

CSS 330 Data wrangling and visualization

Lecture 2

Instructor: Nabigazinova Elnura

Collecting the data

What is the DATA

Data are characteristics or information, usually numerical, that are collected through observation. In a more technical sense, **data** are a set of values of qualitative or quantitative variables about one or more persons or objects, while a datum (singular of **data**) is a single value of a single variable. From [Wikipedia](#)

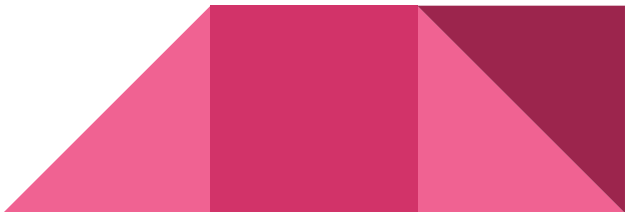


Data Sources

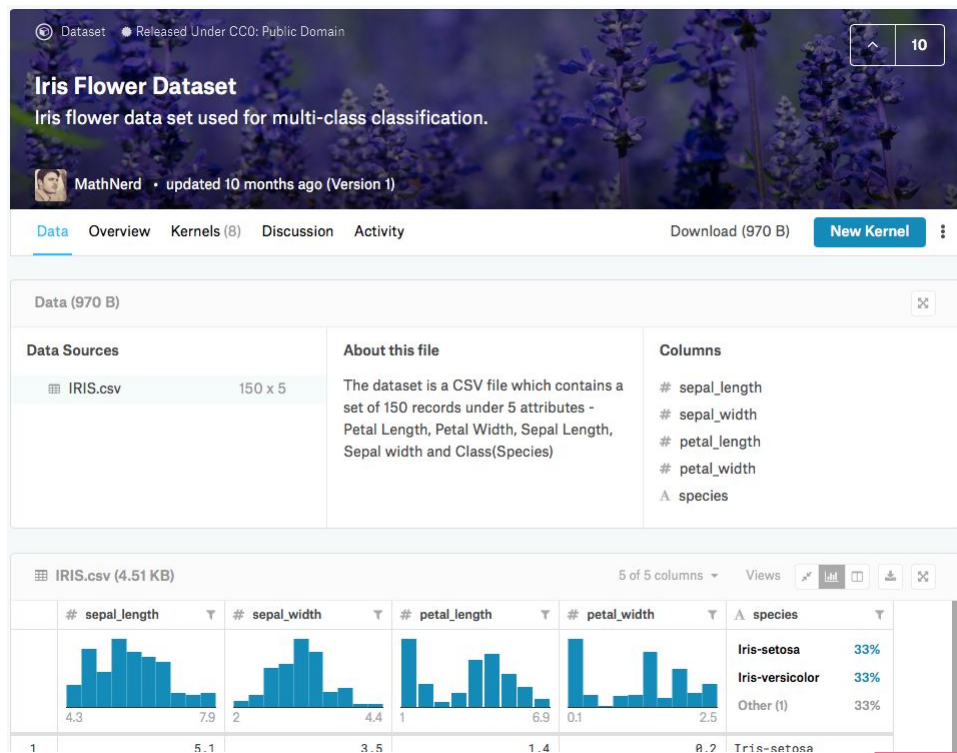
Structured data

- From databases
- Form public datasets
- Tables
- Lists
- etc.

Unstructured data


- Web sites
 - Log files
 - API
 - Network traffic*?
 - etc.
- 

Public Datasets




Web scraping

Web scraping, web harvesting, or web data extraction is data scraping used for extracting data from websites. Web scraping software may access the World Wide Web directly using the Hypertext Transfer Protocol, or through a web browser. While web scraping can be done manually by a software user, the term typically refers to automated processes implemented using a bot or web crawler. It is a form of copying, in which specific data is gathered and copied from the web, typically into a central local database or spreadsheet, for later retrieval or analysis.
From Wikipedia



Scraping Rules

1. You should check a site's terms and conditions before you scrape them. It's their data and they likely have some rules to govern it.
 2. Be nice - A computer will send web requests much quicker than a user can. Make sure you space out your requests a bit so that you don't hammer the site's server.
 3. Scrapers break - Sites change their layout all the time. If that happens, be prepared to rewrite your code.
 4. Web pages are inconsistent - There's sometimes some manual clean up that has to happen even after you've gotten your data.
- 

Code planning

- Check the structure of the site, is there any patterns how data is placed?
- Open the source code of the site, overview all tag that are used in the HTML source code.
- If you're in Chrome or Firefox, highlight necessary lines, right-click, and select Inspect Element. (In Safari you will need to enable Develop menu in Preferences first)
- Define distinguishing parameters how data can be found in the code.



