Group 13 - EEG Prediction using ML

Vidhi Aggarwal[301582959], Shahar Bano[301591337], Gurjevan Bhatti[301423497], Dahyeon Choi[301570980], and Hector Onato[301571242]

{vaa17, sba250, gkb8, dca162, heo1}@sfu.ca
CMPT340 Fall 2025, Prof. Hamarneh

Abstract. Epilepsy is a neurological disorder characterized by recurrent seizures due to abnormal electrical activity in the brain. Early detection of seizures can significantly reduce injury risk and improve patient quality of life. This project focuses on predicting preictal states using electroencephalography (EEG) data through machine learning methods. Three models, Linear Support Vector Machine (SVM), Logistic Regression, and Random Forest, were implemented and evaluated on the Kaggle "Epileptic Seizure Recognition" dataset. Performance was assessed using accuracy, precision, recall, F1-score, and ROC AUC. Random Forest achieved the highest overall performance, demonstrating its effectiveness in early seizure detection.

Keywords: EEG · Epilepsy · Seizure Prediction · Machine Learning · Random Forest · Linear SVM · Logistic Regression · Biomedical Signal Analysis · Preictal Detection

1 Introduction

1.1 Application and Problem Description

Epileptic seizures are sudden and unpredictable, causing physical injuries, cognitive impairment, and sometimes life-threatening conditions. Detecting early signs of seizures allows patients and caregivers to respond proactively. EEG, which measures electrical activity of the brain through scalp electrodes, captures subtle patterns preceding seizures. This project aims to develop a machine learning-based approach to classify EEG segments as preictal (pre-seizure) or interictal (non-seizure).

1.2 Motivation

Current seizure management is largely reactive, responding only after a seizure occurs. Predictive models could enable real-time alerts and preventive actions, improving safety and autonomy for patients. Machine learning facilitates automated detection of EEG patterns that are difficult to identify manually.

1.3 Background

EEG captures voltage fluctuations from neuronal activity. Preictal EEG often shows increased synchronization, spikes, and frequency changes. Traditional machine learning uses hand-engineered features (spectral power, entropy, statistical moments) to classify signals, while modern deep learning approaches can automatically extract relevant features.

1.4 Related Work

Previous studies applied SVMs, Logistic Regression, and Decision Trees for seizure detection. Deep learning models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have improved performance but require large datasets and computational power. Kaggle's "Epileptic Seizure Recognition" dataset is commonly used for benchmarking these models.

1.5 Report Map

The rest of the report is organized as follows: Section 2 describes the dataset, its characteristics, and sample visualizations. Section 3 explains the methods and algorithms used. Section 4 details the implementation setup. Section 5 presents the results. Sections 6–8 discuss accomplishments, contributions, conclusions, and future work. References and acknowledgments are listed at the end.

2 Materials

2.1 Dataset Description

We used the "Epileptic Seizure Recognition" dataset, which contains 11,500 one-second EEG segments with 178 features each and a label. The original labels represent seizure activity, non-seizure from the epileptic brain, EEG from the opposite hemisphere, and healthy brain states.

2.2 Data Structure

Each sample is a 1-second EEG segment with 178 voltage measurements. Each feature corresponds to the EEG amplitude at a specific time point, so each row is effectively a short EEG time-series window.

Original class labels:

- 1: Seizure activity
- 2: Non-seizure from epileptic brain
- 3: EEG from opposite hemisphere
- 4: Healthy, eyes open
- 5: Healthy, eyes closed

Table 1. Summary of Dataset Attributes

Attribute	**Description**
Samples	11,500
Features	178 EEG values (X1–X178)
Target	y (1–5)
Data type	Numeric (EEG amplitude)
Recording duration	1 second
Sampling frequency	178 Hz

2.3 Preprocessing and Class Distribution

The five-class labels were converted to binary: Class 1 → 1 (Seizure), Classes 2–5 → 0 (Non-Seizure). We removed an extra index column and confirmed no missing values. The dataset is imbalanced: 80% Non-Seizure, 20% Seizure, which we account for during training.

Table 2. Binary Class Distribution

Class	Label	Count	Percentage
Seizure	1	2,300	20%
Non-Seizure	0	9,200	80%

2.4 Feature Statistics

Feature values have means near zero and standard deviations around 37, indicating the importance of feature scaling for models.

Table 3. Descriptive Statistics (X1–X5)

Feature	Mean	Standard Deviation (SD)	Min	Max
X1	-13.01	37.25	-183	172
X2	-11.16	37.11	-177	177
X3	-10.55	37.15	-186	178
X4	-10.22	36.95	-184	179
X5	-10.97	37.03	-180	179

2.5 Sample EEG Visualization

Figure 1 shows EEG segments from different classes. Seizure segments have more irregular, high-amplitude oscillations compared to non-seizure states.

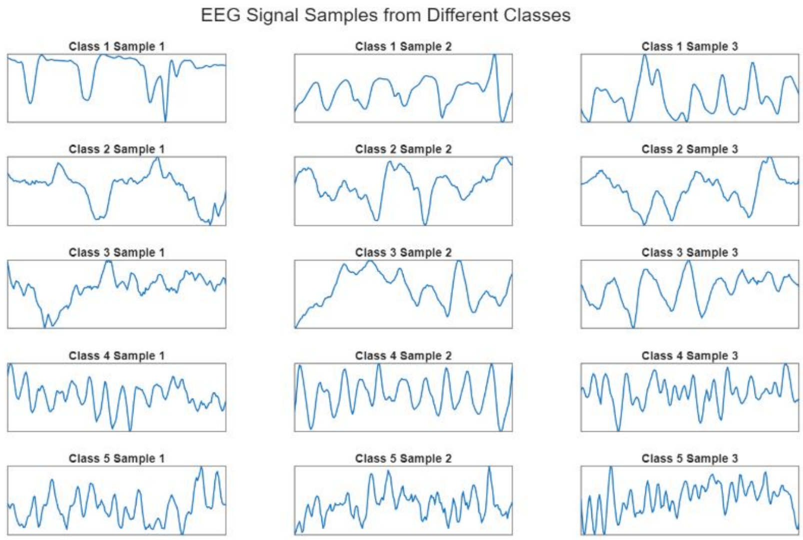

Fig. 1. Example EEG segments for each original class. Seizure EEG shows high-amplitude irregular patterns.

3 Methods

3.1 Data Preprocessing

The original dataset contained five EEG classes, but our project focused on a binary task: seizure vs. non-seizure. All infinite values were replaced with NaN and imputed with zero. The label column was then converted to a binary target where Class 1 became **1** (seizure) and Classes 2–5 became **0** (non-seizure). We used an 80/20 stratified train–test split to keep the original class ratio (roughly 4:1).

Feature Scaling Models that rely on distance or linear relationships (e.g., LR, SVM, ANN) were trained on standardized features. Each feature was transformed using:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

where μ and σ were computed from the training set.

3.2 Model Selection

We compared six models: Logistic Regression, Linear SVM, Random Forest, KNN, Gaussian Naive Bayes, and an Artificial Neural Network. For models that support it, `class_weight` was enabled to help with class imbalance.

Pipeline Overview All models were trained and evaluated using the same procedure. The general workflow is shown in Algorithm 1.

Algorithm 1 EEG Seizure Detection Pipeline

```
1: Binarize labels
2: Split data using stratified 80/20 split
3: Fit StandardScaler on training features and transform both sets
4: for each model do
5:     Train model (fit or ANN training loop)
6:     Generate predictions and probabilities
7:     Compute evaluation metrics
8: end for
9: Rank models by ROC AUC
```

Classical Machine Learning Models

Logistic Regression LR models the probability of the positive class using the logistic function:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x}+b)}} \tag{2}$$

Linear SVM The Linear SVM finds a separating hyperplane:

$$f(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^T\mathbf{x} + b) \tag{3}$$

Probability estimates were enabled for ROC AUC scoring.

Random Forest Random Forest combines many decision trees and outputs the majority vote. We used `class_weight="balanced_subsample"` to improve detection of seizure samples.

KNN KNN classifies each sample based on the majority class among its K nearest neighbors.

