Information Retrieval

CS 547/DS 547

Worcester Polytechnic Institute

Department of Computer Science

Instructor: Prof. Kyumin Lee

Upcoming Schedule

- March 1: Midterm Exam
- March 3: due date of HW3
- March 17: due date of project proposal
- March 21: due date of proposal presentation slides

Midterm

- The exam will be held at 6pm next Wednesday in class.
- The exam is closed book.
- You may prepare and use one standard 8.5" by 11" piece of paper with any notes you think appropriate or significant.
- You may use a calculator if it make you feel comfortable. But no other electronic devices are allowed (e.g., cell phone, tablet and computer).

Previous Class…

Boolean Retrieval Model

Previous Class…

Boolean Retrieval Model

Inverted index

Previous Class…

Query Optimization

Previous Class…

Query Optimization

Preprocessing Documents
→Tokenization, Normalization, Stemming, Stop words

Previous Class…

Query Optimization

Preprocessing Documents
→Tokenization, Normalization, Stemming, Stop words

Skip pointers & Positional index

Previous Class…

Wild-card queries
→ Permuterm Index

Previous Class…

TF and IDF

Previous Class...

TF and IDF

tf-idf weighting

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$$

Previous Class…

Cosine Similarity

Previous Class…

Cosine Similarity

Computing scores in a complete search system

Previous Class…

Statistical Language Models

Crawler

Previous Class…

Statistical Language Models

Crawler

Web APIs

Previous Class…

PageRank

Evaluation

Recommenders

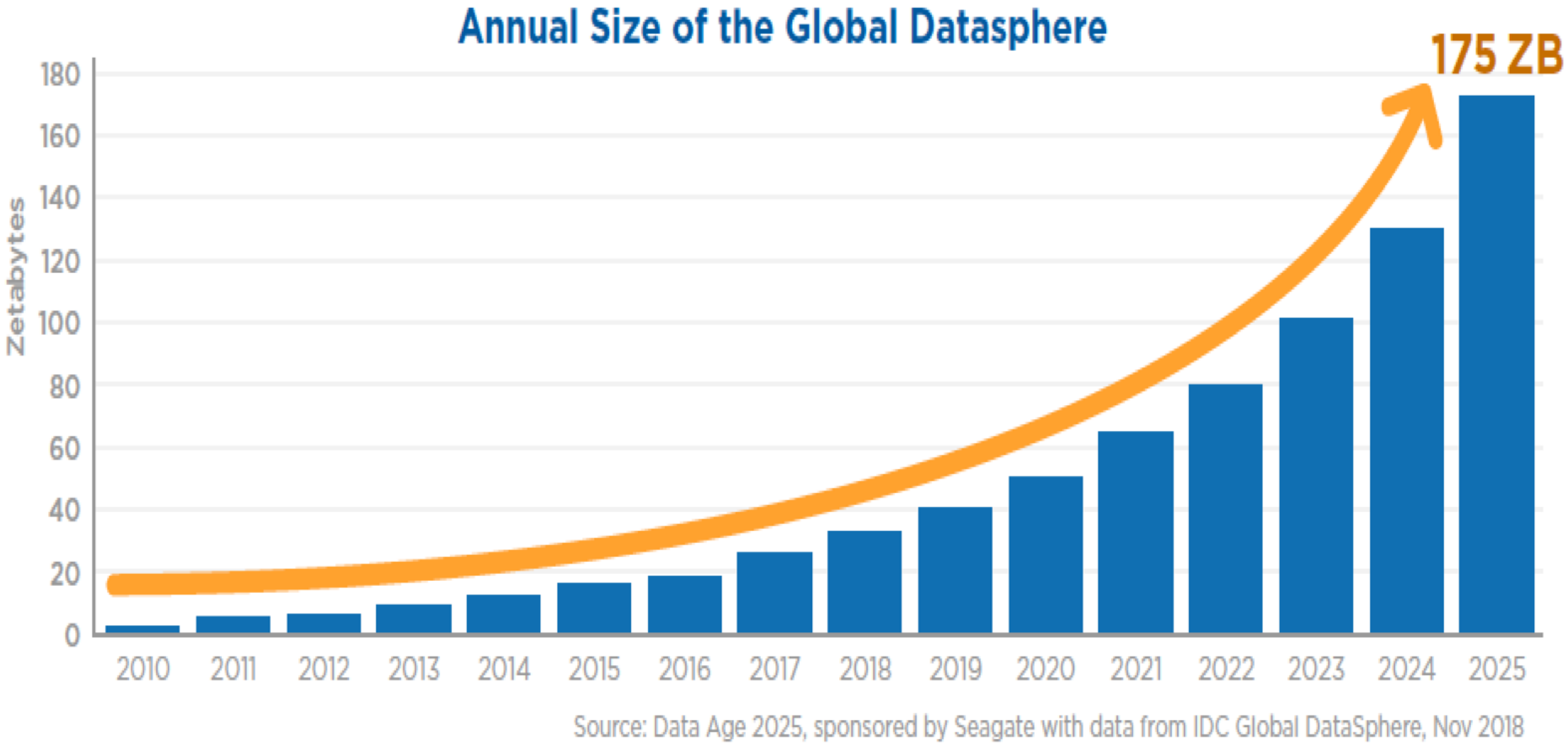


Image source: https://www.forbes.com/sites/tomcoughlin/2018/11/27/175-zettabytes-by-2025/?sh=49acded65459

In 2022

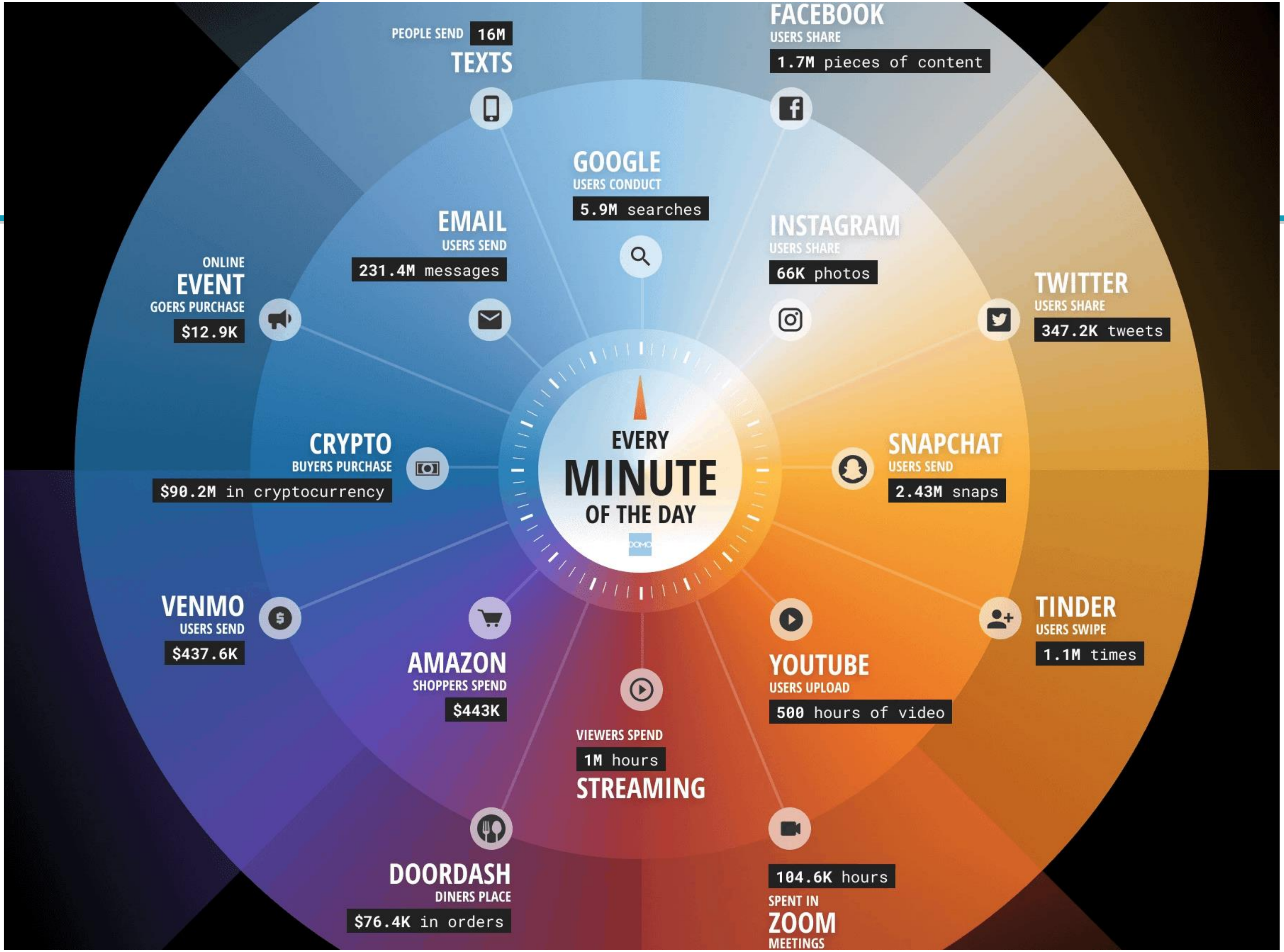


Image source: https://www.digitalinformationworld.com/2022/09/new-report-shares-mind-blowing-amount.html

How Many Products Does Amazon Sell?

amazon.com

Amazon itself sells over 12 million products. If you take into account all products sold on the Amazon marketplace by third-party sellers, that number rises to more than 353 million products.

How Many Orders Does Amazon Get A Day?

Amazon ships approximately **1.6 million packages a day**.

That works out to more than **66 thousand orders per hour**, and **18.5 orders per second.**

Amazon Sales Statistics: How Much Amazon Makes in a Day

In 2019, Amazon made $141.25 billion in retail product sales. This comes out to an average of $385 million **each day.**

https://landingcube.com/amazon-statistics/

About Spotify

Spotify transformed music listening forever when it launched in 2008. Discover, manage and share over 100 million tracks and 5 million podcasts titles, for free, or upgrade to Spotify Premium to access exclusive features for music including improved sound quality and an on-demand, offline, and ad-free music listening experience. Today, Spotify is the world's most popular audio streaming subscription service with 489 million users, including 205 million subscribers in more than 180 markets.

