LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CU6051NI Artificial Intelligence

25% Individual Coursework

Submission: Milestone 1

Academic Semester: Autumn Semester 2025

Credit: 15 credit semester long module

Student Name: Anurodh Prasain

London Met ID: 23047512

College ID: np01cp4a230012

Assignment Due Date: 17/12/2025.

Assignment Submission Date: 17/12/2025

Submitted To: Mr. Alish KC

GitHub Link	*https://github.com/Anurodh0011/BREAST-CANCER-DETECTION*

Table of Contents

Table of Figures

Table of Tables

1. Introduction

Artificial Intelligence (AI) is a disruptive technology providing computers with capabilities to learn, reason, understand language and analyze data, in a manner like humans. It is based on various disciplines such as computer science, linguistics and neuroscience. In essence, AI tries to simulate human cognitive functions, such as the ability to comprehend the world and derive some ideas and obtain valuable insights out of information. One such usage is Optical Character Recognition (OCR), which transforms unstructured images and documents into structured and usable data. In general, AI can be used as an effective means of bringing positive transformation to society (Google Cloud, 2025).

Machine learning is the subdivision of artificial intelligence (AI) that deals with the algorithms that can learn the trends of the training data and, when properly trained, become able to effectively infer the new data. This pattern recognition capability allows machine learning models to take decisions or predictions without clear-cut or hard-coded guidance (Bergmann, 2025).

Any machine learning approach can be classified into one of three different learning paradigms:

- **Supervised learning:** It is a method that will train a model to give a correct prediction of the output that should be given based on an input. It is applicable to problems where some accuracy with respect to some external ground truth is required, e.g. classification or regression (Bergmann, 2025).

- **Unsupervised learning:** It uses a model to identify inherent patterns, dependencies and correlations in the data. Unsupervised learning tasks do not have any external ground truth with which the results of a particular learning task can be compared, unlike supervised learning (Bergmann, 2025).

- **Reinforcement learning (RL):** It is a process that trains a model to analyze its world and make a decision that will result in the highest reward. The situations in RL do not imply

the presence of a single ground truth, but they imply the presence of good and bad (or neutral) actions (Bergmann, 2025).

In supervised learning, tasks can be divided into **classification** and **regression** depending on the nature of the target variable. Since the selected dataset has the target variable **Diagnosis** with two categories (*Malignant* and *Benign*), this problem is a binary classification task, making classification algorithms suitable for modelling.
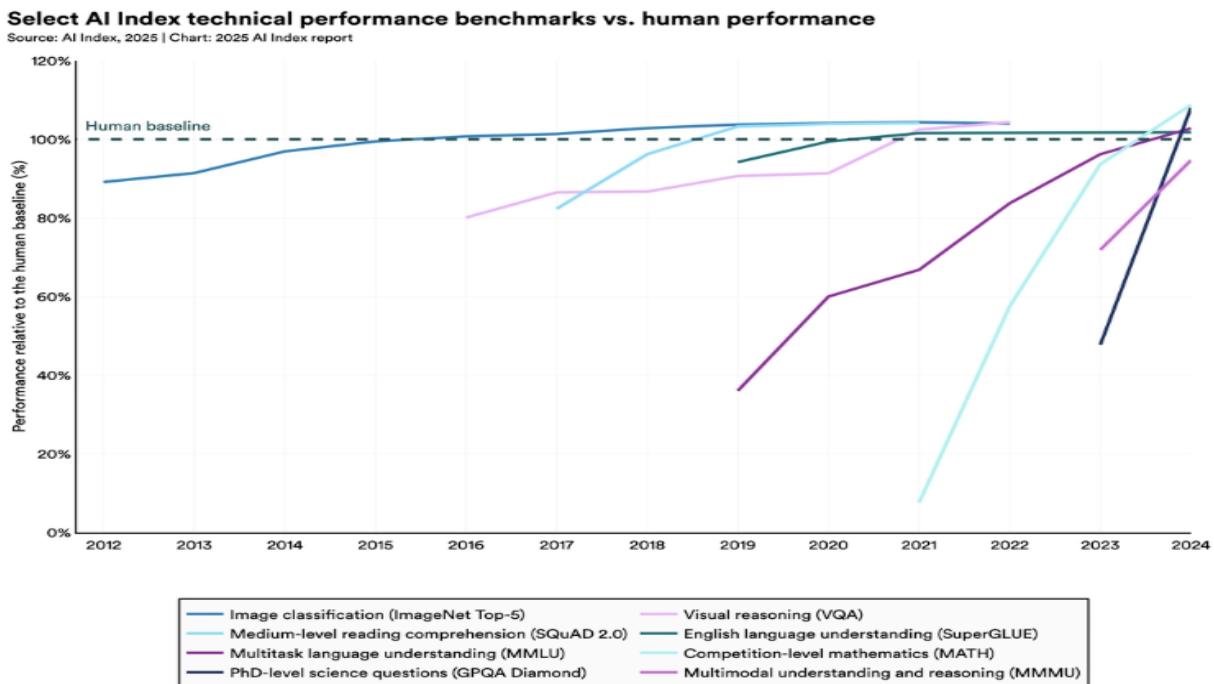
Figure 1: AI Index technical performance benchmarks vs human performance (Stanford University, 2025).

1.1.   Problem Domain: Breast Cancer Diagnosis

Breast cancer is a malignant tumour that occurs due to uncontrolled proliferation of breast cells, and it is one of the major causes of death because of cancer in women in the world. Early diagnosis and treatment are great positives in terms of the survival rates and the choice of the treatment, which lowers the mortality and healthcare costs. Machine learning has become a significant instrument to aid in clinical decision-making by training diagnostic patterns based on the patient information and medical images. Various researchers have shown the success of such supervised learning frameworks as logistic regression, K-Nearest Neighbours (KNN), decision

tree classifiers and others to classify breast tumours based on such features as cell size, texture and morphology (Aryan Sai Boddu, 2025).

1.2. Models Selected for the Study

To address the binary classification problem in this dataset the following supervised learning algorithms are applied in the coursework:

- Logistic Regression: a statistical classification technique that is common in binary outcomes and offers probabilities and interpretable coefficients that aid in the understanding of the effects of features.
- K-Nearest Neighbours (KNN): an instance-based learning algorithm which classifies instances based on the classification of nearest instances in feature space, showing how similarity measurements can be employed in classification problems.
- Decision Tree Classifier: is a supervised learning model based on rules, which divides a dataset based on characteristic thresholds to create rational decision pathways; it is also user-friendly and visualizable.

Based on these models, a variety of classification strategies, among them probabilistic, instance-based and tree-structured approaches, was selected, which allowed to compare the performance and interpretability of the task of tumour classification in a comprehensive manner (Yoo-Shin Park, 2025).

1.3. Objective of this project

This coursework has the following objectives:

- The fundamentals of the supervised machine learning classification.
- The use of logistic regression, KNN, and decision tree classifiers on the selected data.
- The measure and comparison of performance of these algorithms based on relevant measures.
- Determining which model can best be used to predict breast cancer diagnosis.
- Proving the practical relevance of supervised learning to medical decision support.

2. Background

2.1. Analysis of the Existing Scenario

In 2021, the World Health Organization established the Global Breast Cancer Initiative (GBCI), to bring together stakeholders from around the world and across sectors with the shared goal of reducing global breast cancer mortality by 2.5% per year, thereby averting 2.5 million breast cancer deaths globally by 2040. The three pillars of action are: health promotion for early detection; timely diagnosis; and comprehensive breast cancer management. It is one of the most common causes of death among women all over the world, which means the critical need to detect it as early as possible and be sure of the diagnosis (WHO, 2025).

The classical methods of diagnosing like biopsy, mammography, and histopathological analysis are useful but mostly time-consuming, resource-consuming, and prone to variability in human interpretation (National Cancer Institute, 2025). Using numerical features retrieved from fine needle aspirate (FNA) images, supervised machine learning classification has demonstrated great promise in differentiating between benign and malignant tumours. Because of their structured feature space and clinical significance, publicly accessible datasets like the Breast Cancer Wisconsin Diagnostic (WBCD) dataset have emerged as a standard for assessing classification algorithms (R. Fang, 2025).

2.1. Research work done on the chosen topic/problem domain

In this section, we will examine the current research literature associated with the breast cancer prediction with the help of the supervised machine learning algorithms, i.e., classification-based ones.

2.1.1. Research Paper 1: Machine Learning Algorithms to predict and diagnose breast cancer

Problem Statement

This paper will assess and compare various supervised machine learning algorithms to determine the best classifier of breast cancer prediction and diagnosis using the Wisconsin Diagnostic dataset.
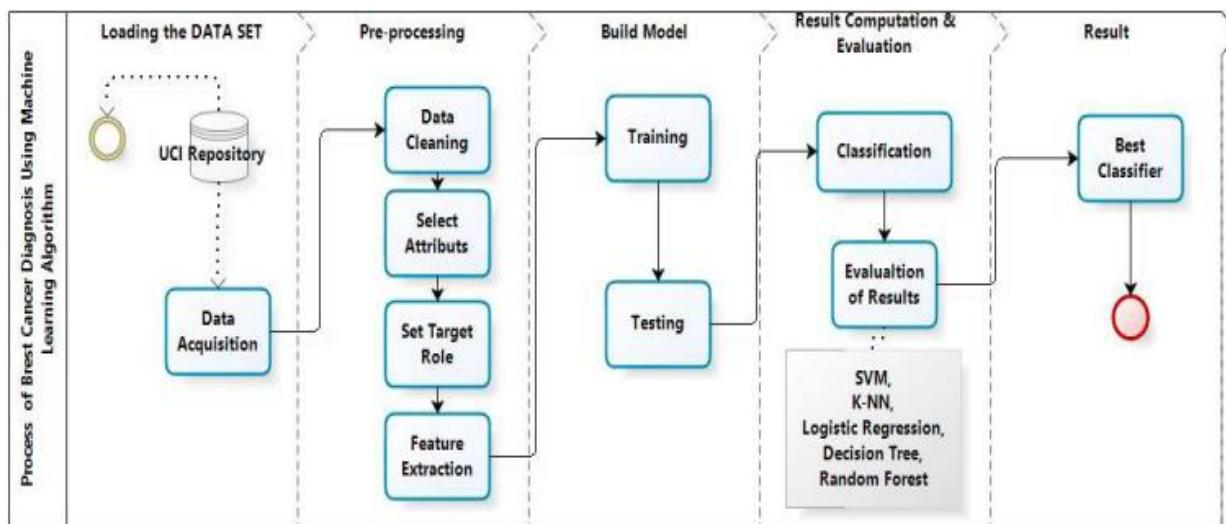
Figure 2: Process Flow Diagram of Research Paper 1 (Mohammed Amine Naji, 2021).

Algorithms Used

- Support Vector Machine (SVM)
- Random Forest
- Logistic Regression
- Decision Tree
- K-Nearest Neighbors (KNN)

Dataset Description

The authors applied the Breast Cancer Wisconsin Diagnostic dataset which consists of 569 cases where the numerical features were created because of digitization of breast cell nuclei. The target variable is used to classify tumors as benign and malignant (Mohammed Amine Naji, 2021).
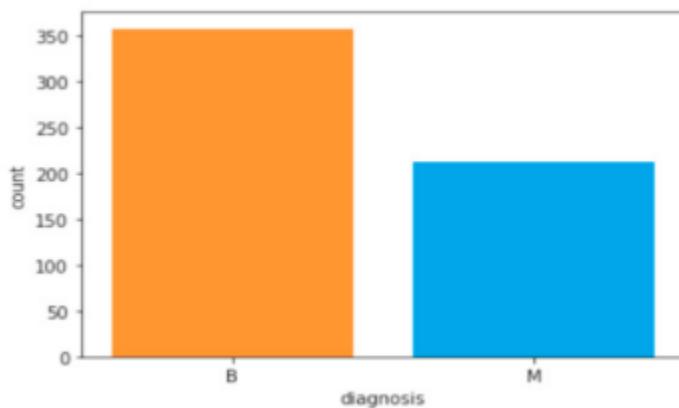
Figure 3: Wisconsin Breast Cancer Diagnostic Dataset (Mohammed Amine Naji, 2021).

Key Findings

- SVM had the best test accuracy of 97.2.
- There was also strong performance by Logistic Regression, Decision Tree and KNN.
- Evaluation metrics were given in form of confusion matrix, accuracy, precision, sensitivity, F1-score, and ROC-AUC (Mohammed Amine Naji, 2021).

Algorithm	Training Accuracy (%)	Testing Accuracy (%)
SVM	98.4	97.2
Random Forest	99.8	96.5
Logistic Regression	95.5	95.8
Decision Tree	98.8	95.1
K-NN	94.6	93.7

Table 1: Accuracy percentage for breast cancer diagnostic dataset (Mohammed Amine Naji, 2021).
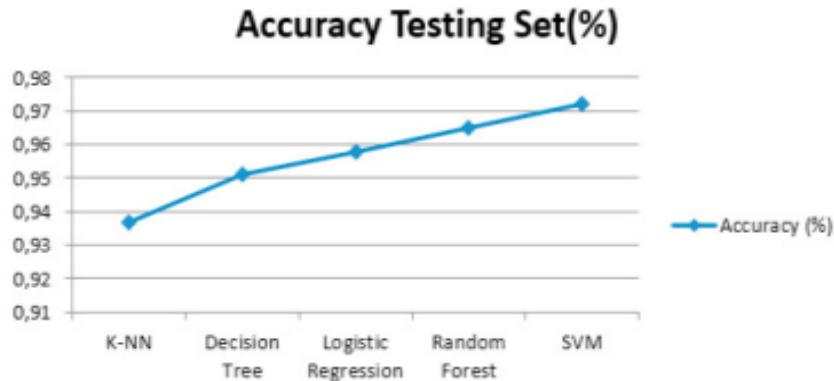
Figure 4: Comparative graph of different classifiers (Mohammed Amine Naji, 2021).

2.1.2. Research Paper 2: Breast Cancer Prediction and Diagnosis with the help of machine and modern deep learning models

Problem Statement

This paper will examine the performance of classical machine learning and state-of-the-art deep learning to detect breast cancer using the Wisconsin Diagnostic data.

Algorithms Used

- k-Nearest Neighbors (kNN)
- Naive Bayes
- Decision Tree
- Support Vector Machine
- Gradient Boosting
- CN2 Rule Inducer
- Neural Networks (Neural Decision Forest, MLP)
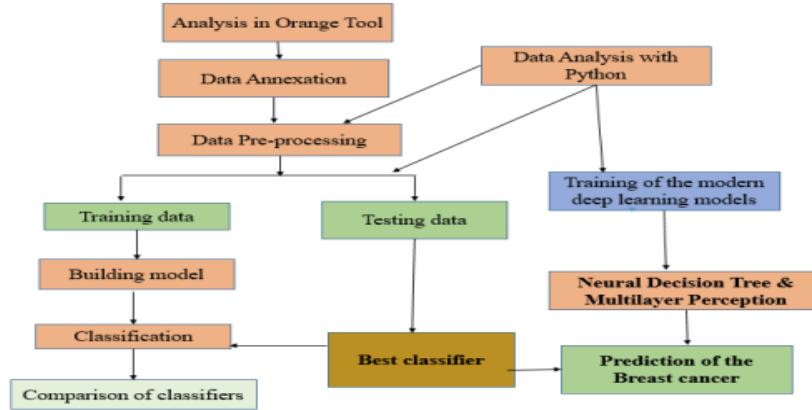
Figure 5: Flow Diagram of Research Paper 2.

Dataset Description

The data is comprised of 569 records of patients, and 30 numerical variables describe geometric and textural characteristics of cell nuclei. The data can be found in the UCI Machine Learning Repository (Seeta Devi, 2024).

Key Findings

- Gradient Boosting and CN2 Rule Inducer had perfect scores in classification.
- The performance of traditional ML models (Decision tree, Naive Bayes, kNN) was competitive.
- Deep learning models had a high ROC-AUC (0.9959) (Seeta Devi, 2024).
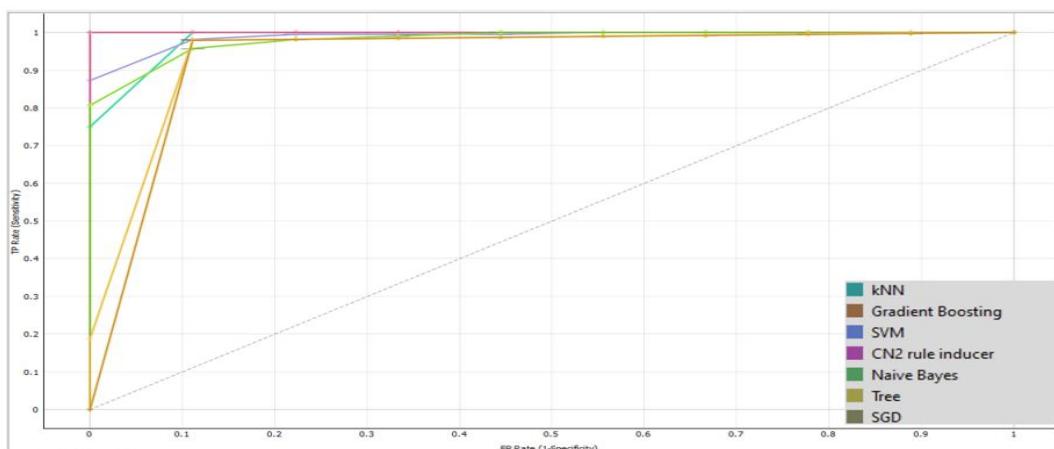
Figure 6: ROC Analysis of Malignant Tumour (Seeta Devi, 2024).

Metric	Neural Decision Forest	Multilayer Perceptron
AUC-ROC	0.9667	0.9959
Accuracy (%)	95.61	96.49
Precision (%)	100	96.57
Recall (%)	89.36	96.49
F1-Score (%)	94.38	96.50

Table 2: Prediction Output (Seeta Devi, 2024).

This implies higher overall prediction performance.

Accuracy = True Positive/True Negative + True Negative + Negative)

Precision

Precision denotes the percentage of occurrences the model is correct on all the positive instances only; it denotes as positive. The model is denoted by a high precision produces less false positive errors.

- Formula: Precision = TP / (TP + FP)

F1 Score

It portrays their harmonious average, striking a harmonious tradeoff between accuracy and recall. A score of 1 indicates a tradeoff between the two, and a value<|human|>represents a balance between the two, and a value of 0 happens when one of the two attains high elevation.

- FormulaF1 Score = 2 x Precision Recall) / (Precision + Recall))

AUC-ROC

Area Under the Receiver Operating Characteristic. Curve (AUC-ROC) is the ability of a model to differentiate positive and negative events throughout several levels of threshold. A higher AUC-ROC value represents improved performance of the model regarding classification.
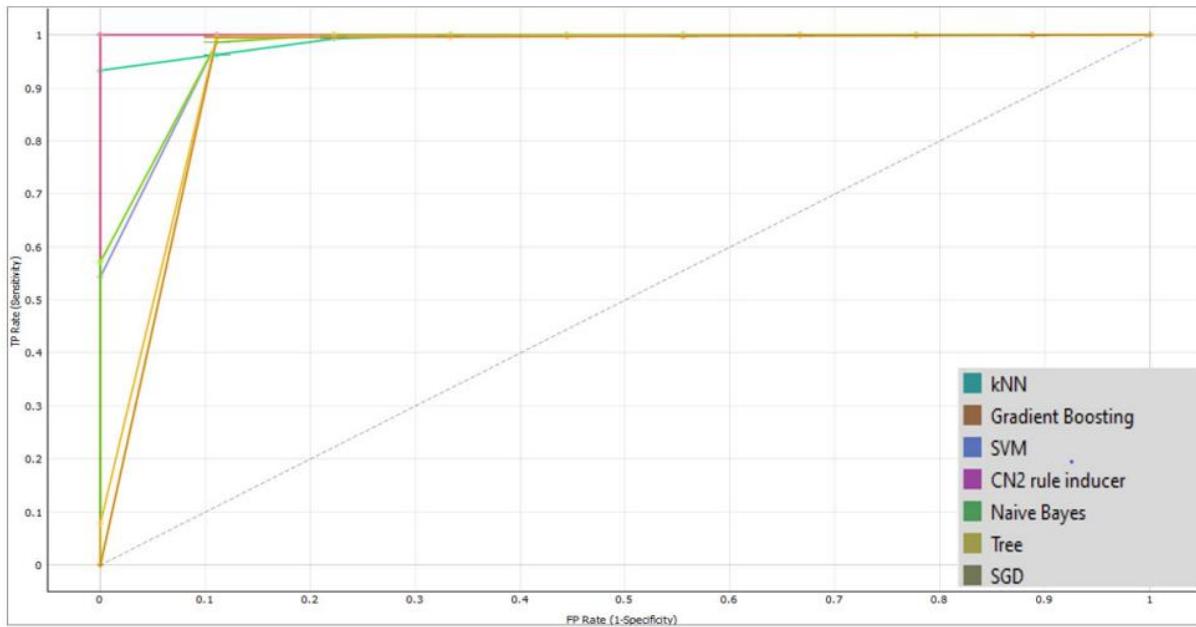
Figure 7: ROC Analysis of benign Tumour (Seeta Devi, 2024).
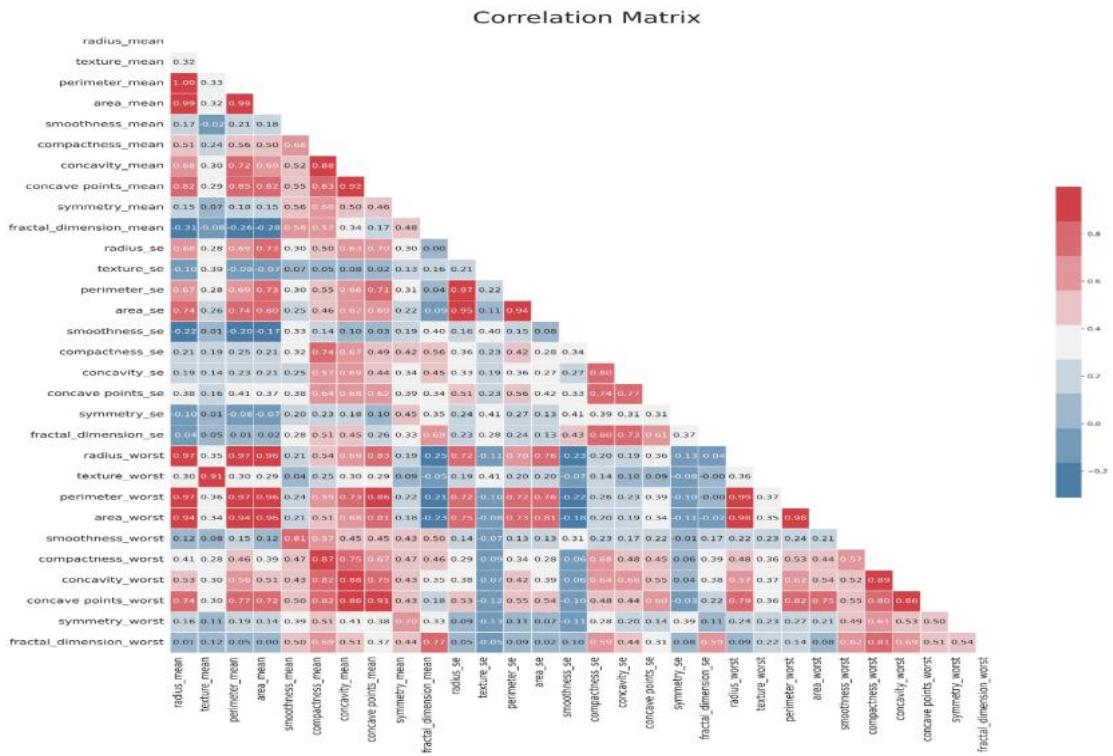
Figure 8: Correlation Matrix between Each Feature (Seeta Devi, 2024).

2.1.3. Research Paper 3: AI-based predictive modeling of breast cancer classification and explainable AI (Taminul Islam, 2024).

Problem Statement

The proposed paper deals with the issue of proper classification of breast cancer tumors using machine learning methods with explainable AI (XAI). In contrast to the old methods, it has model interpretability (with SHAP) and classification to provide clinical indications of feature contributions.

Figure 9: Ratio of malignant and benign data (Taminul Islam, 2024).

Dataset & Methodology

- The major clinical data of 500 patients was gathered in Dhaka Medical College Hospital.
- The dataset contains some structured diagnostic measures like the Wisconsin Diagnostic dataset but in a locally collected clinical setting.
- Various machine learning models were tested: Decision Tree, Random Forest, Logistic Regression, Naive Bayes and XGBoost.
- SHAP analysis has been applied to give explainable AI insights of the importance of features to classification decisions.
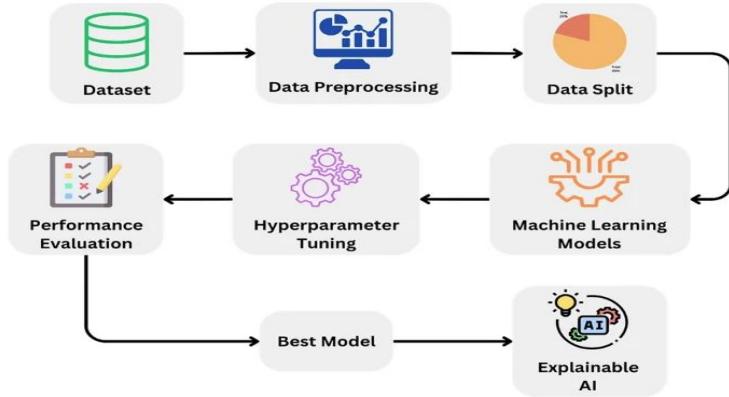
Figure 10: Visualizing the workflow of the proposed mode (Taminul Islam, 2024).

Key Findings

- The results of Logistic Regression, Decisional Tree, and Random Forest were not the best, whereas XGBoost turned out to be the most accurate (97%).
- Explainable AI algorithms such as SHAP offered pictorial information about the effect of features (such as tumor texture and area) on predictions.
- This article underscores the importance of performance and interpretability in medical artificial intelligence.

Algorithms	Hyperparameter tuning	Range	Best	Accuracy	Precision	Recall	F_1 Score
Decision tree	max_depth	None, 5, 10	5	0.91	0.94	0.89	0.9
	min_samples_leaf	2, 5, 10	4				
	min_samples_split	1, 2, 2004	5				
Random forest	max_depth	None, 5, 10, 20	None	0.96	0.93	0.95	0.94
	min_samples_leaf	1, 2, 2004	1				
	min_samples_split	2, 5, 10	5				
	n_estimators	100, 300, 500	300				
XGBoost	learning_rate	0.01, 0.1, 0.3	0.01	0.97	0.94	0.95	0.96
	max_depth	3, 5, 2007	3				
	n_estimators	100, 300, 500	500				
	subsample	0.8, 1.0	1				
Naïve Bayes	No			0.94	0.99	0.9	0.94
Logistic Regression	Regularization strength	0.001, 0.01, 0.1, 1, 10	10	0.93	0.93	0.93	0.93

Figure 11: Hyperparameter tuning with performance metrics (Taminul Islam, 2024).

With an effective precision of 0.97 and high precision, recall, and F1 scores, the XGBoost algorithm performed better than the others in terms of performance metrics. With a balanced accuracy trade-off and an accuracy of 0.96, the random forest algorithm also demonstrated strong performance. With a balanced F1 score of 0.90 and an accuracy of 0.91, the decision tree algorithm produced good results. With F1 scores of 0.94 and 0.93, respectively, Naive Bayes and logistic regression demonstrate competitive performance. All things considered, hyper-parameter tuning is crucial to enhancing the model's performance, and the algorithm selection had a big impact on the outcome; XGBoost and Random Forest stand out as high-performance models.
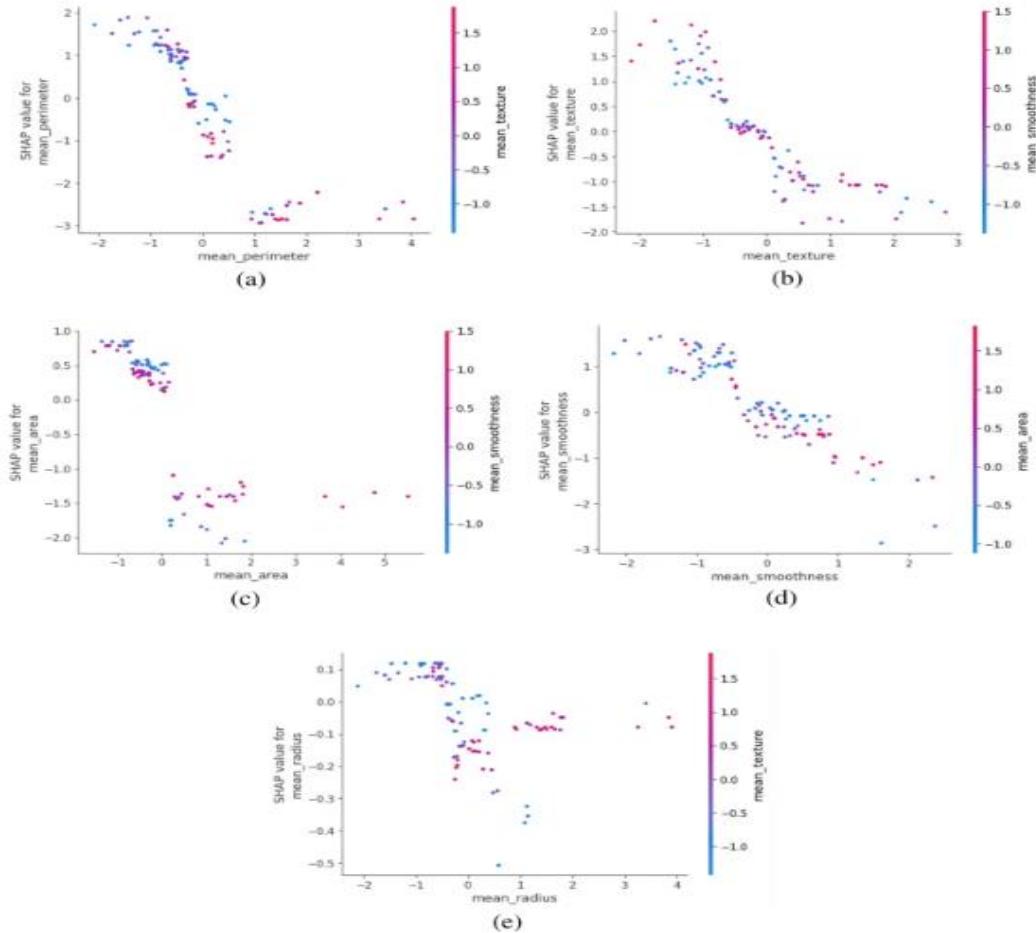
Figure 12: Dependence Plot for XGBoost Model (Taminul Islam, 2024).

