Information Storage and Retrieval

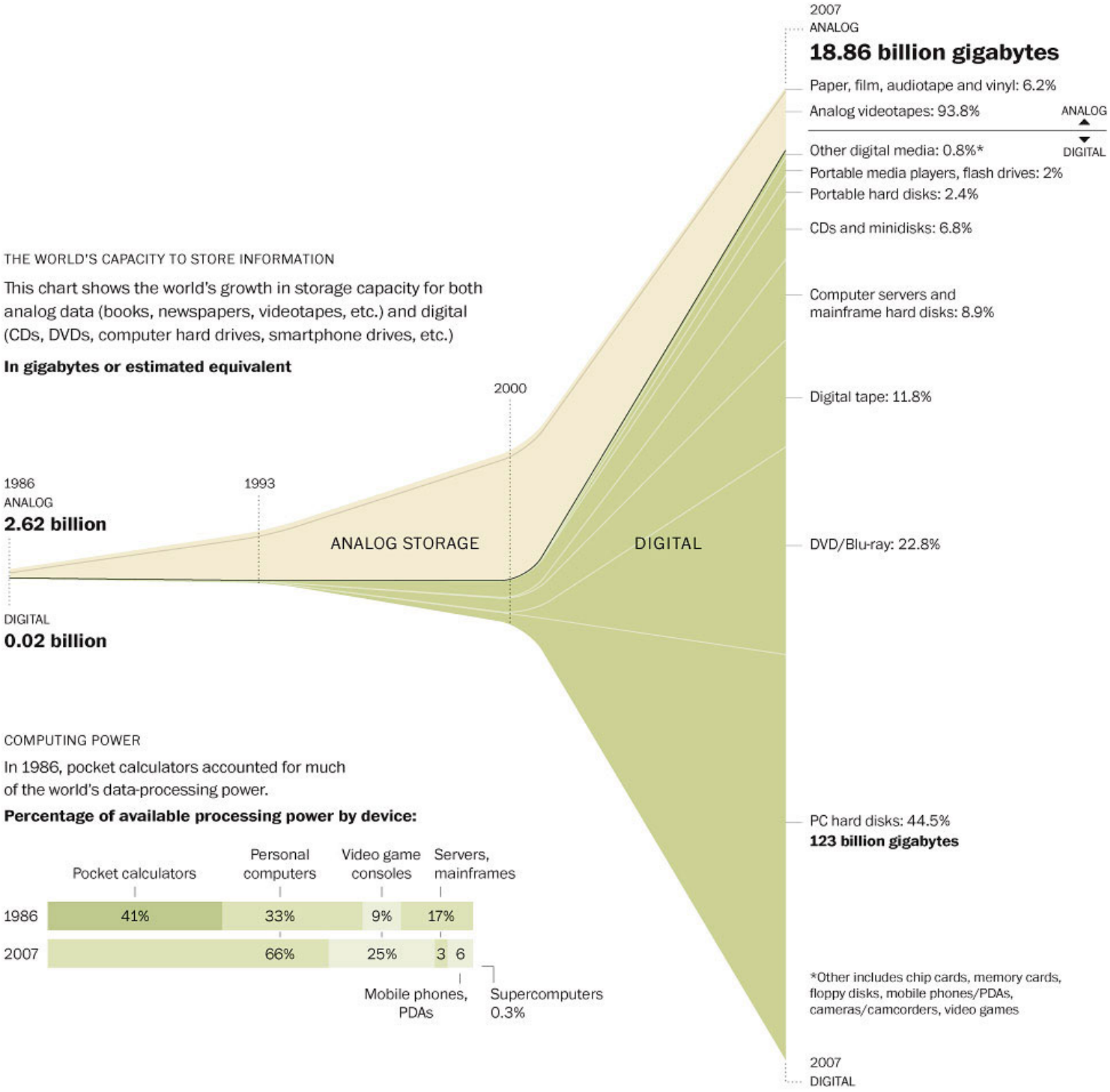
CSCE 670
Texas A&M University
Department of Computer Science & Engineering
Instructor: Prof. James Caverlee

Recommenders
6/8 March 2018

Today

- Recommendation systems: Intro
- Collaborative Filtering
  - Finding nearest neighbors
  - How to aggregate ratings?
- Content-based Recommenders
- Netflix Prize (start)

Some slides adapted from MMD: *Mining of Massive Datasets* and from Julian McAuley


THE WORLD'S CAPACITY TO STORE INFORMATION
This chart shows the world's growth in storage capacity for both analog data (books, newspapers, videotapes, etc.) and digital (CDs, DVDs, computer hard drives, smartphone drives, etc.)
In gigabytes or estimated equivalent
1986
ANALOG
2.62 billion
DIGITAL
0.02 billion
1993
2000
ANALOG STORAGE
DIGITAL
2007
ANALOG
18.86 billion gigabytes
Paper, film, audiotape and vinyl: 6.2%
Analog videotapes: 93.8%
ANALOG
DIGITAL
Other digital media: 0.8%*
Portable media players, flash drives: 2%
Portable hard disks: 2.4%
CDs and minidisks: 6.8%
Computer servers and mainframe hard disks: 8.9%
Digital tape: 11.8%
DVD/Blu-ray: 22.8%
PC hard disks: 44.5%
123 billion gigabytes
*Other includes chip cards, memory cards, floppy disks, mobile phones/PDAs, cameras/camcorders, video games
2007
DIGITAL
COMPUTING POWER
In 1986, pocket calculators accounted for much of the world's data-processing power.
Percentage of available processing power by device:
Pocket calculators
Personal computers
Video game consoles
Servers, mainframes
1986
41%
33%
9%
17%
2007
66%
25%
3
6
Mobile phones, PDAs
Supercomputers 0.3%


THE MOBILE WEB RECEIVES 217 NEW USERS.
YOUTUBE USERS UPLOAD 48 HOURS OF NEW VIDEO.
EMAIL USERS SEND 204,166,667 MESSAGES.
GOOGLE RECEIVES OVER 2,000,000 SEARCH QUERIES.
FACEBOOK USERS SHARE 684,478 PIECES OF CONTENT.
WORDPRESS USERS PUBLISH 347 NEW BLOG POSTS.
571 NEW WEBSITES ARE CREATED.
EVERY MINUTE of the DAY
CONSUMERS SPEND $272,070 ON WEB SHOPPING.
FOURSQUARE USERS PERFORM 2,083 CHECK-INS.
TWITTER USERS SEND OVER 100,000 TWEETS.
FLICKR
APPLE

How Many Products Does Amazon Sell?

by **Paul Grey** on 15 December 2013 in **E-Commerce**

Amazon.com is the self-styled "Greatest Store on Earth." It's been said that Amazon aims one day to sell everything to everyone.

Exactly how much choice do you have when shopping with Amazon?

Today Amazon sells over 200 million products in the USA, which are categorised into 35 departments. There are almost 5 million items in the Clothing department, almost 20 million in Sports & Outdoors, and over 4 million Office Products. There 7 million items in the Amazon Jewelry department, 24 million in Electronics, 1.4 million products in the Beauty department, 570 thousand Baby products, and 600 thousand Grocery items.

That's in the USA. This table lists my estimates of the number of products offered on the main Amazon websites around the world.

Amazon.com	USA	232 million
Amazon.co.uk	UK	132 million
Amazon.de	Germany	118 million

How many tracks does it have?

Spotify: 30 million+

Apple Music: 30 million+

The two services have roughly the same number of tracks; some 30 million at their disposal.

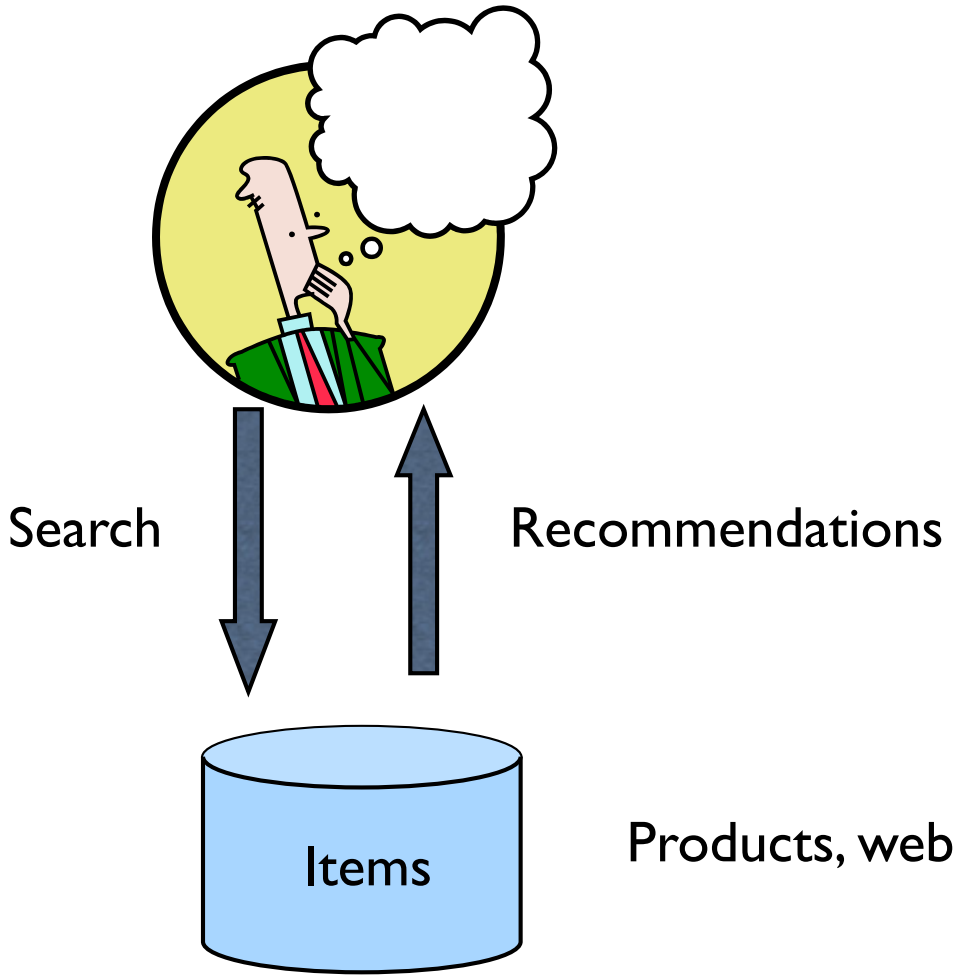
Apple Music users can listen to their own personal iTunes library through the app as well as the Apple Music streaming library. Spotify allows you to listen to non-Spotify music too, but adding local files has to be done through a desktop Mac or PC. For example, you can't listen to your iTunes library through Spotify itself.

