**Author(s):** Rahul Sami, 2009

# Citation Key

for more information see: http://open.umich.edu/wiki/CitationPolicy

## Use + Share + Adapt

{ Content the copyright holder, author, or law permits you to use, share and adapt. }

PD-GOV **Public Domain – Government**: Works that are produced by the U.S. Government. (USC 17 § 105)

PD-EXP **Public Domain – Expired**: Works that are no longer protected due to an expired copyright term.

PD-SELF **Public Domain – Self Dedicated**: Works that a copyright holder has dedicated to the public domain.

ZERO **Creative Commons – Zero Waiver**

BY **Creative Commons – Attribution License**

BY-SA **Creative Commons – Attribution Share Alike License**

BY-NC **Creative Commons – Attribution Noncommercial License**

BY-NC-SA **Creative Commons – Attribution Noncommercial Share Alike License**

GNU-FDL **GNU – Free Documentation License**

## Make Your Own Assessment

{ Content Open.Michigan believes can be used, shared, and adapted because it is ineligible for copyright. }

PD-INEL **Public Domain – Ineligible**: Works that are ineligible for copyright protection in the U.S. (USC 17 § 102(b)) *laws in your jurisdiction may differ

{ Content Open.Michigan has used under a Fair Use determination. }

FAIR USE **Fair Use**: Use of works that is determined to be Fair consistent with the U.S. Copyright Act. (USC 17 § 107) *laws in your jurisdiction may differ

Our determination **DOES NOT** mean that all uses of this 3rd-party content are Fair Uses and we **DO NOT** guarantee that your use of the content is Fair.

To use this content you should **do your own independent analysis** to determine whether or not your use will be Fair.

# Lecture 12: Explanations; Scalable Implementation; Manipulation

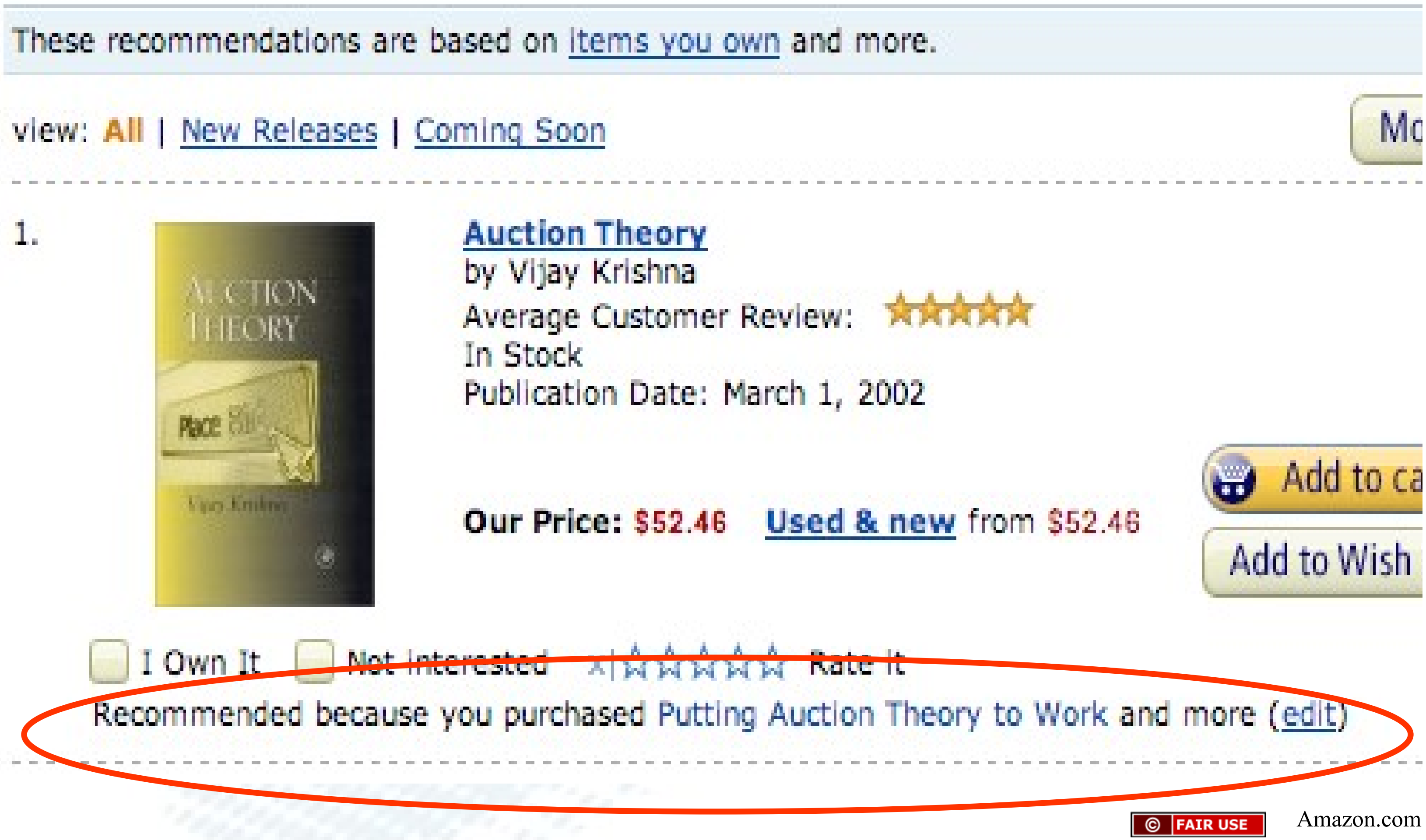
SI583: Recommender Systems



SCHOOL OF INFORMATION
UNIVERSITY OF MICHIGAN

# Explanations in recommender systems

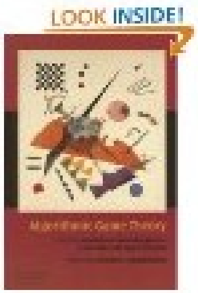
- Moving away from the black-box oracle model
- *justify* why a certain item is recommended
- maybe also *converse* to reach a recommendation


