Amazon

November 21, 2022

1 import required libraries

[60]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import re
import seaborn as sns
%matplotlib inline
```

2 Import the Data

[61]:
```
df = pd.read_csv('Amazon - Movies and TV Ratings.csv')
df.head()
```

[61]:

	user_id	Movie1	Movie2	Movie3	Movie4	Movie5	Movie6	Movie7	\
0	A3R5OBKS7OM2IR	5.0	5.0	NaN	NaN	NaN	NaN	NaN	
1	AH3QC2PC1VTGP	NaN	NaN	2.0	NaN	NaN	NaN	NaN	
2	A3LKP6WPMP9UKX	NaN	NaN	NaN	5.0	NaN	NaN	NaN	
3	AVIY68KEPQ5ZD	NaN	NaN	NaN	5.0	NaN	NaN	NaN	
4	A1CV1WROP5KTTW	NaN	NaN	NaN	NaN	5.0	NaN	NaN	

	Movie8	Movie9	…	Movie197	Movie198	Movie199	Movie200	Movie201	\
0	NaN	NaN	…	NaN	NaN	NaN	NaN	NaN	
1	NaN	NaN	…	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	…	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	…	NaN	NaN	NaN	NaN	NaN	
4	NaN	NaN	…	NaN	NaN	NaN	NaN	NaN	

	Movie202	Movie203	Movie204	Movie205	Movie206
0	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN

```
[5 rows x 207 columns]
```

[62]:
```
df.info
```

[62]:
```
<bound method DataFrame.info of             user_id  Movie1  Movie2  Movie3
Movie4  Movie5  Movie6  Movie7  \
0     A3R5OBKS7OM2IR     5.0     5.0     NaN     NaN     NaN     NaN     NaN
1      AH3QC2PC1VTGP     NaN     NaN     2.0     NaN     NaN     NaN     NaN
2     A3LKP6WPMP9UKX     NaN     NaN     NaN     5.0     NaN     NaN     NaN
3      AVIY68KEPQ5ZD     NaN     NaN     NaN     5.0     NaN     NaN     NaN
4     A1CV1WROP5KTTW     NaN     NaN     NaN     NaN     5.0     NaN     NaN
…                ...     …       …       …       …       …       …       …
4843  A1IMQ9WMFYKWH5     NaN     NaN     NaN     NaN     NaN     NaN     NaN
4844  A1KLIKPUF5E88I     NaN     NaN     NaN     NaN     NaN     NaN     NaN
4845   A5HG6WFZLO10D     NaN     NaN     NaN     NaN     NaN     NaN     NaN
4846  A3UU690TWXCG1X     NaN     NaN     NaN     NaN     NaN     NaN     NaN
4847   AI4J762YI6S06     NaN     NaN     NaN     NaN     NaN     NaN     NaN

      Movie8  Movie9  …  Movie197  Movie198  Movie199  Movie200  Movie201  \
0        NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
1        NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
2        NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
3        NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
4        NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
…       …       …  …         …         …         …         …         …
4843     NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
4844     NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
4845     NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
4846     NaN     NaN  …       NaN       NaN       NaN       NaN       NaN
4847     NaN     NaN  …       NaN       NaN       NaN       NaN       NaN

      Movie202  Movie203  Movie204  Movie205  Movie206
0          NaN       NaN       NaN       NaN       NaN
1          NaN       NaN       NaN       NaN       NaN
2          NaN       NaN       NaN       NaN       NaN
3          NaN       NaN       NaN       NaN       NaN
4          NaN       NaN       NaN       NaN       NaN
…         …         …         …         …         …
4843       NaN       NaN       NaN       NaN       5.0
4844       NaN       NaN       NaN       NaN       5.0
4845       NaN       NaN       NaN       NaN       5.0
4846       NaN       NaN       NaN       NaN       5.0
4847       NaN       NaN       NaN       NaN       5.0

[4848 rows x 207 columns]>
```

[63]:
```
df.shape
```

[63]: (4848, 207)

[64]:
```
df.size
```

[64]: 1003536

[65]:
```
df.dtypes
```

[65]:
```
user_id      object
Movie1      float64
Movie2      float64
Movie3      float64
Movie4      float64
              …
Movie202    float64
Movie203    float64
Movie204    float64
Movie205    float64
Movie206    float64
Length: 207, dtype: object
```

[66]:
```
df.count()
```

[66]:
```
user_id     4848
Movie1         1
Movie2         1
Movie3         1
Movie4         2
             …
Movie202       6
Movie203       1
Movie204       8
Movie205      35
Movie206      13
Length: 207, dtype: int64
```

3 maximum number of views

[67]:
```
df.describe().T["count"].sort_values(ascending=False)[0:6]
```

[67]:
```
Movie127    2313.0
Movie140     578.0
Movie16      320.0
Movie103     272.0
Movie29      243.0
```

```
Movie91      128.0
Name: count, dtype: float64
```

[68]:
```
df.index
```

[68]:
```
RangeIndex(start=0, stop=4848, step=1)
```

[69]:
```
df.columns
```

[69]:
```
Index(['user_id', 'Movie1', 'Movie2', 'Movie3', 'Movie4', 'Movie5', 'Movie6',
       'Movie7', 'Movie8', 'Movie9',
       …
       'Movie197', 'Movie198', 'Movie199', 'Movie200', 'Movie201', 'Movie202',
       'Movie203', 'Movie204', 'Movie205', 'Movie206'],
      dtype='object', length=207)
```

[70]:
```
df_filtered = df.fillna(value=0)
df_filtered
```

[70]:

	user_id	Movie1	Movie2	Movie3	Movie4	Movie5	Movie6	Movie7	\
0	A3R5OBKS7OM2IR	5.0	5.0	0.0	0.0	0.0	0.0	0.0	
1	AH3QC2PC1VTGP	0.0	0.0	2.0	0.0	0.0	0.0	0.0	
2	A3LKP6WPMP9UKX	0.0	0.0	0.0	5.0	0.0	0.0	0.0	
3	AVIY68KEPQ5ZD	0.0	0.0	0.0	5.0	0.0	0.0	0.0	
4	A1CV1WROP5KTTW	0.0	0.0	0.0	0.0	5.0	0.0	0.0	
…	…	…	…	…	…	…	…	…	
4843	A1IMQ9WMFYKWH5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4844	A1KLIKPUF5E88I	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4845	A5HG6WFZLO1OD	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4846	A3UU690TWXCG1X	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4847	AI4J762YI6SO6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

	Movie8	Movie9	…	Movie197	Movie198	Movie199	Movie200	Movie201	\
0	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
2	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
4	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
…	…	…	…	…	…	…	…	…	
4843	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
4844	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
4845	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
4846	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	
4847	0.0	0.0	…	0.0	0.0	0.0	0.0	0.0	

	Movie202	Movie203	Movie204	Movie205	Movie206
0	0.0	0.0	0.0	0.0	0.0

```
1          0.0       0.0       0.0       0.0       0.0
2          0.0       0.0       0.0       0.0       0.0
3          0.0       0.0       0.0       0.0       0.0
4          0.0       0.0       0.0       0.0       0.0
…          …         …         …         …         …
4843       0.0       0.0       0.0       0.0       5.0
4844       0.0       0.0       0.0       0.0       5.0
4845       0.0       0.0       0.0       0.0       5.0
4846       0.0       0.0       0.0       0.0       5.0
4847       0.0       0.0       0.0       0.0       5.0

[4848 rows x 207 columns]
```

[71]:
```
df_filtered1 = df_filtered.drop(columns='user_id')
df_filtered1
```

[71]:
```
      Movie1  Movie2  Movie3  Movie4  Movie5  Movie6  Movie7  Movie8  Movie9  \
0        5.0     5.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
1        0.0     0.0     2.0     0.0     0.0     0.0     0.0     0.0     0.0
2        0.0     0.0     0.0     5.0     0.0     0.0     0.0     0.0     0.0
3        0.0     0.0     0.0     5.0     0.0     0.0     0.0     0.0     0.0
4        0.0     0.0     0.0     0.0     5.0     0.0     0.0     0.0     0.0
…        …       …       …       …       …       …       …       …       …
4843     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
4844     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
4845     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
4846     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
4847     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0

      Movie10  …  Movie197  Movie198  Movie199  Movie200  Movie201  \
0         0.0  …       0.0       0.0       0.0       0.0       0.0
1         0.0  …       0.0       0.0       0.0       0.0       0.0
2         0.0  …       0.0       0.0       0.0       0.0       0.0
3         0.0  …       0.0       0.0       0.0       0.0       0.0
4         0.0  …       0.0       0.0       0.0       0.0       0.0
…         …    …       …         …         …         …         …
4843      0.0  …       0.0       0.0       0.0       0.0       0.0
4844      0.0  …       0.0       0.0       0.0       0.0       0.0
4845      0.0  …       0.0       0.0       0.0       0.0       0.0
4846      0.0  …       0.0       0.0       0.0       0.0       0.0
4847      0.0  …       0.0       0.0       0.0       0.0       0.0

      Movie202  Movie203  Movie204  Movie205  Movie206
0          0.0       0.0       0.0       0.0       0.0
1          0.0       0.0       0.0       0.0       0.0
2          0.0       0.0       0.0       0.0       0.0
3          0.0       0.0       0.0       0.0       0.0
```

```
4          0.0       0.0       0.0       0.0       0.0
…          …         …         …         …         …
4843       0.0       0.0       0.0       0.0       5.0
4844       0.0       0.0       0.0       0.0       5.0
4845       0.0       0.0       0.0       0.0       5.0
4846       0.0       0.0       0.0       0.0       5.0
4847       0.0       0.0       0.0       0.0       5.0

