MOVIE RECOMMENDATION SYSTEM

PROJECT REPORT USING CONTENT-BASED METHODOLOGY

BY-

RAHUL KUMAR BISWAS

TABLE OF CONTENTS

CH-01 INTRODUCTION :

A recommendation system is essentially a filtering system that anticipates the users' choices and then suggests the most accurate results based on the users' past preferences. We have a variety of applications for our recommendation system that we have used throughout the years and are actively applying on various internet platforms.

Recommendation systems are becoming increasingly important in today's extremely busy world. People are always short on time with the myriad tasks they need to accomplish in the limited 24 hours. Therefore, the recommendation systems are important as they help them make the right choices, without having to expend their cognitive resources.

The purpose of a recommendation system basically is to search for content that would be interesting to an individual. Moreover, it involves a number of factors to create personalised lists of useful and interesting content specific to each user/individual. Recommendation systems are Artificial Intelligence based algorithms that skim through all possible options and create a customized list of items that are interesting and relevant to an individual.

These results are based on their profile, search/browsing history, what other people with similar traits/demographics are watching, and how likely are you to watch those movies. This is achieved through predictive modeling and heuristics with the data available.

Users get good recommendations all the time, and it keeps improving as we move forward in the twenty-first century, and they produce virtually precise answers.

In the event of a conflict between any e App Music, any music platform, or any educational app, just prohibit using the app. Additionally, firms must focus on their recommendation system, which is more complex than it appears.

Every user has different interests and preferences based on their numerous pursuits and emotions, such as music when playing, travelling, exercising, or after a relationship disagreement, and so on.

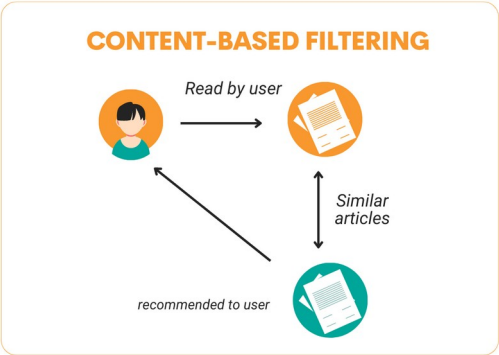
A)OBJECTIVE:

Movie recommendation system provides the mechanism and classifying the users with the same interest and searches for the content that would be so much interesting belonging to different set of users and then creating different kind of lists and providing interesting recommendations to the individual based on the content the love.

The main objective of the recommender system is to used approaches suggest demographic filtering, content-based filtering, collaborative filtering to find the set of movies with every user likes for specific set of users. The movies that have high probability of being liked by the general set of users will be displayed to the user by the recommender in the end and then in another technique we will try to find the users with different interest using the information collected through different activities an Indian in collaborative filtering will test all those users which have same type of interests to get the final set of movies to be recommended to the users individually. So, we will use different categories of recommender filtering techniques and then compare in contrast that results obtained in different methods and will try to improve the results as h dataset for set of movies goes larger and larger above the computational bound of the system which is generally a limitation on the large dataset.

B)METHODOLOGY:

CONTENT BASED FILTERING SYSTEM



We compare the different things with the user's interest profile in the content-based filtering technique. So, fundamentally, the user profile contains material that is much more relevant to using the form of the features.

Previous actions or feedback is often taken into consideration, as is the description of the information that has been modified by users of various selections.

Consider the following scenario: A person buys a favorite item 'M,' but the item is sold out; as a consequence, he must buy the item 'N,' on the suggestion of someone else, because 'N' has the same sort of matching features as the first one. And that is basically the content-based filtration that is shown below.

So, in this case, the numeric number that will be utilized to determine the similarities between the two types of movies will be cosine similarity, and we will get the score extremely quickly.

$$\text{Similarity}(p,q) = \cos\theta = \frac{p \cdot q}{\|p\|\|q\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} p_i^{\,2}}\sqrt{\sum_{i=1}^{n} q_i^{\,2}}}$$

The following are the stages required in obtaining a movie recommendation:

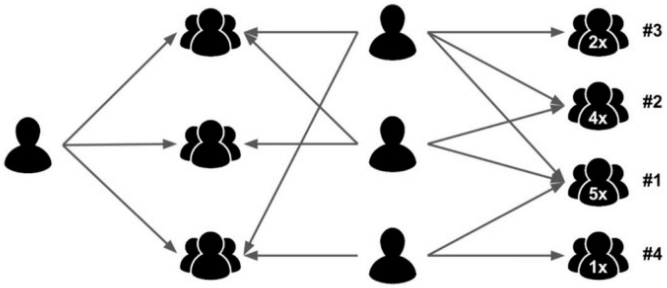
- With the title, you may locate the index of the movie.
- Computing the cosine similarity scores for each film
- Arrange the scores in ascending order, starting with the greatest priority.
- The group is then pruned based on the similarity scores.

