DePaul University

# DSC478: Programming Machine Learning Applications

**Roselyne Tchoua**

**rtchoua@depaul.edu**

**School of Computing, CDM, DePaul University**

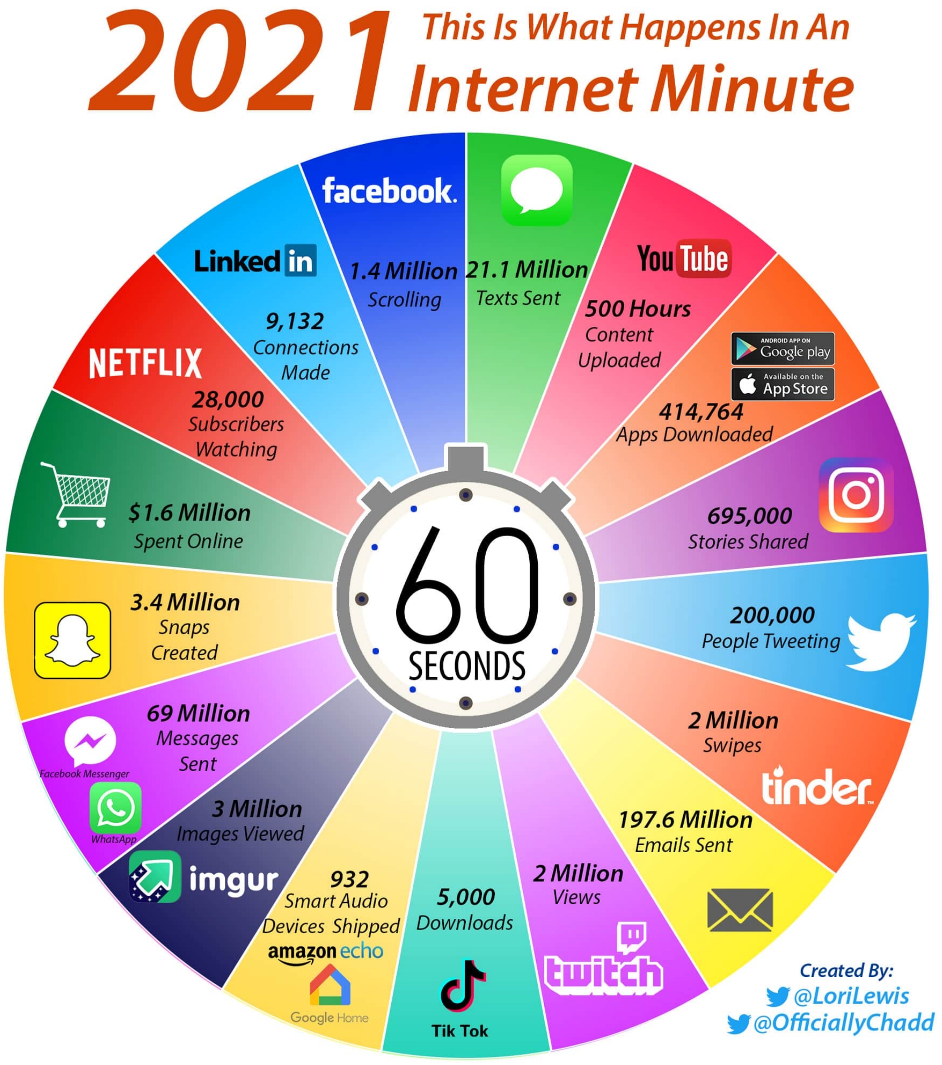
# Predictive User Modeling for Personalization

- The Problem: Dynamically serve customized content (ads, products, deals, recommendations, etc.) to users based on their profiles, preferences, or expected needs

- Example: Recommender systems: Personalized information filtering systems that present items (films, television, video, music, books, news, restaurants, images, web pages, etc.) that are likely to be of interest to a given user

## Why we need it?

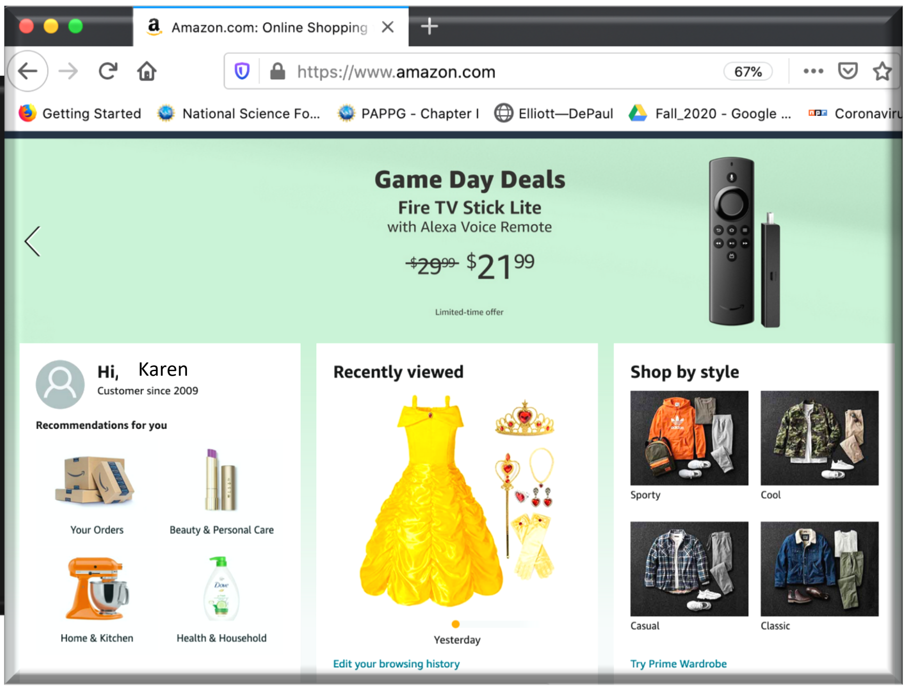
Information spaces are becoming more complex for user to navigate (video/audio streaming, social networks, blogs, ….)



# Why we need it?

For businesses: grow customer loyalty / increase sales

- Amazon → 35% of sales from recommendation; increasing fast!
- Netflix → 40%+ of movie selections from recommendation
- Facebook → 90% of user interactions via personalized feeds

# Predictive User Modeling for Personalization in the news!

https://www.npr.org/2021/10/05/1043377310/facebook-whistleblower-frances-haugen-congress
Former Facebook data scientist Frances Haugen speaks during a hearing of the Senate Commerce, Science and Transportation Subcommittee on Consumer Protection, Product Safety and Data Security on Capitol Hill on Tuesday.
Alex Brandon/AP

Based on our “likes”, “comments”, time spent/stopped on an item, who we follow, where we get our news from… all these can be used to get features.

# Ethical concerns?!

# Common Approaches

- Collaborative Filtering
  - Give recommendations to a user based on preferences of **"similar" users**
  - Preferences on items may be explicit or implicit – **history/interaction**
  - Includes recommendation based on **social / collaborative** content
- Content-Based Filtering
  - Give recommendations to a user based on items with **"similar" content** in the user's profile
- Rule-Based (Knowledge-Based) Filtering
  - Provide recommendations to users based on predefined (or learned) rules
  - age(x, 25-35) and income(x, 70-100K) and children (x, >=3) → recommend(x, Minivan)
- Hybrid Approaches

# The Recommendation Task

- Basic formulation as a prediction problem

> Given a profile $\boldsymbol{P_u}$ for a user $\boldsymbol{u}$, and a target item $\boldsymbol{i_t}$, predict the preference score of user $\boldsymbol{u}$ on item $\boldsymbol{i_t}$

- Typically, the profile $P_u$ contains preference scores by $u$ on some other items, $\{i_1, \ldots, i_k\}$ different from $i_t$
  - preference scores on $i_1, \ldots, i_k$ may have been obtained explicitly (e.g., movie ratings) or implicitly (e.g., time spent on a product page or a news article)

# Recommendation as Rating Prediction

- Two types of entities: *Users* and *Items*
- Utility of item *i* for user *u* is represented by some rating *r* (where $r \in Rating$)
- Each user typically rates a *subset* of items
- Recommender system then tries to estimate/predict the unknown ratings, i.e., to extrapolate rating function *Rec* based on the known ratings:
  - *Rec*: *Users* × *Items* → *Rating*
  - i.e., two-dimensional recommendation framework
- The recommendations to each user are made by offering his/her highest-rated items

Using a user by items matrix containing users' ratings of items.

