Mini Project Report

on

MOVIE RECOMMENDER SYSTEM

Submitted in partial fulfillment for award of

BACHELOR OF TECHNOLOGY

Degree

In

COMPUTER SCIENCE & ENGINEERING

Under the Guidance of:

Ms. Isha Gupta
Assistant Professor

CSE Dept

Submitted By:

SAUMYA AGRAWAL
(2000330100196)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAJ KUMAR GOEL INSTITUTE OF TECHNOLOGY

DELHI-MEERUT ROAD, GHAZIABAD

Affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow

November 2022

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Hello everyone here we are talking about the project I am working on. Here I have developed a mini project on a **movie recommender system**. Everyone loves movies irrespective of age, gender, race, color, or geographical location. We all in a way are connected to each other via this amazing medium. Yet what most interesting is the fact that how **unique** our choices and combinations are in terms of movie preferences. Some people like genre-specific movies be it a thriller, romance, or sci-fi, while others focus on lead actors and directors. When we take all that into account, it's astoundingly difficult to generalize a movie and say that everyone would like it. But with all that said, it is still seen that similar movies are liked by a specific part of the society. The data used for the present research work is taken from Tmdb which is a dataset of 5000 movies and credits with different attributes. The research methodology used in this study is mainly designed as an empirical work based on both secondary dataset and primary dataset, obtained through pre tested questionnaire, internet browsing, websites, books, magazines etc. Secondary data has contributed in a significant way to understand the scope as well as it helped in developing the conceptual framework. This project is a web based movie recommender system for an existing dataset. The project objective is torecommend movies as accurately as possible based on the given movie by the user. A large number of companies are making use of recommendation systems to increase user interaction and enrich a user's shopping experience. Recommendation systems have several benefits, the most important being customer satisfaction and revenue. Movie Recommendation system is very powerful and important system. But, due to the problems associated with pure collaborative approach, movie recommendation systems also suffer with poor recommendation quality and scalability issues.

"Every time I go to a movie, it's magic, no matter what the movie's about."

– **Steven Spielberg**

Abstract

In this hustling world, entertainment is a necessity for each one of us to refresh our mood and energy. Entertainment regains our confidence for work and we can work more enthusiastically. For revitalizing ourselves, we can listen to our preferred music or can watch movies of our choice. For watching favorable movies online, we can utilize movie recommendation systems, which are more reliable, since searching of preferred movies will require more and more time which one cannot afford to waste.
A movie recommendation is important in our social life due to its strength in providing enhanced entertainment. Such a system can suggest a set of movies to users based on their interest, or the popularities of the movies. Although, a set of movie recommendation systems have been proposed, most of these either cannot recommend a movie to the existing users efficiently or to a new user by any means. In this project we propose a movie recommendation system that has the ability to recommend movies to a new user as well as the others. It mines movie databases to collect all the important information, such as, popularity and attractiveness, required for recommendation. It generates movie swarms not only convenient for movie producer to plan a new movie but also useful for movie recommendation. Experimental studies on the real data reveal the efficiency and effectiveness of the proposed system.

In this project we have used collaborative filtering method along with cosine similarity to create vectors and find similarities in those vectors to recommend movie. Also, for the frontend we have used PyCharm application (streamlit).

KEYWORDS: recommender system, content-based, collaborative filtering, similarity, movie

Problem Definition:

Movie recommender systems are meant to give suggestions to the users based on the features they love the most. A highly performing movie recommendation will suggest movies that match the similarities with the highest degree of performance. This study conducts a systematic literature review on movie recommender systems. It highlights the filtering criteria in the recommender systems, algorithms implemented in movie recommender systems, the performance measurement criteria, the challenges in implementation, and recommendations for future research. Some of the most popular machine learning algorithms used in movie recommender systems such as *K*-means clustering, principal component analysis, and self-organizing maps with principal component analysis etc.

The goal of the project is to recommend a movie to the user. Providing related content out of relevant and irrelevant collection of items to users of online service providers
The purpose of any recommender system is to help customers/users narrow down their broad ideasand enable them to finalize the movie they want to watch as per their interest.
Between user and item centric operations, significant dependencies exist. An effective system of recommendation explores this operation and generates acceptable recommendations. For instance, instead of an action movie, a user interested in a historical documentary is more likely to be interested in another historical documentary or an educational program. Different categories of things can display substantial associations in several instances, which can be leveraged to make more specific recommendations. Due to its strength in providing enhanced entertainment and tailored user experience, a film recommendation system is critical. Such a system can recommend a collection of films to users based on their interest or Film popularity.

The takeaway from this is that categorization and genre listing of movies should be taken care of. Misrepresentation of features like genres, actor names, movie keywords can lead to incorrect search results.

Objective

- Improving the Accuracy of the recommendation system
- Improve the Quality of the movie Recommendation system
- Improving the Scalability
- Enhancing the user experience.

SCOPE OF THE STUDY

- Finding out the strengths and weakness of any recommender system
- The objective of this project is to provide accurate movie recommendations to users. The goal of the project is to improve the quality of movie recommendation system, such as accuracy, quality and scalability of system than the pure approaches.
- improving scalability, accuracy and quality of movie recommendation systems
- Finding out the perception and liking of the users about certain movies

METHODOLOGY OF THE STUDY

The data used for the present research work is TMDB dataset of 5000 movies and credits. The research methodology used in this study is mainly designed as an empirical work based on both datasets, obtained through pre tested questionnaire, internet browsing, Secondary data has contributed in a significant way to understand the scope as well as it helped in developing the conceptual framework. The algorithm used in this system is cosine similarity which make use of vectorization method to covert all movies into a vector and the movies which are nearer will be related to each other in some or other way.

Need of the System:

One key reason why we need a recommender system in modern society is that people have too much options to use from due to the prevalence of Internet. In the past, people used to shop in a physical store, in which the items available are limited. For instance, the number of movies that can be placed in a Blockbuster store depends on the size of that store. By contrast, nowadays, the Internet allows people to access abundant resources online. Netflix, for example, has an enormous collection of movies. Although the amount of available information increased, a new problem arose as people had a hard time selecting the items they actually want to see. This is where the recommender system comes in.

