Lab1-Assignment

This notebook describes the assignment for Lab 1 of the text mining course.

Points: each exercise is prefixed with the number of points you can obtain for the exercise.

We assume you have worked through the following notebooks:

- **Lab1.1-introduction**
- **Lab1.2-introduction-to-NLTK**
- **Lab1.3-introduction-to-spaCy**

In this assignment, you will process an English text (**Lab1-apple-samsung-example.txt**) with both NLTK and spaCy and discuss the similarities and differences.

Credits

The notebooks in this block have been originally created by Marten Postma. Adaptations were made by Filip Ilievski.

Tip: how to read a file from disk

Let's open the file **Lab1-apple-samsung-example.txt** from disk.

```
from pathlib import Path

cur_dir = Path().resolve() # this should provide you with the folder
in which this notebook is placed
path_to_file = Path.joinpath(cur_dir, 'Lab1-apple-samsung-
example.txt')
print(path_to_file)
print('does path exist? ->', Path.exists(path_to_file))
```

```
C:\Users\Hackerman\Desktop\(34)Text Mining for AI\repo\Lab1-apple-
samsung-example.txt
does path exist? -> True
```

If the output from the code cell above states that **does path exist? -> False**, please check that the file **Lab1-apple-samsung-example.txt** is in the same directory as this notebook.

```
with open(path_to_file) as infile:
    text = infile.read()

print('number of characters', len(text))
```

```
number of characters 1142
```

[total points: 4] Exercise 1: NLTK

In this exercise, we use NLTK to apply **Part-of-speech (POS) tagging**, **Named Entity Recognition (NER)**, and **Constituency parsing**. The following code snippet already performs sentence splitting and tokenization.

```
import pandas as pd
import nltk
from nltk.tokenize import sent_tokenize
from nltk import word_tokenize

sentences_nltk = sent_tokenize(text)

tokens_per_sentence = []
for sentence_nltk in sentences_nltk:
    sent_tokens = word_tokenize(sentence_nltk)
    tokens_per_sentence.append(sent_tokens)
```

We will use lists to keep track of the output of the NLP tasks. We can hence inspect the output for each task using the index of the sentence.

```
sent_id = 1
print('SENTENCE', sentences_nltk[sent_id])
print('TOKENS', tokens_per_sentence[sent_id])
```

```
SENTENCE The six phones and tablets affected are the Galaxy S III,
running the new Jelly Bean system, the Galaxy Tab 8.9 Wifi tablet, the
Galaxy Tab 2 10.1, Galaxy Rugby Pro and Galaxy S III mini.
TOKENS ['The', 'six', 'phones', 'and', 'tablets', 'affected', 'are',
'the', 'Galaxy', 'S', 'III', ',', 'running', 'the', 'new', 'Jelly',
'Bean', 'system', ',', 'the', 'Galaxy', 'Tab', '8.9', 'Wifi',
'tablet', ',', 'the', 'Galaxy', 'Tab', '2', '10.1', ',', 'Galaxy',
'Rugby', 'Pro', 'and', 'Galaxy', 'S', 'III', 'mini', '.']
```

[point: 1] Exercise 1a: Part-of-speech (POS) tagging

Use `nltk.pos_tag` to perform part-of-speech tagging on each sentence.

Use `print` to **show** the output in the notebook (and hence also in the exported PDF!).

```
pos_tags_per_sentence = []
for tokens in tokens_per_sentence:
    tokens_tagged = nltk.pos_tag(tokens)
    pos_tags_per_sentence.append(tokens_tagged)
print(pos_tags_per_sentence[sent_id])
```

```
[('The', 'DT'), ('six', 'CD'), ('phones', 'NNS'), ('and', 'CC'),
('tablets', 'NNS'), ('affected', 'VBN'), ('are', 'VBP'), ('the',
'DT'), ('Galaxy', 'NNP'), ('S', 'NNP'), ('III', 'NNP'), (',', ','),
('running', 'VBG'), ('the', 'DT'), ('new', 'JJ'), ('Jelly', 'NNP'),
('Bean', 'NNP'), ('system', 'NN'), (',', ','), ('the', 'DT'),
('Galaxy', 'NNP'), ('Tab', 'NNP'), ('8.9', 'CD'), ('Wifi', 'NNP'),
```

```
('tablet', 'NN'), (',', ','), ('the', 'DT'), ('Galaxy', 'NNP'),
('Tab', 'NNP'), ('2', 'CD'), ('10.1', 'CD'), (',', ','), ('Galaxy',
'NNP'), ('Rugby', 'NNP'), ('Pro', 'NNP'), ('and', 'CC'), ('Galaxy',
'NNP'), ('S', 'NNP'), ('III', 'NNP'), ('mini', 'NN'), ('.', '.')]

pd.DataFrame(pos_tags_per_sentence).rename("sentence
{}".format).T.fillna('-')
```

	sentence 0	sentence 1
0	(https, NN)	(The, DT)
1	(:, :)	(six, CD)
2	(//www.telegraph.co.uk/technology/apple/970271...	(phones, NNS)
3	(Documents, NNS)	(and, CC)
4	(filed, VBN)	(tablets, NNS)
5	(to, TO)	(affected, VBN)
6	(the, DT)	(are, VBP)
7	(San, NNP)	(the, DT)
8	(Jose, NNP)	(Galaxy, NNP)
9	(federal, JJ)	(S, NNP)
10	(court, NN)	(III, NNP)
11	(in, IN)	(,, ,)
12	(California, NNP)	(running, VBG)
13	(on, IN)	(the, DT)
14	(November, NNP)	(new, JJ)
15	(23, CD)	(Jelly, NNP)
16	(list, NN)	(Bean, NNP)
17	(six, CD)	(system, NN)
18	(Samsung, NNP)	(,, ,)
19	(products, NNS)	(the, DT)

