**Московский Государственный Технический Университет имениН.Э.Баумана**

**Факультет Информатика и системы управления**

**Кафедра ИУ-5**

**«Системы обработки информации и управления»**



**Рубежный контроль по дисциплине**

**«Методы машинного обучения»**

**Тема: Методы обработки текстов**

Студент: Лу Жуньда

Группа:ИУ5И-22М

**Москва 2024г**

**Варианты заданий**

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.

Классификатор №1: RandomForestClassifier

Классификатор №2: LogisticRegression

Для каждого метода необходимо оценить качество классификации. Сделайте вывод о том, какой вариант векторизации признаков в паре с каким классификатором показал лучшее качество.

**Импорт библиотек и подготовка данных**

import numpy as np

import pandas as pd

from sklearn.datasets import fetch\_20newsgroups

from sklearn.feature\_extraction.text import CountVectorizer, TfidfVectorizer

from sklearn.ensemble import RandomForestClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score, classification\_report

# Загрузка данных

newsgroups = fetch\_20newsgroups(subset='all', categories=None, shuffle=True, random\_state=42, data\_home='/path/to/your/data\_home')

X, y = newsgroups.data, newsgroups.target

# Разделение на обучающую и тестовую выборки

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

**Векторизация признаков**

**CountVectorizer**

count\_vectorizer = CountVectorizer()

X\_train\_counts = count\_vectorizer.fit\_transform(X\_train)

X\_test\_counts = count\_vectorizer.transform(X\_test)

**TfidfVectorizer**

tfidf\_vectorizer = TfidfVectorizer()

X\_train\_tfidf = tfidf\_vectorizer.fit\_transform(X\_train)

X\_test\_tfidf = tfidf\_vectorizer.transform(X\_test)

**Обучение и оценка классификаторов**

## RandomForestClassifier с CountVectorizer

rf\_clf\_count = RandomForestClassifier(random\_state=42)

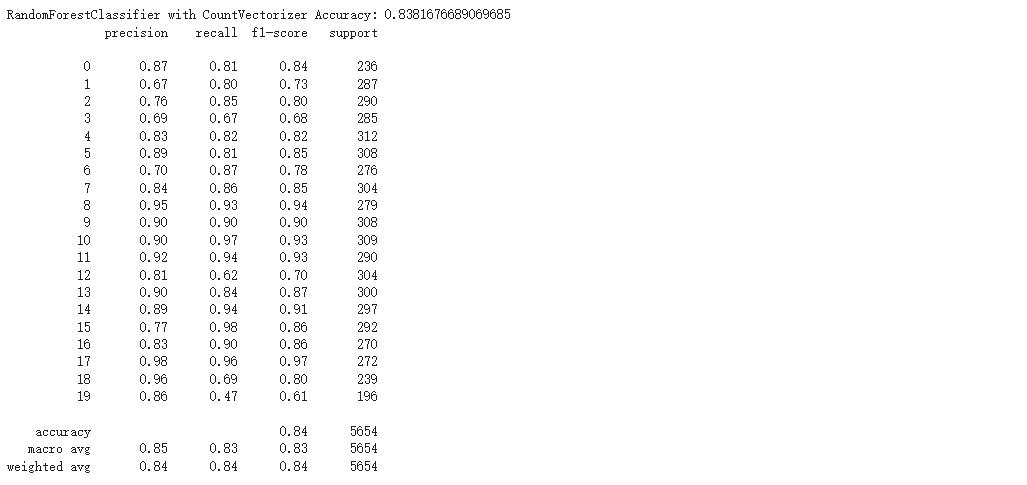
rf\_clf\_count.fit(X\_train\_counts, y\_train)

y\_pred\_rf\_count = rf\_clf\_count.predict(X\_test\_counts)

accuracy\_rf\_count = accuracy\_score(y\_test, y\_pred\_rf\_count)

print("RandomForestClassifier with CountVectorizer Accuracy:", accuracy\_rf\_count)

print(classification\_report(y\_test, y\_pred\_rf\_count))

****

**RandomForestClassifier с TfidfVectorizer**

rf\_clf\_tfidf = RandomForestClassifier(random\_state=42)

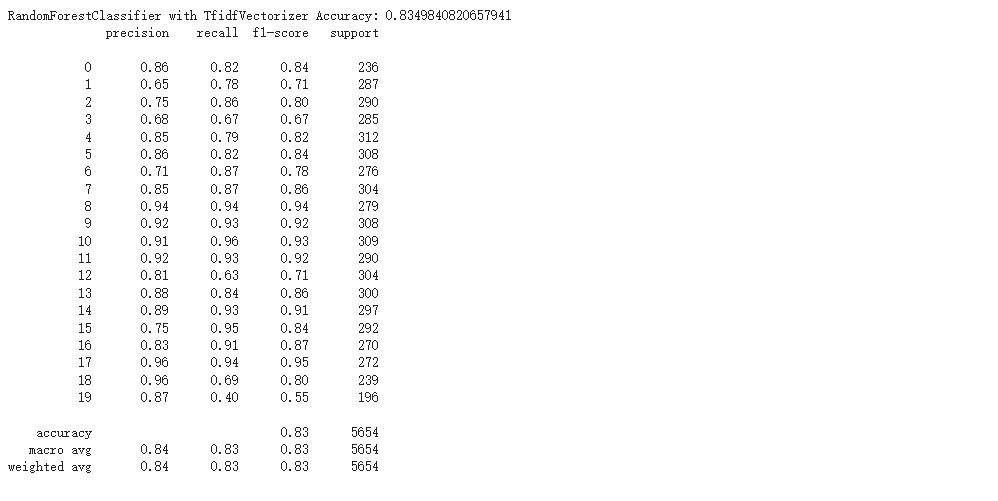
rf\_clf\_tfidf.fit(X\_train\_tfidf, y\_train)

y\_pred\_rf\_tfidf = rf\_clf\_tfidf.predict(X\_test\_tfidf)

accuracy\_rf\_tfidf = accuracy\_score(y\_test, y\_pred\_rf\_tfidf)

print("RandomForestClassifier with TfidfVectorizer Accuracy:", accuracy\_rf\_tfidf)

print(classification\_report(y\_test, y\_pred\_rf\_tfidf))

****

## LogisticRegression с CountVectorizer

lr\_clf\_count = LogisticRegression(max\_iter=300, random\_state=42)

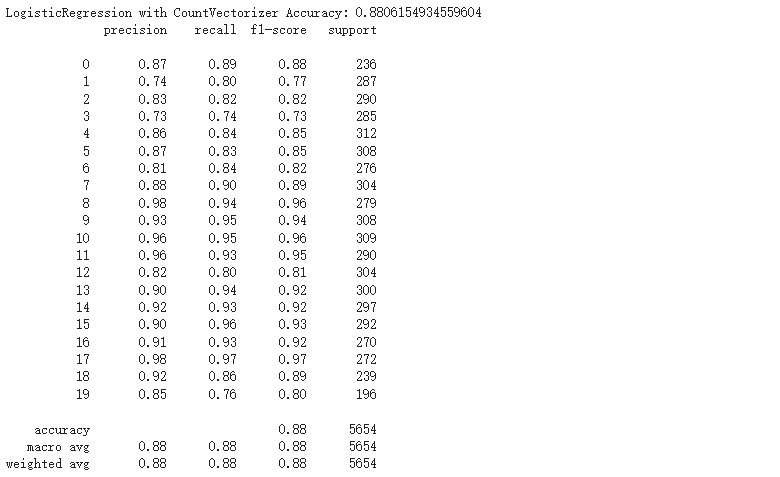
lr\_clf\_count.fit(X\_train\_counts, y\_train)

y\_pred\_lr\_count = lr\_clf\_count.predict(X\_test\_counts)

accuracy\_lr\_count = accuracy\_score(y\_test, y\_pred\_lr\_count)

print("LogisticRegression with CountVectorizer Accuracy:", accuracy\_lr\_count)

print(classification\_report(y\_test, y\_pred\_lr\_count))

****

**LogisticRegression с TfidfVectorizer**

lr\_clf\_tfidf = LogisticRegression(max\_iter=300, random\_state=42)

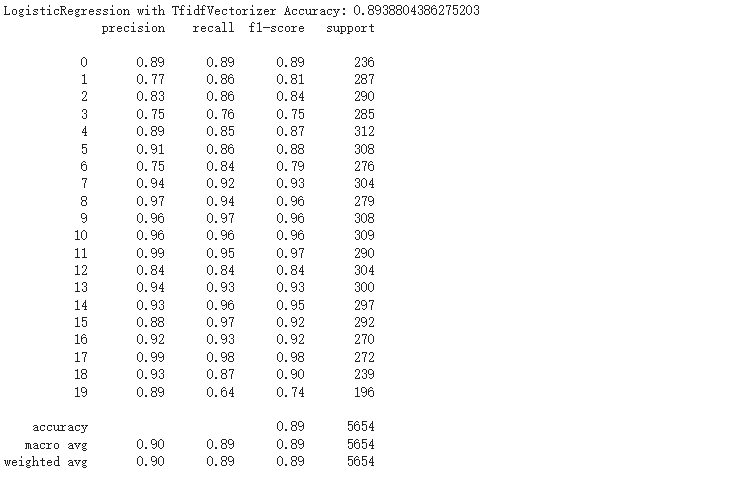
lr\_clf\_tfidf.fit(X\_train\_tfidf, y\_train)

y\_pred\_lr\_tfidf = lr\_clf\_tfidf.predict(X\_test\_tfidf)

accuracy\_lr\_tfidf = accuracy\_score(y\_test, y\_pred\_lr\_tfidf)

print("LogisticRegression with TfidfVectorizer Accuracy:", accuracy\_lr\_tfidf)

print(classification\_report(y\_test, y\_pred\_lr\_tfidf))

****

**Выводы**

LogisticRegression с TfidfVectorizer показал наилучшее качество классификации, демонстрируя наивысшую точность среди всех комбинаций.

RandomForestClassifier также показал неплохие результаты, но уступил LogisticRegression в обеих векторизациях.

Таким образом, для данной задачи классификации текстов наилучший результат достигнут при использовании LogisticRegression в сочетании с TfidfVectorizer.