MAJOR PROJECT

PIPELINE –

Major Problems and their Solution Approaches—

1. The Dataset is too large and Sparse. Sampling helped wherever needed.
2. Visualization of the dataset is also a problem since the dataset is very large. Using some useful libraries like DataPrep.
3. We are unable to train many models on the dataset due to their high time complexity. Not able to take average linkage in hierarchical clustering .
4. Unable to train DBSCAN as of big dataset.
5. I had two options. Either I do the oversampling .Like if a movie has three genres then I can make three rows and assign a genre to each row. But that leads to an almost three times bigger dataset. So I made a new column of a particular genre and assigned 1 if the movie is having that specific genre. For it first I am finding all unique genres by splitting the genre list .
6. Due to large datasets and a slow model like decision tree we cannot train the huge amount of data. If we can increase the training set size to about 20 Lakh and number of unique models at larger values we can easily get the intersection out of it(movies recommended by more number of classifiers trained on independent sets in sorted order) and use it intersection to recommend movies to the user
7. Made a different type of dataset for every different type of algorithm .
8. Other problems are specified in detail in the report.

Outline-

9. Algorithms used: **sorting , collaborative filtering(from scratch to get desired result), venn diagram, correlation, k means , hierarchical , dbscan(unable to use because of high large data), made a new algorithm by combining three to four concepts of correlation clustering collaborative , elbow method , neural network(rbm) , decision tree(random forest based), xgboost, surprise algorithm, knn, svd,sampling,reader.**

Preprocessing of data & Data Analysis :

Preprocessing of df_movies dataset.

In our df_movies dataset we were having 3 columns ‘title’ , ‘movieId’ , ‘genre’

1st problem:

- In this we found in the genre list the genres are in string format so first we traversed the string and then divided the string into multiple genres. And got a list of multiple genres.

Challenge :

- Then I had two options. Either I do the oversampling .Like if a movie has three genres then I can make three rows and assign a genre to each row. But that leads to an almost three times bigger dataset.
- So I made a new column of a particular genre and assigned 1 if the movie is having that specific genre. For it first i am finding all unique genres by splitting the genre list .

2nd Problem:

- We were having some entries that had the same title but different genres. So we are checking if the same title has different genres. Then it is a fault in the dataset. so we are combining there genres
- if a movie A having x,y genre and movie B having y,z genre and we will assign A and B x,y,z genres
- By this we didn't oversample. And also handled the problem.

3rd problem:

- In this we just separated the year of the movie from their title. And made a new column of years of movie.

Challenge:

- Strings were not able to be concat so i just traversed the strings iteratively and found the required results.

Preprocessing and Analysis of ratings & links dataset:

- As we have imported many datasets for our movie recommendation system, we are going to preprocess all the different datasets.

- For preprocessing of the ratings, we basically check if there are any null values present in the dataset, we also check all the unique values to see the range in which the values of rating are present.

- Similarly we do the same things for all the other datasets.
- In the df_links dataset, we found that there were X null values, as the dataset was very large, dropping a few rows would affect the dataset that much.
- So we dropped all the rows with the null values in df_links.
- Similarly, for the df_ratings dataset, we can see that the unique value of the userId is far more than the unique Rating and moviesId.
- So a single user gives 1 or more than 1 rating in this dataset.We will use this information later when designing the prediction algorithm for our dataset.
- Then in this dataset we have extracted year-month data from the timestamp column similar to the above case.
- In this dataset there are about 2 Lakh unique users with a dataset size of about 2 Crore.
- Now after doing this I have created the movie based rating dataset. In this dataset we have movies directly linked to their average ratings.
- After this we have calculated the count of times a movie is watched and map it to the movie-rating dataset. So finally we got a dataset with movies with their average ratings and count of the number of times it is watched.

Preprocessing and Analysis of tags dataset:

- In this dataset we first converted timestamp to datetime. Doing this we will be able to extract the information regarding when the movie is watched by a particular user.
- Now we know that time is not very useful but year and month when the user is watching the movie can be useful. So we have extracted the year and month in separate columns and deleted the datetime column.
- Then using the pieplot we have analyzed the demographics of the viewers in particular year and different months of the year.
- Then I have checked the number of unique values in different columns and by this we found that there are about 30000 unique values in the tags

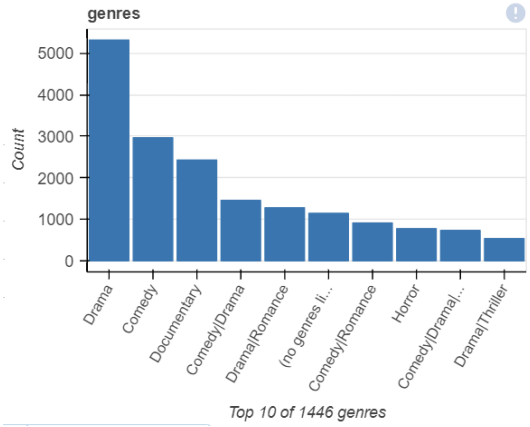
column. Now if we slice the words and do the oversampling there will not be much gain. Also, we cannot make columns to track a unique tag.

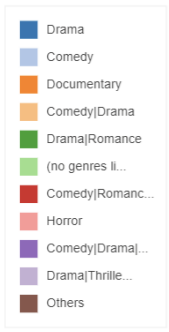
- So, due to the above problems encountered we have directly encoded the tag data. Also, we have encoded the year column (when a particular user is watching the movie) since there will be no data loss in doing this.

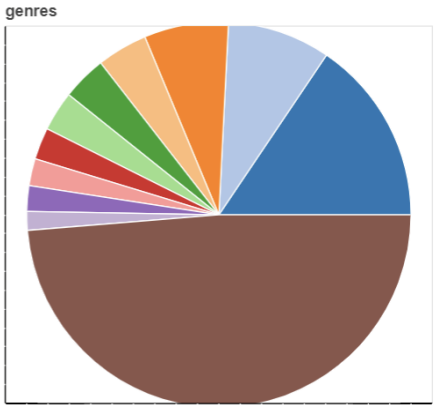
Combining the datasets analyzed above and creating new datasets:

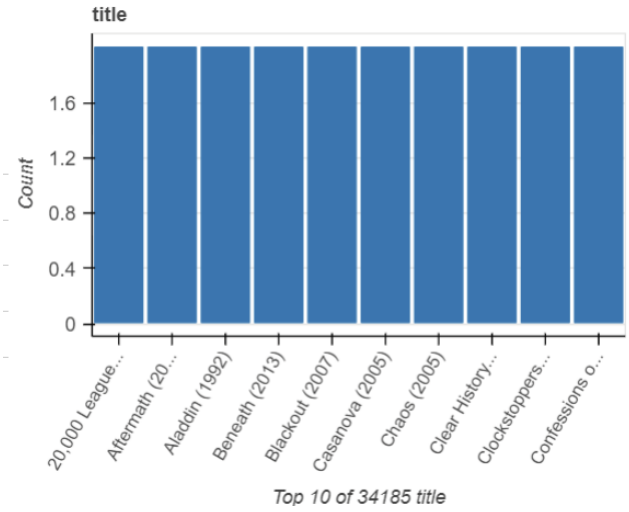
- Here we have created two new datasets each of movie and user based.
- In the movie based we have only 1 row corresponding to each movie which gives its information such as genre, year of release, average rating, etc.
- In the user based dataset we have a record of each user corresponding to each movie.

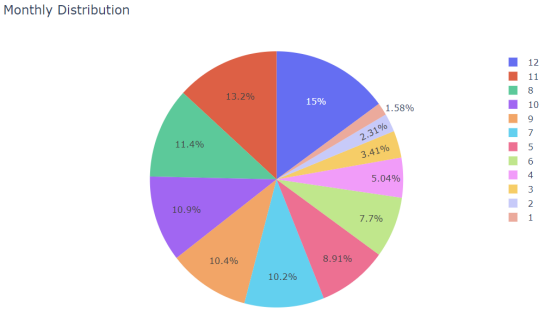
PLOTS :

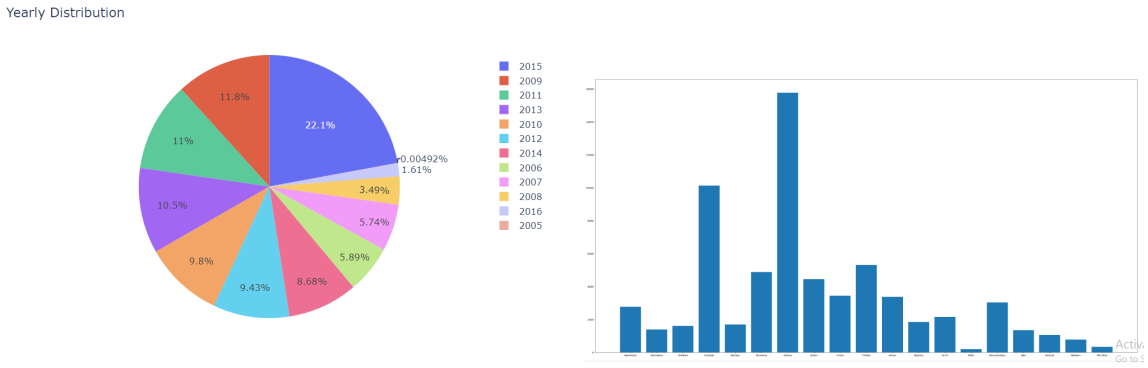












Making the Model from Scratch :

1. <u>(A general recommendation)-</u>

- First we will make a general recommendation like if a new user login is happening. then we will recommend to him the best movies based on the rating and releasing date of that movie.
- American History X (1998) , 2001: A Space Odyssey (1968) ,Minority Report (2002) ,Truman Show, The (1998). (these are the output for this). And all of these movies have imdb rating **approx 9 out of 10.**

2. <u>(Genres based recommendation)-</u>

- Now we made a new recommendation system if a user is giving us his favorite multiple genres. For this we are using the concept of **venn diagram.**
- In this we are applying a for loop and finding the dataset movies in which the given genre is equal to 1 then we are just finding all the dataset.and then using sorting algorithm to give priority to best movies based on their reviews, ratings, etcetera. And taking their intersection in the venn diagram .
- Then again using a sorting algorithm to find the best movies among them.
- Now we got the movieid of movies. Now we will find the title that corresponds to that movie.
- These are the movies I am getting for genres [‘adventure’, action , children].
- Toy Story 2 (1999) ,Finding Nemo (2003) Monsters, Inc. (2001) Shrek (2001) , Lion King(1994) ,Aladdin (1992) , Toy Story (1995)

3.<u>(User and movie based recommendation with correlation concept)-</u>

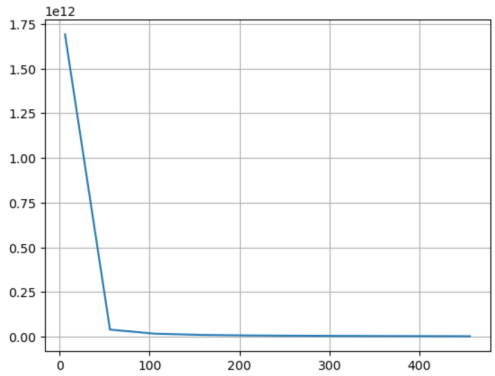
- Now we made a recommendation system in which we will give the user name and movie name and then the function will recommend the movies based on that movie.
- For this we applied several concepts. **We used correlation, pivot tables** for this. In this we are creating a new dataset. In which the columns are the movie names, and the rows are the user id and the entries are the ratings. So basically we are checking that a particular movie is watched by how many users and how it is rated.
- So we are picking the given movie. And finding the correlation with other movies(this is based on rating and users). Then we sorted the movies based on their correlation and gave the top 10 movies as the recommendation to the user.
- **Giving less priority to genres. As if we are considering the user , the genre's data is already with us**.
- These are the movies we recommended for jumanji movies.
- Treasure Planet (2002) ,Three Wishes (1995), The Little Rascals(1994) ,The Lost Weekend(1945) ,Ella Enchanted (2004) ,School Ties (1992),The Brave Little Toaster (1987).

4. Clustering based (Movie based recommendation).

- In this we used **unsupervised learning as a subpart of the function for movie recommendation** like if a user is giving us his favorite movie. Then we are finding the movies related to this by clustering algorithm. In the previous part we found it using correlation. Now we are using clustering.