[4848 rows x 206 columns]
```

[72]:
```
df_max_views = df_filtered1.sum()
df_max_views
```

[72]:
```
Movie1        5.0
Movie2        5.0
Movie3        2.0
Movie4       10.0
Movie5      119.0
             …
Movie202     26.0
Movie203      3.0
Movie204     35.0
Movie205    162.0
Movie206     64.0
Length: 206, dtype: float64
```

4 Which movies have maximum views/ratings?

[73]:
```
max_views = df_max_views.argmax()
max_views
```

[73]:
```
126
```

[74]:
```
len(df_max_views.index)
```

[74]:
```
206
```

5 What is the average rating for each movie? Define the top 5 movies with the maximum ratings

[75]:
```
Average_rating_of_each_movie = sum(df_max_views)/len(df_max_views.index)
Average_rating_of_each_movie
```

[75]:
```
106.44660194174757
```

[76]:
```
amazon_df = pd.DataFrame(df_max_views)
amazon_df.head()
```

[76]:

	0
Movie1	5.0
Movie2	5.0
Movie3	2.0
Movie4	10.0
Movie5	119.0

[77]:
```
amazon_df.columns=['rating']
```

[78]:
```
amazon_df.index
```

[78]:
```
Index(['Movie1', 'Movie2', 'Movie3', 'Movie4', 'Movie5', 'Movie6', 'Movie7',
       'Movie8', 'Movie9', 'Movie10',
       …
       'Movie197', 'Movie198', 'Movie199', 'Movie200', 'Movie201', 'Movie202',
       'Movie203', 'Movie204', 'Movie205', 'Movie206'],
      dtype='object', length=206)
```

[79]:
```
amazon_df.nsmallest(5,'rating')
```

[79]:

	rating
Movie45	1.0
Movie58	1.0
Movie60	1.0
Movie67	1.0
Movie69	1.0

6 Divide the data into training and test data

[80]:
```
import sklearn
```

[81]:
```
from sklearn.model_selection import train_test_split
```

[82]:
```
trainset, testset = train_test_split(df, test_size=0.25)
```

7 Build a recommendation model on training data

[83]:
```
import surprise
from surprise import Reader
from surprise import accuracy
from surprise import Dataset
from surprise.model_selection import train_test_split
from surprise import SVD
from surprise.model_selection import cross_validate
```

[84]:
```
from surprise import SVD
```

[85]:
```
df_melt = df.melt(id_vars = df.columns[0],value_vars=df.columns[1:],
                  var_name="Movies",value_name="Rating")
df_melt
```

[85]:

	user_id	Movies	Rating
0	A3R5OBKS7OM2IR	Movie1	5.0
1	AH3QC2PC1VTGP	Movie1	NaN
2	A3LKP6WPMP9UKX	Movie1	NaN
3	AVIY68KEPQ5ZD	Movie1	NaN
4	A1CV1WROP5KTTW	Movie1	NaN
…	…	…	…
998683	A1IMQ9WMFYKWH5	Movie206	5.0
998684	A1KLIKPUF5E88I	Movie206	5.0
998685	A5HG6WFZLO10D	Movie206	5.0
998686	A3UU690TWXCG1X	Movie206	5.0
998687	AI4J762YI6S06	Movie206	5.0

[998688 rows x 3 columns]

[86]:
```
rdr = Reader()
df = Dataset.load_from_df(df_melt.fillna(0),reader = rdr)
df
```

[86]:
```
<surprise.dataset.DatasetAutoFolds at 0x7f54d5390090>
```

[87]:
```
trainset, testset = train_test_split(df,test_size = 0.25)
```

[88]:
```
 from sklearn.decomposition import TruncatedSVD
```

[89]:
```
svd = SVD()
svd.fit(trainset)
```

[89]: <surprise.prediction_algorithms.matrix_factorization.SVD at 0x7f54cb1f1850>

8 Make predictions on the test data

[90]:
```
pred = svd.test(testset)
```

[91]:
```
accuracy.rmse(pred)
```

```
RMSE: 1.0266
```

[91]: 1.0265592501512613

[92]:
```
accuracy.mae(pred)
```

```
MAE:  1.0122
```

[92]: 1.0122054491299677

[93]:
```
cross_validate(svd, df, measures = ['RMSE', 'MAE'], cv = 3, verbose = True)
```

```
Evaluating RMSE, MAE of algorithm SVD on 3 split(s).

                  Fold 1  Fold 2  Fold 3  Mean    Std
RMSE (testset)    1.0256  1.0265  1.0262  1.0261  0.0004
MAE (testset)     1.0118  1.0122  1.0122  1.0121  0.0002
Fit time          36.51   38.08   40.78   38.46   1.76
Test time         4.00    3.44    3.85    3.76    0.24
```

[93]:
```
{'test_rmse': array([1.02564356, 1.02650914, 1.02624722]),
 'test_mae': array([1.01183495, 1.01220878, 1.01224944]),
 'fit_time': (36.509862184524536, 38.084420680999756, 40.77697539329529),
 'test_time': (4.00281548500061, 3.4420347213745117, 3.8467113971710205)}
```

[ ]: