Web Scraping and Social Media Scraping Project

Name and ID

Zhoashuai Wang 436661
Weida Pan 369868

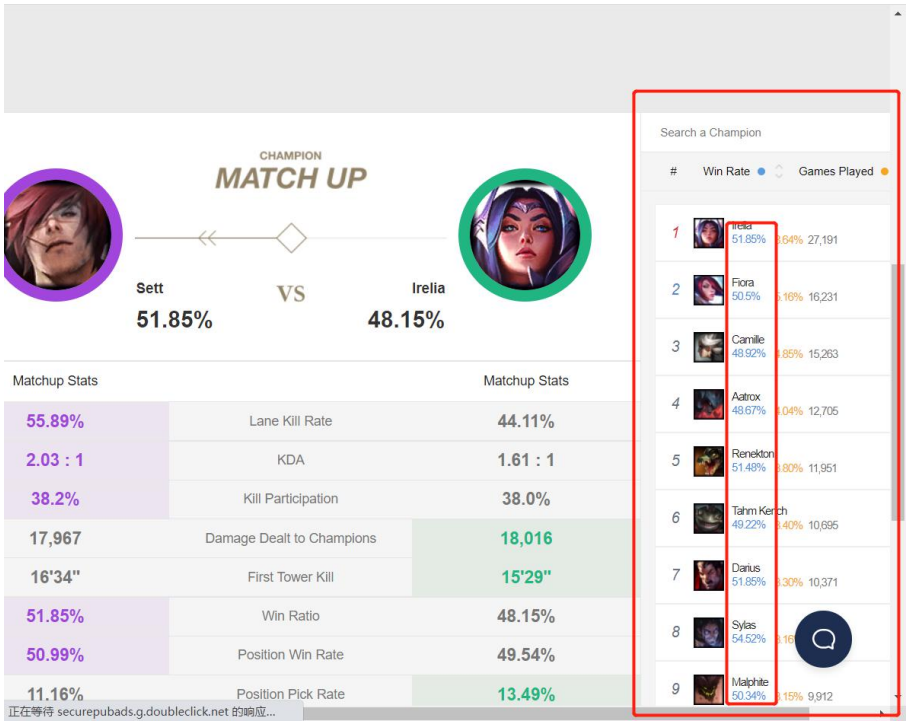
Short description of the topic and the web page

Image 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 craw 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.

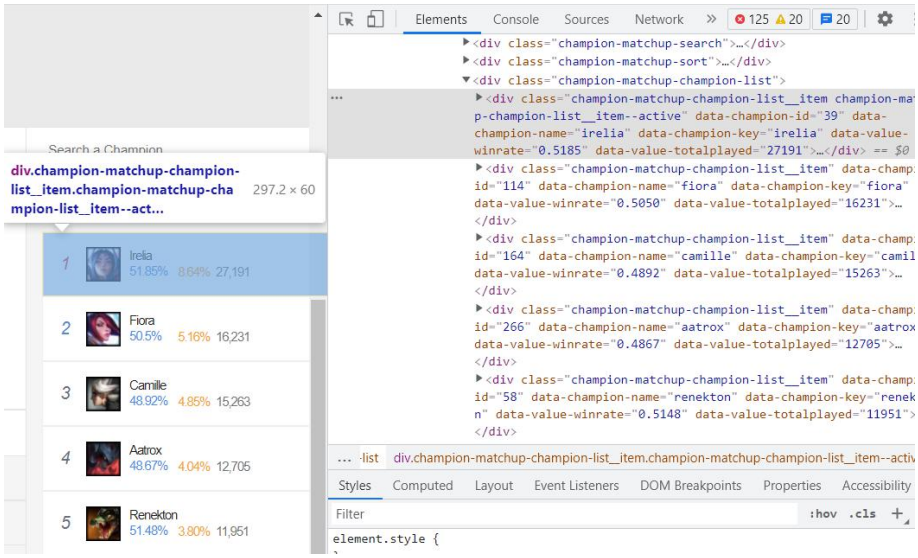
Description of our scraper mechanics and result

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:



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 :

