ISSN No:-2456-2165

Movie Recommender System using Content Based andCollaborative Filtering

Harshal Fulzele[1]
Mihir Bhoite[2]
Prajwal Kanfade[3]
Ashutosh Yadav[4]
[1,2,3,4]Artificial Intelligence, G.H.Raisoni College Of Engineering, Nagpur, India

Madhuri Sahu[5]
Achamma Thomas[6]
[5,6]Assistant Professor ,Artificial Intelligence, G.H.Raisoni College Of Engineering, Nagpur , India

Abstract:- Technology has evolved a lot from basic to advanced such as Machine learning, deep learning, Internet of things, Data Mining and many more. Recommender systems provide users with personalized suggestions for products or services also this system only rely on collaborative filtering. Movies are the source of Entertainment but finding the desired content is the problem. Aim of this paper is to improve the accuracy and performance of the regular filtering technique and also to recommend movies based on the content of the movie which users have watched earlier. Collaborative filtering recommends movies to user A based on the interest of similar user B. Netflix is internally using a cinematch algorithm for the collaborative filtering we are improving the accuracy and the performance of regular technique. Content based filtering will help Netflix boost their turnover by providing similar movies which users have watched earlier on any of the OTT(Over The Top) platforms. We have used a surprise library along with the xgboost regressor which makes our model improve from regular technique. We have also designed the frontend for the content based recommendation system system for Netflix.

Keywords:- *Content Based , Collaborative Filtering, Recommender System, Surprise-Library, User-Based Recommender, Item-Based Recommender.*

I. INTRODUCTION

Netflix is basically an online repository where we can watch web series, movies and documentaries etc. Netflix account movies are recommended to us and it says because you have watched this tv show or series e.g Roman empire you may like this movie or TV shows e.g.:- ;Spartacus, The last Kingdom. Netflix would recommend movies or TV shows similar to movies we watched previously. The question is: How do they understand what we will like? So, for every movie m and movies as mi. a user ui and movie as mj suppose user i rates movie j as rij. It is based on user watched movies and given ratings for them between 1 to 5 star:

Using this rating information of thousands and thousands of users and tens of thousands of movies we can somehow predict the type of movie the user can like in the future. We are using predicted rating as the feature for recommending the movie of similar type.

The Higher the predicted rating, the higher is the chance of recommending that movie for the user. We are using the surprise library for this approach which is also calledthe simple python recommendation system engine. We are improving the cinematch algorithm for this. We used the collaborativefiltering plus the xgboost regressor.

Netflix comes with an algorithm called the cinematch system which has some errors for this they are using root mean square error.

We want to improve the cinematch algorithm and the Content based approach tothe solution.

II. DATA

We got the data from Netflix where we havemovie ids ranging from 1 to 17770 sequentially. Customer id range from 1 to 2649429 there are 48018 users. Ratings are on a five steer scale 1 to 5 and dates have format YYYY-MM-DD It has actually 5 text files where we have all this date. eg:- 1: 1488844,3; 2005-09-66 ". Customer id, rating, date.

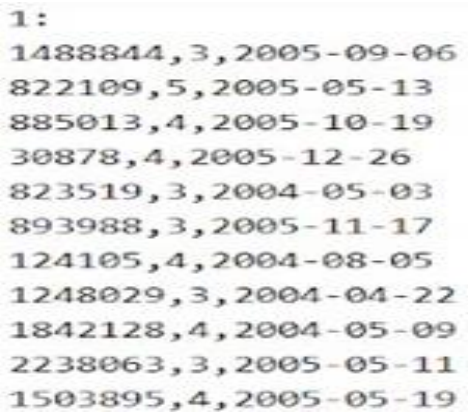

Fig 1 Dataset

ISSN No:-2456-2165

III. EXISTING SYSTEM

Netflix does not provide a content based approach, that is it does not show the similarmovie. If you watched some of the movies on other platforms like Amazon Prime, Hotstar, Voot etc then it does not show the same type of movie. Other than this Netflix internally uses Cinematch algorithm for the collaborative filtering which needed some improvement in the accuracy.

