Introduction to Recommendation Systems

Jin Yao CHIN
Nanyang Technological University
Singapore

11/03/2021

Outline

- **WHERE** can we find recommendation systems?
- **WHY** do we need recommendation systems?
- **WHAT** is recommendation?
- **HOW** do we make recommendations?

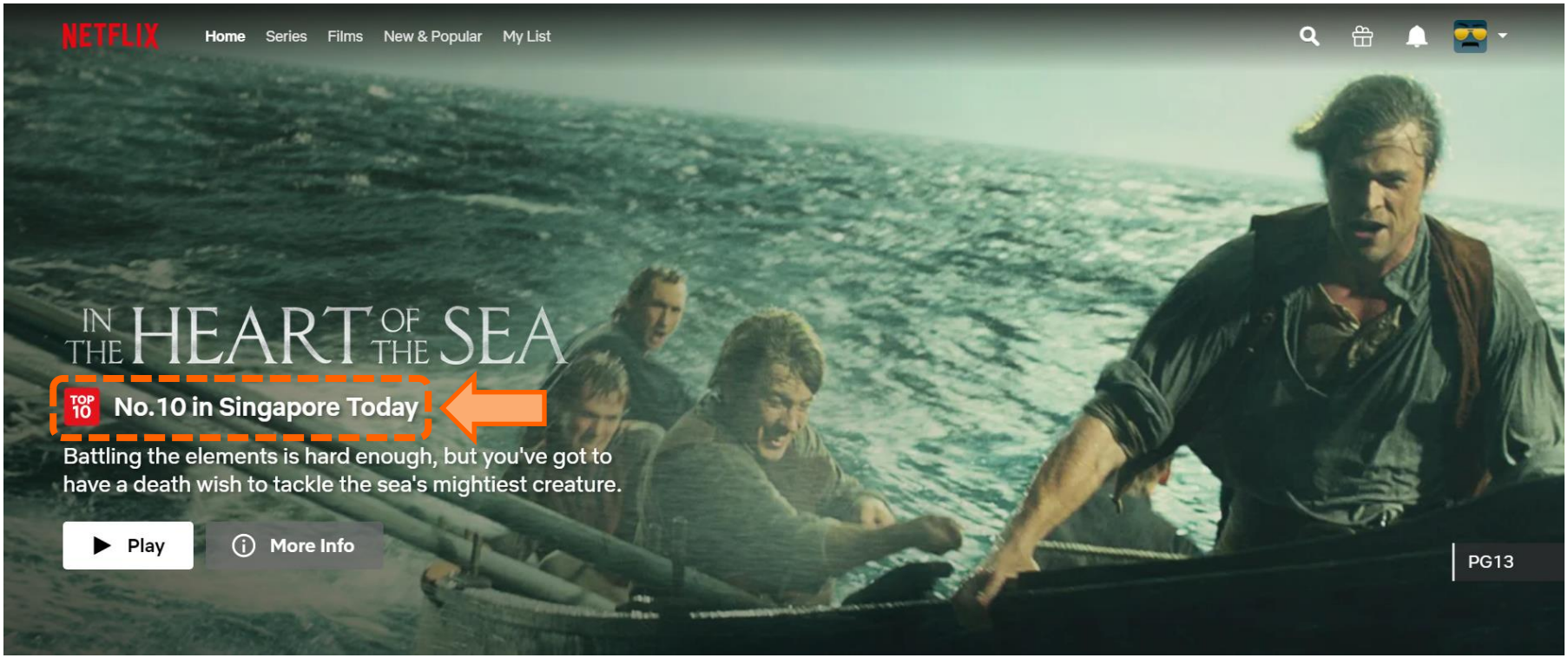
- Different **types** of recommendation systems
- **Challenges** encountered by recommendation systems

1.

WHERE can we find recommendation systems?

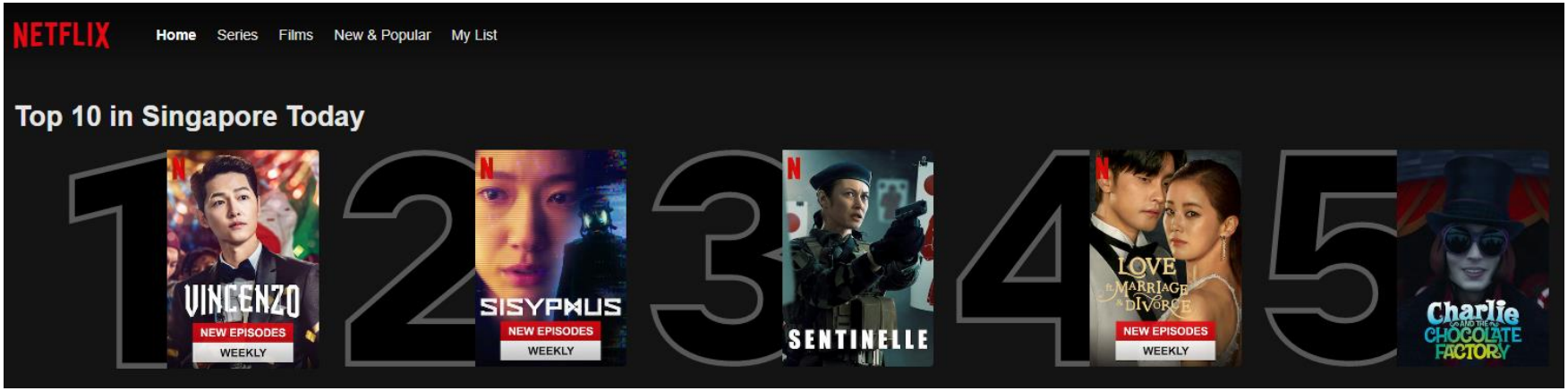
Recommendation Systems – WHERE?

Recommendation Systems – WHERE?



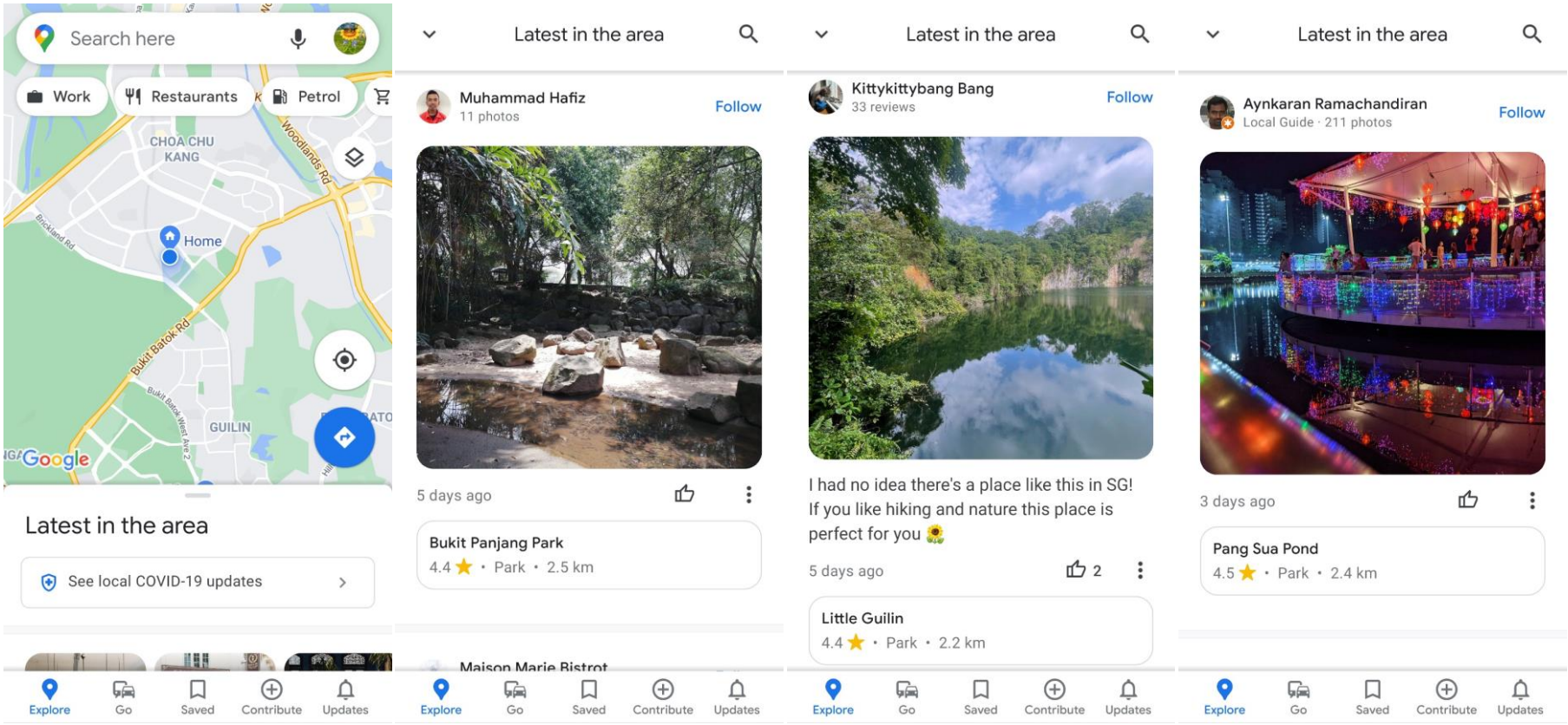
- E.g., **Netflix**
  - Netflix might try to recommend a show which is very popular based on your current ***location*** (E.g., *No. 10 in Singapore Today*)

Recommendation Systems – WHERE?



- E.g., **Netflix**
  - Netflix might try to recommend the Top 10 shows based on your current ***location***
  - However, note that these are ***not*** personalized recommendations
    - Based on other users located in *Singapore*
    - Everyone in *Singapore* gets the same recommendations

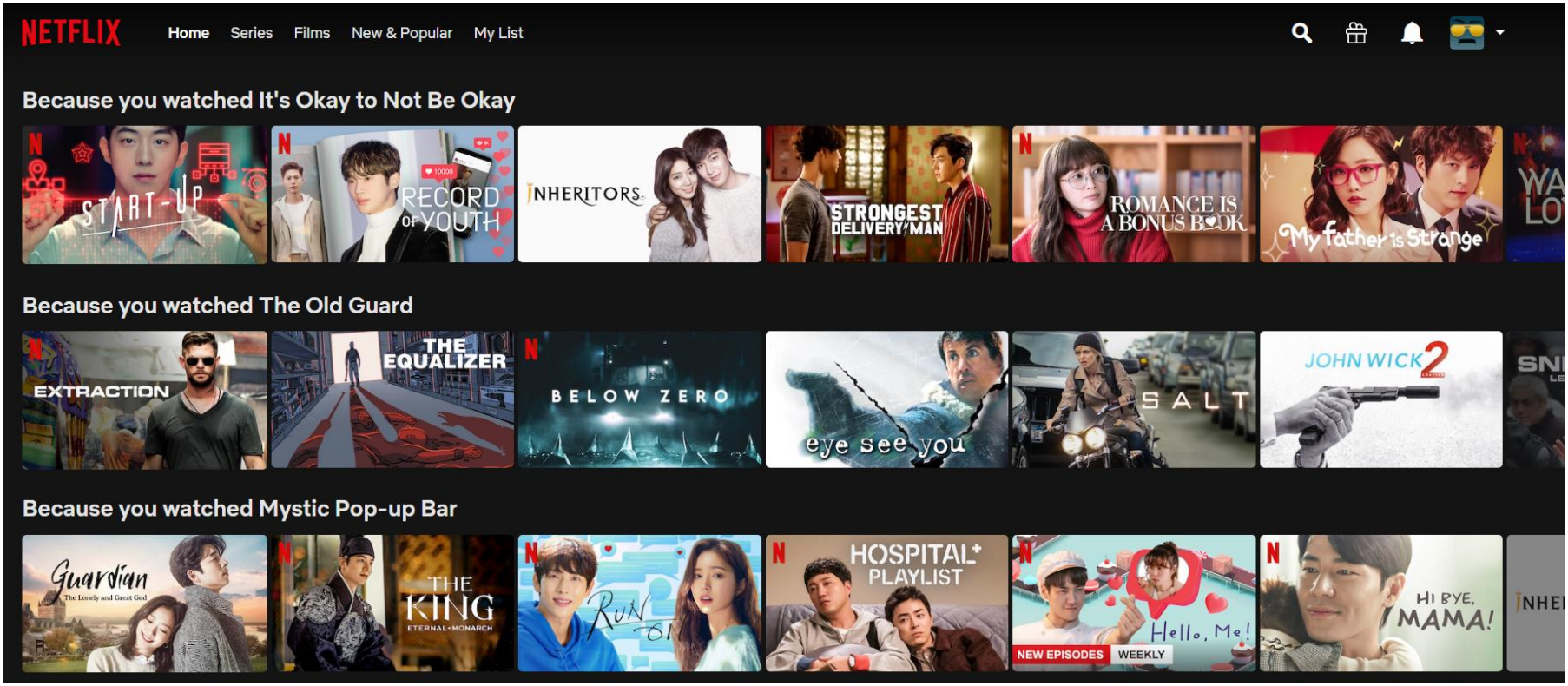
Recommendation Systems – WHERE?



