Natural Language Processing (NLP)
Dr Rob Cooke
Section 5: 10th May 2021
Lo Chinen Ed (MSc)

Introduction

This section focuses on processing of text. That is, rather than elaborate data like numbers and tables we will be looking at text data.

The data we are using are excerpts from a popular movie website. The reviews are for a Disney book in the UK. In this case, the reviews are of Disney's live-action remake of the classic Cinderella.

The intention the continuation is to take data like information we are trying to model and analyse in the form of text. Of course, there are many different ways we can do this. Some of the ways we will be looking at are:

- Bag-of-words - looking for patterns and word relationships.
- Web sites - things social media, review websites etc.
- Twitter.
- Text data from customer surveys - for example feedback and/or partial expertise
- Text based chat exchanges over the internet.

This workshop is focussed on bag-of-words and rather than text production or transformation.

Instructions for Students

Unlike previous sections, the instructions in these notes are not separated out neatly into code for the first lesson or one particular section of the lesson. The section is indicated in the codeblock labels with a **Student Task** indicator.

In working through the following notebook, please do the following:

- Create and copy a notebook to your own area.
- Enter all of the **Python code** into the notebook into **code** blocks in your notebook.
- Run the code and see the output and results.
- **Do not** change the code in the **Python code** blocks.
- Read the **Student task** instructions and perform the task in your notebook. Sometimes, you will need to change the code in the **Student task** blocks. Sometimes, you will need to write your own code in the **Student task** blocks.
- Try **step** and further code blocks to experiment with the commands and to see other things.
- Use the **help** and **?** commands to find out more about the commands you are using.
- **Write clear** notes in your notebook about what you have done and what you learnt from the lesson.

The notebook should be the result of your lesson notebook and **may** be different to the one shown here because they will contain your own code and notes.

1 Importing Libraries and Loading Data

Unlike previous sections, we will be using a lot of different libraries. In this case, we will be using the Natural Language Toolkit (NLTK) and its wordcloud library. We will also be using the Pandas library.

```
!pip install -U spacy
!pip install -U spacy-lookups-data
!python -m spacy download en_core_web_sm
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import nltk, wordcloud, string, re
```

We will load the data from a .csv file into a Pandas dataframe.

```
data = pd.read_csv('data/disney_princess_reviews.csv')
print("Dimensions: ",data.shape,"\n")
print("Column Heads: ",data.columns,"\n")
print("Data Types:\n",data.dtypes)
```

This is fairly easily transferred to Pandas data frame for review and further manipulation:

```
data.head(10)
```

	reviewer	reviewdate	reviewtext	sentiment
0	Lovered	27 March with one adventure in learning Mandy	As a disabled person it is always hard to keep	Pos
1	Cinderella	26 APRIL 2020	Amazing movie. Very touching. Very well done.	Pos
2	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
3	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
4	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
5	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
6	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
7	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
8	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos
9	Cinderella	25 April 2020	Very good movie. Good for the family. Great	Pos

```
data.tail(10)
```

	reviewer	reviewdate	reviewtext	sentiment
1000	Cinderella	1 April 2020	Lovely time watching	Pos
1001	Cinderella	1 April 2020	Amazing cinderella version this is a must watch	Pos
1002	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos
1003	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Neg
1004	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos
1005	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos
1006	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos
1007	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos
1008	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos
1009	Cinderella	1 April 2020	Very good movie. Good for the family. Great	Pos

1000 rows × 4 columns

We can see that 1000 rows of data. Each row is a review from a different customer. The reviews are in a form into one cell.

If bag-of-words is playing the role of a Rosh's representation that enables tentative "statistical" Machine Learning tools to model the data, then we need to do some processing of the text. We need to do some cleaning and transformation of the text into a form that can be processed. The common operations on the text to start the information extraction process are:

- Tokenisation - breaking up the text into individual words.
- Stemming - reducing words to their root form.
- Stop words - removing words that are not important to the analysis.

The data is stored as data.reviewtext. The first is a single list of all words in the evaluations of the review:

```
data.reviewtext[0:10]
```

```
0    As a disabled person it is always hard to keep...
1    Amazing movie. Very touching. Very well done.
2    Very good movie. Good for the family. Great ...
3    Very good movie. Good for the family. Great ...
4    Very good movie. Good for the family. Great ...
5    Very good movie. Good for the family. Great ...
6    Very good movie. Good for the family. Great ...
7    Very good movie. Good for the family. Great ...
8    Very good movie. Good for the family. Great ...
9    Very good movie. Good for the family. Great ...
Name: reviewtext, dtype: object
```

2 Parts of word Tokenisation

Tokenisation is the process of breaking the sentence of text and/or the data set. In the following code, we have created the corpus so that they are separated. Many corpus forms.

First, I count the total number of words and sort corpus. I'll, then, count the columns in the above table.

```
text_corpus = data.reviewtext.sum(axis=0)
text_corpus[0:100]
```

```
'As a disabled person it is always hard to keep Amazing movie. Very touching. Very well done. Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great Very good movie. Good for the family. Great '
```

Then, split that into by frequency:

```
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text_corpus)
print("Number of tokens: ",len(tokens),"\n")
print("First 100 tokens: ",tokens[0:100])
```

```
Number of tokens: 12 ['As', 'a', 'disabled', 'person', 'it', 'is', 'always', 'hard', 'to', 'keep', 'Amazing', 'movie', '.', 'Very', 'touching', '.', 'Very', 'well', 'done', '.', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great', 'Very', 'good', 'movie', '.', 'Good', 'for', 'the', 'family', '.', 'Great']
```

Then, count the frequency:

```
from collections import Counter, defaultdict
counts = Counter(tokens)
print("Number of words: ",len(counts),"\n")
print("Top 100 words: ",counts.most_common(100))
```

```
Number of words: 12 Top 100 words: [('.', 12), ('Great', 12), ('family', 12), ('the', 12), ('for', 12), ('Good', 12), ('movie', 12), ('good', 12), ('Very', 12), ('', 12), ('', 12), ('', 12)]
```

Notice that the top 100 words are all stop words and punctuation. We need to remove these.

```
stopwords = set(nltk.corpus.stopwords.words('english'))
print("Number of stopwords: ",len(stopwords),"\n")
print("Stopwords: ",stopwords)
```

```
Number of stopwords: 179 Stopwords: {'ourselves', 'hadn't', 'yourself', 'ma', 'ain', 'hasn't', 'wouldn't', 'shouldn't', 'wasn't', 'hadn't', 'weren't', 'mustn't', 'shan't', 'needn't', 'doesn't', 'didn't', 'isn't', 'aren't', 'won't', 'don't', 'haven't', 'couldn't', 'might
```

Topic 0:
[0]Crash, [0]Bug, [0]Fix, [0]Linux[0]Kernel, [0]Debian, [0]Ubuntu, [0]Update, [0]Update, [0]Ubuntu, [0]Ok, [0]Bug, [0]Update, [0]Ubuntu, [0]Ok, [0]Ok

Topic 1:
[1]Crash, [1]Bug, [1]Fix, [1]Linux[1]Kernel, [1]Debian, [1]Ubuntu, [1]Update, [1]Update, [1]Ubuntu, [1]Ok, [1]Bug, [1]Update, [1]Ubuntu, [1]Ok, [1]Ok

Topic 2:
[2]Crash, [2]Bug, [2]Fix, [2]Linux[2]Kernel, [2]Debian, [2]Ubuntu, [2]Update, [2]Update, [2]Ubuntu, [2]Ok, [2]Bug, [2]Update, [2]Ubuntu, [2]Ok, [2]Ok

Topic 3:
[3]Crash, [3]Bug, [3]Fix, [3]Linux[3]Kernel, [3]Debian, [3]Ubuntu, [3]Update, [3]Update, [3]Ubuntu, [3]Ok, [3]Bug, [3]Update, [3]Ubuntu, [3]Ok, [3]Ok

Topic 4:
[4]Crash, [4]Bug, [4]Fix, [4]Linux, [4]Debian, [4]Ubuntu, [4]Update, [4]Update, [4]Ubuntu, [4]Ok, [4]Bug, [4]Update, [4]Ubuntu, [4]Ok, [4]Ok

Topic 5:
[5]Crash, [5]Bug, [5]Fix, [5]Linux, [5]Debian, [5]Ubuntu, [5]Update, [5]Update, [5]Ubuntu, [5]Ok, [5]Bug, [5]Update, [5]Ubuntu, [5]Ok, [5]Ok

Topic 6:
[6]Crash, [6]Bug, [6]Fix, [6]Linux, [6]Debian, [6]Ubuntu, [6]Update, [6]Update, [6]Ubuntu, [6]Ok, [6]Bug, [6]Update, [6]Ubuntu, [6]Ok, [6]Ok

Topic 7:
[7]Crash, [7]Bug, [7]Fix, [7]Linux, [7]Debian, [7]Ubuntu, [7]Update, [7]Update, [7]Ubuntu, [7]Ok, [7]Bug, [7]Update, [7]Ubuntu, [7]Ok, [7]Ok

Topic 8:
[8]Crash, [8]Bug, [8]Fix, [8]Linux, [8]Debian, [8]Ubuntu, [8]Update, [8]Update, [8]Ubuntu, [8]Ok, [8]Bug, [8]Update, [8]Ubuntu, [8]Ok, [8]Ok

Topic 9:
[9]Crash, [9]Bug, [9]Fix, [9]Linux, [9]Debian, [9]Ubuntu, [9]Update, [9]Update, [9]Ubuntu, [9]Ok, [9]Bug, [9]Update, [9]Ubuntu, [9]Ok, [9]Ok

These Topics are very similar to each other. But for example, Topic 0 and Topic 1 are both about crash of OSes.

Topic 7: account, open, opened, closed, close, current, opening, reason, direct, send

It seems to me that there is a significant cluster of sentences that relate to account opening and closing. These are key customer service processes for a bank. Particularly, account opening reasons have to account for the bank. So it makes a lot sense to create a topic for sentences in this cluster.

Topic 2: card, debit, visa, acct, address, pin, received, blocked, cards, bank This theme seems to be about **non-debit cards**. Topic 2 is very similar to Topic 7 in terms of its keywords. However, Topic 2 is more focused on non-debit cards. Topic 7 seems to be more general. I think it would be a good idea to investigate this. It seems also that the word cloud **Visa** is a reason that cards are not being **blocked**.

Topic 17: phone, number, hold, minutes, call, security, hour, contact, department, hours I personally find this topic very interesting. It seems to be about the **customer service** of a bank. Customers call a bank when they have a problem. Customers want to talk to a **bank** as quick as that and they do **phone** the bank. Here we can see **security** of customs with contacting the bank **department**.

Appendix: Experiment with different values of λ (lamda) and λ (lag-weight) in the code blocks above. The number of themes will change. Examine the word clouds, the themes and also topic information in order to appreciate the data-driven nature. Different settings may produce quite different results and hence different insights.