- Obtaining the first ten elements of the list, with the exception of the first, which is the movie title in and of itself.
- Obtaining the most important ingredients.

The following are the benefits of content-based filtering:

- We could suggest the unrated items.
- Can suggest movies depending on the user's ratings
- It is not possible to make the user like with un-likes.

Collaborative based Filtering:



Content-based filtering has some limitations, including the ability to propose movies based on only one sort of user choice and the inability to make genre suggestions.

However, a collaborative filtering-based method adds a layer of complexity to establishing a match between a user's similarities and the likes of other users with similar interests.

All of the collaborative filtering in the case of user-based is easy, but it has limitations, the most significant of which is that even the choices of the user where is with time.

So, we may utilize item-based collaborative filtering, which simply examines the things based on their similarity with the items and that it finds comparable matches with the target users using the same

similarity coefficients, such as Pearson's correlations or Cosines similarity. Item-based collaborative filtering is the most static.

To simplify computations, we simply are using the utility matrix by assigning zero values to the discrete columns. According to an item in general, collaborative filtering is preferred since it evaluates the film instead of the quantity of people, making it easier to classify movies and users.

The following are the benefits of collaborative filtering-based systems:

- It is just depending on the content.
- It frequently reads the minds of people who share similar inclinations.
- Make a true quality appraisal of the things

The following are the disadvantages of collaborative filtering-based systems:

- The most prevalent early rater problem occurs when the collaborative filtering approach fails to offer ratings for a video with no user waiting.
- Sparsity is more prevalent in this type of welding procedure since there are so many null values that it is difficult to discover objects that are assessed by the majority of people.

LITERATURE SURVEY:

Movie recommendation systems have become a fundamental component of the modern entertainment industry, helping users discover films that align with their preferences and tastes.

Among the various approaches to recommendation, content-based filtering stands out as a technique that relies on the intrinsic characteristics of movies and user profiles. In this literature survey, we explore the advancements, challenges, and trends in the field of content-based movie recommendation systems

1. Introduction to Content-Based Filtering

Content-based filtering, also known as content-based recommendation, recommends items based on their content attributes. In the context of movie recommendation, this approach relies on the analysis of movie features such as genre, plot, cast, crew, and more. By matching these features to user preferences, content-based systems aim to provide personalized recommendations.

2. Feature Extraction and Representation

TF-IDF and Text Analysis: One common method of extracting movie content features is using term frequency-inverse document frequency (TF-IDF) for textual data, allowing for the representation of movie plots and descriptions in a

numerical format. Techniques like natural language processing (NLP) are used to process and analyze textual content.

Metadata and Collaborative Filtering: Some content-based systems also consider metadata like genres, release years, and even combine collaborative filtering techniques to improve recommendation accuracy.

3. Recommendation Algorithms

Cosine Similarity: Cosine similarity is a prevalent method for calculating the similarity between movies based on their content attributes. It measures the cosine of the angle between two vectors, where each vector represents a movie's content features. The greater the cosine value, the more similar the movies are considered to be.

Machine Learning Models: More advanced content-based recommendation systems utilize machine learning models, such as support vector machines, decision trees, or deep learning architectures to predict user preferences based on content features.

4. Challenges and Limitations

Cold Start Problem: Content-based filtering can face challenges when dealing with new users or movies for which there is limited content data. These are known as the "cold start" problems.

Limited Serendipity: Content-based systems might struggle to recommend diverse content outside a user's established preferences, potentially leading to filter bubbles.

Scalability: Handling a vast amount of content data, especially in large-scale systems, can be computationally intensive.

5. Hybrid Recommendation Systems

To address some of the limitations of content-based filtering, many modern recommendation systems employ hybrid models that combine content-based and collaborative filtering techniques. Hybrid systems aim to leverage the strengths of both approaches to provide more accurate and diverse recommendations.

6. Recent Trends and Future Directions

Deep Learning: With the rise of deep learning, there is growing interest in using neural networks to model complex relationships between users, movies, and content features, thereby improving the accuracy of content-based recommendations.

Explainability: There is an increasing emphasis on making recommendation systems more transparent and interpretable by providing users with explanations for the recommendations made.

Multi-modal Content: Incorporating a broader range of content data, including images and audio, is an emerging trend to provide richer movie representations for recommendation.

CH-O3 : SYSTEM DESIGN & DEVELOPMENT:

Dataset:-

For content-based filtration:

The dataset has been taken from Kaggle. It is used as the standard dataset by the movie recommendation system.

1) Movie dataset is used in “content-based movie recommendation system” in this project.

2) The ratings and movies are taken into account.

3) Total number of movies.

4) Total number of ratings in a dataset.

5) An uniquid is assigned to every movie and the user.

This chapter involves both the hardware and software requirements needed for the project and detailed explanation of the specifications:-

Hardware Requirements:

- A PC with Windows/Linux OS

- Processor with 1.7-2.4gHz speed
- Minimum of 8gb RAM
- 2gb Graphic card

Software Specification

- Text Editor (VS-code)
- Anaconda distribution package (PyCharm Editor)
- Python libraries

Software Requirements:

- Anaconda distribution:

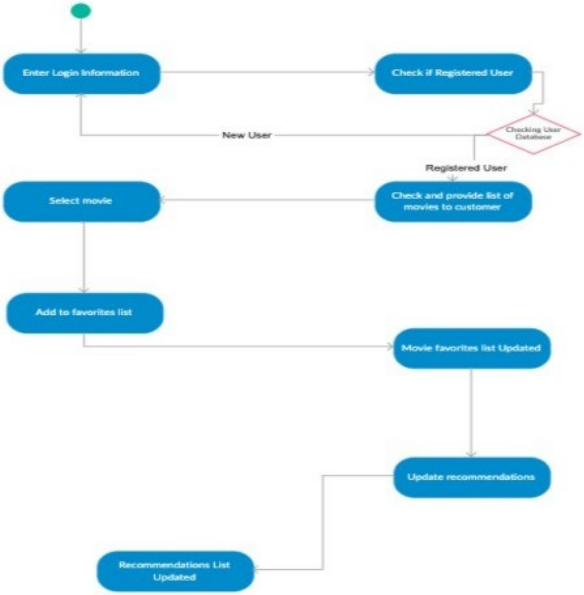
Anaconda is a free and open-source distribution of the Python programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management system and deployment. Package versions

are managed by the package management system conda. The anaconda distribution includes data-science packages suitable for Windows, Linux and MacOS.3

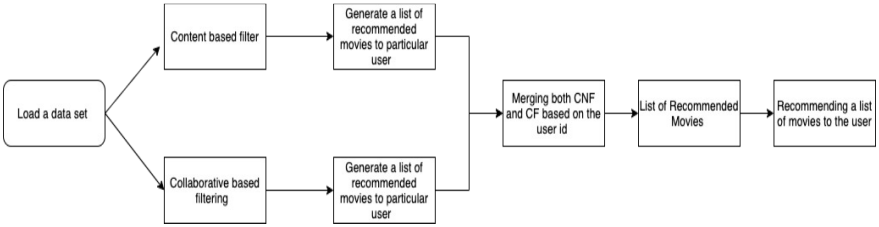
- Python libraries:

For the computation and analysis we need certain python libraries which are used to perform analytics. Packages such as SKlearn, Numpy, pandas, Matplotlib, Flask framework, etc are needed.

ACTIVITY DIAGRAM:



DATA FLOW DIAGRAM:



Initially load the data sets that are required to build a model the data set that are required in this project are movies.csv, ratinfg.csv, users.csv all the data sets are available in the Kaggle.com. Combining both based on the useid a single final list of movies are recommended to the particular user

CH-04 PERFORMANCE ANALYSIS:

Content Based Filtering User profile holds the content that is much more matching to use the form of the features. The previous actions or for the feedback is taken into account a generally takes into account the description of the content that has been edited by the users of different choices.

```
In [3]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os

In [4]:
movies = pd.read_csv('C:\\Users\\KIIT\\OneDrive\\Desktop\\Python\\tmdb_5000_movies.csv')
credits = pd.read_csv('C:\\Users\\KIIT\\OneDrive\\Desktop\\Python\\tmdb_5000_credits.csv')

In [5]:
movies.head(2)
```

Out[5]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_cor
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "Ir Film Partne
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"name": "Wa Pictures", "id

```
In [48]:
new_df = movies[['movie_id','title','tags']]
new_df.head()
```

Out[48]:

	movie_id	title	tags
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...
2	206647	Spectre	[A, cryptic, message, from, Bond's, past, send...
3	49026	The Dark Knight Rises	[Following, the, death, of, District, Attorney...
4	49529	John Carter	[John, Carter, is, a, war-weary,, former, mili...

