Movie Recommendation System

Table of Contents

Abstract

Efficiently locating a favorite movie within a vast collection has emerged as a crucial challenge in the dissemination of information. Particularly in situations where users lack a specific movie preference, the significance of personalized recommendation systems becomes pronounced. Addressing this concern, we present a prototype of a movie recommendation system that aligns with the practical requisites of movie suggestions.We did this by using two filtering techniques: the Content-based and Collaborative filtering. These methods help the system understand what you might like and recommend movies based on that. We focused on making the system useful for real movie suggestions. We combined these methods to handle big movie collections and different user tastes.

Introduction

In today's world, where content is vast and diverse, finding the right movie to watch can be difficult. The Movie Recommendation System aims to simplify this task by offering personalized suggestions based on individual preferences.
At its core, this system utilizes advanced algorithms and data analytics techniques to analyze user interactions, movie ratings, and other relevant factors. By processing this data, the system can predict and recommend movies that align closely with a user's taste. With a conversation, the system can directly interact with users and quickly find out information about the wants and needs of its users. To be able to implement it requires a system that can do this, such as a chatbot.it gathers essential information about their movie preferences, such as genre, favorite actors, directors, previous movies they preferences .Using this data, the chatbot is integrated with movie recommendation system that accesses a vast movie database, applying advanced algorithms to find movies that best match the user's input. The result is a personalized list of movie suggestions presented to the user.

Modules and Software Required

- ➔ Python
- ➔ Platform:jupyter notebook,visualstudio code
- ➔ Numpy
- ➔ Pandas
- ➔ Matplotlib
- ➔ Scikit-learn
- ➔ Telebot

1.Study of Modules

1.1 Numpy

Definition

Numpy is a Python library used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices. It is known for its high-end performance with powerful N-dimensional array objects and the tools it is loaded with to work with arrays.

Numpy faster than Lists

- Numpy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently.
- This behavior is called locality of reference in computer science. This is the main reason why Numpy is faster than lists.
- Numpy aims to provide an array object that is up to 50x faster than traditional Python lists.

1.2 Pandas

Definition

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data.

Why do we use Pandas ?

- Pandas allows us to analyze big data and make conclusions based on statistical theories.
- Pandas can clean messy data sets, and make them readable and relevant. Relevant data is very important in data science.
- Pandas are also able to delete rows that are not relevant, or contain wrong values, like empty or NULL values. This is called cleaning the data.

1.3.Matplotlib

Definition

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multiplatform data visualization library built on Numpy arrays and designed to work with the broader Scipy stack.

Pyplot

Most of the Matplotlib utilities lies under the Pyplot submodule, and are usually imported under the plt alias:

➔ **import matplotlib.pyplot as plt** Now
the Pyplot can be referred to as plt.

Matplotlib Plotting

We can plot various graphs using pyplot:-

- Bar graph : With Pyplot, you can use the bar ( ) function to draw bar graphs. The bar ( ) function takes arguments that describe the layout of the bars.
- Histogram : In Matplotlib, we use the hist ( ) function to create histograms.
- Pie Charts : We can use the pie ( ) function to draw pie charts.

2.Machine Learning

Supervised Learning

Supervised learning is a type of machine learning where the algorithm learns from labeled training data. In supervised learning, you provide the algorithm with a set of input-output pairs (data points along with their corresponding labels) to learn a mapping from inputs to outputs. The goal is to learn a function that can accurately predict the output for new, unseen inputs.

Unsupervised Learning

Unsupervised learning is a type of machine learning where the algorithm learns patterns from unlabeled data. Unlike supervised learning, there are no predefined output labels provided. The goal of unsupervised learning is to discover the underlying structure, relationships, and patterns within the data.

Cost Function

A cost function is a measure of how well a machine learning model performs. It quantifies the error between predicted and expected values and presents that error in the form of a single real number. The cost function is also called the loss function.
For eg→ Here we have used Squared error cost function

$$\text{Cost Function} \qquad J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$$

Gradient Descent

Gradient Descent is an algorithm to determine the minimum value of a cost function J(w,b) at corresponding w and b values. Here we change the values of w and b until we get the minimum value of the Cost function J(w,b). Learning rate is denoted by ‘a’.
w= w-a*d/dw(J(w,b))
b= b-a*d/d(J(w,b))

Linear Regression

Linear regression is a statistical method used for modeling the relationship between a dependent variable denoted as ‘Y’ and one or more independent variables denoted as ‘X’.
The goal of linear regression is to find the best-fitting straight line that minimizes the difference between the predicted values and the actual data points.
Y=w*X+b
(Here w is slope of the line and b is the y-intercept)

Multiple Linear Regression

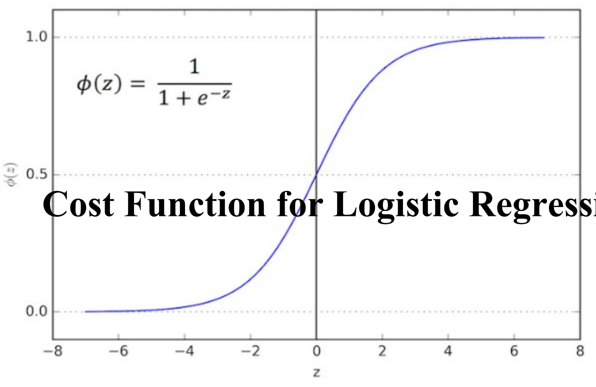
This is a Linear Regression model where there are multiple input features. Here, each input feature has a corresponding weight, and these all can be represented as vectors. So here we use the Vectorization method.
Y vector = (W vector) *(X vector) + b

Logistic Regression

Logistic Regression is a predictive analysis technique used for classification problems. It is a statistical method for predicting binary classes.
The outcome or target variable is dichotomous in nature, meaning there are two possible outcomes: yes or no, true or false, success or failure etc.

Logistic Function

The logistic function, also known as the sigmoid function, is an S-shaped curve that maps any real-valued number to a value between 0 and 1.



S(z) = 1 / (1 + e^-z)

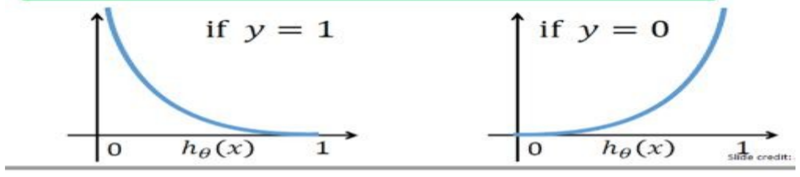
0<s(z)<1

Z=w.x+b

Cost Function for Logistic Regression

Cost function for logistic regression

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

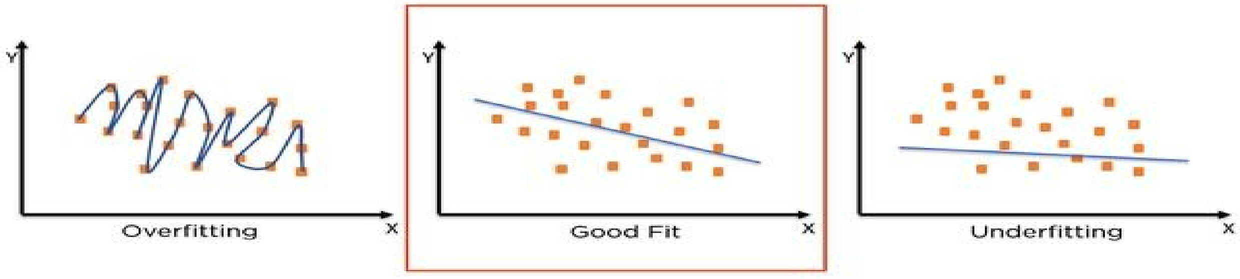


Overfitting:
It occurs when a model is trained too well on the training data. Consequently, while it performs excellently on the training data, it struggles to generalize to new, unseen data, resulting in poor performance on the test data or in real-world applications.