These recommendations are based on items you own and more.
view: All | New Releases | Coming Soon
1.
Auction Theory
by Vijay Krishna
Average Customer Review:
In Stock
Publication Date: March 1, 2002
Our Price: $52.46
Used & new from $52.46
Add to Wish
I Own It
Not Interested
Rate it
Recommended because you purchased Putting Auction Theory to Work and more (edit)

# Amazon explanations (contd.)

amazon.com

**Recommended for You**




**Algorithmic Game Theory**
by Noam Nisan (Editor), et al.
**Our Price: $33.75**
**Used & new** from $24.49

Add to Cart | Add to Wish List

x | ☆☆☆☆☆
I own it
Not interested

**Because you purchased...**

**Prediction, Learning, and Games** (Hardcover)
by Nicolo Cesa-Bianchi (Author), Gabor Lugosi (Author)

x | ☆☆☆☆☆
This was a gift
Don't use for recommendations

---

**Making Markets: How Firms Can Design and Profit from Online Auctions and Exchanges** (Hardcover)
by Ajit Kambil (Author), et al.

x | ☆☆☆☆☆
This was a gift
Don't use for recommendations

# Why have explanations? [Tintarev & Masthoff]

- Transparency
- "Scrutability": correct errors in learnt preference model
- Trust/Confidence in system
- Effectiveness & efficiency(speed)
- Satisfaction/enjoyment

# Example: explanations for transparency and confidence

- "Movie X was recommended to you because it is similar to movie Y, Z that you recently watched"
- "Movie X was recommended to you because you liked other comedies"
- "Other users who bought book X also bought book Y"

# Generating explanations

- Essentially, explain the steps of the CF algorithm, picking the most prominent "neighbors"
  - User-user
  - Item-item
- Harder to do for SVD and other abstract model-fitting recommender algorithms

# Conversational recommenders

Example transcript: (from [McSherry,"Explanation in Recommender Systems, AI Review 2005]):

- *Top case*: please enter your query
- *User*: Type = wandering, month = aug
- *Top Case*: the target case is "aug, tyrol, ..."
  other competing cases include "...."
- *Top case*: What is the preferred location?
- *User*: why?
- *Top case*: It will help eliminate ... alternatives
- *User*: alps..

# Conversational recommenders

- One view: CF using some navigational data as well as ratings

- More structured approach: incremental collaborative filtering
  - similarity metric changes as the query is refined

- e.g., incremental Nearest-Neighbor algorithm [McSherry, AI Review 2005]

# Scalable Implementations

- Learning objective:
  - see some techniques that are used for large-scale recommenders
  - Know where to start looking for more information

# Google News Personalization

[Das et al, WWW'07] describe algo. and arch.

- Specific challenges: News
  - relevant items are frequently changing
  - users long-lived, but often new users
  - Very fast response times needed
- Specific challenges: Google
  - scale! many items, many many users
  - need to parallelize complex computations

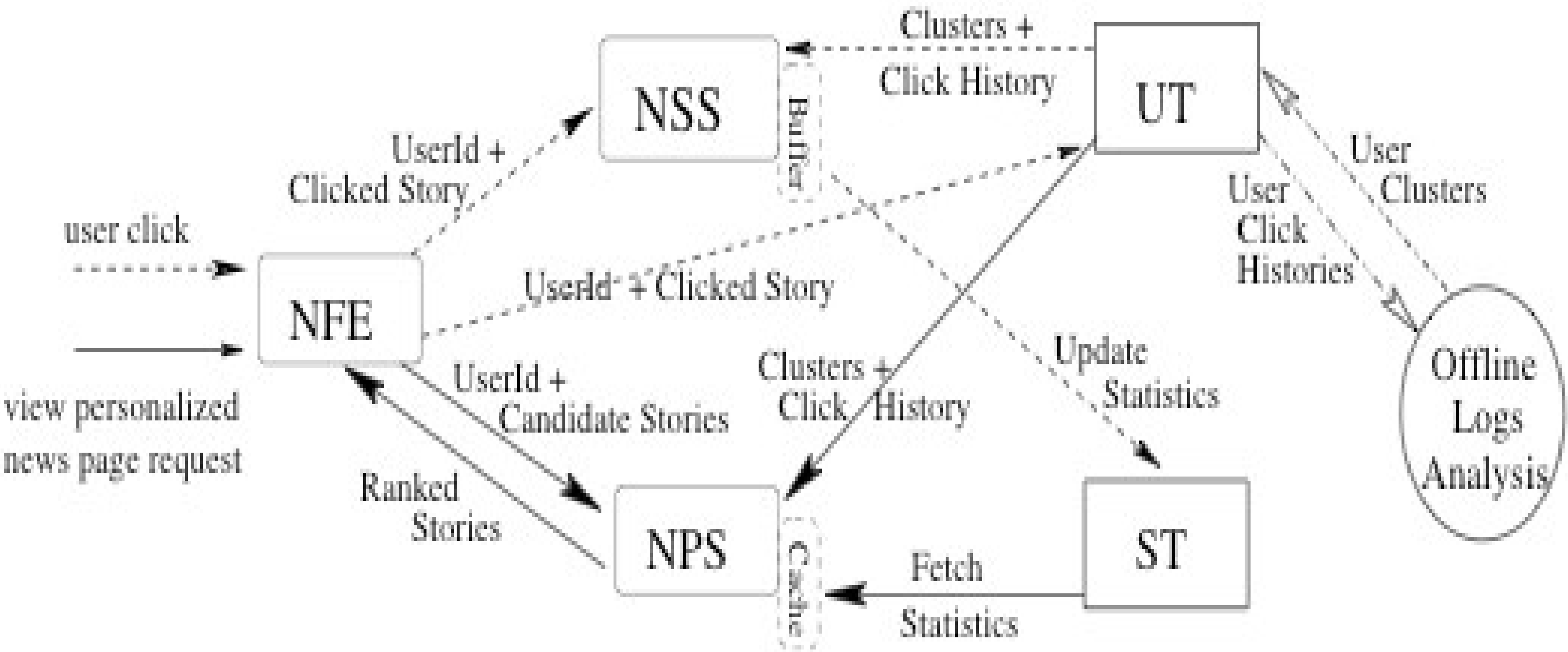
# Algorithms

- Input data: clicks
  - eg, “user J clicked on article X”
- Use a combination of three reco algos:
  - user-user (with a simple similarity measure)
  - SVD (“PLSI”)
  - Item-item (mainly for new users; simple covisitation similarity measure)

