EEG Analysis for Detection Alzheimer's disease

Hani Mediouni and Siwar Terres

7.

1. Introduction

This project aims to analyze EEG signals to identify the presence of Alzheimer's disease. EEG data come from healthy and affected patients, with recordings for both "eyes open" and "eyes closed" states. The goal is to build machine learning models to distinguish patients with the disease from healthy ones. Several algorithms are applied to evaluate the accuracy and robustness of each approach.

2. Data Preprocessing

2.1. Loading EEG Data

The data is organized into folders according to conditions ("Healthy", "AD" for Alzheimer's) and states ("Eyes_open" and "Eyes_closed"). A load function was created to extract this data and store it in a DataFrame:

```
eeg_data_df = load_eeg_data(data_dir)
```

This function allows you to browse folders, extract data files, and load them as digital signals for further analysis.

2.2. Data Cleaning

Raw data may contain outliers, empty files, or corrupted records. A cleaning process has been implemented to filter out these unwanted elements:

- **Identifying and removing noisy files**:Files containing excessive noise, outliers, or inconsistent data have been cleaned.
- **Outlier Removal**:Outliers were identified through statistical methods, such as the use of quartiles and interquartile range (IQR) visualized by a Boxplot. Values outside the acceptable ranges were removed.

- **Empty files and incomplete folders**:Empty files and patient records without usable data were removed from the dataset.

7.

then transform the raw EEG signals into meaningful statistical features that can be used for classification. The decision tree, which is a supervised learning model, is then trained to predict whether a patient has Alzheimer's disease based on these features.

3. Application of Algorithms

3.1. Algorithms on Raw Data

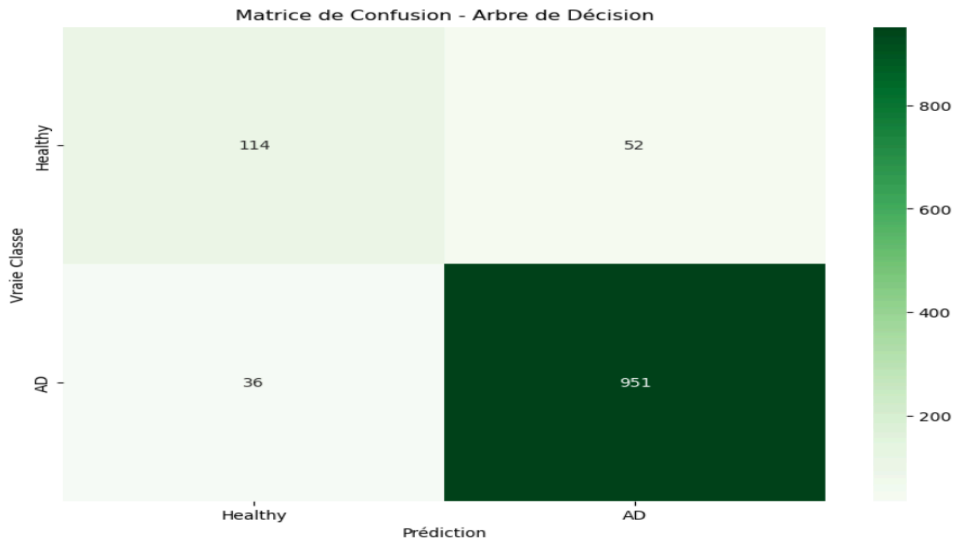
The following algorithms were applied directly to the raw EEG data:

1. **Decision Tree**
2. **Random Forest**
3. **KNN (K-Nearest Neighbors)**
4. **SVM (Support Vector Machine)**

For each algorithm, the following steps were followed:

General Steps:

1. **Creation of the Model**:Using the selected algorithm to create the model.
2. **Training**:Fitting the model on the training data.
3. **Prediction**:Predicting results on test data.
4. **Assessment**:Use ofaccuracy_scoreto measure accuracy.
5. **Visualization**:Displaying a confusion matrix and classification report to evaluate performance. as follows

6. 

7.

For each algorithm, the accuracy was calculated to measure the performance.

