

zhuanlan.zhihu.com

## 天秀！Pandas还能用来写爬虫？

7-8 minutes

谈及Pandas的read.xxx系列的函数，大家的第一反应会想到比较常用的pd.read\_csv()和pd.read\_excel()，大多数人估计没用过pd.read\_html()这个函数。

虽然它低调，但功能非常强大，用于抓取**Table表格型数据**时，简直是个神器。下面来详细介绍一下。

大家逛网页时，经常会看到这样一些数据表格，比如：

## 电影票房数据

名次	电影名称	票房	总场次	人次	票价(元)
1	战狼2	58.58亿	175.4万	1.09亿	36.6
2	战狼2	18.51亿	105.15万	3860.79万	46.7
3	宝贝对不起！再见！的骑士	18.06亿	104.17万	4889.33万	37.1
4	神偷奶爸3	16.72亿	97.37万	4785.91万	35.1
5	加勒比海盜5：死无对证	14.92亿	86.27万	4121.37万	36.2
6	哪吒之魔童降世	13.83亿	144.33万	3642.08万	38
7	红海行动	13.76亿	80.01万	3441.66万	40
8	摔跤吧！爸爸	13.22亿	107.24万	4255.46万	31.1
9	悟空传	13.18亿	72.72万	3621.39万	36.4
10	黄飞鸿之四：狮王之战	12.66亿	105.26万	2478.42万	51.1
11	速度与激情7	12.62亿	49.17万	2646.71万	47.7
12	战狼1	11.88亿	80.6万	2570.33万	46.2
13	唐人街探案2	11.7亿	62.72万	2974.91万	39.3
14	我不是神	11.19亿	92.58万	3126.08万	35.8
15	速度与激情8	11.17亿	92.38万	2861.19万	39
16	寻龙诀	11.04亿	45.42万	2333.84万	47.3
17	疯狂动物城	10.73亿	63.34万	2412.84万	44.5
18	羞羞的谎言	10.27亿	80.2万	2985.66万	34.4
19	唐探	8.42亿	44.82万	2127.89万	40.6

## 世界大学排行榜数据

The screenshot displays the Academic Ranking of World Universities 2019 website. On the left, a table lists the top 15 universities. On the right, a code editor shows a CSS snippet for a table with 5 columns: World Rank, Institution, By location, National/Regional Rank, Total Score, and Score vs. Alumni.

World Rank	Institution	By location	National/Regional Rank	Total Score	Score vs. Alumni
1	Harvard University	USA	1	100.0	100.0
2	Stanford University	USA	2	75.1	45.2
3	University of Cambridge	UK	1	72.3	80.7
4	Massachusetts Institute of Technology (MIT)	USA	3	69.0	72.0
5	University of California, Berkeley	USA	4	67.8	67.1
6	Princeton University	USA	5	60.0	59.6
7	University of Oxford	UK	2	59.7	48.9
8	Columbia University	USA	6	59.1	61.4
9	California Institute of Technology	USA	7	58.6	52.3
10	University of Chicago	USA	8	55.1	59.6
11	University of California, Los Angeles	USA	9-10	50.8	28.6
12	Yale University	USA	9-10	50.8	47.6
13	Cornell University	USA	11	49.8	43.3
14	University of Washington	USA	12	48.7	24.4
15	University College London	UK	3	4	Back to Top
16	Johns Hopkins University	USA	13	47.6	36.8
17	University of Pennsylvania	USA	14	47.3	31.2
18	University of California, San Diego	USA	15	47.1	19.4

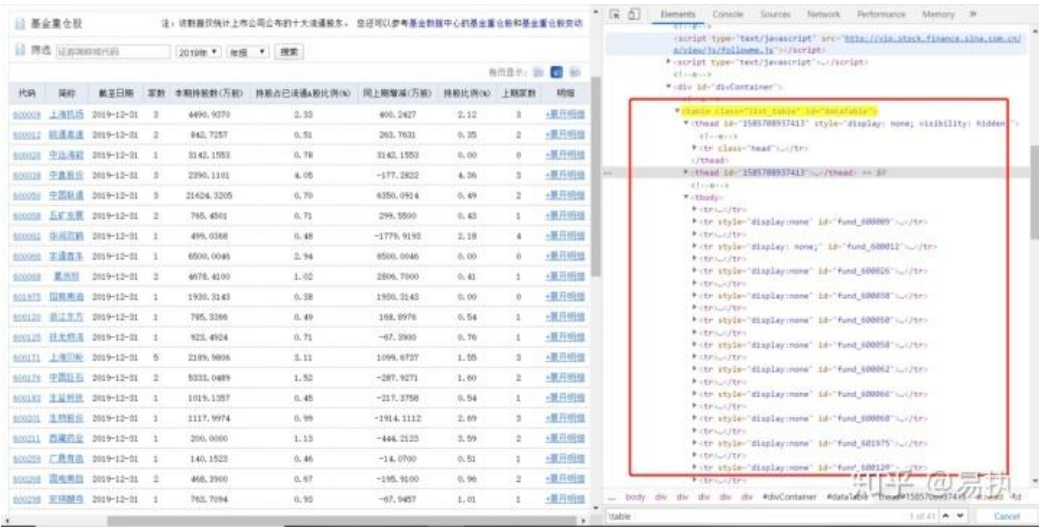
The code editor on the right shows a CSS snippet for a table with 5 columns: World Rank, Institution, By location, National/Regional Rank, Total Score, and Score vs. Alumni. The snippet includes a table structure with 5 columns and 15 rows, and a list of universities with their respective scores and rankings.

```

<table>
  <tr>
    <th>World Rank</th>
    <th>Institution</th>
    <th>By location</th>
    <th>National/Regional Rank</th>
    <th>Total Score</th>
    <th>Score vs. Alumni</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Harvard University</td>
    <td>USA</td>
    <td>1</td>
    <td>100.0</td>
    <td>100.0</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Stanford University</td>
    <td>USA</td>
    <td>2</td>
    <td>75.1</td>
    <td>45.2</td>
  </tr>
  <tr>
    <td>3</td>
    <td>University of Cambridge</td>
    <td>UK</td>
    <td>1</td>
    <td>72.3</td>
    <td>80.7</td>
  </tr>
  <tr>
    <td>4</td>
    <td>Massachusetts Institute of Technology (MIT)</td>
    <td>USA</td>
    <td>3</td>
    <td>69.0</td>
    <td>72.0</td>
  </tr>
  <tr>
    <td>5</td>
    <td>University of California, Berkeley</td>
    <td>USA</td>
    <td>4</td>
    <td>67.8</td>
    <td>67.1</td>
  </tr>
  <tr>
    <td>6</td>
    <td>Princeton University</td>
    <td>USA</td>
    <td>5</td>
    <td>60.0</td>
    <td>59.6</td>
  </tr>
  <tr>
    <td>7</td>
    <td>University of Oxford</td>
    <td>UK</td>
    <td>2</td>
    <td>59.7</td>
    <td>48.9</td>
  </tr>
  <tr>
    <td>8</td>
    <td>Columbia University</td>
    <td>USA</td>
    <td>6</td>
    <td>59.1</td>
    <td>61.4</td>
  </tr>
  <tr>
    <td>9</td>
    <td>California Institute of Technology</td>
    <td>USA</td>
    <td>7</td>
    <td>58.6</td>
    <td>52.3</td>
  </tr>
  <tr>
    <td>10</td>
    <td>University of Chicago</td>
    <td>USA</td>
    <td>8</td>
    <td>55.1</td>
    <td>59.6</td>
  </tr>
  <tr>
    <td>11</td>
    <td>University of California, Los Angeles</td>
    <td>USA</td>
    <td>9-10</td>
    <td>50.8</td>
    <td>28.6</td>
  </tr>
  <tr>
    <td>12</td>
    <td>Yale University</td>
    <td>USA</td>
    <td>9-10</td>
    <td>50.8</td>
    <td>47.6</td>
  </tr>
  <tr>
    <td>13</td>
    <td>Cornell University</td>
    <td>USA</td>
    <td>11</td>
    <td>49.8</td>
    <td>43.3</td>
  </tr>
  <tr>
    <td>14</td>
    <td>University of Washington</td>
    <td>USA</td>
    <td>12</td>
    <td>48.7</td>
    <td>24.4</td>
  </tr>
  <tr>
    <td>15</td>
    <td>University College London</td>
    <td>UK</td>
    <td>3</td>
    <td>4</td>
    <td>Back to Top</td>
  </tr>
  <tr>
    <td>16</td>
    <td>Johns Hopkins University</td>
    <td>USA</td>
    <td>13</td>
    <td>47.6</td>
    <td>36.8</td>
  </tr>
  <tr>
    <td>17</td>
    <td>University of Pennsylvania</td>
    <td>USA</td>
    <td>14</td>
    <td>47.3</td>
    <td>31.2</td>
  </tr>
  <tr>
    <td>18</td>
    <td>University of California, San Diego</td>
    <td>USA</td>
    <td>15</td>
    <td>47.1</td>
    <td>19.4</td>
  </tr>
</table>

```

财经数据



如果查看一下网页的HTML结构（Chrome浏览器F12），会发现它们有个共同的特点，不仅是表格，还是以**Table结构**展示的表格数据，大致的网页结构如下

```
<table class="..." id="...">
  <thead>
    <tr>
      <th>...</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>...</td>
    </tr>
    <tr>...</tr>
    <tr>...</tr>
    ...
    <tr>...</tr>
    <tr>...</tr>
  </tbody>
</table>
```