# COLLABORATIVE FILTERING (CF)

# Collaborative Recommender Systems

Predictions for unseen (target) items are computed based the other users with similar interest scores on items in user *u*'s profile

- i.e., users with similar tastes (aka "nearest neighbors")
- requires computing correlations between user u and other users according to interest scores or ratings
- *k*-nearest-neighbor (*k*nn) strategy

| | Star Wars | Jurassic Park | Terminator 2 | Indep. Day |
|---|---|---|---|---|
| **Sally** | 7 | 6 | 3 | 7 |
| **Bob** | 7 | 4 | 4 | 6 |
| **Chris** | 3 | 7 | 7 | 2 |
| **Lynn** | 4 | 4 | 6 | 2 |
| **Karen** | 7 | 4 | 3 | ? |

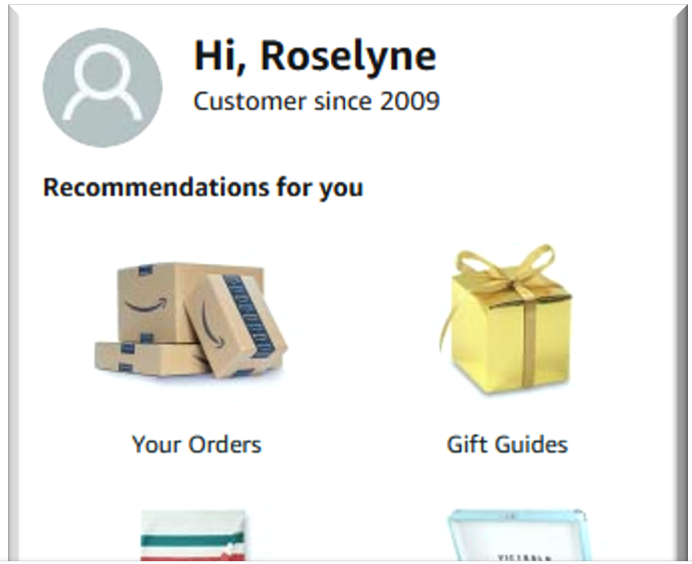
**Can we predict Karen's rating on the unseen item Independence Day?**

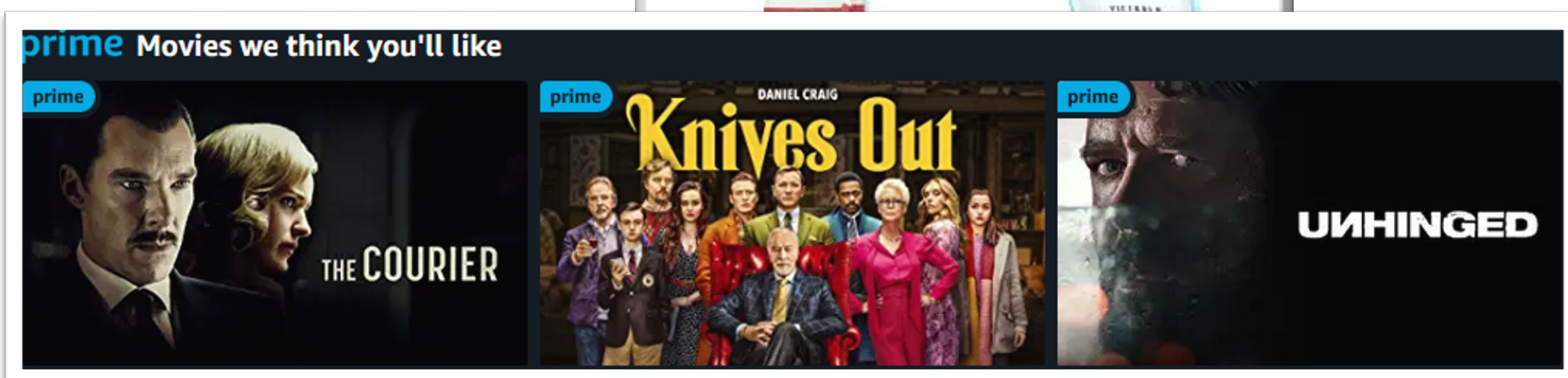
# Collaborative Recommender Systems

…recommendations are based on **items you own** and more…

Knowing that they have:

- "My stuff"
- "Purchases &Rentals"
- "Watch list"



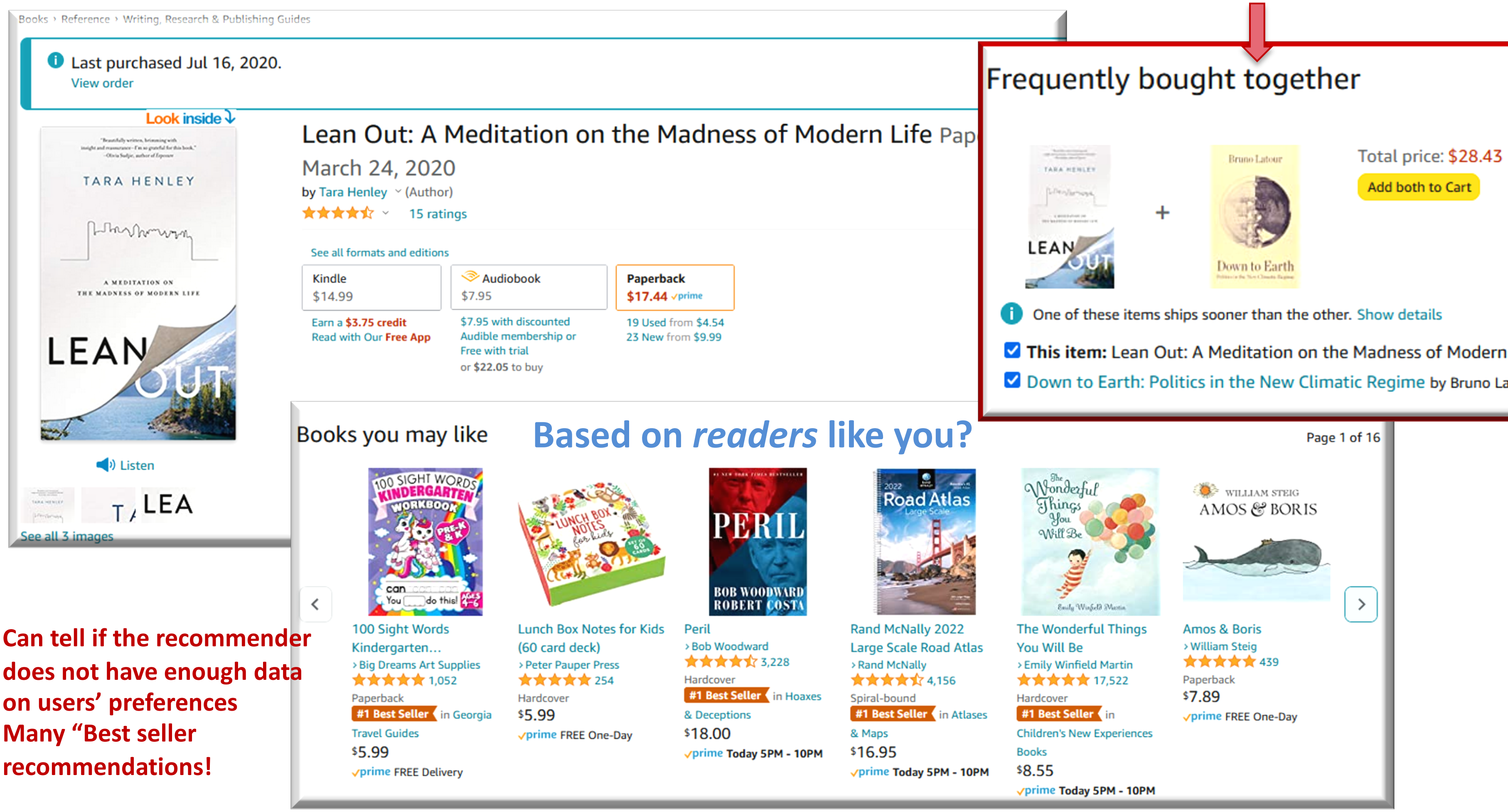
# Collaborative Recommender Systems

Products related to this item
Related how?
**People “like you” bought them?**

Different from
“**Frequently bought together**” (Market Basket Analysis or Association Rule Mining (ARM))
“**Similar description as**…” (Content-based filtering)

# Collaborative Recommender Systems

Not collaborative!

