Московский государственный технический университет им. Н.Э.

Баумана Кафедра «Системы обработки информации и управления»

Лабораторная работа №5

по дисциплине

«Методы машинного обучения»

на тему

«Предобработка и классификация текстовых данных»

Выполнил:

студент группы ИУ5И-22М

Чжан Аньци

Москва — 2022 г.

In [1]:

```
text='Николай Эрнестович Бауман родился в семье владельца о
```

Задание:

1. Токенизация

Для выполнения работы испоьзована библиотека 'Natasha'

In [2]:

```
from razdel import tokenize, sentenize
n_tok_text = list(tokenize(text))
n_tok_text
```

Out[2]:

```
[Substring(0, 1, 'Н'),
 Substring(1, 5, 'и к о л'),
 Substring(5, 6, 'а'),
 Substring(6, 7, 'й'),
 Substring(8, 11, 'Э р н'),
 Substring(11, 12, 'е'),
 Substring(12, 18, 'с т о в и ч'),
 Substring(19, 20, 'Б'),
 Substring(20, 21, 'а'),
 Substring(21, 23, 'у м'),
 Substring(23, 24, 'а'),
 Substring(24, 25, 'н'),
 Substring(26, 33, 'р о д и л с я'),
 Substring(34, 35, 'в'),
 Substring(36, 41, 'с е м ь е'),
 Substring(42, 51, 'в л а д е л ь ц а'),
 Substring(52, 59, 'о б о й н о й'),
 Substring(60, 61, 'и'),
 Substring(62, 71, 'с т о л я р н о й'),
 Substring(72, 82, 'м а с т е р с к о й'),
 Substring(82, 83, '.'),
 Substring(84, 85, 'В'),
 Substring(86, 95, '1891—1895'),
 Substring(96, 101, 'г о д а х'),
 Substring(102, 105, 'б ы л'),
 Substring(106, 115, 'с т у д е н т о м'),
 Substring(116, 126, 'К а з а н с к о г о'),
 Substring(127, 140, 'в е т е р и н а р н о г о'),
 Substring(141, 150, 'и н с т и т у т а'),
 Substring(150, 151, '.'),
 Substring(152, 153, 'В'),
 Substring(154, 158, 'г о д ы'),
 Substring(159, 164, 'у ч ё б ы'),
 Substring(165, 172, 'у в л ё к с я'),
 Substring(173, 184, 'н е л е г а л ь н о й'),
 Substring(185, 198, 'н а р о д н и ч е с к о й'),
 Substring(199, 200, 'и'),
 Substring(201, 213, 'м а р к с и с т с к о й'),
 Substring(214, 225, 'л и т е р а т у р о й'),
 Substring(225, 226, ','),
 Substring(227, 237, 'у ч а с т в о в а л'),
 Substring(238, 239, 'в'),
 Substring(240, 246, 'р а б о т е'),
 Substring(247, 257, 'п о д п о л ь н ы х'),
 Substring(258, 265, 'р а б о ч и х'),
 Substring(266, 273, 'к р у ж к о в'),
 Substring(273, 274, '.')]
```

In [3]:

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

Out[3]:

```
['Н',
 'и к о л',
 'а',
 'й',
 'Э р н',
 'е',
 'с т о в и ч',
 'Б',
 'а',
 'у м',
 'а',
 'н',
 'р о д и л с я',
 'в',
 'с е м ь е',
 'в л а д е л ь ц а',
 'о б о й н о й',
 'и',
 'с т о л я р н о й',
 'м а с т е р с к о й',
 '.',
 'В',
 '1891—1895',
 'г о д а х',
 'б ы л',
 'с т у д е н т о м',
 'К а з а н с к о г о',
 'в е т е р и н а р н о г о',
 'и н с т и т у т а',
 '.',
 'В',
 'г о д ы',
 'у ч ё б ы',
 'у в л ё к с я',
 'н е л е г а л ь н о й',
 'н а р о д н и ч е с к о й',
 'и',
 'м а р к с и с т с к о й',
 'л и т е р а т у р о й',
 ',',
 'у ч а с т в о в а л',
 'в',
 'р а б о т е',
 'п о д п о л ь н ы х',
 'р а б о ч и х',
 'к р у ж к о в',
 '.']
```

In [4]:

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

Out[4]:

```
[Substring(0,
           83,
           'Николай Эрнестович Бауман родился в семье
владельца обойной и столярной мастерской.'),
 Substring(84,
           151,
           'В 1891—1895 годах был студентом\xa0Казанског
о ветеринарного института.'),
 Substring(152,
           274,
           'В годы учёбы увлёкся нелегальной народ
нической и марксистской литературой, участво
вал в работе подпольных рабочих кружков.')]
```

In [5]:

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

Out[5]:

```
(['Николай Эрнестович Бауман родился в семье влад
ельца обойной и столярной мастерской.',
  'В 1891—1895 годах был студентом\xa0Казанского вет
еринарного института.',
  'В годы учёбы увлёкся нелегальной народничес
кой и марксистской литературой, участвовал в
работе подпольных рабочих кружков.'],
 3)
```

In [6]:

```
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
```

In [7]:

```
n_sen_chunk = n_sentenize(text)
n_sen_chunk
```

Out[7]:

```
[['Н',
  'и к о л',
  'а',
  'й',
  'Э р н',
  'е',
  'с т о в и ч',
  'Б',
  'а',
  'у м',
  'а',
  'н',
  'р о д и л с я',
  'в',
  'с е м ь е',
  'в л а д е л ь ц а',
  'о б о й н о й',
  'и',
  'с т о л я р н о й',
  'м а с т е р с к о й',
  '.'],
 ['В',
  '1891—1895',
  'г о д а х',
  'б ы л',
  'с т у д е н т о м',
  'К а з а н с к о г о',
  'в е т е р и н а р н о г о',
  'и н с т и т у т а',
  '.'],
 ['В',
  'г о д ы',
  'у ч ё б ы',
  'у в л ё к с я',
  'н е л е г а л ь н о й',
  'н а р о д н и ч е с к о й',
  'и',
  'м а р к с и с т с к о й',
  'л и т е р а т у р о й',
  ',',
  'у ч а с т в о в а л',
  'в',
  'р а б о т е',
  'п о д п о л ь н ы х',
  'р а б о ч и х',
  'к р у ж к о в',
  '.']]
```

2.Частеречная разметка

In [8]:

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

In [9]:

```
navec = Navec.load('navec_news_v1_1B_250K_300d_100q.tar')
```

In [10]:

```
n_morph = Morph.load('slovnet_morph_news_v1.tar', batch_size=4)
```

In [11]:

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

In [12]:

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

In [13]:

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

```
Н - NOUN
и к о л - X|Foreign=Yes
а - X|Foreign=Yes
й - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
Э р н - PROPN|Foreign=Yes
е - X|Foreign=Yes
с т о в и ч - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
Б - PROPN
а - X|Foreign=Yes
у м - NOUN|Animacy=Inan|Case=Nom|Gender=Masc|Number=Sing
а - NOUN
н - X|Foreign=Yes
р о д и л с я - VERB|Aspect=Perf|Gender=Masc|Mood=Ind|Number=Sing|Tense=Past|VerbFo
rm=Fin|Voice=Mid
в - ADP
с е м ь е - NOUN|Animacy=Inan|Case=Loc|Gender=Fem|Number=Sing
в л а д е л ь ц а - NOUN|Animacy=Anim|Case=Gen|Gender=Masc|Number=Sing
о б о й н о й - NOUN|Animacy=Anim|Case=Gen|Gender=Masc|Number=Sing
и - CCONJ
с т о л я р н о й - ADJ|Case=Gen|Degree=Pos|Gender=Fem|Number=Sing
м а с т е р с к о й - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
. - PUNCT
В - ADP
1891—1895 - ADJ
г о д а х - NOUN|Animacy=Inan|Case=Loc|Gender=Masc|Number=Plur
б ы л - AUX|Aspect=Imp|Gender=Masc|Mood=Ind|Number=Sing|Tense=Past|VerbForm=Fin|Voi
ce=Act
с т у д е н т о м - NOUN|Animacy=Anim|Case=Ins|Gender=Masc|Number=Sing
К а з а н с к о г о - ADJ|Case=Gen|Degree=Pos|Gender=Masc|Number=Sing
в е т е р и н а р н о г о - ADJ|Case=Gen|Degree=Pos|Gender=Masc|Number=Sing
и н с т и т у т а - NOUN|Animacy=Inan|Case=Gen|Gender=Masc|Number=Sing
. - PUNCT
В - ADP
г о д ы - NOUN|Animacy=Inan|Case=Acc|Gender=Masc|Number=Plur
у ч ё б ы - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Sing
у в л ё к с я - VERB|Aspect=Perf|Gender=Masc|Mood=Ind|Number=Sing|Tense=Past|VerbFo
rm=Fin|Voice=Mid
н е л е г а л ь н о й - ADJ|Case=Ins|Degree=Pos|Gender=Fem|Number=Sing
н а р о д н и ч е с к о й - NOUN|Animacy=Inan|Case=Ins|Gender=Fem|Number=Sing
и - CCONJ
м а р к с и с т с к о й - ADJ|Case=Ins|Degree=Pos|Gender=Fem|Number=Sing
л и т е р а т у р о й - NOUN|Animacy=Inan|Case=Ins|Gender=Fem|Number=Sing
, - PUNCT
у ч а с т в о в а л - VERB|Aspect=Imp|Gender=Masc|Mood=Ind|Number=Sing|Tense=Past|V
erbForm=Fin|Voice=Act
в - ADP
р а б о т е - NOUN|Animacy=Inan|Case=Loc|Gender=Fem|Number=Sing
п о д п о л ь н ы х - ADJ|Case=Gen|Degree=Pos|Number=Plur
р а б о ч и х - ADJ|Case=Gen|Degree=Pos|Number=Plur
к р у ж к о в - NOUN|Animacy=Inan|Case=Gen|Gender=Fem|Number=Plur
. - PUNCT
```

Out[13]:

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

3.Лемматизация

In [14]:

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

In [15]:

```
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
```

In [16]:

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

Out[16]:

```
{'Н': 'h',
 'икол': 'икол',
 'а': 'а',
 'й': 'й',
 'Эрн': 'эрна',
 'е': 'е',
 'стович': 'стович',
 'Б': 'б',
 'ум': 'ум',
 'н': 'н',
 'родился': 'родиться',
 'в': 'в',
 'семье': 'семья',
 'владельца': 'владелец',
 'обойной': 'обойный',
 'и': 'и',
 'столярной': 'столярный',
 'мастерской': 'мастерская',
 '.': '.',
 'В': 'в',
 '1891—1895': '1891—1895',
 'годах': 'год',
 'был': 'быть',
 'студентом': 'студент',
 'Казанского': 'казанский',
 'ветеринарного': 'ветеринарный',
 'института': 'институт',
 'годы': 'год',
 'учёбы': 'учеба',
 'увлёкся': 'увлечься',
 'нелегальной': 'нелегальный',
 'народнической': 'народнический',
 'марксистской': 'марксистский',
 'литературой': 'литература',
 ',': ',',
 'участвовал': 'участвовать',
 'работе': 'работа',
 'подпольных': 'подпольный',
 'рабочих': 'рабочий',
 'кружков': 'кружок'}
```

4.Выделение (распознавание) именованных сущностей

named-entity recognition (NER)

In [17]:

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

In [18]:

```
ner = NER.load('slovnet_ner_news_v1.tar')
```

In [19]:

```
ner_res = ner.navec(navec)
```

In [20]:

```
markup_ner = ner(text)
markup_ner
```

Out[20]:

```
SpanMarkup(
    text='Николай Эрнестович Бауман родился в семье
владельца обойной и столярной мастерской. В 189
1—1895 годах был студентом\xa0Казанского ветерин
арного института. В годы учёбы увлёкся нелега
льной народнической и марксистской литератур
ой, участвовал в работе подпольных рабочих кр
ужков.',
    spans=[Span(
         start=116,
         stop=150,
         type='ORG'
     )]
)
```

In [21]:

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

```
Николай Эрнестович Бауман родился в семье владе
льца обойной и
столярной мастерской. В 1891—1895 годах был студе
нтом Казанского
                                                ORG————————
ветеринарного института. В годы учёбы увлёкся
нелегальной
————————————————————————————————————————
народнической и марксистской литературой, уч
аствовал в работе
подпольных рабочих кружков.
```

5.Разбор предложения

In [22]:

```
from natasha import NewsSyntaxParser
```

In [23]:

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

In [24]:

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

```
┌──────────────────► Н          advmod
│  ┌►┌─┌─┌─┌─┌─ и к о л        obl
│  │ │ │ │ │ └► а          appos
│  │ │ │ │ └►┌─ й            appos
│  │ │ │ │   └► Э р н        nmod
│  │ │ │ │   ┌► е          appos
│  │ │ └►┌─└─ с т о в и ч    appos
│  │ │  ┌─└►┌─ Б            appos
│  │ │  │   └► а          appos
│  │ └►┌─┌─ у м            appos
│  │   └──►│ а          flat:foreign
│  │       └► н          appos
└─ └─ ┌──────┌──────── р о д и л с я
     │  │ ┌► в            case
     │  └►└─ с е м ь е      obl
     │ ┌──└►┌─ в л а д е л ь ц а  nmod
     │ │  ┌─└► о б о й н о й    nmod
     │ │  │ ┌► и            cc
     │ │  └►└─ с т о л я р н о й  conj
     │ └──────► м а с т е р с к о й nmod
     └────────► .            punct
```

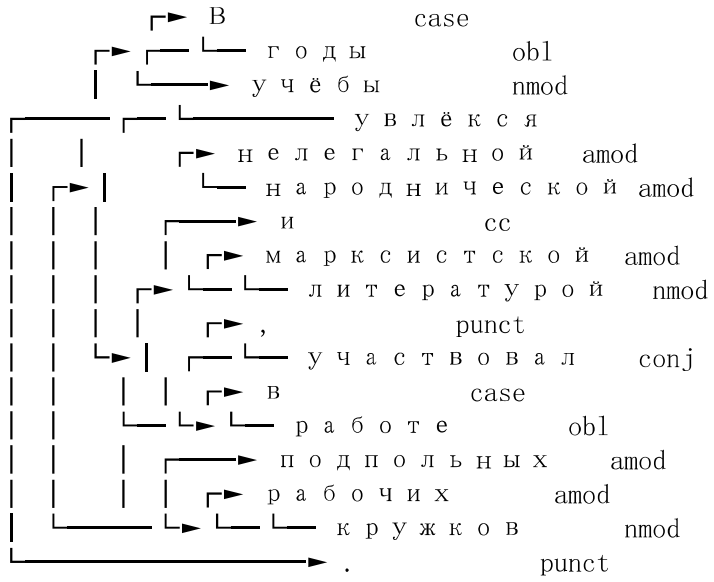
In [25]:

```
n_doc.sents[1].syntax.print()
```

```
      ┌─────► В              case
      │  ┌──► 1891—1895      amod
   ┌► └─└─└─ г о д а х          nmod
   │  ┌► б ы л             cop
┌─ ┌─└──────└─ с т у д е н т о м
│  │  ┌─────► К а з а н с к о г о    amod
│  │  │  ┌──► в е т е р и н а р н о г о amod
│  └─►└─└─└─ и н с т и т у т а      nmod
└──────────► .              punct
```

In [26]:

```
n_doc.sents[2].syntax.print()
```



In [ ]: