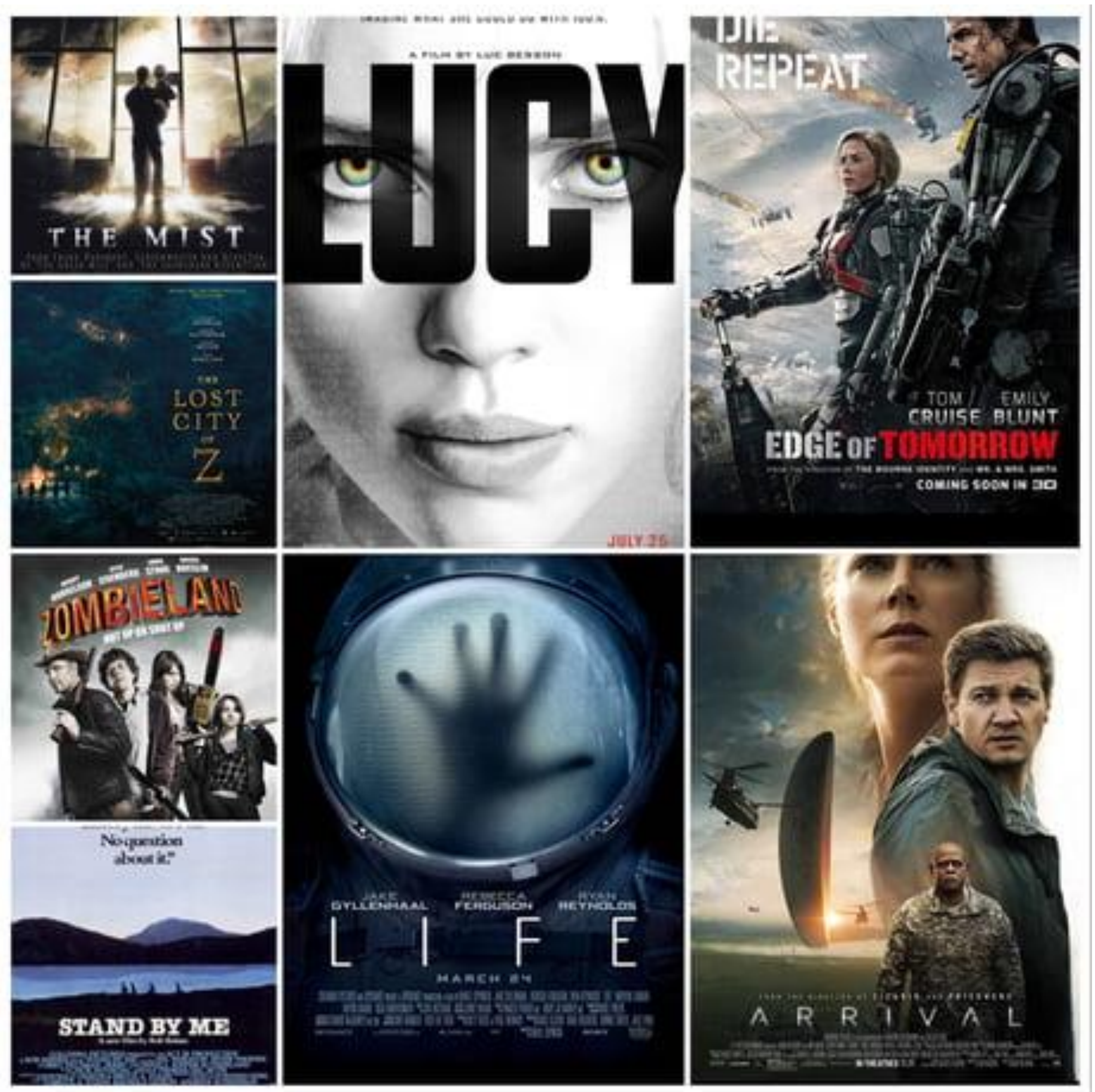
Movie Recommendation System



UEMK, 8TH SEMESTER, FINAL YEAR PROJECT BY GROUP – 47.

MOVIE RECOMMENDATION SYSTEM

Project report in partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology

In

COMPUTER SCIENCE AND ENGINEERING

Submitted By

DEBANJAN BHATTACHARYA	**UNIVERSITY ROLL NO. 12018009019006**
SOUMYADEEP MAJI	**UNIVERSITY ROLL NO. 12018009019307**
SHOURYADEEP MAHATO	**UNIVERSITY ROLL NO. 12018009019576**
ANIKET ROY	**UNIVERSITY ROLL NO. 12018009019188**
ARKADYUTI DAM	**UNIVERSITY ROLL NO. 12018009019654**
SOUMYAJIT ROY	**UNIVERSITY ROLL NO. 12018009019386**

Under the guidance of

Prof. Varsha Poddar

&

Prof. Dr. Sukalyan Goswami

Department of Computer Science

UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA

University Area, Plot No. III – B/5, New Town, Action Area – III, Kolkata – 700160.

