International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Trustworthy Predictions: An Explainable AI Approach to Breast Cancer Diagnosis

Dasugari Vaishnavi, Demada Trisha

ABSTRACT

Breast cancer continues to be one of the leading causes of cancer-related deaths among women worldwide. Early and accurate diagnosis is crucial to improving treatment outcomes and survival rates. This study develops a robust machine learning-based breast cancer classification system utilizing the Wisconsin Diagnostic Breast Cancer (WDBC) dataset. Emphasizing the importance of feature selection, the study identifies the most influential tumor characteristics that significantly contribute to malignancy prediction. Multiple classification algorithms—including Random Forest, Support Vector Machine (SVM), and k-Nearest Neighbors (k-NN)—were implemented and rigorously evaluated to determine their predictive performance. After applying feature scaling and selecting optimal features, the SVM model achieved the highest classification accuracy of 96.49%, demonstrating its effectiveness in distinguishing between benign and malignant tumors. To facilitate practical clinical application, the model was deployed via a web-based interface built using Flask, allowing healthcare professionals to input tumor measurements and receive immediate diagnostic predictions. This deployment bridges the gap between complex machine learning models and real-world usability, supporting early detection efforts in clinical settings. The project underscores the potential of combining advanced computational techniques with intuitive interfaces to improve diagnostic workflows. Future work aims to integrate explainable artificial intelligence (XAI) methods to enhance transparency in prediction outcomes and to extend the model's applicability by incorporating diverse and larger clinical datasets, thereby increasing its robustness and generalizability.

CHAPTER-1

INTRODUCTION

1

1.Introduction :

Breast cancer is one of the most common cancers affecting women worldwide and is a significant cause of mortality. Early diagnosis and timely intervention are essential to improving patient outcomes and reducing mortality rates. Traditional diagnostic methods, such as mammography and biopsy, while effective, can be time-consuming, costly, and sometimes subjective, leading to the need for automated, reliable, and efficient diagnostic tools. Recent advances in machine learning (ML) have opened new avenues for medical diagnostics, enabling the development of predictive models that can assist clinicians in accurately classifying tumors based on complex data patterns. The use of ML in breast cancer diagnosis can help in identifying malignancy with high accuracy, thereby aiding in early detection and personalized treatment planning. In this project, we focus on creating a breast cancer classification system using machine learning algorithms applied to the Wisconsin Diagnostic Breast Cancer (WDBC) dataset. The dataset comprises multiple features extracted from digitized images of fine needle aspirate (FNA) of breast masses. Through feature selection, we aim to identify the most significant predictors of tumor malignancy to improve model efficiency and interpretability. Several machine learning models, including Random Forest, Support Vector Machine (SVM), and k-Nearest Neighbors (k-NN), were trained and evaluated to determine the best performing algorithm. The final model was deployed as a web-based application to facilitate easy access and real-time diagnostic support for healthcare professionals. This introduction of machine learning-based tools into clinical workflows has the potential to enhance diagnostic accuracy, reduce manual workload, and support early cancer detection, ultimately contributing to better patient care.

1.2 SCOPE OF THE PROJECT

This project aims to develop an accurate and practical machine learning-based system for classifying breast tumors as benign or malignant using the Wisconsin Diagnostic Breast Cancer dataset. By focusing on the most relevant features through feature selection, the project seeks to optimize model performance while maintaining interpretability. The deployment of the model as a web-based application ensures that the system is accessible and user-friendly for healthcare professionals, enabling real-time diagnostic support. Although the current implementation is based on a specific dataset, the framework is designed to be adaptable for integration with larger and more diverse clinical datasets in the future. Additionally, the project lays the foundation for incorporating advanced explainability techniques to provide deeper insights into model predictions, thereby enhancing trust and usability in real-world clinical settings.

1.3 OBJECTIVE

The primary objective of this project is to develop a reliable and efficient machine learning-based system for the early detection and classification of breast cancer tumors into benign or malignant categories. This involves identifying the most significant predictive features from the dataset to improve model accuracy and reduce complexity. Additionally, the project aims to compare multiple classification algorithms to select the best-performing model for this task. Another key objective is to implement the trained model into a user-friendly web application, enabling healthcare professionals to quickly and accurately obtain diagnostic predictions. Ultimately, the project strives to demonstrate the practical potential of machine learning in supporting early breast cancer diagnosis and improving patient outcomes.

1.4 EXISTING SYSTEM:

The existing system for breast cancer classification employs the k-Nearest Neighbors (k-NN) algorithm, a straightforward and intuitive machine learning technique widely used for classification problems. The k-NN algorithm operates by identifying the ‘k’ closest data points to a given sample based on a selected distance metric, typically Euclidean distance, and classifies the sample according to the majority class among these neighbors. This method’s simplicity and ability to capture local patterns make it well-suited for medical diagnosis tasks like tumor classification. In the foundational research, the k-NN model demonstrated impressive performance on the Wisconsin Breast Cancer Dataset (WBCD), achieving an accuracy of 97.7% and a precision of 98.2%, indicating its strong potential in correctly identifying malignant tumors while minimizing false positives. However, despite these promising results, the k-NN approach presents several practical challenges and limitations. One critical factor affecting its effectiveness is the choice of the parameter ‘k’, which significantly influences the bias-variance trade-off: a small k may lead to noisy and unstable predictions, while a large k can smooth out important local patterns. The selection of an appropriate distance metric also plays a crucial role in accurately measuring similarity, especially in datasets with heterogeneous or correlated features.

Moreover, the presence of irrelevant or redundant features can degrade k-NN performance since the algorithm treats all features equally when computing distances, potentially leading to misclassification. Another inherent limitation is computational scalability; k-NN requires comparing the input instance to all points in the training set, resulting in high computational cost and slower prediction times as the dataset size grows. This can be a significant bottleneck in real-time clinical environments or when processing large-scale datasets. Additionally, k-NN is sensitive to noisy data and outliers, which can distort neighborhood structures and adversely affect classification accuracy. Due to these challenges, the existing system may benefit from incorporating feature selection techniques, dimensionality reduction, or alternative distance metrics to improve robustness and efficiency. Furthermore, exploring other machine learning algorithms that offer better generalization, interpretability, and scalability—such as Support Vector Machines or Random Forests—can help address the limitations of the k-NN based approach, paving the way for more reliable and practical breast cancer diagnostic tools.

.

1.4.1 EXISTINGSYSTEM DISADVANTAGES:

- Choice of 'k' Parameter: The performance of k-NN heavily depends on the selection of the number of neighbors (k). An inappropriate choice can lead to overfitting (too small k) or underfitting (too large k), reducing the model's accuracy.
- Sensitivity to Distance Metric: k-NN relies on a distance metric (usually Euclidean distance) to measure similarity. If the dataset contains features with different scales or irrelevant features, the distance calculation may become misleading, negatively impacting classification results.
- Computationally Expensive: Since k-NN is a lazy learner, it does not build an explicit model but instead compares each query point to all training samples during prediction. This results in high computation and slower response times, especially for large datasets, making it less suitable for real-time applications.
- Influence of Irrelevant and Redundant Features: The algorithm treats all features equally in distance calculations, so the presence of irrelevant or redundant features can distort similarity measurements and reduce prediction accuracy.
- Sensitivity to Noisy Data and Outliers: Noisy data points and outliers can heavily influence the neighborhood structure, leading to incorrect classifications.
- Lack of Interpretability: While simple, k-NN does not provide insight into the importance of individual features or the reasoning behind predictions, limiting its usefulness in clinical settings where explainability is critical.

1.5 LITERATURE SURVEY

Title: Current and future burden of breast cancer: Global statistics for 2020 and 2040

Author: M. Arnold, E. Morgan, H. Rumgay, A. Mafra, D. Singh, M. Laversanne, J. Vignat, J. R. Gralow, F. Cardoso, S. Siesling, and I. Soerjomataram

Year: 2022

Description: Arnold et al. (2022) present an extensive analysis of the global incidence and mortality rates of breast cancer, projecting trends through 2040. The study emphasizes the rising burden of breast cancer worldwide and underscores the need for improved detection, prevention, and treatment strategies. This paper provides a comprehensive statistical overview, which forms a foundation for prioritizing healthcare resources and research efforts in breast cancer management globally.

Title: Breast cancer classification using deep Q learning (DQL) and gorilla troops optimization (GTO)

Author: S. Almutairi, S. Manimurugan, B.-G. Kim, M. M. Aborokbah, and C. Narmatha

Year: 2023

Description: Almutairi et al. (2023) introduce a novel breast cancer classification framework combining deep Q learning (DQL), a reinforcement learning approach, with gorilla troops optimization (GTO), a metaheuristic algorithm. Their hybrid model enhances feature selection and classification accuracy on breast cancer datasets. This innovative approach leverages both machine learning and optimization techniques to improve diagnostic performance, offering a promising direction for future research in automated cancer detection.

Title: An explainable AI approach for breast cancer metastasis prediction based on clinicopathological data

Author: I. Maouche, L. S. Terrissa, K. Benmohammed, and N. Zerhouni

Year: 2023

Description: Maouche et al. (2023) propose an explainable artificial intelligence (XAI) framework for predicting breast cancer metastasis using clinicopathological data. The study integrates machine learning models with interpretability techniques to provide transparent predictions, facilitating clinical decision-making. This approach addresses the critical need for explainability in AI-driven healthcare applications, ensuring trust and acceptance among medical practitioners.

Title: A hybrid algorithm of ML and XAI to prevent breast cancer: A strategy to support decision making

Author: F. Silva-Aravena, H. Núñez Delafuente, J. H. Gutiérrez-Bahamondes, and J. Morales

Year: 2023

Description: Silva-Aravena et al. (2023) develop a hybrid machine learning and explainable AI algorithm aimed at enhancing breast cancer prevention strategies. By combining predictive modeling with explainability tools, the framework supports healthcare professionals in making informed decisions. This work highlights the growing importance of integrating AI transparency with predictive accuracy to improve clinical outcomes.

Title: An LDA–SVM machine learning model for breast cancer classification

Author: O. J. Egwom, M. Hassan, J. J. Tanimu, M. Hamada, and O. M. Ogar

Year: 2022

Description: Egwom et al. (2022) introduce a hybrid classification model combining Linear Discriminant Analysis (LDA) with Support Vector Machines (SVM) to improve breast cancer detection. Their model leverages LDA for dimensionality reduction followed by SVM for classification, achieving high accuracy on benchmark datasets. This study demonstrates the effectiveness of hybrid machine learning techniques in enhancing diagnostic performance in oncology.

1.6 PROPOSED SYSTEM

Support Vector Machine (SVM) is a powerful supervised machine learning algorithm widely recognized for its effectiveness in classification tasks, especially with high-dimensional data such as medical diagnostic datasets. In this project, SVM is utilized to classify breast tumors as benign or malignant based on key features extracted from the Wisconsin Diagnostic Breast Cancer (WDBC) dataset. The SVM algorithm works by finding the optimal hyperplane that best separates the classes in the feature space, maximizing the margin between the closest points (support vectors) of different classes. This margin maximization helps SVM generalize well to unseen data, reducing the risk of overfitting. Additionally, SVM can handle both linear and non-linear classification problems through the use of kernel functions, such as the Radial Basis Function (RBF) kernel, enabling it to model complex relationships between features. In the proposed system, after appropriate feature selection and scaling, the SVM model achieved a high accuracy of 96.49%, outperforming or matching other classical classifiers. The robustness of SVM to noisy data and its effectiveness with smaller datasets make it particularly suitable for medical diagnosis tasks where data quality and quantity can be limiting factors. The model is deployed as a web-based application to provide healthcare professionals with an accessible and efficient tool for real-time breast cancer classification, aiming to assist in early detection and improve clinical decision-making.

1.6.1 PROPOSED SYSTEM ADVANTAGES:

- High Classification Accuracy: SVM typically provides excellent classification performance, as demonstrated by the 96.49% accuracy achieved on the breast cancer dataset, ensuring reliable diagnosis.
- Effective in High-Dimensional Spaces: SVM performs well even when the number of features is large relative to the number of samples, which is common in medical datasets.
- Robustness to Overfitting: By maximizing the margin between classes, SVM reduces the risk of overfitting, leading to better generalization on unseen data.
- Flexibility with Kernels: The ability to use different kernel functions allows SVM to model complex, non-linear relationships between features, improving classification in diverse scenarios.

CHAPTER 2

PROJECT DESCRIPTION

2.1 GENERAL:

Breast cancer remains one of the most critical health challenges worldwide, and early detection is key to reducing mortality and improving treatment outcomes. This project focuses on developing an effective and practical breast cancer classification system that leverages Support Vector Machine (SVM) — a supervised machine learning algorithm well-suited for medical diagnosis tasks. The system utilizes the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which contains quantitative features extracted from digitized images of fine needle aspirate (FNA) of breast masses. These features represent various tumor characteristics such as texture, area, perimeter, and concavity. To enhance model performance and reduce computational complexity, feature selection techniques were applied to identify the eight most significant predictors related to tumor malignancy. Following data preprocessing, including feature scaling and normalization, multiple machine learning algorithms—Random Forest, k-Nearest Neighbors (k-NN), and Support Vector Machine—were trained and evaluated. Among these, the SVM classifier, with an appropriate kernel function, demonstrated superior accuracy of 96.49%, showcasing its robustness and effectiveness in distinguishing benign from malignant tumors. To translate this model into a practical tool, the trained SVM classifier was integrated into a web-based application developed using the Flask framework. This interface allows users—such as clinicians and healthcare professionals—to input tumor feature values and receive immediate predictions about the tumor class, facilitating quick decision-making in clinical environments. The project not only emphasizes predictive accuracy but also usability, ensuring the system is accessible for real-world medical applications. Future improvements may include expanding the dataset to incorporate diverse patient populations, integrating explainable AI techniques to enhance model transparency, and developing mobile-friendly platforms for broader accessibility.By combining advanced machine learning with user-centric design, this project aims to contribute to the early detection of breast cancer, ultimately improving patient prognosis and healthcare efficiency.

2.2 METHODOLOGIES

2.2.1MODULES NAME:

Modules Name:

- **Data Collection and Preprocessing:**
- **Model Selection**
- **Hyperparameter Tuning**
- **Cross-Validation**
- **Ensemble Methods (Existing Model)**
- **Model Evaluation**
- **Comparison with Existing Methods**

2.2.2 MODULES EXPLANATION:

1)Data Collection and Preprocessing:

The primary dataset used is the Wisconsin Breast Cancer Diagnostic (WBCD) dataset, which includes features of tumors such as size, texture, and shape. Preprocessing steps typically include handling missing values, scaling/normalizing the data to ensure all features contribute equally, and splitting the data into training and testing sets to avoid overfitting.

2) Model Selection:

The project leverages the Support Vector Classifier (SVC) model, a supervised machine learning algorithm particularly effective for classification tasks. SVC creates a decision boundary by maximizing the margin between data points of different classes, providing an optimal separation for accurate classification.

3) Hyperparameter Tuning:

GridSearchCV is used to optimize the model by selecting the best set of hyperparameters through an exhaustive search over a predefined parameter grid. This helps find the most suitable parameters for the SVC, such as the regularization parameter C, kernel type (linear or radial basis function (RBF)), and gamma, which affect the decision boundary's flexibility and generalization ability.

4) Cross-Validation:

To ensure the model's performance generalizes well to unseen data, cross-validation (with 5-folds in your case) is applied. This technique involves partitioning the training data into subsets, training the model on some and validating it on others, which helps mitigate issues of overfitting.

5) Ensemble Methods:

In addition to the SVC model, the existing model uses an ensemble learning approach, combining multiple classifiers—Extra Trees Classifier (ETC), Light Gradient Boosting Machine (LightGBM), Ridge Classifier (RC), and Linear Discriminant Analysis (LDA)—through a voting mechanism. Ensemble methods typically improve model performance by combining the strengths of individual classifiers to reduce bias and variance in the final predictions..

6) M odel Evaluation:

The performance of the models is evaluated using various metrics, including accuracy, precision, recall, and F1-score, to ensure they provide a well-rounded assessment of classification performance. These metrics help determine how well the model distinguishes between benign and malignant tumors.

7) Comparison with Existing Methods:

The results from proposed SVC model are compared against those of the ensemble model, with the goal of assessing whether the fine-tuned SVC can match or exceed the performance of the ensemble approach.

2.3 TECHNIQUE USED OR ALGORITHM USED

2.3.1 EXISTING TECHNIQUE: -

- **k-Nearest Neighbors**

The existing breast cancer classification system utilizes the k-Nearest Neighbors (k-NN) algorithm, which is a widely used instance-based learning technique for classification problems. In k-NN, classification of an unknown data point is performed by identifying the 'k' closest training samples in the feature space based on a distance metric, most commonly the Euclidean distance. Once the nearest neighbors are determined, the algorithm assigns the class label that is most frequent among these neighbors. This method relies on the assumption that similar data points are located close to each other in the feature space. One of the key advantages of k-NN is its simplicity and ease of implementation, requiring no explicit training phase or model building, as it uses the entire training dataset during prediction. However, this also leads to high computational costs during classification, especially with large datasets, since distances must be computed between the query point and every training instance. Moreover, the choice of 'k' significantly impacts performance; a smaller 'k' may lead to noisy and less stable predictions, while a larger 'k' can cause the model to overlook local data structures and reduce sensitivity. Additionally, k-NN assumes all features contribute equally to the distance calculation, which can be problematic in the presence of irrelevant or redundant features. Without proper feature selection or weighting, this can reduce classification accuracy. The algorithm is also sensitive to noisy data and outliers, which can mislead neighborhood definitions. Despite these limitations, in the context of the Wisconsin Breast Cancer Dataset, k-NN has demonstrated strong performance, achieving an accuracy of approximately 97.7% and precision of 98.2%, making it a viable baseline method for breast cancer classification.