IV. OBJECTIVES AND CONSTRAINTS

Let's assume there is a Movie Mj that the user has not yet watched and the algo we build will try to predict how much the Useri will rate the movie. what rating to Movie jrij will be the predicted rating, assume it is then Netflix would recommend the movies to us. Our objective is to minimize the difference. Between rij and rij hat. That is actual rating and predicted rating we can measure this using Root mean squared error or mean absolute percentage error constraints are some for interpretability that is why such movie us recommended which is very important we do not need low latency system Netflix we would' pre compute what to recommend users in the hash table and precompute nightly basis means after 24 hours recommendation may change and always gets improved as more users more movies and better the algorithm.

V. PROPOSED SYSTEM

We build the front end for a content based approach so that users can watch the similar movie which they have watched earlier on any other ott platforms.For this we have thedata from imdb which gives the genre of movie and the actors name publisher name by which we can recommend the movie using the genre of the movie they have watched before.

In Collaborative Filtering given the data we have movies, user and we need to predict the rating which is standard recommendation problem in addition to this we can also see this as the regression problem because we want to predict rating between 1 to 5 that isour yi is 1 to 5 and we do not have xi if wesomehow come with best features then we can predict yi easily we will use xgboost forregression and we can use matrix factorization SVD for user user similarity and movie movie similarity we will use allof those to solve the problem.

VI. IMPORTING LIBRARIES

Pandas, Numpy, Matplotlib, Seaborn Scipy, Sklearn, Surprise date time libraries to knowhow much time it will take to run the code.

VII. EXPLORATORY DATA ANALYSIS

We want to build the CSV such as ui,mj,rij and data. For this we created a file data.csvinto which we will write. At first we will read all the files and store all the files into data.csv. We will look at the rating field using df.describe(). We are sorting all the data using data.df sort_values that is oldest date is the first entry and then it keeps increasing. Using df.describe().

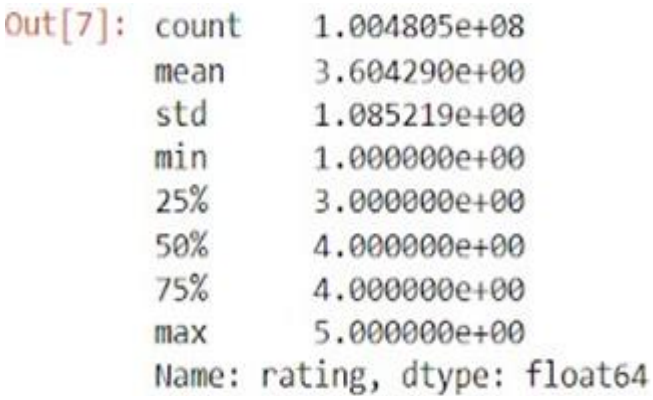

Fig 2 Describe

We came to know the max rating is 5 and the75th percentile is 4.

Next part of preprocessing is if there is any NaN values in the data using df.isnull().any() we found NaN values in ourdataframe

Then we checked if there were anyduplicates by any change using df.duplicated() And there were Zeroduplicates.

Now we performed some basic statistics toknow how many ratings, users & movies.

By using the simple command np.unique, on our dataframe. We also found 17,770 total no of movies.

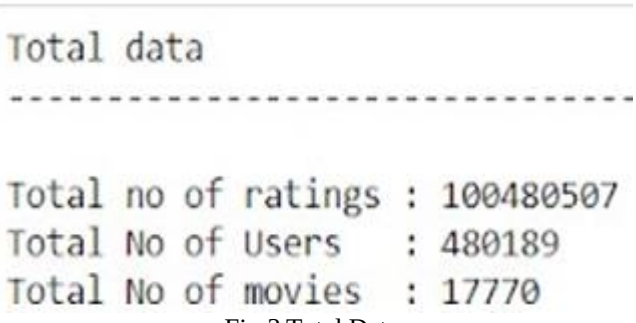

Fig 3 Total Data

Splitting date into train and test (80:20) how do we split the date, given the data we make our model to learn from the date and for future we want to predict the rating by deploying the model into productionisation so it makes complete sense that we have temporal structure for the data and we have data which is best so we took 80% of the train data & 20% as test data and on the basic of time axis we have split the data often 80:20 split then data in train has 80,384405 no of ratings 405031 users and 17434 movies as far as test data is unlearnedwe have 349312 users and 17577 movies and 100 million ratings are broken in 80millions in train and 20 million in test roughly.

