Курс «Базовая обработка данных на языке Python»

автор: Киреев В.С., к.т.н., доцент

Лабораторная работа № 5

Тема: «Обработка текстовых данных с помощью Python. Основы токенизации и векторизации»

Цель работы: изучить методы работы в Python с текстами на естественном языке, с помощью встроенных строковых функций Python, библиотек collections, re, sklearn.

Теоретическая справка

Строковые функции

Строка — это последовательность символов. Python рассматривает все, что находится внутри кавычек, как строку. Это включает буквы, цифры и символы. Python не имеет символьного типа данных, поэтому один символ — это строка длиной 1.

Python предоставляет несколько способов включения переменных в строки. Самый простой и предпочтительный способ форматирования строк — использование f-строк.

```
name = "Alice"
age = 22
print(f"Name: {name}, Age: {age}")
```

upper() и lower() : метод upper() преобразует все символы в верхний регистр. Метод lower() преобразует все символы в нижний регистр.

Python допускает использование отрицательных адресных ссылок для доступа к символам с конца строки, например, -1 относится к последнему символу, -2 относится к предпоследнему символу и т. д.

```
s = "Строка"
print(s[-10])
```

Модуль collections

Контейнеры — это объекты, которые содержат объекты. Они предоставляют способ доступа к содержащимся объектам и итерации по ним. Примерами встроенных контейнеров являются кортежи, списки и словари. Другие включены в модуль Collections . Counter — это подкласс dict. Поэтому это неупорядоченная коллекция, в которой элементы и их соответствующее количество хранятся в виде словаря.

```
from collections import Counter
print(Counter(['B','B','A','B','C','A','B','B','A','C']))
print(Counter({'A':3, 'B':5, 'C':2}))
print(Counter(A=3, B=5, C=2))
```

Визуализация с помощью облака слов

Облако слов — это метод визуализации данных, используемый для представления текстовых данных, в котором размер каждого слова указывает на его частоту или важность. Значимые текстовые точки данных могут быть выделены с помощью облака слов. Облака слов широко используются для анализа данных с веб-сайтов социальных сетей.

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
with open('/content/Collection5/001.txt') as f:
  news=''.join(f.readlines())
wordcloud = WordCloud(width = 800, height = 800,
              background_color ='white',
              min_font_size = 10).generate(news)
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()
```

Регулярные выражения

Регулярное выражение или RegEx — это специальная последовательность символов, которая использует шаблон поиска для поиска строки или набора строк. Он может определять наличие или отсутствие текста, сопоставляя его с определенным шаблоном, а также может разбивать шаблон на один или несколько подшаблонов. В Python есть встроенный модуль под названием «re» , который используется для регулярных выражений в Python.

МетаСимволы	Описание
\	Используется для того, чтобы опустить особое значение символа, следующего за ним.
[]	Представляют список символов
^	Соответствует началу строки
$	Соответствует концу строки

.	Соответствует любому символу, кроме символа новой строки
|	Означает ИЛИ (соответствует любому из символов, разделенных этим символом).
?	Соответствует нулю или одному вхождению
*	Любое количество вхождений (включая 0 вхождений)
+	Один или несколько случаев
{}	Укажите количество вхождений предыдущего регулярного выражения для сопоставления.
()	Заключите группу регулярных выражений

Диапазон обеспечивает гибкость для сопоставления текста с помощью шаблона диапазона, например, диапазона цифр (от 0 до 9), диапазона символов (от A до Z) и т. д. Символ дефиса в классе символов представляет диапазон.

```
import re
print('Range',re.search(r'[a-zA-Z]', 'x'))
```

Регулярный механизм выражений позволяет вам указывать необязательные символы с помощью символа ?. Он позволяет символу или классу символов либо присутствовать один раз, либо не встречаться вообще.

```
import re
print('Color',re.search(r'colou?r', 'color'))
print('Colour',re.search(r'colou?r', 'colour'))
```

Векторизация

Векторизация — это процесс преобразования текстовых данных в числовые векторы. В контексте обработки естественного языка (NLP) векторизация преобразует слова, фразы или целые документы в формат, который может быть понят и обработан моделями машинного обучения. Эти числовые представления фиксируют семантическое значение и контекстные связи текста, позволяя алгоритмам выполнять такие задачи, как классификация, кластеризация и прогнозирование.

Модель Bag of Words представляет текст, преобразуя его в набор слов (или токенов) и их частот, игнорируя грамматику, порядок слов и контекст. Каждый документ представлен как вектор количества слов, причем каждый элемент в векторе соответствует частоте определенного слова в документе.

```
from sklearn.feature_extraction.text import CountVectorizer
documents = ["The cat sat on the mat.", "The dog sat on the log.","Cats and dogs are pets."]
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(documents)
print(X.toarray())
print(vectorizer.get_feature_names_out())
```

Метод главных компонент (Principal Components Analysis, PCA)

Анализ главных компонент — это, по сути, статистическая процедура преобразования набора наблюдений возможно коррелированных переменных в набор значений линейно некоррелированных переменных. Каждый из главных компонентов выбирается таким образом, чтобы он описывал большую часть из них, все еще имеющуюся дисперсию, и все эти главные компоненты ортогональны друг другу. Во всех главных компонентах первый главный компонент имеет максимальную дисперсию.

```
from sklearn.decomposition import PCA
```

```
pca = PCA(n_components=2)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
explained_variance = pca.explained_variance_ratio_
```

PCA можно использовать для визуализации многомерных данных в пространстве с меньшей размерностью, что упрощает их интерпретацию и понимание. Проецируя данные на главные компоненты, можно легче визуализировать закономерности и взаимосвязи между переменными.

Самостоятельное задание

1. Подсчитать число стихов в представленном файле
2. Найти топ-20 по частоте слов, встречающихся в стихах
3. Найти топ-20 по частоте информативных (не предлоги, не мат, т.е. слова описывающий смысл стиха) слов, встречающихся в стихах
4. С помошью регулярных выражений, найти первое слово, встречающееся после имени Олег, во всех стихах.
5. Использовать облако слов, для визуализации п.4.
6. Разделить исходный список на стихи (строка, включающая 4 переноса каретки)
7. Каждый полученный стих из 4. очистить от стоп-слов
8. Векторизовать полученные стихи (каждый стих-строку превратить в вектор, где измерения -все слова, встречавшиеся во всех стихах, а значения в измерениях вектора - это частота этих слов, внутри соответствующего стиха). Таким образом, результат - это матрица стих-слово (документ-терм/токен)
9. С помощью метода главных компонент оставить только два измерения в каждом векторе, и визуализировать стихи с помощью диаграммы рассеяния
10. С помощью частеречной разметки оставить в стихах только прилагательные и подписать ими каждый стих, на графике из п.9.