Выполнил: ЧжаоЛян

Группа: ИУ5И-22М

1 Random Forest Classifier

2 LogisticRegression

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

# Загрузка набора данных SMS Spam Collection
data = pd.read_csv( filepath_or_buffer: 'SMSSpamCollection', sep='\t', header=None, names=['label', 'message'])
data['label'] = data['label'].map({'ham': 0, 'spam': 1})  # Преобразование меток в бинарные значения

X = data['message']
y = data['label']

# Разделение набора данных на тренировочную и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split( *arrays: X, y, test_size=0.2, random_state=42)

# Извлечение признаков с использованием CountVectorizer
count_vectorizer = CountVectorizer(stop_words='english')
X_train_counts = count_vectorizer.fit_transform(X_train)
X_test_counts = count_vectorizer.transform(X_test)

# Извлечение признаков с использованием TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(stop_words='english')
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

```python
# Определение и обучение классификатора Random Forest
rf_classifier_counts = RandomForestClassifier(random_state=42)
rf_classifier_counts.fit(X_train_counts, y_train)
y_pred_rf_counts = rf_classifier_counts.predict(X_test_counts)

rf_classifier_tfidf = RandomForestClassifier(random_state=42)
rf_classifier_tfidf.fit(X_train_tfidf, y_train)
y_pred_rf_tfidf = rf_classifier_tfidf.predict(X_test_tfidf)

# Определение и обучение логистической регрессии
lr_classifier_counts = LogisticRegression(max_iter=1000, random_state=42)
lr_classifier_counts.fit(X_train_counts, y_train)
y_pred_lr_counts = lr_classifier_counts.predict(X_test_counts)

lr_classifier_tfidf = LogisticRegression(max_iter=1000, random_state=42)
lr_classifier_tfidf.fit(X_train_tfidf, y_train)
y_pred_lr_tfidf = lr_classifier_tfidf.predict(X_test_tfidf)
```

```python
# Оценка производительности классификаторов
print("Random Forest с CountVectorizer:")
print("Точность:", accuracy_score(y_test, y_pred_rf_counts))
print(classification_report(y_test, y_pred_rf_counts))

print("Random Forest с TfidfVectorizer:")
print("Точность:", accuracy_score(y_test, y_pred_rf_tfidf))
print(classification_report(y_test, y_pred_rf_tfidf))

print("Логистическая регрессия с CountVectorizer:")
print("Точность:", accuracy_score(y_test, y_pred_lr_counts))
print(classification_report(y_test, y_pred_lr_counts))

print("Логистическая регрессия с TfidfVectorizer:")
print("Точность:", accuracy_score(y_test, y_pred_lr_tfidf))
print(classification_report(y_test, y_pred_lr_tfidf))
```

Выводы:

```
Random Forest с CountVectorizer:
Точность: 0.9766816143497757
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       966
           1       1.00      0.83      0.90       149

    accuracy                           0.98      1115
   macro avg       0.99      0.91      0.95      1115
weighted avg       0.98      0.98      0.98      1115

Random Forest с TfidfVectorizer:
Точность: 0.9811659192825112
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       966
           1       1.00      0.86      0.92       149

    accuracy                           0.98      1115
   macro avg       0.99      0.93      0.96      1115
weighted avg       0.98      0.98      0.98      1115
```

```
Логистическая регрессия с CountVectorizer:
Точность: 0.9856502242152466
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       966
           1       1.00      0.89      0.94       149

    accuracy                           0.99      1115
   macro avg       0.99      0.95      0.97      1115
weighted avg       0.99      0.99      0.99      1115

Логистическая регрессия с TfidfVectorizer:
Точность: 0.9695067264573991
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       966
           1       1.00      0.77      0.87       149

    accuracy                           0.97      1115
   macro avg       0.98      0.89      0.93      1115
weighted avg       0.97      0.97      0.97      1115
```

1. **Наилучшая точность** была достигнута при использовании логистической регрессии с CountVectorizer (0.9857).
2. **Random Forest** также показал хорошие результаты, особенно с TfidfVectorizer (0.9812), который был лучше, чем с CountVectorizer (0.9767).
3. **Precision** для класса 1 (спам) был высоким для всех моделей, но **Recall** для этого класса был ниже, особенно для логистической регрессии с TfidfVectorizer (0.77).
4. **F1-score** для класса 1 был наивысшим у логистической регрессии с CountVectorizer (0.94), что указывает на лучшую сбалансированность между precision и recall.

Общий вывод:Для данного набора данных наилучшей комбинацией является логистическая регрессия с CountVectorizer. Эта комбинация показала наивысшую точность и хорошие метрики precision и recall для обоих классов.