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 Яцзинь
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.

В качестве классификаторов необходимо использовать два классификатора по варианту для моей руппы:

КСоседиКлассификатор (KNeighborsClassifier)

Логистическая регрессия (LogisticRegression)

Загрузить набор данных

Это набор данных рецензий на фильмы, которые будут использоваться для задачи НЛП по анализу настроений. Он имеет форму предложений, где каждому предложению присваивается оценка настроения от 0 до 4 (1 = Очень Плохо 2 = Плохо 3 = Нейтрально 4 = Хорошо 5 = Очень хорошо).

Рисунок 1- Код загрузить набор данных

Преобразовать в вектор объектов

Используйте два разных метода для преобразования текста в векторы объектов (CountVectorizer и TfidfVectorizer).

```
# @title   Преобразовать в вектор объектов
# Method 1: Using CountVectorizer
count_vectorizer = CountVectorizer()
X_train_count = count_vectorizer.fit_transform(train_texts)
X_test_count = count_vectorizer.transform(test_texts)

# Method 2: Using TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(train_texts)
X_test_tfidf = tfidf_vectorizer.transform(test_texts)
```

Рисунок 2- Код преобразовать в вектор объектов

Определить классификатор

В зависимости от параметров класса выбраны следующие классификаторы: KNeighboursClassifier и LogisticRегрессия.

```
[8] # @title Определить классификатор
    # Define classifiers
    knn_classifier = KNeighborsClassifier()
    logistic_classifier = LogisticRegression()
```

Рисунок 3-Код определить классификатор

Обучение и тестирование с помощью CountVectorizer

Используйте классификатор CountVectorizer для обучения и тестирования функций, созданных KNeighborsClassifier и LogisticRegrade соответственно.

```
# @title Обучение и тестирование с помощью CountVectorizer

print("Training and testing with CountVectorizer:")
knn_classifier.fit(X_train_count, train_labels)
knn_pred_count = knn_classifier.predict(X_test_count)
print("KNN Accuracy (CountVectorizer):", accuracy_score(test_labels, knn_pred_count))
print("KNN Classification Report (CountVectorizer):\n", classification_report(test_labels, knn_pred_count))

logistic_classifier.fit(X_train_count, train_labels)
logistic_pred_count = logistic_classifier.predict(X_test_count)
print("Logistic Regression Accuracy (CountVectorizer):", accuracy_score(test_labels, logistic_pred_count))
print("Logistic Regression Classification Report (CountVectorizer):\n", classification_report(test_labels, logistic_pred_count))
```

Рисунок4-Код Обучение и тестирование с помощью CountVectorizer

```
Training and testing with CountVectorizer:
KNN Accuracy (CountVectorizer): 0.6200407170167337
KNN Classification Report (CountVectorizer):
              precision    recall  f1-score   support

           0       0.48      0.22      0.30       760
           1       0.52      0.26      0.35      3287
           2       0.64      0.93      0.76     11129
           3       0.58      0.25      0.35      3982
           4       0.56      0.15      0.23       981

    accuracy                           0.62     20139
   macro avg       0.55      0.36      0.40     20139
weighted avg       0.60      0.62      0.57     20139

Logistic Regression Accuracy (CountVectorizer): 0.6651770197129947
Logistic Regression Classification Report (CountVectorizer):
              precision    recall  f1-score   support

           0       0.50      0.21      0.30       760
           1       0.54      0.37      0.44      3287
           2       0.71      0.90      0.79     11129
           3       0.59      0.44      0.50      3982
           4       0.58      0.32      0.41       981

    accuracy                           0.67     20139
   macro avg       0.59      0.45      0.49     20139
weighted avg       0.64      0.67      0.64     20139
```

Рисунок5- результаты обучение и тестирование с помощью CountVectorizer

Обучение и тестирование с помощью TfidfVectorizer

Используйте классификатор TfidfVectorizer для обучения и тестирования функций, созданных KNeighborsClassifier и LogisticRegrade соответственно.