2.3.2 PROPOSED TECHNIQUE USED OR ALGORITHM USED:

- **SUPPORT VECTOR Machine (SVM):**

The proposed system leverages the Support Vector Machine (SVM) algorithm, which is widely regarded as one of the most effective classifiers for medical data due to its ability to handle complex, high-dimensional datasets and its strong theoretical foundations. SVM operates by identifying a decision boundary, called the hyperplane, that best separates the data points of different classes—in this case, benign versus malignant breast tumors. The optimal hyperplane is defined as the one that maximizes the margin, which is the distance between the hyperplane and the closest data points from each class, referred to as support vectors. By focusing solely on these critical support vectors, SVM minimizes classification error while enhancing generalization to unseen data. One of the main advantages of SVM is its flexibility through the use of kernel functions, which enable it to solve non-linear classification problems by implicitly mapping the original feature space into a higher-dimensional space where linear separation is possible. Common kernels include the Radial Basis Function (RBF), polynomial, and sigmoid kernels, each suited for different data distributions. In the context of breast cancer classification, the RBF kernel is often favored because it can effectively handle non-linear relationships inherent in medical data without excessive computational cost. Before training, the dataset undergoes preprocessing steps such as feature scaling and normalization to ensure that all features contribute equally to the model and to improve convergence speed. Feature selection further refines the input by isolating the most relevant predictors of malignancy, such as area_worst and concave_points_worst, reducing noise and dimensionality. During model training, hyperparameters such as the penalty parameter (C), which controls the trade-off between maximizing the margin and minimizing classification errors, and kernel-specific parameters like gamma (for RBF) are optimized through techniques like grid search and cross-validation. This tuning process is crucial for balancing bias and variance, ensuring robust and accurate predictions..

CHAPTER 3

REQUIREMENTS ENGINEERING

3.1 GENERAL

We can see from the results that on each database, the error rates are very low due to the discriminatory power of features and the regression capabilities of classifiers. Comparing the highest accuracies (corresponding to the lowest error rates) to those of previous works, our results are very competitive.

3.2 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system do and not how it should be implemented.

- PROCESSOR : DUAL CORE 2 DUOS.
- RAM : 4GB DD RAM
- HARD DISK : 250 GB

3.3 SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

- Operating System : Windows 7/8/10
- Platform : Spyder3
- Programming Language : Python
- Front End : Spyder3

3.4 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, Firstly, the system is the first that achieves the standard notion of semantic security for data confidentiality in attribute-based deduplication systems by resorting to the hybrid cloud architecture.

3.5 NON-FUNCTIONAL REQUIREMENTS

The major non-functional Requirements of the system are as follows

Usability

The system is designed with completely automated process hence there is no or less user intervention.

Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform python. The code built by using python is more reliable.

Performance

This system is developing in the high level languages and using the advanced back-end technologies it will give response to the end user on client system with in very less time.

Supportability

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is built into the system.

Implementation

The system is implemented in web environment using Jupyter notebook software. The server is used as the intellignce server and windows 10 professional is used as the platform. Interface the user interface is based on Jupyter notebook provides server system.

CHAPTER 4

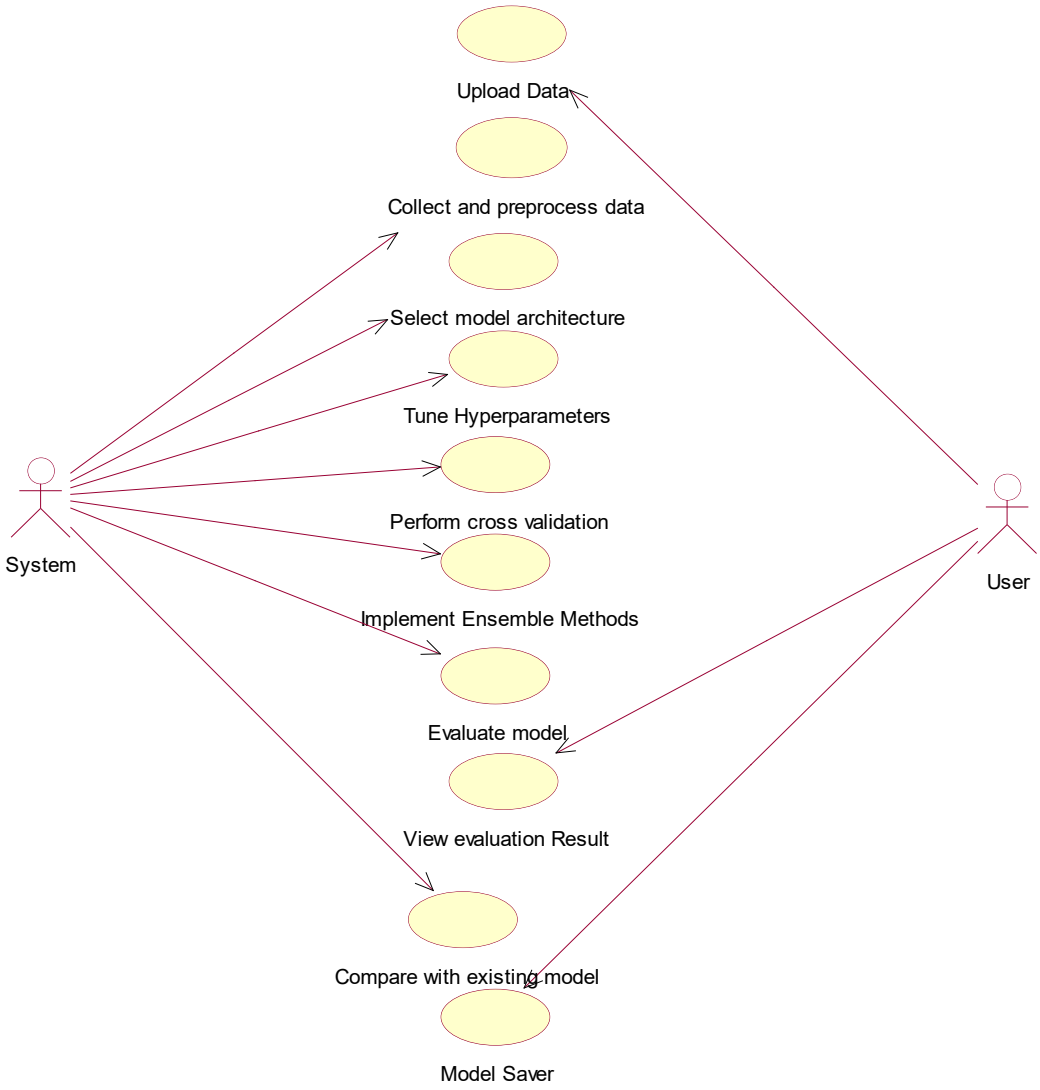
DESIGN ENGINEERING

4.1 GENERAL

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering.