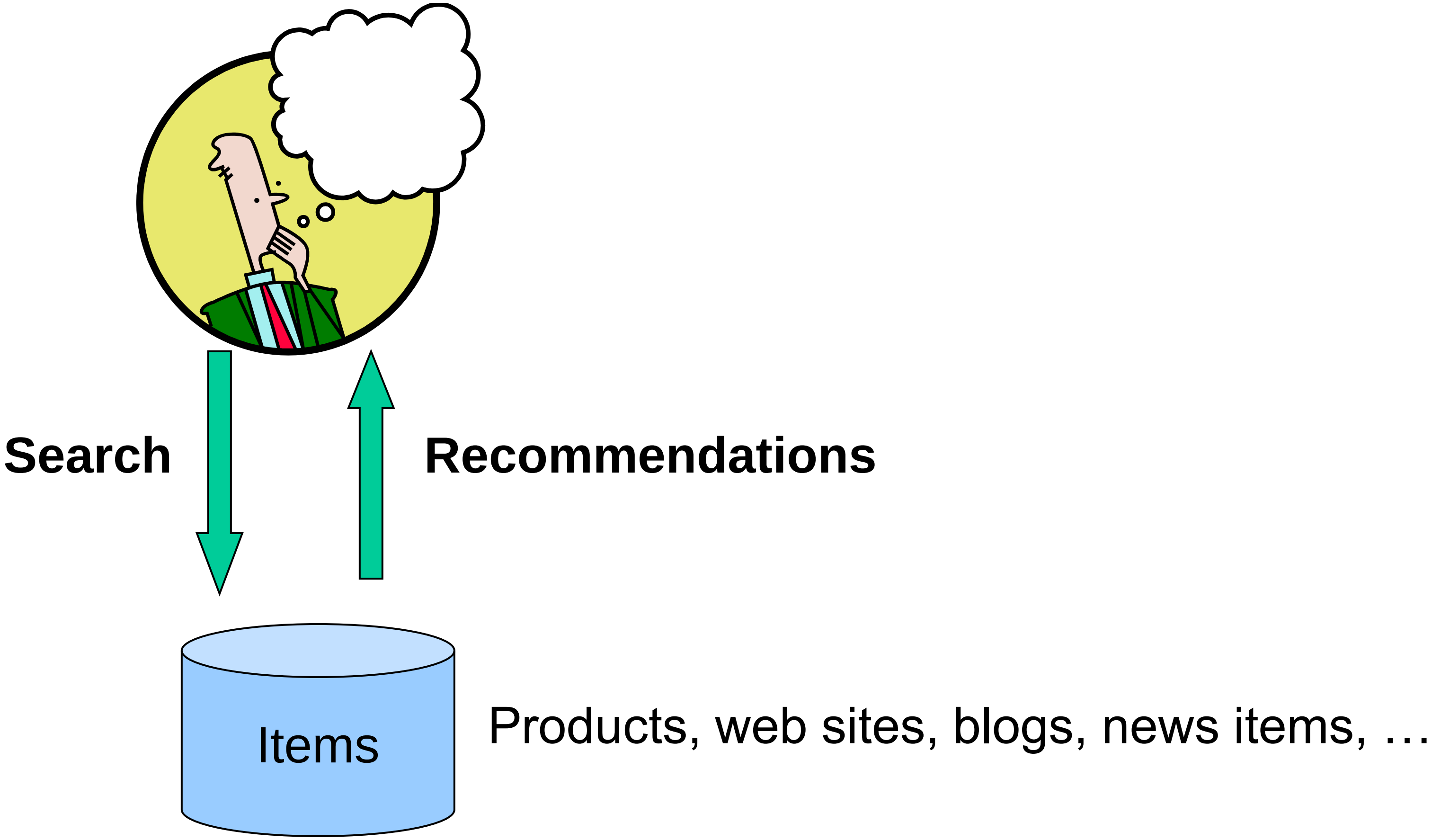
YouTube Usage Statistics

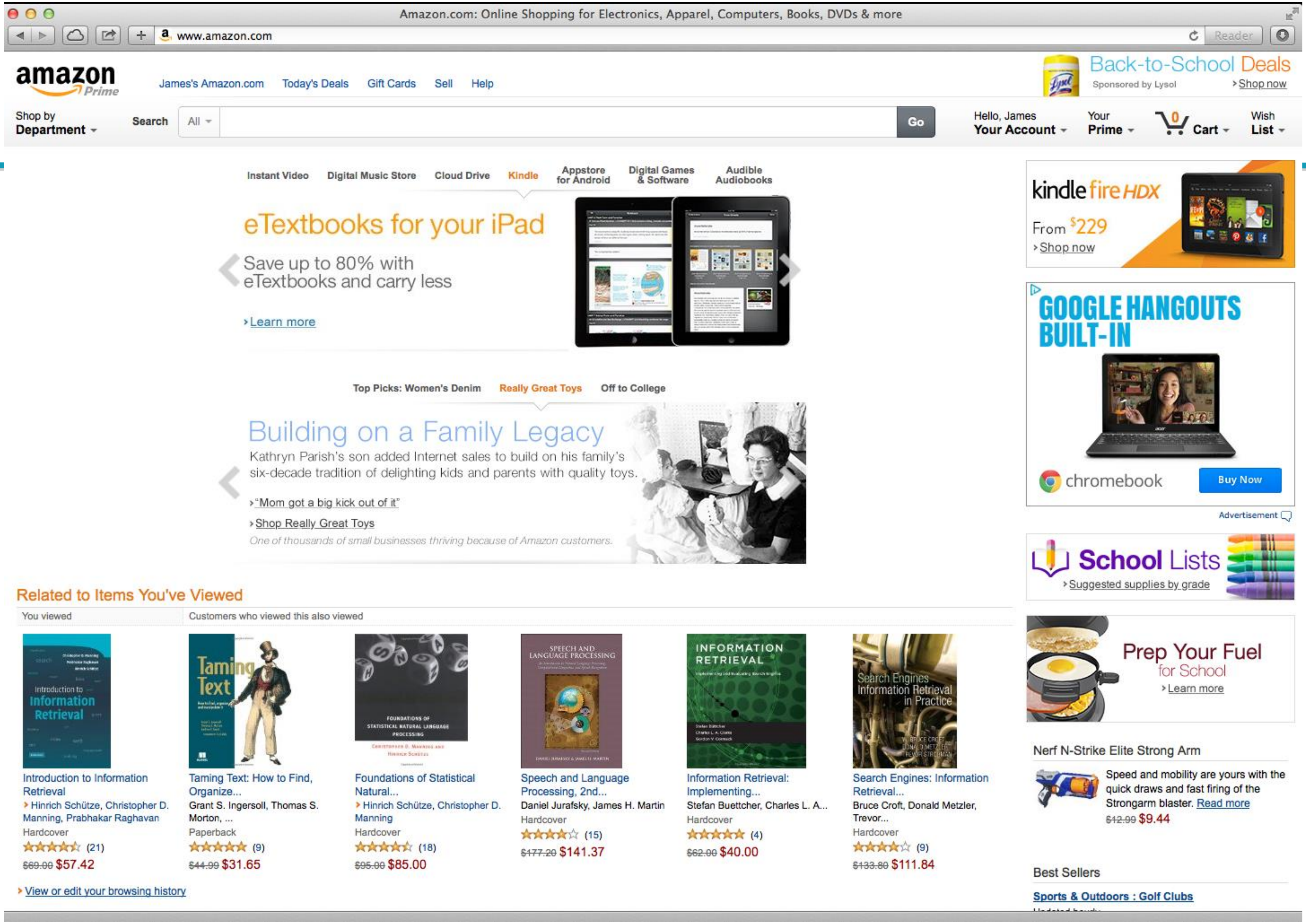
YouTube users consume 1 billion hours' worth of video every day. This figure translates to approximately 5 billion videos every day.

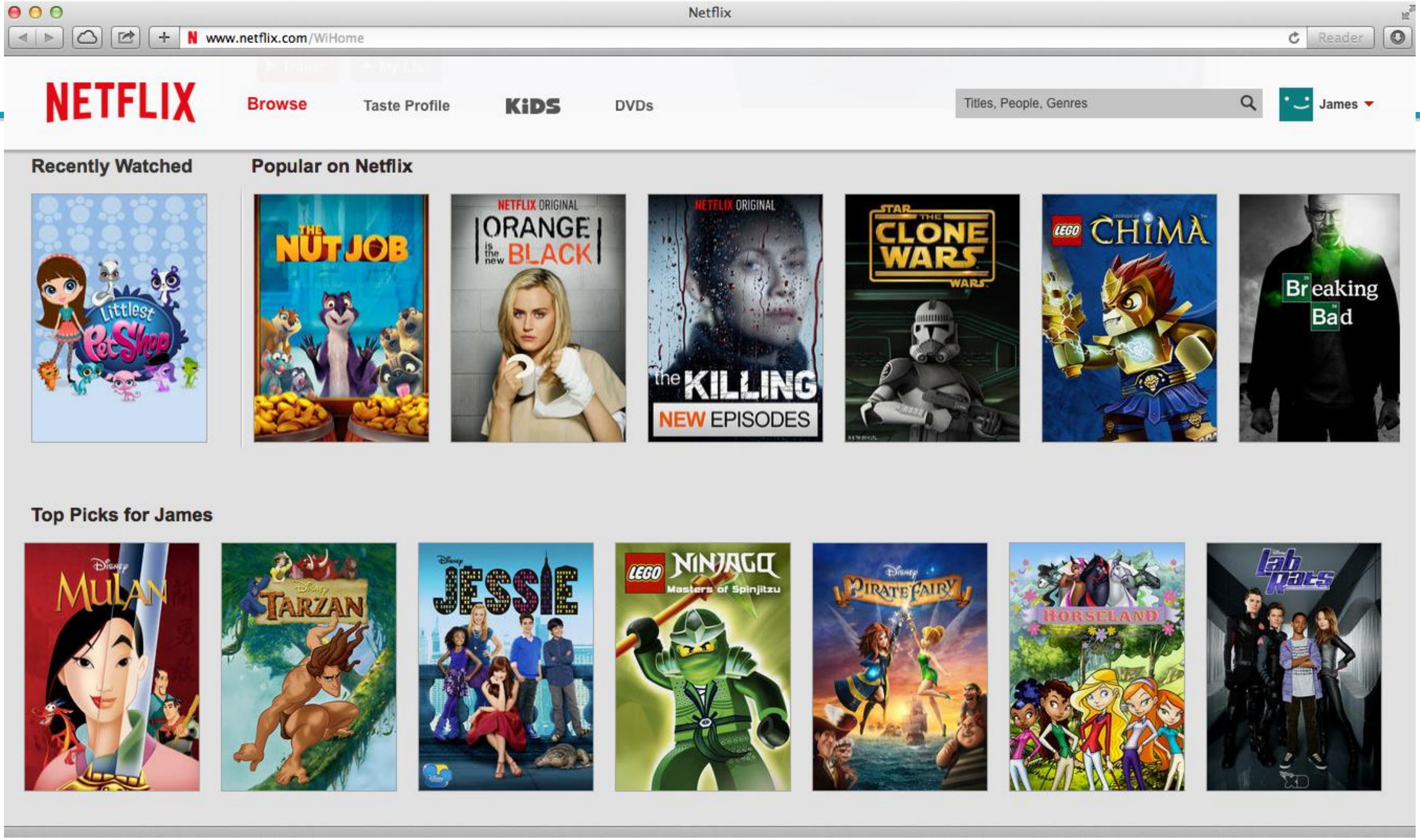
- The average time spent on YouTube by a user is 11.24 minutes.
- It is estimated that YouTube has around 5 billion videos.
- 500 hours of video is uploaded to YouTube every minute.

