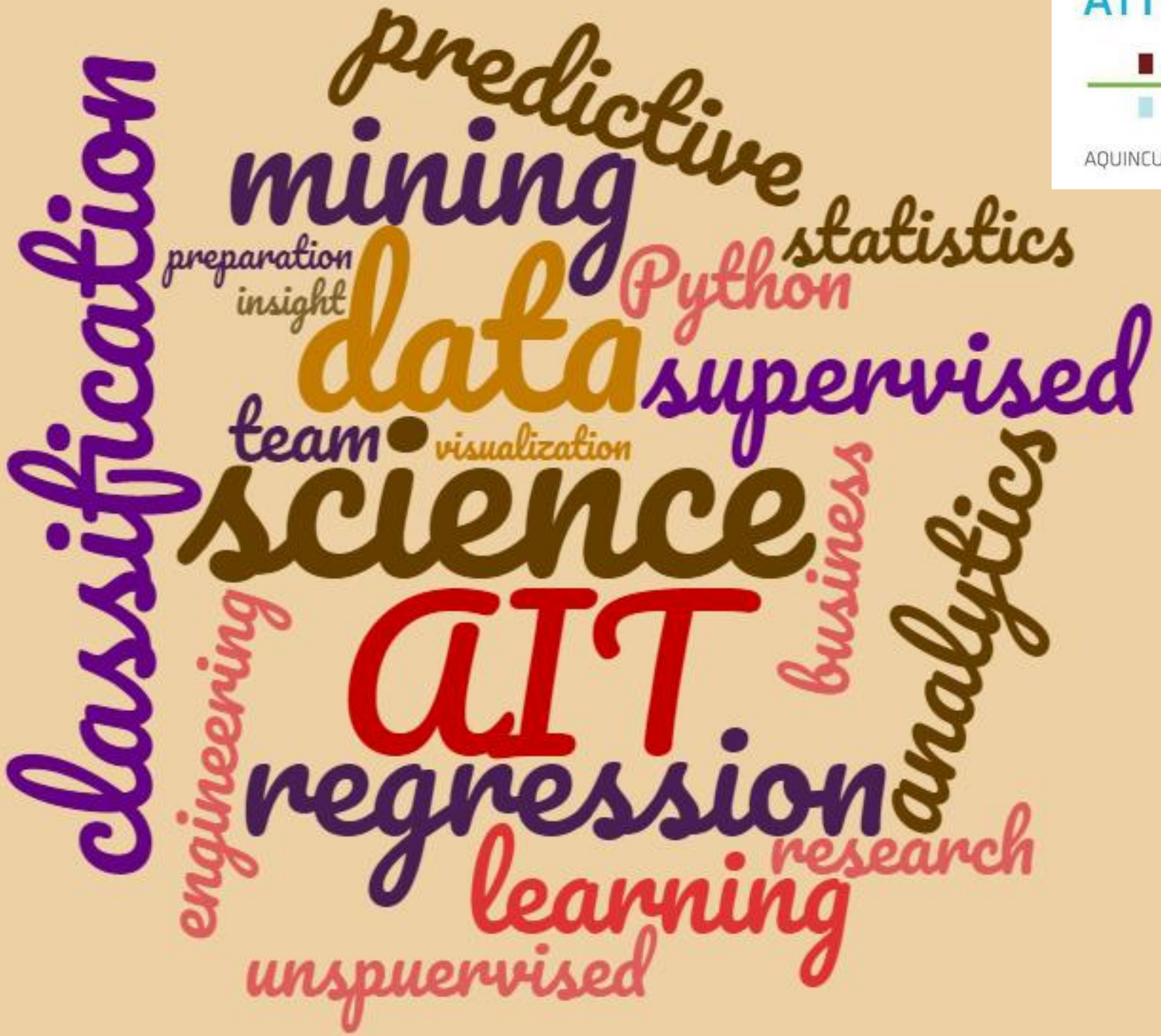
Data Science



Dr. Roland Molontay

May 9, 2023.
Recommendastion systems

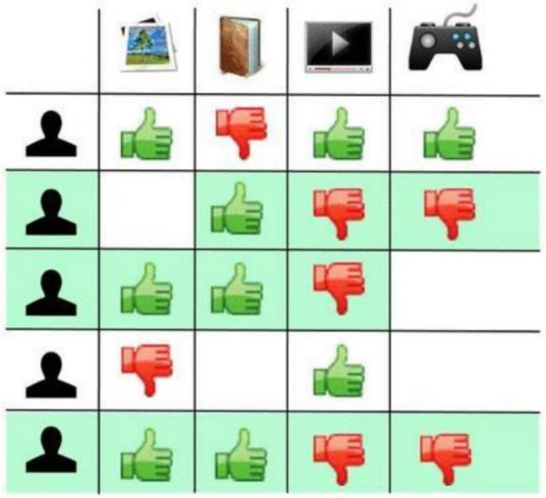
Schedule of the semester

	Monday midnight	*Tuesday class*	*Friday class*
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)		HW4 out	
W11 (04/24)			
W12 (05/01)	HW4 deadline		
W13 (05/08)	MILESTONE 2		
W14 (05/15)		FINAL	-~~PROJECT presentations~~
W15 (05/22)		PROJECT presentations	

Recommender systems everywhere

Recommender systems

- Users/customers and products/items are given
  - We know some features of the users and/or items
  - We know the ratings given by the users for some items
    - The rating can be given explicitly in the form of likes/dislikes, evaluation scores
    - Or implicitly by the fact of purchase or clicks
- Goal: to assist the user by recommending useful/interesting items and assist the business to realize higher profit



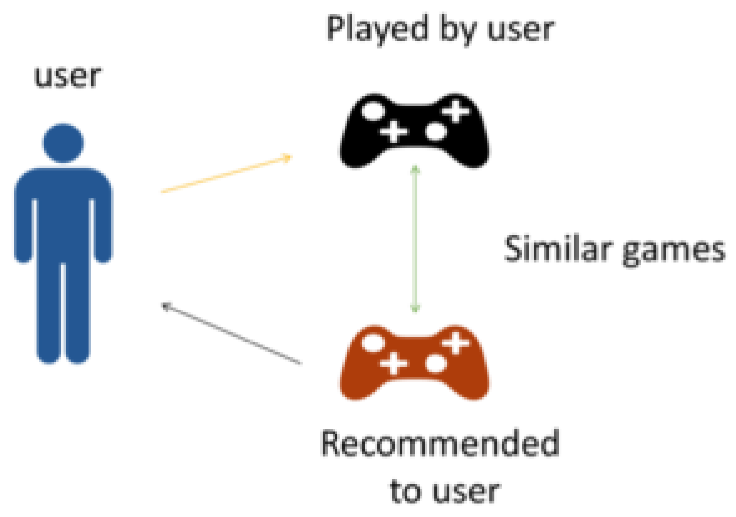
2			4	5	2.94*
5		4			1
		5		2	2.48*
	1		5		4
		4			2
4	5		1		1.12*

What information the recommendation is based on?

- User related data
  - Age
  - Location
  - Profession
- Item related data (e.g. regarding a movie)
  - Budget
  - Genre
  - Director, actors
- User-item ratings
  - Ratings given by the users for some items

