Department of Computer Engineering
St. Francis Institute of Technology
University of Mumbai
2023-2024

A Mini-Project
Report On
“Netflix Recommendation system”

Subject -Machine Learning [CSL 701]

Under the guidance of
Dr. Nazneen Ansari

By

Cyrus Coutinho (Roll no 34, Pid no. 202023)

Seona Dabre (Roll no 41, Pid no. 202097)

Elston Farel (Roll no 42, Pid 202041)

Darshan Jain (Roll no 50, Pid 202060)

Table of Contents

ABSTRACT

Nowadays, Netflix is one of the top content platforms in the world. Netflix distributes different content to the user such as movies and TV shows. In the age of digital streaming platforms and vast cinematic choices, movie recommendation systems have become an integral part of enhancing user experience and content discovery. Netflix recommendation system is a sophisticated movie recommendation system designed to provide personalized movie suggestions to users based on their individual preferences. This abstract provides an overview of the key features and functionalities of netflix recommendation system. "Netflix recommendation system" is a movie recommendation system that uses collaborative filtering and advanced machine learning algorithms to offer customers highly customized movie selections. The goal of netflix recommendation system, which is dedicated to user privacy and data security, is to improve the movie-watching experience by letting users find and enjoy a wide variety of movies that suit their individual tastes and inclinations.

Keywords: *Machine Learning , recommendation, series/ movie, Netflix.*

LIST OF FIGURES

LIST OF ABBREVIATIONS

Sr. No.	Abbreviations	Expanded form
1.	IMDb	Internet Movie Database
2.	ROC	Registrar of Companies
3.	ICAIS	Artificial Intelligence and Smart Systems
4.	HNPCC	Hybrid Novel Pearson Correlation Coefficient
5.	KNN	K-Nearest Neighbor
6.	ICICCS	International Conference on Intelligent Computing and Control Systems
7.	MLBDB	Machine Learning, Big Data and Business Intelligence
8.	ICTC	Information and Communication Technology Convergence
9.	ICSEC	International Computer Science and Engineering Conference
10.	ICSPIS	Iranian Conference on Intelligent Systems and Signal Processing

Chapter 1: INTRODUCTION

1.1 Introduction

The large amount of content on streaming services like Netflix may be both a blessing and a disadvantage in the current digital era. While there is something for everyone, the sheer volume of possibilities can be daunting, leaving people unsure where to begin their leisure adventure. This is where recommendation algorithms come into play, with Netflix's being one of the most sophisticated and important in the streaming market.

Netflix's recommendation system is a shining illustration of the entertainment industry's reliance on data-driven algorithms and machine learning. It goes beyond the standard broadcasting approach, in which everyone receives the same content, and instead tailors the viewing experience to each user's particular tastes and preferences. This level of customization has been critical to Netflix's quick growth and appeal.

This Netflix recommendation system investigation will delve into the inner workings of this technical marvel. We'll talk about the algorithms and approaches that power the system's suggestions, how user data is collected and used, and how this system affects user engagement and content discovery.

1.2 Problem definition

In a world of endless entertainment options on Netflix, the task of deciding what to watch can often feel like an insurmountable challenge. Sometimes, it's difficult to pinpoint our exact taste, making it a daunting task to find the perfect show or movie. This is where the Netflix show recommendation system steps in, becoming an invaluable asset. By utilizing intricate algorithms and user data, this system helps curate a personalized selection of content that matches each viewer's unique preferences, previous choices, and viewing history. As a result, the daunting process of sifting through an overwhelming variety of options is transformed into a seamless and enjoyable experience. With these tailored recommendations, viewers can spend less time searching and more time indulging in the genres and stories that truly resonate with them, enriching their Netflix journey and ensuring they find the perfect fit for their mood and interests.

1.3 Prerequisites

This project requires a good knowledge of Python and Machine Learning. Modules required for this project are pandas, ski-learn, flask and math.

The required modules for this project are –

1) sklearn :
2) pandas
3) flask
4) Math

Chapter 2: RELATED WORK

There is a diverse range of approaches and insights into the field of movie recommendation systems. [1]conducted an analysis of Netflix user interests in Asian countries, while [2] proposed a knowledge graph-based personalized recommendation system. [3] presented a knowledge graph-enabled multi-featured system. [4]provided a comprehensive survey, while [5] and [6] (2023) compared hybrid correlation methods.[7] introduced content-based and sentiment analysis. [8]proposed an improved approach, [9] developed a YouTube and movie recommendation system, and [10]focused on LSTM-CNN. [11] used K-means for weight-based recommendations, [12]explored director-based recommendations, and [3]considered temporal user preferences in content-based recommendations. These papers offer a comprehensive overview of different techniques and insights relevant to your project.

Chapter 3: DATASET

3.1 Description of the data set

To build an effective recommendation system for Netflix, a rich and diverse dataset is required. Netflix typically uses a combination of several datasets and data sources to create a comprehensive understanding of user preferences and content characteristics. Here's a description of the key components of a dataset for a Netflix recommendation system:

The Netflix recommendation system relies on this diverse dataset to create user profiles, generate content embeddings, and apply machine learning algorithms and techniques such as collaborative filtering, matrix factorization, deep learning, and natural language processing to make personalized content recommendations. It's important to note that Netflix takes user privacy seriously and anonymizes or aggregates data to protect user identities while still providing a personalized viewing experience. Ethical and responsible data handling practices are a critical aspect of dataset management for a recommendation system of this scale.

