

NBA 球员数据搜索

小组队名：CLAY

小组成员：队长 叶凯尧 519120990009

队员 朱烨 519120990012

项目介绍：

用户输入 NBA 现役球员名称（如：james harden）。程序从 NBA 官网获取当前赛季该球员相关数据，提取该球员的基本信息，基本数据。将基本信息和基本数据(如场均得分、场均助攻数…)以表格的形式输出。提取球员在半场的出手点坐标，提取球员在半场各区块的出手及命中数，用 python 绘制球员投篮散点图，投篮热区图。添加球场图和球员头像，最终使某个球员的出手分布和命中率展现在一张图中。从而达到使用 python 展示 NBA 球员出手喜好的目的。

所用到的第三方类库：

1. requests
2. pandas
3. numpy
4. matplotlib

项目代码：

首先我们需要 import 第三方类库：

```
import requests
import urllib.request
import pandas as pd
import numpy as np
import matplotlib
from matplotlib import pyplot as plt
from matplotlib.patches import Arc, Circle, Rectangle
from matplotlib.offsetbox import OffsetImage
%matplotlib inline

matplotlib.rc("font", family = "Microsoft YaHei", weight = "bold") #使pyplot.table 可以显示中文
```

其中，%matplotlib inline 是为了在 jupyter notebook 上运行

(1) 设置球场函数（朱烨）

在设置球场函数前，先要引入第三方类 matplotlib，通过 matplotlib，我们导入绘制篮球场主要用到的图形圆弧、圆和矩形

```
import matplotlib
from matplotlib import pyplot as plt
from matplotlib.patches import Arc, Circle, Rectangle
```

准备工作做完后，开始设置球场函数。

首先，设置第一个球场函数 draw_court()

设置球场函数的参数(颜色和线宽) `draw_court(color=' #20458C', lw=2):`

我们将在坐标轴中确定球场各线条所处的位置

所以通过 `ax = plt.gca()` 设置坐标轴。

绘制球场边框：

```
lines_outer_rec = Rectangle(xy=(-250, -47.5), width=500, height=470, linewidth=lw, color=' #F0F0F0', fill=True)
```

了解一个球场的各部分组成以及区域分布后，绘制篮筐、篮板、2 分区的外框线和内框线、罚球圈、三分线的圆弧、外半圈、内半圆、篮球场外框线等元素，因为一个球场所涉及的线条比较多，所以我要多次通过设置它们的坐标和长宽高来实现。注意：参数是坐标 (x, y)，即矩形的画图的起点坐标，这个起点坐标不是一味地从左下角开始画，而是对应整个图中坐标原点，即 (0, 0)。即如下代码

```

lines_outer_rec = Rectangle(xy=(-250, -47.5), width=500, height=470, linewidth=lw, color=color, fill=False)
outer_rec = Rectangle(xy=(-80, -47.5), width=160, height=190, linewidth=lw, color=color, fill=False)
inner_rec = Rectangle(xy=(-60, -47.5), width=120, height=190, linewidth=lw, color=color, fill=False)
circle_punish = Circle(xy=(0, 142.5), radius=60, linewidth=lw, color=color, fill=False)
three_left_rec = Rectangle(xy=(-220, -47.5), width=0, height=140, linewidth=lw, color=color, fill=False)
three_right_rec = Rectangle(xy=(220, -47.5), width=0, height=140, linewidth=lw, color=color, fill=False)
three_arc = Arc(xy=(0, 0), width=477.32, height=477.32, theta1=22.8, theta2=157.2, linewidth=lw, color=color, fill=False)
center_outer_arc = Arc(xy=(0, 422.5), width=120, height=120, theta1=180, theta2=0, linewidth=lw, color=color, fill=False)
center_inner_arc = Arc(xy=(0, 422.5), width=40, height=40, theta1=180, theta2=0, linewidth=lw, color=color, fill=False)
lines_outer_rec = Rectangle(xy=(-250, -47.5), width=500, height=470, linewidth=lw, color=color, fill=False)

```

把这些球场线条用 add_patch 方法加入到坐标轴中：

```

court_elements = [lines_outer_rec, outer_rec, inner_rec, circle_punish, three_left_rec, three_right_rec, three_arc,
                  , center_outer_arc, center_inner_arc, lines_outer_rec]

for element in court_elements:
    ax.add_patch(element)

return ax

```

至此，第一个球场的球场函数就完成了。

设置第二个球场函数，其他步骤都与第一个相同，但由于我们需要提高一个球员在球场上的不同位置的命中率，所以我们得在此基础上进一步进行划分，所以需要进一步划分。

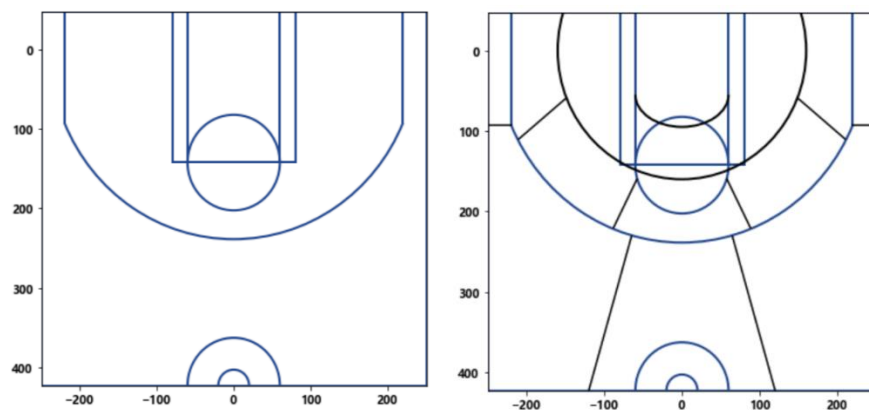
```

plt.plot([-250, -220], [92.5, 92.5], color='black')
plt.plot([250, 220], [92.5, 92.5], color='black')
plt.plot([-210, -150], [110, 60], color='black')
plt.plot([210, 150], [110, 60], color='black')
plt.plot([-88, -58], [220, 160], color='black')
plt.plot([88, 58], [220, 160], color='black')
plt.plot([-120, -65], [422.5, 231], color='black')
plt.plot([120, 65], [422.5, 231], color='black')

```

通过 plt.plot 引入坐标范围，设定线条颜色为黑色

若设置 figure (figsize= (6, 6))，调用两个球场函数，用 plt.show()输出结果如下：



在接下来的绘制散点图和热区图中，就可以直接调用这两个球场函数了。

(2) 获取&分析数据 (叶凯尧):

首先, 我用 input 获取用户需要搜索的球员名

```
a = input("Please enter player's First name(all lower_case)")
b = input("Please enter player's Last name (all lower case)")

player_name = f"{a}_{b}"
```

把输入的用户名分为 First name 和 Last name 两部分, 赋值到 player_name。

```
try:


except IndexError:
    print("该球员这赛季还没上场过呢!")

#排除球员名输入错误的情况
except Exception:
    print("查无此球员, 请输入正确的球员名! 注意输入英文小写字母")
```

在运行搜索时代码前, 先将代码放入 try 和 except 语句中, 以达到程序能打印出用户输入的球员这赛季没有上场因此无数据或用户输入的球员名不存在的情况。

将 player_id 放入 url_1 中

url_1

Request URL: https://china.nba.com/static/data/player/hotzone_james_harden.json
Request Method: GET
Status Code:  200
Remote Address: 61.129.7.47:443
Referrer Policy: no-referrer-when-downgrade

▼ Response Headers

cache-control: no-cache
content-encoding: gzip
content-type: application/json

从 url_1 的开发者工具界面可以看到, 获取数据的方法为 request 的 get 方法, 而获取数据的文本类型为 json, 因此这里我们用 requests.get(url_1) 获取我需要的数据, 赋值到 res_1, 再用 res_1.json()将获取到的数据转换为字典类型, 赋值到 result_1。如下:

```
url_1 = f"https://china.nba.com/static/data/player/hotzone_{player_name}.json"

#获取url_1的数据
res_1 = requests.get(url_1)
result_1 = res_1.json()
```

从 result_1 输出的字典中，我需要的球员的个人资料（以下称为 profile）和球员的场均数据（以下称为 average_stat），同时我们还需要球员在球场各区块的投篮出手次数，命中数及命中率（以下称为 hotzones），因此我将需要的数据从字典中提取出来。

```
#从获取的字典中选择需要的数据
profile = result_1['payload']['profile']
player_hotzones = result_1['payload']['hotZone']['seasons'][1:]

#获取到的player_hotzones是字典嵌套list嵌套字典的形式，利用循环取出需要的'hotzones'和'average_stat'
for key,value in player_hotzones[0].items():
    if key == 'splits':
        info = value[1:]
        for key,value in info[0].items():
            if key == 'full':
                hotzones = value
            if key == 'statAverage':
                average_stat = value
```

```
{ 'profile': { 'abbr': 'HOU',
               'full': { 'c08Attempted': '39',
                          'c08Held': '02.00',
                          'statAverage': { 'assistsPg': 7.4,
```

其中,由于一般赛季分为季前赛和常规赛,而这两者数据都已列表元素的形式包含在 seasons 中 (type (seasons) 为 list), 因此用切片的方式, 提取我们需要的 常规赛数据 (player_hotzones)。

在 `player_hotzones` 中，有我们需要的 `average_stat` 和 `hotzones`，但是 `player_hotzones` 其中是列表嵌套字典嵌套列表嵌套字典的形式，因此我用列表切片和循环字典的方式，将需要的数据分别赋值到 `average_stat` 和 `hotzones` 上。

在 `average_stat` 和 `profile` 中，我们只需要保留部分 `key` 和对应的 `value`，因此我们创建了一个列 (`need_to_delete`) 用来将不需要的 `key` 放到里面，再定义一个 `deleteKeys` 函数，删除在 `profile` 和 `average_stat` 中不需要的数据。如下：

```

#筛选需要的字典内容, 列出不需要的
need_to_delet = ['country', 'code', 'displayAffiliation', 'displayName', 'dob',
                 'experience', 'firstInitial', 'firstName', 'firstNameEn', 'lastName', 'lastNameEn', 'leagueId',
                 'defRebsPg', 'fgaPg', 'fgmPg', 'ftaPg', 'ftmPg', 'games', 'gamesStarted', 'minsPg', 'offRebsPg',

#在删除不需要的keys和values前, 把获取url_2数据需要的'playerId'提取出来
playerId = profile['playerId']

#定义函数deletKeys, 删除不需要keys和values, 留下需要的
def deletKeys(n):
    for i in need_to_delet:
        if i in n:
            del n[i]

deletKeys(profile)
deletKeys(average_stat)

```

至此, 已经取得了要绘制表格所需要的所有数据。在删除 profile 中 key 和 value 前, 有一项 key 名为 playerId, 我将 player_id 提取出来, 以便在接下来获取 url_2 中的数据使用。

```

#在删除不需要的keys和values前, 把获取url_2数据需要的'playerId'提取出来
playerId = profile['playerId']

```

我将 playerId 放入 url_2 中, 同样在开发者工具上, 获取数据的方式是 get 方法, 数据的文本是 json 类型, 因此重复 url_1 中的步骤。其中, 在参数中修改了 headers 的默认值, 否则会连接不上服务器。

```

#获取url_2的数据
url_2 = f"https://stats.nba.com/stats/shotchartdetail?AheadBehind=&CFID=330"

res_2 = requests.get(url_2, headers=headers, timeout=5)
result_2 = res_2.json()

#得到画散点图需要的数据
headers = result_2['resultSets'][0]['headers']
shots = result_2['resultSets'][0]['rowSet']

