Lecture 26

408/508 *Computational Techniques for Linguists*

Reminders

- Term project
- Class Survey

Last Time

- Book: Mrs. Dalloway by Virginia Woolf (1925)
- `http://gutenberg.net.au/ebooks02/0200991.txt`
- Edited down to size, statistics:
  - 362324 characters; 77707 words; 7637 vocab.; 9.8% lexical diversity
  - adjectives: 3692 words; 1195 vocab using `nltk.pos_tag()`
  - verbs: 12417 words; 2286 vocab.
- First thing: a note on encoding: Latin-1 vs. UTF-8.
- Second thing: #punctuationmatters

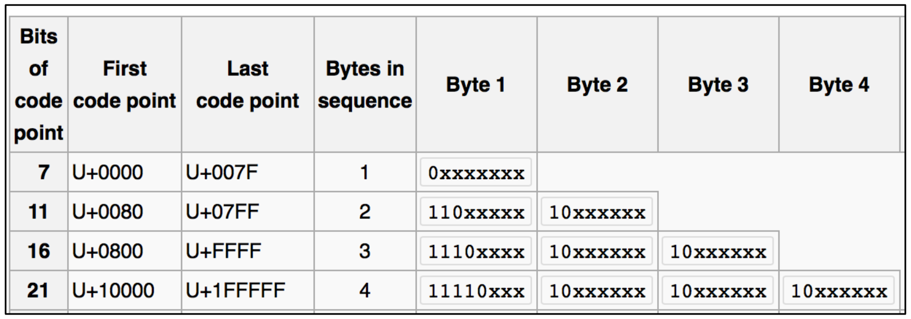
Downloaded file

- Edited down to size `0200991.txt`, latin-1 encoding:

```
>>> f = open('0200991.txt')
>>> raw = f.read()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/Users/sandiway/opt/anaconda3/lib/python3.9/codecs.py", line 322, in decode
    (result, consumed) = self._buffer_decode(data, self.errors, final)
UnicodeDecodeError: 'utf-8' codec can't decode byte 0xe4 in position 10585: invalid continuation byte
>>> f.close()
>>> f = open('0200991.txt','r',encoding='latin-1')
>>> raw = f.read()
>>> len(raw)
362325
>>> import nltk
>>> words = nltk.word_tokenize(raw)
>>> len(words)
77707
```

