

РК2 ММО Яковлев Д.С. ИУ5-21М

Задача 1. Классификация текстов на основе методов наивного Байеса.

Задание

- Необходимо решить задачу классификации текстов на основе любого выбранного датасета. Классификация может быть бинарной или многоклассовой. Целевой признак из выбранного 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 двух метрик качества классификации (например, Accuracy, ROC-AUC).
- Сделать выводы о том, какой классификатор осуществляет более качественную классификацию на выбранном наборе данных. ## Выполнение

```
In [1]: 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.metrics import plot_confusion_matrix
from sklearn.metrics import balanced_accuracy_score
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR, NuSVR, LinearSVR
from sklearn.feature_extraction.text import TfidfVectorizer
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

```
In [2]: # Загрузка данных
mail = pd.read_csv("data/SPAM.csv", header=1, names=['category', 'message'])
mail.head()
```

Out[2]:

	category	message
0	ham	Ok lar... Joking wif u oni...
1	spam	Free entry in 2 a wkly comp to win FA Cup fina...
2	ham	U dun say so early hor... U c already then say...
3	ham	Nah I don't think he goes to usf, he lives aro...
4	spam	FreeMsg Hey there darling it's been 3 week's n...

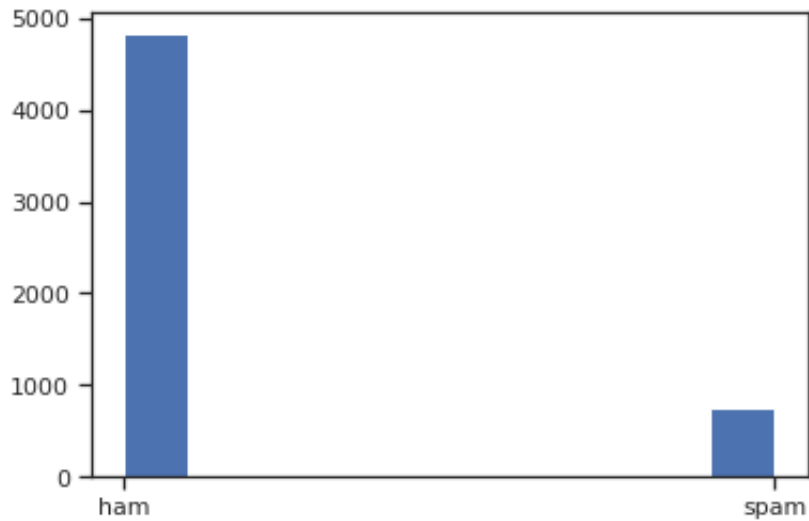
```
In [3]: mail.shape
```

```
Out[3]: (5571, 2)
```

```
In [4]: mail['category'].unique()
```

```
Out[4]: array(['ham', 'spam'], dtype=object)
```

```
In [5]: plt.hist(mail['category'])  
plt.show()
```



В целевом признаке распределение классов не равномерное, поэтому в дальнейшем будем использовать функцию `balanced_accuracy_score` вместо функции `accuracy_score`

```
In [6]: # Сформируем общий словарь для обучения моделей из обучающей и тестовой выборки
vocab_list = mail['message'].tolist()
vocab_list[1:10]
```

```
Out[6]: ["Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 20
05. Text FA to 87121 to receive entry question(std txt rate)T&C's
apply 08452810075over18's",
'U dun say so early hor... U c already then say...',
"Nah I don't think he goes to usf, he lives around here though",
"FreeMsg Hey there darling it's been 3 week's now and no word bac
k! I'd like some fun you up for it still? Tb ok! XxX std chgs to s
end, £1.50 to rcv",
'Even my brother is not like to speak with me. They treat me like
aids patent.',
"As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Ve
ttam)' has been set as your callertune for all Callers. Press *9 t
o copy your friends Callertune",
'WINNER!! As a valued network customer you have been selected to
receive a £900 prize reward! To claim call 09061701461. Claim code
KL341. Valid 12 hours only.',
'Had your mobile 11 months or more? U R entitled to Update to the
latest colour mobiles with camera for Free! Call The Mobile Update
Co FREE on 08002986030',
"I'm gonna be home soon and i don't want to talk about this stuff
anymore tonight, k? I've cried enough today."]
```

