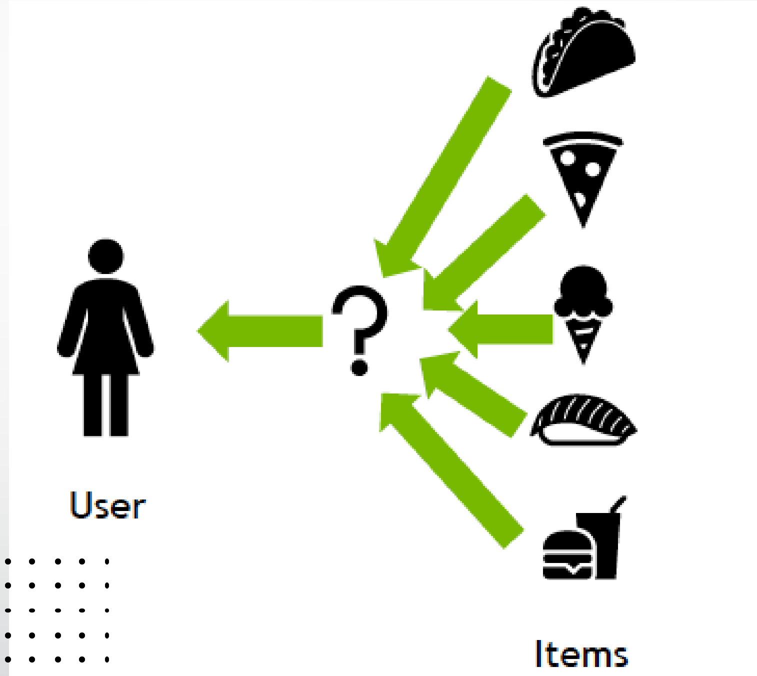
Recommendation System Using Machine Learning Techniques

ADITYA RANA
KARTHIKEY SHARMA

UNDER THE GUIDANCE OF
MR. DHANANJAY KUMAR

Introduction

Recommender systems are a way of suggesting like or similar items and ideas to a user's specific way of thinking.



Movie Recommendation System

- A movie recommendation system is a type of software that suggests movies to users based on their preferences, viewing history, and other factors.

OBJECTIVE

Personalization:

Increased Engagement:

Improved Customer Satisfaction:

Increased Revenue:

Reduced Information Overload:

Significant industrial & research interest in recommendation systems -

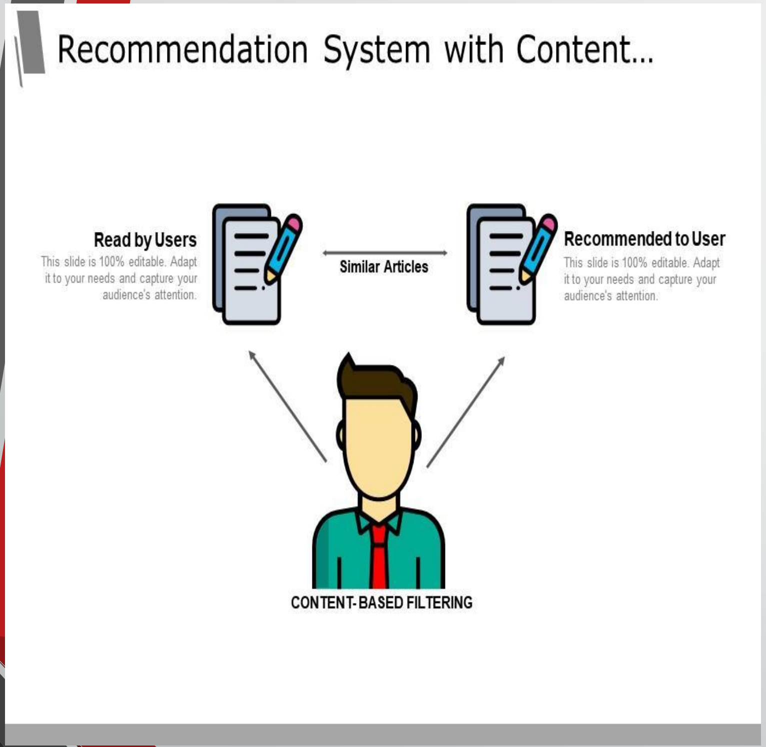
2 out of 3 watched hours come from recommendations

