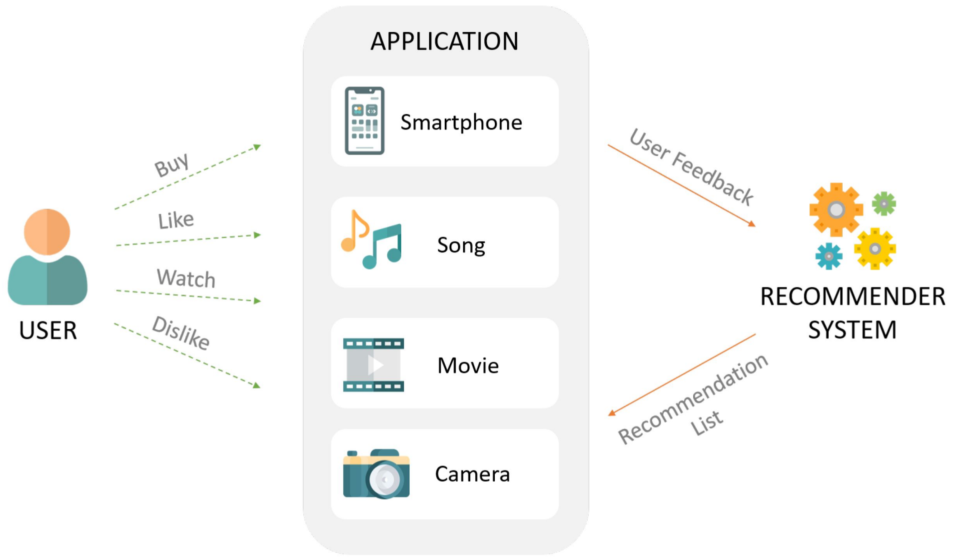
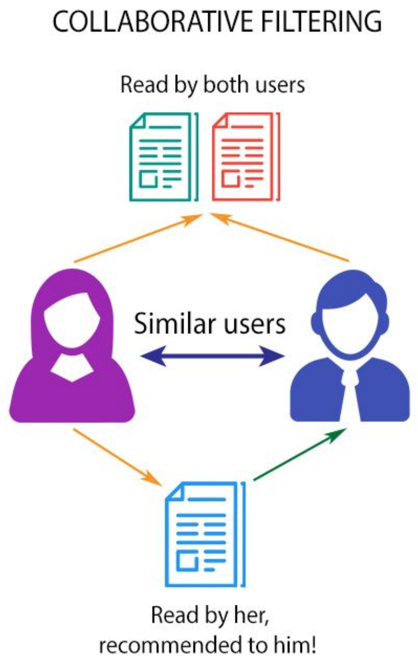
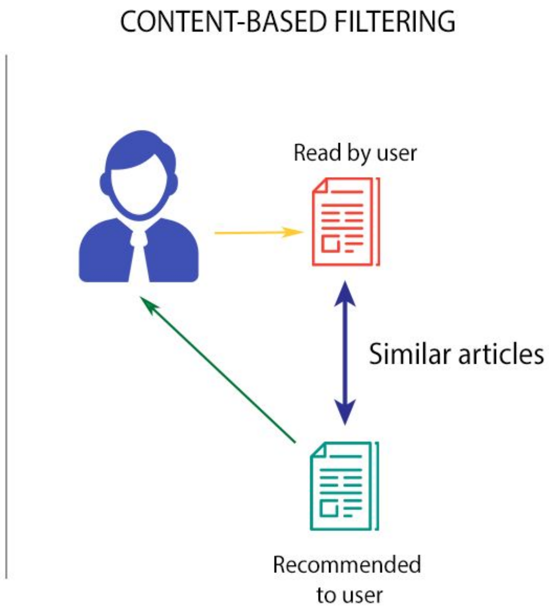
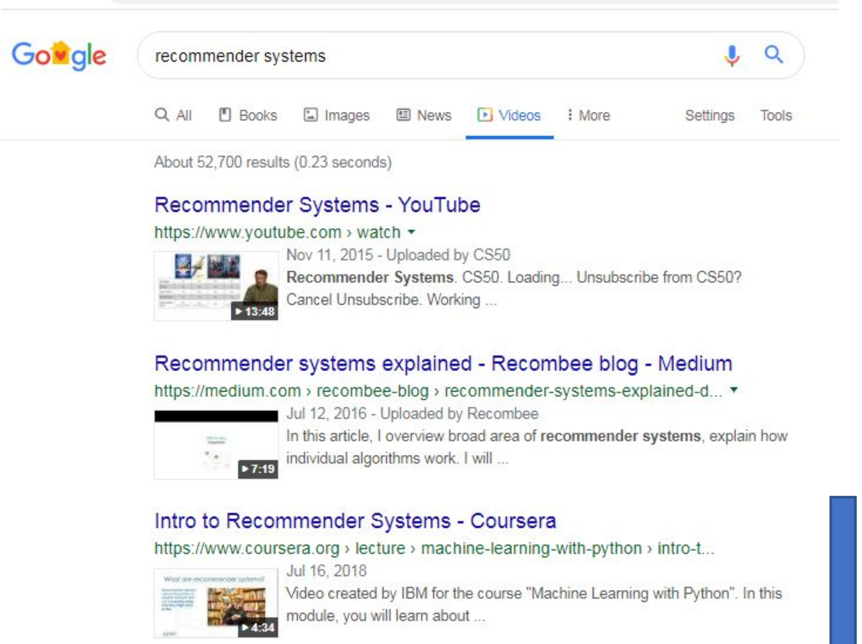


