```
#Walter 18/7/2019

#A small movie recommender using item-based collaborative filtering

#the data (the MovieLens 100k data set) is from the grouplens.org website


import pandas as pd

import numpy as np

r_cols = ['user_id', 'movie_id', 'rating']

ratings = pd.read_csv('ml-100k/u.data', sep='\t', names=r_cols, usecols=range(3), encoding="ISO-8859-1")

m_cols = ['movie_id', 'title']

movies = pd.read_csv('ml-100k/u.item', sep='|', names=m_cols, usecols=range(2), encoding="ISO-8859-1")

ratings = pd.merge(movies, ratings)

#ratings.head()


movieStats = ratings.groupby('title').agg({'rating': [np.size, np.mean]})


movieRatings = ratings.pivot_table(index=['user_id'],columns=['title'],values='rating')

import pandas as pd


r_cols = ['user_id', 'movie_id', 'rating']

ratings = pd.read_csv('ml-100k/u.data', sep='\t', names=r_cols, usecols=range(3), encoding="ISO-8859-1")


m_cols = ['movie_id', 'title']

movies = pd.read_csv('ml-100k/u.item', sep='|', names=m_cols, usecols=range(2), encoding="ISO-8859-1")


ratings = pd.merge(movies, ratings)


#filtering out users that have rted too few or too many movies
```

```
#minm = 75
#maxm = 140
#ratings = ratings.groupby('user_id').filter(lambda x : len(x)< minm or len(x)< maxm)

userRatings = ratings.pivot_table(index=['user_id'],columns=['title'],values='rating')

#userRatings.head()

corrMatrix = userRatings.corr(method='pearson', min_periods=75)

#just to get rid of the spurious data

#corrMatrix.head()

myRatings = userRatings.loc[0].dropna()
#myRatings

#Now, for each movie I rated, I retrieve the list of similar movies from the correlation matrix.
#I then scale those correlation scores by how well I rated the movie they are similar to, so
#movies similar to ones I liked count more than movies similar to ones I hated.
#Note I created a fictitious user in the ratingsdataset

simCandidates = pd.Series(dtype="string")
for i in range(0, len(myRatings.index)):
    print ("Adding sims for " + myRatings.index[i] + "...")
    # Retrieve similar movies to this one that I rated
    sims = corrMatrix[myRatings.index[i]].dropna()
    # Now scale its similarity by how well I rated this movie
    y = myRatings[i]
    if y < 3:
        y = -1 * y
```

```
    elif y > 3:
        y = 1.1 * y
    sims = sims.map(lambda x: x * y)
    # Add the score to the list of similarity candidates
    simCandidates = simCandidates.append(sims)

print ("sorting...")
simCandidates.sort_values(inplace = True, ascending = False)
#print (simCandidates.head(10))

simCandidates = simCandidates.groupby(simCandidates.index).sum()

simCandidates.sort_values(inplace = True, ascending = False)
#simCandidates.head(10)
#Filtering out the movies that I've already rated
filteredSims = simCandidates.drop(myRatings.index)

filteredSims.head(10)
```

```
Return of the Jedi (1983)                 7.178172
Raiders of the Lost Ark (1981)            5.519700
Bridge on the River Kwai, The (1957)      3.582271
Indiana Jones and the Last Crusade (1989)  3.488028
Back to the Future (1985)                 3.357941
Sting, The (1973)                         3.329843
Batman (1989)                             3.246179
Cinderella (1950)                         3.245412
Field of Dreams (1989)                    3.222311
Con Air (1997)                            3.204525
```