Project description
Webscrapping on Pixivimg

Yiqing Hu 455858
Shuai Hu 454835
Jin Huang 404899

Description of the topic and web-page

Beginning from the late 2022, a huge boost in AI industry occurs. The famous AI tool Chatgpt literally solves any regular questions that were sent to it. It is even considered that the AI tools have better performance than humans on some conventional occasions. Thus, the boost in AI industry strikes many other industries where AI might be a perfect substitute of humans. One of the most influenced industries by the AI inflow is the painting industry. Many painters lost their jobs due to the replacement by AI painting or drawing tools, which are capable of generating good drawings and paintings in a very small period of time. Pixiv is a painting website where people usually share their drawings, the majority of its contents are associated with anime. Thus, you may find many anime pictures on the website. During the past, it is a website that only accepts the man-made publications whereas now it opens a separate section for AI generated works. The project here is to scrap the some basic info of AI generated graphs (number of comments, number of likes, update time, number of views). And hopefully look for inner interactions among these data.

Scraper mechanism

Our project includes three scrappers BeautifulSoup, Scrapy and Selenium. For all the three scrappers, we are trying to crawl the basic information for each pictures on each page (The page displays the top50 pictures based on popularity). The basic mechanism are as follows.

First of all, in order to crawl links for pictures on various pages, we set a loop for date interval taking into account that the publications are arranged according to dates. To get the links for a desired day, we checked the construction of the urls and realized that the links are quite uniform. It is composed of a common domain and a formation of date series. Thus, all we have to do is take the common domains and plug in the standard date transformation. By controlling the date transformations we are able to set loops over dates and move through pages.

Next, for each page there are 50 pictures, each picture page is navigated by its own link. In order to get the links for each picture we observed that each pictures' link follows almost the same construction as the links for pages, it has a common domain and a specific "date-id" which varies among pictures. Thus, to get the links for each picture, all we have to do is to find the unique "date-id" for each picture and combine it with the common domain. This is how we look for links for each picture on one page.

Last, for the data that we would like to crawl for each picture, we are lucky to find that all data we are interested are stored under the same tag . Thus, we just have to go into the tag and extract the parts we would like to crawl.

Output

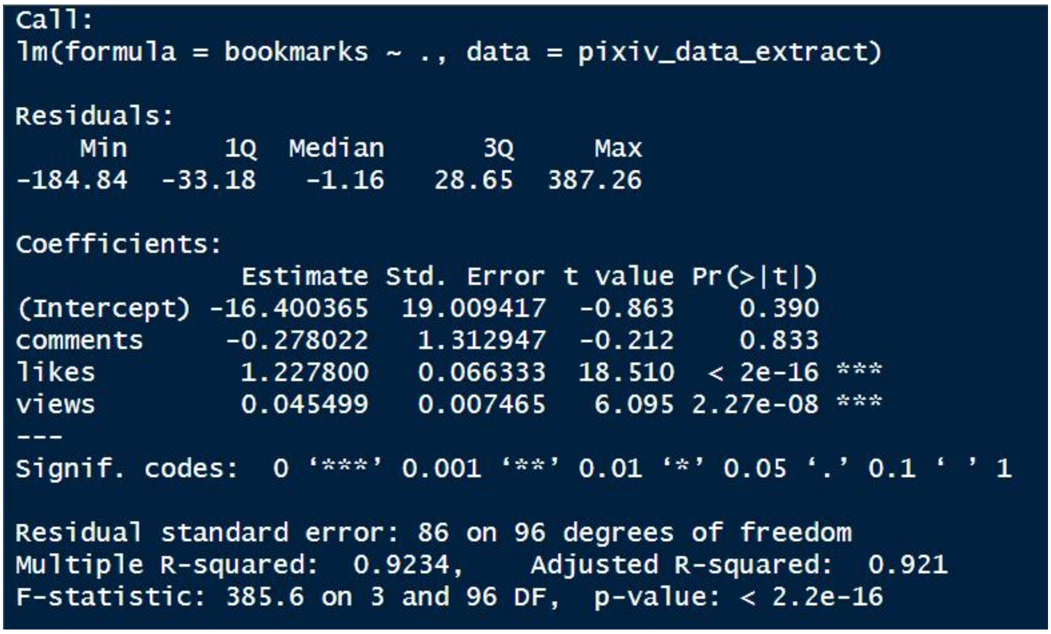
The data we crawl contains the following information.

aiType: AI type , Bookmarks: the count of people who collected the picture, Comments: the count of comments for each picture, crawl_time: a record for when we run our program, create_time:

when the picture was created, date: the date the picture was published, desc: description of a picture, img: the link to the picture, likes: the count of people who liked the picture, uid: user id, uname: user name, update_time: update time, tags: a representative of the picture, pid: picture id, views: count of times the picture was viewed.

Simple analysis

Basing on the crawled data for 100 links of pictures, we conducted a simple linear regression model using OLS. We obtain the following model output. The variables selected are bookmarks, comments, likes, views.



```
Call:
lm(formula = bookmarks ~ ., data = pixiv_data_extract)

Residuals:
    Min      1Q  Median      3Q     Max
-184.84  -33.18   -1.16   28.65  387.26

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -16.400365  19.009417  -0.863    0.390
comments     -0.278022   1.312947  -0.212    0.833
likes         1.227800   0.066333  18.510  < 2e-16 ***
views         0.045499   0.007465   6.095 2.27e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 86 on 96 degrees of freedom
Multiple R-squared:  0.9234,    Adjusted R-squared:  0.921
F-statistic: 385.6 on 3 and 96 DF,  p-value: < 2.2e-16
```

The model estimated above suggests the following conclusion:

A: comments have nothing to do with the behavior of people collecting pictures, since it is insignificant.

B: likes of a picture is positively related to the tendency of people collecting the picture. An additional like given to a picture increases the amount of collection of the picture by 1.2278.

C: number of views to a specific picture is positively related to the tendency of people collecting the picture. An additional increase in view of a specific picture increases the amount of collection by 0.045499.

Separation of the project work

The whole project was allocated based on the scrapper. Since we have three people in our group, each person is responsible for coding a scrapper. Yiqing Hu is responsible for the design of the whole project framework and writing Soup, Jin Huang is responsible for writing Scrapy and Shuai Hu is responsible for writing Selenium.

Computing time for each scraper

Soup : 80.15225505828857s Selenium : 888.7052929401398s Scrapy : 0m6.568s+2m13.024s

Conclusion : Selenium is slowest taking into account it has sleeping time. Thus, it is not recommended to use Selenium for large info scrapping.