▷ E.g., **Google Maps**

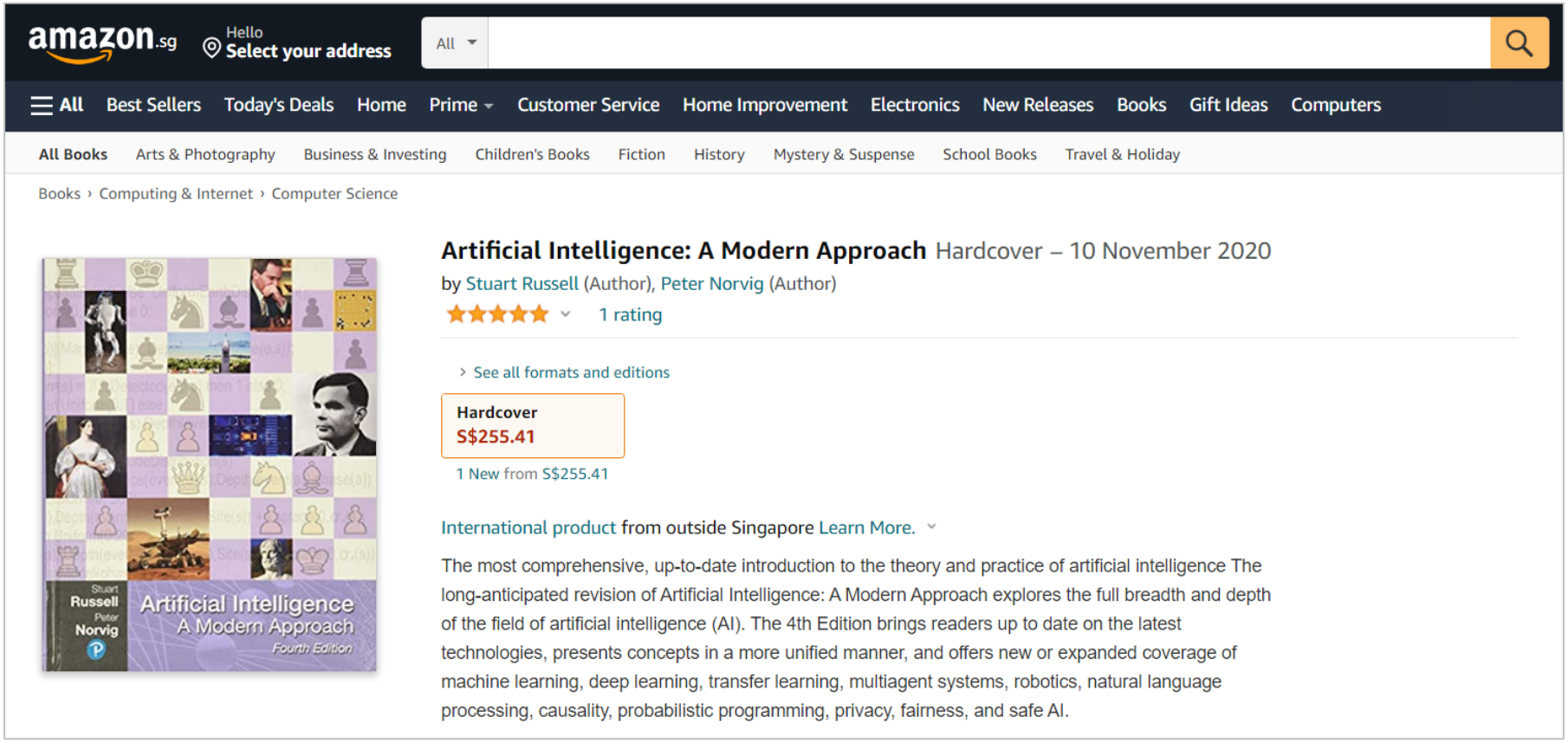
- Given your current ***location***, Google Maps might recommend popular locations *near you*

Recommendation Systems – WHERE?



- E.g., **Netflix**
  - Shows are often recommended based on your viewing ***history*** (i.e., shows that you have watched previously)

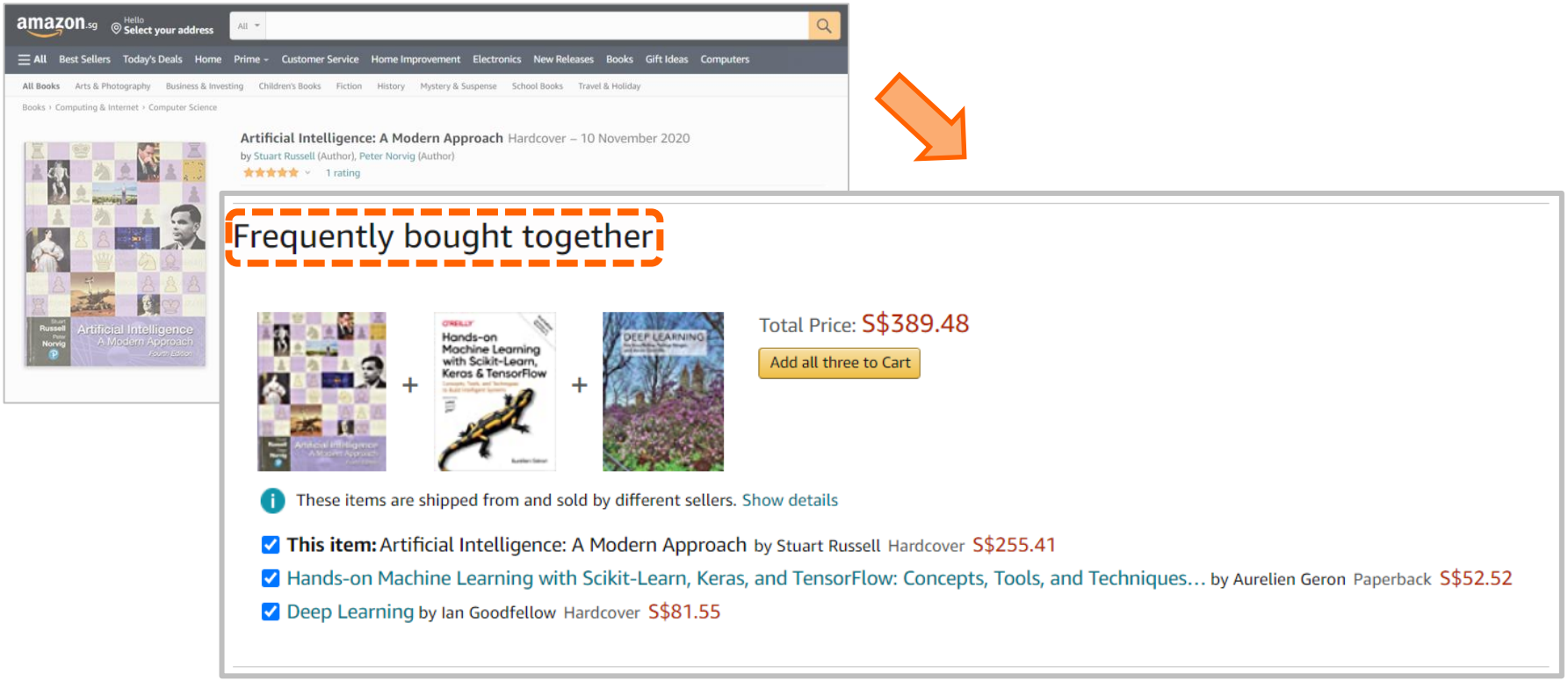
Recommendation Systems – WHERE?



▷ E.g., **Amazon**

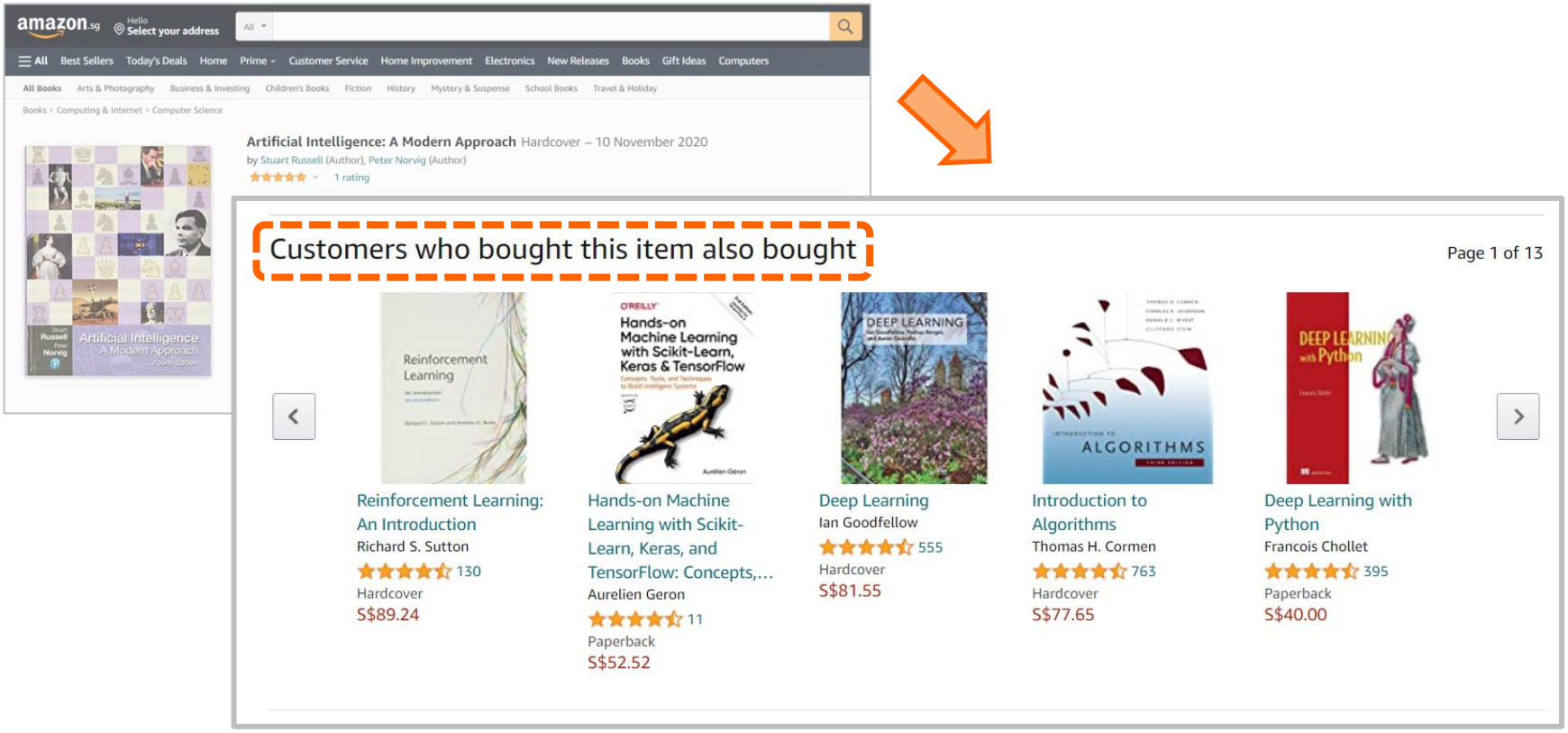
- Let's assume that we are viewing some book in Amazon (e.g., *Artificial Intelligence: A Modern Approach*)

Recommendation Systems – WHERE?



- E.g., **Amazon**
  - On the same page, Amazon would try to recommend other items (i.e., books) that are *frequently bought together*

Recommendation Systems – WHERE?



▷ E.g., **Amazon**

- On the same page, Amazon would try to recommend other items (i.e., books) that were *bought by 'similar' customers*

Recommendation Systems – WHERE?

- Recommendation systems can be found everywhere!
  - Movies, Books, Music, Points of Interests, …
- (Literally) an indispensable part of our daily lives

and many others…

2.

WHY do we need recommendation systems?

Recommendation Systems – WHY?

1. ***Large catalogue***
   - w/ up to millions of items
2. ***Beneficial*** *to both the* ***end users*** *&* *the* ***businesses***

Large Catalogue ☹

- *"Amazon has an inventory of about **12 million items** across all its categories and services"* [1]

- Let's say… I would like to get a new chair (*but I am not entirely sure what kind of a chair would be good*)
  - There could be thousands of chairs being sold
  - Different *types* of chairs: Office, Dining, …
  - Different *colours*, *sizes*, *prices*, …
  - It could be very challenging and tedious to find something ideal!
  - Recommender systems can help to narrow down the options effectively ☺

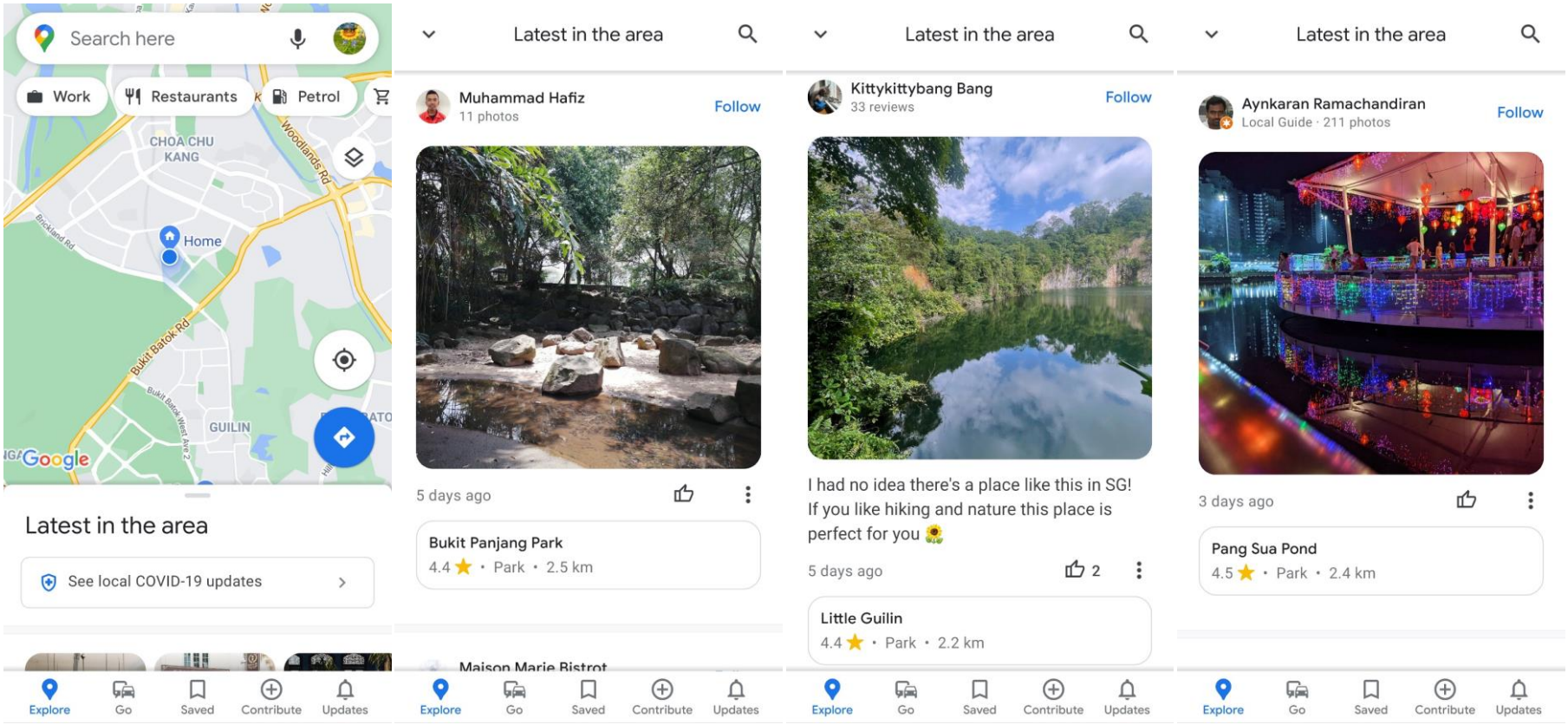
[1] https://www.repricerexpress.com/amazon-statistics/ (Retrieved on 08/03/2021)

Large (and expanding) Catalogue ☹

- Platforms such as YouTube, TikTok, etc. are based on User Generated Content (**UGC**)
- In contrast to platforms such as Netflix, the catalogue is ***expanding rapidly***!

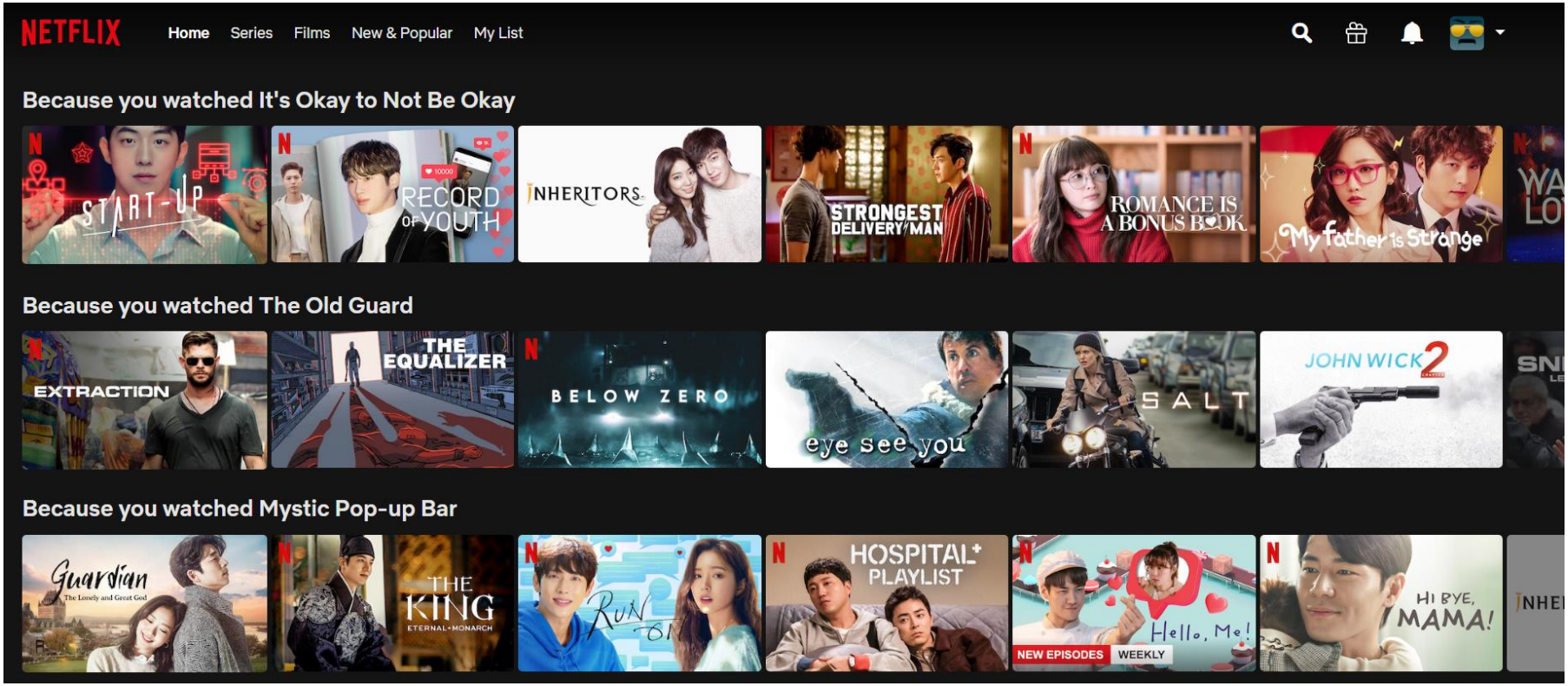
[1] https://sg.oberlo.com/blog/youtube-statistics (Retrieved on 08/03/2021)
[2] https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/ (Retrieved on 08/03/2021)

Beneficial to End Users ☺



- E.g., **Google Maps**
  - The *'explore' feature* assists users in finding ***nearby*** *points of interests*
  - Users might not be aware of these locations before this!

Beneficial to End Users ☺



- E.g., **Netflix**
  - By leveraging your viewing **history**, Netflix could recommend other shows which you are most likely to watch…
  - Similarly, you might not be aware of such shows prior to this!

Beneficial to End Users ☺

- A good recommender system would be akin to having a friend *who knows you even better than yourself*
  - Assists users in finding ***items of interest***
  - Introduces new items to users (*Novelty*)
  - Presents unexpected items to users (*Serendipity*)

Beneficial to Businesses ☺

- Helps item providers (e.g., sellers, content creators, etc.) deliver their products to its intended audience
- Improves customer satisfaction
  - Less time spent on browsing / searching
  - Caters to the preferences of different users
- The underlying recommendation system could be the ***key difference*** between…
  - 2 platforms having the same catalogue
  - 2 platforms with the same type of products

3.

WHAT is recommendation?

Recommendation Systems – WHAT?

- **Where?**
  - Everywhere!
  - Part and parcel of our daily lives
- **Why?**
  - Large (and perhaps expanding) catalogues
  - Good for both end users & businesses
- **What? → *Formalizing the recommendation problem***
  - What are the inputs? Outputs?
  - Are we making good (or bad) recommendations?

A (Virtuous) Cycle

▷ Recommendation systems are **driven by user interactions**

Recommendation System

User

- *More* interactions → *Better* awareness of user preferences
- *Better* awareness of user preferences → *Better* recommendations
- *Better* recommendations → *More* interactions

Recommendations & Feedback

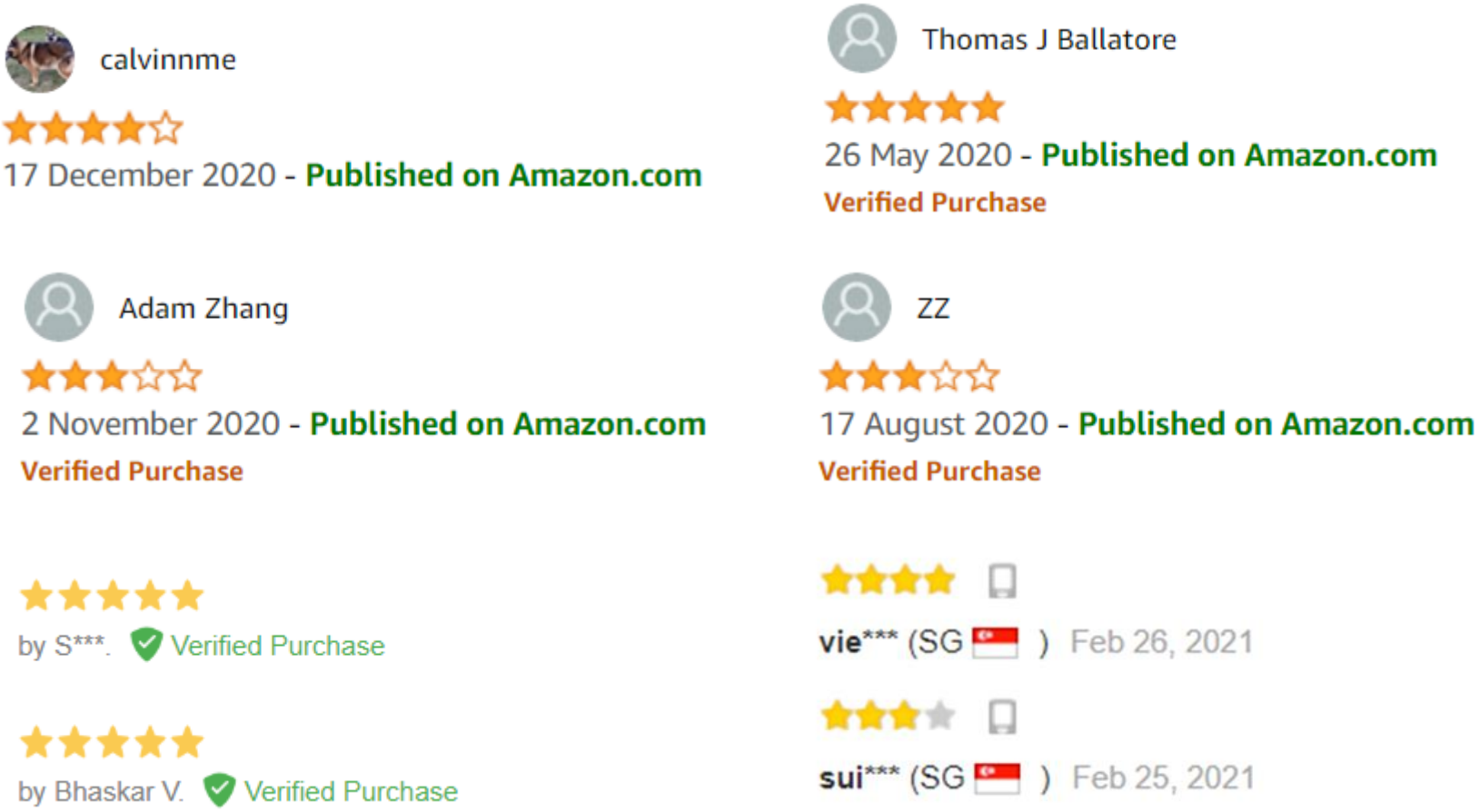
- Recommendation systems
  - **Outputs**: Items of interest (e.g., *Movies*, *Books*, etc.)
  - **Inputs**: User feedback (*Explicit* & *Implicit*)

Recommendation System

User

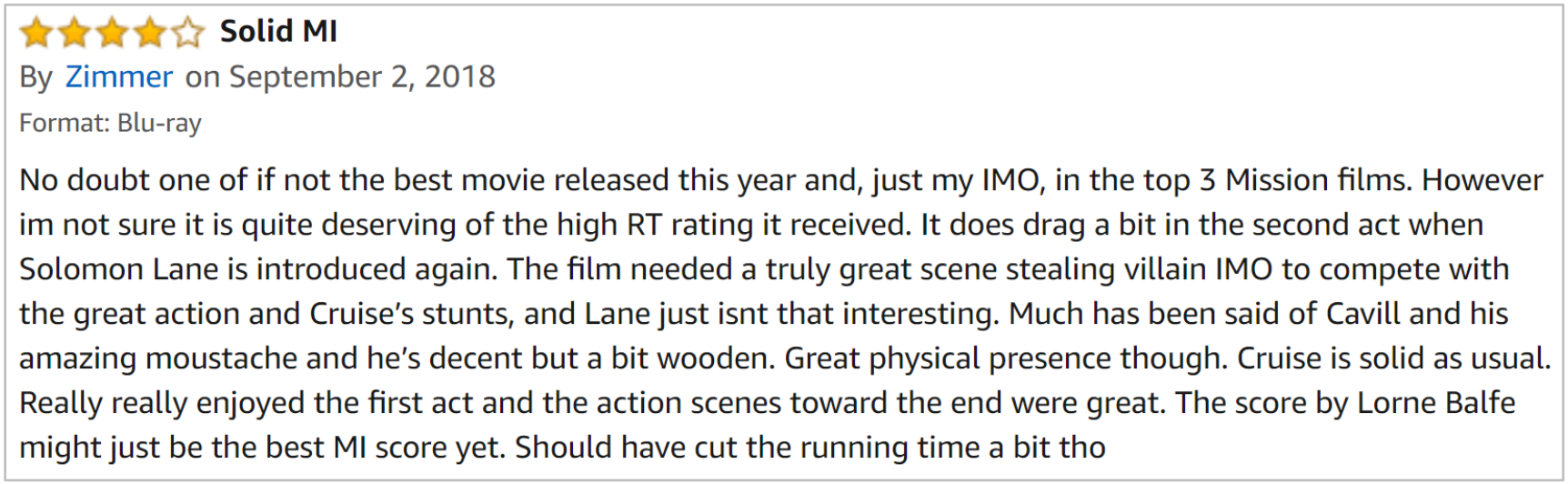
Explicit Feedback

- For example, users can provide a ***rating*** for an item



Explicit Feedback

- For example, users can provide a rating for an item
  - Furthermore, some platforms allow users to provide a ***review***



Explicit Feedback – Rating Prediction

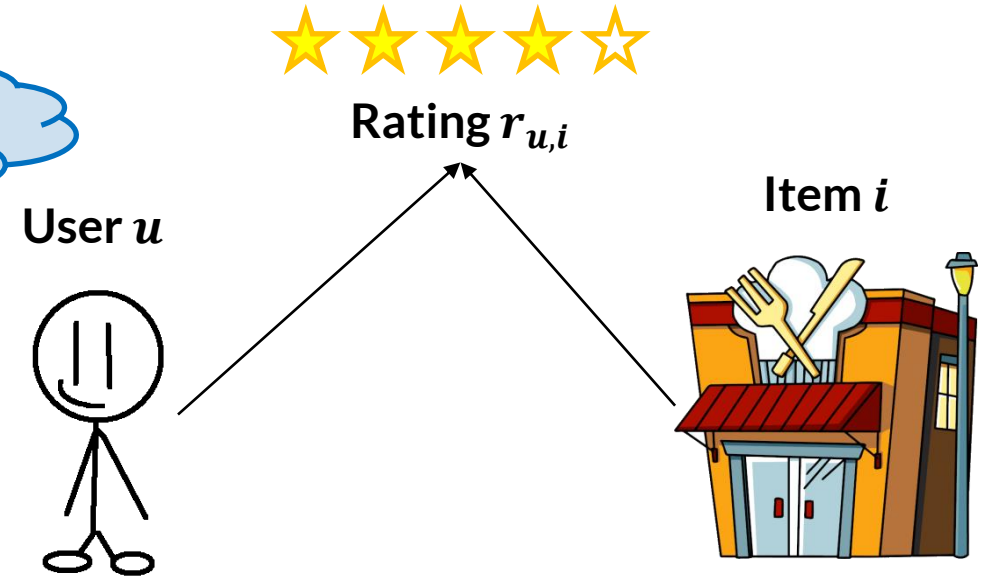
For each user u, we would like to estimate the rating $\hat{r}_{u,i}$ for any **new item** i

Prediction

▷ (Input) **Matrix:** $\boldsymbol{R} \in \mathbb{R}^{N \times M}$

- N users, M items
- $r_{u,i} = \{1, \dots, 5\}$ if user u has interacted with item i, 0 otherwise

Observation



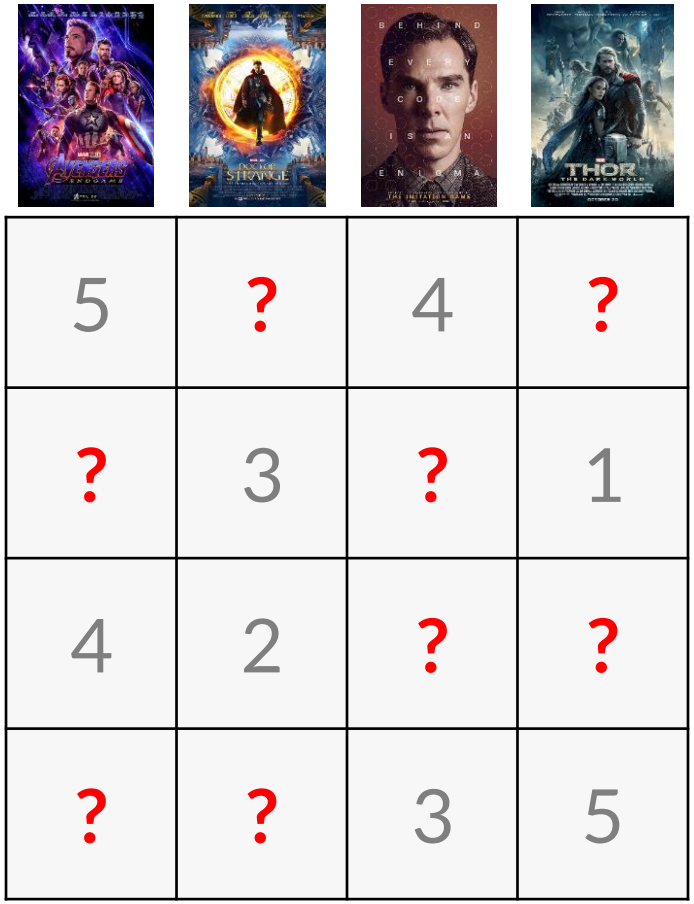
▷ Recommend **new items** that the user would rate highly

Explicit Feedback – Rating Prediction

- Given the *observed entries*, recover the *missing entries*
  - $\{r_{ui}\}$: The set of *observed entries*

Explicit Feedback – Rating Prediction

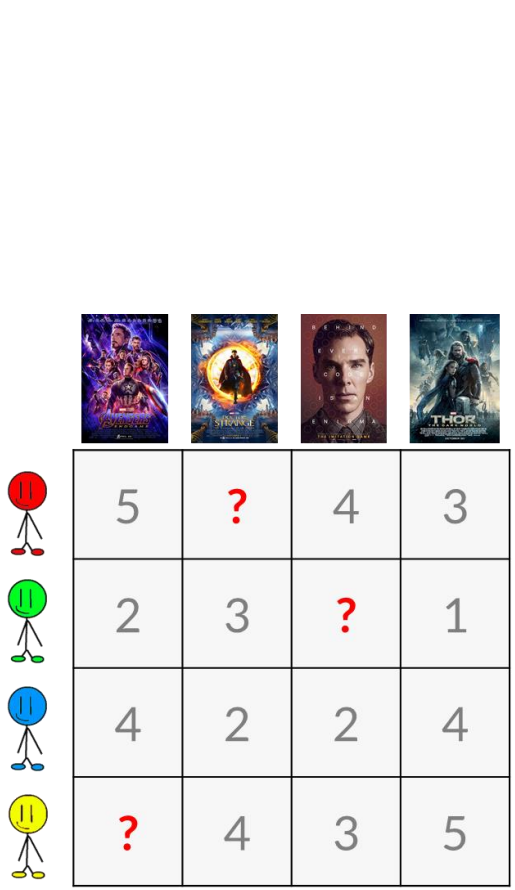
- Given the *observed entries*, recover the *missing entries*
  - $\{r_{ui}\}$: The set of *observed entries*
  - $\{\hat{r}_{ui}\}$: The set of *missing entries* (*to be recovered*)



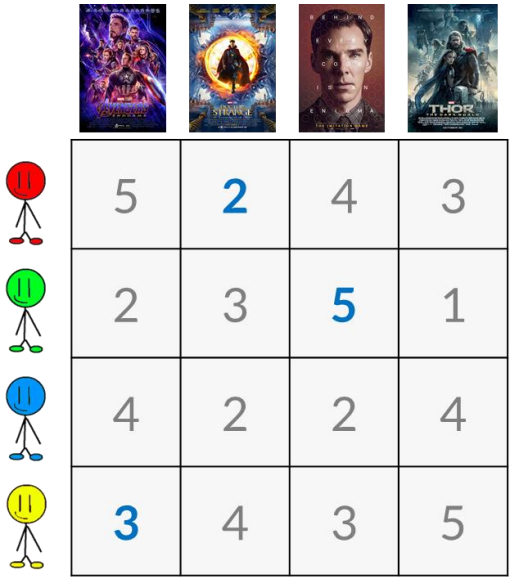
Rating Prediction – Evaluation

- Are we making good or bad recommendations?
  - Compare *predicted ratings* against *actual ratings*
- Evaluation (for rating prediction) are usually done using ***pointwise metrics***
  - Mean Absolute Error (**MAE**)
  - Root Mean Squared Error (**RMSE**)

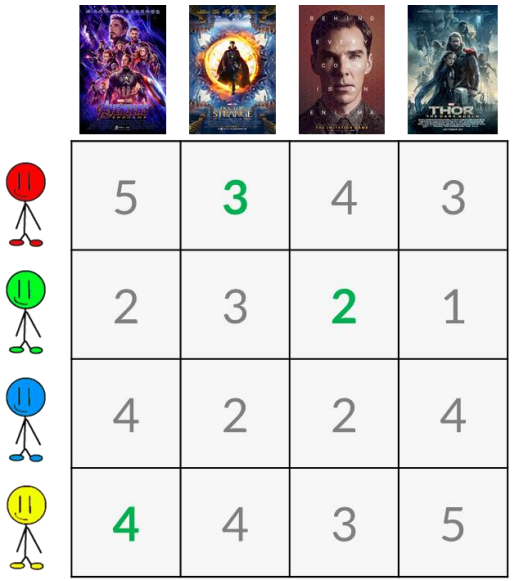
A Simple Example



Missing Entries
(in red)

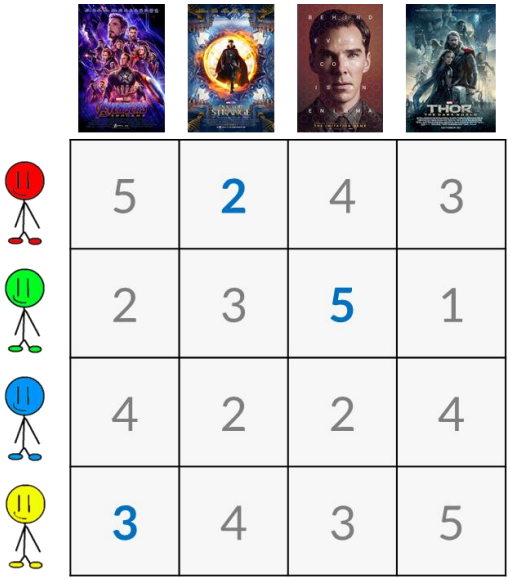


Predicted Ratings
(in blue)



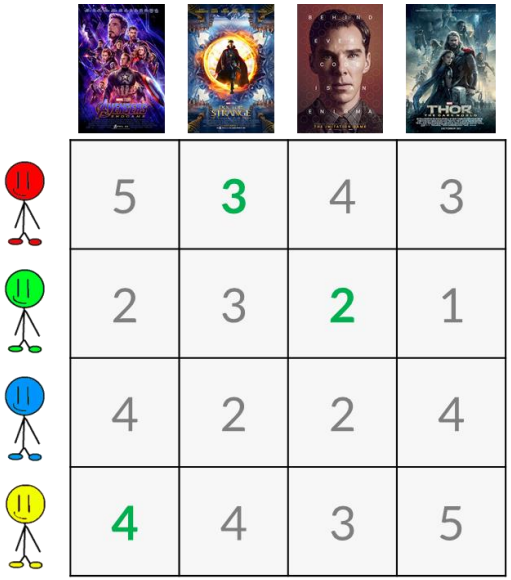
Actual Ratings
(in green)

