EEG-Based Machine Learning: Theory and Applications

Reza Shoorangiz, Stephen J. Weddell, and Richard D. Jones

Contents

R. Shoorangiz (✉)
Department of Electrical and Computer Engineering, University of Canterbury, Christchurch, New Zealand

Department of Medicine, University of Otago, Christchurch, New Zealand

Neurotechnology Research Programme, Christchurch, New Zealand

New Zealand Brain Research Institute, Christchurch, New Zealand
e-mail: reza.shoorangiz@nzbri.org

S. J. Weddell
Department of Electrical and Computer Engineering, University of Canterbury, Christchurch, New Zealand

Neurotechnology Research Programme, Christchurch, New Zealand
e-mail: steve.weddell@canterbury.ac.nz

R. D. Jones
Department of Electrical and Computer Engineering, University of Canterbury, Christchurch, New Zealand

Department of Medicine, University of Otago, Christchurch, New Zealand

Neurotechnology Research Programme, Christchurch, New Zealand

New Zealand Brain Research Institute, Christchurch, New Zealand

School of Psychology, Speech and Hearing at University of Canterbury, Christchurch, New Zealand
e-mail: richard.jones@nzbri.org

N. V. Thakor (ed.), *Handbook of Neuroengineering*,
https://doi.org/10.1007/978-981-16-5540-1_70

Abstract

Electroencephalography is a widely used clinical and research method to record and monitor the brain's electrical activity – the electroencephalogram (EEG). Machine learning algorithms have been developed to extract information from the EEG to help in the diagnosis of several disorders (e.g., epilepsy, Alzheimer's disease, and schizophrenia) and to identify various brain states. Despite the elegant and generally easy-to-use nature of machine learning algorithms in neuroscience, they can produce inaccurate and even false results when implemented incorrectly. In this chapter, we outline the general methodology for EEG-based machine learning, pattern recognition, and classification. First, a description of feature extraction from various domains is presented. This is followed by an overview of supervised and unsupervised feature-reduction methods. We then focus on classification algorithms, performance evaluation, and methods to prevent overfitting. Finally, we discuss two applications of EEG-based machine learning: brain-computer interface (BCI) and detection and prediction of microsleeps.

Keywords

EEG · Machine learning · Feature extraction · Dimensionality reduction · Cross-validation · Performance evaluation

Abbreviations

AdaBoost	adaptive boosting
AUC-PR	area under the curve of the precision recall
AUC-ROC	area under the curve of the receiver operating characteristic
bagging	bootstrap aggregating

BCI	brain-computer interface
CSP	common spatial pattern
EEG	electroencephalogram
FBCSP	filter bank common spatial pattern
FFT	fast Fourier transform
fMRI	functional magnetic resonance imaging
FN	false negative
fNIRS	functional near-infrared spectroscopy
FP	false positive
FSULR	unsupervised learning with ranking based feature selection
GM	geometric mean
HA	Hjorth activity
HC	Hjorth complexity
HM	Hjorth mobility
ICA	independent component analysis
IWBW	intensity-weighted bandwidth
IWMF	intensity-weighted mean frequency
kNN	k-nearest neighbour
KPCA	kernel principal component analysis
LDA	linear discriminant analysis
LOSO	leave one-subject out
PCA	principal component analysis
PPCA	probabilistic principal component analysis
PR	precision recall
Pr	precision
ROC	receiver operating characteristic
SN	sensitivity
SP	specificity
SSVEP	steady-state visual evoked potential
SVM	support vector machine
t-SNE	t-distributed stochastic neighbour embedding
TN	true negative
TP	true positive

1 Introduction

Electroencephalography is a noninvasive method to directly measure neural activity from electrodes placed on the scalp [1]. Synchronous activity of a large population of neurons generates an electric field that is strong enough to reach the scalp, which is recorded as the electroencephalogram (EEG) with a high temporal resolution [2]. Directly recording neural activity is one of the advantages of EEG compared to other neuroimaging methods, such as functional magnetic resonance imaging (fMRI) and functional near-infrared spectroscopy (fNIRS), which measure biochemical activity as a proxy for neural activity [3, 4]. Moreover, due to its high temporal

resolution, EEG captures a wide range of neural oscillations. These rhythms have been categorized into five standard bands: delta (0.5–4 Hz), theta (4–8 Hz), alpha (8–12 Hz), beta (13–30 Hz), and gamma (>30 Hz) [5]. Studies have shown that brain activity in each frequency band is associated with different cognitive functions [5]. These advantages make EEG a viable and practical option to investigate important questions in not only neural engineering and neuroscience but also clinical applications and disease diagnosis.

EEG signals contain a substantial amount of information with respect to spatial, temporal, and spectral aspects. This makes EEG a suitable method to investigate various aspects of brain function and cognition. However, the richness of EEG [5] comes at a cost, where data can be high dimensional and may have a low signal-to-noise ratio, which poses a considerable challenge to process EEG and identify patterns of interest. Machine learning has received considerable attention in the field to address the inherent challenges of EEG.

EEG is usually contaminated with noise and artifacts, such as eye movement, slow drift, and muscle artifact [6]. To increase the signal-to-noise ratio, a preprocessing step is commonly included to minimize artifacts and reduce unwanted noise. This step can include various procedures such as band-pass filtering [7], artifact subspace reconstruction [8], independent component analysis [9], spatial filters [10, 11], minimizing muscle artifact [12], and artifact rejection [13]. In preprocessing, however, one has to be cautious and visualize data to avoid eliminating any meaningful and informative component of EEG.

In this chapter, our aim is to focus on machine learning in EEG, specifically feature extraction, feature reduction, classification, and performance evaluation. Lastly, we provide two applications of machine learning using EEG signals.

2 Machine Learning

Machine learning is a set of algorithms that enable us to automatically identify patterns in the data and make predictions on newly observed measurements [14–16]. This is often the case in neural engineering and neuroscience experiments to (1) contrast between conditions [17, 18], (2) diagnose a disease [19, 20], or (3) identify electrophysiological changes associated with behavior [21, 22]. Despite different applications, the machine learning procedure remains similar in most cases, as shown in Fig. 1.

In general, machine learning has two phases: *training* and *testing*. In the training phase, a set of examples (i.e., data with their corresponding labels) are available. With a given machine learning algorithm, the example data are used to train a model (i.e., tune its parameters) so that it can identify the relationship between input data and the labels. In the testing phase, input data without labels go through the same methodology as the training phase for preprocessing, feature extraction, and feature reduction, and a trained model, which was estimated during training phase, predicts the output (i.e., labels). The main objective during the training phase is to estimate a model that has maximal predictive performance at the time of testing. It is important

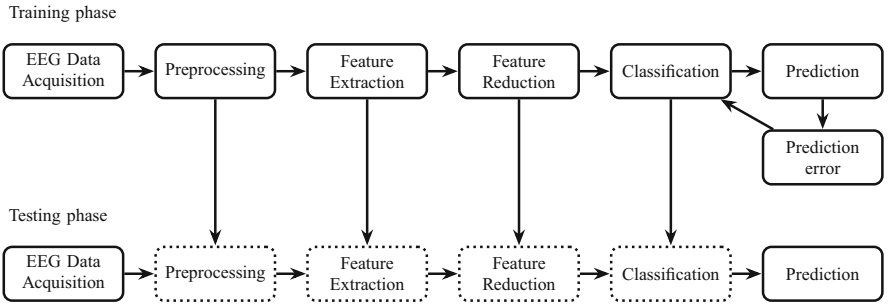

Fig. 1 A general overview of EEG-based machine learning. In the testing phase, preprocessing, feature extraction, and feature-reduction steps should follow the same methodology as the training phase. The classification step in the testing phase uses the optimized classifier in the training phase

to note that the term "predictive" in this context refers to *predicting* or *estimating* the unknown label of an observation. This is different from *temporal prediction* of an event in the future, such as prediction of future epileptic seizures or microsleep events. For the rest of this chapter, we limit our focus to the binary-classification case. However, similar steps with different performance metrics are used to train and test a regression model.

3 Feature Extraction

Feature extraction refers to a set of methods to reduce the dimension of the input data by measuring and extracting specific information. For EEG data, there are a wide range of methods to extract features from one or a combination of time, frequency, or spatial domains.

3.1 Time-Domain Features

Time-domain features are extracted from EEG signals without any transformation. *Zero crossing* is a time-domain feature that indicates the number of times the signal has crossed zero. This measure and the zero-crossing interval have been used for epilepsy detection [23], emotion recognition [24], and sleep staging [25].

Hjorth parameters are a set of three time-domain features describing a single channel of EEG [26]. These features are *activity*, *mobility*, and *complexity*. Hjorth activity (HA) is the variance of an EEG signal (i.e., signal power) and represents the width of the signal. Hjorth mobility (HM) estimates the mean frequency of the signal. Hjorth complexity (HC) estimates the bandwidth of the EEG signal by computing the mobility of the first derivative of EEG relative to the mobility of the EEG itself. Mathematically, the Hjorth parameters are calculated as [26, 27]

$$\mathrm{HA} = \sigma_0^2, \tag{1}$$

$$\mathrm{HM} = \frac{\sigma_1}{\sigma_0}, \tag{2}$$

$$\mathrm{HC} = \frac{\sigma_2 \sigma_0}{\sigma_1^2}, \tag{3}$$

where σ_0 is the standard deviation of the signal and σ_1 and σ_2 are the standard deviations of the first- and second-order derivatives of the signal. Hjorth parameters have been used for sleep staging [28], emotion recognition [29], and epilepsy detection [30].

Nonlinear energy, also known as mean Teager energy, is a feature of EEG that has been widely used for epileptic seizure prediction [31]. The nonlinear energy estimates instantaneous energy of a signal and, in particular, identifies transient changes such as sleep spindles and seizure spikes [32]. It has also been used for automatic sleep staging, where it has been ranked among the top features [25]. Let's assume the time-domain signal is represented by $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$, and the nonlinear energy is calculated by

$$\mathrm{NLE} = \frac{1}{N} \sum_{n=2}^{N-1} x_n^2 - x_{n-1} \times x_{n+1}. \tag{4}$$

Fractal dimension is a measure of self-similarity in different scales and is often used to quantify the complexity of a signal or a process [33]. Due to computational challenges of estimating fractal dimension of a complex signal, several methods have been developed to approximate fractal dimension [33]. *Petrosian fractal dimension* is the simplest approximation of fractal dimension [25]. Petrosian fractal dimension feature simplifies the computation of fractal dimension by transforming the signal to a binary representation and approximating the fractal dimension with the number of sign changes:

$$\mathrm{PFD} = \frac{\log_{10}(N_{\mathrm{EEG}})}{\log_{10}\left(\frac{N_{\mathrm{EEG}}}{(N_{\mathrm{EEG}} + 0.4 N_{\Delta})}\right)}, \tag{5}$$

where N_{EEG} is the number of sample points of the EEG signal and N_{Δ} is the number of sign changes of the signal. Petrosian fractal dimension has been used to measure depth of anesthesia [34], detect drowsiness level [35], and estimate sleep stage [36].

Katz fractal dimension is another method of estimating fractal dimension of a signal [37]. This measure is more accurate than Petrosian fractal dimension but is computationally more expensive. Katz fractal dimension is calculated as [38]

$$\mathrm{KFD} = \frac{\log\left(N_{\mathrm{EEG}} - 1\right)}{\log\left(N_{\mathrm{EEG}} - 1\right) + \log\left(\frac{d}{L}\right)}, \tag{6}$$

where N_{EEG} is the number of EEG-signal points, d is the diameter, and L is the curve length. Assuming that $\boldsymbol{x} = \{x_1, x_2, \ldots, x_{N_{\text{EEG}}}\}$ is the sequence of EEG signal, the diameter and curve length are

$$d = \max_n \left(|x_n - x_{n-1}|\right), \tag{7}$$

$$L = \sum_{n=2}^{N} |x_n - x_{n-1}|. \tag{8}$$

Katz fractal dimension has been used to diagnose patients with Alzheimer's disease [39], diagnose schizophrenic patients [40], and estimate drowsiness level [35].

