МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Систем обработки информации и управления»

ОТЧЕТ

Рубежный контроль № 2
по дисциплине «Методы машинного обучения»

Тема: «Методы обработки текстов..»

ИСПОЛНИТЕЛЬ: Ли Лююй
ФИО
группа ИУ5И-21М
подпись"

ПРЕПОДАВАТЕЛЬ: Гапанюк Ю .Е

Москва - 2024

Заданте

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.

```python
import pandas as pd
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Создание простого двухклассового текстового набора данных
X, y = make_classification(n_samples=1000, n_features=20, n_classes=2, random_state=42)
df = pd.DataFrame({'text': [' '.join(map(str, x)) for x in X], 'label': y})

# Разделение набора данных на обучающий и тестовый
X_train, X_test, y_train, y_test = train_test_split(df['text'], df['label'], test_size=0.

# Векторизация признаков - CountVectorizer
count_vectorizer = CountVectorizer()
X_train_count = count_vectorizer.fit_transform(X_train)
X_test_count = count_vectorizer.transform(X_test)

# Векторизация признаков - TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

# Инициализация классификаторов
nb_classifier = MultinomialNB()
lr_classifier = LogisticRegression()

# Обучение и оценка классификатора - CountVectorizer
nb_classifier.fit(X_train_count, y_train)
lr_classifier.fit(X_train_count, y_train)

nb_count_acc = accuracy_score(y_test, nb_classifier.predict(X_test_count))
lr_count_acc = accuracy_score(y_test, lr_classifier.predict(X_test_count))

# Обучение и оценка классификатора - TfidfVectorizer
nb_classifier.fit(X_train_tfidf, y_train)
lr_classifier.fit(X_train_tfidf, y_train)

nb_tfidf_acc = accuracy_score(y_test, nb_classifier.predict(X_test_tfidf))
lr_tfidf_acc = accuracy_score(y_test, lr_classifier.predict(X_test_tfidf))

# Вывод результатов
print("Результаты с CountVectorizer:")
print("Точность наивного Байеса:", nb_count_acc)
print("Точность логистической регрессии:", lr_count_acc)

print("\nРезультаты с TfidfVectorizer:")
print("Точность наивного Байеса:", nb_tfidf_acc)
print("Точность логистической регрессии:", lr_tfidf_acc)
```

```
Результаты с CountVectorizer:
Точность наивного Байеса: 0.465
Точность логистической регрессии: 0.465

Результаты с TfidfVectorizer:
Точность наивного Байеса: 0.465
Точность логистической регрессии: 0.465
```