```
%matplotlib inline
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from ast import literal_eval
from sklearn.feature_extraction.text import TfidfVectorizer,
CountVectorizer
from sklearn.metrics.pairwise import linear_kernel, cosine_similarity
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.corpus import wordnet
from surprise import Reader, Dataset, SVD
from surprise.model_selection import cross_validate

import warnings; warnings.simplefilter('ignore')

print("Import Success!!")
```

```
Import Success!!
```

```
reader = Reader()

ratings = pd.read_csv('Data-Asset/archive/ratings_small.csv')
ratings.head()
```

```
   userId  movieId  rating   timestamp
0       1       31     2.5  1260759144
1       1     1029     3.0  1260759179
2       1     1061     3.0  1260759182
3       1     1129     2.0  1260759185
4       1     1172     4.0  1260759205
```

```
data = Dataset.load_from_df(ratings[['userId',
                                     'movieId',
                                     'rating']], reader)

svd = SVD()
cross_validate(svd, data, measures=['RMSE', 'MAE'], cv=10,
verbose=True)
```

```
Evaluating RMSE, MAE of algorithm SVD on 10 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Fold 6  Fold
7  Fold 8  Fold 9  Fold 10 Mean    Std
RMSE (testset)    0.8925  0.8929  0.8992  0.8856  0.8880  0.8906
0.8938  0.8911  0.8922  0.8817  0.8908  0.0045
MAE (testset)     0.6880  0.6874  0.6939  0.6801  0.6838  0.6862
0.6896  0.6870  0.6878  0.6731  0.6857  0.0054
Fit time          5.80    5.79    6.43    6.35    5.91    5.94    6.33
```

```
6.21    6.12    6.26    6.11    0.23
Test time          0.26    0.07    0.08    0.06    0.07    0.30    0.07
0.10    0.08    0.08    0.12    0.08

{'test_rmse': array([0.89249452, 0.89285824, 0.89919635, 0.88562408,
0.88798022,
        0.89058695, 0.89380026, 0.89107543, 0.89215089, 0.88174113]),
 'test_mae': array([0.68796473, 0.68736413, 0.6939108 , 0.68010653,
0.68384702,
        0.68616022, 0.68959061, 0.686951  , 0.68778201, 0.67310569]),
 'fit_time': (5.801063776016235,
  5.790342330932617,
  6.426440238952637,
  6.348586559295654,
  5.910297870635986,
  5.939616441726685,
  6.332097768783569,
  6.214759349822998,
  6.115145683288574,
  6.264696359634399),
 'test_time': (0.261411190032959,
  0.07479143142700195,
  0.08278894424438477,
  0.06205630302429199,
  0.07080984115600586,
  0.3041880130767822,
  0.06781840324401855,
  0.10077166557312012,
  0.07917976379394531,
  0.07903027534484863)}

trainset = data.build_full_trainset()
svd.fit(trainset)

<surprise.prediction_algorithms.matrix_factorization.SVD at
0x1f0889787f0>

ratings[ratings['userId'] == 1]

    userId  movieId  rating   timestamp
0        1       31     2.5  1260759144
1        1     1029     3.0  1260759179
2        1     1061     3.0  1260759182
3        1     1129     2.0  1260759185
4        1     1172     4.0  1260759205
5        1     1263     2.0  1260759151
6        1     1287     2.0  1260759187
7        1     1293     2.0  1260759148
8        1     1339     3.5  1260759125
9        1     1343     2.0  1260759131
10       1     1371     2.5  1260759135
```

```
11       1     1405     1.0  1260759203
12       1     1953     4.0  1260759191
13       1     2105     4.0  1260759139
14       1     2150     3.0  1260759194
15       1     2193     2.0  1260759198
16       1     2294     2.0  1260759108
17       1     2455     2.5  1260759113
18       1     2968     1.0  1260759200
19       1     3671     3.0  1260759117
```

```
svd.predict(1, 2455, 3)
```

```
Prediction(uid=1, iid=2455, r_ui=3, est=2.4970420749191016,
details={'was_impossible': False})
```