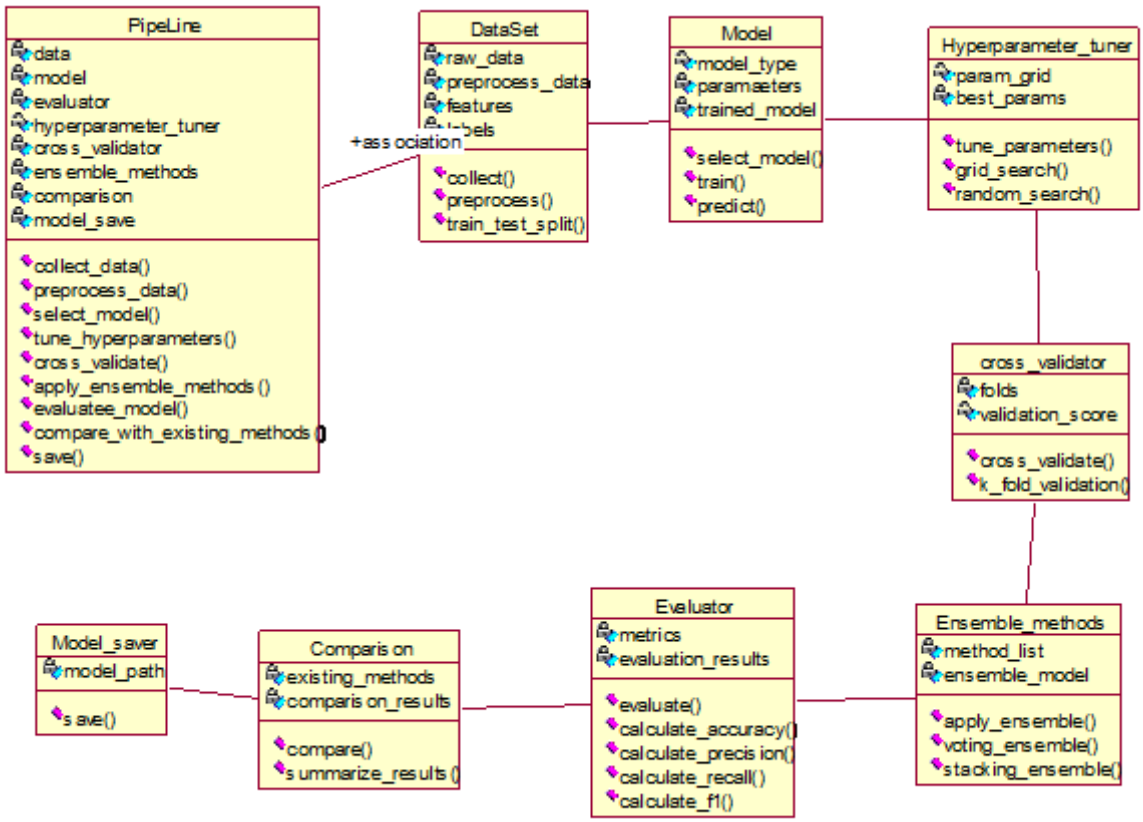
4.2 UML DIAGRAMS

4.2.1 USE CASE DIAGRAM



EXPLANATION: The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

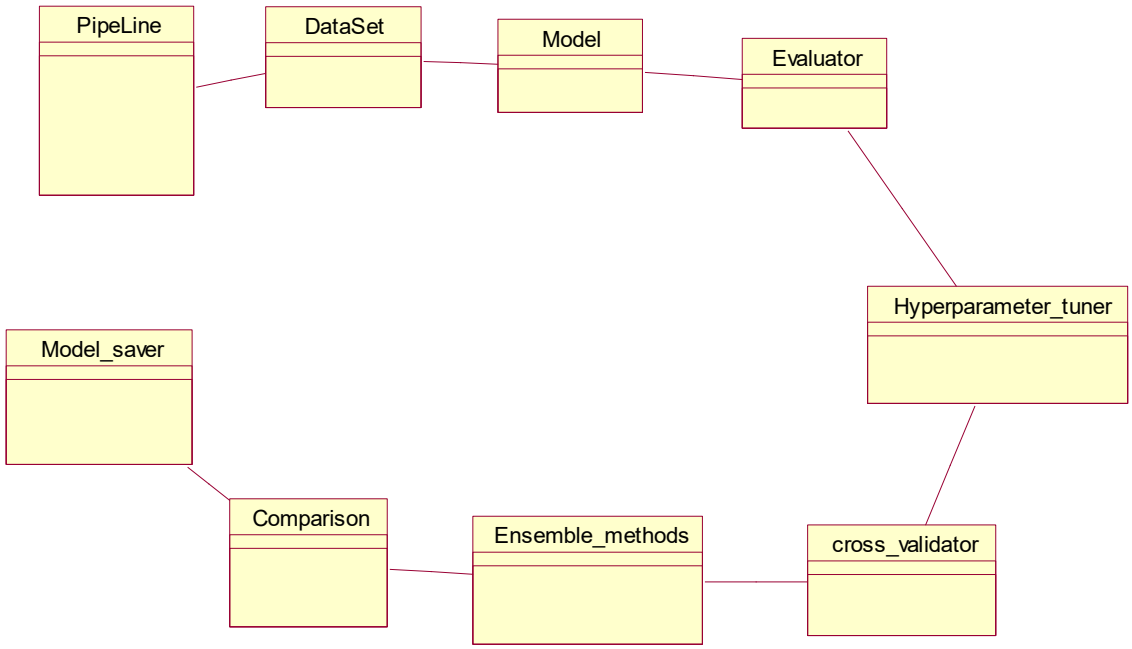
4.2.2 CLASS DIAGRAM



EXPLANATION

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

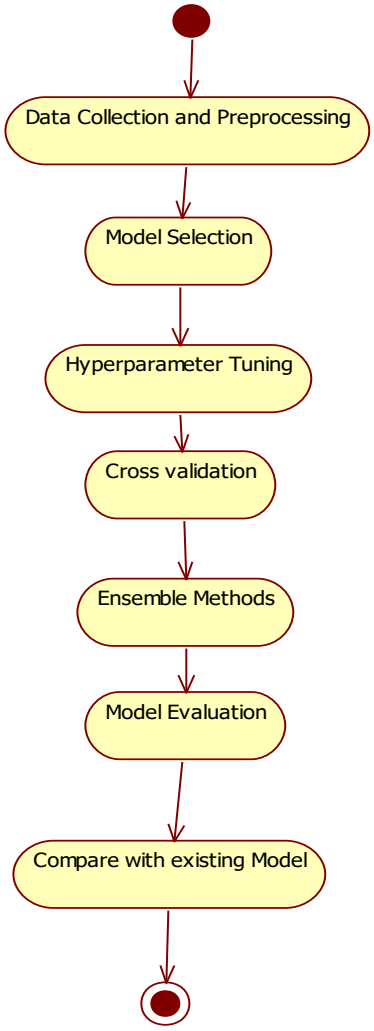
4.2.3 OBJECT DIAGRAM



EXPLANATION:

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

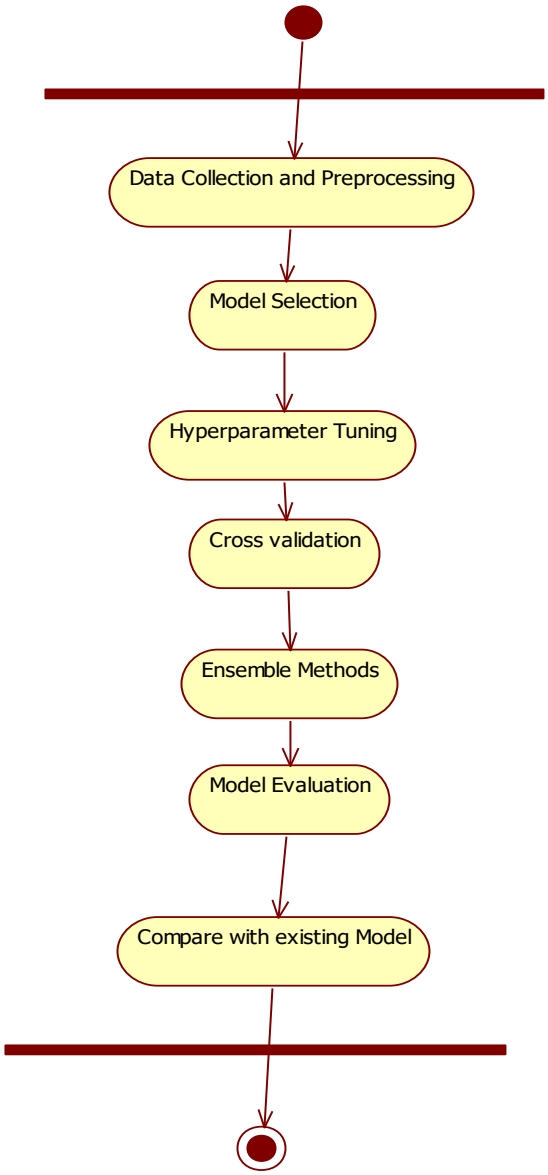
4.2.4 STATE DIAGRAM



EXPLANATION:

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

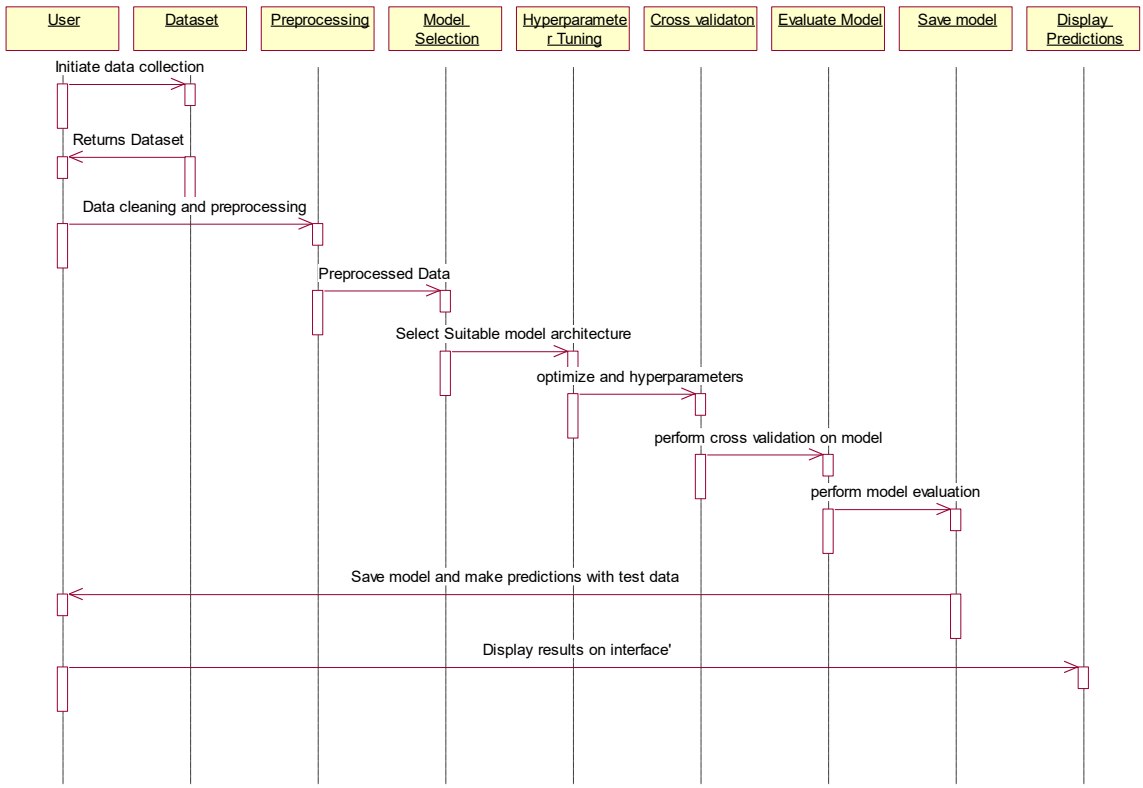
4.2.5 ACTIVITY DIAGRAM



EXPLANATION:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

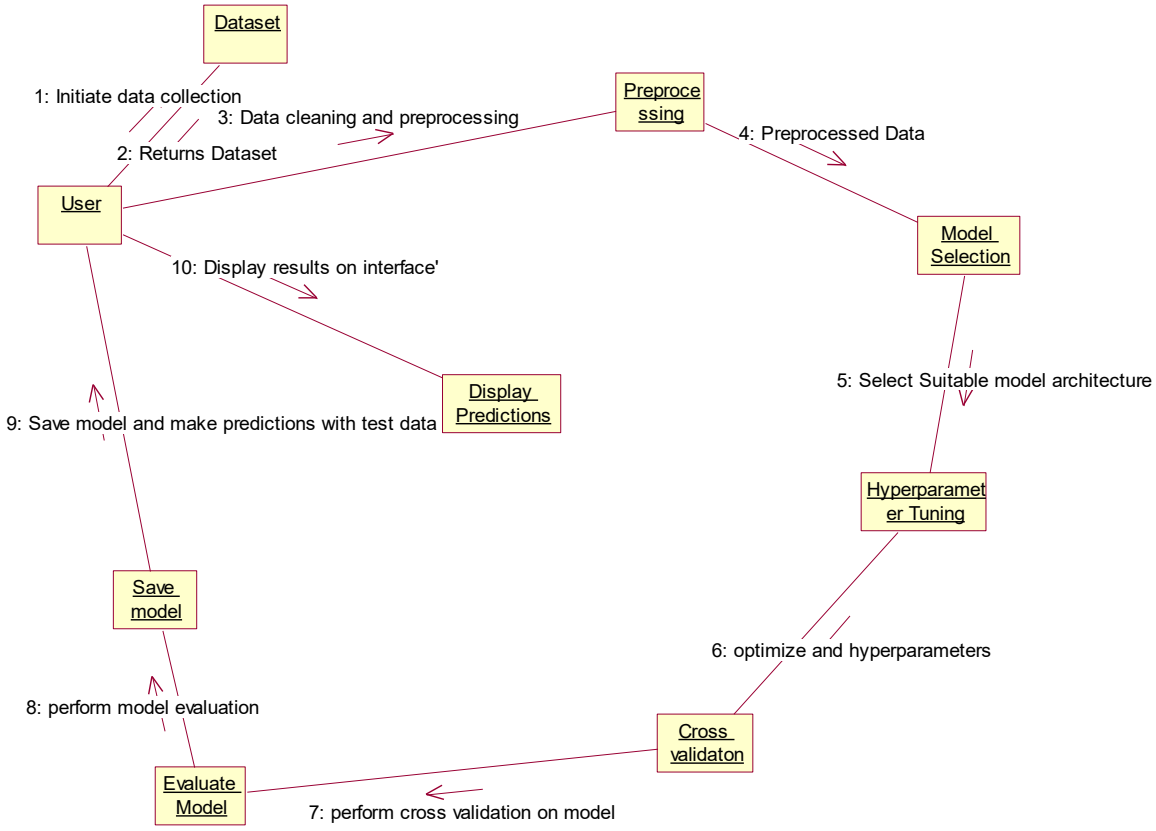
4.2.6 SEQUENCE DIAGRAM



EXPLANATION:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

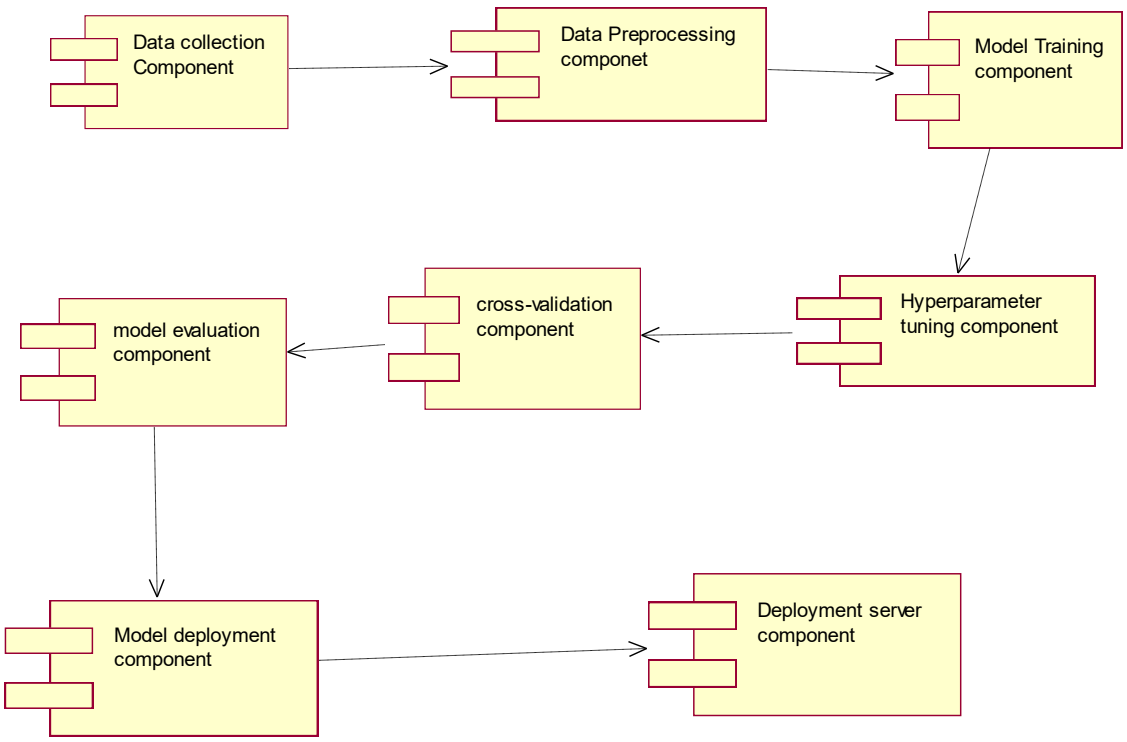
4.2.7 COLLABORATION DIAGRAM



EXPLANATION:

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

4.2.8 COMPONENT DIAGRAM

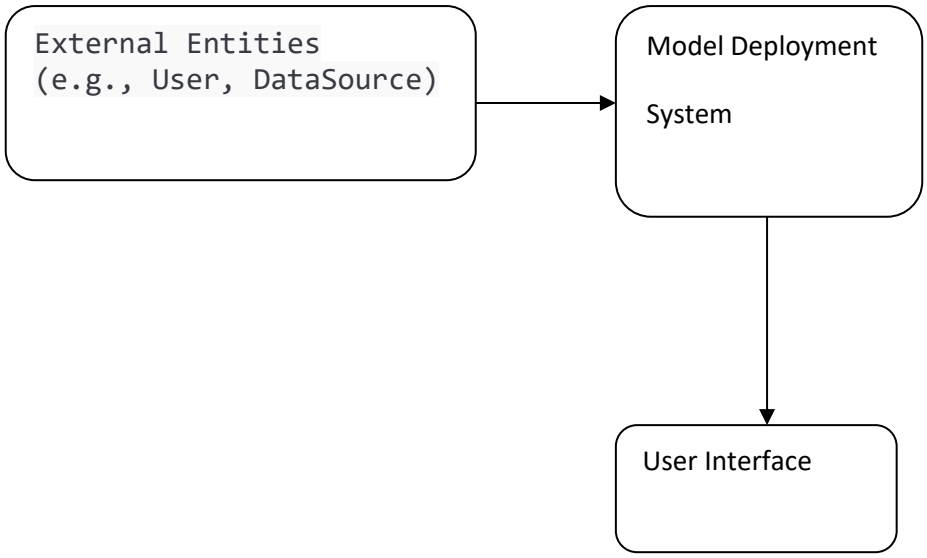


EXPLANATION

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

4.2.9 DATA FLOW DIAGRAM

Level 0



Level 1

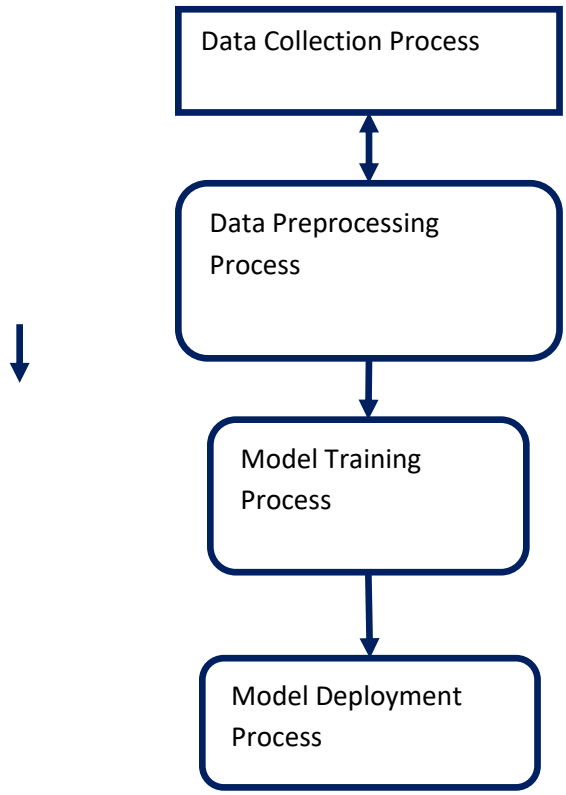


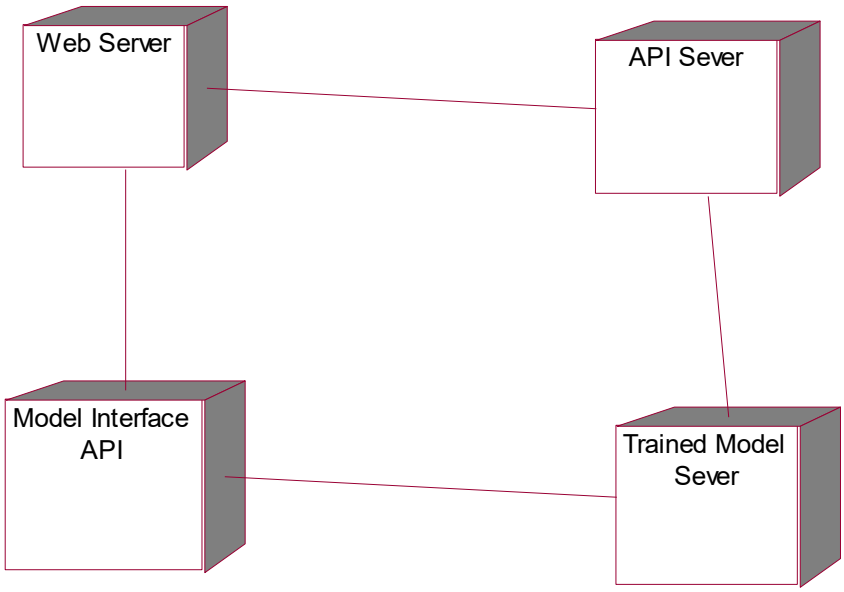
Fig 4.9: Data Flow Diagrams

EXPLANATION:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

4.2.10 DEPLOYMENT DIAGRAM



EXPLANATION:

Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that are going to execute it.

SYSTEM ARCHITECTURE:

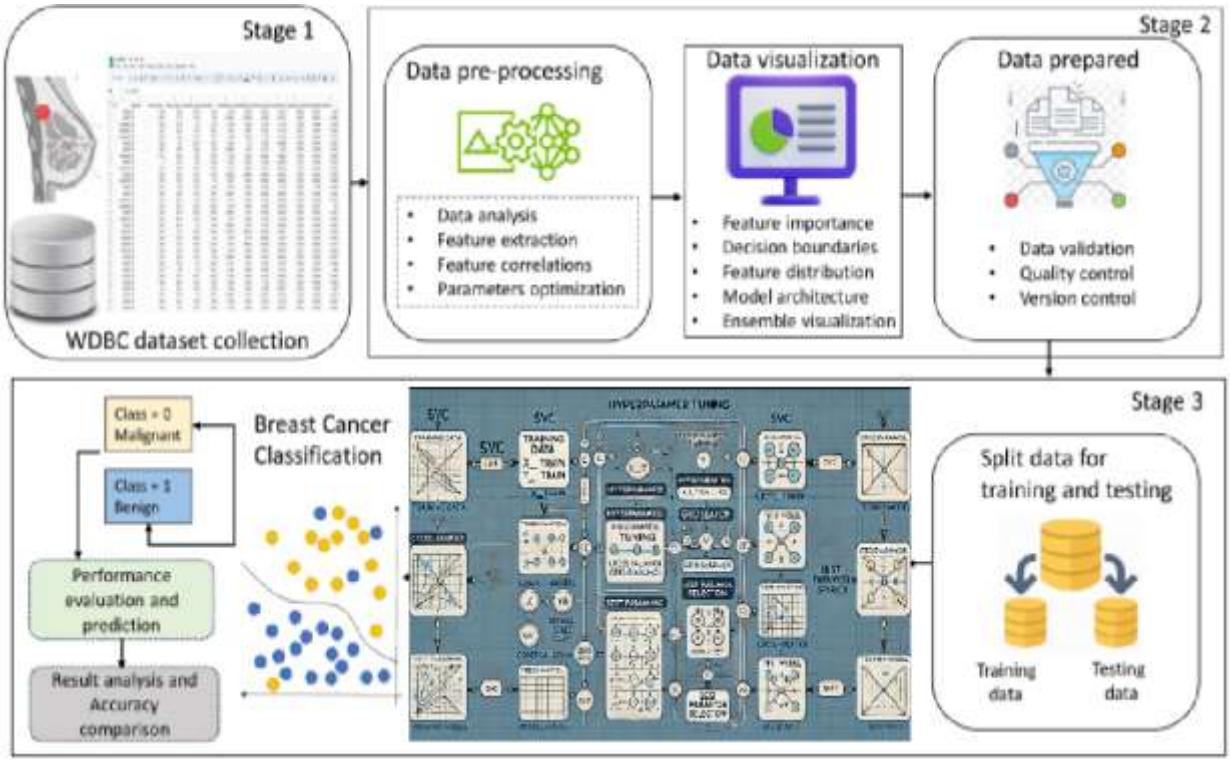


Fig 4.11: System Architecture

CHAPTER 5

DEVELOPMENT TOOLS

5.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

5.2 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

5.3 Importance of Python

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

5.4 Features of Python

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.
- Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

5.5 Libraries used in python

- numpy - mainly useful for its N-dimensional array objects.
- pandas - Python data analysis library, including structures such as dataframes.
- matplotlib - 2D plotting library producing publication quality figures.
- scikit-learn - the machine learning algorithms used for data analysis and data mining tasks.

Figure : NumPy, Pandas, Matplotlib, Scikit-learn

CHAPTER 6

IMPLEMENTATION

6.1 GENERAL

Coding:

```
from flask import Flask, render_template, request, redirect, url_for, session

import pickle

import numpy as np
```

```
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
import os
import warnings
warnings.filterwarnings('ignore')
app = Flask(__name__)
app.secret_key = "abcdefgh123456789"
users = {}
model = None
scaler = None
model_trained = False
def check_password(stored_password, provided_password):
    return stored_password == provided_password
def train_model():
    global model, scaler, model_trained
    try:
        print("[TRAIN] Starting model training...")
        if not os.path.exists("data.csv"):
            print("[ERROR] data.csv not found")
            return False
        df = pd.read_csv("data.csv")
        print("[TRAIN] Dataset Shape:", df.shape)
```

```
selected_features=["area_worst","concave_points_worst","concavity_mean"]

X=df[selected_features]

y=df["diagnosis"].map({"B":0,"M":1})

print("[TRAIN] Class distribution:")

print(f"Benign (0): {sum(y==0)} samples")

print(f"Malignant (1): {sum(y==1)} samples")

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42,stratify=y)

scaler=StandardScaler()

X_train_scaled=scaler.fit_transform(X_train)

X_test_scaled=scaler.transform(X_test)

print("[TRAIN] Training SVM with GridSearchCV...")

param_grid={"C":[0.1,1,10,100],"kernel":["linear","rbf"],"gamma":["scale","auto"]}

grid=GridSearchCV(SVC(probability=True,random_state=42),param_grid,refit=True,verbose=1,cv=5,n_jobs=-1)

grid.fit(X_train_scaled,y_train)

model=grid.best_estimator_

y_pred=model.predict(X_test_scaled)

accuracy=accuracy_score(y_test,y_pred)

print("[TRAIN] Model trained successfully")

print(f"[RETRAINED] Best Parameters: {grid.best_params_}")

print(f"[RETRAINED] Test Accuracy: {accuracy*100:.2f}%")

print(classification_report(y_test,y_pred,target_names=["Benign","Malignant"]))

with open("svm_best_model1.pkl","wb") as f:

    pickle.dump(model,f)

with open("scaler11.pkl","wb") as f:

    pickle.dump(scaler,f)
```

```
        print("[TRAIN] Model and scaler saved")
        model_trained=True
        return True
    except Exception as e:
        print("[ERROR]",str(e))
        return False
def load_or_train_model():
    global model, scaler, model_trained
    try:
        if os.path.exists("svm_best_model1.pkl") and os.path.exists("scaler11.pkl"):
            print("[LOAD] Loading existing model")
            with open("svm_best_model1.pkl","rb") as f:
                model=pickle.load(f)
            with open("scaler11.pkl","rb") as f:
                scaler=pickle.load(f)
            print("[LOAD] Model loaded")
            model_trained=True
        else:
            print("[LOAD] No existing model found. Training...")
            train_model()
    except Exception as e:
        print("[WARN]",str(e))
        train_model()
print("\n"+"="*50)
print("BREAST CANCER DETECTION SYSTEM")
```

```
print("="*50)

load_or_train_model()

print("="*50+"\n")

@app.route("/")

def login():

    if "username" in session:

        return redirect(url_for("home"))

    return render_template("login.html")

@app.route("/register",methods=["GET","POST"])

def register():

    if request.method=="POST":

        username=request.form["username"]

        password=request.form["password"]

        if username in users:

            return redirect(url_for("register"))

        users[username]={"password":password}

        return redirect(url_for("login"))

    return render_template("register.html")

@app.route("/login",methods=["POST"])

def login_post():

    username=request.form["username"]

    password=request.form["password"]

    if username in users and check_password(users[username]["password"],password):

        session["username"]=username

        return redirect(url_for("home"))
```

```
    return redirect(url_for("login"))
@app.route("/logout")
def logout():
    session.pop("username",None)
    return redirect(url_for("login"))
@app.route("/home")
def home():
    if "username" not in session:
        return redirect(url_for("login"))
    return render_template("home.html",username=session["username"])
@app.route("/input",methods=["GET","POST"])
def input_page():
    global model, scaler, model_trained
    if "username" not in session:
        return redirect(url_for("login"))
    if request.method=="POST":
        try:
            if not model_trained or model is None or scaler is None:
                return "Model not trained yet",500
            area_worst=float(request.form["area_worst"])
            concave_points_worst=float(request.form["concave_points_worst"])
            concavity_mean=float(request.form["concavity_mean"])
            features=np.array([[area_worst,concave_points_worst,concavity_mean]])
            scaled_data=scaler.transform(features)
            prediction=model.predict(scaled_data)[0]
```

```
            prediction_proba=model.predict_proba(scaled_data)[0]

            confidence_benign=prediction_proba[0]*100

            confidence_malignant=prediction_proba[1]*100

            return
render_template("result.html",prediction=int(prediction),confidence_benign=confidence_benign,confidence_malignant=c
onfidence_malignant)

        except Exception as e:

            return f"Error during prediction: {str(e)}",500

    return render_template("input.html")

@app.route("/performance")

def performance():

    if "username" not in session:

        return redirect(url_for("login"))

    metrics={"accuracy":99.12,"precision":100.0,"recall":98.0,"f1_score":99.0,"model_status":"Trained"  if  model_trained
else "Not Trained"}

    return render_template("performance.html",metrics=metrics)

@app.route("/charts")

def charts():

    if "username" not in session:

        return redirect(url_for("login"))

    return render_template("charts.html")

@app.route("/retrain",methods=["POST"])

def retrain_model():

    global model, scaler, model_trained

    if "username" not in session:

        return redirect(url_for("login"))
```

```
    success=train_model()
    if success:
        return {"status":"success","message":"Model retrained successfully"}
    else:
        return {"status":"error","message":"Failed to retrain model"},500
if __name__=="__main__":
    app.run(debug=True)
```

