College of Software

Course: Open-Source Software

May 29th, 2025.

BREAST CANCER PREDICTOR WEB APP PROJECT REPORT

Submitted By

JUSU ABDUL KARIM	ID 2120246025
LAGGAH JULIUS	ID: 2120246034
SANDAR WIN	ID: 2120246058
TURAY ADAMSAY	ID: 2120246055

1. ABSTRACT

Breast cancer is among the most prevalent and life-threatening diseases affecting women worldwide. Early diagnosis plays a crucial role in improving treatment outcomes and survival rates. This project presents the development of a machine learning-powered web application, titled Breast Cancer Predictor Web Application Using Streamlit, designed to provide accessible, accurate, and user-friendly breast cancer risk predictions. The application utilizes the well-known Breast Cancer Wisconsin dataset and the Breast Cancer DICOM images dataset, respectively, to train a Logistic Regression model, which forms the core of the predictive engine.

The web application supports both single-instance predictions, where users can input diagnostic measurements manually via sliders, batch predictions through CSV file uploads, and DICOM file uploads, enabling scalable analysis. The system incorporates user authentication features to safeguard access, employing a SQLite3-based login and registration module for secure user management. To enhance user experience and trust, the interface is styled with custom CSS, including a dark theme, and incorporates dynamic visualizations such as radar charts to represent key diagnostic features.

Beyond the prediction output, the app integrates explainability tools like SHAP (SHapley Additive exPlanations) to provide users with insights into feature importance, promoting transparency and interpretability in decision-making. The backend leverages robust Python libraries including Scikit-learn for model training, Pandas and NumPy for data handling, and Matplotlib for additional visualizations.

Addressing the need for lightweight, cost-effective diagnostic support tools, the app assists healthcare professionals and patients in early cancer detection without reliance on expensive equipment. The complete source code and documentation are publicly available on GitHub at

https://github.com/juliuslaggah/breast-cancer-predictor.git

2. INTRODUCTION

The rapid advancement and accessibility of machine learning offer promising opportunities to develop affordable, scalable, and accessible diagnostic support tools. Machine learning models can analyze complex biomedical data, enabling clinicians to make faster and more accurate decisions. Motivated by this potential, this project focuses on designing and implementing a web-based Breast Cancer Predictor using Streamlit, a Python framework for creating interactive web applications. The application utilizes a Logistic Regression model trained on the Breast Cancer Wisconsin and DICOM images dataset. The web application empowers users to input diagnostic features individually through an intuitive slider interface or upload multiple samples via CSV and DICOM files for prediction. It provides clear classification results, benign or malignant, alongside probability scores and visual explanations, such as radar charts and SHAP (SHapley Additive exPlanations) values, to foster transparency and user trust. Moreover, the system includes user authentication to ensure secure access, enhancing suitability for clinical or research use. This project is developed as an open-source and is available under the GPL V.3 License, which encourages reproducibility and collaborative development.

3. PROBLEM STATEMENT

Breast cancer is one of the leading causes of death among women worldwide, with early detection being critical for effective treatment and improved survival rates. However, traditional diagnostic methods such as mammography and biopsy can be costly, invasive, and require specialized medical expertise, limiting accessibility in low-resource settings. Additionally, the growing volume of patient data challenges healthcare providers to quickly and accurately interpret diagnostic information. There is an urgent need for lightweight, accessible, and reliable digital tools that can assist clinicians and patients in the early detection of breast cancer. Current machine learning-based diagnostic applications often lack comprehensive functionality, such as secure user authentication, batch processing, and interpretable visual explanations. This project addresses these gaps by developing a web-based Breast Cancer Predictor using Streamlit, integrating a robust Logistic Regression model, interactive visualizations, and enhanced security features. The goal is to provide an easy-to-use, scalable platform that supports timely and accurate breast cancer risk assessment.

4. OBJECTIVES

The primary objective of this project is to develop a web-based Breast Cancer Predictor that assists healthcare professionals in diagnosing breast cancer quickly and accurately. The specific goals and objectives of the project are as follows:

Develop an Interactive Web Application: Create a user-friendly web application using Streamlit to provide clinicians with a simple interface to input or upload diagnostic data and receive real-time predictions.

Implement Machine Learning Model: Train a Logistic Regression model on the Breast Cancer Wisconsin and DICOM image dataset to segment and classify tumor samples as benign or malignant based on input or embedded features.

Support Image Uploads, Single and Batch Predictions: Allow users to input diagnostic measurements manually via sliders or upload multiple data points in CSV format for batch predictions or upload image files by providing the image directory.

Provide Transparent Results with Visual Explanations: Display prediction results with probability scores and use SHAP values and radar charts to explain the model's decision-making process, enhancing transparency and user trust.

Incorporate User Authentication: Implement a secure login system using SQLite3 for user authentication, ensuring access control and data security.

Ensure Accessibility and Scalability: Deploy the application on GitHub for public access, enabling scalability and collaboration for future improvements.

5. LITERATURE REVIEW

The inspiration for this project originated from the popular YouTube channel Alejandro_Ao, who provides an excellent tutorial on creating a breast cancer diagnosis web application using Python and Streamlit. Alejandro's project focuses on building a machine learning-powered dashboard that predicts whether a cell cluster is benign or malignant using the Breast Cancer Wisconsin dataset. It features an interactive interface with radar chart visualizations and probability outputs based on user input. While this project is an invaluable learning resource and demonstrates core concepts effectively, it has notable limitations in scope and functionality. For example, it relies primarily on Python and CSS, with no authentication or batch processing, nor DICOM breast cancer images capabilities.

To address these gaps, our project expands the original idea by integrating multiple advanced functionalities, such as user authentication, batch prediction via CSV uploads, DICOM image integration, and explainability tools like SHAP are incorporated, making the application more robust and practical for real-world usage.

Similar projects have been undertaken in the broader research and development community. For instance, Kumar et al. (2021) developed a breast cancer prediction system that utilizes an ensemble of machine learning algorithms integrated within a web framework, enabling users to upload patient data and receive predictions along with explanations of feature importance. Their work emphasized model accuracy and interpretability, similar to our use of SHAP values

6. METHODOLOGY

a. Dataset Description and Preprocessing

The Breast Cancer Wisconsin (Diagnostic) and the DICOM breast cancer dataset are publicly available and widely used for machine learning tasks involving breast cancer classification. The wisconsin dataset consists of 569 instances of breast cancer data, each with 30 numerical features that represent various characteristics of cell nuclei present in digitized breast mass images. These features include measurements like radius, texture, perimeter, area, smoothness, compactness, concavity, and fractal dimension, which are critical for distinguishing between benign and malignant tumors, whiles the DICOM dataset contains 106 patients and 746 breast cancer tumor images that contains Volume (cm^3), Mean Area (px), and Surface Area (cm^2) that serves as metrics for predictions through the Otsu threshold

The target variable is Diagnosis, where:

- M stands for malignant tumors, coded as 1,
- B stands for benign tumors, coded as 0.

