Neighbor methods vs. matrix factorization:

case studies of real-life recommendations

By István Pilászy, András Serény, György Dózsa, Balázs Hidasi, Attila Sári, János Gub

Presented by:
István Pilászy <pila@gravityrd.com>
at RecSys'15

GRAVITY
Research & Development

About Gravity R&D

- Founded by 4 people participating in the Netflix Prize competition, tied for the first place
- VC Investment in 2009 and 2011
- Providing Recommendation as a Service
- Clients in 6 continents

About the Netflix Prize

- 2006-2009
- Predict movie ratings (explicit feedback)
- CBF did not work
- Classical CF methods (item-kNN, user-kNN) did not work
- Matrix Factorization was extremely good
- We were fully in love with Matrix Factorization

Case study #1: Netflix Prize demo

GRAVITY
Research & Development

Netflix Prize demo / 1

- In 2009 we created a public demo mainly for investors
- Users can rate movies and get recommendations
- What do you expect from a demo?
  - Be relevant even after 1 rating
  - Users will provide their favorite movies first
  - Be relevant after 2 ratings: both movies should affect the results

Netflix Prize demo / 2

- Using a good MF model with K=200 factors and biases
- Use linear regression to compute user feature vector
- Recs after rating a romantic movie Notting Hill, 1999

OK	Score	Title
✓	4.6916	The_Shawshank_Redemption/1994
✗	4.6858	House,_M.D.:_Season_1/2004
✗	4.6825	Lost:_Season_1/2004
✓	4.5903	Anne_of_Green_Gables:_The_Sequel/1987
✗	4.5497	Lord_of_the_Rings:_The_Return_of_the_King/2003

Netflix Prize demo / 3

- Idea: turn off item bias during recommendation.
- Result are fully relevant
- Even with 10 factors, it is very good

OK	Score	Title
✓	4.3323	Love_Actually/2003
✓	4.3015	Runaway_Bride/1999
✓	4.2811	My_Best_Friend's_Wedding/1997
✓	4.2790	You've_Got_Mail/1998
✓	4.1564	About_a_Boy/2002

Netflix Prize demo / 4

- Now give 5-star rating to Saving Private Ryan / 1998
- Almost no change in the list

OK	Score	Title
✓	4.5911	You've_Got_Mail/1998
✓	4.5085	Love_Actually/2003
✓	4.3944	Sleepless_in_Seattle/1993
✓	4.3625	Runaway_Bride/1999
✓	4.3274	My_Best_Friend's_Wedding/1997

Netflix Prize demo / 5

- Idea: set item biases to zero before computing user feature vector
- 5th rec is romantic + war
- Conclusion: MF is good, but rating and ranking are very different

OK	Score	Title
✓	4.5094	You've_Got_Mail/1998
✓	4.3445	Black_Hawk_Down/2001
✓	4.3298	Sleepless_in_Seattle/1993
✓	4.3114	Love_Actually/2003
✓!	4.2805	Apollo_13/1995

