Recommendation Systems
The Netflix prize

Antonio Ercole De Luca

Università di Pisa

Recommendation Systems Talk Outline:

- Problem Statement
- Daily life examples
- Approaches
  - Unpersonalized
  - Personalized

Recommendation Systems

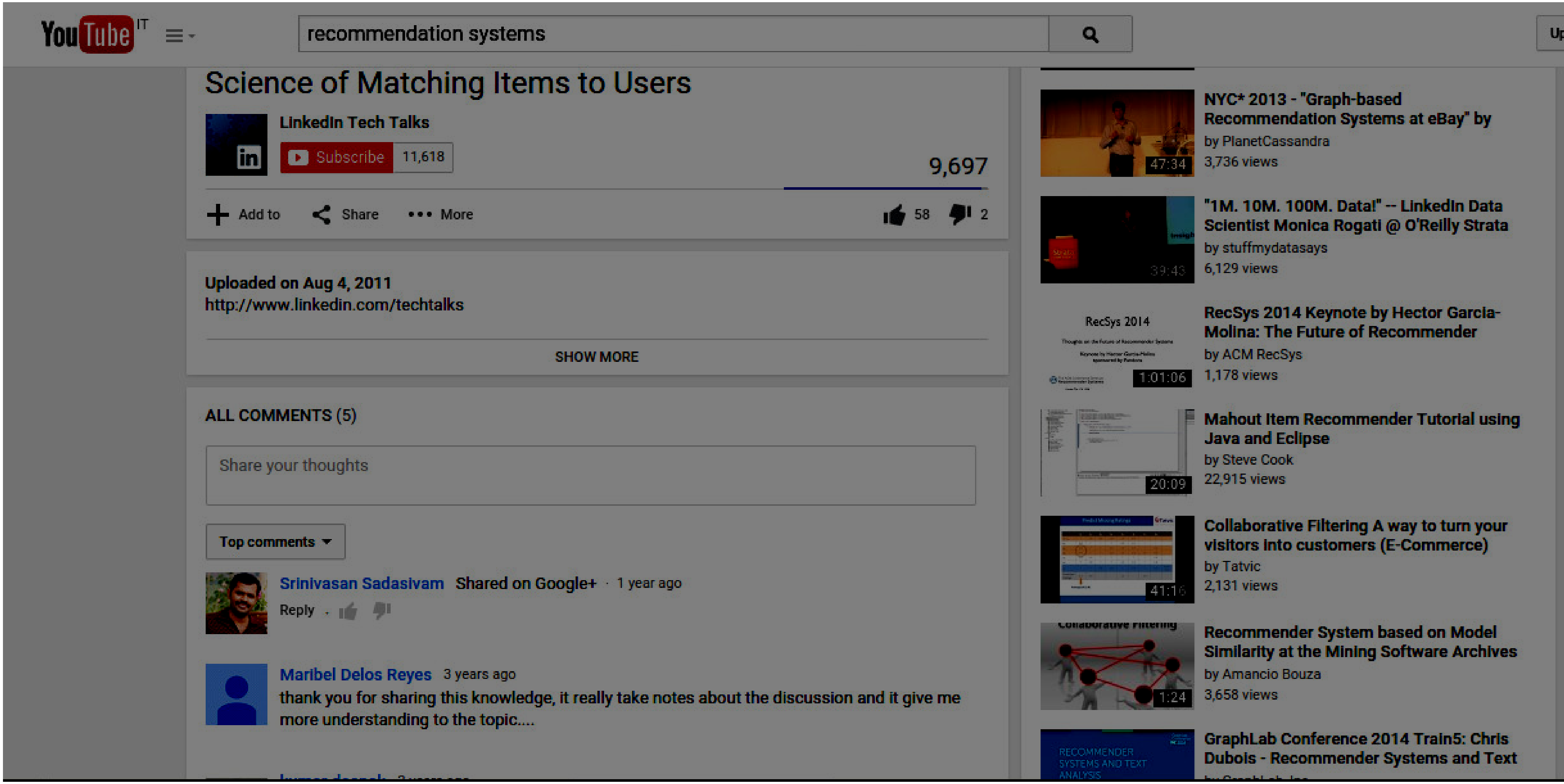
Personalized approach:

- Content Based
- Collaborative Filtering
  - Neighborhood approach
  - Latent Factor Models
  - Restricted Boltzmann machines
  - Ensemble Methods

