Naan Mudhalvan Project - 2024

IMDB Movies Dataset - Recommendation Engine

Name: Logith VL

Reg no: 2021503710

Introduction:

In the realm of online entertainment, platforms like IMDb play a pivotal role in helping users discover movies and television shows based on their preferences. Content-based filtering is a recommendation technique that leverages the attributes of items (in this case, movies and TV shows) to recommend similar items to users. By analyzing the content and characteristics of movies, content-based filtering can provide personalized recommendations tailored to each user's tastes.

Problem Statement:

The problem we aim to address is the overwhelming amount of content available on platforms like IMDb, which can make it challenging for users to discover new movies and television shows that align with their interests. Traditional browsing methods may not be efficient, especially when users are unsure about what they want to watch next. Additionally, relying solely on user ratings or popularity metrics may not always lead to satisfying recommendations, as individual preferences can vary widely.

Objective:

Our objective is to develop a content-based filtering system for IMDb that recommends movies and television shows to users based on the content and characteristics of the items themselves. By analyzing features such as genre, cast, director, plot keywords, and user tags, we aim to create a personalized recommendation engine that suggests relevant content to each user. The goal is to enhance user engagement and satisfaction by providing tailored recommendations that align with their preferences.

IMDB Movies Dataset - Recommendation Engine using Machine Learning:

```
from google.colab import files
upload=files.upload()
```

```
Choose Files  No file chosen    Upload widget is only available when the cell has been executed in the current browser session.
Saving IMDB_Top250Engmovies2_OMDB_Detailed.csv to IMDB_Top250Engmovies2_OMDB_Detailed.csv
```

```
   Unnamed: 0                     Title  Year     Rated   Released  Runtime  \
0           1  The Shawshank Redemption  1994         R  14-Oct-94  142 min
1           2             The Godfather  1972         R  24-Mar-72  175 min
2           3    The Godfather: Part II  1974         R  20-Dec-74  202 min
3           4           The Dark Knight  2008     PG-13  18-Jul-08  152 min
4           5              12 Angry Men  1957  APPROVED  01-Apr-57   96 min

                 Genre              Director  \
0         Crime, Drama        Frank Darabont
1         Crime, Drama  Francis Ford Coppola
2         Crime, Drama  Francis Ford Coppola
3  Action, Crime, Drama     Christopher Nolan
4         Crime, Drama          Sidney Lumet

                                              Writer  \
0  Stephen King (short story "Rita Hayworth and S...
1  Mario Puzo (screenplay), Francis Ford Coppola ...
2  Francis Ford Coppola (screenplay), Mario Puzo ...
3  Jonathan Nolan (screenplay), Christopher Nolan...
4  Reginald Rose (story), Reginald Rose (screenplay)
```

```
[ ] import pandas as pd
    import numpy as np
    import re
    import nltk
    pd.set_option('display.max_columns', None)
```

```
[ ] len(df)

    250
```

Data Preprocessing

```
[ ] df['clean_plot'] = df['Plot'].str.lower()
    df['clean_plot'] = df['clean_plot'].apply(lambda x: re.sub('[^a-zA-Z]', ' ', x))
    df['clean_plot'] = df['clean_plot'].apply(lambda x: re.sub('\s+', ' ', x))
    df['clean_plot']

    0      two imprisoned men bond over a number of years...
    1      the aging patriarch of an organized crime dyna...
    2      the early life and career of vito corleone in ...
    3      when the menace known as the joker emerges fro...
    4      a jury holdout attempts to prevent a miscarria...
                                 ...
    245    the desperate life of a chronic alcoholic is f...
    246    a something supervising staff member of a resi...
    247    a newspaper editor uses every trick in the boo...
    248    an old man makes a long journey by lawn mover ...
```

```
[ ] import nltk

    # Download the punkt tokenizer resource
    nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
[ ] df['clean_plot'] = df['clean_plot'].apply(lambda x: nltk.word_tokenize(x))
    df['clean_plot']
```

```
0      [two, imprisoned, men, bond, over, a, number, ...
1      [the, aging, patriarch, of, an, organized, cri...
2      [the, early, life, and, career, of, vito, corl...
3      [when, the, menace, known, as, the, joker, eme...
4      [a, jury, holdout, attempts, to, prevent, a, m...
                             ...
245    [the, desperate, life, of, a, chronic, alcohol...
246    [a, something, supervising, staff, member, of,...
247    [a, newspaper, editor, uses, every, trick, in,...
248    [an, old, man, makes, a, long, journey, by, la...
249    [a, mumbai, teen, reflects, on, his, upbringin...
Name: clean_plot, Length: 250, dtype: object
```

```
[ ] import nltk

    # Download the stopwords corpus
    nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
[ ] stop_words = nltk.corpus.stopwords.words('english')
    plot = []
    for sentence in df['clean_plot']:
        temp = []
        for word in sentence:
            if word not in stop_words and len(word) >= 3:
                temp.append(word)
        plot.append(temp)
    plot
```

```
[ ] plot
```

```
[['two',
  'imprisoned',
  'men',
  'bond',
  'number',
  'years',
  'finding',
  'solace',
  'eventual',
  'redemption',
  'acts',
  'common',
  'decency'],
 ['aging',
  'patriarch',
  'organized',
  'crime',
  'dynasty',
  'transfers',
  'control',
  'clandestine',
  'empire',
  'reluctant',
  'son'],
 ['early',
  'life',
  'career',
  'vito',
  'corleone',
  'new',
  'york',
```

```
[ ]  df['clean_plot'] = plot
```

```
[ ]  df['clean_plot']
```

```
0      [two, imprisoned, men, bond, number, years, fi...
1      [aging, patriarch, organized, crime, dynasty, ...
2      [early, life, career, vito, corleone, new, yor...
3      [menace, known, joker, emerges, mysterious, pa...
4      [jury, holdout, attempts, prevent, miscarriage...
                             ...
245    [desperate, life, chronic, alcoholic, followed...
246    [something, supervising, staff, member, reside...
247    [newspaper, editor, uses, every, trick, book, ...
248    [old, man, makes, long, journey, lawn, mover, ...
249    [mumbai, teen, reflects, upbringing, slums, ac...
Name: clean_plot, Length: 250, dtype: object
```

```
[ ]  df.head()
```