20		(running, VBG)	(Galaxy, NNP)	
21		(the, DT)	(Tab, NNP)	
22		(``, ``)	(8.9, CD)	
23		(Jelly, RB)	(Wifi, NNP)	
24		(Bean, NNP)	(tablet, NN)	
25		('', '')	(,, ,)	
26		(and, CC)	(the, DT)	
27		(``, ``)	(Galaxy, NNP)	
28		(Ice, NNP)	(Tab, NNP)	
29		(Cream, NNP)	(2, CD)	
30		(Sandwich, NNP)	(10.1, CD)	
31		('', '')	(,, ,)	
32		(operating, VBG)	(Galaxy, NNP)	
33		(systems, NNS)	(Rugby, NNP)	
34		(,, ,)	(Pro, NNP)	
35		(which, WDT)	(and, CC)	
36		(Apple, NNP)	(Galaxy, NNP)	
37		(claims, VBZ)	(S, NNP)	
38		(infringe, VB)	(III, NNP)	
39		(its, PRP$)	(mini, NN)	
40		(patents, NNS)	(., .)	
41		(., .)	-	

sentence 2	sentence 3	sentence 4	sentence 5

0	(Apple, NNP)	(In, IN)	(Samsung, NNP)	(A, DT)
1	(stated, VBD)	(August, NNP)	(,, ,)	(similar, JJ)
2	(it, PRP)	(,, ,)	(which, WDT)	(case, NN)
3	(had, VBD)	(Samsung, NNP)	(is, VBZ)	(in, IN)
4	(â€œacted, VBN)	(lost, VBD)	(the, DT)	(the, DT)
5	(quickly, RB)	(a, DT)	(world, NN)	(UK, NNP)
6	(and, CC)	(US, NNP)	('s, POS)	(found, VBD)
7	(diligently, RB)	(patent, NN)	(top, JJ)	(in, IN)
8	('', '')	(case, NN)	(mobile, NN)	(Samsung, NNP)
9	(in, IN)	(to, TO)	(phone, NN)	('s, POS)
10	(order, NN)	(Apple, NNP)	(maker, NN)	(favour, NN)
11	(to, TO)	(and, CC)	(,, ,)	(and, CC)
12	(``, ``)	(was, VBD)	(is, VBZ)	(ordered, VBD)
13	(determine, VB)	(ordered, VBN)	(appealing, VBG)	(Apple, NNP)
14	(that, IN)	(to, TO)	(the, DT)	(to, TO)
15	(these, DT)	(pay, VB)	(ruling, NN)	(publish, VB)
16	(newly, RB)	(its, PRP$)	(., .)	(an, DT)
17	(released, VBN)	(rival, JJ)	-	(apology, NN)
18	(products, NNS)	($, $)	-	(making, VBG)
19	(do, VBP)	(1.05bn, CD)	-	(clear, JJ)
20	(infringe, VB)	((, ()	-	(that, IN)
21	(many, JJ)	(Â£0.66bn, NN)	-	(the, DT)
22	(of, IN)	(),))	-	(South, JJ)
23	(the, DT)	(in, IN)	-	(Korean, JJ)
24	(same, JJ)	(damages, NNS)	-	(firm, NN)

```
25     (claims, NNS)        (for, IN)                  -        (had,
VBD)
26     (already, RB)   (copying, VBG)                  -         (not,
RB)
27   (asserted, VBN)  (features, NNS)                  -     (copied,
VBN)
28           (by, IN)         (of, IN)                  -       (its,
PRP$)
29      (Apple, NNP)        (the, DT)                  -        (iPad,
NN)
30             (., .)       (iPad, NN)                  -       (when,
WRB)
31           ('', '')        (and, CC)                  -  (designing,
VBG)
32                  -     (iPhone, NN)                  -       (its,
PRP$)
33                  -         (in, IN)                  -         (own,
JJ)
34                  -      (its, PRP$)                  -    (devices,
NNS)
35                  -    (Galaxy, NNP)                  -            (.,
.)
36                  -      (range, NN)                  -
-
37                  -         (of, IN)                  -
-
38                  -   (devices, NNS)                  -
-
39                  -           (., .)                  -
-
40                  -                -                  -
-
41                  -                -                  -
-
```

[point: 1] Exercise 1b: Named Entity Recognition (NER)

Use `nltk.chunk.ne_chunk` to perform Named Entity Recognition (NER) on each sentence.

Use `print` to **show** the output in the notebook (and hence also in the exported PDF!).

```
ner_tags_per_sentence = []
for sent in pos_tags_per_sentence:
    sent_named = nltk.chunk.ne_chunk(sent)
    ner_tags_per_sentence.append(sent_named)
print(ner_tags_per_sentence[0])
```

```
(S
  https/NN
  :/:
```

```
  //www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html/JJ
  Documents/NNS
  filed/VBN
  to/TO
  the/DT
  (ORGANIZATION San/NNP Jose/NNP)
  federal/JJ
  court/NN
  in/IN
  (GPE California/NNP)
  on/IN
  November/NNP
  23/CD
  list/NN
  six/CD
  (ORGANIZATION Samsung/NNP)
  products/NNS
  running/VBG
  the/DT
  ``/``
  Jelly/RB
  (GPE Bean/NNP)
  ''/''
  and/CC
  ``/``
  Ice/NNP
  Cream/NNP
  Sandwich/NNP
  ''/''
  operating/VBG
  systems/NNS
  ,/,
  which/WDT
  (PERSON Apple/NNP)
  claims/VBZ
  infringe/VB
  its/PRP$
  patents/NNS
  ./.)

pd.DataFrame([[(tok, tok.label()) for tok in sent if isinstance(tok,
nltk.tree.tree.Tree)]
              for sent in ner_tags_per_sentence]).rename("sentence
{}".format).T.fillna('-')

                                    sentence 0  \
0  ([(San, NNP), (Jose, NNP)], ORGANIZATION)
1                  ([(California, NNP)], GPE)
2           ([(Samsung, NNP)], ORGANIZATION)
3                       ([(Bean, NNP)], GPE)
```

```
4                   ([(Apple, NNP)], PERSON)
5                                          -
6                                          -

                                        sentence 1  \
0                    ([(Galaxy, NNP)], ORGANIZATION)
1               ([(Jelly, NNP), (Bean, NNP)], PERSON)
2                    ([(Galaxy, NNP)], ORGANIZATION)
3                    ([(Galaxy, NNP)], ORGANIZATION)
4  ([(Galaxy, NNP), (Rugby, NNP), (Pro, NNP)], PE...
5                  ([(Galaxy, NNP), (S, NNP)], PERSON)
6                                                  -

                  sentence 2                        sentence 3  \
0  ([(Apple, NNP)], PERSON)           ([(August, NNP)], GPE)
1  ([(Apple, NNP)], PERSON)       ([(Samsung, NNP)], PERSON)
2                         -               ([(US, NNP)], GSP)
3                         -            ([(Apple, NNP)], GPE)
4                         -     ([(iPad, NN)], ORGANIZATION)
5                         -   ([(iPhone, NN)], ORGANIZATION)
6                         -           ([(Galaxy, NNP)], GPE)

                 sentence 4                                 sentence 5
0  ([(Samsung, NNP)], GPE)                ([(UK, NNP)], ORGANIZATION)
1                         -                    ([(Samsung, NNP)], GPE)
2                         -                  ([(Apple, NNP)], PERSON)
3                         -  ([(South, JJ), (Korean, JJ)], LOCATION)
4                         -                                          -
5                         -                                          -
6                         -                                          -
```

[points: 2] Exercise 1c: Constituency parsing

Use the `nltk.RegexpParser` to perform constituency parsing on each sentence.

Use `print` to **show** the output in the notebook (and hence also in the exported PDF!).

```
constituent_parser = nltk.RegexpParser('''
NP: {<DT>? <JJ>* <NN>*} # NP
P: {<IN>}           # Preposition
V: {<V.*>}          # Verb
PP: {<P> <NP>}      # PP -> P NP
VP: {<V> <NP|PP>*}  # VP -> V (NP|PP)*''')

constituency_output_per_sentence = []
for sent in pos_tags_per_sentence:
    sent = constituent_parser.parse(sent)
    constituency_output_per_sentence.append(sent)
print(constituency_output_per_sentence[-1])
```

```
(S
  (NP A/DT similar/JJ case/NN)
  (PP (P in/IN) (NP the/DT))
  UK/NNP
  (VP (V found/VBD))
  (P in/IN)
  Samsung/NNP
  's/POS
  (NP favour/NN)
  and/CC
  (VP (V ordered/VBD))
  Apple/NNP
  to/TO
  (VP (V publish/VB) (NP an/DT apology/NN))
  (VP
    (V making/VBG)
    (NP clear/JJ)
    (PP (P that/IN) (NP the/DT South/JJ Korean/JJ firm/NN)))
  (VP (V had/VBD))
  not/RB
  (VP (V copied/VBN))
  its/PRP$
  (NP iPad/NN)
  when/WRB
  (VP (V designing/VBG))
  its/PRP$
  (NP own/JJ)
  devices/NNS
  ./.)
```

```
pd.DataFrame(constituency_output_per_sentence).rename("sentence
{}".format).T.fillna('-')
```