UNIVERSITY
OF ENGINEERING & MANAGEMENT
'University Area', Plot No. III-B/5, Main Arterial Road, New Town, Action Area-III, Kolkata - 700 160, W.B., India
City Office : 'ASHRAM', GN-34/2, Salt Lake Electronics Complex, Kolkata - 700 091, W.B., India
(Established by Act XXV of 2014 of Govt. of West Bengal & recognised by UGC, Ministry of HRD, Govt. of India)

Ph. (Office) : 91 33 2357 7649
: 91 33 2357 2969
: 91 33 6888 8608
Admissions : 91 33 2357 2059
Fax : 91 33 2357 8302
E-mail : vc@uem.edu.in
Website : www.uem.edu.in

CERTIFICATE

This is to certify that the project titled **Movie Recommendation System** submitted by **Debanjan Bhattacharya (University Roll No. 12018009019006), Shouryadeep Mahato (University Roll no. 12018009019576), Soumyadeep Maji (University Roll No. 12018009019307), Aniket Roy (University Roll No. 12018009019188), Arkadyuti Dam (University Roll No. 12018009019654)**, and **Soumyajit Roy (University Roll No. 12018009019386)**, students of **UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA**, in partial fulfillment of requirement for the degree of Bachelor of Computer Science Engineering, is a Bonafede work carried out by them under the supervision and guidance of **Prof. Varsha Poddar & Prof. Dr. Sukalyan Goswami** during 8th Semester of academic session of 2018-2022. The content of this report has not been submitted to any other university or institute. I am glad to inform that the work is entirely original and its performance is found to be quite satisfactory.

____

Signature of Guide

____

Signature of Guide

____

Signature of Head of the Department

Other institutes of the Group
University of Engineering & Management (UEM), Jaipur - 6 Km. from Chomu on Sikar Road (NH-11), Udaipuria Mod. Jaipur - 303807, Rajasthan
Institute of Engineering & Management (IEM) - Salt Lake Electronics Complex, Sector - V, Kolkata - 700 091, West Bengal
New York Public School - GE, 4/A, Sector - III, Salt Lake, Kolkata - 700 106, West Bengal (Near Tank No. - 12, Behind NIFT Girls' Hostel)

ACKNOWLEDGEMENT

We would like to take this opportunity to thank everyone whose cooperation and encouragement throughout the ongoing course of this project remains invaluable to us.

We are sincerely grateful to our guide Prof. Varsha Poddar and Prof. Dr. Sukalyan Goswami of the Department of Computer Science, UEM, Kolkata, for his wisdom, guidance and inspiration that helped us to go through with this project and take it to where it stands now.

Last but not the least, we would like to extend our warm regards to our families and peers who have kept supporting us and always had faith in our work.

Debanjan Bhattacharya

Soumyadeep Maji

Shouryadeep Mahato

Aniket Roy

Arkadyuti Dam

Soumyajit Roy

TABLE OF CONTENTS

ABSTRACT

The purpose of this project is to build a ***Movie Recommendation System*** *that suggests users various movies and similar contents depending on the type of recommendation system in play.*

To define a recommendation system, it is a kind of system that provides suggestions to users for certain resources like books, movies, songs, etc., based on some particular data set. Movie recommendation systems usually predict what movies a user will like based on the attributes present in previously liked movies. Such recommendation systems are beneficial for organizations that collect data from large amounts of customers, and wish to effectively provide the best suggestions possible. A lot of factors can be considered while designing a movie recommendation system like the genre of the movie, actors present in it or even the director of the movie. The systems can recommend movies based on one or a combination of two or more attributes.

In this paper, the recommendation system has been built on two types of recommenders: (i) Simple Recommender and (ii) Content-based Recommender. Two different approaches are used to achieve the results for both types of recommendation system. The dataset used for the system is downloaded from a free-source database in .csv format and is named as ***"tmdb_5000_movies.csv"*** *in our project.*

INTRODUCTION

1. What is a recommendation system?

A recommender system, or a recommendation system, is a subclass of the information filtering system that seeks to predict the "rating" or "preference" a user would give to an item.

Recommender systems are used in a variety of areas, with commonly recognized examples taking the form of playlist generators for video and music services, product recommenders for online stores, or content recommenders for social media platforms and open web content recommenders. These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. Recommender systems have also been developed to explore research articles and experts, collaborators, and financial services.

2. What is a movie recommendation system?

A movie recommendation system is a domain-specific recommendation program that seeks to predict the "ratings" or "preferences" of a user towards a particular movie or a set of movies. Therefore, the main focus of our recommendation system is to filter and predict only those movies which a user would prefer given some data about the user him or herself.