Statistics

Viewership

- More than 1 billion unique users visit YouTube each month
- Over 6 billion hours of video are watched each month on YouTube—that's almost an hour for every person o
- 100 hours of video are uploaded to YouTube every minute
- 80% of YouTube traffic comes from outside the US
- YouTube is localized in 61 countries and across 61 languages
- According to Nielsen, YouTube reaches more US adults ages 18-34 than any cable network
- Millions of subscriptions happen each day. The number of people subscribing daily is up more than 3x since
  number of daily subscriptions is up more than 4x since last year

Recommendations



Products, web sites, blogs, news items, …

Why recommendation?

The goal of recommender systems is…

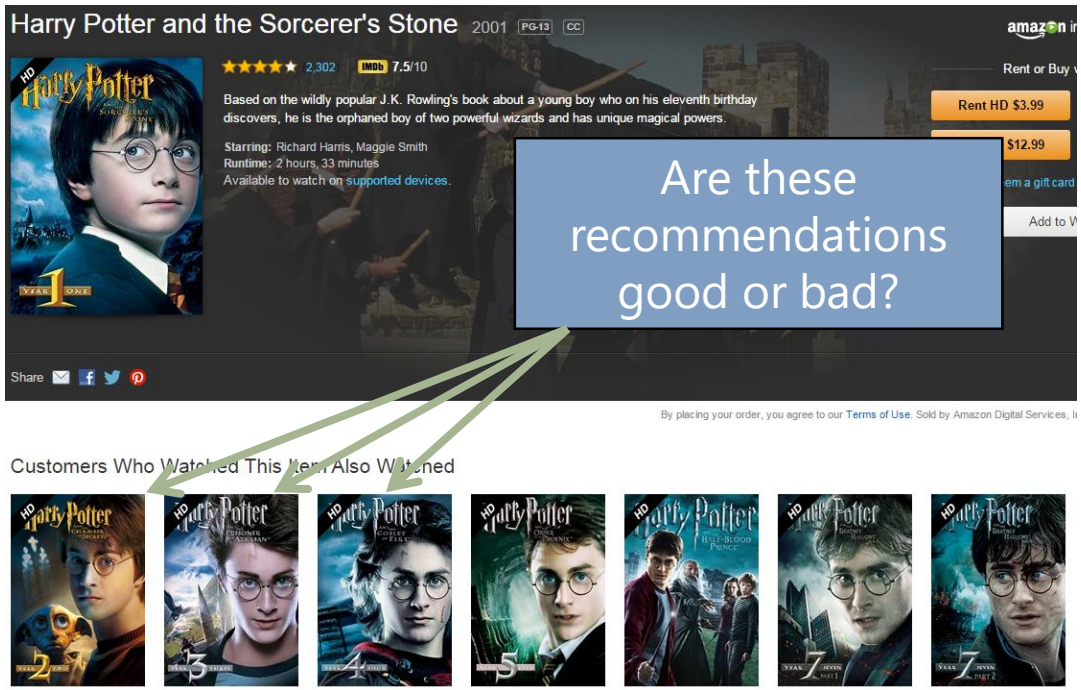
- To help people discover new content

Recommendations for You in Amazon Instant Video See more

Why recommendation?

The goal of recommender systems is…

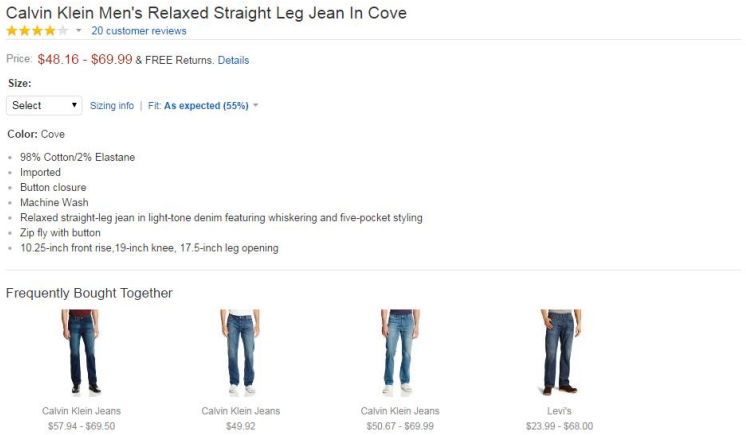
- To help us find the content we were already looking for



Why recommendation?

The goal of recommender systems is…

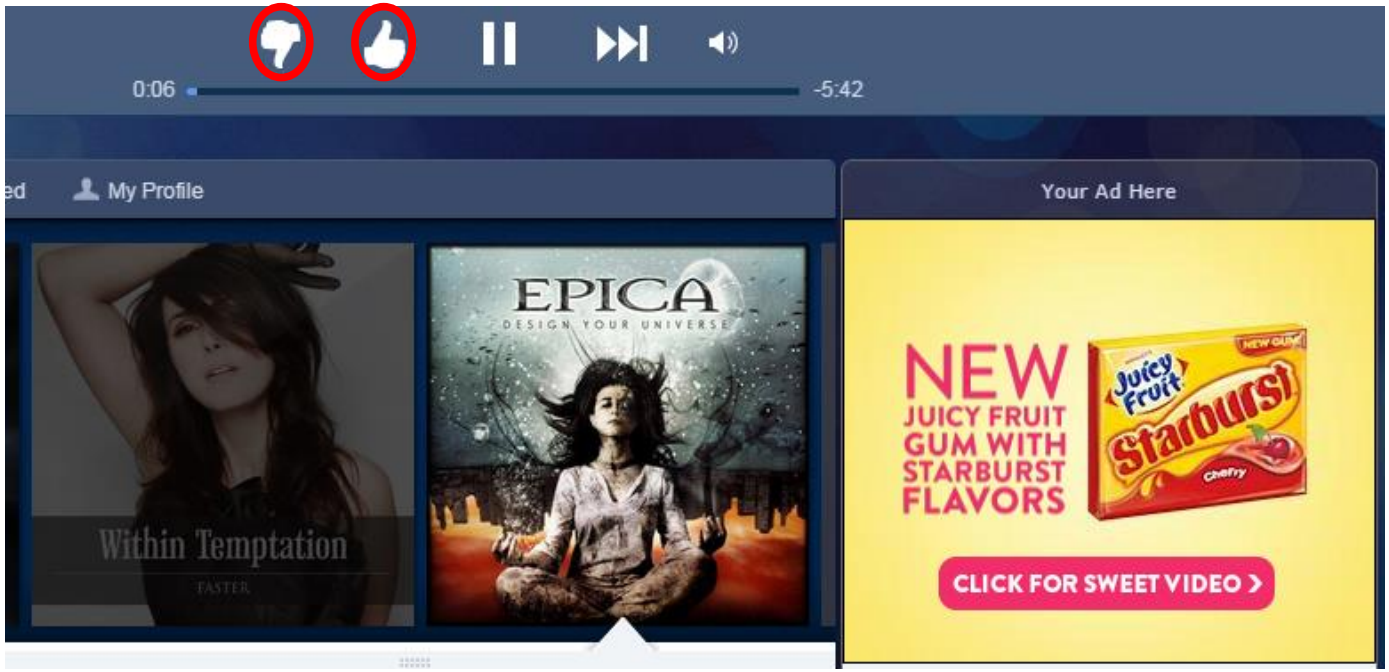
- To discover which things go together



Why recommendation?

The goal of recommender systems is…

- To personalize user experiences in response to user feedback



Why recommendation?

The goal of recommender systems is…

- To identify things that we **like**

Examples of recommenders?

Recommendation Types

- Editorial and hand curated
  - List of favorites
  - Lists of "essential" items
- Simple aggregates
  - Top 10, Most Popular, Recent Uploads
- **Tailored to individual users**
  - **Amazon, Netflix, ...**

$$$


Netflix Prize: Home
http://www.netflixprize.com/
netflix prize
Getting Started
Latest Headlines
NETFLIX
Netflix Prize
Home
Rules
Leaderboard
Register
Update
Submit
Download
NETFLIX
Browse
Recommendations
Friends
Queue
Buy DVDs
Movies For You
Welcome!
The Netflix Prize seeks to substantially improve the accuracy of predictions about how much someone is going to love a movie based on their movie preferences. Improve it enough and you win one (or more) Prizes. Winning the Netflix Prize improves our ability to connect people to the movies they love.
Read the Rules to see what is required to win the Prizes. If you are interested in joining the quest, you should register a team.
You should also read the frequently-asked questions about the Prize. And check out how various teams are doing on the Leaderboard.
Good luck and thanks for helping!
FAQ | Forum | Netflix Home
© 1997-2006 Netflix, Inc. All rights reserved.
Done

Formal Model

- X = set of Customers
- S = set of Items
- Utility function $u: X \times S \rightarrow R$
  - R = set of ratings
  - R is a totally ordered set
  - e.g., 0-5 stars, real number in [0,1]

Utility Matrix

	Spotlight	The Revenant	Mad Max: Fury Road	Room
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

Key Problems

- Gathering "known" ratings for matrix
- Extrapolate unknown ratings from known ratings
  - Mainly interested in high unknown ratings
  - Don't care about finding what you **don't like**, but rather what you do like
- Evaluating extrapolation methods
  - How do we know if we've done a good job?

