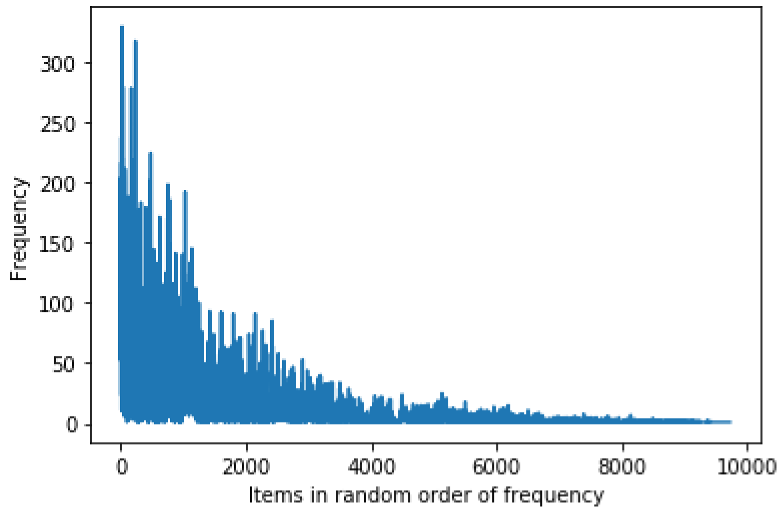
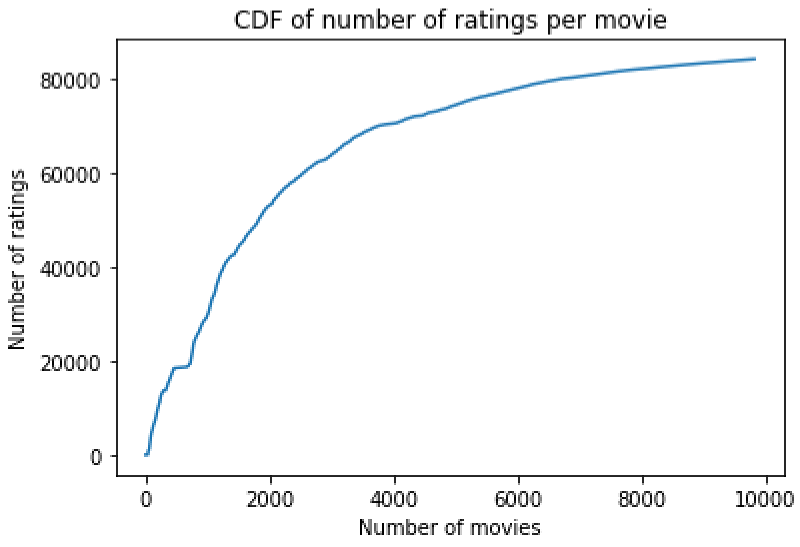
Understanding the dataset:

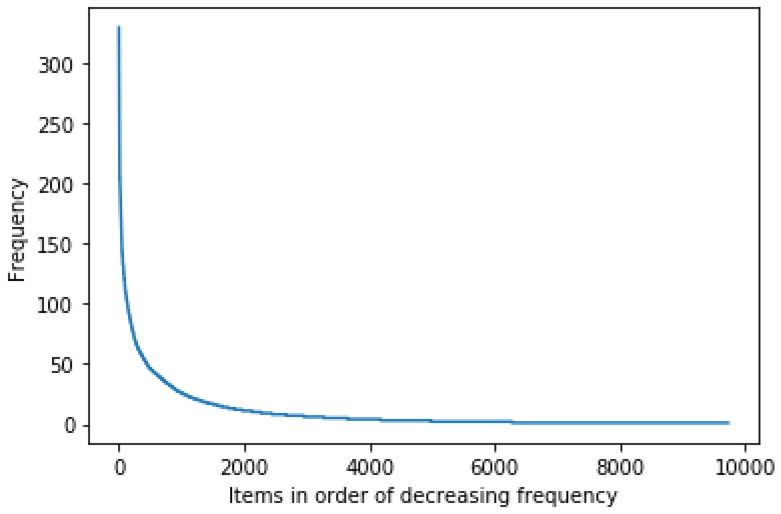
Q1) The plots differ according to the way the movies are arranged. We'll see both the cases, the movies are arranged in descending order of frequency of ratings, and when they are arranged randomly

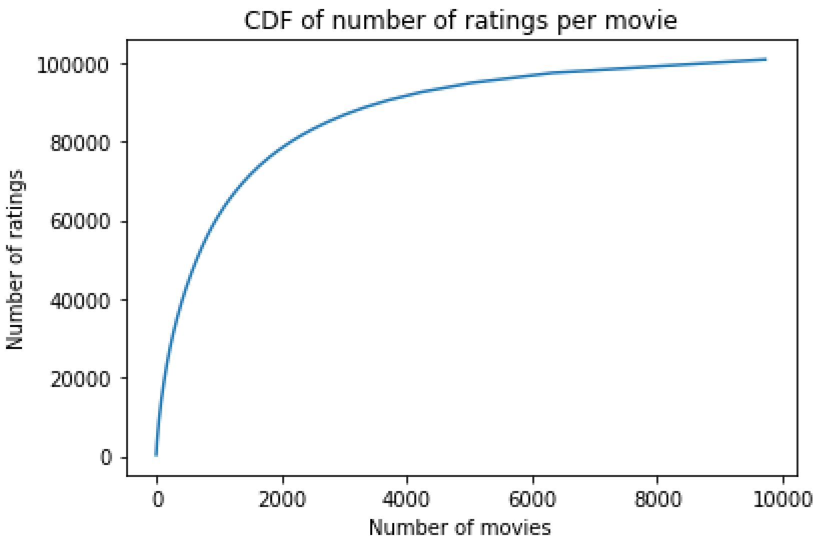
a) Randomly





b) In descending order of frequency of ratings

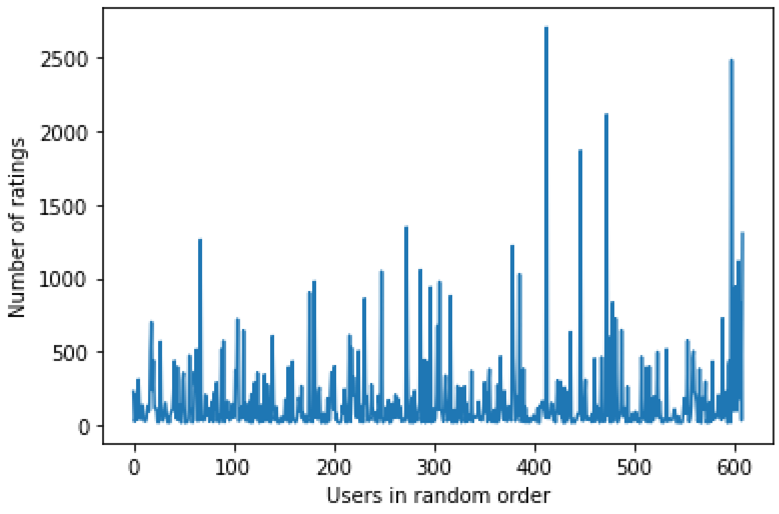




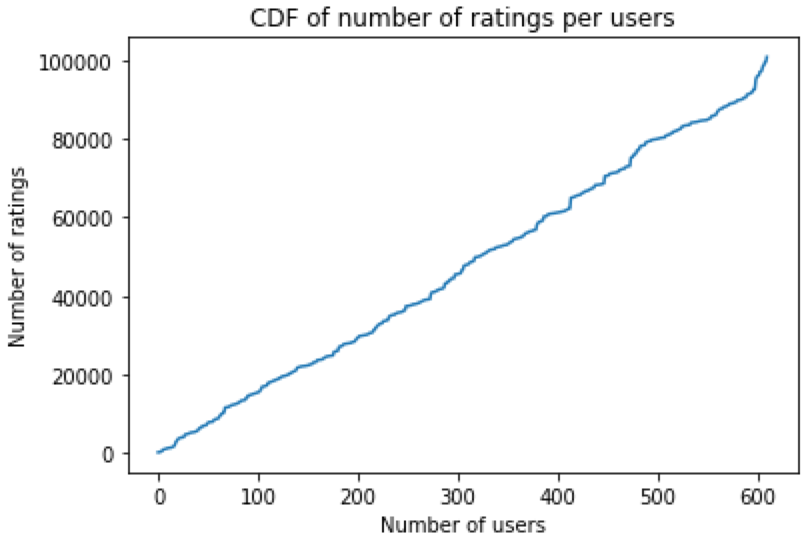
The CDF if the ordered case is similar to the CDF of exponential distributions.

Q2: Similar as the above question, we can have two cases.

a) random

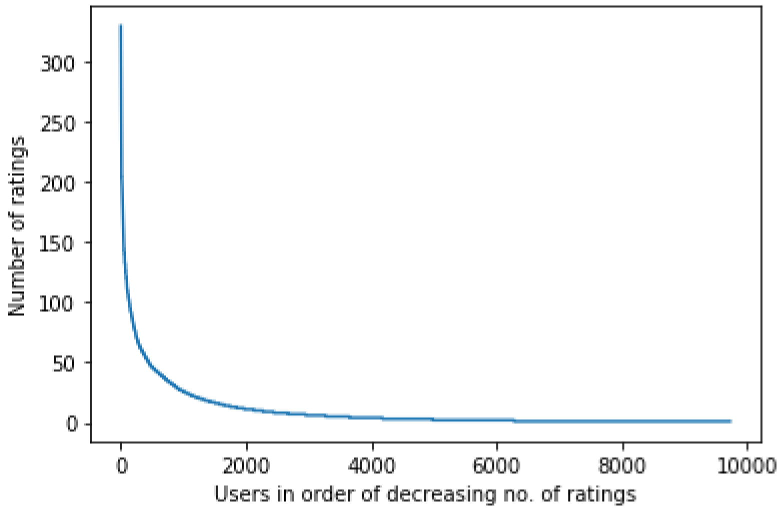


Number of ratings
0
500
1000
1500
2000
2500
0
100
200
300
400
500
600
Users in random order

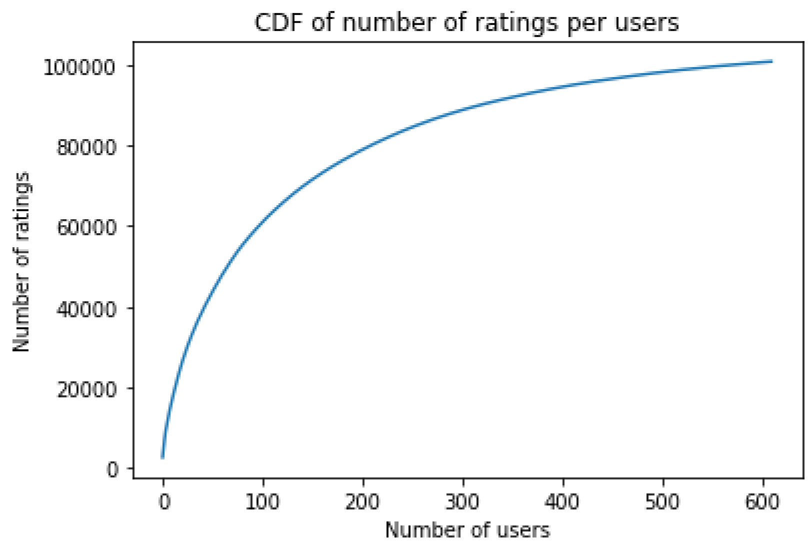


CDF of number of ratings per users
Number of ratings
0
20000
40000
60000
80000
100000
0
100
200
300
400
500
600
Number of users

b) ordered



Number of ratings
0
50
100
150
200
250
300
0
2000
4000
6000
8000
10000
Users in order of decreasing no. of ratings



