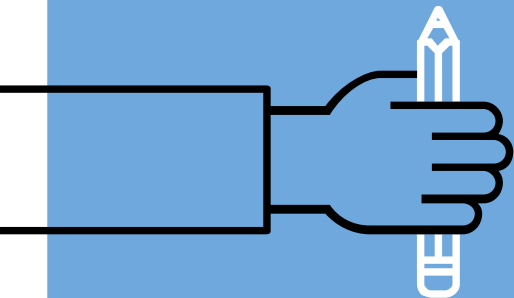
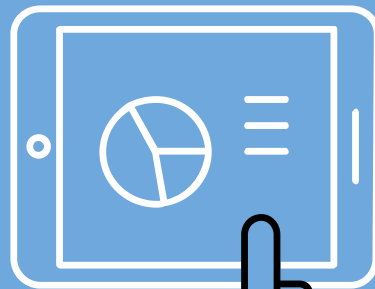
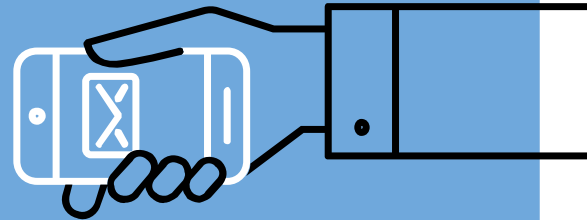
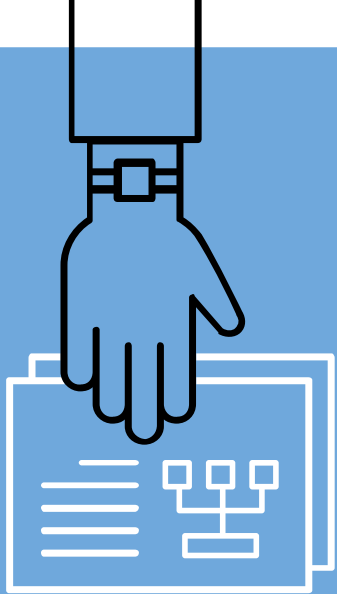


AI WITH RECOMMENDER SYSTEM

Check speaker's notes



TEAM PRESENTATION



Francisco José Cortés Delgado



Tomás Bernal Beltrán

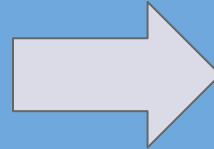
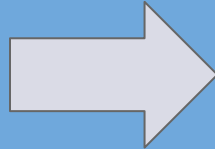


AGENDA

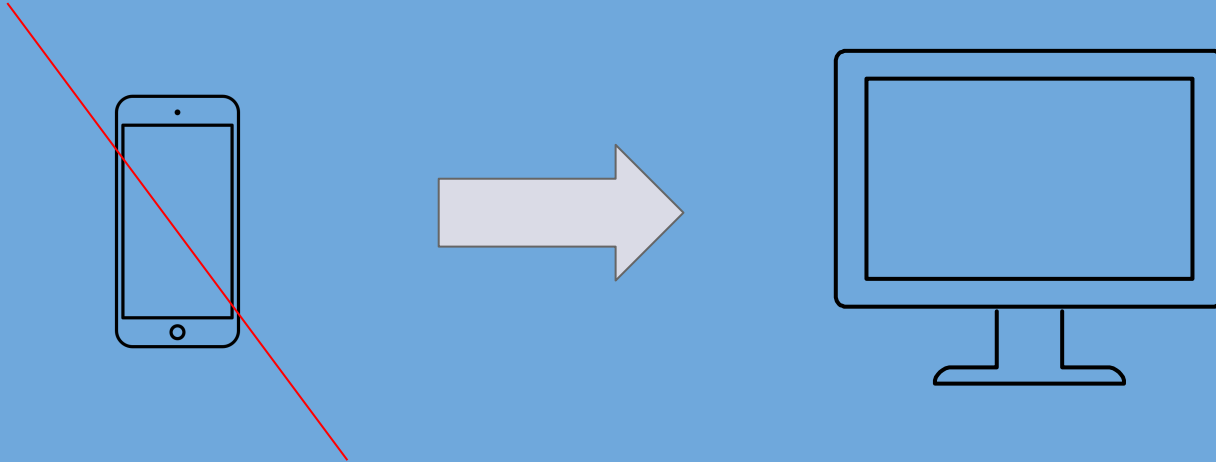
- ▷ *Appointment*
- ▷ *Motivation*
- ▷ *Goals*
- ▷ *Implementation*
- ▷ *Datasets*
- ▷ *Experiments and results*
- ▷ *Conclusions*
- ▷ *Future improvements*
- ▷ *Bibliography*



Appointments



Appointments



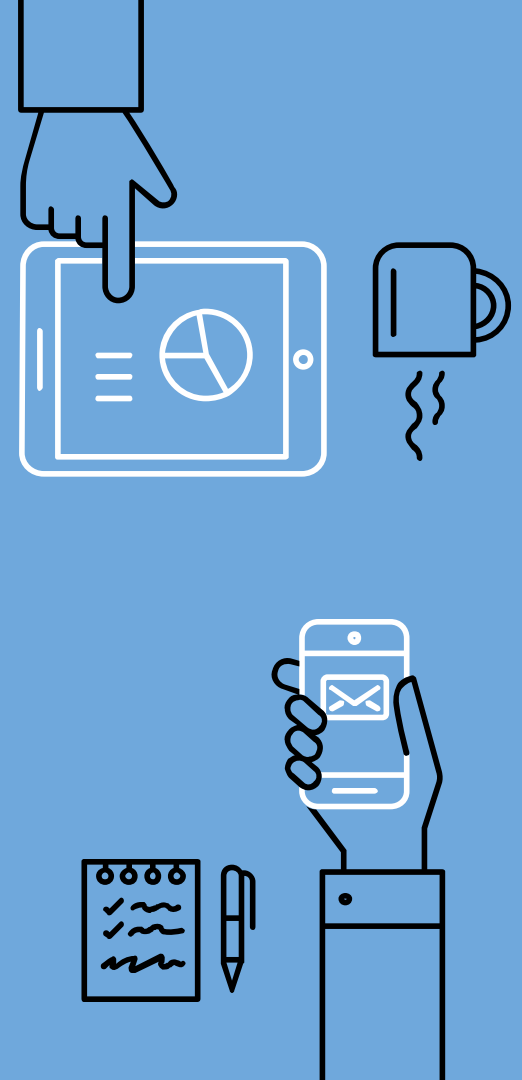
MOTIVATION

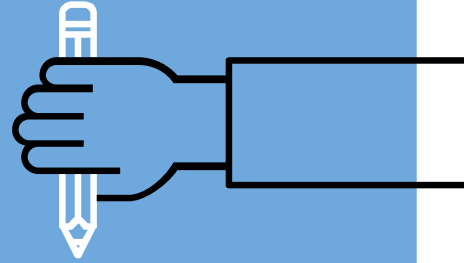
- ▶ AI is a very useful tool making recommendations
- ▶ It allows users to discover like-minded objects that they don't know about
- ▶ It can help people to know what they exactly will like but also It can help students to access questions that they have more problems with



GOALS

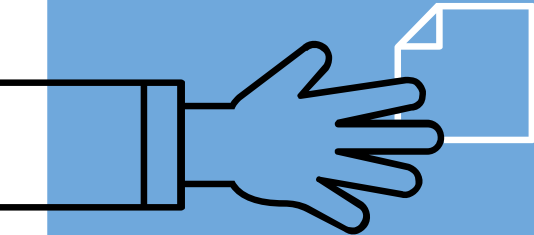
- ▶ Personalized recommendations based on a user's preferences
- ▶ Fix the typical disadvantages of the traditional recommender system with a hybrid system composed by:
 - A collaborative filtering
 - A content-based filtering
 - A knowledge filtering





IMPLEMENTATION








HOW DID WE CREATE OUR HYBRID SYSTEM?



COLLABORATIVE FILTERING [1][2][8]

- ▶ Based on the assumption that users will like items that are similar to the items that are liked before
- ▶ Generates recommendations for a user based on the similarity between items
- ▶ Use the ratings from the users to calculate this “similarity” between items

People's tastes in movies

	Saw	Rec	...	Insidious
A			...	
B			...	?
C			...	

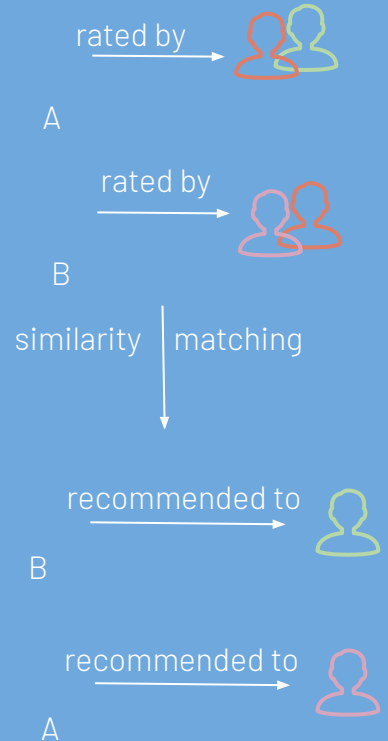
Does B like Insidious?
Probably yes, based on user
C ratings

ITEM-BASED APPROACH

[1][8][11]

- ▶ Takes into account the ratings that each item has received to establish recommendations
- ▶ More accurate and efficient than the user-based approach, since
 - It can be done offline
 - It's not dynamic, objects don't change
- ▶ It prevents the cold start problem and improves scalability

Item based approach



HOW HAVE WE IMPLEMENTED IT?

Using KNN algorithm and cosine similarity [1][3][4]

- ▶ User-Movie ratings stored in a matrix
- ▶ Each movie rated in a range from 1 to 5
- ▶ This User-Movie matrix is standardized
 - $\text{Value} = (\text{value} - \text{rowMean}) / (\text{rowMax} - \text{rowMin})$
- ▶ After this, it is transposed to be able to use the item-based approach

Original

	Saw	Ted	Rec	...
A	5	5	2	...
B	2	4	3	...
C	5	1	5	...

Standardized and Transposed

	A	B	C
Saw	0.3	-1	0.3
Ted	0.3	1	-0.6
Rec	-0.6	0	0.3
...

- ▷ Once the matrix is processed
- ▷ We create the cosine similarity matrix
- ▷ We calculate the cosine similarity of each movie (row) with the others is calculated:
 - $\text{CosineSimilarity} = \cos(0) = (A \times B) / (\|A\| \times \|B\|)$
 - $A \times B = \sum[1,n](A_i \times B_i) = (A_1 \times B_1) + \dots + (A_n \times B_n)$
 - $\|V\| = \sqrt{\sum[1,n](\text{pow}(V_i, 2))}$

Standardized and Transposed

	A	B	C
Saw	0.3	-1	0.3
Ted	0.3	1	-0.6
Rec	-0.6	0	0.3
...

Cosine similarity

	Saw	Ted	Rec
Saw	1	-0.83	-0.13
Ted	-0.83	1	-0.23
Rec	-0.13	-0.23	1
...

- ▶ Once the cosine similarity is calculated we apply the KNN algorithm
- ▶ New recommendation value is calculated using:
 - Cosine similarity of the movies
 - Ratings of the user
 - Filler (threshold) Ours is 2.5

Cosine similarity matrix
(preprocessed)

	Saw	Ted	Rec
Saw	1	-0.83	-0.13
Ted	-0.83	1	-0.23
Rec	-0.13	-0.23	1
...

User Fran ratings

Saw -> 2.0

Ted -> 5.0

Rec -> 4.0

Filler -> 2.5

How to calculate the new recommendation value based on the user ratings? [1]

- ▷ By multiplying the value of each movie with the rating given by the user minus the fixer

Cosine similarity matrix

Fixer -> 2.5

	Saw	Ted	Rec
Saw	1	-0.83	-0.13
Ted	-0.83	1	-0.23
Rec	-0.13	-0.23	1
...

User Fran rates Saw with a 2.0

Saw - Saw -> $1.0 * (2.0 - 2.5) = -0.5$

Saw - Ted -> $-0.83 * (2.0 - 2.5) = 0.41$

Saw - Rec -> $-0.13 * (2.0 - 2.5) = 0.06$

How does the fixer works? [1]

Cosine similarity matrix

	Saw	Ted	Rec
Saw	1	-0.83	-0.13
Ted	-0.83	1	-0.23
Rec	-0.13	-0.23	1
...

Fixer -> 3

User Fran rates Saw with a 2.0

Saw - Saw -> $1.0 * (2.0 - 3.0) = -1.0$

Saw - Ted -> $-0.83 * (2.0 - 3.0) = 0.83$

Saw - Rec -> $-0.13 * (2.0 - 3.0) = 0.13$

User Fran rates Saw with a 4.0

Saw - Saw -> $1.0 * (4.0 - 3.0) = 1.0$

Saw - Ted -> $-0.83 * (4.0 - 3.0) = -0.83$

Saw - Rec -> $-0.13 * (4.0 - 3.0) = -0.13$

Fixer -> 2.5

User Fran rates Saw with a 2.0

Saw - Saw -> $1.0 * (2.0 - 2.5) = -0.5$

Saw - Ted -> $-0.83 * (2.0 - 2.5) = 0.41$

Saw - Rec -> $-0.13 * (2.0 - 2.5) = 0.06$

User Fran rates Saw with a 4.0

Saw - Saw -> $1.0 * (4.0 - 2.5) = 1.5$

Saw - Ted -> $-0.83 * (4.0 - 2.5) = -1.24$

Saw - Rec -> $-0.13 * (4.0 - 2.5) = -0.19$

The value chosen as the threshold determines the recommendation value

What do we do with the recommendation values?[1]

- ▶ Once the new values are calculated
- ▶ The list of movies is sorted from highest to lowest

Individual recommendations

User Fran rates Saw with a 2.0

$$\text{Saw} - \text{Saw} \rightarrow 1.0 * (2.0 - 2.5) = -0.5$$

$$\text{Saw} - \text{Ted} \rightarrow -0.83 * (2.0 - 2.5) = 0.41$$

$$\text{Saw} - \text{Rec} \rightarrow -0.13 * (2.0 - 2.5) = 0.06$$

Recommendations \rightarrow Ted - Rec - Saw

User Fran rates Ted with a 5.0

$$\text{Ted} - \text{Saw} \rightarrow -0.83 * (5.0 - 2.5) = -2.07$$

$$\text{Ted} - \text{Ted} \rightarrow 1.0 * (5.0 - 2.5) = 2.5$$

$$\text{Ted} - \text{Rec} \rightarrow -0.23 * (5.0 - 2.5) = -0.57$$

Recommendations \rightarrow Ted - Rec - Saw

User Fran rates Rec with a 4.0

$$\text{Rec} - \text{Saw} \rightarrow -0.12 * (4.0 - 2.5) = -0.18$$

$$\text{Rec} - \text{Ted} \rightarrow -0.23 * (4.0 - 2.5) = -0.34$$

$$\text{Rec} - \text{Rec} \rightarrow 1.0 * (4.0 - 2.5) = 1.5$$

Recommendations \rightarrow Rec - Saw - Ted

How do we calculate a recommendation when a user rates several movies?[1]

User Fran rates Saw with a 2.0

Saw - Saw $\rightarrow 1.0 * (2.0 - 2.5) = -0.5$

Saw - Ted $\rightarrow -0.83 * (2.0 - 2.5) = 0.41$

Saw - Rec $\rightarrow -0.13 * (2.0 - 2.5) = 0.06$

User Fran rates Ted with a 5.0

Ted- Saw $\rightarrow -0.83 * (5.0 - 2.5) = -2.07$

Ted- Ted $\rightarrow 1.0 * (5.0 - 2.5) = 2.5$

Ted- Rec $\rightarrow -0.23 * (5.0 - 2.5) = -0.57$

User Fran rates Rec with a 4.0

Rec - Saw $\rightarrow -0.12 * (4.0 - 2.5) = -0.18$

Rec - Ted $\rightarrow -0.23 * (4.0 - 2.5) = -0.34$

Rec - Rec $\rightarrow 1.0 * (4.0 - 2.5) = 1.5$

Saw's collective recommendation value

Saw $\rightarrow -0.5 + (-2.07) + (-0.18) = -2.75$

Ted's collective recommendation value

Ted $\rightarrow 0.41 + 2.5 + (-0.34) = 2.57$

Rec's collective recommendation value

Rec $\rightarrow 0.06 + (-0.57) + 1.5 = 0.99$

Collective recommendation

Ted - Rec - Saw

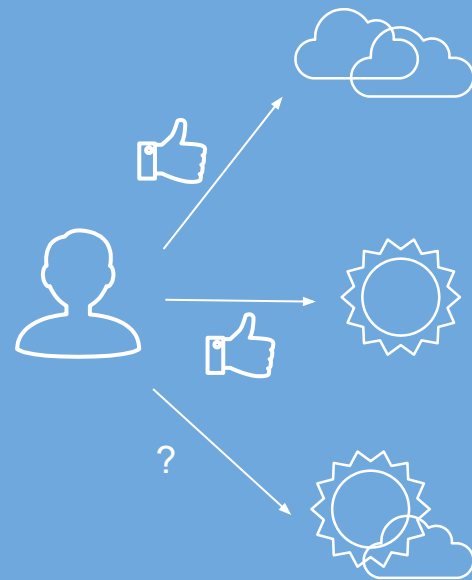
Disadvantages[1][9]

- ▷ The lack of user ratings causes a performance decline
Sparsity problem
- ▷ Many users and items means a large amount of computing power
Scalability problem
- ▷ For a new movie , there isn't enough data to make accurate recommendations
Cold-start / First-rater problem
- ▷ Difficulty creating recommendations for users with unique preferences
Grey-sheep problem

CONTENT-BASED FILTERING [1][5][6]

- ▶ Works with the characteristics of each item and the user's preference profile
- ▶ Recommendations don't get into account the rest of the users
- ▶ Recommends movies that are mostly similar to the user profile
- ▶ When a new movie appears the system uses its characteristics to including it in a recommendation

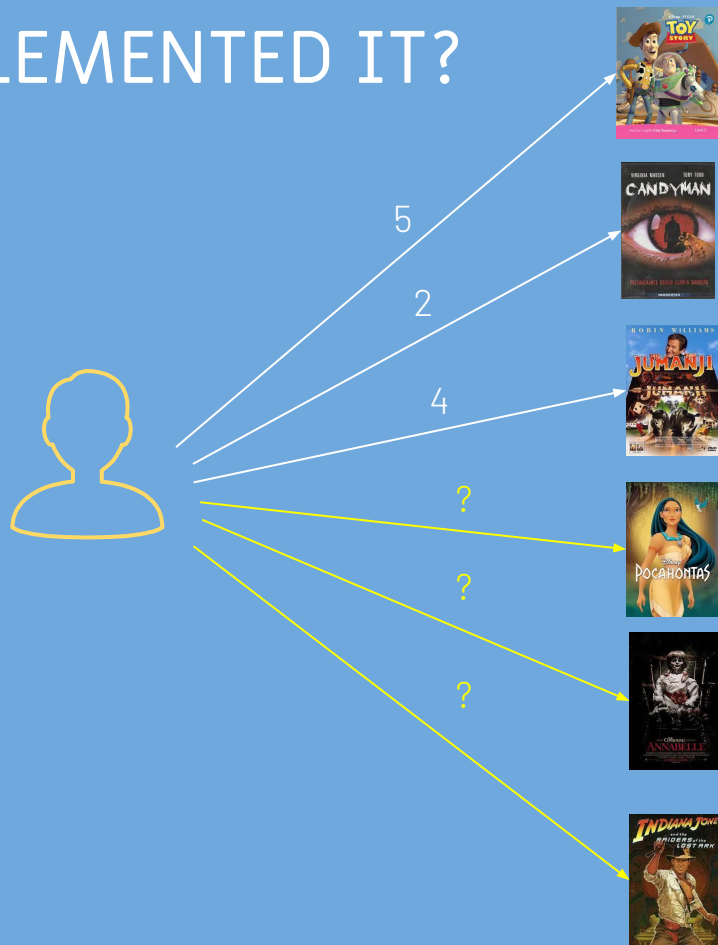
Climatological tastes of people







Probably yes, based on items characteristics

HOW HAVE WE IMPLEMENTED IT?