Underfitting:
It fails even to fit the training data well and as a result, performs poorly on the training data and the test data. The model might be underfitting because it's too simplistic, or because it hasn't been trained for long enough to learn the underlying patterns in the data.

Regularization

Regularization is a technique used in machine learning and statistical modeling to prevent overfitting and improve the generalization performance of a model.This methods typically introduce a penalty term to the loss function during model training. This penalty discourages the model from fitting the training data too closely and instead encourages it to find simpler, more general patterns that can be applied to new, unseen data.



3.Reinforcement Learning

Reinforcement Learning (RL) is the science of decision making. It is about learning the optimal behavior in an environment to obtain maximum reward. In RL, the data is accumulated from machine learning systems that use a trial-and-error method. Data is not part of the input that we would find in supervised or unsupervised machine learning.x

Reinforcement learning uses algorithms that learn from outcomes and decide which action to take next. After each action, the algorithm receives feedback that helps it determine whether the choice it made was correct, neutral or incorrect. It is a good technique to use for automated systems that have to make a lot of small decisions without human guidance.

4.Recommendation Filtering Techniques

4.1.Collaborative Filtering

Collaborative filtering is based on the idea that users who liked it in the past tend to like it again in the future. It relies on user-item interaction data, such as user ratings and reviews.

- **User-Based Collaborative Filtering**

This approach recommends movies to a user based on the preferences and behavior of users with similar tastes. It identifies users who have rated movies similarly to the target user and recommends movies they have liked but the target user hasn't seen yet.

- **Item-Based Collaborative Filtering**

Instead of comparing users, this method identifies movies that are similar to the ones the user has liked. It recommends items that are similar to the user's previous preferences.

- **Matrix Factorization**

This technique decomposes the user-item interaction matrix into latent factors, which represent underlying features of users and items. It helps in identifying hidden patterns and provides recommendations based on these latent factors.

- **Singular Value Decomposition (SVD)**

SVD is a matrix factorization technique that reduces the dimensionality of the user-item interaction matrix. It helps in finding latent features and making recommendations based on them.

4.2.Content-Based Filtering

Content-based filtering suggests movies to users based on the attributes and characteristics of the items they have previously shown interest in. These attributes could include genres, actors, directors, plot keywords, and more.

- **TF-IDF (Term Frequency-Inverse Document Frequency)**

This technique represents movies as vectors based on the presence and importance of words (terms) in their descriptions. It can be used to match user preferences with movie attributes.

- **Feature Extraction and Vectorization**

Machine learning techniques can be used to extract features from movie attributes like genres, cast, crew, and plot summaries. These features can then be used to create vectors for similarity calculations.

4.3.Hybrid Methods

Hybrid methods combine both collaborative and content-based approaches to provide more accurate recommendations.

- **Weighted Hybrid**

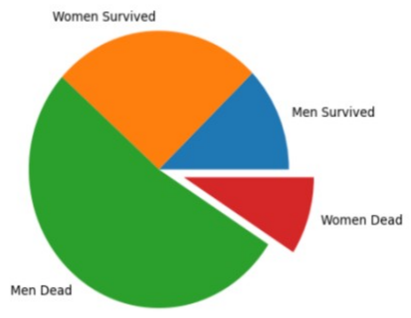
This method combines recommendations from collaborative and content-based filters by assigning different weights to each approach based on their performance or user preferences.

- **Switching Hybrid**

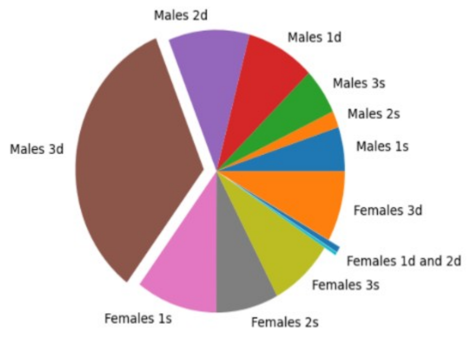
Recommendations from collaborative and content-based filters are presented separately, and users can switch between them based on their current preferences.

5.Tasks

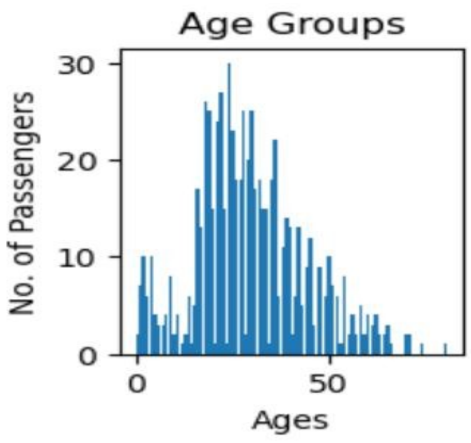
5.1.Analysing Titanic Dataset



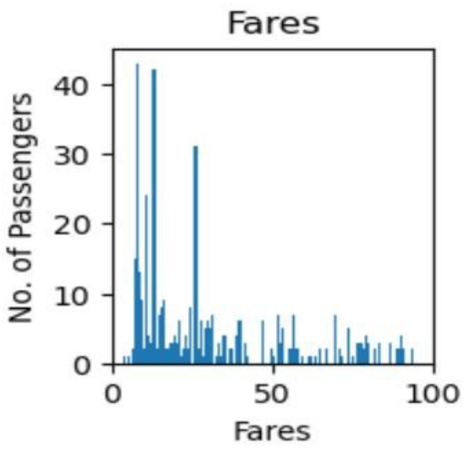
Pie chart showing gender wise survival status



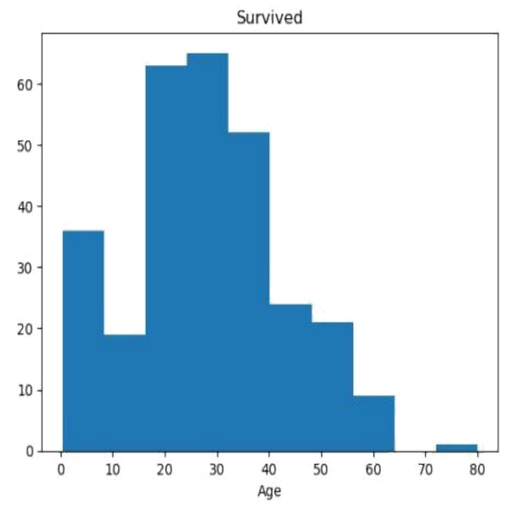
Pie chart showing class wise and gender wise survival status



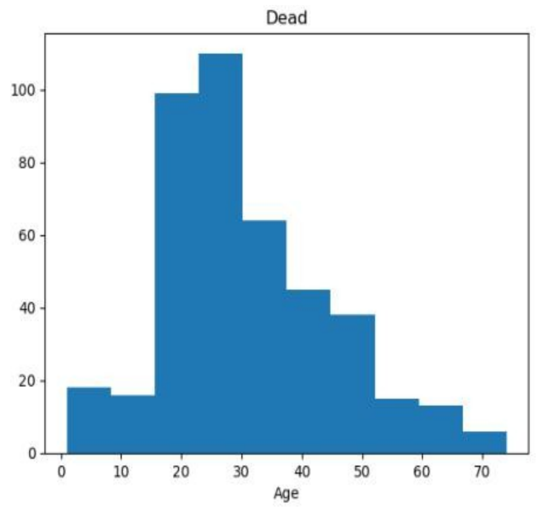
Age wise distribution using Bar graph



Bar graph showing distribution of fares



Histogram showing age wise
distribution of survived passengers



Histogram showing age wise
distribution of dead passengers

5.2.Making Movie Recommendation System Model

Dataset used : The Movies Dataset | Kaggle

Data Cleaning

As Data cleaning is used to improve quality and reliability of the dataset.We handled following errors :
1. Missing values: This may occur due to various reasons like entry errors,incomplete records,etc.We dropped such rows and columns containing NULL values.
2.Converting Data types: Ensuring that variables have appropriate data types is important for analysis and modeling.eg. In this model we converted the data type of movie id to string.
3.Standardizing Data to the same format: Removing spaces between words,converting all text to lowercase,etc. comes under this category.This brought uniformity and made comparison tasks easy.

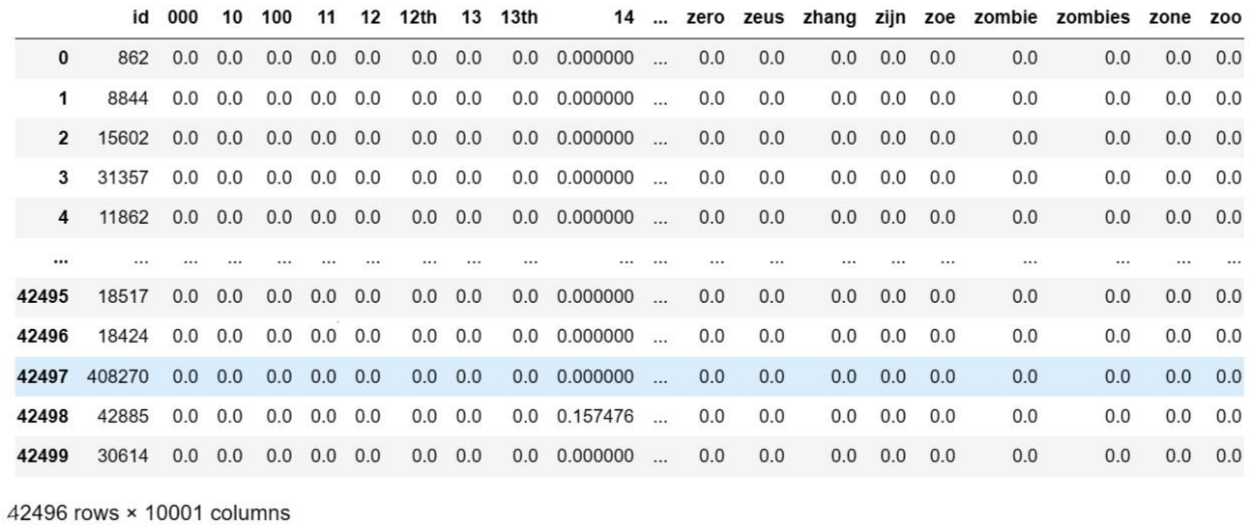
5.2.1 Content-Based Filtering

Extraction of Data

We collected various data points, including actors, directors, and keywords, from separate CSV files. To ensure consistent data handling, we converted the data types of directors and overviews into strings. As part of our data processing, we streamlined the information by combining genres, cast, overviews, and keywords into a single column named 'tags.' This consolidation enhances the efficiency of data analysis.

Applying TF-IDF Vectoriser

We applied tfidf vectoriser on column named ‘tags’ in content-based filtering.



	id	000	10	100	11	12	12th	13	13th	14	...	zero	zeus	zhang	zijn	zoe	zombie	zombies	zone	zoo
0	862	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	8844	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	15602	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	31357	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	11862	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
42495	18517	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
42496	18424	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
42497	408270	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
42498	42885	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.157476	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
42499	30614	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

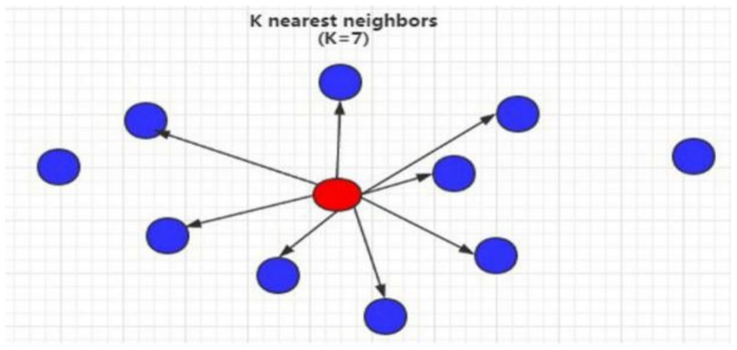
42496 rows × 10001 columns

Centroid and Euclidean Distances

We retrieve the relevant rows based on the user's input and then calculate the mean for each column to derive the centroid vector. Euclidean distances are often used as a similarity measure to compare the distances between data points.So, here we applied it to find the distances between movies and the centroid.

K-Nearest Neighbors (KNN) Technique

After finding the distances and selecting a value for k (usually large to prevent noise) , we applied the KNN algorithm to find the nearest movies.The predictions are made by the kNN algorithm for each input data point based on the nearest neighbors found.



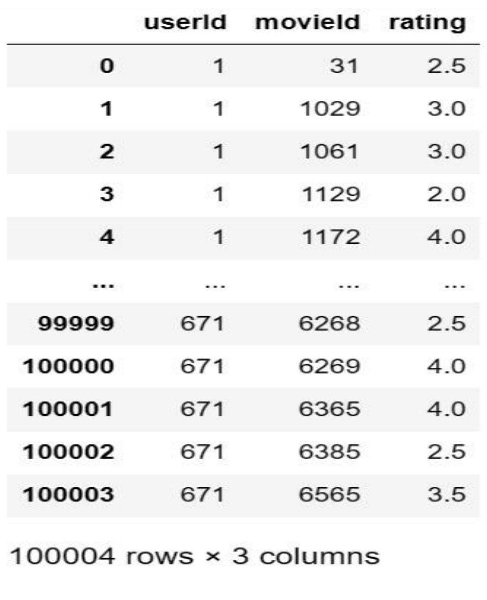
For input : "Harry Potter and the Prisoner of Azkaban,Harry Potter and the Goblet of Fire"
We get output as :

```
['Harry Potter and the Prisoner of Azkaban', 'Harry Potter and the Goblet of Fire', 'Harry Potter and
the Chamber of Secrets', 'Harry Potter and the Order of the Phoenix', 'Harry Potter and the Deathly H
allows: Part 2', "Harry Potter and the Philosopher's Stone", 'Harry Potter and the Half-Blood Princ
e', 'Harry Potter and the Deathly Hallows: Part 1', "Harry, He's Here To Help", 'Willow', 'The Kidnap
pers']
```

5.2.2 Collaborative Filtering

Extraction of Data

We sourced ratings and movie data from CSV files. Movie IDs were transformed into titles using a mapped dictionary.



