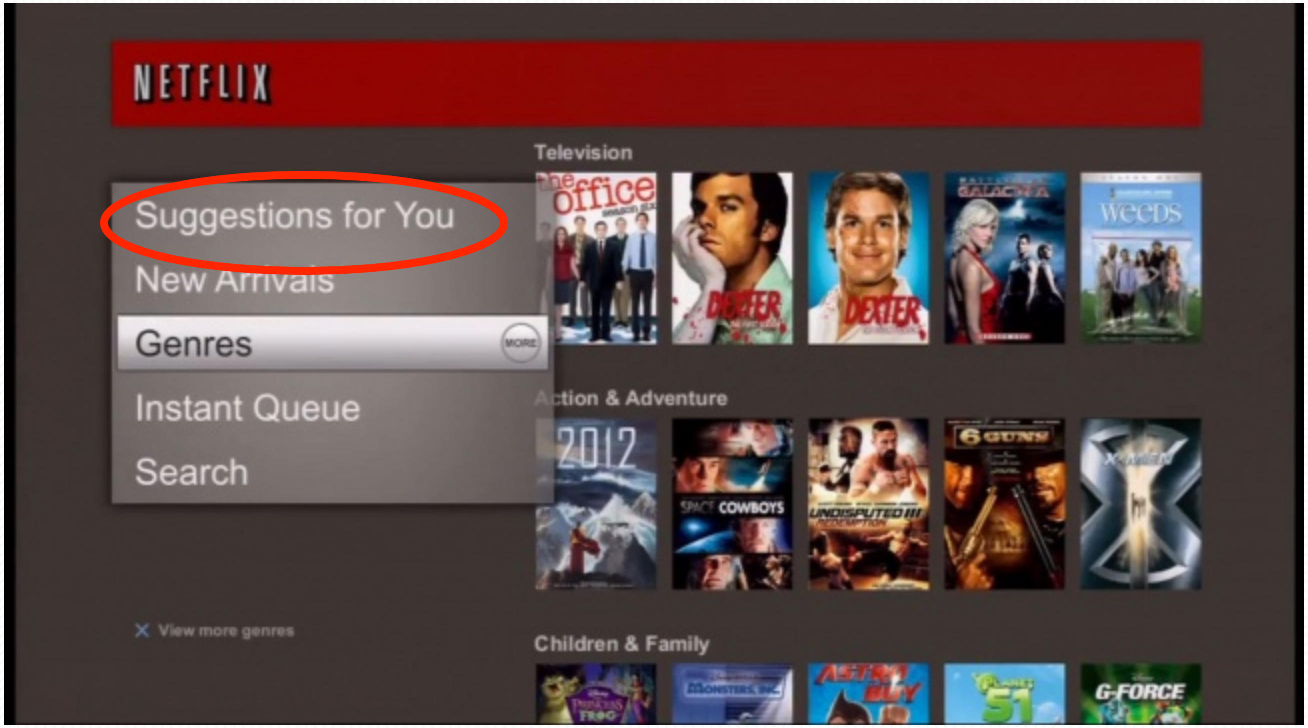
Recommendation System

Netflix Viewing Recommendations



Recommender Systems

DOMAIN: some field of activity where *users* buy, view, consume, or otherwise experience *items*

PROCESS:

1. *users* provide *ratings* on *items* they have experienced

   (Rating may be implicit: view+buy->good, view+no buy ->not good)

2. Take all < *user*, *item*, *rating* > data and build a predictive model
3. For a *user* who hasn't experienced a particular *item*, use model to *predict* how well they will like it (i.e. *predict rating*)

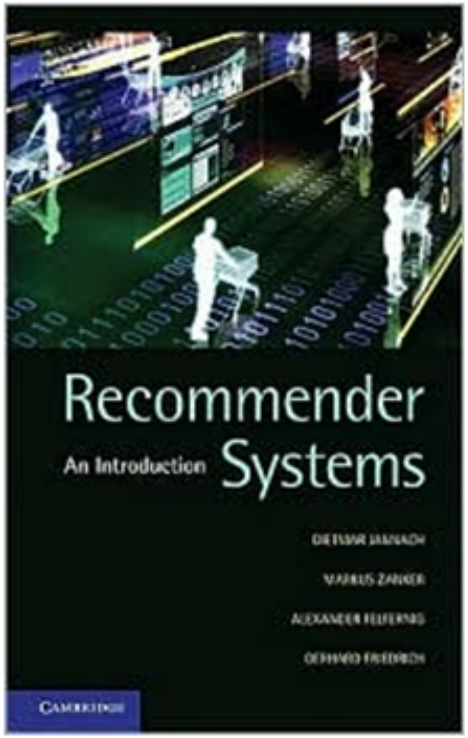
Roles of Recommender Systems

- Help users deal with *paradox of choice*

- Allow online sites to:
  - Increase likelihood of sales
  - Retain customers by providing positive search experience

