Отчёт по лабораторной работе №5 студентки группы ИУ5-21М Дьяконовой Светланы

```
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")

from spacy.lang.ru import Russian
from spacy.lang.en import English
import spacy
from spacy import displacy

import gensim
from gensim.models import word2vec
from gensim.models import Word2Vec

def accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray) -> Dict[int, float]:
    """
    Вычисление метрики accuracy для каждого класса
    y_true - истинные значения классов
    y_pred - предсказанные значения классов
    Возвращает словарь: ключ - метка класса,
```

```
    значение - Accuracy для данного класса
    """
    # Для удобства фильтрации сформируем Pandas DataFrame
    d = {'t': y_true, 'p': y_pred}
    df = pd.DataFrame(data=d)
    # Метки классов
    classes = np.unique(y_true)
    # Результирующий словарь
    res = dict()
    # Перебор меток классов
    for c in classes:
        # отфильтруем данные, которые соответствуют
        # текущей метке класса в истинных значениях
        temp_data_flt = df[df['t']==c]
        # расчет accuracy для заданной метки класса
        temp_acc = accuracy_score(
            temp_data_flt['t'].values,
            temp_data_flt['p'].values)
        # сохранение результата в словарь
        res[c] = temp_acc
    return res
```

```
df = pd.read_csv('train.csv', usecols=['Description', 'Class Index'],
nrows=10000)
df
```

```
      Class Index                                        Description
0               3  Reuters - Short-sellers, Wall Street's dwindli...
1               3  Reuters - Private investment firm Carlyle Grou...
2               3  Reuters - Soaring crude prices plus worries\ab...
3               3  Reuters - Authorities have halted oil export\f...
4               3  AFP - Tearaway world oil prices, toppling reco...
...           ...                                                ...
9995            4  Users of the music player should watch out for...
9996            4  BMC Software has released a new version of Pat...
9997            3  The chief of Beijing-backed China Aviation Oil...
9998            3  BRUSSELS The European Commission has opened an...
9999            4  Operation Digital Gridlock targets peer-to-pee...

[10000 rows x 2 columns]
```

1

```
!python -m spacy download 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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: 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)
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)
✔ Download and installation successful
You can now load the model via spacy.load('en_core_web_sm')
```

```
# Tokenize
nlp = spacy.load('en_core_web_sm')
spacy_text1 = nlp(df['Description'][0])
spacy_text1
```

```
Reuters - Short-sellers, Wall Street's dwindling\band of ultra-cynics,
are seeing green again.
```

```
# POS-tagging
for token in spacy_text1:
    print('{} - {} - {}'.format(token.text, token.pos_, token.dep_))
```

```
Reuters - PROPN - compound
- - PUNCT - punct
Short - PROPN - compound
- - PUNCT - punct
sellers - NOUN - nsubj
, - PUNCT - punct
Wall - PROPN - compound
Street - PROPN - poss
's - PART - case
dwindling\band - NOUN - appos
of - ADP - prep
ultra - ADJ - dep
- - NOUN - dep
cynics - NOUN - pobj
, - PUNCT - punct
are - AUX - aux
seeing - VERB - ROOT
green - ADJ - dobj
again - ADV - advmod
. - PUNCT - punct
```

```
# lemmatization
for token in spacy_text1:
      print(token, token.lemma, token.lemma_)
```

```
Reuters 17690189795809227049 Reuters
- 9153284864653046197 -
Short 7602511986449097345 Short
- 9153284864653046197 -
sellers 15370787710306132986 seller
, 2593208677638477497 ,
Wall 8806908179280924345 Wall
Street 11849903144683346075 Street
's 16428057658620181782 's
dwindling\band 16257729132875505059 dwindling\band
of 886050111519832510 of
ultra 887117815877094537 6 ultra
- 9153284864653046197 -
cynics 14078618470999779890 cynic
, 2593208677638477497 ,
are 10382539506755952630 be
seeing 11925638236994514241 see
green 3487151913243070096 green
again 4502205900248518970 again
. 12646065887601541794 .
```

```
# NER
displacy.render(spacy_text1, style='ent', jupyter=True)
```

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

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

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

2

TFIDF

```
tfidfv = TfidfVectorizer(ngram_range=(1,1))
tfidf_ngram_features = tfidfv.fit_transform(df['Description'])
tfidf_ngram_features
```

```
<10000x20257 sparse matrix of type '<class 'numpy.float64'>'
     with 283180 stored elements in Compressed Sparse Row format>
```

```
X_train, X_test, y_train, y_test =
train_test_split(tfidf_ngram_features, df['Class Index'],
test_size=0.3, random_state=1)
```

```
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

```
/Users/danilafedyukin/opt/anaconda3/lib/python3.8/site-packages/
sklearn/linear_model/_logistic.py:763: ConvergenceWarning: lbfgs
failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
print(classification_report(y_test, y_pred, digits=4,
target_names=list(map(str, list(y_test.unique())))))
```

```
              precision    recall  f1-score   support

           1     0.8806    0.8259    0.8524       741
           4     0.8941    0.9299    0.9116       699
           3     0.8487    0.8178    0.8330       741
           2     0.8275    0.8730    0.8497       819

    accuracy                         0.8610      3000
   macro avg     0.8627    0.8617    0.8617      3000
weighted avg     0.8614    0.8610    0.8607      3000
```

word2vec

```
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
[nltk_data]     /Users/danilafedyukin/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['Description'].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['Class Index'].values[:boundary]
y_test = df['Class Index'].values[boundary:]

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

```
/Users/danilafedyukin/opt/anaconda3/lib/python3.8/site-packages/
sklearn/linear_model/_logistic.py:763: ConvergenceWarning: lbfgs
failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(

              precision    recall  f1-score   support

           1     0.8913    0.5718    0.6967      2410
           2     0.8888    0.7039    0.7856      2236
           3     0.8059    0.3318    0.4701      2378
           4     0.4275    0.9398    0.5876      2276

    accuracy                         0.6323      9300
   macro avg     0.7534    0.6368    0.6350      9300
weighted avg     0.7554    0.6323    0.6334      9300
```