Fusion of cognitive health: Advanced prediction of Alzheimer's disease

Author: Chahat Ahuja, Sejal Badwaik, Ayushee Bhawsagar, Dipali Kamble

Guided By: Prof. Gaurav Agarwal

Cummins College of Engineering for Women, Nagpur

ABSTRACT

This review article presents an integrative approach to developing a predictive model for Alzheimer's disease (AD). Alzheimer's disease is a complex neurodegenerative disease with a significant societal and health burden. Traditional predictive models often focus on single data modalities or biomarkers that may not fully capture the multifactorial nature of AD.

Our approach integrates multiple data sources, including clinical, genetic, neuroimaging, and biomarker data, to create a comprehensive predictive model. We review recent advances in each data modality, discuss challenges in data integration, and suggest strategies to overcome these challenges. Furthermore, we highlight the importance of interpretability and generalizability in predictive modelling for AD. By synthesizing knowledge from different fields, our integrative approach aims to increase the accuracy and reliability of predictive models for the early detection and prognosis of Alzheimer's disease.

INTRODUCTION

Alzheimer's disease (AD) represents a significant and growing global health challenge, with its prevalence expected to increase dramatically in the coming decades. As one of the most prevalent neurodegenerative disorders, AD not only profoundly affects the quality of life of individuals, but also places a significant burden on healthcare systems and society as a whole. In the absence of a cure and the limited efficacy of available treatments, there is an urgent need for innovative approaches to better understand, diagnose, and potentially prevent or delay the onset of AD.

In recent years, predictive modelling has emerged as a promising avenue for addressing the complexities of AD. Through the use of advanced computational techniques and the integration of various data sources, predictive models offer the potential to identify individuals at risk of developing AD before clinical symptoms appear. Such early detection is essential to facilitate early interventions, optimize treatment strategies, and ultimately improve patient outcomes.

This review article provides a comprehensive overview of the current state of predictive modelling for AD, focusing on an integrative approach that includes multiple data modalities and analytical methodologies. We explore different types of data used in predictive modelling, including clinical trials, neuroimaging data, genetic information, biomarkers, and digital health data. In addition, we examine the various computing techniques used, from traditional statistical methods to machine learning algorithms and artificial intelligence approaches.

We further discuss issues and limitations associated with predictive modelling for AD, such as data heterogeneity, sample size limitations, feature selection, model interpretability, and ethical considerations. By critically evaluating existing approaches and identifying areas for improvement, we aim to direct future research efforts toward more robust and clinically relevant predictive models.

Ultimately, the integration of diverse data sources and advanced analytical techniques holds great promise for improving our understanding of AD pathogenesis, improving early detection and risk stratification, and informing personalized interventions. By advancing the predictive modelling of AD, we can make significant strides toward mitigating the devastating impact of this debilitating disease on individuals, families, and society as a whole.

