

Выполнил: ЧжаоЛян

Группа: ИУ5И-22М

## Random Forest Classifier

## Complement Naive Bayes

```
In [1]: import numpy as np
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import ComplementNB
```

```
In [6]: df = pd.read_csv(r'C:\Users\80667\Desktop\文件\ИУ5\研一下\MM0\lab\lab1\数据集\StudentsPerformance.csv')
df
```

Out[6]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75
...	...	...	...	...	...	...	...	...
995	female	group E	master's degree	standard	completed	88	99	95
996	male	group C	high school	free/reduced	none	62	55	55
997	female	group C	high school	free/reduced	completed	59	71	65
998	female	group D	some college	standard	completed	68	78	77
999	female	group D	some college	free/reduced	none	77	86	86

1000 rows x 8 columns

## Предобработка признаков

### TFIDF

TFIDF 

```
In [7]: tfidf = TfidfVectorizer()
tfidf_ngram_features = tfidf.fit_transform(df['parental level of education'])
tfidf_ngram_features
```

Out[7]: <1000x8 sparse matrix of type '<class 'numpy.float64'>' with 2179 stored elements in Compressed Sparse Row format>

### CountVectoriser

## CountVectoriser

```
In [8]: countvec = CountVectorizer()
countvec_ngram_features = countvec.fit_transform(df['parental level of education'])
countvec_ngram_features
```

```
Out[8]: <1000x8 sparse matrix of type '<class 'numpy.int64'>'
        with 2179 stored elements in Compressed Sparse Row format>
```

## Random Forest Classifier

### Random Forest Classifier

```
In [9]: # TFIDF + RFC
X_train, X_test, y_train, y_test = train_test_split(tfidf_ngram_features, df['gender'], test_size=0.3, random_state=42)
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred, digits=4, target_names=list(map(str, list(y_test.unique())))))
```

	precision	recall	f1-score	support
male	0.5224	0.4118	0.4605	170
female	0.3976	0.5077	0.4459	130
accuracy			0.4533	300
macro avg	0.4600	0.4597	0.4532	300
weighted avg	0.4683	0.4533	0.4542	300

```
In [10]: # CountVec + RFC
X_train, X_test, y_train, y_test = train_test_split(countvec_ngram_features, df['gender'], test_size=0.3, random_state=42)
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred, digits=4, target_names=list(map(str, list(y_test.unique())))))
```

	precision	recall	f1-score	support
male	0.5224	0.4118	0.4605	170
female	0.3976	0.5077	0.4459	130
accuracy			0.4533	300
macro avg	0.4600	0.4597	0.4532	300
weighted avg	0.4683	0.4533	0.4542	300

## Complement Naive Bayes

## Complement Naive Bayes

```
In [11]: # TFIDF + CNB
X_train, X_test, y_train, y_test = train_test_split(tfidf_ngram_features, df['gender'], test_size=0.3, random_
model = ComplementNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred, digits=4, target_names=list(map(str, list(y_test.unique())))))
```

	precision	recall	f1-score	support
male	0.5500	0.2588	0.3520	170
female	0.4273	0.7231	0.5371	130
accuracy			0.4600	300
macro avg	0.4886	0.4910	0.4446	300
weighted avg	0.4968	0.4600	0.4322	300

```
In [12]: # CountVec + CNB
X_train, X_test, y_train, y_test = train_test_split(countvec_ngram_features, df['gender'], test_size=0.3, rand
model = ComplementNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred, digits=4, target_names=list(map(str, list(y_test.unique())))))
```

	precision	recall	f1-score	support
male	0.5500	0.2588	0.3520	170
female	0.4273	0.7231	0.5371	130
accuracy			0.4600	300
macro avg	0.4886	0.4910	0.4446	300
weighted avg	0.4968	0.4600	0.4322	300

## Выводы:

CountVectorizer с Random Forest Classifier показал лучшие результаты, чем TFIDF, а с Complement Naive Bayes оба векторизатора показали одинаковые результаты. Random Forest Classifier показал лучшие, чем Complement Naive Bayes результаты.