**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине

«Прикладные интеллектуальные системы и экспертные системы»

Студент Первушин О. С.

Группа М-ИАП-23-1

Руководитель Кургасов В. В.

доцент, канд. пед. наук

Липецк 2023 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

Загрузить выборки по варианту из лабораторной работы №3.

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.

Ход работы

Приведем код программы, реализующий задание (лист. 1).

Листинг 1 — Код программы

import re  
import nltk  
from nltk.corpus import stopwords  
from nltk.stem import SnowballStemmer  
from sklearn.feature\_extraction.text import TfidfVectorizer  
from sklearn.model\_selection import train\_test\_split, GridSearchCV  
from sklearn.datasets import fetch\_20newsgroups  
from sklearn.naive\_bayes import MultinomialNB  
from sklearn.metrics import accuracy\_score, classification\_report  
  
# Загрузка данных  
categories = ['comp.windows.x', 'rec.sport.baseball', 'rec.sport.hockey']  
remove = ('headers', 'footers', 'quotes')  
data = fetch\_20newsgroups(subset='all', categories=categories, remove=remove)  
  
# Предварительная обработка данных  
nltk.download('stopwords')  
stemmer = SnowballStemmer('english')  
en\_stopwords = set(stopwords.words('english'))  
  
def preprocess\_text(text):  
 # Приведение к нижнему регистру  
 text = text.lower()  
 # Удаление символов, цифр и пунктуации  
 text = re.sub(r"[^a-zA-Z]", " ", text)  
 # Токенизация  
 tokens = text.split()  
 # Удаление стоп-слов и стемминг  
 tokens = [stemmer.stem(word) for word in tokens if word not in en\_stopwords]  
 return " ".join(tokens)  
  
# Применение предварительной обработки к данным  
preprocessed\_data = [preprocess\_text(text) for text in data.data]  
  
# Разделение на тренировочный и тестовый набор  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(preprocessed\_data, data.target, test\_size=0.2, random\_state=42)  
  
# Создание TF-IDF векторов  
tfidf\_vectorizer = TfidfVectorizer()  
X\_train\_tfidf = tfidf\_vectorizer.fit\_transform(X\_train)  
X\_test\_tfidf = tfidf\_vectorizer.transform(X\_test)  
  
# Обучение модели с использованием GridSearchCV  
parameters = {'alpha': [0.1, 1.0, 10.0]}  
clf = GridSearchCV(MultinomialNB(), parameters, cv=5)  
clf.fit(X\_train\_tfidf, y\_train)  
  
print("Лучшие параметры: ", clf.best\_params\_)  
  
# Предсказание на тестовом наборе  
y\_pred = clf.predict(X\_test\_tfidf)  
  
# Оценка результатов  
print("Accuracy: ", accuracy\_score(y\_test, y\_pred))  
print(classification\_report(y\_test, y\_pred, target\_names=data.target\_names))

Полученные результаты можно оценить следующим образом (рис. 1).

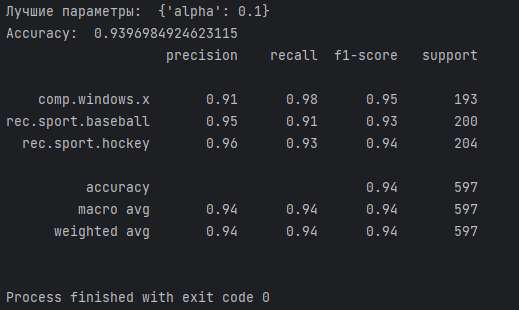


Рисунок 1 — Результаты полученные в ходе выполнения программы

Вывод

Мы получили практические навыки решения задачи классификации текстовых данных.