Title	Genre	Tags	Languages	Country Availability	Runtim
9144 unique values	Drama 7% Comedy 6% Other (8168) 87%	8531 unique values	English 40% Japanese 8% Other (4856) 52%	Japan 7% South Korea 4% Other (8354) 89%	1-2 hour < 30 minu Other (16
Lets Fight Ghost	Crime, Drama, Fantasy, Horror, Romance	Comedy Programmes,Romantic TV Comedies,Horror Programmes,Thai TV Programmes	Swedish, Spanish	Thailand	< 30 mir
HOW TO BUILD A GIRL	Comedy	Dramas,Comedies,Films Based on Books,British	English	Canada	1-2 hour
The Con-Heartist	Comedy, Romance	Romantic Comedies,Comedies,Romantic Films,Thai Comedies,Thai Films	Thai	Thailand	> 2 hrs
Gleboka woda	Drama	TV Dramas,Polish TV Shows,Social Issue TV Dramas	Polish	Poland	< 30 mir
Only a Mother	Drama	Social Issue Dramas,Dramas,Movies Based on Books,Period Pieces,Swedish Movies	Swedish	Lithuania,Poland,France,Italy,Spain,Greece,Belgium,Portugal,Netherlands,Germany,Switzerland,United K...	1-2 hour

Snowroller	Comedy	Sports Movies,Sports Comedies,Comedies,Swedish Movies	Swedish, English, German, Norwegian	Lithuania,Poland,France,Italy,Spain,Greece,Czech Republic,Belgium,Portugal,Hungary,Slovakia,Netherla...	1-2 hour
The Invisible	Crime, Drama, Fantasy, Mystery, Thriller	Thriller Movies,Movies Based on Books,Supernatural Thrillers,Swedish Movies	English	Lithuania,Poland,France,Italy,Spain,Greece,Czech Republic,Belgium,Portugal,Hungary,Slovakia,Netherla...	1-2 hour
The Simple Minded Murderer	Drama	Social Issue Dramas,Dramas,Movies Based on Books,Period Pieces,Classic Movies,Swedish Movies	Scanian, Swedish	Lithuania,Poland,France,Italy,Spain,Greece,Czech Republic,Belgium,Portugal,Hungary,Slovakia,Netherla...	1-2 hour
To Kill a Child	Short, Drama	Dramas,Swedish Movies	Spanish	Lithuania,Poland,France,Italy,Spain,Greece,Czech Republic,Belgium,Portugal,Hungary,Slovakia,Netherla...	< 30 min

Fig 1. Original dataset snapshot

Chapter 4: METHODS AND ALGORITHMS

Content-based recommendation systems recommend items (in this case, movies or shows) to users based on the characteristics and features of the items themselves, as well as the preferences of the user. Here's a breakdown of the key components in your code:

1. Data Preprocessing: The code starts by reading a dataset of Netflix movies and performing various data preprocessing steps. It cleans and transforms the data, including handling missing values, standardizing text data (lowercase and removing spaces), and creating a "soup" of features from different attributes like Genre, Tags, Actors, and Viewer Rating.

2. Cosine Similarity: The code computes the cosine similarity between movies based on the "soup" feature vectors. Cosine similarity is a measure of similarity between two non-zero vectors in an inner product space that measures the cosine of the angle between them. In this context, it's used to find movies that are similar to each other based on their content features.

3. Flask Web Application: The code uses the Flask web framework to create a web application for users to interact with. Users can input movie titles and select language preferences. The application then recommends movies based on the similarity of their content features to the input movies and filters them by language preferences.

4. Recommendation Function: The get_recommendations function takes an input movie title and computes recommendations based on cosine similarity.

5. Routes: The Flask application defines routes for the homepage, about page (where users can input their preferences), and individual movie pages for details.

Overall, this code implements a content-based movie recommendation system that recommends movies similar to the ones users input, with an additional filter for language preferences. It uses cosine similarity to determine the similarity between movies based on their content features.

Steps for developing Netflix recommendation system:

1. Import Libraries and initialize variables.

   Code:

```
#importing libraries
import math
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly_express as px
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

2. Preprocessing the data

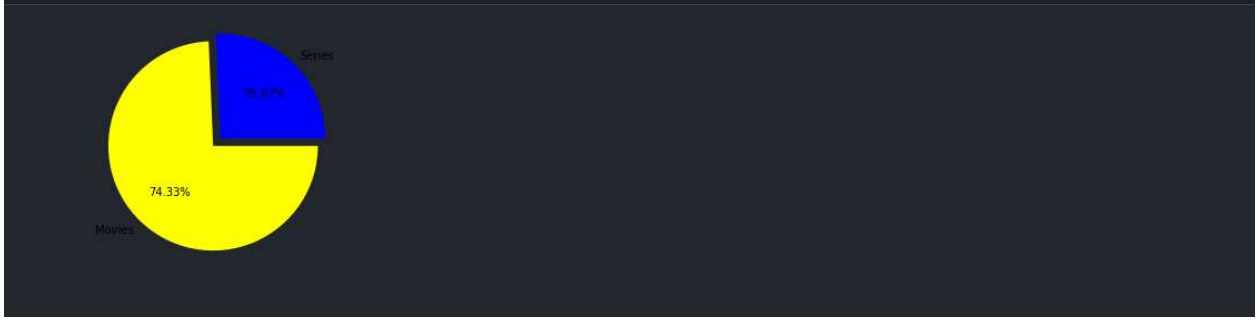
   Code:

```
#read the dataset file
netflix_data = pd.read_csv('NetflixDataset.csv',encoding='latin-1', index_col = 'Title')
netflix_data.head(2)
```

3. Comparison and classification of the dataset

   Code:

```
#comparsion and classification of the dataset
color = ['blue', 'yellow']
label = ['Series', 'Movies']
sizes = [netflix_data[netflix_data['Series or Movie'] == 'Series'].size, netflix_data[netflix_data['Series or Movie'] == 'Movie'].size]
explode = (0.1, 0)
fig, ax = plt.subplots()
ax.pie(sizes, explode, label, color, '%2.2f%%')
ax.axis('equal')
plt.show()
```



Training the model

We will split our dataset into 80:20 ratios where 80 for training and 20 for testing. For classifying our data, we will use the Support Vector Machine classifier technique which we have discussed above.

Code:

```
Language = netflix_data.Languages.str.get_dummies(',')
Lang = Language.columns.str.strip().values.tolist()
Language = netflix_data['Languages']
Language_Count = dict()
for i in Lang:
    p = Language.str.count(i).sum()
    Language_Count[i] = int(p)
print(len(Language_Count))
```

Python

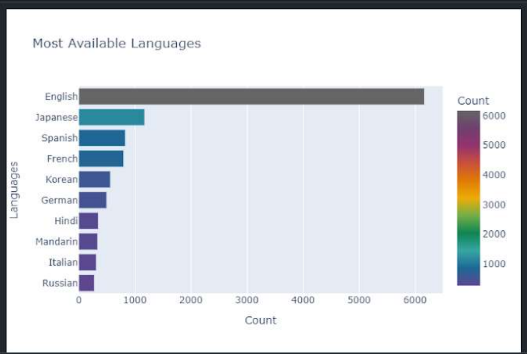
```
184
```

```
Language_Count = {k: v for k, v in sorted(Language_Count.items(), key=lambda item: item[1], reverse = True)}
top_languages = {"Languages": list(Language_Count.keys()), "Count": list(Language_Count.values())}
```

Python

```
fig = px.bar(pd.DataFrame(top_languages)[:10], y = 'Languages', x = 'Count', orientation = 'h', title = 'Most Available Languages', color = 'Count', color_continuous_scale = px.colors.qualitative.Prism).
fig.show()
```

Python




```
Genres = netflix_data.Genre.str.get_dummies(',')
Genre = Genres.columns.str.strip().values.tolist()
Genres = netflix_data['Genre']
Genre_Count = dict()
for i in Genre:
    p = Genres.str.count(i).sum()
    Genre_Count[i] = int(p)
print(len(Genre_Count))
```

Python

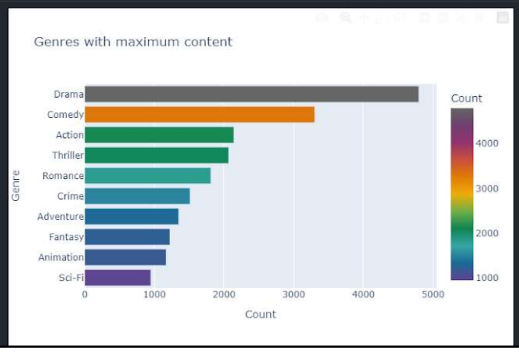
```
28
```

```
Genre_Count = {k: v for k, v in sorted(Genre_Count.items(), key=lambda item: item[1], reverse = True)}
top_genres = {"Genre": list(Genre_Count.keys()), "Count": list(Genre_Count.values())}
```

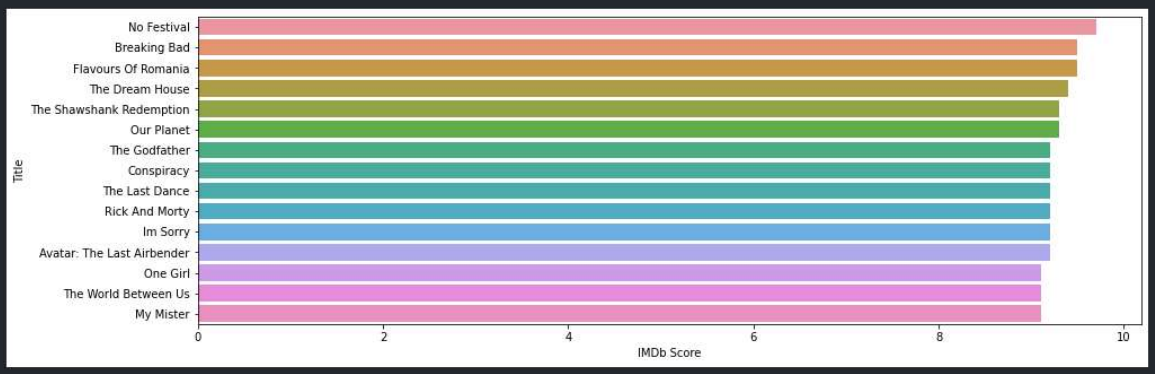
Python

```
fig = px.bar(pd.DataFrame(top_genres)[:10], y = 'Genre', x = 'Count', orientation = 'h', title = 'Genres with maximum content', color = 'Count', color_continuous_scale = px.colors.qualitative.Prism).upda
fig.show()
```

Python



```
[12]
top_15 = netflix_data.sort_values(by = ['IMDb Score'], ascending = False).head(15)
plt.figure(figsize = (15,5))
sns.barplot(data = top_15, y = top_15.index, x = "IMDb Score")
plt.show()
```




```
[19]
print(netflix_data[['IMDb Score']].describe())
netflix_data['IMDb Score'].mode()
#this feature will be used to sort the movie or series list to represent the recommended items
```

```
       IMDb Score
count  9125.000000
mean      6.955134
std       0.896501
min       1.600000
25%       6.500000
50%       7.000000
75%       7.500000
max       9.700000

0    6.6
dtype: float64
```

```
[20]
netflix_data['IMDb Score'] = netflix_data['IMDb Score'].apply(lambda x: 6.6 if x == 0 or math.isnan(x) else x)
print(netflix_data[['IMDb Score']].describe())
#since no value has suffered for change greater than 0.0003 after replacing the null values with mode value, so we replace the null values with 6.6
```

```
       IMDb Score
count  9132.000000
mean      6.954862
std       0.896212
min       1.600000
25%       6.500000
50%       7.000000
75%       7.500000
max       9.700000
```

Chapter 5: EXPERIMENTS

5.1 Data Preparation

The CSV file contains 9404 rows, each row for each email. There are 22 columns. The first column indicates the Email name. The name has been set with numbers and not the recipient's name to protect privacy. The columns are the 9404 most common words in all the Netflix shows and movie names and its Title , Genre, Tags, Languages, Country Availability, Runtime, Director, Writer, Actors, View Rating, IMDb Score, Awards Received, Awards Nominated For, Boxoffice, Release Date, Netflix Release Date, Production House, Netflix Link, Summary, Series or Movie, IMDb Votes, Image, after excluding the non-alphabetical characters/words.

5.2 Eploratory Analysis

Correlation matrix visualization before feature selection.

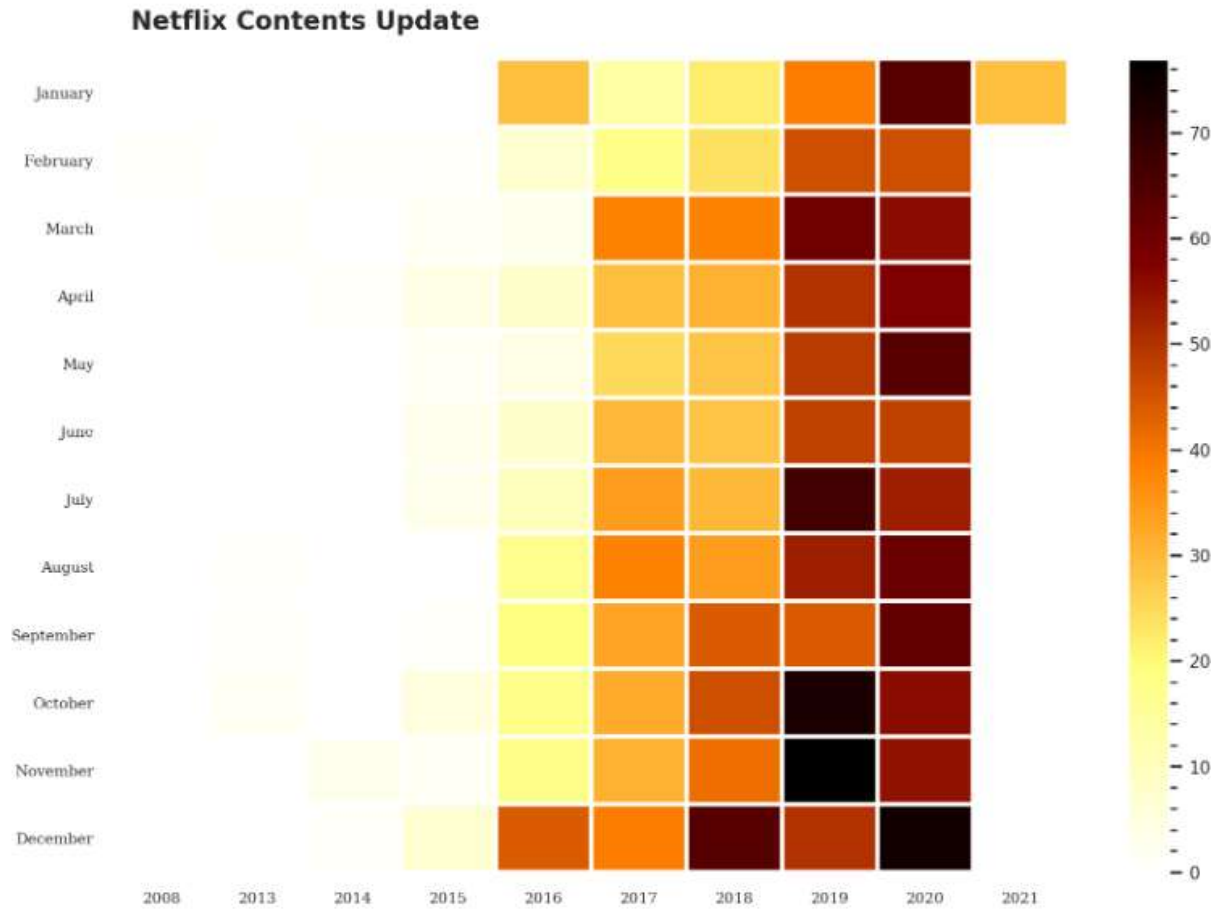


Fig 2. Correlation Matrix

5.3 Feature Selection

Feature Selection using Backward Elimination (P-value) algorithm:

Further, the data was passed through the backward elimination function to select the most relevant features which gave the following result

```
fig = px.bar(pd.DataFrame(top_languages)[:10], y = 'Languages', x = 'Count', orientation = 'h', title = 'Most Available Languages', color = 'Count', color_continuous_scale = px.colors.qualitative.Prism).update_yaxes(categoryorder = 'total ascending')
fig.show()
```

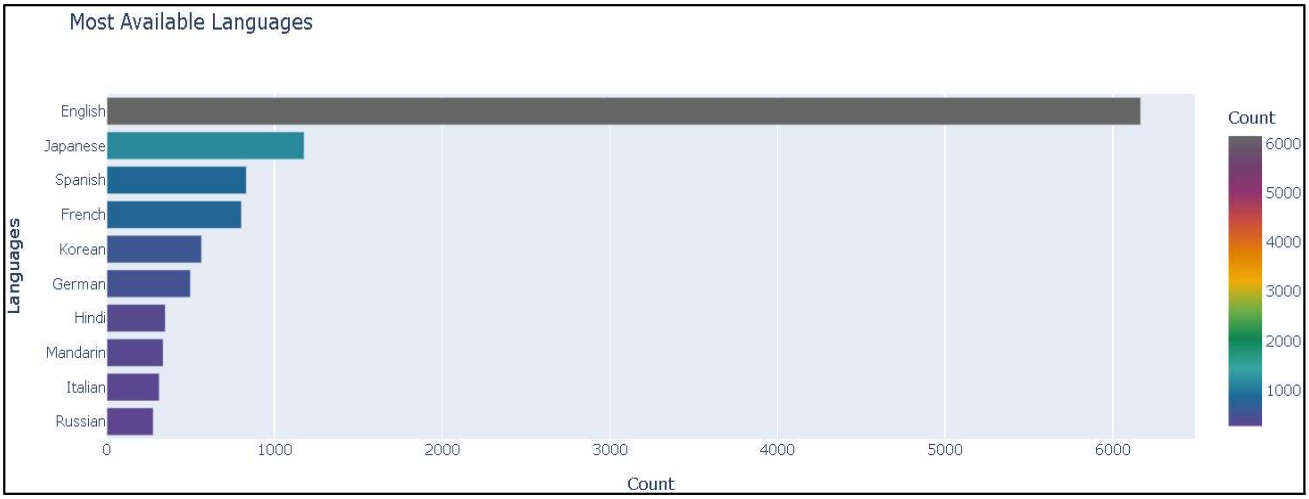


Fig 3. Most Available Languages

```
fig = px.bar(pd.DataFrame(top_genres)[:10], y = 'Genre', x = 'Count', orientation = 'h', title = 'Genres with maximum content', color = 'Count', color_continuous_scale = px.colors.qualitative.Prism).update_yaxes(categoryorder = 'total ascending')
fig.show()
```

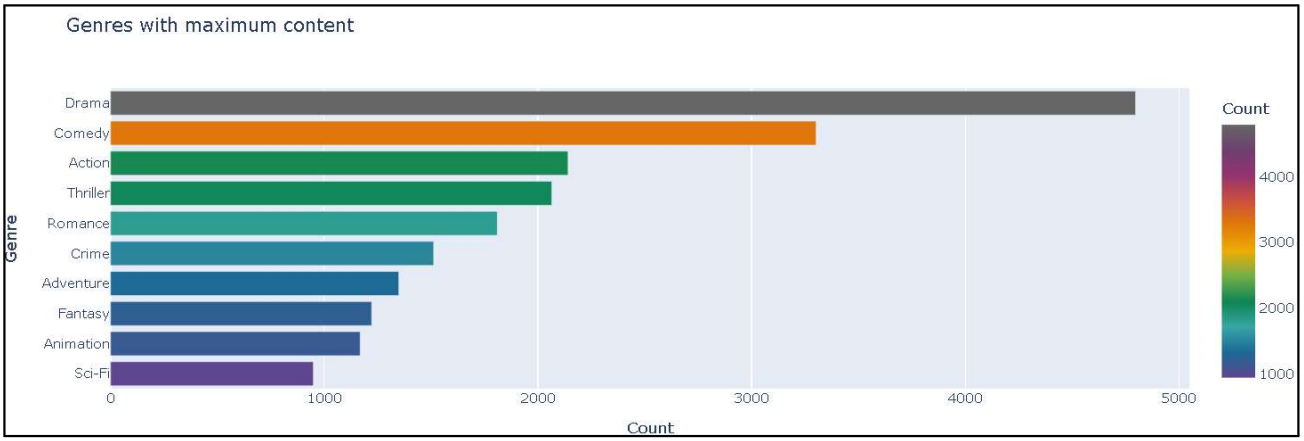


Fig 4. Genres With Maximum Content

```
top_15 = netflix_data.sort_values(by = ['IMDb Score'], ascending =
False).head(15)
plt.figure(figsize = (15,5))
sns.barplot(data = top_15, y = top_15.index, x = "IMDb Score")
plt.show()
```

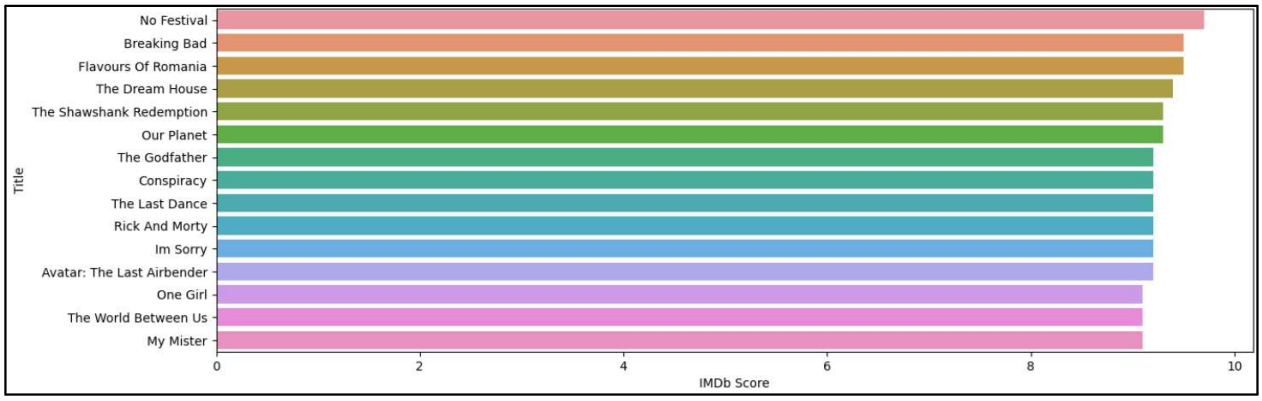


Fig 5. Relation between IMDb Score and Title

5.3 Training and testing

Finally, this resulting data was split into 80% train and 20% test data, which was further passed further to fit, predict and score the model.

Chapter 6: EVALUATION METRICS

Evaluation Metrics: Various evaluation metrics are used to assess the model's performance, including accuracy, precision, recall, F1-score, and ROC curve analysis. The choice of metrics depends on the specific requirements of the email classification task.

To define an algorithm's performance, we examine the parameters such as:

6.1 Confusion Matrix

6.2 Accuracy

The Accuracy parameter indicates the percentage of correct predictions. It does not consider positives or negatives separately, thus other performance measurements are employed in addition to accuracy. The maximum accuracy is indicated by the number 1, which is calculated as:

$$Accuracy = \frac{TN + TP}{TP + FN + FP + TN}$$

6.3 Precision

Precision is defined as the ratio of True Positives to all Positives or the proportion of recollected relevant events. The precision is calculated as follows:

$$Precision = \frac{TP}{TP + FP}$$

6.4 Recall

The recall is a measure of how well our model identifies True Positives (NetflixDataSet). The recall is computed as the ratio of Positive samples that were properly categorized as Positive to the total number of Positive samples. The recall of the model assesses its ability to recognise Positive samples.

$$Recall = \frac{TP}{TP+FN}$$

6.5 F-Measure

The F-score, often known as the F1-score, is the accuracy of a model on a dataset. It is used to evaluate binary classification techniques that classify examples as either 'positive' or 'negative'.The F-score is a method for combining the model's precision and recall; it is defined as the harmonic mean of the precision and recall of the model.

Chapter 7: DISCUSSION ON RESULTS

One main experiment has been conducted to test the feature reduction module as a core concept in this paper. The performance is recorded independently at each processing step of the proposed approach and architecture to show the effect of including such steps and modules. The experiment is responsible for evaluating several similarity measures and types, to find out the most fitting similarity features.

```
result = 0
def get_recommendations(title, cosine_sim):
    global result
    title=title.replace(' ','').lower()
    idx = indices[title]
    # Get the pairwsie similarity scores of all movies with that movie
    sim_scores = list(enumerate(cosine_sim[idx]))
    # Sort the movies based on the similarity scores
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    # Get the scores of the 50 most similar movies
    sim_scores = sim_scores[1:51]
    # Get the movie indices
    movie_indices = [i[0] for i in sim_scores]
    # Return the top 10 most similar movies
    result =  netflix_data.iloc[movie_indices]
    result.reset_index(inplace = True)
    return result
```

```
df = pd.DataFrame()
movienames = ['Annabelle Comes Home','The Nun', 'Insidious: The Last Key', 'Conjuring 2', 'Insidious: Chapter 3']
languages = ['English', 'Hindi']
for moviename in movienames:
    get_recommendations(moviename,cosine_sim2)
    for language in languages:
        df = pd.concat([result[result['Languages'].str.count(language) > 0], df], ignore_index=True)
df.drop_duplicates(keep = 'first', inplace = True)
df.sort_values(by = 'IMDb Score', ascending = False, inplace = True)
```

