Movie Recommendation System

Created By

Abhishek Sethi (2K18/EC/015)

Content

1. <u>Motivation</u>

Recommendation systems are becoming increasingly important in today's extremely busy world. People are always short on time with the myriad tasks they need to accomplish in their limited time. Recommendation systems are extremely useful as they help them make the right choices, without having to expend their cognitive resources.

The purpose of a recommendation system is to search for content that would be interesting to an individual. Moreover, it involves a number of factors to create personalized lists of useful and interesting content specific to each user/individual. These results are based on their profile, search/browsing history, what other people with similar traits/demographics are watching, and how likely are you to watch those movies. This is achieved through predictive modeling and heuristics with the data available.

There are a variety of different modeling techniques in present being used for the recommendation system, which could be developed and transformed further to give an even better result. This motivated us to design a new modeling technique which can perform significantly better than the already existing techniques.

2. <u>Abstract</u>

Nearly everything around us in the internet is a recommendation to us in some way. We have applied different techniques in this project to improve the movies recommended to users rather than old techniques of giving same suggestion to everyone. With this methodology we can make recommendations to user based on his/her watch history, their likes and dislikes and their ratings to a particular genre of movies. We use content-based and collaborative filtering to construct a system that provides precise and accurate recommendations of concerning movies. In the end we also aim to propose a hybrid model which is a mix of content and collaborative filtering and gives more personalized results.

Keywords – Content based Model, Collaborative based model, Hybrid approach, TF-IDF, SVD, IMDB weighted ratings algorithm

Experimental design

3.1 Datasets

For the Recommendation System, we are using ***Movie Lens*** datasetsfrom (grouplens.org):

- **The Full Dataset:** Our dataset originally contains 45,000 movies and we have built our simple recommender on that dataset only. For our content, collaborative and hybrid model we have used a small dataset of 9,000 rows because of the limited computation power available to us.

3.2 ML Algorithm/Technique

Simple recommender:- Simple recommender is the most basic type of recommender we have available. It works on it's specific formula and gives classification based on that regardless of user's personal taste. The Formula is designed by the designer and we are using IMDB formula in this project. It gives recommendations based on Movie popularity and sometimes genre. Idea is that more popular the movie, higher is the chance that people will like it.

We will use the *IMDB rating chart formula*for this model. Mathematically it is shown as:

Weighted rating (WR) = $\left(\frac{V}{V+M}.R\right)+\left(\frac{M}{V+M}.C\right)$

where,

- *v* is the number of votes for the movie
- *M* is the minimum votes required to be listed in the chart
- *R* is the average rating of the movie
- *C* is the mean vote across the whole report

Content Based recommender:- Content based recommender as the name states gives recommendations on the basis of similarity to the content you are already watching. There are many similarity measures available to us out there but we are using cosine similarity because of it's efficiency and correctness. It is also magnitude independent and is very easy to compute.

Mathematically, it is defined as follows:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}^{\intercal}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} = \frac{\sum_{i=1}^{n} \mathbf{x}_i \cdot \mathbf{y}^{\intercal}{}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{x}_i)^2}\sqrt{\sum_{i=1}^{n} (\mathbf{y}_i)^2}}$$

The TF-IDF score is the frequency of a word occurring in a document, down-weighted by the number of documents in which it occurs. This is done to reduce the importance of words that frequently occur in plot overviews and, therefore, their significance in computing the final similarity score.

The mathematical form of TF-IDF is given below wherein score is given as word t in document d from document set D -

$$tf\,idf\,(t, d, D) = tf\,(t, d)\ .\ idf\,(t, D)$$

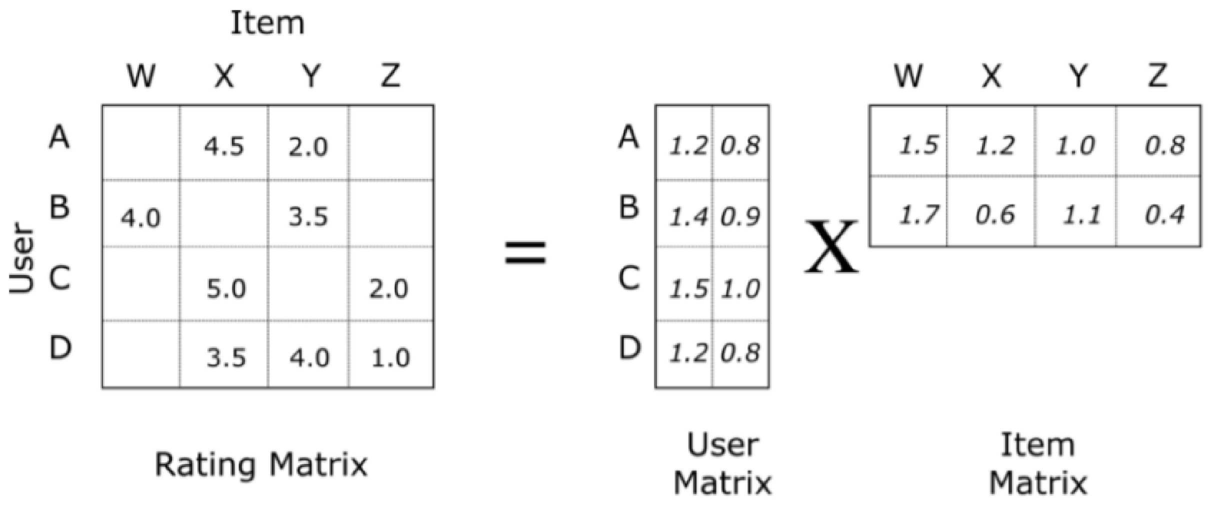
Where:

$$tf\,(t, d) = \log\,(1 + freq\,(t, d))$$

$$idf\,(t, D) = \log\,\left(\frac{N}{count\,(d \in D{:}t \in d)}\right)$$

<u>Collaborative Based recommender:-</u> Collaborative filtering is basically a user similarity recommendation technique where preferences of users can be used to correlate each other and give further recommendations to similar users which have not watched that particular movie or item.

In the case of collaborative filtering, matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. One matrix can be seen as the user matrix where rows represent users and columns are latent factors. The other matrix is the item matrix where rows are latent factors and columns represent items.



Disadvantage of Collaborative filtering-based Recommender:

Collaborative filtering-based recommender also suffers from some limitations:

- **Cold-Start:** It doesn't work with cold-start user or items, since the dot product will be all 0s. It can't recommend anything.
- **Sparsity:** Similarly, it doesn't work with sparse data, since the intersection between 2 users is 0, the dot product is also 0.
- **Scalability:** We need to calculate the user similarity or item similarity matrix. This is a large matrix that doesn't scale with large number of users.

3.3 Research variables

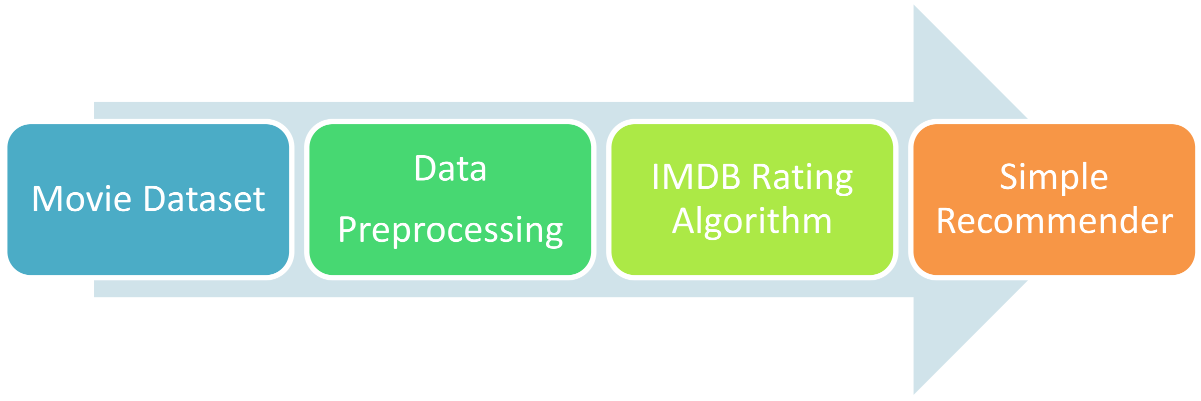
For different recommendation system, different research variables areused:

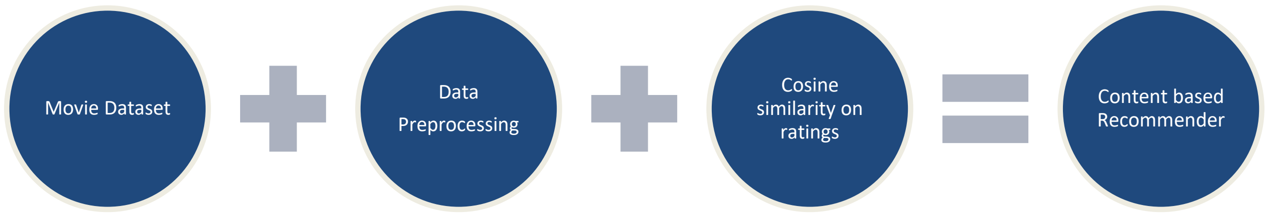
Simple Recommender: Number of votes for the movie, Average rating of the movie, etc.

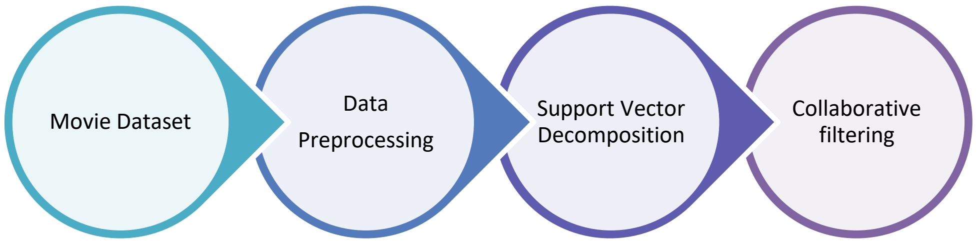
Content based Recommender: Movie genre, movie director, movie overviews and taglines, movie cast, crew, keywords, etc.

Collaborative filtering: Movie rating, Number of votes for the movie, Average rating of the movie

