Лабораторна робота 3. Моделі текстових даних

Мета роботи: Ознайомитись з основними текстовими моделями та їх створення за допомогою бібліотек scikit-learn та genism.

ІП-13 Ал Хадам Мурат

Зчитати файл doc1. Вважати кожен рядок окремим документом корпусу. Виконати попередню обробку корпусу. 1)Представити корпус як модель «Сумка слів». Вивести вектор для слова «film». 2)Представити корпус як модель TF-IDF. Спробувати кластеризувати документи за допомогою ієрархічної агломераційної кластеризації. 3)Представити корпус як модель Word2Vec. Знайти подібні слова до слів shrimp, economy.

```
from sklearn.feature_extraction.text import CountVectorizer,
TfidfVectorizer
from sklearn.cluster import AgglomerativeClustering
from nltk.tokenize import word_tokenize

from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))

from nltk import WordPunctTokenizer
import string


def clean_text(document):
    tokens = WordPunctTokenizer().tokenize(document.lower())
    cleaned_tokens = [token for token in tokens if token not in
stop_words and token not in string.punctuation]
    return ' '.join(cleaned_tokens)

with open('doc1.txt', 'r') as file:
    documents = file.readlines()

documents
```

```
['Whisk the lime juice, oil, chipotle powder, salt, and cumin together
in a large bowl. Add the shrimp and toss to combine.\n',
 'Growth in Japan evaporated in the three months to September,
sparking renewed concern about an economy not long out of a decade-
```

```
long trough.\n',
 'The independent film festival will feature two new international
cinema competitions.\n',
 'Serve the shrimp with the tortillas and salsa.\n',
 'Twelve films competing in the new world cinema documentary
category.\n',
 'The economy had stagnated throughout the 1990s.\n',
 'Actor Daniel Day-Lewis is to be presented with an award for his
career in film at the Berlin Film Festival.']

corpus = [clean_text(document) for document in documents]
corpus

['whisk lime juice oil chipotle powder salt cumin together large bowl
add shrimp toss combine',
 'growth japan evaporated three months september sparking renewed
concern economy long decade long trough',
 'independent film festival feature two new international cinema
competitions',
 'serve shrimp tortillas salsa',
 'twelve films competing new world cinema documentary category',
 'economy stagnated throughout 1990s',
 'actor daniel day lewis presented award career film berlin film
festival']
```

1) Представити корпус як модель «Сумка слів». Вивести вектор для слова «film».

```
vectorizer = CountVectorizer(min_df=0., max_df=1.)  # Використання
коректних параметрів
X = vectorizer.fit_transform(corpus)

print("Вектор для слова 'film':")
film_index = vectorizer.vocabulary_.get('film')
film_vector = X[:, 23].toarray()

print(film_vector)

Вектор для слова 'film':
[[0]
 [0]
 [1]
 [0]
 [0]
 [0]
 [2]]
```

2) Представити корпус як модель TF-IDF. Спробувати кластеризувати документи за допомогою ієрархічної агломераційної кластеризації.

```
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(corpus)

print(tfidf_matrix[1])
```

```
  (0, 25)  0.25246534896358647
  (0, 28)  0.25246534896358647
  (0, 20)  0.25246534896358647
  (0, 47)  0.25246534896358647
  (0, 34)  0.25246534896358647
  (0, 42)  0.25246534896358647
  (0, 45)  0.25246534896358647
  (0, 39)  0.25246534896358647
  (0, 13)  0.25246534896358647
  (0, 19)  0.2095679211864678
  (0, 33)  0.5049306979271729
  (0, 17)  0.25246534896358647
  (0, 52)  0.25246534896358647
```

```
agg_cluster = AgglomerativeClustering(n_clusters=2,
linkage='complete')
clusters = agg_cluster.fit_predict(tfidf_matrix.toarray())

print("Hierarchical Agglomeration Clustering Results:")
for i, cluster in enumerate(clusters):
    print(f"Document {i+1}: Cluster {cluster}")
```

```
Hierarchical Agglomeration Clustering Results:
Document 1: Cluster 0
Document 2: Cluster 0
Document 3: Cluster 1
Document 4: Cluster 0
Document 5: Cluster 1
Document 6: Cluster 0
Document 7: Cluster 1
```

Токенізація корпусу

```
tokenized_corpus = [word_tokenize(doc) for doc in corpus]
```

3)Представити корпус як модель Word2Vec. Знайти подібні слова до слів shrimp, economy.

```
from gensim.models import Word2Vec

word2vec_model = Word2Vec(tokenized_corpus, vector_size=100,
window=30, min_count=1, workers=4)
```

Знайти подібні слова до слів shrimp, economy.

```
similar_shrimp = word2vec_model.wv.most_similar('shrimp')
similar_economy = word2vec_model.wv.most_similar('economy')

print("Words similar to 'shrimp':")
for word, similarity in similar_shrimp:
 print(f"{word}: {similarity}")
```

```
Words similar to 'shrimp':
september: 0.16715987026691437
independent: 0.16235072910785675
cinema: 0.1385204941034317
new: 0.131272092461586
stagnated: 0.11637789011001587
feature: 0.10614291578531265
large: 0.09760137647390366
career: 0.09716439992189407
juice: 0.09657728672027588
twelve: 0.08336859196424484
```

```
print("Words similar to 'economy':")
for word, similarity in similar_economy:
 print(f"{word}: {similarity}")
```

```
Words similar to 'economy':
juice: 0.1889600157737732
cumin: 0.18876023590564728
competitions: 0.18367287516593933
evaporated: 0.16077545285224915
add: 0.15947076678276062
three: 0.13724888861179352
salt: 0.12811677157878876
japan: 0.1232738345861435
throughout: 0.11799672991037369
actor: 0.09485597163438797
```