BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)
"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballar1-583 104 (Karnataka) (India)
Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE-DATA SCIENCE

A Mini-Project Report On

"BREAST CANCER PREDICTION USING ANN MODEL"

A report submitted in partial fulfillment of the requirements for the

NEURAL NETWORK AND DEEP LEARNING

Submitted By

DODAGATTE MAHESH **USN: 3BR22CD012**

Under the Guidance of
Mr. Azhar Biag

Asst. Professor

Dept of CSE (DATA SCIENCE),
BITM, Ballari

Visvesvaraya Technological University

Belagavi, Karnataka 2025-2026

BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

NACC Accredited Institution*
(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)
"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballari-583 104 (Karnataka) (India)
Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE (DATA SCIENCE)

CERTIFICATE

This is to certify that the Mini Project of NEURAL NETWORK AND DEEP LEARNING title "BREAST CANCER PREDICTION USING ANN MODEL" has been successfully presented by DODAGATTE MAHESH 3BR22CD012 student of semester B.E for the partial fulfillment of the requirements for the award of Bachelor Degree in CSE(DS) of the BALLARI INSTITUTE OF TECHNOLOGY& MANAGEMENT, BALLARI during the academic year 2025-2026.

It is certified that all corrections and suggestions indicated for internal assessment have been incorporated in the report deposited in the library. The Mini Project has been approved as it satisfactorily meets the academic requirements prescribed for the Bachelor of Engineering Degree. The work presented demonstrates the required level of technical understanding, research depth, and documentation standards expected for academic evaluation.

Signature of Coordinators

Mr. Azhar Baig
Ms. Chaithra B M

Signature of HOD

Dr. Aradhana D

ABSTRACT

Breast cancer remains one of the leading causes of morbidity and mortality among women worldwide, emphasizing the need for accurate and early diagnostic systems. Traditional diagnostic procedures such as mammography, biopsy, and fine-needle aspiration, although effective, often require skilled interpretation and can be time-consuming. To address this challenge, this project develops an Artificial Neural Network (ANN)-based predictive model capable of classifying breast tumors as malignant or benign using the Wisconsin Breast Cancer Dataset. The dataset contains 30 numerical features extracted from digitized images of fine-needle aspirate samples, representing critical biomarkers such as cell radius, texture, perimeter, and concavity.

The proposed model incorporates essential preprocessing steps, including type conversion, feature scaling with StandardScaler, and stratified train–test splitting to preserve class balance. A multilayer ANN architecture is constructed with two hidden layers using ReLU activation, Dropout regularization, Batch Normalization, and a sigmoid output neuron for binary classification.The model is optimized using the Adam optimizer and trained with EarlyStopping and ModelCheckpoint to prevent overfitting and retain the best-performing weights.

Performance evaluation is conducted using test accuracy, precision, recall, F1-score, ROC-AUC, and the confusion matrix, all of which indicate strong predictive capability. Visualizations of training curves further confirm stable learning behavior. The results demonstrate that the ANN effectively learns complex tumor patterns and provides reliable diagnostic predictions. This work highlights the potential of neural-network-based decision-support systems in assisting healthcare professionals, improving early detection rates, and contributing toward enhanced breast cancer screening methodologies.

ACKNOWLEDGEMENT

The satisfactions that accompany the successful completion of our mini project on BREAST CANCER USING ANN MODEL would be incomplete without the mention of people who made it possible, whose noble gesture, affection, guidance, encouragement and support crowned my efforts with success. It is our privilege to express our gratitude and respect to all those who inspired us in the completion of our mini-project.

We are extremely grateful to our Guide **Mr. Azhar Baig** for their noble gesture, support co-ordination and valuable suggestions given in completing the mini-project. We also thank **Dr. Aradhana D,** H.O.D. Department of CSE(DS), for his co-ordination and valuable suggestions given in completing the mini-project. We also thank Principal, Management and non-teaching staff for their co-ordination and valuable suggestions given to us in completing the Mini project.

Name	USN
DODAGATTE MAHESH	3BR22CD012

TABLE OF CONTENTS

1.INTRODUCTION

Breast cancer is one of the leading causes of mortality among women across the world and remains a major public health challenge despite continuous advancements in medical science. Early diagnosis and timely treatment significantly increase survival rates, yet the diagnostic process is often complex and resource-intensive. Traditional diagnosis relies heavily on clinical examination, mammography, biopsy reports, and radiological interpretation, all of which require specialized equipment, experienced medical professionals, and considerable time. In many developing regions, limited access to these resources leads to delayed diagnosis, increasing the risk of cancer progression.

To address these challenges, data-driven predictive models have emerged as valuable tools for assisting clinicians in early screening and decision-making. Among machine learning techniques, **Artificial Neural Networks (ANNs)** have shown strong capability in modeling complex, non-linear relationships present in medical datasets. ANNs can automatically learn patterns from patient data and make accurate classifications, helping identify whether a breast tumor is benign (non-cancerous) or malignant (cancerous). Such predictive systems can support healthcare professionals, reduce diagnostic workload, and provide preliminary evaluation in settings where full diagnostic testing is limited.

