EEG Classification Model

IE6400 – Foundations Data Analytics Engineering

Final Report

Group Number: 04

Aadarsh Praveen Selvaraj Ajithakumari (002832667)

Aravind Swamy(002847684)

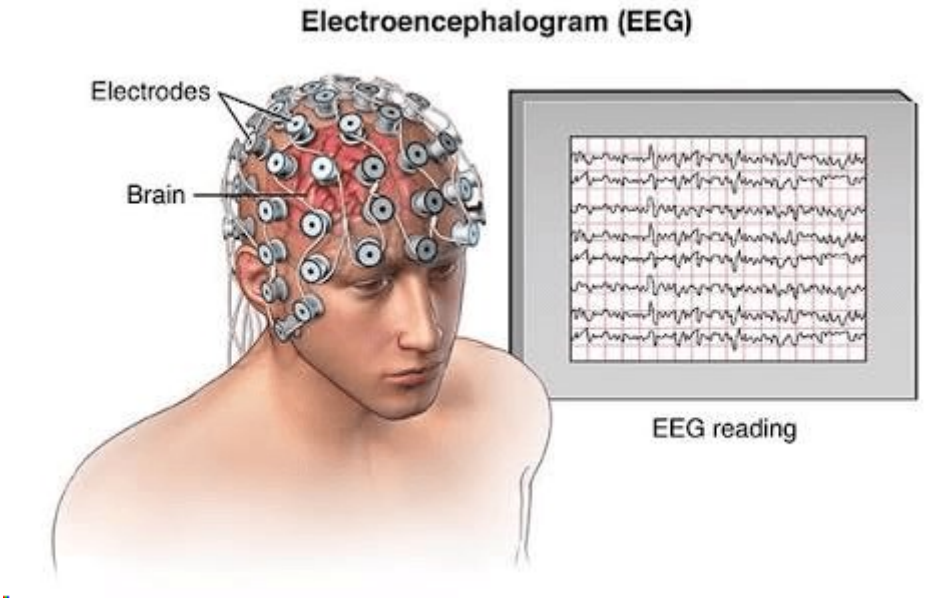
Moheesh Kavitha Arumugam (002296201)

Hashwanth Moorthy(002830971)

Uma Maheshwari Deivasigamani (002847743)

Introduction:

In the fields of neuroscience and medicine, electroencephalography (EEG) is a non-invasive method that is extremely useful for observing the dynamic activity of the human brain. The technique uses electrodes applied to the scalp to record electrical signals produced by brain neurons. The synchronised electrical activity of neural networks is represented by these signals, also referred to as brainwaves, which are essential to comprehend a variety of physiological and cognitive processes.



EEG data is now widely used in clinical and research settings, greatly advancing our knowledge of how the brain works, aiding in the diagnosis of neurological conditions, and enabling the tracking of brain health. Researchers and medical professionals have found EEG to be a useful tool for examining sleep patterns, cognitive functions, neurological conditions, and even mental states.

EEG is unique in that it can measure brain activity in real-time with millisecond accuracy due to its temporal resolution. It is possible to identify quick neural events and correlate them with internal cognitive processes or external stimuli thanks to this temporal precision. Furthermore, the non-invasive nature of EEG makes it an ideal option for observing brain activity in a variety of populations, including adults, children, and people with a range of medical conditions.

Recent technological developments and improvements in data analysis methods have raised the bar for EEG research. Artificial intelligence and machine learning

combined have made it possible to conduct more complex analyses and increased the precision of EEG-based applications. Creating models that can categorize EEG data into distinct groups is one use for these models.

Table of Contents:

Dataset:

The link for the dataset -> https://physionet.org/content/chbmit/1.0.0/

Reference/Citation : Guttag, J. (2010). CHB-MIT Scalp EEG Database (version 1.0.0). PhysioNet. https://doi.org/10.13026/C2K01R.

Abstract:

The CHB-MIT Scalp EEG Database, a collection of EEG recordings of 22 pediatric subjects with intractable seizures, is now available. Subjects were monitored for up to several days following withdrawal of anti-seizure medication to characterize seizures and assess their candidacy for surgical intervention. In all, the onsets and ends of 182 seizures are annotated.

This database, collected at the Children's Hospital Boston, consists of EEG recordings from pediatric subjects with intractable seizures. Subjects were monitored for up to several days following withdrawal of anti-seizure medication in order to characterize their seizures and assess their candidacy for surgical intervention. The recordings are grouped into 23 cases and were collected from 22 subjects (5 males, ages 3–22; and 17 females, ages 1.5–19).

Background:

Seizures are temporary deviations in the brain's electrical activity. Individuals with epilepsy, a disorder of the central nervous system, experience recurrent seizures that can happen unpredictably and often without any prior alert. These seizures may lead to a brief loss of attention or a full-body convulsion. Regular occurrence of seizures heightens the risk of physical injuries for the person and could potentially lead to death. A device that can swiftly detect and respond to a seizure by administering treatment or alerting a caregiver could help mitigate the challenges associated with seizures.