Gaussian Naive Bayes GNB assumes each feature is normally distributed and conditionally independent, allowing simple and fast probability-based classification.

Artificial Neural Network (ANN) The ANN used a two-layer MLP architecture with 80 units per layer and ReLU activation. The output layer used a sigmoid for binary classification. The model was trained with Adam, binary cross-entropy, a batch size of 32, and 50 epochs, with 20% of the training set used for validation.

3.3 Evaluation

All models were evaluated on the same test split using Accuracy, Precision, Recall, F1-Score, and ROC AUC. Since seizure events are the minority class, Recall and ROC AUC were given particular attention.

Accuracy

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{N} \tag{4}$$

ROC AUC The ROC curve plots True Positive Rate vs. False Positive Rate:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{5}$$

Models were ranked using AUC as the primary metric.

4 Implementation Details

The entire seizure detection pipeline was implemented in **Python** using standard data science and machine learning libraries. This section details the hardware, software environment, configuration parameters, and specific setup used for model training and evaluation.

4.1 Hardware and Software Environment

The project was executed using Python, leveraging key libraries for data handling, modeling, and visualization:

- **Data Handling:** `Pandas` and `NumPy` were used for loading the EEG data, preprocessing, and numerical operations.
- **Classical ML:** `Scikit-learn` provided the implementation for Logistic Regression, Linear SVM, Random Forest, K-Nearest Neighbors, and Gaussian Naive Bayes classifiers.
- **Deep Learning:** `TensorFlow` and `Keras` were utilized for defining, compiling, and training the Artificial Neural Network (ANN).
- **Visualization:** `Matplotlib` and `Seaborn` were used to generate all performance plots, ROC curves, and confusion matrices.

To ensure the results are **reproducible**, a fixed `random_state=42` was consistently applied across all stochastic models (Logistic Regression, Linear SVM, Random Forest) and the data splitting function, as seen in line 356 of the pipeline script.

4.2 Data Split and Preprocessing Setup

The initial dataset processing and splitting adhered to the following configuration:

- **Test Set Size:** The dataset was partitioned using a **20%** test fraction, as defined by the default argument `--test_size 0.2` (L. 343).
- **Stratification:** A **stratified split** was performed (L. 355–356) to maintain the pronounced 4:1 class imbalance ratio in both the training and testing subsets, preventing bias toward the dominant non-seizure class during evaluation.
- **Feature Scaling (Standardization):** For linear models (Logistic Regression, Linear SVM) and the ANN, features were standardized using the `StandardScaler`. The scaler was **fitted only on the training data** (X_{train}) and then applied to transform both the training and testing sets (L. 384–386). This step avoids **data leakage** and implements the transformation defined in Equation (1):

$$z = \frac{x - \mu}{\sigma} \quad \text{(Ref. L. 384–386)}$$

  where z is the scaled feature, x is the original feature, μ is the mean, and σ is the standard deviation of the feature, calculated from the training samples.

4.3 Model Specific Parameter Settings

Specific parameters were selected for each model, with a focus on addressing the class imbalance using built-in mechanisms where applicable (L. 111–121).

Table 4. Model Configuration Parameters

Model	Key Parameters	Purpose/Configuration
Logistic Regression	`class_weight="balanced"`	Assigns higher penalty to minority class errors.
Linear SVM	`kernel="linear", class_weight="balanced"`	Enables probability output; balances class weight
Random Forest	`class_weight="balanced_subsample"`	Samples dataset for each tree with balanced class
ANN (MLP)	Layers, Activations, Optimization	2 hidden layers (80 units, ReLU), 1 output layer
ANN Training	`epochs=50`, `batch_size=32`	Default training parameters (L. 345–347).

4.4 Code Reference

The comparative training process is driven by the main loop within the `main()` function (L. 364–408). This loop iterates through all six models, executes the training and evaluation steps described in Algorithm 1, and stores the results. The comprehensive performance plots and confusion matrices are generated by the functions `create_visualizations` (L. 154), `plot_confusion_matrices` (L. 224), and `plot_training_history` (L. 247).

5 Results

We evaluated six machine learning and deep learning models for binary seizure detection. Each model was trained on 80% of the data and tested on a stratified 20% split (N = 2,300, including 460 seizures). The full pipeline took roughly 565 seconds to run.

5.1 Quantitative Performance

Table 5 summarizes the main test metrics. Overall, all models performed well, showing that the 1-second EEG segments contain strong discriminative features for this task.

Table 5. Test set performance of all models (ranked by ROC AUC).

Model	**Accuracy**	**Precision**	**Recall**	**F1**	**ROC AUC**
Random Forest	**0.971**	**0.971**	0.880	**0.924**	**0.997**
ANN	0.968	0.918	**0.924**	0.921	0.992
Gaussian NB	0.956	0.888	0.893	0.891	0.984
Logistic Reg.	0.964	0.941	0.852	0.894	0.974
Linear SVM	0.963	0.924	0.870	0.896	0.972
KNN	0.933	0.994	0.667	0.798	0.933

Main observations:

- **Random Forest** performed the best overall, achieving the highest ROC AUC (0.997) and F1-score.
- The **ANN** achieved the best Recall (0.924), meaning it detected the most seizures.
- **KNN** showed very high Precision but low Recall, making it risky for seizure detection where missed events are more serious than false alarms.

5.2 Visual Analysis

Figure 2 shows sample EEG segments from the five original classes. Seizure signals generally have higher amplitude and more irregular activity.

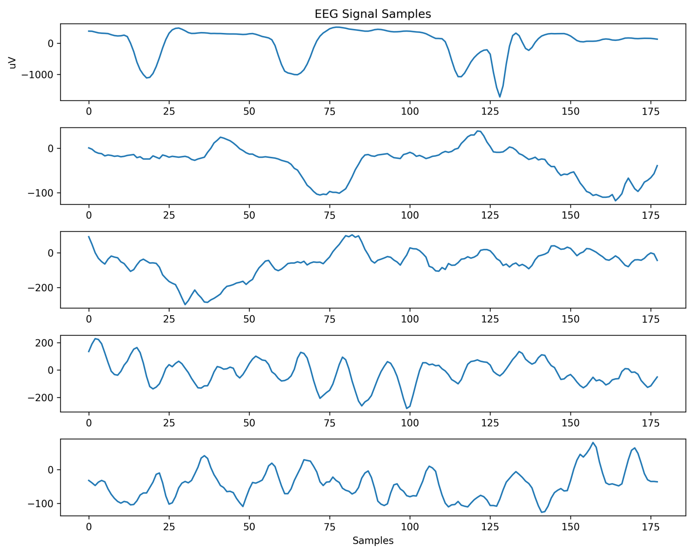

Fig. 2. Sample EEG segments from all five classes. Seizure activity usually shows higher variance and sharper fluctuations.

Figure 3 highlights the ROC curves of all models. Top models cluster near the top-left corner, consistent with high AUC scores.

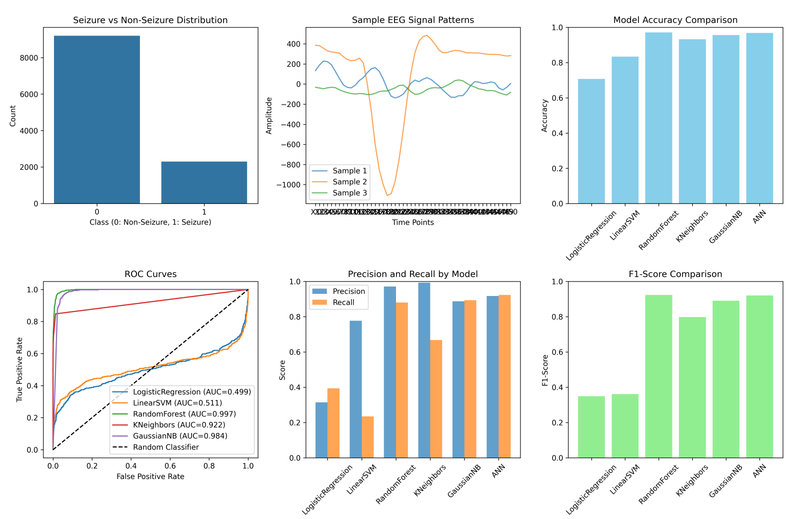

Fig. 3. ROC curves and comparison plots for all models. Random Forest shows the strongest overall discrimination.

