Latent Factor Recommenders

David Yang
xdyang70@gmail.com

Key Idea

- Given a list of users and items, and user-item interactions, predict user behavior
- Use only user-item interactions to build latent vectors
    - **P** = User Latent Vector
    - **Q** = Item Latent Vector
- Make recommendations based on **distance(Q, P)**
- Contrast with content-based filtering, which builds a model for each item, E.g., Pandora hires musicologists to characterize songs, built the Music Genome

Sex **Seriousness**

Item

User	W	X	Y	Z
A		4.5	2.0	
B	4.0		3.5	
C		5.0		2.0
D		3.5	4.0	1.0

Rating Matrix

=

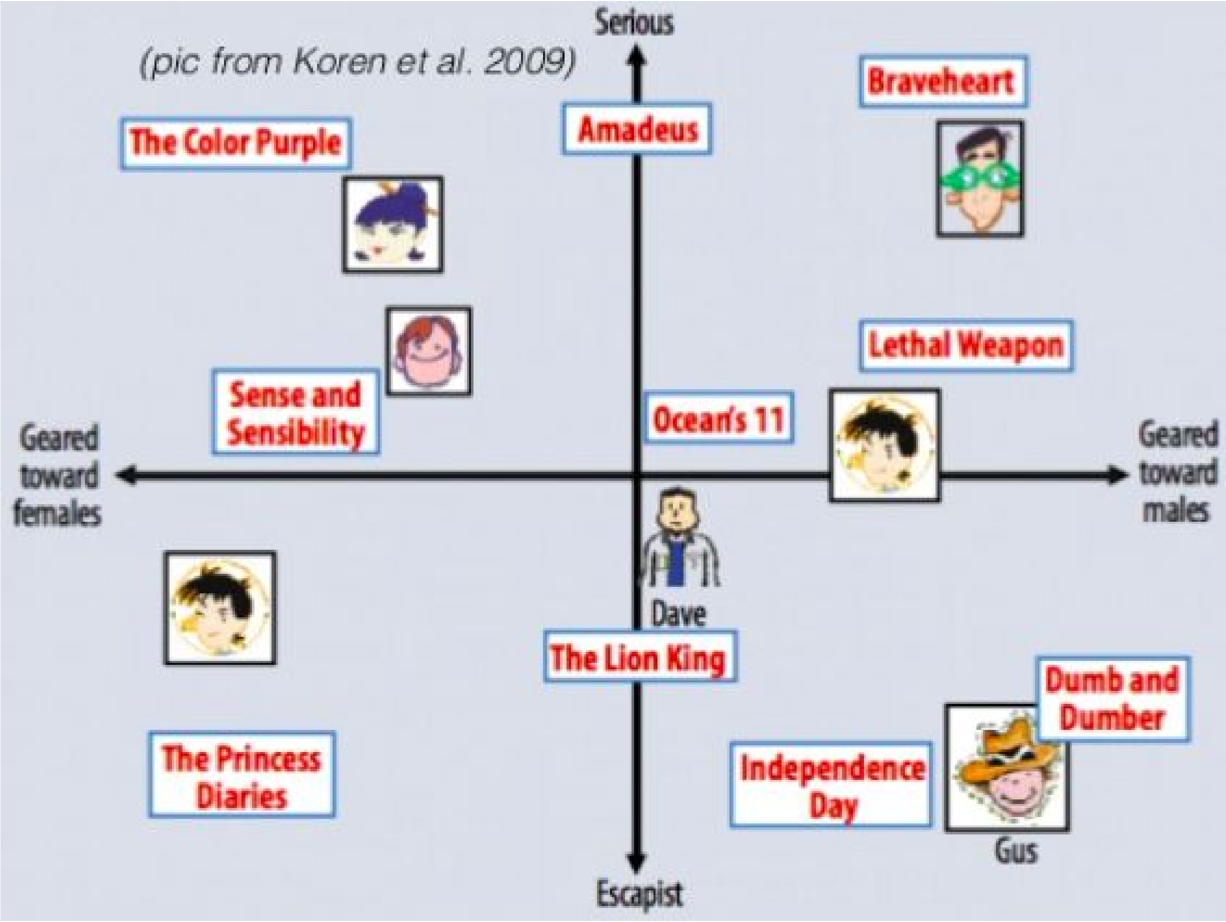
	p1	p2
A	*1.2*	*0.8*
B	*1.4*	*0.9*
C	*1.5*	*1.0*
D	*1.2*	*0.8*