- Considered essential in operation of:
  - Online retailing, e.g. Amazon, Netflix, etc.
  - Social networking sites

Amazon.com Product Recommendations



Recommender Systems: An Introduction

by Dietmar Jannach, Markus Zanker, Alexander Felfernig, Gerhard Friedrich

AVERAGE CUSTOMER RATING:

(Be the first to review)

Gefällt mir | Registrieren, um sehen zu können, was deinen Freunden gefällt.

FORMAT:
Hardcover

NOOKbook (eBook) - not available

Tell the publisher you want this in NOOKbook format

NEW FROM BN.COM

~~$65.00~~ List Price
$52.00 Online Price
(You Save 20%)

Add to Cart

NEW & USED FROM OUR

New starting at **$56.46**(You S
Used starting at **$51.98**(You S

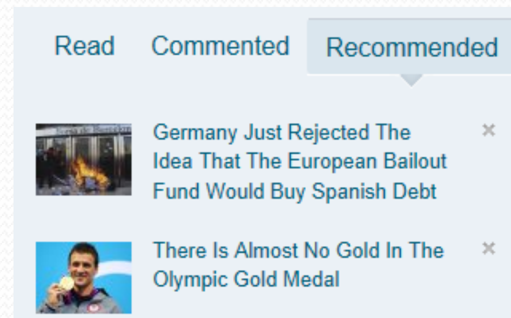
See All Prices

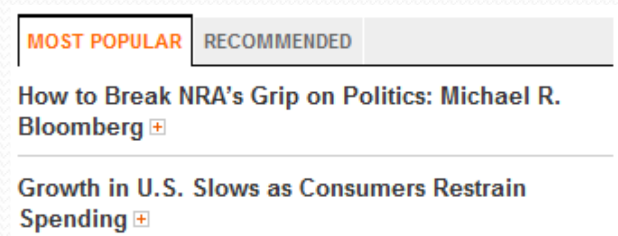
Table of Contents

Customers who bought this also bought

Recommender Systems

- Application areas

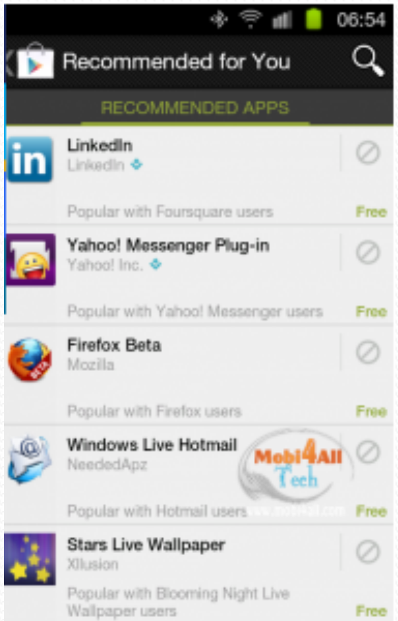


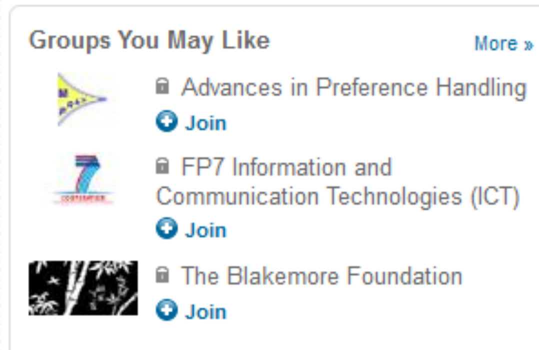


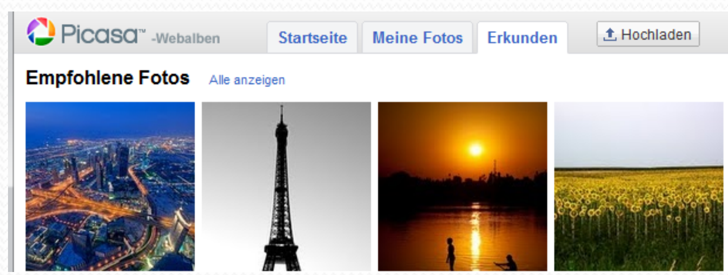
Social Network Recommendations

- Recommendations on essentially every category of interest known to mankind
  - Friends
  - Groups
  - Activities
  - Media (TV shows, movies, music, books)
  - News stories
  - Ad placements
- All based on connections in underlying social network graph and your expressed ‘likes’ and ‘dislikes’

