НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»
Факультет Інформатики та обчислювальної техніки
Кафедра Інформаційних систем та технологій

ЛАБОРАТОРНА РОБОТА №4

з дисципліни «Обробка та аналіз текстових даних на Python»

на тему «Векторизація тексту та застосування TF-IDF»

Варіант №2

Виконала: студентка групи ІС-з21
Коломієць Катерина Миколаївна
24.05.2025

Перевірив: асист. Мягкий М. Ю.

Київ – 2025

GitHub: https://github.com/kate-miiets-uni/oatd-py

Теоретичні відомості

Технології NLP — Natural Language Processing, обробка природної мови — дозволяють комп'ютерам опрацьовувати людську мову, розуміти її значення та контекст, а також пов'язані з нею емоційне забарвлення й наміри. Надалі ці дані можуть використовуватися для створення чогось нового.

Векторизація — це термін, що позначає класичний підхід до перетворення вхідних даних із їхнього початкового формату (наприклад, тексту) у вектори дійсних чисел, які зрозумілі моделям машинного навчання.

TF-IDF (Term Frequency-Inverse Document Frequency) — це числовий статистичний показник, який відображає важливість слова для документа. І хоча ця методика також заснована на частотності, як і «мішок слів», вона використовує більш складні обчислення.

Хід роботи

1. Імпортуємо необхідні бібліотеки. Тут cloudsraper - це бібліотека для обходу захисту на веб-сайтах, які використовують Cloudflare або інші механізми блокування боті;pandas – для зручного зберігання, обробки та аналізу даних у форматі таблиці; sklearn.feature_extraction.text.TfidfVectorizer - це компонент бібліотеки Scikit-learn, який використовується для аналізу текстів.

```
import cloudscraper
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
```

2. Далі надсилаємо запит на сервер про отримання тексту новини. У цей раз стикнулися з проблемою блокування сервером запиту, тому використовуємо cloudscraper.create_scraper(), який створює скрапер, який обходить захист від ботів на веб-сайтах, зокрема ті, що використовують Cloudflare. cloudscraper надсилає HTTP-запити так, щоб сайт сприймав його як реальний браузер. По закінченню обходу захисту, за допомогою scraper.get(url), повертає HTML-код сторінки.

```
url = 'https://nv.ua/ukr/world/countries/mask-pokidaye-politiku-ch
# Надсилаємо запит на отримання тексту за посиланням
scraper = cloudscraper.create_scraper()
response = scraper.get(url)
```

3. Робимо перевірку за статус відповіді, якщо status_code == 200, все добре та виконуємо кроки до виконання задачі.

```
if response.status_code == 200:
    text = response.text
```

4. Векторизуємо текст. Для цього спочатку за допомогою TfidVectorizer() перетворюємо текст у матрицю TF-IDF, при цьому виключаючи стоп-слова. Далі за допомогою .fit_transform() аналізуємо текст та створюємо їхнє числове представлення, на цьому моменті кожне слово отримує вагове значення, яке залежить від частоти його появи у тексті та його унікальності. Вихідним результатом буде матриця (набір чисел), де кожен елемент - це TF-IDF значення конкретного слова у тексті.

```
# Векторизація тексту
vectorizer = TfidfVectorizer(stop_words=["та", "і", "й", "у", "в", "на", "це", "що", "не", "а", "щоб", "про"])
tfidf_matrix = vectorizer.fit_transform([text])
```

5. Далі перетворюємо розріджену матрицю, яка є результатом минулого кроку, у таблицю DataFrame. Це необхідно зробити, адже перегляд даних та модифікації набагато легше виконувати з таблицею.

```
# Створення DataFrame для зберігання матриці у вигляді таблиці
df = pd.DataFrame(tfidf_matrix[0].T.todense(),
                  index=vectorizer.get_feature_names_out(), columns=["TF-IDF"])
```

6. Далі сортуємо слова за спаданням їхньої ваги та виводимо на консоль 10 найвагоміших слів.

```
# Сортування слів за спаданням
df = df.sort_values('TF-IDF', ascending=False)

# Виведення 10-ти найвагоміших слів
print(df[:10])
```

Результат:

```
         TF-IDF
div    0.515842
class  0.383612
ua     0.258647
nv     0.252835
https  0.242664
data   0.185994
href   0.177275
ukr    0.129324
style  0.127871
icon   0.122058
```

Висновок

Отже, у цій лабораторній роботі я дізналася про метод обробки та аналізу тексту TF-IDF, ознайомилася з принципом роботи методу. Також виконала векторизацію тексту,

аналізування ваги кожного слова тексту, попрацювала трохи з бібліотекою pandas та таблицею DataFrame. Вперше стикнулася з проблемою блокування сервером запиту про отримання тексту, завдяки чому познайомилася з бібліотекою cloudsraper та її функцією cloudscraper.create_scraper(), який допомагає оминути захист сайтів від ботів.

Контрольні запитання

1. Що таке TF-IDF?

TF-IDF — статистичний показник, що використовується для оцінки важливості слів у контексті документа, що є частиною колекції документів чи корпусу. Вага слова пропорційна кількості вживань цього слова у документі, і обернено пропорційна частоті вживання слова у інших документах колекції.

2. Яка різниця між Bag-of-Words і TF-IDF?

BoW - це просте представлення тексту, яке враховує лише наявність (або частоту) слів у документі, ігноруючи їх порядок та структуру. Створюється словник з усіх унікальних слів, документ представляється як вектор, де кожен елемент вектора відповідає слову зі словника. Значення елемента вектора показує, чи присутнє слово в документі або скільки разів слово зустрічається в документі. Основна ідея – аналізує наявну множину слів та частоту їх появи.

TF-IDF - це метод, який не тільки враховує частоту слів у документі, але й зважує їхню важливість залежно від того, як часто вони зустрічаються у всьому наборі документів. Спочатку обчислюється Term Frequency (TF) - частота появи слова в документі, а потім обчислюється Inverse Document Frequency (IDF) – частота появи слова в інших документах. Значення IT-IDF визначається добутком TF та IDF. Слова, які часто зустрічаються в конкретному документі, але рідко зустрічаються в інших документах, вважаються більш важливими для цього документа.

3. Як працює TfidfVectorizer?

Спочатку текст кожного документа розбивається на послідовність токенів (зазвичай слів). За замовчуванням TfidfVectorizer використовує простий токенізатор, який розділяє текст за пробілами та знаками пунктуації. Після токенізації TfidfVectorizer будує словник з усіх унікальних токенів, знайдених у всіх документах набору. Для кожного документа та кожного слова зі словника обчислюється його TF. Далі для кожного слова у словнику обчислюється IDF. Після цього обчислюється TF-IDF для кожного слова. На виході TfidfVectorizer надає розріджену матрицю, де кожен рядок відповідає документу, а кожен стовпець відповідає слову зі словника. Кожна комірка матриці містить значення TF-IDF відповідного слова в цьому документі.

4. Як визначається вага слова у TF-IDF?

Вага слова визначається добутком TF та IDF. Чим частіше слово зустрічається в документі, тим вища його TF. Зазвичай TF нормалізується за довжиною документа, щоб уникнути переваги довших документів. Слова, які зустрічаються у багатьох документах, мають низький IDF, оскільки вони вважаються менш інформативними для розрізнення документів. Слова, які зустрічаються в невеликій кількості документів, мають високий IDF, оскільки вони, ймовірно, є більш специфічними для цих документів. IDF обчислюється як логарифм відношення загальної кількості документів до кількості документів, що містять дане слово. Вага слова є високою, якщо слово часто зустрічається в даному документі і рідко зустрічається в інших документах корпусу. І навпаки, вага є низькою, якщо слово рідко зустрічається в даному документі або часто зустрічається у багатьох документах.

5. Чому TF-IDF краще підходить для оцінки важливості слів?

Цей метод забезпечує якісну оцінку частоти появи слова в межах одного документа порівняно з частотою його появи в інших, що дозволяє якісно визначити міру важливості слова в контексті документа.

6. Які недоліки має метод TF-IDF?

Метод ігнорує порядок слів у документі, що є недоліком, адже порядок слів часто впливає на значення тексту. Також метод не відслідковує синоніми, таким чином слова, які мають одне значення, не впливатимуть на вагу одне одного в документі.

7. Як можна нормалізувати текст перед TF-IDF векторизацією?

Для нормалізації тексту можна привести текст до нижнього регістру, щоб уникнути сприймання програмою унікального слова з великим регістром та низьким як різних, видалити знаки пунктуації, видалити стоп-слова, щоб вилучити часто вживані слова, які зазвичай не несуть важливої інформації про зміст документа, стеммінгувати текст, тобто звести слова до їхнього кореня для уніфікації форм слів, лематизувати текст, тобто звести слова до їхньої леми (словникової форми), враховуючи контекст та частину мови.

8. Як знайти слова з найвищими TF-IDF значеннями?

Спочатку необхідно використати TfidVectorizer для обчислення ваги кожного унікального слова, а потім відсортувати від більшого до меншого значення ваги, після чого вивести 10 найбільших.

9. Чому важливо видаляти стоп-слова перед обчисленням TF-IDF?

Видалення стоп-слів перед TF-IDF допомагає зменшити вплив дуже частих, але неінформативних слів. Ці слова мають низький IDF, але можуть мати високий TF, спотворюючи важливість інших слів. Виключення стоп-слів дозволяє TF-IDF зосередитися на змістовних термінах, які краще відображають унікальність документа. Це також може зменшити розмірність векторів.

10. Як впливає розмір корпусу документів на TF-IDF?

Розмір корпусу сильно впливає на IDF: у великому корпусі IDF стає стабільнішим та краще відображає рідкість слів у мові. Це допомагає чіткіше розрізняти важливість слів, зменшуючи вплив випадкових появ. Зі збільшенням корпусу оцінка IDF стає більш узагальненою та надійною. У малому корпусі IDF може бути менш стабільним, а рідкісні слова можуть отримати завищену вагу.