Лабораторна робота № 5

Тема: Бібліотека NLTK мови Python**.** Токенізація текстів.

Мета: Відпрацювати практичні навички написання програм на мові Python для обробки текстової інформації використовуючи методи бібліотеки NLTK.

Література: https://coderlessons.com/tutorials/mashinnoe-obuchenie/uchebnik-nltk/uchebnik-nltk
https://ciksiti.com/uk/chapters/2193-nltk-tutorial-in-python--linux-hint на українській мові

Зміст роботи:

Завдання 1.

Дано текстовий рядок (не менше 5 речень англійською мовою). Виконати токенізацію за реченнями та словами. Зі списку слів вилучити розділові знаки та стоп слова. Визначити автоматично частини мови слів та проаналізувати їх на правильність. Результати кожної дії вивести на екран.

Завдання 2.

Використовуючи отриману інформацію проведіть аналіз за параметрами:

- Підрахувати кількість слів у кожному реченні.
- Визначити кількість стоп слів у вашому тексті.
- Визначити найдовше слово.
- Визначити скільки слів у тексті містять 4 символи.

Завдання 3.

Виконати операції стемінгу та лематизації використовуючи результат завдання 1, як вхідні дані. Порівняти результати стемінгу та лематизації, висновки записати у звіт.

Завдання 4.

Підготувати текстовий файл з текстом до 50 слів українською мовою.

За бажанням створити текстовий файл (на основі файлу стоп_слова_українська_мова.docx) стоп-слів відповідно до обраного тексту.

Провести токенізацію тексту за словами, вилучити розділові знаки.

Вилучити стоп-слова, використовуючи створений список стоп-слів або файл стоп_слова_українська_мова.docx .

Отримати статистичну інформацію обраного тексту а саме: об’єм вхідного тексту, кількість розділових знаків, кількість стоп-слів, кількість слів вихідного тексту.

Провести операції стемінгу та лематезації. Оцінити результат. Висновки записати у звіт.

!!!!!!! Отримані результати завдань вивести у файл та на екран.

Методичні рекомендації.

text = "Hello there! Welcome to this tutorial on tokenizing. After going through this tutorial you will be able to tokenize your text. Tokenizing is an important concept under NLP. Happy learning!"

Для токенізації відповідно до речень використовуйте:

print(sent_tokenize(text))

Результат, який ми отримуємо, такий:

['Hello there!', 'Welcome to this tutorial on tokenizing.', 'After going through this tutorial you will be able to tokenize your text.', 'Tokenizing is important concept under NLP.', 'Happy learning!']

!!!!!Він повертає список з кожним елементом у вигляді пропозиції з тексту.

Для токенізації відповідно до слів, які ми використовуємо:

print(word_tokenize(text))

Результат, який ми отримуємо, такий:

['Hello', 'there', '!', 'Welcome', 'to', 'this', 'tutorial', 'on', 'tokenizing', '.', 'After', 'going', ' through', 'this', 'tutorial', 'you', 'will', 'be', 'able', 'to', 'tokenize', 'your', 'text', '.', 'Tokenizing' , 'is', 'an', 'important', 'conceot', 'under', 'NLP', '.', 'Happy', 'learning', '!']

!!!!!Він повертає список з кожним елементом у вигляді слова з тексту. Тепер вони можуть використовуватися як токени в мовній моделі для навчання.

Повний код:

```
import nltk
```

```
from nltk.tokenize import sent_tokenize, word_tokenize

text = "Hello there! Welcome to this tutorial on tokenizing. After going through this tutorial you will be able to tokenize your text. Tokenizing is an important concept under NLP. Happy learning!"

print(sent_tokenize(text))

print(word_tokenize(text))
```

Стемінг та леметизація

Спочатку ми імпортуємо PorterStemmernltk.stem.porter Далі ми ініціалізуємо stemmer для змінної stemmer.stem().

```
from nltk.stem.porter import PorterStemmer

stemmer = PorterStemmer()

print(stemmer.stem("going"))
```

Лематизація нормалізує слово на основі контексту та словникового запасу тексту. У NLTK ви можете лематизувати пропозиції за допомогою класу WordNetLemmatizer

По-перше, потрібно завантажити ресурс wordnet

```
nltk.download('wordnet')
```

Як тільки він завантажений, потрібно імпортувати клас WordNetLemmatizer

```
from nltk.stem.wordnet import WordNetLemmatizer

lem = WordNetLemmatizer()
```

Щоб використовувати лематизатор, використовуйте метод .lemmatize() Потрібно два аргументи – слово та контекст. Наприклад «v» для контексту.

Робота з файлами

У наступному прикладі відбувається читання вмісту всього файлу, починаючи з поточної позиції курсора (переміщує курсор в кінець файлу):

```
with open('example_text.txt', 'r') as file:
    contents = file.read() print(contents)
```

Наступний приклад демонструє роботу з курсором:

```
with open('example_text.txt', 'r') as file:
    contents = file.read(10) # вказуємо кількість символів для читання # курсор переміщається на 11 символ
rest = file.read() # читаємо з 11 символу
```

Якщо необхідно отримати список, що складається з рядків, то можна скористатися методом readlines:

```
with open('example_text.txt', 'r') as file:
```

lines = file.readlines() print(lines)

Використовуйте наступний спосіб читання з файлу, якщо хочете зробити деякі операції з кожним з рядків, починаючи з поточної позиції файлового курсора до кінця файлу:

```
with open('plan.txt', 'r') as file:
   for line in file:
    print(line)
print(len(line.strip()))
```

Контрольні запитання.

1. Поясніть призначення бібліотеки NLTK.
2. Назвіть методи токенізації текстів за словами, за реченнями?
3. В чому полягає призначення токенізації?
4. В якому вигляді подається результат токенізації?
5. В чому полягає суть стоп-слів?
6. Для чого прибирати стоп-слова з тексту?
7. Як автоматично визначити частини мови слів?
8. Як отримати доступ до методів NLTK ?