МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

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

ОТЧЕТ

Лабораторная работа № 8 (ДЗ)
по дисциплине «Методы машинного обучения»

Тема: «Предобработка текста»

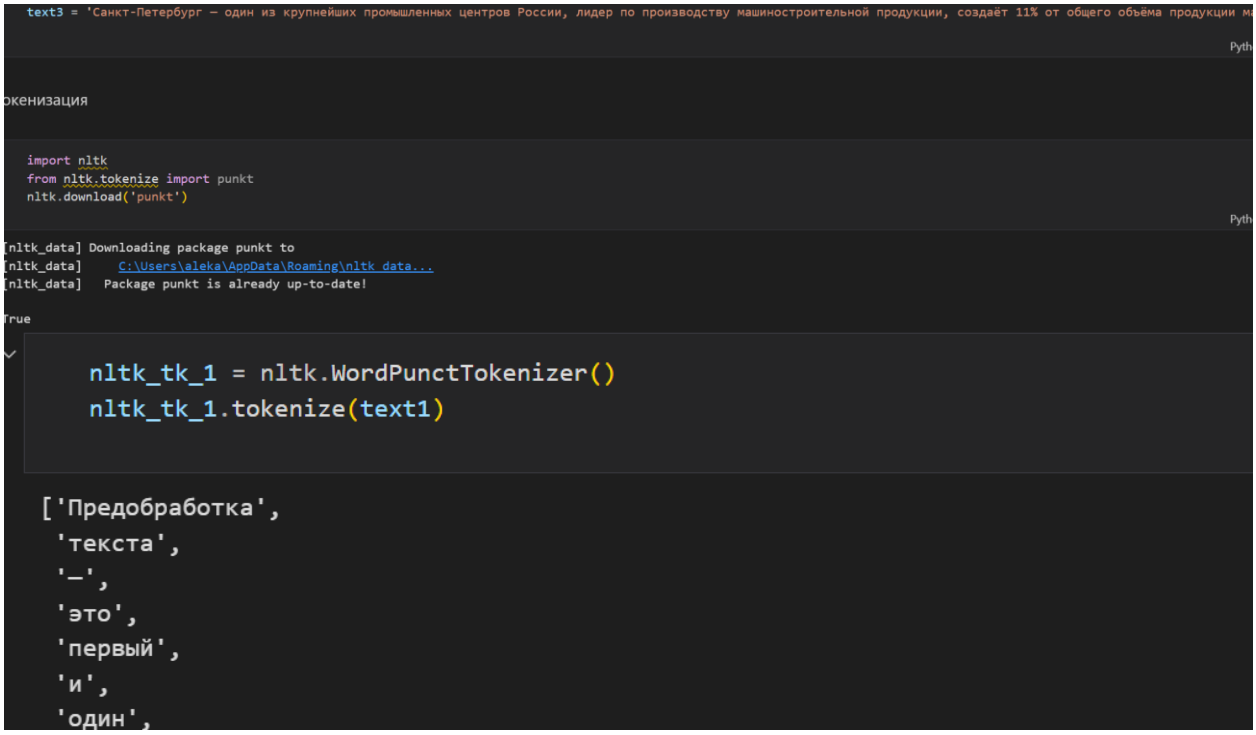
ИСПОЛНИТЕЛЬ: ___Кравцов А.Н.__
ФИО
группа ИУ5-24М _____
подпись
"26" апреля 2024 г.

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

Москва – 2024

Задание

ля произвольного предложения или текста решите следующие задачи: 1. Токенизация. 2. Частеречная разметка. 3. Лемматизация. 4. Выделение (распознавание) именованных сущностей. 5. Разбор предложения.



```
    n_tok_text1 = list(tokenize(text1))
    n_tok_text1
```

```
[Substring(0, 13, 'Предобработка'),
 Substring(14, 20, 'текста'),
 Substring(21, 22, '–'),
 Substring(23, 26, 'это'),
 Substring(27, 33, 'первый'),
 Substring(34, 35, 'и'),
 Substring(36, 40, 'один'),
 Substring(41, 43, 'из'),
 Substring(44, 52, 'наиболее'),
 Substring(53, 59, 'важных'),
 Substring(60, 66, 'этапов'),
 Substring(67, 68, 'в'),
 Substring(69, 78, 'обработке'),
 Substring(79, 92, 'естественного'),
 Substring(93, 98, 'языка'),
 Substring(99, 100, '('),
 Substring(100, 103, 'NLP'),
 Substring(103, 104, ')'),
 Substring(105, 106, 'с'),
 Substring(107, 121, 'использованием'),
 Substring(122, 131, 'нейронных'),
 Substring(132, 137, 'сетей'),
 Substring(137, 138, '.'),
 Substring(139, 143, 'Этот'),
 Substring(144, 151, 'процесс'),
```

```
[_.text for _ in n_tok_text1]
```

```
['Предобработка',
 'текста',
 '—',
 'это',
 'первый',
 'и',
 'один',
 'из',
 'наиболее',
 'важных',
 'этапов',
 'в',
 'обработке',
 'естественного',
```

```
n_sen_text1 = list(sentenize(text1))
n_sen_text1
```

Python

```
[Substring(0,
           138,
           'Предобработка текста — это первый и один из наиболее важных этапов в обработке естественного языка (NLP) с использованием нейронных сетей.'),
 Substring(139,
           304,
           'Этот процесс включает в себя серию операций, предназначенных для преобразования исходного текста в формат, который может быть эффективно обработан нейронными сет
```

```
[_.text for _ in n_sen_text1], len([_.text for _ in n_sen_text1])
```

Python

```
(['Предобработка текста — это первый и один из наиболее важных этапов в обработке естественного языка (NLP) с использованием нейронных сетей.',
  'Этот процесс включает в себя серию операций, предназначенных для преобразования исходного текста в формат, который может быть эффективно обработан нейронными сетями.'],
 2)
```

```
# Этот вариант токенизации нужен для последующей обработки
def n_sentenize(text):
    n_sen_chunk = 
    for sent in se
        tokens = [_.text for _ in tokenize(sent.text)]
        n_sen_chunk.append(tokens)
    return n_sen_chunk
```

(variable) _: Any

Python

```
# Этот вариант токенизации нужен 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
```

11]

Python

```
n_sen_chunk_1 = n_sentenize(text1)
n_sen_chunk_1
```

12]

Python

```
[['Предобработка',
  'текста',
  '—',
  'это',
  'первый',
  'и',
  'один',
  'из',
  'наиболее',
  'важных',
  'этапов',
  'в',
  'обработке',
```

```python
n_sen_chunk_2 = n_sentenize(text2)
n_sen_chunk_2
```

Python

```
[['Шла', 'Саша', 'по', 'шоссе', 'и', 'сосала', 'сушку', '.']]
```

```python
n_sen_chunk_3 = n_sentenize(text3)
n_sen_chunk_3
```

Python

```
[['Санкт-Петербург',
  '–',
  'один',
  'из',
  'крупнейших',
  'промышленных',
  'центров',
  'России',
  ',',
  'лидер',
```

Частеречная разметка

```python
from navec import Navec
from slovnet import Morph
```

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

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

```python
morph_res = n_morph.navec(navec)
```

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

```python
n_text1_markup = list(_ for _ in n_morph.map(n_sen_chunk_1))
[print_pos(x) for x in n_text1_markup]
```

```
n_text1_markup = list(_ for _ in n_morph.map(n_sen_chunk_1))
[print_pos(x) for x in n_text1_markup]
```

20]

```
Предобработка - ADJ|Case=Gen|Degree=Pos|Gender=Masc|Number=Sing
текста - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
— - PUNCT
это - PART
первый - ADJ|Case=Nom|Degree=Pos|Gender=Masc|Number=Sing
и - CCONJ
один - NUM|Case=Nom|Gender=Masc|Number=Sing
из - ADP
наиболее - ADV|Degree=Pos
важных - ADJ|Case=Gen|Degree=Pos|Number=Plur
этапов - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Plur
в - ADP
обработке - NOUN|Animacy=Inan|Case=Loc|Gender=Fem|Number=Sing
естественного - ADJ|Case=Gen|Degree=Pos|Gender=Masc|Number=Sing
языка - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
( - PUNCT
NLP - PROPN|Foreign=Yes
) - PUNCT
с - ADP
использованием - NOUN|Animacy=Inan|Case=Ins|Gender=Neut|Number=Sing
```

```
n_text3_markup = list(n_morph.map(n_sen_chunk_3))
[print_pos(x) for x in n_text3_markup]
```

22]

```
Санкт-Петербург - PROPN|Animacy=Inan|Case=Nom|Gender=Masc|Number=Sing
— - PUNCT
один - NUM|Case=Nom|Gender=Masc|Number=Sing
из - ADP
крупнейших - ADJ|Case=Gen|Degree=Sup|Number=Plur
промышленных - ADJ|Case=Gen|Degree=Pos|Number=Plur
центров - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Plur
России - PROPN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
, - PUNCT
лидер - NOUN|Animacy=Anim|Case=Nom|Gender=Masc|Number=Sing
по - ADP
производству - NOUN|Animacy=Inan|Case=Dat|Gender=Neut|Number=Sing
машиностроительной - ADJ|Case=Gen|Degree=Pos|Gender=Fem|Number=Sing
продукции - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
, - PUNCT
создаёт - VERB|Aspect=Imp|Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin|Voice=Act
11 - NUM
% - SYM
от - ADP
общего - ADJ|Case=Gen|Degree=Pos|Gender=Masc|Number=Sing
объёма - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
продукции - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
машиностроения - NOUN|Animacy=Inan|Case=Gen|Gender=Neut|Number=Sing
по - ADP
стране - NOUN|Animacy=Inan|Case=Dat|Gender=Fem|Number=Sing
. - PUNCT
```

Лемматизация

```
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_doc1 = n_lemmatize(text1)
{_.text: _.lemma for _ in n_doc1.tokens}
```

```
{'Предобработка': 'предобработка',
 'текста': 'текст',
 '—': '—',
 'это': 'это',
```

```
    n_doc2 = n_lemmatize(text2)
    {_.text: _.lemma for _ in n_doc2.tokens}
```

```
{'Шла': 'идти',
 'Саша': 'саша',
 'по': 'по',
 'шоссе': 'шоссе',
 'и': 'и',
 'сосала': 'сосать',
 'сушку': 'сушка',
 '.': '.'}
```

```
    n_doc3 = n_lemmatize(text3)
    {_.text: _.lemma for _ in n_doc3.tokens}
```

```
{'Санкт-Петербург': 'санкт-петербург',
 '—': '—',
 'один': 'один',
 'из': 'из',
 'крупнейших': 'крупный',
 'промышленных': 'промышленный',
 'центров': 'центр',
 'России': 'россия',
 ',': ',',
 'лидер': 'лидер',
 'по': 'по',
 'производству': 'производство',
 'машиностроительной': 'машиностроительный',
```

Выделение (распознавание) именованных сущностей

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

```
ner = NER.load('slovnet_ner_news_v1.tar')
ner_res = ner.navec(navec)
markup_ner3 = ner(text3)
```

```
markup_ner3
```

```
SpanMarkup(
    text='Санкт-Петербург — один из крупнейших промышленных центров России, лидер по производству машиностроительной продукц
    spans=[Span(
         start=0,
         stop=15,
         type='LOC'
     ),
     Span(
         start=58,
         stop=64,
         type='LOC'
     )]
)
```

Выделение (распознавание) именованных сущностей

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

```
ner = NER.load('slovnet_ner_news_v1.tar')
ner_res = ner.navec(navec)
markup_ner3 = ner(text3)
```

```
markup_ner3
```

```
SpanMarkup(
    text='Санкт-Петербург — один из крупнейших промышленных центров России, лидер по производству машиностроительной проду
    spans=[Span(
         start=0,
         stop=15,
         type='LOC'
     ),
     Span(
         start=58,
         stop=64,
         type='LOC'
     )]
)
```

Разбор предложения

```
from natasha import NewsSyntaxParser
```

+ Code

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

```
n_doc1.parse_syntax(syntax_parser)
n_doc1.sents[0].syntax.print()
```

```
                ┌► Предобработка  amod
             ┌─└─ текста
             └──► –               punct
          ┌──────► это             expl
          │  ┌───►┌─── первый      parataxis
          │  │    │  ┌► и          cc
          │  │    └─►└─ один       conj
          │  │ ┌──────► из         case
          │  │ │    ┌► наиболее    advmod
          │  │ │ ┌─►└─ важных      amod
       ┌──└──└─└─└──── этапов      nmod
       │     │  │    ┌► в          case
       │  ┌─►└──└────└─ обработке  nmod
       │  │          ┌► естественного amod
       │ ┌└──────────└─ языка      nmod
       │ │          ┌► (          punct
       │ └─────────►┌└─ NLP        appos
```

```
n_doc3.parse_syntax(syntax_parser)
n_doc3.sents[0].syntax.print()
```