CHAPTER 2

Hardware and Software Requirement

Software Requirements:

- Microsoft Windows 7/8/10 or Linux.
- Any Python editor (version greater than 3.1.0)
- PyCharm editor
- Chrome or any other browser.

Hardware Requirements:

- Intel® Celeron® Processor 847, 1.10 GHz, or equivalent
- Minimum of 512 MB
- 3 GB or more Hard Disk Drive or above.

CHAPTER 3

Recommendation System

3.1 What is Recommendation System?

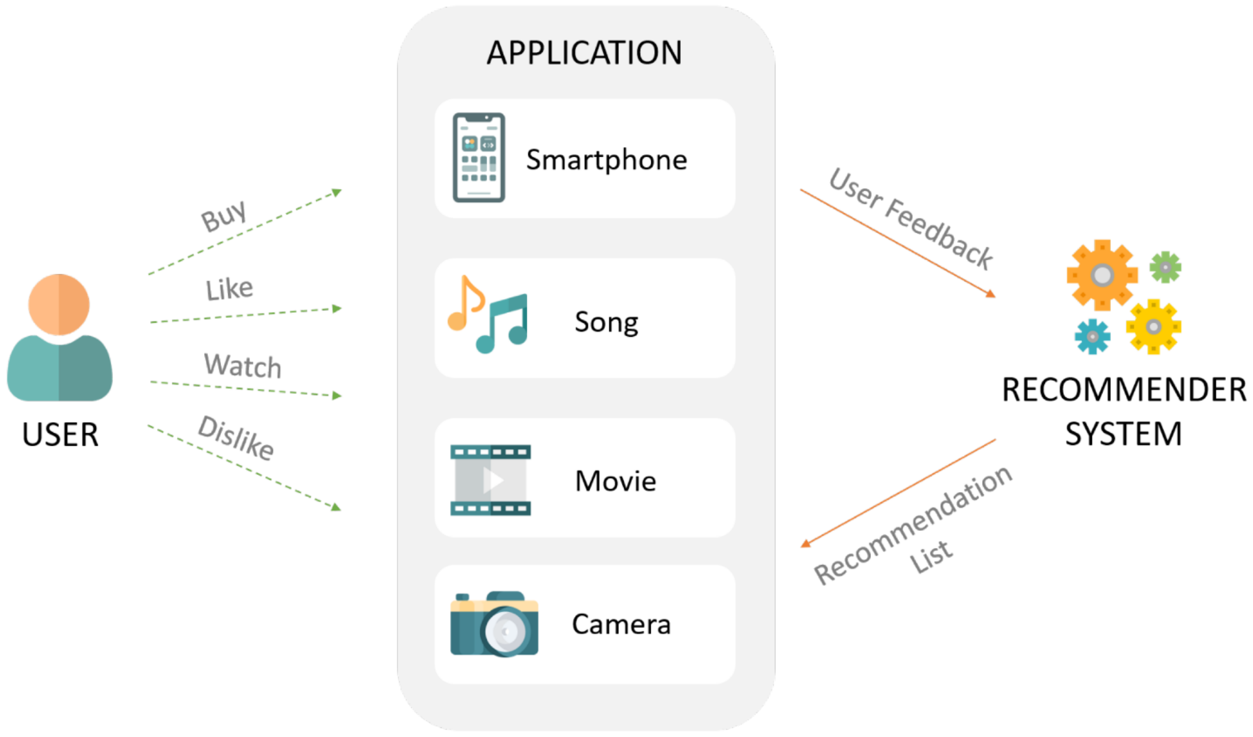


Fig 3.1 WHAT IS RECOMMENDER SYSTEM

Recommender systems are the systems that are designed to recommend things to the user based on many different factors. These systems predict the most likely product that the users are most likely to purchase and are of interest to. Companies like Netflix, Amazon, etc. use recommender systems to help their users to identify the correct product or movies for them.

The recommender system deals with a large volume of information present by filtering the most important information based on the data provided by a user and other factors that take care of the user’s preference and interest. It finds out the match between user and item and imputes the similarities between users and items for recommendation.

Both the users and the services provided have benefited from these kinds of systems. The quality and decision-making process has also improved through these kinds of systems.

There are many different things that can be recommended by the system like movies, books, news, articles, jobs, advertisements, etc. Netflix uses a recommender system to recommend movies & web-series to its users. Similarly, YouTube recommends different videos. There are many examples of recommender systems that are widely used today.

Recommender system -

- Benefits users in finding items of their interest.
- Help item providers in delivering their items to the right user.
- Identity products that are most relevant to users.
- Personalized content.
- Help websites to improve user engagement.

3.2 NEED OF RECOMMENDER SYSTEM

From a user’s perspective, they are catered to fulfil the user’s needs in the shortest time possible. For example, the type of content you watch on Netflix or Hulu. A person who likes to watch only *Korean drama* will see titles related to that only but a person who likes to watch *Action-based* titles will see that on their home screen.

From an organization’s perspective, they want to keep the user as long as possible on the platform so that it will generate the most possible profit for them. With better recommendations, it creates positive feedback from the user as well. What good it will be to the organization to have a library of 500K+ titles when they cannot provide proper recommendations?

Recommendations are a great way to keep you watching but for Raghu the recommendations he gets wrong. But how? Well, as you know that recommendation systems are catered for **a user** but not for **multiple users.** Raghu lives in a joint family and everyone uses a single system to watch what they want. While OTT platforms give you a choice of adding multiple profiles but everyone else has already taken those and he is left with a single profile to share with his grandparents. So, Raghu decides to create his movie recommendation system. Before getting started he should understand the different types of recommendation systems.

