Московский государственный технический университет им. Н.Э.Баумана
Кафедра «Системы обработки информации и управления»

Рубежный контроль №2 по дисциплине «Методы машинного обучения»

Выполнил:
Студент группы ИУ5И-23М

Ся Тунтун

Москва — 2022г.

Решение задачи классификации текстов.

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 два варианта векторизации признаков - на основе CountVectorizer и на основе TfidfVectorizer.

В качестве классификаторов необходимо использовать два классификатора по варианту для Вашей группы:

Группа	Классификатор №1	Классификатор №2
ИУ5-21М, ИУ5И-21М	LogisticRegression	Multinomial Naive Bayes - MNB
ИУ5-22М, ИУ5И-22М	RandomForestClassifier	Complement Naive Bayes - CNB
ИУ5-23М, ИУ5И-23М	LinearSVC	Multinomial Naive Bayes - MNB
ИУ5-24М, ИУ5И-24М	KNeighborsClassifier	Complement Naive Bayes - CNB

Мой вариант: LinearSVC&Multinomial Naive Bayes - MNB

```
[1]  import numpy as np
     import pandas as pd
```

```
[4]  #-*- coding : utf-8 -*-
     # coding: utf-8
     import pandas as pd
     data = pd.read_csv("New Task.csv",encoding="unicode_escape")
```

```
[5]  data.keys()

     Index(['News_Headline', 'Link_Of_News', 'Source', 'Stated_On', 'Date',
            'Label'],
           dtype='object')
```

```
[6]  data = data.drop(columns = ['Link_Of_News','Source','Stated_On','Date'])
```

```
[7] import sklearn
    from sklearn.svm import LinearSVC
    from sklearn.naive_bayes import MultinomialNB
    from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
    from sklearn.model_selection import cross_val_score
```

```
[8] data = data.dropna()
```

```
[9] tfidfv = TfidfVectorizer()
    tfidf_features = tfidfv.fit_transform(data['News_Headline'])
    tfidf_features
```

```
<9960x12545 sparse matrix of type '<class 'numpy.float64'>'
        with 161893 stored elements in Compressed Sparse Row format>
```

```
[10] countv = CountVectorizer()
     countv_features = countv.fit_transform(data['News_Headline'])
     countv_features
```

```
<9960x12545 sparse matrix of type '<class 'numpy.int64'>'
```

```
[11] y = data['Label'].values
```

```
cross_val_score(LinearSVC(), tfidf_features, y, scoring='accuracy', cv=3).mean()
```

```
0.23423694779116466
```

```
[13] cross_val_score(LinearSVC(), countv_features, y, scoring='accuracy', cv=3).mean()
```

```
0.21947791164658634
```

```
[14] cross_val_score(MultinomialNB(), tfidf_features, y, scoring='accuracy', cv=3).mean()
```

```
0.2498995983935743
```

```
[15] cross_val_score(MultinomialNB(), countv_features, y, scoring='accuracy', cv=3).mean()
```

```
0.2545180722891566
```

Лучший accuracy достигается при сочитании MultinomialNB и countv vectorizer