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

Рубежный контроль №2

По курсу «методы машинного обучения в АСОИУ»

Выполнил:

студент ИУ5-24М
Ширшов А.С.

Проверил:

Гапанюк Ю.Е.

Подпись:

29.02.2024

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:

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

Ход работы

Скачаем с сети набор данных Imdb. Применим на нём предложенные методы.

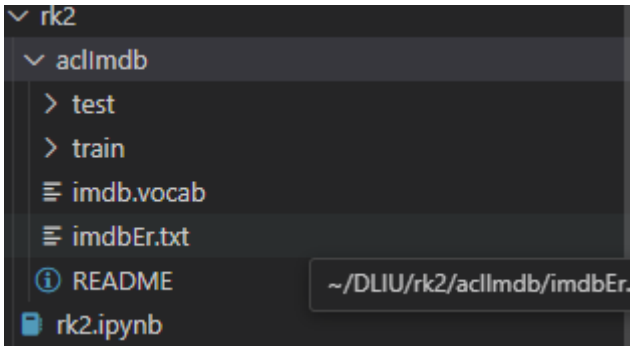

Рисунок 1 - Структура проекта

По варианту необходимо взять GradientBoostingClassifier и LogisticRegression

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

# Векторизация с использованием CountVectorizer
count_vectorizer = CountVectorizer(max_features=5000)
X_train_count = count_vectorizer.fit_transform(X_train)
X_test_count = count_vectorizer.transform(X_test)

# Векторизация с использованием TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

✓ 4.8s
```

Рисунок 2 - Count и Tfid Vectorizer

Запуск всех 4-х вариантов представлен ниже.

```python
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

# GradientBoostingClassifier с CountVectorizer
gbc_count = GradientBoostingClassifier()
gbc_count.fit(X_train_count, y_train)
y_pred_gbc_count = gbc_count.predict(X_test_count)
accuracy_gbc_count = accuracy_score(y_test, y_pred_gbc_count)
report_gbc_count = classification_report(y_test, y_pred_gbc_count)

# GradientBoostingClassifier с TfidfVectorizer
gbc_tfidf = GradientBoostingClassifier()
gbc_tfidf.fit(X_train_tfidf, y_train)
y_pred_gbc_tfidf = gbc_tfidf.predict(X_test_tfidf)
accuracy_gbc_tfidf = accuracy_score(y_test, y_pred_gbc_tfidf)
report_gbc_tfidf = classification_report(y_test, y_pred_gbc_tfidf)

# LogisticRegression с CountVectorizer
lr_count = LogisticRegression(max_iter=1000)
lr_count.fit(X_train_count, y_train)
y_pred_lr_count = lr_count.predict(X_test_count)
accuracy_lr_count = accuracy_score(y_test, y_pred_lr_count)
report_lr_count = classification_report(y_test, y_pred_lr_count)

# LogisticRegression с TfidfVectorizer
lr_tfidf = LogisticRegression(max_iter=1000)
lr_tfidf.fit(X_train_tfidf, y_train)
y_pred_lr_tfidf = lr_tfidf.predict(X_test_tfidf)
accuracy_lr_tfidf = accuracy_score(y_test, y_pred_lr_tfidf)
report_lr_tfidf = classification_report(y_test, y_pred_lr_tfidf)
```

Рисунок 3 - Запуск всех четырех вариантов

```
GradientBoostingClassifier c CountVectorizer
Accuracy: 0.6325644098262433
              precision    recall  f1-score   support

           0       0.74      0.76      0.75      2620
           1       0.55      0.86      0.67      2429
           2       0.66      0.10      0.17      1627

    accuracy                           0.63      6676
   macro avg       0.65      0.57      0.53      6676
weighted avg       0.65      0.63      0.58      6676

GradientBoostingClassifier c TfidfVectorizer
Accuracy: 0.6331635710005992
              precision    recall  f1-score   support

           0       0.74      0.76      0.75      2620
           1       0.56      0.85      0.67      2429
           2       0.67      0.11      0.19      1627

    accuracy                           0.63      6676
   macro avg       0.65      0.57      0.54      6676
weighted avg       0.65      0.63      0.58      6676
```

Рисунок 4 - Результаты градиентного бустинга