```

同样, 获取的字典中嵌套字典嵌套列表嵌套字典的形式, 我将绘制投篮点所需要的数据提取出来。接下来, 用 pandas 的 DataFrame 函数进行数据分析。

```

profile_df = pd.DataFrame(profile, index = ['profile'])
average_stat_df = pd.DataFrame(average_stat, index = ['stats'])
shot_df = pd.DataFrame(shots, columns=headers)

```

最后获取球员头像:

```

#获取球员头像
pic = urllib.request.urlretrieve("http://stats.nba.com/media/players/230x185/" + playerId + ".png")
head_pic = plt.imread(pic[0]) #得到一个表示图像的NumPy数组

```

由于球员头像为图片，所以用 `urllib.request.urlretrieve` 来获取图片文件。同时用 `pyplot` 的 `imread` 方法将图片转换为 `numpy` 数组，赋值到 `head_pic`。

(3) 绘制表格、投篮散点图、出手热区（叶凯尧）

首先是绘制球员头像：

```

print('绘制头像...')
#创建figure, 添加subplot, 消除坐标轴尺度
fig = plt.figure(figsize=(1,1))
fig.add_subplot(111, frameon=False, xticks=[], yticks=[])
img = OffsetImage(head_pic, zoom=0.6)
# (x, y) 控制将球员放在你想要放的位置
img.set_offset((0,0))
# 添加球员图片
fig.gca().add_artist(img)

```

创建 `figure`，添加消除坐标轴刻度的 `subplot`，用 `matplotlib` 的 `OffsetImage` 和 `set_offset` 调整参数后，将球员图片显示在左上角。

接下来是绘制表格：

```

print('绘制表格...')
def print_table(df):
    vals = df.values
    fig = plt.figure(figsize=(12,1))
    fig.add_subplot(111, frameon=False, xticks=[], yticks=[])
    the_table=plt.table(cellText=vals, rowLabels=df.index, colLabels=df.columns, colWidths = [0.1]*vals.shape[1], cellLoc='center')
    the_table.set_fontsize(20)
    the_table.scale(2.5, 2.58)

print_table(profile_df)
print_table(average_stat_df)

```

同样，用 `pyplot` 的 `figure` 和 `add_subplot` 创建“画布”。用 `table` 函数绘制表格，调整其参数，设置列的标题和对应的数据。用 `set_fontsize` 和 `scale` 方法设置字体大小和表格大小，最后将两个表格（`profile` 和 `average_stat`）打印出来。