In [51]:
```
movies['tags'] = movies['genres'] + movies['keywords'] + movies['cast'] + movies['crew']
```

In [52]:
```
movies.head(2)
```

Out[52]:

	movie_id	title	overview	genres	keywords	cast	crew	tags
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...	[Action, Adventure, Fantasy, ScienceFiction]	[culturecIash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver, ...	[JamesCameron]	[Action, Adventure, Fantasy, ScienceFiction, c...
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...	[Adventure, Fantasy, Action]	[ocean, drugabuse, exoticisland, eastindiatrad...	[JohnnyDepp, OrlandoBloom, KeiraKnightley, Ste...	[GoreVerbinski]	[Adventure, Fantasy, Action, ocean, drugabuse,...

In [54]:
```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=5000,stop_words='english')
```

In [55]:
```
vector = cv.fit_transform(new_df['tags']).toarray()
```

In [56]:
```
vector
```

Out[56]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [57]:
```
vector.shape
```

Out[57]:
```
(4806, 5000)
```

In [58]:
```
from sklearn.metrics.pairwise import cosine_similarity
```

In [59]:
```
similarity = cosine_similarity(vector)
similarity.shape
```

Out[59]:
```
(4806, 4806)
```

In [60]:
```
similarity
```

Out[60]:
```
array([[1.        , 0.07142857, 0.05216405, ..., 0.02326211, 0.02571722,
        0.        ],
       [0.07142857, 1.        , 0.07824608, ..., 0.02326211, 0.        ,
        0.        ],
```

In [61]:
```
new_df[new_df['title'] == 'The Lego Movie'].index[0]
```

Out[61]: 744

In [62]:
```
new_df[new_df['title'] == 'Spider-Man'].index[0]
```

Out[62]: 159

In [63]:
```
def recommend(movie):
    index = new_df[new_df['title'] == movie].index[0]
    distances = sorted(list(enumerate(similarity[index])),reverse=True,key = lambda x: x[1])
    for i in distances[1:6]:
        print(new_df.iloc[i[0]].title)
```

In [64]:
```
recommend('Spider-Man')
```

```
Spider-Man 3
Spider-Man 2
The Amazing Spider-Man
The Amazing Spider-Man 2
Evil Dead II
```

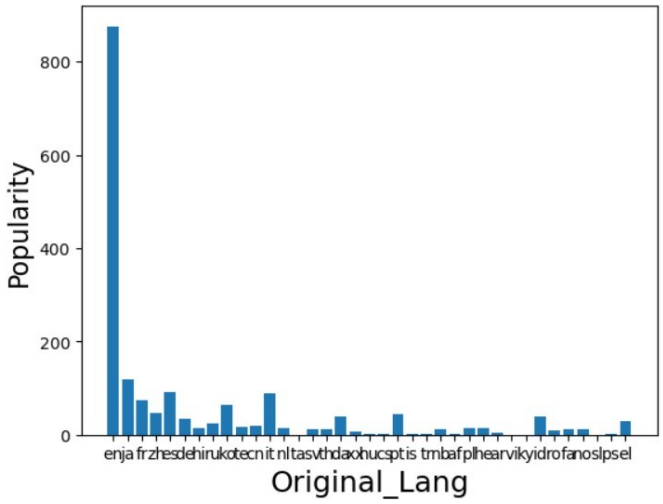
In [65]:
```
recommend('Avatar')
```

```
Lifeforce
Aliens vs Predator: Requiem
Battle: Los Angeles
Titan A.E.
Independence Day
```

Bar graph upon which language is being preferred most while watching a movie:

```
In [28]:
1 x = df['original_language']
2 y = df['popularity']
3 plt.xlabel('Original_Lang', fontsize=18)
4 plt.ylabel('Popularity', fontsize=16)
5 plt.bar(x,y)
```

