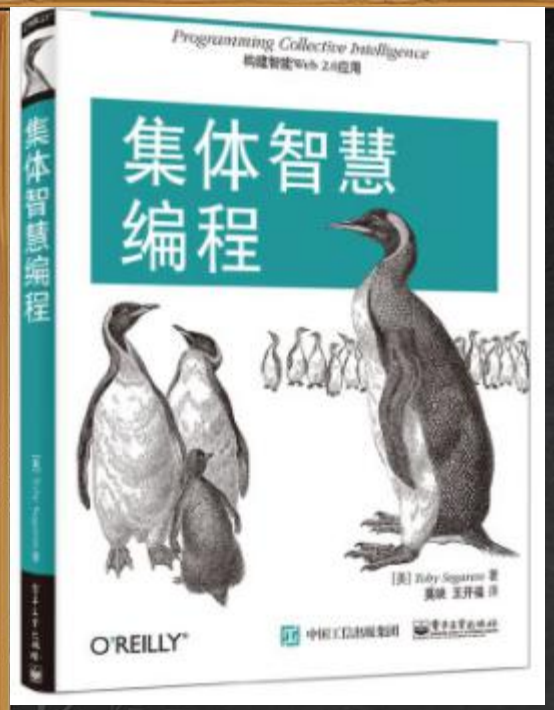


〝提供推荐〞

如何“人以群分，物以类聚”

祥水村东头@AI识堂

《集体智慧编程》读书笔记系列(对应第2章 pp7-28)

本次胶片内容、及涉及相关代码均可移步至Github进行下载

我的代码 Github 地址：
https://github.com/david-cal/
Reading-Note-For-Programming-Collective-Intelligence

目录

01

如何投其所好？

- 互联网世界处处是推荐
- 以“相同”找“不同”
- 偏好相似度评价方法

1 如何投其所好？——互联网世界处处是推荐

- 用已知推荐未知
- 基于用户消费行为大数据，为你推荐你尚未看到过、听到过、用到过的东西.....推荐技术可以帮电商平台实现拉新促活，因此在互联网世界中，到处充斥着各种推荐
- 印象最深刻的一次被推荐体验：因为腿部受伤，在某东上买了一副拐，结果第二天平台开始向我推荐轮椅。。。。
- 右边第1个截屏：某瓣上输入电视剧《大江大河2》后，会显示喜欢这部电视的人也喜欢的电视剧
- 右边第2个截屏：在某东上输入《集体智慧编程》，在推荐菜单中会显示其他相关学习书籍
- 暂时抛开其中的不合理性，这种推荐机制背后是怎样的原理？（注：教材中只讲解了最基础的理论模型，在实践中远比此要复杂）

1 如何投其所好？——以“相同”找“不同”

- 如果一定要用一句话来概括推荐机制——以“相同”找“不同”
- 何为“相同”？

即不同主体已经产生的偏好事件交集，比如张三和李四都看过《大江大河1》、《大江大河2》，且评价都很高

- 何为“不同”？

即不同主体之间的潜在差异，比如张三还看过《长安十二时辰》且评价很高，而李四未看过，那么可以给李四推荐《长安十二时辰》，李四也评价很高的概率可能会比较大

- 当然，上述定性分析过程需要更加严谨的定量分析，通过各种运算得到推荐结果
- 因此，上述问题可以分两步走：

step 1:搜集不同主体之间的偏好数据

step 2:找到评价不同主体之间偏好“相同”程度的评价方法

1 如何投其所好？——偏好数据示例

- 在互联网中可以找到很多现场的偏好数据集，为了更简单的说明问题，此处采用教材pp8中的偏好数据字典示例
- 该偏好数据集是一个嵌套型字典——critics

-外键，影评者

-内键，被评分影片

-值，分数

```
critics={'Lisa Rose': {'Lady in the Water':2.5, 'Snakes on a Plane' :3.5,
'Just My Luck':3.0, 'Superman Returns':3.5, 'You,Me and Dupree':2.5,
'The Night Listener':3.0},
'Gene Seymour':{'Lady in the Water':3.0,'Snakes on a Plane':3.5,
'Just My Luck':1.5,'Superman Returns':5.0,'The Night Listener':3.0,
'You,Me and Dupree':3.5},
'Michael Phillips':{'Lady in the Water':2.5, 'Snakes on a Plane':3.0,
'Superman Returns':3.5,'The Night Listener':4.0},
'Claudia Puig':{'Snakes on a Plane':3.5,'Just My Luck':3.0,
'The Night Listener':4.5,'Superman Returns':4.0,'You,Me and Dupree':2.5},
'Mich Lasalle':{'Lady in the Water':3.0,'Snakes on a Plane':4.0,
'Just My Luck':2.0,'Superman Returns':3.0,'The Night Listener':3.0,
'You,Me and Dupree':2.0},
'Jack Mattews':{'Lady in the Water':3.0,'Snakes on a Plane':4.0,
'The Night Listener':3.0,'Superman Returns':5.0,'You,Me and Dupree':3.5},
'Toby':{'Snakes on a Plane':4.5,'You,Me and Dupree':1.0,'Superman Returns':4.0}}
```

1 如何投其所好？——偏好数据示例

影片偏好数据集概况

- 影评人信息：有7位，他们累计评价了35部次电影（有重复）其中评论最多的一位共评价了6部影片
- 影片信息：有6部，其中被评价最多的一部累计被7个人评价

影评者姓名	评论影片数
Lisa Rose	6
Gene Seymour	6
Michael Phillips	4
Claudia Puig	5
Mich Lasalle	6
Jack Mattews	5
Toby	3