A Simple Example – MAE & RMSE



	5	**2**	4	3
	2	3	**5**	1
	4	2	2	4
	3	4	3	5

Predicted Ratings
(in blue)



	5	**3**	4	3
	2	3	**2**	1
	4	2	2	4
	4	4	3	5

Actual Ratings
(in green)

- $MAE = (|2-3| + |5-2| + |3-4|)/3 \approx 1.66$
- $RMSE = \sqrt{((2-3)^2 + (5-2)^2 + (3-4)^2)/3} \approx 1.91$
  - As compared to MAE, RMSE penalizes *large prediction errors* more *(Errors are squared)*

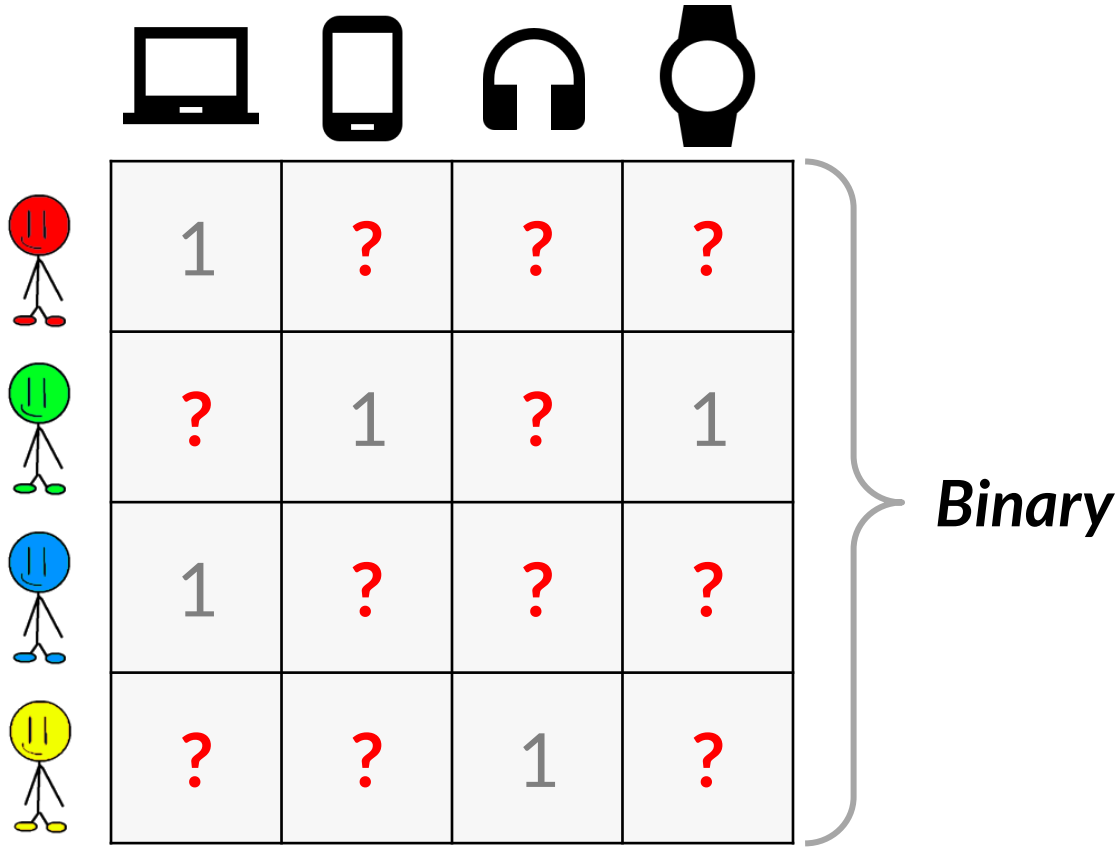
Implicit Feedback

- Explicit Feedback (e.g. ratings, reviews, …)
  - Users need to provide them consciously
  - Might not always be readily available

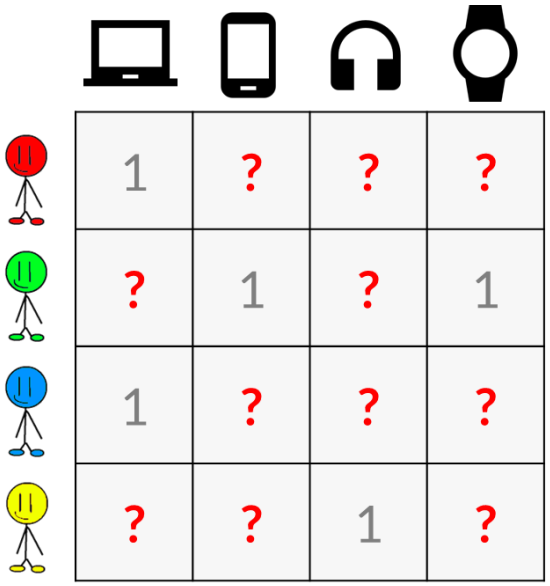
- **Alternative**: Implicit Feedback
  - Collected in the background
  - Examples:
    - User has *clicked* on a link
    - User has *watched* a movie
    - User has *viewed* a product
    - User has *purchased* a product

Implicit Feedback – Top-K Recommendation

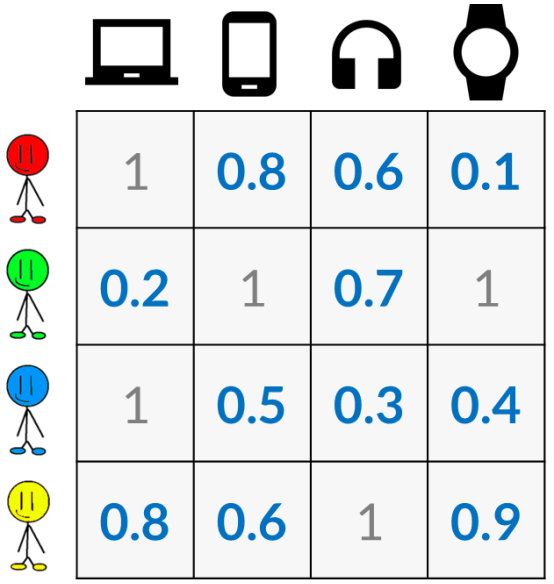
- Given the *observed entries*, predict the likelihoods for the *missing entries*
  - $\{r_{ui}\}$: The set of *observed entries*
  - $\{\hat{r}_{ui}\}$: The set of *missing entries* (*to be predicted*)



A Simple Example

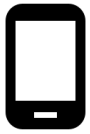


Missing Observations
(in red)



Predicted Likelihoods
(in blue)

Top-2 Recommendations for  **:**

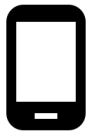


0.8

0.6

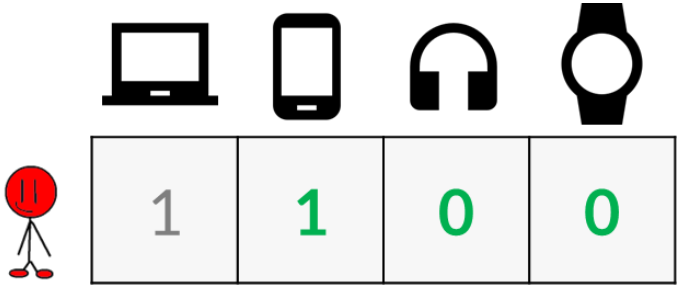
A Simple Example

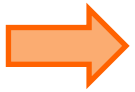
Top-2 Recommendations for  *:*




0.8 *0.6*

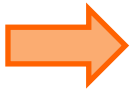
Scenario 1:





- User purchased a phone 
- Recommendation is **good**! ☺

Scenario 2:



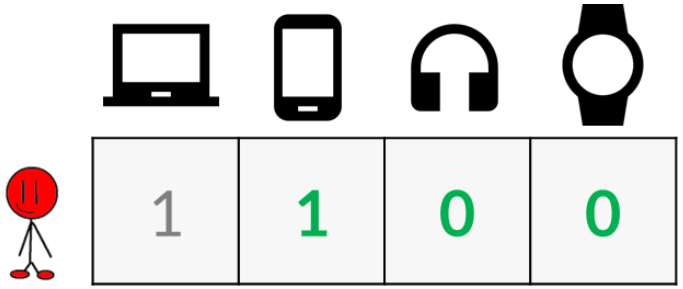
- User purchased a watch 
- Recommendation is **bad**! ☹

Actual Observations
(in green)

Top-K Recommendation – Evaluation

- Are we making good or bad recommendations?
  - Compare *actual observations* against the list of top-K recommended items
- Evaluation are usually done using ***classification metrics*** and/or **ranking-based metrics**
  - **Classification**: Precision, Recall, …
  - **Ranking-based**:
    - Normalized Discounted Cumulative Gain (**nDCG**)
    - Mean Reciprocal Rank (**MRR**)
    - …

Classification vs Ranking-based



Actual Observations
(in green)

Top-2 Recommendations for  **:**

Option 1

or

Option 2

- For **classification-based metrics**, Options 1 & 2 are equally good…
  - The phone (*actual purchase*) is in both top-2 lists
  - ***Order does <u>not</u> matter***
- For **ranking-based metrics**, Option 1 is better than Option 2!
  - The phone (*actual purchase*) is ranked above the headphones for Option 1 ☺

Implicit Feedback – Caveat

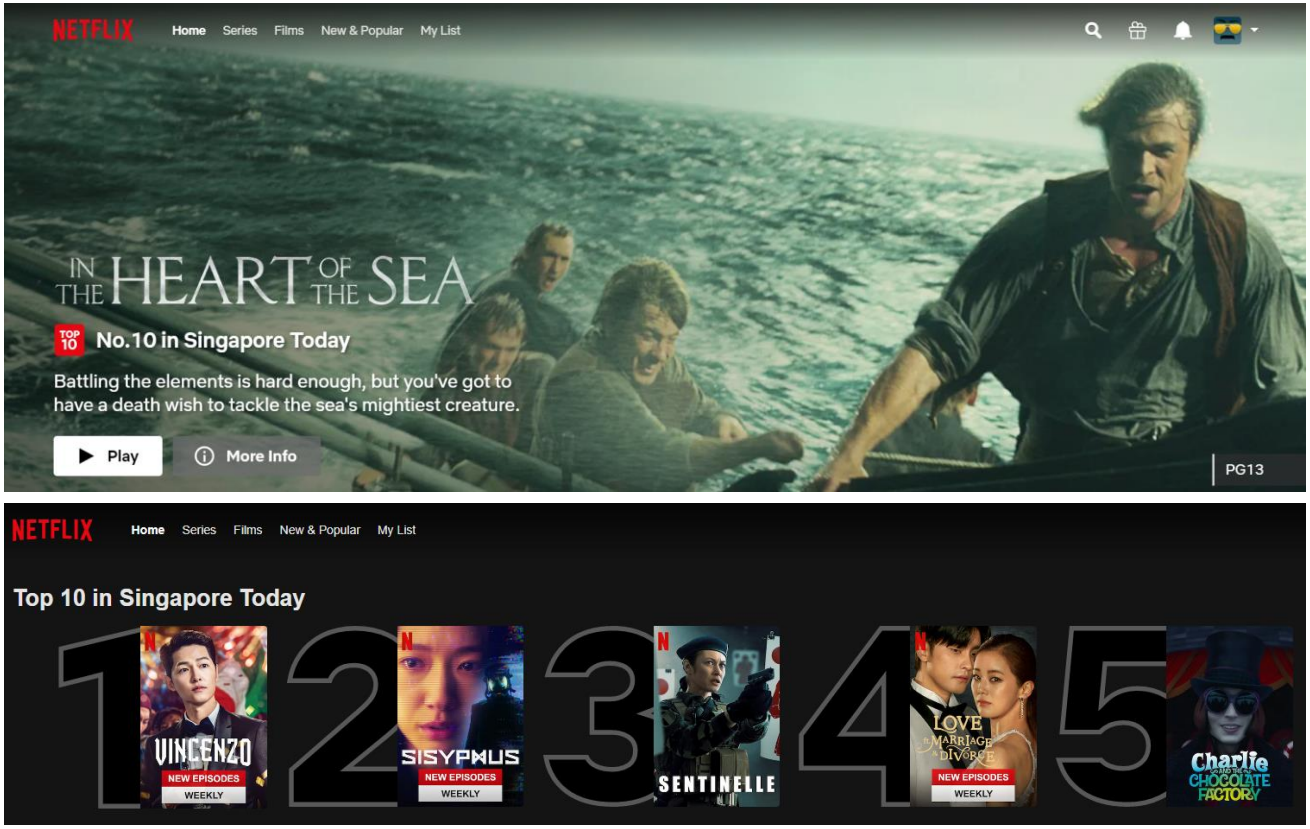
- E.g., User did not *watch* a movie
  - Possibility 1: User dislikes that movie
  - Possibility 2: User is *not aware* of such a movie
- E.g., User has *watched* a movie
  - Does the user like or dislike the movie?
- E.g., User *purchased* a product
  - Does the user like the product?
  - What if it was bought as a gift for someone else?
- For ***implicit feedback***, observations (or the lack thereof) do not necessarily reflect preferences

4.