Social Network Recommendation







The Netflix Prize Contest

- *GOAL*: use *training data* to build a recommender system, which, when applied to *qualifying data*, improves error rate by 10% relative to Netflix's existing system

- *PRIZE*: first team to 10% wins $1,000,000
  - Annual Progress Prizes of $50,000 also possible

The Netflix Prize Contest

- *CONDITIONS*:
  - Open to public
  - Compete as individual or group
  - Submit predictions no more than once a day
  - Prize winners must publish results and license code to Netflix (non-exclusive)

- *SCHEDULE*:
  - Started Oct. 2, 2006
  - To end after 5 years

The Netflix Prize Contest

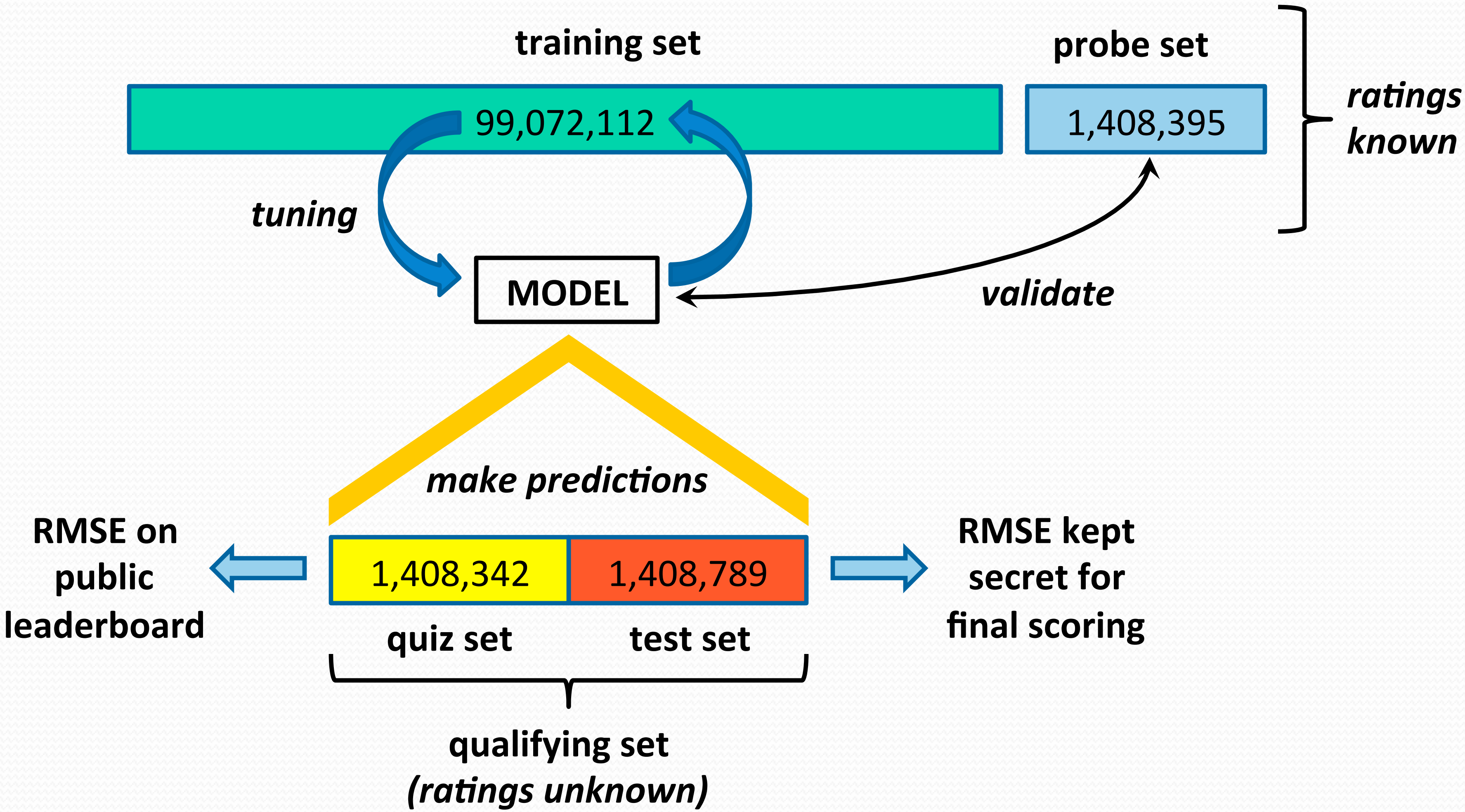
- *PARTICIPATION*:
  - 51051 contestants on 41305 teams from 186 different countries
  - 44014 valid submissions from 5169 different teams

