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

Рубежный контроль №2 по
дисциплине

«Методы машинного обучения»

на тему

«Методы обработки текстов.»

Выполнил:
студент группы ИУ5и-22М
Лун Сыхань

Москва — 2024 г.

Оглавление «Методы обработки текстов.».....错误!未定义书签。

Варианты заданий

Решайте проблемы классификации текста с любым набором данных по вашему выбору. Классификация может быть бинарной или многоуровневой. Целевые объекты в выбранном вами наборе данных могут иметь любое физическое значение; одним из примеров является задача анализа тональности текста.

Необходимо сгенерировать два варианта векторизации признаков — на основе CountVectorizer и на основе TfidfVectorizer.

В качестве классификатора вы должны использовать два классификатора в зависимости от опций вашей группы:

Группа	Классификатор №1	Классификатор №2
ИУ5И-22М	RandomForestClassifier	LogisticRegression

Текстовое описание набора данных:

Было решено использовать набор данных для анализа настроения кинорецензий IMDB, широко применяемый на Kaggle. Этот набор данных содержит тексты рецензий на фильмы и соответствующие им метки настроения (положительные или отрицательные). Текстовые данные в наборе данных были разделены на обучающий и тестовый наборы.

```
[1] !pip install -q scikit-learn pandas
```

```
[5] import pandas as pd
    from sklearn.model_selection import train_test_split
    from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.linear_model import LogisticRegression
    from sklearn.metrics import accuracy_score, classification_report
    import os
    import re
```

```
[6] # 载入数据集
    url = "https://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz"
    !wget $url -O aclImdb_v1.tar.gz
    !tar -xzf aclImdb_v1.tar.gz
```

```
--2024-05-30 13:05:01-- https://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz
Resolving ai.stanford.edu (ai.stanford.edu)... 171.64.68.10
Connecting to ai.stanford.edu (ai.stanford.edu)|171.64.68.10|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 84125825 (80M) [application/x-gzip]
Saving to: 'aclImdb_v1.tar.gz'

aclImdb_v1.tar.gz   100%[===================>]  80.23M  7.36MB/s    in 6.8s

2024-05-30 13:05:08 (11.7 MB/s) - 'aclImdb_v1.tar.gz' saved [84125825/84125825]
```

Предварительная обработка данных и извлечение признаков

Сначала были импортированы необходимые библиотеки и модули, включая экстракторы текстовых объектов (CountVectorizer и TfidfVectorizer), два классификатора (случайный лес и логистическая регрессия) и индикаторы оценки (точность). Далее, путем инициализации объектов CountVectorizer и TfidfVectorizer, текстовые данные преобразуются в матрицу частот слов (CountVectorizer) и матрицу TF-IDF (TfidfVectorizer). Затем эти матрицы функций используются для извлечения функций из текстовых данных обучающего набора и тестового набора. Наконец, целевые функции (метки категорий) обучающего набора и тестового набора извлекаются для обучения и оценки модели.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
import os
import re
```

```
# 使用CountVectorizer进行特征向量化
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
X_test_counts = count_vect.transform(X_test)

# 使用TfidfVectorizer进行特征向量化
tfidf_vect = TfidfVectorizer()
X_train_tfidf = tfidf_vect.fit_transform(X_train)
X_test_tfidf = tfidf_vect.transform(X_test)
```

```
# 数据预处理
X_train = train_data['review']
y_train = train_data['sentiment']
X_test = test_data['review']
y_test = test_data['sentiment']
```

Обучение и оценка модели.

▯ случайный классификатор леса

Классификатор случайного леса — это алгоритм машинного обучения, основанный на деревьях решений. Он состоит из нескольких деревьев решений и делает прогнозы путем объединения этих деревьев.

```
#  使用RandomForestClassifier进行分类并评估
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)
print("RandomForest  with  CountVectorizer  Accuracy:",  accuracy_score(y_test,  y_pred_rf_count))
print(classification_report(y_test,  y_pred_rf_count))

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)
print("RandomForest  with  TfidfVectorizer  Accuracy:",  accuracy_score(y_test,  y_pred_rf_tfidf))
print(classification_report(y_test,  y_pred_rf_tfidf))
```

▯ классификатор логистической регрессии

Логистическая регрессия использует логистическую функцию (также называемую сигмовидной функцией) для преобразования линейной комбинации признаков в значение вероятности, которое представляет вероятность принадлежности выборки к определенной категории.

```
#  使用LogisticRegression进行分类并评估
lr_clf_count  =  LogisticRegression(max_iter=1000,  random_state=42)
lr_clf_count.fit(X_train_counts,  y_train)
y_pred_lr_count  =  lr_clf_count.predict(X_test_counts)
print("LogisticRegression  with  CountVectorizer  Accuracy:",  accuracy_score(y_test,  y_pred_lr_count))
print(classification_report(y_test,  y_pred_lr_count))

lr_clf_tfidf  =  LogisticRegression(max_iter=1000,  random_state=42)
lr_clf_tfidf.fit(X_train_tfidf,  y_train)
y_pred_lr_tfidf  =  lr_clf_tfidf.predict(X_test_tfidf)
print("LogisticRegression  with  TfidfVectorizer  Accuracy:",  accuracy_score(y_test,  y_pred_lr_tfidf))
print(classification_report(y_test,  y_pred_lr_tfidf))
```

Распечатать результаты

```
RandomForest with CountVectorizer Accuracy: 0.84536
              precision    recall  f1-score   support

           0       0.84      0.85      0.85     12500
           1       0.85      0.84      0.84     12500

    accuracy                           0.85     25000
   macro avg       0.85      0.85      0.85     25000
weighted avg       0.85      0.85      0.85     25000

RandomForest with TfidfVectorizer Accuracy: 0.83812
              precision    recall  f1-score   support

           0       0.83      0.85      0.84     12500
           1       0.85      0.83      0.84     12500

    accuracy                           0.84     25000
   macro avg       0.84      0.84      0.84     25000
weighted avg       0.84      0.84      0.84     25000

LogisticRegression with CountVectorizer Accuracy: 0.86668
              precision    recall  f1-score   support

           0       0.86      0.87      0.87     12500
           1       0.87      0.86      0.87     12500

    accuracy                           0.87     25000
   macro avg       0.87      0.87      0.87     25000
weighted avg       0.87      0.87      0.87     25000

LogisticRegression with TfidfVectorizer Accuracy: 0.88316
              precision    recall  f1-score   support

           0       0.88      0.88      0.88     12500
           1       0.88      0.88      0.88     12500

    accuracy                           0.88     25000
   macro avg       0.88      0.88      0.88     25000
weighted avg       0.88      0.88      0.88     25000
```

Вывод:

Эти результаты показывают точность использования различных представлений признаков (CountVectorizer и TfidfVectorizer) и различных классификаторов (Random Forest и Logistic Regression) на данном наборе данных. В частности, точность классификатора случайного леса с использованием CountVectorizer и TfidfVectorizer составляет 0,845 и 0,838 соответственно, а точность классификатора логистической регрессии с использованием CountVectorizer и TfidfVectorizer - 0,867 и 0,883 соответственно, что несколько выше, чем точность классификатора логистической регрессии с использованием TfidfVectorizer для классификатора случайного леса. Feature: показывает точность 0,883, что является самым высоким показателем среди всех моделей. Это говорит о том, что модель логистической регрессии превзошла модель случайного леса в этом наборе данных, а представление признаков с помощью TfidfVectorizer достигло наивысшей точности среди всех моделей, возможно, потому, что оно лучше представляет признаки в текстовых данных. Эти результаты подчеркивают влияние выбора подходящего представления признаков на производительность модели и тот факт, что при обработке текстовых данных крайне важно учитывать особенности данных.