BSDS-209 - NLP for Data Science

Course Instructor - Mr. Nitish Patil

Gauri Sharan - BSc Data Science (Sem 3)

January 6, 2024

Outline

What is NLP?

- Natural Language Processing (NLP) is a subfield of artificial intelligence (AI) that focuses on the interaction between computers and humans using natural language.
- It involves the development of algorithms and models to enable computers to understand, interpret, and generate human language.

Applications of NLP

- Sentiment Analysis
- Named Entity Recognition
- Machine Translation
- Text Summarization
- Speech Recognition
- Chatbots and Virtual Assistants

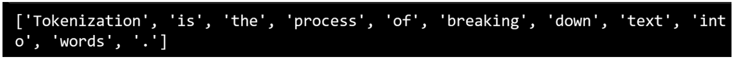
NLP Pipeline

1. Data Acquisition
2. Text Preprocessing
    - Text Cleaning - HTML tag removing, emoji handling, Spelling checker, etc.
    - Basic Preprocessing - (word tokenization, sentence tokenization, stop word removal
    - Advance Preprocessing - POS tagging, Parsing, etc.
3. Featured Engineering
    - One-Hot Encoder
    - Bag-of-Words
    - n-grams
    - tf-idf
    - word2vec
4. Model Building
5. Evaluation and Deployment

Tokenization in Python

```python
import nltk
from nltk.tokenize import word_tokenize

text = ”Tokenization is the process of breaking down
text into words.”
tokens = word_tokenize(text)
print(tokens)
```

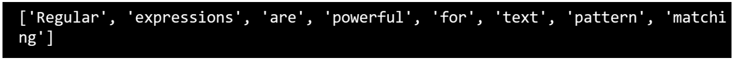


```
['Tokenization', 'is', 'the', 'process', 'of', 'breaking', 'down', 'text', 'int
o', 'words', '.']
```

Regex in Python

```
import re

text = "Regular expressions are powerful for text
pattern matching."
pattern = r'\b\w+\b'  # Matches words
matches = re.findall(pattern, text)
print(matches)
```

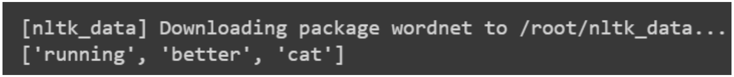


```
['Regular', 'expressions', 'are', 'powerful', 'for', 'text', 'pattern', 'matchi
ng']
```

Lemmatization in Python

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
words = ["running", "better", "cats"]
lemmatized_words = [lemmatizer.lemmatize(word) for
word in words]
print(lemmatized_words)
```

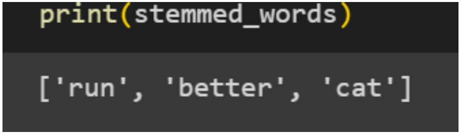


```
[nltk_data] Downloading package wordnet to /root/nltk_data...
['running', 'better', 'cat']
```

Stemming in Python

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()
words = ["running", "better", "cats"]
stemmed_words = [stemmer.stem(word) for word in words]
print(stemmed_words)
```



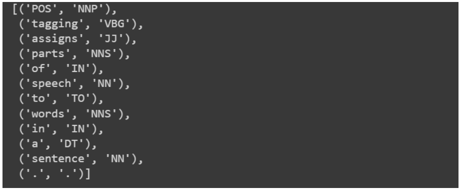
```
print(stemmed_words)

['run', 'better', 'cat']
```

POS Tagging in Python

```
import nltk

text = "POS tagging assigns parts of speech to words
in a sentence."
pos_tags = nltk.pos_tag(nltk.word_tokenize(text))
print(pos_tags)
```

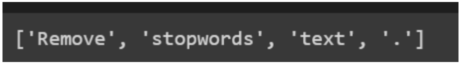


```
[('POS', 'NNP'),
 ('tagging', 'VBG'),
 ('assigns', 'JJ'),
 ('parts', 'NNS'),
 ('of', 'IN'),
 ('speech', 'NN'),
 ('to', 'TO'),
 ('words', 'NNS'),
 ('in', 'IN'),
 ('a', 'DT'),
 ('sentence', 'NN'),
 ('.', '.')]
```

Stopwords Removal in Python

```
from nltk.corpus import stopwords

text = "Remove stopwords from this text."
stopwords_set = set(stopwords.words('english'))
filtered_words = [word for word in
nltk.word_tokenize(text) if word.lower() not in
stopwords_set]
print(filtered_words)
```

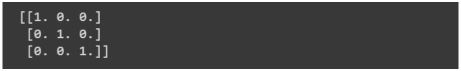


```
['Remove', 'stopwords', 'text', '.']
```

One-Hot Encoding in Python

```
from sklearn.preprocessing import OneHotEncoder

categories = ['category_A', 'category_B',
'category_C']
encoder = OneHotEncoder()
encoded_data = encoder.fit_transform([[cat] for cat
in categories]).toarray()
print(encoded_data)
```

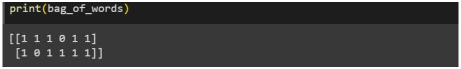


```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

Bag of Words in Python

```
from sklearn.feature_extraction.text import
CountVectorizer

corpus = ['This is the first document.', 'This is the
second document.']
vectorizer = CountVectorizer()
bag_of_words =
vectorizer.fit_transform(corpus).toarray()
print(bag_of_words)
```



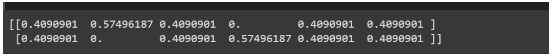
```
print(bag_of_words)

[[1 1 1 0 1 1]
 [1 0 1 1 1 1]]
```

TF-IDF in Python

```python
from sklearn.feature_extraction.text import
TfidfVectorizer

corpus = ['This is the first document.', 'This
is the second document.']
vectorizer = TfidfVectorizer()
tfidf_matrix =
vectorizer.fit_transform(corpus).toarray()
print(tfidf_matrix)
```

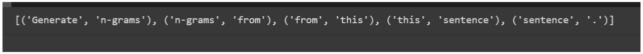


```
[[0.4090901  0.57496187 0.4090901  0.         0.4090901  0.4090901 ]
 [0.4090901  0.         0.4090901  0.57496187 0.4090901  0.4090901 ]]
```

N-grams in Python

```python
from nltk import ngrams

sentence = "Generate n-grams from this sentence."
n = 2  # Change to desired n-gram size
ngram_list =
list(ngrams(nltk.word_tokenize(sentence), n))
print(ngram_list)
```



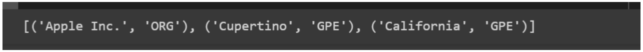
```
[('Generate', 'n-grams'), ('n-grams', 'from'), ('from', 'this'), ('this', 'sentence'), ('sentence', '.')]
```

Named Entity Recognition in Python

```
import spacy

nlp = spacy.load("en_core_web_sm")
text = "Apple Inc. is a technology company based in
Cupertino, California."
doc = nlp(text)

entities = [(ent.text, ent.label_) for ent in
doc.ents]
print(entities)
```



```
[('Apple Inc.', 'ORG'), ('Cupertino', 'GPE'), ('California', 'GPE')]
```