User Profile Approach[7]


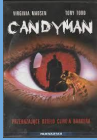



User ratings

			
	5	2	4

X

Matrix of genres
(preprocessed)




	ACTION	HORROR	ANIMATION	ADVENTURE
	0	0	1	1
	0	1	0	0
	1	0	0	1

User ratings




			
	5	2	4

X




Matrix of genres (preprocessed)

	ACTION	HORROR	ANIMATION	ADVENTURE
	0	0	1	1
	0	1	0	0
	1	0	0	1

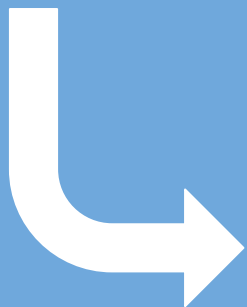
=

	ACTION	HORROR	ANIMATION	ADVENTURE
	0	0	5	5
	0	2	0	0
	4	0	0	4


Weighted genre matrix

	ACTION	HORROR	ANIMATION	ADVENTURE
	0	0	5	5
	0	2	0	0
	4	0	0	4


Weighted genre
matrix



User profile

	ACTION	HORROR	ANIMATION	ADVENTURE
	4	2	5	9




User profile

	ACTION	HORROR	ANIMATION	ADVENTURE
	-0.285	-0.571	-0.142	0.428









X



	ACTION	HORROR	ANIMATION	ADVENTURE
	1	0	1	1
	0	1	0	0
	1	0	0	1







Matrix of genres of the rest of movies

$$\text{value} = (\text{value} - \text{rowMean}) / (\text{rowMax} - \text{rowMin})$$







	ACTION	HORROR	ANIMATION	ADVENTURE
	-0.285	0	-0.142	0.428
	0	-0.571	0	0
	-0.285	0	0	0.428
	0	0	-0.142	0.428
	0	-0.571	0	0
	-0.285	0	0	0.428

Weighted genre matrix




Weighted genre matrix

	ACTION	HORROR	ANIMATION	ADVENTURE
	-0.285	0	-0.142	0.428
	0	-0.571	0	0
	-0.285	0	0	0.428
	0	0	-0.142	0.428
	0	-0.571	0	0
	-0.285	0	0	0.428

Sum of weighted genre matrix

	WEIGHT
	0
	-0.571
	0.142
	0.285
	-0.571
	0.142

RECOMMENDATION

	WEIGHT
	0.142
	0
	-0.571

Deleted watched movies
and
sorted in descending order

Disadvantages[1][6][9]

- ▶ Representation of the characteristics of the hand-designed article, on which the system is based
 - Limited content analysis problem
- ▶ The tendency to recommend with a limited degree of novelty
- ▶ The model can only make recommendations based on the user's existing interests, it will not recommend movies that are really different from these
 - Over-specialization problem

KNOWLEDGE -BASED FILTERING (KBRS) [2][10]

- ▶ Generates recommendations by querying the user's preferences
- ▶ Recommends the items that best satisfy the user's preferences
- ▶ Typical of old recommendation systems
- ▶ More effective when it's combined with other systems

Which genres don't you like?



Horror



Action



Animation

HOW HAVE WE IMPLEMENTED IT?

Using questionnaires with a constraint-based approach [8]

- ▷ This approach is based on the preferences that the user has
- ▷ To create the knowledge base, the system learns from the user
 - Using questionnaires to know the preferences of the user
- ▷ Building the knowledge base is not a trivial task
 - One must have a good knowledge of the product domain
 - Know what features matter to users and know their preferences
- ▷ This approach is based on the preferences that the user has



Which genres don't you like?

☒ _____

☐ _____

☐ _____

User's preferences



Knowledge base

User1 : Horror, Action

User2: Animation, Romantic



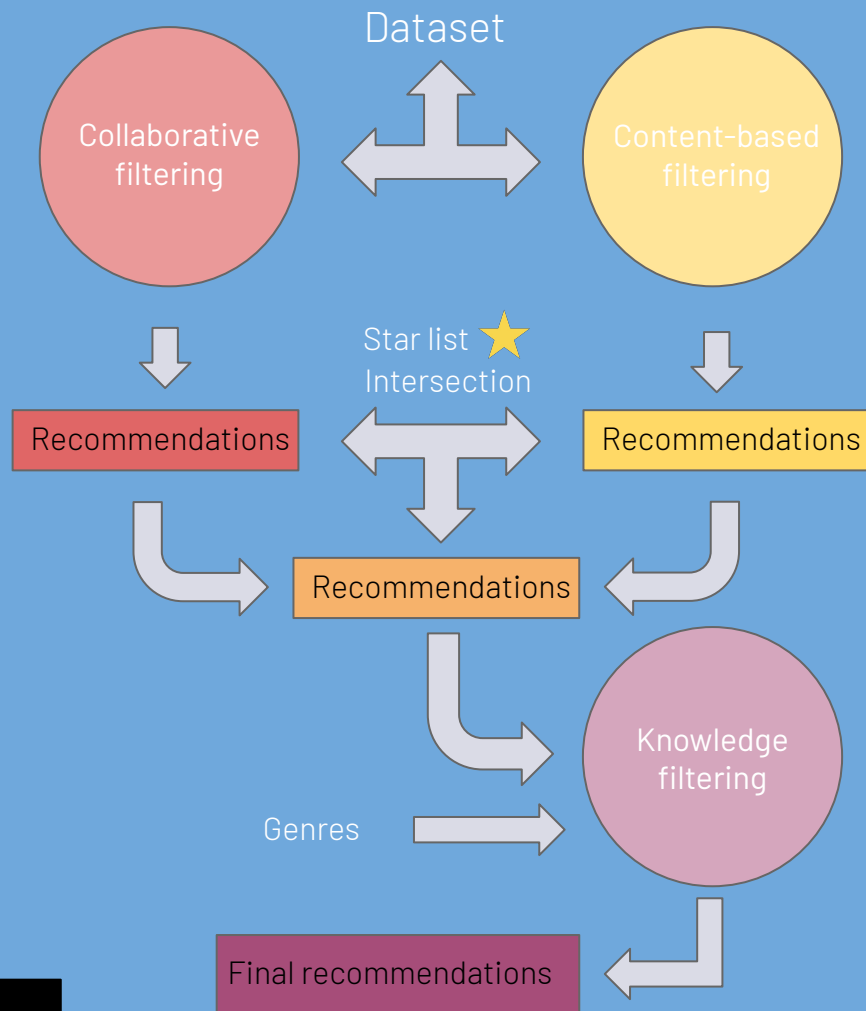
Recommendations

Disadvantages [10]

- ▷ The information obtained from the user does not allow us to make the best recommendations.
- ▷ It only helps us to eliminate movies that have disliked genres.

SOLUTION TO EVERY DISADVANTAGE

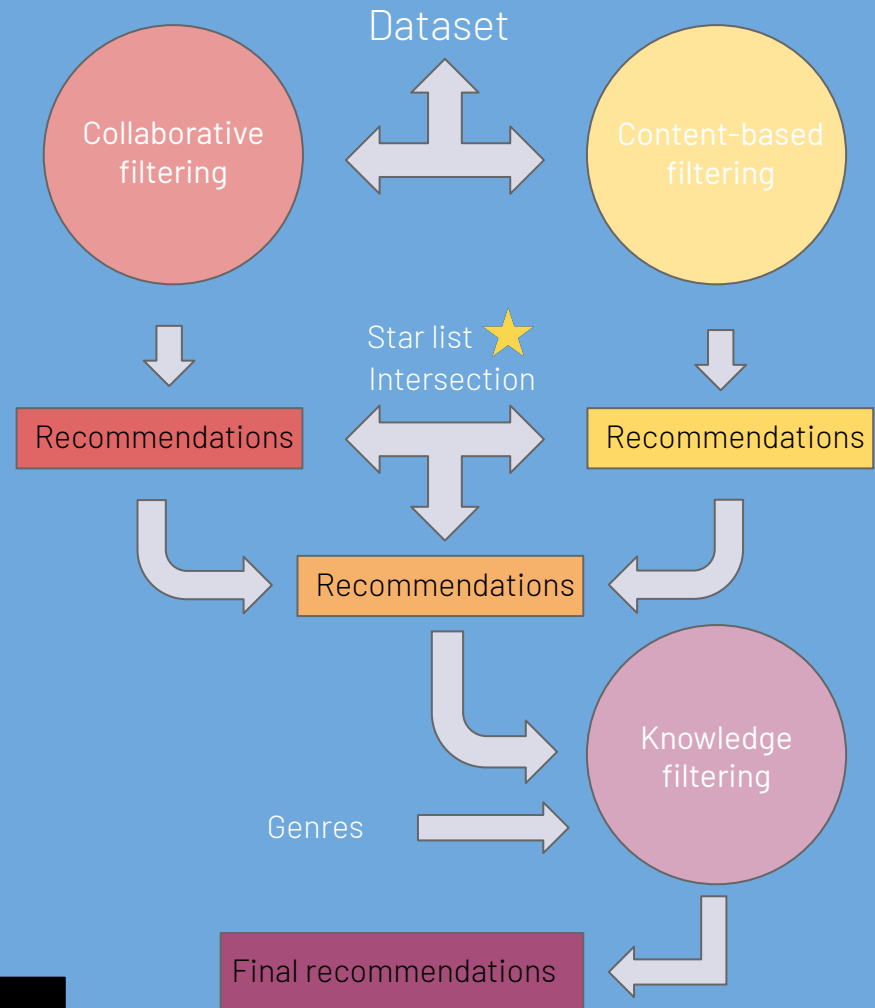
HYBRID
SYSTEM [1][2][8][9]



HYBRID SYSTEM

[1][2][8][9][11]

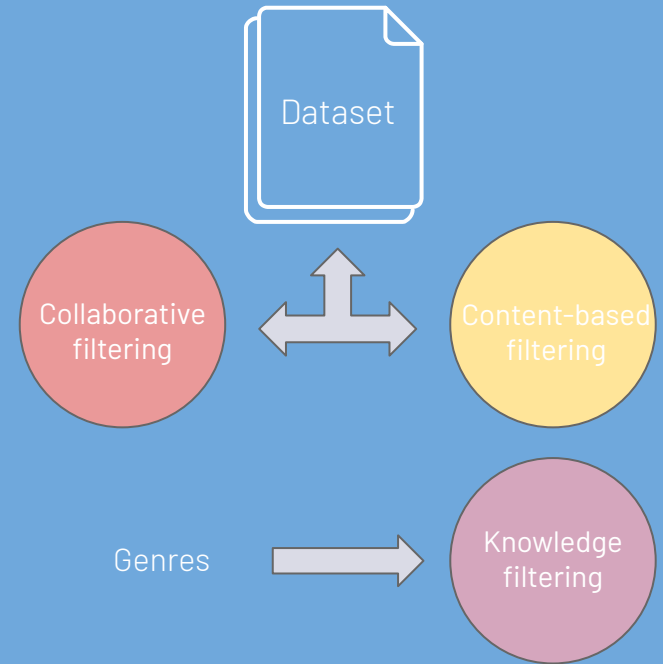
- ▶ We have implemented a hybrid system that combines three methods:
 - Collaborative filtering
 - Content-based filtering
 - Knowledge-based filtering
- ▶ Uses the mixed approach



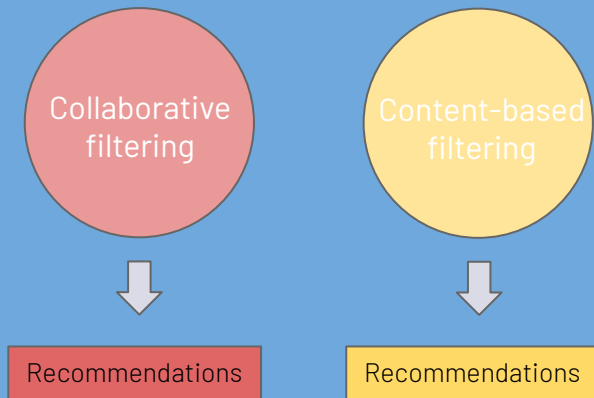
HOW HAVE WE IMPLEMENTED IT?