BeautifulSoup library

Samples:

```
soup.title # <title>The Dormouse's story</title>
soup.title.name # u'title'
soup.title.string # u'The Dormouse's story'
soup.title.parent.name # u'head'
soup.p # <p class="title"><b>The Dormouse's story</b></p>
soup.p['class'] # u'title'
soup.a # <a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>
soup.find_all('a')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```



Objects

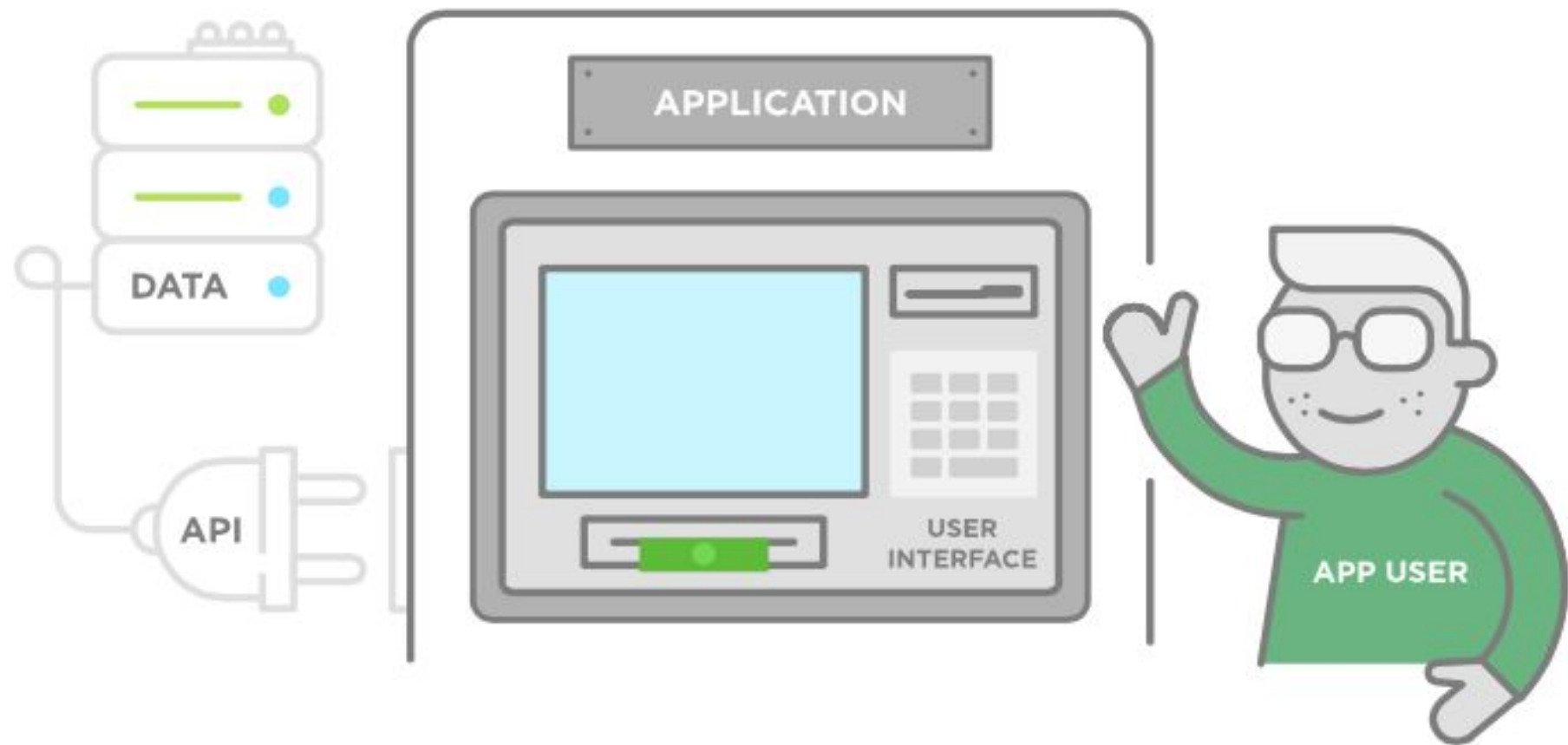
- Beautiful Soup transforms a complex HTML document into a complex tree of Python objects. But you'll only ever have to deal with about four kinds of objects:
 - **Tag** - *A Tag object corresponds to an XML or HTML tag in the original document*
 - **NavigableString** - *A string corresponds to a bit of text within a tag. Beautiful Soup uses the NavigableString class to contain these bits of text*
 - **BeautifulSoup** - *The BeautifulSoup object itself represents the document as a whole. For most purposes, you can treat it as a Tag object*
 - **Comment** - *just a special type of NavigableString*

API

In many current systems we have useful interface to connect to them and receive data. This interface is called API (Application Programming Interface).

API is used by programmers to write programs that can talk to other programs. Thus, access functionality from other systems.






API types

- REST
- SOAP
- RPC
- POSIX
- JavaDoc
- etc.



What is API?

- APIs do a lot of heavy lifting, both in mobile and on the web. They're responsible for nearly everything we do—and with just a few taps or clicks, let you do things like order a pizza, book a hotel, rate a song, or download software.
 - As you may know, the travel site aggregates information from many different hotels. When you click “search,” the site then interacts with each hotel's API, which delivers results for available rooms that meet your criteria. This can all happen within seconds because of an API, which acts like a messenger that runs back and forth between applications, databases, and devices.
- 

Application

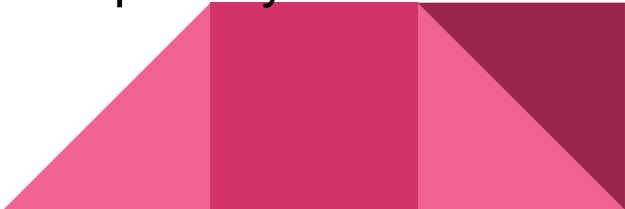
- Think of an application like an ATM. When you walk up to an ATM, you expect it will allow you to access your account and complete a transaction like withdrawing cash.
- Like an ATM, an app provides a function, but it's not doing this all by itself—it needs to communicate both with the user, and with the “bank” it's accessing.



Programming

APIs allow the ATM to communicate with your bank. The programming is the engineering part of the app's software that translates input into output.

In other words, it translates your request for cash to the bank's database, verifies there's enough cash in your account to withdraw the requested amount, the bank grants permission, then the ATM communicates back to the bank how much you withdrew so that the bank can update your balance.



Interface

A user interface (UI) is how we interact with an application. In the case of the ATM, it's the screen, keypad, and cash slot—where the input and output occurs. We enter our pin number, punch in how much cash we'd like to withdraw, then take the cash that's spit out. Interfaces are how we communicate with a machine.

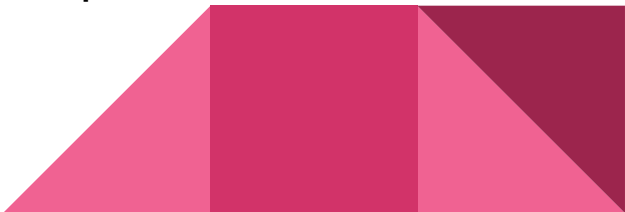
With APIs, it's much the same, only we're replacing users with software.



Make your code mean something

