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»

Выполнила:

студентка ИУ5-23М
Светашева Ю.В.

Проверил:

Гапанюк Ю.Е.

Подпись:

11.05.2024

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:

Группа	Классификатор №1	Классификатор №2
ИУ5-21М, ИУ5И-21М, ИУ5Ц-21М	KNeighborsClassifier	LogisticRegression
ИУ5-22М, ИУ5И-22М	RandomForestClassifier	LogisticRegression
ИУ5-23М, ИУ5И-23М	**LinearSVC**	**LogisticRegression**
ИУ5-24М, ИУ5И-24М	GradientBoostingClassifier	LogisticRegression
ИУ5-25М, ИУ5И-25М, ИУ5И-26М	SVC	LogisticRegression

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

Решение

Загружаем датасет

```
df = pd.read_csv('stock_exchange.csv')
```

```
df.head(2)
```

	date	sentiment	source	subject	text	title
0	2023-12-19 06:40:41	{'class': 'negative', 'polarity': -0.1, 'subje...	CryptoNews	altcoin	Grayscale CEO Michael Sonnenshein believes the...	Grayscale CEO Calls for Simultaneous Approval ...
1	2023-12-19 06:03:24	{'class': 'neutral', 'polarity': 0.0, 'subject...	CryptoNews	blockchain	In an exclusive interview with CryptoNews, Man...	Indian Government is Actively Collaborating Wi...

Разделяем данные на тестовую и обучающую выборку и векторизуем их с помощью CountVectorizer и TfidfVectorizer

```
X, Y = df['text'], df['source']
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

time_arr = []
```

```
# CountVectorizer
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
X_test_counts = count_vect.transform(X_test)
```

```
# TfidfVectorizer
tfidf_vect = TfidfVectorizer()
X_train_tfidf = tfidf_vect.fit_transform(X_train)
X_test_tfidf = tfidf_vect.transform(X_test)
```

Обучаем классификаторы CountVectorizer

```
# LinearSVC
gbc = LinearSVC()
start_time = time.time()
gbc.fit(X_train_counts, y_train)
train_time = time.time() - start_time
time_arr.append(train_time)
pred_gbc_counts = gbc.predict(X_test_counts)
print("Точность (CountVectorizer + LinearSVC):", accuracy_score(y_test, pred_gbc_counts))

# Logistic Regression
lr = LogisticRegression(max_iter=1200)
start_time = time.time()
lr.fit(X_train_counts, y_train)
train_time = time.time() - start_time
time_arr.append(train_time)
pred_lr_counts = lr.predict(X_test_counts)
print("Точность (CountVectorizer + LogisticRegression):", accuracy_score(y_test, pred_lr_counts))
```

```
Точность (CountVectorizer + LinearSVC): 0.6616541353383458
Точность (CountVectorizer + LogisticRegression): 0.6917293233082706
```

Обучаем классификаторы TfidfVectorizer

```
# LinearSVC
gbc = LinearSVC()
start_time = time.time()
gbc.fit(X_train_tfidf, y_train)
train_time = time.time() - start_time
time_arr.append(train_time)
pred_gbc_tfidf = gbc.predict(X_test_tfidf)
print("Точность (TfidfVectorizer + LinearSVC):", accuracy_score(y_test, pred_gbc_tfidf))

# Logistic Regression
lr = LogisticRegression(max_iter=1200)
start_time = time.time()
lr.fit(X_train_tfidf, y_train)
train_time = time.time() - start_time
time_arr.append(train_time)
pred_lr_tfidf = lr.predict(X_test_tfidf)
print("Точность (TfidfVectorizer + LogisticRegression):", accuracy_score(y_test, pred_lr_tfidf))
```

```
Точность (TfidfVectorizer + LinearSVC): 0.6741854636591479
Точность (TfidfVectorizer + LogisticRegression): 0.6850459482038429
```

Выводим отсортированные данные

```
from tabulate import tabulate

data = [
    ["(CountVectorizer + LogisticRegression)", accuracy_score(y_test, pred_lr_counts), time_arr[0]],
    ["(CountVectorizer + LinearSVC)", accuracy_score(y_test, pred_gbc_counts), time_arr[1]],
    ["(TfidfVectorizer + LogisticRegression)", accuracy_score(y_test, pred_lr_tfidf), time_arr[2]],
    ["(TfidfVectorizer + LinearSVC)", accuracy_score(y_test, pred_gbc_tfidf), time_arr[3]]
]

sorted_data = sorted(data, key=lambda x: x[1], reverse=True)

# Вывод отсортированных данных в виде таблицы
print(tabulate(sorted_data, ['Связка','Точность валидации', 'Время обучения'], tablefmt="grid"))
```

Связка	Точность валидации	Время обучения
(CountVectorizer + LogisticRegression)	0.691729	1.54916
(TfidfVectorizer + LogisticRegression)	0.685046	0.586139
(TfidfVectorizer + LinearSVC)	0.674185	3.23617
(CountVectorizer + LinearSVC)	0.661654	11.2966