UnicodeDecodeError

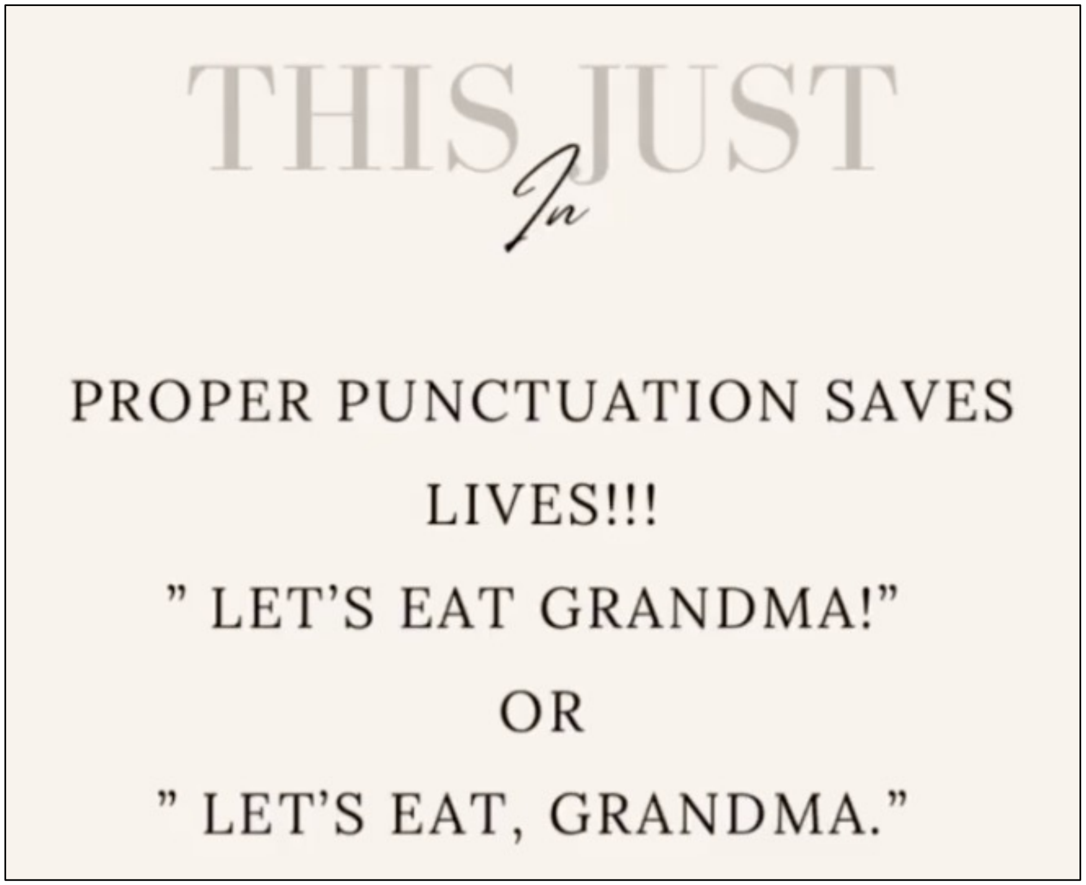
- `'utf-8' codec can't decode byte 0xe4 in position 10585: invalid continuation byte`
- Let's decode this error message:
  - Note:
    - 0x means hex
  - e4 in binary is `11100100`
  - `echo 'ibase=16; obase=2; E4' | bc`
  - continuation bytes (i.e. bytes 2-4) must begin with `10xxxxxx`
  - Mystery solved! `11100100` clashes with `10xxxxxx`



Bits of code point	First code point	Last code point	Bytes in sequence	Byte 1	Byte 2	Byte 3	Byte 4
7	U+0000	U+007F	1	0xxxxxxx			
11	U+0080	U+07FF	2	110xxxxx	10xxxxxx		
16	U+0800	U+FFFF	3	1110xxxx	10xxxxxx	10xxxxxx	
21	U+10000	U+1FFFFF	4	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

#punctuationmatters

- Our word count includes punctuation …



THIS JUST *In*

PROPER PUNCTUATION SAVES LIVES!!!

" LET'S EAT GRANDMA!"

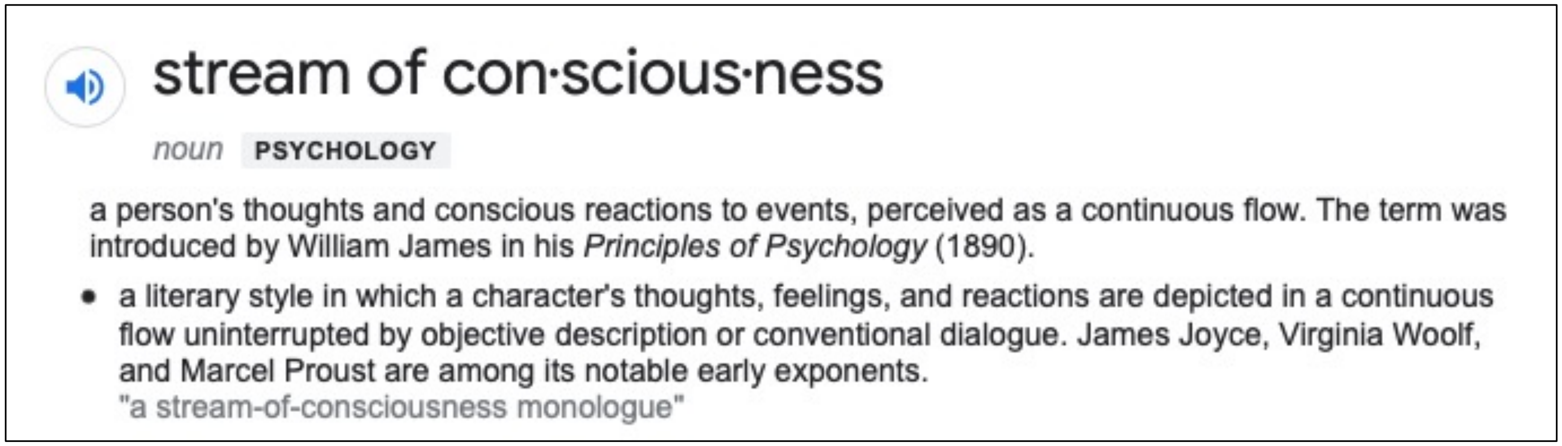
OR

" LET'S EAT, GRANDMA."

from my Facebook feed

Today's Topic

- Literary Style: *Stream of consciousness*
  - we look at using nltk to explore this



stream of con·scious·ness

noun **PSYCHOLOGY**

a person's thoughts and conscious reactions to events, perceived as a continuous flow. The term was introduced by William James in his *Principles of Psychology* (1890).

- a literary style in which a character's thoughts, feelings, and reactions are depicted in a continuous flow uninterrupted by objective description or conventional dialogue. James Joyce, Virginia Woolf, and Marcel Proust are among its notable early exponents.
  "a stream-of-consciousness monologue"

nltk.sents()

nltk book 3.8 Segmentation

- Brown corpus (*pre-segmented*), use `.sents()`:

```
>>> from nltk.corpus import brown
>>> len(brown.words())
1161192
>>> len(brown.sents())
57340
>>> brown.sents()[0]
['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', 'Friday', 'an',
'investigation', 'of', "Atlanta's", 'recent', 'primary', 'election',
'produced', '``', 'no', 'evidence', "''", 'that', 'any', 'irregularities',
'took', 'place', '.']
>>> '{:.2f}'.format(len(brown.words()) / len(brown.sents()))
'20.25' # average sentence length, number of words
```

nltk.sent_tokenize()

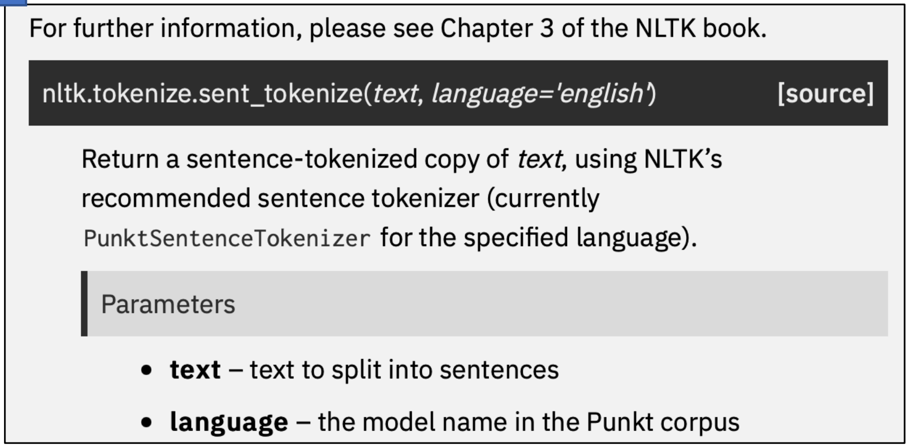
- On *Mrs. Dalloway*:

```
>>> sents = nltk.sent_tokenize(raw)
>>> len(sents)
3590
>>> sents[0]
'Title:      Mrs.
Dalloway\nAuthor:     Virginia
Woolf\n\n\n\n\nMrs. Dalloway said she would
buy the flowers herself.'
>>> sents[1]
'For Lucy had her work cut out for her.'
>>> sents[-1]
'THE END'
>>> sents[-2]
'For there she was.'
```

not words!

each sentence is a string, not word tokenized yet

https://www.nltk.org/api/nltk.tokenize.html



For further information, please see Chapter 3 of the NLTK book.

nltk.tokenize.sent_tokenize(*text*, *language='english'*) [source]

Return a sentence-tokenized copy of *text*, using NLTK's recommended sentence tokenizer (currently `PunktSentenceTokenizer` for the specified language).

Parameters

- **text** – text to split into sentences
- **language** – the model name in the Punkt corpus

Kiss, Tibor **and** Strunk, Jan (2006): Unsupervised Multilingual Sentence Boundary Detection. Computational Linguistics 32: 485-525.

Mrs. Dalloway

```
>>> sents = nltk.sent_tokenize(raw)
0.  'Title:      Mrs. Dalloway\r\nAuthor:     Virginia Woolf\r\n\r\n\r\n
     \r\n\r\nMrs. Dalloway said she would buy the flowers herself.'
1.  'For Lucy had her work cut out for her.'
2.  "The doors would be taken\r\noff their hinges; Rumpelmayer's men were
    coming."
3.  'And then, thought\r\nClarissa Dalloway, what a morning--fresh as if
    issued to children\r\non a beach.'
4.  'What a lark!'
5.  'What a plunge!'
6.  'For so it had always seemed to her,\r\nwhen, with a little squeak of
    the hinges, which she could hear now,\r\nshe had burst open the French
    windows and plunged at Bourton into\r\nthe open air.'
```

Virginia Woolf

- Famous for her stream-of-consciousness style of writing:

```
>>> sents[7] # sentence #8
```

- 'How fresh, how calm, stiller than this of course,\nthe air was in the early morning; like the flap of a wave; the kiss\nof a wave; chill and sharp and yet (for a girl of eighteen as she\nthen was) solemn, feeling as she did, standing there at the open\nwindow, that something awful was about to happen; looking at the\nflowers, at the trees with the smoke winding off them and the rooks\nrising, falling; standing and looking until Peter Walsh said,\n"Musing among the vegetables?"'

```
>>> s = nltk.word_tokenize(sents[7])
>>> len(s)
107
```

cf. Brown corpus average of 20 words/sentence

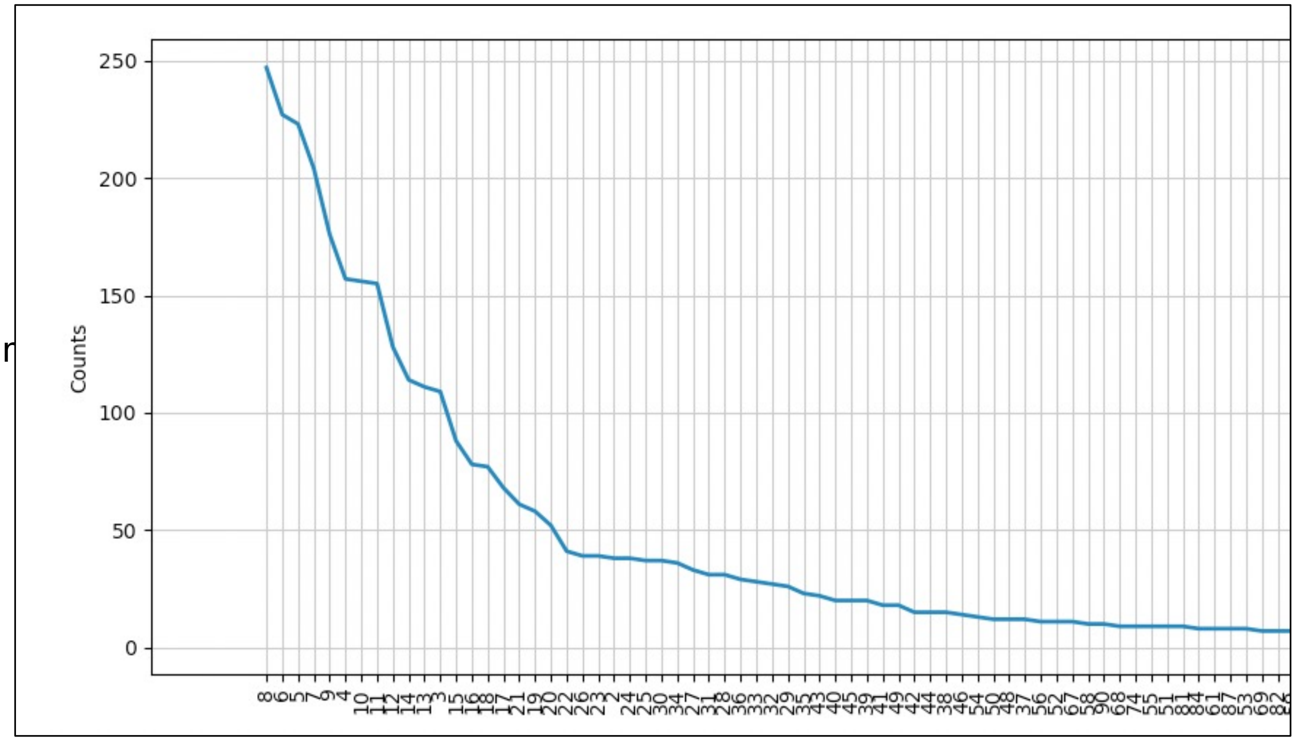
The best stream-of-consciousness novels

- https://www.theguardian.com/books/2009/jan/20/1000-novels-classic-novels
    - **James Joyce: Ulysses (1922)**
      Virginia Woolf: To the Lighthouse (1927)
      William Faulkner: The Sound and the Fury (1929)
      Samuel Beckett: Malone Dies (1951)
- "Let us record the atoms as they fall upon the mind in the order in which they fall," Woolf declared in a famous essay, *Modern Fiction*. "Let us trace the pattern, however disconnected in appearance, which each sight or incident scores upon the consciousness."

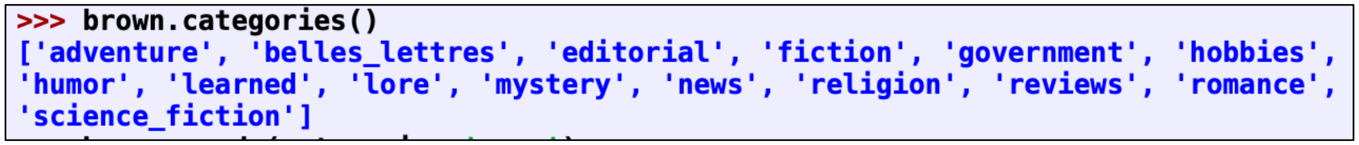
Mrs. Dalloway

- Sentence length distribution:

```
>>> slen = [len(nltk.word_tokenize(sent)) for sent in sents]
>>> len(slen)
3590
>>> fd = nltk.FreqDist(slen)
>>> fd
FreqDist({8: 247, 6: 227, 5:
14: 114, ...})
>>> print(fd)
<FreqDist with 147 samples ar
>>> fd.plot()
```



Brown Corpus Fiction

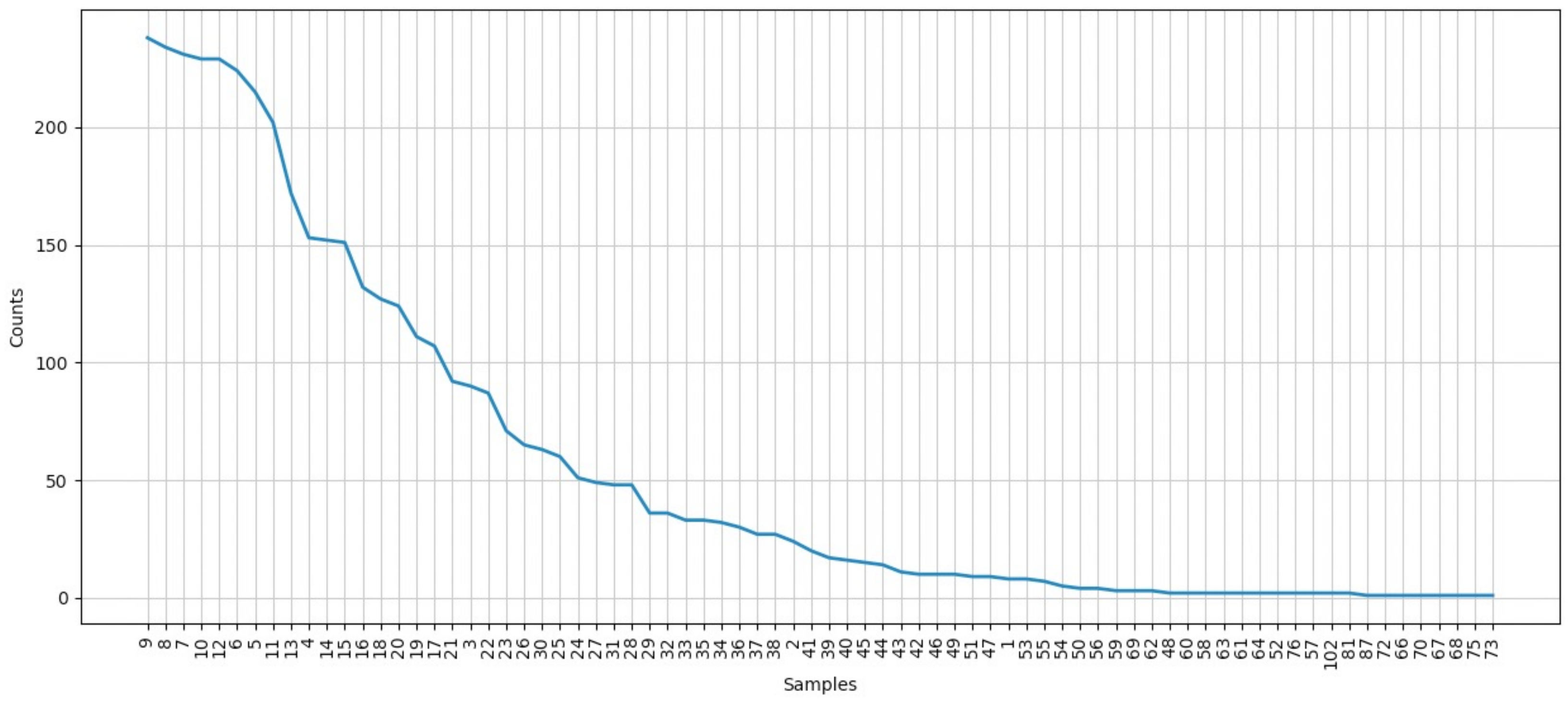


```
>>> brown.categories()
['adventure', 'belles_lettres', 'editorial', 'fiction', 'government', 'hobbies',
'humor', 'learned', 'lore', 'mystery', 'news', 'religion', 'reviews', 'romance',
'science_fiction']
```

- Sentence length distribution:

```
>>> bsents = brown.sents(categories='fiction')
>>> len(bsents)
4249
>>> bslen = [len(sent) for sent in bsents]
>>> fd2 = nltk.FreqDist(bslen)
>>> fd2
FreqDist({9: 238, 8: 234, 7: 231, 10: 229, 12: 229, 6: 224, 5: 215, 11: 202, 13: 172, 4:
153, ...})
>>> print(fd2)
<FreqDist with 76 samples and 4249 outcomes>
>>> fd2.plot()
<AxesSubplot:xlabel='Samples', ylabel='Counts'>
```

Brown Corpus Fiction

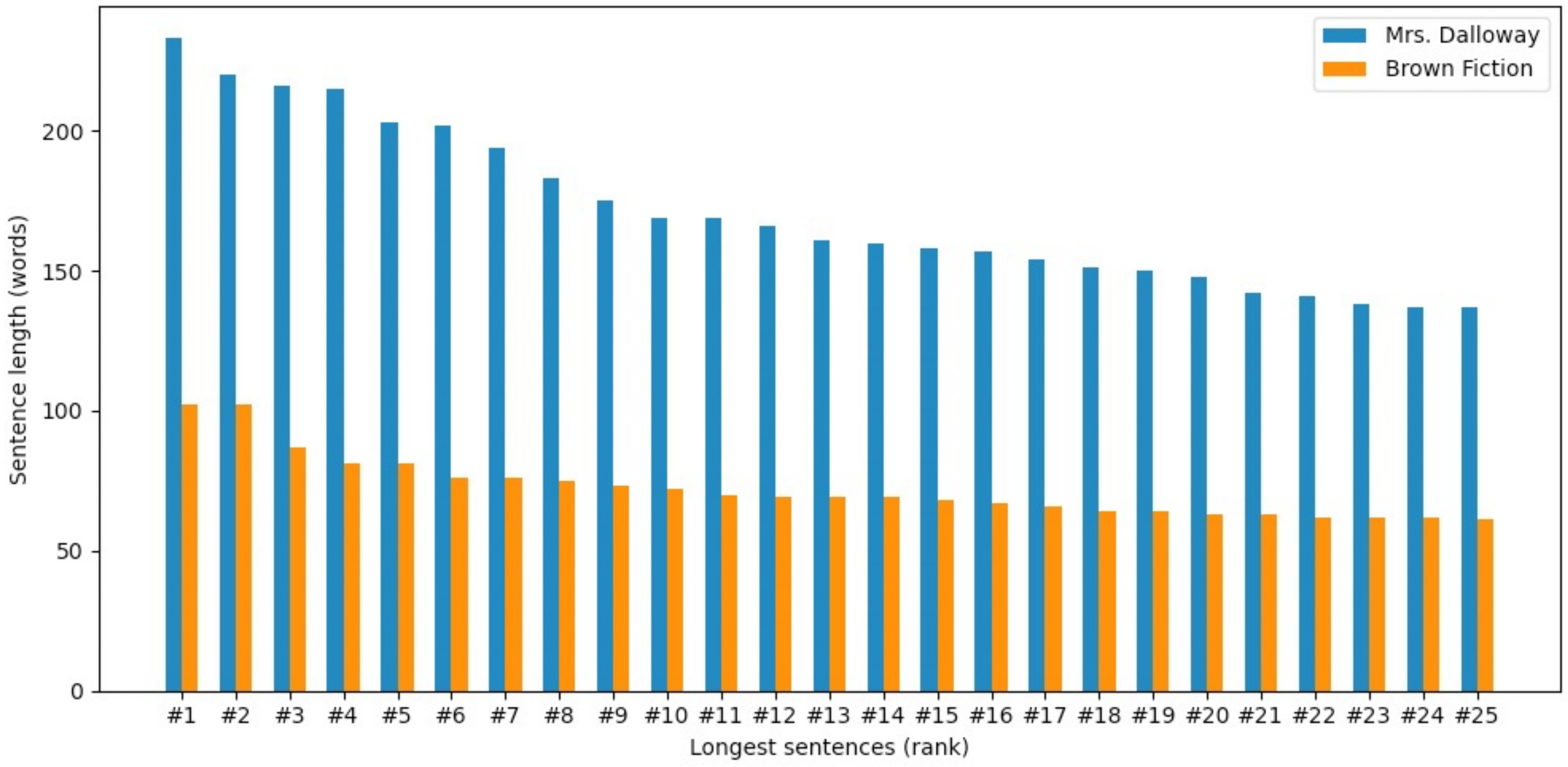


Mrs. Dalloway vs. Brown Corpus Fiction

- The 25 longest sentences:

```
>>> sorted(bslen, reverse=True)[:25] % Brown
[102, 102, 87, 81, 81, 76, 76, 75, 73, 72, 70, 69, 69, 69, 68, 67,
66, 64, 64, 63, 63, 62, 62, 62, 61]
>>> sorted(slen, reverse=True)[:25] % Mrs. Dalloway
[233, 220, 216, 215, 203, 202, 194, 183, 175, 169, 169, 166, 161,
160, 158, 157, 154, 151, 150, 148, 142, 141, 138, 137, 137]
```

