# Evaluating Recommender Behavior For New Users

Daniel Kluver and Joe Konstan
October 6th, 2014

# Core Recsys Question:

What is the best algorithm?

# Core Recsys Question:

What is the best algorithm for this context?

# Core Recsys Question:

What is the best algorithm for this context?

(For current users)

# Core Recsys Question:

What is the best algorithm for this context?

(For new users)

# Core Recsys Question:

How do we get ratings for new users?

What do we do with ratings from new users?

# Core Recsys Question:

What is the best algorithm for this context?

(For new users)

# Core Recsys Question:

What is the best algorithm for this context?

(For new users)

(Cold start)

# Cold Start

Kinda Cold Start (10-20 ratings)

# Cold Start

Kinda Cold Start (10-20 ratings)

Serious Cold Start (0-5 ratings)

# How do Algorithms Behave for users with few ratings?

# How do Algorithms Behave for users with few ratings?

- How well can different algorithms **predict future ratings** of new users?
- How well can different algorithms **rank and recommend** good items for new users?
- How do algorithms behave as measured by other metrics such as **popularity** for new users?

# Algorithms

3 common algorithms:

- A range of different approaches

– ItemItem
– UserUser
– Funk SVD

We will compare this against two baselines:

– ItemBaseline
– UserItemBaseline

# How do we answer our questions?

- Offline analysis
- MovieLens 1M dataset
  - Each user has at least 20 ratings
- Crossfold by user
  - Keep only n (1 – 19) ratings for test users

# Metrics

- Accurate Predictions / Good Ranking
  - RMSE
  - NDCG
- Other Properties
  - SeenItems@20
  - AveragePopularity@20
  - AILS@20
  - Spread@20
- Likable Recommendations
  - Precision@20
  - MAP@20
  - Fallout@20
  - MeanRating@20
  - RMSE@20

# Metrics

- Accurate Predictions / Good Ranking
  - RMSE
  - NDCG
- Other Properties
  - SeenItems@20
  - AveragePopularity@20
  - AILS@20
  - Spread@20
- Likable Recommendations
  - Precision@20
  - MAP@20
  - Fallout@20
  - MeanRating@20
  - RMSE@20

