**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**

**Звіт**

з лабораторної роботи №3 з дисципліни

«Обробка та аналіз текстових даних на Python»

„**Моделі текстових даних**”

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-11 Головня Олександр Ростиславович*

**Перевірив**

(прізвище, ім'я, по батькові)

Київ 2024

**Мета роботи:** Ознайомитись з основними текстовими моделями та їх

створення за допомогою бібліотек scikit-learn та genism.

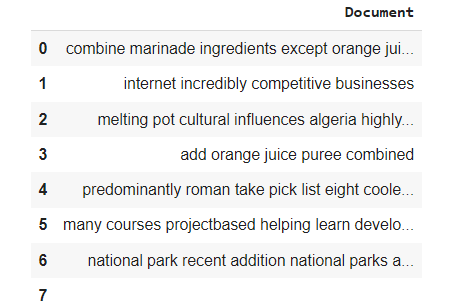
**Завдання до лабораторної роботи:**

Створити програму, яка виконує завдання відповідно до варіанту.

Зчитати файл doc2. Вважати кожен рядок окремим документом корпусу. Виконати попередню обробку корпусу.  
1)Представити корпус як модель «Сумка n-грам», взяти біграми. Вивести вектор для orange juice.  
2)Представити корпус як модель TF-IDF. Спробувати кластеризувати документи за допомогою ієрархічної агломераційної кластеризації.  
3)Представити корпус як модель FastText. Знайти подібні слова до слів Algeria, combine.

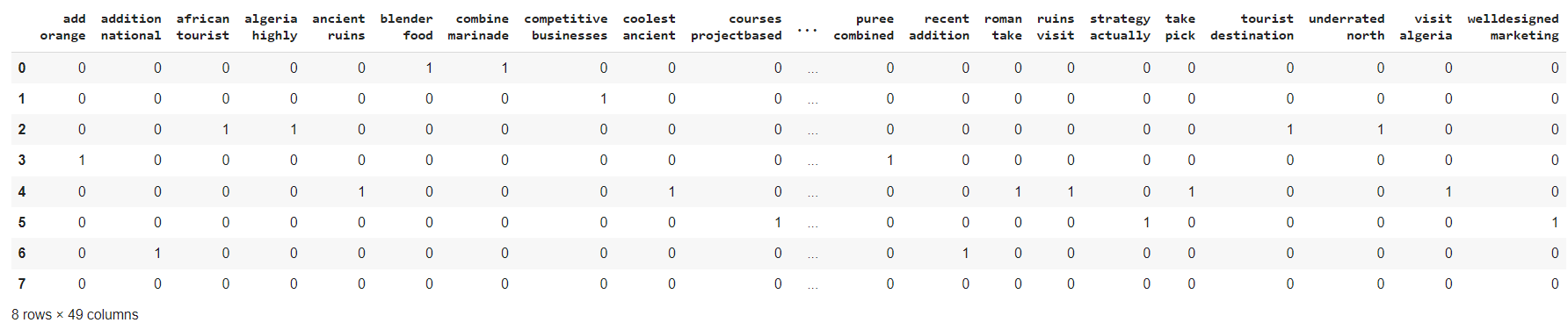
**Результат виконання коду:**

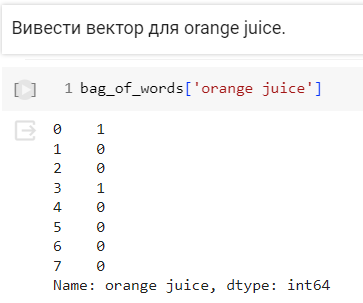
Виконана попередня обробка корпусу.



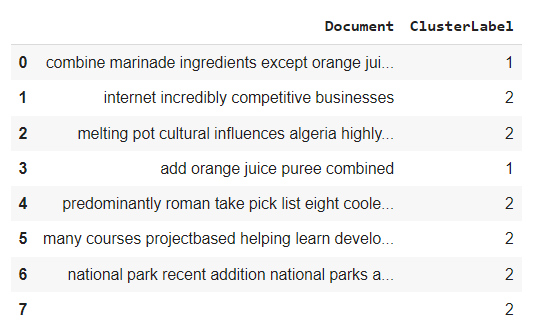
Корпус представлений як модель «Сумка n-грам», взято біграми.



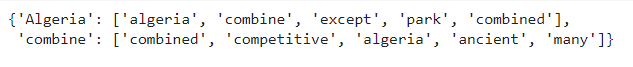




Кластеризація документів за допомогою ієрархічної агломераційної кластеризації.



Подібні слова до слів Algeria, combine.



**Код програми:**

with open('doc2.txt') as file:

    corpus = [line.rstrip() for line in file]

print(corpus)

from nltk.tokenize import WordPunctTokenizer

from nltk.corpus import stopwords

import re

import nltk

nltk.download('stopwords')

stop\_words = stopwords.words('english')

tokenizer = WordPunctTokenizer()

def preprocess\_document(doc):

    doc = re.sub(r'[^a-zA-Z\s]', '', doc, re.I | re.A)

    doc = doc.lower()

    doc = doc.strip()

    tokens = tokenizer.tokenize(doc)

    filtered\_tokens = [token for token in tokens if token not in stop\_words]

    doc = ' '.join(filtered\_tokens)

    return doc

import numpy as np

import pandas as pd

preprocess\_corpus = np.vectorize(preprocess\_document)

preprocessed\_corpus = preprocess\_corpus(corpus)

corpus\_df = pd.DataFrame(preprocessed\_corpus, columns=['Document'])

corpus\_df

from sklearn.feature\_extraction.text import CountVectorizer

count\_vectorizer = CountVectorizer(ngram\_range=(2, 2))

bv\_matrix = count\_vectorizer.fit\_transform(preprocessed\_corpus)

print(bv\_matrix)

import pandas as pd

vocab = count\_vectorizer.get\_feature\_names\_out()

bag\_of\_words = pd.DataFrame(bv\_matrix.toarray(), columns=vocab)

bag\_of\_words

bag\_of\_words['orange juice']

from sklearn.feature\_extraction.text import TfidfVectorizer

tfidf\_vectorizer = TfidfVectorizer()

tv\_matrix = tfidf\_vectorizer.fit\_transform(preprocessed\_corpus)

from sklearn.metrics.pairwise import cosine\_similarity

similarity\_matrix = cosine\_similarity(tv\_matrix)

similarity\_matrix

from scipy.cluster.hierarchy import dendrogram, linkage

import matplotlib.pyplot as plt

links = linkage(similarity\_matrix, 'ward')

plt.figure(figsize=(8, 6))

plt.xlabel('Documents')

plt.ylabel('Distance')

dendrogram(links)

plt.show()

from scipy.cluster.hierarchy import fcluster

cluster\_labels = fcluster(links, max\_dist, criterion='distance')

cluster\_labels = pd.DataFrame(cluster\_labels, columns=[' ClusterLabel'])

pd.concat([corpus\_df, cluster\_labels], axis=1)

from gensim.models.fasttext import FastText

tokenized\_corpus = [doc.split() for doc in preprocessed\_corpus]

ft\_model = FastText(tokenized\_corpus, vector\_size=300, min\_count=1, sg=1)

similar\_words = {search\_term: [item[0] for item in ft\_model.wv.most\_similar([search\_term], topn=5)] for search\_term in ['Algeria', 'combine']}

similar\_words

**Висновок:** Під час розробки лабораторної роботи, я ознайомився на практиці з моделями «Сумка n-грам»(взяв біграм), модель TF-IDF(кластеризував документи за допомогою ієрхічної алгомераційної кластеризації), модель FastText(шукав подібні слова)