Gathering Ratings

- Explicit
  - Ask people to rate items
  - Doesn't work well in practice – people can't be bothered
- Implicit
  - Learn ratings from user actions
    - e.g., purchase implies high rating
  - What about low ratings?

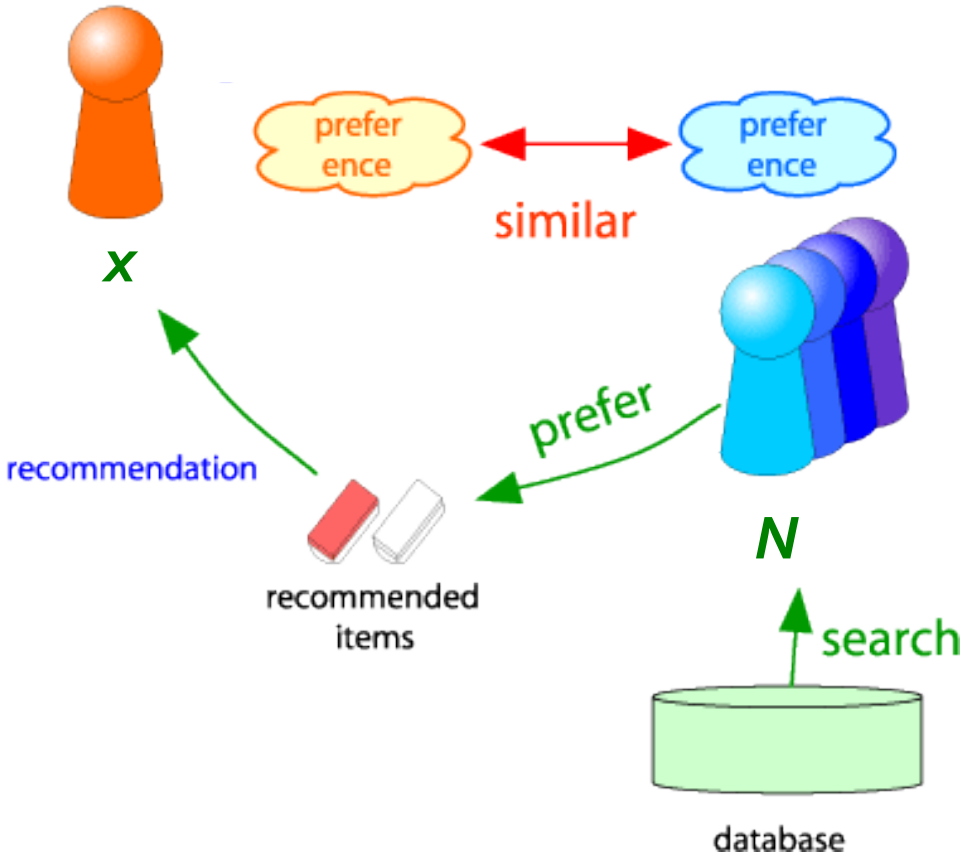
Extrapolating Utilities

- Key problem: matrix U is sparse
  - most people have not rated most items
  - Cold start:
    - New items have no ratings
    - New users have no history
- Three main approaches
  - Collaborative
  - Content-based
  - Latent factor models

Collaborative Recommendations

Collaborative Filtering

- Consider user ***x***
- Find set ***N*** of other users whose ratings are "similar" to ***x***'s ratings
- Estimate ***x***'s ratings based on ratings of users in ***N***



Finding Similar Users

- Let r_x be the vector of user x's ratings
- Jaccard!
  - But it ignores the value of the rating
- Cosine similarity measure
  - sim(x,y) = cos(r_x , r_y)
  - But it treats missing ratings as "negative"
- Pearson correlation coefficient
  - S_{xy} = items rated by both users x and y

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r_x})(r_{ys} - \bar{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \bar{r_x})^2 (r_{ys} - \bar{r_y})^2}}$$

Jaccard

$$\text{Jaccard}(A, B) =$$

$$\text{Jaccard}(U_i, U_j) =$$

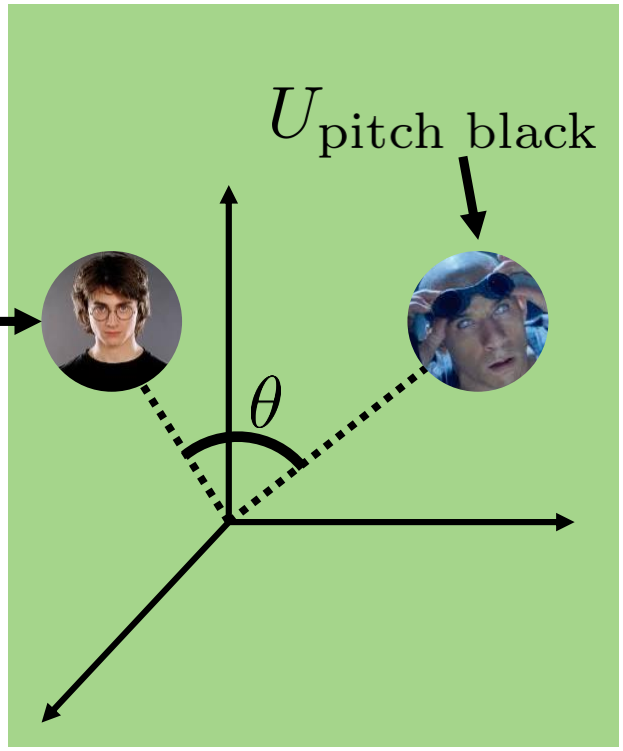
→ Maximum of 1 if the two users purchased **exactly the same** set of items
(or if two items were purchased by the same set of users)

→ Minimum of 0 if the two users purchased **completely disjoint** sets of items
(or if the two items were purchased by completely disjoint sets of users)

Cosine

$U_{\text{harry potter}}$

(vector representation of users who purchased harry potter)



$$\cos(\theta) = 1$$

(theta = 0) → *A* and *B* point in exactly the same direction

$$\cos(\theta) = -1$$

(theta = 180) → *A* and *B* point in opposite directions (won't actually happen for 0/1 vectors)

$$\cos(\theta) = 0$$

(theta = 90) → *A* and *B* are orthogonal

Cosine

Why cosine?

- Unlike Jaccard, works for arbitrary vectors
- E.g. what if we have **opinions** in addition to purchases?

$$R = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 0 & 0 & & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{pmatrix} \rightarrow \begin{pmatrix} -1 & 0 & \cdots & 1 \\ 0 & 0 & & -1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & -1 \end{pmatrix}$$

bought and **liked**

didn't buy

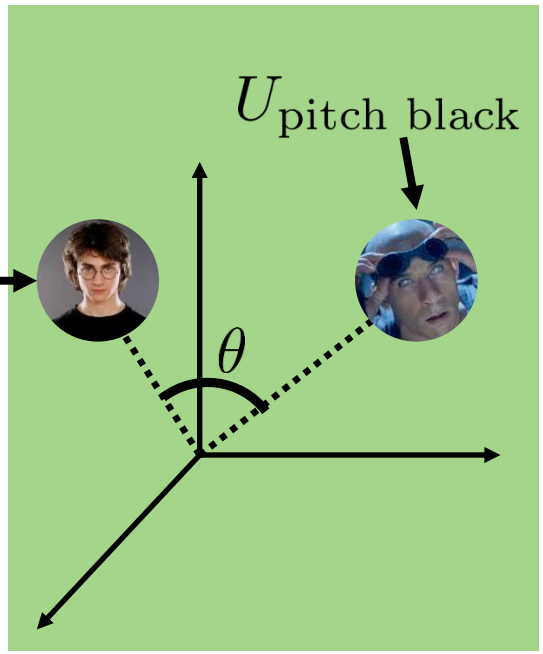
bought and **hated**

Cosine (with ratings)

E.g. our previous example, now with "thumbs-up/thumbs-down" ratings

$U_{\text{harry potter}}$

(vector representation of users' **ratings of** Harry Potter)



$$\cos(\theta) = 1$$

(theta = 0) → Rated by the same users, and they all agree

$$\cos(\theta) = -1$$

(theta = 180) → Rated by the same users, but they **completely disagree** about it

$$\cos(\theta) = 0$$

(theta = 90) → Rated by different sets of users

Pearson correlation

What if we have numerical ratings (rather than just thumbs-up/down)?

$$R = \begin{pmatrix} -1 & 0 & \cdots & 1 \\ 0 & 0 & & -1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & -1 \end{pmatrix} \rightarrow \begin{pmatrix} 4 & 0 & \cdots & 2 \\ 0 & 0 & & 3 \\ \vdots & & \ddots & \vdots \\ 5 & 0 & \cdots & 1 \end{pmatrix}$$

bought and **liked**

didn't buy

bought and **hated**

Pearson correlation

What if we have numerical ratings (rather than just thumbs-up/down)?

- We wouldn't want 1-star ratings to be parallel to 5-star ratings
- So we can subtract the average – values are then **negative** for below-average ratings and **positive** for above-average ratings

items rated by both users

average rating by user *v*

$$\mathrm{Sim}(u,v) = \frac{\sum_{i \in I_u \cap I_v} (R_{u,i} - \bar{R_u})(R_{v,i} - \bar{R_v})}{\sqrt{\sum_{i \in I_u \cap I_v} (R_{u,i} - \bar{R_u})^2 \sum_{i \in I_u \cap I_v} (R_{v,i} - \bar{R_v})^2}}$$

Example

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- Intuitively we want: sim(A,B) > sim(A,C)
- Jaccard: 1/5 < 2/4
- Cosine: 0.386 > 0.322
  - Considers missing ratings as "negative"
  - Solution: subtract the (row) mean

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

- sim(A,B) vs sim (A,C)
  - 0.092 > -0.559

- So far, we can find a rating, but how do we actually generate recommendations?