最后使绘制投篮散点图和出手热区图：

```
#创建figure来画散点图和热区图，调整figsize
fig = plt.figure(figsize=(13.2,6))
print(' 绘制投篮点...')
fig.add_subplot(121)
draw_court()
made = shot_df[shot_df['SHOT_MADE_FLAG']==1]
miss = shot_df[shot_df['SHOT_MADE_FLAG']==0]
plt.scatter(miss.LOC_X, miss.LOC_Y, s=30, marker='x', color='red')
plt.scatter(made.LOC_X, made.LOC_Y, s=30, marker='o', edgecolors='#3A7711', color="#F0F0F0", linewidths=2)
plt.annotate(f"命中率 {average_stat['fgpct']}%", xy=(100, 160), xytext=(145, 418), fontsize='large', fontweight='bold')

# 调整 x 轴, y轴使其与篮球场匹配
plt.xlim(-250,250)
plt.ylim(422.5, -47.5)
# 消除坐标轴刻度
plt.tick_params(labelbottom=False, labelleft=False)

print(' 绘制热区...')
#把热区图画在散点图右边
fig.add_subplot(122)
draw_court2()

#标注各区块的出手次数，命中数，命中率
plt.text(-25, 20, f"{hotzones['c08Pct']}\n{hotzones['c08Made']}/{hotzones['c08Attempted']}")
plt.text(-14, 215, f"{hotzones['c1624Pct']}\n{hotzones['c1624Made']}/{hotzones['c1624Attempted']}")
plt.text(-20, 320, f"{hotzones['c24PlusPct']}\n{hotzones['c24PlusMade']}/{hotzones['c24PlusAttempted']}")
plt.text(-210, 80, f"{hotzones['l1624Pct']}\n{hotzones['l1624Made']}/{hotzones['l1624Attempted']}")
plt.text(170, 80, f"{hotzones['r1624Pct']}\n{hotzones['r1624Made']}/{hotzones['r1624Attempted']}")
plt.text(-240, 0, f"{hotzones['l24PlusPct']}\n{hotzones['l24PlusMade']}/{hotzones['l24PlusAttempted']}")
plt.text(200, 0, f"{hotzones['r24PlusPct']}\n{hotzones['r24PlusMade']}/{hotzones['r24PlusAttempted']}")
plt.text(-160, 165, f"{hotzones['lc1624Pct']}\n{hotzones['lc1624Made']}/{hotzones['lc1624Attempted']}")
plt.text(120, 165, f"{hotzones['rc1624Pct']}\n{hotzones['rc1624Made']}/{hotzones['rc1624Attempted']}")
plt.text(-200, 320, f"{hotzones['lc24PlusPct']}\n{hotzones['lc24PlusMade']}/{hotzones['lc24PlusAttempted']}")
plt.text(160, 320, f"{hotzones['rc24PlusPct']}\n{hotzones['rc24PlusMade']}/{hotzones['rc24PlusAttempted']}")
plt.text(-140, 10, f"{hotzones['l816Pct']}\n{hotzones['l816Made']}/{hotzones['l816Attempted']}")
plt.text(100, 10, f"{hotzones['r816Pct']}\n{hotzones['r816Made']}/{hotzones['r816Attempted']}")

# 调整 x 轴, y轴使其与篮球场匹配
plt.xlim(-250,250)
plt.ylim(422.5, -47.5)
# 消除坐标轴刻度
plt.tick_params(labelbottom=False, labelleft=False)
```

在一开始先创建了一个 figure，是因为要让两张图并排展示，即用 add_subplot 的方法创建两个子图。

在绘制投篮散点图时，调用第一个球场函数 draw_court。分别将 shot_df 中 shot_made_flag 为 1（即命中）和 shot_made_flag 为 0（即未命中）分别赋值到 made 和 miss 中。接着用 pyplot 中的 scatter 函数绘制散点图。其中参数 x, y 分别为 loc_x 和 loc_y，同时设置散点的大小（s=30），形状（o 为命中，x 为未命中），颜色等。最后调整 x 轴和 y 轴，使其与球场函数匹配，同时设置 tick_params 中的参数消除坐标轴刻度。其中用 annotate（pyplot 的注释方法）在右下角标注命中率。

在绘制投篮热区时，将子图放在第一行的第二列。调用第二个球场函数 draw_court2，同时分别在不同区块，用 text 函数放上该区块的出手次数，命中次数和命中率，同样最后调整 x 轴和 y 轴，以及消除坐标轴刻度。

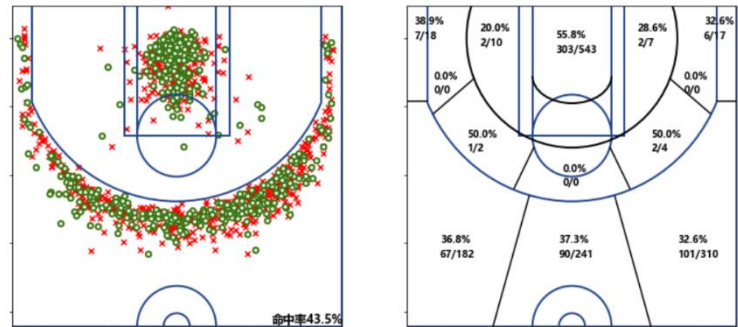
最后，plt.show()，输出如下：

Please enter player's first name(all lower case)james
 Please enter player's Last name (all lower case)harden
 搜索球员中...
 绘制头像...
 绘制表格...
 绘制投篮点...
 绘制热区...



