На Kaggle.com найден newsgroup20-bbc-news - содержит текст и его класс

Датасет

На Kaggle.com найден newsgroup20-bbc-news - содержит текст и его класс

```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from typing import Dict, Tuple
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
%matplotlib inline
```

```
category = 'category'
text = 'text'

data = pd.read_csv('bbc-text.csv')
data = data[[category, text]]
data=data.dropna(axis=0,how='any')
data.head()
```

	category	text
0	tech	tv future in the hands of viewers with home th...
1	business	worldcom boss left books alone former worldc...
2	sport	tigers wary of farrell gamble leicester say ...
3	sport	yeading face newcastle in fa cup premiership s...
4	entertainment	ocean s twelve raids box office ocean s twelve...

```
X_train, X_test, y_train, y_test = train_test_split(data[text], data[category], test_siz
```

```
def calc(v, c):
    model = Pipeline(
        [("vectorizer", v),
         ("classifier", c)])

    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    d = {'t': y_test, 'p': y_pred}
    df = pd.DataFrame(data=d)
    classes = np.unique(y_test)
    res = dict()


    for c in classes:
        temp_data_flt = df[df['t']==c]
        temp_acc = accuracy_score(
            temp_data_flt['t'].values,
            temp_data_flt['p'].values)
        res[c] = temp_acc

    if len(res)>0:
        print('Points \t Accuracy')

    for i in res:
        print('{} \t {:.2%}'.format(i, res[i]))
    print('average: {}\n\n'.format(np.average(list(res.values()))))
```

```
classificators = [LogisticRegression(C=5.0), MultinomialNB(), ComplementNB(), BernoulliN
vectorizers = [TfidfVectorizer(), CountVectorizer()]

for classificator in classificators:
  for vectorizer in vectorizers:
    calc(vectorizer, classificator)
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: Futu
  FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:469: Futu
  "this warning.", FutureWarning)
Points   Accuracy
business         98.57%
entertainment    97.48%
politics         97.63%
sport    99.49%
tech     96.77%
average: 0.9799108414617039
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: Futu
  FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:469: Futu
  "this warning.", FutureWarning)
Points   Accuracy
business         98.10%
entertainment    98.11%
politics         97.04%
sport    99.49%
tech     96.13%
average: 0.9777425676112352


Points   Accuracy
business         97.62%
entertainment    83.02%
politics         97.63%
sport    99.49%
tech     95.48%
average: 0.9464946167855889


Points   Accuracy
business         96.19%
entertainment    98.11%
politics         98.22%
sport    98.48%
```

Вывод

На основе полученного можно сделать вывод, что лучшим методом в данной ситуации является ComplementNB с CountVectorizer со средней точностью 0,98

```
entertainment    98.74%
politics         97.63%
sport    99.49%
tech     95.48%
average: 0.976988776713711


Points   Accuracy
business         97.62%
entertainment    98.74%
politics         97.63%
sport    99.49%
tech     98.71%
average: 0.9843927705693177


Points   Accuracy
business         99.05%
entertainment    95.60%
politics         92.90%
sport    99.49%
tech     92.26%
average: 0.9585899238226065


Points   Accuracy
business         99.05%
```