Lecture 7: Corpora and Data Formats

LING 1340/2340: Data Science for Linguists

Na-Rae Han

Objectives

- Your term project
  - Plan submitted, repo created!
  - Work on your DATA!
- Corpus data: standard and popular formats
  - Encoding, line break
  - Review of common data formats
- Web and social media mining
  - Twitter data demo

Your term project

- Your project plan -- let's see 'em!
- First progress report is due shortly after
  - Focus on data: sourcing, curation and cleaning
- Managing your data
  - You will be manipulating and processing your data.
  - Should you include your data set in your GitHub repo?
    - Depends!

Data standards & exchange formats

	What	Notes, reference
CSV	Comma-separated values	Compatible with Excel
TSV	Tab-separated values	
HTML	Web pages	Not meant as data format
XML	For markup and text encoding	A Gentle Introduction to XML by TEI
JSON	JavaScript Object Notation (Twitter, Jupyter Notebook)	Introducing JSON JSON example (vs. XML)

These are all TEXT files!

They are all TEXT files.

- Encoding: Latin-1, ASCII, UTF-8, UTF-16, CP1252, ...
- Line endings:
  - LF (`'\n'`: OS X & Linux) , CRLF (`'\r\n'`: Windows)
- But underneath it all, these files are all TEXT files with **special formatting syntax** and **special characters** designated for formatting purposes.
  - In command line, you can `cat` and `less` through the files.
  - You can open them up in a **text editor** (Atom, Notepad++) and edit.
  - Some editors/applications are aware of the format-specific syntax and will highlight/render accordingly.
    - Unlike, say, PDF files, style attributes are NOT part of the files themselves. (e.g., markdown file)

File formats and conversion

- "Project Gutenberg Selections" corpus, from the NLTK Corpora page (http://www.nltk.org/nltk_data/).
  - You probably already have it on your system:

```
>>> nltk.corpus.gutenberg.words()
['[', 'Emma', 'by', 'Jane', 'Austen', '1816', ']', ...]
>>> nltk.corpus.gutenberg.root
FileSystemPathPointer('D:\\Lab\\nltk_data\\corpora\\gutenberg')
```

  - Examine the included text files ('austen-emma.txt', 'shakespeare-caesar.txt', ...).
  - What **encoding scheme** do the files have? Is every file UTF-8?
  - What about **line ending**? Do you see Windows style "CRLF" line ending?
  - The file command reports 'milton-paradise.txt' as a **'data' file**, not a plain text file. Is this correct?
  - Let's bring some **consistency** to this corpus! We want:
    - UTF-8 encoding
    - Unix-style LF line ending ("\n")

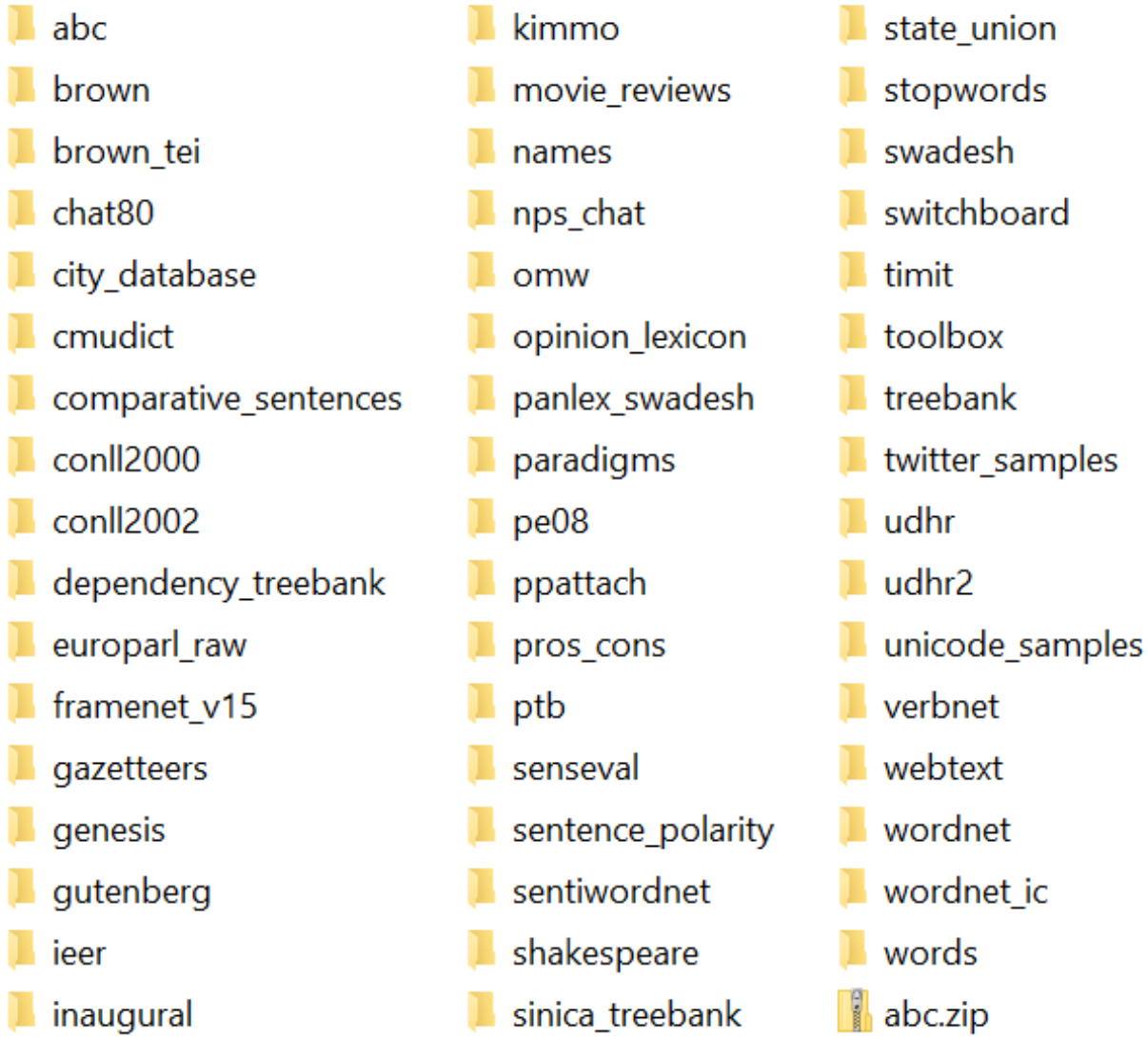
```
narae@T480s MINGW64 ~/Desktop/gutenberg
$ file *
README:                 ASCII text
austen-emma.txt:        ASCII text
austen-persuasion.txt:  ASCII text
austen-sense.txt:       ASCII text
bible-kjv.txt:          ASCII text
blake-poems.txt:        ASCII text
bryant-stories.txt:     ASCII text, with CRLF line terminators
burgess-busterbrown.txt: ASCII text, with CRLF line terminators
carroll-alice.txt:      ASCII text
chesterton-ball.txt:    ISO-8859 text
chesterton-brown.txt:   ASCII text
chesterton-thursday.txt: ASCII text
edgeworth-parents.txt:  ASCII text, with CRLF line terminators
melville-moby_dick.txt: ASCII text, with CRLF line terminators
milton-paradise.txt:    data
shakespeare-caesar.txt: ISO-8859 text
shakespeare-hamlet.txt: ASCII text
shakespeare-macbeth.txt: ASCII text
whitman-leaves.txt:     ASCII text

narae@T480s MINGW64 ~/Desktop/gutenberg
$
```

Every file should have **UTF-8** encoding with the **Unix-style LF line ending**.
Apply conversion to files using either: (1) command-line tools, (2) text editor programs such as Notepad++ and Atom.

Format conversion

- When dealing with corpora, you may need to convert 100+ files at once.
  - On-line services are too cumbersome.
  - Try batch-processing through command line.
- Automatic tools available on command line.
  - Finding out file text file encoding, line ending: `file` command (also `file -i`)
  - Encoding conversion: `iconv` (Linux, OS X, on Git Bash)
  - Line ending conversion: `unix2dos`, `dos2unix`
  - Pandoc http://www.pandoc.org/
    - Universal document coverter
    - HTML, XML, PDF, LaTeX, Markdown, Epub, MS Doc, ...
    - After installation, you can use it via command line

A brief tour of NLTK's many "corpus" data



Many of them are language data, not corpora per se

Diverse genres and data formats represented!

Resource-specific (ad-hoc) formats

- Brown corpus

```
The/at Fulton/np-tl County/nn-tl Grand/jj-tl Jury/nn-tl said/vbd
Friday/nr an/at investigation/nn of/in Atlanta's/np$ recent/jj
primary/nn election/nn produced/vbd ``/`` no/at evidence/nn ''/''
that/cs any/dti irregularities/nns took/vbd place/nn ./.
```

- Korean Treebank corpus:

```
;;05:127: 저는 그 일을 할 수 있는 한 빨리 하겠습니다 .

(S (NP-SBJ 저/NPN+는/PAU)
   (VP (NP-OBJ-LV 그/DAN
                  일/NNC+을/PCA)
       (VP (NP-ADV (S (NP-SBJ (S (NP-SBJ *pro*)
                                 (VP 하/VV+ㄹ/EAN))
                              (NP 수/NNX))
                           (ADJP 있/VJ+는/EAN))
                     (NP 한/NNX))
           (ADVP 빨리/ADV)
           (VP (LV 하/VV+겠/EPF+습니다/EFN))))
   ./SFN)
```

NOT standard
(cf. XML, JSON).
Project-dependent.

It is up to end users to understand the data format, then write code to parse data files.

Refer to documentation!

Do not re-invent the wheel.

- Don't try and parse them manually.
- There are Python libraries. Import and use them.
  - CSV & TSV: `pandas`
  - HTML & XML: Beautiful Soup (`bs4`)
  - JSON:
    - `json` library
    - `pandas.read_json`
- NLP-specific formats (Treebank, Universal Dependency, CoNLL):
  - Look at NLTK, see if it has reader
  - If not, chances are there is parser library written by someone somewhere (likely on GitHub)

Data-mining web & social media

- Twitter sample corpus
  - Static corpus: download from the NLTK data page
- How does one data-mine Twitter?
  - Answer: through **API** (Application Program Interface)
  - Getting acquainted with JSON format
  - Tutorials on on the Learning Resource page
- Libraries used: `tweepy`, `json`

Processing a static Twitter corpus

- "Twitter Samples" corpus can be downloaded from http://www.nltk.org/nltk_data/

```
In [3]: # One json object per line
        jfile = 'D:/Corpora/twitter_samples/positive_tweets.json'
        jlines = open(jfile).readlines()
        jlines[0]
```

```
Out[3]: '{"contributors": null, "coordinates": null, "text": "#FollowFriday @France_Int
        e @PKuchly57 @Milipol_Paris for being top engaged members in my community this
         week :)", "user": {"time_zone": "Paris", "profile_background_image_url": "htt
```

```
In [5]: # using json library to read line.
        import json
        json.loads(jlines[0])
```

```
Out[5]: {'contributors': None,
         'coordinates': None,
         'created_at': 'Fri Jul 24 08:23:36 +0000 2015',
         'entities': {'hashtags': [{'indices': [0, 13], 'text': 'FollowFriday'}],
          'symbols': [],
          'urls': [],
          'user_mentions': [{'id': 3222273608,
            'id_str': '3222273608',
            'indices': [14, 26],
            'name': 'France International',
```

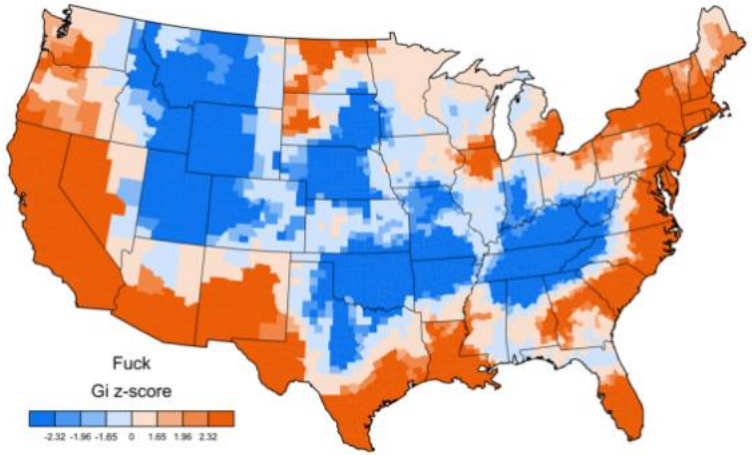
Web mining

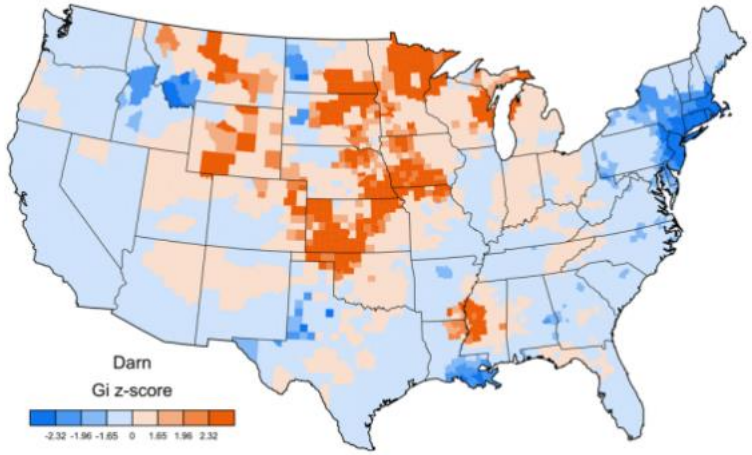
- Involves "web crawling" "web spyder", ...
- `scrapy` is the most popular library.
  - https://scrapy.org/

  ← You will have to install it first.
- You have collected a set of web pages. Now what?
  - A web page typically has tons of non-text, extraneous data such as headers, scripts, etc.
  - You will need to parse each page to extract textual data.
  - Beautiful Soup (bs4) is capable of parsing XML and HTML files.
- OK, so you've processed the web pages as data. Now what?
  - Linguistic analysis?

Mining social media for swear words

- https://stronglang.wordpress.com/2015/07/28/mapping-the-united-swears-of-america/
  - Jack Grieve mined Twitter and mapped prominent swear words by geographic regions within the US





Wrapping up

- Next class
  - Corpus linguistics, annotation
- Your project
  - Work on it! Focus on DATA.