HOW do we make recommendations?

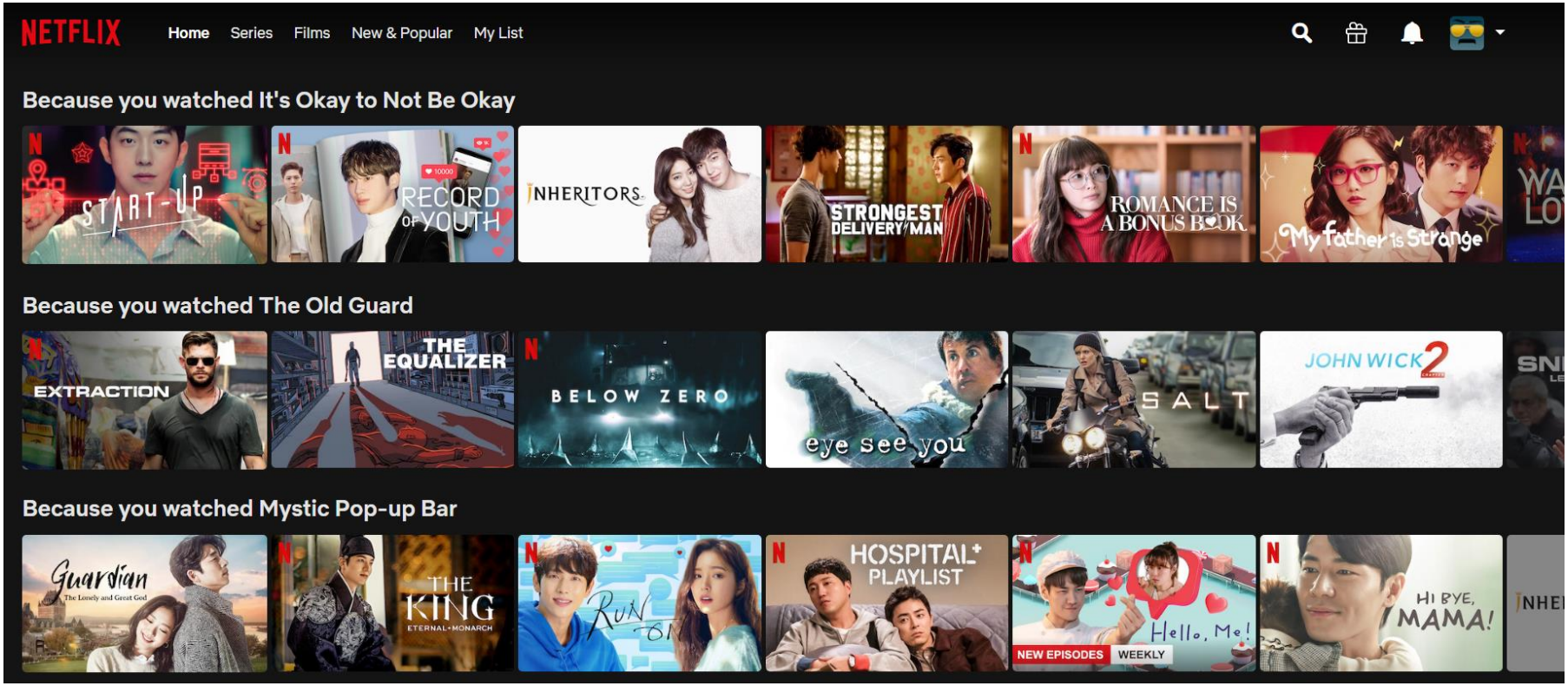
Making Recommendations

- **Non-personalized**
  - Random
  - Most Popular (e.g., Top 10 movies)
  - Most Recent (e.g., Latest uploads on YouTube)



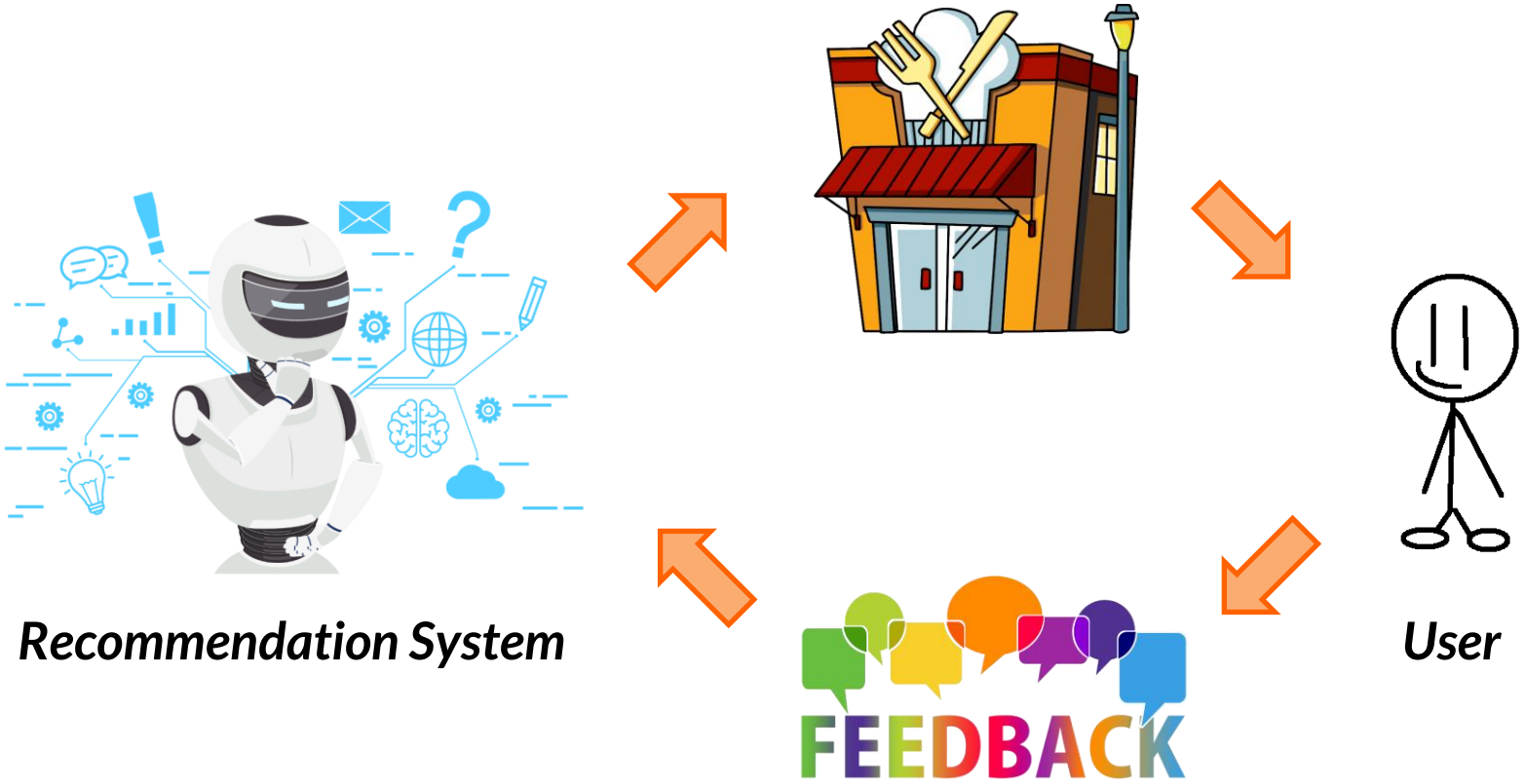
Making Recommendations

- **Personalized**
  - Adapts to the preferences of each user
  - In most cases, provides much better recommendations than non-personalized methods



Personalized Recommendations

- **Collaborative Filtering** (*Concept*)
- **Memory-based Approach** (*Methodology*)
- **Model-based Approach** (*Methodology*)



What is Collaborative Filtering?

- **Key Idea**: People often get the best recommendations from someone with *similar tastes* as themselves!

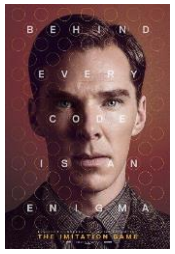
User

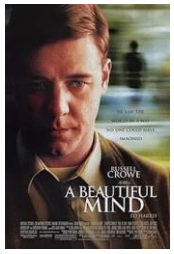
- Thor
- Iron Man 3
- Avengers: Endgame

Friend A

- Captain America
- Doctor Strange
- Avengers: ∞ War

Friend B





- The Imitation Game
- A Beautiful Mind
- Sherlock Holmes

- Friend A has similar preferences as the User
- Intuitively, Friend A's *recommendations* should be better than Friend B

What is Collaborative Filtering?

- **Key Idea**: People often get the best recommendations from someone with *similar tastes* as themselves!

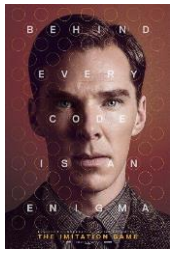
User

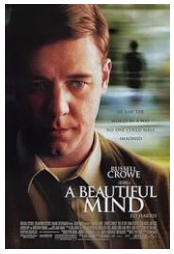
- Thor
- Iron Man 3
- Avengers: Endgame

Friend A

- Captain America
- Doctor Strange
- Avengers: ∞ War

Friend B





- The ***filtering*** (recommending a subset of items) is performed with the help of other <u>similar</u> users (i.e., in a ***collaborative*** manner)

What is Collaborative Filtering?

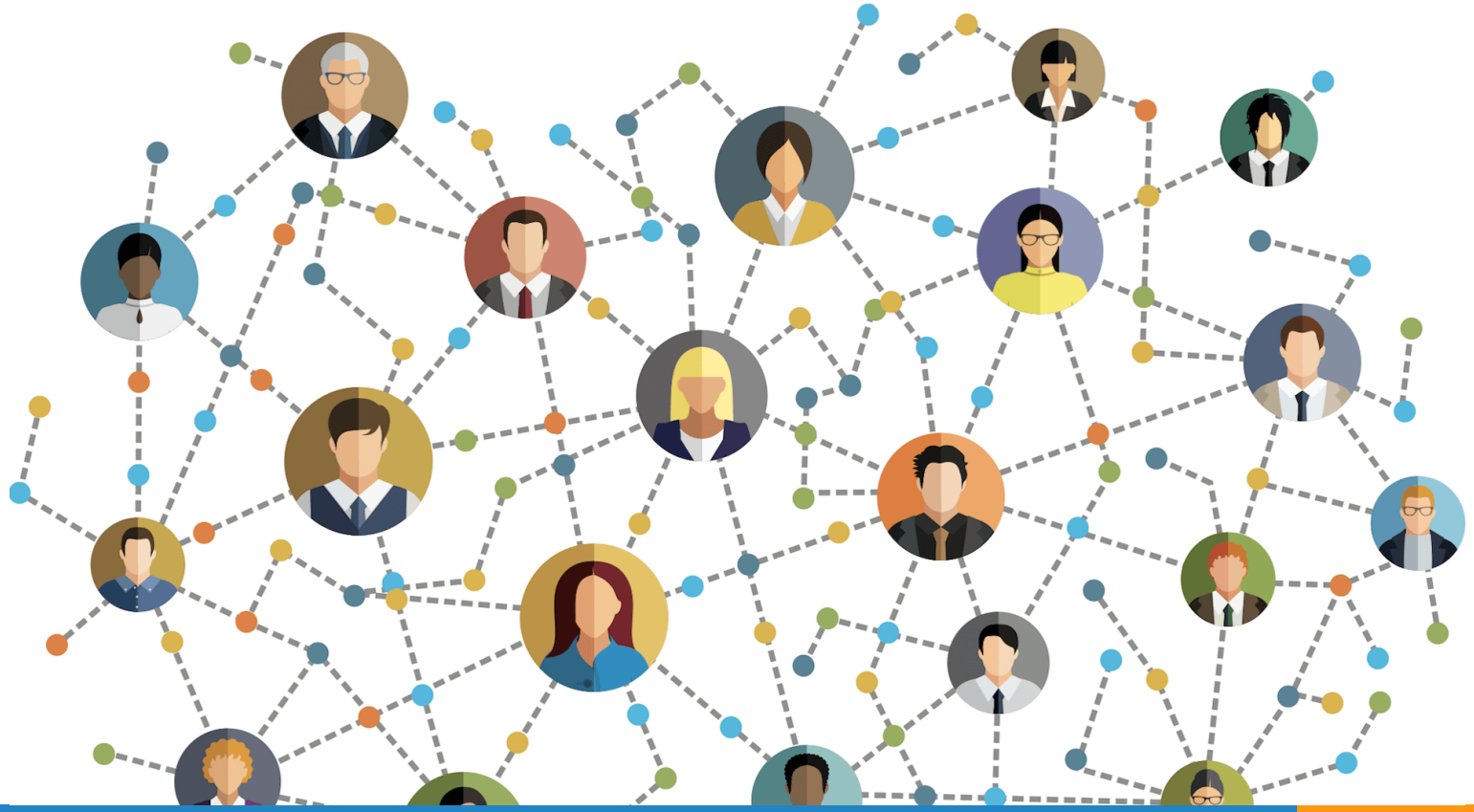
- **Key Idea**: People often get the best recommendations from someone with *similar tastes* as themselves!

