EDA Part 1: Data Engineering

Scraping

Web Servers

- A server is a long running process (also called daemon) which listens on a pre-specified port
- and responds to a request, which is sent using a protocol called HTTP
- A browser must first we must parse the url. Everything after a # is a fragment. Until then its the DNS name or ip address, followed by the URL.

Example

Our notebooks also talk to a local web server on our machines: `http://localhost:8888/Documents/cs109/BLA.ipynb#something`

- protocol is `http`, hostname is `localhost`, port is `8888`
- url is `/Documents/cs109/BLA.ipynb`
- url fragment is `` `#something ``

Request is sent to localhost on port 8888. It says:

```
Request:
GET /request-URI HTTP/version
```

Example with Response: Google

```
GET / HTTP/1.0
Host: www.google.com
```

```
HTTP/1.0 200 OK
Date: Mon, 14 Nov 2016 04:49:02 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is ..."
Server: gws
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Set-Cookie: NID=90=gb5q7b0...; expires=Tue, 16-May-2017 04:49:02 GMT; path=/; domain=.google.com; HttpOnly
Accept-Ranges: none
Vary: Accept-Encoding

<!doctype html><html itemscope=""
itemtype="http://schema.org/WebPage" lang="en">
<head><meta content="Search the world's information,
```

HTTP Status Codes[6]

- 200 OK:
  Means that the server did whatever the client wanted it to, and all is well.
- 201 Created:
  The request has been fulfilled and resulted in a new resource being created. The newly created resource can be referenced by the URI(s) returned in the entity of the response, with the most specific URI for the resource given by a Location header field.
- 400: Bad request
  The request sent by the client didn't have the correct syntax.
- 401: Unauthorized
  Means that the client is not allowed to access the resource. This may change if the client retries with an authorization header.
- 403: Forbidden
  The client is not allowed to access the resource and authorization will not help.
- 404: Not found
  Seen this one before? :) It means that the server has not heard of the resource and has no further clues as to what the client should do about it. In other words: dead link.
- 500: Internal server error
  Something went wrong inside the server.
- 501: Not implemented
  The request method is not supported by the server.

---

[6] (from http://www.garshol.priv.no/download/text/http-tut.htm)

requests

- great module built into python for http requests

```
req = requests.get("https://en.wikipedia.org/wiki/Harvard_University")
```

```
<Response [200]>
```

```
page = req.text
```


```
'<!DOCTYPE html>\n<html class="client-nojs" lang="en" dir="ltr">\n<head>\n
<meta charset="UTF-8"/>\n<title>Harvard University -
Wikipedia</title>\n<script>document.documentElement.className =
document.documentElement.className.replace( /(^|\\s)client-nojs(\\s|$)/,
"$1client-js$2"
);</script>\n<script>(window.RLQ=window.RLQ||[]).push(function(){mw.config.set({
"wgCanonicalNamespace":"","wgCanonicalSpecialPageName":false,"wgNamespaceNumber"
:0,"wgPageName":"Harvard_University","wgTitle":"Harva...'
```

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction

Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools

What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export

Harvard University

From Wikipedia, the free encyclopedia

Coordinates: 42°22′28″N 71°07′01″W

"Harvard" redirects here. For other uses, see Harvard (disambiguation).

Harvard University is a private Ivy League research university in Cambridge, Massachusetts, established in 1636, whose history, influence, and wealth have made it one of the world's most prestigious universities.[7]

Established originally by the Massachusetts legislature and soon thereafter named for John Harvard (its first benefactor), Harvard is the United States' oldest institution of higher learning,[8] and the Harvard Corporation (formally, the *President and Fellows of Harvard College*) is its first chartered corporation. Although

Harvard University

Latin: *Universitas Harvardiana*

Former names	Harvard College
Motto	*Veritas*[1]
Motto in English	Truth
Type	Private research
Established	1636[2]
Endowment	$34.541 billion (2016)[3]

Python data scraping

- Why scrape the web?
- vast source of information, combine with other data sets
- companies have not provided APIs
- automate tasks
- keep up with sites
- fun!

copyrights and permission:

- be careful and polite
- give credit
- care about media law
- don't be evil (no spam, overloading sites, etc.)

Robots.txt

- specified by web site owner
- gives instructions to web robots (aka your script)
- is located at the top-level directory of the web server

e.g.: h'p://google.com/robots.txt

HTML

- angle brackets
- should be in pairs, eg `<p>Hello</p>`
- maybe in implicit bears, such as `<br/>`

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ttle</title>
  </head>
  <body>
    <h1>Body Title</h1>
    <p>Body Content</p>
  </body>
</html>
```

Developer Tools

- ctrl/cmd shift i in chrome
- cmd-option-i in safari
- look for "inspect element"
- locate details of tags

Beautiful Soup

- will normalize dirty html
- basic usage

```
import bs4
## get bs4 object
soup = bs4.BeautifulSoup(source)
## all a tags
soup.findAll('a')
## first a
soup.find('a')
## get all links in the page
link_list = [l.get('href') for l in soup.findAll('a')]
```

HTML is a tree

```
tree = bs4.BeautifulSoup(source)

## get html root node
root_node = tree.html

## get head from root using contents
head = root_node.contents[0]

## get body from root
body = root_node.contents[1]

## could directly access body
tree.body
```

Demographics table we want

Student life

Demographics of student body[124][125][126]

	Undergraduate	Graduate and professional	U.S. census
Asian/Pacific Islander	17%	11%	5%
Black/non-Hispanic	6%	4%	12%
Hispanics of any race	9%	5%	16%
White/non-Hispanic	46%	43%	64%
Mixed race/other	10%	8%	9%
International students	11%	27%	N/A

Student body

In the last six years, Harvard's studer
21,000, across all programs.[127] Har
undergraduate programs, 3,738 stud
10,722 students in professional progr
population is 51% female, the gradua
professional population is 49% femal

Athletics

Main article: Harvard Crimson

The Harvard Crimson competes in 42 intercollegiate sports in the NCAA Division I Ivy League. Harvard has an intense athletic rivalry with Yale University culminating in *The Game*, although the Harvard–Yale Regatta predates the football game. This rivalry is put aside every two years when the Harvard and Yale

Table with sole class `wikitable`

United States, both for students and parents.[122] *College ROI Report: Best Value Colleges* by PayScale puts Harvard 22nd nationwide in the most recent 2016 edition.[123]

Student life

Demographics of student body[124][125][126]

	Undergraduate	Graduate and professional	U.S. census
Asian/Pacific Islander	17%	11%	5%
Black/non-Hispanic	6%	4%	12%
Hispanics of any race	9%	5%	16%
White/non-Hispanic	46%	43%	64%
Mixed race/other	10%	8%	9%
International students	11%	27%	N/A

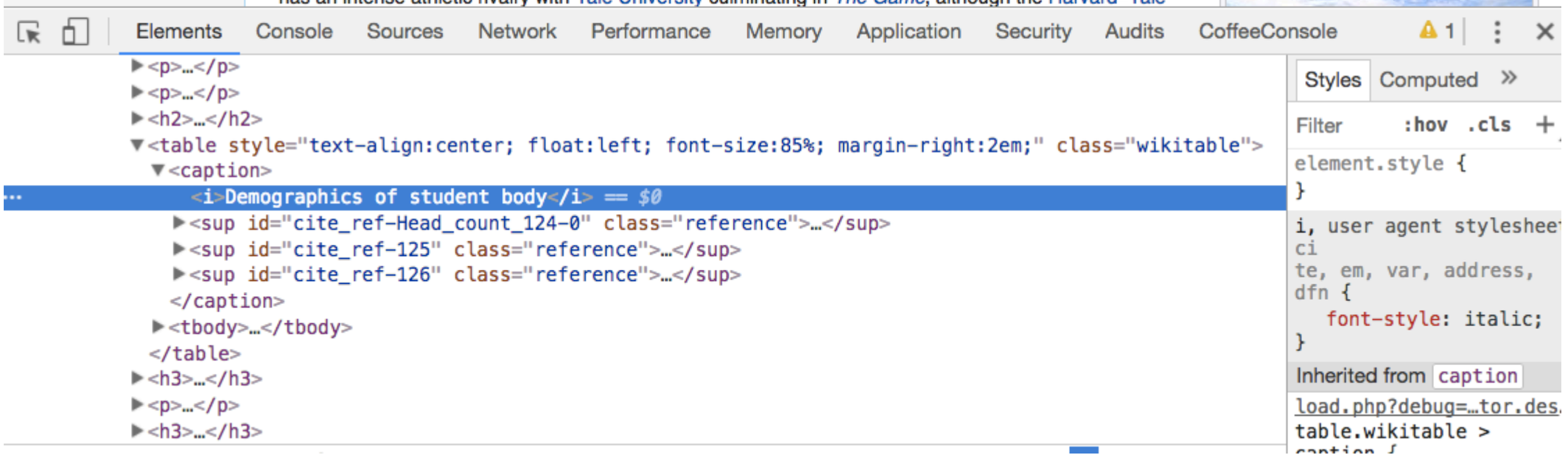
Student body

In the last six years, Harvard's student population ranged from 19,000 to 21,000, across all programs.[127] Harvard enrolled 6,655 students in undergraduate programs, 3,738 students in graduate programs, and 10,722 students in professional programs.[124] The undergraduate population is 51% female, the graduate population is 48% female, and the professional population is 49% female.[124]

Athletics

Main article: Harvard Crimson

The Harvard Crimson competes in 42 intercollegiate sports in the NCAA Division I Ivy League. Harvard has an intense athletic rivalry with Yale University culminating in *The Game*, although the Harvard–Yale



Beautiful Soup Code

```
dfinder = lambda tag: tag.name=='table' and tag.get('class') == ['wikitable']
table_demographics = soup.find_all(dfinder)
rows = [row for row in table_demographics[0].find_all("tr")]
header_row = rows[0]
columns = [col.get_text() for col in header_row.find_all("th") if col.get_text()]
columns = [rem_nl(c) for c in columns]
indexes = [row.find("th").get_text() for row in rows[1:]]
values = []
for row in rows[1:]:
    for value in row.find_all("td"):
        values.append(to_num(value.get_text()))
stacked_values_lists = [values[i::3] for i in range(len(columns))]
stacked_values_iterator = zip(*stacked_values_lists)
df = pd.DataFrame(list(stacked_values_iterator), columns=columns, index=indexes)
```