Here is an example of classification report for decision tree algorithm for raw data

Rapport de classification - Arbre de Décision :

	precision	recall	f1-score	support
0	0.76	0.69	0.72	166
1	0.95	0.96	0.96	987
accuracy			0.92	1153
macro avg	0.85	0.83	0.84	1153
weighted avg	0.92	0.92	0.92	1153

3.3. Algorithms on statistical characteristics

Basic statistics were calculated for each characteristic of the data:

- **Average**(Mean)
- **Maximum**And**Minimum**
- **Standard deviation**
- **median**
- **amplitude**
- **variance**
- **first, third quartile**

4. Data Compression via Autoencoders

The goal of this step is to reduce the dimensionality of EEG data while preserving the essential information needed to distinguish Alzheimer's patients from healthy patients. Autoencoders are used to encode and compress the data, and then reconstruct the data in a way that minimizes information loss.

4.1. Using Autoencoders

Autoencoders are neural networks used to learn a compressed representation of input data. They mainly consist of two parts:

7.

* **Encoder**which reduces the dimensionality of the data.

* D**decoder**which reconstructs the data from the compressed representation.

In our work we used 3 types of encoders which are DAE, SPARSE, DEEP

has. Autoencoder Denoising (DAE)

Autoencoder Denoising is a variant of autoencoder that adds noise to the input data during training. The goal is to force the model to learn robust features by removing this noise during reconstruction. [Reference[1]]

Steps:

- **Adding Gaussian Noise**:Gaussian noise is added to the EEG signals to simulate noisy data.
- **Model training**:The model is trained to reconstruct the original data from the noisy data.
- **Compression**:The encoder learns a noise-free compressed version of the data, allowing for better representation of important features.

b. Sparse Autoencoder

A Sparse Autoencoder introduces a sparsity constraint in the hidden layer of the network, which forces the model to activate a limited number of neurons when representing the data. This results in a more concise and interpretable representation. [Reference[2]]

Steps:

- **Application of sparsity constraints**:Using regularizations to limit the number of activated neurons.
- **Dimensionality Reduction**:The model encodes data using a minimal number of neurons, allowing the most important features to be captured with fewer units.

- **Data reconstruction**:The decoder reconstructs the original data from the compressed vectors, minimizing information loss.

c. Deep Autoencoder

Deep Autoencoders are networks with multiple hidden layers in the encoder and decoder. They allow for finer compression and capture more complex features, which is particularly useful for complex EEG data. [Reference[3]]

7.

Steps:

- **Building a Multi-Layer Network**:The model contains multiple hidden layers for deeper feature extraction.
- **Progressive compression**:At each layer, the data dimension is reduced until reaching the code layer, which represents the most compressed version.
- **Progressive reconstruction**:The decoder reconstructs the data layer by layer, working backwards until the final output is reached, which should match the input data.

4.2. Creating Models from Compressed Data

Once the data is compressed by the autoencoders, it is used to train different machine learning models. This approach allows to evaluate whether the compression has improved the quality and accuracy of the models without losing critical information. [Reference4]

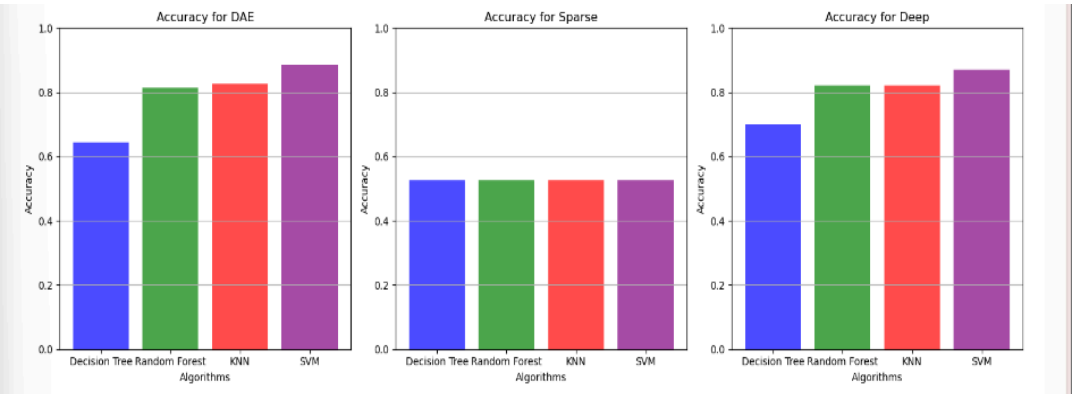
Steps:

- **Compressed Feature Extraction**:The original input data is passed through the encoder to obtain a compressed version.
- **Model training**:Using this compressed data to train classification algorithms such as:
    - **Decision Tree**
    - **Random Forest**
    - **KNN**
    - **SVM**
- **Assessment**:Comparison of model performance on compressed data with that on raw and statistical data.

4.3. Statistics on Compressed Data

After applying the autoencoders, statistics are calculated on the compressed data to check its quality and suitability for training models. This includes measures such as data distribution, feature correlations, and impact on algorithm performance. [Reference5]

7.



Analysis of curves:

The figure shows the performance of four classification algorithms (decision tree, random forest, KNN, SVM) on three different datasets. The DAE dataset performed the best overall, followed by the Deep dataset, while the Sparse dataset performed the worst overall. Random forest appears to be the best performing algorithm for all datasets.

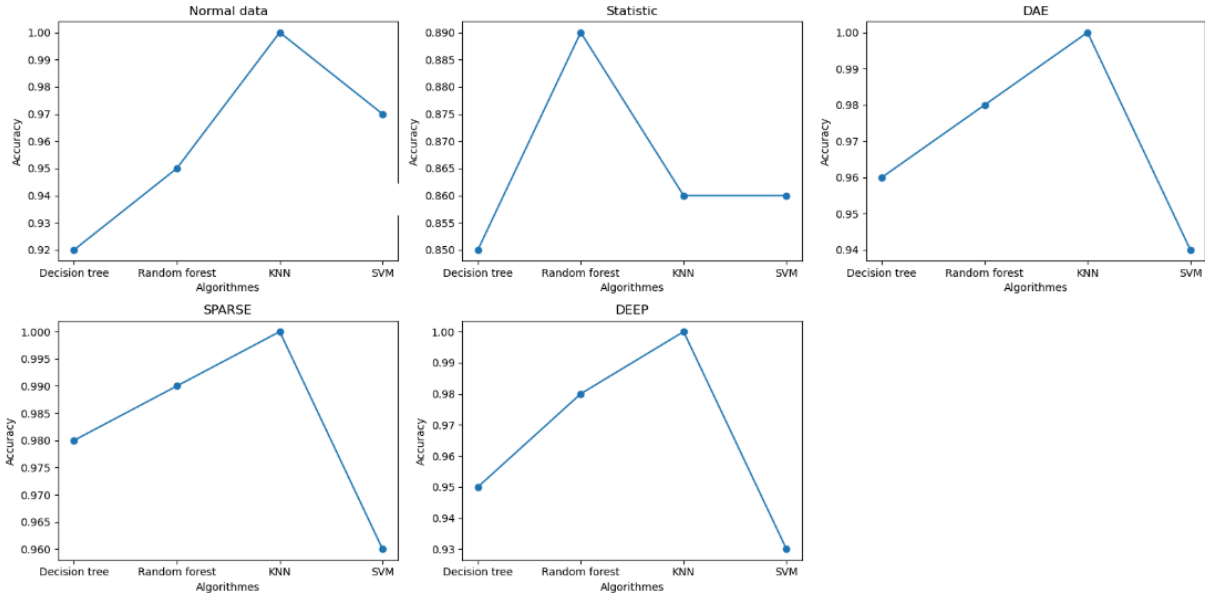
5. Results and Comparison of Models

