

MOVIE RECOMMENDATION SYSTEM USING INFORMATION FILTERING

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

PRADHUMAN SINGH PARMAR (37110584)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

March - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC | 12B Status by UGC | Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of

Mr. PRADHUMAN SINGH PARMAR (37110584) who carried out the project entitled **“MOVIE RECOMMENDATION SYSTEM USING INFORMATION FILTERING”** under my supervision from November 2020 to March 2021.

Internal Guide

DR. S. MURUGAN, M.E., [Ph.D.,]

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **PRADHUMAN SINGH PARMAR** (Reg. No. 37110584) hereby declare that the Project Report entitled "**MOVIE RECOMMENDATION SYSTEM USING INFORMATION FILTERING**" done by me under the guidance of **DR. S. MURUGAN, M.E., [Ph.D.]** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D.**, Dean, School of Computing and **Dr.L.Lakshmanan M.E., Ph.D.**, and **Dr.S.Vigneshwari M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. MURUGAN, M.E., [Ph.D.,]** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Recommendation systems are playing a vital part in our lives, from browsing music videos on YouTube to browsing Journals and Books on Goodreads a recommender algorithm helps in filtering the content based on the browsing performed by the customer and helps by encouraging a browsing customer to become a buying customer by giving personalized recommendations. Goodreads uses it to recommend better Journals and books to its readers. YouTube utilizes it adds videos in your playing queue. Spotify heavily depends on the recommendation algorithm to determine which song to play next on auto play. Hence if a recommendation system isn't precise then it could affect the business performance of the e-commerce platform. There are different types of Information/Data Filtering methods which could be applied to get recommendation results namely in this project we're using Demographic Filtering, Content Based Filtering and Collaborative Filtering. Recommendation systems help users browse movies or content that they may not come across or know about. Conventionally Collaborative Filtering is used to let customers come across new recommendations. A blend of both Content and Item based Filtering has been applied on this project to serve a more appropriate movie recommendation engine.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	vi
1.	INTRODUCTION	01
	1.1. RESEARCH AND SIGNIFICANCE	02
	1.2. OUTLINE OF THE PROJECT	02
	1.3. PROBLEM STATEMENT	03
2.	AIM AND SCOPE	04
	2.1. AIM OF THE PROJECT	04
	2.2. SCOPE OF THE PROJECT	04
	2.3. ADVANTAGES	05
	2.4. RECOMMEDATION TYPES	06
	2.5. DISADVANTAGES OF COLLABORATIVE FILTERING	09
3.	SYSTEM DESIGN AND METHEDELOGY	10
	3.1. SYSTEM IMPLEMENTATION	10
	3.2. SYSTEM & SOFTWARE REQUIREMENTS	11
	3.3. METHEDOLOGY	11
	3.3.1. COLLABORATIVE FILTERING PROCESS	11
	3.3.2. USER RATING SCORE DATA INPUT	12
	3.4. ESTIMATING SIMILARITY INDEX	13
	3.4.1. COSINE VECTOR SIMILARITY	13
	3.4.2. PEARSON CORRELATION COEFFICIENT	14
	3.4.3. EUCLIDEAN DISTANCE SIMILARITY	15
4.	SYSTEM DEVELOPMENT METHEDOLOGY	16
	4.1. ARCHITECTURAL DIAGRAM	16
	4.2. DESCRIPTION OF DIAGRAM	17

4.3. COLLABORATIVE FILTERING	18
4.3.1. ITEM BASED FILTERING	18
4.3.2. USER BASED FILTERING	19
5. RESULTS AND DISCUSSION	20
5.1. RESULTS	20
5.2. OUTPUTS & SCREENSHOTS	20
6. CONCLUSION & FUTURE WORKS	24
6.1. CONCLUSION	24
6.2. FUTURE WORKS	24
REFERENCES	26
APPENDIX	28
A) SOURCE CODE	28

LIST OF FIGURES

FIG.NO	FIG NAME	PAGE NO
2.1	TYPES OF RECOMMENDATION	08
3.1	SYSTEM IMPLEMENTATION	10
3.2	FILTERING PROCESS	11
3.3	ITEM-USER MATRIX	12
3.4	COSINE SIMILARITY	13
3.5	PEARSON CORRELATION	14
4.1	AFTER PRE-PROCESSING	16
4.2	FILTERING ALGORITHM	16
4.3	REDUCING DIMENSIONS	17
4.4	ITEM BASED COLLABORATIVE FILTERING	18
4.5	USER BASED COLLABORATIVE FILTERING	19
5.1	GENRE WISE BAR GRAPH	21
5.2	HOME SCREEN	21
5.3	CHOOSING A FILTERING TECHNIQUE	22
5.4	FEEDING MODEL WITH AN INPUT	22
5.5	GETTING RESULTS WITH LINKS TO CORRESPONDING MOVIES	23
5.6	USING FILTERS FOR AN ADVANCED SEARCH	23

CHAPTER 1

INTRODUCTION

Recommendation system are an essential aspect of the modern-day society. With the likes of Artificial Intelligence and Machine Learning growing rapidly in the world, Huge multimedia companies heavily depend upon recommendation system to improvise the customer's user experience. Recommendation system analyses on user preferences to generate similar products that the user might have not discovered yet. Recommendation is often used to visualize inputs in a more efficient way, with normally perceived models appearing as playlist generators for video and music administrations, item recommenders present for the online stores which sell out their product in an online way, or substance recommenders for social media stages. Recommender systems majorly depend upon two techniques to bring out customized recommendations by making use of either or both collaborative filtering and content-based filtering (otherwise called the personality-based methodology), just as different frameworks, for example, information-based frameworks. The way that a Collaborative filtering technique basically work is by taking in consideration a client's previous history of search/purchase (by constructing a similarity matrix we can know the items which are similar to them) and utilizing it to give similar recommendations, suppose if a **Person A** buys a **Shoe** what a recommender system does is it tries to give the customer recommendation according to their history of purchase which in this case would be matching shoelaces, pair of sock etc. however in our case we are trying to find similar movies and not similar products so the parameters needs to be judged accordingly.

Recommendation's systems help companies understand their audience on a larger scale as their model can focus on each user and find an appropriate list of things that their used would most possibly love and all this which wasn't possible 20 years ago makes it must in any Business Model. RS are a value for money, time and investment for digging into chunks of data for something valuable that they wouldn't have known about their users otherwise.

Systems like these have a very greater importance as they try to understand the behavioural pattern of the user by using the data that it already has of other users. The logic hides behind

trying to first find similar users and only after that comes finding similar movies to reach to our end goal. These kinds of systems are now used everywhere from a casual online book store to a fancy restaurant everyone is trying to get personalized and precise on their recommendations.

1.1 RESEARCH AND SIGNIFICANCE

Data over-burden is a major test for online clients. It is especially an issue when a client needs to rapidly and precisely recognize an asset from a dramatically developing arrangement of data and items. It is likewise hard for traders to introduce the proper items to clients in an ideal way. The rise of the recommendation framework has fairly soothed this test.

A recommendation framework is a kind of data channel, which can gain proficiency with clients' inclinations and pastimes as indicated by their profile or chronicled practices, and afterward foresee their evaluations or inclinations for a given thing. It changes the manner in which organizations speak with clients and fortifies the intelligence between them.

The measurements from concentrates by McKinsey and Tech Emergence bear this out: this sort of proposal framework brought Amazon 35% of its income and 23.7 percent development to Best Buy. Up to 75 percent of video utilization on Netflix comes from the proposal framework and 60 percent of perspectives on YouTube come from their personalized suggestion framework.

Hence, how to fabricate a powerful recommendation framework is of significant importance. Over the course of time there have been different methodologies which have been found affective and precise in terms of recommending things to the users with a greater accuracy.

1.2 OUTLINE OF THIS PROJECT

The main goal of the project is to achieve a robust recommendation system with both item based and user based coming under one system. Also allowing the user to get movie recommendations according to genres (content-based filtering). The system sets off by

taking an input movie from the user and then recommends them a list of movies according to their selected choice of filtering technique as there are couple of options which they can choose from.

