SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

A PROJECT REPORT ON

" Movie Recommendation System"

SUBMITTED TOWARDS THE PARTIAL

FULFILLMENT OF REQUIREMENT OF

DATA SCIENCE AND BIG DATA ANALYTICS LABORATORY

IN THIRD YEAR COMPUTER ENGINEERING BY

1. Mr. Joshi Shubham [3168]
2. Mr. Kandekar Shivam [3171]
3. Ms. Jadhav Kalyani [3159]

(T . E. COMPUTER ENGINEERING)

UNDER THE GUIDANCE OF

Dr. D. R. Patil

DURING THE ACADEMIC YEAR 2022-2023

Department of Computer Engineering,
Amrutvahini College of Engineering,
Sangamner- 422 608

Amrutvahini College of Engineering, Sangamner

CERTIFICATE

This is to certify that the project entitled

1. Mr. Joshi Shubham [3168]
2. Mr. Kandekar Shivam [3171]
3. Ms. Jadhav Kalyani [3159]

T. E. (A) COMPUTER ENGINEERING

have successfully completed the work associated with Data Science and Big Data Analytics Laboratory (310256) titled as

"Movie Recommendation System"

and has submitted the work book associated under my supervision, in the partial fulfillment of Third Year Bachelor of Engineering (Choice Based Credit System) (2019 course) of Savitribai Phule Pune University.

Dr. D. R. Patil **Place: Sangmaner**

[Guide] Date: / /2023

Acknowledgement

It is our proud privilege and duty to acknowledge the kind of help and guidance received from several people in the preparation of this report. It would not have been possible to prepare this report in this form without their valuable help, co-operation and guidance.

First and foremost, we wish to record our sincere gratitude to the management of this college and to our respected Principal for his constant support and encouragement in the preparation of this report and for the availability of library and laboratory facilities needed to prepare this report. Our sincere thanks to Prof. R. L. Paikrao, for his valuable suggestions and guidance throughout the preparation of this report. We express our sincere gratitude to our guide, Prof. Dr. D. R. Patil sir for guiding us in investigations of these activities and in carrying out experimental work. Our numerous discussions were extremely helpful. We hold him in esteem for guidance, encouragement and inspiration received from him. They make us more studious and bolder. Thanks, lot, for their guidance. We enjoyed working with him and inspired lot from him.

Last but not the least we wish to thank our parents for financing our studies and helping us throughout for achieving perfection and excellence. Their personal help in making this report is gratefully acknowledged.

1. Mr. Joshi Shubham [3168]
2. Mr. Kandekar Shivam [3171]
3. Ms. Jadhav Kalyani [3159]

Contents

Abstract

People are always short on time in this extremely busy world. When it's time for a short break just to freshen up the mind, everyone prefers choosing from the options that are just a click away. I mean what could be better than a user being recommended with their favorite song to play, short movie to watch during a break. Yes, recommendation engines also learn your viewing patterns and may suggest content based on time of the day or your watching patterns. The recommender system helps users to find their preferred movies quickly on the home screen, without having to search from the extended content catalog.In today's world, we all use many platforms for entertainment, like youtube, in the initial stages. Further, going forward, many platforms emerged like Aha, Hotstar, Netflix, Amazon prime video, Zee5, Sony Liv, and many more. First, we will see a video or movie based on our interest by searching for the desired movie on the search engine. The recommendation system works here. The system will analyze the video or the movie which we have watched. Analysation may be based on the film genre, cast, director, music director, etc. Based on this analysis made by the recommendation system, we will be getting some recommendations for the next videos. which we have watched. Analysation may be based on the film genre, cast, director, music director, etc. Based on this analysis made by the recommendation system, we will be getting some recommendations for the next videos. For example, if a user listens to rock music every day, his youtube recommendation feed will get full of rock music and music of related genres. In this, items are ranked according to their relevancy and the most relevant ones are recommended to the user. A Recommender system helps to personalize a platform and help users find what they are looking for. From a business perspective, the more relevant content or movies a user finds on any particular platform, the higher their engagement and as a result increased revenue.

Chapter1

Introduction

1.1 Introduction

Recommendation systems are computer programs that suggest recommendations to users depending on a variety of criteria.

These systems estimate the most likely product that consumers will buy and that they will be interested in. Netflix, Amazon, and other companies use recommender systems to help their users find the right product or movie for them.

There are 3 types of recommendation systems.

