计算皮尔逊相关系数

- import json
- import numpy as np
- '''　　功能作用：计算皮尔逊相关系数　　欧氏距离分数是一个非常好的指标，但它也有一些缺点。因此，皮尔逊相关系数常用于推荐引擎'''# 计算user1和user2的皮尔逊相关系数def pearson_score(dataset, user1, user2):
- if user1 not in dataset:
- raise TypeError('User ' + user1 + ' not present in the dataset')
- if user2 not in dataset:
- raise TypeError('User ' + user2 + ' not present in the dataset')
- # 提取两个用户均评过分的电影　rated_by_both = {}
- for item in dataset[user1]:
- if item in dataset[user2]:
- rated_by_both[item] = 1　num_ratings = len(rated_by_both)
- # 如果两个用户都没有评分，得分为0　if num_ratings == 0:
- return 0　# 计算相同评分电影的平方值之和　user1_sum = np.sum([dataset[user1][item] for item in rated_by_both])
- user2_sum = np.sum([dataset[user2][item] for item in rated_by_both])
- # 计算所有相同评分电影的评分的平方和　user1_squared_sum = np.sum([np.square(dataset[user1][item]) for item in rated_by_both])
- user2_squared_sum = np.sum([np.square(dataset[user2][item]) for item in rated_by_both])
- # 计算数据集的乘积之和　product_sum = np.sum([dataset[user1][item] * dataset[user2][item] for item in rated_by_both])
- # 计算皮尔逊相关度　Sxy = product_sum - (user1_sum * user2_sum / num_ratings)
- Sxx = user1_squared_sum - np.square(user1_sum) / num_ratings
- Syy = user2_squared_sum - np.square(user2_sum) / num_ratings
- #　考了分母为0的情况　if Sxx * Syy == 0:
- return 0　# 如果一切正常,返回皮尔逊系数　return Sxy / np.sqrt(Sxx * Syy)
- if __name__=='__main__':
- # 计算两个用户之间的皮尔逊系数　data_file = 'movie_ratings.json'　with open(data_file, 'r') as f:
- data = json.loads(f.read())
- user1 = 'John Carson'　user2 = 'Michelle Peterson'　print ("\nPearson score:")
- print (pearson_score(data, user1, user2))

幕布 - 思维概要整理工具