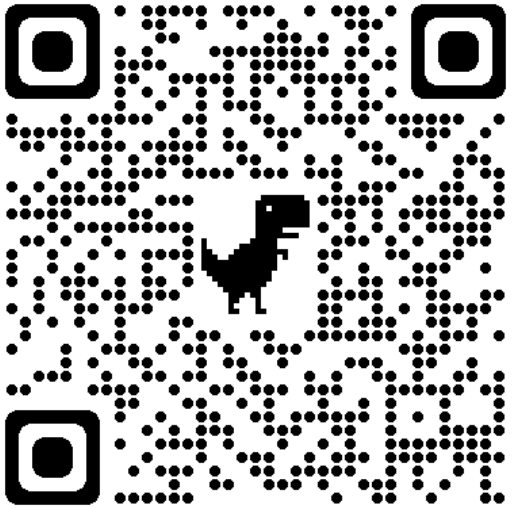
Having Fun with Python – the best libraries for EDA and personal portfolios

Ned Stratton @
PyData London
– Jan 2023

ACTION
BUSINESS INTELLIGENCE

Ned Stratton – Action BI Ltd

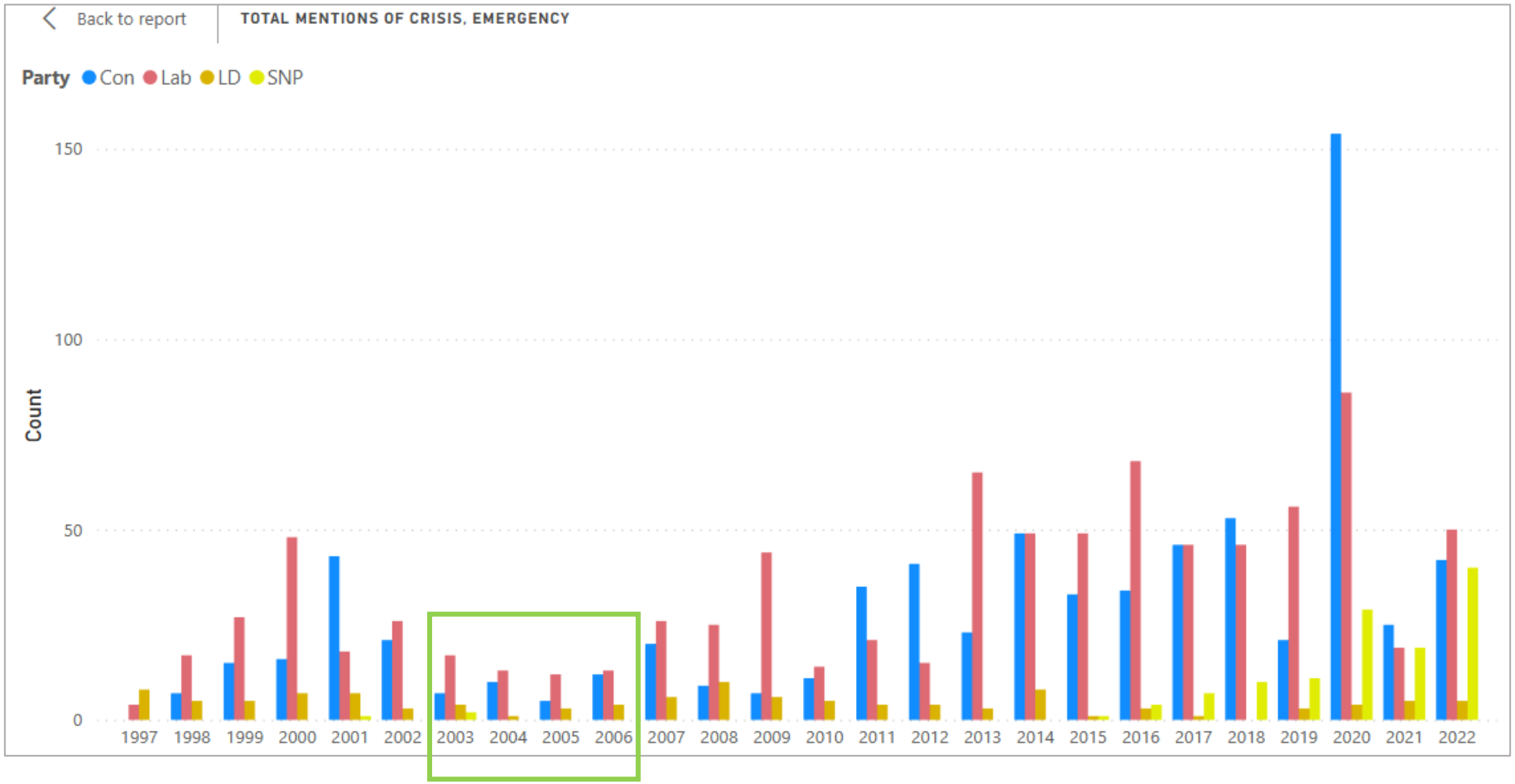
Data Analyst & Rehabilitated Humanities Graduate



