Predict seizures in intracranial EEG Recordings

Project Overview

Problem Statement- From Kaggle

Seizure forecasting systems hold promise for improving the quality of life for patients with epilepsy.

Epilepsy afflicts nearly 1% of the world's population, and is characterized by the occurrence of spontaneous seizures. For many patients, 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 -- and even after surgical removal of epilepsy-causing brain tissue, 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.

Seizure forecasting systems have the potential to help patients with epilepsy lead more normal lives. In order for EEG-based seizure forecasting systems to work effectively, computational algorithms must reliably identify periods of increased probability of seizure occurrence. If these seizure-permissive brain states can be identified, devices designed to warn patients of impeding seizures would be possible. Patients could avoid potentially dangerous activities like driving or swimming, and medications could be administered only when needed to prevent impending seizures, reducing overall side effects.

There is emerging evidence that the temporal dynamics of brain activity can be classified into 4 states: Interictal (between seizures, or baseline), Preictal (prior to seizure), Ictal (seizure), and Post-ictal (after seizures). Seizure forecasting requires the ability to reliably identify a preictal state that can be differentiated from the interictal, ictal, and postictal state. The primary challenge in seizure forecasting is differentiating between the preictal and interictal states. The goal of the competition is to demonstrate the existence and accurate classification of the preictal brain state in dogs and humans with naturally occurring epilepsy.

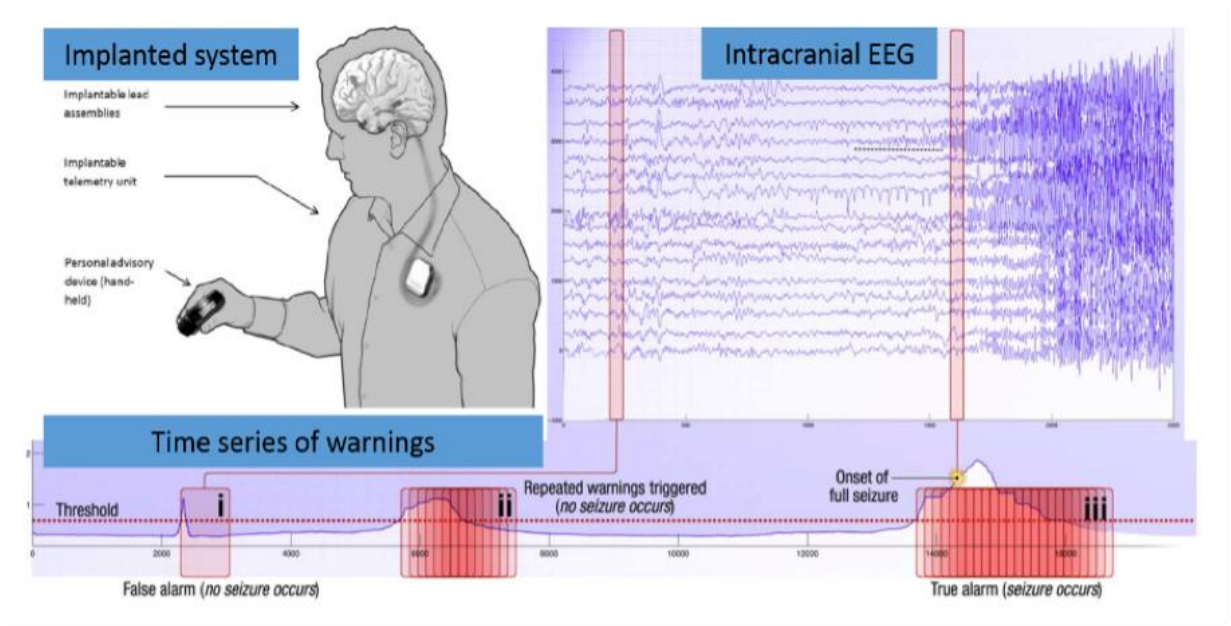


Figure: Seizure forecasting systems

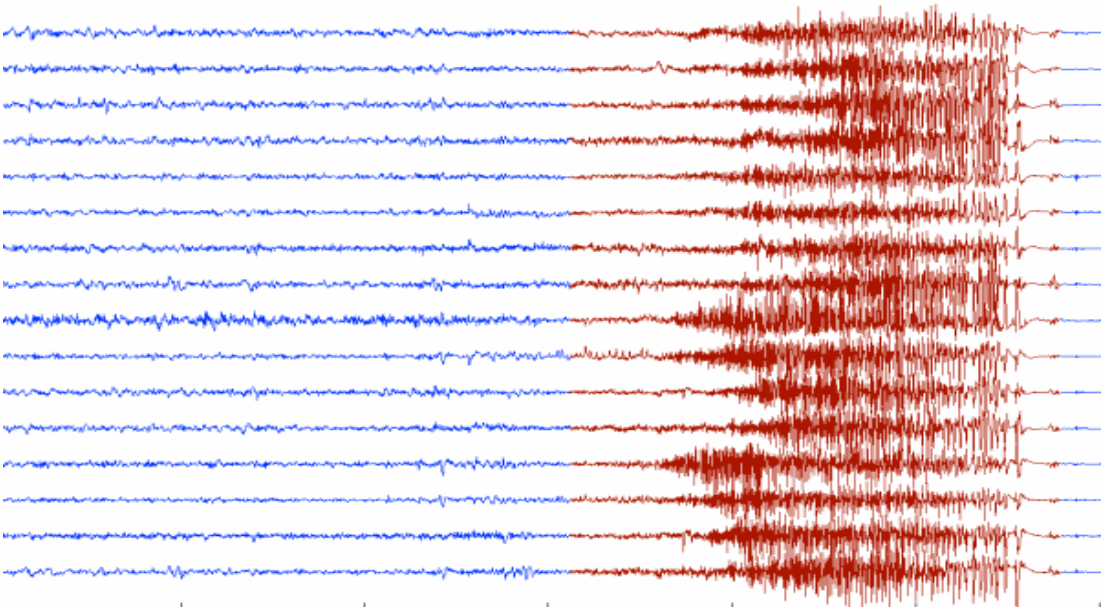


Figure: Increased brain activity during Seizure shown in Red.

Scientific background

Epilepsy is the most common cerebral disorder. It affects over 65 million people and researchers say between 15-20 percent of death is due to to sudden unexpected causes. Epliepsy is characterized by spontaneous seizures due to abnormal synchronization of neural activity. The disease can be controlled in 2/3 of the patients, in the remainder the unpredictability of seizures leads to major impairments with inherent physical risks of loss of consciousness and motor

control. During the last ten years, attempts have been made to predict seizures from EEG data using linear and non-linear time series analysis.

However, the significance of results regarding clinical applicability and neurophysiological adequacy remains uncertain. Most studies, including ours on intracranial long term EEG recordings from patients undergoing presurgical evaluation, applied univariate measures of cortical dynamics, not taking into account the neuronal synchronization underlying the development of ictal activity. Thus, the investigation of synchronizing non-linear dynamical systems has become a major focus in recent studies on time series analysis and applying these methods for analyzing data from such systems.

Definition of Common Terms

Ictal refers to a physiologic state or event such as a seizure.

Pre-ictal refers to the state immediately before the actual seizure, stroke, or headache, though it has recently come to light that some characteristics of this stage (such as visual auras) are actually the beginnings of the ictal state

Post-ictal refers to the state shortly after the event.

Interictal refers to the period between seizures, or convulsions, that are characteristic of an epilepsy disorder. For most people with epilepsy, the interictal state corresponds to more than 99% of their life. The interictal period is often used by neurologists when diagnosing epilepsy since an EEG trace will often show small interictal spiking and other abnormalities known by neurologists as subclinical seizures. Interictal EEG discharges are those abnormal waveforms not associated with seizure symptoms.

The goal of the project is really simple to understand. Create a model to distinguish between EEG signals right before a seizure and the signals between seizures.

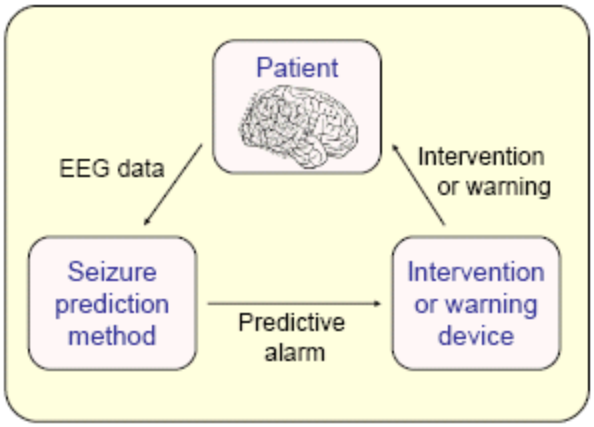


Figure: Basic functioning of a future closed-loop intervention device: The long-term goal of the project is to develop algorithms that are able to predict epileptic seizures with high sensitivity and specificity. These prediction algorithms could be utilized in a "brain defibrillator", in analogy to cardiac defibrillators. A prediction of seizures at an early stage could trigger an intervention to suppress the upcoming seizure by for instance electrical stimulation. Alternatively, a seizure warning device could be invented that enables behavioral adjustments.**[1]**

Data

All data are derived from one hour blocks as described in the data page. Each one hour block is divided into six 10 minute data segments. For the interictal data the one hour blocks may be temporally discontinuous relative to each other as they are selected by a random process. For the preictal data the one hour blocks are taken with respect to the known seizure times.

For the training data, the interictal and preictal data are temporally ordered by index J from the file name **'I_J_K.mat'**. The temporal ordering only applies within each class K (interictal K=0, preictal K = 1), there is no way to determine the temporal relationship between interictal and preictal data, i.e. if one preictal segment occurs between two particular interictal segments. The 'sequence' field info helps you determine if adjacent 10 minute training segments (based on index J) belong to the same one hour block.

For the test data all 10 minute segments are randomized with respect to their temporal order (i.e. for the test data, index J in the filename **'I_J.mat'** has nothing to do with temporal order) and the 'sequence' field is missing.

[1]https://www.bcf.uni-freiburg.de/bccn/research/c2

Data Pipeline

A plan was formulated in the initial days of the project. I kept adding more steps for better feature engineering and improving performance. Therefore a few of the the steps were implemented in the later iterations.

- Interpolate the case when all channels are zero with mean values.(and remove files that have all zero values)
- **ICA**-to separate the signal to useful components.
- Extract features using **Wavelet Transformations.**
- Reduce dimension using **PCA**.
- Use **t-SNE** to further reduce dimensions and visualize seperability of data.
- Classification using different models **random- forest, SVM** etc. with oversampling.
- Obtain **AUC** and plot **ROC** Curve

Metrics used

The data is highly imbalanced- (A common situation in classification problems where the classes are not represented equally. Classification Accuracy is not the correct metric to use when working with an imbalanced dataset. It produces misleading results.)

Metrics used here:

- **AUC-Area Under the Curve ***

  It is used in classification problems in order to determine which of the models used predicts the classes best. It combines False positive rate (FPR) and the True positive rate (TPR), into one single metric.

- **ROC curve - Receiver operating characteristic.**

  A plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings

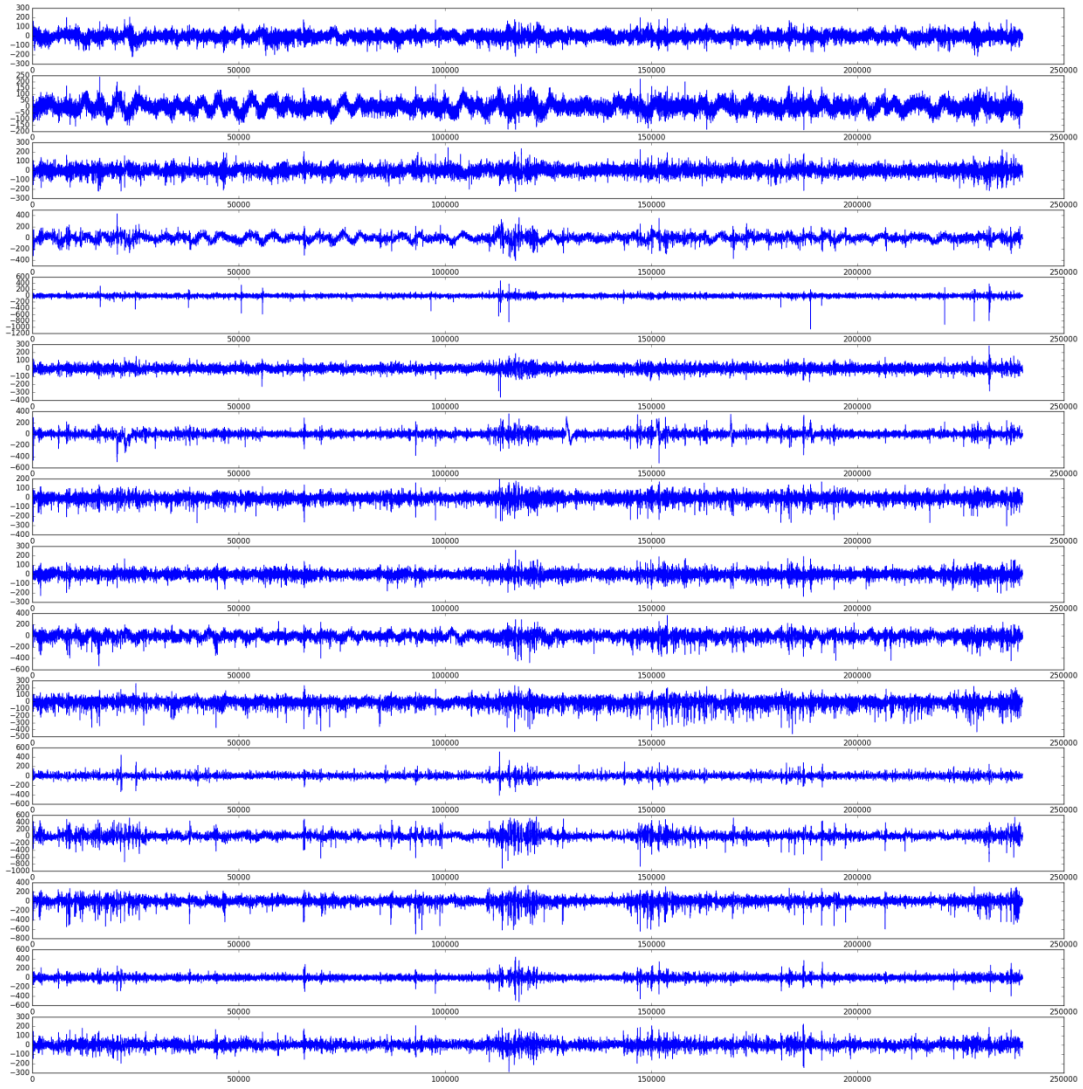
*The evaluation criteria for the Kaggle competition is AUC. So the primary goal is to obtain maximum AUC. Other metrics are only to get a better idea of the performance of the models.

Data Exploration

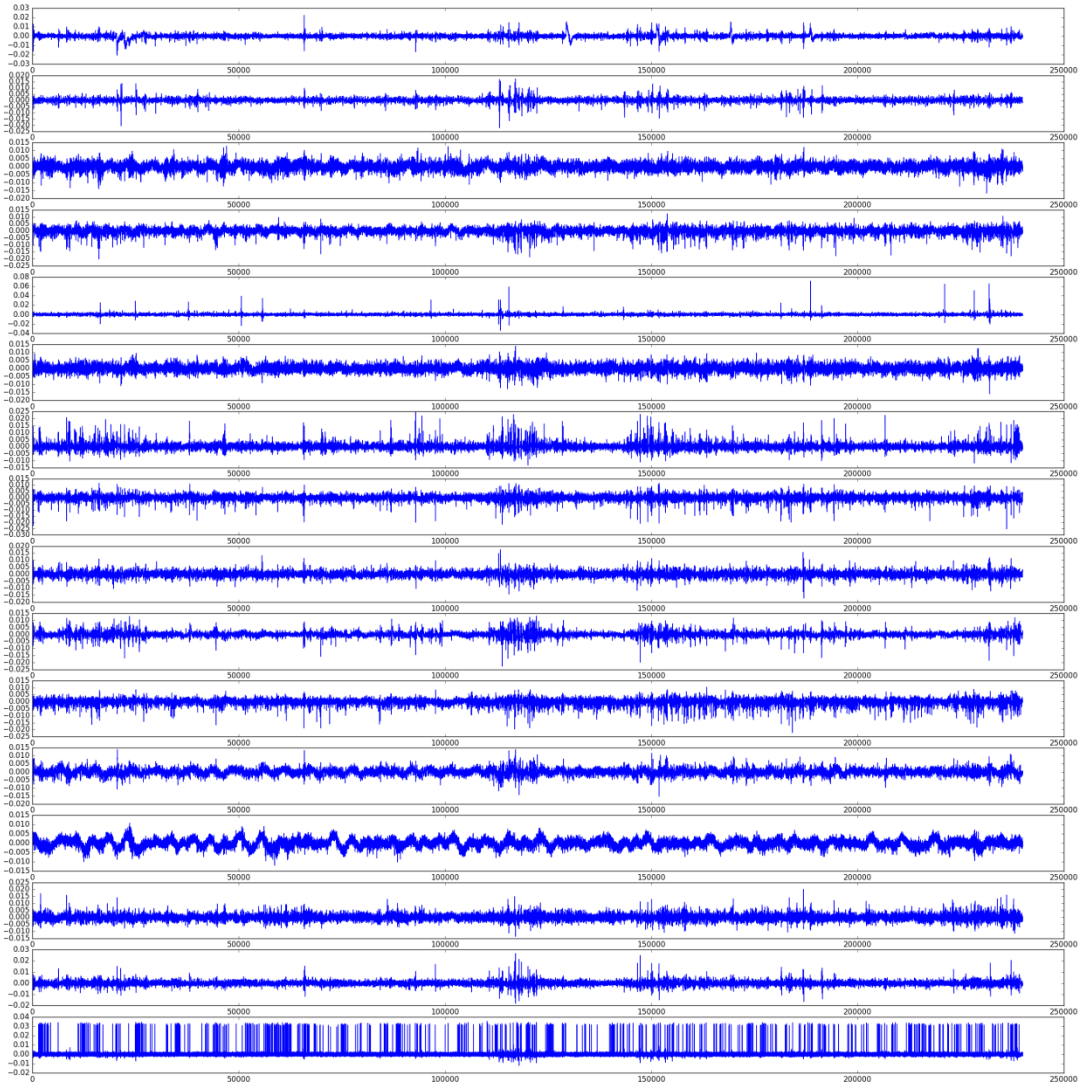
Intracranial EEG (iEEG) data clips are organized in folders containing training and testing data for each human patient. The training data is organized into ten minute EEG clips labeled "Preictal" for pre-seizure data segments, or "Interictal" for non-seizure data segments. Training data segments are numbered sequentially, while testing data are in random order. Within folders data segments are stored in .mat files as follow.

The transformed data was stored in a **npz** file so that time can be effectively reducing not having to perform PCA each time.

