

## 31 | 关联规则挖掘（下）：导演如何选择演员？

2019-02-22 陈旻



讲述：陈旻

时长 07:42 大小 7.06M



上次我给你讲了关联规则挖掘的原理。关联规则挖掘在生活中有很多使用场景，不仅是商品的捆绑销售，甚至在挑选演员决策上，你也能通过关联规则挖掘看出来某个导演选择演员的倾向。

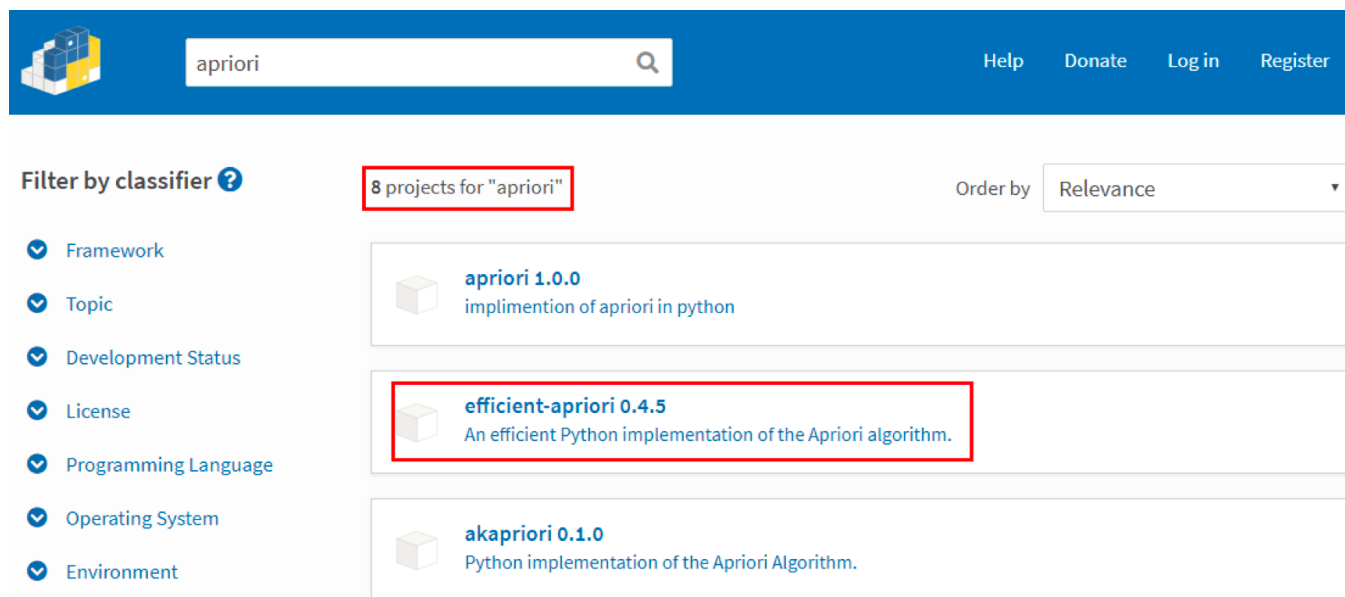
今天我来带你用 Apriori 算法做一个项目实战。你需要掌握的是以下几点：

1. 熟悉上节课讲到的几个重要概念：支持度、置信度和提升度；
2. 熟悉与掌握 Apriori 工具包的使用；
3. 在实际问题中，灵活运用。包括数据集的准备等。

### 如何使用 Apriori 工具包

Apriori 虽然是十大算法之一，不过在 sklearn 工具包中并没有它，也没有 FP-Growth 算法。这里教你个方法，来选择 Python 中可以使用的工具包，你可以通过

<https://pypi.org/> 搜索工具包。

The screenshot shows the PyPI search results for the keyword 'apriori'. The search bar at the top contains 'apriori'. Below the search bar, there are filter options on the left and a list of results on the right. The results list shows three packages: 'apriori 1.0.0', 'efficient-apriori 0.4.5', and 'akapriori 0.1.0'. The 'efficient-apriori 0.4.5' package is highlighted with a red box, indicating it is the recommended one. The description for 'efficient-apriori 0.4.5' is 'An efficient Python implementation of the Apriori algorithm.'.

Filter by classifier ?

8 projects for "apriori"

Order by Relevance

- Framework
- Topic
- Development Status
- License
- Programming Language
- Operating System
- Environment

**apriori 1.0.0**  
implimention of apriori in python

**efficient-apriori 0.4.5**  
An efficient Python implementation of the Apriori algorithm.

**akapriori 0.1.0**  
Python implementation of the Apriori Algorithm.

这个网站提供的工具包都是 Python 语言的，你能找到 8 个 Python 语言的 Apriori 工具包，具体选择哪个呢？建议你使用第二个工具包，即 efficient-apriori。后面我会讲到为什么推荐这个工具包。

首先你需要通过 `pip install efficient-apriori` 安装这个工具包。

然后看下如何使用它，核心的代码就是这一行：

复制代码

```
1 itemsets, rules = apriori(data, min_support, min_confidence)
```

其中 data 是我们要提供的数据集，它是一个 list 数组类型。min\_support 参数为最小支持度，在 efficient-apriori 工具包中用 0 到 1 的数值代表百分比，比如 0.5 代表最小支持度为 50%。min\_confidence 是最小置信度，数值也代表百分比，比如 1 代表 100%。

关于支持度、置信度和提升度，我们再来简单回忆下。

支持度指的是某个商品组合出现的次数与总次数之间的比例。支持度越高，代表这个组合出现的概率越大。


置信度是一个条件概念，就是在 A 发生的情况下，B 发生的概率是多少。

提升度代表的是“商品 A 的出现，对商品 B 的出现概率提升了多少”。

接下来我们用这个工具包，跑一下上节课中讲到的超市购物的例子。下面是客户购买的商品列表：


订单编号	购买的商品
1	牛奶、面包、尿布
2	可乐、面包、尿布、啤酒
3	牛奶、尿布、啤酒、鸡蛋
4	面包、牛奶、尿布、啤酒
5	面包、牛奶、尿布、可乐

具体实现的代码如下：

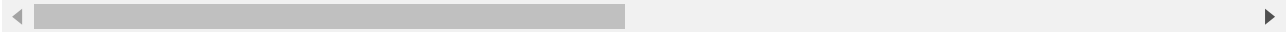
 复制代码

```
1 from efficient_apriori import apriori
2 # 设置数据集
3 data = [('牛奶','面包','尿布'),
4         ('可乐','面包','尿布','啤酒'),
5         ('牛奶','尿布','啤酒','鸡蛋'),
6         ('面包','牛奶','尿布','啤酒'),
7         ('面包','牛奶','尿布','可乐')]
8 # 挖掘频繁项集和频繁规则
9 itemsets, rules = apriori(data, min_support=0.5, min_confidence=1)
10 print(itemsets)
11 print(rules)
12
```

运行结果：

 复制代码

```
1 {1: {'啤酒',): 3, ('尿布',): 5, ('牛奶',): 4, ('面包',): 4}, 2: {'啤酒', '尿布'): 3, (
2 [{啤酒} -> {尿布}, {牛奶} -> {尿布}, {面包} -> {尿布}, {牛奶, 面包} -> {尿布}]
```



你能从代码中看出来，data 是个 List 数组类型，其中每个值都可以是一个集合。实际上你也可以把 data 数组中的每个值设置为 List 数组类型，比如：

```
1 data = [['牛奶', '面包', '尿布'],  
2         ['可乐', '面包', '尿布', '啤酒'],  
3         ['牛奶', '尿布', '啤酒', '鸡蛋'],  
4         ['面包', '牛奶', '尿布', '啤酒'],  
5         ['面包', '牛奶', '尿布', '可乐']]
```

两者的运行结果是一样的，efficient-apriori 工具包把每一条数据集里的项式都放到了一个集合中进行运算，并没有考虑它们之间的先后顺序。因为实际情况下，同一个购物篮中的物品也不需要考虑购买的先后顺序。

而其他的 Apriori 算法可能会因为考虑了先后顺序，出现计算频繁项集结果不对的情况。所以这里采用的是 efficient-apriori 这个工具包。

## 挖掘导演是如何选择演员的

在实际工作中，数据集是需要自己来准备的，比如今天我们要挖掘导演是如何选择演员的数据情况，但是并没有公开的数据集可以直接使用。因此我们需要使用之前讲到的 Python 爬虫进行数据采集。

不同导演选择演员的规则是不同的，因此我们需要先指定导演。数据源我们选用豆瓣电影。

先来梳理下采集的工作流程。

首先我们先在<https://movie.douban.com>搜索框中输入导演姓名，比如“宁浩”。

## 搜索 宁浩



宁浩 Hao Ning

9385 人收藏

演员 / 导演 / 编剧 / 制片人 / 剪辑 / 1977-09-09 / 我不是药神 / 一出好戏 / 老炮儿

> 搜索更多叫“宁浩”的影人

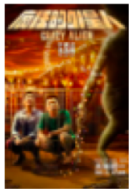


### 流浪地球 (2019)

★★★★★ (尚未上映)

中国大陆 / 科幻 / The Wandering Earth / 125分钟

郭帆 / 屈楚萧 / 李光洁 / 吴孟达 / 赵今麦 / 隋凯 / 屈菁菁 / 张亦驰 / 杨皓宇



### 疯狂的外星人 (2019)

★★★★★ (尚未上映)


中国大陆 / 喜剧 / 科幻 / Crazy Alien / 116分钟

宁浩 / 黄渤 / 沈腾 / 马修·莫里森 / 汤姆·派福瑞 / 凯特·纳尔逊 / 徐峥 / 于和伟 / 雷佳音

页面会呈现出来导演之前的所有电影，然后对页面进行观察，你能观察到以下几个现象：

1. 页面默认是 15 条数据反馈，第一页会返回 16 条。因为第一条数据实际上这个导演的概览，你可以理解为是一条广告的插入，下面才是真正的返回结果。
2. 每条数据的最后一行是电影的演出人员的信息，第一个人员是导演，其余为演员姓名。姓名之间用 “/” 分割。

有了这些观察之后，我们就可以编写抓取程序了。在代码讲解中你能看出这两点观察的作用。抓取程序的目的是为了生成宁浩导演（你也可以抓取其他导演）的数据集，结果会保存在 csv 文件中。完整的抓取代码如下：

 复制代码

```
1 # -*- coding: utf-8 -*-
2 from efficient_apriori import apriori
3 from lxml import etree
4 import time
5 from selenium import webdriver
6 import csv
7 driver = webdriver.Chrome()
8 # 设置想要下载的导演 数据集
9 director = u'宁浩'
10 # 写 CSV 文件
```

```

11 file_name = './' + director + '.csv'
12 base_url = 'https://movie.douban.com/subject_search?search_text='+director+'&cat=1002&st
13 out = open(file_name,'w', newline='', encoding='utf-8-sig')
14 csv_write = csv.writer(out, dialect='excel')
15 # 下载指定页面的数据
16 def download(request_url):
17     driver.get(request_url)
18     time.sleep(1)
19     html = driver.find_element_by_xpath("//*").get_attribute("outerHTML")
20     html = etree.HTML(html)
21     # 设置电影名称, 导演演员 的 XPATH
22     movie_lists = html.xpath("/html/body/div[@id='wrapper']/div[@id='root']/div[1]//div
23     name_lists = html.xpath("/html/body/div[@id='wrapper']/div[@id='root']/div[1]//div[
24     # 获取返回的数据个数
25     num = len(movie_lists)
26     if num > 15: # 第一页会有 16 条数据
27         # 默认第一个不是, 所以需要去掉
28         movie_lists = movie_lists[1:]
29         name_lists = name_lists[1:]
30     for (movie, name_list) in zip(movie_lists, name_lists):
31         # 会存在数据为空的情况
32         if name_list.text is None:
33             continue
34         # 显示下演员名称
35         print(name_list.text)
36         names = name_list.text.split('/')
37         # 判断导演是否为指定的 director
38         if names[0].strip() == director:
39             # 将第一个字段设置为电影名称
40             names[0] = movie.text
41             csv_write.writerow(names)
42     print('OK') # 代表这页数据下载成功
43     if num >= 15:
44         # 继续下一页
45         return True
46     else:
47         # 没有下一页
48         return False
49
50 # 开始的 ID 为 0, 每页增加 15
51 start = 0
52 while start<10000: # 最多抽取 1 万部电影
53     request_url = base_url + str(start)
54     # 下载数据, 并返回是否有下一页
55     flag = download(request_url)
56     if flag:
57         start = start + 15
58     else:
59         break
60 out.close()
61 print('finished')