ucovi-data.com

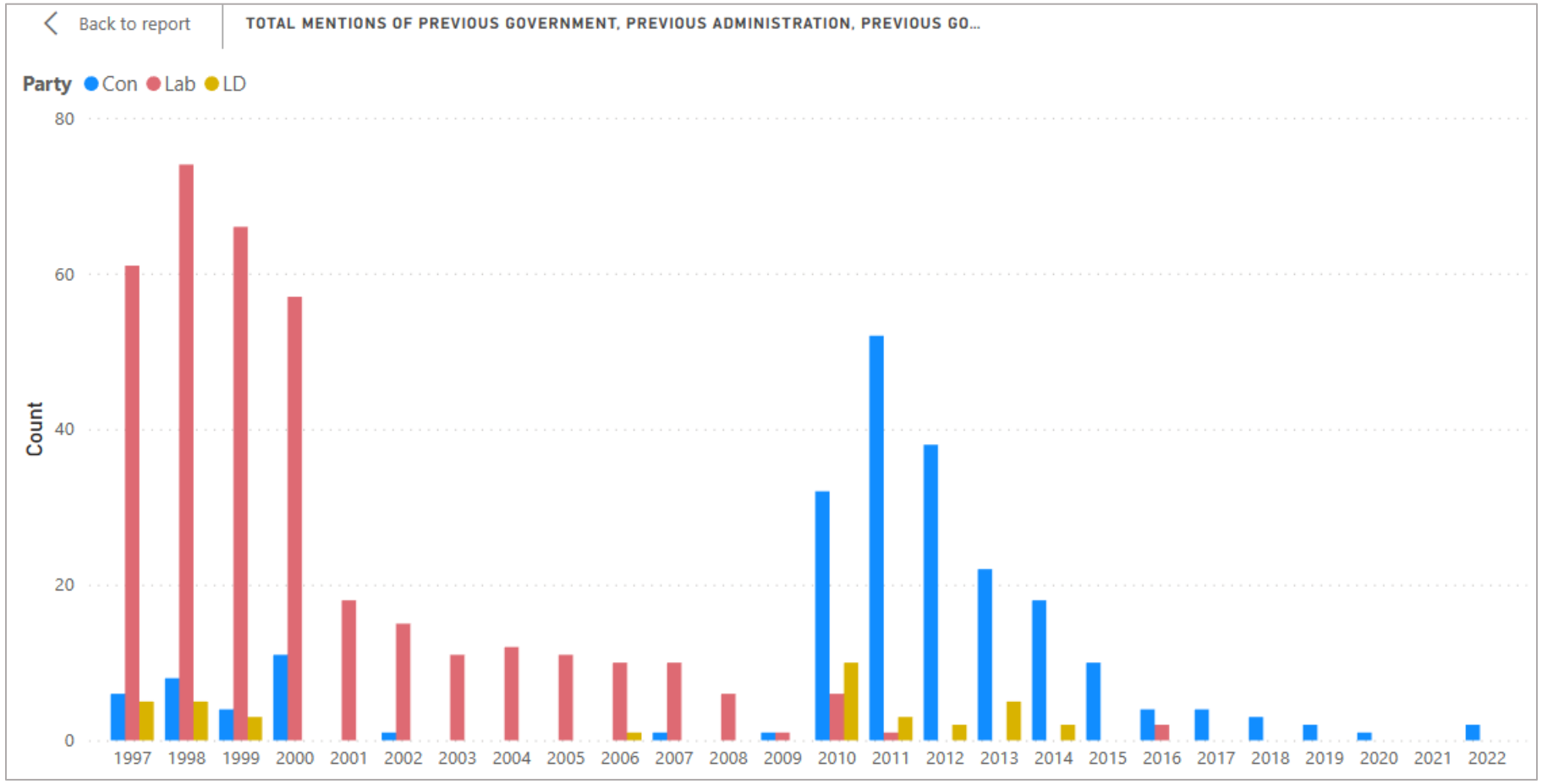
Power BI | SQL | Python | PowerShell | Excel

When was UK politics last not in a crisis?



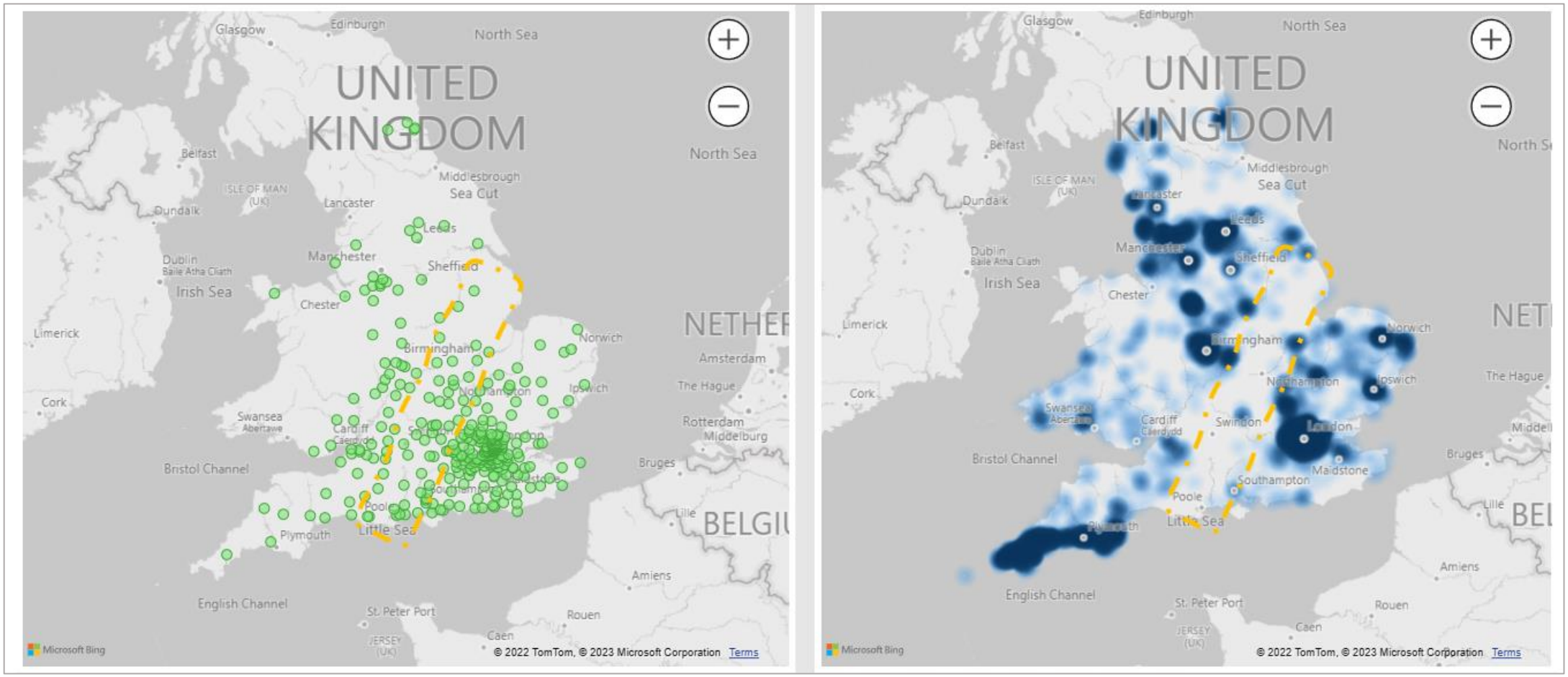
ucovi-data.com/PMQs

How long do politicians blame the previous government for?