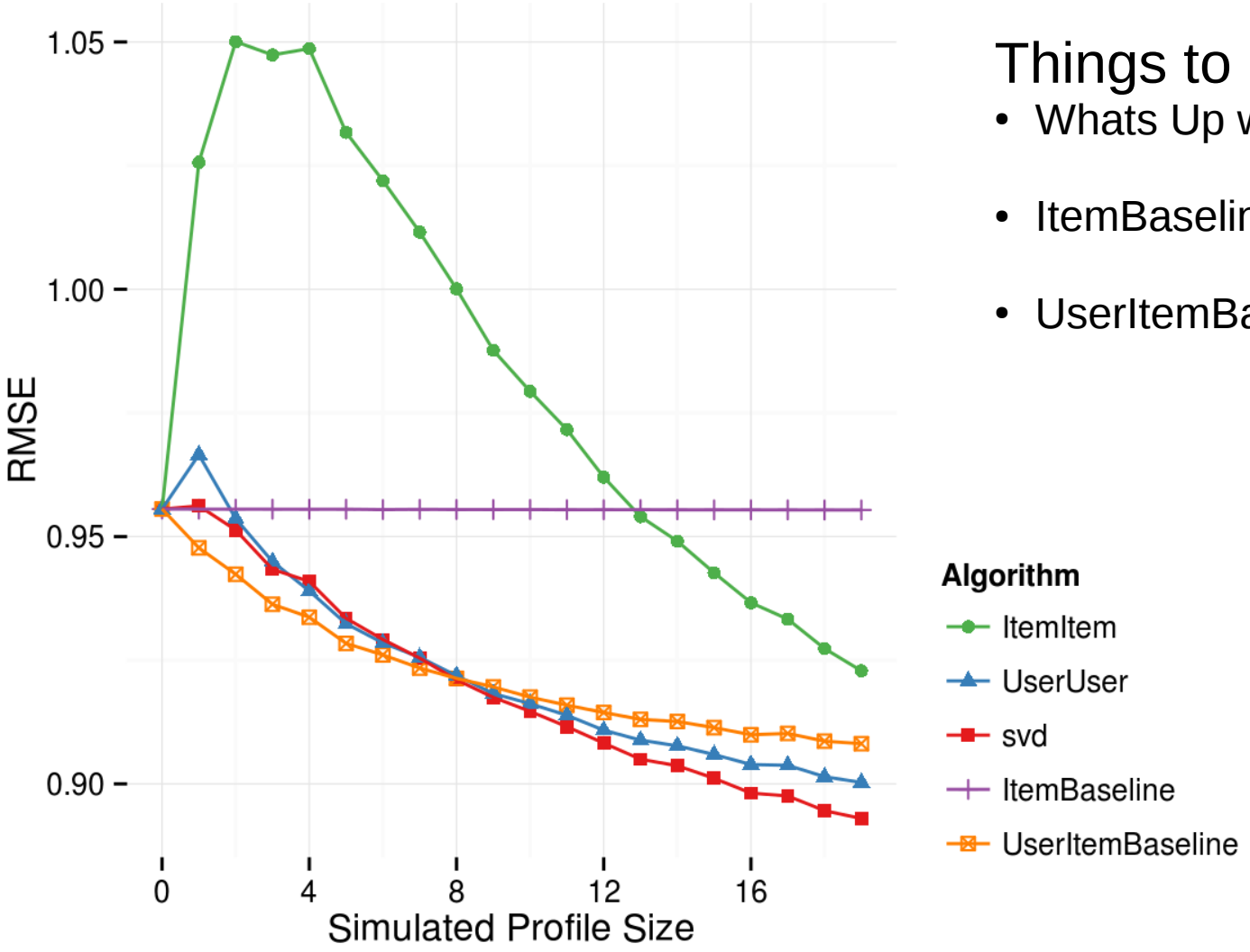
# Results

| Algorithm | Accuracy / Rank | Recommendation | Other Properties |
|---|---|---|---|
| ItemItem | | | |
| UserUser | | | |
| Funk SVD | | | |

Number of ratings to beat baseline

# RMSE

(Measures accuracy of predictions)

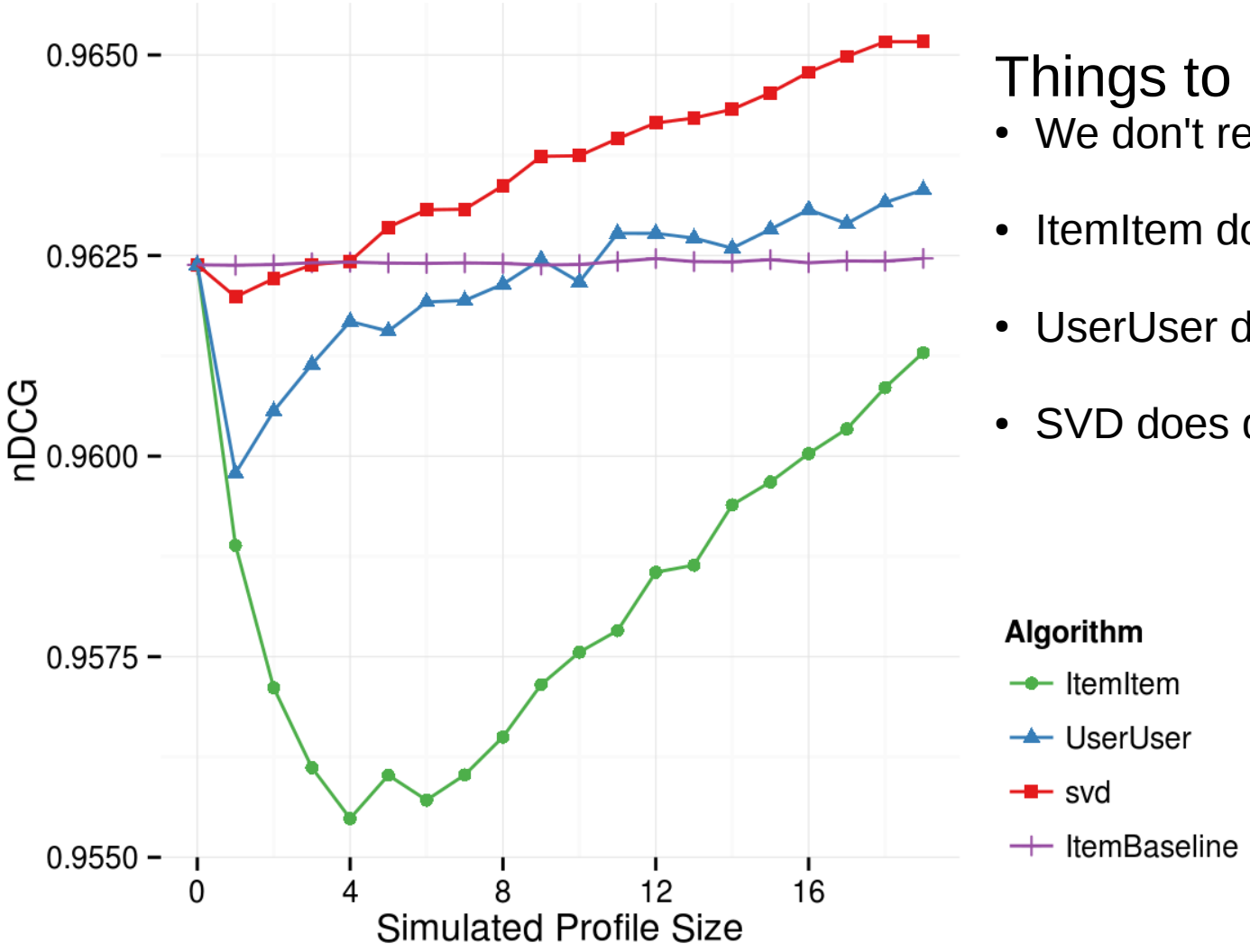



## Things to note:

- Whats Up with ItemItem?
- ItemBaseline Isn't very accurate
- UserItemBaseline is surprisingly accurate

# nDCG

(Measures how well the algorithms orders items based on ratings)




## Things to note:

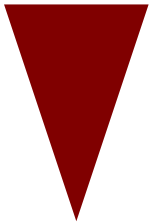
- We don't report UserItemBaseline
- ItemItem does bad
- UserUser does OK
- SVD does quite well

# Results

| Algorithm | Accuracy / Rank | Recommendation | Other Properties |
|---|---|---|---|
| ItemItem | **>19 (bad)** | | |
| UserUser | **9** | | |
| Funk SVD | **4 (Good)** | | |