# Tricks/approximations for scalable computing

- User-user: calculate weighted avg. over only a *cluster* of users
  - J and K in same cluster if they have a high fraction of overlapped clicks
  - clustering is precomputed offline (using a fast MinHash algorithm)
- SVD : Precompute user-side weights; update only item-side weights in real time
  - gives an approximate SVD
- Tweak offline algorithms for parallel computing on Google's map-reduce infrastructure

# Architecture (from Das et al)



# Experiences with the Netflix prize challenge

- Difference: static dataset

- My “architecture” (such as it was):
  - A clustered user-user
    - randomly chosen clusters (not optimal)
    - cluster size to fit user-user calc in 1GB memory
  - Preprocess, create indices (perl scripts)
  - Calculate similarities (in C) ***{memory bottleneck}***
  - Generate predictions (perl)
  - Evaluate accuracy on test set (perl)

# Manipulation..

# Why manipulate a recommender?

- Examples?

# Why manipulate a recommender?

- Examples?
  - Digg/Slashdot: get an article read
  - PageRank: get your site high on results page
  - Books: Author wants his book recommended
  - Spam
- How?

# Example: User-User Algorithm

| user \ item | *A* | *B* | *C* | | . . . | | | | *X* |
|---|---|---|---|---|---|---|---|---|---|
| Joe | 7 | 4 | | 4 | 2 | | 5 | | **?** |
| Sue | 7 | 5 | 6 | 5 | | | 6 | | **8** |
| John | 2 | | 3 | | 7 | | | | **2** |
| | | | | | | | | | |

- i's informativeness score = correlation coefficient of i's past ratings with Joe's past ratings
- Prediction for item X = average of ratings of X, weighted by the rater's scores

# Cloning Attack: Strategic copying

- Attacker may copy past ratings to look informative, gain influence.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Joe | 7 | 4 | | 4 | 2 | | 5 | | **?** |
| FreeMeds | 7 | 4 | | 4 | 2 | | 5 | | **10** |

- Even if ratings are not directly visible, attacker may be able to infer something about ratings from her own recommendations, publicly available statistics

- Worse if many accounts can be created (sybil attack)

# One approach: profile analysis

- This problem of “shilling attacks” has been noted earlier [Lam and Riedl] [O’Mahoney et al]
- Many papers on empirical measurements and statistical detection of attack profiles
- Problem: attackers may get better at disguising their profiles.

# Results we *cannot* achieve

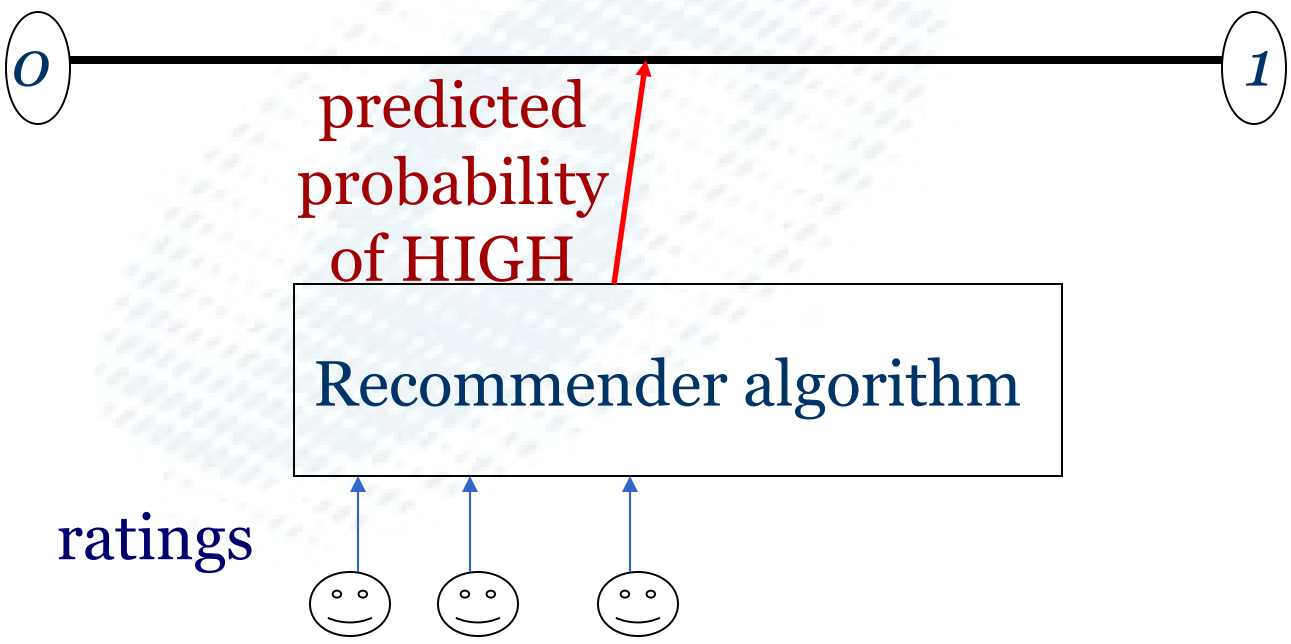
- Prevent any person J from manipulating the prediction on a single item X.
  - Cannot distinguish *deliberate manipulation* from *different tastes* on item X
- "Fairness", ie., two raters with identical information get exactly the same influence, regardless of rating order.
  - Cannot distinguish second rater with identical information from an informationless clone.

# The influence limiter: Key Ideas

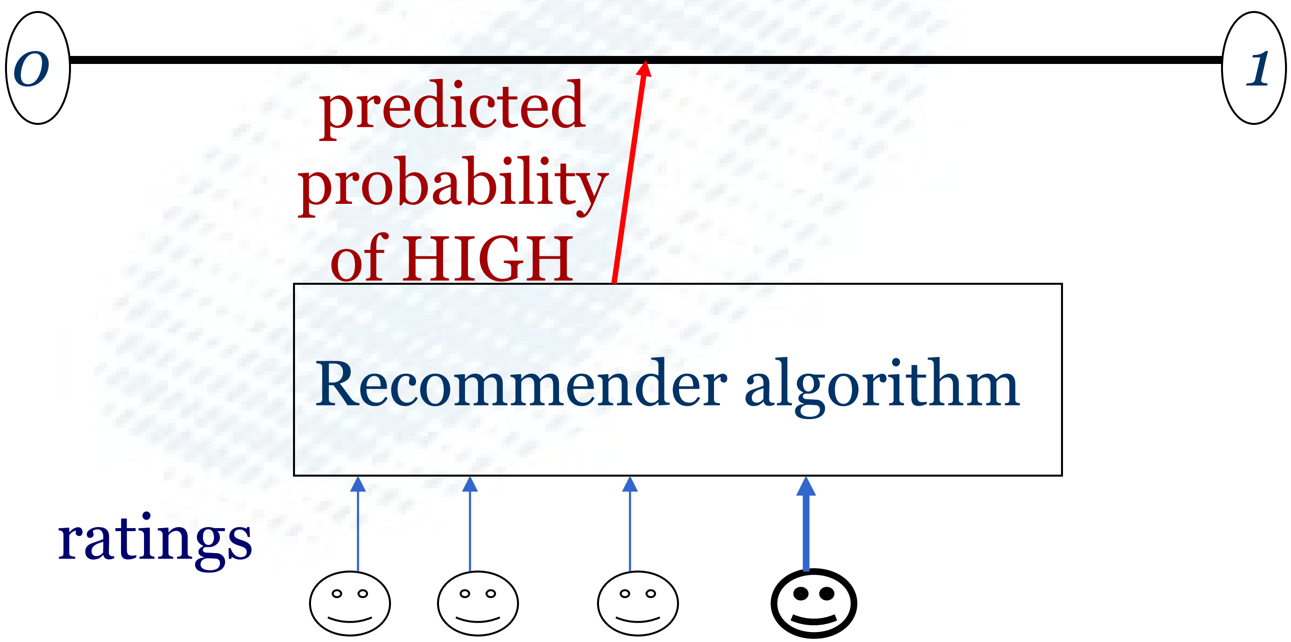
[Resnick and Sami, Proceedings of RecSys ‘07 conference]

- Limit *influence* until rater demonstrates *informativeness*
- *Informative* only if you’re the first to provide the information

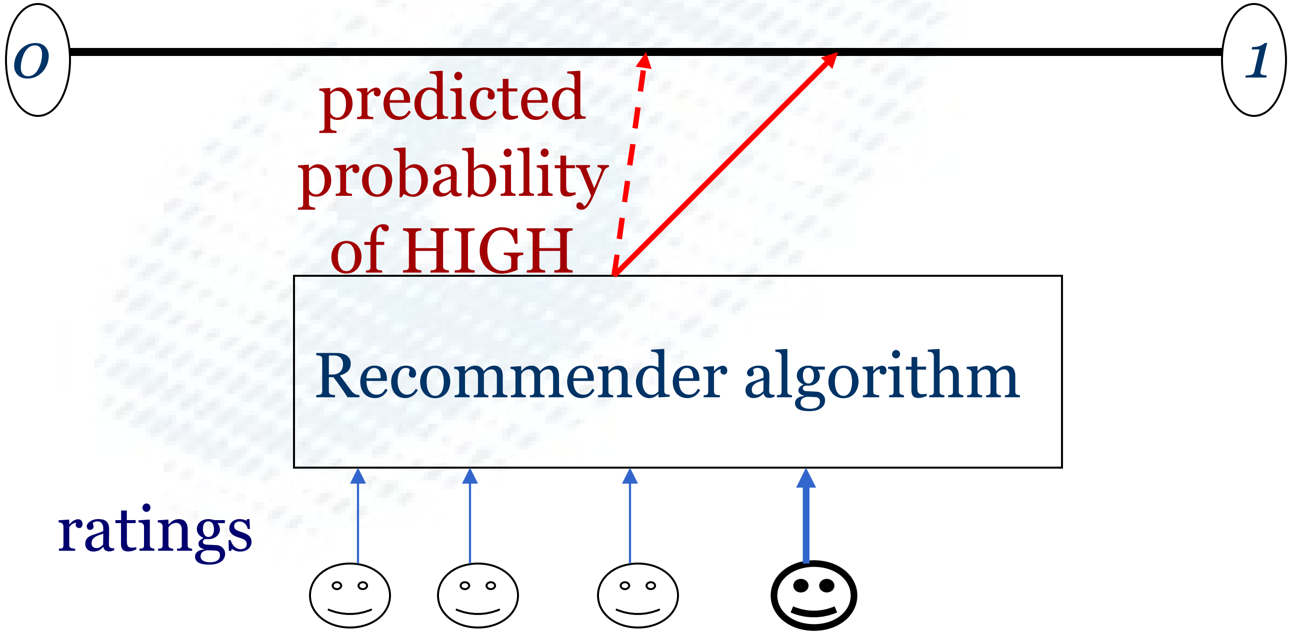
# Predictions on an Item: A Dynamic View