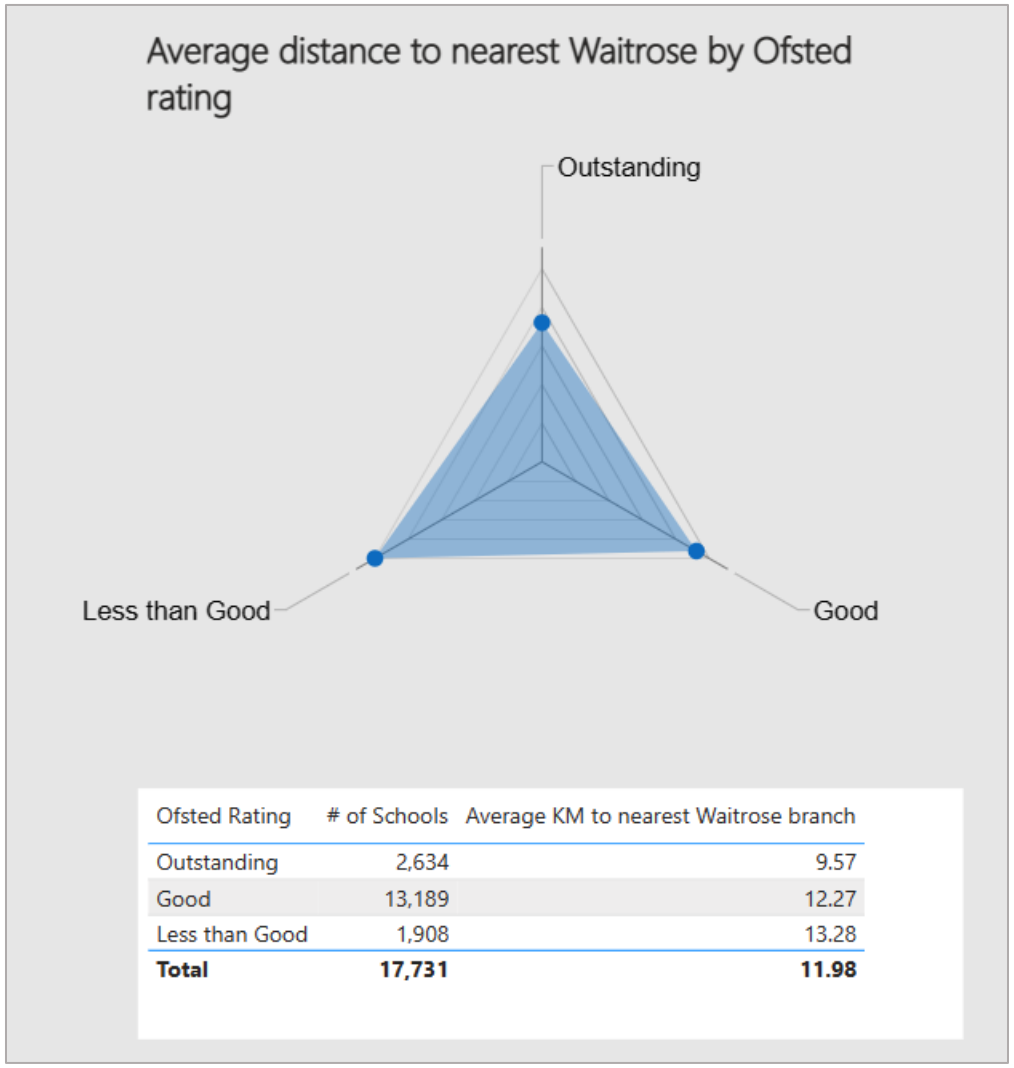


ucovi-data.com/PMQs

Is Waitrose a positive indicator for car accidents and outstanding schools?



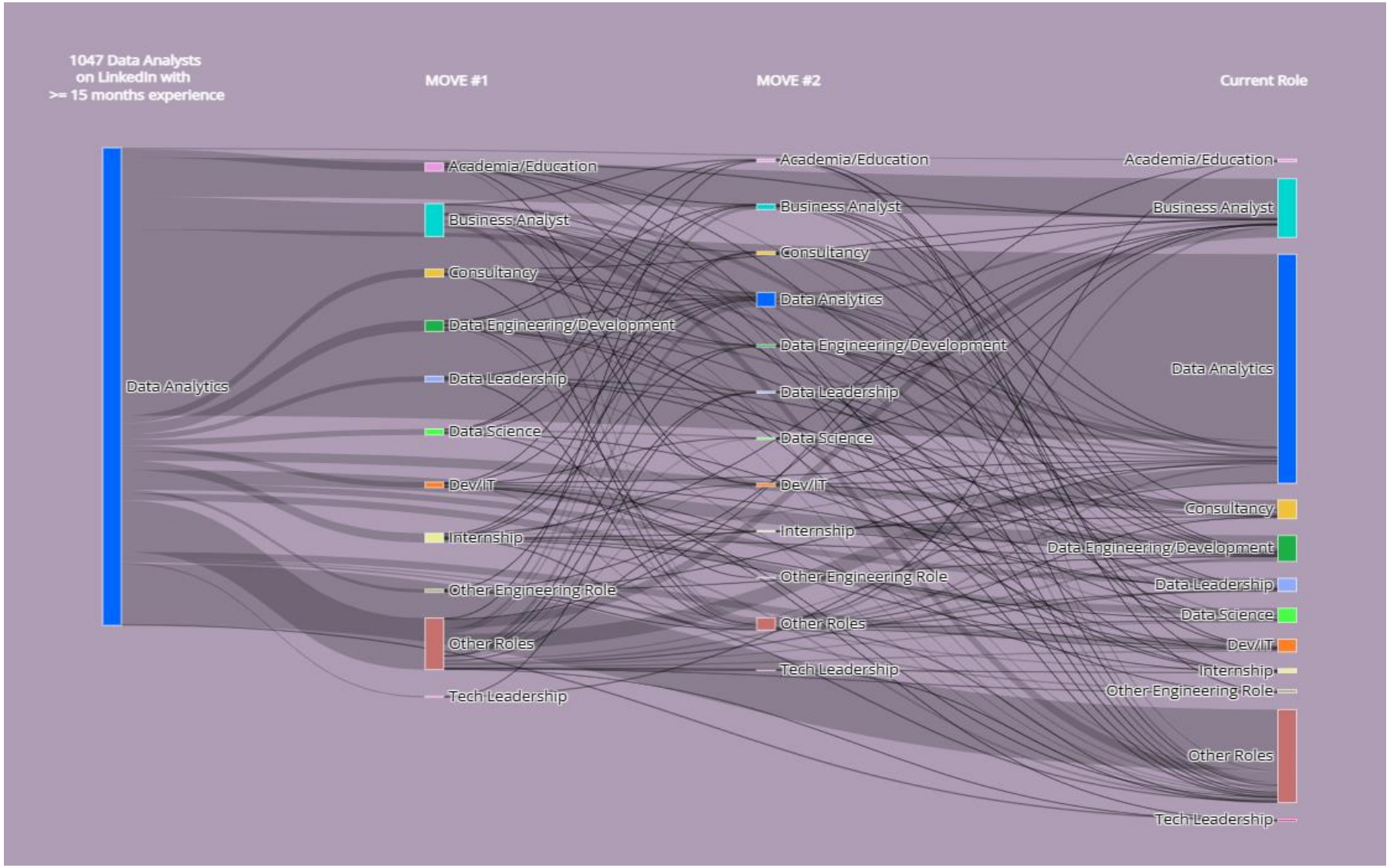
ucovi-data.com/WaitroseIndex



Ofsted Rating	# of Schools	Average KM to nearest Waitrose branch
Outstanding	2,634	9.57
Good	13,189	12.27
Less than Good	1,908	13.28
Total	**17,731**	**11.98**

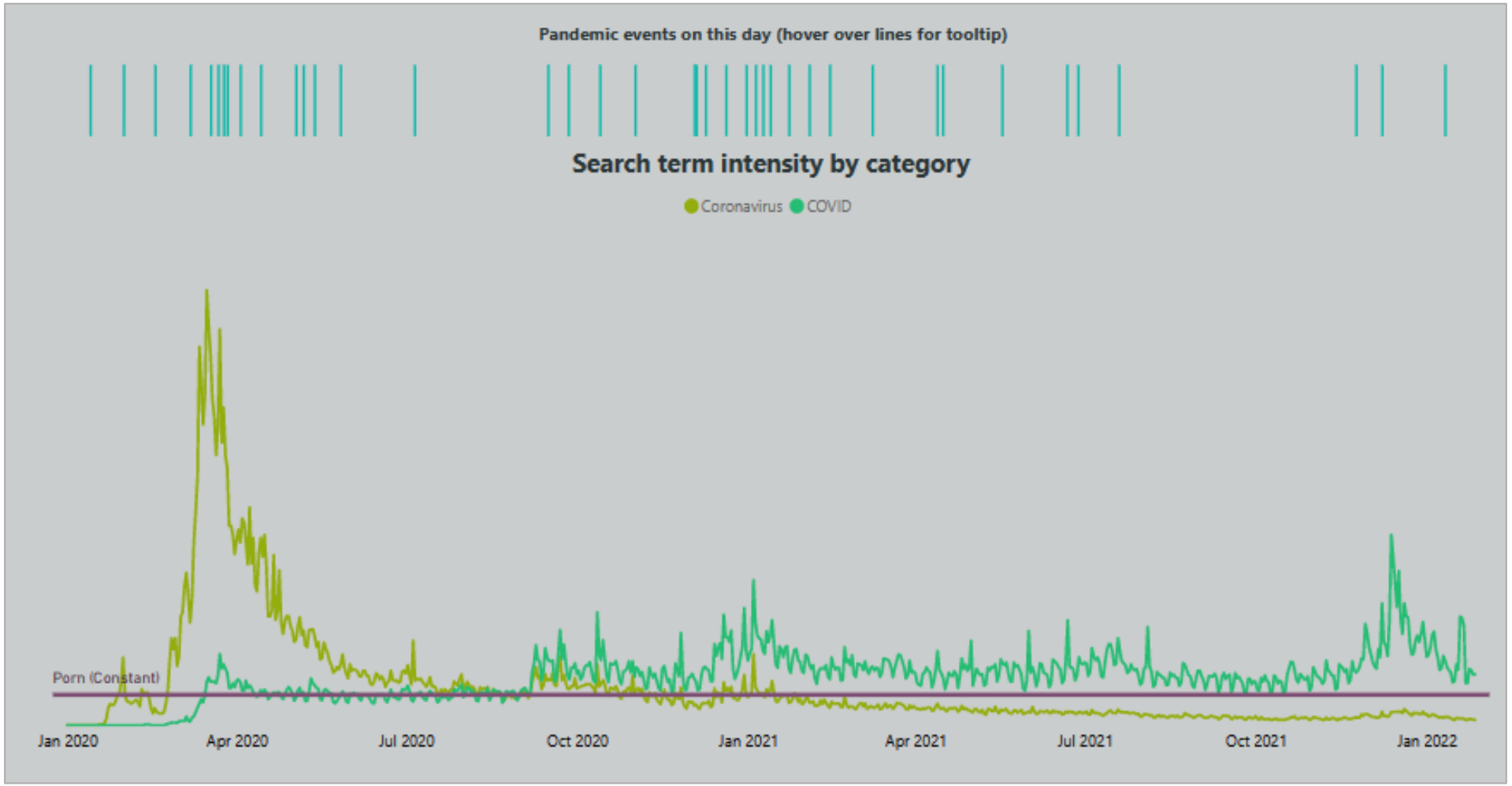
ucovi-data.com/WaitroseIndex

Is data analytics just a feeder role for data science, software dev and IT?



