Movie Recommendation Systems

Submitted by:

Mohit Singh (MCS21002)
Aakanksha Bhardwaj (MCS21013)

Under the supervision of:
Dr. Chandranath Adak

विद्या ददाति विनयं विनयाद् याति पात्रताम्

Department of PG-CSE/IT
Indian Institute of Information Technology, Lucknow

December 2021

TABLE OF CONTENTS

<u>CERTIFICATE</u>

This is to certify that the work titled “Movie Recommendation Systems” submitted by “Mohit Singh and Aakanksha Bhardwaj”
in partial fulfilment for the award of degree of Master of Technology of Indian Institute of Information Technology, Lucknow has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor

Dr. Chandranath Adak

(Asst. Professor IIIT Lucknow)

LIST OF FIGURES

Summary

- ❏ Personalized User Based Movie Recommendation System
- ❏ Searching for a movie to watch is a time-consuming task, there is no recommendation system which recommends movie based on personalized experience and IMDB ratings.
- ❏ To create a model which is based on user convenience
- ❏ Recommendation system can be linked with social networking apps. It can then be used to recommend movie based on the preferences of people linked to your social network.

Chapter 1

Introduction

A recommendation system (also known as platform or engine), is a subset of the information filtering system that allows the user to assess "ratings" or "preferences". These are mainly used in commercial and business applications.

Recommended systems are used in diverse number of ways and are normally recognized as playlist generators for video services and streaming services such as Netflix, YouTube and Spotify. These systems can use a single input such as music or more than one input on platforms such as news, books and search queries. There are also common recommendation systems for specific topics such as restaurants,online dating, experts, collaborators and financial services have also been made.

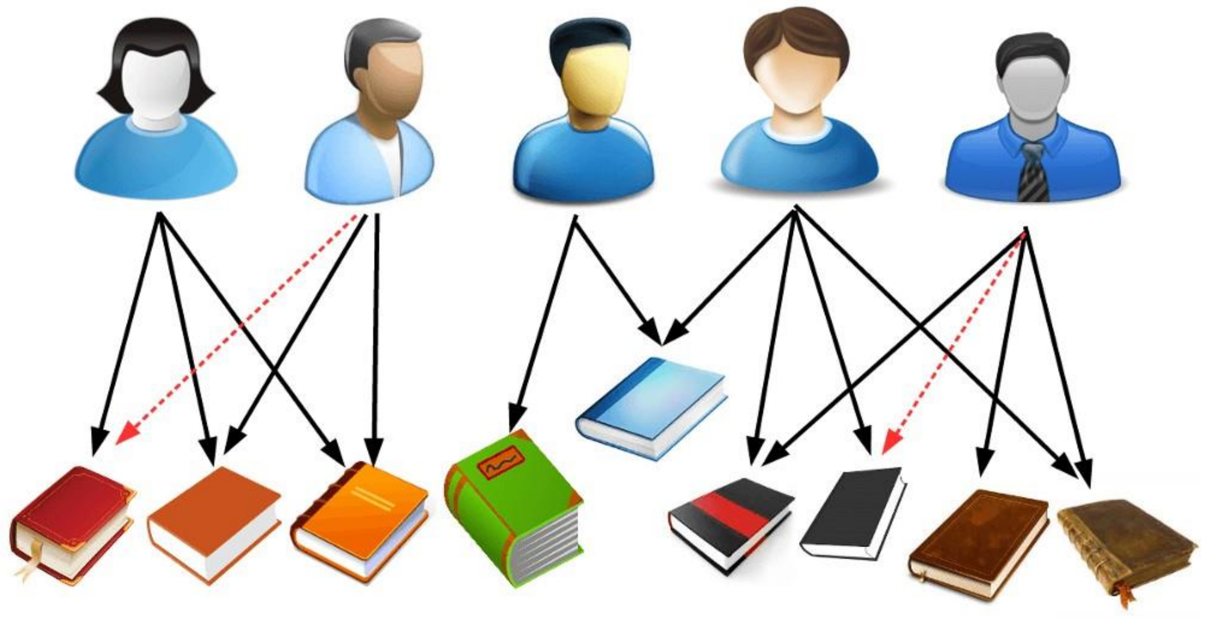


Fig.1

(overview of recommendation systems)

Chapter 2

Background Study

Recommended systems normally use collaborative filtering and content-based filtering (also called personality-based totally procedures), as well as other systems along with understanding-based totally structures. Collaborative filtering methods construct a model from the person's beyond conduct (given or chosen and / or numerical scores) and similar selections taken via different customers. This version is used to estimate the object (or score for the object) that the person is interested in. Content-based filtering approaches use an array of discrete, pre-tag attributes of an item to advocate additional gadgets with comparable attributes. Cutting-edge recommendation systems typically integrate one or extra structures into a hybrid device.

Recommended structures are a useful option for searching algorithms because they help customers search for gadgets they can't find. Note, sponsor structures are implemented the usage of search engines that constitute nontraditional statistics.

There are 4 types of filtering techniques:

1. Collaborative filtering
2. Contect based filtering
3. Multicriteria filtering
4. Hybrid filtering

1.Collaborative Filter:

One approach to the layout of broadly used recommendation systems is collaborative filtering. Collaborative filtering depends on what humans in the past receive within the destiny, and what they prefer within the past. The machine makes pointers to various customers or merchandise using most effective facts approximately score profiles. Through figuring out fellow customers / gadgets with the same score history as the present day consumer or item, they make hints using this surroundings. Collaborative filtering methods are labelled as reminiscence-primarily based and versionbased. A famous instance of memory-based tactics is the person-based totally algorithm, one of the model-based totally processes is the kernel-mer advice.

The main benefit of a collaborative filtering mechanism is that it does not depend upon system analytical cloth and might therefore recommend complicated gadgets such as movies without the need to "recognize" the cloth. Many algorithms have been used to degree user similarity or similarity in systems recommending gadgets. As an example, the k-nearest neighbor (k-nn) method and the pearson correlation previously applied by using allen.

While constructing a version from customer behaviour, there is a distinction among express and implicit sorts of records collection.

Examples of exact statistics series consist of the following:

- asking the person to charge an item on the sliding scale.

- ask the person to look.
- asking the person to keep the least favoured gadgets from the favourites.
- objects offer the person two items and ask them to select one of the suitable ones.
- ask the consumer to create a list of items she likes / see (see Rousseau’s taxonomy or other comparable techniques).

Examples of implicit records collection include the following:

- an outline of what the person sees within the on line save.
- retaining a file of gadgets that the client purchases on-line.
- get a list of gadgets that the consumer has heard or seen on their pc.

Reading a social person's social networks and finding comparable likes and dislikes. Extraordinary filtering techniques frequently suffer from three problems: cold onset, scalability and sparsity.

- bloodless start: for a new consumer or object, there isn't always sufficient records to make definitive tips.
- scalability: in lots of environments where this gadget makes recommendations, there are tens of millions of customers and merchandise. Consequently, a large amount of computational power is often required to calculate the tips.
- sparsity: the variety of products sold on essential e-trade sites is massive. Only the most active users are given the popularity of a small subset of the whole database. Consequently, even the most famous items have a very low rating.

One of the high-quality examples of collaborative filtering is item-to-item collaborative filtering (x even people who purchase y), an algorithm popularized by way of amazon.com's recommendation machine.

Most social networks first use a collaborative clear out to recommend new buddies, corporations and other social connections through analyzing the community of connections among the user and their friends. Collaborative filtering continues to be used as part of the hybrid machine.

2.Content Based Filtering:

Some other common technique while designing recommendation structures is content material-primarily based filtering. The content material-primarily based filtering technique depends on the description of the object and the profile of the person preferences. Those techniques are satisfactory desirable to situations where the name (call, place, description, and so forth.) of an object is thought, however no longer at the consumer. Content material-primarily based tips treat the advice as a user-particular type problem and examine the category for user likes and dislikes based totally at the traits of an item.

On this gadget, key phrases are used to describe gadgets, and a consumer profile is created to signify what sort of object this user likes. In other words, these algorithms attempt to endorse gadgets that are just like those that customers have favored or are presently investigating. It frequently does now not rely upon person sign-in regulations to create temporary profiles. In particular, the one of a kind candidate objects are in comparison to the items the user has taken within the beyond and the objects which are quality matched are recommended. This method is rooted in statistics retrieval and statistics filtering research.

To create a person profile, the device primarily makes a speciality of two varieties of facts:

1. Version of patron choice.

2. History of person interplay with the recommendation machine.

Usually, these strategies use object profiles (i.E. A set of discrete attributes and attributes) to perceive items in the system. To seize the residences of the items inside the gadget, the object illustration algorithm is carried out. The maximum widely used algorithm is tf-idf illustration (additionally known as vector space illustration). Creates a contentbased totally profile of customers based at the waiter vector of device properties. Weight reflects the importance of every characteristic to the user and may be calculated from for my part rated material vectors using one-of-a-kind methods. Simple techniques use common values of the rated object vector, while different superior strategies use device learning techniques which include bayesian class, cluster evaluation, choice trees, and synthetic neural networks.

A main hassle with content-based filtering is whether or not the device can research user options from user movements about the content material supply and use them in different content types. While the machine is constrained to recommending the identical sort of content that the user is already the use of, different service kinds are much less in all likelihood to price the advice device than endorsed by other offerings. Ken. As an example, it's far useful to endorse information testimonies primarily based on browsing information, however it's far greater useful whilst tune, videos, products, discussions and so forth. Can be retrieved from extraordinary services.

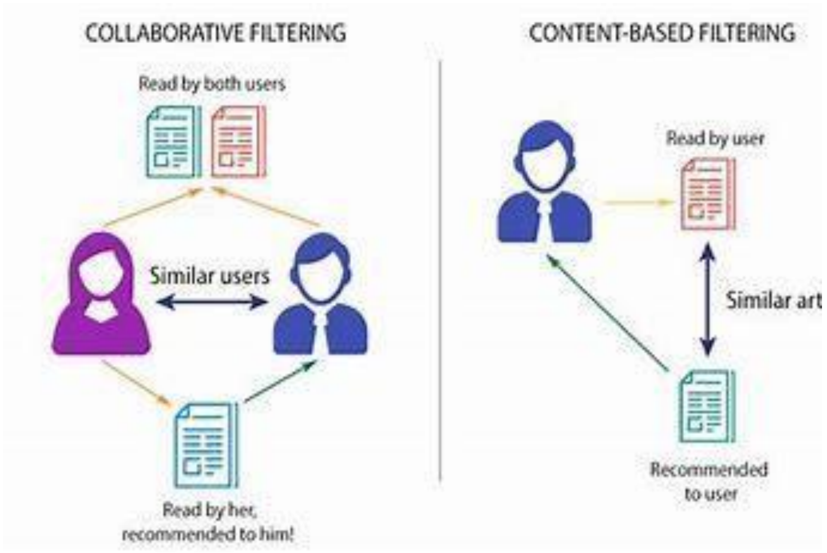


Fig.2

(Difference between content based and collaborative filtering)

3.Multi-criteria recommendation system:

Multi-criteria recommendation systems (mcrs) may be defined as advice systems that embed priority statistics on more than one scales. Rather than growing unmarried-criteria advice techniques primarily based on the user's average choice for the item i, these systems attempt to estimate the rating for unexplained items by using the use of desire information on a couple of criteria that have an impact on the fee. Most researchers see mcrs as a multistandards decision (mcdm) hassle and apply mcdm strategies and techniques to implement mcrs structures.

4.Hybrid recommendation machine:

Maximum advice structures now use a hybrid technique, including collaborative recommendation filtering, contentbased filtering and different procedures. There's no reason not to hybridize a number of the identical technology. The hybrid technique may be carried out in many approaches: by way of setting apart and mixing content-primarily based and collaborative-orientated predictors; adding (and vice versa) content material-based abilties to a collaborative-oriented method; or through incorporating guidelines right into a version. Occasional research examine the overall performance of hybrids with natural collaborative and content-based totally strategies, and show that hybrid strategies can offer extra accurate hints than pure processes. These techniques can be used to conquer some common problems in advocated structures, including cold start and sparsity issues, as well as knowhow engineering obstacles in information-based processes.

An amazing instance of netflix hybrid advice device utilization. The internet site provides similar user-viewing and search conduct (eg, collaborative filtering), as well as presenting recommendation movies (content-primarily based filtering) by sharing functions with the maximum watched films.

Some hybridization strategies consist of:

- ied weight: statistically combining character advice rankings.
- switching: deciding on between endorsed additives and executing the chosen ones.
- mixed: a diffusion of tips are offered collectively to make tips.
- mixture function mixture: a single advice algorithm is given by means of combining features obtained from specific know-how sources.
- ag function strengthening: putting a characteristic or attribute that is a part of the input of the next technology.
- cascade: recommenders are given strict priority, whilst low-priority humans generally tend to have high scoring relationships.
- meta-level: a recommendation approach is applied and generates a few type of version

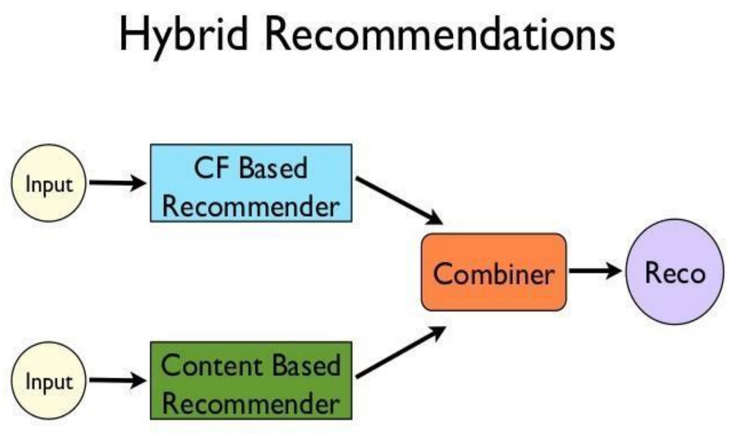


Fig.3

(Hybrid Recommendation Systems)

Chapter 3

Requirements

The **Software requirement** includes a python idle, here we have used anaconda and nvidia cuda-toolkit 9.0 for fast processing of the code.