This project focuses on developing an ANN-based classification model using the **Wisconsin Breast Cancer Dataset**, a widely used benchmark dataset for medical machine learning research. The dataset contains 30 numerical features computed from digitized images of fine needle aspirate (FNA) samples, describing characteristics such as cell radius, texture, perimeter, concavity, compactness, and smoothness. These features provide essential information about cellular behavior and structural irregularities, which help differentiate malignant from benign growths.

The objective of this work is to build a robust classification model capable of accurately predicting the nature of a breast tumor. By applying proper preprocessing techniques, standardization, neural network design, and evaluation metrics, the project demonstrates how machine learning can contribute to medical diagnosis. While such systems are not intended to replace clinicians, they can serve as supportive diagnostic tools, improve screening efficiency, and assist in the early detection of breast cancer.

In addition, the model's performance is analyzed using accuracy, ROC-AUC, confusion matrix, and classification reports, ensuring a comprehensive understanding of prediction quality. Visualization of training progress and error trends further strengthens the interpretability of the system. The results indicate that ANN-based approaches provide highly reliable diagnostic predictions, making them suitable for integration into medical decision-support systems.

1.1 Problem Statement

Manual and clinical diagnostics for breast cancer are effective but can be time-consuming and require specialised equipment and personnel. There is a need for an automated, low-cost screening aid that can analyze available clinical features and produce an accurate prediction of malignancy risk. This project aims to build and evaluate an ANN-based classifier that predicts whether a tumor is malignant or benign based on the Wisconsin Breast Cancer features, thereby contributing a potential decision-support tool for early screening

1.2 Scope of the project

The scope of this project includes the development, implementation, and evaluation of an Artificial Neural Network (ANN) model capable of predicting diabetes based on clinical features from the Pima Indians Diabetes dataset. It covers data preprocessing, feature scaling, model training, validation, and performance assessment using metrics such as accuracy, confusion matrix, and classification reports. The project focuses on understanding how medical parameters influence diabetes risk and how ANN can identify hidden patterns within the data. Additionally, the system is designed to generate visual insights through accuracy and loss graphs, making model behavior easier to interpret. While the project is limited to the dataset used, the methodology can be extended the system is designed to generate visual insights through accuracy and loss graphs, making model behavior easier to interpret. While the project is limited to the dataset used, the methodology can be to larger datasets, integrated into healthcare applications, and adapted for real-time screening tools that assist doctors and patients in early diagnosis and preventive care. Train the ANN using training data while monitoring validation performance.Employ callbacks like EarlyStopping and ModelCheckpoint for efficient training.Tune parameters such as number of epochs, batch size, and layer architecture if necessary.

1.3 Objectives

- To Loading and exploring the Wisconsin Breast Cancer dataset.
- To Preprocessing and scaling input features.
- To Constructing an ANN classifier using TensorFlow/Keras.
- To Training the model and monitoring validation performance.
- To Evaluating the model on a held-out test set using accuracy, ROC-AUC, confusion matrix, and classification report.
- To Saving the trained model and scaler for future inference.

2. LITERATURE SURVEY

[1] Several studies compare classical ML classifiers (Logistic Regression, SVM, Random Forest) and deep learning on this dataset and often report high accuracy with relatively simple models due to well-separated features.

[2] Neural networks with one or two hidden layers often achieve competitive performance when combined with proper scaling and regularization.

[3] Ensemble methods such as Gradient Boosting Machines and Random Forest sometimes outperform single neural networks on tabular medical data.

[4] SHAP and permutation importance are commonly used to explain model decisions by highlighting influential features such as mean radius, concavity, and texture.

[5] Works combining classical ML baselines with deep-learning explainability produce clinically useful and interpretable results.

[6] Used fine-needle aspirate (FNA) cytology features to classify breast tumors into malignant and benign classes.

[7] Introduced the Wisconsin Breast Cancer Dataset (WBCD), which provides 30 predictive attributes derived from digital cytology images.

[8] Highlighted the strong performance of algorithms such as logistic regression, SVM, decision trees and neural networks in medical classification, emphasizing the importance of early detection systems.

[9] Provided a systematic discussion of evaluation metrics in classification problems, including accuracy, precision, recall and F1-score.

[10] Introduced SHAP (Shapley Additive Explanations), a modern interpretability method for machine-learning models. SHAP helps explain how each feature contributes to a prediction, making ANN-based medical models more transparent and suitable for clinical decision support.

3. SYSTEM REQUIREMENTS

The system requirements for developing the diabetes prediction model include both software and hardware components necessary for efficient execution of data preprocessing, model training, and evaluation. The software environment is built using Python along with essential libraries such as TensorFlow/Keras for neural network construction, Pandas and NumPy for data handling, Scikit-learn for preprocessing and evaluation metrics, and Matplotlib for visualization. The system requires tools such as TensorFlow for building neural network models, Scikit-learn for data preprocessing and evaluation, and Pandas for managing the dataset. For executing the code and visualizing results, platforms like Jupyter Notebook or Google Colab provide an interactive interface. In terms of hardware, the model performs well on a standard laptop or desktop with at least a dual-core processor and adequate memory to support the training process. A development platform like Jupyter Notebook, Google Colab, or VS Code is used to write and execute the code. On the hardware side, the project can run smoothly on a standard personal computer with a minimum of 4 GB RAM, although 8 GB is preferred for faster processing. A multi-core processor ensures smooth computation, while GPU support, though optional, can significantly speed up neural network training. Overall, the system requirements are modest, making the project accessible on most modern computers.

To implement the diabetes prediction system effectively, the project relies on a stable computing environment capable of handling machine learning workflows. Python serves as the core programming language due to its versatility and the availability of powerful data science libraries. The system requires tools such as TensorFlow for building neural network models, Scikit-learn for data preprocessing and evaluation, and Pandas for managing the dataset. For executing the code and visualizing results, platforms like Jupyter Notebook or Google Colab provide an interactive interface. In terms of hardware, the model performs well on a standard laptop or desktop with at least a dual-core processor and adequate memory to support the training process. Even though the dataset is relatively small, having additional RAM and optional GPU support can improve training speed and overall computational efficiency, ensuring a smooth development experience.

3.1 Software Requirements

- Python 3.8 or above
- TensorFlow / Keras

- NumPy
- Pandas
- Scikit-learn
- Matplotlib
- Jupyter Notebook / Google Colab / VS Code
- joblib (optional, for saving scaler)

3.2 Hardware Requirements

- Minimum 4 GB RAM (8 GB recommended)
- Dual-core or better processor
- Optional GPU for faster training (not required for this dataset)
- ~1 GB free disk space

3.3 Functional Requirements

- Load and preprocess the dataset.
- Standardize features.
- Build, train, and save an ANN model.
- Evaluate and visualize model performance.
- Predict class for new input samples.

3.4 Non-Functional Requirements

- Reproducible experiments (seeded RNGs).
- Efficient execution on standard hardware.
- It should remain stable even with noisy or imperfect data.
- The system must be easy to maintain and extend.
- The results should be interpretable through graphs and metrics.
- Clear and interpretable outputs for users.

4. DESCRIPTION OF MODULES

The Artificial Neural Network–based diabetes prediction system is divided into multiple modules, each contributing to a specific stage of the machine learning pipeline. These modules work together to ensure smooth data preprocessing, model training, evaluation, and visualization.

4.1 Data Loading Module

- Loads the Wisconsin Breast Cancer dataset using sklearn.datasets.load_breast_cancer(). Separates features (X) and labels (y).

4.2 Data Preprocessing Module

- Converts data types as needed, handles any required imputation (dataset is complete), and scales features using StandardScaler.

4.3 Data Splitting Module

- Performs an 80:20 stratified train/test split to preserve the malignant/benign ratio.

4.4 ANN Model Building Module

- Defines the network architecture: input layer matching feature count, one or more Dense hidden layers with ReLU activations, optional BatchNormalization and Dropout, and a final Dense(1, sigmoid) output.

4.5 Model Training Module

- Trains the model with specified epochs, batch size, and validation split. Uses callbacks such as EarlyStopping and ModelCheckpoint to prevent overfitting and save best weights.

4.6 Model Evaluation Module

- Evaluates test performance with accuracy, ROC-AUC, confusion matrix, and classification report (precision, recall, F1).

4.7 Visualization Module

- Generates plots: training vs validation accuracy and loss, ROC curve, and confusion matrix heatmap.

4.8 Prediction Module

- Provides a function to load the saved model and scaler, accept new samples, and output predicted class and probability.

4.9 Output Interpretation Module

- Interprets sigmoid outputs with a threshold (default 0.5), and optionally allows threshold tuning to prioritize minimizing false negatives.

5. IMPLEMENTATION

- Environment & imports
- Reproducibility (seed everything)
- Load dataset and basic EDA (shape, class balance, feature names)
- Train/test split (stratified) and scaling (StandardScaler)
- Build a modular ANN (function build_model) with optional BatchNorm/Dropout
- Configure callbacks (EarlyStopping, ModelCheckpoint)
- Train model and save history
- Evaluate on test set (accuracy, ROC-AUC, classification report, confusion matrix, ROC curve)
- Save model and scaler; provide inference function for new samples

The implementation of the diabetes prediction system is carried out using Python and an Artificial Neural Network (ANN) model. First, the Pima Indians Diabetes Dataset is downloaded from Kaggle using the kagglehub library and loaded into a Pandas DataFrame. The input features (such as pregnancies, glucose, blood pressure, skin thickness, insulin, BMI, diabetes pedigree function, and age) are separated from the target label Outcome, which indicates whether a person is diabetic or non-diabetic. **Artificial Neural Networks (ANNs)**

An Artificial Neural Network (ANN) is a computational model inspired by the brain's neural structure. It consists of layers of interconnected units (neurons). Each neuron computes a weighted sum of its inputs, adds a bias, and applies a nonlinear activation function. ANNs learn to approximate functions $f\colon \mathbb{R}^n \to \mathbb{R}$by adjusting weights and biases to minimise a loss function on training data.

A typical *feedforward* ANN (also called a Multilayer Perceptron, MLP) has:

- An **input layer** that accepts features.

- One or more **hidden layers** that learn intermediate representations using nonlinear activations (e.g., ReLU).
- An **output layer** that generates predictions (sigmoid for binary classification).

