Rodriguez_Felipe_DSC630_Week_10_Code

November 5, 2023

This recommender system is created using the guide from the article "How To Build Your First Recommender System Using Python & MovieLens Dataset". The first step of creating this system is importing the data and ensuring that it is loaded correctly. For this system, there are two data frames being created. One contains the titles and genres of the movies, and the other contains the ratings of the movies.

[1]:
```
# Ignores warnings
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
import numpy as np
import pandas as pd
```

[3]:
```
df = pd.read_csv('movies.csv')
```

[4]:
```
# imports the movie titles and genres
df.head()
```

[4]:
```
   movieId                               title  \
0        1                    Toy Story (1995)
1        2                      Jumanji (1995)
2        3             Grumpier Old Men (1995)
3        4            Waiting to Exhale (1995)
4        5  Father of the Bride Part II (1995)

                                        genres
0  Adventure|Animation|Children|Comedy|Fantasy
1                   Adventure|Children|Fantasy
2                               Comedy|Romance
3                         Comedy|Drama|Romance
4                                       Comedy
```

[5]:
```
# imports the movie rating
data = pd.read_csv('ratings.csv')
data.head()
```

[5]:
```
   userId  movieId  rating  timestamp
0       1        1     4.0  964982703
```

```
1        1        3      4.0  964981247
2        1        6      4.0  964982224
3        1       47      5.0  964983815
4        1       50      5.0  964982931
```

Once that the data has been loaded in, it needs to be joined so that all the information is in one data frame. This data frame will include the movie titles and the ratings.

[6]:
```
# Merges both data frames on movie id to have all data in one data frame
data = data.merge(df,on='movieId', how='left')
data.head()
```

[6]:
```
   userId  movieId  rating  timestamp                        title  \
0       1        1     4.0  964982703             Toy Story (1995)
1       1        3     4.0  964981247      Grumpier Old Men (1995)
2       1        6     4.0  964982224                  Heat (1995)
3       1       47     5.0  964983815  Seven (a.k.a. Se7en) (1995)
4       1       50     5.0  964982931   Usual Suspects, The (1995)

                                        genres
0  Adventure|Animation|Children|Comedy|Fantasy
1                               Comedy|Romance
2                        Action|Crime|Thriller
3                             Mystery|Thriller
4                       Crime|Mystery|Thriller
```

"The dataset is a collection of ratings by a number of users for different movies"(Nair, 2019), calculating the average rating next will help later on in the recommender system.

[7]:
```
# Creates a data frame of avarage movie rating
Average_ratings = pd.DataFrame(data.groupby('title')['rating'].mean())
Average_ratings.head()
```

[7]:
```
                                         rating
title
'71 (2014)                                  4.0
'Hellboy': The Seeds of Creation (2004)     4.0
'Round Midnight (1986)                      3.5
'Salem's Lot (2004)                         5.0
'Til There Was You (1997)                   4.0
```

Next, the total amount of rating is calculated to understand the proportion of the ratings. "The rating of a movie is proportional to the total number of ratings it has" (Nair, 2019).

[8]:
```
# creates a counnt of how many movie ratings each movie has
Average_ratings['Total Ratings'] = pd.DataFrame(data.groupby('title')['rating'].
 ↪count())
Average_ratings.head()
```

```
[8]:                                       rating  Total Ratings
title
'71 (2014)                                    4.0              1
'Hellboy': The Seeds of Creation (2004)       4.0              1
'Round Midnight (1986)                        3.5              2
'Salem's Lot (2004)                           5.0              1
'Til There Was You (1997)                     4.0              2
```

The recommender will use the ratings from each user for each movie, to create correlations. With that, the data will need to be pivoted to have all the ratings as values.

```
[9]: # Ratings each user has for the movies
movie_user = data.pivot_table(index='userId',columns='title',values='rating')
```

With the data pivoted, the ratings can be used to determine correlations between movies. The first example of this system will use the movie title 'Jumanji (1995)'. Please note in order for this recommender system to work, the title needs to be in the same format as the title in the data frame, or else it will not load recommendations.

```
[10]: # creates correlations using ratings, This is example is for Jumanji (1995)
# To change the movie add a different title from the data frame
correlations = movie_user.corrwith(movie_user['Jumanji (1995)'])
correlations.head(10)
```

```
[10]: title
'71 (2014)                                       NaN
'Hellboy': The Seeds of Creation (2004)          NaN
'Round Midnight (1986)                           NaN
'Salem's Lot (2004)                              NaN
'Til There Was You (1997)                        NaN
'Tis the Season for Love (2015)                  NaN
'burbs, The (1989)                          0.120173
'night Mother (1986)                             NaN
(500) Days of Summer (2009)                 0.397966
*batteries not included (1987)              0.719636
dtype: float64
```

With the correlations added, the number of ratings each movie has will also be a factor. This way a filter can be included to omit ratings that fall below a threshold of total ratings received.

```
[11]: # Creates data frame that inclues total ratings and correlations
recommendation = pd.DataFrame(correlations,columns=['Correlation'])
recommendation.dropna(inplace=True)
recommendation = recommendation.join(Average_ratings['Total Ratings'])
recommendation.head()
```

```
[11]:                              Correlation  Total Ratings
title
```

```
'burbs, The (1989)                0.120173          17
(500) Days of Summer (2009)       0.397966          42
*batteries not included (1987)    0.719636           7
10 Cent Pistol (2015)            -1.000000           2
10 Cloverfield Lane (2016)        1.000000          14
```

Below is the example of Jumanji. This system shows the top 10 movies correlated to Jumanji when the title has more than 100 ratings.

[12]:
```python
# Creates recommendation of highest correlating movies only when there are 100 or more reviews
recc = recommendation[recommendation['Total Ratings']>100].sort_values('Correlation',ascending=False).reset_index()

recc = recc.merge(df,on='title', how='left')
recc.head(11)
```

[12]:
```
                        title  Correlation  Total Ratings  movieId  \
0              Jumanji (1995)     1.000000            110        2
1          Cliffhanger (1993)     0.581001            101      434
2            True Lies (1994)     0.493617            178      380
3   Back to the Future (1985)     0.485140            171     1270
4       Mrs. Doubtfire (1993)     0.480007            144      500
5            Net, The (1995)      0.474888            112      185
6        Trainspotting (1996)     0.464547            102      778
7              Twister (1996)     0.460929            123      736
8     Incredibles, The (2004)     0.460369            125     8961
9  Bourne Identity, The (2002)    0.440918            112     5418
10           Mask, The (1994)     0.440507            157      367

                                   genres
0              Adventure|Children|Fantasy
1             Action|Adventure|Thriller
2   Action|Adventure|Comedy|Romance|Thriller
3                Adventure|Comedy|Sci-Fi
4                            Comedy|Drama
5                   Action|Crime|Thriller
6                     Comedy|Crime|Drama
7       Action|Adventure|Romance|Thriller
8  Action|Adventure|Animation|Children|Comedy
9                Action|Mystery|Thriller
10             Action|Comedy|Crime|Fantasy
```

Source: https://analyticsindiamag.com/how-to-build-your-first-recommender-system-using-python-movielens-dataset/