How to aggregate ratings?

$$r_{c,s} = \operatorname{aggr}_{c' \in \hat{C}} r_{c',s},$$

"similar" users to c

How to aggregate ratings?

$$\text{(a) } r_{c,s} = \frac{1}{N} \sum_{c' \in \hat{C}} r_{c',s},$$

$$\text{(b) } r_{c,s} = k \sum_{c' \in \hat{C}} sim(c, c') \times r_{c',s},$$

$$\text{(c) } r_{c,s} = \bar{r}_c + k \sum_{c' \in \hat{C}} sim(c, c') \times (r_{c',s} - \bar{r}_{c'}),$$

Many other tricks possible

Item-Item Collaborative Filtering

- So far: User-user collaborative filtering
- Another view
  - For item s, find other similar items
  - Estimate rating for item based on ratings for similar items
  - Can use same similarity metrics and prediction functions as in user-user model
- In practice, it has been observed that item-item often works better than user-user

Pros/cons of collaborative filtering

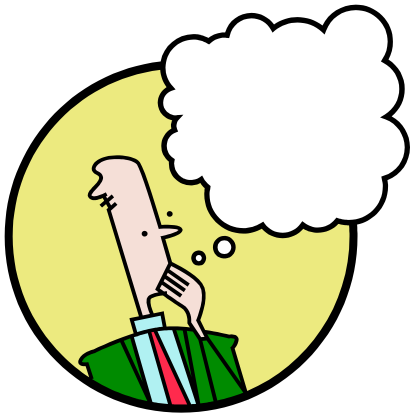
- Works for any kind of item
  - No feature selection needed
- Cold start:
  - Need enough users in the system to find a match
- Sparsity:
  - The user/ratings matrix is sparse
  - Hard to find users that have rated the same items
- First rater:
  - Cannot recommend an item that has not been previously rated
  - New items, esoteric items
- Popularity bias:
  - Cannot recommend items to someone with unique taste
  - Tends to recommend popular items

Content-based recommendations

Content-based recommendations

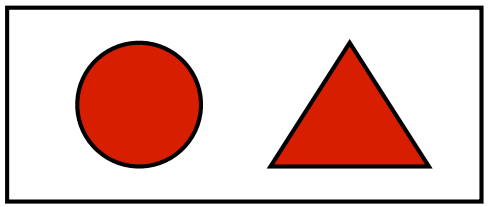
- **Main idea**: recommend items to customer ***x*** similar to previous items rated highly by ***x***
- Movie recommendations
  - recommend movies with same actor(s), director, genre, …
- Websites, blogs, news
  - recommend other sites with "similar" content

Plan of action



likes

Item profiles

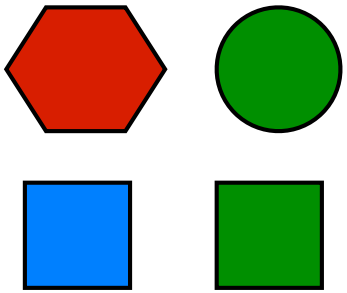


build

Red
Circles
Triangles

User profile

match



recommend

Item Profiles

- For each item, create an item profile
- Profile is a set of features (vectors!)
  - movies: author, title, actor, director,…
  - text: set of "important" words in document
- How to pick important words?
  - Usual heuristic is TF.IDF

User profiles and prediction

- User profile possibilities:
  - Weighted average of rated item profiles
  - Variation: weight by difference from average rating for item
  - …
- Prediction heuristic
  - Given user profile **x** and item profile **i**, estimate
    - $u(\mathbf{x},\mathbf{i}) = \cos(\mathbf{x},\mathbf{i}) = \mathbf{x}.\mathbf{i}/(|\mathbf{x}||\mathbf{i}|)$

Advantages of Content-based Recs?

- No need for data on other users
  - No cold-start or sparsity problems
- Able to recommend to users with unique tastes
- Able to recommend new and unpopular items
  - No first-rater problem
- Can provide explanations of recommended items by listing content-features that caused item to be recommended

Limitations of content-based approach

- Finding the appropriate features
  - e.g., images, movies, music
- Recommendations for new users
  - How to build a profile?
- Overspecialization
  - Never recommends items outside user's content profile
  - People might have multiple interests
  - Unable to exploit quality judgments of other users

Hybrid: Content + Collaborative

Hybrid Methods

- Implement two separate recommenders and combine predictions
- Add content-based methods to collaborative filtering
  - item profiles for new item problem
  - demographics to deal with new user problem

Recommendation Approach	Recommendation Technique	
	Heuristic-based	Model-based
Content-based	Commonly used techniques: • TF-IDF (information retrieval) • Clustering Representative research examples: • Lang 1995 • Balabanovic & Shoham 1997 • Pazzani & Billsus 1997	Commonly used techniques: • Bayesian classifiers • Clustering • Decision trees • Artificial neural networks Representative research examples: • Pazzani & Billsus 1997 • Mooney et al. 1998 • Mooney & Roy 1999 • Billsus & Pazzani 1999, 2000 • Zhang et al. 2002
Collaborative	Commonly used techniques: • Nearest neighbor (cosine, correlation) • Clustering • Graph theory Representative research examples: • Resnick et al. 1994 • Hill et al. 1995 • Shardanand & Maes 1995 • Breese et al. 1998 • Nakamura & Abe 1998 • Aggarwal et al. 1999 • Delgado & Ishii 1999 • Pennock & Horwitz 1999 • Sarwar et al. 2001	Commonly used techniques: • Bayesian networks • Clustering • Artificial neural networks • Linear regression • Probablistic models Representative research examples: • Billsus & Pazzani 1998 • Breese et al. 1998 • Ungar & Foster 1998 • Chien & George 1999 • Getoor & Sahami 1999 • Pennock & Horwitz 1999 • Goldberg et al. 2001 • Kumar et al. 2001 • Pavlov & Pennock 2002 • Shani et al. 2002 • Yu et al. 2002, 2004 • Hofmann 2003, 2004 • Marlin 2003 • Si & Jin 2003
Hybrid	Combining content-based and collaborative components using: • Linear combination of predicted ratings • Various voting schemes • Incorporating one component as a part of the heuristic for the other Representative research examples: • Balabanovic & Shoham 1997 • Claypool et al. 1999 • Good et al. 1999 • Pazzani 1999 • Billsus & Pazzani 2000 • Tran & Cohen 2000 • Melville et al. 2002	Combining content-based and collaborative components by: • Incorporating one component as a part of the model for the other • Building one unifying model Representative research examples: • Basu et al. 1998 • Condliff et al. 1999 • Soboroff & Nicholas 1999 • Ansari et al. 2000 • Popescul et al. 2001 • Schein et al. 2002

Netflix Prize

The Netflix Prize

- Training data
  - 100 million ratings
  - 480,000 users
  - 17,770 movies
  - 6 years of data: 2000-2005
- Test data
  - Last few ratings of each user (2.8 million)
  - Evaluation criterion: root mean squared error (RMSE)
  - Netflix Cinematch system RMSE: 0.9514
- Competition
  - 2700+ teams
  - $1 million grand prize for 10% improvement over Netflix

RMSE

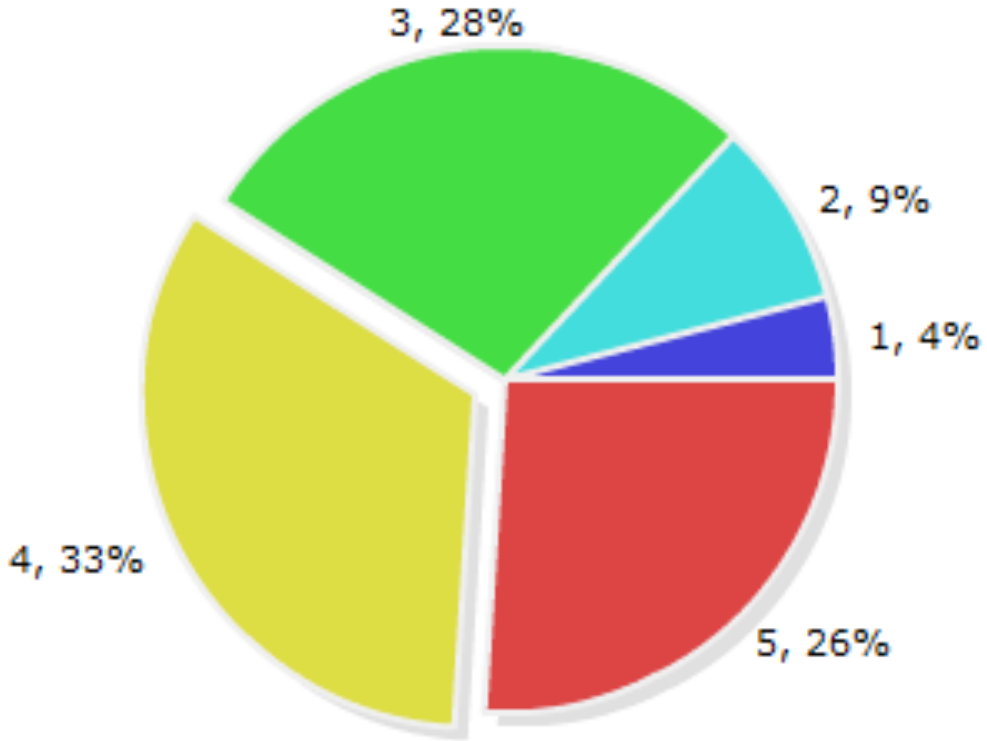
- Compare predictions with known ratings

