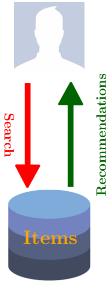
Recommendation Systems

- Recommenders: Motivation and Applications
- Problem Formulation and Evaluation
- Raw Averages based Recommendation
- ANOVA and Bayesian Filtering
- Content Based Filtering
- Collaborative Filtering
  - User-User Collaborative Filtering
  - Item-Item Collaborative Filtering
- Matrix Factorization

Imdad ullah Khan

Recommendation Systems



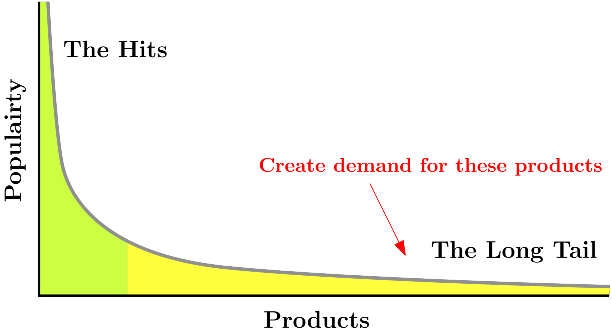
Products, news, friends, websites,movies, courses

The Web, they say, is leaving the era of search and entering one of discovery. What's the difference? Search is what you do when you're looking for something. Discovery is when something wonderful that you didn't know existed, or didn't know how to ask for, finds you.

J. O'Brien, Nov 20, 2006 The race to create a 'smart' Google

Recommendation Systems

- Retailers cannot shelve everything
- Online retailers and digital content providers have millions of products



Recommendation Systems

- Near zero-cost dissemination of information about products
- More choice necessitates better information filtering (customization)

Customization can be

- **Hand-Curated:** Chef's specials, editor's picks, favorites
- **Simple aggregates:** Top 10, Trending, Recent uploads
- **Customized to individual users:** Recommendation Systems

Recommendation Systems

Harvard Business Review 2017 **Perhaps the single most important algorithmic distinction between "born digital"' enterprises and legacy companies is not their people, data sets, or computational resources, but a clear real-time commitment to delivering accurate, actionable customer recommendations.**

- Netflix: 75% of movies watched are recommended [1]
  - "... personalization and recommendations save us more $\$1B$ per year" [2]
- Amazon: 35% of purchases on Amazon come from recommendations [1]
- Google News: recommendation genearate 38% more click-throughs [1]
- Airbnb: "Together, Search Ranking and Similar Listings drive 99% of our booking conversions" [3]
- Alibaba: For 11.11. mega sale, targeted personalized landing pages, resulted in 20% higher conversion rate from previous year [4]

[1] X. Amatriain, (2014) Machine Learning Summer School, CMU

[2] Gomez-Uribe & Hunt, Netflix Inc., (ACM Trans. on MIS 2015)

[3] Grbovic et.al [Airbnb Engineering & Data Science] (2018)

[4] InsideRetail.Asia (2017)

Problem Formulation and Evaluation

Recommendation Systems: Problem Formulation

- n users - $\{c_1, \ldots, c_n\}$ and m items - $\{p_1, \ldots, p_m\}$
- Utility Matrix U: $n \times m$ matrix row/column for each user/item
- $U(i,j)$: rating of user i for item j

	p_1	p_2	p_3						p_j									p_m
u_1	1		2	1		4			2		3	2		5				2
u_2		1				2		1			2			1				3
u_3		1	1	2				1							1		2	
				3		2			5			2			3	4		
		1			2									5				
u_i			3	2	1		4	5		1		3	1		2			1
			4													4		
			5			1										5		
		1		4							1	3		5		1	2	
u_n			3		1	1		2	1					4				5

$U(i,j)$ could be

- $0 - 5$ stars
- $\in [0,1]$
- $\in \{0,1\}$

Computational linear algebra problem of matrix completion

Recommendation Systems: Problem Formulation

- n users - $\{c_1, \ldots, c_n\}$ and m items - $\{p_1, \ldots, p_m\}$
- Utility Matrix U: $n \times m$ matrix row/column for each user/item
- $U(i,j)$: rating of user i for item j

	p_1	p_2	p_3						p_j									p_m
u_1	1		2	1		4			2		3	2		5				2
u_2		1				2		1			2			1				3
u_3		1	1	2				1							1		2	
				3		2			5			2			3	4		
		1			2									5				
u_i			3	2	1		4	5	$\boxed{?}$	1		3	1		2			1
			4													4		
			5			1										5		
		1		4							1	3		5		1	2	
u_n			3		1	1		2	1					4				5

$U(i,j)$ could be

- $0 - 5$ stars
- $\in [0,1]$
- $\in \{0,1\}$

If prediction for $U(i,j)$ is high recommend product j to user i

Recommendation Systems: Challenges

- Gather "known" ratings (populate matrix U)
- Extrapolate unknown ratings from known ones

  $\triangleright$ mainly interested in high ratings, Top k

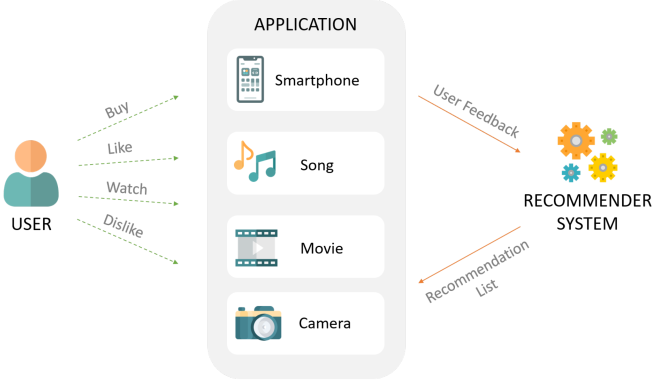
  - For each user c or a subset of users, find

$$R_c = \arg\max_p U(c, p)$$

  - R_c is the recommendation(s) for user c
- Evaluate extrapolation methods

Recommendation Systems: Gathering Data

- Explicitly survey users
- Implicitly learn ratings, e.g. purchase/suggestion to friend implies high rating
- Cold-start problem (new user, new product)



Recommendation Systems: Evaluation

	p_1	p_2	p_3						p_j									p_m
u_1	1		2	1		4			2		3	2		5				2
u_2		1				2		1			2			1				3
u_3		1	1	2				1							1		2	
				3		2			5			2			3	4		
		1			2									5				
u_i			3	2	1		4	5		1		3	1		2			1
			4													4		
			5			1										5		
		1		4							1	3		5		1	2	
u_n			3		1	1		2	1					4				5

Test Set

- Compare predictions $U'(i,j)$ with known (hidden) ratings
- Root-mean-squared-error $\text{RMSE} = \sqrt{\sum_{i,j \in \text{ Test Set}} (U(i,j) - U'(i,j))^2 / |\text{Test Set}|}$
- Spearman's rank correlation, Kindell's Tau

Spearman's rank correlation coefficient

- A measure of similarity between two variables based on rank
- Correlation between ranks of values of the variables

$$\rho_{xy} = \frac{\mathrm{cov}(rg_X, rg_Y)}{\sigma_{rg_X}\sigma_{rg_Y}}$$

X	3	3	3	4	4	4	5	5	5	0	0	1	1	2	2	2
Y	4	3	4	5	4	5	4	4	3	0	1	0	1	3	2	3
Z	1	0	1	2	1	2	1	1	0	3	4	3	4	0	5	0

Spearman's rank correlation coefficient

- A measure of similarity between two variables based on rank
- Correlation between ranks of values of the variables

$$\rho_{xy} = \frac{\mathrm{cov}(rg_X, rg_Y)}{\sigma_{rg_X}\sigma_{rg_Y}}$$

X	3	3	3	4	4	4	5	5	5	0	0	1	1	2	2	2
Y	4	3	4	5	4	5	4	4	3	0	1	0	1	3	2	3
Z	1	0	1	2	1	2	1	1	0	3	4	3	4	0	5	0
rg_X	8	9	10	11	12	13	14	15	16	1	2	3	4	5	6	7
rg_Y	10	6	11	15	12	16	13	14	7	1	3	2	4	8	5	9
rg_Z	4	5	6	7	8	9	10	11	12	13	15	14	16	1	2	3

$\rho_{XY} = 0.8$ $\rho_{XZ} = -0.1$ $\rho_{YZ} = -0.3$

Other goodness measures for recommenders

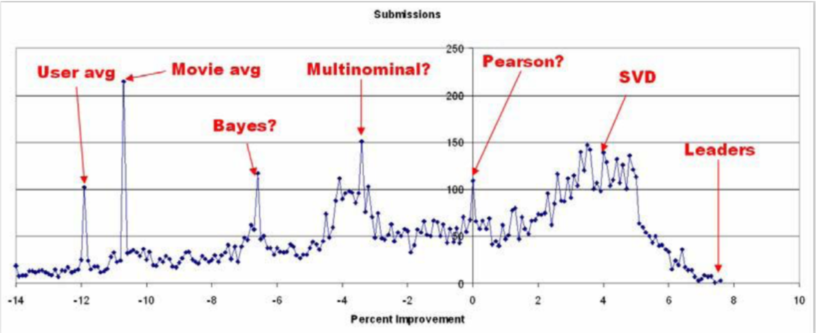
Other aspects of goodness of recommendations

- RMSE etc. might penalize methods for making errors on small rating
- Prediction Diversity: Customers are satisfied with intra-list diversity
- Persistence: re-show recommendations
- Privacy: User profiling and data gathering
- Demographics: important for customer satisfaction with recommendations
- Trust: Explaining how recommendations are found help
- Serendipity: How surprising recommendations are

Recommendation Systems: The Netflix Challenge

- In 2006, Netflix released a dataset movie rating dataset
- Training set: $\sim 1M$ ratings of the form $\langle$user, movie, date of grade, grade$\rangle$, 480,189 users, 17,770 movies
- qualifying data set: 2,817,131 triplets, $\langle$user, movie, date$\rangle$
- grade was known to jury only
- Competition to predict grades of qualifying set that improve accuracy by 10%
- Teams were informed of accuracy on 1,408,342 ratings (validation set)
- Jury used the test set of 1,408,789 ratings to determine winner

Netflix Challenge Methods

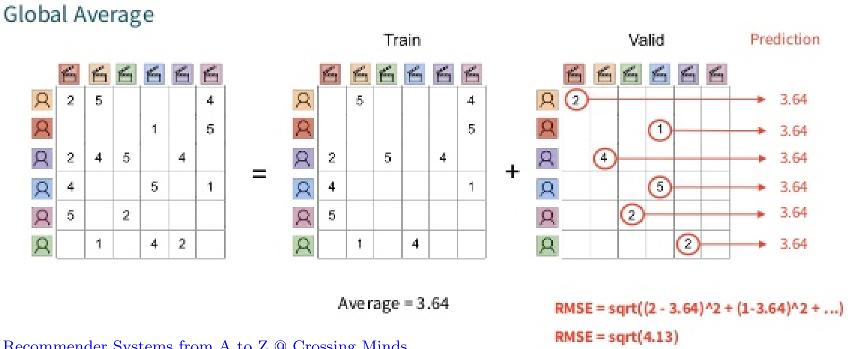


Recommendation using Averages and ANOVA

Recommendation Methods: Raw Averages

Predicting $U(i,j)$

- Assign average rating of all users for any item (MATRIX-AVERAGE)
- Assign average rating of all users for item j, (COLUMN-AVERAGE)
- Assign average rating of all items by user u, (ROW-AVERAGE)
- Mean is an unstable statistics (could use other measures of location)



Recommender Systems from A to Z @ Crossing Minds

Recommendation Methods: Raw Averages

Predicting $U(i,j)$

Let $\textsc{MoM}$ be the matrix mean (mean of means)

$$\textsc{MoM} := \frac{1}{nm} \sum_{c,p \in \text{ Train Set}} U(c,p), \quad \text{then}$$

$$U'(i,j) = \textsc{MoM}$$

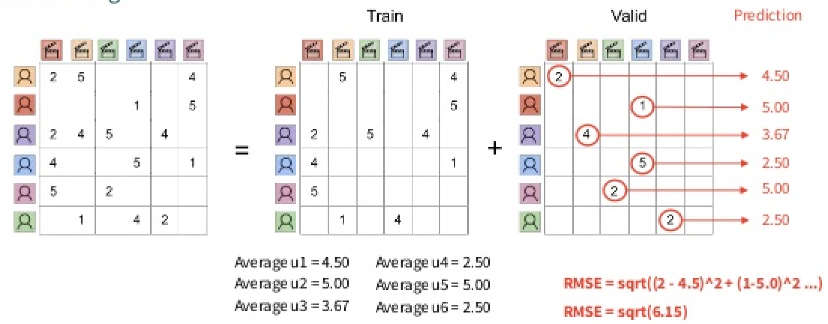
RMSE is just the standard-deviation of the data

Recommendation Methods: Raw Averages

Predicting $U(i,j)$

- Assign average rating of all users for any item (MATRIX-AVERAGE)
- Assign average rating of all users for item j, (COLUMN-AVERAGE)
- Assign average rating of all items by user u, (ROW-AVERAGE)
- Mean is an unstable statistics (could use other measures of location)

Recommender Systems from A to Z @ Crossing Minds



Recommendation Methods: ANOVA

Predicting $U(i,j)$

- Idea: Assign global average (matrix mean) $U(i,j) = \text{MoM}$
- Refinement 1: Product j maybe very (un) popular - highly (un)liked
  - Adjust for this bias
- Let dev_j be the average deviation of item j from MoM (+ve or -ve)
- $U(i,j) = \text{MoM} + dev_j$
- Refinement 2: User i may be very (non) pessimistic (critical)
  - Adjust for this bias too
- Let dev_i be the average deviation of user i from MoM

$$U(i,j) = \text{MoM} + dev_j + dev_i$$

Other methods are generally compared with this baseline

Recommendation Methods: Bayesian Method

Predicting $U(i, j)$

- Give the rating r, that is the most likely (in Bayesian sense)
- For $r \in \{0, 1, 2, 3, 4, 5\}$
- Let $P(r \mid j)$ be the conditional probability of rating r given item j

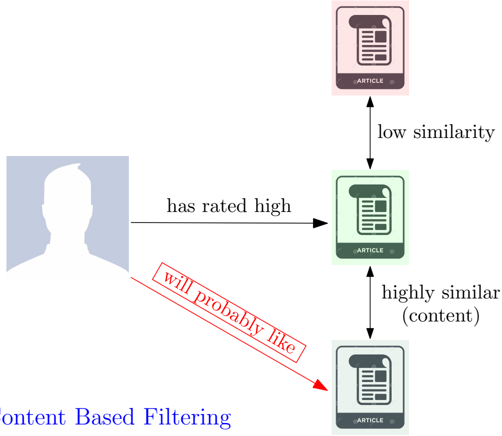
$$P(r \mid j) = \frac{P(j \mid r)\, P(r)}{P(j)}$$

- $P(j \mid r)$ is the conditional probability of item j given rating r
  - Estimate: The fraction of item j among all items rated with r
- $P(r)$ is the prior probability of rating r
- $P(j)$ is the prior probability of item j
- Con: Does not take into account user i at all

Content Based Filtering

Recommendation Methods: Content-based

Content Based Filtering



Content Based Filtering

Recommendation Methods: Content-based

If j is similar to the "taste" of i, then predict $U(i,j)$ high

1. Build Item Profile (based on content) e.g.
   - movies: vector of genre, director, budget, cast, plot, language
   - books, blogs, website, news items: TF-IDF vector, author, topic
2. Build User Profile
   - A vector with the same coordinates as item profile
   - kind of "an average item" that the user likes $\triangleright$ the taste of user
   - Weighted (by ratings) average of the item profiles that the user has rated
3. $U'(i,j) \propto$ (cosine) similarity between item j's and user i's profiles

Recommendation Methods: Content-based

If j is similar to the "taste" of i, then predict $U(i, j)$ high

Pros

- No need of other users' information
- No cold-start or sparsity problem w.r.t items
- Unique taste of user is captured
- Able to provide explanation (by listing contents' features)

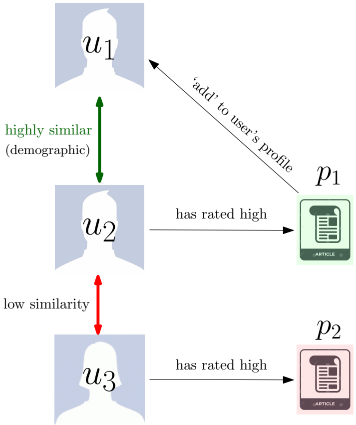
Cons

- Building profile is hard, finding relevant features is hard
- Cold start problem w.r.t users
- User profile is heuristic
- Overspecialization-never recommends items outside user profile
- Does not cater for multiple interests of a user
- Does not utilize judgment of other users