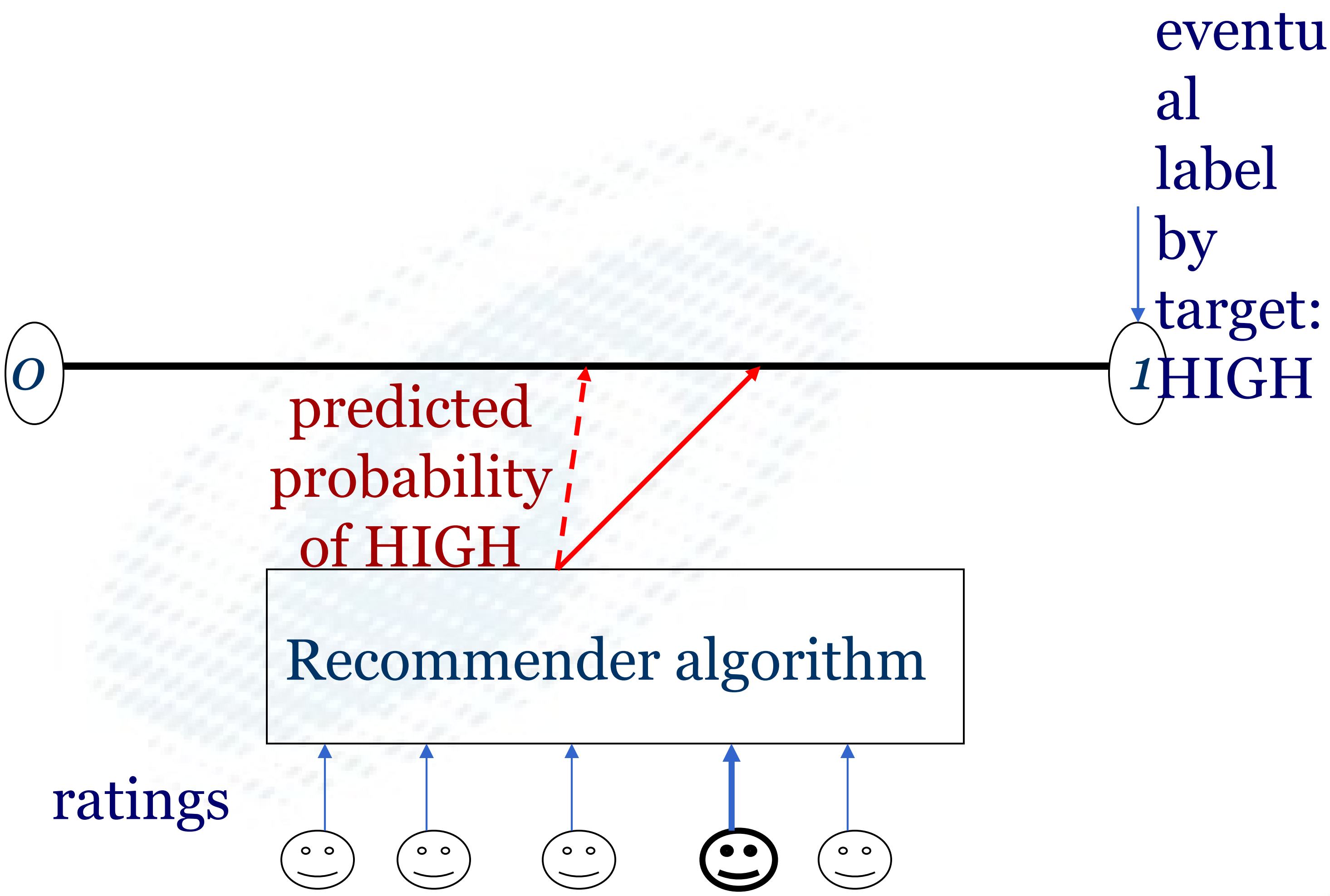
# Predictions on an Item: A Dynamic View



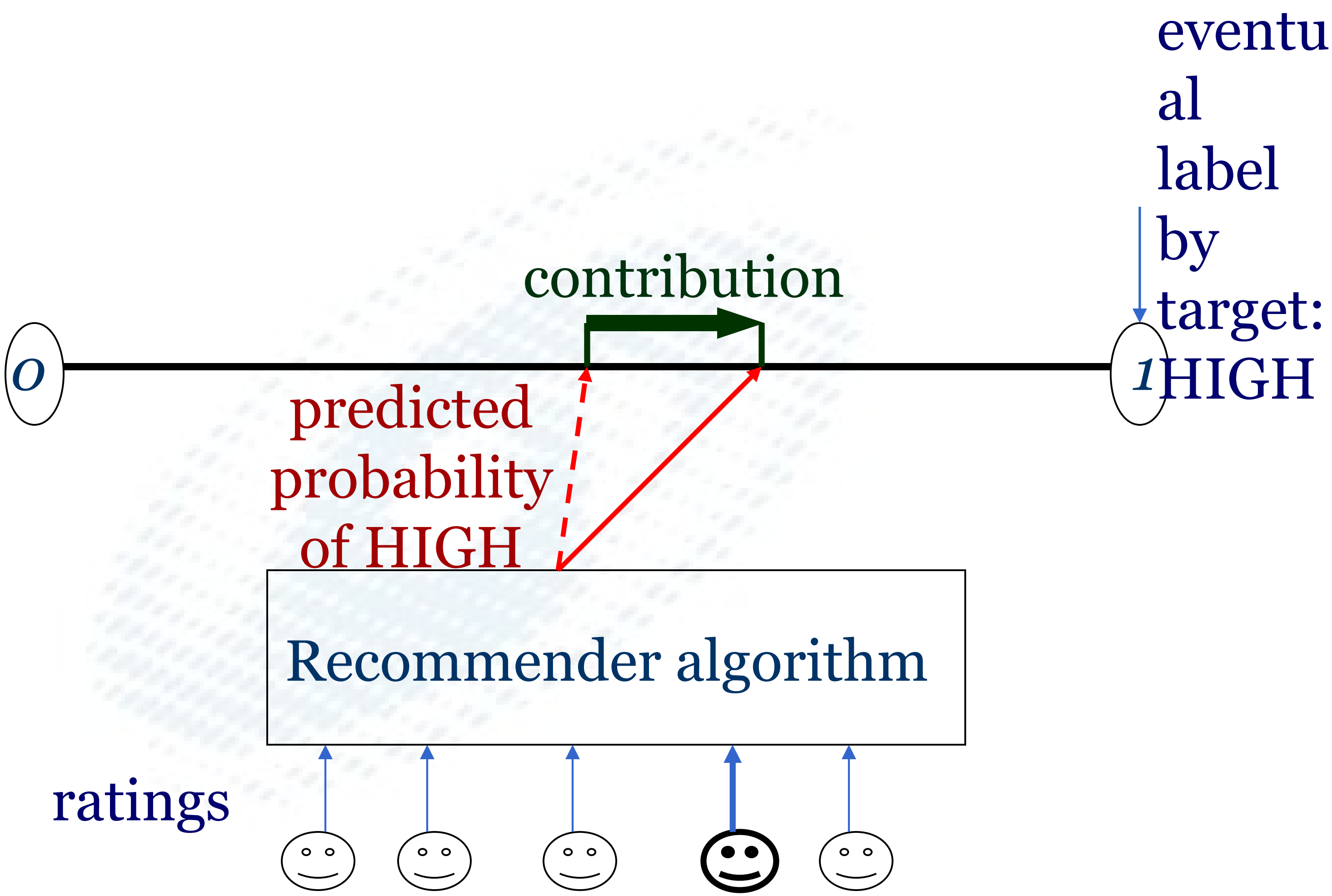
# Predictions on an Item: A Dynamic View