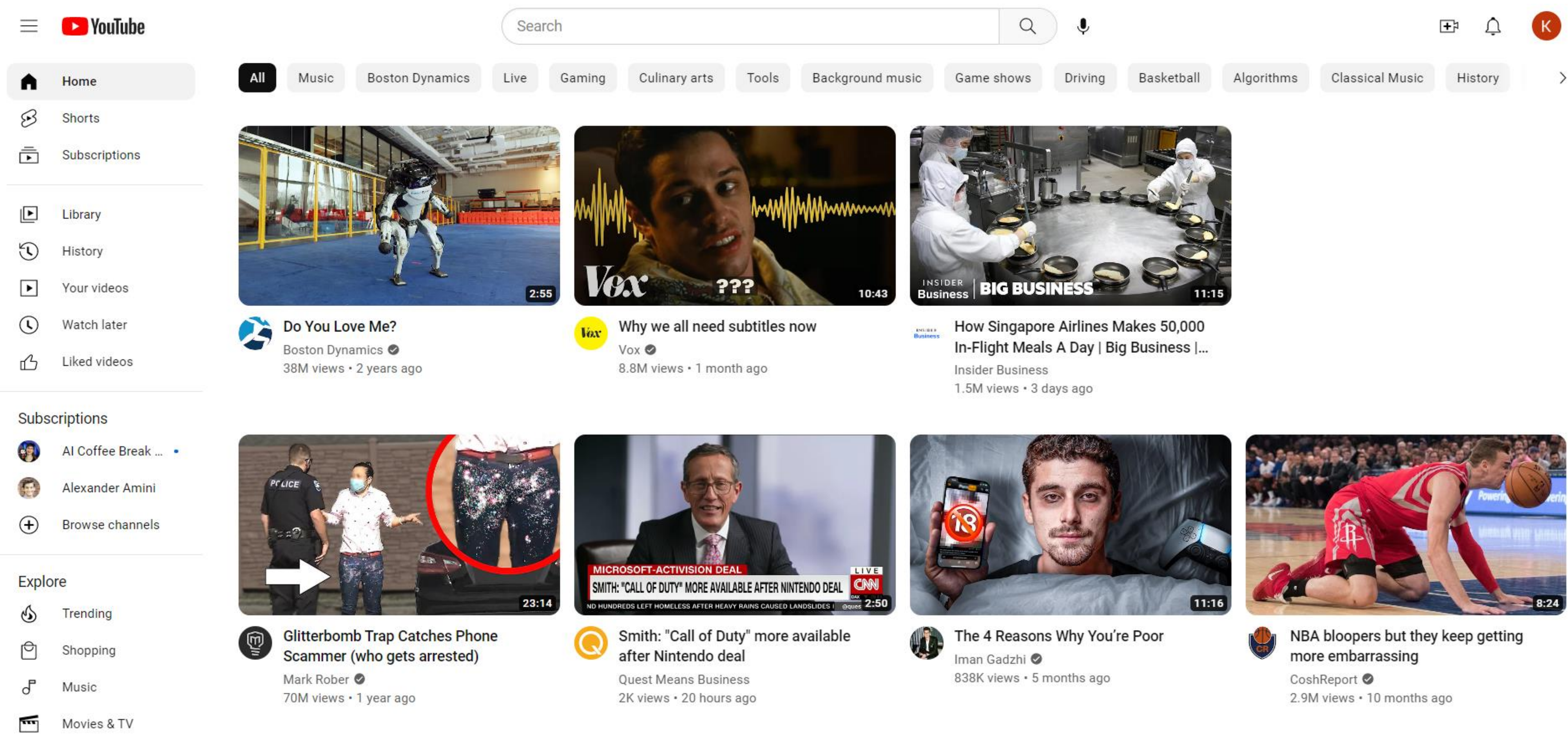
https://www.demandsage.com/youtube-stats/

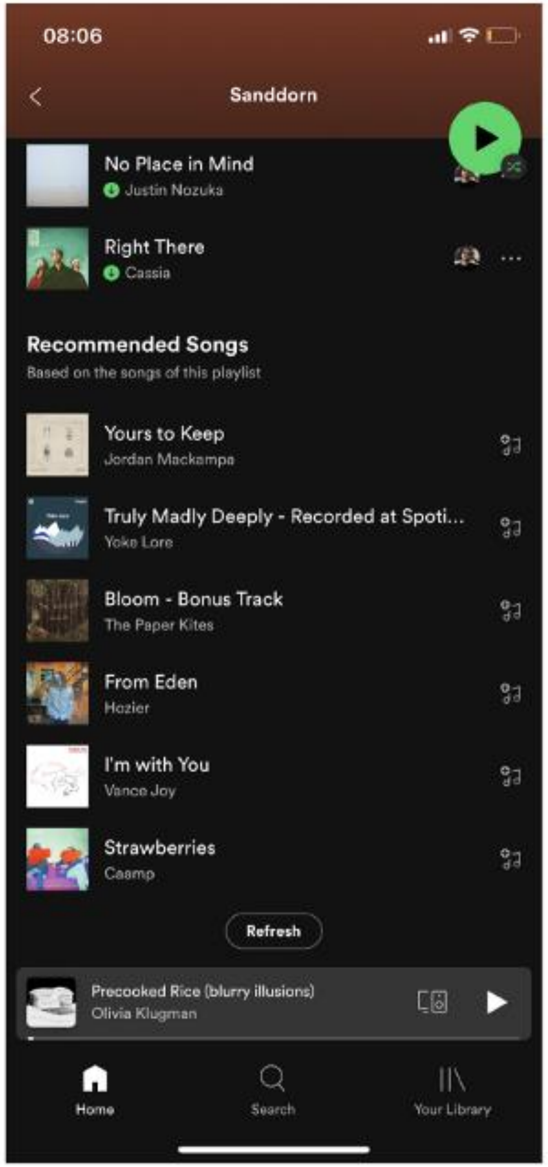
Recommendations



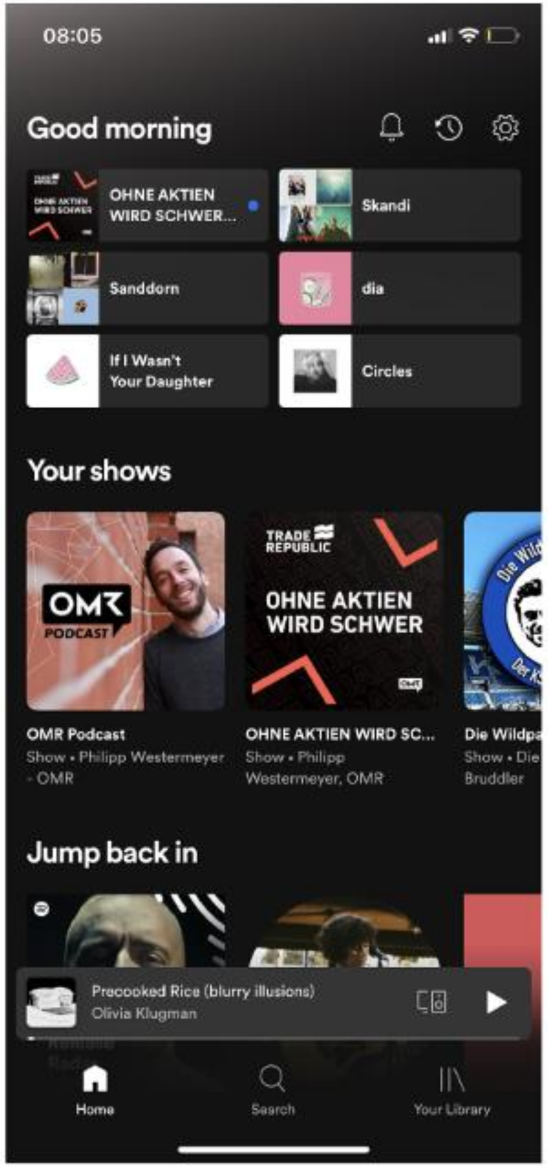

Amazon.com: Online Shopping for Electronics, Apparel, Computers, Books, DVDs & more
www.amazon.com
Reader
amazon Prime
James's Amazon.com
Today's Deals
Gift Cards
Sell
Help
Back-to-School Deals
Sponsored by Lysol
Shop now
Shop by Department
Search
All
Go
Hello, James Your Account
Your Prime
0 Cart
Wish List
Instant Video
Digital Music Store
Cloud Drive
Kindle
Appstore for Android
Digital Games & Software
Audible Audiobooks
eTextbooks for your iPad
Save up to 80% with eTextbooks and carry less
Learn more
kindle fire HDX
From $229
Shop now
GOOGLE HANGOUTS BUILT-IN
chromebook
Buy Now
Advertisement
Top Picks: Women's Denim
Really Great Toys
Off to College
Building on a Family Legacy
Kathryn Parish's son added Internet sales to build on his family's six-decade tradition of delighting kids and parents with quality toys.
"Mom got a big kick out of it"
Shop Really Great Toys
One of thousands of small businesses thriving because of Amazon customers.
School Lists
Suggested supplies by grade
Prep Your Fuel for School
Learn more
Related to Items You've Viewed
You viewed
Customers who viewed this also viewed
Introduction to Information Retrieval
Hinrich Schütze, Christopher D. Manning, Prabhakar Raghavan
Hardcover
(21)
$69.00 $57.42
Taming Text: How to Find, Organize...
Grant S. Ingersoll, Thomas S. Morton, ...
Paperback
(9)
$44.99 $31.65
Foundations of Statistical Natural...
Hinrich Schütze, Christopher D. Manning
Hardcover
(18)
$95.00 $85.00
Speech and Language Processing, 2nd...
Daniel Jurafsky, James H. Martin
Hardcover
(15)
$177.20 $141.37
Information Retrieval: Implementing...
Stefan Buettcher, Charles L. A...
Hardcover
(4)
$62.00 $40.00
Search Engines: Information Retrieval...
Bruce Croft, Donald Metzler, Trevor...
Hardcover
(9)
$133.80 $111.84
View or edit your browsing history
Nerf N-Strike Elite Strong Arm
Speed and mobility are yours with the quick draws and fast firing of the Strongarm blaster. Read more
$12.99 $9.44
Best Sellers
Sports & Outdoors : Golf Clubs


Netflix
www.netflix.com/WiHome
Reader
NETFLIX
Browse
Taste Profile
KiDS
DVDs
Titles, People, Genres
James
Recently Watched
Popular on Netflix
Littlest Pet Shop
THE NUT JOB
NETFLIX ORIGINAL
ORANGE is the new BLACK
NETFLIX ORIGINAL
the KILLING
NEW EPISODES
STAR WARS THE CLONE WARS
LEGO CHIMA
Breaking Bad
Top Picks for James
Disney MULAN
Disney TARZAN
JESSIE
LEGO NINJAGO Masters of Spinjitzu
Disney PIRATE FAIRY
HORSELAND
lab Rats

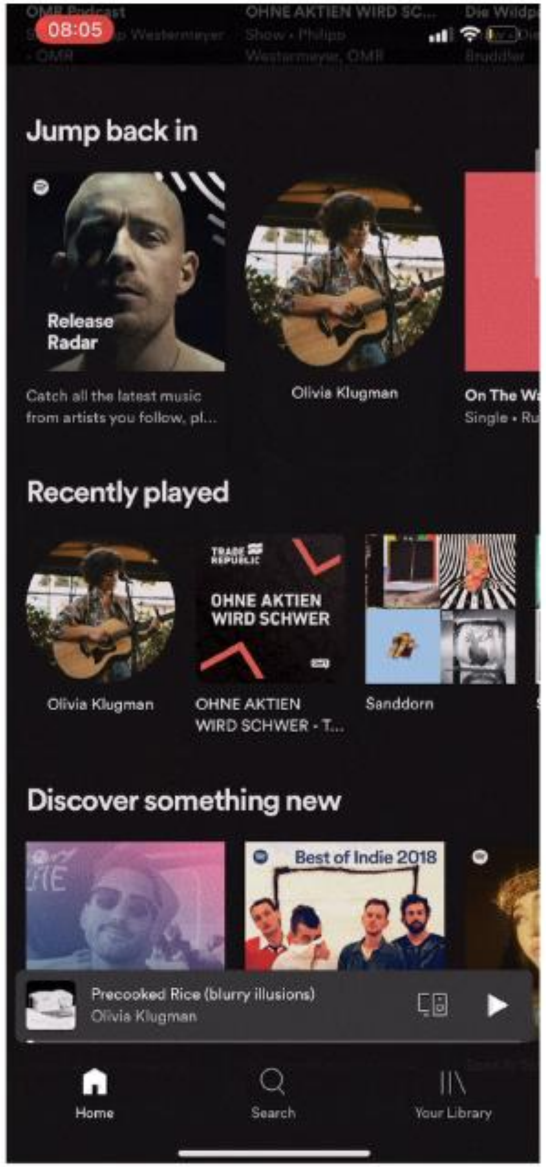

YouTube
Search
Home
Shorts
Subscriptions
Library
History
Your videos
Watch later
Liked videos
Subscriptions
AI Coffee Break ...
Alexander Amini
Browse channels
Explore
Trending
Shopping
Music
Movies & TV
All
Music
Boston Dynamics
Live
Gaming
Culinary arts
Tools
Background music
Game shows
Driving
Basketball
Algorithms
Classical Music
History
2:55
Do You Love Me?
Boston Dynamics
38M views • 2 years ago
Vox
???
10:43
Why we all need subtitles now
Vox
8.8M views • 1 month ago
INSIDER Business
BIG BUSINESS
11:15
How Singapore Airlines Makes 50,000 In-Flight Meals A Day | Big Business |...
Insider Business
1.5M views • 3 days ago
POLICE
23:14
Glitterbomb Trap Catches Phone Scammer (who gets arrested)
Mark Rober
70M views • 1 year ago
MICROSOFT-ACTIVISION DEAL
SMITH: "CALL OF DUTY" MORE AVAILABLE AFTER NINTENDO DEAL
LIVE
CNN
ND HUNDREDS LEFT HOMELESS AFTER HEAVY RAINS CAUSED LANDSLIDES
2:50
Smith: "Call of Duty" more available after Nintendo deal
Quest Means Business
2K views • 20 hours ago
11:16
The 4 Reasons Why You're Poor
Iman Gadzhi
838K views • 5 months ago
8:24
NBA bloopers but they keep getting more embarrassing
CoshReport
2.9M views • 10 months ago



Recommended Songs below



Home Screen with Shelves



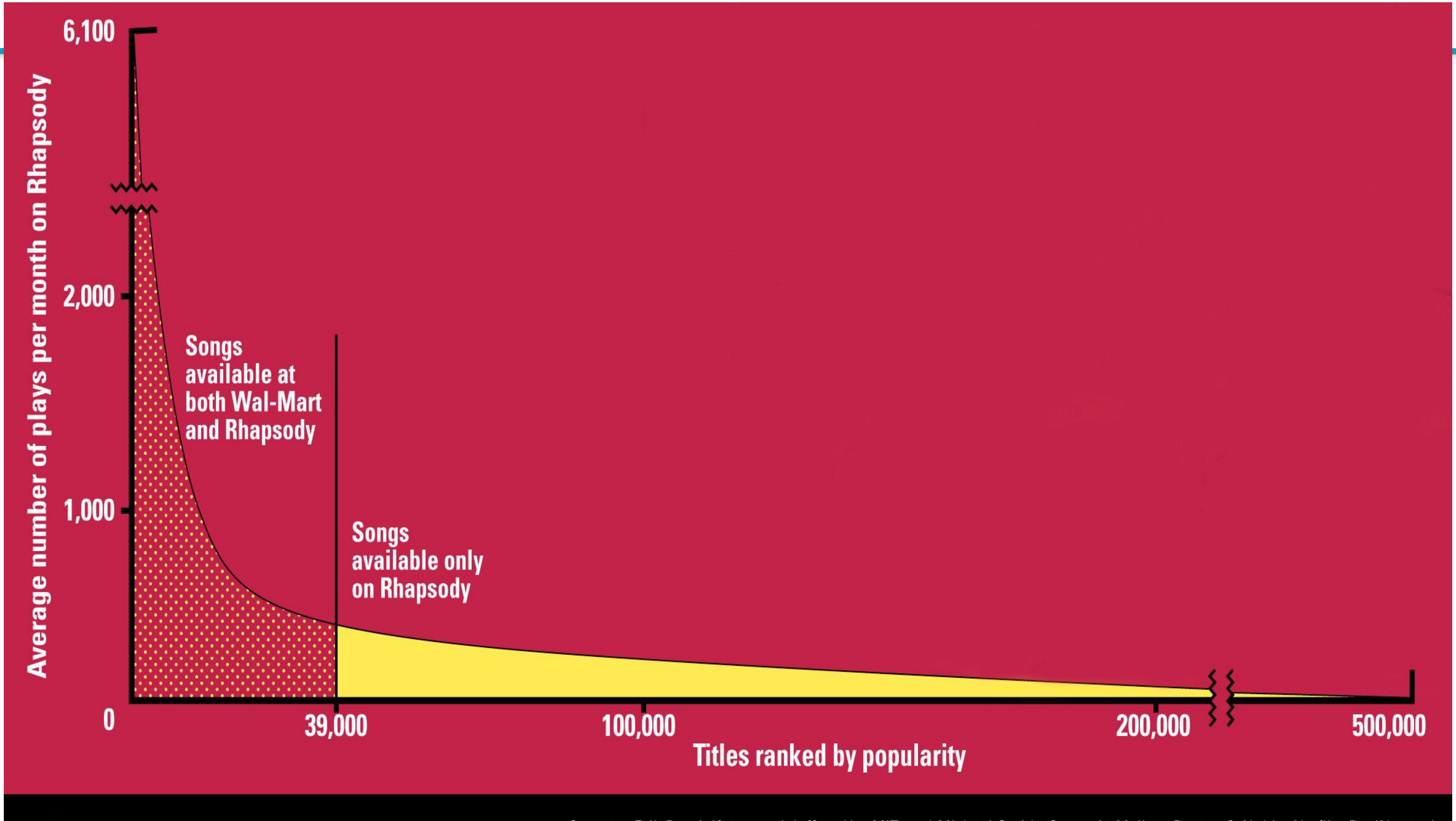
Jump back in, Recently played

Other examples of recommenders?

From Scarcity to Abundance

- Shelf space is a scarce commodity for traditional retailers
  - Also: TV networks, movie theaters,…
- Web enables near-zero-cost dissemination of information about products
  - From scarcity to abundance
- More choice necessitates better filters
  - Recommendation engines
  - How Into Thin Air made Touching the Void a bestseller
    - http://www.wired.com/wired/archive/12.10/tail.html