ISSN No:-2456-2165

VIII. EXPLORATORY DATA ANALYSIS ON TRAIN DATA

Firstly we have seen the distribution of data based on the ratings 1,2,3,4 and 5

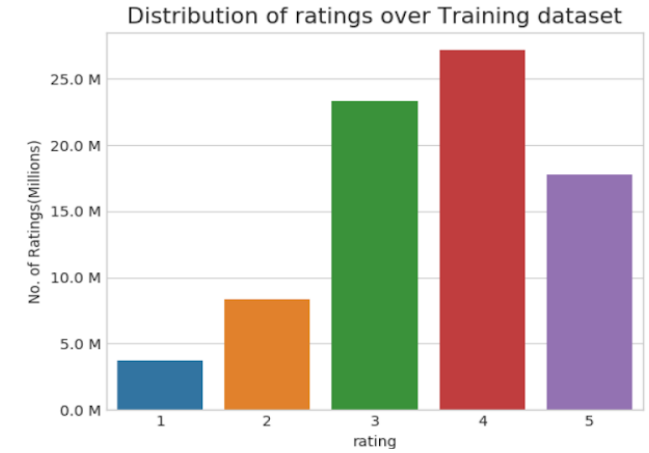

Fig 4 Training Dataset

From this we found 4 ratings is often ratedfrom this we come to know ratings are higher not lower.

Next thing is have added the column called day of week which will help us analyze better we plot the data from no. of ratings per months vs month we have data from 1999 up to 2006

Fig 5 No. of ratings

From this we come to know the amount of ratings have increased sharply and the ratings went close to 4.5 million per month in 2004 and 2005. That is a massive growth from this we came to know that we have awide spectrum of data in train and less in text. Now we analyzed the average rating given by the user it said used id 305344 hehas rated 17112 ratings. And user ID 1461435 has a rating of 9447.

As we were very curious about this data weplotted the PDF and CDF of the data then we came to know that most users give veryfew ratings and few give lots of ratings.

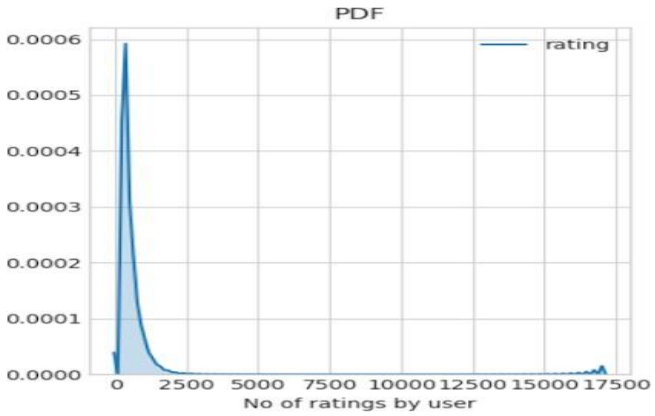

Fig 6 Ratings by Users

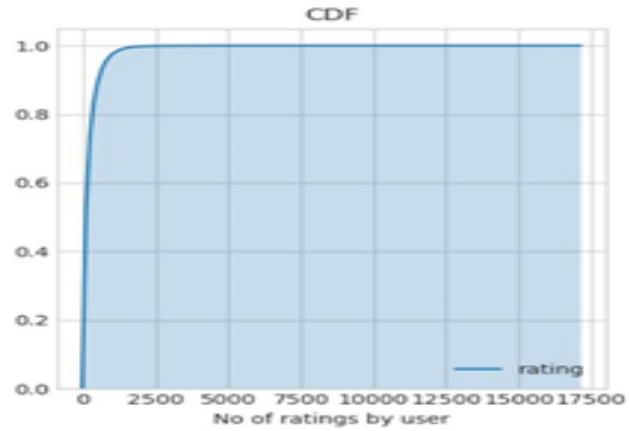

Fig 7 Ratings by Users

We found the average number of movies rated by users. 198 it was the mean which shows that people who use Netflix rate lotsof movies.

count	405041.000000
mean	198.459921
std	290.793238
min	1.000000
25%	34.000000
50%	89.000000
75%	245.000000
max	17112.000000