Recommendation Methods: Content-based

If j is similar to the "taste" of i, then predict $U(i,j)$ high

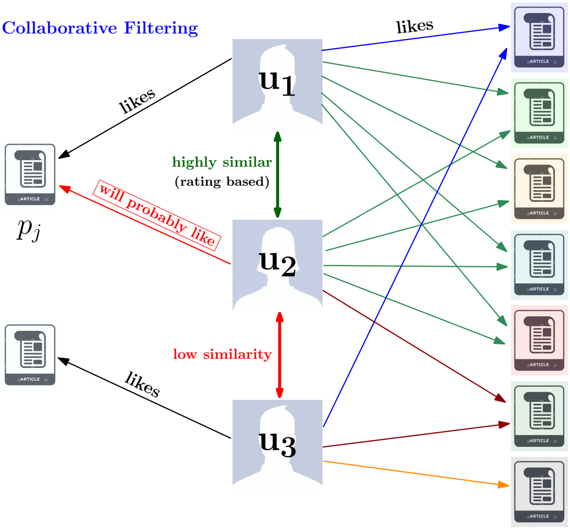
Can take into account other (similar) users judgments as follows.
Somewhat cater for the cold start problem



Collaborative Filtering

User-User Collaborative Filtering

Collaboratively filter (personalize) ratings using only the rating matrix U



User-User Collaborative Filtering

Collaboratively filter (personalize) ratings using only the rating matrix U

- Find the set N of users with similar ratings as of i
  - Find the top k similar rows to the ith row
- Estimate $U(i,j)$ as an "average" of $U(a,j)$'s for $a \in N$
- i has similar 'taste' to $a \in N \implies U(i,j)$ similar to $U(a,j)$

$$U'(i,j) = \frac{\sum_{a\in N} sim(a,i) \times U(a,j)}{\sum_{a\in N} sim(a,i)}$$

- Use cosine similarity to get N ($\because$ interested in similar not high ratings)
- Alternatively, Spearman's rank correlation, Kindell Tau

Item-Item Collaborative Filtering

Collaboratively filter (personalize) ratings using only the rating matrix U

- Find the set W of items similarly rated as j
  - Find the top k similar columns to the jth row
- Estimate $U(i,j)$ as an "average" of $U(i,p)$'s for $p \in W$

$$U'(i,j) = \frac{\sum_{p \in W} U(i,p) \times sim(j,p)}{\sum_{p \in W} sim(j,p)}$$

- Better result by item-item collaborative filtering
  - Because items are easier to model
  - has less complexity than users

Collaborative Filtering

- Works for any kind of items, unlike content based that requires outside knowledge for profiles
- Cold-Start problem: need enough users to find a match
- Sparsity: Hard to find users that have rated the same items
- First rater: cannot recommend items that are not previously rated (new or esoteric items)
- Popularity bias: Does not cater for unique taste of a user, tends to recommend popular items
- Computational Complexity

Generally, hybrid methods (ensemble) are used. Combine predictions of many recommender system (average, weighted average, regression)

Recommendation using Matrix Factorization

Matrix Completion Problem

Recall the recommendation system problem is the problem of matrix completion in computational linear algebra

Given a rating matrix R – users ratings for items, predict $R(i, j)$

	p_1	p_2	p_3						p_j									p_m
u_1	1		2	1		4			2		3	2		5				2
u_2		1				2		1			2			1				3
u_3		1	1	2				1							1		2	
				3		2			5			2			3	4		
		1			2									5				
u_i			3	2	1		4	5	$\boxed{?}$	1		3	1		2			1
			4													4		
			5			1										5		
		1		4							1	3		5		1	2	
u_n			3		1	1		2	1					4				5

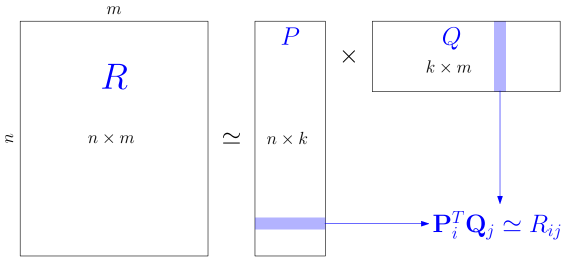
Matrix Factorization

- Given $n \times m$ matrix R For $k \ll m, n$, Find
- $n \times k$ matrix P and $k \times m$ matrix Q such that

$$R = PQ$$

Generally, for very small k, we seek

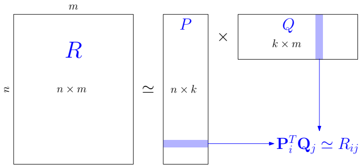
$$R \simeq PQ$$



Matrix Factorization

- Given $n \times m$ matrix R For $k \ll m, n$, Find
- $n \times k$ matrix P and $k \times m$ matrix Q such that

$$R \simeq PQ$$



This is a classic optimization problem can be solved as

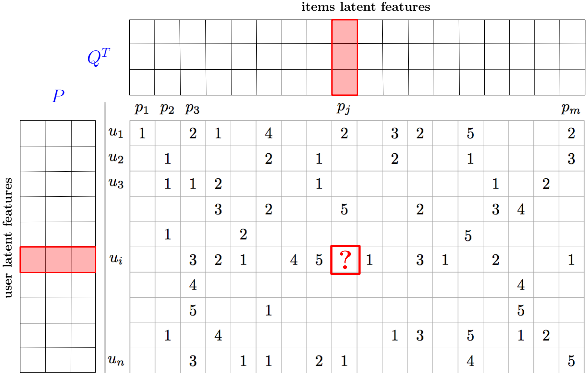
$$\min_{\substack{P \in \mathbb{R}^{n\times k} \\ Q \in \mathbb{R}^{m\times k}}} \sum_{(i,j)} \left(R_{ij} - P_i Q_j^T\right)^2 + \underbrace{\lambda \left(\|P\|_F^2 + \|Q\|_F^2\right)}_{\substack{\text{regularization term} \\ \text{avoids overfitting}}}$$

Later we will discuss low rank approximation (SVD) to solve this problem

Matrix Factorization for Recommenders

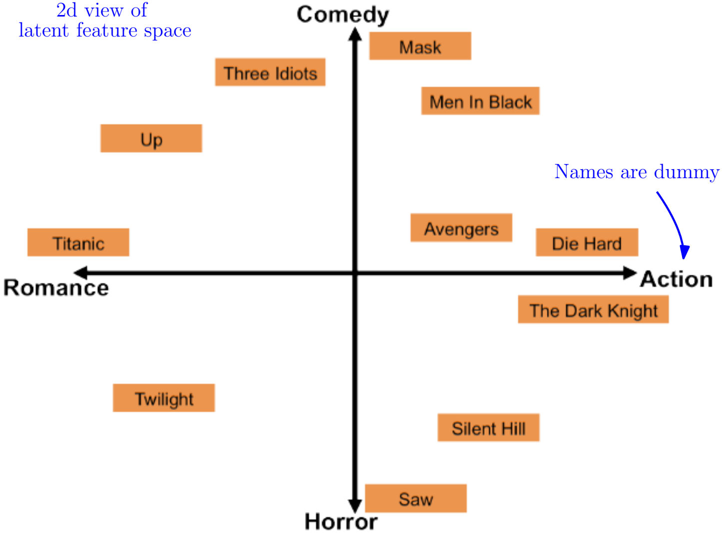
Matrix Factorization for Recommenders $\quad R \simeq PQ$

- P : k-dim representation of users in a latent feature space $\mathbb{R}^k$
- Q : k-dim representation of items latent feature space
- $P_i Q_j^T$: interaction between user i and item j – approximation of R_{ij}



Matrix Factorization for Recommenders

2d view of
latent feature space



Names are dummy

diagram adapted from Cho-Jui Hsieh @ UCLA

Users and movies mapped to latent feature space

Points are coordinates of users and movies (2-dim P and Q)

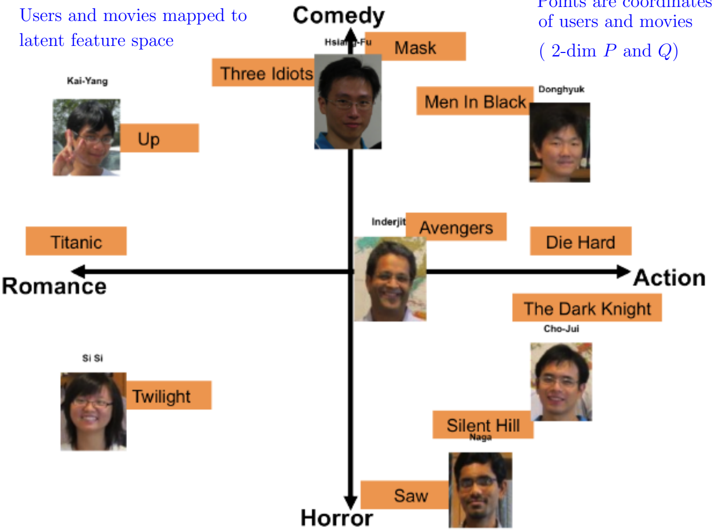


diagram adapted from Cho-Jui Hsieh @ UCLA