Lab1-Assignment

Copyright: Vrije Universiteit Amsterdam, Faculty of Humanities, CLTL

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\amina\OneDrive\Documents\4 Text Mining for AI\text-mining-
group17\Lab 1\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 1139
```

[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 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])
print(len(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', '.']
41
```

[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:
    print(nltk.pos_tag(tokens))
    pos_tags = nltk.pos_tag(tokens)
    pos_tags_per_sentence.append(pos_tags)
```

```
[('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'), ('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',
'RB'), ('Bean', 'NNP'), ("''", "''"), ('and', 'CC'), ('``', '``'),
('Ice', 'NNP'), ('Cream', 'NNP'), ('Sandwich', 'NNP'), ("''", "''"),
('operating', 'VBG'), ('systems', 'NNS'), (',', ','), ('which',
'WDT'), ('Apple', 'NNP'), ('claims', 'VBZ'), ('infringe', 'VB'),
('its', 'PRP$'), ('patents', 'NNS'), ('.', '.')]
[('The', 'DT'), ('six', 'CD'), ('phones', 'NNS'), ('and', 'CC'),
('tablets', 'NNS'), ('affected', 'VBN'), ('are', 'VBP'), ('the',
'DT'), ('Galaxy', 'NNP'), ('S', 'NNP'), ('III', 'NNP'), (',', ','),
('running', 'VBG'), ('the', 'DT'), ('new', 'JJ'), ('Jelly', 'NNP'),
('Bean', 'NNP'), ('system', 'NN'), (',', ','), ('the', 'DT'),
('Galaxy', 'NNP'), ('Tab', 'NNP'), ('8.9', 'CD'), ('Wifi', 'NNP'),
('tablet', 'NN'), (',', ','), ('the', 'DT'), ('Galaxy', 'NNP'),
('Tab', 'NNP'), ('2', 'CD'), ('10.1', 'CD'), (',', ','), ('Galaxy',
'NNP'), ('Rugby', 'NNP'), ('Pro', 'NNP'), ('and', 'CC'), ('Galaxy',
'NNP'), ('S', 'NNP'), ('III', 'NNP'), ('mini', 'NN'), ('.', '.')]
[('Apple', 'NNP'), ('stated', 'VBD'), ('it', 'PRP'), ('had', 'VBD'),
('“', 'NNP'), ('acted', 'VBD'), ('quickly', 'RB'), ('and', 'CC'),
('diligently', 'RB'), ("''", "''"), ('in', 'IN'), ('order', 'NN'),
('to', 'TO'), ('``', '``'), ('determine', 'VB'), ('that', 'IN'),
('these', 'DT'), ('newly', 'RB'), ('released', 'VBN'), ('products',
'NNS'), ('do', 'VBP'), ('infringe', 'VB'), ('many', 'JJ'), ('of',
'IN'), ('the', 'DT'), ('same', 'JJ'), ('claims', 'NNS'), ('already',
'RB'), ('asserted', 'VBN'), ('by', 'IN'), ('Apple', 'NNP'), ('.',
'.'), ("''", "''")]
[('In', 'IN'), ('August', 'NNP'), (',', ','), ('Samsung', 'NNP'),
('lost', 'VBD'), ('a', 'DT'), ('US', 'NNP'), ('patent', 'NN'),
('case', 'NN'), ('to', 'TO'), ('Apple', 'NNP'), ('and', 'CC'), ('was',
'VBD'), ('ordered', 'VBN'), ('to', 'TO'), ('pay', 'VB'), ('its',
'PRP$'), ('rival', 'JJ'), ('$', '$'), ('1.05bn', 'CD'), ('(', '('),
('£0.66bn', 'NN'), (')', ')'), ('in', 'IN'), ('damages', 'NNS'),
('for', 'IN'), ('copying', 'VBG'), ('features', 'NNS'), ('of', 'IN'),
('the', 'DT'), ('iPad', 'NN'), ('and', 'CC'), ('iPhone', 'NN'), ('in',
'IN'), ('its', 'PRP$'), ('Galaxy', 'NNP'), ('range', 'NN'), ('of',
'IN'), ('devices', 'NNS'), ('.', '.')]
[('Samsung', 'NNP'), (',', ','), ('which', 'WDT'), ('is', 'VBZ'),
('the', 'DT'), ('world', 'NN'), ("'s", 'POS'), ('top', 'JJ'),
('mobile', 'NN'), ('phone', 'NN'), ('maker', 'NN'), (',', ','), ('is',
'VBZ'), ('appealing', 'VBG'), ('the', 'DT'), ('ruling', 'NN'), ('.',
'.')]
[('A', 'DT'), ('similar', 'JJ'), ('case', 'NN'), ('in', 'IN'), ('the',
'DT'), ('UK', 'NNP'), ('found', 'VBD'), ('in', 'IN'), ('Samsung',
'NNP'), ("'s", 'POS'), ('favour', 'NN'), ('and', 'CC'), ('ordered',
'VBD'), ('Apple', 'NNP'), ('to', 'TO'), ('publish', 'VB'), ('an',
```

```
'DT'), ('apology', 'NN'), ('making', 'VBG'), ('clear', 'JJ'), ('that',
'IN'), ('the', 'DT'), ('South', 'JJ'), ('Korean', 'JJ'), ('firm',
'NN'), ('had', 'VBD'), ('not', 'RB'), ('copied', 'VBN'), ('its',
'PRP$'), ('iPad', 'NN'), ('when', 'WRB'), ('designing', 'VBG'),
('its', 'PRP$'), ('own', 'JJ'), ('devices', 'NNS'), ('.', '.')]
```

