WISCONSIN BREAST CANCER DATA SET

DATA ANALYSIS & CLASSIFICATION PROJECT

BY RAVAD NADAM

ABSTRACT

Breast cancer is a significant global health challenge where early and accurate diagnosis is critical for improving patient outcomes. While traditional diagnostic methods are effective, they can be subjective and time-consuming. Machine learning (ML) offers a promising, data-driven frontier for creating quantitative and objective diagnostic tools. By analyzing complex clinical datasets, ML models can identify patterns that may be imperceptible to human observers.

This report investigates the application of supervised machine learning to classify breast tumors using the Breast Cancer Wisconsin (Diagnostic) Data Set. This benchmark dataset contains 30 numerical features derived from fine-needle aspirate (FNA) images, each corresponding to a benign or malignant diagnosis.

The primary objective is to develop and evaluate various classification models—including Logistic Regression, Support Vector Machines, Ensemble Methods, and Neural Networks—to accurately differentiate between malignant and benign tumors. The study involves a systematic exploration of the dataset, application of data preprocessing techniques, and rigorous model evaluation using standard metrics such as accuracy, precision, recall, and AUC. Ultimately, this report aims to demonstrate the efficacy of machine learning as a powerful auxiliary tool for breast cancer diagnosis and to identify robust models for this critical task.

Contents

DATASET

The study utilizes the Breast Cancer Wisconsin (Diagnostic) Data Set, a publicly available collection of data originally created by Dr. William H. Wolberg, W. Nick Street, and Olvi L. Mangasarian of the University of Wisconsin. Sourced from the UCI Machine Learning Repository, this dataset is a standard benchmark for binary classification tasks in medical diagnostics.

The data was derived from digitized images of fine-needle aspirate (FNA) procedures on breast masses. The features in the dataset describe the characteristics of the cell nuclei present in these images.

Dataset Characteristics

The dataset consists of 569 instances, or observations, each corresponding to a unique sample. The raw data contains 32 attributes per instance. These attributes include a non-informative ID number, a diagnostic label, and 30 real-valued predictive features.

- Total Instances: 569
- Target Variable: Diagnosis
- Malignant: 212 cases
- Benign: 357 cases
- Predictive Features: 30 (numeric)
- Missing Values: None

For the purpose of machine learning, the ID column is discarded, and the diagnosis column serves as the binary target variable, where 'M' (Malignant) is typically encoded as 1 and 'B' (Benign) as 0.

Feature Description

The 30 predictive features are organized into three groups of ten, reflecting three different metrics calculated for ten fundamental cell nucleus characteristics.

Base Cell Nucleus Measurements:

Ten real-valued characteristics were computed for each cell nucleus identified in the FNA images:

1. Radius: The mean of distances from the center to points on the perimeter.
2. Texture: The standard deviation of gray-scale values.
3. Perimeter: The total length of the cell nucleus contour.
4. Area: The area enclosed by the cell nucleus contour.
5. Smoothness: A measure of the local variation in radius lengths.
6. Compactness: A measure of the shape's complexity, calculated as (perimeter^2 / area) - 1.0.
7. Concavity: The severity and number of concave (indented) portions of the contour.
8. Concave Points: The number of points on the contour that lie on a concave portion.
9. Symmetry: A measure of the shape's symmetry.
10. Fractal Dimension: A "coastline approximation" that provides a measure of the boundary's complexity.

Derived Feature Groups:

For each of the ten base measurements listed above, three metrics were calculated from the set of all cell nuclei in a given image, resulting in the 30 features used for classification:

- **Mean (_mean):** The average value for that characteristic across all nuclei in the image (e.g., radius_mean).
- **Standard Error (_se):** The standard error of the mean for that characteristic (e.g., texture_se).
- **Worst (_worst):** The mean of the three largest values for that characteristic found in the image (e.g., perimeter_worst).

This structure provides a comprehensive, multi-faceted description of the cell nuclei for each sample, forming a robust basis for training machine learning classifiers to predict the nature of the breast mass.

NO	Attribute	Data type	Range
Mean			
1	radius	Decimal	6.981 - 28.11
2	texture	Decimal	9.71 - 39.28
3	perimeter	Decimal	43.79 - 188.5
4	area	Decimal	143.5 - 2501.0
5	smoothness	Decimal	0.05263 - 0.1634
6	compactness	Decimal	0.01938 - 0.3454
7	concavity	Decimal	0.0 - 0.4268
8	concave points	Decimal	0.0 - 0.2012
9	symmetry	Decimal	0.106 - 0.304
10	fractal dimension	Decimal	0.04996 - 0.09744
Standard Error (se)			
11	radius	Decimal	0.1115 - 2.873
12	texture	Decimal	0.3602 - 4.885

13	perimeter	Decimal	0.757 - 21.98
14	area	Decimal	6.802 - 542.2
15	smoothness	Decimal	0.001713 - 0.03113
16	compactness	Decimal	0.002252 - 0.1354
17	concavity	Decimal	0.0 - 0.396
18	concave points	Decimal	0.0 - 0.05279
19	symmetry	Decimal	0.007882 - 0.07895
20	fractal dimension	Decimal	0.0008948 - 0.02984
Worst			
21	radius	Decimal	7.93 - 36.04
22	texture	Decimal	12.02 - 49.54
23	perimeter	Decimal	50.41 - 251.2
24	area	Decimal	185.2 - 4254.0
25	smoothness	Decimal	0.07117 - 0.2226
26	compactness	Decimal	0.02729 - 1.058
27	concavity	Decimal	0.0 - 1.252
28	concave points	Decimal	0.0 - 0.291
29	symmetry	Decimal	0.1565 - 0.6638
30	fractal dimension	Decimal	0.05504 - 0.2075
Non-Feature Columns			
31	id	Id	-
32	diagnosis	String	{M - B}

Table 1. Dataset Features

PREPROCESSING AND CLASSIFICATION

1. Preprocessing

Before a dataset can be used to train machine learning models, it must undergo a series of data preprocessing steps. This crucial phase ensures the data is in a clean, consistent, and optimal format for the learning algorithms, which can significantly enhance model performance and reliability. For the Breast Cancer Wisconsin (Diagnostic) Data Set, the following preprocessing steps were considered and applied.

1.1. Data Cleaning and Formatting

The initial step involved cleaning the raw data to make it suitable for analysis.

Removal of Non-Informative Columns: The id column, being a unique identifier for each patient, provides no predictive value and was therefore removed from the dataset. Similarly, an Unnamed: 32 column was present in the dataset, which contained no data and was also discarded.

Encoding the Target Variable: Machine learning algorithms require numerical inputs. The categorical diagnosis column, containing 'M' for malignant and 'B' for benign, was converted into a binary numerical format. Malignant cases were encoded as 1, and benign cases were encoded as 0. This process transforms the classification problem into a format that algorithms can mathematically process.

1.2. Feature Scaling

The predictive features in the dataset, such as radius_mean and area_mean, have vastly different scales and ranges. For instance, area_mean values are orders of magnitude larger than smoothness_mean values. Many machine learning

algorithms, particularly those that rely on distance calculations (e.g., Support Vector Machines, K-Nearest Neighbors) or gradient-based optimization (e.g., Logistic Regression, Neural Networks), are sensitive to the scale of the input features. Without scaling, features with larger ranges can disproportionately influence the model's training process.

To address this, Standardization (Standard Scaler) was applied. This technique rescales the features so that they have the properties of a standard normal distribution, with a mean of zero and a standard deviation of one. This ensures that each feature contributes equally to the model's learning process, preventing any single feature from dominating due to its scale.

1.3. Handling Outliers

Outliers are data points that deviate significantly from other observations in a dataset. They can arise from measurement errors, data entry mistakes, or genuinely rare events. These extreme values can skew the dataset's statistical properties (like mean and standard deviation) and can disproportionately influence the training of machine learning models, leading to a less accurate model that does not generalize well. Several methods were explored to identify and manage outliers:

Interquartile Range (IQR) Method*:* This method identifies outliers by measuring the statistical dispersion of the data. Any data point that falls below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR is considered an outlier and can be removed.

Z-Score: The Z-score measures how many standard deviations a data point is from the mean. A common threshold is to treat any point with a Z-score greater than 3 or less than -3 as an outlier.

Winsorization: Instead of removing outliers, this technique caps them. For example, all data points below the 5th percentile can be set to the 5th percentile value, and all points above the 95th percentile can be set to the 95th percentile value, thus reducing the effect of extreme values without data loss.

1.4. Handling Class Imbalance

The dataset exhibits a moderate class imbalance, with 357 benign samples (62.7%) and 212 malignant samples (37.3%). Training a model on an imbalanced dataset can lead to a biased classifier that performs well on the majority class (benign) but poorly on the minority class (malignant), which is often the class of greater interest. To mitigate this, techniques to balance the class distribution were explored:

SMOTE (Synthetic Minority Over-sampling Technique): This oversampling method creates new, synthetic examples of the minority class (malignant) in the feature space, thereby increasing its representation in the training data.

Random Undersampling: This method involves randomly removing samples from the majority class (benign) to achieve a more balanced distribution.

Applying these techniques can help the model learn the characteristics of both classes more effectively, leading to better predictive performance, especially for the critical task of identifying malignant tumors.

1.5. Dimensionality Reduction and Feature Selection

While the dataset has only 30 features, which is manageable, techniques for dimensionality reduction and feature selection were also considered as a means to potentially improve model performance and interpretability.

Principal Component Analysis (PCA): An unsupervised technique used to reduce the number of features by transforming them into a smaller set of uncorrelated variables called principal components, which capture most of the variance in the original data.

Recursive Feature Elimination (RFE): A supervised feature selection method that recursively removes the least important features and builds a model on the remaining ones until the desired number of features is reached. These preprocessing steps collectively ensure that the data is robust, well-conditioned, and optimized for the subsequent model training and evaluation phase.

In general, this data set is clean and doesn't require most of the preprocessing steps that we mentioned to produce good classification results.

2. Classification

Classification is a fundamental task in supervised machine learning where the objective is to predict a categorical class label for a given input data point. The process involves training a model on a dataset containing observations with known feature values and corresponding class labels. The trained model learns a function, or decision boundary, that separates the different classes in the feature space. Once trained, the model can be used to predict the class of new, unseen data.

In the context of the Breast Cancer Wisconsin (Diagnostic) Data Set, the task is a binary classification problem. The goal is to build a model that takes the 30 real-valued cytological features as input and assigns one of two possible labels as output: 1 for a malignant tumor or 0 for a benign tumor. This section provides an in-depth overview of the machine learning models selected for this classification task,

detailing their underlying principles, mathematical foundations, and suitability for this specific problem.

2.1 Logistic Regression

Despite its name, Logistic Regression is a linear model used for binary classification. It models the probability that a given input point belongs to a particular class.

The model first computes a linear combination of the input features, identical to a linear regression model:

$$z = w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n = w^t x$$

Where x_1, ..., x_n are the feature values and w_1,...,w_n are the model's learned weights or coefficients. The result, z, is then passed through the sigmoid (or logistic) function, which squashes the output to a value between 0 and 1:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

This output, σ(z), represents the estimated probability P(y=1|x) that the instance x belongs to the positive class (malignant). To make a final prediction, a decision threshold (typically 0.5) is applied. If $\sigma(z) \geq 0.5$, the sample is classified as malignant; otherwise, it is classified as benign. The model learns the optimal weights (w) by minimizing a log-loss (or binary cross-entropy) cost function, which penalizes confident and incorrect predictions more heavily.

Preprocessing Suitability for this Dataset:

Feature Scaling: Absolutely essential. The Wisconsin dataset contains features with vastly different scales (e.g., area_worst can be in the thousands, while smoothness_se is less than 0.1). Without scaling, the features with larger magnitudes would dominate the linear equation (z), causing the gradient descent optimization to converge slowly and potentially leading to a suboptimal model. Standardization is critical to ensure all features contribute fairly.

Outliers: The model can be sensitive to outliers. An extreme value in a feature like concavity_worst could significantly pull the linear decision boundary in its direction, potentially leading to misclassification of other points. Outlier handling is therefore recommended.

2.2 Support Vector Machine (SVM)

A Support Vector Machine is a powerful classifier that finds the optimal hyperplane that best separates the classes in the feature space. The "optimal" hyperplane is the one with the maximum margin—the largest possible distance between the hyperplane and the nearest data points from each class. These nearest points are called support vectors.

The objective is to find a hyperplane $w^T x - b = 0$ that maximizes the margin, which is equivalent to minimizing $\|w\|^2$. For non-linearly separable data, SVM employs the kernel trick. This technique avoids explicit mapping of data to a high-dimensional space by using a kernel function, K (x_i, x_j), which computes the dot product of the feature vectors in that higher-dimensional space. The Radial Basis Function (RBF) kernel is commonly used:

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right)$$

Here, γ is a hyperparameter that defines how much influence a single training example has. A small γ results in a smoother, more general decision boundary, while a large γ can lead to a more complex boundary that may overfit.

Preprocessing Suitability for this Dataset:

Feature Scaling: Mandatory and critical. The RBF kernel and the margin calculation are based on the Euclidean distance $\|x_i - x_j\|^2$ between data points. If features are not scaled, features like perimeter_mean would completely dominate the distance calculation over features like symmetry_mean, making the model effectively blind to the smaller-scale features.

Outliers: Fairly robust. The decision boundary is defined only by the support vectors. Therefore, outliers that are not support vectors have no influence. However, an extreme outlier could become a support vector and potentially skew the optimal hyperplane.

2.3 Decision Tree

A Decision Tree is a non-linear, interpretable model that partitions the data based on feature values. The tree structure consists of nodes that test features, branches that represent the outcomes of these tests, and leaf nodes that hold the final class predictions.

The algorithm iteratively selects the best feature and threshold to split the data. The "best" split is the one that maximizes the homogeneity of the resulting child nodes. This is measured using metrics like Gini Impurity or Information Gain (Entropy).

- **Gini Impurity:** Measures the frequency at which any element from the set would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the subset. For a node with classes c_1, ..., c_k and proportion p_i for class i: $Gini = 1 - \sum_{i=1}^{k}(p_i)^2$

  A Gini score of 0 indicates perfect purity (all samples belong to one class).
- **Information Gain:** Measures the reduction in entropy. Entropy is the measure of uncertainty or randomness in the data. $Entropy = -\sum_{i=1}^{k} pi \log_2(p_i)$

  The algorithm chooses the split that results in the highest information gain (i.e., the greatest reduction in entropy).

Preprocessing Suitability for this Dataset:

Feature Scaling: Not required. The model evaluates splits on one feature at a time (e.g., "is radius_mean < 15.0?"). The absolute scale of other features is irrelevant to this decision, making tree-based models immune to feature scaling issues.

Outliers: Decision Trees can be sensitive to outliers. A single extreme data point might cause the tree to create a specific split just to accommodate it. This can lead to unnecessary splits and overfitting, where the tree learns noise instead of the actual data pattern. As a result, the model may not generalize well to new data.

2.4 Ensemble Methods

Ensemble methods achieve better predictive performance by combining the predictions of multiple individual models (often called "weak learners").

2.4.1 Random Forests

Random Forest is an ensemble method based on the bagging (bootstrap aggregating) technique. It constructs a multitude of Decision Trees and merges their outputs.

- **Bootstrap Sampling:** Multiple random samples are drawn from the training set with replacement. Each sample is the same size as the original dataset.
- **Tree Building:** A Decision Tree is trained on each bootstrap sample.
- **Feature Randomness:** At each node of a tree, instead of searching over all features for the best split, the algorithm only considers a random subset of features. This two-tiered randomness helps to decorrelate the individual trees, reducing the variance of the final model. For classification, the final prediction is determined by a majority vote among all the trees in the forest.

Preprocessing Suitability for this Dataset:

As a tree-based method, feature scaling is not required. The ensemble nature makes it more robust to outliers than a single Decision Tree, as the influence of an outlier that affects one tree is averaged out over the entire forest.

2.4.2 Gradient Boosting, XGBoost, and LightGBM

Gradient Boosting is an ensemble technique that builds models in a sequential, stage-wise fashion. Unlike Random Forest, where trees are built in parallel, boosting methods build trees one after another, where each new tree is trained to correct the errors of the previous ones.

- An initial simple model (e.g., the mean of the target variable) is created.
- The algorithm calculates the errors (residuals) made by the current model.

- A new weak learner (typically a shallow decision tree) is fit to these residuals.
- The predictions from this new tree are added to the overall model's predictions, with a small learning rate to prevent overfitting.

This process is repeated until a stopping criterion is met. The model essentially performs gradient descent in the function space, where the "parameters" are the predictions themselves.

XGBoost (Extreme Gradient Boosting) and **LightGBM** (Light Gradient Boosting Machine) are highly optimized and efficient implementations of the gradient boosting framework, incorporating features like regularization, parallel processing, and advanced tree-growth strategies to achieve state-of-the-art performance.

Preprocessing Suitability for this Dataset:

Like other tree-based models, boosting algorithms do not require feature scaling. They are also generally robust to outliers.

2.5 K-Nearest Neighbors (KNN)

KNN is a simple, non-parametric, and instance-based (or "lazy") learning algorithm. It classifies a data point based on the majority class of its nearest neighbors.

- The entire training dataset is stored in memory.
- When a new, unclassified data point is presented, the algorithm calculates its distance to every point in the training set. The most common distance metric is the Euclidean distance:

$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$

It identifies the 'k' nearest neighbors (the 'k' points with the smallest distances).

- The new data point is assigned to the class that is most common among its 'k' neighbors.

Preprocessing Suitability for this Dataset:

Feature Scaling: Essential and critical. KNN's reliance on Euclidean distance makes it highly susceptible to feature scales. Without standardization, features with large ranges like area_mean would completely overwhelm features with small ranges like fractal_dimension_se in the distance calculation.

Dimensionality Reduction: Performance can degrade in high-dimensional spaces (the "curse of dimensionality"). For the 30 features in this dataset, PCA could be beneficial.

2.6 Naive Bayes

The Naive Bayes classifier is a probabilistic model based on Bayes' Theorem with a "naive" assumption of conditional independence among features. It calculates the posterior probability of a class (C_k) given a set of features ($x_1, ..., x_n$) using Bayes' Theorem:

$$P(C_k|x_1,\ldots,x_n) = \frac{P(C_k).P(x_1,\ldots,x_n|C_k)}{P(x_1,\ldots,x_n)}$$

Due to the "naive" independence assumption, $P(C_k|x_1,\ldots,x_n)$ is simplified to $\prod_{i=1}^{n} P(X_i|C_k)$. The model then predicts the class with the highest posterior probability. For the continuous features in this dataset, the Gaussian Naive Bayes variant is used, which assumes that the features for each class follow a Gaussian distribution.

Preprocessing Suitability for this Dataset:

Feature Scaling: Not required, as the algorithm uses statistical properties (mean, standard deviation) of each feature for probability calculations, not direct distance comparisons.

Data Distribution: While it assumes a Gaussian distribution for features, the model is known to be surprisingly effective even when this assumption is not perfectly met.

2.7 Deep Learning (Neural Network)

A Deep Learning model, or Artificial Neural Network, is a model inspired by the structure of the human brain. A Multi-Layer Perceptron (MLP), a type of feedforward neural network, is well-suited for this classification task. The MLP consists of an input layer (one neuron per feature), one or more hidden layers, and an output layer.

- **Forward Propagation:** Data flows from the input layer through the hidden layers. Each neuron receives weighted inputs from the previous layer, adds a bias, and passes the result through a non-linear activation function (e.g., ReLU - Rectified Linear Unit: $F(x) = \max(0, x)$).
- **Output Layer:** The final layer uses a sigmoid activation function to produce a probability between 0 and 1 for the binary classification.
- **Backpropagation:** The network "learns" by minimizing a loss function (e.g., binary cross-entropy). The error is propagated backward through the network, and an optimization algorithm like Adam adjusts the weights and biases to reduce this error.

Preprocessing Suitability for this Dataset:

Feature Scaling: Essential. Gradient-based optimization algorithms used in neural networks converge much more efficiently and reliably when features are standardized.

Outliers: Can be sensitive to outliers, which might lead to large weight updates during training and hinder convergence. Outlier handling is beneficial.

RESULTS AND OBSERVATIONS

1. Dataset

Highly Correlated Features:

radius_mean, perimeter_mean, and area_mean are extremely positively correlated with each other (close to 1.0). This makes sense geometrically, as larger radius → larger perimeter → larger area. Similar strong correlation patterns are seen among their _worst and _se counterparts, indicating redundancy.

Strong Predictors of Diagnosis (Malignant):

The following features show strong positive correlation with malignant diagnosis (diagnosis = 1): concavity_mean, concave points_mean, radius_worst, perimeter_worst, area_worst. These likely reflect how aggressive or irregular the tumor is. Features like smoothness_mean, symmetry_mean, and fractal_dimension_mean have weaker or negative correlations.

Negatively Correlated Features:

fractal_dimension_mean, symmetry_mean, and texture_mean tend to have lower or negative correlation with malignancy, possibly indicating benign traits.

Redundant Dimensions:

Several feature groups like radius_mean, radius_worst, and radius_se provide overlapping information. Dimensionality reduction or dropping highly correlated features can prevent model overfitting.

2. Classic Machine Learning Models

All model evaluations were conducted using a test-train split ratio of 0.85 for training and 0.15 for testing.

2.1 With No preprocessing

Using 3-fold cross-validation, the model that got the highest accuracy with stock parameters was **LightGBM** with accuracy of **0.975.** The training time is calculated of 3-fold cross-validation running on a local server on AMD Ryzen 5 5600 CPU with 16GB of DDR4 RAM

Model	Params	CV Score	Training Time (s)
LightGBM	{'enable_categorical': True, 'learning_rate': 0.1, 'n_estimators': 300, 'num_leaves': 20}	0.975155	0.08

XGBoost	{'learning_rate': 0.2, 'max_depth': 6, 'n_estimators': 100, 'xgb_colsample': 0.6, 'xgb_gamma': 0.0, 'xgb_subsample': 0.6}	0.964803	0.05
Logistic Regression	{'C': 100, 'max_iter': 1000, 'penalty': 'l1', 'solver': 'liblinear'}	0.962733	1.09
Random Forest	{'criterion': 'gini', 'max_depth': 10, 'n_estimators': 300}	0.960663	0.43
Gradient Boosting	{'learning_rate': 0.2, 'max_depth': 3, 'n_estimators': 300}	0.954451	1.09
SVM	{'C': 10, 'gamma': 'scale', 'kernel': 'linear'}	0.952381	11.28
Naive Bayes	{'var_smoothing': 1e-09}	0.937888	0.00
K-Nearest Neighbors	{'algorithm': 'auto', 'n_neighbors': 15, 'weights': 'uniform'}	0.923395	0.00

Table 2. Classification Results

2.2 With Preprocessing

	precision	recall	f1-score	support
Benign (0)	1.00	0.98	0.99	48

Malignant (1)	0.98	1.00	0.99	60
accuracy			0.99	108
macro avg	0.9	0.99	0.99	108
weighted avg	0.99	0.99	0.99	108

