#Отчет по лабораторной работе №5 по ММО Кудрявцев С.Д. ИУ5-22М

Spacy токенизация

```
from spacy.lang.ru import Russian
import spacy
import pandas as pd

df = pd.read_csv("SPAM.csv")
df.head()
```

```
  Category                                            Message
0      ham  Go until jurong point, crazy.. Available only ...
1      ham                      Ok lar... Joking wif u oni...
2     spam  Free entry in 2 a wkly comp to win FA Cup fina...
3      ham  U dun say so early hor... U c already then say...
4      ham  Nah I don't think he goes to usf, he lives aro...
```

```
text = df['Message'][0]
!python -m spacy download en_core_web_sm
nlp = spacy.load('en_core_web_sm')
```

```
Collecting en_core_web_sm==2.2.5
  Downloading
https://github.com/explosion/spacy-models/releases/download/en_core_we
b_sm-2.2.5/en_core_web_sm-2.2.5.tar.gz (12.0 MB)
ent already satisfied: spacy>=2.2.2 in /usr/local/lib/python3.7/dist-
packages (from en_core_web_sm==2.2.5) (2.2.4)
Requirement already satisfied: plac<1.2.0,>=0.9.6 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (1.1.3)
Requirement already satisfied: setuptools in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (57.4.0)
Requirement already satisfied: catalogue<1.1.0,>=0.0.7 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (1.0.0)
Requirement already satisfied: blis<0.5.0,>=0.4.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (0.4.1)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (4.64.0)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (2.23.0)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (2.0.6)
Requirement already satisfied: srsly<1.1.0,>=1.0.2 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
```

```
>en_core_web_sm==2.2.5) (1.0.5)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (1.0.7)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (3.0.6)
Requirement already satisfied: wasabi<1.1.0,>=0.4.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (0.9.1)
Requirement already satisfied: numpy>=1.15.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (1.21.6)
Requirement already satisfied: thinc==7.4.0 in
/usr/local/lib/python3.7/dist-packages (from spacy>=2.2.2-
>en_core_web_sm==2.2.5) (7.4.0)
Requirement already satisfied: importlib-metadata>=0.20 in
/usr/local/lib/python3.7/dist-packages (from catalogue<1.1.0,>=0.0.7-
>spacy>=2.2.2->en_core_web_sm==2.2.5) (4.11.3)
Requirement already satisfied: zipp>=0.5 in
/usr/local/lib/python3.7/dist-packages (from importlib-metadata>=0.20-
>catalogue<1.1.0,>=0.0.7->spacy>=2.2.2->en_core_web_sm==2.2.5) (3.8.0)
Requirement already satisfied: typing-extensions>=3.6.4 in
/usr/local/lib/python3.7/dist-packages (from importlib-metadata>=0.20-
>catalogue<1.1.0,>=0.0.7->spacy>=2.2.2->en_core_web_sm==2.2.5) (4.2.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.13.0-
>spacy>=2.2.2->en_core_web_sm==2.2.5) (2021.10.8)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.13.0-
>spacy>=2.2.2->en_core_web_sm==2.2.5) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in
/usr/local/lib/python3.7/dist-packages (from requests<3.0.0,>=2.13.0-
>spacy>=2.2.2->en_core_web_sm==2.2.5) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1
in /usr/local/lib/python3.7/dist-packages (from
requests<3.0.0,>=2.13.0->spacy>=2.2.2->en_core_web_sm==2.2.5) (1.24.3)
✔ Download and installation successful
You can now load the model via spacy.load('en_core_web_sm')
```

```
spacy_text = nlp(text)
for t in spacy_text:
    print(t)
```

```
Go
until
jurong
point
,
crazy
..
```

```
Available
only
in
bugis
n
great
world
la
e
buffet
...
Cine
there
got
amore
wat
...
```

Частеречная разметка

```
for token in spacy_text:
    print('{} - {} - {}'.format(token.text, token.pos_, token.dep_))
```

```
Go - VERB - ROOT
until - ADP - prep
jurong - PROPN - compound
point - NOUN - pobj
, - PUNCT - punct
crazy - ADJ - amod
.. - PUNCT - punct
Available - ADJ - ROOT
only - ADV - advmod
in - ADP - prep
bugis - PROPN - nmod
n - PROPN - pobj
great - PROPN - advcl
world - PROPN - intj
la - PROPN - compound
e - PROPN - compound
buffet - NOUN - ROOT
... - PUNCT - punct
Cine - PROPN - nsubj
there - PRON - advmod
got - VERB - ROOT
amore - ADJ - amod
wat - PROPN - dobj
... - PUNCT - punct
```

Лемматизация

```
for token in spacy_text:
      print(token, token.lemma, token.lemma_)
```

```
Go 8004577259940138793 go
until 2906777282977359384 until
jurong 14096818269429646414 jurong
point 15479733260938818482 point
, 2593208677638477497 ,
crazy 16792598155547231470 crazy
.. 8848021949395737739 ..
Available 4887332976578131782 available
only 13398675276606405380 only
in 3002984154512732771 in
bugis 3557720637082594454 bugis
n 13210364986222294696 n
great 8881679497796027013 great
world 17034894182720521 82 world
la 6804705863737483857 la
e 1720370409040345145 e
buffet 95032404870770799 50 buffet
... 10875615029400813363 ...
Cine 3040539950272399101 Cine
there 2112642640949226496 there
got 2013399242189103424 get
amore 15198862986666964157 amore
wat 8445541832527975875 wat
... 10875615029400813363 ...
```

Выделение именованных сущностей

```
text1 = "Robert 'Bob' Abram Bartlett (August 15, 1875 – April 28,
1946) was a Newfoundland-born American Arctic explorer of the late
19th and early 20th centuries."
spacy_text1 = nlp(text1)
for ent in spacy_text1.ents:
    print(ent.text, ent.label_)
```

```
Robert 'Bob' Abram Bartlett PERSON
August 15, 1875 – DATE
April 28, 1946 DATE
Newfoundland GPE
American NORP
Arctic PRODUCT
the late 19th and early 20th centuries DATE
```

```
from spacy import displacy
displacy.render(spacy_text1, style='ent', jupyter=True)
```

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

Разбор предложения

```
from spacy import displacy

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

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

Классификация с помощью TfidfVectorizer

```
import numpy as np
import pandas as pd
from typing import Dict, Tuple
from scipy import stats
from IPython.display import Image
from sklearn.datasets import load_iris, load_boston
from sklearn.feature_extraction.text import CountVectorizer,
TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor,
KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score,
classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error,
mean_squared_log_error, median_absolute_error, r2_score
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR,
NuSVR, LinearSVR
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

```
vocab_list = df['Message'].tolist()
tfidfv = TfidfVectorizer(ngram_range=(1,3))
tfidf_ngram_features = tfidfv.fit_transform(vocab_list)
```

```
def VectorizeAndClassify(vectorizers_list, classifiers_list):
    for v in vectorizers_list:
        for c in classifiers_list:
            pipeline1 = Pipeline([("vectorizer", v), ("classifier",
c)])
            score = cross_val_score(pipeline1, df['Message'],
df['Category'], scoring='accuracy', cv=3).mean()
            print('Векторизация - {}'.format(v))
            print('Модель для классификации - {}'.format(c))
```

```
            print('Accuracy = {}'.format(score))
            print('============================')

vectorizers_list = [TfidfVectorizer()]
classifiers_list = [LogisticRegression(C=3.0), KNeighborsClassifier()]
VectorizeAndClassify(vectorizers_list, classifiers_list)
```

```
Векторизация - TfidfVectorizer()
Модель для классификации - LogisticRegression(C=3.0)
Accuracy = 0.9784643255796249
============================
Векторизация - TfidfVectorizer()
Модель для классификации - KNeighborsClassifier()
Accuracy = 0.9271391194481494
============================
```

```
X_train, X_test, y_train, y_test = train_test_split(df['Message'],
df['Category'], test_size=0.7, random_state=1)

def sentiment(v, c):
    model = Pipeline(
        [("vectorizer", v),
         ("classifier", c)])
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print_accuracy_score_for_classes(y_test, y_pred)

sentiment(TfidfVectorizer(), LogisticRegression(C=5.0))
```

```
Метка      Accuracy
ham   0.9958555358200119
spam  0.8164435946462715
```

```
sentiment(TfidfVectorizer(ngram_range=(1,3)),
LogisticRegression(C=5.0))
```

```
Метка      Accuracy
ham   0.9976317347542925
spam  0.8164435946462715
```

```
sentiment(TfidfVectorizer(ngram_range=(2,3)),
LogisticRegression(C=5.0))
```

```
Метка      Accuracy
ham   0.9997039668442865
spam  0.40535372848948376
```

```
sentiment(TfidfVectorizer(ngram_range=(1,4)),
LogisticRegression(C=5.0))
```

```
Метка      Accuracy
ham   0.9979277679100059
spam  0.8068833652007649
```

```
sentiment(TfidfVectorizer(ngram_range=(2,4)),
LogisticRegression(C=5.0))
```

```
Метка       Accuracy
ham    0.9997039668442865
spam   0.35564053537284895
```

```
import re
import pandas as pd
import numpy as np
from typing import Dict, Tuple
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.feature_extraction.text import CountVectorizer,
TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from nltk import WordPunctTokenizer
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')
```

```
from gensim.test.utils import datapath
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
model =
gensim.models.KeyedVectors.load_word2vec_format(datapath('word2vec_pre
_kv_c'), binary=False)
```

```
class EmbeddingVectorizer(object):
    '''
    Для текста усредним вектора входящих в него слов
    '''
    def __init__(self, model):
        self.model = model
        self.size = model.vector_size

    def fit(self, X, y):
        return self

    def transform(self, X):
        return np.array([np.mean(
            [self.model[w] for w in words if w in self.model]
            or [np.zeros(self.size)], axis=0)
            for words in X])

corpus = []
stop_words = stopwords.words('english')
tok = WordPunctTokenizer()
for line in df['Message'].values:
```

```
    line1 = line.strip().lower()
    line1 = re.sub("[^a-zA-Z]"," ", line1)
    text_tok = tok.tokenize(line1)
    text_tok1 = [w for w in text_tok if not w in stop_words]
    corpus.append(text_tok1)

model_imdb = word2vec.Word2Vec(corpus, workers=4, min_count=10,
window=10, sample=1e-3)

def sentiment(v, c):
    model = Pipeline(
        [("vectorizer", v),
         ("classifier", c)])
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(classification_report(y_test, y_pred, digits=4,
target_names=list(map(str, list(np.unique(y_test))))))

boundary = 700
X_train = corpus[:boundary]
X_test = corpus[boundary:]
y_train = df['Category'].values[:boundary]
y_test = df['Category'].values[boundary:]

sentiment(EmbeddingVectorizer(model_imdb.wv), LogisticRegression())
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/
_classification.py:1318: UndefinedMetricWarning: Precision and F-score
are ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

	precision	recall	f1-score	support
ham	0.8672	1.0000	0.9289	4225
spam	0.0000	0.0000	0.0000	647
accuracy			0.8672	4872
macro avg	0.4336	0.5000	0.4644	4872
weighted avg	0.7520	0.8672	0.8055	4872

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/
_classification.py:1318: UndefinedMetricWarning: Precision and F-score
are ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification
.py:1318: UndefinedMetricWarning: Precision and F-score are ill-
defined and being set to 0.0 in labels with no predicted samples. Use
```

```
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```