3-content-based-recommender-system

March 13, 2023

[4]:
```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

movies_df = pd.read_csv(r'C:\Users\Bilal\Desktop\RS␣
 ↪assignment\ml-latest-small\movies.csv')

vectorizer = CountVectorizer(tokenizer=lambda x: x.split('|'))
genre_matrix = vectorizer.fit_transform(movies_df['genres'])

genre_similarity = cosine_similarity(genre_matrix)

def recommend_movies(movie_name, n_recommendations):
    movie_index = movies_df.loc[movies_df['title'] == movie_name].index[0]

    movie_scores = genre_similarity[movie_index]

    top_movies = movie_scores.argsort()[::-1][1:n_recommendations+1]

    return movies_df.loc[top_movies, 'title'].tolist()

movie_name = input("Enter movie name")
n_recommendations = 3
recommendations = recommend_movies(movie_name, n_recommendations)
print('Top', n_recommendations, 'recommended movies for', movie_name, ':')
print(recommendations)
```

```
Enter movie nameSudden Death (1995)
Top 3 recommended movies for Sudden Death (1995) :
['Vigilante Diaries (2016)', 'Invincible Shaolin (1978)', 'Fair Game (1995)']
```

[3]:
```

```

```
Enter movie nameSabrina (1995)
Top 3 recommended movies for Sabrina (1995) :
['When in Rome (2010)', "Say It Isn't So (2001)", 'Leap Year (2010)']
```

[ ]:
```

```