RECOMMENDER SYSTEMS

WHAT RECOMMENDER SYSTEMS DO WE KNOW?

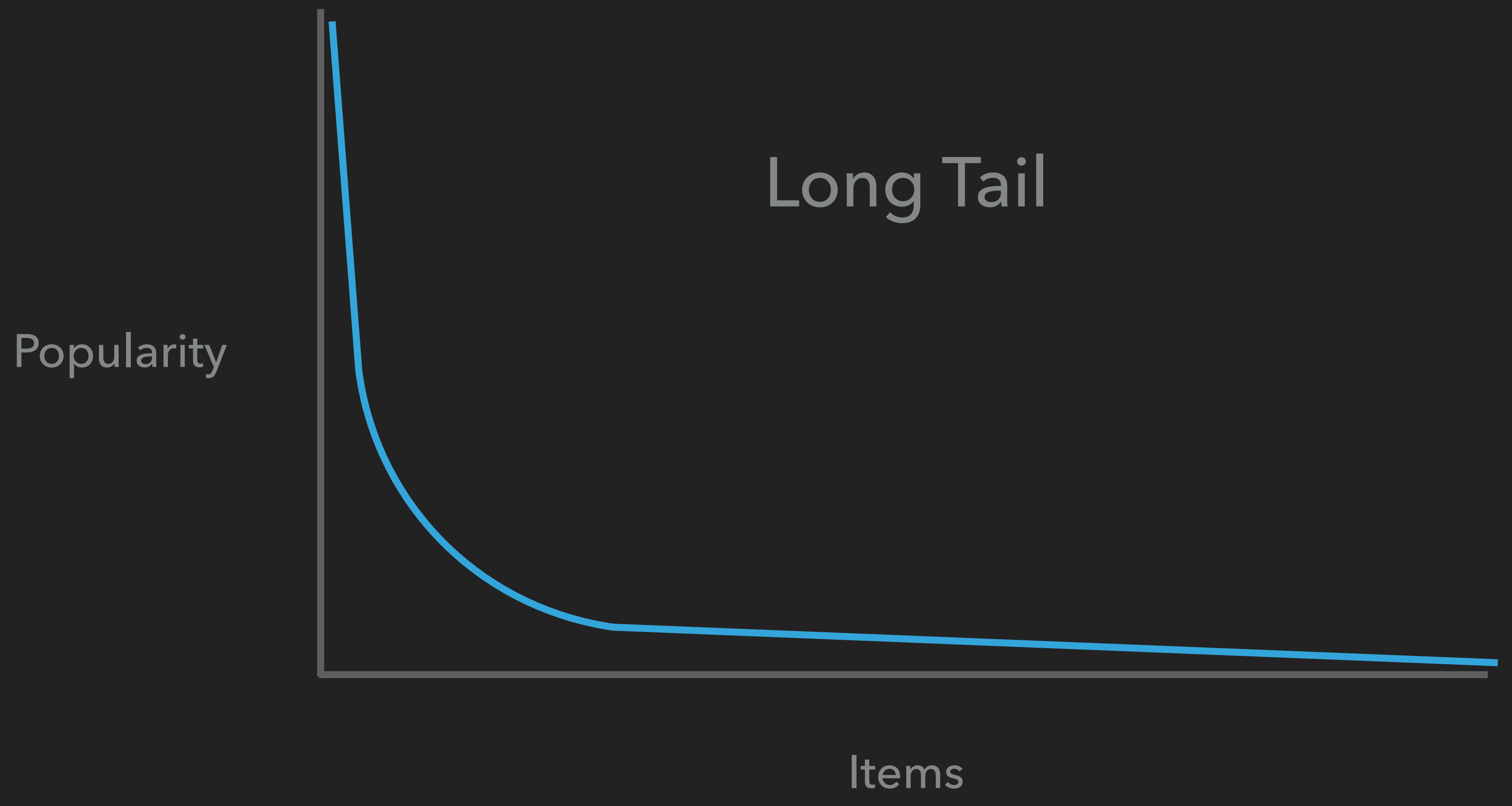
- Netflix, Spotify, Reddit, Amazon…

WHICH TYPES ARE THERE?

- Generic vs Personalised
- Memory Based vs Model Based
- Content Based Filtering vs Collaborative Filtering
- Hybrid Models

WHY DO WE NEED THEM?

- Too much choice



NON PERSONALISED RECOMMENDATIONS

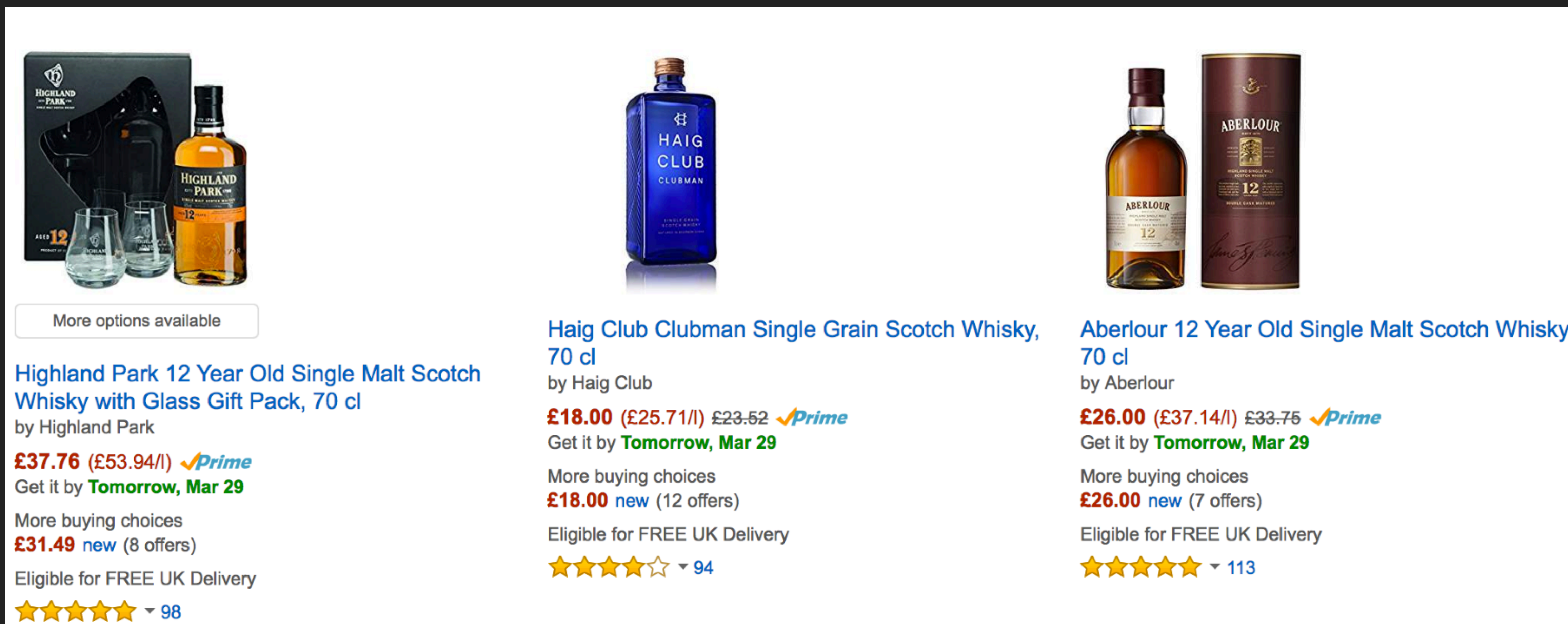
NON PERSONALISED RECOMMENDER SYSTEMS

- Everyone sees the same recommendations;
- Demographic filtering

EXAMPLES:

- Amazon product ratings
- eBay
- Reddit
- Hacker News

AMAZON RATINGS



REDDIT'S 'BEST' RANKING ALGORITHM

- How to rank comments?
  - Net upvotes

$$Score = Upvotes - Downvotes$$

  - Upvote proportion

$$Score = \frac{Upvotes}{\text{Total votes}}$$

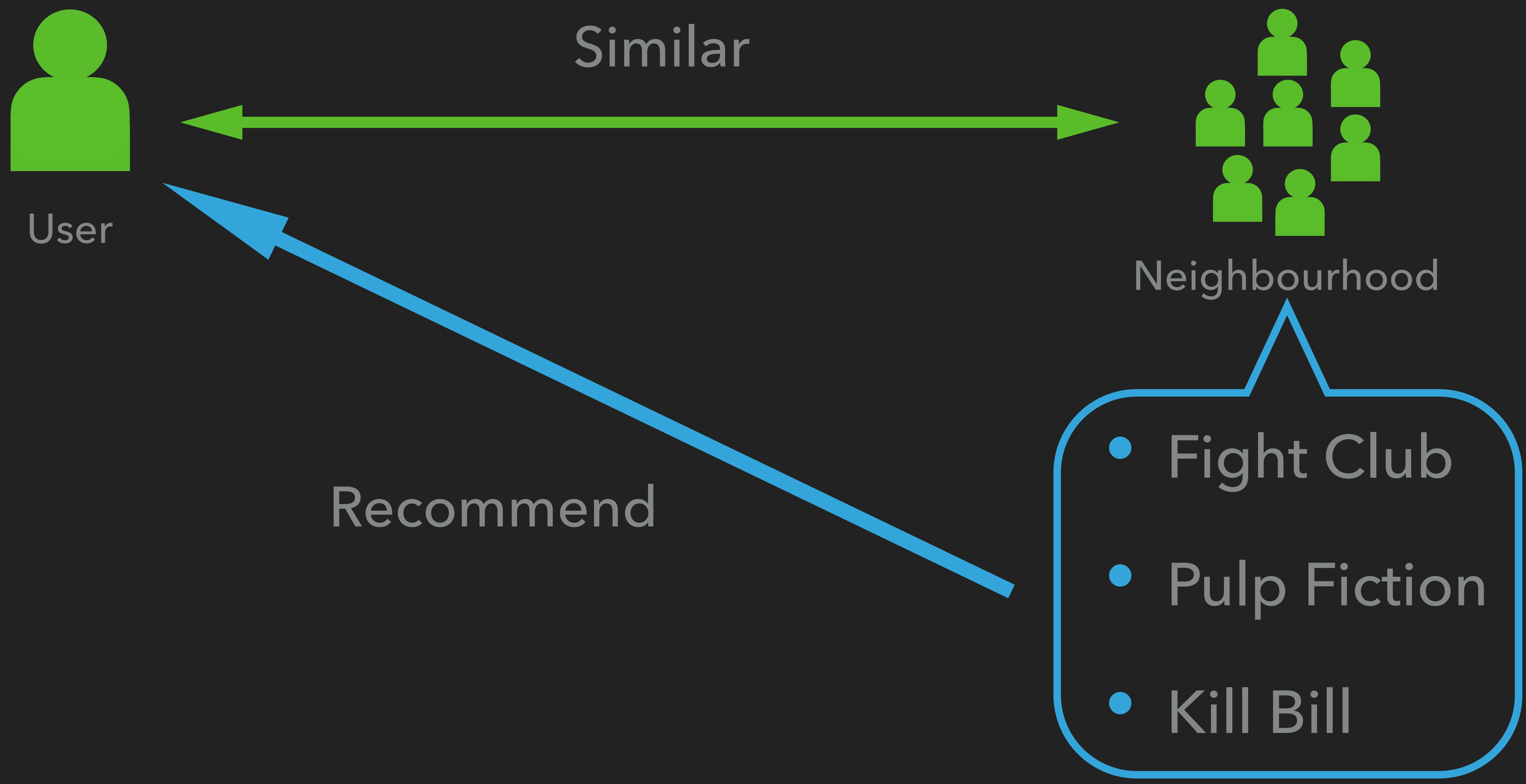
REDDIT'S 'BEST' RANKING ALGORITHM

- Solution - Lower bound of Wilson score
  - What would be the score of a comment if everyone had voted on it?
  - Based on the current votes predict the real average rating.

More details on: http://www.evanmiller.org/how-not-to-sort-by-average-rating.html

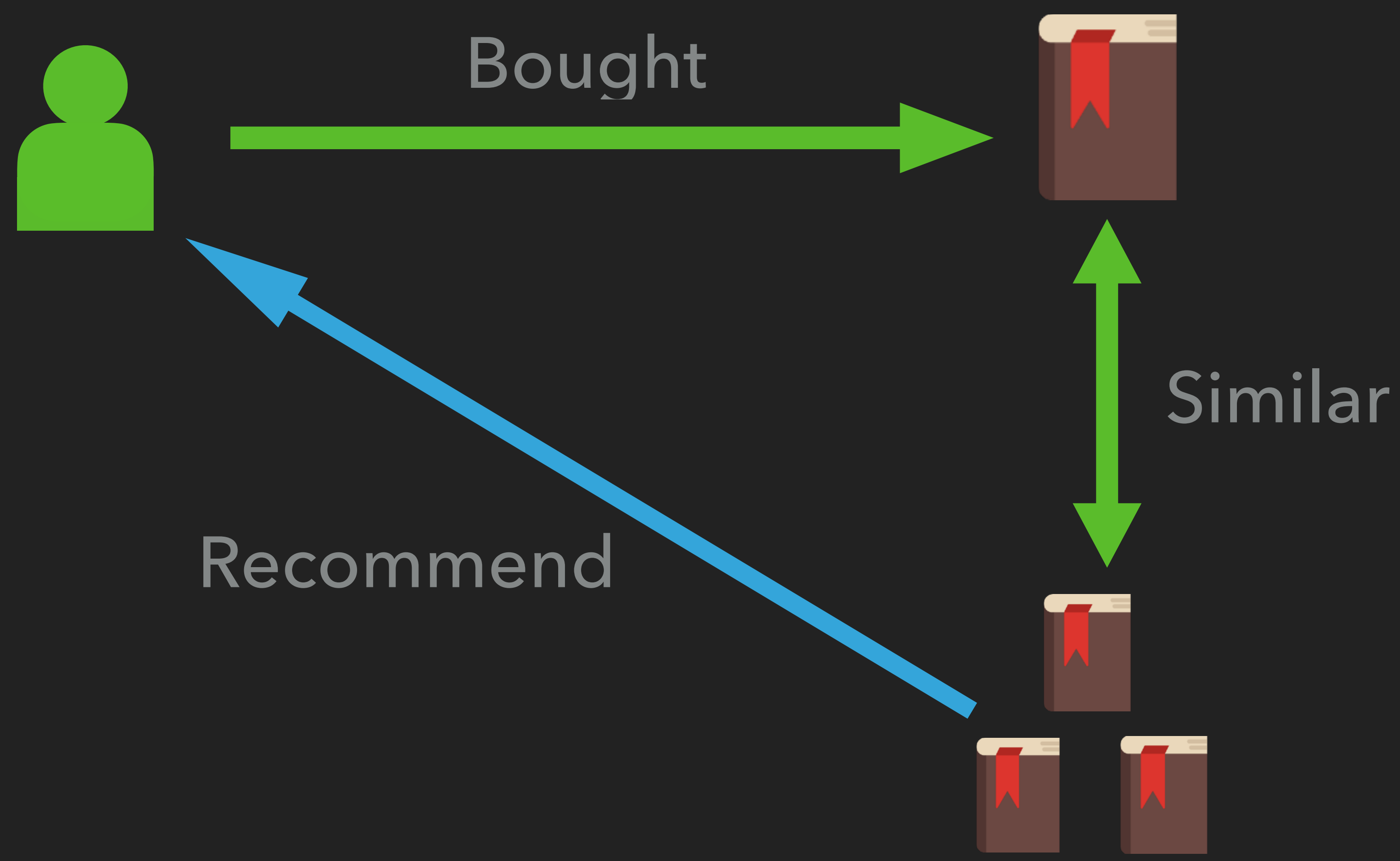
PERSONALISED RECOMMENDATIONS

NEIGHBOURHOOD BASED MODELS



NEIGHBOURHOOD BASED: ITEM TO ITEM VS USER TO USER

What if we look for similar items?



NEIGHBOURHOOD BASED MODELS

USER TO USER

$$\hat{r_{u,i}} = \frac{\sum_v^K s_{u,v} \times r_{v,i}}{\sum_v^K s_{u,v} + \lambda}$$

NEIGHBOURHOOD BASED MODELS

ITEM TO ITEM

$$\hat{r_{u,i}} = \frac{\sum_j^K s_{i,j} * r_{u,j}}{\sum_j^K s_{i,j} + \lambda}$$

NEIGHBOURHOOD BASED MODELS

USER TO USER

$$\hat{r_{u,i}} = \frac{\sum_v^K s_{u,v} \times r_{v,i}}{\sum_v^K s_{u,v} + \lambda}$$

ITEM TO ITEM

$$\hat{r_{u,i}} = \frac{\sum_j^K s_{i,j} * r_{u,j}}{\sum_j^K s_{i,j} + \lambda}$$