	countryEn	displayNameEn	draftYear	height	jerseyNo	position	weight
profile	United States	James Harden	2009	1米96	13	后卫	99,8 公斤

	assistsPg	blocksPg	fgpct	foulsPg	pointsPg	rebsPg	stealsPg	turnoversPg
stats	7.4	0.9	43.5	3.4	34.4	6.4	1.7	4.5



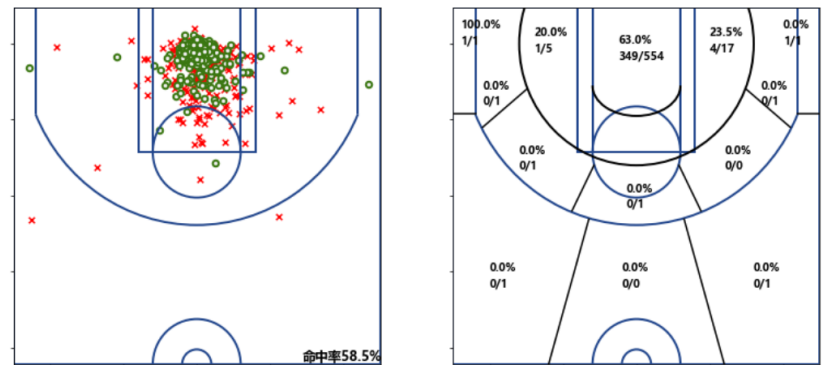
搜索不同球员，将得到相对应的输出数据：

Please enter player's First name(all lower case)ben
 Please enter player's Last name (all lower case)simmons
 搜索球员中...
 绘制头像...
 绘制表格...
 绘制投篮点...
 绘制热区...



	countryEn	displayNameEn	draftYear	height	jerseyNo	position	weight
profile	Australia	Ben Simmons	2016	2米08	25	后卫-前锋	108,9 公斤

	assistsPg	blocksPg	fgpct	foulsPg	pointsPg	rebsPg	stealsPg	turnoversPg
stats	8.2	0.6	58.5	3.2	16.7	7.8	2.1	3.6

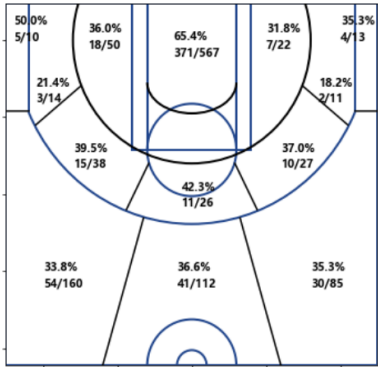
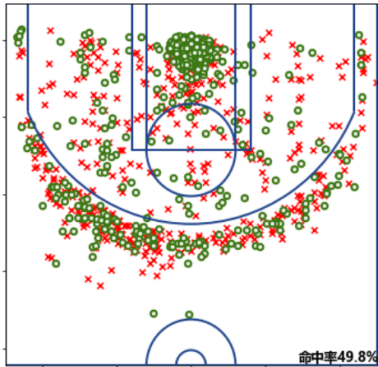


Please enter player's First name(all lower_case)lebron
Please enter player's Last name (all lower case)james
搜索球员中...
绘制头像...
绘制表格...
绘制投篮点...
绘制热区...



	countryEn	displayNameEn	draftYear	height	jerseyNo	position	weight
profile	United States	LeBron James	2003	2米06	23	前锋	113,4 公斤

	assistsPg	blocksPg	fgpct	foulsPg	pointsPg	rebsPg	stealsPg	turnoversPg
stats	10.6	0.5	49.8	1.8	25.7	7.9	1.2	4.0



通过这些数据我们对不同球员进行对比和分析。

总结

我们利用 requests 库，从网站上获取需要的数据，用 pandas 和 numpy 库 对数据进行分析，最后通过 matplotlib 讲图像呈现出来，已达通过搜索球员名，输出球员信息，数据，投篮散点图，出手热区图的目的。