3. Types of movie recommender systems being used.

a.) ***Simple Recommender*** *– The Simple Recommender offers generalized recommendations to every user based on movie popularity and (sometimes) genre. The basic idea behind this recommender is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the average audience. We will sort our movies based on ratings and popularity and display the top movies of our list. As an added step, we can pass in a genre argument to get the top movies of a particular genre.*

Methodology for Simple Recommender System: -

We will use the TMDB Ratings to come up with our Top Movies Chart. We will use IMDB's weighted rating formula to construct our chart. Mathematically, it is represented as follows:

Weighted Rating (WR) = (vv + $m.R$) + (mv + $m.C$)(vv + m.R) + (mv + m.C),

where,

- **v** is the number of votes for the movie
- **m** is the minimum votes required to be listed in the chart
- **R** is the average rating of the movie
- **C** is the mean vote across the whole report

The next step is to determine an appropriate value for m, the minimum votes required to be listed in the chart. We will use **95th percentile** as our cutoff. In other words, for a movie to feature in the charts, it must have more votes than at least 95% of the movies in the list.

b.) **Content Based Recommender** – The recommender we built in the previous section suffers some severe limitations. For one, it gives the same recommendation to everyone, regardless of the user's personal taste. If a person who loves romantic movies (and hates action) were to look at our Top 15 Chart, she/he wouldn't probably like most of the movies. If she/he were to go one step further and look at our charts by genre, she/he wouldn't still be getting the best recommendations.

For instance, consider a person who loves Dilwale Dulhania Le Jayenge, My Name is Khan and Kabhi Khushi Kabhi Gham. One inference we can obtain is that the person loves the actor Shahrukh Khan and the director Karan Johar. Even if she/he were to access the romance chart, she/he wouldn't find these as the top recommendations.

To personalize our recommendations more, we are going to build an engine that computes similarity between movies based on certain metrics and suggests movies that are most similar to a particular movie that a user liked. Since we will be using movie metadata (or content) to build this engine, this also known as Content Based Filtering.

We will build two types of Content Based Recommenders based on:

- Movie Overviews and Taglines
- Movie Cast, Crew, Keywords and Genre

Methodology for Content Based Recommender:

Cosine Similarity – We will be using the Cosine Similarity to calculate a numeric quantity that denotes the similarity between two movies. Mathematically, it is defined as follows:

$$cosine(x,y) = x.y^{T}/|x||.||y||$$

Since we have used the TF-IDF Vectorizer, calculating the Dot Product will directly give us the Cosine Similarity Score. Therefore, we will use sklearn's linear kernel instead of cosine similarities since it is much faster.

PROBLEM STATEMENT

Project Title **-** *To build a movie recommendation system that filters movies based on specific types of data taken from the user. In our project, we have priorities specific data fields like ratings, votes, movie descriptions, genre, overviews and taglines, movie cast, crew, and certain keywords to extract accurate information and then display the movies accordingly.*

Project Requirements **-** *The main language to be used in this project is Python and the concept of data processing is being used to get the desired result in the form of movie recommendations.*

Modules used in the project *–*

1. ***Backend part*** **–** *Pandas, NumPy, Literal_eval, and Sklearn (TfidfVectorizer, CountVectorizer, linear_kernel, cosine_similarity).*
2. ***Front-end part*** **–** *Streamlit, Pickle, and Requests.*

PROPOSED SOLUTION

A.) Backend Solution

1. *Types of recommenders:*

i. ***Simple Recommender***
ii. ***Content-based Recommender***

2. *About the dataset:*

The dataset consists of movies released on or before July 2017. Data points include cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages.

This dataset also has files containing 26 million ratings from 270,000 users for all 45,000 movies. Ratings are on a scale of 1-5 and have been obtained from the official GroupLens website.

- **The Full Dataset**: Consists of 26,000,000 ratings and 750,000 tag applications applied to 45,000 movies by 270,000 users. Includes tag genome data with 12 million relevance scores across 1,100 tags.
- **The Small Dataset:** Comprises of 100,000 ratings and 1,300 tag applications applied to 9,000 movies by 700 users.

