# codingraja-movies-dataaset

October 7, 2023

```
[ ]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[ ]: movies=pd.read_csv('movies.csv') # first 5 rows
     movies.head()
```

```
[ ]:    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
```

```
[ ]: #Last 5 rows
```

```
[ ]: movies.tail()
```

```
[ ]:        movieId                               title               genres
     62418   209157                           We (2018)                Drama
     62419   209159           Window of the Soul (2001)          Documentary
     62420   209163                    Bad Poems (2018)         Comedy|Drama
     62421   209169                 A Girl Thing (2001)   (no genres listed)
     62422   209171     Women of Devil's Island (1962)  Action|Adventure|Drama
```

```
[ ]: #Total rows and columns in the dataset
```

```
[ ]: movies.shape
```

[ ]: (62423, 3)

[ ]:
```
#There is some noise in the title column that is () which is going to be␣
↪removed to make the data smoother
```

[ ]:
```
import re
def clean_title(title):
  return re.sub("[^a-zA-Z0-9]"," ",title)
```

[ ]:
```
movies['clean_title']=movies['title'].apply(clean_title)
```

[ ]:
```
movies
```

[ ]:
```
       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)
…          …                                 …
62418   209157                           We (2018)
62419   209159           Window of the Soul (2001)
62420   209163                    Bad Poems (2018)
62421   209169                 A Girl Thing (2001)
62422   209171      Women of Devil's Island (1962)

                                            genres  \
0      Adventure|Animation|Children|Comedy|Fantasy
1                       Adventure|Children|Fantasy
2                                   Comedy|Romance
3                             Comedy|Drama|Romance
4                                           Comedy
…                                             …
62418                                        Drama
62419                                  Documentary
62420                                 Comedy|Drama
62421                           (no genres listed)
62422                       Action|Adventure|Drama

                             clean_title
0                         Toy Story  1995
1                           Jumanji  1995
2                  Grumpier Old Men  1995
3                 Waiting to Exhale  1995
4       Father of the Bride Part II  1995
…                                    …
62418                              We  2018
```

```
62419          Window of the Soul  2001
62420                   Bad Poems  2018
62421                A Girl Thing  2001
62422     Women of Devil s Island  1962

[62423 rows x 4 columns]
```

[ ]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
vector=TfidfVectorizer(ngram_range=(1,2))
tfidf=vector.fit_transform(movies['clean_title'])
```

[ ]:
```
#Building the search engine
```

[ ]:
```
from sklearn.metrics.pairwise import cosine_similarity
def search(title):
  title=clean_title(title)
  query_vec=vector.transform([title])
  similarity=cosine_similarity(query_vec,tfidf).flatten()
  indices=np.argpartition(similarity,-5)[-5:]
  results=movies.iloc[indices][::-1]
  return results
```

[ ]:
```

```

[ ]:
```
#Building an interactive search box with jupyter
```

[ ]:
```
import ipywidgets as  widgets
from IPython.display import display
movie_input=widgets.Text(value='Toy Story',description='Movie Title:
↪',disabled=False)
movie_list=widgets.Output()
def ontype(data):
  with movie_list:
    movie_list.clear_output()
    title=data['new']
    if(len(title)>5):
      display(search(title))
movie_input.observe(ontype,names='value')
display(movie_input,movie_list)
```

```
Text(value='Toy Story', description='Movie Title:')

Output()
```

[ ]:
```

```

[ ]:
```
#ratings
```

[ ]:
```
ratings=pd.read_csv('ratings.csv')
```

[ ]:
```
ratings
```

[ ]:
```
       userId  movieId  rating     timestamp
0           1    296.0     5.0  1.147880e+09
1           1    306.0     3.5  1.147869e+09
2           1    307.0     5.0  1.147869e+09
3           1    665.0     5.0  1.147879e+09
4           1    899.0     3.5  1.147869e+09
…         …       …       …             …
85373     647   9010.0     2.5  1.330432e+09
85374     647  27402.0     4.0  1.506807e+09
85375     647  27660.0     3.0  1.456428e+09
85376     647  27904.0     3.5  1.509057e+09
85377     647      NaN     NaN           NaN

