Importing flat files from the web

INTERMEDIATE IMPORTING DATA IN PYTHON

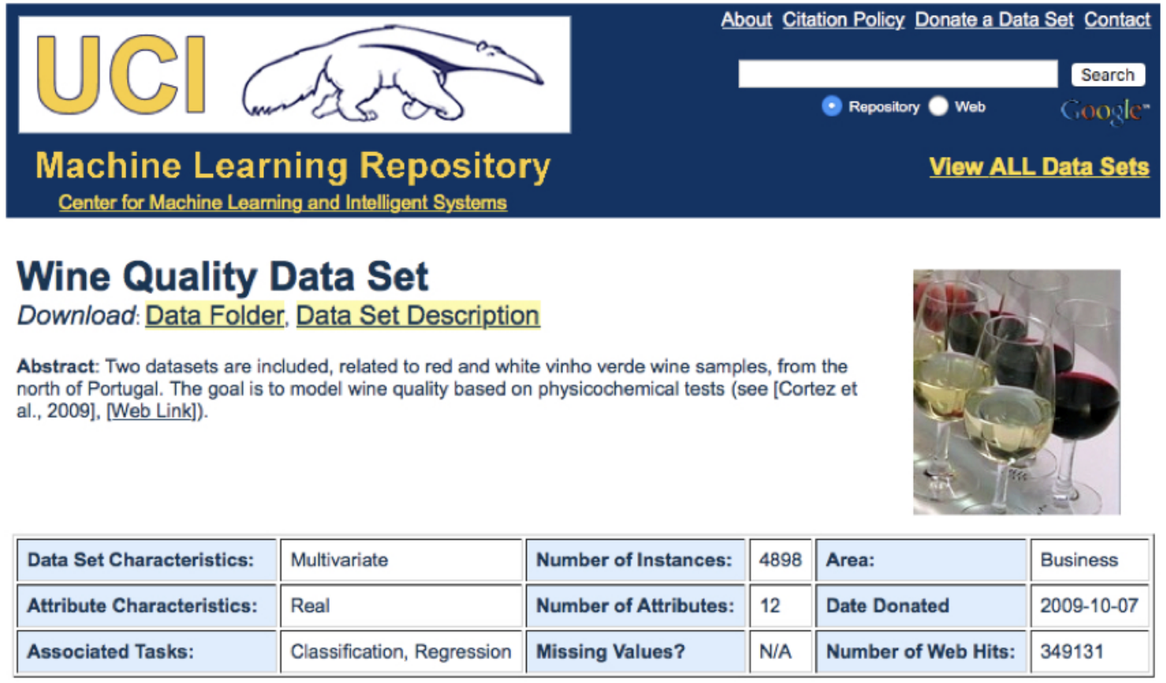
Hugo Bowne-Anderson
Data Scientist at DataCamp

datacamp

You're already great at importing!

- Flat files such as .txt and .csv
- Pickled files, Excel spreadsheets, and many others!
- Data from relational databases
- You can do all these locally
- What if your data is online?

Can you import web data?



- You can: go to URL and click to download files
- BUT: not reproducible, not scalable

You'll learn how to...

- Import and locally save datasets from the web
- Load datasets into pandas DataFrames
- Make HTTP requests (GET requests)
- Scrape web data such as HTML
- Parse HTML into useful data (BeautifulSoup)
- Use the urllib and requests packages

The urllib package

- Provides interface for fetching data across the web
- `urlopen()` - accepts URLs instead of file names

How to automate file download in Python

```
from urllib.request import urlretrieve
url = 'http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv'
urlretrieve(url, 'winequality-white.csv')
```

```
('winequality-white.csv', <http.client.HTTPMessage at 0x103cf1128>)
```

6. How to automate file download in Python
All we have done here is imported a function called urlretrieve from the request subpackage of the urllib package, we assigned the relevant URL as a string to the variable url. We then used the urlretrieve function to write the contents of the url to a file 'winequality-white dot csv'. Now it's your turn to do the same but for red wine!

Let's practice!