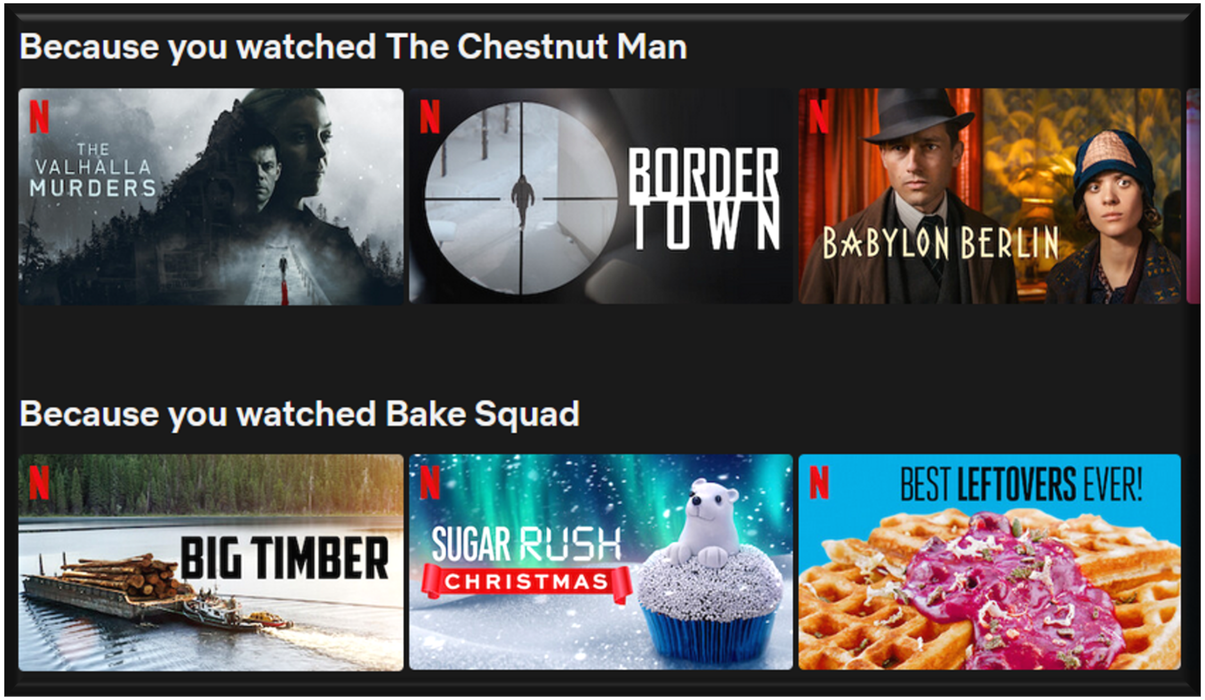



Based on *readers* like you?

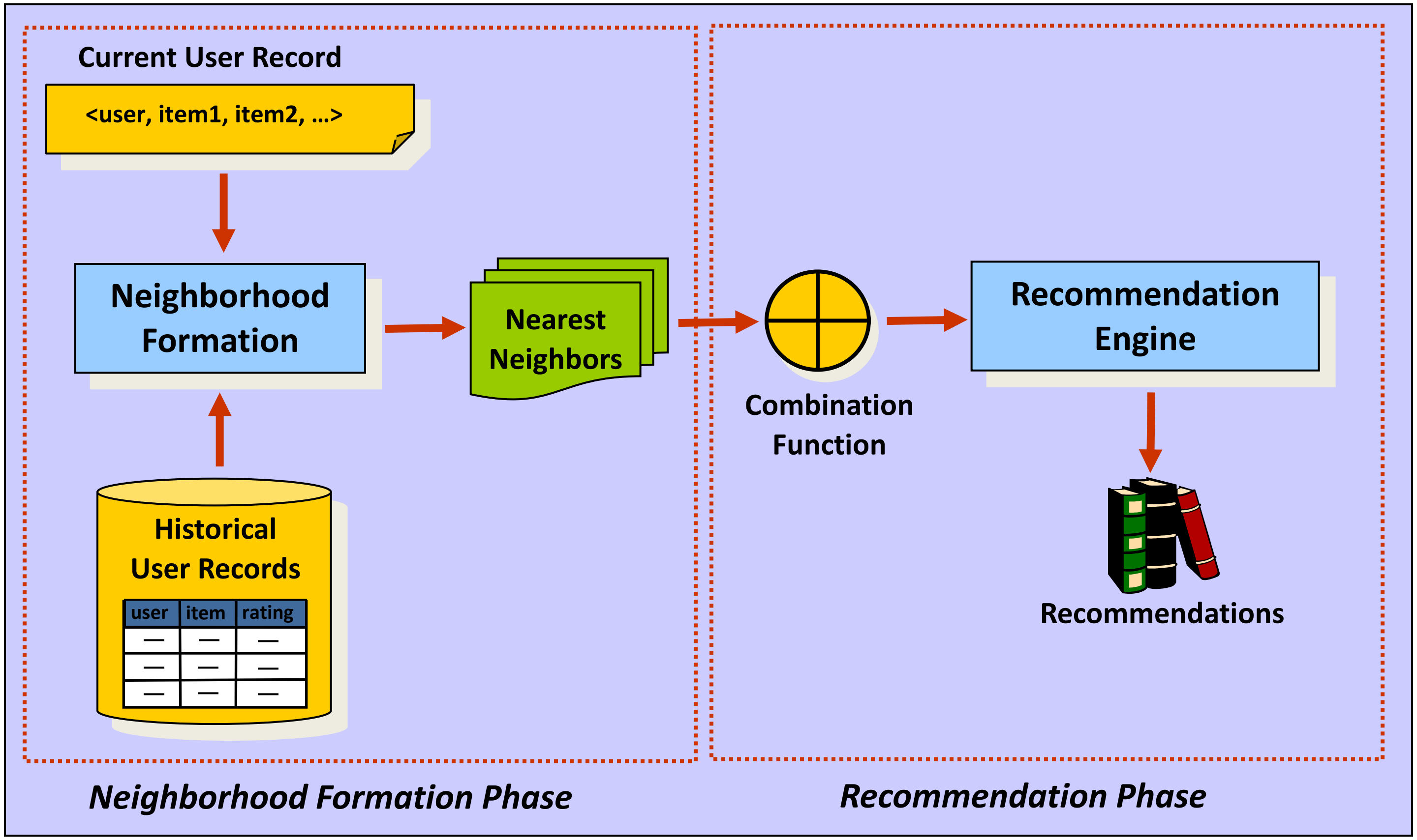
Can tell if the recommender does not have enough data on users' preferences
Many "Best seller recommendations!

# Collaborative Recommender Systems

- Get more recommendations by rating more movies
- Or system can base recommendations based on what you watch…
- Can compare you to other people “like you”



# Basic Collaborative Filtering Process



# User-Based Collaborative Filtering

- User-User Similarity: Pearson Correlation

$$s(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R}_u)(R_{u,j} - \overline{R}_u)}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R}_u)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R}_u)^2}}$$

$\overline{R}_u$ = mean rating for user *u*

$R_{u,i}$ = rating of user *u* on item *i*

***sim*(*i,j*)** = Pearson correlation between *users i and j*

- Making Predictions: K-Nearest-Neighbor

$$p_{a,i} = \overline{R}_a + \frac{\sum_{u=1}^{k}(R_{u,i} - \overline{R}_u) \times sim(a,u)}{\sum_{u=1}^{k} sim(a,u)}$$

$P_{a,i}$ = *predicted rating of user a on item I*

$\overline{R}_a$ = mean rating for target user *a*

***Sim(a,u)*** similarity (Pearson) between user *a* and neighbor *u*

# Example User-Based Collaborative System

| | Item1 | Item 2 | Item 3 | Item 4 | Item 5 | Item 6 | Correlation with Alice |
|---|---|---|---|---|---|---|---|
| Alice | 5 | 2 | 3 | 3 | | ? | |
| User 1 | 2 | | 4 | | 4 | 1 | -1.00 |
| User 2 | 2 | 1 | 3 | | 1 | 2 | 0.33 |
| User 3 | 4 | 2 | 3 | 2 | | 1 | .90 |
| User 4 | 3 | 3 | 2 | | 3 | 1 | 0.19 |
| User 5 | | 3 | | 2 | 2 | 2 | -1.00 |
| User 6 | 5 | 3 | | 1 | 3 | 2 | 0.65 |
| User 7 | | 5 | | 1 | 5 | 1 | -1.00 |

Prediction

Best match

*Using k-nearest neighbor with k = 1*

# Item-Based Collaborative Filtering

- Find similarities among the items **based on ratings across users**
  - Often measured based on a variation of Cosine measure
- Prediction of item I for user *a* is based on the past ratings of user *a* on items similar to *i*.

| | Star Wars | Jurassic Park | Terminator 2 | Indep. Day |
|---|---|---|---|---|
| **Sally** | 7 | 6 | 3 | 7 |
| **Bob** | 7 | 4 | 4 | 6 |
| **Chris** | 3 | 7 | 7 | 2 |
| **Lynn** | 4 | 4 | 6 | 2 |
| | | | | |
| **Karen** | 7 | 4 | 3 | ? |

