Data Science Project

*Elaborated by :* ligne courte

TAJOURI Mohamed Iyadh

BOUSSELMI Mohamed Aziz

KAMEL Said

KAMEL Alaa

BEJAOUI Samy

Contents

[I. General Introduction 3](#_Toc72924810)

[II. Project Context 4](#_Toc72924811)

[III. Business Understanding 5](#_Toc72924812)

[A. Business Objectives 5](#_Toc72924813)

[B. Data Science Goals 5](#_Toc72924814)

[IV. Data Collection and Data Preparation 6](#_Toc72924815)

[A. Internal Data: 6](#_Toc72924816)

[B. Data source Description: 6](#_Toc72924817)

[C. Data understanding and data preparation: 6](#_Toc72924818)

[1. TXT data preparation: 6](#_Toc72924819)

[2. EDF files data preparation: 13](#_Toc72924820)

[3. External data preparation: 17](#_Toc72924821)

[V. Modeling 22](#_Toc72924822)

[A. CNN: 22](#_Toc72924823)

[B. RNN 25](#_Toc72924824)

[C. Logistic Regression 26](#_Toc72924825)

[D. XGBoost 29](#_Toc72924826)

[VI. Deployment 31](#_Toc72924827)

[A. Introduction 31](#_Toc72924828)

[B. Tools and technologies 31](#_Toc72924829)

[1. Python 31](#_Toc72924830)

[2. Flask 31](#_Toc72924831)

[C. Web application 32](#_Toc72924832)

[1. Login page 33](#_Toc72924833)

[2. Home Page 34](#_Toc72924834)

[3. Patient’s list page 35](#_Toc72924835)

[4. Analyze EEG page 35](#_Toc72924836)

[5. Data visualization page 36](#_Toc72924837)

[VII. Conclusion 37](#_Toc72924838)

# General Introduction

Electroencephalography (EEG) is an electrophysiological monitoring method to record electrical activity of the brain. It is typically noninvasive, with the electrodes placed along the scalp, although invasive electrodes are sometimes used such as in electrocorticography. It is frequently used for the diagnosis and management of various neurological conditions such as epilepsy, somnipathy, coma, encephalopathies, and others. Despite having lower spatial resolution than brain imaging techniques such as magnetic resonance imaging (MRI) and computed tomography (CT), EEG is a popular diagnostics tool among physicians due to its excellent temporal resolution, low cost, and noninvasive nature.

Generally, symptoms are not always guaranteed to be present in EEG data, but diagnosis of typical neurological disorders involves very long term monitoring of the patient. In this process a large amount of data is generated. This paired with the problem of there being a dearth of expert neurophysiology investigators makes way for creation of automated task based systems. It is these reasons why automatic interpretation of EEG by machine learning techniques has gained popularity in recent times.

It is shown that the combination of raw time series and RNNs eliminates the need to extract handcrafted features and allows the classifier to automatically learn relevant patterns, surpassing their results.

# Project Context

Epilepsy affects almost 1% of the population and most of the approximately 20–30% of patients with refractory epilepsy have one or more seizures per month. Seizure detection devices allow an objective assessment of seizure frequency and a treatment tailored to the individual patient. A rapid recognition and treatment of seizures through closed-loop systems could potentially decrease morbidity and mortality in epilepsy.

However, no single detection device can detect all seizure types. Therefore, the choice of a seizure detection device should consider the patient-specific seizure semiologies.

This review of the literature evaluates seizure detection devices and their effectiveness for different seizure types.

Our aim is to summarize current evidence, offer suggestions on how to select the most suitable seizure detection device for each patient and provide guidance to physicians, families and researchers when choosing or designing seizure detection devices. Further, this review will guide future prospective validation studies.

# 

# Business Understanding

## Business Objectives

Without the help of an automated system, the process of seizure detection will take much longer, that’s why we are opting to provide fast services for neurologists. Aside from the duration of the process, the latter takes a toll on the experts who are spending countless hours reading and analyzing EEG signals therefore, the system will relieve them from this tiring task and will present them with a detailed analysis of these signals.

## Data Science Goals

Visualization of the EEG brain signals in order to get an accurate classification with the help of external data:

○In order to classify and get such results the application of machine learning algorithms such as KNN, SVM, single Decision Tree was the go-to method.

* ○      Nevertheless, since 2016, substantial research has embarked on the field of identifying epilepsy using deep learning models, such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN)
* ○      We chose to use deep learning algorithms rather than machine learning algorithms like SVM, KNN since the former has proven to be extremely influential in classification … Its **algorithms** can **learn** appropriate features on its own which is extremely helpful in analyzing and detecting epilepsy from several eeg signals.
* ○      These visualizations consist of graphs and curves that will aid us in determining the epileptic and no epileptic results.

# Data Collection and Data Preparation

## Internal Data:

The dataset used in this work was acquired by a University research team from Bonn.

The dataset file contains some basic statistics about the TUH EEG Epilepsy Corpus, a corpus developed to motivate the development of new methods for automatic analysis of EEG files using machine learning. This corpus is a subset of the TUH EEG Corpus and contains sessions from patients with epilepsy. To balance the corpus, some sessions are provided from patients that do not have epilepsy.

## Data source Description:

Dataset Name: TUH EEG Epilepsy Corpus

Total Txt Files: 561 (428 are epileptic and 133 are non-epileptic)

Total EDF Files: 1648 (1360 are epileptic and 288 are non-epileptic)

Total number of patients: 237 (133 are epileptic and 104 are non-epileptic)

## Data understanding and data preparation:

TXT data preparation

### TXT data preparation:

In order to extract the data, first of all, we found all the paths to all the txt files and saved them in a variable called target Pattern.



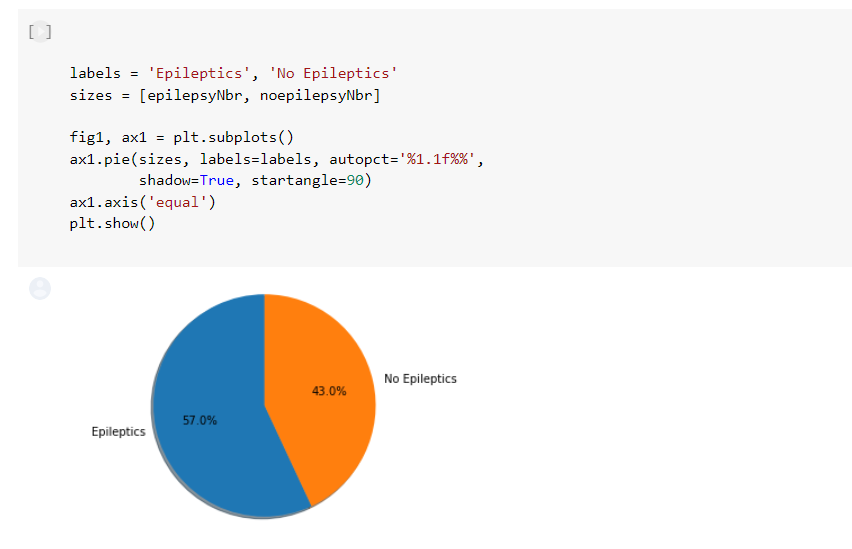
*Figure 1: Finding path of txt and edf*

We then checked the number of total patients that we have in our dataset:



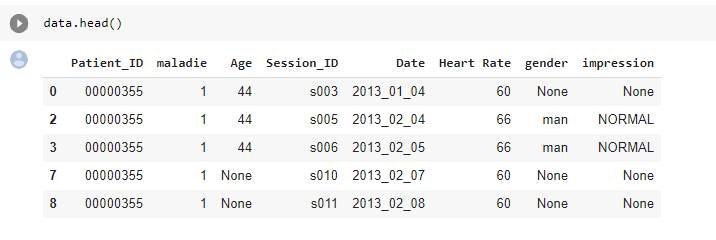
*Figure 2: Number of Patients*

We also checked the percentage of epileptic and non-epileptic patients using a pie chart and we can see that 57% are epileptic while 43% are non-epileptic

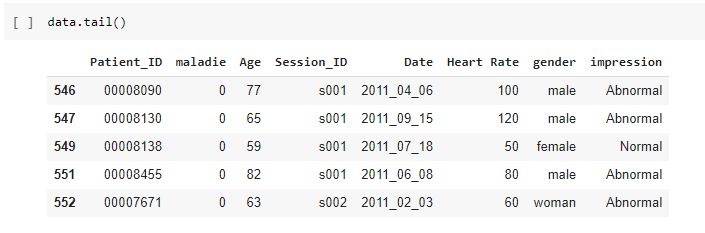


*Figure 3: Epileptic Pie Chart*

We visualized the first 5 lines containing information that we gathered from our txt files as well as the last 5 lines.