```
In [7]: vocabVect = CountVectorizer()
vocabVect.fit(vocab_list)
corpusVocab = vocabVect.vocabulary_
print('Количество сформированных признаков - {}'.format(len(corpusV
ocab)))
```

Количество сформированных признаков - 8707

```
In [8]: tfidfV = TfidfVectorizer(ngram_range=(1,3))
tfidf_ngram_features = tfidfV.fit_transform(vocab_list)
tfidf_ngram_features
```

```
Out[8]: <5571x104900 sparse matrix of type '<class 'numpy.float64'>'
with 217288 stored elements in Compressed Sparse Row forma
t>
```

Будем проверять классификаторы *LinearSVC* и метод *K* соседей.

В качестве наивных Байесовских используем методы *Complement Naive Bayes (CNB)* и *Bernoulli Naive Bayes*.

Предположительно лучшую точность среди Байесовских классификаторов покажет *CNB*, поскольку данный метод подходит для наборов с сильным дисбалансом классов.

Проверим это предположение:

Разделим выборку на обучающую и тестовую.

```
In [11]: X_train, X_test, y_train, y_test = train_test_split(mail['message']  
    , mail['category'], test_size=0.5, random_state=1)
```

Будем использовать метрики качества *balanced_accuracy* и матрицу ошибок.

```

In [12]: 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_dataflt = df[df['t']==c]
        # расчет accuracy для заданной метки класса
        temp_acc = balanced_accuracy_score(
            temp_dataflt['t'].values,
            temp_dataflt['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]))

```

```
In [19]: def sentiment(v, c):
    model = Pipeline(
        [ ("vectorizer", v),
          ("classifier", c)] )
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

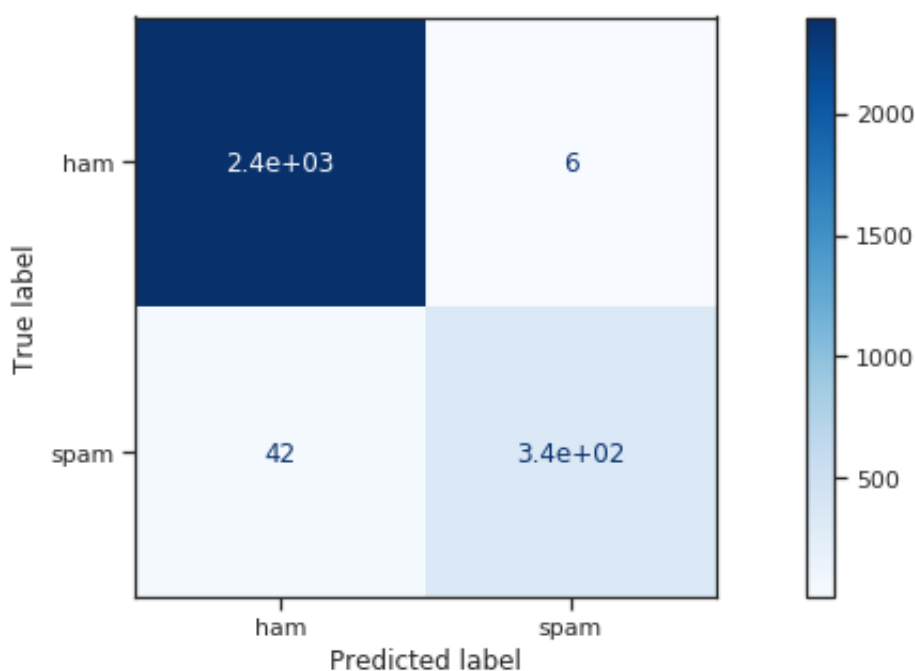
    # Accuracy
    print_accuracy_score_for_classes(y_test, y_pred)

    # Матрица ошибок
    fig, ax = plt.subplots( figsize=(15,5))
    plot_confusion_matrix(model, X_test, y_test, cmap=plt.cm.Blues,
ax=ax)
```

```
In [14]: sentiment(TfidfVectorizer(ngram_range=(1,3)), LinearSVC())
```

```
/home/lisobol/tensorflow_env/my_tensorflow/lib/python3.7/site-pack
ages/sklearn/metrics/_classification.py:1859: UserWarning: y_pred
contains classes not in y_true
  warnings.warn('y_pred contains classes not in y_true')
```

Метка	Accuracy
ham	0.997504159733777
spam	0.8900523560209425



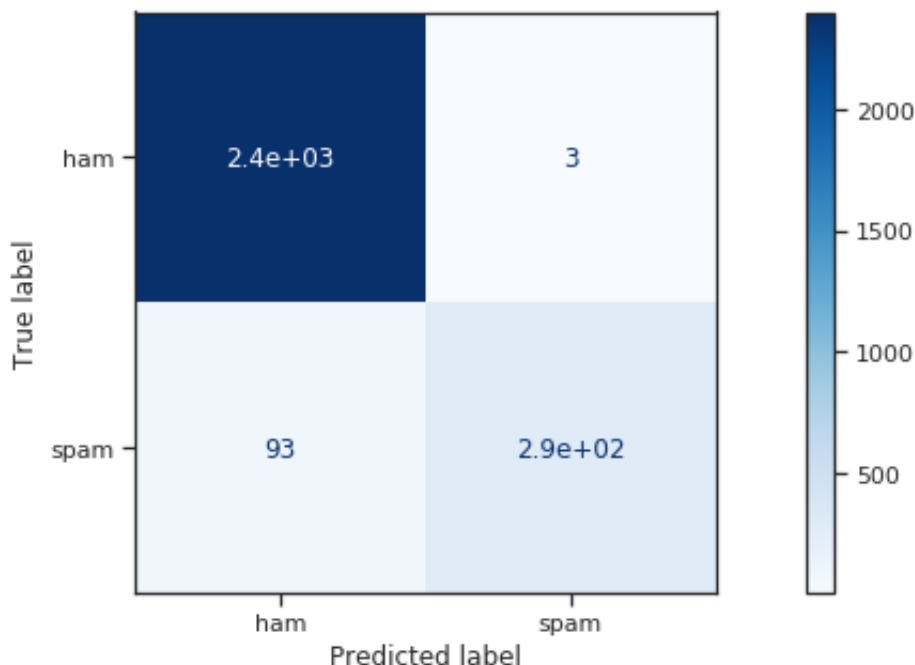
Для метода К соседей найдем в цикле лучшее кол-во соседей

```
In [20]: # for k in range (1, 15):  
#         print(k)  
#         sentiment(TfidfVectorizer(ngram_range=(1,3)), KNeighborsClassifier(n_neighbors=k))
```

```
In [16]: # Лучшее значение - 3 соседа  
sentiment(TfidfVectorizer(ngram_range=(1,3)), KNeighborsClassifier(  
n_neighbors=3))
```

```
/home/lisobol/tensorflow_env/my_tensorflow/lib/python3.7/site-pack  
ages/sklearn/metrics/_classification.py:1859: UserWarning: y_pred  
contains classes not in y_true  
    warnings.warn('y_pred contains classes not in y_true')
```

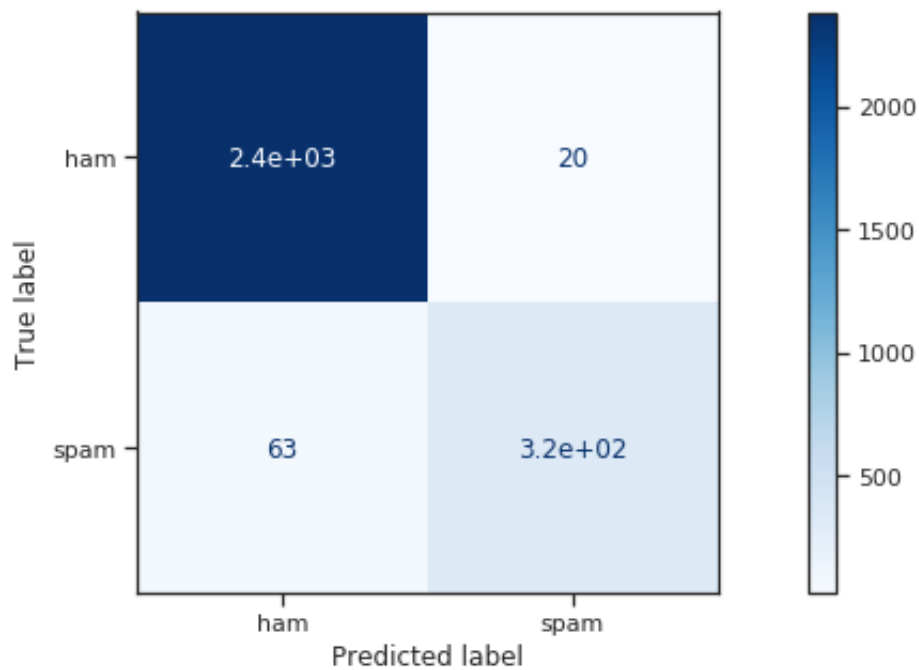
Метка	Accuracy
ham	0.9987520798668885
spam	0.756544502617801




```
In [17]: sentiment(TfidfVectorizer(), ComplementNB())
```

```
/home/lisobol/tensorflow_env/my_tensorflow/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1859: UserWarning: y_pred contains classes not in y_true  
  warnings.warn('y_pred contains classes not in y_true')
```

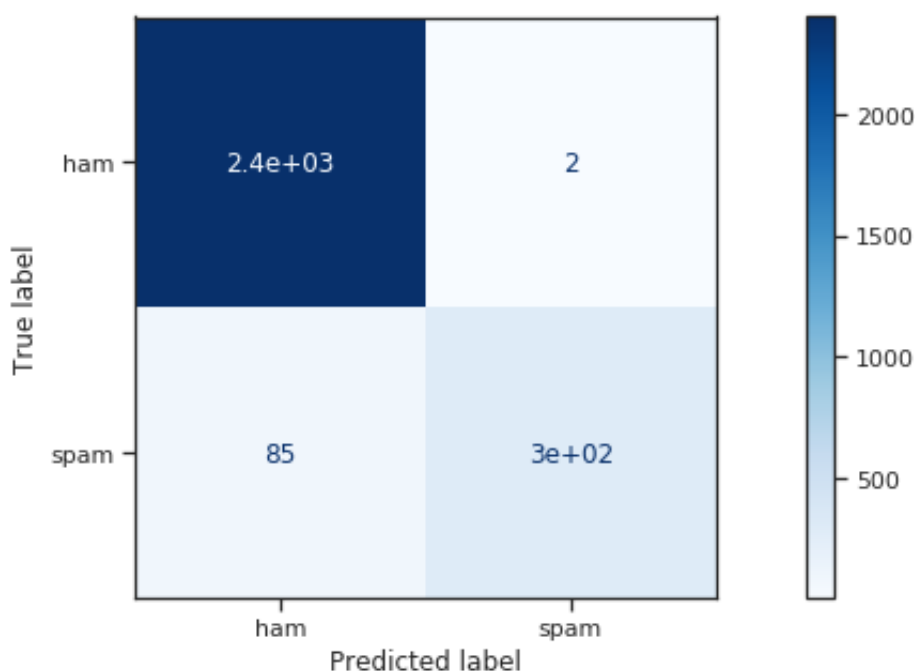
Метка	Accuracy
ham	0.9916805324459235
spam	0.8350785340314136



```
In [18]: sentiment(TfidfVectorizer(), BernoulliNB())
```

```
/home/lisobol/tensorflow_env/my_tensorflow/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1859: UserWarning: y_pred contains classes not in y_true
  warnings.warn('y_pred contains classes not in y_true')
```

Метка	Accuracy
ham	0.9991680532445923
spam	0.7774869109947644



Вывод:

Поскольку выборка несбалансированная и все классификаторы делают незначительное количество ошибок при предсказании класса ham(не спам), то будем смотреть точность, с какой модели предсказывают класс spam. Можно увидеть, что наилучший результат показал классификатор LinearSVR, а худший - метод К соседей с 3 соседями(также был проведен эксперимент, определяющий оптимальное кол-во соседей, но даже при этом этот метод оказался худшим). Так же было подтверждено предположение, что лучшую точность среди Байесовских классификаторов покажет CNB, так как он предназначен для классов с дисбалансом.