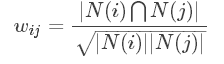
**实验报告**

在最基本的电影演员搜索中，基本上和以前做过的实验差不多，我们在此基础上加了一个排序功能，可以根据电影的评分或是评论数量进行排序。然后是分类检索功能，这个就是在HTML里加了几个<label>框，然后设定value，最后对相应的域进行组合搜索。然后是剧照搜索功能，在爬网页的时候我们就把每部电影的一些剧照保存下来，然后以一定的规则来命名并储存在图片库里，然后输入目标图片的本地地址，它就会将目标图片和图片库里的图片用一个基于sift的算法进行匹配，随后找出一张最相近的图片，根据前面决定好的命名规则，就可以找出该电影的相关信息。

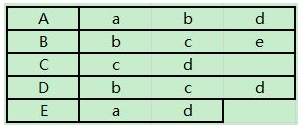
接着是个性化推荐，基于物品的协同过滤算法就是找到和“目标用户”喜欢的物品相似的物品，然后把相似的物品推荐给“目标用户”。

1计算物品之间的相似度

设|N(i)|表示喜欢物品i的用户，则物品i物品j的相似度为：

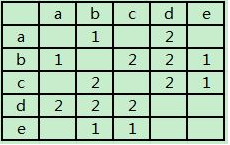


1.1首先需要建立用户物品倒排表。设用大写字母表示用户，小写字母表示物品，则建立的用户物品倒排表为：



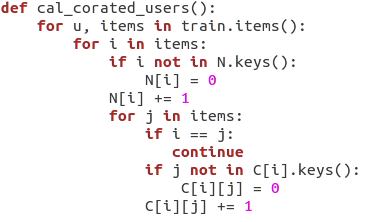
在python里存储成字典的形式，key是用户id，value是一个字典，这个字典的key是物品id，value是用户兴趣度。

1.2共现矩阵C表示同时喜欢两个物品的用户数，是根据用户物品倒排表计算出来的。如根据上面的用户物品倒排表可以计算出如下的共现矩阵C：



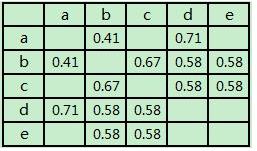
共现矩阵的对角线元素全为0，且是实对称稀疏矩阵。

实现代码如下：

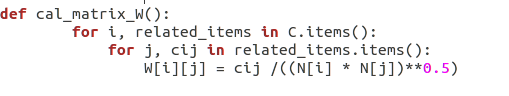


1.3共现矩阵C其实就是公式的分子，矩阵N表示喜欢某物品的用户数，那么余弦相似度矩阵很容易就计算出来了，示例的矩阵N，以及余弦相似度矩阵如下所示：





实现代码如下：

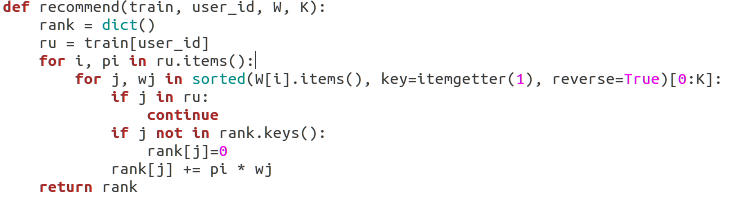


计算余弦相似度矩阵W比较费时，但由于重新计算的频率不高，所以可以做离线运算并用pickle模块存储在本地。

2根据物品的相似度和用户的历史行为给用户生成推荐列表

最终推荐的是什么物品，是由预测兴趣度决定的。物品j预测兴趣度=用户喜欢的物品i的兴趣度×物品i和物品j的相似度。

实现代码如下：



接着是根据人脸识别，图像搜索分为两步，首先是找出图片中的人脸，我们用了openCV中已经训练好的数据集，通过CascadeClassifier这个级联分类器类找出人脸。然后就是对找到的人脸进行匹配，要用到一个face\_recognization的人脸识别库。Load\_image函数将图片转化为一个numpy array，face\_encoding函数将这个numpy array变为一个128维的face encoding，然后用face\_compare函数把它与图片库中的进行对比，若其高于设置的阈值，则视为匹配成功。

分工：

黄晟原：数据爬取、建立索引、分类检索、剧照搜索。

刘乐典：推荐功能、界面、最终整合。

桂钰波：人脸识别、人脸匹配。

姜宇轩：电影搜索、排序。

使用方法：运行research.py即可