Московский Государственный Технический Университет имени Н.Э. Баумана

Факультет Информатика и системы управления

**Кафедра ИУ-5 «Системы обработки информации и управления»**

**Отчёт по рубежному контролю № 2 :**

# Методы обработки текстов

**По дисциплине**

«Методы Машинного Обучения»

Выполнила студентка Хэ Синьчэнь
Группа ИУ5И-24М

**Москва 2024г**

**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:**

**ИУ5-24М, ИУ5И-24М GradientBoostingClassifier LogisticRegression**

1. Подготовка данных

Загрузите и прочитайте набор данных о рецензиях на фильмы IMDb.

Преобразуйте данные в формат, подходящий для обучения и тестирования моделей машинного обучения.

```
pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.4)
```

```
[12] import numpy as np
     import pandas as pd
     from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
     from sklearn.model_selection import train_test_split
     from sklearn.svm import LinearSVC
     from sklearn.linear_model import LogisticRegression
     from sklearn.metrics import accuracy_score, classification_report
     import nltk
     from nltk.corpus import movie_reviews
     import random
```

```
[13] # Загрузите и прочитайте набор данных рецензий на фильмы IMDb
     nltk.download('movie_reviews')
```

```
[nltk_data] Downloading package movie_reviews to /root/nltk_data...
[nltk_data]   Unzipping corpora/movie_reviews.zip.
True
```

```
[14] documents = [(list(movie_reviews.words(fileid)), category)
                  for category in movie_reviews.categories()
                  for fileid in movie_reviews.fileids(category)]
     random.shuffle(documents)
```

```
[15]
     # Преобразование документов в DataFrame
     reviews = [' '.join(words) for words, category in documents]
     sentiments = [category for words, category in documents]
     data = pd.DataFrame({'review': reviews, 'sentiment': sentiments})
```

```
     # Просмотр информации о наборе данных
     print(data.head())
     print(data.info())
```

```
                                              review sentiment
0  chill factor is a carbon copy of speed with on...       neg
1  in the year 2029 , captain leo davidson ( mark...       neg
2  gord brody ( tom green ) is an aspiring animat...       neg
3  it was once said that in order to truly enjoy ...       neg
4  synopsis : captain picard and the crew of the ...       pos
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   review     2000 non-null   object
 1   sentiment  2000 non-null   object
dtypes: object(2)
memory usage: 31.4+ KB
None
```

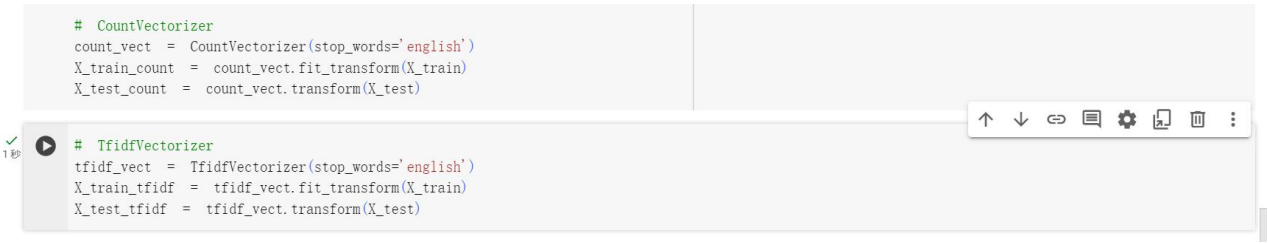
2.Предварительная обработка данных
Текст рецензии на фильм был извлечен в качестве признаков (X).
Метки настроения (положительные или отрицательные) были преобразованы в бинарные метки (1 для положительных рецензий и 0 для отрицательных).

```
0秒  # Предварительная обработка данных
     X = data['review']
     y = data['sentiment'].map({'pos': 1, 'neg': 0})   # 将情感标签转换为二分类
```

3. Разделение набора данных
Набор данных делится на обучающий и тестовый, причем тестовый набор составляет 20%.

```
1秒  # Разделите обучающее и тестовое множество
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

4. Извлечение признаков
Извлеките текстовые признаки с помощью векторизатора CountVectorizer, который вычисляет частоту встречаемости каждого слова в тексте.
Извлечение признаков текста с помощью TfidfVectorizer, который вычисляет частоту слов и обратную частоту документов (TF-IDF) - компромисс между частотой слова и частотой его появления в документе.




5. обучение и оценка моделей
Созданы две модели классификации: LinearSVC и LogisticRegression.
Обучите модели на основе признаков CountVectorizer и TfidfVectorizer, соответственно.
Оцените производительность каждой модели, используя отчеты о точности и классификации (включая точность, отзыв и F1).

```python
# Модели обучения и оценки
for name, model in models.items():
        if 'CountVectorizer' in name:
                model.fit(X_train_count, y_train)
                y_pred = model.predict(X_test_count)
        else:
                model.fit(X_train_tfidf, y_train)
                y_pred = model.predict(X_test_tfidf)

        print(f"Results for {name}:")
        print(f"Accuracy: {accuracy_score(y_test, y_pred)}")
        print(classification_report(y_test, y_pred))
        print("="*60)
```

```
Results for LinearSVC (CountVectorizer):
Accuracy: 0.845
              precision    recall  f1-score   support

           0       0.85      0.85      0.85       209
           1       0.84      0.84      0.84       191

    accuracy                           0.84       400
   macro avg       0.84      0.84      0.84       400
weighted avg       0.84      0.84      0.84       400

============================================================
Results for LogisticRegression (CountVectorizer):
Accuracy: 0.85
              precision    recall  f1-score   support

           0       0.86      0.86      0.86       209
           1       0.84      0.84      0.84       191

    accuracy                           0.85       400
   macro avg       0.85      0.85      0.85       400
weighted avg       0.85      0.85      0.85       400

============================================================
Results for LinearSVC (TfidfVectorizer):
Accuracy: 0.855
              precision    recall  f1-score   support

           0       0.86      0.86      0.86       209
           1       0.84      0.85      0.85       191

    accuracy                           0.85       400
   macro avg       0.85      0.85      0.85       400
weighted avg       0.86      0.85      0.86       400

============================================================
Results for LogisticRegression (TfidfVectorizer):
Accuracy: 0.8475
              precision    recall  f1-score   support

           0       0.87      0.84      0.85       209
           1       0.83      0.86      0.84       191

    accuracy                           0.85       400
   macro avg       0.85      0.85      0.85       400
weighted avg       0.85      0.85      0.85       400

============================================================
```