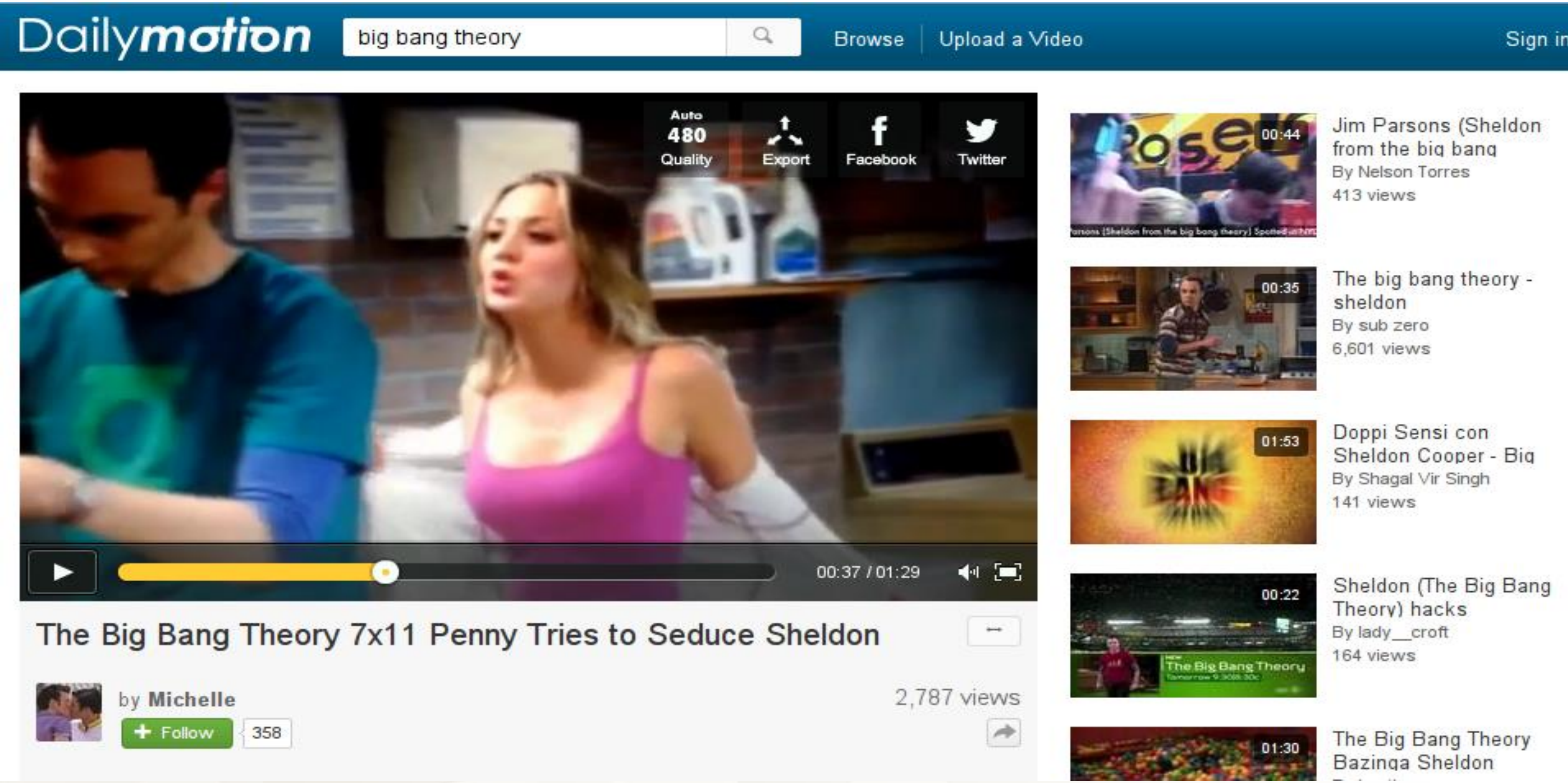
About real life problems

GRAVITY
Research & Development

About real life problems: item2item recs

- What is item-to-item recommendation?
  - People who viewed this also viewed: …
  - Viewed, watched, bought, liked, favorited, etc.

- Ignoring the current user
- The recommendation should be relevant to the current item
- Very common scenario


Dailymotion
big bang theory
Browse
Upload a Video
Sign in
Auto
480
Quality
Export
Facebook
Twitter
00:37 / 01:29
The Big Bang Theory 7x11 Penny Tries to Seduce Sheldon
by Michelle
Follow
358
2,787 views
00:44
Jim Parsons (Sheldon from the big bang
By Nelson Torres
413 views
00:35
The big bang theory - sheldon
By sub zero
6,601 views
01:53
Doppi Sensi con Sheldon Cooper - Big
By Shagal Vir Singh
141 views
00:22
Sheldon (The Big Bang Theory) hacks
By lady__croft
164 views
The Big Bang Theory
01:30
The Big Bang Theory Bazinga Sheldon

About real life problems / 2

- In Academic papers
  - 50% explicit feedback
  - 50% implicit feedback
    - 49.9% personal
    - 0.1% item2item
- At gravityrd.com:
  - 1% explicit feedback
  - 99% implicit feedback
    - 15% personal
    - 84% item2item
- Sites where rating is crucial tend to create their own rec engine
- Even if there is explicit rating, there are more implicit feedback

About real life problems / 3

- Data characteristics (after data retention):
  - Number of active users: 100k – 100M
  - Number of active items : 1k – 100M
  - Number of relations between them: 10M – 10B
- Response time: must be within 200ms
- We cannot give 199ms for MF prediction + 1ms business logic

Time complexity of MF for implicit feedback

- During training
  - E = #events,   U = #users,   I = #items
  - implicit ALS: E*K^2 + U*K^3 + I*K^3
    - with Coordinate Descent: E*K + U*K^2 + I*K^2
    - with CG: the same, but more stable.
  - BPR: E * K
  - CliMF: O(E * K * avg(user support))
- During recommendation: I * K
- Not practical if I > 100k, K > 100
- You have to increase K as I grows

Case study #2: item-to-item recommendations with SVD

GRAVITY
Research & Development

i2i recommendations with SVD / 2

- Recommendations should seem relevant
- You can expect that movies of the same trilogy are similar to each other (tetralogies too)
- We defined the following metric:
  - For movies A and B of a trilogy, check if B is amongst the top-5 most similar items of A.
    Score: 0 or 1
  - A trilogy can provide 6 such pairs (12 for tetralogies)
  - Sum up this for all trilogies
- We used a custom movie dataset
- Good metric for CF item-to-item, bad metric for CBF item-to-item

i2i recommendations with SVD / 3

- Evaluating for SVD with different number of factors

K	10	20	50	100	200	500	1000	1500
score	72	82	95	96	106	126	152	158

- Using cosine similarity between SVD feature vectors
- more factors provide better results
- Why not use the original space?
- Who wants to run SVD with 500 factors?
- Score of neighbor method (using cosine similarity between original vectors): 169

I2i recommendations with SVD / 4

- What does a 200-factor SVD recommend to Kill Bill: Vol. 1
- Really bad recommendation

OK	Cos Sim	Title
✓	0.299	Kill Bill: Vol. 2
✘	0.273	Matthias, Matthias
✘	0.223	The New Rijksmuseum
✘	0.199	Naked
✘	0.190	Grave Danger

i2i recommendations with SVD / 5

- What does a 1500-factor SVD recommend to Kill Bill: Vol. 1
- Good, but uses lots of CPU
- But that is an easy domain, with 20k movies!

OK	Cos Sim	Title
✓	0.292	Kill Bill: Vol. 2
✓!	0.140	Inglourious Basterds
✓!	0.133	Pulp Fiction
✘	0.131	American Beauty
✓!	0.125	Reservoir Dogs

Case Study #3: Implementing an item-to-item method

GRAVITY
Research & Development

Implementing an item-to-item method / 1

We implemented the following article:

Noam Koenigstein and Yehuda Koren. "Towards scalable and accurate item-oriented recommendations." Proceedings of the 7th ACM conference on Recommender systems. ACM, 2013.

- They define a new metric for i2i evaluation:
  MPR (Mean Percentile Rank):
  If user visits A, and then B, then recommend for A, and see the position of B in that list.
- They propose a new method (EIR, Euclidean Item Recommender) , that assigns feature vector for each item, so that if A is close to B, then users frequently visit B after A.
- They don't compare it with pure popularity method

Implementing an item-to-item method / 2

Results on a custom movie dataset:

- SVD and other methods can't beat the new method
- **Popularity method is better or on-pair with the new method**
- Recommendations for Pulp Fiction:

SVD	New method
Reservoir Dogs	A Space Odyssey
Inglourious Basterds	A Clockwork Orange
Four Rooms	The Godfather
The Shawshank Redemption	Eternal Sunshine of the Spotless Mind
Fight Club	Mulholland Drive

GRAVITY Research & Development

Implementing an item-to-item method / 3

Comparison

method	metadata similarity (larger is better)	MPR (smaller is better)
cos	**7.54**	0.68
jaccard	**7.59**	0.68
Assoc.rule	6.44	0.68
pop	1.65	**0.25**
random	1.44	0.50
EIR	5.00	**0.25**

Implementing an item-to-item method / 4

Summary of this method:

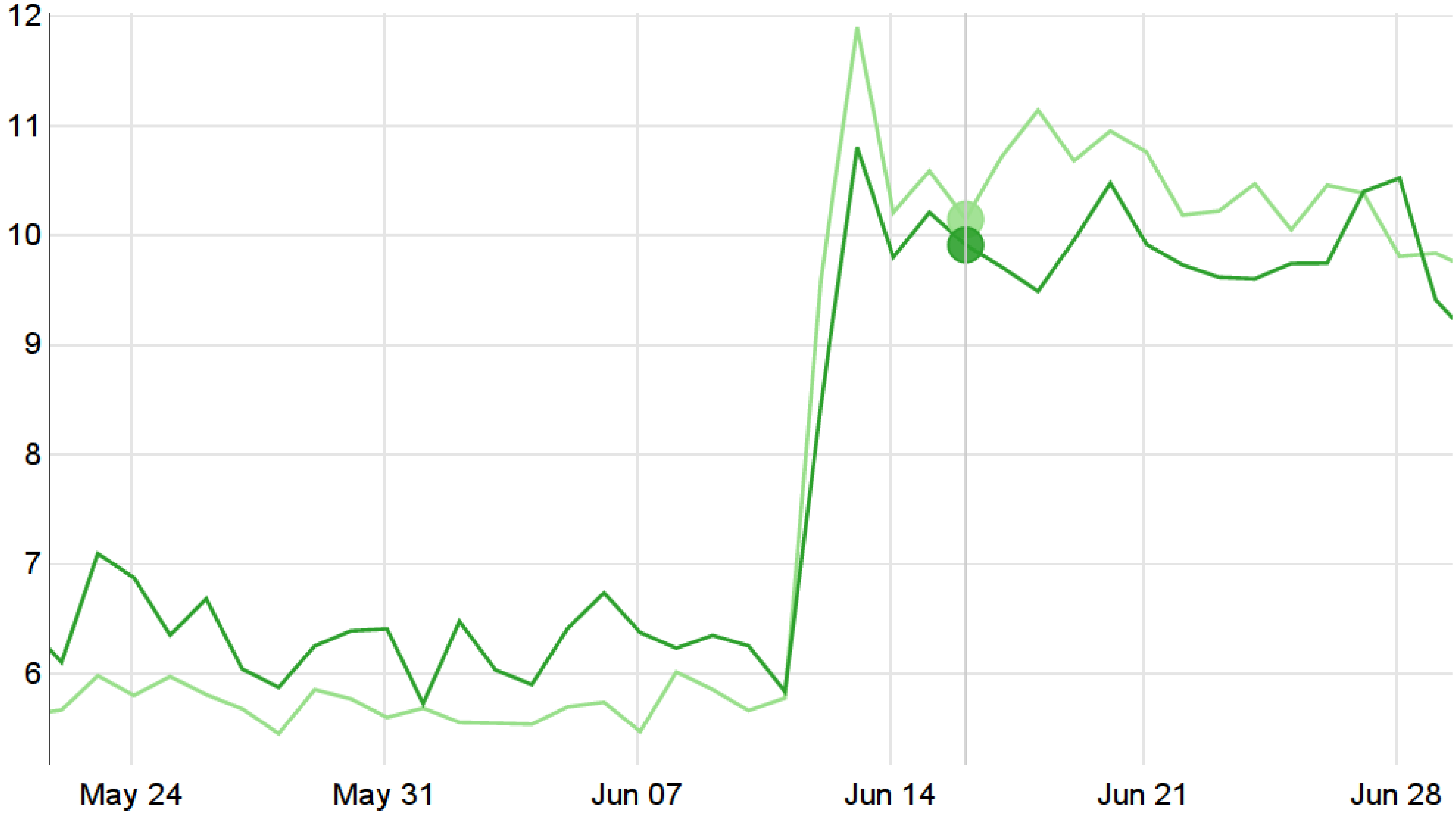
- This method is better in MPR than many other methods
- It is on pair with Popularity method
- It is worse in metadata-based similarity
- Sometimes recommendations look like they were random
- Sensitive to the parameters
- Very few articles are dealing with CF item-to-item recs

Case studies on CTR

GRAVITY
Research & Development

Case studies on CTR / 1

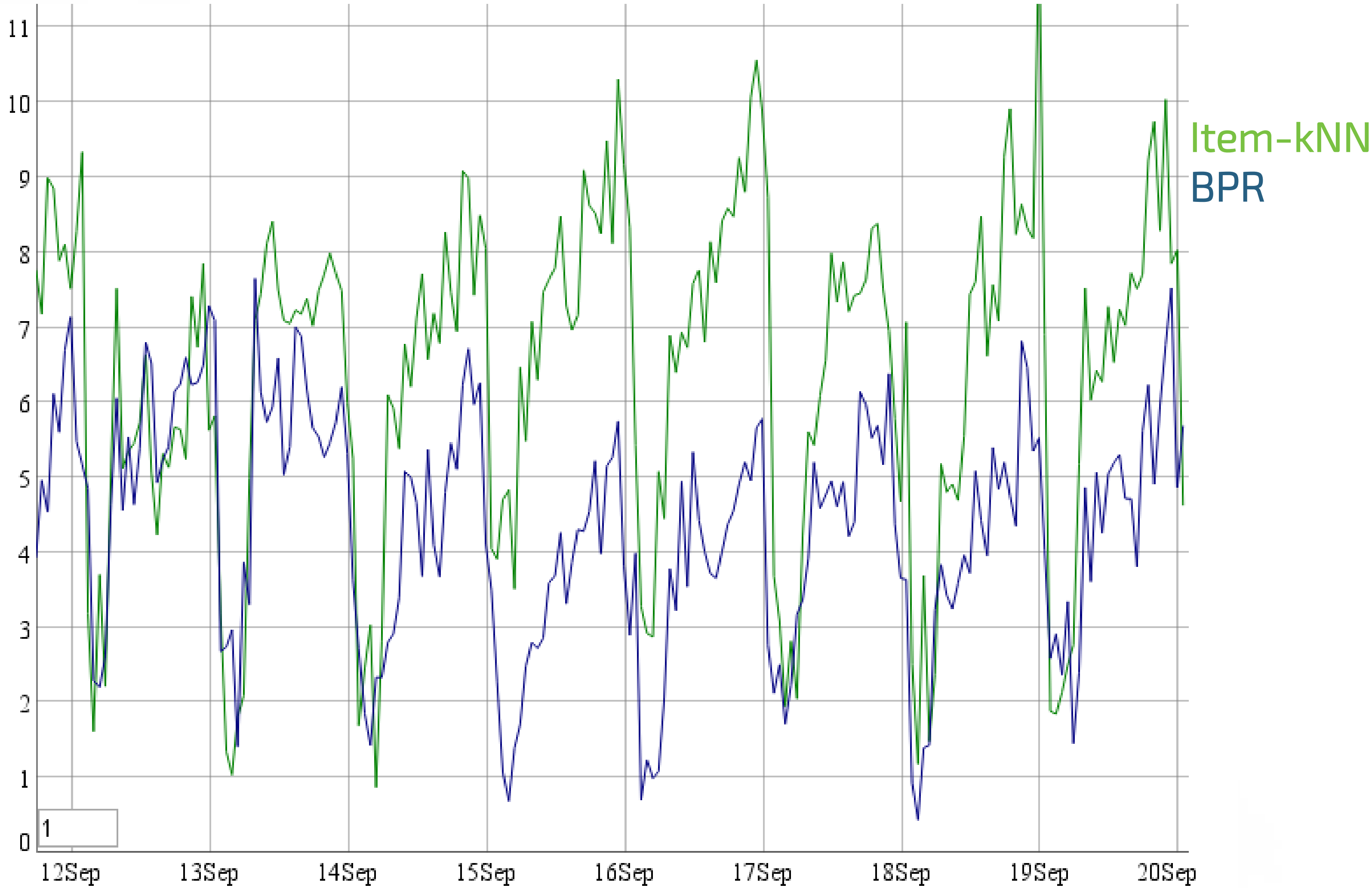
CTR almost doubled when we switched from IALS1 to item-kNN on a site where users and items are the same


12
11
10
9
8
7
6
May 24
May 31
Jun 07
Jun 14
Jun 21
Jun 28

Case studies on CTR / 2

Comparison of BPR vs. item-kNN on a classified site, for item-to-item recommendations.

Item-kNN is the winner



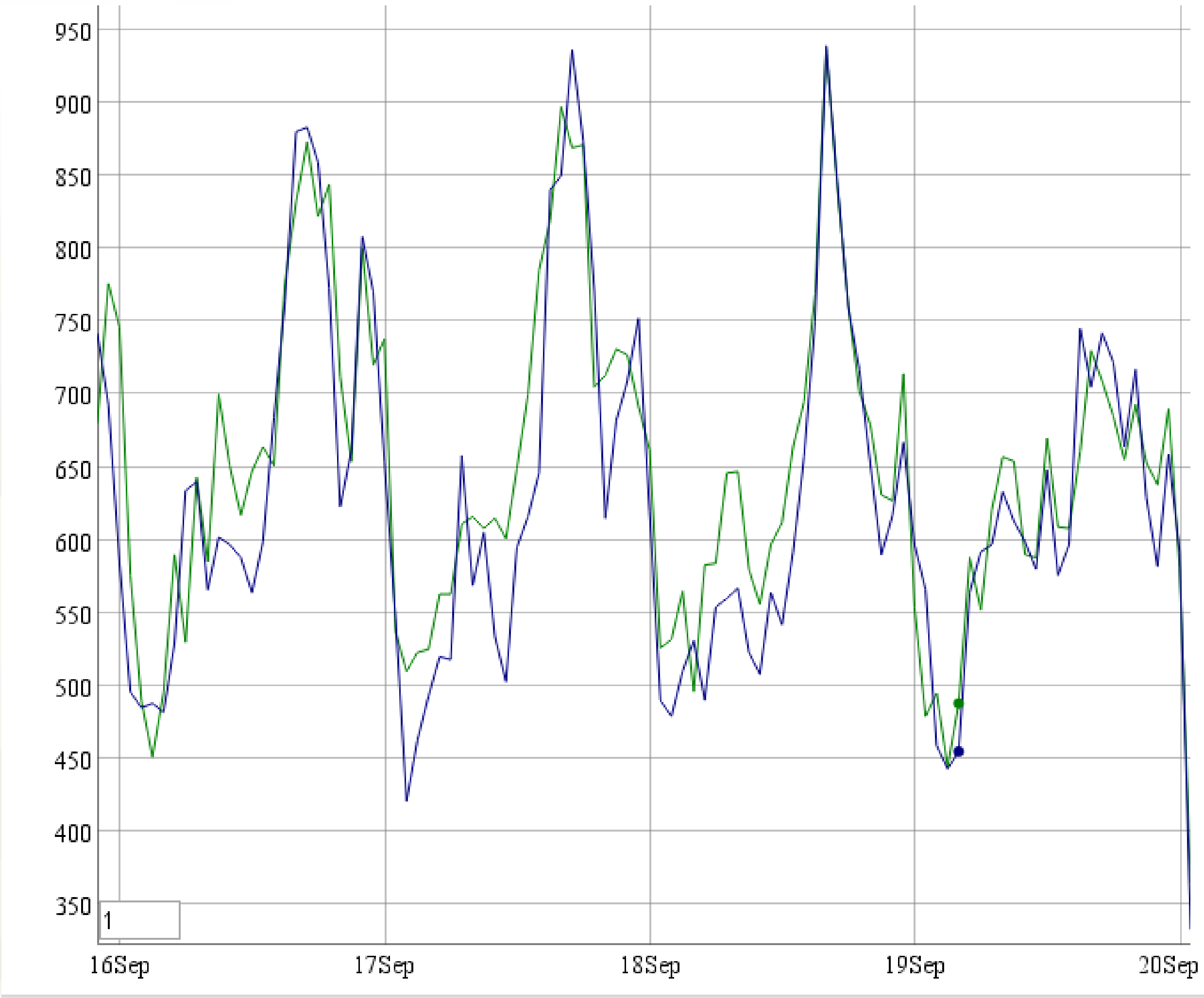
GRAVITY Research & Development

Case studies on CTR / 3

Using BPR vs. item-kNN on a video site for personal recommendations.

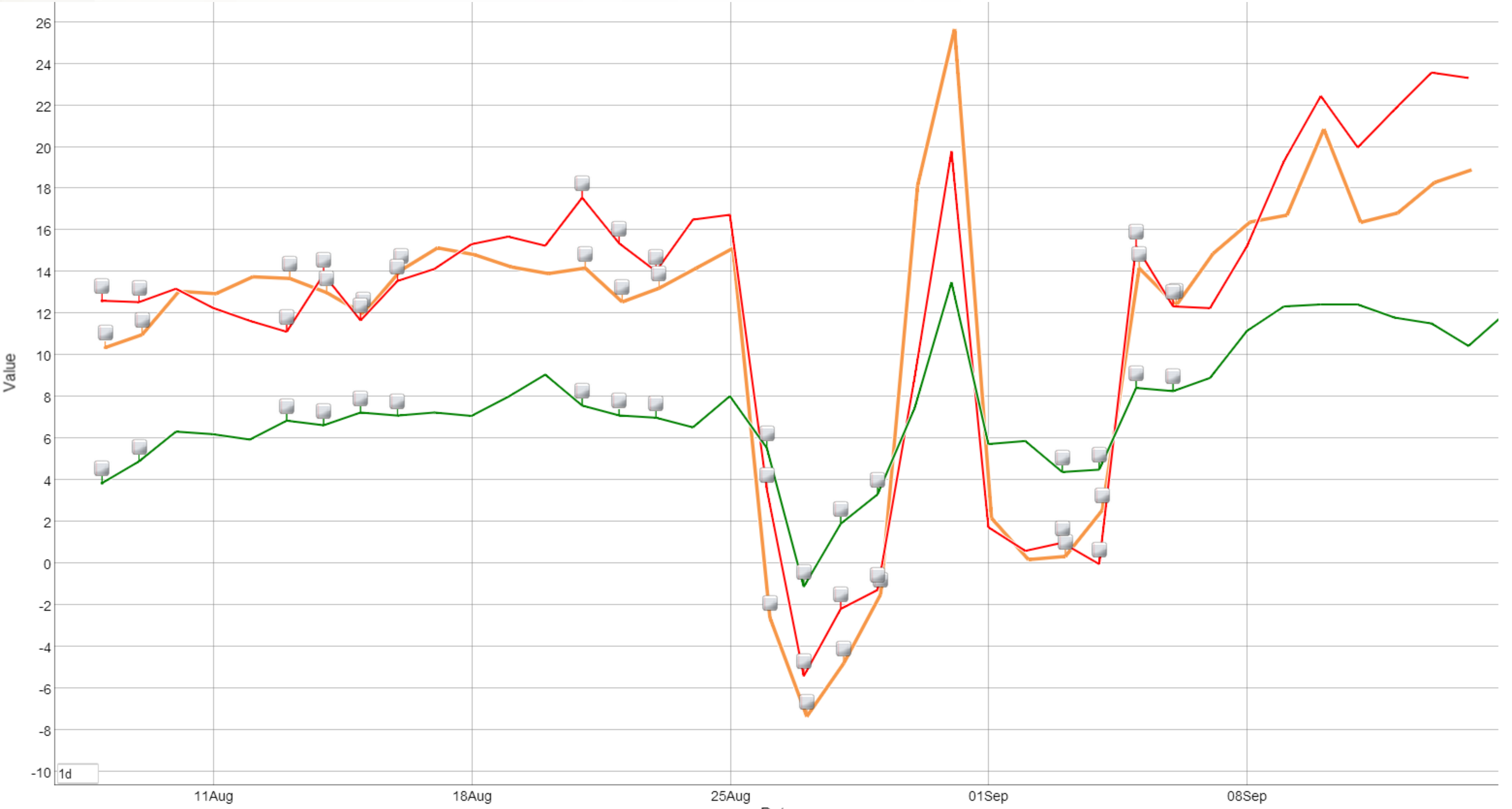
Measuring number of clicks on recommendations.

Result: 4% more clicks for BPR.


950
900
850
800
750
700
650
600
550
500
450
400
350
1
16Sep
17Sep
18Sep
19Sep
20Sep

BPR
Item-kNN

Case studies on CTR / 4



Conclusions / 1

Problems with MF:

- Lots of parameters to tune
- Needs many iteration over the data
- if there is no inter-connection between two item sets, they can get similar feature vectors.
- Sensitive to noise in data (e.g. users with few events)
- Not the best for item-to-item recs, especially when many neighbors exist already

Conclusions / 2

When to use Matrix Factorization

- you have one, dense domain (e.g. movies), with not too many items (e.g. less than 100k)
- feedback is taste-based
- For personalized recommendations (e.g. newsletter)
- try both with A/B testing
- try blending
  - Must be smart in blending (e.g. using it for high supported items)
- Usually better for offline evaluation metrics