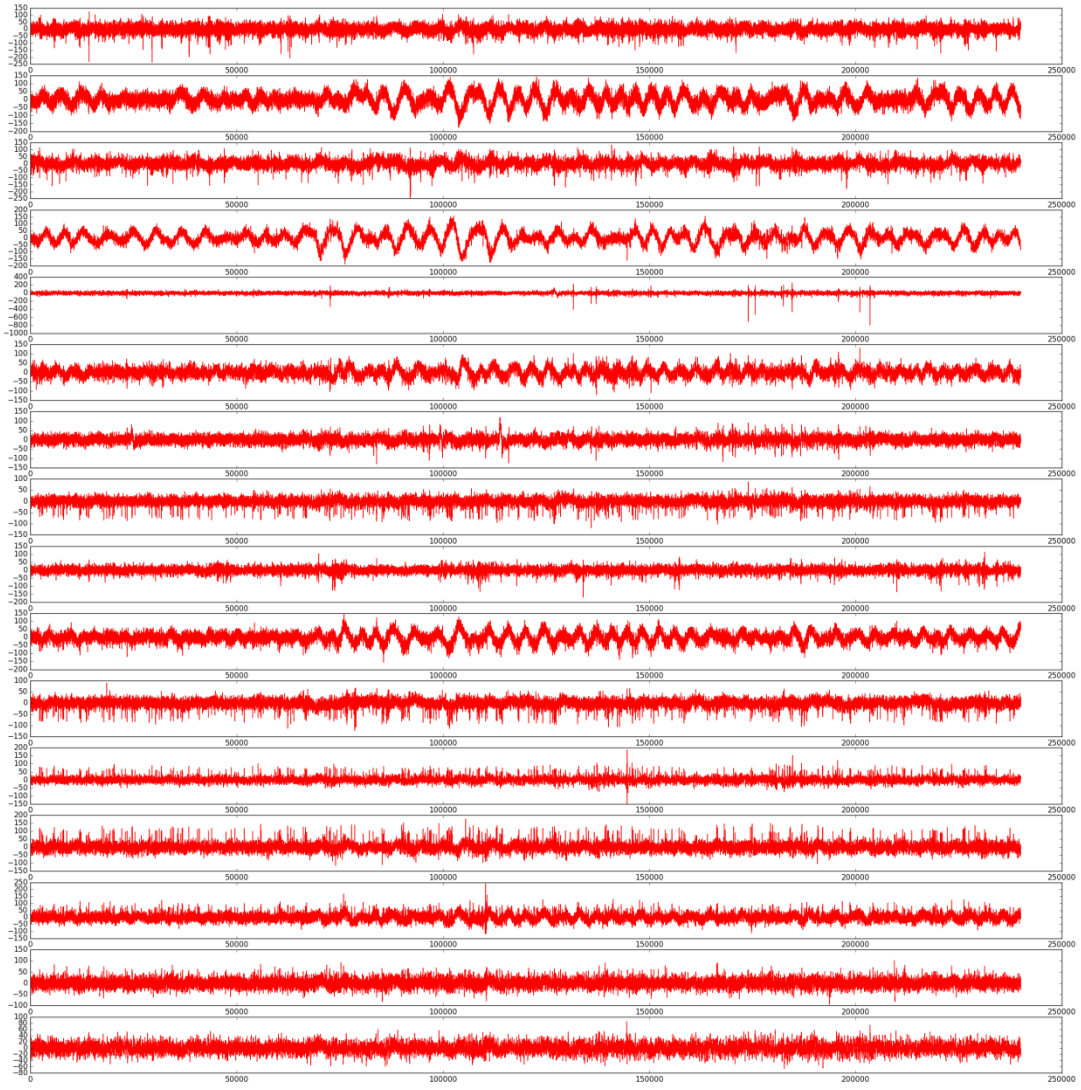
Visualizing raw signals from the 16 channels :
Interictal State



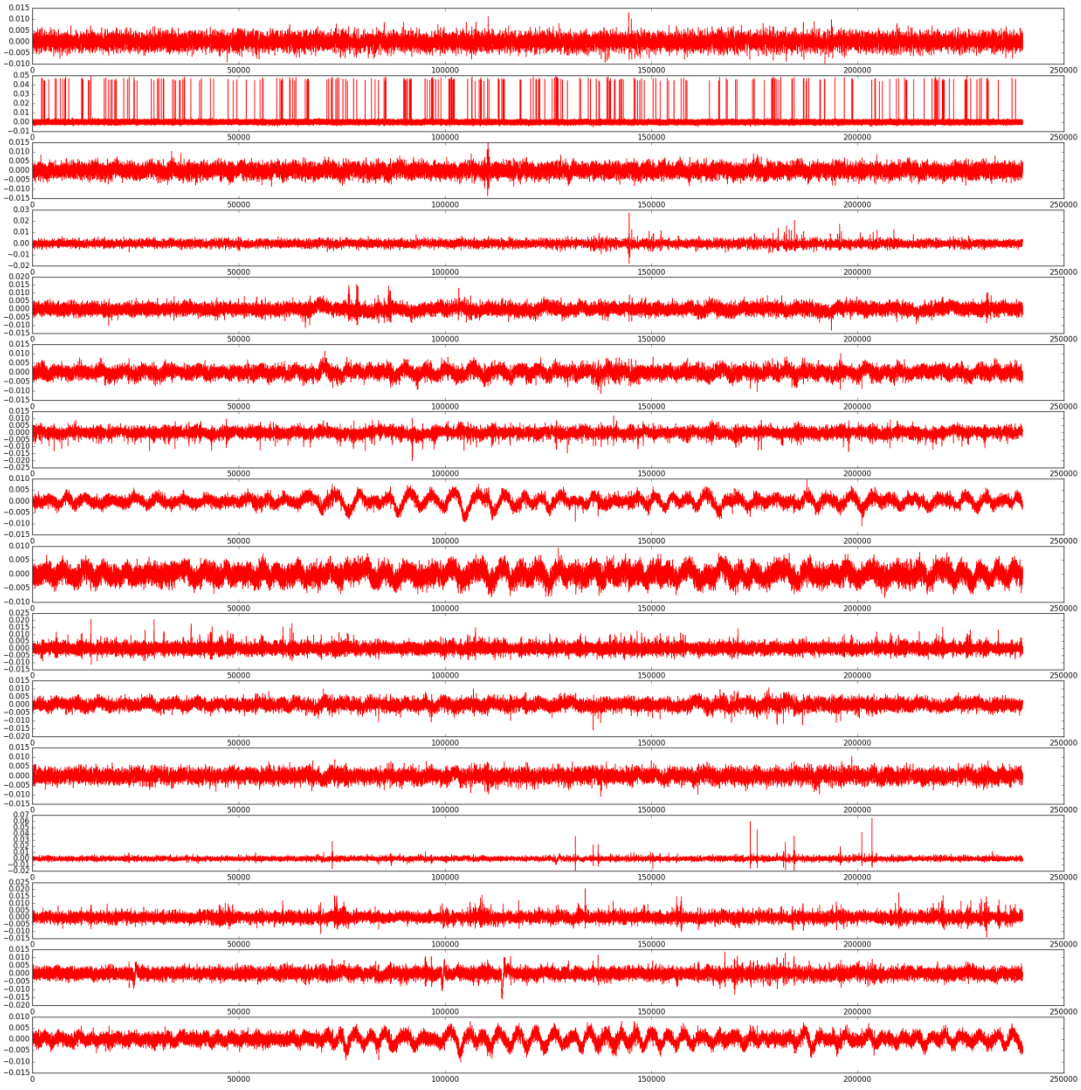
Visualizing raw signals from the 16 channels :
Interictal State after ICA



Visualizing raw signals from the 16 channels :
Preictal State



Visualizing raw signals from the 16 channels :
Preictal State after ICA



Methodology

The methodology can be summarized as- extract the best features, reduce noise and use Gridsearch to find best performance in different predictive models (such as ensemble models).

Benchmark

The minimum benchmark would be predicting that there is no difference between the 2 states. Predicting all zeroes or all ones leads to AUC of 0.5. This is equivalent to random guessing. The optimized predictive models in this study outperform the benchmark by a good margin

Algorithms and Techniques

Many of the competitors resorted to using fast Fourier transforms for the initial data extraction. I chose Wavelet transformation since it has some advantages over FFTs. After the transformation was performed the data was reduced to a supervised learning problem but with 60000 columns and 6041 data points.

Wavelet Transformations

The wavelet transform is a relatively new concept (about 10 years old). A key advantage it has over Fourier transforms is temporal resolution: it captures both frequency and location information (location in time).

Why use wavelets?

For testing and feature extraction, it is beneficial to down sample data so you have fewer number of samples, while still approximating the signal reasonably well. The wavelet-level parameter can be thought of as "down sampling level".

Principal Component Analysis

Principal component analysis (PCA) is a technique used to give emphasis to variation and bring out patterns in a dataset. It's often used to make data easy to explore and visualize.

After feature extraction PCA is the next natural step. Performing PCA we were able to reduce 60000 features to 5000 features by retaining 97% of the variance. This was really helpful because

the number of features is reduced to 8% of the original and this helps reduce the code runtime. The original data has a lot of noise and is very difficult to handle because of 60K columns.

t-SNE

t-distributed stochastic neighbor embedding (t-SNE) is another popular algorithm for dimensionality reduction and is very good at visualizing data separability. The t-SNE algorithm provides an effective method to visualize a complex dataset but main problem is that it is quadratic in the number of samples, which makes it unscalable to large datasets. So this is why PCA was performed first on the large dataset and then t-SNE. . I did not observe the separation between prerictal and interictal data that I expected to see. This is probably because of Data Leakage*.

Predictive Models used

Logistic Regression(LR)

Logistic regression is a supervised learning algorithm. This means that the relationship between the independent and dependent variables are obtained through training examples. The binary logistic model is used to estimate the probability of a binary response based on one or more predictor (or independent) variable.

I always start with a logistic regression for any classification problem as a general rule of thumb. It also provides for a slightly more challenging baseline than the one earlier specified. Logistic regression outputs the **probabilities of a specific class.** Those probabilities can be converted into **class predictions.**

However I do not plan on using LR as my final model because of the complexity of the data. LR is a pretty well-behaved classification algorithm that can be trained as long as you expect your features to be roughly linear and the problem to be linearly separable. I don't wish to go to more details about LR because of the poor performance of the umoptimized version of the model, I don't plan to proceed further with LR for the final model.

SVM

Support Vector Machines (SVMs) are a powerful supervised learning algorithm used for classification or for regression. SVMs are a discriminative classifier: that is, they draw a boundary between clusters of data.

SVM is a discriminative classifier formally defined by a separating hyperplane. In other words,

given labeled training data, the algorithm outputs an optimal hyperplane which categorizes new examples.

The main reason to use an SVM instead of a Logistic Regression is because the data is not linearly separable (which is observed in this problem). In that case, you will have to use an SVM with a non-linear kernel (e.g. RBF). One of the major downside of SVMs is that they can be painfully inefficient to train. I gave up on Gridsearch Kernel SVM because of the time taken for the fitting of the model. SVM is not recommended for most "industry scale" applications. The size of the data made it difficult to train even after using all the cores. So I stuck to using SVM with a linear kernel since it is difficult to optimize SVM for several different parameters.

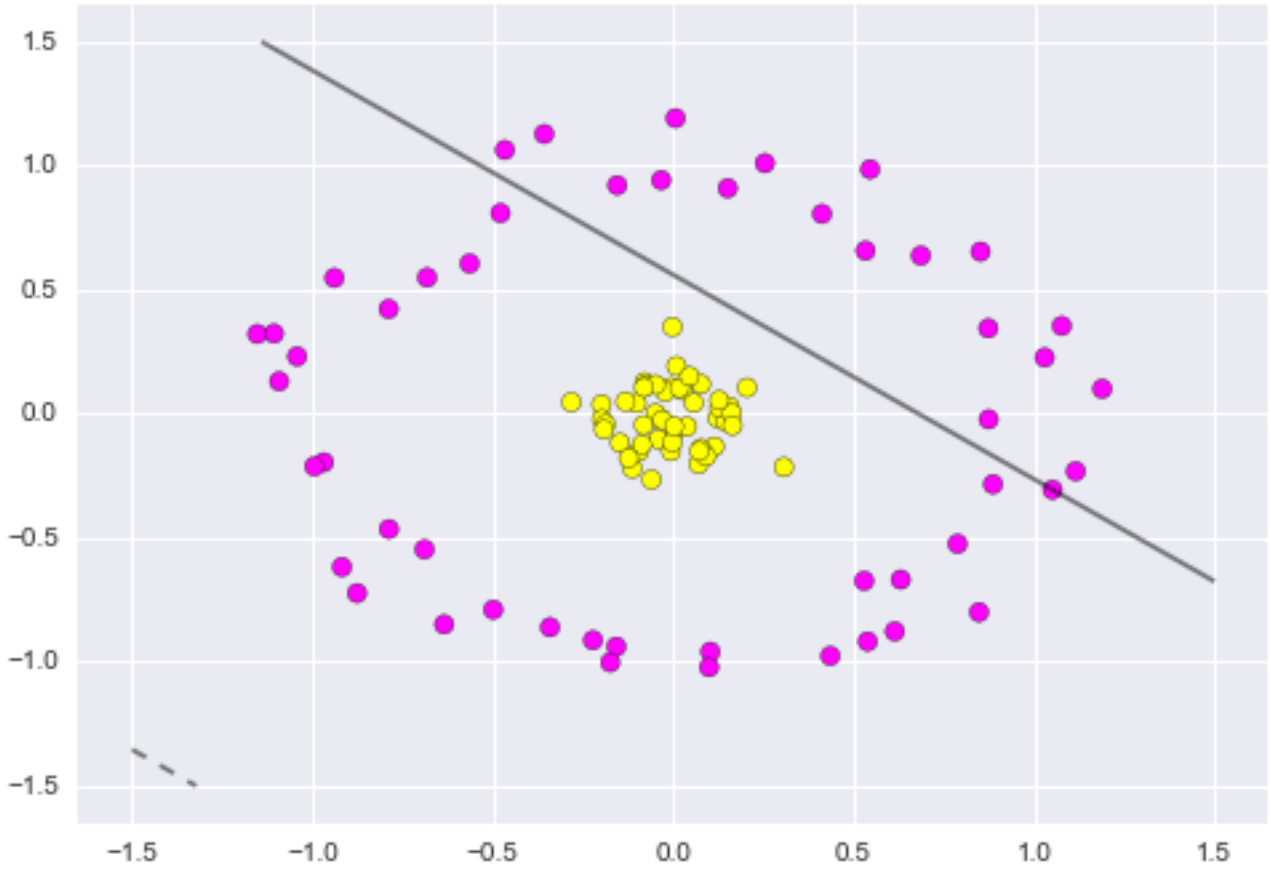


Figure: SVM with Linear Kernel. It doesn't work well when the data is not linearly separable. Model is not complex enough to fit the data. Clearly, no linear discrimination will ever separate these data. One way we can adjust this is to apply a kernel, which is some functional transformation of the input data.

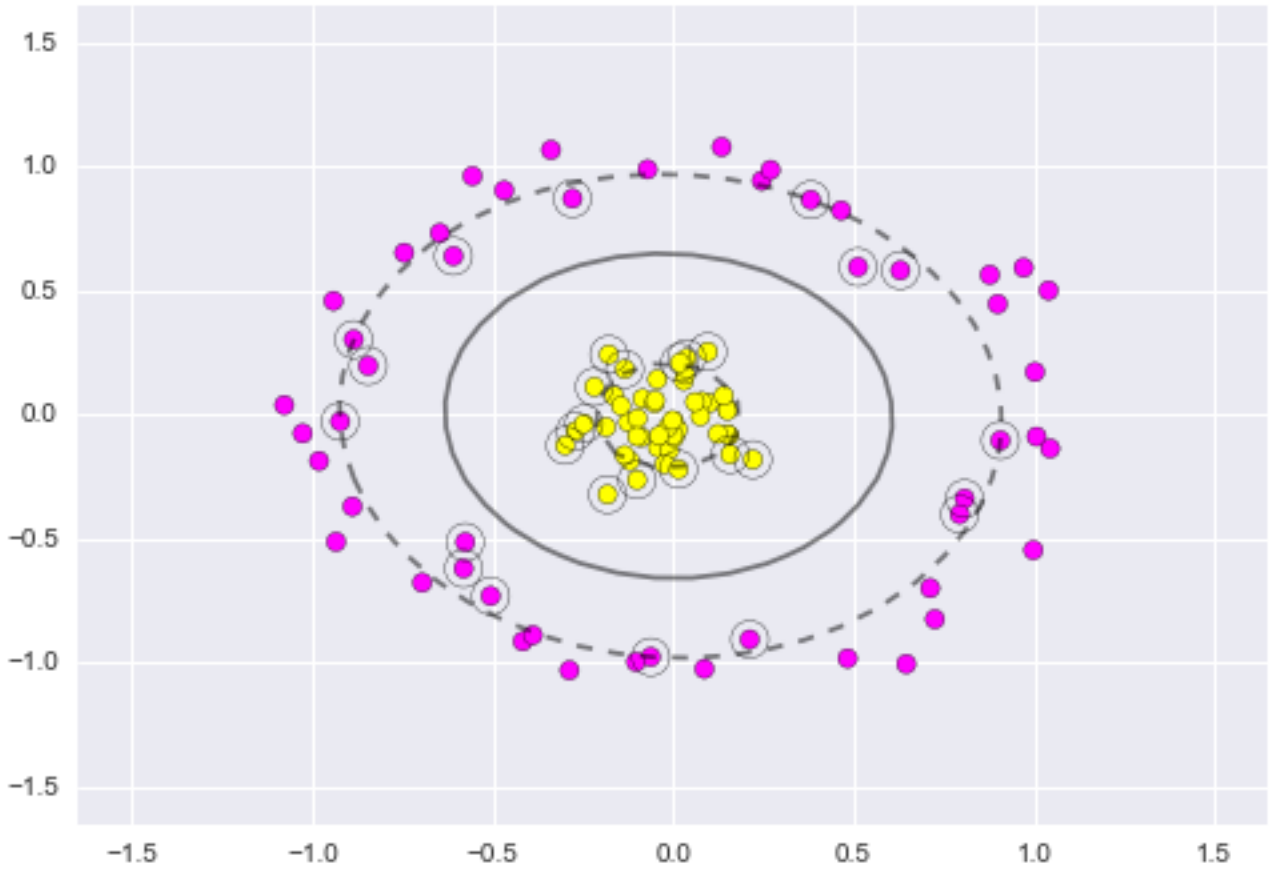


Figure: SVM with RBF Kernel. Kernel tricks allows for classification of highly complex datasets. How it works? Increase the dimensions of feature vectors if they are non-linearly separable in low dimensions. It's quite likely that they become separable in high dimensions

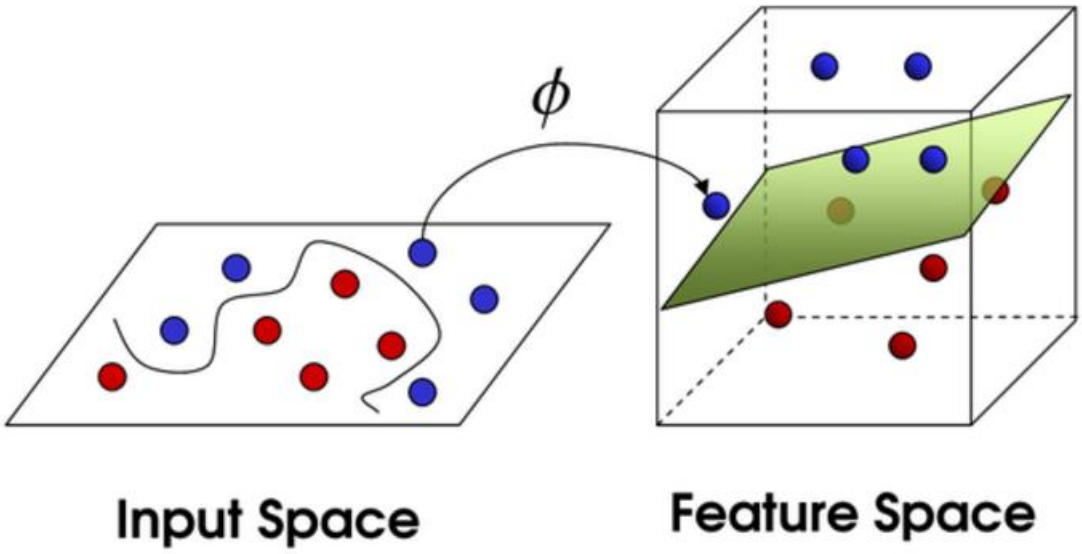


Figure: We can see that with the additional dimension, the data becomes trivially linearly separable. Kernel converts 2D data into 3D to make the data linearly separable

Random Forest

Random forest is a type of ensemble learning, as it relies on an ensemble of decision trees. Random decision forests correct for decision trees' habit of overfitting to their training set. RF will be discussed in greater detail since I try to optimize the different hyper parameters in the improvement section.

Ensemble Learning

Ensemble learning involves the combination of several models to solve a single prediction problem. It works by generating multiple classifiers/models which learn and make predictions independently. Those predictions are then combined into a single prediction that should be as good as or better than the prediction made by any one classifier.

Error can be divided into bias and variance. A too complex model has low bias but large variance, while a too simple model has low variance but large bias, both leading a high error for two different reasons. Using Ensemble techniques there are two ways to reduce error- variance reduction for a complex model, or bias reduction for a simple model, which refers to random forest and boosting respectively. Random Forest is a special case of Bagging. It reduces variance of a large number of "complex" models with low bias.

Bagging in RF

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. A random sample is selected with replacement from the training set and trees are fit to these samples. The models are fitted using the chosen bootstrap samples and combined by voting (for classification).

RF uses a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. This process is sometimes called "feature bagging". The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the the trees, causing them to become correlated.

RF offers a lot of flexibility in terms of parameter to optimize. They operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) of the individual trees. This process is exactly what you want to do if you want to minimize the variance & overfitting of your model class. That is they correct for decision trees' habit of overfitting to their training set. It has a distinct advantage over Logistic Regression that they do not expect linear features or even features that interact linearly.

Parameter Optimization

The main parameters to adjust are **n_estimators** and **max_features.** The former is the number of trees in the forest. The larger the better, but also the longer it will take to compute. The performance will stop getting significantly better beyond a critical number of trees. In this project I have reported the results for 20 trees and 200 trees (keeping other conditions constant . There is a significant difference between both the cases.

The latter is the size of the random subsets of features to consider when splitting a node. The lower the greater the reduction of variance, but also the greater the increase in bias. ‘Criteria’ is an argument that allows us to control how the decision tree algorithm splits nodes. Gini is intended for continuous attributes and Entropy for attributes that occur in classes. Gini” will tend to find the largest class, and “entropy” tends to find groups of classes that make up ~50% of the data. I have utilized both the criteria while using Gridsearch. ’Gini’ seems to be the better choice for this problem.

Parallelization

It is possible to have parallel construction of the trees and the parallel computation of the predictions through the n_jobs parameter. Given the size of the data it was very important to parallelize the work. By using n_jobs=-1 all cores available on the machine were used.

Data Preprocessing and Implementation

Challenges Faced

Size of the data was a big problem. Total data was 60 GB of .mat files (MATLAB format). Wavelet Transformation was used to extract the features and down sample the data.

Missing data-Analyzing the different files it was found out that some of the data in the files are all zero. In some cases it was partially filled with zeroes.

Note from the competition admin says "Dropouts are not predictive of preictal or interictal class. Segments with 100 % data drop-out can be ignored. Segments with less than 100% drop-out should be analyzed because they contain data."

Data-imbalance problem. Let's say a person has one seizure a month. This means **10000:1** Interical to Ictal time*.

Overcoming Challenges

Size of the data-

The size of the data was significantly reduced as compared to the original data size by taking the following steps.

1. Wavelet Transformation for feature extraction
2. PCA- for dimension reduction
3. t-SNE-- for dimension reduction

Missing data-

Steps taken to combat the problem:

1. Visualize raw signals from the 16 channels to look for unusual behavior.
2. All files with zero values were determined and removed.
3. All files with partial missing data was interpolated at the missing locations

Data-imbalance problem

Overcoming this problem was crucial to build a meaningful model.

1. Oversampling the minority class.

Results improved significantly after this step. The AUC of all the models were around 0.5-0.55 before this step. This approach is really simple. Copies of instances from the preictal class were added (under-represented class) till they reached a significant proportion as compared to the larger class.

2. Undersampling the majority class.

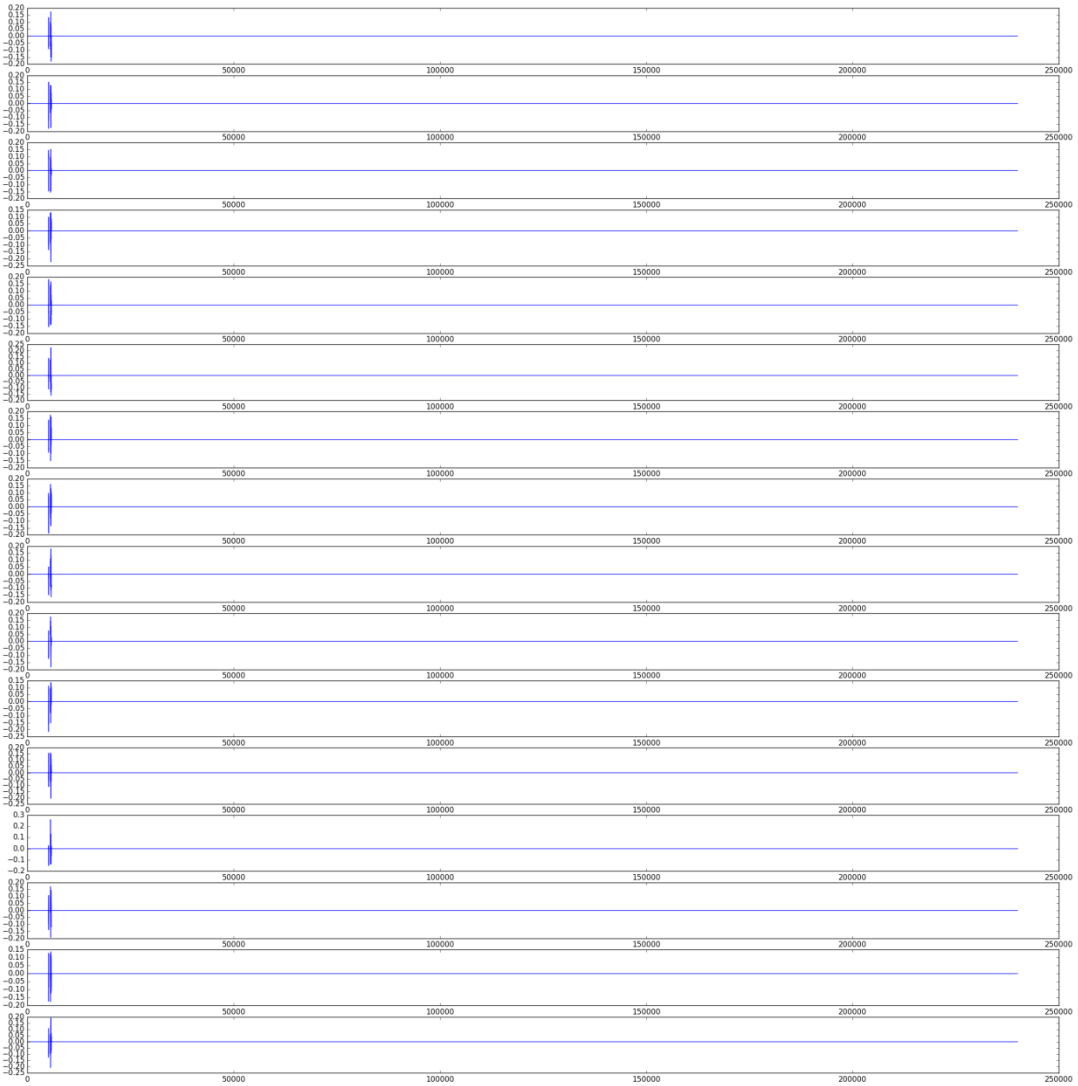
I considered this approach but avoided it because this would lead to a very big loss in information. The smaller class is very small compared to the large class. The ratio is 7:93. This approach is equivalent to dropping 86(93-7) points of Interictal class for every 7 points of the preictal class to obtain perfect class balance.

Approach for Future Work

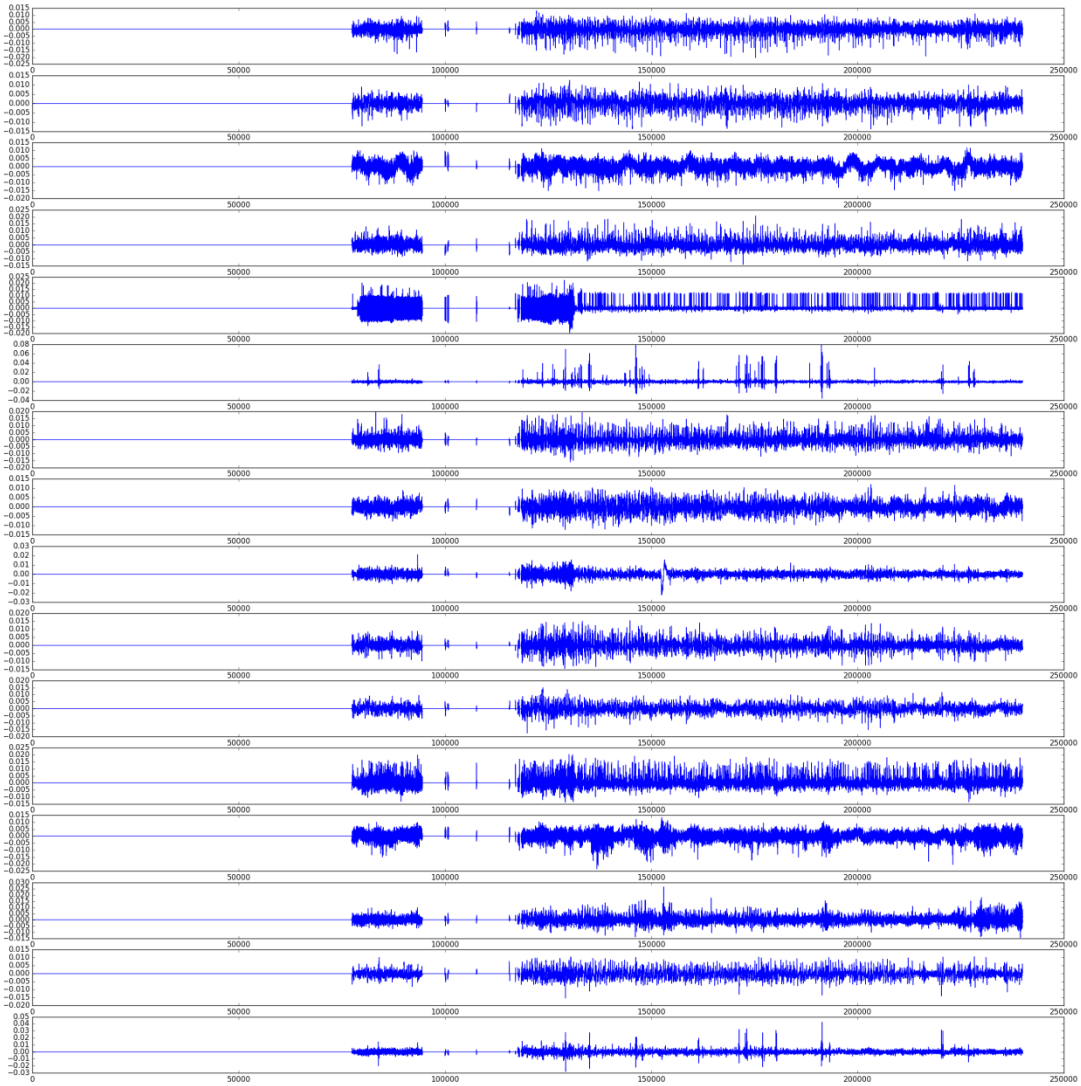
SMOTE or the Synthetic Minority Over-sampling Technique.

There are systematic algorithms that you can use to generate synthetic samples. SMOTE is an oversampling method. It works by creating synthetic samples from the minor class instead of creating copies. The algorithm selects two or more similar instances (using a distance measure) and perturbing an instance one attribute at a time by a random amount within the difference to the neighboring instances.

One of many Interictal files with corrupt data :

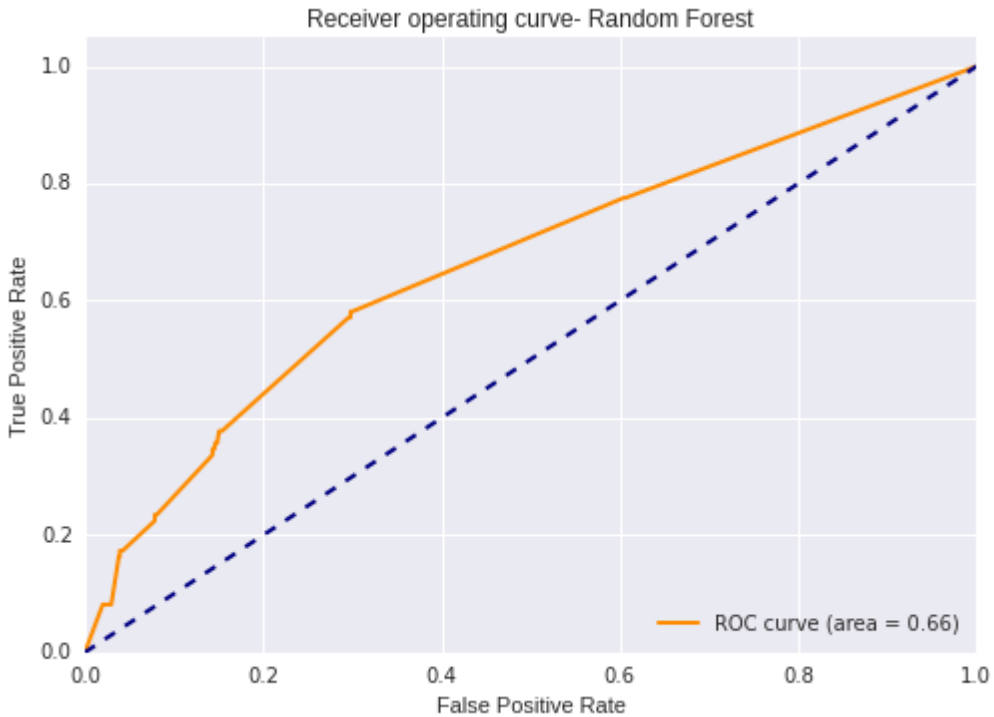


One of many Interictal files with Partial Dropout :

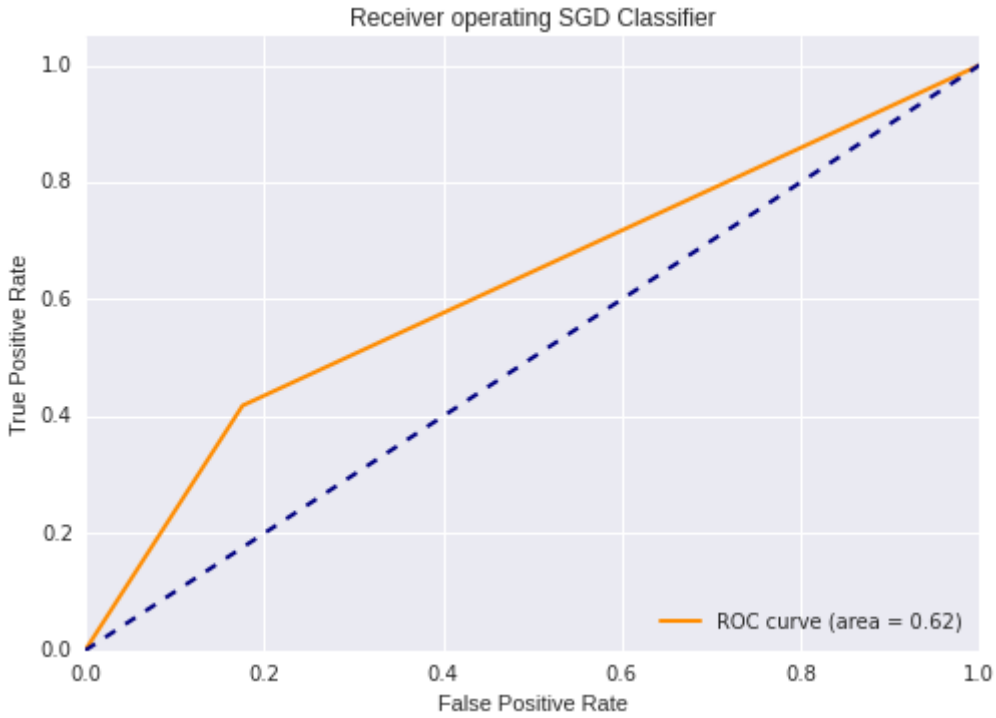


Logistic Regression

Random Forest



SGD Classfier



Comments on the Performance

For Logistic Regression the performance is close to Random. I decided not to go forward with it. SGD Classifier showed better performance than Logistic Regression. Random Forest, the performance is really good for an optimized model. I decide further to optimize different parameters to see the effect on AUC. Ensemble Models perform really well in classification problems with unbalanced data.

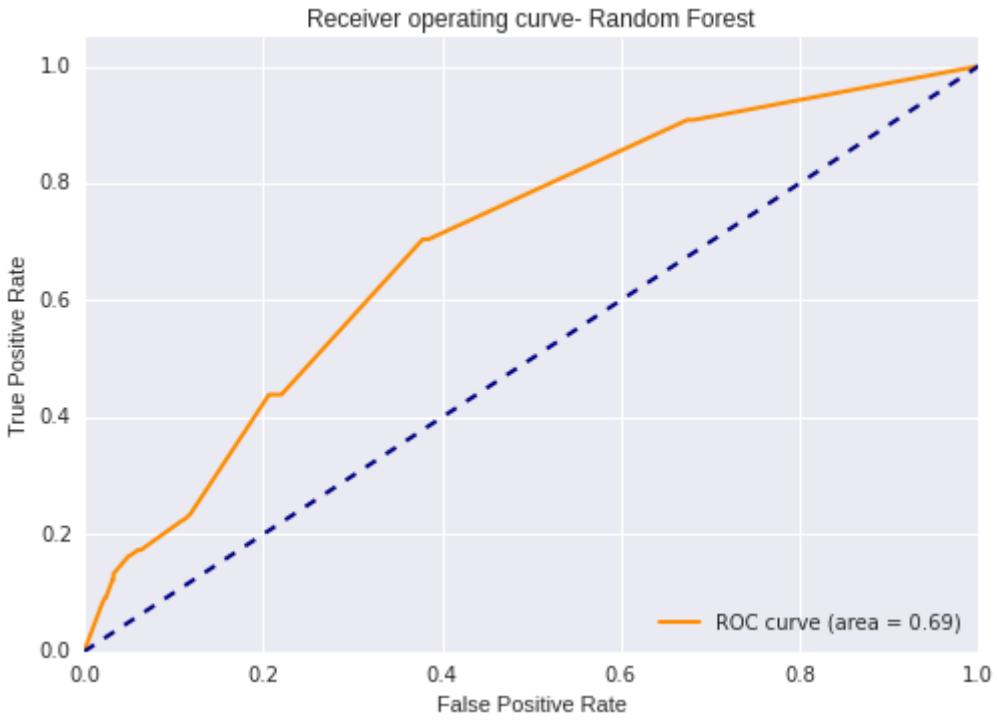
Improvement and Refinement

As mentioned before the data imbalance was a big problem. While doing classification the test split had a very small number of positive seizure cases. For example we initially had 6041 cases and after splitting, out of which 449 cases were positive. In other words the two classes were present in the ratio 7%: 93%.