- My system predicted you would rate
  - The Shawshank Redemption as **4.3 stars**
    - In reality, you gave it **5 stars**
  - The Matrix with **3.9 stars**
    - In reality, you gave it **4 stars**

- RMSE = sqrt(1/2 * (4.3 - 5)^2 + (3.9 - 4)^2))

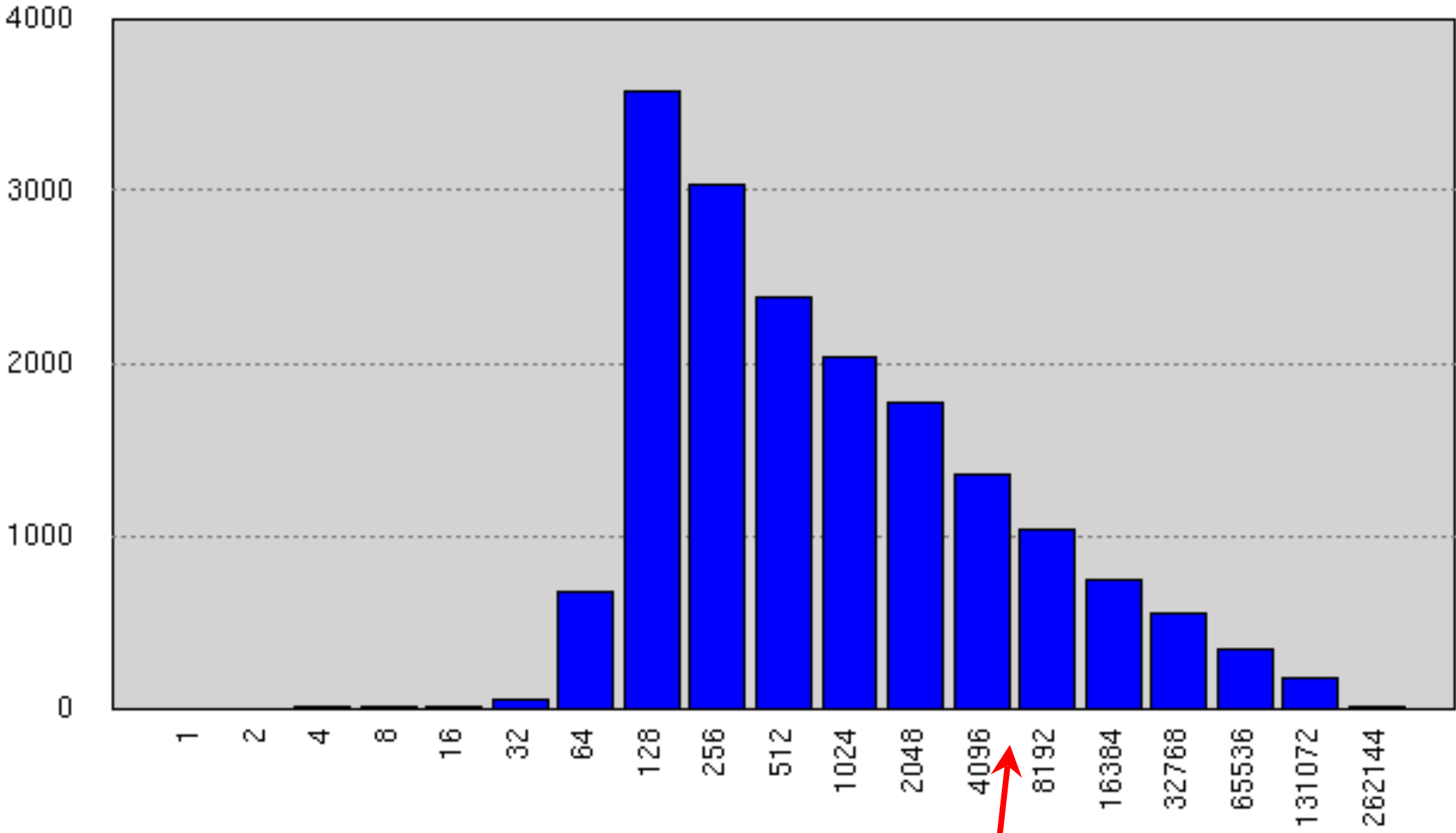
$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Overall rating distribution



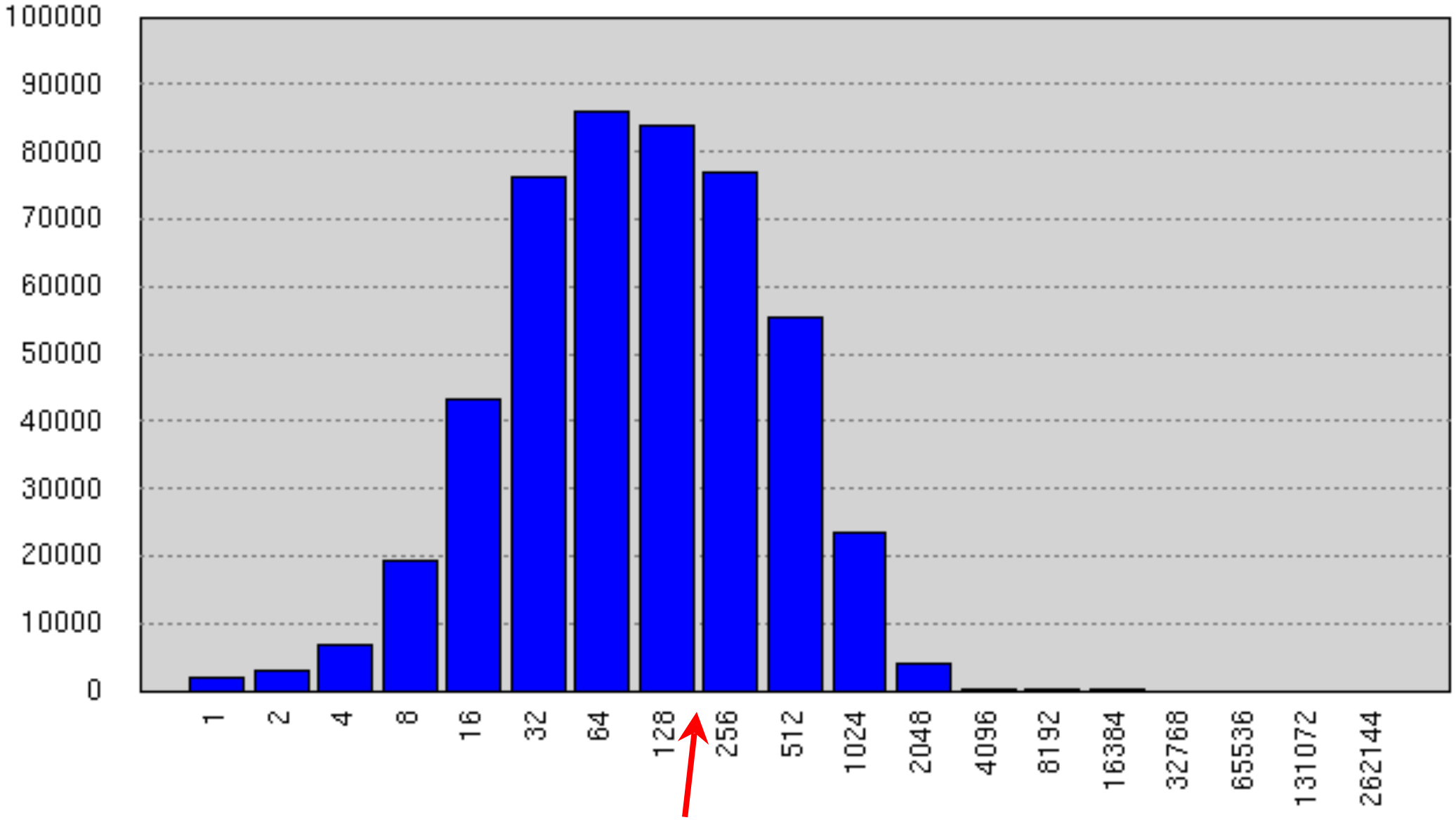
- Third of ratings are 4s
- Average rating is 3.68

#ratings per movie



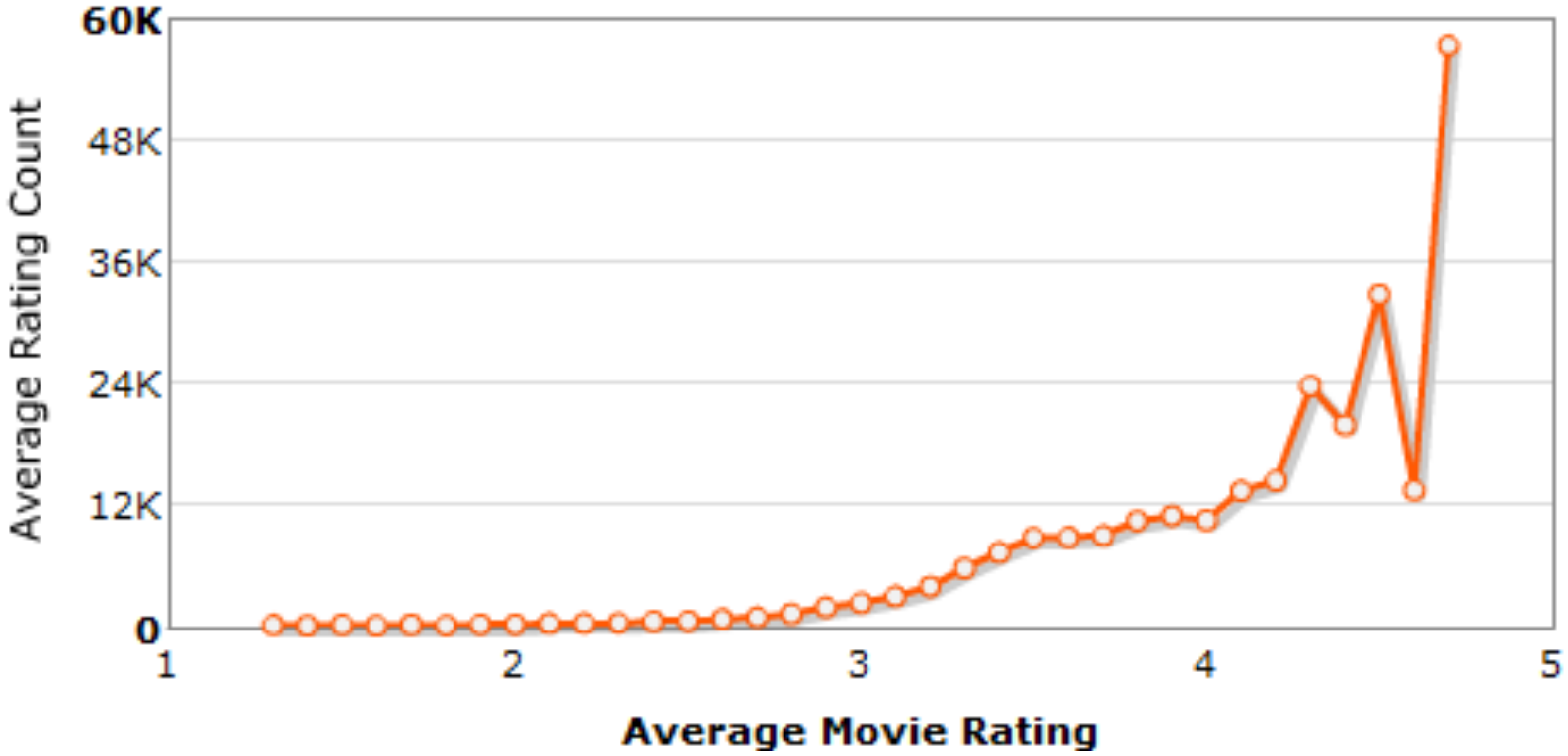
- Avg #ratings/movie: 5627

#ratings per user



- Avg #ratings/user: 208

Average movie rating by movie count



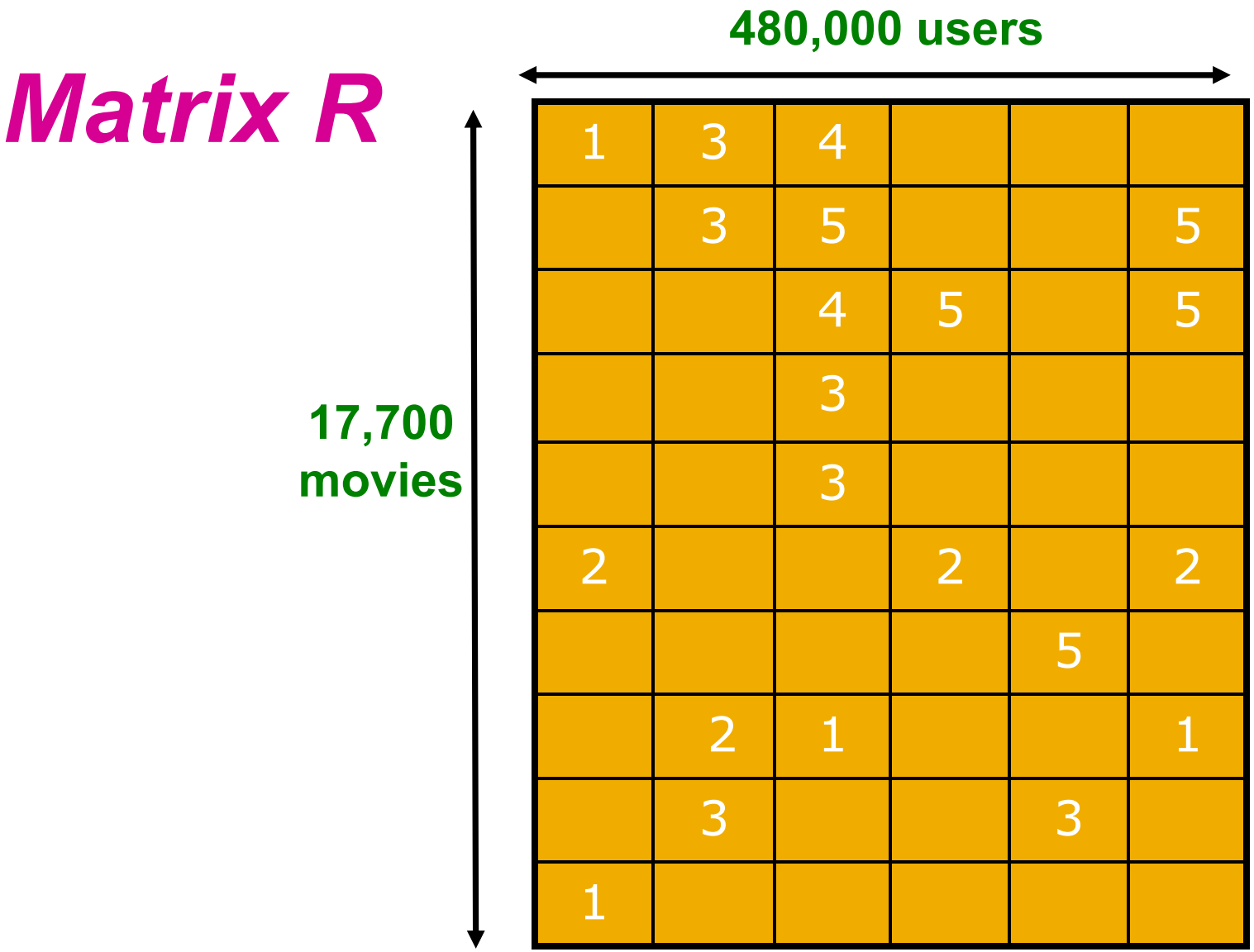
- More ratings to better movies

From TimelyDevelopment.com

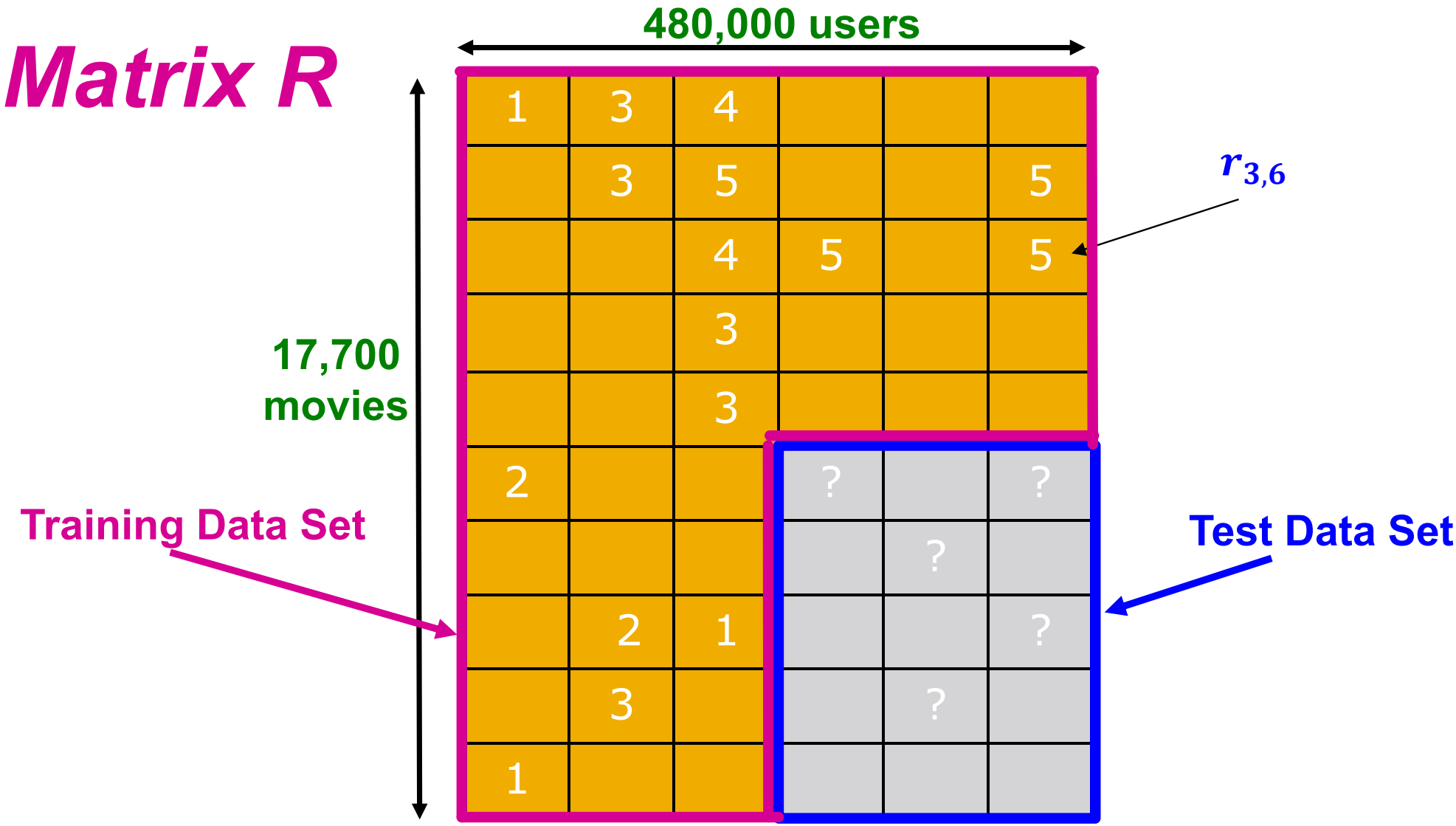
Most loved movies

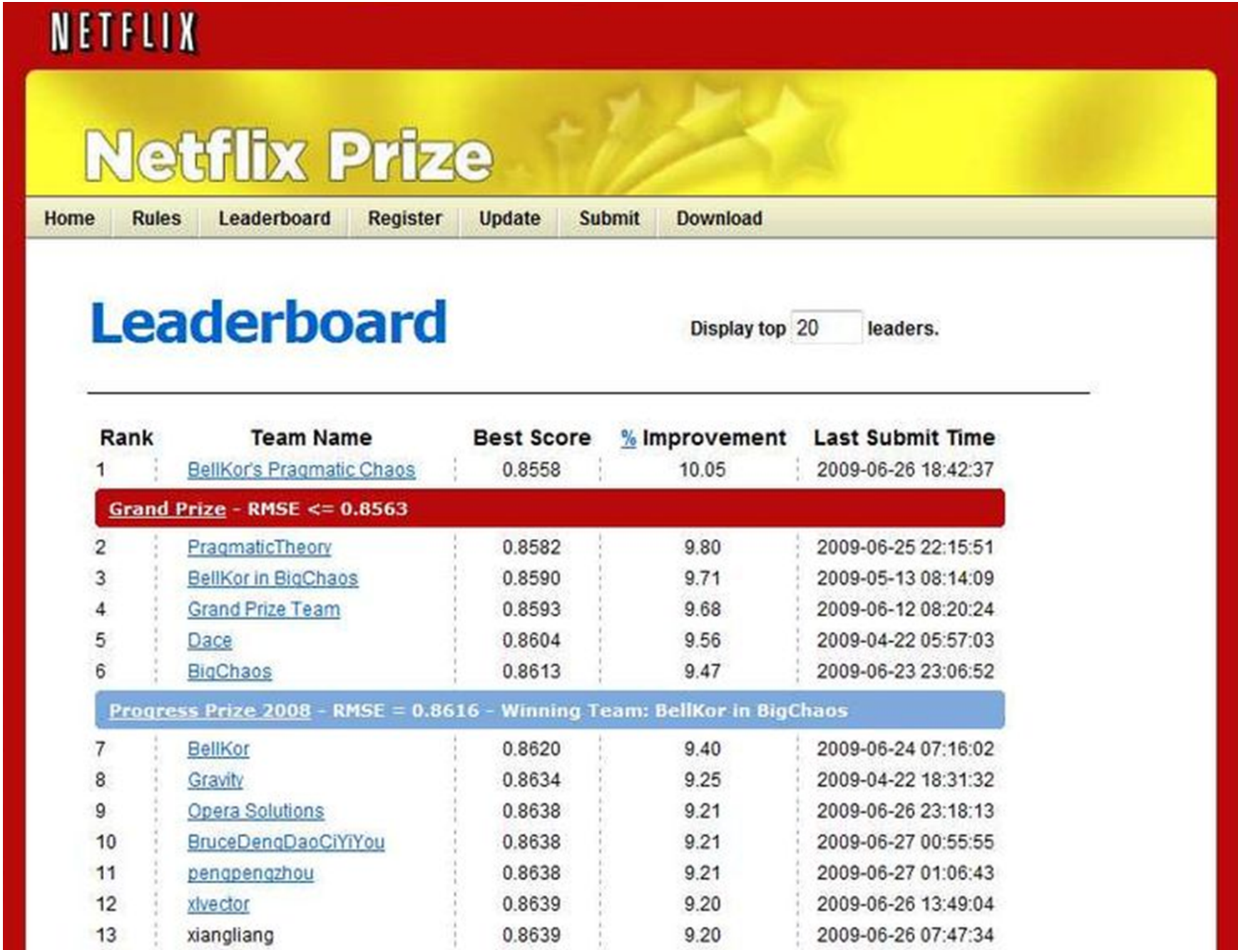
Title	Avg rating	Count
The Shawshank Redemption	4.593	137812
Lord of the Rings: The Return of the King	4.545	133597
The Green Mile	4.306	180883
Lord of the Rings: The Two Towers	4.460	150676
Finding Nemo	4.415	139050
Raiders of the Lost Ark	4.504	117456
Forrest Gump	4.299	180736
Lord of the Rings: The Fellowship of the ring	4.433	147932
The Sixth Sense	4.325	149199
Indiana Jones and the Last Crusade	4.333	144027

The Netflix Utility Matrix R



Utility Matrix R: Evaluation





June 26th submission triggers 30-day "last call"

- **Ensemble team formed**
  - Group of other teams on leaderboard forms a new team
  - Relies on combining their models
  - Quickly also get a qualifying score over 10%
- **BellKor**
  - Continue to get small improvements in their scores
  - Realize that they are in direct competition with Ensemble
- **Strategy**
  - Both teams carefully monitoring the leaderboard
  - Only sure way to check for improvement is to submit a set of predictions
