Graph Based Recommendation System for Movies

Gaurav Kolhe
G01032413
Dept. of Electrical and Computer Engineering
George Mason University
Fairfax, USA
gkolhe@gmu.edu

Sameer Kumar
G01099808
Dept. of Applied Information Technology
George Mason University
Fairfax, USA
skumar26@gmu.edu

***Abstract*—This proposed work consists of the procedure, elaboration, methodology and conclusion of a recommender system for movies. There is a rapid growth of data content from different social and entertainment networks. Recommendation systems are typical information filtering applications to generate better results and relevant content based on search query. In this paper, the technique has been implemented on a movie dataset acquired from IMDB. Transformation to bipartite graph to allow ease of graph-based algorithms for the sake of implementation. Different types of filtering procedures like Collaborative, Demographic and Content-based will be taken into consideration in-conjunction with the graph transversal algorithm. The work will further delve into discussing the results and the conclusion of the attempted filtering procedures to list the advantages and drawbacks of each method, which will be utilized to further improve the work in this area**

***Index Terms*— Movie, Recommender System, Collaborative, Content-based, Demo-graphic, Machine learning, Visualization.**

I. Introduction

In recent times, social media networks and entertainment channels are producing massive amounts of data content across the entire globe. These huge volumes of content complicate the task of choosing a favorite movie. One simply cannot hand pick interesting and relatable content based on choice. There are infinite options to choose from. Therefore, we need a supervised system which can filter out the irrelevant content and give relatable suggestions to the user. Recommender systems solves this problem and provides best suggestions to user based on their preferences and existing data. There are numerous e-commerce sites which implement this technique to provide user their definite search results. Netflix implements such a technique to provide similar genres movies to user based on their watchlist or search history. Amazon is second on the list which fulfills this need by overlapping the recommender system for product suggestions rather than movies. YouTube auto suggests next song in the queue based on the current song type. Social networking also administers similar recommender system where the user gets friend suggestion based on mutual friends or location.

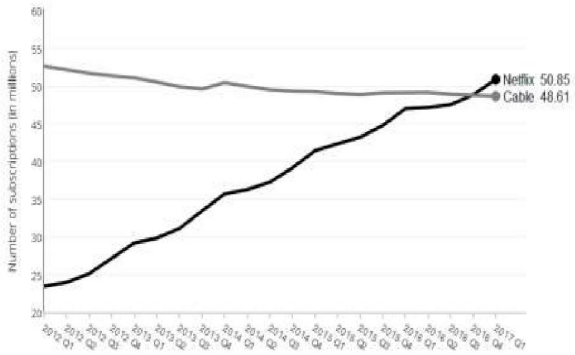

Figure 1: Number of subscriptions in USA

Recommender systems consists of a wide variety of applications ranging from movies, books, products, social network, music etc. Nowadays, almost every other online platform implements this technique for better reviews from users. Moreover, all this data gets collected by these systems to work on them for future recommendations and preferences to user. LinkedIn uses your collected data for future references by providing connections related to user activities.

The above graph depicts the relation between Netflix users and regular cable tv users in United states of America. It shows that the number of Netflix subscriptions outcrossed the normal cable tv subscriptions over the years in United States. It is recorded that Netflix streams over 150 million hours of video every day over the globe and it keeps on increasing with time.

Looking at the facts of Netflix, one can say that they contain huge amount of data regarding user activities and user reviews which can be used further to analyze and increase the total revenue profits and enhancing the customer satisfaction for better user rating. There are numerous advantageous of

implementing a recommender system. Few of those benefits include:

- **Revenue:** Recommender system helps in increasing the revenue of the company to great heights.
- **Personalization:** Recommender system tend to model according to user personal choice. It helps in improving the services provided and design the website according to user choices or preferences.
- **Customer Satisfaction:** Customer reviews are an important factor for any network. If a user is satisfied by the content recommended to him then the feedback would be helpful in retaining many customers.
- **Discovery:** When users are recommended with similar new products based on a different purchase then they tend to visit it again for more surprises.
- **Reports:** Client or administrator can alter their website according to what user will prefer. These reports provided to clients can help them generate new offers on pending products to increase sale profits.

II. Related Work

Recommendation system is currently a trending topic and therefore many advancements have been carried out in its research. Sarwar et al. (2001) proposed and divided the collaborative filtering into two sub-categories:

1) Memory based collaborative filtering

Memory based collaborative algorithms executes by utilizing the user-content completely to obtain a prediction. Based on the data provided, it verifies it with most related users, especially the target. They have common interests therefore can be regarded as neighbors. These neighbors can be calculated using various statistical approaches. In the end, top-n similar data items are concluded for the target user. This filtering procedure is also referred to as user-based collaborative filtering system. The high-end benefits of executing a user-based collaborative filtering algorithm is scalability and sparsity. Usually, many of the recommender system uses data with a greater number of users and item rather than aiming to use the correct and useful data corresponding to actual rating. User-based collaborative filtering algorithm utilizes only the important data content to generate higher efficiency and better results.

2) Model based collaborative filtering

Model-based Collaborative filtering algorithm provides the base structure model for user ratings only. Many supervised and unsupervised machine learning algorithms such as clustering, rule-based and Bayesian network are applied to build this model. All these different techniques execute in its own way. Clustering model treats collaborative filtering to be executed as a classification model while Bayesian model treats it as a probabilistic model. The rule-based model considers it as association-rule mining model. They all formulate different approaches but uses a common methodology of "Item-based collaborative filtering algorithm".

