Cultural Analytics

ENGL 64.05

Fall 2019

Prof. James E. Dobson

Nov 11, 2019

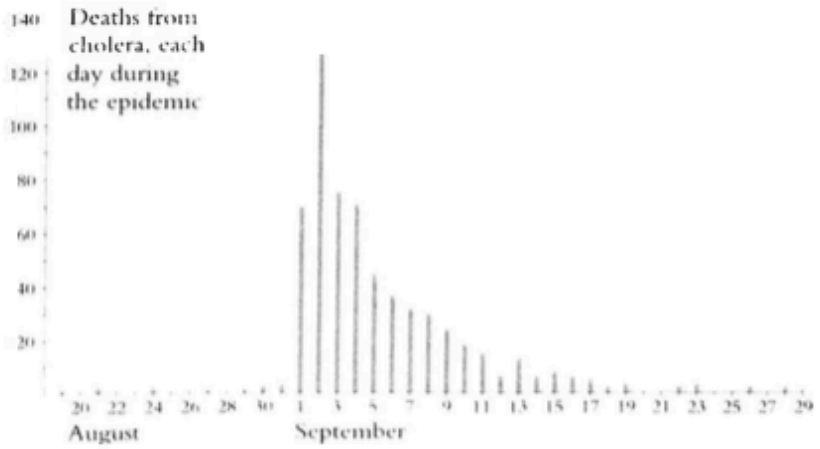
Data as Media

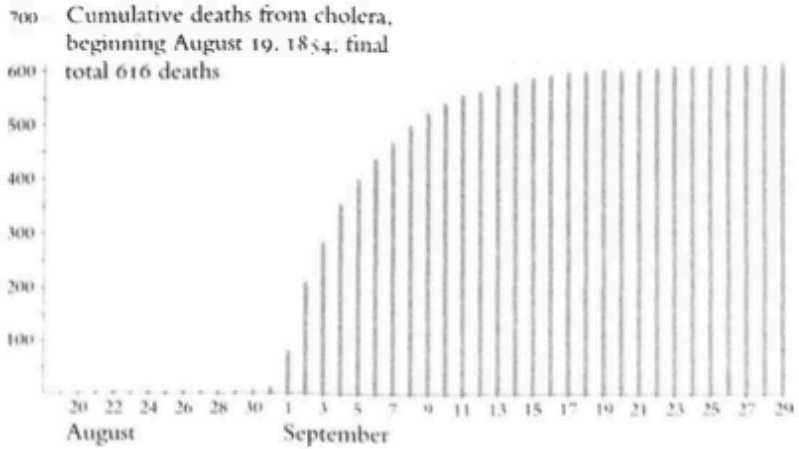
- Media studies is especially interested in the *affordances* and *limitations* of media format.
- Media-specific analysis pays close attention to the materiality of the form of the medium.
- Posner and Klein write that "these debates about medium specificity allow us to begin to identify certain features that inhere in and around data, such as information segmentation or scales of measurement, that impact its significance and use" (3).
- Data are materialized when we format them into our tools.
- Visualizations are specific media renderings of data. How are these subject to similar critiques as data? Are visualizations to be taken as data themselves?

Tufte's Logic of Data Display and Analysis

1. Place the data in an appropriate context for assessing cause and effect…But descriptive narration is not causal explanation; the passage of time is a poor explanatory variable (29)
2. Make quantitative comparisons. The deep, fundamental question in statistical analysis is Compared with what? (30).
3. Consider alternative explanations and contrary cases (32).
4. Assess possible errors in the numbers reported in graphics (34).