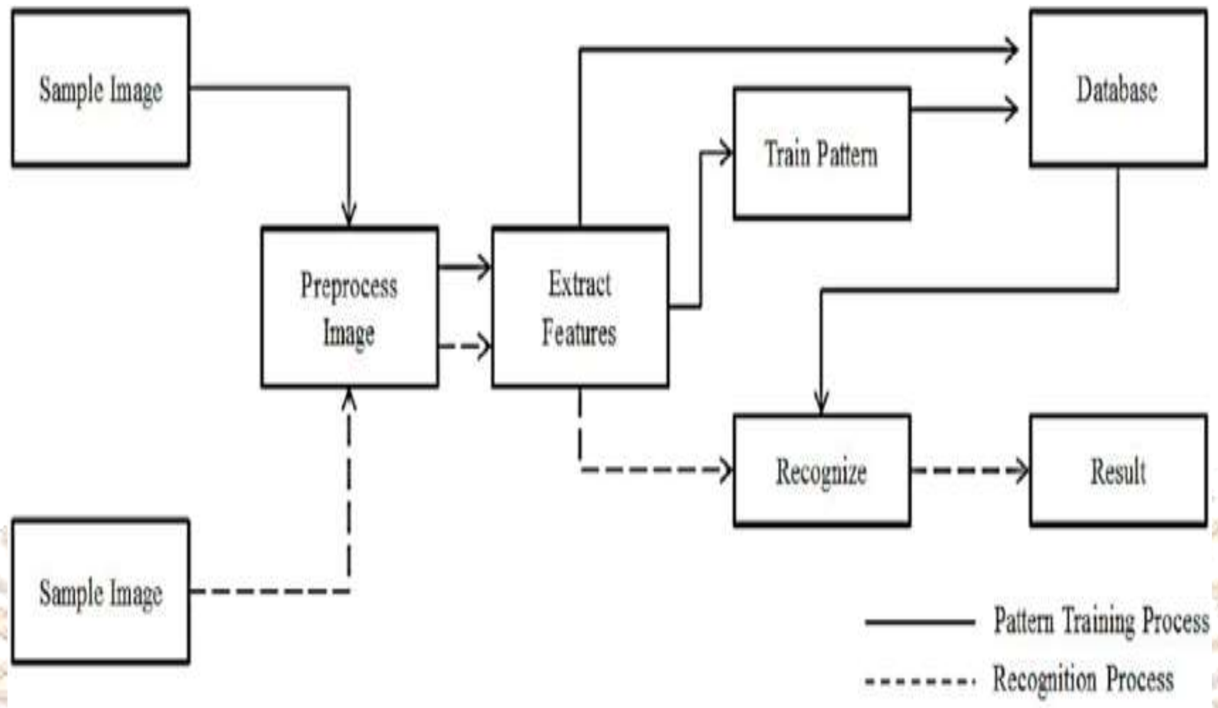


Fig: Block Diagram

Here is a brief description of each term:

1. Sample Image: Refers to an example or instance of an image used for analysis, processing, or training in a computer vision or image processing task.

2. Image Preprocessing: Preprocessing involves various operations performed on an image before further analysis or processing. This can include tasks such as resizing, noise reduction, colour balancing and image enhancement.

3. Feature Extraction: In the context of image processing or machine learning, feature extraction involves identifying and extracting relevant information or characteristics from an image. These features can include borders, textures, colours or more complex patterns.

4. Pattern Training: This usually refers to the process of training a machine learning model, such as a classifier or repressor, using a set of labelled data (patterns) to learn patterns or relationships in the data.

5. Database: In the context of image recognition or pattern recognition, a database can refer to a collection of images or other data used for training, testing, or reference purposes.

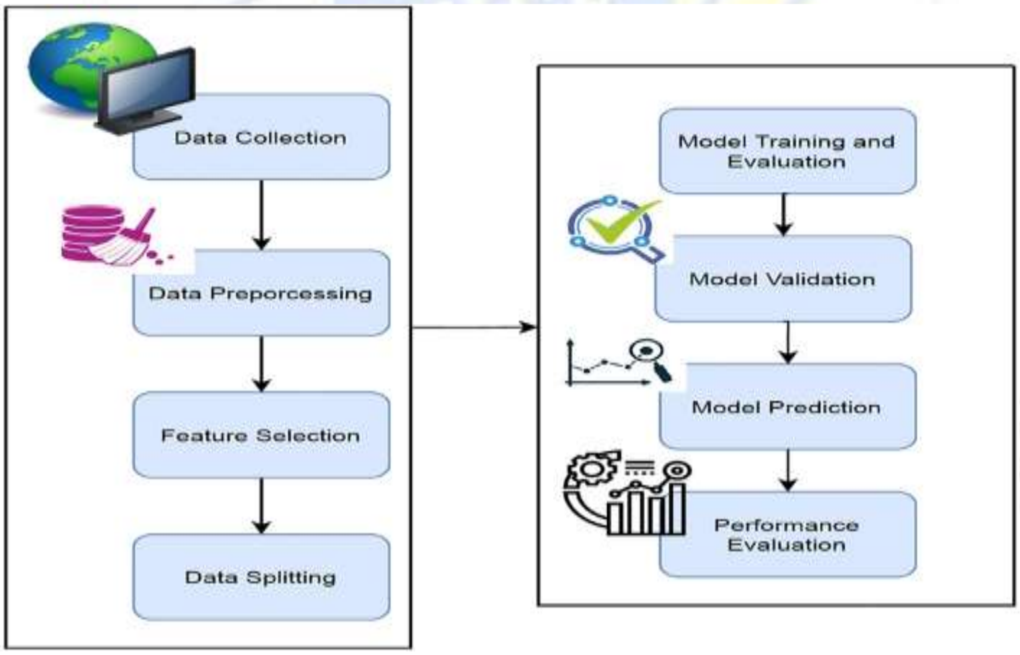
6. Recognize: Recognition involves the process of identifying or categorizing an object, pattern, or entity based on its properties or characteristics. In the context of image recognition, this often involves comparing features extracted from an input image to those stored in a database or learned during training.