Обучение и тестирование с помощью TfidfVectorizer

```
# @title Обучение и тестирование с помощью TfidfVectorizer
#Training and testing with TfidfVectorizer

print("\nTraining and testing with TfidfVectorizer:")
knn_classifier.fit(X_train_tfidf, train_labels)
knn_pred_tfidf = knn_classifier.predict(X_test_tfidf)
print("KNN Accuracy (TfidfVectorizer):", accuracy_score(test_labels, knn_pred_tfidf))
print("KNN Classification Report (TfidfVectorizer):\n", classification_report(test_labels, knn_pred_tfidf))

logistic_classifier.fit(X_train_tfidf, train_labels)
logistic_pred_tfidf = logistic_classifier.predict(X_test_tfidf)
print("Logistic Regression Accuracy (TfidfVectorizer):", accuracy_score(test_labels, logistic_pred_tfidf))
print("Logistic Regression Classification Report (TfidfVectorizer):\n", classification_report(test_labels, logistic_pred_tfidf))
```

Рисунок6- результаты обучение и тестирование с помощью TfidfVectorizer

```
# @title Обучение и тестирование с помощью CountVectorizer

print("Training and testing with CountVectorizer:")
knn_classifier.fit(X_train_count, train_labels)
knn_pred_count = knn_classifier.predict(X_test_count)
print("KNN Accuracy (CountVectorizer):", accuracy_score(test_labels, knn_pred_count))
print("KNN Classification Report (CountVectorizer):\n", classification_report(test_labels, knn_pred_count))

logistic_classifier.fit(X_train_count, train_labels)
logistic_pred_count = logistic_classifier.predict(X_test_count)
print("Logistic Regression Accuracy (CountVectorizer):", accuracy_score(test_labels, logistic_pred_count))
print("Logistic Regression Classification Report (CountVectorizer):\n", classification_report(test_labels, logistic_pred_count))
```

Рисунок7- результаты обучение и тестирование с помощью TfidfVectorizer

```
Training and testing with CountVectorizer:
KNN Accuracy (CountVectorizer): 0.6200407170167337
KNN Classification Report (CountVectorizer):
              precision    recall  f1-score   support

           0       0.48      0.22      0.30       760
           1       0.52      0.26      0.35      3287
           2       0.64      0.93      0.76     11129
           3       0.58      0.25      0.35      3982
           4       0.56      0.15      0.23       981

    accuracy                           0.62     20139
   macro avg       0.55      0.36      0.40     20139
weighted avg       0.60      0.62      0.57     20139

Logistic Regression Accuracy (CountVectorizer): 0.6651770197129947
Logistic Regression Classification Report (CountVectorizer):
              precision    recall  f1-score   support

           0       0.50      0.21      0.30       760
           1       0.54      0.37      0.44      3287
           2       0.71      0.90      0.79     11129
           3       0.59      0.44      0.50      3982
           4       0.58      0.32      0.41       981

    accuracy                           0.67     20139
   macro avg       0.59      0.45      0.49     20139
weighted avg       0.64      0.67      0.64     20139
```


```
Training and testing with CountVectorizer:
KNN Accuracy (CountVectorizer): 0.6200407170167337
KNN Classification Report (CountVectorizer):
              precision    recall  f1-score   support

           0       0.48      0.22      0.30       760
           1       0.52      0.26      0.35      3287
           2       0.64      0.93      0.76     11129
           3       0.58      0.25      0.35      3982
           4       0.56      0.15      0.23       981

    accuracy                           0.62     20139
   macro avg       0.55      0.36      0.40     20139
weighted avg       0.60      0.62      0.57     20139

Logistic Regression Accuracy (CountVectorizer): 0.6651770197129947
Logistic Regression Classification Report (CountVectorizer):
              precision    recall  f1-score   support

           0       0.50      0.21      0.30       760
           1       0.54      0.37      0.44      3287
           2       0.71      0.90      0.79     11129
           3       0.59      0.44      0.50      3982
           4       0.58      0.32      0.41       981

    accuracy                           0.67     20139
   macro avg       0.59      0.45      0.49     20139
weighted avg       0.64      0.67      0.64     20139
```