+ Code + Text

Program 9 : Write a program for POS Tagging and Word Embeddings.

```
import nltk
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
True
```

Implementing POS tagging

```
ıses·a·breadth-first·search·and·Hash·Tree·to·calculate·the·itemset·associations·efficientl
```

```
to/TO
generate/VB
(mychunk association/NN rules,/NN and/CC)
it/PRP
is/VBZ
designed/VBN
to/TO
work/VB
on/IN
the/DT
(mychunk databases/NNS)
that/WDT
contain/VBP

transactions./IN
With/IN
the/DT
(mychunk help/NN)
of/IN
these/DT
(mychunk association/NN rule,/VBD)
it/PRP
determines/VBZ
how/WRB
strongly/RB
```

```
(mychunk or/CC)
how/WRB
(mychunk weakly/JJ)
two/CD
(mychunk objects/NNS)
are/VBP
(mychunk connected./JJ)
This/DT
(mychunk algorithm/JJ)
uses/VBZ
a/DT
(mychunk breadth-first/JJ)
(mychunk search/NN and/CC)
(mychunk Hash/NNP Tree/NNP)
to/TO
calculate/VB
the/DT
(mychunk itemset/NN associations/NNS)
efficiently./VBP
It/PRP
is/VBZ
the/DT
(mychunk iterative/JJ)
(mychunk process/NN)
for/IN
finding/VBG
the/DT
(mychunk frequent/JJ)
(mychunk itemsets/NNS)
from/IN
the/DT
(mychunk large/JJ)
(mychunk dataset./NN))
```

```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

Chunking : Entity detection

```
tokens = nltk.word_tokenize(text_original)
print(tokens)
tag = nltk.pos_tag(tokens)
print(tag)
grammar = "NP: {<DT>?<JJ>*<NN>}"
cp  =nltk.RegexpParser(grammar)
result = cp.parse(tag)
print(result)
```

```
(NP association/NN)
rules/NNS
,/,
and/CC
it/PRP
```

```
is/VBZ
designed/VBN
to/TO
work/VB
on/IN
the/DT
databases/NNS
that/WDT
contain/VBP
transactions/NNS
./.
With/IN
(NP the/DT help/NN)
of/IN
(NP these/DT association/NN)
(NP rule/NN)
,/,
it/PRP
determines/VBZ
how/WRB
strongly/RB
or/CC
how/WRB
weakly/JJ
two/CD
objects/NNS
are/VBP
connected/VBN
./.
This/DT
algorithm/JJ
uses/VBZ
(NP a/DT breadth-first/JJ search/NN)
and/CC
Hash/NNP
Tree/NNP
to/TO
calculate/VB
(NP the/DT itemset/NN)
associations/NNS
efficiently/RB
./.
It/PRP
is/VBZ
(NP the/DT iterative/JJ process/NN)
for/IN
finding/VBG
the/DT
frequent/JJ

itemsets/NNS
from/IN
(NP the/DT large/JJ dataset/NN)
./.)
```

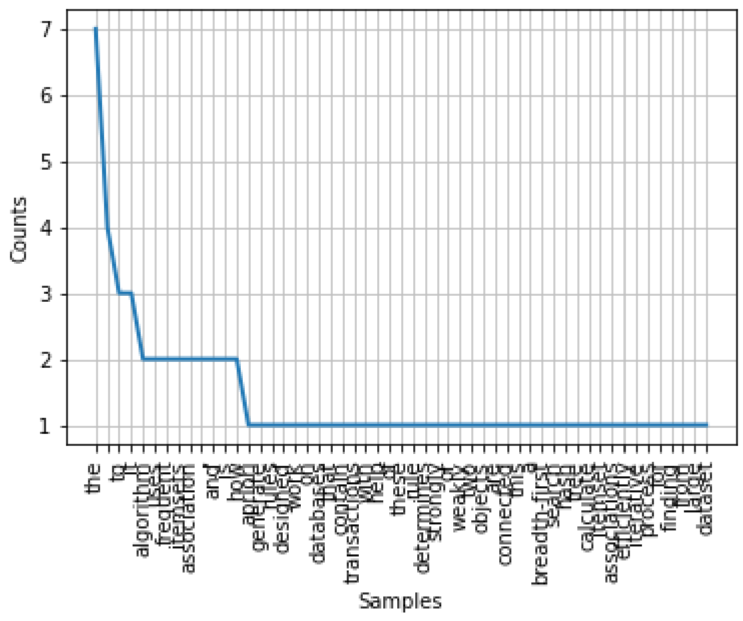
Counting POS tags

```
from collections import Counter
ower_case = text_original.lower()
tokens = nltk.word_tokenize(lower_case)
tags = nltk.pos_tag(tokens)
counts = Counter( tag for word,  tag in tags)
print(counts)
```

```
Counter({'NN': 11, 'DT': 10, 'JJ': 8, 'NNS': 7, 'VBZ': 5, 'IN': 5, '.': 4, 'TO': 3,
```

Frequency distribution

```
fd = nltk.FreqDist(tokens)
fd.plot()
```



Obtaining collocations (Bigrams and Trigrams)

```
bigrams_list = list(nltk.bigrams(tokens))
print(bigrams_list)
```

```
[('the', 'apriori'), ('apriori', 'algorithm'), ('algorithm', 'uses'), ('uses', 'frequ
```

```
trigrams_list = list(nltk.trigrams(tokens))
print(trigrams_list)
```

```
[('the', 'apriori', 'algorithm'), ('apriori', 'algorithm', 'uses'), ('algorithm', 'us
```

Word-embeddings

```
# Load library
from nltk.corpus import stopwords
# You will have to download the set of stop words the first time
import nltk
nltk.download('stopwords')
# Load stop words
stop_words = stopwords.words('english')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
#Using Wordvec2
from sklearn.feature_extraction.text import CountVectorizer
vectorizer=CountVectorizer()
data_corpus=["The Apriori algorithm uses frequent itemsets to generate association rules,
vocabulary=vectorizer.fit(data_corpus)
X= vectorizer.transform(data_corpus)
print(X.toarray())
vocabulary.get_feature_names()
```

```
[[2 2 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 2 3 1 2 1 1 1 1 1 1 1 1
  1 1 1 1 7 1 1 3 1 1 1 2 1 1 1]]
['algorithm',
 'and',
 'apriori',
 'are',
 'association',
 'associations',
 'breadth',
 'calculate',
 'connected',
 'contain',
 'databases',
 'dataset',
 'designed',
 'determines',
 'efficiently',
 'finding',
 'first',
 'for',
 'frequent',
 'from',
 'generate',
 'hash',
 'help',
 'how',
 'is',
 'it',
 'itemset',
 'itemsets',
 'iterative',
 'large',
 'objects',
 'of',
 'on',
 'or',
 'process',
 'rule',
 'rules',
 'search',
 'strongly',
 'that',
 'the',
 'these',
 'this',
 'to',
 'transactions',
 'tree',
 'two',
 'uses',
 'weakly',
 'with',
 'work']
```

```
#Using Gensim
```

```
nltk.download('all')
```

```
[nltk_data]    | Downloading package maxent_ne_chunker to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data]    | Downloading package punkt to /root/nltk_data...
[nltk_data]    |   Package punkt is already up-to-date!
[nltk_data]    | Downloading package book_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping grammars/book_grammars.zip.
[nltk_data]    | Downloading package sample_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping grammars/sample_grammars.zip.
[nltk_data]    | Downloading package spanish_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping grammars/spanish_grammars.zip.
[nltk_data]    | Downloading package basque_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping grammars/basque_grammars.zip.
[nltk_data]    | Downloading package large_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping grammars/large_grammars.zip.
[nltk_data]    | Downloading package tagsets to /root/nltk_data...
[nltk_data]    |   Unzipping help/tagsets.zip.
[nltk_data]    | Downloading package snowball_data to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    | Downloading package bllip_wsj_no_aux to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping models/bllip_wsj_no_aux.zip.
[nltk_data]    | Downloading package word2vec_sample to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping models/word2vec_sample.zip.
[nltk_data]    | Downloading package panlex_swadesh to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    | Downloading package mte_teip5 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/mte_teip5.zip.

[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger is already up-
[nltk_data]    |       to-date!
```

```
[nltk_data]    | Downloading package averaged_perceptron_tagger_ru to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping
[nltk_data]    |       taggers/averaged_perceptron_tagger_ru.zip.
[nltk_data]    | Downloading package perluniprops to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping misc/perluniprops.zip.
[nltk_data]    | Downloading package nonbreaking_prefixes to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/nonbreaking_prefixes.zip.
[nltk_data]    | Downloading package vader_lexicon to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    | Downloading package porter_test to /root/nltk_data...
[nltk_data]    |   Unzipping stemmers/porter_test.zip.
[nltk_data]    | Downloading package wmt15_eval to /root/nltk_data...
[nltk_data]    |   Unzipping models/wmt15_eval.zip.
[nltk_data]    | Downloading package mwa_ppdb to /root/nltk_data...
[nltk_data]    |   Unzipping misc/mwa_ppdb.zip.
[nltk_data]    |
[nltk_data]  Done downloading collection all
True
```

```
import nltk
import gensim
from nltk.corpus import abc

model= gensim.models.Word2Vec(abc.sents())
X= list(model.wv.vocab)
data=model.most_similar('science')
print(data)
```

```
[('law', 0.9403499364852905), ('general', 0.9275458455085754), ('policy', 0.92519211
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: DeprecationWarning: (
  import sys
```

```
#1.To Find the degree of similarity between two words
model.similarity('woman','man')
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: (

0.9352239
```

```
model.similarity('boat','ship')
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: DeprecationWarning: (
  """Entry point for launching an IPython kernel.
0.9053039
```

```
#2.To Find the odd one out from a set of words
model.doesnt_match('breakfast cereal dinner lunch'.split())
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: (

/usr/local/lib/python3.7/dist-packages/gensim/models/keyedvectors.py:895: FutureWarni
  vectors = vstack(self.word_vec(word, use_norm=True) for word in used_words).astype(
'dinner'
```

```
#3.Getting word vectors of a word
word_vectors = model['science']
word_vectors
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: (

array([-0.10011931, -0.341144  ,  0.09218448, -0.47717318, -0.24085222,
        0.03314723,  0.29564837, -0.09993076, -0.49595845, -0.03311085,
        0.39076772,  0.37152696,  0.5569941 , -0.19554017, -0.08119448,
        0.32476667, -0.10634214,  0.23837957, -0.36516923,  0.32095012,
        0.3782207 , -0.28557962, -0.31299403,  0.1587764 ,  0.2605071 ,
        0.04869302,  0.05717864,  0.0870984 ,  0.24957213, -0.01582777,
        0.23588116, -0.11062937,  0.26411793,  0.27790704, -0.00585647,
        0.05394433,  0.27971494,  0.48528555, -0.3210489 ,  0.40206233,
        0.06894344,  0.04103887, -0.0520575 ,  0.17915024,  0.06959112,
        0.26795033, -0.05289154,  0.43142503, -0.18717143,  0.3006417 ,
        0.34157106, -0.552793  , -0.13877136, -0.01814231, -0.2841337 ,
        0.10410302, -0.12432279,  0.24023694,  0.2657793 , -0.3491189 ,
        0.31979972,  0.1383694 , -0.07176611,  0.24089631, -0.26650172,
       -0.459595  , -0.12666653, -0.00379726,  0.16939965,  0.35632008,
       -0.27436304, -0.18867153, -0.3520612 , -0.09935838, -0.24125136,
       -0.44397894, -0.07189265,  0.07477235,  0.17232987, -0.06283064,
       -0.17903367, -0.30875754,  0.07456908,  0.5338551 , -0.36556947,
       -0.19004126, -0.24947084, -0.17440559,  0.26127008,  0.07161208,
       -0.10773071,  0.0349858 , -0.01452465,  0.09074266, -0.05123617,
        0.39108428,  0.2528702 ,  0.3259142 ,  0.35310516,  0.3098357 ],
      dtype=float32)
```