Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Дисциплина «Технологии машинного обучения»

Отчёт

по рубежному контролю №2

Тема: «Технологии использования и оценки моделей машинного обучения.»

Вариант 12

Студент:

Крюков Г. М.

Группа ИУ5-61Б

Преподаватель:

Гапанюк Ю.Е.

Москва, 2020 г.

Задание

Задание №1

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 признаки на основе CountVectorizer или TfidfVectorizer.

В качестве классификаторов необходимо использовать два классификатора, не относящихся к наивным Байесовским методам (например, LogisticRegression, LinearSVC), а также Multinomial Naive Bayes (MNB), Complement Naive Bayes (CNB), Bernoulli Naive Bayes.

Для каждого метода необходимо оценить качество классификации с помощью хотя бы одной метрики качества классификации (например, Accuracy).

Сделать выводы о том, какой классификатор осуществляет более качественную классификацию на моем наборе данных.

Выполнение задания

Подключим необходимые библиотеки и загрузим набор данных

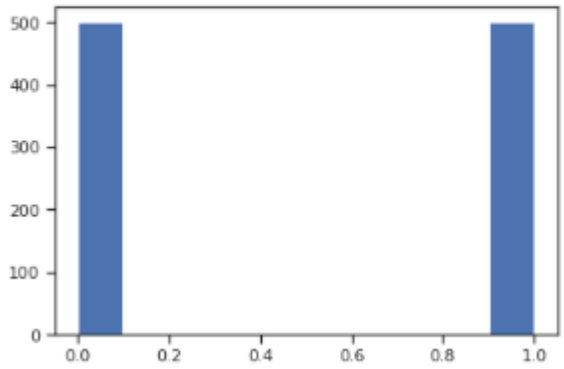
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.svm import SVC
from sklearn.linear_model import SGDClassifier
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB
from sklearn.pipeline import Pipeline
from sklearn.tree import DecisionTreeClassifier
```

```
# Загрузка данных
data = pd.read_csv("/content/yelp_labelled.txt", delimiter='\t', header=None, names=['text', 'value'])
data.head()
```

	text	value
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday of...	1
4	The selection on the menu was great and so wer...	1

```
# В целевом признаке распределение классов равномерное
plt.hist(data['value'])
plt.show()
```




```
vectorizer = TfidfVectorizer(lowercase=True, stop_words=None,
                             use_idf=True, ngram_range=(1,1),
                             smooth_idf=False)
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

```
X_train
```

```
<750x1711 sparse matrix of type '<class 'numpy.float64'>'
        with 7315 stored elements in Compressed Sparse Row format>
```

```
X_test
```

```
<250x1711 sparse matrix of type '<class 'numpy.float64'>'
        with 2120 stored elements in Compressed Sparse Row format>
```

```
def test_model(classifiers_list):
    for c in classifiers_list:
        c.fit(X_train, y_train)
        y_pred = c.predict(X_test)
        print('Модель для классификации - {}'.format(c))
        print('====================================')
        print('accuracy_score:  ', accuracy_score(y_test, y_pred))
        print('precision_score: ', precision_score(y_test, y_pred))
        print('recall_score:    ', recall_score(y_test, y_pred))
        print('f1_score:        ', f1_score(y_test, y_pred))
        print('====================================')
```

```
classifiers_list = [DecisionTreeClassifier(random_state=1),
                    SVC(kernel='rbf'),
                    MultinomialNB(),
                    ComplementNB(),
                    BernoulliNB()]
test_model(classifiers_list)
```

```
Модель для классификации - DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=1, splitter='best')
====================================
accuracy_score:   0.736
precision_score:  0.7016129032258065
recall_score:     0.75
f1_score:         0.725
====================================
Модель для классификации - SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
====================================
accuracy_score:   0.84
precision_score:  0.8333333333333334
recall_score:     0.8189655172413793
f1_score:         0.8260869565217391
====================================
Модель для классификации - MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
====================================
accuracy_score:   0.784
precision_score:  0.75
recall_score:     0.8017241379310345
f1_score:         0.7749999999999999
====================================
Модель для классификации - ComplementNB(alpha=1.0, class_prior=None, fit_prior=True, norm=False)
====================================
accuracy_score:   0.776
precision_score:  0.7586206896551724
recall_score:     0.7586206896551724
f1_score:         0.7586206896551724
====================================
Модель для классификации - BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
====================================
accuracy_score:   0.728
precision_score:  0.6643835616438356
recall_score:     0.8362068965517241
f1_score:         0.7404580152671757
====================================
```

Вывод:

Метод LogisticRegression осуществил более качественную бинарную классификацию на наборе данных.