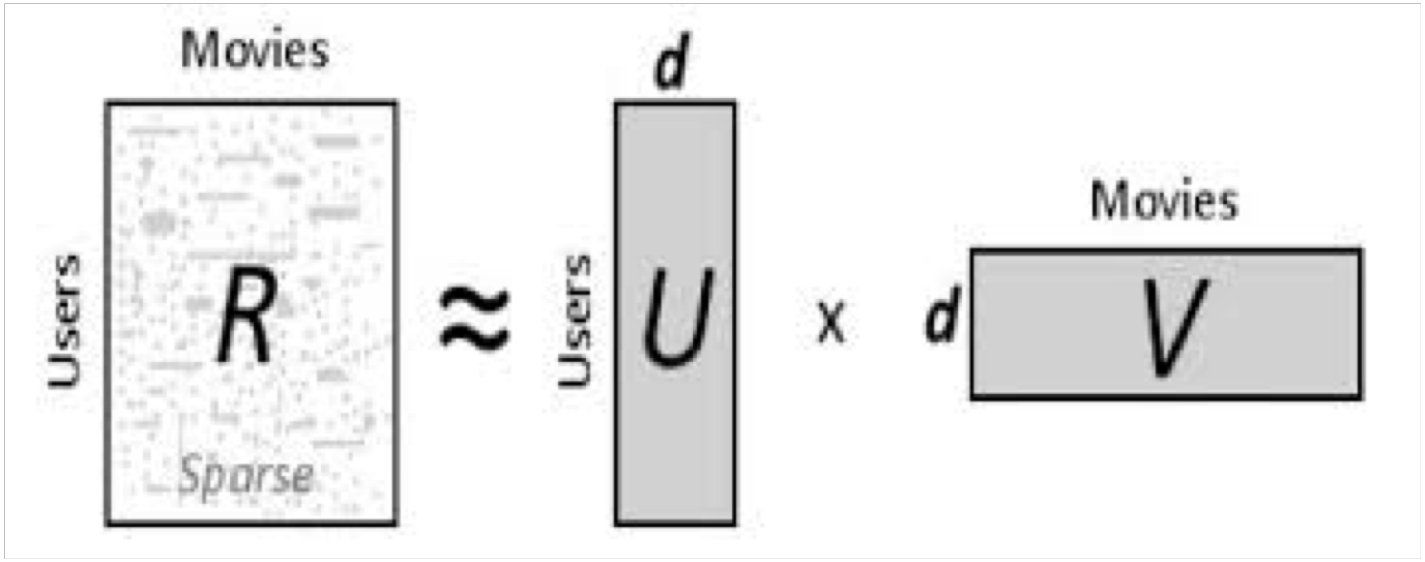
- Matrix factorization methods can be used in recommender systems to derive a set of latent (hidden) factors from the rating patterns and characterize both users and items by such vectors of factors.
- In the movie domain, those latent factors can correspond to some important aspects of a movie such as the genre or uninterpretable.

Recommender System Fundamentals - Revisited

- Input
  - Rating matrix *R*:
    - r_{ij}: rating user c_i assigns to item s_i
    - *R* is a *sparse* matrix
  - User attribute matrix *X*: x_{ij} – *attribute* x_j *of user* c_i
  - Item attribute matrix *Y*: y_{ij} – *attribute* y_j *of item* s_i
- Output
  - Predicted interaction matrix (predicted utility) $\hat{R}$