# Predictions on an Item: A Dynamic View

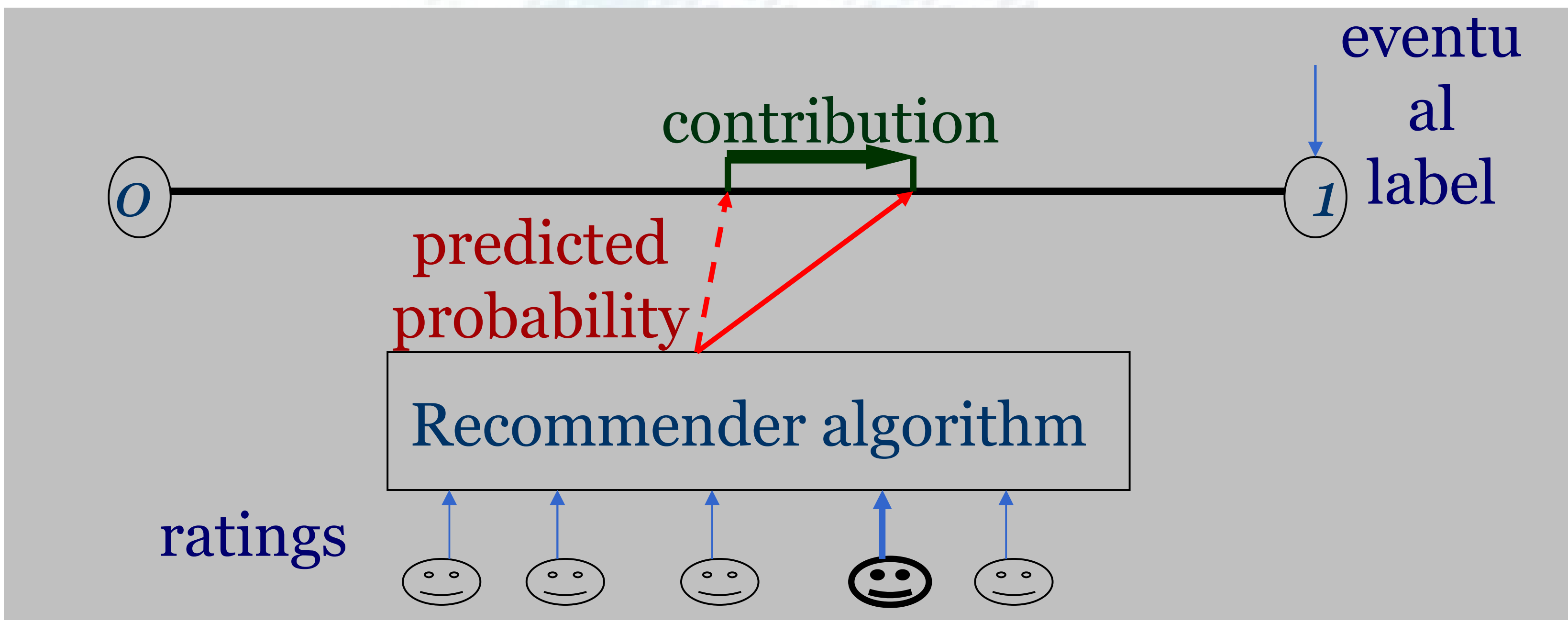


# Predictions on an Item: A Dynamic View



# Our approach

- Information-theoretic measure of contribution and damage
- Limit *influence* a rater can have had based on past *contribution*
- This limits *net damage* an attacker can cause