[85378 rows x 4 columns]
```

[ ]:
```
ratings.shape
```

[ ]:
```
(85378, 4)
```

[ ]:
```
ratings.dtypes
```

[ ]:
```
userId         int64
movieId      float64
rating       float64
timestamp    float64
dtype: object
```

[ ]:
```
#craeting the similar users
```

[ ]:
```
movie_id=1
similar_users=ratings[(ratings['movieId']==movie_id) & (ratings['rating'] >=␣
↪4)]['userId'].unique()
```

[ ]:
```
similar_users
```

[ ]:
```
array([  3,   5,   8,  12,  13,  36,  43,  50,  51,  57,  64,  75,  77,
        82,  86,  90,  93,  95,  96,  98, 109, 110, 111, 120, 125, 127,
       132, 143, 147, 152, 158, 160, 162, 166, 167, 171, 175, 186, 188,
       200, 211, 216, 217, 221, 227, 229, 230, 233, 235, 236, 249, 256,
       257, 259, 261, 265, 297, 298, 302, 304, 312, 323, 329, 340, 350,
       354, 355, 358, 359, 364, 368, 369, 371, 372, 381, 386, 392, 396,
       402, 405, 409, 411, 414, 421, 422, 424, 428, 435, 436, 437, 439,
       446, 447, 449, 459, 468, 469, 477, 484, 495, 497, 502, 508, 513,
```

```
        519, 531, 537, 540, 541, 543, 548, 551, 553, 561, 567, 572, 573,
        580, 581, 582, 592, 593, 597, 601, 606, 607, 609, 611, 623, 624,
        626, 627, 628, 631, 636, 638, 644])
```

[ ]:
```
similar_users.shape
```

[ ]:
```
(137,)
```

[ ]:
```
#if user id is in similar users and ratings is greater then 4
```

[ ]:
```
similar_users = ratings[(ratings["movieId"] == movie_id) & (ratings["rating"] >
 ↪4)]["userId"].unique()
similar_users
```

[ ]:
```
array([ 36,  75,  86,  90,  93,  95,  96,  98, 111, 120, 127, 143, 152,
       158, 160, 162, 171, 186, 188, 211, 217, 229, 230, 235, 249, 257,
       259, 297, 298, 302, 323, 329, 355, 359, 369, 371, 381, 392, 402,
       411, 428, 435, 439, 447, 449, 468, 469, 477, 484, 513, 519, 537,
       540, 541, 548, 551, 553, 561, 567, 573, 582, 593, 607, 609, 611,
       623, 624, 626, 628, 631, 644])
```

[ ]:
```
similar_user_recs = ratings[(ratings["userId"].isin(similar_users)) &
 ↪(ratings["rating"] > 4)]["movieId"]
similar_user_recs
```

[ ]:
```
5101       1.0
5105      34.0
5111     110.0
5114     150.0
5127     260.0
         …
85171    356.0
85173    380.0
85182    588.0
85183    589.0
85186    593.0
Name: movieId, Length: 4526, dtype: float64
```

[ ]:
```
similar_user_recs.value_counts()
```

[ ]:
```
1.0         71
318.0       35
593.0       25
356.0       23
296.0       23
            ..
112818.0     1
```

```
111617.0     1
106487.0     1
106100.0     1
117176.0     1
Name: movieId, Length: 1936, dtype: int64
```

[ ]:
```
similar_user_recs.value_counts()/len(similar_users)
```

[ ]:
```
1.0         1.000000
318.0       0.492958
593.0       0.352113
356.0       0.323944
296.0       0.323944
              …
112818.0    0.014085
111617.0    0.014085
106487.0    0.014085
106100.0    0.014085
117176.0    0.014085
Name: movieId, Length: 1936, dtype: float64
```

##similar user recommendations with the movies greater than 10 percentage

[ ]:
```
similar_user_recs=similar_user_recs[similar_user_recs>.1]
```

[ ]:
```
similar_user_recs
```

[ ]:
```
5101       1.0
5105      34.0
5111     110.0
5114     150.0
5127     260.0
         …
85171    356.0
85173    380.0
85182    588.0
85183    589.0
85186    593.0
Name: movieId, Length: 4526, dtype: float64
```

##Finding how much all users like movies

[ ]:
```
all_users=ratings[(ratings['movieId'].isin(similar_user_recs.index)) &
 ↪(ratings['rating']> 4)]
```

[ ]:
```
all_users
```

```
[ ]:        userId  movieId  rating     timestamp
     620         3  52950.0     4.5  1.566089e+09
     2411       12  49272.0     4.5  1.167575e+09
     2416       12  52952.0     4.5  1.209130e+09
     2716       13  41569.0     4.5  1.237971e+09
     2733       13  49272.0     5.0  1.238026e+09
     …        …       …       …           …
     78871     606  68793.0     4.5  1.489793e+09
     78900     606  72998.0     4.5  1.473355e+09
     78944     606  82461.0     4.5  1.503538e+09
     83141     626  42015.0     4.5  1.137300e+09
     83336     628  60074.0     4.5  1.480627e+09

     [193 rows x 4 columns]
```

```
[ ]: all_user_recs=all_users['movieId'].value_counts()
```

```
[ ]: all_user_recs
```

```
[ ]: 72998.0    25
     49272.0    17
     56367.0    14
     81564.0     7
     58998.0     6
                ..
     71991.0     1
     73101.0     1
     82041.0     1
     49422.0     1
     42015.0     1
     Name: movieId, Length: 67, dtype: int64
```

```
[ ]: all_user_recs=all_user_recs/len(all_users['userId'].unique())
```

```
[ ]: all_user_recs
```

```
[ ]: 72998.0    0.265957
     49272.0    0.180851
     56367.0    0.148936
     81564.0    0.074468
     58998.0    0.063830
                  …
     71991.0    0.010638
     73101.0    0.010638
     82041.0    0.010638
     49422.0    0.010638
     42015.0    0.010638
```

```
Name: movieId, Length: 67, dtype: float64
```

[ ]:
```
percentages=pd.concat([similar_user_recs,all_user_recs],axis=1)
percentages.columns=['similar','all']
```

[ ]:
```
percentages
```

[ ]:
```
         similar       all
5101.0       1.0       NaN
5105.0      34.0  0.010638
5111.0     110.0       NaN
5114.0     150.0       NaN
5127.0     260.0       NaN
…            …       …
85171.0    356.0       NaN
85173.0    380.0       NaN
85182.0    588.0       NaN
85183.0    589.0       NaN
85186.0    593.0       NaN

[4526 rows x 2 columns]
```

[ ]:
```
#finding the scores between similar column and all columns
```

[ ]:
```
percentages['scores']=percentages['similar']/percentages['all']
```

[ ]:
```
percentages
```

[ ]:
```
         similar       all  scores
5101.0       1.0       NaN     NaN
5105.0      34.0  0.010638  3196.0
5111.0     110.0       NaN     NaN
5114.0     150.0       NaN     NaN
5127.0     260.0       NaN     NaN
…            …       …       …
85171.0    356.0       NaN     NaN
85173.0    380.0       NaN     NaN
85182.0    588.0       NaN     NaN
85183.0    589.0       NaN     NaN
85186.0    593.0       NaN     NaN

[4526 rows x 3 columns]
```

[ ]:
```
percentages=percentages.sort_values('scores',ascending=False)
```

[ ]:
```
percentages
```

[ ]:
```
           similar       all      scores
56587.0  195159.0  0.010638  18344946.0
67695.0  188345.0  0.010638  17704430.0
43936.0  161634.0  0.010638  15193596.0
59418.0   94959.0  0.010638   8926146.0
43871.0   94864.0  0.010638   8917216.0
…              …         …           …
85171.0     356.0       NaN         NaN
85173.0     380.0       NaN         NaN
85182.0     588.0       NaN         NaN
85183.0     589.0       NaN         NaN
85186.0     593.0       NaN         NaN

[4526 rows x 3 columns]
```

[ ]:
```
percentages.head(10).merge(movies,left_index=True,right_on='movieId')
```

[ ]:
```
         similar       all      scores  movieId  \
12005  195159.0  0.010638  18344946.0    56587
13192  188345.0  0.010638  17704430.0    67695
10659  161634.0  0.010638  15193596.0    43936
12337   94959.0  0.010638   8926146.0    59418
10639   94864.0  0.010638   8917216.0    43871
13158   91630.0  0.010638   8613220.0    67267
10650  122912.0  0.021277   5776864.0    43917
12521   41573.0  0.010638   3907862.0    60566
11230   34405.0  0.010638   3234070.0    49422
13484   27869.0  0.010638   2619686.0    69699

                                                   title  \
12005                              Bucket List, The (2007)
13192                            Observe and Report (2009)
10659                                      16 Blocks (2006)
12337                             American Crime, An (2007)
10639                                       Firewall (2006)
13158                              Sunshine Cleaning (2008)
10650                                    Eight Below (2006)
12521  Just Another Love Story (Kærlighed på film) (2…
11230  OSS 117: Cairo, Nest of Spies (OSS 117: Le Cai…
13484                                   Love Streams (1984)

                      genres  \
12005           Comedy|Drama
13192          Action|Comedy
10659         Crime|Thriller
12337                  Crime
10639   Crime|Drama|Thriller
```

```
13158                   Comedy|Drama
10650  Action|Adventure|Drama|Romance
12521             Crime|Drama|Thriller
11230           Adventure|Comedy|Crime
13484                   Comedy|Drama

                                                clean_title
12005                               Bucket List  The  2007
13192                             Observe and Report  2009
10659                                    16 Blocks  2006
12337                             American Crime  An  2007
10639                                     Firewall  2006
13158                             Sunshine Cleaning  2008
10650                                  Eight Below  2006
12521  Just Another Love Story  K rlighed p  film   2…
11230  OSS 117  Cairo  Nest of Spies  OSS 117  Le Cai…
13484                                 Love Streams  1984
```

[ ]:
```python
#Building a recommendation function
```

[ ]:
```python
def find_similar_movies(movie_id):
  similar_users=ratings[(ratings['movieId']==movie_id) &␣
↪(ratings['rating']>4)]['userId'].unique()
  similar_user_recs=ratings[(ratings['userId'].isin(similar_users)) &␣
↪(ratings['rating']>4)]['movieId']

  similar_user_recs=similar_user_recs.value_counts()/len(similar_users)
  similar_user_recs=similar_user_recs[similar_user_recs > .10]

  all_users=ratings[(ratings['movieId'].isin(similar_user_recs.index)) &␣
↪(ratings['rating'] > 4)]
  all_user_recs=all_users['movieId'].value_counts()/len(all_users['userId'].
↪unique())

  percentages=pd.concat([similar_user_recs,all_user_recs],axis=1)
  percentages.columns=['similar','all']

  percentages['scores']=percentages['similar']/percentages['all']
  percentages=percentages.sort_values('scores',ascending=False)

  return percentages.head(10).
↪merge(movies,left_index=True,right_on='movieId')[['scores','title','genres']]
```

[ ]:
```python

```

##CREATING AN INTERACTIVE RECOMMENDATION WIDGET

[ ]:

```
import ipywidgets as  widgets
movie_input_name=widgets.Text(
    value='Toy Story',
    description='Movie Title:',
    disabled=False
)

recommendation_list=widgets.Output()

def on_type(data):
  with recommendation_list:
    recommendation_list.clear_output()
    title=data['new']
    if len(title) > 5:
      results=search(title)
      movie_id=results.iloc[0]['movieId']
      display(find_similar_movies(movie_id))

movie_input_name.observe(on_type,names='value')
display(movie_input_name,recommendation_list)
```

```
Text(value='Toy Story', description='Movie Title:')
```

```
Output()
```

[ ]:

```
```