    - This alerts the other team of your latest score

- **Submissions limited to 1 a day**
  - Only 1 final submission could be made in the last 24h
- **24 hours before deadline…**
  - **BellKor** team member in Austria notices (by chance) that **Ensemble** posts a score that is slightly better than BellKor's
- **Frantic last 24 hours for both teams**
  - Much computer time on final optimization
  - Carefully calibrated to end about an hour before deadline
- **Final submissions**
  - **BellKor** submits a little early (on purpose), 40 mins before deadline
  - **Ensemble** submits their final entry 20 mins later
  - **….and everyone waits….**

Netflix Prize

COMPLETED

Home | Rules | Leaderboard | Update | Download

Leaderboard

Showing Test Score. Click here to show quiz score

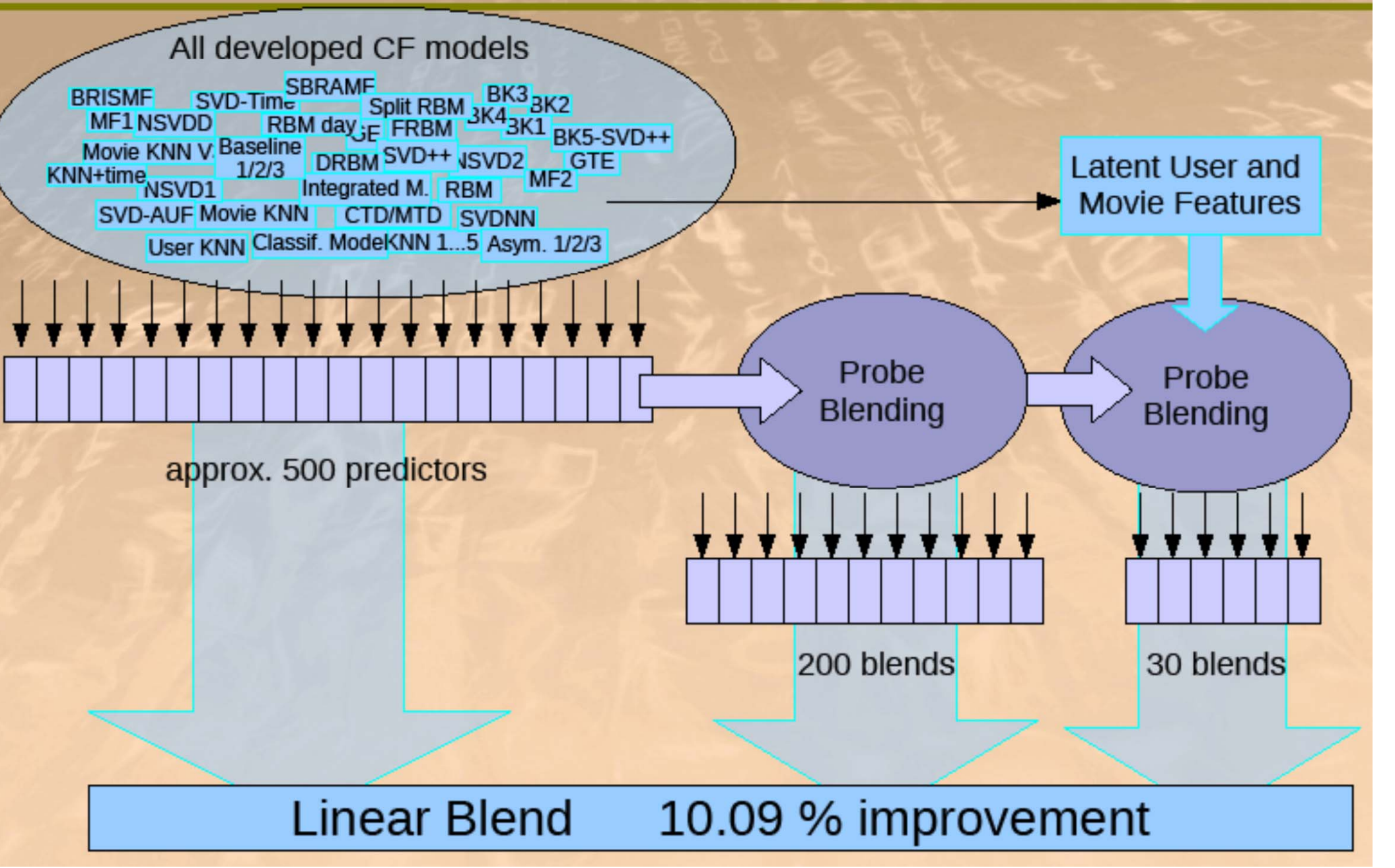
Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	[illegible]	[illegible]	[illegible]
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8642	9.27	2009-07-15 14:53:22
14	Gravity	0.8643	9.26	2009-04-22 18:31:32
15	Ces	0.8651	9.18	2009-06-21 19:24:53
16	Invisible Ideas	0.8653	9.15	2009-07-15 15:53:04
17	Just a guy in a garage	0.8662	9.06	2009-05-24 10:02:54
18	J Dennis Su	0.8666	9.02	2009-03-07 17:16:17
19	Craig Carmichael	0.8666	9.02	2009-07-25 16:00:54


