CS 6375 HW 2

Collaborative Filtering

Parker Whitehead

pmw180000@utdallas.edu

Import needed libraries

In [ ]:
```
import fileutility as ft
from collaborativefiltering import MemoryBasedCollaborativeFiltering as MBCF
import numpy as np
import math
import os
np.version.version
```

Out[ ]:
```
'1.22.3'
```

Grab number of movies in movie_titles.txt Grab dict of users:index in TestingRatings.txt

In [ ]:
```
num_movies = ft.extract_num_movies('netflix\movie_titles.txt')
print(f'Number of movies: {num_movies}')

user_count, user_dict = ft.extract_user_dict('netflix\TrainingRatings.txt')
print(f'Number of users: {user_count} \nNumber of elements in dict: {len(user_dict)}')
```

```
Number of movies: 17770
Number of users: 28978
Number of elements in dict: 28978
```

Construct Train matrix of users & ratings.

In [ ]:
```
from numpy import float32


V = np.zeros((user_count,num_movies+1), dtype=float32) # Since movie index starting at
with open('netflix\TrainingRatings.txt') as f:
    for line in f:
        line = line.split(',')
        r = user_dict[int(line[1])]
        c = int(line[0])
        rating = float(line[2])
        V[r,c] = rating
print(f'Num of rows: {V.shape[0]}\nNum of columns: {V.shape[1]}')
```

```
Num of rows: 28978
Num of columns: 17771
```

Initialize and Train collaborative filtering object.

In [ ]:
```
cf = MBCF()
cf.fit(V)
```

```
sum
count
avg
copy
setzeros
diags
placement
diff
power
diff empty
diff square sum
initial denom
sqrt
reciprocal
created W
```

In [ ]:
```

```

Create the Test Dataset

In [ ]:
```
num_test_users = ft.get_num_lines('netflix\TestingRatings.txt')

Test_X = np.zeros((num_test_users,2),dtype=int)
Test_Y = np.zeros(num_test_users)

index = 0
with open('netflix\TestingRatings.txt') as f:
    for line in f:
        line = line.split(',')
        Test_X[index,0] = int(line[0])
        Test_X[index,1] = user_dict[int(line[1])]
        Test_Y[index] = float(line[2])
        index+=1
print(f'Test_X shape: {Test_X.shape}\nTest_Y shape{Test_Y.shape}')
print(f'First line: {Test_X[0][0]}, {Test_X[0][1]}, {Test_Y[0]}')
```

```
Test_X shape: (100478, 2)
Test_Y shape(100478,)
First line: 8, 14771, 1.0
```

Test dataset on model

In [ ]:
```
MAE, RMSE = cf.predict(Test_X, Test_Y)

print(f'Mean Absolute Error: {MAE:.4f}')
print(f'Root Mean Standard Error: {RMSE:.4f}')
```

```
Mean Absolute Error: 0.7486
Root Mean Standard Error: 0.9434
```