Table 3. Classification Results

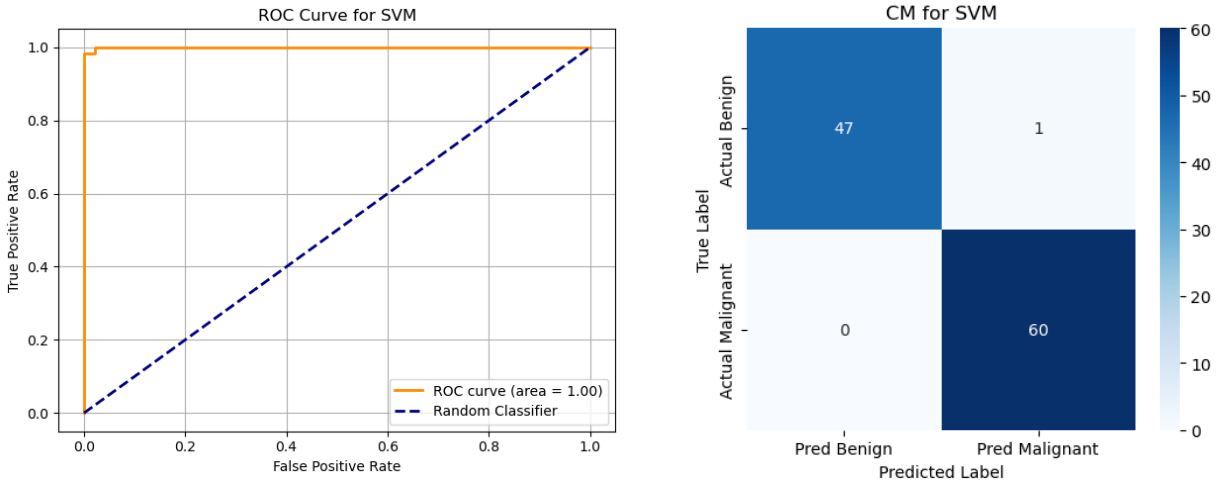

Figure 1. ROC Curve and Confusion Matrix

The highest classification performance with preprocessing techniques applied was achieved using a Support Vector Machine (SVM) model. With 3-fold cross-validation, the best accuracy reached was 0.9835. This performance was obtained using the following pipeline:

Preprocessing steps:

- **SMOTE:** Synthetic Minority Over-sampling Technique to address class imbalance

- **StandardScaler:** Feature standardization to normalize data distribution
- **Feature selection:** Removal of highly correlated features, particularly those involving redundant area values

Model configuration:

- **Algorithm:** Support Vector Machine (SVM)
- **Kernel:** Radial Basis Function (RBF)
- **Parameters:** C=10, gamma='scale'

3. Deep Learning Neural Network model

3.1 With No Preprocessing

	precision	recall	f1-score	support
Benign (0)	0.98	0.98	0.98	57
Malignant (1)	0.97	0.97	0.97	29
accuracy			0.98	86
macro avg	0.97	0.97	0.97	86
weighted avg	0.98	0.98	0.98	86

Table 4. Classification Results

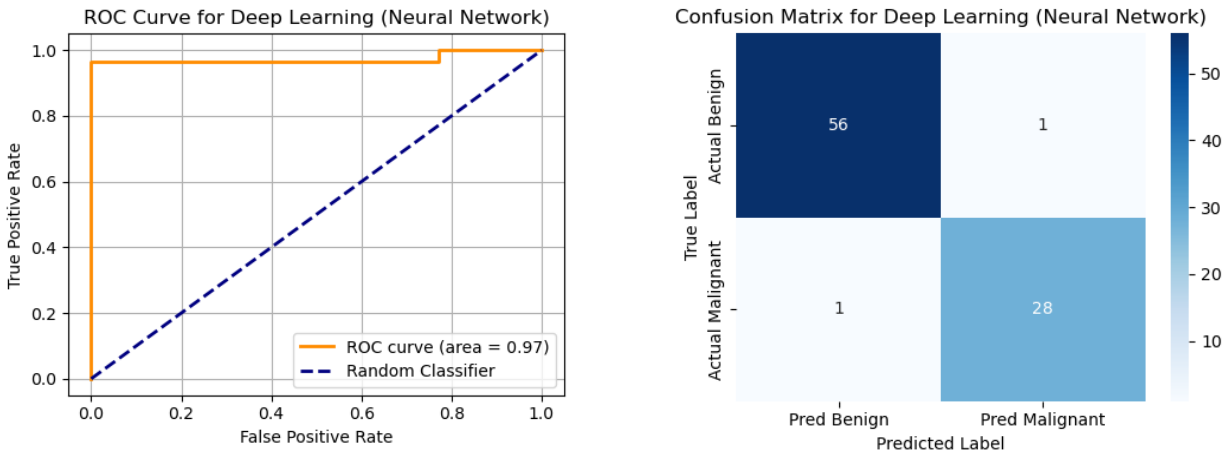

Figure 2. ROC Curve and Confusion Matrix

The model was trained with the following fixed parameters:

- optimizer: adam
- learning_rate: 0.001
- activation: relu
- dropout_rate: 0.2
- neurons_layer1: 64
- neurons_layer2: 32
- epochs: 100
- batch_size: 32

3.2 With Preprocessing

	precision	recall	f1-score	support
Benign (0)	1.00	0.96	0.98	51
Malignant (1)	0.97	1.00	0.98	57

accuracy			0.98	108
macro avg	0.98	0.98	0.98	108
weighted avg	0.98	0.98	0.98	108

Table 5. Classification Results

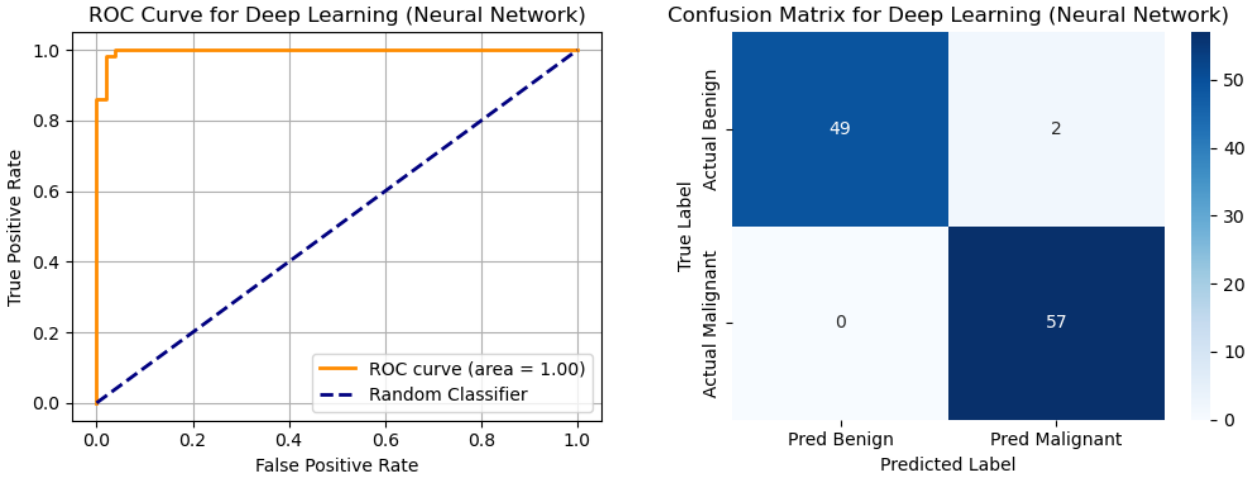

Figure 3. ROC Curve and Confusion Matrix

REFERENCE

Dua, D., & Graff, C. (2017). Breast Cancer Wisconsin (Diagnostic) Data Set. UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences.

https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic

UC Irvine Machine Learning Repository. (2016). Breast Cancer Wisconsin (Diagnostic) Data Set. Kaggle.

https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data/data

scikit-learn developers. (n.d.). Data preprocessing. scikit-learn: Machine Learning in Python.

https://scikit-learn.org/stable/data_transforms.html

scikit-learn developers. (n.d.). Supervised learning. scikit-learn: Machine Learning in Python.

https://scikit-learn.org/stable/supervised_learning.html

scikit-learn developers. (n.d.). Visualizations. scikit-learn: Machine Learning in Python.

https://scikit-learn.org/stable/visualizations.html

TensorFlow. (n.d.). TensorFlow Python API Documentation.

https://www.tensorflow.org/api_docs/python/tf

XGBoost contributors. (n.d.). XGBoost Documentation.

https://xgboost.readthedocs.io/en/stable/

Microsoft. (n.d.). LightGBM Python API Documentation.

https://lightgbm.readthedocs.io/en/stable/Python-API.html

Güneşer, C. (2009). Classification of Wisconsin Breast Cancer Database [Master's thesis, Dokuz Eylül University, Graduate School of Natural and Applied Sciences]. AVESIS.

https://avesis.deu.edu.tr/dosya?id=4389821b-c464-4b90-845c-b188eef89cbd