[ ]

	Unnamed: 0	Title	Year	Rated	Released	Runtime	Genre	Director	Writer	Actors	Plot	Language	Country	Awards	Poster	Ratings.Source	Rati
0	1	The Shawshank Redemption	1994	R	14-Oct-94	142 min	Crime, Drama	Frank Darabont	Stephen King (short story "Rita Hayworth and S...	Tim Robbins, Morgan Freeman, Bob Gunton, Willi...	Two imprisoned men bond over a number of years...	English	USA	Nominated for 7 Oscars. Another 19 wins & 30 n...	https://images-na.ssl-images-amazon.com/images...	Internet Movie Database	
1	2	The Godfather	1972	R	24-Mar-72	175 min	Crime, Drama	Francis Ford Coppola	Mario Puzo (screenplay), Francis Ford Coppola ...	Marlon Brando, Al Pacino, James Caan, Richard ...	The aging patriarch of an organized crime dyna...	English, Italian, Latin	USA	Won 3 Oscars. Another 23 wins & 27 nominations.	https://images-na.ssl-images-amazon.com/images...	Internet Movie Database	
2	3	The Godfather: Part II	1974	R	20-Dec-74	202 min	Crime, Drama	Francis Ford Coppola	Francis Ford Coppola (screenplay), Mario Puzo ...	Al Pacino, Robert Duvall, Diane Keaton, Robert...	The early life and career of Vito Corleone in ...	English, Italian, Spanish, Latin, Sicilian	USA	Won 6 Oscars. Another 10 wins & 20 nominations.	https://images-na.ssl-images-amazon.com/images...	Internet Movie Database	
3	4	The Dark Knight	2008	PG-13	18-Jul-08	152 min	Action, Crime, Drama	Christopher Nolan	Jonathan Nolan (screenplay), Christopher	Christian Bale, Heath Ledger, Aaron	When the menace known as the Joker	English, Mandarin	USA, UK	Won 2 Oscars. Another 151 wins & 153	https://images-na.ssl-images-amazon.com/images...	Internet Movie Database	

```
[ ]  df['Genre'] = df['Genre'].apply(lambda x: x.split(','))
     df['Actors'] = df['Actors'].apply(lambda x: x.split(',')[:4])
     df['Director'] = df['Director'].apply(lambda x: x.split(','))
```

```
[ ]  df['Actors'][0]
```

```
['Tim Robbins', ' Morgan Freeman', ' Bob Gunton', ' William Sadler']
```

```
[ ]  def clean(sentence):
         temp = []
         for word in sentence:
             temp.append(word.lower().replace(' ', ''))
         return temp
```

```
[ ]  df['Genre'] = [clean(x) for x in df['Genre']]
     df['Actors'] = [clean(x) for x in df['Actors']]
     df['Director'] = [clean(x) for x in df['Director']]
```

```
[ ]  df['Actors'][0]
```

```
['timrobbins', 'morganfreeman', 'bobgunton', 'williamsadler']
```

```
[ ]  columns = ['clean_plot', 'Genre', 'Actors', 'Director']
     l = []
     for i in range(len(df)):
         words = ''
         for col in columns:
             words += ' '.join(df[col][i]) + ' '
         l.append(words)
     l
```

```
benkingsley candicebergen edwardfox johngielgud richardattenborough ',
 "irish rogue wins heart rich widow assumes dead husband aristocratic position
stanleykubrick ",
 'rocky balboa small time boxer gets supremely rare chance fight heavy weight c
burtyoung carlweathers johng.avildsen ',
```

```
[ ] from sklearn.feature_extraction.text import TfidfVectorizer

    # Assuming 'clean_plot' contains the input text data as a list of strings
    clean_plots = df['clean_plot'].apply(lambda x: ' '.join(x))  # Convert list of tokens to strings

    # Initialize the TfidfVectorizer
    tfidf = TfidfVectorizer()

    # Fit-transform the input data
    features = tfidf.fit_transform(clean_plots)

    # Now 'features' contains the TF-IDF features for the input data
```

```
[ ] from sklearn.metrics.pairwise import cosine_similarity
    cosine_sim = cosine_similarity(features, features)
    print(cosine_sim)
```

```
[[1.         0.         0.         ... 0.         0.         0.        ]
 [0.         1.         0.09562443 ... 0.         0.         0.        ]
 [0.         0.09562443 1.         ... 0.         0.         0.        ]
 ...
 [0.         0.         0.         ... 1.         0.         0.        ]
 [0.         0.         0.         ... 0.         1.         0.        ]
 [0.         0.         0.         ... 0.         0.         1.        ]]
```