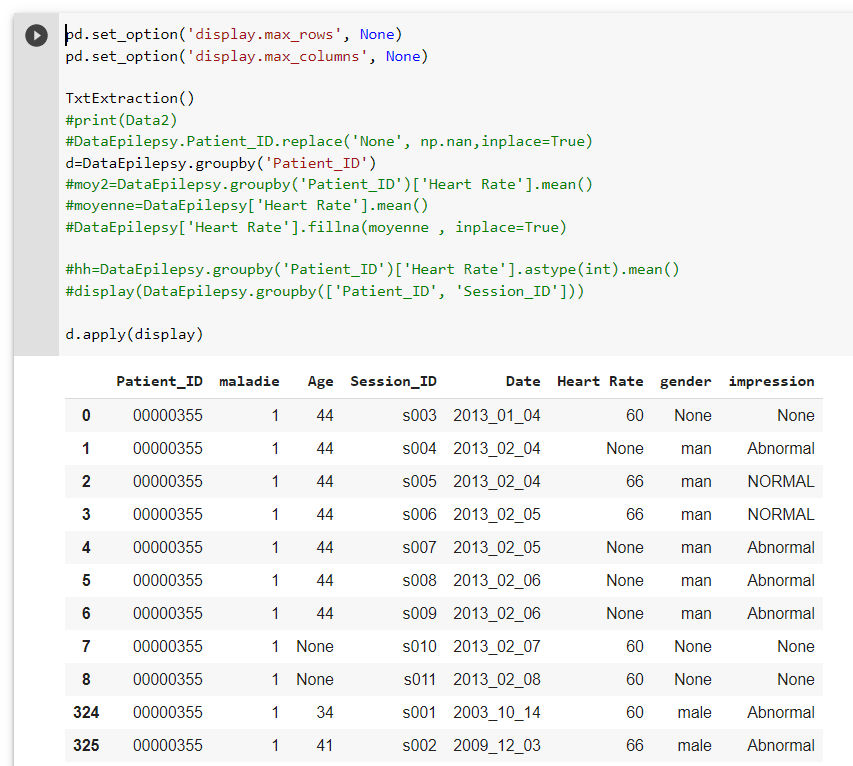


*Figure 4 : First Five lines*



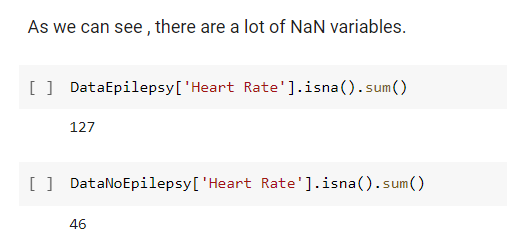
*Figure 6: Last Five Lines*

We then decided to group the sessions by the patient id so that our data becomes a bit more organized and it’d be easier for us to continue the preparation.



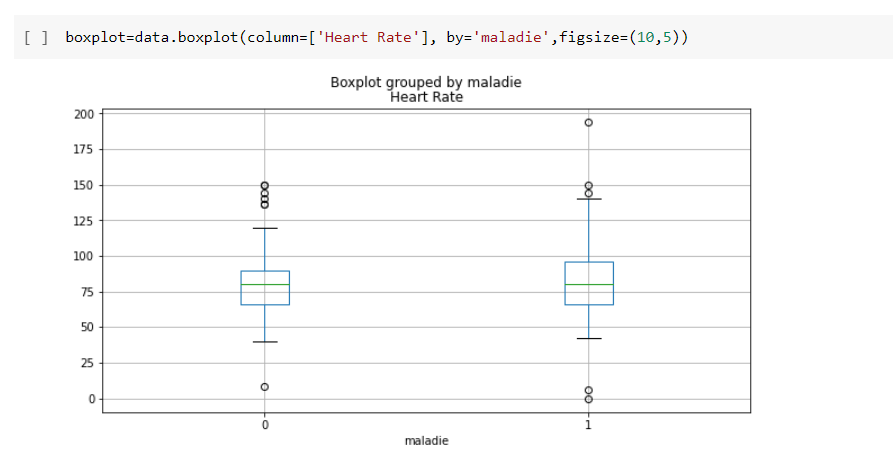
*Figure 7: Group by Session*

We checked for NaN values in our variables so that we could replace them with the appropriate values.



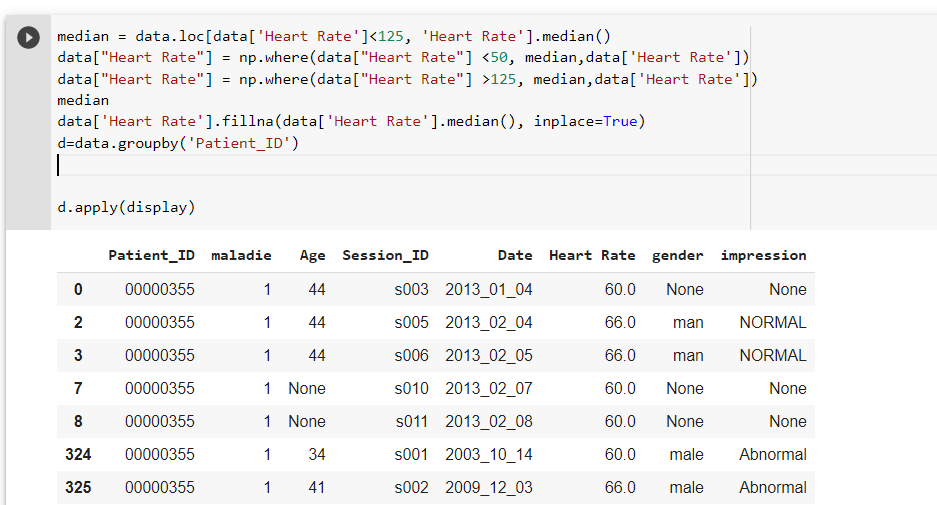
*Figure 8: NaN values in ‘Heart Rate’*

We also verified if there were any outliers in the ‘Heart Rate’ column that needed to be fixed.



*Figure 9: Outliers in ‘Heart Rate’*

As we can see in the boxplot above, there are a few values that are either too high (exp: 197) or too low (exp: 2) and to fix this, we first calculated the value of the Median and then replaced the Heart Rate values that are below 50 or above 125 or the ones that are missing (NaN) with the value of the median:



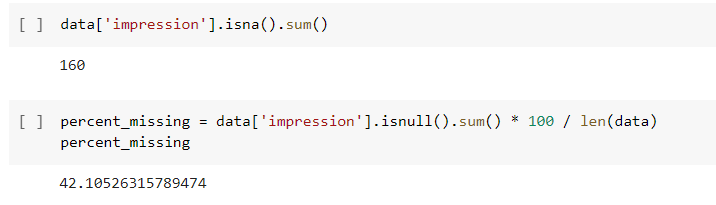
*Figure 10: Changing Outlier Values to Median*

As for the missing values in the ‘Age’ column, we created 2 functions; the first one to get the age of the patient found in one of his sessions and the second one to replace the missing values with the one that we extracted using our first function.



*Figure 11: Filling missing ‘Age’ values*

we noticed that the value None occurs a lot in the ‘Impression’ column so we calculated the number of mission values and then checked its percentage:



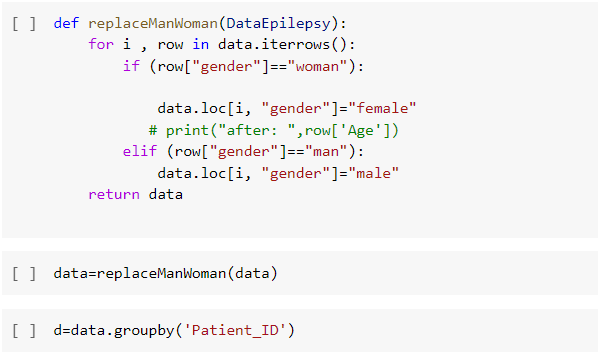
*Figure 12: Impression ‘None’ Percentage*

And as we can see in the figure above, 42% of the values are missing so we decided to delete it due to it not being important.

At last, we replaced the missing values in the gender column and also noticed that in some cases, the doctor would use the word ‘woman’ or ‘lady’ instead of ‘female’ or in the case of the male gender they would use the word ‘guy’ or ‘gentleman’ instead of ‘male’ so we made sure to fix those values:



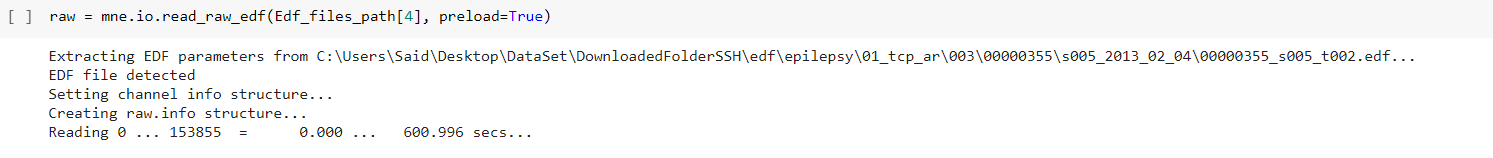
*Figure 13 : Filling missing ‘gender’ values*



*Figure 14: Unifying gender values to male / female*

### EDF files data preparation:

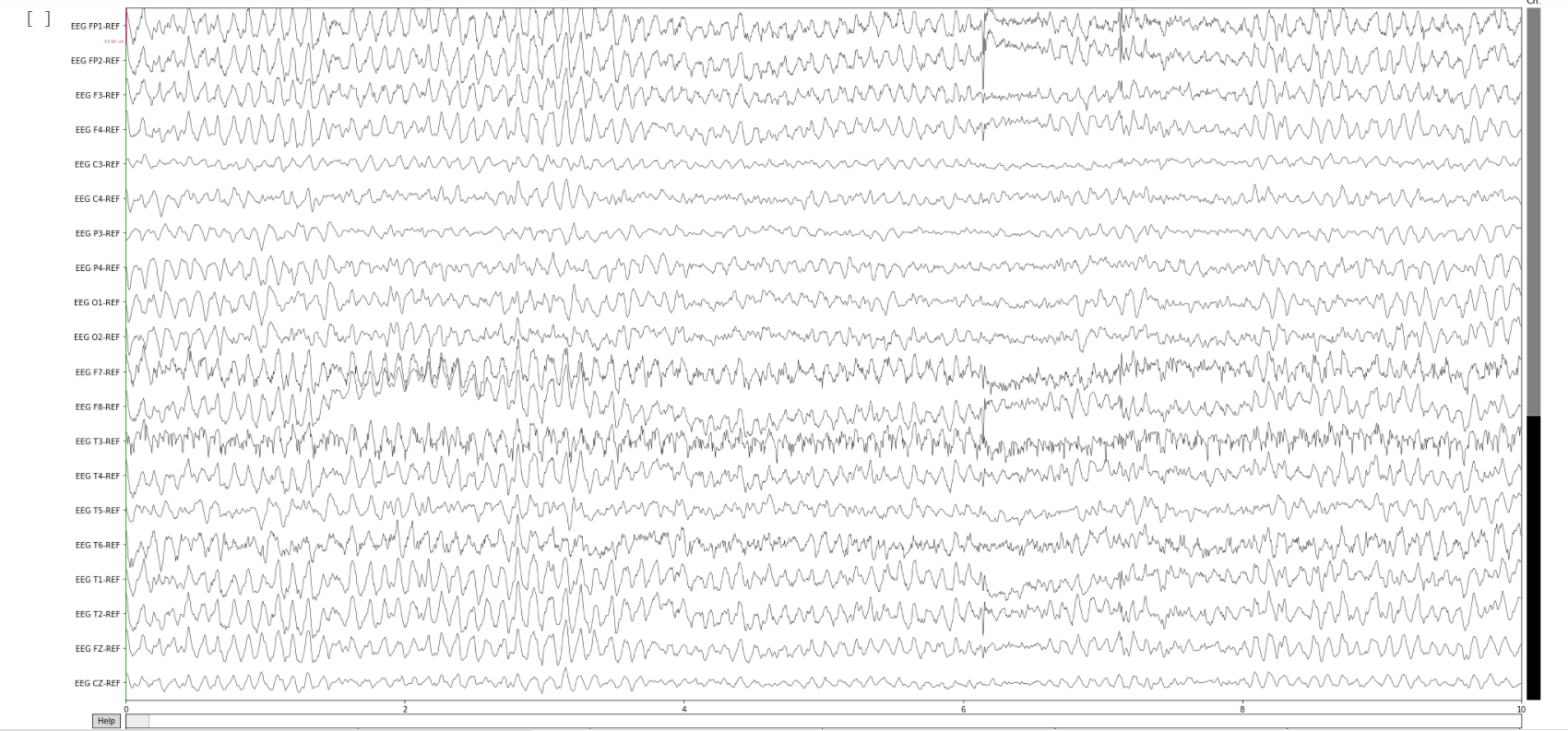
During this phase we will be using Python MNE Library. First, we load and read them. edf file:



*figure 15: Loading edf files*

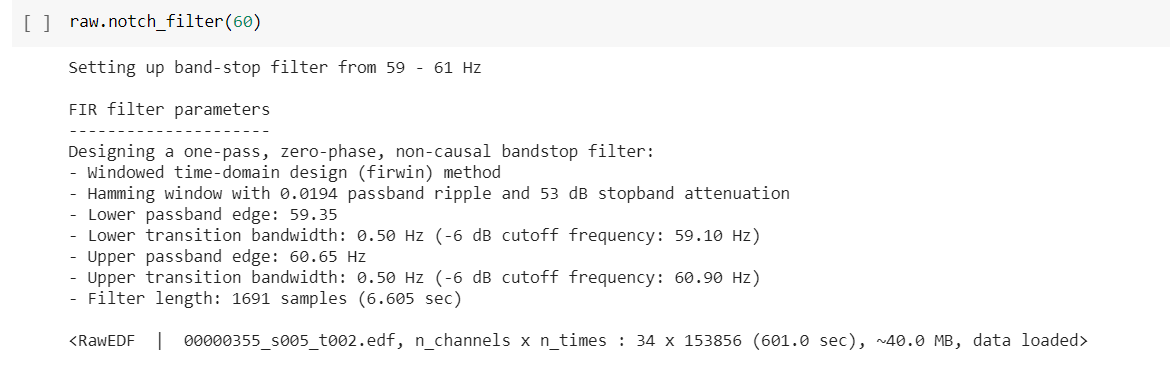
Then we displayed the raw data of a random edf file:





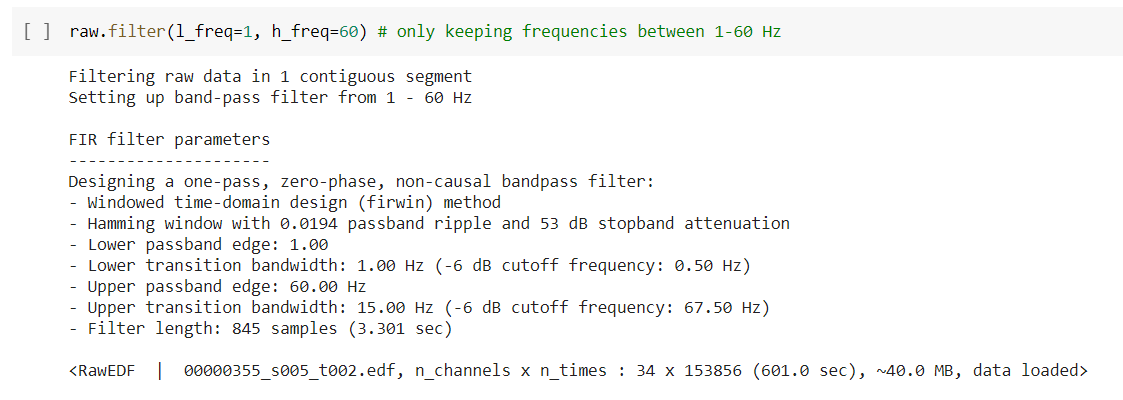
*Figure 16: RAW edf display*

Now that we have the raw data, we can filter the data to select the frequencies we're interested in. First of all, we want to remove the 60 Hz power line noise:



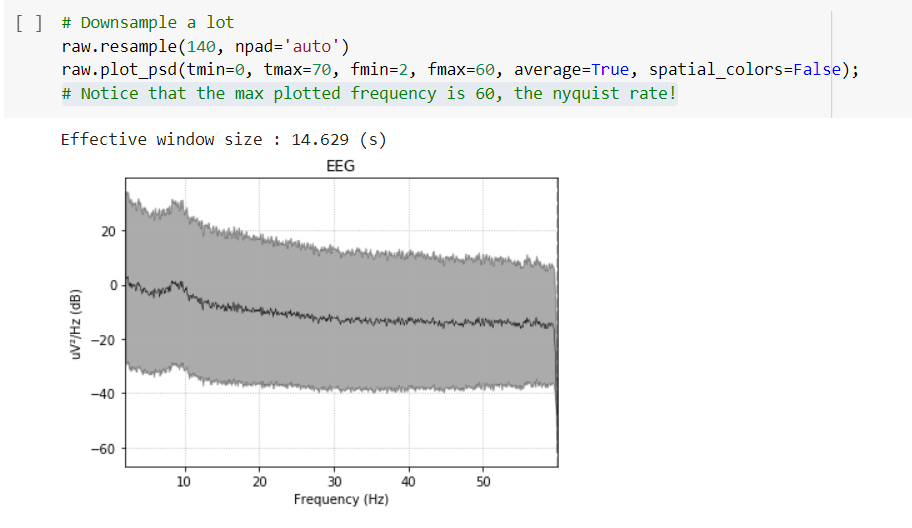
*figure 17: Removing 60Hz power line noise*

Now we remove very high and very low frequencies that are unlikely to contain the signal that is relevant to us:



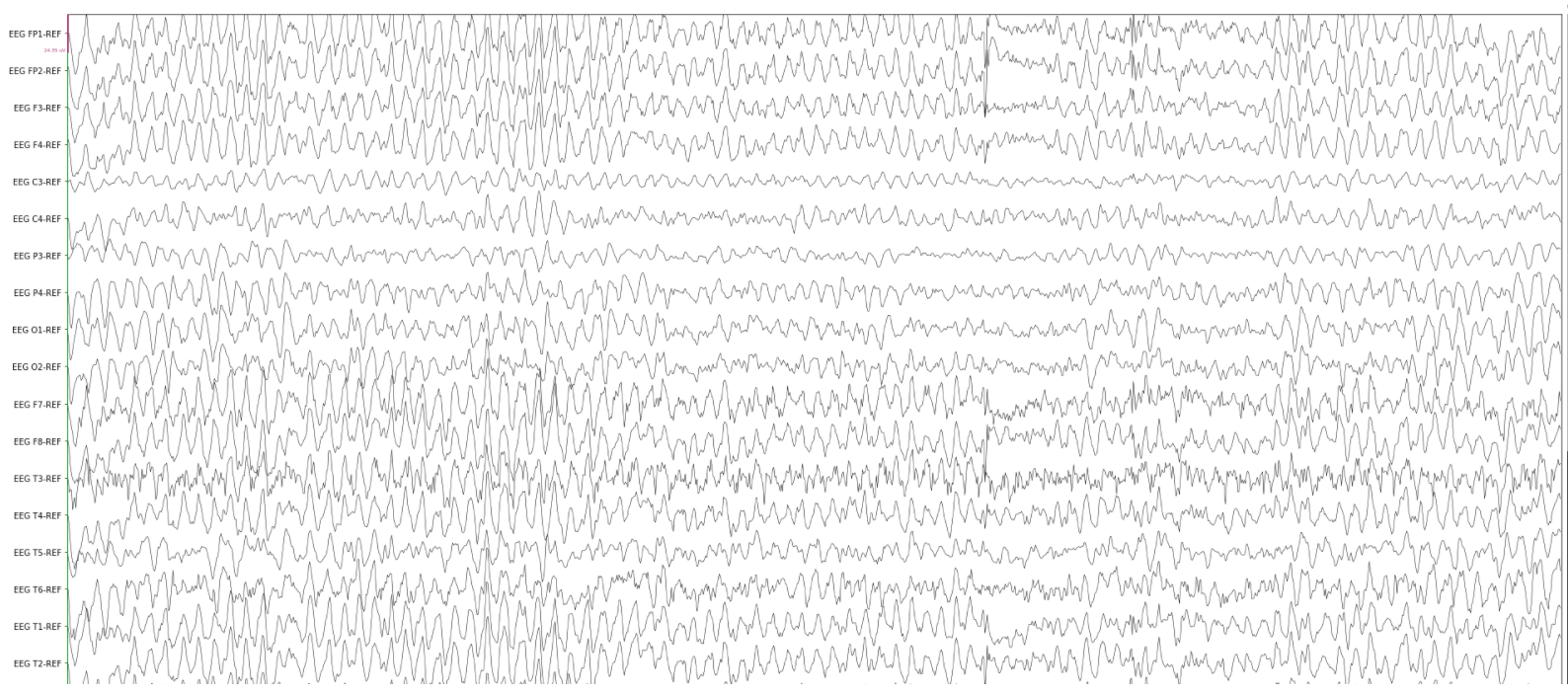
*figure 18: Removing very high and low frequencies*

Now we can downsample the data:

**

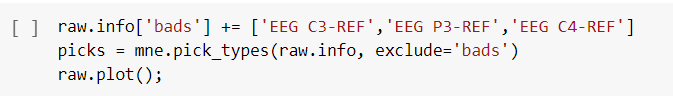
*Figure 19: Downsampling the data*

After filtering and downsampling, we can look at the raw data again to see if there are any bad channels:



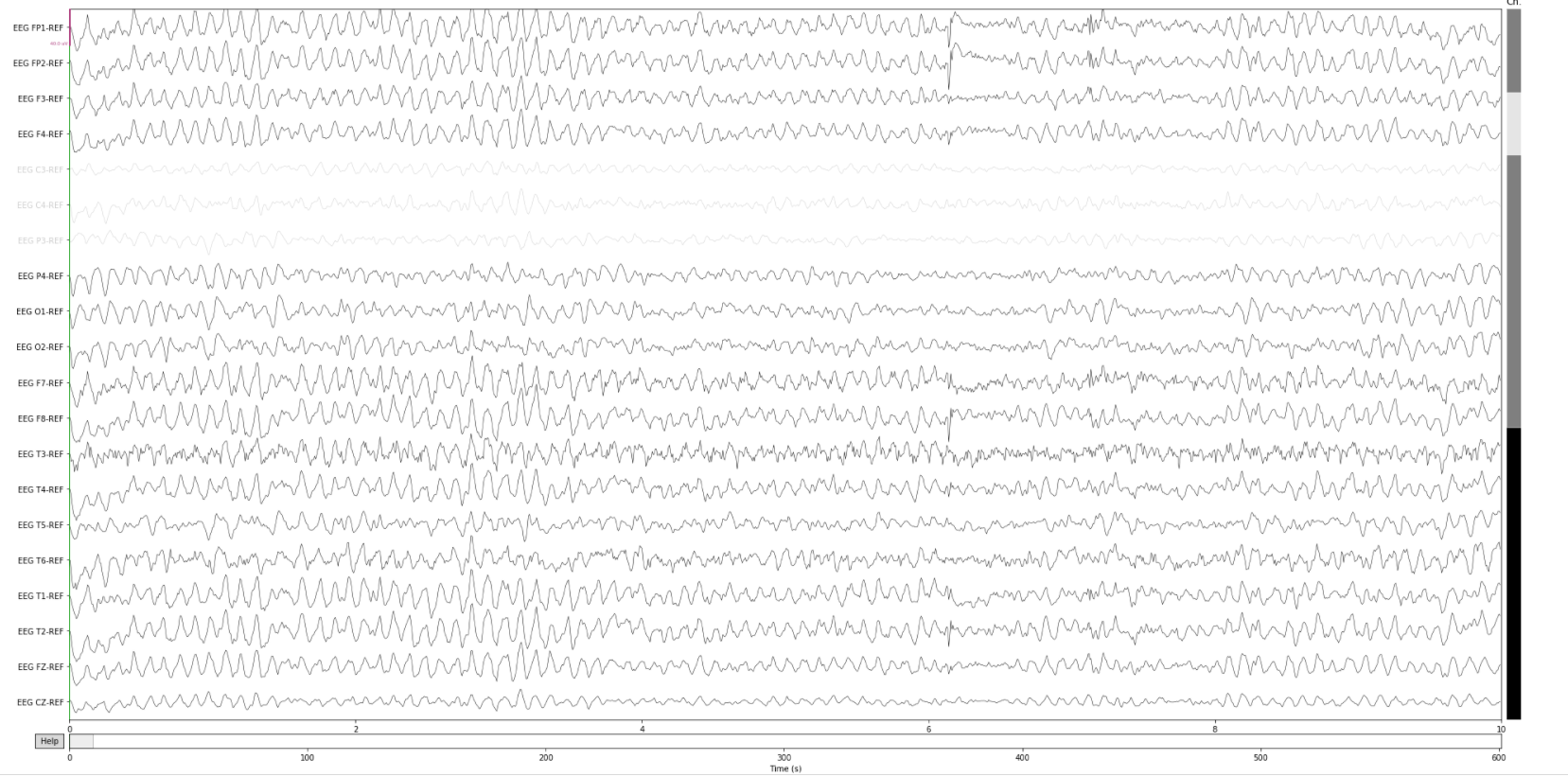
*Figure 20: Display filtered and downsampled data*

We noticed from the plot above that the channels EEG C3-REF, EEG C4-REF and EEG P3-REF look significantly noisier than the others. We can then flag it as a 'bad' channel and remove it:

**

*Figure 21: Removing noisy channels*

We display finally the preparated edf file:

**

*Figure 22: Display prepared EDF file*

### External data preparation:

The TUH EEG Corpus is a free public dataset. It’s goal is to enable deep learning research in neuroscience by releasing the largest publicly available unencumbered database of EEG recordings. This ongoing project currently includes over 30,000 EEGs spanning the years from 2002 to present.

First of all, we downloaded and displayed the data:





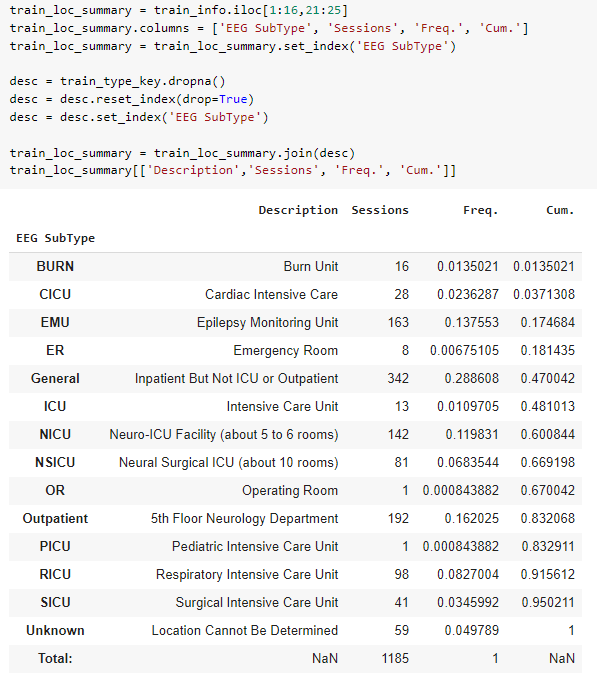
*Figure 23 : Display of the external data*

Then we proceeded to create a summary of the data:



*Figure 24: Summary of the different EEG types*

After displaying the different EEG types, we proceeded to display the different EEG subtypes:



*Figure 25: Display of different EEG subtypes*

Now that we’re done with our external data preparation,

it’s time to convert our EDF files to CSV so that we can apply the deep learning and machine learning algorithms on our data later on. To do that :

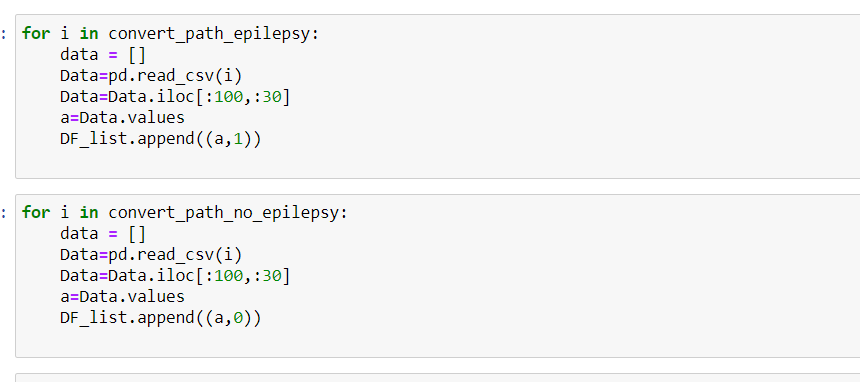


*Figure 26: Converting Epilepsy files to CSV*

**

*Figure 27: Converting Non Epilepsy files to CSV*

Finally, we extracted all of the data found in all of the CSV files and appended it to an Array.

**

*Figure 28: Appending CSV to DF\_List*

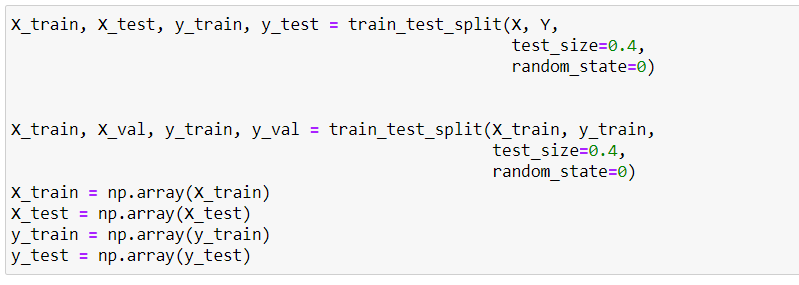
# Modeling

The modeling stage is where the various techniques of data mining are selected and applied to the data. It is important to have knowledge and understanding for the fundamentals of data mining, including the scoring techniques and algorithms that can be brought to bear in our project. For this part we are working with scikit-learn, simple and efficient tools for data mining and data analysis. In the context of the detection model, the objective is to assign patients to either epileptic or non-epileptic. Furthermore, the need of an appropriate technique of classification is very essential. Amongst these techniques we have: CNN, RNN, XGBoost and Logistic Regression. They are all commonly used in building detection models. In this chapter we will discuss the necessary steps to develop a detection model. In addition, the statistics behind this detection are also explained.

So let’s Start with the first **Deep Learning** Algorithm which is CNN:

## CNN:

In deep learning, a convolutional neural network (**CNN**, or **ConvNet**) is a class of deep neural networks, most commonly applied to analyze visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps.



*Figure 28: Declaring Train and Test variables*

We first declared our X\_train, X\_test, y\_train et y\_test variables that we’re going to use in our algorithm.

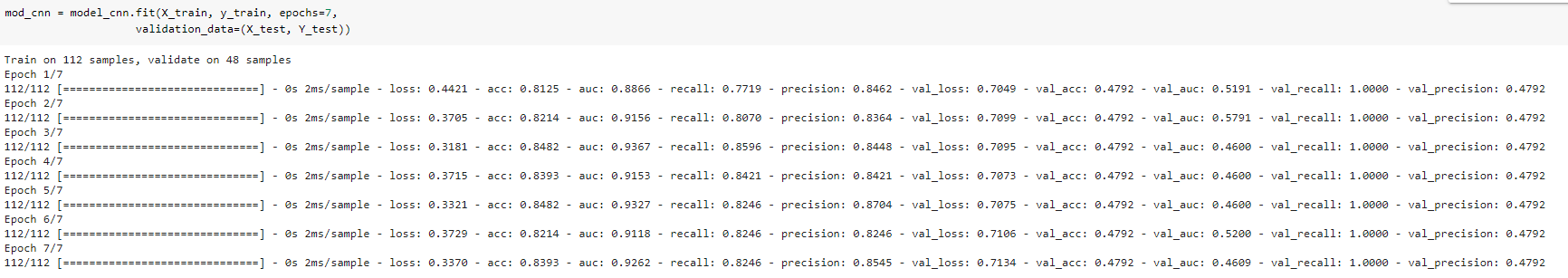


*Figure 29 : Building CNN model-1*



*Figure 30: Building CNN model-2*

We then added all of the necessary layers to our CNN model, now it’s time for us to call the “fit” method to train our model using the X\_train, y\_train, X\_test and y\_test that we declared earlier .



*Figure 31: Training our CNN model*

and now we can see the results and we can see the *Accuracy,* *Precision* as well as the *Recall* values as shown in the figure above.

## RNN

RNNs are a powerful and robust type of neural network, and belong to the most promising algorithms in use because it is the only one with an internal memory.

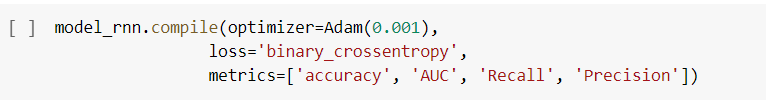
RNN’s can remember important things about the input they received, which allows them to be very precise in predicting what’s coming next. This is why they're the preferred algorithm for sequential data like time series, speech, text, financial data, audio, video, weather and much more.

Now it’s time to implement our Second-Deep Learning algorithm which is RNN (Recurrent Neural Network). The first thing that we did was add our LSTM layers as well as the Dense layers.



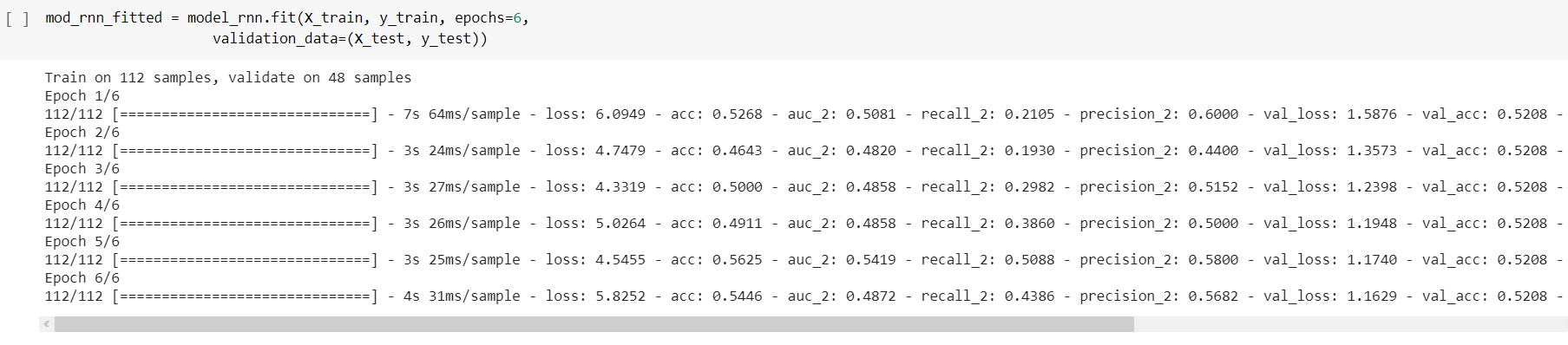
*Figure 32: Building RNN model*

Then, we compiled our model:



*Figure 33: Compiling RNN model*

Finally, we trained our model using our train and test variables:



*Figure 34: RNN training*

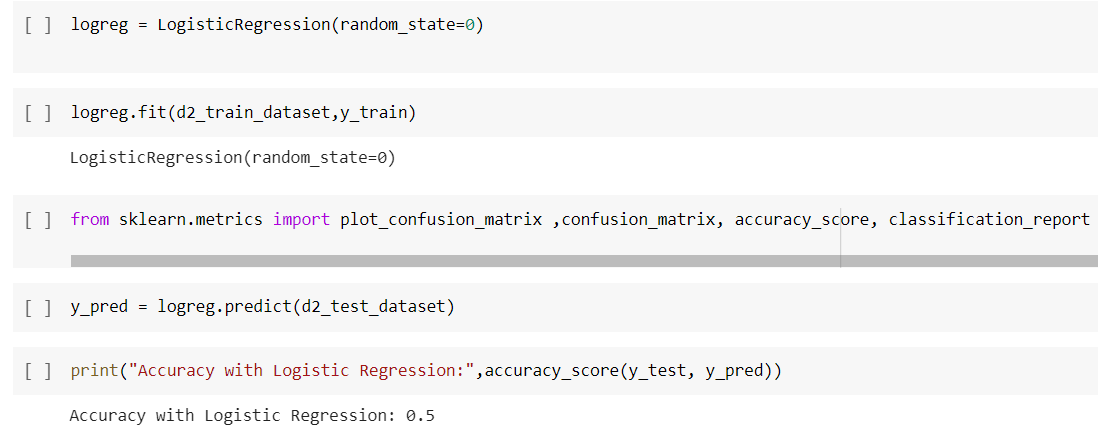
We can see in the figure shown above the *Accuracy,* *Precision* as well as the *Recall* values.

## Logistic Regression

**Logistic regression** is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

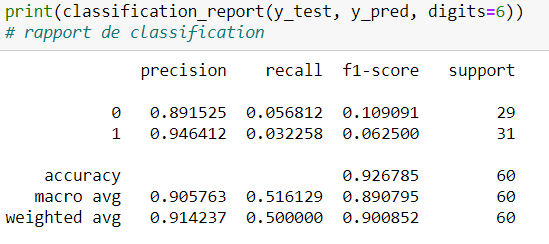
Sometimes logistic regressions are difficult to interpret, the Intellectus Statistics tool easily allows you to conduct the analysis.

The first algorithm that we chose to implement is a Machine Learning algorithm which is Logistic Regression,



*Figure 35: Logistic Regression Training*

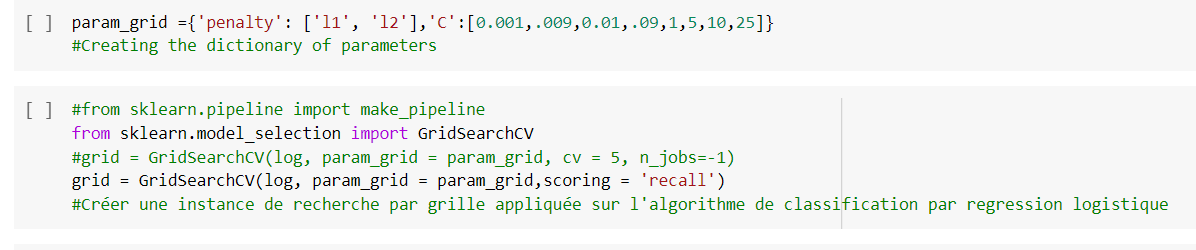
As we can see, we imported our Logistic Regression class and then trained our model, afterwards we called the predict method and we got an accuracy of 0.5.



*Figure 36: Logistic Regression Report*

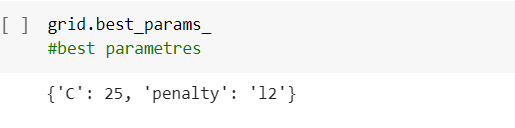
We also printed the classification report to get the values of the precision, recall and f1 score and the results are not that great and we need to improve our model in the next section.

To do that :



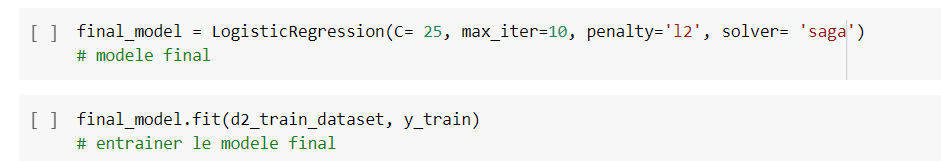
*Figure 37: Logistic Regression Grid Search*

We fixed our hyperparameters as shown in the figure and then applied a GirdSearch Cross Validation to improve our model.



*Figure 38: Logistic Regression Best Parameters*

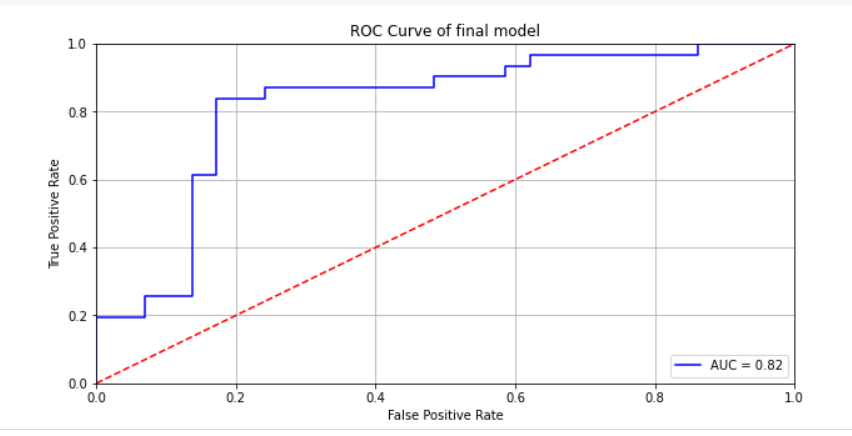
We then trained our model again with the new parameters and called the best\_params function to extract the best hyperparameters to use.

**

*Figure 39: New Model with Hyperparameters*

Now that we extracted the best hyperparameters we called our Logistic Regression class one last time and re-trained our model to get the best results:

We saw the accuracy go up to 90%



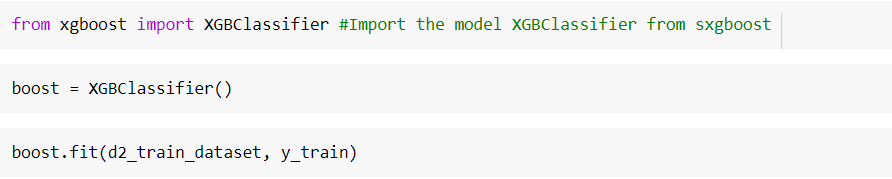
*Figure 40: ROC Curve after applying Hyperparameters*

Here we have our ROC Curve, we know that generally speaking the closer the curve is to the top left corner, the better so we can see now that our model is now more accurate.

## XGBoost

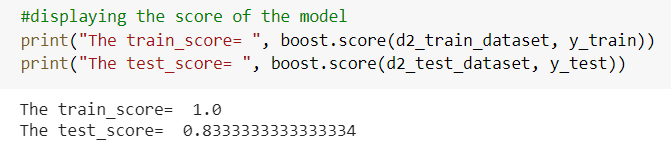
Lastly, we implemented the XGBoost machine learning algorithm.

XGBoostis a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks



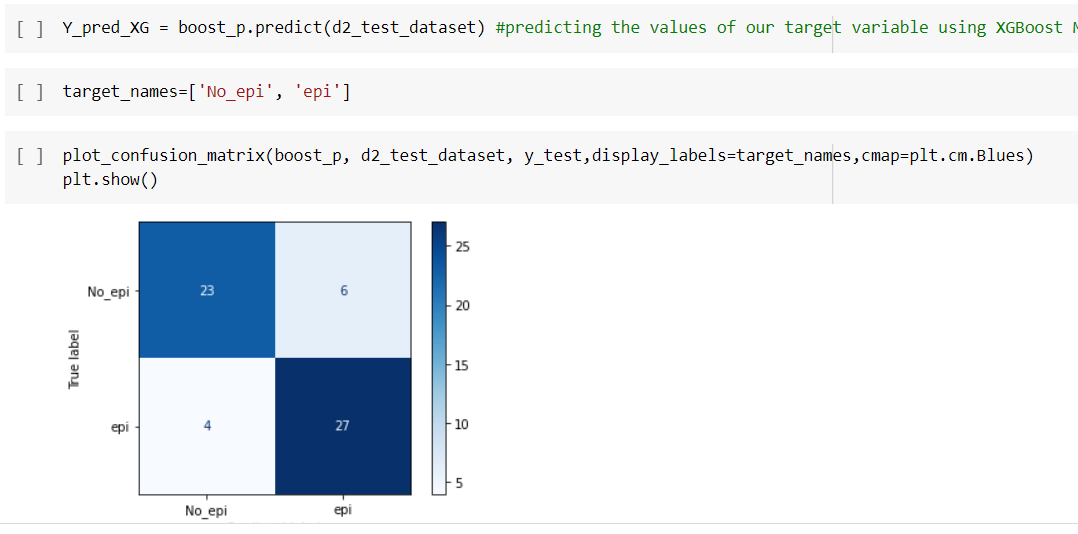
*Figure41: XGBoost Training*

We first imported the XGBClassifier class from XGBoost, called it and then used the fit () method to train our model:



*Figure 42: XGBoost Training results*

As we can see the train score that we got was 1.0 and the test score was 0.83



*Figure 43: XGBoost Confusing Matrix*

Then, we plotted the confusion matrix to display the true/false positives as well as the true/false negatives to get a better idea of how accurate our model is currently, and we can see that we got 4 false negatives out of 27 total and 6 false positives out of a total of 33, so we conclude that the precision that we got is quite high.

# Deployment

## Introduction

The deployment of machine learning models is the process for making your models available in production environments, where they can provide predictions to other software systems. To make our models available we made a Website using Python and flask.

## Tools and technologies

### Python

Python is the most widely used open-source programming language among computer scientists. This language has propelled itself to the forefront of infrastructure management, data analysis or in the field of software development



figure: Python

### Flask

Flask is an open-source micro framework for web development in Python. It is classified as micro framework because it is very light. Flask aims to keep the kernel simple but expandable.



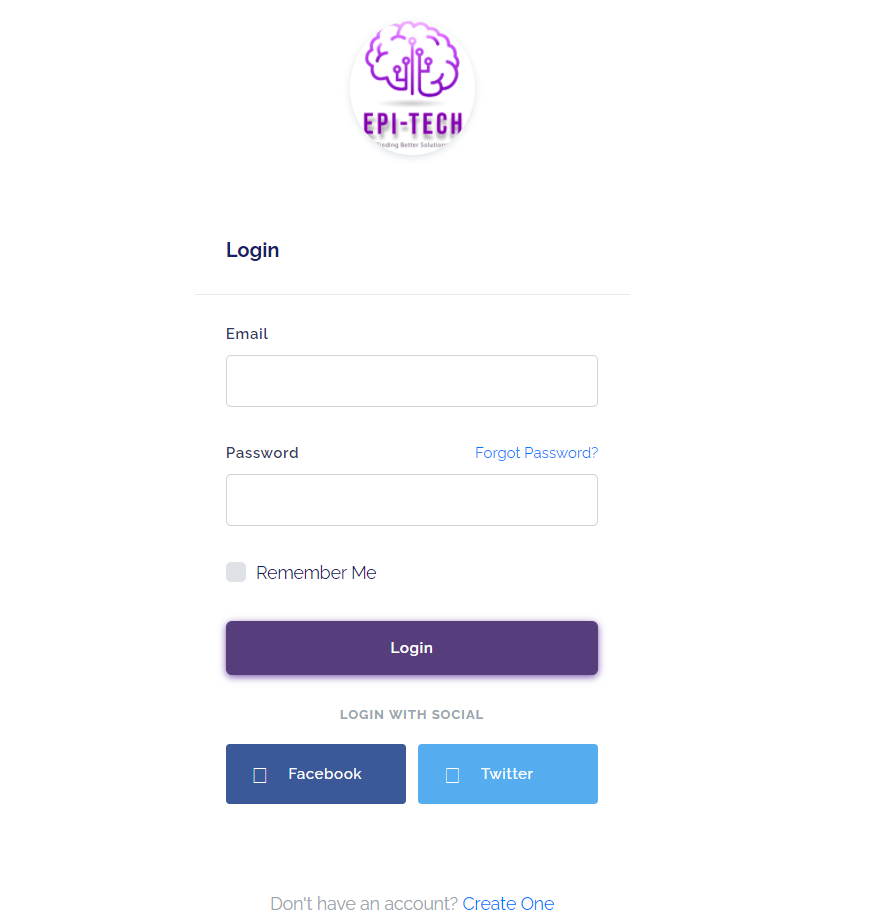
figure: Flask

## Web application

The system that we offer is scalable, operational and will provide all the necessary information.

