

MOVIE RECOMMENDATION SYSTEM

P.RUPA : 21B01A05E5 : CSE

SK.RIZWANA : 21B01A05G8 : CSE

T.DEEPIKA : 21B01A05I1 : CSE

V.CHETANA SRI : 21B01A05J3 : CSE

February 9,2024

Introduction

Discovering movies aligned with our unique interests can be a difficult task. Navigating through countless options is time-consuming. Our ML project tackles these challenges. By using machine learning algorithms, we can analyze individual preferences, and provide recommendations that resonate with the user's cinematic taste.

Movies based on genre



Figure: projects

Horror



Figure: projects

Anime



Figure: projects

Approach

The core focus lies in harnessing the power of predictive modeling, enabling us to discern and understand user preferences comprehensively. By leveraging these algorithms, we aim to revolutionize the movie recommendation landscape, presenting users with precisely tailored suggestions that align seamlessly with their unique tastes. This meticulous approach ensures a sophisticated and personalized cinematic experience, eliminating the challenges associated with traditional, broad-stroke recommendations.

Dataset Description

Columns: id, overview ,title, genre

Combined overview and genre to obtain tag column

Learnings

- **Libraries used**

sklearn, numpy, pandas ,pickle

streamlit(<http://localhost:8501>)

loading data set using kaggle API , resolving errors,
finding suitable algorithm , collaborating front end
and back end using streamlit.

Challenges

- We felt difficult in writing the machine learning code and searching for appropriate technique for our project ,connecting the application with our Python code .But finally, we cracked the logic and finished the code with in the time.

Technique implementation

- Choosing count vectorization for a content-based recommendation system ,Count vectorization is a straightforward technique that converts text documents into vectors of term counts.
- Simplicity and Efficiency
- Count vectorization can handle noisy text data reasonably well.
- Count vectors can be easily compared using similarity measures such as cosine similarity or Euclidean distance. This makes it straightforward to find similar documents or items for recommendation purposes.
- Count vectorization produces sparse vectors where each dimension corresponds to a unique term in the vocabulary.

Demo/Screenshots

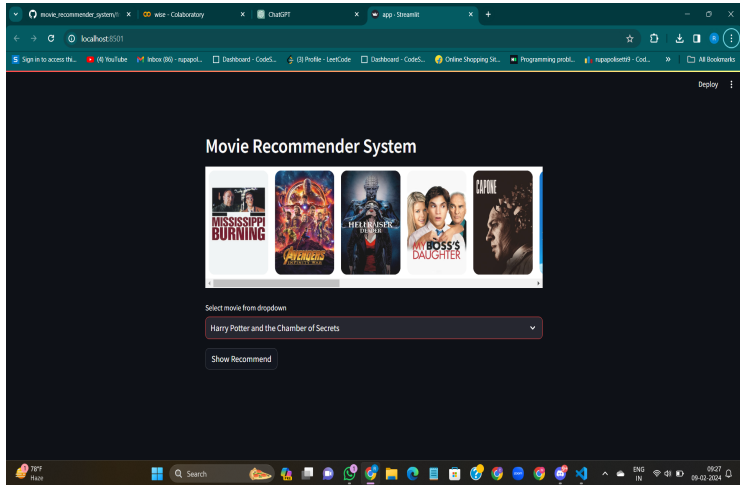


Figure: projects

Demo/Screenshots

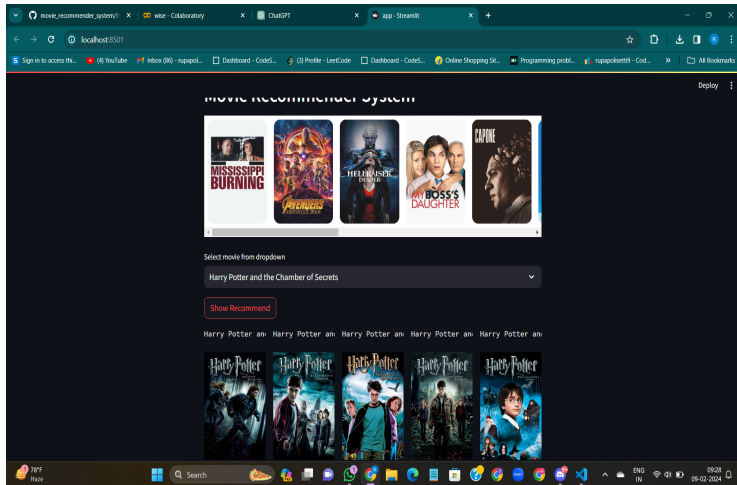
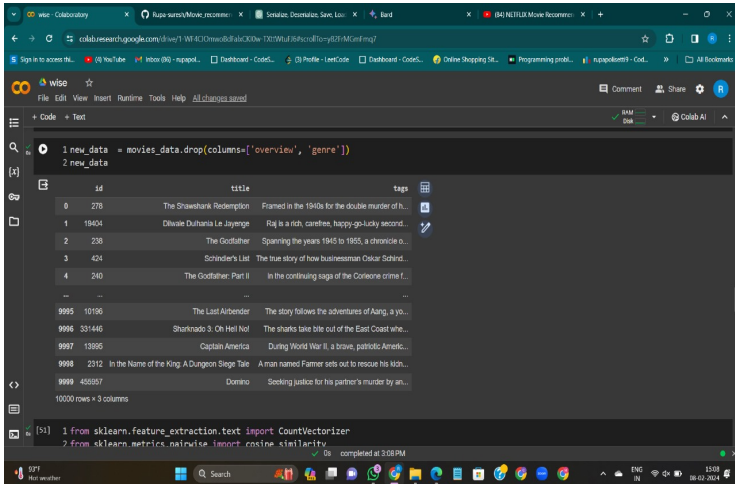