7. Result: The result or output of the recognition process, indicating what was identified or recognized in the input image based on the analysis and processing steps performed.

MATERIALS AND METHODS

The proposed approach consists of three basic steps. Firstly, the Alzheimer's disease dataset was loaded into pandas for data preprocessing. This study utilized a longitudinal dataset, so a timeline of the study was necessary to gain further insight into the data. Our first step was to determine how cross-sectional the data appear to be, if either at a baseline or at a particular time. A complete analysis of the data was then conducted, including a comparison of the main study parts and the corresponding data collected during each visit. In this work, longitudinal MRI data is our primary data source.

The Machine Learning techniques were applied to Alzheimer's disease datasets to bring a new dimension to predict Disease at an early stage. The raw Alzheimer's disease datasets are inconsistent and redundant, which affects the accuracy of algorithms. Before evaluating machine-learning algorithms, data must be effectively prepared for analysis by removing unwanted attributes, missing values, and redundant records. Building a machine-learning model requires splitting the data into training and testing sets. In the following data preparation step, the training data were used to create a model, which was then applied to test data to predict Alzheimer's Disease. The model was trained from training set data, and test set data were used to test unseen data. Cross-validation was carried out by dividing the dataset into three subsets. Model predictions are made using one subset of the data (test data) and the performance of the model is evaluated using the other subsets (training and validation) of the data. The data was pre-processed and we randomly split it 80:20 with 80% going to training and 20% to testing.



Data Preparation

Various data mining techniques were used to clean and pre-process the data in this phase. As part of this, missing values are handled, features are extracted and features are transformed, and so on. We identified 9 rows with missing values in the SES column. This problem is solved in two ways. The simplest solution is to drop the rows with missing values. Another way to fill in missing values is through imputation, which means replacing them with the corresponding values. The model should perform better if we impute since we only have 140 measurements. 9 rows with missing values are removed in the SES attribute and the median value is used for imputation.

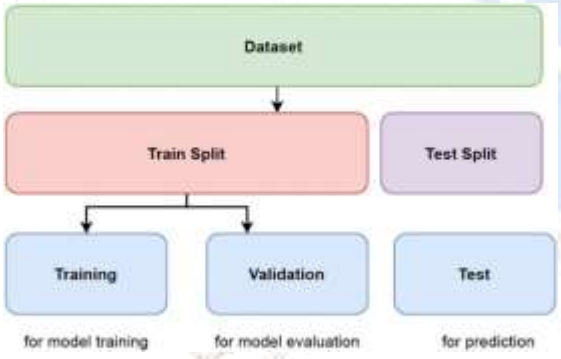
Data analysis

In this section, we discussed the relationships between each feature of the MRI test and dementia. In order to formulate the relationship of the data explicitly using a graph, we performed this process of Exploratory Data Analysis to estimate correlations before extracting or analysing the data. The information could be used to later interpret the nature of the data and determine which method to use to analyse it.

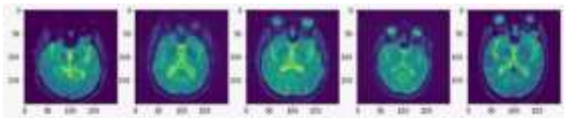
Selection of functions

Feature selection is very important in machine learning. In this work, feature selection is applied to Alzheimer's disease clinical data, where we have thousands of samples. Element selection has three methods, for example: filter methods, Wrapper methods, and nested methods. The filtration method is a common method used in the pre-processing stage. Wrapper methods are another method that forms the core of a subset of functions. Finally, the Embedded method combines the filter and wrapper methods.

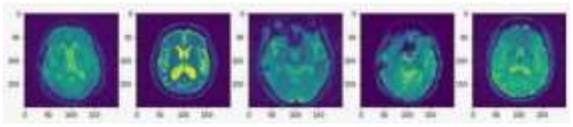
Data preparation and distribution]



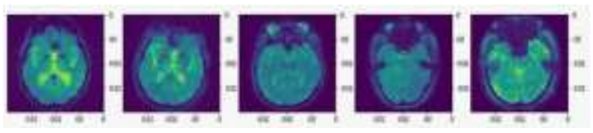
Training Datasets for Alzheimer's Disease



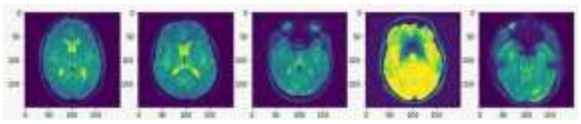
Testing Datasets for Alzheimer's Disease



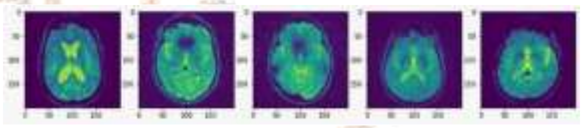
Training Datasets for Control Disease



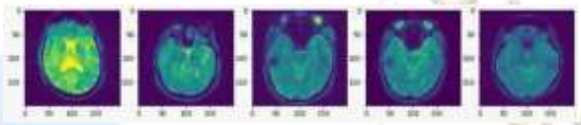
Testing Datasets for Control Disease



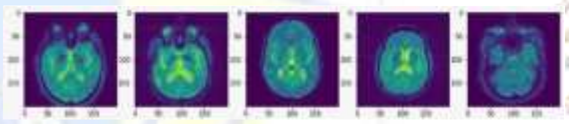
Training Datasets for Early Mild Cognitive Impairment (EMCI)



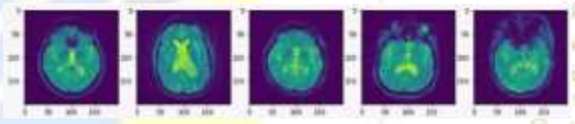
Testing Datasets for Early Mild Cognitive Impairment (EMCI)



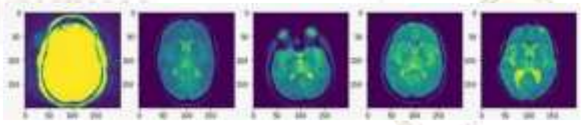
Training Datasets for Late Mild cognitive Impairment (LMCI)



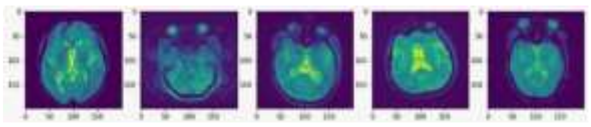
Testing Datasets for Late Mild Cognitive Impairment (LMCI)



Training Datasets for Mild Cognitive Impairment (MCI)



Testing Datasets for Mild Cognitive Impairment (MCI)



Classifier Models

Support Vector Machine (SVM) It is a powerful and versatile machine learning algorithm widely used for classification and regression tasks. The basic principle of SVM is to find the optimal hyperplane that best separates data points belonging to different classes in a high-dimensional space. Linear SVM, which uses a linear kernel, aims to find a linear decision boundary, so it is suitable for problems where the classes are linearly separable. However, SVMs can be extended to handle non-linear relationships using different kernel functions, such as polynomial or radial basis function (RBF) kernels.