Mrs. Dalloway vs. Brown Corpus Fiction Top25



Mrs. Dalloway vs. Brown Corpus Fiction

- https://matplotlib.org/stable/gallery/lines_bars_and_markers/barchart.html#sphx-glr-gallery-lines-bars-and-markers-barchart-py
- Let's build a bar chart using matplotlib:

```
>>> slen25 = sorted(slen, reverse=True)[:25]
>>> slen25
[233, 220, 216, 215, 203, 202, 194, 183, 175, 169,
169, 166, 161, 160, 158, 157, 154, 151, 150, 148, 142,
141, 138, 137, 137]
>>> bslen25 = sorted(bslen, reverse=True)[:25]
>>> bslen25
[102, 102, 87, 81, 81, 76, 76, 75, 73, 72, 70, 69, 69,
69, 68, 67, 66, 64, 64, 63, 63, 62, 62, 62, 61]
>>> labels = ['#'+str(n) for n in range(1,26)]
>>> labels
['#1', '#2', '#3', '#4', '#5', '#6', '#7', '#8', '#9',
'#10', '#11', '#12', '#13', '#14', '#15', '#16',
'#17', '#18', '#19', '#20', '#21', '#22', '#23',
'#24', '#25']
>>> import matplotlib.pyplot as plt
>>> width = 0.3
```

```
>>> import numpy as np
>>> x = np.arange(len(labels))
>>> x
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11,
12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24])
>>> fig, ax = plt.subplots()
>>> ax.set_xlabel('Longest sentences (rank)')
Text(0.5, 0, 'Longest sentences (rank)')
>>> ax.set_ylabel('Sentence length (words)')
Text(0, 0.5, 'Sentence length (words)')
>>> ax.set_xticks(x, labels)
>>> bars1 = ax.bar(x-width/2, slen25, width,
label='Mrs. Dalloway')
>>> bars2 = ax.bar(x+width/2, bslen25, width,
label='Brown Fiction')
>>> ax.legend()
<matplotlib.legend.Legend object at 0x168378df0>
>>> plt.show()
```

matplotlib

https://matplotlib.org/stable/index.html

Data:

- labels `['#1', '#2', '#3', '#4', '#5', '#6', '#7', '#8', '#9', '#10', '#11', '#12', '#13', '#14', '#15', '#16', '#17', '#18', '#19', '#20', '#21', '#22', '#23', '#24', '#25'`
- x `array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24])`

Functions:

- **matplotlib.pyplot.subplots()** `fig, ax = plt.subplots()`
  - Returns:**fig** *Figure* **ax** *Axes*
- **Axes.set_ylabel(*ylabel*)** `ax.set_ylabel('Sentence length (words)')`
  - Set the label for the y-axis.
- **Axes.set_xlabel(*xlabel*)** `ax.set_xlabel('Longest sentences (rank)')`
  - Set the label for the x-axis.
- **Axes.set_xticks(*ticks, labels*)** `ax.set_xticks(x, labels)`
  - Set the xaxis' tick locations and optionally labels.
- **Axes.bar(*x, height, width=0.8*)** `bars1 = ax.bar(x-width/2, slen25, width, label='Mrs. Dalloway')`
- `bars2 = ax.bar(x+width/2, bslen25, width, label='Brown Fiction')`
  - Make a bar plot. The bars are positioned at x. Their dimensions are given by height and width.
- **Axes.legend()** `ax.legend()`
  - Place a legend on the Axes.

Today's Topic

- Literary Style: *Stream of consciousness*
  - we look at using nltk to explore this
  - easy to spot this using sentence length
  - cf. free indirect style
    - "Free Indirect Speech is a form of narration written in the third person while maintaining some essential elements of a first-person narrator. The author can thus describe the inner workings of their characters; their private emotions and thoughts, while still remaining at an observational distance."
    - *Emma* by Jane Austen
    - She was a very pretty girl, and her beauty happened to be of a sort which Emma particularly admired. She was short, plump, and fair, with a fine bloom, blue eyes, light hair, regular features, and a look of great sweetness, and, before the end of the evening, Emma was as much pleased with her manners as her person, and quite determined to continue the acquaintance.