Number of ratings to beat baseline

# Results

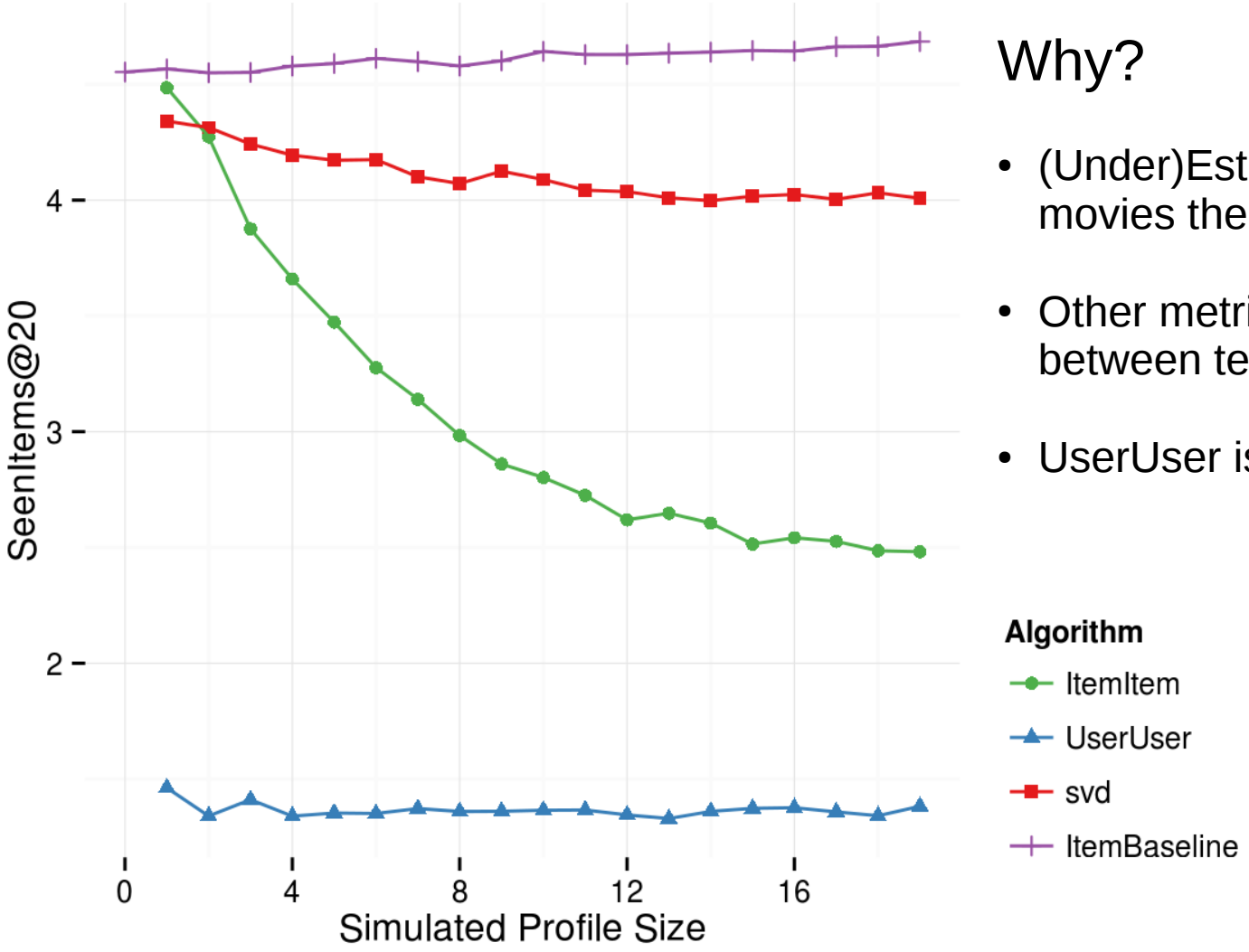


| Algorithm | Accuracy / Rank | Recommendation | Other Properties |
|---|---|---|---|
| ItemItem | **>19 (bad)** | | |
| UserUser | **9** | | |
| Funk SVD | **4 (Good)** | | |

Number of ratings to beat baseline

# SeenMovies@20

(Average number of Movies in both the test set and the top 20 recommendations)