影片名	被评论次数
Snakes on a Plane	7
Superman Returns	7
You,Me and Dupree	6
The Night Listener	6
Lady in the Water	5
Just My Luck	4

1 如何投其所好？——偏好相似度评价方法之一，欧式距离法

- “人以群分，物以类聚”，三观相同的人自然走的更近，可以“用物理上的距离”概念转换为“不同主题的偏好相似程度”
- 因此，可以将每个个体抽象成坐标系中的点，用几何学中的欧几里得距离来衡量不同点之间的距离，并将点之间的距离大小转换为相似程度
- 数学表达

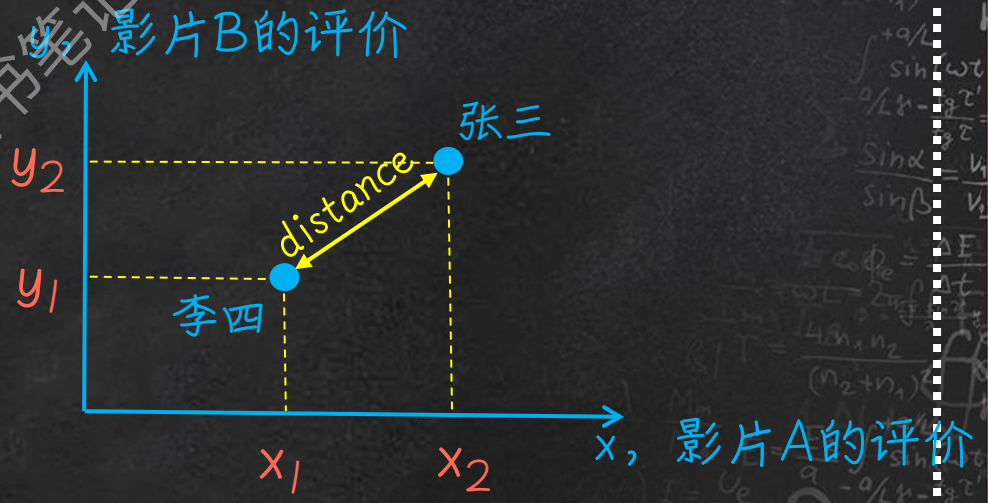
• 欧氏距离计算，以二维坐标系为例

$$d=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

• 距离转换为相似度，“加1取倒”

$$sim=\frac{1}{1+d}$$

- 距离的几何表示张三和李四的偏好相似度



- 距离与相似度的关系

• 距离越大、相似度越小

• 距离越小、相似度越大

• 基于欧氏距离的相似度取值范围（0，1]

1 如何投其所好？——偏好相似度评价方法之二，p系数法

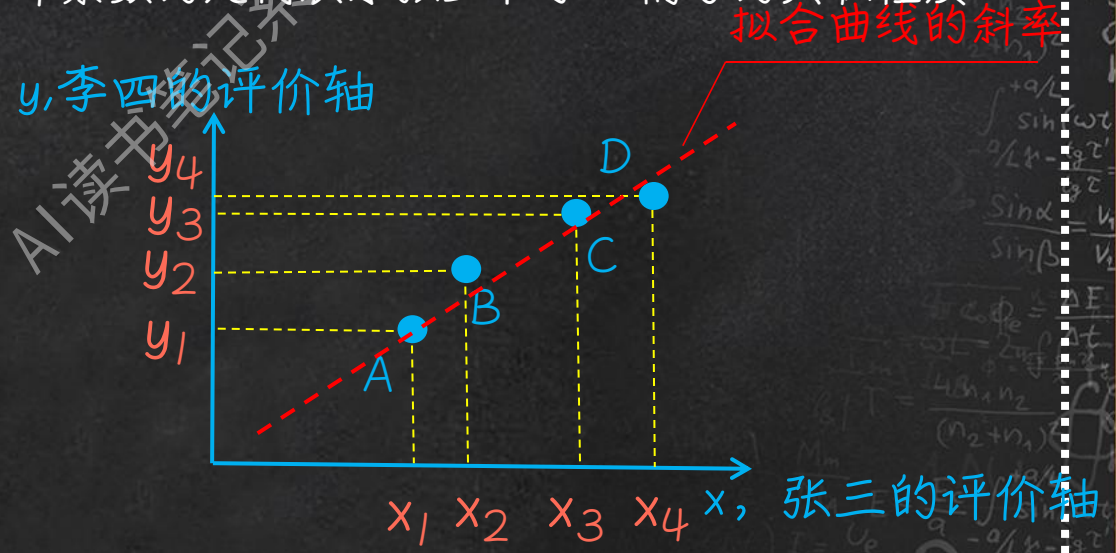
- pearson系数（皮尔逊系数，也简称p系数），用于衡量两组线性数据一同变化的趋势，即一组数据的变化与另一组数据的变化是否具有关联
- A、B两组数据之间四种可能的关联关系
  - A逐渐变大时、B也逐渐变大
  - A逐渐变小时、B也逐渐变小
  - A逐渐变大（小）时、B也逐渐变小（大）
  - A与B之间的变化没有关系
- 3种数学表达

$$\rho_{X,Y}=\frac{\operatorname{cov}(X,Y)}{\sigma_X\sigma_Y}=\frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X\sigma_Y}=\frac{E(XY)-E(X)E(Y)}{\sqrt{E(X^2)-E^2(X)}\sqrt{E(Y^2)-E^2(Y)}}$$

$$\rho_{X,Y}=\frac{N\sum XY-\sum X\sum Y}{\sqrt{N\sum X^2-(\sum X)^2}\sqrt{N\sum Y^2-(\sum Y)^2}}$$

$$\rho_{X,Y}=\frac{\sum XY-\frac{\sum X\sum Y}{N}}{\sqrt{(\sum X^2-\frac{(\sum X)^2}{N})(\sum Y^2-\frac{(\sum Y)^2}{N})}}$$