Using a Mixed Hybrid System [1]

- ▷ First, the dataset is extracted to collect information
 - Each movie is rated in a range from 1 to 5
 - Missing fields are rated with 0
- ▷ Secondly the genres that the user hates are obtained
- ▷ This information is extracted in order to be served to the algorithms



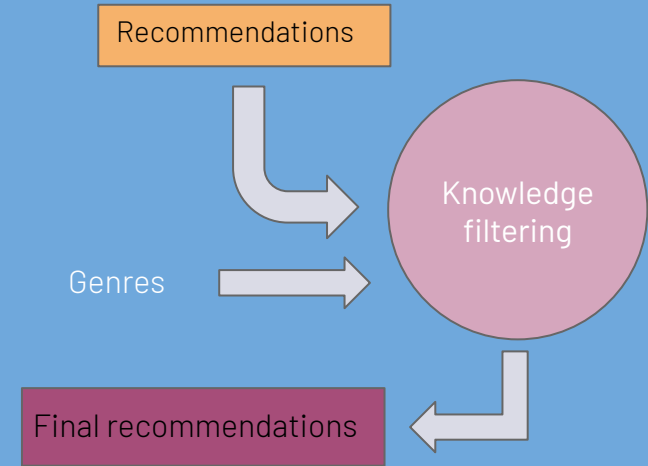
- ▷ Once each algorithm has the data in the format it requires
- ▷ We apply both algorithms separately
- ▷ Every algorithm generates its own recommendations



- ▶ We create a structure called Star List which includes the intersection of both recommendations
- ▶ Second, some of the collaborative recommendations that are not on the list are included
- ▶ Third, some of the content-based recommendations that are not on the list are included



- Once the list is complete
- We apply the Knowledge-based filter
- Removing each recommendation from the list that belongs to some genre/s that the user hates



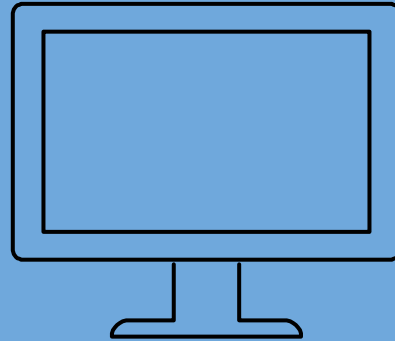
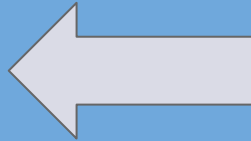
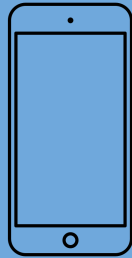
Advantages [1][8][9]

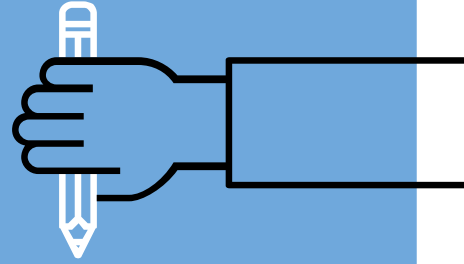
- ▷ Several studies have proved that hybrid methods can provide more accurate recommendations than regular approaches
- ▷ It helps to overcome some of the problems in recommender systems:
 - Cold-start problem, solved thanks to Content-based filtering, since it only needs the characteristics of a movie to be able to recommend it
 - Lower accuracy, thanks to the use of the Collaborative filtering and the Star list
 - Only safe recommendations, solved thanks to Collaborative filtering
 - Recommending movies that belong to a genre that the user hates, solved thanks to the use of Knowledge-based filtering

Disadvantages [1][8][9]

- ▷ The computation time is usually higher than pure approaches
 - Could lead to a slow app
- ▷ It still has problems such as scalability and sparse data
 - These problems occur when we increase the number of users and items

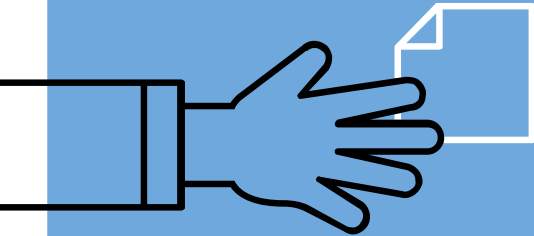
How to implement this in our topic?





DATASETS

WHAT DATA DO WE USE FOR OUR RECOMMENDATIONS?



Dataset description

- ▶ We use two different datasets:
 - "Toy" Dataset, contains 35 ratings on 9 movies
 - MovieLens Dataset, contains 100836 ratings on 9742 movies
- ▶ Both datasets consist of two .csv files, each line has the following format:
 - movies.csv ->
movieId(Integer),title(String),genres(String)
 - ratings.csv ->
userId(Integer),movieId(Integer),rating(Real)
- ▶ rating values are in the range [1.0-5.0] with 0.5 increments

movies.csv

```
1,Toy Story,Animation:Adventure
2,Jumanji,Adventure:Children:Fantasy
...
150,Apollo 13,Adventure:Drama:IMAX
...
3157,Stuart Little,Children:Comedy
...
```

ratings.csv

```
1,1,5.0
1,150,4.0
...
47,2,3.0
...
```

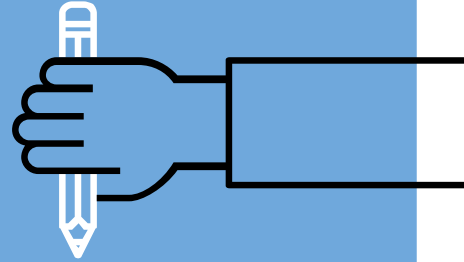

- ▷ In these datasets there are missing values, most movies have not been rated by any user
- ▷ Both datasets are preprocessed, the dataframe entries are transformed, rating the missing information fields with a 0
- ▷ Both datasets, are standardized

“Toy” Dataset

- ▷ This data was randomly created on December 23, 2021 by ourselves

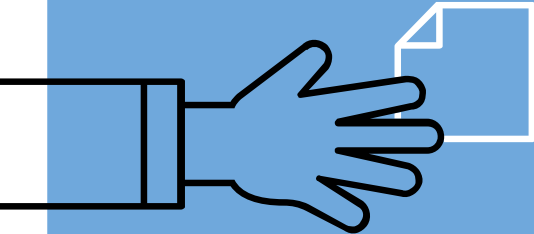
MovieLens Dataset

- ▷ This dataset was generated on September 26, 2018 by MovieLens

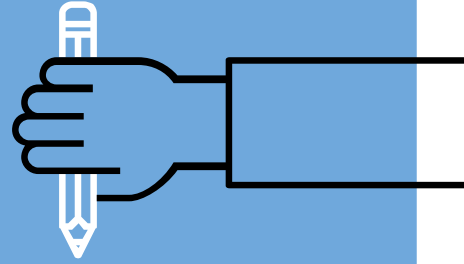


RESEARCH ENVIRONMENT

WHAT ENVIRONMENT HAVE WE USED IN
THE IMPLEMENTATION OF THE WORK?

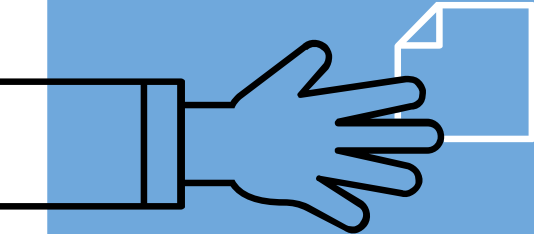


- ▷ The project has been implemented in Java
 - Version 8.302.08.1
- ▷ Using the programming environment Eclipse
 - Version 2021-12 (4.22.0)
- ▷ The equipment used was:
 - MacBook Pro (2021)
 - Processor -> M1 Pro
 - RAM -> 16 GB

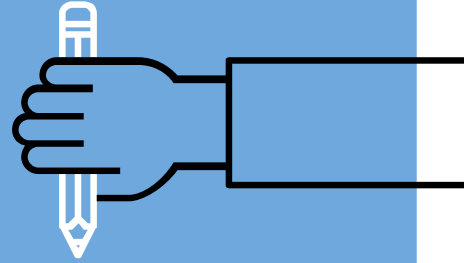


MEASURES

WHAT MEASUREMENTS DO WE USE IN OUR
EXPERIMENTAL RESULTS?

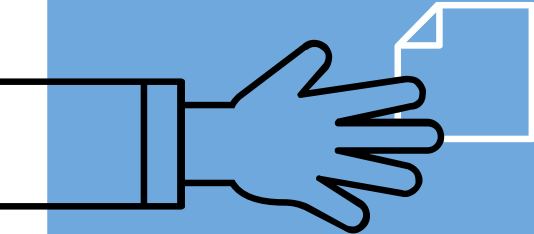


- ▷ In the group of experiments related to scalability
 - Results represent the time it takes our algorithm to generate a recommendation
 - Measured in seconds
- ▷ The other experiments do not use any measure
 - Beyond the appearance or not of a movie in the recommendations
 - Since we test how the hybrid system solves the different problems of the recommender systems



EXPERIMENTS AND RESULTS

DO WE SOLVE THE PROBLEMS THAT
RECOMMENDATION FILTERS HAVE?