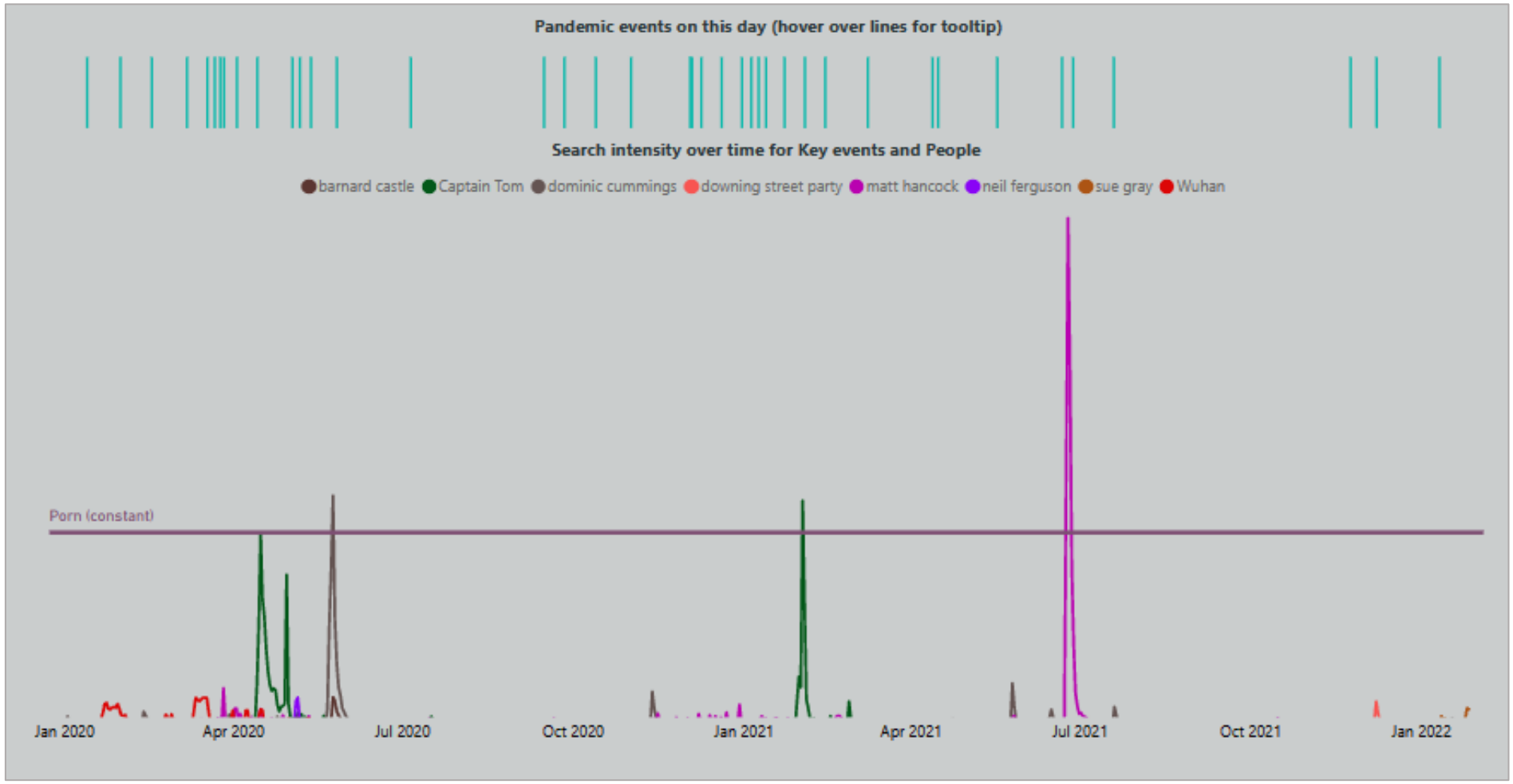
ucovi-data.com/DataAnalysts

When did we stop saying Coronavirus and start saying COVID?



ucovi-data.com/CoronaVersusCovid

Who was the biggest pandemic villain: Dominic Cummings or Matt Hancock?

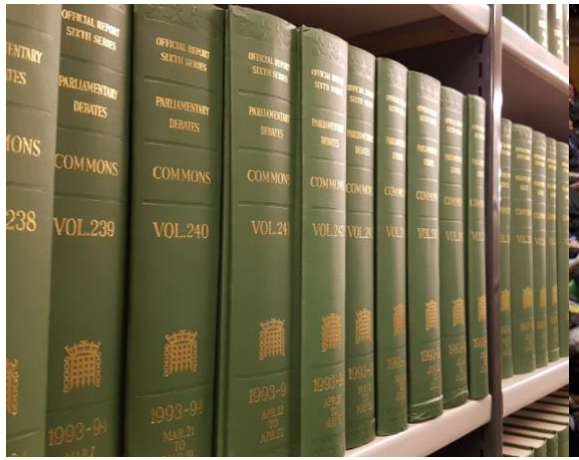


ucovi-data.com/CoronaVersusCovid

LinkedIn

Google Trends

Waitrose



Text Analysis: NLTK | re

Web Scraping: selenium | bs4 | requests

Geo Distance Calculation: geopy

APIs: requests | pytrends (relies on pandas)

File Reading and Management: csv | **open**

PMQs Part 1 - APIs

```
#FINAL

debateq = "https://hansard-api.parliament.uk/debates/debate/{0}.json"
```

```
#FINAL

#toscrape is a list of lists, with each inner list being a debate ID
appendlist = []
for t in toscrape:
    if len(t[0]) == 0:
        print('No debates')
    elif len(t[0]) >= 2:
        print('More than 1 debate with Questions in title - investigate manually')
    else:
        datx = datetime.strptime(t[1],'%d/%m/%Y')
        datf = datetime.strftime(datx,"%Y-%m-%d")
        debx = requests.get(debateq.format(t[0][0]))
        debj = json.loads(debx.text)
        appendlist.append([t[1],debj])
```

PMQs Part 2 – Regexes to clean API data

(Removing web tags and other non-natural language from transcripts of PMQs debates speeches on Hansard.com)

```
def cleanup(x):

    cleanups_log = []
    straight_with_space = ['\n','\t','\r','(LD):','(Con):','(Lab):','(Ind):','(Lab/Co-op):','\x96',
                           '\x97','\u2060','\u07de','\u2004','\xa0','\u2002','\u2003','\u2014']
    for s in straight_with_space:
        x = x.replace(s,' ')
    straight_w_custom = [['\u1d9c','c'],
                         ['\u011f','g'],['\u0177','y'],['\u0175','w'],['\x93','"'],['\u201c','"'],['\u201d','"'],
                         ['\u2019',"'"],['\x92',"'"],['\x94','"'],['\u0103','a']]
    for s in straight_w_custom:
        x = x.replace(s[0],s[1])
        cleanups_log.append([s[0],s[1]])

    #regex replacements

        # mps names in brackets
    membreg = ['Member for [A-Za-z\s,\-]{1,40}\([A-Z][A-Za-z]{1,20} [A-Z][A-Za-z]{1,20}\)',
               'Member for [A-Za-z\s,\-]{1,40}\([A-Z][A-Za-z]{1,14} [A-Z][A-Za-z]{1,15} [A-Z][A-Za-z]{1,15}\)']
    xmembreg = '|'.join(membreg)
    if re.search(xmembreg,x):
        mbmr = re.finditer(xmembreg,x)

        for d in mbmr:
            new_d = re.sub('\s\([A-Za-z\s]{1,}\)$','',d[0])
            x = x.replace(d[0],new_d)
            cleanups_log.append([d[0],new_d])

    complexr = ['<span ','data-house-id="[a-zA-Z0-9_\-]{1,}"',
                'data-volume-number="[a-zA-Z0-9_\-]{1,}"',
                'id="[a-zA-Z0-9_\-]{1,}"','class="[a-zA-Z0-9\-]{1,}"',
     'data-column-number="[a-zA-Z0-9\-]{1,}"','</span>',
                '</Question>','</QuestionText>','Q[1-9]{1,2}\. ',
                '<Question HRSContentId="\{[A-Z0-9\-]{1,}\}">','<QuestionText HRSContentId="\{[A-Z0-9\-]{1,}\}">',
                '<em>Interruption.</em>','<em>Interruption</em>','\[Interruption\]','\[Interruption\.\]',
               '<em>','</em>',r'\[Hon\. Members: "[\W\w]{1,28}"\]','\[[A-Za-z0-9\.;\s,\-]{1,65}\]','\[R\] '
                , #for training - remember that the first version didn't have a '-'
                '\[','\]']

    for cm in complexr:
        x = re.sub(cm,' ',x) #repeated space to space
        cleanups_log.append([cm,' '])

    x = re.sub('\s\s+',' ',x) #repeated space to space
    x = re.sub('^\s+','',x) #leading space
    x = re.sub('^Q[0-9]+\.','',x) #leading Q with number for the questions
    x = re.sub(' \([0-9]+\)\s*','',x) #question entries brackets
    x = re.sub(' \[[0-9]+\]\s*','',x) #question entries square brackets
    x = re.sub('\s+$','',x) #trailing space
    x = re.sub('[<>]','',x) #triangular brackets
    x = re.sub('^[0-9]{1,2}\.','',x) #leading number at start of question with no Q
    return x,cleanups_log
```