- Relies (mostly) on the historical interactions
  - Of the *target user*
  - As well as those of other similar users!

- Not something new
  - Has been proposed a long time ago
  - However, it's still a very powerful tool
  - Most (if not all) recommendation systems are still based on this concept!

Memory-based Approaches

- **Memory-based Approach** (*Methodology*)
  - Based on similarities between users (or items)
  - Leverages upon the *historical interactions* of the *most similar users (or items)* to perform recommendations



Memory-based Approaches

- **Memory-based Approach** (*Methodology*)
  - Based on similarities between users (or items)
  - Leverages upon the *historical interactions* of the *most similar users (or items)* to perform recommendations
- **Similarity Measures**
  - Jaccard Similarity
  - Cosine Similarity
  - Pearson Correlation
  - …
- **Examples**
  - User-based k-Nearest Neighbour (**UserKNN**)
  - Item-based k-Nearest Neighbour (**ItemKNN**)

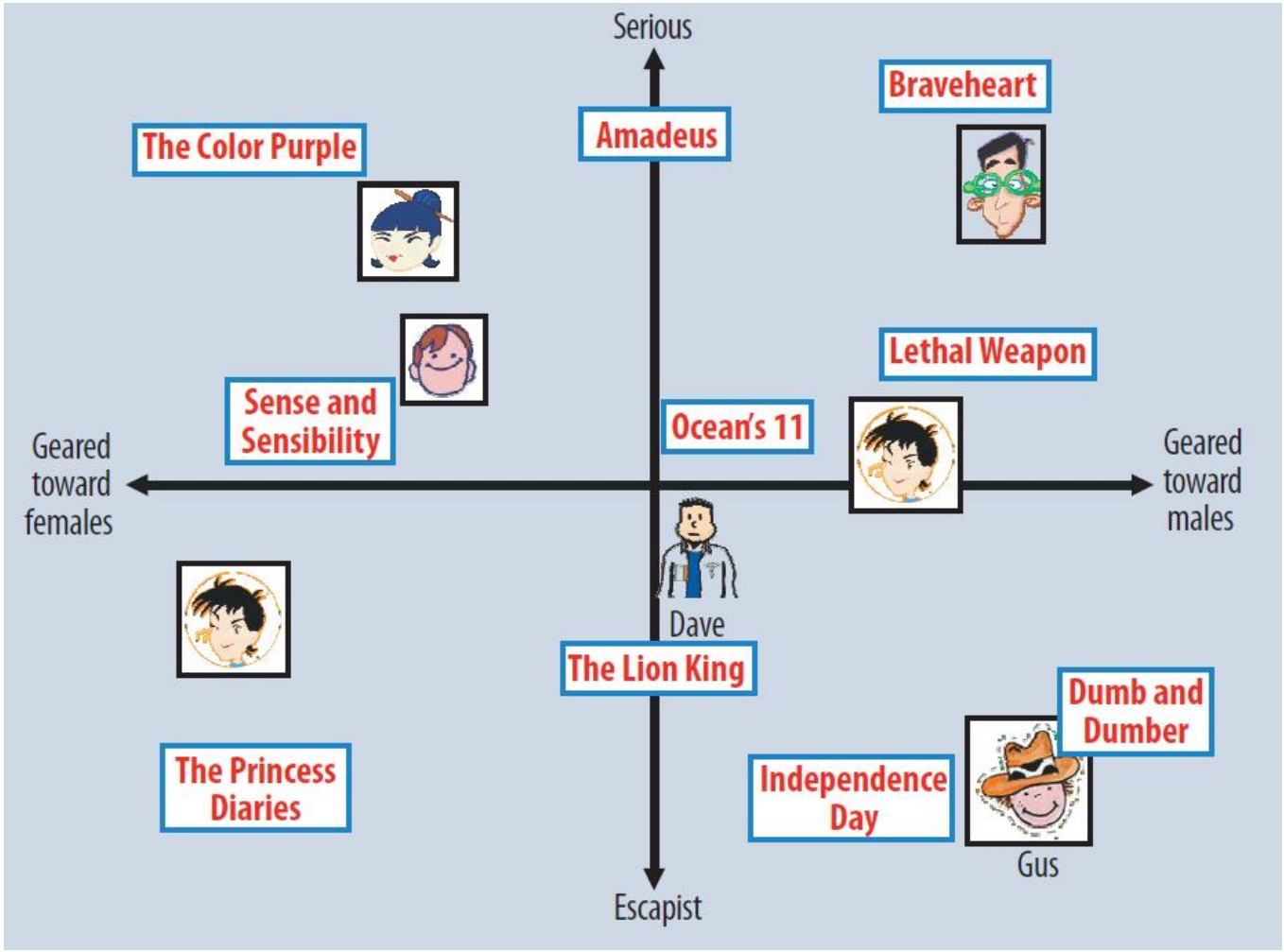
Memory-based Approaches

- **Benefits**
  - Intuitive & Explainable
  - *"We are recommending Movie X to you because other similar users (who have also watched Movies Y and Z) liked Movie X as well"*
- **Drawbacks**
  - Cannot handle sparse data well
    - E.g., If most users only interact with small number of items, it would be challenging (or maybe impossible) to find similar users
  - Scalability
    - Less computational efficient when the number of users and items grow

Model-based Approaches

- **Model-based Approach** (*Methodology*)
  - Models are built using different data mining (and more recently, machine / deep learning) techniques
- **Examples**
  - Bayesian Networks
  - Clustering-based Models
  - *Latent Factor Models (**LFMs**)*
    - Matrix Factorization (**MF**)
    - …
  - …

Latent Factor Models



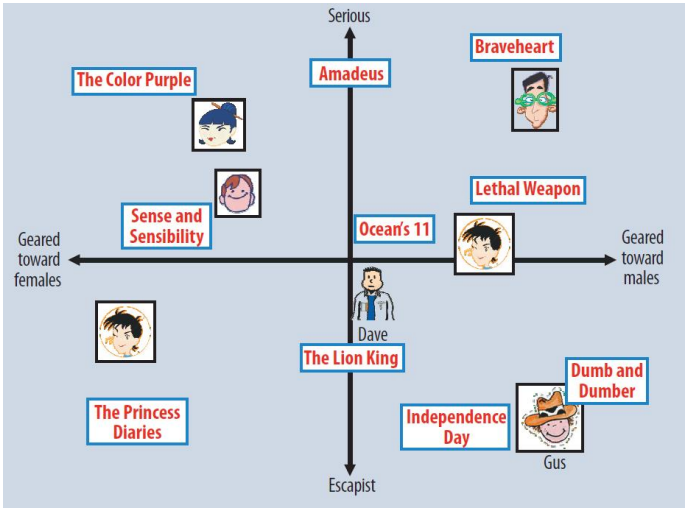
Users and items embedded in a 2-dimensional latent space (From [1])

[1] Matrix Factorization Techniques for Recommender Systems, IEEE Computer, Volume 42, Issue 8, Pages 30 – 37, 2009

Latent Factor Models

- Shared Latent Space
  - Each user is represented by a vector (of latent factors)
  - Each item is represented by a vector (of latent factors)

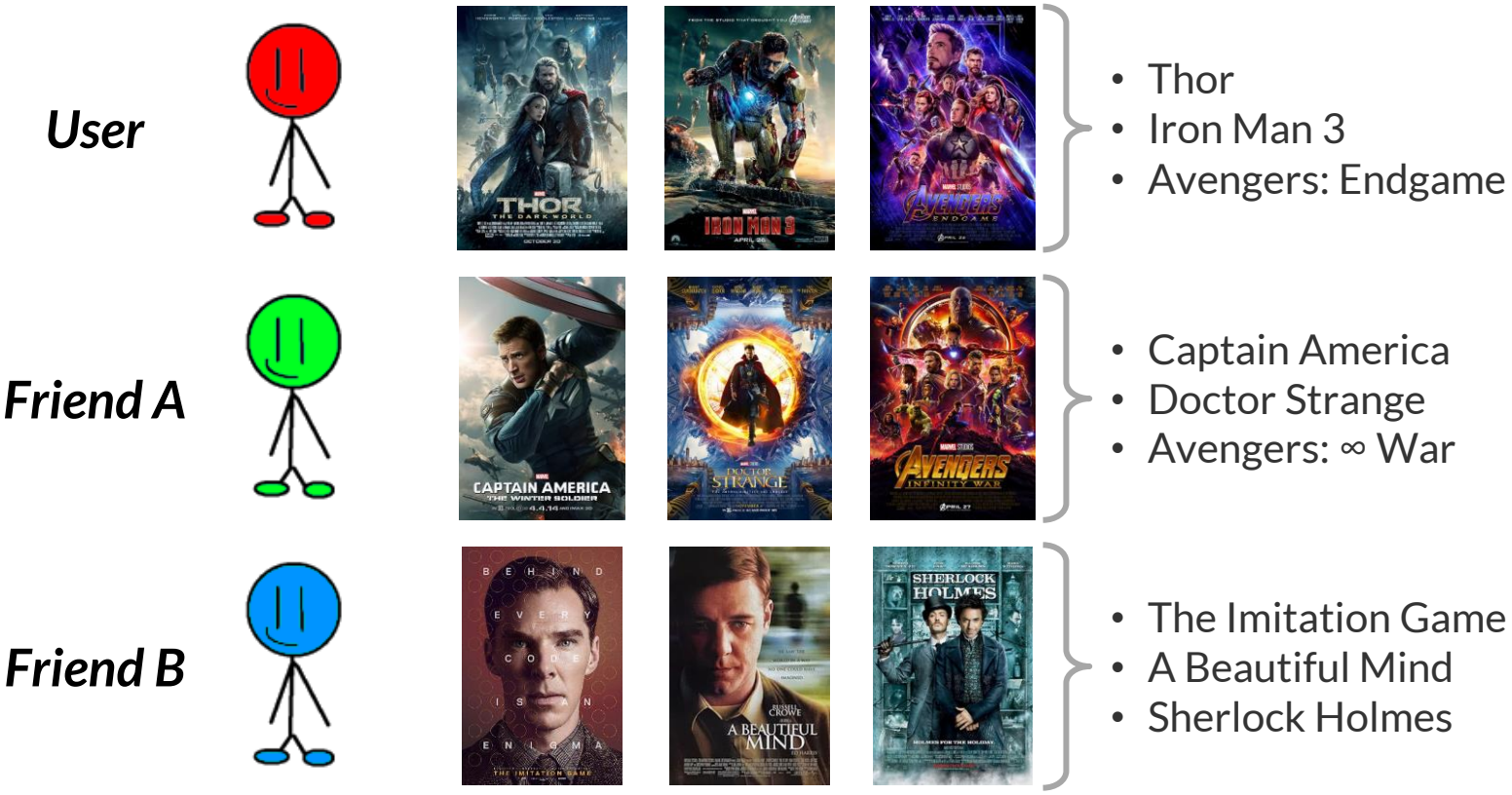
- Latent Factors
  - Learned automatically from historical interactions
  - These could be *movie genres*, *item attributes*, …
  - But they are latent, so we do not really know their true meaning



*** Latent**: hidden or concealed

Latent Factor Models

- Similar users are embedded close to each other
- Similar items are embedded close to each other



- *Distance*(User, Friend A) < *Distance*(User, Friend B)

Latent Factor Models

- How to compute the user-item rating (or likelihood) using latent factors?
  - **Simple method**: Dot product

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

a = 1st vector

b = 2nd vector

n = dimension of the vector space

a_i = component of vector a

b_i = component of vector b

  - **Alternatives**: Learning the user-item interaction function using *neural networks* (e.g. [1]), …

[1] Neural Collaborative Filtering, WWW 2017

Model-based Approaches

- **Benefits**
  - Works well even with sparse data
  - Efficient and scalable
  - *# of latent factors <<< # of users & # of items*
    - *User-item matrix* is compressed into a *low-dimensional latent space*
    - Dot product between two vectors is very fast
- **Drawbacks**
  - Interpretability
    - Black box
    - What is the true meaning of the latent factors? ☹

Hybrid Models

- E.g., Combination of memory-based & model-based
  - Overcomes the limitations of individual approaches
  - Could end up with increased complexity
- E.g., Training multiple models and combining their predictions

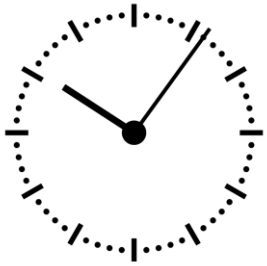
5.

