Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Катедра інформатики та програмної інженерії

Звіт

з комп’ютерного практикуму № 8 з дисципліни

«Аналіз даних в інформаційних системах»

«Аналіз текстів»

Виконав студент ІП-11 Лесів Владислав Ігорович

Перевірив Олійник Ю. О.

Київ 2023

**Комп’ютерний практикум 8**

**Аналіз текстів**

**Мета роботи:** ознайомитись з методами аналізу текстів.

**Основне завдання**.

Дані для виконання: текстові дані у форматі csv-файлів або дані з відкритих джерел.

1. Нормалізація та попередня обробка даних.
   1. провести очищення текстових даних від стоп-слів/тегів/розмітки;
   2. виконати токенізацію текстових елементів;
   3. провести лематизацію текстових елементів. Зберегти результат в окремий файл;
   4. створити Bag of Words для всіх нормалізованих слів. Зберегти результат в окремий файл;
   5. порахувати метрику TF-IDF для 10 слів, що найчастіше зустрічаються в корпусі;

**Хід роботи.**

**Основне завдання.**

Для аналізу я обрав текст про Київ з одного з відкритих джерел.

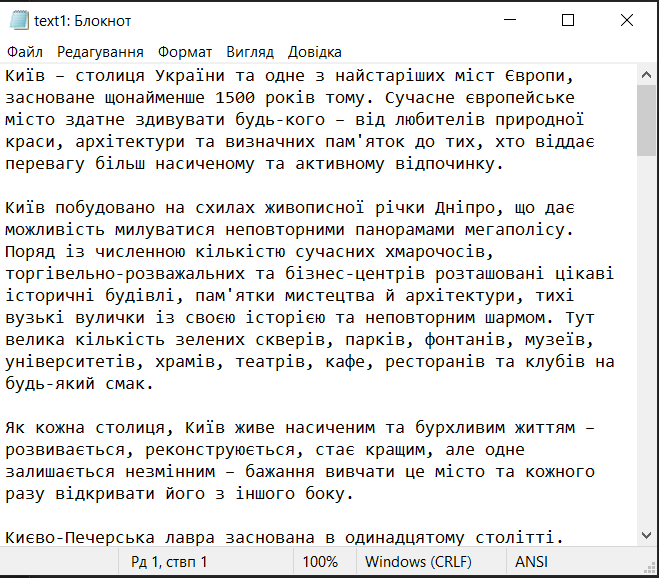


Рисунок 1 – Приклад тексту

Для початку читаю файл, з допомогою модуля spacy завантажую мовну модель, а за модулем nltk токенізую текст за реченнями.

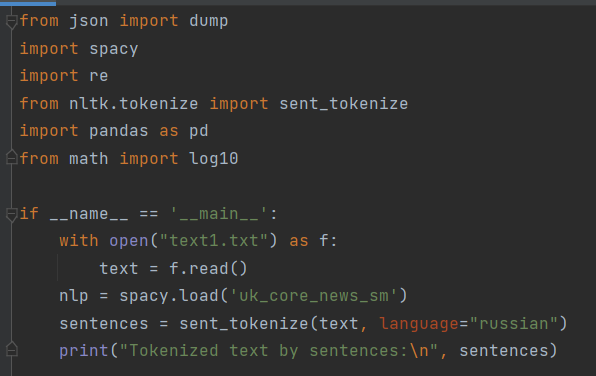


Рисунок 2 – Код до завантаження моделі та токенізації за реченнями

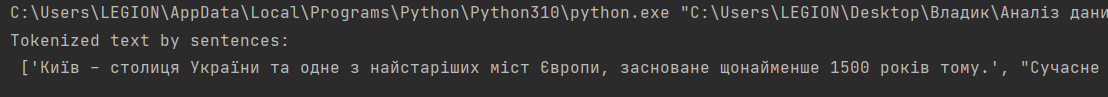


Рисунок 3 – Результат токенізації за реченнями

Далі здійснимо лематизацію та очистимо текст від стоп-слів та пунктуації. Для цього я створюю функцію lemmatize().

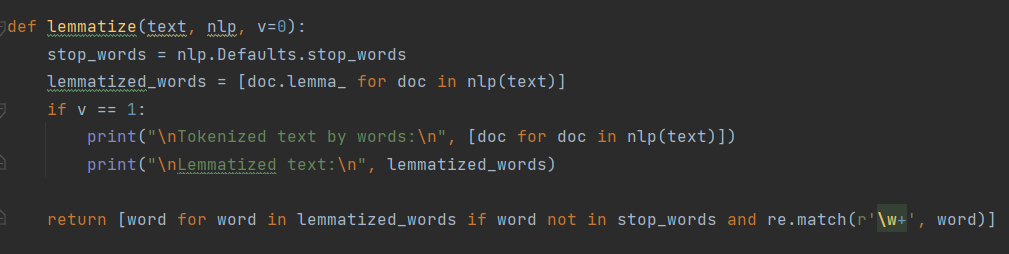
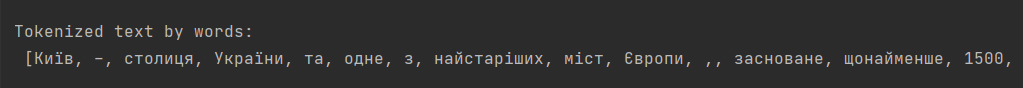


Рисунок 4 – Код для лематизації та очищення тексту

Завантажуємо стоп-слова з мовної моделі, далі з її ж допомогою виконуємо лематизацію. Наприкінці перевіряємо, чи лематизовані елементи не є стоп-словами і чи власне містять лише букви, тобто відсіюємо пунктуацію.



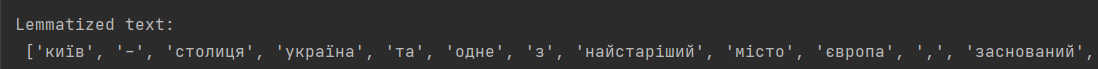


Рисунок 5 – Результат токенізації й лематизації

Записуємо результат готового очищеного тексту у форматі JSON за допомогою функції write. Перевіряємо результат.

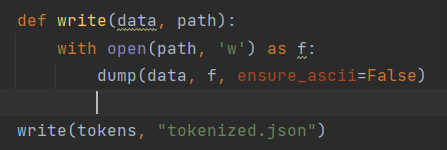


Рисунок 6 – Функція запису у файл

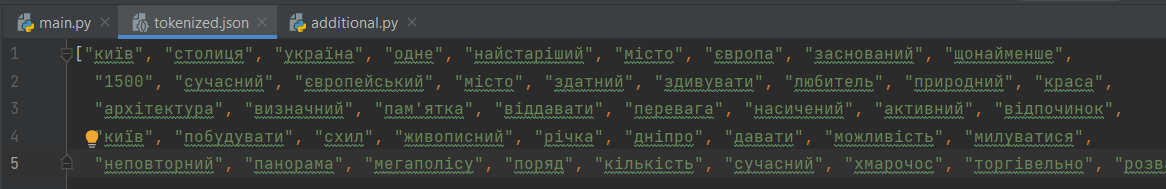


Рисунок 7 – Очищений текст, записаний у JSON

Бачимо, що текст нормалізований, як і потрібно.

Далі створюю Bag Of Words. Для цього створюю функцію BOW. Я використовую цикл і словник, тож слова пораховані в правильному порядку. Записую дані так само в JSON.

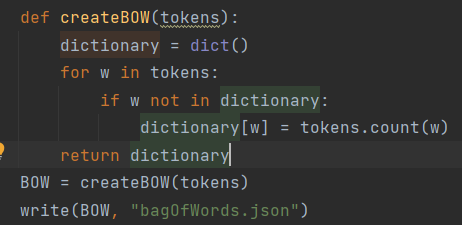


Рисунок 8 – Створення Bag of Words

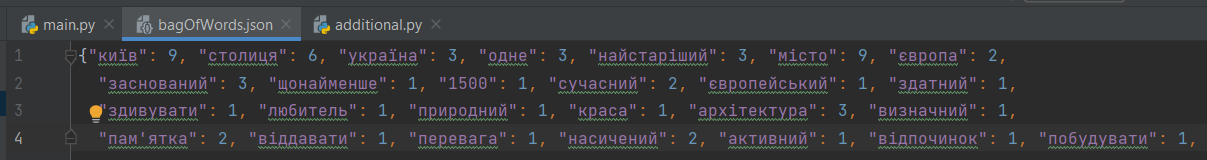


Рисунок 9 – Bag of Words, записаний у файл

Шукаю 10 найчастіше вживаних слів.

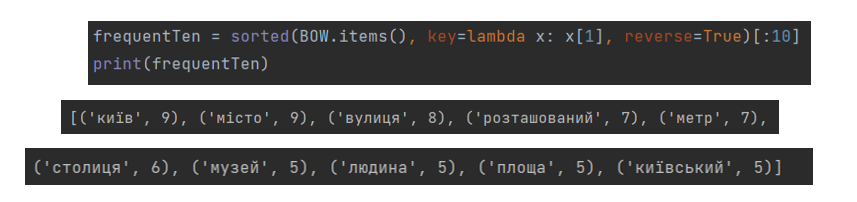


Рисунок 10 – Результат пошуку 10 найчастіших слів

Рахую TF. Для цього нормалізую текст окремо за реченнями, і відповідно для кожного речення для кожного слова рахую TF за формулою.

Далі рахую IDF так само за формулою. Перемножую TF з IDF й отримуємо TF-IDF, результат записую у файл.

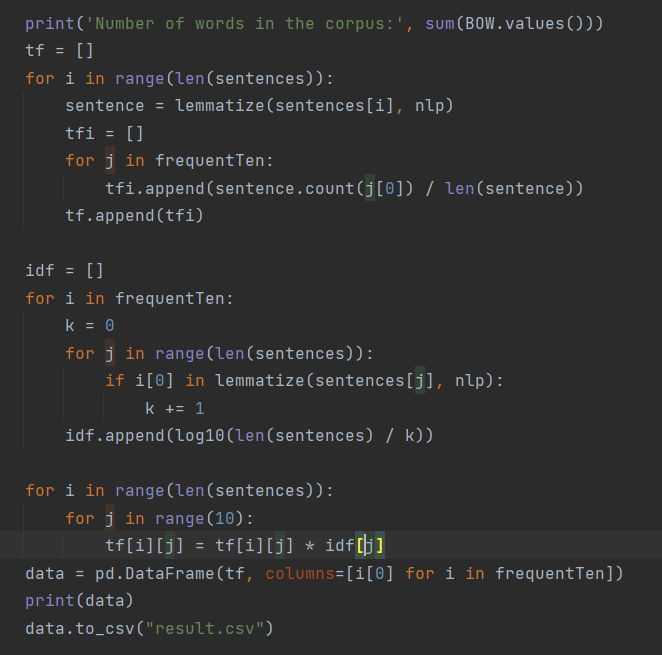


Рисунок 11 – Код знаходження TF-IDF

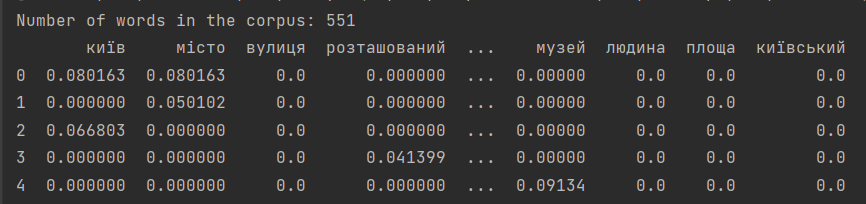


Рисунок 12 – Результат обчислення

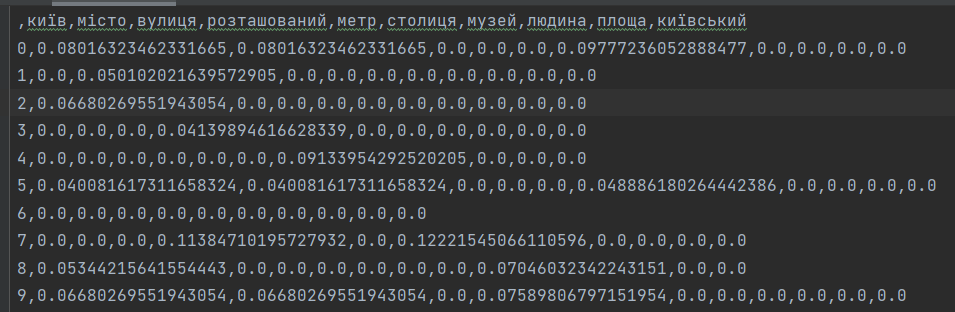


Рисунок 13 – Файл з усіма обчисленими TF-IDF

**Висновок.**

Отже, у цій роботі я ознайомився з методами аналізу текстів.

У результаті лабораторної роботи було проведено очистку, токенізацію й лематизацію тексту, створено Bag of Words, а також пораховано значення TF-IDF для 10 найвживаніших слів. Відповідно побачили, які слова між найвживаніших є важливішими у тому чи іншому реченні.

Використовуючи PyCharm та модулі spacy та nltk, отримуємо коректний результат.