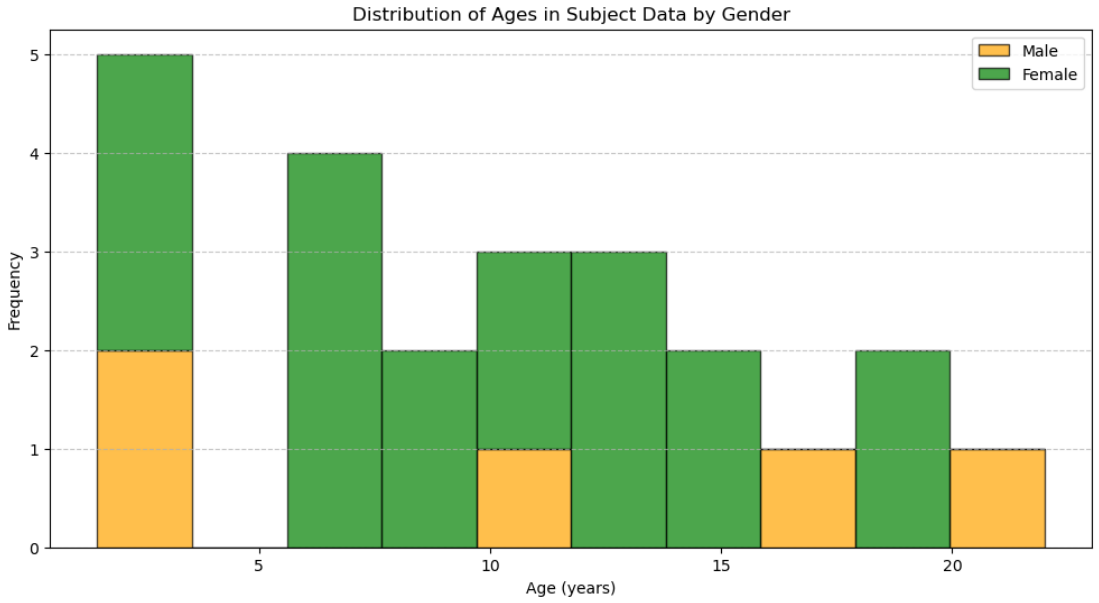
This database, collected at the Children's Hospital Boston, consists of EEG recordings from pediatric subjects with intractable seizures. Subjects were monitored for up to several days following withdrawal of anti-seizure medication in order to characterize their seizures and assess their candidacy for surgical intervention.

Data Preprocessing:

The data is taken from the website above and saved locally. Because the downloaded format is.zip. It's removed and saved in the same place.

	Case	Gender	Age (years)
0	chb01	F	11.0
1	chb02	M	11.0
2	chb03	F	14.0
3	chb04	M	22.0
4	chb05	F	7.0
5	chb06	F	1.5
6	chb07	F	14.5
7	chb08	M	3.5
8	chb09	F	10.0
9	chb10	M	3.0
10	chb11	F	12.0
11	chb12	F	2.0
12	chb13	F	3.0
13	chb14	F	9.0
14	chb15	M	16.0
15	chb16	F	7.0
16	chb17	F	12.0
17	chb18	F	18.0
18	chb19	F	19.0
19	chb20	F	6.0

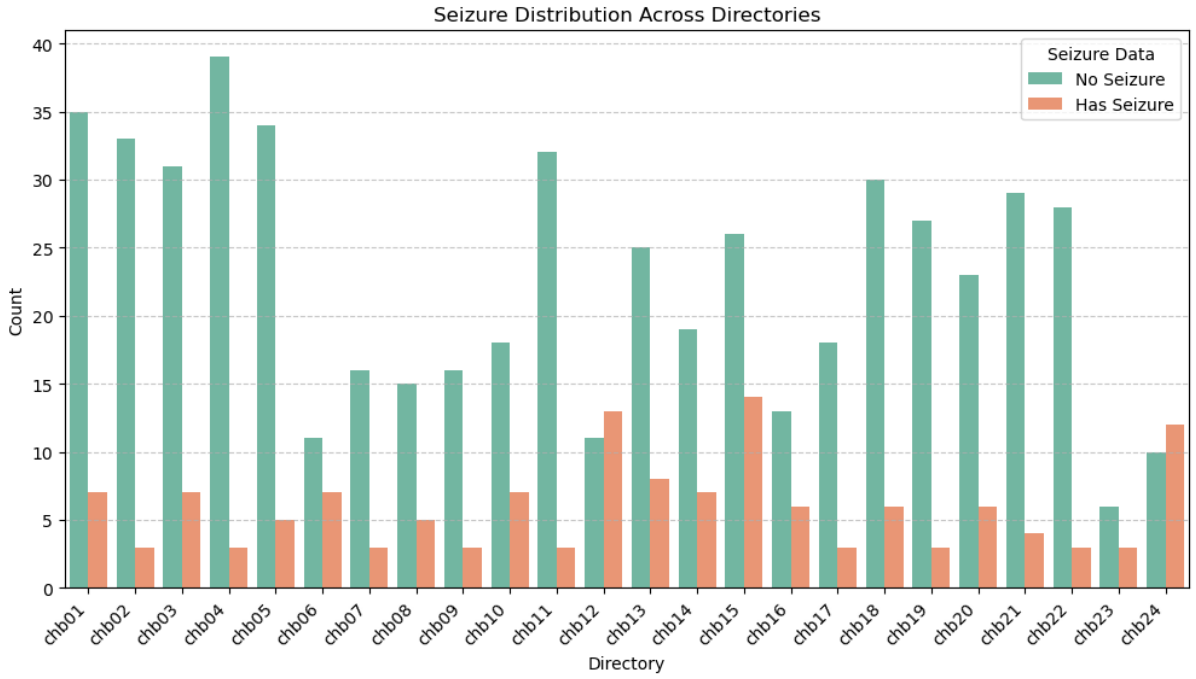
The code reads data from a file given by the variable subject_file_path by using the pandas library in Python. The delimiter '\t' indicates that the data should be in a tab-separated format. The pandas DataFrame subject_data is created using the read_csv function. This DataFrame most likely contains subject-related information, and the script prints a header labelled "Subject Details" to indicate what's to come. A thorough overview of the subject's data is provided by the displayed DataFrame, which also offers a structured representation of the data for additional study or investigation.



Using a histogram with ten bins, the chart shows the age distribution for each gender in different colours (green for females, orange for males). The age distributions of the male and female subjects can be easily compared thanks to the stacked bars. The resulting plot helps identify any noteworthy trends or gender differences by offering a brief visual insight into the age demographics within the subject data.

	Directory	File Name	Seizures Data Present
0	chb01	chb01_01.edf	No Seizure
1	chb01	chb01_02.edf	No Seizure
2	chb01	chb01_03.edf	Has Seizure
3	chb01	chb01_04.edf	Has Seizure
4	chb01	chb01_05.edf	No Seizure
...	...	...	...
681	chb24	chb24_18.edf	No Seizure
682	chb24	chb24_19.edf	No Seizure
683	chb24	chb24_20.edf	No Seizure
684	chb24	chb24_21.edf	Has Seizure
685	chb24	chb24_22.edf	No Seizure

The code segment that is provided creates pandas DataFrames (records_file and record_seizures_file) by processing two CSV files that contain file paths. Establishing a connection between the two DataFrames based on matching file names and directories, it takes the 'Directory' and 'File Name' columns out of the file paths. The records_file DataFrame then has a new column called "Seizures Data Present" that indicates whether or not seizure data is present in each file. A thorough summary of record file names, associated directories, and seizure status is given in the resulting DataFrame. This well-organized data provides insights into the presence or absence of seizures in each individual record, which is useful for additional analysis and decision-making.

