ARATHI V S

CODING_RAJA_TECHNOLOGIES-PROJECT

MOVIE RECOMMENDATION SYSTEM

A movie recommendation system is a fancy way to describe a process that tries to predict your preferred items based on your or people similar to you

IMPORTING LIBRARIES

```
import numpy as np
import pandas as pd
```

GET DATA

```
column_names = ['user_id', 'item_id', 'rating', 'timestamp']
df = pd.read_csv("/content/u.data", sep = '\t', names = column_names)
```

```
df
```

	user_id	item_id	rating	timestamp
0	0	50	5	881250949
1	0	172	5	881250949
2	0	133	1	881250949

```
     4      186    302    3  891717742
   ...      ...    ...  ...        ...
 99998      880    476    3  880175444
 99999      716    204    5  879795543
100000      276   1090    1  874795795
100001       13    225    2  882399156
100002       12    203    3  879959583

100003 rows × 4 columns
```

```
df.head()
```

```
   user_id  item_id  rating  timestamp
0        0       50       5  881250949
1        0      172       5  881250949
2        0      133       1  881250949
3      196      242       3  881250949
4      186      302       3  891717742
```

```
movie_titles=pd.read_csv("/content/Movie_Id_Titles.txt")
movie_titles
```

```
      item_id                                     title
0           1                          Toy Story (1995)
1           2                          GoldenEye (1995)
2           3                         Four Rooms (1995)
3           4                         Get Shorty (1995)
4           5                            Copycat (1995)
...       ...                                       ...
1677     1678                         Mat' i syn (1997)
1678     1679                          B. Monkey (1998)
1679     1680                      Sliding Doors (1998)
1680     1681                       You So Crazy (1994)
1681     1682  Scream of Stone (Schrei aus Stein) (1991)

1682 rows × 2 columns
```

```
df = pd.merge(df,movie_titles,on='item_id')
df.head()
```

	user_id	item_id	rating	timestamp	title
0	0	50	5	881250949	Star Wars (1977)
1	290	50	5	880473582	Star Wars (1977)
2	79	50	4	891271545	Star Wars (1977)
3	2	50	5	888552084	Star Wars (1977)
4	8	50	5	879362124	Star Wars (1977)

EDA

```
df.describe()
```

	user_id	item_id	rating	timestamp
count	100003.000000	100003.000000	100003.000000	1.000030e+05
mean	462.470876	425.520914	3.529864	8.835288e+08
std	266.622454	330.797791	1.125704	5.343791e+06
min	0.000000	1.000000	1.000000	8.747247e+08
25%	254.000000	175.000000	3.000000	8.794487e+08
50%	447.000000	322.000000	4.000000	8.828269e+08
75%	682.000000	631.000000	4.000000	8.882600e+08
max	943.000000	1682.000000	5.000000	8.932866e+08

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 100003 entries, 0 to 100002
Data columns (total 5 columns):
 #   Column     Non-Null Count   Dtype 
---  ------     --------------   ----- 
 0   user_id    100003 non-null  int64 
 1   item_id    100003 non-null  int64 
 2   rating     100003 non-null  int64 
 3   timestamp  100003 non-null  int64 
 4   title      100003 non-null  object
dtypes: int64(4), object(1)
memory usage: 4.6+ MB
```

```
df.isna().sum()
```

```
    user_id      0
    item_id      0
    rating       0
    timestamp    0
    title        0
    dtype: int64
```

VISUALIZATION

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('white')
%matplotlib inline
```

