МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Систем обработки информации и управления»

ОТЧЕТ

Лабораторная работа №__8__
по дисциплине «Методы машинного обучения»

Тема: «Предобработка текста»

ИСПОЛНИТЕЛЬ: ____Савельев А.А.____
ФИО
группа ИУ5-24М _____
подпись
"__"_____2024 г.

ПРЕПОДАВАТЕЛЬ: ___Гапанюк Ю.Е.__
ФИО
_____
подпись
"__"_____2024 г.

Москва - 2024

Лабораторная работа №8 Предобработка текста

Цель лабораторной работы: изучение методов предобработки текстов.

Задание

Для произвольного предложения или текста решить следующие задачи:

- Токенизация.
- Частеречная разметка.
- Лемматизация.
- Выделение (распознавание) именованных сущностей.
- Разбор предложения.

Выполнение работы

Исходный текст:

```
text = "На вершине горы, покрытой снегом, стоит одинокий дом. В этом доме живет старик со своим верным псом. Они проводят дни в уединении, наблюдая за красотой природы и слушая песни ветра. Время от времени старик отправляется в деревню за продуктами и новостями из мира, но всегда возвращается обратно, в свой уютный дом, где его ждет тепло и покой."
```

Токенизация

Токенизация - это процесс разделения текста на более мелкие части, которые называются токенами. Токенами могут быть слова, фразы, символы или другие элементы текста, в зависимости от цели обработки.

1. **Подготовка данных для обработки естественного языка (NLP)**: В NLP многие алгоритмы и модели работают с текстом на уровне отдельных слов или фраз. Токенизация помогает разбить текст на такие отдельные элементы, с которыми модели могут работать.
2. **Облегчение выделения структуры текста**: При анализе текста часто важно понимать его структуру, например, предложения, абзацы или различные части речи. Токенизация помогает выделить эти структурные элементы.
3. **Удаление шума**: Иногда в тексте присутствуют символы или элементы, которые не несут смысловой нагрузки и могут

затруднять анализ. Токенизация может помочь убрать такие шумовые элементы.

```
import nltk
from nltk.tokenize import punkt
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/leatherman/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

```
True
```

```
from nltk import tokenize
dir(tokenize)[:18]
```

```
['BlanklineTokenizer',
 'LegalitySyllableTokenizer',
 'LineTokenizer',
 'MWETokenizer',
 'NLTKWordTokenizer',
 'PunktSentenceTokenizer',
 'RegexpTokenizer',
 'ReppTokenizer',
 'SExprTokenizer',
 'SpaceTokenizer',
 'StanfordSegmenter',
 'SyllableTokenizer',
 'TabTokenizer',
 'TextTilingTokenizer',
 'ToktokTokenizer',
 'TreebankWordDetokenizer',
 'TreebankWordTokenizer',
 'TweetTokenizer']
```

```
nltk_tk = nltk.WordPunctTokenizer()
nltk_tk.tokenize(text)
```

```
['На',
 'вершине',
 'горы',
 ',',
 'покрытой',
 'снегом',
 ',',
 'стоит',
 'одинокий',
 'дом',
 '.',
 'В',
 'этом',
```

```
 'доме',
 'живет',
 'старик',
 'со',
 'своим',
 'верным',
 'псом',
 '.',
 'Они',
 'проводят',
 'дни',
 'в',
 'уединении',
 ',',
 'наблюдая',
 'за',
 'красотой',
 'природы',
 'и',
 'слушая',
 'песни',
 'ветра',
 '.',
 'Время',
 'от',
 'времени',
 'старик',
 'отправляется',
 'в',
 'деревню',
 'за',
 'продуктами',
 'и',
 'новостями',
 'из',
 'мира',
 ',',
 'но',
 'всегда',
 'возвращается',
 'обратно',
 ',',
 'в',
 'свой',
 'уютный',
 'дом',
 ',',
 'где',
 'его',
```

```
 'ждет',
 'тепло',
 'и',
 'покой',
 '.']
```

Токенизация по предложениям

```
nltk_tk_sents = nltk.tokenize.sent_tokenize(text)
print(len(nltk_tk_sents))
nltk_tk_sents
```

```
4

['На вершине горы, покрытой снегом, стоит одинокий дом.',
 'В этом доме живет старик со своим верным псом.',
 'Они проводят дни в уединении, наблюдая за красотой природы и слушая песни ветра.',
 'Время от времени старик отправляется в деревню за продуктами и новостями из мира, но всегда возвращается обратно, в свой уютный дом, где его ждет тепло и покой.']
```

```
from razdel import tokenize, sentenize

n_tok_text = list(tokenize(text))
n_tok_text
```

```
[Substring(0, 2, 'На'),
 Substring(3, 10, 'вершине'),
 Substring(11, 15, 'горы'),
 Substring(15, 16, ','),
 Substring(17, 25, 'покрытой'),
 Substring(26, 32, 'снегом'),
 Substring(32, 33, ','),
 Substring(34, 39, 'стоит'),
 Substring(40, 48, 'одинокий'),
 Substring(49, 52, 'дом'),
 Substring(52, 53, '.'),
 Substring(54, 55, 'В'),
 Substring(56, 60, 'этом'),
 Substring(61, 65, 'доме'),
 Substring(66, 71, 'живет'),
 Substring(72, 78, 'старик'),
 Substring(79, 81, 'со'),
 Substring(82, 87, 'своим'),
 Substring(88, 94, 'верным'),
 Substring(95, 99, 'псом'),
 Substring(99, 100, '.'),
 Substring(101, 104, 'Они'),
 Substring(105, 113, 'проводят'),
```

```
 Substring(114, 117, 'дни'),
 Substring(118, 119, 'в'),
 Substring(120, 129, 'уединении'),
 Substring(129, 130, ','),
 Substring(131, 139, 'наблюдая'),
 Substring(140, 142, 'за'),
 Substring(143, 151, 'красотой'),
 Substring(152, 159, 'природы'),
 Substring(160, 161, 'и'),
 Substring(162, 168, 'слушая'),
 Substring(169, 174, 'песни'),
 Substring(175, 180, 'ветра'),
 Substring(180, 181, '.'),
 Substring(182, 187, 'Время'),
 Substring(188, 190, 'от'),
 Substring(191, 198, 'времени'),
 Substring(199, 205, 'старик'),
 Substring(206, 218, 'отправляется'),
 Substring(219, 220, 'в'),
 Substring(221, 228, 'деревню'),
 Substring(229, 231, 'за'),
 Substring(232, 242, 'продуктами'),
 Substring(243, 244, 'и'),
 Substring(245, 254, 'новостями'),
 Substring(255, 257, 'из'),
 Substring(258, 262, 'мира'),
 Substring(262, 263, ','),
 Substring(264, 266, 'но'),
 Substring(267, 273, 'всегда'),
 Substring(274, 286, 'возвращается'),
 Substring(287, 294, 'обратно'),
 Substring(294, 295, ','),
 Substring(296, 297, 'в'),
 Substring(298, 302, 'свой'),
 Substring(303, 309, 'уютный'),
 Substring(310, 313, 'дом'),
 Substring(313, 314, ','),
 Substring(315, 318, 'где'),
 Substring(319, 322, 'его'),
 Substring(323, 327, 'ждет'),
 Substring(328, 333, 'тепло'),
 Substring(334, 335, 'и'),
 Substring(336, 341, 'покой'),
 Substring(341, 342, '.')]
```

```
[_.text for _ in n_tok_text]
```

```
['На',
 'вершине',
 'горы',
```

```
 ',',
 'покрытой',
 'снегом',
 ',',
 'стоит',
 'одинокий',
 'дом',
 '.',
 'В',
 'этом',
 'доме',
 'живет',
 'старик',
 'со',
 'своим',
 'верным',
 'псом',
 '.',
 'Они',
 'проводят',
 'дни',
 'в',
 'уединении',
 ',',
 'наблюдая',
 'за',
 'красотой',
 'природы',
 'и',
 'слушая',
 'песни',
 'ветра',
 '.',
 'Время',
 'от',
 'времени',
 'старик',
 'отправляется',
 'в',
 'деревню',
 'за',
 'продуктами',
 'и',
 'новостями',
 'из',
 'мира',
 ',',
 'но',
 'всегда',
```

```
 'возвращается',
 'обратно',
 ',',
 'в',
 'свой',
 'уютный',
 'дом',
 ',',
 'где',
 'его',
 'ждет',
 'тепло',
 'и',
 'покой',
 '.']
```

```
n_sen_text = list(sentenize(text))
n_sen_text
```

```
[Substring(0, 53, 'На вершине горы, покрытой снегом, стоит одинокий дом.'),
 Substring(54, 100, 'В этом доме живет старик со своим верным псом.'),
 Substring(101,
           181,
           'Они проводят дни в уединении, наблюдая за красотой природы и слушая песни ветра.'),
 Substring(182,
           342,
           'Время от времени старик отправляется в деревню за продуктами и новостями из мира, но всегда возвращается обратно, в свой уютный дом, где его ждет тепло и покой.')]
```

```
[_.text for _ in n_sen_text], len([_.text for _ in n_sen_text])
```

```
(['На вершине горы, покрытой снегом, стоит одинокий дом.',
  'В этом доме живет старик со своим верным псом.',
  'Они проводят дни в уединении, наблюдая за красотой природы и слушая песни ветра.',
  'Время от времени старик отправляется в деревню за продуктами и новостями из мира, но всегда возвращается обратно, в свой уютный дом, где его ждет тепло и покой.'],
 4)
```

Токенизация для последующей обработки:

```
def n_sentenize(text):
    n_sen_chunk = []
    for sent in sentenize(text):
        tokens = [_.text for _ in tokenize(sent.text)]
```

```
        n_sen_chunk.append(tokens)
    return n_sen_chunk

n_sen_chunk = n_sentenize(text)
n_sen_chunk
```

```
[['На',
  'вершине',
  'горы',
  ',',
  'покрытой',
  'снегом',
  ',',
  'стоит',
  'одинокий',
  'дом',
  '.'],
 ['В',
  'этом',
  'доме',
  'живет',
  'старик',
  'со',
  'своим',
  'верным',
  'псом',
  '.'],
 ['Они',
  'проводят',
  'дни',
  'в',
  'уединении',
  ',',
  'наблюдая',
  'за',
  'красотой',
  'природы',
  'и',
  'слушая',
  'песни',
  'ветра',
  '.'],
 ['Время',
  'от',
  'времени',
  'старик',
  'отправляется',
  'в',
  'деревню',
  'за',
```

```
  'продуктами',
  'и',
  'новостями',
  'из',
  'мира',
  ',',
  'но',
  'всегда',
  'возвращается',
  'обратно',
  ',',
  'в',
  'свой',
  'уютный',
  'дом',
  ',',
  'где',
  'его',
  'ждет',
  'тепло',
  'и',
  'покой',
  '.']]
```

Частеричная разметка

Частеричная разметка текста (частеречная разметка) - это процесс присвоения частей речи каждому слову в предложении. Часть речи определяет роль слова в предложении: существительное, глагол, прилагательное и т.д. Частеречная разметка является важным этапом в обработке естественного языка и используется во многих задачах, таких как морфологический анализ, синтаксический анализ, машинный перевод и другие.

```
from navec import Navec
from slovnet import Morph

# Файл необходимо скачать по ссылке
https://github.com/natasha/navec#downloads
navec = Navec.load('navec_news_v1_1B_250K_300d_100q.tar')

# Файл необходимо скачать по ссылке
https://github.com/natasha/slovnet#downloads
n_morph = Morph.load('slovnet_morph_news_v1.tar', batch_size=4)

morph_res = n_morph.navec(navec)

def print_pos(markup):
    for token in markup.tokens:
        print('{} - {}'.format(token.text, token.tag))
```

```
n_text_markup = list(_ for _ in n_morph.map(n_sen_chunk))
[print_pos(x) for x in n_text_markup]
```

```
На - ADP
вершине - NOUN|Animacy=Inan|Case=Loc|Gender=Fem|Number=Sing
горы - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
, - PUNCT
покрытой - VERB|Aspect=Perf|Case=Gen|Gender=Fem|Number=Sing|
Tense=Past|VerbForm=Part|Voice=Pass
снегом - NOUN|Animacy=Inan|Case=Ins|Gender=Masc|Number=Sing
, - PUNCT
стоит - VERB|Aspect=Imp|Mood=Ind|Number=Sing|Person=3|Tense=Pres|
VerbForm=Fin|Voice=Act
одинокий - ADJ|Case=Nom|Degree=Pos|Gender=Masc|Number=Sing
дом - NOUN|Animacy=Inan|Case=Nom|Gender=Masc|Number=Sing
. - PUNCT
В - ADP
этом - DET|Case=Loc|Gender=Masc|Number=Sing
доме - NOUN|Animacy=Inan|Case=Loc|Gender=Masc|Number=Sing
живет - VERB|Aspect=Imp|Mood=Ind|Number=Sing|Person=3|Tense=Pres|
VerbForm=Fin|Voice=Act
старик - NOUN|Animacy=Anim|Case=Nom|Gender=Masc|Number=Sing
со - ADP
своим - DET|Case=Ins|Gender=Masc|Number=Sing
верным - ADJ|Case=Ins|Degree=Pos|Gender=Masc|Number=Sing
псом - NOUN|Animacy=Anim|Case=Ins|Gender=Masc|Number=Sing
. - PUNCT
Они - PRON|Case=Nom|Number=Plur|Person=3
проводят - VERB|Aspect=Imp|Mood=Ind|Number=Plur|Person=3|Tense=Pres|
VerbForm=Fin|Voice=Act
дни - NOUN|Animacy=Inan|Case=Acc|Gender=Masc|Number=Plur
в - ADP
уединении - NOUN|Animacy=Inan|Case=Loc|Gender=Neut|Number=Sing
, - PUNCT
наблюдая - VERB|Aspect=Imp|Tense=Pres|VerbForm=Conv|Voice=Act
за - ADP
красотой - NOUN|Animacy=Inan|Case=Ins|Gender=Fem|Number=Sing
природы - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
и - CCONJ
слушая - VERB|Aspect=Imp|Tense=Pres|VerbForm=Conv|Voice=Act
песни - NOUN|Animacy=Inan|Case=Acc|Gender=Fem|Number=Plur
ветра - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
. - PUNCT
Время - NOUN|Animacy=Inan|Case=Acc|Gender=Neut|Number=Sing
от - ADP
времени - NOUN|Animacy=Inan|Case=Gen|Gender=Neut|Number=Sing
старик - NOUN|Animacy=Anim|Case=Nom|Gender=Masc|Number=Sing
отправляется - VERB|Aspect=Imp|Mood=Ind|Number=Sing|Person=3|
Tense=Pres|VerbForm=Fin|Voice=Mid
в - ADP
```

```
деревню - NOUN|Animacy=Inan|Case=Acc|Gender=Fem|Number=Sing
за - ADP
продуктами - NOUN|Animacy=Inan|Case=Ins|Gender=Masc|Number=Plur
и - CCONJ
новостями - NOUN|Animacy=Inan|Case=Ins|Gender=Fem|Number=Plur
из - ADP
мира - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
, - PUNCT
но - CCONJ
всегда - ADV|Degree=Pos
возвращается - VERB|Aspect=Imp|Mood=Ind|Number=Sing|Person=3|
Tense=Pres|VerbForm=Fin|Voice=Mid
обратно - ADV|Degree=Pos
, - PUNCT
в - ADP
свой - DET|Case=Acc|Gender=Masc|Number=Sing
уютный - ADJ|Animacy=Inan|Case=Acc|Degree=Pos|Gender=Masc|Number=Sing
дом - NOUN|Animacy=Inan|Case=Acc|Gender=Masc|Number=Sing
, - PUNCT
где - ADV|Degree=Pos
его - PRON|Case=Acc|Gender=Masc|Number=Sing|Person=3
ждет - VERB|Aspect=Imp|Mood=Ind|Number=Sing|Person=3|Tense=Pres|
VerbForm=Fin|Voice=Act
тепло - NOUN|Animacy=Inan|Case=Nom|Gender=Neut|Number=Sing
и - CCONJ
покой - NOUN|Animacy=Inan|Case=Nom|Gender=Masc|Number=Sing
. - PUNCT

[None, None, None, None]
```

Лемматизация

Лемматизация - это процесс приведения слова к его базовой или словарной форме, которая называется леммой. Лемма представляет собой каноническую (нормализованную) форму слова, которая позволяет объединить различные грамматические формы одного и того же слова.

```python
from natasha import Doc, Segmenter, NewsEmbedding, NewsMorphTagger,
MorphVocab

def n_lemmatize(text):
    emb = NewsEmbedding()
    morph_tagger = NewsMorphTagger(emb)
    segmenter = Segmenter()
    morph_vocab = MorphVocab()
    doc = Doc(text)
    doc.segment(segmenter)
    doc.tag_morph(morph_tagger)
    for token in doc.tokens:
```

```
        token.lemmatize(morph_vocab)
    return doc

n_doc = n_lemmatize(text)
{_.text: _.lemma for _ in n_doc.tokens}
```

```
{'На': 'на',
 'вершине': 'вершина',
 'горы': 'гора',
 ',': ',',
 'покрытой': 'покрыть',
 'снегом': 'снег',
 'стоит': 'стоить',
 'одинокий': 'одинокий',
 'дом': 'дом',
 '.': '.',
 'В': 'в',
 'этом': 'этот',
 'доме': 'дом',
 'живет': 'жить',
 'старик': 'старик',
 'со': 'с',
 'своим': 'свой',
 'верным': 'верный',
 'псом': 'пес',
 'Они': 'они',
 'проводят': 'проводить',
 'дни': 'день',
 'в': 'в',
 'уединении': 'уединение',
 'наблюдая': 'наблюдать',
 'за': 'за',
 'красотой': 'красота',
 'природы': 'природа',
 'и': 'и',
 'слушая': 'слушать',
 'песни': 'песня',
 'ветра': 'ветер',
 'Время': 'время',
 'от': 'от',
 'времени': 'время',
 'отправляется': 'отправляться',
 'деревню': 'деревня',
 'продуктами': 'продукт',
 'новостями': 'новость',
 'из': 'из',
 'мира': 'мир',
 'но': 'но',
 'всегда': 'всегда',
 'возвращается': 'возвращаться',
```

```
 'обратно': 'обратно',
 'свой': 'свой',
 'уютный': 'уютный',
 'где': 'где',
 'его': 'он',
 'ждет': 'ждать',
 'тепло': 'тепло',
 'покой': 'покой'}
```

Выделение именованных сущностей

Выделение или распознавание именованных сущностей (NER - Named Entity Recognition) в тексте - это процесс идентификации и классификации конкретных объектов или сущностей в тексте, таких как имена людей, местоположения, даты, организации и т.д.

```
from slovnet import NER
from ipymarkup import show_span_ascii_markup as show_markup

# Файл необходимо скачать по ссылке
https://github.com/natasha/slovnet#downloads
ner = NER.load('slovnet_ner_news_v1.tar')
ner_res = ner.navec(navec)
markup_ner = ner(text)

markup_ner
```

```
SpanMarkup(
    text='На вершине горы, покрытой снегом, стоит одинокий дом. В этом
доме живет старик со своим верным псом. Они проводят дни в уединении,
наблюдая за красотой природы и слушая песни ветра. Время от времени
старик отправляется в деревню за продуктами и новостями из мира, но
всегда возвращается обратно, в свой уютный дом, где его ждет тепло и
покой.',
    spans=[]
)
```

```
show_markup(markup_ner.text, markup_ner.spans)
```

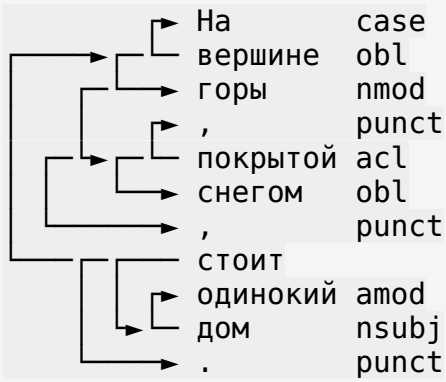
```
На вершине горы, покрытой снегом, стоит одинокий дом. В этом доме
живет старик со своим верным псом. Они проводят дни в уединении,
наблюдая за красотой природы и слушая песни ветра. Время от времени
старик отправляется в деревню за продуктами и новостями из мира, но
всегда возвращается обратно, в свой уютный дом, где его ждет тепло и
покой.
```

Разбор предложения

```
from natasha import NewsSyntaxParser
```

```
emb = NewsEmbedding()
syntax_parser = NewsSyntaxParser(emb)

n_doc.parse_syntax(syntax_parser)
n_doc.sents[0].syntax.print()
```



```
На       case
вершине  obl
горы     nmod
,        punct
покрытой acl
снегом   obl
,        punct
стоит
одинокий amod
дом      nsubj
.        punct
```

```
from spacy.lang.ru import Russian
import spacy
from spacy import displacy

nlp = spacy.load('ru_core_news_sm')
spacy_text1 = nlp(text)
for t in spacy_text1[:20]: print(t)
```

```
На
вершине
горы
,
покрытой
снегом
,
стоит
одинокий
дом
.
В
этом
доме
живет
старик
со
своим
верным
псом
```

```
displacy.render(spacy_text1, style='dep', jupyter=True)
```

```
<IPython.core.display.HTML object>
```

Вывод:

В ходе выполнения домашнего задания была проведена предобработка текста, которая подготовила его к разнообразным типам анализа, основанным как на анализе отдельных слов, так и полных предложений.