API consumers are capable of sending GET, POST, PUT, and DELETE verbs, which greatly enhance the clarity of a given request.

Generally, the four primary HTTP verbs are used as follows:

- GET - Read a specific resource (by an identifier) or a collection of resources.
 - PUT - Update a specific resource (by an identifier) or a collection of resources.
Can also be used to create a specific resource if the resource identifier is known before-hand.
 - DELETE - Remove/delete a specific resource by an identifier.
 - POST - Create a new resource. Also a catch-all verb for operations that don't fit into the other categories.
- 

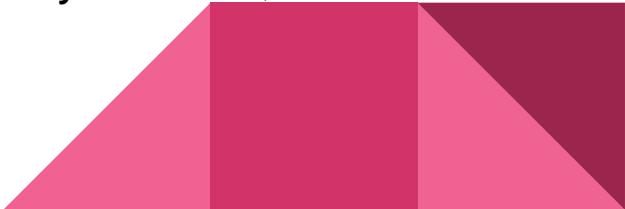
HTTP response codes to indicate status

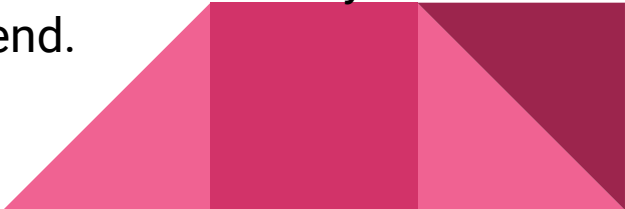
Response status codes are part of the HTTP specification. There are quite a number of them to address the most common situations.

Suggested usages for the "Top 10" HTTP Response Status Codes are as follows:

1. **200 OK** - General success status code. This is the most common code. Used to indicate success.
2. **201 CREATED** - Successful creation occurred (via either POST or PUT). Set the Location header to contain a link to the newly-created resource (on POST). Response body content may or may not be present.



3. **204 NO CONTENT** - Indicates success but nothing is in the response body, often used for DELETE and PUT operations.
 4. **400 BAD REQUEST** - General error for when fulfilling the request would cause an invalid state. Domain validation errors, missing data, etc. are some examples.
 5. **401 UNAUTHORIZED** - Error code response for missing or invalid authentication token.
 6. **403 FORBIDDEN** - Error code for when the user is not authorized to perform the operation or the resource is unavailable for some reason (e.g. time constraints, etc.).
 7. **404 NOT FOUND** - Used when the requested resource is not found, whether it doesn't exist or if there was a 401 or 403 that, for security reasons, the service wants to mask.
- 

8. **405 METHOD NOT ALLOWED** - Used to indicate that the requested URL exists, but the requested HTTP method is not applicable. For example, POST /users/12345 where the API doesn't support creation of resources this way (with a provided ID). The Allow HTTP header must be set when returning a 405 to indicate the HTTP methods that are supported. In the previous case, the header would look like "Allow: GET, PUT, DELETE"
 9. **409 CONFLICT** - Whenever a resource conflict would be caused by fulfilling the request. Duplicate entries, such as trying to create two customers with the same information, and deleting root objects when cascade-delete is not supported are a couple of examples.
 10. **500 INTERNAL SERVER ERROR** - Never return this intentionally. The general catch-all error when the server-side throws an exception. Use this only for errors that the consumer cannot address from their end.
- 

API usage sample

www.boredapi.com/api/activity/

The screenshot shows a REST client interface with a GET request to `http://www.boredapi.com/api/activity/`. The response status is 200 OK with a time of 527 ms. The response body is displayed in JSON format, showing an activity object with fields: activity, accessibility, type, participants, price, and key.

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (10) Test Results Status: 200 OK Time: 527 ms

Pretty Raw Preview JSON

```
1 {  
2   "activity": "Learn origami",  
3   "accessibility": 0.3,  
4   "type": "diy",  
5   "participants": 1,  
6   "price": 0.2,  
7   "key": "8394738"  
8 }
```

Authorization

Generally most APIs use some sort of authorization through api-keys.

API key is a symbolic string that defines that specific user is using the API.

There are many different API authorization mechanisms:

OAuth, Token, etc.



Conclusion

Datasets

1. <https://www.kaggle.com/>
2. <https://www.reddit.com/r/bigquery/wiki/datasets>
3. <https://www.springboard.com/blog/free-public-data-sets-data-science-project/>
4. <https://www.tableau.com/learn/articles/free-public-data-sets>
5. <https://data.egov.kz/>

Web scraping

1. <https://github.com/justmarkham/trump-lies>
2. <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

API

1. <https://www.upwork.com/hiring/development/intro-to-apis-what-is-an-api/>

Assignment 1 Deadline 31.01.21 23:59

1. Collect data or find open dataset in any kind of field which contains **at least 2K samples** (*by using Web scraping/API/open datasets*)
2. Prepare detailed report on collected data (what kind of data, what does the data mean, for what we can use it)
3. Attach code if you collect by using API and Web scraping and file (csv, tsv, xlsx)
4. If you collect data from open dataset attach file (csv, tsv, xlsx) and source link
5. Send me report in pdf, csv/tsv/xlsx file, code*