One of the strengths of SVM is its ability to efficiently process high-dimensional data, which makes it particularly suitable for scenarios where the number of features is large compared to the number of samples. SVM is known for its strong generalization, which is crucial in applications such as medical diagnostics, including the detection of Alzheimer's disease. SVMs are robust to bias and their decision boundaries are influenced by support vectors, which are the data points closest to the decision boundary. This property makes SVMs less sensitive to outliers.

However, SVMs also have some disadvantages. They can be sensitive to the choice of hyper parameters, such as the regularization parameter, and appropriate tuning of these parameters is essential for optimal performance. Additionally, SVMs can struggle with large datasets as their computational complexity can increase with the number of samples. Interpreting a decision made by an SVM, especially in high-dimensional spaces, can be challenging, leading to a lack of transparency that can be a problem in applications where interpretability is key.

Support Vector Machines offer a robust and efficient approach for classification tasks, including Alzheimer's disease detection, by finding optimal decision boundaries in high-dimensional spaces. Although they show strengths in generalization and robustness to over fitting, their sensitivity to hyper parameters and interpretability issues should be carefully considered when applying SVMs in practical scenarios.

Random Forest (RF) (bagging) Think of a forest, not trees, but decision trees, each a simple model that makes predictions based on a series of yes-no questions. A random forest is like a strong group decision making process in this forest. It combines the predictions of many different decision trees, each built with a slightly different view of the data, to arrive at a more accurate and robust final prediction. This approach, which combines different individual predictions, helps avoid the pitfalls of any single tree and leads to better overall performance.

Random Forest is a powerful machine learning algorithm that belongs to the ensemble learning family. Unlike traditional stand-alone decision trees, Random Forest creates a large number of decision trees and combines their predictions to achieve robust and accurate results. Each decision tree is trained on a random subset of the training data and a random subset of the features, thereby introducing diversity among the trees. The ensemble nature of Random Forest helps mitigate overfitting and increases the generalizability of the model. The algorithm uses a technique called bootstrap aggregating to create these diverse subsets of data. During the training process, each tree independently produces a prediction, and the final prediction is determined by aggregating the outputs, often through majority voting for classification or averaging for regression. One of the notable strengths of Random Forest is its ability to efficiently handle both classification and regression tasks. Additionally, it provides a feature importance score, allowing users to understand the importance of each feature when making predictions. While Random Forest is known for its high accuracy and robustness to noisy data, its main disadvantages include the possibility of increased computational complexity and memory usage, especially for large numbers of trees. Despite these considerations, Random Forest remains a popular choice for a wide variety of applications due to its versatility, ease of implementation, and ability to provide robust performance in a variety of scenarios.

Random Forest is a powerful ensemble learning technique widely used in machine learning for both classification and regression tasks. It belongs to the family of decision tree algorithms, but incorporates the concept of ensemble learning to improve predictive accuracy and robustness.

In Random Forest, multiple decision trees are trained on different subsets of training data and subsets of features. During the training process, each tree is built separately, and the final prediction is made by aggregating the predictions of all trees in the forest.

Random Forest is a powerful and versatile ensemble learning algorithm used in machine learning for both classification and regression tasks. It is an extension of the decision tree algorithm and works by constructing a large number of decision trees during training and outputting the mode (for classification) or average prediction (for regression) of each tree.

The basic idea of Random Forest is to introduce randomness into the tree building process to improve the overall performance and generalization of the model. This randomness is introduced in two main ways: by using a random subset of the training data for each tree, and by considering only a random subset of features at each split in the decision tree.