User Matrix

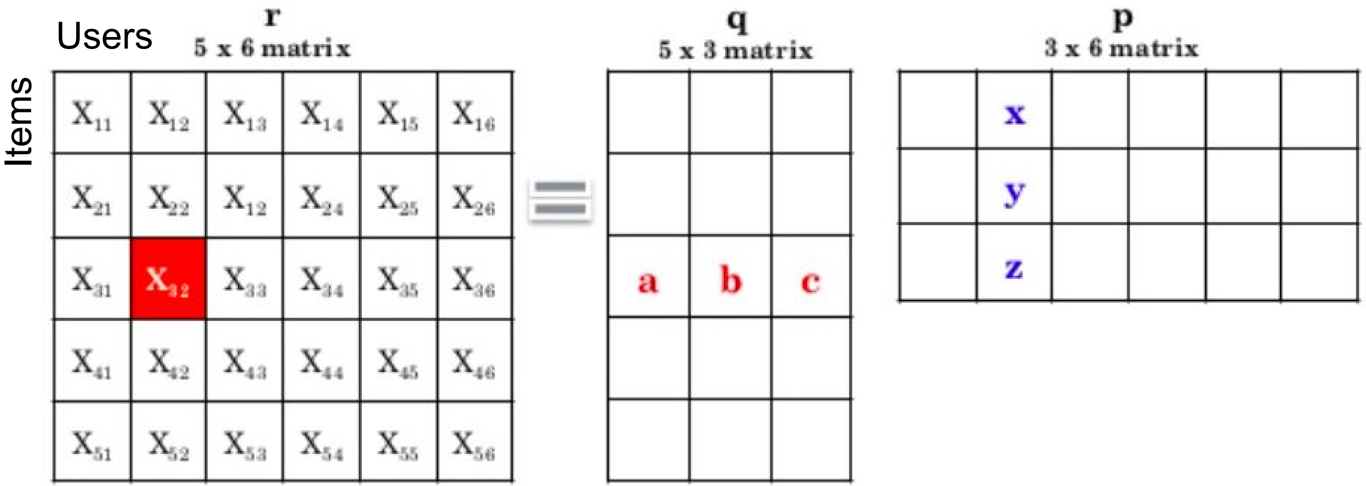
X

	W	X	Y	Z
q1	*1.5*	*1.2*	*1.0*	*0.8*
q2	*1.7*	*0.6*	*1.1*	*0.4*

Item Matrix


(pic from Koren et al. 2009)
Serious
Escapist
Geared toward females
Geared toward males
The Color Purple
Amadeus
Braveheart
Lethal Weapon
Sense and Sensibility
Ocean's 11
Dave
The Lion King
The Princess Diaries
Independence Day
Dumb and Dumber
Gus

Matrix Factorization



$$X_{32} = (a, b, c) \,.\, (x, y, z) = a * x + b * y + c * z$$

Rating Prediction

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

User Preference Factor Vector

Movie Preference Factor Vector

Challenge of SVD: Missing Values

Solution: model observing ratings only

$$\min_{q^\cdot,p^\cdot} \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2)$$

Avoid overfitting via regularization

Algorithm 1. SGD (stochastic gradient descent)

Algorithm 2. ALQ (alternating least squares)

