Machine Learning Engineer Nanodegree
Capstone Project

Jean-Christophe Quillet
15/01/2017

DEFINITION

1. Domain Background

The project proposed is within the healthcare domain, and more specifically address the prediction of epilepsy seizures.
Epilepsy afflicts nearly 1% of the world's population, and is characterized by the occurrence of spontaneous seizures which can result in physical injuries. Anticonvulsant medications can be given at sufficiently high doses to prevent seizures, but patients frequently suffer side effects. For 20-40% of patients with epilepsy, medications are not effective. Epilepsy surgery may be an option when treatments are ineffective, but many patients continue to experience spontaneous seizures.
Despite the fact that seizures occur infrequently, patients with epilepsy experience persistent anxiety due to the possibility of a seizure occurring. Therefore, seizure forecasting systems have the potential to help patients with epilepsy to live more normal lives.

A possibility is to develop an electrical brain activity (EEG) based seizure forecasting systems able to identify periods of increased probability of seizure occurrence.
If seizure-permissive brain states can be identified, devices designed to warn patients of impeding seizures could be developed. Patients could punctually avoid potentially dangerous activities like driving or swimming, and medications could be administered only when needed to prevent impending seizures, reducing overall side effects.

This project is the subject of a Kaggle competition proposed by the Melbourne University AES (link: https://www.kaggle.com/c/melbourne-university-seizure-prediction). There is currently no technology available that can reliably predict imminent seizures. A first limited trial, by the university research team, using a small device implanted in the brain has showed possibilities to predict seizures [1]. The dataset currently available is the result of a larger study led by this research group.

I am personally focused on finding ways to utilize Machine Learning methods for innovative healthcare purposes. The study of EEG signals in particular holds a lot of promising future healthcare applications. This topic is also for me the opportunity to work in depth on signal processing methods which have applications in many other domains.

2. Problem Statement

The problem is to demonstrate the existence and accurate classification of a pre-seizure brain state in humans suffering from epilepsy from existing intracranial EEG recordings.
I propose to use a supervised learning approach to infer a classifier from the dataset to distinguish iEEG signal segment in a pre-seizure period from a baseline period.
The performance of the model will be assessed by the prediction accuracy on the available test sets.

3. Evaluation Metrics

The model prediction performance on the test set is assessed by the F1 score obtained on the test set. The equation for the F1 score is the following:

$$F1 = 2 \times \frac{(precision \times recall)}{(precision + recall)}$$

Considering the importance of predicting seizures correctly for the patient comfort and the large imbalance between preictal states and interictal states periods, it is also necessary to calculate the recall on preictal (pre-seizure) prediction, which represent the ability of the model to predict the positive samples.
The equation for the recall is the following:

$$Recall = \frac{True\ positive\ predictions}{(True\ positive\ predictions + False\ negative\ predictions)}$$

Finally, Kaggle uses the area under the ROC curve (AUC) score to evaluate the submissions and the measure is used here as an additional metric to compare the models.
The ROC curve is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied and is created by plotting the true positive rate (recall) against the false positive rate (FPR) at various threshold settings. It is thus the recall as a function of FPR. The AUC is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative.

For all of these metrics, the Sklearn python implementation is used.

ANALYSIS

1. Data Exploration

Input data:
The temporal dynamics of brain activity can be classified into 4 states:

- Interictal (between seizures, or baseline),
- Preictal (prior to seizure),
- Ictal (seizure)
- Post-ictal (after seizures).

The goal of this project is to differentiate the preictal and interictal states.

For this project, I have at my disposal a set of labelled data composed of iEEG signals from baseline and prior to seizure periods, and a table listing the files that can be safely used as well as the label of data.

Human brain activity was recorded in the form of intracranial EEG (iEEG) which are measures of electrical voltage fluctuation. For all the patients, iEEG was sampled simultaneously from 16 electrodes at 400 Hz (fig.1), and recorded voltages were referenced to the electrode group average.
Each file in the dataset corresponds to ten-minute-long data segments from measures covering an hour prior to a seizure (preictal states), and ten-minute-long data segments of interictal states. Each data segment includes 16 channels.

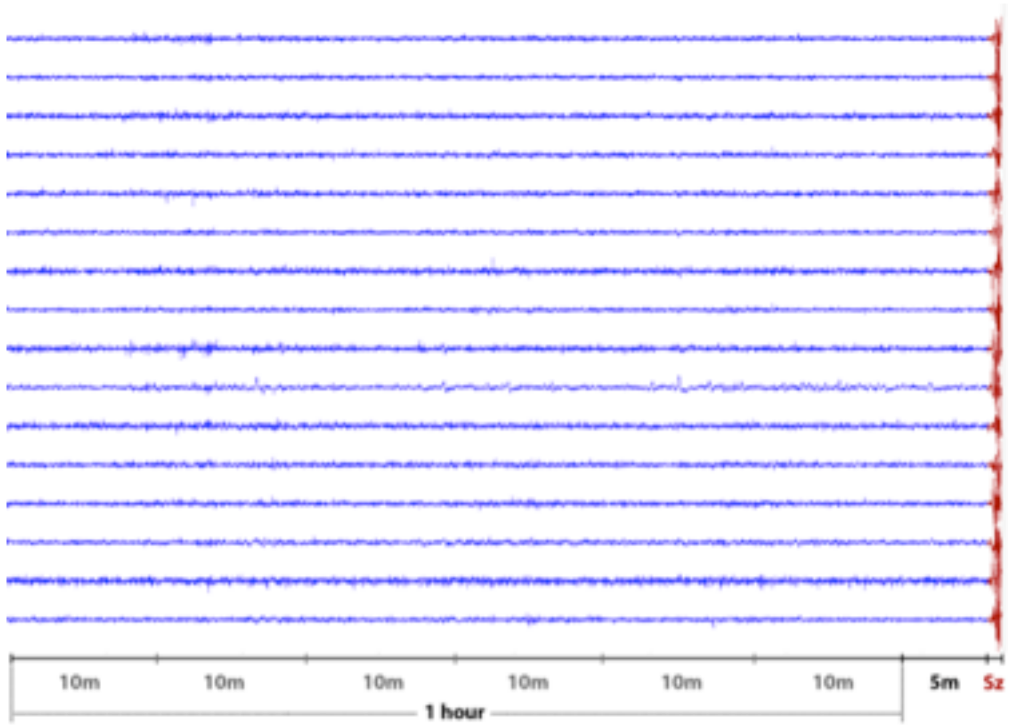

Figure 1: Preictal one-hour iEEG signal recording (16 channels)

Preictal data segments are provided covering one hour prior to seizure with a five-minute seizure horizon. (i.e. from 1:05 to 0:05 before seizure onset.)
The interictal data segments were chosen randomly from the full data record, with the restriction that interictal segments be at least 4 hours away from any seizure.
It has to be noted that some data segments contain 100% data drop-out and the data may also contain artifacts such as large amplitude rapid signal transitions which need to be sorted.

The full train set available is composed of recordings for 3 patients.
The present analysis is carried out on a subset corresponding to the patient 3 and composed of:

- 150 preictal samples of 10 minutes each, grouped in sequences of 6 successive samples over one full hour of recording, for a total of 25 hours,
- 1872 interictal samples, of 10 minutes each, grouped in sequences of 6 successive samples over one full hour of recording, for a total of 312 hours,
- an extra dataset, containing 105 additional preictal sample of 10 minutes each, unrelated to any sequence of recording.

I use this training set as a train/test dataset for validation of my model. The test set is composed of one-fifth of the organized preictal samples and one-sixth of the interictal samples. The train set contains all the remaining samples.
One challenge in this project is to build a representative CV. The 10-minutes preictal samples are provided for continuous periods of 1 hour. Similarly, interictal samples are provided for continuous periods of 1 hour. Considering that the activity state of the patient, its environment during the recording, and other conditions can have an effect on the EEG signal and to avoid contamination between these one-hour recordings, the split between the test and training sets has to preserve the integrity of one the hour long preictal sequences that correspond to 6 successive 10-minute samples to avoid potential contamination during cross-validation.

Data processing:

In this project the ten-minute-long data segments are divided in 30-second long epochs that are considered independently. For each epoch, each of the 16 channels are processed separately and then considered together as one training sample. The classification prediction of a 10-minutes samples is the result of the predictions over the 20 corresponding epochs.

EEG rhythmic activity is traditionally divided into frequency groups by neurologists for analysis. These groups seem to be related to mental activity, as rhythmic activity within a certain frequency range was noted to have a certain biological significance.

Δ waves < 4 Hz = Deep sleep, (ref. 0)
θ waves = 4-7 Hz = Light sleep, (ref. 1)
α waves = 8-14 Hz = Awake with eyes closed, (ref. 2)
β waves = 15-30 Hz = Awake and thinking, interacting, doing calculations, etc. (ref. 3)
γ waves = 30+ Hz = Might be related to consciousness and/or perception

Most of the cerebral signal observed in the scalp EEG, or its energy, falls in the range of 1–30 Hz. For this reason, I decide to calculate the frequency features of the signal within each of these frequency bands, to be able to differentiate potential changes occurring at each frequency level in a preictal state. In particular, the gamma band (particularly around 40 Hz) has been hypothesized to be a sign of correlation between different areas of the brain and even has been suggested to be a marker of consciousness or attention. I divide further the gamma band into 3 bands:

- 30 – 45 Hz, (ref. 4)
- 45 – 70 Hz, (ref. 5)
- 70+ Hz, (ref. 6)

There are 7 frequency bands considered in total in this project, referenced from 0 to 6.

The data samples are processed into a set of features using multiple signal treatment methods. This allows to efficiently compare the signal samples. The features are described below, details and formulas are provided in Annex 1.

First the signals are processed through a Discrete Fourier Transform, using a Fast Fourier transform algorithm, to allow their spectral analysis. From that DFT, we calculate the power spectral density of the signal. The following features are calculated from the PSD:

- Relative Power Spectral Density for each frequency band (7 features),
- Shannon Entropy for each frequency band (7 features),
- Spectral Edge Frequency (50% at 50 Hz) (1 feature),
- Correlation between channels of the RPSD for each frequency band (7 features),

The signals are also processed through a discrete wavelet transform through the calculation of 6th order Daubechies wavelets (7 coefficients). That gives some temporal information in addition to spectral information [2]. From that DWT, we calculate the following features:

- power spectral density at each scale (7 features).

The signal dynamic signature is studied through the analysis of its fractal dimension using:

- Katz method (1 feature),
- Higuchi method (1 feature).

Another set of properties in the time domain is computed, the Hjorth Parameters (Activity, Mobility, and Complexity). They are commonly used in the analysis of EEG signals for feature extraction. The activity parameter represents the total power of the signal, the mobility parameter represents the mean frequency, and the complexity parameter represents an estimate of the bandwidth of the signal:

- Hjorth Activity (1 feature),
- Hjorth Mobility (1 feature),
- Hjorth Complexity (1 feature).

The temporal cross-correlation is calculated between the channels of an epoch with no time delay (1 feature).

Statistical features the data samples are also computed for each epoch:

- Mean (1 feature),
- Standard Deviation (1 feature),
- Kurtosis (1 feature),
- Skewness (1 feature).

This selection of features is inspired by the multiple approaches described in the current published researches [3] [4]. Studies have been carried out over limited dataset and no single solution has shown a decisive superiority on epilepsy seizure detection.

Last, those features values are squared and the results add as features.
The total number of features is 78, multiplied by the 16 channels.

2. Algorithm and Techniques

I propose to use a supervised learning approach to infer a classifier from the dataset that will be able to distinguish iEEG signal segment in a pre-seizure period from a baseline period. The prediction performance of such classifier can first be assessed against the test subset defined.

From different classifiers considered (including SVM, Adaboost, Random Forest), based on initial scores obtained, the selected algorithm for this project is XGBoost. XGBoost is an implementation of gradient boosted decision trees that has become very popular in Kaggle Machine Learning competitions in the last 2 years. It is designed for speed and performance and has been found to be faster than previous Gradient boosted trees algorithms. With XGBoost, the data is stored in units with columns sorted by feature value and computed once. Then this block structure allows to do multiple parallel processing of columns statistics for split finding [5].
The principle of ensembles is to use multiple models (weak learner) and combine their results.

The following parameters can be tuned to optimize the XGBoost classifier:

- max_depth, the maximum tree depth for base learners,
- learning_rate, the learning rate,
- n_estimators, the number of boosted trees to fit,
- gamma, the minimum loss reduction required to make a further partition on a leaf node of the tree base learners,
- min_child_weight, the minimum sum of instance weight,
- subsample, the subsample ratio of the training instance,

- colsample_bytree, the subsample ratio of columns when constructing each tree.

