Web Scraping and Data Preprocessing

ICDSS - William Profit (wtp18)

Overview

Web Scraping

Data Visualisation

Data Balancing & Preprocessing

Why?

ML needs data => use programs to gather data from the internet

The more data the better

Want best quality of data

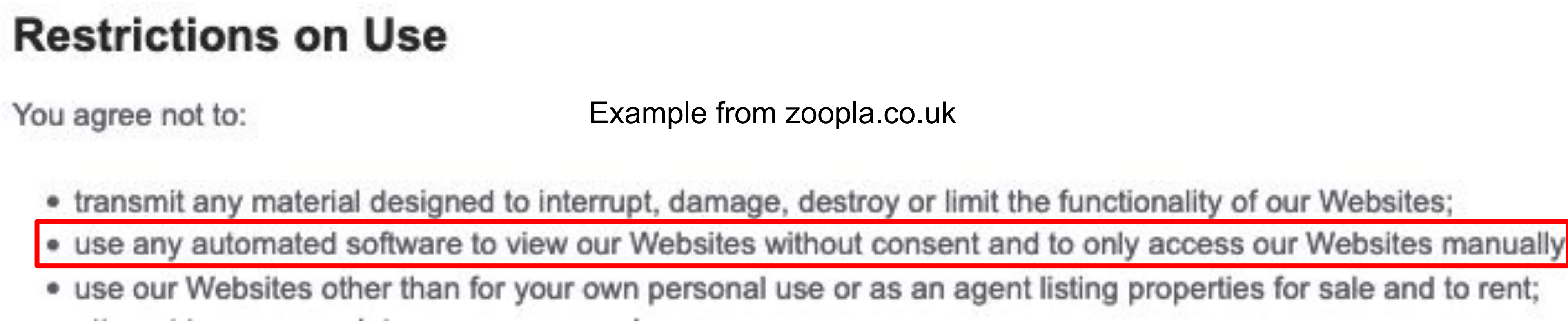
Format is important (preprocessing)

-> Web scraping can be difficult because it is very website-dependent. This lecture will act as a general overview.

DISCLAIMER

Could be illegal, check regulations and terms of use

If you mess up, you're responsible (don't sue us pls)



Restrictions on Use

You agree not to:

- transmit any material designed to interrupt, damage, destroy or limit the functionality of our Websites;
- use any automated software to view our Websites without consent and to only access our Websites manually
- use our Websites other than for your own personal use or as an agent listing properties for sale and to rent;

Example from zoopla.co.uk

Use with Respect

Do not abuse or overload websites

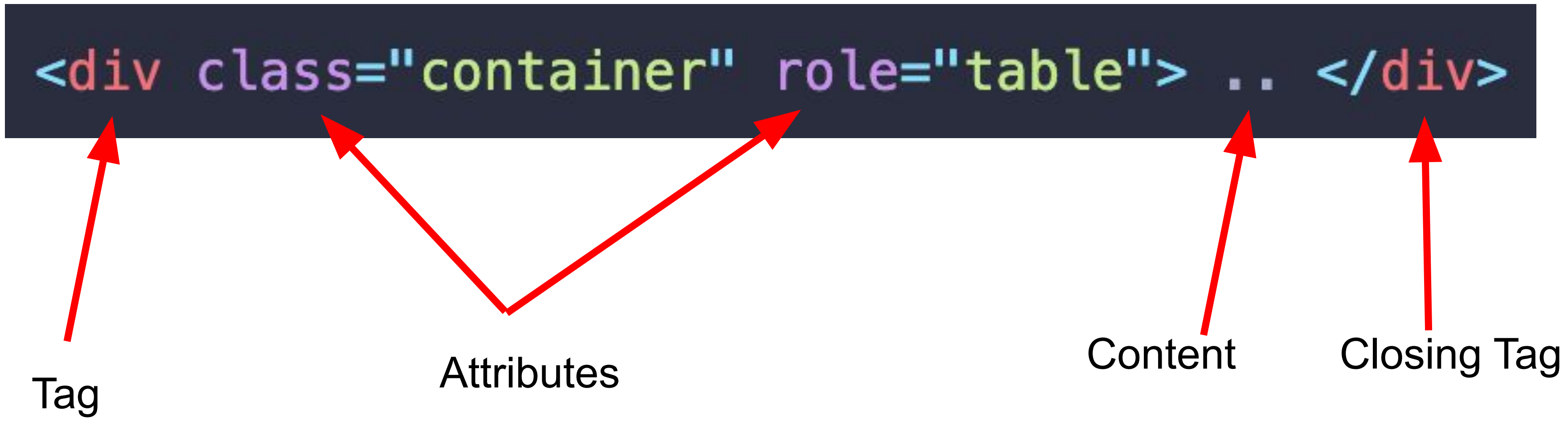
You might get IP blocked if making too many requests

Example for today: web scraping weather data

We’ll be scraping data in the hope of creating a model predicting temperature given multiple features

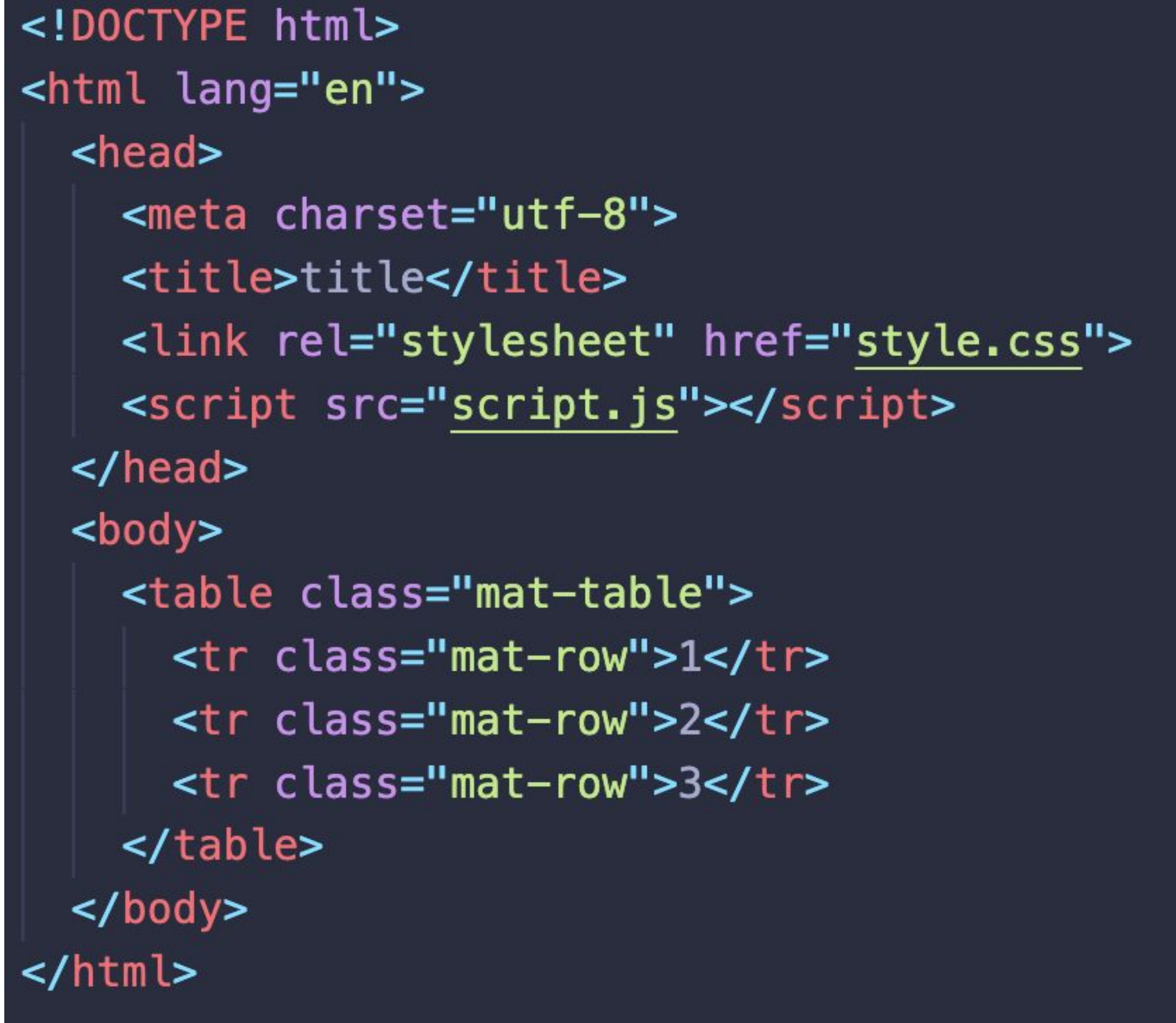
Web pages - HTML

Hyper Text Markup Language



```
<div class="container" role="table"> .. </div>
```

Web page structure



```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>title</title>
    <link rel="stylesheet" href="style.css">
    <script src="script.js"></script>
  </head>
  <body>
    <table class="mat-table">
      <tr class="mat-row">1</tr>
      <tr class="mat-row">2</tr>
      <tr class="mat-row">3</tr>
    </table>
  </body>
</html>
```

Web protocol - HTTP

Hyper Text Transfer Protocol

Protocol used to transmit HTML pages and content

-> Set of rules and agreements that everyone follows when dealing with HTML pages over the network

HTTP - Request methods

- GET : request data i.e. page, image
- POST : submit a form i.e. login form
- PUT : modify data i.e. update account information
- DELETE : delete data i.e. delete a post

=> We will only be using GET

HTTP - Status codes

- 1xx : information i.e. keeping connection alive
- 2xx : success!
- 3xx : redirection
- 4xx : client error (you messed up) i.e. passing invalid info
- 5xx : server error (I messed up) i.e. you crashed their server

Web Scraping steps

- Find a website with data you want
- Look at structure
- Identify fields you want to extract
- Write script
- Get the data

Demo time!

Let’s web scrape some weather data from

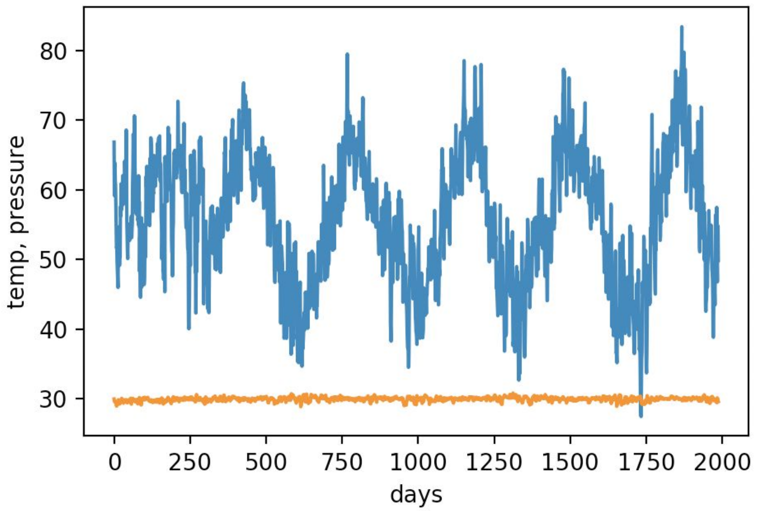
https://williamprofit.github.io/ICDSS-Lecture-Webscraping/

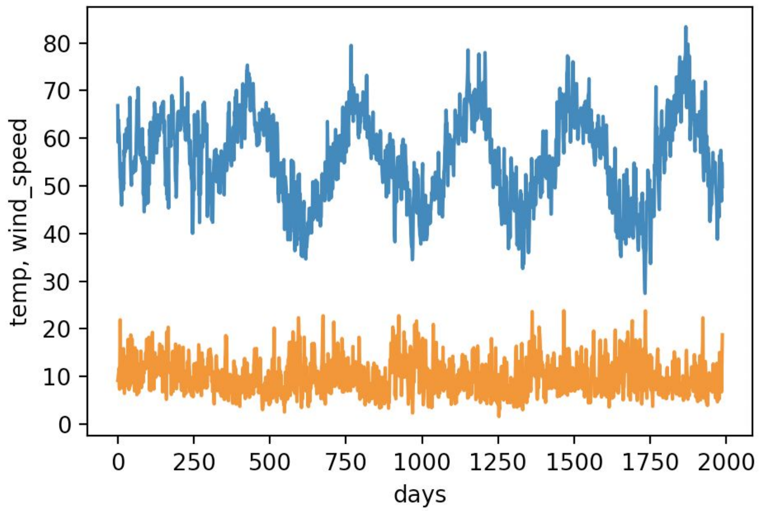
Visualising our data

- Plot features against predicted variable
- Compute correlation coefficients

$$cov(x, y) = \frac{\sum_{i=1}^{n} (x_i - mean(x))(y_i - mean(y))}{n - 1}$$

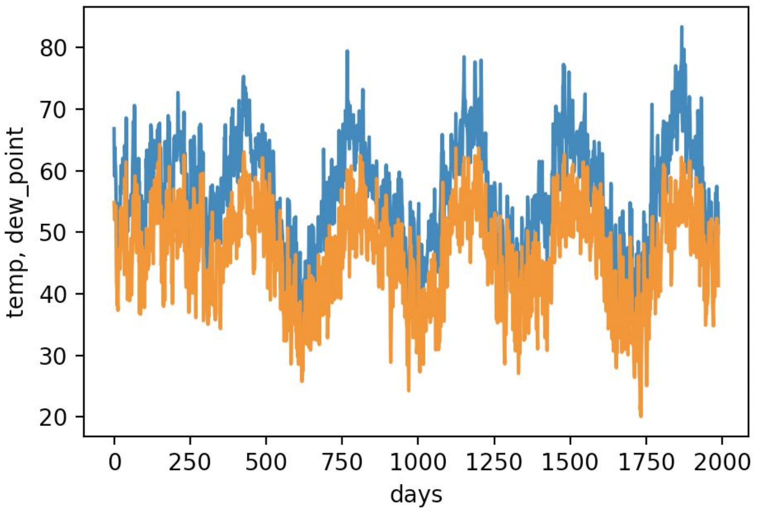
0.07

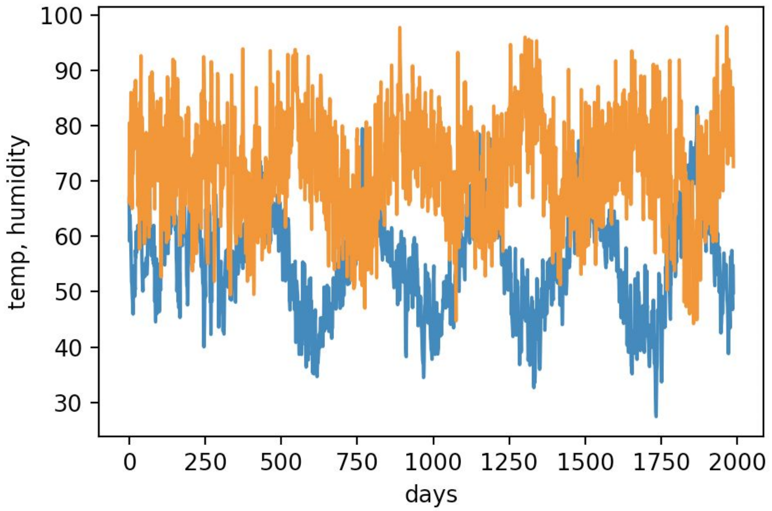




-1.45

66.87





-36.69

Balancing

Example: predicting rain, but data contains 70% rainy days and 30% clear

If some class is over-represented, a few solutions

- Trim the over-represented class
- Augment the other classes
- Get more data

Preprocessing - Normalisation

Have all features range from 0 to 1

-> Makes training easier

-> Less biased by original range

$$X_{normalised} = \frac{X - min(X)}{max(X) - min(X)}$$

Preprocessing - One Hot Encoding

Transform categorical data into vectors with all 0s and a single 1

Example: **weather = {sunny, cloudy, raining}**,

encoded as **sunny=0, cloudy=1, raining=2**

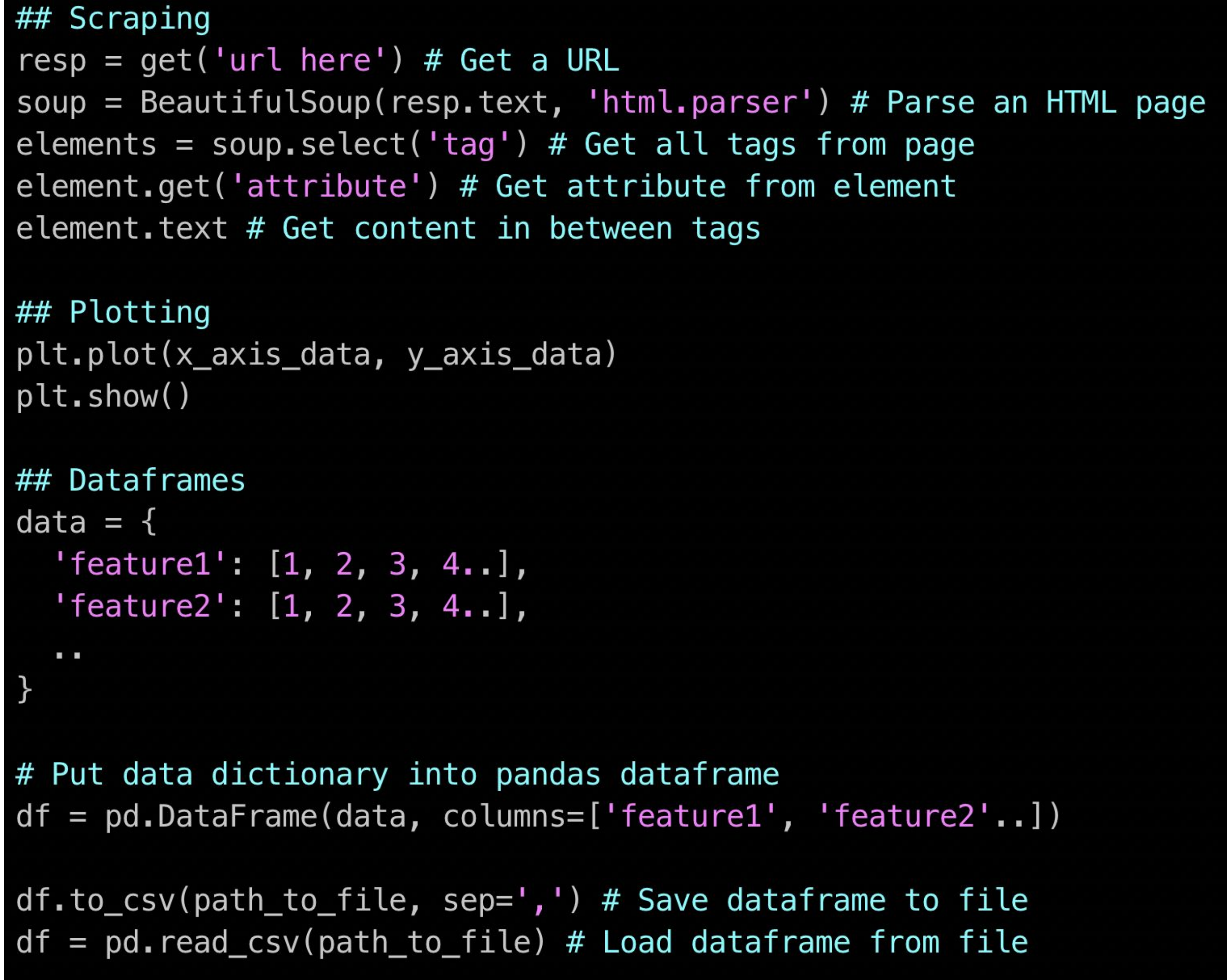
But this gives **raining (=2) > sunny (=0)** ! We are creating a bias..

One hot encoding gives:

- sunny = (1, 0, 0)
- cloudy = (0, 1, 0)
- raining = (0, 0, 1)

Let’s predict some temperatures!

Cheatsheet



```
## Scraping
resp = get('url here') # Get a URL
soup = BeautifulSoup(resp.text, 'html.parser') # Parse an HTML page
elements = soup.select('tag') # Get all tags from page
element.get('attribute') # Get attribute from element
element.text # Get content in between tags

## Plotting
plt.plot(x_axis_data, y_axis_data)
plt.show()

## Dataframes
data = {
  'feature1': [1, 2, 3, 4..],
  'feature2': [1, 2, 3, 4..],
  ..
}

# Put data dictionary into pandas dataframe
df = pd.DataFrame(data, columns=['feature1', 'feature2'..])

df.to_csv(path_to_file, sep=',') # Save dataframe to file
df = pd.read_csv(path_to_file) # Load dataframe from file
```