Movie Recommendation

```
[ ] index = pd.Series(df['Title'])
    index.head()
```

```
0    The Shawshank Redemption
1               The Godfather
2      The Godfather: Part II
3             The Dark Knight
4                12 Angry Men
Name: Title, dtype: object
```

```
[ ] def recommend_movies(title):
        movies = []
        idx = index[index == title].index[0]
        # print(idx)
        score = pd.Series(cosine_sim[idx]).sort_values(ascending=False)
        top10 = list(score.iloc[1:11].index)
        # print(top10)

        for i in top10:
            movies.append(df['Title'][i])
        return movies
```

```
[ ]  recommend_movies('The Dark Knight Rises')
```

```
['The Dark Knight',
 'The Lord of the Rings: The Fellowship of the Ring',
 'Batman Begins',
 'Sin City',
 'Django Unchained',
 'Die Hard',
 'The Lord of the Rings: The Two Towers',
 'The Exorcist',
 'Star Wars: Episode IV - A New Hope',
 'Guardians of the Galaxy']
```

```
[ ]  index[index == 'The Dark Knight Rises'].index[0]
```

```
51
```

```
[ ]  pd.Series(cosine_sim[3]).sort_values(ascending=False)
```

```
3      1.000000
51     0.160944
187    0.100171
201    0.088910
89     0.087022
         ...
103    0.000000
104    0.000000
105    0.000000
106    0.000000
249    0.000000
Length: 250, dtype: float64
```

```
[ ]  recommend_movies('The Shawshank Redemption')
```

```
['Pulp Fiction',
 "Hachi: A Dog's Tale",
 'The Great Escape',
 'Rope',
 'Goodfellas',
 'Beauty and the Beast',
 'A Night at the Opera',
 'The Lord of the Rings: The Return of the King',
 'Paris, Texas',
 'The Bridge on the River Kwai']
```

```
[ ]  recommend_movies('The Avengers')
```

```
['Guardians of the Galaxy Vol. 2',
 'The Martian',
 'Interstellar',
 'Aliens',
 'Guardians of the Galaxy',
 'Kill Bill: Vol. 1',
 'Before Sunrise',
 'The Thing',
 'Blade Runner',
 'Zootopia']
```

Conclusion:

In conclusion, the development of a content-based filtering system for IMDb offers a promising solution to the challenge of content discovery in the vast sea of available movies and television shows. By leveraging the rich metadata available on IMDb, we can create a recommendation engine that considers the intrinsic characteristics of each item to make personalized suggestions to users. Through this approach, we aim to improve user satisfaction, increase user engagement, and enhance the overall user experience on the platform.

Colab link:

https://colab.research.google.com/drive/1YAtvuzjYS9wy8xIsUsyrCv22INkGm4HE#scrollTo=qPCsd63viDwh&uniqifier=2

Course completion certificate:

https://skills.yourlearning.ibm.com/certificate/share/bcaff2ac79ewogICJvYmplY3RJZCIgOiAiUExBTi0zRTlBQjQwREIwNTEiLAogICJvYmplY3RUeXBlIiA6ICJBQ1RJVklUWSIsCiAgImxlYXJuZXJDTlVNIiA6ICIyNDExNDAwUkVHIgp9bd59bc3261-10