Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине «Прикладные интеллектуальные системы и экспертные системы»

Классификация текстовых данных

Студент Бахмутский М.В.

Группа М-ИАП-22

Руководитель Кургасов В.В.

Липецк 2022 г.

Цель работы

Получить практические навыки решения задачи классификации текстовых данных в среде Jupiter Notebook. Научиться проводить предварительную обработку текстовых данных, настраивать параметры методов классификации и обучать модели, оценивать точность полученных моделей.

Задание кафедры

1) Загрузить выборки по варианту из лабораторной работы №2

2) Используя GridSearchCV произвести предварительную обработку данных и настройку методов классификации в соответствие с заданием, вывести оптимальные значения параметров и результаты классификации модели (полнота, точность, f1-мера и аккуратности) с данными параметрами. Настройку проводить как на данных со стеммингом, так и на данных, на которых стемминг не применялся.

3) По каждому пункту работы занести в отчет программный код и результат вывода.

4) Оформить сравнительную таблицу с результатами классификации различными методами с разными настройками. Сделать выводы о наиболее подходящем методе классификации ваших данных с указанием параметров метода и описанием предварительной обработки

Вариант 1

Вариант	**Методы**
1	KNN, RF, LR

Ход работы

1) Загрузить выборки по варианту из лабораторной работы №2

- pandas - предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временными рядами.

- numpy - поддерживает многомерные массивы, высокоуровневые математические функций, предназначенные для работы с многомерными массивами

- pyplot - это коллекция функций в стиле команд, которая позволяет использовать matplotlib почти так же, как MATLAB

- nltk - пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python.

- sklearn - включает все алгоритмы и инструменты, которые нужны для задач классификации, регрессии и кластеризации, методы оценки производительности модели машинного обучения.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.pipeline import Pipeline
from sklearn.naive_bayes import MultinomialNB
from nltk.stem import *
from nltk import word_tokenize
import itertools
```

Рисунок 1 – Необходимые библиотеки

```
categories = ['comp.graphics', 'comp.os.ms-windows.misc', 'rec.autos']
remove = ['headers', 'footers', 'quotes']
twenty_train = fetch_20newsgroups(subset='train', shuffle=True, random_state=42, categories=categories, remove=remove)
twenty_test = fetch_20newsgroups(subset='test', shuffle=True, random_state=42, categories=categories, remove=remove)
```

Рисунок 2 – Выгрузка данных по варианту

2) Используя GridSearchCV произвести предварительную обработку данных и настройку методов классификации в соответствие с заданием,

вывести оптимальные значения параметров и результаты классификации модели (полнота, точность, f1-мера и аккуратности) с данными параметрами. Настройку проводить как на данных со стеммингом, так и на данных, на которых стемминг не применялся.

```
parameters = {
    'RandomForestClassifier': {
        'vect__max_features': (1000,5000,10000),
        'vect__stop_words': ('english', None),
        'tfidf__use_idf': (True, False),
        'clf__criterion': ['gini','entropy','log_loss'],
        'clf__max_depth': [3,5,10,None]
    },
    'LogisticRegression': {
        'vect__max_features': (1000,5000,10000),
        'vect__stop_words': ('english', None),
        'tfidf__use_idf': (True, False),
        'clf__penalty': ['l1','l2'],
        'clf__C': [0.001,0.01,0.1,1,10,100,1000]
    },
    'KNeighborsClassifier': {
        'vect__max_features': (1000,5000,10000),
        'vect__stop_words': ('english', None),
        'tfidf__use_idf': (True, False),
        'clf__n_neighbors': (1, 3, 5, 10),
        'clf__p': (1, 2)
    }
}

gs = {}
for clf, param in parameters.items():
    text_clf = Pipeline([
        ('vect', CountVectorizer()),
        ('tfidf', TfidfTransformer()),
        ('clf', eval(clf)())
    ])
    gs[clf] = GridSearchCV(text_clf, param, n_jobs=-1, error_score=0.0)
    gs[clf].fit(X = twenty_train['data'], y = twenty_train['target'])
```

Рисунок 3 – Сетки параметрического поиска

На данном рисунке представлено параметры и ограничения по которым будет проводится поиск по сетке

3) Оформим сравнительную таблицу с результатами классификации различными методами.

	precision	recall	f1-score	support
comp.graphics	0.83	0.74	0.78	389
comp.os.ms-windows.misc	0.83	0.76	0.79	394
rec.autos	0.79	0.94	0.86	396
accuracy			0.81	1179
macro avg	0.82	0.81	0.81	1179
weighted avg	0.82	0.81	0.81	1179

	precision	recall	f1-score	support
comp.graphics	0.84	0.85	0.84	389
comp.os.ms-windows.misc	0.90	0.74	0.81	394
rec.autos	0.84	0.97	0.90	396
accuracy			0.85	1179
macro avg	0.86	0.85	0.85	1179
weighted avg	0.86	0.85	0.85	1179

	precision	recall	f1-score	support
comp.graphics	0.59	0.35	0.44	389
comp.os.ms-windows.misc	0.61	0.37	0.46	394
rec.autos	0.46	0.82	0.59	396
accuracy			0.51	1179
macro avg	0.55	0.51	0.49	1179
weighted avg	0.55	0.51	0.49	1179

Рисунок 4 — Итоговая таблица

Из полученных данных мы видим, что наилучшую классификацию показал логистический регрессионный классификатор с вероятностью 0,85

Вывод

В ходе выполнения данной лабораторной работы были получены практические навыки решения задачи классификации текстовых данных в среде Jupiter Notebook. Научились проводить предварительную обработку текстовых данных, настраивать параметры методов классификации и обучать модели, оценивать точность полученных моделей.