For achieving this we're going to take use of python libraries like pandas, matplotlib, numpy arrays etc. the entire system will be deployed using streamlit, another python library for data visualization and laying out the data.

1.3 PROBLEM STATEMENT

The movie recommendation system will be built using artificial algorithms that analyze user's favorite genres and recommend movies according to their liking. The response will be based on the liking of the user. The User will submit queries depending on their liking of their movies. The System analyses the liking and then recommends the user movies. Providing related content out of relevant and irrelevant collection of items to users of online service providers. Our aim is to recommend movies to users based on content of items as well as based upon user's opinions.

CHAPTER 2

AIM AND SCOPE

2.1 AIM OF THE PROJECT

The objective of recommender systems is to provide recommendations based on recorded information on the users' preferences. These systems use information filtering techniques to process information and provide the user with potentially more relevant items. The major aim here is to establish and understand how these things work, the methodology followed in accomplishing a collaborative filtering system, our aim is to present collaborative filtering techniques based on users and items along with genre-based selection on movies (content-based filtering).

2.2 SCOPE OF THE PROJECT

Three main approaches using which the data is being assessed are used for our recommender systems. One is content-based filtering i.e., Content-based filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback. To demonstrate content-based filtering, let's hand-engineer some features for the Google Play store. The following figure shows a feature matrix where each row represents an app and each column represents a feature. Features could include categories (such as Education, Casual, Health), the publisher of the app, and many others. To simplify, assume this feature matrix is binary: a non-zero value means the app has that feature.

You also represent the user in the same feature space. Some of the user-related features could be explicitly provided by the user. For example, a user selects "Entertainment apps" in their profile. Other features can be implicit, based on the apps they have previously installed. For example, the user installed another app published by Science R Us. The other is collaborative filtering, where we try to group similar users together and use information about the group to make recommendations to the user. Movies Recommendation system, there are two ways in which this practice can be carried out one being user centric and the

other technique being Item centric both have different conditions points to construct the recommendation outcomes.

2.3 ADVANTAGES

Benefits of a recommender engine can be seen below:

1. Personalized Results

Through personalized email messages and targeted blasts, a recommendation engine can encourage elevated amounts of traffic to your site, thus increasing the opportunity to scoop up more data to further enrich a customer profile.

2. Deliver Relevant Content

By analysing the customer's current site usage and previous browsing history, a recommendation engine can deliver relevant product recommendations as he or she shops based on said profile. The data is collected in real time so the software can react as shopping habits change on the fly.

3. Engage Users

Users become more engaged when personalized product recommendations are made to them across the customer journey. Through individualized product recs, customers are able to delve more deeply into your product line without having to dive into.

4. Convert Users to Customers

Converting users into customers takes a special touch. Personalized interactions from a recommendation engine show your customer that he or she is valued as an individual, in turn, engendering long-term loyalty.

5. Increase Average Order Value

Average order values typically go up when an engine is leveraged to display personalized options as users are more willing to spend generously on items they thoroughly covet.

6. Increase Number of Items per Order

In addition to the average order value rising, the number of items per order also typically rises when an engine is employed. When the customer is shown options that meet his or her interest, they are far more likely to add items to their active purchase cart.

7. Control Merchandising and Inventory Rules

A recommendation engine can add your marketing and inventory control directives to a customer's profile to feature products that are on clearance or overstocked so as to avoid unnecessary shopping friction and tone deafness.

8. Reduce Workload and Overhead

The volume of data required to create a personal shopping experience for each customer is usually far too large to be managed manually. Using an engine automates this process, easing the workload for your IT staff.

9. A Recommendation Engine Provides Reports

Detailed reports are an integral part of a personalization system. Accurate and up-to-the-minute reporting will allow you to make informed decisions about the direction of a campaign or the structure of a product page.

10. Offer Advice and Direction

An experienced recommendation provider like Google can offer advice on how to use the data collected from your recommendation engine. Acting as a partner and a consultant, the provider should have the industry know-how needed to help guide you and your ecommerce site to a prosperous future.

2.4 Recommendation Types

In the following section we describe the five broad types of recommenders ordered roughly from most simple to most complex.

1. Most Popular Items: The simplest strategy is to simply offer the customer whatever is most popular, be that a movie, a book, or an article of clothing. Without doing anything more than looking in your sales records you could accomplish this. No data science required. It's not particularly personalized but could be useful if you know very little about your visitor. It does require some basic content attributes to create subcategories that can match your visitor's current browsing. For example, if you are offering a wide variety of merchandise like everything from tools to clothing, or movies, books, or news that appeal to different interests then you'll need to try to match items at least in the same category just viewed by your visitor. Despite these limitations Home Depot regularly uses 'best sellers' and the GAP regularly uses 'latest products' and 'arriving soon' to increase revenue. Most likely you will consider 'most popular' recommenders as a supplementary strategy.
2. Collaborative Filtering: Collaborative filtering **focuses on the user** and other users found to be mathematically similar to the user. In theory **no specific attributes are required for the content** which CB can infer. Later we will see that adding content attributes can enhance performance but is not technically required. The underlying premise is that if two users have a strong similarity of likes and dislikes in the past that they will continue to have strong similarity in the future. CB will match people who like romance films to those films that have strong romantic content without the requirement for defining 'romance'. Once the similarity is established then items consumed by one user can be recommended to other similar users. The existence of a post-selection rating, an immediate like/dislike indication, and/or a pre-existing user profile is necessary to make this technique viable. CB attempts to predict the user rating for an unseen item. Accuracy of the prediction can be determined by comparing the predicted rating to the rating actually given when the recommended item is consumed.
3. Hybrid Models: There are two interpretations of Hybrid recommenders and you should think of them as two sides to the same coin.

- a. Brute Force or Knowledge Based
- b. Combined CF/CB Systems

Brute Force or Knowledge Based

This variation is quite easy to understand since it involves the addition of rules by human subject matter experts. Good Product Marketing Managers can frequently define what products do and do not go together, and which may be complementary versus supplementary.

Combined CF/CB Systems

As you read about CF and CB recommenders it may have already occurred to you that it would be beneficial to have the benefit of both. Note that some versions of CB systems already use both customer and item attributes although the item attributes in this case are typically synthetic calculated variables that may not have any easily understood logic associated. In this case the combined CF/CB systems are different based on the many ways these two system types can be combined.

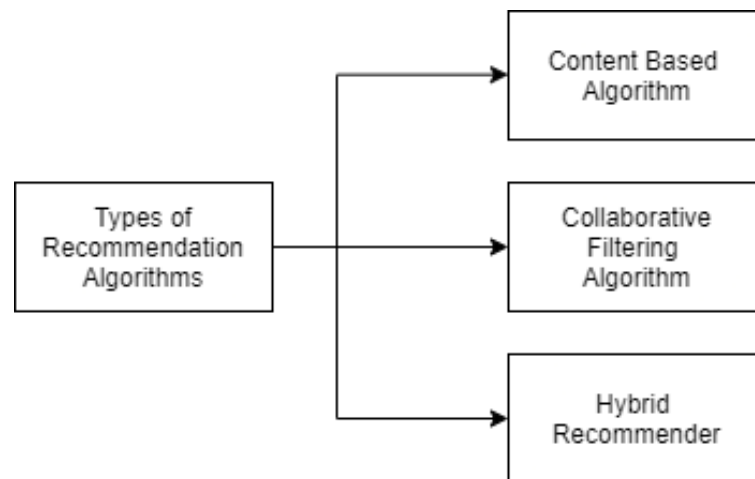


Figure 2.1 Types of Recommendations

2.5 DIADVANTAGES OF COLLABORATIVE FILTERING

Since the number of users and items in each application has steadily increased at the same time as the growth of World Wide Web, collected input data has been a big problem in producing an accurate prediction and in running 20 recommendation system using collaborative filtering. There are two main challenges in user-based collaborative filtering [8, 9, 10, 15].

2.5.1 Data Sparsity

User-based collaborative filtering depends on explicit feedback, such as ratings given by user to item. User-item input data matrix could have a few rating scores of the total number of items available, even though users are very active. In addition, because users tend not to rate actively, calculating similarity over curated set of items could be a challenge. These problems give rise to inaccurate performance of the recommendation system. Even the cold-start problem is caused by the data sparsity. Collaborative Filtering predicts items based on user's previous preference behavior. That is, it could not predict recommendable items to new users unless new users rate many items. Also, new items could be considered for recommendation, because they have less rating scores by a sufficient number of users.

2.5.2 Data Scalability

For over millions of users and millions of items in user-item input data matrix, the nearest neighbor algorithm is required for high scalability of computation between users as the values of two vectors. Also, recommendation systems could not quickly react to online requirements and immediately make recommendations as it was a time-consuming job.

CHAPTER 3

SYSTEM DESIGN AND METHEDOLOGY

3.1 SYSTEM IMPLEMENTATION

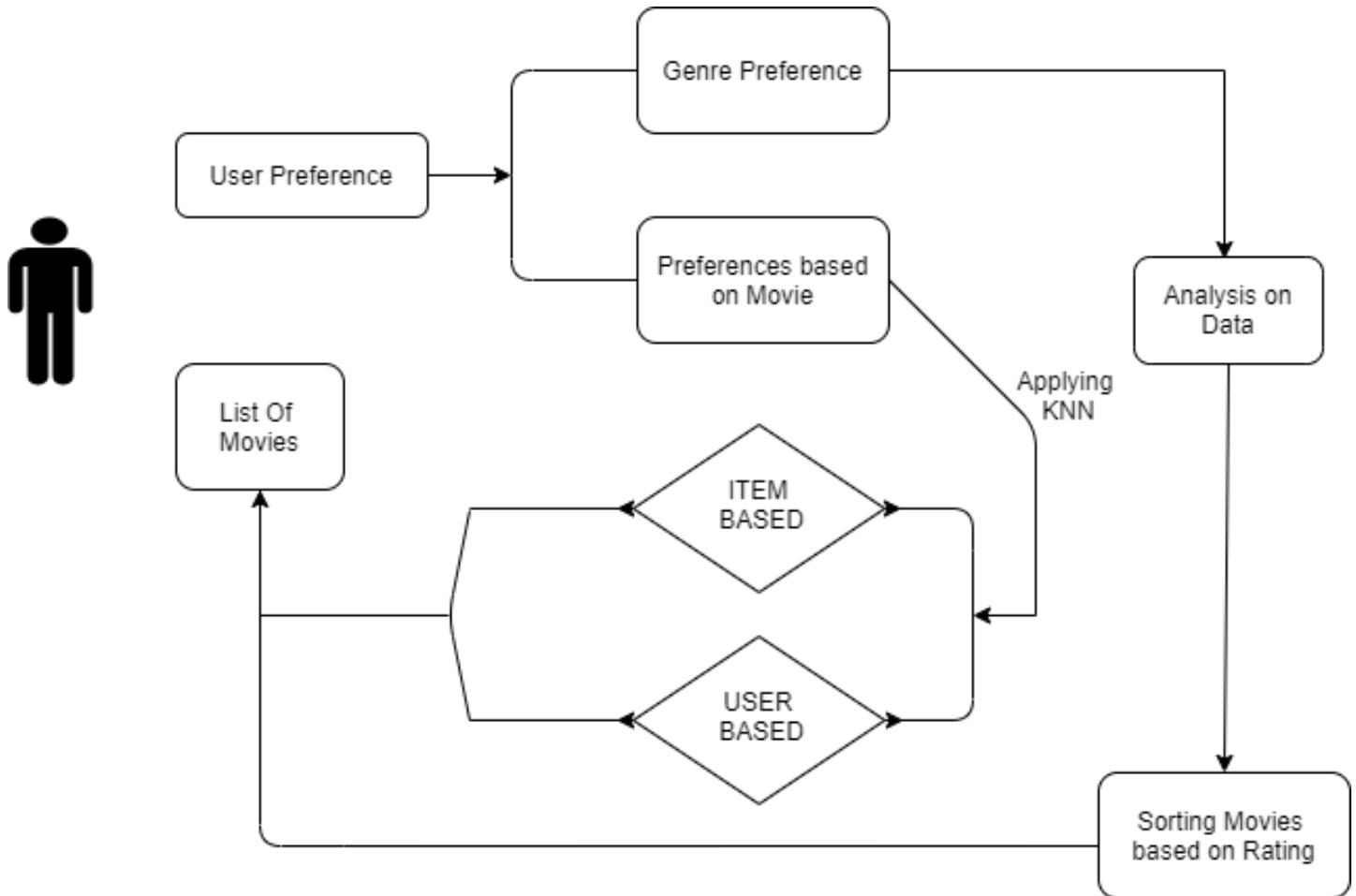


Figure 3.1- System Implementation

The above diagram represents the flow of the system from the point the input is being taken from the user to the point passing it back to the user in the form of recommendations. Upon receiving the input from the user, it moves on. Depending upon the kind of filtering technique the entry passes on to the analysis phase or to the algorithm implementation implementation

3.2 SYSTEM & SOFTWARE REQUIREMENTS

- Processor: 2.6 GHz Intel Core i5
- Memory: 8 GB 1600 MHz DDR3
- Operation System: OS X El Capitan Version 10.11.1
- Language: python
- Synder , Anaconda or VScode
- Pandas, matplotlib, streamlit
- Sklearn, Plotly, seaborn
- Jupyter Notebooks

3.3 METHEDOLOGY

3.3.1 Collaborative Filtering Process

In a fundamental scenario, collaborative filtering (CF) processing can be mainly divided into three steps; Step 1) collecting user ratings data matrix, Step 2) selecting similar neighbors by measuring the rating similarity, and then Step 3) generating prediction as seen diagram [Figure 3.2] [4, 6, 7, 8, 9].

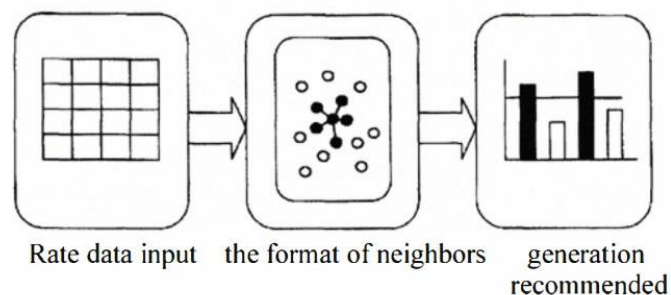


Figure 3.2- Filtering process

3.3.2 User Rating Score Data Input

Generally, input data in recommendation system based on the CF technology consists of user, item, and user opinions on observed items as a matrix 6×5 as shown in [figure 3.3]. Symbol 6 symbolizes the total number of users and 5 symbolizes the total number of items. $R_{u,i}$ is the score of item I_n rated by user U_m .

	Item1	Item2	Item3	Item4	Item5
User1	3	5	5	1	1
User2	4	5	5	2	3
User3	5	4	2	4	5
User4	3	5	4	3	5
User5	1	4	5	2	5
User6	2	5	4	1	2

Figure 3.3 item-User matrix

Once CF computes the similarity between users (in UBCF) or items (in IBCF) and then finds the set of most similar user or similar items, it generates prediction of the target user's interest as the most significant step in CF.

3.4 ESTIMATING SIMILARITY INDEX

3.4.1 Cosine Vector Similarity

Cosine vector similarity is one of the popular metrics in statistics. Since it notionally considers only the angle of two vectors without the magnitude, it is a very useful measurement with data missing preference information as long as it can count the number of times that term appears in the data [17].

In the following formula, the cosine vector similarity looks into the angle between two vectors (the target Item i and the other Item j) of ratings in n - dimensional item space. $R_{i,k}$ is the rating of the target Item i by User k . $R_{j,k}$ is the rating of the other Item j by user k . n is the total number of all rating users to Item i and Item j .

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

Figure 3.4 Cosine Similarity

When the angle between two vectors is near 0 degree (they are in the same direction), Cosine similarity value, $\text{sim}(i,j)$, is 1, meaning very similar. When the angle between two vectors is near 90 degree, $\text{sim}(i,j)$ is 0, meaning irrelevant. When the angle between two vectors is near 180 degree (they are in the opposite direction), $\text{sim}(i,j)$ is -1, meaning very dissimilar. In case of information retrieval using CF, $\text{sim}(i,j)$ ranges from 0 to 1. This is because the angle between two term frequency vectors cannot be greater than 90 degrees [17].

3.4.2 Pearson Correlation Coefficient

Pearson correlation coefficient is one of the popularly used methods in CF to measure how larger a number in one series is, relative to the corresponding number. As following formula shows, it is used to measure the linear correlation

between two vectors (Item i and Item j).

$$r = \frac{\sum (x - m_x)(y - m_y)}{\sqrt{\sum (x - m_x)^2 \sum (y - m_y)^2}}$$

Figure 3.5
Pearson
Correlation

It measures the tendency of two series of numbers, paired up one-to-one, to move together [. When two vectors have a high tendency, the correlation, $sim(i, j)$, is close to 1. When two vectors have a low tendency, $sim(i, j)$ is close to 0. When two vectors have opposite tendency, $sim(i, j)$ is close to -1. As mentioned above in figure 3.5, item-based similarity is computed with the co-rated items where users rated both (Item i and Item j). $R_{i,k}$ is the rating of the target Item i given by User k . $R_{j,i}$ is the rating of the other Item j given by User i . A_i is the average rating of the target Item i for all the co-rated users, and A_j is the average rating of the other Item j for all the co-rated users. n is the total number of ratings users gave to Item i and Item j .

3.4.3 Euclidean Distance Similarity

Euclidean distance method is based on the distance between items. It forms coordinates to put preference values between items and measures Euclidean distance between each point. When distance value between two

points, $sim(i, j)$ is large, it means the two points are not similar. When $sim(i, j)$ is small, it means two points are similar. This is Euclidean distance formula is given below.

$$sim(i, j) = \sqrt{\sum_{k=1}^n (R_{i,k} - R_{j,k})^2}$$

$R_{i,k}$ is the ratings of the target Item i given by User k . $R_{j,k}$ is the ratings of the other Item j given by User k . n is the total number of rating users to Item i and Item

CHAPTER 4

SOFTWARE DEVELOPMENT METHODOLOGY

4.1 ARCHITECTURAL DESIGN

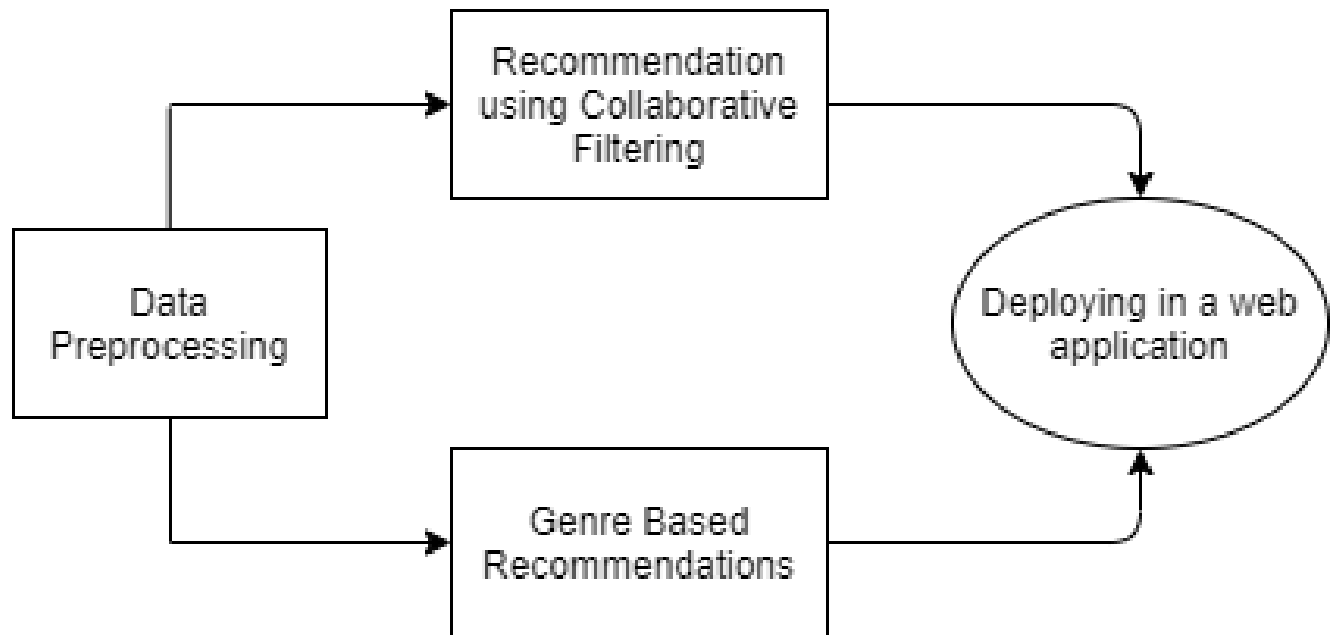


Figure 4.1 After Data Preprocessing

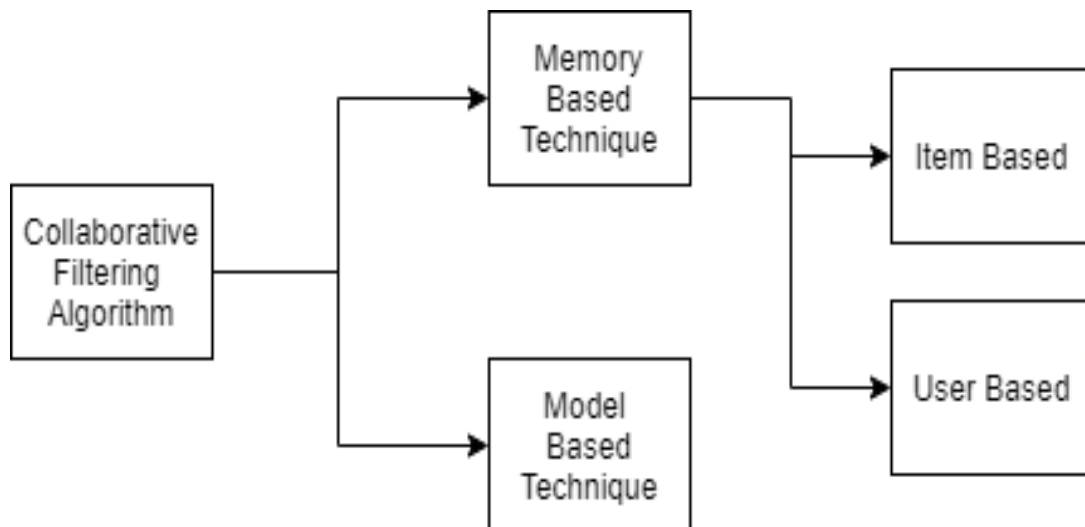


Figure 4.2 Filtering Algorithm

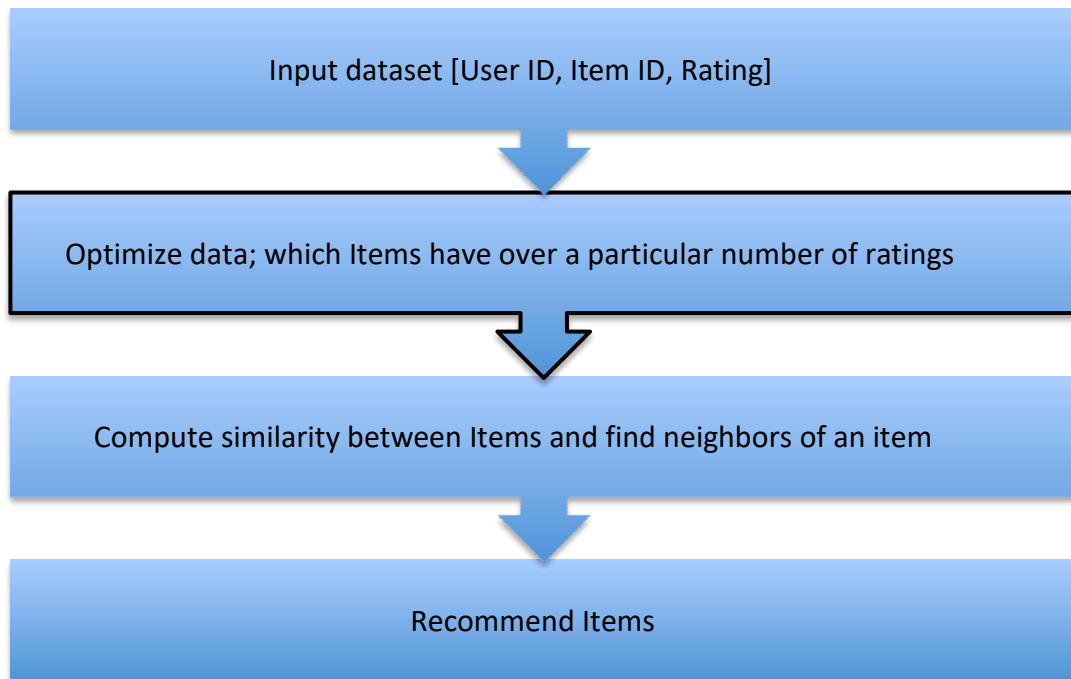


Figure 4.3 Reducing Dimensions

4.2 DESCRIPTION OF DIAGRAM

In the Figure 4.1 after the data has been preprocessed there are two ways of proceeding towards an end result, now depending upon the filtering technique adopted the end result can be computed using collaborative filtering or upon the Genre based movie selection.

For the different way in which how a collaborative filtering technique can be applied are being described in the [Figure 4.2], there are two ways one is a conventional memory-based method while other is a more complex yet impressive model-based method. Here is a scenario of IBCF applying dimension reduction as seen in [Figure 4.3]. This is mainly divided into four steps. This approach is based on general collaborative filtering algorithm. To compute similarity between items, the algorithm uses an optimized data by reducing dimension of items that have the number of ratings less than a specific value. For example, if it needs to consider items that have over 20 ratings from users, it extracts data in terms of items having over 20 ratings. In other words, such items are rated by over 20 users.

4.3 COLLABORATIVE FILTERING

Recommendation systems in various applications have tried to provide users with an accurate recommendation to meet the needs of the user and to bring higher benefits to companies. Collaborative filtering is an effective and well-known technology in recommendation systems. Many web sites, particularly Ecommerce sites, have used collaborative filtering technology in their recommendation systems to personalize the browsing experience for each user as seen [Figure 1]. As successful use cases of collaborative filtering, Amazon increased sales by 29%, Netflix increased movie rentals by 60%, and Google news increased click-through rates by 30.9% .

4.3.1 ITEM BASED FILTERING

Item-based collaborative filtering approach is to predict items by inquiring into similarities between the items and other items that are already associated with the user. For example, as seen in [Figure 4.4], let's say Item A and Item C are very similar. If a User likes Item A, IBCF can recommend Item C to the User. IBCF needs a set of items that the target user has already rated to calculate similarities between items and a target item. And then, it generates prediction in terms of the target item by combining the target user's previous preferences based on these item similarities. In IBCF, users' preference data can be collected in two ways. One is that user explicitly gives rating score to item within a certain numerical scale. The other is that it implicitly analyzes user's purchase records or click-through rate.

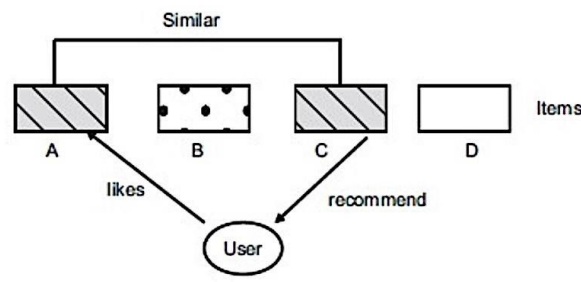


Figure 4.4 Item-based collaborative filtering

4.3.2 USER BASED FILTERING

User-based collaborative filtering approach is to predict items to the target user that are already items of interest for other users who are similar to the target user. For example, as seen [Figure 4.5], let User 1 and User 3 have very similar preference behavior. If User 1 likes Item A, UBCF can recommend Item A to User 3. UBCF needs the explicit rating scores of items rated by users to calculate similarities between users and exploits k-nearest neighbor algorithms to find the nearest neighbors based on user similarities. And then, it generates prediction in terms of items by combining the neighbor user's rating scores based on similarity weighted averaging.

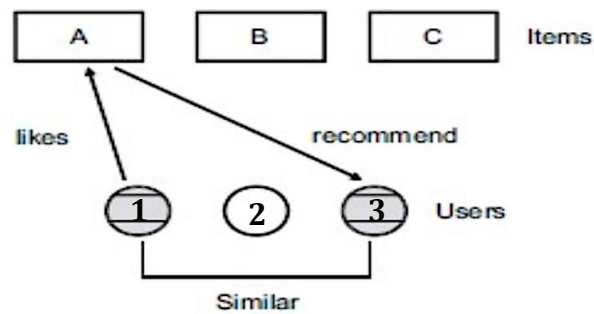


Figure 4.5 User-based collaborative filtering

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 RESULTS

A working system with an efficient accuracy metric was achieved. In this paper we assessed different techniques in Recommendation System in present situation beginning from the customary ones, likewise future examination patterns in Recommendation System would turn out. This would be useful and helpful to the researchers trying to understand the depths of a recommendation system to zero in on different issues relating to RS.

This paper majorly focuses upon the process flow on how the recommendation system for movies is worked out. The things that we noticed based upon the input taken from user by taking factors like Genre, Average Weighted Rating, View Count, Rating count etc. into consideration along with Data analytics performed on each Genre to bring out the best out of Genre(s) was that the model was able to achieve an accuracy of ~71 percent.

The evaluation metric that has been used to achieve the result is hit rate. Hit Rate is the occasions that a right prediction was made in proportion to the quantity of absolute prediction names.

5.2 OUTPUT &SCREENSHOTS

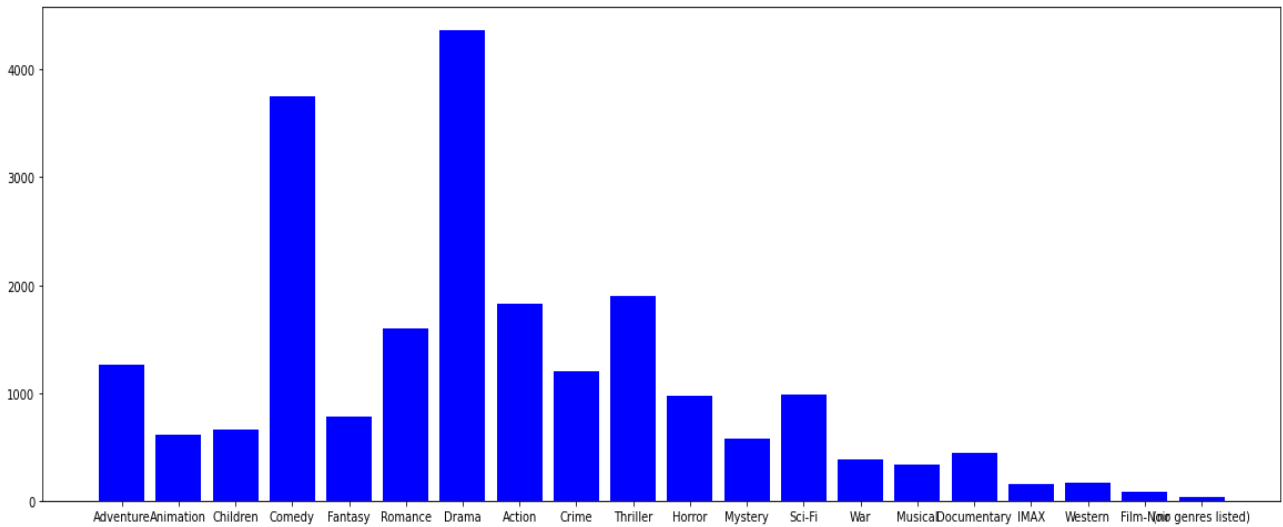


Figure 5.1 Genre based graph

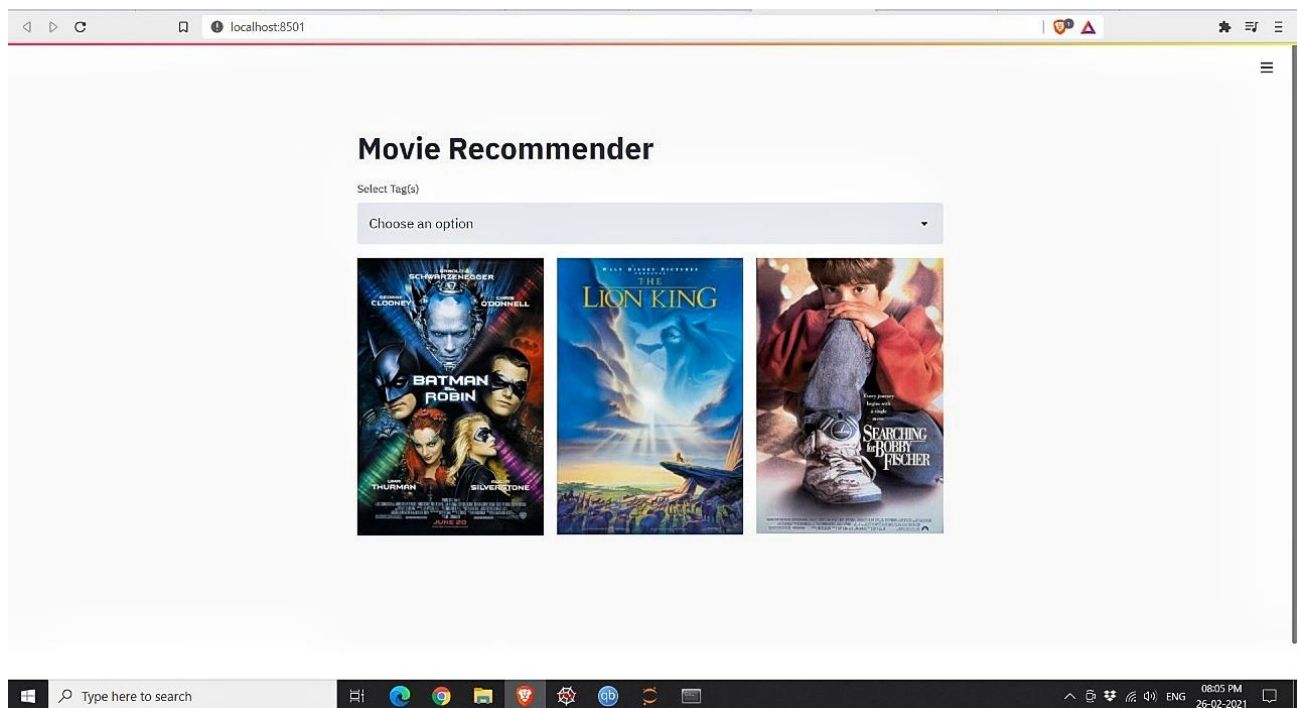


Figure 5.2 Home Screen

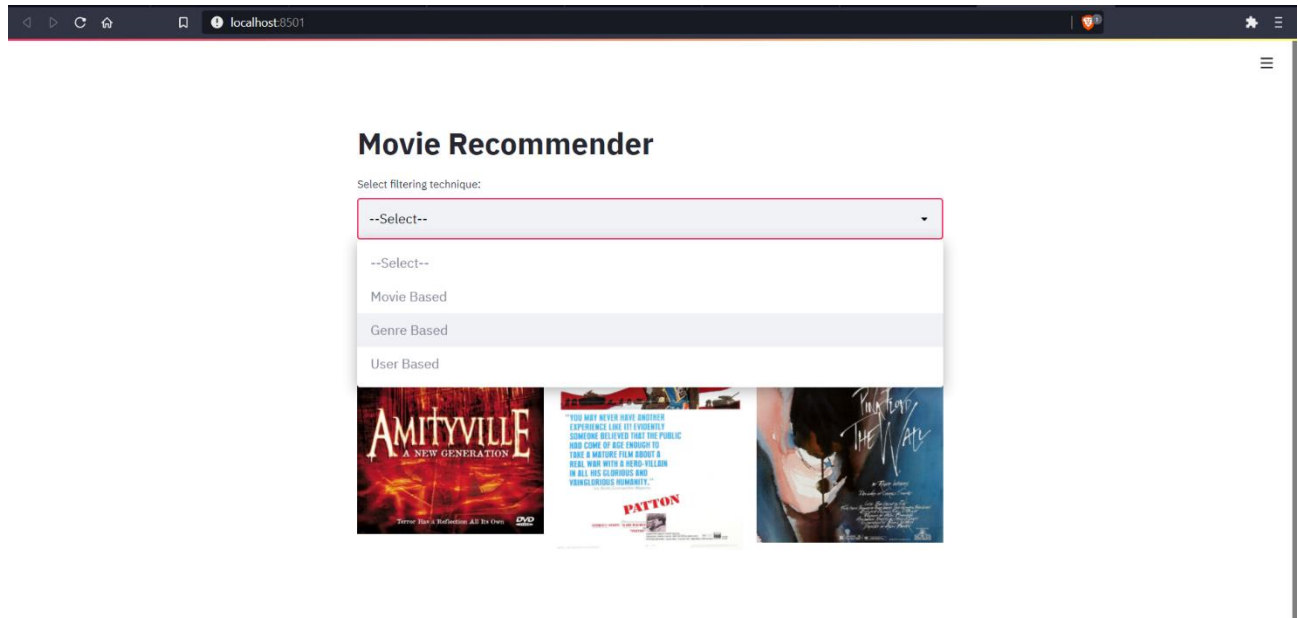


Figure 5.3 Choosing a technique

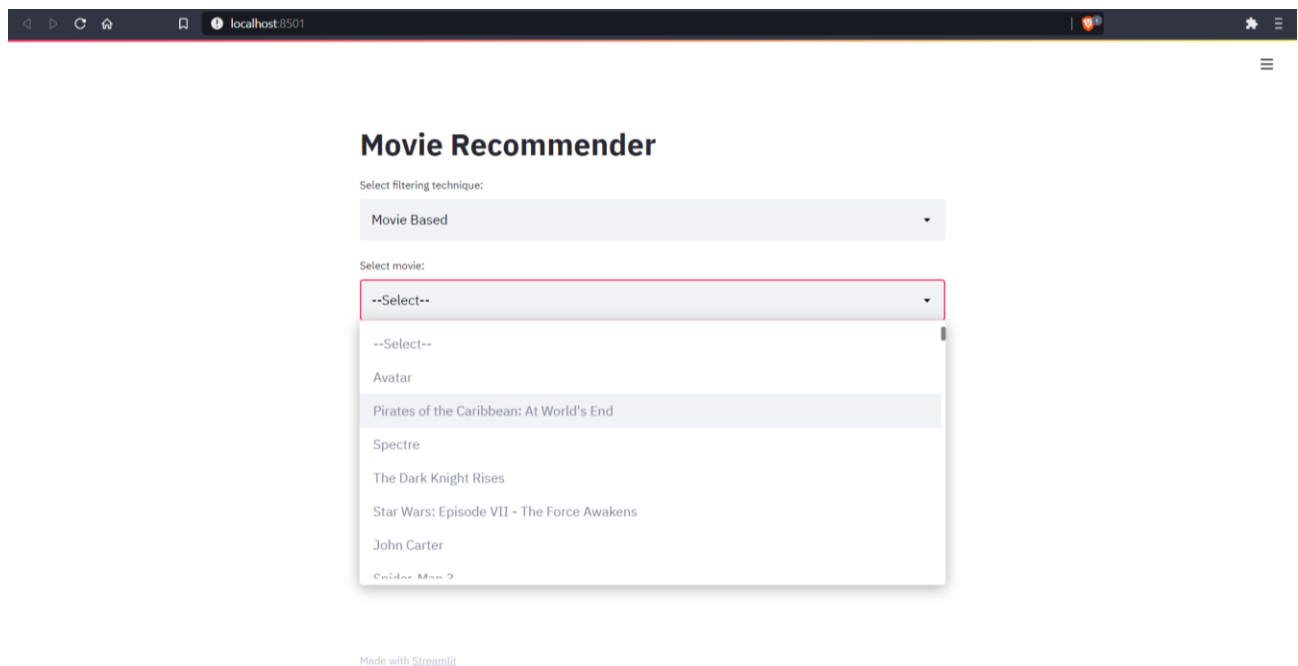


Figure 5.4 feeding the model with an input

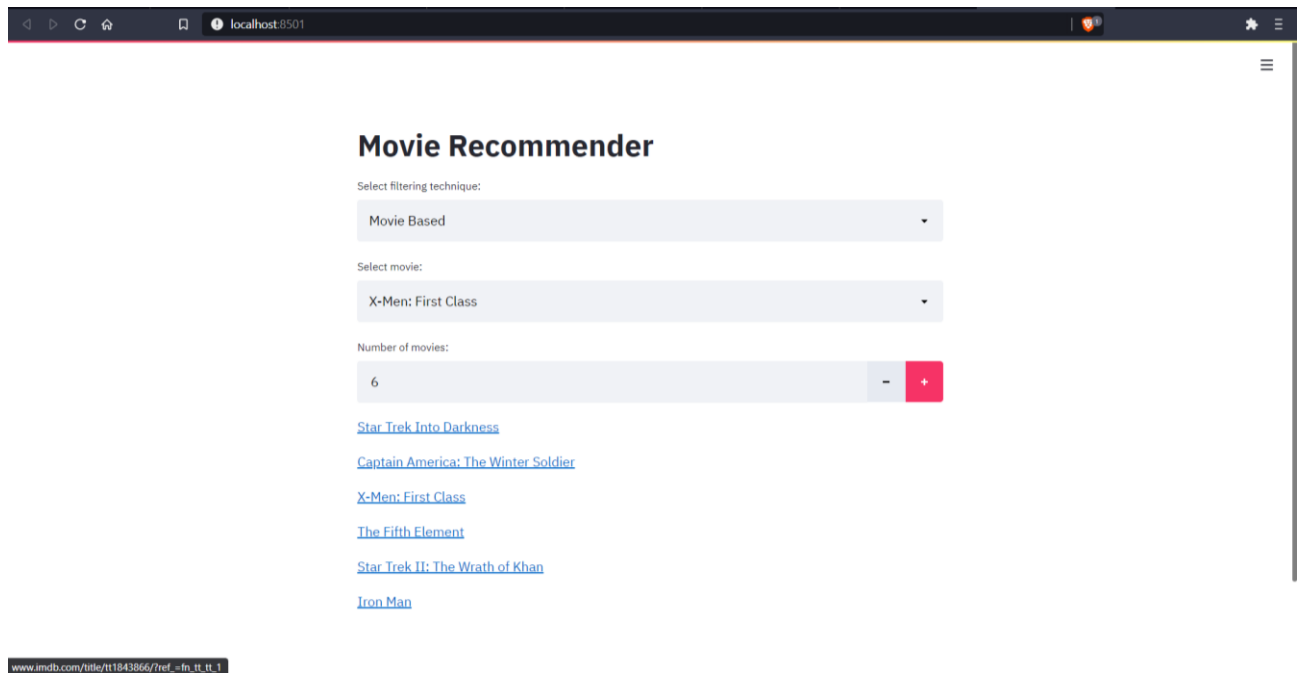


Figure 5.5 Getting results with links to the corresponding movies

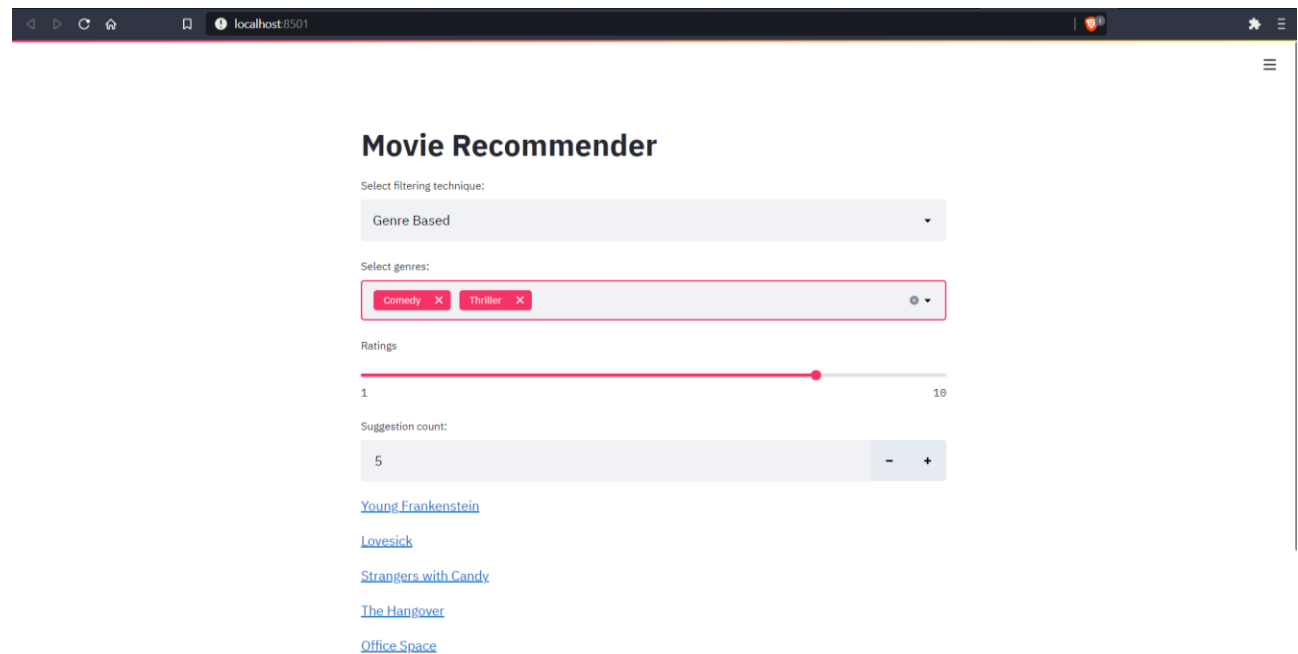


Figure 5.6 Using filters for an Advanced search

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

Recommendation systems have been an important in E-commerce on the web for the customer to suggest items what they would be interested. With the increasing number of users and items, recommendation systems encounter the main shortcoming: data sparsity and data scalability problems, which bring out the poor quality of prediction and the inefficient time consuming. In this paper, I have proposed item-based collaborative filtering approach applying dimension reduction to improve the predictive accuracy and recommendation quality in overcoming the existing limitations. By reducing the noise of dimensional data, it focuses on typical and popular items to compute the similarity between them and to predict the most similar items to users. The experimental results show that this approach makes a considerable impact on providing better accuracy of prediction and much faster execution time in comparison with traditional UBCF and IBCF. It results in improving the quality of recommendation system using collaborative filtering. The potential limitation would use this approach with dataset widely consisting of not enough ratings by users, expecting less accuracy. Therefore, to overcome this challenge, I propose an approach to mix both explicit and implicit ratings to alleviate the data sparsity problem further in this aspect

6.2 FUTURE WORKS

In the nearing future, we will zero in on setting mindful recommender frameworks also, investigate different proficient ways to deal with setting choice along with other future patterns in RS, to make RS more efficient more exact more reliable for the general audience, a greater understanding towards user's likes and dislikes enabling the model to choose accordingly by personalizing distinct recommendations.