```



代码中涉及到了几个模块，我简单讲解下这几个模块。

在引用包这一段，我们使用 csv 工具包读写 CSV 文件，用 efficient\_apriori 完成 Apriori 算法，用 lxml 进行 XPath 解析，time 工具包可以让我们在模拟后有个适当停留，代码中我设置为 1 秒钟，等 HTML 数据完全返回后再进行 HTML 内容的获取。使用 selenium 的 webdriver 来模拟浏览器的行为。

在读写文件这一块，我们需要事先告诉 python 的 open 函数，文件的编码是 utf-8-sig（对应代码：encoding= 'utf-8-sig' ），这是因为我们会用到中文，为了避免编码混乱。

编写 download 函数，参数传入我们要采集的页面地址（request\_url）。针对返回的 HTML，我们需要用到之前讲到的 Chrome 浏览器的 XPath Helper 工具，来获取电影名称以及演出人员的 XPath。我用页面返回的数据个数来判断当前所处的页面序号。如果数据个数 >15，也就是第一页，第一页的第一条数据是广告，我们需要忽略。如果数据个数 =15，代表是中间页，需要点击“下一页”，也就是翻页。如果数据个数 <15，代表最后一页，没有下一页。

在程序主体部分，我们设置 start 代表抓取的 ID，从 0 开始最多抓取 1 万部电影的数据（一个导演不会超过 1 万部电影），每次翻页 start 自动增加 15，直到 flag=False 为止，也就是不存在下一页的情况。

你可以模拟下抓取的流程，获得指定导演的数据，比如我上面抓取的宁浩的数据。这里需要注意的是，豆瓣的电影数据可能是不全的，但基本上够我们用。

疯狂的外星人（2019）	黄渤	沈腾	马修·莫	汤姆·派	凯特·纳	徐峥	于和伟	雷佳音
疯狂的石头（2006）	郭涛	刘桦	连晋	黄渤	徐峥	优惠	罗兰	王迅
无人区（2013）	徐峥	黄渤	余男	多布杰	王双宝	巴多	杨新鸣	郭虹
心花路放（2014）	黄渤	徐峥	袁泉	周冬雨	陶慧	岳小军	沈腾	张俪
疯狂的赛车（2009）	黄渤	戎祥	九孔	徐峥	王双宝	巴多	董立范	高捷
黄金大劫案（2012）	雷佳音	陶虹	程媛媛	山崎敬一	郭涛	范伟	孙淳	刘桦
香火（2003）	李强							
奇迹世界（2007）	张曦	黄渤	王玮	聂鑫	王迅	岳小军		
绿草地（2005）	巴德玛							
星期四星期三（2001）								

有了数据之后，我们就可以用 Apriori 算法来挖掘频繁项集和关联规则，代码如下：

```

1 # -*- coding: utf-8 -*-
2 from efficient_apriori import apriori
3 import csv
4 director = u'宁浩'
5 file_name = './'+director+'.csv'
6 lists = csv.reader(open(file_name, 'r', encoding='utf-8-sig'))
7 # 数据加载
8 data = []
9 for names in lists:
10     name_new = []
11     for name in names:
12         # 去掉演员数据中的空格
13         name_new.append(name.strip())
14     data.append(name_new[1:])
15 # 挖掘频繁项集和关联规则
16 itemsets, rules = apriori(data, min_support=0.5, min_confidence=1)
17 print(itemsets)
18 print(rules)

```

代码中使用的 apriori 方法和开头中用 Apriori 获取购物篮规律的方法类似，比如代码中都设定了最小支持度和最小置信系数，这样我们可以找到支持度大于 50%，置信系数为 1 的频繁项集和关联规则。

这是最后的运行结果：

```

1 {1: {('徐峥',): 5, ('黄渤',): 6}, 2: {('徐峥', '黄渤'): 5}}
2 [{徐峥} -> {黄渤}]

```

你能看出来，宁浩导演喜欢用徐峥和黄渤，并且有徐峥的情况下，一般都会用黄渤。你也可以用上面的代码来挖掘下其他导演选择演员的规律。

## 总结

Apriori 算法的核心就是理解频繁项集和关联规则。在算法运算的过程中，还要重点掌握对支持度、置信度和提升度的理解。在工具使用上，你可以使用 efficient-apriori 这个工具包，它会把每一条数据中的项（item）放到一个集合（篮子）里来处理，不考虑项（item）之间的先后顺序。



在实际运用中你还需要灵活处理，比如导演如何选择演员这个案例，虽然工具的使用会很方便，但重要的还是数据挖掘前的准备过程，也就是获取某个导演的电影数据集。



最后给你留两道思考题吧。请你编写代码挖掘下张艺谋导演使用演员的频繁项集和关联规则，最小支持度可以设置为 0.1 或 0.05。另外你认为 Apriori 算法中的最小支持度和最小置信度，一般设置为多少比较合理？

欢迎你在评论区与我分享你的答案，也欢迎点击“请朋友读”，把这篇文章分享给你的朋友或者同事。



极客时间

# 数据分析实战 45 讲

即学即用的数据分析入门课

陈旻

清华大学计算机博士



新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

## 精选留言 (8)

[写留言](#)**third**

2019-02-22

👍 2

个人的直觉感觉，这个应该跟数据集的大小和特点有关。

编辑回复: 对 和数据集特点有关系，不过数据集大的情况下，不好观察特征。我们可以通过设置最小值支持度和最小置信度来观察关联规则的结果。

一般来说最小支持度常见的取值有0.5, 0.1, 0.05。最小置信度常见的取值有1.0, 0.9, 0.8。可以通过尝试一些取值，然后观察关联结果的方式来调整最小值尺度和最小置信度的取值。

**third**

2019-02-22

👍 1

感觉：1，张艺谋喜欢用那些组合的人  
2.某些组合出现的匹配率

最小支持度为0.1，{1: {' 巩俐 '): 9, (' 李雪健 '): 5}}

...

[展开](#)**third**

2019-02-22

👍 1

对于Xpath的query的删除，来找到需要查找的内容，表示艰难。

自己总结的是，

1.保留div[1]

2.删除名字比较长的class...

[展开](#)

编辑回复: 我的技巧就是不断的尝试，另外XPath是有自己规则的，99%的情况下都是以//开头，因为想要匹配所有的元素，然后找一些关键的特征来进行匹配，比如class='item-root'的节点，

或者id='root'都是很好的特征。通过观察id或class，可以自己编写XPath，这样写的XPath会更短。总之，都是要不断的尝试，才能找到自己想要找的内容。寻找XPath的过程就是一个找规律的过程。



白夜

2019-02-22

👍 1

最小支持度可以设置的小，而如果最小支持度小，置信度就要设置的相对大一点，不然即使提升度高，也有可能是巧合。这个参数跟数据量以及项的数量有关。

理解对吗？

...

展开 ∨

编辑回复: 一般来说最小置信度都会大一些，比如1.0, 0.9或者0.8。最小支持度和数据集大小和特点有关，可以尝试一些数值来观察结果，比如0.1, 0.5



mickey

2019-03-02

👍

Efficient-Apriori

An efficient pure Python implementation of the Apriori algorithm. Works with Python 3.6 and 3.7.

efficient-apriori在2.7下不能用，只能用于3.6和3.7。老师应该告知一下环境有所变化。

展开 ∨



mickey

2019-03-01

👍

安装工具包报错，请问怎样解决？

```
E:\DevelopTool\Python\Python27\Scripts>pip install efficient-apriori
```

DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020.

Please upgrade your Python as Python 2.7 won't be maintained after that date. A future versio...

展开 ∨



**JingZ**

2019-02-28



# 关联规则挖掘

出现错误 selenium.common.exceptions.WebDriverException: Message:

'chromedriver' executable needs to be in PATH. 参考

<https://blog.csdn.net/liaoningxinmin/article/details/82686185> 按正常的套路Mac下载了Chromedriver，将解压好的文件放入/usr/local/bin目录中，由于mac的很多目录...

展开 ∨



**王彬成**

2019-02-24



1 ) 最小支持度设置为0.1

{1: {'倪大红',): 3, ('孙红雷',): 3, ('巩俐',): 9, ('李保田',): 3, ('李雪健',): 4, ('章子怡',): 3, ('葛优',): 3}}

[]

2 ) 最小支持度设置为0.05...

展开 ∨