Building a recommender system using Movie lens Data set

In [1]:
```
#Load the data from the csv files

import numpy as np
import pandas as pd
import warnings

ratings_df = pd.read_csv('ratings.csv')
movies_df = pd.read_csv("movies.csv")
ratings_df.head(10)
```

Out[1]:

	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 [2]:
```
movies_df.head(10)
```

Out[2]:

	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

We need to merge the datsets to bring two datasets together into one.

In [3]:
```
# Merge the two data sets
ratings_df = ratings_df.merge(movies_df,on='movieId', how='left')
warnings.filterwarnings('ignore')
ratings_df.head(10)
```

Out[3]:

	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

The dataset we have is a collection of ratings by a number of users for different movies. We need to find out the average rating for each and every movie in the dataset.

In [4]:
```
# Calculate the average ratings of the movies
Average_ratings_df = pd.DataFrame(ratings_df.groupby('title')['rating'].mean())
Average_ratings_df.head(10)
```

Out[4]:

	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

title	rating
'night Mother (1986)	3.000000
(500) Days of Summer (2009)	3.666667
***batteries not included (1987)**	3.285714

No. of ratings for a movie also plays a crucial role in the overall rating of the movie. Therefore, we will also consider the total ratings cast for each movie.

In [5]:
```
# Calculate the total ratings of the movies
Average_ratings_df['Total Ratings'] = pd.DataFrame(ratings_df.groupby('title')['rating'
Average_ratings_df.head(10)
```

Out[5]:

title	rating	Total Ratings
'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

We will create a pivot table where the rows are userIds and the columns represent the movies. The values of the matrix represent the rating for each movie by each user.

In [6]:
```
# Pivot the data to show the rating for each movie by each user.
movie_user_df = ratings_df.pivot_table(index='userId',columns='title',values='rating')
movie_user_df.head()
```

Out[6]:

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)	..
userId											

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)	..
userId											
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	..

5 rows × 9719 columns

To find the correlation value for the movie with all other movies in the data, we will pass all the ratings of the picked movie to the corrwith method to compute the pairwise correlation.

In [7]:
```
# Take the input from user and calculate the correlation with other movies
movie_name=input("enter the movie name:")
correlations = movie_user_df.corrwith(movie_user_df[movie_name])
correlations.head()
```

```
enter the movie name:Jumanji (1995)
```

Out[7]:
```
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
```

Join the correlations that we got with the total ratings to be able to filter out the ones with no ratings and to have the ability to filter on no. of ratings

In [8]:
```
# For filtering out the recommendations, drop all the blank rows
recommendation = pd.DataFrame(correlations,columns=['Correlation'])
recommendation.dropna(inplace=True)
recommendation = recommendation.join(Average_ratings_df['Total Ratings'])
recommendation.head()
```

Out[8]:

title	Correlation	Total Ratings
'burbs, The (1989)	0.120173	17
(500) Days of Summer (2009)	0.397966	42

title	Correlation	Total Ratings
***batteries not included (1987)**	0.719636	7
10 Cent Pistol (2015)	-1.000000	2
10 Cloverfield Lane (2016)	1.000000	14

In [9]:

```
# Filter out recommendations with more than 75 ratings and display top 10
recc = recommendation[recommendation['Total Ratings']>75].sort_values('Correlation',asc
recc = recc.merge(movies_df,on='title', how='left')
recc.head(10)
```

Out[9]:

	title	Correlation	Total Ratings	movieId	genres
0	Jumanji (1995)	1.000000	110	2	Adventure|Children|Fantasy
1	Prestige, The (2006)	0.631529	90	48780	Drama|Mystery|Sci-Fi|Thriller
2	Game, The (1997)	0.594167	77	1625	Drama|Mystery|Thriller
3	Cliffhanger (1993)	0.581001	101	434	Action|Adventure|Thriller
4	Broken Arrow (1996)	0.575196	84	95	Action|Adventure|Thriller
5	Back to the Future Part II (1989)	0.564930	87	2011	Adventure|Comedy|Sci-Fi
6	Santa Clause, The (1994)	0.557586	81	317	Comedy|Drama|Fantasy
7	Back to the Future Part III (1990)	0.525240	88	2012	Adventure|Comedy|Sci-Fi|Western
8	True Lies (1994)	0.493617	178	380	Action|Adventure|Comedy|Romance|Thriller
9	Back to the Future (1985)	0.485140	171	1270	Adventure|Comedy|Sci-Fi

Reference:

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