Later, Lops et al. (2011) proposed a technique different from the collaborative approach which utilizes the content of the items and user profile matching to that content. This implementation is referred as Content-based filtering. Lops (2011) states that this type of implementation consists of matching the user profile attributes with the content object. This concludes the degree of user's interest in that object. User profile matched against data object must be accurate otherwise the results would be highly inappropriate.

III. System Design

The project will utilize the Python language as the backend processing tool. We have also used Structure Query Language (SQL) to extract the data that we highly want to emphasize. The model will be deployed for the mobile platform and would recommend the movies as accurately as possible. If possible as a learning, we would try to accommodate a function to punish the model, if the recommendation is not valid. Figure 2 shows the overview of the proposed architecture.

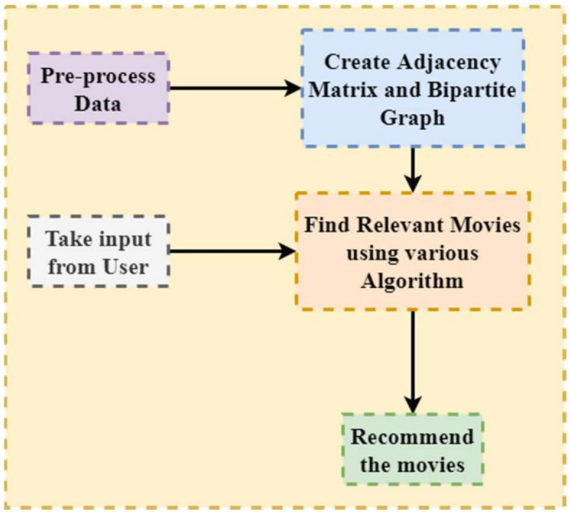

Figure 2: Architecture of Graph Based Recommendation System

IV. Data

The data set for this project is obtained from [Kumar] . The source has a collection of ratings of movies from MovieLens website. This database covers around 100,000 ratings and 1,300 tag for almost 9,000 movies which are given by the 671 users. The data set has mainly two files: movies.csv, ratings.csv along with other files such as links.csv and tags.csv.

Number of users	671
Number of movies	9000
Number of ratings	100,000

Table 1: Overview of the database

It has many attributes which can used appropriately to visualize many different aspects of recommended movies. Some of the attributes of the dataset are:

- userId
- movieId
- tag
- rating
- title
- timestamp

Ratings are made on a 5-star scale, with half-star increments (0.5 stars - 5.0 stars). The dataset needs pre-processing to a great extent as the all its attributes are not required for this implementation. Therefore, data cleaning is a must. We need to merge the csv file to create a unique database.

The one main challenge in this database is that the cold start i.e. when we have no information about the user, or the user doesn't match with the other existing user. While putting this implementation to the work, this model will most likely encounter such instances where a new user with no history. In this case the collaborative filtering is of no use for recommendation. To solve this problem, we must get some feedback from the new user and before we begin to recommend some movies.

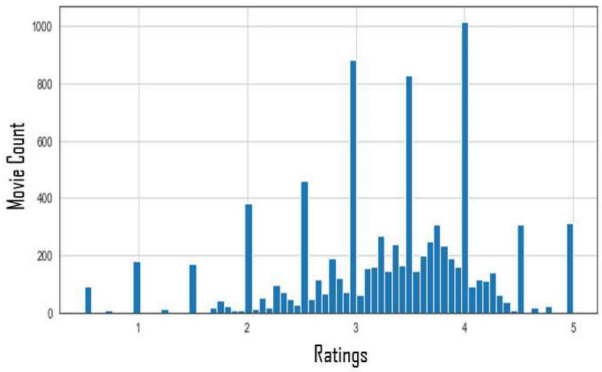

Figure 3: Movie count vs Ratings

Figure 3 explains relation about the number of movies along with the average ratings being provided to each one of them. The movies are grouped based on its count in each rating bar. The graph specifically points out that there are numerous movies that are rated in range of 4 and above. The graph provides some useful insights about the data. The data is not normalized. The bar histogram is biased towards the higher rating count. Recommendation system would be very helpful in these cases to evenly distribute the data all over the range.

Moving on, to evaluate the movie popularity we plotted a graph between the movie count with number of times those movies have been watched by different users. We concluded from the graph that the data acquired was not balanced on another factor which corresponds that very few movies were watched many times and a lot of movies in fact were watched only handful of times. For instance, there are approximately 5000+ movies watched only 10 or fewer times by different users and hardly any movie being watched more than 250 times.

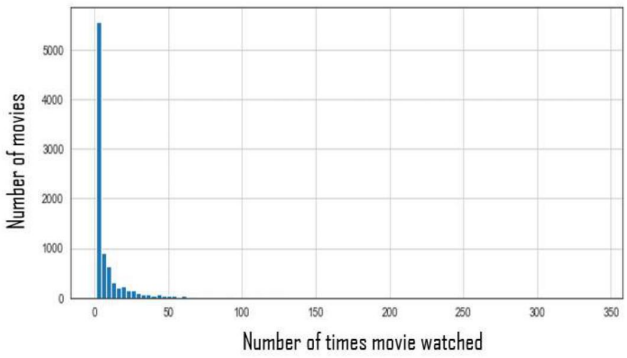

Figure 4: Insights for movie popularity

To facilitate graph traversal techniques and collaborative filtering, the data is transformed into bipartite graph representation. Bipartite graph (or bigraph) is a graph whose vertices can be divided into two disjoint and independent sets U and V such that every edge connects a vertex in U to one in V. Vertex sets U and V are usually called the parts of the graph. In our case, a graph of movie and users, with an edge between a user and a movie if the user has watched that movie. This is a natural example of an *affiliation network*, a type of bipartite graph used in social network analysis.

To combine and illustrate the insights together we formulated the scatterplot with two separate histograms together. The dots represent the number of ratings in each rating group. If there would have been a recommendation system for the above data, the graph would move towards its right and the data will balance out due to equal or similar number of ratings in each group and user's being able to watched different movies based on their liking plus what they might like. This way the movies would be better distributed among the users and genres will do the same.

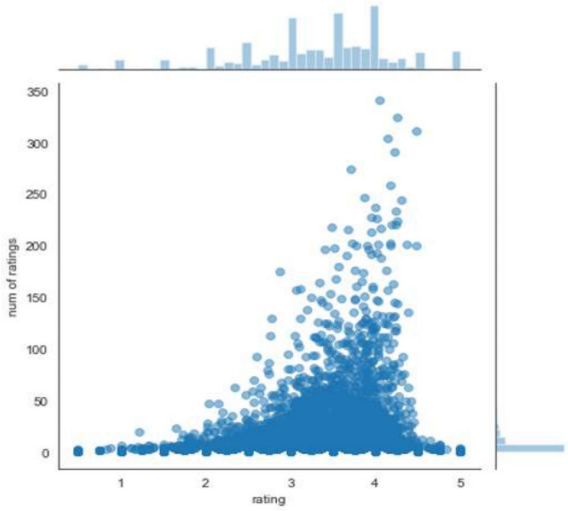

Figure 5: Scatterplot for movie categorization

From the data we have, we first created an adjacency matrix to facilitate the creation of the bipartite graph. In adjacency matrix, we have a square matrix to represent a finite graph. The elements of the matrix indicate 0 if there is no edge between two nodes, and 1 otherwise. We use weights in place of 1 in adjacency matrix to denote the rating the user has assigned to a movie. There are 9123 nodes for movies and 671 nodes for the user. The adjacency matrix size is 9123 x 671. From this representation a bipartite graph is created as shown in Figure 6.

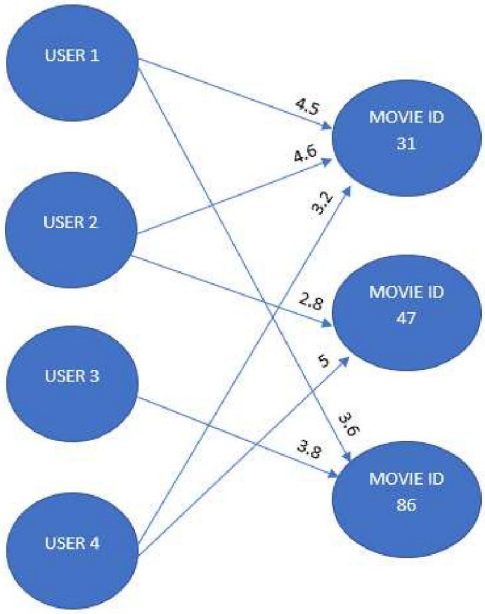

Figure 6: Bipartite Graph

Figure 2 depicts the relation between movies and user for the implemented adjacency matrix where each user may or may not have watched all the movies and each movie has not been rated by every user. For instance, User 1 watched Movie ID 86 and Movie ID 31 only and rated them both based on individual understanding of the genre.

This recommendation system helps in providing suitable movie suggestions to its user based on several different aspects. The recommendations are not based on the result of the modelling but rather on the insights extracted from the data. It will be collaborative process of finding valuable insights about certain aspects of movies or users. Graph based implementation is an addition to the already stated recommendations system being available online. Graph-based algorithm will help us better understand the working of the recommendation system and how data extracted from user activities can be modelled to be able to assist the user again for optimized results.

The data had many outliers and errors associated with it. There were many records with missing values which needed to be corrected before the data pre-processing task. No records were removed from the data set as it will create one sided biasing of the data. Different methods were used to fill up the blank data cells. Average rating was used to manually override the empty values for some movies to create somewhat balanced data set. Genres were misspelled for few movies which were corrected by replacing the correct word.

V. Proposed Approaches

Below are the steps that we will be taking to realize the final recommendation system:

- Data Acquisition
  - Meta-data Extraction
  - Data Insights
- Data Pre-processing
- Implementation of filtering algorithms
  - Content-based filtering
    - Genre Similarity
    - Statistical Analysis
  - Collaborative-based filtering
    - User Similarity
    - Neural Network – Deep Learning
    - SVD
  - Hybrid filtering
- Semantic similarity measure
  - Euclidian Distance
- Graph-based recommendation
- Visualizations

As discussed, we have obtained the data from [Mahata] and arranged the data into the relatable representation. In this project aside from the aforementioned algorithm we are adding

extra facilities. For example, we would like to introduce a list of genres in which we will be listing the movies in the descending order. If the user is more likely inclined to the horror genre, we can suggest the top-rated movie. This method can also be applied in the cold-start of the recommendation system, a series of question can be asked to the user based on the genre, and the top-rated movies can be recommended to the user. However, average rating per movie and genre-wise rating is not available in the dataset. By leveraging MySQL, we overcome this problem. Using SQL, we can pin-point exact data points required for concluding desired results. The data set consist of hundreds of user and movies associated with each other. These movies are rated based on the user's likeliness of the movie. By implementing a threshold value for ratings, we can conclude list of precise movies in each genre that are rated good based on the pre-set threshold value. These movies can be recommended to users specifically looking into the same genre type.

MOVIEID	AVERAGE_RATING
1	3.9
2	3.4
3	3.2
4	2.4
5	3.3
6	3.9
7	3.3
8	3.8
9	3.2
10	3.5
11	3.7
12	2.9
13	3.9
14	3.5
15	2.3
16	3.9

Figure 7: Average movie Ratings

Figure 2 shows us the list of all movies and its associated average ratings provided by the users. The provided ratings are classified out of 5. Furthermore, not all movies are rated equally based on the number of users who rated. Therefore, if a movie has been rated by many users it surely classifies as trending. Genre associated with these movies can be taken into consideration for recommending movies based on similar taste of users. Figure 3 shows the list of movies along with the number of users who rated it.

Figure 3 depicts the number of ratings provided to each movie by users who have watched it. According to the figure, we can conclude that Toy Story is more famous than other listed movies based on the difference in ratings count. Therefore, MySQL has been proven helpful in organizing the data further into meaningful and required chunks out of the original data set which can more efficiently processed.

MOVIEID	TITLE	NUMBER_OF_USER_RATED
1	Toy Story (1995)	247
2	Jumanji (1995)	107
6	Heat (1995)	104
10	GoldenEye (1995)	122
25	Leaving Las Vegas (1995)	101
32	Twelve Monkeys (a.k.a. 12 Monkeys) (1995)	196
34	Babe (1995)	148
36	Dead Man Walking (1995)	104
39	Clueless (1995)	120
47	Seven (a.k.a. Se7en) (1995)	201
50	Usual Suspects, The (1995)	201
110	Braveheart (1995)	228
111	Taxi Driver (1976)	118

Figure 8: Ratings count per movie

A. Content-based Filtering

1. Genre Similarity

In this type of filtering we initially look at the profile of the user. The aim of this contend based filtering is to find the content which will be more relevant to the user. The meta information that is used for doing the approach is the genre. The algorithm for the content-based filtering is given in Algorithm I. As discussed previously, the list per genre is created with the movies listed in the descending order of their ratings. Cosine Similarity is calculated for different genres to extract likeliness between various genres and related movies. The similarity measure helps in providing knowledge for recommending suitable movies to different users. For instance, if user A prefers to watch horror and comedy movies and the system recommends him war genre which is totally irrelevant. Therefore, formulating exact category of movies for specific users should be followed up by calculating the similarity between them and henceforth providing the user with their favorite movies.

2. Statistical Analysis

Additionally, we incorporated another methodology in content-based filtering which utilizes statistical analysis of the movie data inhibiting required genre list to provide certain number of movies for that genre to the user for him to choose based on different genres. The system analyzes the user statistics based on extracted data and utilize it to build different groups of relatable movies for every genre the user prefers. Thereafter, the recommender chooses the best of movies for each group and provide it to the user to watch based on preference. The list for every group is calculated based on the average ratings provided by different users and then recommending certain number of movies from that list.

Algorithm I: Content-based Filtering

1: List all the movies user has watched.
2: List all the distinct genres for each of the movies watched.
3: Calculate the sum of the rating for each genre.
4: Divide each sum by the number of movies for that genre.
5: Pick top genre and list the top 5 movies
6: Recommend this movie to the user.

B. Collaborative filtering

1. User Similarity

The aim of the collaborative filtering is to find the similar uses for the purpose of recommendation. We use cosine similarity to match the network of two users. The cosine similarity would be the highest if both the user has seen same number of movies and have given the same rating. If two users have similarity above the threshold, we can most likely suggest the movies that another user have seen but the first user hasn't. If the other user has seen some movie which first user hasn't seen but has given less rating to the movie, then we can abide from recommending that movie to the user. On the contrary, if the movie has a good rating, then it becomes a potential candidate for the recommendation.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figure 9: Similarity measure

We obtain similarity of the user with respect to all other users. We choose the first few users who are closer to the user for which we are trying to recommend. We haven't decided the threshold for finding the top-most candidate which are like the user for which we want to recommend the movie. The algorithm for the collaborative filtering is shown in Algorithm II.

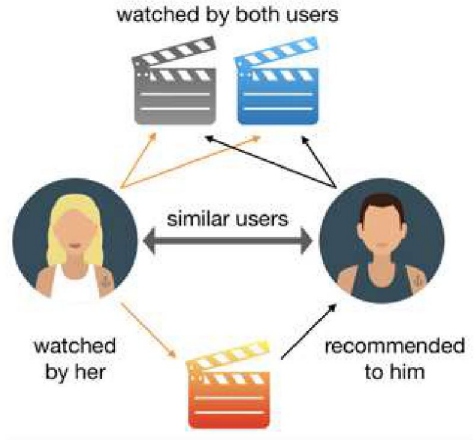

Figure 10: Collaborative learning

The above-mentioned chart describes the working of collaborative algorithm where both the users have watched similar movies which corresponds to user similarity among them. The female user has watched some movie which the other has not watched. Though the two users are similar on calculation of movie choices and taste, we recommend the movie to the male user which the female user has watched.

2. Neural Network – Deep Learning

Neural network has also been implemented in collaborative filtering algorithm. The algorithm helps in predicting values based on some training data provided to it. It assists in recommending movies based on collaborative activity of many users. It is not genre specific neither it is based on user preference. It suggests movies based on a prediction model which might help users to expand their area of interest by watching different genres apart from their taste. For example, the model predicts if movie A is rated good by many users and the probability of getting rated is high then the movie is recommended to that user regardless of the user's favorite genres. This algorithm favors the movies that had not been watched by many users and must be rated for future recommendations. This helps in maintaining balanced recommendations for all users thereby raising the likeliness that the user will watch it.

```
Neural Network userID = 1
Standoff (2016)
Four Weddings and a Funeral (1994)
Louis C.K.: Live at The Comedy Store (2015)
The Danish Girl (2015)
The Boy (2016)
Beasts of No Nation (2015)
Spy (2015)
George Carlin: Jammin' in New York (1992)
Visit, The (2015)
Bad Asses on the Bayou (2015)
The Hateful Eight (2015)
Swimming Upstream (2002)
Pawn Sacrifice (2015)
Zenon: The Zequel (2001)
Ashes of Time (Dung che sai duk) (1994)
Elsa & Fred (2014)
Daddy's Home (2015)
```

Figure 11: Recommendation for user 1 via Deep learning

Neural network works on basic algorithm of deep learning where different values are predicted by combining content and collaborative approach together. In deep learning of collaborative algorithm, the training of matrix factorization runs simultaneously along with auto-encoder incorporating different item characteristics.