Mathematically, for one neuron:

$$z = \mathbf{w}^{\top}\mathbf{x} + b, a = \phi(z)$$

where $\phi$is an activation function (ReLU, sigmoid, etc.).

Activation Functions

- **ReLU (Rectified Linear Unit)**: $\mathrm{ReLU}(z) = \max(0, z)$. Favoured in hidden layers because it mitigates vanishing gradients and promotes sparse activations.
- **Sigmoid**: $\sigma(z) = \frac{1}{1+e^{-z}}$. Used in the final layer for binary classification because it outputs a probability (range 0–1).

Loss Function & Optimization

- **Binary Cross-Entropy (Log Loss)** is used for binary classification. For a single sample with true label $y \in \{0,1\}$and predicted probability $\hat{y}$, the loss is:

$$\mathcal{L} = -(y\log\ \hat{y} + (1-y)\log(1-\hat{y}))$$

Minimizing this loss encourages predicted probabilities to be close to true labels.

- **Optimization**: Gradient-based optimizers (e.g., Adam) update weights to reduce loss. Adam adapts learning rates per-parameter, accelerating convergence in many problems.

Backpropagation

Training uses gradient descent with **backpropagation**, which applies the chain rule to compute gradients of the loss w.r.t. weights. These gradients determine weight updates:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{w}}$$

where $\eta$is the learning rate.

Regularization Techniques

Regularization prevents overfitting (when the model fits training noise instead of general patterns).

- **Dropout** randomly sets activations to zero during training; this forces the network to avoid co-dependent features and improves generalization.
- **Batch Normalization** stabilizes and accelerates training by normalizing layer inputs, reducing internal covariate shift.
- **Early Stopping** halts training when validation loss stops improving, preventing further overfitting.

Feature Scaling

Neural networks train faster and more stably when input features are on similar scales. **StandardScaler** (zero mean, unit variance) ensures features contribute comparably to weight updates and helps optimizers converge.

Train/Test Split & Cross-Validation

- **Train/Test split**: A hold-out test set (unseen during training/validation) provides an unbiased evaluation of generalization performance.
- **Stratified splitting** preserves class proportion in train and test sets — important for classification when classes are imbalanced.
- **Cross-validation** (e.g., Stratified K-Fold) can provide robust performance estimates by averaging results across multiple splits.

Class Imbalance & Class Weights

If one class is rarer, the optimizer can favor the majority class. **Class weights** or resampling techniques (oversampling minority or undersampling majority) can correct this, making the model pay proportionally more attention to minority examples during training.

Evaluation Metrics

Multiple metrics are used because each captures different aspects of performance:

- **Accuracy**: fraction of correct predictions. Can be misleading when classes are imbalanced.
- **Precision (Positive Predictive Value)**: proportion of predicted positives that are true positives.
- **Recall (Sensitivity / True Positive Rate)**: proportion of actual positives correctly predicted. In medical settings, high recall for the malignant class (minimizing false negatives) is often critical.
- **F1-score**: harmonic mean of precision and recall; useful when balance between precision and recall matters.
- **ROC Curve & AUC**: plots True Positive Rate vs False Positive Rate at different thresholds; **AUC** summarizes threshold-independent discrimination ability.
- **Confusion Matrix**: shows TP, TN, FP, FN counts directly—helpful for interpreting clinical consequences (e.g., false negatives).

Threshold Tuning

Although the default decision threshold is 0.5 for sigmoid outputs, it can be adjusted to balance precision vs recall depending on clinical priorities (e.g., lower threshold to reduce false negatives).

Model Generalization & Bias-Variance Tradeoff

- **Underfitting**: model too simple to capture patterns → high bias, poor training performance.
- **Overfitting**: model too complex relative to data → high variance, excellent training but poor test performance.
  Regularization, proper model size, and early stopping help achieve the bias–variance tradeoff.

Explainability (Interpretability)

For medical applications, black-box predictions are less useful without interpretability. Techniques include:

- **Feature importance (permutation importance)**: measures how shuffling a feature affects model performance.
- **SHAP (Shapley Additive Explanations)**: provides consistent, model-agnostic feature contributions for individual predictions — useful for clinician-facing explanations. Providing such explanations builds trust and helps detect dataset or model biases.