Increases its watch times by 50% per year

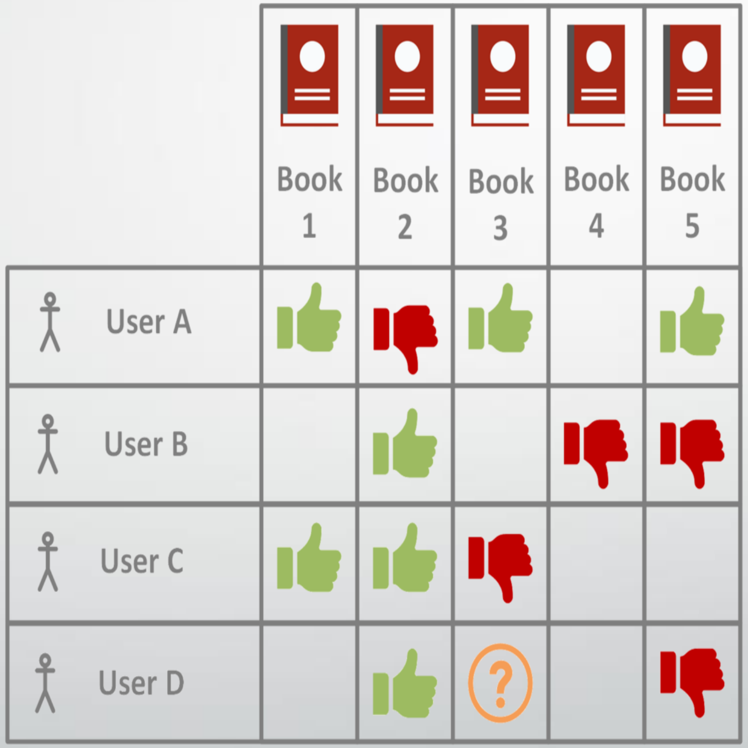
35% of all sales are generated by recommendations

Types Of Recommendation System:

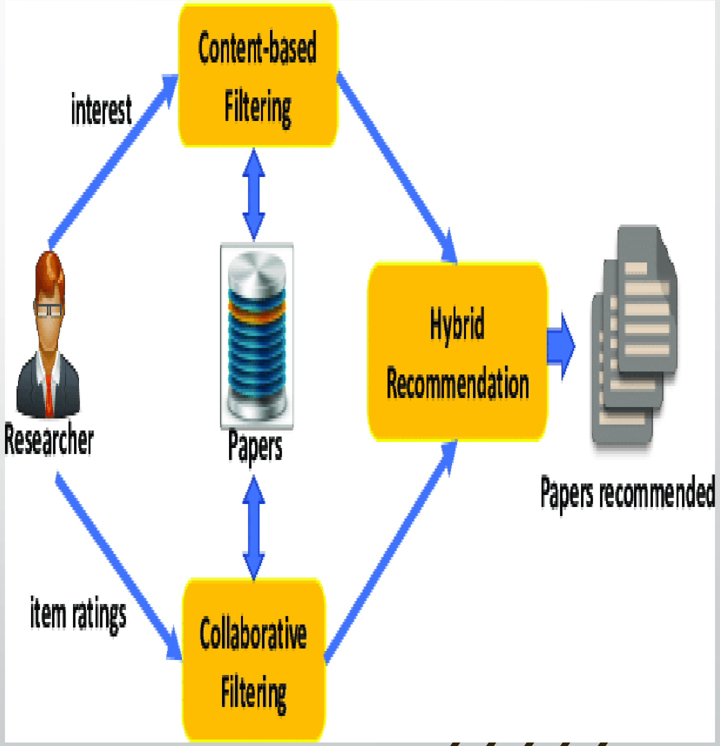
Content Based Recommendation



Collaborative Filtering



- Hybrid



Methodology:

```
ratings.head()
```

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931

```
movies.head()
```

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

Dataset:

Data Pre-processing:

We merge 2 datasets based on "movieId"