- p系数的几何表示张三和李四偏好的关联程度



- 距离与相似度的关系
  - 基于p系数的相似度取值范围[-1, 1]
  - p=1时，两组变量完全正相关
  - p=-1时，两组变量完全负相关
  - p=0时，两组变量没有关联关系

1 如何投其所好？——偏好相似度评价方法之二，p系数法

$$\rho_{X,Y}=\frac{\operatorname{cov}(X,Y)}{\sigma_X\sigma_Y}=\frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X\sigma_Y}=\frac{E(XY)-E(X)E(Y)}{\sqrt{E(X^2)-E^2(X)}\sqrt{E(Y^2)-E^2(Y)}}$$

$$\rho_{X,Y}=\frac{N\sum XY-\sum X\sum Y}{\sqrt{N\sum X^2-(\sum X)^2}\sqrt{N\sum Y^2-(\sum Y)^2}}$$

$$\rho_{X,Y}=\frac{\sum XY-\frac{\sum X\sum Y}{N}}{\sqrt{(\sum X^2-\frac{(\sum X)^2}{N})(\sum Y^2-\frac{(\sum Y)^2}{N})}}$$

两积的和 - 两和的积的均

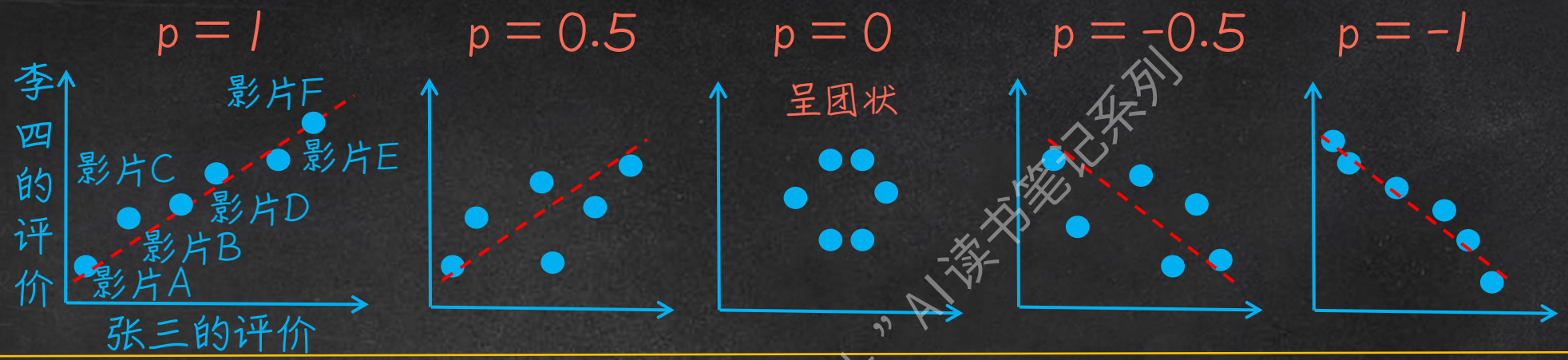
（单方和 - 单和方的均）*（单方和 - 单和方的均）

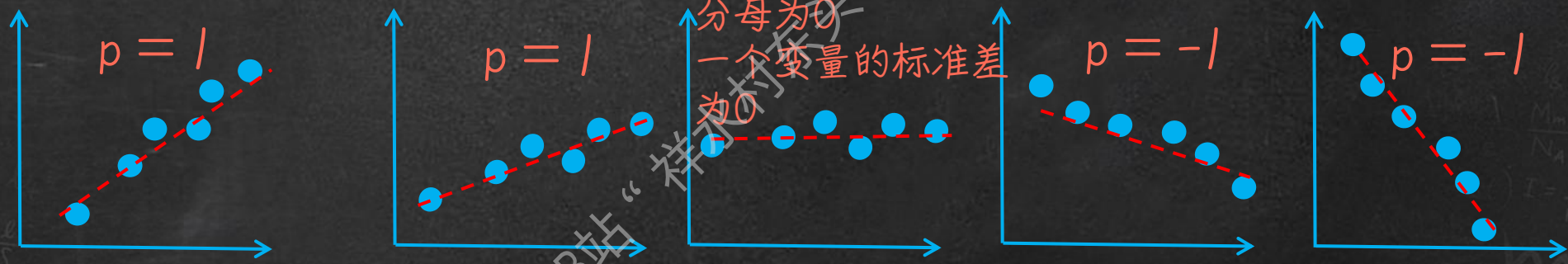
从统计学看，p系数由两部分构成
-分子：两变量协方差
-分母：两变量标准差

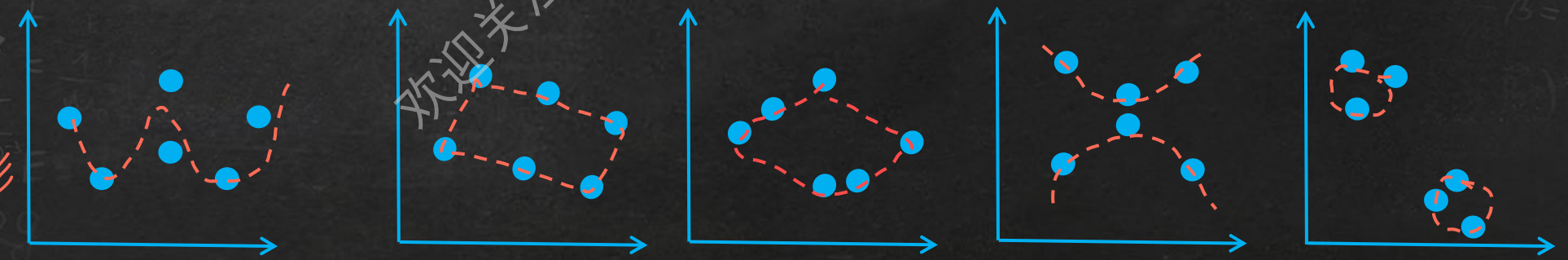
教材p13代码所采用公式
-求各组变量的和(sum1,sum2)、平方的和(sum1Sq,sum2Sq)
-求两组变量乘积的和(pSum)
-求分子，pSum -(sum1 * sum2 /n)
-求分母，sqrt((sum1Sq-pow(sum1,2)/n)*(sum1Sq-pow(sum1,2)/n))

p值	相关性描述
0.8-1.0	极强相关
0.6-0.8	强相关
0.4-0.6	中等程度相关
0.2-0.4	弱相关
0.0-0.2	极弱相关或不相关

1 如何投其所好？——偏好相似度评价方法之二，p系数法







02

人以群分

2 何谓人以群分？

- “人以群分”

以人和人之间的相似度作为参照体系

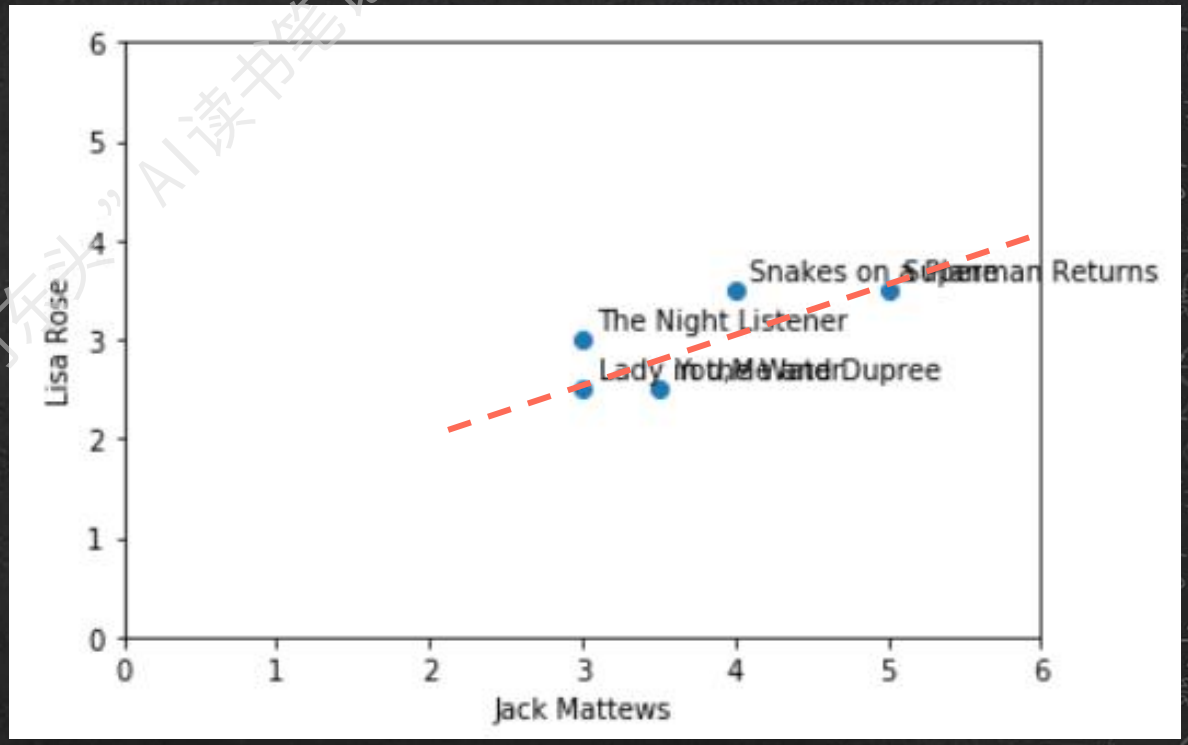
- 坐标轴：不同人的评价体系
- 坐标系中的点：不同被评价对象
- 以右侧图为例，可观察Jack和Lisa的偏好相似程度

-x轴：用户Jack的评价标准

-y轴：用户Lisa的评价标准

-点：2位用户对6部影片的评价

- 以p系数为例，2位用户评分p系数约为0.75，即Jack喜欢的影片，Lisa很可能也会喜欢



03

物以类聚

3 何谓物以类聚？

- “物以类聚”

以物和物之间的相似度作为参照体系

- 坐标轴：不同物品的被评价体系
- 坐标系中的点：不同用户
- 以右侧图为例

可观察《Snake》和《You》的口碑相似程度

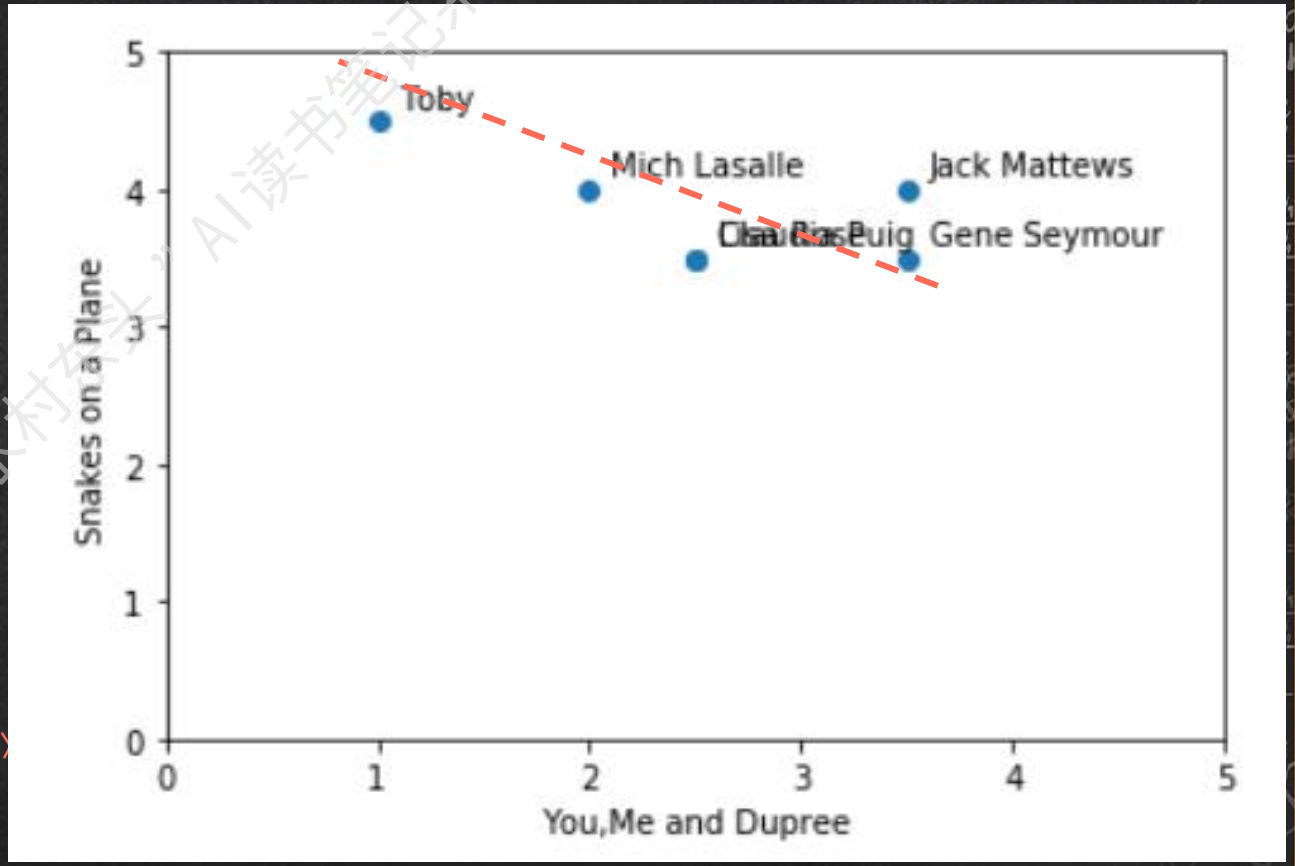
-x轴：《Snake》的被评价分数

-y轴：《You》的被评价分数

-点：每一位用户对2部影片的评价

- 以p系数为例，2部影片的口碑相似度为-0.65

即用户对2部影片的评价呈负相关，喜欢影片《Snake》的人很可能不会喜欢影片《You》，反之亦然



04

构建物品推荐

- 如何用已知偏好，推荐未知物品？

4.1 为人推荐

场景：根据张三的历史看片偏好，为其推荐一部未看过的影片，如何实现？

- step 1 定要素，确定参与偏好相似评价的各个要素
  - ✓ 确定评价主体、被评价对象、评价度量（影片评分）
  - ✓ 若采用“人以群分”模式，则评价主体为“人”，被评价对象为“影片”
  - ✓ 若采用“物以类聚”模式，则评价主体为“影片”，被评价对象为“人”
- step 2 相似数：确定相似度计算方法，并求出两两主体之间的相似度
  - ✓ 确定相似度评价方法，欧式距离法 or p系数法 or .....?
  - ✓ 若采用“人以群分”模式，则需要计算7位影评人两两之间的相似度
  - ✓ 若采用“物以类聚”模式，则需要计算6部影片两两之间的相似度
- step 3 加权度：基于相似度和评价度量为每个主体未评价过的对象计算推荐指数
  - ✓ 若采用“人以群分”模式，根据张三与其他影评人的相似度，为张三尚未看过的每部影片求出加权评分，并最终得到每一部未看影片的推荐指数
  - ✓ 若采用“物以类聚”模式，根据张三看过的所有影片与其未看过影片的相似度，并求出未看过影片的加权评分，并最终得到每一部未看影片的推荐指数