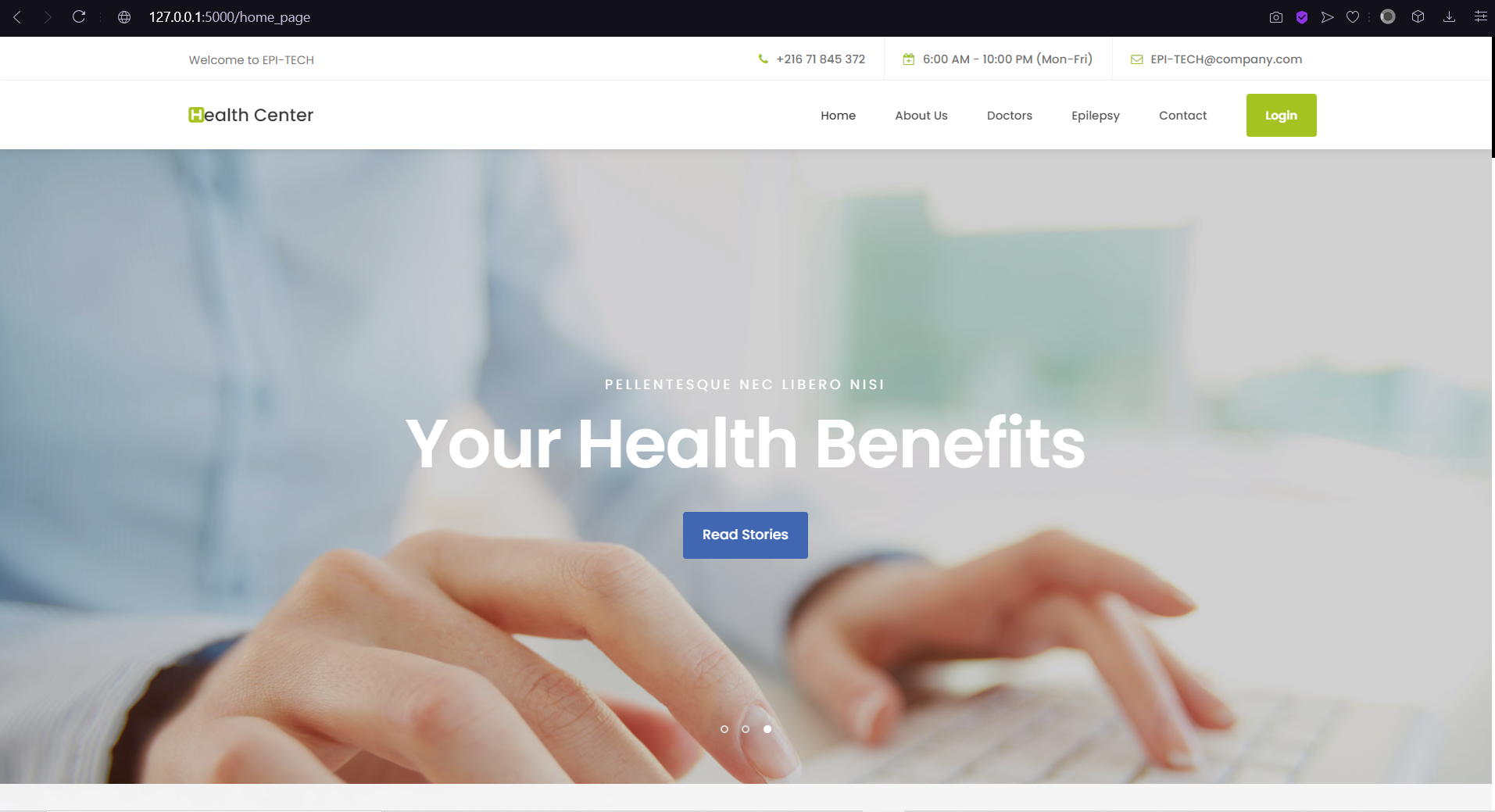
The Neurologist has the possibility to consult a detailed dashboard containing the data and different statistics about his patients. Added to that, he can fill out a form in which he can specify information about a new test for a patient and in the end get a report that will determine whether the patient is epileptic or not.

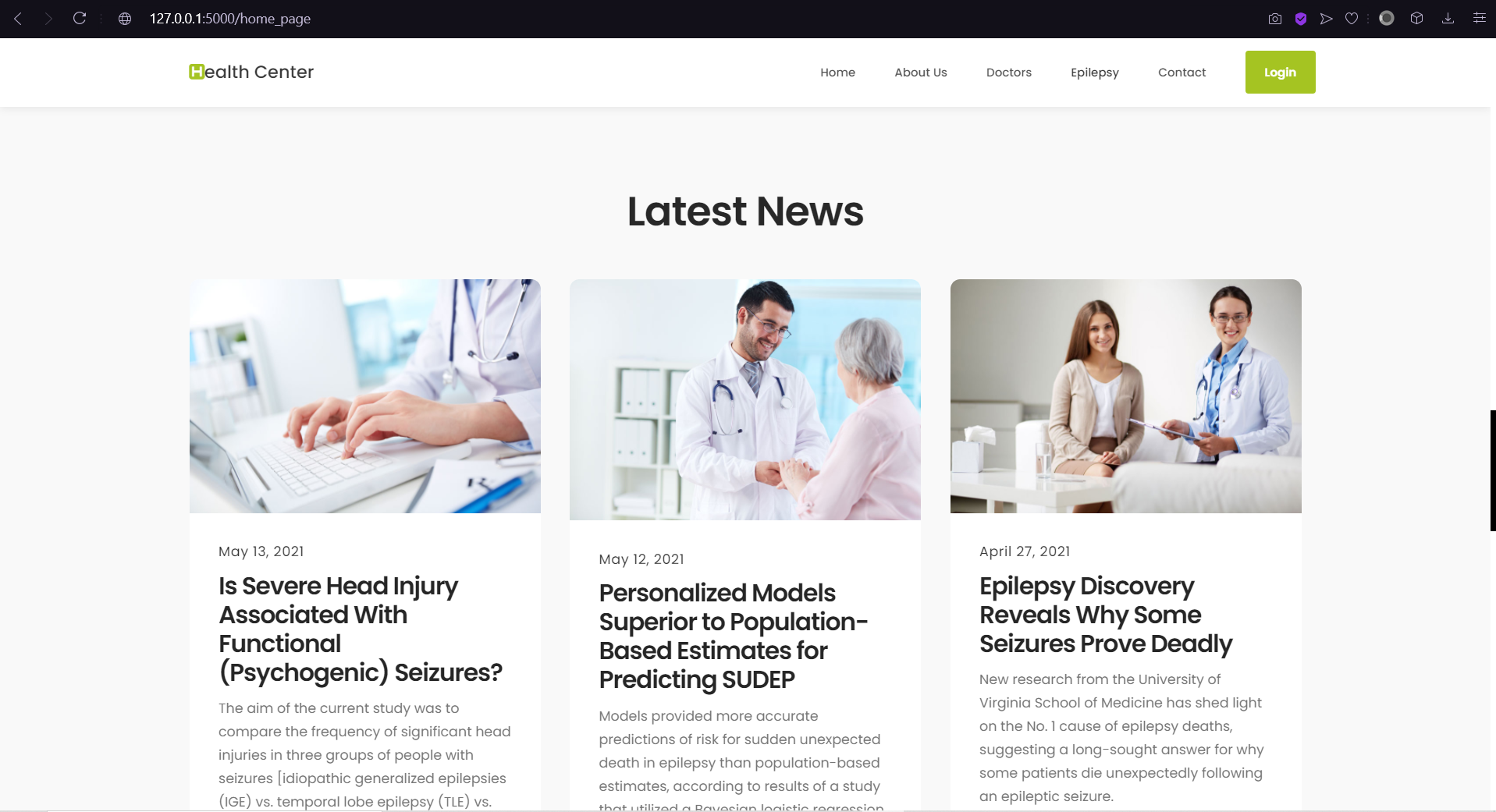
### Login page



As shown in the figure above, neurologists can login with their email address and the password provided to them by their medical establishment

