Development Notes

2023-07-18

## Introduction

These are the development notes for scraping data from [Top Universities](www.topuniversities.com) and [Times Higher Education](www.timeshighereducation.com). Since, both websites present data as dynamic content they require the use of a headless browser to simulate the user clicking through the sites to generate the content.

The best tool for that kind of scraping in the R ecosystem is [RSelenium](https://docs.ropensci.org/RSelenium/).

## Installing RSelenium

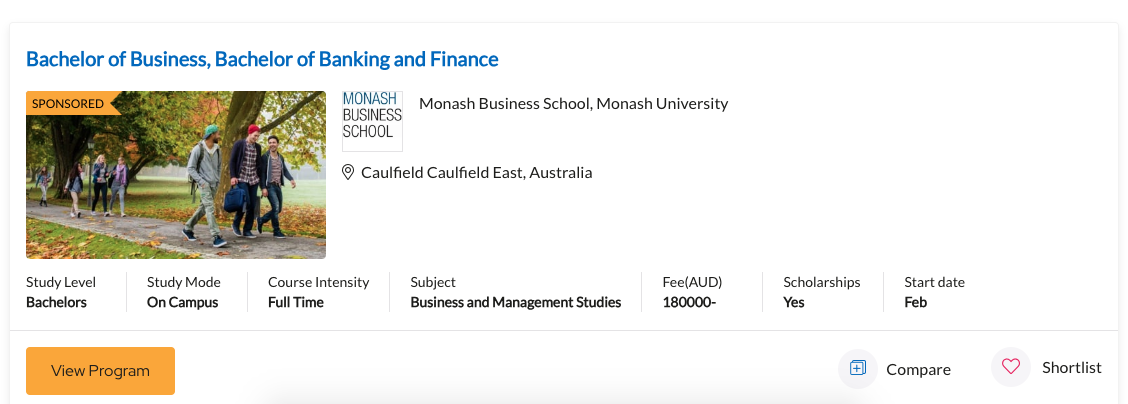
RSelenium requires the Rjava package. Rjava, in turn, requires access to the Java runtime. I recommend installing [Azul Java](https://www.azul.com/downloads/#zulu). Instructions vary with OS.

We also need a browser that works with RSelenium. The provided scripts work with Chrome.

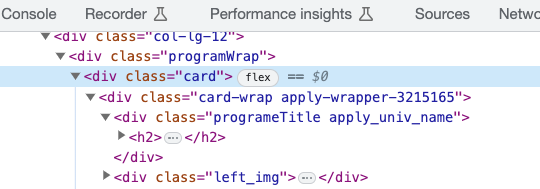
## Scraping Top Universities

The topuniversities.com websites provides a [programs finder](https://www.topuniversities.com/programs?qs_qp=topnav). We used this tool to extract information for programs related to Bioinformatics, Machine Learning and Data Science.

This is how the website presents the data.



However, to scrap the data, it is necessary to understand the structure of the generated markup in each page. In this case, the data is presented inside a div element with a card class. This code cards <- remDr$findElements(using="css", ".card") finds all such elements. The helper function extract\_row is applied to all found cards to extract all the available data.



The script for scraping the initial program list is scrap\_programs.R. This script has multiple functions to achieve this objective:

* extract\_row
* scrap\_page
* scrap\_programs

To start the scraping, the scripts uses RSelenium’s rsDriver to connect to a browser:

Once the connection with the browser is establish we navigate to a starting url.

The relevant data from this site is information from the programs. We scraped data for the subjects of Biological Sciences, Computer Science and Information Systems, Data Science, Genetics, Mathematics, Medicine Related Studies, Pharmacology, Pharmacy and Pharmacology, Physics and Astronomy, Statistics and Operational Research in Bahrain, Cyprus, Egypt, Iran, Iraq, Israel, Jordan, Kuwait, Lebanon, Oman, Palestine, Qatar, Saudi Arabia, Syria, Turkey, United Arab Emirates (UAE) and Yemen. The url above encodes a query for those programs. The website presents the data in pages of 25 elements.



Since the website presents a page with 25 programs at a time, the scripts iterates through all the pages. The function scrap\_page(i) scraps page i from this query. The 25 results from each page are saved to the temp folder.

