Рубежный контроль №2

Демьянчук Григорий Валентинович ИУ5-22М Вариант 1

Задание

Необходимо решить задачу классификации текстов на основе любого выбранного Вами дата
быть бинарной или многоклассовой. Целевой признак из выбранного Вами датасета может и
смысл, примером является задача анализа тональности текста.

Необходимо сформировать признаки на основе `CountVectorizer` или `TfidfVectorizer`.

В качестве классификаторов необходимо использовать один из классификаторов, не относящ
Байесовским методам (например, `LogisticRegression`), а также Multinomial Naive Bayes (MNB)
(CNB), Bernoulli Naive Bayes. Для каждого метода необходимо оценить качество классификаци
метрики качества классификации (например, `accuracy`).

Сделайте выводы о том, какой классификатор осуществляет более качественную классифика
данных.

Решение

Загрузка и предобработка данных

```
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd
import numpy as np
```

```
#rcv_train = fetch_rcv1(subset='train')
#rcv_test = fetch_rcv1(subset='test')
df = pd.read_csv('/content/datasets_2050_3494_SPAM text message 20170820 - Data (1).csv')
```

```
df.head()
```

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...

```
message = np.array(df['Message'])
category = np.array(df['Category'])
# build train and test datasets

# Train/test splitting for 41 categories of news
from sklearn.model_selection import train_test_split
message_train, message_test, category_train, category_test = train_test_split(message, categc


from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

## Build Bag-Of-Words on train phrases
cv = CountVectorizer(stop_words='english',max_features=10000)
cv_train_features = cv.fit_transform(message_train)


# build TFIDF features on train reviews
tv = TfidfVectorizer(min_df=0.0, max_df=1.0, ngram_range=(1,2),
                     sublinear_tf=True)
tv_train_features = tv.fit_transform(message_train)


cv_test_features = cv.transform(message_test)
tv_test_features = tv.transform(message_test)
```

Обучение моделей

```
from sklearn.metrics import accuracy_score


from sklearn import metrics
import numpy as np

def accuracy(classifier,
                        train_features, train_labels,
                        test_features, test_labels):
    classifier.fit(train_features, train_labels)
    print('Accuracy:', metrics.accuracy_score(test_labels, classifier.predict(test_features))



from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB


lr = LogisticRegression(solver='lbfgs',penalty='l2', max_iter=100, C=1,multi_class='auto')
```

```
lr_accuracy = accuracy(classifier=lr,train_features=cv_train_features, train_labels=category_
                       test_features=cv_test_features, test_labels=category_test)
```

```
Accuracy: 0.9721973094170404
```

```
mu = MultinomialNB()

mu_accuracy = accuracy(classifier=mu,train_features=cv_train_features, train_labels=category_
                       test_features=cv_test_features, test_labels=category_test)
```

```
Accuracy: 0.9838565022421525
```

```
co = ComplementNB()

co_accuracy = accuracy(classifier=co,train_features=cv_train_features, train_labels=category_
                       test_features=cv_test_features, test_labels=category_test)
```

```
Accuracy: 0.9766816143497757
```

```
be = BernoulliNB()

be_accuracy = accuracy(classifier=be,train_features=cv_train_features, train_labels=category_
                       test_features=cv_test_features, test_labels=category_test)
```

```
Accuracy: 0.967713004484305
```

Вывод

Метод Multinomial Naive Bayes (MNB), лучше всего решает поставленную задачу бинарной кла