Recommender Systems





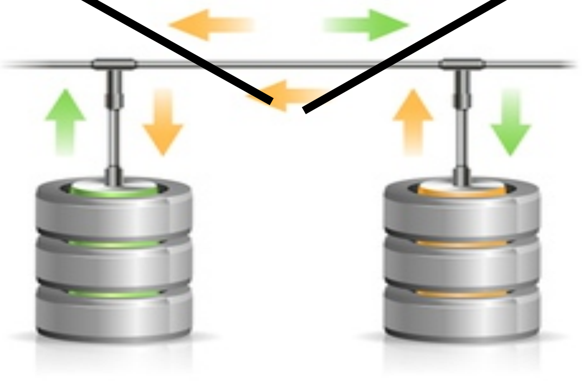
Search Engine
Vs
Recommender System



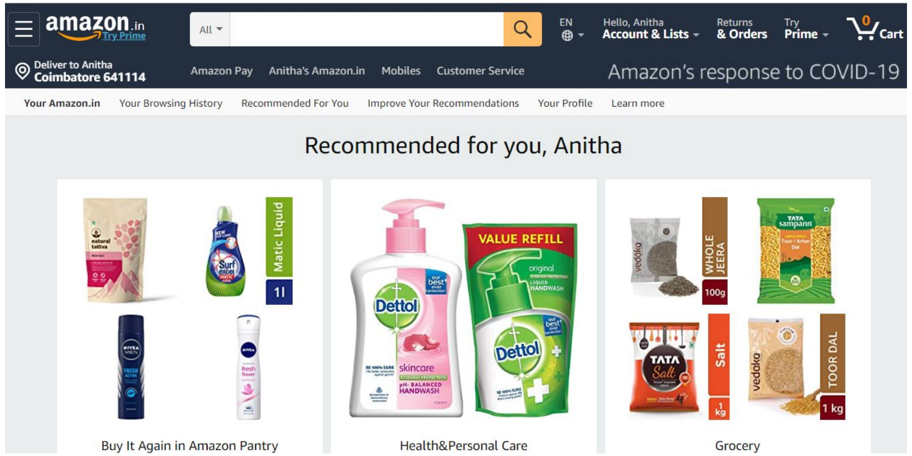
Recommender system:
- Push system
- System initiative
- Provide recommendation based on past preferences

INTERNET

Search engine:
- Pull system
- User initiative
- Work based on keywords



Problem:
- Difficult to filter the appropriate data from large volume of data
- Domain knowledge needed



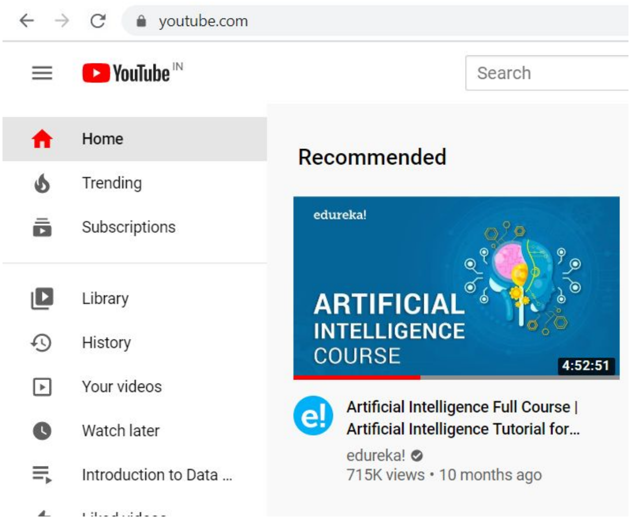
Difference

- **In a search engine system**, the user knows what he is looking for, and he makes the query
  - It searches in a database and identifies items that correspond to keywords or characters specified by the user
- **In a recommender system**, the system generates recommendations to users based on their past preferences.

- The main task of a search engine to **respond to the query** of a user by searching whatever matches with the keywords of the query whereas recommendation engine **shows only those items which are useful for the user** and discard others.

Recommender System

- Recommender system is one of the applications of machine learning.
- Recommender system is used in different ways in our daily lives.
  - From e-commerce to online advertisement applications use recommender systems
- A **Recommender System** is defined as a tool designed to **interact** with large and complex information spaces, capable of **predicting** the future preference of a set of items for a user, and **recommend** items that are likely to be of interest to the user, in an **automated fashion.**
- Many commercial websites suggests user's future preferences.
  - Movie recommender systems - Netflix, YouTube
  - Product recommender systems – Amazon, Flipkart
  - Others – LinkedIn, Facebook



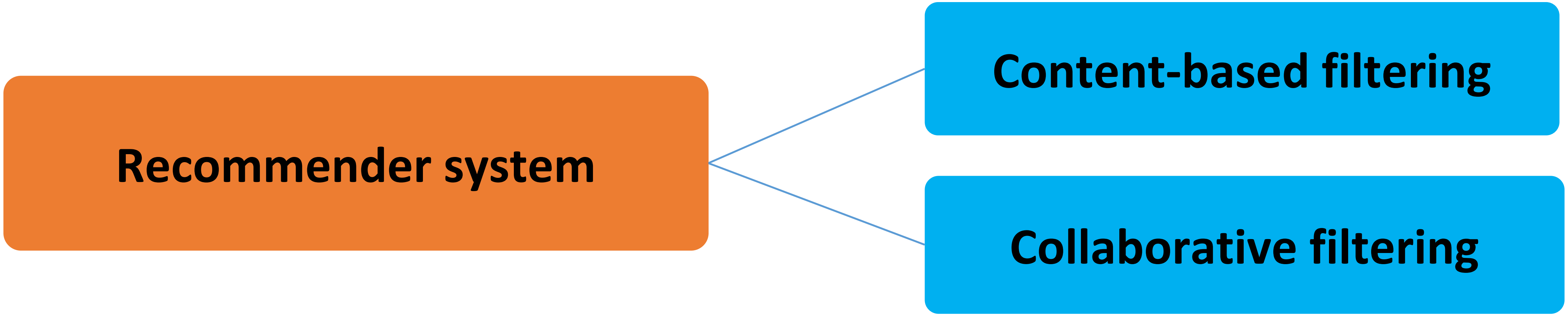
Why and when do we need recommender systems?

- In this Internet era, **the quantity of information is huge** and the recommender systems are extremely useful in several domains.
- People are not able to be experts in all these domains in which they are users, and they do not have enough time to spend looking for the perfect TV or book to buy.
- Companies using recommender systems focus on increasing sales as a result of very personalized offers and an enhanced customer experience.
- Recommender systems are really interesting when dealing with the following issues:
  - solutions for large amounts of good data;
  - reduction of cognitive load on the user;
  - allowing new items to be revealed to users.

- Data required for recommender systems come from:
  - ***Explicit* :** user ratings, comments and likes
  - ***Implicit* :** search queries, purchase histories and other information about user and item

- Recommender systems use two kinds of information:
  - ***Characteristic information*** : information about items (keywords, categories, etc.) and users (preferences, profiles, etc.).
  - ***User-item interactions*** : information such as ratings, number of purchases, likes, etc.

Types of recommender system



Hybrid Recommenders

- Hybrid approaches can be implemented in several ways:
  - by making content-based and collaborative predictions separately and then combining them;
  - by adding content-based capabilities to a collaborative approach (and vice versa);
  - by unifying the approaches into one model.

Formal Model :Recommender Systems

	Flash	Arrow	Spiderman	Batman	Supergirl
User1	1	5			2
User2		5			
User3				3	
User4	1		4	3	

- Recommendation System Model
- •Utility matrix (user-item interaction matrix)

  U: C x S → R

- Where C is set of m Customers and S is a set of n Items, and a matrix U with size m*n to denote the past ratings R of users. (clicked, watched, purchased, liked, rated, etc.)
- Each cell in the matrix represents the associated opinion that a user holds.
- For instance, U[i, j] denotes how user i likes item j.

- In reality, the user-item matrix can be more than millions * millions (e.g., Amazon, Youtube), and the majority of entries are missing
  - Utility matrix is sparse.
- The goal of recommender systems is to fill those missing entries.

Items / User	I1	I2	I3	I4	I5	I6	I7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Key problems

- Gathering known ratings for **Utility matrix.**
  - System gathering data either **explicitly** (rating) or **implicitly** (purchasing implies high rating).
- **Infer** unknown ratings from the known ones.
- Evaluating the **inference** methods.

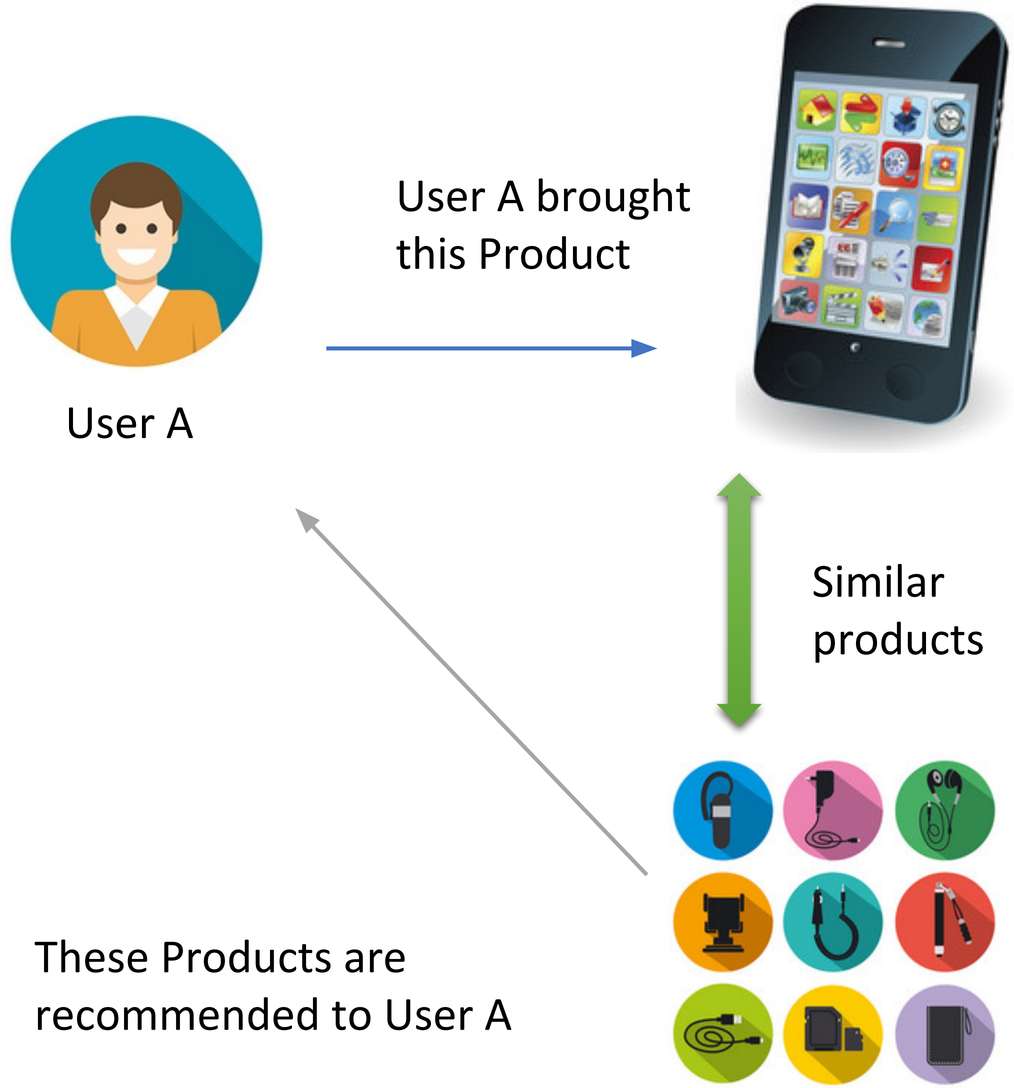
Items / User	I1	I2	I3	I4	I5	I6	I7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Modelling User Preferences

- Both, CBF and CF recommender systems, require to understand the user preferences.
- Movie recommender from Netflix, where the users rank the movies with 1 to 5 stars;
- Product recommender system from Amazon, where usually the tracking information of the purchases is used (0 - not bought; 1 - viewed; 2 - bought)
- The most common types of labels used to estimate the user preferences are:
    - Boolean expressions (is bought?; is viewed?)
    - Numerical expressions (e.g., star ranking)
    - Up-Down expressions (e.g., like, neutral, or dislike)
    - Weighted value expressions (e.g., number of reproductions or clicks)

Content based filtering approach

"Show me more of the same what I've liked"



- Content-based filter does not involve the other users.
- This approach will **recommend items which are similar/related** to those the user liked before.
- Content based approaches use **profile** about user's preferences and **descriptions** of items along with user-item interactions.
- As the user provides **more inputs or takes actions on those recommendations**, the engine becomes more and more accurate.

Music recommender system

music director, artist, melody, harmony, rhythm, instrumentation…

age, sex...

Item Description

User Profile

Content based filtering

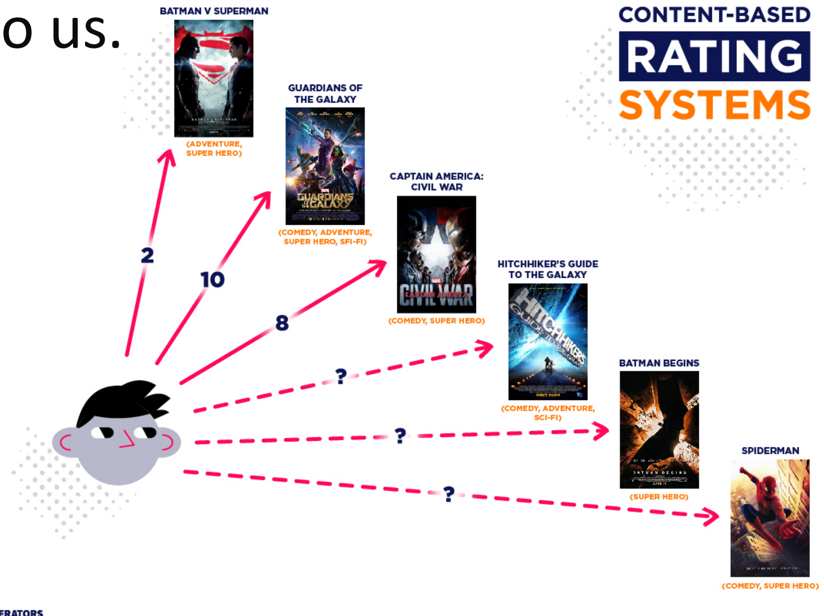
Music recommender

	Song1	Song2	Song3	Song4
Artist1	1	0	1	1
Artist2	0	1	0	0

Recommend similar song by Artist1

Movie recommendation system

- Whenever we are looking for a movie or web series on Netflix, we get the same genre movie recommended by Netflix.
- But how does this work? How does Netflix compute what I like?
- This is all done through content-based systems.
- The similarity of different movies is computed to the one you are currently watching and all the similar movies are recommended to us.



Text features

- Profile = set of "important" words in item (document)
- How to pick important words?
  - Usual heuristic from text mining is **TF-IDF** (Term frequency * Inverse Doc Frequency)

Sidenote: TF-IDF

f_{ij} = frequency of term (feature) i in doc (item) j

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

Note: we normalize TF to discount for "longer" documents

n_i = number of docs that mention term i

N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

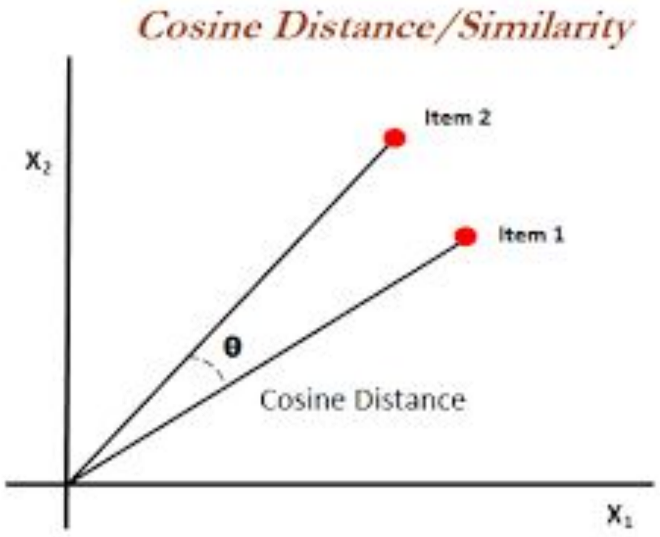
TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Doc profile = set of words with highest **TF-IDF** scores, together with their scores

- **Cosine Similarity:-** This type of metric is used to compute the similarity textual data.
- Consider an example where we have to find similar news or similar movies.

How is it done?

- We convert these textual data in the form of vectors and check for cosine angle between those two vectors if the angle between them is 0
- It means they are similar or else they are not.



$$sim(a,b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} {a_i}^2}\sqrt{\sum_{i=1}^{n} {b_i}^2}}$$

Sidenote: TF-IDF

f_{ij} = frequency of term (feature) i in doc (item) j

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

Note: we normalize TF to discount for "longer" documents

n_i = number of docs that mention term i

N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Doc profile = set of words with highest **TF-IDF** scores, together with their scores

- d1: the best Italian restaurant enjoy the best pasta
- d2: American restaurant enjoy the best hamburger
- d3: Korean restaurant enjoy the best bibimbap
- d4: the best the best American restaurant

Total word counts:

d1=8

d2=6

d3=6

d4=6

	Italian	restau rant	enjoy	the	best	pasta	American	ham burger	Korean	bibimbap
d1	1	1	1	2	2	1	0	0	0	0
d2	0	1	1	1	1	0	1	1	0	0
d3	0	1	1	1	1	0	0	0	1	1
d4	0	1	0	2	2	0	1	0	0	0

- TF = how frequently a term occurs in a document
- IDF = Log (Total # of Docs / # of Docs with the term in it)

word	TF				IDF	TF * IDF			
	d1	d2	d3	d4		d1	d2	d3	d4
Italian	1/8	0/6	0/6	0/6	log(4/1)=0.6	0.075	0	0	0
Restaurant	1/8	1/6	1/6	1/6	log(4/4)=0	0	0	0	0
enjoy	1/8	1/6	1/6	0/6	log(4/3)=0.13	0.016	0.02	0.02	0
the	2/8	1/6	1/6	2/6	log(4/4)=0	0	0	0	0
best	2/8	1/6	1/6	2/6	log(4/4)=0	0	0	0	0
pasta	1/8	0/6	0/6	0/6	log(4/1)=0.6	0.075	0	0	0
American	0/8	1/6	0/6	1/6	log(4/2)=0.3	0	0.05	0	0.05
hamburger	0/8	1/6	0/6	0/6	log(4/1)=0.6	0	0.1	0	0
Korean	0/8	0/6	1/6	0/6	log(4/1)=0.6	0	0	0.1	0
bibimbap	0/8	0/6	1/6	0/6	log(4/1)=0.6	0	0	0.1	0

	Italian	restaurant	enjoy	the	best	pasta	American	hamburger	Korean	bibimbap
d1	0.075	0	0.016	0	0	0.075	0	0	0	0
d2	0	0	0.02	0	0	0	0.05	0.1	0	0
d3	0	0	0.02	0	0	0	0	0	0.1	0.1
d4	0	0	0	0	0	0	0.05	0	0	0