The Long Tail

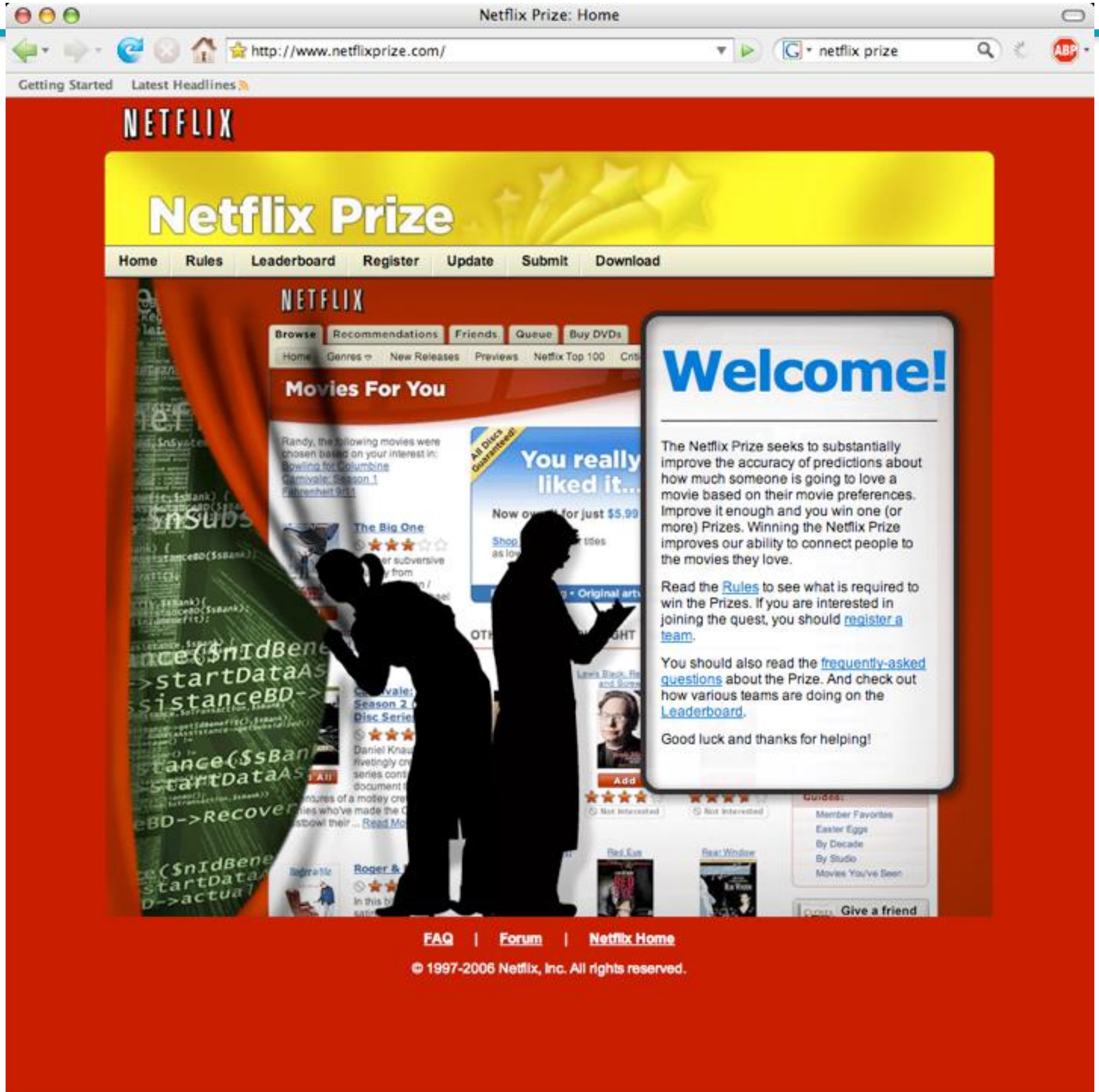


Source: Chris Anderson (2004)

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
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!
Movies For You
FAQ | Forum | Netflix Home
© 1997-2006 Netflix, Inc. All rights reserved.

Formal Model

- X = set of Customers (Users)
- 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

	The Lego Movie	The Fault in Our Stars	Guardians of the Galaxy	Star Wars
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 the known ones
  - Mainly interested in high unknown ratings
  - Don’t care about finding what you **don’t like**, but rather what you 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, clicks, listen to a song
  - What about low ratings?
    - Called negative sampling
      - Random sampling, popularity-based sampling, and so on.

(2) Extrapolating Utilities

- Key problem: Utility 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
  - Content-based
  - Collaborative
  - Latent factor based

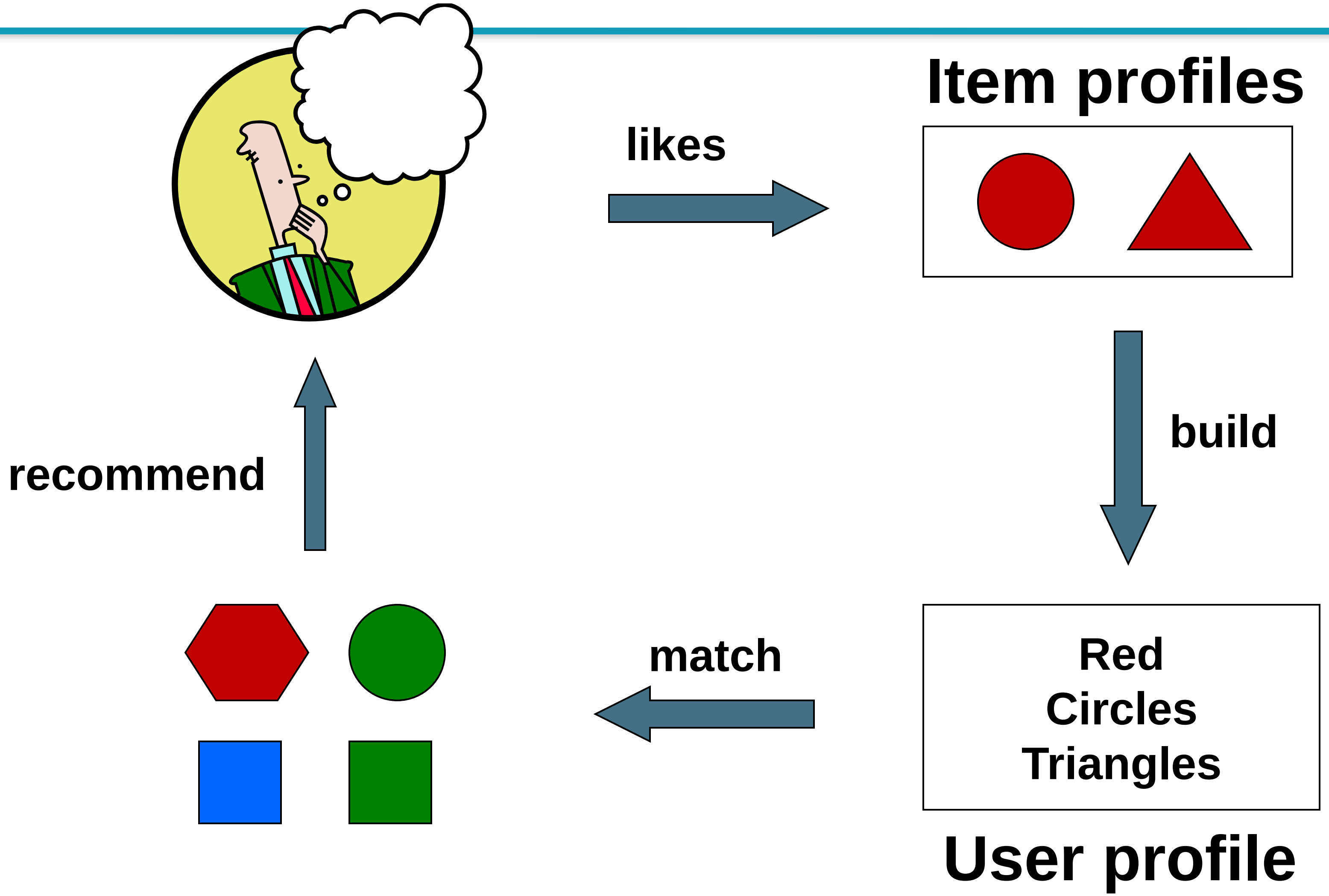
- These days, people focus on neural network/deep learning based approaches to capture more complex relation between users and items

Content-based Recommender Systems

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



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 features?
  - Usual heuristic is TF-IDF

Sidenote: TF-IDF

f_{ij} = frequency of term (feature) *i* in doc (item) *j*

Note: we normalize TF to discount for "longer" documents

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{ik}}$$

Whichever doc/item has the max # of frequency of term i

n_i = number of docs that mention term *i*

N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Doc/item profile = set of words with highest TF-IDF scores, together with their scores

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 Approach

- 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 & unpopular items
  - No first-rater problem
- Can provide explanations of recommended items by listing content-features that caused an item to be recommended

Limitations of content-based approach

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

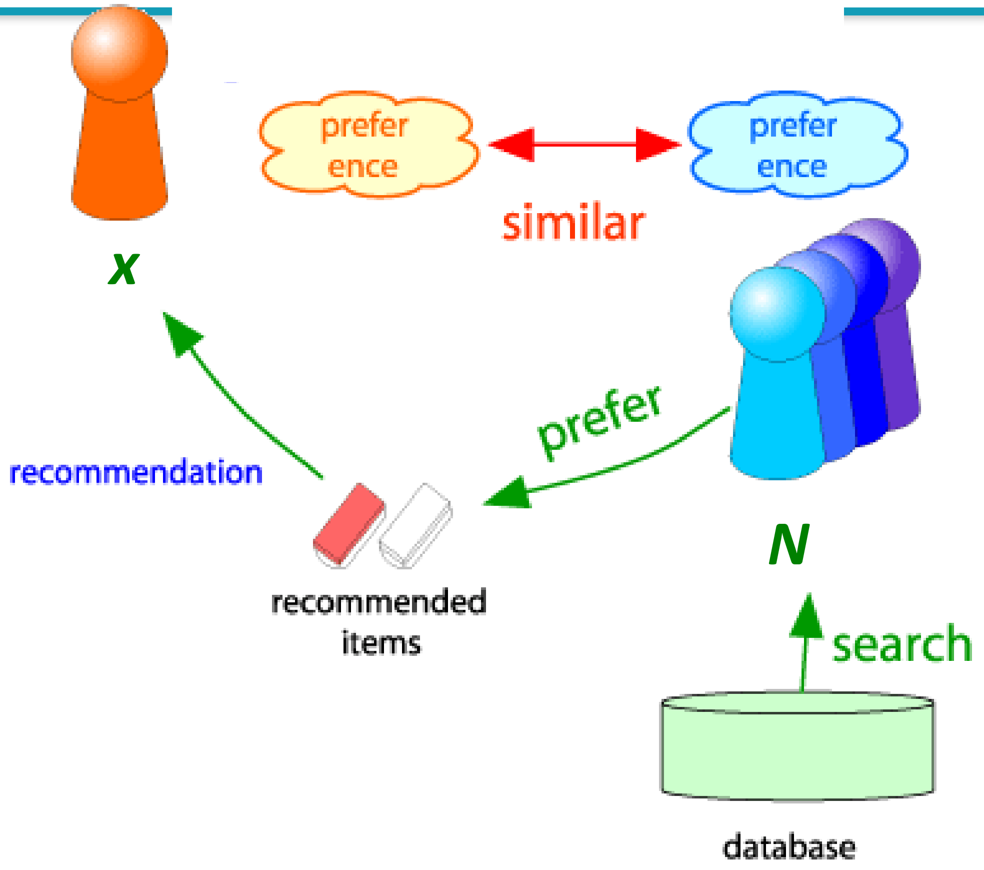
Collaborative Recommendations

Collaborative Recommendations

- User-based recommendation
- Item-based recommendation

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

r_x = [*, _, _, *, ***]
r_y = [*, _, **, **, _]