回顾：影片偏好数据集概况

- 在互联网中可以找到很多现场的偏好数据集，为了更简单的说明问题，此处采用教材pp8中的偏好数据字典示例
- 该偏好数据集是一个嵌套型字典——critics

-外键，影评者

-内键，被评分影片

-值，分数

```
critics={'Lisa Rose':{'Lady in the Water':2.5, 'Snakes on a Plane' :3.5,
'Just My Luck':3.0, 'Superman Returns':3.5, 'You,Me and Dupree':2.5,
'The Night Listener':3.0},
'Gene Seymour':{'Lady in the Water':3.0,'Snakes on a Plane':3.5,
'Just My Luck':1.5,'Superman Returns':5.0,'The Night Listener':3.0,
'You,Me and Dupree':3.5},
'Michael Phillips':{'Lady in the Water':2.5, 'Snakes on a Plane':3.0,
'Superman Returns':3.5,'The Night Listener':4.0},
'Claudia Puig':{'Snakes on a Plane':3.5,'Just My Luck':3.0,
'The Night Listener':4.5,'Superman Returns':4.0,'You,Me and Dupree':2.5},
'Mich Lasalle':{'Lady in the Water':3.0,'Snakes on a Plane':4.0,
'Just My Luck':2.0,'Superman Returns':3.0,'The Night Listener':3.0,
'You,Me and Dupree':2.0},
'Jack Mattews':{'Lady in the Water':3.0,'Snakes on a Plane':4.0,
'The Night Listener':3.0,'Superman Returns':5.0,'You,Me and Dupree':3.5},
'Toby':{'Snakes on a Plane':4.5,'You,Me and Dupree':1.0,'Superman Returns':4.0}}
```

回顾：影片偏好数据集概况

- 影评人信息：有7位，他们累计评价了35部次电影（有重复）其中评论最多的一位共评价了6部影片
- 影片信息：有6部，其中被评价最多的一部累计被7个人评价

影评者姓名	评论影片数
Lisa Rose	6
Gene Seymour	6
Michael Phillips	4
Claudia Puig	5
Mich Lasalle	6
Jack Mattews	5
Toby	3

影片名	被评论次数
Snakes on a Plane	7
Superman Returns	7
You,Me and Dupree	6
The Night Listener	6
Lady in the Water	5
Just My Luck	4

回顾：影片偏好数据集概况

影评者 / 影片	Lisa Rose	Gene Seymour	Michael Phillips	Claudia Puig	Mich Lasalle	Jack Mattews	Toby
Snakes on a Plane	3.5	3.5	3	3.5	4	4	4.5
Superman Returns	3.5	5	3.5	4	3	5	4
You,Me and Dupree	2.5	3.5	N/A	2.5	2	3.5	1
The Night Listener	3	3	4	4.5	3	3	N/A
Lady in the Water	2.5	3	2.5	N/A	3	3	N/A
Just My Luck	3	1.5	N/A	3	2	N/A	N/A

4.2 在“人以群分”模式下进行推荐（p15-17）
——基于用户的协作型过滤

- step 1，定要素：以7位影评人为评价主体，通过他们的影片评分计算用户之间的相似度
- step 2，相似数：假设采用p系数法，计算7位影片人、两两之间的相似偏好相似度，基于每两个人之间共同看过的影片评分，可得到21组相似度值

影评人A　影评人B　p系数

	person_1	person_2	similarity
11	Michael Phillips	Claudia Puig	1.000000
13	Michael Phillips	Jack Mattews	0.134840
12	Michael Phillips	Mich Lasalle	-0.258199
14	Michael Phillips	Toby	-1.000000
19	Mich Lasalle	Toby	0.924473
18	Mich Lasalle	Jack Mattews	0.211289
5	Lisa Rose	Toby	0.991241
4	Lisa Rose	Jack Mattews	0.747018
3	Lisa Rose	Mich Lasalle	0.594089
2	Lisa Rose	Claudia Puig	0.566947
1	Lisa Rose	Michael Phillips	0.404520

影评人A　影评人B　p系数

0	Lisa Rose	Gene Seymour	0.396059
20	Jack Mattews	Toby	0.662849
9	Gene Seymour	Jack Mattews	0.963796
8	Gene Seymour	Mich Lasalle	0.411765
10	Gene Seymour	Toby	0.381246
7	Gene Seymour	Claudia Puig	0.314970
6	Gene Seymour	Michael Phillips	0.204598
17	Claudia Puig	Toby	0.893405
15	Claudia Puig	Mich Lasalle	0.566947
16	Claudia Puig	Jack Mattews	0.028571

➢ step 3 加权度：基于相似度和评价度量为每个主体未评价过的对象计算推荐指数，以计算Toby未看过的3部影片推荐指数为例(p15)，关键步骤：一剔负 1、二加权 2、三求和 3、四相除 4

p系数<0时，意味着两者呈负相关，因此不需要推荐
欧氏距离法相似度为(0,1]，无负数，故不存在此问题

对比影评人　影评人间p系数　Toby未看过影片的加权评分

	person_1	person_2	similarity	The Night Listener	weight_The Night Listener	Lady in the Water	weight_Lady in the Water	Just My Luck	weight_Just My Luck
19	Mich Lasalle	Toby	0.924473	3.0	2.773420	3.0	2.773420	2.0	1.848947
5	Lisa Rose	Toby	0.991241	3.0	2.973722	2.5	2.478102	3.0	2.973722
20	Jack Mattews	Toby	0.662849	3.0	1.988547	3.0	1.988547	NaN	NaN
10	Gene Seymour	Toby	0.381246	3.0	1.143739	3.0	1.143739	1.5	0.571870
17	Claudia Puig	Toby	0.893405	4.5	4.020323	NaN	NaN	3.0	2.680215