XGboost, and boosting classifier in general, are prone to overfitting and are not robust against outliers and noise.
In order to limit the risk of overfitting it is possible to consider bagging several models trained on subset of the features and to use averaging or voting between those models.

3. Benchmark Model

The model prediction performance is assessed by comparison with a Gaussian naive Bayes algorithm. Naive Bayes classifier performs generally well in many practical applications, are scalable with large dataset and can work well with noise. It will constitute a reasonable benchmark model.
The test set used for submission evaluation by Kaggle in the competition is composed of samples without label. As a first approach focusing on one patient, I use a labeled train/test dataset. The test set is composed of 20% of the subset samples with the same proportion of preictal and interictal samples as the train set. This method allows more control over the test dataset for improving the model. The model performance will be compared over the test set with the benchmark model performance. The constitution of the test set is described in the Data Exploration paragraph.
The Sklearn implementation of the Gaussian naive Bayes algorithm was used. The model has no parameter to tune.

METHODOLOGY

1. Data Preprocessing

The first task is to prepare the data for the analysis, since sequences of iEEG signals cannot be easily compared without some pre-processing.

The data pre-processing program implements the following workflow:

- Open and run through the .csv file listing the safe files from the dataset and their label,
- Successively open the safe *.mat (Matlab) files and convert them into pandas dataframes that can be handled more easily,
- Merge those dataframes,
- Remove from the dataframes the rows containing no or only null values,
- Subdivide the iEEG signals into 30-second-long subsamples (representing 12 000 datapoints per channel with the 400Hz sampling rate),
- Remove the subsamples containing less than 80% of signal recording without drop-off, since it is could disturb the estimation of the features (10-minutes samples may therefore be split in less than 20 epochs),
- Process a Fast Fourrier Transform (use a Numpy function) of the samples,
- Process a Discrete Wavelet Tranform (use a pywt function),
- Calculate features from the sample DFT and DWT that are characteristics of the signal:
  - Shannon Entropy for each spectral band,
  - Relative Power Spectral Density for each spectral band,
  - Spectral Edge Frequency,
- Calculate features directly from the signal:

    - Hjorth parameters (Activity, Mobility, Complexity)
    - Kurtosis
    - Skewness
    - Fractal Dimension (Katz, Higuchi)
    - Mean
    - Standard deviation

  Those characteristics are used to differentiate EEG signals in neurological studies, the project will determine which are useful for seizure-risk identification.
- Calculate feature scaling using the MinMax method,
- Calculate squared features
- Store the dataframes created in .csv files for future use. There is one file for each one of the 16 channels, and each table row corresponds to an epoch sample. This helps to flexibly organize the data for training and classification.

The preprocessing of the data is initiated with the script "sample_processing.py", that iterate through the selection of *.mat sample recordings, calls the script calculating the features for each sample, collects in dataframes those features (one row per epoch and per channel) and exports to .csv format files the dataset obtained.

The calculation of the features itself is done using the script "converter.py", that opens a 10-minute samples *.mat file, extracts 30-second successive epochs, controls whether each epoch contains enough data, calculates the features and returns a python dictionary of the calculated features.

Finally, the feature scaling is done by the script "feature_post_processing.py", that calculates the squared features and applies the sklearn min_max_scaler function to all the features, and saves the dataset in *.csv format.

2. Implementation

The implementation process includes the following steps:

- Features vector construction from the calculated features,
- Train-test split and cross-validation strategy,
- Classifier training,
- Test,
- Metrics evaluation,

The programs for the training of the classifier implements the following workflow:

- Open the .csv preprocessed datasets as pandas dataframes,
- Concatenate the selection of features corresponding to each one of the dataset sub-samples,
- Split the dataset between a training set and a test set.
- Partition further the training set into cross-validation subsets to implement a 5-folds cross-validation for model optimization
- Implement the XGoost classifier parameter tuning using 5-folds cross-validation and return the configuration giving the best F1 score (step detailed in the paragraph "Refinement"),
- Define the parameters for one of the following classifiers:
    - XGBoost

    - Gaussian Naïve Bayes (the benchmark solution)
- Train the selected classifier over the train set,
- Save the trained model for later use,
- Make predictions using the selected classifier over the test set,
- Save those predictions and return the evaluation of the metrics.

Training/test set:

One of the complexity of this project was to define how to divide and use the test samples provided. As detailed in the Data Exploration paragraph, the original 10-minute test samples were divided into 30-second successive sub-samples, and features where calculated for each channel of each sub-samples.
During the course of the project I have tested different methods to construct the feature vectors. After testing those options, I decided to organize the dataset with one row representing one sub-sample and containing the features corresponding to all 16 channels of this sub-sample. With this method, predictions for a 10-minute test sample would be the result of predictions over its consecutive subsamples.
My intuition was that characteristic signs in a signal of a preictal state would potentially not be continuous over a full sample period, but rather localized and discrete. Therefore, on one hand the detection of shorter time pattern could be improved, but on the other hand the prediction rate would be lower for the sub-samples class. I chose then to use the highest probability for the preictal class prediction over the different epochs of a 10-minute sample to predict the class of the whole sample. This assumption proved to be correct with systematically in a higher recall value for the prediction of the 10-minute test samples than for the 30-second subsamples.

The dataset is split as follow to preserve the integrity of the one-hour long sequences:

- 5-folds split for the 150 10-minute preictal samples corresponding to 25 hours of recording, without shuffling the training sub-samples.
  One fold is kept for the test set.
- 6-folds split for the 1872 10-minute interictal samples corresponding to 312 hours of recording, without shuffling the training sub-samples.
  One fold is kept for the test set.
- The remaining group of 105 10-minute preictal samples that are not sorted is kept in its entirety.

The process of preparing the dataset for the train and test tasks is implemented by the script "train_test_cv_split.py".

Models:

The solution is a combination of 3 XGBoost models trained over 3 distinct subsets of features for the same samples. The reason for it was to help reduce overfitting. The probability estimates for the sample class was calculated as a simple average of the probabilities of predictions of the 3 models.

Each model was trained and used to predict the sub-samples class and then the highest probability prediction amongst each sample's sub-samples was attributed to the sample. Then the probability prediction for this sample was calculated as the average prediction issuing from the 3 models.
The 3 sets of features are the following:

- 1st Model features: {'shannon entropy'},
  The features set, including the squared features, is composed of 16 x 14 features.
- 2nd Model features: {'PSD wavelet', 'mean', 'standard deviation', 'hjorth activity', 'hjorth mobility', 'hjorth complexity', 'skewness', 'kurtosis', 'Katz FD', 'Higuchi FD'}
  The features set, including the squared features, is composed of 16 x 32 features.
- 3rd Model features: {'power spectral density', 'correlation matrix (frequency)', 'spectral edge frequency', 'correlation matrix (channel)'}
  The features set, including the squared features, is composed of 16 x 32 features.

The script "model_train.py" defines an XGBoost model and its parameters, loads the train set and fit the model to it, returns and saves the metrics score for subsample class predictions over the test set, and saves the trained model in *.pkl format.

The script "model_test.py" loads a trained model and the test set, makes the prediction for the subsamples classes and returns the metrics score. Then it computes the full samples class prediction and returns the metrics score for the test samples class predictions. The prediction tables are saved as a *.csv file with as columns the samples names, the predicted class, the prediction probabilities, and the true sample class.

The script "predict.py" implements the bagging of the 3 models predictions by averaging for each sample the prediction made by the 3 models, and returns the metrics score.

Metrics:

The metrics chosen for the were evaluated for each of the following steps:
- on the sub-samples class predictions for the cross-validation evaluation (see paragraph "refinement of the models"), implemented in the script "model_cv.py",
- on the the sub-samples class predictions on the test set (see paragraph "models evaluation and validation"), implemented in the script "model_test.py",
- on the sample class prediction on the test set (see paragraph "models evaluation and validation"), implemented in the script "model_test.py",
- on the final prediction combining the 3 models for the test set (see paragraph "models evaluation and validation"), implemented in the script "predict.py".

The metrics regarding the sup-samples class prediction were used to assess the robustness of the models between the cross-validation test sets and the test set.

The metrics regarding the samples class prediction were used to assess the performance of the solution at predicting preictal states from iEEG signals as defined in the problem statement.

3. Refinement

To improve the solution, the XGBoost classifier models were refined from the initial solution based on default values for the parameters.

First, the XGBost models used where defined with the following default parameter values:

{'n_estimators': 100, 'max_depth': 3, 'min_child_weight': 1, 'gamma': 0, 'subsample': 1, 'colsample_bytree': 1, 'learning_rate': 0.1}

Those parameters were tuned in this order to increase the performance of the solution.

The models performances were evaluated using a cross-validation method over 5-folds of the training set. The cross-validation split of the train set is defined in the paragraph "Implementation".
Each of the parameters listed above was modified within a range of values and the parameter value giving the best metrics scores (primarily the F1 score) was kept.

The script "model_cv.py" implements this refinement process, starting from a python dictionary of parameters values, evaluating the average metrics over the 5 cross-validation train/test folds and returning the best set of parameters values and the metrics score.

The initial performance for each model was measured in terms of F1 score, recall and AUC. Those results are particularly low (Table 1) because of the small number of estimators, which was the first parameter to be tuned, and the limited depth for decision trees.
A first increase of the number of estimator to 500 and of the "max_depth" parameter to 4 improves sensibly the performance of the models (Table 2).

Metrics	1st Model	2nd Model	3rd Model
F1 score	0.086	0.086	0.086
Recall on preictal samples	0.057	0.055	0.055
ROC AUC	0.517	0.517	0.517

Table 1: Metrics score for CV achieved by the initial models

Metrics	1st Model	2nd Model	3rd Model
F1 score	0.198	0.198	0.198
Recall on preictal samples	0.167	0.167	0.167
ROC AUC	0.559	0.559	0.559

Table 2: Metrics score for CV achieved by the models (n_estimators=500, max_depth=4)

After preliminary trials, the cross-validation test was run over the following sets of parameters for the models:

1st Model: {'max_depth':[6, 7, 8, 9], 'learning_rate': [0.1, 0.08, 0.06],
'n_estimators': [100, 500, 600, 800, 1000],
'gamma':[0, 0.1, 0.2, 0.3], 'min_child_weight':[4, 6, 8, 10],
'subsample':[0.5, 0.6, 0.7, 0.8], 'colsample_bytree':[0.8, 0.9]}

2nd Model: {'max_depth':[6, 7, 8, 9], 'learning_rate': [0.1, 0.08, 0.06],
'n_estimators': [100, 500, 600, 800, 1000, 1200],
'gamma':[0, 0.1, 0.2, 0.3], 'min_child_weight':[4, 6, 8, 10],
'subsample':[0.5, 0.6, 0.7, 0.8], 'colsample_bytree':[0.8, 0.9]}

1st Model: {'max_depth':[6, 7, 8, 9], 'learning_rate': [0.1, 0.08, 0.06],
'n_estimators': [100, 500, 800, 1000, 1200, 1500],
'gamma':[0, 0.1, 0.2, 0.3], 'min_child_weight':[6, 8, 10, 12],
'subsample':[0.5, 0.6, 0.7, 0.8], 'colsample_bytree':[0.8, 0.9]}

RESULTS

1. Model Evaluation and Validation

The final models and parameters were chosen according to their performance amongst the tried parameters combinations. The performance was assessed according to the mean test F1 score over 5-folds cross-validation subsets.
To validate the robustness of the models, their performance was assessed on the test set, independent from the training set, and their metrics score recorded (Table 3, 4 & 5).

1st model:

The parameters of the first XGBoost model are the following:
{‘max_depth’: 9, ‘learning_rate’: 0.1, ‘n_estimators’: 1000, ‘gamma’: 0.1, ‘min_child_weight’: 10, ‘subsample’: 0.6, ‘colsample_bytree’: 0.7}

Metrics	CV mean test score	Test score
F1 score	0.228	0.270
Recall on preictal samples	0.212	0.735
ROC AUC	0.576	0.668

Table 3: Metrics score for CV and test achieved by the 1st model

2nd model:

The parameters of the second XGBoost model are the following:
{'max_depth': 9, 'learning_rate': 0.1, 'n_estimators': 1200, 'gamma': 0, 'min_child_weight': 6, 'subsample': 0.4, 'colsample_bytree': 0.8}

Metrics	CV mean test score	Test score
F1 score	0.225	0.337
Recall on preictal samples	0.212	0.823
ROC AUC	0.575	0.747

Table 4: Metrics score for CV and test achieved by the 2nd model

3rd model:

The parameters of the third XGBoost model are the following:
{'max_depth': 9, 'learning_rate': 0.1, 'n_estimators': 1500, 'gamma': 0.2, 'min_child_weight': 12, 'subsample': 0.5, 'colsample_bytree': 0.8}

Metrics	CV mean test score	Test score
F1 score	0.226	0.276
Recall on preictal samples	0.214	0.735
ROC AUC	0.575	0.674

Table 5: Metrics score for CV and test achieved by the 3rd model

The results obtained with the test set are in all 3 cases a little better for all 3 metrics than the average score obtained with the cross-validation folds.
This improvement can be caused by two factors:

- First, the models making prediction on the test set are trained on a larger dataset than during cross-validation, since it includes the whole train set, leading to a higher prediction capacity.

- Secondly, all preictal samples may not present a comparable richness of indicators of a coming seizure. This risk is limited but exists here since the test set contains preictal samples covering only 5 different periods of 1-hour of recording. As a reminder, the train set contained 20 different periods of 1-hour of recording plus some additional unsorted samples that could not be place in the test set.

The results are a first confirmation that the models generalize well on the unknown test dataset.

The final solution is a combination of the 3 models and implements a simple "bagging" of the models by averaging their predictions, with also interesting predictions capabilities (Table 4).

Metrics	Score
F1 score	0.327
Recall on preictal samples	0.794
ROC AUC	0.731

Table 4: Metrics score achieved by the final solution on the test set

The 2nd model outperforms the two other models on the test set and the bagging therefore does not improve the highest score. We cannot conclude though that this would be the case on any new dataset since the cross-validation scores were very close between all 3 models.
The relatively modest F1 score can be explained by the imbalanced dataset, with 10 times more interictal samples than preictal ones, while the false positive prediction rate equals 33%.
The recall score confirms the good potential of the solution at predicting the positives samples.

2. Justification

The performance of the final solution was compared to a benchmark solution implementing 3 Naïve Bayes models (NB) trained on the same dataset.

Metrics	Score
F1 score	0.218
Recall on preictal samples	0.823
ROC AUC	0.602

Table 5: Metrics score achieved by the benchmark model on the test set

The final solution achieves a higher F1 score than the benchmark solution (Table 5). The higher recall value achieved by the NB model comes with a very high rate of false positive predictions, the specificity of the model being equal to 0.38.
With a comparable capability for positive case prediction, the final solution outperforms largely the benchmark model at predicting the negatives samples and hence constitute a clear improvement. A high specificity is essential for a practical use of such a solution, since the patients spent most of their time in interictal state. It remains the main weak point of the final solution that would need to be improved, being equal to 0.66 here.

Conclusion

1. Free-form visualization

The features utilized in this project are selected from a variety of previous studies on seizure detection from EEG recordings. There is no consensus yet as of what features are the most interesting to consider when trying to predict seizures, which is the reason why I selected a rather wide range of features for the present project.
One of the interesting outcome of this project is the possibility to evaluate the respective importance of the features for each of the models developed. This would be a starting point for trying to improve the solution by defining a reduced set of more relevant features, and by refining the bandwidth on which the features were calculated.

A bar chart can appropriately display the relative importance of the features for each model. A few features show a significantly greater discriminating power, but the study of the 100 more important features also shows none of them should be neglected (fig.2, fig.3 & fig.4).
In all the graphs, the first number of a feature name indicates the channel it represents, the last number, if any, indicates a frequency bandwidth from 0 to 6 as defined in the paragraph "Data processing". This is the number to consider when evaluating which features are globally more interesting since it indicates a specific type of features, rather than the channel number.

The analysis of the cumulative importance over the 16 channels of each feature for each model provide the following information:

- For the 1st model, no feature stands out as less relevant,
- For the 2nd model, the power spectral density from wavelet for the frequency band no 6 is a less relevant feature.
- For the 3rd model, the power spectral density for the frequency band no 6 is a less relevant feature.
- For all models, the squared features provide less information.

