Matrix Factorization Methods for Recommender Systems

Data Mining
Prof. Dawn Woodard
School of ORIE
Cornell University

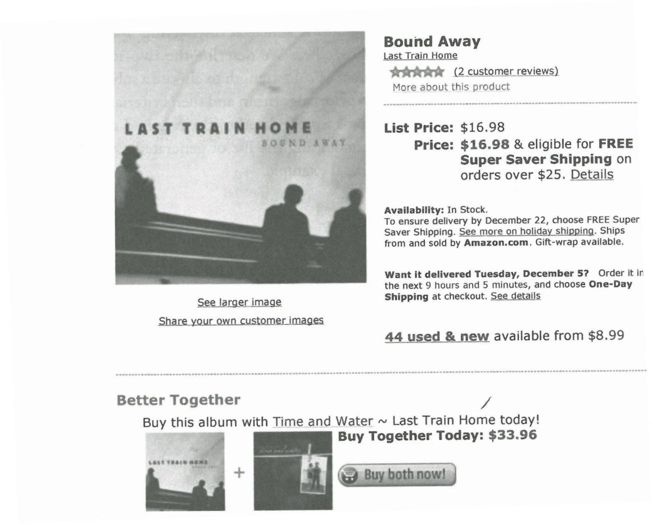
Recommender Systems

Recommender Systems:

- Achieve a similar goal to association rules, but use different techniques
- Have data on rating or purchase of products by various customers
- While association rules learn rules of the form “If A is purchased then B is likely to be purchased”, recommender systems identify products that have been rated highly/purchased by customers with similar tastes to yours
- Today’s lecture is drawn from Koren, Bell, and Volinsky (2009; “Matrix Factorization Techniques for Recommender Systems”)

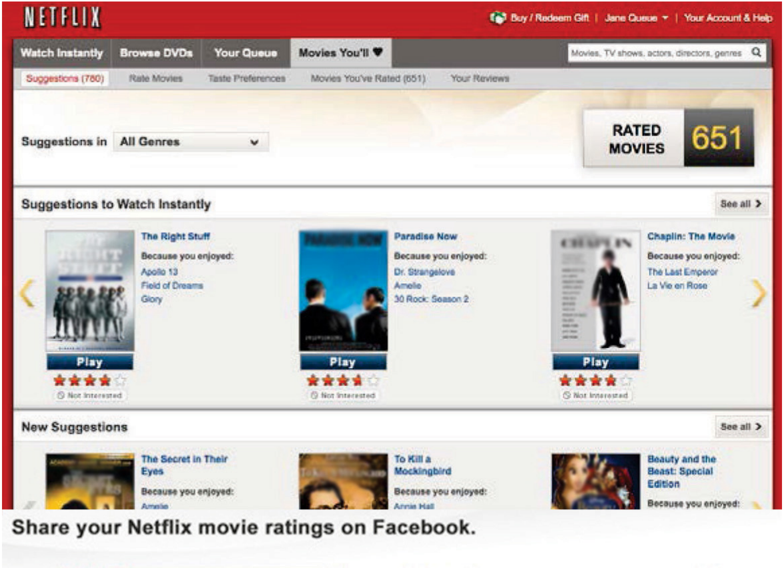
Recommender Systems

From Amazon.com (Fig. from SPB text):



Recommender Systems

From Netflix:



Recommender Systems

Data takes the form of a matrix of ratings of different products by different users, where many of the values are missing because the user has not rated the product:

		Movies						
		101	102	103	104	105	106	107
Users		Rambo	Rocky	Garden State	Before Sunset	Training Day	Thor	Black Swan
1	Alice	5.0	3.0	2.5				
2	Bob	2.0	2.5	5.0	2.0			
3	Charlie	2.5			4.0	4.5		5.0
4	Damon	5.0		3.0	4.5		4.0	
5	Eddie	4.0	3.0	2.0	4.0	3.5	4.0	

Figure 1: User ratings for movies on a scale of 1-5

Recommender Systems

The goal is to fill in the missing data: if Alice had rated Black Swan, what rating would she have given it?

	Users	Movies						
		101	102	103	104	105	106	107
		Rambo	Rocky	Garden State	Before Sunset	Training Day	Thor	Black Swan
1	Alice	5.0	3.0	2.5				
2	Bob	2.0	2.5	5.0	2.0			
3	Charlie	2.5			4.0	4.5		5.0
4	Damon	5.0		3.0	4.5		4.0	
5	Eddie	4.0	3.0	2.0	4.0	3.5	4.0	

Figure 1: User ratings for movies on a scale of 1-5

Matrix Factorization

Matrix Factorization:

- The method used to win the Netflix Prize in 2009
- Idea: Try to explain the ratings by identifying a moderate # of abstract "factors" that characterize the items.
- Learn these factors from the data
- For movies, these factors might capture:
  -
  -
  -
- If a movie has a high value for the factor capturing amount of action, for instance, the movie has a lot of action

Matrix Factorization

- The users each have values for the SAME set of factors.
- For users, each factor measures:

Recommender Systems

Illustration: 2 factors capture female- vs. male-oriented and serious vs. escapist:

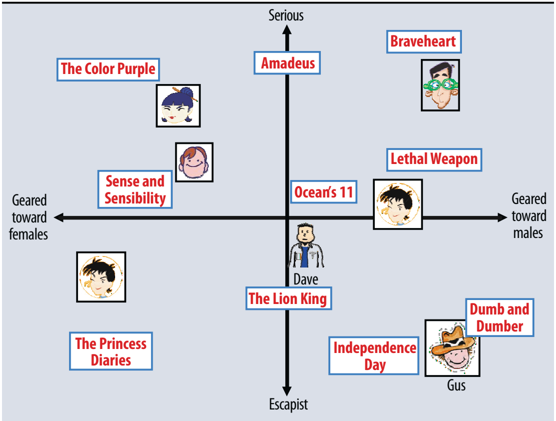


Figure 2 in Koren, Bell, and Volinsky (2009)

Recommender Systems

The factor values of some movies are shown, along with those of some hypothetical users:

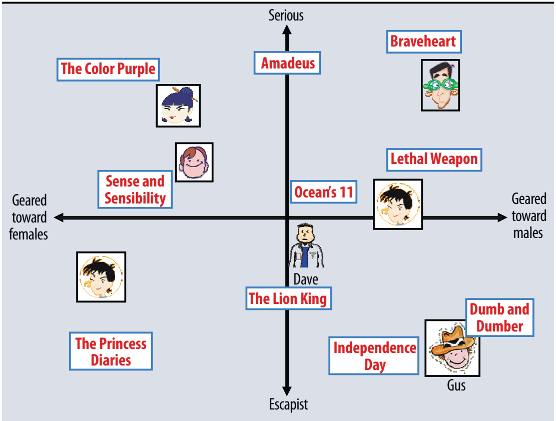


Figure 2 in Koren, Bell, and Volinsky (2009)

Matrix Factorization

- To calculate the estimated rating for a movie by a user, we would take the **inner product** of the movie's factor vector and the user's factor vector
- Example:

Recommender Systems

So we would expect Gus to love Independence Day, to hate The Color Purple, and to rate Braveheart about average

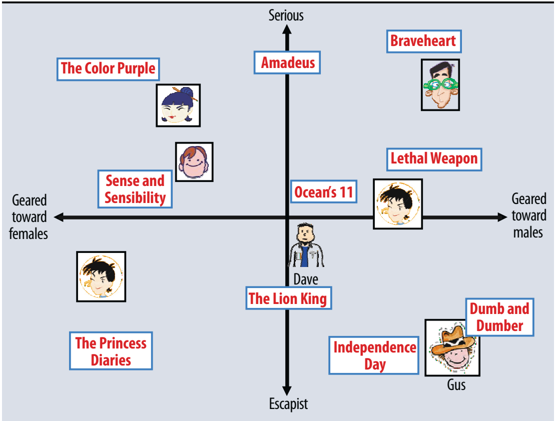


Figure 2 in Koren, Bell, and Volinsky (2009)

Matrix Factorization

Matrix Factorization in general:

- We want to identify some number K of abstract factors, and estimate the factor vectors for all the items and all the users.
- For each item i, write the unknown length-K factor (column) vector as q_i
- For each user u, write the unknown length-K factor (column) vector as p_u
- For the user-item pairs with known rating, write it r_{ui}
- The estimated rating is defined as

$$\hat{r}_{ui} = q_i^T p_u.$$

-

Matrix Factorization

How to estimate the q_i and p_u?

- We want the difference between $\hat{r}_{ui}$ and r_{ui} to be close for the user-item pairs for which we have observed a rating
- So perhaps we could choose q_i and p_u to minimize the sum of squared errors:

Matrix Factorization

- This approach tends to overfit to the data.
- The method works better if we create an incentive for the q_i and p_u to be close to the origin (when it does not increase the error much)
- So we instead minimize the objective

$$\sum_{(u,i)\in\mathcal{A}} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

  where $\lambda > 0$ penalizes the length of q_i and p_u
- λ controls:

- We can choose λ by cross-validation (minimize RMSE of observed ratings on validation data)!

The Netflix Prize

- In 2006 Netflix announced a contest to improve its methods for movie recommendation
- Netflix released a training set of more than 100 million ratings from 500,000 anonymized customers, for more than 17,000 movies
- Each movie is rated on a scale 1-5
- Competing teams submitted predicted user ratings for a test set of 3 million ratings
- Then Netflix would calculate the RMSE on the test set
- The first team to improve over the Netflix algorithm by $>$ 10% was to win $1,000,000

The Netflix Prize

- The prize was won in 2009 by the team “BellKor’s Pragmatic Chaos” with 10.09% improvement over Netflix’s method
- Unfortunately Netflix was sued for privacy violations after it was discovered that some users could be identified by matching the anonymized ratings with film ratings on the Internet Movie Database

Recommender Systems

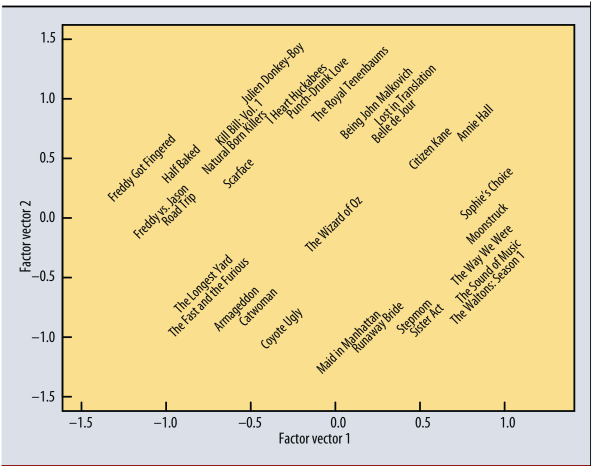


Figure 3. The first two vectors from a matrix decomposition of the Netflix Prize data. Selected movies are placed at the appropriate spot based on their factor vectors in two dimensions. The plot reveals distinct genres, including clusters of movies with strong female leads, fraternity humor, and quirky independent films.

Figure 3 in Koren, Bell, and Volinsky (2009)