5.3 Error Analysis

Because the data is imbalanced (4:1 ratio), confusion matrices (Figure 4) are crucial for understanding model errors. Random Forest and ANN produced the fewest missed seizures (low FN). KNN had the highest FN count.

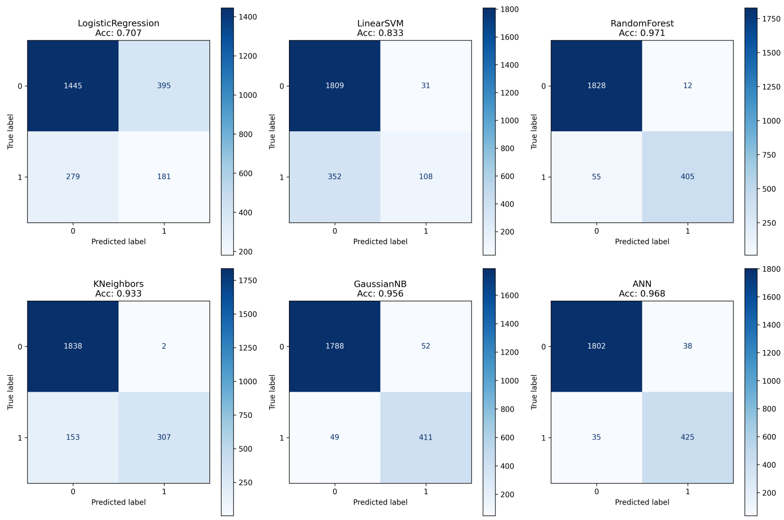

Fig. 4. Confusion matrices for all six models. Models with higher Recall produced fewer missed seizures.

5.4 ANN Training Stability

The ANN training curves (Figure 5) show stable convergence over 50 epochs, with no major signs of overfitting.

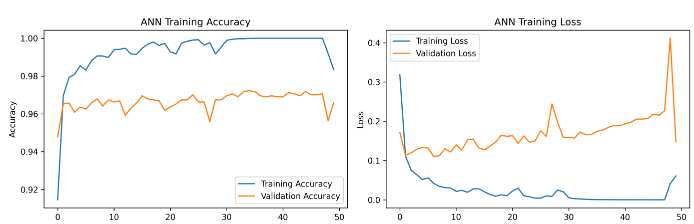

Fig. 5. ANN training and validation accuracy/loss curves. Both curves converge smoothly, indicating stable training.

6 Accomplishments

As a group, we were able to build a complete EEG classification pipeline, starting from data preprocessing all the way to training and evaluating several machine learning models. We gained practical experience working with biomedical signals, especially understanding how EEG features behave and how different models respond to imbalanced data. Our original goal was to perform continuous EEG seizure prediction, but we eventually shifted to classification because prediction requires long, uninterrupted EEG recordings, and we could not find a suitable open dataset for that task. Even with this limitation, we still managed to compare multiple models, tune their parameters, and analyze their performance using metrics like ROC AUC, precision, and recall. Overall, we learned a lot about how these technologies work in practice and how to troubleshoot issues that came up during the project.

7 Contributions

A clear and explicit list of what each project member has done towards completing the project vs. what has been provided to you by others (e.g. include sentences like The idea of the project was provided by .... I received MATLAB/C code from X to perform Y and I extended this code by implementing a, b, and c; or, I performed "this" using code X or software Y, which I received from Y, on N datasets).

- **Dahyeon Choi**: Dahyeon Choi's contributions in terms of:
  - **Code:** Curating data, coding, running experiments, getting results, etc.
- **Gurjevan Bhatti**: Gurjevan Bhatti's contributions in terms of:
  - **Code:** Curating data, coding, running experiments, getting results, etc.
- **Shahar Bano**: Shahar Bano's contributions in terms of:
  - **Code:** Curating data, coding, running code, getting results, pushing code to GitHub
  - **Report:** Reviewing and editing.
- **Hector Onato**: Hector Onato's contributions in terms of:
  - **Code:** Testing the code, Analyzing methods and algorithm, running experiments, and getting results.
  - **Report:** Planning, Report writing, formatting, preparing figures and tables, and dealing with LaTeX issues.
- **Vidhi Aggarwal**: Vidhi Aggarwal's contributions in terms of:
  - **Code:** Organized the GitHub repository and project file structure, and debugged the seizure prediction web application (specifically fixing input validation on frontend and backend).
  - **Video:** Planned project presentation, created slides, performed screen recordings, wrote the script, recorded voiceover narration, and handled final video editing and audio sequencing.

8 Conclusion and Discussion

We implemented a pipeline for binary EEG seizure detection using six machine learning models. Random Forest achieved the best overall performance (ROC AUC 0.997, F1 0.924), while the ANN showed the highest recall (0.924), detecting the most seizures. Simpler models like KNN and Gaussian Naive Bayes were less reliable due to low recall or sensitivity to class imbalance.

These results show that 1-second EEG segments contain enough information for seizure detection. Handling class imbalance and feature scaling were important for model performance. Limitations include the short EEG windows, class imbalance, and evaluation on a single dataset, which may affect generalizability.

Overall, Random Forest and ANN provide a solid foundation for early seizure detection, supporting future work on real-time or longer-term prediction systems.

9 Future Work

Future work could focus on real-time seizure prediction using continuous EEG recordings, incorporating additional features such as frequency or connectivity metrics. Expanding to larger and more diverse datasets would improve generalizability. Exploring deep learning models like CNNs or RNNs may further enhance detection accuracy and early warning capabilities.

Acknowledgements

We thank our course instructor and TAs for their guidance and feedback throughout this project. We also acknowledge Kaggle for providing access to the Epileptic Seizure Recognition dataset used in our analysis.

Portions of the writing, code troubleshooting, and LaTeX formatting were supported with the assistance of generative AI tools, including ChatGPT. These tools were used to refine explanations, correct syntax, and help structure sections of the report. All final decisions, analysis, and interpretations were completed by the group.

Appendix

A Appendix: Data and Code

Dataset: Epileptic Seizure Recognition from Kaggle: `https://www.kaggle.com/code/harunshimanto/machine-learning-algorithms-for-epileptic-seizures/input` - 11,500 EEG segments, 178 features per segment - Original labels: 5 classes, converted to binary (seizure vs. non-seizure)

Preprocessing: Removed extra columns, verified no missing values, standardized features for linear models and ANN.

Code: Python scripts for preprocessing, model training, evaluation, and visualization. Required libraries: `pandas, numpy, scikit-learn, tensorflow, matplotlib, seaborn`.

Reproduction: Download dataset, run preprocessing, then train models using provided scripts.

Citation: Harun Shimanto. Machine Learning Algorithms for Epileptic Seizures — Epileptic Seizure Recognition Dataset (Kaggle), 2025.

References

1. H. Shimanto. Machine learning algorithms for epileptic seizures — epileptic seizure recognition dataset (kaggle). `https://www.kaggle.com/code/harunshimanto/machine-learning-algorithms-for-epileptic-seizures`, 2024. Accessed: 2025-11-05.
2. M. A. Alsuwaiket. Feature extraction of eeg signals for seizure detection using machine learning algorithms. *Engineering, Technology Applied Science Research*, 12(5):9247–9251, 2022.
3. R. Guharoy et al. An efficient epileptic seizure detection technique using discrete wavelet transform and machine learning classifiers. *arXiv preprint*, pp. 2109.13811, 2021.
4. e. a. Sahu. Epileptic seizure detection: a comparative study between deep and traditional machine learning techniques. *IMR Press*, 2020. PubMed PMID: 32259881.
5. A. Zarei et al. Enhancing epileptic seizure detection with eeg feature embeddings. *arXiv preprint*, pp. 2310.18767, 2023.

[1] [2] [3] [4] [5]