PMQs Part 3 – N-Grams and Lemmatization

Stemming: "Criticized" > "Critic"

Lemmatization: "Criticized" > "Criticize"

N-grams (2): "The dog barked" > "The dog" | "dog barked"

```python
import csv
from nltk import word_tokenize, ngrams,WordNetLemmatizer,PorterStemmer,sent_tokenize # had to download punkt
from nltk.corpus import wordnet as wordn #show if the word is an adjective, noun, verb or adverb
from collections import Counter
from nltk.corpus import stopwords
from nltk.sentiment import SentimentIntensityAnalyzer
import pandas as pd
import re
import os
import datetime
from textblob import TextBlob
import statistics
english_sw = stopwords.words('english') # same as comment above for stopwords corpus
```

```python
lemmatizer = WordNetLemmatizer()
ps = PorterStemmer()
```

```python
def smart_lemma(x):
    try:
        wordtype = wordn.synsets(x)[0].pos() # get the first pos tag of the word. 'a' and 's' are adjectives,'v'=verb
        if wordtype == 'n':
            out = lemmatizer.lemmatize(x) #noun is default for lemmatizer. THis should standardise plurals
        elif wordtype == 'v':
            out = lemmatizer.lemmatize(x, pos='v') #standardize verbs
        else:
            #leave alone
            out = x
    except:
        out = x
    return out

def ngramlist(x,y,swmax,bgswitch=False):
    #return list of distinct ngrams with counts, where x is a list of speeches, y is n grams, and swmax is the limit.
    #... to the number of REDACTED stopwords that can appear in the gram for it to be counted and returned.
    #bgswitch tells the function that it is running on bigrams - so the ngrams_useful call should factor this in and
    #...both words in bigrams against the extendend stopwords
    gramslist = []
    for spch in x:

        if y > 1:
            toks = tokensprep(list(word_tokenize(spch)))
            rawgrams = [list(b) for b in list(ngrams(toks,y))]
            cleangrams = [re.sub('\.$','',' ".join(c).lower()) for c in rawgrams if ngrams_useful(c,swmax,bgswitch)]
            gramslist.extend(cleangrams)
        else:
            toks = tokensprep(list(word_tokenize(spch.replace('…',' '))))
            cleangrams = [re.sub('\.$','',tko.lower()) for tko in toks if not re.search('[^a-z]',
                                tko.lower()) and tko.lower() not in english_sw]
            if len(cleangrams) > 0:
                cleangramslemma = [smart_lemma(cl) for cl in cleangrams]
                gramslist.extend(cleangramslemma)

    ngram_cts = Counter(gramslist)
    return [[i[0],i[1]] for i in ngram_cts.items()]
```

Waitrose Branches 1 – Simple Web Scraping

```
import requests
from bs4 import BeautifulSoup
import time
import re
waitstart = 'https://www.waitrose.com/content/waitrose/en/bf_home/bf/616.html' #Clapham Common for demo

reqstart = requests.get(waitstart).text
optionsoup = BeautifulSoup(reqstart,'lxml').find('div',class_="branch-finder-form")

options = []
for o in optionsoup.find_all('option'):

    if not o['value'] == "":
        options.append([o['value'],o.text])


templatelink = 'https://www.waitrose.com/content/waitrose/en/bf_home/bf/{0}.html'

for o in options:
    pcodes = []
    storelink = templatelink.format(o[0])
    storereq = requests.get(storelink).text
    storesoup = BeautifulSoup(storereq,'lxml').find('div',class_="col branch-details").find_all('p')
    for ss in storesoup:
        pc = parse_postcode(ss.text) #PARSE POSTCODE is a SIMPLE REGEX func to extract a UK postcode
        if pc:
            pcodes.append(pc)
    o.extend(pcodes)
    time.sleep(0.5)

```

Waitrose Branches 2 – Distance Calculation from Lat/Long Coordinate pairs

```python
import geopy.distance
def distance_km (waitr,sch):
    return geopy.distance.geodesic(waitr, sch).km
```

```python
for sc in schools_ll: # a csv file of UK schools with Long/lat coordinates read into a list of lists

    sccords = (float(sc[11]),float(sc[12]))
    distances = []
    for wt in waitrose_ll: #waitrose branches with long/lat coords
        sname = wt[1]
        spc = wt[2]
        coords = (float(wt[6]),float(wt[7]))
        dista = round(distance_km(sccords,coords),3)
        distances.append([sname,spc,dista])
    nearest = sorted(distances,key=lambda k: k[2])[0]
    sc.extend(nearest)
```

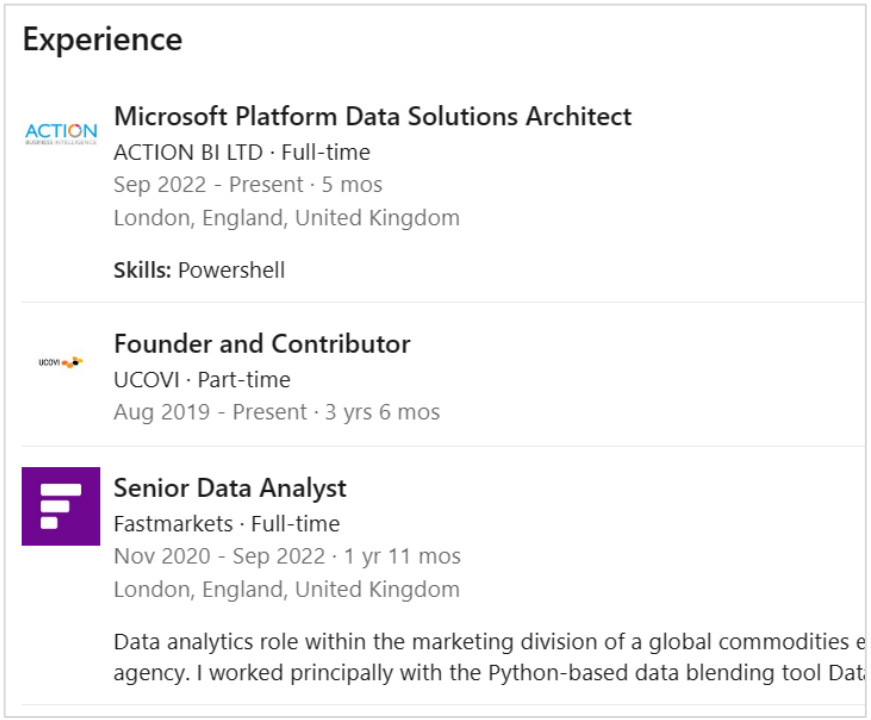
UK Postcodes & Lats/Longs: https://geoportal.statistics.gov.uk