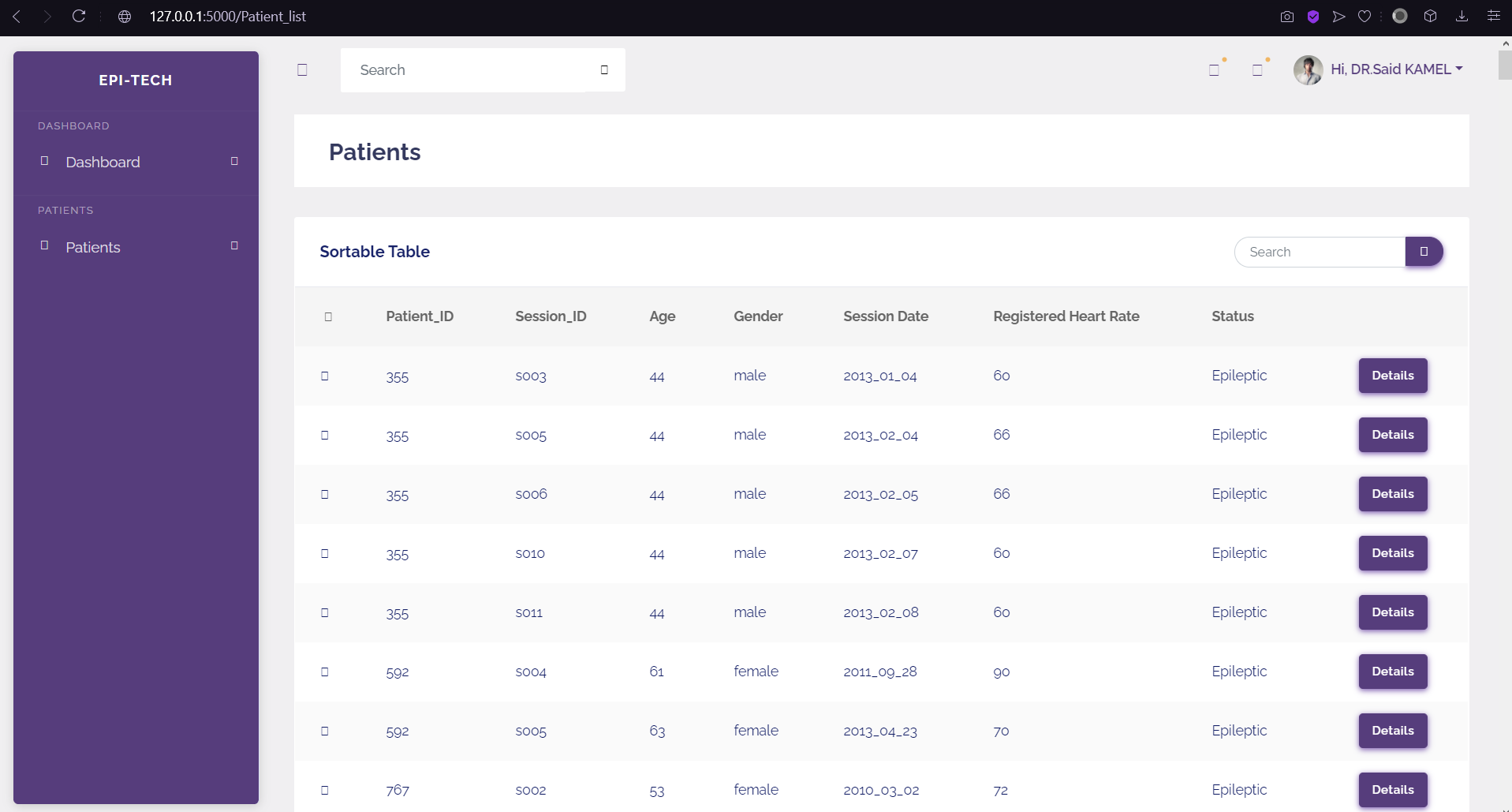
### Home Page





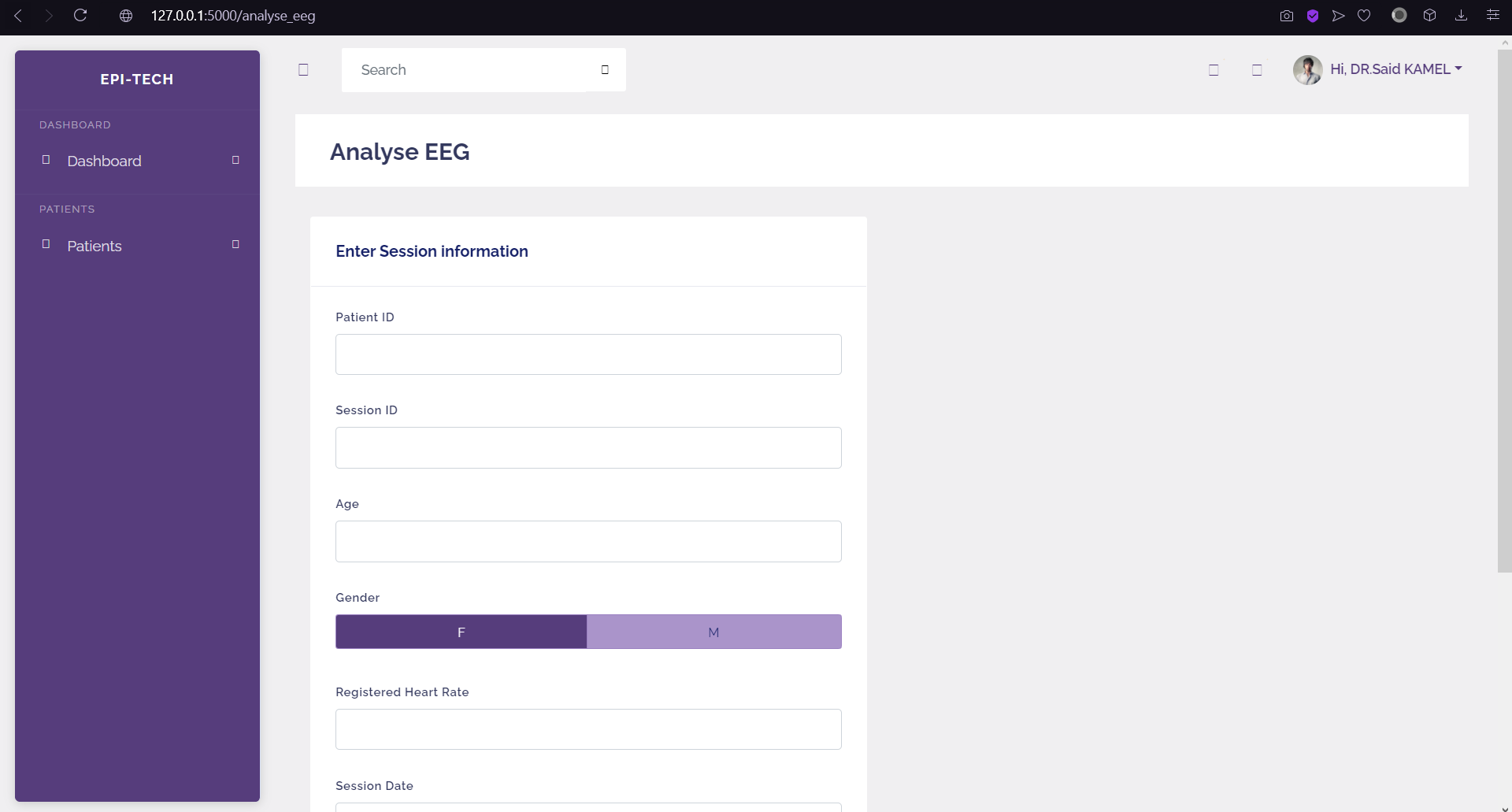
### Patient’s list page

Here is the list of patients with information about their age, gender, session date, registered heart rate and their status (epileptic / non-epileptic).



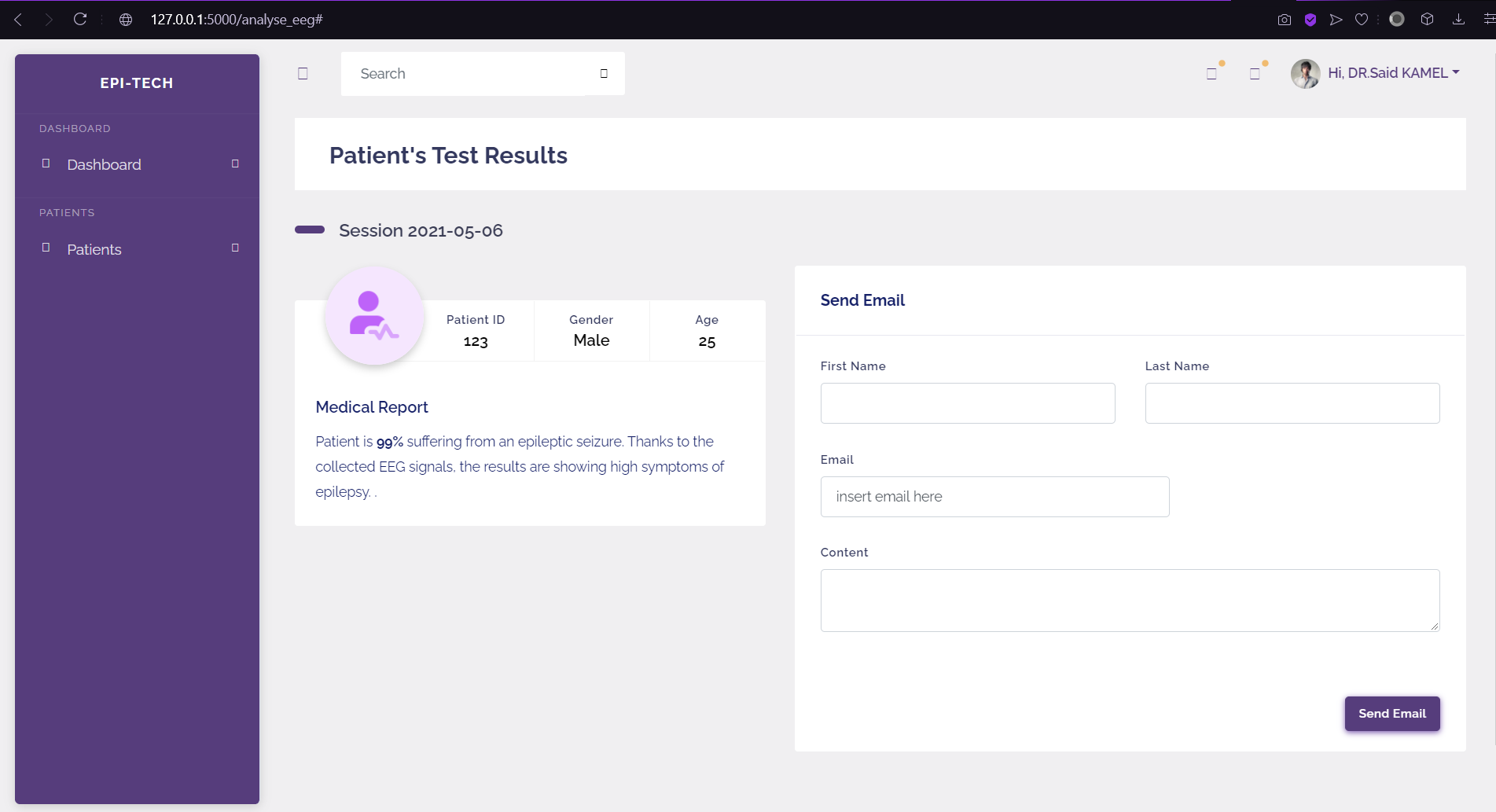
### Analyze EEG page

Here the neurologist will fill the form in order to analyze EEG brain signals and get accurate result about the specified patient



### Data visualization page

After filling the form here, we can see the Data visualization that the neurologist will get



# Conclusion

This part provided us with an overview of the general context of our project, we clarified our objectives and the needs of our client by specifying the business objectives.

We have gone through understanding the provided data, analyzing it in order to extract any information that could prove to be useful to our client and preparing our Internal Data for machine learning algorithms (data cleaning, merging different datasets,). Next, in order to enhance the decision-making process, we extracted additional data from the web.

This project has been a great opportunity for us to learn new Data Science techniques and expand our knowledge in this field as it gives us the chance to deal with real-world Data and work together towards a difficult set of goals.