Mean curve (line) length is an approximation of Katz fractal dimension [25]. Letting $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$ be an EEG signal, the mean curve length is calculated by

$$\text{MCL} = \frac{1}{N} \sum_{n=1}^{N} |x_n - x_{n-1}|. \tag{9}$$

Curve length has been used to predict epileptic seizure [41] and estimate different stages of sleep [25].

Hurst exponent is a measure of long-range self-similarity within a time series [42,43]. The Hurst exponent can take a value between 0 and 1, where a value of 0.5 corresponds to random data. To calculate the Hurst exponent, assume that the signal is given by $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$. Deviation of the first k data points from the mean of the first n data points is given by

$$W_k = \sum_{t=1}^{k} x_t - \frac{k}{n} \sum_{t=1}^{n} x_t, \quad \begin{matrix} 1 \le k \le n \\ 1 \le n \le N \end{matrix}. \tag{10}$$

The range $\text{R}(n)$ is defined as the maximum difference between the deviations of the first n-points:

$$\begin{aligned} \text{R}(n) &= \max(0, W_1, \ldots, W_n) \\ &\quad - \min(0, W_1, \ldots, W_n), \qquad 1 \le n \le N. \end{aligned} \tag{11}$$

The Hurst exponent is then given by

$$H \times n + C_H = \frac{\log\left(\frac{\text{R}(n)}{\text{S}(n)}\right)}{\log(n)}, \; 1 \le n \le N, \tag{12}$$

where C_H is a finite constant independent of n and $\text{S}(n)$ is the empirical standard deviation of the first n points. The Hurst exponent can be computed by fitting a

line to the right-hand side of (12). The Hurst exponent has been used to identify epileptiform EEG [44], recognize emotions [45], and estimate sleep stage [28].

3.2 Frequency-Domain Features

Frequency-domain features are extracted measures from the frequency representation of the EEG signals. The fast Fourier transform (FFT) is commonly used to identify the frequency components of EEG signals, which is given by

$$\boldsymbol{x}_f = \sum_{n=1}^{N} x_n e^{-i2\pi f(n-1)T_s}, \tag{13}$$

where f is the frequency, N is the total number of points in the signal, x_n is the n^{th} point of the signal, and T_s is the sampling time [6, 46]. Frequency components of a signal can be measured up to the Nyquist frequency (i.e., half of sampling frequency F_s). In addition, for a signal of length N, it is possible to compute $\frac{N}{2}+1$ frequency components that are uniformly distributed from 0 to Nyquist frequency (i.e., $f = \frac{k\times F_s}{N}, k \in \{0, 1, \cdots, \frac{N}{2}\}$). For instance, a 4-s EEG segment recorded with a sampling frequency of 250 Hz has a frequency resolution of $\frac{125}{\left(\frac{4\times 250}{2}\right)} = \frac{1}{4} =$ 0.25 Hz. From this example, it is clear that the frequency resolution depends on the length of signal and is independent of the sampling frequency. An assumption of the Fourier transform is that the signal is stationary, which means that the statistics (e.g., mean and standard deviation) of signal do not change with time. However, the EEG reflects the dynamics of brain function that is inherently nonstationary [47]. One way to get around this issue is by applying the FFT to a short segment of EEG which is reasonably stationary. Notwithstanding, although analyzing shorter data segments leads to stationary signals for FFT, it reduces the frequency resolution of the results. Moreover, to minimize spectral leakage, a windowing function needs to be applied to the signal, such as Hamming or Hanning window, before computing the FFT [48, 49].

There are other signal processing methods to quantify frequency components and to perform spectral analysis [48, 49], such as wavelet transform [50], Hilbert transform [51], and matching pursuit [52, 53]. These methods are commonly used to identify time-frequency representation of data. For the rest of this section, we focus on feature extraction from the frequency-domain representation of a signal.

Power spectral density estimates the distribution of power over frequency components in a given signal. Welch's method [54], also known as modified periodogram, is one of the widely used methods to estimate power spectral density [55]. Welch's method (modified periodogram) computes the power spectral density of a signal by averaging the periodogram of smaller overlapping windowed segments and, as a result, has a lower variance compared to the periodogram of the whole epoch [1]. Power spectral density has been used to detect and predict

microsleeps [22, 56], identify hallucinations in Alzheimer's disease [19], estimate sleep stage [28], and predict drowsiness [57].

Spectral entropy identifies the complexity or regularity of the EEG [31]. To calculate spectral entropy, the probability distribution of the signal is approximated by its power spectral density. The spectral entropy is then calculated by

$$S_{EN} = -\frac{1}{N_f} \sum_{f=f_l}^{f_u} \mathrm{PSD}(f) \log\left(\mathrm{PSD}(f)\right), \tag{14}$$

where N_f is the number of frequency bins and f_l and f_u are the lower and upper frequency limits, respectively. Spectral entropy has been used to classify schizophrenic patients [58], detect epileptic seizures [59], and estimate stage of sleep [25].

Intensity-weighted mean frequency (IWMF), also known as gravity frequency, finds the weighted average frequency of a signal relative to its power spectral density [60] via

$$\mathrm{IWMF} = \frac{\sum_f f \times \mathrm{PSD}(f)}{\sum_f \mathrm{PSD}(f)}. \tag{15}$$

Intensity-weighted bandwidth (IWBW), also known as frequency variability, is defined as variance of the frequency [60]. Using IWMF and spectral density, the calculation of IWBW is

$$\mathrm{IWBW} = \sqrt{\frac{\sum_f \mathrm{PSD}(f)\,(\mathrm{IWMF} - f)^2}{\sum_f \mathrm{PSD}(f)}}. \tag{16}$$

Both the IWMF and the IWBW have been used to detect drowsiness [60, 61] and seizures [31].

3.3 Spatial-Domain Features

Common spatial pattern (CSP) is a supervised algorithm for feature extraction from multichannel EEG signals from two conditions [62]. It is a data-driven method that aims to find a decomposition of EEG signals where the distance between two conditions in the new space is maximized. Let $\mathbf{X}_k = \{\mathbf{X}_{k,1}, \mathbf{X}_{k,2}, \cdots, \mathbf{X}_{k,N_k}\}$ be all EEG segments for condition $k \in \{1, 2\}$. The objective of CSP is to find a set of J spatial filters $\mathbf{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_J\}$ that project EEG signals into a new space with maximum distance between two conditions. For each spatial filter, the optimization problem can be written as [63]

$$\max_{\boldsymbol{w}_j} J(\boldsymbol{w}_j) = \frac{\boldsymbol{w}_j^\top \boldsymbol{\Sigma}_1 \boldsymbol{w}_j}{\boldsymbol{w}_j^\top \boldsymbol{\Sigma}_2 \boldsymbol{w}_j}, \tag{17}$$

where $\boldsymbol{\Sigma}_{k\in\{1,2\}}$ is the covariance matrix for k^{th} condition, $\left|\boldsymbol{w}_j\right| = 1$, and spatial filters are orthogonal. The covariance matrices can be computed as

$$\boldsymbol{\Sigma}_k = \frac{\sum_n \mathbf{X}_{k,n}\mathbf{X}_{k,n}^\top}{N_k}. \tag{18}$$

Solving (17) is equivalent to solving the generalized eigenvalue problem. In Matlab, this can be simply done by `W = eig(`$\boldsymbol{\Sigma}_1$`,` $\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2$`)` [63] to find the optimal spatial filters. These filters are then used to project EEG data into the new space using multiplication of spatial filters by EEG data ($\mathbf{WX}$). The variances of the projected signals (or logarithmic transformed) are extracted as features.

Filter bank CSP (FBCSP) is an extension of classical CSP, where the input data is first filtered into different frequency bands and a CSP is then found for the data in each frequency band [64]. Although FBCSP has shown superior performance compared to classical CSP [65], it is more likely to result in overfitting since it generates a large number of features.

A drawback of CSP is its susceptibility to noise, which can cause overfitting. Different methods have been developed to overcome this issue. One of these methods is to penalize complex models by incorporating a regularization term [66]. These methods attempt to optimize performance and generalizability of CSP by reducing complexity. The Bayesian framework has also been investigated to incorporate sparse priors to automatically select the optimal number of spatial filters [67,68]. Other methods implement feature selection techniques to select the optimum number of filters iteratively [69].

4 Feature Reduction

Extracted features from EEG data usually have high dimensionality [70]. For instance, one may filter EEG collected from 60 electrodes into 2-Hz sub-bands from 0 to 80 Hz and extract 3 features per electrode and sub-band. This leads to a total of $3 \times 60 \times 40 = 7200$ features, which is likely to be more than the number of observations. In machine learning, this is known as the *curse of dimensionality* and can result in overfitting [14,15]. When a model overfits, it can accurately identify the examples it has seen, but it performs poorly on new and previously unseen observations.

Feature reduction aims to improve generalizability of the machine learning model by reducing the number of features used in the predictive model [71–73]. Moreover, feature reduction may provide a better understanding of the underlying neural process for scientific questions. There are two major categories of feature reduction: *supervised* and *unsupervised*.

4.1 Unsupervised

Unsupervised feature reduction refers to the techniques that reduce the number of features without using the labels of training data [14]. In general, there are two approaches to reduce the number of features: *feature selection* [74] and *feature transformation* [14, 75]. In feature selection, a subset of the input features are selected, whereas in feature transformation, the reduced features (i.e., meta-features) are a lower-rank approximation of the input data [75, 76].

Several unsupervised feature-reduction methods have been developed [14, 70, 72, 76]. Mitra et al. [74] developed a method to select features to reduce redundancy. Similarly, Singh et al. [76] developed an unsupervised learning with ranking-based feature selection (FSULR) to eliminate redundant features. Feature transformation methods such as independent component analysis (ICA), principal component analysis (PCA), and t-distributed stochastic neighbor embedding (t-SNE) have been used to reduce features [70, 77–79]. In this section, the focus is restricted to PCA, which is a widely used technique to perform unsupervised feature reduction [80].

PCA is a linear feature-reduction method that decomposes correlated features into uncorrelated principal components (latent variables or meta-features) [15]. Formally, let $\mathbf{X} \in \mathcal{R}^{N \times D}$ be the feature matrix with N observations, and each observation has D features. PCA aims to linearly project data, yielding

$$\mathbf{Z} = \mathbf{W}\mathbf{X}^{\top}, \tag{19}$$

where $\mathbf{W} \in \mathcal{R}^{L \times D}$ is the loading matrix and $\mathbf{Z}$ is the matrix of latent variables (meta-features). PCA assumes that the loading matrix $\mathbf{W}$ is orthogonal (i.e., $\mathbf{W}\mathbf{W}^{\top} = \mathbf{I}$). To find the loading matrix, eigenvalue decomposition can be applied to the covariance of the feature matrix $\mathbf{\Sigma} = \frac{\mathbf{X}^{\top}\mathbf{X}}{N}$. The meta-features are then calculated as the projection of original features onto eigenvectors of the covariance matrix, as shown in Fig. 2. To reduce the dimensionality of features, we keep the leading eigenvectors corresponding to higher eigenvalues and discard the rest. These components will explain most of the variance in the original feature matrix. However, there is no consensus on how much of the variance should be explained by the retained principal components. Some researchers use a fixed value (such as 95%) as a cutoff, while others choose a predefined number of components (such as ten components). Another method to choose the optimum number of components is to use cross-validation (see Sect. 6).

PCA is a powerful feature-reduction technique, especially when dealing with high dimensional data. However, PCA is sensitive to the scale of data, which means that features with higher variances can mislead PCA [15]. This can be resolved by standardizing the feature matrix before applying PCA; that is, for each feature, remove its mean and divide by its standard deviation. After performing PCA, the meta-features (i.e., transformed features) will be used in the classification step.