Lets create a ratings dataframe of average rating and number of rating

```
df.groupby('title')['rating'].mean().sort_values(ascending=False).head()
```

```
    title
    They Made Me a Criminal (1939)                5.0
    Marlene Dietrich: Shadow and Light (1996)     5.0
    Saint of Fort Washington, The (1993)          5.0
    Someone Else's America (1995)                 5.0
    Star Kid (1997)                               5.0
    Name: rating, dtype: float64
```

```
df.groupby('title')['rating'].count().sort_values(ascending=False).head()
```

```
    title
    Star Wars (1977)               584
    Contact (1997)                 509
    Fargo (1996)                   508
    Return of the Jedi (1983)      507
    Liar Liar (1997)               485
    Name: rating, dtype: int64
```

```
ratings = pd.DataFrame(df.groupby('title')['rating'].mean())
ratings.head()
```

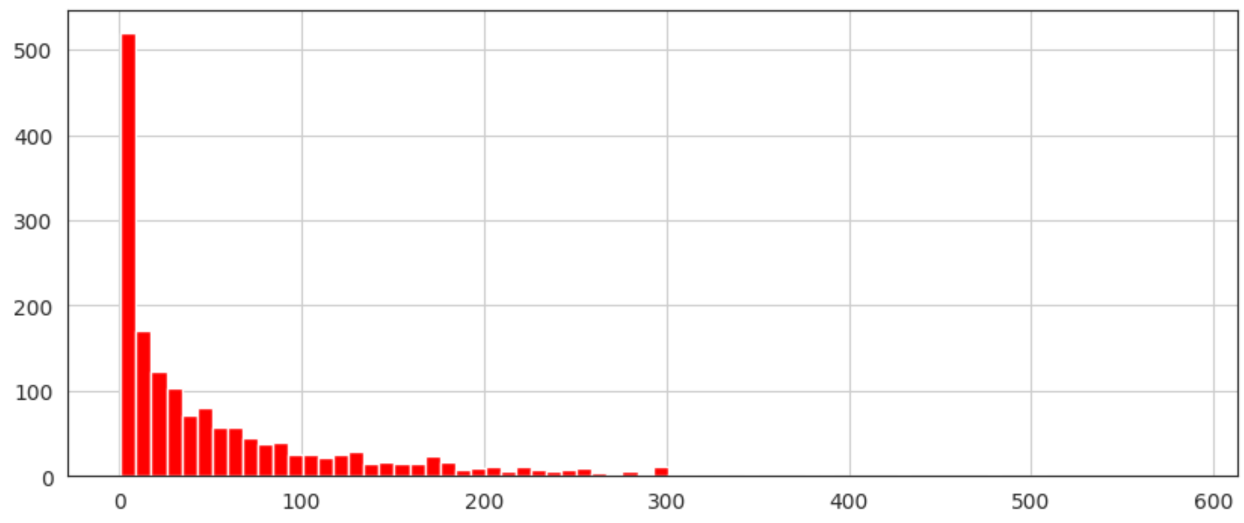
	rating
title	
'Til There Was You (1997)	2.333333
1-900 (1994)	2.600000
101 Dalmatians (1996)	2.908257
12 Angry Men (1957)	4.344000
187 (1997)	3.024390

number of ratings column

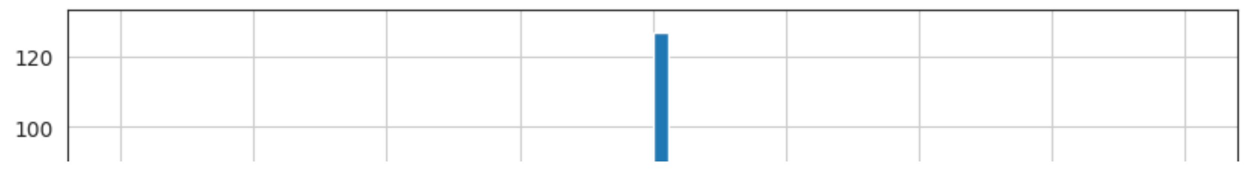
```
ratings['num of ratings'] = pd.DataFrame(df.groupby('title')['rating'].count())
ratings.head()
```