Content based approach

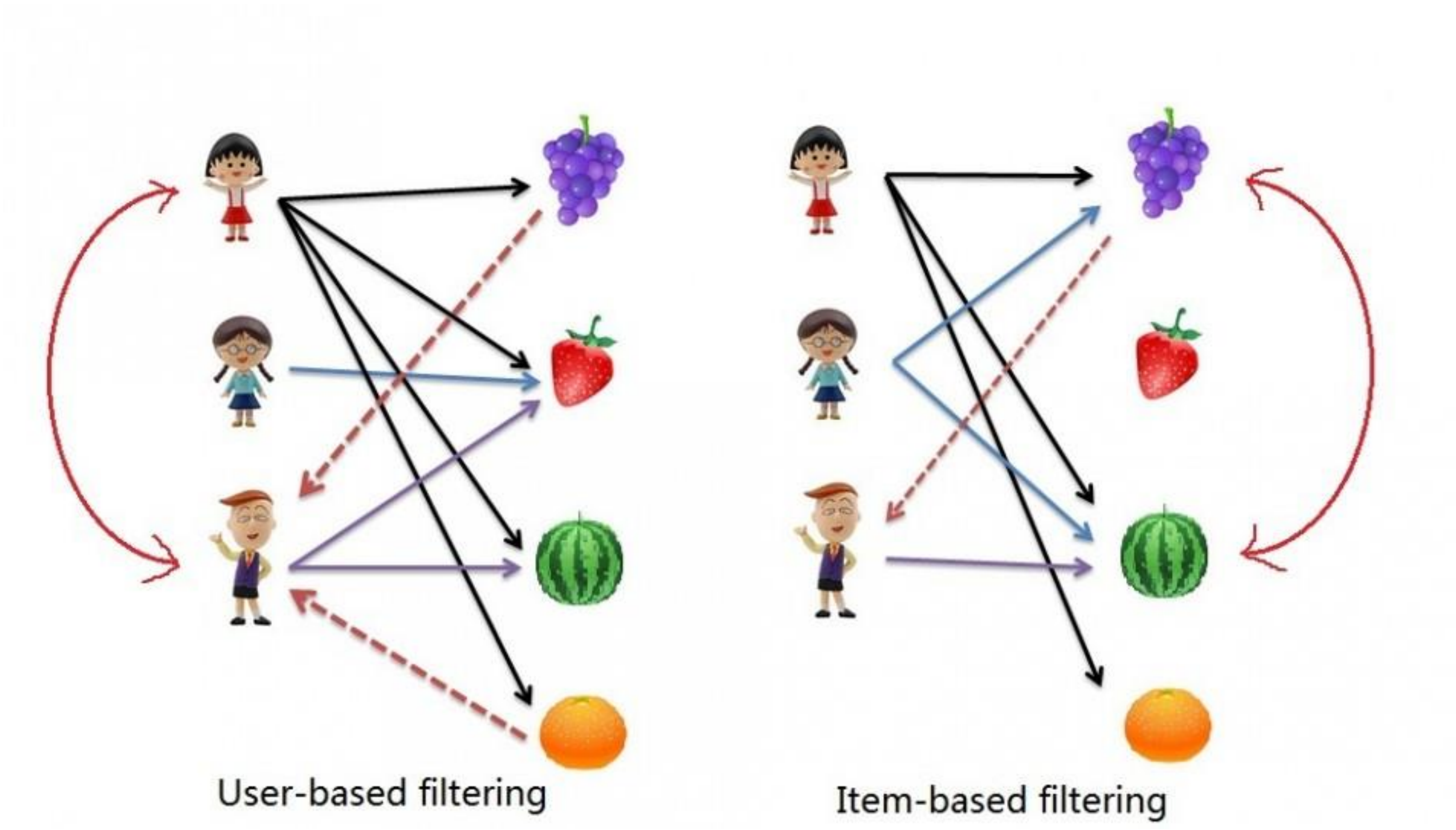
- The recommendation is based on the previous ratings of the users and on the similarity of items
- The items are characterized by some attributes
  - We define similarity between the items based on the attributes
- An item is recommended to a user if it is similar to an item that the user rated highly
- Limitations
  - What about new users?
  - Do not include attributes of users
  - What metric use for similarity between items?



Collaborative filtering

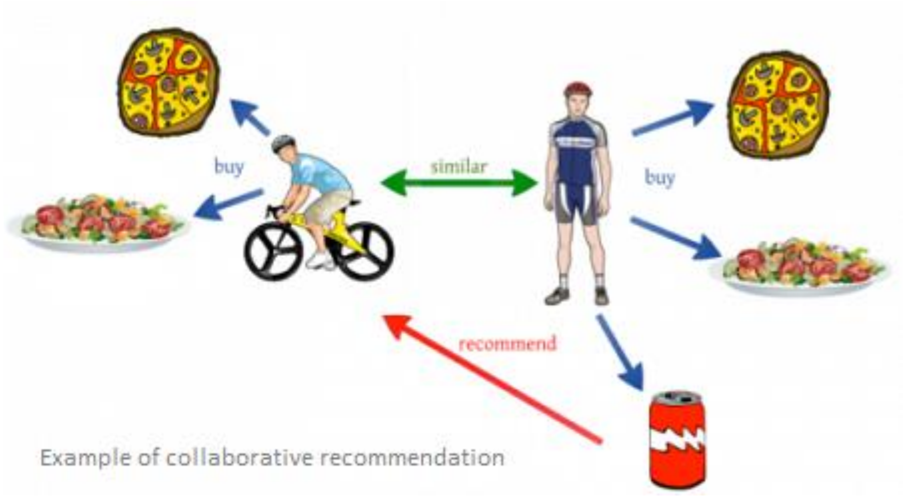
- Principle: Individual preferences are correlated
  - If Alice likes X and Y, and Bob likes X, Y and Z, then it is more likely that Alice likes Z
- Collaborative filtering does not rely on the characteristics of users or items, solely on the ratings
  - We don’t need any additional information on the users/items
- User-based collaborative filtering
  - The recommendation is based on the ratings of users with similar taste
- Item-based collaborative filtering
  - Prediction is based on the similarity between items using user’s ratings on those items
- Hybrid method
  - Combine the two approaches

Collaborative filtering II.



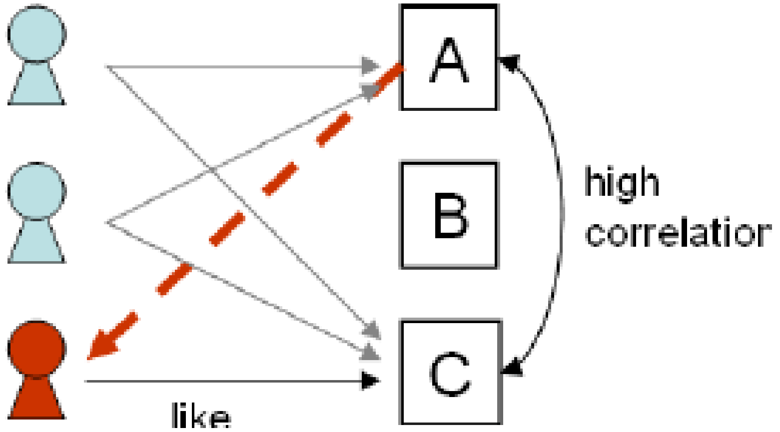
User-based collaborative filtering

- Principle: A user likes the items that are liked by users who are similar to him/her
  - Similarity is based on the ratings
- The predicted rating may be determined by a kNN approach
  - The predicted rating of a given item from a certain user is the average of the ratings of the *k* most similar users who rated the given item



Item-based collaborative filtering

- We are looking for similar items
  - The similarity is based on the ratings of users
- We recommend items that are similar to the ones that were liked by the user
- It is advantageous if there are more users than items



The model of collaborative filtering

- Collaborative filtering is solely based on the ratings stored in the rating matrix (user-item interaction matrix)
- This matrix is a sparse matrix with lots of missing entries

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

Rows correspond to users

The entries correspond to the known ratings of the items given by the users

Columns correspond to items

How to fill in missing entries?

- Goal: predicting the missing entries, i.e. predicting the ratings of the users
  - If the predicted rating is high, we recommend the item

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	
User 1	5	?	1	?	?	...
User 2	?	?	5	?	4	...
User 3	5	4	2	?	?	...
User 4	?	3	?	2	5	...
User 5	1	?	5	?	4	...
User 6	5	4	?	?	2	...
	...	...	...	...	...	...

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

Nearest neighbor based methods

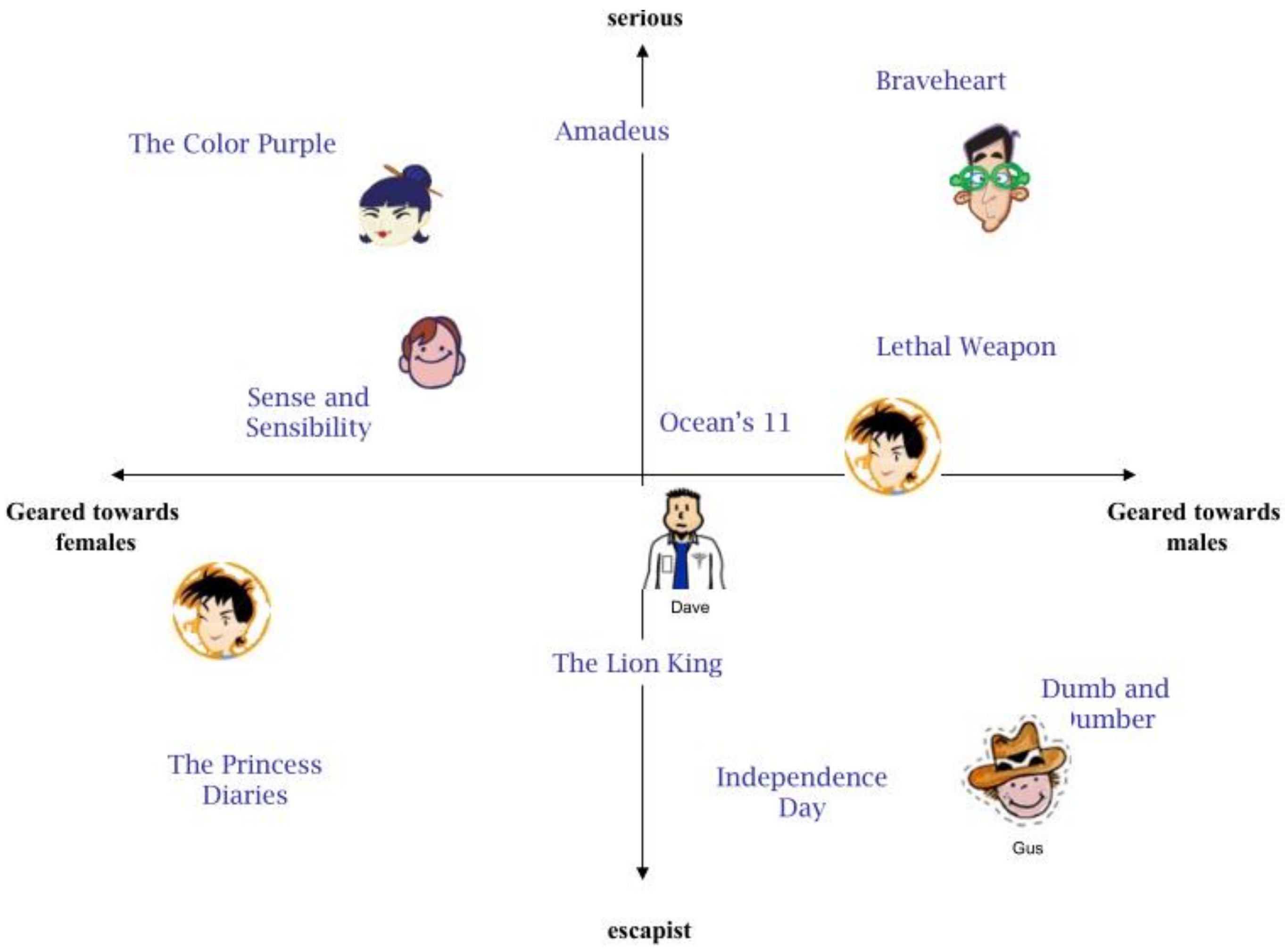
- User-based and items-based approaches are both possible
- User-based approach:
  - User is represented by an (incomplete) row vector
  - We consider his/her *k* nearest neighbors (with a chosen dissimilarity) who rated the given item
  - We take the (weighted) average of the ratings of *k* users
    - This is basically a kNN regression
    - Other regression methods are also possible

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	
User 1	5	?	1	?	?	...
User 2	?	?	5	?	4	...
User 3	5	4	2	?	?	...
User 4	?	3	?	2	5	...
User 5	1	?	5	?	4	...
User 6	5	4	?	?	2	...
	...	...	...	...	...	...

Latent factors

- We transform the items and users to a smaller dimensional space spanned by some latent factors
- Idea: The ratings of the items depend on these latent factors
- For each item and rating, the model assigns weights to the latent factors
  - The weights are extracted from the user-item interaction matrix
- The latent factors (e.g. for movies) can be interpreted as how romantic, how adventurous they are

Illustrating latent factors



Matrix factorization

- We estimate the rating matrix *X* as the product of two matrices
- Based on the known entries of *X* we are looking for *U* and *V* in such a way that their product approximates the known elements of *X* as closely as possible

$$\underset{U}{\begin{bmatrix} 2 & 1 \\ 2 & 2 \\ 3 & 2 \\ 1 & 1 \\ \dots & \dots \end{bmatrix}} \times \underset{V}{\begin{bmatrix} 2 & 2 & 1 & 3 & \dots \\ 1 & 0 & 3 & 3 & \dots \end{bmatrix}} \approx \underset{X}{\begin{bmatrix} 5 & ? & 4 & ? & \dots \\ ? & 4 & ? & ? & \dots \\ ? & 5 & 4 & ? & \dots \\ 4 & ? & 4 & 5 & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}}$$

Matrix factorization

- Number of lazent factors: K
- Hypothesis : $X \approx UV, \qquad U \in \mathbb{R}^{m\times K}, V \in \mathbb{R}^{K\times n}$

$$U = \begin{bmatrix} -u_1^T - \\ \vdots \\ -u_m^T \end{bmatrix}, \qquad V = \begin{bmatrix} | & & | \\ v_1 & \cdots & v_n \\ | & & | \end{bmatrix}$$

- Cost function (MSE):

$$\sum_{i,j} \left(x_{i,j} - \sum_{k=0}^{K} u_{i,k} v_{k,j} \right)^2$$

  - Optimization method: (stochastic) gradient descent method
  - We can add the usual regularization term: $+\lambda\left(\sum_{i,j}\left(u_{i,j}^2 + v_{i,j}^2\right)\right)$

