Consider the ratings dataset below containing data on UserID, MovieID, Rating, and Timestamp. Each line of this file represents one rating of one movie by one user and has the following format: UserID::MovieID::Rating::Timestamp

Ratings are made on a 5 star scale with half star increments.

UserID: represents the ID of the user

MovieID: represents the ID of the movie

Timestamp: represents seconds from midnight Coordinated Universal Time (UTC) of January 1, 1970.

Objective: Predict a movie-movie recommendation model.

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_csv('Recommend.csv',names=['user_id', 'movie_id', 'rating','timestamp'])

df
```

Out[2]:

	user_id	movie_id	rating	timestamp
0	196	242	3	881250949
1	186	302	3	891717742
2	22	377	1	878887116
3	244	51	2	880606923
4	166	346	1	886397596
...	...	...	...	...
99995	880	476	3	880175444
99996	716	204	5	879795543
99997	276	1090	1	874795795

	user_id	movie_id	rating	timestamp
99998	13	225	2	882399156
99999	12	203	3	879959583

100000 rows × 4 columns

Create a train test split of 75/25 by declaring 75/25 declaring number of users and movies

In [3]:
```
from sklearn.model_selection import train_test_split
n_users = df.user_id.unique().shape[0]
n_movies = df.movie_id.unique().shape[0]
train_data, test_data= train_test_split(df,test_size=0.25)
```

Populate the train matrix(user_id x movie_id) with ratings such that [user_id index, movie_id index]= given rating

In [6]:
```
train_data_matrix = np.zeros((n_users, n_movies))
for line in train_data.itertuples():
    train_data_matrix[line[1]-1, line[2]-1]= line[3]
train_data_matrix
```

Out[6]:
```
array([[5., 3., 4., ..., 0., 0., 0.],
       [4., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [5., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 5., 0., ..., 0., 0., 0.]])
```

Populate the test matrix(user_id x movie_id) with ratings such that [user-id index, movie_id index] = given rating

In [7]:
```
test_data_matrix = np.zeros((n_users, n_movies))
for line in test_data.itertuples():
    test_data_matrix[line[1]-1, line[2]-1]= line[3]
test_data_matrix
```

Out[7]:
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
```

```
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

Create a cosine similarity matrices for movies and predict a movie-movie recommendation model

In [10]:
```
from sklearn.metrics import pairwise_distances
movie_similarity = pairwise_distances(train_data_matrix.T,metric='cosine')
movie_pred = train_data_matrix.dot(movie_similarity)
np.array([np.abs(movie_similarity).sum(axis=1)])
movie_pred
```

Out[10]:
```
array([[529.34336941, 563.79740673, 596.20312653, ..., 735.60471439,
        715.68515867, 704.51924654],
       [119.26521569, 142.02490873, 140.16706549, ..., 161.12038293,
        162.18067772, 162.90120315],
       [102.65652807, 108.25259751, 107.03939949, ..., 112.1286667 ,
        118.29210216, 119.51285059],
       ...,
       [ 43.43428632,  57.07284422,  56.19586081, ...,  72.74203747,
         70.25985418,  70.23903141],
       [199.53548344, 220.14902596, 236.23080468, ..., 267.37143781,
        262.77536636, 265.69586961],
       [292.02641008, 297.84766669, 329.29925712, ..., 415.        ,
        397.21555288, 399.70734371]])
```

In [ ]: