Breast Cancer Diagnosis Using Neural Networks

Introduction

Breast cancer is one of the most prevalent and life-threatening diseases affecting women worldwide. Early and accurate diagnosis significantly improves treatment outcomes. In this project, we employ Neural Networks for automated breast cancer diagnosis using the Wisconsin Breast Cancer Dataset. Multiple models, including traditional and advanced neural architectures, are compared for classification, clustering, and associative memory tasks.

Dataset Description

- **Dataset Name**: Breast Cancer Wisconsin (Diagnostic) Data Set
- **Source**: Kaggle - UCI ML Repository
- **Features**: 30 real-valued input features computed from digitized images of fine needle aspirates (FNA) of breast masses
- **Target Variable**: Diagnosis (M = malignant, B = benign)
- **Total Records**: 569
- **No. of Features**: 30 (excluding ID and diagnosis label)

Preprocessing Techniques Used

Data preprocessing is critical to improve the quality of inputs to neural networks. Techniques employed include:

- Dropping irrelevant columns (e.g., ID)
- Label encoding target values
- Handling missing data
- Feature scaling

- Categorical encoding where applicable

Data Exploration

Initial data analysis included:

- Diagnosis distribution: ~62% benign, ~38% malignant
- Correlation heatmaps to identify multicollinearity
- Boxplots and pairplots for visual feature separation
- PCA for dimensionality reduction visualization

Handling Missing Values

The dataset is relatively clean, but any missing values were handled as follows:

- For numerical features: replaced with the median
- For categorical features: mode imputation

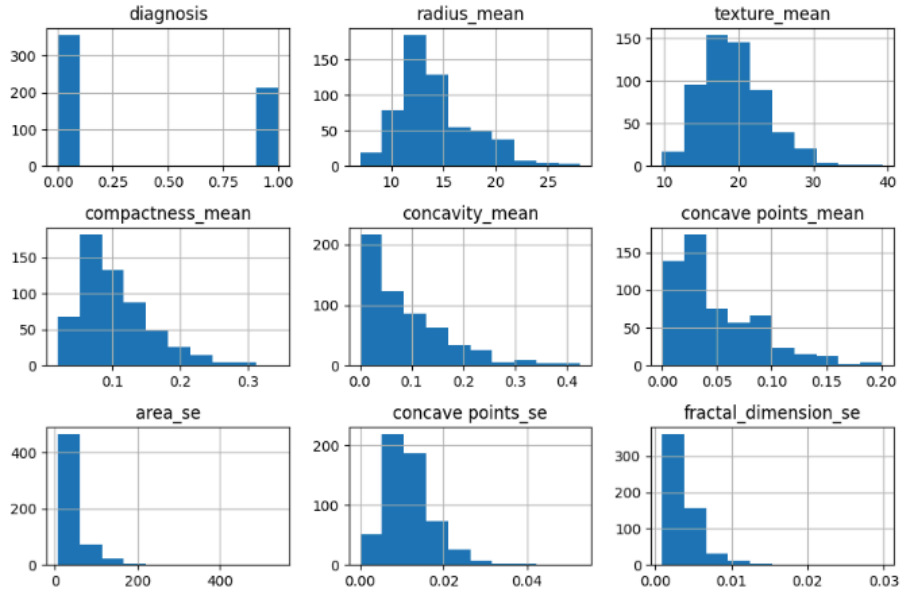
Feature Scaling

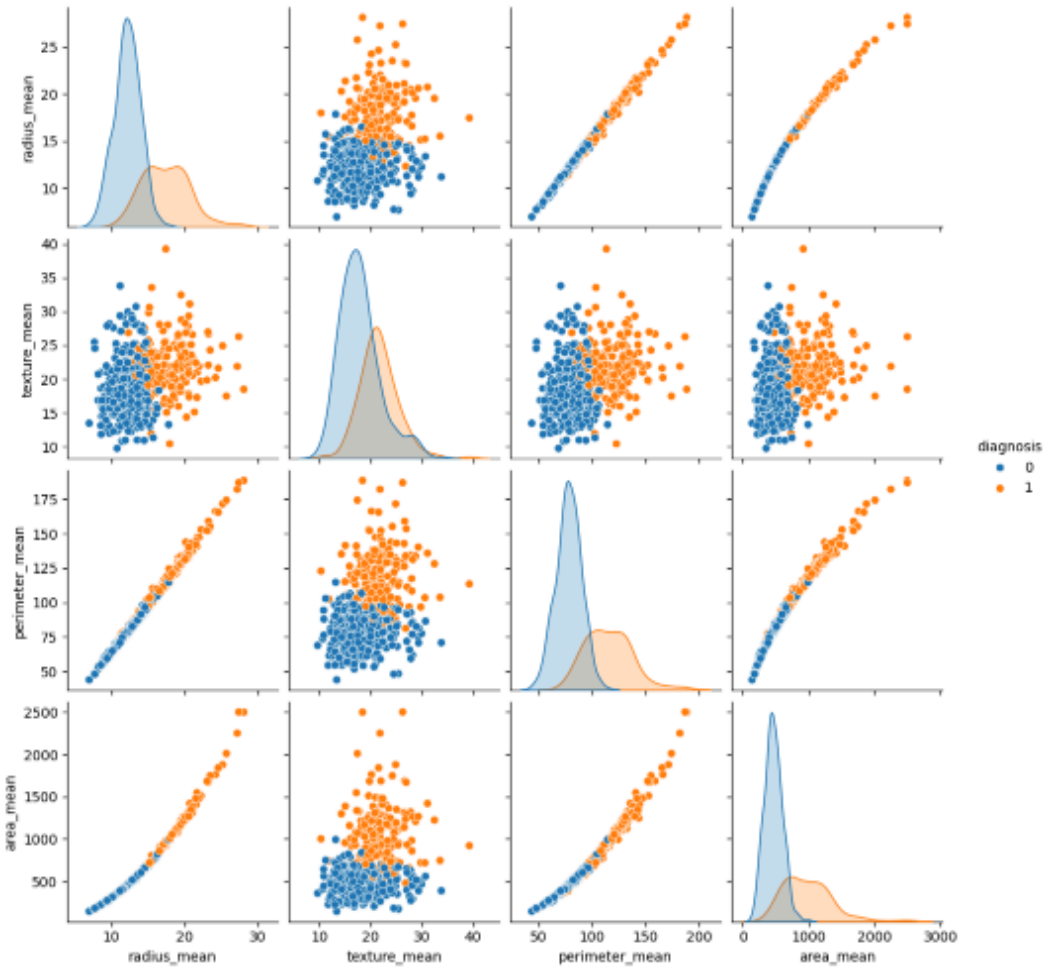
All input features were scaled using StandardScaler to zero-mean and unit-variance, ensuring consistency across neural network weight updates.

Encoding Categorical Variables

- The target column "diagnosis" was label encoded:
    - M → 1 (Malignant)
    - B → 0 (Benign)

Exploratory Data Analysis (EDA)





Model Architectures and Descriptions

1. MLP (Keras, Adam Optimizer)

The MLP model using Keras and the Adam optimizer is a deep feedforward artificial neural network composed of multiple layers: an input layer, one or more hidden layers, and an output layer. In this implementation, the model takes 30 input features derived from the dataset and passes them through two hidden layers with ReLU activations. The final output layer uses a sigmoid activation function to perform binary classification—malignant or benign. The Adam optimizer is chosen for its adaptive learning rate capabilities, making it suitable for faster convergence and improved generalization. Regularization techniques like dropout and L2 were applied to reduce overfitting. This model achieved the highest performance, showing its effectiveness in capturing complex nonlinear relationships in medical data.

- **Framework**: TensorFlow/Keras
- **Hidden Layers**: 2 hidden layers (128 and 64 neurons)
- **Activation**: ReLU, Sigmoid
- **Output**: 1 neuron, sigmoid
- **Loss**: Binary Crossentropy
- **Optimizer**: Adam
- **Epochs**: 50
- **Accuracy**: ~98.2%

2. MLP (Keras, SGD Optimizer)

This MLP model shares the same architecture as the Adam-based one but uses the **Stochastic Gradient Descent (SGD)** optimizer instead. Unlike Adam, which adapts learning rates, SGD uses a constant learning rate and can be sensitive to its initial setting. While it generally requires more epochs to converge and is slower than Adam, it can sometimes lead to better generalization in simpler models. The same regularization strategies (dropout, batch normalization, and L2 regularization) are applied here. The SGD model is helpful for learning optimization behavior and showcases the importance of choosing the right optimizer based on data characteristics.

- Identical architecture as Adam model but trained with Stochastic Gradient Descent

- Requires more epochs and careful tuning of learning rate
- **Accuracy:** ~95.6%

3. MLP (Scikit-learn)

The scikit-learn MLP model offers a high-level, user-friendly implementation of feedforward neural networks via the MLPClassifier class. Although not as flexible as Keras in defining complex architectures, it allows quick training and evaluation of models. The hidden layers and activation functions are similar to those used in Keras, and the optimizer is usually Adam by default. This model is useful for baseline comparisons and achieves impressive results with minimal tuning. It is especially suited for beginners or for conducting rapid prototyping on structured datasets like the breast cancer dataset.

- Used MLPClassifier from sklearn.neural_network
- **Parameters:** hidden_layer_sizes=(128, 64), solver='adam'
- Simpler interface, useful for quick comparisons
- **Accuracy**: ~98.2%

4. Perceptron

The Perceptron is one of the earliest and simplest forms of neural networks. It consists of a single layer that computes a linear combination of input features and applies a step function to classify inputs. While it is computationally efficient, it is inherently limited because it can only solve linearly separable problems. In the context of breast cancer diagnosis, where data features often have complex interactions, the Perceptron underperforms compared to multi-layer architectures. However, it serves as a valuable educational tool and a baseline for understanding the need for deeper neural networks.

- Single-layer linear model, used for baseline comparison
- Limited capacity, performs poorly on non-linearly separable data
- **Accuracy**: ~97.3%

5. BAM (Bidirectional Associative Memory)

BAM is a type of recurrent neural network used to store and retrieve associative data pairs. In this project, BAM is applied to associate binary-encoded symptom patterns with corresponding

disease classifications. Unlike traditional classifiers, BAM can perform bidirectional recall, either from symptoms to diseases or vice versa. This model is beneficial in medical expert systems where knowledge-based pattern association is required. While not suitable for real-valued, high-dimensional data like the full breast cancer dataset, it effectively demonstrates how binary relationships between features and outcomes can be modeled and recalled.

- **Use Case**: Associating binary symptom patterns with disease outcomes
- Requires binarized inputs and outputs
- Performs associative recall based on Hamming distance
- Not ideal for complex real-valued features but useful for demonstrating associative models

Regularization Techniques Applied

L1 Regularization (Lasso)

- Encourages sparsity, reduces overfitting by zeroing less useful weights
- Integrated via kernel_regularizer = regularizers.l1(0.01) in Keras

L2 Regularization (Ridge)

- Penalizes large weights, maintains stability
- More stable than L1 for most neural models

Dropout

- Applied between hidden layers to prevent overfitting
- Drop rate: 0.3 – 0.5

Batch Normalization

- Added after dense layers to stabilize training

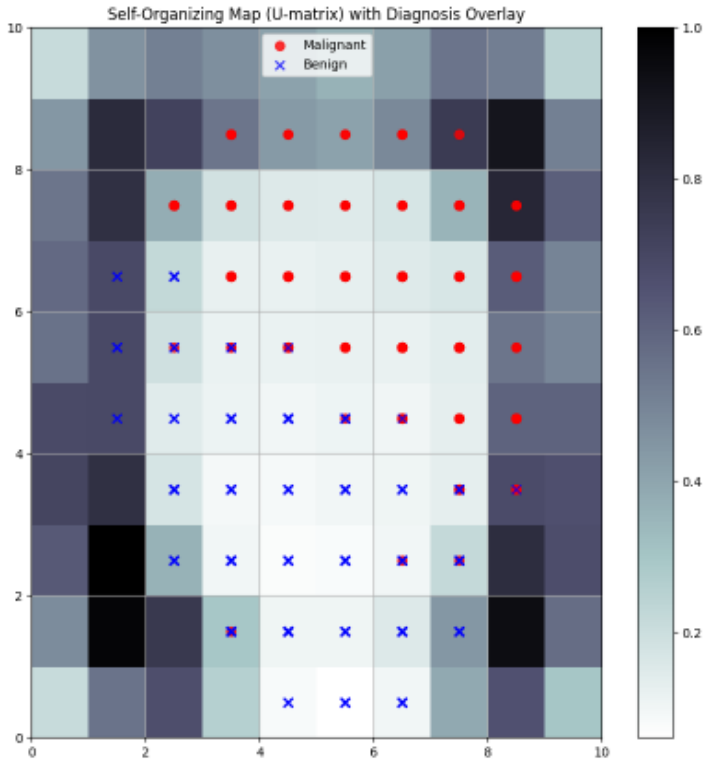
- Accelerates convergence and improves generalization

Advanced Topics

a. Self-Organizing Maps (SOM)

Self-Organizing Maps are an unsupervised learning method used primarily for clustering and visualization of high-dimensional data. In this application, SOMs are used to cluster patients into risk groups based on their feature profiles. The algorithm maps multidimensional data to a two-dimensional grid while preserving topological relationships. SOMs are particularly effective at visually identifying patterns in data, such as grouping malignant and benign cases, and discovering hidden subtypes in patient populations. This capability makes SOMs a powerful tool for exploratory data analysis and unsupervised pattern discovery in medical datasets.

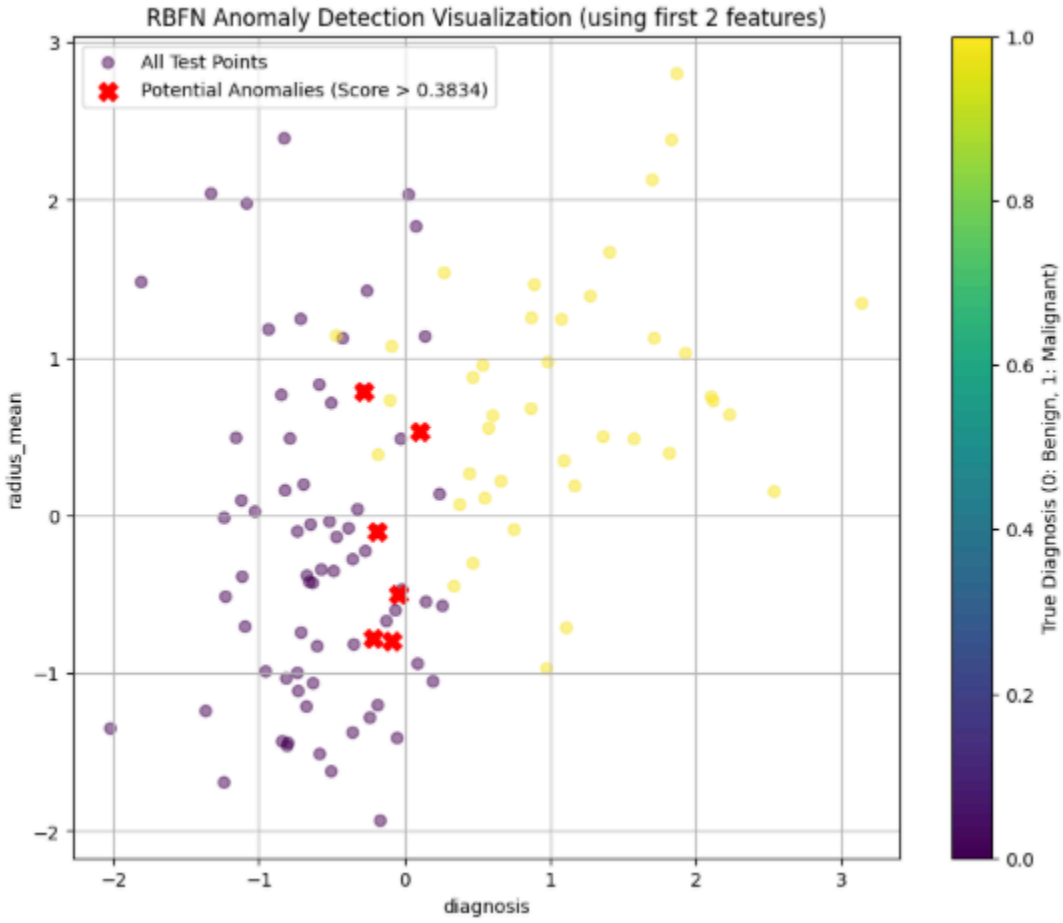
- **Purpose**: Clustering patients into risk groups based on feature similarity
- Implemented using MiniSom
- **Visual output:** 2D grid heatmaps showing clusters
- **Insight:** Clear separation between malignant and benign clusters