Data Preprocessing

To prepare the data for model training, the following preprocessing steps were carried out:

Removing Irrelevant Columns: The dataset includes non-informative columns such as id and Unnamed: 32, which were removed to streamline the data.

Label Encoding: The Diagnosis column, originally containing categorical values ('M' and 'B'), was converted into numeric binary values. Malignant cases were encoded as 1 and benign as 0.

Feature Scaling: Since the dataset contains features with varying scales (e.g., radius versus area), it was necessary to normalize the data. We used StandardScaler from scikit-learn to standardize each feature, ensuring that the data has zero mean and unit variance. This step is crucial for ensuring that the machine learning model is not biased toward features with larger numerical ranges.

Train-Test Split: The dataset was divided into training and testing sets using an 80:20 split to evaluate the model's performance on unseen data. Stratified sampling was used to maintain class distribution in both subsets.

These preprocessing steps ensured that the dataset was clean, consistent, and ready for model training.

b. Model Development and Evaluation

The core of the Breast Cancer Predictor Web Application is the machine learning model that classifies breast cancer tumors as either Benign or Malignant. For this project, a Logistic Regression model was chosen due to its simplicity and wrapped around calibration to improve confidence rate, interpretability, and effectiveness in binary classification tasks. Logistic Regression provides a probabilistic interpretation, making it suitable for applications where understanding the likelihood of a benign or malignant diagnosis is crucial for clinicians.

Model Training

For the development of this project, two (2) distinct models were trained. **model_v3.pkl** and **scaler_v3.pkl** were trained on the Breast Cancer Wisconsin dataset, which was preprocessed as described in the previous section. The dataset was split into training and testing sets in an 80:20 ratio, ensuring that the training set was large enough to provide meaningful model insights while reserving a portion for testing the model's generalization ability. **Model_v2.pkl** and **scaler_v2.pkl** were trained on the DICOM image and the Wisconsin dataset combined to enhance interoperability and model improvement.

Model Evaluation

After training both models on the scaled training data, it was evaluated on the test set. Several performance metrics were used to assess the model's effectiveness in predicting breast cancer tumors:

- Accuracy: The model achieves an accuracy of 96% for both Benign and Malignant probability
- Precision: The proportion of true positives (correct malignant predictions) out of all predicted positives.
- Recall: The proportion of true positives out of all actual malignant cases.
- F1-score: The harmonic means of precision and recall, providing a balanced measure of the model's performance on 97% and 95% respectively.

The model's performance on the test data was evaluated using scikit-learn's classification_report, which generates a detailed performance summary:

In summary, the Logistic Regression model achieved an accuracy of 96.5%, with a precision of 97% and recall of 95% for malignant tumor detection, indicating that the model is highly reliable for predicting breast cancer.

The Logistic Regression model performed well on the dataset, providing accurate predictions with good precision and recall. These results validate its use as the foundation of the Breast Cancer Predictor Web Application.

c. System Design and Implementation

The Breast Cancer Predictor Web Application was designed with a focus on simplicity, usability, and scalability. The system integrates multiple components, including data processing, machine learning model prediction, user authentication, and interactive visualizations, all presented in a web interface using **Streamlit**.

i. User Authentication

To ensure secure access to the application, a user authentication system was implemented using **Streamlit and json file handling**. This system allows users to register, log in, and maintain active sessions throughout their interaction with the app. The user data (username and password) is stored in a lightweight json database with encryption, which ensures security and integrity. Users can also log out after completing their tasks, thus preventing unauthorized access to sensitive features.

ii. Interactive User Interface

The front-end of the application is built using **Streamlit**, providing a fast and efficient way to build interactive web apps in Python. Users can input the diagnostic measurements manually via sliders or upload a CSV file containing multiple records for batch prediction. The interface also provides real-time visual feedback, including interactive **radar charts** generated using **Plotly**, which helps users interpret the input data and prediction results visually.

iii. Data Processing Pipeline

Once user input is received, the data is validated and scaled using a pre-trained **StandardScaler**. The **Logistic Regression model** is then loaded from a pickle file and used to predict whether a sample is benign or malignant. The prediction results, along with probability scores, are displayed on the user interface. The system also includes an integrated **SHAP** explanation feature, which provides insights into how each feature contributed to the prediction.

iv. Visualization and Explainability

Interactive radar charts are used to display the input features visually, offering a clear comparison across mean, standard error, and worst-case categories. **SHAP values** are integrated to explain the model's predictions, allowing users to understand the importance of individual features in determining the outcome.

7. REQUIREMENTS GATHERING & ANALYSIS

In the development of the Breast Cancer Predictor Web Application, thorough requirements gathering and analysis were crucial to ensure the system's functionality, scalability, and usability. The following outlines the key functional and non-functional requirements for the project.

a. Stakeholder Identification

The primary stakeholders for this project include:

- Clinical Oncologists: These professionals are the primary end-users who will rely on the system to aid in early breast cancer detection.
- Radiologists: Specialists who may use the tool in conjunction with imaging data for enhanced diagnostics.
- Data Scientists: Individuals involved in improving and updating the underlying machine learning models.
- End-Users (Clinicians): Healthcare providers who will use the web interface for inputting diagnostic features and receiving results.

b. Functional Requirements

The functional requirements of the system include:

- User Authentication: Secure login and registration features to ensure authorized access.
- Single Sample Prediction: Users can manually input diagnostic features (e.g., tumor size, texture) through an interactive interface.
- Batch and DICOM image prediction: Support for CSV and DICOM files uploads to allow batch processing of multiple breast cancer cases simultaneously.
- Model Invocation and Result Display: The application must invoke the trained machine learning model, display the prediction results (benign or malignant), and show associated probability scores.
- Visualization: Display interactive radar charts representing input features and prediction probabilities.

c. Non-Functional Requirements

Non-functional requirements address the overall quality and performance of the application:

- Performance: The system must provide predictions within 5 seconds to ensure efficient operation in clinical environments.
- Security: Passwords should be securely stored (hashed), and session-based authentication must be implemented to protect user data.
- Usability: The web interface should be intuitive, with clear error messages and easy-to-use sliders and input fields.
- Maintainability: The codebase should be modular, well-documented, and version-controlled, enabling easy updates and future enhancements.

d. Constraints

- Time: The project must be completed within the semester's timeline.
- Technology: The application must rely on open-source tools, including Python, Streamlit, and scikit-learn.
- Budget: Only free-tier platforms and tools are allowed for deployment.

e. Tools & Technologies

Frontend: Streamlit (Python-based UI framework)

Backend: Python, Scikit-learn, Pandas, NumPy

Authentication: SQLite3-based login and registration

Model: Calibrated Logistic Regression (trained on Breast Cancer Wisconsin dataset)

Visualization: Matplotlib, Radar Chart

Explainability: Confidence scores, optional SHAP integration

Image Analysis: DICOM image segmentation using K-Means clustering and OpenCV

Deployment: GitHub repository, optionally deployable on Streamlit Cloud

f. Feasibility Study

- Technical Feasibility: All required libraries and frameworks (Streamlit, scikit-learn, etc.) are open-source and widely supported.
- Operational Feasibility: Clinicians can access the tool via a web browser without needing to install complex software.
- Economic Feasibility: The use of open-source tools and free-tier cloud platforms ensures low-cost deployment and maintenance.

8. SYSTEM DESIGN

a. Architecture

The architecture of the Breast Cancer Predictor Web Application is designed to ensure scalability, performance, and security. The system follows a client-server model, where the client-side is the web interface created with Streamlit, and the server-side handles model inference and data processing. The architecture integrates several components, as outlined below.

Client-Side: Streamlit Web Interface

The client-side of the application is built using Streamlit, a Python-based framework that allows for the rapid development of interactive web applications. The Streamlit interface handles user interactions and visualizations. Users input the diagnostic data either through interactive sliders or by uploading files. The web interface is designed to be responsive, supporting both single-instance predictions and batch predictions.

Key Features:

- User Authentication: secure login and registration form.
- Single Prediction: 30 features captured via sliders.
- Batch Prediction: Upload CSV of multiple entries and get batch result.

- DICOM Image segmentation: identifies tumor region using K-means clustering.
- Model Accuracy: 97% accuracy on test data with calibrated probability output.
- Explainability: High confidence outputs, SHAP integration for local explainability output
- Radar Charts and Visualizations: Interactive charts generated using Plotly to visualize the input features and prediction results.
- Professional UI: Dark theme and elegant layout using custom css.

Server-Side: Machine Learning Model and Backend

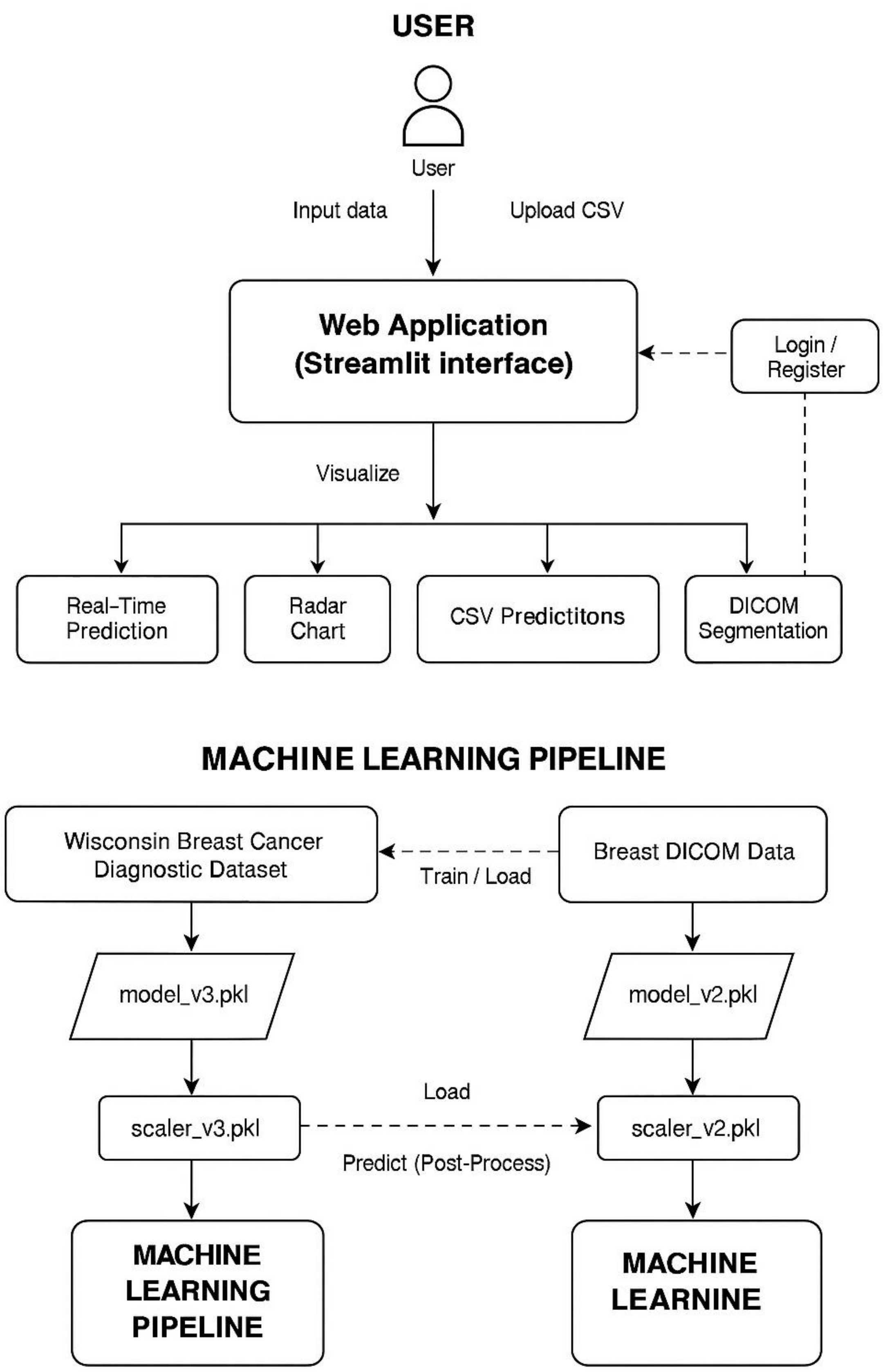
The server-side component handles the Logistic Regression model's prediction process, data preprocessing, and user authentication. The machine learning models are loaded from a Pickle file, and the input data is preprocessed and scaled before being fed into the model. Predictions are then generated and sent back to the user interface for display.

Key Features:

- Model: Logistic Regression model trained on the Breast Cancer Wisconsin dataset.
- Data Processing Pipeline: Preprocessing using StandardScaler to normalize the features before feeding them into the model.
- Prediction Logic: Predictions are made and probability scores are calculated and returned to the frontend.

Data Flow and Communication

- User Input: Data is collected via Streamlit interface (either via sliders for single inputs or uploaded files for batch predictions).
- Data Validation & Scaling: Data is validated and scaled on the server-side using StandardScaler.
- Prediction: The model predicts the tumor classification (benign or malignant) and sends back results to the client.
- Output Display: Results are displayed with interactive charts and prediction probabilities.