3.4 Experimental Diagram





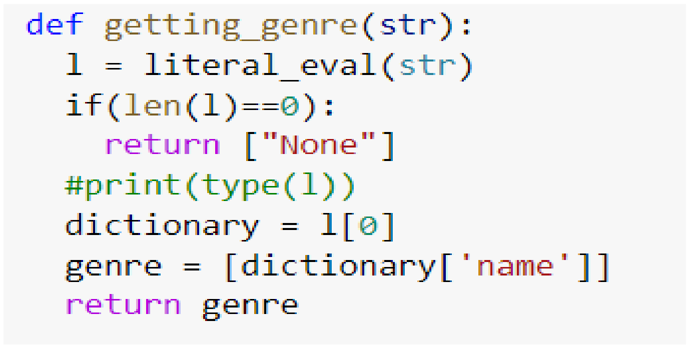


4. Research Methodology

4.1 Data Pre-processing Technique

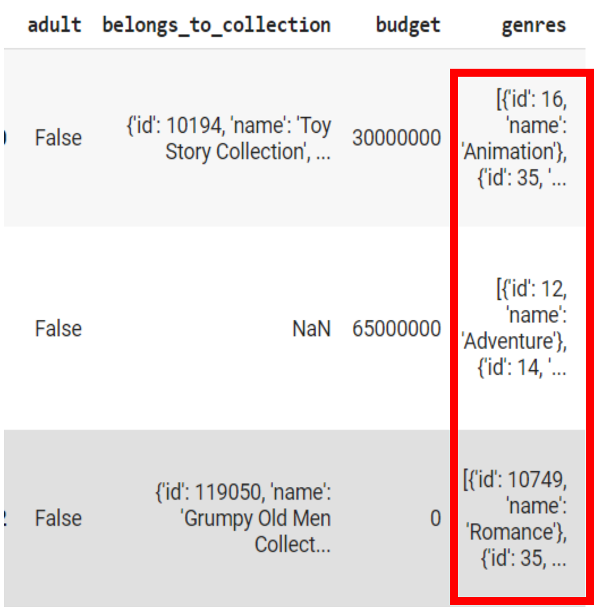
In this, we started off with the understanding of our data as to how are values stored in our table. How many values are null values, their data types.Plenty of type conversions through astype() is done in our code.

One very interesting was the genre class in which the data was stored in a very vague format and had multiple values in order of their relativity. We created a function for simplifying it and choosing the genre which was the closest to the given movie. The snippet for the function is attached below.

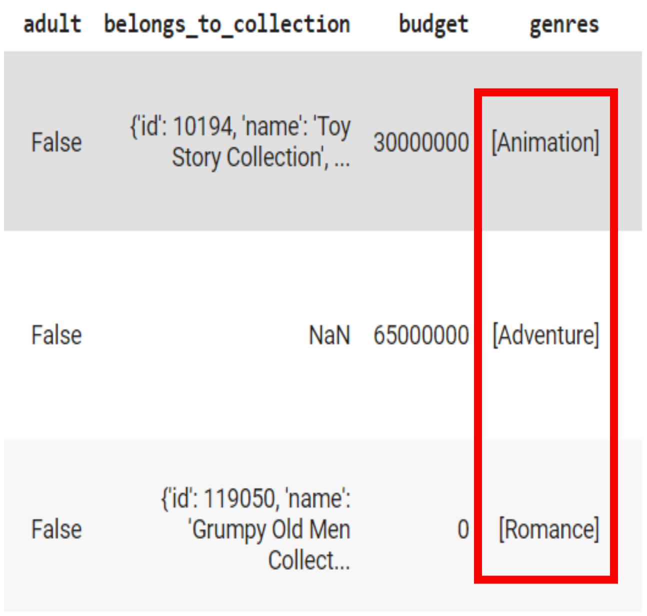


```
def getting_genre(str):
  l = literal_eval(str)
  if(len(l)==0):
    return ["None"]
  #print(type(l))
  dictionary = l[0]
  genre = [dictionary['name']]
  return genre
```

Before **After**



adult	belongs_to_collection	budget	genres
False	{'id': 10194, 'name': 'Toy Story Collection', ...	30000000	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...
False	NaN	65000000	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...
False	{'id': 119050, 'name': 'Grumpy Old Men Collect...	0	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...



adult	belongs_to_collection	budget	genres
False	{'id': 10194, 'name': 'Toy Story Collection', ...	30000000	[Animation]
False	NaN	65000000	[Adventure]
False	{'id': 119050, 'name': 'Grumpy Old Men Collect...	0	[Romance]

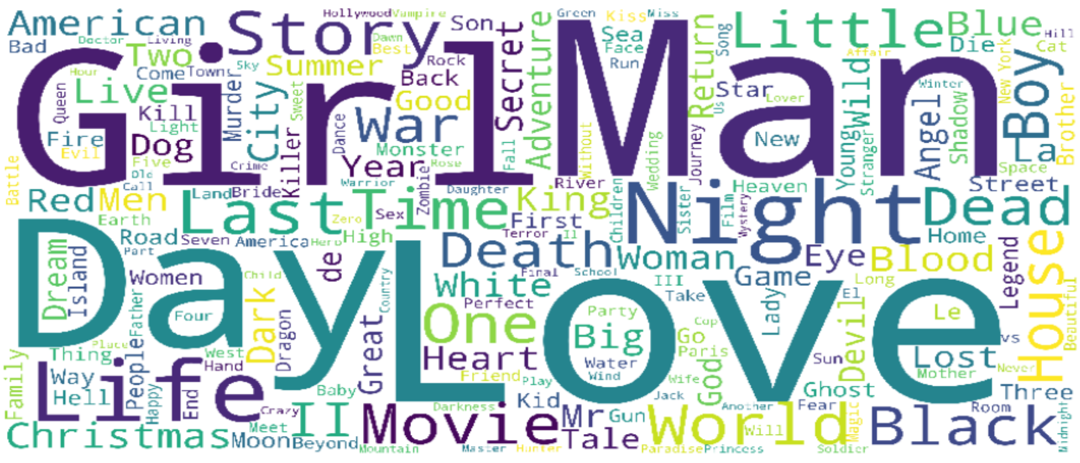
4.2

<u>DataAnalysis Technique</u>

We used advanced graphs to derive insights from our data. Few of which are shown below :

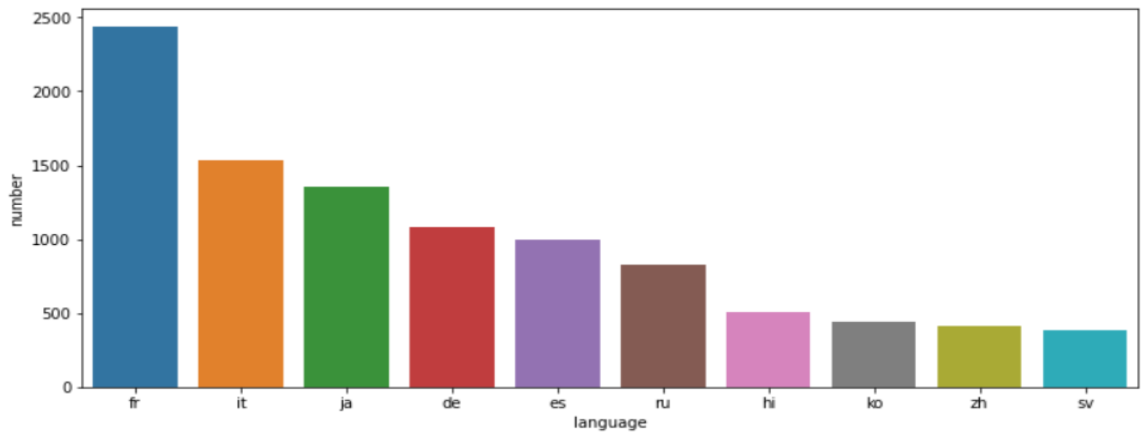
- Checking which is the most used name in the Movie titles, this is generated using wordcloud. -

```
title_wordcloud = WordCloud(stopwords=STOPWORDS, background_color='white', height=2000, wi
dth=4000).generate(title_corpus)
plt.figure(figsize=(16,8))
plt.imshow(title_wordcloud)
plt.axis('off')
plt.show()
```



- Checking the Language of movies and seeing which language has the maximum number of movies :

```
plt.figure(figsize=(12,5))
sns.barplot(x='language', y='number', data=lang_df.iloc[1:11])
plt.show()
```

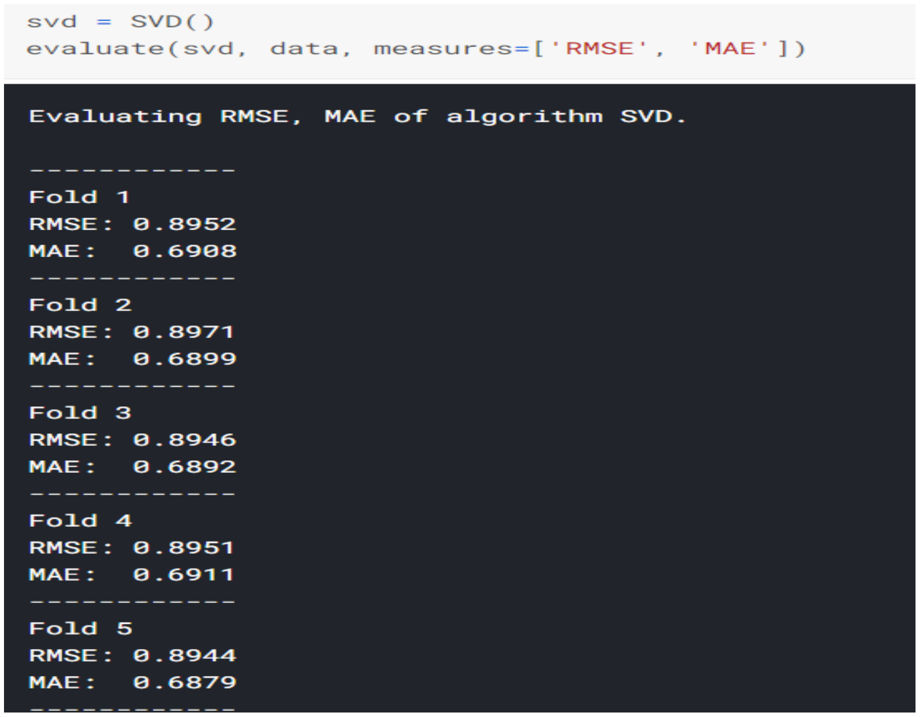


4.3 Performance Measure

For a recommendation system, there is no factor for performance measure because a recommendation engine recommends you rather than predicts value for you. Basically there is no quantitative metric for us to judge our recommendations. So in exchange of that,

- Content Based Model :
  In our content based model, we will use cosine similarity to generate the relation of similarity between 2 movies. After checking the cosine similarity between our user's preference and our available movies. The recommendation is made.

- Collaborative based model:
  In our collaborative based model we will use SVD(singular value decomposition) to reduce the dimensions of our data and generate similarities to do the recommendations for us. In every step, SVD() calculates the RMSE values for us of the matrix factorization results hence it makes a useful performance measure for us in selecting our matrix for final recommendation.
  As you can see, RMSE, is less than 1(i.e 0.89 around).



```
svd = SVD()
evaluate(svd, data, measures=['RMSE', 'MAE'])
```

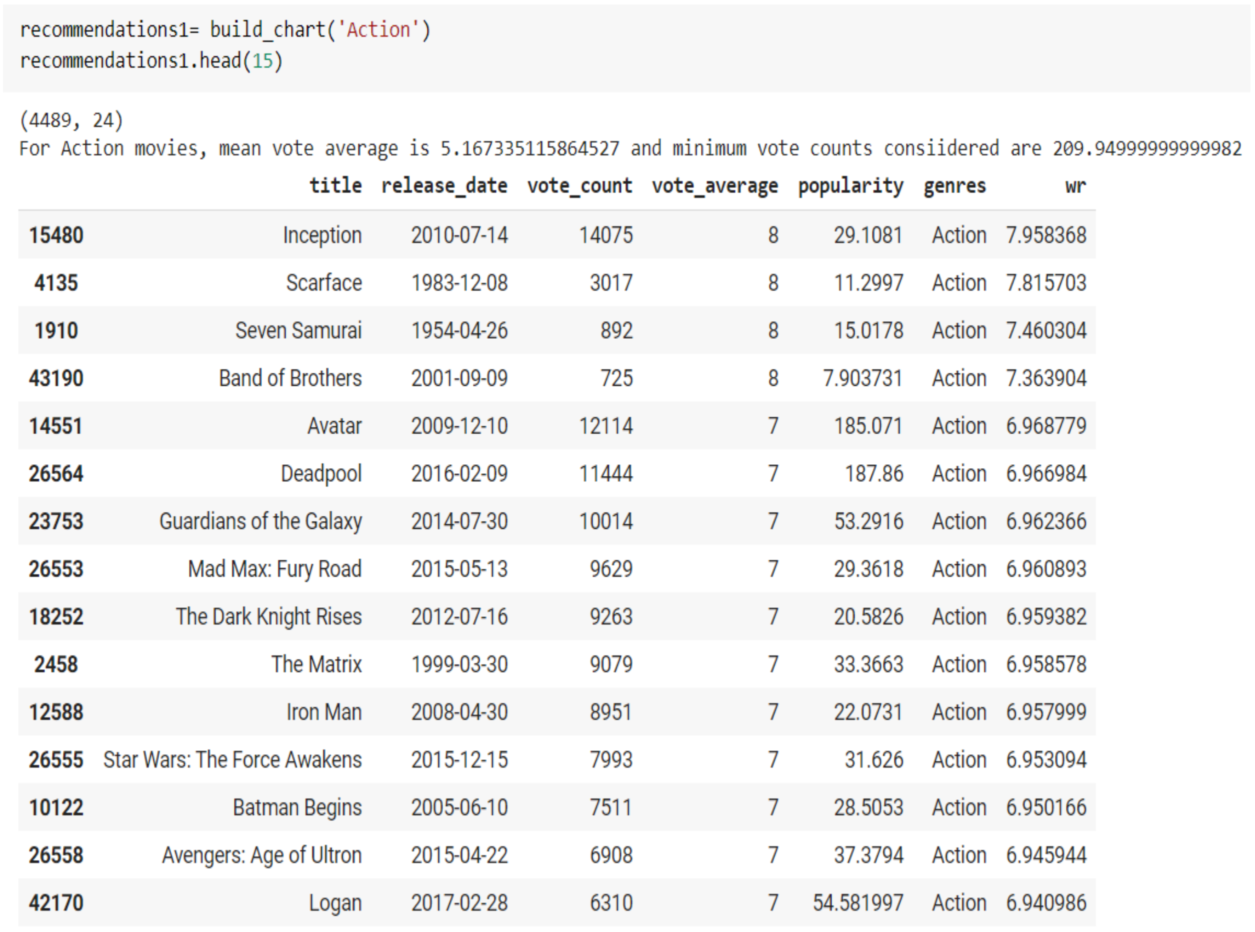
```
Evaluating RMSE, MAE of algorithm SVD.

------------
Fold 1
RMSE: 0.8952
MAE:  0.6908
------------
Fold 2
RMSE: 0.8971
MAE:  0.6899
------------
Fold 3
RMSE: 0.8946
MAE:  0.6892
------------
Fold 4
RMSE: 0.8951
MAE:  0.6911
------------
Fold 5
RMSE: 0.8944
MAE:  0.6879
------------
```

5. **Results –**

Following are the results we got with different type of recommenders:

1. **Simple Recommender:**

Recommendation for **'Action'** genre :



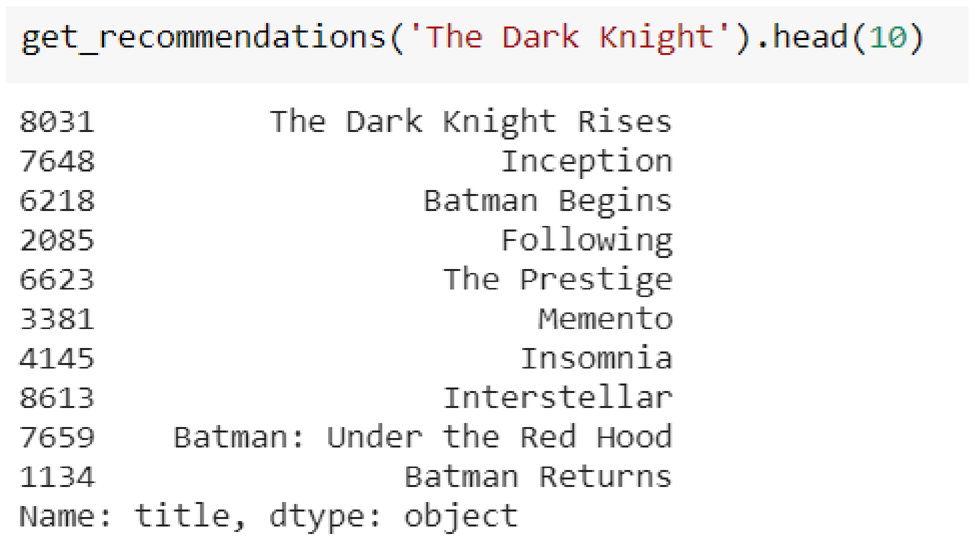
```
recommendations1= build_chart('Action')
recommendations1.head(15)
```

```
(4489, 24)
For Action movies, mean vote average is 5.167335115864527 and minimum vote counts consiidered are 209.94999999999982
```