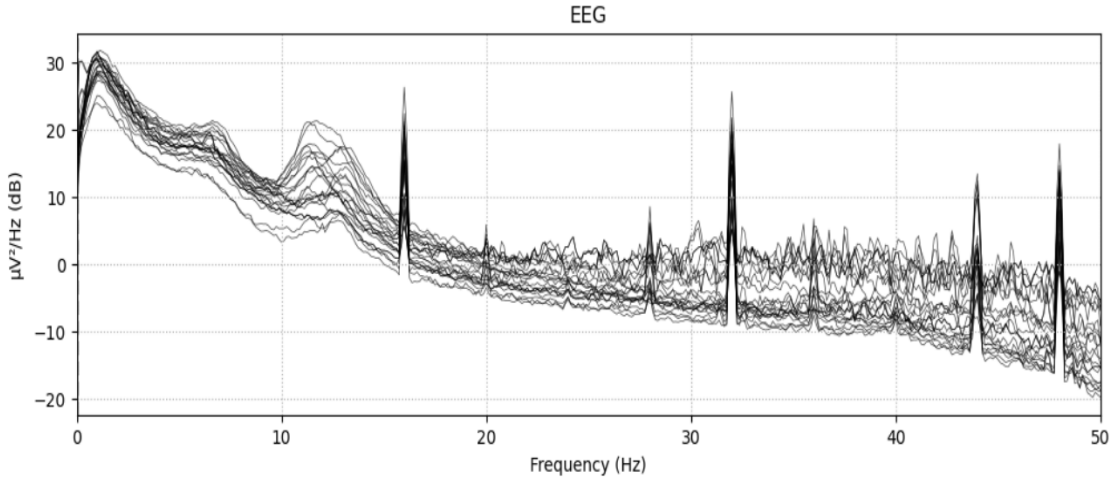


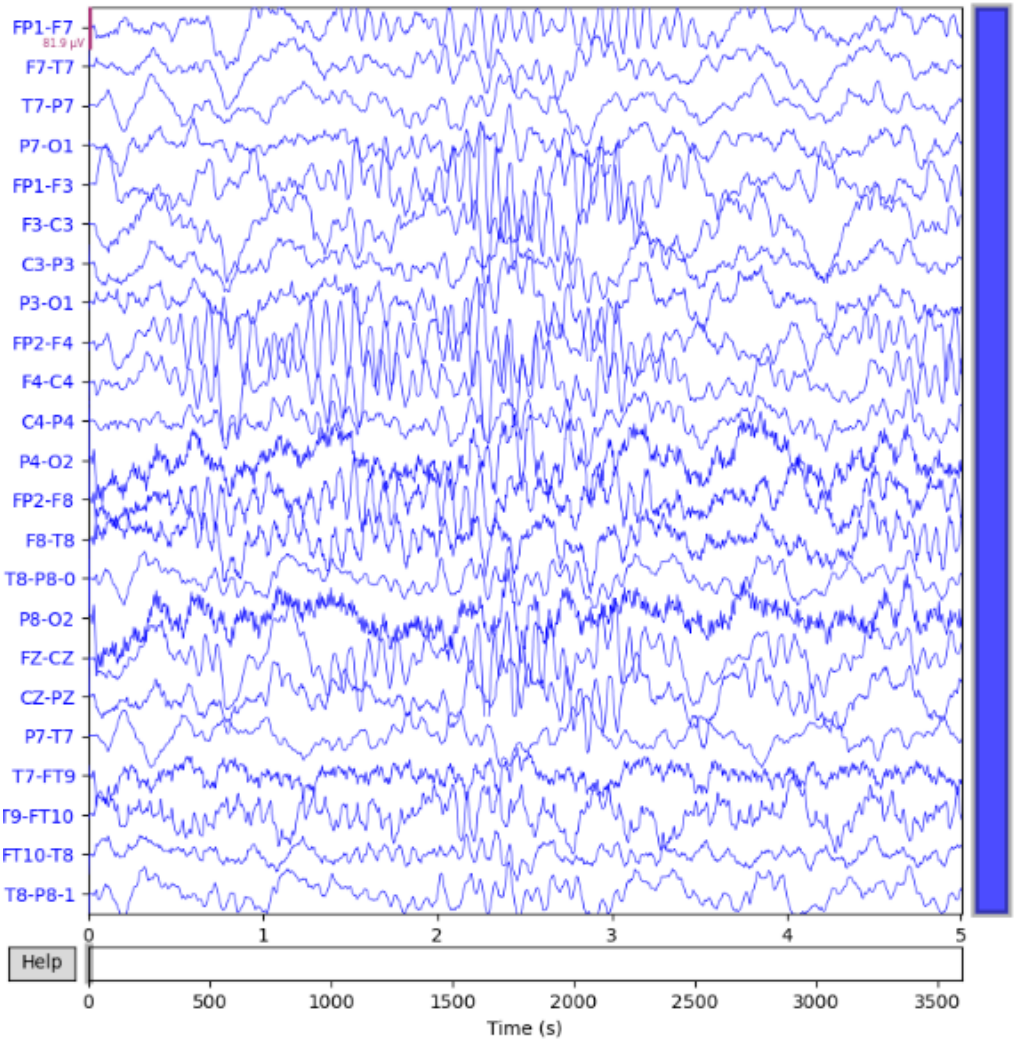
The diagram offers a concise visual summary that makes it easy to identify directories that have a greater number of seizure records. Understanding the patterns of seizure data distribution across different file directories is made easier with the help of this visualisation, which also suggests possible correlations or areas of interest for additional research. The addition of gridlines, axis labels, a title, and a legend improves the chart's readability and increases its usefulness as a tool for analysing and presenting seizure data distribution.

The code segment that is provided creates pandas DataFrames (records_file and record_seizures_file) by processing two CSV files that contain file paths. Establishing a

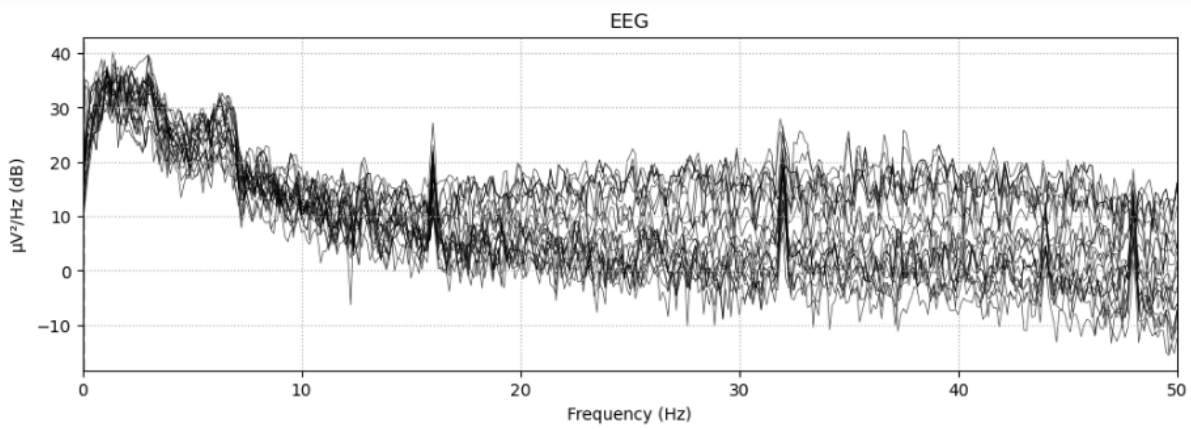
Now plotting only the region where seizure has taken place,