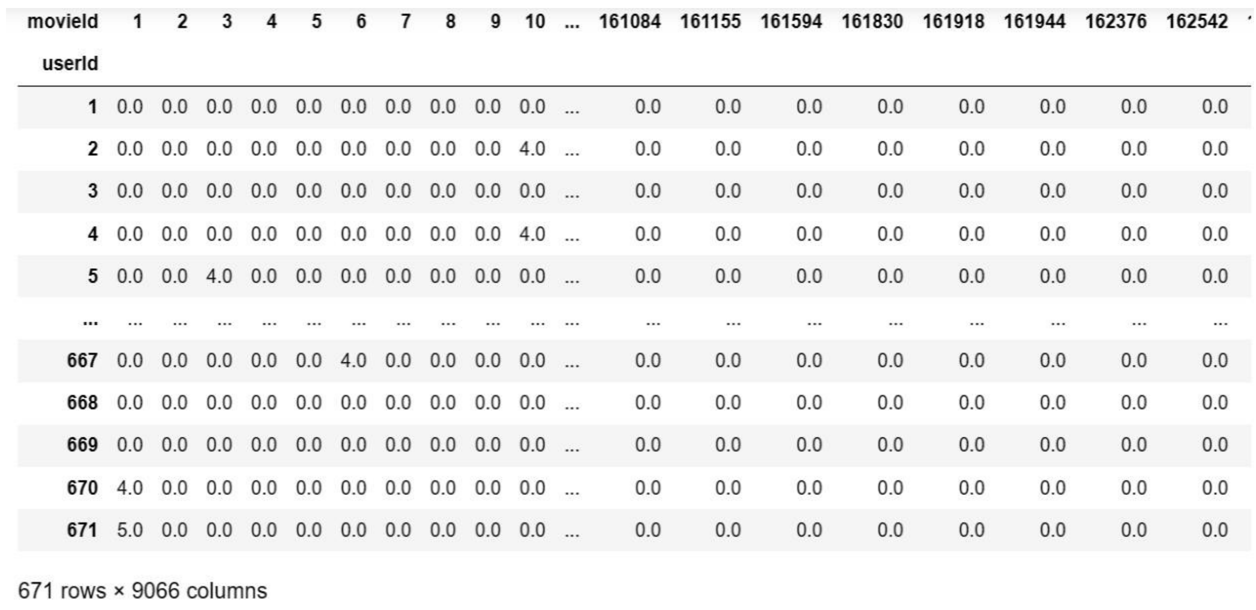
	userId	movieId	rating
0	1	31	2.5
1	1	1029	3.0
2	1	1061	3.0
3	1	1129	2.0
4	1	1172	4.0
...	...	...	...
99999	671	6268	2.5
100000	671	6269	4.0
100001	671	6365	4.0
100002	671	6385	2.5
100003	671	6565	3.5

100004 rows × 3 columns

Applying cosine similarity

We created a pivot table with user IDs, movie IDs, and ratings to understand how users rate different movies. Using cosine similarity, we measured the likeness of highly-rated users' preferences. By comparing their ratings, we identified movies they haven't seen. These unseen movies were then suggested as personalized recommendations, enhancing their movie choices.

Pivot Table:

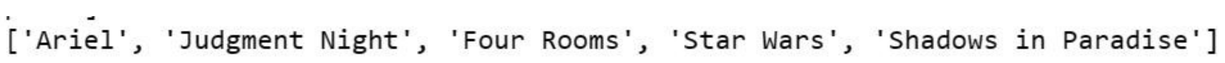


movieId	1	2	3	4	5	6	7	8	9	10	...	161084	161155	161594	161830	161918	161944	162376	162542
userId																			
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	4.0	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
667	0.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
668	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
669	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
670	4.0	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
671	5.0	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

671 rows × 9066 columns

Recommendation

Incoming inputs via the chatbot are captured and then channeled into distinct recommendation functions, each catering to specific needs. Depending on the nature of the input, the appropriate recommendation function is invoked. The results generated by these functions are subsequently displayed. This dynamic process ensures that users receive customized recommendations aligned with their queries, enhancing their interaction with the chatbot.
ex. For input : "Harry Potter and the Prisoner of Azkaban,Harry Potter and the Goblet of Fire"
We get output as:



```
['Ariel', 'Judgment Night', 'Four Rooms', 'Star Wars', 'Shadows in Paradise']
```

5.3.Model Fusion Chatbot

We utilized BotFather, a specialized service within Telegram, to facilitate the implementation of our model through a chatbot. This service enables the creation and administration of personalized bots tailored to our needs. By interacting with BotFather, we established a bot that aligns with our requirements.

- **How did we proceed……**

1.Initiating conversation:

When you send the command ‘ /start’ , the chatbot responds with a warm welcome. This friendly interaction is made possible using a function called ‘ message_handler()’ . It's as if the chatbot is politely greeting you as you step into its virtual space.

