MOVIE RECOMMENDATION SYSTEM

FINAL PROJECT | BIG DATA
INTRUCTOR: DO TRONG HOP

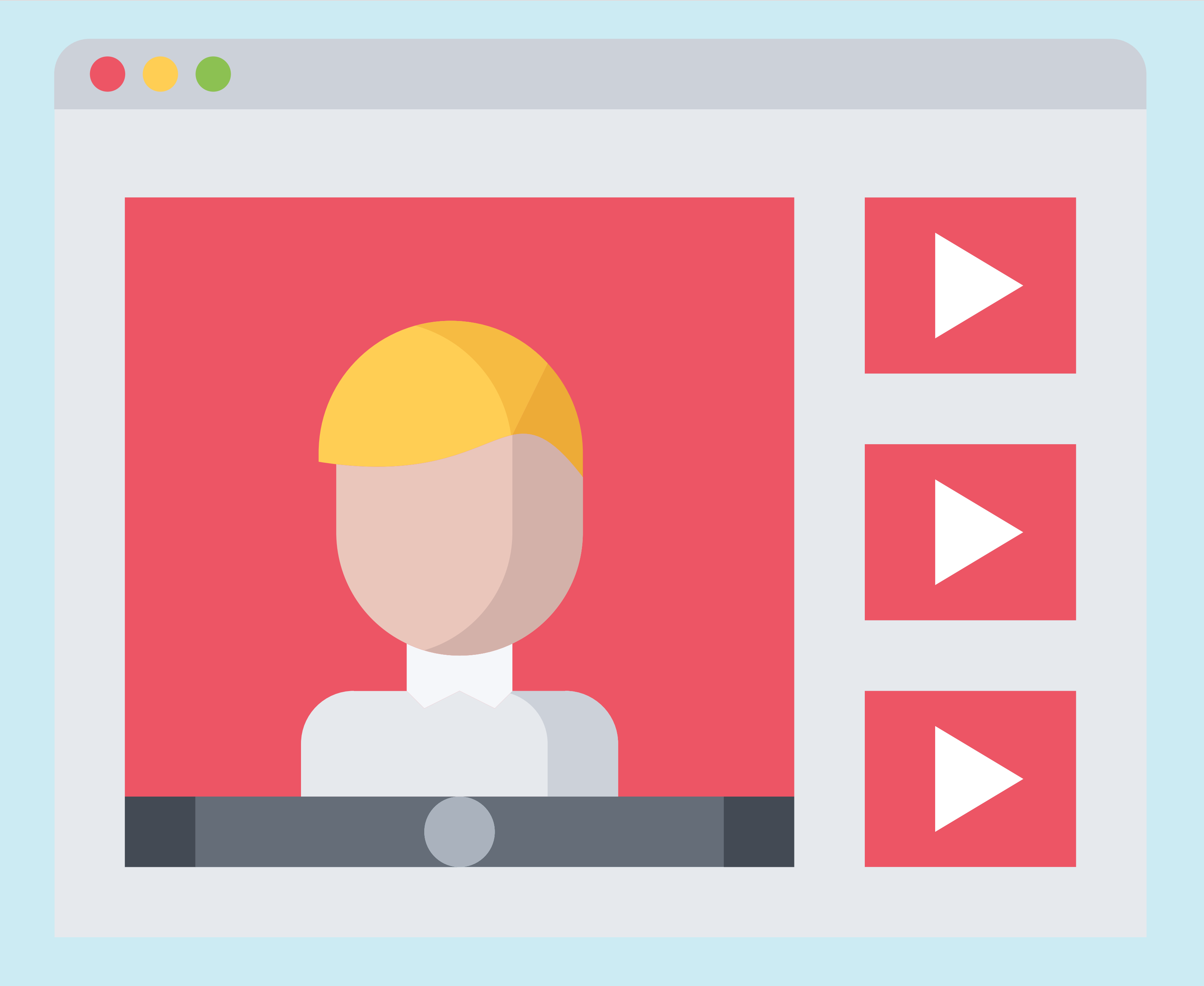
GROUP 7:
NGUYEN THI KHANH HA - 18520692
DINH XUAN HUNG - 18520791
LAM LE DINH KHANG - 18520885

Contents

01

INTRODUCTION

Why do we need recommendation system?



WHY DO WE NEED RECOMMENDER SYSTEM?

DIGITAL INFORMATION EXPLOSIVE

INFORMATION OVERLOAD

THE NUMBER OF CHOICES IS OVERWHELMING

DELIVER RELEVANT INFORMATION

ALLEVIATE THE PROBLEM OF INFORMATION OVERLOAD

RECOMMENDATION SYSTEM

Recommender system was defined as a means of assisting and augmenting the social process of using recommendations of others to make choices when there is no sufficient personal knowledge or experience of the alternatives

Currently, such recommendation system are used in multiple forms on major websites, such as: online movie rentals, streaming services, news websites, online bookstores and many other places online.

- ✓ Build customer trust and loyalty
- ✓ Improve decision making
- ✓ Enhance revenues
- ✓ Reduce transaction costs

MOVIE RECOMMENDATION SYSTEM

With the desire to contribute to helping entrepreneurs win the trust and interest of customers, we proceed to build a recommendation system in order to suggest movies to customers.

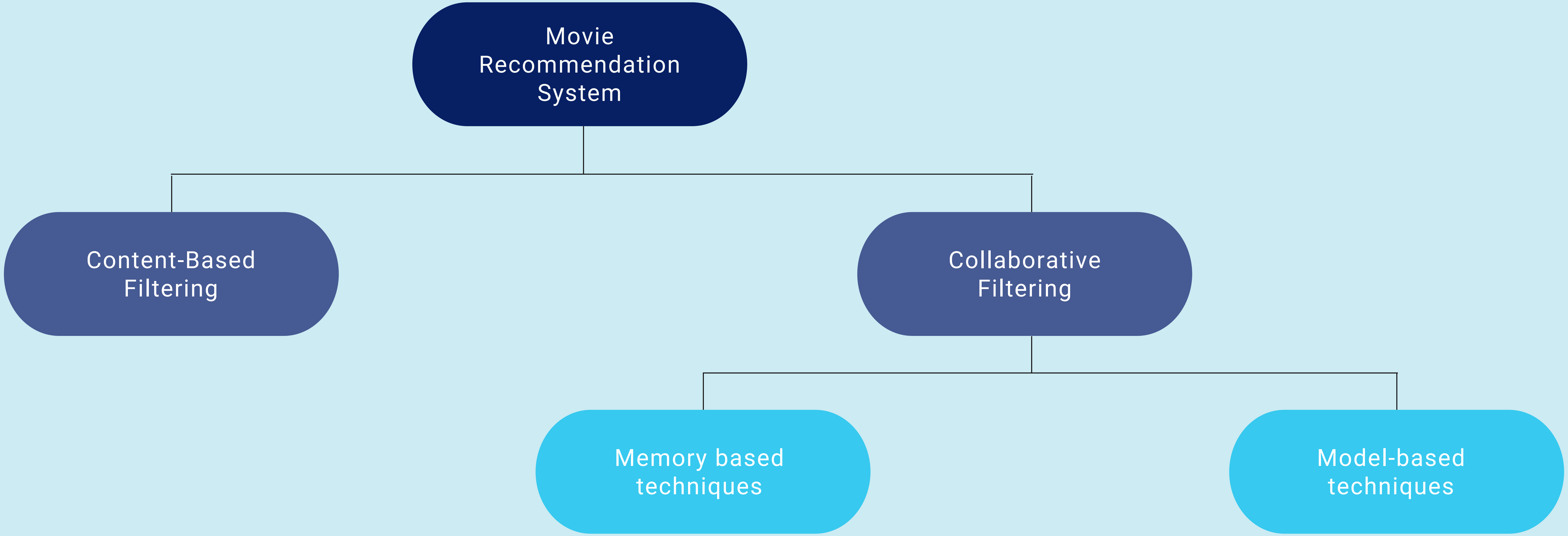
- Can we recommend a movie based on the user's previous preference and rating?
- Can we recommend the movies based on the given movie from the user?
- Given the similar types of users can we recommend the movies rated by one user to the other similar users?
- Can we recommend a movie, given there’s not enough data about the movie such as cast, release year, genres, ...?
- Can we compare different algorithms for the recommender systems?

02

MOVIE RECOMMENDATION SYSTEM

Types of movies recommendation system

MOVIE RECOMMENDATION SYSTEM TYPES



Movie Recommendation System

Content-Based Filtering

Collaborative Filtering

Memory based techniques

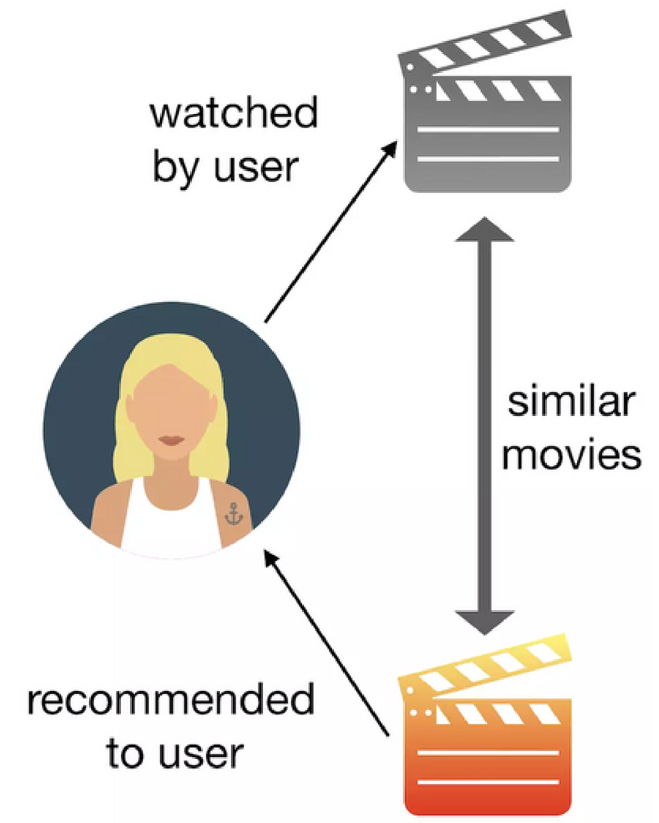
Model-based techniques

Movie Recommendation System

Content-Based Filtering

It recommends other movies which are similar to that selected movie, in terms of features. Thus it needs a lot information about movie.

- Σ Cosine Similarity
- Σ Jaccard Similarity

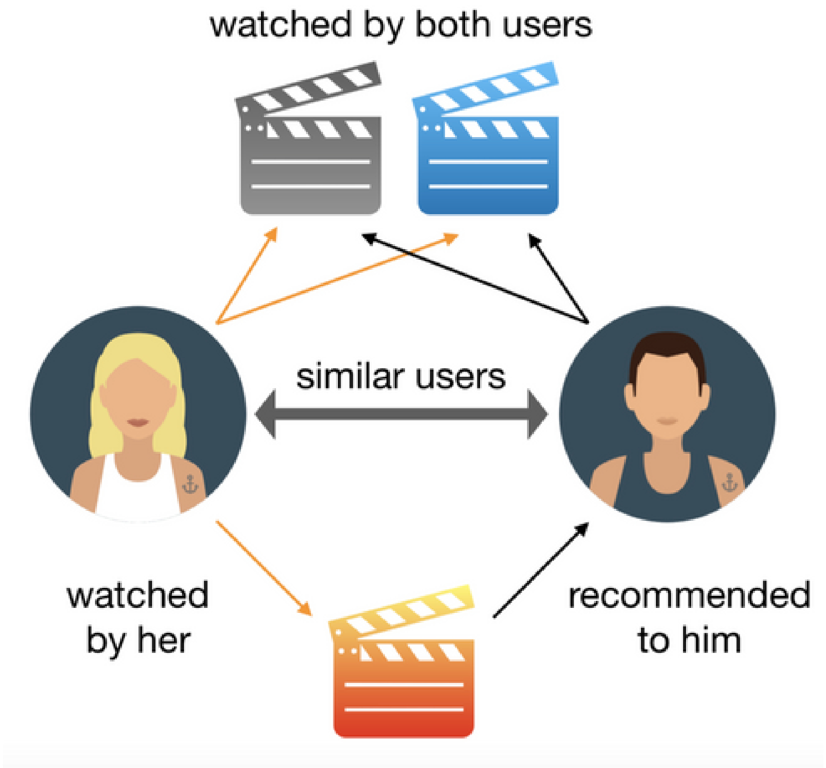


- ✗ Different products do not get much exposure to the user.
- ✗ Businesses cannot be expanded as the user does not try different types of products.

Collaborative Filtering

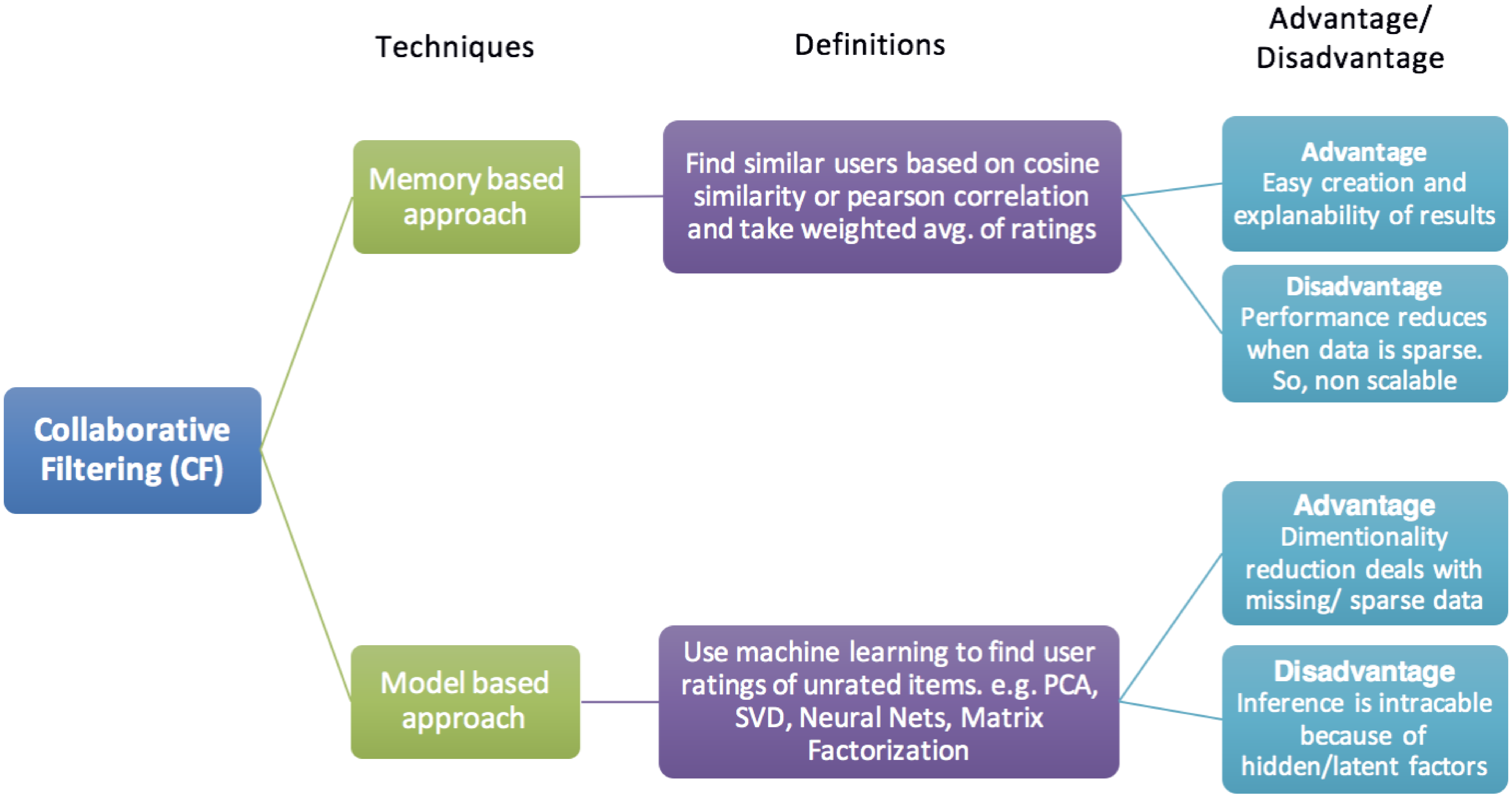
It recommends other movies which are rated highly by similar users. Thus it finds similarity between users.

- Σ K-Nearest Neighbor's (KNN)
- Σ Singular value decomposition(SVD)
- Σ Alternating Least Squares (ALS).



- ✗ People are fickle-minded
- ✗ There are many more users than items
- ✗ This algorithm is very susceptible to shilling attacks

02.2 COLLABORATIVE FILTERING



Techniques

Definitions

Advantage/ Disadvantage

Collaborative Filtering (CF)

Memory based approach

Find similar users based on cosine similarity or pearson correlation and take weighted avg. of ratings

Advantage
Easy creation and explanability of results

Disadvantage
Performance reduces when data is sparse. So, non scalable

Model based approach

Use machine learning to find user ratings of unrated items. e.g. PCA, SVD, Neural Nets, Matrix Factorization

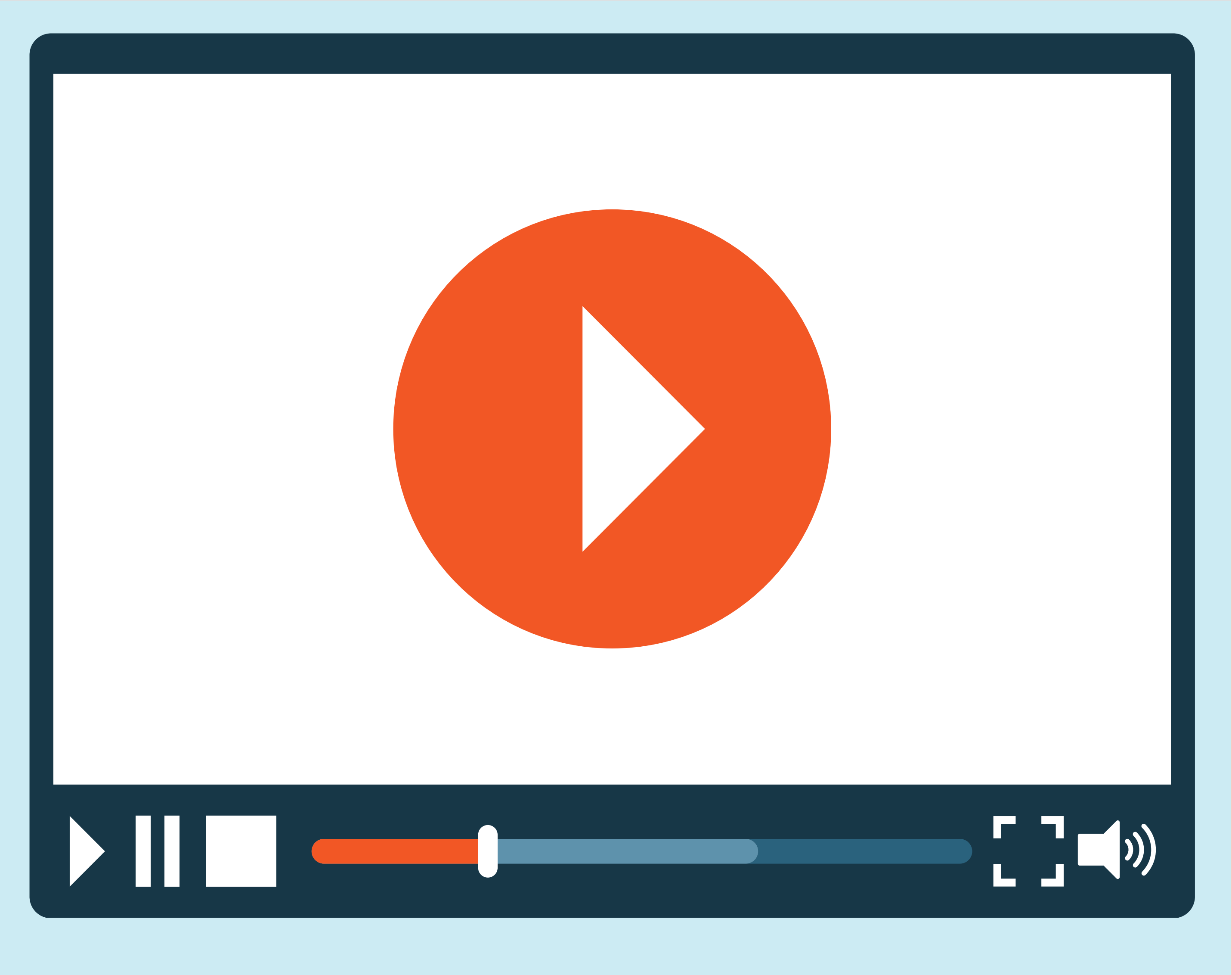
Advantage
Dimentionality reduction deals with missing/ sparse data