Different types of recommendation systems

Different Types?

- Due to the different item types
  - E.g., Movies, News, Points of Interests (**POIs**), …

- Due to the available information
  - **Content**: Text, Images, User demographics, Item attributes, …
  - **Context**: Time, Location, …



POI Recommendation Systems

- In contrast to movies, books, etc., POIs are *'physical items'*
- When users interact with POIs, they are limited by ***physical constraints***
  - Distance
    - Recommendations cannot be too far away
  - Time
    - Travel time required to reach recommended location
- As a result, bad recommendations could be somewhat more *'costly'*

POI Recommendation Systems

- **Context-sensitive**
  - Distance
    - Current location (e.g., Home vs Workplace)
  - Time
    - Weekday vs Weekend
    - Time of the day (Morning/Afternoon/Night)

- *E.g.*, Workplace + Friday Evening → Restaurant / Bar

- E.g., Home + Saturday Evening → Cinema

- *E.g.*, Home + Sunday Morning → Park

POI Recommendation Systems

- **Locality**
  - People's activities in urban environments are usually *concentrated* in certain geographical regions
  - E.g., near our home or workplace

- **Challenges** for POI recommendation
  - Very sparse data [1]
    - Density for **Netflix** (*Movies*): 1.2%
    - Density for **Yelp** & **Foursquare** (POIs): 0.1%
  - Incorporating the rich context information adequately

[1] A Neural Approach for POI Recommendation, KDD 2017

Group Recommendation Systems

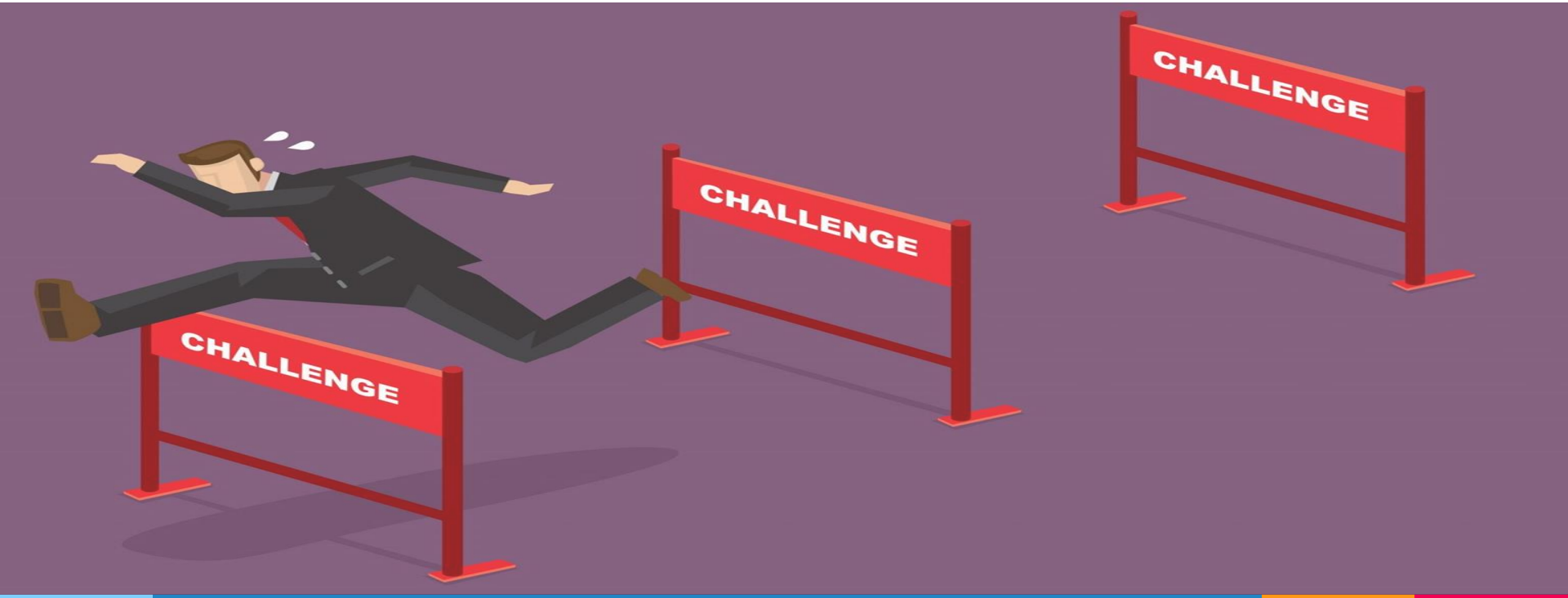
- Recommending items to a group of users
  - A group of co-workers
  - A group of friends
  - A couple
  - A family
  - …
- Users in a group tend to have…
  - Different preferences
  - Different levels of influence (Leaders vs Followers)
  - …
- To be covered later… ☺

6.

Challenges encountered by recommendation systems

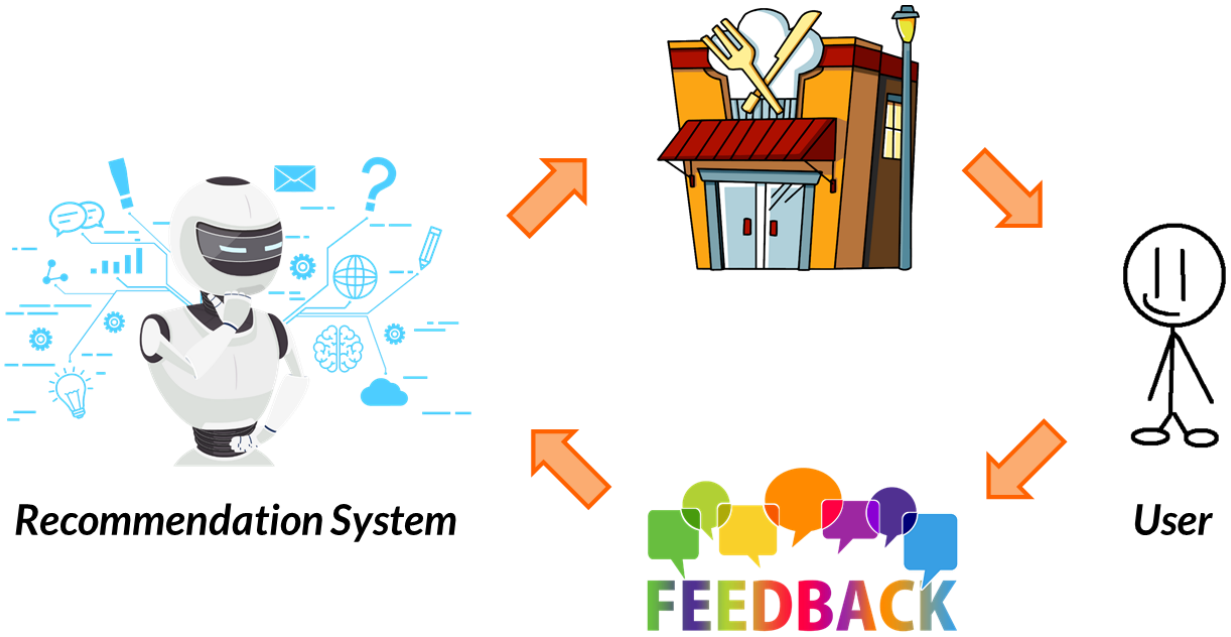
Challenges

- Cold-start Problem
- Popularity Bias
- Shilling Attacks
- …



Cold-start Problem

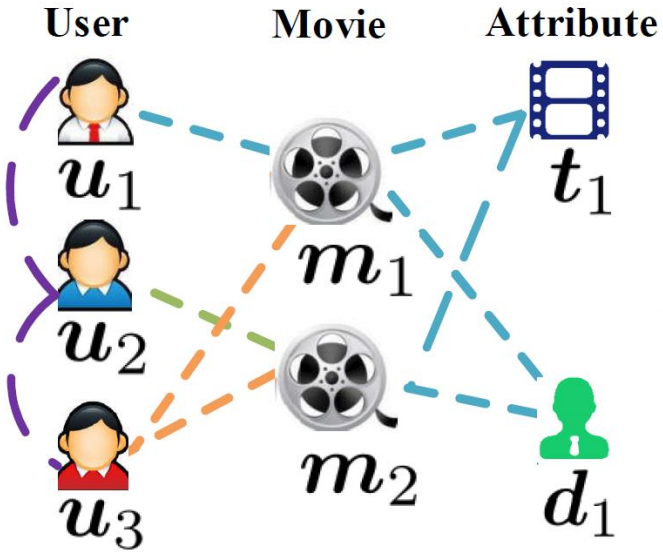
▷ Recommendation systems are **driven by user interactions**



- How about ***new users or items*** w/ no historical interactions?
- How about users or items w/ little historical interactions?
  - Difficult to *calculate similarity* and/or *derive a good representation*

Cold-start Problem

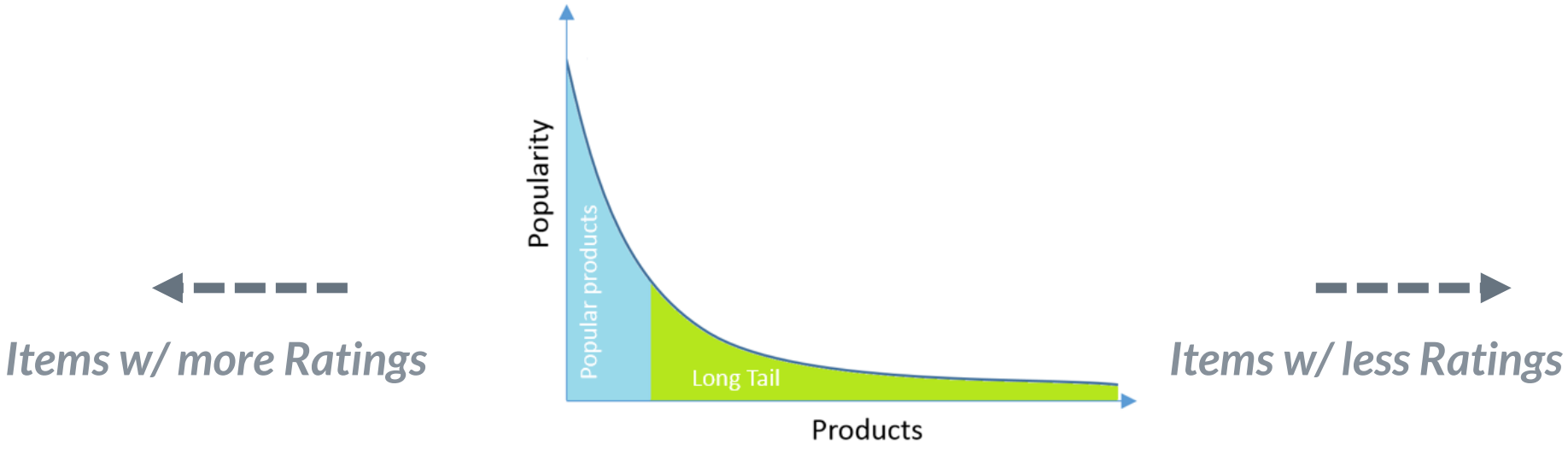
- In some cases, we could utilize the content information
  - E.g., **User demographics** (Age, Gender, Occupation, …)
  - E.g., **Item attributes** (Genre, Year, Actors, …)



- Possible solutions
  - Content-based Filtering
  - Hybrid Model (Collaborative Filtering + Content)

Popularity Bias

- Most recommendation systems suffer from ***popularity bias***
  - Popular items tend to get recommended over and over again
  - Less popular (or niche) products are less likely to be discovered



Shilling Attacks

- ***Collaborative Filtering***
  - "Users who shared similar preferences in the past will likely agree in the future as well"
  - Based on ***"wisdom of the crowd"***
    - i.e., what *others* think of an item or a group of items to compute recommendations
  - **Shortcoming**: *Unable* to distinguish *genuine user profiles* from fake ones

Shilling Attacks

- Characterised by…
  - Several ***fake user profiles***
  - Often by an ***adversarial party***
  - Harvest recommendation outcomes towards a ***malicious desire***

- ***"Malicious Desire"***
  - *Personal gain, market penetration, causing mischief*, etc.

Robustness Against Shilling Attacks

- Existing work tends to cover ***3 main directions***
  1. Attack Designs
  2. Detection Algorithms
  3. Defence Strategies

Thanks!

Any questions?

Email:
S160005@e.ntu.edu.sg