Re-presenting Data





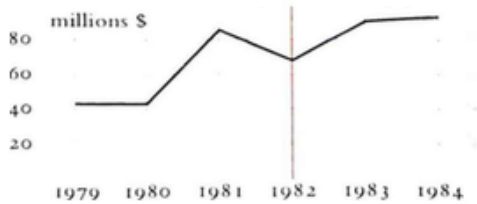
(Tufte, 29)

Interpreting Data with Visualizations

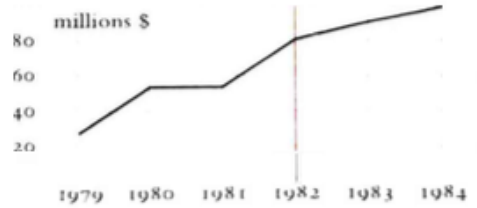
Cultural Analytics

Tufte writes that "time-series are exquisitely sensitive to choice of intervals and end points. Nonetheless, many aggregations arc perfectly sensible, reducing the tedious redundancy and uninteresting complexity of large data files; for example, the daily data amalgamate times of death originally recorded to the hour and even minute. If in doubt, graph the detailed underlying data to assess the effects of aggregation" (37).

Above, this chart shows *quarterly* revenue data in a financial graphic for a legal case. Several dips in revenue are visible.



Aggregating the quarterly data into years, this chart above shows revenue by *fiscal year* (beginning July 1, ending June 30). Note the dip in 1982, the basis of a claim for damages.



Shown above are the same quarterly revenue data added up into *calendar years*. The 1982 dip has vanished.

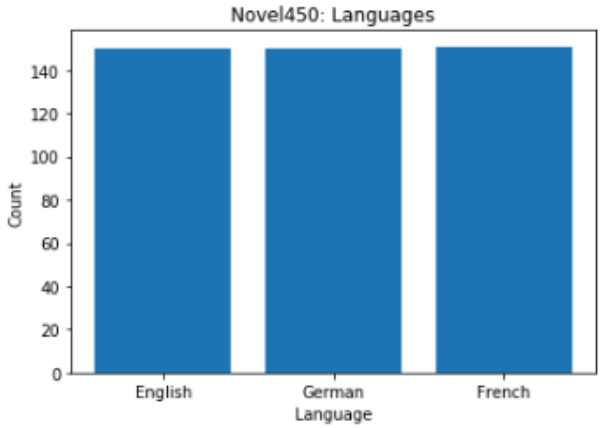
Matplotlib Bar

Cultural Analytics

```
In [1]: from glob import glob
        data = glob('data/Novel450/*')
```

```
In [2]: total_novels = len(data)
        english_novels = len([x for x in data if x.__contains__('EN_')])
        german_novels = len([x for x in data if x.__contains__('DE_')])
        french_novels = len([x for x in data if x.__contains__('FR_')])
```

```
In [3]: import matplotlib.pyplot as plt
        %matplotlib inline
        data = [english_novels,german_novels,french_novels]
        labels = ["English","German","French"]
        plt.title("Novel450: Languages")
        plt.ylabel("Count")
        plt.xlabel("Language")
        plt.bar(range(len(data)),data,tick_label=labels)
        plt.show()
```



Matplotlib Pie

Cultural Analytics

In [1]:
```
from glob import glob
data = glob('data/Novel450/*')
```

In [2]:
```
total_novels = len(data)
english_novels = len([x for x in data if x.__contains__('EN_')])
german_novels = len([x for x in data if x.__contains__('DE_')])
french_novels = len([x for x in data if x.__contains__('FR_')])
```

In [3]:
```
import matplotlib.pyplot as plt
%matplotlib inline
fig, ax = plt.subplots(figsize=(6, 3),subplot_kw=dict(aspect="equal"))
wedges, texts = ax.pie([english_novels,german_novels,french_novels])
ax.set_title("Novel450: Languages")
ax.legend(wedges, ["English","German","French"],
          title="Languages",
          loc="center left",
          bbox_to_anchor=(1, 0, 0.5, 1))
plt.show()
```

Pandas DataFrame

Cultural Analytics

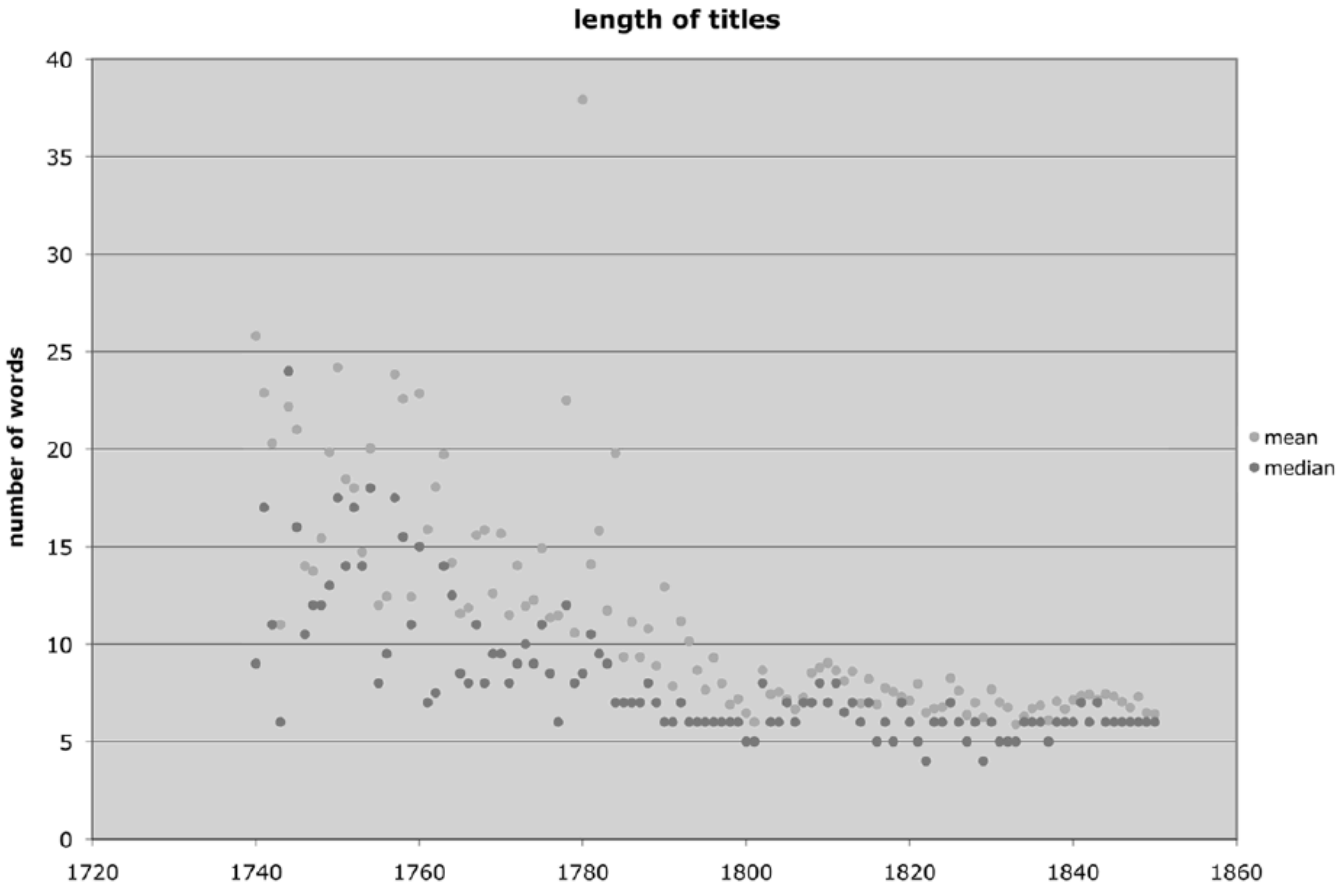
In [1]:
```
from glob import glob
data = glob('data/Novel450/*')
```

In [2]:
```
total_novels = len(data)
english_novels = len([x for x in data if x.__contains__('EN_')])
german_novels = len([x for x in data if x.__contains__('DE_')])
french_novels = len([x for x in data if x.__contains__('FR_')])
```

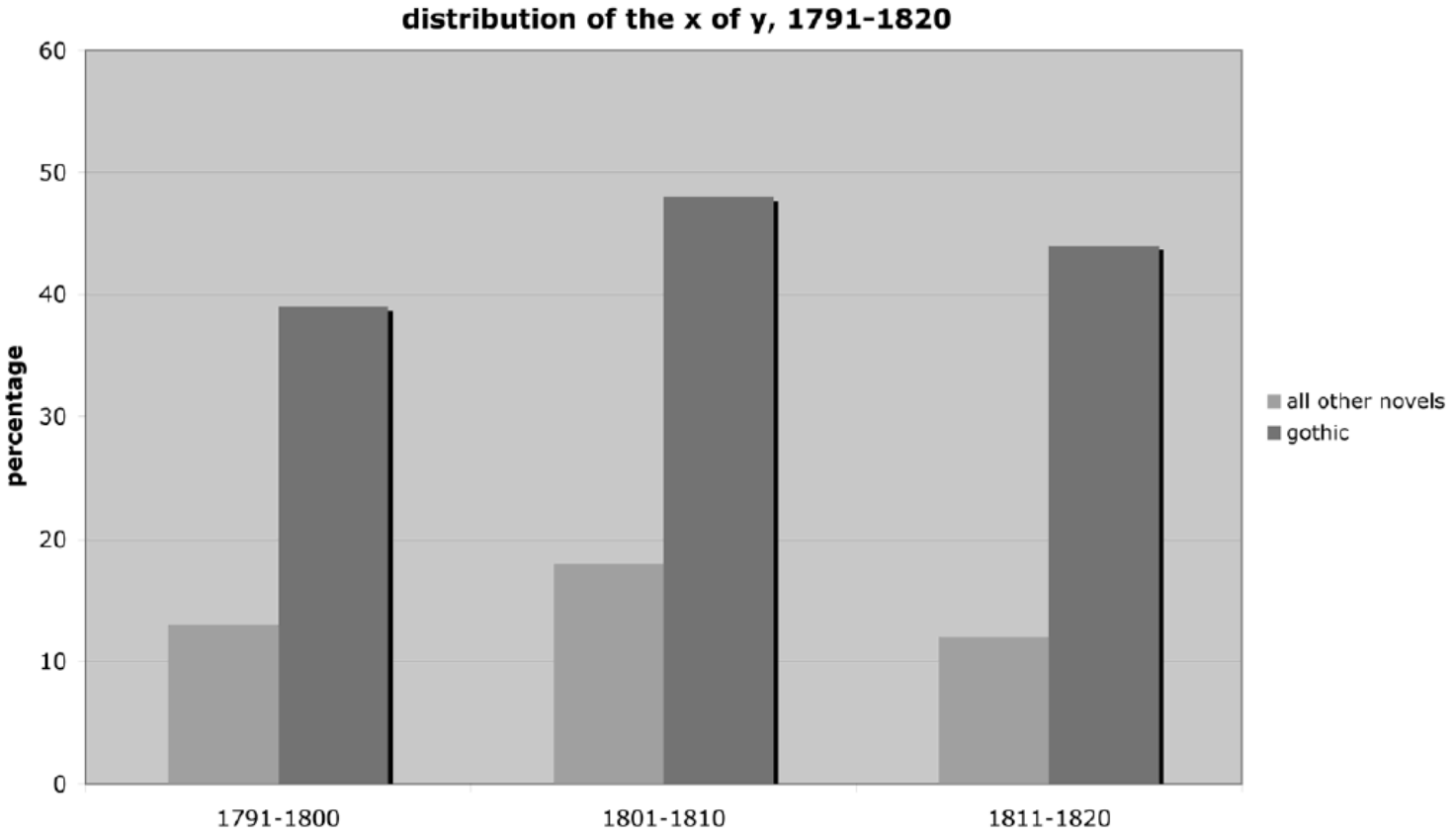
In [3]:
```
import pandas as pd
data = [['English Novels',english_novels],
        ['German Novels',german_novels],
        ['French Novels',french_novels]]
df = pd.DataFrame(data,columns=['Language','Count'])
df
```

Out[3]:

	Language	Count
0	English Novels	150
1	German Novels	150
2	French Novels	151



Moretti 2009



Moretti 2009

Table 9.5. Euclidean distances from *Moby Dick* based on 578 features

Rank	Author	Title	Distance
0	Melville, Herman	*Moby-Dick; or, The Whale*	0
1	Melville, Herman	*Omoo: A Narrative of Adventures in the South Seas*	0.057784134
2	Melville, Herman	*Mardi and a Voyage Thither*	0.073077699
3	Cooper, James	*The Crater; or, Vulcan's Peak: A Tale of the Pacific*	0.073798396
4	Cooper, James	*The Sea Lions; or, The Lost Sealers*	0.08339971
5	Melville, Herman	*Typee: A Peep at Polynesian Life*	0.101393295
6	Ballantyne, Robert	*The Coral Island: A Tale of the Pacific Ocean*	0.117425226
7	Poe, Edgar Allan	*The Narrative of Arthur Gordon Pym of Nantucket*	0.13125092
8	Stevenson, Robert	*Island Nights' Entertainments*	0.146050418
9	Williams, William	*The Journal of Llewellin Penrose, a Seaman*	0.176751153
10	Payn, James	*A Prince of the Blood*	0.18207467

Jockers 2013

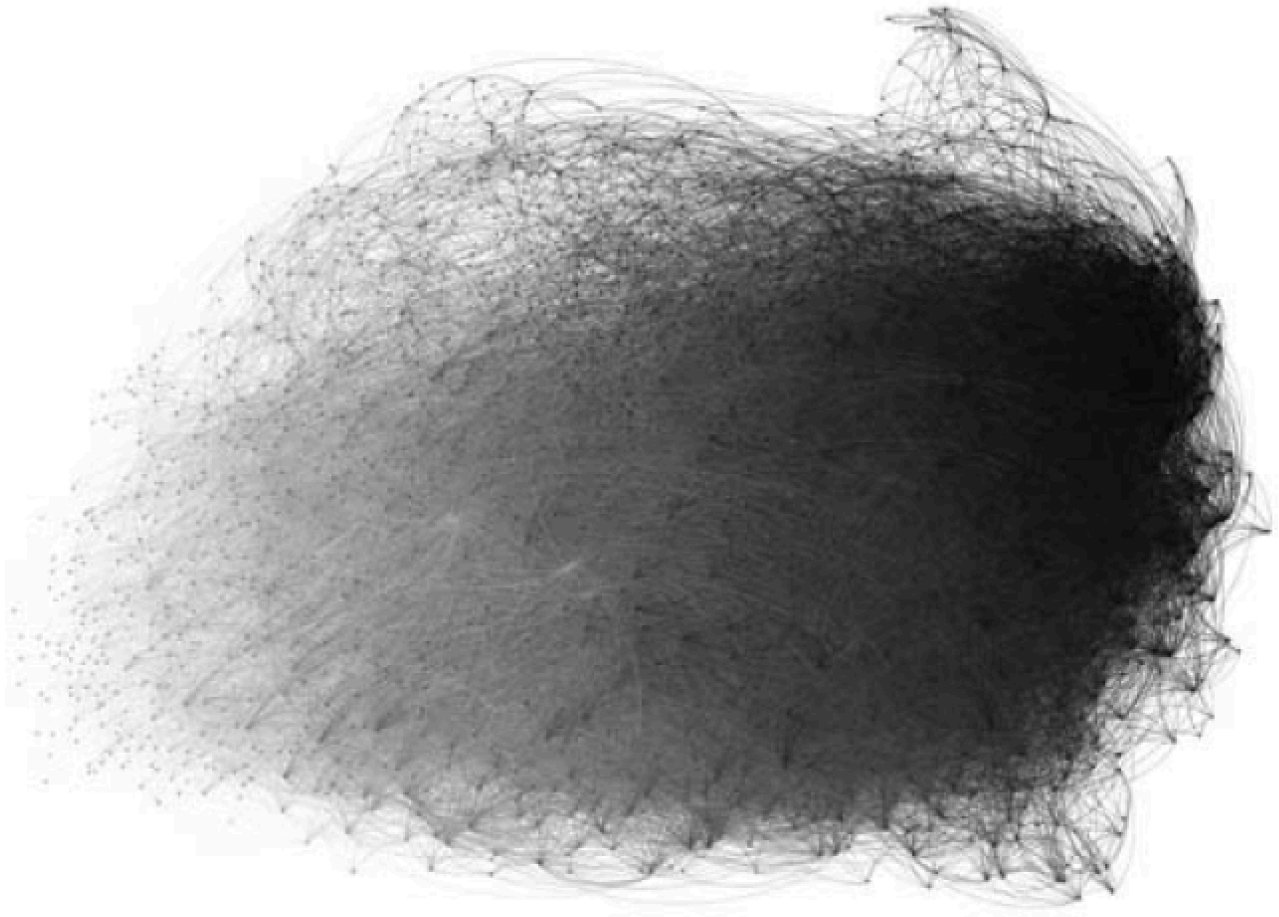


Figure 9.3. Nineteenth-century novel network with date shading

Jockers 2013

4.6 Distinctive words in novels compared to other kinds of fiction across four categories (moving clockwise from the top left): "discrepancy," "insight," "negation," and "tentativeness."

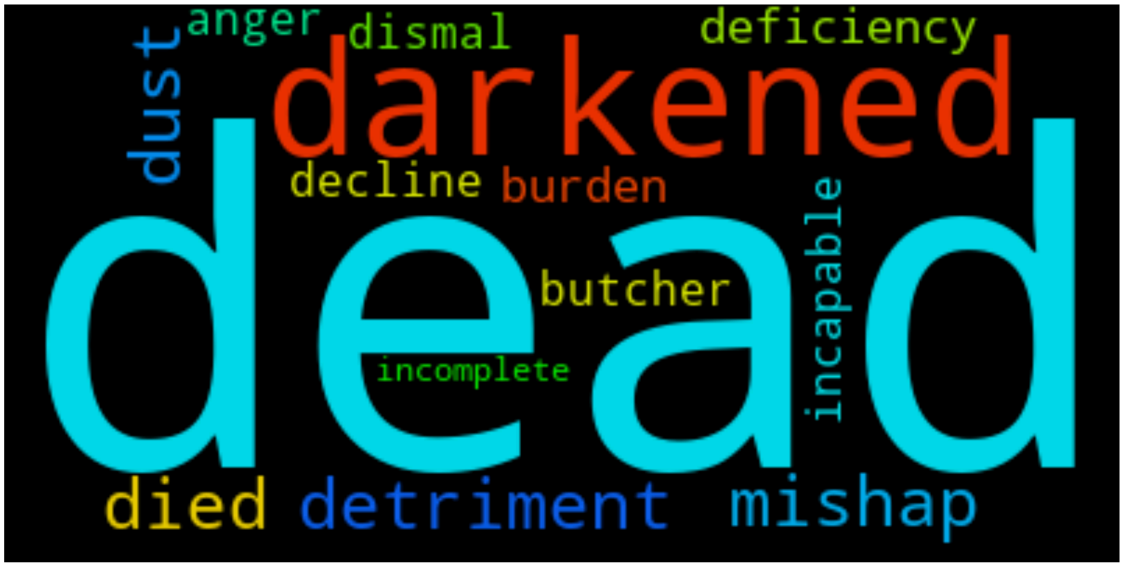
Piper 2019

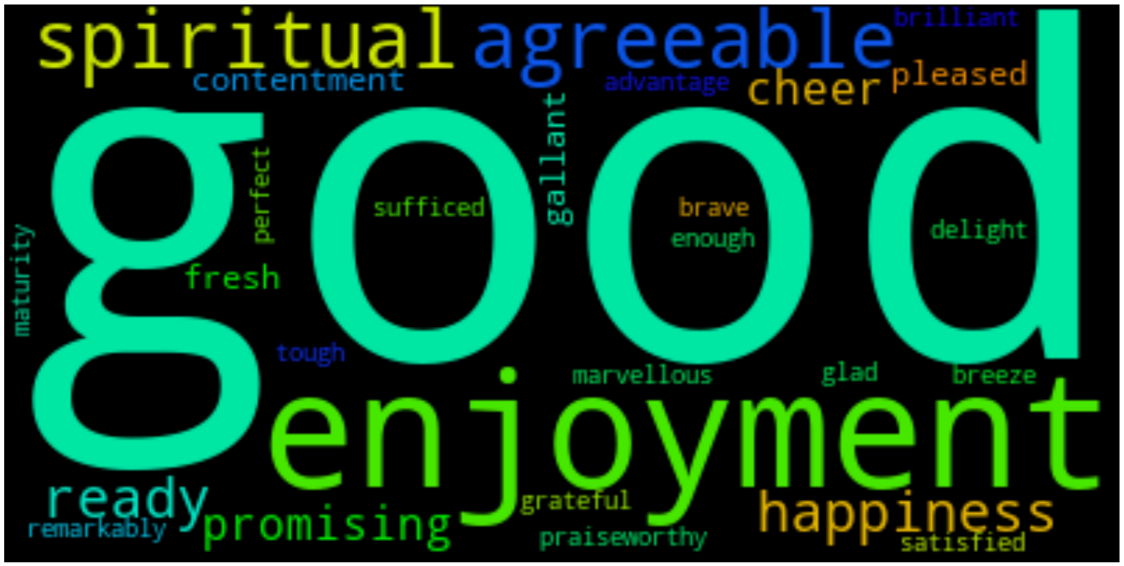
Python WordCloud

- You can install via Jupyter:
  `!pip install wordcloud`
- Comes with tools to tokenize and remove stopwords and produce representation of most frequent words.
- Can also be used with custom list of frequencies for other types of data and objects.

Word Clouds

Hawthorne's
The Scarlet Letter,
Segment 4
Bing negative and
Bing positive words





In [1]:
```
# import required libraries
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from nltk.tokenize import sent_tokenize, word_tokenize
```

In [2]:
```
# load Bing sentiment lexicons for positive and negative words
bing_positive = open('data/lexicons/positive-words.txt').read()
bing_negative = open('data/lexicons/negative-words.txt').read()

# convert to lists
bing_positive = bing_positive.split('\n')
bing_negative = bing_negative.split('\n')
```

In [3]:
```
# Read and Convert Hawthorne's The Scarlet Letter into 100 Units
input_text = "data/Novel450/EN_1850_Hawthorne,Nathaniel_TheScarletLetter_Novel.txt"

# make into a collection of 100 documents
raw_text = open(input_text).read()
tokens = word_tokenize(raw_text)
segment_length =  int(len(tokens)/100)
collection = list()

for i in range(100):
        segment = tokens[segment_length*i:segment_length*(i+1)]
        collection.append(' '.join(segment))
```

In [4]:
```
tokens = word_tokenize(collection[6])
types = set(tokens)

positive_terms=list()
for word in types:
    if word in bing_positive:
        positive_terms.append((word,tokens.count(word)))

negative_terms=list()
for word in types:
    if word in bing_negative:
        negative_terms.append((word,tokens.count(word)))
```

In [5]:
```
positive_wordcloud = WordCloud(max_words=100).generate_from_frequencies(positive_terms)
fig = plt.figure(figsize=(35, 20), dpi=75)
plt.imshow(positive_wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```