Experiment 1. Good recommendations

Goal

- ▷ In this experiment we want to prove that our recommendation makes sense taking into account the ratings of the user and the genres that the user hates

Assumptions

- ▷ In this experiment the preprocessing of the data doesn't matter, the result will be the same it only affects to the time of the execution of our algorithm

Assumptions

- ▶ Genres hated by the user:

Horror

Drama

- ▶ User ratings:

Toy Story (1995) with a 5.0 Adventure:Animation:Children:Comedy:Fantasy

Jumanji (1995) with a 4.0 Adventure:Children:Fantasy

Casper (1995) with a 3.5 Adventure:Children

Toy Story 2 (1999) with a 5.0 Adventure:Animation:Children:Comedy:Fantasy

Toy Story 3 (2010) with a 3.5 Adventure:Animation:Children:Comedy:Fantasy:IMAX

Shrek Forever After (a.k.a. Shrek - The Final Chapter) (2010) with a 5.0

Adventure:Animation:Children:Comedy:Fantasy:IMAX

Shrek (2001) with a 5.0 Shrek 2 (2004) with a 2.5

Adventure:Animation:Children:Comedy:Fantasy:Romance

Garfield- The Movie (2004) with a 4.5 Animation:Children:Comedy

Course of the experiment

- ▷ We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give. In this experiment the genre that the user hates are horror and drama
- ▷ Afterwards we must indicate what movies have the user rated

Results

1 Aladdin (1992) : Adventure | Animation | Children | Comedy | Musical
2 Jurassic Park (1993) : Action | Adventure | Sci-Fi | Thriller
3 Beauty and the Beast (1991) : Animation | Children | Fantasy | Musical | Romance | IMAX
4 Star Wars- Episode IV - A New Hope (1977) : Action | Adventure | Sci-Fi
5 "Mask, The (1994)" : Action | Comedy | Crime | Fantasy
6 Home Alone (1990) : Children | Comedy
7 Mission- Impossible (1996) : Action | Adventure | Mystery | Thriller
8 Terminator 2- Judgment Day (1991) : Action | Sci-Fi
9 "Nightmare Before Christmas, The (1993)" : Animation | Children | Fantasy | Musical
10 Speed (1994) : Action | Romance | Thriller
11 Ace Ventura- Pet Detective (1994) : Comedy
12 Seven (a.k.a. Se7en) (1995) : Mystery | Thriller
13 Ace Ventura- When Nature Calls (1995) : Comedy
14 Happy Gilmore (1996) : Comedy
15 Pretty Woman (1990) : Comedy | Romance
16 True Lies (1994) : Action | Adventure | Comedy | Romance | Thriller
17 Batman (1989) : Action | Crime | Thriller
18 Twelve Monkeys (a.k.a. 12 Monkeys) (1995) : Mystery | Sci-Fi | Thriller
19 Twister (1996) : Action | Adventure | Romance | Thriller
20 Pinocchio (1940) : Animation | Children | Fantasy | Musical

4546 Cloverfield (2008) : Action | Mystery | Sci-Fi | Thriller
4547 Sherlock Holmes (2009) : Action | Crime | Mystery | Thriller
4548 Safe House (2012) : Action | Crime | Mystery | Thriller
4549 Alex Cross (2012) : Action | Crime | Mystery | Thriller
4550 "Walk Among the Tombstones, A (2014)" : Action | Crime | Mystery | Thriller
4551 Predestination (2014) : Action | Mystery | Sci-Fi | Thriller
4552 Self/less (2015) : Action | Mystery | Sci-Fi | Thriller
4553 Wind River (2017) : Action | Crime | Mystery | Thriller
4554 Maze Runner- The Death Cure (2018) : Action | Mystery | Sci-Fi | Thriller
4555 Cowboys & Aliens (2011) : Action | Sci-Fi | Thriller | Western | IMAX
4556 "Big Easy, The (1987)" : Action | Crime | Mystery | Romance | Thriller
4557 "X-Files- Fight the Future, The (1998)" : Action | Crime | Mystery | Sci-Fi | Thriller
4558 Minority Report (2002) : Action | Crime | Mystery | Sci-Fi | Thriller
4559 Sin City (2005) : Action | Crime | Film-Noir | Mystery | Thriller

Comments

- ▶ As expected, the movies present at the beginning the recommendation of the hybrid system, belongs to the genre that the user likes, thanks to the content based but also some movies that belong to other genres could be shown thanks to the collaborative like Aladdin which also belongs to the musical genre
- ▶ Furthermore, we can clearly see that movies which belong to the horror and drama genres are not present in our final recommendations
- ▶ We have noticed that no matter how many times we repeat this experiment, the result is always the same because the data is not changing

Experiment 2. Cold start problem

Goal

- In this experiment we want to demonstrate how the hybrid system solves the problem that arises when adding movies to the catalog that have not been rated by anyone

Assumptions

- In this experiment the preprocessing of the data doesn't matter, the result will be the same it only affects to the time of the execution of our algorithm
- To check if the cold start problem is solved by our system, we need to insert new movies in our dataset that have not been rated by any user
- Once we have chosen the new movies, we have to make a user positively rate movies similar to the new ones, in order to check if the system takes these new movies into account when making recommendations

Assumptions

- ▶ Genres hated by the user:

Animation

Drama

- ▶ User ratings:

Dracula- Dead and Loving It (1995) with a 5.0 Comedy:Horror

"Prophecy, The (1995)" with a 4.0 Fantasy:Horror:Mystery

Relative Fear (1994) with a 5.0 Horror:Thriller

"Silence of the Lambs, The (1991)" with a 5.0 Crime:Horror:Thriller

Thinner (1996) with a 3.5 Horror:Thriller

"Conjuring, The (2013)" with a 5.0 Horror:Thriller

The Conjuring 2 (2016) with a 4.0 Horror

The Purge- Election Year (2016) with a 2.5 Action:Horror:Sci-Fi

Sharknado 4- The 4th Awakens (2016) with a 1.5 Action:Adventure:Horror:Sci-Fi

Winnie the Pooh and the Day of Concern (1972) with a 1.0 Animation:Children

Blair Witch (2016) with a 3.0 Horror:Thriller

Course of the experiment

- ▶ First, the new movies are added to the dataset, we add the following lines to the movies.csv file:
193610,The Nun,Horror:Thriller
193611,Annabelle Comes Home,Horror
193612,The Curse of La Llorona,Horror
- ▶ We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- ▶ In this experiment the genre that the user hates are animation and drama
- ▶ Afterwards we must indicate what movies have the user rated

Results

1008 It (2017)
1009 Annabelle Comes Home
1010 The Curse of La Llorona
1011 Army of Darkness (1993)
1012 "Great Yokai War, The (Yôkai daisensô) (2005)"
1013 Dead Snow (Død snø) (2009)
1014 "Monster Squad, The (1987)"

4812 Pearl Jam Twenty (2011)
4813 "Civil War, The (1990)"
4814 Virunga (2014)
4815 The Nun
4816 "Wiz, The (1978)"
4817 Chitty Chitty Bang Bang (1968)
4818 Charlie and the Chocolate Factory (2005)
4819 Meet Dave (2008)

Comments

- ▶ As we can see the recommendation of the content filtering includes the new movies, due to the genres to which the movie belongs
- ▶ The movies appear in a good position, especially Annabelle Comes Home and The Curse of La Llorona, as these movies are pure Horror movies, the genre that has the most weight in the user's taste profile
- ▶ We have noticed that no matter how many times we repeat this experiment, the result is always the same because the data is not changing

Experiment 3. Only safe recommendations problem

Goal

- In this problem we want to demonstrate how the hybrid system solves the problem that arises when a user has very specific tastes (Grey-sheep problem), that is when a user only evaluates movies of a certain genre or genres

Assumptions

- In this experiment the preprocessing of the data doesn't matter, the result will be the same it only affects to the time of the execution of our algorithm
- To check if the Grey-sheep problem is solved by our system we do not need to make changes in the dataset. But we should choose ratings that determine specific tastes, for example only give positive ratings to pure comedy movies

Assumptions

► Genres hated by the user:

Horror

Thriller

► User ratings:

Dracula- Dead and Loving It (1995) with a 1.0 Comedy:Horror

"Prophecy, The (1995)" with a 2.0 Fantasy:Horror:Mystery

Relative Fear (1994) with a 1.0 Horror:Thriller

"Silence of the Lambs, The (1991)" with a 1.0 Crime:Horror:Thriller

Thinner (1996) with a 1.5 Horror:Thriller

"Conjuring, The (2013)" with a 2.0 Horror:Thriller

The Conjuring 2 (2016) with a 1.0 Horror

The Purge- Election Year (2016) with a 1.0 Action:Horror:Sci-Fi

Sharknado 4- The 4th Awakens (2016) with a 1.5 Action:Adventure:Horror:Sci-Fi

Winnie the Pooh and the Day of Concern (1972) with a 1.0 Animation:Children

Blair Witch (2016) with a 3.0 Horror:Thriller

Bad Moms (2016) with a 4.5 Comedy

Why Him? (2016) with a 4.0 Comedy

Comedy Central Roast of David Hasselhoff (2010) with a 5.0 Comedy

Gabriel Iglesias- Hot and Fluffy (2007) with a 4.5 Comedy

Ghost Graduation (2012) with a 5.0 Comedy

Jackass Presents- Bad Grandpa (2013) with a 4.5 Comedy

Grown Ups 2 (2013) with a 5.0 Comedy

Ocho apellidos vascos (2014) with a 5.0 Comedy

Too Many Cooks (2014) with a 4.0 Comedy

War Dogs (2016) with a 3.5 Comedy

Course of the experiment

- ▷ We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- ▷ In this experiment the genre that the user hates are animation and drama
- ▷ Afterwards we must indicate what movies have the user rated.

Results

1 Fear of a Black Hat (1994) : Comedy
2 Love & Human Remains (1993) : Comedy | Drama
3 "Inkwell, The (1994)" : Comedy | Drama
4 Delta of Venus (1995) : Drama
5 House Party 3 (1994) : Comedy
6 Pushing Hands (Tui shou) (1992) : Drama
7 Above the Rim (1994) : Crime | Drama
8 Pie in the Sky (1996) : Comedy | Romance
9 Miami Rhapsody (1995) : Comedy
10 Mad Love (1995) : Drama | Romance
11 Sunset Park (1996) : Drama
12 Two Much (1995) : Comedy | Romance
13 Chasers (1994) : Comedy
14 Jimmy Hollywood (1994) : Comedy | Crime | Drama
15 Bread and Chocolate (Pane e cioccolata) (1973) : Comedy | Drama
16 "Cure, The (1995)" : Drama
17 Reckless (1995) : Comedy | Fantasy
18 Poetic Justice (1993) : Drama

7298 Kiki's Delivery Service (Majo no takkyūbin) (1989) : Adventure | Animation | Children | Drama | Fantasy
7299 "Rudolph, the Red-Nosed Reindeer (1964)" : Adventure | Animation | Children | Fantasy | Musical
7300 How to Train Your Dragon (2010) : Adventure | Animation | Children | Fantasy | IMAX
7301 Rise of the Guardians (2012) : Adventure | Animation | Children | Fantasy | IMAX
7302 Harry Potter and the Deathly Hallows- Part 2 (2011) : Action | Adventure | Drama | Fantasy | Mystery | IMAX
7303 El Cid (1961) : Action | Adventure | Drama | Romance | War
7304 Robin Hood (2010) : Action | Adventure | Drama | Romance | War
7305 "Misérables, Les (2012)" : Drama | Musical | Romance | IMAX
7306 "Rescuers, The (1977)" : Adventure | Animation | Children | Crime | Drama
7307 "Spiderwick Chronicles, The (2008)" : Adventure | Children | Drama | Fantasy | IMAX
7308 Where the Wild Things Are (2009) : Adventure | Children | Drama | Fantasy | IMAX
7309 "Twilight Saga- Breaking Dawn - Part 2, The (2012)" : Adventure | Drama | Fantasy | Romance | IMAX
7310 Cinderella (1950) : Animation | Children | Fantasy | Musical | Romance
7311 "Christmas Carol, A (2009)" : Animation | Children | Drama | Fantasy | IMAX
7312 "Princess and the Frog, The (2009)" : Animation | Children | Fantasy | Musical | Romance
7313 Anastasia (1997) : Adventure | Animation | Children | Drama | Musical

Comments

- ▶ As we can see the system recommends movies of different genres, not only of the genre to which the user has rated, since thanks to the recommendations of the collaborative filtering, movies of different genres that have been liked by users with similar profiles to the user of the experiment will be recommended
- ▶ We have noticed that no matter how many times we repeat this experiment, the result is always the same because the data is not changing.

