In [1]:

```
#importing libraries
import pandas as pd
import numpy as np
import re
import surprise
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
#Data Preperation
data = pd.read_csv('Amazon - Movies and TV Ratings.csv')
data.head()
```

Out[3]:

	user_id	Movie1	Movie2	Movie3	Movie4	Movie5	Movie6	Movie7	Movie8	Mov
0	A3R5OBKS7OM2IR	5.0	5.0	NaN	NaN	NaN	NaN	NaN	NaN	N
1	AH3QC2PC1VTGP	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	N
2	A3LKP6WPMP9UKX	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	N
3	AVIY68KEPQ5ZD	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	N
4	A1CV1WROP5KTTW	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN	N

5 rows × 207 columns

In [4]:

```
data.shape
```

Out[4]:

```
(4848, 207)
```

In [5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4848 entries, 0 to 4847
Columns: 207 entries, user_id to Movie206
dtypes: float64(206), object(1)
memory usage: 7.7+ MB
```

In [9]:

```
#Checking which movies having maximum views
data.describe().T['count'].sort_values(ascending=False)[:2].to_frame()
```

Out[9]:

	count
Movie127	2313.0
Movie140	578.0

In [11]:

```
#Checking which movies having maximum ratings
data.drop('user_id',axis=1).sum().sort_values(ascending=False)[:2].to_frame()
```

Out[11]:

	0
Movie127	9511.0
Movie140	2794.0

In [12]:

```
#Top 5 movies with maximum ratings
data.drop('user_id',axis=1).mean().sort_values(ascending=False)[:5].to_frame()
```

Out[12]:

	0
Movie1	5.0
Movie55	5.0
Movie131	5.0
Movie132	5.0
Movie133	5.0

In [13]:

```
#Top 5 movies with the least audience
data.describe().T['count'].sort_values(ascending=True)[:5].to_frame()
```

Out[13]:

	count
Movie1	1.0
Movie71	1.0
Movie145	1.0
Movie69	1.0
Movie68	1.0

In [18]:

```
#Recommendation Model
from surprise import Dataset
df = data.melt(id_vars = data.columns[0],value_vars=data.columns[1:],var_name="Movies",value_name="Rating")
df
```

Out[18]:

	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 × 3 columns

In [24]:

```
from surprise import Reader
rd = Reader()
ds = Dataset.load_from_df(df.fillna(0),reader=rd)
ds
```

Out[24]:

```
<surprise.dataset.DatasetAutoFolds at 0x2864cd45088>
```

In [29]:

```
#Splitting the data into train and test datasets
from surprise.model_selection import train_test_split
trainset, testset = train_test_split(ds,test_size=0.25)
```

In [30]:

```
#Building a recommendation model on training data
from surprise import SVD
svd = SVD()
svd.fit(trainset)
```

Out[30]:

```
<surprise.prediction_algorithms.matrix_factorization.SVD at 0x2865ecb6808>
```

In [31]:

```
#predictions on the test data
pred = svd.test(testset)
```

In [32]:

```
from surprise import accuracy
accuracy.rmse(pred)
```

```
RMSE: 1.0255
```

Out[32]:

```
1.0255192276925031
```

In [34]:

```
accuracy.mae(pred)
```

```
MAE:  1.0117
```

Out[34]:

```
1.011739839945971
```

In [36]:

```
from surprise.model_selection import cross_validate
cross_validate(svd, ds, 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)    1.0251  1.0260  1.0272  1.0261  0.0008  
MAE (testset)     1.0116  1.0120  1.0125  1.0120  0.0003  
Fit time          48.17   43.49   42.18   44.61   2.57    
Test time         3.81    3.01    3.46    3.43    0.33    
```

Out[36]:

```
{'test_rmse': array([1.02514184, 1.02596167, 1.02717048]),
 'test_mae': array([1.01162888, 1.01201032, 1.01245434]),
 'fit_time': (48.17183303833008, 43.48506188392639, 42.18087077140808),
 'test_time': (3.813087224960327, 3.0086162090301514, 3.457048177719116)}
```

In [ ]:

```

```