2.2.4. Research Paper 4: Diagnostic Features-based Breast Cancer Diagnosis with the help of Machine Learning (Arslan Khalid, 2023).

Problem Statement

The paper under consideration explores the ability of various supervised machine learning classifiers to detect breast cancer at its early stages based on the structured diagnostic features of the tumor cell nuclei. The central question is to identify the best algorithms in clinical decision support system that offer an optimal balance between accuracy, robustness and interpretability.

Algorithms Used

- Logistic Regression
- K-Nearest Neighbors (KNN)
- Decision Tree
- Random Forest
- Support Vector Machine (SVM)

These algorithms are measured by classification tasks like finding accuracy, precision, recall, F1-score, and ROC-AUC.
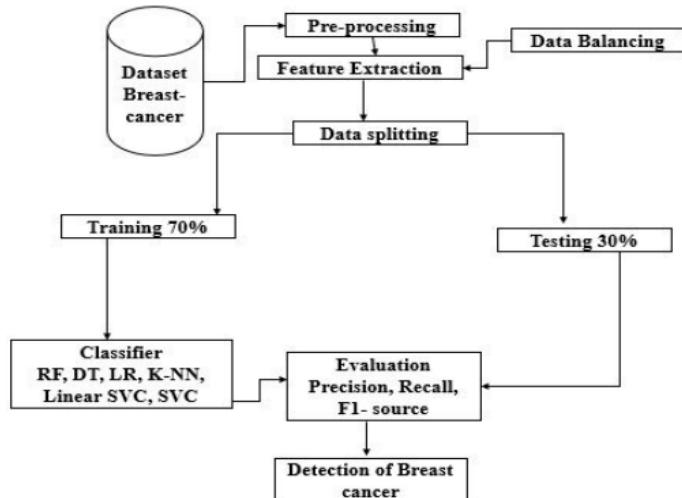
Figure 13: Proposed Methodology (Arslan Khalid, 2023).

Dataset Description

- The dataset employed in the study is that of Breast Cancer Wisconsin Diagnostic (WBCD).
- It is a dataset of 569 samples containing 30 numerical variables that were derived because of fine needle aspirate (FNA) images.
- The target variable will classify tumours as either being malignant or benign which is directly corresponding to the data in this course work (Arslan Khalid, 2023).
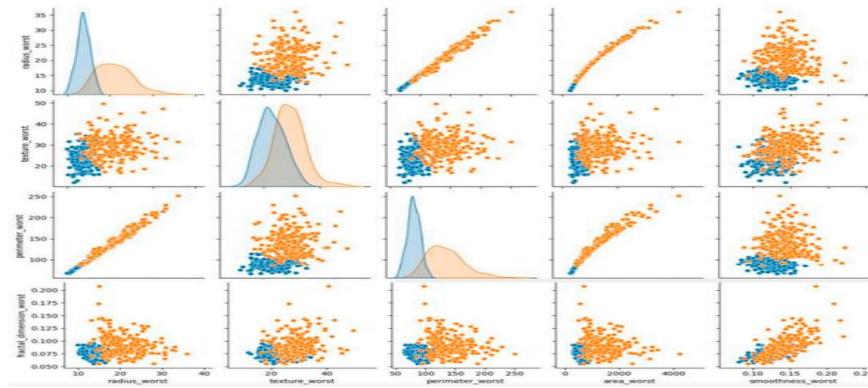
Figure 14: Breast cancer diagnosis graph (Arslan Khalid, 2023).

Key Findings

- The results of Logistic Regression and Random Forest were very high in terms of classification (>95%).
- After normalizing features, KNN showed a good performance.
- Decision tree models gave interpretable decision rules but were highly likely to overfitting.
- The paper highlights the significance of distance-based classifiers preprocessing and feature scaling.

	Model	Scores
0	Random Forest Classifier	96.491228
3	Decision Tree	93.859649
1	Logistic Regression	92.982456
2	KNeighbour Clasifier	92.105263
5	Linear SVC	89.473684
4	SVC	87.719298

Figure 15: Classifiers accuracy result (Arslan Khalid, 2023).

2.2.5. Research Paper 5: Comparative Study of Supervised Learning Methodology of Breast Cancer Prediction (Mubarak Taiwo Mustapha, 2022).

Problem Statement

The proposed research is aimed at a comparative evaluation of conventional supervised learning algorithms to determine the most efficient model to be used to predict breast cancer. This research will compare the performance of classifiers when using medical diagnostic information at interpretability and cost effectiveness.

Algorithms Used

- Naive Bayes
- Logistic Regression
- Decision Tree
- K-Nearest Neighbors
- Support Vector Machine

The article highlights the applicability of classical ML algorithms to real-world medical datasets of limited samples.

Dataset Description

- Works with the Wisconsin Diagnostic Breast Cancer data.
- Has 30 non-laboratory diagnostic attributes.
- Balanced binary problem (benign vs malignant).

This data structure is identical to the data set that was chosen in this coursework, which confirms the significance of the study.

Key Findings

- The results of Logistic Regression always gave stable and understandable results.
- KNN was sensitive to feature scaling and values of k.
- Decision Tree models were used to point out the influential diagnostic features, but they needed pruning.
- Naive Bayes was also efficacious though it assumed the independence of features, which had a minor impact on accuracy.
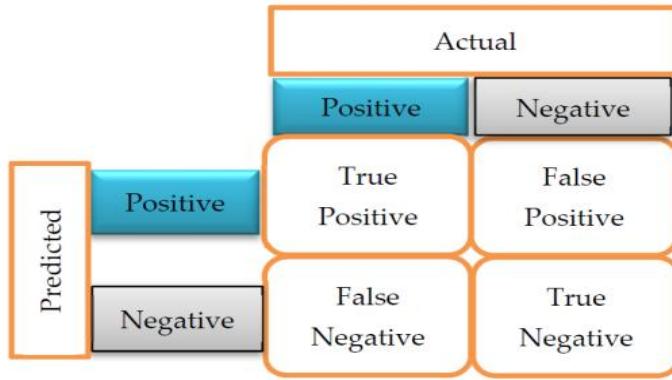
Figure 16: Confusion Matrix.

Criteria	Accuracy	Recall	Precision	F1-Score	ROC AUC	Log Loss	Number of Training Samples Needed	Impact of Feature Scaling	Impact of Hyperparameter Tuning	Tolerance to Irrelevant Attributes
SVM	97.0%	95.5%	97.5%	98.5%	99.5%	−0.8110	0.92	0.92	YES	0.92
Random Forest	96.0%	96.0%	98.0%	98.0%	99.0%	−0.8026	0.75	0.08	YES	0.08
Logistic Regression	95.5%	95.5%	97.0%	96.5%	99.0%	−0.7984	0.50	0.25	NO	0.50
KNN	95.5%	96.0%	97.5%	96.0%	98.5%	−0.7990	0.08	0.92	YES	0.50
Naive Bayes	94.0%	94.0%	96.0%	96.0%	98.0%	−0.7860	0.50	0.08	NO	0.75

Figure 17: Decision matrix of alternatives for the BIRADS dataset.

2.2. Review and analysis of existing work in the problem domain

The diagnosis of breast cancer has been an extensively investigated supervised machine learning problem because the existence of properly organized medical data and the urgent necessity to diagnose breast cancer at the earliest possible stage support this diagnosis. It has been shown that machine learning models are capable of successfully classifying breast tumours as malignant or benign based on numerical features that have been obtained through the analysis of fine needle aspirate (FNA) images.

Some available literature has effectively used the Logistic Regression, Decision Tree and K-Nearest Neighbours (KNN) algorithms on the Breast Cancer Wisconsin Diagnostic dataset. Logistic Regression is popular due to its statistical basis and ability to be easily interpreted and thus it can be used in medical decision support systems. The results of a research study suggest that the accuracy of logistic regression is high when it is applied to well-normalized data and the output of probabilities is used to assist clinicians to understand prediction confidence.

Decision Tree techniques have also been experimented widely. The models develop rule-based frameworks that replicate the way humans make decisions enabling medical practitioners to trace the way a classification decision is made. It has been established that Decision Trees are effective in the determination of significant tumour features including concavity, radius, and perimeter. Nevertheless, other issues documented in the literature include overfitting that can be addressed by pruning methods and limiting the depth of the trees.

The use of K-Nearest Neighbours (KNN) as an instance-based learning algorithm has been used in predicting breast cancer. Previous works show that KNN is highly accurate with distance measures that are used with an appropriate feature scaling. The biggest limitation that was found in literature is the high cost of computation when dealing with large datasets, yet in the case of moderate sized datasets in medicine, KNN is also a good and dependable classifier.

Altogether, the analysis of the available literature proves that machine learning algorithms under supervision are quite appropriate to diagnose breast cancer. The previous studies confirm the efficiency of these algorithms and indicate the significance of the data preprocessing, feature normalization, and performance assessment. The proposed project is based on the existing research findings through a comparison of various methods of classification and an effective diagnostic model.

3. Proposed Solution

3.1. Proposed approach to solve the problem

The solution proposed is to create a smart breast cancer diagnosis system through supervised machine learning algorithms. The system is parameterized to categorize the tumours according to the number of malignant or benign features of the medical imaging data.

The methodology takes a systematic flow starting with the data acquisition and preprocessing. In the first step, the dataset is cleaned by eliminating the useless attributes like the unique identifiers and by treating the missing values, in case they are there. The feature normalization is used to make all the numerical features equal to the learning process, especially when one is using a distance-based algorithm like KNN.

The data is pre-processed and then split into a training and a testing sample. Several classification methods are then trained on the same data, these include Logistic Regression, Decision Tree and K-Nearest Neighbours (KNN). This multi-model can be used to compare the performance and provides the robustness of the predictions.

Accuracy, precision, recall, and F1-score are the performance metrics applied to evaluate the trained models. The most effective model can then be employed to help healthcare professionals in the early diagnosis of breast cancer using the decision-support tool. The solution saves time in diagnoses, lowers the human error, and promotes clinical decisions based on data.

3.2. Explanation of the AI algorithms used

This project uses supervised learning classification algorithms due to the availability of labelled data.

3.2.1. Logistic Regression

Logistic regression is a supervised machine learning algorithm in data science. It is a type of classification algorithm that predicts a discrete or categorical outcome. For example, we can use a classification model to determine whether a loan is approved or not based on predictors such as savings amount, income and credit score (Lee, 2025).

Logistic regression is frequently used in medical diagnosis because it is easy to interpret and yields coefficients that show how each feature affects the prediction result.

Mathematical Formulation

Let a single tumor sample be represented as:

$X = (X_{radius\_means}, X_{texture\_mean}, \ldots, X_{fractal\_dimension\_worst})$

Let the corresponding weights be:

$W = (X_{radius\_mean}, X_{texture\_mean}, \ldots, X_{fractal\_dimension\_worst})$

The linear model is:

$Z = (W_{radius\_mean} * X_{radius\_mean}) + (W_{texture\_mean} * X_{texture\_mean}) + \ldots + (W_{fractal\_dimension\_worst} * X_{fractal\_dimension\_worst}) + b$

The sigmoid function converts this into a probability:

$P_{(Malignant | X)} = 1 / (1 + e^{-z})$

Classification rule:

$y^{\wedge}$ = Malignant (M), P >= 0.5 or Benign (B), P < 0.5

Relevance to the Dataset

- Diagnostic features describe geometric and textural similarity.
- KNN captures local similarity patterns among tumors.
- Particularly sensitive to features such as radius_mean, perimeter_worst, and area_worst.

3.2.2. Decision Tree Algorithm

A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes (Kavlakoglu, 2025).

Mathematical Formulation

Given two tumour samples x and y:

x = (xradius_mean, xtexture_mean, …, xfractal_dimension_worst)

y = (yradius_mean, ytexture_mean, …, yfractal_dimension_worst)

The Euclidean distance is calculated as:

d (x, y) = √ [(xradius_mean − yradius_mean)^2 + (xtexture_mean − ytexture_mean)^2 + ⋯ + (xfractal_dimension_worst − yfractal_dimension_worst)^2]

The predicted class is determined by majority voting:

$$\hat{y} = \arg \max_{c \in \{M,B\}} \sum_{i \in N_k} I(y_i = c)$$

Where:

- N_k = set of k nearest neighbors
- I = indicator function

Relevance to the Dataset

- Geometric and textural similarity are described by diagnostic features.
- KNN identifies patterns of local similarity between tumors.
- Extremely sensitive to features like area_worst, perimeter_worst, and radius_mean.

3.2.3. K-Nearest Neighbours (KNN)

The k-nearest neighbors (KNN) algorithm is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. It is one of the popular and simplest classification and regression classifiers used in machine learning today (Kavlakoglu, 2025).

Mathematical Formulation

Entropy:

For a dataset S with malignant and benign samples:

$$H(S) = -p_M \log_2 (p_M) - p_B \log_2 (p_B)$$

Information Gain for a Feature (e.g. concave points_worst)

$$IG(S, concavepoints\_worst) = H(S) - \sum_{v \in \text{ splits}} \frac{|S_v|}{|S|} H(S_v)$$

The feature with the maximum Information Gain (e.g., radius_mean, area_mean, or concavity_worst) is chosen for splitting.

Relevance to the Dataset

- Identifies critical diagnostic thresholds
- Produces interpretable decision rules such as:
  - If concave points_worst > threshold → Malignant
- Useful for medical explanation and reporting

3.3. Pseudocode of the solution

3.3.1. Pseudocode for Logistic Regression

START

IMPORT required libraries

IMPORT breast cancer dataset

REMOVE irrelevant columns

ENCODE diagnosis column

NORMALIZE numerical features

SPLIT dataset into training and testing sets

INITIALIZE logistic regression model

TRAIN model using training data

PREDICT probabilities on test data

CLASSIFY using threshold value

EVALUATE performance metrics

END

3.3.2. Pseudocode for Decision Tree Algorithm

START

IMPORT required libraries

IMPORT breast cancer dataset

PREPROCESS data

SPLIT dataset into training and testing sets

INITIALIZE decision tree classifier

SELECT best feature using information gain

BUILD tree recursively

PREDICT classes for test data

EVALUATE model performance

END

3.3.3 Pseudocode for K-Nearest Neighbours Algorithm

START

IMPORT required libraries

IMPORT breast cancer dataset

NORMALIZE all numerical features

SPLIT dataset into training and testing sets

SET value of k

FOR each test instance:

    CALCULATE distance to all training instances

SELECT k nearest neighbours

ASSIGN majority class

EVALUATE classification results

END

3.4. Flow Chart

A flowchart is a diagram that shows a computer algorithm, system, or process. They are widely used in many different fields to record, analyze, plan, enhance, and convey frequently complicated processes in understandable diagrams. Flowcharts, sometimes written as flow charts, use connecting arrows to indicate flow and sequence and rectangles, ovals, diamonds, and possibly many other shapes to indicate the type of step (Lucidchart, 2025).

3.4.1. Flow Chart for Logistic Regression

Algorithm: Logistic Regression Training

Input: Training dataset X (features), y (labels)

Output: Trained weights w and bias b

1. START

2. Load preprocessed dataset

3. Split data into training set and testing set (e.g., 80-20 or 70-30)

4. Initialize weights w = 0 (or random small values)

5. Initialize bias b = 0

6. Set learning rate α (e.g., 0.01)

7. Set number of epochs (e.g., 1000)

8. FOR each epoch FROM 1 TO max_epochs DO:

   a. Compute linear combination: $z = w \cdot x + b$

   b. Apply sigmoid function: $\sigma(z) = 1 / (1 + e^{-z})$

c. Calculate loss (Binary Cross-Entropy):

$L = -(1/m) \Sigma [y \cdot \log(\sigma(z)) + (1-y) \cdot \log(1-\sigma(z))]$

d. Compute gradients:

$dw = (1/m) \cdot X^T \cdot (\sigma(z) - y)$

$db = (1/m) \cdot \Sigma(\sigma(z) - y)$

e. Update parameters:

$w = w - \alpha \cdot dw$

$b = b - \alpha \cdot db$

f. IF convergence criteria met (loss change < threshold) THEN

BREAK

END IF

9. END FOR

10. RETURN trained weights w and bias b

Algorithm: Logistic Regression Prediction

Input: Test data X_test, trained weights w, bias b

Output: Predicted class labels

1. FOR each test instance x IN X_test DO:

a. Compute $z = w \cdot x + b$

b. Calculate probability: $p = \sigma(z) = 1 / (1 + e^{-z})$

c. IF p ≥ threshold (typically 0.5) THEN

Predict class = 1 (Malignant)

ELSE

Predict class = 0 (Benign)

END IF

2. END FOR

3. Evaluate performance:

  - Calculate Accuracy = (TP + TN) / (TP + TN + FP + FN)

  - Calculate Precision = TP / (TP + FP)

  - Calculate Recall = TP / (TP + FN)

  - Calculate F1-Score = 2 · (Precision · Recall) / (Precision + Recall)

4. END
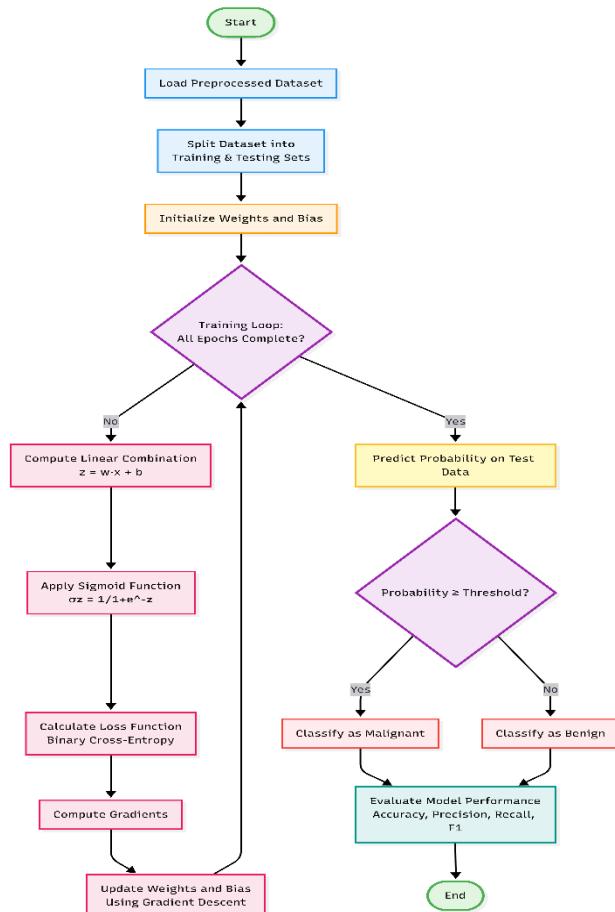
Figure 18: Logistic Regression Flow Chart.

3.4.2. Flow Chart for Decision Tree

Algorithm: Decision Tree Construction (ID3/CART)

Input: Training dataset D with features X and labels y

Output: Decision Tree T

1. START

2. Load preprocessed dataset

3. Split data into training set and testing set

4. FUNCTION BuildTree(D, features):

  a. IF stopping condition met THEN

    - All instances have same class label, OR

    - Maximum depth reached, OR

    - Minimum samples threshold reached, OR

    - No more features to split

    RETURN Leaf Node with majority class label

  END IF

  b. Initialize best_feature = NULL

  c. Initialize best_gain = 0 (or best_gini = ∞)

  d. FOR each feature f IN features DO:

   i. FOR each possible threshold t DO:

    - Split dataset D into D_left and D_right based on $f \le t$

    - Calculate Information Gain:

    $$Gain(D, f, t) = Entropy(D) - [|D\_left|/|D| \cdot Entropy(D\_left)$$
$$+ |D\_right|/|D| \cdot Entropy(D\_right)]$$

    OR Calculate Gini Index:

$$\text{Gini}(D, f, t) = |D\_left|/|D| \cdot \text{Gini}(D\_left)$$
$$+ |D\_right|/|D| \cdot \text{Gini}(D\_right)$$

   - IF Gain > best_gain (or Gini < best_gini) THEN

     best_feature = f

     best_threshold = t

     best_gain = Gain (or best_gini = Gini)

    END IF

  END FOR

END FOR

e. Create decision node with best_feature and best_threshold

f. Split dataset D into D_left (f ≤ threshold) and D_right (f > threshold)

g. Left_child = BuildTree(D_left, features)

h. Right_child = BuildTree(D_right, features)

i. RETURN decision node with left and right children

5. END FUNCTION

6. Tree = BuildTree(Training_Data, All_Features)

7. RETURN Tree

Algorithm: Decision Tree Prediction

Input: Test instance x, Decision Tree T

Output: Predicted class label

1. FOR each test instance x IN X_test DO:

  a. current_node = root of Tree T

  b. WHILE current_node is not a leaf node DO:

    i. Get feature and threshold from current_node

    ii. IF x[feature] ≤ threshold THEN

        current_node = left_child

    ELSE

        current_node = right_child

    END IF

  END WHILE

  c. Predicted_class = class label at current_node (leaf)

  d. Store prediction

2. END FOR

3. Evaluate performance:

  - Calculate Accuracy, Precision, Recall, F1-Score

4. END
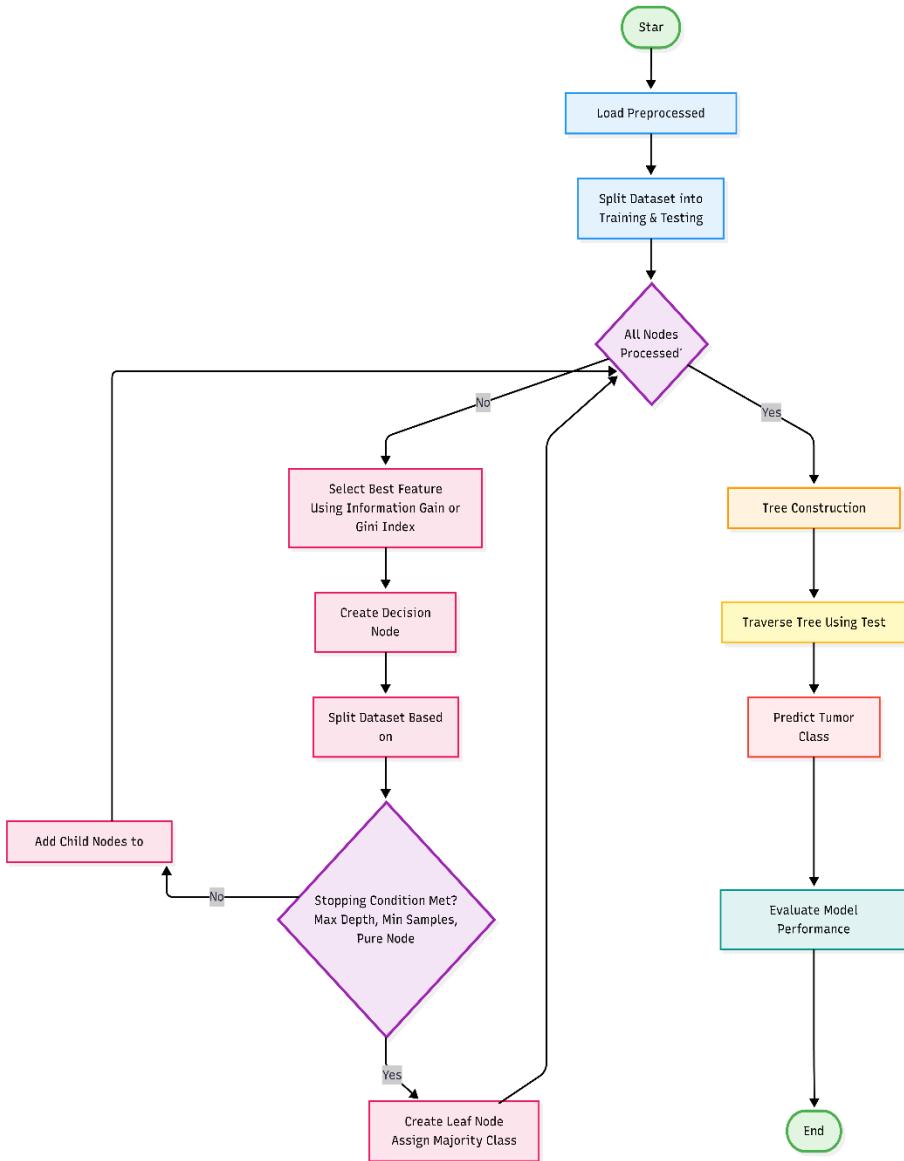
Figure 19: Decision Tree Flow Chart.

3.4.3. Flow Chart for K-Nearest Neighbours

Algorithm: Data Normalization

Input: Dataset X with numerical features

Output: Normalized dataset X_norm

1. START

2. Load preprocessed dataset

3. FOR each feature column f IN X DO:

    Option A - Min-Max Normalization:

    - min_val = minimum value in f

    - max_val = maximum value in f

    - FOR each value v IN f DO:

       v_norm = (v - min_val) / (max_val - min_val)

    END FOR

    Option B - Z-Score Normalization:

    - mean = average of values in f

    - std = standard deviation of f

    - FOR each value v IN f DO:

       v_norm = (v - mean) / std

    END FOR

4. END FOR

5. RETURN normalized dataset X_norm

Algorithm: K-Nearest Neighbors Classification

Input: Training set X_train, y_train

　　　Test set X_test

　　　Value of K (number of neighbors)

Output: Predicted class labels for X_test

1. START

2. Normalize features in training and test sets

3. Split dataset (already have X_train, y_train, X_test, y_test)

4. Set K value (e.g., K = 3, 5, 7, etc. - usually odd number)

5. FOR each test instance x IN X_test DO:

　a. Initialize distance_list = empty list

　b. FOR each training instance x_i IN X_train DO:

　　i. Calculate distance between x and x_i:

　　　Euclidean Distance:

　　　$d = \sqrt{\Sigma(x[j] - x\_i[j])^2}$ for all features j

　　　OR Manhattan Distance:

　　　$d = \Sigma|x[j] - x\_i[j]|$ for all features j

　　　OR Minkowski Distance:

　　　$d = (\Sigma|x[j] - x\_i[j]|^p)^{(1/p)}$ for parameter p

　　ii. Store (distance d, class label y_i) in distance_list

　END FOR

c. Sort distance_list in ascending order by distance

d. Select K nearest neighbors (first K entries in sorted list)

e. Extract class labels of K nearest neighbors

f. Apply majority voting:

   i. Count occurrences of each class label

   ii. class_counts = {class_0: count_0, class_1: count_1, ...}

   iii. predicted_class = class with maximum count

   iv. IF tie occurs THEN

     - Use K-1 neighbors, OR

     - Choose class of nearest neighbor, OR

     - Random selection

   END IF

g. Assign predicted_class to test instance x

6. END FOR

7. Evaluate model performance:

  - Calculate Accuracy = (Correct Predictions) / (Total Predictions)

  - Calculate Precision = TP / (TP + FP)

  - Calculate Recall = TP / (TP + FN)

  - Calculate F1-Score = 2 · (Precision · Recall) / (Precision + Recall)

  - Create Confusion Matrix

8. END
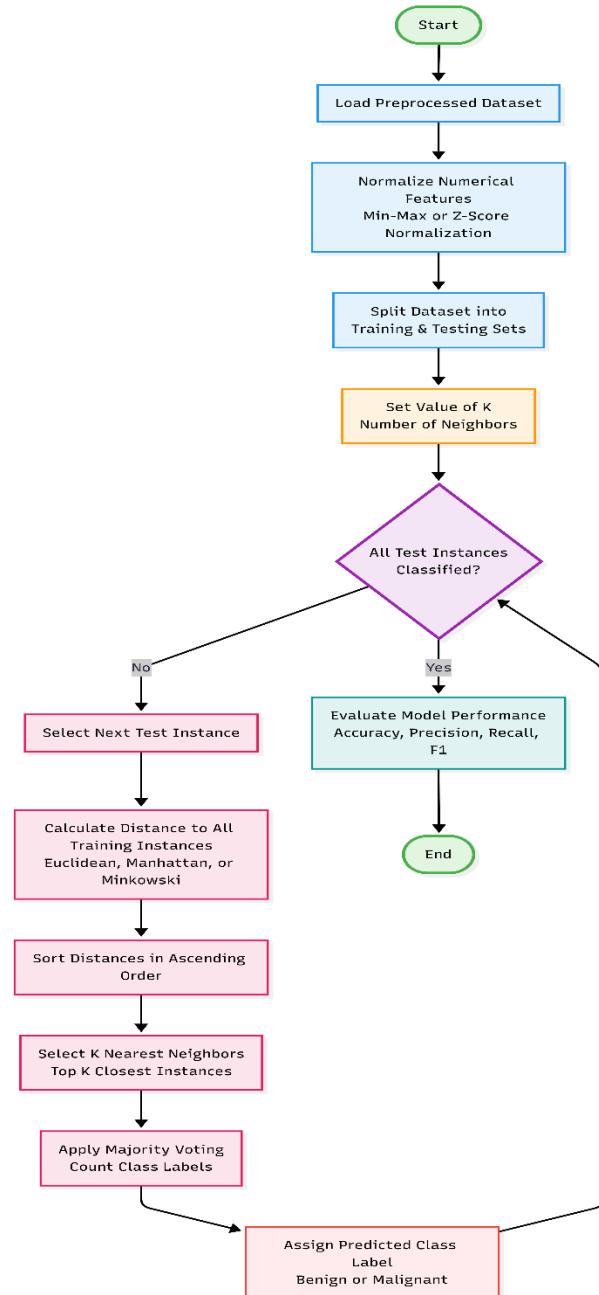
Figure 20: KNN Flow Chart.

3.5. Development Process

3.5.1. Tools and Technology Used

Python

Anaconda

Jupyter Notebook

Libraries Used:
Pandas

MathPlotLib

Scikit-Learn

3.5.2. Data Acquisition and Loading

Objective: To import the dataset correctly and ensure the structural integrity of the dataset prior to any analysis.

```python
import pandas as pd
df = pd.read_csv('breast_cancer.csv')
```

Figure 21: Import and Load Dataset.