The **hardware requirements** include a laptop/PC with anaconda installed, GPU present for faster execution.

The **functional requirements** include using python, database knowledge, pandas

Chapter 4

Detailed Design and Implementation

Classification based

First I will build a recommendation system, which is a classification model in which the feature and feature of the movie are used to determine whether the user liked the product. When new customers arrive, our classification gives the binary value of this user's preferred product or not, thus we can recommend the movie to the user.

Classifier:

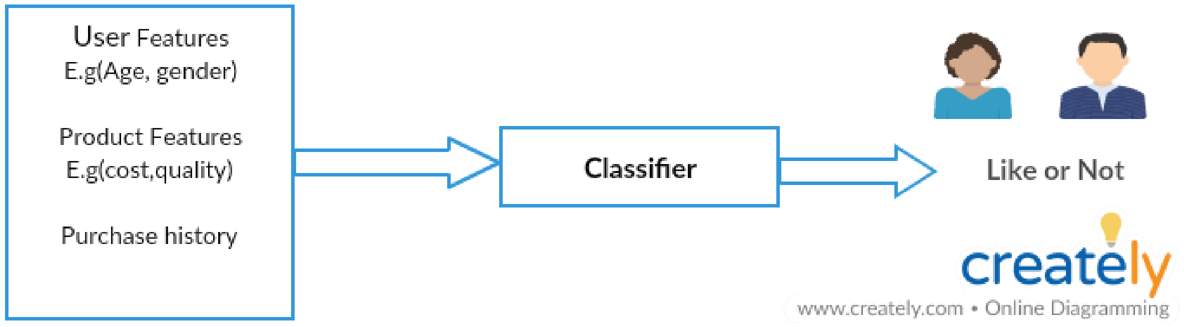


Fig.4

(Classification of Systems)

In the above example, our classifier can recommend a movie to the user based on whether or not the user can assign a binary value based on this input, using the edge, gender and product attributes, genres, ratings and movie history. Ken.

Collaborative Filter:
Collaborative filtering models based on what people like and things people like with other people's interests.

Two types of cooperative filter models,
I. The best neighborhood
II. The Matrix factor
I will briefly describe each method of collaborative filtering,

Nearest Neighbor Collaborative Filter:
These types of recommendation systems are recommended based on the nearest neighbor, the nearest neighbor approach is used to identify similar customers or similar products, It can be viewed in two ways, i.User based filter ii. Item Based Filtering

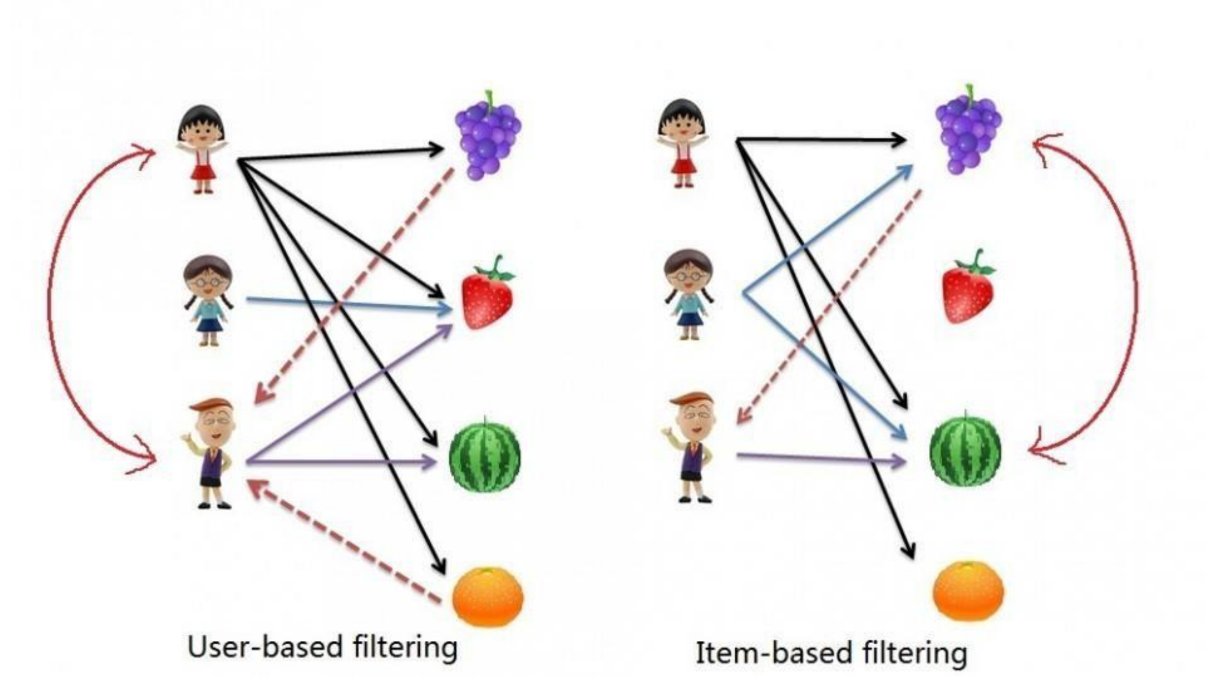


Fig.5

(Filtering Explained)

User-based collaborative filtering:

Find customers who have the same taste of products as the current consumer, similarity depends on consumer buying behavior, so we can refer items to the current consumer based on the buying behavior of the neighbors.

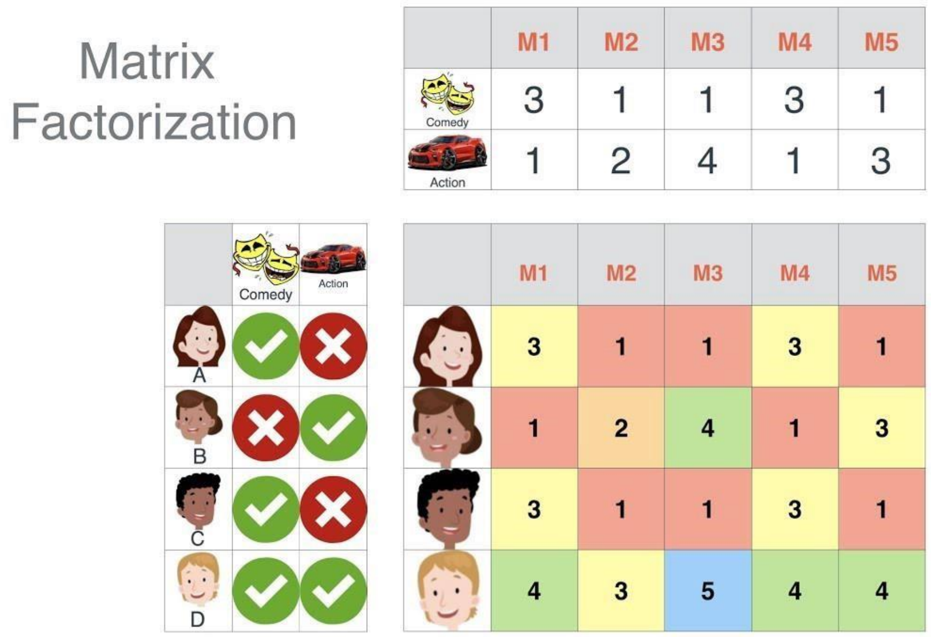
Item based collaborative filter:

Recommended items that resemble a buy-to-buy customer, similarity depends on purchase co-occurrences Items A and B are purchased by customers who are both X and Y.
Matrix factor:

This is basically an important technique in model-based collaborative filtering and matrix factor recommendation systems.

I need to explain the matrix factorization,

When a user feeds a particular movie (say they can rate one to five), this collection of opinions can be represented as a matrix. Each row represents each user, each column represents different movies. The matrix is so small that not



	M1	M2	M3	M4	M5
Comedy	3	1	1	3	1
Action	1	2	4	1	3

	Comedy	Action
A	✓	✗
B	✗	✓
C	✓	✗
D	✓	✓

	M1	M2	M3	M4	M5
	3	1	1	3	1
	1	2	4	1	3
	3	1	1	3	1
	4	3	5	4	4

Fig.6

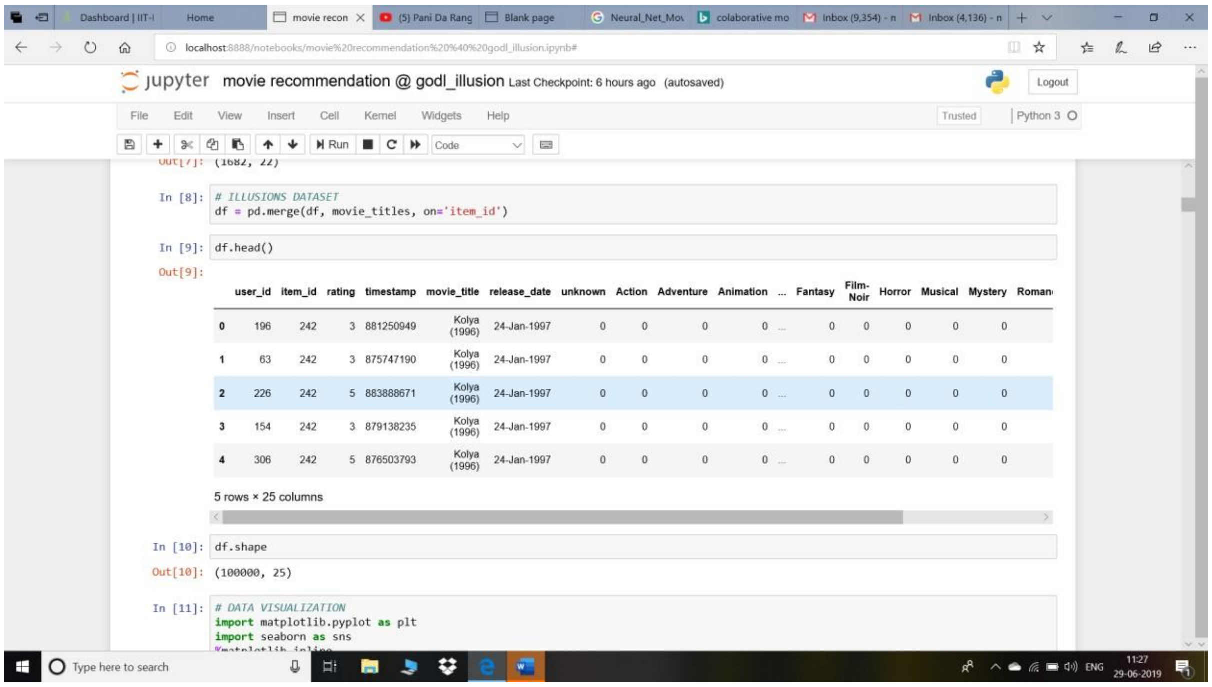
(Matrix Factorization)

Hybrid Recommendation systems:

Hybrid recommendation systems are a aggregate of collaborative and content-based suggestions and can be more effective. Hybrid processes may be implemented by way of setting apart content material-based totally and collaboration-primarily based predictors after which including them.

Chapter 5

Results and Snapshots of the Working Model



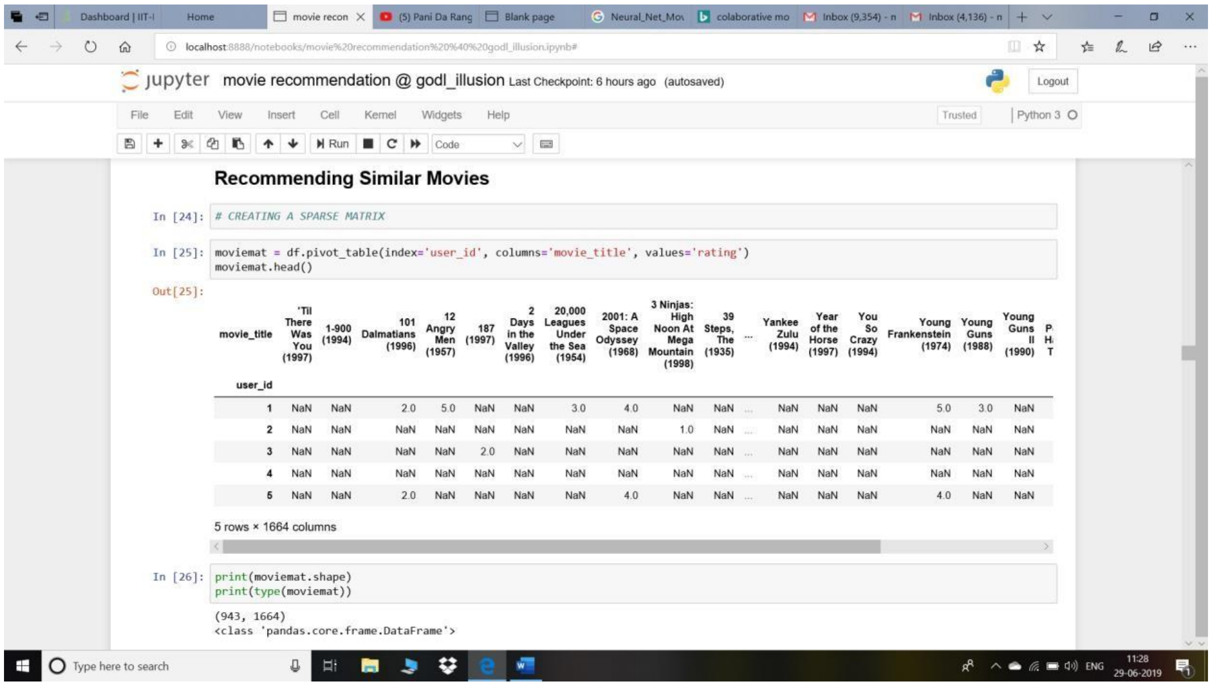
jupyter movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

```
Out[7]: (1682, 22)

In [8]: # ILLUSIONS DATASET
        df = pd.merge(df, movie_titles, on='item_id')

In [9]: df.head()

Out[9]:
```

	user_id	item_id	rating	timestamp	movie_title	release_date	unknown	Action	Adventure	Animation	...	Fantasy	Film-Noir	Horror	Musical	Mystery	Roman
0	196	242	3	881250949	Kolya (1996)	24-Jan-1997	0	0	0	0	...	0	0	0	0	0	
1	63	242	3	875747190	Kolya (1996)	24-Jan-1997	0	0	0	0	...	0	0	0	0	0	
2	226	242	5	883888671	Kolya (1996)	24-Jan-1997	0	0	0	0	...	0	0	0	0	0	
3	154	242	3	879138235	Kolya (1996)	24-Jan-1997	0	0	0	0	...	0	0	0	0	0	
4	306	242	5	876503793	Kolya (1996)	24-Jan-1997	0	0	0	0	...	0	0	0	0	0	

5 rows × 25 columns

```
In [10]: df.shape

Out[10]: (100000, 25)

In [11]: # DATA VISUALIZATION
         import matplotlib.pyplot as plt
         import seaborn as sns
```



jupyter movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

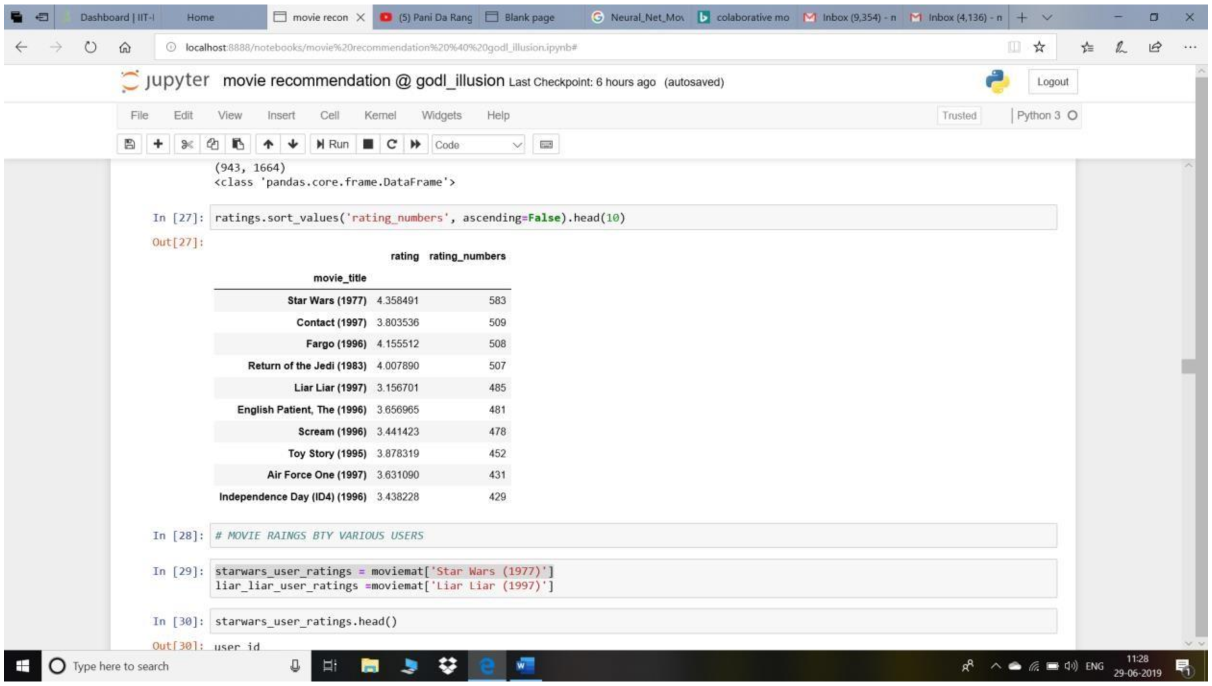
Recommending Similar Movies

```
In [24]: # CREATING A SPARSE MATRIX

In [25]: moviemat = df.pivot_table(index='user_id', columns='movie_title', values='rating')
         moviemat.head()

Out[25]:
```

movie_title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	39 Steps, The (1935)	...	Yankee Zulu (1994)	Year of the Horse (1997)	You So Crazy (1994)	Young Frankenstein (1974)	Young Guns (1988)	Young Guns II (1990)	P H T
user_id																		
1	NaN	NaN	2.0	5.0	NaN	NaN	3.0	4.0	NaN	NaN	...	NaN	NaN	NaN	5.0	3.0	NaN	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
5	NaN	NaN	2.0	NaN	NaN	NaN	NaN	4.0	NaN	NaN	...	NaN	NaN	NaN	4.0	NaN	NaN	

5 rows × 1664 columns

```
In [26]: print(moviemat.shape)
         print(type(moviemat))

(943, 1664)
<class 'pandas.core.frame.DataFrame'>
```



jupyter movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

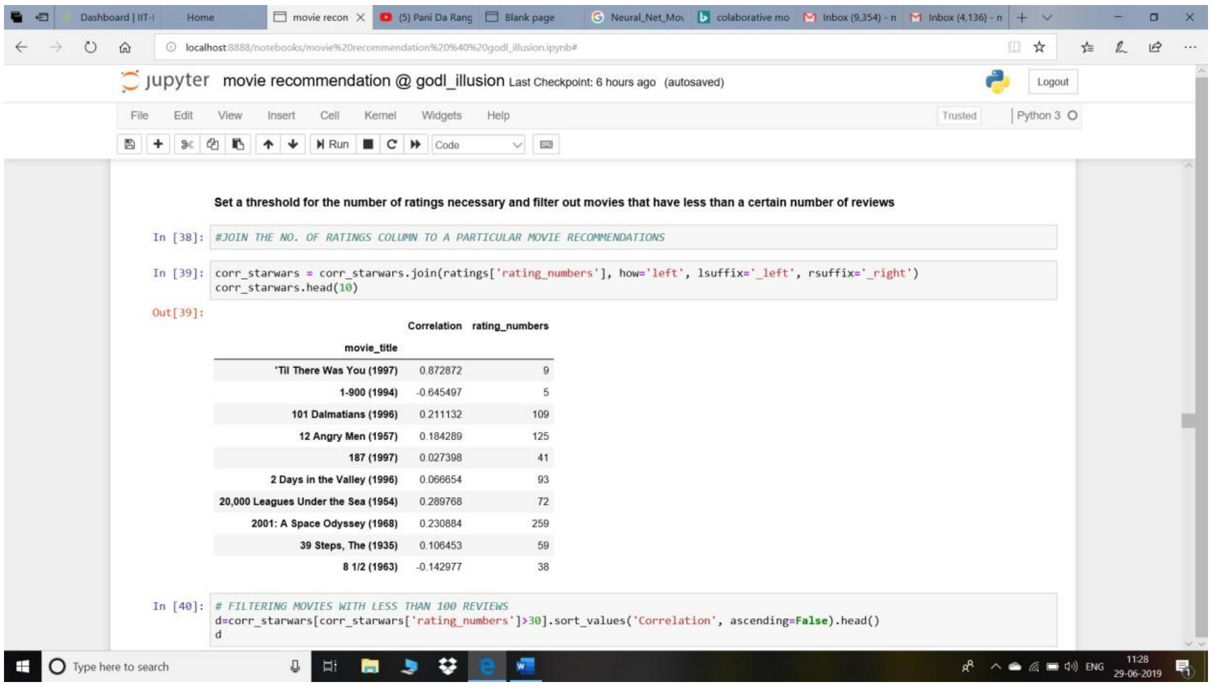
```
(943, 1664)
<class 'pandas.core.frame.DataFrame'>
```

```
In [27]: ratings.sort_values('rating_numbers', ascending=False).head(10)
```

Out[27]:

movie_title	rating	rating_numbers
Star Wars (1977)	4.358491	583
Contact (1997)	3.803536	509
Fargo (1996)	4.155512	508
Return of the Jedi (1983)	4.007890	507
Liar Liar (1997)	3.156701	485
English Patient, The (1996)	3.656965	481
Scream (1996)	3.441423	478
Toy Story (1995)	3.878319	452
Air Force One (1997)	3.631090	431
Independence Day (ID4) (1996)	3.438228	429

```
In [28]: # MOVIE RAINGS BTY VARIOUS USERS

In [29]: starwars_user_ratings = moviemat['Star Wars (1977)']
         liar_liar_user_ratings =moviemat['Liar Liar (1997)']

In [30]: starwars_user_ratings.head()

Out[30]: user_id
```



jupyter movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

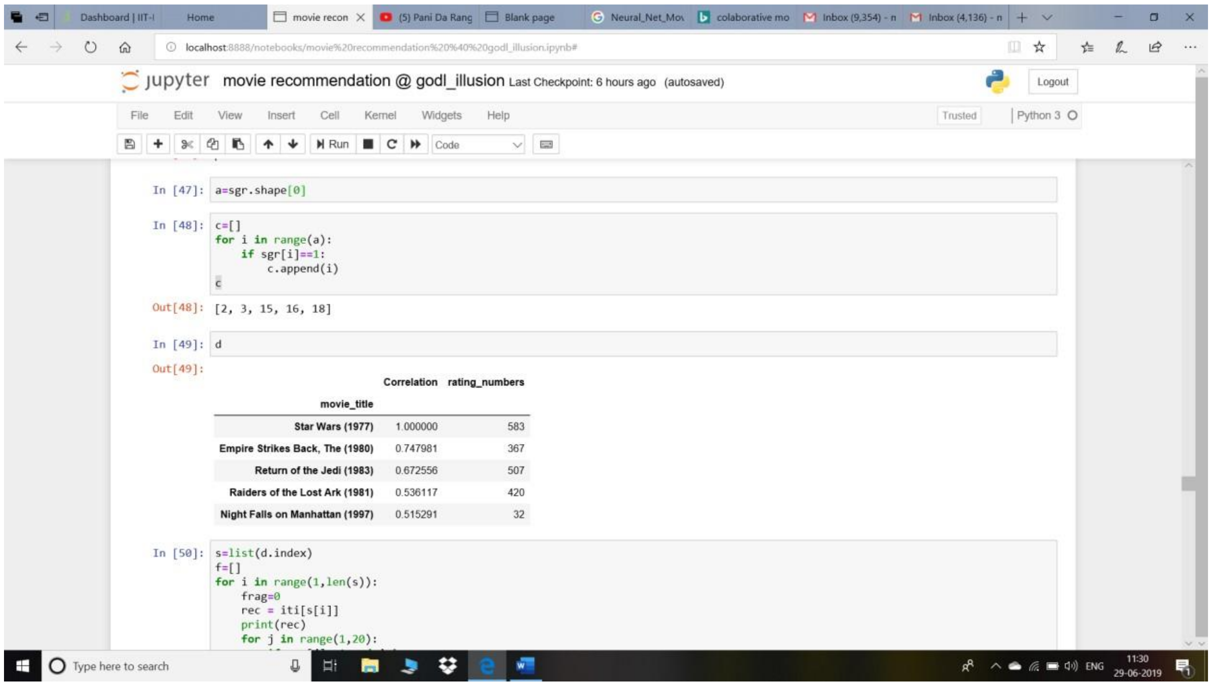
Set a threshold for the number of ratings necessary and filter out movies that have less than a certain number of reviews

```
In [38]: #JOIN THE NO. OF RATINGS COLUMN TO A PARTICULAR MOVIE RECOMMENDATIONS

In [39]: corr_starwars = corr_starwars.join(ratings['rating_numbers'], how='left', lsuffix='_left', rsuffix='_right')
         corr_starwars.head(10)
```

Out[39]:

movie_title	Correlation	rating_numbers
'Til There Was You (1997)	0.872872	9
1-900 (1994)	-0.645497	5
101 Dalmatians (1996)	0.211132	109
12 Angry Men (1957)	0.184289	125
187 (1997)	0.027398	41
2 Days in the Valley (1996)	0.066654	93
20,000 Leagues Under the Sea (1954)	0.289768	72
2001: A Space Odyssey (1968)	0.230884	259
39 Steps, The (1935)	0.106453	59
8 1/2 (1963)	-0.142977	38

```
In [40]: # FILTERING MOVIES WITH LESS THAN 100 REVIEWS
         d=corr_starwars[corr_starwars['rating_numbers']>30].sort_values('Correlation', ascending=False).head()
         d
```



jupyter movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

```
In [47]: a=sgr.shape[0]
```

```
In [48]: c=[]
         for i in range(a):
             if sgr[i]==1:
                 c.append(i)
         c
```

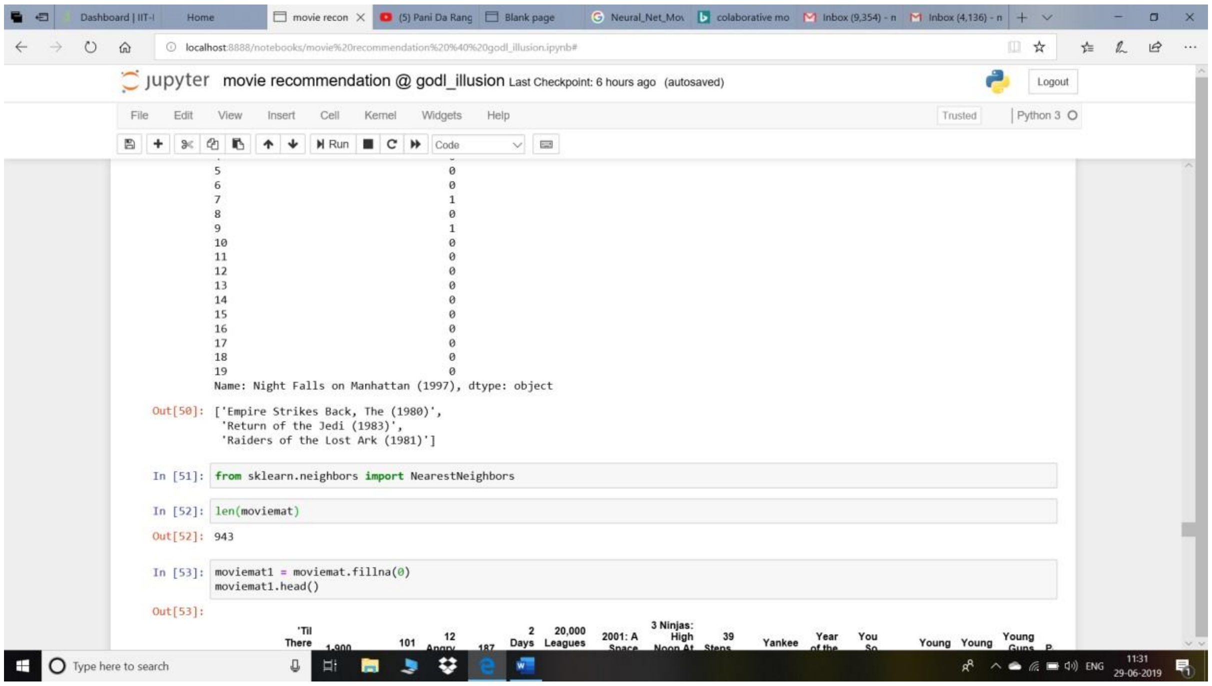
```
Out[48]: [2, 3, 15, 16, 18]
```

```
In [49]: d
```

Out[49]:

movie_title	Correlation	rating_numbers
Star Wars (1977)	1.000000	583
Empire Strikes Back, The (1980)	0.747981	367
Return of the Jedi (1983)	0.672556	507
Raiders of the Lost Ark (1981)	0.536117	420
Night Falls on Manhattan (1997)	0.515291	32

```
In [50]: s=list(d.index)
         f=[]
         for i in range(1,len(s)):
             frag=0
             rec = iti[s[i]]
             print(rec)
             for j in range(1,20):
```



jupyter movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

```
5                                0
6                                0
7                                1
8                                0
9                                1
10                               0
11                               0
12                               0
13                               0
14                               0
15                               0
16                               0
17                               0
18                               0
19                               0
Name: Night Falls on Manhattan (1997), dtype: object
```

```
Out[50]: ['Empire Strikes Back, The (1980)',
          'Return of the Jedi (1983)',
          'Raiders of the Lost Ark (1981)']
```

```
In [51]: from sklearn.neighbors import NearestNeighbors
```

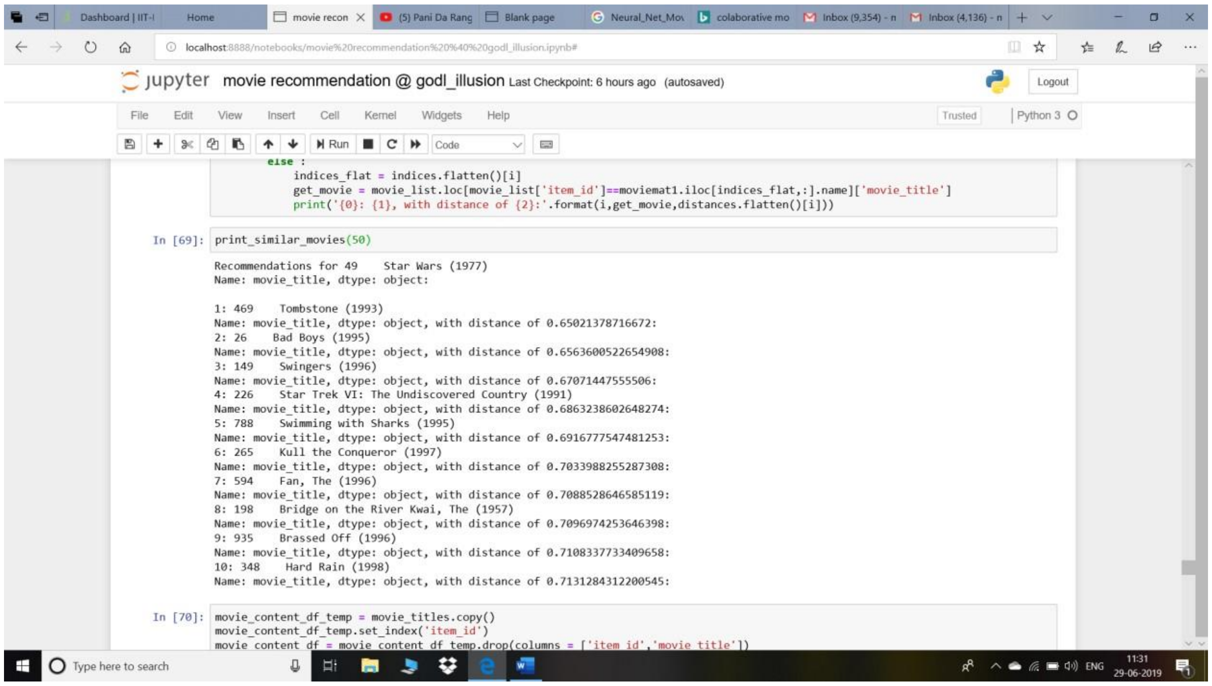
```
In [52]: len(moviemat)
```

```
Out[52]: 943
```

```
In [53]: moviemat1 = moviemat.fillna(0)
         moviemat1.head()
```

Out[53]:

```
   'Til           101   12          2   20,000  2001: A  3 Ninjas:   39                Year   You                         Young
  There  1,900          Angry  187  Days Leagues  Space     High     Steps   Yankee   of the  So        Young   Young    Guns   P
                                                          Noon At
```