During the training phase, a number of decision trees are grown using different subsets of the training data. Each tree is constructed independently, making Random Forest a parallelizable algorithm suitable for large datasets. The randomness introduced helps to decorrelate individual trees, making the model less prone to overfitting and more robust to noisy data.

When making predictions, each tree in the forest produces an output, and the final prediction is determined by aggregating those outputs. For classification this usually involves getting a majority vote, while for regression it involves averaging the predictions. This ensemble approach helps to improve the overall accuracy and reliability of the model.

Random Forests are known for their excellent performance, flexibility and resistance to reassembly. They can handle a wide variety of data types, including categorical and numeric functions, and are less sensitive to tuning hyperparameters compared to individual decision trees. Random forests are widely used in various fields such as finance, healthcare and remote sensing, where accurate and reliable predictions are crucial.

K-Nearest Neighbour's (KNN) k-Nearest Neighbour's (k-NN) is a simple but powerful supervised machine learning algorithm used for both classification and regression tasks. The basic idea of k-NN is to predict the label or value of a data point by considering the majority class or average value of its k-nearest neighbours in the feature space. A distance metric, often the Euclidean distance, is used to measure the similarity or dissimilarity between data points.

In the case of classification, when predicting the label of a new data point, the algorithm identifies its k-nearest neighbours based on the chosen distance metric. The predicted class is then determined by the majority vote among these neighbours. For regression tasks, the predicted value is calculated as the average of the target values of the k-nearest neighbours.

One notable feature of k-NN is its reliance on local information. The algorithm does not learn a global model during the training phase, but instead stores the entire training dataset. This makes k-NN particularly suitable for scenarios where the decision boundary is complex and non-linear. However, it can be sensitive to the choice of distance metric and k value and may not perform well in high-dimensional spaces.

k-NN is a non-parametric instance-based learning algorithm, meaning that it does not assume a specific functional form for the underlying data distribution. Although k-NN is expensive for large datasets, it is easy to understand, implement, and interpret. It finds application in various fields such as pattern recognition, image processing, and recommendation systems, where relationships between data points are critical for accurate predictions.

Imagine attending a party where you don't know anyone. K-Nearest Neighbour (KNN) is like a social butterfly that helps you find out who to hang out with. KNN is a machine learning algorithm used for both classification and regression. It works by first choosing a value of K that represents the number of nearest neighbours you want to consider. Then, for a new data point (like you at the party), KNN calculates the distances between it and all existing data points (like other party goers). Finally, based on these distances, it identifies K nearest neighbours.

In the case of classification, KNN predicts the class (like group of friends) of a new data point by looking at the most frequent class among its K nearest neighbours. So, if you find yourself surrounded by people talking about technology, you're more likely to engage in that conversation based on KNN's prediction.

For regression, KNN predicts a continuous value (like average age at a party) by computing the average value of K nearest neighbours. This allows

KNN to make educated guesses about new data points based on the company they keep.

While KNN is simple and easy to understand, choosing the right value of K is critical. A K that is too small can be sensitive to noise in the data, while a K that is too large can miss important details. KNN is a versatile tool, but like any good party guest, it needs a little fine-tuning to suit a particular situation.

k-Nearest Neighbour's (k-NN) is a simple but powerful supervised machine learning algorithm used for both classification and regression tasks. The primary concept of k-NN is to make predictions based on the majority class or average of the k-nearest data points in the feature space.

In the case of classification, given a new data point, the algorithm identifies k-nearest neighbours by measuring the distance between the new point and all other data points in the training set using a distance metric such as Euclidean distance. The class label of most of these neighbours is assigned to the new point. The choice of k, the number of neighbours, is a crucial parameter that affects the performance of the algorithm and can be determined using cross-validation.