b. Radial Basis Function Network (RBFN)

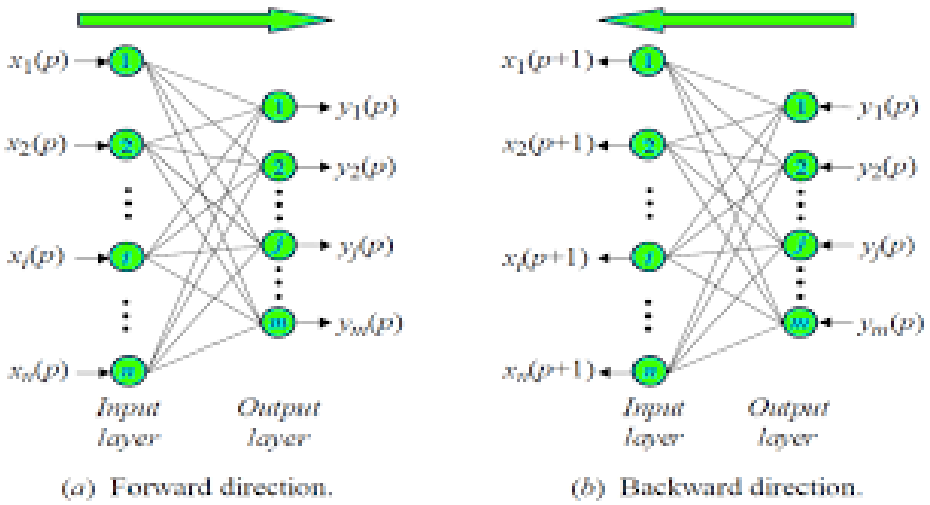
The RBFN is a three-layer neural network that uses radial basis functions (typically Gaussian functions) in its hidden layer to transform input data nonlinearly before passing it to the output layer. The RBFN is especially effective for tasks like anomaly detection, where the goal is to identify rare or unusual patterns—such as outlier cancer cases. It performs well in scenarios where class boundaries are not linearly separable and is robust to noisy data. In this project, RBFNs are used to flag potential anomalies within the diagnostic dataset, helping clinicians focus on unusual and potentially high-risk cases.

- **Purpose**: Detecting anomalous or rare cancer cases
- **Architecture:** Input → RBF layer (Gaussian) → Linear output layer
- Custom implementation using k-means to find centers
- Effective for unsupervised anomaly detection



c. Bidirectional Associative Memory (BAM)

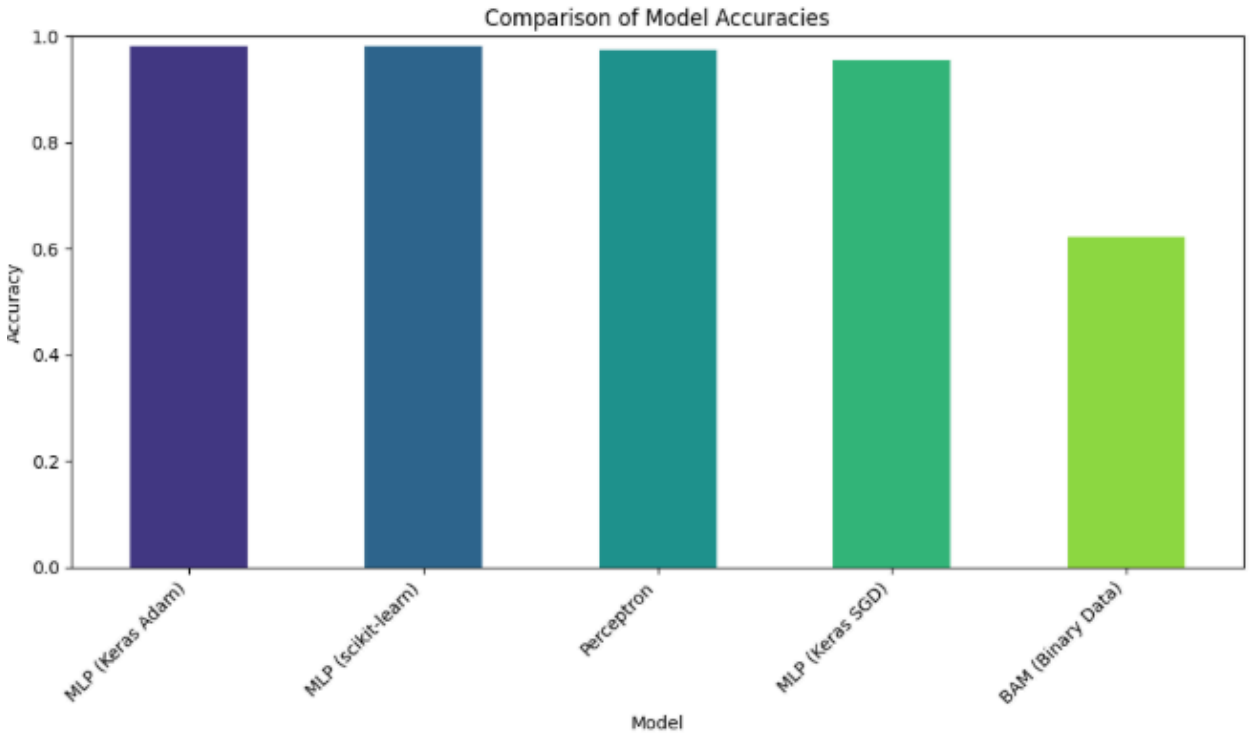
- Trained on binary-encoded symptoms-diseases pairs
- Bidirectional recall capability tested by altering inputs
- Demonstrates usefulness in pattern association tasks (e.g., diagnosis support)



Fig[1]: BAM structure

Comparison of Perceptron Models

Model	Accuracy	Convergence	Generalization	Flexibility
Perceptron	~97.3%	Fast	Low	Low
MLP (scikit-learn)	~98.2%	Moderate	Good	Medium
MLP (Keras Adam)	~98.2%	Fast	Excellent	High
MLP (Keras SGD)	~95.6%	Slow	Good	High
BAM (Binary Data)	~62.2%	Slow	Low	Low



Key Differences

- **Optimization**: Adam converges faster than SGD.
- **Frameworks**: Keras allows greater architectural flexibility than scikit-learn.
- **Model Depth**: MLPs outperform single-layer Perceptrons by learning non-linear patterns.
- **Advanced Models**: SOM and RBFN provide unsupervised insights, while BAM enables pattern association.

Results & Conclusion

- The best-performing model was **MLP (Keras, Adam)** with nearly **98.2%** accuracy.
- Advanced techniques like SOM provided additional clinical insight by grouping patients into meaningful clusters.

- BAM models were effective in simulating associative recall, useful for knowledge-based systems.
- Regularization and normalization greatly improved generalization and training stability.

Observations

- Most feature importance was attributed **to radius_mean, texture_mean, and perimeter_mean.**
- Feature scaling and dropout improved model robustness.
- SOM visualizations clearly demonstrated separability in feature space.

Potential Improvements

- Implement ensemble methods combining MLP, RBFN, and decision trees.
- Fine-tune hyperparameters using GridSearchCV or Bayesian optimization.
- Deploy the best-performing model as a web application or diagnostic tool.
- Include time-series analysis for progression tracking.

References

1. UCI Machine Learning Repository – Breast Cancer Wisconsin Dataset
2. Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow
3. Kohonen, T. (1990). The Self-Organizing Map
4. Haykin, S. Neural Networks: A Comprehensive Foundation
5. https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data
6. Fig[1]:https://www.geeksforgeeks.org/machine-learning/ann-bidirectional-associative-memory-bam/