	sentence 0 \
0	[(https, NN)]
1	(:, :)
2	[(//www.telegraph.co.uk/technology/apple/97027...
3	(Documents, NNS)
4	[[(filed, VBN)]]
5	(to, TO)
6	[(the, DT)]
7	(San, NNP)
8	(Jose, NNP)
9	[(federal, JJ), (court, NN)]
10	[(in, IN)]
11	(California, NNP)
12	[(on, IN)]
13	(November, NNP)
14	(23, CD)
15	[(list, NN)]
16	(six, CD)

```
17                                    (Samsung, NNP)
18                                   (products, NNS)
19                     [[(running, VBG)], [(the, DT)]]
20                                        (``, ``)
21                                      (Jelly, RB)
22                                      (Bean, NNP)
23                                         ('', '')
24                                       (and, CC)
25                                        (``, ``)
26                                       (Ice, NNP)
27                                     (Cream, NNP)
28                                   (Sandwich, NNP)
29                                         ('', '')
30                                [[(operating, VBG)]]
31                                      (systems, NNS)
32                                              (,, ,)
33                                        (which, WDT)
34                                        (Apple, NNP)
35                                   [[(claims, VBZ)]]
36                                  [[(infringe, VB)]]
37                                         (its, PRP$)
38                                      (patents, NNS)
39                                              (., .)

                                     sentence 1  \
0                                   [(The, DT)]
1                                     (six, CD)
2                                  (phones, NNS)
3                                      (and, CC)
4                                 (tablets, NNS)
5                            [[(affected, VBN)]]
6                    [[(are, VBP)], [(the, DT)]]
7                                  (Galaxy, NNP)
8                                       (S, NNP)
9                                     (III, NNP)
10                                       (,, ,)
11  [[(running, VBG)], [(the, DT), (new, JJ)]]
12                                  (Jelly, NNP)
13                                    (Bean, NNP)
14                                [(system, NN)]
15                                       (,, ,)
16                                  [(the, DT)]
17                                 (Galaxy, NNP)
18                                    (Tab, NNP)
19                                     (8.9, CD)
20                                   (Wifi, NNP)
21                                [(tablet, NN)]
22                                       (,, ,)
23                                  [(the, DT)]
24                                 (Galaxy, NNP)
```

```
25                                    (Tab, NNP)
26                                       (2, CD)
27                                    (10.1, CD)
28                                        (,, ,)
29                                  (Galaxy, NNP)
30                                   (Rugby, NNP)
31                                     (Pro, NNP)
32                                      (and, CC)
33                                  (Galaxy, NNP)
34                                       (S, NNP)
35                                     (III, NNP)
36                                   [(mini, NN)]
37                                        (., .)
38                                             -
39                                             -

                                           sentence 2  \
0                                       (Apple, NNP)
1                                  [[(stated, VBD)]]
2                                          (it, PRP)
3                                    [[(had, VBD)]]
4                                 [[(â€œacted, VBN)]]
5                                      (quickly, RB)
6                                          (and, CC)
7                                   (diligently, RB)
8                                           ('', '')
9                        [[(in, IN)], [(order, NN)]]
10                                          (to, TO)
11                                          (``, ``)
12  [[(determine, VB)], [[('that', 'IN')], [('thes...
13                                       (newly, RB)
14                                [[(released, VBN)]]
15                                    (products, NNS)
16                                   [[(do, VBP)]]
17  [[(infringe, VB)], [(many, JJ)], [[('of', 'IN'...
18                                      (claims, NNS)
19                                      (already, RB)
20                                [[(asserted, VBN)]]
21                                         [(by, IN)]
22                                      (Apple, NNP)
23                                           (., .)
24                                          ('', '')
25                                                 -
26                                                 -
27                                                 -
28                                                 -
29                                                 -
30                                                 -
31                                                 -
32                                                 -
```

33	-
34	-
35	-
36	-
37	-
38	-
39	-

	sentence 3 \
0	[(In, IN)]
1	(August, NNP)
2	(,, ,)
3	(Samsung, NNP)
4	[[(lost, VBD)], [(a, DT)]]
5	(US, NNP)
6	[(patent, NN), (case, NN)]
7	(to, TO)
8	(Apple, NNP)
9	(and, CC)
10	[[(was, VBD)]]
11	[[(ordered, VBN)]]
12	(to, TO)
13	[[(pay, VB)]]
14	(its, PRP$)
15	[(rival, JJ)]
16	($, $)
17	(1.05bn, CD)
18	((, ()
19	[(Â£0.66bn, NN)]
20	(),))
21	[(in, IN)]
22	(damages, NNS)
23	[(for, IN)]
24	[[(copying, VBG)]]
25	(features, NNS)
26	[[(of, IN)], [(the, DT), (iPad, NN)]]
27	(and, CC)
28	[(iPhone, NN)]
29	[(in, IN)]
30	(its, PRP$)
31	(Galaxy, NNP)
32	[(range, NN)]
33	[(of, IN)]
34	(devices, NNS)
35	(., .)
36	-
37	-
38	-
39	-

	sentence 4
0	(Samsung, NNP)
1	(,, ,)
2	(which, WDT)
3	[[(is, VBZ)], [(the, DT), (world, NN)]]
4	('s, POS)
5	[(top, JJ), (mobile, NN), (phone, NN), (maker,...
6	(,, ,)
7	[[(is, VBZ)]]
8	[[(appealing, VBG)], [(the, DT), (ruling, NN)]]
9	(., .)
10	-
11	-
12	-
13	-
14	-
15	-
16	-
17	-
18	-
19	-
20	-
21	-
22	-
23	-
24	-
25	-
26	-
27	-
28	-
29	-
30	-
31	-
32	-
33	-
34	-
35	-
36	-
37	-
38	-
39	-

	sentence 5
0	[(A, DT), (similar, JJ), (case, NN)]
1	[[(in, IN)], [(the, DT)]]
2	(UK, NNP)
3	[[(found, VBD)]]
4	[(in, IN)]
5	(Samsung, NNP)
6	('s, POS)

```
7                                        [(favour, NN)]
8                                             (and, CC)
9                                   [[(ordered, VBD)]]
10                                        (Apple, NNP)
11                                            (to, TO)
12       [[(publish, VB)], [(an, DT), (apology, NN)]]
13  [[(making, VBG)], [(clear, JJ)], [[('that', 'I...
14                                       [[(had, VBD)]]
15                                            (not, RB)
16                                   [[(copied, VBN)]]
17                                         (its, PRP$)
18                                       [(iPad, NN)]
19                                         (when, WRB)
20                                [[(designing, VBG)]]
21                                         (its, PRP$)
22                                         [(own, JJ)]
23                                     (devices, NNS)
24                                             (., .)
25                                                  -
26                                                  -
27                                                  -
28                                                  -
29                                                  -
30                                                  -
31                                                  -
32                                                  -
33                                                  -
34                                                  -
35                                                  -
36                                                  -
37                                                  -
38                                                  -
39                                                  -
```

Augment the RegexpParser so that it also detects Named Entity Phrases (NEP), e.g., that it detects *Galaxy S III* and *Ice Cream Sandwich*

```
constituent_parser_v2 = nltk.RegexpParser('''
NP: {<DT>? <JJ>* <NN>*} # NP
P: {<IN>}           # Preposition
V: {<V.*>}          # Verb
PP: {<P> <NP>}      # PP -> P NP
VP: {<V> <NP|PP>*}  # VP -> V (NP|PP)*
NEP: {<NNP>*}       # NEP -> NNP*''')

constituency_v2_output_per_sentence = []
for sent in pos_tags_per_sentence:
    sent = constituent_parser_v2.parse(sent)
    constituency_v2_output_per_sentence.append(sent)
print(constituency_v2_output_per_sentence[5])
```

```
(S
  (NP A/DT similar/JJ case/NN)
  (PP (P in/IN) (NP the/DT))
  (NEP UK/NNP)
  (VP (V found/VBD))
  (P in/IN)
  (NEP Samsung/NNP)
  's/POS
  (NP favour/NN)
  and/CC
  (VP (V ordered/VBD))
  (NEP Apple/NNP)
  to/TO
  (VP (V publish/VB) (NP an/DT apology/NN))
  (VP
    (V making/VBG)
    (NP clear/JJ)
    (PP (P that/IN) (NP the/DT South/JJ Korean/JJ firm/NN)))
  (VP (V had/VBD))
  not/RB
  (VP (V copied/VBN))
  its/PRP$
  (NP iPad/NN)
  when/WRB
  (VP (V designing/VBG))
  its/PRP$
  (NP own/JJ)
  devices/NNS
  ./.)
```

```
pd.DataFrame(constituency_v2_output_per_sentence).rename("sentence
{}".format).T.fillna('-')
```

	sentence 0
0	[(https, NN)]
1	(:, :)
2	[(//www.telegraph.co.uk/technology/apple/97027...
3	(Documents, NNS)
4	[[(filed, VBN)]]
5	(to, TO)
6	[(the, DT)]
7	[(San, NNP), (Jose, NNP)]
8	[(federal, JJ), (court, NN)]
9	[(in, IN)]
10	[(California, NNP)]
11	[(on, IN)]
12	[(November, NNP)]
13	(23, CD)
14	[(list, NN)]
15	(six, CD)
16	[(Samsung, NNP)]

```
17                                    (products, NNS)
18                     [[(running, VBG)], [(the, DT)]]
19                                           (``, ``)
20                                      (Jelly, RB)
21                                    [(Bean, NNP)]
22                                         ('', '')
23                                       (and, CC)
24                                         (``, ``)
25        [(Ice, NNP), (Cream, NNP), (Sandwich, NNP)]
26                                         ('', '')
27                                 [[(operating, VBG)]]
28                                       (systems, NNS)
29                                               (,, ,)
30                                        (which, WDT)
31                                     [(Apple, NNP)]
32                                   [[(claims, VBZ)]]
33                                  [[(infringe, VB)]]
34                                         (its, PRP$)
35                                      (patents, NNS)
36                                               (., .)

                                     sentence 1  \
0                                   [(The, DT)]
1                                     (six, CD)
2                                  (phones, NNS)
3                                      (and, CC)
4                                 (tablets, NNS)
5                           [[(affected, VBN)]]
6                   [[(are, VBP)], [(the, DT)]]
7        [(Galaxy, NNP), (S, NNP), (III, NNP)]
8                                      (,, ,)
9   [[(running, VBG)], [(the, DT), (new, JJ)]]
10                  [(Jelly, NNP), (Bean, NNP)]
11                               [(system, NN)]
12                                     (,, ,)
13                                  [(the, DT)]
14                  [(Galaxy, NNP), (Tab, NNP)]
15                                    (8.9, CD)
16                                 [(Wifi, NNP)]
17                               [(tablet, NN)]
18                                     (,, ,)
19                                  [(the, DT)]
20                  [(Galaxy, NNP), (Tab, NNP)]
21                                       (2, CD)
22                                    (10.1, CD)
23                                       (,, ,)
24   [(Galaxy, NNP), (Rugby, NNP), (Pro, NNP)]
25                                    (and, CC)
26       [(Galaxy, NNP), (S, NNP), (III, NNP)]
27                                 [(mini, NN)]
```