UK Schools Data (has postcode, lat and long for each school: https://www.get-information-schools.service.gov.uk/Downloads)

LinkedIn Data Analysts – Advanced Scraping

[LinkedIn Group Members Export | PhantomBuster]

https://phantombuster.com/automations/linkedin/2852/linkedin-group-members-export



LinkedIn Data Analysts – Advanced Scraping

```python
from time import sleep
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.proxy import Proxy, ProxyType
from selenium.webdriver.common.by import By
#https://medium.com/nerd-for-tech/linked-in-web-scraper-using

#https://www.us-proxy.org/
from bs4 import BeautifulSoup as bs
```

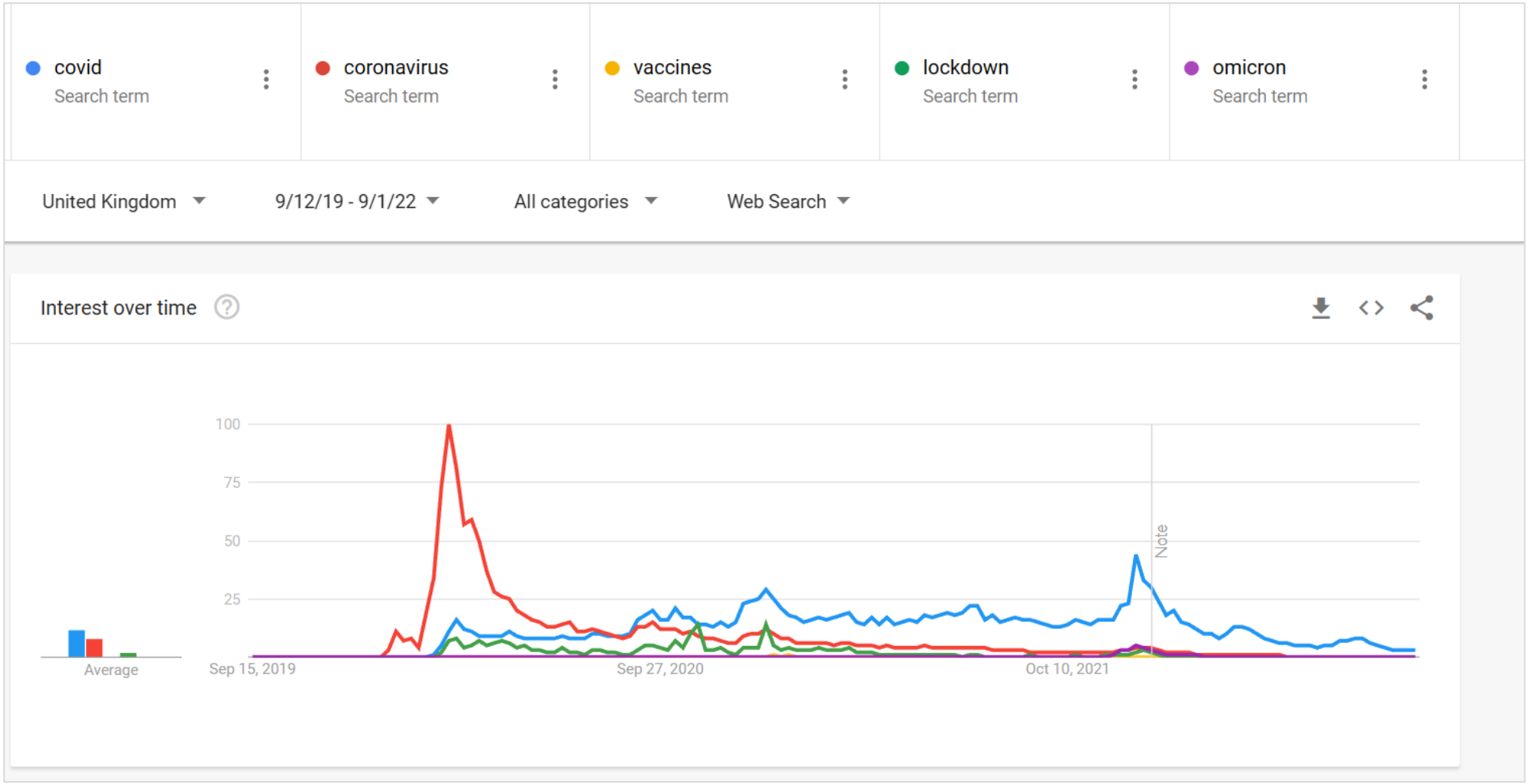
```python
pth = 'C:\\Users\\nedst\\chromedriver.exe'

randsleeper = {1:11.2,2:13.7,3:14.5,4:12.3,
               5:14.9,6:13.1,7:11.8,8:15.3,
               9:15.7,10:14,11:16.7,12:32,13:19,14:21,15:29,16:18.9,17:25,
              18:21,19:23,20:15.7,21:14.4,22:22.4,23:16.3,24:8.8,25:20.1}
chrome = webdriver.Chrome(pth)
chrome.get("https://www.linkedin.com/login")
sleep(2)
chrome.find_element_by_id('username').send_keys(email)
chrome.find_element_by_id('password').send_keys(password)
chrome.find_element_by_id('password').send_keys(Keys.RETURN)
sleep(10)

for no in notdone[:50]:
    chrome.get(gurl + no)
    sleep(1)
    try:
        css_sel = 'section.artdeco-card.ember-view.break-words.pb3'
        experience = [x for x in bs(chrome.page_source,
                                    'lxml').select(css_sel) if x.find('div',{'id':'experience'})]
        # css selector for when you need to select an element with a space-separated class
        career = []
        #classes
        jtitle = 'mr1 t-bold'
        timeframe = 't-14 t-normal t-black--light'
        jtitlei = "mr1 hoverable-link-text t-bold"
        newcss = 'div.display-flex.flex-row.justify-space-between'
        #newcss = 'div.display-flex.flex-column.full-width'
        jobslist = experience[0].find('div',{'class':'pvs-list__outer-container'}).select(newcss)
        jobslist = [x for x in jobslist if not x.find('a',{'data-field':'experience_company_logo'})]
        for j in jobslist:
            templ = []
            for sp in j.find_all('span'):
                if 'class' in sp.attrs:
                    if ' '.join(sp['class']) in [jtitle,timeframe,jtitlei]:
                        txtsp = sp.find('span',{'aria-hidden':'true'}).text
                        templ.append(txtsp)
            career.append(templ)
    except:
        career = ['ERROR']
    profcareer.append([no,career])
    sleep(1)
    chrome.get("https://www.linkedin.com/feed")
    sleep(randsleeper[random.randint(1,25)])
```

