```
# -*- coding: utf-8 -*-
"""
Created on Wed Aug 19 14:28:12 2018

@author: Walter Stevens

Simple example of KNN (K-Nearest-Neighbors)using MovieLens data.

"""
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))
#ratings.head()



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

#movieProperties.head()

#normalising the number of ratings

movieNumRatings = pd.DataFrame(movieProperties['rating']['size'])
movieNormalizedNumRatings = movieNumRatings.apply(lambda x: (x - np.min(x)) / (np.max(x) -
np.min(x)))
#movieNormalizedNumRatings.head()

movieDict = {}
```

```
#iterating through every line in the file

with open(r'ml-100k/u.item') as f:
    temp = ''
    for line in f:
        #line.decode("ISO-8859-1")
        fields = line.rstrip('\n').split('|')
        movieID = int(fields[0])
        name = fields[1]
        genres = fields[5:25]
        genres = map(int, genres)
        movieDict[movieID] = (name, np.array(list(genres)),
movieNormalizedNumRatings.loc[movieID].get('size'),
movieProperties.loc[movieID].rating.get('mean'))

from scipy import spatial

def ComputeDistance(a, b):
    genresA = a[1]
    genresB = b[1]
    genreDistance = spatial.distance.cosine(genresA, genresB)
    popularityA = a[2]
    popularityB = b[2]
    popularityDistance = abs(popularityA - popularityB)
    return genreDistance + popularityDistance

# An arbitrarily chosen distance function!

#ComputeDistance(movieDict[3], movieDict[5])

import operator
```

```
def getNeighbors(movieID, K):
    distances = []
    for movie in movieDict:
        if (movie != movieID):
            dist = ComputeDistance(movieDict[movieID], movieDict[movie])
            distances.append((movie, dist))
    distances.sort(key=operator.itemgetter(1))
    neighbors = []
    for x in range(K):
        neighbors.append(distances[x][0])
    return neighbors

K = 15
avgRating = 0
neighbors = getNeighbors(1, K)
for neighbor in neighbors:
    avgRating += movieDict[neighbor][3]
    print (movieDict[neighbor][0] + " " + str(movieDict[neighbor][3]))

avgRating /= K
```