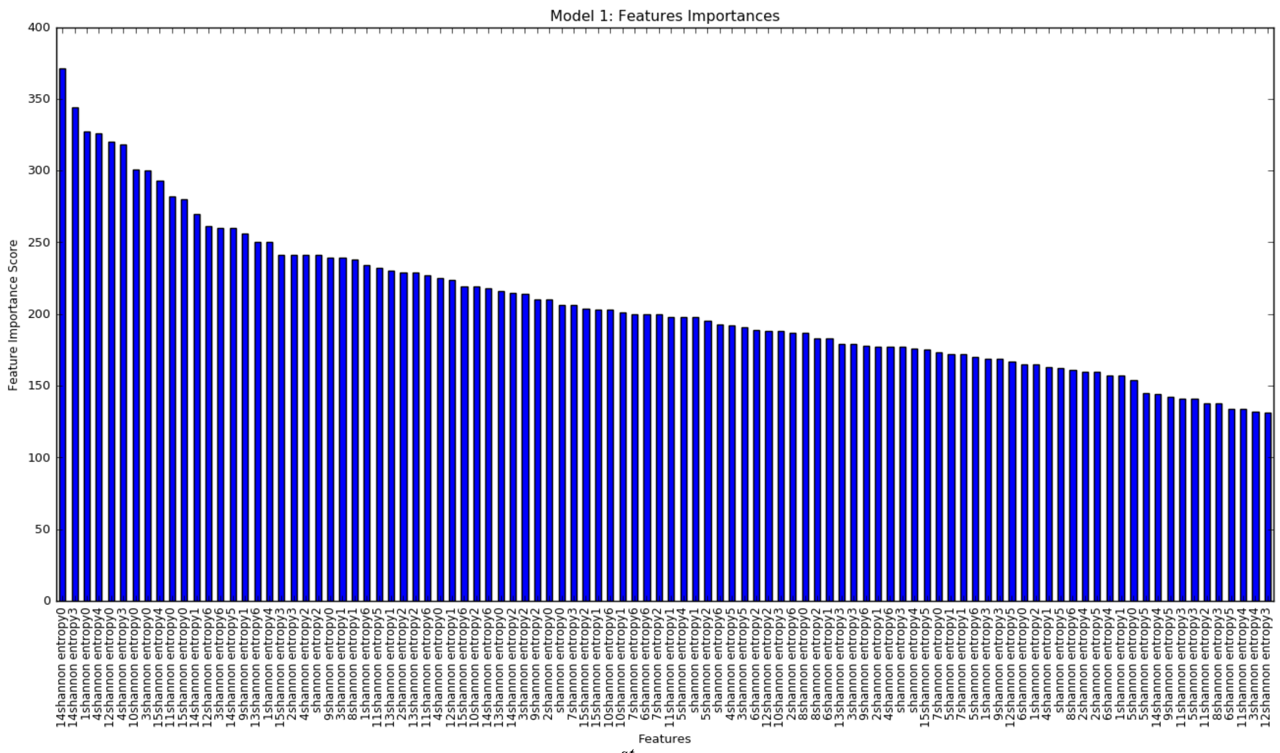

Figure 2: Feature importance for the 1^{st} model 100 most important features

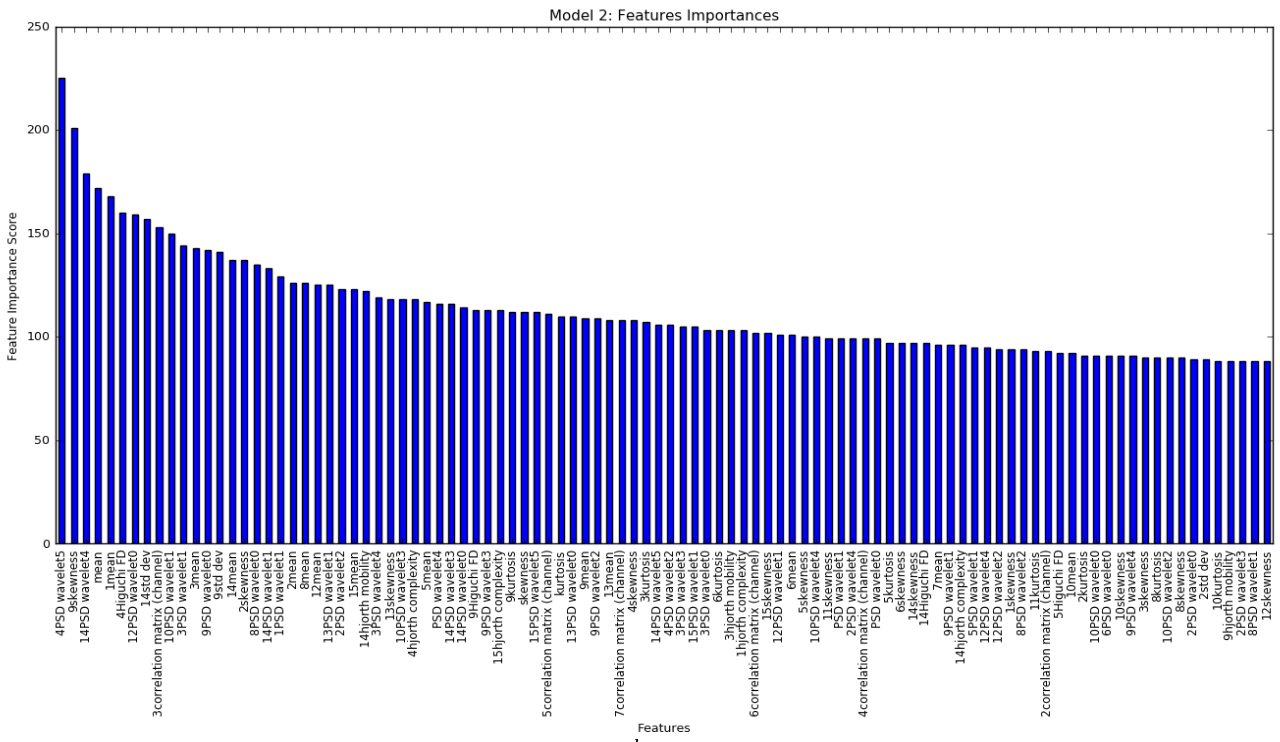

Figure 3: Feature importance for the 2nd model 100 most important features

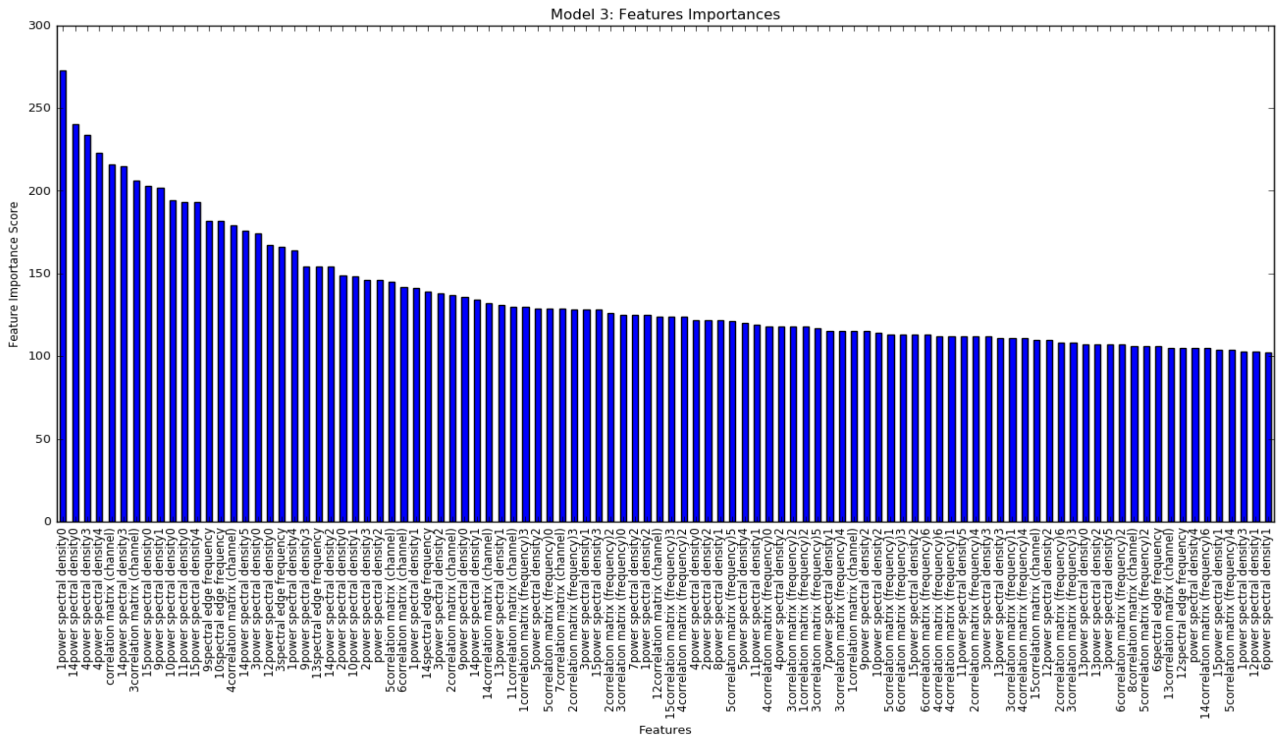

Figure 4: Feature importance for the 3rd model 100 most important features

2. Reflection

The process I used for this project included the following main steps:

- Analysis of the dataset,
- Study of the current progress in the field of epileptic seizure detection from EEG,
- Study of processing techniques for signal analysis,
- Definition of data processing and features calculation,
- Selection of a classifier and of a benchmark classifier,
- Definition of the train-test split and cross-validation strategy,
- Definition of the evaluation metrics,
- Training and refinement of the models,
- Test of the final solution.

The task of defining a proper processing for the data made available for the competition proved to be the most challenging and interesting one. Researchers have long been interested in using EEG signal in the case of seizure detection and multiple features have been studied accordingly. I took the time to understand the meaning and usage of each features and made a selection of promising ones. I had to rework my feature and reprocess the samples, which was time consuming, after my first attempt which did not provide a satisfying result.

The next step was the selection of a model and a benchmark solution. For the latter, I used the suggestion made after my Capstone proposal to use a naive Bayes model. This helped me understand how to define a base model when lacking an appropriate benchmark solution.

The implementation of a representative cross-validation and train/test split method was also a challenging task in this project. The difficulty came from the necessity to preserve the integrity of the 1-hour segments of recording within each cross-validation and test sets. This meant keeping all the 30-second sub-samples obtained after processing the six 10-minutes samples composing an hour of recording into one test or train set. The lack of preictal samples compared to the negative samples, with 25 hours of preictal recording, limited the possibilities. I my first attempt, not complying with this rule resulted in over optimistic recall scores due to pollution of the datasets.

3. Improvement

Some future improvements can be considered on the implementation used for this project regarding the data processing:

- The use of more advanced signal filtering techniques could help retrieving and removing potential artifacts in the signal, while identifying spikes related to seizure prediction [6]. This is a subject of research in itself in the domain of EEG study. Noise reduction filters could be used as well. It should be tested in comparison of this project results, since there is a risk of removing valuable information from the signal when denoising the signal.
  The gain would be to reduce false positive detection from non-relevant artifacts.
- A more discretized signal analysis along narrower frequency bands would provide a finer representation of the signal characteristics. Then, a thorough research through the possible combinations of features to evaluate their relevance could help define a better set of features for seizure detection.

An important axis for improvement to consider is how the solution generalizes to other patients. The present project was conducted on a single patient and demonstrated that preictal states could be detected using iEEG signals for this patient.
A test of the final solution, carried out on some of the data available from the patient 1 (test set composed of 131 preictal samples and 310 interictal samples), gave lower performance results than those with the patient 3 test set. The recall on preictal sample for the bagged solution was equal to 0.237.
Interestingly, the 3 models performed very differently on the patient 1 test set:

- The recall for model 1 was equal to 0.78 with a low specificity of 0.25
- The recall for the 3rd and 2nd model was equal to 0.25 and 0.37 with a higher specificity of 0.65 and 0.78.

The characteristics of EEG signals and the potential markers of a preictal state probably differ from one patient to another. The next step for improving the solution should be to repeat the process implemented in this project on multiple patients, and then to analyze the features and their relevance for those patients to try to develop a more global solution. The most efficient solution may then be to retrain a classifier for each patient.

Annex 1: Features description

Relative Power Spectral Density

The power spectral density represents the distribution of power into the frequency components composing that signal.
It can be calculated as the square of the absolute value of the Fourier transform:

$$S_{xx} = |X(f)^2|$$

where $X(f)$ is the discrete Fourier transform of the signal for frequency f.

Here we consider the PSD value Pj for each frequency band j relative to the total PSD value over the range 0-180 Hz:

$$Pj = \frac{Ej}{Etot}$$

where Ej is the sum of the energy over one frequency band and $Etot$ the sum of the energy over all the frequency bands.

Shannon Entropy

The Shannon entropy (Shannon, 1948) gives a useful criterion for analyzing and comparing probability distribution, it provides a measure of the information of any distribution. It appears as a measure of the degree of order/disorder of the signal, or unpredictability.
The formula for calculating Shannon Entropy for the power spectral density is:

$H = -\sum_{i=1}^{n} P_i \, log(P_i)$, with P_i the relative power spectral density

Spectral Edge Frequency

The spectral edge frequency is the frequency at which a certain percentage of the signal power lies below in the signal's frequency domain. Here we calculate the frequency below which 50% of the signal power within the range 0 – 50Hz lies.
It is calculated

Correlation

Correlation is the measure of similarity between two signals. Its value is comprised between +1 and -1.
For both the temporal cross-correlation and the relative power spectral density correlation between the channels, I use the Pandas implementation of Pearson standard correlation, that compute pairwise correlation of columns.

pandas.DataFrame.**corr**(method='pearson', min_periods=1)

Relative wavelet energy

Time-frequency analyses such as wavelet analysis allow to resolve the spectral content with some temporal resolution. Features extracted in the time-frequency domain provide information about the spectral and temporal content simultaneously and have been widely used in the tracking of spectral changes in EEG time series [1][2][7].

The wavelet is a smooth and quickly vanishing oscillating function (fig.) and a wavelet family is the set of elementary functions generated by dilations a multiple resolution levels and translations of a unique admissible mother wavelet.

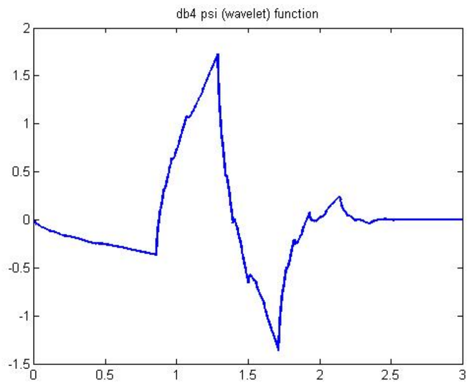

Figure 2: Daubechies wavelet example

The discrete wavelet transform (DWT) provides a representation of the signal and its values are the coefficients in a wavelet series.
The energy of the the signal at each resolution level can be expressed has:

$$Ej = \sum \left| C_j \right|^2$$, with C_j the wavelet coefficient of resolution level j

The relative wavelet energy at a scale j is equal to:

$$Pj = \frac{Ej}{E_{tot}}$$, with E_{tot} the sum of the energies at each scale.

Here, I use a discrete Daubechies wavelet transform 6th order, using the “wavedec” function of the pywt python library to calculate the coefficients:
pywt.**wavedec**(*data = epoch*, *wavelet = ‘db6’*, *mode=’symmetric’*, *level=None*, *axis=-1*)

The function returns an ordered list of coefficients.

Fractal dimension (FD)

A fractal dimension (FD) is an index for characterizing fractal patterns or sets by quantifying their complexity as a ratio of the change in detail to the change in scale.
The FD thus measures the rate of addition of structural details with increased magnification, scale or resolution and therefore can be seen as a quantifier of complexity. FDs of the EEG computed in this manner will always be between 1 and 2 since it characterizes the complexity of the signal considered in the 2-dimensions plane of voltage and time. A signal with lower complexity has a FD close to 1 and for a more complex signal FD is close to 2.
A number of methods have been reported for estimating the FDs of EEGs, here we focus on 2, Katz’ and Higushi’s FD [8][9].

Katz FD:
In the Katz's algorithm, FD is obtained directly from the time series and is defined as:

$$D = \frac{log_{10}(L)}{log_{10}(d)}$$

where L is the total length of the EEG time series and d is the Euclidian distance between the first point in the series and the point that provides the furthest distance with respect to the first point.

Higuchi FD:
Consider x(1), x(2), ..., x(N) the time sequence to be analyzed. Construct k new time series as

$$X_m^k = \left\{X(m), X(m+k), \dots, X(m + \left\lfloor \frac{N-m}{k} \right\rfloor k\right\}, for\ m = 1, 2, \dots, k$$

where m indicates the initial time value, k indicates the time interval between points (delay), $\lfloor a \rfloor$ means integer part of a.

For each of the curves or time series X_m^k constructed, the average length $L_m(k)$ is computed as:

$$L_m(k) = \left(\sum_{i=1}^{\left\lfloor \frac{N-m}{k} \right\rfloor} |X(m+ik) - X(m+(i-1)k)|\right)\left(\frac{N-1}{\left\lfloor \frac{N-m}{k} \right\rfloor k}\right)$$

where N is the length of the time series X(t).

An average length is computed for all the time series having the same delay (or scale) , as the mean of the k lengths $L_m(k)$ for m = 1, .. k.

This procedure is repeated for each k ranging from 1 to k_{max}, and the sum of average lengths $L(k)$ for each k equals:

$$L(k) = \sum_{m=1}^{k} L_m(k)$$

The total average length for scale k, $L(k)$, is proportional to k^{-D}, where D is the FD by Higuchi's method. In the curve of ln $(L(k))$ versus ln $(1/k)$, the slope of the least squares linear best fit is the estimate of the fractal dimension.

Hjorth parameters

The Hjorth Parameters (Activity, Mobility, and Complexity) are commonly used in the analysis of EEG signals for feature extraction. The activity parameter represents the total power of the signal, the mobility parameter represents the mean frequency, and the complexity parameter represents an estimate of the bandwidth of the signal.
The parameters are calculated using the following formulas:

$$Hjorth\ activity = var(y(t))$$

$$Hjorth\ mobility = \sqrt{\frac{var(\frac{dy(t)}{dt})}{var(y(t))}}$$

$$Hjorth\ complexity = \frac{Hjorth\ mobility(\frac{dy(t)}{dt})}{Hjorth\ mobility(y(t))}$$

The temporal cross-correlation is calculated between the channels of an epoch with no time delay (1 feature).

Kurtosis

Kurtosis is the fourth central moment divided by the square of the variance. This number measures how long the tail is of a probability density function. For this measure, higher kurtosis means more of the variance is the result of infrequent extreme deviations, as opposed to frequent modestly sized deviations.
This project implements the Scipy implementation of the Kurtosis calculation based on Fisher's definition:

scipy.stats.**kurtosis**(data, axis=0, fisher=True, bias=True)

Skewness

Skewness measures how asymmetric a probability density function is around its mean value.
This project implements the Scipy implementation of the Skewness calculation:

scipy.stats.**skew**(data, axis=0, bias=True, nan_policy='propagate')

References:

[1] Cook MJ, O'Brien TJ, Berkovic SF, Murphy M, Morokoff A, Fabinyi G, D'Souza W, Yerra R, Archer J, Litewka L, Hosking S, Lightfoot P, Ruedebusch V, Sheffield WD, Snyder D, Leyde K, Himes D (2013) Prediction of seizure likelihood with a long-term, implanted seizure advisory system in patients with drug-resistant epilepsy: a first-in-man study. LANCET NEUROL 12:563-571.

[2] Xiaoli Li. Wavelet Spectral Entropy for Indication of Epileptic Seizure in Extracranial (2006). 6 – 73

[3] Kais Gadhoumi, Jean-Marc Lina, Florian Mormann, Jean Gotman; Seizure prediction for therapeutic devices: A review. (2016) 270–282

[4] Andrzejak RG, Chicharro D, Elger CE, Mormann F (2009) Seizure prediction: Any better than chance? Clin Neurophysiol.

[5] Tianqi Chen, Carlos Guestrin. XGBoost: A Scalable Tree Boosting System (2016)

[6] Karoly, P. J., Freestone, D. R., Boston, R., Grayden, D. B., Himes, D., Leyde, K., ... & Cook, M. J. (2016). Interictal spikes and epileptic seizures: their relationship and underlying rhythmicity. Brain, aww019.

[7] Rosso OA1, Blanco S, Yordanova J, Kolev V, Figliola A, Schürmann M, Başar E. Wavelet entropy: a new tool for analysis of short duration brain electrical signals (2001) 105(1):65-75.

[8] Cindy Goh, Brahim Hamadicharef, Goeff Henderson, Emmanuel Ifeachor. Comparison of Fractal Dimension Algorithms for the Computation of EEG Biomarkers for Dementia. 2nd Inter- national Conference on Computational Intelligence in Medicine and Healthcare (CIMED2005), Jun 2005, Lisbon, Portugal. <inria-00442374>

[9] R. Esteller ; G. Vachtsevanos ; J. Echauz ; B. Litt. A Comparison of Waveform Fractal Dimension Algorithms (2002). 1057-7122