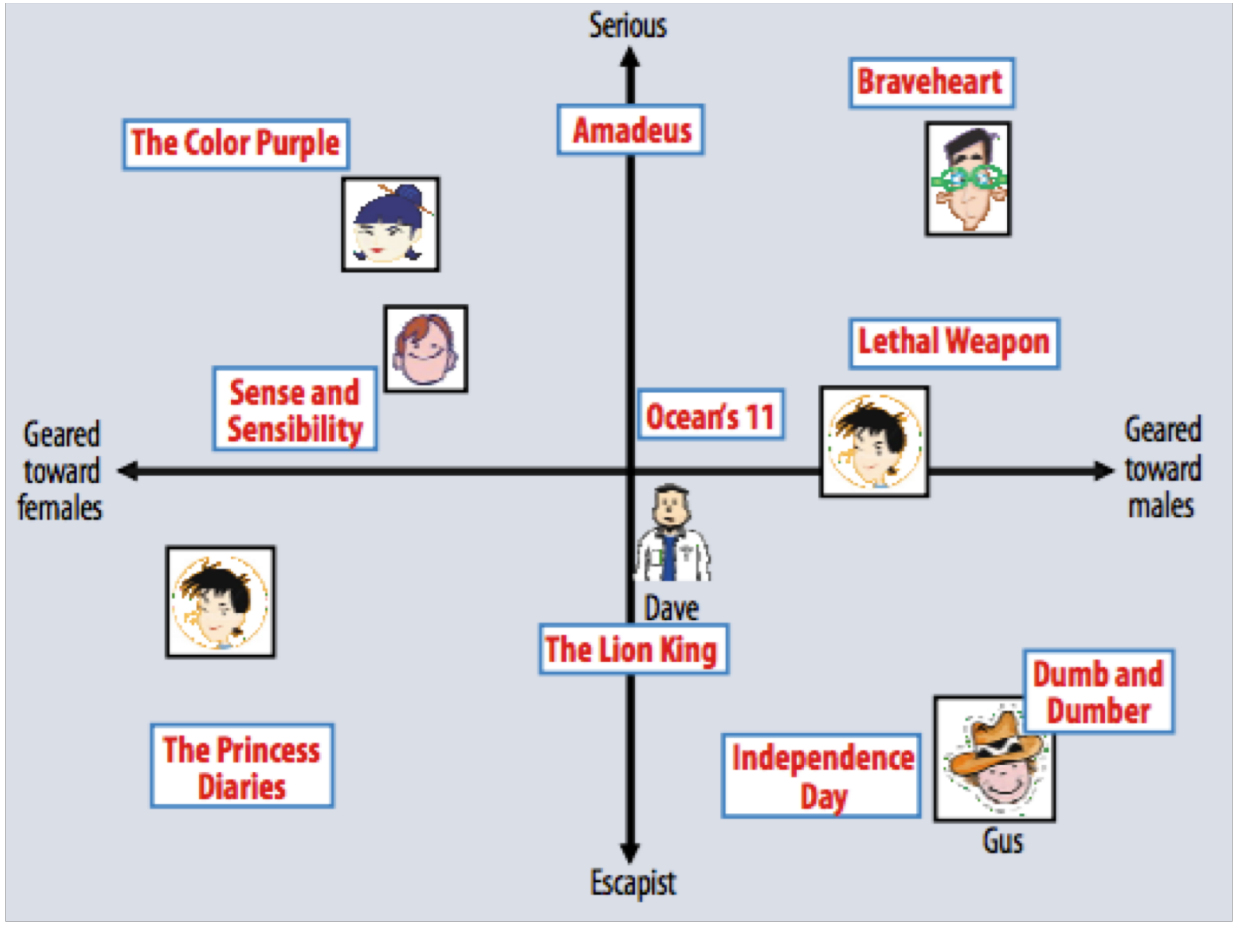


Laten Factor Approach - Movie Recommendation



Alice	= 10% Action fan	+10% Comedy fan	+50% Romance fan	+⋯
Bob	= 50% Action fan	+30% Comedy fan	+10% Romance fan	+⋯
Titanic	= 20% Action	+00% Comedy	+70% Romance	+⋯
Toy Story	= 30% Action	+60% Comedy	+00% Romance	+⋯

Laten Factor Approach - Movie Recommendation

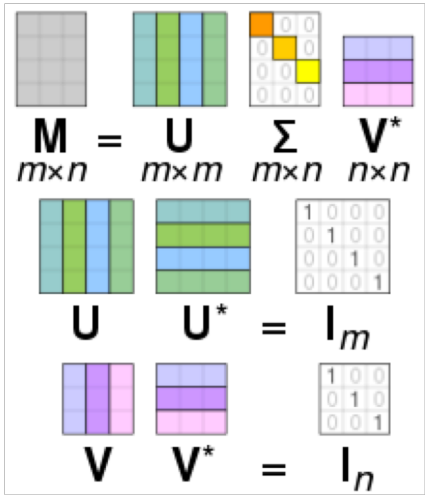


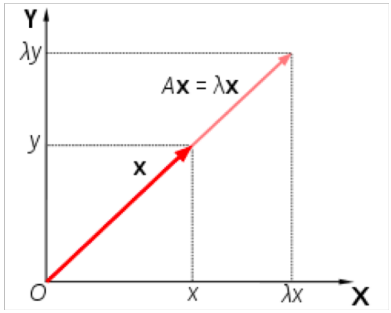
SVD: $A=U\Sigma V^T$

- To discover the latent factors of Users and Items, SVD (Singular Value Decomposition) can be applied.

$A=U\Sigma V^T$

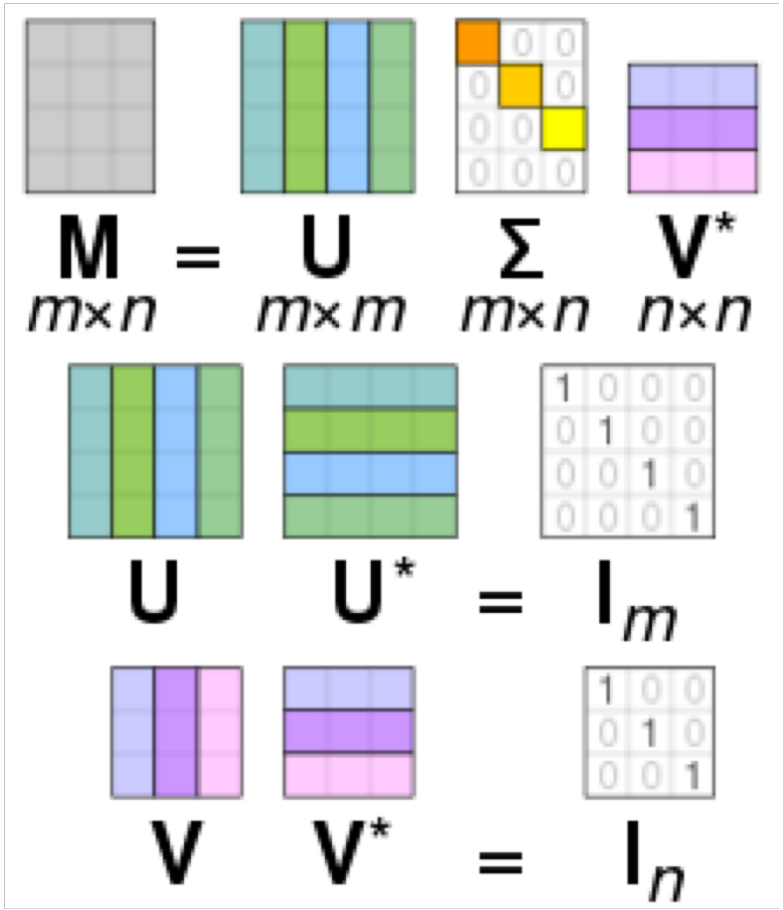
- The columns of V (right-singular vectors) are eigenvectors of A^TA.
- The columns of U (left-singular vectors) are eigenvectors of AA^T.
- The non-zero elements of Σ (non-zero singular values) are the square roots of the non-zero eigenvalues of A^TA or AA^T.
- The columns of U and V are orthonormal and the matrix Σ is diagonal with positive real entries.
- The singular values are the diagonal entries of the Σ matrix and are arranged in descending order





* In linear algebra, an eigenvector or characteristic vector of a linear transformation is a non-zero vector that changes by only a scalar factor when that linear transformation is applied to it.