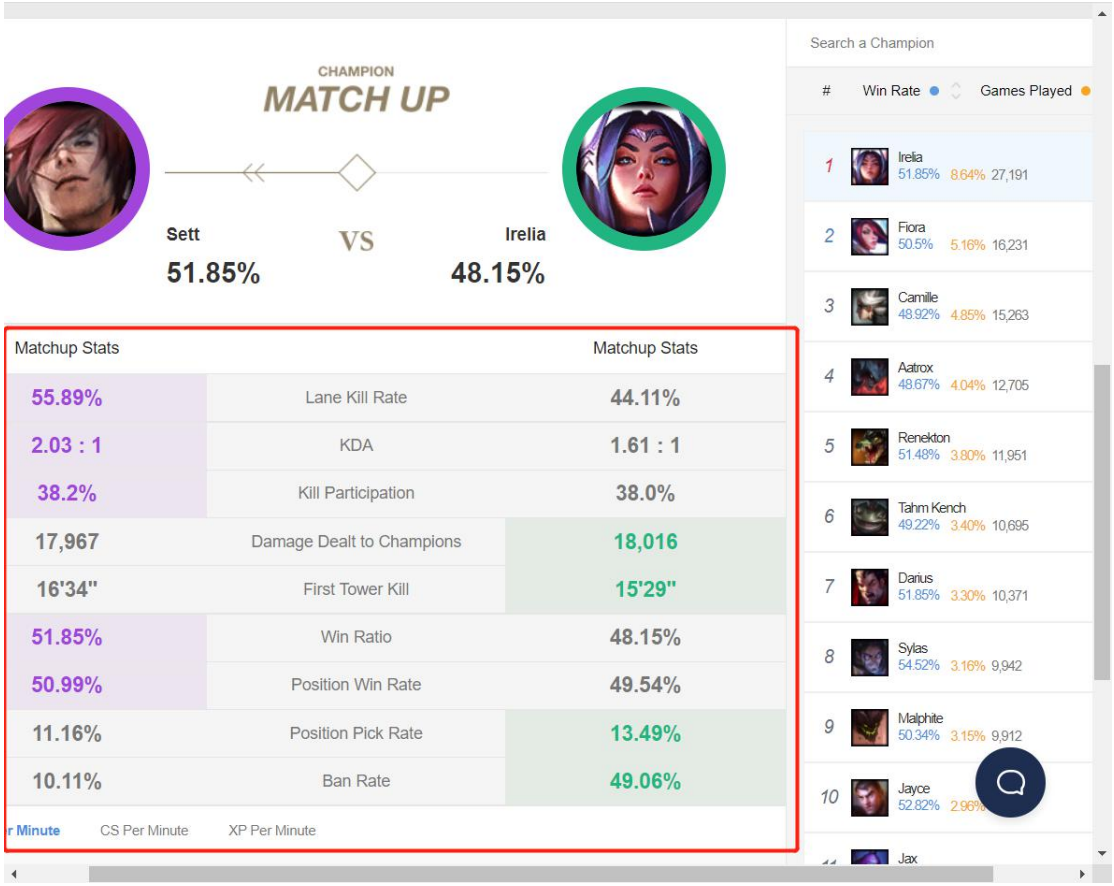
scrapy crawl gamelol_spider -o gamelol.csv

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:

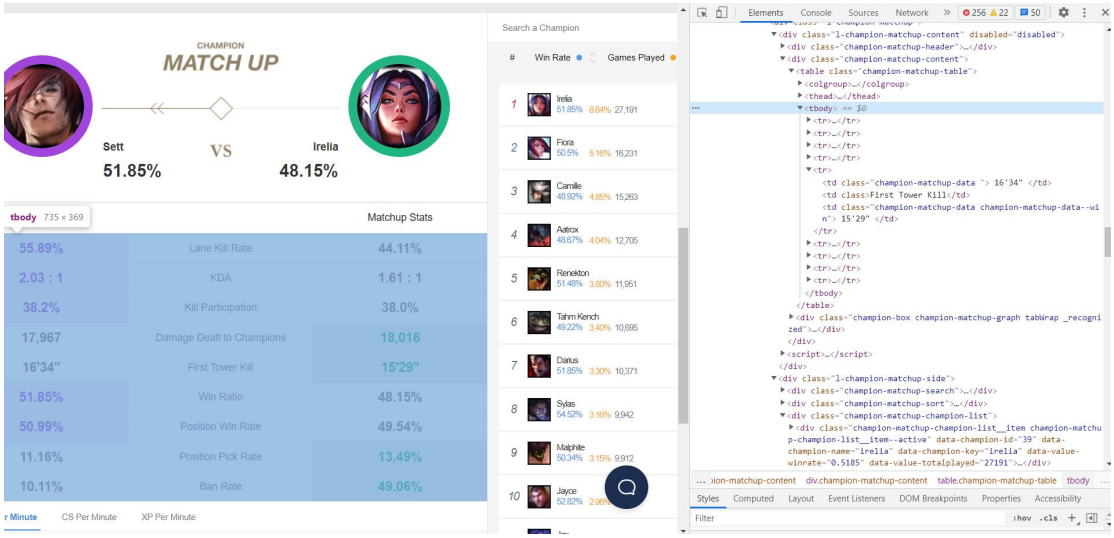
```
class GamelolSpiderSpider(scrapy.Spider):
    name = 'gamelol_spider'
    allowed_domains = ['euw.op.gg']
    start_urls = ['http://euw.op.gg/champion/sett/statistics/top/matchup']
    def parse(self, response):
        hero_lists = response.xpath("//div[@class='champion-matchup-champion-list']/div")
        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:



Check the website source code:



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.

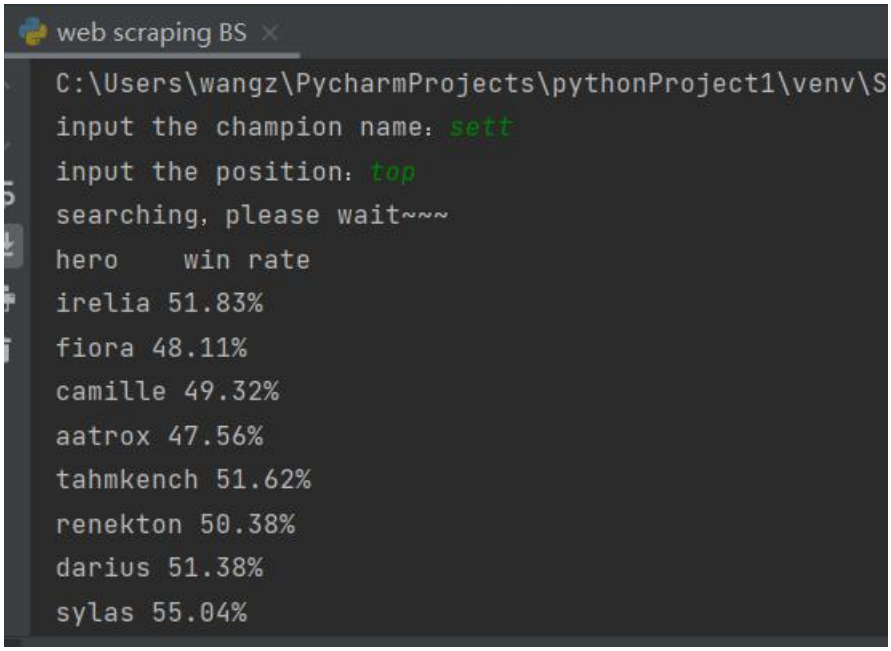
```
# loop all button
print('hero    win rate')
for button in buttons:
    # click button
    browser.execute_script("arguments[0].click();", button)
    # to find the left part of the table
    wait = ui.WebDriverWait(browser, wait_time)
    try:
        wait.until(
            lambda driver: driver.find_elements_by_xpath("//table[@class='champion-matchup-table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

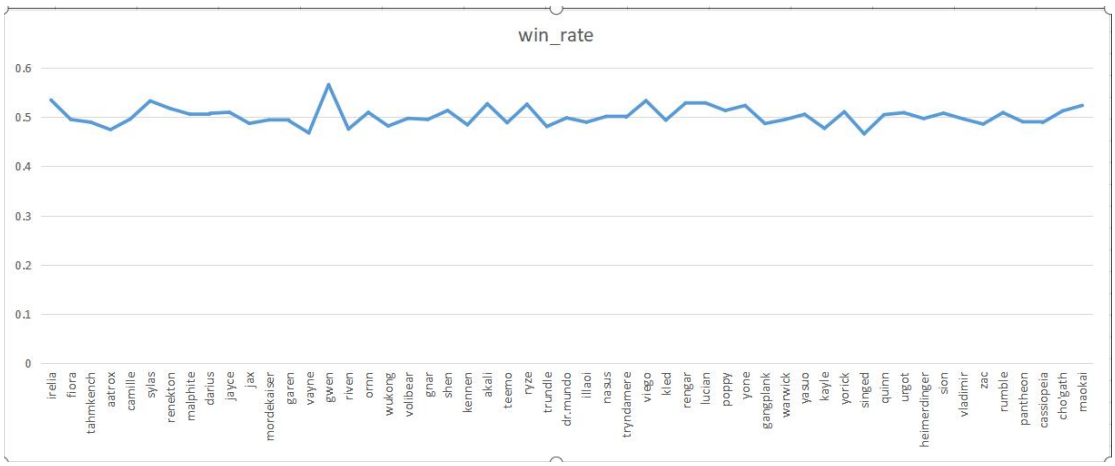
The output data we get and

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



	A	B
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