	title	release_date	vote_count	vote_average	popularity	genres	wr
15480	Inception	2010-07-14	14075	8	29.1081	Action	7.958368
4135	Scarface	1983-12-08	3017	8	11.2997	Action	7.815703
1910	Seven Samurai	1954-04-26	892	8	15.0178	Action	7.460304
43190	Band of Brothers	2001-09-09	725	8	7.903731	Action	7.363904
14551	Avatar	2009-12-10	12114	7	185.071	Action	6.968779
26564	Deadpool	2016-02-09	11444	7	187.86	Action	6.966984
23753	Guardians of the Galaxy	2014-07-30	10014	7	53.2916	Action	6.962366
26553	Mad Max: Fury Road	2015-05-13	9629	7	29.3618	Action	6.960893
18252	The Dark Knight Rises	2012-07-16	9263	7	20.5826	Action	6.959382
2458	The Matrix	1999-03-30	9079	7	33.3663	Action	6.958578
12588	Iron Man	2008-04-30	8951	7	22.0731	Action	6.957999
26555	Star Wars: The Force Awakens	2015-12-15	7993	7	31.626	Action	6.953094
10122	Batman Begins	2005-06-10	7511	7	28.5053	Action	6.950166
26558	Avengers: Age of Ultron	2015-04-22	6908	7	37.3794	Action	6.945944
42170	Logan	2017-02-28	6310	7	54.581997	Action	6.940986

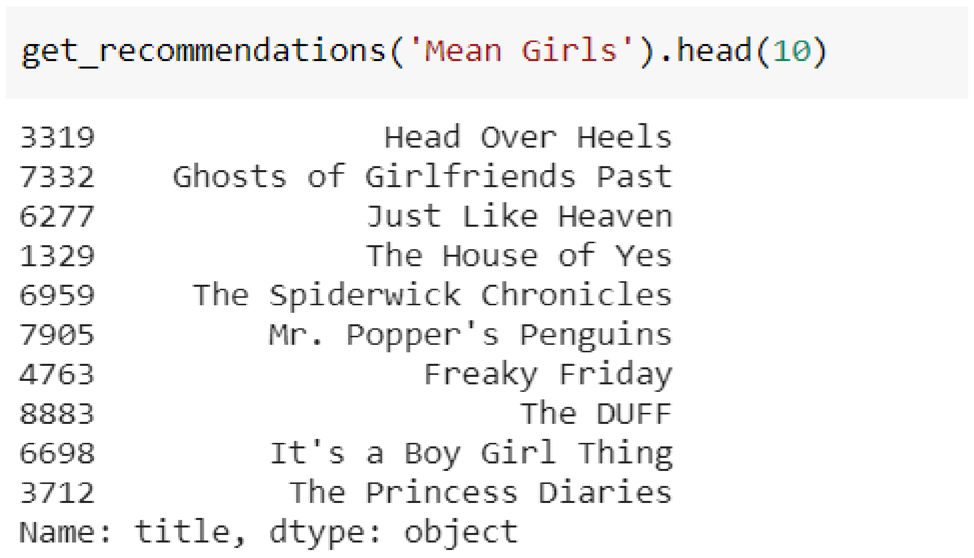
2: Content based recommender

Result generated for movies similar to **'The Dark Knight'** and **'Mean Girls'** respectively :



```python
get_recommendations('The Dark Knight').head(10)
```

```
8031          The Dark Knight Rises
7648                      Inception
6218                  Batman Begins
2085                      Following
6623                   The Prestige
3381                        Memento
4145                       Insomnia
8613                   Interstellar
7659    Batman: Under the Red Hood
1134                 Batman Returns
Name: title, dtype: object
```

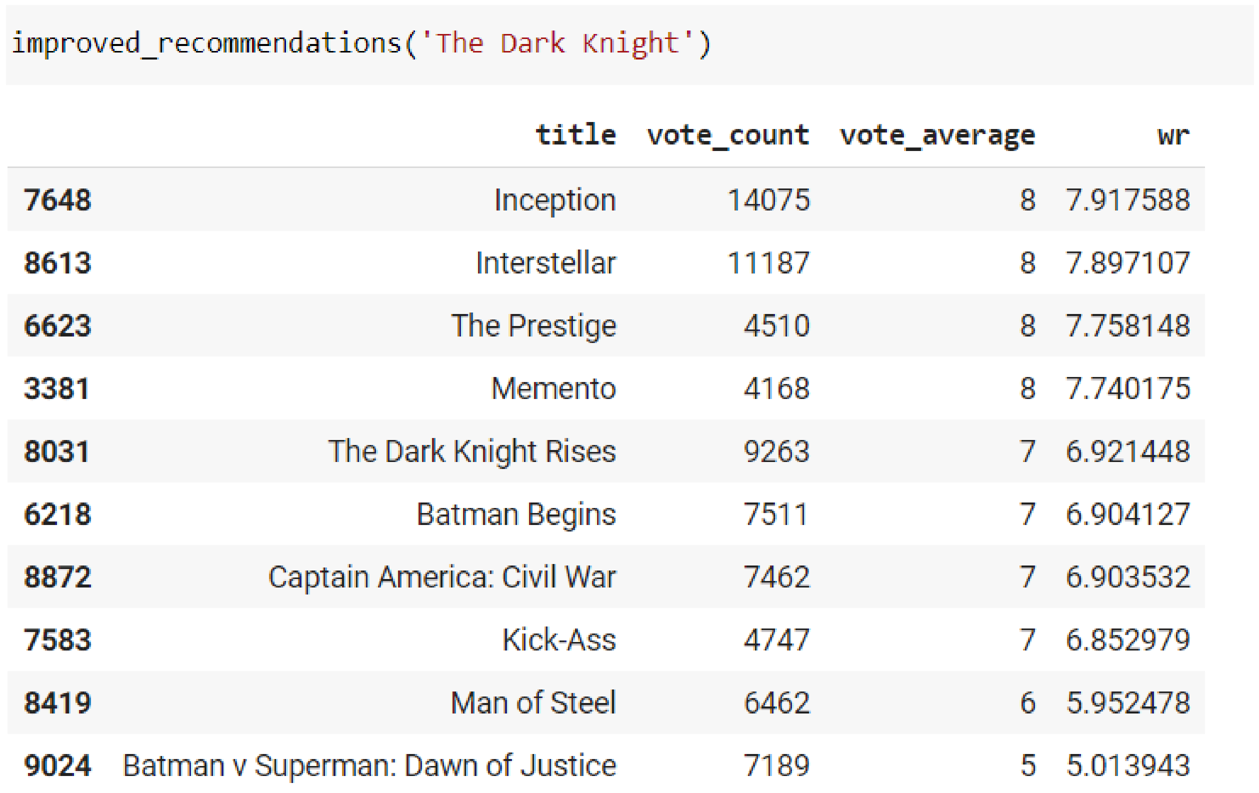


```python
get_recommendations('Mean Girls').head(10)
```

```
3319                Head Over Heels
7332    Ghosts of Girlfriends Past
6277               Just Like Heaven
1329               The House of Yes
6959     The Spiderwick Chronicles
7905          Mr. Popper's Penguins
4763                  Freaky Friday
8883                       The DUFF
6698          It's a Boy Girl Thing
3712           The Princess Diaries
Name: title, dtype: object
```

3 : (Content + Simple Recommender)

Result generated for the movie **‘The Dark Knight’** :

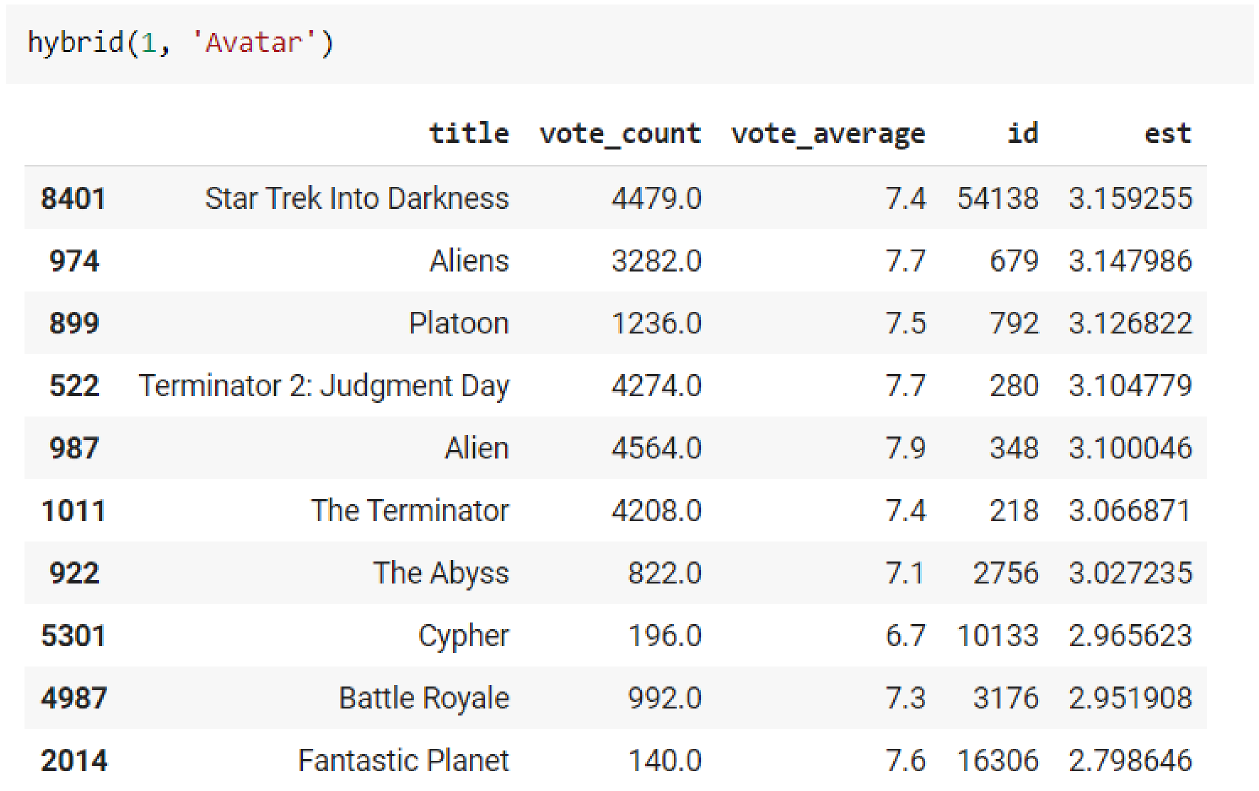


```
improved_recommendations('The Dark Knight')
```

	title	vote_count	vote_average	wr
7648	Inception	14075	8	7.917588
8613	Interstellar	11187	8	7.897107
6623	The Prestige	4510	8	7.758148
3381	Memento	4168	8	7.740175
8031	The Dark Knight Rises	9263	7	6.921448
6218	Batman Begins	7511	7	6.904127
8872	Captain America: Civil War	7462	7	6.903532
7583	Kick-Ass	4747	7	6.852979
8419	Man of Steel	6462	6	5.952478
9024	Batman v Superman: Dawn of Justice	7189	5	5.013943