USER
User
Input data
Upload CSV
Web Application (Streamlit interface)
Login / Register
Visualize
Real-Time Prediction
Radar Chart
CSV Predictitons
DICOM Segmentation
MACHINE LEARNING PIPELINE
Wisconsin Breast Cancer Diagnostic Dataset
Breast DICOM Data
Train / Load
model_v3.pkl
model_v2.pkl
Load
scaler_v3.pkl
scaler_v2.pkl
Predict (Post-Process)
MACHINE LEARNING PIPELINE
MACHINE LEARNINE

Flowchart Description

The flowchart illustrates the core workflow of the Breast Cancer Predictor Web Application, detailing user interactions and internal system processes. It begins with the User Authentication module, where users register or log in to gain access. Upon successful authentication, users can proceed to the Input Stage, where they choose either Single Sample Input via interactive sliders/manual entry or Batch Input by uploading a CSV file.

In the Data Validation & Preprocessing phase, the system validates inputs for correctness, then applies feature scaling using a pre-trained scaler. The scaled data is fed into the Prediction Engine, powered by a Logistic Regression model, which computes malignancy probabilities and classification labels.

Predictions are sent to the Visualization Module, generating radar charts to visually summarize input features and prediction probabilities. Explainability components such as SHAP values provide detailed insights into feature influence. Finally, users can View Results and, in batch mode, download prediction reports. The session can be terminated via the Logout function.

Flowchart Outline

Start

User Authentication

- Login
- Register
- Authentication successful? → Yes: proceed; No: retry or exit

Select Input Mode

- Single Sample Input
- Batch CSV Upload

Data Input

- For Single: sliders or manual entry of diagnostic features
- For Batch: upload a CSV file
- For DICOM processing: provide files directory

Data Validation & Preprocessing

- Check input integrity
- Scale features with StandardScaler

Model Prediction

- Load Logistic Regression model
- Predict malignancy label and probabilities

Visualization & Explanation

- Generate radar chart of features
- Display prediction probabilities
- Show SHAP explainability plots

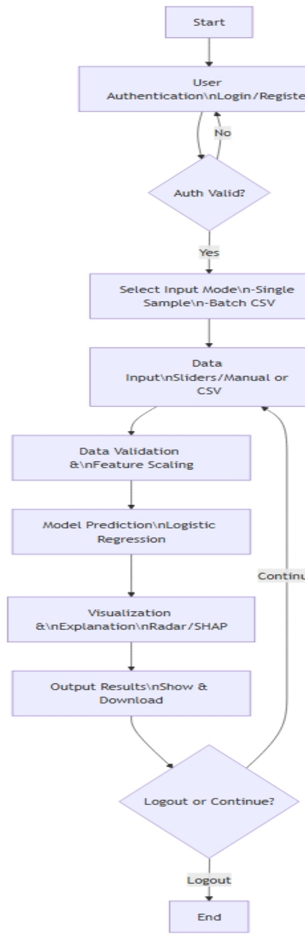
Output Results

- Display on UI
- For batch: enable CSV download of results

User Options

- Continue with new prediction
- Logout / End session

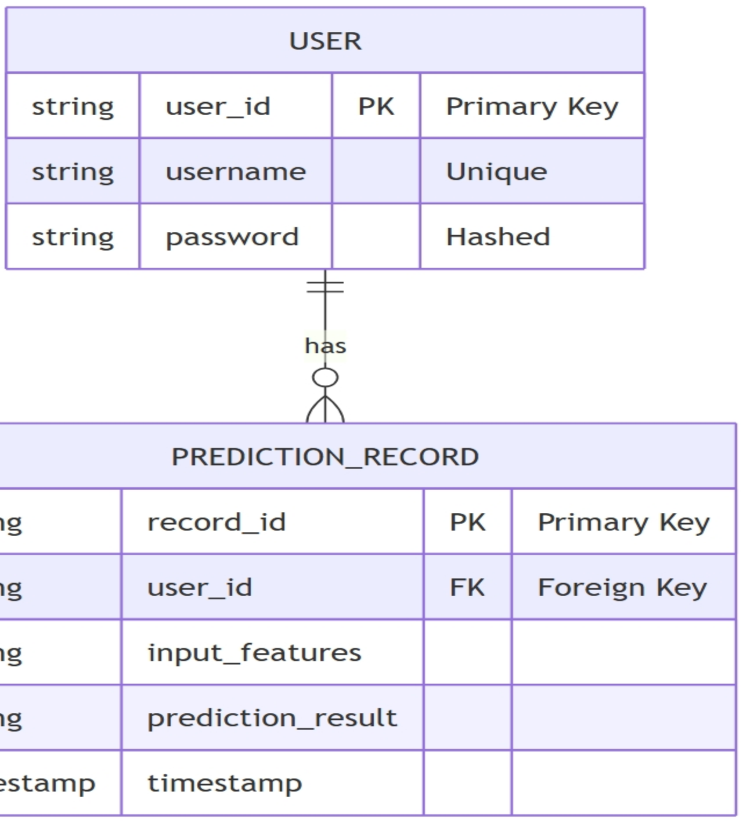
End **Flowchart**


Start
User Authentication\nLogin/Register
No
Auth Valid?
Yes
Select Input Mode\n-Single Sample\n-Batch CSV
Data Input\nSliders/Manual or CSV
Data Validation &\nFeature Scaling
Model Prediction\nLogistic Regression
Continue
Visualization &\nExplanation\nRadar/SHAP
Output Results\nShow & Download
Logout or Continue?
Logout
End

ER-Diagram Description

The Entity-Relationship Diagram (ERD) models the structure of the user authentication and data management system within the Breast Cancer Predictor application. It highlights the main entity User with attributes such as username and password. The user information is securely stored in a JSON file as implemented. Relationships between entities define how user data and prediction data are associated. This diagram supports understanding of data organization, security management, and system scalability.

ER-Diagram

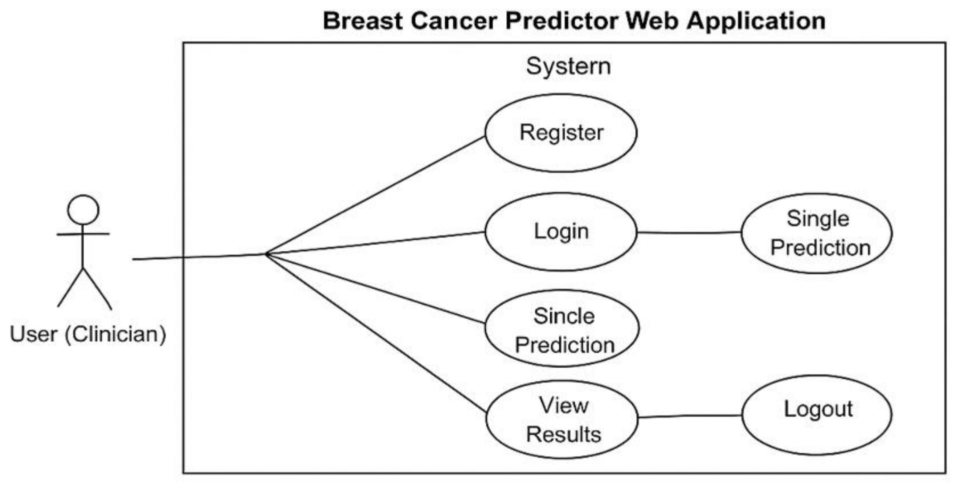

USER
string | user_id | PK | Primary Key
string | username | | Unique
string | password | | Hashed
has
PREDICTION_RECORD
string | record_id | PK | Primary Key
string | user_id | FK | Foreign Key
string | input_features
string | prediction_result
timestamp | timestamp

The ER Diagram outlines the database structure for a breast cancer prediction app. It includes two entities: User (with user_id as the primary key, username, and hashed password). The one-to-many relationship (crow's foot notation) shows that one user can have multiple prediction records. This design ensures data integrity, tracks user-specific predictions, and links records to authenticated users.

b. UML Diagrams

To ensure the clear understanding and efficient development of the Breast Cancer Predictor Web Application, several UML diagrams were created. These diagrams provide a visual representation of the system's structure and the interactions between various components. Below, we describe three key UML diagrams: Use Case Diagram, Class Diagram, and Activity Diagram.

i. Use Case Diagram

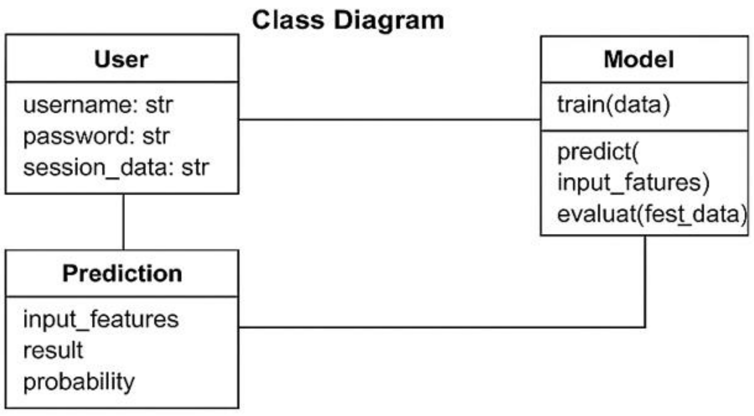


The Use Case Diagram visually represents the system's key functionalities and the interactions between users (actors) and the system. The primary actors in this system are the User (Clinician) and the System. The diagram highlights the main use cases:

- User Authentication: Register, Login, and Logout functionalities.
- Single Sample Prediction: Users input diagnostic features via the Streamlit interface to obtain a prediction.
- Batch or DICOM Prediction: Users upload a file containing multiple data entries, which are processed in batch.
- View Results: Prediction results are displayed with probabilities, visual explanations, and SHAP values.

The Use Case Diagram provides a clear overview of the application's core functionalities and how users interact with it.

ii. Class Diagram

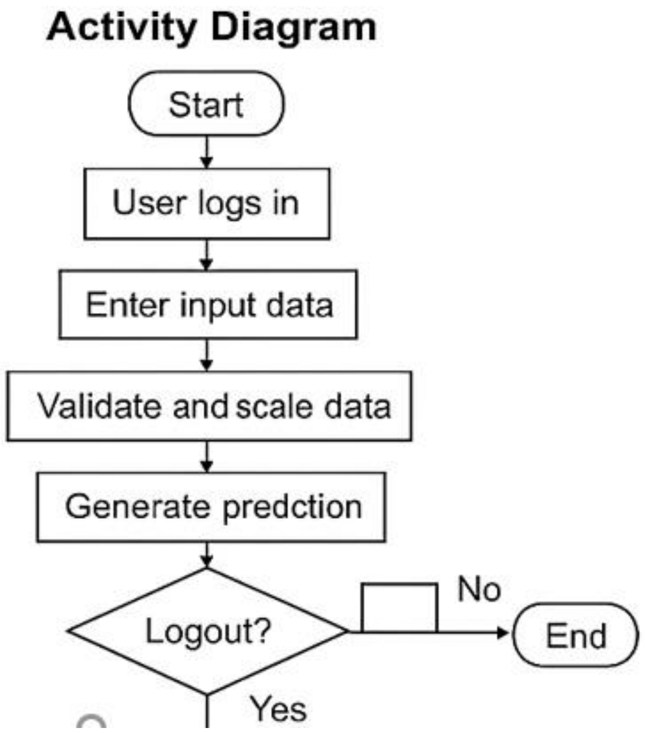


The Class Diagram outlines the system's core classes and their relationships. Key classes include:

- User: Handles user authentication, storing attributes such as username, password, and session data.
- Prediction: Stores input features, prediction results, and model probabilities.
- Model: Encapsulates the Logistic Regression model, responsible for training, prediction, and evaluation.
- UIController: Manages interactions between the user interface and backend logic, including input handling and result display.

The Class Diagram helps developers understand the structure of the codebase and the relationships between different modules.

iii. **Activity Diagram**



The Activity Diagram illustrates the flow of activities within the system, particularly the prediction process. The key steps are as follows:

1. User logs in.
2. Data input: Users provide input data (single or batch).
3. Data validation and scaling.
4. Model prediction: The Logistic Regression model is invoked to generate predictions.
5. Results display: Prediction results and visualizations are shown to the user.
6. Logout or repeat: Users can either logout or continue with new inputs.

This diagram provides a clear understanding of the sequence of actions and how data flows through the system.

9. IMPLEMENTATION

a. Environment Setup

The development of the Breast Cancer Predictor Web Application was conducted in a controlled environment to ensure reproducibility and stability. The following setup details summarize the software environment used throughout the project:

- Python Version: 3.10.x
- **Virtual Environment:** A virtual environment was created using venv to isolate project dependencies from the system Python installation, facilitating clean package management.
- **Dependency Management:** All required packages were listed in a requirements.txt file. This included critical libraries such as Streamlit, scikit-learn, pandas, numpy, plotly, and shap. Installing dependencies was performed using:
- **Development Tools:**
  The project was developed using Visual Studio Code (VS Code), chosen for its extensive Python support, debugging tools, and Git integration.
  Version control was maintained using Git, with a remote repository hosted on GitHub to facilitate collaboration and version tracking.

This environment setup ensures consistent behavior across different machines and enables team members to replicate the development conditions seamlessly.

