Machine learning classification of schizophrenia from EEG brain networks

Master's Thesis
(M4R Project)

by
Matthew T. Fredericks
CID: 01070697

Department of Mathematics
Faculty of Natural Sciences
Imperial College London

Supervisors:
Prof. Henrik J. Jensen
Hardik Rajpal

June 11, 2019

Declaration of Authorship

The work contained in this thesis is my own work unless otherwise stated.

Signed:

Matthew Fredericks

Acknowledgements

I'd first like to thank my supervisor, Henrik Jensen, for making this project possible. His expertise and enthusiasm made the research of the past year enjoyable from start to finish.

I'd also like to thank Hardik Rajpal, my secondary supervisor and resident expert on causal analysis of brain data. I am especially grateful for his technical assistance, which included teaching me to use the computing cluster system from scratch.

Lastly, Stefan Brugger, for his part in providing us with the datasets needed to get the project off the ground.

Abstract

Starting from EEG time series data of over 300 million data points across 16 patients, we aim to investigate how accurately we can classify patients as healthy or schizophrenic. Rather than attempting to data mine the raw time series directly, we first construct weighted directed networks using information-theoretic causality measures. Then using various network metrics as predictors, we employ machine learning algorithms including LASSO regression, random forests, and neural networks, to classify schizophrenia. By using two different measures to construct network models of the brain, we find that classification algorithms perform better on networks with high edge density than ones with more restrictive edge selection. We obtain accuracies of up to 93%. Lastly, we investigate which network metrics are found by the classification algorithms to be most important in characterising the schizophrenic brain.

Contents

1. Introduction

Network theory has seen extensive usage in neuroscience; from studies on the completely mapped neuronal connectome of the nematode worm *C. elegans* [1], to cross-brain causality analysis between a trio of improvising musicians [2, 3]. A network is comprised of a set of nodes, interconnected by edges (or links). For networks that model the brain, the nodes represent brain regions, and edges represent a form of either physical connection or functional association between electrical signal readings. After appropriately defining a node and edge set, a number of topological network properties become available for analysis.

One relevant area in neuroscience is the ongoing research into schizophrenia. About 0.5-1% of the population worldwide suffer from schizophrenia [4], and its often debilitating symptoms include auditory hallucinations and paranoid or bizarre delusions [5]. At a neuronal level, this is generally attributed to what is known as the 'dysconnection hypothesis' [6] which refers to abnormal relationships between neurons, compatible with (but not necessarily implying) anatomical disconnection [7]. Network-based studies support this hypothesis, having found that brains of schizophrenia patients are less efficiently wired, with abnormally clustered network hubs [8]. Many other studies investigating schizophrenia with network theory have been conducted, many of which can be found in the references at the end of this report. A good overview and starting point is given by [9].

Our study will follow in these footsteps, but employ some novel techniques inspired from information theory and machine learning.

The remainder of this chapter will outline the details, goals and motivation behind our study. Chapter 2 will introduce all relevant mathematical theory. In chapter 3 we describe and explain choices in the methodology for each step of the EEG study. The reader will then be guided through the results of our analysis in chapter 4, where we also discuss neurological interpretations. Chapter 5 is the conclusion, in which we summarise our findings and also discuss weaknesses and further study. Example code can be found in the appendices.

1.1. In this study

We use a dataset of EEG (electroencephalogram) time series data measuring brain activity recorded from 9 healthy control patients and 7 schizophrenia patients. Using a natural node set of EEG sensors, we will define weighted links between nodes using two different information-based causality measures (PMIME [10] and PCMCI [11]). By examining the time series from each node, the causality measures should not only quantify the strength of causal link between each pair of nodes, but the direction of causation. We can then obtain multiple (depending on time intervals) weighted directed networks for each patient. Having constructed our brain networks, we wish to answer questions about differences between control patients and schizophrenia patients. To this end, we will train a number of machine learning algorithms to classify schizophrenia networks based on topological metrics. The process is summarised in figure 1.1.

The aims of this study:

- Construct weighted directed brain networks using modern causality measures.
- Find the network metrics that are most indicative of a schizophrenic brain, and check agreement with well-established results.
- Be successful in classifying schizophrenia in patients unseen by our algorithms.

Similar studies have incorporated information-based causality measures in brain networks [2,12], or used machine learning algorithms to classify brain networks (alcoholism in [13], autism in [14], ex-combatants in [15]). However, to our knowledge, information theory, networks, and machine learning have not been combined before on any brain data.

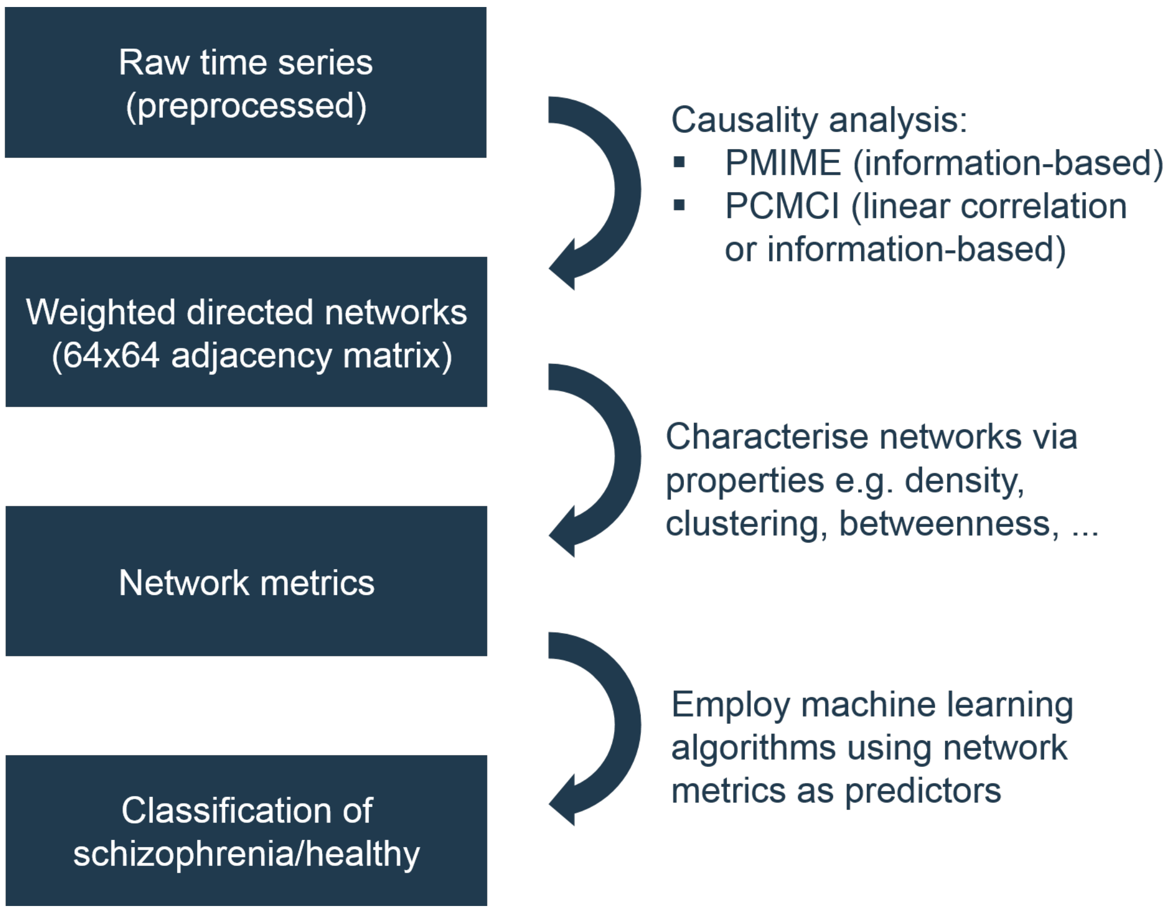
Raw time series (preprocessed)
Causality analysis:
PMIME (information-based)
PCMCI (linear correlation or information-based)
Weighted directed networks (64x64 adjacency matrix)
Characterise networks via properties e.g. density, clustering, betweenness, ...
Network metrics
Employ machine learning algorithms using network metrics as predictors
Classification of schizophrenia/healthy

Figure 1.1: summary of the working pipeline for this project.

1.2. Significance

Directed networks allow arguably more realistic modelling of the underlying mechanism behind node communication in the brain. But directed network analysis in neuroscience (for example [16]) is less prevalent than one might expect, possibly due to the lack of an established and reliable causality measure. In this study we will apply PMIME and PCMCI (which have been shown to work in numerous settings including neuroscience [17], financial stocks [18] and climate interactions [11]) to construct our network, hoping to achieve the same success.

The causality measures we will use have some clear advantages in that they are nonlinear (can detect nonlinear dependencies), nonparametric (so unrestricted by model assumptions), quantify link strengths, and detect directionality. The importance of weighted (as opposed to binary) network analysis has been stressed, due to the utility of weak edges in characterising alterations such as those caused by schizophrenia [19]. The alternative is to binarise the network via thresholding - this eliminates noise but disregards potential weak edges. Arbitrary choice of threshold would greatly affect the derived network, so it is necessary to systematically explore the range of threshold values (good practice of this is in [8]).

The common problem in comparative brain studies similar to ours is the small sample size of patients. This lowers the dataset's capacity to make generalising inferences. For schizophrenia, in which there is high between-subject variability in candidate traits [20], this is especially problematic - we could very easily model patient idiosyncrasies instead of general features of schizophrenia. In machine learning terminology this is called overfitting. By applying machine learning techniques such as cross-validation, which are designed to highlight overfitting, we can at least make informed inferences. It is important to realise that these techniques are useful in assessing the extent of overfitting, not for erasing the problem from the data.

Usually analysis of network metrics to differentiate two groups is done by statistical hypothesis tests (e.g. pairwise t-tests), which aim to reject the hypothesis that the two groups are indistinguishable. But the machine learning techniques we will use should not only quantify the differences more tangibly, but potentially be useful for classifying undiagnosed patients in future.

Clearly, machine learning techniques (known also as data mining techniques, after the general field) come with many advantages over common methods in complex network science. But interestingly, the overlap of these two disciplines is in the interest of both sides: for example, network science community detection outperforms the data mining equivalent (clustering) [13]. It has also been shown that direct analysis of adjacency matrices using data mining techniques does not compare to analysis of topological metrics in the derived networks [13]. The reason there is not more integration of the two fields is seemingly due to contingent rather than conceptual reasons [13].

2. Mathematical background

Our EEG study will begin with raw *time series* data corresponding to each EEG sensor for each participant. The sensors provide a natural set of nodes for use in a *network-theoretic* analysis. However the method of defining interconnections between nodes is less obvious; we seek an intelligent *causality measure* that is able to capture directionality of brain signals between two sources. For this we delve into *information theory.* Having constructed our brain networks, we will train *machine learning algorithms* to classify schizophrenia based on network metrics.

In this chapter we discuss the relevant theory in a general context; choices made in the context of our EEG study will be discussed and justified in chapter 3.

2.1. Time series

In this section we briefly introduce time series processes - a broad class of stochastic processes that are observed with time. Throughout we will cite definitions from [21].

Definition 1. A **time series** process $\{X_t\}_{t \in T}$ is a family of real-valued random variables indexed by t. The index set is denoted by T.

Usually, t denotes time and takes discrete values sampled at equal intervals, while X_t (for fixed t) is a continuous random variable.

Definition 2. A time series process $\{X_t\}$ is **strictly stationary** if $\forall n \geq 1$, $\forall t_1, \dots, t_n \in T$, and $\forall \tau : t_1 + \tau, \dots, t_n + \tau \in T$, we have $F_{X_{t_1}, \dots, X_{t_n}} = F_{X_{t_1+\tau}, \dots, X_{t_n+\tau}}$, where F represents a joint cumulative density function.

This essentially says that the probability structure of a strictly stationary process is invariant to time shifts.

Definition 3. A time series process $\{X_t\}$ is **weakly stationary** if $\forall n \geq 1$, $\forall t_1, \dots, t_n \in T$, and $\forall \tau : t_1 + \tau, \dots, t_n + \tau \in T$, the joint moments of orders 1 and 2 of $\{X_{t_1}, \dots, X_{t_n}\}$ are finite and equal to the corresponding joint moments of $\{X_{t_1+\tau}, \dots, X_{t_n+\tau}\}$.

Following directly from this definition:

$$\begin{aligned} E[X_t] &= \mu, \quad \forall t \in T \\ Cov[X_t, X_{t+\tau}] &= E[X_t X_{t+\tau}] - \mu^2 \\ &= E[X_0 X_\tau] - \mu^2 \text{ (by applying a time shift of } -t) \\ &= E[X_{-\tau} X_0] - \mu^2 \text{ (by applying a time shift of } -t-\tau) \end{aligned}$$

i.e. the covariance of X_t and $X_{t+\tau}$ is a function only of the absolute time lag $|\tau|$. For this reason, weak stationarity is sometimes also referred to as covariance stationarity. From this point onward, assume stationarity refers to weak/covariance stationarity unless explicitly stated otherwise.

We give two relevant examples of discrete time series processes (useful for simulation purposes):

Definition 4. A **white noise process** is a sequence of uncorrelated random variables, usually denoted $\{\epsilon_t\}$, such that $E[\epsilon_t] = \mu$ and $Var[\epsilon_t] = \sigma^2 \ \forall t$.

Definition 5. $\{X_t\}$ is a ***p*-th order autoregressive process** (AR(p)) if it has the following form:

$$X_t = \phi_{1,p} X_{t-1} + \phi_{2,p} X_{t-2} + \cdots + \phi_{p,p} X_{t-p} + \epsilon_t,$$

where ϵ_t is a zero-mean white noise process.

Note that the stationarity of an AR(p) process depends on its parameters ϕ. [21]

2.2. Information theory

Information theory was first introduced in 1948 by C. E. Shannon, in the context of signal transmission [22]. It provides a set of useful measures with far-reaching applications in mathematics, computing, physics, electrical engineering, and economics [23]. Throughout the section we will loosely follow the structure and definitions used in [24].

2.2.1. Information for single events

Let x denote a stochastic event, belonging to an event set X, with associated probability $p(x)$.

Definition 6. [22] For an event $x \in X$, the **Shannon information** of x is denoted $I(x) = -log(p(x))$.

The intuition is that the occurrence of an event with probability 1 provides precisely 0 information, while lower probability events are more 'informative'. It also provides a measure of uncertainty of an event, hence we will use the words 'information' and 'uncertainty' interchangeably. The units of information are dependent on the base of logarithm used; in base 2 we use 'bits', in base e we use 'nats', and in base 10 we use 'hats'.

Now consider a second event y, belonging to an event set Y. Let $p(x, y)$ denote the joint probability of x and y i.e. the probability of the joint event (x, y) belonging to the two-dimensional joint event set $X \times Y$.

Definition 7. For a joint event $(x, y) \in X \times Y$, the **joint information** of (x, y) is denoted $I(x, y) = -log(p(x, y))$.

We can interpret joint information as the uncertainty of the *simultaneous* occurrence of events. In the same manner we can define an information measure based on conditional probability to describe the uncertainty of one event conditional on another.

Definition 8. For events $x \in X$ and $y \in Y$, the **conditional information** of x given y is denoted $I(x|y) = -log(p(x|y))$.

Conditional information tells us the uncertainty of an event given another event, but doesn't explicitly give us a measure of the shared uncertainty. Thus we introduce a related measure:

Definition 9. For events $x \in X$ and $y \in Y$, the **mutual information** between x and y is denoted $I(x;y) = log(\frac{p(x|y)}{p(x)}) = I(x) - I(x|y)$.

Therefore mutual information describes the amount of uncertainty that one event can remove from the other. Note mutual information between single events can be positive or negative. By applying the identity for conditional probability, it is easy to show that mutual information is a symmetric measure:

$$I(x;y) = log\Big(\frac{p(x|y)}{p(x)}\Big) = log\Big(\frac{p(x,y)}{p(x)p(y)}\Big) = log\Big(\frac{p(y|x)}{p(y)}\Big) = I(y;x)$$

From this it is also easy to see that if x and y are independent events, then $p(x,y) = p(x)p(y)$ which implies $I(x;y) = log(1) = 0$; there is no shared information. Conversely, $I(x;y) = 0$ must imply independence. Therefore $I(x;y) = 0$ and independence of x and y are logically equivalent statements.

2.2.2. Information for random variables

Now let X denote a discrete random variable with outcomes $\{x_i\} = \mathbb{X}$ and probability mass function p which assigns each outcome with a probability $p(x_i)$. We introduce the concept of entropy as a measure of the expected uncertainty contained within an entire event set.

Definition 10. For a discrete random variable X with domain $\mathbb{X}$, the **information entropy** of X is denoted by:

$$H(X) = -\sum_{x_i \in \mathbb{X}} p(x_i) log(p(x_i)) = E[I(x_i)],$$

where E is expectation.

Therefore information entropy is interpretable as an average uncertainty over all outcomes. Note that entropy depends only on the outcome probabilities and not the outcomes themselves, making it a useful measure for variables with non-numerical outcomes. Also note that entropy increases with number of outcomes (but zero-probability events don't contribute as $0 \cdot log(0) \equiv 0$).

Define a second discrete random variable Y with domain $\mathbb{Y}$. Let p be the joint probability mass function such that $p(x_i, y_j)$ is the probability of the joint event $(X = x_i \cap Y = y_j) \in \mathbb{X} \times \mathbb{Y}$.

Definition 11. For discrete random variables X and Y with domains $\mathbb{X}$ and $\mathbb{Y}$, the **joint entropy** of X and Y is denoted by:

$$H(X,Y) = -\sum_{\mathbb{X}\times\mathbb{Y}} p(x_i,y_j)log(p(x_i,y_j)) = E[I(x_i,y_j)]$$

Joint entropy is interpretable as the average amount of shared information between two random variables.

Definition 12. For discrete random variables X and Y with domains $\mathbb{X}$ and $\mathbb{Y}$, the **conditional entropy** of X on Y is denoted by:

$$H(X|Y) = -\sum_{\mathbb{X}\times\mathbb{Y}} p(x_i,y_j)log(p(x_i|y_j)) = E[I(x_i|y_j)]$$

Conditional entropy describes the average amount of remaining uncertainty in one variable, having acknowledged the information provided by the other.

Definition 13. For discrete random variables X and Y with domains $\mathbb{X}$ and $\mathbb{Y}$, the **mutual information (rate)** [24] between X and Y is denoted by:

$$I(X;Y) = \sum_{\mathbb{X}\times\mathbb{Y}} p(x_i,y_j)log\Big(\frac{p(x_i|y_j)}{p(x_i)}\Big) = E[I(x_i;y_j)]$$

Note: this is usually known in the literature as just "mutual information", but it is not to be confused with the mutual information between single events.

Mutual information rate can be interpreted as the average amount of information one variable provides about another. Like mutual information, mutual information rate is also a symmetric measure. However a key difference is that mutual information rate is non-negative.

All definitions covered in this subsection may be generalised for continuous random variables by appropriately replacing summations with integrals.

2.3. Causality measures

Mutual information is useful for quantifying the extent to which two random variables X and Y are related. However it cannot tell us whether X causes Y or vice versa, because it is a symmetric measure. It is known as a zero-lag association [25], and cannot detect directionality.

The question of causality is further complicated by the many possibilities for false positive detection. For example, there could be a common driver of X and Y (that is not necessarily a variable present in the dataset), or the X and Y could be linked in the real world but only indirectly. There is also likely to be some autocorrelation within individual variables that we must account for. The ideal causality measure will control false positive rate by means of some known significance level (typically 5%) while maximising the 'true' detection power [11].

2.3.1. Granger causality

We start with the original concept devised by Granger in 1969. [26]

Definition 14. [26] Consider two stationary time series processes, $\{X_t\}$ and $\{Y_t\}$. Let $P_t(X|Y)$ be the optimal unbiased least-squares predictor of X_t using the values of $\{Y_t\}$. Then define the predictive error series as $\{\epsilon_t(X|Y)\} = \{X_t - P_t(X|Y)\}$, and let $\sigma^2(X|Y)$ be the variance of $\{\epsilon_t(X|Y)\}$. $\hat{Y}$ denotes the set of all past values of $\{Y_t\}$; $\hat{U}$ denotes the set of all past information 'in the universe'. Then we say that Y **"Granger-causes"** X if $\sigma^2(X|\hat{U}) < \sigma^2(X|\hat{U}\backslash\hat{Y})$.

In other words, Y Granger-causes X if we can predict the present value of X more accurately with the inclusion of the historical values of Y.

While Granger causality is still applicable in the field of neuroscience [14], it requires us to impose a linear (autoregressive time series model) framework, which will result in the loss of any nonlinear information [12]. Secondly, knowledge of $\hat{U}$ is an unrealistic assumption to make, which becomes a problem for high-dimensional datasets due to the reduction in detection power [11]. Using LASSO regression [27] in place of standard regression will tackle this particular issue, by essentially putting a penalty on number of dimensions before optimising the objective function. Even so, such methods still suffer from reliability regarding false positives, especially in the nonlinear case [11]. Thirdly, we would like a measure for use on multivariate time series; Granger causality is limited mostly to bivariate time series analysis [11] (which restricts the analysis to direct causality).

2.3.2. PMIME

We look to instead use a nonlinear, multivariate causality measure with tried and tested results on neuro data. Two possible candidates include direct transfer entropy (DTE) [24] (a hybrid of partial directed coherence and transfer entropy), and partial mutual information from mixed embedding (PMIME) [10]. Of these, PMIME was found to be the most useful causality measure via experimentation (analytical models, cross-brain EEG analysis of reader-listener, musician-audience) [24], so we will focus on just this measure.

Definition 15. [10,24] Consider a multivariate stationary process composed from K time series $\{x_t, y_t, z_{1,t}, \ldots, z_{K-2,t}\}_{t=1}^{n}$, representing variables $X, Y, Z_1, \ldots, Z_K$. Define a future vector, $x^T = (x_{n+1}, \ldots, x_{n+T})$ with time horizon T. Also define historical vector $W = (x_n, \ldots, x_{n-L_1}, y_n, \ldots, y_{n-L_2}, z_{1,n}, \ldots, z_{1,n-L_3}, \cdots, z_{K-2,n}, \ldots, z_{K-2,n-L_K})$ which contains the past values up to lag L_i for the ith time series ($i = 1, \ldots, K$).

Now initialise an empty embedding vector, $w_0 = \emptyset$. Then set $s = 1$ and initiate the following iterative scheme for w_s:

1. Use a k-nearest neighbours estimate of mutual information to obtain the mutual information rate between x^T and $W\backslash w_{s-1}$, conditional on w_{s-1}.
2. $w_s = \left(w_{s-1}, \underset{*\in W\backslash w_{s-1}}{argmax}\left\{I(x^T;*|w_{s-1})\right\}\right)$ (append the latter value to vector).

3. STOP iterative scheme if $\frac{I(x^T;w_{s-1})}{I(x^T;w_s)} > A$. Else set $s = s + 1$ and GO TO 1.

If w is the final embedding vector, then denote the components of x_t in w as w^x, the components of y_t in w as w^y, and the components of $z_{1,t}, \dots, z_{K-2,t}$ in w as w^z. Then to quantify the causal effect of Y on X conditioned on other variables, we define the **partial mutual information from mixed embedding** as

$$PMIME_{Y \to X|Z} = \frac{I(x^T; w^y|w^x, w^z)}{I(x^T; w)} \in [0, 1]$$

In simpler terms, $PMIME_{Y \to X|Z}$ iteratively selects the lagged components out of all variables which are most informative about the future of X, and outputs the mutual information between future X and selected Y components, normalised by the mutual information between future X and all selected components. The outstanding advantage of PMIME is that it does not need to consider causality for every possible combination of lagged components.

Notes about parameter choices [10]:

- When all variables are of the same type (e.g. EEG signals) it is natural to assume the same maximum lag for all variables, $L_i = L \; \forall i = 1, \dots, K$. Choice of L is not critical; larger is better but at the cost of longer computation.
- The time horizon T depends on the problem, but $T = 1$ is widely used.
- Choice of threshold A controls the sensitivity to which we want to detect causalities. If A is too close to 1 however, we risk detecting false positive causalities, as well as longer computation time. The default is $A = 0.97$ having performed best in simulations.
- The kNN estimation of mutual information is formulated in [28]. $k = 5$ is used in [24].

The benefit of an information-based approach is that we are not imposing any model on the data, which would be assuming prior knowledge that we don't have. It is therefore able to detect nonlinear dependencies. Another clear advantage of PMIME is that the strength of causality is quantified, enabling us to construct weighted networks (the benefits of which will be discussed in chapter 3; networks are introduced in section 2.4).

PMIME also has tried and tested results on EEG data [17] and financial stock index data [18]. It is stressed in [24] that there is no best measure for all applications. Thus we will consider one more measure, which is also information-based and shares the same benefits as PMIME.

2.3.3. PCMCI

PCMCI [11, 25] is a two-stage procedure involving a condition-selection step to remove irrelevant variables, followed by extra conditional independence testing to address false positives. PCMCI allows discovery of causal links as well as the time lag associated with this link. The procedure is highly adaptable, as the precise method for each step is up to the user - choices are detailed in [11]. Linear (ParCorr) and nonlinear (conditional mutual information, CMI [29]) independence tests are possible. Below we outline the general procedure with CMI-based test, and use italics to indicate these choice-specific methods.

Definition 16. Consider a multivariate stationary process composed from N time series variables $X_1, \ldots, X_N$. **PCMCI** *with CMI independence test* consists of two stages:

1. For each of the variables $X_{j,t}$ perform **PC$_1$ algorithm** (based on PC algorithm [30]) to estimate its causal 'parents' amongst all variables X_i $(i = 1, \ldots, N)$ and lags $\tau = 1, \ldots, \tau_{max}$. Denote this set of $N\tau_{max}$ candidate parents as P_j. We iteratively remove candidates $X_{i,t-\tau}$ from P_j that are deemed conditionally independent from $X_{j,t}$ in an α-level test. *This test is based on the kNN estimate of* $I(X_{i,t-\tau}; X_{j,t}|P_j \backslash X_{i,t-\tau})$. We must assume the causal Markov condition, meaning that (when conditioning) the past is irrelevant once we know the direct 'parents'.

2. For $j = 1, \ldots, N$ let P_j correspond to the parents selected for $X_{j,t}$ in the step above. For ALL $N^2\tau_{max}$ variable pairs $(X_{i,t-\tau}, X_{j,t})$ run an α-level **MCI (momentary conditional independence) test**. This tests the same hypothesis as the step above, but no longer conditioned on the p_X strongest parents of $X_{i,t}$ (appropriately shifted by τ). Denote these as $\{P_i(\tau)\}$. *This test is based on the kNN estimate of* $I(X_{i,t-\tau}; X_{j,t}|P_j \backslash X_{i,t-\tau}, \{P_i(\tau)\})$.

The extra criterion in the second test addresses false positive control for the highly-interdependent time series case (particularly false positives from autocorrelation). PCMCI avoids conditioning on irrelevant variables giving it more detection power. It has much the same benefits as PMIME does, being a weighted multivariate measure (with link strength given by the final test statistic values) and also being nonlinear and nonparametric (if using information-based testing [29]).

Notes about parameter choices [11]:

- Maximum lag τ_{max} should be taken as large as necessary to cover all significant lagged dependencies. Graphs can be plotted to pick τ_{max} by eye. Large choice only affects computation time, not estimation.
- Significance level α can be optimised, but a recommended value when using CMI-based test is $\alpha = 0.2$. Note that we shouldn't interpret this α in the same way we are used to for more precise hypothesis testing.
- For choice of p_x (the number of strongest parents of $X_{i,t}$ to exclude from conditioning on), usually $p_x = 1 \sim 3$ suffices to reduce false positive rate due to autocorrelation.
- The kNN estimation of mutual information is again (as in PMIME) the formulation described in [28].

PCMCI has been applied extensively to climate interactions [25] where it was shown to outperform many alternative techniques. Another application includes the spread of fake news on Twitter during the 2016 US presidential election [31].

A comparison of PMIME and PCMCI (ParCorr) in mathematical detail will only appear later (for reasons that will become clear) in section 3.2.1.

2.4. Network theory

2.4.1. Basic concepts

A network is a structure used to model a set of objects (called **nodes**) and their interconnections (called **edges**). Network theory is essentially derived from graph theory, the branch of pure mathematics concerned with networks.

Definition 17. [32] Formally, a **network** is specified by the pair (N, E) where N is the set of nodes (labelled $i = 1, \dots, n$), and E is the set of node pairs (so that each pair corresponds to an edge).

A **simple network** has no self-edges (an edge that links a node to itself) or multi-edges (multiple edges connecting the same node pair).

In a **directed network** each edge has directionality, and so E is a set of *ordered* node pairs.

In a **weighted network**, each edge has a weight attached. Otherwise we have an unweighted network, in which every edge is weighted equally.

A network can be represented by an **adjacency matrix** A, with dimensions $n \times n$. The ij-entry represents the weight $w_{ij} \in \mathbb{R}$ of the edge from node i to node j. A will be symmetric if the network is undirected, and binary if the network is unweighted.

2.4.2. Network metrics

From a network we can calculate a number of useful metrics to quantify its features and topology. Definitions in this subsection are specific to weighted networks (where applicable), and are taken from [33] unless specified otherwise. The definitions for directed networks are mostly omitted here, but are given in [33]. Biological interpretations of each metric will be discussed later.

Definition 18. **Node degree** (k_i) is the number of edges the given node has attached. The weighted equivalent is called **node strength** and is denoted (for node i):

$$k_i^w = \sum_{j \in N} w_{ij}$$

For directed networks, we instead consider node **in-strength** and **out-strength** which sum over w_{ji} and w_{ij} respectively (and similarly for in/out-degree). We may optionally normalise node strengths by node degree.

Definition 19. **Node betweenness** (b_i^w) is the fraction of all shortest weighted paths in the network that pass through the given node. We omit the mathematical definition as it is no more enlightening than the qualitative description.

The idea of path length must be adjusted for weighted networks (where we use some transformation, usually **edge length = 1/edge weight**) and directed networks (where we can only travel on appropriately directed edges).

The definitions above are node properties (also known as centrality measures) rather than properties belonging to the entire network. We can get around this by instead considering, for example, the **degree distribution**. Similarly we may want to include some edge properties in our analysis, such as the **edge weight distribution**. Variables of interest based on these distributions could be the mean, standard deviation, number of weak/moderate/strong values (with user-chosen thresholds).

Definition 20. The **characteristic path length** of the network is the average shortest path length between all pairs of nodes. This is denoted by:

$$L^w = \frac{1}{n}\sum_{i\in N}\Big(\frac{1}{n-1}\sum_{j\in N, j\neq i} d^w_{ij}\Big)$$

where d^w_{ij} is the **shortest weighted path length** between i and j.

This is inversely related to the **global efficiency** (E^w_G) of the network, which is calculated by replacing d^w_{ij} with $1/d^w_{ij}$ in the above. This is perhaps a more useful measure because $1/d^w_{ij}$ is always defined, whereas d^w_{ij} can be infinity (when i and j are disconnected).

Definition 21. Modularity measures the extent to which a network has community structure (divisibility into non-overlapping modules $\{m\}$). It is denoted by:

$$Q^w = \frac{1}{W}\sum_{i,j\in N}\Big(w_{ij} - \frac{k^w_i k^w_j}{W}\Big) I\{m_i = m_j\}$$

where W is the total of all edge weights, I is an indicator function, and m_i is the module containing node i.

Definition 22. The **mean clustering coefficient** (related to **local efficiency** [9]) quantifies the prevalence of clustering around nodes on average, by counting the triangles around each node, and normalising by its degree. It is denoted by:

$$C^w = \frac{1}{n}\sum_{i\in N}\Big(\frac{1}{k_i(k_i-1)}\sum_{j,h\in N}(w_{ij}w_{ih}w_{jh})^{1/3}\Big)$$

Definition 23. The **assortativity** of a network is a correlation coefficient between the degrees of all nodes on two opposite ends of an edge. It can be interpreted as a measure of resilience of the network in the face of node or edge removal. Assortativity is denoted by:

$$R^w = \frac{\frac{1}{|E|}\sum_{(i,j)\in E} w_{ij}k^w_i k^w_j - \Big(\frac{1}{2|E|}\sum_{(i,j)\in E} w_{ij}(k^w_i + k^w_j)\Big)^2}{\frac{1}{2|E|}\sum_{(i,j)\in E} w_{ij}\big((k^w_i)^2 + (k^w_j)^2\big) - \Big(\frac{1}{2|E|}\sum_{(i,j)\in E} w_{ij}(k^w_i + k^w_j)\Big)^2}$$

Definition 24. [34] **Cost efficiency** (or economic efficiency) is defined as the difference between global efficiency (E_G) and **wiring cost** (K). A simple definition for wiring cost is just the connection density i.e. the fraction of existing edges over all possible edges in the network. A generalisation of wiring cost for weighted directed networks would be the sum of all defined edge lengths (optionally normalised by number of edges). In which case we denote the cost efficiency by:

$$E_C^w = E_G^w - K^w = \frac{1}{n(n-1)} \sum_{i \in N} \Big(\sum_{j \in N, j \neq i} \frac{1}{d_{ij}^w} \Big) - \sum_{i,j \in N} \frac{1}{w_{ij}} I\{w_{ij} \neq 0\}$$

where I is an indicator function.

Definition 25. [35, 36] **Small-worldness** is a metric used commonly in neuroscience, and describes networks that are highly clustered but has small characteristic path length. This is quantified by the metric:

$$\sigma^w = \frac{C^w}{C^w_{rand}} \Big/ \frac{L^w}{L^w_{rand}}$$

where the numerator is clustering coefficient normalised by its value for a random graph with the same number of nodes and edge density, and the denominator is characteristic path length similarly normalised by its random graph counterpart. The network is said to be small-world if $\sigma^w > 1$ (and denominator close to 1).

However, small-worldness has been shown to have weaknesses; in particular, it is density-dependent, and the constraint $\sigma^w > 1$ does not always characterise a small-world network. This is neatly summarised in [36].

It is crucial to note that network metrics are to be handled with care when used as relative measures. Clustering coefficient, modularity, efficiency, economic efficiency, and assortativity were found to differ significantly with network size [37].

2.4.3. Other classes of network

Lastly we introduce some terms commonly used in the literature but not yet covered above.

Complex networks are intrinsically linked to complex systems - the study of systems with many interacting components that together give rise to emergent hierarchical structures [38]. For example, the brain is a complex system, in which consciousness is an emergent property that cannot be observed by considering individual parts in isolation.

A **functional network** has edges that do not exist physically. For example, if edges represent correlation or social interaction. Conversely, a **structural network** is composed of physical connections, such as an electrical power grid. The brain is an interesting example because it has both a structural and functional network (edges could be based on anatomical wiring or signal correlation).

The edges of a **temporal network** may each be switched on and off with time. For example, road closures in a city's live road network.

A **multilayer network** is a convenient way to simultaneously encode (in multiple layers) different mechanisms/interaction types over the same node set e.g. structural and functional brain connectivity [39], or different brain frequency bands [40].

2.5. Machine learning

Machine learning algorithms 'train' themselves on data to perform specialised tasks. They are valuable tools in the field of **data mining**, which is all about the unearthing of patterns in large datasets [13]. The goal is to develop algorithms that will learn to generalise beyond the training data for reliable use on unseen examples [41]. We focus on just **supervised learning**, which is concerned with the situation where all datapoints are attached to an **outcome variable** (or label), as well as a number of **predictor variables**. For example, a sensible choice of predictors for the outcome variable 'house price' could be 'number of bedrooms' and 'local crime rate'. Even more specifically, the type of learning problem we will look at is **binary classification**, in which the outcome variable is binary (e.g. healthy or sick). We can test the performance of algorithms by training them on a partition of the dataset and evaluating classification performance (based on some loss function) on the remainder of the dataset.

2.5.1. Supervised learning techniques

We start by introducing one of the most elemental and familiar techniques used in machine learning.

Definition 26. A **logistic regression** model uses a logistic function to link a linear combination of predictor values to a probability estimate used for classification.

$$\eta_i = \beta_0 + \sum_{j=1}^{m} x_{ij}\beta_j \quad (i = 1, \ldots, n)$$

$$log\Big(\frac{p_i}{1-p_i}\Big) = \eta_i \quad \Longleftrightarrow \quad p_i = \frac{1}{1+e^{-\eta_i}}$$

Here x_{ij} represents the value of predictor j for datapoint i. The β coefficients are usually chosen via likelihood maximisation or square error minimisation given the actual outcomes y_i. p_i is the probability $P(y_i = 1)$, which we base a classification decision for the ith datapoint on. η_i is interpretable as the log-odds of the outcome $y_i = 1$ happening.

One problem with logistic regression is that it assumes independent observations [42]. In our study, we will violate this assumption because observations based on networks from the same patients will be dependent. We cannot get around this by building separate models for each patient (i.e. a segmented model) because the outcome (healthy or schizophrenic) remains constant for each individual. Moreover such a model structure is not helpful for operational use when faced with a completely new patient. For the same reason we will not look at modelling within-patient variability and between-patient variability separately (fixed/random effects model [42]).

Thus we will move onto neural networks and random forests, which capture patterns in the data without any underlying model assumptions. We will also introduce support vector machines, because despite having no theoretical justification in scenarios with dependent observations, they have still seen a lot of success regardless (see [43] for an analysis of this). These three techniques are also shown to perform well (on average) compared to logistic regression [44]. We will just present the basic concepts of these techniques, as the precise mechanisms are not central to the themes of this study.

An **(artificial) neural network** is a biologically inspired algorithm that feeds inputs through a progressive series of decision-making layers, culminating in a final prediction or decision. Layers are made up of nodes (or neurons) and inter-layer connections between nodes are each weighted numerically. After a feed-through, back-propagation is performed in order to optimise the weights of the neural network (e.g. using gradient descent), with respect to some loss function.

A **decision tree** recursively partitions the data in the predictor space, according to what results in the best split of outcome in the partition. This creates a series of branches that start from a root node and end in one of multiple leaf nodes, where a classification decision is made. Though easy to interpret, they are highly variable and require careful 'pruning' (see next subsection).

The solution to this is a **random forest**, a classifier that makes decisions based on majority vote of a collection of unpruned decision trees. However each tree is built on a random sample of the data, and at each split only a random subset of the predictors can be used (to introduce variety in the chosen predictors).

A **support vector machine (SVM)** attempts to find a separating hyperplane that optimally classifies the data in each dimension/predictor variable. Employing different kernel functions, we can obtain nonlinear separating boundaries. Calibration techniques that intelligently map outputs to probabilities (e.g. Platt scaling [45]) can greatly improve classification performance [44].

2.5.2. Performance measures

Suppose we have trained a classification algorithm and want to evaluate its performance. One way of measuring this is to look at classification accuracy. Though this is a fairly primitive measure, it is sufficient for our purposes since we are only interested in separability of healthy and schizophrenic groups, rather than detailed model comparison. Nonetheless, we will look briefly at one other measure:

Definition 27. Let S be a vector of scores and Y be a vector of corresponding outcomes (0 or 1). We assume a classification context where scores below some cutoff c result in the classification $\hat{Y} = 0$. Define the **receiver operating characteristic (ROC) curve** as a plot of the true positive rate, $F_0 = P(S < c|Y = 0)$ on vertical axis, against false positive rate, $F_1 = P(S < c|Y = 1)$ on horizontal axis.

The worst classifier (no discriminatory power) is then characterised by the diagonal $F_0 = F_1$, and the best classifier (perfect separation) is characterised as the line which reaches top left corner $(F_1 = 0, F_0 = 1)$ for some cutoff c. A possible measure of performance is therefore the **area under the ROC curve (AUC)**, which can be expressed as:

$$\begin{aligned} AUC = \int_0^1 F_0 dF_1 &= \int_{-\infty}^{\infty} F_0(c)F_1'(c)dc \\ &= \int_{-\infty}^{\infty} \Big[\int_{-\infty}^{c} f_0(s)ds\Big] f_1(c)dc \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I\{s < c\} f_0(s) f_1(c) ds dc \\ &= E_{s,c}[I\{s < c\}] \\ &= P(s < c) \end{aligned}$$

which is the probability that a randomly chosen score from a $Y = 0$ individual is lower than a randomly chosen score from a $Y = 1$ individual.

2.5.3. Obstacles in machine learning

The first obstacle arises even before the machine learning process has begun; that is, selecting the algorithm. We detailed several algorithms above, but there is in fact no way of knowing which will give you the best performance - it is highly dependent on your problem [41, 44]. Thus it may be wise to stick with simpler models as there will be less factors to consider/parameters to train (think Occam's razor!). Additionally, the results (whether success or failure) of an algorithm that you understand better will be easier to interpret and explain.

Arguably the most important issue to be aware of in machine learning is **overfitting**. This refers to the common pitfall of training a model that is too closely fit to seen data, and does not generalise well to unseen data. This inadvertent bias can creep in when using a model that is too complex and tries to explain too much of the seen data (including 'noise'), masking the simpler underlying trend. Depending on the algorithm, it can be very easy to overfit, even when aware of the hazard. Some ways to combat this are:

- **Regularisation**: a term penalising parameter dimension is added to the objective function. For example LASSO regression [27] adds a regularisation term for coefficient size, meaning that coefficients are shrunk during the optimisation stage, effectively performing variable selection in the process.
- **Cross-validation**: simulate multiple instances of training/testing data on a single dataset. This is done by splitting the data into k subsets, testing on one subset while training on the rest, for all the k possible partitions. This is known as k-fold cross-validation, or leave-one-out cross-validation when k equals the number of datapoints i.e. subsets are size 1.
- **Pruning** (for decision trees): in a decision tree, the splitting criteria at each node is controlled by user-specified parameters. To achieve a more parsimonious model, we 'prune' back the tree according to some optimal choice of parameter (e.g. minimiser of cross-validated standard error). This systematically removes unimportant splits with regard to classification, reducing chance of overfitting.

Another more preventable cause of overfitting is training a model on data which we later treat as unseen. For example, performing variable selection before partitioning into training and testing datasets. The performance on the 'unseen' data will then be optimistic.

A last issue to be aware of is the **'curse of dimensionality'**. This is neatly explained in [41] which we shall summarise and paraphrase here. In essence, standard intuition from our three-dimensional world no longer applies in very high dimensions. Similarity-based reasoning that machine learning algorithms depend on breaks down, because in high dimensions all examples look alike. For example, in high dimensions, most of the mass of a multivariate normal distribution is increasingly distant from the mean. So a high-dimensional orange (made up of points spread uniformly apart) will have most of its volume in the skin, not the pulp! If we were to inscribe this orange inside a high-dimensional box, most of the volume of the box would be outside of the boundary of the orange. This spells trouble for machine learning, where we often want to approximate shapes in data. Therefore it is important that we make efforts to reduce dimensionality (e.g. variable selection via LASSO regression).

3. Methodology

This chapter will be a detailed guide through all procedures used in this study. We reference the mathematics introduced in the previous chapter, and place them in the context of our EEG analysis.

3.1. The raw data

The data was supplied to us by Stefan Brugger from UCL, and consists of 9 healthy control patients and 7 schizophrenia patients. Among healthy patients there are 5 male and 4 female, with average age 33.6 years and a standard deviation of 10.8 years. Among schizophrenia patients there are 3 male and 4 female, with average age 26.4 years and a standard deviation of 10.0 years.

EEG recordings for all patients are taken from an eyes closed sample. 64 EEG channels (nodes) were used, with sampling frequency 1000Hz. The data is 'broadband' and not filtered to any specific EEG frequency band. The total observation time was 2784s for healthy patients and 2129s for schizophrenia patients.

Preprocessing (cleaning, channel removal etc.) of the data were done prior to us obtaining the files, and so we have to assume these were implemented correctly and appropriately. The SASICA [46] regime was used with MARA component selection, code available from https://github.com/dnacombo/SASICA.

Python-MNE [47, 48], which we used to analyse EEG data, is available from https://martinos.org/mne/stable/index.html.

The resulting time series were assumed to be stationary (definition 3), otherwise use of any of the causality measures becomes invalid, and analysis becomes impossible.

3.2. Network construction

From the raw time series data, the next task is to construct networks. From a wide range of choices, we decide to construct weighted, directed networks. At the cost of a more complex procedure, this allows for network models that are more faithful to the underlying mechanism behind communication in the brain. As stressed in [19], weightedness can be crucial in characterising alterations such as those caused by schizophrenia.

First we slice each time series into equally sized intervals. We choose a size of 1000 (i.e. 1 second of recording) as it was found to be large enough to capture interactions while still enabling efficient computation of causalities.

We also include the entire EEG node set (other than the reference electrode), as a decision to use any other subset would be unjustified without sufficient knowledge in neuroscience. Node selection methods such as the 'greedy algorithm' [13] do exist, but we do not use any in this study. Some patients had nodes ('bad channels') removed prior to us obtaining the data, and we should be aware that this can affect the metric values we calculate later [37].

We use PMIME (definition 15) and PCMCI (definition 16) as two separate means to assess between-node causality values:

- PMIME (for MATLAB) [10, 49] is available from http://users.auth.gr/dkugiu/Software_en.html.
- PCMCI/Tigramite (for Python) [11,29,50–52] is available from https://github.com/jakobrunge/tigramite.

The goal is to obtain an $n \times n$ adjacency matrix A (where n is the number of EEG nodes) for each second of each patient's EEG recordings. Both algorithms take as input the n time series of length 1000, and use methods described in the previous chapter to quantify between-node causality strengths. Both algorithms use in-built statistical tests to select only those interactions found to be statistically significant. Further justification for the use of PMIME and PCMCI can be found in section 2.3.

PCMCI has the added dimension of lagged dependencies (i.e. outputs an $n \times n \times \tau_{max}$ matrix which we denote B, where τ_{max} is the maximum lag chosen by the user). Thus we require an additional processing step to obtain the final adjacency matrix A:

$$A_{ij} = max_{\tau=1,\ldots,\tau_{max}}\{abs(B_{ij\tau})\}$$

Note that absolute value is necessary since PCMCI not only considers directionality of causality (which we are interested in), but also the positive/negative direction of this relationship (which we disregard as we do not care about whether electrical signals are driven up/down).

One problem we ran into was the sheer amount of computation time taken by PCMCI (with CMI-based test), despite having access to Imperial College's high performance computing cluster. This problem could perhaps be mitigated by making use of Tigramite's capacity to facilitate parallelised code; unfortunately this was beyond our programming capabilities. **As a result we instead used PCMCI (ParCorr)**. This involves replacing the CMI independence test with a test based on the **linear partial correlation**. The methods are analogous but ParCorr uses a conditional correlation coefficient in place of conditional mutual information; see [11] for details. Our motivation for this decision was as follows:

- ParCorr is very fast (though only a linear measure). For our data it was hundreds of times faster than the CMI_knn method.
- It could offer an interesting comparison with PMIME which is an information measure. This subtle difference could result in some interesting contrasts in results (discussed in section 3.2.1). If found to always produce similar results, this would eliminate the need to consider the more complicated information measure in the context of classifying schizophrenia.

Precise details of function use are contained in tables 3.1 and 3.2. We stuck to recommended parameter values in all cases.

It is worth noting that this section of the analysis is by far the lengthiest, with months' worth of computing time even for under 2 hours of recordings across the 16 patients. This is to be expected as we had approximately 5000 data intervals, each of which consists of over 60 time series of length 1000, amounting to around 300 million data points. Then performing a causality analysis for every interval is bound to take a while. Fortunately, we were able to use Imperial College's computing cluster, essentially enabling us to run long computations on up to 50 computing nodes at once without supervision. See appendix A for example code.

Function: PCMCI		
Parameter	**Value used**	**Interpretation**
cond_ind_test	ParCorr(significance='analytic')	conditional independence test method
Function: run_pcmci		
tau_max	5	maximum lag
pc_alpha	None	test significance level for PC step ('None'=optimise automatically)
Function: _return_significant_parents		
alpha_level	0.01	test significance level for MCI step

Table 3.1: parameter values for PCMCI (ParCorr) in Python.

Function: PMIMEsig		
Parameter	**Value used**	**Interpretation**
LMax	5	maximum lag
T	1	time horizon
nnei	5	number of nearest neighbours for kNN estimation
nsur	100	number of surrogates used for significance test
alpha	0.05	level of significance test

Table 3.2: parameter values for PMIME in MATLAB.

3.2.1. Difference between ParCorr and PMIME networks

Since ParCorr and PMIME are entirely different causality measures, the resulting networks will have different physical interpretations. Hence, network metrics will also have different interpretations. It is important we try to understand what these differences are.

ParCorr uses independence tests based on (linear) partial correlations to select causal parents. PMIME on the other hand uses (nonlinear) partial mutual information. We can better understand the conceptual differences by comparing the simpler unconditional formulae of covariance (which is correlation 'unnormalised' by individual variances) and mutual information.

Assume X and Y are discrete random variables with probability mass functions $p(x)$,

$p(y)$ and joint probability mass function $p(x, y)$. For covariance we have the following definition:

$$\begin{aligned} Cov[X,Y] &= E[XY] - E[X]E[Y] \\ &= \sum_{x,y} xyp(x,y) - \big(\sum_{x} xp(x)\big)\big(\sum_{y} yp(y)\big) \\ &= \sum_{x,y} xy[p(x,y) - p(x)p(y)] \end{aligned} \tag{3.1}$$

For mutual information, we have from definition 13:

$$\begin{aligned} I(X;Y) &= \sum_{x,y} p(x,y) log\Big(\frac{p(x|y)}{p(x)}\Big) \\ &= \sum_{x,y} p(x,y) log\Big(\frac{p(x,y)}{p(x)p(y)}\Big) \\ &= \sum_{x,y} p(x,y)[logp(x,y) - log(p(x)p(y))] \end{aligned} \tag{3.2}$$

Equations 3.1 and 3.2 illustrate that covariance and mutual information are both weighted measures over the variable space. The 'weights' we refer to are the terms in square brackets, and these quantify how 'far' each instance of the random variables are from being independent. But the important distinction is that while covariance is a weighted measure of the variables themselves, mutual information is a weighted measure of joint probabilities.

The implication of this is that ParCorr is conducting a type of averaging on the time series (for each time lag), and the PCMCI framework turns this into causalities. A linear parametric framework must be assumed, for which precise details can be found in [11]. The whole process is almost analogous to Granger causality, which we introduced in section 2.3.1.

PMIME on the other hand bases causality on weighted measures of the joint probability distributions between time series. It makes no parametric assumptions and hence is a nonlinear measure. We therefore expect a more accurate causality analysis if there are nonlinear dependencies in the system we try to model; studies have demonstrated this [17]. **But whether a more accurate causal network will improve machine learning classification of schizophrenia is a separate matter**.

We have looked at how ParCorr and PMIME *measures* differ. It is a much harder question to answer how ParCorr and PMIME *networks* differ. In fact, the precise difference in modelling implications is unknown. Making neurological interpretations such as "PMIME models the physical phenomena of directed neuronal communication, whereas ParCorr simply models a functional correlation between brain activities" is tempting, but at this stage is purely speculatory.

Network differences between ParCorr and PMIME may arise for two (or possibly more) reasons:

- Differences in causality measures discussed above place different emphasis on causal significance, giving rise to different network structure.
- Finite set of data points. It could be that in the limit of infinite statistics, the two measures tend to agree completely.

However we are unable to identify whether one or both of these reasons are contributing to network differences.

Our results in fact found that ParCorr matrices were much denser (in terms of edges) than PMIME matrices. It is tempting to conclude from this that PMIME is a much stricter measure than ParCorr. For our study this is true, but this is purely a result of the parameters we choose for each measure. Our response to this is discussed next.

3.2.2. Network backboning

In an attempt to maintain some symmetry between ParCorr and PMIME networks (remembering that network density affects metric values [37]), we reduced the density of the ParCorr matrices to match the density of PMIME matrices. This could have been done by controlling p-value of PCMCI significance tests; a large number of alternatives are also described in [13]. However, we used a **'backboning'** method [53] specialised for dense networks, which involves setting a threshold on edge significance after correcting for random noise. Code is available from http://www.michelecoscia.com/?page_id=287.

3.3. Network analysis

Having now obtained a network (i.e. adjacency matrix) for each second of each patient's recording, we would like to translate this into more meaningful information. To achieve this we calculate a number of metrics for each network. This task was greatly simplified by making use of the Brain Connectivity Toolbox (for MATLAB, others available) [33], which is available from https://sites.google.com/site/bctnet/Home. Code for all metrics in table 3.3 was taken from here, with the exception of cost efficiency which we calculate later after appropriate scaling.

Small-worldness was also considered for the analysis, though not a part of the Brain Connectivity Toolbox. However, due to technical difficulties (due to most backboned ParCorr matrices being disconnected networks) and time constraints, we were unfortunately unable to do so.

Another point worth mentioning is that PMIME matrices have a special interpretation for row and column sums. The difference between the ith row sum and ith column sum can be used to indicate whether the ith EEG node is a 'source' or 'sink' of information flow. Using Python-MNE we can then easily project this spatial information onto a diagram of EEG node locations (as seen in [3]). Unfortunately we did not have the time to pursue this area.

Metric	Meaning
in degree mean	mean across nodes of their in-degree
in degree sd	standard deviation across nodes of their in-degree
out degree mean*	mean across nodes of their out-degree
out degree sd	standard deviation across nodes of their out-degree
degree difference sd	standard deviation across nodes of their in/out-degree difference
in strength mean	mean across nodes of their in-strength (normalised by in-degree)
in strength sd	standard deviation across nodes of their in-strength (normalised by in-degree)
out strength mean*	mean across nodes of their out-strength (normalised by out-degree)
out strength sd	standard deviation across nodes of their out-strength (normalised by out-degree)
strength difference sd	standard deviation across nodes of their in/out-strength difference (normalised by in/out degree)
edge betweenness mean	mean across edges of their betweenness (tendency to be on shortest paths)
edge betweenness sd	standard deviation across edges of their betweenness
node betweenness mean	mean across nodes of their betweenness
node betweenness sd	standard deviation across nodes of their betweenness
density	proportion of edges existing among all possible edges
wiring cost	sum of edge lengths normalised by number of edges
global efficiency	related to the average time taken to travel between any two nodes in the network
modularity	degree to which network has community structure
clustering coefficient mean	mean across nodes of their clustering coefficient (prevalence of clustering within neighbourhood)
clustering coefficient sd	standard deviation across nodes of their clustering coefficient
assortativity oi	correlation between out-degree (at source) and in-degree (at target)
assortativity io	correlation between in-degree (at source) and out-degree (at target)
assortativity oo	correlation between out-degree (at source) and out-degree (at target)
assortativity ii	correlation between in-degree (at source) and in-degree (at target)
cost efficiency	difference between scaled global efficiency and scaled wiring cost

Table 3.3: list of network metrics considered - see section 2.4.2 for mathematical definitions. * Asterisks indicate metric was later discarded - details in section 3.4.1.

3.4. Machine learning on network metrics

At this stage we have two datasets - one for (backboned) ParCorr and one for PMIME. Each row/observation of the dataset corresponds to a single network, with predictors representing each network metric, and binary outcome, schizophrenia $=\{0,1\}$. We implemented all procedures described in this section using the R programming language.

3.4.1. Preliminary data analysis

In the ParCorr dataset there were 12 rows containing missing values, due to some metrics being incomputable after the backboning algorithm deleted a large number of edges. We deleted these observations after judging that the bias this might cause was negligible, considering the dataset contained nearly 5000 observations.

We also delete two variables: out-degree mean and out-strength mean. This is because they are directly collinear with in-degree mean and in-strength mean, respectively. On a global level, total in-degree/strength and total out-degree/strength are equal, so it is no surprise that the means are collinear.

Before running any machine learning algorithms, it is often a good idea to plot some visualisations, such as comparative histograms for each metric. In this case we can safely scale metric values (between 0 and 1) for each individual patient, to mitigate the effects of between-patient variation in experimental conditions etc (although some of these differences may be important).

For each variable we can perform a **permutation test** on the absolute difference in mean between healthy and schizophrenic groups. This involves calculating the observed mean difference, and seeing where this lies in relation to the mean differences for all possible permutations of outcome labels for the data (keeping group ratio the same). In practice, we only take a large sample of possible permutations. We then check the percentage of times the observed absolute difference is exceeded, and conclude that the groups are differently distributed if this percentage is below 5%.

3.4.2. Standardisation and cross-validation

By **standardising** a variable, we mean to subtract its mean and divide by its standard deviation. Standardising all variables puts them all on the same scale. This makes regression coefficient magnitudes comparable, which is particularly necessary for LASSO regression due to its penalty on coefficient magnitude. Coefficients will also become interpretable as the increase in log-odds when the corresponding variable increases by one standard deviation.

Since we are interested in classification performance on *unseen* data, it's unrealistic to check algorithm performance on data it has been trained on. As such, we must set aside an independent testing dataset that contains no information about the training dataset used to train the algorithm. Therefore we must not standardise any variables before splitting the data into testing and training, as then the testing dataset will hold information about the training dataset, and won't be truly 'unseen'. So in order to best simulate operational conditions of model use, we must standardise the testing set using the mean and standard deviation of the training set.

It also no longer makes sense to standardise each patient individually, as this would require finding the training means/std. devs for each patient and scaling this patient's testing data by the same quantity (which implicitly assumes this patient is not unseen). Thus we standardise patients together, despite this meaning potential accuracy losses due to variation in experimental conditions per patient etc.

Calculating cost efficiency, which is a difference between two scaled variables, must also be done with care.

This can all be incorporated into our k**-fold cross validation** procedure:

1. Randomly split the data into k subsets.
2. For i=1:k
   (a) Allocate testing to subset i and training to the rest.
   (b) Standardise training set and use train mean and std. dev. to scale testing set.
   (c) Calculate cost efficiency for training set and standardise it. Calculate cost efficiency for testing set and scale it by training cost efficiency mean and std. dev.
   (d) Train classification algorithm on training set.
   (e) Predict outcomes for training and testing sets using algorithm.
   (f) Calculate accuracies (by comparison with actual outcome) for both sets.
3. Compute mean training and testing accuracy across the k folds.

Higher k increases computation time and leads to more correlated training sets in each fold, but lower k gives rise to larger between-fold variance of accuracy estimates. We use $k = 5$. See appendices C.1 and C.2 for example code.

Lack of data is often declared to be a weakness in studies of EEG activity, as it is difficult and expensive to gather many patients with a particular disease. Cross-validation combats this by simulating performance on unseen data, while also highlighting the extent of overfitting. Training accuracy is optimised by the model and so is generally higher than testing accuracy. If the disparity between the two is large, clearly the algorithm is not generalising well to unseen data, and is instead fitting on idiosyncrasies of the training data. It is important to realise that we are not erasing the problem of overfitting, but rather, alerting ourselves to its extent under a particular model.

Performance measures other than accuracy are also possible (e.g. AUC) but our focus is more on interpreting prediction accuracy rather than detailed model comparison.

3.4.3. Classification algorithms

As each may perform better under different circumstances, we implement a number of algorithms (introduced in section 2.5.1):

- Logistic regression (see table 3.4)
- LASSO regression (see table 3.5)
- Random forest (see table 3.6 and appendix C.3 for example code)

- Support vector machine with polynomial kernel (see table 3.7)
- Single-hidden-layer neural network (see table 3.8 and C.4 for example code)

Though we performed some parameter tuning to maximise cross-validated testing accuracy (e.g. appendix C.5), this was in one dimension at most. Optimising over the entire space of possible parameters is certainly a direction to explore in future.

Neural network was found to be highly variable in performance and so we added the parameter 'num_tries' to refit multiple times and pick the neural network which minimises loss function.

Function: glm		
Parameter	**Value**	**Interpretation**
family	binomial(link="logit")	specify link function for general linear model (logistic regression uses logistic function)
maxit	5000	maximum iterations for iterated weighted least squares algorithm
(cutoff)	(optimise for accuracy)	threshold for classification decisions

Table 3.4: parameter values for logistic regression in R.

Function: glmnet (package: glmnet)		
Parameter	**Value**	**Interpretation**
alpha	1	specify regularisation penalty (1 for LASSO)
standardise	FALSE	choose to standardise manually
lambda	cv.glmnet(...)$lambda.1se	specify LASSO shrinkage parameter. (Value is chosen to be 1 std. error away from value that minimises cross-validated error. This is for robustness against overfitting.)
family	"binomial"	specify form of general linear model
(cutoff)	(optimise for accuracy)	threshold for classification decisions

Table 3.5: parameter values for LASSO regression in R.

Function: randomForest (package: randomForest)		
Parameter	**Value**	**Interpretation**
importance	TRUE	variable importance assessment
nodesize	(optimise for accuracy)	minimum terminal node size (if equal to 1, decision trees branch until node contains only 1 observation)

Table 3.6: parameter values for random forest in R.

Function: svm (package: e1071)		
Parameter	**Value**	**Interpretation**
kernel	"polynomial"	parametric form of decision boundary
degree	(optimise for accuracy)	polynomial degree of kernel

Table 3.7: parameter values for support vector machine in R.

Function: nnet (package: nnet)		
Parameter	**Value**	**Interpretation**
size	(optimise for accuracy)	number of neurons in hidden layer
MaxNWts	5000	maximum number of weights
maxit	300	maximum iterations per neural network fit
(num_tries)	10	number of neural networks to fit before choosing best fit

Table 3.8: parameter values for single-hidden-layer neural network in R.

3.4.4. Interpreting results

To make sense of our results we look at the following:

Algorithm performance measures:

- **Optimal cross-validated testing accuracy** - to assess discriminatory power in classifying schizophrenia for unseen data (after optimisation of parameters).
- **Loss from training accuracy** - to assess the extent of overfitting.

Variable performance measures:

- **Logistic regression coefficient magnitudes** - to compare linear effect size between variables. Coefficient sign is also of interest (positive means high variable values are indicative of schizophrenia; negative means high variable values are indicative of healthy). But this is generally a naive measure, as changing the included predictors in the regression model will change the magnitudes. Also magnitude alone is misleading if standard errors vary wildly.
- **Logistic regression coefficient p-values** - p-values for the null hypothesis that coefficient size equals zero i.e. that variable is unrelated to schizophrenia. By taking the negative log-transformation of the p-values we may obtain a heuristic importance measure similar to the others in this list.
- **LASSO regression coefficient magnitudes** - similar to logistic regression counterpart. However perhaps a more valid measure, as this time coefficient magnitudes were included in the optimisation procedure as a penalty, to encourage shrinkage of unimportant variable coefficients.
- **Random forest Gini-based importance** - measures the average decrease in node impurity when splitting a decision tree based on variable in question. In other words, how good is the variable at separating healthy and schizophrenic patients.
- **Variable AUC** - using variable as a score we compute the AUC (see definition 27) as a measure of group separation. We saw this can be interpreted as the probability that a randomly chosen schizophrenic patient has higher value in this variable than a randomly chosen healthy patient. An AUC of 0.5 represents the worst result. The AUC values corresponding to perfect separation are 1 (meaning high variable values for schizophrenia) and 0 (meaning high variable values for healthy). Hence in the plots of the next chapter we visualise AUC minus 0.5, and look for large magnitudes.

This is in fact closely related to the **Gini index** used by random forest. Note that AUC is a measure derived from the raw data, and not a by-product of any algorithm.

We note that all of these measures give variable importance with regard to classifying ability. So for example, if modularity were found to be 'important' for schizophrenia, this would mean that schizophrenia patients have high modularity **compared to healthy patients**. It does not necessarily mean schizophrenia patients all have very high modularity.

4. Results and interpretations

In this chapter we report the results of our procedures, and offer possible neurological interpretations. Such interpretations are based on background reading rather than our own expert knowledge.

4.1. An unforeseen finding

First we call attention to a major unforeseen finding of our study. In section 3.2.2 we mentioned that in order for ParCorr and PMIME matrices to have comparable densities, we used a sophisticated thresholding method [53] on ParCorr matrices. However we discovered that classification performance across all algorithms was better on the unthresholded ParCorr matrices!

Network densities are shown below in table 4.1. The improvement (to be detailed in section 4.4) appears to come down to the sheer amount of information contained in the unthresholded matrices, despite being less rigorously 'pruned'. Note that in spite of the near-full densities, these matrices still give information about schizophrenia status. This heavily suggests that it is the **edge weights that have explanatory power**, and that **rigorous procedures to select edges are in fact detrimental in this context**. Interestingly, this agrees with findings that weak edges are instrumental in classifying schizophrenia [19]. It is definitely worth follow-up studies to see if there is a sweet spot for network density with regard to classification accuracy.

ParCorr (unthresholded)		ParCorr (backboned)		PMIME	
Healthy	**Schizophrenia**	**Healthy**	**Schizophrenia**	**Healthy**	**Schizophrenia**
0.978	0.991	0.074	0.035	0.055	0.052
0.984		0.054		0.054	

Table 4.1: network densities for the three batches of matrices.

As such, for the remainder of this chapter we will be considering three batches of networks, with the added batch **'Unthresholded ParCorr'**, taking care not to compare metric values for this batch with the others.

4.2. Exploratory plots

In any data analysis project, one should first visualise the data. We plotted scaled histograms for each metric after scaling for each patient, in order to compare distributions for healthy and schizophrenic groups. We include just one such plot here, for lack of space, and as most were not particularly enlightening.

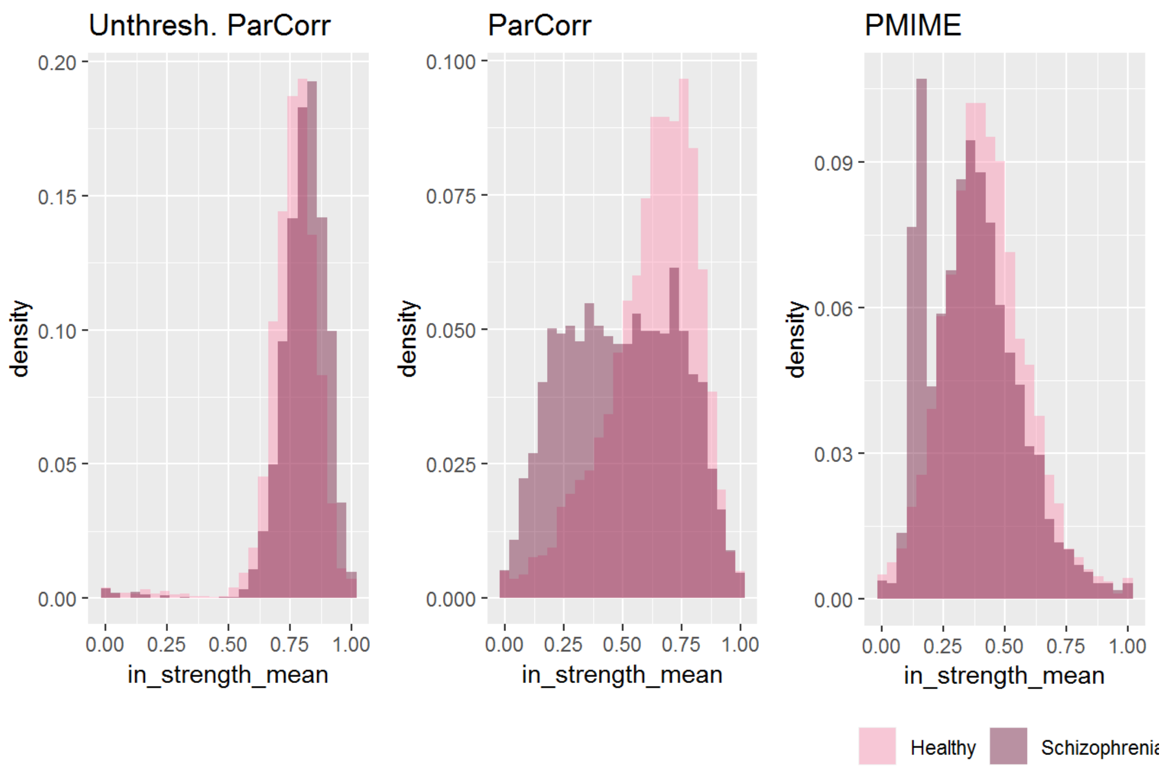

Figure 4.1: Comparative histograms for mean in-strength after scaling for each patient.

The (backboned) ParCorr plot of figure 4.1 in particular seems to agree with the well-established result that the schizophrenic brain has weaker functional connectivity [7, 19, 54]. The PMIME plot also reveals a large spike - existing only for the schizophrenia group - in the number of matrices with low in-strength. This is highly irregular considering the shape of the rest of the distribution, and possibly warrants further investigation.

Next we showcase some 2D scatter plots of the data in figures 4.2 and 4.3.

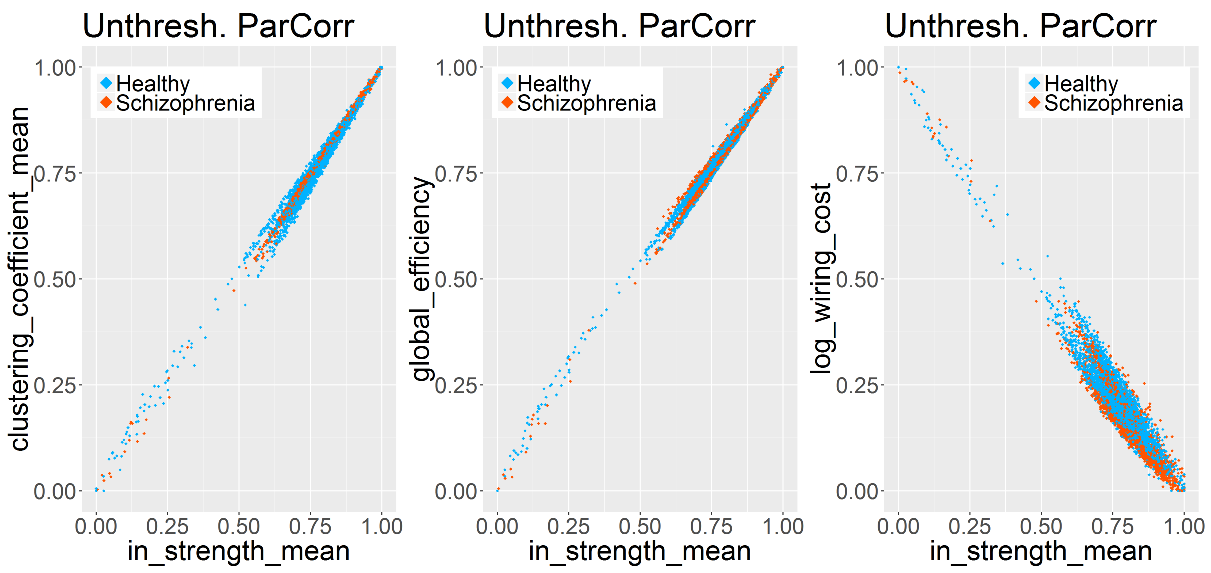

Figure 4.2: scatter plots exhibiting unexpected linearity for unthresholded ParCorr batch. Each point represents a single network based on 1000 data points (1 second of recording).

We noticed that for unthresholded ParCorr networks, in-strength mean had a strikingly linear relationship with clustering coefficient mean, global efficiency and log(wiring cost); see figure 4.2. In future studies it may therefore be sensible to exclude these collinear variables (for unthresholded ParCorr), as just knowing in-strength mean is sufficient for the rest. These relations are fairly intuitive - higher mean in-strengths create a more efficient but more costly network (and clustering coefficient measures local efficiency [9]). However it is unknown why this was only observed for unthresholded ParCorr.

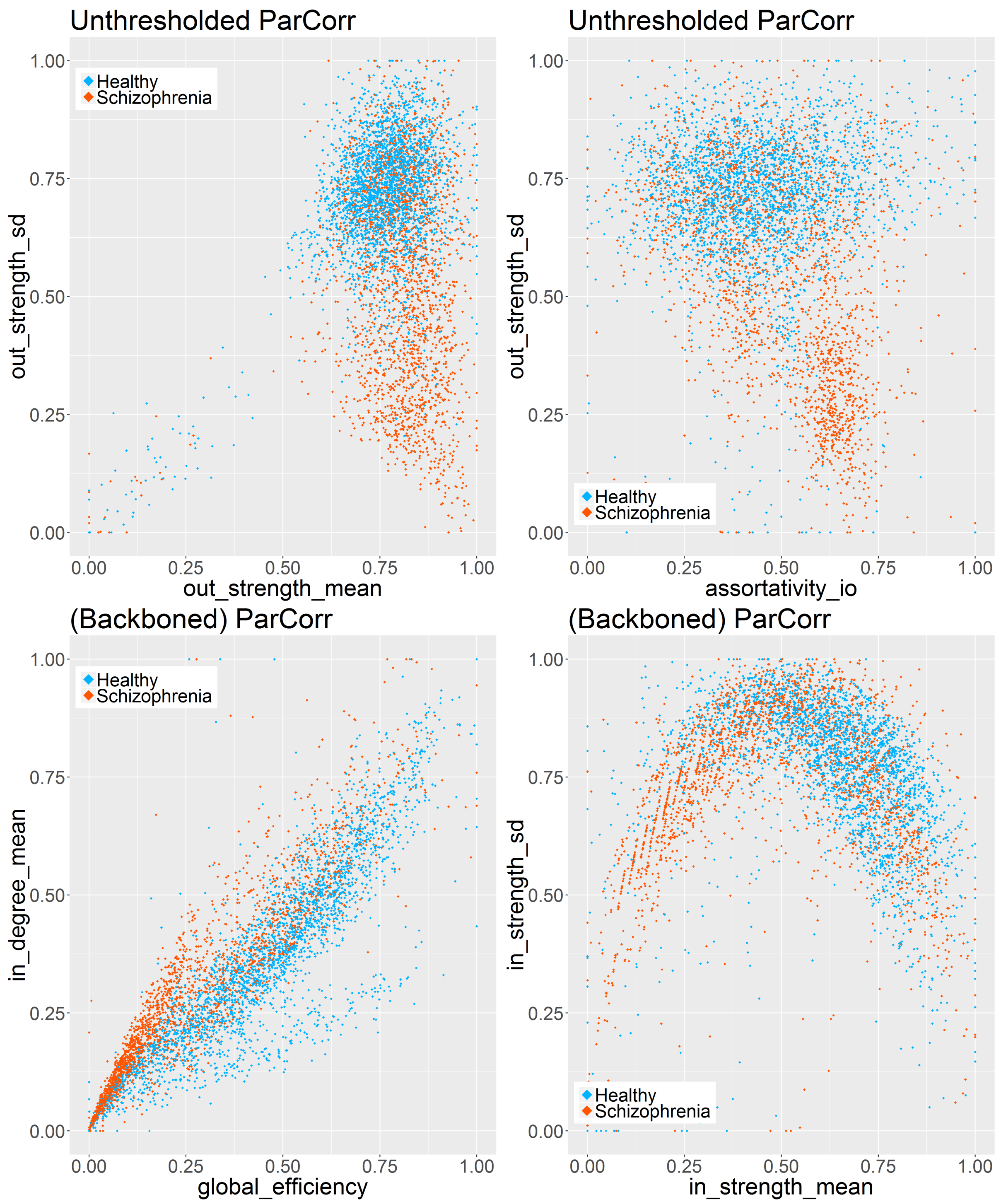

Figure 4.3: scatter plots after variable scaling; network batches given in plot titles. Each point represents a single network based on 1000 data points (1 second of recording).

The plots in figure 4.3 show that a high degree of separability is possible even in two dimensions (though for most variable combinations this wasn't possible). The top two plots demonstrate that out-strength standard deviation is an especially good separator of the two groups for unthresholded ParCorr networks. Low values of out-strength sd (which means low variation in node out-strengths) are mostly only observed in schizophrenic brains, or healthy brains with low mean out-strength. We can interpret this as schizophrenic brains having greater connection diversity, as out-strength is high with a low amount of variation. This agrees with other studies [7, 19].

Though it is only slight, the top-right plot of figure 4.3 indicates that the schizophrenic brain has higher in-out assortativity. This means nodes of high in-degree are more likely to be connected with nodes of high out-degree, and this has been associated with robustness to targeted hub removal in [8] (for undirected assortativity). Increased assortativity for schizophrenics was also observed in [8], and increased robustness to random attack was observed in [7].

Now we move on to the next batch of networks, (backboned) ParCorr. The bottom-left plot of figure 4.3 shows a clear tendency for schizophrenia brain networks to have lower global efficiency than healthy brain networks, given the same mean in-degree. Schizophrenic brains were also found to be less efficiently wired in [8, 54] (although [55] reports increased global efficiency).

Lastly, the bottom-right plot of figure 4.3 shows quite clearly some sort of quadratic relationship between in-strength sd and in-strength mean. We are unsure whether this result has any significance, but its rough interpretation is that when mean in-strength is particularly low or high, there is less variation across all node in-strengths. This is exhibited by both healthy and schizophrenic brains. On average, the schizophrenia group has lower in-strength, which supports results about weaker functional connectivity for schizophrenia [19, 54].

No noteworthy scatter plots were found for the PMIME batch of networks. We will see this reflected in its relatively low classification performance, in section 4.4.

4.3. Parameter optimisation

We now turn to the results of our machine learning algorithms. For details of the parameters we used, see section 3.4.3. Any unmentioned parameters can be assumed to have been set to their default values in R.

For logistic and LASSO regression, cross-validated testing accuracy peaks at probability cutoffs close to 0.5, as seen in figure 4.4. This is expected, and just shows the models are giving well-calibrated probability estimates. We omit the optimisation plot for logistic regression as it is similar to LASSO regression.

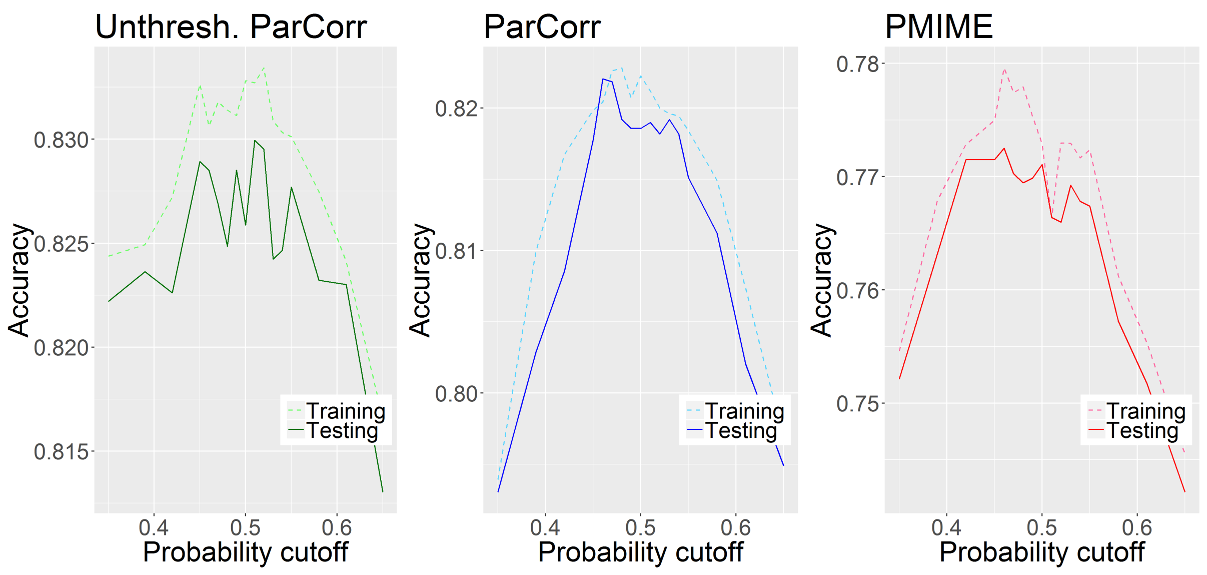

Figure 4.4: Cross-validated performance of LASSO regression while varying probability cutoff. From left to right, optimal values were 0.51, 0.46, 0.46.

The default for random forests is to set the minimum terminal node size to 1, meaning trees are fitted until all points are classified. This is reflected in the figure 4.5 which shows a training accuracy of 1 for nodesize 1. This is clearly overfit, yet this doesn't seem too problematic as testing accuracies remain high and stable. As expected, increasing nodesize reduces overfitting. However testing accuracy also decreases and thus there is no merit to using nodesizes higher than 10.

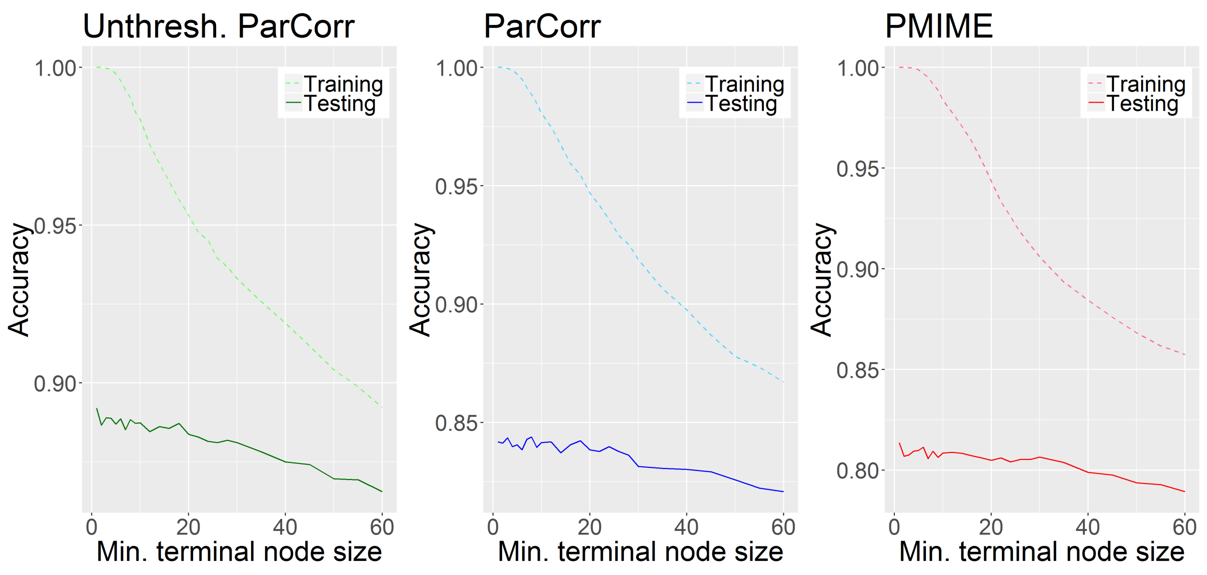

Figure 4.5: Cross-validated performance of random forest while varying nodesize. From left to right, optimal values were 1, 8, 1.

For support vector machine we optimise the polynomial degree of the classification boundary. The results, in figure 4.6, are quite illogical and we should perhaps consider disregarding this model. We should expect to see training accuracy strictly increase with degree as parametric degrees of freedom increases. The reason this does not happen can probably be attributed to unsuccessful numerical optimisation.

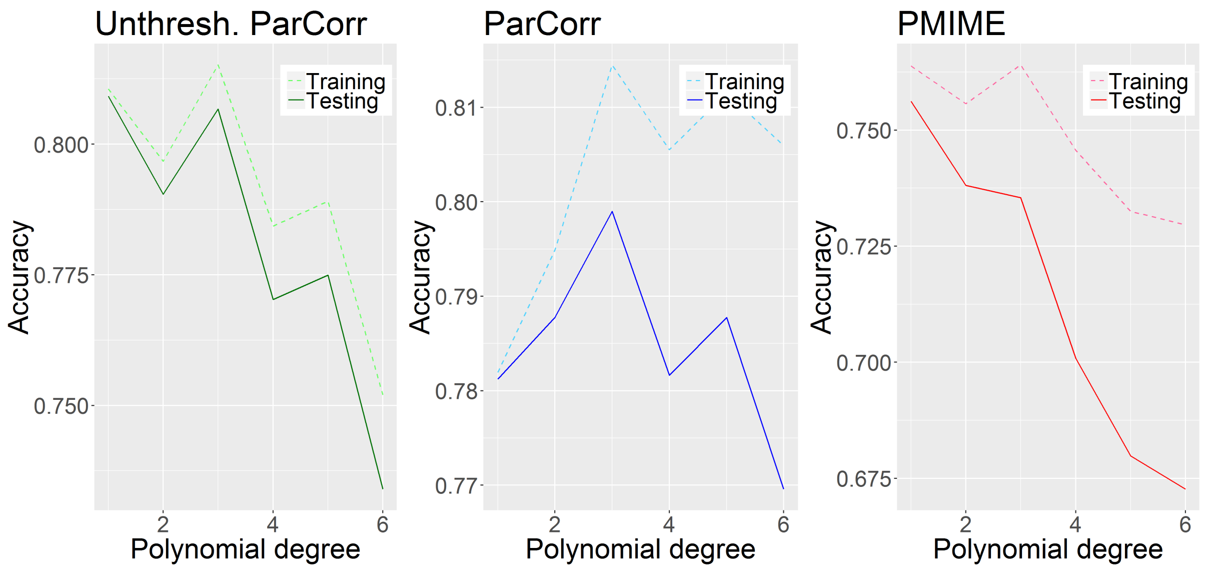

Figure 4.6: Cross-validated performance of support vector machine while varying polynomial degree. From left to right, optimal values were 1, 3, 1.

For the neural network we optimise the number of neurons in the single hidden layer. Example code in appendix C.5. Figure 4.7 shows that training and testing accuracy quickly diverge as the model becomes overfit on training data. This behaviour meets our theoretical expectations. After around 25 neurons, the neural network is capable of completely fitting the training data. It is unlikely to be a coincidence that we also have around the same number of predictors!

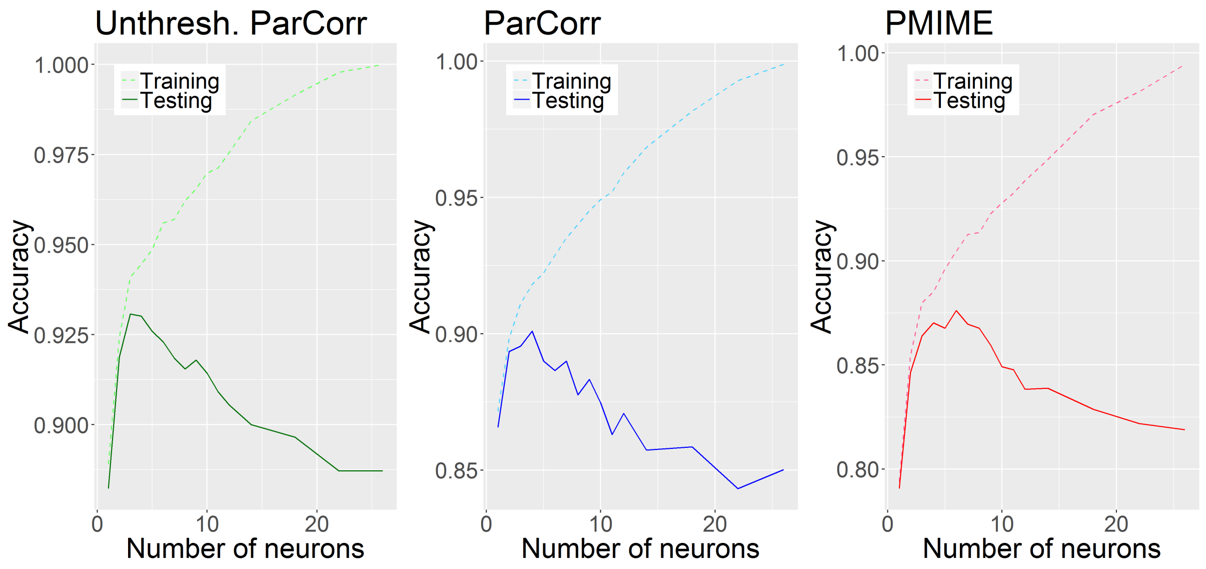

Figure 4.7: Cross-validated performance of neural network while varying number of neurons. From left to right, optimal values were 3, 4, 6.

4.4. Algorithm performance

Having optimised the models with respect to parameters, we can look at the resulting performance in terms of classification accuracy. Tables 4.2, 4.3, 4.4 below report the results, and this is summarised by a graphic in figure 4.8.

Unthresholded ParCorr		
Model	Testing accuracy	Loss from training accuracy
Logistic regression	0.878	0.003
LASSO regression	0.831	0.001
Random forest	0.886	0.114
Support vector machine	0.808	0.002
Neural network	0.928	0.013
Average	0.866	0.026

Table 4.2: Classification accuracies for unthresholded ParCorr.

(Backboned) ParCorr		
Model	Testing accuracy	Loss from training accuracy
Logistic regression	0.840	0.003
LASSO regression	0.821	0.001
Random forest	0.840	0.149
Support vector machine	0.799	0.016
Neural network	0.900	0.017
Average	0.840	0.037

Table 4.3: Classification accuracies for (backboned) ParCorr.

PMIME		
Model	Testing accuracy	Loss from training accuracy
Logistic regression	0.788	0.007
LASSO regression	0.772	0.004
Random forest	0.806	0.194
Support vector machine	0.758	0.005
Neural network	0.871	0.032
Average	0.799	0.048

Table 4.4: Classification accuracies for PMIME.

Our hypothesis for these results is that the sheer amount of information contained in the high-density unthresholded matrices provided more comprehensive metric data for machine learning. The practical implications of this are large - ParCorr networks are several times faster to compute than PMIME matrices. Knowing that networks based on linear causality measures are sufficient for accurate patient diagnosis would be valuable information to physicians, for example.

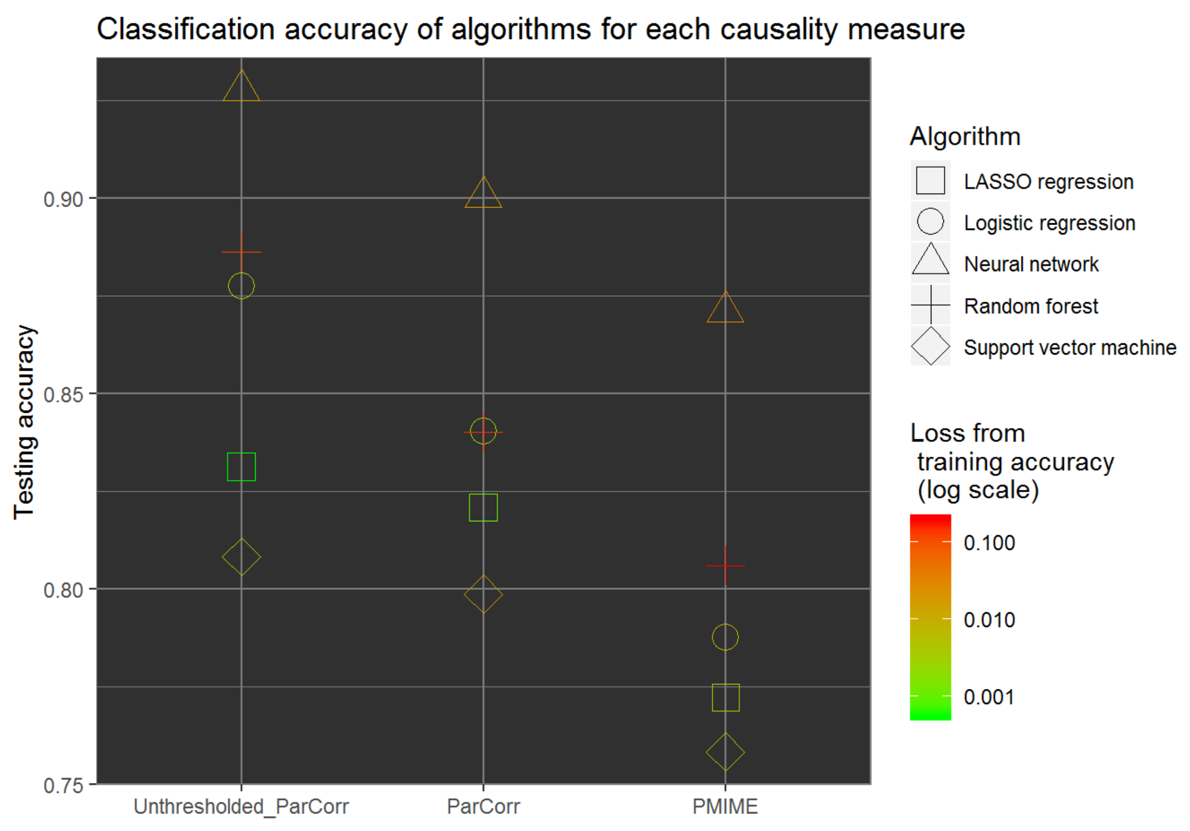

Figure 4.8: summary of algorithm performance for all three batches.

4.4.1. Algorithm comparison

Figure 4.8 makes the superiority of the neural network for this data very apparent. We used a single-hidden-layer neural network which is the simplest possible class of neural network. This suggests further improvement may even be possible with more hidden layers.

Random forests are the next best performer. They also have the largest loss from training accuracy, but this is due to the nature of the algorithm, which aims for a perfect fit on the training data, rather than a flaw to beware of. This is also the case for neural networks.

The regression models have performed reasonably in spite of theoretic restrictions such as independence of observations. They are also very robust to overfitting. Surprisingly logistic regression performs strictly better than LASSO regression, despite the latter being a strict upgrade to the former, in theory. It's possible that this purely boils down to the efficiency of the numerical optimiser, which differs between the two.

Support vector machine chose linear boundaries for unthresholded ParCorr and PMIME (as seen in figure 4.6), which explains the high robustness to overfitting (low model complexity means less capacity to 'overexplain'). Unsurprisingly this gives the worst classifier (though not by far).

4.4.2. Batch comparison

We've already touched on why unthresholded ParCorr gives the best performance - machine learning extracts insights from data, and unthresholded provides the most raw numbers. However we did not address why performance on ParCorr might be better than on PMIME. Our hypothesis to this is a rather unexciting one - in backboning the ParCorr matrices we accentuated the difference in density between healthy and schizophrenic patients (see table 4.1). This density difference is much smaller in PMIME. Since network metrics all depend on density, the algorithms can predict schizophrenia by instead predicting network sparsity. From a purely practical viewpoint, we can argue that nothing matters as long as predictions are accurate; the downside is that model interpretation (which we cover next) becomes less valid.

4.5. Variable importance

We now try to gauge the importance of the different predictor variables with regard to predicting schizophrenia outcome, opening the way for neurological interpretation.

We start by reporting the results of the permutation test for absolute mean difference in each metric, described in section 3.4.1. For unthresholded ParCorr matrices, all metrics are significant at the 5% level. For (backboned) ParCorr, only assortativity io was insignificant. For PMIME, only wiring cost was insignificant. These results tell us there is very good separability of the two groups within almost all of our chosen metrics. Other than this, the results are not especially useful, and so we move on to analysis of our classification models.

We analyse each batch in turn, using the variable performance measures from section 3.4.4. Our variable importance results are lengthy and so we summarise them in section 4.5.4, where we also plot stacked histograms showing combined importance.

Support vector machine and neural networks are 'black box' algorithms (as inner workings are not reasonably possible to interpret) and so do not feature in this section.

Cost efficiency was excluded automatically by logistic regression models in R due to high collinearities, and hence it does not appear under transformed p-value plots.

As discussed in 3.2.1, the network metrics for ParCorr and PMIME will gain slightly different neurological interpretations, so we should not be surprised to see entirely different sets of metrics being relevant for each batch.

4.5.1. Unthresholded ParCorr

First we refer to the regression model results of figures 4.9 and 4.10. Of the three batches, unthresholded ParCorr was the only one to show major changes in the regression model after applying a LASSO penalty. This appears to be due to the coefficient sizes being larger in general (compared to 4.14 and 4.19), causing the regularisation penalty to have a more drastic effect on the original logistic regression model. Although normally the LASSO regression model should hold more weight, we saw in section 4.4 that the logistic regression model in fact performs better.

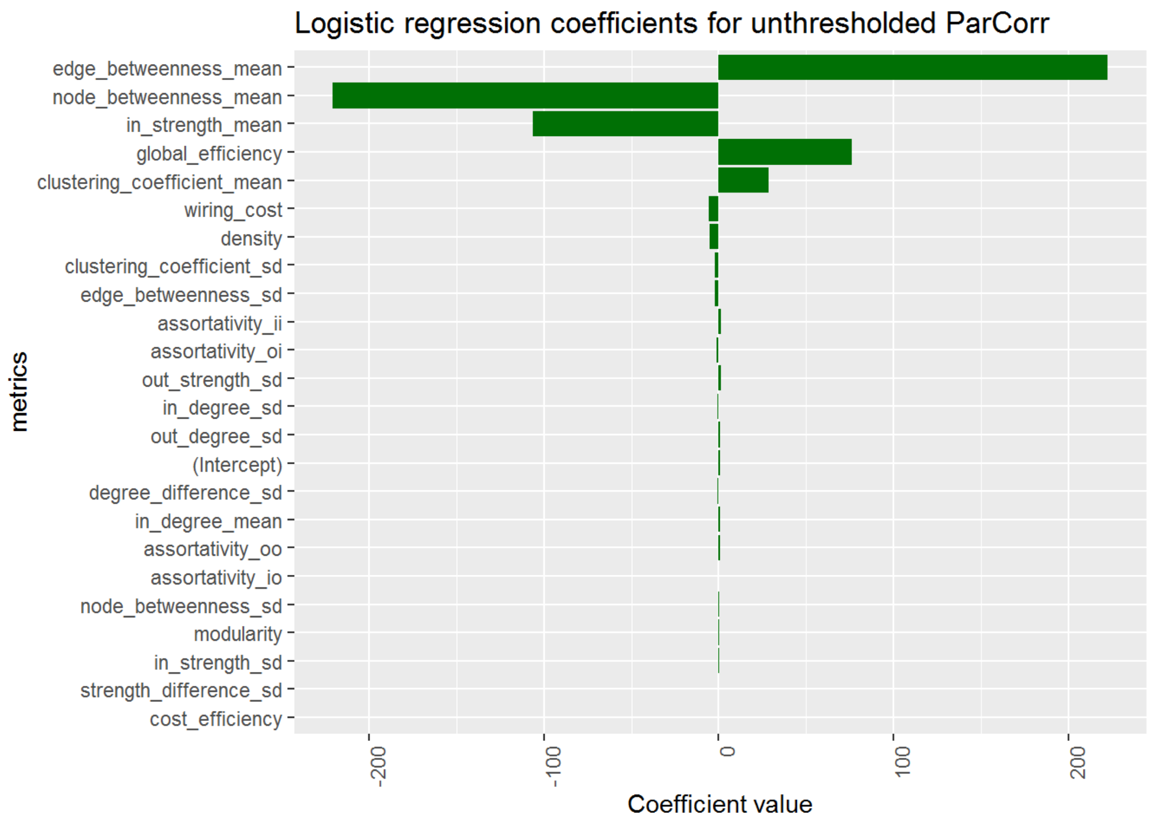

Figure 4.9: logistic regression coefficients. Positive sign means model associates high metric values with schizophrenia. Negative means low values associate with schizophrenia.

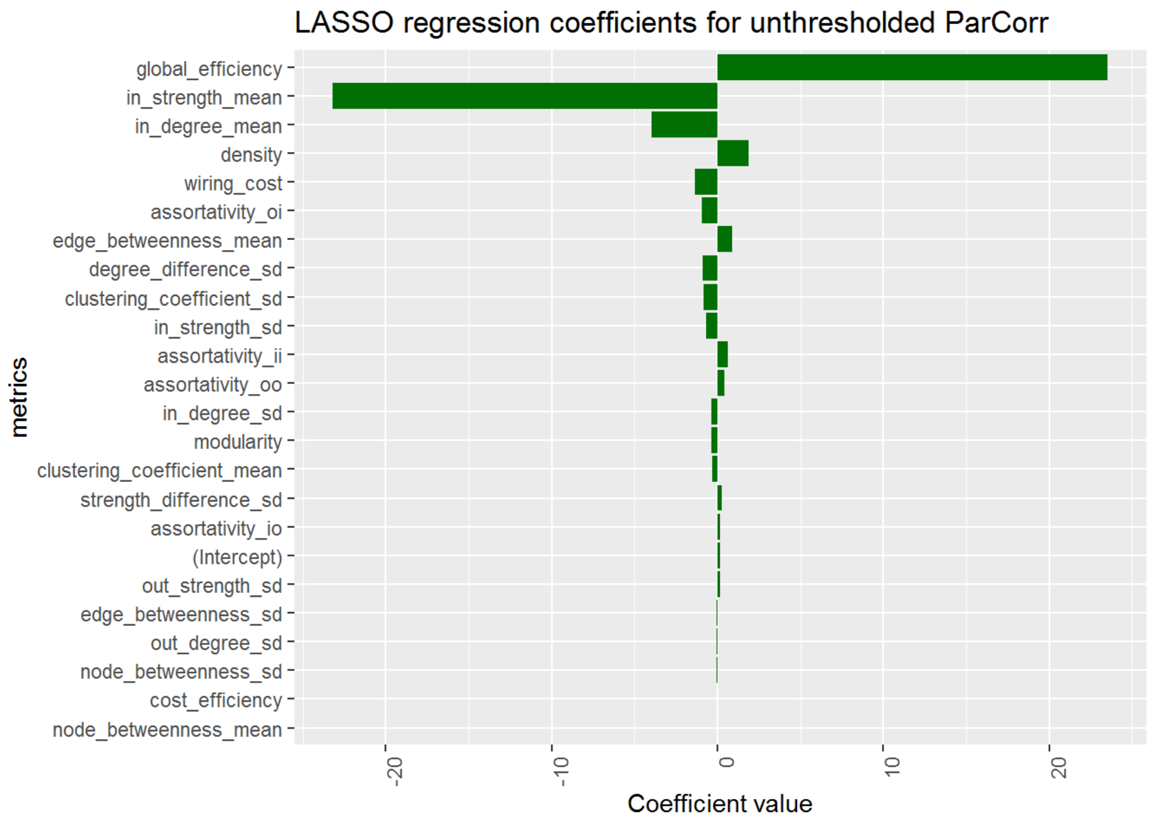

Figure 4.10: LASSO regression coefficients, which have undergone shrinkage.

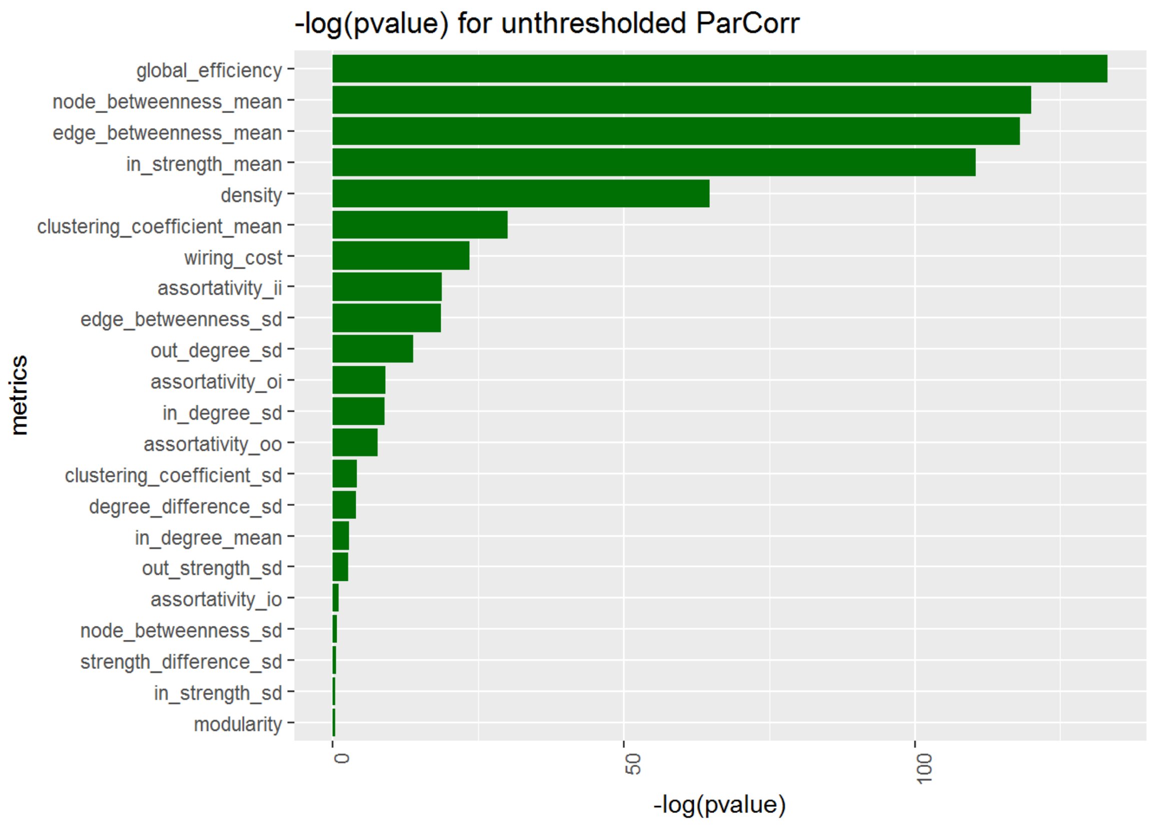

Figure 4.11: logistic regression transformed p-values, as a heuristic importance measure.

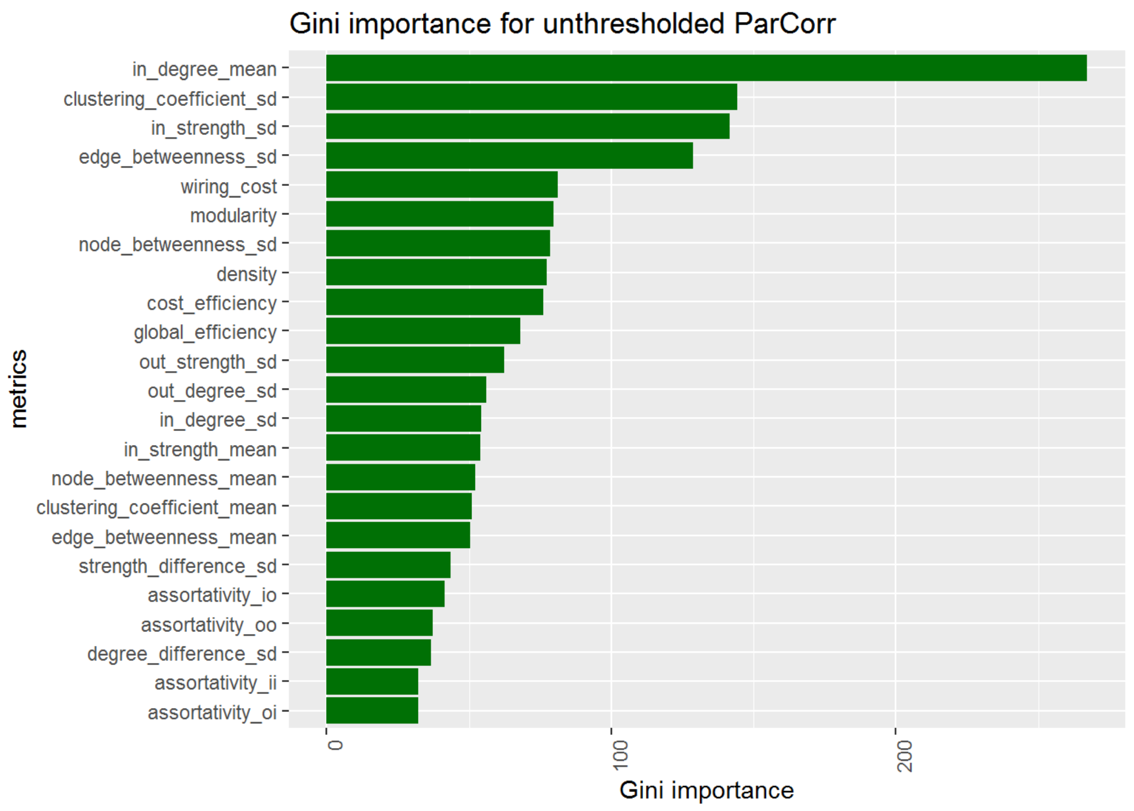

Figure 4.12: Random forest Gini importance, which quantifies group separating ability.

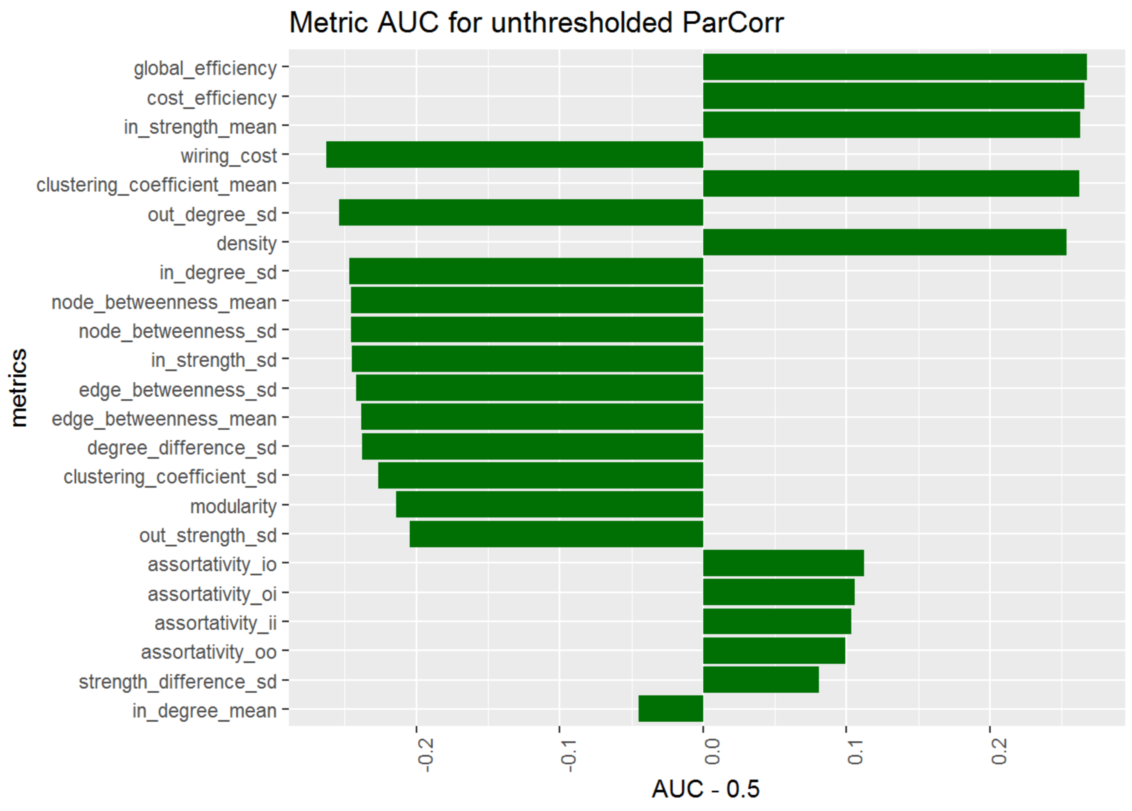

Figure 4.13: Rescaled AUC using individual variables as scores, resulting in a measure of group separation. Positive sign means schizophrenia group has higher values in this metric. Negative sign means schizophrenia group has lower values.

Before further analysing the regression coefficients, notice that there are some major disagreements with AUC importance (figure 4.13) with regard to direction of association. For example, edge betweenness mean and in-strength mean point in opposite directions for the two measures. Since AUC is derived from the data, we are more inclined to trust these results. For example, it agrees with table 4.1 that schizophrenia networks are more dense. Therefore the regression coefficients come into question. Other than blaming a poorly optimised model, there is not much we can say to explain such abnormal results. It's possible that coefficients were heavily skewed by a small number of extreme values - this would not greatly impact AUC, which looks for separability purely based on orderings. This is unlikely as variables were all standardised, but not impossible. Further investigation is necessary.

In any case, logistic regression and AUC (figures 4.9 and 4.13) agree that low node betweenness is indicative of schizophrenia. We don't attach too much weight to this observation however, as the addition of the LASSO penalty debunks the claim, as seen in figure 4.10. The LASSO regression model instead emphasises high global efficiency for schizophrenia, which is also reinforced by the AUC. Higher global efficiency for schizophrenia was also observed in [55], but other research [8,54] has shown reduced efficiency.

Figure 4.12 presents Gini-based importance from random forest which, like AUC, relates to group separability. Yet for unknown reasons they are in complete conflict with each other - in-degree mean is the most important separator in a random forest context, but the least important when looking at it in isolation (via AUC). The only explanation we can offer for this is that in-degree only shows its worth as a separator when looked at in combination with other variables. This dependence could be verified by building a regression model that includes all interaction terms with in-degree.

The AUC plot (figure 4.13) continues to defy our expectations. for example it indicates that the schizophrenic brain is more cost-efficient and also more clustered (which contradicts [7, 54]). These unexpected results most likely come down to peculiarities with the unthresholded ParCorr matrices, which perhaps don't reflect the properties of brain networks commonly used in neuroscience. This isn't to say our networks are flawed; the meaning of 'clustered' and 'cost efficient' could just represent something different for our networks. It could also be down to the data itself, although then we'd expect abnormal results for all batches.

4.5.2. (Backboned) ParCorr

Since backboned matrices have undergone edge deletion, metric interpretations may now have relevance to network resilience (robustness to edge removal). However we do not pursue this point further, as interpretations are unclear in the context of correlation-based edges (which model a functional relation rather than a physical phenomenon).

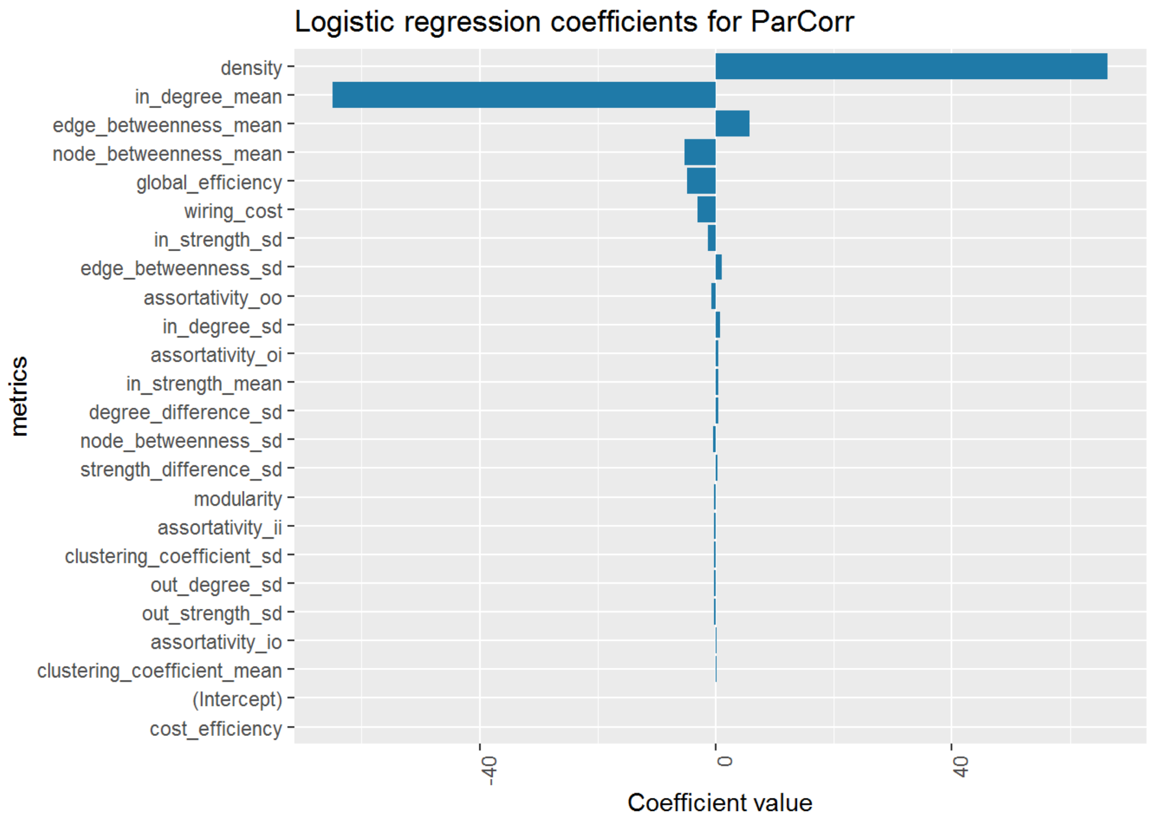

Figure 4.14: logistic regression coefficients. Positive sign means model associates high metric values with schizophrenia. Negative means low values associate with schizophrenia.

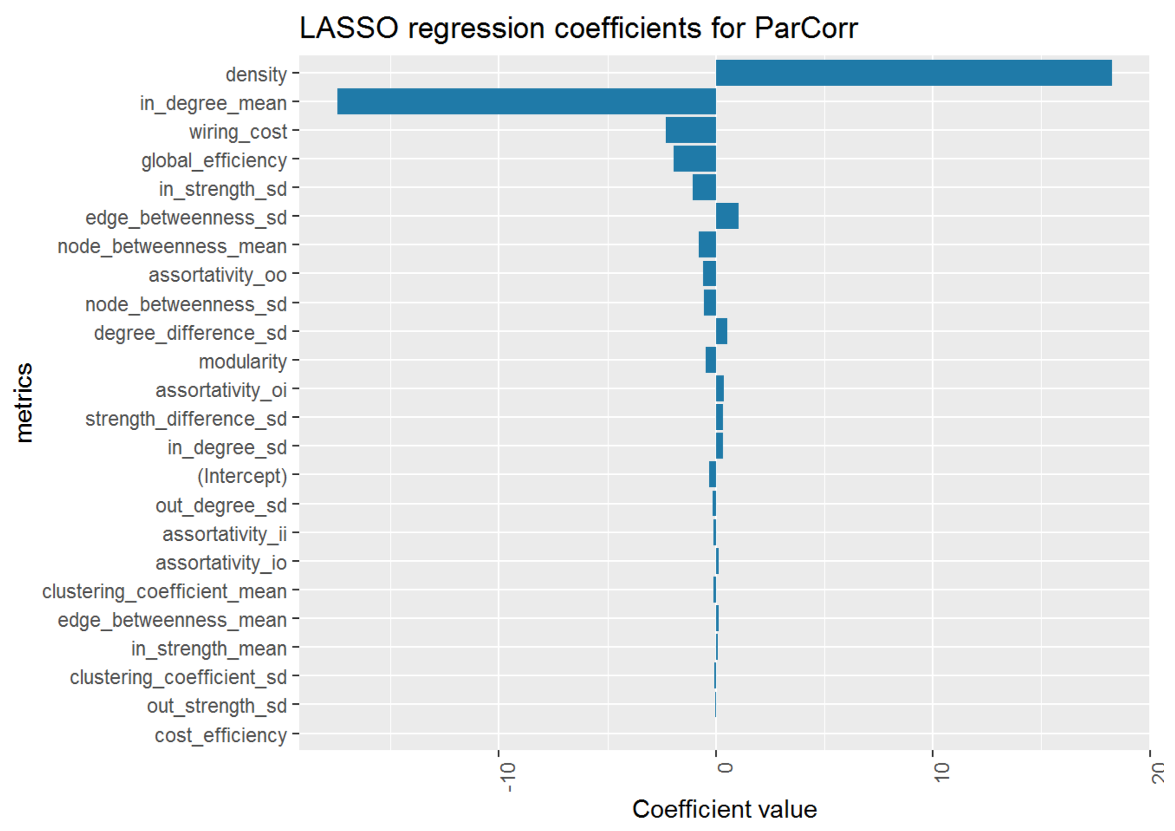

Figure 4.15: LASSO regression coefficients, which have undergone shrinkage.

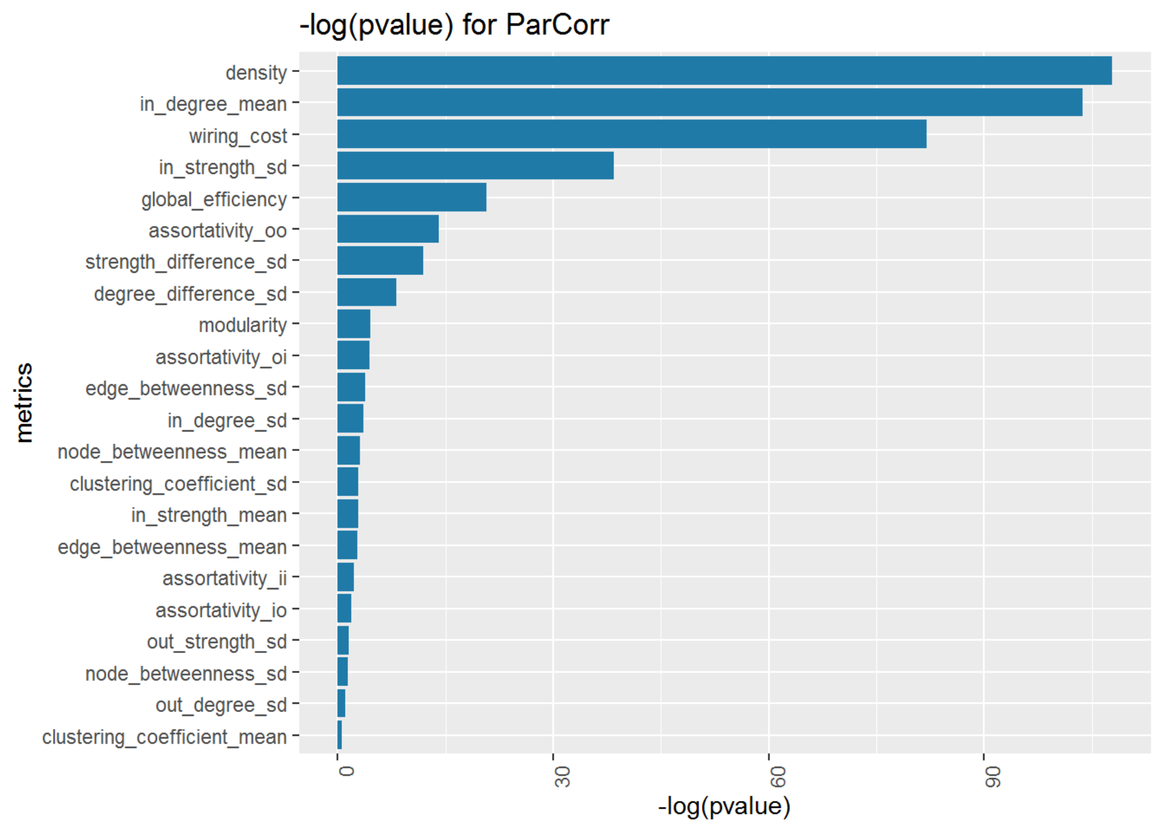

Figure 4.16: logistic regression transformed p-values, as a heuristic importance measure.

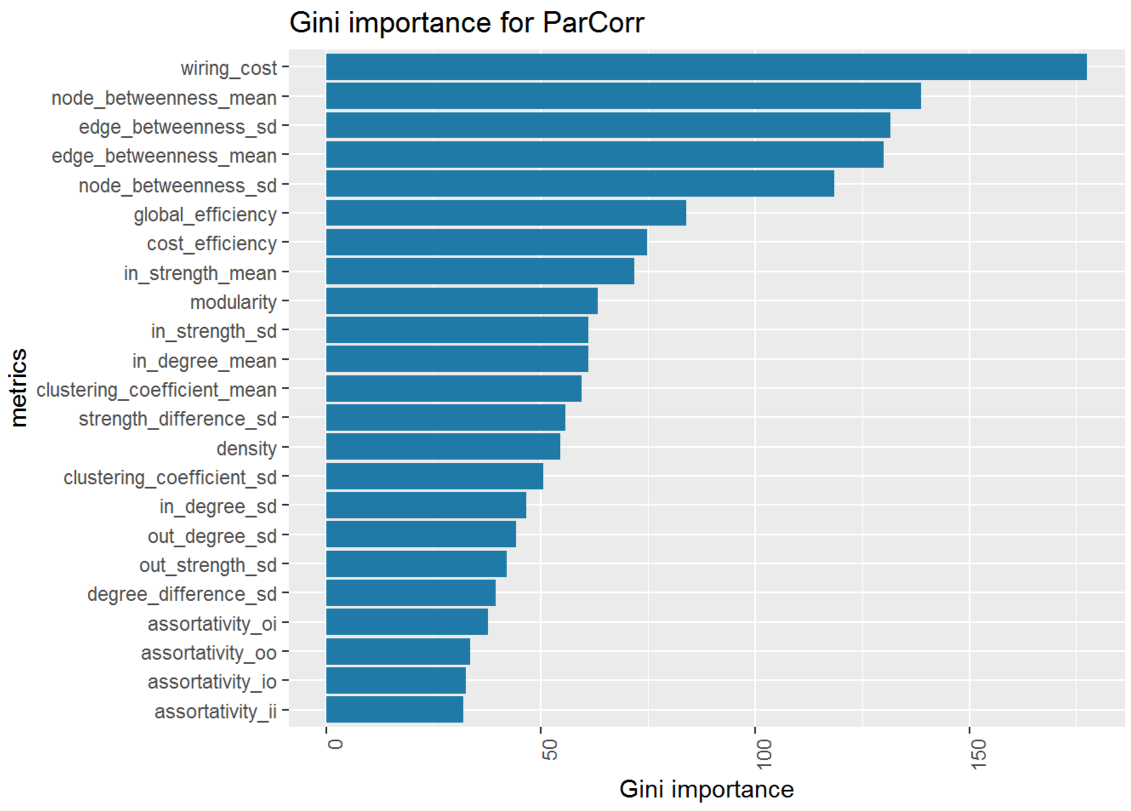

Figure 4.17: Random forest Gini importance, which quantifies group separating ability.

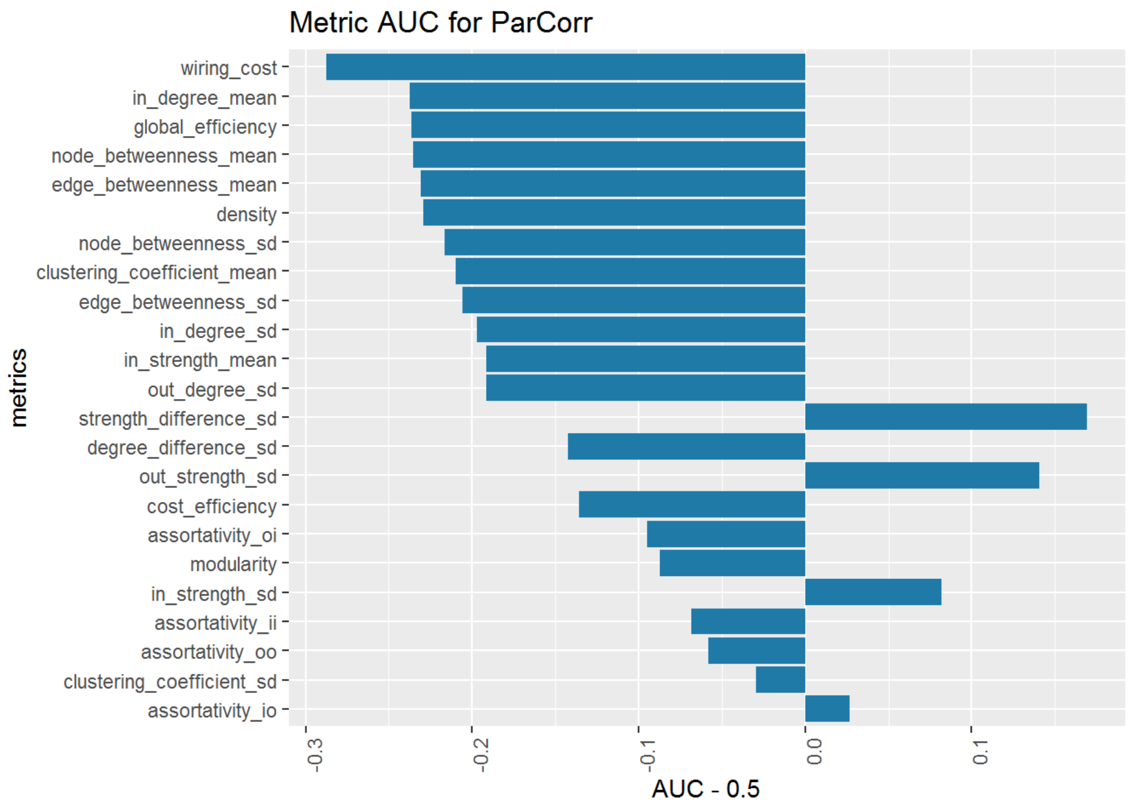

Figure 4.18: Rescaled AUC using individual variables as scores, resulting in a measure of group separation. Positive sign means schizophrenia group has higher values in this metric. Negative sign means schizophrenia group has lower values.

We first look at regression-based importance. Figures 4.14 and 4.15 show that high density and low mean in-degree are characteristic of schizophrenia (compared to healthy). Figure 4.16 reinforces this. This is in perfect agreement with the hypothesis of weaker functional connectivity but greater connection diversity in the schizophrenic brain [7, 19].

But although we now have agreement with authoritative results, we also have disagreement within our own study! Recall from table 4.1 that schizophrenia networks were in fact **less** dense than healthy networks. This is reinforced by the AUC (figure 4.18) which also says that schizophrenia networks are less dense. Clearly, the data-derived AUC is again a safer bet for making insights about the data than regression model coefficients.

The separability importance measures (figures 4.17 and 4.18) agree with each other for the most part. In both, wiring cost is the most important metric. However, figure 4.18 shows that low values of wiring cost indicate schizophrenia, and this is additionally supported by both regression models, with significant p-value 4.16. This is more evidence contradicting the results in [8], which says the schizophrenic brain has increased connection distance.

4.5.3. **PMIME**

In the regression models (figures 4.19, 4.20, 4.21) we see density again ranked as an important metric. This seems counterintuitive given table 4.1 which (for PMIME) shows near equal densities for healthy and schizophrenic groups. Figures 4.22 and 4.23 agree with table 4.1 and show that density does not have high separating power as a variable. So yet again, regression coefficients for density and facts about the raw data are in conflict. Evidently the regression coefficients need to be further scrutinised in order to explain these observations.

A more encouraging result is that regression models for (backboned) ParCorr (figures 4.14 and 4.15) and regression models for PMIME (figures 4.19 and 4.20) are in strong agreement with each other (i.e. both support the theories of [7, 19] about weaker functional connectivity and greater connection diversity for schizophrenia). These two batches were engineered by us to be comparable (density-wise) and so it is reassuring to see parallels between the two for variable importance.

However, (backboned) ParCorr and PMIME networks do not agree for random forest and AUC plots (figures 4.22 and 4.23). The significance of this is not well-understood by us at this stage, but it may reveal vital information about the similarities and differences between the two batches of networks.

In figures 4.22 and 4.23 we see the emergence of many standard deviation-based metrics, that were not particularly prominent for other batches. This means that for PMIME, differences between schizophrenic and healthy brains are often down to metric variation across nodes, rather than the metric values themselves. In this way, algorithms can still obtain reasonable classification accuracy despite network density being almost the same between the two groups. We also see the emergence of modularity as an important metric for the first time in this study. Schizophrenia patients have increased modularity, which contradicts authoritative results [55].

It is important to notice the relatively low magnitude of AUC in figure 4.23 compared to the AUC for other batches (figures 4.13 and 4.18).

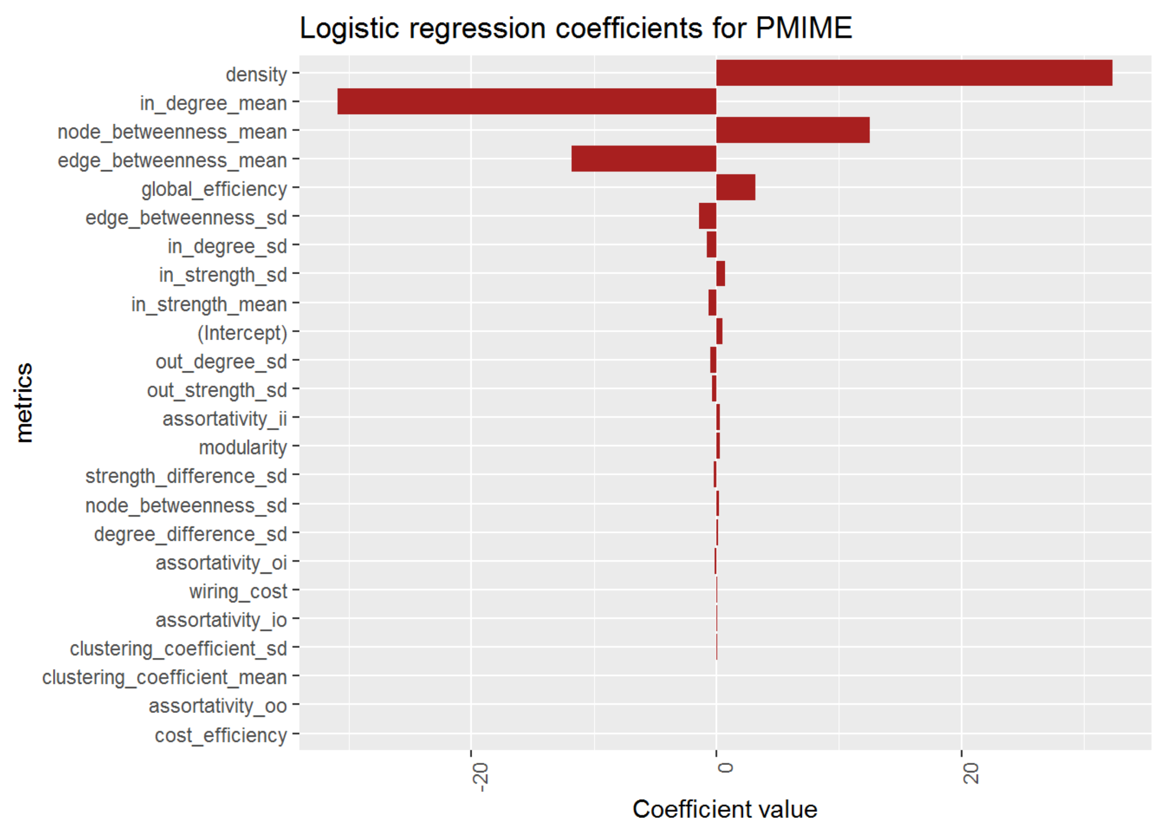

Figure 4.19: logistic regression coefficients. Positive sign means model associates high metric values with schizophrenia. Negative means low values associate with schizophrenia.

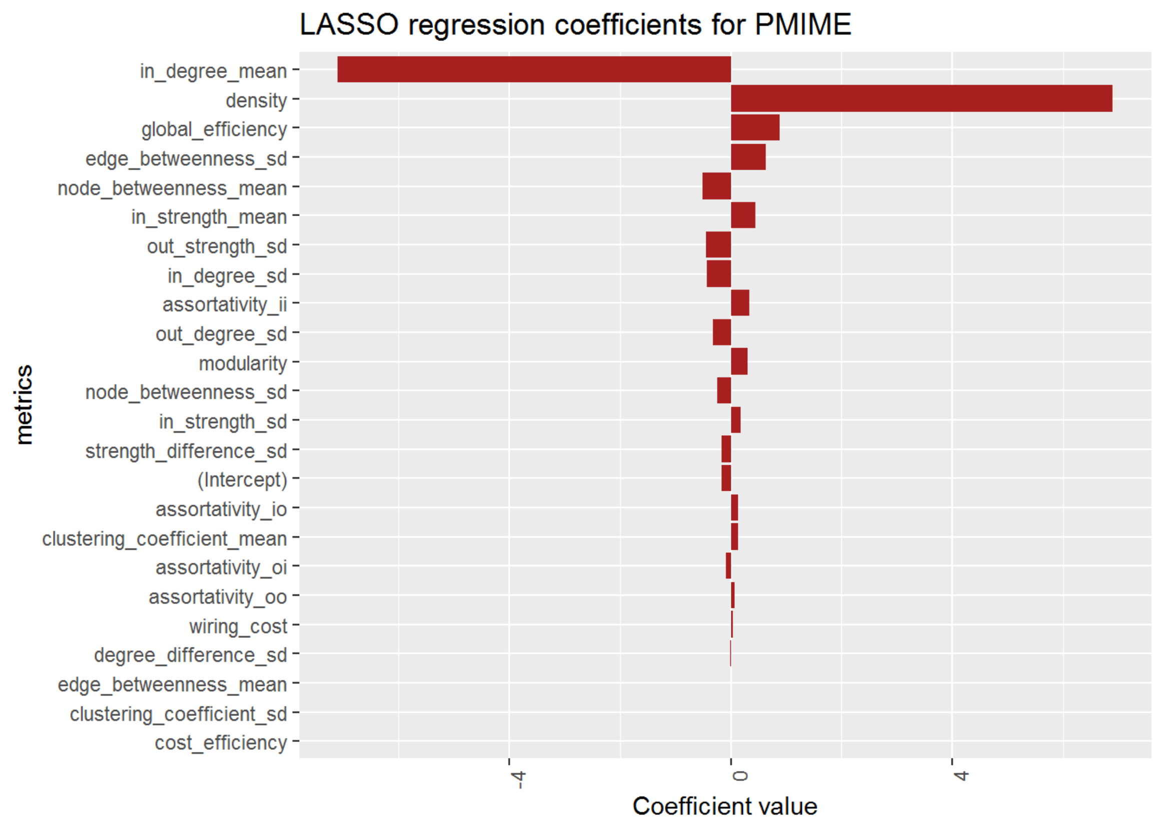

Figure 4.20: LASSO regression coefficients, which have undergone shrinkage.

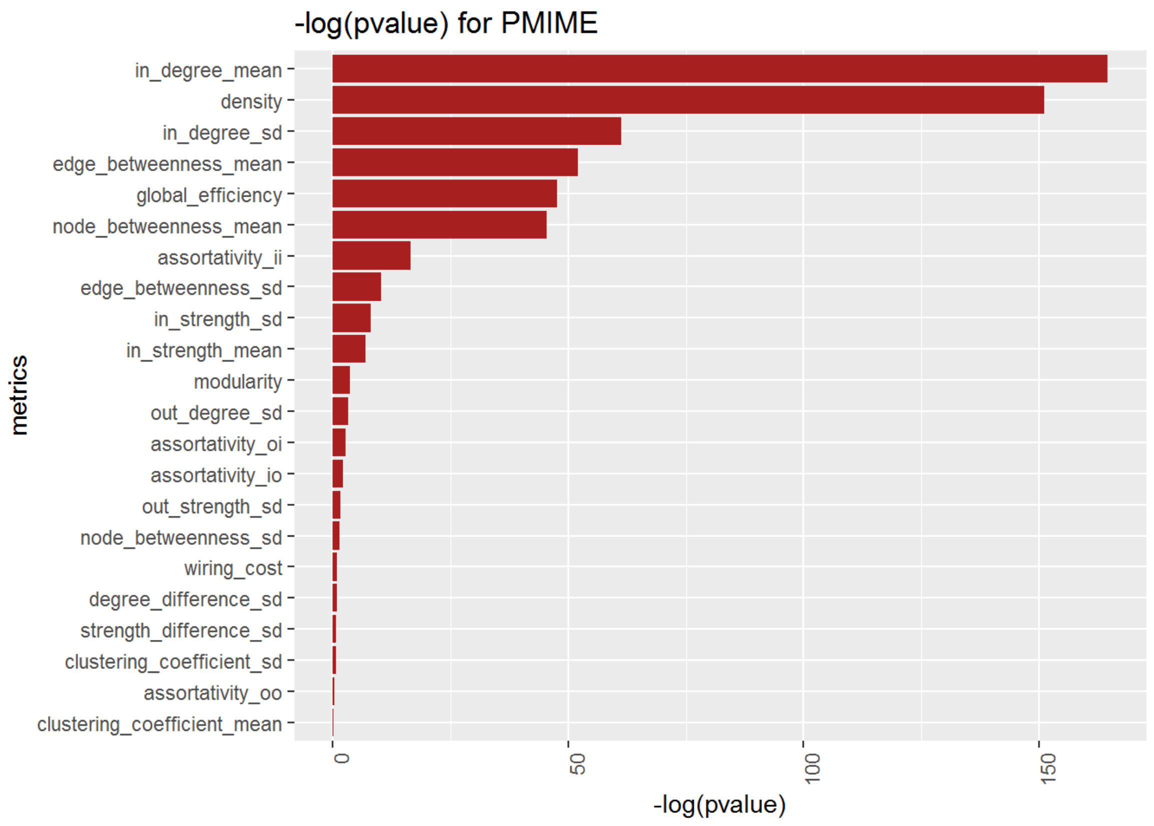

Figure 4.21: logistic regression transformed p-values, as a heuristic importance measure.

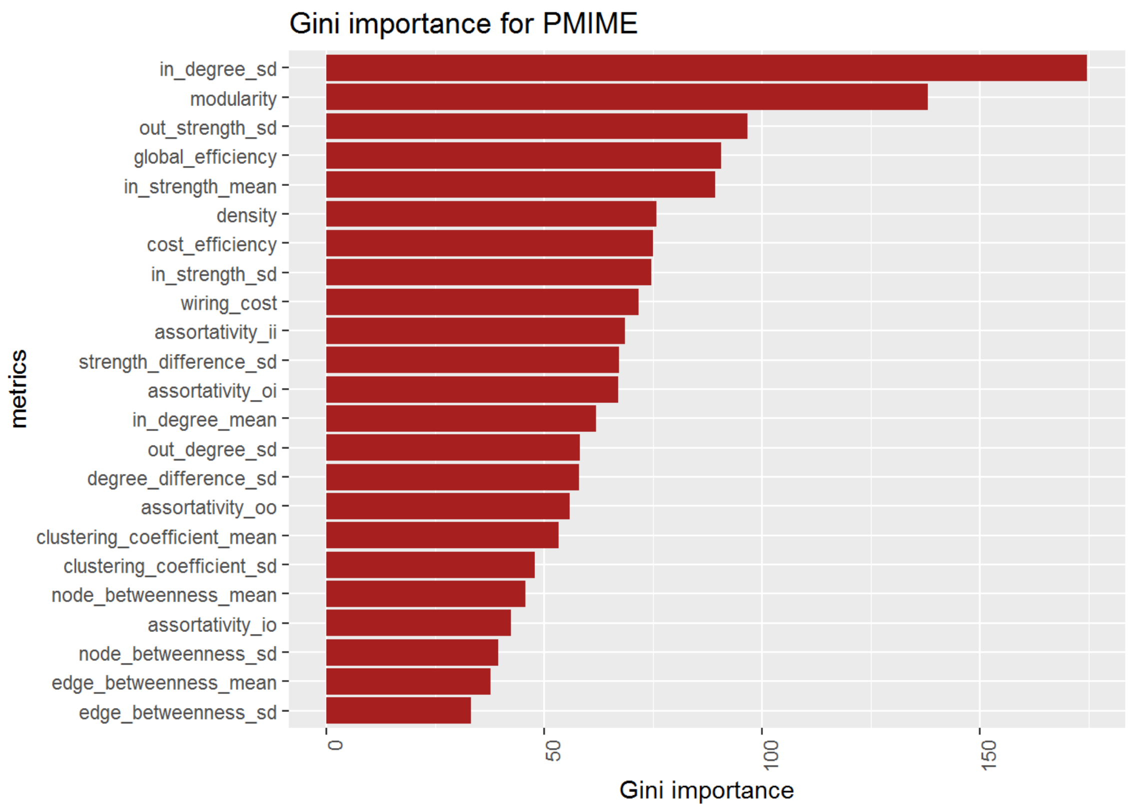

Figure 4.22: Random forest Gini importance, which quantifies group separating ability.

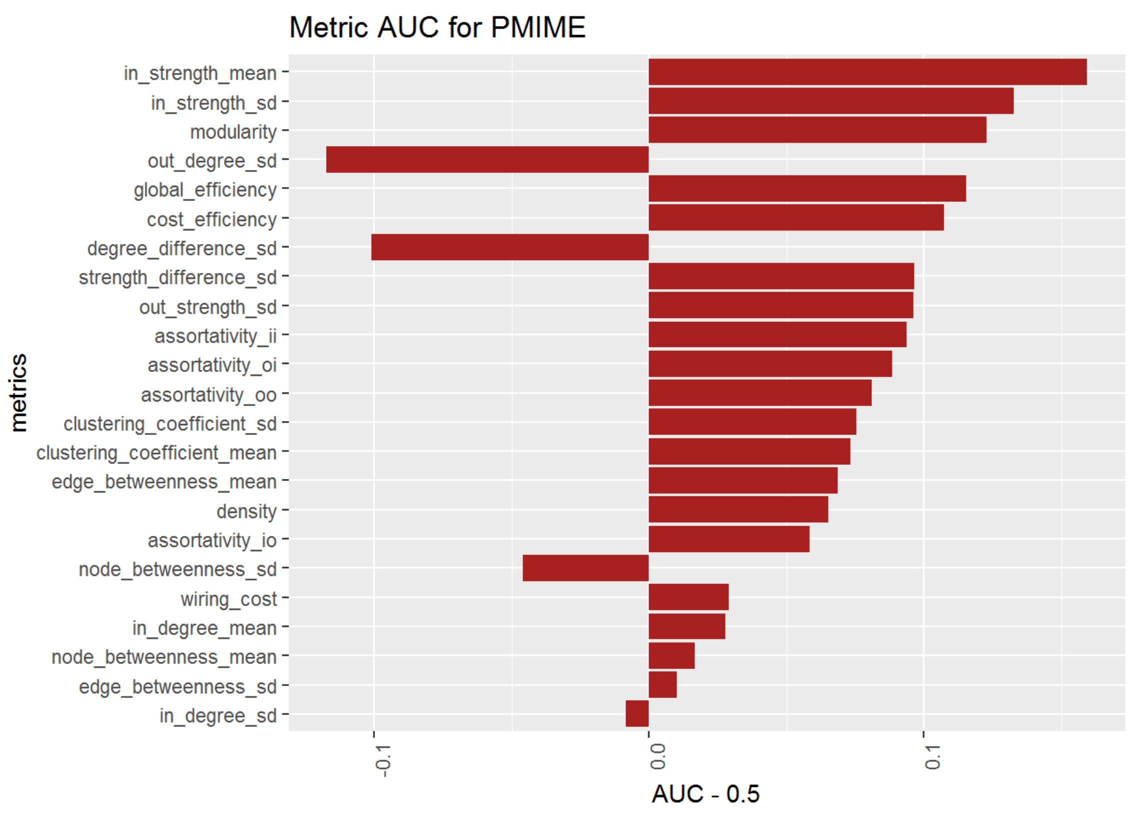

Figure 4.23: Rescaled AUC using individual variables as scores, resulting in a measure of group separation. Positive sign means schizophrenia group has higher values in this metric. Negative sign means schizophrenia group has lower values.

4.5.4. Summary and combined importance plots

We summarise the results for each batch of matrices using combined variable importance plots (figures 4.24, 4.25, 4.26) after scaling the 5 measures. We should beware of over-interpreting these plots, as there are many inter-dependencies between the 5 measures. Their main purpose is to serve as a visual aid.

From the previous subsections it is also clear that we should take variable importance results with a pinch of salt - we saw in section 4.5.1 that they do not always behave how we might expect. Moreover, results from algorithms (regression coefficients in particular) and results from raw data (AUC) were often in total disagreement. It is more reliable to look at scatter plots such as those in 4.2, which we can interpret with more confidence.

But suppose now we acknowledge our variable importance results. Figure 4.24 reveals 4 principal network metrics - global efficiency, in-strength mean, edge betweenness mean, and node betweenness mean. This is very different to the other two batches (figures 4.25 and 4.26) which, with the exception of wiring cost, are mostly in agreement. These two batches appear to both agree with the authoritative result that schizophrenic brains have greater connection diversity but weaker functional connectivity [7, 19].

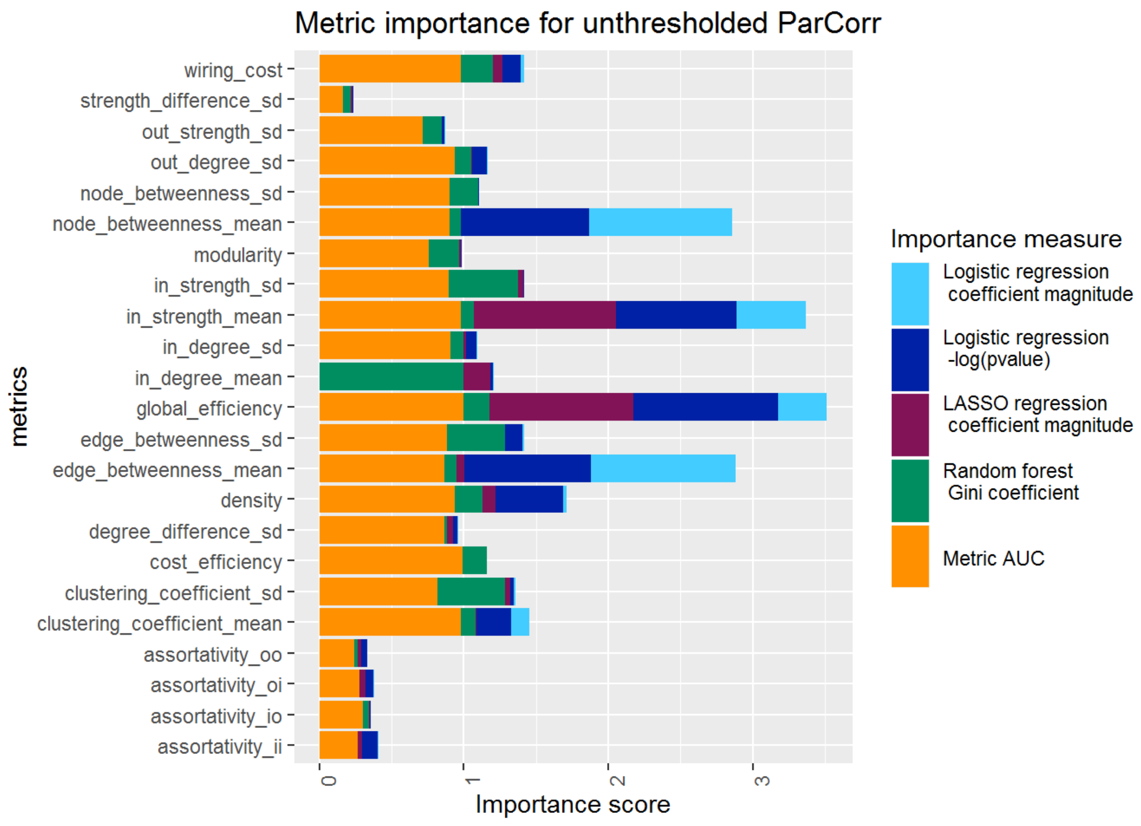

Figure 4.24: combined variable importance for unthresholded ParCorr.

It is tempting to conclude that better agreement with well-established results means a better measure. However this is not necessarily the case. As we warned at the end of section 4.5.1, common results in neuroscience may be based on entirely different network constructions. It is highly likely that (backboned) ParCorr and PMIME only agree better with common results because the networks resemble the ones used to derive said results. Unthresholded ParCorr matrices have densities of close to 100%, which (to our knowledge) is atypical in the field. Hence it is no surprise that we contradict a lot of previous research findings. With drastic change in density comes drastic change in network metrics [37] and so it is easy to imagine that metrics characteristic of schizophrenia will also change.

The fact that backboned ParCorr gives results closer to PMIME than unthresholded ParCorr seems to indicate that network density dictates network metrics more than choice of causality measure. This is of interest to our discussion in section 3.2.1. Further experimentation over different densities would be required to test this claim. If true, then in a classification context, PMIME is unnecessary as backboning ParCorr matrices is a much faster process that returns the same information. That said, this becomes less relevant if unthresholded ParCorr matrices are conclusively found to outperform both backboned ParCorr and PMIME.

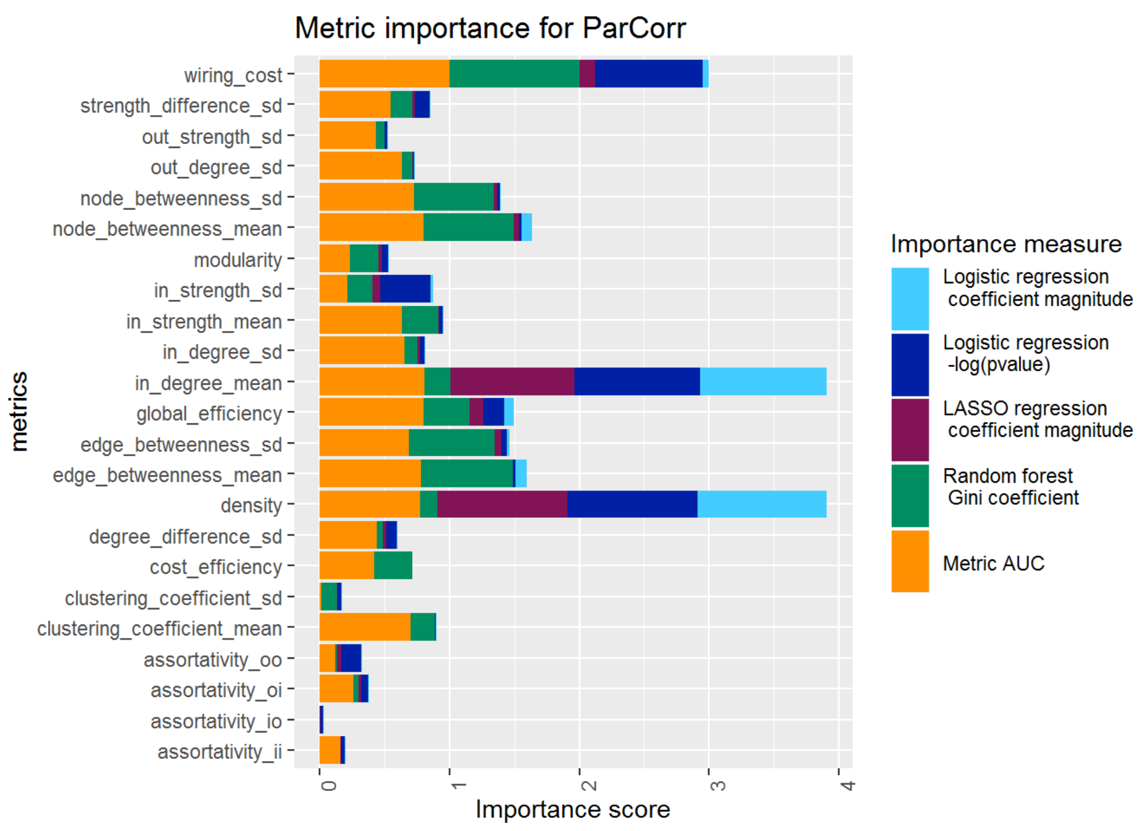

Figure 4.25: combined variable importance for (backboned) ParCorr.

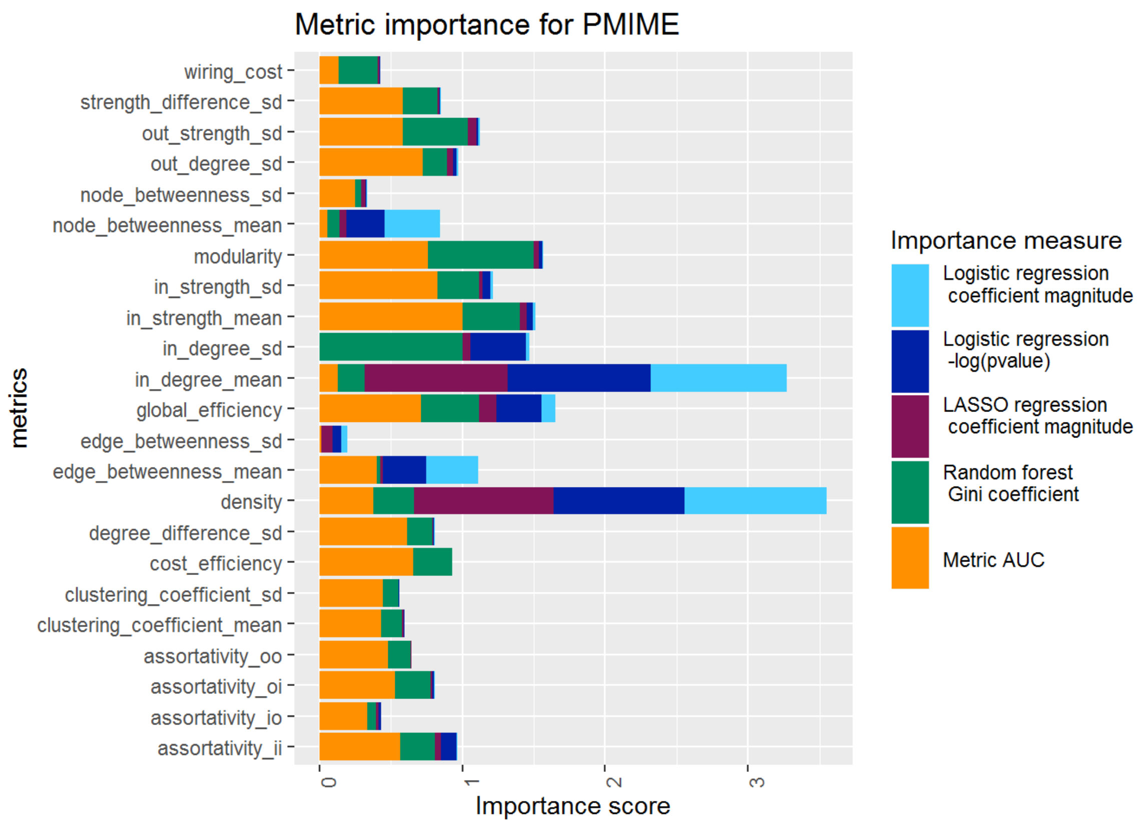

Figure 4.26: combined variable importance for PMIME.

5. Conclusions

This chapter concludes the report. We summarise our findings and discuss the weaknesses associated with our study. We finish by proposing several avenues for future research.

5.1. Summary of findings

We began by considering two causality measures based on conditional mutual information - PMIME and PCMCI (with CMI-based test). However, the latter was found to be far too computationally taxing to be practically useful for our EEG data. If anything, this proved the supremacy of PMIME as a mutual information-based measure that remains efficient on high-dimensional data.

We instead chose to consider the linear measure, PCMCI (ParCorr), alongside PMIME. We discuss differences between the two in section 3.2.1. Unthresholded ParCorr matrices had edge densities close to 100%, and so we used a matrix 'backboning' technique to bring density down to the same level as PMIME (around 5%), for better comparability.

However, it turned out that the best machine learning classification performance was decisively on the unthresholded batch of ParCorr matrices. This was achieved by the single-hidden-layer neural network, with classification accuracy 93%.

Our hypothesis for these results is that the sheer amount of information contained in the high-density unthresholded matrices provided more comprehensive network metric data for machine learning. This agrees with findings that weak edges are instrumental in classifying schizophrenia [19]. The practical implications of this are large, as ParCorr matrices are several times faster to compute than PMIME matrices. Knowing that networks based on linear causality measures are sufficient for accurate patient diagnosis would be valuable information to physicians, for example.

This is perhaps our most important finding: we have demonstrated that even if the causality measure produces more accurate causal networks appropriate for other studies, this means little in the context of a machine learning study.

We also looked at which network metrics were most indicative of a schizophrenic brain, for each batch of networks. Results varied rather wildly. In particular, regression coefficients often gave results that contradicted known facts about the data. This led us to conclude that simple scatter plots (see section 4.2) and data-derived metrics such as AUC are more useful in extracting data insights.

The low density batches of networks appeared to better agree with well-established results, such as weaker functional connectivity but greater connection diversity in schizophrenic brains. However, we pointed out that this could simply be due to most authoritative results working with low density networks. Metrics are affected by density [37], and for high densities, metrics may not even hold the same meanings. Thus we should not be discouraged by the atypical results for variable importance in unthresholded ParCorr networks.

Perhaps the most useful observation was that the two batches with the same average density had similar metrics selected as important, despite being constructed from different causality measures. This seems to indicate that network density dictates the set of important metrics, rather than choice of causality measure.

5.2. Weaknesses and areas for improvement

Although we reported some interesting findings, it is important that we do not overlook the weaknesses behind our results.

The low sample size of patients is arguably the biggest weakness, as it affects all parts of our study. In particular, our machine learning results will suffer from being overfit on this small sample of patients; the 93% classification accuracy could quite easily be a consequence of our algorithms exploiting some feature found only in our dataset. For example, maybe all schizophrenia patients also suffered from some form of anxiety. By dividing the patients' time series into many intervals and treating them as different instances of schizophrenia, we are merely fooling our algorithm into training on more examples. We can thus expect that our accuracies are highly optimistic, despite our best efforts via cross-validation to simulate performance on unseen data.

Another major confounding factor we have not given much attention is the effect of patient age and gender. More data would be required to assess the extent of this effect. One possible approach to tackle this issue would be to include fixed/random effects [42] in our regression models. In simple terms, this gives the model the extra capacity it needs to explain additional variables like age and gender, removing their bias from the rest of the analysis.

A definite flaw in our procedure that was not yet mentioned is the rather arbitrary choice of network metrics. We even found evidence of linear relations between metrics (figure 4.2) which we did not have time to address in this study. The resulting variable collinearity is more than capable of hampering classification model performance. To improve our procedure, we should look at performing some form of variable selection from a larger set of possible network metrics. One such example of a scheme can be found in [15].

The validity of backboning matrices to obtain equal density between ParCorr and PMIME batches is also questionable. This is because the two causality measures result in networks with potentially different interpretations (e.g. the meaning of an edge). Therefore it is futile to attempt to directly compare the networks by equalising densities, as they are not directly comparable to begin with. It may have been more rational to backbone both ParCorr and PMIME using the same significance threshold. However, this is not to say our choice of backboning didn't result in any useful insights.

Lastly we mention a more general criticism of interpreting EEG studies. EEG forces downsampling, and studies show that this leads to networks which are unlikely to be truly reflective of brain activity [13]. This should serve more as a precautionary note than a problem with our analysis.

5.3. Future studies

Throughout the report, a number of points were raised for potential directions of future study. In addition to the areas for improvement described above, we briefly mention what we believe are the next steps to be taken from this study.

Finding a sweet spot between ParCorr network density and classification performance holds priority. In this study we simply found that networks with near 100% edge density gave better classification performance than those with 5% density. Testing intermediate densities could reinforce or completely overturn our hypothesis that inclusion of the maximal amount of edge data led to the best results.

Trying a larger range of causality measures would also solidify our understanding of the results presented here. This could involve experimenting on whether directed measures improve classification performance. We could test the very oldest causality measure (Granger causality) or some of the very latest: [56] presents a measure based on convolutional neural networks that harnesses the power of deep learning.

References

[1] Towlson EK, Vértes PE, Ahnert SE, Schafer WR, Bullmore ET. The rich club of the c. elegans neuronal connectome. *The Journal of Neuroscience*, 33(15):6380–6387, 2013. Available from: https://doi.org/10.1523/JNEUROSCI.3784-12.2013.

[2] Wan X, Crüts B, Jensen HJ. The causal inference of cortical neural networks during music improvisations. *PLoS ONE*, 9(12):e112776, 2014. Available from: https://doi.org/10.1371/journal.pone.0112776.

[3] Dolan D, Jensen HJ, Mediano PAM, Molina-Solana M, Rajpal H, Rosas F, et al. The improvisational state of mind: A multidisciplinary study of an improvisatory approach to classical music repertoire performance. *Frontiers in Psychology*, 9:1341, 2018. Available from: https://doi.org/10.3389/fpsyg.2018.01341.

[4] Kandel ER. Disorders of thought and volition: Schizophrenia. In *Principles of neural science*, pages 1188–1208. McGraw Hill, 2000.

[5] Knyazeva MG, Jalili M. Eeg-based functional networks in schizophrenia. *Computers in Biology and Medicine*, 41(12):1178–1186, 2011. Available from: https://doi.org/10.1016/j.compbiomed.2011.05.004.

[6] Friston K, Brown HR, Siemerkus J, Stephan KE. The dysconnection hypothesis (2016). *Schizophrenia Research*, 176(2-3):83–94, 2016. Available from: https://doi.org/10.1016/j.schres.2016.07.014.

[7] Lynall ME, Bassett DS, Kerwin R, McKenna PJ, Kitzbichler M, Muller U, et al. Functional connectivity and brain networks in schizophrenia. *The Journal of Neuroscience*, 30(28):9477–9487, 2010. Available from: https://doi.org/10.1523/JNEUROSCI.0333-10.2010.

[8] Bassett DS, Bullmore ET, Verchinski BA, Mattay VS, Weinberger DR, Meyer-Lindenberg A. Hierarchical organization of human cortical networks in health and schizophrenia. *The Journal of Neuroscience*, 28(37):9239–9248, 2008. Available from: https://doi.org/10.1523/JNEUROSCI.1929-08.2008.

[9] Sporns O, Bullmore ET. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10:186–198, 2009. Available from: https://doi.org/10.1038/nrn2618.

[10] Kugiumtzis D. Direct-coupling information measure from nonuniform embedding. *Physical Review E*, 87(6):062918, 2013. Available from: https://doi.org/10.1103/PhysRevE.87.062918.

[11] Runge J, Nowack P, Kretschmer M, Flaxman S, Sejdinovic D. Detecting causal associations in large nonlinear time series datasets, 2017. Available from: https://arxiv.org/abs/1702.07007 [Accessed 14 Jan 2019].

[12] Razak FA. *Mutual information based measures on complex interdependent networks of neuro data sets*. PhD thesis, Imperial College London, 2013. Available from: http://hdl.handle.net/10044/1/11579 [Accessed 14 Jan 2019].

[13] Zanin M, Papo D, Sousa PA, Menasalvas E, Nicchi A, Kubik E, et al. Combining complex networks and data mining: why and how. *Physics Reports*, 635:1–44, 2016. Available from: https://doi.org/10.1016/j.physrep.2016.04.005.

[14] Pollonini L, Patidar U, Situ N, Rezaie R, Papanicolaou AC, Zouridakis G. Functional connectivity networks in the autistic and healthy brain assessed using granger causality. In *32nd Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 1730–1733, Buenos Aires, Argentina, 31 August - 4 September 2010. IEEE. Available from: https://doi.org/10.1109/IEMBS.2010.5626702.

[15] Quintero-Zea A, López JD, Smith K, Trujillo N, Parra MA, Escudero J. Phenotyping ex-combatants from eeg scalp connectivity. *IEEE Access*, 6:55090–55098, 2018. Available from: https://doi.org/10.1109/ACCESS.2018.2872765.

[16] Liao W, Ding J, Marinazzo D, Xua Q, Wang Z, Yuan C et al. Small-world directed networks in the human brain: Multivariate granger causality analysis of resting-state fmri. *NeuroImage*, 54(4):2683–2694, 2011. Available from: https://doi.org/10.1016/j.neuroimage.2010.11.007.

[17] Koutlis C, Kugiumtzis D. Discrimination of coupling structures using causality networks from multivariate time series. *Chaos*, 26:093120, 2016. Available from: https://doi.org/10.1063/1.4963175.

[18] Papana A, Kyrtsou C, Kugiumtzis D, Diks C. Financial networks based on granger causality: A case study. *Physica A: Statistical Mechanics and its Applications*, 482:65–73, 2017. Available from: https://doi.org/10.1016/j.physa.2017.04.046.

[19] Bassett DS, Nelson BG, Mueller BA, Camchong J, Lim KO. Altered resting state complexity in schizophrenia. *NeuroImage*, 59(3):2196–2207, 2012. Available from: https://doi.org/10.1016/j.neuroimage.2011.10.002.

[20] Preston GA, Weinberger DR. Intermediate phenotypes in schizophrenia: a selective review. *Dialogues in Clinical Neuroscience*, 7(2):165–179, 2005.

[21] Walden A. Time series lecture notes. Available from: http://wwwf.imperial.ac.uk/~atw/MS8.html [Accessed 14 Jan 2019].

[22] Shannon CE. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. Available from: https://doi.org/10.1002/j.1538-7305.1948.tb01338.x.

[23] Cover TM, Thomas JA. *Elements of Information Theory*. Wiley, Hoboken, NJ, Second edition, 2006.

[24] Wan X. *Time series causality analysis and EEG data analysis on music improvisation.* PhD thesis, Imperial College London, 2014. Available from: http://hdl.handle.net/10044/1/23956 [Accessed 14 Jan 2019].

[25] Runge J. *Detecting and quantifying causality from time series of complex systems.* PhD thesis, Humboldt University of Berlin, 2014. Available from: https://doi.org/10.18452/17017.

[26] Granger CWJ. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969. Available from: https://doi.org/10.2307/1912791.

[27] Tibshirani R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

[28] Kraskov A, Stögbauer H, Grassberger P. Estimating mutual information. *Physical Review E*, 69(6):066138, 2004. Available from: https://doi.org/10.1103/PhysRevE.69.066138.

[29] Runge J. Conditional independence testing based on a nearest-neighbor estimator of conditional mutual information. In Storkey A, Perez-Cruz F, editor, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84, pages 938–947, Lanzarote, Spain, 9-11 April 2018. PMLR. Available from: http://proceedings.mlr.press/v84/runge18a.html [Accessed 7th June 2019].

[30] Spirtes P, Glymour C. A fast algorithm for discovering sparse causal graphs. *Social Science Computer Review*, 9(1):62–72, 1991. Available from: https://doi.org/10.1177/089443939100900106.

[31] Bovet A, Makse HA. Influence of fake news in twitter during the 2016 us presidential election. *Nature Communications*, 10:7, 2019. Available from: https://doi.org/10.1038/s41467-018-07761-2.

[32] Newman MEJ. *Networks: An Introduction.* Oxford University Press, New York, NY, 2010.

[33] Rubinov M, Sporns O. Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, 52(3):1059–1069, 2010. Available from: https://doi.org/10.1016/j.neuroimage.2009.10.003.

[34] Achard S, Bullmore E. Efficiency and cost of economical brain functional networks. *PLoS Computational Biology*, 3(2):e17, 2007. Available from: https://doi.org/10.1371/journal.pcbi.0030017.

[35] Watts DJ, Strogatz SH. Collective dynamics of ‘small-world’ networks. *Nature*, 393:440–442, 1998. Available from: https://doi.org/10.1038/30918.

[36] Bullmore ET, Bassett DS. Small-world brain networks revisited. *The Neuroscientist*, 23(5):499–516, 2017. Available from: https://doi.org/10.1177/1073858416667720.

[37] Joudaki A, Salehi N, Jalili M, Knyazeva MG. Eeg-based functional brain networks: Does the network size matter? *PLoS ONE*, 7(4):e35673, 2012. Available from: https://doi.org/10.1371/journal.pone.0035673.

[38] Jensen HJ. Probability and statistics in complex systems, introduction to. In Meyers R, editor, *Encyclopedia of Complexity and Systems Science*. Springer, New York, NY, 2009. Available from: https://doi.org/10.1007/978-0-387-30440-3_419.

[39] Battiston F, Nicosia V, Chavez M, Latora V. Multilayer motif analysis of brain networks. *Chaos*, 27:047404, 2017. Available from: https://doi.org/10.1063/1.4979282.

[40] Buldú JM, Porter MA. Frequency-based brain networks: From a multiplex framework to a full multilayer description. *Network Neuroscience*, 2(4):418–441, 2018. Available from: https://doi.org/10.1162/netn_a_00033.

[41] Domingos P. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012. Available from: https://doi.org/10.1145/2347736.2347755.

[42] McCullagh P, Nelder JA. *Generalized Linear Models*. Chapman and Hall, Boca Raton, FL, Second edition, 1989.

[43] Steinwart I, Hush D, Scovel C. Learning from dependent observations. *Journal of Multivariate Analysis*, 100(1):175–194, 2009. Available from: https://doi.org/10.1016/j.jmva.2008.04.001.

[44] Caruana R, Niculescu-Mizil A. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 161–168, Pittsburgh, PA, 25-29 June 2006. ACM. Available from: https://doi.org/10.1145/1143844.1143865.

[45] Platt JC. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[46] Chaumon M, Bishop DV, Busch NA. A practical guide to the selection of independent components of the electroencephalogram for artifact correction. *Journal of Neuroscience Methods*, 250:47–63, 2015. Available from: https://doi.org/10.1016/j.jneumeth.2015.02.025.

[47] Gramfort A, Luessi M, Larson E, Engemann DA, Strohmeier D, Brodbeck C, et al. Mne software for processing meg and eeg data. *NeuroImage*, 86:446–460, 2014. Available from: https://doi.org/10.1016/j.neuroimage.2013.10.027.

[48] Gramfort A, Luessi M, Larson E, Engemann DA, Strohmeier D, Brodbeck C, et al. Meg and eeg data analysis with mne-python. *Frontiers in Neuroscience*, 7(267), 2013. Available from: https://doi.org/10.3389/fnins.2013.00267.

[49] Kugiumtzis D, Vlachos I. Nonuniform state-space reconstruction and coupling detection. *Physical Review E*, 82(1):016207, 2010. Available from: https://doi.org/10.1103/PhysRevE.82.016207.

[50] Runge J, Petoukhov V, Donges JF, Hlinka J, Jajcay N, Vejmelka M, et al. Identifying causal gateways and mediators in complex spatio-temporal systems. *Nature Communications*, 6:8502, 2015. Available from: https://doi.org/10.1038/ncomms9502.

[51] Runge J. Quantifying information transfer and mediation along causal pathways in complex systems. *Physical Review E*, 92(6):62829, 2015. Available from: https://doi.org/10.1103/PhysRevE.92.062829.

[52] Runge J, Heitzig J, Petoukhov V, Kurths J. Escaping the curse of dimensionality in estimating multivariate transfer entropy. *Physical Review Letters*, 108(25):258701, 2012. Available from: https://doi.org/10.1103/PhysRevLett.108.258701.

[53] Coscia M, Neffke F. Network backboning with noisy data. In *33rd International Conference on Data Engineering*, pages 425–436, San Diego, CA, 19-22 April 2017. IEEE. Available from: https://doi.org/10.1109/ICDE.2017.100.

[54] Liu Y, Liang M, Zhou Y, He Y, Hao Y, Song M, et al. Disrupted small-world networks in schizophrenia. *Brain*, 131(4):945–961, 2008. Available from: https://doi.org/10.1093/brain/awn018.

[55] Alexander-Bloch AF, Gogtay N, Meunier D, Birn R, Clasen L, Lalonde F, et al. Disrupted modularity and local connectivity of brain functional networks in childhood-onset schizophrenia. *Frontiers in Systems Neuroscience*, 4(147), 2010. Available from: https://doi.org/10.3389/fnsys.2010.00147.

[56] Nauta M, Bucur D, Seifert C. Causal discovery with attention-based convolutional neural networks. *Machine Learning and Knowledge Extraction*, 1(1):312–340, 2019. Available from: https://doi.org/10.3390/make1010019.

Appendices

Here we showcase excerpts from the code used for our analysis.

A. Network construction

For both ParCorr and PMIME, we first write code to compute and save a causality matrix given a raw data slice, and then write a shell script for submission to the computing cluster, that will automate this process.

A.1. ParCorr Python code e.g. 'bashrun6140.py'

PCMCI functions are available from https://github.com/jakobrunge/tigramite.

```python
import numpy as np
import tigramite
from tigramite import data_processing as pp
from tigramite import plotting as tp
from tigramite.pcmci import PCMCI
from tigramite.independence_tests import ParCorr, GPDC, CMIknn, CMIsymb
from tigramite.models import LinearMediation, Prediction
import cython
import sklearn
import sys

pat = '6140' #patient number
raw = np.load('pat'+str(pat)+'.npy') #load raw patient .npy data
data = raw[:,:]
data = data.swapaxes(0, 1) #flip shape of array
var_names = np.load('ch_names'+str(pat)+'.npy').tolist() #EEG channel
    names for this patient

#function to retrieve PCMCI (ParCorr) matrices given raw data and
    maximum lag:
def parcorr_fn(dataframe,var_names,tau_max):
    parcorr = ParCorr(significance='analytic')
    pcmci = PCMCI(dataframe=dataframe, cond_ind_test=parcorr, var_names
        =var_names, verbosity=0)
    results = pcmci.run_pcmci(tau_max=tau_max, pc_alpha=None)
    val_matrix = results['val_matrix'] #weighted directed matrix
    q_matrix = pcmci.get_corrected_pvalues(p_matrix=results['p_matrix'
        ], fdr_method='fdr_bh')
    link_matrix = pcmci._return_significant_parents(pq_matrix=q_matrix,
         val_matrix=val_matrix, alpha_level=0.01)['link_matrix'] #
        boolean matrix of significance test results
    return(val_matrix, link_matrix)
```

```
#function to convert to 2D matrix of significant edges:
def matconv_fn(val_matrix, link_matrix):
    dims = np.shape(val_matrix)[1]
    combined_matrix = np.multiply(val_matrix, link_matrix) #numeric*
       boolean matrix
    final_matrix = np.zeros((dims,dims)) #initialise
    for i in np.arange(dims):
        for j in np.arange(dims):
            final_matrix[i,j] = max(combined_matrix[i,j,:], key=abs) #
               selects biggest magnitude value across all lags
    return(final_matrix)


tau_max = 5 #maximum lag
i = int(sys.argv[1]) #retrieve slice index from file input

data1 = data[i:(i+1000),] #obtain data slice
dataframe1 = pp.DataFrame(data1) #convert to form for PCMCI input
val_matrix, link_matrix = parcorr_fn(dataframe1,var_names,tau_max) #
   perform ParCorr
final_matrix = matconv_fn(val_matrix, link_matrix) #obtain 2D matrix of
    significant edges
np.save('T_mat'+str(i)+'_pat'+str(pat), final_matrix) #save to current
   directory
```

A.2. ParCorr PBS shell script

```
#PBS -l walltime=72:00:00
#PBS -l select=1:ncpus=2:mem=2gb #set job specifications

module load anaconda3/personal
cp -a $PBS_O_WORKDIR/. $TMPDIR/ #copy all files to temporary directory

declare -i i #declare integer variable

for i in {0..100000..1000}; do #loop over slices
    python bashrun6140.py ${i}; #run Python script to save 'T_mat...'
       matrix
    cp $TMPDIR/T_* $PBS_O_WORKDIR/; #copy files starting 'T_' back to
       work directory
    rm $TMPDIR/T_*;
done;
```

A.3. PMIME MATLAB code e.g. 'bashrun6140.m'

PMIME functions are available from http://users.auth.gr/dkugiu/Software_en.html.

```
function [] = bashrun6140(i)

pat = '6140'; %patient number

raw = load(strcat('pat',pat,'.mat'));
cell = struct2cell(raw);
K = size(cell,1); %no. of variables
```

```
N = size(cell{1},2); %no. of observations
M = zeros(N,K);
for j = 1:K
    M(:,j) = cell{j};
end

[RM,ecC] = PMIMEsig(M(i+1:i+1000,:),5,1,5,100,0.05,0);
dlmwrite(strcat('P_mat',int2str(i),'_pat',pat,'.txt'),RM,'delimiter','\
   t');

end
```

A.4. PMIME PBS shell script

```
#PBS -l walltime=72:00:00
#PBS -l select=1:ncpus=2:mem=2gb

module load matlab
cp -a $PBS_O_WORKDIR/. $TMPDIR/ #copy all files to temporary directory

declare -i i #declare integer variable

for i in {0..100000..1000}; do #loop over slices
    matlab -nodisplay -nodesktop -r "bashrun6140($i)"; #run MATLAB
       script to save 'P_mat...' matrix
    cp $TMPDIR/P_* $PBS_O_WORKDIR/; #copy files starting 'P_' back to
       work directory
    rm $TMPDIR/P_*;
done;
```

B. Network metrics

Network metrics are calculated in MATLAB for each adjacency matrix, using code from https://sites.google.com/site/bctnet/Home. We omit the code here as there is not much to be learned from it. This section is purely a placeholder and just serves as a reminder of the steps in our procedure.

C. Machine learning in R

Assuming we have a dataframe of network metric values, our next goal is to use these metrics as predictors in classifying schizophrenia patients. We include here some of the more technical code (omitting plots etc.) and examples for random forest and neural network algorithms only.

C.1. Standardisation of training and testing subsets

```
#function which takes permutation of indices as input, and outputs
   standardised data frames
process_data <- function(train_inds, #indexes for training subset
                         test_inds,  #indexes for testing subset
```

```
                            letter){    #batch: U=unthresh. ParCorr, T=
                                ParCorr, P=PMIME
  if (letter=='U'){all <- all_U} #assign dataframe
  if (letter=='T'){all <- all_T}
  if (letter=='P'){all <- all_P}
  train <- all[train_inds,]      #training dataset
  test <- all[test_inds,]        #testing dataset

  #standardise the predictors based on TRAIN data
  trainmeans <- attr(scale(train[,1:22]),"scaled:center") #extract
     train means
  trainsds <- attr(scale(train[,1:22]),"scaled:scale")  #extract train
     std. devs
  train[,1:22] <- scale(train[,1:22]) #standardise train data
  test[,1:22] <- scale(test[,1:22],   #standardise test data
                     center=trainmeans,  #use training means
                     scale=trainsds)          #use training std. devs

  #cost efficiency
  ce <- train[,15]-train[,14] #variable based on train data (global
     efficiency - wiring cost)
  cemean <- mean(ce)
  cesd <- sd(ce)
  train[,23] <- scale(ce)
  test[,23] <- scale(test[,15]-test[,14], center=cemean, scale=cesd)
  return(list(train, test))
}
```

C.2. k-fold cross-validated classification accuracy for general model

```
kfold <- function(k,           #number of folds
                  modelpred,  #classification function which predicts
                     labels of inputs given training data
                  par,        #optional parameters needed for modelpred
                      function
                  letter){    #batch of matrices ('U', 'T' or 'P')
  train_acc <- vector("numeric", k)  #initialise vector of accuracy on
     train data per fold
  test_acc <- vector("numeric", k)   #initialise vector of accuracy on
     test data per fold
  train_coefs <- vector("list", k)   #initialise list of coefficient
     values per fold

  if (letter=='U'){n <- n_U} #assign number of observations for batch
  if (letter=='T'){n <- n_T}
  if (letter=='P'){n <- n_P}
  randseq <- sample(n, n)    #initialise random sequence (unchanged for
      all folds)
  size <- floor(n/k)         #size of test set for each fold

  for (fold in 1:k){         #now fold:
    test_inds <- randseq[((fold-1)*size+1):(fold*size)] #take (
       progressive) slices as testing
    train_inds <- (1:n)[-test_inds] #remaining indices for training
    datalist <- process_data(train_inds, test_inds, letter) #construct
       standardised datasets using these indices
    train <- datalist[[1]]       #assign training data
    test <- datalist[[2]]        #assign testing data
    train_y <- train[,25]        #training labels
    train_x <- train[,-c(24,25)] #training inputs
    test_y <- test[,25]          #testing labels
    test_x <- test[,-c(24,25)]   #testing inputs

    mtrain <- modelpred(train_x, train_x, train_y, par) #predict labels
        for training inputs
    mtest <- modelpred(test_x, train_x, train_y, par)   #predict labels
        for testing inputs

    train_preds <- mtrain[[1]] #obtain predictions for training inputs
    train_coefs[[fold]] <- mtrain[[2]] #optional model coefficients
    test_preds <- mtest[[1]] #obtain predictions for testing inputs

    train_acc[fold] <- sum(train_y==train_preds)/length(train_y) #
       compare with actual y for training accuracy
    test_acc[fold] <- sum(test_y==test_preds)/length(test_y)     #
       compare with actual y for testing accuracy
  }
  means <- c(mean(train_acc), mean(test_acc)) #average accuracies over
     all the folds
  coefs <- 0
  for (i in 1:k){coefs <- coefs + train_coefs[[i]]}
  coefs <- coefs/k #average optional model coefficients over all the
     folds
  return(list(means, coefs)) #output cross-validated train and test
     accuracies, and mean coef vals
}
```

C.3. Random forest

The function in C.2 requires as input another function which predicts labels (outcomes) for some inputs, given training inputs with known labels. There is also the option to output some model coefficients. Below we write such a function for random forest, with coefficients representing Gini importance per predictor.

```
require(randomForest)

rf_pred <- function(inputs, train_x, train_y, par){
  rf <- randomForest(train_y ~., train_x, importance=TRUE, nodesize=par
     )
  preds <- predict(rf, inputs)
  coefs <- importance(rf)[,4] #record Gini importance
  return(list(preds, coefs))
}
```

C.4. Neural network

Function for neural network to plug into function from C.2.

```r
require(nnet)

nn_pred <- function(inputs, train_x, train_y,
                    par){ #par = list(size, MaxNWts, maxit, num_tries)
  num_tries <- par[[4]]
  nn_list <- vector("list", num_tries) #initialise list of nns
  vals <- vector("numeric", num_tries) #initialise vector of final
     optimisation vals
  for (i in 1:num_tries){
    nn <- nnet(train_y ~., data=train_x,
               size=par[[1]], MaxNWts=par[[2]], maxit=par[[3]], trace=
                  FALSE)
    nn_list[[i]] <- nn #record this nn
    vals[i] <- nn$value #record final optimisation value
  }
  min <- which(vals==min(vals)) #index of nn with the lowest final
     optimisation value
  bestnn <- nn_list[[min]] #retrieve this nn
  preds <- predict(bestnn, inputs, type="class") #predict using best nn
  coefs <- NA #neural network gives no coefficients
  return(list(preds, coefs))
}
```

C.5. Parameter optimisation example

Use functions above to find parameter value that maximises cross-validated testing accuracy for neural network.

```r
pars <- c(seq(1,10,1),11,12,14,18,22,26) #test these parameter values
n <- length(pars)
trainaccsU <- vector("numeric", n) #initialise train accuracies
testaccsU <- vector("numeric", n) #initialise test accuracies
folds <- 5 #cross-validation folds

count = 0
for (par in pars){
  count = count+1
  nnU <- kfold(k=folds, modelpred=nn_pred, par=list(par,5000,300,10),
     letter='U') #obtain cross-validated accuracies for neural network
     with this parameter
  trainaccsU[count] <- nnU[[1]][1]
  testaccsU[count] <- nnU[[1]][2]
}

parmaxU <- pars[which(testaccsU==max(testaccsU))]
cat("For␣unthresholded␣ParCorr␣the␣optimal␣number␣of␣neurons␣is",
    parmaxU, "\n")
```