4. Hybrid Recommender (Content + collaborative) :

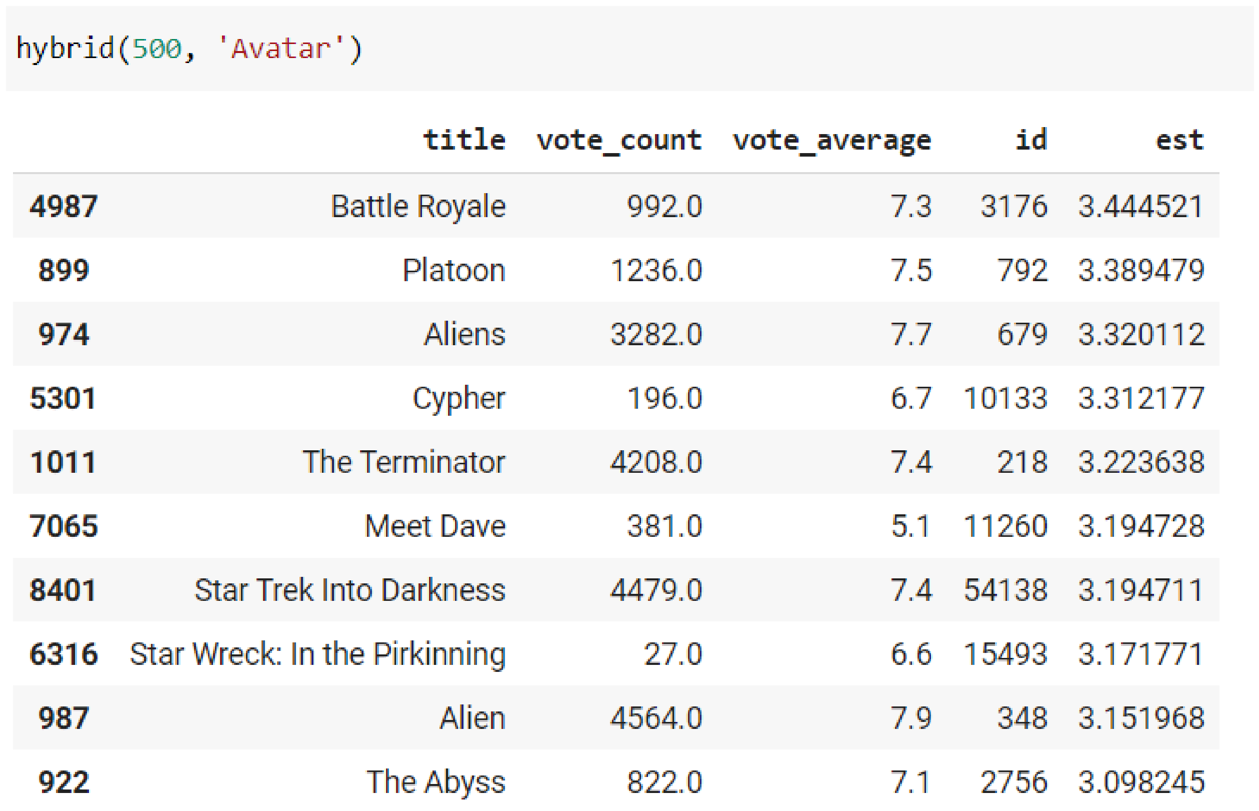
Recommendation for **user 1** for movies related to **‘Avatar’**



```
hybrid(1, 'Avatar')
```

	title	vote_count	vote_average	id	est
8401	Star Trek Into Darkness	4479.0	7.4	54138	3.159255
974	Aliens	3282.0	7.7	679	3.147986
899	Platoon	1236.0	7.5	792	3.126822
522	Terminator 2: Judgment Day	4274.0	7.7	280	3.104779
987	Alien	4564.0	7.9	348	3.100046
1011	The Terminator	4208.0	7.4	218	3.066871
922	The Abyss	822.0	7.1	2756	3.027235
5301	Cypher	196.0	6.7	10133	2.965623
4987	Battle Royale	992.0	7.3	3176	2.951908
2014	Fantastic Planet	140.0	7.6	16306	2.798646

Recommendation for **user 500** for movies related to **‘Avatar’ :**



```
hybrid(500, 'Avatar')
```

	title	vote_count	vote_average	id	est
4987	Battle Royale	992.0	7.3	3176	3.444521
899	Platoon	1236.0	7.5	792	3.389479
974	Aliens	3282.0	7.7	679	3.320112
5301	Cypher	196.0	6.7	10133	3.312177
1011	The Terminator	4208.0	7.4	218	3.223638
7065	Meet Dave	381.0	5.1	11260	3.194728
8401	Star Trek Into Darkness	4479.0	7.4	54138	3.194711
6316	Star Wreck: In the Pirkinning	27.0	6.6	15493	3.171771
987	Alien	4564.0	7.9	348	3.151968
922	The Abyss	822.0	7.1	2756	3.098245

6. <u>Conclusion and Future work–</u>

In this paper we have introduced a recommender system for movie recommendation. We have proposed three types of models, Simple recommender which is based on IMDB weighted charts formula, Content based recommender which gives content based similarity and we have used cosine similarity as a measure to propose such system and the Collaborative Filtering model which gives recommendations based on user similarity. As you all know by now that this is not a prediction per se but actually recommendation so there is no right or wrong, it’s just matter of opinion and the designer to tweak it’s system based on it’s customer’s base preference.

A hybrid approach is taken between context based filtering and collaborative filtering to implement the system. This approach overcomes drawbacks of each individual algorithm and improves the performance of the system.In future we can work on hybrid recommender using clustering and similarity for better performance. Our approach can be further extended to other domains to recommend songs, video, venue, news, books, tourism and e-commerce sites, etc

For future work, We would like to have a larger data set that will enable more meaningful results using our system. Additionally we would like to incorporate different machine learning and clustering algorithms and study the comparative results.

7.<u>References :</u>

1. Geetha, G., Safa, M., Fancy, C., & Saranya, D. (2018, April). A hybrid approach using collaborative filtering and content based filtering for recommender system. In Journal of Physics: Conference Series (Vol. 1000, No. 1, p. 012101).

2. Christakou, C., Vrettos, S., &Stafylopatis, A. (2007). A hybrid movie recommender system based on neural networks. International Journal on Artificial Intelligence Tools, 16(05), 771-792.