Algorithm 5: Collaborative Deep Learning

initialize V, U;
preprocess content of items;
for i **to** N **do**
  foreach $(X_{batch}, V_{batch}) \in batches$ **do**
    $trainAutoEncoder(X_{batch}, V_{batch})$;
  end
  $theta \leftarrow AutoEncoder(X)$;
  $UtU \leftarrow U^T U$;
  foreach $i \in items$ **do**
    $UtU \leftarrow \text{UtU} + \alpha U_{obs}^T U_{obs} + \lambda_v I$;
    $Utr \leftarrow (1+\alpha) U_{obs} r_{obs} + \lambda_v theta_i$;
    $V_i \leftarrow solve(UtU, Utr_{obs})$;
  end
  $VtV \leftarrow V^T V$;
  foreach $u \in users$ **do**
    $VtV \leftarrow VtV + \alpha V_{obs}^T V_{obs} + \lambda_u I$;
    $Vtr \leftarrow (1+\alpha) V_{obs} r_{obs}$;
    $U_u \leftarrow solve(VtV, Vtr_{obs})$;
  end
end

Figure 12: Deep NN

Deep neural network works on the above-mentioned logic and predict values based on history action of the user or item attributes.

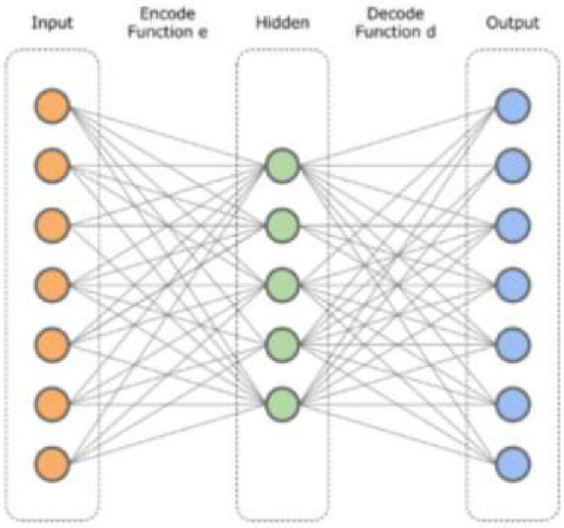

Figure 13: Neural network layers

There are several layers of execution in the neural network statistics. These circles in each layer are usually referred to as neurons as in human brains because neural network learns from given data to predict new values so does the brain for human. The neurons are represented in layers namely input layer, hidden layer and output layer. The input layer receives the data to be processed by the deep learning network. In this case, movie list, ratings, user count are the input values to the first layer of the network. The hidden layer moves on to calculate and perform algorithms on the input data. Now, there are more than one hidden layer because the name itself suggest "hidden" meaning more than just one. The count of neurons per layer and the number of layers needs to be defined beforehand. The output layer gives out the predicted values as per the given historical data (given data) to the neural network. Recommendations are the output provided by the system.

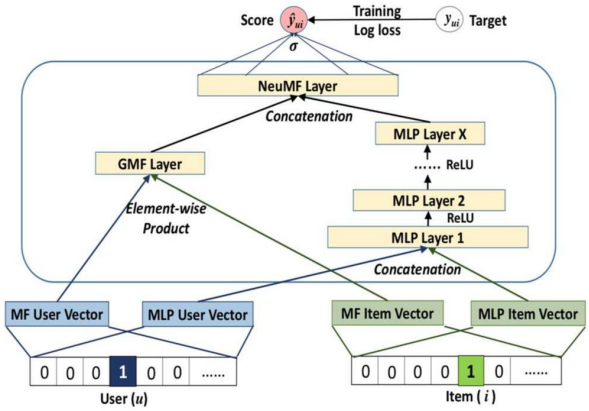

Figure 14: Internal NN working

3. Single Value Decomposition

Another filtering algorithm SVD (Single value decomposition) is implemented to analyze results by comparing with other algorithms and pointing out the best resulted recommendations. Single value decomposition also performed considerably well and resulted positive.

Algorithm II: Collaborative Filtering

1: Compute the cosine similarity among each user.
2: Filter out the similar users.
3: Find the movies which the targeted user hasn't watch but the similar users have seen.
4: Sort the movies in the descending order of their ratings.
5: Pick the top 5 movies
6: Recommend this movie to the user.

C. Hybrid Filtering – Modelling flowchart

Hybrid filtering system is a combination of various other algorithms stacked together to provide precise recommendations to user in different aspects. Content based filtering tend to provide with list of movies in each genre liked by the user and collaborative filtering stimulates the list based on other users' choices and likeliness. Each one does only its part of recommending movies. Content based cannot give you movie suggestions based on other users' ratings of movies with similar interest. Collaborative gives you similar recommendations based on user but not content (genre) of the movies liked. To expand the suggestion list to provide numerous varieties of movies in every sense to the user, hybrid

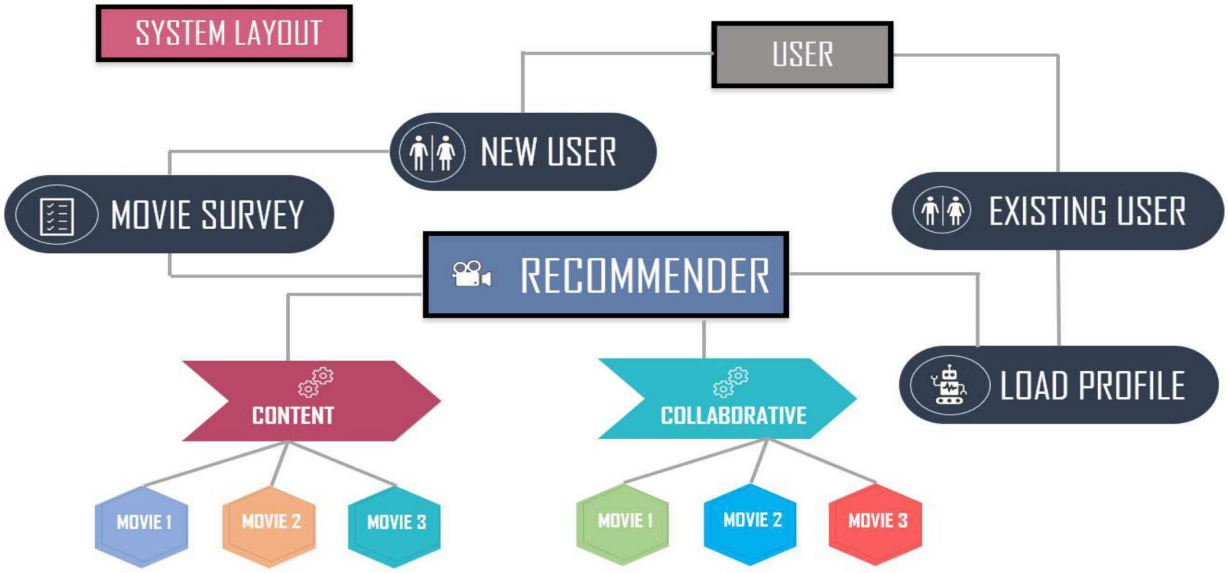

Figure 15: Recommendation system layout

collaboration is required. It combines the results of all different filtering algorithm and shoots list of movies in each category. It will provide you similar genre movies, trending movies based on another user's interest which may be suitable for you. In addition, it will also provide top movies with preset threshold in every genre the user prefers. For instance, User 4 likes to watch horror, comedy and action movies. The recommender will provide the user with similar movies in genre category, similar interesting movies in trend and supposedly top-10 movies in each of the three genres the user likes.

VI. Experimental Results

We have implemented the above model using the Python. The input to the model is the list of the movies the user has watched followed by the ratings for each movie. We have printed the movie recommendation using two algorithms. We list the top 3 movies for each category. As of now the program prints the movieId which can then be mapped to the names of the movie.

Figure 16: Division of data for modelling

The data gets separated into training and testing set for building the model more precise. Neural network executes based on a good training set to provide useful predictions to the testing set. Figure 16 represents a partial output of the recommendation system.

```
# Key-value pair for each movie
# MovieID : Rating
Vector = {324:3.5,
          329: 4,
          504: 1,
          430: 2.5,
          660: 3.0,
          10: 2.0,
          123: 4,
         }
# Pass the vector to the Recommendation System
Recommend(Vector)
```

```
Using Content-based Algorithm
Top Genre is: Comedy
Movies recommended using Content-based Algorithm: 34, 628, 2
Using Collborative based Algorithm
Similar Users for threshold above 0.8 are: '342','1206','327'
Recommended Movies are: 421, 34, 730
```

Figure 17: Key-value movie pair

The results were concluded based on several aspects of different algorithms used. RMSE values were calculated and compared for different algorithm and the best one is chosen among all. The resulting RMSE concluded was best for deep learning neural network which was calculated to be around 0.8. The lower the RMSE value, the better the model. Neural network modelling was found out to be more precise to the resulting recommendations when compared to other algorithms. Therefore, Deep neural network can state to be the best among all.

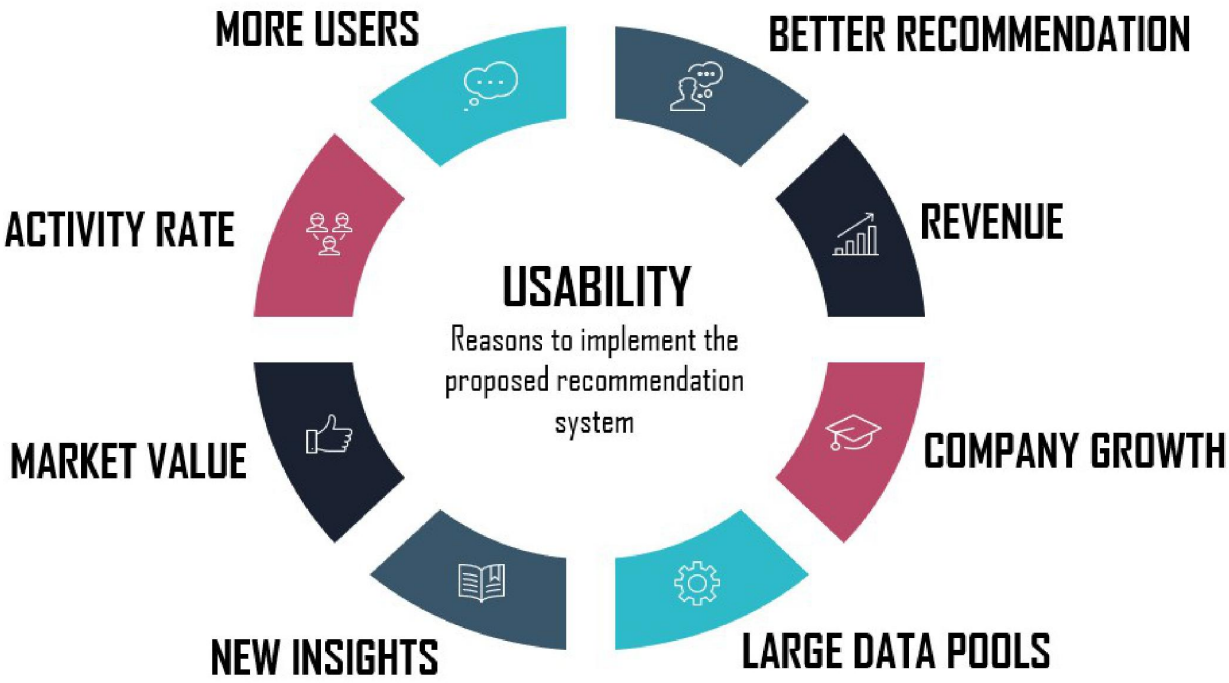

Figure 18: Application of Recommendation System

Though, the whole procedure of calculating RMSE values for different algorithms was just to conclude the best model rather than resulting recommendation. The different algorithms cannot be compared with one another as they all have separate output values. Content-based filtering cannot be compared to collaborative as it is more focused on providing recommendations based on user's genre interest and collaborative approach is more bent towards suggesting movies based on other user's activities and choices with similar interest. They both recommend different movies based on there algorithm. Therefore, both can be referred to work independently and provide accurate results.

$$RMSE(model) = \sqrt{\frac{1}{|R_{test}|} \sum_{(u,i,r) \in R_{test}} (model(u,i) - r)^2}$$

Figure 19: RMSE Calculation

Neural network performed the best based on RMSE value (0.8). Neural network helps in recommending movies that may be of interest to the user since it got rated high by other users. Predicting these values and then recommending movies is a plus point in the system which makes the recommender more robust and adaptive.

VII. Discussions

While running the above model, if the length of the vector is small, the algorithm faces difficulty in recommending the movie. We have observed that by providing at least 7 movies, we can get as much as 3 similar users in the dataset. The number of similar users also depends on the cosine similarity threshold that we have set. By experimenting this threshold can be adjusted. We have found that from figure 9, a greater number of users can be found when the cosine similarity threshold is 0.5. Further we can leverage the Euclidean distance for finding the similarity between many users. Apart from this, clustering methods such as K-means can be used to form the clusters of the similar users.

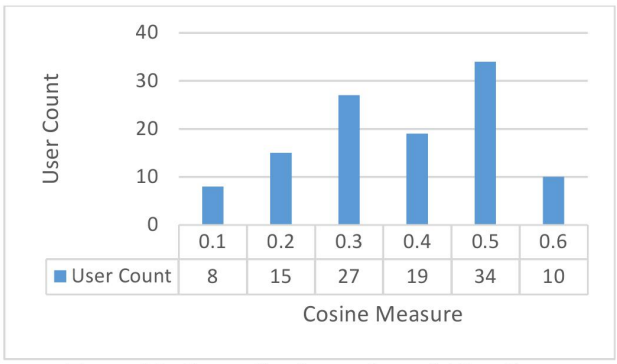

Figure 20: Cosine Similarity plot with User Count

The above figure plots the evaluated cosine values to its user count measure. For instance, for value 0.1, there are 8 similar users associated with it.

VIII. Conclusions

After performing all the required preliminary implementations on the given data set, we concluded that Comedy and Adventure constitutes the maximum number of users rating so far. "Toy Story" residing at the top of the table for most reviewed movie with over two hundred ratings from users. Content based filtering provided us with the best suited genre for user as Comedy. Collaborative filtering algorithm concluded similar interest users together to recommend them movies based on there likeliness. Few movies such as Babe (1995), Black beauty (1994) and many more were fount out to be appropriate for recommending to users.

Furthermore, the implementation of Graph based recommendation system has been queued into line of implementation and will be later concluded with results for the same. There were many insights based on the results obtained like Comedy genre was popular among several users and average rating was also considerably high for it. These insights can lead us to diverse knowledge about user rating mechanism and user liking, Users prefer comedy movies over action movies for some reason. These insights can help us better recommend movies since comedy movies would get better reviews compared to any other genre therefore, certain users will be recommended comedy genre rather action or any other. Graph based implementation will also give us some detailed and valuable insights to investigate which can further help us better understand the recommendation system evaluation.

A good recommendation system can be very helpful in boosting an organization in many ways. The application of recommendation system can raise many standards for a company for future aspects. The figure above gives insights about the typical advantages of having a precise movie suggestion system for the users.

Precision on Top-N: $Precision(u) = \frac{|Recommended(u) \cap Testing(u)|}{|Recommended(u)|}$

Recall on Top-N: $Recall(u) = \frac{|Recommended(u) \cap Testing(u)|}{|Testing(u)|}$

Figure 21: Precision & Recall Calculation

Precision and recall are two extremely important model evaluation metrics. While precision refers to the percentage of your results which are relevant, recall refers to the percentage of total relevant results correctly classified by your algorithm. More practical offline evaluation measure is recall or precision evaluating percentage of correctly recommended items (out of recommended or relevant items). Suppose that we computed recall at 10 and found it is 40% in our top-10 recommendation system. This means that 40% of the total number of the relevant items appear in the top results. Using this extra two evaluation methods, we can further, optimize the recommendation system.

A popular algorithm for recommending movies is the Apriori algorithm. This algorithm defines the way in which we mine data and how we evaluate its usefulness. However, this algorithm was inefficient and un-scalable to the amount of data we had in our database. Hence, we conclude that, the Apriori can't be used as a viable means of recommendation and would require additional pre-processing steps.

In this work, we have successfully, implemented the content and collaborative based algorithm for recommending the movies. We have found that, the Neural Network based recommendation system had the best RMSE. We further proposed the hybrid recommendation system which is the product of both content and collaborative based recommendation system.

References

[1] Techlabs, M., & Techlabs, M. (2017, October 05). 5 Advantages Recommendation Engines can Offer to Businesses. Retrieved from https://towardsdatascience.com/5-advantages-recommendation-engines-can-offer-to-businesses-10b663977673

[2] (n.d.). Retrieved from https://www.kaggle.com/ibtesama/getting-started-with-a-movie-recommendation-system/data

[3] Analysis of Movie Recommender System using Collaborative Filtering. (2017). International Journal of Recent Trends in Engineering and Research,3(5), 338-346. oi:10.23883/ijrter.2017.3232.zpbxj

[4] Kumar, M., Yadav, D., Singh, A., & Kr., V. (2015). A Movie Recommender System: MOVREC. International Journal of Computer Applications,124(3), 7-11. doi:10.5120/ijca2015904111

[5] Leka O. (2016, November 15). IMDB Movies Dataset. Retrieved from https://www.kaggle.com/orgesleka/imdbmovies/home

[6] Ariff, N. M., Bakar, M. A., & Rahim, N. F. (2018). Comparison between content-based and collaborative filtering recommendation system for movie suggestions. doi:10.1063/1.5054256

[7] Mahata, A., Saini, N., Saharawat, S., & Tiwari, R. (2017). Intelligent Movie Recommender System Using Machine Learning. Intelligent Human Computer Interaction Lecture Notes in Computer Science,94-110. doi:10.1007/978-3-319-52503-7_8

[8] Ojokoh, B., Aboluje, O. O., & Igbe, T. (2018). A Collaborative Content-Based Movie Recommender System. International Journal of Business Intelligence and Data Mining,1(1), 1. doi:10.1504/ijbidm.2018.10012014

Appendix

The result from the python script are demonstrated below. It shows the Movie name along with the ratings.

```
#################################### Similar Movies ####################################
One Flew Over the Cuckoo's Nest (1975) 4.25
Exotica (1994) 4.26
Cool Hand Luke (1967) 4.27
Cure, The (1995) 4.25
Hearts and Minds (1996) 4.25
Ponette (1996) 4.25
Men with Guns (1997) 4.25
Love Is the Devil (1998) 4.25
Room at the Top (1959) 4.25
My Life (1993) 4.25
Running on Empty (1988) 4.25
Lenny (1974) 4.25
Believer, The (2001) 4.25
Requiem for a Heavyweight (1962) 4.25
Ten (2002) 4.25
Salaam Bombay! (1988) 4.25
Tokyo Story (Tôkyô monogatari) (1953) 4.25
Vivre sa vie: Film en douze tableaux (My Life to Live) (1962) 4.25
Macbeth (a.k.a. Tragedy of Macbeth, The) (1971) 4.25
Best of Youth, The (La meglio gioventù) (2003) 4.25
Long Day's Journey Into Night (1962) 4.25
Coach Carter (2005) 4.25
Julia (1977) 4.25
Cria Cuervos (1976) 4.25
Kite Runner, The (2007) 4.25
Fish Tank (2009) 4.25
Certified Copy (Copie conforme) (2010) 4.25
Hunt, The (Jagten) (2012) 4.25
Short Term 12 (2013) 4.25
Fantastic Mr. Fox (2009) 4.02
The Lego Movie (2014) 4.07
Boxtrolls, The (2014) 4.5
The Good Dinosaur (2015) 4.5


#################################### Trending Movies ####################################
Métisse (Café au Lait) (1993) 4.0
Ballad of Narayama, The (Narayama Bushiko) (1958) 4.0
General, The (1998) 3.6
Blues Brothers, The (1980) 3.89
Swingers (1996) 4.22
Shanghai Triad (Yao a yao yao dao waipo qiao) (1995) 4.05
Four Weddings and a Funeral (1994) 3.68
Louis C.K.: Live at The Comedy Store (2015) 4.0
The Danish Girl (2015) 4.0
Beasts of No Nation (2015) 4.0
Spy (2015) 3.7
George Carlin: Jammin' in New York (1992) 5.0
The Hateful Eight (2015) 4.06
Zenon: The Zequel (2001) 4.0
Ashes of Time (Dung che sai duk) (1994) 4.0
```

Timeline:
02/25 : Project Proposal (Done)
03/08 – 03/15 : Data Preprocessing (Done)
03/22 – 04/01 : Building the Model (Done)
04/01 – 05/06 : Presentation, Optimizing Model, Demo, Report Writing. (Done)

ASSIGNMENT

Sameer Kumar: Team Lead, Analyst, Developer
Project Task Progress: MySQL, Data Pre-processing, Cosine Similarity

Gaurav Kolhe: Developer, Documentation
Project Task Progress: Python, Content based filtering, Collaborative based filtering

The work will be mostly splitted such that each of the member gets equal share of participation in the project.