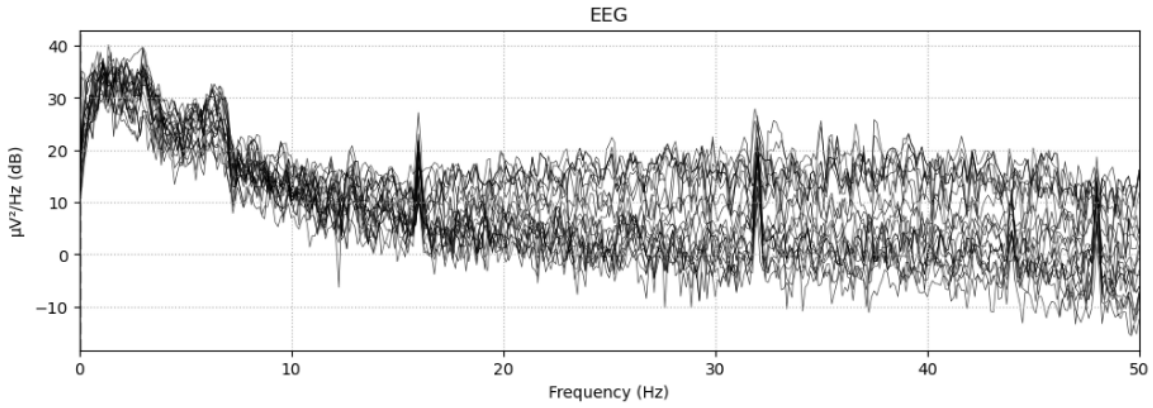
Power Spectral Density Plot:





For the above sample file the sample has seizure on seizure start time in 2996 seconds and seizure end time in 3036. Now plotting graphs for the specific frequency,





FEATURE EXTRACTION:

Time-Domain Feature Extraction

Time-domain feature extraction is essential in signal processing as it provides a succinct representation of a signal's characteristics in its raw temporal form. These features, such as mean, standard deviation, skewness, and kurtosis, offer interpretable insights into a signal's central tendency, variability, and distribution shape. Their computational efficiency makes them suitable for real-time applications and machine learning tasks, where quick and meaningful analyses are crucial. In domains like biomedical signal processing, time-domain features play a vital role in disease diagnosis and monitoring, offering valuable information about physiological signals.

Frequency-Domain Feature Extraction

Frequency domain feature extraction is vital for identifying distinct patterns and periodic behaviours in signals. It characterises signal dynamics, aids in noise analysis, and improves signal classification through spectral content. Particularly crucial in biomedical signal processing, it helps diagnose and monitor medical conditions. In communication systems, frequency domain analysis is essential for modulation, demodulation, and filtering operations. Overall, it provides a comprehensive understanding of signal characteristics, enabling efficient data compression and handling non-stationary signals.

The brief-Time Fourier Transform (STFT) is a time-frequency analysis technique that reveals a signal's frequency content across brief, overlapping time segments. STFT gives a time-varying representation of a signal's spectral content by applying the Fourier Transform to consecutive windows of a signal. This allows for the investigation of non-stationary signals and records how the frequencies of the signal fluctuate over time. STFT is commonly utilised in signal processing applications including audio and voice processing.

Models:

1. **Overview of Convolutional Neural Networks (CNNs):**

A particular kind of deep learning neural network called a CNN is made especially for handling and interpreting visual data. Numerous computer vision tasks, including image recognition and classification, have seen their successful application. CNNs, however, can also be modified to process sequential data, such as time-series, which makes them appropriate for handling EEG signals.

Important characteristics:

Convolutional Layers: These layers use convolution operations to find patterns in the input data that are spatial in nature.

Pooling Layers: These are used to reduce the input's spatial dimensions and downsample it.

Fully Connected Layers: To generate the final classification, connect the output from the preceding layers.

Application to EEG Analysis: Pattern recognition in EEG signals benefits from CNNs' ability to capture hierarchical features in data. They are an attractive option for EEG classification tasks because they can automatically extract pertinent spatial and temporal features from raw EEG data.

2. Overview of Long Short-Term Memory (LSTM):

Long Short-Term Memory (LSTM) networks are a type of RNN that can be used to model sequential data that has long-range dependencies. Because LSTMs have memory cells that can hold information for a long time, unlike traditional RNNs, they are appropriate for tasks involving time-series data.

Important characteristics:

Memory Cells: These allow the model to retain and lose knowledge over extended periods of time.

Gates: Control the information flow inside the network (Forget, Input, Output).

Application to EEG Analysis:

Since LSTMs are adept at modelling the intricate relationships between various time points, they are a good choice for capturing temporal dependencies in EEG data. They are therefore a good option for tasks where the timing and context of neural events are important.

3. Random Forest:

Overview: During training, Random Forest generates a large number of decision trees and outputs the mode of the classes for classification problems. It is an ensemble learning technique. Every decision tree in the forest adds to the final prediction after being trained on a random subset of the data.

Important characteristics:

Decision Trees: The fundamental components of the ensemble.

Ensemble learning improves performance by combining predictions from several models.

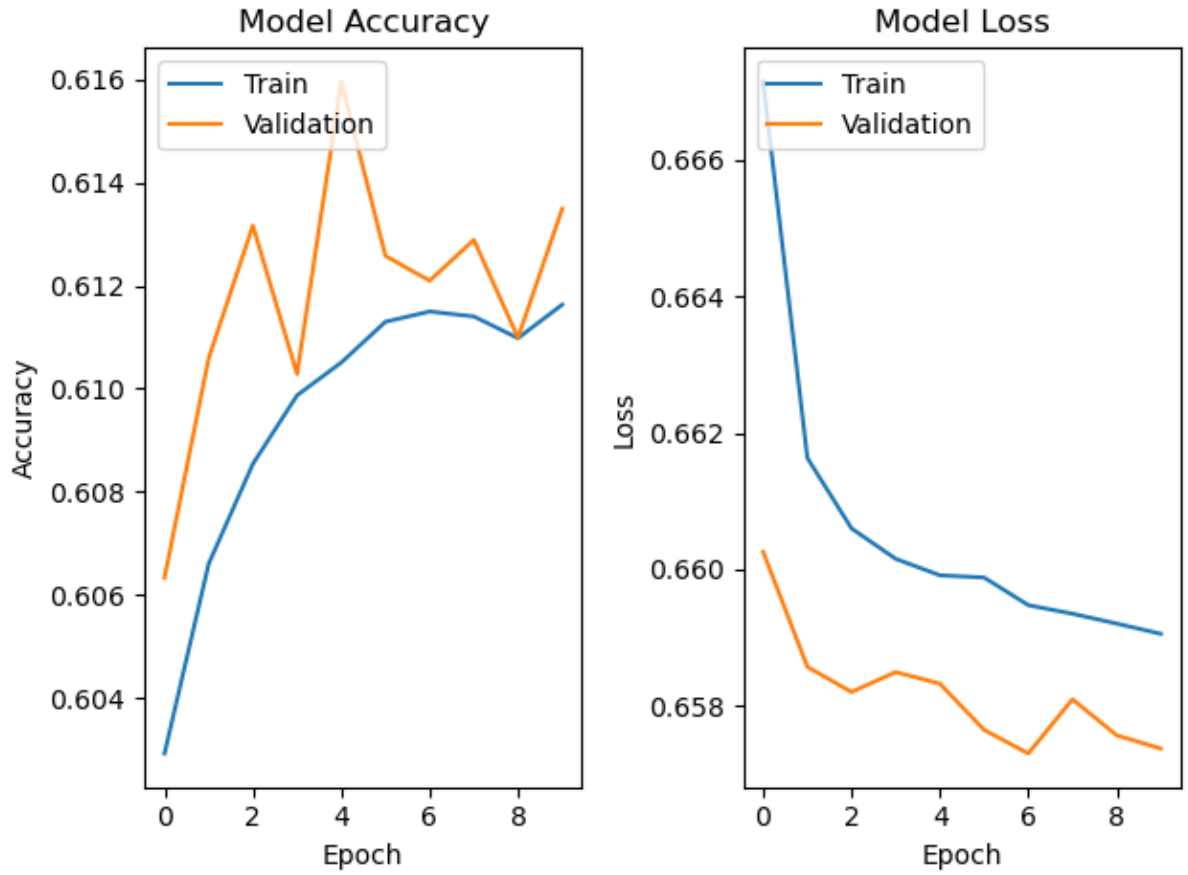
Applicability in EEG Analysis:

Random Forest is a versatile and robust model that can be effective in EEG classification tasks. It is less prone to overfitting and can handle a large number of features, making it suitable for scenarios where the EEG dataset has a high dimensionality.

Decision Tree Classifier:

In order to address the unequal distribution of classes, the provided Python script implements a decision tree classifier as part of a machine learning workflow. Initially, features and labels are combined to prepare the data, which is assumed to have been loaded via a commented-out function. Then, to lessen the impact of class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) is used. Thirty percent of the data is used for testing, and the remaining seventy percent is used for training. Using the resampled training data, a Decision Tree classifier is trained, and predictions are made using the test data. Accuracy and F1 score metrics are used to assess the model's performance, and the results are printed to the console. This script offers a thorough illustration of how to manage unbalanced data and train a classifier to generate predictions. Evaluation metrics are used to provide light on the model's efficacy.

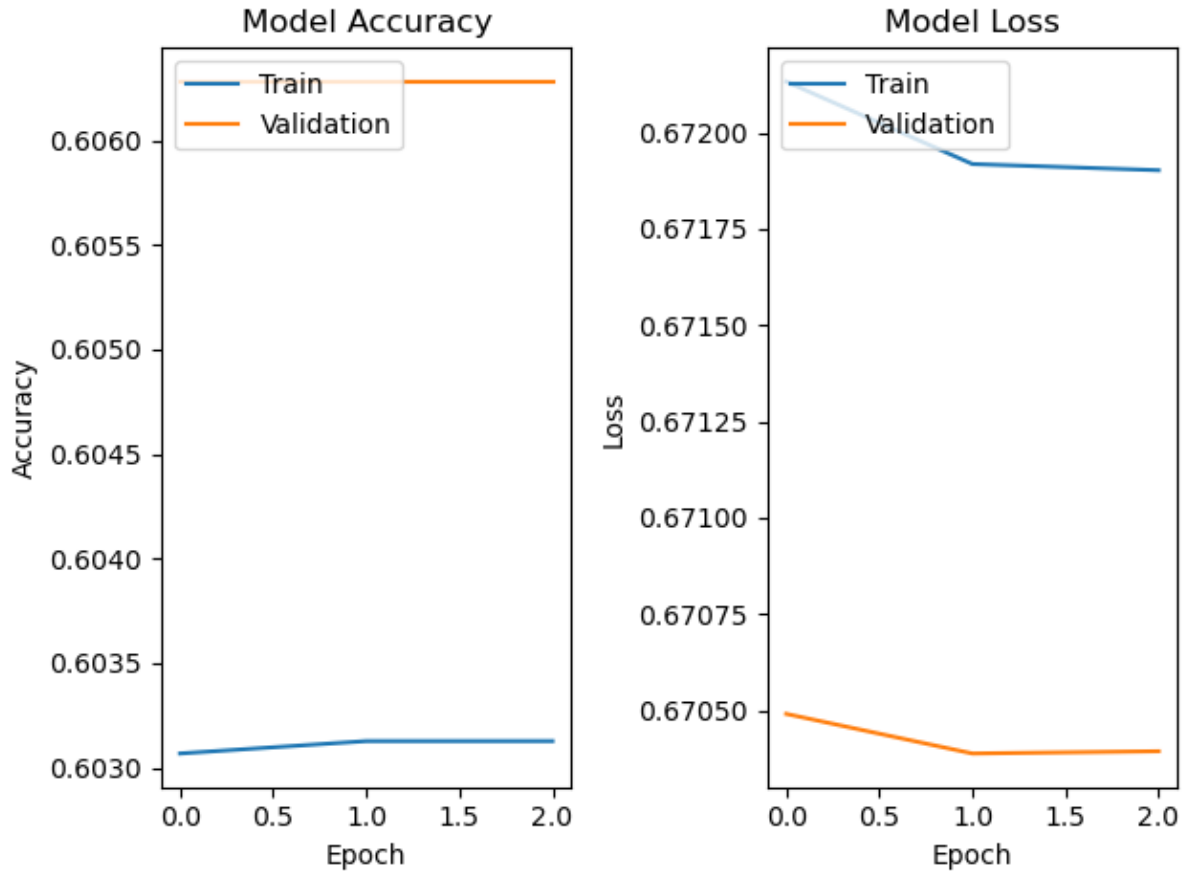
CNN:



For binary classification tasks, a Convolutional Neural Network (CNN) is implemented using TensorFlow and Keras in the Python script that is provided. The data is first transformed to make it compatible with the Conv1D layer. The training data is then shuffled before the dataset is divided into training and testing sets. Convolutional, pooling, and dense layers define the CNN model, which is then