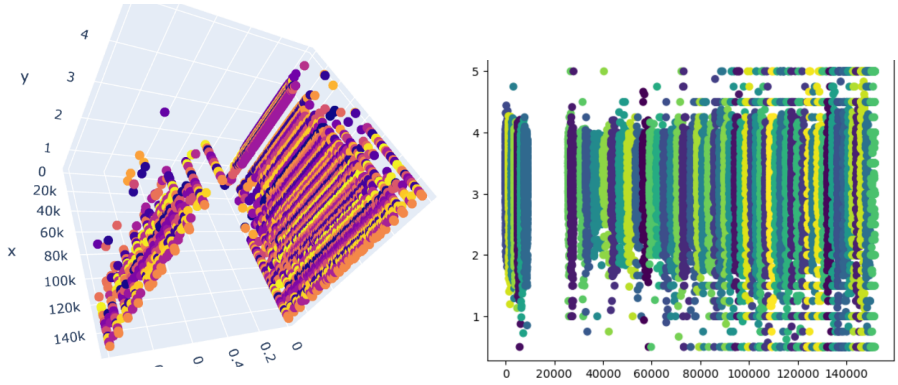
Elbow method:

- In this basically we are training our model on the dataset. And then finding the value of inertia for the clusters and plotted them against no. of clusters . and the no. of clusters where the elbow is coming is our best no. of clusters.



We can see 60 to 65 is the best no. of clusters for our model.

K Means based model:

- First we have normalized the rating count column between 0 to 5. Then trained the model.
- So we first applied the elbow method in which we got the best no. of clusters and got this graph.
- Now we implemented the K Means algorithm and got the classes corresponding to each movie.
- And now created a new dataset with a new column of classes.
- Now for a given movie we are finding its corresponding movies and sorting those based on their rating counts and rating and releasing year. Thus we are recommending a movie based on this.
- The movies based on jumanji -> 'Toy Story (1995)', 'Grumpier Old Men (1995)', 'Waiting to Exhale (1995)', 'Father of the Bride Part II (1995)', 'Heat (1995)', 'Sabrina (1995)', 'Tom and Huck (1995)', 'Sudden Death (1995)', 'GoldenEye (1995)', 'American President, The (1995)', 'Dracula: Dead and Loving It (1995)', 'Balto (1995)', 'Nixon ',
- 

- ```
  The Silhoutee Score for K-means  is  0.5473765159015384
  The value of calinski_harabasz score for K-means on PCA reduced dataset is  175477.16406234386
  The value of davies_bouldin score for K-means on PCA reduced dataset is  0.8580750002841878
  ```

Hierarchical clustering:

- In this we are training our hierarchical model. And doing the same approach. And getting the results.
- Here we are normalizing our rating count column as this is leading to a very bad clustering. So we used the rating count column as unchanged
- The silhouette score is 0.3091
- The calinski_harabasz score is 42496.82
- The davies_bouldin score 1.30
  ```

- Toy Story (1995)', 'Grumpier Old Men (1995)', 'Waiting to Exhale (1995)', 'Father of the Bride Part II (1995)', 'Heat (1995)', 'Sabrina (1995)', 'Tom and Huck (1995)', 'Sudden Death (1995)', 'GoldenEye (1995)',

5. Collaborative Filtering(user based recommendation).

- in this function we are taking user id as input. and finding how many movies this user watched and then finding the users who have watched the same movies. and picking a user with the highest watched similar movies.
- then we are finding the last movie watched by another user and highly rated by him. then we are **using the clustering algorithm** to cluster that movie with another movie and finding the top movies in that cluster and recommending those movies to the user.
- We are also using the concept of venn diagram like we have used the concept of correlation and clustering to find the best movies based on those movies so we are finding all those movies and recommending the intersecting movies. to the user.
- So in this model we have combined the concept of collaborative filtering , clustering, correlation , sorting etc.
- Movies for user no. 11846. ['Toy Story (1995)', 'Grumpier Old Men (1995)', 'Waiting to Exhale (1995)', 'Father of the Bride Part II (1995)', 'Heat (1995)', 'Sabrina (1995)'

Part b

- In this we are recommending movies by the same algo but after getting movieId we are recommending movies based on the movie_based function that we gave above. Pacifier, The (2005),Frank (2014),You're Next (2011),Five, The (Gonin) (1995) ,Fletch (1985),Phoebe in Wonderland (2008)

6. Surprise Algorithm(Rating based recommendation)-

- In this algorithm, we are basically training the algorithm on the whole dataset and then testing the values and using that to predict the movies for all the users.
- In this we have implemented the Surprise algorithm, and from scratch we have built the movie recommendation system which will work for our dataset.
- We run this algorithm on the ratings dataset, we have already preprocessed this dataset.

- Now, first as we are using a custom dataset for the surprise library we are going to first define the reader format for that, as our dataset is not similar to the default dataset used by the surprise library.
- Now that we have customized our surprise library to work on our dataset, we are gonna build our algorithm now.
- Before that we are going to random sample our data,using the .sample library.
- Now we create a class name Ownsurprise, in it we first define the baseline algorithm we are going to use, we also create a prediction list in which we are going to store our predictions.
- We now initialize a defaultdict, in this we are going to store all the userId as keys and for values we are going to recommend each of them n movies. Now we define our fit function, in which we basically fit our training dataset using the .fit function, we also predict the values and store it in the predictions list using the test function.
- Now we define another function top_predict, in this function we basically pass the number of movies we recommend to each user, the default value for this taken as 5. Now we are going to traverse the predictions list we created, in this for every userId we are appending a unique InnerId that is ItemId in this case and all the ratings that were predicted for that user.
- Now, we iterate through the dictionary top_pred, in this we are basically sorting all the user ratings using the 1:1 sorting algorithm in descending order.
- Now we define the show function, in this we are basically, taking a UserId as an input and for that user we are printing all the recommended movies that our algorithm predicted and we are also storing that in a list. We basically did this: we iterated through our top_pred dictionary, if the key was found we retrieved all the values stored in that key, else we printed the user not found.
- Now at last we define a movies_recommend function this is basically going to recommend the user the movies based on Name basis, at first it was only returning the movie index's.
- Now in this function we basically use the stored movie Id for each user and use it to iterate through our merged dataset that we have created.Using the movie index's we are retrieving the data and printing it for the user.

SVD Algorithm:

- Now we have used the Surprise algorithm using 2 base line algorithms, first we have used SVD in this.
- In this algorithm we are basically calculating all the ratings provided by each user. Using that information we are creating a pattern on what type of rating movies our user is watching, it also comparing the ratings of other user's which have similar taste to this user, now for the movies this user has not watched yet, it is predicting different ratings based on this pattern and suggesting the top n rating movies which the user will likely watch.
- Strawberry and Chocolate (Fresa y chocolate) (1993),Secret Garden, The (1993),Wild Bunch, The (1969).

```
The movies recommended are : Strawberry and Chocolate (Fresa y chocolate) (1993)

The movies recommended are : Secret Garden, The (1993)

The movies recommended are : Wild Bunch, The (1969)
```

-

KNN Algorithm:

- Here we are using KNNWithMeans Algorithm, which is part of the KNN algorithm. In this case it takes into account the mean rating of each user.
- In this algorithm, we are basically computing all the ratings provided by each user and based on that data, we are using test data,which replaces the user interactions with a negative interaction and it predicts the rating for that.
- Using that predicted rating, it gives us the top nearest movies recommended to use.
- Strawberry and Chocolate (Fresa y chocolate) (1993),Insider(1999),Godfather The(1972)

```
The movies recommended are : MÃ©nage (Tenue de soirÃ©e) (1986)

The movies recommended are : Godfather, The (1972)