2009
NETFLIX
DATE 09·21·09
PAY TO THE ORDER OF BellKor's Pragmatic Chaos
$1,000,000 00
AMOUNT ONE MILLION
00/100
FOR The Netflix Prize
Reed Hastings

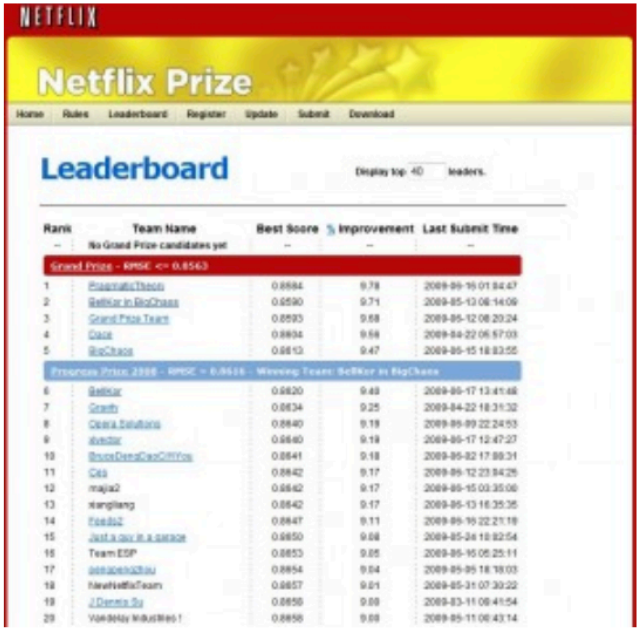
The big picture

Solution of BellKor's Pragmatic Chaos



CASEY JOHNSTON, ARS TECHNICA BUSINESS 04.16.12 08:20 AM

NETFLIX NEVER USED ITS $1 MILLION ALGORITHM DUE TO ENGINEERING COSTS



Netflix awarded a $1 million prize to a developer team in 2009 for an algorithm that increased the accuracy of the company's recommendation engine by 10 percent. But it doesn't use the million-dollar code, and has no plans to implement it in the future, Netflix announced on its blog

What the Failed $1M Netflix Prize Says About Business Advice

Ryan Holiday, CONTRIBUTOR

An inside look at the hidden side of marketing, PR and strategy. FULL BIO

Opinions expressed by Forbes Contributors are their own.

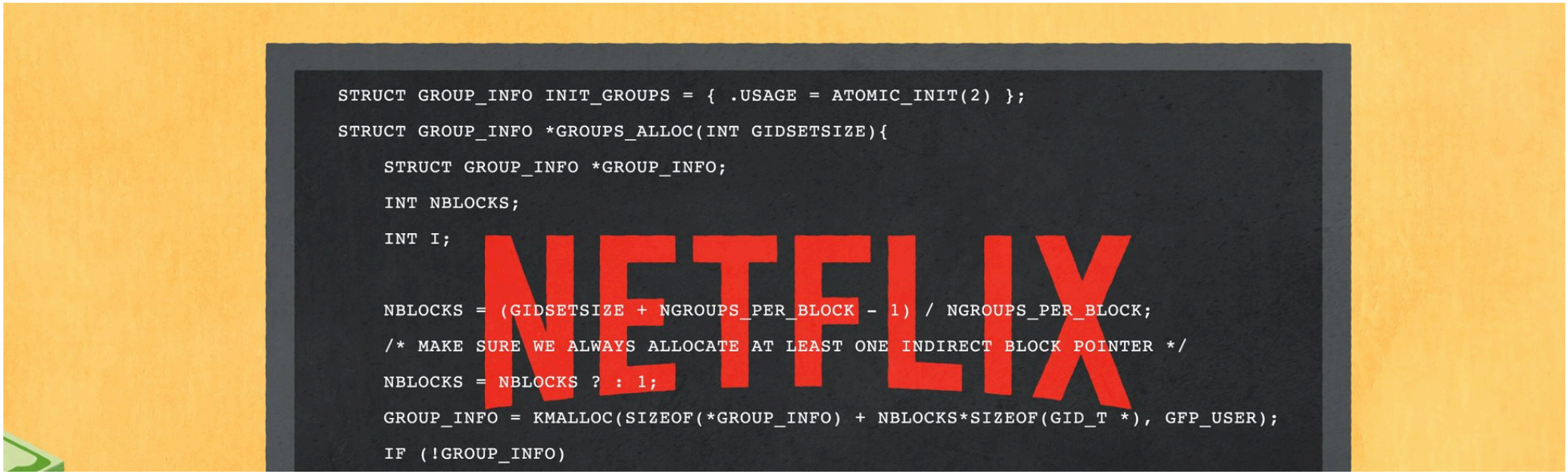
Netflix (Photo credit: Wikipedia)

If you've picked up a business book in the last four years, you undoubtedly heard it: the endless championing of Netflix's $1 Million Prize.

It made for great "the future of business" fodder. A technology company crowdsources its R&D by ponying up $1,000,000 to anyone who can improve their recommendation engine. Netflix, stickler for user-experience as it is, was willing to pay all this money even if it meant making their algorithm just 10% better. Amazing! If you were looking for an anecdote to cheerlead Web 2.0 to, this was almost too good to be true.

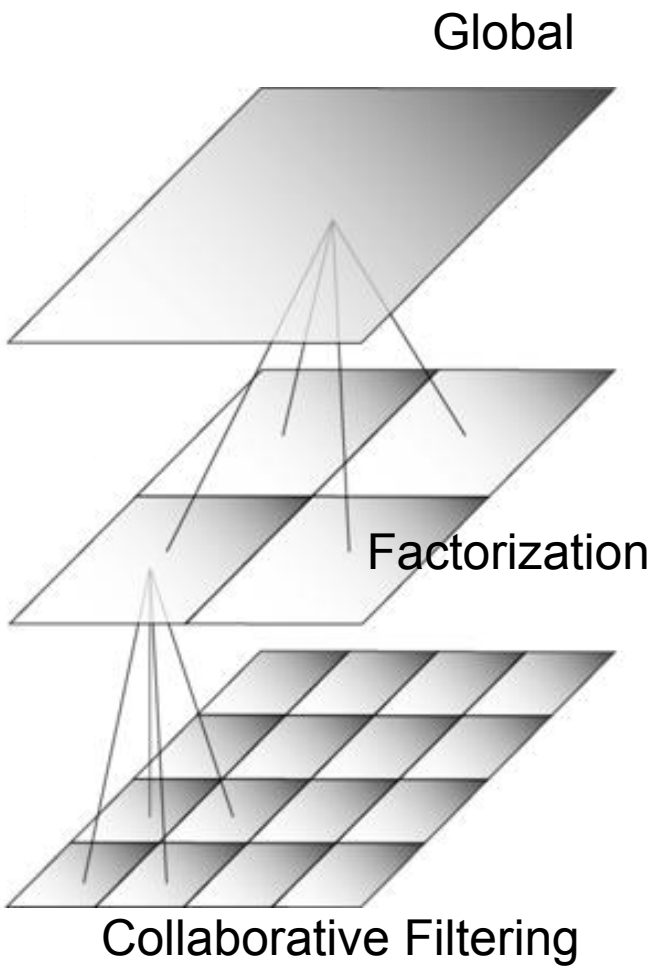
THE NETFLIX PRIZE: HOW A $1 MILLION CONTEST CHANGED BINGE-WATCHING FOREVER

By DAN JACKSON
Published On 07/07/2017
@danielvjackson



BellKor Recommender System

- The winner of the Netflix Challenge
- Multi-scale modeling of the data
  - **Global:**
    - Overall deviations of users/movies
  - **Factorization:**
    - Addressing "regional" effects
  - **Collaborative Filtering:**
    - Extract local patterns



Modeling Local and Global Effects

- **Global:**
  - Mean movie rating: **3.7 stars**
  - *The Sixth Sense* is **0.5** stars above average
  - Joe rates **0.2** stars below average
  - => Baseline estimate: Joe will rate *The Sixth Sense* **4 stars**
- **Local Neighborhood (Collaborative Filtering)**
  - Joe didn't like related movie *Signs*
  - =>Final estimate: Joe will rate *The Sixth Sense* **3.8 stars**

Recap: Collaborative Filtering

- **Earliest and most popular collaborative filtering method**
- Derive unknown ratings from those of "**similar**" movies (item-item variant)
- Define **similarity measure s_{ij}** of items ***i*** and ***j***
- Select ***k***-nearest neighbors, compute the rating
  - ***N(i; x):*** items most similar to ***i*** that were rated by ***x***

$$\hat{r}_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij} similarity of items ***i*** and ***j***
r_{xj} rating of user ***x*** on item ***j***
N(i;x) set of items similar to item ***i*** that were rated by ***x***

Modeling Local and Global Effects

- **In practice we get better estimates if we model deviations:**

$$\hat{r}_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$

baseline estimate for r_{xi}

$$b_{xi} = \mu + b_x + b_i$$

μ = overall mean rating
b_x = rating deviation of user x
= (*avg. rating of* user x) – μ
b_i = (*avg. rating of* movie i) – μ

Problems/Issues:

1) Similarity measures are "arbitrary

2) Pairwise similarities neglect interdependencies among users

3) Taking a weighted average can be restricting

Solution: Instead of s_{ij} use w_{ij} that we estimate directly from data