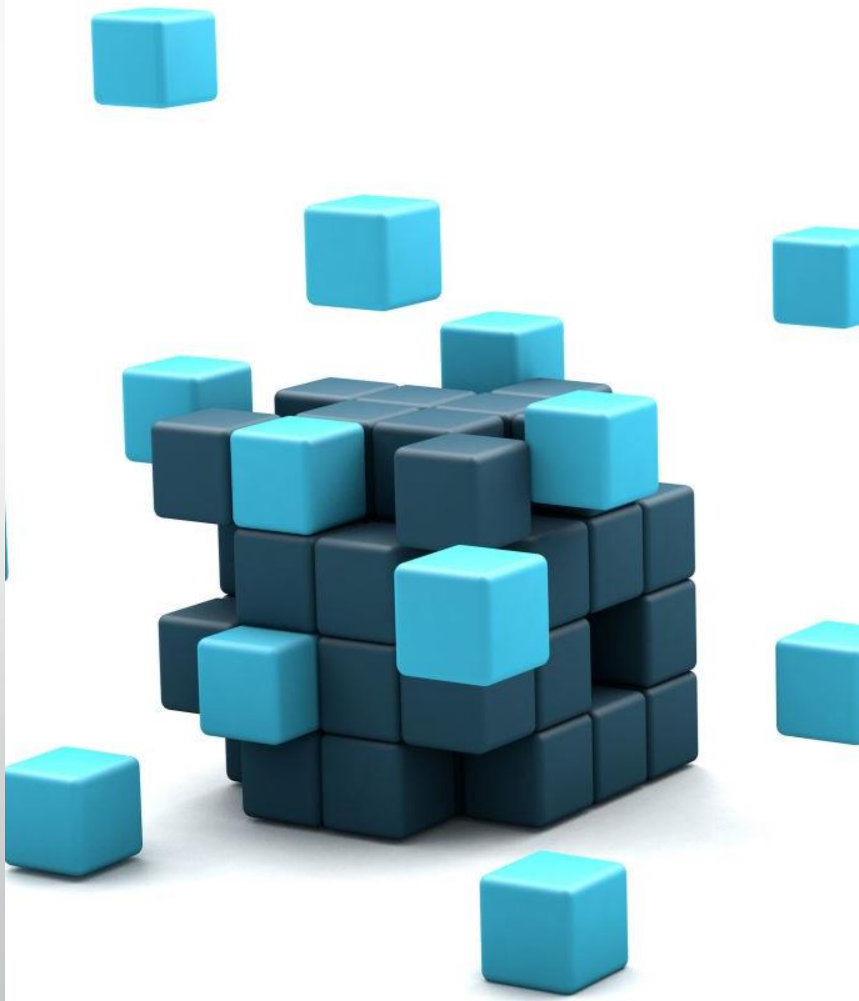
Dropping unnecessary columns .

	userId	movieId	rating	title
0	1	1	4.0	Toy Story (1995)
1	5	1	4.0	Toy Story (1995)
2	7	1	4.5	Toy Story (1995)
3	15	1	2.5	Toy Story (1995)
4	17	1	4.5	Toy Story (1995)

After Merging :

User Item Rating Matrix

The User Item Rating Matrix is a two-dimensional matrix that represents the ratings that users give to different items.



User-Item Rating Matrix:

'71 (2014)	'Hellboy': The Seeds of Creation (2004)	'Round Midnight (1986)	'Salem's Lot (2004)	'Til There Was You (1997)	'Tis the Season for Love (2015)	'burbs, The (1989)	'night Mother (1986)	(500) Days of Summer (2009)	*batteries not included (1987)	...	Zulu (2013)	[REC] (2007)	[REC]^2 (2009)	[REC]^3 3 Génesis (2012)	anohana: The Flower We Saw That Day - The Movie (2013)
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	...	0.0	1.0	1.0	1.0	0.0

'Rate' Dictionary:

```
rate
```

```
Output exceeds the size limit. Open the full output data in a text editor
{1: ['13th Warrior, The (1999)',
  '20 Dates (1998)',
  'Abyss, The (1989)',
  'Adventures of Robin Hood, The (1938)',
  'Alice in Wonderland (1951)',
  'Alien (1979)',
  'All Quiet on the Western Front (1930)',
  'American Beauty (1999)',
  'American History X (1998)',
  'American Tail, An (1986)',
  'Apocalypse Now (1979)',
  'Austin Powers: International Man of Mystery (1997)',
  'Back to the Future (1985)',
  'Back to the Future Part III (1990)',
  'Bambi (1942)',
  'Basic Instinct (1992)',
  'Batman (1989)',
  'Batman Returns (1992)',
  'Bedknobs and Broomsticks (1971)',
  'Beetlejuice (1988)',
  'Being John Malkovich (1999)',
  'Best Men (1997)',
  'Big (1988)',
  'Big Lebowski, The (1998)',
  'Big Trouble in Little China (1986)',
...
  'Shanghai Noon (2000)',
  'Shaolin Soccer (Siu lam juk kau) (2001)',
  'Shaolin Temple (Shao Lin si) (1976)',
  'Sharknado (2013)',
  ...]}
```

'Rows_indexes' Dictionary:

```
rows_indexes
```

```
Output exceeds the size limit. Open the full output data in a text editor
{1: [48,
  66,
  202,
  245,
  325,
  327,
  346,
  405,
  420,
  440,
  563,
  674,
  744,
  746,
  787,
  827,
  836,
  840,
  917,
  925,
  946,
  983,
  1025,
  1048,
  1062,
...
  7577,
  7580,
  7581,
  7585,
  ...]}
```

'not_rated' DICTIONARY

```
not_rated
```

Output exceeds the size limit. Open the full output data in a text editor

```
{1: ["''71 (2014)",
  "''Hellboy': The Seeds of Creation (2004)",
  "''Round Midnight (1986)",
  "''Salem's Lot (2004)",
  "''Til There Was You (1997)",
  "''Tis the Season for Love (2015)",
  "''burbs, The (1989)",
  "''night Mother (1986)",
  '(500) Days of Summer (2009)',
  '*batteries not included (1987)',
  '...All the Marbles (1981)',
  '...And Justice for All (1979)',
  '00 Schneider - Jagd auf Nihil Baxter (1994)',
  '1-900 (06) (1994)',
  '10 (1979)',
  '10 Cent Pistol (2015)',
  '10 Cloverfield Lane (2016)',
  '10 Items or Less (2006)',
  '10 Things I Hate About You (1999)',
  '10 Years (2011)',
  '10,000 BC (2008)',
  '100 Girls (2000)',
  '100 Streets (2016)',
  '101 Dalmatians (1996)',
  '101 Dalmatians (One Hundred and One Dalmatians) (1961)',
...
  "Blackadder's Christmas Carol (1988)",
  "Blackbeard's Ghost (1968)",
  'Blackboard Jungle (1955)',
  'Blackfish (2013)',
  ...]}
```

Nearest Neighbour Recommender:

- We will be using item-item collaborative filtering with nearest neighbor and cosine similarity

$$\text{Similarity}(p,q) = \cos\theta = \frac{p \cdot q}{\|p\|\|q\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} {p_i}^2}\sqrt{\sum_{i=1}^{n} {q_i}^2}}$$

EXPERIMENTS

Manhattan Distance

```
error_rate = rmse(predictions , ground_truth)

print("RMSE: {:.5f}".format(error_rate))
RMSE: 1.06580
```

Euclidean Distance

```
error_rate = rmse(predictions , ground_truth)
print("RMSE: {:.5f}".format(error_rate))
RMSE: 1.02687
```

Minkowski

```
error_rate = rmse(predictions , ground_truth)
print("RMSE: {:.5f}".format(error_rate))
RMSE: 1.02687
```

Chebyschev

```
error_rate = rmse(predictions , ground_truth)
print("RMSE: {:.5f}".format(error_rate))
RMSE: 0.98084
```

Cosine

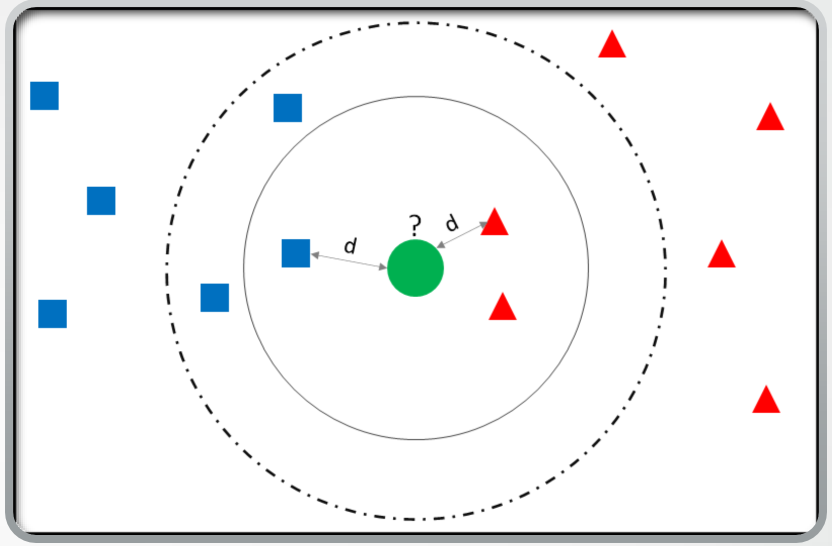
```
error_rate = rmse(predictions , ground_truth)
print("RMSE: {:.5f}".format(error_rate))
RMSE: 0.97137
```

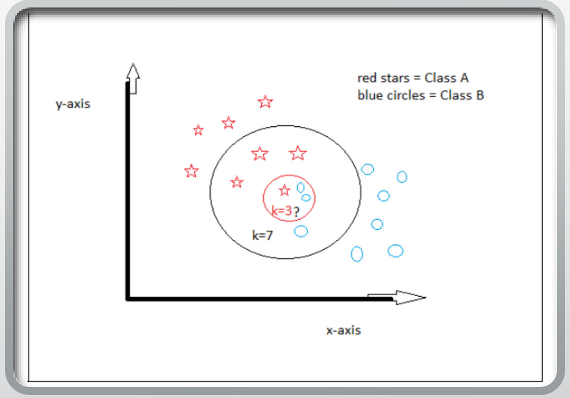
Calculating Scores

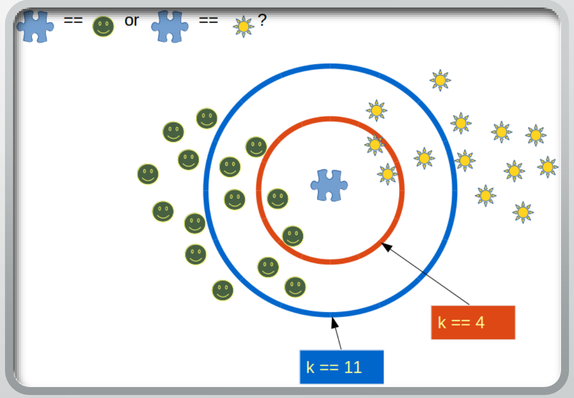
Transposed pivot table

Fit function

item_distances
item_indices







Item_distances

```
item_distances
```

```
array([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
        0.00000000e+00],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 2.92893219e-01,
        2.92893219e-01],
       [2.22044605e-16, 2.92893219e-01, 2.92893219e-01, 2.92893219e-01,
        2.92893219e-01],
       ...,
       [1.11022302e-16, 3.29179607e-01, 3.67544468e-01, 3.67544468e-01,
        3.67544468e-01],
       [0.00000000e+00, 4.43651360e-01, 4.89357079e-01, 4.91573686e-01,
        5.09709662e-01],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
        2.92893219e-01]])
```