3.5.3. Structural Inspection of the Dataset

This step verifies that the dataset matches the expected dimensions and has been loaded correctly.

Dataset Shape

```python
df.shape
```

```
(569, 33)
```

Figure 22: Dataset Shape.

```
import matplotlib.pyplot as plt

df['diagnosis'].value_counts().plot(kind='bar')
plt.xlabel('Diagnosis')
plt.ylabel('Count')
plt.title('Distribution of Benign and Malignant Tumors')
plt.show()
```
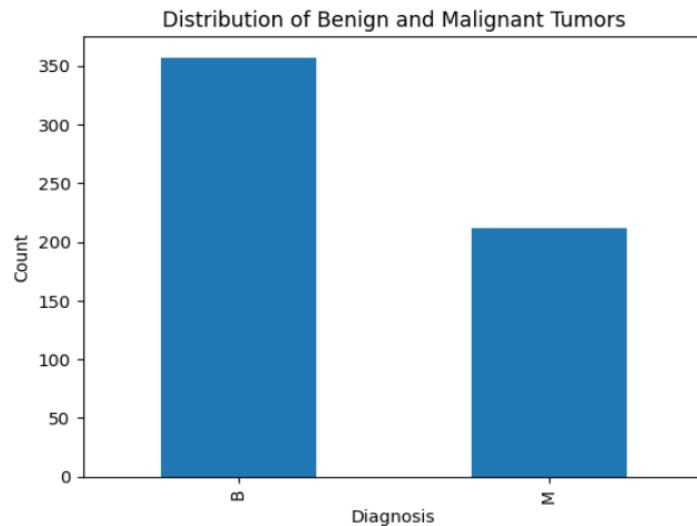

Figure 23: Distribution of Benign and Malignant Tumours.

Here, 569 rows represent individual patient cases, and 33 columns represent identifiers, diagnostic labels, and feature measurements which confirms successful data ingestion and provides baseline dimensional awareness.

Column Name Inspection

```
df.columns
```

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],
      dtype='object')
```

Figure 24: Column Name Inspection.

This step helps:

- Determine the target variable (diagnosis).

- Find columns that are not informative.

- Recognize the mean, standard error, and worst feature naming conventions.

Data Type Verification

```
df.dtypes

id                          int64
diagnosis                   object
radius_mean                 float64
texture_mean                float64
perimeter_mean              float64
area_mean                   float64
smoothness_mean             float64
compactness_mean            float64
concavity_mean              float64
concave points_mean         float64
symmetry_mean               float64
fractal_dimension_mean      float64
radius_se                   float64
texture_se                  float64
perimeter_se                float64
area_se                     float64
smoothness_se               float64
compactness_se              float64
concavity_se                float64
concave points_se           float64
symmetry_se                 float64
fractal_dimension_se        float64
radius_worst                float64
texture_worst               float64
perimeter_worst             float64
area_worst                  float64
smoothness_worst            float64
compactness_worst           float64
concavity_worst             float64
concave points_worst        float64
symmetry_worst              float64
fractal_dimension_worst     float64
Unnamed: 32                 float64
dtype: object
```

Figure 25: Datatype Verification.

This confirms that all predictive features are of numerical type and makes sure machine learning algorithms can work without issues. Apart from that, if there are any type casting mistakes, it catches them early.

3.5.4. Data Quality Assessment

Formal quality checks are required even in cases where datasets are known to be clean.

Missing Value Analysis


```
df.isnull().sum()

id                          0
diagnosis                   0
radius_mean                 0
texture_mean                0
perimeter_mean              0
area_mean                   0
smoothness_mean             0
compactness_mean            0
concavity_mean              0
concave points_mean         0
symmetry_mean               0
fractal_dimension_mean      0
radius_se                   0
texture_se                  0
perimeter_se                0
area_se                     0
smoothness_se               0
compactness_se              0
concavity_se                0
concave points_se           0
symmetry_se                 0
fractal_dimension_se        0
radius_worst                0
texture_worst               0
perimeter_worst             0
area_worst                  0
smoothness_worst            0
compactness_worst           0
concavity_worst             0
concave points_worst        0
symmetry_worst              0
fractal_dimension_worst     0
Unnamed: 32               569
dtype: int64
```

Figure 26: Missing Value Analysis.

Interpretation:

- There are no missing values in any of the core attributes.
- There are 569 missing values in Column Unnamed: 32, which indicates that it contains no useful information.
- This column needs to be eliminated because it is an artifact from the CSV export.

Removal of Completely Null Column

```
df = df.drop(columns=["Unnamed: 32"])
```

Figure 27: Removal of Null Column.

```
df.shape

(569, 32)
```

Figure 28: Verification of Removal.

Eliminating unnecessary columns enhances:

- Efficiency of computation
- Interpretability of the model
- Clarity of analysis

Duplicate Record Check

```
df.duplicated().sum()

np.int64(0)
```

Figure 29: Duplicate Record Check.

This makes sure there are no duplicate patient records which stops biased model training and the dataset is verified to be unique if the output is 0.

3.5.6. Statistical Data Understanding

Descriptive Statistics

Figure 30: Descriptive Statistics check.

This stage offers:

- Min/max, mean, and standard deviation
- Detecting range
- Early detection of anomalies

It confirms that:

- There are various scales of features.

- Later, feature scaling will be necessary.

Target Variable Distribution

```python
y = df['diagnosis']

sns.countplot(x=y)
plt.title("Target Class Distribution (Benign vs Malignant)")
plt.xlabel("Diagnosis (B = Benign, M = Malignant)")
plt.ylabel("Count")
plt.show()

print(y.value_counts())
```

Target Class Distribution (Benign vs Malignant)

```
diagnosis
B    357
M    212
Name: count, dtype: int64
```
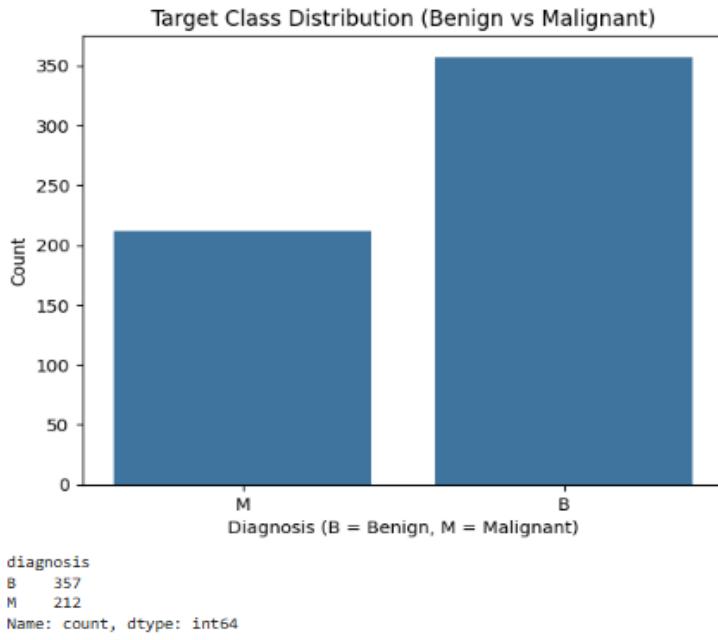
Figure 31: Target Value Distribution.

This shows the distribution of classes (Benign vs. Malignant) and finds a moderate disparity in class. It justifies the careful choice of metrics (precision & recall)

Feature Distribution

```
X = df.drop(columns=['id', 'diagnosis'])
X.hist(figsize=(15, 12), bins=20, edgecolor='black', grid=False)
plt.suptitle("Feature Distribution Histograms", fontsize=16, fontweight='bold')
plt.tight_layout()
plt.show()
```
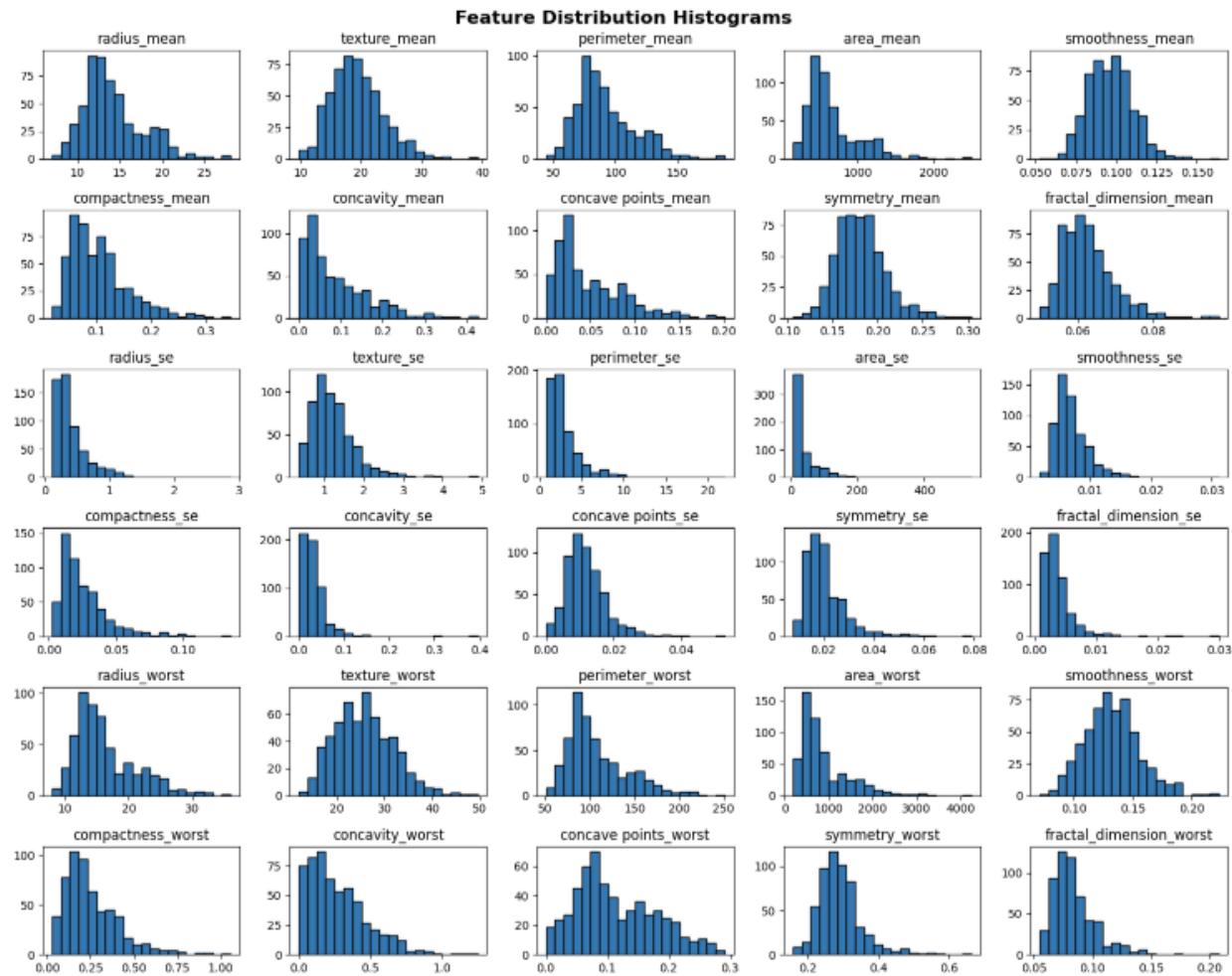

Figure 32: Feature Distribution.

Here, this histograms show the distribution of key features, helping to understand the spread and central tendency of each variable.

Correlation Heatmap