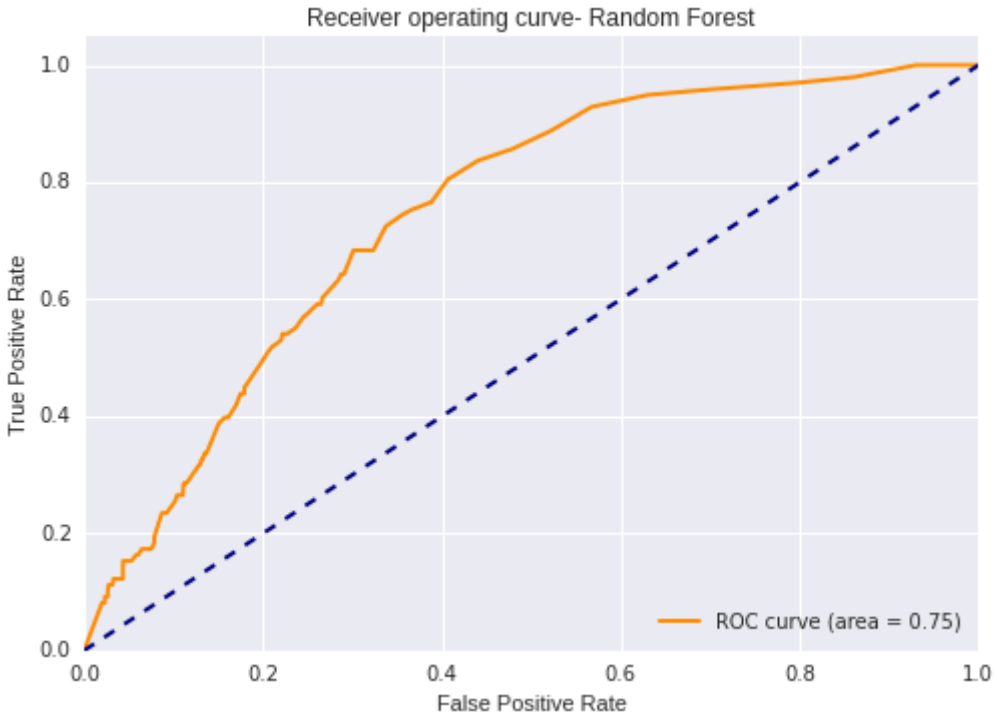
In the initial runs I got an AUC of 0.94 with SVM and 0.85 with KNN. This was much higher than I expected. This was due to a mistake that I made when I was oversampling. As I oversampled from the entire dataset there were copies of the same data points in the testing and training set during cross validation. Due to which there was data leakage aka overfitting.
As a corrective measure I left 1041 points for testing and 5000 points for oversampling and training. The test set was never exposed to any of the model while training.

Refining the model:
Gridsearch and Randomized search are ways to optimize the model. They provide ways to find the right combination of parameters that minimize the specified error metric.
Gridsearch performs exhaustive search over specified parameter values for an estimator.
Randomized search implements a randomized search over parameters, where each setting is sampled from a distribution over possible parameter values. Randomized search has a distinct advantage over gridsearch because the time for completion is much shorter.

Random Forest Randomized Search (n=20 estimators)



Random Forest Randomized Search (n=200 estimators)



Random Forest Grid Search (n=200 estimators)

Justification

The big question is "Can we using Machine Learning techniques to predict Seizures?" Here the question is better realized as can we distinguish between EEG signals before a seizure and normal signals.

Here is a summary of AUC results obtained.

MODEL	Area under the Curve
Logistic regression	0.53
SVM	0.54
Random Forest	0.62
Random Forest Randomized Search (n=20 estimators)	0.69
Random Forest Randomized Search (n=200 estimators)	0.75
Random Forest Grid Search(n=200 estimators)	0.74

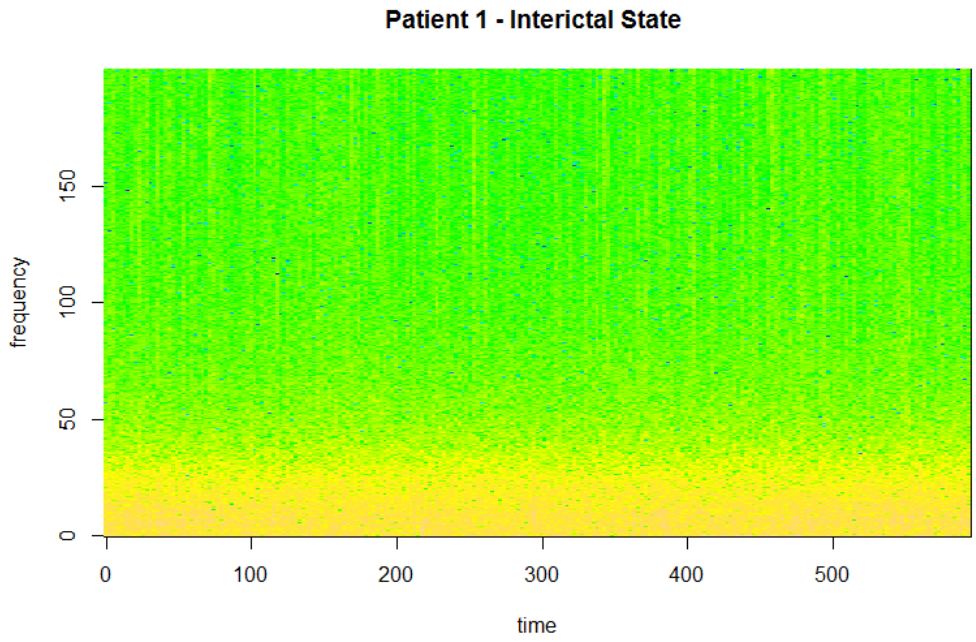
The AUC scores obtained are far above the benchmark set. From the given data, the approach I

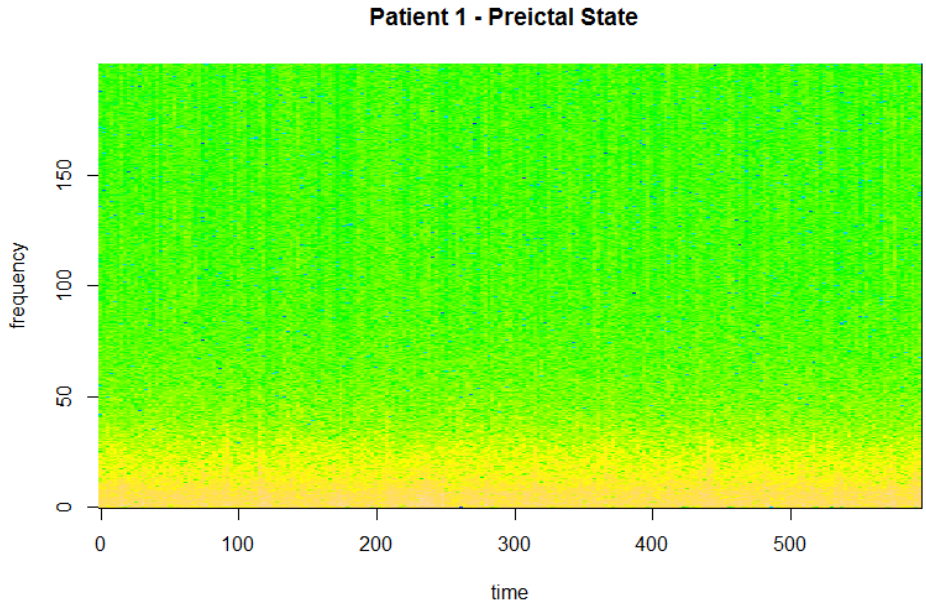
had taken has helped distinguish between preictal and Interictal data. The performance can be further improved. I talk about it more in the 'improvement' section.

Free-Form Visualization

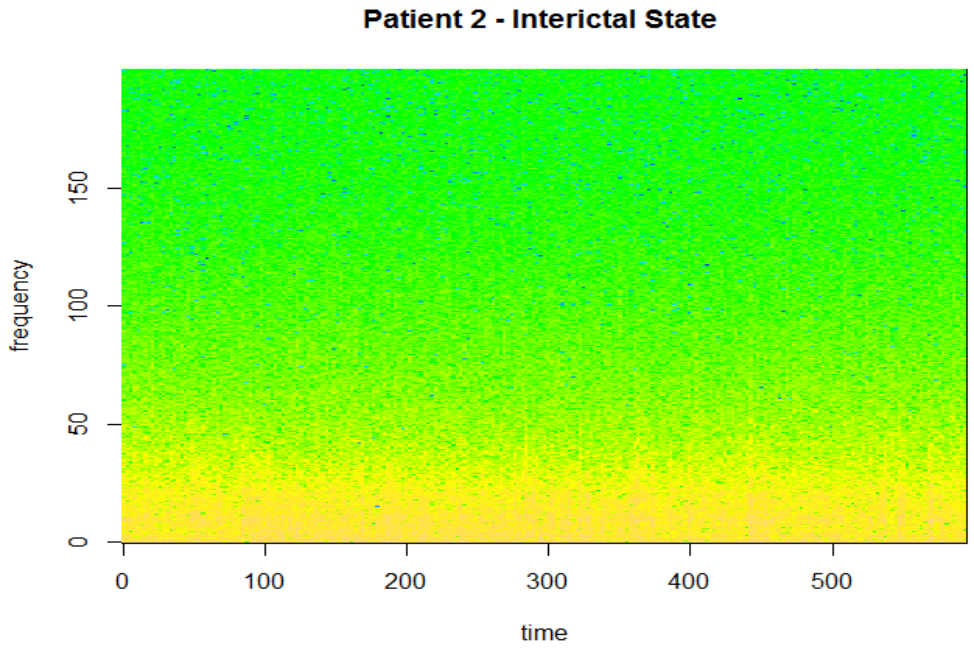
Spectrogram of patients

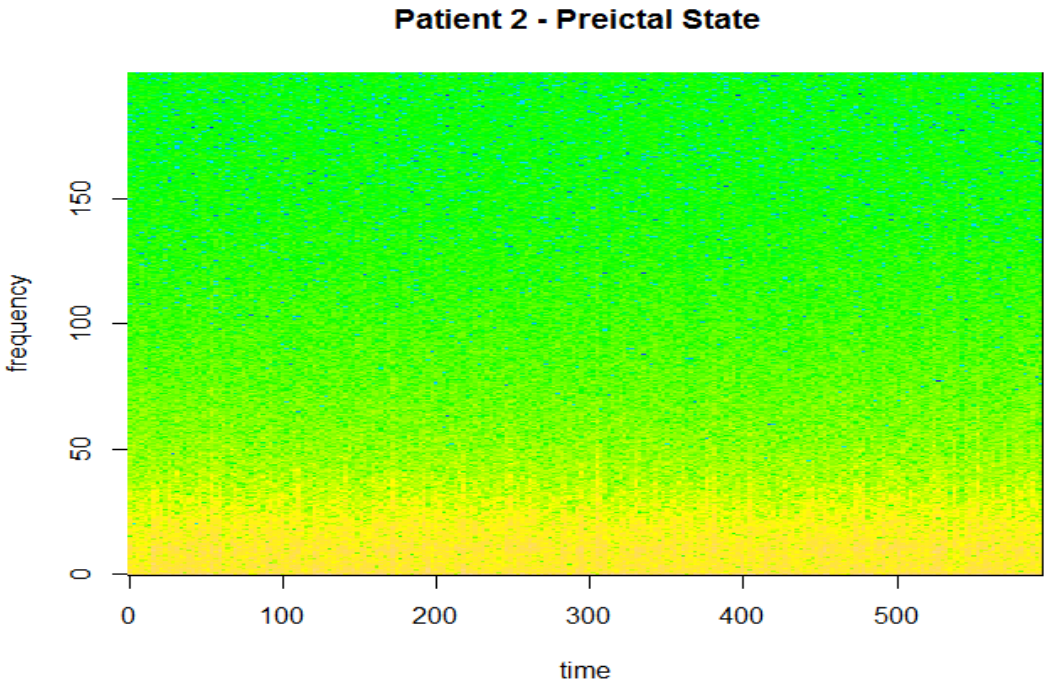
Patient 1



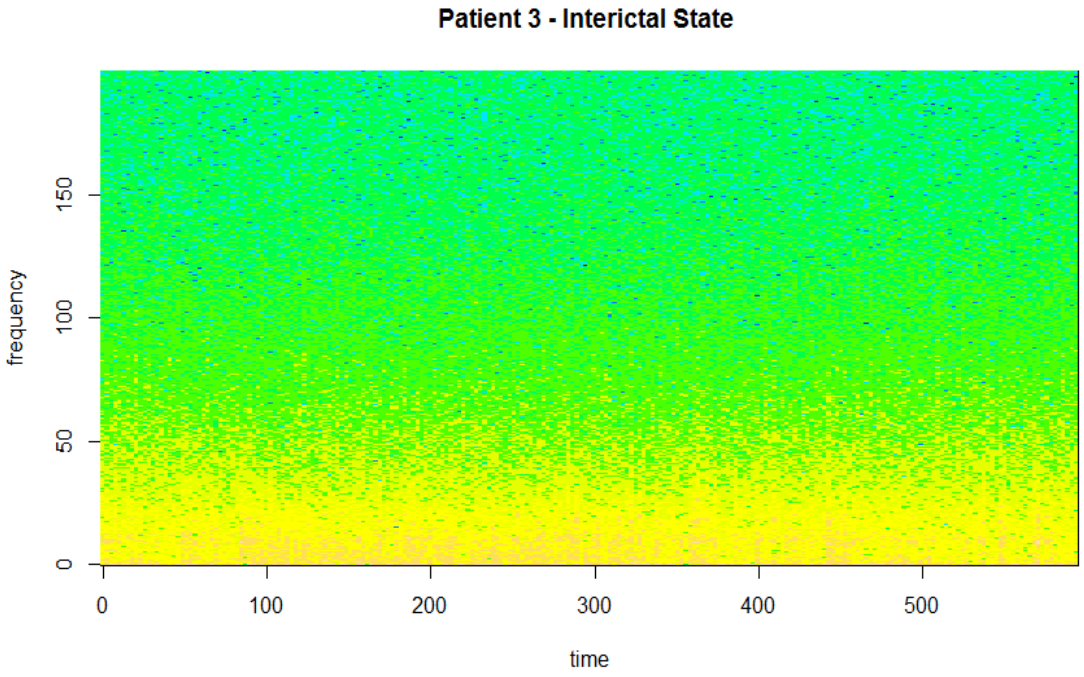


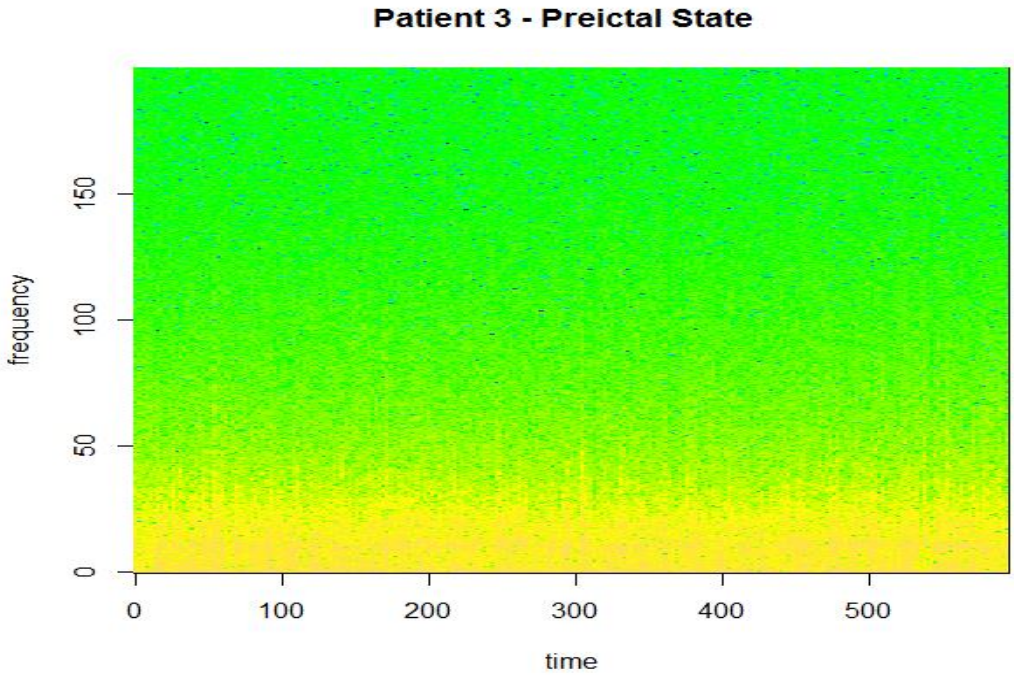
Patient 2


Patient 2 - Interictal State
frequency
0
50
100
150
0
100
200
300
400
500
time


Patient 2 - Preictal State
frequency
0
50
100
150
0
100
200
300
400
500
time

Patient 3


Patient 3 - Interictal State
frequency
0
50
100
150
0
100
200
300
400
500
time


Patient 3 - Preictal State
frequency
0
50
100
150
0
100
200
300
400
500
time

Spectrograms are a great way to visualize EEG data. Here I used Time-frequency approach or spectrogram image processing technique to analyze EEG signals using signal processing packages in R. In the spectrograms produced I did not see a significant difference between preictal and interictal spectrograms in the three patients.

Reflection

This was probably the hardest machine learning problem I have worked on. The size of the data, the nature of the problem and the use of signal processing techniques. Were big challenges. There were many times where I was getting visualizations, plots, spectrograms where I could not find any meaning or I could not see a difference between preictal and interictal state. I attempted different techniques and when the performance was not good I went back to the beginning and added a few more steps such as ICA and t-SNE and more visualization to obtain more insights . Some of the methods which I planned on using such as DBSCAN I avoided, because of lack of addition of value.

In retrospect I was able to learn a lot of new techniques related to handling signal data. The experience of working on a problem that has large dirty data from beginning to deriving meaningful results in a useful real-world application was thrilling.

Further Improvements- Next Steps, New Approaches

The AUC scores obtained were quite satisfactory and it is the metric of interest as specified for the Kaggle competition. I received some insights from the winner of a previous version of the same competition. I would like to incorporate some of these ideas and some others of my own to further improve performance.

1)Using median centering to calibrate probabilities for each model.

2)Using Ensemble techniques. Making a predictive model from weighted average of different models: (XGBoost, Random Forest, Bagged SVM etc).

3) Deep Learning approach such as LSTM.

Data Leakage*

Some of the competitors had a discovered a major issue with the data provided for the competition .1 hour interictal segments were selected that was not necessarily 4 hours away from seizures. This was a violation of a rule that the competition rules were specified. This could be due to a bug in the data collection/preparation.

*Link

Acknowledgments

This competition is sponsored by MathWorks, the National Institutes of Health (NINDS), the American Epilepsy Society and the University of Melbourne, and organized in partnership with the Alliance for Epilepsy Research, the University of Pennsylvania and the Mayo Clinic.

https://www.kaggle.com/c/melbourne-university-seizure-prediction/forums/t/23417/recent-papers-on-seisure-prediction-pdfs

REFERENCES

1. https://github.com/unpingco/Python-for-Signal-Processing
2. https://github.com/streety/kaggle-seizure-prediction-
3. http://small-yellow-duck.github.io/seizure_detection.html
4. https://www.kaggle.com/c/melbourne-university-seizure-prediction/forums/t/23417/recent-papers-on-seisure-prediction-pdfs
5. https://www.quora.com/What-are-the-advantages-of-different-classification-algorithms
6. http://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
7. https://en.wikipedia.org/wiki/Bootstrap_aggregating
8. https://en.wikipedia.org/wiki/Random_forest