МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

ОТЧЕТ

Рубежный контроль №2
по курсу «Технологии машинного обучения»

Вариант 9

ИСПОЛНИТЕЛЬ: ___Меркулова Н.А.__
ФИО
группа ИУ5-64Б _____
подпись
"__"_____2020 г.

ПРЕПОДАВАТЕЛЬ: ___Гапанюк Ю.Е.___
ФИО
_____
подпись
"__"_____2020 г.

Москва - 2020

1. Условие

Задача 1. Классификация текстов на основе методов наивного Байеса.

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).
Сделате выводы о том, какой классификатор осуществляет более качественную классификацию на Вашем наборе данных.

2. Выполнение

См. на следующей странице

In [25]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score, precision_score
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB
from sklearn.metrics import accuracy_score
from sklearn.svm import LinearSVC
from sklearn.feature_extraction.text import TfidfVectorizer

%matplotlib inline
sns.set(style="ticks")
```

In [3]:

```
data = pd.read_csv('../data/south-park.csv')
data
```

Out[3]:

	Season	Episode	Character	Line
0	10	1	Stan	You guys, you guys! Chef is going away. \n
1	10	1	Kyle	Going away? For how long?\n
2	10	1	Stan	Forever.\n
3	10	1	Chef	I'm sorry boys.\n
4	10	1	Stan	Chef said he's been bored, so he joining a gro...
...	...	...	...	...
70891	9	14	Stan	I think you're pushing it.\n
70892	9	14	Randy	How about twenty?\n
70893	9	14	Stan	That's not disciprine.\n
70894	9	14	Randy	Right right. Does vodka count?\n
70895	9	14	Stan	Dad!\n

70896 rows × 4 columns

In [4]:

```
data = data.drop(columns = ['Season', 'Episode'])
```

In [5]:

```
data['Character'].value_counts()
```

Out[5]:

```
Cartman             9774
Stan                7680
Kyle                7099
Butters             2602
Randy               2467
                    ...
Some KKK members       1
Reveler 4              1
Louse 3                1
Paparazzo 10           1
Volunteer 4            1
Name: Character, Length: 3950, dtype: int64
```

In [6]:

```
data = data[data['Character'].isin(['Cartman', 'Stan', 'Kyle', 'Randy', 'Butters'])]
data
```

Out[6]:

	Character	Line
0	Stan	You guys, you guys! Chef is going away. \n
1	Kyle	Going away? For how long?\n
2	Stan	Forever.\n
4	Stan	Chef said he's been bored, so he joining a gro...
9	Cartman	I'm gonna miss him. I'm gonna miss Chef and I...
...	...	...
70891	Stan	I think you're pushing it.\n
70892	Randy	How about twenty?\n
70893	Stan	That's not disciprine.\n
70894	Randy	Right right. Does vodka count?\n
70895	Stan	Dad!\n

29622 rows × 2 columns

Разделим выборку на обучающую и тестовую:

In [7]:

```
X = data.drop('Character', axis=1)
Y = data['Character']
```

In [8]:

```
X
```

Out[8]:

	Line
0	You guys, you guys! Chef is going away. \n
1	Going away? For how long?\n
2	Forever.\n
4	Chef said he's been bored, so he joining a gro...
9	I'm gonna miss him. I'm gonna miss Chef and I...
...	...
70891	I think you're pushing it.\n
70892	How about twenty?\n
70893	That's not disciprine.\n
70894	Right right. Does vodka count?\n
70895	Dad!\n

29622 rows × 1 columns

In [9]:

```
Y
```

Out[9]:

```
0           Stan
1           Kyle
2           Stan
4           Stan
9        Cartman
          ...
70891       Stan
70892      Randy
70893       Stan
70894      Randy
70895       Stan
Name: Character, Length: 29622, dtype: object
```

In [10]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_sta
print('{}, {}'.format(X_train.shape, X_test.shape))
print('{}, {}'.format(Y_train.shape, Y_test.shape))
```

```
(22216, 1), (7406, 1)
(22216,), (7406,)
```

In [12]:

```
vectorizer = TfidfVectorizer()
vectorizer.fit(X_train + X_test)
```

Out[12]:

```
TfidfVectorizer(analyzer='word', binary=False, decode_error='strict',
                dtype=<class 'numpy.float64'>, encoding='utf-8',
                input='content', lowercase=True, max_df=1.0, max_featu
res=None,
                min_df=1, ngram_range=(1, 1), norm='l2', preprocessor=
None,
                smooth_idf=True, stop_words=None, strip_accents=None,
                sublinear_tf=False, token_pattern='(?u)\\b\\w\\w+\\b',
                tokenizer=None, use_idf=True, vocabulary=None)
```

In [13]:

```
X_train
```

Out[13]:

	Line
12000	Dude, asshole, you're keeping a lot of other c...
38924	This is gonna be fun.\n
31154	You can say that again.\n
3765	Uh! \n
13854	No, let me tell you somethin', fellers! You al...
...	...
26626	What?! You said nobody would know!\n
42206	Cartman just hit the button, and the ship flew...
12703	Whoa, wait wait, we don't wanna just lie about...
28952	There he goes again.\n
523	Hybrid cars don't cause smugness, people do. ...

22216 rows × 1 columns

In [17]:

```
X_train_vec = vectorizer.transform(X_train['Line'])
X_test_vec = vectorizer.transform(X_test['Line'])
```

In [18]:

```
X_train_vec.shape
```

Out[18]:

```
(22216, 1)
```

In [43]:

```
def test(model):
    print(model)
    model.fit(X_train_vec, Y_train)
    print("accuracy:", accuracy_score(Y_test, model.predict(X_test_vec)))
```

In [44]:

```
test(LogisticRegression(solver='lbfgs', multi_class='auto'))
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept
=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verb
ose=0,
                   warm_start=False)
accuracy: 0.32919254658385094
```

In [45]:

```
test(LinearSVC())
```

```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='l2', random_state=None, tol=0.00
01,
          verbose=0)
accuracy: 0.32919254658385094
```

In [46]:

```
test(MultinomialNB())
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
accuracy: 0.32919254658385094
```

In [47]:

```
test(ComplementNB())
```

```
ComplementNB(alpha=1.0, class_prior=None, fit_prior=True, norm=False)
accuracy: 0.08817175263300027
```

In [48]:

```
test(BernoulliNB())
```

```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
accuracy: 0.32919254658385094
```