TF-IDF Matrix

	d1	d2	d3	d4
d1	1			0
d2		1		0.5
d3			1	0
d4	0	0.5	0	1

Cosine Similarity

sim(d4,d1) =

$$\underline{(0.075*0) + (0*0) + (0.016*0) + (0*0) + (0*0) + (0.075*0) + (0*0.05) + (0*0) + (0*0) + (0*0)}$$

$$\boldsymbol{sim(a,b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} {a_i}^2}\sqrt{\sum_{i=1}^{n} {b_i}^2}}}$$

Search document:
The best the best American restaurant

Recommendation:
1) Document d4
2) Take the corresponding row from cosine similarity matrix
3) Sort it in reverse
[d4=1, d3=0, d2=0.4, d1=0]
4) Recommend d2 with highest similarity

- d1: the best Italian restaurant enjoy the best pasta
- d2: American restaurant enjoy the best hamburger
- d3: Korean restaurant enjoy the best bibimbap
- d4: the best the best American restaurant

	d1	d2	d3	d4
d1	1			0
d2		1		0.5
d3			1	0
d4	0	0.4	0	1

Document	TF-IDF Bag of Words	Cosine similarity with d4
The best Italian restaurant enjoy the best pasta	[0.075, 0, 0.016, 0, 0, 0.075, 0, 0, 0, 0]	0
American restaurant enjoy the best hamburger	[0, 0, 0.02, 0, 0, 0, 0.05, 0.1, 0, 0]	0.5
Korean restaurant enjoy the best bibimbap	[0, 0, 0.02, 0, 0, 0, 0, 0, 0.1, 0.1]	0
The best the best American restaurant	[0, 0, 0, 0, 0, 0, 0.05, 0, 0, 0]	1

What is the primary technique used in content-based recommendation systems?

a) Collaborative filtering

b) Matrix factorization

c) Natural language processing

d) Feature extraction and similarity calculation

In content-based recommendation, what does the "content" refer to?

a) User preferences

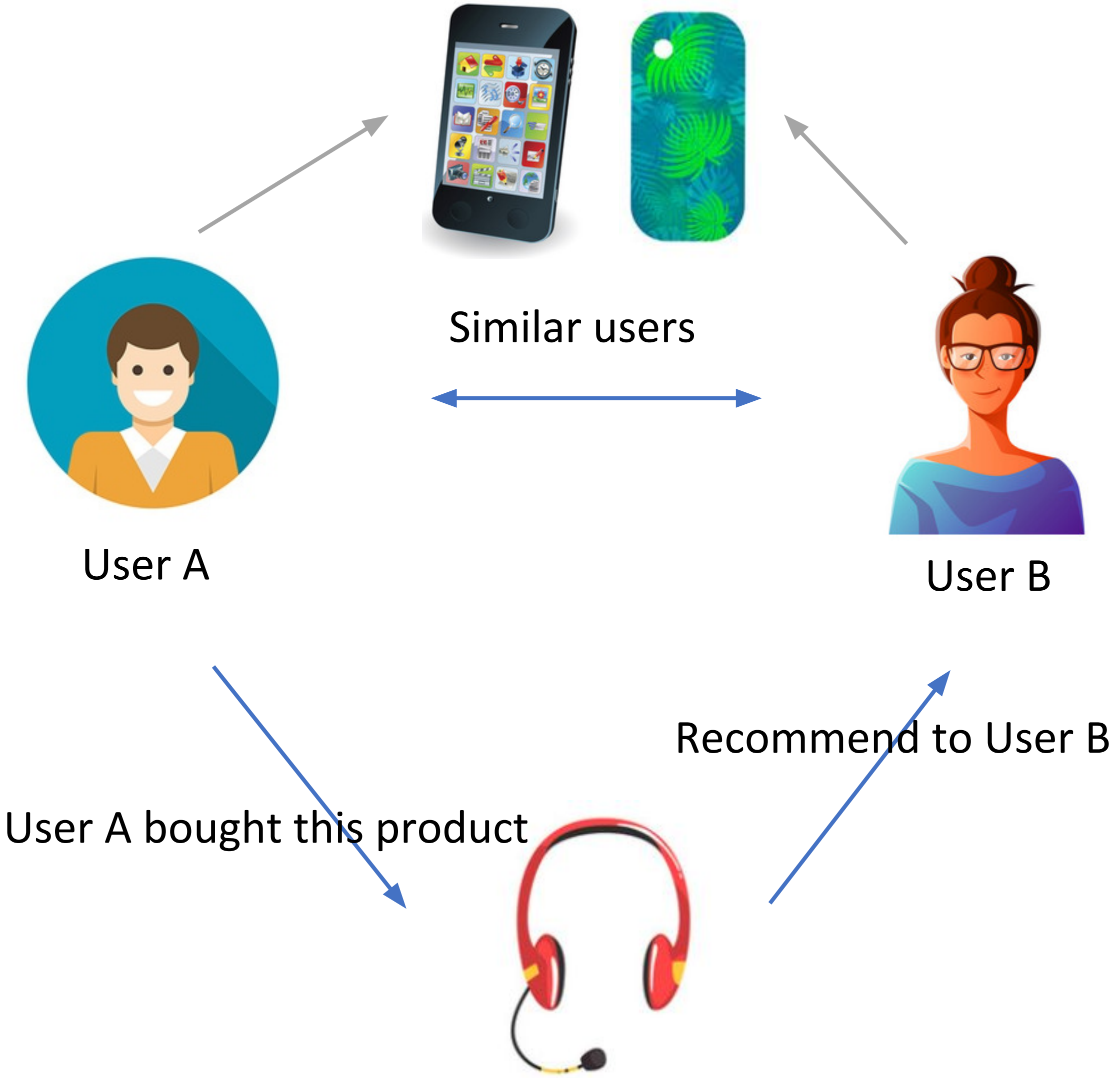
b) Ratings given by users

c) Characteristics or features of items

d) Social connections between users

Collaborative filtering approach

“Tell me what’s popular among my like-minded users”



similar users tend to like similar

- Collaborative filtering approaches recommend new product based on the past interactions recorded between users and items.
- The key idea behind this is that **similar users share the same interest** and recommend similar items are liked by a user.
- Collaborative filtering is one of the most frequently used approaches and usually provides better results than content-based recommendations.

There are two categories of Collaborative Filtering:

- **User-based**: measure the similarity between target users and other users.
- **Item-based**: measure the similarity between the items that target users rates/ interacts with and other items.

- One of the **main advantages** of this type of system is that it does not need to “understand” what the item it recommends is.

- The **main drawbacks** of this kind of method is the need for a user community, as well as the ***cold-start*** **effect** for new users in the community.
    - The cold-start problem appears when the system cannot draw any inference or recommendation for the users (or items) since it has not yet obtained the sufficient information of them.

Item Profiles

- For each item, create an **item profile**
- Profile is a set of features
  - **Movies:** author, title, actor, director,…
  - **Images, videos:** metadata and tags
  - **People:** Set of friends

User Profiles

- User has rated items with profiles $\mathbf{i_1},\ldots,\mathbf{i_n}$
- Simple: (weighted) average of rated item profiles
- Variant: Normalize weights using average rating of user

Formal model : Recommender systems

- **Utility matrix (user-item interaction matrix)**

  U: C x S → R

- Where **C** is set of ***m*** Customers and **S** is a set of ***n*** Items, and a matrix U with size ***m*n*** to denote the **past ratings R of users.** (clicked, watched, purchased, liked, rated, etc.)
- Each cell in the matrix represents the associated opinion that a user holds.
- For instance, **U[i, j] denotes how user i likes item j.**

Items / User	I1	I2	I3	I4	I5	I6	I7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- In reality, the user-item matrix can be more than millions * millions (e.g., Amazon, Youtube), and the majority of entries are missing
  - Utility matrix is sparse.
- The goal of recommender systems is to fill those missing entries.

Items / User	I1	I2	I3	I4	I5	I6	I7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Key problems

- Gathering known ratings for **Utility matrix.**
  - System gathering data either **explicitly** (rating) or **implicitly** (purchasing implies high rating).
- **Infer** unknown ratings from the known ones.
- Evaluating the **inference** methods.

Items / User	I1	I2	I3	I4	I5	I6	I7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Collaborative filtering

1. Build a **utility matrix** of items each user brought/viewed/rated.
2. Compute **similarity score** between users.
3. Find the k similar users.
4. Recommend the item they bought/viewed/rated that the user haven't yet.

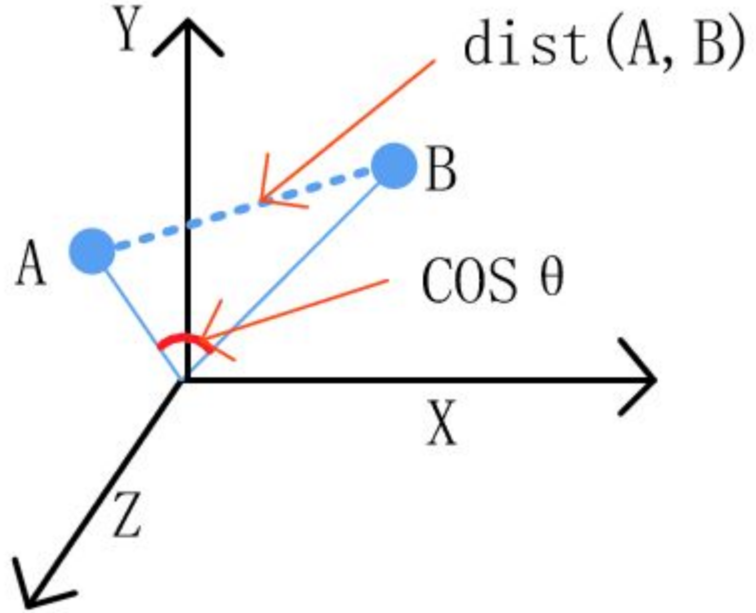
Utility matrix
or
User-item interactions matrix

Items / Users	I1	I2	I3	I4	I5	I6	I7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Similarity Score

- There are two options to calculate similarity score,

1. Cosine similarity.
2. Centered cosine similarity (Pearson Correlation)



$$sim(a,b) = \cos(a,b)$$

$$\cos(\mathrm{a},\mathrm{b}) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} {a_i}^2}\sqrt{\sum_{i=1}^{n} {b_i}^2}}$$

Similarity score – Cosine

$$sim(a,b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} {a_i}^2}\sqrt{\sum_{i=1}^{n} {b_i}^2}}$$

STEP 1:

Utility matrix : U

	I1	I2	I3	I4	I5	I6	I7
A	4	0	0	5	1	0	0
B	5	5	4	0	0	0	0
C	0	0	0	2	4	5	0
D	0	3	0	0	0	0	3

STEP 2:

	A	B	C	D
A	1	0.38	0.32	0
B		1	0	0.44
C			1	0
D				1

Similarity score

$$sim(A,B) = \frac{\sum_{i=1}^{7} U[A,Ii].U[B,Ii]}{\sqrt{\sum_{i=1}^{7} U[A,Ii]^2}\sqrt{\sum_{i=1}^{7} U[B,Ii]^2}}$$

$$= \frac{(4*5) + (0*5) + (0*4) + (5*0) + (1*0) + (0*0) +}{\sqrt{(4^2 + 5^2 + 1^2)}\sqrt{(5^2 + 5^2 + 4^2)}}$$

=0.38

STEP 3: Consider k similar users. Here K = 1

Similarity score - Pearson

$$sim(a,b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$$

Utility matrix

	I1	I2	I3	I4	I5	I6	I7
A	4 – 10/3	0	0	5-10/3	1-10/3	0	0
B	5-14/3	5-14/3	4-14/3	0	0	0	0
C	0	0	0	2-11/3	4-11/3	5-11/3	0
D	0	3-6/2	0	0	0	0	3-6/2

Mean
A-10/3
B-14/3
C-11/3
D-6/2

1. Calculate mean value
2. Calculate centered value

Centered utility matrix

	I1	I2	I3	I4	I5	I6	I7
A	2/3	0	0	5/3	-7/3	0	0
B	1/3	1/3	-2/3	0	0	0	0
C	0	0	0	-5/3	1/3	4/3	0
D	0	0	0	0	0	0	0

Centered utility matrix

	I1	I2	I3	I4	I5	I6	I7
A	2/3	0	0	5/3	-7/3	0	0
B	1/3	1/3	-2/3	0	0	0	0
C	0	0	0	-5/3	1/3	4/3	0
D	0	0	0	0	0	0	0

$$sim(A,B) = \frac{\sum_{i=1}^{7} U[A,Ii].U[B,Ii]}{\sqrt{\sum_{i=1}^{7} U[A,Ii]^2}\sqrt{\sum_{i=1}^{7} U[B,Ii]^2}}$$

$$= \frac{(\frac{2}{3}*\frac{1}{3}) + (0*\frac{1}{3}) + (0*\frac{-2}{3}) + (\frac{5}{3}*0) + (\frac{-7}{3}*0) + (0*0) +}{\sqrt{(\frac{2}{3}^2 + \frac{5}{3}^2 + \frac{-7}{3}^2)}\sqrt{(\frac{1}{3}^2 + \frac{1}{3}^2 + \frac{-2}{3}^2)}}$$

=0.09

	A	B	C	D
A	1	0.09	-0.55	-
B	0.09	1	0	-
C	-0.55	0	1	-
D	-	-	-	1

Similarity score

Rating prediction

- Let user X's ratings be R_x
- Let N be the set of K users, most similar to user X, who also rated item i
- Prediction of user X for item i

Option 1:

$$R_{xi} = \frac{1}{K}\sum_{y \in N} R_{yi}$$

= 1/1 * 5 = 5

Option 2:

$$R_{xi} = \sum_{y \in N} sim(x,y) * R_{yi} / \sum_{y \in N} sim(x,y)$$

= 0.09*5 / 0.09 = 5

	I1	I2	I3	I4	I5	I6	I7
A	4	5		5	1		
B	5	5	4				
C				2	4	5	
D		3					3

	A	B	C	D
A	1	0.09	-0.55	-
B	0.09	1	0	-
C	-0.55	0	1	-
D	-	-	-	1

No.	Collaborative Recommendation Engine	Content Based Recommendation Engine
1.	A collaborative recommendation engine emphasizes on the user preference.	A content based recommendation engine emphasizes on the content features.
2.	In collaborative filtering, a recommendation engine requires the user profile to suggest relevant content.	In content based filtering, a recommendation system uses the content profile too which includes the content features.
3.	The collaborative recommendation systems feed on the user ratings, reviews, thumbs ups & downs, and other feedback on various products or services. So, the products with no ratings or feedback can't be recommended to any user. Neither a new user who hasn't given any reviews or ratings can get any recommendation by the collaborative recommendation engine. This is called the cold **start problem.**	The content-based recommendation systems are product features oriented and hence don't have such problems.
4.	A collaborative recommendation engine doesn't always ensure precise recommendations. Because the users with similar tastes may not like the same products always.	A content based recommendation engine can provide more accurate recommendations as it focuses on the features of the content a user likes.