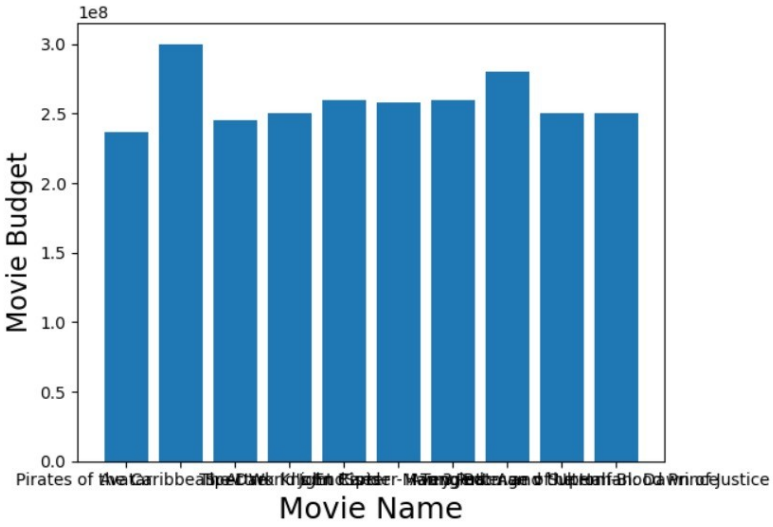
```
Out[28]: <BarContainer object of 4803 artists>
```



This graph shows first 10 films annual budget:

```
In [71]:
1 x = df1['title']
2 y = df1['budget']
3 plt.xlabel('Movie Name', fontsize=18)
4 plt.ylabel('Movie Budget', fontsize=16)
5 plt.bar(x,y)
```

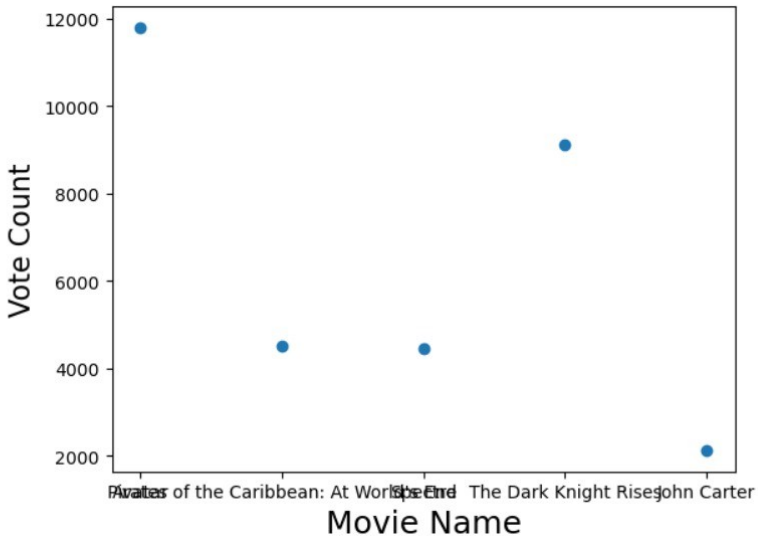
```
Out[71]: <BarContainer object of 10 artists>
```



Movie and their vote counts:

In [80]:
```
x = df2['title']
y = df2['vote_count']
plt.xlabel('Movie Name', fontsize=18)
plt.ylabel('Vote Count', fontsize=16)
plt.scatter(x,y)
```

Out[80]: <matplotlib.collections.PathCollection at 0x241df541630>



Conclusion:

Content-based movie recommendation systems continue to evolve, and they remain a vital component of the broader recommendation landscape.

While they excel at providing personalized recommendations based on intrinsic movie features, overcoming challenges like the cold start problem and limited serendipity is crucial.

Hybrid models and the integration of deep learning techniques are promising directions for improving the performance of content-based systems, ultimately enhancing the movie-watching experience for users.

As the entertainment industry continues to grow, content-based filtering will undoubtedly play a

significant role in helping users navigate an ever-expanding catalog of movies.

FUTURE WORK:

In case of content-based filtering method we can look up on the cast and crew also where we have only considered the genre and also, we can see at the movies are compatible or not.

Comparison of collaborative filtering-based approaches and different kind of similarity measurements would be a good one for the recommender system .

We can use matrix factorization for calculating the number of factors involved. We can also apply deep learning techniques to for the enhance the recommender system and optimising the efficiency of the system.

We can work on different areas such as video some books even recommending some songs to the users

of the mobile phones based on the platforms of the different apps available on the Play Store.

Various techniques such as clustering classification can be used to get the better version of our recommender system which for the enhance the accuracy of the overall model.