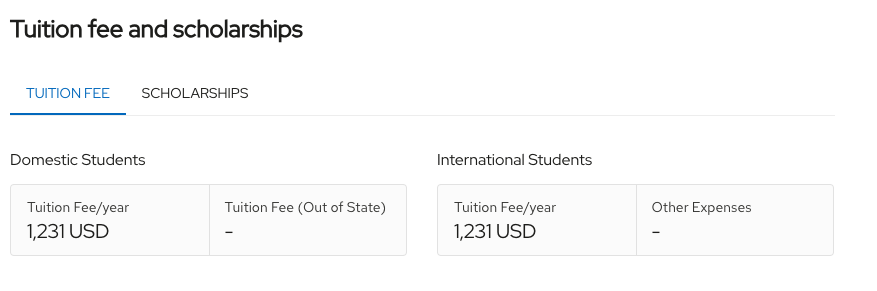
Since the requirements are not available in this page, we save a link to the program’s page on topuniversities.com to scrap the data from requirements and tuition in a second pass.

After scraping all the pages, we merge all the programs into a single data frame and save it.

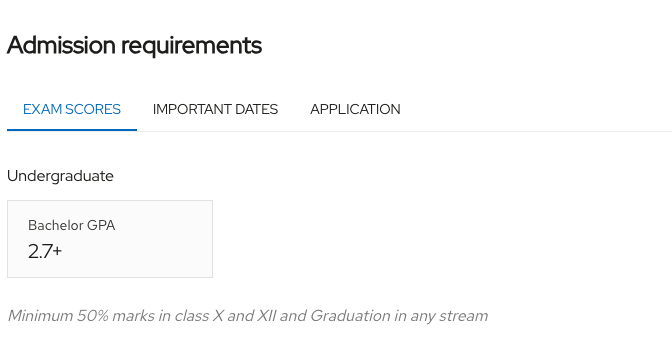
The script scrap\_programs\_tution\_fees.R takes the list of urls scraped previously and downloads the tuition fee and requirements data from the program pages.

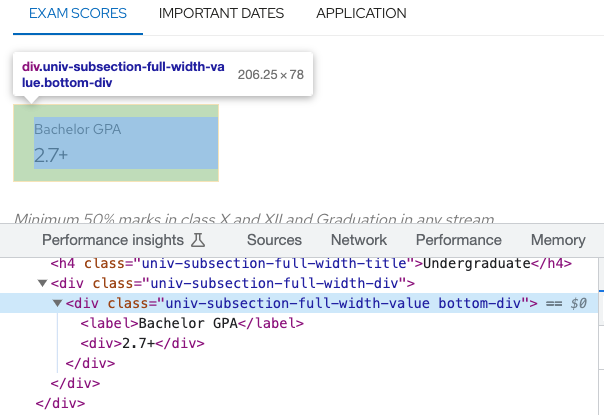
Since these are static html pages, it is not necessary to use a browser to access this data. The html pages are downloaded with rvest::read\_html.

This is how the webpages presents the tution fee data:



And the requirements:

 The associated markup:



To extract the necessary information, we download the html page using read\_html, and then we specify css selectors that have the information we need to the html\_elements.

### Cleaning up the topuniversities data

After scraping the data from all the programs, it is necessary to clean it by removing duplicates, parsing the requirements information and converting tuition fees to the same currency (British Pounds), removing programs that are not related to Machine Learning and Data Science.

Parsing each programs requirements involves taking the string from the html webpage and converting it to columns in our dataset. Each column corresponds to a requirement (TOEFL, IELTS, etc.). The value for each program in the column represents the minimum requirement of the program.

Currencies reported for the tuition fees are reportes in Euro, US Dollars, British Pounds and Singapore Dollars. I used the following exchange rates to convert currencies to British Pounds:

| Currency | Exchange |
| --- | --- |
| USD | 1.30 |
| EUR | 1.16 |
| SGD | 1.72 |

The final dataset:

Data summary

|  |  |
| --- | --- |
| Name | df |
| Number of rows | 5356 |
| Number of columns | 35 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 13 |
| numeric | 22 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- |
| university | 0 | 1.00 | 3 | 46 | 0 | 149 | 0 |
| university\_link | 0 | 1.00 | 48 | 104 | 0 | 154 | 0 |
| requirements | 0 | 1.00 | 8 | 203 | 0 | 1837 | 0 |
| study\_level | 0 | 1.00 | 3 | 9 | 0 | 3 | 0 |
| location | 0 | 1.00 | 0 | 97 | 1231 | 187 | 0 |
| program\_title | 0 | 1.00 | 7 | 134 | 0 | 4618 | 0 |
| program\_link | 0 | 1.00 | 61 | 176 | 0 | 5353 | 0 |
| subject | 0 | 1.00 | 7 | 42 | 0 | 39 | 0 |
| study\_mode | 3776 | 0.29 | 6 | 9 | 0 | 3 | 0 |
| course\_intensity | 2704 | 0.50 | 9 | 9 | 0 | 2 | 0 |
| duration | 825 | 0.85 | 8 | 10 | 0 | 31 | 0 |
| currency | 2851 | 0.47 | 3 | 3 | 0 | 4 | 0 |
| duration\_units | 825 | 0.85 | 6 | 6 | 0 | 1 | 0 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| fee | 2851 | 0.47 | 20964.14 | 14131.95 | 1190 | 9250.00 | 14700.0 | 33250 | 124318.0 | ▇▆▁▁▁ |
| duration\_length | 825 | 0.85 | 36.63 | 16.64 | 1 | 24.00 | 36.0 | 48 | 216.0 | ▇▆▁▁▁ |
| fee\_gbp | 2851 | 0.47 | 21388.79 | 15638.57 | 1190 | 9250.00 | 14700.0 | 33250 | 161613.4 | ▇▂▁▁▁ |
| toefl | 2001 | 0.63 | 85.44 | 19.90 | 0 | 79.00 | 87.0 | 90 | 550.0 | ▇▁▁▁▁ |
| ielts | 23 | 1.00 | 6.15 | 2.48 | 2 | 6.00 | 6.0 | 6 | 185.0 | ▇▁▁▁▁ |
| bachelor\_gpa | 4054 | 0.24 | 2.61 | 1.95 | 2 | 2.00 | 2.7 | 3 | 70.0 | ▇▁▁▁▁ |
| cambridge\_cae\_advanced | 3108 | 0.42 | 171.91 | 18.37 | 0 | 169.00 | 176.0 | 176 | 193.0 | ▁▁▁▁▇ |
| pte\_academic | 2570 | 0.52 | 58.87 | 6.79 | 0 | 54.00 | 59.0 | 62 | 176.0 | ▁▇▁▁▁ |
| a\_levels | 3488 | 0.35 | 9.14 | 28.51 | 0 | 0.00 | 0.0 | 0 | 128.0 | ▇▁▁▁▁ |
| international\_baccalaureate | 3010 | 0.44 | 29.91 | 23.37 | 0 | 27.25 | 30.0 | 34 | 1090.0 | ▇▁▁▁▁ |
| ucas\_tariff | 4083 | 0.24 | 94.48 | 38.40 | 0 | 80.00 | 104.0 | 112 | 440.0 | ▃▇▁▁▁ |
| atar | 4817 | 0.10 | 85.70 | 7.70 | 70 | 80.00 | 85.0 | 92 | 99.0 | ▃▅▇▇▃ |
| sat | 4175 | 0.22 | 1058.66 | 375.89 | 0 | 1100.00 | 1100.0 | 1290 | 1500.0 | ▂▁▁▇▆ |
| act | 5238 | 0.02 | 25.18 | 1.39 | 23 | 25.00 | 25.0 | 26 | 28.0 | ▅▇▅▃▁ |
| gre | 5343 | 0.00 | 234.62 | 78.09 | 150 | 150.00 | 304.0 | 304 | 304.0 | ▇▁▁▁▇ |
| gpa | 5318 | 0.01 | 2.80 | 0.37 | 2 | 2.71 | 3.0 | 3 | 3.0 | ▂▁▁▂▇ |
| btec\_qualifications | 4236 | 0.21 | 4.17 | 19.76 | 0 | 0.00 | 0.0 | 0 | 128.0 | ▇▁▁▁▁ |
| op | 5292 | 0.01 | 18.81 | 6.29 | 1 | 21.00 | 21.0 | 21 | 21.0 | ▁▁▁▁▇ |
| sqa\_highers | 4991 | 0.07 | 17.75 | 38.96 | 0 | 0.00 | 0.0 | 0 | 128.0 | ▇▁▁▁▁ |
| sqa\_advanced | 5271 | 0.02 | 1.41 | 13.02 | 0 | 0.00 | 0.0 | 0 | 120.0 | ▇▁▁▁▁ |
| as\_levels | 5302 | 0.01 | 27.85 | 38.21 | 0 | 0.00 | 20.0 | 20 | 120.0 | ▇▁▁▁▂ |
| year\_12\_scores | 5353 | 0.00 | 60.00 | 0.00 | 60 | 60.00 | 60.0 | 60 | 60.0 | ▁▁▇▁▁ |

## Scraping timeshighereducation.com

The relevant data from this website is in two pages:

The website also renders data dynamically, so we connect to it through a browser. The data is presented in a single page, so it is easier to scrap. The script that does this job is scrap\_rankings.R.

## Geocoding and merging

To geocode the downloaded data, I used the [Google Maps API](https://developers.google.com/maps/) and the tidygeocoder [package](https://jessecambon.github.io/tidygeocoder/). The script that does this is geocode\_programs.R. The Google maps API requires an API key.

The final data for the Shiny app requires merging the data from different sources. Some of the university names are different, so a little bit of manual processing was necessary. That is achieved in the merge\_the\_topuniversities.R script.

The final data is in the data\_output folder.