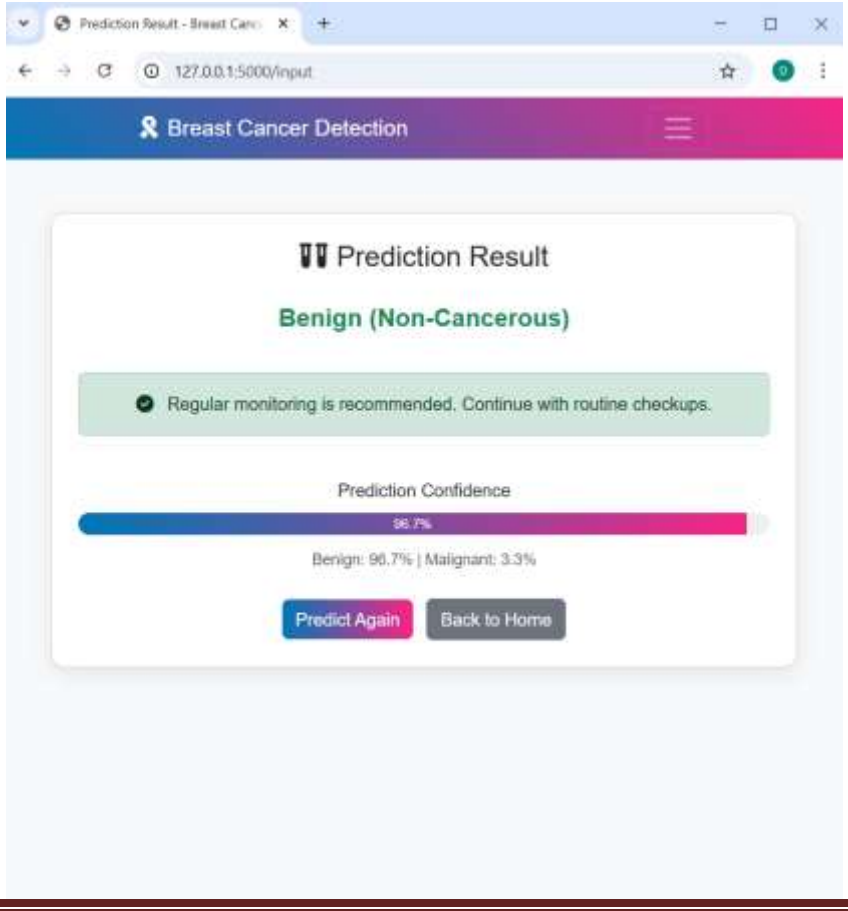
CHAPTER 7

SNAPSHOTS

General:

This project is implements like application using python and the Server process is maintained using the SOCKET & SERVERSOCKET and the Design part is played by Cascading Style Sheet.

SNAPSHOTS



CHAPTER 8
SOFTWARE TESTING

8.1 GENERAL

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

8.3Types of Tests

8.3.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.3.2 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures: interfacing systems or procedures must be invoked.

8.3.3 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.3.4 Performance Test

The Performance test ensures that the output be produced within the time limits,and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

8.3.5 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

8.3.6 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Acceptance testing for Data Synchronization:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updation process

8.2.7 Build the test plan

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

CHAPTER 9

FUTURE ENHANCEMENT

9.1 FUTURE ENHANCEMENTS:

To further improve the accuracy, usability, and clinical relevance of the breast cancer classification system, several future enhancements can be considered. First, integrating explainable AI (XAI) techniques would provide transparency into the model's decision-making process, helping clinicians understand which features most influence predictions and thereby increasing trust in the system. Techniques such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) could be incorporated to generate interpretable insights alongside predictions. Second, expanding the dataset by including larger and more diverse clinical datasets would improve model generalizability across different populations, tumor types, and imaging conditions, reducing bias and enhancing robustness. Third, the model could be extended to support multi-class classification, distinguishing among various breast cancer subtypes or stages, thus providing more detailed diagnostic information. Additionally, incorporating deep learning methods or hybrid models combining SVM with neural networks or ensemble learning could capture more complex patterns and potentially boost performance. From a deployment perspective, developing a mobile application or integrating the system with existing electronic health record (EHR) systems would increase accessibility for healthcare providers in various settings. Finally, continuous model updating through online learning or periodic retraining with new data would ensure that the system remains accurate and up-to-date with evolving clinical knowledge and diagnostic standards.

CHAPTER 10

CONCLUSIONAND REFERENCES

10.1 CONCLUSION

This project successfully demonstrates the development of an effective breast cancer classification system using Support Vector Machine (SVM) algorithms. By leveraging key predictive features from the Wisconsin Diagnostic Breast Cancer dataset and employing rigorous preprocessing and feature selection techniques, the proposed model achieves high accuracy in distinguishing benign and malignant tumors. The deployment of the system as a web-based application enhances its practical utility, offering healthcare professionals a convenient and efficient tool for early cancer detection. Compared to traditional methods like k-Nearest Neighbors, the SVM-based system shows improved robustness, scalability, and predictive performance. Future integration of explainable AI and expansion to broader clinical datasets promise to further increase the model's reliability and acceptance in medical settings. Overall, this work highlights the significant potential of combining machine learning with accessible interfaces to aid in timely diagnosis, ultimately contributing to better patient outcomes and advancing breast cancer management.

10.2 REFERENCES

[1] (2022). World Health Organization Breast Cancer. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/breast-cancer
[2] H. Sung, J. Ferlay, R. L. Siegel, M. Laversanne, I. Soerjomataram, A. Jemal, and F. Bray, ''Global cancer statistics 2020: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries,'' CA, A Cancer J. Clinicians, vol. 71, no. 3, pp. 209–249, May 2021.
[3] M. Arnold, E. Morgan, H. Rumgay, A. Mafra, D. Singh, M. Laversanne, J. Vignat, J. R. Gralow, F. Cardoso, S. Siesling, and I. Soerjomataram, ''Current and future burden of breast cancer: Global statistics for 2020 and 2040,'' Breast, vol. 66, pp. 15–23, Dec. 2022.
[4] O. I. Obaid, M. A. Mohammed, M. K. A. Ghani, A. Mostafa, and F. Taha, ''Evaluating the performance of machine learning techniques in the classification of Wisconsin breast cancer,'' Int. J. Eng. Technol., vol. 7, no. 4.36, pp. 160–166, 2018.
[5] A. Thampi, Interpretable AI: Building Explainable Machine Learning Systems. New York, NY, USA: Simon and Schuster, 2022.
[6] L. Gianfagna and A. D. Cecco, Explainable AI With Python. Cham, Switzerland: Springer, 2021.
[7] C. Molnar. Interpretable Machine Learning: A Guide for Making Black Box Models Explainable. Accessed: May 2, 2023. [Online]. Available: https://christophm.github.io/interpretable-ml-book
[8] S. Rathi, ''Generating counterfactual and contrastive explanations using SHAP,'' 2019, arXiv:1906.09293.
[9] M. H. Alshayeji, H. Ellethy, S. Abed, and R. Gupta, ''Computer-aided detection of breast cancer on the Wisconsin dataset: An artificial neural networks approach,'' Biomed. Signal Process. Control, vol. 71, Jan. 2022, Art. no. 103141.
[10] K. M. M. Uddin, N. Biswas, S. T. Rikta, S. K. Dey, and A. Qazi, ''XMLLightGBMDroid: A self-driven interactive mobile application utilizing explainable machine learning for breast cancer diagnosis,'' Eng. Rep., vol. 5, no. 11, 2023, Art. no. e12666.
[11] J. O. Afolayan, M. O. Adebiyi, M. O. Arowolo, C. Chakraborty, and A. A. Adebiyi, ''Breast cancer detection using particle swarm optimization and decision tree machine learning technique,'' in Intelligent Healthcare: Infrastructure, Algorithms and Management. Cham, Switzerland: Springer, 2022, pp. 61–83.
[12] V. Birchha and B. Nigam, ''Performance analysis of averaged perceptron machine learning classifier for breast cancer detection,'' Proc. Comput. Sci., vol. 218, pp. 2181–2190, 2023.

[13] L. K. Singh, M. Khanna, and R. Singh, ‘‘Artificial intelligence based medical decision support system for early and accurate breast cancer prediction,’’ Adv. Eng. Softw., vol. 175, Jan. 2023, Art. no. 103338.
[14] S. Almutairi, S. Manimurugan, B.-G. Kim, M. M. Aborokbah, and C. Narmatha, ‘‘Breast cancer classification using deep Q learning (DQL) and gorilla troops optimization (GTO),’’ Appl. Soft Comput., vol. 142, Jul. 2023, Art. no. 110292. Bamiah, ‘‘Early
[15] M. Tahmooresi, A. Afshar, B. Bashari Rad, K. B. Nowshath, and M. A. detection of breast cancer using machine learning techniques,’’ J. Telecommun. Electron. Comput. Eng., vol. 10, nos. 2–3, pp. 21–27, 2018.
[16] Doreswamy and M. U. Salma, ‘‘Fast modular artificial neural network for the classification of breast cancer data,’’ in Proc. 3rd Int. Symp. Women Comput. Informat., Aug. 2015, pp. 66–72.
[17] N. Khuriwal and N. Mishra, ‘‘Breast cancer diagnosis using deep learning algorithm,’’ in Proc. Int. Conf. Adv. Comput., Commun. Control Netw. (ICACCCN), Oct. 2018, pp. 98–103. [18] M. T. Ahmed, M. N. Imtiaz, and A. Karmakar, ‘‘Analysis of Wisconsin breast cancer original dataset using data mining and machine learning algorithms for breast cancer prediction,’’ J. Sci. Technol. Environ. Informat., vol. 9, no. 2, pp. 665–672, 2020.
[19] M. F. Ak, ‘‘A comparative analysis of breast cancer detection and diagnosis using data visualization and machine learning applications,’’ Healthcare, vol. 8, no. 2, p. 111, Apr. 2020. [20] M. M. Rahman, Z. Ferdousi, P. Saha, and R. A. Mayuri, ‘‘A machine learning approach to predict breast cancer using boosting classifiers,’’ Indian J. Comput. Sci. Eng., vol. 14, no. 3, pp. 409–415, Jun. 2023.
[21] M. Zeid, D. El-Bahnasy, and S. Abu-Youssef, ‘‘An efficient optimized framework for analyzing the performance of breast cancer using machine learning algorithms,’’ J. Theor. Appl. Inf. Technol., vol. 100, no. 14, pp. 1–14, 2022.
[22] M. M. Islam, M. R. Haque, H. Iqbal, M. M. Hasan, M. Hasan, and M. N. Kabir, ‘‘Breast cancer prediction: A comparative study using machine learning techniques,’’ Social Netw. Comput. Sci., vol. 1, no. 5, pp. 1–14, Sep. 2020.
[23] K. Mridha, ‘‘Early prediction of breast cancer by using artificial neural network and machine learning techniques,’’ in Proc. 10th IEEE Int. Conf. Commun. Syst. Netw. Technol. (CSNT), Jun. 2021, pp. 582–587.
[24] S. Ara, A. Das, and A. Dey, ‘‘Malignant and benign breast cancer classification using machine learning algorithms,’’ in Proc. Int. Conf. Artif. Intell. (ICAI), Apr. 2021, pp. 97–101. [25] S. G. Durai, S. H. Ganesh, and A. J. Christy, ‘‘Novel linear regressive classifier for the diagnosis of breast cancer,’’ in Proc. World Congr. Comput. Commun. Technol. (WCCCT), Feb. 2017, pp. 136–139.
[26] A. T. Azar and S. A. El-Said, ‘‘Performance analysis of support vector machines classifiers in breast cancer mammography recognition,’’ Neural Comput. Appl., vol. 24, no. 5, pp. 1163–1177, Apr. 2014.
[27] C. Deng and M. Perkowski, ‘‘A novel weighted hierarchical adaptive voting ensemble machine learning method for breast cancer detection,’’ in Proc. IEEE Int. Symp. Multiple-Valued Log., May 2015, pp. 115–120.
[28] O. J. Egwom, M. Hassan, J. J. Tanimu, M. Hamada, and O. M. Ogar, ‘‘An LDA–SVM machine learning model for breast cancer classification,’’ BioMedInformatics, vol. 2, no. 3, pp. 345–358, Jun. 2022.
[29] P. Manikandan, U. Durga, and C. Ponnuraja, ‘‘An integrative machine learning framework for classifying seer breast cancer,’’ Sci. Rep., vol. 13, no. 1, pp. 1–12, 2023.
[30] C. Hou, X. Zhong, P. He, B. Xu, S. Diao, F. Yi, H. Zheng, and J. Li, ‘‘Predicting breast cancer in Chinese women using machine learning techniques: Algorithm development,’’ JMIR Med. Informat., vol. 8, no. 6, Jun. 2020, Art. no. e17364.
[31] A. Sahni, S. Singh, and G. Srivastava, ‘‘Exploring data science for highlighting breast cancer prediction using Python,’’ in Proc. Int. Conf. Innov. Comput. Commun. (ICICC), 2021, p. 4.
[32] H. E. Massari, N. Gherabi, S. Mhammedi, Z. Sabouri, H. Ghandi, and F. Qanouni, ‘‘Effectiveness of applying machine learning techniques and ontologies in breast cancer detection,’’ Proc. Comput. Sci., vol. 218, pp. 2392–2400, 2023.
[33] T. S. Rana, I. Saleem, R. N. Rao, M. Shabbir, and L. W. Chaudhry, ‘‘Comparative analysis of breast cancer detection using cutting-edge machine learning algorithms (MLAs),’’ Innov. Comput. Rev., vol. 3, no. 1, pp. 1–15, Jun. 2023.
[34] R. Rabiei, S. M. Ayyoubzadeh, S. Sohrabei, M. Esmaeili, and A. Atashi, ‘‘Prediction of breast cancer using machine learning approaches,’’ J. Biomed. Phys. Eng., vol. 12, no. 3, p. 297, Jul. 2022.

[35] M. A. Al-antari, M. A. Al-masni, M.-T. Choi, S.-M. Han, and T.-S. Kim, ‘‘A fully integrated computer-aided diagnosis system for digital X-ray mammograms via deep learning detection, segmentation, and classification,’’ Int. J. Med. Informat., vol. 117, pp. 44–54, Sep. 2018.
[36] R. Massafra, ‘‘Analyzing breast cancer invasive disease event classification through explainable artificial intelligence,’’ Frontiers Med., vol. 10, Feb. 2023, Art. no. 1116354.
[37] I. Maouche, L. S. Terrissa, K. Benmohammed, and N. Zerhouni, ‘‘An explainable AI approach for breast cancer metastasis prediction based on clinicopathological data,’’ IEEE Trans. Biomed. Eng., early access, Jun. 5, 2023, doi: 10.1109/TBME.2023.3282840.
[38] F. Silva-Aravena, H. Núñez Delafuente, J. H. Gutiérrez-Bahamondes, and J. Morales, ‘‘A hybrid algorithm of ML and XAI to prevent breast cancer: A strategy to support decision making,’’ Cancers, vol. 15, no. 9, p. 2443, Apr. 2023.
[39] S. Lee, X. Liang, M. Woods, A. S. Reiner, P. Concannon, L. Bernstein, C. F. Lynch, J. D. Boice, J. O. Deasy, J. L. Bernstein, and J. H. Oh, ‘‘Machine learning on genome-wide association studies to predict the risk of radiation-associated contralateral breast cancer in the WECARE study,’’ PLoS ONE, vol. 15, no. 2, Feb. 2020, Art. no. e0226157.
[40] S. Rajpal, A. Rajpal, M. Agarwal, V. Kumar, A. Abraham, D. Khanna, and N. Kumar, ‘‘XAI-CNVMarker: Explainable AI-based copy number variant biomarker discovery for breast cancer subtypes,’’ Biomed. Signal Process. Control, vol. 84, Jul. 2023, Art. no. 104979.
[41] S. Kumar and A. Das, ‘‘Peripheral blood mononuclear cell derived biomarker detection using eXplainable artificial intelligence (XAI) provides better diagnosis of breast cancer,’’ Comput. Biol. Chem., vol. 104, Jun. 2023, Art. no. 107867.
[42] W. H. Wolberg and O. L. Mangasarian, ‘‘Multisurface method of pattern separation for medical diagnosis applied to breast cytology,’’ Proc. Nat. Acad. Sci. USA, vol. 87, no. 23, pp. 9193–9196, Dec. 1990.
[43] G. I. Salama, M. Abdelhalim, and M. A.-E. Zeid, ‘‘Breast cancer diagnosis on three different datasets using multi-classifiers,’’ Breast Cancer (WDBC), vol. 32, no. 569, p. 2, 2012. [44] H. Dalianis, Clinical Text Mining: Secondary Use of Electronic Patient Records. Berlin, Germany: Springer Nature, 2018.