Алгоритмы и анализ данных, 2025

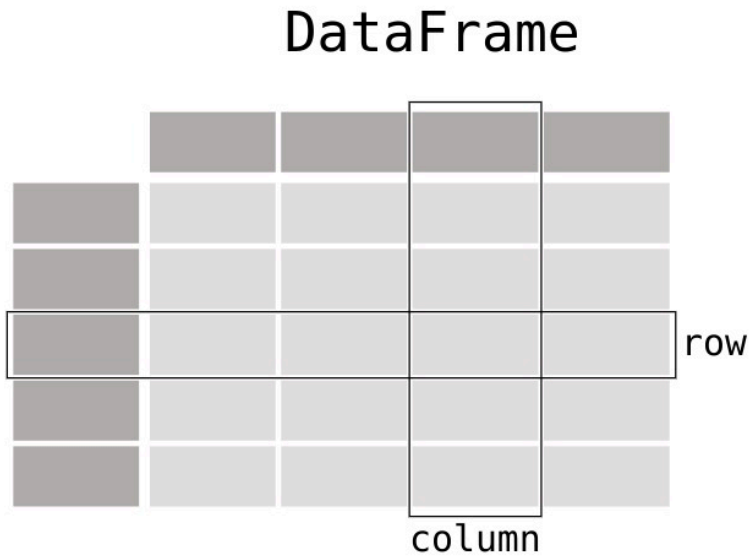
Классические методы векторизации текстов

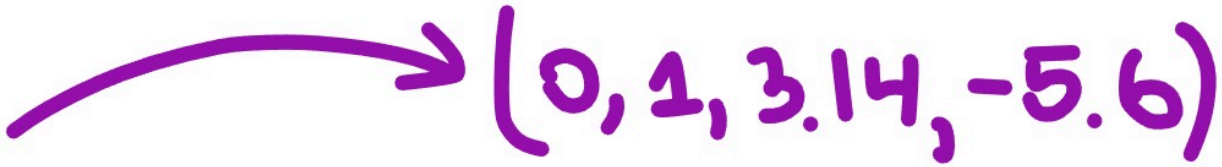
Алексеев Илья

Обработка естественного языка

- векторизация текста
- предобработка текста
- регулярные выражения

Векторизация





[Припев: Big Baby Tape]
Свет мой, зеркало, скажи, покажи мне, кто тут G
Кто был на районе, поставлял барыгам кирпичи?
Передайте мне ключи, передайте мне ключи
Mirror, mirror on the wall, крадусь на них — я вор в ночи

Векторизация

1. Базовые подходы:
    - **Мешок слов (Bag Of Words)**
    - **Tf-Idf**
2. Матричные разложения:
    - LSA/LDA
    - BigARTM
3. Нейросетевые подходы:
    - Word2Vec (Skip-gram, CBoW, FastText, Glove, ...)
    - BERT, sentence embeddings

Что такое текст?

Обучающая коллекция документов (текстов):

$$D = \{d_1, d_2 \dots d_N\}$$

Документ:

$$d_i = (w_1, w_2, \dots w_n),$$

где w_i — токен (слово) из вокабулярия (словаря) V.

Токенизация — разделение текста на токены, элементарные единицы текста

В большинстве случае токен это слово!

Если пользоваться методом `.split()`, токен — последовательность букв, разделённая пробельными символами

Можно использовать регулярные выражения и модули `re`, `regex`.

Можно использовать специальные токенизаторы, например из `nltk`:

- `RegexpTokenizer`
- `BlanklineTokenizer`
- И ещё около десятка штук

```
from nltk.tokenize import word_tokenize

example = 'Но не каждый хочет что-то исправлять:('
word_tokenize(example, language='russian')
```

```
['Но', 'не', 'каждый', 'хочет', 'что-то', 'исправлять', ':(']
```

```
from nltk.tokenize import sent_tokenize

sent = 'Hey! Is Mr. Bing waiting for you?'
nltk.tokenize.sent_tokenize(sent)
```

```
['Hey!', 'Is Mr. Bing waiting for you?']
```

Bag of Words (Count Vectorizer)

Предположим:

- Порядок токенов в тексте не важен
- Важно лишь сколько раз токен w входит в текст d

Term-frequency, число вхождений слова в текст: $\mathrm{tf}(w, d)$

Векторизация:

$$v(d) = (\mathrm{tf}(w_i, d))_{i=1}^{|V|}$$

Bag of Words (Count Vectorizer)

```
from sklearn.feature_extraction.text import CountVectorizer

s = [
    'my name is Joe',
    'your name are Joe',
    'my father is Joe'
]
vectorizer = CountVectorizer()
vectorizer.fit_transform(s).toarray()
```

```
array([[0, 0, 1, 1, 1, 1, 0],
       [1, 0, 0, 1, 0, 1, 1],
       [0, 1, 1, 1, 1, 0, 0]])
```

Проблемы

- Нет учёта контекста и порядка слов
- Учет слов, которые не несут дискриминативной информации
- Огромное признаковое пространство

TODO hashing vectorizer