相乘 2 等于　相乘 等于　相乘 等于

1 剔除负的p系数

参考影评人　其他人对Toby未看影片的评分，　Toby和Claudia都未看过的，可记为空值

求和 3　求和

3.84　12.89　*不算Puig 2.95　8.38　*不算Jack 3.18　8.07

4 加权评分和 ÷ 相似度和

影片《Night》推荐指数：3.45　影片《Lady》推荐指数：2.77　影片《Luck》推荐指数：2.42

综上，在“人以群分”算法模式下，对于Toby未看过的3部影片，我们优先向其推荐影片《Night》

函数代码清单

函数名称	功能	参数	返回
sim_distance (p11)	采用欧氏距离法对两个待比较对象计算相似度	参数1：偏好数据集（外键-人;内键-影片） 参数2：对象A（人评分数据） 参数3：对象B（人评分数据）	相似度值
sim_pearson (p13)	采用p系数法对两个待比较对象计算相似度		
topMatches (p14)	逐一计算指定对象与其他对象的相似度，并进行排序	参数1：偏好数据集 参数2：指定人，如Toby 参数3：取排序后的Top n相似对象 参数4：相似度计算方法，默认为sim_pearson	Top n 相似度值列表，[0.9,0.7...]
getRecommendations (p16)	为指定影评人未看过的影片进行推荐	参数1：偏好数据集 参数2：指定人，如Toby 参数3：相似度计算方法，默认为sim_pearson	列表，[(推荐指数1,未看影片1),(推荐指数2,未看影片2)...]

4.3 在“物以类聚”模式下进行推荐（p22-25）
——基于物品的协作型过滤

从“基于人的协作型过滤”切换到“基于物品的协作型过滤”，需要首先对原始偏好数据字典进行内外键调换

基于人协作的数据字典（原始）

- 影评人A
  - 1号影片：分数
  - 2号影片：分数
- 影评人B（外键）
  - （内键）（值）
- .....

内外键转换

基于物品协作的数据字典（转换后）

- 1号影片
  - 影评人A：分数
  - 影评人B：分数
  - .....
- 2号影片
- .....

4.3 在“物以类聚”模式下进行推荐（p22-25）
——基于物品的协作型过滤

- step 1，定要素：以6部电影为评价主体，通过7位影评人的评分计算各影片之间的相似度
- step 2，相似数：假设采用p系数法，计算6部影片、两两之间的相似偏好相似度，基于每两部影片之间共同的影评者评分，可得到15组相似度值

影片A　　影片B　　p系数

	item_1	item_2	similarity
14	You,Me and Dupree	The Night Listener	0.294298
13	Superman Returns	The Night Listener	0.252650
12	Superman Returns	You,Me and Dupree	0.191825
8	Snakes on a Plane	The Night Listener	0.320377
6	Snakes on a Plane	Superman Returns	0.309017
5	Snakes on a Plane	Just My Luck	0.255397
7	Snakes on a Plane	You,Me and Dupree	0.188638
3	Lady in the Water	You,Me and Dupree	0.449490

影片A　　影片B　　p系数

4	Lady in the Water	The Night Listener	0.387426
0	Lady in the Water	Snakes on a Plane	0.348331
1	Lady in the Water	Just My Luck	0.348331
2	Lady in the Water	Superman Returns	0.240253
10	Just My Luck	You,Me and Dupree	0.320377
11	Just My Luck	The Night Listener	0.298935
9	Just My Luck	Superman Returns	0.207992

➢ step 3 加权度：基于相似度和评价度量为每个主体未评价过的对象计算推荐指数，以计算Toby未看过的3部影片推荐指数为例(p23)，关键步骤：一剔负1、二加权2、三求和3、四相除4

p系数<0时，意味着两者呈负相关，因此不需要推荐
欧氏距离法相似度为(0,1]，无负数，故不存在此问题

参考影评人　Toby评分　Toby未看过影片的加权评分

	person	item	mark	The Night Listener	weight_The Night Listener	Lady in the Water	weight_Lady in the Water	Just My Luck	weight_Just My Luck
32	Toby	Snakes on a Plane	4.5	0.320377	1.441698	0.348331	1.567492	0.255397	1.149286
33	Toby	You,Me and Dupree	1.0	0.294298	0.294298	0.449490	0.449490	0.320377	0.320377
34	Toby	Superman Returns	4.0	0.252650	1.010601	0.240253	0.961012	0.207992	0.831966

2 相乘　等于　相乘　等于　相乘　等于

1 剔除负的p系数

Toby已看过影片　未看影片与Toby已看影片的相似度（特别注意：教材中在计算影片相似度时又改用欧式距离法）

3 求和 0.86　　求和 2.74　　1.02　　2.96　　0.77　　2.29

4 加权评分和 ÷ 相似度和

影片《Night》推荐指数：3.17　　《Lady》推荐指数：2.86　　《Luck》推荐指数：2.93

综上，在“物以类聚”算法下，对于Toby未看过的3部影片中，我们依然优先向其推荐影片《Night》

函数代码清单

函数名称	功能	参数	返回
transformsPrefs (p18，新增)	调换原始偏好数据字典的内外键	参数1：偏好数据集（外键-人;内键-影片）	偏好数据集（内键-人;外键-影片）
sim_distance (p11，复用)	采用欧氏距离法对两个待比较对象计算相似度	参数1：偏好数据集（外键-影片;内键-人） 参数2：对象A（影片评分数据） 参数3：对象B（影片评分数据）	相似度值
sim_pearson (p13，复用)	采用p系数法对两个待比较对象计算相似度		
topMatches (p14，复用)	逐一计算指定对象与其他对象的相似度，并进行排序	参数1：偏好数据集 参数2：指定影片，如《Lady》 参数3：取排序后的Top n相似度值，默认为5 参数4：相似度计算方法，默认为sim_distance	Top n 相似度值列表，[0.9,0.7...]
calculateSimilarItems (p24，新增)	计算每一部影片与其他影片的相似度值	参数1：偏好数据集 参数2：取排序后的Top n相似度值，默认为10	字典，{影片1：[(相似度，影片2),....}
getRecommendationsItems (p25，新增)	为指定影评人未看过的影片进行推荐	参数1：偏好数据集 参数2：指定人，如Toby 参数3：相似度计算方法，默认为sim_pearson	列表，[(推荐指数1,未看影片1)，(推荐指数2,未看影片2)...]

4.4 再思考之一，书中待商榷点

- 问题1：皮尔逊函数中的错误

p13，当两个影评人之间没有共同看过的影片时，两者相似度应为0，而不是1

- 问题2：影片相似度计算结果可能错误

p24，教材基于欧氏距离法计算各影片之间的相似度结果可能错误；如教材中《Lady》与《Listener》相似度结果为0.28，我计算的结果为0.38

- 问题3：教材在人/物两种推荐模式中所采用的相似度计算方法不一致，不利于对比分析

p16，在基于人的协作型过滤中（即“人以群分”），教材采用的相似度评估算法为p系数法

e.x.为Toby的影片推荐指数结果为：《Listner》>《Lady》>《Luck》

p25，在基于物品的协作型过滤中（即“物以类聚”），教材采用的相似度评估算法为欧式距离法

e.x.为Toby的影片推荐指数结果为：《Listner》>《Luck》>《Lady》

- 问题4：calculateSimilarItems重复冗余计算

p23，函数calculateSimilarItems在计算每个影片与其他影片的相似度时，计算方法重复

以教材中算法为例，共需要为6部影片两两之间运算30次相似度结果；这之中有一半的运算属于重复冗余运算，比如《Listner》与《Lady》运算了1次相似度，然后《Lady》与《Listner》又运算了一次；实际上仅需要运算15次相似度即可。

4.4 再思考之二
基于“欧氏距离法” or “p系数法”评价相似度，谁更好？

- p17,对比两种相似度评价的推荐结果
- 还是以Toby为例，Top 3推荐影片及推荐指数由右侧图表所示
- 上述两种相似度计算方法推荐结果一致，仅仅是推荐指数稍有差异

为Toby推荐	欧式距离法所得到的推荐结果	p系数法所得到的推荐结果
Top 1	《Listener》 3.46	《Listener》 3.35
Top 2	《Lady》 2.78	《Lady》 2.83
Top 3	《Luck》 2.42	《Luck》 2.53

4.4 再思考之三
基于“人” or 基于“物”的相似度推荐，谁更好？

- 对比两种协作过滤模式的推荐结果
- 还是以Toby为例，Top 3推荐影片及推荐指数由右侧图表所示
- 上述两种模式的第一个推荐结果无差异，但第二/三个推荐结果有差异，但是第二/三名的推荐指数非常接近
- p27对两种协作过滤模式进行了讨论，作者建议优先考虑使用“物以类聚”模式，原因：
  - ✓ 物品之间的相似度比较变化没有人之间的变化频繁，这一点在大型电商网站中尤为明显，换句话说人之间的相似度不确定性更大

为Toby推荐	欧式距离法 + “人以群分”模式	欧式距离法 + “物以类聚”模式
Top 1	《Listener》 3.46	《Listener》 3.17
Top 2	《Lady》 2.78	《Luck》 2.93
Top 3	《Luck》 2.42	《Lady》 2.87

4.4 再思考之四
基于“字典法” or “数据帧法”实现代码，谁更好？

- 教材中基本采用“字典”作为数据分析与运算的容器，即“键-值”对形式，因此在代码中有大量对字典操作的内容，优点是数据读取灵活、运算速度快，缺点是数据之间的关系呈现能力不足
- 在本人的github还给出了另一种实现方法，即采用“数据帧”（dataframe）作为数据分析与运算的容器，即“行列”形式，优点是数据关系清晰、可视化能力强，缺点是当数据集较大时，运算速度较慢、数据读取不太灵活
- 以上仅仅是个人看法，每个人可以根据自己熟悉的工具来选择解决问题的方法

05

案例实践

- 真实影评数据

真实影评数据集概况

- 根据p25的指引下载数据，本项目中主要用到两个数据集文件
- 该数据集有明尼苏达大学的GroupLens项目组开发
- 该数据集涉及943位用户评分、1682部影片、以及10万条评分记录
- 文件 u.item：包含1682部影片信息，每一条信息由4部分构成：

-影片ID、片名、上映时间、查询链接

- 文件 u.data：包含10万条评分记录，每一条记录由4部分构成：

-用户ID、影片ID、评分、评分日期

```
1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)|0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)|0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms%20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty%20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)|0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com/Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0
|0|0|0|0|1|0|0|0|0|0|0|0|0|0
```

196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596
298	474	4	884182806
115	265	2	881171488
253	465	5	891628467
305	451	3	886324817
6	86	3	883603013
62	257	2	879372434

- 基于上述数据集，为指定用户进行推荐。
- 因该案例中所使用的函数全部复用之前已编写好的代码，故在此不再赘述，只要搞懂那些函数的功能和具体实现方式，再操作这个真实案例则毫无压力，仅仅是数据集变大了而已。

“人生之旅历途甚长，所争决不在一年半日，
万不可因此着急失望，招精神上之萎靡”

——梁启超致梁思成家书
献给正在逐梦路上努力奔跑的你我他

感谢聆听

THANK YOU

本次胶片内容、及涉及相关代码均可移步至Github进行下载
感谢您的投币三连！

我的代码 Github 地址：
https://github.com/david-cal/
Reading-Note-For-Programming-Collective-Intelligence