The Netflix Prize Data

- Netflix released three datasets
  - 480,189 *users* (anonymous)
  - 17,770 *movies*
  - *ratings* on integer scale 1 to 5

- <u>*Training set*</u>: 99,072,112 < *user, movie* > pairs with *ratings*
- <u>*Probe set*</u>: 1,408,395 < *user, movie* > pairs with *ratings*
- <u>*Qualifying set*</u> of 2,817,131 < *user, movie* > pairs with <u>*no*</u> *ratings*

Model Building and Submission Process

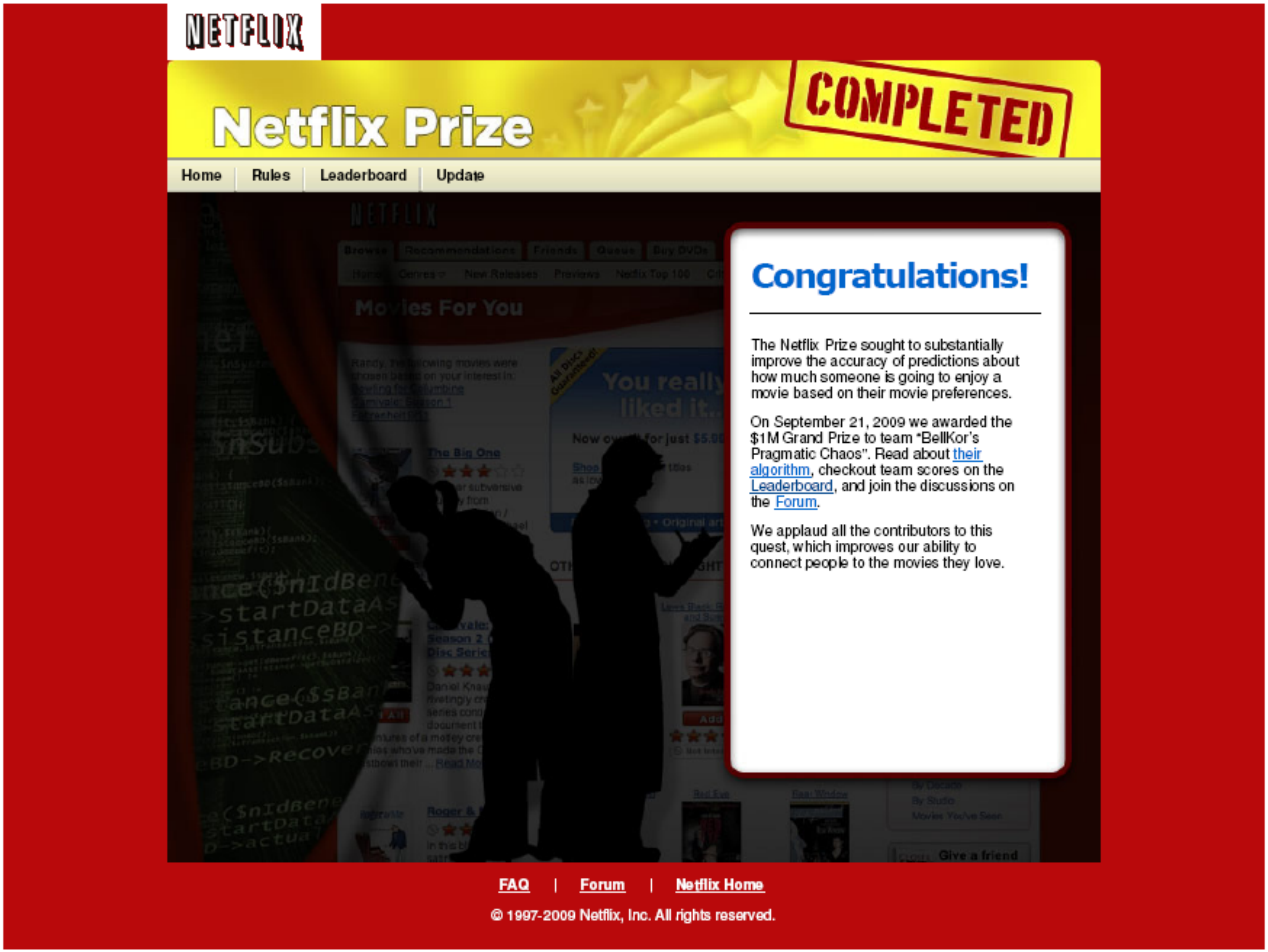


Netflix Prize

Leaderboard

Display top 100 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
--	**No Grand Prize candidates yet**	–	--	--
	Grand Prize - RMSE <= 0.8563			
1	BellKor in BigChaos	0.8604	9.56	2008-12-03 16:46:15
	Progress Prize - RMSE <= 0.8625			
2	BigChaos	0.8626	9.33	2008-12-04 19:18:27
3	BellKor	0.8630	9.29	2008-12-04 19:25:59
4	PragmaticTheory	0.8638	9.21	2008-11-28 11:46:23
5	Gravity	0.8654	9.04	2008-11-27 21:18:37
6	My Brain and His Chain	0.8668	8.89	2008-09-30 02:19:47
7	Just a guy in a garage	0.8672	8.85	2008-12-07 06:51:12
8	When Gravity and Dinosaurs Unite	0.8675	8.82	2008-10-05 14:16:53
9	Opera Solutions	0.8676	8.81	2008-12-02 22:08:45
10	acmehill	0.8677	8.80	2008-12-05 08:01:00
11	scientist	0.8677	8.80	2008-12-02 01:10:13
12	Ces	0.8711	8.44	2008-08-25 05:00:23
13	Dace	0.8711	8.44	2008-12-07 03:46:04
	Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell			
14	KorBell	0.8712	8.43	2007-10-01 23:25:23
15	basho	0.8714	8.41	2008-05-21 22:06:00
16	pengpengzhou	0.8714	8.41	2008-11-05 01:11:13
17	blednotik	0.8717	8.38	2008-11-26 00:12:12


NETFLIX
Netflix Prize
COMPLETED
Home
Rules
Leaderboard
Update
Congratulations!
The Netflix Prize sought to substantially improve the accuracy of predictions about how much someone is going to enjoy a movie based on their movie preferences.
On September 21, 2009 we awarded the $1M Grand Prize to team "BellKor's Pragmatic Chaos". Read about their algorithm, checkout team scores on the Leaderboard, and join the discussions on the Forum.
We applaud all the contributors to this quest, which improves our ability to connect people to the movies they love.
FAQ | Forum | Netflix Home
© 1997-2009 Netflix, Inc. All rights reserved.

One Algorithm from Winning Team

(time-dependent matrix factorization)

This leads to the prediction rule

$$\hat{r}_{ui} = \mu + b_i(t_{ui}) + b_u(t_{ui}) + |\mathrm{R}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{R}(u)} e^{-\beta_u \cdot |t_{ui} - t_{uj}|}((r_{uj} - b_{uj})w_{ij} + c_{ij}). \tag{15}$$

The involved parameters, $b_i(t_{ui}) = b_i + b_{i,\mathrm{Bin}(t_{ui})}$, $b_u(t_{ui}) = b_u + \alpha_u \cdot \mathrm{dev}_u(t_{ui}) + b_{u,t_{ui}}$, β_u, w_{ij} and c_{ij}, are learned by minimizing the associated regularized squared error

$$\sum_{(u,i) \in \mathcal{K}} \Big(r_{ui} - \mu - b_i - b_{i,\mathrm{Bin}(t_{ui})} - b_u - \alpha_u \mathrm{dev}_u(t_{ui}) - b_{u,t_{ui}} - |\mathrm{R}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{R}(u)} e^{-\beta_u \cdot |t_{ui} - t_{uj}|}((r_{uj} - b_{uj})w_{ij} + c_{ij})\Big)^2 + \lambda_{12}(b_i^2 + b_{i,\mathrm{Bin}(t_{ui})}^2 + b_u^2 + \alpha_u^2 + b_{ut}^2 + w_{ij}^2 + c_{ij}^2). \tag{16}$$

Minimization is performed by stochastic gradient descent.

Yehuda Koren, *Comm. ACM*, **53**, 89 (2010)

Collaborative Filtering Recommendation

Nearest Neighbors in Action

	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6	movie 7	movie 8	movie 9	movie 10	…	movie 17770
user 1			1		2							3
user 2		2		3	3			4		**?**		
user 3							5	3				
user 4	2				3			2				2
user 5		2		3		5		4		2		4
user 6			2									
user 7			2					4	2			
user 8	3	1			3	4		5		4		
user 9									3			
user 10			1		2							2
…												
user 480189		4			3			3				

Identical preferences – strong weight

Similar preferences – moderate weight

Item-based collaborative filtering recommendation

- Scalability issues arise with U2U if many more users than items
  ($m >> n$, $m = |users|$, $n = |items|$)
  - e.g. Amazon.com
  - Space complexity $O(m^2)$ when pre-computed
  - Time complexity for computing Pearson $O(m^2n)$

- High sparsity leads to few common ratings between two users

- Basic idea: "Item-based CF exploits relationships between items first, instead of relationships between users"

Item-based collaborative filtering

- Basic idea:
  - Use the rating on other items to make predictions
- Example:
  - Look for items that are similar to Item5

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

the rating for

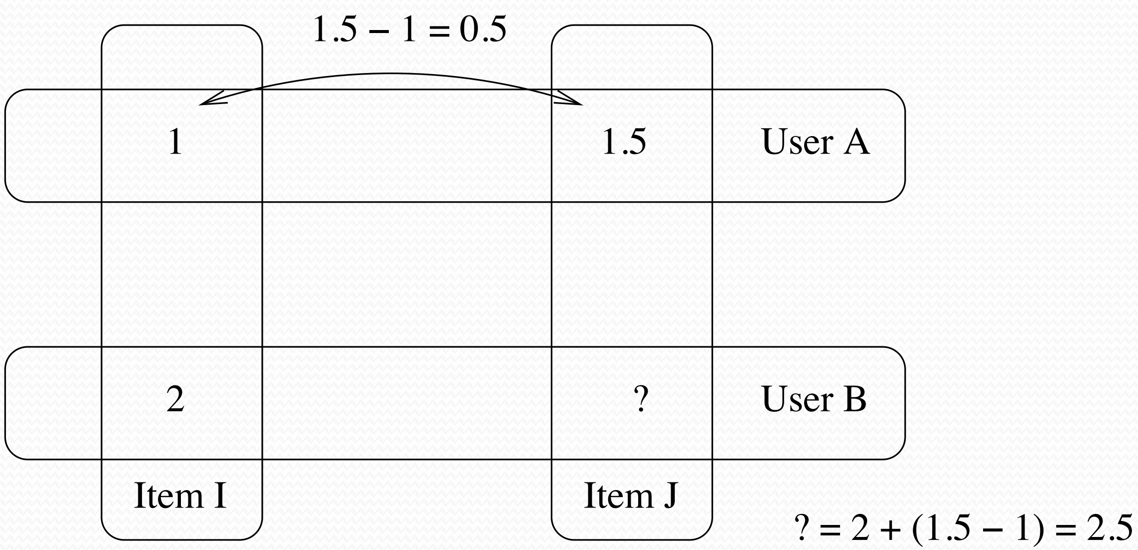
Slope One Recommender

- Suppose there are users $u_1, u_2, \ldots, u_k$ who have rated items A and B and the ratings are $A_1, A_2, \ldots, A_k$ and $B_1, B_2, \ldots, B_k$
- We want to find a function in the form of $f(x) = x + b$ to describe the relationship between the ratings for A and the ratings for B.
- ?? Solve the equations:
  - $B_1 = f(A_1) = A_1 + b$
  - $B_2 = f(A_2) = A_2 + b$
  - $B_3 = f(A_3) = A_3 + b$
  - …
  - $B_k = f(A_k) = A_k + b$

Slope One Recommender

- There may not be solution to b that agrees with all A_i and B_i
- Use regression to find the relationship, i.e., find the value of b:
  - that minimizes $$\sum_{i=1}^{k}(A_i + b - B_i)^2$$
  - This solves to: $$b = \frac{\sum_{i=1}^{k}(B_i - A_i)}{k}$$
- You can now make prediction of B_x if you know A_x.

Slope One Recommender



- If there are more items, we can average the predictions.

- Begin by computing the average preference value difference between all item pairs
- Items 102 and 101: $\frac{(3.5-5)+(5-2)+(3.5-4.5)}{3} = \frac{0.5}{3}$
- Items 103 and 101: $\frac{(4-2)+(1-4.5)}{2} = \frac{-1.5}{2}$
- Items 104 and 101: $\frac{(2-2)+(4-4.5)}{2} = \frac{-0.5}{2}$
- Items 103 and 102: $\frac{(4-5)+(1-3.5)}{2} = \frac{-3.5}{2}$
- And so on…

	101	102	103	104
User X	5	3.5		
User Y	2.0	5.0	4.0	2.0
User Z	4.5	3.5	1	4.0

- When done we have a table we can use to look up the average difference between any two items
- This completes the preprocessing step
- Empty cells contain inverses that are omitted here

Item	101	102	103	104
101	-			
102	0.17	-		
103	-0.75	-1.75	-	
104	-0.25	-1.25	0.5	-

- Let’s recommend an item for user X
- There are two potential candidates: item 103 and item 104
- We want to predict X’s preferences for both items and recommend the one user X would prefer
- We need to do this using all of X’s existing items: 101,102

	101	102	103	104
User X	5	3.5		
User Y	2.0	5.0	4.0	2.0
User Z	4.5	3.5	1	4.0

- Predict X's preference for item 103 using item 101
  - Look up the pre-computed average preference difference between items 103 and 101: -0.75
  - Use this to predict X's rating for item 103 based on item 101
    - X's rating for item 101: 5
    - X's predicted preference for item 103 using 101: -0.75 + 5 = 4.25

Item	101	102	103	104
101	-			
102	0.17	-		
103	-0.75	-1.75	-	
104	-0.25	-1.25	0.5	-

	101	102	103	104
User X	5	3.5		

- Predict X's preference for item 103 using item 102
  - Look up the pre-computed average preference difference between items 103 and 102: -1.75
  - Use this to predict X’s rating for item 103 based on item 102
    - X’s rating for item 102: 3.5
    - X’s predicted preference for item 103 using 102: -1.75 + 3.5 = 1.75

Item	101	102	103	104
101	-			
102	0.17	-		
103	-0.75	-1.75	-	
104	-0.25	-1.25	0.5	-

	101	102	103	104
User X	5	3.5		

- Predict X's preference for item 103
  - Preference for item 103 based on item 101: 4.25
  - Preference for item 103 based on item 102: 1.75
  - Averaging these predictions together gives us a final prediction of X’s preference value for item 103
  - $\frac{4.25+1.75}{2} = 3$

- Next predict X's preference for item 104 using 101
  - Look up the pre-computed average preference difference between items 104 and 101: -0.25
  - Use this to predict X’s rating for item 104 based on item 101
    - X’s rating for item 101: 5
    - X’s predicted preference for item 104: -0.25 + 5 = 4.75

Item	101	102	103	104
101	-			
102	0.17	-		
103	-0.75	-1.75	-	
104	-0.25	-1.25	0.5	-

	101	102	103	104
User X	5	3.5		

- Predict X's preference for item 104 using 102
  - Look up the pre-computed average preference difference between items 104 and 102: -1.25
  - Use this to predict X’s rating for item 104 based on item 102
    - X’s rating for item 102: 3.5
    - X’s predicted preference for item 104: -1.25 + 3.5 = 2.25

Item	101	102	103	104
101	-			
102	0.17	-		
103	-0.75	-1.75	-	
104	-0.25	-1.25	0.5	-

	101	102	103	104
User X	5	3.5		

- Predict X's preference for item 104
  - Preference for item 104 based on item 101: 4.75
  - Preference for item 104 based on item 102: 2.25
  - Averaging these predictions together gives us a final prediction of X’s preference value for item 104
  - $\frac{4.75+2.25}{2} = 3.5$
- Now make a recommendation based on the predicted values
  - Item 103: 3
  - Item 104: 3.5
  - Item 104 has the highest predicted preference value
  - Recommend item 104 to user X