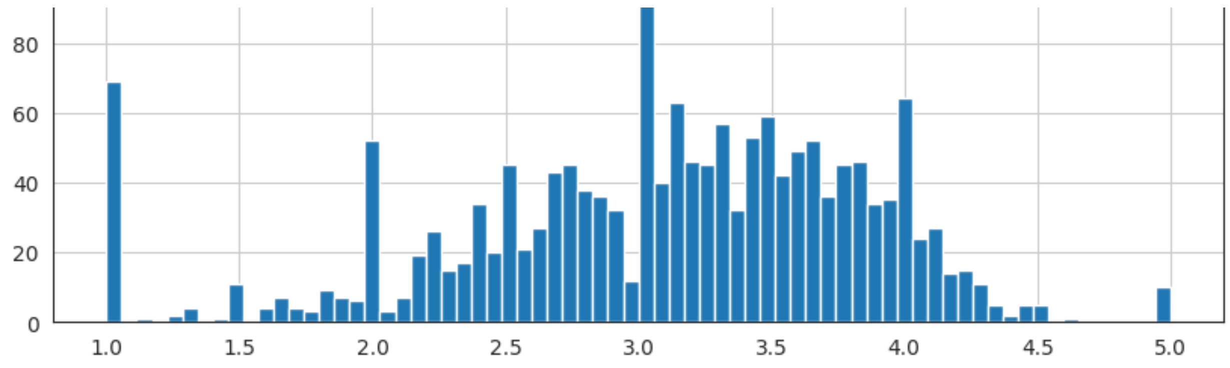
title	rating	num of ratings
'Til There Was You (1997)	2.333333	9
1-900 (1994)	2.600000	5
101 Dalmatians (1996)	2.908257	109
12 Angry Men (1957)	4.344000	125
187 (1997)	3.024390	41

```
plt.figure(figsize=(10,4))
ratings['num of ratings'].hist(bins=70,color='red')
plt.show()
```

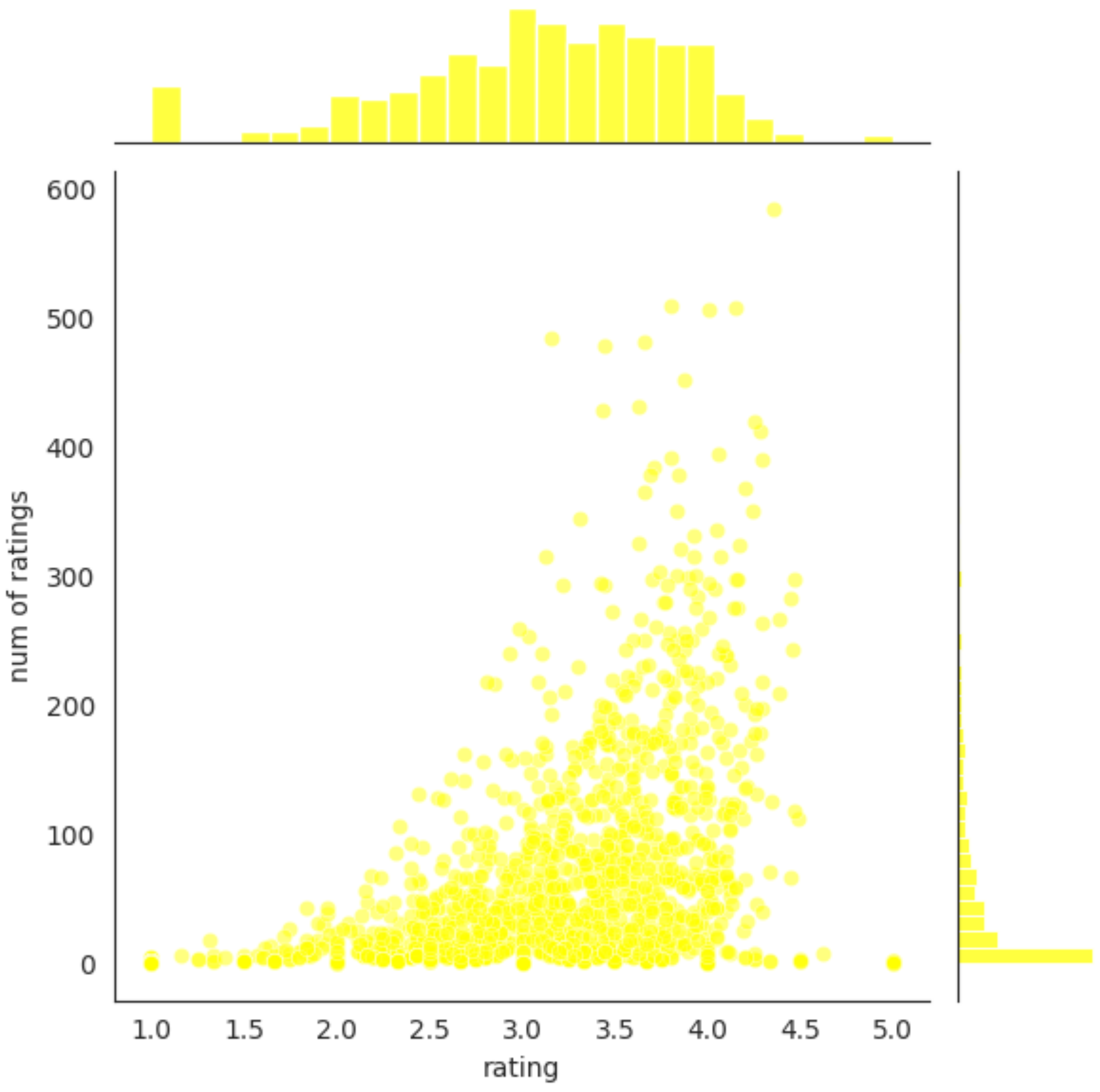


```
plt.figure(figsize=(10,4))
ratings['rating'].hist(bins=70)
plt.show()
```





```
sns.jointplot(x='rating',y='num of ratings', data=ratings, alpha=0.5,color='yell
plt.show()
```



Okay! Now that we have a general idea of what the data looks like, let's move on to creating a simple recommendation system:

Recommending Similar Movies

Now let's create a matrix that has the user ids on one access and the movie title on another axis. Each cell will then consist of the rating the user gave to that movie. Note there will be a lot of NaN values, because most people have not seen most of the movies.

```
moviemat = df.pivot_table(index = 'user_id',columns = 'title', values = 'rating'
moviemat.head()
```

title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: Spac Odysse (1968
user_id								
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
1	NaN	NaN	2.0	5.0	NaN	NaN	3.0	4.
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
3	NaN	NaN	NaN	NaN	2.0	NaN	NaN	Na
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

5 rows × 1664 columns

Most rated movie:

```
ratings.sort_values('num of ratings', ascending = False).head(10)
```

	rating	num of ratings
title		
Star Wars (1977)	4.359589	584
Contact (1997)	3.803536	509
Fargo (1996)	4.155512	508
Return of the Jedi (1983)	4.007890	507
Liar Liar (1997)	3.156701	485
English Patient, The (1996)	3.656965	481
Scream (1996)	3.441423	478

Scream (1996)	3.441423	478
Toy Story (1995)	3.878319	452
Air Force One (1997)	3.631090	431
Independence Day (ID4) (1996)	3.438228	429

Let's choose two movies: starwars, a sci-fi movie, And Liar Liar, a comedy.

```
ratings.head()
```

	rating	num of ratings
title		
'Til There Was You (1997)	2.333333	9
1-900 (1994)	2.600000	5
101 Dalmatians (1996)	2.908257	109
12 Angry Men (1957)	4.344000	125
187 (1997)	3.024390	41

Now let's grab the user ratings for those two movies:

```
starwars_user_ratings = moviemat['Star Wars (1977)']
liarliar_user_ratings = moviemat['Liar Liar (1997)']
starwars_user_ratings.head()
```

```
user_id
0    5.0
1    5.0
2    5.0
3    NaN
4    5.0
Name: Star Wars (1977), dtype: float64
```

We can then use corrwith() method to get correlations between two pandas series:

```
similar_to_starwars = moviemat.corrwith(starwars_user_ratings)
similar_to_liarliar = moviemat.corrwith(liarliar_user_ratings)
```

```
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2821: Ru
  c = cov(x, y, rowvar, dtype=dtype)
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2680: Ru
  c *= np.true_divide(1, fact)
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2821: Ru
  c = cov(x, y, rowvar, dtype=dtype)
```

```
/usr/local/lib/python3.10/dist-packages/numpy/lib/function_base.py:2680: Ru
  c *= np.true_divide(1, fact)
```

Let's clean this by removing NaN values and using a DataFrame instead of a series:

```
corr_starwars = pd.DataFrame(similar_to_starwars, columns = ['Correlation'])
corr_starwars.dropna(inplace=True)
corr_starwars.head()
```

	Correlation
title	
'Til There Was You (1997)	0.872872
1-900 (1994)	-0.645497
101 Dalmatians (1996)	0.211132
12 Angry Men (1957)	0.184289
187 (1997)	0.027398

Now if we sort the dataframe by correlation, we should get the most similar movies, however note that we get some results that don't really make sense. This is because there are a lot of movies only watched once by users who also watched star wars (it was the most popular movie).

```
corr_starwars.sort_values('Correlation',ascending=False).head(10)
```

	Correlation
title	
Hollow Reed (1996)	1.0
Commandments (1997)	1.0
Cosi (1996)	1.0
No Escape (1994)	1.0
Stripes (1981)	1.0
Star Wars (1977)	1.0
Man of the Year (1995)	1.0
Beans of Egypt, Maine, The (1994)	1.0
Old Lady Who Walked in the Sea, The (Vieille qui marchait dans	1.0

la mer, La) (1991)	1.0
Outlaw, The (1943)	1.0

Let's fix this by filtering out movies that have less than 100 reviews (this value was chosen based off the histogram from earlier).

```
corr_starwars = corr_starwars.join(ratings['num of ratings'])
corr_starwars.head()
```

	Correlation	num of ratings
title		
'Til There Was You (1997)	0.872872	9
1-900 (1994)	-0.645497	5
101 Dalmatians (1996)	0.211132	109
12 Angry Men (1957)	0.184289	125
187 (1997)	0.027398	41

Now sort the values and notice how the titles make a lot more sense:

```
corr_starwars[corr_starwars['num of ratings']>100].sort_values('Correlation', as
```

	Correlation	num of ratings
title		
Star Wars (1977)	1.000000	584
Empire Strikes Back, The (1980)	0.748353	368
Return of the Jedi (1983)	0.672556	507
Raiders of the Lost Ark (1981)	0.536117	420
Austin Powers: International Man of Mystery	0.377433	130

Now the same for the comedy Liar Liar:

```
corr_liarliar = pd.DataFrame(similar_to_liarliar, columns = ['Correlation'])
corr_liarliar.dropna(inplace = True)
corr_liarliar.head()
```

	Correlation
title	
'Til There Was You (1997)	0.118913

101 Dalmatians (1996)	0.469765
12 Angry Men (1957)	0.066272
187 (1997)	0.175145
2 Days in the Valley (1996)	0.040739

```
corr_liarliar = corr_liarliar.join(ratings['num of ratings'])
corr_liarliar[corr_liarliar['num of ratings']>100].sort_values('Correlation', as
```

	Correlation	num of ratings
title		
Liar Liar (1997)	1.000000	485
Batman Forever (1995)	0.516968	114
Mask, The (1994)	0.484650	129
Down Periscope (1996)	0.472681	101
Con Air (1997)	0.469828	137