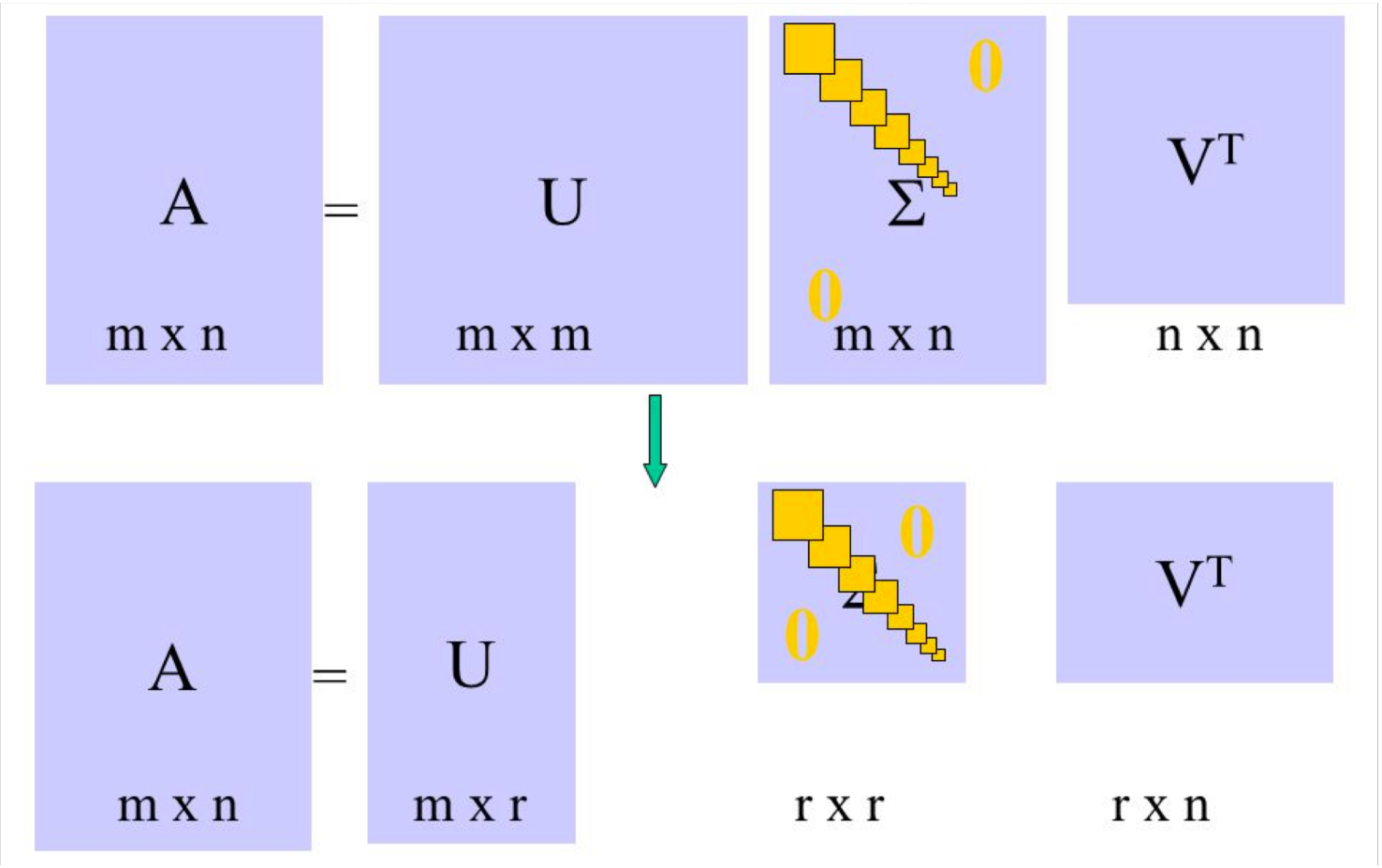
SVD: $A=U\Sigma V^T$



From https://en.wikipedia.org/wiki/Singular_value_decomposition

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

- Full rank vs. Reduced rank



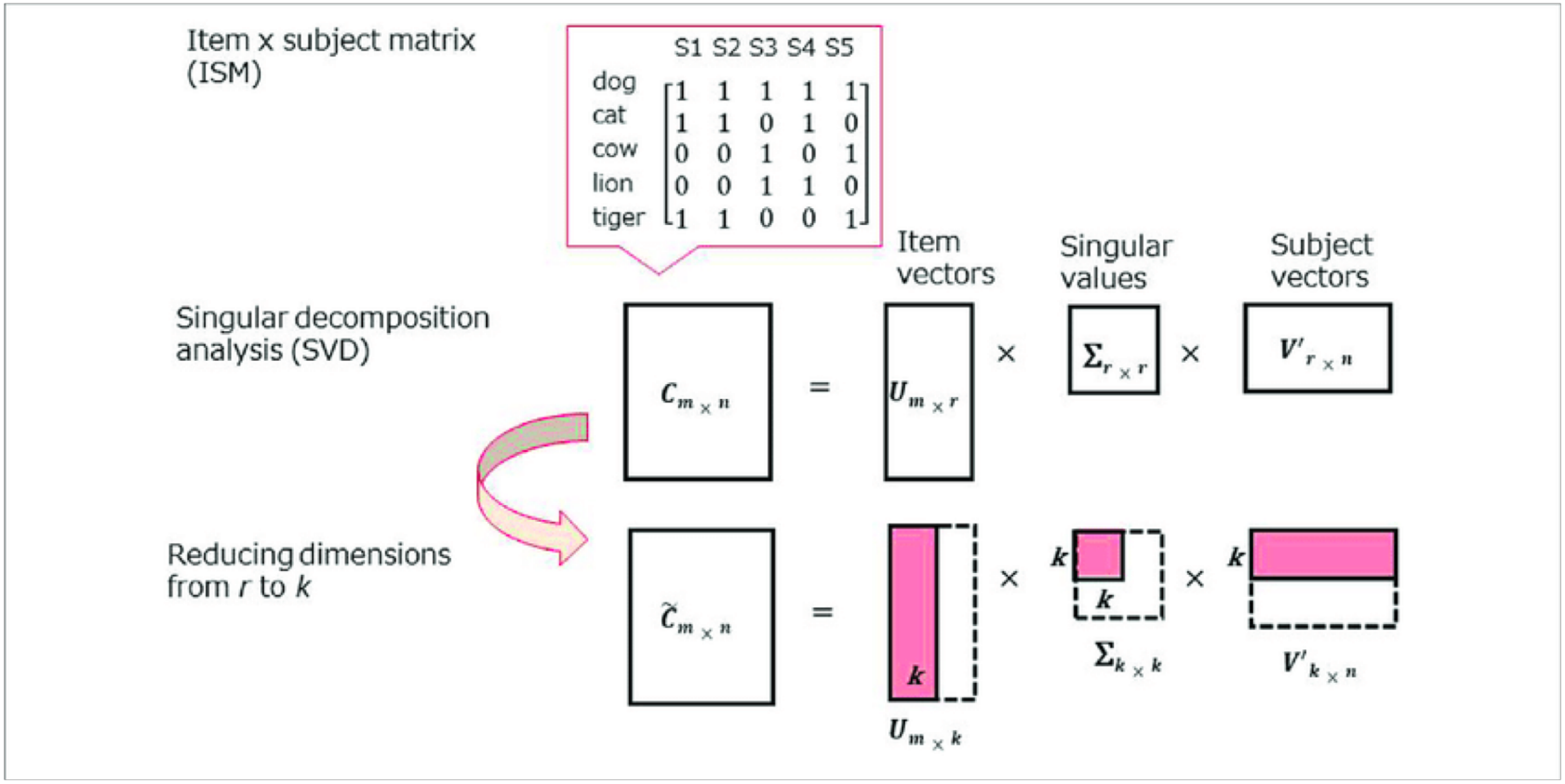
SVD Example

$$A = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix}$$

$$AA^T = \begin{pmatrix} 17 & 8 \\ 8 & 17 \end{pmatrix}$$

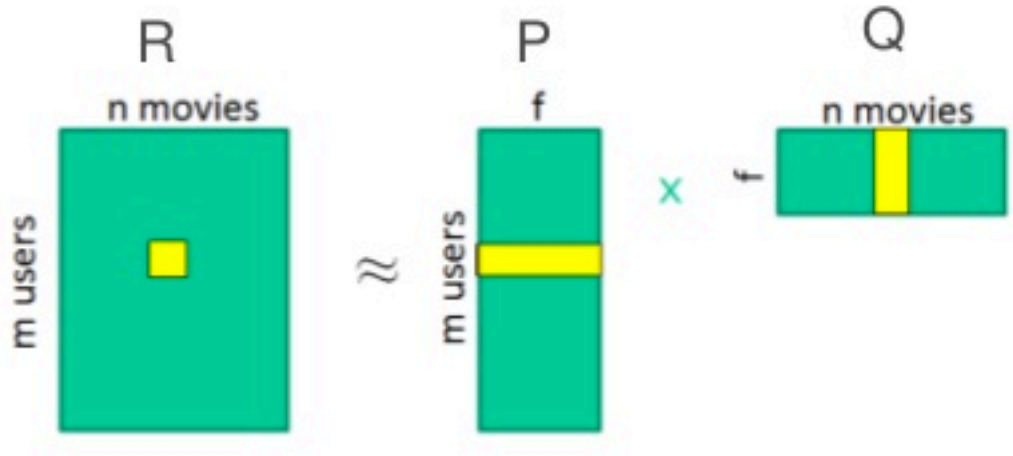
$$A = U\Sigma V^T = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}$$

From SVD to MF



From SVD to MF

In SVD, $A=U\Sigma V^T$, $U\Sigma$ can be combined, thus, reducing it to two components, $A=U^+V^T$,



$$r_{ui} = p_u \cdot q_i = \sum_{f \in \text{latent factors}} \text{affinity of } u \text{ for } f \times \text{affinity of } i \text{ for } f$$

Alice	= 10% Action fan	+10% Comedy fan	+50% Romance fan	+⋯
Bob	= 50% Action fan	+30% Comedy fan	+10% Romance fan	+⋯
Titanic	= 20% Action	+00% Comedy	+70% Romance	+⋯
Toy Story	= 30% Action	+60% Comedy	+00% Romance	+⋯

$$p_{\text{Alice}} = (10\%,\ 10\%,\ 50\%,\ \cdots)$$
$$p_{\text{Bob}} = (50\%,\ 30\%,\ 10\%,\ \cdots)$$
$$q_{\text{Titanic}} = (20\%,\ 00\%,\ 70\%,\ \cdots)$$
$$\text{oy Story} = (30\%,\ 60\%,\ 00\%,\ \cdots)$$