```
print(df.shape)
print(df.head())
```

```
(118, 22)
            Title                                Genre  \
114     The Others          Horror, Mystery, Thriller
50   The Conjuring          Horror, Mystery, Thriller
169     Hereditary  Drama, Horror, Mystery, Thriller
105          Split                    Horror, Thriller
201    Conjuring 2          Horror, Mystery, Thriller

                                                  Tags          Languages  \
114  20th Century Period Pieces,Thrillers,Mysteries...            English
50   Horror Films,Thrillers,Supernatural Horror Fil...     English, Latin
169  Supernatural Horror Movies,Horror Movies,Teen ...  English, Spanish
105  Psychological Thrillers,Horror Movies,Thriller...            English
201  Horror Films,Supernatural Horror Films,Films B...            English

                                Country Availability   Runtime  \
114  Italy,Sweden,Switzerland,Turkey,Iceland,India,...  1-2 hour
50   France,Belgium,Lithuania,Switzerland,United Ki...  1-2 hour
169      South Korea,United Kingdom,Canada,Japan,India   > 2 hrs
105  Belgium,Netherlands,South Korea,Poland,France,...  1-2 hour
201  South Korea,Switzerland,Brazil,Germany,Italy,A...   > 2 hrs

              Director                                          Writer  \
114  Alejandro Amenábar                              Alejandro Amenábar
50            James Wan                        Chad Hayes, Carey W. Hayes
...
105   428760.0  https://occ-0-768-769.1.nflxso.net/dnm/api/v6/...
201   226713.0  https://occ-0-2773-2774.1.nflxso.net/dnm/api/v...

[5 rows x 22 columns]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Output:

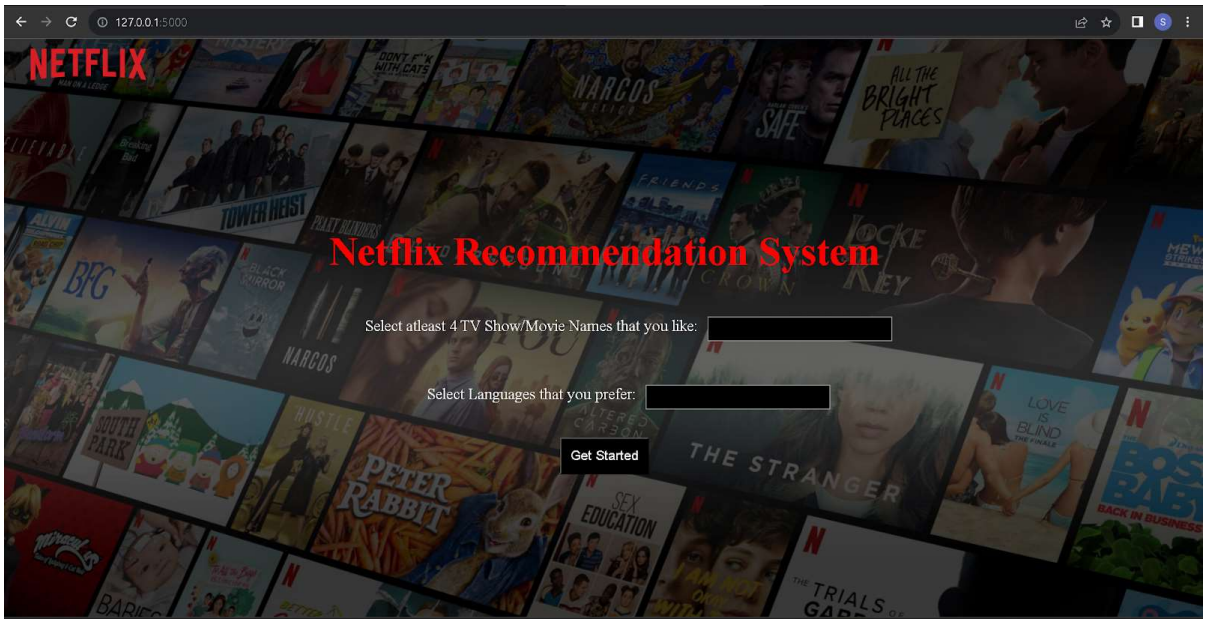


Fig 7.1: Main home page

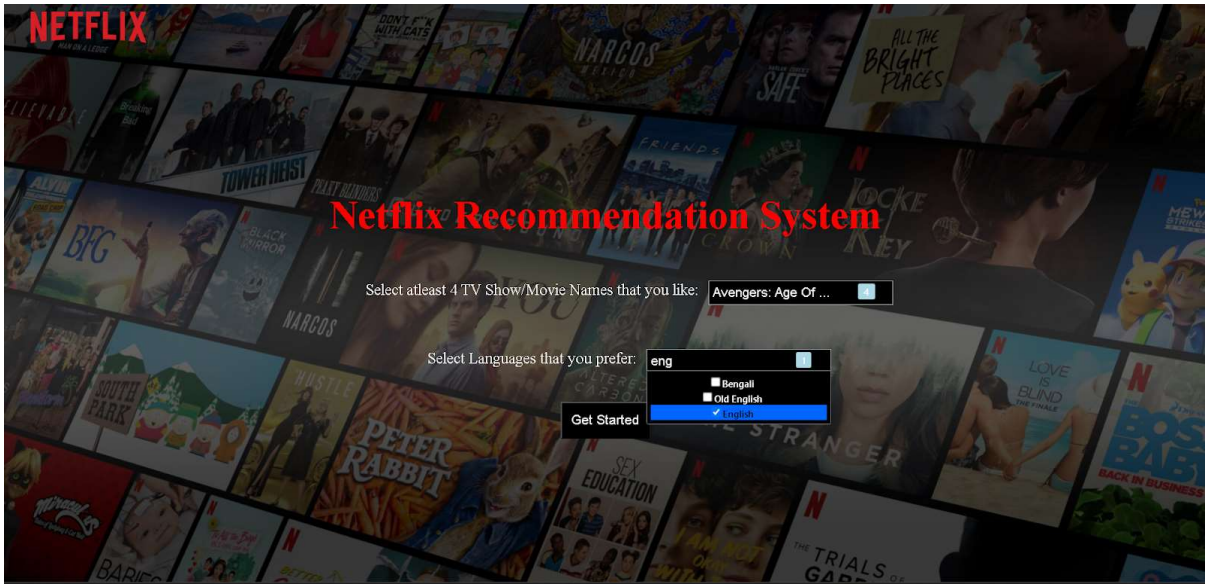


Fig 7.2: Input data format

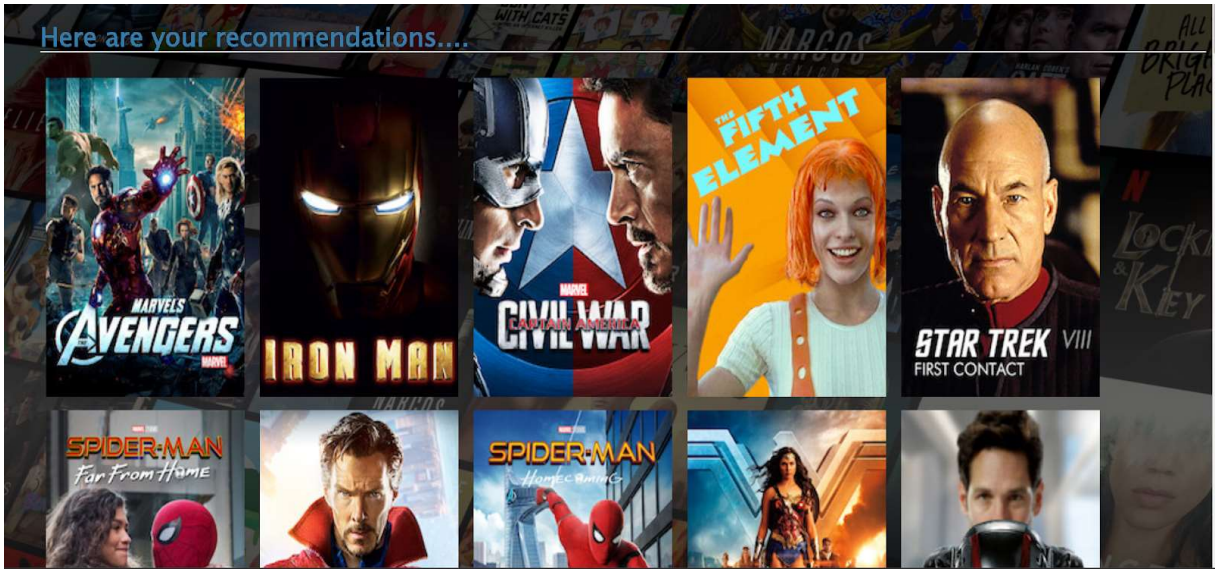


Fig 7.3: Recommendation page

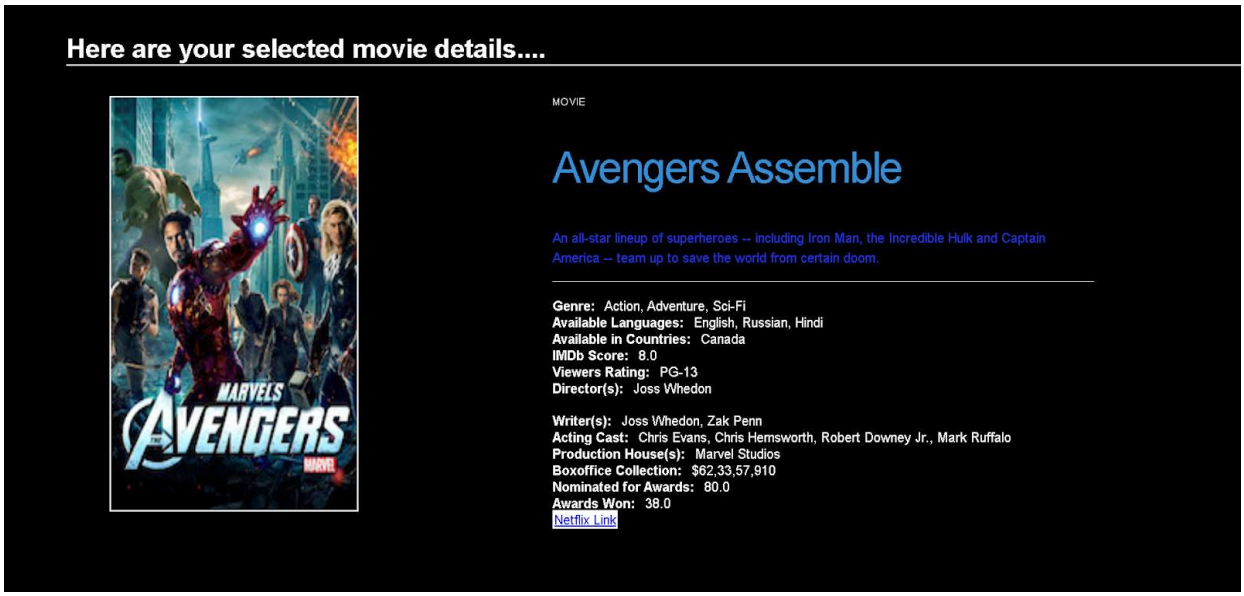


Fig 7.4: More information about the movie/series

Chapter 8: CODE

8.1 Libraries and Functions Used

Code:

app.py

```
import math
from flask import Flask, render_template, request
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

def prepare_data(x):
        return str.lower(x.replace(" ", ""))

def create_soup(x):
    return x['Genre'] + ' ' + x['Tags'] + ' ' +x['Actors']+' '+
x['ViewerRating']

def get_recommendations(title, cosine_sim):
    global result
    title=title.replace(' ','').lower()
    idx = indices[title]
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:51]
    movie_indices = [i[0] for i in sim_scores]
    result =  netflix_data.iloc[movie_indices]
    result.reset_index(inplace = True)
    return result

netflix_data = pd.read_csv('app/NetflixDataset.csv',encoding='latin-1',
index_col = 'Title')
netflix_data.index = netflix_data.index.str.title()
netflix_data = netflix_data[~netflix_data.index.duplicated()]
netflix_data.rename(columns={'View Rating':'ViewerRating'}, inplace=True)
Language = netflix_data.Languages.str.get_dummies(',')
Lang = Language.columns.str.strip().values.tolist()
Lang = set(Lang)
Titles = set(netflix_data.index.to_list())
```

```
netflix_data['Genre'] = netflix_data['Genre'].astype('str')
netflix_data['Tags'] = netflix_data['Tags'].astype('str')
netflix_data['IMDb Score'] = netflix_data['IMDb Score'].apply(lambda x:
6.6 if math.isnan(x) else x)
netflix_data['Actors'] = netflix_data['Actors'].astype('str')
netflix_data['ViewerRating'] = netflix_data['ViewerRating'].astype('str')
new_features = ['Genre', 'Tags', 'Actors', 'ViewerRating']
selected_data = netflix_data[new_features]
for new_feature in new_features:
    selected_data.loc[:, new_feature] = selected_data.loc[:,
new_feature].apply(prepare_data)
selected_data.index = selected_data.index.str.lower()
selected_data.index = selected_data.index.str.replace(" ",'')
selected_data['soup'] = selected_data.apply(create_soup, axis = 1)

count = CountVectorizer(stop_words='english')
count_matrix = count.fit_transform(selected_data['soup'])
cosine_sim2 = cosine_similarity(count_matrix, count_matrix)
selected_data.reset_index(inplace = True)
indices = pd.Series(selected_data.index, index=selected_data['Title'])
result = 0
df = pd.DataFrame()

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html', languages = Lang, titles =
Titles)

@app.route('/about',methods=['POST'])
def getvalue():
    global df
    movienames = request.form.getlist('titles')
    languages = request.form.getlist('languages')
    for moviename in movienames:
        get_recommendations(moviename,cosine_sim2)
        for language in languages:
            df = pd.concat([result[result['Languages'].str.count(language)
> 0], df], ignore_index=True)
```

```
    df.drop_duplicates(keep = 'first', inplace = True)
    df.sort_values(by = 'IMDb Score', ascending = False, inplace = True)
    images = df['Image'].tolist()
    titles = df['Title'].tolist()
    return render_template('result.html',  titles =  titles, images =
images)

@app.route('/moviepage/<name>')
def movie_details(name):
    global df
    details_list = df[df['Title'] == name].to_numpy().tolist()
    return render_template('moviepage.html', details = details_list[0])


if __name__ == '__main__':
    app.run(debug=False)
```

\

Chapter 9: CONCLUSION

The Netflix Recommendation System represents a groundbreaking achievement in the realm of digital entertainment. With a user base spanning the globe and a vast library of content, Netflix has leveraged cutting-edge technology, data science, and machine learning to transform the way viewers discover and engage with movies and TV shows.

In conclusion, the Netflix Recommendation System is not just a technological marvel but also a testament to the power of data-driven decision-making in the entertainment industry. It has reimagined how audiences connect with content, fostering a deeper, more immersive viewing experience. As Netflix continues to innovate and refine its recommendation algorithms, it is poised to remain a leader in the ever-evolving world of streaming entertainment, ultimately delivering more value to its subscribers and shaping the future of digital media consumption.

Chapter 10: REFERENCES

[1] N. Kamarudin, A. Daheche and A. Khmag, "Netflix User and Movies Interest Analysis for Asian Countries," 2022 9th International Conference on Electrical and Electronics Engineering (ICEEE), Alanya, Turkey, 2022, pp. 339-343, doi: 10.1109/ICEEE55327.2022.9772585.

[2] B. Zhao, "A Personalized Movie Recommendation System Based on Knowledge Graph," 2023 IEEE 3rd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China, 2023, pp. 588-592, doi: 10.1109/ICIBA56860.2023.10165087.

[3] L. S. Nair and J. Cheriyan, "Multi-Featured Movie Recommendation Using Knowledge Graph," 2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT), Bengaluru, India, 2023, pp. 590-594, doi: 10.1109/IDCIoT56793.2023.10053435.

[4] R. Lavanya, U. Singh and V. Tyagi, "A Comprehensive Survey on Movie Recommendation Systems," 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), Coimbatore, India, 2021, pp. 532-536, doi: 10.1109/ICAIS50930.2021.9395759.

[5] S. M. Shoaib and J. K, "Comparison of Hybrid Novel Pearson Correlation Coefficient (HNPCC) with K-Nearest Neighbor (KNN) Model to Improve Accuracy for Movie Recommendation System," 2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2023, pp. 1-6, doi: 10.1109/ACCAI58221.2023.10200272.

[6] S. Katkam, A. Atikam, P. Mahesh, M. Chatre, S. S. Kumar and S. G. R, "Content-based Movie Recommendation System and Sentimental analysis using ML," 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2023, pp. 198-201, doi: 10.1109/ICICCS56967.2023.10142424.

[7] S. Agrawal and P. Jain, "An improved approach for movie recommendation system," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2017, pp. 336-342, doi: 10.1109/I-SMAC.2017.8058367.

[8] S. S. K, M. Srivastava and Mohana, "YouTube and Movie Recommendation System Using Machine Learning," 2023 Second International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 2023, pp. 1352-1356, doi: 10.1109/ICEARS56392.2023.10084999.

[9] H. Wang, N. Lou and Z. Chao, "A Personalized Movie Recommendation System based on LSTM-CNN," 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China, 2020, pp. 485-490, doi: 10.1109/MLBDBI51377.2020.00102.

[10] M. T. Himel, M. N. Uddin, M. A. Hossain and Y. M. Jang, "Weight based movie recommendation system using K-means algorithm," 2017 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), 2017, pp. 1302-1306, doi: 10.1109/ICTC.2017.8190928.

[11] O. Chunhapran, C. Phromsuthirak, M. Hama and M. Maliyaem, "Movie Recommendation System Using Director-Based," 2022 26th International Computer Science and Engineering Conference (ICSEC), Sakon Nakhon, Thailand, 2022, pp. 151-155, doi: 10.1109/ICSEC56337.2022.10049347.

[12] B. R. Cami, H. Hassanpour and H. Mashayekhi, "A content-based movie recommender system based on temporal user preferences," 2017 3rd Iranian Conference on Intelligent Systems and Signal Processing (ICSPIS), Shahrood, Iran, 2017, pp. 121-125, doi: 10.1109/ICSPIS.2017.8311601.