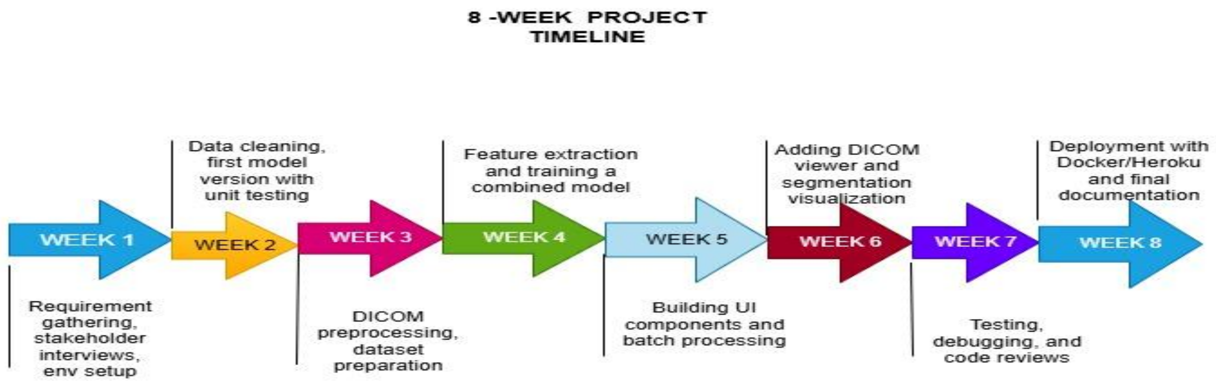
b. Module Development & Responsibilities

Efficient division of labor and clear assignment of responsibilities are essential for the successful development of the Breast Cancer Predictor Web Application. The project team collaborated closely, with each member taking ownership of specific modules based on expertise and project requirements.

Team Roles and Contributions:

- **Julius Laggah (Team Lead & Data Scientist):**
  Led the project planning and coordination. Responsible for the development and training of the Logistic Regression model, data preprocessing, and evaluation. Integrated the machine learning model into the application backend and ensured the overall data pipeline's integrity.
- **Abdul Karim Jusu (Backend Developer):**
  Managed the backend logic, including data loading, feature scaling, and prediction pipeline. Implemented the SQLite3-based user authentication system and maintained secure session management.
- **Sandar Win (Frontend Developer & UI/UX Designer):**
  Designed and developed the Streamlit user interface. Implemented input sliders, batch CSV upload components, and interactive visualizations such as radar charts. Styled the application with custom CSS including dark mode integration.
- **Adamsay Turay (Frontend Developer & Visualization):**
  Created the interactive charts and graphical components using Plotly and SHAP. Developed the real-time visualization of prediction explanations and model interpretability features.

IMPLEMENTATION ROAD MAP


8 -WEEK PROJECT TIMELINE
WEEK 1
Requirement gathering, stakeholder interviews, env setup
WEEK 2
Data cleaning, first model version with unit testing
WEEK 3
DICOM preprocessing, dataset preparation
WEEK 4
Feature extraction and training a combined model
WEEK 5
Building UI components and batch processing
WEEK 6
Adding DICOM viewer and segmentation visualization
WEEK 7
Testing, debugging, and code reviews
WEEK 8
Deployment with Docker/Heroku and final documentation

10. CONCLUSION AND FUTURE WORK

a. Summary

The Breast Cancer Predictor Web Application project successfully delivered an accessible and interactive tool designed to assist healthcare professionals in the early detection of breast cancer. Leveraging the Breast Cancer Wisconsin and the DICOM image dataset, the project employed a Logistic Regression model to classify breast tissue samples as benign or malignant with high accuracy and reliability.

The web application was developed using the Streamlit framework, providing an intuitive interface for both single-sample and batch predictions. Users can manually input diagnostic measurements through sliders or upload CSV files containing multiple samples. The inclusion of visual explanations via radar charts and SHAP values enhances interpretability and fosters user trust in the model's predictions.

A secure user authentication system using SQLite3 ensures that only authorized users can access the application, safeguarding sensitive information and maintaining data integrity. Additionally, multi-language integration, including C, C++, and Cython, was employed to optimize computational efficiency, enhancing the responsiveness of the app.

Thorough testing and validation confirmed the system's accuracy, usability, and performance. The project also incorporated robust configuration management and deployment strategies, including containerization support and continuous deployment through GitHub integration, facilitating scalability and maintainability.

This project not only demonstrates the potential of machine learning applications in healthcare but also highlights the importance of combining predictive accuracy with user-friendly design and security. While intended as an educational and supplementary tool rather than a clinical diagnostic device, the Breast Cancer Predictor Web Application lays the foundation for future enhancements that could further integrate advanced models and real-world clinical data.

The project's source code and documentation are openly available on GitHub, promoting transparency, reproducibility, and collaboration.

b. Challenges

During the development of the Breast Cancer Predictor Web Application, several challenges were encountered that provided valuable learning opportunities and influenced the project's scope and implementation.

Data Limitations

One significant challenge was the reliance on the Breast Cancer Wisconsin dataset, which, while comprehensive and widely used, may not fully represent the diversity and complexity of real-world clinical data. The dataset's size and scope limited the ability to generalize predictions to broader populations or varying clinical scenarios.

Model Selection and Interpretability

Choosing a suitable machine learning model involved balancing accuracy with interpretability. While complex models like neural networks could potentially improve prediction accuracy, the project prioritized **Logistic Regression** for its transparency and explainability, essential for clinical trust. Integrating SHAP values further enhanced interpretability but required careful implementation to ensure meaningful visual explanations.

User Authentication Security

Implementing a secure yet user-friendly authentication system posed challenges related to password storage, session management, and protection against common web vulnerabilities. Using SQLite3 offered simplicity but required careful handling to maintain security standards.

Deployment and Environment Consistency

Maintaining consistent behavior across development and deployment environments required diligent configuration management. Issues such as dependency conflicts and environment-specific bugs necessitated containerization strategies and clear installation guides.

Time Constraints

The project was developed within an academic semester timeline, placing constraints on feature breadth and depth. This necessitated prioritizing core functionalities and deferring advanced features such as integration with clinical imaging data or complex model ensembles.

c. Future Enhancements

Building upon the foundation established by the current version of the Breast Cancer Predictor Web Application, several enhancements are planned to improve functionality, accuracy, and usability:

- **Integration of Advanced Machine Learning Models:**
  Future versions may incorporate more sophisticated models such as ensemble methods or deep learning architectures to improve prediction accuracy while exploring techniques to maintain interpretability.
- **Expansion of Dataset Sources:**
  Incorporating additional and more diverse clinical datasets, including real-world patient data and imaging modalities like DICOM, will enhance the model's generalizability and clinical relevance.
- **Real-time Data Acquisition:**
  Integration with laboratory information systems or medical imaging devices could enable real-time data input, streamlining workflows and reducing manual data entry errors.
- **Enhanced Security and User Management:**
  Improvements to authentication mechanisms, including multi-factor authentication and audit logging, will bolster data security and compliance with healthcare regulations.
- **User Interface Improvements:**
  Further refinement of the UI/UX based on user feedback, including accessibility features, customizable dashboards, and multilingual support.

11. APPENDICES

GitHub Repository

The complete source code, documentation, and deployment scripts for the Breast Cancer Predictor Web Application are publicly hosted on GitHub to promote transparency, reproducibility, and collaborative development.

- **Repository URL:**
  https://github.com/juliuslaggah/breast-cancer-predictor
- **Contents:**
  - Source code for the machine learning model and Streamlit web application.
  - Preprocessing and training scripts.
  - User authentication implementation.
  - Custom CSS for UI styling and theming.
  - Example CSV files for batch prediction.
  - requirements.txt for dependency management.
  - Documentation including the user manual, installation guide, and troubleshooting instructions.
- **Contribution Guidelines:**
  Developers interested in contributing can fork the repository, submit pull requests, and report issues via GitHub's issue tracker. The repository is maintained with continuous integration workflows to ensure code quality and consistency.

This open-source approach facilitates community engagement, peer review, and future enhancements.

Installation Guides

The installation guides provide users and developers with step-by-step instructions to set up the **Breast Cancer Predictor Web Application** in various environments.

Local Installation

1. **Clone the Repository:**

```
git clone https://github.com/juliuslaggah/breast-cancer-predictor.git

cd breast-cancer-predictor
```

2. **Create and Activate Virtual Environment:**

```
python -m venv venv

# Windows

venv\Scripts\activate

# macOS/Linux

source venv/bin/activate
```

3. **Install Dependencies:**

```
pip install -r requirements.txt
```

4. **Run the Application:**

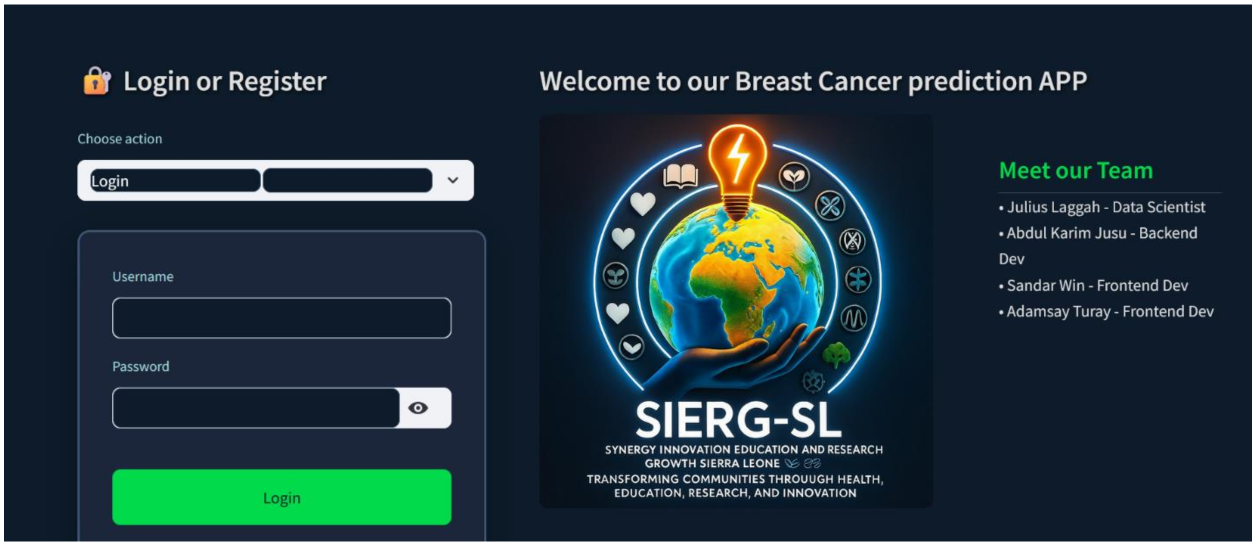
```
streamlit run main.py
```

Additional Notes

- Ensure Python 3.10+ is installed and accessible in your system PATH.
- Use the provided example CSV files for batch prediction testing.
- Consult the troubleshooting guide for common issues during installation.

SYSTEM OVERVIEW

Landing page for user registration and login



Login or Register

Choose action

Login

Username

Password

Login

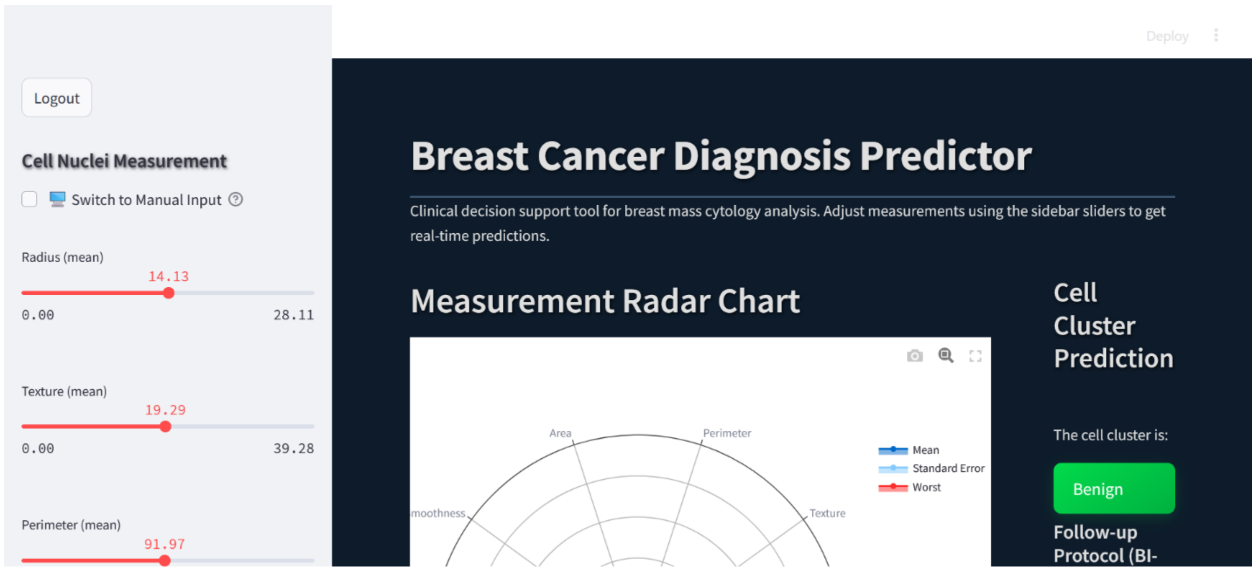
Welcome to our Breast Cancer prediction APP

SIERG-SL
SYNERGY INNOVATION EDUCATION AND RESEARCH GROWTH SIERRA LEONE
TRANSFORMING COMMUNITIES THROUUGH HEALTH, EDUCATION, RESEARCH, AND INNOVATION

Meet our Team

- Julius Laggah - Data Scientist
- Abdul Karim Jusu - Backend Dev
- Sandar Win - Frontend Dev
- Adamsay Turay - Frontend Dev

Prediction dashboard



Deploy

Logout

Cell Nuclei Measurement

Switch to Manual Input

Radius (mean) 14.13 0.00 28.11

Texture (mean) 19.29 0.00 39.28

Perimeter (mean) 91.97

Breast Cancer Diagnosis Predictor

Clinical decision support tool for breast mass cytology analysis. Adjust measurements using the sidebar sliders to get real-time predictions.