Fig 8 Netflix Rate

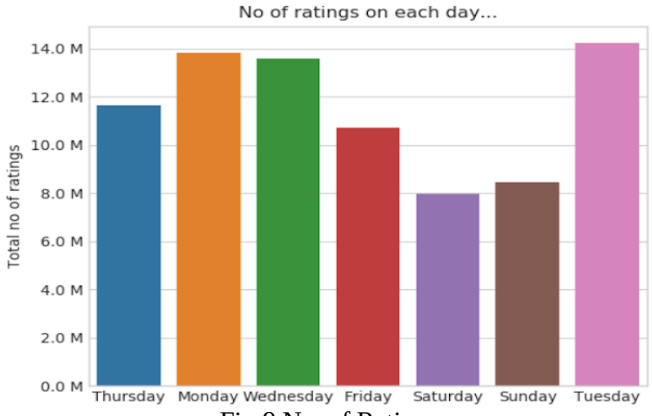

Fig 9 No. of Ratings

ISSN No:-2456-2165

Then we zoomed in the range of 75th percentile and max values and we plot thequantile.

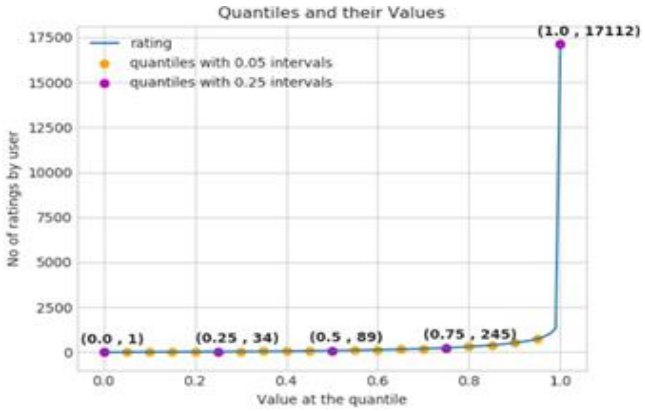

Fig 10 Ratings

Then we came to know that the 95th percentile is also very low so we zoomed in95th to 100 percentile values. So, we came to know how many ratings there were.

There were 20,305 ratings which was ok.

The median users rated 89 movies. So percentile of users rated below 89 movies and 50 percentile of users above 89 movies.

Then we plot numbers of ratings per weekday. And we found Saturday and Sunday traffic is much favored because on Sunday and Saturday people go for outings.

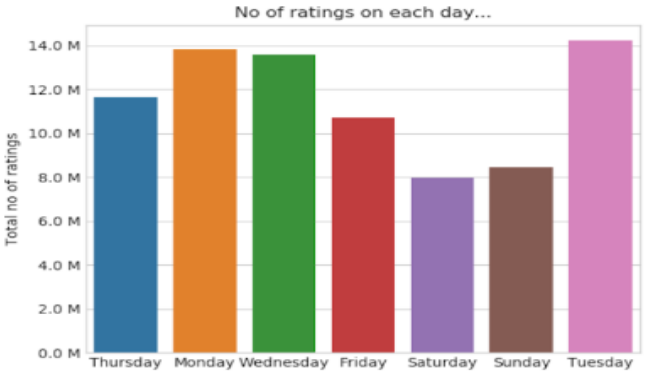

Fig 11 Total No. of ratings

IX. CREATING SPARSE MATRIX FROM DATA FRAME

We have table movie ID user id and rating new we can discard the date because it is ofNo use.

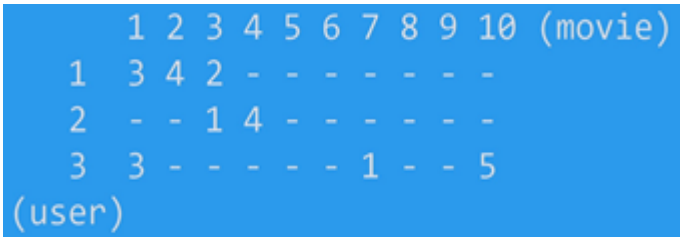

Fig 12 Data Frame

When I can rate some movies not all movies as most ratings do not exist it is a sparse matrix. Using spicy were we converted data to sparse matrices.