Figure: projects

Demo/Screenshots



The screenshot shows a Google Colab notebook with the following content:

```
1 new_data = movies_data.drop(columns=['overview', 'genre'])
2 new_data
```

	id	title	tags
0	278	The Shawshank Redemption	Framed in the 1940s for the double murder of h...
1	19404	Dilwale Duhaiya Le Jayenge	Raj is a rich, carefree, happy-go-lucky second...
2	238	The Godfather	Spanning the years 1945 to 1955, a chronicle o...
3	424	Schindler's List	The true story of how businessman Oskar Schind...
4	240	The Godfather: Part II	In the continuing saga of the Corleone crime f...
...
9995	10196	The Last Airbender	The story follows the adventures of Aang, a yo...
9996	331446	Sharknado 3: Oh Hell No!	The sharks take bite out of the East Coast whe...
9997	13895	Captain America	During World War II, a brave, patriotic Americ...
9998	2312	In the Name of the King: A Dungeon Siege Tale	A man named Farmer sets out to rescue his kidn...
9999	456957	Domino	Seeking justice for his partner's murder by an...

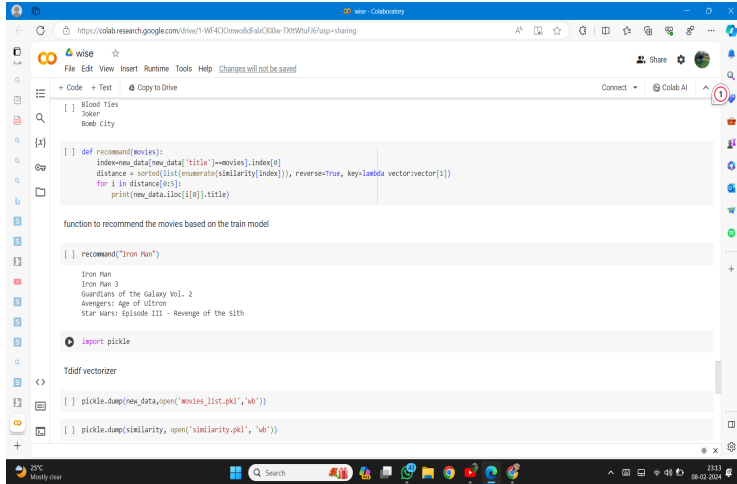
```
1 from sklearn.feature_extraction.text import CountVectorizer
2 from sklearn.metrics.pairwise import cosine_similarity
```

10000 rows x 3 columns

Completed at 3:08 PM

Figure: projects

Demo/Screenshots



The screenshot shows a Google Colab notebook interface. The browser address bar displays the URL: <https://colab.research.google.com/drive/1-WF4O0mwo8dFalsCKDw-TXttWuF6f?usp=sharing>. The notebook is titled "wise" and has a status bar indicating "Changes will not be saved". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with options like "+ Code", "+ Text", and "Copy to Drive".

The notebook content is as follows:

```
[ ] Blood Ties
    Joker
    Bomb City
```

```
[ ] def recommend(movies):
    index=new_data[new_data['title']==movies].index[0]
    distance = sorted(list(enumerate(similarity[index])), reverse=True, key=lambda vector:vector[1])
    for i in distance[0:5]:
        print(new_data.iloc[i[0]].title)
```

function to recommend the movies based on the train model

```
[ ] recommend("Iron Man")
```

Output:

```
Iron Man
Iron Man 3
Guardians of the Galaxy Vol. 2
Avengers: Age of Ultron
Star Wars: Episode III - Revenge of the Sith
```

```
import pickle

Tfidf vectorizer

[ ] pickle.dump(new_data,open('movies_list.pkl','wb'))

[ ] pickle.dump(similarity, open('similarity.pkl', 'wb'))
```

The bottom of the image shows a Windows taskbar with the date and time: 23:13, 06-02-2024.

Figure: projects

Conclusion

- user can easily know the movies he can watch on his interest.
- like traditional methods there will not be any waste of time in searching for reviews on some other sites.
- Machine learning models make the work of user easy and interactive.

Thank You