The movies recommended are : Insider, The (1999)
```

-

XG-Boost Model:

- In this model we have used the XBBoost Classifier for the classification task included in the code.
- Here I have started with first upgrading the movies dataset to make it suitable for this method. Firstly we have checked for null values in the dataset and then replaced them with the 0. Then I have dropped the unnecessary columns. Then divided the dataset into x and y sets and then making the train and test sets. Here y set comparises of movieId column.
- After this we have scaled the columns of the dataset to apply the xgboost algorithm on it. Scaling makes the features equivalent and reduces the biases towards the features.
- Then using the inbuilt XGBoost model in the sklearn library we have trained the model on the training set. After this using the test set data we have predicted the movie for the given input based on our trained model.
- **Limitations**– Cannot train large dataset due to limitation of memory and time. This algorithm takes lot of time for larger datasets and for even larger it consumes 30GB RAM(upper limit available memory).
- Also we have not trained the model n-times then taken the intersection due to the above limitations. But by doing these upgrades we can reach another milestone in this task.
- Lamerica(1994),Two if by Sea(1996),In the Bleak Midwinter(1995),Catwalk(1996)

movieId	title
53	Lamerica (1994)
53	Lamerica (1994)
53	Lamerica (1994)
53	Lamerica (1994)
53	Lamerica (1994)
...	...
64	Two if by Sea (1996)
64	Two if by Sea (1996)
106	Nobody Loves Me (Keiner liebt mich) (1994)
96	In the Bleak Midwinter (1995)
108	Catwalk (1996)

Decision Tree Based Model:

- This model is a user based model which only takes the userId of the known user and predicts movies for this user. The movies which are recommended are different from the movies which are already watched by the user.
- For this task we have taken the 2Crore user dataset and merged it with merged movie based dataset and then merging this dataset with updated movies datasets. This gives us a modified dataset which will be required further.
- Then checking this dataset for any null values which may be added due to merging and deleting those rows.
- Finally I have created a function that takes this dataset, another dataset which is required to extract movie name from the movie Id and a user Id of the person to which we have to recommend the movie.
- The upgraded model which I have created can be somewhat related to the Random Forest Classifier model but have some differences. Due to the large dataset I have done the sampling of the dataset for each running unit. Due to this large amount of data is lost (2 Crore to 0.5 Lakh).
- Due to this loss the different Decision tree train, many times gives the intersection phi of the movies commonly recommended by the different classifiers trained on different sets.
- After training I have used the already available data of the asked user to create the testing set but modified certain values such as rating, user watched count, etc. This modification will allow us to get a good movie to be recommended to the user. Then after these I have combined the movies given by the classifier to get the set of movies which are not watched by the user and can be recommended.
- **Limitations**— Due to large dataset and a slow model like decision tree we cannot train the huge amount of data. If we can increase the training set size to about 20 Lakh and number of unique models at larger values we can easily get the intersection out of it(movies recommended by more number of classifiers trained on independent sets in sorted order) and use it intersection to recommend movies to the user.
- The Shawshank Redemption(1994),Babai(2015),The Third Man(1949),Einstein(2015),Seven Samurai(Shichinin no samurai)(1954),The Battle of Chernobyl(2006),Secrets & Lies(1996)

	movieId	title
0	318	Shawshank Redemption, The (1994)
1	137612	Babai (2015)
2	1212	Third Man, The (1949)
3	132448	Einstein (2015)
4	2019	Seven Samurai (Shichinin no samurai) (1954)
5	141373	The Battle of Chernobyl (2006)
6	1041	Secrets & Lies (1996)

	movieId	title
0	126967	Letter from Siberia (1957)
1	146335	Sem Pena (2014)
2	100315	2 Become 1 (Tin sun yut dui) (2006)
3	109529	Everybody Street (2013)
4	133323	The Fruit Hunters (2012)
5	146016	Elder Sister (1966)

Neural Network Based Model:

- In this model we first create a dataset of movie Id which is index using an index column to get id representation in a sequence. Here I am training a boltzmann based model to which I will be giving a data row for movie data which contains useful information about the data. Here I have used a column which is the project data of rating and count.
- Then merging this dataset with the user 2 Crore dataset to get it linked to user data for the movie. Then grouping the dataset on the basis of user Id using the groupby function.
- Then creating the training list in which each value is a list containing a value for a particular user filled rest all zero. The number of lists is equal to the number of rows in the dataset and each list contains elements total equal to the number of users in the dataset.
- Then we have created the placeholders of tensorflow which include biases of hidden layer, biases of visible layer and weight matrix.
- Then we set the activation functions relu and sigmoid. Here we are creating the tensorflow placeholder for the input data and then computing the state of the hidden layer state using the activation function. Also there we compute visible layer state using the hidden layer activation.
- Then we set RBM parameters, functions and error functions like setting the learning rate,creating methods to update weights and biases, etc.
- Then we are initializing the variables used in the further program. Then we create a session and initialize it using the globat_variable_initializer. After this we run our program for some fixed number of epochs, fixed learning rate and fix batch size to train the neural network.

- Now to recommend the movie we arrange the movies based on the recommendation score which is calculated based on the v0 value as the train_list of the user to be recommended.
- Finally arranging this recommendation score in descending gives us the movies to be recommended in a sequence.
- **Limitations**– Cannot train large dataset so taken only 1000 movies and limited set of user data for them. If we train this model on large dataset we get better results.
- Toy Story(1995),Jumanji(1995),Grumpier Old Men(1995),Waiting to Exhale(1995),Father of the Bride Part II(1995)

	movieId	rating	index	r_score	title
0	1	3.894802	0	0.201118	Toy Story (1995)
1	2	3.221086	1	0.045221	Jumanji (1995)
2	3	3.180094	2	0.036187	Grumpier Old Men (1995)
3	4	2.879727	3	0.004789	Waiting to Exhale (1995)
4	5	3.080811	4	0.032233	Father of the Bride Part II (1995)

Recommended Movies by Each Model:

Input for all the models are not the same.

Algorithms	Movies Recommended
A general recommendation	• American History X (1998) , 2001: A Space Odyssey (1968) ,Minority Report (2002) ,Truman Show, The (1998).
Genres based recommendation	• Toy Story 2 (1999) ,Finding Nemo (2003) Monsters, Inc. (2001) Shrek (2001) , Lion King(1994) ,Aladdin (1992) , Toy Story (1995)

User and movie based recommendation with correlation concept	• Treasure Planet (2002) ,Three Wishes (1995), The Little Rascals(1994) ,The Lost Weekend(1945)
Clustering based (Movie based recommendation)(Kmeans)	• Toy Story (1995)', 'Grumpier Old Men (1995)', 'Waiting to Exhale (1995)', 'Father of the Bride Part II (1995)', 'Heat (1995)', 'Sabrina(1995)'
Clustering based (Movie based recommendation)(Hierarchical clustering)	• Toy Story (1995)', 'Grumpier Old Men (1995)', 'Waiting to Exhale (1995)', 'Father of the Bride Part II (1995)', 'Heat (1995)', 'Sabrina (1995)'
Collaborative Filtering(user based recommendation)(Part a)	• Toy Story (1995)', 'Grumpier Old Men (1995)', 'Waiting to Exhale (1995)', 'Father of the Bride Part II (1995)', 'Heat (1995)', 'Sabrina (1995)'
Collaborative Filtering(user based recommendation)(Part b)	• Pacifier, The (2005),Frank (2014),You're Next (2011),Five, The (Gonin) (1995) ,Fletch (1985),Phoebe in Wonderland (2008)
Surprise Algorithm(Rating based recommendation)(SVD)	• Strawberry and Chocolate (Fresa y chocolate) (1993),Secret Garden, The (1993),Wild Bunch, The (1969).
Surprise Algorithm(Rating based recommendation)(KNN)	• Strawberry and Chocolate (Fresa y chocolate) (1993),Insider(1999),Godfather The(1972)

XG-Boost Model	• Lamerica(1994),Two if by Sea(1996),In the Bleak Midwinter(1995),Catwalk(1996)
Decision Tree Based Model	• The Shawshank Redemption(1994),Babai(2015),The ThirdMan(1949),Lies(1996)
Neural Network Based Model	• ToyStory(1995),Jumanji(1995),Grumpier Old Men(1995),Waiting to Exhale(1995),Father of the Bride Part II(1995)

Individual contribution:

Nikhil Agrawal(B21EE043):

- Ideation and path planning
- handled and preprocessed the DF_movies dataset.
- Models trained and analyzed:
  - General Recommendation Model (useful for new user)
  - Genres based recommendation Model.
  - Users and Movies based recommender system using correlation concept(scratch) .
  - Movies based recommender system using clustering(k means, hierarchical) and sorting.
  - User based Collaborative filtering model combining clustering algorithms , sorting algorithm, venn diagram . (scratch)

Rahul Gurjar(B21EE055):

- Ideation and Path planning.
- Preprocessed and analyzed ratings and tags dataset.
- Models trained and analyzed:
  - XGBoost based Model
  - Decision Tree based Model (Random Forest Type)
  - Neural Network Based Model (RBM)--Generative Probabilistic Model

Priyam Gaurav(B21EE092):

- Ideating and Path planning.
- Preprocessed and analyzed ratings,link dataset.
- Models trained and analyzed:
    - Surprise Model
    - SVD
    - KNN(With Means)
    - Implemented A movie recommendation Model using above 2 algorithms at a time.