**《程序设计基础1》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **2019级计算机学院03班** | | | **姓名** | **Derrick Zhu** |
| **实验题目** | **基于协同过滤算法的电影推荐** | | | | | |
| **实验时间** | **10.10~10.24** | | **实验地点** | **D1421** | | |
| **实验成绩** |  | | **实验性质** | **□验证性 □设计性 ■综合性** | | |
| 教师评价：  □算法/实验过程正确；□源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | | |
| 一、实验目的  学习协同过滤算法，利用MoviesLens 1M数据集实现电影推荐。 | | | | | | |
| 二、实验项目内容  对MoviesLens1M数据集进行分析，完成如下功能： 1. 读取MoviesLens1M数据，存储为Pandas的DataFrame；  2. 将DataFrame数据改为自己代码所需要的结构；  3. 学习协同过滤算法，写出协同过滤算法相应的函数；  4. 输入两个用户，输出两个用户公共电影的评分；  5. 输入两个用户，计算出用户偏好的相似度；  6. 输入一位用户，输出与指定用户相似度最高的5位用户；  7. 输入一位用户，输出输出推荐电影；  8. 程序界面请自行设计，命令行、GUI、网页、手机App均可。 | | | | | | |
| 三、实验过程或算法（源程序）  #实验四-基于协同过滤算法的电影推荐  #计算机学院03班 朱绎彰 20194113  #2019.12.22  '''  实验要求：  1. 读取MoviesLens1M数据，存储为Pandas的DataFrame；  2. 将DataFrame数据改为自己代码所需要的结构；  3. 学习协同过滤算法，写出协同过滤算法相应的函数；  4. 输入两个用户，输出两个用户公共电影的评分；  5. 输入两个用户，计算出用户偏好的相似度；  6. 输入一位用户，输出与指定用户相似度最高的5位用户；  7. 输入一位用户，输出输出推荐电影；  8. 程序界面请自行设计，命令行、GUI、网页、手机App均可。  '''  import pandas as pd  # 数据读取和处理，合并在一个大的dataframe  rnames = ['user\_id', 'movie\_id', 'rating', 'timestamp']  ratings = pd.read\_csv('C:\\Users\\admin\\Desktop\\程序设计实验报告-4\\ml\_1m\\ml\_1m\\ratings.dat', \                        sep='::', header=None, names=rnames, usecols = [0,1,2],engine= 'python')  mnames = ['movie\_id', 'title', 'genres']  movies = pd.read\_csv('C:\\Users\\admin\Desktop\\程序设计实验报告-4\\ml\_1m\\ml\_1m\\movies.dat', \                       sep='::', header=None, names=mnames, usecols = [0,1],engine= 'python')  data = pd.merge(ratings, movies)  global df  # 转换成User-Item矩阵  df = ratings.pivot(index='user\_id', columns='movie\_id', values='rating')  # 以下是公用函数  # 构建共同的评分向量  def build\_xy(user\_id1, user\_id2):          bool\_array = df.loc[user\_id1].notnull() & df.loc[user\_id2].notnull()          return df.loc[user\_id1, bool\_array], df.loc[user\_id2, bool\_array]  # 余弦相似度  def cosine(user\_id1, user\_id2):          x, y = build\_xy(user\_id1, user\_id2)          # 分母          denominator = (sum(x\*x)\*sum(y\*y))\*\*0.5          try:              value = sum(x\*y)/denominator          except ZeroDivisionError:              value = 0          return value  # 计算最近的邻居  def computeNearestNeighbor(user\_id, metric='cosine', k=5):          """          metric: 度量函数          k:      返回k个邻居          返回：pd.Series，其中index是邻居名称，values是距离          """          return df.drop(user\_id).index.to\_series().apply(cosine, args=(user\_id,)).nlargest(k)  # 向给定用户推荐（返回：pd.Series）  def recommend(user\_id):      # 找到距离最近的用户id      nearest\_user\_id = computeNearestNeighbor(user\_id, metric='cosine').index[0]      print('最近邻用户id：' + str(nearest\_user\_id))      # 找出邻居评价过、但自己未曾评价的乐队（或商品）      # 结果：index是商品名称，values是评分      recommendList = df.loc[nearest\_user\_id, df.loc[user\_id].isnull() & \                             df.loc[nearest\_user\_id].notnull()].sort\_values()      print(recommendList)      return recommendList  # 下面进行实验要求功能的实现  # 1.输入两个用户，输出两个用户公共电影的评分；  def t1():      print('please input 2 users id:')      x=int(input())      y=int(input())      print(build\_xy(x,y))  # 2.输入两个用户，计算出用户偏好的相似度；  def t2():      print('please input 2 users`id and get the degree of similarity')      x=int(input())      y=int(input())      print(cosine(x,y))    # 3.输入一位用户，输出与指定用户相似度最高的5位用户；  def t3():      print('please input a usee`s id and get 5 other most similar users` id')      x=int(input())      print(computeNearestNeighbor(x))  # 4.输入一位用户，输出输出推荐电影；  def t4():      print('please input id and get the recommended movies~ ')      x=int(input())      print(recommend(x))    #封装调用  print('''  基于协同过滤算法的电影推荐有以下功能：  1. 输入两个用户，输出两个用户公共电影的评分；  2. 输入两个用户，计算出用户偏好的相似度；  3. 输入一位用户，输出与指定用户相似度最高的5位用户；  4. 输入一位用户，输出输出推荐电影；  输入‘i’表示调用第i个功能。  ''')  def t():      i=int(input())      if i==1:          t1()      if i==2:          t2()      if i==3:          t3()      if i==4:          t4()      else:          print('重新输入i可再次调用')          t()  t() | | | | | | |
| 四、实验结果及分析和（或）源程序调试过程  **4. 输入两个用户，输出两个用户公共电影的评分：**  C:\Users\admin\Documents\Tencent Files\2547859435\Image\C2C\AZB6)9QWI`A_L31$9C6L~G8.png  **5. 输入两个用户，计算出用户偏好的相似度：**  C:\Users\admin\Documents\Tencent Files\2547859435\Image\C2C\6J5{2F7{YT9O0L(X]@1AKO3.png  **6. 输入一位用户，输出与指定用户相似度最高的5位用户：**  C:\Users\admin\Documents\Tencent Files\2547859435\Image\C2C\%9J`U%NUH2[6CN{8H2HZ4CO.png  **7. 输入一位用户，输出输出推荐电影：**  C:\Users\admin\Documents\Tencent Files\2547859435\Image\C2C\1LBR28)HXK`)(S33UN04OMR.png  **8. 封装调用：**  C:\Users\admin\Documents\Tencent Files\2547859435\Image\C2C\([BHQ(NS)HAJ%_U5I_(F1T3.png | | | | | | |