Рубежный контроль №2

Писарчук Надежда ИУ5-22М

**Тема: Методы обработки текстов.

Решение задачи классификации текстов.

Необходимо решить задачу классификации текстов на основе любого выбранного Вами датасета (кроме примера, который рассматривался в лекции). Классификация может быть бинарной или многоклассовой. Целевой признак из выбранного Вами датасета может иметь любой физический смысл, примером является задача анализа тональности текста.

Необходимо сформировать два варианта векторизации признаков - на основе CountVectorizer и на основе TfidfVectorizer.

В качестве классификаторов необходимо использовать два классификатора по варианту для Вашей группы: RandomForestClassifier, Complement Naive Bayes (CNB)

Для каждого метода необходимо оценить качество классификации. Сделайте вывод о том, какой вариант векторизации признаков в паре с каким классификатором показал лучшее качество.

B [22]:
```
# This Python 3 environment comes with many helpful analytics librarie
# It is defined by the kaggle/python Docker image: https://github.com/
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

from typing import Dict, Tuple
from scipy import stats
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer, TfidfVect
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifie
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, c
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import mean_absolute_error, mean_squared_error, m
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR, NuSVR
from sklearn.naive_bayes import ComplementNB
import seaborn as sns
```

```
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")

# Input data files are available in the read-only "../input/" director
# For example, running this (by clicking run or pressing Shift+Enter)

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/)
# You can also write temporary files to /kaggle/temp/, but they won't

pd.set_option("display.max_columns", None)
```

```
/kaggle/input/covid-19-nlp-text-classification/Corona_NLP_test.csv
/kaggle/input/covid-19-nlp-text-classification/Corona_NLP_train.csv
```

B [2]:

```
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

def print_accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray):
    """
    Вывод метрики accuracy для каждого класса
    """
```

```
    accs = accuracy_score_for_classes(y_true, y_pred)
    if len(accs)>0:
        print('Метка \t Accuracy')
    for i in accs:
        print('{} \t {}'.format(i, accs[i]))
```

B [3]:
```
train = pd.read_csv('/kaggle/input/covid-19-nlp-text-classification/Co
test = pd.read_csv('/kaggle/input/covid-19-nlp-text-classification/Cor
```

B [4]:
```
print(train.shape)
print(test.shape)
```

```
(41157, 6)
(3798, 6)
```

B [5]:
```
train.head()
```

Out[5]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
4	3803	48755	NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative

B [6]:
```
train.Sentiment.value_counts()
```

Out[6]:
```
Positive              11422
Negative               9917
Neutral                7713
Extremely Positive     6624
Extremely Negative     5481
Name: Sentiment, dtype: int64
```

B [7]:
```
train.Sentiment = train.Sentiment.replace({'Extremely Positive':'Posit
test.Sentiment = test.Sentiment.replace({'Extremely Positive':'Positiv

lenc = LabelEncoder()
test.Sentiment = lenc.fit_transform(test.Sentiment)
train.Sentiment = lenc.fit_transform(train.Sentiment)
```

B [8]:
```
train.head()
```

Out[8]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	1

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	2
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	2
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	2

B [9]:
```
x_train = train['OriginalTweet']
y_train = train['Sentiment']
x_test = test['OriginalTweet']
y_test = test['Sentiment']
```

Очистка данных

B [10]:
```
import re
def preprocess_sentence(w):
    # отделение слов и знаков пунктуации пробелом
    # eg: "he is a boy." => "he is a boy ."
    w = re.sub('\t\n', '', w)
    w = re.sub(r'http\S+', '', w)
    w = re.sub(r"([?.!,])", r" \1 ", w)
    w = re.sub(r'[" "]+', " ", w)


    # удаляем все кроме (a-z, A-Z, ".", "?", "!", ",")
    w = re.sub(r"[^a-zA-Za-яA-Я?.!,`']+", " ", w)

    w = w.strip()

    return w
```

B [11]:
```
x_train = x_train.apply(preprocess_sentence)
x_test = x_test.apply(preprocess_sentence)
```

B [12]:
```
# Сформируем общий словарь для обучения моделей из обучающей и тестово
vocab_list = x_train.tolist() + x_test.tolist()
print(len(vocab_list))
vocab_list[1:10]
```

```
44955
```

Out[12]:

```
['advice Talk to your neighbours family to exchange phone numbers cre
ate contact list with phone numbers of neighbours schools employer ch
emist GP set up online shopping accounts if poss adequate supplies of
regular meds but not over order',
 'Coronavirus Australia Woolworths to give elderly , disabled dedicat
ed shopping hours amid COVID outbreak',
 "My food stock is not the only one which is empty . . . PLEASE , don
't panic , THERE WILL BE ENOUGH FOOD FOR EVERYONE if you do not take
more than you need . Stay calm , stay safe . COVID france COVID COVID
coronavirus confinement Confinementotal ConfinementGeneral",
 "Me , ready to go at supermarket during the COVID outbreak . Not bec
ause I'm paranoid , but because my food stock is litteraly empty . Th
e coronavirus is a serious thing , but please , don't panic . It caus
es shortage . . . CoronavirusFrance restezchezvous StayAtHome confine
ment",
 'As news of the region s first confirmed COVID case came out of Sull
ivan County last week , people flocked to area stores to purchase cle
aning supplies , hand sanitizer , food , toilet paper and other goods
, Tim Dodson reports',
 "Cashier at grocery store was sharing his insights on Covid To prove
his credibility he commented I'm in Civics class so I know what I'm t
alking about .",
 "Was at the supermarket today . Didn't buy toilet paper . Rebel toil
```

B [13]:
```
train['OriginalTweet'][0:10][4]
```

Out[13]:
```
"Me, ready to go at supermarket during the #COVID19 outbreak.\r\r\n\
r\r\nNot because I'm paranoid, but because my food stock is litteraly
empty. The #coronavirus is a serious thing, but please, don't panic.
It causes shortage...\r\r\n\r\r\n#CoronavirusFrance #restezchezvous #
StayAtHome #confinement https://t.co/usmuaLq72n" (https://t.co/usmuaL
q72n")
```

B [14]:
```
vocabVect = CountVectorizer()
vocabVect.fit(vocab_list)
corpusVocab = vocabVect.vocabulary_
print('Количество сформированных признаков - {}'.format(len(corpusVoca
```

```
Количество сформированных признаков - 54625
```

B [15]:
```
for i in list(corpusVocab)[1:10]:
    print('{}={}'.format(i, corpusVocab[i]))
```

```
phil=36328
gahan=18915
chrisitv=8237
and=1841
advice=672
talk=47454
to=49038
your=54300
neighbours=32617
```

B [16]:
```
tfidfv = TfidfVectorizer(ngram_range=(1,3))
tfidf_ngram_features = tfidfv.fit_transform(vocab_list)
tfidf_ngram_features
```

Out[16]:
```
<44955x1381637 sparse matrix of type '<class 'numpy.float64'>'
        with 3613058 stored elements in Compressed Sparse Row format>
```

Type *Markdown* and LaTeX: α^2

B [23]:
```
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, x_train[:10000], y_trai
            print('Векторизация - {}'.format(v))
            print('Модель для классификации - {}'.format(c))
            print('Accuracy = {}'.format(score))
            print('============================')
```

B [24]:
```
vectorizers_list = [CountVectorizer(vocabulary = corpusVocab), TfidfVe
classifiers_list = [RandomForestClassifier(), ComplementNB()]
VectorizeAndClassify(vectorizers_list, classifiers_list)
```

```
Векторизация - CountVectorizer(vocabulary={'aa': 0, 'aaa': 1, 'aaaaak
ubosan': 2, 'aaaaas': 3,
                               'aaaand': 4, 'aaachatterjee': 5, 'aaanews
': 6,
                               'aaannnddd': 7, 'aaanortheast': 8, 'aabut
an': 9,
                               'aacopd': 10, 'aacounty': 11, 'aacountygo
vt': 12,
                               'aadeshrawal': 13, 'aadya': 14, 'aadyasit
ara': 15,
                               'aafp': 16, 'aahealth': 17, 'aahh': 18, '
aai': 19,
                               'aaisp': 20, 'aajeevika': 21, 'aajtak': 2
2,
                               'aakash': 23, 'aalonzowatt': 24, 'aalto':
25,
                               'aaltouniversity': 26, 'aalwajih': 27,
                               'aamaadmi': 28, 'aamaadmiparty': 29,
...})
Модель для классификации - RandomForestClassifier()
Accuracy = 0.6666002833036754
============================
Векторизация - CountVectorizer(vocabulary={'aa': 0, 'aaa': 1, 'aaaaak
ubosan': 2, 'aaaaas': 3,
                               'aaaand': 4, 'aaachatterjee': 5, 'aaanews
': 6,
                               'aaannnddd': 7, 'aaanortheast': 8, 'aabut
an': 9,
                               'aacopd': 10, 'aacounty': 11, 'aacountygo
vt': 12,
                               'aadeshrawal': 13, 'aadya': 14, 'aadyasit
ara': 15,
                               'aafp': 16, 'aahealth': 17, 'aahh': 18, '
aai': 19,
                               'aaisp': 20, 'aajeevika': 21, 'aajtak': 2
2,
                               'aakash': 23, 'aalonzowatt': 24, 'aalto':
25,
                               'aaltouniversity': 26, 'aalwajih': 27,
                               'aamaadmi': 28, 'aamaadmiparty': 29,
...})
Модель для классификации - ComplementNB()
Accuracy = 0.6480996030016919
============================
Векторизация - TfidfVectorizer(vocabulary={'aa': 0, 'aaa': 1, 'aaaaak
ubosan': 2, 'aaaaas': 3,
                               'aaaand': 4, 'aaachatterjee': 5, 'aaanews
': 6,
                               'aaannnddd': 7, 'aaanortheast': 8, 'aabut
an': 9,
                               'aacopd': 10, 'aacounty': 11, 'aacountygo
vt': 12,
                               'aadeshrawal': 13, 'aadya': 14, 'aadyasit
ara': 15,
                               'aafp': 16, 'aahealth': 17, 'aahh': 18, '
aai': 19,
```

Лучший результат покаазала модель RandomForestClassifier с CountVectorizer

B [ ]: