9.2 Exercise: Recommender System

Keiuntae Smith

DSC630 Predictive Analytics

1 Aug 2022

In [1]:
```
# Load libraries
import pandas as pd
import numpy as np
```

Load the Datasets that will be used in this exercise

In [2]:
```
# load the ratings dataset and preview the first 10 rows
data = pd.read_csv('ratings.csv')
data.head(10)
```

Out[2]:

	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
5	1	70	3.0	964982400
6	1	101	5.0	964980868
7	1	110	4.0	964982176
8	1	151	5.0	964984041
9	1	157	5.0	964984100

In [3]:
```
# load the movie titles dataset and preview the first 10 rows
movie_titles_genre = pd.read_csv("movies.csv")
movie_titles_genre.head(10)
```

Out[3]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy
5	6	Heat (1995)	Action|Crime|Thriller
6	7	Sabrina (1995)	Comedy|Romance
7	8	Tom and Huck (1995)	Adventure|Children
8	9	Sudden Death (1995)	Action
9	10	GoldenEye (1995)	Action|Adventure|Thriller

In [4]:
```
# Merge the two dataframes into one
data = data.merge(movie_titles_genre,on='movieId', how='left')

# Preview the first 10 rows of the new dataframe
data.head(10)
```

Out[4]:

	userId	movieId	rating	timestamp	title	genres
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	1	3	4.0	964981247	Grumpier Old Men (1995)	Comedy|Romance
2	1	6	4.0	964982224	Heat (1995)	Action|Crime|Thriller
3	1	47	5.0	964983815	Seven (a.k.a. Se7en) (1995)	Mystery|Thriller
4	1	50	5.0	964982931	Usual Suspects, The (1995)	Crime|Mystery|Thriller
5	1	70	3.0	964982400	From Dusk Till Dawn (1996)	Action|Comedy|Horror|Thriller
6	1	101	5.0	964980868	Bottle Rocket (1996)	Adventure|Comedy|Crime|Romance
7	1	110	4.0	964982176	Braveheart (1995)	Action|Drama|War
8	1	151	5.0	964984041	Rob Roy (1995)	Action|Drama|Romance|War
9	1	157	5.0	964984100	Canadian Bacon (1995)	Comedy|War

In [5]:
```
#create a new dataframe that displays average rating (mean) for each movie in the data
mean_rating = pd.DataFrame(data.groupby('title')['rating'].mean())
mean_rating.head(10)
```

Out[5]:

	rating
title	
'71 (2014)	4.000000
'Hellboy': The Seeds of Creation (2004)	4.000000
'Round Midnight (1986)	3.500000
'Salem's Lot (2004)	5.000000
'Til There Was You (1997)	4.000000
'Tis the Season for Love (2015)	1.500000
'burbs, The (1989)	3.176471
'night Mother (1986)	3.000000
(500) Days of Summer (2009)	3.666667
*batteries not included (1987)	3.285714

Total Number of Ratings

In [6]:
```
#create a dataframe that shows the total ratings cast for each movie
mean_rating['Total Ratings'] = pd.DataFrame(data.groupby('title')['rating'].count())
mean_rating.head(10)
```

Out[6]:

	rating	Total Ratings
title		
'71 (2014)	4.000000	1
'Hellboy': The Seeds of Creation (2004)	4.000000	1
'Round Midnight (1986)	3.500000	2
'Salem's Lot (2004)	5.000000	1
'Til There Was You (1997)	4.000000	2
'Tis the Season for Love (2015)	1.500000	1
'burbs, The (1989)	3.176471	17
'night Mother (1986)	3.000000	1
(500) Days of Summer (2009)	3.666667	42
*batteries not included (1987)	3.285714	7

Correlation Calculation

In [7]:
```
#Calculating The Correlation by making a pivot table (rows=userID, columns=Movie Title)
movie_user = data.pivot_table(index='userId',columns='title', values='rating')
```

In [8]:
```
# Preview the first ten rows pivot dataframe
movie_user.head(10)
```

Out[8]:

title	'71 (2014)	'Hellboy': The Seeds of Creation (2004)	'Round Midnight (1986)	'Salem's Lot (2004)	'Til There Was You (1997)	'Tis the Season for Love (2015)	'burbs, The (1989)	'night Mother (1986)	(500) Days of Summer (2009)	*batteries not included (1987)	...	Zulu (2013)	[REC] (2007)	[REC]² (2009)	[REC]³ 3 Génesis (2012)	anohana: The Flower We Saw That Day - The Movie (2013)	eXistenZ (1999)	xXx (2002)	xXx: State of the Union (2005)	¡Three Amigos! (1986)	(F
userId																					
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
6	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
7	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	1.0	NaN	NaN	
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

10 rows × 9719 columns

Movie Selection

In [9]:
```
#create a correlation value using the 'corrwith' function in conjunction with a movie choice
correlations = movie_user.corrwith(movie_user['Godfather, The (1972)'])

#display the first five rows of computed pairwise correlation between rows and columns
correlations.head()
```

```
/Users/keiuntaesmith/opt/anaconda3/lib/python3.9/site-packages/numpy/lib/function_base.py:2634: RuntimeWarning: Degrees of freedom <= 0 for slice
  c = cov(x, y, rowvar, dtype=dtype)
/Users/keiuntaesmith/opt/anaconda3/lib/python3.9/site-packages/numpy/lib/function_base.py:2493: RuntimeWarning: divide by zero encountered in true_divide
  c *= np.true_divide(1, fact)
```

Out[9]:
```
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
dtype: float64
```

In [10]:
```
# create a new variable named 'recommend' and drop all the empty values
recommend = pd.DataFrame(correlations,columns=['Correlation'])
recommend.dropna(inplace=True)

#Merge the ratings to the correlation table
recommend = recommend.join(mean_rating['Total Ratings'])

#Preview the first 5 rows
recommend.head()
```

Out[10]:

	Correlation	Total Ratings
title		
'burbs, The (1989)	-0.745465	17
(500) Days of Summer (2009)	0.093103	42
*batteries not included (1987)	-0.852803	7
10 Cent Pistol (2015)	1.000000	2
10 Cloverfield Lane (2016)	-0.422890	14

In [11]:
```
#filter all movies that are correlated to 'Godfather, The (1972)' using the sort_values function
recc = recommend[recommend['Total Ratings']>100].sort_values('Correlation',ascending=False).reset_index()
```

In [12]:
```
# merge the original movie dataset to show all fields
recc = recc.merge(movie_titles_genre,on='title', how='left')
```

In [13]:
```
#display the recommended list dataframe to include the movie selection and ten recommended movies
recc.head(11)
```

Out[13]:

	title	Correlation	Total Ratings	movieId	genres
0	Godfather, The (1972)	1.000000	192	858	Crime|Drama
1	Godfather: Part II, The (1974)	0.782643	129	1221	Crime|Drama
2	Schindler's List (1993)	0.456661	220	527	Drama|War
3	Fight Club (1999)	0.445205	218	2959	Action|Crime|Drama|Thriller
4	Saving Private Ryan (1998)	0.441377	188	2028	Action|Drama|War
5	Goodfellas (1990)	0.439937	126	1213	Crime|Drama
6	Inception (2010)	0.432878	143	79132	Action|Crime|Drama|Mystery|Sci-Fi|Thriller|IMAX
7	Star Wars: Episode V - The Empire Strikes Back...	0.428278	211	1196	Action|Adventure|Sci-Fi
8	Reservoir Dogs (1992)	0.423716	131	1089	Crime|Mystery|Thriller
9	Outbreak (1995)	0.421361	101	292	Action|Drama|Sci-Fi|Thriller
10	Clockwork Orange, A (1971)	0.420624	120	1206	Crime|Drama|Sci-Fi|Thriller

Movie Recommender System Process

1. Load libraries
2. Load the ratings dataset and preview the first 10 rows
3. Load the movie titles dataset and preview the first 10 rows
4. Merge the two dataframes into one
5. Preview the first 10 rows of the new dataframe
6. Create a new dataframe that displays average rating (mean) for each movie in the data
7. Create a dataframe that shows the total ratings cast for each movie
8. Calculating the Correlation by making a pivot table (rows=userID, columns=Movie Title)
9. Preview the first ten rows pivot dataframe
10. Create a correlation value using the 'corrwith' function in conjunction with a movie choice
11. Display the first five rows of computed pairwise correlation between rows and columns
12. Create a new variable named 'recommend' and drop all the empty values
13. Merge the ratings to the correlation table
14. Filter all movies that are correlated to 'Godfather, The (1972)' using the sort_values function
15. Merge the original movie dataset to show all fields
16. Display the recommended list dataframe to include the movie selection and ten recommended movies (Nair, 2019)

After following the 16 steps above, I was able to produce a list of impressive movies after I selected my favorite movie of all time "The Godfather". The list is as follows:

• Godfather: Part II, The (1974) • Schindler's List (1993) • Fight Club (1999) • Saving Private Ryan • Goodfellas (1990) • Inception (2010) • Star Wars: Episode V - The Empire Strikes Back (1980) • Reservoir Dogs (1992) • Outbreak (1995) • Clockwork Orange, A (1971)

To my present surprise, the recommender system actually produced 10 movies that I actually really enjoyed, with the exception of Clockwork Orange. So I guess I have a new movie to watch this weekend.

Bibliography Nair, A. (2019, September 25). How To Build Your First Recommender System Using Python & MovieLens Dataset. Retrieved from analyticsindiamag.com: https://analyticsindiamag.com/how-to-build-your-first-recommender-system-using-python-movielens-dataset/

In [ ]:
```
```