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 Сухоруких А.О.

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 2

2	RF, MNB, SVM

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

Выгрузка данных из датасета

```
categories = ['comp.windows.x', 'rec.sport.baseball', 'rec.sport.hockey']
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)
```

✓ 2.2s

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.

```
%%time
parameters = {
    'RAndomForestClassifier': {
        '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)
    },
    'GaussianNB': {
        'vect__max_features': (1000,5000,10000),
        'vect__stop_words': ('english', None),
        'tfidf__use_idf': (True, False),
        'clf__criterion': ('gini', 'entropy'),
        'clf__max_depth': [*range(1,5,1), *range(5,101,20)]
    },
    'svm': [{
        'vect__max_features': (1000,5000,10000),
        'vect__stop_words': ('english', None),
        'tfidf__use_idf': (True, False),
        'clf__loss': ['squared_hinge'],
        'clf__penalty': ('l1', 'l2')
    },
        {
        'vect__max_features': (1000,5000,10000),
        'vect__stop_words': ('english', None),
        'tfidf__use_idf': (True, False),
        'clf__loss': ['hinge'],
        'clf__penalty': ['l2']
    }],
}

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'])
✓ 6m 58.8s
```

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.

`

```
    comp.windows.x       0.75      0.56      0.64       395
rec.sport.baseball       0.43      0.57      0.49       397
  rec.sport.hockey       0.51      0.48      0.49       399

          accuracy                           0.53      1191
         macro avg       0.56      0.53      0.54      1191
      weighted avg       0.56      0.53      0.54      1191

                    precision    recall  f1-score   support

    comp.windows.x       0.87      0.80      0.83       395
rec.sport.baseball       0.63      0.81      0.71       397
  rec.sport.hockey       0.82      0.64      0.72       399

          accuracy                           0.75      1191
         macro avg       0.77      0.75      0.75      1191
      weighted avg       0.77      0.75      0.75      1191

                    precision    recall  f1-score   support

    comp.windows.x       0.98      0.93      0.96       395
rec.sport.baseball       0.85      0.90      0.87       397
  rec.sport.hockey       0.90      0.88      0.89       399
...
          accuracy                           0.91      1191
         macro avg       0.91      0.91      0.91      1191
      weighted avg       0.91      0.91      0.91      1191
```

Рисунок 4 — Итоговая таблица

Из полученных данных мы видим, что наиучшую классификацию показал метод наивного байсевского классфикатора с вероятностью 0,96

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.