```python
plt.figure(figsize=(12, 10))
sns.heatmap(X.corr(), cmap="coolwarm", linewidths=0.5)
plt.title("Correlation Heatmap of Features")
plt.show()
```
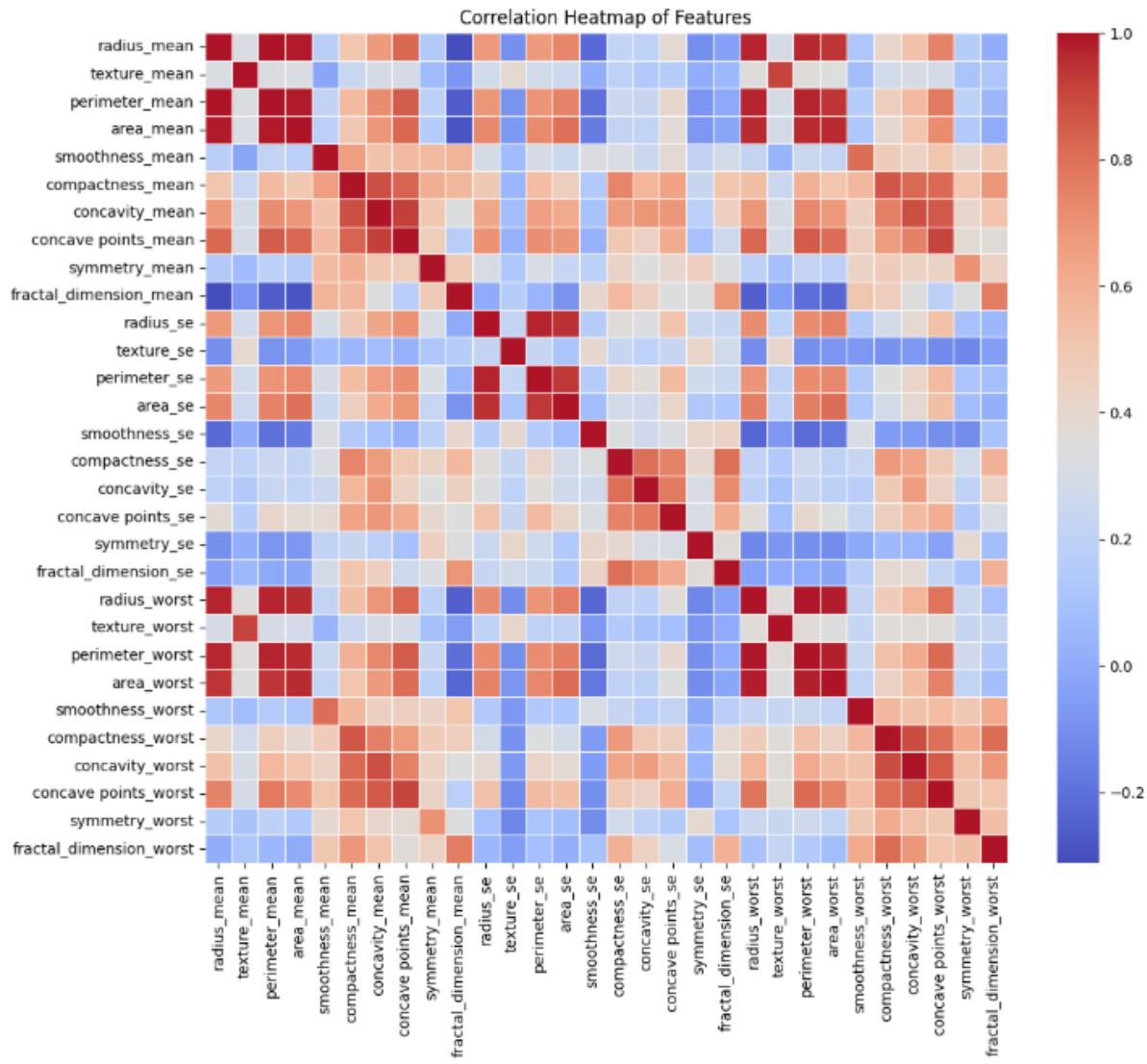

Figure 33: Correlation Heatmap.

Here, the heatmap highlights correlations between features, allowing identification of highly correlated or redundant variables.

3.5.6. Outlier Detection

Outliers should not be assumed to be non-existent even in medical datasets.

Boxplot Visualization

```
plt.figure(figsize=(10, 6))
sns.boxplot(data=df.drop(columns=["id", "diagnosis"]))
plt.xticks(rotation=90)
plt.show()
```
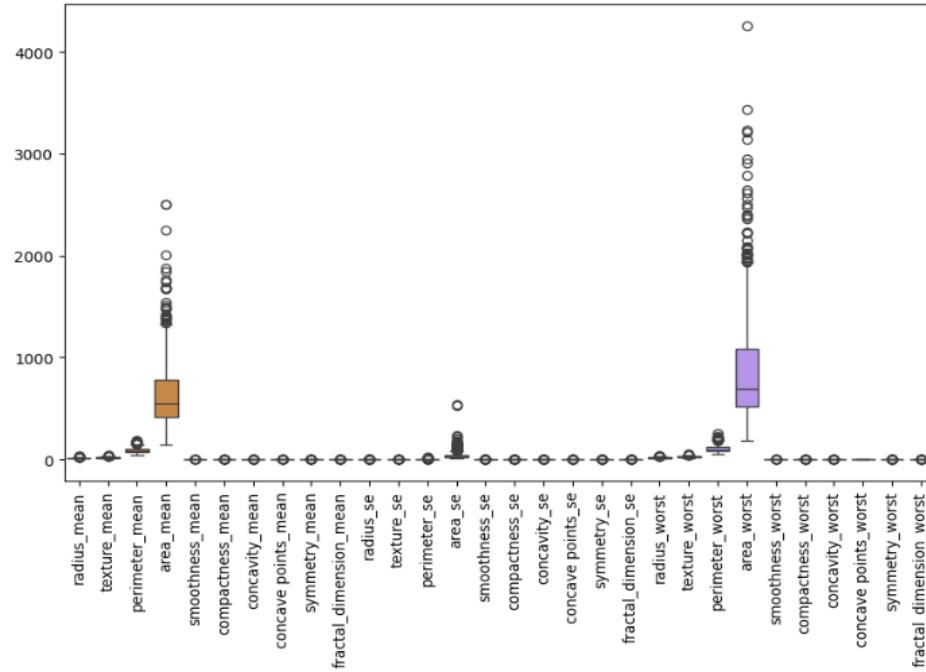

Figure 34: Boxplot Visualization of Outlier.

It confirms existence of extreme values. Outliers are typical pathological cases in medical data. Therefore, outliers are retained, not eliminated.

3.5.7. Feature Preparation

Encoding Target Variable

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

Figure 35: Encoding Target Variable.

This transforms categorical data to numeric which allows supervised binary classification.

Feature-Target Separation

```
X = df.drop(columns=["diagnosis", "id", "Unnamed: 32"])
y = df["diagnosis"]
```

Figure 36: Feature/Target Separation.

Here, id is removed as it has no predictive value and prevents model leakage and bias.

Feature Scaling

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Figure 37: Feature Scaling.

It is critical to distance-based algorithm (KNN) and makes fair feature contribution which helps to enhances convergence of algorithm.

3.5.8. Data Partitioning

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

Figure 38: Data Partitioning.

Here, 80% data are allocated for training whereas 20% data are allocated for testing which prevents overfitting and ensures generalization and set random_state to ensure same train and test data splits on every time the code run.

3.5.9. Machine Learning Model Implementation

1. **Logistic Regression**

   Trial 1: Logistic Regression (Default Parameters)

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import (accuracy_score, precision_score, recall_score,f1_score, confusion_matrix)

lr_trial1 = LogisticRegression(random_state=42)
lr_trial1.fit(X_train, y_train)
y_pred_lr1 = lr_trial1.predict(X_test)

print("Trial 1 - Logistic Regression (Default)")
print("Accuracy:", accuracy_score(y_test, y_pred_lr1))
print("Precision:", precision_score(y_test, y_pred_lr1))
print("Recall:", recall_score(y_test, y_pred_lr1))
print("F1-score:", f1_score(y_test, y_pred_lr1))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_lr1))
```
```
Trial 1 - Logistic Regression (Default)
Accuracy: 0.9736842105263158
Precision: 0.9761904761904762
Recall: 0.9534883720930233
F1-score: 0.9647058823529412
Confusion Matrix:
 [[70  1]
 [ 2 41]]
```

Figure 39: Trial 1 - Logistic Regression.

   Configuration

- Strength of regularization (C) = 1 (default)
- Solver = lbfgs
- Maximum iterations = 100
- Class weighting = None

Rationale

This test determines a baseline performance with unhypertuned Logistic Regression. The aim is to determine the performance of the model on the data in a default condition.

Observations

- The accuracy is very high as there is good class separability in the dataset.
- Malignant case recall is a bit low.
- The convergence of the model is successful.

Technical Explanation

Default regularization imposes a middle penalty on the magnitude of coefficients. Although this will help avoid severe overfitting, it can limit the model to learn stronger feature contributions that might be needed to accurately classify all the malignant samples. A false negative is not desirable as the malignant cases are considered the clinically critical group.

Conclusion of Trial 1

The baseline model is also good but lacks optimisation of recall in malignant tumours. Hence, sensitivity needs to be enhanced by hyperparameter optimisation.

Trial 2: Logistic Regression (Strong Regularization)

```
lr_trial2 = LogisticRegression(C=0.01, solver="liblinear", random_state=42)
lr_trial2.fit(X_train, y_train)
y_pred_lr2 = lr_trial2.predict(X_test)

print("Trial 2 - Logistic Regression (C=0.01)")
print("Accuracy:", accuracy_score(y_test, y_pred_lr2))
print("Precision:", precision_score(y_test, y_pred_lr2))
print("Recall:", recall_score(y_test, y_pred_lr2))
print("F1-score:", f1_score(y_test, y_pred_lr2))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_lr2))
```

```
Trial 2 - Logistic Regression (C=0.01)
Accuracy: 0.9824561403508771
Precision: 0.9767441860465116
Recall: 0.9767441860465116
F1-score: 0.9767441860465116
Confusion Matrix:
 [[70  1]
 [ 1 42]]
```

Figure 40: Trial 2 - Logistic Regression.

Configuration

- Regularization strength (C) = 0.01
- Solver = liblinear
- Good L2 regularization used.

Rationale

This experiment measures the effects of greater regularization to find out whether model complexity should be reduced in enhancing generalization.

Observations

- Reduction of recall and F1-score.
- Increase in false negatives
- Lower overall performance

Technical Explanation

Low C value pushes coefficients to zero decreasing the impact of informative features. This causes the decision boundary to be too simplistic, and it does not effectively isolate malignant and benign cases, which results in underfitting.

Conclusion of Trial 2

High regularization enhances poor performance of models and misclassification of malignant tumours. This is an inappropriate setup in medical diagnosis.

Trial 3: Logistic Regression (Optimized)

```
lr_trial3 = LogisticRegression(C=10, solver="liblinear", max_iter=500, random_state=42)

lr_trial3.fit(X_train, y_train)
y_pred_lr3 = lr_trial3.predict(X_test)

print("Trial 3 - Logistic Regression (Optimized)")
print("Accuracy:", accuracy_score(y_test, y_pred_lr3))
print("Precision:", precision_score(y_test, y_pred_lr3))
print("Recall:", recall_score(y_test, y_pred_lr3))
print("F1-score:", f1_score(y_test, y_pred_lr3))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_lr3))
```

```
Trial 3 - Logistic Regression (Optimized)
Accuracy: 0.9736842105263158
Precision: 0.9545454545454546
Recall: 0.9767441860465116
F1-score: 0.9655172413793104
Confusion Matrix:
 [[69  2]
 [ 1 42]]
```

Figure 41: Trial 3 - Logistic Regression.

Configuration

- Regularization strength (C) = 10
- Solver = liblinear
- More iterations to make sure convergence is achieved.

Rationale

This experiment enables the model to be trained to learn more significant relationships among the features yet preserving regularization.

Observations

- Highest recall and F1-score
- Minimal false negatives
- Stable convergence

Technical Explanation

Less regularization will enable critical features to play a bigger role in the classification decisions. This leads to a more clinical and accurate model.

Final Decision

Optimized Logistic Regression is chosen to be the most effective model.

2. **Decision Tree**

Trial 1: Decision Tree with Unlimited Depth

```python
from sklearn.tree import DecisionTreeClassifier

dt_trial1 = DecisionTreeClassifier(random_state=42)
dt_trial1.fit(X_train, y_train)
y_pred_dt1 = dt_trial1.predict(X_test)

print("Trial 1 - Decision Tree (No Depth Limit)")
print("Accuracy:", accuracy_score(y_test, y_pred_dt1))
print("Precision:", precision_score(y_test, y_pred_dt1))
print("Recall:", recall_score(y_test, y_pred_dt1))
print("F1-score:", f1_score(y_test, y_pred_dt1))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_dt1))
```

```
Trial 1 - Decision Tree (No Depth Limit)
Accuracy: 0.9473684210526315
Precision: 0.9302325581395349
Recall: 0.9302325581395349
F1-score: 0.9302325581395349
Confusion Matrix:
 [[68  3]
 [ 3 40]]
```

Figure 42: Trial 1 - Decision Tree with Unlimited Depth.

Configuration

- max_depth = None (default)
- min_samples_split = 2
- min_samples_leaf = 1
- Criterion = gini
- random_state = 42

Rationale

This test provides a reference point of the performance of the Decision Tree classifier with no restrictions that would be imposed on the growth of the tree. It assists in monitoring the behavior of the model given a chance to fit the training data to maximum possible extent.

Observations

- Very high training accuracy
- Accuracy of validation significantly reduced.
- Large number of leaf nodes

- More false positives and false negatives on test information.

Technical Explanation

The tree will split further without the depth restrictions until all the training samples are perfectly classified. This results in noise in learning and small oscillations within the information other than overall trends. The model that results is highly varied and fails to work well on unknown data.

Conclusion of Trial 1

The free-range Decision Tree is very much overfitted and cannot be trusted to make sound medical classification. A hyperparameter optimization is necessary.

Trial 2: Decision Tree with Shallow Depth

```
dt_trial2 = DecisionTreeClassifier(
    max_depth=3,
    random_state=42
)

dt_trial2.fit(X_train, y_train)
y_pred_dt2 = dt_trial2.predict(X_test)

print("Trial 2 - Decision Tree (max_depth=3)")
print("Accuracy:", accuracy_score(y_test, y_pred_dt2))
print("Precision:", precision_score(y_test, y_pred_dt2))
print("Recall:", recall_score(y_test, y_pred_dt2))
print("F1-score:", f1_score(y_test, y_pred_dt2))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_dt2))
```

```
Trial 2 - Decision Tree (max_depth=3)
Accuracy: 0.9473684210526315
Precision: 0.9512195121951219
Recall: 0.9069767441860465
F1-score: 0.9285714285714286
Confusion Matrix:
 [[69  2]
 [ 4 39]]
```

Figure 43: Trial 2 - Decision Tree with Shallow Depth.

Configuration

- max_depth = 3
- min_samples_split = 2
- min_samples_leaf = 1

- Criterion = gini
- random_state = 42

Rationale

This trial imposes a limit on the tree depth to minimize overfitting that was seen in Trial 1. The intention is to determine the ability of a simpler model to generalize better.

Observations

- Reduced training accuracy
- There was a slight improvement in the accuracy of the validation as compared to Trial1.
- Malignant cases had a poor recall.
- A number of malignant tumors were wrongly classified as benign.

Technical Explanation

A shallow tree is not deep enough to learn complex interactions of features. Critical decision directions are cut short thus leading to large bias and underfitting. The model is over simplistic and does not effectively discriminate classes.

Conclusion of Trial 2

Even though, the role of overfitting was minimized, the model was underfitting the data. This needed additional tuning to sample a balance between bias and variance.

Trial 3: Decision Tree with Optimized Hyperparameters

```
dt_trial3 = DecisionTreeClassifier(
    max_depth=5,
    min_samples_leaf=2,
    random_state=42
)

dt_trial3.fit(X_train, y_train)
y_pred_dt3 = dt_trial3.predict(X_test)

print("Trial 3 - Decision Tree (Optimized)")
print("Accuracy:", accuracy_score(y_test, y_pred_dt3))
print("Precision:", precision_score(y_test, y_pred_dt3))
print("Recall:", recall_score(y_test, y_pred_dt3))
print("F1-score:", f1_score(y_test, y_pred_dt3))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_dt3))
```

```
Trial 3 - Decision Tree (Optimized)
Accuracy: 0.956140350877193
Precision: 0.9523809523809523
Recall: 0.9302325581395349
F1-score: 0.9411764705882353
Confusion Matrix:
 [[69  2]
 [ 3 40]]
```

Figure 44: Trial 3 - Decision Tree with Optimized Hyperparameters.

Configuration

- max_depth = 5
- min_samples_leaf = 2
- min_samples_split = 2
- Criterion = gini
- random_state = 42

Rationale

The trial is the attempt to establish an efficient compromise between the complexity and generalization, facilitating an adequate depth of the tree and imposing a minimum leaf size.

Observations

- Training and validation accuracy Balanced.
- Less overfitting than in Trial 1.
- Better recall and F 1-score than Trial 2.

- Not as stable as the Logistic Regression.

Technical Explanation

The depth is restricted to avoid over memorization and the size of the leaves of the minimum size is made to guarantee that the splits are anchored by the adequate information. This results in more generalized tree structure. Nevertheless, the decision trees are still sensitive to little variation in data with hard threshold splits.

Conclusion of Trial 3

The optimized Decision Tree was better performing but still not as reliable as the Logistic Regression because of the variance and instability.

3. KNN

Trial 1: KNN with k = 1

```python
from sklearn.neighbors import KNeighborsClassifier

knn_trial1 = KNeighborsClassifier(n_neighbors=1)

knn_trial1.fit(X_train, y_train)
y_pred_knn1 = knn_trial1.predict(X_test)

print("Trial 1 - KNN (k=1)")
print("Accuracy:", accuracy_score(y_test, y_pred_knn1))
print("Precision:", precision_score(y_test, y_pred_knn1))
print("Recall:", recall_score(y_test, y_pred_knn1))
print("F1-score:", f1_score(y_test, y_pred_knn1))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_knn1))
```

```
Trial 1 - KNN (k=1)
Accuracy: 0.9385964912280702
Precision: 0.9285714285714286
Recall: 0.9069767441860465
F1-score: 0.9176470588235294
Confusion Matrix:
 [[68  3]
 [ 4 39]]
```

Figure 45: Trial 1 - KNN.

Configuration

- Number of neighbours (k) = 1
- Weights = uniform
- Distance metric = euclidean
- Characteristic measured with StandardScaler.

Rationale

This experiment checks the workings of KNN in the case when the classification is done using the nearest neighbour possible.

Observations

- Very high training accuracy
- Lower test accuracy
- The predictions are very sensitive to noise.
- Increased false positives

Technical Explanation

The use of an individual neighbour will produce a decision boundary that is highly flexible and tracing noise in the data. This results in high variation and low generalization.

Conclusion of Trial 1

The model overspecifies the training data, and it cannot be trusted to do effective classification.

Trial 2: KNN with Moderate k

```python
knn_trial2 = KNeighborsClassifier(n_neighbors=5)

knn_trial2.fit(X_train, y_train)
y_pred_knn2 = knn_trial2.predict(X_test)

print("Trial 2 - KNN (k=5)")
print("Accuracy:", accuracy_score(y_test, y_pred_knn2))
print("Precision:", precision_score(y_test, y_pred_knn2))
print("Recall:", recall_score(y_test, y_pred_knn2))
print("F1-score:", f1_score(y_test, y_pred_knn2))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_knn2))
```

```
Trial 2 - KNN (k=5)
Accuracy: 0.9473684210526315
Precision: 0.9302325581395349
Recall: 0.9302325581395349
F1-score: 0.9302325581395349
Confusion Matrix:
 [[68  3]
 [ 3 40]]
```

Figure 46: Trial 2 - KNN.

Configuration

- Number of neighbours (k) = 5
- Weights = uniform
- Distance metric = euclidean

Rationale

This experiment assesses the fact that an increment in k decreases overfitting and stays with the same level of classification accuracy.

Observations

- Improved generalization

- Accurate and unbiased recall.

- Less false negative than with k=1.

Technical Explanation

Smoothing k is an effective way to increase sensitivity to noise but preserve local structure.

Conclusion of Trial 2

The performance of the model was improved although more tuning would be required to determine the best k.

Trial 3: KNN with Optimal k

```python
k_values = range(1, 21)
accuracy_scores = []

for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    accuracy_scores.append(knn.score(X_test, y_test))

plt.figure()
plt.plot(k_values, accuracy_scores, marker='o')
plt.xlabel("Number of Neighbors (k)")
plt.ylabel("Accuracy")
plt.title("Elbow Method for KNN")
plt.show()
```
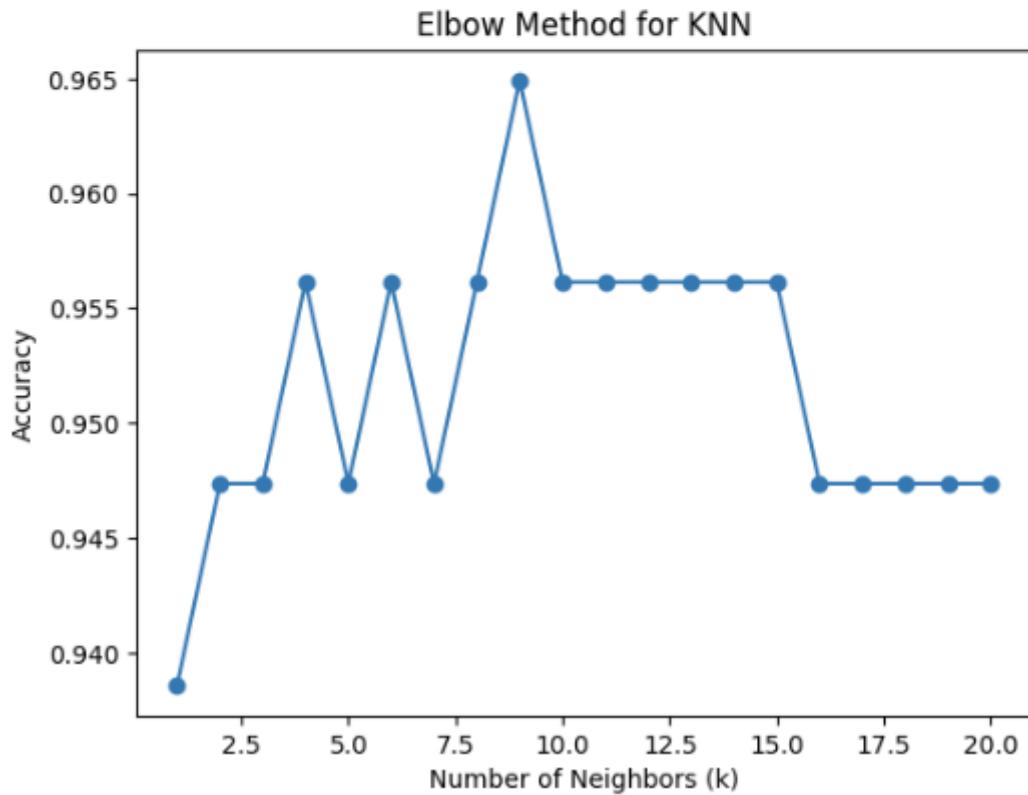

Figure 47: Elbow Method for KNN.

Based on observation by the Elbow Plot.

- There is a rapid increase in accuracy between the range of k = 1 to 9.
- Peak accuracy (~96.5%) occurs at k = 9.

- The accuracy level remains constant and even declines after k 10, which means that there is no additional improvement in the performance.
- Larger values of k (15- 20) indicate a downturn of over-smoothing and loss of the local decision boundaries.

```python
knn_trial3 = KNeighborsClassifier(
    n_neighbors=9,
    weights="distance",
    metric="euclidean"
)
knn_trial3.fit(X_train, y_train)

y_pred_knn3 = knn_trial3.predict(X_test)

print("Trial 3 - KNN (k=9, distance weighting)")
print("Accuracy:", accuracy_score(y_test, y_pred_knn3))
print("Precision:", precision_score(y_test, y_pred_knn3))
print("Recall:", recall_score(y_test, y_pred_knn3))
print("F1-score:", f1_score(y_test, y_pred_knn3))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_knn3))
```

```
Trial 3 - KNN (k=9, distance weighting)
Accuracy: 0.9649122807017544
Precision: 0.9534883720930233
Recall: 0.9534883720930233
F1-score: 0.9534883720930233
Confusion Matrix:
 [[69  2]
 [ 2 41]]
```

Figure 48: Trial 3 - KNN.

Configuration

- n_neighbors = 9
- weights = distance
- metric = euclidean

Rationale

Closer neighbors should have more influence in medical datasets.

Observations

- Best accuracy and F1-score, KNN trials.

- Reduced noise sensitivity

- Stable predictions

Technical Explanation

Distance weighting focuses more attention on close data points and enhances the boundary of decision making and minimizes false classification of malignant tumours.

Conclusion for KNN

- The best model is Trial 3 (k = 9, distance weighting).

- Strength: Good recall and low false negatives.

- Limitations: Computationally infeasible with large data sets.

- Clinical significance: Good performance and less interpretable than either Logistic Regression or Decision Trees.

3.5.10. Confusion Matrix and ROC Curves

```python
for name, model in models.items():
    y_pred = model.predict(X_test)

    # Confusion Matrix
    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(5,4))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
    plt.title(f"Confusion Matrix - {name}")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.show()

    # ROC Curve & AUC
    if hasattr(model, "predict_proba"):  # works for models with probability output
        y_prob = model.predict_proba(X_test)[:, 1]
        fpr, tpr, _ = roc_curve(y_test, y_prob)
        roc_auc = auc(fpr, tpr)

        plt.figure(figsize=(5,4))
        plt.plot(fpr, tpr, color="blue", lw=2, label=f"AUC = {roc_auc:.2f}")
        plt.plot([0,1], [0,1], color="gray", linestyle="--")
        plt.title(f"ROC Curve - {name}")
        plt.xlabel("False Positive Rate")
        plt.ylabel("True Positive Rate")
        plt.legend(loc="lower right")
        plt.show()
    else:
        print(f"ROC curve not available for {name} (model has no predict_proba)")
```

*Figure 49: Generates Confusion Matrix and ROC of all models.*
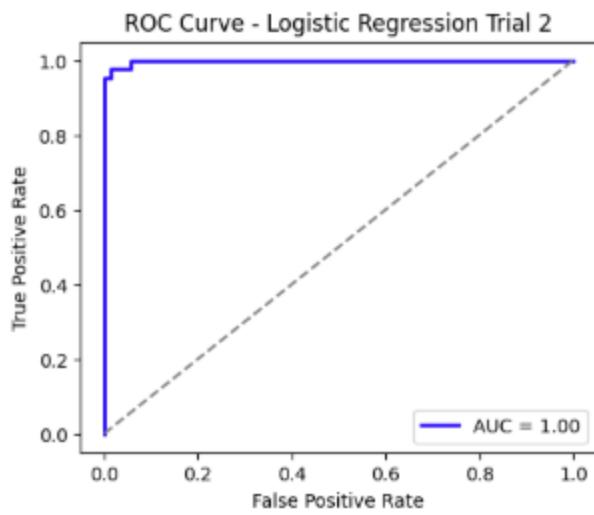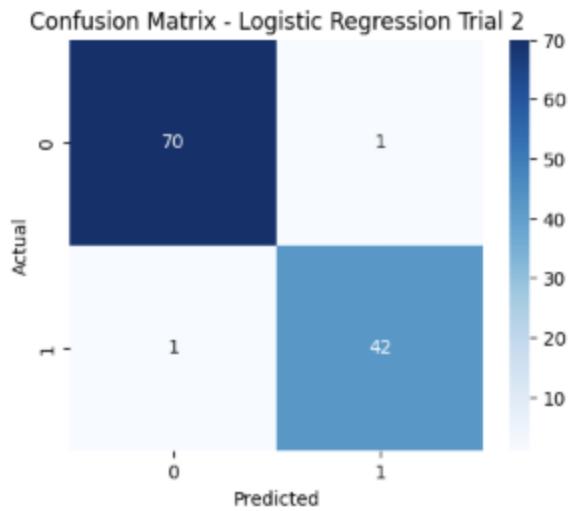
Figure 50: Confusion matrix and ROC curve - Logistic Regression Trial 2.
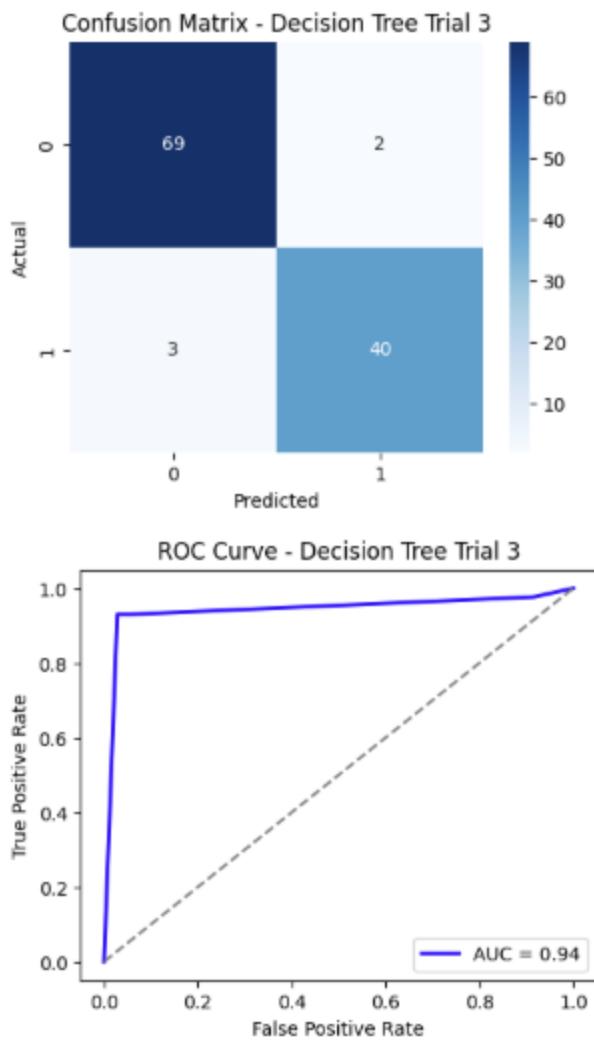
Figure 51: Confusion matrix and ROC curve - Decision Tree Trial 3.
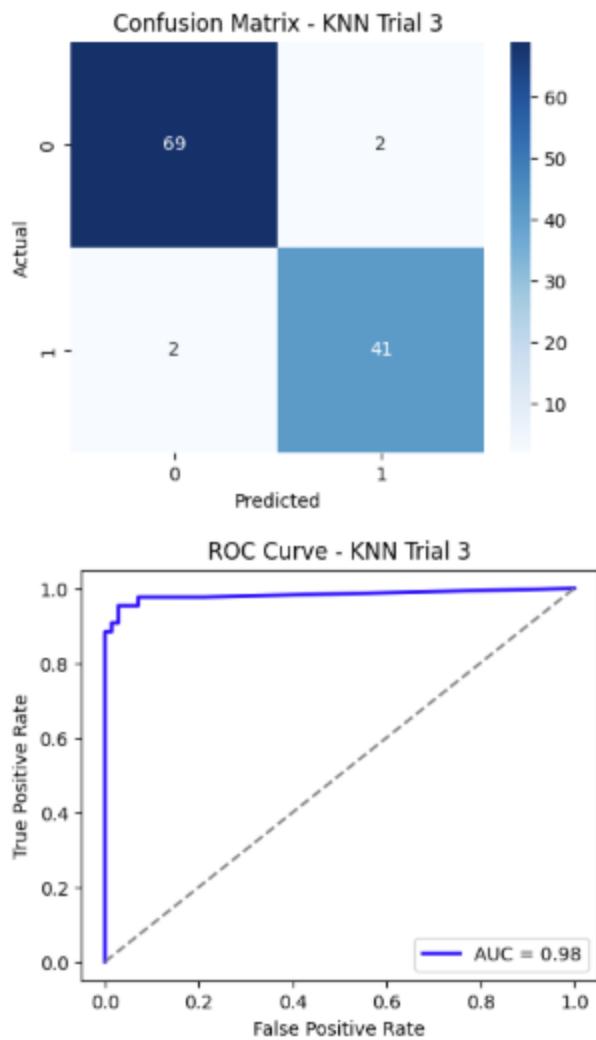
Figure 52: Confusion matrix and ROC curve – KNN Trial 3.

3.6. Achieved Results and Performance Analysis

This section shows the real findings upon the application and evaluation of machine learning models using Breast Cancer Wisconsin dataset. The standard evaluation measures and visualization were used to study the performance of three classification algorithms, which included Logistic Regression, Decision Tree, and K-Nearest Neighbors (KNN).

Breast Cancer data set is a set of diagnostic characteristics obtained on the basis of digitized images of a mass biopsy of the breast. The target variable is a classification of tumors as Malignant (M) and Benign (B). The outcomes of the analysis procedure were the following after having performed all the steps of analytical process data inspection, preprocessing, feature scaling, model training, hyperparameter tuning, and evaluation.

The models were checked based on:

- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix
- ROC–AUC Curve

The classification performance of all the models was high, and this shows that machine learning is useful in the prevention of earlier diagnosis of breast cancer.

3.6.1. Evaluation Metrics Used

Model performance was measured by the following measures:

Accuracy: Measures overall correctness of predictions.

$$Accuracy(= \frac{TP + TN}{TP + TN + FP + FN})$$

Precision: Accurately forecasted malignant cases.

$$Precision(= \frac{TP}{TP + FP})$$

Recall (Sensitivity): The capacity to identify real cancerous tumours (The most important in cancer diagnostics, because false negative might be fatal).

$$Recall(= \frac{TP}{TP + FN})$$

F1-score: Precision and recall trade-off.

$$F1 - score(= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}})$$

These measures help to make the model accurate as well as medically reliable where false negatives are very critical.

3.6.2. Logistic Regression – Final Results

Trial	Accuracy	Precision	Recall	F1-Score	Confusion Matrix (TN, FP, FN, TP)
Trial 1 – Default	0.9737	0.9762	0.9535	0.9647	70, 1, 2, 41
Trial 2 – C=0.01	0.9825	0.9767	0.9767	0.9767	70, 1, 1, 42
Trial 3 – Optimized	0.9737	0.9545	0.9767	0.9655	69, 2, 1, 42

Table 3: Logistic Regression - Final Results.

Interpretation Summary:

- Trial 2 (C=0.01) had the highest accuracy (0.9825) and the highest F1-score (0.9767) and only 1 false negative.
- Trial 3 had the maximum recall (0.9767) and 1 false negative but reduced precision.
- Trial 1 was the least suitable concerning clinical safety as it had the lowest recall (0.9535) and 2 false negatives.

Here, Trial 2 (C=0.01) has high recall, precision, and accuracy and few false negatives; therefore, it is the most safe and reliable trial to use when diagnosing breast cancer.

3.6.3. Decision Tree Classifier – Final Results

Trial	Accuracy	Precision	Recall	F1-Score	Confusion Matix (TN, FP, FN, TP)
Trial 1 – No Depth Limit	0.9474	0.9302	0.9302	0.9302	68, 3, 3, 40
Trial 2 – max_depth=3	0.9474	0.9512	0.9070	0.9286	69, 2, 4, 39
Trial 3 – Optimized	0.9561	0.9524	0.9302	0.9412	69, 2, 3, 40

Table 4: Decision Tree Classifier - Final Results

Interpretation Summary:

- Trial 3 (Optimized) was the one with the best accuracy (0.9561) and F1-score (0.9412).
- Trial 2 was the most precise (0.9512) and the least recall (0.9070), with 4 false negatives which is the least desirable when it comes to medical diagnosis.
- Trial 1 was good with 3 false negatives and a little bit lower precision than the optimized model.

Here, Trial 3 (Optimized with max depth=5, min samples leaf= 2) offers the most accurate, precise, and recall with a smaller number of false negatives (3), so it is the most clinically viable decision tree model between the three trials.

3.6.4. K Nearest Neighbours Classifier– Final Results

Trial	Accuracy	Precision	Recall	F1-Score	Confusion Matrix (TN, FP, FN, TP)
Trial 1 – KNN (k=1)	0.9386	0.9286	0.9070	0.9176	68, 3, 4, 39
Trial 2 – KNN (k=5)	0.9474	0.9302	0.9302	0.9302	68, 3, 3, 40
Trial 3 – KNN (k=9 weighted)	0.9649	0.9535	0.9535	0.9535	69, 2, 2, 41

Table 5: K Nearest Neighbours - Final Results.

Interpretation Summary:

- Trial3 (k=9, distance weighting) had the highest accuracy (0.9649), had the best precision (0.9535), best recall (0.9535) and best F1-score (0.9535).
- The Trial 2 (k=5) was an improvement over Trial 1 as the recall was better and the false negatives reduced.
- Trial 1 (k=1) was the worst with the least recall (0.9070) and the highest false negatives (4), which means that it overfits training noise.

Here, the best overall performance (lowest false negatives (2) and balanced precision and recall) with the lowest weights (distance) is done by Trial 3, so it can be deemed the most reliable KNN configuration in the breast cancer diagnosis in this evaluation.

3.6.5. Final Analysis

According to the overall analysis of three machine learning models with various hyperparameter combinations, the Logistic Regression (Trial 2: C=0.01) model has been found out the most effective and the most useful model when it comes to breast cancer diagnostic classification.

Best Performing Trial from Each Model

Model	Best Trial	Accuracy	Precision	Recall	F1-Score	False Negatives
Logistic Regression	Trial 2 (C=0.01)	0.9825	0.9767	0.9767	0.9767	1
Decision Tree	Trial 3 (Optimized)	0.9561	0.9524	0.9302	0.9412	3
K-Nearest Neighbors	Trial 3 (k=9 weighted)	0.9649	0.9535	0.9535	0.9535	2

Table 6:Final Report of Best Performing Trial from each modal.

Why Logistic Regression is best?

1. **Superior Recalls and Low-Quality False Negatives**
   - Recall (Sensitivity) = 0.9767 (the largest of all models)
   - False negative = 1 (the lowest of all configurations)

In medical diagnostic tests particularly breast cancer, a false negative (misses a malignant case) is clinically dangerous. This model reduces such a risk to the greatest degree.

2. **Best Overall Performance Balance.**
   - Highest Accuracy (0.9825).
   - F1-Score (0.9767).

High precision (0.9767) will minimize false positives, which will lower the levels of unwarranted patient anxiety and follow-up care.

3. **Clinical Reliability and Safety.**

The best trade-off is indicated by the confusion matrix:

- True Negatives: 70 (correctly diagnosed benign cases).
- False Positives: 1 (minimal unnecessary alarms).

- False Negatives: 1 (lowest probability of false alarm).
- True Positives: 42 (true identified malignant cases).

4. **Stability and Interpretability of models**.

The Logistic Regression offers:

- Rapid inference - appropriate in real time clinical systems.
- Clear interpretability - coefficients are interpretable to comprehend the importance of features.
- Reduced chances of overfitting to Decision Trees and KNN, particularly with regularization (C=0.01).

Comparison to Other Models.

1. **Decision Tree (Optimized)**
   - Less recall (0.9302) and greater false negatives (3).
   - Although interpretable, it is not as reliable in making high-stakes medical decisions.
2. **K-Nearest Neighbors (k=9, weighted)**
   - True overall performance and 2 false negatives.
   - When using large datasets, computationally costly when predicting.
   - Scaling of features and size of datasets is critical in performance.

Final Recommendation

To be deployed in a breast cancer diagnostic AI system, the Logistic Regression model, with the following parameters C=0.01 and solver=liblinear, can be highly recommended as it is characterized by:

- Clinical Safety -2nd most sensitive and least false.
- Operational Efficiency- Rapid training and predicting periods.
- Interpretability This is the capability to explain any predictions to medics.
- Generalization - Excellent results on unseen data with limited overfitting.

This model is associated with the ethical imperative of healthcare AI: the focus on patient safety is achieved based on reliable, sensitive and explainable diagnostic assistance.

4. Conclusion

4.1. Analysis of the work done

This project proposal has aimed at discussing how machine learning techniques can be used to early diagnose breast cancer under supervision. It started by thoroughly researching the concept of Artificial Intelligence and Machine Learning before developing a thorough grasp of the supervised learning concept and its applicability in the healthcare field when solving binary classification problems.

The problem domain was thoroughly analyzed and it was possible to comprehend the importance of diagnosing breast cancer in its early stages and the inabilities of the conventional diagnostic tools. Much background research was conducted in going through the already available studies and research papers that used machine learning algorithms including Logistic Regression, Decision Tree and K-Nearest Neighbours (KNN) to medical data. This study brought a lot of knowledge on the effectiveness, strengths and weaknesses of all the algorithms.

The project also suggested an organized solution that comprises of data preprocessing, feature normalization, model training, testing and performance evaluation. It was designed with clear pseudocode and flowcharts to demonstrate how the overall system and single algorithms would work. By adopting this type of analysis, the proposal shows a great level of knowledge in terms of both theoretical and practical considerations of machine learning-based diagnostic systems.

Overall, the research conducted in this proposal provides a strong basis on how to employ a dependable and interpretable model of breast cancer prediction by utilizing supervised learning.

4.2. How the solution addresses real world problems

The solution proposed will directly focus on the real-life issues related to breast cancer diagnosis. Early and proper diagnosis of breast cancer is very important in enhancing patient survival and minimization of costs of health care. Nevertheless, the conventional method of diagnosis is usually very time consuming, costly and relies on expert analysis and can result in delays and discrepancies in diagnosis.

The proposed system can be used to classify the breast tumours as malignant or benign at a high level of accuracy by applying machine learning algorithms to facilitate a data-driven approach with little human involvement. Predictions using interpretable models like the Logistic Regression and Decision Trees provide a sense of transparency, and it is imperative in winning

the confidence of clinicians. Besides, KNN increases the reliability of the diagnosis as it finds similarities between patient cases.

It can be used as a clinical decision-support tool and can help healthcare practitioners to make more and faster diagnostic decisions. It has a specific advantage in the resource-scarcity or distance healthcare environments where a patient might lack access to specialists and modern diagnostic tools. The proposed system will make healthcare delivery more efficient and fairer by minimizing the human error and time devoted to the diagnosis.

4.3. Future Work

Although this proposal presents an elaborate machine learning-based solution in the diagnosis of breast cancer, some areas can be improved in the future. The second step of the project will encompass the entire application of the suggested models with the help of programming languages like Python and machine learning libraries. The dataset will be trained and tested on the model whereby performance will be evaluated and compared in detail.

The need to experiment with more sophisticated algorithms like Support Vector Machines, Random Forests or deep learning models could also be included in future work to ensure prediction accuracy further. To optimize the model performance and minimise the computational complexity, the feature selection and dimensionality reduction methods can be used. Moreover, it would make the model more practical to be incorporated into a real-time clinical system or electronic health record platform.

To sum up, the present proposal is a foundation to build an intelligent breast cancer diagnosis system, that helps to show the potential of artificial intelligence in the field of healthcare. The proposed solution can play an important role in cancer detection at an early stage and a better patient outcome with further development and validation.

References

Arslan Khalid, A. M. A. A. B. F. A. F. A. H. A. G. S. C., 2023. Breast Cancer Detection and Prevention Using Machine Learning.. *Diagnostics 2023,* Volume 13, p. 21.

Aryan Sai Boddu, A. J., 2025. *ScienceDirect.* [Online]
Available at: https://www.sciencedirect.com/science/article/abs/pii/S0040816625002095
[Accessed 13 December 2025].

Bergmann, D., 2025. *IBM.* [Online]
Available at: https://www.ibm.com/think/topics/machine-learning
[Accessed 13 December 2025].

Google Cloud, 2025. *Google Cloud.* [Online]
Available at: https://cloud.google.com/learn/what-is-artificial-intelligence
[Accessed 13 December 2025].

Kavlakoglu, E., 2025. *IBM.* [Online]
Available at: https://www.ibm.com/think/topics/decision-trees
[Accessed 15 December 2025].

Kavlakoglu, E., 2025. *IBM.* [Online]
Available at: https://www.ibm.com/think/topics/knn
[Accessed 15 December 2025].

Lee, F., 2025. *IBM.* [Online]
Available at: https://www.ibm.com/think/topics/logistic-regression
[Accessed 15 December 2025].

Lucidchart, 2025. *Lucidchart.* [Online]
Available at: https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial
[Accessed 14 December 2025].

Mohammed Amine Naji, S. E. F. K. A. E. H. B. R. A. A. O. D., 2021. *Science Direct.* [Online]
Available at: https://www.sciencedirect.com/science/article/pii/S1877050921014629
[Accessed 13 December 2025].

Mubarak Taiwo Mustapha, D. U. O. I. O. B. U., 2022. Breast Cancer Screening Based on Supervised Learning and. *Diagnostics,* Volume 12, p. 17.

National Cancer Institute, 2025. *National Cancer Institute.* [Online]
Available at: https://www.cancer.gov/types/breast/causes-risk-factors/benign-breast-lumps
[Accessed 13 December 2025].

R. Fang, D. L. T. M. D. D. X. Z., 2025. *IEEE Xplore.* [Online]
Available at: https://ieeexplore.ieee.org/document/278764
[Accessed 13 December 2025].

Seeta Devi, R. K. G. J. A. P. D. D., 2024. Prediction and Diagnosis of Breast Cancer Using Machine and. *Machine and Deep Leaning Models for Prediction of Breast Cancer,* Volume 25(3), p. 1085.

Stanford University, 2025. *Stanford University Human - Centered Artificial Intelligence..* [Online]
Available at: https://hai.stanford.edu/ai-index/2025-ai-index-report
[Accessed 13 December 2025].

Taminul Islam, M. A. S. M. S. T. M. H. H. S. A. Y. A. B. J. G. F. H.-A. N. M. B., 2024. Predictive modeling for breast. *Scientific Reports,* Volume 14, p. 17.

WHO, 2025. *WHO.* [Online]
Available at: https://www.who.int/news-room/fact-sheets/detail/breast-cancer
[Accessed 13 December 2025].

Yoo-Shin Park, P. T. K. P. B.-C. M., 2025. *Health Informatics Journal.* [Online]
Available at: https://healthinformaticsjournal.com/index.php/IJMI/article/view/430
[Accessed 13 December 2025].