notebook

April 27, 2025

0.1 1. Tools for text processing

What are the most frequent words in Herman Melville's novel, Moby Dick, and how often do they occur?

In this notebook, we'll scrape the novel Moby Dick from the website Project Gutenberg (which contains a large corpus of books) using the Python package requests. Then we'll extract words from this web data using BeautifulSoup. Finally, we'll dive into analyzing the distribution of words using the Natural Language ToolKit (nltk) and Counter.

The Data Science pipeline we'll build in this notebook can be used to visualize the word frequency distributions of any novel that you can find on Project Gutenberg. The natural language processing tools used here apply to much of the data that data scientists encounter as a vast proportion of the world's data is unstructured data and includes a great deal of text.

Let's start by loading in the three main Python packages we are going to use.

[2]:
```python
# Importing requests, BeautifulSoup, nltk, and Counter
import requests
from bs4 import BeautifulSoup
import nltk
from collections import Counter
```

0.2 2. Request Moby Dick

To analyze Moby Dick, we need to get the contents of Moby Dick from somewhere. Luckily, the text is freely available online at Project Gutenberg as an HTML file: https://www.gutenberg.org/files/2701/2701-h/2701-h.htm .

Note that HTML stands for Hypertext Markup Language and is the standard markup language for the web.

To fetch the HTML file with Moby Dick we're going to use the request package to make a GET request for the website, which means we're getting data from it. This is what you're doing through a browser when visiting a webpage, but now we're getting the requested page directly into Python instead.

[4]:
```python
# Getting the Moby Dick HTML
r = requests.get('https://s3.amazonaws.com/assets.datacamp.com/production/
 ↪project_147/datasets/2701-h.htm')
```

```python
# Setting the correct text encoding of the HTML page
r.encoding = 'utf-8'

# Extracting the HTML from the request object
html = r.text

# Printing the first 2000 characters in html
print(html[0:2000])
```

```
DEBUG:urllib3.connectionpool:Starting new HTTPS connection (1): s3.amazonaws.com
DEBUG:urllib3.connectionpool:https://s3.amazonaws.com:443 "GET
/assets.datacamp.com/production/project_147/datasets/2701-h.htm HTTP/1.1" 200
1501037
```

```
<?xml version="1.0" encoding="utf-8"?>

<!DOCTYPE html
   PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd" >

<html xmlns="http://www.w3.org/1999/xhtml" lang="en">
  <head>
    <title>
      Moby Dick; Or the Whale, by Herman Melville
    </title>
    <style type="text/css" xml:space="preserve">

    body { background:#faebd0; color:black; margin-left:15%; margin-right:15%;
text-align:justify }
    P { text-indent: 1em; margin-top: .25em; margin-bottom: .25em; }
    H1,H2,H3,H4,H5,H6 { text-align: center; margin-left: 15%; margin-right: 15%;
}
    hr  { width: 50%; text-align: center;}
    .foot { margin-left: 20%; margin-right: 20%; text-align: justify; text-
indent: -3em; font-size: 90%; }
    blockquote {font-size: 100%; margin-left: 0%; margin-right: 0%;}
    .mynote    {background-color: #DDE; color: #000; padding: .5em; margin-left:
10%; margin-right: 10%; font-family: sans-serif; font-size: 95%;}
    .toc       { margin-left: 10%; margin-bottom: .75em;}
    .toc2      { margin-left: 20%;}
    div.fig    { display:block; margin:0 auto; text-align:center; }
    div.middle { margin-left: 20%; margin-right: 20%; text-align: justify; }
    .figleft   {float: left; margin-left: 0%; margin-right: 1%;}
    .figright  {float: right; margin-right: 0%; margin-left: 1%;}
    .pagenum   {display:inline; font-size: 70%; font-style:normal;
               margin: 0; padding: 0; position: absolute; right: 1%;
               text-align: right;}
    pre        { font-family: times new roman; font-size: 100%; margin-left:
```

```
10%;}

    table      {margin-left: 10%;}

a:link {color:blue;
                text-decoration:none}
link {color:blue;
                text-decoration:none}
a:visited {color:blue;
                text-decoration:none}
a:hover {color:red}

</style>
  </head>
  <body>
<pre xml:space="preserve">

The Project Gutenberg EBook of Moby Dick; or The Whale, by Herman Melville

This eBook is for the use of anyone anywh
```

0.3 3. Get the text from the HTML

This HTML is not quite what we want. However, it does contain what we want: the text of Moby Dick. What we need to do now is wrangle this HTML to extract the text of the novel. For this we'll use the package BeautifulSoup.