From SVD to MF

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

Changed to

Table 2.4. *Ratings database for SVD-based recommendation.*

	User1	User2	User3	User4
Item1	3	4	3	1
Item2	1	3	2	~~0~~ 5
Item3	2	4	1	5
Item4	3	3	5	2

From SVD to MF

Table 2.4. *Ratings database for SVD-based recommendation.*

	User1	User2	User3	User4
Item1	3	4	3	1
Item2	1	3	2	~~0~~ 5
Item3	2	4	1	5
Item4	3	3	5	2

Table 2.5. *First two columns of decomposed matrix and singular values* Σ.

U_2		V_2		Σ_2	
−0.4312452	0.4931501	−0.3593326	0.36767659	12.2215	0
−0.5327375	−0.5305257	−0.5675075	0.08799758	0	4.9282
−0.5237456	−0.4052007	−0.4428526	0.56862492		
−0.5058743	0.5578152	−0.5938829	−0.73057242		

From SVD to MF

Table 2.5. *First two columns of decomposed matrix and singular values* Σ.

U_2		V_2		Σ_2	
−0.4312452	0.4931501	−0.3593326	0.36767659	12.2215	0
−0.5327375	−0.5305257	−0.5675075	0.08799758	0	4.9282
−0.5237456	−0.4052007	−0.4428526	0.56862492		
−0.5058743	0.5578152	−0.5938829	−0.73057242		

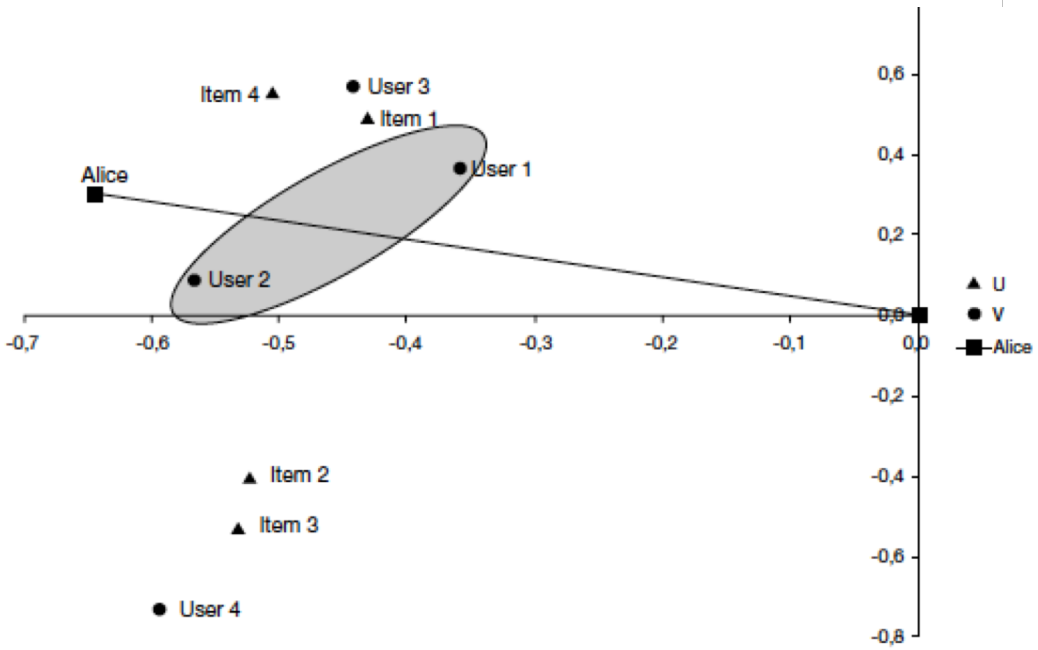


Figure 2.3. SVD-based projection in two-dimensional space.

For Alice with the rating vector [5, 3, 4, 4],

$$Alice_{2D} = Alice \times U_2 \times \Sigma_2^{-1} = [-0.64, 0.30]$$

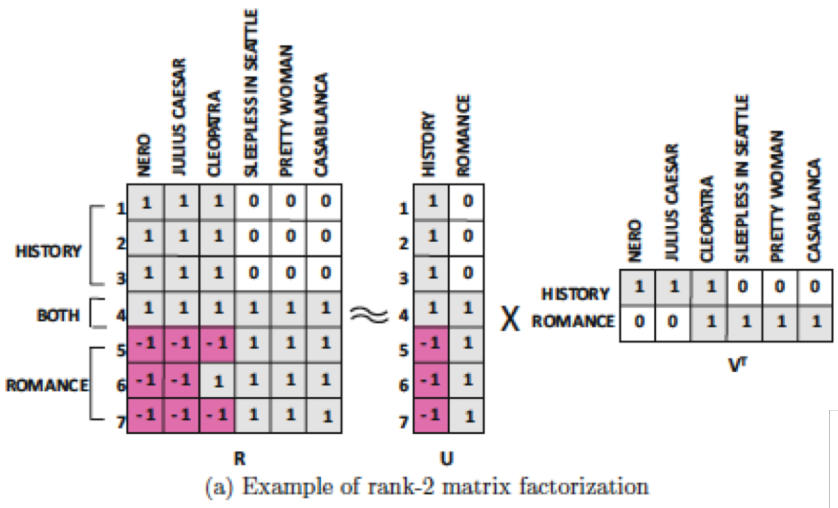
SVD and Missing Values

- SVD requires A to be dense, but in reality, A is sparse (contain missing entries).
- One option is to use the mean of the columns (or the rows) – this works, but highly biased.
- The alternative is to use a function that allows us to compute all the vectors of p_u and q_i (p_u make up the rows of U and q_u make up the columns of V^T, such that
  - $r_{ui}=p_u \cdot q_i$ for all u and i
  - All the vectors pu are mutually orthogonal, as well as the vectors qi
  - We can find such vectors of p_u and q_i by solving

$$\min_{p_u,q_i} \sum_{r_{ui} \in R} (r_{ui} - p_u \cdot q_i)^2$$

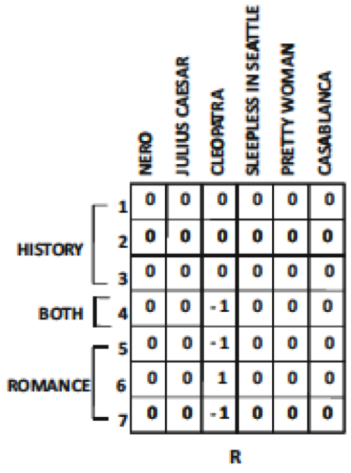
  - An approximate solution to this optimization problem can be found using a gradient descent procedure such as SGD (Stochastic Gradient Descent).

MF and Residual Matrix



(a) Example of rank-2 matrix factorization

$$\min_{p_u, q_i} \sum_{r_{ui} \in R} (r_{ui} - p_u \cdot q_i)^2$$



(b) Residual matrix

Gradient Descent Algorithm

Algorithm *GD*(Ratings Matrix: R, Learning Rate: α)
begin
  Randomly initialize matrices U and V;
  $S = \{(i,j) : r_{ij}$ is observed$\}$;
  while not(convergence) **do**
  begin
    Compute each error $e_{ij} \in S$ as the observed entries of $R - UV^T$;
    for each user-component pair (i,q) **do** $u_{iq}^+ \Leftarrow u_{iq} + \alpha \cdot \sum_{j:(i,j)\in S} e_{ij} \cdot v_{jq}$;
    for each item-component pair (j,q) **do** $v_{jq}^+ \Leftarrow v_{jq} + \alpha \cdot \sum_{i:(i,j)\in S} e_{ij} \cdot u_{iq}$;
    for each user-component pair (i,q) **do** $u_{iq} \Leftarrow u_{iq}^+$;
    for each item-component pair (j,q) **do** $v_{jq} \Leftarrow v_{jq}^+$;
    Check convergence condition;
  end
end

Updates until convergence:

$$U \Leftarrow U + \alpha EV$$
$$V \Leftarrow V + \alpha E^T U$$

* For python implemtation of SGD, see http://nicolas-hug.com/blog/matrix_facto_4

Wrap-up

- There are many other matrix decomposition approaches
  - For more, see https://en.wikipedia.org/wiki/Matrix_decomposition
- In some cases, the prediction quality was worse when compared with memory-based prediction techniques; in other cases it was better (Sarwar et al. 2000a).
- It was used to win the Netflix Prize competition.
- MF approaches are commonly used in real-world recommender systems.

Association Rule Mining

- A common technique used to identify rulelike relationship patterns in large-scale sales transactions.
- A typical rule could be, “If a customer purchases baby food, then he also buys diapers in 70% of the cases.”
- For recommendation of items, “If user X liked both item1 and item2, then X will most probably also like item5.”

Association Rule Mining

- Association rules are often written in the form of X => Y, with X and Y being both subsets of P and X∩Y = ∅.
- Measures

$$support = \frac{number\ of\ transactions\ containing\ X \cup Y}{number\ of\ transactions} \qquad (2.12)$$

$$confidence = \frac{number\ of\ transactions\ containing\ X \cup Y}{number\ of\ transactions\ containing\ X} \qquad (2.13)$$

Association Rule Mining - Example

Table 2.6. *Transformed ratings database for rule mining.*

	Item1	Item2	Item3	Item4	Item5
Alice	1	0	0	0	?
User1	1	0	0	1	1
User2	1	0	1	0	1
User3	0	0	0	1	1
User4	0	1	1	0	0

A user who liked Item1 will also like 5 (Item1 => Item5)

- Support: 2/4
- Confidence: 2/2

Association Rule Mining

- Making recommendations:

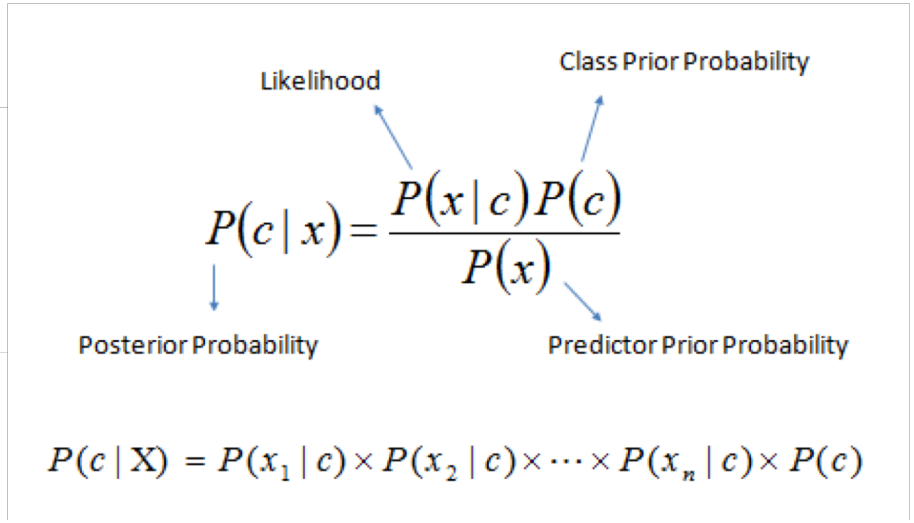
(1) Determine the set of $X \Rightarrow Y$ association rules that are relevant for Alice – that is, where Alice has bought (or liked) all elements from X. Because Alice has bought *Item1*, the aforementioned rule is relevant for Alice.
(2) Compute the union of items appearing in the consequent Y of these association rules that have not been purchased by Alice.
(3) Sort the products according to the confidence of the rule that predicted them. If multiple rules suggested one product, take the rule with the highest confidence.
(4) Return the first N elements of this ordered list as a recommendation.

$$score_{item_i} = \sum_{rules\ recommending\ item_i} (support_{rule} * confidence_{rule}) \qquad (2.14)$$

Probabilistic Recommendation

- One way to implement collaborative filtering with a probabilistic method is to view the prediction problem as a classification problem, which can be generally be described as the task of "assigning an object to one of several predefined categories."
- Bayes theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$



Probabilistic Recommendation - Example

Table 2.7. *Probabilistic models: the rating database.*

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

$$P(Y|X) = \frac{\prod_{i=1}^{d} P(X_i|Y) \times P(Y)}{P(X)}$$

$P(X_i \mid Y)$

P(X) is a constant value

P(Y):

P(Item5=1) = 2/4

P(Item5=2) = 0

…

$$\begin{aligned} P(X|Item5=1) &= P(Item1=1|Item5=1) \times P(Item2=3|Item5=1) \\ &\quad \times P(Item3=3|Item5=1) \times P(Item4=2|Item5=1) \\ &= 2/2 \times 1/2 \times 1/2 \times 1/2 \\ &= 0.125 \end{aligned}$$

$$\begin{aligned} P(X|Item5=2) &= P(Item1=1|Item5=2) \times P(Item2=3|Item5=2) \\ &\quad \times P(Item3=3|Item5=2) \times P(Item4=2|Item5=2) \\ &= 0/0 \times \cdots \times \cdots \times \cdots \\ &= 0 \end{aligned}$$

$$P(Item5 = 1|X) = 2/4 \times 0.125 = 0.0625$$