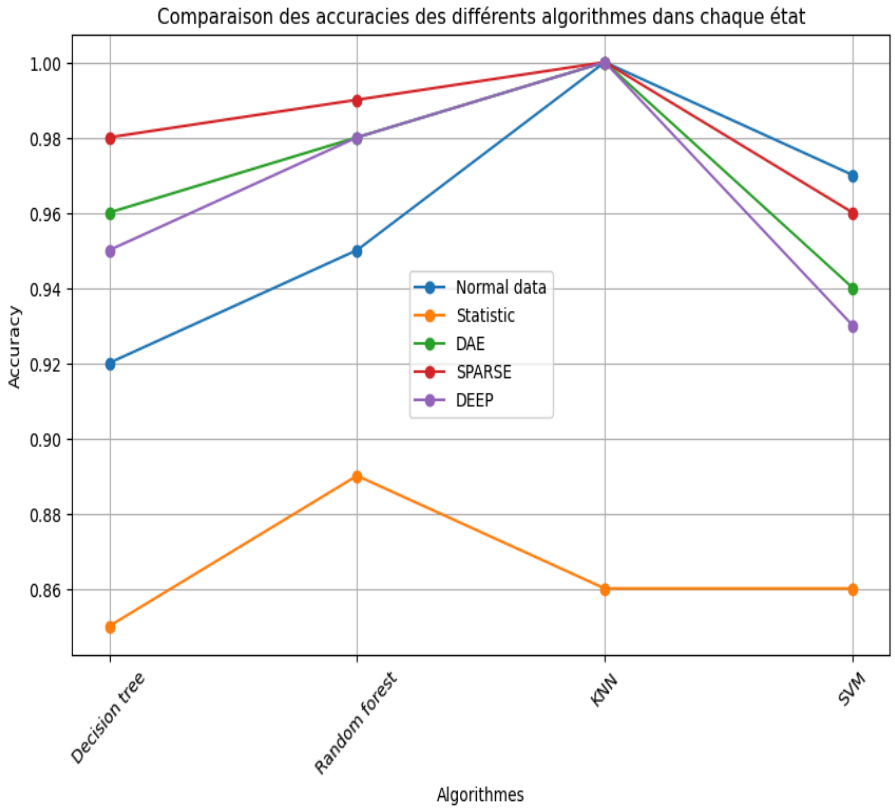
5.1. Performance Summary Table

Algorithm	Data Brutes	Data Statistics	DAE	SPARSE	DEEP
Decision Tree	0.92	0.85	0.96	0.98	0.95
Random Forest	0.95	0.89	0.98	0.99	0.98
KNN	1	0.86	1	1	1
SVM	0.97	0.86	0.94	0.96	0.93

5.2. Graphical representation of the results found

7.





Best Algorithm

- **Sparse Autoencoder**gave the best results, in terms of accuracy and generalization.

7.

- **Statistical Analysis**showed the worst results, probably due to the loss of information due to the reduction in data complexity.

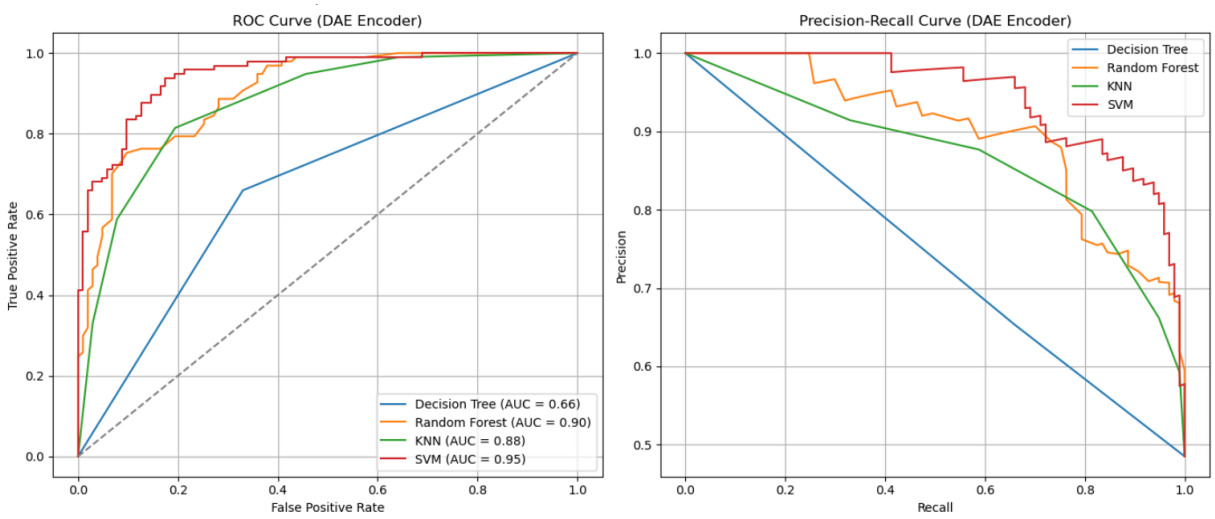
6. Performance Visualization

6.1. Performance Curves

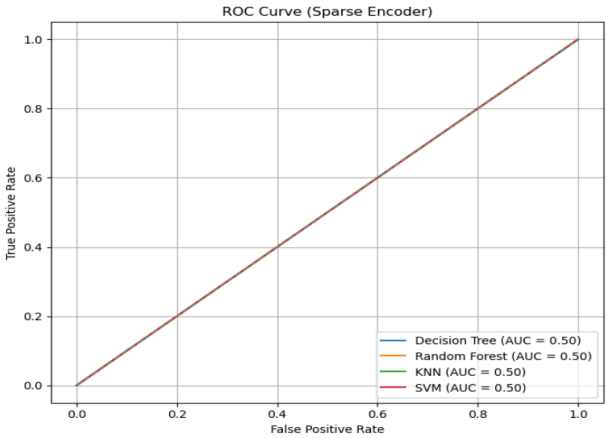
The following curves were plotted for each algorithm:

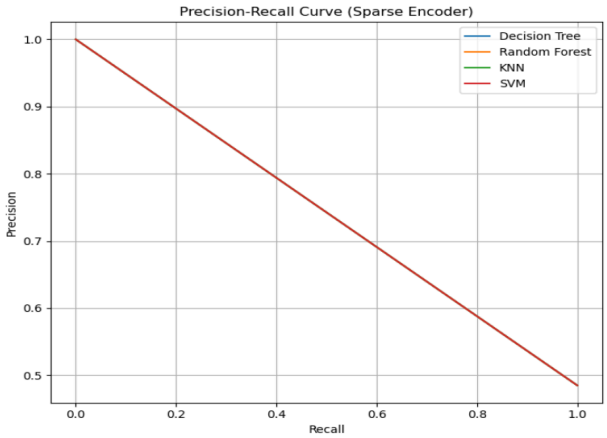
ROC curveto measure classification performance.

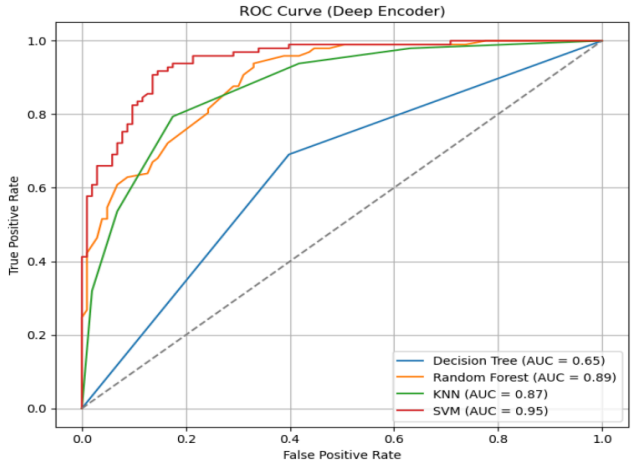
Precision-Recall Curvesto assess precision and recall.

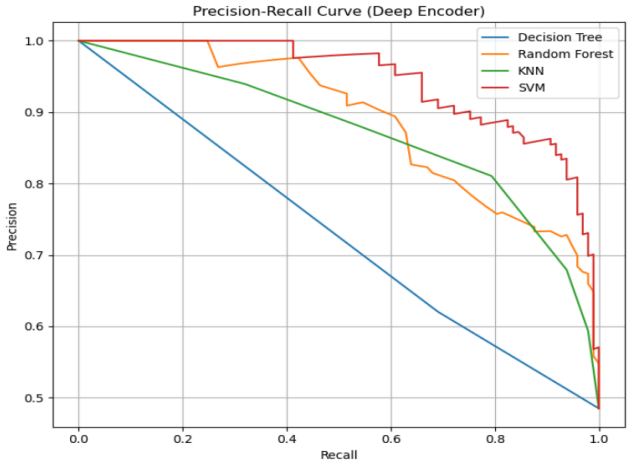


7.









7. Conclusion

- Application of machine learning algorithms on raw data showed acceptable performance.
- Data compression via autoencoders has made it possible to maintain or even improve accuracy while reducing dimensionality.
- The results show that**Sparse Autoencoder**is the most robust approach, providing a good balance between accuracy and efficiency.
- Simple statistical models lack critical information, leading to poor results.

7.

- The use of advanced machine learning techniques, such as autoencoders, is proving to be an effective approach for EEG analysis in the detection of Alzheimer's disease.

References

1. Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P.-A. (2008). “Extracting and Composing Robust Features with Denoising Autoencoders.” In *Proceedings of the 25th International Conference on Machine Learning*.↩
2. Ng, A. (2011). “Sparse Autoencoder.” CS294A Lecture Notes, Stanford University.↩
3. Hinton, GE, & Salakhutdinov, RR (2006). “Reducing the Dimensionality of Data with Neural Networks.” *Science*, 313(5786), 504–507.↩
4. Bengio, Y. (2009). “Learning Deep Architectures for AI.” *Foundations and Trends® in Machine Learning*, 2(1), 1–127.
5. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

7.

7.