Firstly, a word on the name of the package: Beautiful Soup? In web development, the term "tag soup" refers to structurally or syntactically incorrect HTML code written for a web page. What Beautiful Soup does best is to make tag soup beautiful again and to extract information from it with ease! In fact, the main object created and queried when using this package is called BeautifulSoup.

[6]:
```python
# Creating a BeautifulSoup object from the HTML
soup = BeautifulSoup(html, "html.parser")

# Getting the text out of the soup
text = soup.get_text()

# Printing out text between characters 32000 and 34000
print(text[32000:34000])
```

```
r which the beech tree
        extended its branches." -Darwin's Voyage of a Naturalist.


        "'Stern all!' exclaimed the mate, as upon turning his head, he saw the
        distended jaws of a large Sperm Whale close to the head of the boat,
        threatening it with instant destruction;-'Stern all, for your
        lives!'" -Wharton the Whale Killer.
```

"So be cheery, my lads, let your hearts never fail, While the bold harpooneer is striking the whale!" -Nantucket Song.

"Oh, the rare old Whale, mid storm and gale
In his ocean home will be
A giant in might, where might is right,
And King of the boundless sea."
-Whale Song.

CHAPTER 1. Loomings.

Call me Ishmael. Some years ago-never mind how long precisely-having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail about a little and see the watery part of the world. It is a way I have of driving off the spleen and regulating the circulation. Whenever I find myself growing grim about the mouth; whenever it is a damp, drizzly November in my soul; whenever I find myself involuntarily pausing before coffin warehouses, and bringing up the rear of every funeral I meet; and especially whenever my hypos get such an upper hand of me, that it requires a strong moral principle to prevent me from deliberately stepping into the street, and methodically knocking people's hats off-then, I account it high time to get to sea as soon as I can. This is my substitute for pistol and ball. With a philosophical flourish Cato throws himself upon his sword; I quietly take to the ship. There is nothing surprising in this. If they but knew it, almost all men in their degree, some time or other, cherish very nearly the same feelings towards the ocean with me.

Ther

0.4 4. Extract the words

We now have the text of the novel! There is some unwanted stuff at the start and some unwanted stuff at the end. We could remove it, but this content is so much smaller in amount than the text of Moby Dick that, to a first approximation, it is okay to leave it in.

Now that we have the text of interest, it's time to count how many times each word appears, and for this we'll use nltk – the Natural Language Toolkit. We'll start by tokenizing the text, that is, remove everything that isn't a word (whitespace, punctuation, etc.) and then split the text into a list of words.

[8]:
```
# Creating a tokenizer
tokenizer = nltk.tokenize.RegexpTokenizer('\w+')

# Tokenizing the text
tokens = tokenizer.tokenize(text)

# Printing out the first 8 words / tokens
tokens[0:8]
```

0.5 5. Make the words lowercase

OK! We're nearly there. Note that in the above 'Or' has a capital 'O' and that in other places it may not, but both 'Or' and 'or' should be counted as the same word. For this reason, we should build a list of all words in Moby Dick in which all capital letters have been made lower case.

[10]:
```
# Create a list called words containing all tokens transformed to lower-case
words = [token.lower() for token in tokens]

# Printing out the first 8 words / tokens
words[:8]
```

0.6 6. Load in stop words

It is common practice to remove words that appear a lot in the English language such as 'the', 'of' and 'a' because they're not so interesting. Such words are known as stop words. The package nltk includes a good list of stop words in English that we can use.

[ ]:
```
# Getting the English stop words from nltk
sw = ...

# Printing out the first eight stop words
# ... YOUR CODE FOR TASK 6 ...
```

0.7 7. Remove stop words in Moby Dick

We now want to create a new list with all words in Moby Dick, except those that are stop words (that is, those words listed in sw).

```
[ ]: # Create a list words_ns containing all words that are in words but not in sw
     # ... YOUR CODE FOR TASK 7 ...

     # Printing the first 5 words_ns to check that stop words are gone
     # ... YOUR CODE FOR TASK 7 ...
```

0.8 8. We have the answer

Our original question was:

What are the most frequent words in Herman Melville's novel Moby Dick and how often do they occur?

We are now ready to answer that! Let's answer this question using the Counter class we imported earlier.

```
[ ]: # Initialize a Counter object from our processed list of words
     count = ...

     # Store 10 most common words and their counts as top_ten
     top_ten = ...

     # Print the top ten words and their counts
```

0.9 9. The most common word

Nice! Using our variable top_ten, we now have an answer to our original question.

The natural language processing skills we used in this notebook are also applicable to much of the data that Data Scientists encounter as the vast proportion of the world's data is unstructured data and includes a great deal of text.

So, what word turned out to (not surprisingly) be the most common word in Moby Dick?

```
[ ]: # What's the most common word in Moby Dick?
     most_common_word = 'horse'
```