CSR matrix is the command CSR is compressed sparse row matrix. If the sparsity of the matrix is 99 percentile then there is no rating given there and 1% is nonzero rating..

We computed sparsity of the matrix on train data and we got 99.8 to percentile sparsity and on text data it is 99.95%. It shows datais extremely sparse.

X. FINDING GLOBAL AVERAGE OF ALL MOVIES RATINGS, AVERAGE RATING PER ZEROS, AND AVERAGE RATING PER MOVIES

We have a sparse Matrix. We computed themean of all ratings called the Global mean. Then we computed the average rating for users. Then we plot the average ratings per users and average ratings per movie. This tells us that this user is Critical or Lenient and whether the movie is super-hot or not.

XI. COLD START PROBLEM

If we slice a problem with the time as 80: 20split there might be some users who are present in train data and not in the test data.

There might be some users who joined late and also there might be new movies. For a person or user we have 1500 data where there is no data. This cold start problem in arecommendation system. And when we looked for the cold start problem with users it found that 15% of the users we are not present to new users. Total movies are 17770 and we have 17424 in trend data, thatis 346 movies did not appear in train data.

That is 1.95 % which is low. A cold start cankill our recommendation system so we have to keep these in the back of our mind.

XII. COMPILE SIMILARITY MATRIX

The training data has 405 K rows and 17k columns each column is movie and row is users. User-user similarity for these we have UI as sparse vector 17k dimension and assume u j has 17 k dimensions. And if we try to take similarity of Ui and Uj using cosine similarity By taking dot product between the two values that is Ui^T Uj That is is by using these we we find the top similar users it will took around 41 days to completethe training set so we will try to reduce the dimensions using SVD. So, it will speed up the process. Instead of 17k dimensions use SVD or PCA dimensions reduction technique to reduce dimension till 500 dimensions.

But now we are also taking the same time because after SVD we have dense matrices so now PCA and SVD are not working here.We are stuck in a big problem now.

ISSN No:-2456-2165

We maintain a binary vector for users, which tells us whether we already computed or not stop if not computed then compute the top 1000 most similar users for this given user , we add this to our data structure so that we can just access without computing it again.

In production time we have to recomputed similarities if it is computed a long time ago because users references changes over time if we could maintain some kind of timer so we have chosen here to make dictionaries of dictionaries which is like ki is user1 and similar users are stored in the values. This is a software engineering hack which could speed up things.

XIII. COMPUTING MOVIE : MOVIE SIMILARITY

Movies are 405k which is very large and this movie vector is very sparse. So, we can find it by the similarity Matrix movie i transpose of movie j.

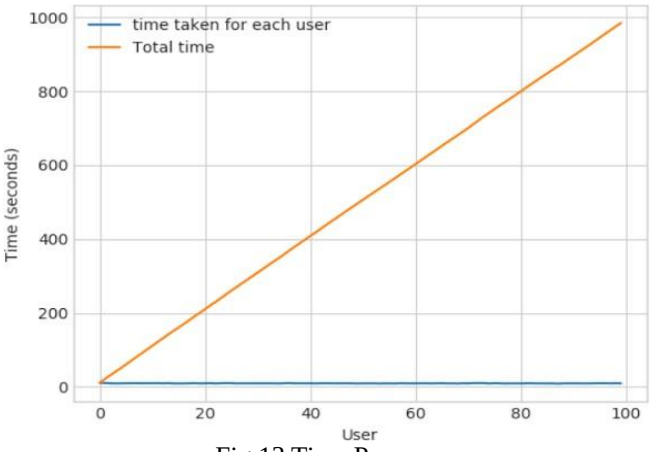

Fig 13 Time Per user

- From above plot, It took roughly **8.88**sec for computing similar users for one user
- We have 405,041 users with us in thetraining set.

405041×8.88=3596764.08sec=59946.068min=999.1 hours=41.629213889 days...

- Even if we run on 4 cores parallelly (a typical system nowadays), It will still take almost 10 and 1/2 days.
- Instead, we will try to reduce the dimensions using SVD, so that it mightspeed up the process...

This Matrix will be dense, which is 144 million computations, which is good. It tooknearly 10 minutes for us to find a movie we care for what is similar to that movie which we will solve as earlier with a dictionary.

Even though we have a similarity measure of each movie, with all other movies, We generally don't care much about the least similar movies.Most of the times, only top_xxxsimilar items matter. It may be 10 or 100.

We picked up movie vampire journals and we listed similar movies and then we computed a similarity matrix for Vampire generals. Then we found a hundred similar movies using cosine similarity, we found top movies similar to Vampire general were modern vampires, sleep Vampire etc. Whichare very similar movies.

Top 10 similar movies

```
movie_titles.loc[sim_indices[:10]]
```

movie_id	year_of_release	title
323	1999.0	Modern Vampires
4044	1998.0	Subspecies 4: Bloodstorm
1688	1993.0	To Sleep With a Vampire
13962	2001.0	Dracula: The Dark Prince
12053	1993.0	Dracula Rising
16279	2002.0	Vampires: Los Muertos
4667	1996.0	Vampirella
1900	1997.0	Club Vampire
13873	2001.0	The Breed
15867	2003.0	Dracula II: Ascension

Fig 14 Similar Movies to Vampiregeneral

XIV. MACHINE LEARNING MODEL

Surprise library makes your data handling very easy. If we give Data in triplet format. It handles everything for us. It provides various ready to use algorithms like KNN, Logistic regression, SVD,PMF,NMF. SurPRISE is a simple Python recommendation system engine.

Then we imported surprise, so install Surprise library code is

We designed the approach were we took the data and took a sample of data for training our machine learning models at first we came with 13 features such as

- *GAvg* : Average rating of all the ratings
- *Similar users rating of this movie:* sur1, sur2, sur3, sur4, sur5 (top 5 similar users who rated that movie..)
- *Similar movies rated by this user:* smr1, smr2, smr3, smr4, smr5 (top 5 similar movies rated by this movie..)
- *UAvg* : User's Average rating
- *MAvg* : Average rating of this movie
- *Rating* : Rating of this movie by this user.

ISSN No:-2456-2165

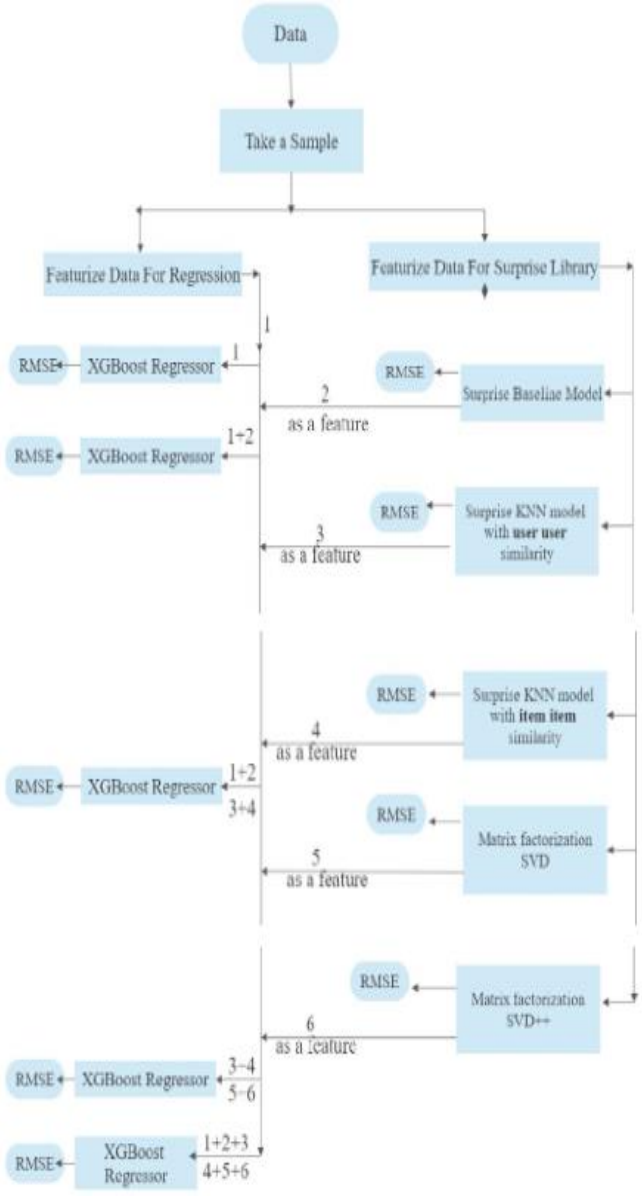

Fig 15 Movie Recommender System Approach

➢ *We Feature Data for Regression and use*

XGBoost regression with RMSE as error. Then we featured data for the surprise library and used the baseline model of the surprise library with RMSE. Using step 1 and 2 we again used XG boost regression with features along with RMSE. Then we use surprise KNN with user-user similarity and RMSE as error and again we use surprise KNN model with item-item similarity as step 4 combined step 1,2, 3 and 4 as the features and used XG boost regressor with RMSE as error. Then for feature 5 we use Matrix factorization SVDand for feature 6 we used matrix factorization svd ++ as a feature using all the feature sets 1 to 6 we implemented XG boost regression and RMSE as the error.

- *Step 1*

Is to sample the data our train has 405k * 17k and test has 349k * 17k so, we use 10k users and 1K movies as train data and 5K users and 500 movies as tests. And then we work on the model and find which model is best and then use that model for allthe data.

- *Step 2*

We used 13 features for the xgboost regression such as Global average, similar user rating of this movie , Users average rating, average rating of the movie, rating of the movie by users.We need to transform data for a surprise model, then we applied an actual machine learning model XG boost with 13 features. We trained with two error matrices RMSE and MAPE.

XV. XGBOOST WITH 13 FEATURES

We got RMSE as 1.076 and MAPE as 34.50 and the most important feature is users average score and second is movingaverage.

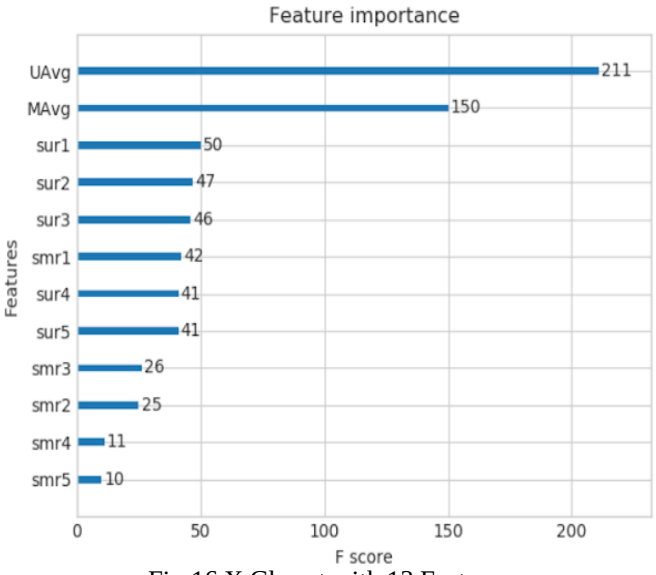

Fig 16 X Gboost with 13 Features

In surprise, the baseline model method is SGD and the learning rate is 0.001. We wantto minimize the difference of rating actual and predicted. We are using L2 regularization. The RSME here is 1.073 and MAPE 34.04 and we had 13 features RMSE 1.076 and now we got slightly better.

Now, we will mix 13 features and the baseline model. We have 13 features and the 14th feature will be BSIPR . It is the outputof the baseline model, and then apply XGboost on top of this. RMSE is 1.076 and MAPE is 34.49, and BSIPR is a list of important features we came to know.

Surprise KNN is our next tape which is internally using similar users and similar movies. k is similar to users of user (U) andwho rated movie (i) it is the cosine similarity. But we use here baseline pearsoncorrelation Coefficient and now we have RMSE is 1.0726 and MAPE as 35.02 , then we use Matrix factorization technique in which RMSE is 1.072 and MAPE is 35.01 And test is lowest among all that wetrain till now, then we went with svd++ inwhich RSME is 1.072 and MAPE is also same.

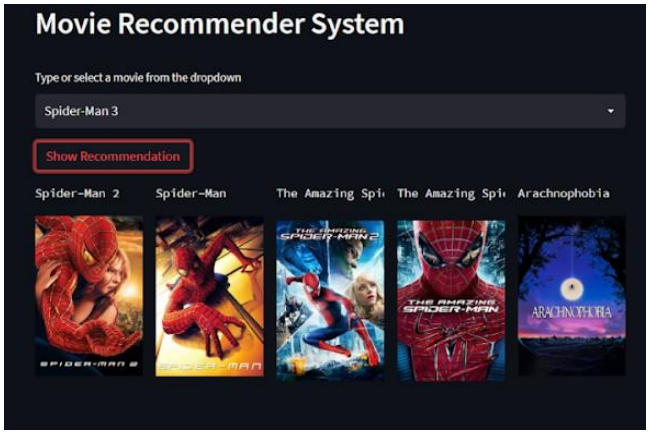

Fig 17 Content Based Movie Recommendation

In collaborative filtering SVD has lowestRSME among all the models.

4.5 Comparision between all models

```
pd.DataFrame(models_evaluation_test).to_csv('sample/small/small_sample_resu
models = pd.read_csv('sample/small/small_sample_results.csv', index_col=0)
models.loc['rmse'].sort_values()
```

```
svd              1.0726046873826458
knn_bsl_u        1.0726493739667242
knn_bsl_m         1.072758832653683
svdpp            1.0728491944183447
bsl_algo         1.0730330260516174
xgb_knn_bsl_mu   1.0753229281412784
xgb_all_models    1.075480663561971
first_algo       1.0761851474385373
xgb_bsl          1.0763419061709816
xgb_final        1.0763580984894978
xgb_knn_bsl      1.0763602465199797
Name: rmse, dtype: object
```

Fig 18 Result

XVI. RESULT

We designed the front end for a Content based Recommendation system. we can select the movie from the dropdown or we can even search the movie name for e.g if we select spiderman and then click on show recommendations it shows all the similar movies related to spiderman or type of the spiderman.

XVII. CONCLUSION

In this paper we used content based filteringas well as collaborative based filtering to improve the recommendation system.

Content based filtering is working with streamlit successfully and recommending the similar type of movies. In Collaborative filtering Lower the RMSE the better model is we sorted all the model and among all of them best was the SVD with 1.0726 RMSE34, and all values are very close but when we compared with the percentage difference it is 0.35% improvement which is also very good in terms of All the data, as soon as we train large data set we will get the bestresult.

REFERENCES

[1]. C. S. M. Wu, D. Garg, and U. Bhandary, “Movie Recommendation System Using Collaborative Filtering,” In 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS),pp. 11-15, IEEE, 2018 Nov.

[2]. R. E. Nakhli, H. Moradi, and M. A. Sadeghi, “Movie Recommender System Based on Percentage of View,” In 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI), pp. 656-660, IEEE.

[3]. H. W. Chen, Y. L. Wu, M. K. Hor, and C.Y. T ang, “Fully content based movie recommender system with feature extractionusing neural network,” In 2017 International Conference on Machine Learning and Cybernetics (ICMLC) , vol. 2, pp. 504-509,Jul, 2017, IEEE

[4]. Seroussi Y., “Utilizing user texts to improve recommendations,” User Modeling, Adaptation, and Personalization, pp.403–406, 2010.

[5]. Buttler D., “A short survey of document structure similarity algorithms,” in Proceedings of the 5th International Conference on Internet Computing, 2004.

[6]. Goldberg D., Nichols D., Oki B. M., and Terry D., “[Using collaborative filtering to weave an information Tapestry],” Communications of the ACM, vol. 35, no.12, pp. 61–70, 1992.

[7]. Beel J., Langer S., and Genzmehr M., “Mind-Map based User Modelling and Research Paper Recommendations,” in workin progress, 2014.

[8]. MacQueen J... Some methods for classification and analysis of multivariate observations. In Proc. Of the 5th Berkeley Symp. On Mathematical Statistics and Probability, pages 281-297. University of California Press, 1967.

[9]. Ball G. and Hall D... A Clustering Technique for Summarizing Multivariate Data. Behavior Science, 12:153-155, March 1967. Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems.

[10]. Choi, Sungwoon, et al. "Reinforcement Learning based Recommender System using Biclustering Technique." arXiv pre printarXiv:1801.05532 (2018).

[11]. G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” IEEE Trans. Knowl. Data Eng., vol. 17, no.6, pp. 734–749, (2005).

[12]. P. Resnick, H.R. Varian, “Recommender systems,” Communications of the ACM 40(3) (1997) 56–58.