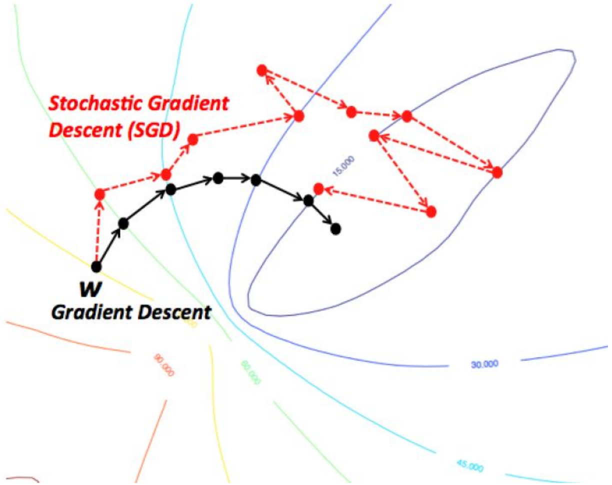
SGD

- Incremental Learning
- For each training case, predict r_{ui} and compute the prediction error

$$e_{ui} \stackrel{def}{=} r_{ui} - q_i^T p_u.$$

- Update the parameters

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$
$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$



ALS

- The objective function is not convex, since both q_i and p_u are unknown vectors
- But, if we fix one vector, the problem becomes quadratic in the other vector.
- Thus, LSE (least squares estimation) can be used alternatively, i.e.,

$$Q^T \leftarrow \left(P^T P\right)^{-1} P^T R$$

$$P \leftarrow RQ\left(Q^T Q\right)^{-1}$$

Extensions

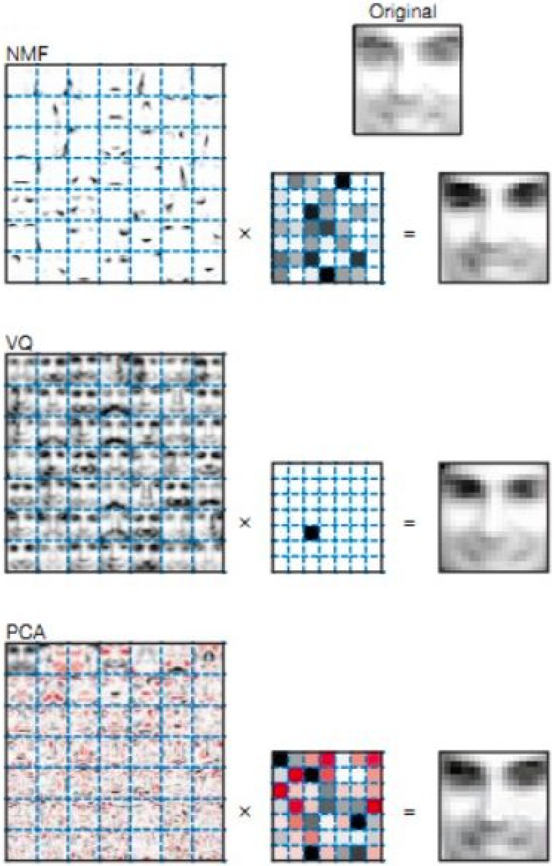
- Add Bias Term

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u$$

- Handle Temporal Dynamics

$$r_{ui}(t) = \mu + b_u(t) + b_i(t) + q_i^T p_u(t)$$

- Other Factorization Methods
  - MMF (Nonnegative Matrix Factorization)
  - PCA (principal component analysis)
  - SVD (singular value decomposition)
  - VQ (vector quantization)



Summary

Pros

- Latent factors provide a dense vector to represent user taste or item flavor.
- User-Item Interactions (user behavior) are fully utilized, thus can be viewed as a case of Collaborative Filter.

Cons

- “Cold Start’: new user/item has no historical data to use, thus hard to derive latent factors.
- Require heavy computing effort and relative hard to be parallelized over distributed clusters.

Example 3

In this example, we will learn how to use TensorFlow for NMF (non-negative matrix factorization) and Spark for MF with ALS (alternating least squares).

See “Example 3 Matrix Factorization.ipynb”