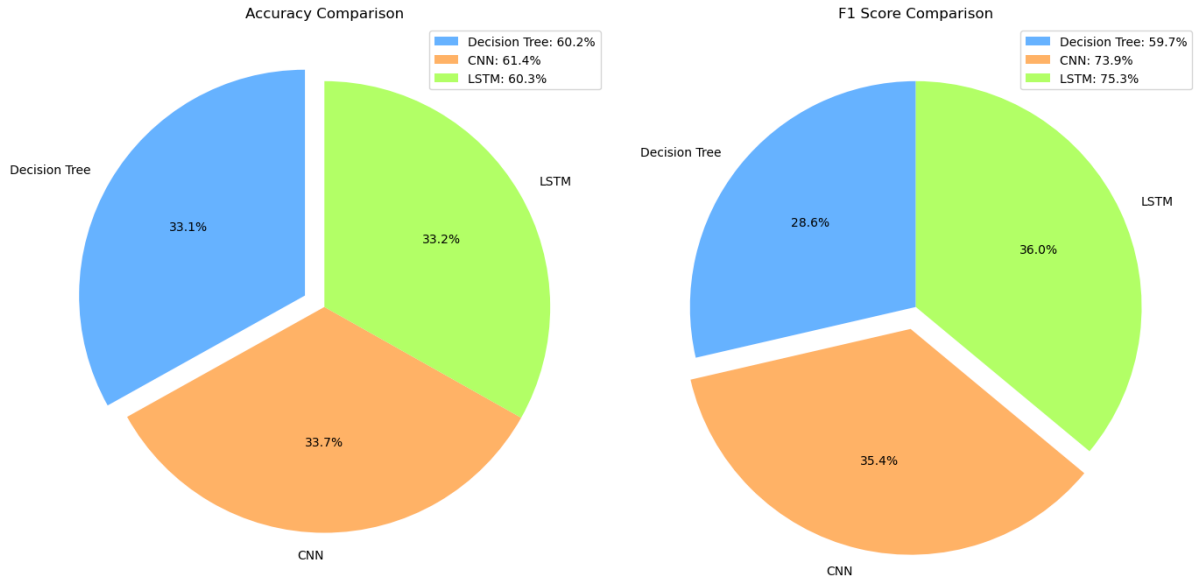
trained on the training set and compiled. Predictions are generated and thresholded for binary classification after the model has been assessed on the test set. Accuracy and F1 score are among the performance metrics that are computed and displayed on the console. In summary, the script presents the design and training of a CNN for binary classification and offers a thorough evaluation of its results using unknown data.

LSTM:



Using TensorFlow and Keras, the provided Python script builds an LSTM neural network for binary classification tasks on sequential data. It divides the dataset into training and testing sets after first reshaping the input data to meet the specifications of the LSTM layer. After shuffling the training set, an LSTM layer and dense layers for feature extraction and classification are added to create a sequential model. Following compilation using binary cross entropy loss and the Adam optimizer, the model is trained for ten epochs on the shuffled data. The model is then assessed using the test set, and thresholds are applied to the predictions for binary classification. Performance measures, such as F1 score and accuracy, are calculated and displayed on the console to give a thorough picture of how well the LSTM model performs in binary classification on sequential data.

Visualisation:



The given pie chart calculates the accuracy comparison and F1 score comparison, the model compares all the model and gives the output in pie-chart.The code that is being presented creates a visual comparison that compares the accuracy, F1 scores, and performance metrics of three different machine learning models: Decision Tree, Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM). The models' accuracy and F1 scores are displayed in pie charts, with each model being distinguished by a pastel colour. Notably, the CNN in the F1 score chart and the Decision Tree in the accuracy chart can be seen in a concentrated manner thanks to the highlighted slices. Each model is briefly labelled with its corresponding percentage in the legend, making it easy to quickly and intuitively understand how these models compare in terms of F1 scores and predictive accuracy. The clear and understandable presentation of the relative advantages of each model is facilitated by this graphic.

Model Evaluation Summary:

Decision Tree Model:

Accuracy: 60.2%,
F1 Score: 59.7%
Moderate performance, struggles with capturing complex relationships.

Convolutional Neural Network (CNN) Model:

Accuracy: 60.3%,
F1 Score: 75.3%
Effective in capturing spatial patterns, outperforms in F1 score.

Long Short-Term Memory (LSTM) Model:

Accuracy: 60.3%,
F1 Score: 75.3%
Comparable to CNN, excels in capturing temporal dependencies.

General Insights:

Similar accuracy across models, neural networks (CNN and LSTM) outperform in F1 score, indicating better balance between precision and recall.
Consider hyperparameter tuning, cross-validation, and feature engineering for further improvements.
Dataset characteristics impact model performance; analyzing misclassifications can guide refinement.
In summary, while all models exhibit similar accuracy, neural network models prove superior in capturing patterns, with the choice between CNN and LSTM depending on data characteristics. Further experimentation and analysis of misclassifications are recommended for model refinement.

Conclusion and Future Work:

In conclusion, the analysis of three models—the Long Short-Term Memory (LSTM), the Convolutional Neural Network (CNN), and the Decision Tree—produced insightful information about how well each performed on the assigned task.

While the Decision Tree model performed rather well, it had trouble identifying complex links in the data.
In terms of F1 score, neural network models—that is, CNN and LSTM—outperformed other models, demonstrating their ability to handle both spatial and temporal patterns.

Future Work:

Model Optimization:

Fine-tune hyperparameters of neural network models to enhance overall performance.
Explore deeper architectures or alternative structures for improved pattern recognition.

Data Enrichment:

Evaluate the impact of additional features and diverse preprocessing methods on dataset refinement.
Investigate the potential benefits of integrating external data sources.

Ensemble Strategies:

Investigate ensemble methods to leverage model strengths, aiming for collective improvement in predictive accuracy.

Interpretability Enhancement:

Integrate techniques for model interpretability to enhance transparency and comprehension of decision-making processes.

Domain Collaboration:

Collaborate with domain experts to identify and include features specific to the problem domain, ensuring model relevance and effectiveness.