- Let r_x be the vector of user x's ratings
- **Jaccard similarity measure**
  - **Problem:** Ignores the value of the rating
- **Cosine similarity measure**
  - sim(x, y) = cos(r_x, r_y) = $\frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$
  - **Problem:** Treats missing ratings as "negative"
- **Pearson correlation coefficient**
  - S_{xy} = items rated by both users x and y

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

r_x, r_y as sets:
r_x = {1, 4, 5}
r_y = {1, 3, 4}

r_x, r_y as points:
r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

$\overline{r_x}$, $\overline{r_y}$ … avg. rating of **x**, **y**

Similarity Metric

	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: 0.380 > 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

Solution: subtract the (row) mean

	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

Notice cosine sim. is pearson correlation when data is centered at 0

Rating Predictions

From similarity metric to recommendations:

- Let $\boldsymbol{r_x}$ be the vector of user ***x***'s ratings
- Let ***N*** be the set of ***k*** users most similar to ***x*** who have rated item ***i***
- **Prediction for item i of *user x*:**
  - $r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$
  - $r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$

Shorthand:
$s_{xy} = sim(x, y)$

User-User CF (|N|=2)

users

movies	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

- unknown rating

- rating between 1 to 5

User-User CF (|N|=2)

users

movies

	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

Neighbor selection:
Identify users similar to user **5**, and rated item **1**

- estimate rating of movie **1** by user **5**

User-User CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	
Similarity:	**-0.45**		**0.21**		**1.0**	**-0.15**			**0.47**		**-0.71**	

Neighbor selection:
Identify users similar to user **5**, and rated item **1**

User-based CF (|K|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	
Similarity:	-0.45		<u>0.21</u>		1.0	-0.15			<u>0.47</u>		-0.71	

Compute similarity weights:

$s_{5,3}=0.21$, $s_{5,9}=0.47$

Predict by taking weighted average:

$r_{1,5} = (0.21*3 + 0.47*5) / (0.21+0.47) = 4.4$

$$r_{xi} = \frac{\sum_{j\in K(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

User-based CF (|K|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		4.4	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	
Similarity:	-0.45		0.21		1.0	-0.15			0.47		-0.71	

Compute similarity weights:

$s_{5,3}=0.21$, $s_{5,9}=0.47$

Predict by taking weighted average:

$r_{1,5} = (0.21*3 + 0.47*5) / (0.21+0.47) = 4.4$

$$r_{xi} = \frac{\sum_{j \in K(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

Issue with the user-based CF

- So far: user-based collaborative filtering

- Another view is **item-based CF**.

Item-based CF

- The item-based approach works by comparing items based on their pattern of ratings across users. The similarity of items *i* and *j* is computed as follows:

$$sim(i,j) = \frac{\sum_{\mathbf{u} \in U} (r_{\mathbf{u},i} - \bar{r}_{\mathbf{u}})(r_{\mathbf{u},j} - \bar{r}_{\mathbf{u}})}{\sqrt{\sum_{\mathbf{u} \in U} (r_{\mathbf{u},i} - \bar{r}_{\mathbf{u}})^2} \sqrt{\sum_{\mathbf{u} \in U} (r_{\mathbf{u},j} - \bar{r}_{\mathbf{u}})^2}}$$

Recommendation phase

- After computing the similarity between items we select a set of *k* most similar items to the target item and generate a predicted value of user x's rating

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

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

Item-Item CF (|N|=2)

users

movies	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

- unknown rating

- rating between 1 to 5

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

- estimate rating of movie **1** by user **5**

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	2	4		1	2		3		4	3	5		**0.41**
4		2	4		5			4			2		**-0.10**
5			4	3	4	2					2	5	**-0.31**
6	1		3		3			2			4		**0.59**

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use Pearson correlation as similarity:

1) Subtract mean rating m_i from each movie i

$m_1 = (1+3+5+5+4)/5 = \mathbf{3.6}$

row 1: *[-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]*

2) Compute cosine similarities between rows

Item-Item CF (|N|=2)

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		1.00
2			5	4			4			2	1	3	-0.18
3	2	4		1	2		3		4	3	5		**0.41**
4		2	4		5			4			2		-0.10
5			4	3	4	2					2	5	-0.31
6	1		3		3			2			4		**0.59**

Compute similarity weights:

$s_{1,3}=0.41$, $s_{1,6}=0.59$

Item-Item CF (|N|=2)

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		**2.6**	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	2	4		1	2		3		4	3	5		**0.41**
4		2	4		5			4			2		**-0.10**
5			4	3	4	2					2	5	**-0.31**
6	1		3		3			2			4		**0.59**

Predict by taking weighted average:

$r_{1.5}$ = **(0.41*2 + 0.59*3) / (0.41+0.59) = 2.6**

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

Item-Item vs. User-User

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.8	
Bob		0.5		0.3
Carol	0.9		1	0.8
David			1	0.4

- **In practice, it has been observed that item-item often works better than user-user**
- **Why?** Items are simpler, users have multiple tastes

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

Hybrid Methods

Hybrid Methods

- Add content-based methods to collaborative filtering
  - item profiles for new item problem
  - demographics to deal with new user problem
- Implement two separate recommenders and combine predictions
  - Perhaps using a linear model
  - E.g., global baseline + collaborative filtering

Global baseline estimate

- **Estimate Joe's rating for the movie The Sixth Sense**
  - No feature selection needed
  - Problem: Joe has not rated any movie similar to The Sixth Sense

- **Global baseline estimate**
  - Mean movie rating: **3.7 stars**
  - The Six Sense is **0.5 stars** above avg
  - Joe rates **0.2 stars** below avg
  - Baseline estimate: **3.7 + 0.5 – 0.2 = 4 stars**

Combining Global Baseline with CF

- **Global Baseline estimate:**
  - *Joe* will give *The Sixth Sense* 4 stars
- **Local neighborhood (CF/NN):**
  - *Joe* didn't like related movie *Signs*
  - Rated it 1 star below his average rating
- **Final estimate**
  - *Joe* will rate *The Sixth Sense* 4 – 1 = 3 stars

CF: Common practice

Before:

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

- Define **similarity** s_{ij} of items i and j
- Select k nearest neighbors $N(i; x)$
  - Items most similar to i, that were rated by x
- Estimate rating r_{xi} as the weighted average:

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

baseline estimate for r_{xi}

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

- μ = overall mean movie rating
- b_x = rating deviation of user x
  = *(avg. rating of user* x*)* $- \mu$
- b_i = rating deviation of movie i

Evaluation

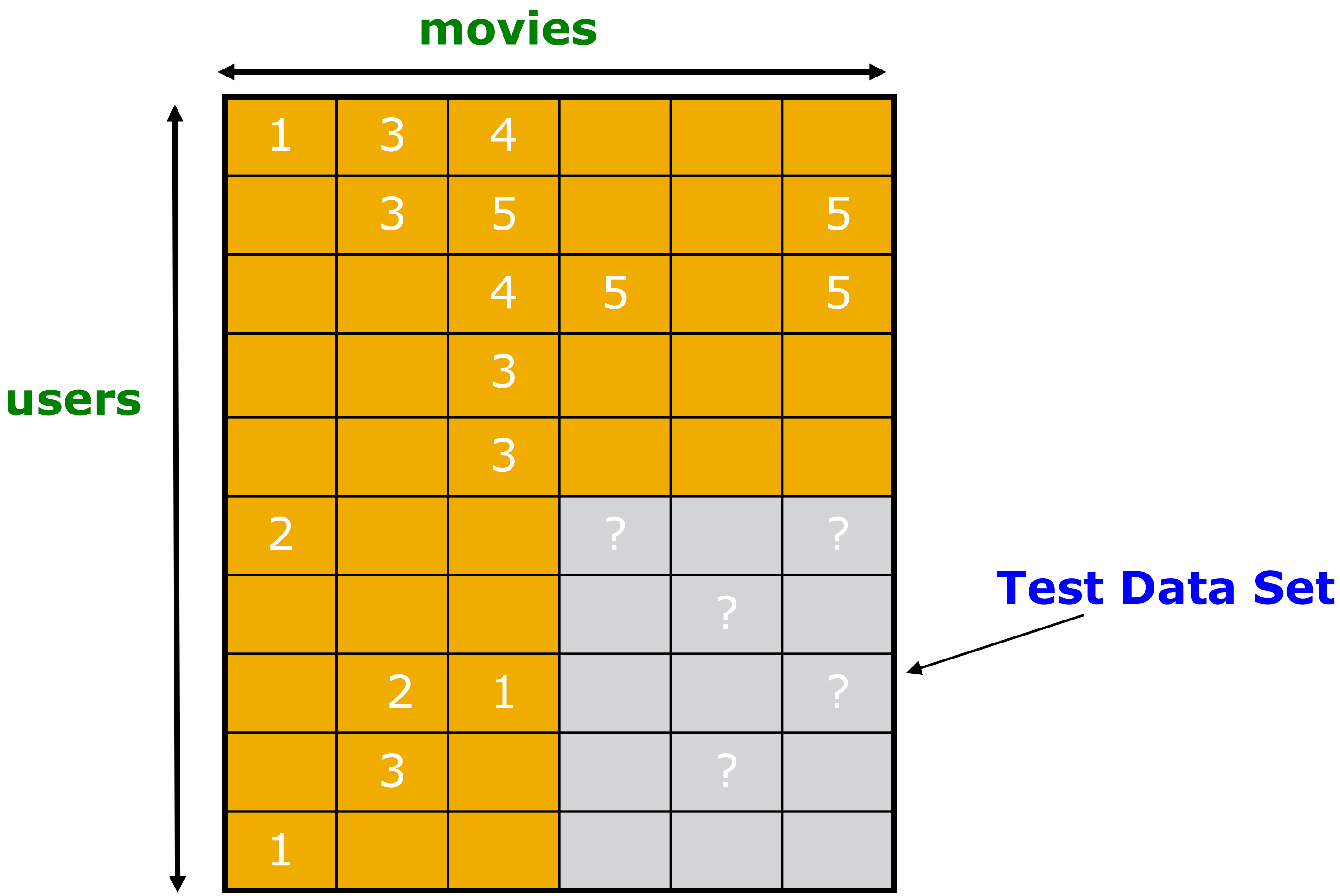
Evaluation

movies

users

1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			2		2
				5	
	2	1			1
	3			3	
1					

Evaluation



Evaluation Predictions

- **Root-mean-square error** (RMSE)
  - $\sqrt{\frac{1}{|R|}\sum_{(i,x)\in R}(\hat{r}_{xi} - r_{xi})^2}$ where $\hat{r}_{xi}$ is predicted, r_{xi} is the true rating of ***x*** on ***i***
- **Problems with Error Measures**
  - **Narrowly focus on accuracy sometimes misses the point**
    - Order of predictions
- **In practice, we care only to predict high ratings:**
  - RMSE might penalize a method that does well for high ratings and badly for others
- **Alternative: Precision@*k*, Hit Rate@*k* *(i.e., recall of positive interactions)*, and NDCG@*k***
  - ***E.g., k=10***

Three ways running Jupyter Notebook

- 1. Install Jupyter Notebook
  - https://jupyter.org/install

- 2. install Anaconda (if you installed it before, Jupyter Notebook was already installed together)
  - https://www.anaconda.com/products/individual
  - Beyond all of the normal (non-data centric) packages that Python comes with, Anacoda comes with even more!