INTERMEDIATE IMPORTING DATA IN PYTHON

HTTP requests to import files from the web

INTERMEDIATE IMPORTING DATA IN PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

datacamp

URL

- Uniform/Universal Resource Locator
- References to web resources
- Focus: web addresses
- Ingredients:
  - Protocol identifier - http:
  - Resource name - datacamp.com
- These specify web addresses uniquely

The vast majority of URLs are web addresses, but they can refer to a few other things, such as file transfer protocols (FTP) and database access. We'll currently focus on those URLs that are web addresses OR the locations of websites. Such a URL consists of 2 parts, a protocol identifier http or https and a resource name such as datacamp dot com. The combination of protocol identifier and resource name uniquely specifies the web address!

HTTP

- HyperText Transfer Protocol
- Foundation of data communication for the web
- HTTPS - more secure form of HTTP
- Going to a website = sending HTTP request
  - GET request
- `urlretrieve()` performs a GET request
- HTML - HyperText Markup Language

"The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web." Note that HTTPS is a more secure form of HTTP. Each time you go to a website, you are actually sending an HTTP request to a server. This request is known as a GET request, by far the most common type of HTTP request. We are actually performing a GET request when using the function urlretrieve. The ingenuity of urlretrieve also lies in fact that it not only makes a GET request but also saves the relevant data locally.

GET requests using urllib

```
from urllib.request import urlopen, Request
url = "https://www.wikipedia.org/"
request = Request(url)
response = urlopen(request)
html = response.read()
response.close()
```

4. GET requests using urllib
To extract the html from the wikipedia home page, you import the necessary functions, specify the URL, package the GET request using the function Request, send the request and catch the response using the function urlopen. This returns an HTTPResponse object, which has an associated read method. You then apply this read method to the response, which returns the HTML as a string, which you store in the variable html.

GET requests using requests

- Used by “her Majesty's Government, Amazon, Google, Twilio, NPR, Obama for America, Twitter, Sony, and Federal U.S. Institutions that prefer to be unnamed”

GET requests using requests

- One of the most downloaded Python packages

```
import requests
url = "https://www.wikipedia.org/"
r = requests.get(url)
text = r.text
```

6. GET requests using requests
Lets now see requests at work. Here, you import the package requests, specify the URL, package the request, send the request and catch the response with a single function requests dot get; apply the text method to the response which returns the HTML as a string.

Let's practice!

INTERMEDIATE IMPORTING DATA IN PYTHON

Scraping the web in Python

INTERMEDIATE IMPORTING DATA IN PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

datacamp

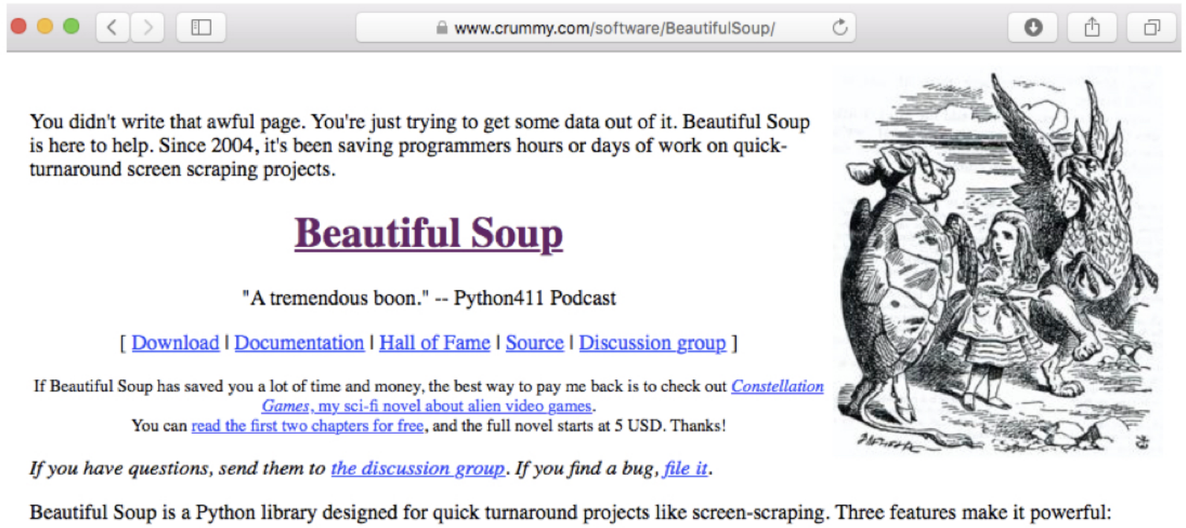
HTML

- Mix of unstructured and structured data
- Structured data:
  - Has pre-defined data model, or
  - Organized in a defined manner
- Unstructured data: neither of these properties

```
<!DOCTYPE html>
<html>
    <head>
        <!-- SEO -->
    <meta charset="utf-8" />
<title>DataCamp: The Easy Way To Learn R &amp;amp; Data Science Online</title>
```

BeautifulSoup

- Parse and extract structured data from HTML



BeautifulSoup? In web development, the term "tag soup" refers to structurally or syntactically incorrect HTML code written for a web page. What Beautiful Soup does best is to make tag soup beautiful again and to extract information from it with ease! In fact, the main object created and queried when using this package is called BeautifulSoup and it has a very important associated method called prettify! Lets now see BeautifulSoup in Beautiful Action!

3. BeautifulSoup
In general, to turn HTML that you have scraped from the world wide web into useful data, you'll need to parse it and extract structured data from it.

- Make tag soup beautiful and extract information

BeautifulSoup

```
from bs4 import BeautifulSoup
import requests
url = 'https://www.crummy.com/software/BeautifulSoup/'
r = requests.get(url)
html_doc = r.text
soup = BeautifulSoup(html_doc)
```

4. BeautifulSoup
Once again, you use requests to scrape the HTML from the web. Then you create a BeautifulSoup object from the resulting HTML and prettify.

Prettified Soup

```
print(soup.prettify())
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/TR/REC-html40/transitional.dtd">
<html>
 <head>
  <meta content="text/html; charset=utf-8" http-equiv="Content-Type"/>
  <title>
   Beautiful Soup: We called him Tortoise because he taught us.
  </title>
  <link href="mailto:leonardr@segfault.org" rev="made"/>
  <link href="/nb/themes/Default/nb.css" rel="stylesheet" type="text/css"/>
  <meta content="Beautiful Soup: a library designed for screen-scraping HTML and XML." name="Description"/>
  <meta content="Markov Approximation 1.4 (module: leonardr)" name="generator"/>
  <meta content="Leonard Richardson" name="author"/>
 </head>
 <body alink="red" bgcolor="white" link="blue" text="black" vlink="660066">
  <img align="right" src="10.1.jpg" width="250"/>
  <br/>
  <p>
```

5. Prettified Soup
Printing the prettified Soup and the original HTML, you can see that for, example, the prettified Soup is indented in the way you would expect properly written HTML to be.

Exploring BeautifulSoup

- Many methods such as:

```
print(soup.title)
```

```
<title>Beautiful Soup: We called him Tortoise because he taught us.</title>
```

```
print(soup.get_text())
```

```
Beautiful Soup: We called him Tortoise because he taught us.
You didn't write that awful page. You're just trying to
get some data out of it. Beautiful Soup is here to
help. Since 2004, it's been saving programmers hours or
days of work on quick-turnaround screen scraping
projects.
```

6. Exploring BeautifulSoup
You'll explore a few of the methods that you can apply to your soupified HTML in the following exercises, such as title and get_text, which extract the title and text, respectively.

Exploring BeautifulSoup

- `find_all()`

7. Exploring BeautifulSoup
You'll also work with the Soupy method find_all in order to extract the URLs of all of the hyperlinks in the HTML.

```
for link in soup.find_all('a'):
    print(link.get('href'))
```

```
bs4/download/
#Download
bs4/doc/
#HallOfFame
https://code.launchpad.net/beautifulsoup
https://groups.google.com/forum/?fromgroups#!forum/beautifulsoup
http://www.candlemarkandgleam.com/shop/constellation-games/
http://constellation.crummy.com/Constellation%20Games%20excerpt.html
https://groups.google.com/forum/?fromgroups#!forum/beautifulsoup
https://bugs.launchpad.net/beautifulsoup/
http://lxml.de/
http://code.google.com/p/html5lib/
```

Let's practice!

INTERMEDIATE IMPORTING DATA IN PYTHON