1. **Demographic Filtering:** The recommendations are the same for every user. They are generalized, not personalized. These types of systems are behind sections like “Top Trending”.
2. **Content-based Filtering:** These suggest recommendations based on the item metadata (movie, product, song, etc). Here, the main idea is if a user likes an item, then the user will also like items similar to it.
3. **Collaboration-based Filtering:** These systems make recommendations by grouping the users with similar interests. For this system, metadata of the item is not required.

In this project, we are building a Content-based recommendation engine for movies. Scikit-learn is a powerful machine learning library in Python that provides various tools and algorithms for data analysis and modeling. While it is not specifically designed for recommendation systems, it can be used to build a simple recommendation model based on collaborative filtering. Collaborative filtering is a popular technique for building recommendation systems. It works by analyzing the behavior and preferences of users to identify patterns and make recommendations. In the case of a movie recommendation system, collaborative filtering looks at the historical movie ratings provided by users and finds similar users or movies based on their rating patterns.

1.2 Motivation

The motivation behind building a movie recommendation model using the Scikit-learn library in Python is to provide personalized recommendations to users and enhance their movie-watching experience. Here are some key motivations for developing such a model: Personalized User Experience: Movie recommendation systems aim to deliver personalized suggestions based on user preferences and behavior. By leveraging machine learning algorithms in Scikit-learn, we can analyze patterns in user data and provide tailored recommendations. This enhances the user experience by helping users discover new movies they are likely to enjoy and saves them time spent searching for movies manually.

1.3 Problem Definition

Develop a movie recommendation model using the scikit-learn library in python.The problem definition of a movie recommendation model using the Scikit-learn library in Python is to develop a system that can provide accurate and personalized movie recommendations to users based on their preferences and behavior. The goal is to create a model that can effectively analyze historical movie ratings and user data to identify patterns and similarities, and then generate recommendations for movies that the user is likely to enjoy.

1.4 Objectives

1. To develop in depth understanding and implementation of the key technologies in data science and big data analytics.
2. Improve User Satisfaction: The primary objective of a movie recommendation model is to enhance user satisfaction by providing personalized and relevant movie recommendations

1.5 Scope

Data Collection and Preprocessing: The model's scope includes gathering relevant data, such as movie ratings and user preferences, from various sources. This involves data cleaning, handling missing values, and transforming the data into a suitable format for analysis and model training.Safety Monitoring: Analyzing vaccine data can help identify and monitor any potential adverse events or side effects associated with COVID-19 vaccines.

Chapter 2

System Overview

2.1 Introduction of Libraries

1. **Numpy:**

   It is a very important library on which almost every data science or machine learning Python packages such as SciPy (Scientific Python), Mat−plotlib (plotting library), Scikit-learn, etc. depends on to a reasonable extent. NumPy is very useful for performing mathematical and logical operations on Arrays.
   e.g.: import numpy as np

2. **Pandas:**

   Pandas is a game-changer for data science and analytics, particularly if you came to Python because you were searching for something more powerful than Excel and VBA. Pandas uses fast, flexible, and expressive data structures designed to make working with relational or labeled data both easy and intuitive.
   e.g.: import pandas as pd

3. **Matplotlib:**

   Matplotlib is a Python library for data visualizations. It helps you to mostly plot 2- D dimensional graphs. It is built upon Numpy and Scipy framework. There are various types of graph plotting can be done using Matplotlib.
   The following graph plots can be done.

4. **Seaborn:**

   Seaborn helps to visualize the statistical relationships, To understand how variables in a dataset are related to one another and how that relationship is dependent on other variables, we perform statistical analysis. This Statistical analysis helps to visualize the trends and identify various patterns in the dataset.

Chapter 3

Software And Hardware Requirements

4.1 Software Requirements

1. Anaconda Navigator (Jupyter Notepad).
2. Google Colab.

4.2 Hardware Requirements

1. Operating System: - Windows 10
2. RAM: - 8GB
3. Hard Disk: - 1TB
4. Processor: - Intel core i3

Chapter 4

Theory

Movie recommendation systems are designed to suggest movies to users based on their preferences, historical data, and various algorithms. There are several theories and approaches behind these systems. Here are a few key theories and concepts that play a role in movie recommendation systems: Collaborative Filtering: Collaborative filtering is one of the fundamental theories behind recommendation systems. It suggests that users who have similar preferences in the past will have similar preferences in the future. In the context of movie recommendations, collaborative filtering analyzes user behavior, such as ratings or viewing history, to identify patterns and recommend movies that other users with similar tastes have enjoyed. This can be done using two main methods: User-based collaborative filtering: This approach recommends movies to a user based on the preferences of users with similar tastes. For example, if User A and User B have rated or watched similar movies, and User A enjoys a movie that User B has already seen but User A hasn't, the system may recommend that movie to User A Item-based collaborative filtering: This approach recommends movies based on the similarity between items (movies) themselves. It identifies movies that are similar to the ones a user has already enjoyed and recommends those similar movies. For example, if a user has rated or watched Movie A and Movie B is similar to Movie A based on certain features (genre, actors, directors, etc.), the system may recommend Movie B to the user Content-based Filtering: Content-based filtering focuses on the characteristics of movies and users' preferences to make recommendations. It recommends movies that are similar to the ones a user has already shown interest in based on shared features like genre, actors, directors, or keywords. For example, if a user has enjoyed several action movies in the past, the system may recommend other action movies that share similar characteristics. Hybrid Approaches: Many recommendation systems employ a combination of collaborative filtering and content-based filtering techniques to provide more accurate and diverse recommendations. By integrating both user behavior and movie attributes, hybrid approaches attempt to overcome the limitations of individual methods and offer more personalized and comprehensive recommendations.

Chapter 5

Analytics

Following Analytics performed for given dataset:

1. Describe the dataset:

In [1]:
```
import pandas as pd

# https://files.grouplens.org/datasets/movielens/ml-25m.zip
movies = pd.read_csv("movies.csv")
```

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

Out[2]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In [3]:
```
import re

def clean_title(title):
    title = re.sub("[^a-zA-Z0-9 ]", "", title)
    return title
```

In [4]:
```
movies["clean_title"] = movies["title"].apply(clean_title)
```

In [5]:
```
movies
```

Out[5]:

	movieId	title	genres	clean_title
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	Toy Story 1995
1	2	Jumanji (1995)	Adventure|Children|Fantasy	Jumanji 1995
2	3	Grumpier Old Men (1995)	Comedy|Romance	Grumpier Old Men 1995
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	Waiting to Exhale 1995
4	5	Father of the Bride Part II (1995)	Comedy	Father of the Bride Part II 1995
...	...	...	...	...
62418	209157	We (2018)	Drama	We 2018
62419	209159	Window of the Soul (2001)	Documentary	Window of the Soul 2001
62420	209163	Bad Poems (2018)	Comedy|Drama	Bad Poems 2018
62421	209169	A Girl Thing (2001)	(no genres listed)	A Girl Thing 2001
62422	209171	Women of Devil's Island (1962)	Action|Adventure|Drama	Women of Devils Island 1962

62423 rows × 4 columns

The dataset used in movie recommendation systems contains information about movies, users, and their interactions. The specific details and attributes of the dataset may vary depending on the source and purpose of the recommendation system. However, here are some common components typically found in a movie recommendation dataset: Movie Data: This includes information about individual movies such as title, genre, release year, director, actors, synopsis, runtime, language, and other relevant attributes. It may also include additional metadata like average ratings, popularity scores, box office earnings, and user-generated tags. User Data: User data consists of information about the users of the recommendation system. This may include unique user identifiers, demographic information (age, gender, location), user preferences, past viewing history, and explicit or implicit feedback such as ratings, reviews, watch history, bookmarks, likes, and dislikes. Ratings and Interactions: This component contains data on user ratings or explicit feedback given to movies. It includes user-movie interactions such as ratings on a numerical scale (e.g., 1-5 stars), reviews, comments, or other forms of user feedback. These ratings and interactions serve as valuable signals to train recommendation algorithms. Contextual Data: Some recommendation systems incorporate contextual information to provide more personalized recommendations. Contextual data may include the time of day, day of the week, location, device type, user behavior patterns, and other contextual factors that can influence movie preferences. Social Data: In social-based recommendation systems, the dataset may include information about users' social connections, friends, followers, and their activities within the platform. This social data helps identify users with similar tastes and facilitates collaborative filtering-based recommendations. Additional Metadata: Depending on the specific recommendation system, there may be additional metadata available, such as movie posters, trailers, production studio information, awards, and other relevant details that enhance the movie browsing and recommendation experience.