Netflix

Most Loved Movies	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

Most Rated Movies
Miss Congeniality
Independence Day
The Patriot
The Day After Tomorrow
Pretty Woman
Pirates of the Caribbean

Highest Variance
The Royal Tenenbaums
Lost In Translation
Pearl Harbor
Miss Congeniality
Napolean Dynamite
Fahrenheit 9/11

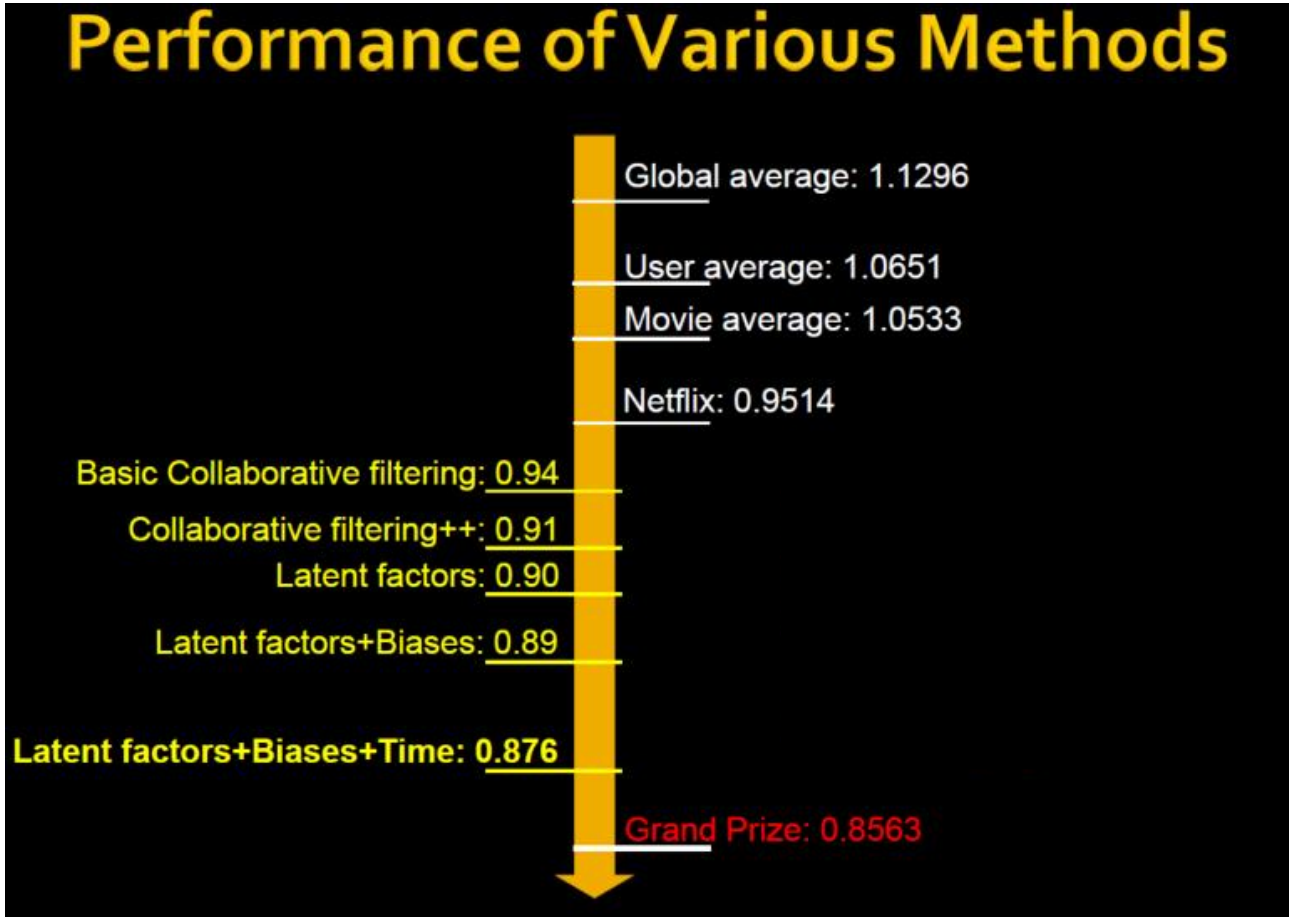
Netflix Prize

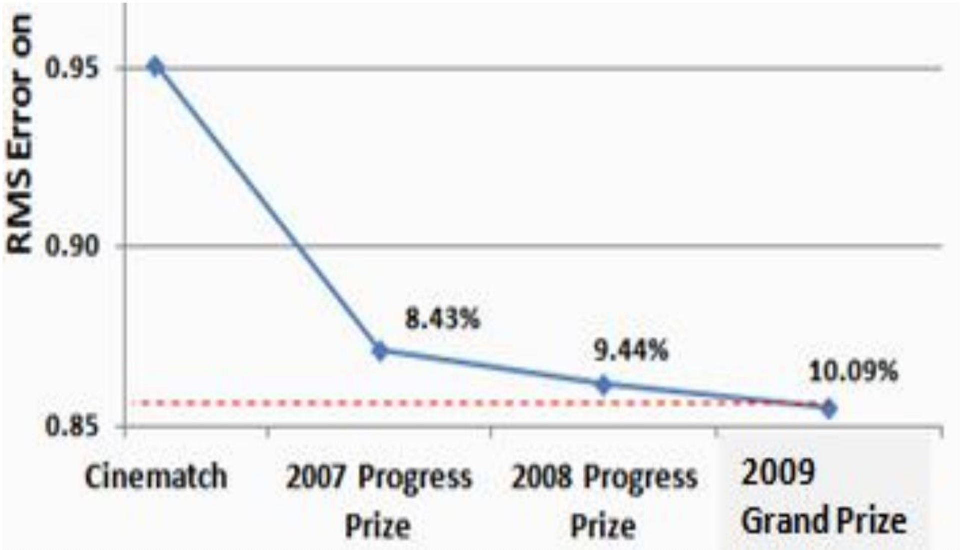
- Build a recommendation system
- Started in 2006
- Finished in 2009
- Prize: 1 million $
- The most famous data mining competition
- Goal: to improve the recommendation system of Netflix by 10%
- More than 2700 R&D teams started to work on the problem

Netflix data

- Training data:
  - 100 million movie ratings
  - 18 thousand movies and 480 thousand users
  - In average ~ 5600 ratings/movie
  - In average ~ 208 ratings/user
  - 6 years data (2000-2005)
  - Ratings are integers between 1 and 5
- Validation data:
  - The last few ratings for every user (2.8 million ratings)
  - Evaluation criteria: RMSE
    - Using the algorithm of Netflix - RMSE: 0.9514
      - 10% improvement would be an RMSE of 0.856
  - The true labels of the validation data are naturally hidden from the teams
    - The teams upload their predicted labels and the system evaluate their results

Improving results…





The leaderboard 30 days before the end date

NETFLIX

Netflix Prize

Home | Rules | Leaderboard | Register | Update | Submit | Download

Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	BellKor's Pragmatic Chaos	0.8558	10.05	2009-06-26 18:42:37
Grand Prize - RMSE <= 0.8563				
2	PragmaticTheory	0.8582	9.80	2009-06-25 22:15:51
3	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
4	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
5	Dace	0.8604	9.56	2009-04-22 05:57:03
6	BigChaos	0.8613	9.47	2009-06-23 23:06:52
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
7	BellKor	0.8620	9.40	2009-06-24 07:16:02
8	Gravity	0.8634	9.25	2009-04-22 18:31:32
9	Opera Solutions	0.8638	9.21	2009-06-26 23:18:13
10	BruceDengDaoCiYiYou	0.8638	9.21	2009-06-27 00:55:55
11	pengpengzhou	0.8638	9.21	2009-06-27 01:06:43
12	xlvector	0.8639	9.20	2009-06-26 13:49:04
13	xiangliang	0.8639	9.20	2009-06-26 07:47:34

The last 30 days

- A new team is formed called Ensemble
  - By the merger of a few teams from the top 10
  - They combine their achievements and try to beat the leader, BellKor
- BellKor
  - Manage to have further little improvements
  - Also realize that Ensemble is a dangerous competitor
- Strategy
  - Both teams have their eyes on the leaderboard
  - The only way they can check whether a new method improves the result is to upload, but then the other teams also get informed

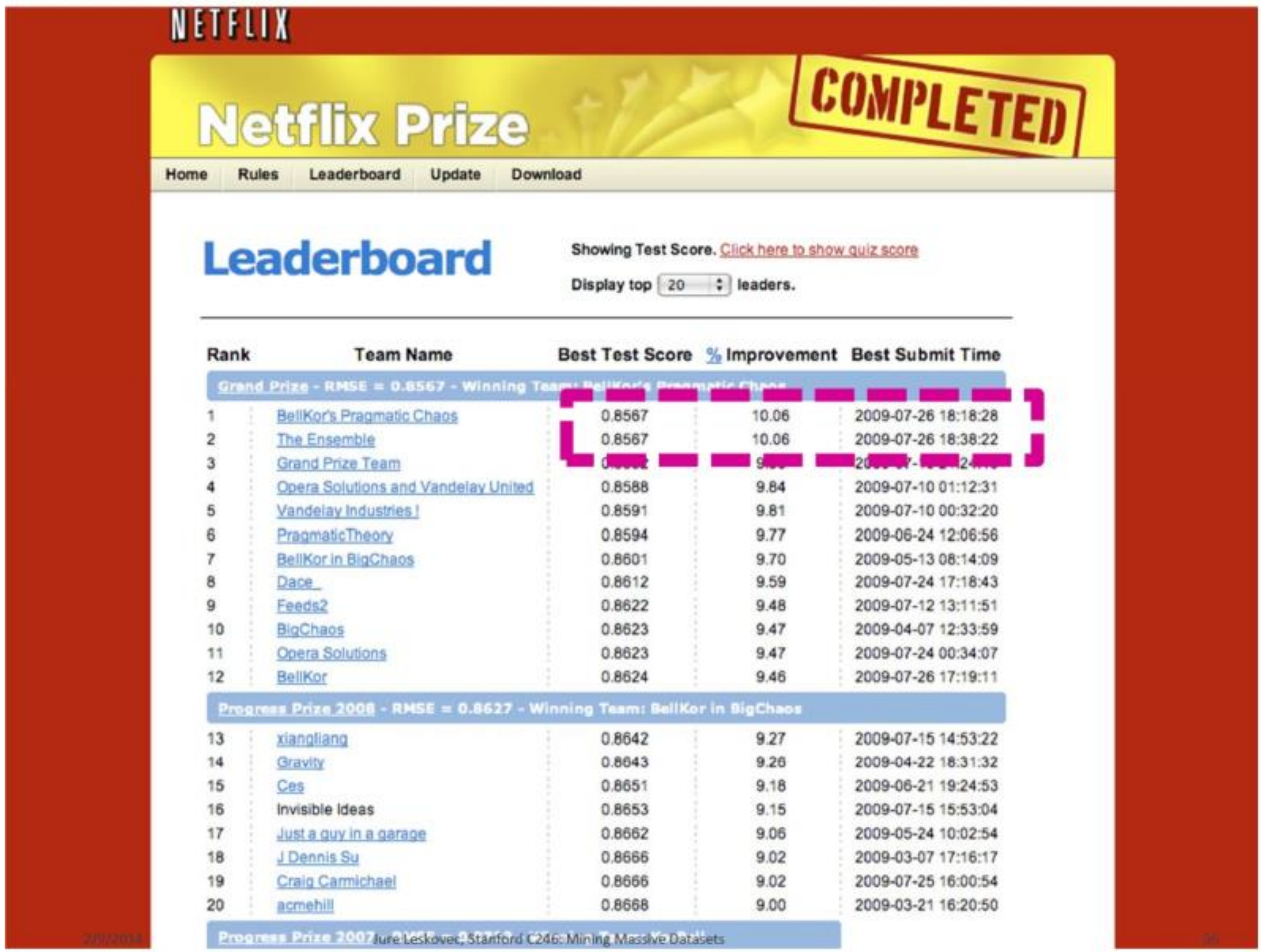
The last 24 hours

- New rule at the end: 1 upload/day
- It means that in the last 24 hours a team can only upload once
- 24 hours to the end, BellKor realizes that the Ensemble team has better result

- Crazy 24 hours have started, both teams try t hurry
- 1 hour before the deadline both teams are ready
  - At what time should they upload their results?
  - Bellkor upload their results 40 min before the deadline
  - Ensemble upload their results 20 min before the deadline

And everybody is waiting…

The final results



NETFLIX

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
20	acmehill	0.8668	9.00	2009-03-21 16:20:50

Progress Prize 2007

And the prize goes to…

Million $ Awarded Sept 21st 2009

Was it worth it?

3 Years Later…

"We evaluated some of the new methods offline but the **additional accuracy gains** that we measured **did not seem to justify the engineering effort** needed to bring them into a **production environment**."

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229