Practical 7

Title: Text Analytics

[ ]:
```python
import nltk
nltk.download("punkt")
nltk.download("stopwords")
nltk.download("wordnet")
nltk.download("averaged_perceptron_tagger")
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data…
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

[ ]:
```
True
```

Tokenization

[ ]:
```python
from nltk import word_tokenize, sent_tokenize
```

[ ]:
```python
corpus = "Tokenization is the first step in text analytics. The process of␣
 ↪breaking down a text paragraph into smaller chunkssuch as words or sentences␣
 ↪is called Tokenization."
```

[ ]:
```python
print(word_tokenize(corpus))
print(sent_tokenize(corpus))
```

```
['Tokenization', 'is', 'the', 'first', 'step', 'in', 'text', 'analytics', '.',
'The', 'process', 'of', 'breaking', 'down', 'a', 'text', 'paragraph', 'into',
'smaller', 'chunkssuch', 'as', 'words', 'or', 'sentences', 'is', 'called',
'Tokenization', '.']
['Tokenization is the first step in text analytics.', 'The process of breaking
down a text paragraph into smaller chunkssuch as words or sentences is called
Tokenization.']
```

POS tagging

```
[ ]: from nltk import pos_tag
```

```
[ ]: tokens = word_tokenize(corpus)
     print(pos_tag(tokens))
```

```
[('Tokenization', 'NN'), ('is', 'VBZ'), ('the', 'DT'), ('first', 'JJ'), ('step',
'NN'), ('in', 'IN'), ('text', 'JJ'), ('analytics', 'NNS'), ('.', '.'), ('The',
'DT'), ('process', 'NN'), ('of', 'IN'), ('breaking', 'VBG'), ('down', 'RP'),
('a', 'DT'), ('text', 'NN'), ('paragraph', 'NN'), ('into', 'IN'), ('smaller',
'JJR'), ('chunkssuch', 'NN'), ('as', 'IN'), ('words', 'NNS'), ('or', 'CC'),
('sentences', 'NNS'), ('is', 'VBZ'), ('called', 'VBN'), ('Tokenization', 'NN'),
('.', '.')]
```

Stop word removal

```
[ ]: from nltk.corpus import stopwords
     stop_words = set(stopwords.words("english"))
```

```
[ ]: tokens = word_tokenize(corpus)
     cleaned_tokens = []
     for token in tokens:
       if (token not in stop_words):
         cleaned_tokens.append(token)
     print(cleaned_tokens)
```

```
['Tokenization', 'first', 'step', 'text', 'analytics', '.', 'The', 'process',
'breaking', 'text', 'paragraph', 'smaller', 'chunkssuch', 'words', 'sentences',
'called', 'Tokenization', '.']
```

Stemming

```
[ ]: from nltk.stem import PorterStemmer
```

```
[ ]: stemmer = PorterStemmer()
```

```
[ ]: stemmed_tokens = []
     for token in cleaned_tokens:
       stemmed = stemmer.stem(token)
       stemmed_tokens.append(stemmed)
     print(stemmed_tokens)
```

```
['token', 'first', 'step', 'text', 'analyt', '.', 'the', 'process', 'break',
'text', 'paragraph', 'smaller', 'chunkssuch', 'word', 'sentenc', 'call',
'token', '.']
```

Lemmatization

[ ]:
```
from nltk.stem import WordNetLemmatizer
```

[ ]:
```
lemmatizer = WordNetLemmatizer()
```

[ ]:
```
lemmatized_tokens = []
for token in cleaned_tokens:
  lemmatized = lemmatizer.lemmatize(token)
  lemmatized_tokens.append(lemmatized)
print(lemmatized_tokens)
```

```
['Tokenization', 'first', 'step', 'text', 'analytics', '.', 'The', 'process',
'breaking', 'text', 'paragraph', 'smaller', 'chunkssuch', 'word', 'sentence',
'called', 'Tokenization', '.']
```

[ ]:
```

```

TF-IDF

[ ]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

[ ]:
```
corpus = [
    "Tokenization is the first step in text analytics. The process of breaking␣
 ↪down a text paragraph into smaller chunks such as words or sentences is␣
 ↪called Tokenization."
]
```

[ ]:
```
vectorizer = TfidfVectorizer()
```

[ ]:
```
matrix = vectorizer.fit(corpus)
matrix.vocabulary_
```

[ ]:
```
{'tokenization': 20,
 'is': 9,
 'the': 19,
 'first': 6,
 'step': 16,
 'in': 7,
 'text': 18,
 'analytics': 0,
 'process': 13,
 'of': 10,
 'breaking': 2,
 'down': 5,
 'paragraph': 12,
 'into': 8,
 'smaller': 15,
 'chunks': 4,
 'such': 17,
 'as': 1,
 'words': 21,
 'or': 11,
 'sentences': 14,
 'called': 3}
```

[ ]:

```
tfidf_matrix = vectorizer.transform(corpus)
print(tfidf_matrix)
```

```
  (0, 21)	0.17149858514250882
  (0, 20)	0.34299717028501764
  (0, 19)	0.34299717028501764
  (0, 18)	0.34299717028501764
  (0, 17)	0.17149858514250882
  (0, 16)	0.17149858514250882
  (0, 15)	0.17149858514250882
  (0, 14)	0.17149858514250882
  (0, 13)	0.17149858514250882
  (0, 12)	0.17149858514250882
  (0, 11)	0.17149858514250882
  (0, 10)	0.17149858514250882
  (0, 9)	0.34299717028501764
  (0, 8)	0.17149858514250882
  (0, 7)	0.17149858514250882
  (0, 6)	0.17149858514250882
  (0, 5)	0.17149858514250882
  (0, 4)	0.17149858514250882
  (0, 3)	0.17149858514250882
  (0, 2)	0.17149858514250882
  (0, 1)	0.17149858514250882
  (0, 0)	0.17149858514250882
```

[ ]:

```
print(vectorizer.get_feature_names_out())
```

```
['analytics' 'as' 'breaking' 'called' 'chunks' 'down' 'first' 'in' 'into'
 'is' 'of' 'or' 'paragraph' 'process' 'sentences' 'smaller' 'step' 'such'
 'text' 'the' 'tokenization' 'words']
```