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 в автоматизированных системах»

Тема «Методы обработки текстов»

ИСПОЛНИТЕЛЬ: ___Калюта Н.И.___
ФИО
группа ИУ5-22М _____
подпись
"30"___05___2024 г.

ПРЕПОДАВАТЕЛЬ: ___ _____
ФИО
_____
подпись
"__"_____2024 г.

Москва - 2024

Задача 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 Вашей группы: ИУ5-22М - RandomForestClassifier, LogisticRegression

Импортирование необходимых библиотек

```
[2]  import pandas as pd
     import time
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
     from sklearn.linear_model import LogisticRegression
     from sklearn.metrics import accuracy_score
```

Подключение гугл диска

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Загрузка данных из CSV-файла

```
[13] train_data = pd.read_csv('/content/drive/MyDrive/RK/Electric_train.csv', encoding='latin1')
     test_data = pd.read_csv('/content/drive/MyDrive/RK/Electric_test.csv', encoding='latin1')
```

```
train_data.head()
```

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type	Clean Alternative Fuel Vehicle (CAFV) Eligibility	Electric Range	Base MSRP	Legislative District	DOL Vehicle ID	Vehicle Location	Electric Utility	2020 Census Tract
0	WBY8P6C58K	King	Seattle	WA	98115.0	2019	BMW	I3	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	153	0	43.0	259254397	POINT (-122.3008235 47.6862671)	CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)	5.303300e+10
1	5YJSA1DN4D	Kitsap	Bremerton	WA	98312.0	2013	TESLA	MODEL S	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	208	69900	35.0	127420940	POINT (-122.6961203 47.5759584)	PUGET SOUND ENERGY INC	5.303508e+10
2	5YJSA1E26J	King	Kent	WA	98042.0	2018	TESLA	MODEL S	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	249	0	47.0	170287183	POINT (-122.1145138 47.3581107)	PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)	5.303303e+10
3	WBY2Z2C54E	King	Bellevue	WA	98004.0	2014	BMW	I8	Plug-in Hybrid Electric Vehicle (PHEV)	Not eligible due to low battery range	14	0	41.0	205545868	POINT (-122.202397 47.619252)	PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)	5.303302e+10
4	5YJXCDE23J	King	Bellevue	WA	98004.0	2018	TESLA	MODEL X	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	238	0	41.0	237977386	POINT (-122.202397 47.619252)	PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)	5.303302e+10

```
[36] train_data.shape
```

```
(149060, 17)
```

```
test_data.head()
```

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type	Clean Alternative Fuel Vehicle (CAFV) Eligibility	Electric Range	Base MSRP	Legislative District	DOL Vehicle ID	Vehicle Location	Electric Utility	2020 Census Tract
0	1GYKPSRL2R	Spokane	Newman Lake	WA	99025.0	2024.0	CADILLAC	LYRIQ	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b...	0.0	0.0	4.0	261204632.0	POINT (-117.075672 47.736783)	BONNEVILLE POWER ADMINISTRATION||AVISTA CORP||...	5.306301e+10
1	KM8KRDAF1P	Snohomish	Mountlake Terrace	WA	98043.0	2023.0	HYUNDAI	IONIQ 5	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b...	0.0	0.0	1.0	260855038.0	POINT (-122.306706 47.792043)	PUGET SOUND ENERGY INC	5.306105e+10
2	7SAYGDEF3P	King	Seattle	WA	98101.0	2023.0	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b...	0.0	0.0	43.0	236290512.0	POINT (-122.3340795 47.6099315)	CITY OF SEATTLE - (WA)||CITY OF TACOMA - (WA)	5.303301e+10
3	7SAYGDEE6P	King	Newcastle	WA	98056.0	2023.0	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b...	0.0	0.0	41.0	241560420.0	POINT (-122.1819876 47.5098889)	PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)	5.303302e+10
4	JTMEB3FV3M	San Juan	Friday Harbor	WA	98250.0	2021.0	TOYOTA	RAV4 PRIME	Plug-in Hybrid Electric Vehicle (PHEV)	Clean Alternative Fuel Vehicle Eligible	42.0	0.0	40.0	142824888.0	POINT (-123.0679454 48.5327669)	BONNEVILLE POWER ADMINISTRATION||ORCAS POWER &...	5.305596e+10

```
[35] test_data.shape
```

```
(36992, 17)
```

Удаляем строки, содержащие пропущенные значения (NaN)

```
[8] train_data.dropna(inplace=True)
    test_data.dropna(inplace=True)
```

```
X_train = train_data['City']
y_train = train_data['Electric Vehicle Type']

X_test = test_data['City']
y_test = test_data['Electric Vehicle Type']
```

Проверить наличие пропущенных значений (NaN) в различных наборах данных.

```
[10] def check_missing(data, name):
         missing = data.isnull().sum()
         print(f'У {name} {missing} пропущенных строк')
```

```
check_missing(train_data, 'train_data')
check_missing(test_data, 'test_data')
check_missing(X_train, 'X_train')
check_missing(X_test, 'X_test')
check_missing(y_train, 'y_train')
check_missing(y_test, 'y_test')
```

```
У train_data VIN (1-10)                                  0
County                                                   0
City                                                     0
State                                                    0
Postal Code                                              0
Model Year                                               0
Make                                                     0
Model                                                    0
Electric Vehicle Type                                    0
Clean Alternative Fuel Vehicle (CAFV) Eligibility        0
Electric Range                                           0
Base MSRP                                                0
Legislative District                                     0
DOL Vehicle ID                                           0
Vehicle Location                                         0
Electric Utility                                         0
2020 Census Tract                                        0
dtype: int64 пропущенных строк
У test_data VIN (1-10)                                   0
County                                                   0
City                                                     0
State                                                    0
Postal Code                                              0
Model Year                                               0
Make                                                     0
Model                                                    0
Electric Vehicle Type                                    0
Clean Alternative Fuel Vehicle (CAFV) Eligibility        0
Electric Range                                           0
Base MSRP                                                0
Legislative District                                     0
DOL Vehicle ID                                           0
Vehicle Location                                         0
Electric Utility                                         0
2020 Census Tract                                        0
dtype: int64 пропущенных строк
У X_train 0 пропущенных строк
У X_test 0 пропущенных строк
У y_train 0 пропущенных строк
У y_test 0 пропущенных строк
```

Векторизация текста

```
[12] # Векторизация с помощью CountVectorizer
     count_vect = CountVectorizer()
     X_train_counts = count_vect.fit_transform(X_train)
     X_test_counts = count_vect.transform(X_test)

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

Выполним оценку точности двух моделей машинного обучения (RandomForestClassifier и LogisticRegression) на двух разных типах векторизованных данных (CountVectorizer и TfidfVectorizer), полученных из текста

```
[15] def evaluate_model(vectorizer_name, vectorizer_train, vectorizer_test, model, model_name):
         start_time = time.time()
         obj_model = model
         obj_model.fit(vectorizer_train, y_train)
         predictions = obj_model.predict(vectorizer_test)

         accuracy = accuracy_score(y_test, predictions)
         duration = (time.time() - start_time) / 60

         print(f'Точность {vectorizer_name} + {model_name}: {accuracy:.4f}, время обучения классификатора: {duration:.2f} мин.', )

     # Для CountVectorizer
     evaluate_model('CountVectorizer', X_train_counts, X_test_counts, RandomForestClassifier(), 'RandomForestClassifier')
     evaluate_model('CountVectorizer', X_train_counts, X_test_counts, LogisticRegression(max_iter=1000), 'LogisticRegression')

     # Для TfidfVectorizer
     evaluate_model('TfidfVectorizer', X_train_tfidf, X_test_tfidf, RandomForestClassifier(), 'RandomForestClassifier')
     evaluate_model('TfidfVectorizer', X_train_tfidf, X_test_tfidf, LogisticRegression(max_iter=1000), 'LogisticRegression')
```

```
Точность CountVectorizer + RandomForestClassifier: 0.7791, время обучения классификатора: 0.28 мин.
Точность CountVectorizer + LogisticRegression: 0.7794, время обучения классификатора: 0.03 мин.
Точность TfidfVectorizer + RandomForestClassifier: 0.7788, время обучения классификатора: 0.29 мин.
Точность TfidfVectorizer + LogisticRegression: 0.7794, время обучения классификатора: 0.03 мин.
```

Анализ качества классификации

- Точность: Точность всех моделей практически идентична - около 0.7792. Это говорит о том, что задача не очень сложная, или, возможно, выборка ограничена и модели переобучены.
- Время обучения: LogisticRegression работает значительно быстрее, чем RandomForestClassifier, как с CountVectorizer, так и с TfidfVectorizer.

Вывод:

В данном случае LogisticRegression с CountVectorizer или TfidfVectorizer показывает практически одинаковое качество, но при этом работает существенно быстрее.

В конечном итоге выбор лучшего варианта зависит от приоритетов:

- Если скорость важнее всего: Логистическая регрессия с CountVectorizer.
- Если интерпретируемость важна: Логистическая регрессия с CountVectorizer.
- Если необходимо попробовать более сложную модель: RandomForestClassifier с CountVectorizer или TfidfVectorizer.