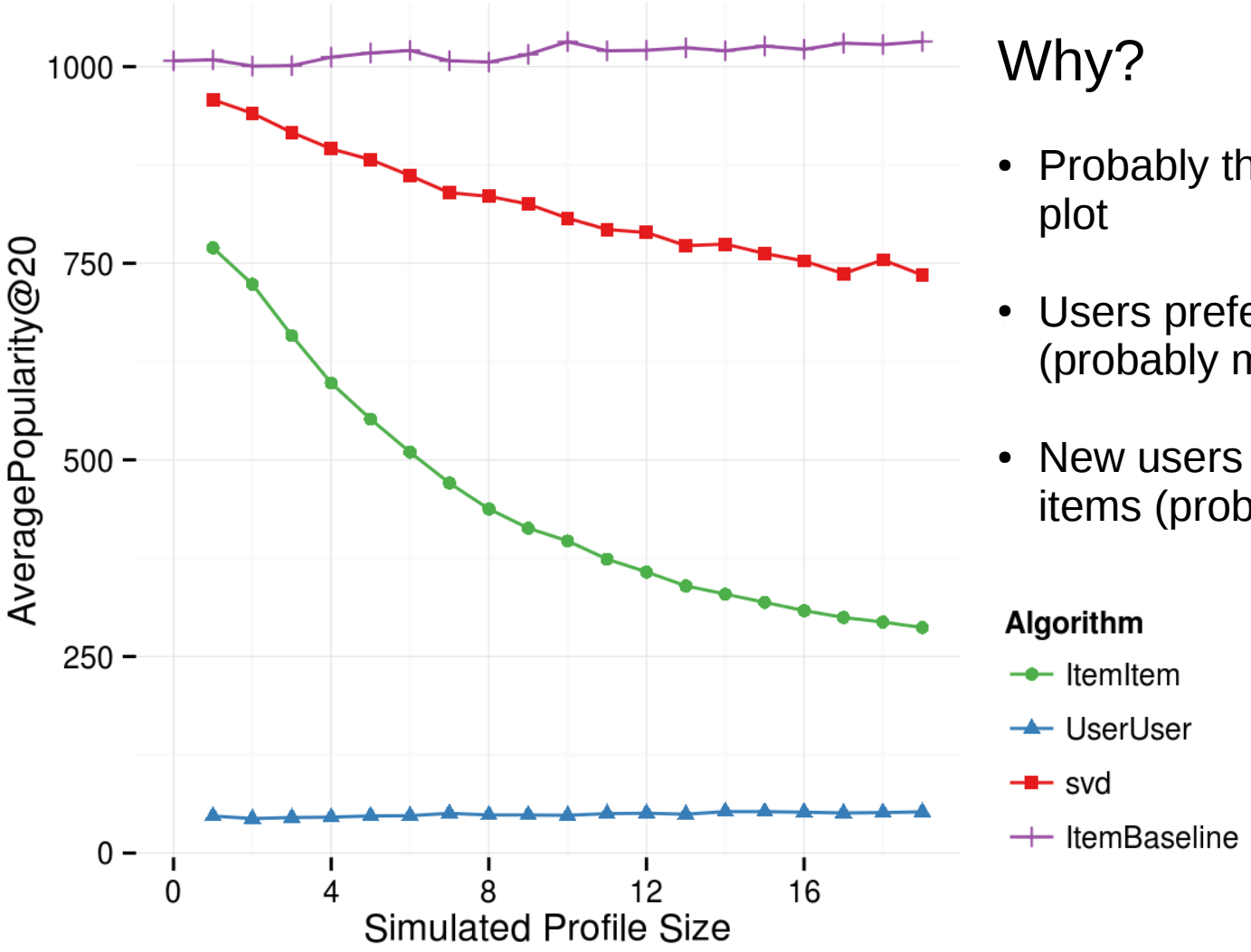



## Why?

- (Under)Estimate how many recommended movies the user has seen
- Other metrics look at the intersection between test set and recommendation
- UserUser is too small

# Popularity@20

(Average popularity of movies in the top 20 recommendations)




## Why?

- Probably the driving factor behind the last plot
- Users prefer a certain about of novelty (probably more than baseline)
- New users need to see some familiar items (probably more than UserUser)