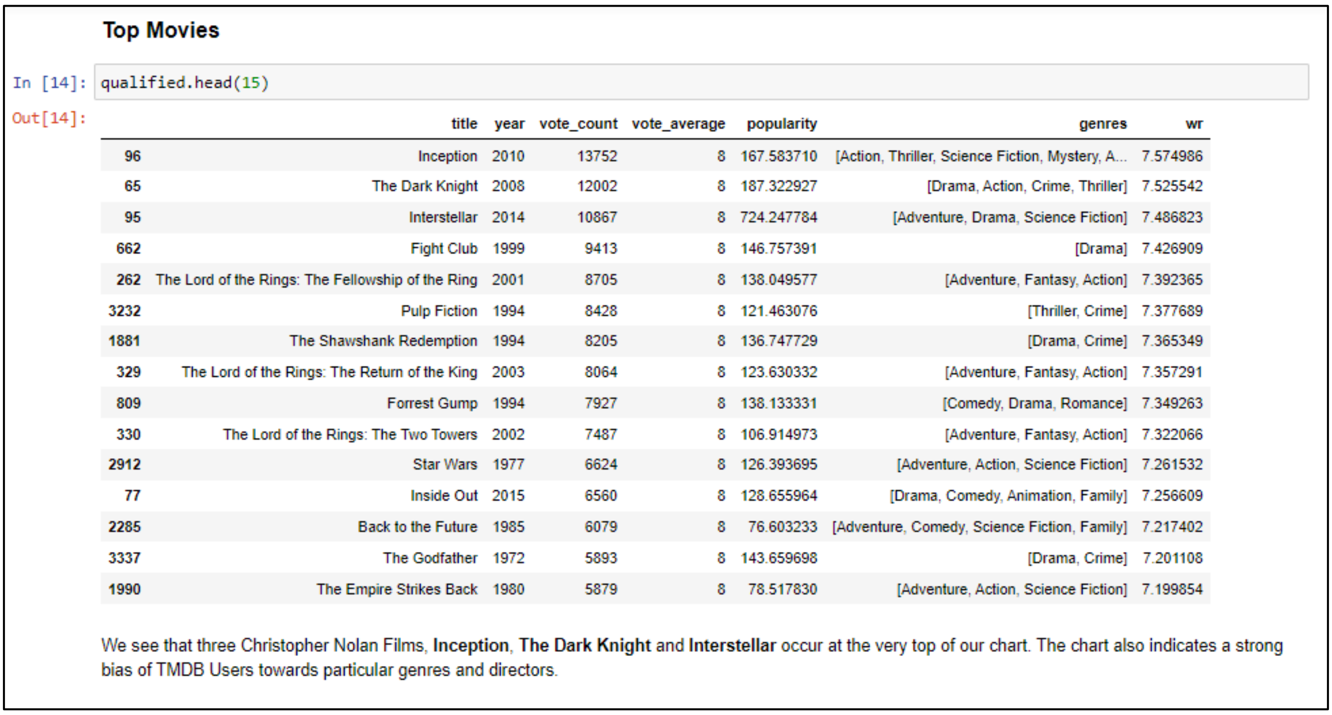
We will build our Simple Recommender using movies from the Full Dataset whereas all personalized recommender systems will make use of the small dataset.

3. *Modules used:*

The major modules that are used in this part of the solution are as follows:

I. ***Pandas*** – a python library used for data processing.
II. ***NumPy*** – a python library used for working with arrays. It also has functions that can be used in other domains such as linear algebra, Fourier transform, and matrices.
III. ***ast*** – a python module that helps python application process tress of the python abstract syntax grammar.
IV. ***Sklearn*** – stands for Scikit-learn is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.
V. ***Warnings -*** Base category for warnings triggered during the process of importing a module (ignored by default).

4. Simple Recommender:



Top Movies

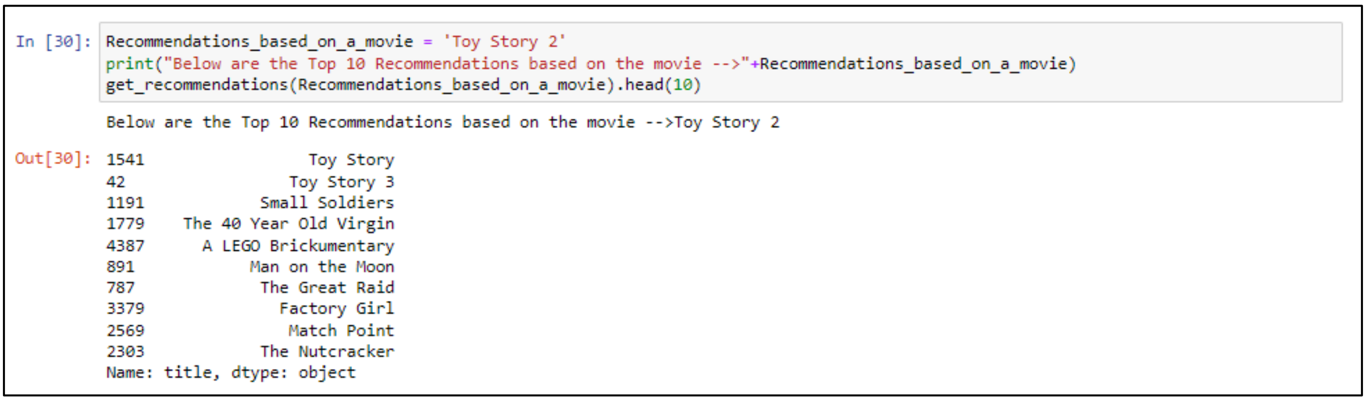
```
In [14]: qualified.head(15)
```

Out[14]:

	title	year	vote_count	vote_average	popularity	genres	wr
96	Inception	2010	13752	8	167.583710	[Action, Thriller, Science Fiction, Mystery, A...	7.574986
65	The Dark Knight	2008	12002	8	187.322927	[Drama, Action, Crime, Thriller]	7.525542
95	Interstellar	2014	10867	8	724.247784	[Adventure, Drama, Science Fiction]	7.486823
662	Fight Club	1999	9413	8	146.757391	[Drama]	7.426909
262	The Lord of the Rings: The Fellowship of the Ring	2001	8705	8	138.049577	[Adventure, Fantasy, Action]	7.392365
3232	Pulp Fiction	1994	8428	8	121.463076	[Thriller, Crime]	7.377689
1881	The Shawshank Redemption	1994	8205	8	136.747729	[Drama, Crime]	7.365349
329	The Lord of the Rings: The Return of the King	2003	8064	8	123.630332	[Adventure, Fantasy, Action]	7.357291
809	Forrest Gump	1994	7927	8	138.133331	[Comedy, Drama, Romance]	7.349263
330	The Lord of the Rings: The Two Towers	2002	7487	8	106.914973	[Adventure, Fantasy, Action]	7.322066
2912	Star Wars	1977	6624	8	126.393695	[Adventure, Action, Science Fiction]	7.261532
77	Inside Out	2015	6560	8	128.655964	[Drama, Comedy, Animation, Family]	7.256609
2285	Back to the Future	1985	6079	8	76.603233	[Adventure, Comedy, Science Fiction, Family]	7.217402
3337	The Godfather	1972	5893	8	143.659698	[Drama, Crime]	7.201108
1990	The Empire Strikes Back	1980	5879	8	78.517830	[Adventure, Action, Science Fiction]	7.199854

We see that three Christopher Nolan Films, **Inception**, **The Dark Knight** and **Interstellar** occur at the very top of our chart. The chart also indicates a strong bias of TMDB Users towards particular genres and directors.

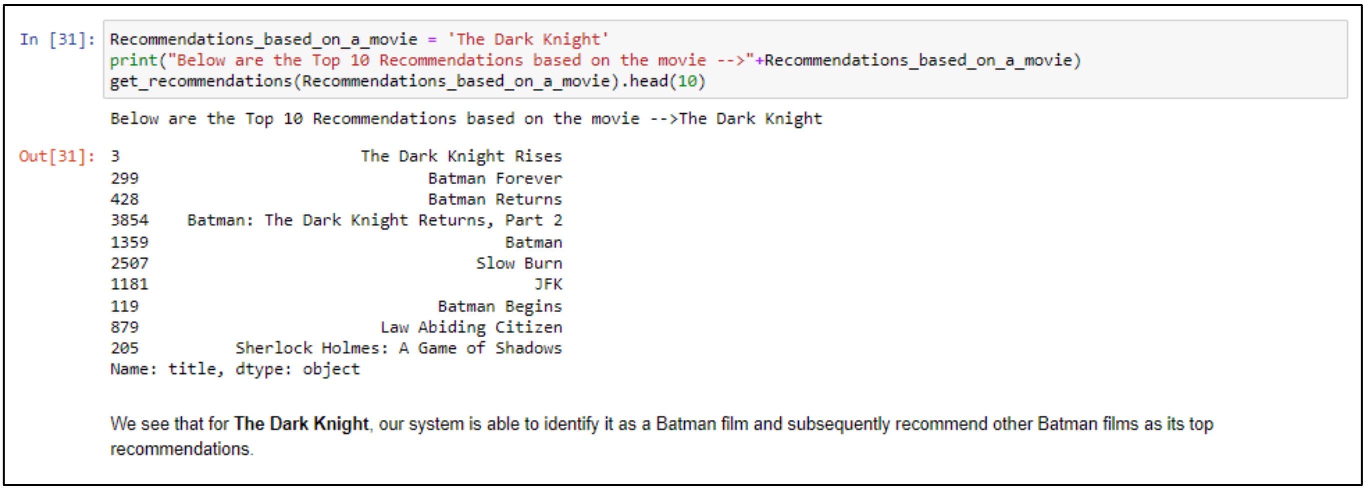
5. Content-based Recommender:



```
In [30]: Recommendations_based_on_a_movie = 'Toy Story 2'
         print("Below are the Top 10 Recommendations based on the movie -->"+Recommendations_based_on_a_movie)
         get_recommendations(Recommendations_based_on_a_movie).head(10)

         Below are the Top 10 Recommendations based on the movie -->Toy Story 2

Out[30]: 1541                Toy Story
         42                Toy Story 3
         1191           Small Soldiers
         1779     The 40 Year Old Virgin
         4387       A LEGO Brickumentary
         891             Man on the Moon
         787              The Great Raid
         3379              Factory Girl
         2569               Match Point
         2303            The Nutcracker
         Name: title, dtype: object
```



```
In [31]: Recommendations_based_on_a_movie = 'The Dark Knight'
         print("Below are the Top 10 Recommendations based on the movie -->"+Recommendations_based_on_a_movie)
         get_recommendations(Recommendations_based_on_a_movie).head(10)

         Below are the Top 10 Recommendations based on the movie -->The Dark Knight

Out[31]: 3                        The Dark Knight Rises
         299                             Batman Forever
         428                             Batman Returns
         3854    Batman: The Dark Knight Returns, Part 2
         1359                                    Batman
         2507                                 Slow Burn
         1181                                       JFK
         119                              Batman Begins
         879                         Law Abiding Citizen
         205           Sherlock Holmes: A Game of Shadows
         Name: title, dtype: object
```

We see that for **The Dark Knight**, our system is able to identify it as a Batman film and subsequently recommend other Batman films as its top recommendations.

B.) Front-end Solution (Website)

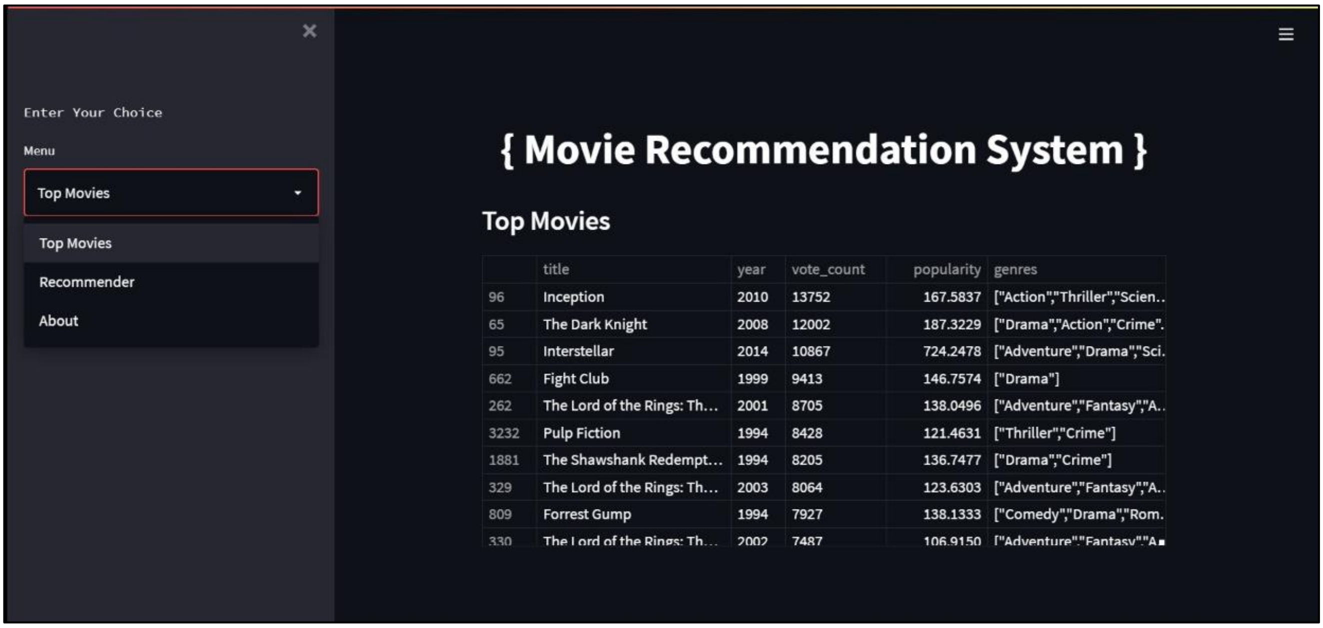
Following are the major modules used to achieved the result:

I. ***Streamlit* –** *Streamlit is an open-source app framework in Python language. It helps us create web apps for data science and machine learning in a short time. It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc.*

II. ***Pickle* –** *Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk.*

III. ***Requests* –** *The requests module allows you to send HTTP requests using Python. The HTTP request returns a Response Object with all the response data (content, encoding, status, etc.).*

1. Simple Recommender displaying the top movies based on overall user ratings.

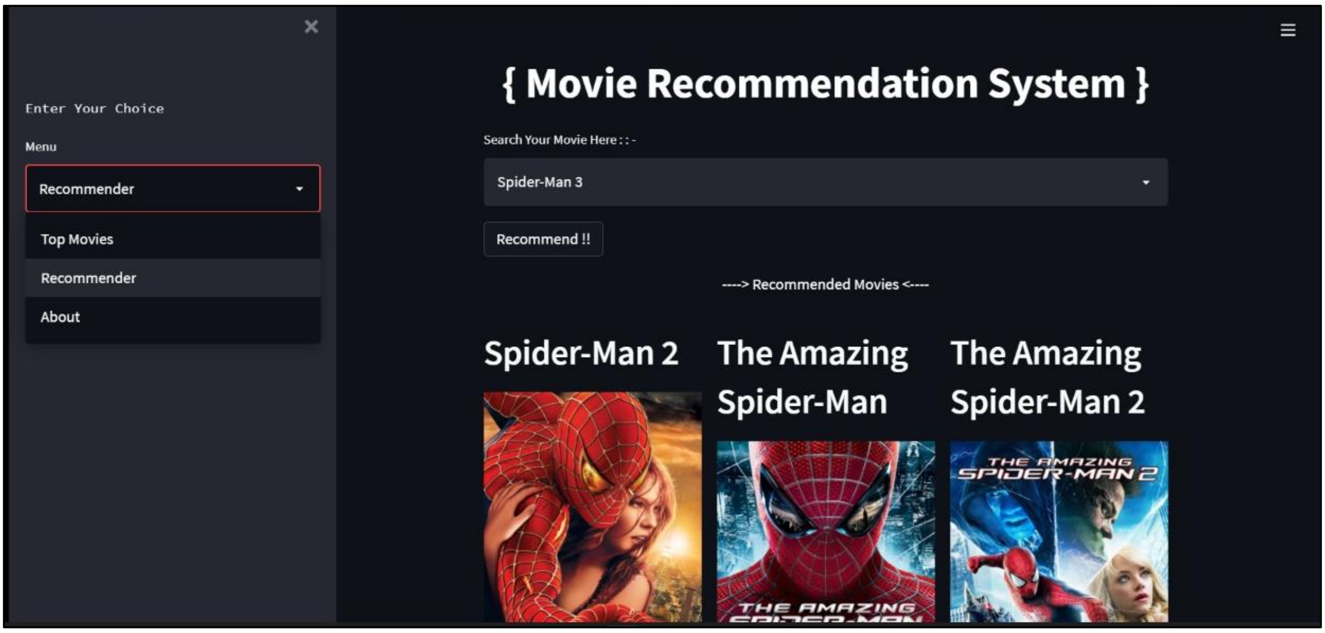


{ Movie Recommendation System }

Top Movies

	title	year	vote_count	popularity	genres
96	Inception	2010	13752	167.5837	["Action","Thriller","Scien..
65	The Dark Knight	2008	12002	187.3229	["Drama","Action","Crime".
95	Interstellar	2014	10867	724.2478	["Adventure","Drama","Sci.
662	Fight Club	1999	9413	146.7574	["Drama"]
262	The Lord of the Rings: Th…	2001	8705	138.0496	["Adventure","Fantasy","A..
3232	Pulp Fiction	1994	8428	121.4631	["Thriller","Crime"]
1881	The Shawshank Redempt…	1994	8205	136.7477	["Drama","Crime"]
329	The Lord of the Rings: Th…	2003	8064	123.6303	["Adventure","Fantasy","A..
809	Forrest Gump	1994	7927	138.1333	["Comedy","Drama","Rom.
330	The Lord of the Rings: Th…	2002	7487	106.9150	["Adventure","Fantasy","A

2. ***<u>Content-based Recommender showing output of the movies similar to the required one based on the similarity score used in the engine.</u>***



