ЛАБОРАТОРНАЯ РАБОТА №2

Тема лабораторной работы: работа с классификацией видов тестирования.

**a. Цель работы**

Цель данной работы заключается в проведении детального анализа и составлении всестороннего отчета по различным аспектам тестирования программного обеспечения, выбранного в качестве объекта исследования. В рамках этого анализа будет рассмотрена классификация тестирования, определены его виды и типы, а также описаны используемые методы и уровни тестирования, применимые к конкретному проекту.

**b. Описание программного проекта**

Проект представляет собой систему машинного обучения для нормализации слов текста, написанную на языке Python. Используемые библиотеки включают NumPy для работы с массивами, Pandas для обработки данных, Scikit-learn для машинного обучения, NLTK для задач обработки естественного языка, а также Matplotlib и Seaborn для визуализации данных.

Основная модель проекта — градиентный бустинг, реализованный с помощью класса GradientBoostingClassifier из библиотеки Scikit-learn. Программа выполняет сбор и подготовку данных, обработку текста (лемматизация, стемминг), обучение моделей, оценку их производительности и визуализацию результатов.

Визуализация включает сравнение метрик производительности, что позволяет наглядно представить результаты работы моделей и улучшить процесс нормализации текста.

**c. Классификация тестирования для проекта**

Для анализа классификации тестирования используется следующая структура:

1. **Виды тестирования:**

* **Функциональное тестирование:** Проверка основных функций приложения, таких как предобработка текстов, нормализация слов, обучение модели и предсказание результатов.
* **Нефункциональное тестирование:** Проверка производительности, надежности и удобства использования программы
* **Регрессионное тестирование:** Проверка корректности работы программы после внесения изменений в код.

1. **Типы тестирования:**

* **Модульное тестирование:** Тестирование отдельных функций и методов, таких как текстовая предобработка, стемминг и лемматизация.
* **Интеграционное тестирование:** Тестирование взаимодействия между модулями, например, проверка взаимодействия между этапами предобработки, нормализации и обучения модели.
* **Системное тестирование:** Тестирование всей системы в целом.

1. **Методы тестирования:**

* **Черный ящик:** Тестирование функциональности приложения без знания внутренней структуры кода.
* **Белый ящик:** Тестирование внутренних структур и алгоритмов приложения, например, проверка корректности работы алгоритма градиентного бустинга.
* **Серый ящик:** Комбинация методов черного и белого ящика для комплексного тестирования.

1. **Уровни тестирования:**

* **Низкоуровневое тестирование:** Модульное тестирование отдельных функций и методов.
* **Среднеуровневое тестирование:** Интеграционное тестирование взаимодействия между модулями.
* **Высокоуровневое тестирование:** Системное тестирование всего приложения.

**d.Примеры дефектов, характерных для каждого вида тестирования**

1. **Функциональное тестирование:**

* Ошибка в предобработке текста: данные обрабатываются некорректно или частично.
* Ошибка в нормализации слов: слова не приводятся к стандартным формам или нормализуются неправильно.
* Ошибка в обучении модели: модель не обучается или выдает неверные результаты при предсказании.

1. **Нефункциональное тестирование:**

* Ошибка производительности: приложение работает медленно при обработке больших объемов текста.
* Ошибка надежности: приложение падает при обработке некорректных данных.
* Ошибка удобства использования: интерфейс сложен для понимания или использования.

1. **Регрессионное тестирование:**

* Ошибка после изменения кода: внесенные изменения ломают ранее работавшие функции или алгоритмы.

**e. Выводы по работе**

Проведенный анализ позволяет выделить ключевые направления для тестирования выбранного программного проекта. Особое внимание следует уделить функциональному тестированию основных операций, таких как предобработка текстов и нормализация слов, а также нефункциональным аспектам, включая производительность и удобство использования.

Регрессионное тестирование играет важную роль в обеспечении стабильности приложения после внесения изменений в код. Применение различных типов и уровней тестирования позволит достичь более полного охвата тестами и выявить потенциальные дефекты на ранних этапах разработки.

**f. Список использованных источников**

1. Документация по работе с библиотекой Pandas. URL: <https://pandas.pydata.org/>
2. Документация по работе с библиотекой NumPy. URL: <https://numpy.org/>
3. Документация по работе с библиотекой matplotlib.pyplot. URL: <https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html>