Science, Engineering and Technology Vol. 5, No. 1, pp. 177-191
www.setjournal.com *https://doi.org/10.54327/set2025/v5.i1.182*

A Comparison of Machine Learning Algorithms for Predicting Alzheimer's Disease Using Neuropsychological Data

Zakaria Mokadem[1], Mohamed Djerioui[1], Bilal Attallah[1], Youcef Brik[1]

[1]*LASS Laboratory, Faculty of Technology, University of M'sila, University Pole Road Bordj Bou Arreridj, M'sila 28000, Algeria.*

Abstract

Alzheimer's disease (AD) is a gradient degeneration of essential cognitive activities such as memory, thinking, and cognition. AD mainly affects elderly individuals and is recognized as the most common cause of dementia. This study investigates the predictive performance of nine supervised machine learning algorithms—Logistic Regression, Decision Tree, Random Forest, K-Nearest Neighbors, Support Vector Machine, Gaussian Naïve Bayes, Multi-Layer Perceptron, eXtreme Gradient Boost, and Gradient Boosting—using neuropsychological assessment data. We applied two classification techniques—binary and multiclass—to classify 1761 subjects into three categories: cognitively normal (CN), mild cognitive impairment (MCI), and Alzheimer's disease (AD). Binary classification tasks focused on CNvsAD and CNvsMCI subsets, while multiclass classification used the full dataset (TriClass). Hyperparameter tuning was performed to optimize model performance. The results indicate that ensemble learning models, particularly Gradient Boosting (GB) and Random Forest (RF), exhibited superior accuracy compared to other algorithms. Most models for the CNvsAD subset achieved the highest accuracy (97.74%), while GB achieved the best performance (94.98%) for the CNvsMCI subset. For multiclass classification, RF achieved the highest accuracy at 84.70%. These findings highlight the robustness and efficiency of ensemble learning algorithms, especially in handling complex, non-linear data structures. This study underscores the potential of RF and GB as reliable tools for early detection and classification of Alzheimer's disease using neuropsychological data.

Keywords: *Alzheimer's disease, Dementia, Machine learning, Classification, Neuropsychological assessment.*

1. Introduction

Alzheimer's disease (AD) is the most common cause of dementia and mainly affects people who are 60 years or older. There are three different stages of AD, early stage, middle stage, and late stage. In the early stage, patients begin to show simple and progressive loss of cognition and memory, with symptoms such as mild forgetfulness and problems with concentration. In the middle stage, as the disease progresses, patients experience trouble remembering events, learning new things, and planning complicated events. In the late stage, patients lose some physical abilities, such as walking, sitting, eating, and speaking [1], [2].

Several machine learning (ML) Algorithms in the medical field have been used to analyze medical data and identify patterns that can accurately predict diseases. ML can assist in solving diagnostic challenges across various medical domains and is extensively utilized for disease detection [3]. In this study, we compared the performance of nine different ML algorithms—including Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN), Support Vector

Corresponding author: *Zakaria Mokadem (zakaria.mokadem@univ-msila.dz)*
Received: *24 August 2024*; Revised: *11 December 2024*; Accepted: *16 December 2024*; Published: *23 December 2024*

Machine (SVM), Gaussian Naïve Bayes (GNB), Multi-Layer Perceptron (MLP), eXtreme Gradient Boost (XGBoost) and Gradient Boosting (GB)—on the classification of AD based on the results of five neuropsychological assessments: Geriatric Depression Scale (GDSCALE), Mini-Mental State Exam (MMSE), Global Clinical Dementia Rating (GCDR), Functional Activities Questionnaire (FAQ) and Neuropsychiatric Inventory Questionnaire (NPI-Q). These neuropsychological questionnaires are commonly used to assess different levels of cognition and they are frequently used for AD diagnosis.

This study aims to propose a robust machine learning model that accurately predicts AD, helping patients receive timely clinical intervention and avoid delays in diagnosis. Moreover, our study provides valuable insights into the use of ML algorithms as tools for diagnosing AD at a minimal cost. The following list includes some notable contributions:

- To propose methods for addressing the issue of misdiagnosis caused by variability in neuropsychological assessment results, which typically focus on specific brain functions.
- To compare the performance of various types of machine learning algorithms, including parametric, non-parametric, linear, nonlinear, and ensemble techniques, in predicting AD.
- To develop robust ML models capable of accurately predicting AD.

The rest of this paper is structured as follows: Section 2 provides a concise survey and a comprehensive overview of relevant existing research and highlights gaps in the literature. Section 3 presents the materials and methods, including the dataset description, an introduction to the ML algorithms used, proposes the framework, and explains the performance metrics utilized to evaluate the results. Section 4 presents the results including the optimal hyperparameters obtained and the outcomes of the performance comparison of the models. Section 5 discusses the results, while Section 6 highlights the key findings, provides essential conclusions, and suggests potential directions for future research.

2. Literature Review

Significant progress has been made in diagnosing AD using ML algorithms, especially in early detection and classification of the disease. This literature review aims to provide a comprehensive analysis of relevant research in this field, highlighting key findings and methodologies.

Several studies have focused on predicting AD using various data combinations, including brain imaging, blood and Cerebrospinal Fluid (CSF) biomarkers, genetic information, and neuropsychological assessments. For example, Wang et al. [4] proposed a multimodal method for AD prediction, based on 3D Magnetic Resonance Imaging (MRI), using support vector machines (SVM) classifier. Kruthika et al. [5] proposed a multistage classifier ML method using SVM, Naïve Bayes (NB), and K-Nearest Neighbors (KNN) algorithms, to classify different AD subjects. Al-Khuzaie et al. [6] adopted a deep learning (DL) technique to discriminate between AD and healthy patients, based on 2D medical images. Kaya et al. [7] presented an optimized Convolutional Neural Networks (CNNs) algorithm for detecting early AD stages from the MRI images using Particle Swarm Optimization (PSO). Doaa Ahmed Arafa et al [8] proposed a CNN algorithm for AD classification using the MRI images collected by the Kaggle Dataset. The study evaluated two methods: a simple CNN architecture and a fine-tuned VGG16. Venugopalan et al. [9] suggested the use of 3D-CNNs for the classification of AD patients, based on genetic analysis of Single Nucleotide Polymorphisms (SNPs), clinical data, and MRI images. Castillo-Barnes et al. [10] presented a comparison between different genes of AD subjects using the ANalysis Of VAriance algorithm (ANOVA) followed by Principal Component Analysis (PCA) for feature selection, and SVM for AD classification. Jo et al. [11] developed a DL model for AD classification using Tau Positron Emission Tomography (Tau PET) scans. The model combines 3D-CNN and Layer-wise Relevance Propagation (LRP) algorithms to extract informative features from Tau PET images. Cui et al. [12] proposed a framework that involves longitudinal analysis of consecutive MRI scans in parallel with clinical neuropsychological assessments to compute the progression of the disease over time. Rohini and Surendran [13] proposed a group of supervised ML methods such as Multivariate Linear Regression (MLR), Logistic Regression (LR), and SVM that are applied to

baseline neuropsychological assessments, genetic data, and MRI images. Khan and Zubair [14] proposed an automated classification system based on Random Forest (RF) to classify AD patients into five distinct stages using MRI images and neuropsychological assessments. Almubark et al. [15] compared the performance of various machine learning algorithms, including Fully Connected Multi-Layer Perceptron (MLP), Support Vector Machine (SVM), Random Forest (RF), Gradient Boosting (GB), and AdaBoost (AB) classifiers, using behavioral data derived from neuropsychological assessments and/or cognitive tasks. Duc et al. [16] developed a 3D-CNN model for AD diagnosis using a neuropsychological assessment and functional MRI images. Khan et al. [17] developed a hybrid ML model to classify AD using the ADNI dataset. In the first experiment, eXtreme Gradient Boost (XGBoost), RF, and SVM classifiers were applied, while in the second experiment, only the RF classifier was used. Then in the final experiment, a hybrid model, combining all classifiers. Seshadri et al. [18] developed a multitask multimodal DL model using BiLSTM coupled to multiple hidden dense layers to predict AD based on two neuropsychological assessments.

