Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Автоматизированные системы обработки информации и управления»

Отчет по рубежному контролю № 2

«Методы построения моделей машинного обучения»

По курсу
«Методы Машинного Обучения»

Выполнила:
Шаххуд Ф.М.
Студентка группы ИУ5И-22М

Москва, 2020

Цель работы

Классификация текстов на арабском языке с использованием разных видов классификато
классификаторов.

Набор данных

Набор данных представляет собой набор арабских текстов, который охватывает современ
газетных статьях.

Набор данных состоит из 111 728 документов и 319 254 124 слов, структурированных в тек
онлайн-газет: Assabah [9], Hespress [10] и Akhbarona [11] с использованием полуавтоматичес
Интернете.

Документы в наборе данных подразделяются на 5 классов: спортивные, политические, кул
разнообразные.

```
import numpy as np
import pandas as pd
from typing import Dict, Tuple
from scipy import stats
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score, classification
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_squared_l
from sklearn.naive_bayes import GaussianNB, MultinomialNB, ComplementNB, BernoulliN
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR, NuSVR, LinearSVR
from sklearn.metrics import confusion_matrix
from sklearn.metrics import plot_confusion_matrix
import warnings
warnings.filterwarnings("ignore")
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: Futur
  import pandas.util.testing as tm
```

Глядя на первые пять строк нашего набора данных:

```
data = pd.read_csv("/content/drive/My Drive/arabic_dataset_classifiction.csv",names
data.head()
```

	text	target
0	text	targe
1	...بين أستوديوهات ورزازات وصحراء مرزوكة وآثار ولي	0
2	...قررت النجمة الأمريكية أوبرا وينفري ألا يقتصر ع	0
3	...أخبارنا المغربية الوزاني تصوير الشملالي ألهب ا	0

```
def accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray) -> Dict[int,Tuple[float,float]]:

    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)
        temp_f1=f1_score(temp_data_flt['t'].values,
            temp_data_flt['p'].values, average='micro')
        # сохранение результата в словарь
        res[c] = (temp_acc,temp_f1)
    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 \t \t f1_score')
    for i in accs:
        print('{} \t {} \t {}'.format(i, accs[i][0],accs[i][1]))
```

Количество строки в нашем наборе данных:

```
data.shape
```

```
(111729, 2)
```

```
data.text.shape
```

```
(108790,)
```

```
data.target.shape
```

```
(108790,)
```

Итак, мы увидели, что в нашем наборе данных есть несколько строк с нулевым значением. Чтобы справиться с этим, мы собираемся включить все строки с нулевым значением и ост содержат данные.

```
data=data.dropna()
```

Наша новая форма данных:

```
data.shape
```

```
(108790, 2)
```

```
vocab_list = data['text'].tolist()
vocab_list[1:10]
```

```
['ر أيضا عن تراجع كبير في ميدان العلم والمعرفة لدى المغاربة والمسلمين بصفة عامة'
 'ع بعائدات بيع هذا الشاي لأكاديمية أسستها وينفري وتعنى بتوفير فرص تعليم للشبان'
 'ر أسماء فنية عربية مغربية نذكر منها وائل جسار فارس كرم سعيدة شرف سعيد موسكير'
 'مرة التي تعرض لها موكله مبرزا انه رجل قانون لا يخوض في امور لها علاقة بالسياسة'
 'ئحة الكريهة ويحرصون على شراء النعناع قبل الوصول إليها لغرض تخفيف شدة الرائحة'
 'ي المسرحية التي تعالج قضايا المرأة وقضايا اجتماعية عدة في قالب فني ساخر وجميل'
 'ي بعينك لو بس تعرف دلل بياع اليانصيب التي ربما سيكون لجمهور تيميتار نصيب منها'
 'ئلتها وأصدقائها وجميع الذين أحبوها لقد حل علينا هذا الخبر جميعا كوقع الصاعقة'
 'رسالة أخيرة له قائلة استر ده ربنا بيستر كما أعربت عن تفاؤلها وثقتها في القضاء'
```

Количество уникальных слов в наборе данных.

```
vocabVect = CountVectorizer()
vocabVect.fit(vocab_list)
corpusVocab = vocabVect.vocabulary_
print('Количество сформированных признаков - {}'.format(len(corpusVocab)))
```

```
Количество сформированных признаков - 423177
```

Примеры некоторых слов, взятых из набора данных с их частотами.

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

```
بين=136351
أستوديوهات=6605
ورزازات=367728
وصحراء=374884
مرزوكة=289711
وآثار=315610
وليلي=387452
ثم=158196
الرباط=55944
```

```
test_features = vocabVect.transform(vocab_list)
```

```
test_features
```

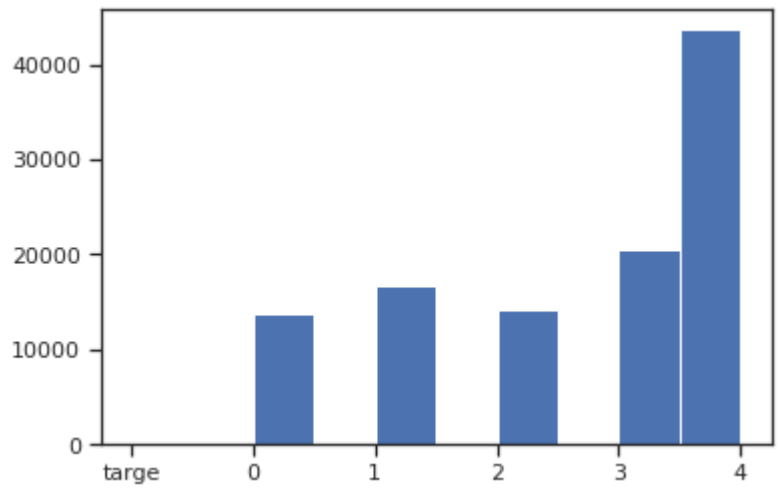
```
<108790x423177 sparse matrix of type '<class 'numpy.int64'>'
        with 20016309 stored elements in Compressed Sparse Row format>
```

Подгонка и прогнозирование данных обучения и испыта нескольких векторизаторов и классификаторов

Мы собираемся разделить наши данные между данными обучения и тестирования в пропо

```
X_train, X_test, y_train, y_test = train_test_split(data['text'], data['target'], t
```

```
plt.hist(data['target'])
plt.show()
```



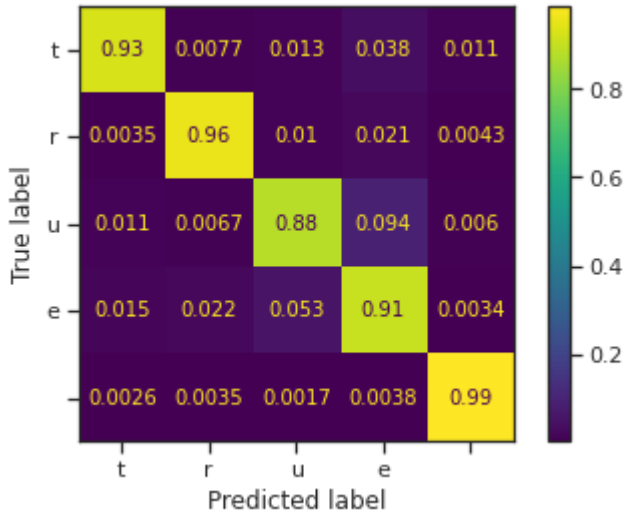

```
def sentiment(v, c):
    model = Pipeline(
        [("vectorizer", v),
         ("classifier", c)])
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print_accuracy_score_for_classes(y_test, y_pred)
    labels=['0', '1', '2', '3',  '4']
    plot_confusion_matrix(model, X_test, y_test, labels,display_labels='true', normali
```

```
plot_confusion_matrix(model,X_test, y_test,labels,display_labels= true ,normali
```

Использование счетчика векторов и логистической регрессии

```
sentiment(CountVectorizer(), LogisticRegression(C=3.0))
```

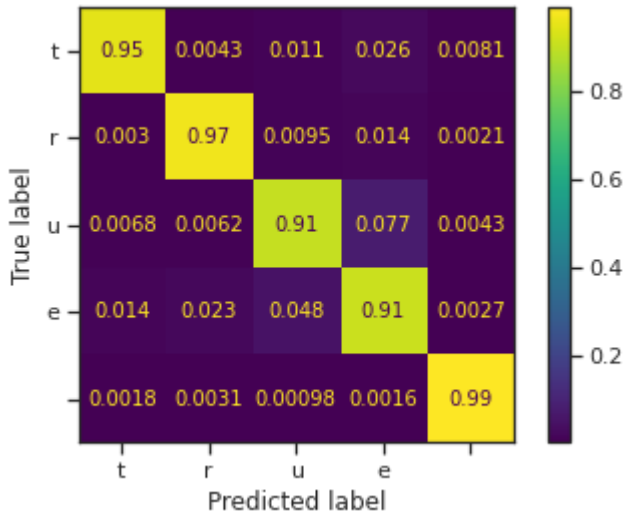
Метка	Accuracy	f1_score
0	0.9303995006242197	0.9303995006242197
1	0.9606473594548552	0.9606473594548552
2	0.8823828402968384	0.8823828402968384
3	0.9074203465623253	0.9074203465623253
4	0.9885162469006916	0.9885162469006916



Использование Tfid Vectorizer и LinearSVC

```
sentiment(TfidfVectorizer(), LinearSVC())
```