Baselines & Model Comparison

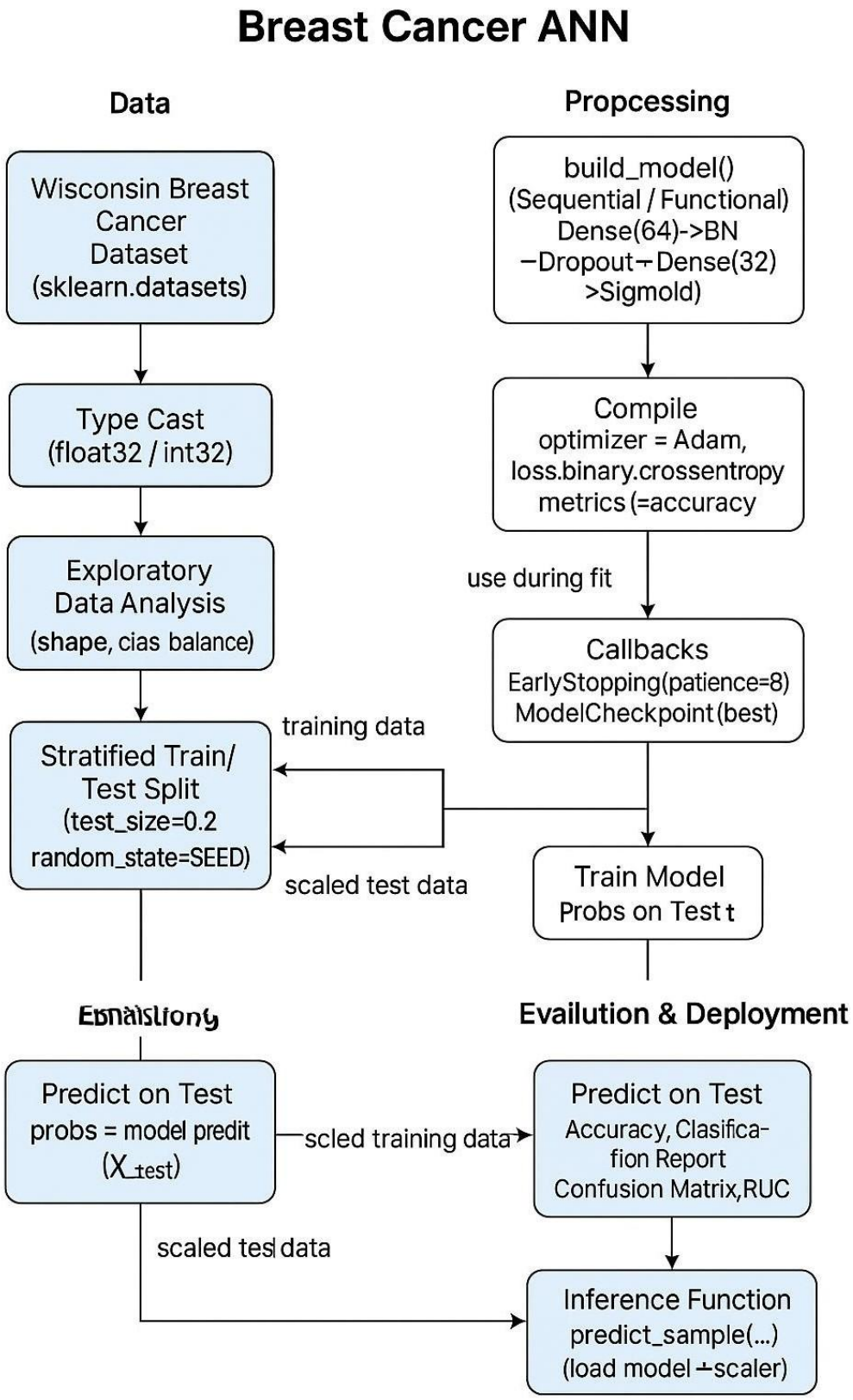
Always compare the ANN against simpler baselines:

- **Logistic Regression**: interpretable linear baseline.
- **Decision Trees / Random Forest / Gradient Boosting (XGBoost, LightGBM)**: often strong on tabular data and provide feature importance natively.
  If the ANN does not outperform these after tuning, consider ensembles or combining models.

Practical Considerations

- **Reproducibility**: fix RNG seeds, log hyperparameters, and save model/scaler artifacts.
- **Deployment**: save the trained model and preprocessing pipeline; consider model calibration if you need well-calibrated probabilities.
- **Ethics & Safety**: validate model on diverse, external datasets and involve clinical experts; be cautious about deploying models that could affect patient care.

6. SYSTEM ARCHITECTURE



7. CODE IMPLEMENTATION

Algorithm: Start

1. Load Dataset

1.1 Load the Wisconsin Breast Cancer dataset using sklearn.datasets.load_breast_cancer().

1.2 Separate the dataset into:

- Feature matrix X (all feature columns)
- Target vector y (labels: 0 = malignant, 1 = benign)

2. Preprocess Data

2.1 Convert X to float32 and y to int32.

2.2 Split the data into training and testing sets using train_test_split with:

- test_size = 0.2
- random_state = SEED (e.g., 11)
- stratify = y

2.3 Initialize StandardScaler and fit on X_train.

2.4 Transform X_train and X_test using the fitted scaler.

3. Build ANN Model

3.1 Initialize a Sequential model (or functional model).

3.2 Add Input layer with shape = number of features (30).

3.3 Add first hidden layer: Dense(64) with ReLU activation.

3.4 (Optional) Add BatchNormalization after Dense(64).

