Московский государственный технический университет им. Н.Э. Баумана
Факультет «Головной учебно-исследовательский и методический центр
профессиональной реабилитации лиц с ограниченными возможностями здоровья (инвалидов)»
Кафедра «Системы обработки информации и управления»

Рубежный контроль 2

по дисциплине «Методы машинного обучения в АСОИУ»

" Методы обработки текстов "

СТУДЕНТ:
студент группы ИУ5Ц-21М
Москалик А.А.

ПРЕПОДАВАТЕЛЬ:
Гапанюк Ю.Е.

Москва, 2024

Задание:

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 - ИУ5-21М, ИУ5И-21М, ИУ5Ц-21М
Классификатор №1 – KneighborsClassifier
Классификатор №2 – LogisticRegression

Ход работы

Подготовка

Импорт библиотек и загрузка набора данных

```
import numpy as np
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score

# Загрузка данных
data = pd.read_csv("/kaggle/input/spam-text-message-classification/SPAM text message 20170820 - Data.csv", encoding="latin-1")
data = data[['Category', 'Message']]
data.columns = ['label', 'sms']
```

Разделение выборки на обучающую и тестовую

```
X_train, X_test, y_train, y_test = train_test_split(data['sms'], data['label'], test_size=0.2, random_state=42)
```

Векторизация признаков

В этом коде происходит процесс векторизации текстовых данных. Векторизация — это преобразование текстовых данных в числовой формат, который может быть использован моделями машинного обучения.

```
count_vectorizer = CountVectorizer()
X_train_counts = count_vectorizer.fit_transform(X_train)
X_test_counts = count_vectorizer.transform(X_test)

tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

Обучение моделей

1. KNeighborsClassifier с CountVectorizer

```
knn_count = KNeighborsClassifier()
knn_count.fit(X_train_counts, y_train)
y_pred_knn_count = knn_count.predict(X_test_counts)
accuracy_knn_count = accuracy_score(y_test, y_pred_knn_count)
print(f"KNeighborsClassifier с CountVectorizer: {accuracy_knn_count:.4f}")
```

```
KNeighborsClassifier с CountVectorizer: 0.9256
```

2. LogisticRegression с CountVectorizer

```
logreg_count = LogisticRegression(max_iter=1000)
logreg_count.fit(X_train_counts, y_train)
y_pred_logreg_count = logreg_count.predict(X_test_counts)
accuracy_logreg_count = accuracy_score(y_test, y_pred_logreg_count)
print(f"LogisticRegression с CountVectorizer: {accuracy_logreg_count:.4f}")
```

```
LogisticRegression с CountVectorizer: 0.9865
```

3. LogisticRegression с CountVectorizer

```
knn_tfidf = KNeighborsClassifier()
knn_tfidf.fit(X_train_tfidf, y_train)
y_pred_knn_tfidf = knn_tfidf.predict(X_test_tfidf)
accuracy_knn_tfidf = accuracy_score(y_test, y_pred_knn_tfidf)
print(f"KNeighborsClassifier с TfidfVectorizer: {accuracy_knn_tfidf:.4f}")
```

```
KNeighborsClassifier с TfidfVectorizer: 0.9193
```

4. LogisticRegression с CountVectorizer

```
logreg_tfidf = LogisticRegression(max_iter=1000)
logreg_tfidf.fit(X_train_tfidf, y_train)
y_pred_logreg_tfidf = logreg_tfidf.predict(X_test_tfidf)
accuracy_logreg_tfidf = accuracy_score(y_test, y_pred_logreg_tfidf)
print(f"LogisticRegression с TfidfVectorizer: {accuracy_logreg_tfidf:.4f}")
```

```
LogisticRegression с TfidfVectorizer: 0.9749
```

Оценка качества классификации

```
results = {
    'Model': ['KNeighborsClassifier', 'KNeighborsClassifier', 'LogisticRegression', 'LogisticRegression'],
    'Vectorizer': ['CountVectorizer', 'TfidfVectorizer', 'CountVectorizer', 'TfidfVectorizer'],
    'Accuracy': [accuracy_knn_count, accuracy_knn_tfidf, accuracy_logreg_count, accuracy_logreg_tfidf]
}

results_df = pd.DataFrame(results)
print(results_df)
```

	Model	Vectorizer	Accuracy
0	KNeighborsClassifier	CountVectorizer	0.925561
1	KNeighborsClassifier	TfidfVectorizer	0.919283
2	LogisticRegression	CountVectorizer	0.986547
3	LogisticRegression	TfidfVectorizer	0.974888

Вывод:

Наилучшие результаты показала модель LogisticRegression с использованием CountVectorizer с точностью 0.9865. Следовательно, для данной задачи классификации текстов наилучшей комбинацией является использование LogisticRegression в паре с CountVectorizer.