GHENT UNIVERSITY

FACULTY OF SCIENCES

Massive multivariate models to infer the factors determining bias in connectivity between brain sources reconstructed from sensor measures

Toghrul Jafarov

Student ID: 01710091

Promotor: Prof. Daniele Marinazzo

Master dissertation submitted for obtaining the degree:

Master of Science in Statistical Data Analysis

Academic year 2017 - 2018

The author and the promoter give permission to consult this master dissertation and to copy it or parts of it for personal use. Each other use falls under the restrictions of the copyright, in particular concerning the obligation to mention explicitly the source when using results of this master dissertation.

Gent, June 10, 2018

The promotor,

Prof. Daniele Marinazzo

The author,

Toghrul Jafarov

Acknowledgments

First, I would like to thank Prof. Daniele Marinazzo for the time he spent working with me as well as his availability. It makes no doubt that the results of this thesis would not have been the same without his many advices. I am also grateful to him for allowing me to work on such an interesting topic.

I would also like to thank Paolo Presti and Frederik Van de Steen for their availability as well as for help to perform data simulation on the High Performance Computer of Ghent University.

Finally, I would like to address my gratitude to my friends and family for their support.

Contents

LIST OF TABLES

LIST OF ILLUSTRATIONS

Abstract

This master thesis is a simulation-based study of different factors determining bias in connectivity between brain sources reconstructed using electroencephalography (EEG) techniques. In particular, the goal of this document is to propose and analyze algorithms to carry out feature ranking with different algorithms i.e. giving a measure of usefulness for each input variable.

In addition to this, we will evaluate factors determining connectivity between brain sources reconstructed from sensor measures, based on false positive rate (FPR). The idea is to have a set of features which explain false connection between brain sources (via false positive rate).

This thesis attempt to explore comparative study of state-of-the-art feature selection methods. Recursive Feature Elimination algorithm will further be compared with Univariate Feature Selection (Chi-squared statistics) and Gradient Boosting Machine. Selecting the most representative features will provide a better understanding of the underlying process.

High Performance Computer of Ghent University has been used for massive data simulation.

This Master thesis is structured as follow. *Chapter 1* contains a brief overview of EEG and source localization problem. Afterwards, related work on this subject explained in detail: experiment design, performance metrics and different source localization and connectivity estimate explanation. We end up by presenting research question.

In *Chapter 2* we discuss theoretical background of machine learning and statistical methods that we care going to use: Binary classification, different types of Feature selection techniques.

In *Chapter 3* whole process of feature selection process is detailed from data generation, preprocessing, explorative data analysis and feature selection.

At the end, in *Chapter 4* we discuss our findings and propose future works related to this thesis.

Keywords: Machine learning, feature selection, EEG, Brain Connectivity, classification, Gradient Boosting Machine, Univariate feature selection, Chi-squared statistics, Recurrent feature elimination, EDA, Light GBM, LCMV, eLORETA, Granger Causality, Time Reversed Granger Causality, Partial Directed Coherence.

Chapter 1

I. Introduction

1.1 Background

Electroencephalogram (EEG) is a noninvasive way to record electrical activity originated in the brain. EEG is measured on the scalp, in order to infer information about location of the neural populations generating the recorded activity, and their connectivity. It's necessary to project the activity back into the brain where it originated from, using inverse models.

Unfortunately, this approach does not completely undo the mixing of activity originated in the original propagation of brain activity to the scalp, and this leads to a bias in measures of statistical dependencies between brain areas.

This project is about modelling interacting sources of activity at different locations in the brain and then performing a meta-model linking the precision of connectivity estimate to several factors, such as the level of noise, the depth of the sources, their relative position, etc.

In this thesis, we have investigated the possibility to use machine learning (ML) and statistical analysis to identify the factors which have the most impact on brain connectivity. The research covers the entire process: data simulation, data preprocessing, explorative data analysis and feature selection.

1.2 Related Work

A recent study performed in the promotor's lab (Alessandra Anzolin, 2018) has investigated the extent to which the residual mixing after source reconstruction influences estimates of directed functional connectivity. The goal of this work was applying meta-modelling on the factors determining bias in connectivity between brain sources reconstructed from sensor measures. Signals from close regions in the brain are highly mixed, which makes difficult to identify the true interactions.

Simulated data was performed based on the following factors: depths of the two sources, distance of the sources from center of brain, signal to noise ratio, length of the signal. Simulation study was performed starting from the generation of a simulated dataset resenting a brain signal

whose connectivity pattern is imposed. The signals were then projected on the scalp (sensor level) solving the forward problem and using New York Head model for the geometry and conductibility of the head. The inverse problem was then solved using two different algorithms: the linearly constrained minimum variance (LSMV) and the exact low-resolution brain electromagnetic tomography (eLORETA). Causal connectivity estimation was performed using three different algorithms on the signals reconstructed at source space level: Granger Causality (GC), Time Reversed Granger Causality (TRGC) and Partial Directed Coherence (PDC).

Here we wanted to consider all the main factors which can introduce a bias in connectivity, performing a massive number of simulations, and use a classifier to rank their effects when it comes to predict a wrong estimate.

1.2.1 Experiment design

In order to simulate EEG signal, MVAR model of order two was used. Precisely, three-time series were simulated and only one connection between two of them. In order to simulate the brain activity in a more realistic way, 500 times series were generated representing the background and noisy cerebral activity. These time series should be mutually statistic independent (Alessandra Anzolin, 2018).

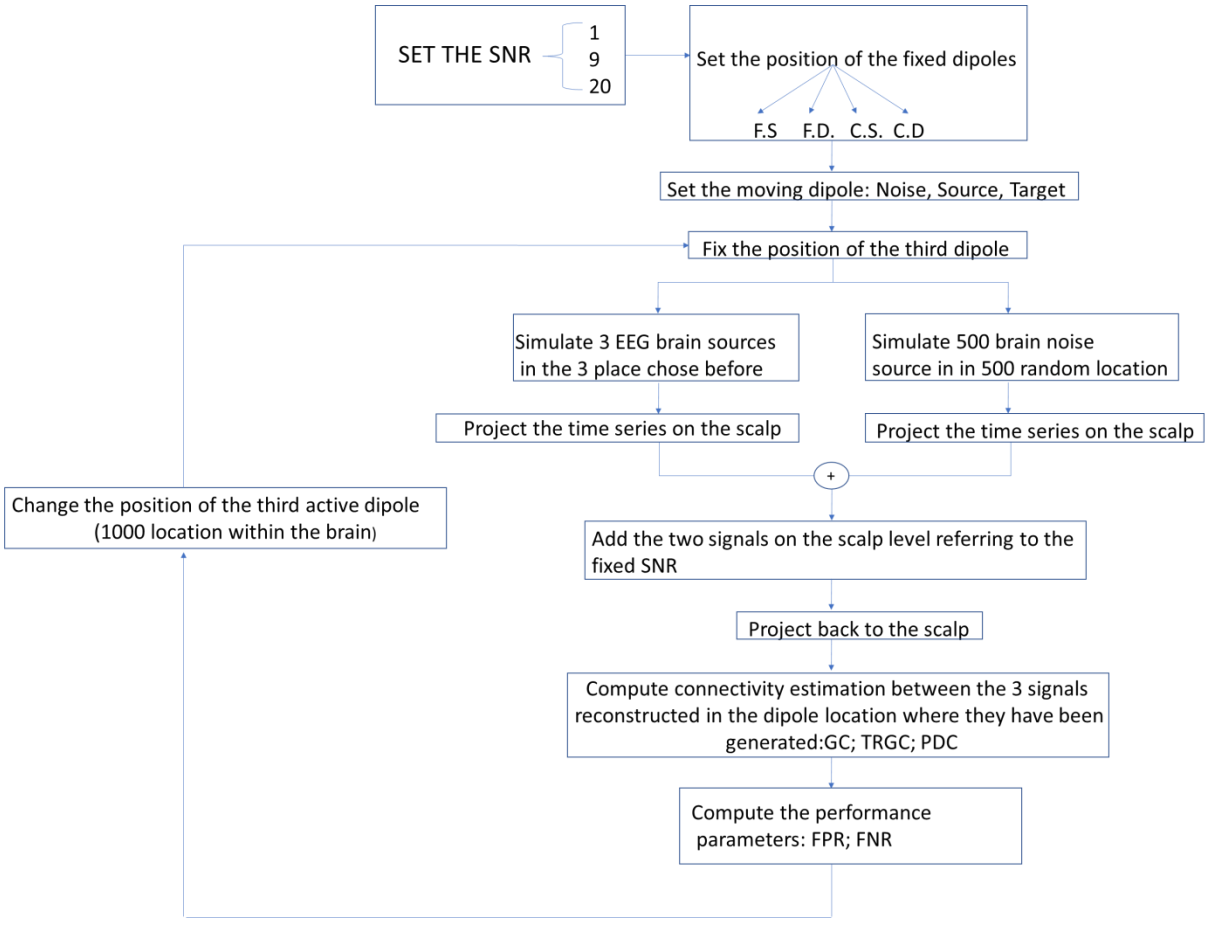

Figure 1 Block diagram reporting the main steps of the simulation framework

1.2.2 Simulated time series location

Our approach consisted in fixing the position of two active dipoles and moving the third one over 1000 locations equally distributed over the whole brain.

The 5000 additional noisy elements where randomly distributed within the brain. In Figure 2 all the possible dipole positions are represented. Brain activity was modelled with 1006 electric equivalent dipoles, equally distributed within the brain. New York Head model was used, so that we were able to find the dipole coordinates by subsampling the 75,000 MNI coordinates available in the ICBM152 model (Alessandra Anzolin, 2018).

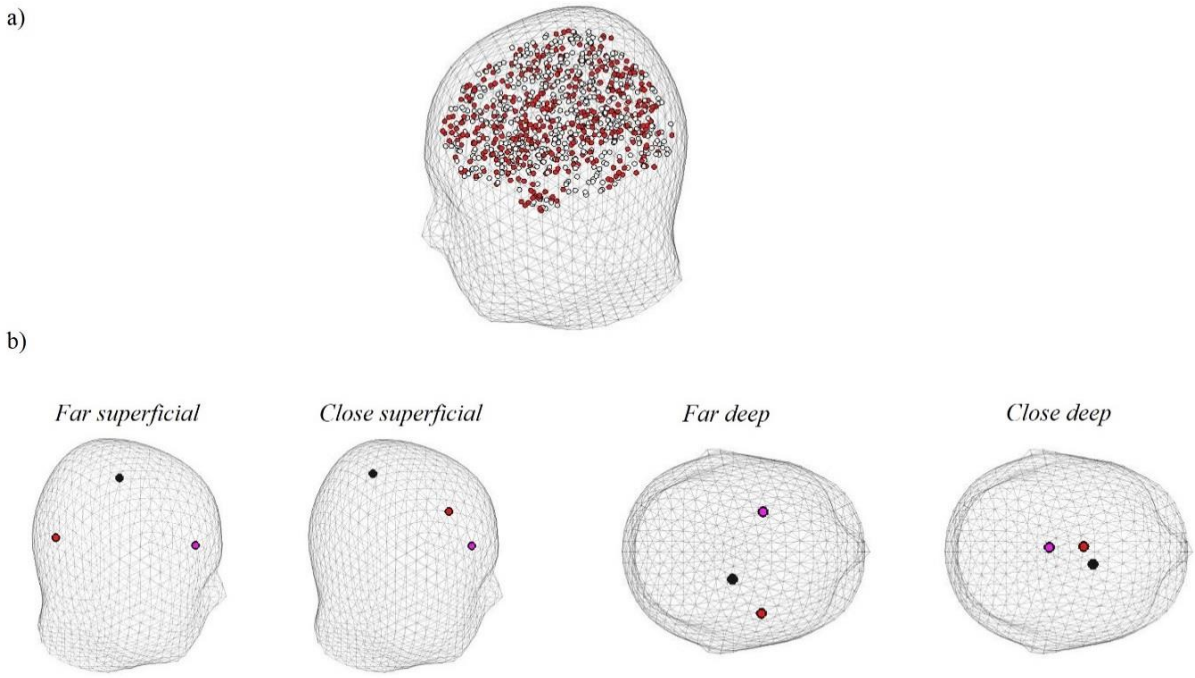

Figure 2 All the possible dipole positions in the brain

In Figure 2, panel a) shows the 1006 locations with which was modelled the brain activity. Red circle represents an example of the 500 locations associated with the brain noise activity. Panel b) presents the four conditions for the two fixed active dipoles, which are the red (source) and the purple (target) one. The black circle represents the non-interactive dipole (noise).

For each distance and depth level, other three different cases by analyzed changing the considered fixed dipoles. In the first case the fixed dipoles were the source and the target, in the second case the source and the noise and in the last case, the target and the noise. Thus, in this framework, 12 different conditions were considered: 4 distance and depth cases and 3 different type of moving dipoles.

1.2.3 Forward problem, inverse problem and connectivity estimates

After the signals generation, the time series representing the source activity and the noise were projected on the sensors space and summed according to the selected SNR value. Spatially and temporally uncorrelated vectors mimicking measurement noise are sampled from a univariate standard normal distribution (Alessandra Anzolin, 2018). Thus, the overall pseudo-EEG data were obtained according to the following weighted sum:

$$x(t) = 0.9\frac{x^{brain}(t)}{||x^{brain}(t)||_F} + 0.1\frac{x^{noise}(t)}{||x^{noise}(t)||_F} \quad (1.1)$$

Where $x(t)$ is the resulting EEG signal, x^{brain} is the signal representing the brain activity (comprehensive of activated MVAR dipoles and of the 500 noise signal, properly added with respect of the SNR), and x^{noise} stands for the measurement noise. The obtained simulated signal on the scalp was then projected to the source space according to two different inverse problem solutions:

- LCMV
- eLORETA

After that, in order to quantify the effect of the volume conduction and for different connectivity estimation algorithms we employed:

- Granger Causality (GC);
- Time Reversed Granger Causality (TR_GC);
- Partial Directed Coherence (PDC).

1.2.4 Performance parameters

To have a quantitative evaluation of the spurious connections and false detections two different performance parameters were defined. The false positive rate (*FPR*) tell us about the amount of spurious connections obtained when connectivity estimation finds out connections while actually the time series investigated were no-interactive. The false negative rate (*FNR*) is a measure of the amount of loss connections obtained when the imposed connectivity pattern provides a connection between the time series which was no detected by the estimator.

Both FPR and FNR were computed comparing the imposed connectivity pattern with the adjacency matrix provided by the estimator. They are respectively defined as:

$$FPR = \frac{FP}{FP + TN} \quad (1.2)$$

$$FNR = \frac{FN}{FN + TP} \tag{1.3}$$

Where the FP and the FN are respectively the spurious connections and the loss connections provided by the estimators while TP and TN are the right estimations, this the estimator detects or not a connection between two series when it is present or not.

All simulations were iterated five million times in order to improve the statistics. This was made possible thanks to the Flemish Supercomputer Center (VSC) which is virtual center making supercomputer infrastructure available for both the academic and industrial world. This center is managed by the Research Foundation – Flanders (FWO) in partnership with five Flemish university associations (Alessandra Anzolin, 2018).

1.3 Research Question

The research question of this work can be summarized as:

"*Comparative study of state-of-the-art feature selection methods in order to rank the feature importance for brain connectivity*"

Chapter 2

II. Theory

1.4 Machine Learning

Machine learning is a field of computer science for solving problems related to the data, and divided into two tasks: *supervised* and *unsupervised* learning. In order to understand feature selection, we should understand what supervised learning is? Supervised learning can be defined as a machine learning task that infers a function from labeled data. To this end, a model is learned on training data which consists of inputs-outputs pairs. The goal is to learn a mapping function from inputs to outputs, such as to minimize the error while mapping unseen examples (i.e. that don't belong to the training data). The optimal scenario would be for the model to map new inputs to the exact outputs. This implies that the learning algorithm must generalize from the training data in order to handle unseen inputs correctly. Supervised learning has plenty of applications and is often used for computer vision, speech recognition, time-series predictions and others (Lundborg, 2017).

1.4.1 Linear Classification

Given a set of data points, each belonging to one of two classes, the goal is to decide to which class any given new data points belong. A linear classifier makes the decision based on the value of a linear combination of the feature vectors. The linear classifiers can be seen as a function $y = f(w, x)$ which maps a set of inputs *x* to an output *y*, through a set of weights *w*. The algorithm tries to find a hyperplane (in the 2-dimensional case, a line) which maximizes the separation between the two categories (Lundborg, 2017).

1.4.2 Feature Selection

In the last decade, the area of feature selection has received a great amount of attention by machine learning researchers (Shanab, 2011). Feature selection aims at finding the best subset of features from the entire number of features that can represent the input data efficiently and can still provide good prediction results (Chandrashekar, 2014); (Moustakidis, 2012). Feature selection uses a search algorithm to find one or more informative subsets of features according to predefined criteria.

Feature selection should be able to ding the important or relevant features (i.e. the features that are relevant to the given prediction task) (Yang et al., 2010). If a problem has N features, the number of all subsets of features is equal to 2^N. Therefore, the optimal set of features is one (or could be more) of an exponential number of possible subsets, and comparing all of these subsets to find the best is intractable for $N > 20$. Feature selection determines the feature relevance according to an evaluation criterion associated with the given method (Hamed, 2017).

In general, feature selection methods can be divided into three types: Filter methods, Wrapper methods and Embedded methods (Sathya, 2011).

(a) Filter methods involve the methods that perform feature selection before building the classifier and do not incorporate learning.
(b) Wrapper methods incorporate a machine learning in measuring the quality of the subsets of features without incorporating knowledge about the specific structure of the classification or regression function.
(c) Embedded methods are different from Filter and Wrapper methods, in that with embedded methods, the learning part and the feature selection part cannot be separated.

1.4.2.1 Filter methods

Filter methods include the methods where feature selection is independent of the classifier to be applied to the selected features. These methods determine the feature importance by inspecting the intrinsic properties of the data. Commonly, filter methods calculate a feature relevance score for all the features and remove low-scoring features. The main advantages of filter methods are their ability to deal with high-dimensional datasets (i.e. not affected by the curse of dimensionality). In addition, they are independent of the classification algorithm, faster than other feature selection methods as they are not computationally intensive. Filter methods ignore the impact of the selected subset of features on the performance of the induction algorithm (Kohavi R. &., 1997).

1.4.2.2 Wrapper methods

Wrapper methods incorporate a machine learning in measuring the quality of the subsets of features without incorporating knowledge about the specific structure of the classification or regression function. Wrapper methods involve wrapping the feature selection around the classifier construction (Draminski, 2008). However, wrapper methods are more computationally intensive than filter methods since wrapper methods require constructing a new predictor for every candidate feature subset. A common shortcoming of the wrapper methods is that they are prone to overfitting and very computationally intensive.

1.4.2.3 Embedded methods

Embedded methods are different from Filter and Wrapper methods, in that with embedded methods, the learning part and the feature selection part cannot be separated (Guyon, 2003). Embedded methods perform the feature selection in the process of learning which saves the time required for two-step induction as in the wrapper methods. The search for the best subset of features is built into the classifier construction. Embedded methods have more efficiency over wrappers in better utilizing the available data without the to split the training into training and validations sets (i.e. efficient use of the available data). Thus, embedded methods are more of a "white box" methods, since the feature selection is based directly on the classifier. In addition, they find a solution faster since there is no need to retrain a predictor from scratch for every examined subset of variables (Guyon, 2003).

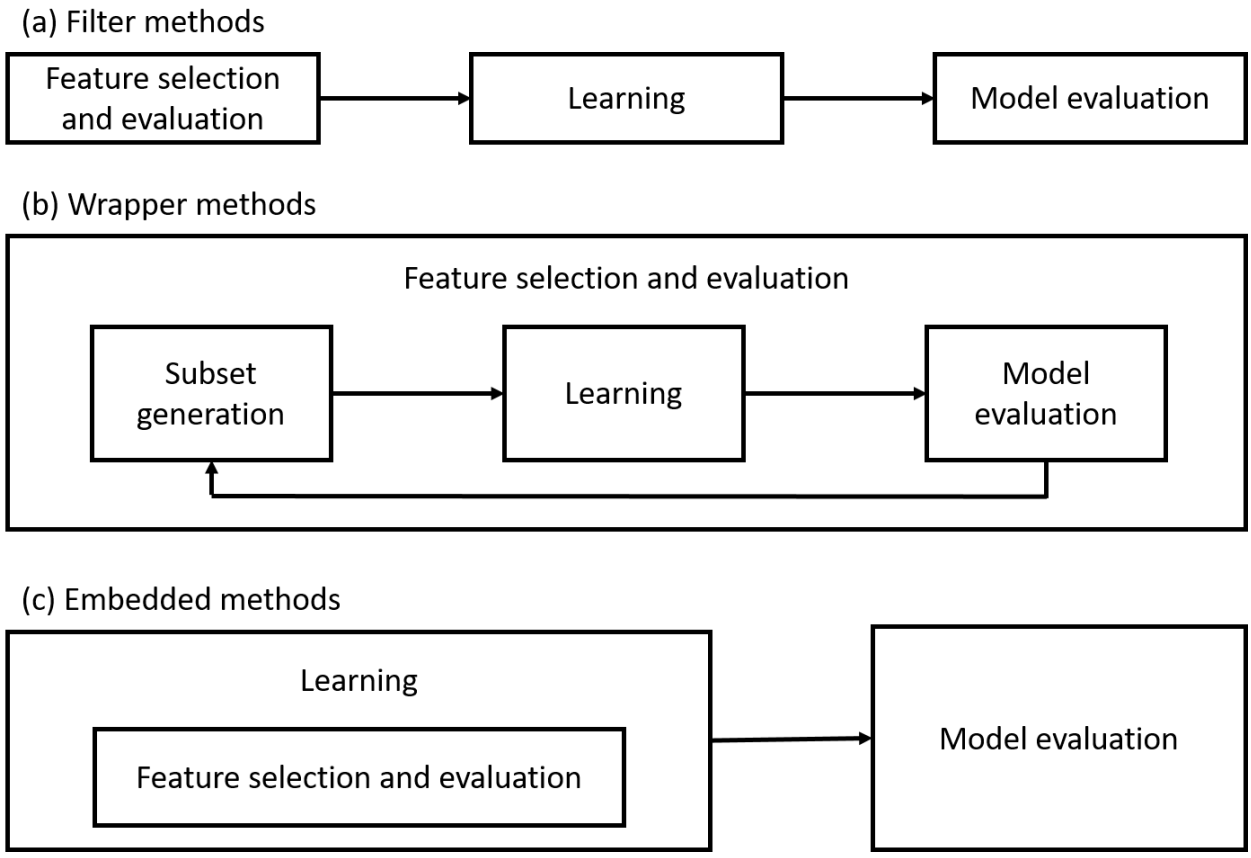

Figure 3 Filter, wrapper and embedded feature selection methods (adapted from Hilario et al.; 2008)

1.5 Algorithms

1.5.1 Chi-squared statistic (χ^2)

Chi-Squared $\boldsymbol{\chi^2}$ is a filter method that evaluates features individually by measuring their chi-squared statistic with respect to the classes. In statistics, the χ^2 test is applied to test the independence of two events, where two events A and B are defined to be independent if $P(AB) = P(A)P(B)$ or, equivalently, $P(A|B) = P(A)$ and $P(B|A) = P(B)$. In feature selection, the two events are occurrence of the term and occurrence of the class. We then rank terms with respect to the following quantity:

$$\chi_c^2 = \sum_{i=0}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{2.1}$$

where, subscript c is the degree of freedom, O is observed value and E is expected value.

χ^2 is a measure of how much expected counts E and observed counts N deviate from each other. A high value of χ^2 indicates that the hypothesis of independence, which implies that expected and observed counts are similar, is incorrect. For example, $\chi^2 \approx 231 > 10.83$ and based on Table 3, we can reject the hypothesis that two features are independent with only a 0.001 change of being wrong. Equivalently, we say that the outcome $\chi^2 \approx 231 > 10.83$ is statistically significant at the 0.001 level (Cambridge University Press, 2009).

If the two events are dependent, then the occurrence of the term makes the occurrence of the class more likely (or less likely), so it should be helpful as a feature. This is the rationale of χ^2 feature selection.

p	χ^2 critical value
0.1	2.71
0.05	3.84
0.01	6.63
0.005	7.88
0.001	10.83

Table 1 Critical values of the chi-squared distribution with one degree of freedom

1.5.2 Recursive Feature Elimination

Recursive Feature Elimination (*RFE*) is to select features by recursively considering smaller and smaller sets of features based on an external estimator that assigns weights to features (*i.e.* the coefficients of linear model). First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coefficient attribute or through feature importance attribute. Then, the least important features are pruned from current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached (Pedregosa, 2011).

1.5.3 Gradient Boosting Machine (Light GBM)

Gradient boosting is a powerful machine learning technique introduced by Friedman (2001). Most recently, a new tree bossing method has come to stage and quickly gained popularity. XGBoost due to popularity on data science competition platform Kaggle. Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of weak prediction models, typically decision trees. Boosting can be interpreted as an optimization algorithm on a suitable cost function. Like other boosting methods, gradient boosting combines weak learners into a single strong learner, in an iterative fashion (Tianqi Chen, 2016).

Light GBM is new version of XGBoost released by Microsoft in 2017, which is faster version of the previous gradient boosting algorithm.

1.5.4 Random Forests

Random forests are some of the easiest models to use and work "out-of-the-box" for a wide range of problems. Therefore, they are often chosen for feature selection tasks due to their ease of use as well as their performances. They exist multiple algorithms to compute feature importance using random forests ([Qi, 2012]):

- **Permutation based variable importance**. This algorithm consists in permuting the variables at test time and looking at the accuracy loss. This technique is part of the wrapper algorithms. It can thus be used with any learning algorithm.
- **Revised Random Forest feature importance**. This algorithm reflects the quality of the node splits. It is in fact similar to Gini importance, but the difference is that the depth importance takes into account the position of the node in the trees. This formula has been proposed by [Chen et al., 2007].
- **Gini importance**. This method is directly based on the Gini index which measures the level of impurity / inequality of the samples assigned to a node based on a split at its parent. For instance, under the binary classification case, let p represent the proportion of class 1 samples assigned to a certain node n and $1 - p$ as the proportion of class 2 samples. The Gini index at n is defined as:

$$G_n = 2p(1 - p) \tag{2.2}$$

The Gini importance value of a feature in a single tree is then defined as the sum of the Gini index reduction (from parent to children) over all nodes in which the specific feature is used to split. The overall importance in the forest is defined as the sum or the average of its importance value among all trees in the forest.

1.6 Error Estimation Method: Cross-validation

Cross-validation is a statistical method of evaluating and comparing learning algorithms by repeatedly partitioning the given data set into two disjoint subsets: the training and the test subset. The training subset is used to build the classifier, then the samples belong to the test subset is used to test the trained classifier. The process is repeated with several partitions and gives an estimate of the classification performance. The most common form of cross-validation is k-fold cross-validation (Kohavi R. , 1995).

- **K-fold cross-validation**: The *k*-fold cross-validation partitions the given data set into k equally sized subsets. Then, training is done on $k-1$ subsets and testing is done on the remaining subset. This process is repeated k times (folds) with each subset is taken to be o test set in turn (Abusamra, 2013).
- **Leave-one-out cross-validation**: In this method we use k-fold cross-validation where k is equal to the number of samples in the data set. In each "fold", $n-1$ samples are used as training set and a single sample is used for testing. This procedure is repeated fir all samples. This method is computationally expensive as it requires the construction of n different classifiers. However, it is more suitable for smaller datasets (Abusamra, 2013).

Chapter 3

III. Methods

1.7 Data simulation

Massive data set was simulated based on the framework of the previous work. High Performance Computer of Ghent University was used for simulation. Framework to simulate brain signals and calculate different source localization and connectivity estimate was written in Matlab programming language.

Obtained simulated signals on the scalp was then projected to the source space according to two different inverse problem solutions:

- LCMV
- eLORETA

After that, in order to quantify the effect of the volume conduction and for different connectivity estimation algorithms we employed:

- Granger Causality (GC)
- Time Reversed Granger Causality (TR_GC)

We have the following 7 factors / predictors which have been used for simulation:

Feature name	Feature description
Snr	Signal to noise ratio
Len	Length of the simulated signal
Distance source	Distance between the two sources
Depth1	Distance of the first source from the middle of the brain
Depth2	Distance of the second source from the middle of the brain
Localization source	Source localization algorithms (0 – LCMV, 1 – e-LORETA)

Connectivity estimate	Connectivity estimate algorithms (0 – GC, 1 – TR_GC)

Table 2 List of factors / features used for data simulation

We ended up with 20 million of observations, i.e. 5 million simulations (for each observation randomly generating value for each factors) times 2 source localization and 2 connectivity estimates algorithms. Total volume of the data set was 2Gb. One of the challenges of this thesis was to handle big data set. We were obliged to use algorithms which could handle big data.

1.8 Data Preprocessing

Python has been chosen as a programming language for this thesis, and it required to convert simulated data from Matlab file into Python data frame in order to perform further analysis. The structure of the Matlab file was like Table 3.

Snr	Len	Distance source	Depth 1	Depth 2	Localization source (LCMV)	Localization source (eLORETA)	Connectivity estimate (GC)	Connectivity estimate (TR_GC)
0.65	1179	89.452	93.91	46.15	0	0.5	0	0.5
0.72	1448	111.06	67.46	53.96	1	1	1	1

Table 3 Example of simulated data set on Matlab softwares

On the right part of the table we have False Positive Rate corresponding localization source and connectivity estimate. These values are target value for our project. Next, we have pivoted right side of the table by adding two predictors (localization source and connectivity estimate) with target variable. In order to differentiate simulations from each other unique identifier has been added for each row. In this way we can easily partition data set for training and test data sets.

ID	Snr	Len	Distance source	Depth 1	Depth 2	Localization source	Connectivity estimate	Target
1	0.65	1179	89.452	93.91	46.15	0	0	0
1	0.65	1179	89.452	93.91	46.15	0	1	0.5
1	0.65	1179	89.452	93.91	46.15	1	0	0
1	0.65	1179	89.452	93.91	46.15	1	1	0.5
2	0.72	1448	111.06	67.46	53.96	0	0	1

2	0.72	1448	111.06	67.46	53.96	0	1	1
2	0.72	1448	111.06	67.46	53.96	1	0	1
2	0.72	1448	111.06	67.46	53.96	1	1	1

Table 4 Preprocessed simulated data in the format of Python Data Frame

1.9 Explorative Data Analysis (EDA)

Before modeling or hypothesis testing, let's summarize main characteristics of data set. Data set contains two binary categorical variables (localization source and connectivity estimate), and five continuous variables. Target variable (number of false positive rate) is a continuous variable with only three different values (0.0, 0.5, and 1.0). Instead of analyzing regression problem we have decided to convert continuous values into categorical and continue as a classification problem.

The distribution of the continuous variables is differing with each other. Distance source, depth of the sources are normally distributed whereas, length of the signal and signal to noise ratio are uniformly distributed.

Data set doesn't contain any missing values. Data set doesn't contain any outliers.

1.10 Pairwise correlation

First, we start base-line feature selection with analyzing relation / correlation between features. In order to analyze the correlation between the features we will look at correlation matrix. Pairwise correlation of features is calculated by using Pearson correlation coefficient. We can observe no correlation between any features expect distance between sources and depths (correlation coefficient 0.22). This gives some intuition about by knowing depth of one source and distance between sources we can find out the depth of the second source. But this correlation is very weak.

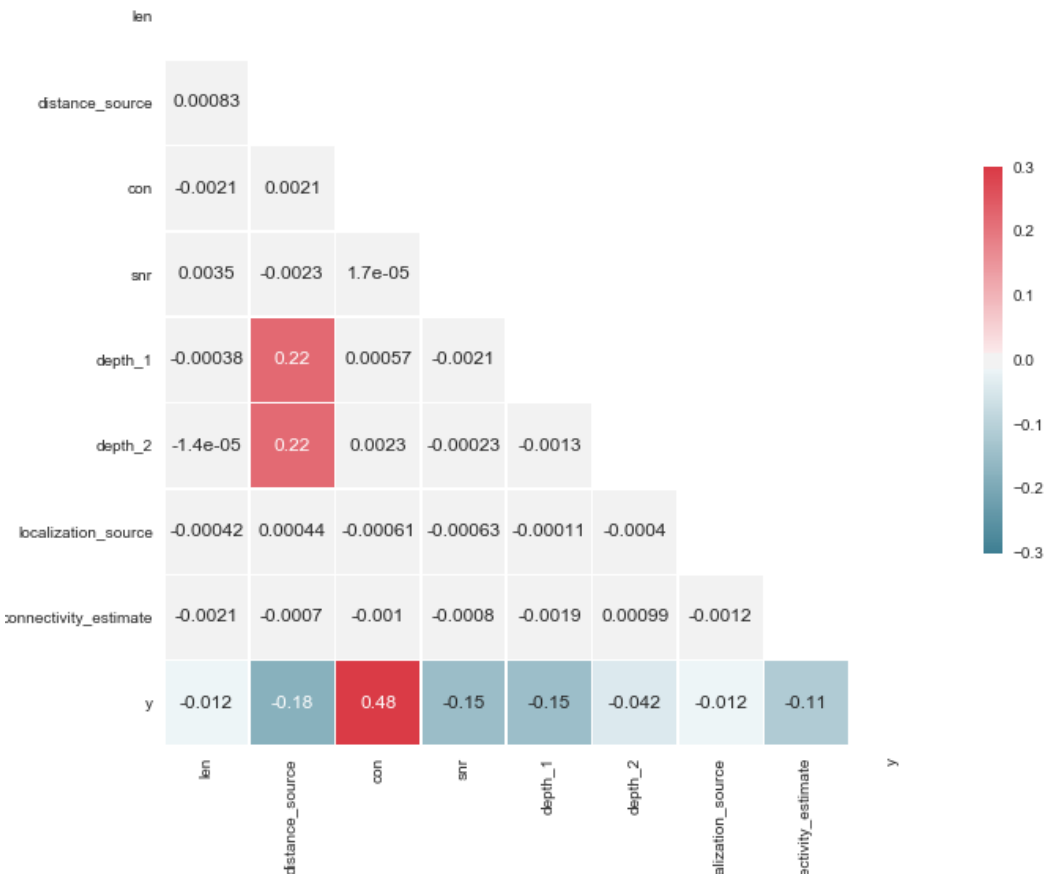

Figure 4 Pairwise Pearson correlation matrix of features

1.11 Target variable

Target variable has three classes with imbalanced distribution. From Figure 6 we can observe that class 0.5 has very low presence compare to others.