For regression tasks, instead of labelling classes, k-NN calculates the average or weighted average of the target values of the k-nearest neighbours. This predicted value becomes the output for the new data point.

One of the strengths of k-NN is its simplicity and intuitive nature. It is a non-parametric algorithm, meaning that it does not assume any underlying data distribution. However, the performance of the algorithm can be sensitive to the choice of k and the distance metric. Larger values of k lead to smoother decision boundaries but may oversimplify the model, while smaller values may make the model more sensitive to noise.

k-NN is particularly useful when the decision boundaries in the feature space are complex and irregular. However, its disadvantage lies in its computational cost during prediction, as it requires the calculation of distances to all training data points. Despite this, k-NN finds application in various fields, including pattern recognition, image recognition, and recommendation systems. Additionally, it is a lazy learning algorithm, meaning it does not build a model during the training phase, making it suitable for dynamic or frequently updated datasets.

Decision Tree (DT) A decision tree is a popular supervised machine learning algorithm used for both classification and regression tasks. It works by recursively partitioning the data set into subsets based on the input function values, with the goal of creating a tree structure of decision nodes and leaf nodes. Each decision node represents a test of a specific function, and each end node contains the predicted output.

The process of constructing a decision tree involves selecting the best function to partition the data at each decision node. The "best" feature is determined by a criterion such as Gini impurity for classification tasks or root mean squared error for regression tasks. The selected function is used to split the data set into two or more subsets, and this process is repeated recursively for each subset until a stopping criterion is met. Stopping criteria can include maximum tree depth, minimum number of samples at a leaf node, or others to avoid overrunning.

Decision trees are intuitive and easy to interpret, making them valuable for understanding the model's decision-making process. However, they are prone to overfitting, especially when the tree is too deep and captures noise in the training data. Techniques such as pruning, which involves removing branches of a tree that do not contribute significantly to its performance, can be used to mitigate overfitting.

Decision trees are used in a variety of fields, including finance, healthcare, and marketing, where it is essential to understand the factors influencing decisions. Although decision trees can be powerful on their own, they are often used as building blocks for ensemble methods such as random forests, where multiple trees are combined to improve overall model performance and robustness.

A decision tree is a popular machine learning algorithm that is widely used for both classification and regression tasks. The essence of a decision tree lies in its hierarchical structure of decision nodes, where each node represents a test on an attribute and each branch represents the result of that test. Starting from the root node, the tree recursively partitions the data based on the values of various functions, eventually leading to the leaf nodes where the final predictions are made.

The decision-making process in a decision tree involves selecting the most informative element at each node based on certain criteria, such as the Gini impurity for classification tasks or the root mean square error for regression tasks. The goal is to create a tree that effectively divides the data into homogeneous subsets, making predictions more accurate and more interpretable.

One of the strengths of decision trees is their ability to process both numerical and categorical data, as well as automatically select the most important features during the training process. However, decision trees are prone to overfitting and capture noise in the training data, which can be mitigated by using techniques such as pruning or using ensemble methods such as random forests.

Decision trees find applications in fields as diverse as finance, healthcare, and natural language processing due to their intuitive representation of decision processes and the ease with which they can be interpreted. They serve as a basic building block for more complex algorithms and are valuable tools for understanding and solving a wide variety of machine learning problems. A decision tree is a versatile and intuitive machine learning algorithm used for both classification and regression tasks. It works by recursively partitioning the input space into subsets based on the values of the input features, ultimately leading to a predictive model represented as a tree structure. Each internal node of the tree corresponds to a decision based on a specific feature, and each leaf node represents a predicted outcome.

The construction of a decision tree involves choosing the best function to partition the data at each internal node. The "best" feature

is determined by a criterion such as information gain (for classification) or reduction of mean squared error (for regression). The process continues recursively, creating branches and nodes, until a stopping criterion, such as a predefined depth or a minimum number of samples at a leaf node, is met.