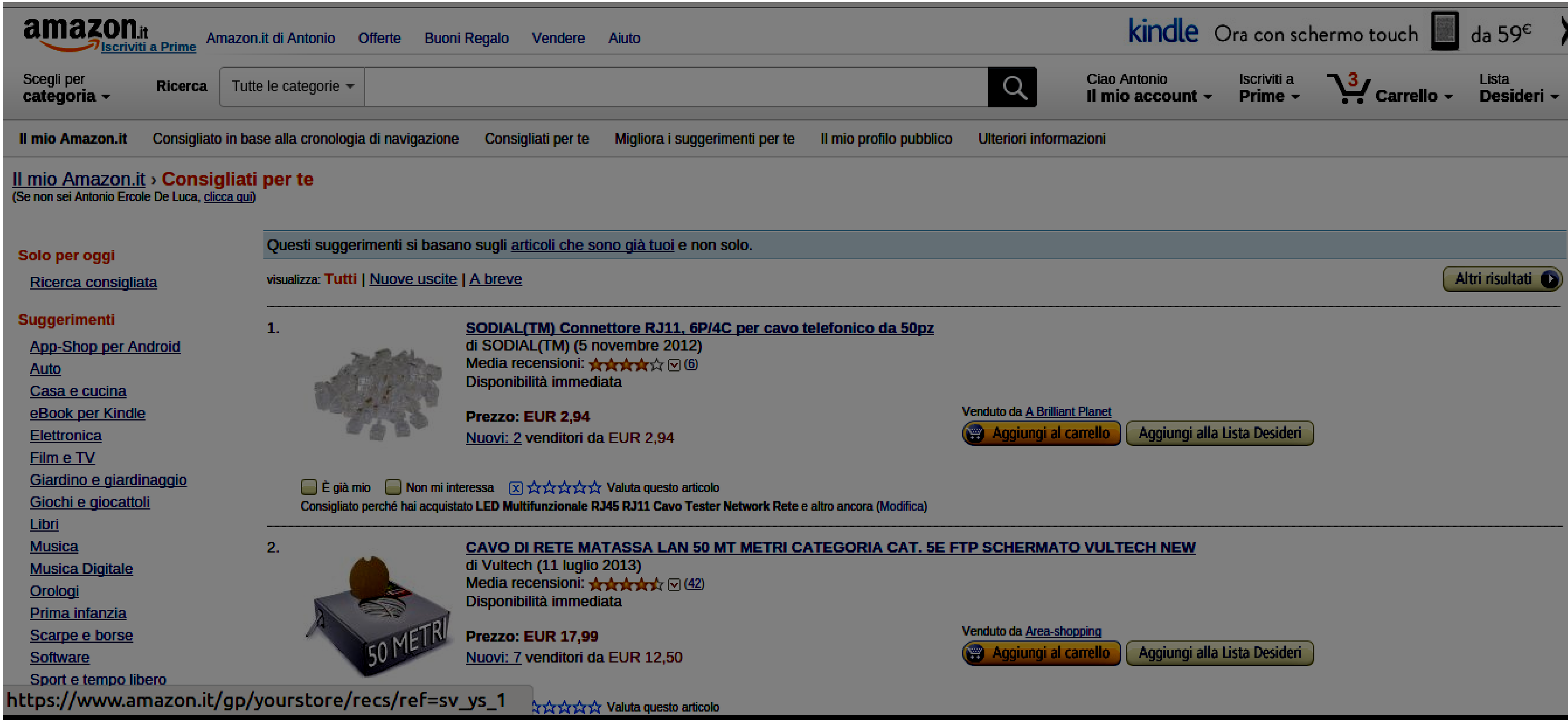
Problem Statement

- Predicting users responses to options
- Suggest new products to customer

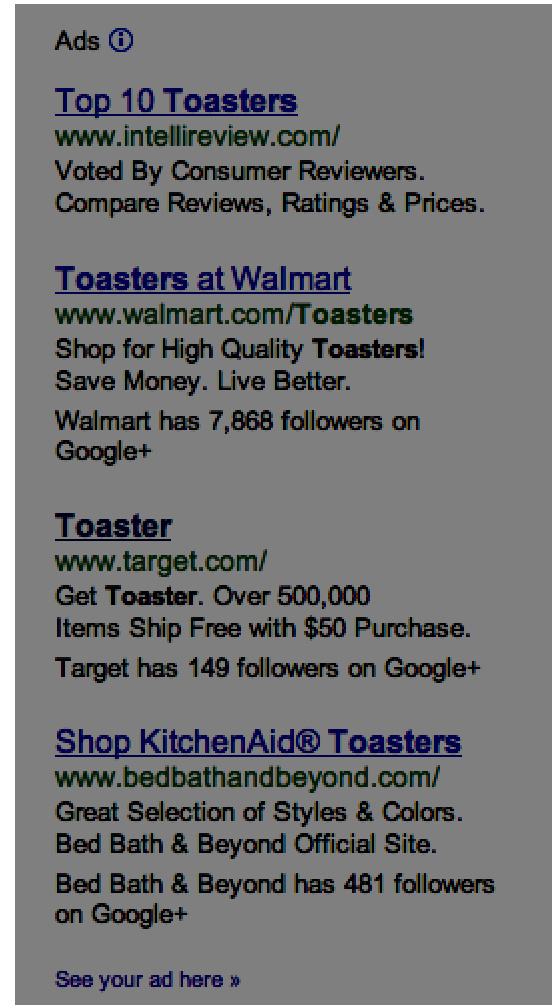
Example - Youtube



Examples - Amazon



Examples - Google Ads



Why This?

- Web based company needs to face costs:
  - Data Farms / Servers
  - Employees
- Business model based on advertisement are yet consolidated
- Is a Winner to Winner approach
  - Customers finds interesting products
  - Companies finds customers

How is accomplished?

- Technical details in a few minutes
- First:
  - Introduction to the Netflix prize

Netflix Prize

- Open Competition - 2006
- Training data set of
  - ≈ 100 million ratings that
  - ≈ 480 thousand users gave to
  - ≈ 17 thousand movies.

Netflix Prize (2)

- 1 Million Dollars prize
- To whom outperformed 10%
- Netflix's algorithm
- Gave a big boost to research on this field
- Won in 2009

Netflix Dataset

- Is a Matrix
- A row - for each user
- A column - for each movie
- Ratings are entries in the matrix
- values between 1...5
- Contains Null ratings

Matrix Example

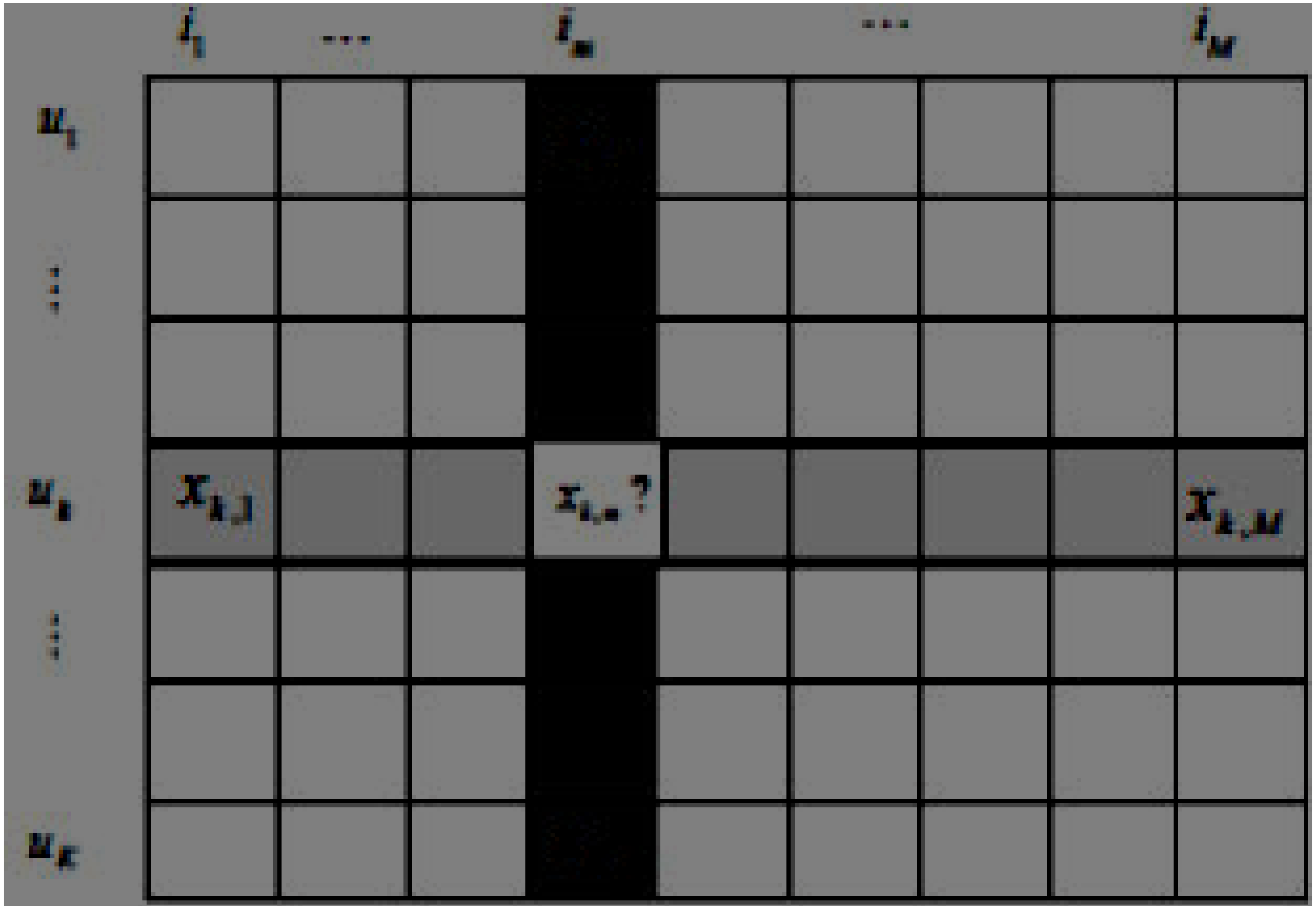


Figure 2.4 User x Item Matrix

Source (Wang, P. and J.T. 2005)

Objective

- Estimate null ratings on a blinded test set
- Evaluation Measure:
    - Root Mean Square Error

$$\mathrm{RMSE} = \sqrt{\sum \frac{(y_{pred} - y_{ref})^2}{N}}$$

Root mean square error

- Netflix's algorithm is called:
- Cinematch
- Scores an RMSE of 0.9514

Recommendation Systems

Approaches:

Recommendation Systems Approaches:

- Unpersonalized
  - Suggesting most sold items
- Personalized
  - Suggesting most interesting items for the specific user

Recommended Systems
Non Personalized

- Suggest most sold items
- Scoring items independently from the user history
- Cheap = >
  - Needs only the summary of data
  - No need for BIG computations
- Fast = >
  - Leverages Cached results

Non Personalized (2) Netflix

- Examples in Netflix:
- Average Ratings:
  - Averaging through all movies
  - Averaging through a single movie

Recommended Systems Non Personalized (3)

- On the probe* set:

Effect	RMSE
Overall mean	1.1296
Movie effect	1.0527
Netflix's Cinematch *2	0.9474

* Scalable Collaborative Filtering with Jointly Derived Neighborhood In
Robert M. Bell and Yehuda Koren

*2 http://www.netflixprize.com/faq#probe

Recommended Systems Non Personalized (4)

- Used as baseline of more sophisticated approaches
- Inside the Data Normalization preprocessing phase

Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model
Yehuda Kore

Recommended Systems Non Personalized (5)

- Data Normalization preprocessing phase
- Remove from every rating the average rating of:
- all movies
- the single movie
- the single user

Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model
Yehuda Kore

Recommendation Systems

Personalized Approaches:

Recommendation Systems
<u>Personalized</u> Approaches:

Using user's history to suggest items

- Content Based
  - Exploiting domain specific additional resources
- Collaborative Filtering
  - Exploiting crowd behaviors

Recommendation Systems Content Based

- Using domain specific attributes for items
- Products:
  - Color, size, features, uses, categories, etc
- Movie:
  - Genre, Director, Actors, Year, Language, etc
- Documents:
  - Bag-of-word, tf-idf, Identified

Recommendation Systems Content Based (2)

- Based on user's history identify his interested attributes and tastes
- Identify new items <u>similar</u> to his tastes

Recommendation Systems Content Based (3)

- It needs a notion of similarity
  - Euclidean distance
  - Jaccard similarity
  - Cosine Similarity
  - Etc

Recommendation Systems Content Based (4)

- It needs similarity based data structures for efficient retrievals
  - R*-trees
  - Kd-trees
  - Vp-trees

- Similarity function must satisfies the triangle inequality !

Recommendation Systems Content Based (5)

- Requires gathering external informations
- Suffers Cold Start problem

Recommendation Systems
Personalized Approaches:

Using user's history to suggest items

- Content Based
  - Exploiting domain specific additional resources
- Collaborative Filtering
  - Exploiting crowd behaviors

Recommendation Systems
Collaborative Filtering:

Uses only past user behaviors!

Main Approaches:

- Neighborhood Models
- Latent Factor Models
- Restricted Boltzmann Machines
- Ensemble Methods

Recommendation Systems Neighborhood Models:

Rationale:

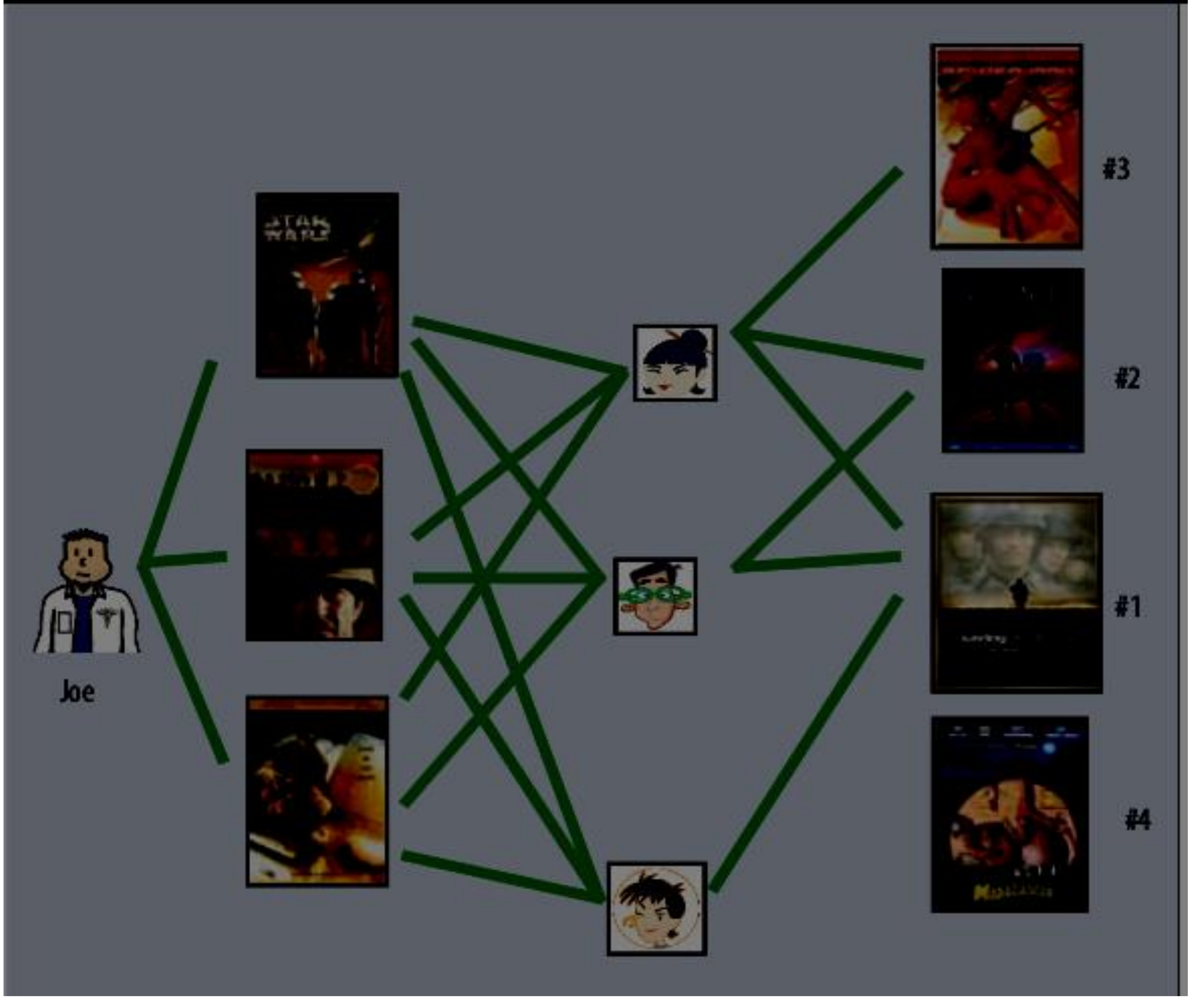
- Similar users have the same tastes
- Similar items are rated similarly by the same user
- Basically two approaches, use relationships between pairs of:
  - Users
  - Items

Neighborhood Models
User-User Relationships:

- Identify like-minded users who complement each others ratings

1. Selecting the like-minded users (neighbors)
2. Use interpolation weights to differentiate between different users
3. Compute all the ratings of the unseen items as a weighted average of the ratings of the neighbors

Neighborhood Models User-User Example:

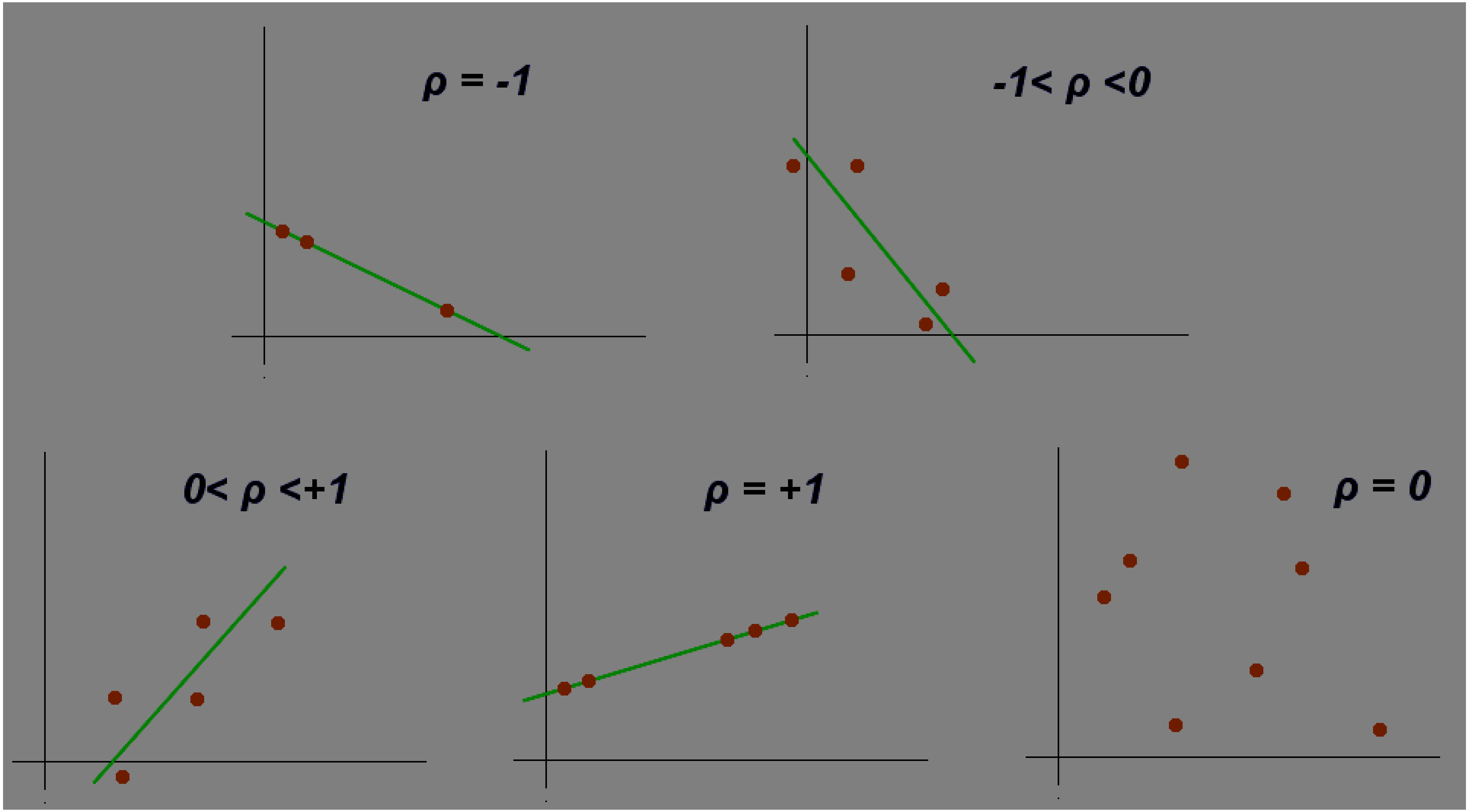


MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS
Yehuda Koren, Robert Bell and Chris Volinsky

Neighborhood Models
User-User similarity:

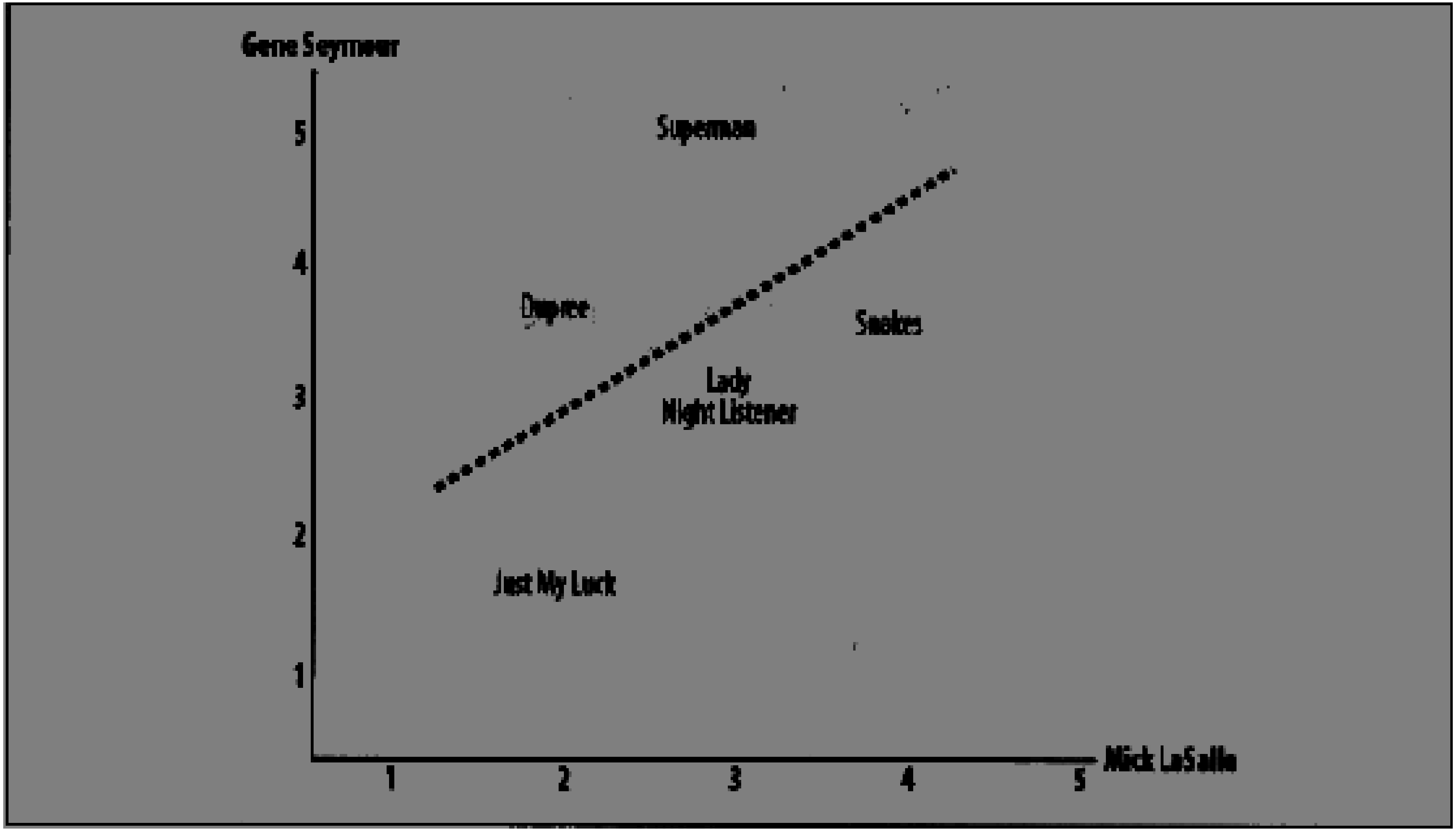
- How compute User-User Similarity?
  - Euclidean Distance
  - Pearson correlation coefficient
- Pearson correlation coefficient
  - Measures how two users agrees on their ratings
  - It's between -1 and 1

Pearson correlation coefficient:



http://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient

Pearson correlation coefficient: (2)



Programming Collective Intelligence Building Smart Web 2.0 Applications
Toby Segaran

Pearson correlation coefficient: (3)

- Predict unseen ratings as a weighted votes over u's neighbors' ratings of the item i

  $N(u;i)$ Set of similar users

  s_{uv}

$$r_{ui} \leftarrow \frac{\sum_{v \in \mathrm{N}(u;i)} s_{uv} r_{vi}}{\sum_{v \in \mathrm{N}(u;i)} s_{uv}}$$

Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights
Robert M. Bell

Neighborhood Models User-User Relationships: (2)

- Weighted version of K-nearest neighbors algorithm
- With k chosen by cross-validation

Neighborhood Models
User-User Relationships: (3)

- Pearson Correlation is not a metric!
- The algorithm do not scale well with respect to the numbers of users!

Neighborhood Models
Item-Item relationship:

- The similarity is computed between items
- The rating of a user is computed as the weighted average of his ratings about the k most similar items

Neighborhood Models Item-Item relationship: (2)

- The k-nearest neighbors are precomputed and stored in memory for every item
- Netflix problem has less movies than users
- => Fastest Solution
- This solution is more suitable for the Netflix problem
- => Better solution (improved accuracy)

Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights
Robert M. Bell

Recommendation Systems Neighborhood Models: (2)

- They lack in formal models
- Why should do they work?

Recommendation Systems
Latent Factor Models:

Analogy with the content based approach

Latent Factor Models - Content based - Analogy

In the content based :

- For every item we have a set of attributes that it embraces
- For every user we have values that identifies his preferences for the attributes
- Their weighted sum identify how a movie reflects the tastes of a user

Latent Factor Models - Content based - Analogy (2)

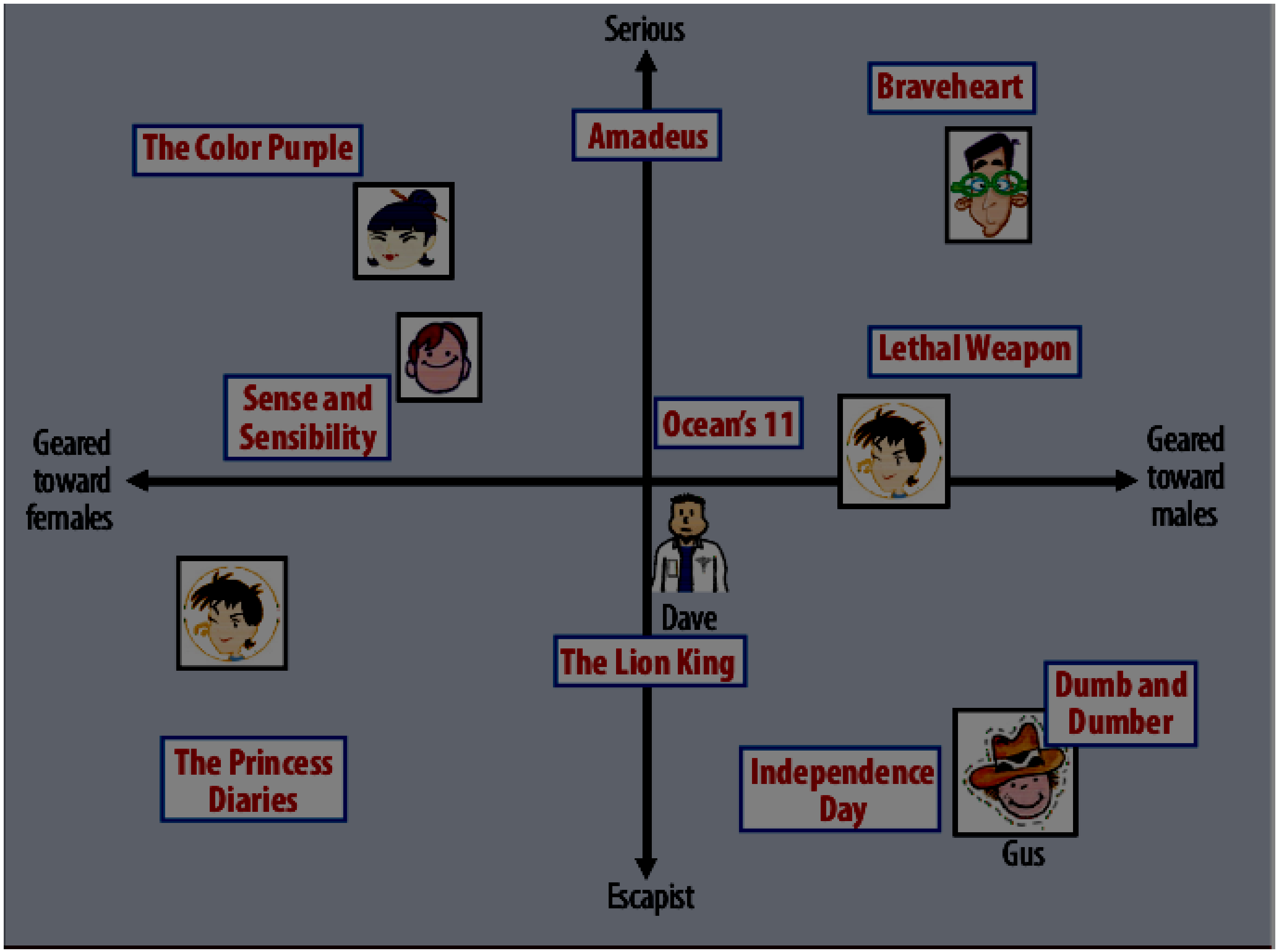
In the content based :

- We need to identify for every movie their attributes

Latent Factor Models:

supposes the presence of these factors without knowing them

Latent Factor Models (2)



Latent Factor Models
Matrix Decomposition

- How was it done?
- We search for two matrix:

- Q and P

Latent Factor Models
Matrix Decomposition (2)

Q is a U x D matrix, where

- U is the number of users
- D is the number of latent dimensions

Latent Factor Models
Matrix Decomposition (3)

P is a D x I matrix, where

- I is the number of items (movies)
- D is the number of latent dimensions

Latent Factor Models Matrix Decomposition (4)

Such that: $R' = QP$

So every entry of R' is:

$$r_{ui}' = q_i^T p_u$$

Latent Factor Models Matrix Decomposition (5)

We search Q and P such that

R' is similar as possible to R (the ratings matrix) for the known ratings

So we want to minimize this function:

$$min \sum \left(r_{ui} - q_i^T p_u\right)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

Latent Factor Models Matrix Decomposition (6)

- Stochastic Gradient Descent Algorithm
- it modifies the parameters to the opposite direction of the gradient of a magnitude proportional to the error

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

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

Matrix Decomposition Example

$$\begin{bmatrix} 5 & 2 & 4 & 4 & 3 \\ 3 & 1 & 2 & 4 & 1 \\ 2 & & 3 & 1 & 4 \\ 2 & 5 & 4 & 3 & 5 \\ 4 & 4 & 5 & 4 & \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \\ u_{41} & u_{42} \\ u_{51} & u_{52} \end{bmatrix} \times \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \end{bmatrix}$$

$$R' = QP$$

Matrix Decomposition Example (2)

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.10: Matrices U and V with all entries 1

Matrix Decomposition Example (3)

$$\begin{bmatrix} x & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} x+1 & x+1 & x+1 & x+1 & x+1 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.11: Making u_{11} a variable

Matrix Decomposition Example (4)

$$\begin{bmatrix} 2.6 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 3.6 & 3.6 & 3.6 & 3.6 & 3.6 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.12: The best value for u_{11} is found to be 2.6

Matrix Decomposition Example (5)

$$\begin{bmatrix} 2.6 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} y & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 2.6y+1 & 3.6 & 3.6 & 3.6 & 3.6 \\ y+1 & 2 & 2 & 2 & 2 \\ y+1 & 2 & 2 & 2 & 2 \\ y+1 & 2 & 2 & 2 & 2 \\ y+1 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.13: v_{11} becomes a variable y

Matrix Decomposition Example (6)

$$\begin{bmatrix} 2.6 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1.617 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 5.204 & 3.6 & 3.6 & 3.6 & 3.6 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.617 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.14: Replace y by 1.617

Matrix Decomposition Example (7)

$$\begin{bmatrix} 2.6 & 1 \\ 1 & 1 \\ z & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1.617 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 5.204 & 3.6 & 3.6 & 3.6 & 3.6 \\ 2.617 & 2 & 2 & 2 & 2 \\ 1.617z+1 & z+1 & z+1 & z+1 & z+1 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.617 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.15: u_{31} becomes a variable z

Matrix Decomposition Example (8)

$$\begin{bmatrix} 2.6 & 1 \\ 1 & 1 \\ 1.178 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1.617 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 5.204 & 3.6 & 3.6 & 3.6 & 3.6 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.905 & 2.178 & 2.178 & 2.178 & 2.178 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.617 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Figure 9.16: Replace z by 1.178

Matrix Decomposition Example (9)

$$\begin{bmatrix} 5 & 2 & 4 & 4 & 3 \\ 3 & 1 & 2 & 4 & 1 \\ 2 & & 3 & 1 & 4 \\ 2 & 5 & 4 & 3 & 5 \\ 4 & 4 & 5 & 4 & \end{bmatrix}$$

$$\begin{bmatrix} 5.204 & 3.6 & 3.6 & 3.6 & 3.6 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.905 & 2.178 & 2.178 & 2.178 & 2.178 \\ 2.617 & 2 & 2 & 2 & 2 \\ 2.617 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Matrix Decomposition

- Gradient Descent modify the solution only of an epsilon toward the local optimum
- For searching global optimum it needs to be executed many times with different initial random values of the matrices
- The number of Latent dimension is determined by cross-validation

Latent Factor Models (8)

- The model can be complicated more with:
  - Variables that models different biases
    - User bias
    - Movie bias
  - Variables as Time dependent Random Variables

Netflix Prize

- With those techniques, in 2008, BellKor & Big Chaos group outperformed netflix algorithm of 9.46%

Netflix Prize
End of the Story

- In 2009, the team BellKor's Pragmatic Chaos achieved a 10.05% improvement over Cinematch
- It used an ensemble of different models trained with different parameters
  - KNN
  - Restricted Boltzmann machines
  - Matrix Factorization
  - Temporal Effects

Thank You!

Questions?