LING 388: Computers and Language

Lecture 17

Today's Topics

- Stylometry contd.
  - reading homework: Mendenhall1887.pdf
  - **idea**: use word length statistics to figure out authorship

Live Example

- Course website (*Charles Dickins*): `oliver_twist.txt`
- Python:
  - `import nltk`
  - `raw = open('oliver_twist.txt', encoding='utf-8', errors='ignore').read()`
  - `words = nltk.word_tokenize(raw)`

  Mendenhall's *word-spectrum* based on word length:
  - `len1 = [len(word) for word in words[0:1000]]`
  - `len2 = [len(word) for word in words[1000:2000]]`
  - `len3 = [len(word) for word in words[2000:3000]]`

histogram plot

data columns

```
matplotlib.pyplot.hist(x, bins)
```

Parameters:

x : ***(n,) array or sequence of (n,) arrays***

Input values, this takes either a single array or a sequence of arrays which are not required to be of the same length.

bins : ***int or sequence or str, default:*** `rcParams["hist.bins"]` ***(default:*** `10`***)***

If *bins* is an integer, it defines the number of equal-width bins in the range.

If *bins* is a sequence, it defines the bin edges, including the left edge of the first bin and the right edge of the last bin; in this case, bins may be unequally spaced. All but the last (righthand-most) bin is half-open. In other words, if *bins* is:

```
[1, 2, 3, 4]
```

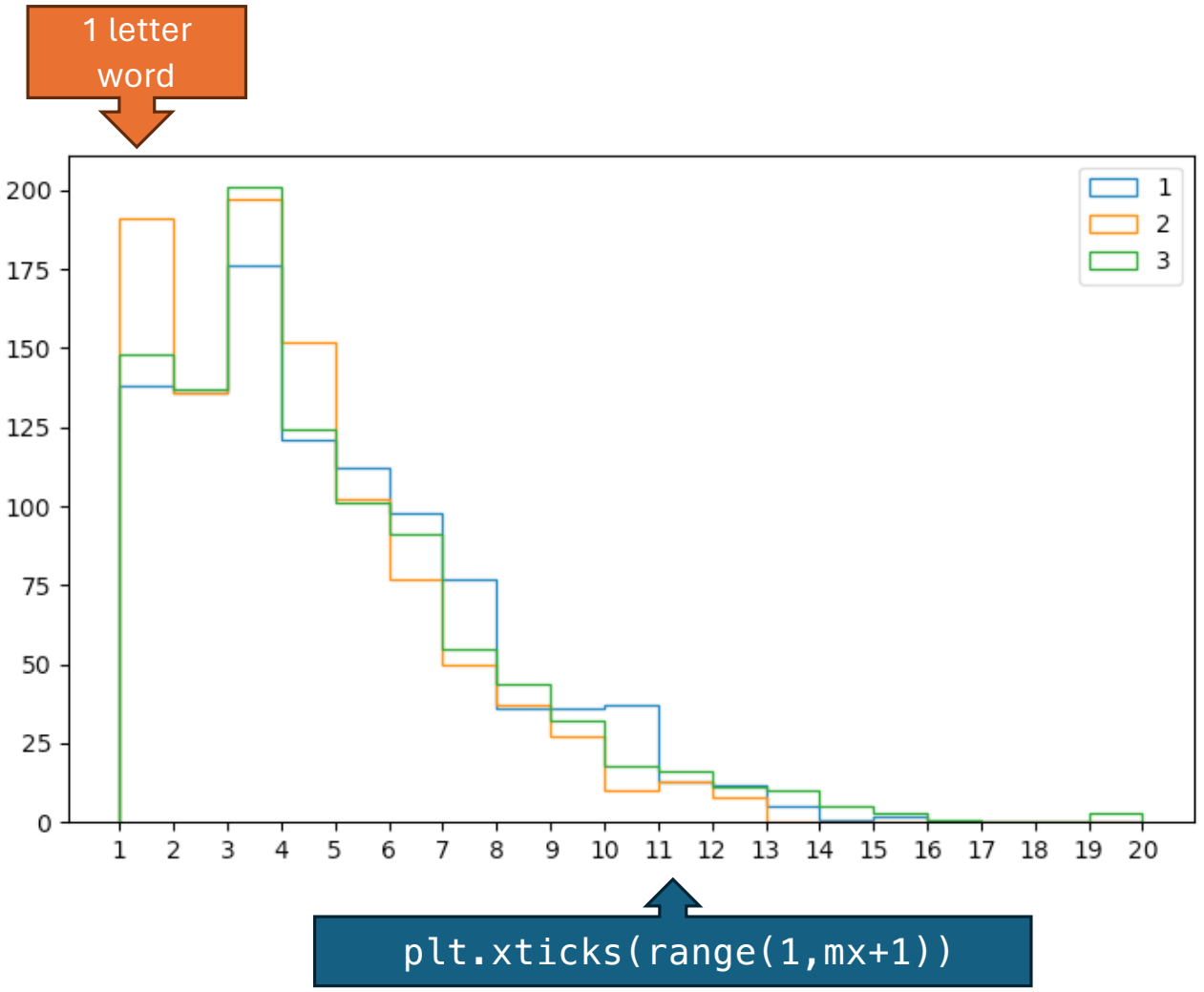
then the first bin is `[1, 2)` (including 1, but excluding 2) and the second `[2, 3)`. The last bin, however, is `[3, 4]`, which *includes* 4.

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html

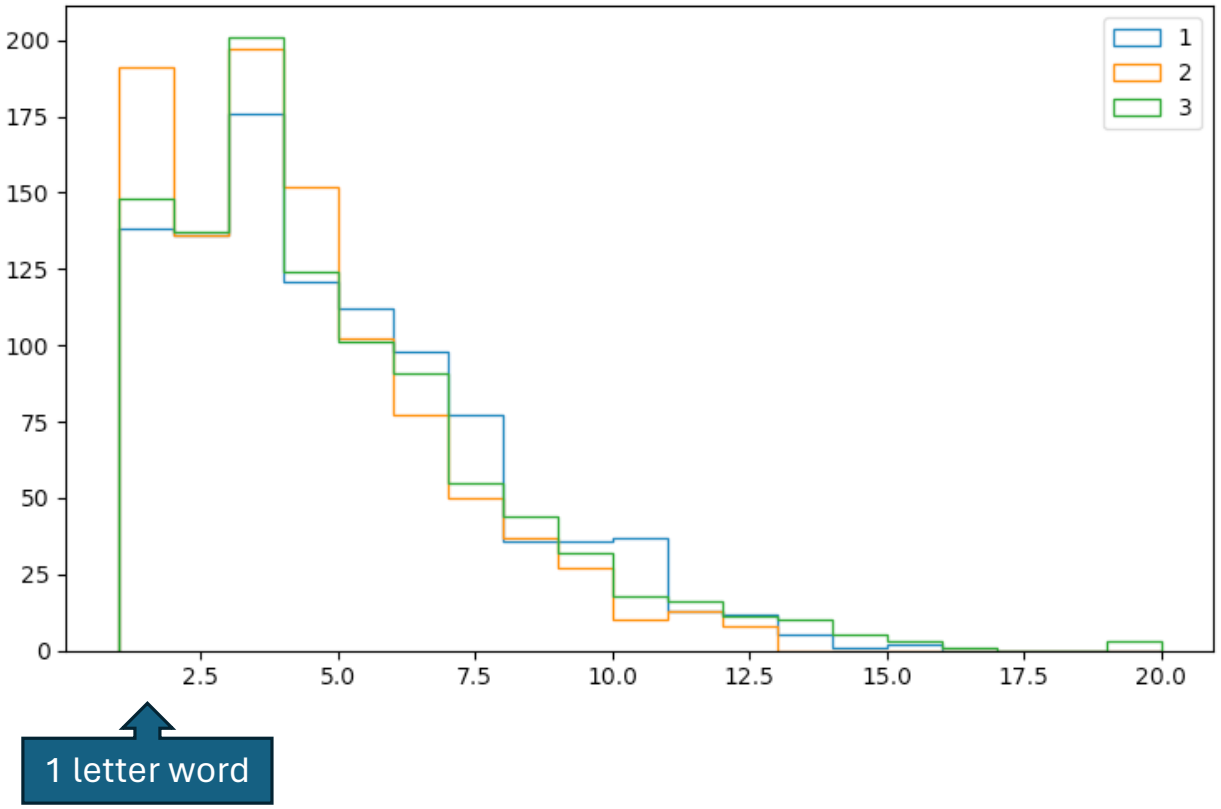
histogram plot

- Let's use matplotlib.pyplot to plot them together.
  - `import matplotlib.pyplot as plt`
  - `mx = max(max(len3),max(len2),max(len1))`
  - `mx` (*longest lword across the two chunks*)
  - `20`
  - `plt.hist(len1, range(1,mx+1), histtype='step', label='1')`
  - `plt.hist(len2, range(1,mx+1), histtype='step', label='2')`
  - `plt.hist(len3, range(1,mx+1), histtype='step', label='3')`
  - `plt.xticks(range(1,mx+1))`
  - `plt.legend()`
  - `plt.show()`

histogram plot



without specifying xticks()



Lecture 12: *taking out the punctuation*

- How can we identify punctuation?
  - strings of length 1? *how about words like 'A' 'a' or 'I'?*
  - non-alphabetic strings

> `str.`**`isalnum`**`()`
>
> Return `True` if all characters in the string are alphanumeric and there is at least one character, `False` otherwise. A character `c` is alphanumeric if one of the following returns `True`: `c.isalpha()`, `c.isdecimal()`, `c.isdigit()`, or `c.isnumeric()`.

  - Conditional list comprehension for punctuation:
    - `[word for word in words if not word.isalnum()]`
    - Excludes (correctly): `? " : ; .` *etc.*
    - Excludes (incorrectly): *Mr. three-penny devil-may-care*

histogram plot without punctuation

- Better code:

```
any(c.isalpha() for c in 'three-penny')
True
```

- Filter out any word that doesn't have an alphabetic character:

```
>>> len(words)
199836
>>> words2 = [word for word in words if any(c.isalpha() for c in
word)]
>>> len(words2)
160639
```

Documentation

any(*iterable*)

Return `True` if any element of the *iterable* is true. If the iterable is empty, return `False`. Equivalent to:

```
def any(iterable):
    for element in iterable:
        if element:
            return True
    return False
```

`str`.**isalpha**()

Return `True` if all characters in the string are alphabetic and there is at least one character, `False` otherwise. Alphabetic characters are those characters defined in the Unicode character database as "Letter", i.e., those with general category property being one of "Lm", "Lt", "Lu", "Ll", or "Lo". Note that this is different from the Alphabetic property defined in the section 4.10 'Letters, Alphabetic, and Ideographic' of the Unicode Standard.

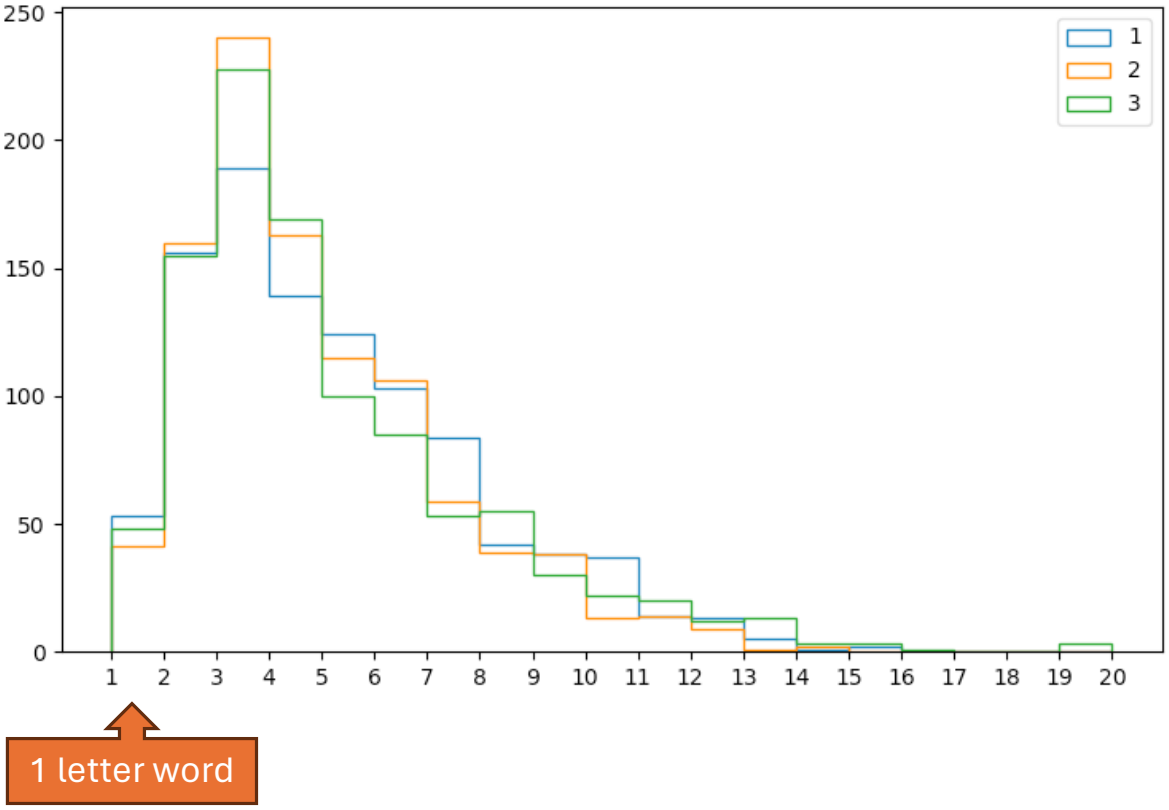
histogram plot without punctuation

Mendenhall's *word-spectrum* based on word length:

- `len1 = [len(word) for word in words2[0:1000]]`
- `len2 = [len(word) for word in words2[1000:2000]]`
- `len3 = [len(word) for word in words2[2000:3000]]`

```
plt.hist(len1, range(1,mx+1), histtype='step', label='1')
plt.hist(len2, range(1,mx+1), histtype='step', label='2')
plt.hist(len3, range(1,mx+1), histtype='step', label='3')
plt.xticks(range(1,mx+1))
plt.legend()
plt.show()
```

histogram plot without punctuation



Oliver Twist (5 groups of 1000)

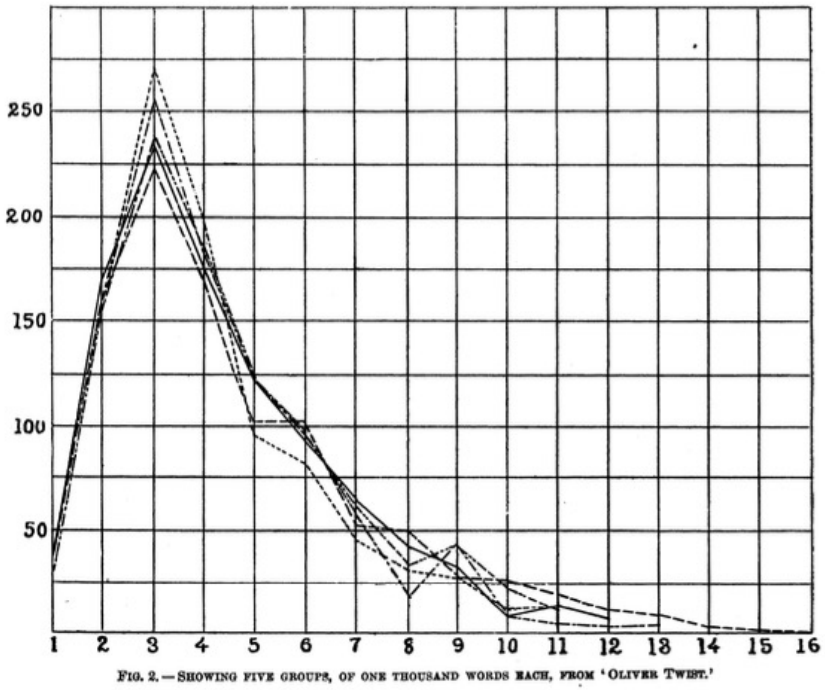
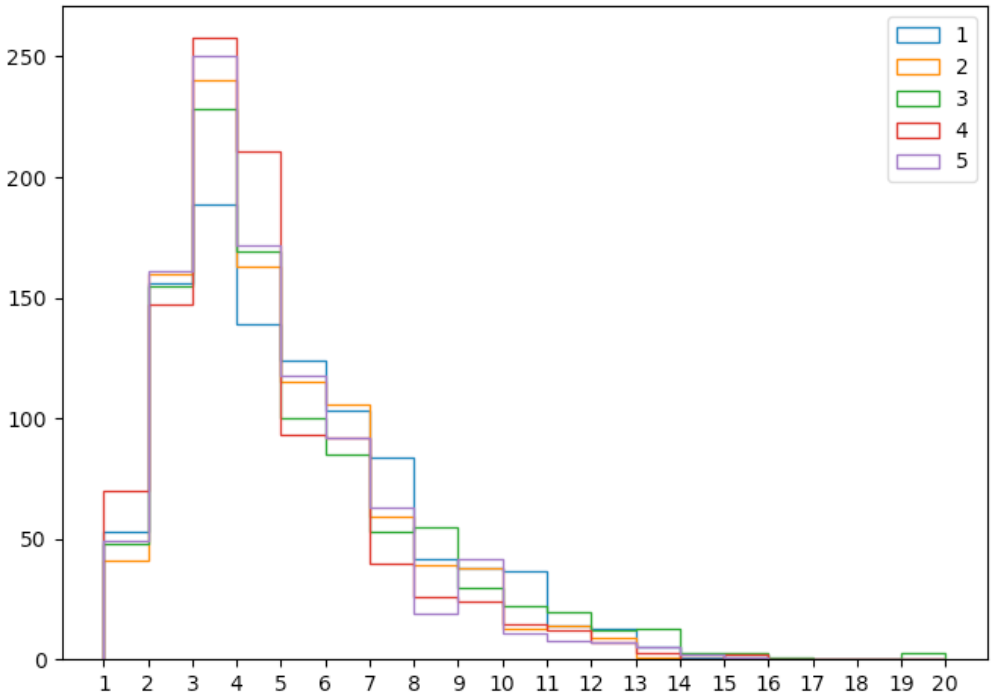


FIG. 2.—SHOWING FIVE GROUPS, OF ONE THOUSAND WORDS EACH, FROM 'OLIVER TWIST.'



More is better?

- Mendenhall asserts 10,000 is better than 5,000, better than 1,000.

When the number of words in a group is increased to five thousand, the accidental irregularities begin to disappear, the curve becomes smoother, approximating more nearly to the normal curve which, it is assumed, is characteristic of the writer. Fig. 4 exhibits two groups, each of five thousand words, from 'Oliver Twist,' and it will be seen that considerable differences still ex-

Fig. 5 exhibits these two groups of five thousand words combined in one of ten thousand, giving a curve of greater smoothness, and approximating still more closely to the normal curve of the writer.

More is better?

5,000

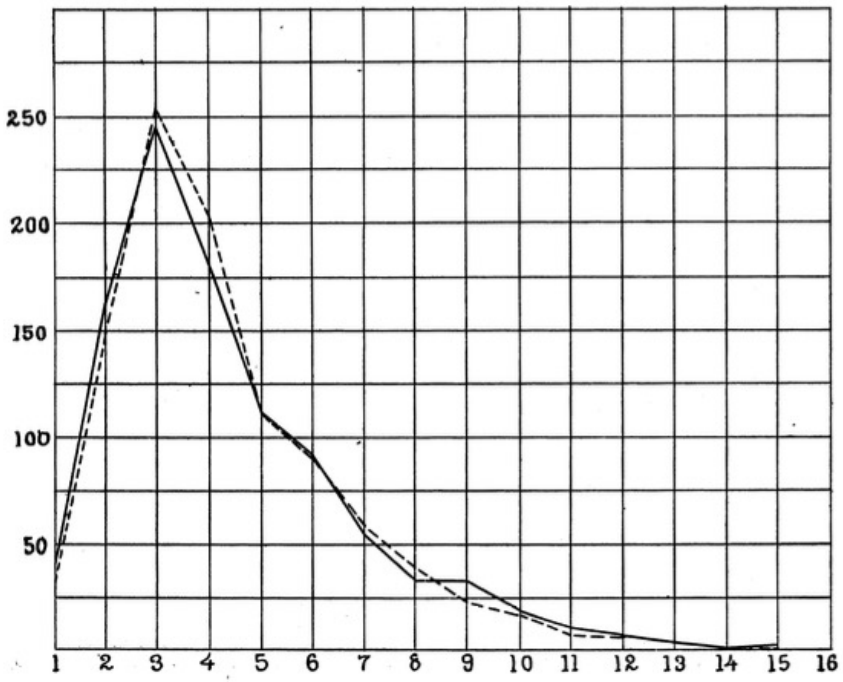


FIG. 4.—TWO GROUPS, OF FIVE THOUSAND WORDS EACH, FROM 'OLIVER TWIST.'

10,000

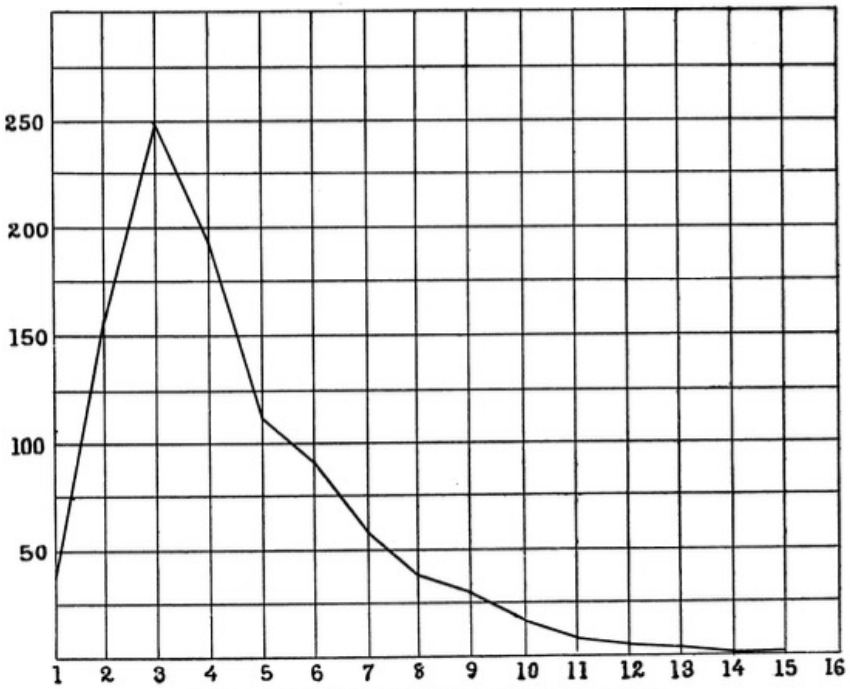


FIG. 5.—CURVE FOR TEN THOUSAND WORDS FROM 'OLIVER TWIST.'