Item_indices

```
item_indices
```

```
array([[7085, 8860, 9602, 9604, 1015],
       [7888,    1, 5773,    2, 3913],
       [   2,  324,  659,  656,  278],
       ...,
       [9716, 2179, 4889, 9682, 9044],
       [9717, 5753, 7894, 3548,  742],
       [3144, 8255, 9718, 5990, 1922]], dtype=int64)
```

```
items_dic
```

```
Output exceeds the size limit. Open the full output data in a text editor
{"'71 (2014)": ['Resolution (2012)',
  'Tournament, The (2009)',
  'Wrong (2012)',
  'Wrong Cops (2013)',
  'Beyond Re-Animator (2003)'],
 "'Hellboy': The Seeds of Creation (2004)": ['Space Battleship Yamato (2010)',
  "'Hellboy': The Seeds of Creation (2004)",
  'Monsters (2010)',
  "'Round Midnight (1986)",
  'Hidden Fortress, The (Kakushi-toride no san-akunin) (1958)'],
 "'Round Midnight (1986)": ["'Round Midnight (1986)",
  'Alice in Wonderland (1933)',
  'Attack of the 50 Foot Woman (1958)',
  'Atomic Submarine, The (1959)',
  'Agony and the Ecstasy, The (1965)'],
 "'Salem's Lot (2004)": ["'Salem's Lot (2004)",
  'All This, and Heaven Too (1940)',
  "'Til There Was You (1997)",
  'Absence of Malice (1981)',
  '84 Charing Cross Road (1987)'],
 "'Til There Was You (1997)": ["'Til There Was You (1997)",
  "'Salem's Lot (2004)",
  'Man and a Woman, A (Un homme et une femme) (1966)',
  'Best Seller (1987)',
  'All This, and Heaven Too (1940)'],
...
  "What's Your Number? (2011)",
  'Outsourced (2006)',
  'If These Walls Could Talk (1996)',
  'Gloomy Sunday (Ein Lied von Liebe und Tod) (1999)'],
 ...}
```

Final Output

```
We Recommend to view these movies too:

 Heart Condition (1990) with similarity: 0.8165
 Rare Birds (2001) with similarity: 0.8165
 Deuces Wild (2002) with similarity: 0.8165
 Who Is Cletis Tout? (2001) with similarity: 0.8165
 Jesus' Son (1999) with similarity: 0.8165
 Spy Who Loved Me, The (1977) with similarity: 0.7778
 Back to the Future Part II (1989) with similarity: 0.7657
 Batman Forever (1995) with similarity: 0.7395
 Presidio, The (1988) with similarity: 0.7143
 Hardball (2001) with similarity: 0.7071
 Gunga Din (1939) with similarity: 0.7071
 West Beirut (West Beyrouth) (1998) with similarity: 0.7071
 Golden Bowl, The (2000) with similarity: 0.7071
 Crimson Tide (1995) with similarity: 0.6952
 On Her Majesty's Secret Service (1969) with similarity: 0.6
 Pretty Woman (1990) with similarity: 0.6885
 Die Hard (1988) with similarity: 0.6878
 True Lies (1994) with similarity: 0.6841
 Cliffhanger (1993) with similarity: 0.6831
 Batman & Robin (1997) with similarity: 0.6773
 Terminator 2: Judgment Day (1991) with similarity: 0.6772
 For Your Eyes Only (1981) with similarity: 0.6682
 Armed and Dangerous (1986) with similarity: 0.6682
 Firewalker (1986) with similarity: 0.6667
 Romeo Is Bleeding (1993) with similarity: 0.6667
 Action Jackson (1988) with similarity: 0.6667
 Iron Eagle II (1988) with similarity: 0.6667
 Lion King, The (1994) with similarity: 0.6608
 Memento (2000) with similarity: 0.6607
 X2: X-Men United (2003) with similarity: 0.6565
```

RMSE Calculation

Dataset

Training 80%

Testing 20%

Testing Data

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

- estimate rating of movie 1 by user 5

Stanford

Pre Computation

Transpose Training Data

Calculate Cosine Similiarity Score

Normalize the score

Subtract 1 from score (Closeness)

$$\boxed{r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \; r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}}$$

N(i;x)…set of items rated by ***x*** *and* similar to ***i***
s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***

Predicted Rating

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

RMSE Calculation

RMSE Experiments

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,5)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.8998185880847233
```

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,7)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.8307663570262754
```

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,10)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.7644090537483834
```

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,12)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.7247805970127025
```

RMSE Experiments

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,14)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.8481775360991225
```

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,16)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.9896272862551964
```

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,18)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.7514188784854684
```

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,20)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.7887368904291836
```

RMSE Experiments

```
import random
mean_rmse_value=0
total=0
for i in range(10):
    current_user=random.randrange(0,121)
    mean_rmse_value+=rmse_value(current_user,user_movie_matrix,30)
    total+=1
mean_rmse_value/=total
print(mean_rmse_value)
```

```
0.8769470794952173
```

Problems and their Solutions

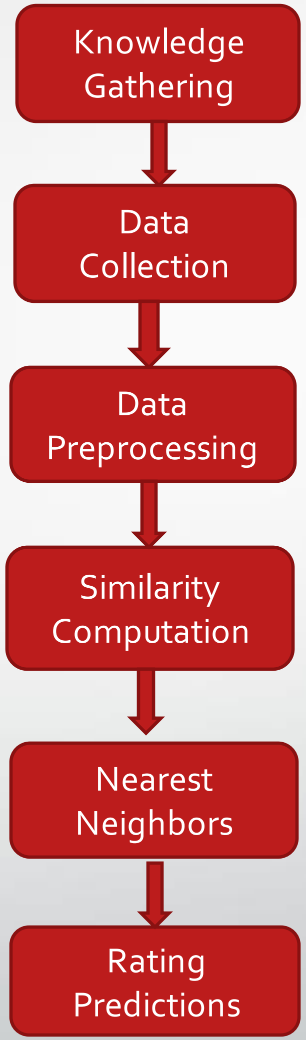
User Cold Start: There needs to be enough other users already in the system to find a match.

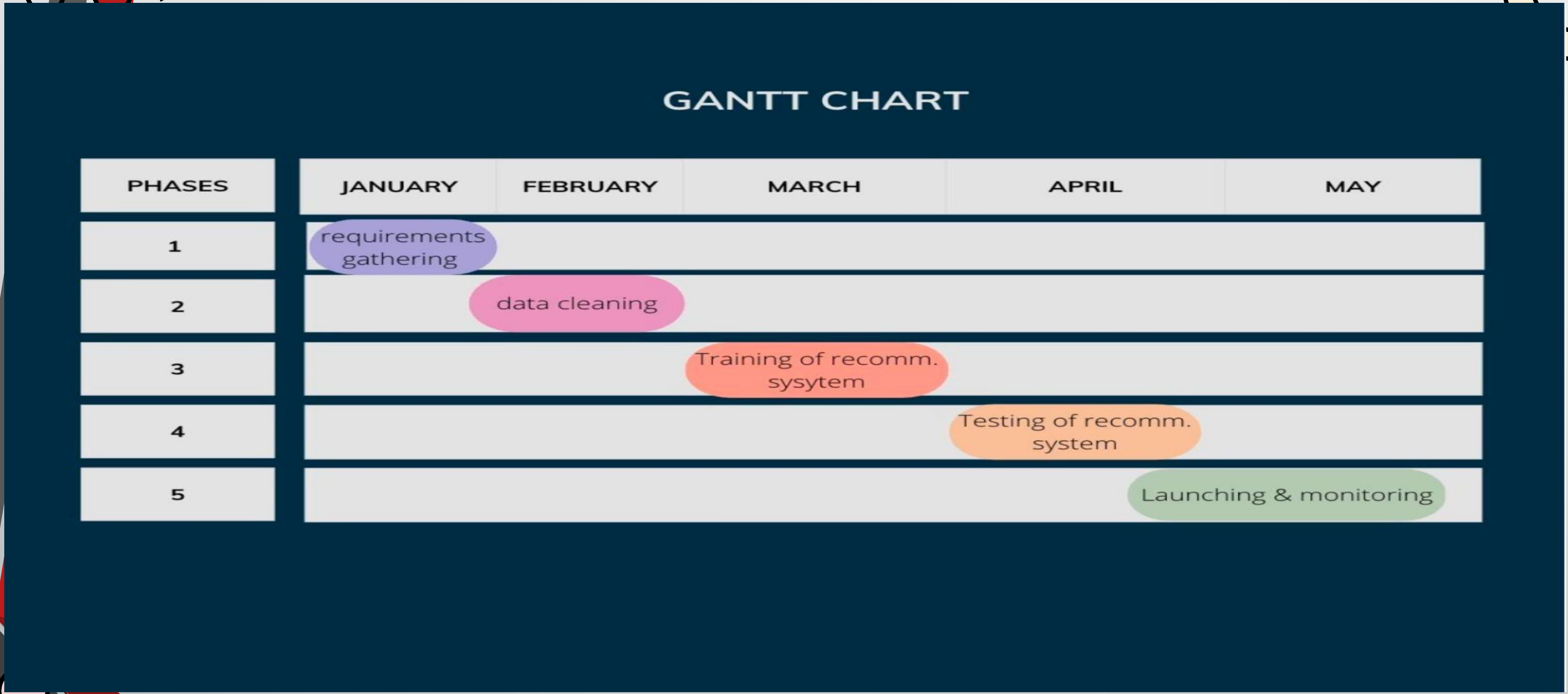
Data sparsity: Not all users have rated all items. By focusing on the similarity of items rather than users, the system can provide recommendations even when data is sparse.