2. All Similar Movies:

Movie recommendation systems use various techniques to select similar movies based on user preferences and movie attributes. Here are a few common methods employed in selecting similar movies:

In [20]:
```
rec_percentages["score"] = rec_percentages["similar"] / rec_percentages["all"]
```

In [21]:
```
rec_percentages = rec_percentages.sort_values("score", ascending=False)
```

In [22]:
```
rec_percentages.head(10).merge(movies, left_index=True, right_on="movieId")
```

Out[22]:

	similar	all	score	movieId	title	genres	clean_title
17067	1.000000	0.040459	24.716368	89745	Avengers, The (2012)	Action|Adventure|Sci-Fi|IMAX	Avengers The 2012
20513	0.103711	0.005289	19.610199	106072	Thor: The Dark World (2013)	Action|Adventure|Fantasy|IMAX	Thor The Dark World 2013
25058	0.241054	0.012367	19.491770	122892	Avengers: Age of Ultron (2015)	Action|Adventure|Sci-Fi	Avengers Age of Ultron 2015
19678	0.216534	0.012119	17.867419	102125	Iron Man 3 (2013)	Action|Sci-Fi|Thriller|IMAX	Iron Man 3 2013
16725	0.215043	0.012052	17.843074	88140	Captain America: The First Avenger (2011)	Action|Adventure|Sci-Fi|Thriller|War	Captain America The First Avenger 2011
16312	0.175447	0.010142	17.299824	86332	Thor (2011)	Action|Adventure|Drama|Fantasy|IMAX	Thor 2011
21348	0.287608	0.016737	17.183667	110102	Captain America: The Winter Soldier (2014)	Action|Adventure|Sci-Fi|IMAX	Captain America The Winter Soldier 2014
25071	0.214049	0.012856	16.649399	122920	Captain America: Civil War (2016)	Action|Sci-Fi|Thriller	Captain America Civil War 2016
25061	0.136017	0.008573	15.865628	122900	Ant-Man (2015)	Action|Adventure|Sci-Fi	AntMan 2015
14628	0.242876	0.015517	15.651921	77561	Iron Man 2 (2010)	Action|Adventure|Sci-Fi|Thriller|IMAX	Iron Man 2 2010

In [8]:
```
# pip install ipywidgets
#jupyter labextension install @jupyter-widgets/jupyterlab-manager
```

In [9]:
```
import ipywidgets as widgets
from IPython.display import display

movie_input = widgets.Text(
    value='Toy Story',
    description='Movie Title:',
    disabled=False
)
movie_list = widgets.Output()

def on_type(data):
    with movie_list:
        movie_list.clear_output()
        title = data["new"]
        if len(title) > 5:
            display(search(title))

movie_input.observe(on_type, names='value')


display(movie_input, movie_list)
```

3.**Checking datasets datatypes:**

```
In [10]: movie_id = 89745

         #def find_similar_movies(movie_id):
         movie = movies[movies["movieId"] == movie_id]

In [11]: ratings = pd.read_csv("ratings.csv")

In [12]: ratings.dtypes

Out[12]: userId        int64
         movieId       int64
         rating      float64
         timestamp     int64
         dtype: object
```

In a movie recommendation system, various data types can be utilized to represent different aspects of the system. Here are some common data types used in movie recommendation systems:

1.Numerical Data: Numerical data types are used to represent quantitative information. For example, ratings given by users to movies can be represented as numerical values, such as a scale of 1 to 5 stars. Other numerical data can include movie popularity scores, box office earnings, runtime duration, and user demographic information like age or ratings given by a user.

2. Categorical Data: Categorical data types are used to represent discrete and non-numeric attributes. Examples include movie genres (e.g., action, comedy, drama), language, country of origin, and user gender. Categorical data is typically encoded using techniques such as one-hot encoding or integer encoding to make it suitable for machine learning algorithms.

3.Text Data: Text data types are used to represent textual information related to movies or user-generated content such as movie titles, movie synopses, user reviews, or movie tags. Text data is often preprocessed through techniques like tokenization, stemming, or word embeddings to extract meaningful features for recommendation algorithms.

4.Code for movie recommendation:

In [24]:

```
import ipywidgets as widgets
from IPython.display import display

movie_name_input = widgets.Text(
    value='Toy Story',
    description='Movie Title:',
    disabled=False
)
recommendation_list = widgets.Output()

def on_type(data):
    with recommendation_list:
        recommendation_list.clear_output()
        title = data["new"]
        if len(title) > 5:
            results = search(title)
            movie_id = results.iloc[0]["movieId"]
            display(find_similar_movies(movie_id))

movie_name_input.observe(on_type, names='value')

display(movie_name_input, recommendation_list)
```

Building a complete movie recommendation system involves implementing various algorithms and components, which is beyond the scope of a single code snippet. However, I can provide you with a simplified example of a content-based movie recommendation system using Python and scikit-learn. This example demonstrates how to recommend similar movies based on movie attributes using cosine similarity. Please note that this is a basic implementation and may not represent the complexity and sophistication of a real-world recommendation system.

we test the recommendation system by providing a movie title to recommend similar movies. The function recommend_similar_movies is called with the desired movie title, and the recommendations are printed.

2 Result of movie recommendation system:

In the simplified movie recommendation system example provided, the result would be a list of recommended movies that are similar to the given movie based on their genre. Here's an example of the potential result:

In [24]:

```
import ipywidgets as widgets
from IPython.display import display

movie_name_input = widgets.Text(
    value='Toy Story',
    description='Movie Title:',
    disabled=False
)
recommendation_list = widgets.Output()

def on_type(data):
    with recommendation_list:
        recommendation_list.clear_output()
        title = data["new"]
        if len(title) > 5:
            results = search(title)
            movie_id = results.iloc[0]["movieId"]
            display(find_similar_movies(movie_id))

movie_name_input.observe(on_type, names='value')

display(movie_name_input, recommendation_list)
```

Movie Title: Iron man

	score	title	genres
14628	64.863931	Iron Man 2 (2010)	Action|Adventure|Sci-Fi|Thriller|IMAX
19678	37.526367	Iron Man 3 (2013)	Action|Sci-Fi|Thriller|IMAX
20513	35.679273	Thor: The Dark World (2013)	Action|Adventure|Fantasy|IMAX
16312	30.352718	Thor (2011)	Action|Adventure|Drama|Fantasy|IMAX
16725	28.860481	Captain America: The First Avenger (2011)	Action|Adventure|Sci-Fi|Thriller|War
25058	26.437765	Avengers: Age of Ultron (2015)	Action|Adventure|Sci-Fi
13277	26.374521	X-Men Origins: Wolverine (2009)	Action|Sci-Fi|Thriller
18241	26.166699	Amazing Spider-Man, The (2012)	Action|Adventure|Sci-Fi|IMAX
12431	23.498873	Hancock (2008)	Action|Adventure|Comedy|Crime|Fantasy

A well-designed movie recommendation system can significantly improve movie discovery, increase user engagement, and enhance user satisfaction by presenting them with relevant and personalized movie suggestions.It's important to note that building an effective movie recommendation system requires a thorough understanding of data analysis, machine learning algorithms, and user behavior. Additionally, the system's performance can be further enhanced by incorporating additional data sources, such as movie reviews, social connections.

Chapter 6

Conclusion

In conclusion, a movie recommendation system is an effective tool that utilizes algorithms and user preferences to suggest relevant movies to individuals. These systems analyze various factors such as movie genres, ratings, user history, and other relevant data to generate personalized recommendations. Movie recommendation systems offer several benefits to both users and movie platforms. For users, they provide a convenient way to discover new movies based on their interests and preferences, saving time and effort in the movie selection process. These systems can introduce users to a wider range of movies they may not have otherwise come across.For movie platforms, recommendation systems help improve user engagement and satisfaction. By suggesting movies that align with users' tastes, these platforms can increase user retention, drive user activity, and enhance overall user experience. Additionally, recommendation systems can also assist in promoting less popular or niche movies, helping to diversify movie consumption patterns.

References

1. https://github.com/rashida048/Some-NLPProjects/blob/master/movie_dataset.csv
2. Ghuli, P., Ghosh, A., Shettar, R.: A collaborative filtering recommendation engine in a distributed environment. In: 2014 International Conference on Contemporary Computing and Informatics (IC3I). IEEE (2014)
3. Wakil, K., et al.: Improving web movie recommender system based on emotions. (IJACSA) Int.J. Adv. Comput. Sci. Appl. 6(2) (2015)

4. Books: -

   Data science and big data analytics

   Publication: Technical.

   To learn how to analytics the data and visualization data in this book, syntax and how to retrieve data.