Decision trees are interpretable and provide a clear visualization of the decision-making process. However, they are prone to overfitting and capture noise in the training data that may not generalize well to new, unseen data. Strategies such as pruning, limiting tree depth, or setting a minimum number of samples per leaf can help mitigate overstocking.

Decision trees are widely used in a variety of fields, including finance, healthcare, and natural language processing, due to their ability to process both numerical and categorical data. They serve as building blocks for ensemble methods such as random forests, which combine multiple decision trees to increase predictive accuracy and robustness. Overall, decision trees are valued for their simplicity, interpretability and effect.

Model Validation

Model verification reduces the problem of overfitting. Cross-validation is performed to train the ML model and is used to calculate the accuracy of the model. Building a noise-free ML model is a challenging task. Thus, in this research work, cross-validation is performed, which divides the entire data set

into an equal-sized parts. The ML model is trained for each iteration with n-1 divisions. The performance of the method is analysed as the average of all n-folds. In this work, the ML model was trained and tested 10 times by applying 10-fold cross-validation to the model.

RESULTS AND DISCUSSION

We evaluate various performance metrics such as accuracy and precision. To determine the best parameters for each model, we perform a 4-fold cross-validation: decision tree, SVM, random forests, and convolutional neural network. Finally, we compare the accuracy of each model. After building

the models, several metrics and techniques were used to identify problems with overfitting and parameter tuning. Performance

evaluation can be binary or multi-class and is described using a confusion matrix. A learning model was developed to distinguish real people with Alzheimer's disease from a given population, and a new machine learning classifier was developed and validated to predict and separate real people with Alzheimer's disease.

FUTURE SCOPE

In future work, we wish to train the network on an even larger data set. With access to open-source databases, researchers have access to a large set of data from various scanning sites. If a CNN is trained with a large number of data points from different locations, it has a higher chance of generalizing even to unexpected data. Second, we will also study different combinations of cores in the same network and compare their performance. One of the main core combos to focus on would be the first temp core.

With a larger window, better temporal similarity could be captured. ROIs extracted from a longer window length can be compared to resting-state functional networks such as the default mode network. With well-studied network parameters and an adequate amount of data, we want to explore the possibility of developing fMRI-based biomarkers for AD and MCI. Research is underway to identify MRI-based biomarkers that can predict the risk of developing Alzheimer's disease or the rate of disease progression.

In the future, we want to train and test datasets on convolutional neural network (CNN) and hybrid models.

CONCLUSION

Alzheimer's disease is a major health problem, and rather than offering treatment, it is more important to reduce risk, provide early intervention, and diagnose symptoms early and accurately. As can be seen from the literature survey, many efforts have been made to detect Alzheimer's disease using various machine learning algorithms and micro simulation methods; however, it still remains a challenging task to identify relevant attributes that can detect Alzheimer's disease very early.

Future work will focus on extracting and analysing new features that are more likely to help in Alzheimer's disease detection and removing redundant and irrelevant features from existing feature sets to improve the accuracy of detection techniques. By adding metrics like MMSE and Education to our model, we will be able to train it to distinguish between healthy adults and people with Alzheimer's disease.

REFERENCES

1. Random forest model for feature-based Alzheimer’s disease conversion prediction from early mild cognitive impairment subjects.
2. Comparative analysis of Alzheimer’s disease classification by CDR level using CNN, feature selection, and machine learning techniques.
3. Support Vector Machines in the diagnosis of Alzheimer’s Disease.
4. Alzheimer’s Disease detection empowered with transfer learning.
5. Alzheimer’s disease detection by using Deep Learning Algorithms.
6. Predictive models for Alzheimer’s Disease: An Integrated Approach.
7. Alzheimer’s disease: risk factors and potentially protective measures.
8. Early-Stage Alzheimer’s Disease Prediction using machine learning models.
9. Detection and analysis of Alzheimer’s disease using various machine learning algorithms.
10. Detection and analysis of Alzheimer’s disease using Support Vector Machine with linear kernel model.