Disadvantage
Inference is intracable because of hidden/latent factors

03

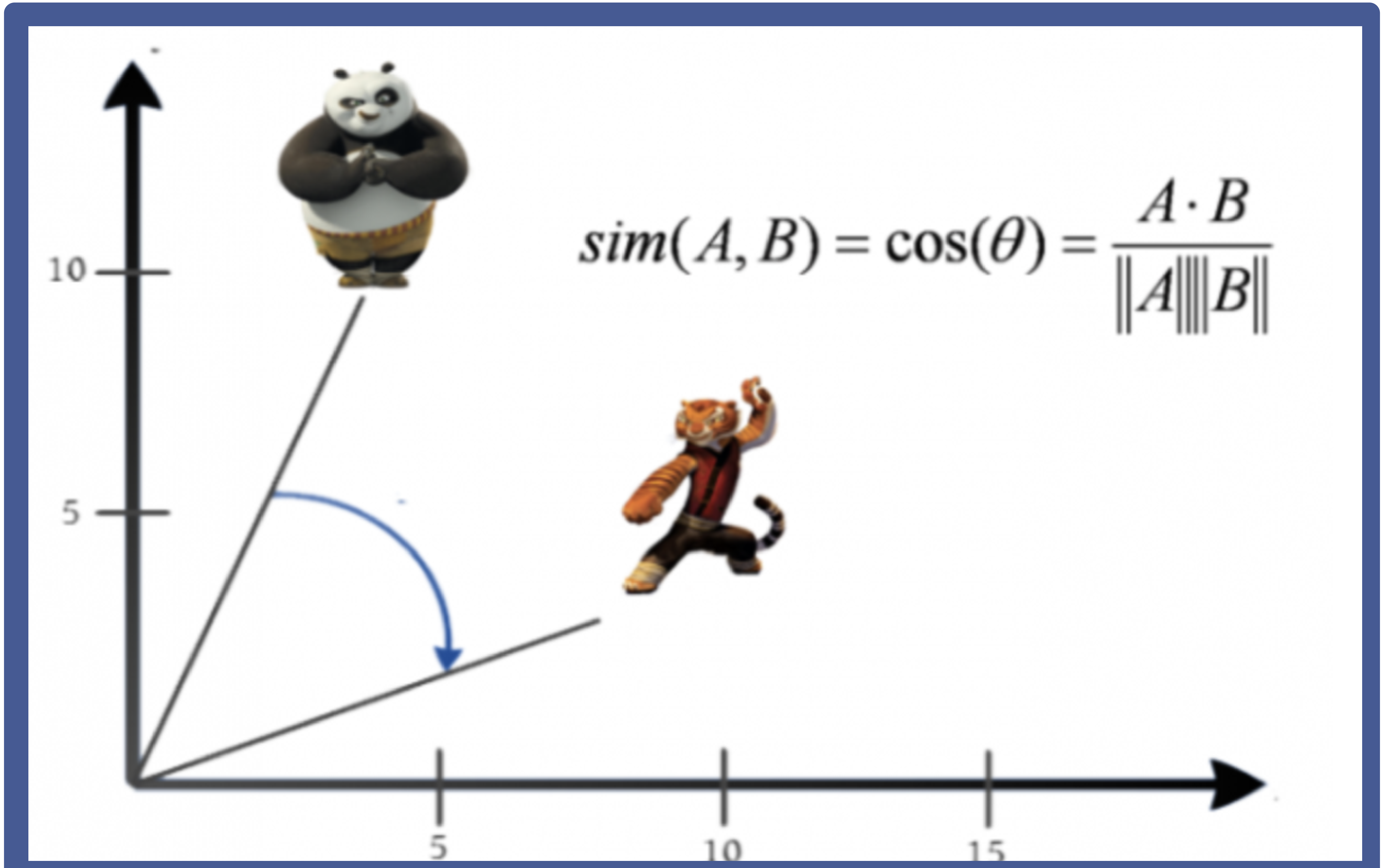
IMPLEMENTATION

Implementation Movies Recommendation System



03.1

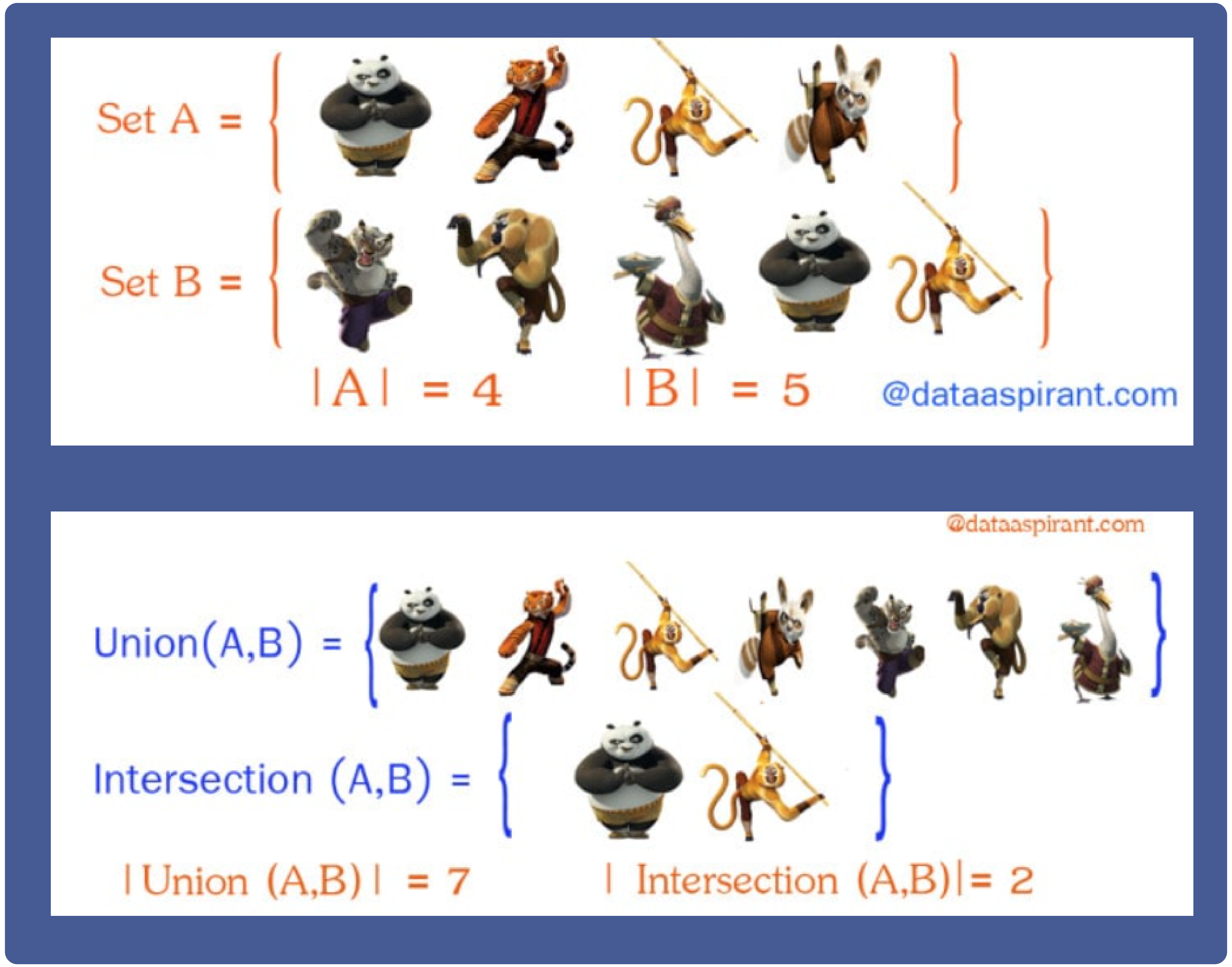
COSINE SIMILARITY

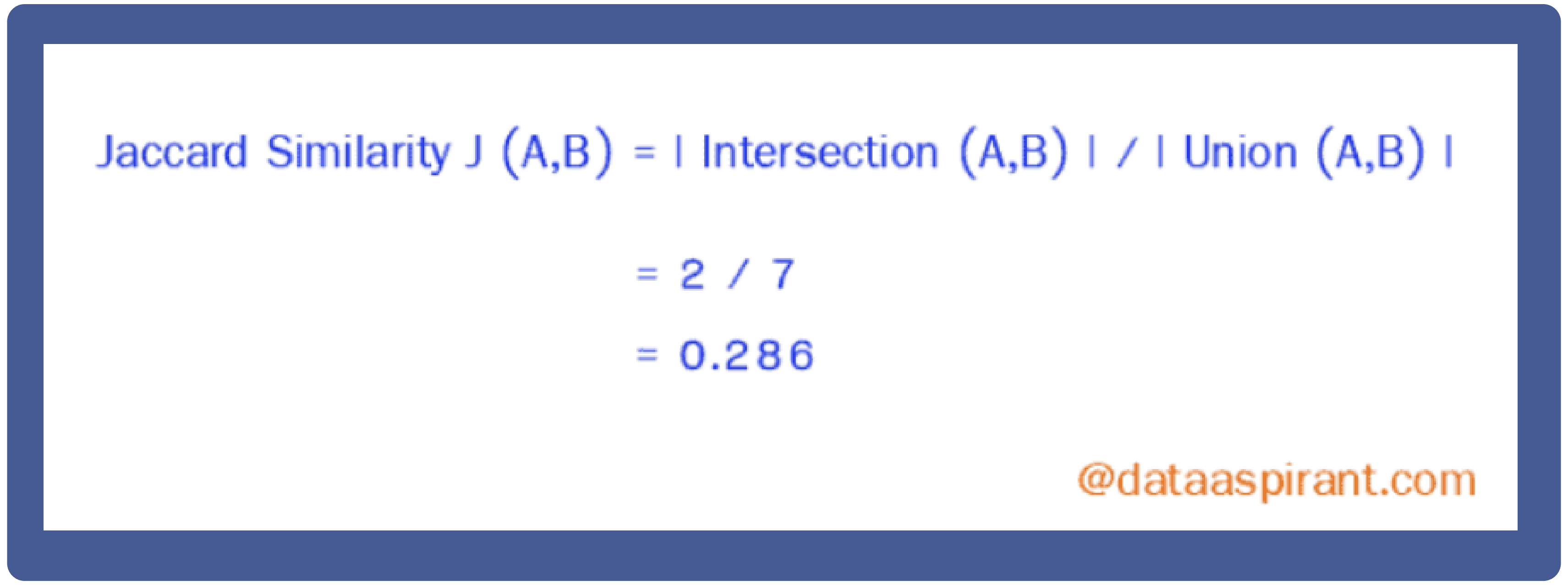


$$sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

03.2

JACCARD SIMILARITY





Jaccard Similarity J (A,B) = | Intersection (A,B) | / | Union (A,B) |

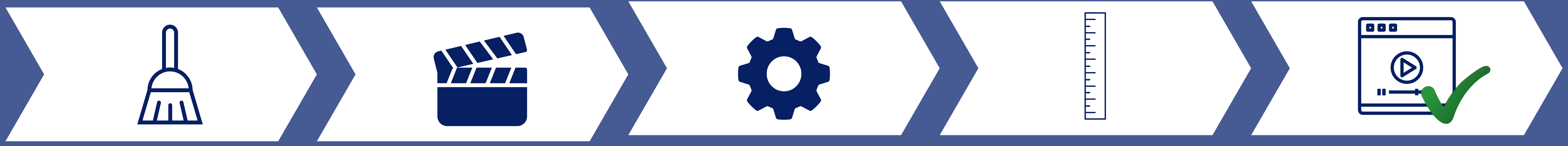
= 2 / 7

= 0.286

@dataaspirant.com

03.3

KNN ALGORITHM



CLEAN AND RESHAPING THE DATA

Input MOVIE NAME

KNN MODEL

Calculating EUCLIDIAN DISTANCES

Recommending the K movies

LIMITATIONS OF KNN

✗ **Not Scalable**

Lack of the ability to scale to much larger sets of data when more and more users and movies are added into our database.

✗ **Popularity Bias**

A system recommending the movies with the most interactions without any personalization.

✗ **Cold Start problem**

When movies added to the catalogue have either none or very little interactions while recommenders rely on the movie’s interactions to make recommendations.

03.4

ALS ALGORITHM

Movie Data Set
Step 1: Take the movie dataset

Item

User	W	X	Y	Z
A		4.5	2.0	
B	4.0		3.5	
C		5.0		2.0
D		3.5	4.0	1.0

Rating Matrix
Step 2: Convert in into rating matrix

Step 3: Decompose into lower dimension matrix

Constant

A	1.2	0.8
B	1.4	0.9
C	1.5	1.0
D	1.2	0.8

✖

W	X	Y	Z
1.1	1.6	2	0.5
1.5	0.5	1.8	0.6

Step 4: Take a matrix constant and tune another matrix

Run until convergence

A	1.2	0.8
B	1.4	0.9
C	1.5	1.0
D	1.2	0.8

✖

Constant

W	X	Y	Z
1.1	1.6	2	0.5
1.5	0.5	1.8	0.6

Step 5: Take the new updated matrix as constant and update other

Find product of two matrices to get predicted rating matrix

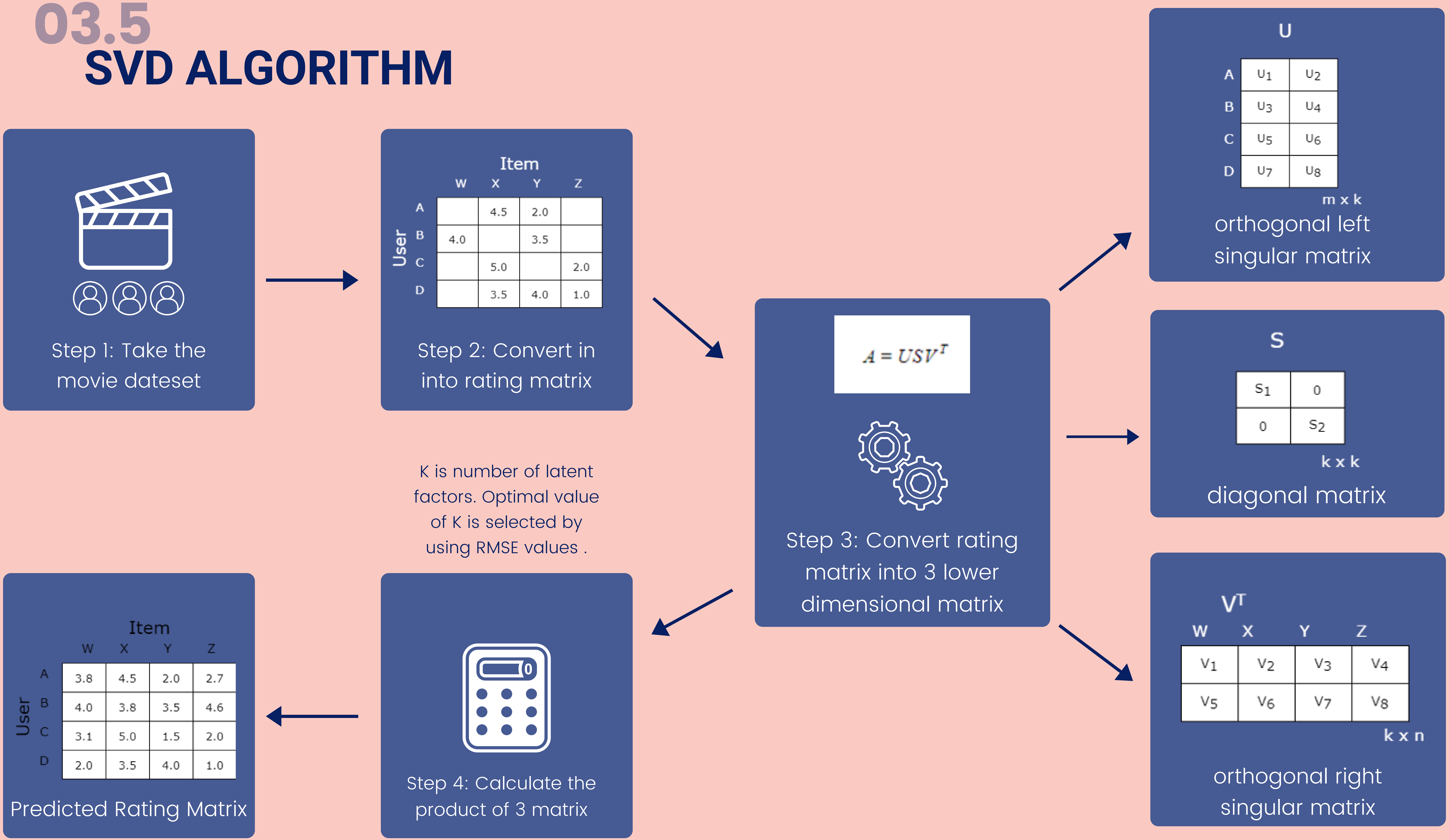
Item

User	W	X	Y	Z
A	4.6	4.5	2.0	4.6
B	3.8	5.0	3.5	3.4
C	4.1	5.0	2.6	2.0
D	2.8	3.5	4.0	1.0

Predicted rating matrix

03.5

SVD ALGORITHM



Step 1: Take the movie dateset

Item

User

	W	X	Y	Z
A		4.5	2.0	
B	4.0		3.5	
C		5.0		2.0
D		3.5	4.0	1.0

Step 2: Convert in into rating matrix

$$A = USV^T$$

Step 3: Convert rating matrix into 3 lower dimensional matrix

U

A	u_1	u_2
B	u_3	u_4
C	u_5	u_6
D	u_7	u_8

m x k

orthogonal left singular matrix

S

s_1	0
0	s_2

k x k

diagonal matrix

V^T

W	X	Y	Z
v_1	v_2	v_3	v_4
v_5	v_6	v_7	v_8

k x n

orthogonal right singular matrix

K is number of latent factors. Optimal value of K is selected by using RMSE values .

Step 4: Calculate the product of 3 matrix

Item

User

	W	X	Y	Z
A	3.8	4.5	2.0	2.7
B	4.0	3.8	3.5	4.6
C	3.1	5.0	1.5	2.0
D	2.0	3.5	4.0	1.0

Predicted Rating Matrix

04

Dataset

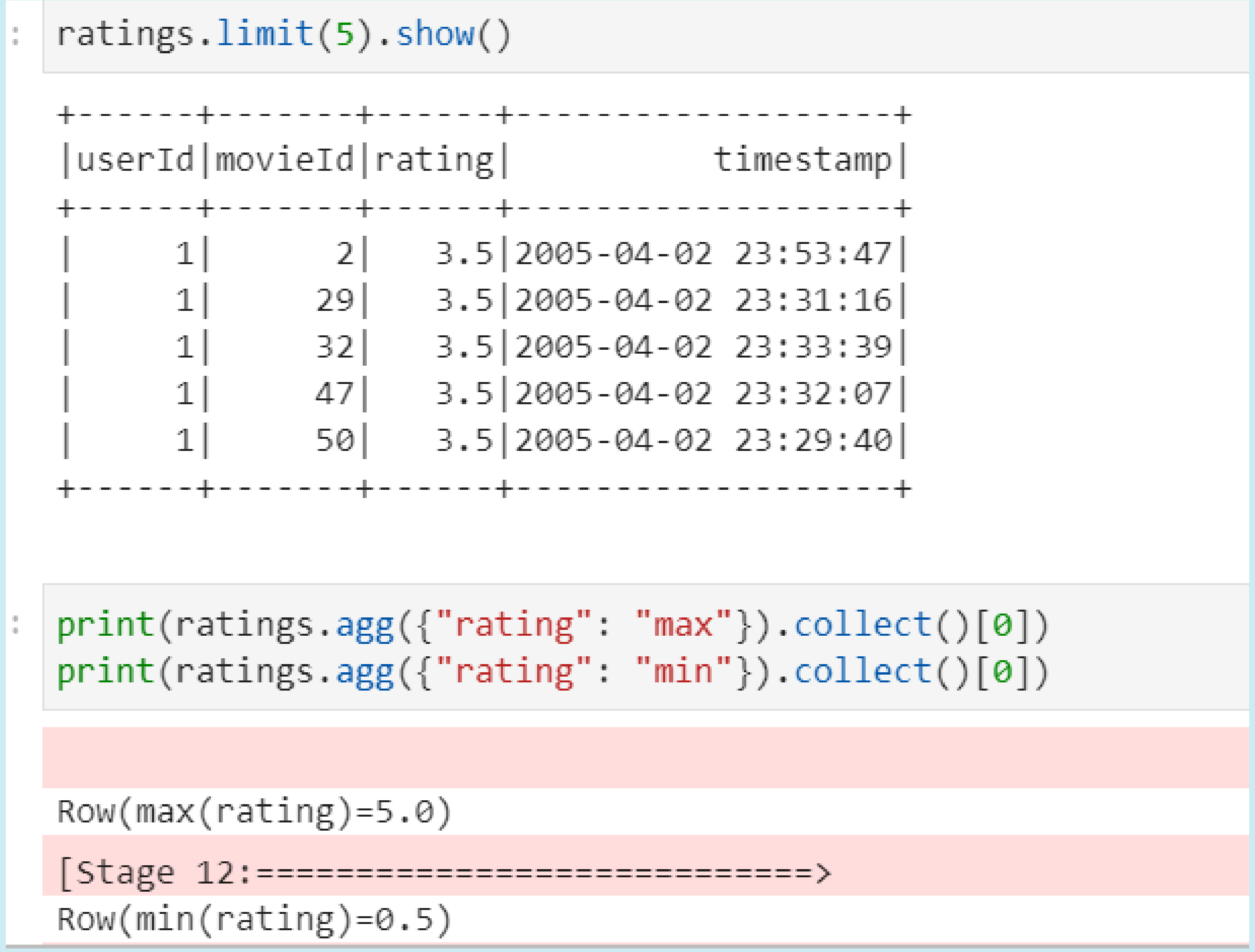
MOVIELENS 20M

https://www.kaggle.com/grouplens/movielens-20m-dataset

```
# Data is downloaded from https://www.kaggle.com/bandikarthik/movie-recommendation-system
movies = spark.read.csv('../MovieLens/movie.csv', header=True, inferSchema=True)
ratings = spark.read.csv('../MovieLens/rating.csv',  header=True, inferSchema=True)
```

```
movies.limit(5).show()
```

```
+-------+--------------------+--------------------+
|movieId|               title|              genres|
+-------+--------------------+--------------------+
|      1|    Toy Story (1995)|Adventure|Animati...|
|      2|      Jumanji (1995)|Adventure|Childre...|
|      3|Grumpier Old Men ...|      Comedy|Romance|
|      4|Waiting to Exhale...|Comedy|Drama|Romance|
|      5|Father of the Bri...|              Comedy|
+-------+--------------------+--------------------+
```



```
ratings.limit(5).show()
```

```
+------+-------+------+-------------------+
|userId|movieId|rating|          timestamp|
+------+-------+------+-------------------+
|     1|      2|   3.5|2005-04-02 23:53:47|
|     1|     29|   3.5|2005-04-02 23:31:16|
|     1|     32|   3.5|2005-04-02 23:33:39|
|     1|     47|   3.5|2005-04-02 23:32:07|
|     1|     50|   3.5|2005-04-02 23:29:40|
+------+-------+------+-------------------+
```

```
print(ratings.agg({"rating": "max"}).collect()[0])
print(ratings.agg({"rating": "min"}).collect()[0])
```

```
Row(max(rating)=5.0)
[Stage 12:============================>
Row(min(rating)=0.5)
```

05

RESULT

Result of implementation

05.1

CONSINE SIMILARITY ALGORITHM

```
cosine_recomm=cosine_dist.join(movies_df, movies_df['movieId']==cosine_dist.movieId)\
.sort('cosine_sim',ascending=False).take(10)
cosine_recomm_df = spark.createDataFrame(cosine_recomm)
cosine_recomm_df.join(movies, on="movieId")
```

movieId	cosine_sim	movieId	genre	title	genres
105835	1.0000000000000002	105835	[comedy, drama, t...	Double, The (2013)	Comedy|Drama|Thri...
147845	1.0000000000000002	147845	[comedy, drama, t...	Manson Family Vac...	Comedy|Drama|Thri...
64327	1.0000000000000002	64327	[comedy, drama, t...	Fools' Parade (1971)	Comedy|Drama|Thri...
6193	1.0000000000000002	6193	[comedy, drama, t...	Poolhall Junkies ...	Comedy|Drama|Thri...
5416	1.0000000000000002	5416	[comedy, drama, t...	Cherish (2002)	Comedy|Drama|Thri...
2438	1.0000000000000002	2438	[comedy, drama, t...	Outside Ozona (1998)	Comedy|Drama|Thri...
92906	1.0000000000000002	92906	[comedy, drama, t...	Girls on the Road...	Comedy|Drama|Thri...
82097	1.0000000000000002	82097	[comedy, drama, t...	Karthik Calling K...	Comedy|Drama|Thri...
8330	1.0000000000000002	8330	[comedy, drama, t...	Our Man in Havana...	Comedy|Drama|Thri...
30767	1.0000000000000002	30767	[comedy, drama, t...	Sitcom (1998)	Comedy|Drama|Thri...