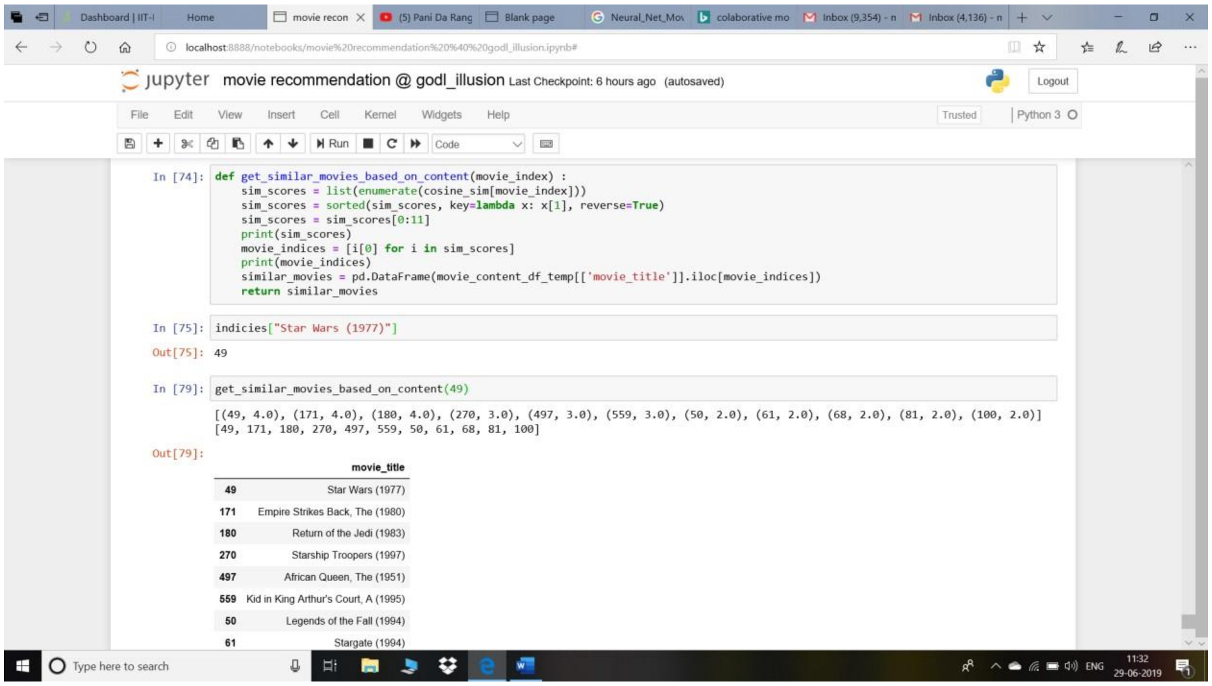
movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

```
        else :
            indices_flat = indices.flatten()[i]
            get_movie = movie_list.loc[movie_list['item_id']==moviemat1.iloc[indices_flat,:].name]['movie_title']
            print('{0}: {1}, with distance of {2}:'.format(i,get_movie,distances.flatten()[i]))
```

```
In [69]: print_similar_movies(50)
```

```
Recommendations for 49    Star Wars (1977)
Name: movie_title, dtype: object:

1: 469    Tombstone (1993)
Name: movie_title, dtype: object, with distance of 0.65021378716672:
2: 26    Bad Boys (1995)
Name: movie_title, dtype: object, with distance of 0.6563600522654908:
3: 149    Swingers (1996)
Name: movie_title, dtype: object, with distance of 0.67071447555506:
4: 226    Star Trek VI: The Undiscovered Country (1991)
Name: movie_title, dtype: object, with distance of 0.6863238602648274:
5: 788    Swimming with Sharks (1995)
Name: movie_title, dtype: object, with distance of 0.6916777547481253:
6: 265    Kull the Conqueror (1997)
Name: movie_title, dtype: object, with distance of 0.7033988255287308:
7: 594    Fan, The (1996)
Name: movie_title, dtype: object, with distance of 0.7088528646585119:
8: 198    Bridge on the River Kwai, The (1957)
Name: movie_title, dtype: object, with distance of 0.7096974253646398:
9: 935    Brassed Off (1996)
Name: movie_title, dtype: object, with distance of 0.7108337733409658:
10: 348    Hard Rain (1998)
Name: movie_title, dtype: object, with distance of 0.7131284312200545:
```

```
In [70]: movie_content_df_temp = movie_titles.copy()
         movie_content_df_temp.set_index('item_id')
         movie_content_df = movie_content_df_temp.drop(columns = ['item_id','movie_title'])
```



movie recommendation @ godl_illusion Last Checkpoint: 6 hours ago (autosaved)

```
In [74]: def get_similar_movies_based_on_content(movie_index) :
             sim_scores = list(enumerate(cosine_sim[movie_index]))
             sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
             sim_scores = sim_scores[0:11]
             print(sim_scores)
             movie_indices = [i[0] for i in sim_scores]
             print(movie_indices)
             similar_movies = pd.DataFrame(movie_content_df_temp[['movie_title']].iloc[movie_indices])
             return similar_movies
```

```
In [75]: indicies["Star Wars (1977)"]

Out[75]: 49
```

```
In [79]: get_similar_movies_based_on_content(49)

[(49, 4.0), (171, 4.0), (180, 4.0), (270, 3.0), (497, 3.0), (559, 3.0), (50, 2.0), (61, 2.0), (68, 2.0), (81, 2.0), (100, 2.0)]
[49, 171, 180, 270, 497, 559, 50, 61, 68, 81, 100]
```

Out[79]:

	movie_title
49	Star Wars (1977)
171	Empire Strikes Back, The (1980)
180	Return of the Jedi (1983)
270	Starship Troopers (1997)
497	African Queen, The (1951)
559	Kid in King Arthur's Court, A (1995)
50	Legends of the Fall (1994)
61	Stargate (1994)

Chapter 6

Limitations of the Project

- I couldn’t get a suitable dataset to exactly demonsatrate my plans of merging Social data along with standardized ratings (such as IMDB) and content and collaborative results although these have been taken out individually in the project.
- Hybrid technique includes only KNN merged with content based and collaborative filtering.

Chapter 7

Conclusion and Future Scope

In future I plan to combine all of this techniques with social connections. Currently in the apps we use they don’t consider social connections while recommendating.

Some applications of recommendation systems also include:

- E-government recommendation system
- E-Business Recommendation System
- E-commerce / e-shopping recommendation system
- E-library recommendation system
- E-learning recommendation system
- E-Tourism Recommendation System
- E-Resource Service Recommendation System
- E-Group Activity Recommendation System

REFRENCES

Books

- *Kim Falk (January 2019), Practical Recommendation System, Manning Publications, ISBN 9781617292705*
- *Bharat Bhaskar; K. Sreekumar (2010). Recommended system in e-commerce. Cup. ISBN 978-0-07068067-8. Archived from the original on 2010-09-01.*
- *Er Gerhard Frederick (2010). Recommended Systems: An Introduction. Cup. ISBN 978-0-521-493369. Archived from the original on 2015-08-31.*

Scientific Articles

- *Prem Melville, Raymond J. Mooney and Ramdas Nagarajan. (2002) A Content-Boosted Collaboration Filter for Improved Recommendations. In Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002), pp. 187-192, Edmonton, Canada, July 2002.*
- *Frank Meyer. (2012), Recommended systems in industrial contexts. Arxiv e-print.*

References

- Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011, pp. 1-35
- Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Bosagh Zadeh WTF:The who-to-follow system at Twitter, Proceedings of the 22nd international conference on World Wide Web