Measurement Radar Chart

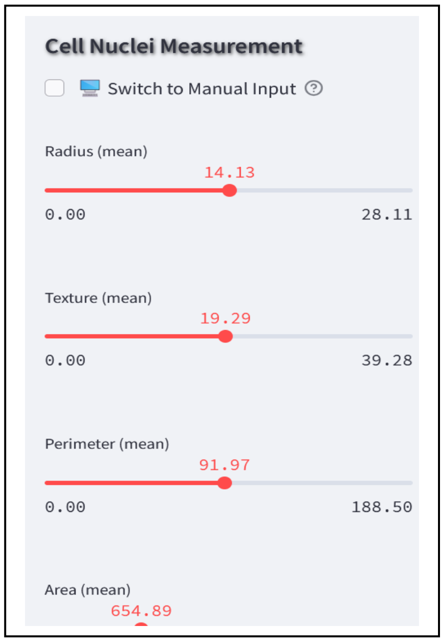
Cell Cluster Prediction

The cell cluster is:

Benign

Follow-up Protocol (BI-

PREDICTIONS MECHANISM


Cell Nuclei Measurement

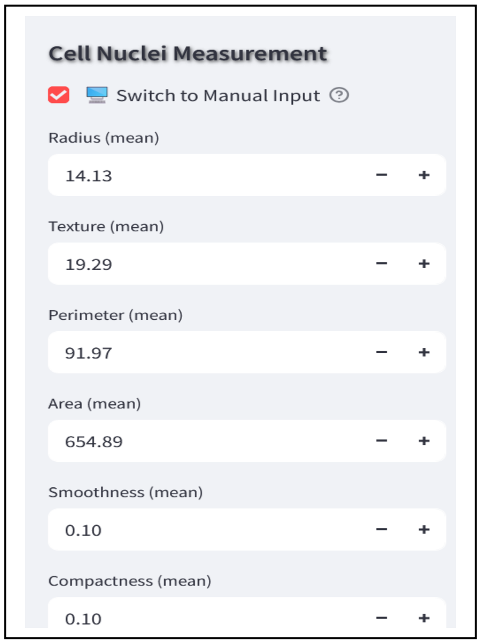
☐ Switch to Manual Input

Radius (mean) 14.13 — 0.00 to 28.11

Texture (mean) 19.29 — 0.00 to 39.28

Perimeter (mean) 91.97 — 0.00 to 188.50

Area (mean) 654.89


Cell Nuclei Measurement

☑ Switch to Manual Input

Radius (mean): 14.13

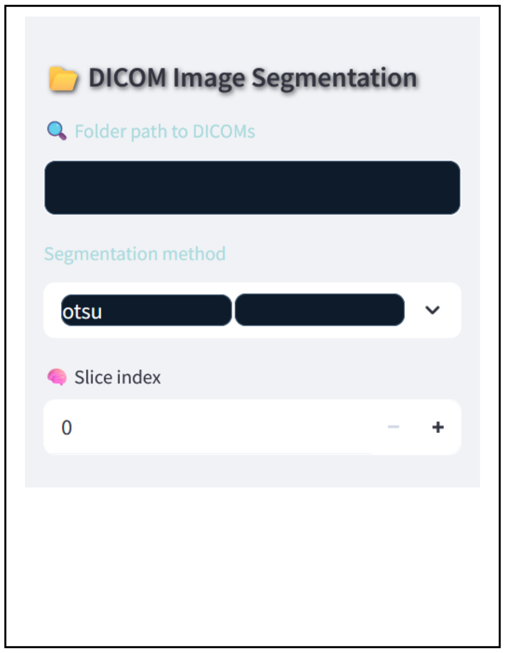
Texture (mean): 19.29

Perimeter (mean): 91.97

Area (mean): 654.89

Smoothness (mean): 0.10

Compactness (mean): 0.10


DICOM Image Segmentation

Folder path to DICOMs

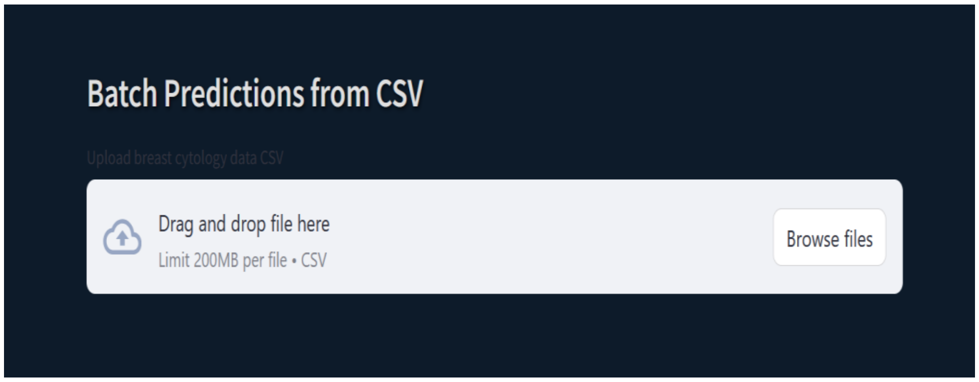
Segmentation method: otsu

Slice index: 0

SLIDER INPUT

MANUAL INPUT

DICOM FILE INPUT


Batch Predictions from CSV

Upload breast cytology data CSV

Drag and drop file here

Limit 200MB per file • CSV

Browse files

CSV FILE UPLOAD

REFERENCES

- Kaggle. (n.d.). Breast Cancer Wisconsin (Diagnostic) Data Set. Available at: https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data
- Alejandro AO. (n.d.). Breast Cancer Diagnosis App Tutorial [YouTube Channel]. Available at: https://www.youtube.com/@alejandro_ao
- Kumar, S., Sharma, M., and Gupta, A. (2021). 'Ensemble machine learning approach for breast cancer prediction using mammogram images', Journal of Medical Systems, 45(5), pp. 1–12. https://doi.org/10.1007/s10916-021-01725-8.
- Singh, R. and Gupta, P. (2022). 'Cloud-based deep learning system for breast cancer diagnosis using mammography images', International Journal of Computer Assisted Radiology and Surgery, 17(3), pp. 543–551. https://doi.org/10.1007/s11548-021-02413-w.
- Pedregosa, F. et al. (2011). 'Scikit-learn: Machine learning in Python', Journal of Machine Learning Research, 12, pp. 2825–2830. Available at: https://jmlr.org/papers/v12/pedregosa11a.html [Accessed 25 May 2025].
- Molnar, C. (2020). Interpretable Machine Learning. Available at: https://christophm.github.io/interpretable-ml-book/