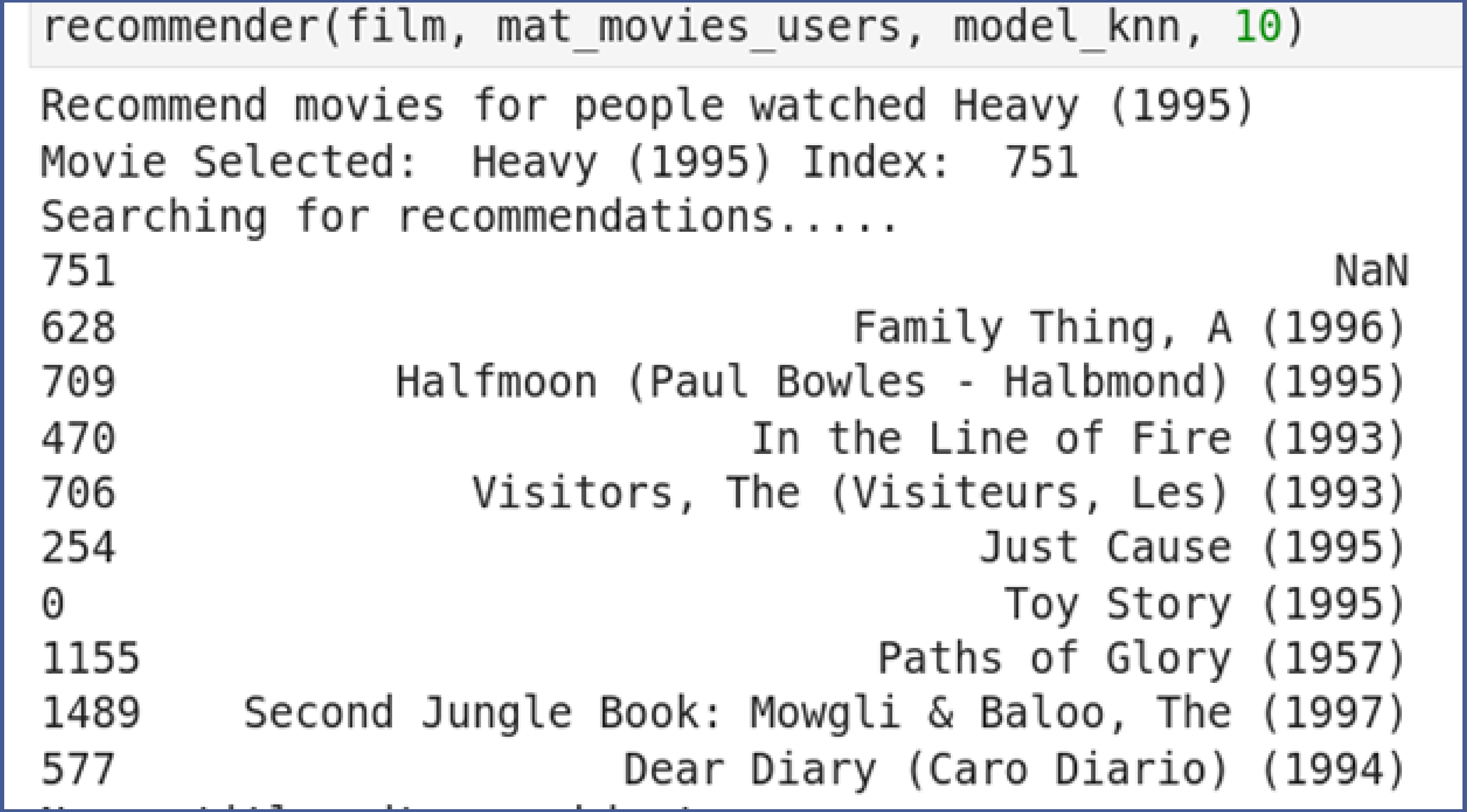
JACCARD SIMILARITY ALGORITHM

```
jaccard_recomm=jaccard_sim.join(movies_df, movies_df.movieId==jaccard_sim.movieId)\
.sort('jaccard_similarity',ascending=False).take(10)
jaccard_recomm_df = spark.createDataFrame(jaccard_recomm)
jaccard_recomm_df.join(movies, on="movieId")
```

movieId	jaccard_similarity	movieId	genre	title	genres
105835	1.0	105835	[comedy, drama, t...	Double, The (2013)	Comedy|Drama|Thri...
147845	1.0	147845	[comedy, drama, t...	Manson Family Vac...	Comedy|Drama|Thri...
64327	1.0	64327	[comedy, drama, t...	Fools' Parade (1971)	Comedy|Drama|Thri...
6193	1.0	6193	[comedy, drama, t...	Poolhall Junkies ...	Comedy|Drama|Thri...
5416	1.0	5416	[comedy, drama, t...	Cherish (2002)	Comedy|Drama|Thri...
2438	1.0	2438	[comedy, drama, t...	Outside Ozona (1998)	Comedy|Drama|Thri...
92906	1.0	92906	[comedy, drama, t...	Girls on the Road...	Comedy|Drama|Thri...
82097	1.0	82097	[comedy, drama, t...	Karthik Calling K...	Comedy|Drama|Thri...
8330	1.0	8330	[comedy, drama, t...	Our Man in Havana...	Comedy|Drama|Thri...
30767	1.0	30767	[comedy, drama, t...	Sitcom (1998)	Comedy|Drama|Thri...

05.3

KNN ALGORITHM

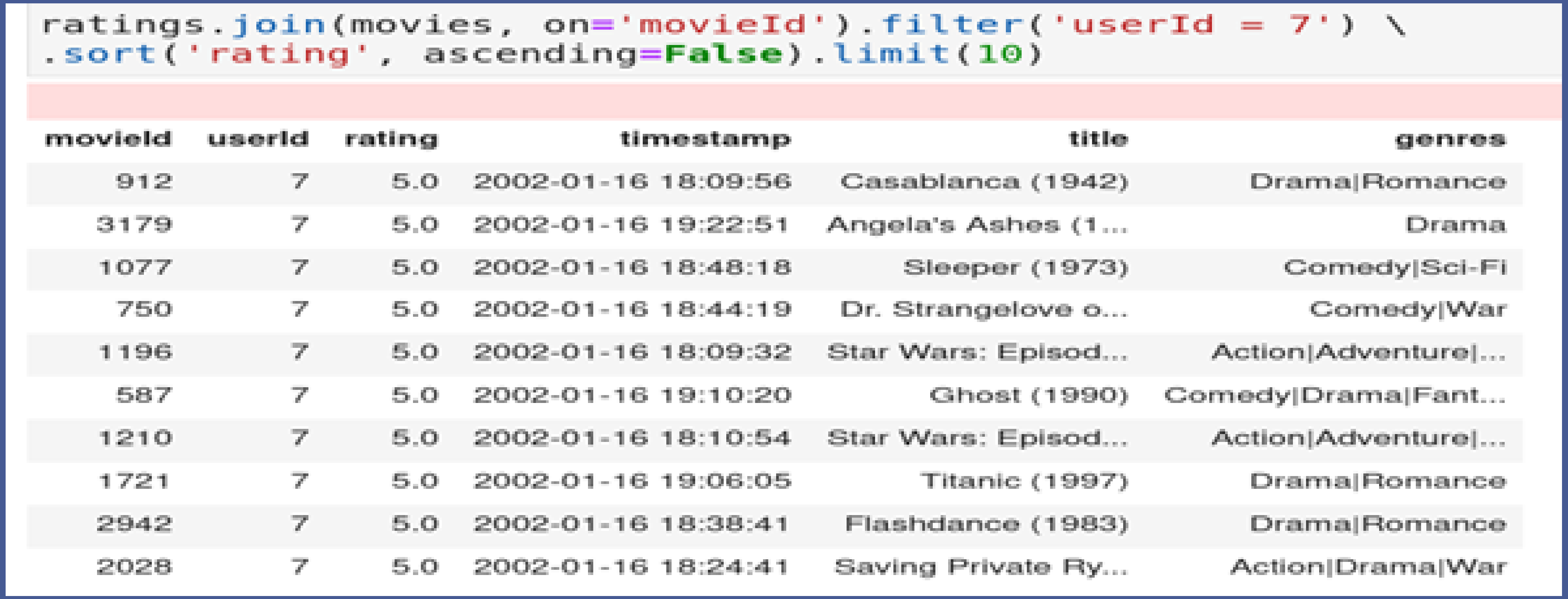


```
recommender(film, mat_movies_users, model_knn, 10)
```

```
Recommend movies for people watched Heavy (1995)
Movie Selected:  Heavy (1995) Index:  751
Searching for recommendations.....
751                                                      NaN
628                                      Family Thing, A (1996)
709                  Halfmoon (Paul Bowles - Halbmond) (1995)
470                                  In the Line of Fire (1993)
706                     Visitors, The (Visiteurs, Les) (1993)
254                                           Just Cause (1995)
0                                              Toy Story (1995)
1155                                      Paths of Glory (1957)
1489        Second Jungle Book: Mowgli & Baloo, The (1997)
577                                Dear Diary (Caro Diario) (1994)
```

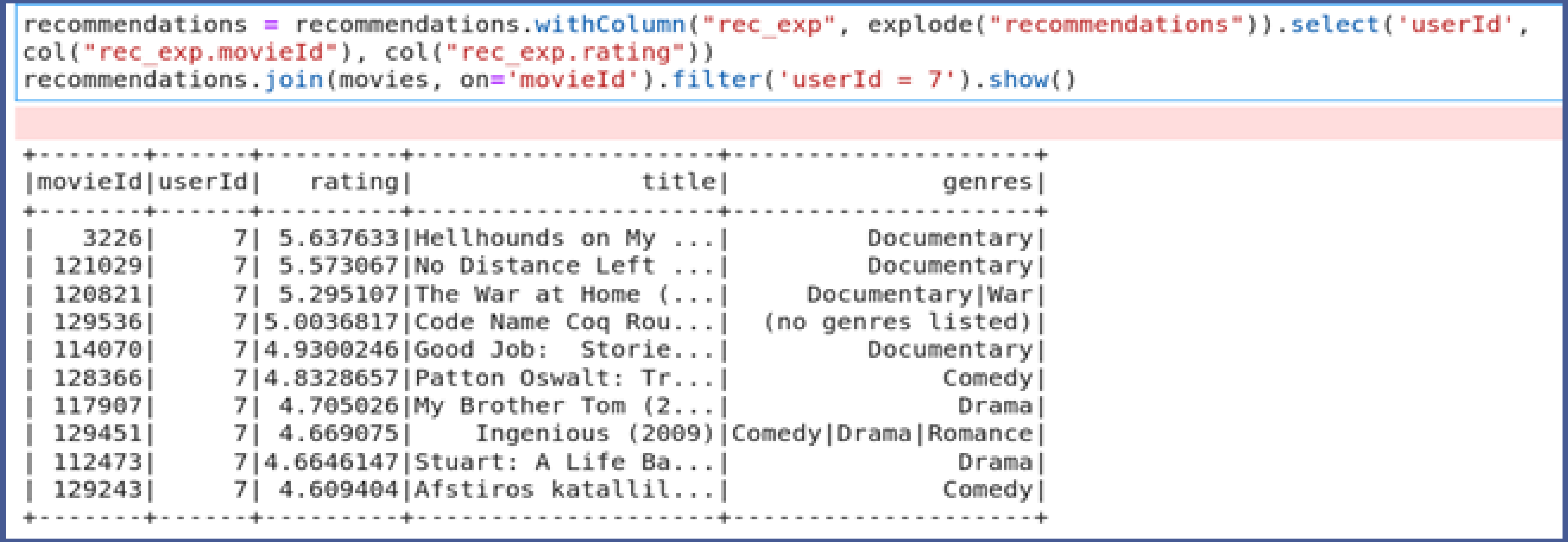
05.4

ALS ALGORITHM



```
ratings.join(movies, on='movieId').filter('userId = 7') \
.sort('rating', ascending=False).limit(10)
```

movieId	userId	rating	timestamp	title	genres
912	7	5.0	2002-01-16 18:09:56	Casablanca (1942)	Drama|Romance
3179	7	5.0	2002-01-16 19:22:51	Angela's Ashes (1...	Drama
1077	7	5.0	2002-01-16 18:48:18	Sleeper (1973)	Comedy|Sci-Fi
750	7	5.0	2002-01-16 18:44:19	Dr. Strangelove o...	Comedy|War
1196	7	5.0	2002-01-16 18:09:32	Star Wars: Episod...	Action|Adventure|...
587	7	5.0	2002-01-16 19:10:20	Ghost (1990)	Comedy|Drama|Fant...
1210	7	5.0	2002-01-16 18:10:54	Star Wars: Episod...	Action|Adventure|...
1721	7	5.0	2002-01-16 19:06:05	Titanic (1997)	Drama|Romance
2942	7	5.0	2002-01-16 18:38:41	Flashdance (1983)	Drama|Romance
2028	7	5.0	2002-01-16 18:24:41	Saving Private Ry...	Action|Drama|War



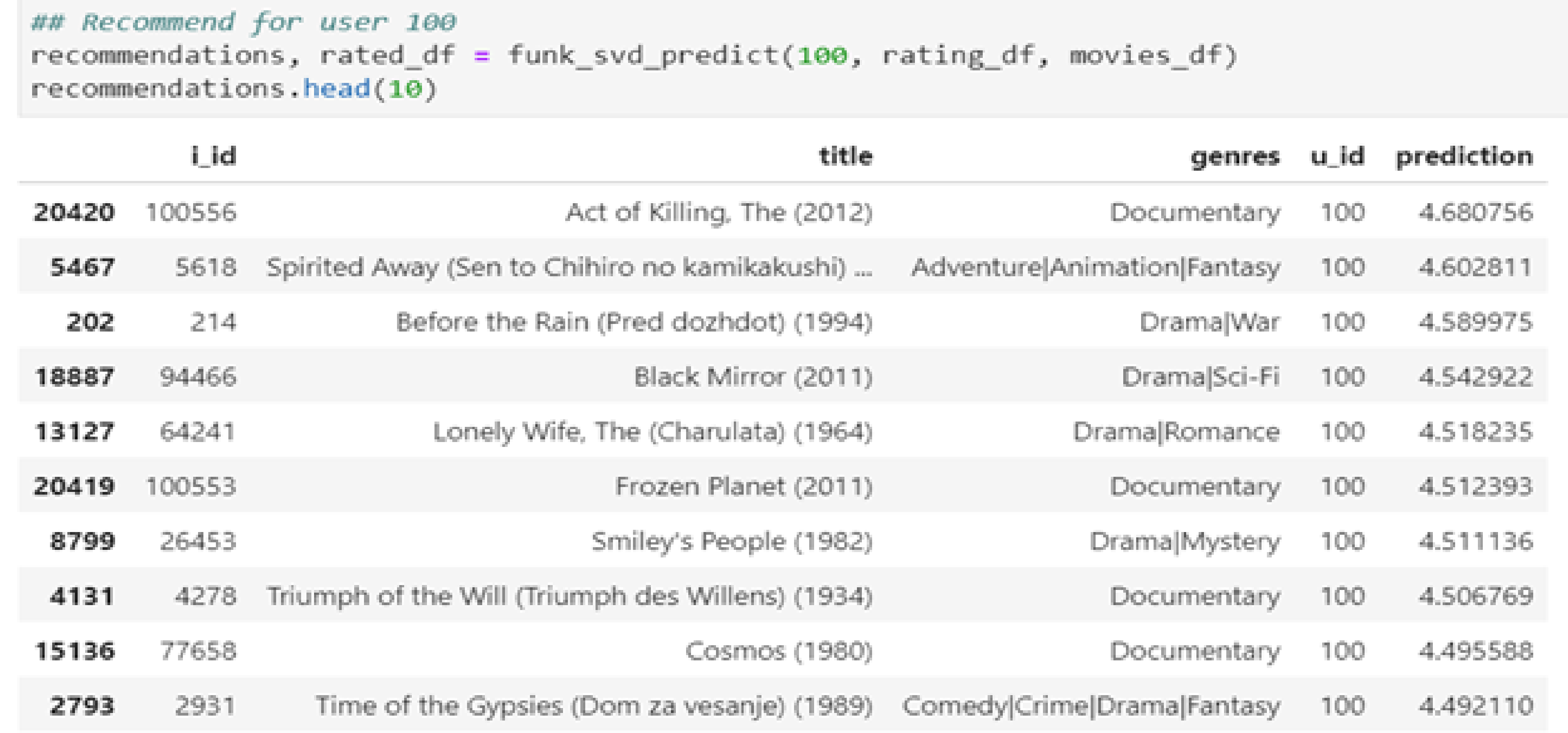
```
recommendations = recommendations.withColumn("rec_exp", explode("recommendations")).select('userId',
col("rec_exp.movieId"), col("rec_exp.rating"))
recommendations.join(movies, on='movieId').filter('userId = 7').show()
```

```
+-------+------+---------+--------------------+--------------------+
|movieId|userId|   rating|               title|              genres|
+-------+------+---------+--------------------+--------------------+
|   3226|     7| 5.637633|Hellhounds on My ...|         Documentary|
| 121029|     7| 5.573067|No Distance Left ...|         Documentary|
| 120821|     7| 5.295107|The War at Home (...|     Documentary|War|
| 129536|     7|5.0036817|Code Name Coq Rou...|  (no genres listed)|
| 114070|     7|4.9300246|Good Job:  Storie...|         Documentary|
| 128366|     7|4.8328657|Patton Oswalt: Tr...|              Comedy|
| 117907|     7| 4.705026|My Brother Tom (2...|               Drama|
| 129451|     7| 4.669075|    Ingenious (2009)|Comedy|Drama|Romance|
| 112473|     7|4.6646147|Stuart: A Life Ba...|               Drama|
| 129243|     7| 4.609404|Afstiros katallil...|              Comedy|
+-------+------+---------+--------------------+--------------------+
```

SVD ALGORITHM

```
ratings.join(movies, on='movieId').filter('userId = 100') \
.sort('rating', ascending=False).limit(10)
```

movieId	userId	rating	timestamp	title	genres
50	100	5.0	1996-06-25 16:24:49	Usual Suspects, T...	Crime|Mystery|Thr...
293	100	5.0	1996-06-25 16:28:27	Léon: The Profess...	Action|Crime|Dram...
680	100	5.0	1996-06-25 16:58:31	Alphaville (Alpha...	Drama|Mystery|Rom...
1449	100	5.0	1997-06-09 16:38:17	Waiting for Guffm...	Comedy
235	100	4.0	1996-06-25 16:28:27	Ed Wood (1994)	Comedy|Drama
162	100	4.0	1996-06-25 16:43:19	Crumb (1994)	Documentary
223	100	4.0	1996-06-25 16:31:02	Clerks (1994)	Comedy
260	100	4.0	1997-06-09 16:40:56	Star Wars: Episod...	Action|Adventure|...
265	100	4.0	1996-06-25 16:29:49	Like Water for Ch...	Drama|Fantasy|Rom...
288	100	4.0	1996-06-25 16:24:07	Natural Born Kill...	Action|Crime|Thri...



```
## Recommend for user 100
recommendations, rated_df = funk_svd_predict(100, rating_df, movies_df)
recommendations.head(10)
```

	i_id	title	genres	u_id	prediction
20420	100556	Act of Killing, The (2012)	Documentary	100	4.680756
5467	5618	Spirited Away (Sen to Chihiro no kamikakushi) ...	Adventure|Animation|Fantasy	100	4.602811
202	214	Before the Rain (Pred dozhdot) (1994)	Drama|War	100	4.589975
18887	94466	Black Mirror (2011)	Drama|Sci-Fi	100	4.542922
13127	64241	Lonely Wife, The (Charulata) (1964)	Drama|Romance	100	4.518235
20419	100553	Frozen Planet (2011)	Documentary	100	4.512393
8799	26453	Smiley's People (1982)	Drama|Mystery	100	4.511136
4131	4278	Triumph of the Will (Triumph des Willens) (1934)	Documentary	100	4.506769
15136	77658	Cosmos (1980)	Documentary	100	4.495588
2793	2931	Time of the Gypsies (Dom za vesanje) (1989)	Comedy|Crime|Drama|Fantasy	100	4.492110

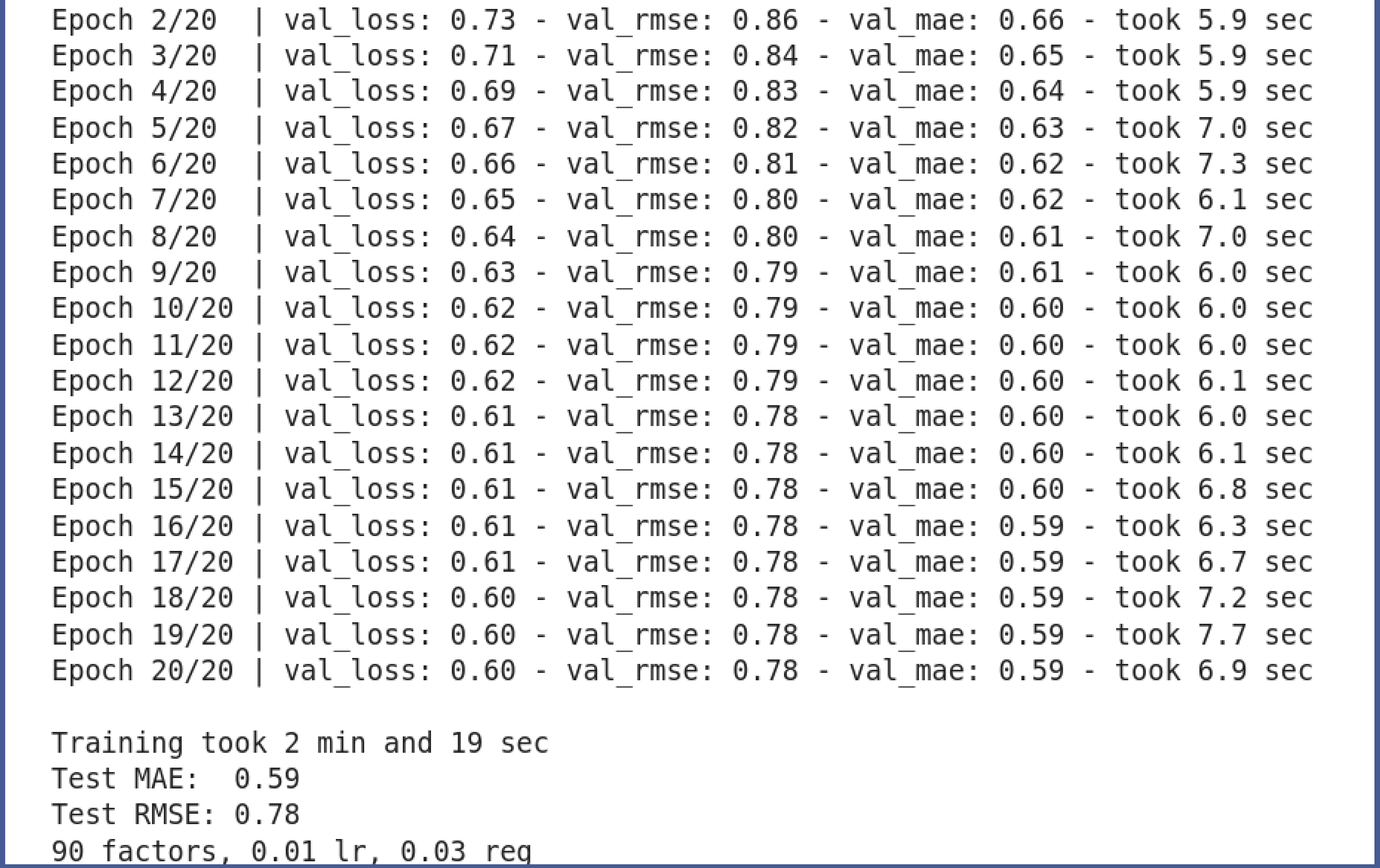
06

Performance Comparision

Movies recommendation system

06.1

SVD ALGORITHM

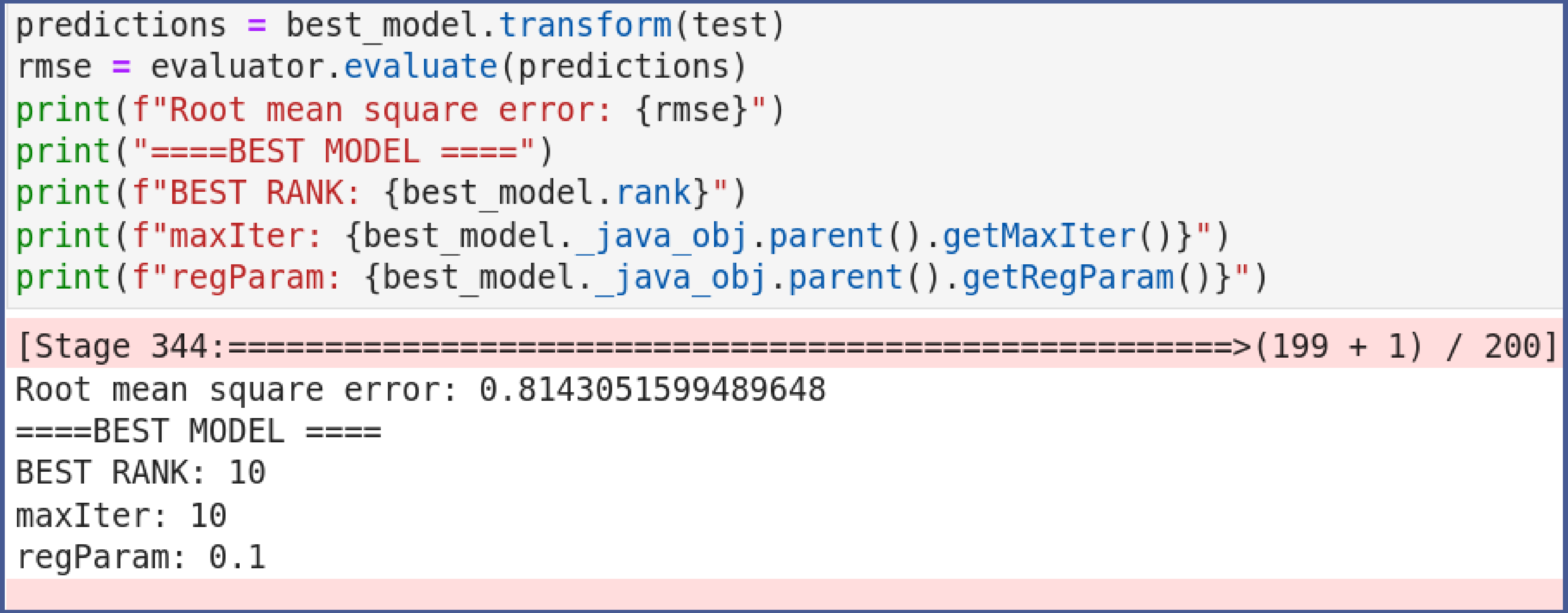


```
Epoch 1/20  | val_loss: 0.76 - val_rmse: 0.87 - val_mae: 0.67 - took 5.9 sec
Epoch 2/20  | val_loss: 0.73 - val_rmse: 0.86 - val_mae: 0.66 - took 5.9 sec
Epoch 3/20  | val_loss: 0.71 - val_rmse: 0.84 - val_mae: 0.65 - took 5.9 sec
Epoch 4/20  | val_loss: 0.69 - val_rmse: 0.83 - val_mae: 0.64 - took 5.9 sec
Epoch 5/20  | val_loss: 0.67 - val_rmse: 0.82 - val_mae: 0.63 - took 7.0 sec
Epoch 6/20  | val_loss: 0.66 - val_rmse: 0.81 - val_mae: 0.62 - took 7.3 sec
Epoch 7/20  | val_loss: 0.65 - val_rmse: 0.80 - val_mae: 0.62 - took 6.1 sec
Epoch 8/20  | val_loss: 0.64 - val_rmse: 0.80 - val_mae: 0.61 - took 7.0 sec
Epoch 9/20  | val_loss: 0.63 - val_rmse: 0.79 - val_mae: 0.61 - took 6.0 sec
Epoch 10/20 | val_loss: 0.62 - val_rmse: 0.79 - val_mae: 0.60 - took 6.0 sec
Epoch 11/20 | val_loss: 0.62 - val_rmse: 0.79 - val_mae: 0.60 - took 6.0 sec
Epoch 12/20 | val_loss: 0.62 - val_rmse: 0.79 - val_mae: 0.60 - took 6.1 sec
Epoch 13/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.60 - took 6.0 sec
Epoch 14/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.60 - took 6.1 sec
Epoch 15/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.60 - took 6.8 sec
Epoch 16/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.59 - took 6.3 sec
Epoch 17/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.59 - took 6.7 sec
Epoch 18/20 | val_loss: 0.60 - val_rmse: 0.78 - val_mae: 0.59 - took 7.2 sec
Epoch 19/20 | val_loss: 0.60 - val_rmse: 0.78 - val_mae: 0.59 - took 7.7 sec
Epoch 20/20 | val_loss: 0.60 - val_rmse: 0.78 - val_mae: 0.59 - took 6.9 sec

Training took 2 min and 19 sec
Test MAE:  0.59
Test RMSE: 0.78
90 factors, 0.01 lr, 0.03 reg
```

06.2

ALS ALGORITHM



```
predictions = best_model.transform(test)
rmse = evaluator.evaluate(predictions)
print(f"Root mean square error: {rmse}")
print("====BEST MODEL ====")
print(f"BEST RANK: {best_model.rank}")
print(f"maxIter: {best_model._java_obj.parent().getMaxIter()}")
print(f"regParam: {best_model._java_obj.parent().getRegParam()}")
```

```
[Stage 344:====================================================>(199 + 1) / 200]
Root mean square error: 0.8143051599489648
====BEST MODEL ====
BEST RANK: 10
maxIter: 10
regParam: 0.1
```

06

CONCLUSION

Movies recommendation system

95%

Movies Recommendation System

The most important problem that we aim to is to build an optimal movie recommendation system, as a result, we experiment the movie recommendation system with the data set obtained only at a good level on the entire datasets.

From the results in this model, in the future we want to perform the problem with a larger dataset and apply methods and techniques to increase system quality to bring the best results.

Group 7

NGUYEN THI KHANH HA	18520692
DINH XUAN HUNG	18520791
LAM LE DINH KHANG	18520885

Thank you for listening