Class	Frequency (%)
0.0	55.2
0.5	7.5
1.0	37.2

Table 5 Class frequency (%) of target variable

Target variable, number of False Positive Rate (FPR), had three values, with imbalanced distribution. In order to solve this problem two minor classes (0.5 and 1.0) were combined into one. We transform from multiclass classification into binary classification problem, which makes easy on the further analysis.

Chapter 4

IV. Results

1.12 Univariate feature selection

Chi-squared univariate feature selection has performed by using SelectKBest function from scikit-learn library. Feature importance (top 5) based on chi-squared statistics gives the following results with corresponding score (chi-squared test):

Feature name	Chi-squared statistics
Length of the signal	745,281
Depth of first source	482,544
Signal to noise ratio	453,158
Distance between sources	441,112
Depth of second source	49,390

Table 6 Feature importance based on chi-squared statisticss

1.13 Gradient Boosting Machine

Before calculating feature importance based on Gradient Boosting Machine, first we have tuned hyper parameters. Hyper parameter tuning is technique to find out the best values for model, by evaluating each possible parameter combination with cross-validation.

Following parameters of Light GBM has been tuned by performing 3-fold cross-validation and using ROC AUC metrics. Number of trees was fixed to 50. In order to overpass the computation issues, 25% of the data set was used for parameter tuning. 25% of the data was randomly sampled without replacement.

Light GBM parameter	Possible values to tune
Num_leaves	20, 30

Lerarning_rate	0.01, 0.1
Max_depth	4, 5, 6

Table 7 Hyper parameter tuning for Light GBM model with list of values

By using optimal model parameters (learning rate = 0.1, max depth = 6, number of leaves = 30) Light GBM was performed on data set with 200 number of estimators and logistic loss (or cross-entropy loss) as evaluation metric.

Feature importance based on Gradient Boosting Machine model is as following table.

Feature name	Chi-squared statistics
Depth of first source	1801
Distance between sources	933
Localization source	721
Length of signal	663
Depth of second source	652

Table 8 Feature importance based on Gradient Boosting Machine

1.14 Recursive Feature Elimination (RFE)

As an estimator Light GBM was used with tuned parameter values. Top 3 features were asked to find out by eliminating one feature on each iteration.

Feature name
Distance between sources
Signal to noise ratio
Depth of first source

Table 9 Feature importance based on Recursive Feature Elimination (RFE)

Chapter 4

V. Discussion

1.15 Conclusion

When starting this work, the goal was to find factors explaining the brain connectivity by using machine learning techniques, concretely by exploring different feature selection techniques.

We have completed a feature selection process including data simulation and data exploration. Converting multi-class classification into binary helped to simplify the interpretation and get better results. Considering the large volume of the data set, less computationally (feature selection) methods were chosen, like: Chi-squared statistics, Recursive Feature Elimination, and Gradient Boosting Machine.

By using majority voting technique, we can conclude that *depth of the first source*, *distance between sources*, and *signal to noise ratio* are the most important features.

1.16 Future work

The most limiting factor is by far the volume of the data set, which require computation power in order to perform different feature selection techniques and conduct classification based on the selected features from each technique. In this point, we can compare performance of classification models to better evaluate features.

VI. Bibliography

Abusamra, H. (2013). A Comparative Study of Feature Selection and Classification. *King Abdullah University of Science and Technology*, 20-25.

Alessandra Anzolin, P. P. (2018). Effect of head volume conduction on directed connectivity estimated between reconstructed EEG sources. *Biorxiv*, 1-29.

Cambridge University Press. (2009). *Chi2 Feature selection*. Retrieved from stanford.edu/: https://nlp.stanford.edu/IR-book/html/htmledition/feature-selectionchi2-feature-selection-1.html

Chandrashekar, G. &. (2014). A survey on feature selection methods. *Computers and Electrical Engineering*, (pp. 16-28).

Draminski, M. R.-I. (2008). Monte carlo feature selection for supervised classification. *Bioinformatics*, (pp. 110-117).

Guyon, I. &. (2003). An introduction to variable and feature selection. *The*, (pp. 1157-1182).

Hamed, T. (2017). Recursive Feature Addition: a Novel Feature Selection Technique, Including a Proof of Concept in Network Security. *Guelph, Ontario*, 5-9.

Kohavi, R. &. (1997). Wrappers for feature subset selection. *Artificial intelligence*, 273-324.

Kohavi, R. (1995). A Study of cross-validation and bootstrap for accuracy estimation and model selection. *In Proceedings of the 14th International Joint Conference on Artificial*, (pp. 1137-1143).

Lundborg, A. (2017). Text classifcation of short messages. *LUND UNIVERSITY*, 11-18.

Moustakidis, S. &. (2012). A fast svm-based wrapper feature selection method driven by a fuzzy complementary criterion. *Pattern Analysis and Applications*, (pp. 379-397).

Pedregosa, F. V. (2011). Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research*, 2825-2830.

Sathya, D. R. (2011). Discriminant analysis based feature selection in kdd intrusion dataset. *International Journal of Computer Applications*, 1-7.

Shanab, A. K. (2011). Impact of noise and data sampling. *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*, (pp. 172-177).

Tianqi Chen, C. G. (2016). *XGBoost: A Scalable Tree Boosting System.* University of Washington.