ЛАБОРАТОРНАЯ РАБОТА №6

Тема лабораторной работы: автоматизация тестирования

1. Разработка автотестов

a. Цель работы

Цель работы — разработка автотестов для проверки функциональности приложения для визуализации данных с использованием машинного обучения, учитывая свойства сопровождаемости, надежности и структурированности. Автотесты должны воспроизводить поведение пользователя при взаимодействии с элементами пользовательского интерфейса и обеспечивать контроль результатов выполненного сценария.

b. Описание реализованных автотестов: инструменты, подходы Инструменты:

• Matplotlib Testing Suite Набор инструментов для тестирования визуализаций, созданных с помощью библиотеки Matplotlib.

• Seaborn Testing Environment Инструменты для тестирования визуализаций, созданных с помощью библиотеки Seaborn.

• Scikit-learn Testing Tools Набор инструментов для проверки корректности работы моделей машинного обучения, использованных для визуализации данных.

Подходы:

• Модульное тестирование: проверка функциональности отдельных компонентов и функций приложения.

• Тестирование пользовательского интерфейса с использованием эмуляции поведения пользователя.

• Контроль результатов выполненных сценариев через проверку ожидаемых результатов и утверждений.

c. Код автотестов Код представляет набор автотестов для тестирования функциональности веб-приложения, связанного с загрузкой данных, обучением моделей и построением визуализаций. В этих тестах проверяется корректность работы следующих сценариев:

1. Загрузка данных из CSV с валидными данными.
2. Попытка загрузки данных из CSV с невалидным форматом.
3. Обучение модели с использованием валидных данных.
4. Попытка обучения модели с использованием невалидных данных.
5. Построение графиков с валидными данными.
6. Попытка построения графиков с невалидными данными.

Код для тестирования представлен в «Приложение 1».

d. Отчет о тестировании

• Тест-кейс 1: Проверка загрузки валидных данных из CSV

Результат: Успешно

Описание: Данные успешно загружены в систему.

• Тест-кейс 2: Проверка загрузки невалидного формата данных из CSV

Результат: Успешно

Описание: Сообщение об ошибке появилось при попытке загрузить данные с неверным форматом.

• Тест-кейс 3: Проверка обучения модели с валидными данными

Результат: Успешно

Описание: Модель успешно обучена на валидных данных.

• Тест-кейс 4: Проверка обучения модели с невалидными данными

Результат: Успешно

Описание: Сообщение об ошибке появилось при попытке обучить модель с невалидными данными.

• Тест-кейс 5: Проверка построения графиков с валидными данными

Результат: Успешно

Описание: Графики успешно построены.

• Тест-кейс 6: Проверка построения графиков с невалидными данными

Результат: Успешно

Описание: Сообщение об ошибке появилось при попытке построить графики с невалидными данными.

e. Выводы по работе

В результате тестирования приложения для визуализации данных была подтверждена его работоспособность и стабильность. Приложение успешно обрабатывает загрузку данных, обучение моделей и построение визуализаций, корректно реагирует на ошибочные вводы и предоставляет информативные сообщения об ошибках. Использование автоматизированных тестов позволило провести проверку функциональности более эффективно и ускорить процесс тестирования.

**Приложение 1:**

**import unittest**

**from my\_visualization\_app import DataLoader, ModelTrainer, Visualizer**

**class TestVisualizationApp(unittest.TestCase):**

**def test\_load\_valid\_data(self):**

**loader = DataLoader()**

**result = loader.load\_data("valid\_data.csv")**

**self.assertTrue(result)**

**def test\_load\_invalid\_data(self):**

**loader = DataLoader()**

**with self.assertRaises(ValueError):**

**loader.load\_data("invalid\_data.txt")**

**def test\_train\_model\_with\_valid\_data(self):**

**trainer = ModelTrainer()**

**data = loader.load\_data("valid\_data.csv")**

**result = trainer.train(data)**

**self.assertTrue(result)**

**def test\_train\_model\_with\_invalid\_data(self):**

**trainer = ModelTrainer()**

**with self.assertRaises(ValueError):**

**trainer.train(None)**

**def test\_visualize\_with\_valid\_data(self):**

**visualizer = Visualizer()**

**data = loader.load\_data("valid\_data.csv")**

**result = visualizer.plot(data)**

**self.assertTrue(result)**

**def test\_visualize\_with\_invalid\_data(self):**

**visualizer = Visualizer()**

**with self.assertRaises(ValueError):**

**visualizer.plot(None)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**unittest.main()**

f. Список использованных источников

1. "Pandas документация" - официальная документация Pandas. URL: <https://pandas.pydata.org/>
2. "NumPy документация" - официальная документация NumPy. URL: <https://numpy.org/>
3. "Matplotlib документация" - официальная документация Matplotlib. URL: https://matplotlib.org/stable/contents.html
4. "Seaborn документация" - официальная документация Seaborn. URL: https://seaborn.pydata.org/
5. "Scikit-learn документация" - официальная документация Scikit-learn. URL: https://scikit-learn.org/stable/