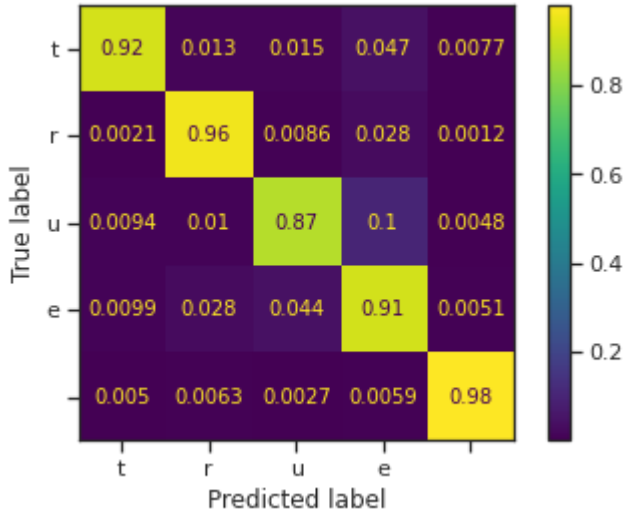
Метка	Accuracy	f1_score
0	0.9507906783187682	0.9507906783187683
1	0.971465076660988	0.971465076660988
2	0.9052556673782657	0.9052556673782657
3	0.912800447177194	0.912800447177194
4	0.9925942842228892	0.9925942842228892



Использование графа векторизатора и полиномиального наивн

```
sentiment(CountVectorizer(), MultinomialNB())
```

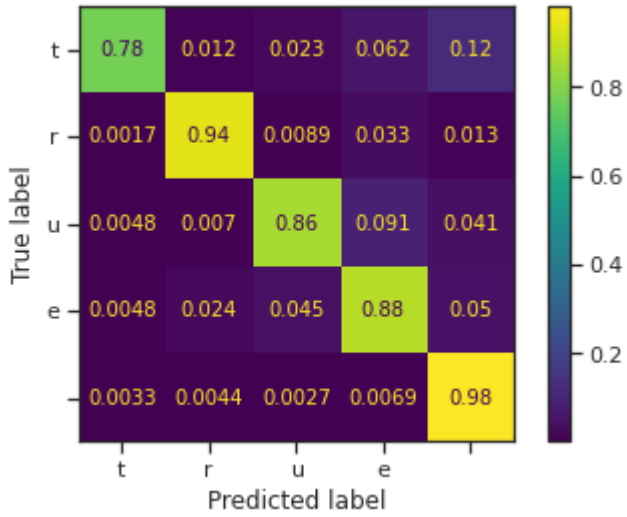
Метка	Accuracy	f1_score
0	0.9171868497711194	0.9171868497711194
1	0.9597103918228279	0.9597103918228279
2	0.8727254244180136	0.8727254244180136
3	0.9132196757965344	0.9132196757965344
4	0.9801318021662534	0.9801318021662534



Использование графа векторизатора и наивного байесовского

```
sentiment(CountVectorizer(binary=True), BernoulliNB())
```

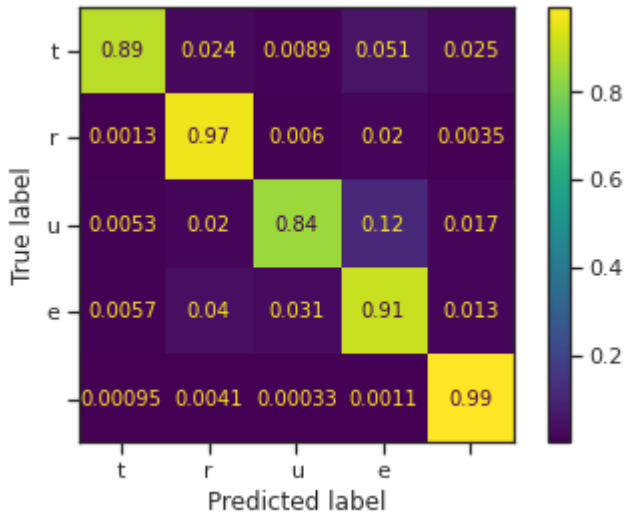
Метка	Accuracy	f1_score
0	0.7796504369538078	0.7796504369538076
1	0.9436115843270869	0.9436115843270869
2	0.8561553319101352	0.8561553319101352
3	0.8760480715483511	0.8760480715483511
4	0.9827743703510374	0.9827743703510374



Использование Tfid Vectorizer и дополнения наивного байесовс

```
sentiment(TfidfVectorizer(), ComplementNB())
```

Метка	Accuracy	f1_score
0	0.8905534748231377	0.8905534748231377
1	0.9695059625212947	0.9695059625212947
2	0.8429399207075328	0.8429399207075328
3	0.9107741755170486	0.9107741755170486
4	0.9934751402844839	0.9934751402844839



Вывод

Мы видим, что лучшим классификатором при работе с данными был LinearSVC, поскольку с данных с точностью более 90 процентов. Это было сделано с использованием Tfid Vectorize Frequency (TF) и Inverse Document Frequency (IDF). в то время как LinearSVC возвращает «наи которая разделяет или классифицирует данные.

Список литературы

1. Набор данных был взят с сайта: https://data.mendeley.com/datasets/v524p5dhpj/2
2. Лекции по курсу «Методы машинного обучения»