3.3 TYPES OF RECOMMENDER SYSTEM

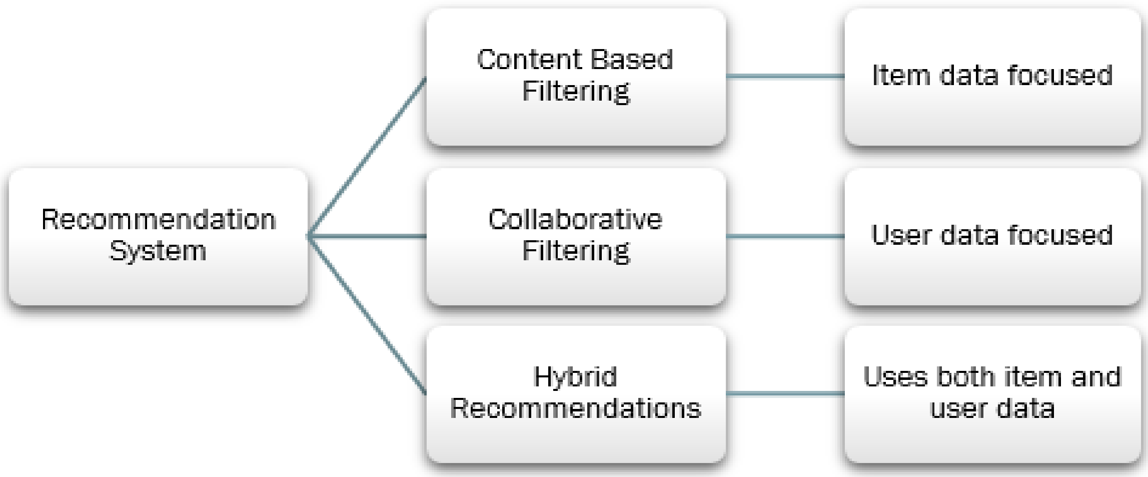


Fig 3.2 TYPES

1. <u>Popularity-Based Recommendation System</u>

It is a type of recommendation system which works on the principle of popularity and or anything which is in trend. These systems check about the product or movie which are in trend or are most popular among the users and directly recommend those.

For example, if a product is often purchased by most people then the system will get to know that that product is most popular so for every new user who just signed it, the system will recommend that product to that user also and chances becomes high that the new user will also purchase that.

<u>Merits of popularity-based recommendation system</u>

- It does not suffer from cold start problems which means on day 1 of the business also it can recommend products on various different filters.
- There is no need for the user's historical data.

Demerits of popularity-based recommendation system

- Not personalized
- The system would recommend the same sort of products/movies which are solely based upon popularity to every other user.

Example

Google News and YouTube

2. Classification Model

The model that uses features of both products as well as users to predict whether a user will like a product or not.

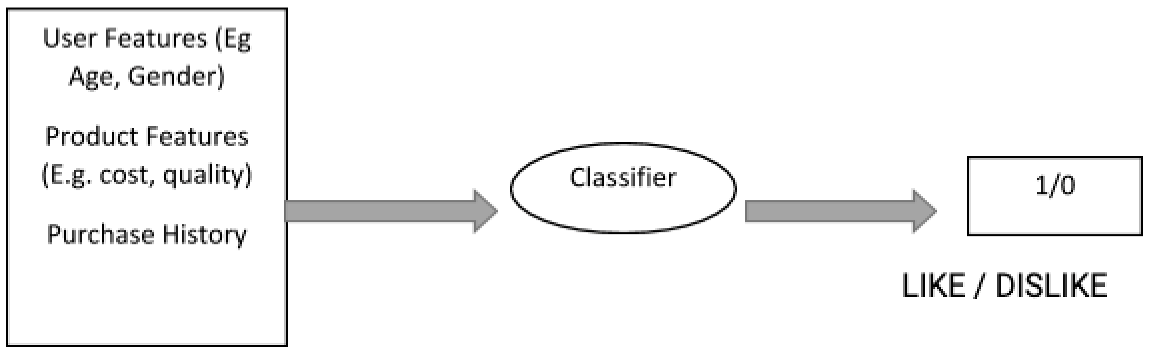

User Features (Eg Age, Gender)
Product Features (E.g. cost, quality)
Purchase History
Classifier
1/0
LIKE / DISLIKE

Fig 3.3 Classification model

The output can be either 0 or 1. If the user likes it then 1 and vice-versa.

Limitations of Classification Model

- It is a rigorous task to collect a high volume of information about different users and also products.
- Also, if the collection is done then also it can be difficult to classify.
- Flexibility issue.

3. Collaborative-Based Recommendation System

It is considered to be one of the very smart recommender systems that work on the similarity between different users and also items that are widely used as an e-commerce website and also online movie websites. It checks about the taste of similar users and does recommendations. The similarity is not restricted to the taste of the user moreover there can be consideration of similarity between different items also. The system will give more efficient recommendations if we have a large volume of information about users and items.

User-Based:

Where we try to find similar users based on their item choices and recommend the items. A user-item rating matrix is created at first. Then, we find the correlations between the users and recommend items based on correlation.

Item Based:

Where we try to find a similar item based on their user's choices and recommend the items. A user-user item rating matrix is created at first. Then, we find the correlations between the items and recommend items based on correlation.

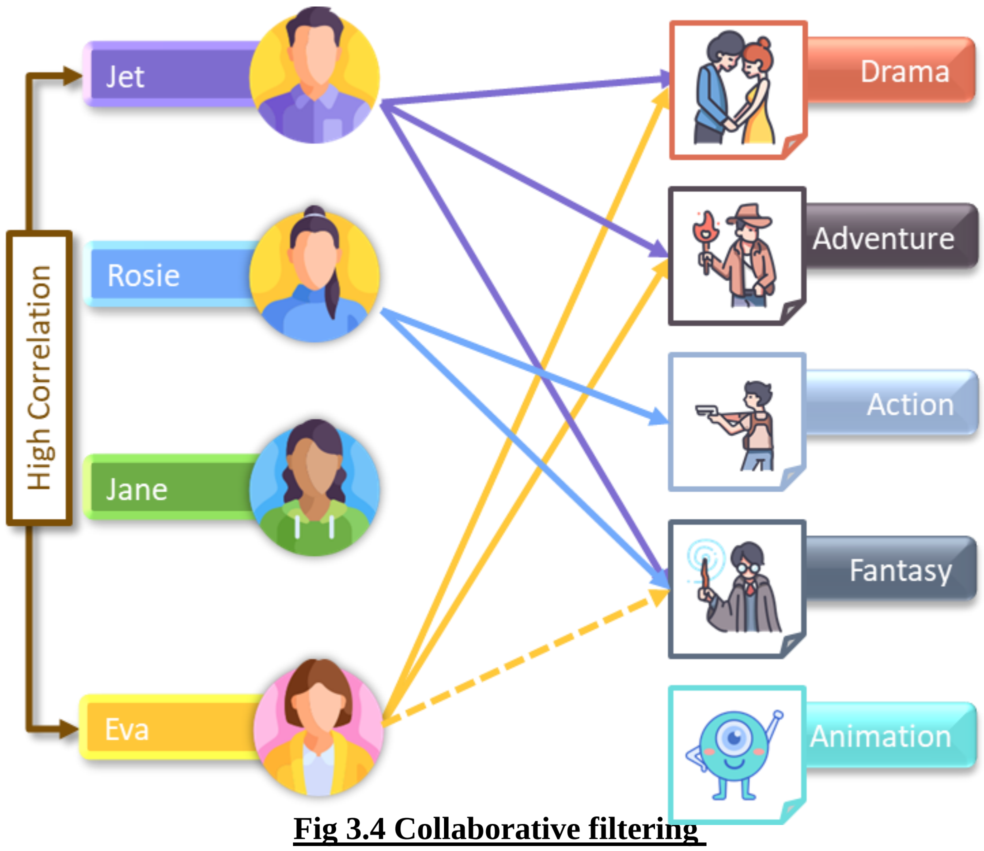


Fig 3.4 Collaborative filtering

4. Content-Based Filtering

It is another type of recommendation system which works on the principle of similar content. If a user is watching a movie, then the system will check about other movies of similar content or the same genre of the movie the user is watching. There are various fundamentals attributes that are used to compute the similarity while checking about similar content.

The fundamental differences between the two approaches are that content-based systems recommend items based on content features (no need for data about other users; recommendations about niche items, etc.) whereas collaborative filtering is based on user behaviour only and recommends items based on users with similar patterns (no domain knowledge; serendipity, etc.). A content-based filtering method works by making movie proposals to the user based on the content in the movies. It recognizes that clustering in the collaborative filtering recommendations may not match the preferences of the users

There are different scenarios where we need to check about the similarities, so there are different metrics to be used. For computing the similarity between numeric data, Euclidean distance is used, for textual data, cosine similarity is calculated and for categorical data, Jaccard similarity is computed.

Merits

- There is no requirement for much of the user's data.
- We just need item data that enable us to start giving recommendations to users.
- A content-based recommender engine does not depend on the user's data, so even if a new user comes in, we can recommend the user as long as we have the user data to build his profile.
- It does not suffer from a cold start.

Demerits

Items data should be in good volume and Features should be available to compute the similarity.

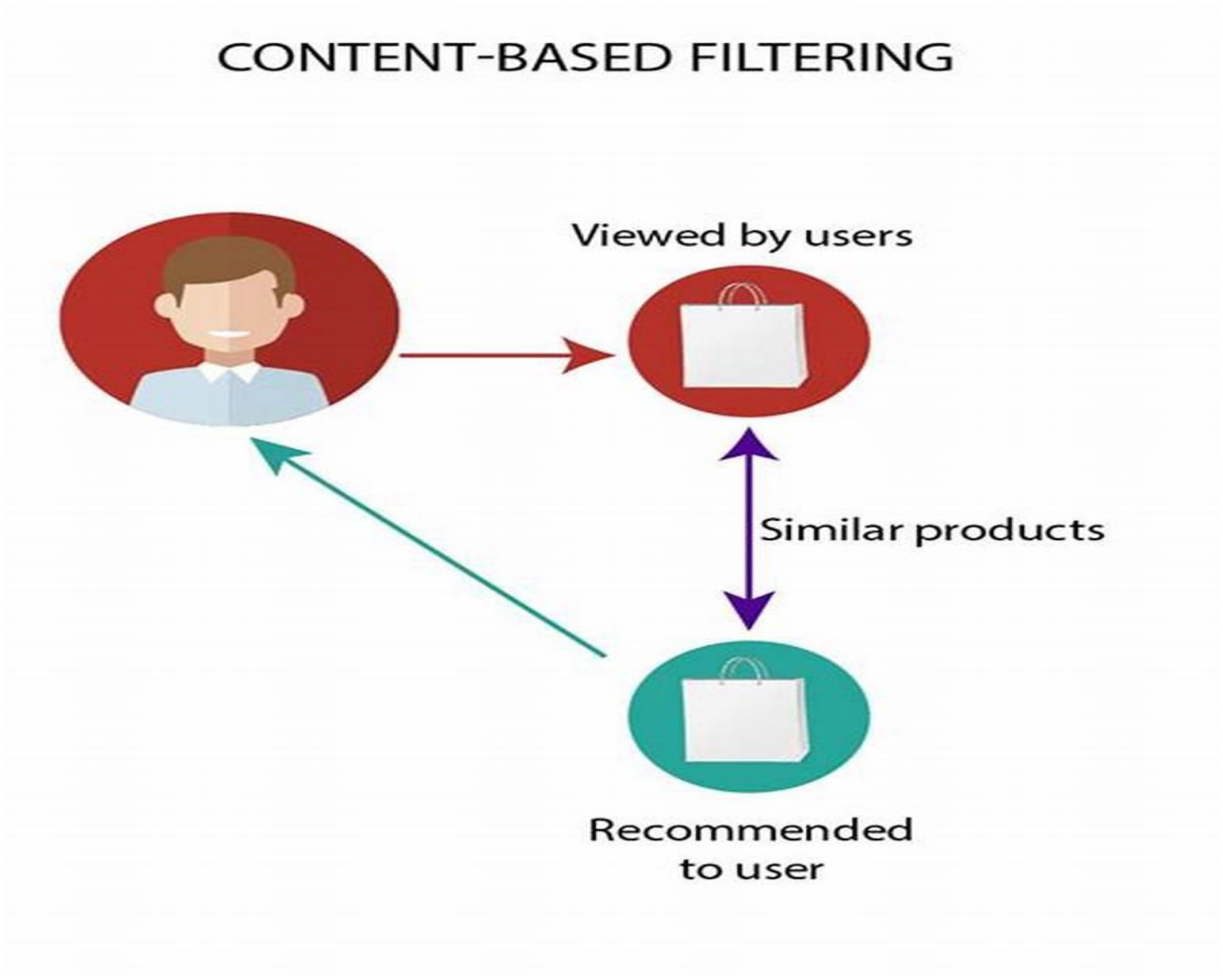


Fig 3.5 Content based filtering

3.4 DATASET

We have been using the TMDB 5K dataset for this research project.

This dataset is made of:

• 100,000 ratings on 5000 movies from 943 users

• At least 20 movies were rated by each user. The information was gathered from the TMDB website (https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata?select=tmdb_5000_movies.csv)

Columns in dataset:

- homepage
- id
- original_title
- overview
- popularity
- production_companies
- production_countries
- release_date
- spoken_languages
- status

- tagline
- vote_average

3.5 APPLICATIONS

Here are some of the examples of the pioneers in creating algorithms for recommendation systems and using them to serve their customers better in a personalized manner. These are:

GroupLens:
– Helped in developing initial recommender systems by pioneering collaborative filtering model
– It also provided many data-sets to train models including MovieLens and BookLens

Amazon:
– Implemented commercial recommender systems
– They also implemented a lot of computational improvements

Netflix Prize:
– Pioneered Latent Factor/ Matrix Factorization models

Google-Youtube:
– Hybrid Recommendation Systems
– Deep Learning based systems
– Social Network Recommendations

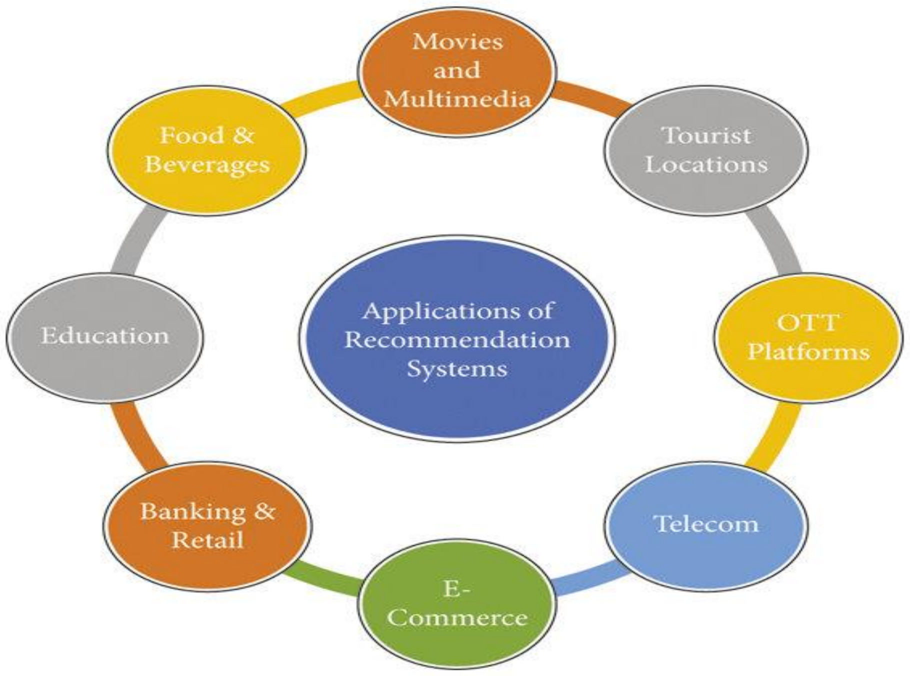


Fig 3.6 Applications

3.6 Project Planning

Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment. Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure. The logical dependencies between tasks are defined using an activity network diagram that enables identification of the critical path. Float or slack time in the schedule can be calculated using project management software. Then the necessary resources can be estimated and costs for each activity can be allocated to each resource, giving the total project cost. At this stage, the project plan may be optimized to achieve the appropriate balance between resource usage and project duration to comply with the project objectives. Once established and agreed, the plan becomes what is known as the baseline. Progress will be measured against the baseline throughout the life of the project

3.7 Purposes

Recommender systems are beneficial to both service providers and users. They reduce transaction costs of finding and selecting items in an online shopping environment. Recommendation systems have also proved to improve decision making process and quality. In e-commerce setting, recommender systems enhance revenues, for the fact that they are effective means of selling more products. In scientific libraries, recommender systems support users by allowing them to move beyond catalog searches. Therefore, the need to use efficient and accurate recommendation techniques within a system that will provide relevant and dependable recommendations for users cannot be over-emphasized.