- Suppose: **sim(Star Wars, Indep. Day) > sim(Jur. Park, Indep. Day) > sim(Termin., Indep. Day)**

- Predicted rating for Karen on Indep. Day will be 7, because she rated Star Wars 7
  - That is if we only use the most similar item
  - Otherwise, we can use the k-most similar items and again use a weighted average

# Item-Based Collaborative Filtering

- item similarity measures
  - cosine

$$sim(i,j) = cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{\|i\| * \|j\|} = \frac{\sum_{u\in U} R_{u,i}R_{u,j}}{\sqrt{\sum_{u\in U} R_{u,i}^2}\sqrt{\sum_{u\in U} R_{u,j}^2}}$$

  - adjusted cosine

$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i}-\bar{R}_u)(R_{u,j}-\bar{R}_u)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_u)^2}}$$

  - pearson correlation

$$sim(i,j) = \frac{Cov(i,j)}{\sigma_i\sigma_j} = \frac{\sum_{u\in U}(R_{u,i}-\bar{R}_i)(R_{u,j}-\bar{R}_j)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_i)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_j)^2}}$$

**Adjusted cosine**: items are vectors in the m dimensional user space (difference in rating scale between users is not taken into account)

- **R(u,i) = rating of user u on item i.**
- **R(i) = average rating of the i-th item.**

- **R(u,i) = rating of user u on item i.**
- **R(u) = average of the u-th user.**

# Item-Based Collaborative Filtering

| | Item1 | Item 2 | Item 3 | Item 4 | Item 5 | Item 6 |
|---|---|---|---|---|---|---|
| Alice | 5 | 2 | 3 | 3 | | ? |
| User 1 | 2 | | 4 | | 4 | 1 |
| User 2 | 2 | 1 | 3 | | 1 | 2 |
| User 3 | 4 | 2 | 3 | 2 | | 1 |
| User 4 | 3 | 3 | 2 | | 3 | 1 |
| User 5 | | 3 | | 2 | 2 | 2 |
| User 6 | 5 | 3 | | 1 | 3 | 2 |
| User 7 | | 5 | | 1 | 5 | 1 |
| Item similarity | 0.76 | 0.79 | 0.60 | 0.71 | 0.75 | |

Prediction

Best match

# Collaborative Filtering Evaluation

- Split users into train/test sets
- For each user a in the test set:
  - split a's votes into observed (*I*) and to-predict (*P*)
  - measure average absolute deviation between predicted and actual votes in *P*
  - **MAE** = mean absolute error
  - Or **RMSE** = root mean squared error
- Average over all test users

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}|$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2$$

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2}$$

$$R^2 = 1 - \frac{\sum (y_i - \hat{y})^2}{\sum (y_i - \bar{y})^2}$$

Where,

$\hat{y}$ – *predicted value of y*

$\bar{y}$ – *mean value of y*

# Data Sparsity Problem

- Cold start problem
  - How to recommend new items? What to recommend to new users?
- Straightforward approaches
  - Ask/force users to rate a set of items
  - Use another method (e.g., content-based, demographic or simply non-personalized e.g., "**best seller**", "**top 10 in the US**") in the initial phase
- Alternatives
  - Use better algorithms (beyond nearest-neighbor approaches)
    - In nearest-neighbor approaches, the set of sufficiently similar neighbors might be too small to make good predictions
    - Use model-based approaches (clustering; dimensionality reduction, etc.)

# Example Algorithm for Sparse Data

Recursive Collaborative Filtering (CF)

- Assume there is a very close neighbor ***n*** of u who has not yet rated the target item ***i*** .
- Apply CF-method recursively and predict a rating for item ***i*** for the neighbor
- Use this predicted rating instead of the rating of a more distant direct neighbor

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

**sim = 0.85**

**Predict rating for User1**

# More Model-Based Approach

- Matrix factorization techniques, statistics
  - singular value decomposition, principal component analysis
- Approaches based on clustering
- Association rule mining (ARM)
  - Market Basket Analysis (MBA)
  - Customers who bought this, also bought that… "**Frequently bought together**"
- Probabilistic models
  - clustering models, Bayesian networks, probabilistic Latent Semantic Analysis
- Various other machine learning approaches

# Dimensionality Reduction for CF

- Basic idea: Trade more complex offline model building for faster online prediction generation
- Singular Value Decomposition for dimensionality reduction of rating matrices
  - Captures important factors/aspects and their weights in the data
  - factors can be genre, actors but also non-understandable ones
  - Assumption that k dimensions capture the signals and filter out noise (K = 20 to 100)
- Approach also popular in information retrieval (Latent Semantic Indexing), data compression, …

# Netflix Prize & Matrix Factorization

17,700 movies

480,000 users

| | | | | | |
|---|---|---|---|---|---|
| 1 | 3 | 4 | | | |
| | 3 | 5 | | | 5 |
| | | 4 | 5 | | 5 |
| | | 3 | | | |
| | | 3 | | | |
| 2 | | | 2 | | 2 |
| | | | | 5 | |
| | 2 | 1 | | | 1 |
| | 3 | | | 3 | |
| 1 | | | | | |

*The $1 Million Question*


2008: *Factorization meets the neighborhood: a multifaceted collaborative filtering model*, Y. Koren, ACM SIGKDD

# Netflix Paper

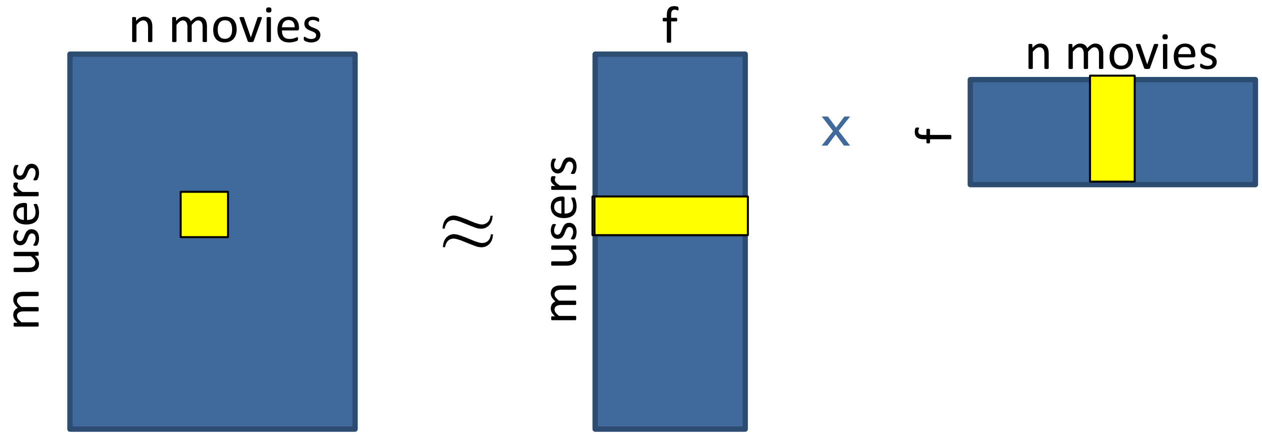
2008: *Factorization meets the neighborhood: a multifaceted collaborative filtering model*, Y. Koren, ACM SIGKDD

- Merge neighborhood models with latent factor models
- Latent factor models
  - good to capture weak signals in the overall data
- Neighborhood models
  - good at detecting strong relationships between close items
- Combination in one prediction single function
  - Local search method such as stochastic gradient descent to determine parameters
  - Add penalty for high values to avoid over-fitting

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

$$\min_{p_*, q_*, b_*} \sum_{(u,i)\in K} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2)$$

# Matrix Factorization of Ratings Data




Based on the idea of Latent Factor Analysis

- Identify latent (unobserved) factors that "explain" observations in the data
- In this case, observations are user ratings of movies
- The factors may represent combinations of features or characteristics of movies and users that result in the ratings

# Matrix Factorization

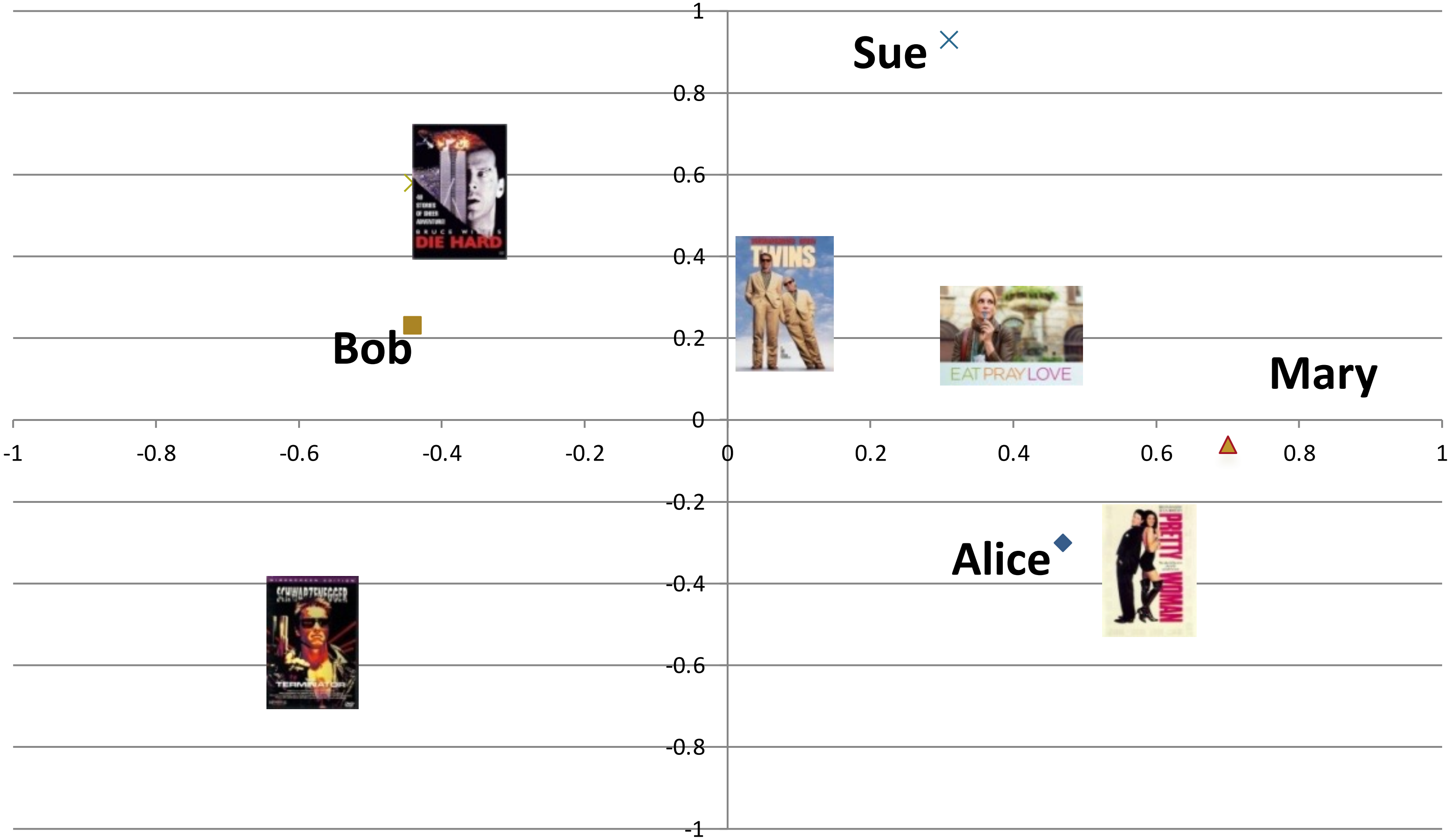
| $P_k$ | Dim1 | Dim2 |
|---|---|---|
| **Alice** | 0.47 | -0.30 |
| **Bob** | -0.44 | 0.23 |
| **Mary** | 0.70 | -0.06 |
| **Sue** | 0.31 | 0.93 |

| $Q_k^T$ |  |  |  |  |  |
|---|---|---|---|---|---|
| **Dim1** | -0.44 | -0.57 | 0.06 | 0.38 | 0.57 |
| **Dim2** | 0.58 | -0.66 | 0.26 | 0.18 | -0.36 |

**Prediction:** $\hat{r}_{ui} = p_k(Alice) \times q_k^T(EPL)$

- Note: Can also do factorization via Singular Value Decomposition (SVD)
- SVD: $M_k = U_k \times \Sigma_k \times V_k^T$

# Lower Dimensional Feature Space



Using content of items to make recommendations

# CONTENT-BASED RECOMMENDATIONS

# Content-Based Recommendation

- Collaborative filtering does **NOT** require any information about the items,
  - However, it might be reasonable to exploit such information
  - e.g., recommend fantasy novels to people who liked fantasy novels in the past
- What do we need:
  - Some information about the available items such as the genre ("content")
  - Some sort of *user profile* describing what the user likes (the preferences)
- The task:
  - Learn user preferences
  - Locate/recommend items that are "similar" to the user preferences

# Content-Based Recommendation

- Predictions for unseen (target) items are computed based on their similarity (in terms of content) to items in the user profile.
- e.g., user profile $P_u$ contains

Highly recommended ← 


Mildly recommended ← 

# Content Representation and Item Similarities

| Title | Genre | Author | Type | Price | Keywords |
|---|---|---|---|---|---|
| The Night of the Gun | Memoir | David Carr | Paperback | 29.90 | Press and journalism, drug addiction, personal memoirs, New York |
| The Lace Reader | Fiction, Mystery | Brunonia Barry | Hardcover | 49.90 | American contemporary fiction, detective, historical |
| Into the Fire | Romance, Suspense | Suzanne Brockmann | Hardcover | 45.90 | American fiction, Murder, Neo-nazism |
| ... | | | | | |

Represent items as vectors over features

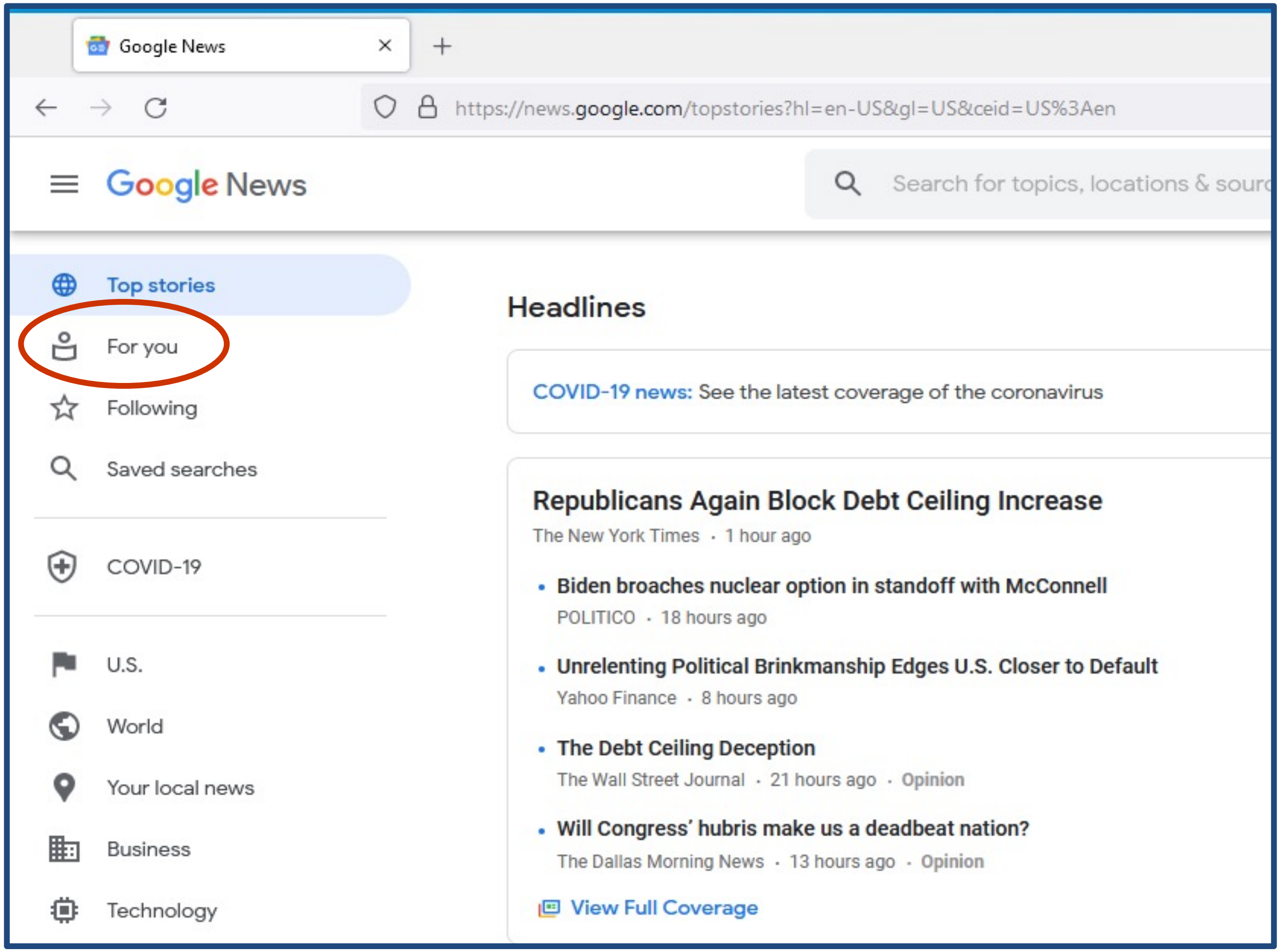
- Features may be items attributes, keywords, tags, etc.
- Often items are represented a keyword vectors based on textual descriptions with TFxIDF or other weighting approaches
  - applicable to any type of item (images, products, news stories) as long as a textual description is available or can be constructed
- Items (and users) can then be compared using standard vector space similarity measures (e.g., Cosine similarity)

# Content-Based Recommendation

Basic approach

- Represent items as vectors over features
- User profiles are also represented as aggregate feature vectors
  - Based on items in the user profile (e.g., items liked, purchased, viewed, clicked on, etc.)
- Compute the similarity of an unseen item with the user profile based on the keyword overlap (e.g., using the **Dice coefficient – common when dealing with sets of items**)
- $\text{sim}(b_i, b_j) = \frac{2 * |keywords(b_i) \cap keywords(b_j)|}{|keywords(b_i)| + |keywords(b_j)|}$
- Other similarity measures such as Cosine can also be used
- Recommend items most similar to the user profile

# Content-Based Recommendation




Example of Google News Recommendations on 10/6/2021
Top news, followed by personal news recommendations etc.

# Content-Based Recommendation

- How can the search engine determine the “user’s intent”?

**Query: “Madonna and Child”**

?

?

PeopleNews

E-MAIL IM

**Madonna Ready for Another Baby?**

Wednesday Nov 24, 2004 1:55pm EST

By Todd Gold

Three months after finishing her Re-Invention Tour, Madonna is currently enjoying quiet time with her family in London, she's just published her fourth book for young readers, *The Adventures of Abdi* – and, at 46, she tells PEOPLE she wouldn't mind getting pregnant again.

She's not making any definite plans, but the pop icon says: "I'm going to have fun with my husband and see what happens."

CREDIT: JO HALE / GETTY

- Need to “learn” the user profile:
  – User is an art historian?
  – User is a pop music fan?

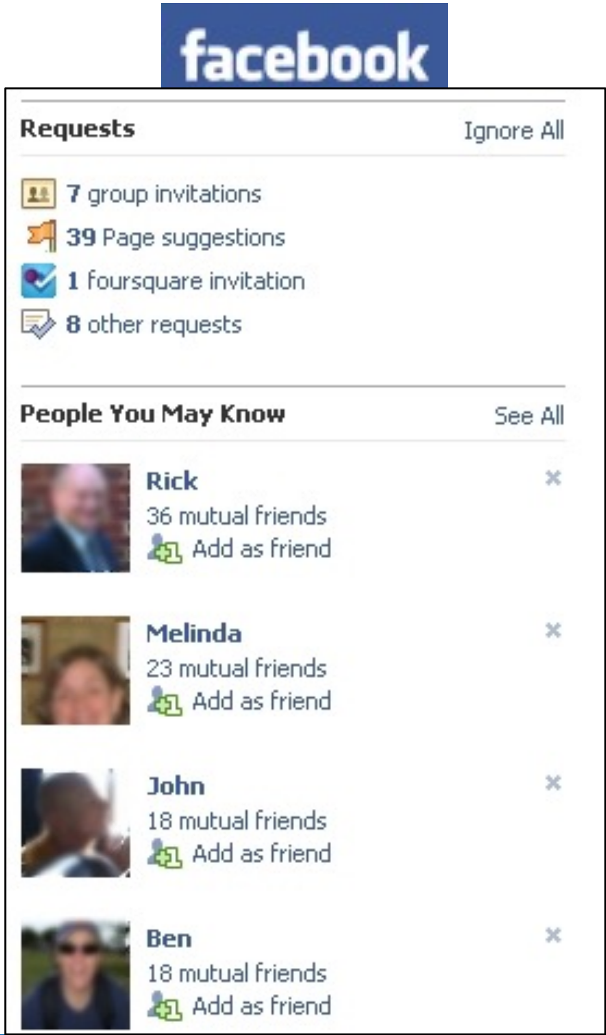
# Content-Based Recommenders

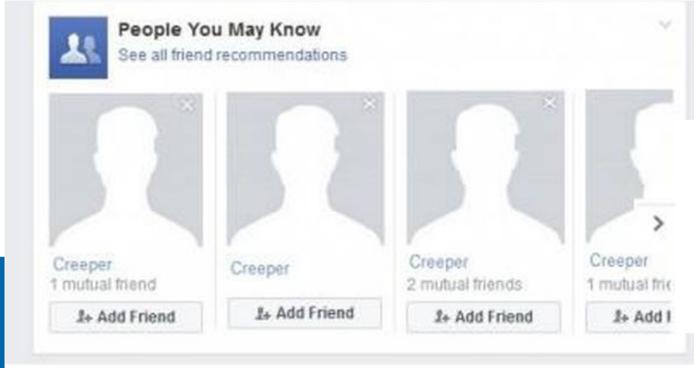
- More examples:
  - Spotify for music: playlist generation
  - YouTube suggestions
  - eBooks on amazon prime
  - News recommendations
- **Ethical** issue here:
  - Feed more conspiracy theories to users who like them…
    - the more engaged we are the more ads we see
  - Should companies make money from topics that polarize us? From misinformation etc.

# Social Recommendations

A form of collaborative filtering using social network data

- User profiles represented as sets of links to other nodes (users or items) in the network
- Prediction problem: infer a currently non-existent link in the network (graph analysis)
- Ethical questions about where the “content” comes from …





https://www.forbes.com/sites/curtissilver/2016/06/28/how-facebooks-people-you-may-know-section-just-got-creepier/?sh=616be8d35f5a

Watch 7 Days Free

# Social / Collaborative Tags

# Example: Tags describe the Resource

**Tags Customers Associate with This Product** (What's this?)
Click on a tag to find related items, discussions, and people.

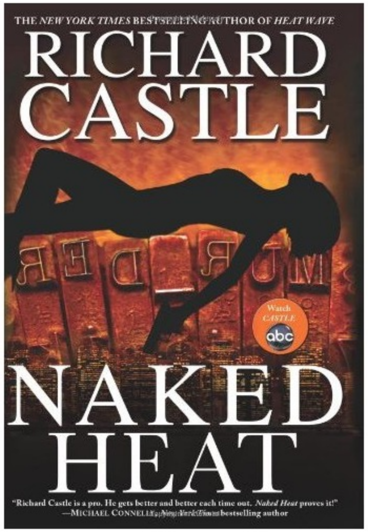
**Check the boxes next to the tags you consider relevant or enter your own tags in the field below.**

- nathan fillion (24)
- castle (22)
- nikki heat (21)
- crime drama (16)
- beckett (14)
- stana katic (14)
- mystery (10)
- abc (6)
- super saver shipping (2)
- boycott over 9 99 (1)
- cascett (1)
- ficticious fiction (1)

