Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и

вычислительной техники

Отчет по командному итоговому проекту

Дисциплина: Машинное обучение и большие данные

Подготовили:

Кабанов И.А.

Калмацкий А.Н.

группа 22П-3

Уфа, 2025 год

1. Введение

В рамках данного проекта была проведена комплексная работа по обработке текстовых данных на множестве языков, тематическому моделированию, векторизации, классификации и парсингу данных из открытых источников. Целью проекта являлось создание модели для автоматического определения языка текста с высокой точностью, с последующем использованием модели в веб-приложении через REST API.

1. Импорт данных и подключение библиотек

* Использованы библиотеки: pandas, nltk, pymorphy3, sklearn, requests, BeautifulSoup, matplotlib, wordcloud и др.
* Загружен датасет TextLanguage.csv с 10337 текстовыми записями на 17 языках.
* Проведена проверка структуры данных и распределения по языкам.

1. Предварительная обработка данных

Основные этапы обработки текста:

* Приведение текста к нижнему регистру
* Удаление знаков препинания, цифр и специальных символов
* Токенизация и удаление стоп-слов для 17 языков
* Лемматизация с использованием pymorphy3 для русского и стеммеров Snowball для других языков
* Для некоторых языков (турецкий, греческий, малаялам, тамильский, каннада, хинди) применялось только приведение к нижнему регистру

Результатом стала колонка с лемматизированным текстом, готовым к векторизации.

1. Визуализация данных

* Построены облака слов для каждого языка с использованием библиотеки WordCloud.
* Для языков с нестандартными алфавитами использовались соответствующие шрифты из семейства Noto.
* Облака слов позволили визуально оценить частотные слова и особенности текстов на разных языках.

1. Векторизация текстов

* Применены два подхода:
  + Bag of Words (BoW) с использованием CountVectorizer с n-граммами (1-3) и фильтрацией по частоте
  + TF-IDF с TfidfVectorizer на уровне символов (2-5 грамм), ограничение по количеству признаков до 20000
* Получены разреженные матрицы признаков для последующего обучения моделей.

1. Классификация языков

* Использованы три модели:
  + RandomForestClassifier
  + LogisticRegression
  + SGDClassifier с логистической регрессией (log\_loss) и балансировкой классов
* Данные разделены на обучающую и тестовую выборки (60/40).
* Метрики качества (precision, recall, f1-score) для каждой модели:

|  |  |  |
| --- | --- | --- |
| Модель | Точность (accuracy) | Среднее F1 (macro avg) |
| RandomForestClassifier | 0.92 | 0.92 |
| LogisticRegression | 0.95 | 0.95 |
| SGDClassifier | 0.96 | 0.96 |

* Лучшие результаты показал SGDClassifier с точностью 96%.
* Построены матрицы ошибок (confusion matrix) для визуализации ошибок классификации.

1. Парсинг данных из открытых источников

* Собраны тексты с сайтов:
  + habr.com (русскоязычные статьи)
  + reddit.com (тематические сабреддиты на разных языках)
* Получено 462 новых текста для расширения обучающей выборки.
* Проведена предварительная обработка и классификация новых данных с использованием обученной модели SGDClassifier.
* Использован итеративный подход с отбором предсказаний с высокой уверенностью (вероятность > 0.8 или < 0.2) для добавления в обучающую выборку и дообучения модели.
* Итоговая классификация новых данных показала высокую точность и позволила расширить датасет.

1. Экспорт модели

* Модель SGDClassifier и векторизатор TfidfVectorizer сохранены в файлы sgdclf\_fp.pkl и vectorizer\_fp.pkl соответственно с помощью pickle.
* Это позволяет использовать обученную модель в дальнейшем через API или другие приложения.

1. Создание API

Для создания REST API был использован язык программирования Python с фреймворком FastApi. Было сделано импортирование ранее созданных моделей классификации и векторизации, а также методы по обработке получаемого текста для модели.

Был создан POST запрос /predict который на вход получает текст и возвращает язык текста.

1. Создание веб-приложения

Для создания веб-приложения был использован язык программирования JavaScript с реактивным фреймворком Vue.js, что позволило добавить дополнительные функции приложения, не требующие перезапуска.

Использованы библиотеки: axios – для http-запросов к API; vue-router – для многостраничности; vue-i18n – для мультиязычности.

Были созданы 3 вкладки: “Модель” – на которой можно отправить запрос и узнать, какой язык текста. Также поддерживается голосой ввод; “Статические данные” – на которой можно узнать подробную статистику изначального датасета для модели. Для виджетов был использован сервис Yandex DataLens; “Как использовать?” – на которой можно ознакомится с основными функциями веб-приложения, а также информацию об авторах.

1. Написание документации для REST API

Была написана документация для REST API, в которой подробно расписано основные методы, ошибки, входная и выходная информация.

1. Выводы

* Проведена успешная обработка многоязычного текстового датасета с применением современных методов NLP.
* Лемматизация и удаление стоп-слов значительно улучшили качество признаков.
* Векторизация с TF-IDF на уровне символов показала высокую эффективность.
* Модель SGDClassifier продемонстрировала наилучшие результаты по точности и полноте.
* Итеративный подход к разметке новых данных с использованием модели позволил расширить обучающую выборку и повысить качество классификации.
* Экспорт модели обеспечивает возможность ее интеграции в реальные приложения.
* Создан REST API на Python с использованием фреймворка FastApi.
* Создано мультиязычное веб-приложение на Vue.js.
* Написана подробная документация для REST API.