Tech-Ninja…

```python
#AUTOMATED CHROME DRIVER DOWNLOAD/INSTALL
from selenium import webdriver
from webdriver_manager.chrome import ChromeDriverManager
chrome = webdriver.Chrome(ChromeDriverManager().install())

#RUNNING WITHOUT CHROME WINDOW OPENING

options = webdriver.ChromeOptions()
op.add_argument('headless')

#ONLINE PDF DOWNLOAD AND SAVE

options.add_experimental_option('prefs', {
    "download.default_directory":"C:\\Users\\username\\ScriptPDFs\\",
    "download.prompt_for_download":False,
    "download.director_upgrade": True,
    "plugins.always_open_pdf_externally":True
})
```

Google Trends Search Data – Custom API Library

https://trends.google.com/trends



Google Trends Search Data – Custom API Library

https://github.com/GeneralMills/pytrends

```python
# library imports
from pytrends.request import TrendReq
import datetime
import csv
from time import sleep
```

```python
# pytrends object (function depends on this)
pytrends = TrendReq(hl='en-GB', tz=0)
```

```python
#main API call/parse function to get searchterms as % of constant search term within time frames (_dates)
def prepare_data(_const,_srch,_dates):

    outlist = []

    def normalize(x,y):
        x = float(x)
        y = float(y)
        return round(x/y,4)

    query_list = _const
    query_list.extend(_srch) #search terms as list with the constant value the first item
    pytrends.build_payload(query_list, cat=0, timeframe=_dates, geo='GB', gprop='')
    querydf = pytrends.interest_over_time()

    for _term in _srch:
        querydf[_term] = querydf.apply(lambda x: normalize(x[_term],x[_const][0]),axis=1)

    querydf.drop('isPartial',axis=1,inplace=True)
    querydf.drop(_const[0],axis=1,inplace=True)
    querydict = querydf.to_dict()

    for i in querydict:
        for _date in querydict[i]:
            d = _date.date()
            outlist.append([i,d,querydict[i][_date]])
    return outlist
```

```python
searchterm_batches = [
        ['Coronavirus','COVID','Wuhan'],
          ['vaccine','variant','furlough'],
          ['long COVID','lockdown','NHS','Captain Tom'],
          ['Tier 1','Tier 2','Tier 3','Tier 4'],
          ['scotch egg','herd immunity','barnard castle'],
          ['matt hancock','neil ferguson','dominic cummings'],
          ['Pfizer','AstraZeneca','Moderna','travel corridor'],
          ['red list','amber list','green list','South African variant'],
          ['Brazilian variant','Indian variant','Delta variant'],
          ['R number','infection rates','COVID symptoms','quarantine'],
          ['self isolation','NHS app','test and trace','5G'],
            ['booster jab','vaccine passport','plan b','omicron'],
        ['downing street party','sue gray']
]


mainlist = []
constant = "ryanair"
daterng = '2021-08-15 2021-08-21' #range of dates to get daily data
for _batch in searchterm_batches:
#for _batch in newbatch: # custom for captain tom

    batchdata = prepare_data([constant],_batch,daterng)
    sleep(4)
    mainlist.append(batchdata)
```

Cross-Project – File Read and Write Using CSV/Open

```
import csv

#WRITING (to append, switch the 'w' in open with 'a' and it will start the writing...
#...from after the last line of the input file var)

_pmqmaster_rawtd = "C:\\***\\***\\OneDrive\\Documents\\****\\***\\***\\***\\***\\Files\\rawcontribs.txt"
with open(_pmqmaster_rawtd,'w',newline='',encoding='utf-8') as fl:
    wrt = csv.writer(fl,delimiter='\t')
    wrt.writerow(['Name','Link','RawText','Date','Role','Party','Name'])

    for cn in contribs:
        if cn[1] is not None:


            cnrow = [cn[2], #unrefined text name
             '',
             '/search/MemberContributions?house=Commons&memberId={0}'.format(cn[1]), # link for mp
             cn[4], #raw text
             cn[0], #date
             mp_data_list[cn[1]][3], #role
             mp_data_list[cn[1]][1], #party
             mp_data_list[cn[1]][0], #name
            ]
            wrt.writerow(cnrow)


#READING

with open(_pmqmaster_rawtd,'r') as flp:
    flp.readline() #skips headers
    rdr = csv.reader(flp,delimiter = '\t')
    for row in rdr:
        pcodes.append(row)
```

Text Analysis: NLTK | re

Web Scraping: selenium | bs4 | requests

Geo Distance Calculation: geopy

APIs: requests | pytrends (relies on pandas)

File Reading and Management: csv | **open**

Creativity with data:

Dataset-driven or idea-driven?

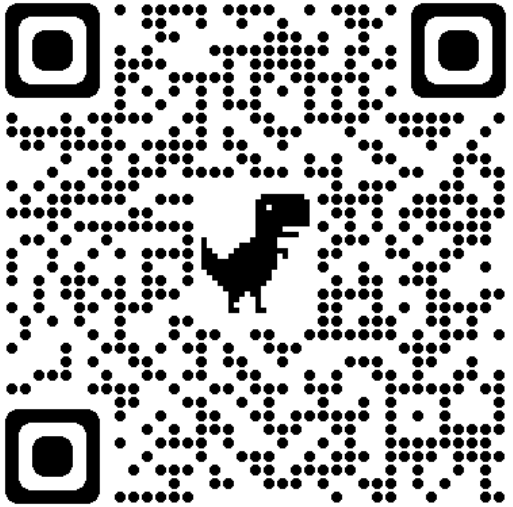
Why have a personal portfolio anyway?

- Learning and development
- 'Keeping your hand in'
- Display of keenness and interest
- Sanity – 'Man is born free and yet is everywhere in chains' (Rousseau)

And why Python?

- Open source
- Powerful, well-supported, and widely-used
- Feeder into PowerShell & JavaScript (if that's your thing)
- OOP & very versatile

Thanks for listening! (Any questions?)



ucovi-data.com

My LinkedIn: https://www.linkedin.com/in/nedstratton1/