{ Movie Recommendation System }

Menu: Recommender

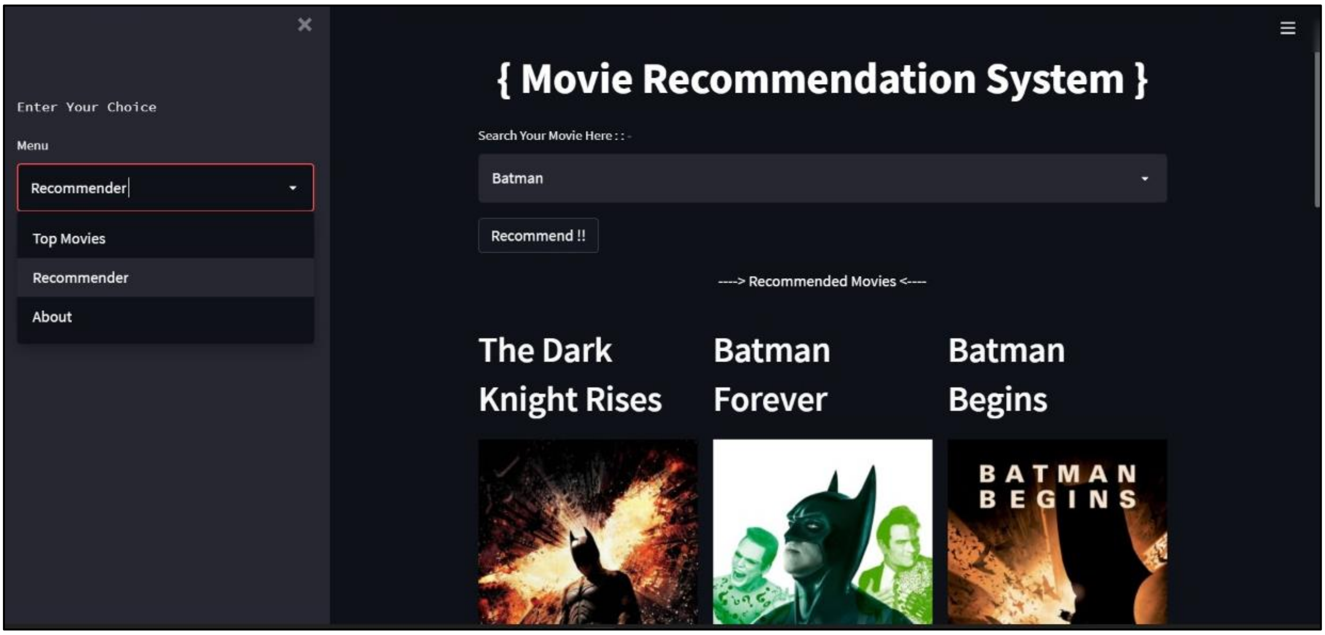
Search Your Movie Here : : - Spider-Man 3

----> Recommended Movies <----

Spider-Man 2

The Amazing Spider-Man

The Amazing Spider-Man 2



{ Movie Recommendation System }

Menu: Recommender

Search Your Movie Here : : - Batman

----> Recommended Movies <----

The Dark Knight Rises

Batman Forever

Batman Begins

CONCLUSION & FUTURE SCOPE

Conclusion:

In this project, we have built 2 different types of recommendation engines based on different ideas and algorithms. They can be summarized as follows:

1. **Simple Recommender**: This system used overall TMDB Vote Count and Vote Averages to build Top Movies Charts, in general and for a specific genre. The IMDB Weighted Rating System was used to calculate ratings on which the sorting was finally performed.
2. **Content Based Recommender**: We built a content-based engines; one that took movie overview and taglines as input and the other which took metadata such as cast, crew, genre and keywords to come up with predictions. We also devised a simple filter to give greater preference to movies with more votes and higher ratings.

Future Scope:

Cosine similarity calculation do not work well when we don't have enough rating for movie or when user's rating for some movie is exceptionally either high or low. As an improvement on this project some other methods such as adjusted cosine similarity can be used to compute similarity.

Adjusted cosine similarity, which is similar to cosine similarity, is measured by normalizing the user vectors Ux and Uy and computing the cosine of the angle between them. However, unlike cosine similarity, when computing the dot product of the two user vectors, adjusted cosine similarity uses the deviation between each of the user’s item ratings, denoted Ru, and their average item rating, denoted ¯Ru, in place of the user’s raw item rating.

In equation form, the adjusted cosine similarity computation is expressed as:

$$sim(u_x, u_y) = \frac{\sum\limits_{i \in P_{u_x,u_y}} (r_{u_x,i} - \bar{r}_{u_x})(r_{u_y,i} - \bar{r}_{u_y})}{\sqrt{\sum\limits_{i \in P_{u_x,u_y}} (r_{u_x,i} - \bar{r}_{u_x})^2}\sqrt{\sum\limits_{i \in P_{u_x,u_y}} (r_{u_y,i} - \bar{r}_{u_y})^2}}$$

where Pux,uy represents the subset of items i ∈ I for which both users have rated, Rux,i is the rating of user Ux on item i and Ruy,i in the rating of user Uy on item i. The main advantage of this approach is that in item-based collaborative filtering, the item vectors consist of ratings from different users who often have varying rating scales.

BIBLIOGRAPHY

1. ***Kaggle.com –*** *Obtained the dataset for your project*
2. ***Streamlit.io –*** *Learning and implementation of Streamlit concept in Python and using the syntaxes to get the desired results.*
3. ***Stackoverflow.com –*** *Concept on how to use HTML codes in a Python program for easier website modification.*
4. ***Ivyproschool.com –*** *Initial idea about the movie recommendation system.*
5. ***Themoviedb.com –*** *Obtaining the API key and display images for the content-based recommender.*