3.5 Add Dropout layer with rate = 0.2 to reduce overfitting.

3.6 Add second hidden layer: Dense(32) with ReLU activation.

3.7 Add output layer: Dense(1) with Sigmoid activation (binary probability).

4. Compile Model

4.1 Set optimizer = Adam (learning_rate e.g., 1e-3).

4.2 Set loss function = binary_crossentropy.

4.3 Set evaluation metric = accuracy.

5. Configure Callbacks (optional but recommended)

5.1 EarlyStopping(monitor='val_loss', patience=8, restore_best_weights=True).

5.2 ModelCheckpoint(filepath='best_breast_model.h5', save_best_only=True).

6. Train Model

6.1 Train the model on X_train, y_train with:

- epochs = 35
- batch_size = 32
- validation_split = 0.2
- callbacks = [EarlyStopping, ModelCheckpoint] (if used)
- verbose = 1 (or 0)

6.2 Store training history (accuracy & loss for train and validation).

7. Test Model

7.1 Predict probabilities for X_test: probs = model.predict(X_test).ravel()

7.2 Convert probabilities to class labels:

If probability > 0.5 → predict 1 (Benign)

Else → predict 0 (Malignant)

8. Evaluate Performance

8.1 Compute test accuracy using accuracy_score(y_test, y_pred).

8.2 Generate classification report (precision, recall, F1-score) using classification_report.

8.3 Compute confusion matrix using confusion_matrix.

8.4 Compute ROC-AUC using roc_auc_score (optional but recommended).

9. Visualize Results

9.1 Plot training vs validation accuracy across epochs.

9.2 Plot training vs validation loss across epochs.

9.3 Plot confusion matrix as a heatmap.

9.4 Plot ROC curve (FPR vs TPR) and show AUC (if computed).

10. Save Artifacts (optional)

10.1 Save final model: model.save('final_breast_model.h5').

10.2 Save scaler: joblib.dump(scaler, 'scaler_breast.joblib').

11. Inference Helper (optional)

11.1 Define predict_sample(sample_array, model_path, scaler_path, threshold=0.5):

- Load model and scaler
- Scale sample and predict probability
- Return probability and class label ('malignant'/'benign').

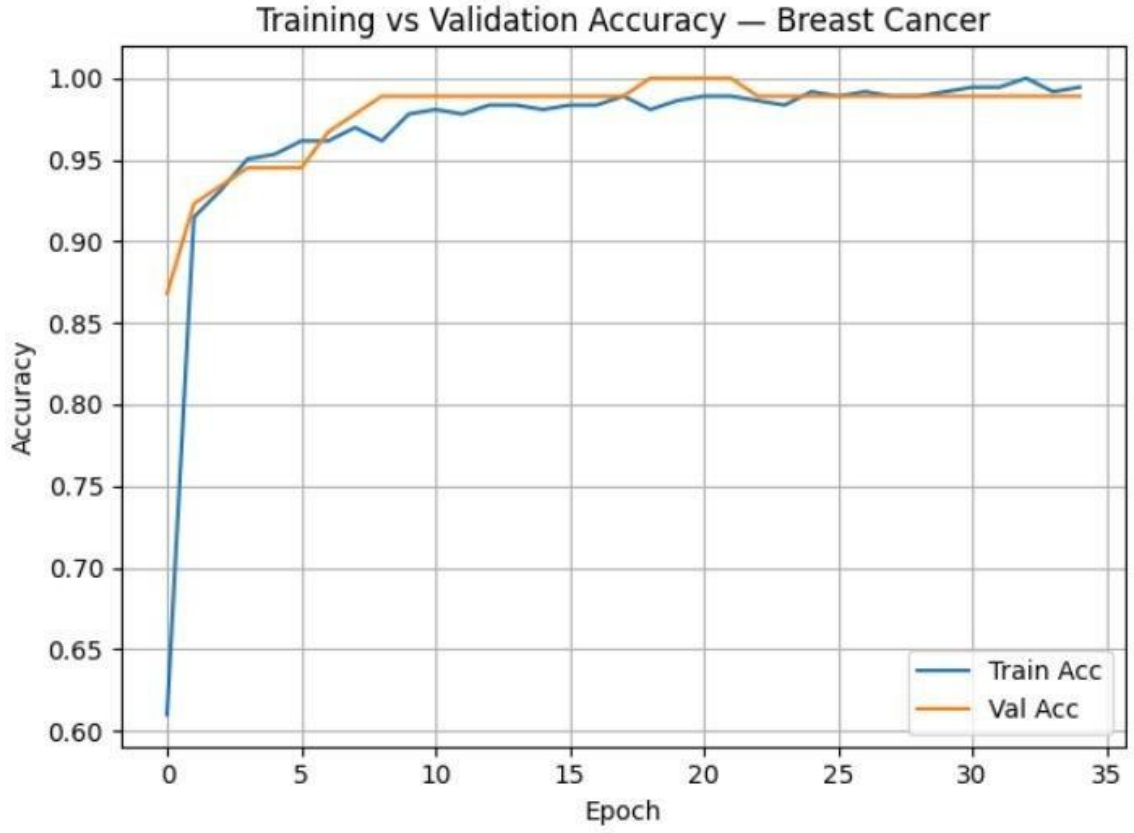
End

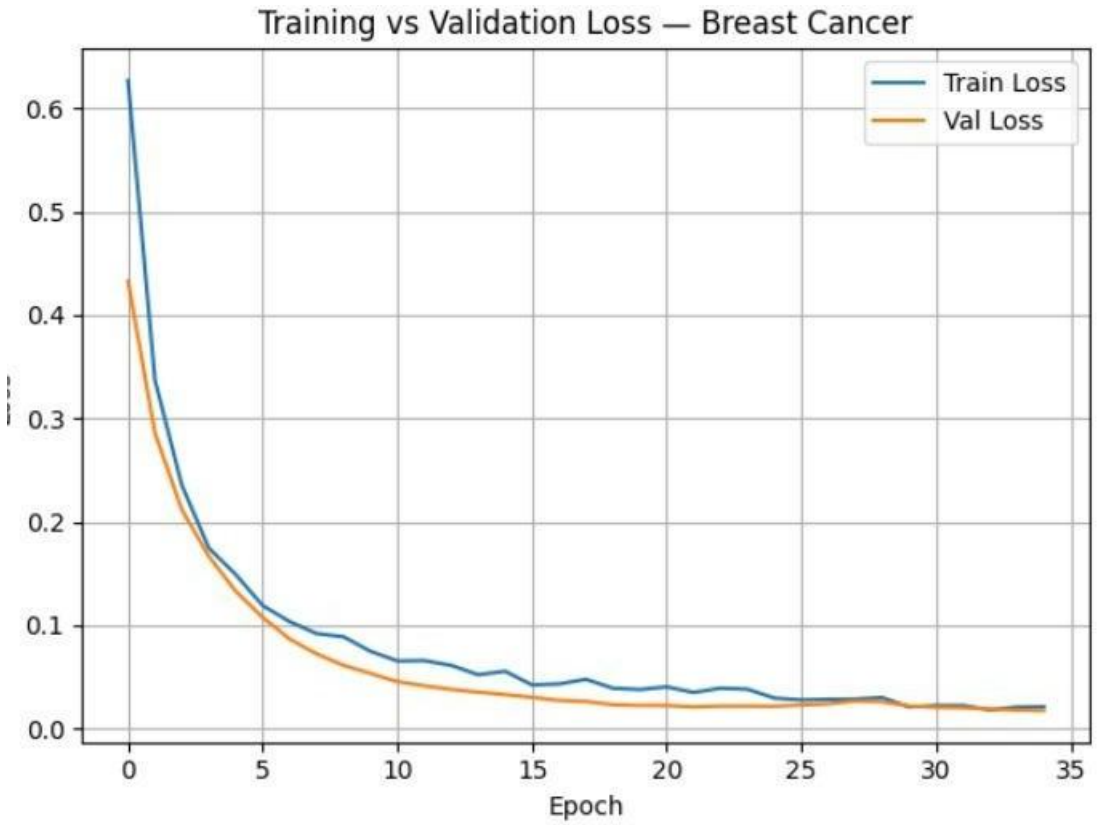
8.RESULT

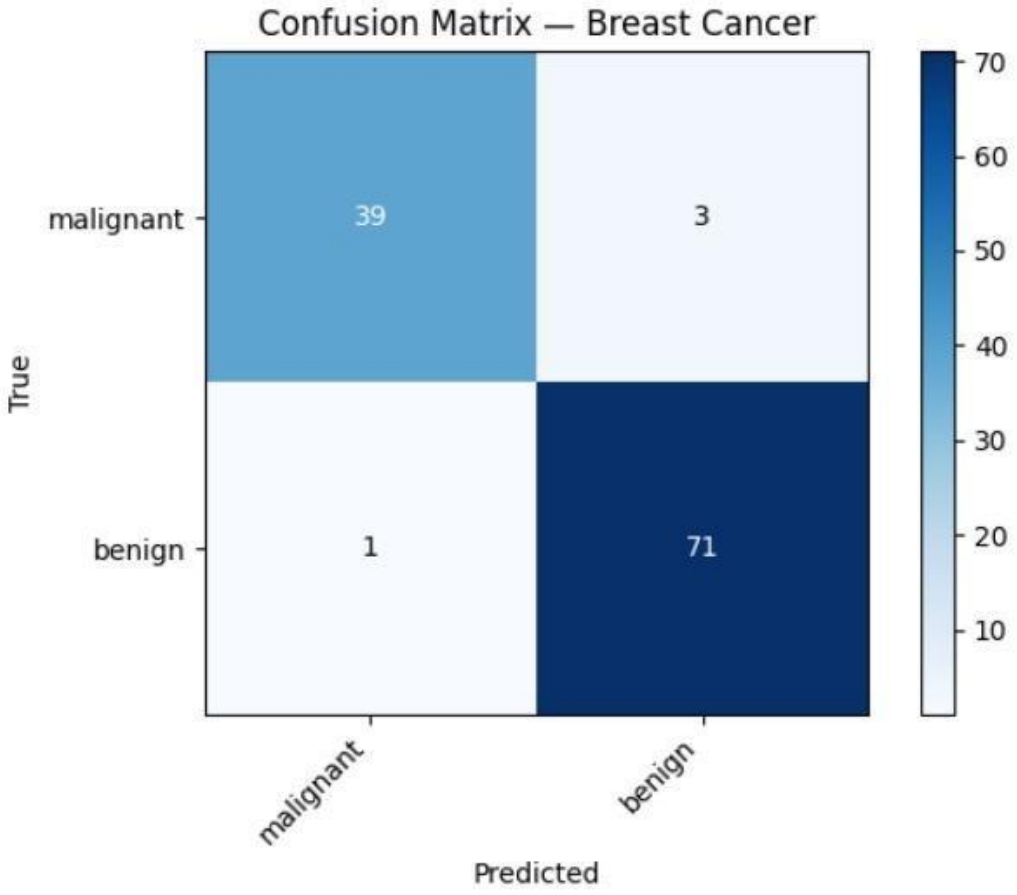
```
=== Dataset Preview ===
   mean radius  mean texture  mean perimeter    mean area  mean smoothness  \
0    17.990000     10.380000      122.800003  1001.000000          0.11840
1    20.570000     17.770000      132.899994  1326.000000          0.08474
2    19.690001     21.250000      130.000000  1203.000000          0.10960
3    11.420000     20.379999       77.580002   386.100006          0.14250
4    20.290001     14.340000      135.100006  1297.000000          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  ...  worst radius  worst texture  worst perimeter  \
0                 0.07871  ...     25.379999      17.330000       184.600006
1                 0.05667  ...     24.990000      23.410000       158.800003
2                 0.05999  ...     23.570000      25.530001       152.500000
3                 0.09744  ...     14.910000      26.500000        98.870003
4                 0.05883  ...     22.540001      16.670000       152.199997


    worst area  worst smoothness  worst compactness  worst concavity  \
0  2019.000000            0.1622             0.6656           0.7119
1  1956.000000            0.1238             0.1866           0.2416
2  1709.000000            0.1444             0.4245           0.4504
3   567.700012            0.2098             0.8663           0.6869
4  1575.000000            0.1374             0.2050           0.4000

   worst concave points  worst symmetry  worst fractal dimension
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902
2                0.2430          0.3613                  0.08758
3                0.2575          0.6638                  0.17300
4                0.1625          0.2364                  0.07678

[5 rows x 30 columns]

Shape: (569, 30)

Class Distribution (0=malignant, 1=benign):
1    357
0    212
Name: count, dtype: int64

=== FINAL TEST ACCURACY: 0.9649 ===
```

Classification Report:

	precision	recall	f1-score	support
malignant	0.9750	0.9286	0.9512	42
benign	0.9595	0.9861	0.9726	72
accuracy			0.9649	114
macro avg	0.9672	0.9573	0.9619	114
weighted avg	0.9652	0.9649	0.9647	114




Training vs Validation Loss — Breast Cancer
Train Loss
Val Loss
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
25
30
35
Epoch


Confusion Matrix — Breast Cancer
malignant
benign
True
39
3
1
71
70
60
50
40
30
20
10
malignant
benign
Predicted

9. CONCLUSION

The aim of this project was to develop an Artificial Neural Network (ANN) model capable of accurately predicting whether a breast tumor is malignant or benign using the Wisconsin Breast Cancer Dataset. Through systematic data preprocessing, feature scaling, careful model construction, and the application of regularization techniques such as Dropout and EarlyStopping, the ANN achieved strong predictive performance. The model was evaluated using multiple metrics—including accuracy, precision, recall, F1-score, ROC-AUC, and the confusion matrix—to ensure a comprehensive assessment of its effectiveness.

The results demonstrate that the ANN successfully learned the underlying patterns in the dataset and provided reliable classification outcomes, making it a valuable tool for supporting early detection of breast cancer. By providing fast, automated predictions, such systems can assist medical professionals in screening tasks and reduce diagnostic workload. While the model performs well, further improvements can be achieved through hyperparameter tuning, use of larger and more diverse datasets, integration of explainability tools such as SHAP, and comparison with advanced ensemble methods.

Overall, this project highlights the potential of neural network-based models in medical decision-support systems and reinforces how machine learning can contribute meaningfully to healthcare applications, especially in early diagnosis and risk prediction.

10. REFERENCES

[1] **Dua, D., & Graff, C.** (2019). *UCI Machine Learning Repository: Breast Cancer Wisconsin Dataset*. University of California, Irvine. Available at: https://archive.ics.uci.edu.

[2] **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., et al.** (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.

[3] **Chollet, F.** (2018). *Deep Learning with Python*. Manning Publications.

[4] **Goodfellow, I., Bengio, Y., & Courville, A.** (2016). *Deep Learning*. MIT Press.

[5] **Sokolova, M., & Lapalme, G.** (2009). *A Systematic Analysis of Performance Measures for Classification Tasks*. Information Processing & Management, 45(4), 427–437.

[6] **Wolberg, W. H., Street, W. N., & Mangasarian, O. L.** (1995). *Machine Learning Techniques to Diagnose Breast Cancer from Fine-Needle Aspirates.* Cancer Letters, 77(2–3), 163–171.

[7] **Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., & Fotiadis, D. I.** (2015). *Machine Learning Applications in Cancer Prognosis and Prediction.* Computational and Structural Biotechnology Journal, 13, 8–17.

[8] **Zhang, Z.** (2016). *Introduction to Machine Learning: kNN and Logistic Regression*. Annals of Translational Medicine, 4(12), 218.

[9] **Kingma, D. P., & Ba, J.** (2015). *Adam: A Method for Stochastic Optimization.* International Conference on Learning Representations (ICLR).

[10] **Shapley, L. S.** (1953). *A Value for n-Person Games.* Contributions to the Theory of Games. Princeton University Press. (Used as the foundation of SHAP explainability.)