	sentence 2	\
0	[(Apple, NNP)]	
1	[[(stated, VBD)]]	
2	(it, PRP)	
3	[[(had, VBD)]]	
4	[[(â€œacted, VBN)]]	
5	(quickly, RB)	
6	(and, CC)	
7	(diligently, RB)	
8	('', '')	
9	[[(in, IN)], [(order, NN)]]	
10	(to, TO)	
11	(``, ``)	
12	[[(determine, VB)], [[('that', 'IN')], [('thes...	
13	(newly, RB)	
14	[[(released, VBN)]]	
15	(products, NNS)	
16	[[(do, VBP)]]	
17	[[(infringe, VB)], [(many, JJ)], [[('of', 'IN'...	
18	(claims, NNS)	
19	(already, RB)	
20	[[(asserted, VBN)]]	
21	[(by, IN)]	
22	[(Apple, NNP)]	
23	(., .)	
24	('', '')	
25	-	
26	-	
27	-	
28	-	
29	-	
30	-	
31	-	
32	-	
33	-	
34	-	
35	-	
36	-	

	sentence 3	\

0	[(In, IN)]
1	[(August, NNP)]
2	(,, ,)
3	[(Samsung, NNP)]
4	[[(lost, VBD)], [(a, DT)]]
5	[(US, NNP)]
6	[(patent, NN), (case, NN)]
7	(to, TO)
8	[(Apple, NNP)]
9	(and, CC)
10	[[(was, VBD)]]
11	[[(ordered, VBN)]]
12	(to, TO)
13	[[(pay, VB)]]
14	(its, PRP$)
15	[(rival, JJ)]
16	($, $)
17	(1.05bn, CD)
18	((, ()
19	[(Â£0.66bn, NN)]
20	(),))
21	[(in, IN)]
22	(damages, NNS)
23	[(for, IN)]
24	[[(copying, VBG)]]
25	(features, NNS)
26	[[(of, IN)], [(the, DT), (iPad, NN)]]
27	(and, CC)
28	[(iPhone, NN)]
29	[(in, IN)]
30	(its, PRP$)
31	[(Galaxy, NNP)]
32	[(range, NN)]
33	[(of, IN)]
34	(devices, NNS)
35	(., .)
36	-

	sentence 4 \
0	[(Samsung, NNP)]
1	(,, ,)
2	(which, WDT)
3	[[(is, VBZ)], [(the, DT), (world, NN)]]
4	('s, POS)
5	[(top, JJ), (mobile, NN), (phone, NN), (maker,...
6	(,, ,)
7	[[(is, VBZ)]]
8	[[(appealing, VBG)], [(the, DT), (ruling, NN)]]
9	(., .)
10	-

11	-
12	-
13	-
14	-
15	-
16	-
17	-
18	-
19	-
20	-
21	-
22	-
23	-
24	-
25	-
26	-
27	-
28	-
29	-
30	-
31	-
32	-
33	-
34	-
35	-
36	-

	sentence 5
0	[(A, DT), (similar, JJ), (case, NN)]
1	[[(in, IN)], [(the, DT)]]
2	[(UK, NNP)]
3	[[(found, VBD)]]
4	[(in, IN)]
5	[(Samsung, NNP)]
6	('s, POS)
7	[(favour, NN)]
8	(and, CC)
9	[[(ordered, VBD)]]
10	[(Apple, NNP)]
11	(to, TO)
12	[[(publish, VB)], [(an, DT), (apology, NN)]]
13	[[(making, VBG)], [(clear, JJ)], [[('that', 'I...
14	[[(had, VBD)]]
15	(not, RB)
16	[[(copied, VBN)]]
17	(its, PRP$)
18	[(iPad, NN)]
19	(when, WRB)
20	[[(designing, VBG)]]
21	(its, PRP$)

```
22                                          [(own, JJ)]
23                                        (devices, NNS)
24                                                (., .)
25                                                     -
26                                                     -
27                                                     -
28                                                     -
29                                                     -
30                                                     -
31                                                     -
32                                                     -
33                                                     -
34                                                     -
35                                                     -
36                                                     -
```

[total points: 1] Exercise 2: spaCy

Use Spacy to process the same text as you analyzed with NLTK.

```
import spacy
nlp = spacy.load('en_core_web_sm')

doc = nlp(text) # insert code here
```

small tip: You can use **sents = list(doc.sents)** to be able to use the index to access a sentence like **sents[2]** for the third sentence.

```
sentences_cy = list(doc.sents)

pos_tags_per_sentence_cy = [[(tok.text, tok.tag_) for tok in sent] for
sent in sentences_cy]
pos_tags_per_sentence_cy[0]
```

```
[('https://www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html',
  'NNS'),
 ('\n\n', '_SP'),
 ('Documents', 'NNS'),
 ('filed', 'VBD'),
 ('to', 'IN'),
 ('the', 'DT'),
 ('San', 'NNP'),
 ('Jose', 'NNP'),
 ('federal', 'JJ'),
 ('court', 'NN'),
 ('in', 'IN'),
 ('California', 'NNP'),
 ('on', 'IN'),
 ('November', 'NNP'),
 ('23', 'CD'),
```

```
 ('list', 'NN'),
 ('six', 'CD'),
 ('Samsung', 'NNP'),
 ('products', 'NNS'),
 ('running', 'VBG'),
 ('the', 'DT'),
 ('"', '``'),
 ('Jelly', 'NNP'),
 ('Bean', 'NNP'),
 ('"', "''"),
 ('and', 'CC'),
 ('"', '``'),
 ('Ice', 'NNP'),
 ('Cream', 'NNP'),
 ('Sandwich', 'NN'),
 ('"', "''"),
 ('operating', 'NN'),
 ('systems', 'NNS'),
 (',', ','),
 ('which', 'WDT'),
 ('Apple', 'NNP'),
 ('claims', 'VBZ'),
 ('infringe', 'VBP'),
 ('its', 'PRP$'),
 ('patents', 'NNS'),
 ('.', '.'),
 ('\n', '_SP')]
```

```
ne_cy = [(x.text, x.label_) for x in doc.ents]
ne_cy
```

```
[('https://www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html',
  'TIME'),
 ('San Jose', 'GPE'),
 ('California', 'GPE'),
 ('November 23', 'DATE'),
 ('six', 'CARDINAL'),
 ('Samsung', 'ORG'),
 ('the "Jelly Bean', 'LAW'),
 ('Apple', 'ORG'),
 ('six', 'CARDINAL'),
 ('the Galaxy S III', 'ORG'),
 ('Jelly Bean', 'ORG'),
 ('8.9', 'CARDINAL'),
 ('2 10.1', 'DATE'),
 ('Galaxy Rugby Pro', 'ORG'),
 ('Galaxy S III', 'PERSON'),
 ('Apple', 'ORG'),
 ('Apple', 'ORG'),
 ('August', 'DATE'),
```

```
 ('Samsung', 'ORG'),
 ('US', 'GPE'),
 ('Apple', 'ORG'),
 ('1.05bn', 'MONEY'),
 ('iPad', 'ORG'),
 ('Galaxy', 'FAC'),
 ('Samsung', 'ORG'),
 ('UK', 'GPE'),
 ('Samsung', 'ORG'),
 ('Apple', 'ORG'),
 ('South Korean', 'NORP'),
 ('iPad', 'ORG')]
```

We will use lists to keep track of the output of the NLP tasks. We can hence inspect the output for each task using the index of the sentence.

[total points: 7] Exercise 3: Comparison NLTK and spaCy

We will now compare the output of NLTK and spaCy, i.e., in what do they differ?

[points: 3] Exercise 3a: Part of speech tagging

Compare the output from NLTK and spaCy regarding part of speech tagging.

- To compare, you probably would like to compare sentence per sentence. Describe if the sentence splitting is different for NLTK than for spaCy. If not, where do they differ?
- After checking the sentence splitting, select a sentence for which you expect interesting results and perhaps differences. Motivate your choice.
- Compare the output in `token.tag` from spaCy to the part of speech tagging from NLTK for each token in your selected sentence. Are there any differences? This is not a trick question; it is possible that there are no differences.

```
if len(sentences_cy) == len(sentences_nltk):
    for i, (a, b) in enumerate(zip(sentences_nltk, sentences_cy)):
        if a != b.text.strip():
            print(f"Sentence {i}:")
            print(sentences_nltk[i])
            print(sentences_cy[i])
            print()
```

```
Sentence 2:
Apple stated it had â€œacted quickly and diligently" in order to
"determine that these newly released products do infringe many of the
same claims already asserted by Apple."
Apple stated it had â€œacted quickly and diligently" in order to
"determine that these newly released products do infringe many of the
same claims already asserted by Apple.

Sentence 3:
In August, Samsung lost a US patent case to Apple and was ordered to
```

```
pay its rival $1.05bn (Â£0.66bn) in damages for copying features of
the iPad and iPhone in its Galaxy range of devices.
"
In August, Samsung lost a US patent case to Apple and was ordered to
pay its rival $1.05bn (Â£0.66bn) in damages for copying features of
the iPad and iPhone in its Galaxy range of devices.
```

The sentence parsing is almost identical regardless if done by nltk or spaCy. The minor differences are that spaCy may leave some whitespaces at the end of the sentence and nltk tends to leave in the quotation mark if there is one at the end.

```
sent_id = 5
print(sentences_nltk[sent_id], '\n')
for i, (a, b) in enumerate(zip(pos_tags_per_sentence[sent_id],
pos_tags_per_sentence_cy[sent_id][:-1])):
    if a != b:
        print(f"nltk: {str(pos_tags_per_sentence[sent_id]
[i]):25}spaCy: {pos_tags_per_sentence_cy[sent_id][i]}")
```

```
A similar case in the UK found in Samsung's favour and ordered Apple
to publish an apology making clear that the South Korean firm had not
copied its iPad when designing its own devices.

nltk: ('found', 'VBD')         spaCy: ('found', 'VBN')
nltk: ('iPad', 'NN')           spaCy: ('iPad', 'NNP')
```

The 5th sentence was chosen because it contains many named entities which can be tricky to classify. There are two part of speech discrepencies: spaCy correctly classified 'iPad' as a proper noun and 'found' was classified as a past participle verb and a past tense verb by spaCy and nltk respectively. After examining the sentece, it can be seen that here nltk is right.

[points: 2] Exercise 3b: Named Entity Recognition (NER)

- Describe differences between the output from NLTK and spaCy for Named Entity Recognition. Which one do you think performs better?

```
ne_nltk = [(tok, tok.label()) for sent in ner_tags_per_sentence for
tok in sent if isinstance(tok, nltk.tree.tree.Tree)]
df = pd.DataFrame.from_dict(
    {('nltk', 'entity'): [i for i, _ in ne_nltk],
     ('nltk', 'type'): [i for _, i in ne_nltk],
     ('spaCy', 'entity'): [i for i, _ in ne_cy],
     ('spaCy', 'type'): [i for _, i in ne_cy]}, orient='index').T
df.columns = pd.MultiIndex.from_tuples(df.columns)
df
```

	nltk	
	entity	type
0	[(San, NNP), (Jose, NNP)]	ORGANIZATION
1	[(California, NNP)]	GPE
2	[(Samsung, NNP)]	ORGANIZATION

3	[(Bean, NNP)]	GPE
4	[(Apple, NNP)]	PERSON
5	[(Galaxy, NNP)]	ORGANIZATION
6	[(Jelly, NNP), (Bean, NNP)]	PERSON
7	[(Galaxy, NNP)]	ORGANIZATION
8	[(Galaxy, NNP)]	ORGANIZATION
9	[(Galaxy, NNP), (Rugby, NNP), (Pro, NNP)]	PERSON
10	[(Galaxy, NNP), (S, NNP)]	PERSON
11	[(Apple, NNP)]	PERSON
12	[(Apple, NNP)]	PERSON
13	[(August, NNP)]	GPE
14	[(Samsung, NNP)]	PERSON
15	[(US, NNP)]	GSP
16	[(Apple, NNP)]	GPE
17	[(iPad, NN)]	ORGANIZATION
18	[(iPhone, NN)]	ORGANIZATION
19	[(Galaxy, NNP)]	GPE
20	[(Samsung, NNP)]	GPE
21	[(UK, NNP)]	ORGANIZATION
22	[(Samsung, NNP)]	GPE
23	[(Apple, NNP)]	PERSON
24	[(South, JJ), (Korean, JJ)]	LOCATION
25	None	None
26	None	None
27	None	None
28	None	None
29	None	None

	spaCy entity	type
0	https://www.telegraph.co.uk/technology/apple/9...	TIME
1	San Jose	GPE
2	California	GPE
3	November 23	DATE
4	six	CARDINAL
5	Samsung	ORG
6	the "Jelly Bean	LAW
7	Apple	ORG
8	six	CARDINAL
9	the Galaxy S III	ORG
10	Jelly Bean	ORG
11	8.9	CARDINAL
12	2 10.1	DATE
13	Galaxy Rugby Pro	ORG
14	Galaxy S III	PERSON
15	Apple	ORG
16	Apple	ORG
17	August	DATE
18	Samsung	ORG
19	US	GPE

```
20                                                  Apple       ORG
21                                                1.05bn     MONEY
22                                                  iPad       ORG
23                                                Galaxy        FAC
24                                               Samsung       ORG
25                                                    UK        GPE
26                                               Samsung       ORG
27                                                 Apple       ORG
28                                          South Korean      NORP
29                                                  iPad       ORG
```

More named entities are recognized by spaCy. Furthemore, most of the entities recognized by nltk are misclassified. The classification done by spaCy is much better and most of the entities are classified correctly. But spaCy too has a few errors because if Jelly Bean is law, I should be in jail.

[points: 2] Exercise 3c: Constituency/dependency parsing

Choose one sentence from the text and run constituency parsing using NLTK and dependency parsing using spaCy.

- describe briefly the difference between constituency parsing and dependency parsing
- describe differences between the output from NLTK and spaCy.

```
print(constituency_output_per_sentence[sent_id], '\n')
spacy.displacy.render(sentences_cy[sent_id], jupyter=True,
style='dep', options={'distance': 100})
constituency_output_per_sentence[sent_id].draw()
```

```
(S
  (NP A/DT similar/JJ case/NN)
  (PP (P in/IN) (NP the/DT))
  UK/NNP
  (VP (V found/VBD))
  (P in/IN)
  Samsung/NNP
  's/POS
  (NP favour/NN)
  and/CC
  (VP (V ordered/VBD))
  Apple/NNP
  to/TO
  (VP (V publish/VB) (NP an/DT apology/NN))
  (VP
    (V making/VBG)
    (NP clear/JJ)
    (PP (P that/IN) (NP the/DT South/JJ Korean/JJ firm/NN)))
  (VP (V had/VBD))
  not/RB
  (VP (V copied/VBN))
```

```
  its/PRP$
  (NP iPad/NN)
  when/WRB
  (VP (V designing/VBG))
  its/PRP$
  (NP own/JJ)
  devices/NNS
  ./.)
```

```
<IPython.core.display.HTML object>
```

Constituancy parsing shows the tree structure of a sentence. The root node is the sentece itself, low level nodes are usually clauses and phrases, and the leaf nodes are the words constituting the sentence.

The dependency graph on the other hand shows how each word of the sentence is related to the other words.

End of this notebook