CONTENT BASED FILTERING

Users / Items are similar when they have similar characteristics

- 22 years old
- Student
- Scottish

- 21 years old
- Student
- Scottish

- 12 years old
- Single malt
- Scottish

- 15 years old
- Single malt
- Scottish

FINDING CONTENT BASED SIMILARITIES

Taking textual descriptions of users or items:

- Tokenisation: Bag of Words
- TF-IDF:

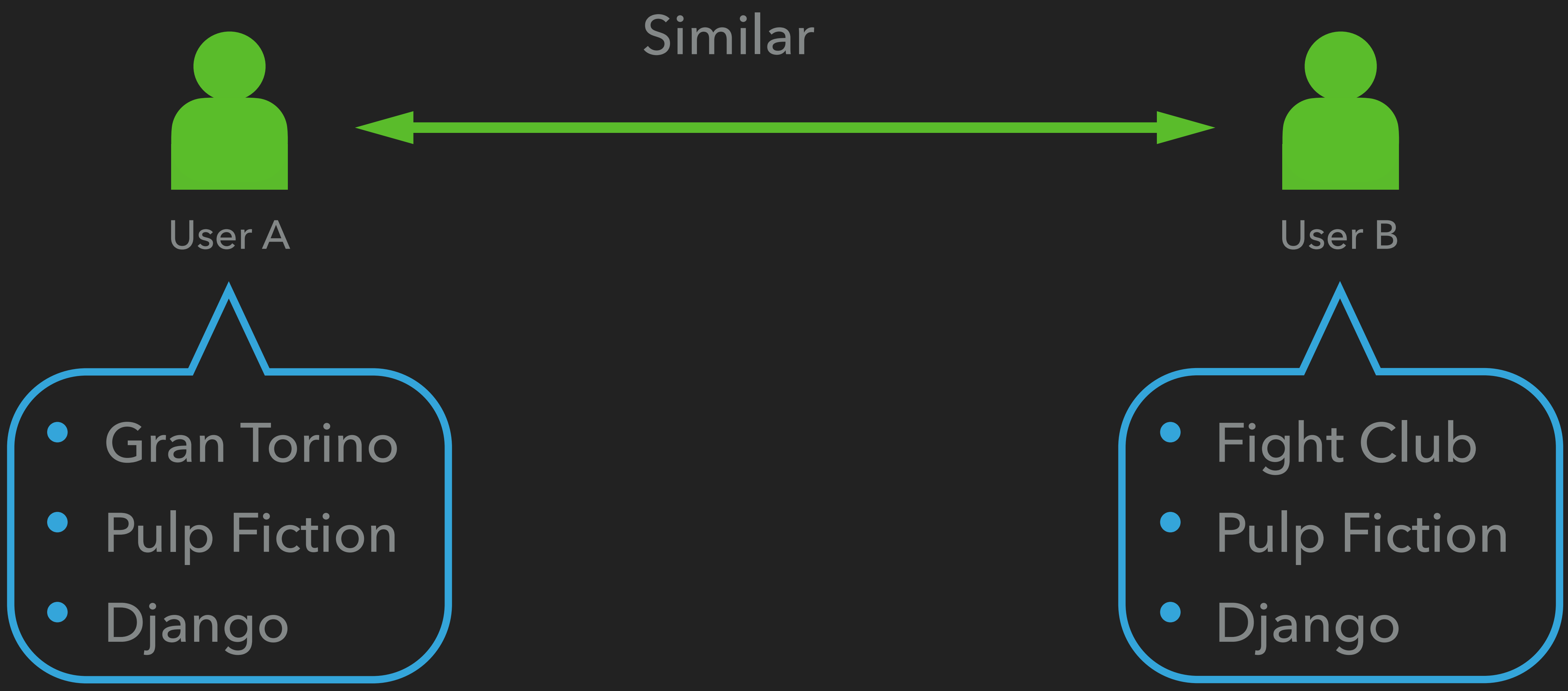
$$\text{TF-IDF} = tf \times log(\frac{N}{n_t})$$

- Cosine Similarity:

$$cos_{sim}(a, b) = \frac{a.b}{||a||.||b||}$$

COLLABORATIVE FILTERING

- Users are similar when they have similar tastes
- Items are similar when they are consumed by the same users



FINDING COLLABORATIVE FILTERING SIMILARITIES

Take the items rated by each user C_a or the ratings of each user V_a:

- Jaccard Similarity:

$$\text{Jacccard\_Sim}(a, b) = \frac{|C_a \cap C_b|}{|C_a \cup C_b|}$$

- Cosine Similarity

$$cos_{sim}(a, b) = \frac{a.b}{||a||.||b||}$$

NETFLIX PRIZE

- Prize: 1 million $
- 48,000 teams from 182 different
- Goal: Improve by 10% the RMSE
- Data: 100 million ratings from 500.000 users to 18.000 movies
- From Oct 2006 to Sept 2009
- Won by a difference of 20 min

MODEL BASED: SIMON FUNK'S SVD

The rating results of the linear combination of user factors and item factors

$$r_{u,i} \approx p_u^T \cdot q_i$$

$$\begin{array}{cc} & \text{Items} \\ \text{Users} & \begin{bmatrix} ? & 3 & ? & 5 & ? & 3 & ? \\ ? & 4 & ? & ? & 2 & ? & 5 \\ 3 & ? & ? & 5 & ? & 2 & ? \\ ? & ? & ? & ? & ? & ? & ? \\ ? & 5 & 4 & ? & ? & ? & 4 \\ 5 & ? & ? & ? & 5 & 1 & ? \\ ? & ? & ? & 4 & ? & ? & ? \\ 3 & ? & ? & ? & ? & ? & 2 \\ ? & 2 & 3 & ? & ? & ? & 4 \\ 4 & ? & 2 & ? & 5 & ? & 3 \\ ? & 2 & ? & ? & 5 & ? & ? \\ ? & 4 & ? & 2 & ? & 3 & ? \end{bmatrix} \end{array} = \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} \\ u_{2,1} & u_{2,2} & u_{2,3} \\ u_{3,1} & u_{3,2} & u_{3,3} \\ u_{4,1} & u_{4,2} & u_{4,3} \\ u_{5,1} & u_{5,2} & u_{5,3} \\ u_{6,1} & u_{6,2} & u_{6,3} \\ u_{7,1} & u_{7,2} & u_{7,3} \\ u_{8,1} & u_{8,2} & u_{8,3} \\ u_{9,1} & u_{9,2} & u_{9,3} \\ u_{10,1} & u_{10,2} & u_{10,3} \\ u_{11,1} & u_{11,2} & u_{11,3} \\ u_{12,1} & u_{12,2} & u_{12,3} \end{bmatrix} \times \begin{bmatrix} i_{1,1} & i_{1,2} & i_{1,3} & i_{1,4} & i_{1,5} & i_{1,6} & i_{1,7} \\ i_{2,1} & i_{2,2} & i_{2,3} & i_{2,4} & i_{2,5} & i_{2,6} & i_{2,7} \\ i_{3,1} & i_{3,2} & i_{3,3} & i_{3,4} & i_{3,5} & i_{3,6} & i_{3,7} \end{bmatrix}$$

ESTIMATING THE RATING FROM HIDDEN FACTORS

$$r_{u,i} \approx \mu + b_i + b_u + p_u^T \cdot q_i$$

Going through each known ratings with SGD:

$$b_u(k+1) = b_u(k) + \gamma * (e_{u,i}(k) - \lambda_1 * b_u(k))$$

$$b_i(k+1) = b_i(k) + \gamma * (e_{u,i}(k) - \lambda_1 * b_i(k))$$

$$q_i(k+1) = q_i(k) + \gamma * (e_{u,i}(k) * q_i(k) - \lambda_1 * q_i(k))$$

$$p_u(k+1) = p_u(k) + \gamma * (e_{u,i}(k) * p_u(k) - \lambda_1 * p_u(k))$$

PROBLEMS WITH RECOMMENDER SYSTEMS

- Cold-Start
- Lack of Novelty & Diversity (The Bubble)
- Privacy Concerns
- Scalability

COLD START

- How do these systems make recommendations for new users?

- Clustering
- Ask them for data
- Start with random recommendations

HYBRID RECOMMENDATION SYSTEMS

- Switching models
- Weighted Recommendations
- Graph based

QUESTIONS?

REFERENCES

- Reddit's Algorithm: Reddit Blog and Evan Miller
- Recommender Systems:
  - Mining Massive Datasets - Chapter 9
  - Recommender Systems Handbook
- SVD: Simon Funk original blog post
- SVD with implicit ratings: SVD ++ paper
- "Matrix Factorization Techniques For Recommender Systems"
- Netflix Prize: "All Together Now"

Icons by madebyoliver and nikita-golubev from www.flaticon.com