MACQUARIE University
SYDNEY·AUSTRALIA

Lessons learned from Data Analysis projects

Brian Ballsun-Stanton
Faculty of Arts
Wednesday 30 October 2019

Table of Contents

MACQUARIE University

Longer title: Lessons learned from Data Analysis projects in Natural Language Processing with Japanese and Security Studies data - Shell Scripts, Jupyter Notebooks, and the value of doctests

MACQUARIE University
SYDNEY·AUSTRALIA

Recent Research

Japanese Advertising Research

- Corpus donated by a Japanese advertising agency. Very simple CSV in Japanese.
- “47 insurance companies with a total segmented and filtered token count, after removing stopwords of 29,486 [tokens].”
- Parsed via MeCAB (Using the mecab-ipadic-neologd dictionary, trained on web content.)

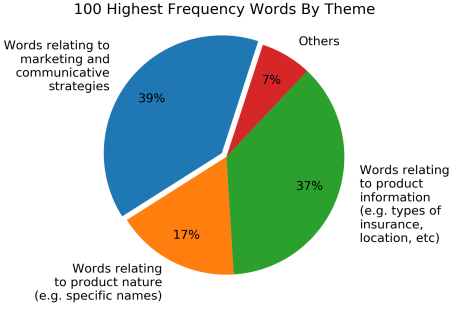


Figure 1: From ‘Securing Life in Trust: Corpus-based Keyword Analysis of Japanese Insurance TV Commercials’ by Svetanant and Ballsun-Stanton. Do not distribute.

MACQUARIE University

- Studying keyword use of local groups.
- Data set from mastodon currently downloading. At 1GB of JSON already.
- Using Twitter Historical Power Track to run a single massive query for $2,000
- My first official “big-data” (It doesn’t run on my laptop)
- Interacting with the raw APIs via python’s ‘requests’ module.

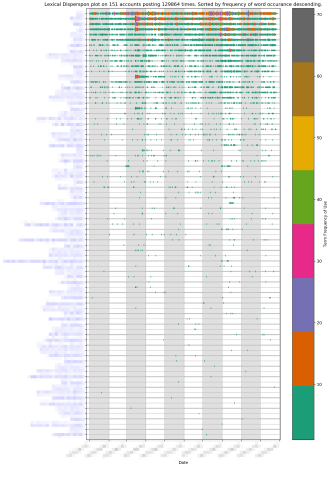
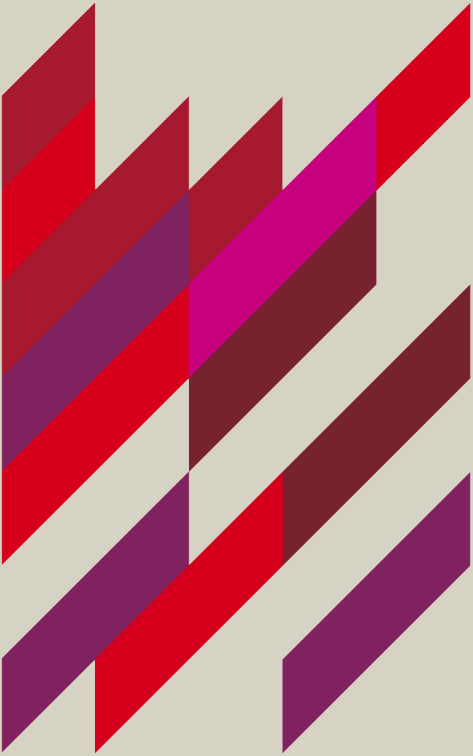


Figure 2: Lexical Dispersion plot by Brian Ballsun-Stanton. Blurred due to sensitive ongoing research.



MACQUARIE
University
SYDNEY·AUSTRALIA

Natural Language Processing

Tools which have produced results

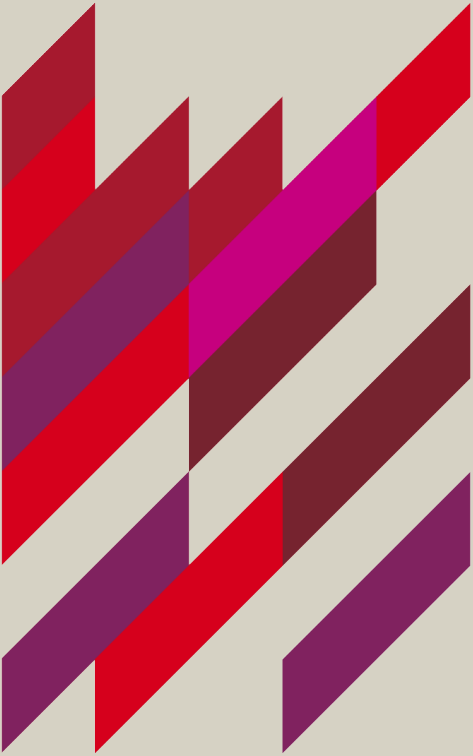
MACQUARIE University

- Python3
- The NLTK book.
- MeCAB for Japanese
- Log-likelihood computation by Rayson and Garside 2000
- Checking dates on stack overflow posts
- Filesender
- Making sure my code is resume-able
- Git
- rsub/rmate
- byobu

Python Modules

MACQUARIE University

- Using spaCy for tokenisation, lemmatisation. NLTK for stemming.
- Matplotlib for a Lexical Dispersion Plot
- tqdm
- dateparser and Delorean
- tldextract
- matplotlib
- sqlitedict
- wordcloud



MACQUARIE
University
SYDNEY·AUSTRALIA

Doctesting

When do I doctest?

MACQUARIE University

Very slim form of unit testing:

“Making sure each function I write does what I want it to do and returns output I want it to return.”

Great tool for catching errors introduced when changing a function, especially when it gives data to other functions.

Forces compartmentalisation and documentation of functions.

Really effective in Jupyter Notebooks to test a cell.

```
"""
This is the "example" module.

The example module supplies one function, factorial().  For example,

>>> factorial(5)
120
"""

def factorial(n):
    """Return the factorial of n, an exact integer >= 0.

    >>> [factorial(n) for n in range(6)]
    [1, 1, 2, 6, 24, 120]
    >>> factorial(30)
    265252859812191058636308480000000
    >>> factorial(-1)
    Traceback (most recent call last):
        ...
    ValueError: n must be >= 0

    Factorials of floats are OK, but the float must be an exact integer:
    >>> factorial(30.1)
    Traceback (most recent call last):
        ...
    ValueError: n must be exact integer
    >>> factorial(30.0)
    265252859812191058636308480000000

    It must also not be ridiculously large:
    >>> factorial(1e100)
    Traceback (most recent call last):
        ...
    OverflowError: n too large
    """
```

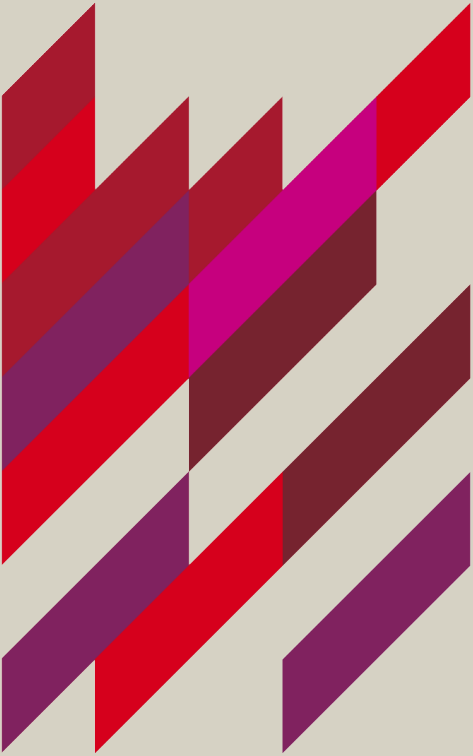
Figure 3: From docs.python.org/3.7/library/doctest.html

Doctesting in Jupyter

MACQUARIE University

In [11]:
```
def extract_domain_count(csvdict):
    """In priorname+"domain_summary.csv"
        Extract all URLS from column labeled "Final Link" if exists, else "Link"
        Cols: domain, Count frequency of domain.


    >>> domains = extract_domain_count(TEST_CSV_DICT)

    >>> domains['theaustralian.com.au']
    118


    """

    links = []
    domain_count = collections.OrderedDict()
    for row in csvdict:

        link = row.get('Final Link', row.get('Link', None))
        if link:
            tld = tldextract.extract(link)
            links.append("{}.{}".format(tld.domain, tld.suffix))
    for link in links:
        domain_count[link] = links.count(link)
    #https://stackoverflow.com/a/613218/263449
    return collections.OrderedDict(sorted(domain_count.items(), key=lambda kv: kv[1], reverse=True))
```

Figure 4: From code I wrote to help a Masters student analyse their social media data.



MACQUARIE
University
SYDNEY·AUSTRALIA

Shell Scripts and Jupyter Notebooks

Different tools for different purposes

MACQUARIE University

- I use Jupyter Notebooks when I’m likely to be sharing the code with researchers.
- See the LIGO black hole mybinder notebook.
- Notebooks are fundamentally narrative and chunked. Tell a story. Encourage Documentation.
- Shell scripts (and python run from the shell) is a ”Just run everything.” Scales better. Less nonsense, more capable. Run and forget.

Thank you!

MACQUARIE University

Source code for this presentation is available at: `https://github.com/Denubis/Lessons-learned-from-Data-Analysis-projects`