Most recent studies analyzing data related to AD classification and prediction primarily focus on neuroimaging data. However, collecting such data is not feasible for all Alzheimer's patients due to its high cost and limited accessibility, particularly in certain regions. As a result, studies using only these data are often restricted to patients residing in major cities or those with access to advanced medical imaging and analysis facilities, which can introduce significant bias into ML models and favor these particular patient groups, potentially impacting the prediction performance of the underrepresented populations. Moreover, Alzheimer's patients commonly exhibit notable behavioral disturbances and psychological symptoms, which are key features that can assist in diagnosing AD [19]. Nonetheless, many studies have not thoroughly addressed this aspect, with only a few incorporating, at most, one or two neuropsychological assessments.

Our study seeks to address these gaps by simultaneously analyzing five different neuropsychological assessments, which can be collected using low-cost, first-line diagnostic tools, and using them to compare the performance of nine supervised ML algorithms in AD prediction.

3. Materials and Methods

In this section, we describe the procedures and tools used in this work. The dataset is thoroughly detailed, the methods—including nine machine learning algorithms—are explained, and the performance metrics are presented.

3.1. Dataset description

The neuropsychological data used in this study is acquired from the Alzheimer's Disease Neuroimaging Initiative database ADNI [20]. The main objective of ADNI has been to evaluate whether the integration of MRI, PET, biological markers, and neuropsychological assessments can effectively measure the development of the early stage of AD [21]. ADNI provides unlimited access to the database and encourages researchers to use its data to understand the development of AD.

In this study, the data collected during Phase 1 of ADNI (ADNI-1) were used. The initial dataset contained numerous empty and redundant fields that could affect the classification performance. The dataset was cleaned by removing the non-essential and empty columns. The classes, which were represented by non-numerical values, were transformed into numerical ones to ensure consistency in data processing. The final dataset consisted of 1761 samples, categorized into three groups: 616 Cognitively Normal (CN), 878 Mild Cognitive Impairment (MCI), and 267 Alzheimer's disease (AD). The dataset included five neuropsychological assessments —GDSCALE, MMSE, GCDR, FAQ, and NPI-Q— collected at different time intervals. The age and gender information are included in the dataset. Figure 1 illustrates the gender distribution of the subjects, while Table 1 presents the demographic information.

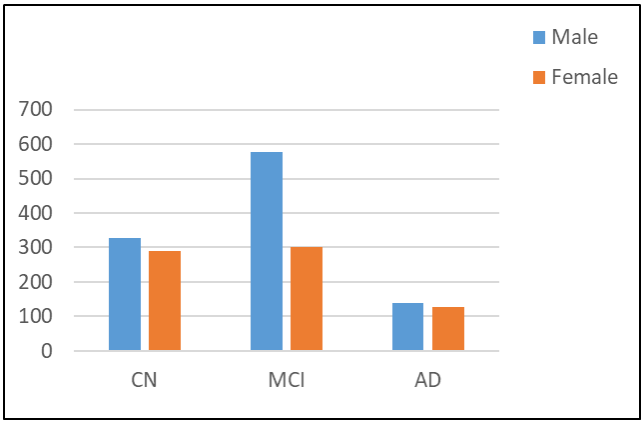

Figure 1. Gender distribution of the subjects.

Table 1. Demographic information of the subjects by clinical group, gender, and age: Mean ± Std.

Clinical Group	Male	Female	Age (Mean ± Std)
CN	326	290	77.96 ± 6.8
MCI	578	300	76.72 ± 7.0
AD	140	127	77.07 ± 7.6

Table 2 provides comprehensive information about the five neuropsychological assessments used, including their purposes and scoring methods.

Table 2. Overview of neuropsychological assessments used.

Neuropsychological Assessment	Description
GDSCALE	It was used to evaluate the symptoms of depression in elderly patients based on 15 questions.
MMSE	It is an assessment based on a questionnaire used to test cognitive decline. The scale ranges from 0 to 30.
GCDR	It is designed to measure the advancement of Dementia and the progression of dementia, particularly in patients with MCI. The scale for this assessment ranges from 0 to 18.
FAQ	It is used to assess the ability of the patient to perform the usual daily activities independently. The scale ranges from 0 to 30.
NPI-Q	It assesses a range of neuropsychiatric symptoms, including agitation, aggression, anxiety, apathy, depression, disinhibition, irritability, and sleep disturbance. The scale ranges from 0 to 12.

3.2. Machine learning algorithms

3.2.1. Logistic Regression classifier

LR is a linear discriminative method used to solve classification and regression problems. The LR classifier achieves good accuracy in linearly separable classes. LR classification model is a statistical technique for binary classification that can be extended to multiclass classification. It can classify new samples by measuring the probability of belonging to each class and selecting the class that has the highest probability [22], [23]. The expression of the predictor in the LR model is:

$$P(y_i|x_i) = sigmoid(x_i\theta + \theta_0) = \frac{1}{1+\exp(x_i\theta+\theta_0)} \quad (1)$$

Where θ and θ_0 represent weight's matrix and Bais' matrix, respectively.

3.2.2. Decision Tree classifier

DT is a hierarchical supervised ML algorithm used for classification and regression tasks. It consists of decision rules that are applied to partition the dataset feature space into subspecies of one class. Building a DT classifier is based on recursively partitioning the feature space of the training set to obtain an optimal set of decision rules that provide the best classification. The node at which the tree begins in a DT is called the" root node", and the node at which the tree ends is called the" leaf node". An internal node typically branches off into two directions or more, every non-terminal node (i.e., not-leaf node) represents a test node for features, and each branch represents an outcome of the test [24].

Feature selection measures are used to select the best partitions that distinct classes. The entropy (E) and information gain (IG) are applied to measure each node in the DT. The feature with the greatest Information Gain or the lowest Entropy is selected as the most suitable option for splitting the data at a specific node. E and IG are both metrics that quantify the amount of data associated with the prediction of the class. The entropy formula is as follows:

$$E = -\sum_{i=1}^{n} p_i \log(p_i) \quad (2)$$

Where the P_i are the ratios of elements of each class.

The information gain formula is as follows:

$$IG = E_{parent} - Average(E_{children}) \quad (3)$$

3.2.3. Random Forest classifier

RF is an ensemble learning algorithm that uses multiple DTs and combines them to improve prediction performance. RF classifier constructs each DT in the ensemble using a bootstrapped sample of the training dataset, where each sample has an equal chance in selection. Additionally, the ensembles are constructed using a sample of data randomly selected from the training dataset with replacement every time. Each tree in the ensemble acts as an autonomous classifier to determine the class label of an unlabeled instance using an aggregating technique called majority voting, where each

classifier casts one vote for its predicted class label. Then the class label with the most votes is used to classify the unlabeled instance [25], [26].

3.2.4. K-Nearest Neighbors classifier

KNN is a cluster analysis technique. However, it is considered the most important and popular classifier. A KNN classifier works by identifying the *k* closest samples in the training dataset and using them to determine the label of new data based on the majority label among these nearest neighbors [27]. Where *k* is the number of neighbors based on the similarity in the distance.

The optimal value of *k* chosen is crucial because the small value of *k* can add noise to the classification process which has an important impact on the classification accuracy, and the big value of *k* makes computing operations more complex and consumes more time [28]. The Euclidean distance formula is commonly used to calculate the distance between samples [29]. This formula is defined as follows:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (4)$$

3.2.5. Support Vector Machine classifier

SVM classifier is a supervised learning technique that aims to create a hyperplane between two classes in a dataset. The hyperplane is positioned in a way that maximizes the distance between the closest labeled instances from each of the classes which permits the classification of new unlabeled instances (i.e., feature vectors) into one of the classes. These closest labeled instances are known as support vectors. The optimal hyperplane of the linear kernel can be defined as:

$$y = \omega x^t + b \quad (5)$$

Where ω is the weight vector, x is the input feature vector, and b is the bias [30].

The values of ω and b must satisfy the following inequalities for all elements of the training set:

$$\begin{cases} \omega x_i^t + b \leq 1\,, \ if\ y_i = 1 \\ \omega x_i^t + b \geq 1\,, \ if\ y_i = -1 \end{cases} \quad (6)$$

Where x_i represent the feature vectors representations and y_i are the classes labels of a training compound *i*.

The goal of training an SVM model is to find the values of ω and b that make the hyperplane separate the data and maximize margin $1/\|\omega\|^2$. A kernel function can be applied to a nonlinear classification problem to transform the original input feature space into a higher dimensional space. This technique helps to linearly separate the data points by a linear classifier model, which could help to do a fast computation in high dimensional space. The most common kernels are Polynomial, Gaussian, and Radial Basis Function kernels (RBF).

3.2.6. Gaussian Naïve Bayes classifier

GBN classifier is a probabilistic classifier based on Bayes' theorem. In GNB, each feature in each class is considered an independent variable, and the classification probability for each variable is calculated according to a Gaussian (i.e., Normal) distribution [30]. The classification label is derived from the input using the posterior probability of Bayes' rule. The feature distribution can be defined as the probability of an event *X* occurring given the probability of another event that has already occurred *y*, and it can be expressed as:

$$P(y|X_i) = \frac{P(y).P(X_i|y)}{P(X_i)} \quad (7)$$

Where $P(X)$ and $P(y)$ are distributions of the X and y events respectively.

The mean and standard deviation are computed for the feature set of each class, and this information is used to estimate the likelihood of the features. The Likelihood is given by:

$$P(X_i|y = \hat{y}) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left(-\frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2\right)} \quad (8)$$

Where y, X_i, σ, and μ represent the class, observations (i.e., features), standard deviation for the selected class, and mean of the class, respectively.

3.2.7. Multi-Layer Perceptron classifier

MLP classifier is a type of Artificial Neural Network (ANN). The MLP model is composed of three layers, which are the input layer, the hidden layer, and the output layer [31]. The MLP model processes signals by sequentially computing the output of each layer from the inputs of the preceding layer, propagating the output

signal from the input layer to the output layer. During the training process of an MLP model, the error signal — which is the difference between the desired and actual outputs— is propagated backward from the output layer to the inner layers. This backward propagation of the error signal helps optimizes the accuracy of the model by repeatedly adjusting the weights and biases until the desired output is achieved. The equation below describes the relationship between the inputs and the output of a neuron:

$$y = \sum_{i=1}^{N} \omega_i x_i + b \quad (9)$$

Where y, x_i, ω_i, and b represent the output, inputs, weights, and bias, respectively.

3.2.8. eXtreme Gradient Boost classifier

XGBoost classifier is an ensemble classifier that uses an iterative process to enhance performance. With this approach, each successive model is trained to correct the errors made by previous models, repeating until further improvement is no longer achievable. This strategy prevents repeated errors across models, strengthening the overall model's accuracy. XGBoost classifier is designed to optimize a specific objective function, which drives it toward more accurate predictions by combining loss and regularization terms. Regularization controls model complexity, reducing overfitting and variance, which helps the model generalize better and results in stable, reliable predictions.

3.2.9. Gradient Boosting Classifier

GB classifier follows a stepwise additive model, where weak learners (the Decision Trees models in our case) are added sequentially to the ensemble. Each new learner is trained to correct the errors made by the combined previous models, incrementally improving predictions. The GB algorithm uses gradient descent optimization, adjusting each learner based on the gradient (slope) of the loss function to reduce prediction errors. In this process, each weak learner contributes to minimizing the loss function, which measures the difference between actual and predicted values. By iteratively reducing the residual errors left by previous learners, the ensemble becomes increasingly accurate, as each model minimizes the error across all predictions. However, the iterative nature of adding and adjusting each learner makes this approach computationally expensive.

3.3. Proposed framework

The nine classifiers (LR, DT, RF, KNN, SVM, GNB, MLP, XGB, and CB) were built by using the models and tools available in scikit-learn based on two steps:

- The first step involves training each model individually on each dataset using different hyperparameter configurations to identify the appropriate and optimal hyperparameters. The optimal hyperparameters were selected by the GridSearchCV tuner from the scikit-learn library with 5-fold cross-validation. GridSearchCV uses a predefined set of hyperparameters of the models and exhaustively searches through all possible combinations to find the best combination that produces the highest performance accuracy for each model.
- The second step involves retraining the models by using the optimal hyperparameters obtained from the first step to confirm the performance results of each algorithm. Finally, the results are compared to select the most robust classifier that achieves the best performance.

These two steps were repeated for both modes of classification: binary and multiclass. For the binary classification task, the final dataset was divided into two subsets: CNvsAD and CNvsMCI, while the full dataset, named TriClass, was used for the multiclass classification task.

For each classification mode, 80% of each subset was used to train the models, and the remaining 20% was reserved to test the performance of each model.

Figure 2 illustrates the block diagram of the proposed framework, its main components, and their interactions.

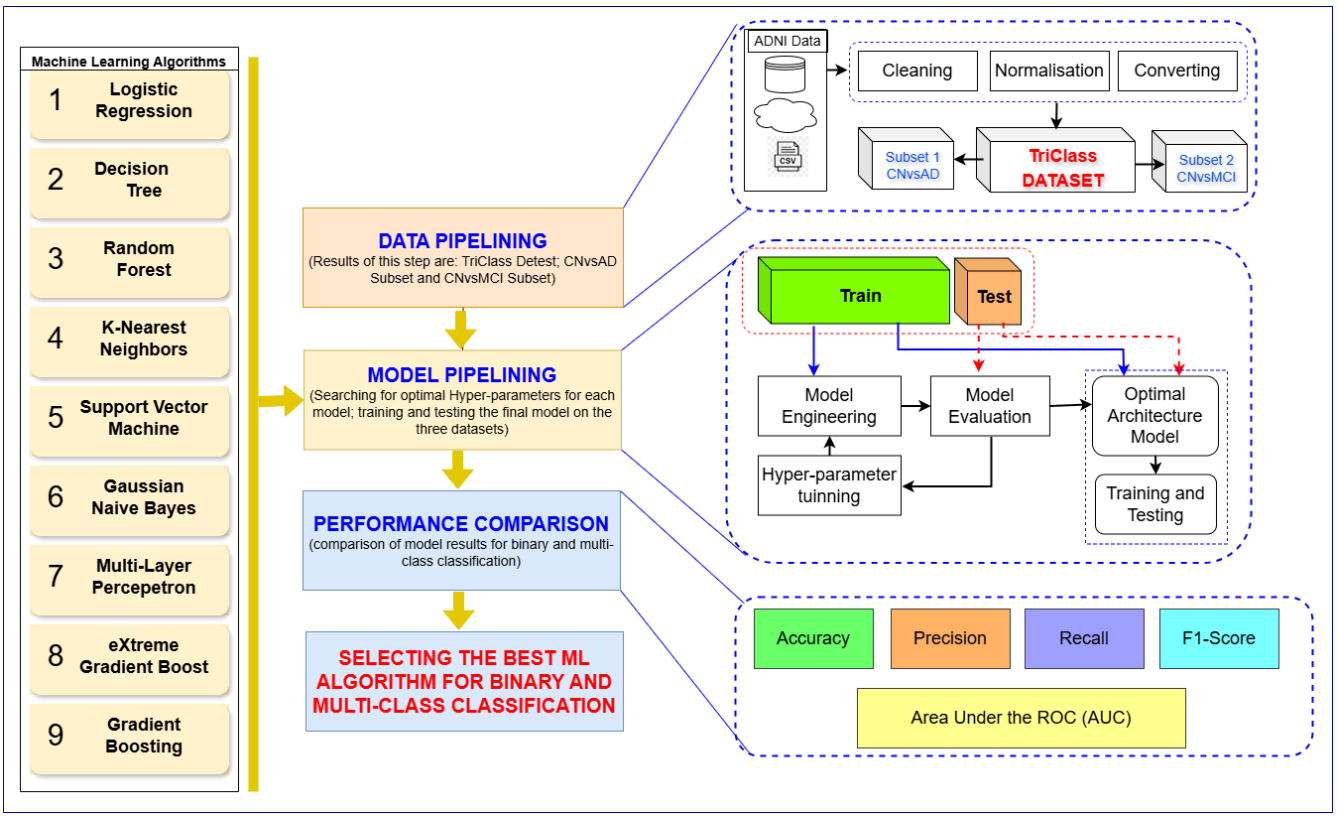

Figure 2. Detailed block diagram illustrating the structure of the proposed framework.

3.3.1. Hyper-parameter Optimization

To determine the optimal hyperparameters for the training process we used predefined ranges and trained each model with all possible combinations. The accuracy obtained from training the ML models individually was compared and the hyperparameters that yielded the highest accuracy were selected as the optimal ones. This process was performed for the subsets (CNvsAD and CNvsMCI) for binary classification, and the full dataset (TriClass) used for multiclass classification. The hyperparameter configuration ranges of the models are:

- For the LR model, L1 and L2 Regularization, C parameter, and Solver were tuned.
- For the DT model, we tuned the cost function by utilizing either the Gini index or entropy. Moreover, we adjusted Max_depth, max_features, and max_leaf_nodes.
- For the RF model, the Max_depth, max_features, max_leaf_nodes, and the cost function were tuned.
- For the KNN model, the n_neighbors and distance metrics were adjusted using either Euclidean, Manhattan, or Minkowski metric.
- For the SVM model, we used a different kernel and tuned the value of Gamma and C parameters.
- For the GNB model, we adjusted only Var_smoothing.
- For MLP model, the Hyper-parameters, including the dimensions of the hidden layer, solver, and activation function were tuned.
- For XGBoost model, Max_depth, n_estimators, and the learning_rate hyperparameters were tuned.
- For GB model, we tuned the Max_depth, n_estimators, max_leaf_nodes, and the learning_rate.

3.4. Performance metrics

The performance metrics were calculated based on the values of the confusion matrix, which is considered very important for evaluating classification models. The confusion matrix allows to identify the values of True Positives (TP) when the classification model correctly predicts a true target value, False Negatives (FN) when the classification model incorrectly predicts a false target value, True Negatives (TN) when the classification model correctly predicts a false target value, and False Positives

(FP) when the classification model incorrectly predicts a true target value. By using these values from the confusion matrix, the following metrics were calculated for each proposed model:

- **Accuracy:** The accuracy is the ratio of the sum of true cases and the total number of all the cases.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

- **Precision:** The precision is the proportion of the positive cases that were predicted correctly.

$$Precision = \frac{TP}{TP + FP} \quad (11)$$

- **Recall:** The recall is the proportion of the correctly identified positive cases.

$$Recall = \frac{TP}{TP + FN} \quad (12)$$

- **F1 Score:** F1 score is calculated as:

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (13)$$

- **Area Under the ROC (Receiver Operating Characteristic) curve (AUC)**: The AUC is a specific performance metric used to assess the ability of a classification model to distinguish between two classes, ranging from 0 to 1.

4. Results

In the following section, we describe the findings obtained in the two steps of the proposed framework, for binary and multiclass classification. We successfully implemented and fitted nine ML models. We split each dataset (CNvsAD, CnvsMCI, and TriClass) into two groups: a training set containing 80% of the total data, which was used to train the ML models, and a testing set containing 20% of the total data, which was used to evaluate the performance of the trained models.

4.1. Optimal hyperparameters

The optimal hyperparameters were automatically tuned with the help of the GridSearchCV tuner. Table 3 shows the optimal hyperparameters achieved based on the best accuracy.

Table 3. Optimal hyperparameters achieved from the first step.

ML model	Optimal hyperparameter		
	CNvsAD subset	**CNvsMCI subset**	**TriClass dataset**
Logistic Regression	C=1000; Penality=L1; Solver=liblinear	C=100; Penality=L1; Solver=liblinear	C=1000; Penality=L1; Solver=liblinear
Decision Tree	Criterion=gini; max_depth=4; max_leaf=4	Criterion=gini; max_depth=4; max_leaf=2	Criterion=gini; max_depth=4; max_leaf=2
Random Forest	Criterion=gini; max_depth=4; max_features=auto; n_estimators=200	Criterion=entropy; max_depth=9; max_features=log2; n_estimators=200	Criterion=gini; max_depth=4; max_features=auto; n_estimators=200
k-Nearest Neighbors	N_neighbours=20; metric=Manhattan	N_neighbours=7; metric=Manhattan	N_neighbours=14; metric=Manhattan
Support Vector Machine	C=1000; gamma=1; kernel=RBF	C=1000; gamma=1; kernel=linear	C=1000; gamma=1; kernel=RBF
Gaussian Naïve Bayes	Var_smoothing=4.5e-04	var smoothing=4.9e-06	var smoothing=3.9e-06
Multi-Layers perceptron	Learning_rate= adaptive; hidden layer=10; solver=adam; activation function= tanh.	Learning_rate= adaptive; hidden layer=21; solver=adam; activation function= sigmoid.	Learning_rate= adaptive; hidden layer=40; solver=adam; activation function= tanh.
eXterme Gradient Boost	Learning_rate=0.001; max_depth=4; n_estimators=200	Learning_rate=0.001; max_depth=2; n_estimators=200	Learning_rate=0;0001; max_depth=4; n_estimators=200
Gradient Boosting	max_depth=2; max_leaf=2; n_estimators=200	max_depth=4; max_leaf=2; n_estimators=200	max_depth=4; max_leaf=2; n_estimators=200

4.2. Performance Comparison

After obtaining the optimal hyperparameters of each model from the first step, we evaluated the overall accuracy of the models by averaging the performance accuracy of each fold of cross-validation repeated five times. To better visualize the classification performance of proposed models, Figure 3 shows the performance accuracy of the ML models performed for the binary classification, while Figure 4 presents the model accuracies of multiclass classification.

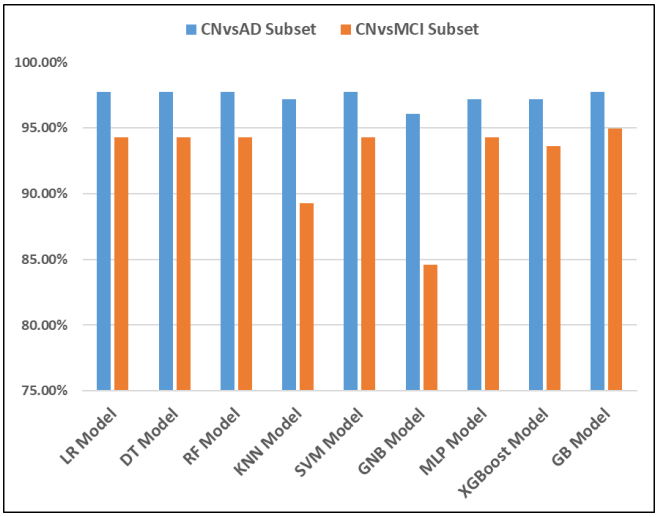

Figure 3. Overall accuracies of the nine models trained for binary classification.

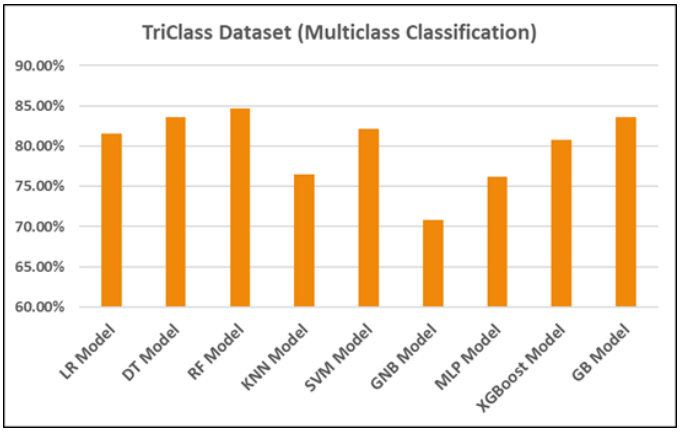

Figure 4. Overall accuracies of the nine models trained for multiclass classification.

For more details, Table 4 and Table 5 present the performance metrics of the models that use the CNvsAD and CNvsMCI subsets (binary classification).

For multiclass classification, Table 6 summarizes the performance of the results of the models trained on the TriClass dataset.

The tables present several metrics used to evaluate the models' performance, including accuracy, precision, recall, and F1-score.

Table 4. Performance metrics of the models trained on ADvsCN subset (binary classification).

Model	Accuracy	Precision	Recall	F1 score
LR	**97.74%**	**95.92%**	**98.48%**	**97.10%**
DT	**97.74%**	**95.92%**	**98.48%**	**97.10%**
RF	**97.74%**	**95.92%**	**98.48%**	**97.10%**
KNN	97.18%	95.00%	98.11%	96.40%
SVM	**97.74%**	**95.92%**	**98.48%**	**97.10%**
GNB	96.05%	93.27%	97.35%	95.03%
MLP	97.18%	95.00%	98.11%	96.40%
XGBoost	97.18%	95.00%	98.11%	96.40%
GB	**97.74%**	**95.92%**	**98.48%**	**97.10%**

Table 5. Performance metrics of the models trained on CNvsMCI subset (binary classification).

Model	Accuracy	Precision	Recall	F1 score
LR	94.31%	94.57%	93.67%	94.06%
DT	94.31%	94.74%	93.54%	94.04%
RF	94.31%	94.57%	93.67%	94.06%
KNN	89.30%	88.75%	89.56%	89.05%
SVM	94.31%	94.41%	93.80%	94.08%
GNB	84.62%	84.42%	85.60%	84.45%
MLP	94.31%	94.97%	93.67%	94.06%
XGBoost	93.65%	93.58%	93.23%	93.40%
GB	**94.98%**	**95.45%**	**94.23%**	**94.74%**

Table 6. Performance metrics of models trained on TriClass dataset (multiclass classification).

Model	Accuracy	Precision	Recall	F1 score
LR	81.59%	76.18%	74.00%	74.88%
DT	83.57%	79.65%	**78.97%**	79.28%
RF	**84.70%**	**82.11%**	78.65%	**80.06%**
KNN	76.49%	73.21%	72.81%	72.61%
SVM	82.15%	77.74%	72.79%	74.38%
GNB	70.82%	68.10%	69.66%	67.45%
MLP	76.20%	73.95%	74?63%	73.45%
XGBoost	80.74%	79.08%	77.02%	77.96%
GB	83.57%	79.78%	77.39%	78.40%

The experimental results of the performance comparison of LR, DT, RF, KNN, SVM, GNB, MLP, XGBoost, and GB classifiers showed that, for binary classification using the CNvsAD subset, the LR, DT, RF, SVM, and GB models achieved the highest accuracies among all models, with a score of 97.74%. The MLP, XGBoost, and KNN classifiers followed closely behind with accuracy scores of 97.18%. In contrast, the GNB

classifier showed the lowest accuracy among all models, with an overall score of only 96.05%. For the CNvsMCI subset, the GB model yielded the highest accuracy, with a score of 94.98%.

In multiclass classification, when using TriClass dataset, the RF classifier achieved the highest accuracy, with a score of 80.70%

4.3. Receiver Operating Characteristic Curve (ROC)

Figure 5 and Figure 6 show the ROC curves for binary classification for each class pair: CN-AD using the CNvsAD subset, and CN-MCI using the CNvsMCI subset. For the CN-AD classes, most models achieved a high AUC value (AUC = 0.98), indicating that they provide consistent and accurate predictions for these classes, which appear to be linearly separable. While, the LR, DT, RF, SVM, MLP, and GB models achieved an AUC of 0.94 for the CN-MCI classes, suggesting that distinguishing between the CN and MCI classes is more challenging for some models. The GNB model showed poor AUC overall.

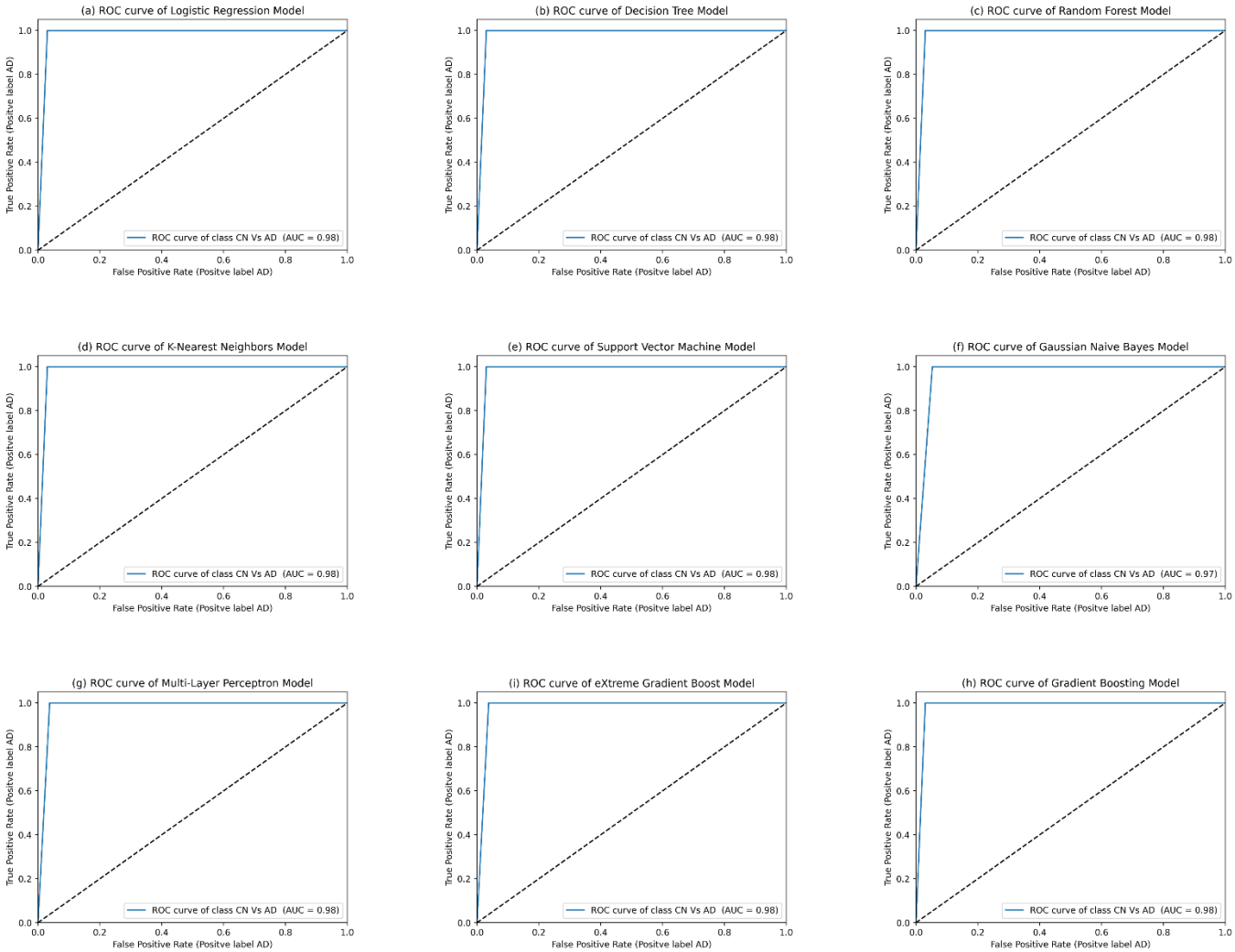

Figure 5. ROC curves for pair class CN-AD that trained and tested on CNvsAD subset: *(a) Logistic Regression; (b) Decision Tree; (c) Random Forest; (d) K-Nearest Neighbors; (e) Support Vector Machine; (f) Gaussian Naïve Bayes; (g) multi-layer perceptron; (h) eXtreme Gradient Boost; and (i) Gradient Boosting.*

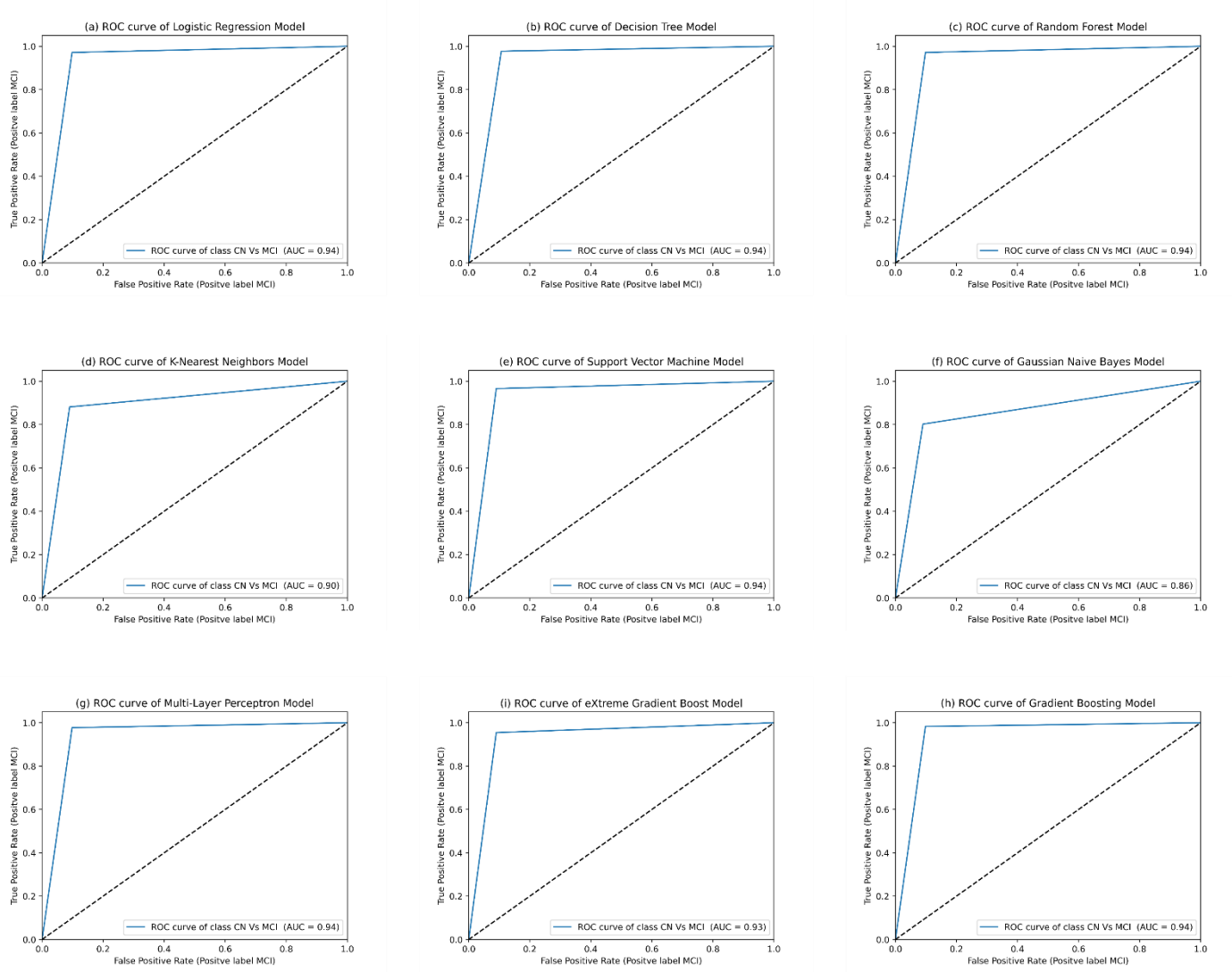

Figure 6. ROC curves for pair class CN-MCI that trained and tested on CNvsMCI subset: *(a) Logistic Regression; (b) Decision Tree; (c) Random Forest; (d) K-Nearest Neighbors; (e) Support Vector Machine; (f) Gaussian Naïve Bayes; (g) multi-layer perceptron; (h) eXtreme Gradient Boost; and (i) Gradient Boosting*.

The multiclass ROC curves for the three classes were plotted in Figure 7 using the prediction from all models using the One-vs-Rest strategy, each class has its own ROC curve. The blue curve is for CN class, the green is for MCI and the red curve is for AD. Based on the plotted data, it is apparent that the CN-Class performs best in the LR, DT, RF, and SVM models, having the highest Area Under the Curve (AUC=0.94), indicating consistent and accurate predictions for this class. The MCI-Class, however, has the highest AUC (0.88) for the RF model. The AD-Class performs best in XGBoost model having an AUC of 0.94, which outperforms all other models.

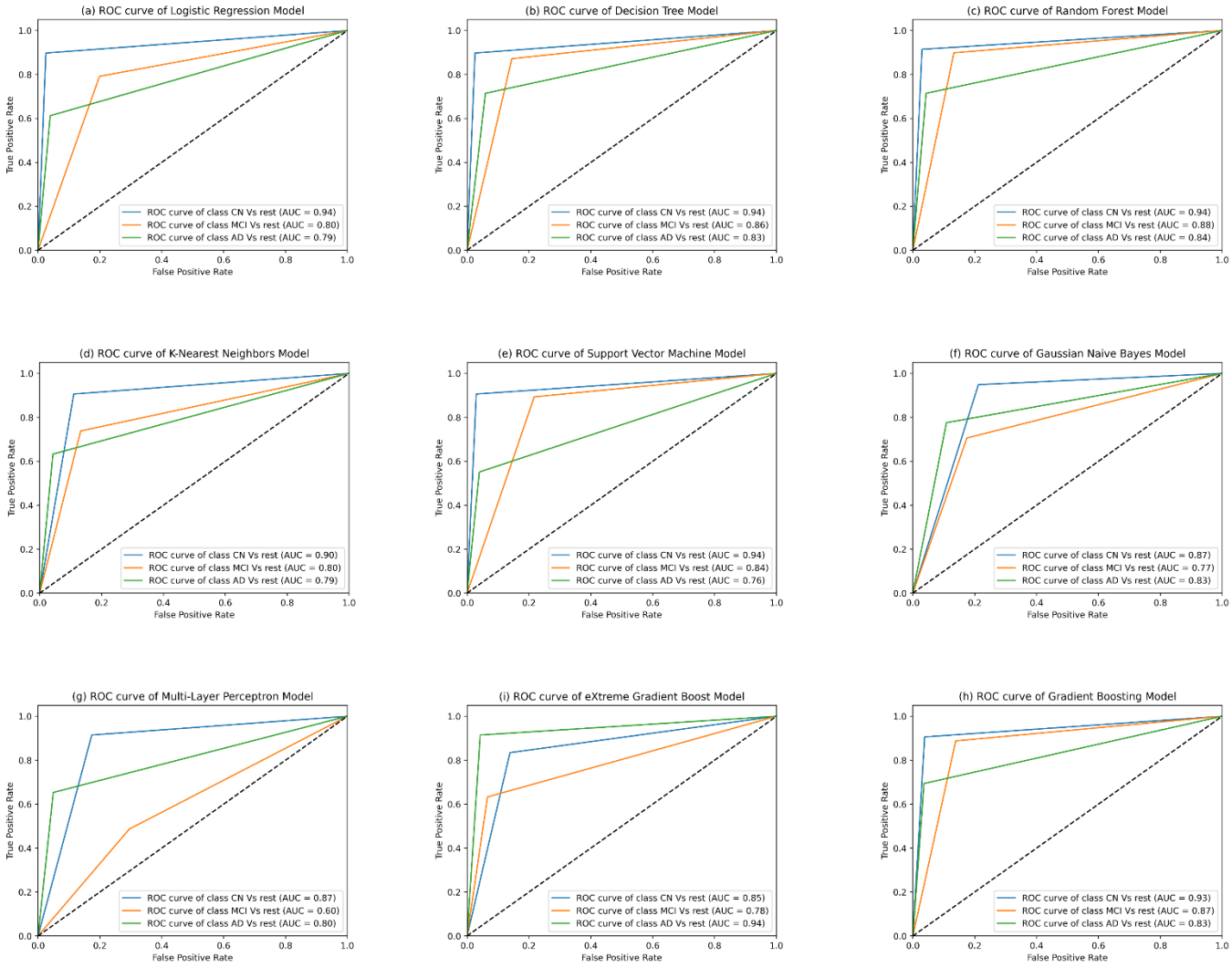

Figure 7. ROC curves for multiclass classification that trained and tested on TriClass dataset: *(a) Logistic Regression; (b) Decision Tree; (c) Random Forest; (d) K-Nearest Neighbors; (e) Support Vector Machine; (f) Gaussian Naïve Bayes; (g) multi-layer perceptron; (h) eXtreme Gradient Boost; and (i) Gradient Boosting*.

5. Discussion

The findings of this study indicate that the performance of any machine learning model is influenced by the correct selection of the ML algorithms and their associated hyperparameters. Therefore, selecting the most appropriate hyperparameters is crucial for achieving optimal performance. Using the GridSearchCV tuner helped to effectively adjust and optimize the hyperparameters of the ML models. The selected hyperparameters included using the regularization techniques (L1 and L2) to address overfitting in the LR model, adjusting the number of weak learners in the XGBosst, GB and RF classifiers to enhance performance and reduce overfitting, modifying the number of neighbors for the KNN model, and adjusting the kernel and C value for the SVM model. While GridSearchCV is a widely used technique for hyperparameter tuning, it does have some limitations, one of the main drawbacks is that it can be very time-consuming, as it needs to test every possible combination of hyperparameters. This can be a challenge when dealing with large datasets or complex models, requiring significant computational power and time. However, there are alternative optimization techniques available that can help address this issue. For example, RandomizedSearchCV and HalvingGridSearchCV from scikit-learn can test a broader range of hyper-parameter values within the same computation time as GridSearchCV, making them more efficient options for hyperparameter tuning.

Comparison of the results from the nine models revealed that the LR, DT, RF, SVM, and GB models achieved the highest classification accuracies for the CN-

AD pair (97.74%), indicating that when data is linearly separable, these ML models perform well, making it difficult to evaluate their individual classification performance. However, when analyzing the results for the second subset (CNvsMCI) in binary classification, a significant gap in model performance emerged. For instance, the GB model achieved the highest accuracy when trained on the CNvsMCI subset (94.98%), and the RF model performed very well when trained on the TriClass dataset, with an accuracy score of 84.70%. This suggests that ensemble learning methods such as GB and RF are highly suitable for the diagnosis of Alzheimer's disease based on neuropsychological assessments. The advantage of ensemble learning algorithms is their ability to improve accuracy by combining multiple models (such as decision trees in our case), often resulting in better performance than individual models. They are more robust to overfitting, especially in high-dimensional data, as they mitigate the impact of errors from individual models using techniques like averaging or voting. Ensemble methods are particularly effective in handling imbalanced and high-dimensional datasets. By reducing both bias and variance, they achieve an optimal balance between these factors, enhancing overall model performance. In contrast, Individual models often rely on specific assumptions about the data distribution or the relationships between features and target variables. For example, LR assumes a linear relationship between input features and the logarithmic probability of the target, while DT may overfit the training data, capturing noise rather than general patterns. These limitations are common among most individual models and can reduce their ability to generalize effectively to new data.

Although the results of this study show that the classification of AD based on neuropsychological markers is promising, it has some limitations. For example, some diseases such as Schizophrenia and Parkinson's disease can exhibit similar psychological symptoms as AD, which could lead to misdiagnosis. Additionally, the limited sample size affects the results and highlights the need for future research to enhance the dataset by incorporating additional markers, such as brain imaging, and genetic and biological markers. Moreover, using hybrid modeling with various datasets could potentially lead to the development of more robust and accurate models for diagnosing AD.

6. Conclusion

This study assessed the effectiveness of nine supervised ML algorithms, including LR, DT, RF, KNN, SVM, GNB, MLP, XGBoost, and GB, for predicting Alzheimer's disease. The aim was to explore the capability of each model to correctly classify individuals with AD using neuropsychological assessments. We used the GridSearchCV function from the scikit-learn library to tune the hyperparameters of each model. Obtaining the optimal hyperparameters allowed us to build nine robust ML models, train them, and compare their classification performance. The results indicated that ensemble learning models, such as Gradient Boosting and Random Forest, demonstrate superior predictive power in Alzheimer's disease classification. These models outperform other algorithms by effectively identifying patterns in complex datasets and making accurate classifications. Their strength lies in handling non-linearly separable data, where the classes cannot be divided by a simple linear boundary, and in multiclass classification tasks, such as categorizing data into CN, MCI, and AD. This makes GB and RF particularly robust and efficient for analyzing neuropsychological data and addressing the challenges of AD diagnosis.

The future work involves combining the neuropsychological data with various biological and neuroimaging markers to develop a robust model capable of predicting the early stages of AD.

Acknowledgments

We acknowledge the Alzheimer's Disease Neuroimaging Initiative for permitting us to access and use their database.

Competing Interest Statement

The authors declare no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Data Availability

The data used in this study was obtained from the ADNI database (http://adni.loni.usc.edu) and is available to all researchers with permission.

References

[1] R. A. Sperling, P. S. Aisen, L. A. Beckett, et al., "Toward defining the preclinical stages of Alzheimer's disease: recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease," *Alzheimer's & Dementia*, vol. 3, no. 7, pp. 280–292, 2011, doi: 10.1016/j.jalz.2011.03.003.

[2] L. Bäckman, S. Jones, A. K. Berger, E. J. Laukka, and B. Small, "Multiple cognitive deficits during the transition to Alzheimer's disease," *Journal of Internal Medicine*, vol. 3, no. 256, pp. 195–204, 2004, doi: 10.1111/j.1365-2796.2004.01386.x.

[3] A. Illán, J. Górriz, J. Ramírez, D. Salas-González, M. López, F. Segovia, P. Padilla, and C. G. Puntonet, "Projecting independent components of SPECT images for computer-aided diagnosis of Alzheimer's disease," *Pattern Recognition Letters*, vol. 31, no. 11, pp. 1342–1347, 2010.

[4] Y. Wang, X. Liu, and C. Yu, "Assisted diagnosis of Alzheimer's disease based on deep learning and multimodal feature fusion," *Complexity*, vol. 2021, no. 1, art. no. 6626728, 2021.

[5] K. R. Kruthika, H. D. Maheshappa, and Alzheimer's Disease Neuroimaging Initiative, "Multistage classifier-based approach for Alzheimer's disease prediction and retrieval," *Informatics in Medicine Unlocked*, vol. 14, pp. 34–42, 2019, doi: 10.1016/j.imu.2018.12.003.

[6] F. E. Al-Khuzaie, O. Bayat, and A. D. Duru, "Diagnosis of Alzheimer's disease using 2D MRI slices by convolutional neural network," *Applied Bionics and Biomechanics*, vol. 2021, no. 1, art. no. 6690539, 2021, doi: 10.1155/2021/6690539.

[7] M. Kaya and Y. Çetin-Kaya, "A novel deep learning architecture optimization for multiclass classification of Alzheimer's disease level," *IEEE Access*, 2024, vol. 12, pp. 1–11.

[8] D. A. Arafa, H. E.-D. Moustafa, H. A. Ali, et al., "A deep learning framework for early diagnosis of Alzheimer's disease on MRI images," *Multimedia Tools and Applications*, vol. 83, no. 2, pp. 3767–3799, 2024.

[9] J. Venugopalan, L. Tong, H. R. Hassanzadeh, and M. D. Wang, "Multimodal deep learning models for early detection of Alzheimer's disease stage," *Scientific Reports*, vol. 11, pp. 1–13, 2021, doi: 10.1038/s41598-020-74399-w.

[10] D. Castillo-Barnes, L. Su, J. Ramírez, et al., "Autosomal dominantly inherited Alzheimer disease: Analysis of genetic subgroups by machine learning," *Information Fusion*, vol. 58, pp. 153–167, 2020, doi: 10.1016/j.inffus.2020.01.001.

[11] T. Jo, K. Nho, S. L. Risacher, and A. J. Saykin, "Deep learning detection of informative features in tau PET for Alzheimer's disease classification," *BMC Bioinformatics*, vol. 21, no. 21, pp. 1–13, 2020.

[12] R. Cui, M. Liu, and G. Li, "Longitudinal analysis for Alzheimer's disease diagnosis using RNN," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 1398–1401, 2018, doi: 10.1109/ISBI.2018.8363833.

[13] M. Rohini and D. Surendran, "Toward Alzheimer's disease classification through machine learning," *Soft Computing*, vol. 25, no. 4, pp. 2589–2597, 2021, doi: 10.1007/s00500-020-05292-x.

[14] A. Khan and S. Zubair, "An improved multi-modal based machine learning approach for the prognosis of Alzheimer's disease," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 2688–2706, 2020, doi: 10.1016/j.jksuci.2020.04.004.

[15] I. Almubark, L. C. Chang, K. F. Shattuck, et al., "A 5-min cognitive task with deep learning accurately detects early Alzheimer's disease," *Frontiers in Aging Neuroscience*, vol. 12, art. no. 603179, 2020, doi: 10.3389/fnagi.2020.603179.

[16] N. T. Duc, S. Ryu, M. N. I. Qureshi, et al., "3D-deep learning based automatic diagnosis of Alzheimer's disease with joint MMSE prediction using resting-state fMRI," *Neuroinformatics*, vol. 18, no. 1, pp. 71–86, 2020, doi: 10.1007/s12021-019-09419-w.

[17] A. Khan and S. Zubair, "Development of a three-tiered cognitive hybrid machine learning algorithm for effective diagnosis of Alzheimer's disease," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 1–19, 2022, doi: 10.1016/j.jksuci.2022.07.016.

[18] N. Seshadri, S. McCalla, and R. Shah, "Early prediction of Alzheimer's disease with a multimodal multitask deep learning model," *Journal of Student Research*, vol. 10, no. 1, pp. 1–10, 2021.

[19] S. F. Muller, "Behavioral disturbances in dementia," *Dialogues in Clinical Neuroscience*, vol. 5, no. 1, pp. 49–59, 2003.

[20] Alzheimer's Disease Neuroimaging Initiative (ADNI), Accessed: Apr. 20, 2024. [Online]. Available: https://adni.loni.usc.edu/.

[21] T. Ye, C. Zu, B. Jie, D. Shen, and D. Zhang, "Discriminative multi-task feature selection for multi-modality classification of Alzheimer's disease," *Brain Imaging and Behavior*, vol. 10, no. 3, pp. 739–749, 2016, doi: 10.1007/s11682-015-9437-x.

[22] Y. Liang, C. Liu, X.-Z. Luan, K.-S. Leung, T.-M. Chan, Z.-B. Xu, and H. Zhang, "Sparse logistic regression with a L1/2 penalty for gene selection in cancer classification," *BMC Bioinformatics*, vol. 14, pp. 1–12, 2013.

[23] R. Wang, N. Xiu, and C. Zhang, "Greedy projected gradient-newton method for sparse logistic regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 2, pp. 527–538, 2019.

[24] A. J. Myles, R. N. Feudale, Y. Liu, N. A. Woody, and S. D. Brown, "An introduction to decision tree modelling,"

Journal of Chemometrics, vol. 18, no. 6, pp. 275–285, 2004.

[25] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[26] J. Ali, R. Khan, N. Ahmad, and I. Maqsood, "Random forests and decision trees," *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 5, pp. 272, 2012.

[27] R. Agrawal, "K nearest neighbor for uncertain data," *International Journal of Computer Applications*, vol. 105, no. 11, 2014.

[28] A. S. Thakur and N. Sahayam, "Speech recognition using Euclidean distance," *International Journal of Emerging Technology and Advanced Engineering*, vol. 3, no. 3, pp. 587–590, 2013.

[29] W. S. Noble, "What is a support vector machine?" *Nature Biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

[30] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*, 2nd ed. Springer, 2006, pp. 1122–1128.

[31] M. Hashimoto, M. Ohtaka, K. Ara, I. Kanno, R. Imamura, K. Mikami, S. Nomiya, and H. Onabe, "Neural networks—A systematic introduction," *Journal of Nuclear Science and Technology*, vol. 46, no. 1, pp. 76–82, 2009.