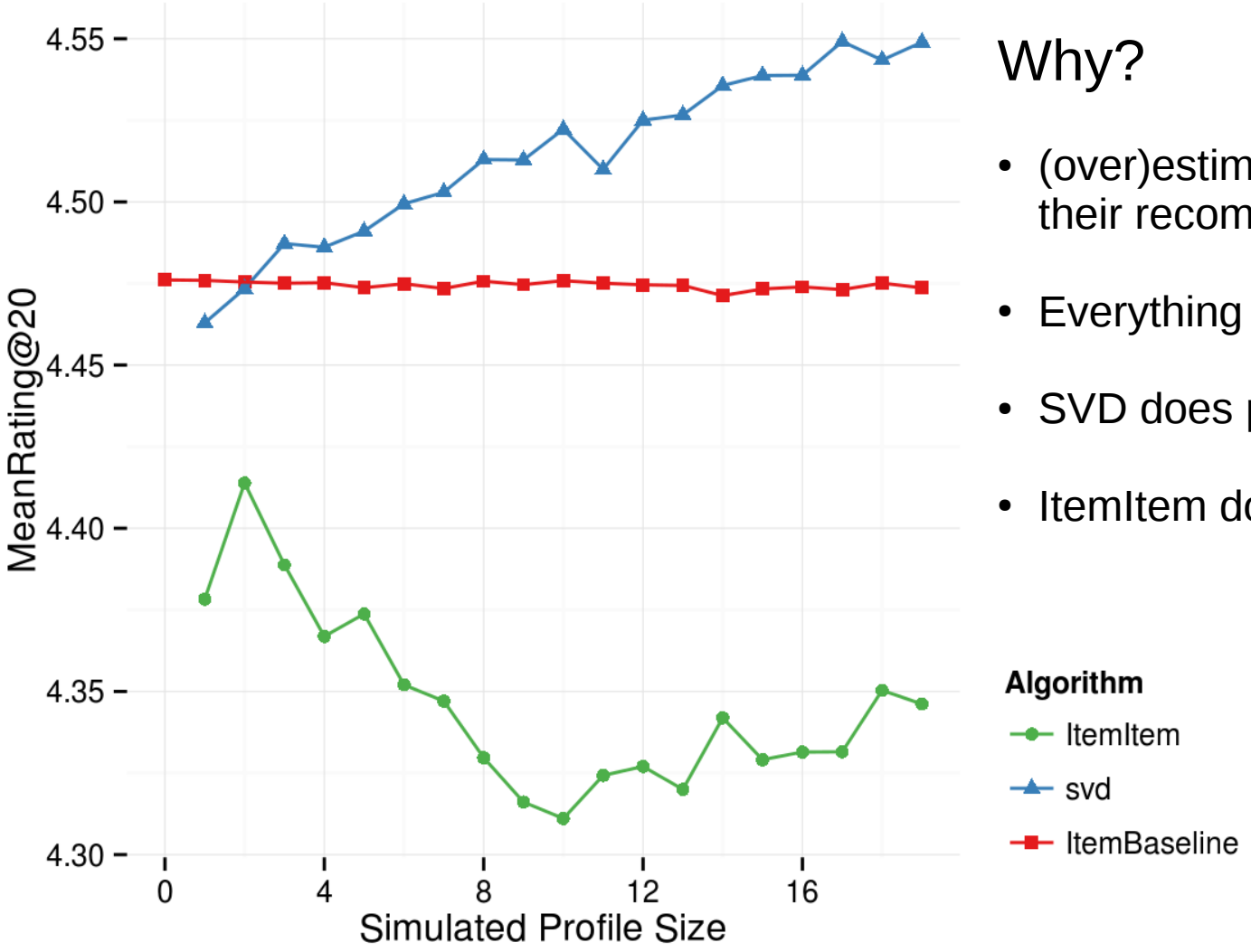
# Results

| Algorithm | Accuracy / Rank | Recommendation | Other Properties |
|---|---|---|---|
| ItemItem | **>19 (bad)** | | |
| UserUser | **9** | | **Too Obscure (bad)** |
| Funk SVD | **4 (Good)** | | **Too Popular?** |

Number of ratings to beat baseline

# MeanRating@20

(Average rating for items in the 20 recommendations)




## Why?

- (over)estimate how much the user likes their recommendations
- Everything is above 4 stars (yay!)
- SVD does pretty well
- ItemItem doesn't

# Results

| Algorithm | Accuracy / Rank | Recommendation | Other Properties |
|---|---|---|---|
| ItemItem | **>19 (bad)** | **>19 (bad)** | |
| UserUser | **9** | **?** | **Too Obscure (bad)** |
| Funk SVD | **4 (Good)** | **2 (Good)** | **Too Popular?** |

Number of ratings to beat baseline

# Conclusions

What did we learn?

- If you have less than 4 ratings user a baseline
- If you need a general algorithm that works well, use SVD
- UserUser can be used for its predictions, but beware its obscure recommendations.
- ItemItem should not be used for cold start users.

Questions?