Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Лабораторна робота №3

Аналіз текстів з використанням мови Python

Тема: Моделі текстових даних

Варіант: 1

Виконав

студент групи ІП-11:

Панченко С. В.

Перевірив:

Тимофєєва Ю. С

Київ 2023

ЗМІСТ

1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Ознайомитись з основними текстовими моделями та їх створення за допомогою scikit-learn та genism.

2 ЗАВДАННЯ

Варіант 1.

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 TD-IDF. Спробувати кластеризувати документи за допомогою ієрархічної агломераційної кластеризації.
3. Представити корпус як модель Word2Vec. Знайти подібні слова до слів shrimp, economy.

3 ВИКОНАННЯ

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

Зчитаємо текст з файлу та імпортуємо модулі nltk, pandas, numpy, re.

```
In [113…
import nltk
import pandas as pd
import numpy as np
import re
with open('data/doc1.txt', 'r') as file:
    corpus = file.readlines()
corpus
```

```
Out[113]: ['Whisk the lime juice, oil, chipotle powder, salt, and cumin together in a large bowl. A
dd the shrimp and toss to combine.\n',
 'Growth in Japan evaporated in the three months to September, sparking renewed concern a
bout an economy not long out of a decade-long trough.\n',
 'The independent film festival will feature two new international cinema competition
s.\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 B
erlin Film Festival.']
```

Рисунок 3.1 - Зчитування файлу

Задамо теми кожного документу.

```
In [114…
labels = ['food', 'economics', 'films', 'food',
          'films', 'economics', 'films']
corpus_df = pd.DataFrame({'Document': corpus, 'Category': labels})
corpus_df
```

Out[114]:

	Document	Category
0	Whisk the lime juice, oil, chipotle powder, sa...	food
1	Growth in Japan evaporated in the three months...	economics
2	The independent film festival will feature two...	films
3	Serve the shrimp with the tortillas and salsa.\n	food
4	Twelve films competing in the new world cinema...	films
5	The economy had stagnated throughout the 1990s.\n	economics
6	Actor Daniel Day-Lewis is to be presented with...	films

Рисунок 3.2 - Теми документів

Визначимо стоп-слова англійської мови.

```
In [115…
wpt = nltk.WordPunctTokenizer()
stop_words = nltk.corpus.stopwords.words('english')
```

Рисунок 3.3 - Стоп-слова

Визначимо функцію, що виконує попередню обробку документу. Застосуємо декоратор np.vectorize для того, щоб функція могла працювати з корпусами.

```
In [116…
@np.vectorize
def preproc_doc(doc):
    doc = re.sub(r'[^a-zA-Z\s]', '', doc, re.I | re.A)
    doc = doc.lower()
    doc = doc.strip()
    tokens = wpt.tokenize(doc)
    filtered_tokens = [token for token in tokens if token not in stop_words]
    doc = ' '.join(filtered_tokens)
    return doc

p_corpus = preproc_doc(corpus)
```

Рисунок 3.4 - Обробка документів

Представимо корпус як модуль "Сумка слів". Використаємо для цього клас CountVectorizer зі sklearn.feature_extraction.text.

```
In [117…
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(min_df=0., max_df=1.)
cv_matrix = cv.fit_transform(p_corpus)
cv_matrix = pd.DataFrame(cv_matrix.toarray(),
                         columns=cv.get_feature_names_out())
cv_matrix
```

Out[117]:

	actor	add	award	berlin	bowl	career	category	chipotle	cinema	combine	...	three	throughout	together
0	0	1	0	0	1	0	0	1	0	1	...	0	0	1
1	0	0	0	0	0	0	0	0	0	0	...	1	0	0
2	0	0	0	0	0	0	0	0	1	0	...	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0
4	0	0	0	0	0	0	1	0	1	0	...	0	0	0
5	0	0	0	0	0	0	0	0	0	0	...	0	1	0
6	1	0	1	1	0	1	0	0	0	0	...	0	0	0

7 rows × 55 columns

Рисунок 3.5 - Сумка слів

Виведемо вектор слова "film".

```
In [118…  cv_matrix['film'].values

Out[118]: array([0, 0, 1, 0, 0, 0, 2])
```

Рисунок 3.6 - Вектор слова "film"

3.2 Представити корпус як модель TD-IDF. Спробувати кластеризувати документи за допомогою ієрархічної агломераційної кластеризації.

Перетворимо матрицю з частотою термінів на матрицю tfidf.

```
In [119…  from sklearn.feature_extraction.text import TfidfTransformer
          tt = TfidfTransformer(norm='l2', use_idf=True)
          tt_matrix = tt.fit_transform(cv_matrix)
          tt_matrix = tt_matrix.toarray()
          vocab = cv.get_feature_names_out()
          tv_matrix = pd.DataFrame(np.round(tt_matrix, 2), columns=vocab)
          tv_matrix
```

Out[119]:

	actor	add	award	berlin	bowl	career	category	chipotle	cinema	combine	...	three	throughout	together
0	0.00	0.26	0.00	0.00	0.26	0.00	0.00	0.26	0.00	0.26	...	0.00	0.00	0.26
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.28	0.00	0.00
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	...	0.00	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00	0.37	0.00	0.31	0.00	...	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.61	0.00
6	0.31	0.00	0.31	0.31	0.00	0.31	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00

7 rows × 55 columns

Рисунок 3.7 - Матриця TF-IDF

Обрахуємо подібність векторів за допомогою косинусної відстані.

```
In [120…  from sklearn.metrics.pairwise import cosine_similarity
          similarity_matrix = cosine_similarity(tv_matrix)
          similarity_matrix = pd.DataFrame(similarity_matrix)
          similarity_matrix
```

Out[120]:

	0	1	2	3	4	5	6
0	1.000000	0.000000	0.000000	0.095032	0.000000	0.000000	0.000000
1	0.000000	1.000000	0.000000	0.000000	0.000000	0.117419	0.000000
2	0.000000	0.000000	1.000000	0.000000	0.184013	0.000000	0.230036
3	0.095032	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000
4	0.000000	0.000000	0.184013	0.000000	1.000000	0.000000	0.000000
5	0.000000	0.117419	0.000000	0.000000	0.000000	1.000000	0.000000
6	0.000000	0.000000	0.230036	0.000000	0.000000	0.000000	1.000000

Рисунок 3.8 - Матриця подібності

На основі матриці подібності побудуємо матрицю зв'язку.

```
In [121…
from scipy.cluster.hierarchy import dendrogram, linkage
columns = ['Document\Cluster 1', 'Document\Cluster 2',
           'Distance', 'Cluster Size']
links = pd.DataFrame(linkage(similarity_matrix, 'ward'),
                     columns=columns)
links
```

Out[121]:

	Document\Cluster 1	Document\Cluster 2	Distance	Cluster Size
0	2.0	6.0	1.104333	2.0
1	1.0	5.0	1.248158	2.0
2	0.0	3.0	1.279817	2.0
3	4.0	7.0	1.360619	3.0
4	8.0	9.0	1.564520	4.0
5	10.0	11.0	1.709146	7.0

Рисунок 3.9 - Матриця зв'язку

Візуалізуємо зв'язки за допомогою дендрограми.

```
In [122…
import matplotlib.pyplot as plt
plt.figure(figsize=(8, 3))
plt.title('Дендрограма')
plt.xlabel('Документи')
plt.ylabel('Відстань')
d = dendrogram(links, )
```

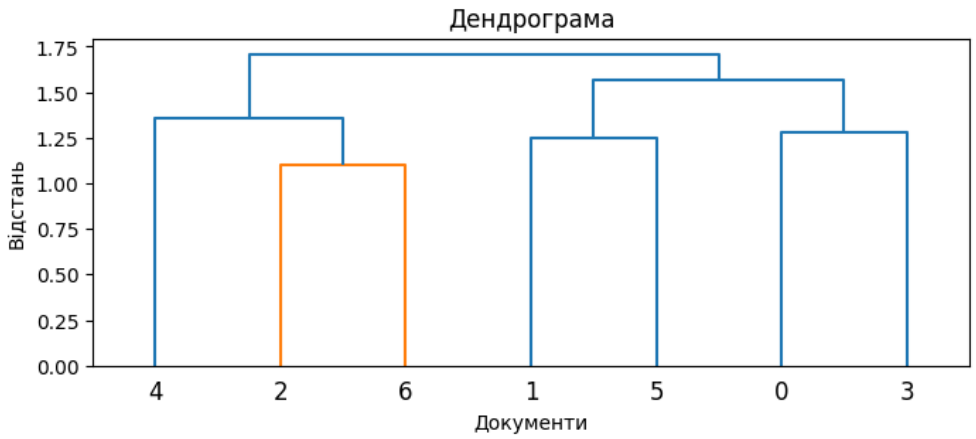

Рисунок 3.10 - Дендрограма

Отримаємо мітки кластерів. Бачимо, що кластери зливаються приблизно на

відстані 1.5.

```
In [123…
from scipy.cluster.hierarchy import fcluster
max_dist = 1.5
cluster_labels = fcluster(links, max_dist,
                          criterion='distance')
cluster_labels = pd.DataFrame(cluster_labels,
columns=['ClusterLabel'])
corpus_df = pd.concat([corpus_df, cluster_labels], axis=1)
corpus_df
```

Out[123]:

	Document	Category	ClusterLabel
0	Whisk the lime juice, oil, chipotle powder, sa...	food	3
1	Growth in Japan evaporated in the three months...	economics	2
2	The independent film festival will feature two...	films	1
3	Serve the shrimp with the tortillas and salsa.\n	food	3
4	Twelve films competing in the new world cinema...	films	1
5	The economy had stagnated throughout the 1990s.\n	economics	2
6	Actor Daniel Day-Lewis is to be presented with...	films	1

Рисунок 3.11 - Додавання міток

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

Побудуємо модель Word2Vec.

```
In [124…
from gensim.models import word2vec
tokenized_corpus = [wpt.tokenize(document) for document in p_corpus]
feature_size = 100
window_context = 30
min_word_count = 1
sample = 1e-3
w2v_model = word2vec.Word2Vec(tokenized_corpus,
                              vector_size=feature_size,
                              window=window_context,
                              min_count=min_word_count,
                              sample=sample)
```

Рисунок 3.12 - Модель Word2Vec

Подивимося подібні слова до "shrimp", "economy".

```
In [125…
similar_words = {
  search_term:
    [item[0] for item in w2v_model.wv.most_similar([search_term],topn=5)]
      for search_term in ['shrimp', 'economy']}
similar_words
```

```
Out[125]: {'shrimp': ['evaporated', 'juice', 'stagnated', 'long', 'new'],
           'economy': ['months', 'long', 'shrimp', 'cinema', 'world']}
```

Рисунок 3.13 - Виведення подібних слів

4 ВИСНОВОК

Під час виконання даної лабораторної роботи я ознайомився з основними текстовими моделями та їхнім створенням за допомогою scikit-learn та genism.

У першому завданні було створено модель “Сумка слів” та виведено вектор слова "film”.

У другому завданні побудували модель TD-IDF, визначено подібність слів за допомогою косинусної відстані, проведено кластеризацію.

У третьому завданні побудовано модель Word2Vec та виведено найбільш подібні слова до “shrimp” та “economy”.