Recommendation systems have been a functioning zone of examination for a decade or somewhere in the vicinity and keeps on being a fascinating exploration area. Despite some challenges there have been some researches carried out in this field which has given greater

enhancements to the current working systems boosting the accuracy metric by adding more features to the working system. The practical dependency on Recommendation frameworks will increase as time progresses. Constraints and dynamics like data privacy, user interest, diversity in the input data set will play a very crucial role in making a physically reliable Recommendation system. Aspects like Data Privacy also play an important role, the bigger the dataset the more extensive the risk becomes if Data leaking occurs by a chance.

Predictive accuracy is substantially improved when blending multiple predictors. Our experience is that most efforts should be concentrated in deriving substantially different approaches, rather than refining a single technique. Consequently, hence a solution with an ensemble of many methods would most certainly be more successful.

REFERENCES

- [1] Schafer, J. Ben, Joseph Konstan, and John Riedl. 1999. "Recommender Systems In E-Commerce." In 1St ACM Conference On Electronic Commerce, 158-166.
- [2] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. 2009. "A Survey Of Collaborative Filtering Techniques." Advances In Artificial Intelligence 2009: 1-19. doi:10.1155/2009/421425.
- [3] Melville, Prem, and Vikas Sindhwani. 2010. "Recommender Systems." Encyclopedia Of Machine Learning.
- [4] Xingyuan Li.2011 "Collaborative Filtering Recommendation Algorithm Based on Cluster", International Conference on Computer Science and network Technology(ICCSNT), IEEE, 4: 2682-2685.
- [5] Francesco Ricc, Lior Rokach, Bracha Shapira.2011. Recommender Systems Handbook. NY:Springer, 1-35
- [6] Sarwar, George Kaypi, Joseph Konstan and John Riedl.2000. "Application of Dimensionality Reduction in Recommender Systems -- A 6 Study." In ACM WebKDD Workshop.
- [7] Sarwar, George Kaypi, Joseph Konstan, John Riedl.2000. "Analysis of recommendation algorithms for E-commerce." In the Second ACMConference on Electronic Commerce, 158–167 42
- [8] Sarwar, George Kaypi, Joseph Konstan, John Riedl.2001. "Item-based Collaborative Filtering Recommendation Algorithms." In the 10th International World Wide Web Conference, 285-295
- [9] Gong, Songjie. 2010. "A Collaborative Filtering Recommendation Algorithm Based On User Clustering And Item Clustering." JSW 5 (7). doi:10.4304/jsw.5.7.745-752.
- [10] Yan Shi, Xiao, HongWu Ye, and SongJie Gong. 2008. "A Personalized Recommender Integrating Item-Based And User-Based Collaborative Filtering." ISBIM '08 International Seminar On Business And Information Management 1 (2008): 264-267. [11] Mangalindan,

JP. 2012. "Amazon'S Recommendation Secret." Fortune.
<http://fortune.com/2012/07/30/amazons-recommendation-secret/>.

[12] Koren, Yehuda.2009. "Collaborative Filtering with Temporal Dynamics." 15th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 09), ACM (2009): 447-455.

[13] Liu, Jiahui, Dolan, Peter, Pedersen, Elin Rønby.2010. " Personalized news recommendation based on click behavior", In: Rich, et al. (eds.) In the 14th Int. Conf. on Intelligent User Interfaces (IUI), ACM, (2010): 31–40.

[14] Owen, Sean, Anil, Robin, Dunning, Ted, Friedman, Ellen. 2011 . Mahout in action. Shelter Island NY: Manning. ()

[21] GroupLens,. 2013. "Movielens". <http://grouplens.org/datasets/movielens/>

APPENDIX

FILE A:

```
import streamlit as st
```

```
import pandas as pd
```

```
import numpy as np
```

```
from scipy.sparse import csr_matrix
```

```
from sklearn.neighbors import NearestNeighbors
```

```
import matplotlib.pyplot as plt
```

```
def userbased(movie_name):
```

```
    movies = pd.read_csv("movies.csv")
```

```
    ratings = pd.read_csv("ratings.csv")
```

```
    final_dataset = ratings.pivot(index='movieId',columns='userId',values='rating')
```

```
    final_dataset.fillna(0,inplace=True)
```

```
    no_user_voted = ratings.groupby('movieId')['rating'].agg('count')
```

```
    no_movies_voted = ratings.groupby('userId')['rating'].agg('count')
```

```
    f,ax = plt.subplots(1,1,figsize=(16,4))
```

```

# ratings['rating'].plot(kind='hist')

plt.scatter(no_user_voted.index,no_user_voted,color='mediumseagreen')

plt.axhline(y=10,color='r')

plt.xlabel('MovieId')

plt.ylabel('No. of users voted')

final_dataset = final_dataset.loc[no_user_voted[no_user_voted > 10].index,:]

f,ax = plt.subplots(1,1,figsize=(16,4))

plt.scatter(no_movies_voted.index,no_movies_voted,color='mediumseagreen')

plt.axhline(y=50,color='r')

plt.xlabel('UserId')

plt.ylabel('No. of votes by user')

#st.pyplot(f)

final_dataset=final_dataset.loc[:,no_movies_voted[no_movies_voted > 50].index]

sample = np.array([[0,0,3,0,0],[4,0,0,0,2],[0,0,0,0,1]])

sparsity = 1.0 - ( np.count_nonzero(sample) / float(sample.size) )

csr_sample = csr_matrix(sample)

csr_data = csr_matrix(final_dataset.values)

final_dataset.reset_index(inplace=True)

knn = NearestNeighbors(metric='cosine', algorithm='brute', n_neighbors=20,

```

```

n_jobs=-1)

knn.fit(csr_data)

n_movies_to_reccomend = 10

movie_list = movies[movies['title'].str.contains(movie_name)]

if len(movie_list):

    movie_idx= movie_list.iloc[0]['movieId']

    movie_idx = final_dataset[final_dataset['movieId'] == movie_idx].index[0]


    distances , indices =
knn.kneighbors(csr_data[movie_idx],n_neighbors=n_movies_to_reccomend+1)

    rec_movie_indices =
sorted(list(zip(indices.squeeze().tolist(),distances.squeeze().tolist())),\

        key=lambda x: x[1]))[:0:-1]


    recommend_frame = []


    for val in rec_movie_indices:

        movie_idx = final_dataset.iloc[val[0]]['movieId']

        idx = movies[movies['movieId'] == movie_idx].index

recommend_frame.append({'Title':movies.iloc[idx]['title'].values[0],'Distance':val[1]})

```

```

        df =
pd.DataFrame(recommend_frame,index=range(1,n_movies_to_reccomend+1))

        st.table(df['Title'])

        #st.dataframe(df['Title'])

        return df

    else:

        return "No movies found. Please check your input"

```

FILE B:

```

import streamlit as st

import numpy as np

import seaborn as sns

import pandas as pd

import matplotlib.pyplot as plt

from scipy.sparse import csr_matrix

from sklearn.neighbors import NearestNeighbors

from userbased import*

#from trending import*

```

```
import random
```

```
st.title('Movie Recommender')
```

```
links = pd.read_csv("links.csv")
```

```
#links.head()
```

```
auti_df=links.drop(links.index[auti])
```

```
links_list=[]
```

```
links_list=list(auti_df['img_link'])
```

```
q=st.selectbox("Select Tag(s)",['Choose an option','Best of all Times','According to  
Genre','Item Based Filtering', 'User Based Filtering'])
```

```
if(q=='User Based Filtering'):
```

```
    movie_name = st.text_input("Your favourite name movie goes here")
```

```
    k=st.button("Suggest me Movies")
```

```
    if(k):
```

```
        userbased(movie_name)
```

```
elif(q=='Choose an option'):
```

```
    rands=[]
```

```
    for i in range(3):
```

```

rands.append(random.randint(0,406))

#print(rands)

#st.image(links_list[rands[0]],width = 400)


col1, col2, col3= st.beta_columns(3)

with col1:

    #st.header("A cat")

    st.image(links_list[rands[0]], use_column_width=True)

with col2:

    #st.header("A dog")

    st.image(links_list[rands[1]], use_column_width=True)

with col3:

    #st.header("An owl")

    st.image(links_list[rands[2]], use_column_width=True

```