Experiment 4. Recommendations with Hated Genres problem

Goal

- In this experiment we want to demonstrate how the hybrid system solves the problem of containing movies in the recommendation that belong to genres that the user hates

Assumptions

- In this experiment the preprocessing of the data doesn't matter, the result will be the same it only affects to the time of the execution of our algorithm.
- To check if the problem of recommendations with genres that the user hates is solved by our hybrid system, we do not need to make changes in the dataset.
- We don't have to choose specific ratings either, we simply have to make ratings and indicate the genres the user hates

Assumptions

► Genres hated by the user:

Animation

Children

Documentary

► User ratings:

Comedy Central Roast of David Hasselhoff (2010) with a 2.0 Comedy

Gabriel Iglesias- Hot and Fluffy (2007) with a 1.5 Comedy

Ghost Graduation (2012) with a 3.0 Comedy

Jackass Presents- Bad Grandpa (2013) with a 4.5 Comedy

Grown Ups 2 (2013) with a 5.0 Comedy

Ocho apellidos vascos (2014) with a 5.0 Comedy

Too Many Cooks (2014) with a 2.0 Comedy

War Dogs (2016) with a 4.5 Comedy

Bad Moms (2016) with a 4.0 Comedy

Why Him? (2016) with a 3.5 Comedy

Hamlet (1948) with a 3.5 Drama

Rocky (1976) with a 5.0 Drama

The Conjuring 2 (2016) with a 4.0 Horror

"Prophecy, The (1995)" with a 4.0 Fantasy:Horror:Mystery

Relative Fear (1994) with a 5.0 Horror:Thriller

"Silence of the Lambs, The (1991)" with a 5.0 Crime:Horror:Thriller

"Conjuring, The (2013)" with a 5.0 Horror:Thriller

The Purge- Election Year (2016) with a 2.5 Action:Horror:Sci-Fi

Sharknado 4- The 4th Awakens (2016) with a 1.5

Action:Adventure:Horror:Sci-Fi

Mr. & Mrs. Smith (2005) with a 5.0 Comedy:Romance

Godzilla- Final Wars (Gojira- Fainaru uôzu)(2004) with a 2.5

Action:Adventure:Fantasy:Sci-Fi

Thor (2011) with a 5.0 Action:Adventure:Drama:Fantasy:IMAX

Thor- The Dark World (2013) with a 4.5 Action:Adventure:Fantasy:IMAX

Course of the experiment

- ▷ We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- ▷ In this experiment the genre that the user hates are animation, children and documentary
- ▷ Afterwards we must indicate what movies have the user rated

Results

1 "Wes Craven's New Nightmare (Nightmare on Elm Street Part 7- Freddy's Finale, A) (1994)" : Drama | Horror | Mystery | Thriller
2 Seven (a.k.a. Se7en) (1995) : Mystery | Thriller
3 Forrest Gump (1994) : Comedy | Drama | Romance | War
4 Beyond Bedlam (1993) : Drama | Horror
5 Bad Boys (1995) : Action | Comedy | Crime | Drama | Thriller
6 Mallrats (1995) : Comedy | Romance
7 "Shawshank Redemption, The (1994)" : Crime | Drama
8 "Mask, The (1994)" : Action | Comedy | Crime | Fantasy
9 Die Hard- With a Vengeance (1995) : Action | Crime | Thriller
10 True Lies (1994) : Action | Adventure | Comedy | Romance | Thriller
11 Casino (1995) : Crime | Drama
12 "Fugitive, The (1993)" : Thriller
13 Jurassic Park (1993) : Action | Adventure | Sci-Fi | Thriller
14 Batman (1989) : Action | Crime | Thriller
15 Pulp Fiction (1994) : Comedy | Crime | Drama | Thriller
16 Assassins (1995) : Action | Crime | Thriller
17 Castle Freak (1995) : Horror
18 Frisk (1995) : Drama
19 "Young Poisoner's Handbook, The (1995)" : Crime | Drama
20 Waterworld (1995) : Action | Adventure | Sci-Fi
21 Star Wars- Episode IV - A New Hope (1977) : Action | Adventure | Sci-Fi
22 Stalingrad (1993) : Drama | War
23 Demolition Man (1993) : Action | Adventure | Sci-Fi
24 Interview with the Vampire- The Vampire Chronicles (1994) : Drama | Horror
25 Léon- The Professional (a.k.a. The Professional) (Léon) (1994) : Action | Crime | Drama | Thriller
26 RoboCop 3 (1993) : Action | Crime | Drama | Sci-Fi | Thriller
27 Naked Gun 3 1/3- The Final Insult (1994) : Action | Comedy
28 Braveheart (1995) : Action | Drama | War

8281 Killer's Kiss (1955) : Crime | Film-Noir
8282 "Asphalt Jungle, The (1950)" : Crime | Film-Noir
8283 "Big Country, The (1958)" : Romance | Western
8284 Detour (1945) : Crime | Film-Noir
8285 "Clain, The (2000)" : Romance | Western
8286 House of Games (1987) : Crime | Film-Noir | Mystery | Thriller
8287 "Cotton Club, The (1984)" : Crime | Musical
8288 Rio Grande (1950) : Romance | Western
8289 From Justin to Kelly (2003) : Musical | Romance
8290 You Only Live Once (1937) : Crime | Film-Noir
8291 "Killers, The (1946)" : Crime | Film-Noir
8292 Easter Parade (1948) : Musical | Romance
8293 "Pure Formality, A (Pura formalità, Una) (1994)" : Crime | Film-Noir | Mystery | Thriller
8294 "Trois, Le (Hole, The) (Night Watch, The) (1960)" : Crime | Film-Noir
8295 The Great Train Robbery (1903) : Crime | Western
8296 "Misérables, Les (1998)" : Crime | Drama | Romance | War
8297 Johnny Eager (1942) : Crime | Drama | Film-Noir | Romance
8298 "Maltese Falcon, The (1941)" : Film-Noir | Mystery
8299 Shenandoah (1965) : Drama | War | Western
8300 The Alamo (2004) : Drama | War | Western
8301 Gilda (1946) : Drama | Film-Noir | Mystery | Romance
8302 In a Lonely Place (1950) : Drama | Film-Noir | Mystery | Romance
8303 "Very Long Engagement, A (un long dimanche de fiançailles) (2004)" : Drama | Mystery | Romance | War
8304 Brick (2005) : Crime | Drama | Film-Noir | Mystery
8305 Calamity Jane (1953) : Musical | Western
8306 Love Me Tender (1956) : Musical | Western
8307 Laura (1944) : Crime | Film-Noir | Mystery
8308 "Big Sleep, The (1946)" : Crime | Film-Noir | Mystery
8309 South Pacific (1958) : Musical | Romance | War
8310 Oklahoma! (1955) : Musical | Romance | Western

Comments

- ▶ As we can see the recommendations made by the hybrid system are recommendations that in no case belong to any of the genres that the user hates
- ▶ As expected, the final recommendation does not contain any movie that belongs to any genre that the user hates, this is due to knowledge filtering

Experiment 4. Recommendations with Hated Genres problem

Goal

- In this experiment we want to demonstrate how the hybrid system solves the problem of containing movies in the recommendation that belong to genres that the user hates

Assumptions

- In this experiment the preprocessing of the data doesn't matter, the result will be the same it only affects to the time of the execution of our algorithm.
- To check if the problem of recommendations with genres that the user hates is solved by our hybrid system, we do not need to make changes in the dataset.
- We don't have to choose specific ratings either, we simply have to make ratings and indicate the genres the user hates

Assumptions

► Genres hated by the user:

Animation

Children

Documentary

► User ratings:

Comedy Central Roast of David Hasselhoff (2010) with a 2.0 Comedy

Gabriel Iglesias- Hot and Fluffy (2007) with a 1.5 Comedy

Ghost Graduation (2012) with a 3.0 Comedy

Jackass Presents- Bad Grandpa (2013) with a 4.5 Comedy

Grown Ups 2 (2013) with a 5.0 Comedy

Ocho apellidos vascos (2014) with a 5.0 Comedy

Too Many Cooks (2014) with a 2.0 Comedy

War Dogs (2016) with a 4.5 Comedy

Bad Moms (2016) with a 4.0 Comedy

Why Him? (2016) with a 3.5 Comedy

Hamlet (1948) with a 3.5 Drama

Rocky (1976) with a 5.0 Drama

The Conjuring 2 (2016) with a 4.0 Horror

"Prophecy, The (1995)" with a 4.0 Fantasy:Horror:Mystery

Relative Fear (1994) with a 5.0 Horror:Thriller

"Silence of the Lambs, The (1991)" with a 5.0 Crime:Horror:Thriller

"Conjuring, The (2013)" with a 5.0 Horror:Thriller

The Purge- Election Year (2016) with a 2.5 Action:Horror:Sci-Fi

Sharknado 4- The 4th Awakens (2016) with a 1.5

Action:Adventure:Horror:Sci-Fi

Mr. & Mrs. Smith (2005) with a 5.0 Comedy:Romance

Godzilla- Final Wars (Gojira- Fainaru uôzu)(2004) with a 2.5

Action:Adventure:Fantasy:Sci-Fi

Thor (2011) with a 5.0 Action:Adventure:Drama:Fantasy:IMAX

Thor- The Dark World (2013) with a 4.5 Action:Adventure:Fantasy:IMAX

Course of the experiment

- ▷ We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- ▷ In this experiment the genre that the user hates are animation, children and documentary
- ▷ Afterwards we must indicate what movies have the user rated

Results

1 "Wes Craven's New Nightmare (Nightmare on Elm Street Part 7- Freddy's Finale, A) (1994)" : Drama | Horror | Mystery | Thriller
 2 Seven (a.k.a. Se7en) (1995) : Mystery | Thriller
 3 Forrest Gump (1994) : Comedy | Drama | Romance | War
 4 Beyond Bedlam (1993) : Drama | Horror
 5 Bad Boys (1995) : Action | Comedy | Crime | Drama | Thriller
 6 Mallrats (1995) : Comedy | Romance
 7 "Shawshank Redemption, The (1994)" : Crime | Drama
 8 "Mask, The (1994)" : Action | Comedy | Crime | Fantasy
 9 Die Hard- With a Vengeance (1995) : Action | Crime | Thriller
 10 True Lies (1994) : Action | Adventure | Comedy | Romance | Thriller
 11 Casino (1995) : Crime | Drama
 12 "Fugitive, The (1993)" : Thriller
 13 Jurassic Park (1993) : Action | Adventure | Sci-Fi | Thriller
 14 Batman (1989) : Action | Crime | Thriller
 15 Pulp Fiction (1994) : Comedy | Crime | Drama | Thriller
 16 Assassins (1995) : Action | Crime | Thriller
 17 Castle Freak (1995) : Horror
 18 Frisk (1995) : Drama
 19 "Young Poisoner's Handbook, The (1995)" : Crime | Drama
 20 Waterworld (1995) : Action | Adventure | Sci-Fi
 21 Star Wars- Episode IV - A New Hope (1977) : Action | Adventure | Sci-Fi
 22 Stalingrad (1993) : Drama | War
 23 Demolition Man (1993) : Action | Adventure | Sci-Fi
 24 Interview with the Vampire- The Vampire Chronicles (1994) : Drama | Horror
 25 Léon- The Professional (a.k.a. The Professional) (Léon) (1994) : Action | Crime | Drama | Thriller
 26 RoboCop 3 (1993) : Action | Crime | Drama | Sci-Fi | Thriller
 27 Naked Gun 3 1/3- The Final Insult (1994) : Action | Comedy
 28 Braveheart (1995) : Action | Drama | War

8281 Killer's Kiss (1955) : Crime | Film-Noir
 8282 "Asphalt Jungle, The (1950)" : Crime | Film-Noir
 8283 "Big Country, The (1958)" : Romance | Western
 8284 Detour (1945) : Crime | Film-Noir
 8285 "Claim, The (2000)" : Romance | Western
 8286 House of Games (1987) : Crime | Film-Noir | Mystery | Thriller
 8287 "Cotton Club, The (1984)" : Crime | Musical
 8288 Rio Grande (1950) : Romance | Western
 8289 From Justin to Kelly (2003) : Musical | Romance
 8290 You Only Live Once (1937) : Crime | Film-Noir
 8291 "Killers, The (1946)" : Crime | Film-Noir
 8292 Easter Parade (1948) : Musical | Romance
 8293 "Pure Formality, A (Pura formalità, Una) (1994)" : Crime | Film-Noir | Mystery | Thriller
 8294 "Trois, Le (Hole, The) (Night Watch, The) (1960)" : Crime | Film-Noir
 8295 The Great Train Robbery (1903) : Crime | Western
 8296 "Misérables, Les (1998)" : Crime | Drama | Romance | War
 8297 Johnny Eager (1942) : Crime | Drama | Film-Noir | Romance
 8298 "Maltese Falcon, The (1941)" : Film-Noir | Mystery
 8299 Shenandoah (1965) : Drama | War | Western
 8300 The Alamo (2004) : Drama | War | Western
 8301 Gilda (1946) : Drama | Film-Noir | Mystery | Romance
 8302 In a Lonely Place (1950) : Drama | Film-Noir | Mystery | Romance
 8303 "Very Long Engagement, A (un long dimanche de fiançailles) (2004)" : Drama | Mystery | Romance | War
 8304 Brick (2005) : Crime | Drama | Film-Noir | Mystery
 8305 Calamity Jane (1953) : Musical | Western
 8306 Love Me Tender (1956) : Musical | Western
 8307 Laura (1944) : Crime | Film-Noir | Mystery
 8308 "Big Sleep, The (1946)" : Crime | Film-Noir | Mystery
 8309 South Pacific (1958) : Musical | Romance | War
 8310 Oklahoma! (1955) : Musical | Romance | Western

Comments

- ▶ As we can see the recommendations made by the hybrid system are recommendations that in no case belong to any of the genres that the user hates
- ▶ As expected, the final recommendation does not contain any movie that belongs to any genre that the user hates, this is due to knowledge filtering

Scalability/Time problem experiments

Experiment 5. Recommending a user with 15 ratings (without Dataset preprocessing)

Goal

- In this experiment we want to study how much time does it take for our algorithm to preprocess our dataset

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same. We are making this experiment without preprocessing nothing
- We are making this experiment without preprocessing our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are horror and drama
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times.

Results

211,759 s	462,662 s	304,991 s	255,872 s
207,676 s	210,713 s	371,606 s	
209,615 s	209,082 s	209,444 s	

Mean -> 265,342 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 15 ratings and 2 hated genres is 265,342 seconds
- ▶ So, if we subtract from the result the result of experiment 6 which also have 15 ratings and 2 heated genres, we will get the average time of preprocessing the dataset.
- ▶ In summary, the time of preprocessing the dataset is: 243,363 seconds

Experiment 6. Recommending a user with 15 ratings

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 15 in particular without the preprocessing of the cosine similarity matrix and the matrix of genres.
- But with the preprocess of the Dataset.

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment without preprocessing our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are fantasy and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times.

Results

22,272 s	22,732 s	21,594 s	23,157 s
21,305 s	21,409 s	22,085 s	
22,546 s	21,305 s	21,380 s	

Mean -> 21,9785 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 15 ratings and 2 hated genres is 21,9785 seconds

Experiment 7. Recommending a user with 30 ratings

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 30 in particular without the preprocessing of the cosine similarity matrix and the matrix of genres.
- But with the preprocess of the Dataset.

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment without preprocessing our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are fantasy and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times.

Results

24,047 s	23,651 s	24,164 s	26,560 s
23,691 s	23,532 s	25,106 s	
22,098 s	25,749 s	21,347 s	

Mean -> 23,9945 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 30 ratings and 2 hated genres is 23,9945 seconds

Experiment 8. Recommending a user with 60 ratings

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 60 in particular without the preprocessing of the cosine similarity matrix and the matrix of genres.
- But with the preprocess of the Dataset.

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment without preprocessing our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are fantasy and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times.

Results

28,882 s	28,551 s	28,272 s	28,277 s
28,448 s	28,629 s	26,431 s	
28,301 s	29,159 s	29,268 s	

Mean -> 28,4218 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 60 ratings and 2 hated genres is 28,4218 seconds

Experiment 9. Recommending a user with 120 ratings

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 120 in particular without the preprocessing of the cosine similarity matrix and the matrix of genres.
- But with the preprocess of the Dataset.

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment without preprocessing our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are fantasy and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times.

Results

35,576 s	35,507 s	33,551 s	34,976 s
35,596 s	37,976 s	34,502 s	
35,393 s	37,862 s	32,690 s	

Mean -> 35,3629 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 120 ratings and 2 hated genres is 35,3629 seconds

Experiment 10. Recommending a user with 240 ratings

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 240 in particular without the preprocessing of the cosine similarity matrix and the matrix of genres.
- But with the preprocess of the Dataset.

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment without preprocessing our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- ▶ We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- ▶ In this experiment the genre that the user hates are fantasy and thriller
- ▶ Afterwards we must indicate what movies have the user rated
- ▶ Once we have the algorithm input ready, we run the algorithm 10 times.

Results

47,759 s	49,844 s	46,281 s	46,931 s
50,296 s	48,500 s	50,300 s	
49,620 s	50,714 s	51,698 s	

Mean -> 49,1943 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 240 ratings and 2 hated genres is 49,1943 seconds

Experiment 11. Recommending a user with 15 ratings (with preprocessing)

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 15 in particular with the preprocessing of the dataset, the cosine similarity matrix and the matrix of genres

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment with the preprocessing of our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are drama and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times

Results

3,662 s	3,662 s	3,662 s	3,662 s
3,662 s	3,662 s	3,662 s	
3,662 s	3,662 s	3,662 s	

Mean -> 3,662 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 15 ratings and 2 hated genres is 3,662 seconds

Experiment 12. Recommending a user with 30 ratings (with preprocessing)

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 30 in particular with the preprocessing of the dataset, the cosine similarity matrix and the matrix of genres

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment with the preprocessing of our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are drama and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times

Results

5,884 s	5,884 s	5,884 s	5,884 s
5,884 s	5,884 s	5,884 s	
5,884 s	5,884 s	5,884 s	

Mean -> 5,884 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 30 ratings and 2 hated genres is 5,884 seconds

Experiment 13. Recommending a user with 60 ratings (with preprocessing)

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 60 in particular with the preprocessing of the dataset, the cosine similarity matrix and the matrix of genres

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment with the preprocessing of our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are drama and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times

Results

9,552 s	9,552 s	9,552 s	9,552 s
9,552 s	9,552 s	9,552 s	
9,552 s	9,552 s	9,552 s	

Mean -> 9,552 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 60 ratings and 2 hated genres is 9,552 seconds

Experiment 14. Recommending a user with 120 ratings (with preprocessing)

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 120 in particular with the preprocessing of the dataset, the cosine similarity matrix and the matrix of genres

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment with the preprocessing of our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are drama and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times

Results

16,179 s	16,179 s	16,179 s	16,179 s
16,179 s	16,179 s	16,179 s	
16,179 s	16,179 s	16,179 s	

Mean -> 16,179 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 120 ratings and 2 hated genres is 16,179 seconds

Experiment 15. Recommending a user with 240 ratings (with preprocessing)

Goal

- In this experiment we want to study the scalability of our system, that is, to study the execution time for a given number of ratings, 240 in particular with the preprocessing of the dataset, the cosine similarity matrix and the matrix of genres

Assumptions

- To test the runtime, we did not need to make any changes to the dataset, neither to the user's genres that the user hates nor to the user's ratings, that is, during the 10 tests we performed for this experiment the input data were the same
- We are making this experiment with the preprocessing of our data
- The user's hated genres and ratings can be observed in the files hatedGenres.txt and userRatings.txt

Course of the experiment

- We start the algorithm, simulating the interaction of a user, so we must enter the genres that the user hates and the ratings that the user needs to give
- In this experiment the genre that the user hates are drama and thriller
- Afterwards we must indicate what movies have the user rated
- Once we have the algorithm input ready, we run the algorithm 10 times

Results

30,580 s	30,580 s	30,580 s	30,580 s
30,580 s	30,580 s	30,580 s	
30,580 s	30,580 s	30,580 s	

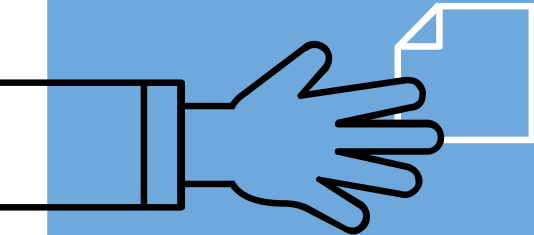
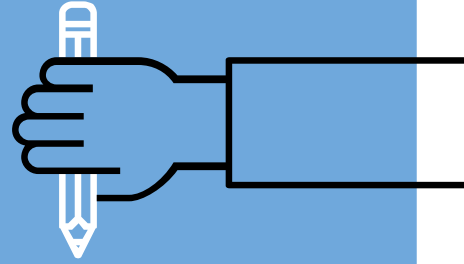
Mean -> 30,580 seconds

Comments

- ▶ As we can see the average run time of the algorithm with 240 ratings and 2 hated genres is 30,580 seconds

CONCLUSIONS

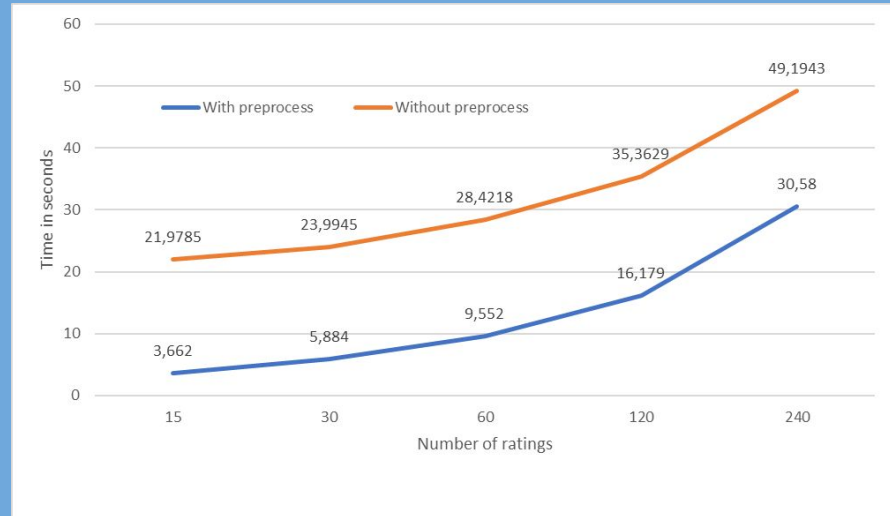
ARE OUR SYSTEM GOOD?

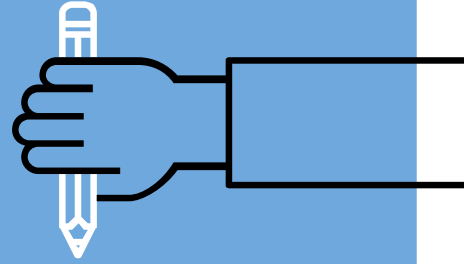


ACHIEVING GOALS

- ▷ GOOD RECOMMENDATIONS
- ▷ FIX OUR MAIN DISADVANTAGES
- ▷ PRE PROCESSING DATASET TIME: 243,363 seconds

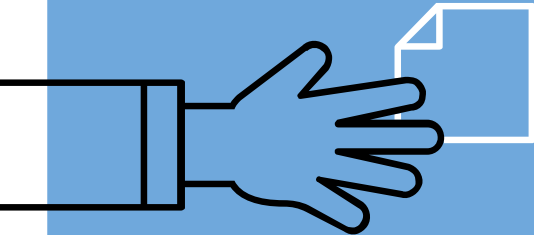
- ▷ TIME GRAPHIC:



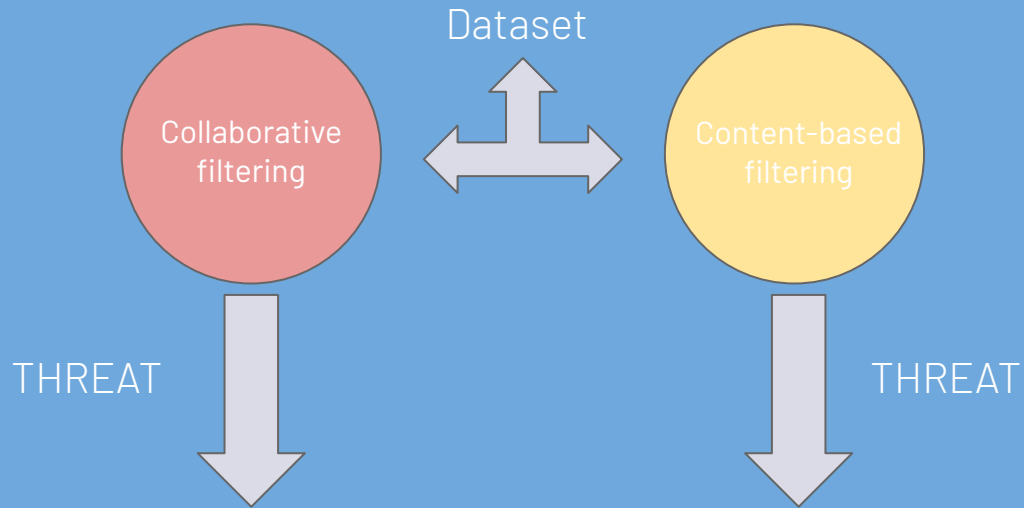


POSSIBLE FUTURE UPGRADES

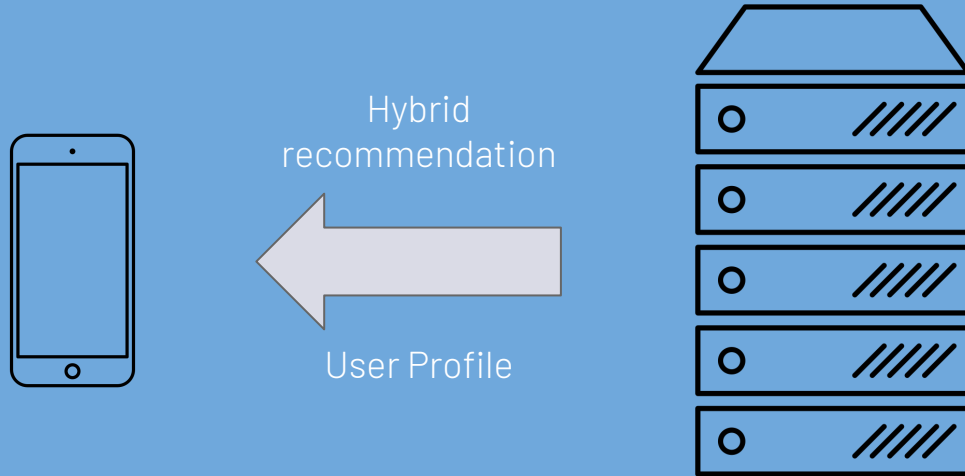
WHAT'S NEXT?




Future improvements



Future improvements



Future improvements


	Q1	Q2	Q3
	1	0	1

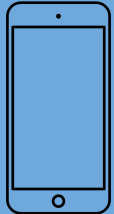


Questions\ Genres	G1	G2	G3	G4
Q1	1	0	0	1
Q2	0	1	0	0
Q3	1	0	1	0




Session Profile

	Q1	Q2	Q3	Q4
	X	X	X	X



User Profile

	G1	G2	G3	G4
	X	X	X	X

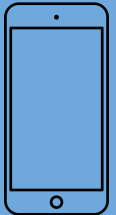
+

Session Profile

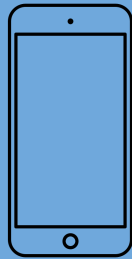
	G1	G2	G3	G4
	X	X	X	X

User Profile

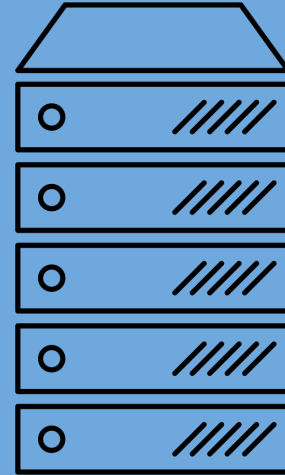
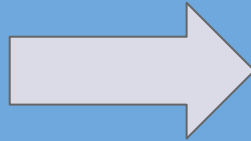
	G1	G2	G3	G4
	X	X	X	X



Future improvements



Questions answered



Bibliography

- [1] Barragas A., Costa-Montenegro E., Burguillo J., Rey-López M., Mikic-Fonte F., Peleteiro A. (2010) *A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition*
- [2] Tran, T. (2007). *Combining Collaborative Filtering and Knowledge-Based Approaches for Better Recommendation Systems*
- [3] Gupta, M., Thakkar, A., Gupta, V., Pratap Singh, D. (2020). *Movie recommender system using collaborative filtering*
- [4] Bell, R., Koren, Y. (2007). *Improved neighborhood-based collaborative filtering*
- [5] Robin Van Meteren, Maarten Van Someren (2017). *Using Content-based filtering for recommendation*
- [6] GOOGLE DEVELOPERS. Recommendation Systems, Content-based Filtering.
<<https://developers.google.com/machine-learning/recommendation/content-based/basics>> [Accessed: november 13rd, 2021]
- [7] “26 Content based Recommender Systems” My Course, Saaed Aghabozorgi, IBM Cooperation
<<https://www.youtube.com/watch?v=YMZmLx-AUvY>> [Accessed: february 6th, 2022]
- [8] Criado Gonzalez, M. (2018). *Análisis e implementación de un sistema de recomendación para la lista de la compra*
- [9] “2.1.1. Hybrid Recommendation Systems” AI-First <<https://www.youtube.com/watch?v=5VyIUiv5zsc>> [Accessed: november 5th, 2021]
- [10] “2.1.4. Lab: Designing a Hybrid Knowledge-based Recommendation Systems” AI-First
<[youtube.com/watch?v=nYo90s8NDk0&t=259s](https://www.youtube.com/watch?v=nYo90s8NDk0&t=259s)> [Accessed: november 14th, 2021]
- [11] Melville, P., Sindhvani, V. (2010). *Recommender Systems*

THANKS!

Any questions?

You can find us at:



misintaxis@gmail.com



misintaxis.com



[misintaxis](https://www.linkedin.com/company/misintaxis)



[misintaxis](https://www.instagram.com/misintaxis)



[misintaxis](https://twitter.com/misintaxis)



[misintaxis](https://www.tiktok.com/@misintaxis)

