

Web Scraping and Social Media Scraping Project

Name and ID

Zhoashuai Wang 436661

Weida Pan 369868

Short description of the topic and the web page

Imagine you are a League of Legends player. And you would like to know the win rate about the champion you are playing VS other champions, then you could choose your counter champion to win more games.

So this project is going to crawl the famous LOL data website:

<https://euw.op.gg/champion/statistics>

In this website, there are all champion information and ranking data. We will focus on the champion position and win rate VS other champion.

For all 3 parts, we will use champion 'sett', position 'top' as example, so the web will be like that : **<https://euw.op.gg/champion/sett/statistics/top/matchup>**

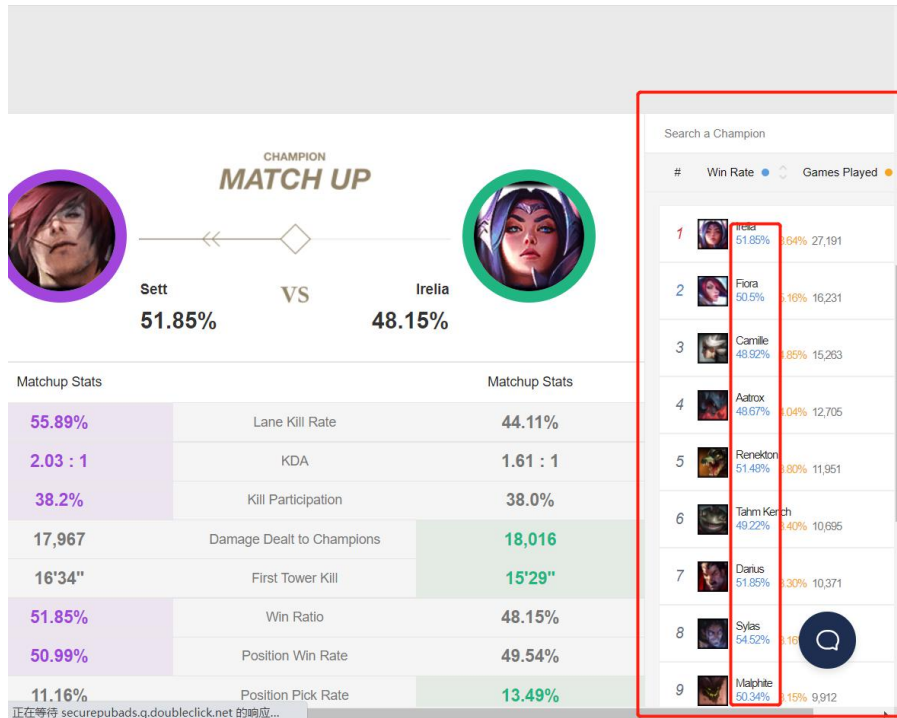
Replace {sett} and {top} to check another champion's match up data.

Our goal is : Giving one champion and the position, the output will be this champion's match-up win rate.

Description of our scraper mechanics and result

1. For beautiful soup part

There page is a half-dynamic page, we used BS to get the static part:



Check the web source code:

Search a Champion

div.champion-matchup-champion-list_item.champion-matchup-champion-list_item--act...

297.2 x 60

```

<div class="champion-matchup-search">...</div>
<div class="champion-matchup-sort">...</div>
<div class="champion-matchup-champion-list">
  <div class="champion-matchup-champion-list_item champion-matchup-champion-list_item--active" data-champion-id="39" data-champion-name="irelia" data-champion-key="irelia" data-value-winrate="0.5185" data-value-totalplayed="27191">...</div>
  <div class="champion-matchup-champion-list_item" data-champion-id="114" data-champion-name="fiora" data-champion-key="fiora" data-value-winrate="0.5050" data-value-totalplayed="16231">...</div>
  <div class="champion-matchup-champion-list_item" data-champion-id="164" data-champion-name="camille" data-champion-key="camille" data-value-winrate="0.4892" data-value-totalplayed="15263">...</div>
  <div class="champion-matchup-champion-list_item" data-champion-id="266" data-champion-name="aatrox" data-champion-key="aatrox" data-value-winrate="0.4867" data-value-totalplayed="12705">...</div>
  <div class="champion-matchup-champion-list_item" data-champion-id="58" data-champion-name="renekton" data-champion-key="renekton" data-value-winrate="0.5148" data-value-totalplayed="11951">...</div>
</div>

```

... list div.champion-matchup-champion-list_item.champion-matchup-champion-list_item--active

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility

Filter

element.style {

There are all in one div class, so first we find all this div class:

```
items = soup.find_all('div', class_='champion-matchup-champion-list__item')
```

Then get it one by one by using for loop:

```

for i in items:
    # The data-champion-name attribute value in the div is the hero name
    name = i['data-champion-name']
    # The data-value-winrate attribute in the div attribute is the winning rate of the hero
    rate = float(i['data-value-winrate'])
    print(name, '{}%'.format(round(rate * 100, 2)))

```

2. For scrapy part

Our spider name is gamelol_spider, so we used the command to run :

scrapy crawl gamelol_spider -o gamelol.csv

Just like the bs part, we use the same structure, there are 2 items in our spider. We could find the champion name and win rate by using i_item.xpath:

```
class GamelolSpiderSpider(scrapy.Spider):
    name = 'gamelol_spider'
    allowed_domains = ['euw.op.gg']
    start_urls = ['http://euw.op.gg/champion/sett/statistics/top/matchup']
    def parse(self, response):
        hero_lists = response.xpath("//div[@class='champion-matchup-champion-list']/div")
        for i_item in hero_lists:
            gamelol_item=GamelolItem()
            gamelol_item['hero_name'] = i_item.xpath('..//@data-champion-name').extract()
            gamelol_item['win_rate'] =i_item.xpath('..//@data-value-winrate').extract()
            yield gamelol_item
```

3. For selenium part

For this part, the website will open automatically to loop all champion. When it loops, we will click the champion icon button to check the small window:

CHAMPION MATCH UP

Sett 51.85% VS Irelia 48.15%

| Matchup Stats | | Matchup Stats | |
|---------------|---------------------------|---------------|--|
| 55.89% | Lane Kill Rate | 44.11% | |
| 2.03 : 1 | KDA | 1.61 : 1 | |
| 38.2% | Kill Participation | 38.0% | |
| 17,967 | Damage Dealt to Champions | 18,016 | |
| 16'34" | First Tower Kill | 15'29" | |
| 51.85% | Win Ratio | 48.15% | |
| 50.99% | Position Win Rate | 49.54% | |
| 11.16% | Position Pick Rate | 13.49% | |
| 10.11% | Ban Rate | 49.06% | |

Minute CS Per Minute XP Per Minute

Search a Champion

| # | Champion | Win Rate | Games Played |
|----|------------|----------|--------------|
| 1 | Irelia | 51.85% | 8,64% |
| 2 | Flora | 50.5% | 5.16% |
| 3 | Camille | 48.92% | 4.85% |
| 4 | Aatrox | 48.67% | 4.04% |
| 5 | Renekton | 51.48% | 3.80% |
| 6 | Tahm Kench | 49.22% | 3.40% |
| 7 | Darius | 51.85% | 3.30% |
| 8 | Sylas | 54.52% | 3.16% |
| 9 | Malphite | 50.34% | 3.15% |
| 10 | Jayce | 52.82% | 2.96% |

Check the website source code:

The screenshot shows a League of Legends Champion Matchup page for Sett vs Irelia. The page displays match statistics for both champions, a list of 10 champions, and a table of matchup stats. The source code is visible on the right, showing the HTML structure of the page.

| Matchup Stats | |
|---------------|---------------------------|
| 55.89% | Lane Kill Rate |
| 2.03 : 1 | KDA |
| 38.2% | Kill Participation |
| 17,967 | Damage Dealt to Champions |
| 16'34" | First Tower Kill |
| 51.85% | Win Ratio |
| 50.99% | Position Win Rate |
| 11.16% | Position Pick Rate |
| 10.11% | Ban Rate |

We could find all data: Lane kill Rate, KDA, Win ratio and so on. We could just take what we need.

So our selenium code idea is like that :

First find the button:

```
# find button
wait = ui.WebDriverWait(browser, wait_time)
try:
    wait.until(lambda driver: driver.find_elements_by_xpath("//div[@class='champion-matchup-list__champion']//span[1]"))
    buttons = browser.find_elements_by_xpath("//div[@class='champion-matchup-list__champion']//span[1]")
except Exception as error1:
    buttons = browser.find_elements_by_xpath("//div[@class='champion-matchup-list__champion']//span[1]")
time.sleep(10)
```

Then find the left part of the table. Loop all champion button.

```
# loop all button
print('hero    win rate')
for button in buttons:
    # click button
    browser.execute_script("arguments[0].click();", button)
    # to find the left part of the table
    wait = ui.WebDriverWait(browser, wait_time)
    try:
        wait.until(
            lambda driver: driver.find_elements_by_xpath("//table[@class='champion-matchup-table']//td[1]"))
        col = browser.find_elements_by_xpath("//table[@class='champion-matchup-table']//td[1]")
    except Exception as error1:
        browser.execute_script("arguments[0].click();", button)
        col = browser.find_elements_by_xpath("//table[@class='champion-matchup-table']//td[1]")
        time.sleep(10)
```

Finally output the 5th line data: win ratio. (we could get more data if needed)

```
print(button.text, col[5].text)
```

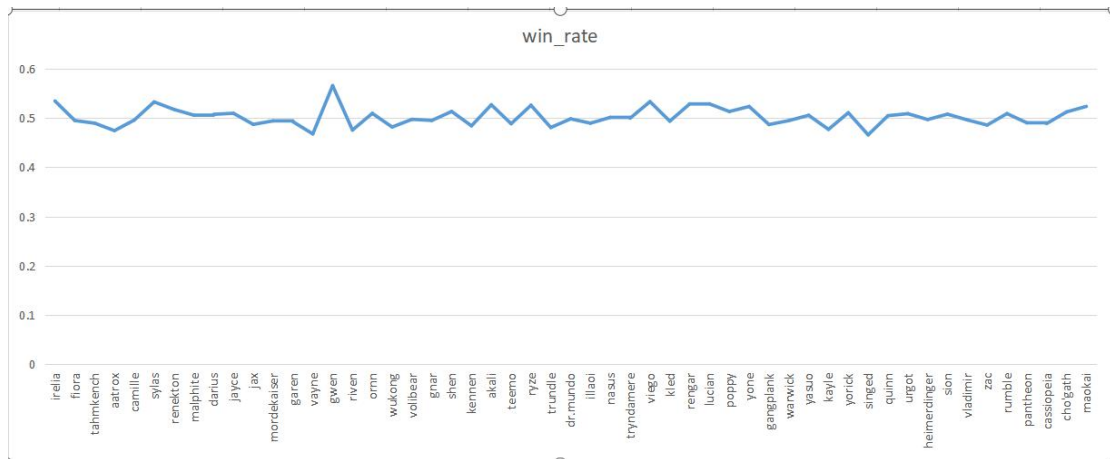
The output data we get and

For one example champion's data, we get like below. Actually, we could input more champions and get more data when we need.

```
web scraping BS x
C:\Users\wangz\PycharmProjects\pythonProject1\venv\S
input the champion name: sett
input the position: top
searching, please wait~~~
hero    win rate
irelia  51.83%
fiora   48.11%
camille 49.32%
aatrox  47.56%
tahmkench 51.62%
renekton 50.38%
darius  51.38%
sylas   55.04%
```

| | A | B | C | D | E | F | G | H | I |
|----|------------|----------|---|---|---|---|---|---|---|
| 1 | hero_name | win_rate | | | | | | | |
| 2 | irelia | 0.5183 | | | | | | | |
| 3 | fiora | 0.4811 | | | | | | | |
| 4 | camille | 0.4932 | | | | | | | |
| 5 | aatrox | 0.4756 | | | | | | | |
| 6 | tahmkench | 0.5162 | | | | | | | |
| 7 | renekton | 0.5038 | | | | | | | |
| 8 | darius | 0.5138 | | | | | | | |
| 9 | sylas | 0.5504 | | | | | | | |
| 10 | malphite | 0.4935 | | | | | | | |
| 11 | jayce | 0.5481 | | | | | | | |
| 12 | ornn | 0.4836 | | | | | | | |
| 13 | jax | 0.4706 | | | | | | | |
| 14 | mordekaise | 0.4722 | | | | | | | |
| 15 | vayne | 0.4376 | | | | | | | |
| 16 | riven | 0.5135 | | | | | | | |
| 17 | gnar | 0.4685 | | | | | | | |
| 18 | garen | 0.4909 | | | | | | | |
| 19 | kennen | 0.4919 | | | | | | | |
| 20 | trundle | 0.5069 | | | | | | | |
| 21 | shen | 0.5322 | | | | | | | |
| 22 | volibear | 0.4807 | | | | | | | |
| 23 | wukong | 0.4993 | | | | | | | |
| 24 | teemo | 0.4868 | | | | | | | |
| 25 | akali | 0.5022 | | | | | | | |
| 26 | ryze | 0.5215 | | | | | | | |
| 27 | gwen | 0.5654 | | | | | | | |
| 28 | illaoi | 0.4763 | | | | | | | |
| 29 | lucian | 0.5227 | | | | | | | |
| 30 | tryndamere | 0.4751 | | | | | | | |

We could check max() win rate and order the win rate. Or plot them to see the trend.



So we could say that collected data can be used for further analysis if needed.

Task division

Zhaoshuai Wang : Selenium part , Scrapy part , descriptions part

Weida Pan: Beautiful soup part, some of Scrapy part, descriptions part