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 прогнозирования оттока клиентов, 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.

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 алгоритмах обучения и предсказания:** Неправильная работа моделей машинного обучения, такие как неверная классификация клиентов, что отражается на точности предсказаний.
* 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 в производительности:** Долгое время обучения модели или предсказания может свидетельствовать о низкой производительности программы.
* **Ошибка надежности:** Возможны сбои в работе программы при обработке больших объемов данных или при высоких нагрузках.
* **Ошибка удобства использования**: Сложность интерфейса или неудобство взаимодействия с программой для конечного пользователя.

1. **Регрессионное тестирование:**

* **Ошибка результатов после изменений в коде:** После внесения новых изменений, программа может показывать некорректные результаты, что указывает на регрессию в функциональности.
* **Ошибка несоответствия в данных:** Возможные ошибки в данных после их обновления или внесения новых записей могут приводить к неправильным результатам.

**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, надежность и удобство использования. Регрессионное тестирование играет важную роль в обеспечении стабильности приложения после внесения изменений в код. Применение различных типов (модульное, интеграционное, системное) и уровней (низкоуровневое, среднеуровневое, высокоуровневое) 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>