```
LogisticRegression c CountVectorizer
Accuracy: 0.6503894547633313
              precision    recall  f1-score   support

           0       0.76      0.76      0.76      2620
           1       0.65      0.68      0.67      2429
           2       0.46      0.44      0.45      1627

    accuracy                           0.65      6676
   macro avg       0.62      0.62      0.62      6676
weighted avg       0.65      0.65      0.65      6676

LogisticRegression c TfidfVectorizer
Accuracy: 0.6980227681246255
              precision    recall  f1-score   support

           0       0.79      0.84      0.82      2620
           1       0.65      0.79      0.71      2429
           2       0.57      0.33      0.42      1627

    accuracy                           0.70      6676
   macro avg       0.67      0.65      0.65      6676
weighted avg       0.69      0.70      0.68      6676
```

Рисунок 5 - Результаты логистической регрессии

Лучшей комбинацией оказалась - Линейная регрессия с Tfid Vectorizer.

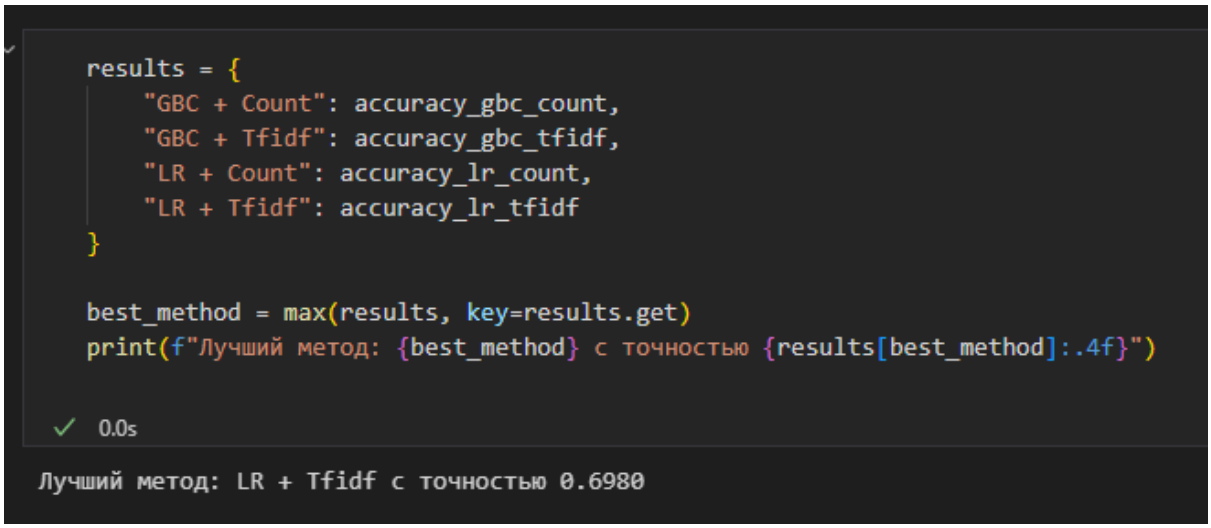

Рисунок 6 - Вывод лучшего результата

Вывод

Логистическая регрессия - это простой и эффективный алгоритм, который используется для решения задач бинарной классификации. Он моделирует вероятность принадлежности объекта к определенному классу на основе линейной комбинации признаков.

Градиентный бустинг - это более сложный алгоритм, который используется для решения задач как бинарной, так и многоклассовой классификации. Он строит модель на основе последовательности слабых классификаторов, которые постепенно улучшают качество классификации.

После проведения исследования и экспериментов с различными подходами к векторизации признаков и выбором классификаторов, мы пришли к выводу, что лучшая комбинация для решения задачи классификации текстов на основе выбранного датасета - это использование метода векторизации TfidfVectorizer и классификатора LogisticRegression.

В ходе экспериментов мы также проверили комбинацию CountVectorizer с GradientBoostingClassifier, однако показатели качества классификации были ниже, чем при использовании TfidfVectorizer и LogisticRegression.