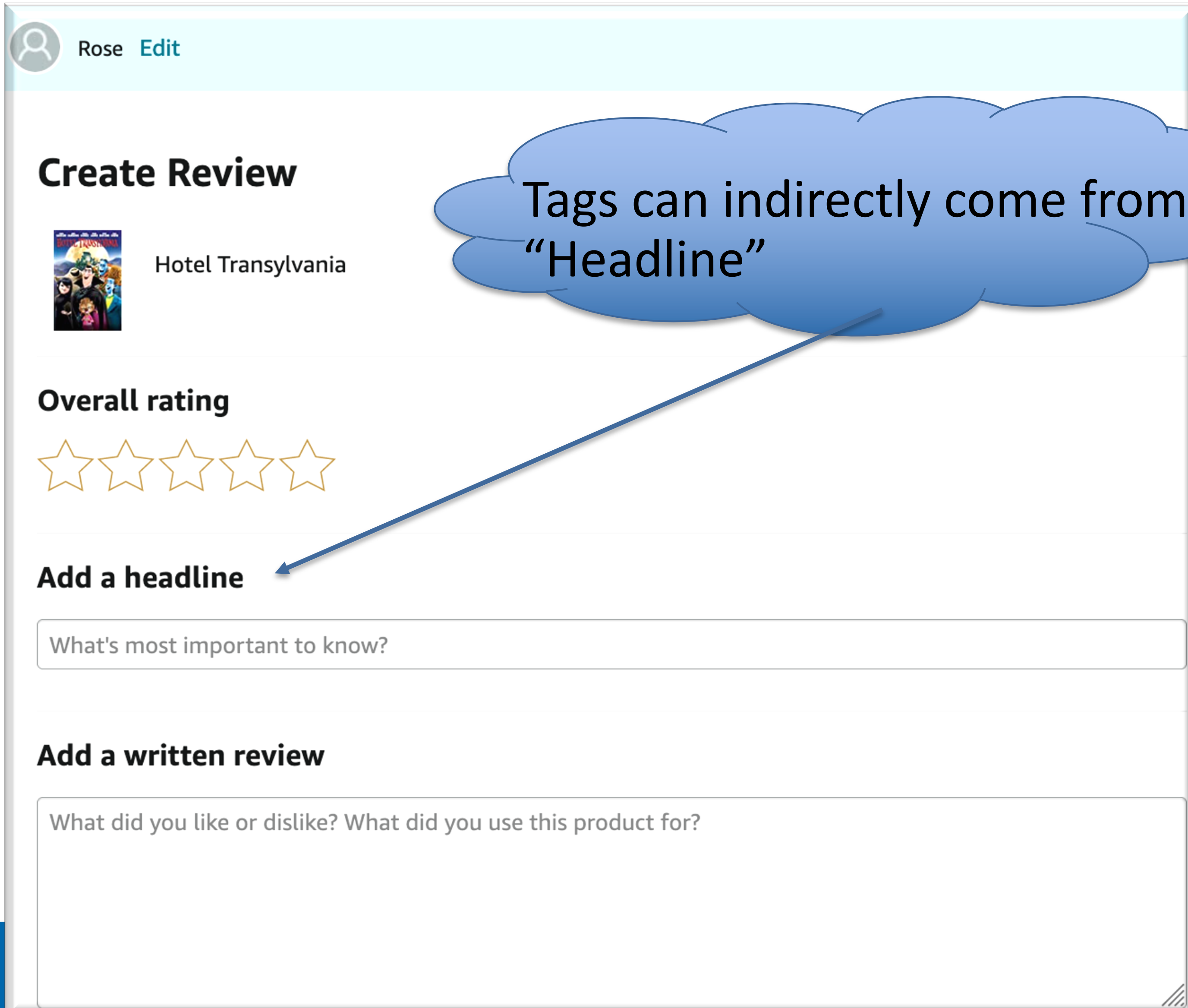
**Agree with these tags?**

Tags can describe

- The resource (genre, actors, etc.)
- Organizational (toRead)
- Subjective (awesome)
- Ownership (abc)
- etc.



# Tag Recommendation




Tags can indirectly come from "Headline"

# Tags Describe the User

These systems are "collaborative."

- Recommendation / Analytics based on the "wisdom of crowds."

Rai Aren's profile
co-author
"Secret of the Sands"

**Location:** Canada

**Web Page:** www.secretofthesands.com

**In My Own Words:**
RAI AREN

Rai loves the stories of Lord of the Rings, Star Wars, Star Trek, Indiana Jones (her first kitty cat is named Indiana, Indy for short), and The Matrix (take the red pill!), to name a few. She loves getting lost in these enchanting worlds and studying their underlying philosophies. Ancient Egypt has held a particular fascination for her since childhood.

Rai feels that novels have the abi...
Read more

**Interests**
Reading, writing novels (there are lots of fascinating & very cool ones to come, so stay tuned!), travel, movies, being good to mama earth & all of her inhabitants :)

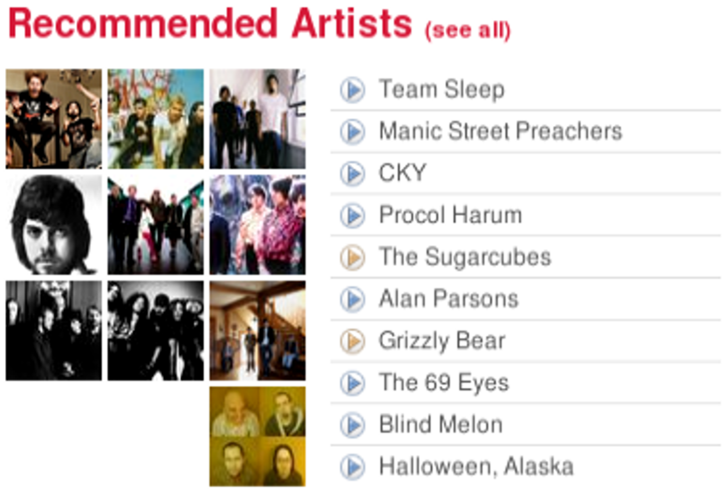
**Frequently Used Tags**
action action adventure action thriller adventure archaeology childrens books egypt fantasy fiction historical fiction horror humor indiana jones inspirational kindle kindle authors kindle book love love story magic memoir mystery novel paranormal paranormal romance romance science fiction suspense thriller young adult

› **See all 1,832 tagged items**

# Example: Using Tags for Recommendation

## Last.fm recommendations

- Recommendations:
  - Primarily Collaborative Filtering
  - Item-Item (artist recommendations
  - User-User (Neighbors)
  - Could use:
    tags, audio, metadata
- Evaluating (rel. feedback)
  - Tracking Love/Ban behavior




# COMBINING CONTENT-BASED AND COLLABORATIVE RECOMMENDATIONS

# Combining Content-Based and Collaborative Recommendation

- Example: Semantically Enhanced CF
  - Extend item-based collaborative filtering to incorporate both similarities based on ratings (or usage) as well as semantic similarity based on content / semantic information
- Semantic knowledge about items
  - Can be extracted automatically from the Web based on domain-specific reference ontologies
  - Used in conjunction with user-item mappings to create a combined similarity measure for item comparisons
  - Singular value decomposition used to reduce noise in the content data
- Semantic combination threshold
  - Used to determine the proportion of semantic and rating (or usage) similarities in the combined measure

# Semantically Enhanced Hybrid Recommendation

An extension of the item-based algorithm

- Use a combined similarity measure to compute item similarities:

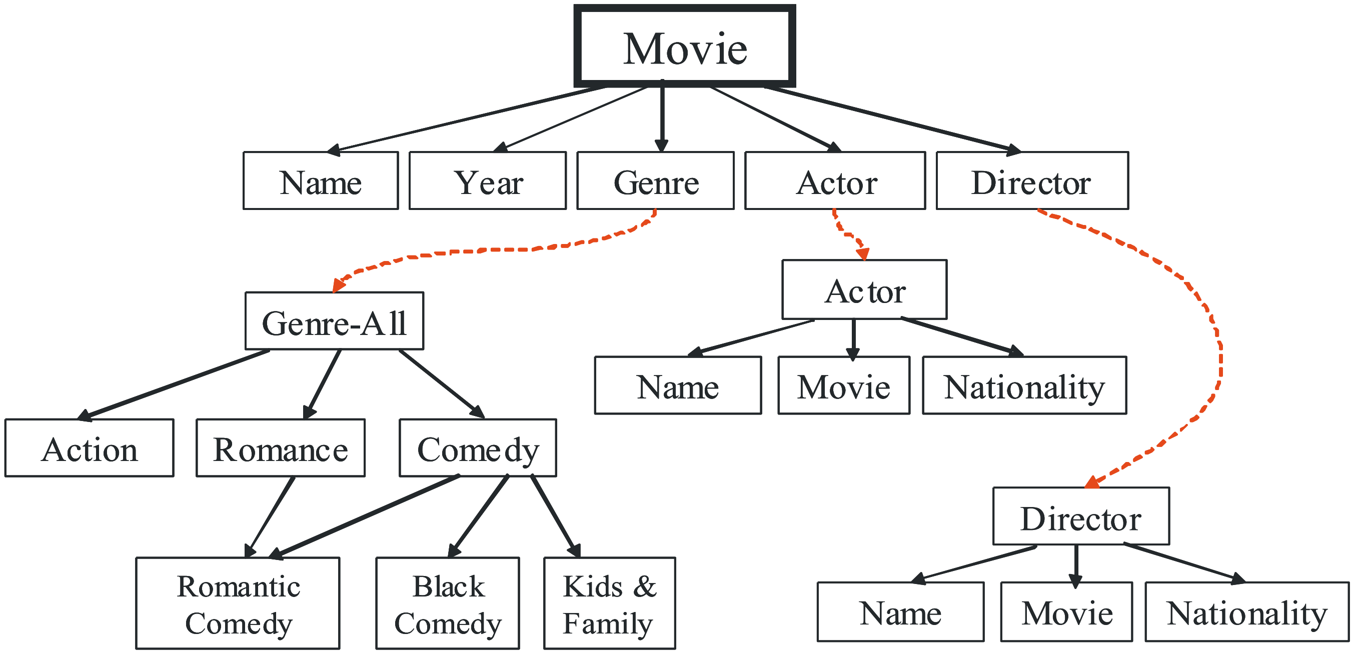
$$\boxed{CombinedSim(i_p, i_q) = (1 - \alpha) \cdot SemSim(i_p, i_q) + \alpha \cdot RateSim(i_p, i_q)}$$

- where,
  - *SemSim* is the similarity of items $i_p$ and $i_q$ based on semantic features (e.g., keywords, attributes, etc.); and
  - *RateSim* is the similarity of items $i_p$ and $i_q$ based on user ratings (as in the standard item-based CF)
- $\alpha$ is the semantic combination parameter:
  - $\alpha$ = 1 → only user ratings; no semantic similarity
  - $\alpha$ = 0 → only semantic features; no collaborative similarity

# Semantically Enhanced CF

Movie data set

- Movie ratings from the movielens data set
- Semantic info. extracted from IMDB based on the following ontology



# Data Mining Approach to Personalization

- Basic Idea
  - generate aggregate user models (usage profiles) by discovering user access patterns through Web usage mining (offline process)
    - Clustering user transactions
    - Clustering items
    - Association rule mining
    - Sequential pattern discovery
  - match a user's profile against the discovered models to provide dynamic content (online process)

- Advantages
  - Can be applied to different types of user data (ratings, pageviews, items purchased, etc.)
  - Helps enhance the scalability of collaborative filtering

# Conceptual Representation of User Profile Data

Items

User Profiles

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| **user0** | 15 | 5 | 0 | 0 | 0 | 185 |
| **user1** | 0 | 0 | 32 | 4 | 0 | 0 |
| **user2** | 12 | 0 | 0 | 56 | 236 | 0 |
| **user3** | 9 | 47 | 0 | 0 | 0 | 134 |
| **user4** | 0 | 0 | 23 | 15 | 0 | 0 |
| **user5** | 17 | 0 | 0 | 157 | 69 | 0 |
| **user6** | 24 | 89 | 0 | 0 | 0 | 354 |
| **user7** | 0 | 0 | 78 | 27 | 0 | 0 |
| **user8** | 7 | 0 | 45 | 20 | 127 | 0 |
| **user9** | 0 | 38 | 57 | 0 | 0 | 15 |

# Example: Using Clusters for Web Personalization

**User navigation sessions**

| | A.html | B.html | C.html | D.html | E.html | F.html |
|---|---|---|---|---|---|---|
| user0 | 1 | 1 | 0 | 0 | 0 | 1 |
| user1 | 0 | 0 | 1 | 1 | 0 | 0 |
| user2 | 1 | 0 | 0 | 1 | 1 | 0 |
| user3 | 1 | 1 | 0 | 0 | 0 | 1 |
| user4 | 0 | 0 | 1 | 1 | 0 | 0 |
| user5 | 1 | 0 | 0 | 1 | 1 | 0 |
| user6 | 1 | 1 | 0 | 0 | 0 | 1 |
| user7 | 0 | 0 | 1 | 1 | 0 | 0 |
| user8 | 1 | 0 | 1 | 1 | 1 | 0 |
| user9 | 0 | 1 | 1 | 0 | 0 | 1 |

**Result of Clustering**

| | | A.html | B.html | C.html | D.html | E.html | F.html |
|---|---|---|---|---|---|---|---|
| Cluster 0 | user 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| | user 4 | 0 | 0 | 1 | 1 | 0 | 0 |
| | user 7 | 0 | 0 | 1 | 1 | 0 | 0 |
| Cluster 1 | user 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| | user 3 | 1 | 1 | 0 | 0 | 0 | 1 |
| | user 6 | 1 | 1 | 0 | 0 | 0 | 1 |
| | user 9 | 0 | 1 | 1 | 0 | 0 | 1 |
| Cluster 2 | user 2 | 1 | 0 | 0 | 1 | 1 | 0 |
| | user 5 | 1 | 0 | 0 | 1 | 1 | 0 |
| | user 8 | 1 | 0 | 1 | 1 | 1 | 0 |

PROFILE 0 (Cluster Size = 3)

--------------------------------------

1.00 C.html
1.00 D.html

PROFILE 1 (Cluster Size = 4)

--------------------------------------

1.00 B.html
1.00 F.html
0.75 A.html
0.25 C.html

PROFILE 2 (Cluster Size = 3)

--------------------------------------

1.00 A.html
1.00 D.html
1.00 E.html
0.33 C.html

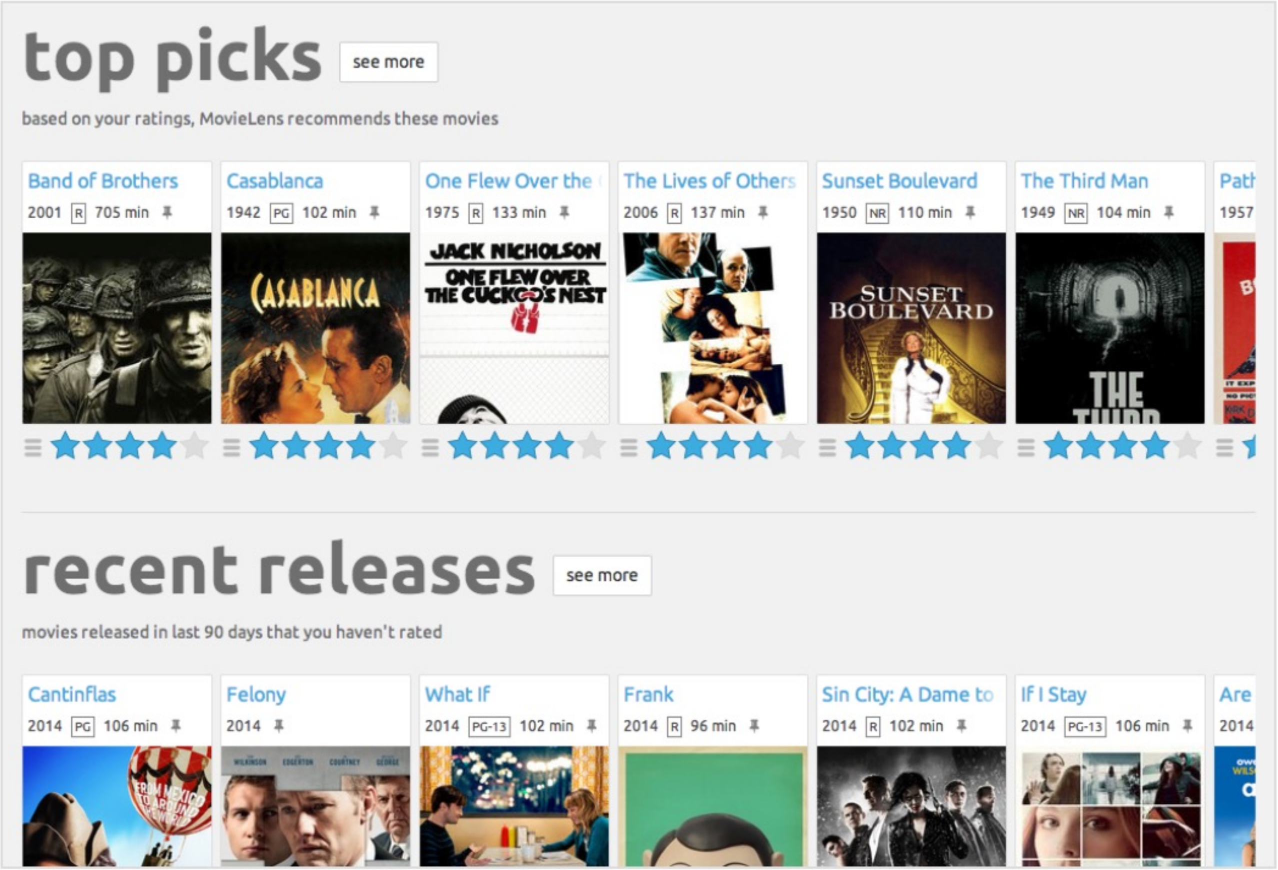
**Given an active session A → B, the best matching profile is Profile 1. This may result in a recommendation for page F.html, since it appears with high weight in that profile.**

# Clustering and Collaborative Filtering

## clustering based on ratings: movielens

"MovieLens helps you find movies you will like. Rate movies to build a custom taste profile, then MovieLens recommends other movies for you to watch."

***https://movielens.org/***



# Clustering and Collaborative Filtering

:: tag clustering example