Diversity: Item-based collaborative filtering can help provide diverse recommendations that cover a wide range of user preferences.

Popularity Bias: Cannot recommend items to someone with unique tastes. Tends to recommend popular items.

WORK FLOW


Knowledge Gathering
Data Collection
Data Preprocessing
Similarity Computation
Nearest Neighbors
Rating Predictions


GANTT CHART
PHASES
JANUARY
FEBRUARY
MARCH
APRIL
MAY
1
requirements gathering
2
data cleaning
3
Training of recomm. sysytem
4
Testing of recomm. system
5
Launching & monitoring

Literature Review:

"Item-Based Collaborative Filtering Recommendation Algorithms" by Sarwar et al.

Research Question	Methodology	Key Findings
Effectiveness of item-based collaborative filtering	Experimental study	Item-based algorithm is more accurate and scalable than user-based algorithm.
Advantages of item-based collaborative filtering	Comparative analysis	More resilient to sparsity, requires less resources, provides personalized recommendations.
How item-based collaborative filtering works	Technical description	Uses similarity between items based on user ratings to generate recommendations.
Limitations of item-based collaborative filtering	Critical analysis	Relies heavily on user ratings, may suffer from cold-start problem, tends to recommend popular items.
Improving item-based collaborative filtering	Future directions	Incorporating contextual information and advanced machine learning techniques.

Sharma, Ritu, Dinesh Gopalani, and Yogesh Meena. "Collaborative filtering-based recommender system: Approaches and research challenges."

Research question	Methodology	Key findings
Approaches and challenges in collaborative filtering-	Literature review	Collaborative filtering is widely used but faces challenges such as data sparsity, cold start problem, and scalability issues.
Collaborative filtering using matrix factorization	Technical description	Matrix factorization aims to factorize the rating matrix into low-dimensional matrices representing user and item factors
Collaborative filtering using neighborhood-	Technical description	Neighborhood-based methods compute similarity between users or items based on rating patterns.
Hybrid approaches for collaborative filtering	Comparative analysis	Hybrid methods combine multiple techniques to improve accuracy.

Conclusion:

In summary, Collaborative Filtering is a widely used technique in recommender systems for generating personalized recommendations by finding similar user/item pairs.

However, it faces challenges such as user and item cold-start problems, sparsity, and scalability issues. To overcome these challenges, researchers have explored combining CF with other techniques and incorporating factors such as trust, cross-domain information, context, and time-variant features.

However, there is still a lot of room for improvement and further research in this area .

We can also explore new methods to combine collaborative filtering with other approaches to resolve scalability and cold start issues. CF based approaches can be enhanced by incorporating trust, cross-domain information, context and time-variant features.

Bibliography

- Sharma, Ritu, Dinesh Gopalani, and Yogesh Meena. "Collaborative filtering-based recommender system: Approaches and research challenges." In 2017 3rd international conference on computational intelligence & communication technology (CICT), pp. 1-6. IEEE, 2017 Vedavathi, N., and R. Suhas Bharadwaj.
- "Deep Flamingo Search and Reinforcement Learning Based Recommendation System for E-Learning Platform using Social Media." Procedia Computer Science 215 (2022): 192-201. Sarwar, Badrul, George Karypis, Joseph Konstan, and John Riedl.
- "Item-based collaborative filtering recommendation algorithms." In Proceedings of the 10th international conference on World Wide Web, pp. 285-295. 2001.

Special Thanks to Mr. Dhananjay Kumar

Karthikey Sharma

209302122

Aditya Rana

209302339