CHAPTER – 7
TESTING

CHAPTER 7

TESTING

7.1 System Testing

System testing plays a critical role in the implementation of our drug response prediction model, ensuring that each component functions as expected and meets project requirements for accuracy, usability, and reliability. The primary focus is to verify the effectiveness of the data preprocessing pipeline, Artificial Neural Network (ANN) model prediction, and web interface. This chapter outlines the testing methodologies, test cases, and evaluation metrics that were applied to validate the system's performance, usability, and predictive accuracy.

7.2 Testing Methodology

The system testing approach follows a structured process involving multiple stages:

- **Unit Testing**: Individual modules, including data preprocessing, model training, and prediction functions, were tested independently to verify their correctness. Unit testing helps identify and address specific issues within each component before integration.

- **Integration Testing**: After unit testing, integration testing was conducted to ensure smooth interactions between the model and the interface. This phase confirmed that data flowed accurately from the input fields through preprocessing and into the ANN model, generating predictions without errors.

- **System Testing**: The entire application was tested to evaluate the end-to-end workflow, from data input through the web interface to prediction output. This step confirmed that all components work together cohesively and support the system's goals.

- **User Acceptance Testing (UAT)**: UAT assessed the web interface's usability and responsiveness from a user's perspective, ensuring it is intuitive, responsive, and provides clear IC50 predictions for researchers and clinicians.

7.3 Test Cases

The following test cases were designed to comprehensively evaluate system functionality, accuracy, and usability:

Table 7.1: Test Cases

Test Case ID	Description	Expected Outcome	Status
TC-1	Verify data loading and preprocessing without errors	Data loads and preprocesses correctly	Passed
TC-2	Validate feature selection and encoding of categorical features	Features are selected and encoded accurately	Passed
TC-3	Check model training with selected hyperparameters	Model trains successfully and achieves stable loss reduction	Passed
TC-4	Assess model prediction accuracy on test data	Model accurately predicts IC50 values with acceptable MSE and R^2	Passed
TC-5	Test data input and prediction through Flask interface	User inputs data and receives IC50 predictions in real time	Passed
TC-6	Ensure dropdown selection and field validation in interface	Only valid inputs are accepted, with dropdowns functioning correctly	Passed
TC-7	Validate error handling for missing or invalid inputs	User receives clear feedback on invalid entries	Passed
TC-8	Confirm prediction display in the user interface	Predicted IC50 values display correctly without delay	Passed
TC-9	Verify compatibility on different web browsers	Interface functions properly on major browsers (Chrome, Firefox, Edge)	Passed

7.4 Evaluation Metrics

The model's performance was assessed with specific metrics to ensure its predictive accuracy and generalizability:

- **Mean Squared Error (MSE)**: MSE quantifies the average squared difference between predicted and actual IC50 values. A lower MSE reflects improved accuracy, which is essential for reliable predictions.
- **R-squared (R^2)**: R^2 measures the proportion of variance in IC50 values explained by the model. A higher R^2 score indicates that the model effectively captures the relationship between genomic features and drug response.
- **Response Time**: The system's response time is monitored to ensure real-time predictions. A response time below 2 seconds is the target, supporting clinical and research usability.
- **User Feedback and Usability**: During UAT, feedback on the web interface's usability was collected. Positive feedback on ease of use and clear output display indicates readiness for practical deployment.

7.5 Test Results and Analysis

The testing process confirmed that each component functions as expected and that the model achieves satisfactory predictive accuracy. Key observations include:

- **Accurate Predictions**: The model demonstrated low MSE and high R^2, confirming strong predictive performance on test data. These metrics validate the model's ability to generalize effectively, providing reliable IC50 predictions for practical application.
- **Responsive Web Interface**: The Flask-based interface consistently returned predictions within the target response time. Input fields and dropdowns functioned correctly, and error-handling mechanisms enhanced the user experience by guiding valid input entries.
- **Cross-Browser Compatibility**: The interface was tested on major web browsers, including Chrome, Firefox, and Edge, ensuring consistent performance and accessibility across platforms.

- **Error Handling and Validation**: User feedback highlighted the intuitive design of the interface and effective prompts for incorrect input. These validation features ensure that users enter valid data for accurate predictions, improving the system's robustness.

7.6 Challenges and Resolutions

The following challenges were encountered during testing, with corresponding resolutions:

- **Data Variability**: Initial data preprocessing revealed inconsistencies in the dataset, including missing values and scaling discrepancies. These issues were addressed by implementing imputation techniques for missing data and standardizing continuous variables.

- **Prediction Accuracy**: Achieving high accuracy required multiple rounds of hyperparameter tuning, particularly for learning rate and batch size. Fine-tuning these parameters optimized model performance, reducing MSE and enhancing prediction accuracy.

- **Interface Optimization**: Early versions of the web interface experienced minor issues with responsiveness and validation. Additional testing and adjustments to the HTML and CSS improved functionality, resulting in a more intuitive and user-friendly experience.

- **Deployment Compatibility**: Testing the application on various browsers required adjustments in design and server configuration to ensure accessibility across platforms. Compatibility enhancements increased the system's flexibility and reliability.