针对网页结构类似的表格类型数据，pd.read\_html()就派上了大用场了，它可以将网页上的表格都抓取下来，并以DataFrame的形式装在一个列表中返回。具体是这么个流程：



先介绍一下read\_html的一些主要的参数

## read\_html

- io : str or file-like  
接收网址、文件、字符串。网址不接受https, 尝试去掉s后爬去
- header: int or list-like or None  
指定列标题所在的行
- attrs : dict or None, optional  
传递一个字典, 用其中的属性筛选出特定的表格
- parse\_dates: bool  
解析日期

接下来以爬取新浪财经的基金重仓股为例演示一下, URL为:

```
http://vip.stock.finance.sina.com.cn/q/go.php/vComStockHold/kind/jjzc/index.phtml?
p=1
```

这部分有6页, 点击不同的页数可以发现, 请求URL主要是p参数在变动, p=n代表了第n页, 所以一个for循环就可以遍历所有网址啦。URL的变动规律了解之后, 就可以愉快的爬数据了, 上代码

```
import pandas as pd

df = pd.DataFrame()

for i in range(6):
    url =
    'http://vip.stock.finance.sina.com.cn/q/go.php/vComStockHold/kind/jjzc/index.phtml?
    p={page}'.format(page=i+1)
    df = pd.concat([df,pd.read_html(url)[0]])
    print("第{page}页完成~".format(page=i+1))

df.to_csv('./data.csv', encoding='utf-8', index=0)
```

	代码	简称	截至日期	家数	本期持股数(万股)	持股占已流通A股比例(%)	同上期增减(万股)
1	600009	上海机场	2019-12-31	3	4490.937	2.33	400.2427
2	600012	铁龙高速	2019-12-31	2	842.7257	0.51	263.7631
3	600026	中远海能	2019-12-31	1	3142.1553	0.78	3142.1553
4	600038	中直股份	2019-12-31	3	2390.1101	4.05	-177.2822
5	600050	中国联通	2019-12-31	3	21624.3205	0.7	6350.0914
6	600058	五矿发展	2019-12-31	2	765.4501	0.71	299.55
7	600062	华润双鹤	2019-12-31	1	499.0368	0.48	-1779.9193
8	600066	宇通客车	2019-12-31	1	6500.0046	2.94	6500.0046
9	600068	龙洲坝	2019-12-31	2	4678.41	1.02	2806.7
10	601975	招商南油	2019-12-31	1	1930.3143	0.38	1930.3143
11	600120	浙江东方	2019-12-31	1	785.3366	0.49	168.8976
12	600125	铁龙物流	2019-12-31	1	923.4924	0.71	-67.39
13	600171	上海贝岭	2019-12-31	5	2189.9806	3.11	1099.6737
14	600176	中国巨石	2019-12-31	2	5333.0489	1.52	-287.9271
15	600183	生益科技	2019-12-31	1	1019.1357	0.45	-217.3758
16	600201	生物股份	2019-12-31	1	1117.9974	0.99	-1914.1112
17	600211	西藏药业	2019-12-31	1	200.0	1.13	-444.2123
18	600259	广晟有色	2019-12-31	1	140.1523	0.46	-14.07
19	600268	国电南自	2019-12-31	2	468.39	0.67	-195.91
20	600298	安琪酵母	2019-12-31	1	763.7094	0.93	-67.9457
21	600309	万华化学	2019-12-31	1	3453.17	1.1	-1408.1692
22	600316	洪都航空	2019-12-31	1	437.9005		1407.64
23	600317	营口港	2019-12-31	1	2015.9443	0.31	-90.03

整个过程不需要用到正则表达式或者xpath等工具，短短的几行代码就可以将数据嗖嗖地爬下来了，是不是超级无敌方便？

日后在爬一些小型数据时，只要遇到这种**Table类型的表格**，就可以直接祭出read\_html这个神器啦，别人还在琢磨正则、xpath怎么写的时候，你已经把数据爬完了，想想就很舒服！

相关文章：

- 1. [提高数据的颜值！一起看看Pandas中的那些Style](#)
- 2. [Pandas数据处理三板斧——map、apply、applymap详解](#)
- 3. [Pandas数据分析——超好用的Groupby详解](#)
- 4. [Pandas数据分析——Merge数据拼接图文详解](#)
- 5. [Pandas数据处理——玩转时间序列数据](#)
- 6. [Pandas数据处理——盘点那些常用的函数（上）](#)
- 7. [Pandas数据处理——盘点那些常用的函数（下）](#)
- 8. [提速百倍的Pandas性能优化方法，让你的Pandas飞起来！](#)

原创不易，如果觉得有点用，希望可以[点个赞](#)，拜谢各位老铁！