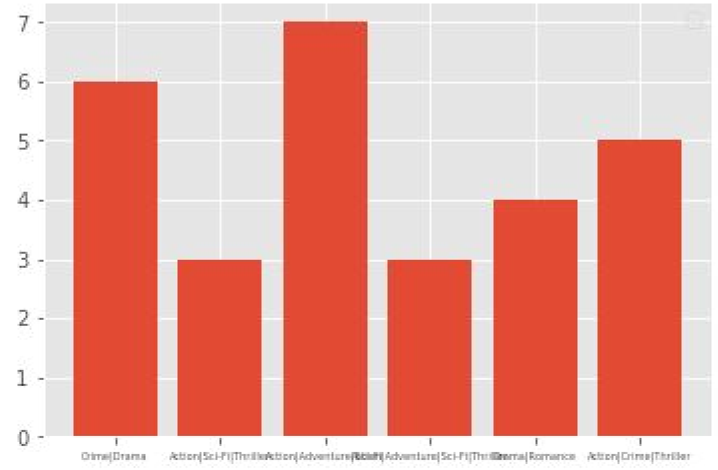
As we can see in the randomised order graphs, there are some outliers in the data. Some of the users have rated more than 1200 movies, while some other have rated less than a hundred movies.

Q3) The criteria I used for most popular 100 movies is the number of ratings, If a movie is rated quite a few times, then it means it has a reach to a lot of users, and hence, is popular. The list:
Forrest Gump (1994)
Shawshank Redemption, The (1994)
Pulp Fiction (1994)
Silence of the Lambs, The (1991)
Matrix, The (1999)
Star Wars: Episode IV - A New Hope (1977)
Jurassic Park (1993)
Braveheart (1995)
Terminator 2: Judgment Day (1991)
Schindler's List (1993)
Fight Club (1999)
Toy Story (1995)
Star Wars: Episode V - The Empire Strikes Back (1980)
Usual Suspects, The (1995)
American Beauty (1999)
Seven (a.k.a. Se7en) (1995)
Independence Day (a.k.a. ID4) (1996)
Apollo 13 (1995)
Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)
Lord of the Rings: The Fellowship of the Ring, The (2001)
Star Wars: Episode VI - Return of the Jedi (1983)
Godfather, The (1972)

Fugitive, The (1993)
Batman (1989)
Saving Private Ryan (1998)
Lord of the Rings: The Two Towers, The (2002)
Lord of the Rings: The Return of the King, The (2003)
Aladdin (1992)
Fargo (1996)
Sixth Sense, The (1999)
True Lies (1994)
Twelve Monkeys (a.k.a. 12 Monkeys) (1995)
Lion King, The (1994)
Back to the Future (1985)
Speed (1994)
Gladiator (2000)
Shrek (2001)
Men in Black (a.k.a. MIB) (1997)
Dances with Wolves (1990)
Mission: Impossible (1996)
Ace Ventura: Pet Detective (1994)
Memento (2000)
Mask, The (1994)
Dark Knight, The (2008)
Pirates of the Caribbean: The Curse of the Black Pearl (2003)
Alien (1979)
Beauty and the Beast (1991)
Die Hard (1988)
Mrs. Doubtfire (1993)
Die Hard: With a Vengeance (1995)
Groundhog Day (1993)
Inception (2010)
Princess Bride, The (1987)
Good Will Hunting (1997)
Finding Nemo (2003)
Stargate (1994)
Indiana Jones and the Last Crusade (1989)
Star Wars: Episode I - The Phantom Menace (1999)
Titanic (1997)
Batman Forever (1995)
Monty Python and the Holy Grail (1975)
Pretty Woman (1990)
Dumb & Dumber (Dumb and Dumber) (1994)
X-Men (2000)
Léon: The Professional (a.k.a. The Professional) (Léon) (1994)

One Flew Over the Cuckoo's Nest (1975)
GoldenEye (1995)
Monsters, Inc. (2001)
Reservoir Dogs (1992)
Terminator, The (1984)
Kill Bill: Vol. 1 (2003)
Eternal Sunshine of the Spotless Mind (2004)
American History X (1998)
Godfather: Part II, The (1974)
Babe (1995)
Goodfellas (1990)
Aliens (1986)
Truman Show, The (1998)
Incredibles, The (2004)
Blade Runner (1982)
Twister (1996)
Beautiful Mind, A (2001)
E.T. the Extra-Terrestrial (1982)
Spider-Man (2002)
Rock, The (1996)
Austin Powers: The Spy Who Shagged Me (1999)
Clockwork Orange, A (1971)
Ghostbusters (a.k.a. Ghost Busters) (1984)
Minority Report (2002)
Amelie (Fabuleux destin d'Amélie Poulain, Le) (2001)
Willy Wonka & the Chocolate Factory (1971)
Ocean's Eleven (2001)
Batman Begins (2005)
Home Alone (1990)
Fifth Element, The (1997)
Waterworld (1995)
Ghost (1990)
Catch Me If You Can (2002)
Breakfast Club, The (1985)
Net, The (1995)

A bit more depth into data appears when we check the genres of the data.



This is the plot of genres with more than 2 movies in the top 100 list.
Crime|Drama (6) and Action|Adventure|Sci-Fi(7) are the most popular genres.

Other thing that highlights the skewed amount of ratings is the stat that **16 percent of all the ratings in the dataset are only for these 100 movies, which are about 0.01 percent of the number of movies in the dataset**.

Recommendation Modelling

Sampling Approach: The approach I've used to determine the split between test and train is uniform sampling. Given that the ratings matrix is in order of users rating movies (e.g. first all ratings by user 'a', then all ratings by user 'b'), a uniform sampling make sure that we acknowledge the possibility of a missing rating entry for a lot of different users. The approach is coded by using random library in python, which has a function random.sample().
Both the functions for judging accuracy metrics have been implemented.

Q4)

Error Table:

	User based(Raw)	User Based(Mean centred)	User Based(Mean centred & IUF)
Pearson--RMSE	1.3844	0.9088	0.9083
Pearson--MAE	0.9259	0.69188	0.6916
Cosine--RMSE	1.00008	0.9133	
Cosine--MAE	0.759	0.7008	
Std-Dev RMSE		1.138127	
Std-Dev MAE		0.7937	

RMSE-> Root Mean Square Error, MAE->Mean Absolute Error

Raw vs Mean Centred:
As seen in the table, there is a considerable amount of improvement as we go for mean centred approach in the prediction function. This is as expected as mean centring takes users' rating fashion into account.

Standard-Deviation Method:
Taking standard deviation into account gives us a result that is better than Raw ratings but not than mean centred one. This may be attributed to the fact the after using standard deviation the ratings may not even stay in the range we expect the ratings to be in, depending on the deviation of similar users. Hence, error using this method may be more, but it can be used when we just want to know the top 'k' items to recommend.

Choosing K:

k	pearson (mean)	pearson(raw)
10	0.69188	0.9259
20	0.6824	1.0227
30	0.6796	1.11838
40	0.6795	1.1944
50	0.6796	1.255
60	0.6801	1.307
80	0.6801	1.38256
100	0.6803	1.444
120	0.6805	1.4854

These results are based on using MAE metric.
This is the most interesting result.
So the approach is testing different Ks and running a same test on each one of them. From this table, we see that k=40 is the best value for mean centred matrix. But, when I ran the same test for raw matrix, the results were quite different, the best value then seems to be k=10. This can be explained by the logic that taking more users into account without considering the fashion in which they exhibit rating behaviour will increase the error.
Even in the mean centred version, we don't see a lot of improvement from k=10 to k=40. I decided against aiming for the marginal improvement in mean centred case on the cost of extra computation and increased error in raw versions, hence I chose k=10.
For Long_Tail Problem:
Inverse User Frequency is used in calculating pearson correlation as a solution to this problem, and results are slightly better than normal pearson correlation. Refer the last column of the table

Q5)

	User based(Mean centred)	Item based(Mean Centred)
Pearson--RMSE	0.9088	1.15935
Pearson--MAE	0.69188	0.8255
Cosine--RMSE	0.9133	1.1552
Cosine--MAE	0.7008	0.8223

Pearson vs Cosine:
As we can see in the table, there is a very slight difference between the results obtained by using these two metrics, however, cosine similarity does a little better on this dataset. Given the we are doing mean centring in prediction function, we account for some of the bias exhibited by users' rating fashion. Hence, this result is not absurd.

Item vs User:
Results tell us that user based measures perform better than item based filtering. This may be due to the accounting of rating behaviour of users in the mean centring method used for user based approach.

Q6 (**Hybrid Approach**):

	User based(Mean centred)	Item based(Mean Centred)	Hybrid-II	Hybrid-I
Pearson--RMSE	0.9088	1.15935	0.93127	0.9464
Pearson--MAE	0.69188	0.8255	0.71	0.7178
Cosine--RMSE	0.9133	1.1552	0.9422	0.9564
Cosine--MAE	0.7008	0.8223	0.7225	0.7277

Two approaches:
Hybrid-1 -> Taking Mean of both ratings
Hybrid-2->
The approach I am taking is to take a weighted mean of the ratings provided by user-user and item-item methods.
So the rating=
(weight_user*user_based_rating+weight_item*item_based_rating)/(weight_user+weight_item)
Now what remains is to know how do we decide weight_item and weight_user.
For that I decided,

weight_item=error_found_by_using_only_user_based_approach
weight_user=error_found_by_using_only_item_based_approach
An intuitive explanation can be that this method takes into account both types of similarities, and penalises the method with has more error by giving more weight to the other method, hence may perform better, but results aren't quite supportive of the guess. Given that user based similarity methods have performed better than item based method, the result of this hybrid approach is better than item based approach but very close to user based approach, but falling short of reaching the mark.

Q7:

	User based(Mean centred)	Item based(Mean Centred)	KMeans (mean centring)	KMeans for data smoothing
Pearson--RMSE	0.9088	1.15935	1.0093086	
Pearson--MAE	0.69188	0.8255	0.774135	
Cosine--RMSE	0.9133	1.1552		
Cosine--MAE	0.7008	0.8223		
Cluster RMSE				0.9622
Cluster MAE				0.738136

The comparison table comes out as above. As we can see, the data smoothing approach works better than item based measures but falls short of the error rate achieved in user based method. This might have happened because the matrix is very sparse, with **98.3% of matrix being null** values (in train set), hence, data smoothing can't be expected to work very well if the context is not well sized.

Final Remarks:
User based measure with similarity function considering Inverse User Frequency seems best on this dataset after seeing the results obtained, KMeans has lesser computational requirements with respect to time taken in offline phase but error rate also increases considerably. Other approaches improve on none of the factors with a significant factor.

Note: You might not want to re-do the computation done in offline phase, so might use this link below to access the numpy matrices that I created and load it using numpy.load(), I'm not zipping the data along with this file because it is a bit large.
https://drive.google.com/open?id=143HiyBOV8aSfqdNBkE6y_AQ06HtYDJ0X