2.Getting Movie Recommendations:

If you type ‘/recommend’ the chatbot takes on the role of a helpful movie guide.It’s like a knowledgeable friend who suggests movies for you.It all starts by asking your favorite movie genre.This is managed through another ‘message_handler()’ function, designed to respond to the command ‘/recommend’.
To make the process interactive,the chatbot presents you with options to choose from.The ‘markup.add()’ function plays a role in offering these options, making it easy to pick your favorite genre and preferred language for movies.
After you choose a genre, the register_next_step_handler() function steps in. It's like the chatbot knows to guide you to the next part of the conversation. Depending on your genre choice, different functions like get_language, get_actor, get_director, and get_titles are activated, prompting you to provide further details.

3. Diverse functions :

To make the chatbot efficient, we've defined various functions that correspond to each step of gathering your preferences. For example, the get_actor function captures your favorite actor's name, get_director captures your preferred director, and get_titles takes note of your top movie titles.These functions store your inputs and prepare them for the final recommendation step.

6. Errors and Solutions

Problems faced while building movie recommendation system:

In movies_metadata the column id is in string format and contained some irrelevant values of date format

Solution:

That corresponding date value is shown in error message so find that row and drop that row
Ex: The Error message showed "2014-01-01" so find its index and drop that row
print(movies["id"].index[movies["id"]=="2014-01-01"].tolist()) It prints 35587

movies=movies.drop(35587)

Problems faced while building chatbot:

- A proper program flow was not seen initially ,like in other programming languages.
- Storing answers given by the user
- Removing the Button Keyword after options are selected

Solutions:

- Using ‘bot.register_next_step_handler()’helped maintain proper flow
- The string had to be sliced to remove the unwanted parameters
- ‘type.ReplyKeyboardRemove(selective=False)’helped to remove the Button Keyword

7.Timeline

---

12 July
Learning modules such as Numpy,Pandas,Matplotlib

17 July
Matplotlib Data Visualisation on Titanic dataset.

20 July
Learning the Machine Learning Basics

25 July
Dataset Finalization(The Movies Dataset) and Data Cleaning and Preprocessing.

31 July
Research on recommendation techniques by analyzing research papers

3 August
Finalization of architecture.

6 August
Group Presentation in Seminar hall of Electrical Department.

8 August
Divided team for the implementation of movie recommendations and making of chatbot.

10 August
Implementation done till content based, Collaborative and working for integrating with chatbot.

11 August
Decided to make a telebot.

13 August
Making the introductory part for chatbot.

14 August
Working on integration of chatbot and model and solved error getting in integration.

16 August
Integration of KNN in a content based model.

18 August
Working on the documentation part and resolving the error in the model.

8. Conclusion

In conclusion, the Movie Recommendation System project, developed using the Telebot framework along with K-Nearest Neighbors (KNN), TF-IDF techniques, successfully addresses the challenge of providing personalized movie recommendations to users. This system leverages various components and methodologies to enhance the user experience and deliver accurate and relevant movie suggestions. By combining collaborative and content-based filtering techniques in a hybrid approach, the system addresses the challenges of diversity and accuracy in recommendations. By taking user preferences such as genre, past watched movies, actors, directors, and utilizing advanced techniques, the system generates relevant suggestions. The integration of Telebot as the user interface streamlines the interaction, making it user-friendly. The KNN algorithm identifies movies similar to user choices.Additionally, cosine similarity aids in measuring textual similarity for better genre, actor, and director-based recommendations. This system achieves its goal of delivering personalized movie suggestions, enriching user engagement and entertainment experiences.

9. References and Learning Resources

Machine learning Specialization:
https://www.youtube.com/playlist?list=PLkDaE6sCZn6FNC6YRfRQc_FbeQrF8BwGI

The Movies Dataset:
https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset

Research papers:
https://www.researchgate.net/publication/283042228_A_Movie_Recommender_System_MOVREC