- 3. Use a cloud computing (e.g., Google Colab)
  - https://colab.research.google.com/notebooks/intro.ipynb#recent=true

Google Colab

- **Google** Colaboratory is a free online cloud-based **Jupyter notebook** environment that allows us to train our machine learning and deep learning models on CPUs, GPUs, and TPUs.

https://www.youtube.com/watch?v=inN8seMm7UI

Recommendation Demo

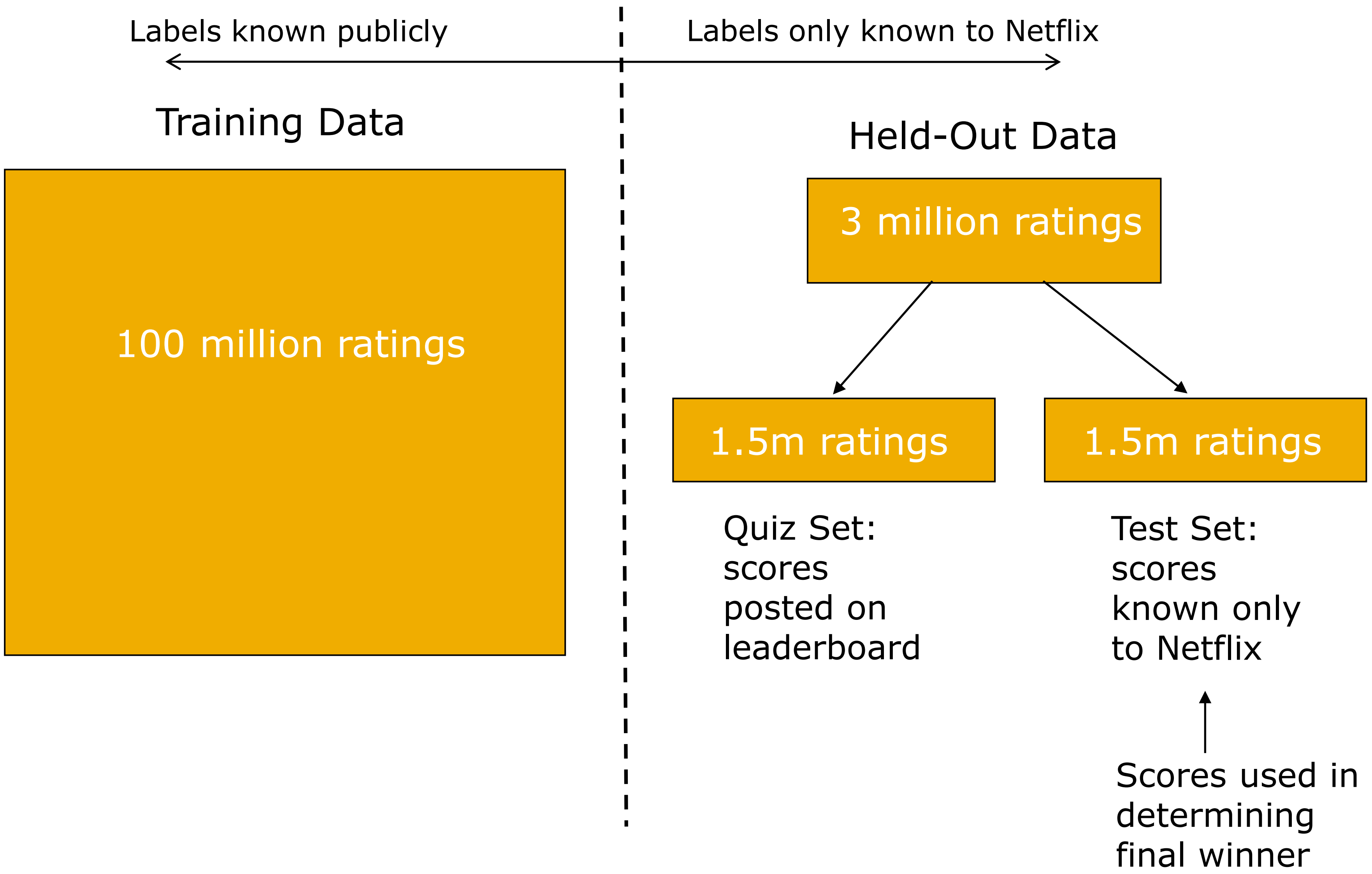
- recommenderDemo.ipynb

Latent Factor Models

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 Square Error (RMSE) $= \sqrt{\frac{1}{|R|} \sum_{(i,x)\in R} (\hat{r}_{xi} - r_{xi})^2}$
  - **Netflix's system RMSE: 0.9514**
- **Competition**
  - 2,700+ teams
  - **$1 million** prize for 10% improvement on Netflix

Competition Structure



The Netflix Utility Matrix *R*

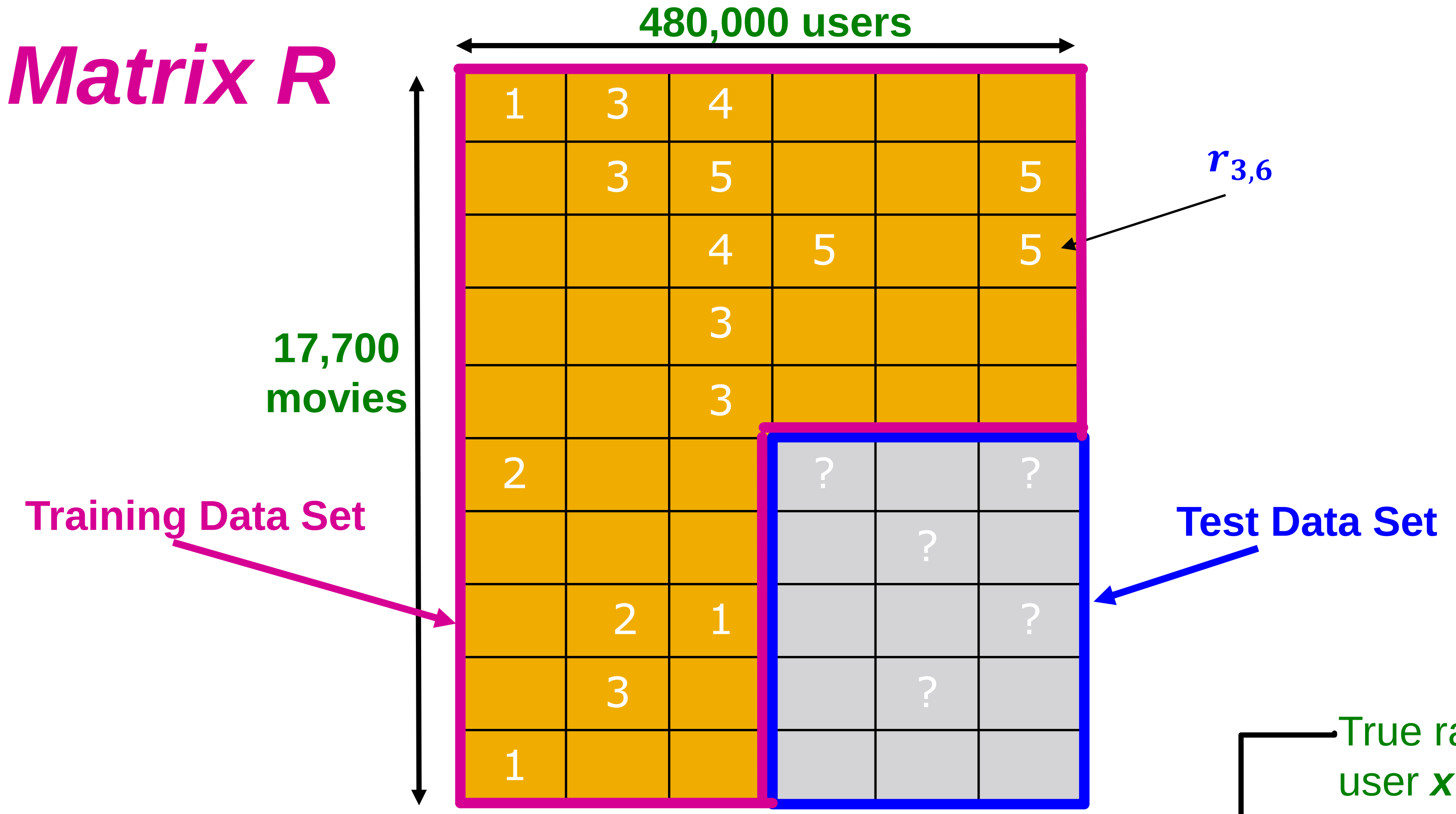
Matrix R

480,000 users

17,700 movies

1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			2		2
				5	
	2	1			1
	3			3	
1					

Utility Matrix *R*: Evaluation

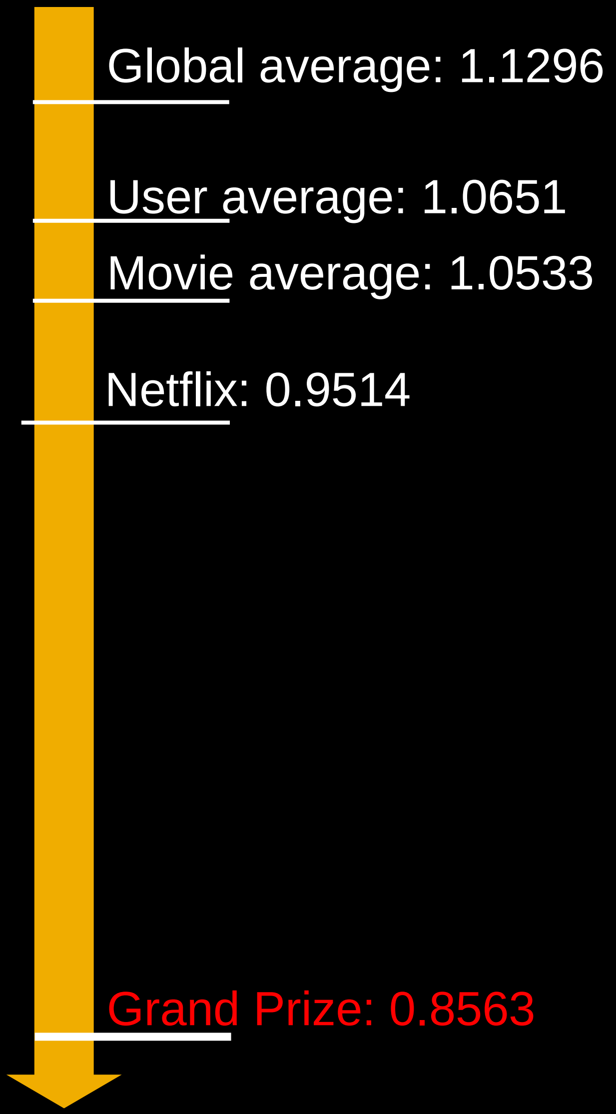


True rating of user ***x*** on item ***i***

$$\textbf{RMSE} = \sqrt{\frac{1}{|R|}\sum_{(i,x)\in R}\left(\hat{r}_{xi} - r_{xi}\right)^2}$$

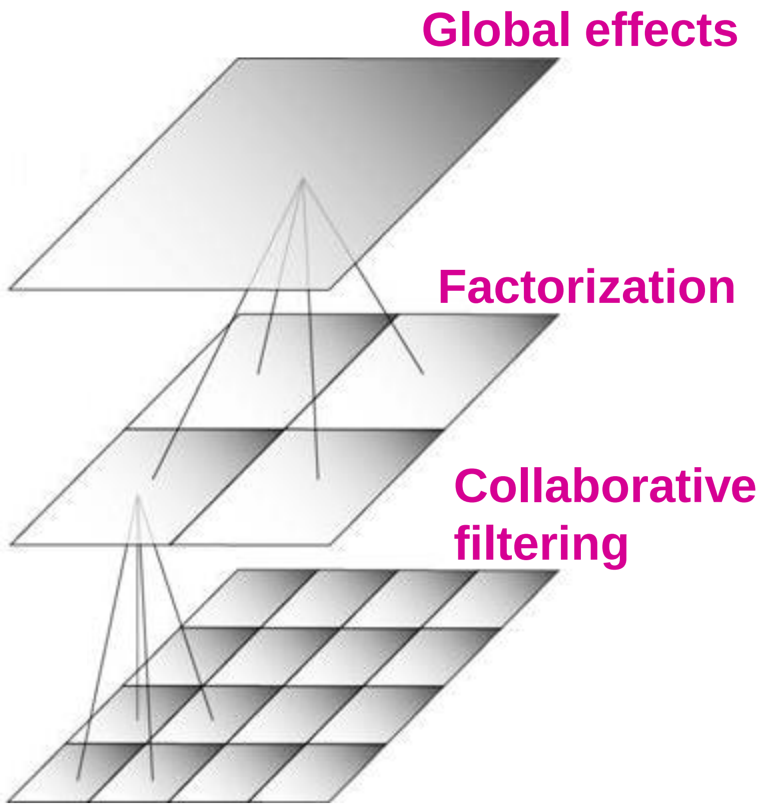
Predicted rating

Performance of Various Methods



BellKor Recommender System

- **The winner of the Netflix Challenge**
- **Multi-scale modeling of the data:** Combine top level, “regional” modeling of the data, with a refined, local view:
  - **Global:**
    - Overall deviations of users/movies
  - **Factorization:**
    - Addressing “regional” effects
  - **Collaborative filtering:**
    - Extract local patterns



Modeling Local & Global Effects

- **Global:**
  - Mean movie rating: **3.7 stars**
  - *The Sixth Sense* is **0.5** stars above avg.
  - Joe rates **0.2** stars below avg.
    ⇒ Baseline estimation:
    ***Joe* will rate *The Sixth Sense* 4 stars**

- **Local neighborhood (CF/NN):**
  - *Joe* didn't like related movie *Signs*
  - Rated it 1 star below his average rating

- **Final estimate**
  - *Joe* will rate *The Sixth Sense* 4 – 1 = 3 stars

Modeling Local & 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}

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

$\boldsymbol{\mu}$ = overall mean rating
$\boldsymbol{b_x}$ = rating deviation of user $\boldsymbol{x}$
= (*avg. rating of* user $\boldsymbol{x}$) – $\boldsymbol{\mu}$
$\boldsymbol{b_i}$ = (*avg. rating of* movie $\boldsymbol{i}$) – $\boldsymbol{\mu}$

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**

Idea: Interpolation Weights w_{ij}

- Use a **weighted sum** rather than **weighted avg.**:

$$\widehat{r_{xi}} = b_{xi} + \sum_{j \in N(i;x)} w_{ij}(r_{xj} - b_{xj})$$

- **A few notes:**
  - $\boldsymbol{N(i;x)}$ … set of movies rated by user ***x*** that are similar to movie ***i***
  - $\boldsymbol{w_{ij}}$ is the **interpolation weight** (some real number)
    - Note, we allow: $\sum_{j \in N(i;x)} \boldsymbol{w_{ij}} \neq \mathbf{1}$
  - $\boldsymbol{w_{ij}}$ models interaction between pairs of movies (it does not depend on user ***x***)

Idea: Interpolation Weights w_{ij}

- $\widehat{r_{xi}} = b_{xi} + \sum_{j \in N(i,x)} w_{ij}(r_{xj} - b_{xj})$
- **How to set w_{ij}?**
  - Remember, error metric is: $\sqrt{\frac{1}{|R|}\sum_{(i,x)\in R}(\hat{r}_{xi} - r_{xi})^2}$ or equivalently Sum of Squared Error (**SSE**): $\boldsymbol{\sum_{(i,x)\in R}(\hat{r}_{xi} - r_{xi})^2}$
  - Find ***w_{ij}*** that minimize **SSE** on **training data!**
    - Models relationships between item ***i*** and its neighbors ***j***
  - ***w_{ij}*** can be **learned/estimated** based on ***x*** and all other users that rated ***i***

Recommendations via Optimization

1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			2		2
				5	
	2	1			1
	3			3	
1					

- **Goal:** Make good recommendations
  - Quantify goodness using **RMSE:**
    Lower RMSE $\Rightarrow$ better recommendations
  - Want to make good recommendations on items that user has not yet seen. Can't really do this!
  - **Let's build a system such that it works well on known (user, item) ratings**
    And **hope** the system will also predict well the **unknown ratings**

Recommendations via Optimization

- **Idea:** **Let's set values *w* such that they work well on known (user, item) ratings**
- **How to find such values *w*?**
- **Idea:** Define an objective function and solve the optimization problem

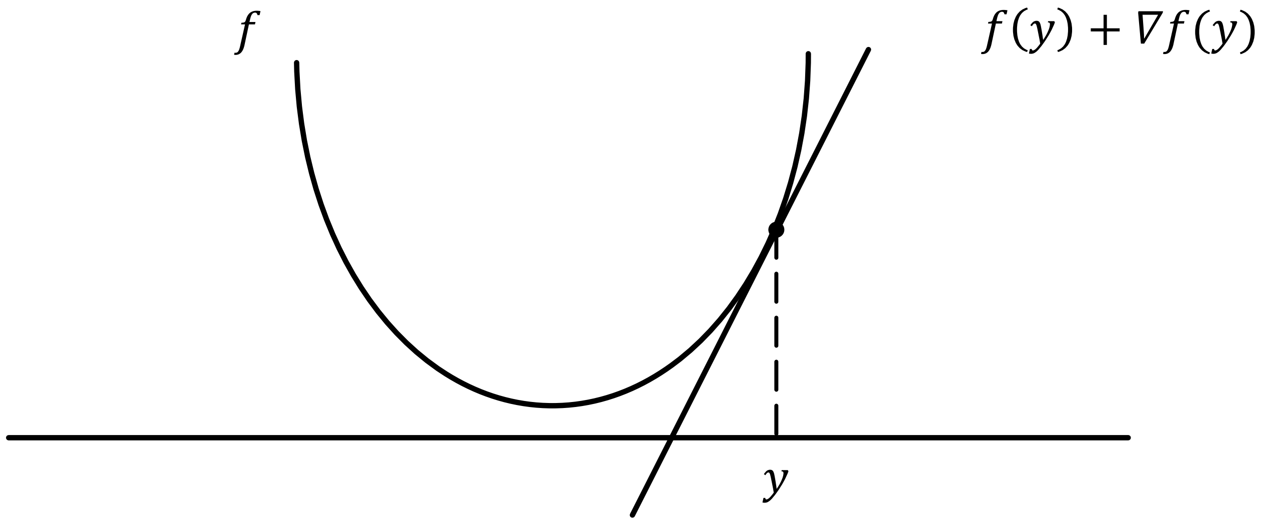
- Find $\boldsymbol{w_{ij}}$ that minimize **SSE** on **training data**!

$$J(w) = \sum_{x,i \in R} \left(\underbrace{\left[b_{xi} + \sum_{j \in N(i;x)} w_{ij}(r_{xj} - b_{xj}) \right]}_{\text{Predicted rating}} - \underbrace{r_{xi}}_{\text{True rating}} \right)^2$$

- Think of ***w*** as a vector of numbers

Detour: Minimizing a function

- **A simple way to minimize a function $f(x)$:**
  - Compute the derivative $\nabla f(x)$
  - **Start at some point y and evaluate $\nabla f(y)$**
  - **Make a step in the reverse direction of the gradient: $y = y - \nabla f(y)$**
  - **Repeat until converged**



Interpolation Weights

- **So far:** $\widehat{r_{xi}} = b_{xi} + \sum_{j \in N(i;x)} w_{ij}\left(r_{xj} - b_{xj}\right)$
  - Weights $\boldsymbol{w_{ij}}$ derived based on their role; **no use of an arbitrary similarity measure** ($\boldsymbol{w_{ij}} \neq \boldsymbol{s_{ij}}$)
  - Explicitly account for interrelationships among the neighboring movies
- **Next: Latent factor model**
  - Extract "regional" correlations