# Our Model

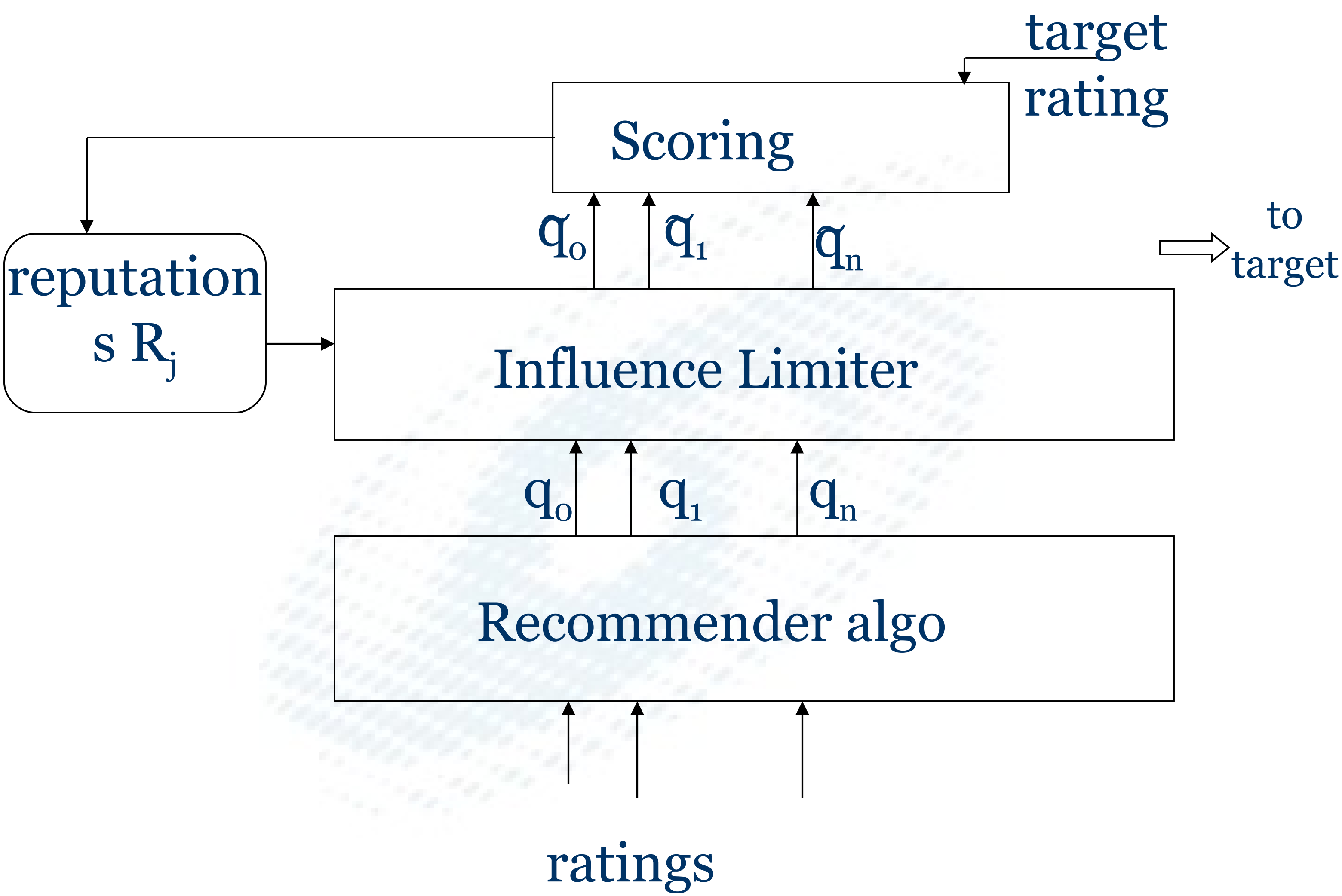
- Binary rating system (HIGH/LOW)
- Recommendations for a single target person
- Any recommender algorithm
- Powerful attackers:
  - Can create up to *n* sybil identities
  - Can “clone” existing rating profiles
- No assumptions on non-attackers:
  - Attacker’s sybils may form majority
  - Do not *depend* on honest raters countering attacks

# Overview of Results

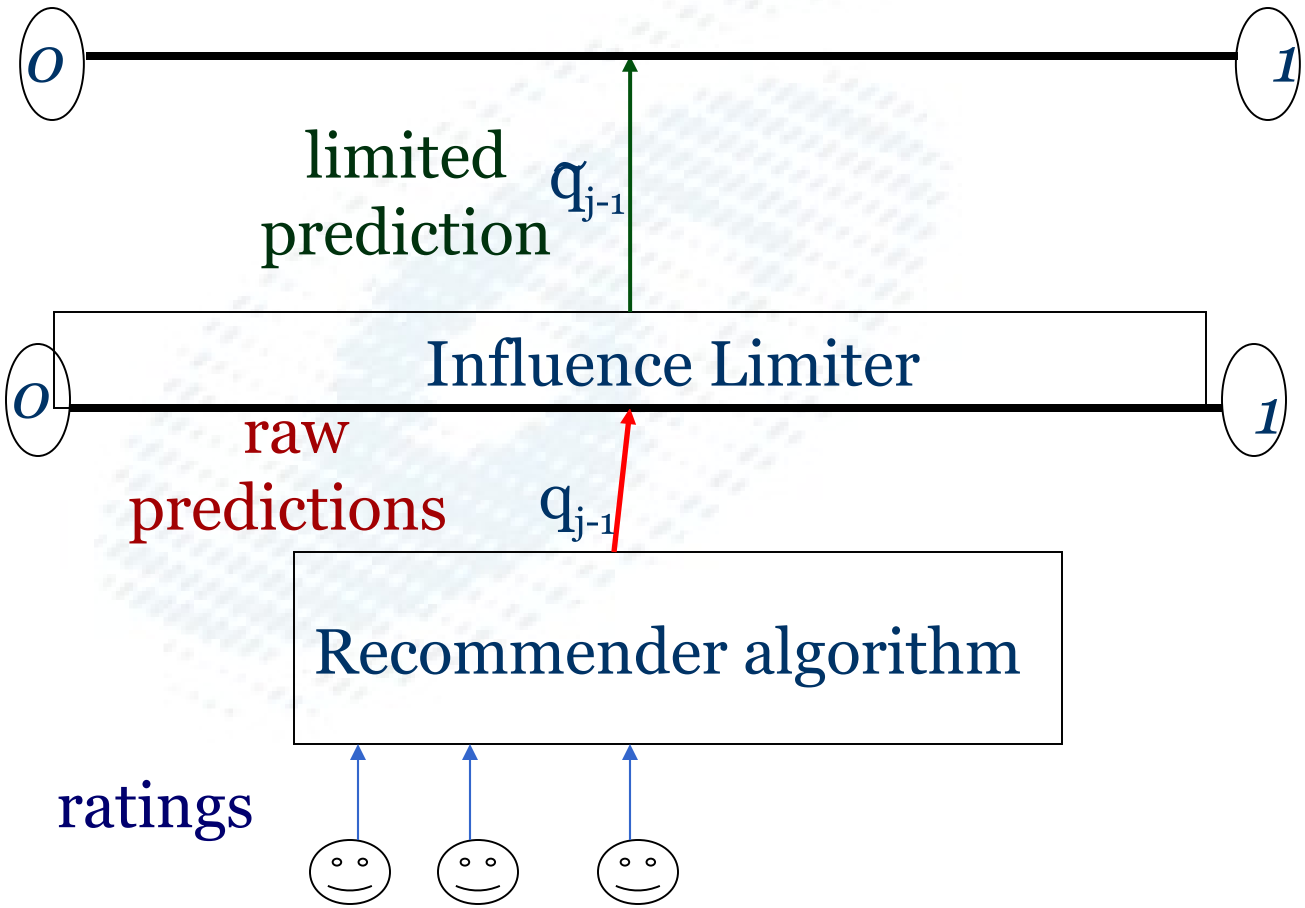
"Influence-limiter" algorithm can be overlaid on any recommender algorithm to satisfy (with caveats):

- **Limited damage**: An attacker with up to *n* sybils can never cause net total damage greater than *O(1)* units of prediction error
- **Bounded information loss**: In expectation, O(log *n)* units of information discarded from each genuine rater in total.

# Influence Limiter: Architecture

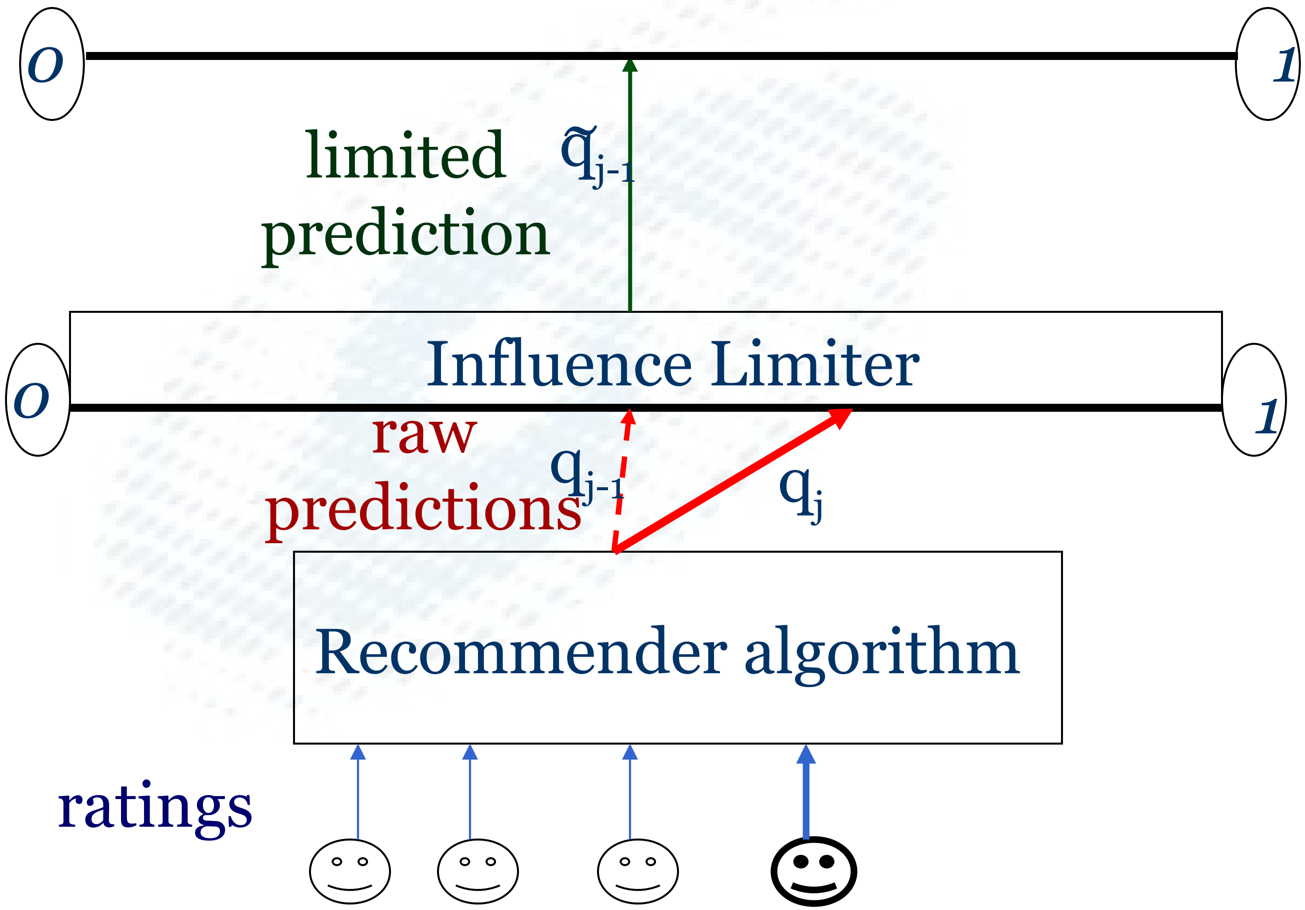


# Influence Limiter Algorithm: Illustration



# Influence Limiter Algorithm: Illustration

A rater with R=0.25 puts in a rating



# Influence Limiter Algorithm: Illustration

A rater with R=0.25 puts in a rating