3.8 CODE

3.8.1 PYTHON JUYPTER NOTEBOOK CODE

```
import numpy as np # linear algebra
import pandas as pd  # data processing, CSV file I/O (e.g. pd.read_csv)

movies=pd.read_csv("tmdb_5000_movies.csv")
credits=pd.read_csv("tmdb_5000_credits.csv")

movies.head(2)
movies.shape
credits.head(2)
credits.shape

 movies = movies.merge(credits,on='title')
movies.head(2)
#following is list of attributeswhich is required for our recommeder system =
#id
 #genre
#keywords
#overview
#title
#cast
#crew
movies = movies[['movie_id','genres','keywords','overview','title','cast','crew']]
movies.head(2)
movies.isnull().sum()
movies.dropna(inplace=True)
movies.duplicated().sum()
movies.iloc[0].genres
```

```
#'[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"},
{"id": 878, "name": "Science Fiction"}]'
 #['action','adventure','fantasy','scifi']
import ast
def convert(text):
        L = []
        for i in ast.literal_eval(text):
                L.append(i['name'])
         return L

movies['genres'] = movies['genres'].apply(convert)
movies['keywords'] = movies['keywords'].apply(convert)
movies.head()

def convert3(obj):
        L=[]
        counter=0
        for i in ast.literal_eval(obj):
                if counter!=3:
                        L.append(i["name"])
                        counter+=1
        return L

movies['cast'] = movies['cast'].apply(convert3)
movies.head()

def fetch_director(text):
        L = []
        for i in ast.literal_eval(text):
                if i['job'] == 'Director':
                        L.append(i['name'])
        return L
```

```
movies['crew'] = movies['crew'].apply(fetch_director)
movies.head()
movies.overview[0]
movies['overview'] = movies.overview.apply(lambda x : x.split())
movies.sample(5)

movies['genres'] = movies['genres'].apply(lambda x:[i.replace(" ","") for i in x ])
movies['cast'] = movies['cast'].apply(lambda x:[i.replace(" ","") for i in x ])
movies['crew']= movies['crew'].apply(lambda x:[i.replace(" ","") for i in x ])
movies['keywords'] = movies['keywords'].apply(lambda x:[i.replace(" ","") for i in x ])

movies.head()
movies['tags']=movies['overview']+movies['genres'] + movies['keywords']+ movies['cast'] +
movies['crew']
movies.head()

new_df=movies[['movie_id','title','tags']]
new_df
new_df['tags'] = new_df['tags'].apply(lambda x: " ".join(x))
new_df.head()

import nltk
from nltk.stem.porter import PorterStemmer
ps=PorterStemmer()
def stem(text):
        y=[]
         for i in text.split():
                y.append(ps.stem(i))
         return " ".join(y)
new_df['tags'] = new_df['tags'].apply(stem)
new_df['tags'] = new_df['tags'].apply(lambda x:x.lower())
 new_df.head()
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=5000,stop_words='english')
vectors = cv.fit_transform(new_df['tags']).toarray()
vectors[0]
cv.get_feature_names_out()

from sklearn.metrics.pairwise import cosine_similarity
similarity = cosine_similarity(vectors)
similarity.shape

sorted(list(enumerate(similarity[0])),reverse=True,key = lambda x:x[1])[1:6]

def recommend(movie):
    movie_index = new_df[new_df['title'] == movie].index[0]
    distances=similarity[movie_index]
    movie_list = sorted(list(enumerate(distances)),reverse=True,key = lambda x:x[1])[1:6]
    for i in movie_list:
        print(new_df.iloc[i[0]].title)

recommend("Titanic")

import pickle
pickle.dump(new_df.to_dict(),open('movie_dict.pkl','wb'))
pickle.dump(similarity,open('similarity.pkl','wb'))
```

3.8.2 PYCHARM CODE

```
import streamlit as st
import pickle
import requests

st.title("Movie Recommender system")

movies = pickle.load(open('movie_list.pkl','rb'))
similarity = pickle.load(open('similarity.pkl','rb'))

def fetch_poster(movie_id):
    url = 
"https://api.themoviedb.org/3/movie/{}?api_key=5a3a0b55f9ce06f05082ef2548929741
&language=en-US".format(movie_id)
    data = requests.get(url)
    data = data.json()
    poster_path = data['poster_path']
    full_path = "https://image.tmdb.org/t/p/w500/" + poster_path
    return full_path

def recommend(movie):
    index = movies[movies['title'] == movie].index[0]
    distances=similarity[index]
    movie_list= sorted(list(enumerate(distances)), reverse=True, key=lambda x: 
x[1])[1:6]
    recommended_movie_names = []
    recommended_movie_posters = []
    for i in movie_list:
        movie_id = movies.iloc[i[0]].movie_id
        recommended_movie_names.append(movies.iloc[i[0]].title)
        recommended_movie_posters.append(fetch_poster(movie_id))
    return recommended_movie_names,recommended_movie_posters

movie_list = movies['title'].values
selected_movie = st.selectbox(
    "Type or select a movie from the dropdown",
    movie_list
)
if st.button('Show Recommendation'):
    recommended_movie_names,recommended_movie_posters = 
recommend(selected_movie)
    col1, col2, col3, col4, col5 = st.columns(5)
    with col1:
        st.text(recommended_movie_names[0])
        st.image(recommended_movie_posters[0])
    with col2:
        st.text(recommended_movie_names[1])
        st.image(recommended_movie_posters[1])

    with col3:
        st.text(recommended_movie_names[2])
        st.image(recommended_movie_posters[2])
    with col4:
        st.text(recommended_movie_names[3])
        st.image(recommended_movie_posters[3])
    with col5:
        st.text(recommended_movie_names[4])
        st.image(recommended_movie_posters[4])
```

CHAPTER 4

SCREENSHOTS:

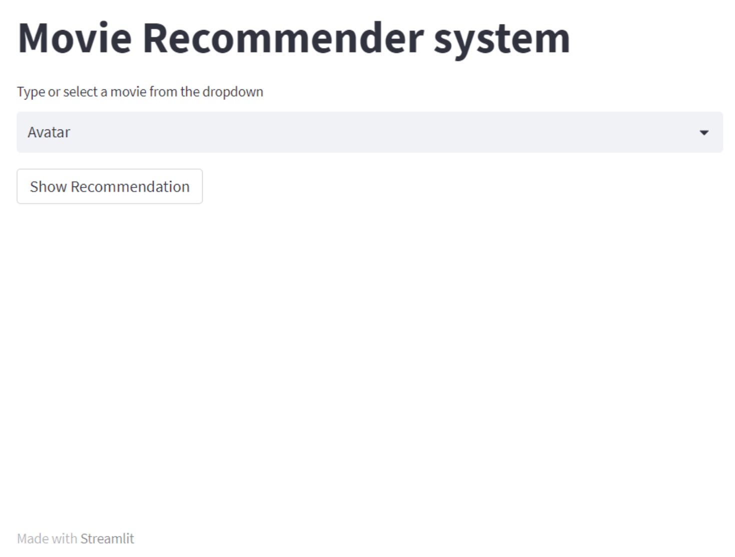
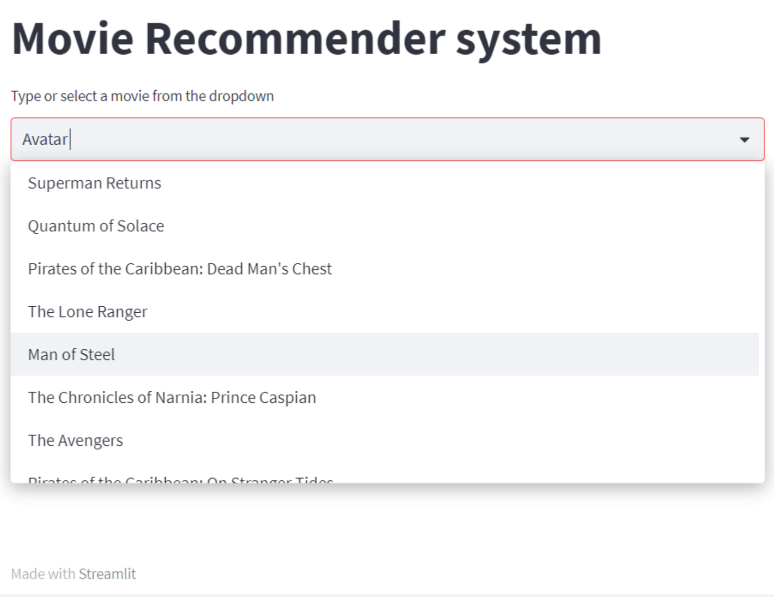
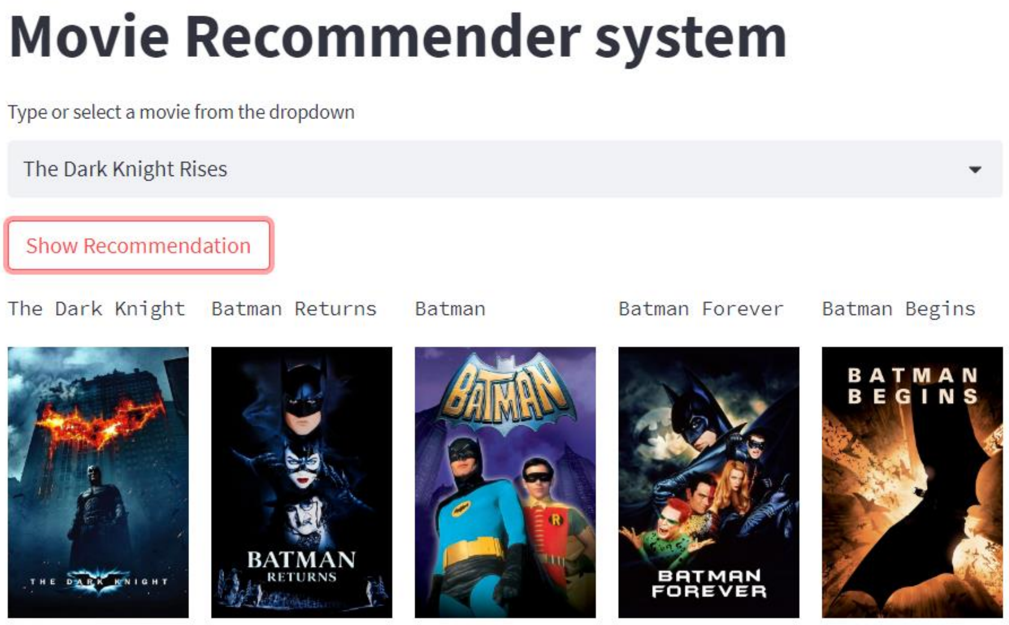


Fig 4.1 HOME RECOMMEND PAGE



Movie Recommender system
Type or select a movie from the dropdown
Avatar
Superman Returns
Quantum of Solace
Pirates of the Caribbean: Dead Man's Chest
The Lone Ranger
Man of Steel
The Chronicles of Narnia: Prince Caspian
The Avengers
Made with Streamlit

Fig 4.2 DROP DOWN LIST

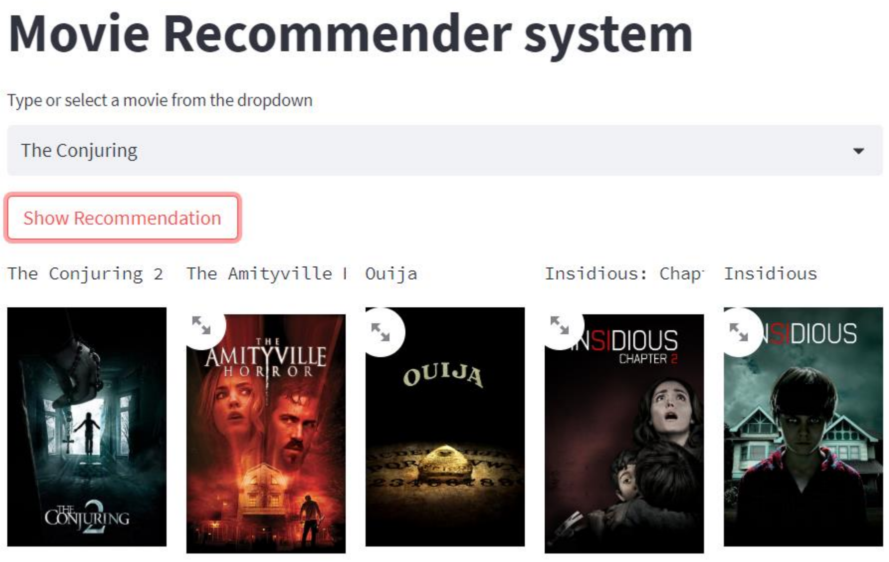


Movie Recommender system

Type or select a movie from the dropdown

The Dark Knight Rises

Fig 4.3 MOVIE 1



Movie Recommender system

Type or select a movie from the dropdown

The Conjuring

Fig 4.4 MOVIE 2

CHAPTER 5

Advantages of “RECOMMENDER SYSTEM”

Right here are a couple of things a recommendation system can do for your business.

- **Drive Traffic**

A recommendation engine can bring traffic to your site. It accomplishes this with customized e-mail messages and targeted blasts.

- **Provide Relevant Material**

By analyzing the customer’s present site use and his previous browsing history, a recommendation engine can deliver appropriate product suggestions as he stores. The data is gathered in real-time so the software can respond as his shopping habits change.

- **Engage Customers**

Consumers end up being more engaged in the website when individualized item recommendations are made. They are able to dive even more deeply into the product line without needing to carry out search after search.

- **Transform Shoppers to Clients**

Converting buyers into consumers takes an unique touch. Individualized communications from a recommendation engine reveal your customer that he is valued as an individual. In turn, this engenders his loyalty.

- **Increase Average Order Value**

Average order values generally go up when a recommendation engine in uses to show tailored alternatives. Advanced metrics and reporting can definitively reveal the efficiency of a project.

- **Boost Number of Items per Order**

In addition to the average order value rising, the number of products per order likewise typically increases when a recommendation engine is used. When the customer is revealed options that fulfill his interest, he is most likely to add choices to his purchase.

- **Control Retailing and Inventory Rules**

A recommendation engine can add your very own marketing and inventory control directives to the customer’s profile to feature items that are promotionally prices, on clearance or overstocked. It offers you’re the versatility to regulate exactly what items are highlighted by the recommendation system.

- **Lower Work and Overhead**

The volume of data required to create an individual shopping experience for each customer is typically far too huge to be handled manually. Utilizing an engine automates this process, reducing the workload of your IT staff and your spending plan.

- **Provide Reports**

Providing guides is an integral part of a personalization system. Providing the client precise and up to the minute reporting permits him to make solid choices about his website and the direction of a project.

- **Offer Recommendations and Direction**

An experienced carrier can provide suggestions on ways to utilize the data gathered and reported to the customer. Acting as a partner and a consultant, the supplier needs to have the expertise to assist direct the ecommerce site to a prosperous future.

CHAPTER 6

Limitations

Besides the above achievements and the successful completion of the project, we still feel the project has some limitations, listed as below:

- To have a system that is user friendly and easy to understand and use.
- To create a data set that has all relevant information about a particular movie
- The biggest challenge was to have the most appropriate movie recommended list.
- To make our system diversifiable so that it can satisfy users of different geographical locations.
- To give weights to different attributes.

Chapter 7
TESTING

Software Testing

Why Software Testing is Needed Testing represents an interesting anomaly for the software engineer. The engineer creates a series of test cases. In fact,testing is the one step in the software process that could be viewed as destructive rather than constructive. Testing requires that the developer discard preconceived notions of the "correctness" of software just developed and overcome a conflict of interest that occurs when errors are uncovered If testing is conducted successfully, it will uncover errors in the software. As a secondary benefit, testing demonstrates that software functions appear to be working according to specification, that behavioral and performance requirements appear to have been met. In addition, data collected as testing is conducted provide a good indication of software reliability and some indication of software quality as a whole. But testing cannot show the absence of errors and defects, it can show only that software errors and defects are present. It is important to keep this (statement in mind as testing is being conducted.

Testing Strategy

There are types of testing that we implement. They are as follows: While deciding on the focus of testing activities, study project priorities. For example, for an on line system, pay more attention to response time. Spend more time on the features used frequently. Decide on the effort required for testing based on the usage of the system. If the system is to be used by alarge number of users, evaluate the impact on users due to a system failure before deciding on the effort.

CHAPTER 8

Future Scope

Recommender system has developed for many years, whichever entered a low point. In the past few years, the development of machine learning, large-scale network and high-performance computing is promoting new development in this field. We will consider the following aspects in future work.

- Use collaborative filtering recommendation. After getting enough user data, collaborative filtering recommendation will be introduced. As we discussed in Section 2.2, collaborative filtering is based on the social information of users, which will be analyzed in the future research.[37]

- Introduce more precise and proper features of movie.[1] Typical collaborative filtering recommendation use the rating instead of object features. In the future we should extract features such as color and subtitle from movie which can provide a more accurate description for movie

- Introduce user dislike movie list. The user data is always useful in recommender systems. In the future we will collect more user data and add user dislike movie list. We will input dislike movie list into the recommender system as well and generate scores that will be added to previous result. By this way we can improve the result of recommender system.

- Introduce machine learning. For future study, dynamic parameters will be introduced into recommender system, we will use machine learning to adjust the weight of each feature automatically and find the most suitable weights.

- Make the recommender system as an internal service. In the future, the recommender system is no longer a external website that will be just for testing. We will make it as an internal APIs for developers to invoke. Some movie lists in the website will be sorted by recommendation.

CHAPTER 9

CONCLUSION

Due to high number of implementations, recommendation systems are gaining popularity today. All The details accessible on the internet cannot be handled by users. For content providers, therefore it is important to filter and customize the content displayed to each user. Showing the correct suggestions lead to better user experience and higher sales in turn. As per the algorithms discussed in this report, according to various scenarios, a recommendation system may have a combination of algorithms. So, if we have metadata about the movies, i.e. genre, artists, production house, then we can recommend films to the consumer on that basis. User-User filtering is affected by the cold start problem. The device needs to wait until certain orders are made by the consumer and priced. Only then can you identify similar users and make suggestions. However, by seeking similarities between items and recommending related items to users, Item-Item filtering overcomes this problem.

But the more you do it, the better you become. And also, I got a new experience on working over machine learning algorithms, PyCharm, various libraries of python, cleaning of data and many more. Thank you so much for reading my report your feedback is precious to us and we are waiting for you feedbacks if any changes are required let us know. Thank you.

CHAPTER 10

BIBLIOGRAPHY

- **https://towardsdatascience.com/recommender-system-a1e4595fc0f0#:~:text=A%20Recommender%20System%20refers%20to%20a%20system%20that%20is%20capable,to%20the%20prevalence%20of%20Internet.**
- **https://www.naukri.com/learning/articles/movie-recommendation-system-using-machine-learning/#_Toc93278933**
- **https://www.mygreatlearning.com/blog/masterclass-on-movie-recommendation-system/**
- **http://203.201.63.46:8080/jspui/bitstream/123456789/6218/1/PR3215%20-%20movie_recommendation_system-report%20-%20PAVAN%20KUMAR%20P%20B.pdf**
- **https://labelyourdata.com/articles/movie-recommendation-with-machine-learning#:~:text=A%20movie%20recommendation%20system%2C%20or,their%20past%20choices%20and%20behavior.**
- **https://towardsdatascience.com/how-to-build-a-movie-recommendation-system-67e321339109**
- **https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie-recommendation-system/**
- **http://www.diva-portal.org/smash/get/diva2:935353/FULLTEXT02.pdf**