```
print(pos_tags_per_sentence)
```

```
[[('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'), ('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',
'RB'), ('Bean', 'NNP'), ("''", "''"), ('and', 'CC'), ('``', '``'),
('Ice', 'NNP'), ('Cream', 'NNP'), ('Sandwich', 'NNP'), ("''", "''"),
('operating', 'VBG'), ('systems', 'NNS'), (',', ','), ('which',
'WDT'), ('Apple', 'NNP'), ('claims', 'VBZ'), ('infringe', 'VB'),
('its', 'PRP$'), ('patents', 'NNS'), ('.', '.')], [('The', 'DT'),
('six', 'CD'), ('phones', 'NNS'), ('and', 'CC'), ('tablets', 'NNS'),
('affected', 'VBN'), ('are', 'VBP'), ('the', 'DT'), ('Galaxy', 'NNP'),
('S', 'NNP'), ('III', 'NNP'), (',', ','), ('running', 'VBG'), ('the',
'DT'), ('new', 'JJ'), ('Jelly', 'NNP'), ('Bean', 'NNP'), ('system',
'NN'), (',', ','), ('the', 'DT'), ('Galaxy', 'NNP'), ('Tab', 'NNP'),
('8.9', 'CD'), ('Wifi', 'NNP'), ('tablet', 'NN'), (',', ','), ('the',
'DT'), ('Galaxy', 'NNP'), ('Tab', 'NNP'), ('2', 'CD'), ('10.1', 'CD'),
(',', ','), ('Galaxy', 'NNP'), ('Rugby', 'NNP'), ('Pro', 'NNP'),
('and', 'CC'), ('Galaxy', 'NNP'), ('S', 'NNP'), ('III', 'NNP'),
('mini', 'NN'), ('.', '.')], [('Apple', 'NNP'), ('stated', 'VBD'),
('it', 'PRP'), ('had', 'VBD'), ('“', 'NNP'), ('acted', 'VBD'),
('quickly', 'RB'), ('and', 'CC'), ('diligently', 'RB'), ("''", "''"),
('in', 'IN'), ('order', 'NN'), ('to', 'TO'), ('``', '``'),
('determine', 'VB'), ('that', 'IN'), ('these', 'DT'), ('newly', 'RB'),
('released', 'VBN'), ('products', 'NNS'), ('do', 'VBP'), ('infringe',
'VB'), ('many', 'JJ'), ('of', 'IN'), ('the', 'DT'), ('same', 'JJ'),
('claims', 'NNS'), ('already', 'RB'), ('asserted', 'VBN'), ('by',
'IN'), ('Apple', 'NNP'), ('.', '.'), ("''", "''")], [('In', 'IN'),
('August', 'NNP'), (',', ','), ('Samsung', 'NNP'), ('lost', 'VBD'),
('a', 'DT'), ('US', 'NNP'), ('patent', 'NN'), ('case', 'NN'), ('to',
'TO'), ('Apple', 'NNP'), ('and', 'CC'), ('was', 'VBD'), ('ordered',
'VBN'), ('to', 'TO'), ('pay', 'VB'), ('its', 'PRP$'), ('rival', 'JJ'),
('$', '$'), ('1.05bn', 'CD'), ('(', '('), ('£0.66bn', 'NN'), (')',
')'), ('in', 'IN'), ('damages', 'NNS'), ('for', 'IN'), ('copying',
'VBG'), ('features', 'NNS'), ('of', 'IN'), ('the', 'DT'), ('iPad',
'NN'), ('and', 'CC'), ('iPhone', 'NN'), ('in', 'IN'), ('its', 'PRP$'),
('Galaxy', 'NNP'), ('range', 'NN'), ('of', 'IN'), ('devices', 'NNS'),
('.', '.')], [('Samsung', 'NNP'), (',', ','), ('which', 'WDT'), ('is',
'VBZ'), ('the', 'DT'), ('world', 'NN'), ("'s", 'POS'), ('top', 'JJ'),
```

```
('mobile', 'NN'), ('phone', 'NN'), ('maker', 'NN'), (',', ','), ('is',
'VBZ'), ('appealing', 'VBG'), ('the', 'DT'), ('ruling', 'NN'), ('.',
'.')], [('A', 'DT'), ('similar', 'JJ'), ('case', 'NN'), ('in', 'IN'),
('the', 'DT'), ('UK', 'NNP'), ('found', 'VBD'), ('in', 'IN'),
('Samsung', 'NNP'), ("'s", 'POS'), ('favour', 'NN'), ('and', 'CC'),
('ordered', 'VBD'), ('Apple', 'NNP'), ('to', 'TO'), ('publish', 'VB'),
('an', 'DT'), ('apology', 'NN'), ('making', 'VBG'), ('clear', 'JJ'),
('that', 'IN'), ('the', 'DT'), ('South', 'JJ'), ('Korean', 'JJ'),
('firm', 'NN'), ('had', 'VBD'), ('not', 'RB'), ('copied', 'VBN'),
('its', 'PRP$'), ('iPad', 'NN'), ('when', 'WRB'), ('designing',
'VBG'), ('its', 'PRP$'), ('own', 'JJ'), ('devices', 'NNS'), ('.',
'.')]]
```

[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 pos_tags in pos_tags_per_sentence:
    print(nltk.chunk.ne_chunk(pos_tags))
    ner_tags_per_sentence.append(nltk.chunk.ne_chunk(pos_tags))
```

```
(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
  ./.)
(S
  The/DT
  six/CD
  phones/NNS
  and/CC
  tablets/NNS
  affected/VBN
  are/VBP
  the/DT
  (ORGANIZATION Galaxy/NNP)
  S/NNP
  III/NNP
  ,/,
  running/VBG
  the/DT
  new/JJ
  (PERSON Jelly/NNP Bean/NNP)
  system/NN
  ,/,
  the/DT
  (ORGANIZATION Galaxy/NNP)
  Tab/NNP
  8.9/CD
  Wifi/NNP
  tablet/NN
  ,/,
  the/DT
  (ORGANIZATION Galaxy/NNP)
  Tab/NNP
  2/CD
  10.1/CD
```

```
  ,/,
  (PERSON Galaxy/NNP Rugby/NNP Pro/NNP)
  and/CC
  (PERSON Galaxy/NNP S/NNP)
  III/NNP
  mini/NN
  ./.)
(S
  (PERSON Apple/NNP)
  stated/VBD
  it/PRP
  had/VBD
  “/NNP
  acted/VBD
  quickly/RB
  and/CC
  diligently/RB
  ''/''
  in/IN
  order/NN
  to/TO
  ``/``
  determine/VB
  that/IN
  these/DT
  newly/RB
  released/VBN
  products/NNS
  do/VBP
  infringe/VB
  many/JJ
  of/IN
  the/DT
  same/JJ
  claims/NNS
  already/RB
  asserted/VBN
  by/IN
  (PERSON Apple/NNP)
  ./.
  ''/'')
(S
  In/IN
  (GPE August/NNP)
  ,/,
  (PERSON Samsung/NNP)
  lost/VBD
  a/DT
  (GSP US/NNP)
```

```
  patent/NN
  case/NN
  to/TO
  (GPE Apple/NNP)
  and/CC
  was/VBD
  ordered/VBN
  to/TO
  pay/VB
  its/PRP$
  rival/JJ
  $/$
  1.05bn/CD
  (/(
  £0.66bn/NN
  )/)
  in/IN
  damages/NNS
  for/IN
  copying/VBG
  features/NNS
  of/IN
  the/DT
  (ORGANIZATION iPad/NN)
  and/CC
  (ORGANIZATION iPhone/NN)
  in/IN
  its/PRP$
  (GPE Galaxy/NNP)
  range/NN
  of/IN
  devices/NNS
  ./.)
(S
  (GPE Samsung/NNP)
  ,/,
  which/WDT
  is/VBZ
  the/DT
  world/NN
  's/POS
  top/JJ
  mobile/NN
  phone/NN
  maker/NN
  ,/,
  is/VBZ
  appealing/VBG
  the/DT
```

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

```
print(ner_tags_per_sentence)
```

```
[Tree('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'),
Tree('ORGANIZATION', [('San', 'NNP'), ('Jose', 'NNP')]), ('federal',
'JJ'), ('court', 'NN'), ('in', 'IN'), Tree('GPE', [('California',
'NNP')]), ('on', 'IN'), ('November', 'NNP'), ('23', 'CD'), ('list',
'NN'), ('six', 'CD'), Tree('ORGANIZATION', [('Samsung', 'NNP')]),
('products', 'NNS'), ('running', 'VBG'), ('the', 'DT'), ('``', '``'),
```

```
('Jelly', 'RB'), Tree('GPE', [('Bean', 'NNP')]), ("''", "''"), ('and',
'CC'), ('``', '``'), ('Ice', 'NNP'), ('Cream', 'NNP'), ('Sandwich',
'NNP'), ("''", "''"), ('operating', 'VBG'), ('systems', 'NNS'), (',',
','), ('which', 'WDT'), Tree('PERSON', [('Apple', 'NNP')]), ('claims',
'VBZ'), ('infringe', 'VB'), ('its', 'PRP$'), ('patents', 'NNS'), ('.',
'.')]), Tree('S', [('The', 'DT'), ('six', 'CD'), ('phones', 'NNS'),
('and', 'CC'), ('tablets', 'NNS'), ('affected', 'VBN'), ('are',
'VBP'), ('the', 'DT'), Tree('ORGANIZATION', [('Galaxy', 'NNP')]),
('S', 'NNP'), ('III', 'NNP'), (',', ','), ('running', 'VBG'), ('the',
'DT'), ('new', 'JJ'), Tree('PERSON', [('Jelly', 'NNP'), ('Bean',
'NNP')]), ('system', 'NN'), (',', ','), ('the', 'DT'),
Tree('ORGANIZATION', [('Galaxy', 'NNP')]), ('Tab', 'NNP'), ('8.9',
'CD'), ('Wifi', 'NNP'), ('tablet', 'NN'), (',', ','), ('the', 'DT'),
Tree('ORGANIZATION', [('Galaxy', 'NNP')]), ('Tab', 'NNP'), ('2',
'CD'), ('10.1', 'CD'), (',', ','), Tree('PERSON', [('Galaxy', 'NNP'),
('Rugby', 'NNP'), ('Pro', 'NNP')]), ('and', 'CC'), Tree('PERSON',
[('Galaxy', 'NNP'), ('S', 'NNP')]), ('III', 'NNP'), ('mini', 'NN'),
('.', '.')]), Tree('S', [Tree('PERSON', [('Apple', 'NNP')]),
('stated', 'VBD'), ('it', 'PRP'), ('had', 'VBD'), ('“', 'NNP'),
('acted', 'VBD'), ('quickly', 'RB'), ('and', 'CC'), ('diligently',
'RB'), ("''", "''"), ('in', 'IN'), ('order', 'NN'), ('to', 'TO'),
('``', '``'), ('determine', 'VB'), ('that', 'IN'), ('these', 'DT'),
('newly', 'RB'), ('released', 'VBN'), ('products', 'NNS'), ('do',
'VBP'), ('infringe', 'VB'), ('many', 'JJ'), ('of', 'IN'), ('the',
'DT'), ('same', 'JJ'), ('claims', 'NNS'), ('already', 'RB'),
('asserted', 'VBN'), ('by', 'IN'), Tree('PERSON', [('Apple', 'NNP')]),
('.', '.'), ("''", "''")]), Tree('S', [('In', 'IN'), Tree('GPE',
[('August', 'NNP')]), (',', ','), Tree('PERSON', [('Samsung',
'NNP')]), ('lost', 'VBD'), ('a', 'DT'), Tree('GSP', [('US', 'NNP')]),
('patent', 'NN'), ('case', 'NN'), ('to', 'TO'), Tree('GPE', [('Apple',
'NNP')]), ('and', 'CC'), ('was', 'VBD'), ('ordered', 'VBN'), ('to',
'TO'), ('pay', 'VB'), ('its', 'PRP$'), ('rival', 'JJ'), ('$', '$'),
('1.05bn', 'CD'), ('(', '('), ('£0.66bn', 'NN'), (')', ')'), ('in',
'IN'), ('damages', 'NNS'), ('for', 'IN'), ('copying', 'VBG'),
('features', 'NNS'), ('of', 'IN'), ('the', 'DT'), Tree('ORGANIZATION',
[('iPad', 'NN')]), ('and', 'CC'), Tree('ORGANIZATION', [('iPhone',
'NN')]), ('in', 'IN'), ('its', 'PRP$'), Tree('GPE', [('Galaxy',
'NNP')]), ('range', 'NN'), ('of', 'IN'), ('devices', 'NNS'), ('.',
'.')]), Tree('S', [Tree('GPE', [('Samsung', 'NNP')]), (',', ','),
('which', 'WDT'), ('is', 'VBZ'), ('the', 'DT'), ('world', 'NN'),
("'s", 'POS'), ('top', 'JJ'), ('mobile', 'NN'), ('phone', 'NN'),
('maker', 'NN'), (',', ','), ('is', 'VBZ'), ('appealing', 'VBG'),
('the', 'DT'), ('ruling', 'NN'), ('.', '.')]), Tree('S', [('A', 'DT'),
('similar', 'JJ'), ('case', 'NN'), ('in', 'IN'), ('the', 'DT'),
Tree('ORGANIZATION', [('UK', 'NNP')]), ('found', 'VBD'), ('in', 'IN'),
Tree('GPE', [('Samsung', 'NNP')]), ("'s", 'POS'), ('favour', 'NN'),
('and', 'CC'), ('ordered', 'VBD'), Tree('PERSON', [('Apple', 'NNP')]),
('to', 'TO'), ('publish', 'VB'), ('an', 'DT'), ('apology', 'NN'),
('making', 'VBG'), ('clear', 'JJ'), ('that', 'IN'), ('the', 'DT'),
```

```
Tree('LOCATION', [('South', 'JJ'), ('Korean', 'JJ')]), ('firm', 'NN'),
('had', 'VBD'), ('not', 'RB'), ('copied', 'VBN'), ('its', 'PRP$'),
('iPad', 'NN'), ('when', 'WRB'), ('designing', 'VBG'), ('its',
'PRP$'), ('own', 'JJ'), ('devices', 'NNS'), ('.', '.')])]
```

[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 pos_tags in pos_tags_per_sentence:
    print(constituent_parser.parse(pos_tags))

constituency_output_per_sentence.append(constituent_parser.parse(pos_t
ags))
```

```
(S
  (NP https/NN)
  :/:
  (NP
    //www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html/JJ)
  Documents/NNS
  (VP (V filed/VBN))
  to/TO
  (NP the/DT)
  San/NNP
  Jose/NNP
  (NP federal/JJ court/NN)
  (P in/IN)
  California/NNP
  (P on/IN)
  November/NNP
  23/CD
  (NP list/NN)
  six/CD
  Samsung/NNP
  products/NNS
  (VP (V running/VBG) (NP the/DT))
  ``/``
  Jelly/RB
```

```
  Bean/NNP
  ''/''
  and/CC
  ``/``
  Ice/NNP
  Cream/NNP
  Sandwich/NNP
  ''/''
  (VP (V operating/VBG))
  systems/NNS
  ,/,
  which/WDT
  Apple/NNP
  (VP (V claims/VBZ))
  (VP (V infringe/VB))
  its/PRP$
  patents/NNS
  ./.)
(S
  (NP The/DT)
  six/CD
  phones/NNS
  and/CC
  tablets/NNS
  (VP (V affected/VBN))
  (VP (V are/VBP) (NP the/DT))
  Galaxy/NNP
  S/NNP
  III/NNP
  ,/,
  (VP (V running/VBG) (NP the/DT new/JJ))
  Jelly/NNP
  Bean/NNP
  (NP system/NN)
  ,/,
  (NP the/DT)
  Galaxy/NNP
  Tab/NNP
  8.9/CD
  Wifi/NNP
  (NP tablet/NN)
  ,/,
  (NP 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
  (NP mini/NN)
  ./.)
(S
  Apple/NNP
  (VP (V stated/VBD))
  it/PRP
  (VP (V had/VBD))
  “/NNP
  (VP (V acted/VBD))
  quickly/RB
  and/CC
  diligently/RB
  ''/''
  (PP (P in/IN) (NP order/NN))
  to/TO
  ``/``
  (VP (V determine/VB) (PP (P that/IN) (NP these/DT)))
  newly/RB
  (VP (V released/VBN))
  products/NNS
  (VP (V do/VBP))
  (VP
    (V infringe/VB)
    (NP many/JJ)
    (PP (P of/IN) (NP the/DT same/JJ)))
  claims/NNS
  already/RB
  (VP (V asserted/VBN))
  (P by/IN)
  Apple/NNP
  ./.
  ''/'')
(S
  (P In/IN)
  August/NNP
  ,/,
  Samsung/NNP
  (VP (V lost/VBD) (NP a/DT))
  US/NNP
  (NP patent/NN case/NN)
  to/TO
  Apple/NNP
  and/CC
```

```
  (VP (V was/VBD))
  (VP (V ordered/VBN))
  to/TO
  (VP (V pay/VB))
  its/PRP$
  (NP rival/JJ)
  $/$
  1.05bn/CD
  (/(
  (NP £0.66bn/NN)
  )/)
  (P in/IN)
  damages/NNS
  (P for/IN)
  (VP (V copying/VBG))
  features/NNS
  (PP (P of/IN) (NP the/DT iPad/NN))
  and/CC
  (NP iPhone/NN)
  (P in/IN)
  its/PRP$
  Galaxy/NNP
  (NP range/NN)
  (P of/IN)
  devices/NNS
  ./.)
(S
  Samsung/NNP
  ,/,
  which/WDT
  (VP (V is/VBZ) (NP the/DT world/NN))
  's/POS
  (NP top/JJ mobile/NN phone/NN maker/NN)
  ,/,
  (VP (V is/VBZ))
  (VP (V appealing/VBG) (NP the/DT ruling/NN))
  ./.)
(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
  ./.)
```

```
print(constituency_output_per_sentence)
```

```
[Tree('S', [Tree('NP', [('https', 'NN')]), (':', ':'), Tree('NP',
[('//www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html', 'JJ')]), ('Documents',
'NNS'), Tree('VP', [Tree('V', [('filed', 'VBN')])]), ('to', 'TO'),
Tree('NP', [('the', 'DT')]), ('San', 'NNP'), ('Jose', 'NNP'),
Tree('NP', [('federal', 'JJ'), ('court', 'NN')]), Tree('P', [('in',
'IN')]), ('California', 'NNP'), Tree('P', [('on', 'IN')]),
('November', 'NNP'), ('23', 'CD'), Tree('NP', [('list', 'NN')]),
('six', 'CD'), ('Samsung', 'NNP'), ('products', 'NNS'), Tree('VP',
[Tree('V', [('running', 'VBG')]), Tree('NP', [('the', 'DT')])]),
('``', '``'), ('Jelly', 'RB'), ('Bean', 'NNP'), ("''", "''"), ('and',
'CC'), ('``', '``'), ('Ice', 'NNP'), ('Cream', 'NNP'), ('Sandwich',
'NNP'), ("''", "''"), Tree('VP', [Tree('V', [('operating', 'VBG')])]),
('systems', 'NNS'), (',', ','), ('which', 'WDT'), ('Apple', 'NNP'),
Tree('VP', [Tree('V', [('claims', 'VBZ')])]), Tree('VP', [Tree('V',
[('infringe', 'VB')])]), ('its', 'PRP$'), ('patents', 'NNS'), ('.',
'.')]), Tree('S', [Tree('NP', [('The', 'DT')]), ('six', 'CD'),
('phones', 'NNS'), ('and', 'CC'), ('tablets', 'NNS'), Tree('VP',
[Tree('V', [('affected', 'VBN')])]), Tree('VP', [Tree('V', [('are',
'VBP')]), Tree('NP', [('the', 'DT')])]), ('Galaxy', 'NNP'), ('S',
'NNP'), ('III', 'NNP'), (',', ','), Tree('VP', [Tree('V', [('running',
'VBG')]), Tree('NP', [('the', 'DT'), ('new', 'JJ')])]), ('Jelly',
'NNP'), ('Bean', 'NNP'), Tree('NP', [('system', 'NN')]), (',', ','),
Tree('NP', [('the', 'DT')]), ('Galaxy', 'NNP'), ('Tab', 'NNP'),
('8.9', 'CD'), ('Wifi', 'NNP'), Tree('NP', [('tablet', 'NN')]), (',',
','), Tree('NP', [('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'), Tree('NP', [('mini', 'NN')]), ('.', '.')]),
Tree('S', [('Apple', 'NNP'), Tree('VP', [Tree('V', [('stated',
```

```
'VBD')])]), ('it', 'PRP'), Tree('VP', [Tree('V', [('had', 'VBD')])]),
('“', 'NNP'), Tree('VP', [Tree('V', [('acted', 'VBD')])]), ('quickly',
'RB'), ('and', 'CC'), ('diligently', 'RB'), ("'", "'"), Tree('PP',
[Tree('P', [('in', 'IN')]), Tree('NP', [('order', 'NN')])]), ('to',
'TO'), ('``', '``'), Tree('VP', [Tree('V', [('determine', 'VB')]),
Tree('PP', [Tree('P', [('that', 'IN')]), Tree('NP', [('these',
'DT')])])]), ('newly', 'RB'), Tree('VP', [Tree('V', [('released',
'VBN')])]), ('products', 'NNS'), Tree('VP', [Tree('V', [('do',
'VBP')])]), Tree('VP', [Tree('V', [('infringe', 'VB')]), Tree('NP',
[('many', 'JJ')]), Tree('PP', [Tree('P', [('of', 'IN')]), Tree('NP',
[('the', 'DT'), ('same', 'JJ')])])]), ('claims', 'NNS'), ('already',
'RB'), Tree('VP', [Tree('V', [('asserted', 'VBN')])]), Tree('P',
[('by', 'IN')]), ('Apple', 'NNP'), ('.', '.'), ("'", "'")]),
Tree('S', [Tree('P', [('In', 'IN')]), ('August', 'NNP'), (',', ','),
('Samsung', 'NNP'), Tree('VP', [Tree('V', [('lost', 'VBD')]),
Tree('NP', [('a', 'DT')])]), ('US', 'NNP'), Tree('NP', [('patent',
'NN'), ('case', 'NN')]), ('to', 'TO'), ('Apple', 'NNP'), ('and',
'CC'), Tree('VP', [Tree('V', [('was', 'VBD')])]), Tree('VP',
[Tree('V', [('ordered', 'VBN')])]), ('to', 'TO'), Tree('VP',
[Tree('V', [('pay', 'VB')])]), ('its', 'PRP$'), Tree('NP', [('rival',
'JJ')]), ('$', '$'), ('1.05bn', 'CD'), ('(', '('), Tree('NP',
[('£0.66bn', 'NN')]), (')', ')'), Tree('P', [('in', 'IN')]),
('damages', 'NNS'), Tree('P', [('for', 'IN')]), Tree('VP', [Tree('V',
[('copying', 'VBG')])]), ('features', 'NNS'), Tree('PP', [Tree('P',
[('of', 'IN')]), Tree('NP', [('the', 'DT'), ('iPad', 'NN')])]),
('and', 'CC'), Tree('NP', [('iPhone', 'NN')]), Tree('P', [('in',
'IN')]), ('its', 'PRP$'), ('Galaxy', 'NNP'), Tree('NP', [('range',
'NN')]), Tree('P', [('of', 'IN')]), ('devices', 'NNS'), ('.', '.')]),
Tree('S', [('Samsung', 'NNP'), (',', ','), ('which', 'WDT'),
Tree('VP', [Tree('V', [('is', 'VBZ')]), Tree('NP', [('the', 'DT'),
('world', 'NN')])]), ("'s", 'POS'), Tree('NP', [('top', 'JJ'),
('mobile', 'NN'), ('phone', 'NN'), ('maker', 'NN')]), (',', ','),
Tree('VP', [Tree('V', [('is', 'VBZ')])]), Tree('VP', [Tree('V',
[('appealing', 'VBG')]), Tree('NP', [('the', 'DT'), ('ruling',
'NN')])]), ('.', '.')]), Tree('S', [Tree('NP', [('A', 'DT'),
('similar', 'JJ'), ('case', 'NN')]), Tree('PP', [Tree('P', [('in',
'IN')]), Tree('NP', [('the', 'DT')])]), ('UK', 'NNP'), Tree('VP',
[Tree('V', [('found', 'VBD')])]), Tree('P', [('in', 'IN')]),
('Samsung', 'NNP'), ("'s", 'POS'), Tree('NP', [('favour', 'NN')]),
('and', 'CC'), Tree('VP', [Tree('V', [('ordered', 'VBD')])]),
('Apple', 'NNP'), ('to', 'TO'), Tree('VP', [Tree('V', [('publish',
'VB')]), Tree('NP', [('an', 'DT'), ('apology', 'NN')])]), Tree('VP',
[Tree('V', [('making', 'VBG')]), Tree('NP', [('clear', 'JJ')]),
Tree('PP', [Tree('P', [('that', 'IN')]), Tree('NP', [('the', 'DT'),
('South', 'JJ'), ('Korean', 'JJ'), ('firm', 'NN')])])]), Tree('VP',
[Tree('V', [('had', 'VBD')])]), ('not', 'RB'), Tree('VP', [Tree('V',
[('copied', 'VBN')])]), ('its', 'PRP$'), Tree('NP', [('iPad', 'NN')]),
('when', 'WRB'), Tree('VP', [Tree('V', [('designing', 'VBG')])]),
```

```
('its', 'PRP$'), Tree('NP', [('own', 'JJ')]), ('devices', 'NNS'),
('.', '.')])]
```

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 NNP etc.''')

constituency_v2_output_per_sentence = []
for ner_tags in pos_tags_per_sentence:
    print(constituent_parser.parse(ner_tags))

constituency_v2_output_per_sentence.append(constituent_parser.parse(ne
r_tags))
```

```
(S
  (NP https/NN)
  :/:
  (NP
    //www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html/JJ)
  Documents/NNS
  (VP (V filed/VBN))
  to/TO
  (NP the/DT)
  San/NNP
  Jose/NNP
  (NP federal/JJ court/NN)
  (P in/IN)
  California/NNP
  (P on/IN)
  November/NNP
  23/CD
  (NP list/NN)
  six/CD
  Samsung/NNP
  products/NNS
  (VP (V running/VBG) (NP the/DT))
  ``/``
  Jelly/RB
  Bean/NNP
  ''/''
  and/CC
```

```
  ``/``
  Ice/NNP
  Cream/NNP
  Sandwich/NNP
  ''/''
  (VP (V operating/VBG))
  systems/NNS
  ,/,
  which/WDT
  Apple/NNP
  (VP (V claims/VBZ))
  (VP (V infringe/VB))
  its/PRP$
  patents/NNS
  ./.)
(S
  (NP The/DT)
  six/CD
  phones/NNS
  and/CC
  tablets/NNS
  (VP (V affected/VBN))
  (VP (V are/VBP) (NP the/DT))
  Galaxy/NNP
  S/NNP
  III/NNP
  ,/,
  (VP (V running/VBG) (NP the/DT new/JJ))
  Jelly/NNP
  Bean/NNP
  (NP system/NN)
  ,/,
  (NP the/DT)
  Galaxy/NNP
  Tab/NNP
  8.9/CD
  Wifi/NNP
  (NP tablet/NN)
  ,/,
  (NP 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
  (NP mini/NN)
  ./.)
(S
  Apple/NNP
  (VP (V stated/VBD))
  it/PRP
  (VP (V had/VBD))
  “/NNP
  (VP (V acted/VBD))
  quickly/RB
  and/CC
  diligently/RB
  ''/''
  (PP (P in/IN) (NP order/NN))
  to/TO
  ``/``
  (VP (V determine/VB) (PP (P that/IN) (NP these/DT)))
  newly/RB
  (VP (V released/VBN))
  products/NNS
  (VP (V do/VBP))
  (VP
    (V infringe/VB)
    (NP many/JJ)
    (PP (P of/IN) (NP the/DT same/JJ)))
  claims/NNS
  already/RB
  (VP (V asserted/VBN))
  (P by/IN)
  Apple/NNP
  ./.
  ''/'')
(S
  (P In/IN)
  August/NNP
  ,/,
  Samsung/NNP
  (VP (V lost/VBD) (NP a/DT))
  US/NNP
  (NP patent/NN case/NN)
  to/TO
  Apple/NNP
  and/CC
  (VP (V was/VBD))
  (VP (V ordered/VBN))
  to/TO
```

```
  (VP (V pay/VB))
  its/PRP$
  (NP rival/JJ)
  $/$
  1.05bn/CD
  (/(
  (NP £0.66bn/NN)
  )/)
  (P in/IN)
  damages/NNS
  (P for/IN)
  (VP (V copying/VBG))
  features/NNS
  (PP (P of/IN) (NP the/DT iPad/NN))
  and/CC
  (NP iPhone/NN)
  (P in/IN)
  its/PRP$
  Galaxy/NNP
  (NP range/NN)
  (P of/IN)
  devices/NNS
  ./.)
(S
  Samsung/NNP
  ,/,
  which/WDT
  (VP (V is/VBZ) (NP the/DT world/NN))
  's/POS
  (NP top/JJ mobile/NN phone/NN maker/NN)
  ,/,
  (VP (V is/VBZ))
  (VP (V appealing/VBG) (NP the/DT ruling/NN))
  ./.)
(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
  ./.)
```

```
print(constituency_v2_output_per_sentence)
```

```
[Tree('S', [Tree('NP', [('https', 'NN')]), (':', ':'), Tree('NP',
[('//www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html', 'JJ')]), ('Documents',
'NNS'), Tree('VP', [Tree('V', [('filed', 'VBN')])]), ('to', 'TO'),
Tree('NP', [('the', 'DT')]), ('San', 'NNP'), ('Jose', 'NNP'),
Tree('NP', [('federal', 'JJ'), ('court', 'NN')]), Tree('P', [('in',
'IN')]), ('California', 'NNP'), Tree('P', [('on', 'IN')]),
('November', 'NNP'), ('23', 'CD'), Tree('NP', [('list', 'NN')]),
('six', 'CD'), ('Samsung', 'NNP'), ('products', 'NNS'), Tree('VP',
[Tree('V', [('running', 'VBG')]), Tree('NP', [('the', 'DT')])]),
('``', '``'), ('Jelly', 'RB'), ('Bean', 'NNP'), ("''", "''"), ('and',
'CC'), ('``', '``'), ('Ice', 'NNP'), ('Cream', 'NNP'), ('Sandwich',
'NNP'), ("''", "''"), Tree('VP', [Tree('V', [('operating', 'VBG')])]),
('systems', 'NNS'), (',', ','), ('which', 'WDT'), ('Apple', 'NNP'),
Tree('VP', [Tree('V', [('claims', 'VBZ')])]), Tree('VP', [Tree('V',
[('infringe', 'VB')])]), ('its', 'PRP$'), ('patents', 'NNS'), ('.',
'.')]), Tree('S', [Tree('NP', [('The', 'DT')]), ('six', 'CD'),
('phones', 'NNS'), ('and', 'CC'), ('tablets', 'NNS'), Tree('VP',
[Tree('V', [('affected', 'VBN')])]), Tree('VP', [Tree('V', [('are',
'VBP')]), Tree('NP', [('the', 'DT')])]), ('Galaxy', 'NNP'), ('S',
'NNP'), ('III', 'NNP'), (',', ','), Tree('VP', [Tree('V', [('running',
'VBG')]), Tree('NP', [('the', 'DT'), ('new', 'JJ')])]), ('Jelly',
'NNP'), ('Bean', 'NNP'), Tree('NP', [('system', 'NN')]), (',', ','),
Tree('NP', [('the', 'DT')]), ('Galaxy', 'NNP'), ('Tab', 'NNP'),
('8.9', 'CD'), ('Wifi', 'NNP'), Tree('NP', [('tablet', 'NN')]), (',',
','), Tree('NP', [('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'), Tree('NP', [('mini', 'NN')]), ('.', '.')]),
Tree('S', [('Apple', 'NNP'), Tree('VP', [Tree('V', [('stated',
'VBD')])]), ('it', 'PRP'), Tree('VP', [Tree('V', [('had', 'VBD')])]),
('“', 'NNP'), Tree('VP', [Tree('V', [('acted', 'VBD')])]), ('quickly',
'RB'), ('and', 'CC'), ('diligently', 'RB'), ("''", "''"), Tree('PP',
```

```
[Tree('P', [('in', 'IN')]), Tree('NP', [('order', 'NN')])]), ('to',
'TO'), ('``', '``'), Tree('VP', [Tree('V', [('determine', 'VB')]),
Tree('PP', [Tree('P', [('that', 'IN')]), Tree('NP', [('these',
'DT')])])])]), ('newly', 'RB'), Tree('VP', [Tree('V', [('released',
'VBN')])]), ('products', 'NNS'), Tree('VP', [Tree('V', [('do',
'VBP')])]), Tree('VP', [Tree('V', [('infringe', 'VB')]), Tree('NP',
[('many', 'JJ')]), Tree('PP', [Tree('P', [('of', 'IN')]), Tree('NP',
[('the', 'DT'), ('same', 'JJ')])])]), ('claims', 'NNS'), ('already',
'RB'), Tree('VP', [Tree('V', [('asserted', 'VBN')])]), Tree('P',
[('by', 'IN')]), ('Apple', 'NNP'), ('.', '.'), ("''", "''")]),
Tree('S', [Tree('P', [('In', 'IN')]), ('August', 'NNP'), (',', ','),
('Samsung', 'NNP'), Tree('VP', [Tree('V', [('lost', 'VBD')]),
Tree('NP', [('a', 'DT')])]), ('US', 'NNP'), Tree('NP', [('patent',
'NN'), ('case', 'NN')]), ('to', 'TO'), ('Apple', 'NNP'), ('and',
'CC'), Tree('VP', [Tree('V', [('was', 'VBD')])]), Tree('VP',
[Tree('V', [('ordered', 'VBN')])]), ('to', 'TO'), Tree('VP',
[Tree('V', [('pay', 'VB')])]), ('its', 'PRP$'), Tree('NP', [('rival',
'JJ')]), ('$', '$'), ('1.05bn', 'CD'), ('(', '('), Tree('NP',
[('£0.66bn', 'NN')]), (')', ')'), Tree('P', [('in', 'IN')]),
('damages', 'NNS'), Tree('P', [('for', 'IN')]), Tree('VP', [Tree('V',
[('copying', 'VBG')])]), ('features', 'NNS'), Tree('PP', [Tree('P',
[('of', 'IN')]), Tree('NP', [('the', 'DT'), ('iPad', 'NN')])]),
('and', 'CC'), Tree('NP', [('iPhone', 'NN')]), Tree('P', [('in',
'IN')]), ('its', 'PRP$'), ('Galaxy', 'NNP'), Tree('NP', [('range',
'NN')]), Tree('P', [('of', 'IN')]), ('devices', 'NNS'), ('.', '.')]),
Tree('S', [('Samsung', 'NNP'), (',', ','), ('which', 'WDT'),
Tree('VP', [Tree('V', [('is', 'VBZ')]), Tree('NP', [('the', 'DT'),
('world', 'NN')])]), ("'s", 'POS'), Tree('NP', [('top', 'JJ'),
('mobile', 'NN'), ('phone', 'NN'), ('maker', 'NN')]), (',', ','),
Tree('VP', [Tree('V', [('is', 'VBZ')])]), Tree('VP', [Tree('V',
[('appealing', 'VBG')]), Tree('NP', [('the', 'DT'), ('ruling',
'NN')])]), ('.', '.')]), Tree('S', [Tree('NP', [('A', 'DT'),
('similar', 'JJ'), ('case', 'NN')]), Tree('PP', [Tree('P', [('in',
'IN')]), Tree('NP', [('the', 'DT')])]), ('UK', 'NNP'), Tree('VP',
[Tree('V', [('found', 'VBD')])]), Tree('P', [('in', 'IN')]),
('Samsung', 'NNP'), ("'s", 'POS'), Tree('NP', [('favour', 'NN')]),
('and', 'CC'), Tree('VP', [Tree('V', [('ordered', 'VBD')])]),
('Apple', 'NNP'), ('to', 'TO'), Tree('VP', [Tree('V', [('publish',
'VB')]), Tree('NP', [('an', 'DT'), ('apology', 'NN')])]), Tree('VP',
[Tree('V', [('making', 'VBG')]), Tree('NP', [('clear', 'JJ')]),
Tree('PP', [Tree('P', [('that', 'IN')]), Tree('NP', [('the', 'DT'),
('South', 'JJ'), ('Korean', 'JJ'), ('firm', 'NN')])])]), Tree('VP',
[Tree('V', [('had', 'VBD')])]), ('not', 'RB'), Tree('VP', [Tree('V',
[('copied', 'VBN')])]), ('its', 'PRP$'), Tree('NP', [('iPad', 'NN')]),
('when', 'WRB'), Tree('VP', [Tree('V', [('designing', 'VBG')])]),
('its', 'PRP$'), Tree('NP', [('own', 'JJ')]), ('devices', 'NNS'),
('.', '.')])]
```

[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)
sents = list(doc.sents)
print(f"The following are all the Name Entity Phrases in the text:
{doc.ents}")

print('The following sentences were extracted from the text and
tokenized:')
i = 0
for sent in sents:
    i += 1
    print(f'{i}. {sent}')

    x = 0
    for token in sent:
        x += 1
        print(f'Token {x} with POS and TAG: {token.text} {token.pos_}
{token.tag_}')
```

```
The following are all the Name Entity Phrases in the text: (San Jose,
California, November 23, six, Samsung, Jelly Bean, Apple, six, the
Galaxy S III, Jelly Bean, Galaxy Tab 8.9 Wifi, Galaxy Tab 2 10.1,
Apple, Apple, August, Samsung, US, Apple, 1.05bn, 0.66bn, iPad,
iPhone, Samsung, UK, Samsung, Apple, South Korean, iPad)
The following sentences were extracted from the text and tokenized:
1. https://www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html

Documents filed to the San Jose federal court in California on
November 23 list six Samsung products running the "Jelly Bean" and
"Ice Cream Sandwich" operating systems, which Apple claims infringe
its patents.

Token 1 with POS and TAG:
https://www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-
lawsuit-six-more-products-under-scrutiny.html PROPN NNP
Token 2 with POS and TAG:

 SPACE _SP
Token 3 with POS and TAG: Documents PROPN NNPS
Token 4 with POS and TAG: filed VERB VBD
Token 5 with POS and TAG: to ADP IN
Token 6 with POS and TAG: the DET DT
Token 7 with POS and TAG: San PROPN NNP
```

```
Token 8 with POS and TAG: Jose PROPN NNP
Token 9 with POS and TAG: federal ADJ JJ
Token 10 with POS and TAG: court NOUN NN
Token 11 with POS and TAG: in ADP IN
Token 12 with POS and TAG: California PROPN NNP
Token 13 with POS and TAG: on ADP IN
Token 14 with POS and TAG: November PROPN NNP
Token 15 with POS and TAG: 23 NUM CD
Token 16 with POS and TAG: list NOUN NN
Token 17 with POS and TAG: six NUM CD
Token 18 with POS and TAG: Samsung PROPN NNP
Token 19 with POS and TAG: products NOUN NNS
Token 20 with POS and TAG: running VERB VBG
Token 21 with POS and TAG: the DET DT
Token 22 with POS and TAG: " PUNCT ``
Token 23 with POS and TAG: Jelly PROPN NNP
Token 24 with POS and TAG: Bean PROPN NNP
Token 25 with POS and TAG: " PUNCT ''
Token 26 with POS and TAG: and CCONJ CC
Token 27 with POS and TAG: " PUNCT ``
Token 28 with POS and TAG: Ice PROPN NNP
Token 29 with POS and TAG: Cream PROPN NNP
Token 30 with POS and TAG: Sandwich PROPN NNP
Token 31 with POS and TAG: " PUNCT ''
Token 32 with POS and TAG: operating NOUN NN
Token 33 with POS and TAG: systems NOUN NNS
Token 34 with POS and TAG: , PUNCT ,
Token 35 with POS and TAG: which PRON WDT
Token 36 with POS and TAG: Apple PROPN NNP
Token 37 with POS and TAG: claims VERB VBZ
Token 38 with POS and TAG: infringe VERB VBP
Token 39 with POS and TAG: its PRON PRP$
Token 40 with POS and TAG: patents NOUN NNS
Token 41 with POS and TAG: . PUNCT .
Token 42 with POS and TAG:
 SPACE _SP
2. 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.

Token 1 with POS and TAG: The DET DT
Token 2 with POS and TAG: six NUM CD
Token 3 with POS and TAG: phones NOUN NNS
Token 4 with POS and TAG: and CCONJ CC
Token 5 with POS and TAG: tablets NOUN NNS
Token 6 with POS and TAG: affected VERB VBN
Token 7 with POS and TAG: are AUX VBP
Token 8 with POS and TAG: the DET DT
Token 9 with POS and TAG: Galaxy PROPN NNP
```

```
Token 10 with POS and TAG: S PROPN NNP
Token 11 with POS and TAG: III PROPN NNP
Token 12 with POS and TAG: , PUNCT ,
Token 13 with POS and TAG: running VERB VBG
Token 14 with POS and TAG: the DET DT
Token 15 with POS and TAG: new ADJ JJ
Token 16 with POS and TAG: Jelly PROPN NNP
Token 17 with POS and TAG: Bean PROPN NNP
Token 18 with POS and TAG: system NOUN NN
Token 19 with POS and TAG: , PUNCT ,
Token 20 with POS and TAG: the DET DT
Token 21 with POS and TAG: Galaxy PROPN NNP
Token 22 with POS and TAG: Tab PROPN NNP
Token 23 with POS and TAG: 8.9 NUM CD
Token 24 with POS and TAG: Wifi PROPN NNP
Token 25 with POS and TAG: tablet NOUN NN
Token 26 with POS and TAG: , PUNCT ,
Token 27 with POS and TAG: the DET DT
Token 28 with POS and TAG: Galaxy PROPN NNP
Token 29 with POS and TAG: Tab PROPN NNP
Token 30 with POS and TAG: 2 NUM CD
Token 31 with POS and TAG: 10.1 NUM CD
Token 32 with POS and TAG: , PUNCT ,
Token 33 with POS and TAG: Galaxy PROPN NNP
Token 34 with POS and TAG: Rugby PROPN NNP
Token 35 with POS and TAG: Pro PROPN NNP
Token 36 with POS and TAG: and CCONJ CC
Token 37 with POS and TAG: Galaxy PROPN NNP
Token 38 with POS and TAG: S PROPN NNP
Token 39 with POS and TAG: III PROPN NNP
Token 40 with POS and TAG: mini NOUN NN
Token 41 with POS and TAG: . PUNCT .
Token 42 with POS and TAG:
 SPACE _SP
3. Apple stated it had “acted quickly and diligently" in order to
"determine that these newly released products do infringe many of the
same claims already asserted by Apple.
Token 1 with POS and TAG: Apple PROPN NNP
Token 2 with POS and TAG: stated VERB VBD
Token 3 with POS and TAG: it PRON PRP
Token 4 with POS and TAG: had AUX VBD
Token 5 with POS and TAG: “ PUNCT ``
Token 6 with POS and TAG: acted VERB VBN
Token 7 with POS and TAG: quickly ADV RB
Token 8 with POS and TAG: and CCONJ CC
Token 9 with POS and TAG: diligently ADV RB
Token 10 with POS and TAG: " PUNCT ''
Token 11 with POS and TAG: in ADP IN
Token 12 with POS and TAG: order NOUN NN
```

```
Token 13 with POS and TAG: to PART TO
Token 14 with POS and TAG: " PUNCT ``
Token 15 with POS and TAG: determine VERB VB
Token 16 with POS and TAG: that SCONJ IN
Token 17 with POS and TAG: these DET DT
Token 18 with POS and TAG: newly ADV RB
Token 19 with POS and TAG: released VERB VBN
Token 20 with POS and TAG: products NOUN NNS
Token 21 with POS and TAG: do AUX VBP
Token 22 with POS and TAG: infringe VERB VB
Token 23 with POS and TAG: many ADJ JJ
Token 24 with POS and TAG: of ADP IN
Token 25 with POS and TAG: the DET DT
Token 26 with POS and TAG: same ADJ JJ
Token 27 with POS and TAG: claims NOUN NNS
Token 28 with POS and TAG: already ADV RB
Token 29 with POS and TAG: asserted VERB VBN
Token 30 with POS and TAG: by ADP IN
Token 31 with POS and TAG: Apple PROPN NNP
Token 32 with POS and TAG: . PUNCT .
4. "
In August, Samsung lost a US patent case to Apple and was ordered to
pay its rival $1.05bn (£0.66bn) in damages for copying features of the
iPad and iPhone in its Galaxy range of devices.
Token 1 with POS and TAG: " PUNCT ''
Token 2 with POS and TAG:
 SPACE _SP
Token 3 with POS and TAG: In ADP IN
Token 4 with POS and TAG: August PROPN NNP
Token 5 with POS and TAG: , PUNCT ,
Token 6 with POS and TAG: Samsung PROPN NNP
Token 7 with POS and TAG: lost VERB VBD
Token 8 with POS and TAG: a DET DT
Token 9 with POS and TAG: US PROPN NNP
Token 10 with POS and TAG: patent NOUN NN
Token 11 with POS and TAG: case NOUN NN
Token 12 with POS and TAG: to ADP IN
Token 13 with POS and TAG: Apple PROPN NNP
Token 14 with POS and TAG: and CCONJ CC
Token 15 with POS and TAG: was AUX VBD
Token 16 with POS and TAG: ordered VERB VBN
Token 17 with POS and TAG: to PART TO
Token 18 with POS and TAG: pay VERB VB
Token 19 with POS and TAG: its PRON PRP$
Token 20 with POS and TAG: rival ADJ JJ
Token 21 with POS and TAG: $ SYM $
Token 22 with POS and TAG: 1.05bn NUM CD
Token 23 with POS and TAG: ( PUNCT -LRB-
Token 24 with POS and TAG: £ SYM $
```

```
Token 25 with POS and TAG: 0.66bn NUM CD
Token 26 with POS and TAG: ) PUNCT -RRB-
Token 27 with POS and TAG: in ADP IN
Token 28 with POS and TAG: damages NOUN NNS
Token 29 with POS and TAG: for ADP IN
Token 30 with POS and TAG: copying VERB VBG
Token 31 with POS and TAG: features NOUN NNS
Token 32 with POS and TAG: of ADP IN
Token 33 with POS and TAG: the DET DT
Token 34 with POS and TAG: iPad PROPN NNP
Token 35 with POS and TAG: and CCONJ CC
Token 36 with POS and TAG: iPhone PROPN NNP
Token 37 with POS and TAG: in ADP IN
Token 38 with POS and TAG: its PRON PRP$
Token 39 with POS and TAG: Galaxy PROPN NNP
Token 40 with POS and TAG: range NOUN NN
Token 41 with POS and TAG: of ADP IN
Token 42 with POS and TAG: devices NOUN NNS
Token 43 with POS and TAG: . PUNCT .
5. Samsung, which is the world's top mobile phone maker, is appealing
the ruling.

Token 1 with POS and TAG: Samsung PROPN NNP
Token 2 with POS and TAG: , PUNCT ,
Token 3 with POS and TAG: which PRON WDT
Token 4 with POS and TAG: is AUX VBZ
Token 5 with POS and TAG: the DET DT
Token 6 with POS and TAG: world NOUN NN
Token 7 with POS and TAG: 's PART POS
Token 8 with POS and TAG: top ADJ JJ
Token 9 with POS and TAG: mobile ADJ JJ
Token 10 with POS and TAG: phone NOUN NN
Token 11 with POS and TAG: maker NOUN NN
Token 12 with POS and TAG: , PUNCT ,
Token 13 with POS and TAG: is AUX VBZ
Token 14 with POS and TAG: appealing VERB VBG
Token 15 with POS and TAG: the DET DT
Token 16 with POS and TAG: ruling NOUN NN
Token 17 with POS and TAG: . PUNCT .
Token 18 with POS and TAG:
 SPACE _SP
6. 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.
Token 1 with POS and TAG: A DET DT
Token 2 with POS and TAG: similar ADJ JJ
Token 3 with POS and TAG: case NOUN NN
Token 4 with POS and TAG: in ADP IN
Token 5 with POS and TAG: the DET DT
```

```
Token 6 with POS and TAG: UK PROPN NNP
Token 7 with POS and TAG: found VERB VBD
Token 8 with POS and TAG: in ADP IN
Token 9 with POS and TAG: Samsung PROPN NNP
Token 10 with POS and TAG: 's PART POS
Token 11 with POS and TAG: favour NOUN NN
Token 12 with POS and TAG: and CCONJ CC
Token 13 with POS and TAG: ordered VERB VBD
Token 14 with POS and TAG: Apple PROPN NNP
Token 15 with POS and TAG: to PART TO
Token 16 with POS and TAG: publish VERB VB
Token 17 with POS and TAG: an DET DT
Token 18 with POS and TAG: apology NOUN NN
Token 19 with POS and TAG: making NOUN NN
Token 20 with POS and TAG: clear ADJ JJ
Token 21 with POS and TAG: that SCONJ IN
Token 22 with POS and TAG: the DET DT
Token 23 with POS and TAG: South ADJ JJ
Token 24 with POS and TAG: Korean ADJ JJ
Token 25 with POS and TAG: firm NOUN NN
Token 26 with POS and TAG: had AUX VBD
Token 27 with POS and TAG: not PART RB
Token 28 with POS and TAG: copied VERB VBN
Token 29 with POS and TAG: its PRON PRP$
Token 30 with POS and TAG: iPad PROPN NNP
Token 31 with POS and TAG: when SCONJ WRB
Token 32 with POS and TAG: designing VERB VBG
Token 33 with POS and TAG: its PRON PRP$
Token 34 with POS and TAG: own ADJ JJ
Token 35 with POS and TAG: devices NOUN NNS
Token 36 with POS and TAG: . PUNCT .
```

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.

[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?

At first sight there are no noticable differences in the sentence splitting between NLTK and spaCy, however upon closer inspection there is a difference in the type. In NLTK the sentences are of type string and in spaCy they are a class.

- After checking the sentence splitting, select a sentence for which you expect interesting results and perhaps differences. Motivate your choice. We expect sentence 1 "https://www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-lawsuit-six-more-products-under-scrutiny.html

  Documents filed to the San Jose federal court in California on November 23 list six Samsung products running the "Jelly Bean" and "Ice Cream Sandwich" operating systems, which Apple claims infringe its patents." to have interesting results and differences when analysed with NLTK versus spaCy. NLTK splits the link into 3 tokens namely https, :, //www.telegraph.co.uk/technology/apple/9702716/Apple-Samsung-lawsuit-six-more-products-under-scrutiny.html, while spaCy considers the link as one single token. Furthermore, spaCy considers the empty space after the link a token with the tag SPACE_SP. Whereas, with NLTK there is no token between the link and the token "Document".

- 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.

  The following tokens have different tags with NLTK and spaCy. Each token is followed by the tag given by NLTK and then the tag given by spaCy:

  "Document" NNS, NNPS

  "filed" VBN, VBD

  "to" TO, IN

  "operating" VBJ, NN

  "infringe" VB, VBP

[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?

  With NLTK the output for the Named Entitiy Phrase "San Jose" is (ORGANIZATION San/NNP Jose/NNP), for spaCy we have a set of all NEP's in the text (San Jose, California, November 23, six, Samsung, Jelly Bean, Apple, six, the Galaxy S III, Jelly Bean, Galaxy Tab 8.9 Wifi, Galaxy Tab 2 10.1, Apple, Apple, August, Samsung, US, Apple, 1.05bn, 0.66bn, iPad, iPhone, Samsung, UK, Samsung, Apple, South Korean, iPad).

SpaCy performs better beacuse NLTK uses chunking which requires more manual rule creation. On the other hand spaCy shows all of the NEP's in one go which is more efficient.

[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.

In the cells below we run constituency parsing using NLTK and dependency parsing using spaCy on sentence 2, respectively. With constituency parsing a sentence is represented as a hierarchical structure where words are grouped into phrases. The output consists of a tree structure.

Dependency parsing, on the other hand, focuses on grammatical relationships between words. The output is a graph where words are connected based on their dependencies, such as subject-verb and verb-object relationships.

In NLTK, the output looks like a structured tree, while in spaCy the output is a dependency graph that visually represents how words are related.

```
constituency_v2_output_per_sentence = []
i = 0
for ner_tags in pos_tags_per_sentence:
    i += 1
    if i == 2:
        sentence = constituent_parser.parse(ner_tags)
        constituency_v2_output_per_sentence.append(sentence)
        print(constituency_v2_output_per_sentence)
```

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

```
from spacy import displacy
doc = nlp(text)
sents = list(doc.sents)
```

```
sentence = sents[1]

displacy.render(sentence, jupyter=True, style='dep')
```

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

End of this notebook