Other variants of PCA have been developed to overcome shortcomings of PCA. Kernel PCA (KPCA) has been developed to extend PCA for nonlinear feature

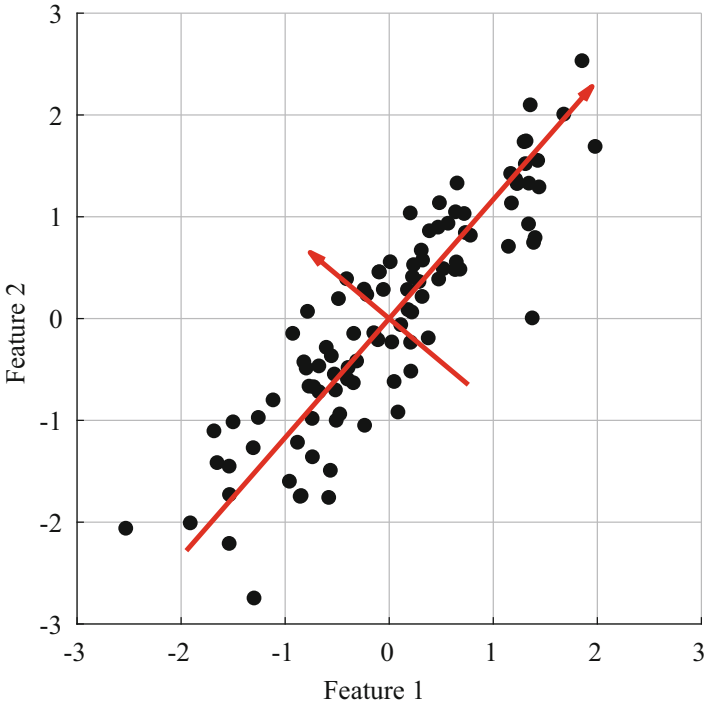

Fig. 2 Using PCA on two correlated features to find uncorrelated meta-features. The red lines correspond to the principal direction of the covariance matrix (direction of eigenvectors)

reduction [81]. Probabilistic PCA (PPCA) is a reformulation of PCA in the form of a probabilistic generative model [82]. An advantage of PPCA compared to PCA is that it explicitly models an additive Gaussian noise for the observations [16]. Bayesian variants of PCA have also been developed to incorporate sparsity-promoting priors (e.g., automatic relevance determination) to automatically identify the optimum number of components [83, 84].

4.2 Supervised

Supervised feature-reduction techniques exploit outcome labels (either categorical or continuous) when reducing the number of features. Feature selection encompasses the majority of supervised feature-reduction methods, in which a subset of relevant feature are selected and redundant and irrelevant features are discarded [71–73]. Feature selection methods can be divided into three categories: *filter*, *wrapper*, and *embedded* methods.

Filter methods use a rank function to score features based on their "relevancy" to the outcome labels [85]. Filter methods are computationally efficient and reduce propensity for overfitting. There are different rank functions to estimate relevancy of features such as correlation and mutual information. The most relevant features are then selected for the predictive model.

The correlation coefficient, which is often used to rank features, measures the degree of linear association between two random variables. This takes a value between -1 and 1 corresponding to complete negative and complete positive correlation, respectively. A correlation of 0 represents no linear association. Formally, let $\boldsymbol{f}_i \in \mathcal{R}^N$ be the vector corresponding to feature i and $\boldsymbol{y} \in \mathcal{R}^N$ be the vector of outcome labels (i.e., gold standard) for N observations. Pearson's correlation coefficient can be calculated as [85]

$$r_i = \frac{\sum_n \left(f_{i,n} - \bar{f}_i\right)(y_n - \bar{y})}{\sqrt{\sum_n \left(f_{i,n} - \bar{f}_i\right)^2}\sqrt{\sum_n (y_n - \bar{y})^2}}, \tag{20}$$

where $\bar{f}_i$ and $\bar{y}$ are the average of feature i and outcome labels, respectively. Pearson's correlation is susceptible to outliers and may achieve poor results when applied to noisy data. To alleviate this issue, Spearman correlation – a nonparametric version of the Pearson's correlation – can be used. Spearman correlation ranks the data first and then computes the correlation coefficient for the ranked data. Ranking the data eliminates the effect of an outlier without removing it from the data, and, thereby, Spearman correlation is less susceptible to outliers [6].

Mutual information measures the dependency between two variables [86, 87]. A value of 0 for mutual information indicates that the two variables are independent, whereas a positive value represents the level of dependency between the two variables. Formally, mutual information can be calculated as

$$I(\boldsymbol{f}_i, \boldsymbol{y}) = H(\boldsymbol{y}) - H(\boldsymbol{y}|\boldsymbol{x}_i), \tag{21}$$

where $\boldsymbol{f}_i$ and $\boldsymbol{y}$ correspond to the two variables (i.e., i^{th} feature vector and outcome labels), $H(\boldsymbol{x})$ is the entropy of variable $\boldsymbol{x}$, and $H(\boldsymbol{x}|\boldsymbol{y})$ is the conditional entropy of $\boldsymbol{x}$ given $\boldsymbol{y}$. The entropy of outcome label $\boldsymbol{y}$ is given by

$$H(\boldsymbol{y}) = -\sum_n \mathrm{p}(y_n)\log(\mathrm{p}(y_n)), \tag{22}$$

where $\mathrm{p}(x)$ is the probability mass function of discrete variable x. When x is continuous, the summation in (22) is replaced with integration and $\mathrm{p}(x)$ becomes the probability density function. The conditional entropy of feature i, given the outcome labels, is calculated by

$$H(\boldsymbol{y}|\boldsymbol{f}_i) = -\sum_{y}\int_{f} \mathrm{p}(f_i, y)\log(\mathrm{p}(y|f_i)), \tag{23}$$

where, using Bayes' rule, the conditional probability of the outcome given feature i is

$$\mathrm{p}(y|f_i) = \frac{\mathrm{p}(f_i|y)\mathrm{p}(y)}{\mathrm{p}(f_i)}. \tag{24}$$

Given the probability function of feature i and outcome labels, we can compute the mutual information of the two variables using (22)–(24). However, the probability distributions of these variables are not known a priori and therefore are required to be estimated from the data. One approach is to fit a Gaussian distribution to the observations of feature i, which is equivalent to a normal distribution with a mean and standard deviation estimated empirically from the data. This method is computationally fast and efficient for features that are normally distributed, but it fails to accurately model data that are not normally distributed or data that has a multimodal distribution. In these cases, a Gaussian mixture model or kernel density estimation can be used to estimate the probability distribution of data with higher accuracy [86, 88], but these models have higher computational complexity and take longer to evaluate.

Wrapper methods make use of a classification or regression method to select features iteratively [71–73]. These techniques find a subset of features based on the performance of a machine learning algorithm. Wrapper methods are generally divided into two categories: *forward selection* and *backward elimination*. In forward selection, each feature is first used separately to train a classification (or regression) model. The feature with the highest performance (or lowest error) is selected. Next, a combination of the best feature and one other feature is used to train another model. The combination of features that provides the highest performance is selected. This process continues until a stopping criterion is met. This can be a preselected number of features (e.g., ten features), a predefined threshold for the performance measure of interest (e.g., 90% accuracy), or improvement of the performance over iterations (e.g., 0.1% improvement). Backward elimination implements a similar idea but starts with all of the features. At each iteration, one feature is eliminated and the performance is evaluated. The feature with the lowest impact on performance is then eliminated.

Embedded methods are implemented within the training phase, and the features are selected during estimation of parameter values. For instance, regularizing classification algorithms with the L_1 norm has been widely used to shrink the coefficient of irrelevant features to zero in the training phase [89, 90]. This essentially removes the contribution of irrelevant and redundant features from the final predictive model.

5 Classification Algorithms

A classifier learns how to make predictions in the training phase and from the training data, which contains features and their corresponding categorical labels. After the training phase, the classifier can make predictions about new unseen data. When the problem at hand has a continuous outcome, regression methods are used to recognize patterns in the training data. A classification problem is binary when the outcome label can take two unique values, such as diagnosis of Alzheimer's disease [91]. In a multi-class classification problem, the outcome label has more than two states, such as sleep staging [92], emotion recognition [93], and BCI [94]. Although multi-class classifications have been used for numerous applications, we focus on binary classification in this section.

5.1 Linear Discriminant Analysis

Linear discriminant analysis (LDA) aims to find a hyperplane that maximally separates the features of different classes [14, 15]. LDA is a generative classifier that assumes that the data of each class is normally distributed and that the data of both classes have the same covariance matrix [14]. Let's assume that $\mathbf{F}_k \in \mathcal{R}^{N_k \times D}$ is the feature matrix for the k^{th} class (i.e., $k \in \{1, 2\}$). The mean and covariance matrix for LDA can be calculated as [14]

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \boldsymbol{f}_{n,k}, \tag{25}$$

$$\boldsymbol{\Sigma} = \frac{1}{\sum_k N_k - 2} \sum_{k=1}^{2} \sum_{n=1}^{N_k} \left(\boldsymbol{f}_{n,k} - \boldsymbol{\mu}_k\right) \left(\boldsymbol{f}_{n,k} - \boldsymbol{\mu}_k\right)^\top, \tag{26}$$

where $\boldsymbol{f}_{n,k}$ is a column vector corresponding to the features of the n^{th} observation of the k^{th} class. Since we are considering a binary outcome, we can classify a new observation by

$$\begin{aligned} d = {} & \hat{\boldsymbol{f}}^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right) \\ & - \frac{1}{2} \left(\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1\right)^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right) \\ & + \log\left(\frac{N_2}{N_1}\right), \end{aligned} \tag{27}$$

where N_k is the number of observations in class $k \in \{1, 2\}$ and a negative value of d classifies new observation $\hat{\boldsymbol{f}}$ to the first class and a positive value corresponds to the second class [14]. This is the generative view of LDA.

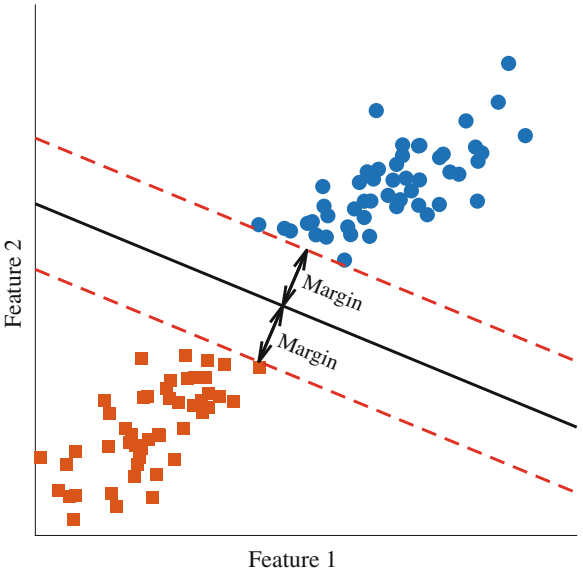

Fig. 3 Decision boundary of a linear SVM classifier (black line) and its margin. The blue dots and red squares correspond to the data of two classes

Another view of LDA is Fisher discriminant analysis which aims to maximize between-class variance while minimizing within-class variance. This can be written as the *Rayleigh quotient*, which can be solved by generalized eigenvalue decomposition [14]. Both generative and discriminative views of LDA lead to identical results.

5.2 Support Vector Machine

Support vector machine (SVM) is a linear classifier that finds a hyperplane between the data of two classes [15,95]. SVM is a maximum margin classifier, which means that the decision boundary is determined such that it maximizes the margin between the decision hyperplane and its surrounding data points [96], as shown in Fig. 3.

Finding a decision boundary with maximum distance from data of both classes reduces the chance of misclassification [95]. The closest data points to the decision boundary are called *support vectors*. These support vectors have the highest influence on the decision boundary.

Assume that $\boldsymbol{f}_n \in \mathcal{R}^D$ is the feature vector for the n^{th} observation and its corresponding outcome label is $y_n \in \{-1, 1\}$. For a classifier such as LDA that does not maximize the margin, the optimum boundary is given by

$$y_n \left(\boldsymbol{w}^\top \boldsymbol{f}_n + b \right) = 1, \tag{28}$$

where $\boldsymbol{w}$ is the weight vector and b is the bias term. Assuming that the data is linearly separable, the optimum boundary for SVM is given by [16]

$$y_n \left(\boldsymbol{w}^\top \boldsymbol{f}_n + b \right) \geq 1. \tag{29}$$

This leads to maximizing the margin during the training phase. However, a large weight vector $\boldsymbol{w}$ can satisfy this constraint. Therefore, the objective function of SVM is to minimize $\frac{|\boldsymbol{w}|^2}{2}$, subject to the constraints given by (29). This optimization can be done using quadratic programming. The testing phase of SVM is similar to that of LDA. Given a test data point $\hat{\boldsymbol{f}}$, the SVM prediction is done by computing

$$\hat{y} = \text{sign}(\boldsymbol{w}^\top \hat{\boldsymbol{f}} + b), \tag{30}$$

where $\hat{y}$ is the predicted class and $\text{sign}(x)$ is the sign function.

In most real-world applications, however, the data are not linearly separable. In this case, the perfect decision boundary does not exist, and the classifier will make at least one mistake. Therefore, the aim is to find a decision boundary with the highest accuracy. This is done by adding *slack variables*, $\xi_n \geq 0$, to account for classification error of each observation. When an observation is correct and has a high margin, its associated slack variable is zero. However, the value of a slack variable increases when its associated observation falls within the margin or gets misclassified. Incorporating slack variables in (31) gives

$$y_n \left(\boldsymbol{w}^\top \boldsymbol{f}_n + b \right) \geq 1 - \xi_n. \tag{31}$$

Similar to the case of linearly separable data, an objective function is required. To incorporate slack variables (error terms), the objective function to minimize is given by

$$J(\boldsymbol{w}) = \frac{|\boldsymbol{w}|^2}{2} + C \sum_{n=1}^{N} \xi_n, \tag{32}$$

where $C > 0$ is a regularization parameter which controls the balance between the error terms and the margin. A large value for C finds a smaller margin to reduce training error, whereas a small C attempts to find a larger margin with a higher training error.

When the true decision boundary between data of the two classes is nonlinear, finding a linear hyperplane may lead to poor classification performance. In this case, the *kernel trick* can be used to map input data to a different space where the decision boundary is linear [97, 98]. Then, linear classifiers can optimally find the decision boundary in the new space. Two of the most commonly used kernels are radial basis function [99] and polynomial [100]. However, kernel tricks substantially increase the computational complexity [101]. This becomes an issue when dealing with a large dataset that has many observations and features.

5.3 k-Nearest Neighbor

The k-nearest neighbor (kNN) is a nonlinear classifier that makes minimal assumptions about the data [14]. For a given test observation, kNN finds the k observations in the training data that have the lowest distance from the test observation. It then uses the majority label of those training points as the predicted class for the test observation. Closeness between data points is defined by a distance measure, where Euclidean distance is a common choice [102].

The kNN classifier uses a simple method to predict the label of a new observation, without explicitly modeling the training data. In practice, however, kNN can become unstable, especially when dealing with noisy data such as EEG [102]. In this case, increasing k would improve the generalization error. Moreover, kNN requires computing the distance of a new observation from all training data, which leads to high computational complexity.

5.4 Decision Tree

Decision tree is a simple and flexible classification method [14, 103] that has been widely used [104]. A decision tree classifies an observation by a set of hierarchical rules that form a tree structure. There are several algorithms to form a decision tree based on training data, such as CART, ID3, C4.5, and C5.0 [14, 15, 105–107]. Decision tree performs feature selection during the training phase, allows multiple use of a feature in different rules, and is easy to interpret [14, 108]. However, decision tree is prone to overfitting and the tree size can overgrow [14, 108].

5.5 Ensemble Methods

Ensemble learning refers to making predictions using multiple classifiers [14, 109]. It is desirable to achieve a classifier with maximal performance, but this might not be achieved with a single classifier. The idea of ensemble learning is that a higher performance can be achieved when multiple *weak* classifiers, which individually perform better than a random classifier, are combined. The achieved performance with ensemble learning is expected to be higher than the performance of each individual classifier. The three commonly used ensemble methods are *stacking*, *boosting*, and *bootstrap aggregating* (*bagging*).

Stacking, also known as stack generalization, is an ensemble method that combines the output of multiple classifiers [14]. The base classifiers are first trained independently. Base classifiers are also known as weak learners, weak classifiers, or base learners. Stacking finds a weighted average of the outputs of multiple classifiers to maximize the classification performance, which can be viewed as a type of model averaging. Stacking has shown a superior performance compared to a single classifier [110].

Boosting is an iterative method that combines multiple weak classifiers to create a strong and adaptive model [15]. At each iteration of the boosting method, data are first weighted to highlight the error from the previous classifiers. The weighted data are then used to train a new base learner, which aims to correct mistakes of previous classifiers. The base learner can be any classification algorithm. Notwithstanding, the most commonly used base learners are LDA, SVM, and decision tree [40, 109, 111].

Boosting methods have been widely used in the literature, and different variations of boosting have been developed [112]. The most widely used boosting is adaptive boosting (AdaBoost) [113]. Formally, let $\mathbf{F} = \{\boldsymbol{f}_1, \cdots, \boldsymbol{f}_N\}$ be the feature matrix for N observations, and let $\boldsymbol{y} = \{y_1, \cdots, y_N\}$ be the corresponding outcome labels (either 1 or -1). At the first iteration, the initial weight of each observation is set to $w_{1,n} = \frac{1}{N}$. Using the weight vector, the feature matrix, and the outcome labels, a base classifier h_1 is trained. Following this, the misclassification error of h_1 is calculated by

$$\epsilon_1 = \sum_{n=1}^{N} \begin{cases} 0 \text{ if } y_n = h_1(\boldsymbol{f}_n) \\ 1 \text{ if } y_n \neq h_1(\boldsymbol{f}_n) \end{cases}, \tag{33}$$

where $h_1(\boldsymbol{f}_n)$ is the prediction of classifier h_1 for the observation $\boldsymbol{f}_n$. The next step is to calculate the weight of this classifier, which is given by [112]

$$\alpha_1 = \frac{1}{2} \ln\left(\frac{1-\epsilon_1}{\epsilon_1}\right). \tag{34}$$

At this point, the first iteration is completed. For the consecutive iterations, the weight vector $\boldsymbol{w}_t$ is updated by

$$w_{t+1,n} = \frac{w_{t,n} \exp\left(-\alpha_t y_n h_t(\boldsymbol{f}_n)\right)}{Z_t}, \tag{35}$$

where Z_t is the normalization factor. In (35), it can be seen that the classification error from the current iteration has been taken into account when computing the weight vector for the next iteration. The iterative process continues until a predefined number of iterations T are completed (i.e., T base classifiers are trained). Making predictions for an unseen observation $\hat{\boldsymbol{f}}$ is done by computing a weighted average of classifier outputs, which is given by

$$\hat{y} = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{f})\right). \tag{36}$$

The prediction is a weighted average of the base learners according to their classification accuracy.

A disadvantage of AdaBoost is its propensity to overfit to the training data [114]. In the presence of label noise (i.e., the gold standard contains incorrect labels),

AdaBoost highly overfits to the wrong labels since it makes multiple attempts to classify all training data correctly [115]. However, this property of AdaBoost has been exploited to identify and remove observations with label noise [115, 116].

Bootstrap aggregating (bagging) also uses a mixture of base classifiers to predict an observation [117]. As opposed to boosting methods that train base learners sequentially, bagging performs parallel training of all of the classifiers. Initially, multiple subsets of the training data are created by bootstrapping. Each subset is then used to train a base classifier. After the training phase, each classifier is used to predict the label of a new unseen observation. The final prediction is done by using majority voting of the base classifiers. It has been shown that, compared to boosting, bagging is more robust when the data is noisy [114]. Moreover, bagging is more robust against label noise compared to boosting [115].

A decision tree is often used as a base classifier for the bagging method. On its own, decision tree is a simple method with relatively high flexibility that often leads to overfitting. However, when a decision tree is used as the base classifier in a bagging algorithm, it usually achieves high performances [15]. This combination is known as *random forest* [118]. Random forests have been shown to achieve high accuracy and have been widely used in the literature [119, 120].

5.6 Regularization

One of the big challenges in machine learning is to avoid overfitting, where the machine learning model can correctly identify the outcome label of all (or most) examples it has already seen but performs poorly on estimation of the outcome label of new unseen observations. Therefore, the trained model generalizes poorly to new observations [15]. Overfitting becomes more evident when the machine learning algorithm is highly flexible and able to fit closely to noise in the training data [14]. An important approach to reduce the likelihood of overfitting is to perform feature reduction (see Sect. 4).

A more explicit way to minimize overfitting is to use regularization terms. The concept of regularizing a model rests on penalizing the complexity of the model, thereby smoothing the decision boundary and allowing mistakes during training phase. The most commonly used regularizations are l_1 and l_2 norms [90, 121–126]. For a vector $\boldsymbol{w} = \{w_1, \cdots, w_K\}$, the l_p norm is computed by [124]

$$l_p = |\boldsymbol{w}|_p = \left(\sum_{k=1}^{K} |w_k|^p \right)^{\frac{1}{p}}. \tag{37}$$

Without a regularization term, the objective of the training phase is to minimize the error between the actual and the estimated outcome labels. This is done by minimizing a loss function $\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y})$, where $\boldsymbol{y}$ is the actual outcome labels and $\hat{\boldsymbol{y}} = G(\mathbf{F}, \boldsymbol{w})$ is the estimated outcome with a predictive model G that uses a weight

vector $\boldsymbol{w}$ to predict labels for feature matrix F. To incorporate regularization, the objective function J to minimize during the training phase should be a combination of the loss function and a regularization term:

$$J = \mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}) + \lambda R(\boldsymbol{w}), \tag{38}$$

where $R(\boldsymbol{w})$ is the regularization term and λ is a user-defined parameter to control balance between the complexity and the training accuracy of the model. When $\lambda = 0$, no regularization is applied, and the objective function given in (38) simplifies to the loss function. With a $\lambda > 0$, the objective function J becomes a compromise between training accuracy and smoothness of the decision boundary. Larger values of λ assign higher weight to the smoothness of the decision boundary, whereas smaller values put more emphasis on the correct identification of training examples [70, 124].

Cross-validation methods are used to select the optimal value for λ [123, 126]. This is done by performing cross-validation for different values of λ and selecting the value that corresponds to the highest predictive performance. Similar to the discussion in the cross-validation section, selecting an optimal value for λ from the whole dataset may lead to a biased estimate of the predictive performance. Nested cross-validations can be used to alleviate this issue, which refers to performing a cross-validation on the training data of another cross-validation. For instance, a *leave-one-subject-out* (LOSO) cross-validation with ten subjects partitions the data into nine subjects for the training phase and one subject for testing. To select the most appropriate λ, a set of LOSO cross-validations are then carried out on the nine subjects. After selecting λ, the first cross-validation uses data of the left-out subject to estimate an unbiased performance measure. The estimated predictive performance can then be used to compare different machine learning algorithms.

6 Cross-Validation

A machine learning algorithm, whether a classification or regression model, requires an independent set of test data to evaluate its predictive performance and generalization error after the training phase. However, we do not always have access to an independent test dataset. Therefore, the data at hand have to be exploited not only to train a model but also to evaluate the performance and generalizability of the trained model. This becomes even more important when the machine learning models are highly flexible. For instance, a kNN classifier with $k = 1$ correctly classifies all the training data, but the decision boundary is highly nonlinear and biased toward the training data. This model may well perform very poorly at prediction of the correct label of new unseen observations. On the other hand, when k is very large, the kNN classifier tends to favor the class with the highest number of training observations, which may also fail to correctly predict the label of new observations, especially for highly imbalanced datasets. In this situation, cross-validation can be used to

evaluate classification performance of the trained model for different parameters. The parameters that achieve the highest predictive performance are then selected.

Cross-validation is a simple technique to evaluate the generalization error of a model. The cross-validation partitions the training data into K nonoverlapping partitions, where a subset of $K-1$ partitions are used for training the model and the other partition is used to test it. This is called a *K-fold* cross-validation [14, 15]. In K-fold cross-validation, the whole process of training and testing is repeated K times, so that every partition has been used once as the test dataset. The overall performance of the model is then estimated as the average of predictive performance of the K models. Five- and ten-fold cross-validations are commonly used [14]. *Leave-one-out* cross-validation is also commonly used and is a special case when K is equal to the number of training observations.

One has to be cautious when using a cross-validation to avoid biasing the performance estimates, for instance, when feature-reduction methods are applied to the training data. One may apply feature selection based on the whole training dataset and apply the cross-validation to the selected features only. A cross-validation to compute the predictive performance of this model (i.e., after feature selection) will result in a biased estimate because the "test data" has been already used for feature selection [14]. To perform unbiased cross-validation in this case, we have to first partition the data into K partitions and leave one for testing. The rest of $K-1$ partitions are then used to perform feature selection and, subsequently, to train a classifier. We repeat this process K times to achieve an unbiased estimate of the classification performance.

As mentioned earlier, an unbiased estimate of predictive performance requires an independent test dataset. This is specifically important when dealing with EEG data, which are likely to be temporally correlated. Li et al. [127] found a within-subject correlation between the dynamics of EEG during a resting-wakefulness condition and the P3 component of evoked related potentials while performing a task. Other studies have exploited within-subject signatures of EEG to develop EEG-based *fingerprint* and *biometric* system applications [128, 129]. Findings of these studies suggest that subject-specific signatures of EEG are present in the data, even from multiple sessions; thereby, within-subject EEG segments from different time points may not be independent. As a result, while evaluating performance of a subject-independent model, performing K-fold cross-validation on the concatenated features from EEG of multiple subjects may result in a biased estimate of performance, which does not reflect the true generalization error. This is in line with the literature where subject-dependent performance of a model is often substantially higher than subject-independent performance [130, 131]. To evaluate an unbiased estimate of performance for a subject-independent model, we consider it essential to partition subjects for cross-validation. For instance, a five-fold cross-validation with ten subjects becomes an iterative process of using eight subjects for training and the other two for testing the model, irrespective of events of interest within each subject. Moreover, a LOSO cross-validation would refer to using data from one subject for evaluating a model that has been trained with data from the remaining subjects.

7 Performance Evaluation

Quantitative measures are required when evaluating the performance of a machine learning algorithm after the training phase. Several measures have been developed to evaluate various aspects of a predictive model's performance [132, 133]. In this section, we provide an overview of commonly used performance measures.

The *confusion matrix*, also called a contingency table, is a matrix that contains information regarding the number of correct and incorrect classifications for each class [134]. In a binary-classification problem, there are two classes that are usually identified as *positive* and *negative*; the positive class is the class of interest (e.g., epileptic seizures). A true positive is when a positive class is correctly predicted, whereas a false positive (Type I error) is when a negative class has been wrongly predicted as a positive class. The same concept for the negative class leads to a true negative and a false negative (Type II error). The confusion matrix of a binary problem, therefore, has four elements which are the total numbers of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). These provide an overview of predictive performance for the trained model.

Sensitivity (Sn), also known as the true positive rate or recall, is the proportion of correctly identified observations in the positive class. Similarly, *specificity* (Sp), also known as the true negative rate, is the proportion of correctly predicted instances in the negative class. *Precision* (Pr), also known as the positive predictive value, is the proportion of correctly identified positive instances relative to the total number of positive predictions. These measures can be calculated as [132]

$$\mathrm{Sn} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{39}$$

$$\mathrm{Sp} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}, \tag{40}$$

$$\mathrm{Pr} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{41}$$

Sensitivity measures how accurately the model can predict the positive class, without taking predictions of the negative class into account. Specificity, on the other hand, is a measure of the predictive ability of a model for the negative class. In contrast, both specificity and precision of a model are affected by false positives. Therefore, although these measures are quantifying various aspects of the performance of a predictive model, they do not provide the overall performance individually. For instance, a perfect precision (i.e., $\mathrm{Pr} = 1$) informs that the model correctly predicted the label of all of the negative-class observations and, hence, no false positives. However, it does not provide any information regarding the sensitivity of the model.

Accuracy is a widely used measure to estimate predictive performance of a classification model. Accuracy measures the ratio between correctly predicted labels and the total number of observations, as calculated by

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{42}$$

When the dataset is balanced (i.e., the number of negative and positive observations is equal), an accuracy of 0.85 corresponds to correct predicting of the label of 85% of data. However, the accuracy measure can be quite misleading when estimating predictive performance of a class-imbalanced dataset [135–137]. For instance, in a dataset with 85 observations from the negative class and 15 observations from the positive class, a classifier that predicts everything as the negative class would still indicate a high accuracy of 0.85.

F-measure is a measure that estimates predictive performance as the harmonic mean of sensitivity and precision [133, 134]. It can take a value from 0 to 1, which correspond to the worst and the best performances, respectively, and is calculated by [133]

$$\text{F-measure} = \frac{(1+\beta)^2 \times \text{Sn} \times \text{Pr}}{\beta^2 \times \text{Sn} + \text{Pr}}, \tag{43}$$

where β is a user-defined parameter to adjust the relative importance of sensitivity versus precision.

The *geometric mean* (GM) is a performance measure, based on sensitivity and specificity [133–135, 138], which is calculated by

$$\text{GM} = \sqrt{\text{Sn} \times \text{Sp}}. \tag{44}$$

The *phi correlation coefficient*, also known as Matthews correlation coefficient, is another performance metric that has been widely used to evaluate performance on imbalanced datasets [134, 137, 139]. It is calculated by

$$\text{phi} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP}+\text{FP})(\text{TN}+\text{FN})(\text{TP}+\text{FN})(\text{TN}+\text{FP})}}. \tag{45}$$

All of the aforementioned measures exploit predicted labels to estimate performance of a predictive model at a specific threshold for the classifier's output. In contrast, the *receiver operating characteristic* (ROC) curve uses the output of a classifier before thresholding and estimates the trade-off trajectory between sensitivity and specificity at different thresholds [140]. The ROC curve is estimated by using different thresholds to dichotomize the classifier's output. For each threshold, the sensitivity and false positive rate (i.e., $1 - \text{Sp}$) are evaluated. Finally, a ROC curve is formed by plotting sensitivity versus the false positive rate (shown in Fig. 4). The ROC curve of a random classifier is a straight line connecting the sensitivity and the false positive rate of (0, 0) to their corresponding (1, 1) values (the blue line in Fig. 4) [140]. On the other hand, the perfect classifier corresponds to a sensitivity of 1 and a false positive rate of 0 on the ROC curve. The area under the curve of ROC (AUC-ROC) is commonly used as a performance metric for classifiers

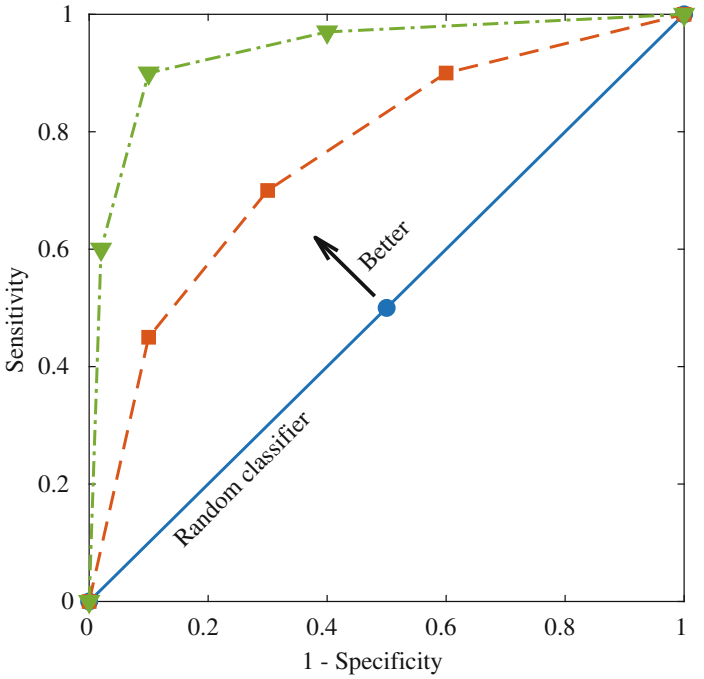

Fig. 4 ROC curve of three hypothetical classifiers. The blue line corresponds to a random classifier. The classifier corresponding to the green curve is better than the classifier corresponding to the red line

[133, 138, 140]. The AUC-ROC takes a value between 0 and 1, where 1 corresponds to a perfect classifier, 0.5 corresponds to a random classifier, and 0 corresponds to a perfectly inverted classifier.

The *precision-recall* (PR) curve is another method and is similar to ROC curve. The PR curve uses the classifier's output prior to thresholding [133]. The PR curve plots the trade-off between sensitivity and precision at different thresholds (as shown in Fig. 5). For a random classifier, the PR curve is a horizontal line at $\frac{P}{(P+N)}$, where P and N are the total number of the positive and the negative classes, respectively. A classifier that performs better than random achieves a PR curve above that of a random classifier. For example, a random classifier for a class-balanced dataset is a horizontal line at 0.5 (shown as blue line in Fig. 5), whereas better-than-random classifiers have PR curves above 0.5 (shown as red and green lines in Fig. 5). It has been suggested that the PR curve provides a better assessment of a classifier's performance when the dataset is highly skewed (i.e., class imbalanced) [133, 141, 142]. The area under the curve of PR (AUC-PR) provides a single measure that can be used as a performance metric.

It has been suggested that more than a single performance metric is required to assess an imbalanced learning problem [133, 143, 144]. This is due to the shortcomings and potential biases of individual performance metrics. For instance,

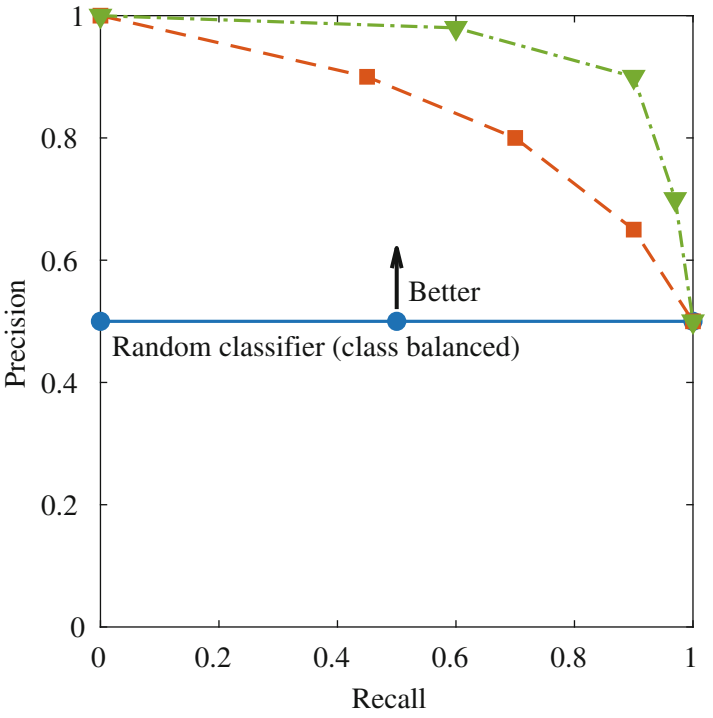

Fig. 5 PR curve of three hypothetical classifiers. The blue line corresponds to a random classifier when the data is class balanced. The classifier corresponding to the green curve has better performance than the classifier corresponding to the red line

precision does not provide any information regarding the number of false negatives, whereas sensitivity does not contain information on false positives. However, the F-measure, GM, and phi metrics provide information about different combinations of the contingency table. Both phi and F-measure offer insight on the functionality of a classifier, whereas GM provides information about the balance between sensitivity and specificity [133, 135, 138, 143]. In addition, sensitivity to the class-imbalance distribution is another potential issue. Precision, phi, F-measure, and AUC-PR are all sensitive to the imbalance ratio of the data [133, 142]. Therefore, we suggest reporting multiple performance metrics when dealing with class-imbalanced datasets.

8 Applications

An immense amount of research has focused on machine learning in EEG-based systems. There are numerous applications for EEG-based machine learning. An important application is to use machine learning to identify and extract biomarkers

from EEG for neurological disorders, such as Alzheimer's disease [145], Parkinson's disease [146], epilepsy and epileptic seizures [44], and dementia [147]. Other applications of machine learning in EEG include brain-computer interface (BCI) [148], sleep staging [25], drowsiness detection [60], estimation of depth of anesthesia [149], and microsleep detection and prediction [21,56,110,150]. Despite different applications, implementation of the machine learning procedure in these EEG systems follows similar steps as described in this chapter. For the rest of this section, we provide further details for two applications of machine learning in EEG. These are brain-computer interface (BCI) and microsleep detection and prediction.

8.1 Brain-Computer Interface

A BCI system enables users to interact with their surrounding using brain activity [151, 152]. BCI systems are of particular importance for people with severe disabilities, where BCI systems empower them to control their prosthetics and/or environment without using any muscles or peripheral nerves [153]. These systems commonly use EEG to record electrical activity of the brain because EEG is low-cost, has high temporal resolution, and has a low associated risk [114, 152].

One class of BCI systems focuses on *motor imagery* [152]. In this paradigm, a participant mentally simulates performing a series of movements. The aim of the BCI system is then to distinguish different types of movements using brain activity. Several studies have investigated motor imagery BCI and have achieved relatively acceptable performances (e.g., [154, 155]). Using a similar concept, other systems have been developed to control robotic arms and unmanned aerial vehicles [148, 156]. In these systems, a diverse range of feature extraction methods have been employed, including CSP [154, 157], coefficients of wavelet transform [158], spectral features [159], convolutional neural networks [160], and autoencoder [161]. Additionally, a range of classifiers have been used to separate motor imagery tasks, such as LDA [157], SVM [154], kNN [158], ensemble classifier [162], naive Bayes [65], and deep neural networks [163].

P300 speller is another paradigm of BCI [152, 164]. In the P300 speller, participants are presented with a table of characters where the intensity of one row or column is randomly increased. Participants are instructed to focus on the letter of interest, which randomly gets highlighted. This change in intensity produces a reaction in brain activity of the participant which happens approximately 300 ms after the letter is highlighted – i.e., P300. Using the P300 pattern, a BCI system can identify the letter of interest. The P300 speller paradigm has been widely studied in the literature and has achieved relatively good performances (e.g., [165, 166]). Several classifiers have been used to identify the letter of interest in a P300-speller paradigm, such as LDA [167, 168], SVM [168, 169], deep neural networks [165, 170], ensemble classifier [171, 172], and random forest [173].

There are other BCI paradigms such as steady-state visual evoked potential (SSVEP), auditory, visual, and hybrid [152]. These paradigms have also been the subject of many studies (e.g., [174–178]). There are numerous studies investigating

different BCI paradigms, and the number of publications is increasing. The findings of these studies show a promising future to improve quality of life for those who suffer from severe neurological and musculoskeletal disorders.

8.2 Microsleep Detection and Prediction in Time

Microsleeps are brief (<15 s) episodes of unintentional sleep-related loss of consciousness [179, 180]. A microsleep happens without warning and is usually accompanied by behavioral cues, such as head nodding, droopy eyes, and slow eye closure [180]. Most people have experienced microsleeps while reading a book or attending a lecture. However, microsleeps can lead to serious injuries and fatalities if they occur during certain tasks such as driving. Furthermore, many normally rested healthy individuals have been shown to experience frequent microsleeps while performing an extended monotonous task [180–182]. Therefore, detection and prediction of microsleeps is critical for occupations that require an extended unimpaired visuomotor performance, such as driving. Ultimately, prediction and prior warning of microsleeps can prevent accidents and save lives.

Several studies have investigated detection of microsleeps from EEG signals [22, 110, 183, 184]. In these studies, participants performed a continuous tracking task to simulate a driving task. Microsleeps were then defined from tracking performance and facial video. For each state of microsleep, a corresponding window of EEG was used to extract features. Using these features, various machine learning algorithms were then applied to detect microsleeps.

The prediction of imminent microsleeps has also been the subject of several studies [21, 56, 150, 185–188]. In these studies, selection of the EEG window corresponding to a microsleep state was done in a manner so that the EEG window preceded its corresponding microsleep state by a certain amount of time [150].

In terms of performance, microsleep detection and prediction systems have achieved relatively high AUC-ROC values (e.g., 0.95 [21, 188]). However, the precision of these systems is relatively low (e.g., 0.36 [188] and 0.42 [21] for microsleep prediction 0.25 s ahead). One of the challenges associated with microsleep systems is that microsleep data has an inherently high class imbalance. Additionally, the class-imbalance ratio varies across individuals. This introduces complexity for training the system and evaluating its performance.

9 Conclusion

An immense amount of research has focused on EEG and its applications in medicine, neuroscience, rehabilitation, and other fields. Integration of the EEG and machine learning fields has provided a framework to develop accurate EEG-based predictive systems. Such advances have resulted in EEG-based BCI systems that can substantially improve the quality of life for those suffering from severe neural and neuromuscular disorders.

In this chapter, we have provided an overview of machine learning algorithms for EEG-based systems. We divided the process into EEG data acquisition, preprocessing, feature extraction, feature reduction, classification, and performance evaluation. For each step, a brief summary was provided and potential challenges were discussed. However, the field of machine learning is vast, and therefore this chapter makes no attempt to review all of the existing literature. Instead, we have provided an overview of different steps that can be combined to develop an EEG-based predictive system.

We consider that machine learning will play an increasingly important role in EEG-based systems and their applications. In particular, deep neural networks will become an increasingly popular choice to develop EEG-based systems. These methods provide a framework to benefit from both model-based and data-driven approaches, which requires minimal processing for EEG data.

References

1. Tong, S.,Thakor, N.V.: Quantitative EEG analysis methods and clinical applications. Artech House engineering in medicine & biology series. Artech House (2009)
2. Lopes da Silva, F.: EEG and MEG: Relevance to neuroscience. Neuron **80**(5), 1112–1128 (2013). https://doi.org/10.1016/j.neuron.2013.10.017
3. Logothetis, N.K.: What we can do and what we cannot do with fMRI. Nature **453**, 869 (2008). https://doi.org/10.1038/nature06976
4. Irani, F., Platek, S.M., Bunce, S., Ruocco, A.C., Chute, D.: Functional near infrared spectroscopy (fNIRS): An emerging neuroimaging technology with important applications for the study of brain disorders. Clin. Neuropsychol. **21**(1), 9–37 (2007). https://doi.org/10.1080/13854040600910018
5. Buzsaki, G.: Rhythms of the brain. Oxford University Press, New York (2006)
6. Cohen, M.X.: Analyzing neural time series data: Theory and practice. The MIT Press, Cambridge (2014)
7. Widmann, A., Schröger, E., Maess, B.: Digital filter design for electrophysiological data – A practical approach. J. Neurosci. Meth. **250**, 34–46 (2015). https://doi.org/10.1016/j.jneumeth.2014.08.002
8. Mullen, T.R., Kothe, C.A.E., Chi, Y.M., Ojeda, A., Kerth, T., Makeig, S., Jung, T.-P., Cauwenberghs, G.: Real-time neuroimaging and cognitive monitoring using wearable dry EEG. IEEE Trans. Biomed. Eng. **62**(11), 2553–2567 (2015). https://doi.org/10.1109/TBME.2015.2481482
9. Delorme, A., Sejnowski, T., Makeig, S.: Enhanced detection of artifacts in EEG data using higher-order statistics and independent component analysis. NeuroImage **34**(4), 1443–1449 (2007). https://doi.org/10.1016/j.neuroimage.2006.11.004
10. Cohen, M.X.: Comparison of linear spatial filters for identifying oscillatory activity in multichannel data. J. Neurosci. Meth. **278**, 1–12 (2017). https://doi.org/10.1016/j.jneumeth.2016.12.016
11. Jonmohamadi, Y., Poudel, G., Innes, C., Weiss, D., Krueger, R., Jones, R.: Comparison of beamformers for EEG source signal reconstruction. Biomed. Signal Process. Control **14**(Supplement C), 175–188 (2014). https://doi.org/10.1016/j.bspc.2014.07.014
12. Clercq, W.D., Vergult, A., Vanrumste, B., Van Paesschen, W., Van Huffel, S.: Canonical correlation analysis applied to remove muscle artifacts from the electroencephalogram. IEEE Trans. Biomed. Eng. **53**(12), 2583–2587 (2006). https://doi.org/10.1109/TBME.2006.879459

13. Nolan, H., Whelan, R., Reilly, R.B.: FASTER: Fully automated statistical thresholding for EEG artifact rejection. J. Neurosci. Meth. **192**(1), 152–162 (2010). https://doi.org/10.1016/j.jneumeth.2010.07.015
14. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Series in Statistics, 2nd edn. Springer, New York (2009)
15. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. MIT Press, Cambridge (2013)
16. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Berlin/Heidelberg (2006)
17. Müller, K.-R., Tangermann, M., Dornhege, G., Krauledat, M., Curio, G., Blankertz, B.: Machine learning for real-time single-trial EEG-analysis: From brain–computer interfacing to mental state monitoring. J. Neurosci. Meth. **167**(1), 82–90 (2008). https://doi.org/10.1016/j.jneumeth.2007.09.022
18. Lemm, S., Schafer, C., Curio, G.: BCI competition 2003-data set III: Probabilistic modeling of sensorimotor mu; rhythms for classification of imaginary hand movements. IEEE Trans. Biomed. Eng. **51**(6), 1077–1080 (2004). https://doi.org/10.1109/TBME.2004.827076
19. Dauwan, M., Linszen, M.M.J., Lemstra, A.W., Scheltens, P., Stam, C.J., Sommer, I.E.: EEG-based neurophysiological indicators of hallucinations in Alzheimer's disease: Comparison with dementia with Lewy bodies. Neurobiol. Aging **67**, 75–83 (2018). https://doi.org/10.1016/j.neurobiolaging.2018.03.013
20. Kumar, Y., Dewal, M.L., Anand, R.S.: Epileptic seizure detection using DWT based fuzzy approximate entropy and support vector machine. Neurocomputing **133**, 271–279 (2014). https://doi.org/10.1016/j.neucom.2013.11.009
21. Buriro, A.B., Shoorangiz, R., Weddell, S.J., Jones, R.D.: Predicting microsleep states using EEG inter-channel relationships. IEEE Trans. Neural Syst. Rehabilitat. Eng. **26**(12), 2260–2269 (2018). https://doi.org/10.1109/TNSRE.2018.2878587
22. Davidson, P.R., Jones, R.D., Peiris, M.T.R.: EEG-based lapse detection with high temporal resolution. IEEE Trans. Biomed. Eng. **54**(5), 832–839 (2007). https://doi.org/10.1109/TBME.2007.893452
23. Shahidi Zandi, A., Tafreshi, R., Javidan, M., Dumont, G.A.: Predicting epileptic seizures in scalp EEG based on a variational Bayesian Gaussian mixture model of zero-crossing intervals. EEE Trans. Biomed. Eng. **60**(5), 1401–1413 (2013). https://doi.org/10.1109/TBME.2012.2237399
24. Kwon, Y.-H., Shin, S.-B., Kim, S.-D.: Electroencephalography based fusion two-dimensional (2D)-convolution neural networks (CNN) model for emotion recognition system. Sensors **18**(5), 1383 (2018). https://doi.org/10.3390/s18051383
25. Şen, B., Peker, M., Çavuşoğlu, A., Çelebi, F.: A comparative study on classification of sleep stage based on EEG signals using feature selection and classification algorithms. J. Med. Syst. **38**(3), 1–21 (2014). https://doi.org/10.1007/s10916-014-0018-0
26. Hjorth, B.: EEG analysis based on time domain properties. Electroencephalography Clin. Neurophysiol. **29**(3), 306–310 (1970). https://doi.org/10.1016/0013-4694(70)90143-4
27. Vidaurre, C., Krämer, N., Blankertz, B., Schlögl, A.: Time domain parameters as a feature for EEG-based brain-computer interfaces. Neural Netw. **22**(9), 1313–1319 (2009). https://doi.org/10.1016/j.neunet.2009.07.020
28. Chapotot, F., Becq, G.: Automated sleep–wake staging combining robust feature extraction, artificial neural network classification, and flexible decision rules. Int. J. Adap. Control Signal Process. **24**(5), 409–423 (2010). https://doi.org/10.1002/acs.1147
29. Padilla-Buritica, J.I., Martinez-Vargas, J.D., Castellanos-Dominguez, G.: Emotion discrimination using spatially compact regions of interest extracted from imaging EEG activity. Front. Comput. Neurosci. **10**, 55 (2016). https://doi.org/10.3389/fncom.2016.00055
30. Cecchin, T., Ranta, R., Koessler, L., Caspary, O., Vespignani, H., Maillard, L.: Seizure lateralization in scalp EEG using Hjorth parameters. Clin. Neurophysiol. **121**(3), 290–300 (2010). https://doi.org/10.1016/j.clinph.2009.10.033

31. Greene, B.R., Faul, S., Marnane, W.P., Lightbody, G., Korotchikova, I., Boylan, G.B.: A comparison of quantitative EEG features for neonatal seizure detection. Clin. Neurophysiol. **119**(6), 1248–1261 (2008). https://doi.org/10.1016/j.clinph.2008.02.001
32. Imtiaz, S.A., Saremi-Yarahmadi, S., Rodriguez-Villegas, E.: Automatic detection of sleep spindles using Teager energy and spectral edge frequency. In: Proceedings of IEEE Biomedical Circuits System Conference, pp. 262–265. IEEE (2013). https://doi.org/10.1109/BioCAS.2013.6679689
33. Di Ieva, A., Grizzi, F., Jelinek, H., Pellionisz, A.J., Losa, G.A.: Fractals in the neurosciences, Part I: General principles and basic neurosciences. The Neuroscientist **20**(4), 403–417 (2014). https://doi.org/10.1177/1073858413513927
34. Hosseini, S.A.: A computational framework to discriminate different anesthesia states from EEG signal. Biomed. Eng. Appl. Basis Commun. **30**(03), 1850020 (2018). https://doi.org/10.4015/S1016237218500205
35. Pavithra, M., NiranjanaKrupa, B., Sasidharan, A., Kutty, B.M., Lakkannavar, M.: Fractal dimension for drowsiness detection in brainwaves. In: Proc. Int. Conf. Contemp. Comput. Informat. (IEEE, 2014) pp. 757–761. https://doi.org/10.1109/IC3I.2014.7019676
36. Yan, R., Zhang, C., Spruyt, K., Wei, L., Wang, Z., Tian, L., Li, X., Ristaniemi, T., Zhang, J., Cong, F.: Multi-modality of polysomnography signals' fusion for automatic sleep scoring. Biomed. Signal Process. Control **49**, 14–23 (2019). https://doi.org/10.1016/j.bspc.2018.10.001
37. Paramanathan, P., Uthayakumar, R.: Application of fractal theory in analysis of human electroencephalographic signals. Comput. Biol. Med. **38**(3), 372–378 (2008). https://doi.org/10.1016/j.compbiomed.2007.12.004
38. Polychronaki, G.E., Ktonas, P.Y., Gatzonis, S., Siatouni, A., Asvestas, P.A., Tsekou, H., Sakas, D., Nikita, K.S.: Comparison of fractal dimension estimation algorithms for epileptic seizure onset detection. J. Neural Eng. **7**(4), 046007 (2010)
39. Ahmadlou, M., Adeli, H., Adeli, A.: Fractality and a wavelet-chaos-methodology for EEG-based diagnosis of Alzheimer disease. Alzheimer Dis. Assoc. Disorders **25**(1) (2011)
40. Sabeti, M., Katebi, S.D., Boostani, R., Price, G.W.: A new approach for EEG signal classification of schizophrenic and control participants. Expert Syst. Appl. **38**(3), 2063–2071 (2011). https://doi.org/10.1016/j.eswa.2010.07.145
41. D'Alessandro, M., Vachtsevanos, G., Hinson, A., Esteller, R., Echauz, J., Litt, B.: A genetic approach to selecting the optimal feature for epileptic seizure prediction. In: Proceedings of International Conference IEEE Engineering Medical Biology Society, Vol. 23, pp. 1703–1706. IEEE (2001). https://doi.org/10.1109/IEMBS.2001.1020544
42. Blythe, D., Haufe, A.J.S., Müller, K.-R., Nikulin, V.V.: The effect of linear mixing in the EEG on Hurst exponent estimation. NeuroImage **99**, 377–387 (2014). https://doi.org/10.1016/j.neuroimage.2014.05.041
43. Carreras, B.A., van Milligen, B.P., Pedrosa, M.A., Balbín, R., Hidalgo, C., Newman, D.E., Sánchez, E., Frances, M., García-Cortés, I., Bleuel, J., Endler, M., Riccardi, C., Davies, S., Matthews, G.F., Martines, E., Antoni, V., Latten, A., Klinger, T.: Self-similarity of the plasma edge fluctuations. Phys. Plasmas **5**(10), 3632–3643 (1998). https://doi.org/10.1063/1.873081
44. Yuan, Q., Zhou, W., Li, S., Cai, D.: Epileptic EEG classification based on extreme learning machine and nonlinear features. Epilepsy Res. **96**(1–2), 29–38 (2011) . https://doi.org/10.1016/j.eplepsyres.2011.04.013
45. Wang, X.-W., Nie, D., Lu, B.-L.: Emotional state classification from EEG data using machine learning approach. Neurocomputing **129**, 94–106 (2014). https://doi.org/10.1016/j.neucom.2013.06.046
46. Oppenheim, A.V., Willsky, A.S., Nawab, S.H.: Signals & Systems. Prentice-Hall, Inc., Upper Saddle River (1996)
47. Kaplan, A.Y., Fingelkurts, A.A., Fingelkurts, A.A., Borisov, S.V., Darkhovsky, B.S.: Nonstationary nature of the brain activity as revealed by EEG/MEG: Methodological, practical and

conceptual challenges. Signal Process. **85**(11), 2190–2212 (2005) . https://doi.org/10.1016/j.sigpro.2005.07.010
48. Muthuswamy, J., Thakor, N.V.: Spectral analysis methods for neurological signals. J. Neurosci. Meth. **83**(1), 1–14 (1998). https://doi.org/10.1016/S0165-0270(98)00065-X
49. Gross, J.: Analytical methods and experimental approaches for electrophysiological studies of brain oscillations. J. Neurosci. Meth. **228**, 57–66 (2014). https://doi.org/10.1016/j.jneumeth.2014.03.007
50. Mallat, S.: A wavelet tour of signal processing, 3rd edn (pp. 795–805). Academic Press, Boston (2009)
51. Freeman, W.J.: Hilbert transform for brain waves. Scholarpedia **2**(1), 1338 (2007). https://doi.org/10.4249/scholarpedia.1338
52. Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Process. **41**(12), 3397–3415 (1993) . https://doi.org/10.1109/78.258082
53. Durka, P.J., Blinowska, K.J.: Analysis of EEG transients by means of matching pursuit. Ann. Biomed. Eng. **23**(5), 608–611 (1995) . https://doi.org/10.1007/BF02584459
54. Welch, P.: The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. IEEE Trans. Audio Electroacous. **15**(2), 70–73 (1967). https://doi.org/10.1109/TAU.1967.1161901
55. Babiloni, C., Pistoia, F., Sarà, M., Vecchio, F., Buffo, P., Conson, M., Onorati, P., Albertini, G., Rossini, P.M.: Resting state eyes-closed cortical rhythms in patients with locked-in-syndrome: An EEG study. Clin. Neurophysiol. **121**(11), 1816–1824 (2010) . https://doi.org/10.1016/j.clinph.2010.04.027
56. Shoorangiz, R., Weddell, S.J., Jones, R.D.: Bayesian multi-subject factor analysis to predict microsleeps from EEG power spectral features. In: Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 39, pp. 4183–4186. IEEE (2017) . https://doi.org/10.1109/EMBC.2017.8037778
57. Lin, F., Ko, L., Chuang, C., Su, T., Lin, C.: Generalized EEG-based drowsiness prediction system by using a self-organizing neural fuzzy system. IEEE Trans. Circuits Syst. I Regular Papers **59**(9), 2044–2055. (2012) https://doi.org/10.1109/TCSI.2012.2185290
58. Sabeti, M., Katebi, S., Boostani, R.: Entropy and complexity measures for EEG signal classification of schizophrenic and control participants. Artif. Intell. Med. **47**(3), 263–274 (2009). https://doi.org/10.1016/j.artmed.2009.03.003
59. Kannathal, N., Choo, M.L., Acharya, U.R., Sadasivan, P.K.: Entropies for detection of epilepsy in EEG. Comput. Meth. Prog. Biomed. **80**(3), 187–194 (2005). https://doi.org/10.1016/j.cmpb.2005.06.012
60. Chen, L.-l., Zhao, Y., Zhang, J., Zou, J.-Z.: Automatic detection of alertness/drowsiness from physiological signals using wavelet-based nonlinear features and machine learning. Expert Syst. Appl. **42**(21), 7344–7355 (2015). https://doi.org/10.1016/j.eswa.2015.05.028
61. Yeo, M.V.M., Li, X., Shen, K., Wilder-Smith, E.P.V.: Can SVM be used for automatic EEG detection of drowsiness during car driving? Safety Sci. **47**(1), 115–124 (2009). https://doi.org/10.1016/j.ssci.2008.01.007
62. Ramoser, H., Muller-Gerking, J., Pfurtscheller, G.: Optimal spatial filtering of single trial EEG during imagined hand movement. IEEE Trans. Rehabilitat. Eng. **8**(4), 441–446 (2000). https://doi.org/10.1109/86.895946
63. Blankertz, B., Tomioka, R., Lemm, S., Kawanabe, M., Muller, K.-R.: Optimizing spatial filters for robust EEG single-trial analysis. IEEE Signal Process. Mag. **25**(1), 41–56 (2008). https://doi.org/10.1109/MSP.2008.4408441
64. Ang, K.K., Chin, Z.Y., Zhang, H., Guan, C.: Filter bank common spatial pattern (FBCSP) in brain-computer interface. In: Proceedings of International Joint Conference on Neural Network, pp. 2390–2397. IEEE, (2008). https://doi.org/10.1109/IJCNN.2008.4634130
65. Ang, K.K., Chin, Z.Y., Wang, C., Guan, C., Zhang, H.: Filter bank common spatial pattern algorithm on BCI competition IV datasets 2a and 2b. Front. Neurosci. **6**, 39 (2012). https://doi.org/10.3389/fnins.2012.00039

66. Lotte, F., Guan, C.: Regularizing common spatial patterns to improve BCI designs: Unified theory and new algorithms. IEEE Trans. Biomed. Eng. **58**(2), 355–362 (2011). https://doi.org/10.1109/TBME.2010.2082539
67. Kang, H., Choi, S.: Bayesian common spatial patterns for multi-subject EEG classification. Neural Netw. **57**(Supplement C), 39–50 (2014). https://doi.org/10.1016/j.neunet.2014.05.012
68. Wu, W., Chen, Z., Gao, X., Li, Y., Brown, E.N., Gao, S.: Probabilistic common spatial patterns for multichannel EEG analysis. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 639–653 (2015). https://doi.org/10.1109/TPAMI.2014.2330598
69. Ang, K.K., Chin, Z.Y., Zhang, H., Guan, C.: Mutual information-based selection of optimal spatial–temporal patterns for single-trial EEG-based BCIs. Pattern Recogn. **45**(6), 2137–2144 (2012). https://doi.org/10.1016/j.patcog.2011.04.018
70. Mwangi, B., Tian, T.S., Soares, J.C.: A review of feature reduction techniques in neuroimaging. Neuroinformatics **12**(2), 229–244 (2014). https://doi.org/10.1007/s12021-013-9204-3
71. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Comput. Electr. Eng. **40**(1), 16–28 (2014) . https://doi.org/10.1016/j.compeleceng.2013.11.024
72. Cai, J., Luo, J., Wang, S., Yang, S.: Feature selection in machine learning: A new perspective. Neurocomputing **300**, 70–79 (2018). https://doi.org/10.1016/j.neucom.2017.11.077
73. Miao, J., Niu, L.: A survey on feature selection. Proc. Comput. Sci. **91**, 919–926 (2016). https://doi.org/10.1016/j.procs.2016.07.111
74. Mitra, P., Murthy, C.A., Pal, S.K.: Unsupervised feature selection using feature similarity. IEEE Trans. Pattern Anal. Mach. Intell. **24**(3), 301–312 (2002). https://doi.org/10.1109/34.990133
75. Lee, J.A., Verleysen, M.: Nonlinear Dimensionality Reduction, 1st edn. Springer Publishing Company, Incorporated (2007)
76. Singh, D.A.A.G., Balamurugan, S.A.A., Leavline, E.J.: An unsupervised feature selection algorithm with feature ranking for maximizing performance of the classifiers. Int. J. Automat. Comput. **12**(5), 511–517 (2015) . https://doi.org/10.1007/s11633-014-0859-5
77. Subasi, A., Gursoy, M.I.: EEG signal classification using PCA, ICA, LDA and support vector machines. Expert Syst. Appl. **37**(12), 8659–8666 (2010). https://doi.org/10.1016/j.eswa.2010.06.065
78. ai Li, M., Yong Luo, X., fu Yang, J.: Extracting the nonlinear features of motor imagery EEG using parametric t-SNE. Neurocomputing **218**, 371–381 (2016). https://doi.org/10.1016/j.neucom.2016.08.083
79. Birjandtalab, J., Pouyan, M.B., Cogan, D., Nourani, M., Harvey, J.: Automated seizure detection using limited-channel EEG and non-linear dimension reduction. Comput. Biol. Med. **82**, 49–58 (2017). https://doi.org/10.1016/j.compbiomed.2017.01.011
80. Yu, X., Chum, P., Sim, K.-B.: Analysis the effect of PCA for feature reduction in non-stationary EEG based motor imagery of BCI system. Optik **125**(3), 1498–1502 (2014). https://doi.org/10.1016/j.ijleo.2013.09.013
81. Schölkopf, B., Smola, A., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. **10**(5), 1299–1319 (1998)
82. Tipping, M.E., Bishop, C.M.: Probabilistic principal component analysis. J. R. Stat. Soc. Ser. B (Statistical Methodology) **61**(3), 611–622 (1999). https://doi.org/10.1111/1467-9868.00196
83. Bishop, C.M.: Variational principal components. In: Proceedings of International Conference on Artificial Neural Networks, Vol. 9, pp. 509–514. Institution of Engineering and Technology (1999). https://doi.org/10.1049/cp_19991160
84. Ding, X., He, L., Carin, L.: Bayesian robust principal component analysis. IEEE Trans. Image Process. **20**(12), 3419–3430 (2011). https://doi.org/10.1109/TIP.2011.2156801
85. Guyon, I.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
86. Kwak, N., Choi, C.-H.: Input feature selection by mutual information based on Parzen window. IEEE Trans. Pattern Anal. Mach. Intell. **24**(12), 1667–1671 (2002). https://doi.org/10.1109/TPAMI.2002.1114861

87. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE Trans. Neural Netw. **5**(4), 537–550 (1994). https://doi.org/10.1109/72.298224
88. Arvaneh, M., Guan, C., Ang, K., Quek, C.: Mutual information-based optimization of sparse spatio-spectral filters in brain-computer interface. Neural Comput. Appl. **25**(3–4), 625–634 (2014). https://doi.org/10.1007/s00521-013-1523-7
89. Yang, J., Zhang, L., Xu, Y., Yang, J.-Y.: Beyond sparsity: the role of L1-optimizer in pattern classification. Pattern Recogn. **45**(3), 1104–1118 (2012). https://doi.org/10.1016/j.patcog.2011.08.022
90. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B (Methodological) **58**(1), 267–288 (1996)
91. Liu, M., Zhang, D., Shen, D.: Ensemble sparse classification of Alzheimer's disease. NeuroImage **60**(2), 1106–1116 (2012). https://doi.org/10.1016/j.neuroimage.2012.01.055
92. Fell, J., Röschke, J., Mann, K., Schäffner, C.: Discrimination of sleep stages: A comparison between spectral and nonlinear EEG measures. Electroencephalography Clin. Neurophysiol. **98**(5), 401–410 (1996). https://doi.org/10.1016/0013-4694(96)95636-9
93. Lin, Y., Wang, C., Jung, T., Wu, T., Jeng, S., Duann, J., Chen, J.: EEG-based emotion recognition in music listening. IEEE Trans. Biomed. Eng. **57**(7), 1798–1806 (2010). https://doi.org/10.1109/TBME.2010.2048568
94. Schlögl, A., Lee, F., Bischof, H., Pfurtscheller, G.: Characterization of four-class motor imagery EEG data for the BCI-competition 2005. J. Neural Eng. **2**(4), L14–L22 (2005). https://doi.org/10.1088/1741-2560/2/4/l02
95. Quitadamo, L.R., Cavrini, F., Sbernini, L., Riillo, F., Bianchi, L., Seri, S., Saggio, G.: Support vector machines to detect physiological patterns for EEG and EMG-based human–computer interaction: A review. J. Neural Eng. **14**(1), 011001 (2017)
96. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. IEEE Intell. Syst. Appl. **13**(4), 18–28 (1998). https://doi.org/10.1109/5254.708428
97. Schölkopf, B.: The kernel trick for distances. In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) Advances in Neural Information Processing Systems, pp. 301–307. MIT Press (2001)
98. Roth, V., Steinhage, V.: Nonlinear discriminant analysis using kernel functions. In: Solla, S.A., Leen, T.K., Müller, K. (eds.) Advances in Neural Information Processing Systems, pp. 568–574. MIT Press (2000)
99. Nicolaou, N., Georgiou, J.: Detection of epileptic electroencephalogram based on permutation entropy and support vector machines. Expert Syst. Appl. **39**(1), 202–209 (2012). https://doi.org/10.1016/j.eswa.2011.07.008
100. Tito, M., Cabrerizo, M., Ayala, M., Jayakar, P., Adjouadi, M.: A comparative study of intracranial EEG files using nonlinear classification methods. Ann. Biomed. Eng. **38**(1), 187–99 (2010)
101. Lawrence, N.D., Schölkopf, B.: Estimating a kernel Fisher discriminant in the presence of label noise. In: Proceedings of International Conference on Machine Learning, ICML '01, Vol. 18, pp. 306–313. ACM (2001)
102. Kayikcioglu, T., Aydemir, O.: A polynomial fitting and k-NN based approach for improving classification of motor imagery BCI data. Pattern Recogn. Lett. **31**(11), 1207–1215 (2010). https://doi.org/10.1016/j.patrec.2010.04.009
103. Safavian, S.R., Landgrebe, D.: A survey of decision tree classifier methodology. IEEE Trans. Syst. Man Cybern. **21**(3), 660–674 (1991). https://doi.org/10.1109/21.97458
104. Polat, K., Güneş, S.: Classification of epileptiform EEG using a hybrid system based on decision tree classifier and fast Fourier transform. Appl. Math. Comput. **187**(2), 1017–1026 (2007). https://doi.org/10.1016/j.amc.2006.09.022
105. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
106. Pandya, R., Pandya, J.: C5.0 algorithm to improved decision tree with feature selection and reduced error pruning. Int. J. Comput. Appl. **117**(16), 18–21 (2015)

107. Jin, C., De-lin, L., Fen-Xiang, M.: An improved ID3 decision tree algorithm. In: Proceedings of International Conference on Computer Science and Education, Vol. 4, pp. 127–130. IEEE (2009). https://doi.org/10.1109/ICCSE.2009.5228509
108. Nisbet, R., Miner, G., Yale, K.: Chapter 8 - Advanced algorithms for data mining. In: Nisbet, R., Miner, G., Yale, K. (eds.) Handbook of Statistical Analysis and Data Mining Applications, 2nd edn., pp. 149–167. Academic Press, Boston (2018)
109. Sun, S., Zhang, C., Zhang, D.: An experimental evaluation of ensemble methods for EEG signal classification. Pattern Recogn. Lett. **28**(15), 2157–2163 (2007). https://doi.org/10.1016/j.patrec.2007.06.018
110. Peiris, M.T.R., Davidson, P.R., Bones, P.J., Jones, R.D.: Detection of lapses in responsiveness from the EEG. J. Neural Eng. **8**(1), 016003 (2011)
111. Hassan, A.R., Subasi, A.: Automatic identification of epileptic seizures from EEG signals using linear programming boosting. Comput. Meth. Prog. Biomed. **136**, 65–77 (2016). https://doi.org/10.1016/j.cmpb.2016.08.013
112. Zhou, Z.-H.: Ensemble Methods: Foundations and Algorithms. Chapman and Hall/CRC (2012)
113. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**(1), 119–139 (1997). https://doi.org/10.1006/jcss.1997.1504
114. Tahernezhad-Javazm, F., Azimirad, V., Shoaran, M.: A review and experimental study on the application of classifiers and evolutionary algorithms in EEG-based brain–machine interface systems. J. Neural Eng. **15**(2), 021007 (2018). https://doi.org/10.1088/1741-2552/aa8063
115. Frenay, B., Verleysen, M.: Classification in the presence of label noise: A survey. IEEE Trans. Neural Netw. Learn. Syst. **25**(5), 845–869 (2014). https://doi.org/10.1109/TNNLS.2013.2292894
116. Verbaeten, S., Van Assche, A.: Ensemble methods for noise elimination in classification problems. In: Windeatt, T., Roli, F. (eds.) Lecture Notes in Computer Science, pp. 317–325. Springer, Berlin/Heidelberg (2003)
117. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996). https://doi.org/10.1007/BF00058655
118. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
119. Fraiwan, L., Lweesy, K., Khasawneh, N., Wenz, H., Dickhaus, H.: Automated sleep stage identification system based on time–frequency analysis of a single EEG channel and random forest classifier. Comput. Meth. Prog. Biomed. **108**(1), 10–19 (2012). https://doi.org/10.1016/j.cmpb.2011.11.005
120. Shen, K., Ong, C., Li, X., Hui, Z., Wilder-Smith, E.P.V.: A feature selection method for multilevel mental fatigue EEG classification. IEEE Trans. Biomed. Eng. **54**(7), 1231–1237 (2007). https://doi.org/10.1109/TBME.2007.890733
121. Tibshirani, R.: Regression shrinkage and selection via the lasso: A retrospective. J. R. Stat. Soc. Ser. B (Statistical Methodology) **73**(3), 273–282 (2011). https://doi.org/10.1111/j.1467-9868.2011.00771.x
122. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. J. R. Stat. Soc. Ser. B (Statistical Methodology) **67**(2), 301–320 (2005)
123. Bunea, F., She, Y., Ombao, H., Gongvatana, A., Devlin, K., Cohen, R.: Penalized least squares regression methods and applications to neuroimaging. NeuroImage **55**(4), 1519–1527 (2011). https://doi.org/10.1016/j.neuroimage.2010.12.028
124. Carroll, M.K., Cecchi, G.A., Rish, I., Garg, R., Rao, A.R.: Prediction and interpretation of distributed neural activity with sparse models. NeuroImage **44**(1), 112–122 (2009). https://doi.org/10.1016/j.neuroimage.2008.08.020
125. Ryali, S., Supekar, K., Abrams, D.A., Menon, V.: Sparse logistic regression for whole-brain classification of fMRI data. NeuroImage **51**(2), 752–764 (2010). https://doi.org/10.1016/j.neuroimage.2010.02.040

126. Mohr, H., Wolfensteller, U., Frimmel, S., Ruge, H.: Sparse regularization techniques provide novel insights into outcome integration processes. NeuroImage **104**, 163–176 (2015). https://doi.org/10.1016/j.neuroimage.2014.10.025
127. Li, F., Liu, T., Wang, F., Li, H., Gong, D., Zhang, R., Jiang, Y., Tian, Y., Guo, D., Yao, D., Xu, P.: Relationships between the resting-state network and the P3: Evidence from a scalp EEG study. Sci. Rep. **5**, 15129 (2015). https://doi.org/10.1038/srep15129
128. Zhao, Q., Peng, H., Hu, B., Liu, Q., Liu, L., Qi, Y., Li, L.: Improving individual identification in security check with an EEG based biometric solution. In: Yao, Y., Sun, R., Poggio, T., Liu, J., Zhong, N., Huang, J. (eds.) Proceedings of International Conference on Brain Information, pp. 145–155. Springer, Berlin/Heidelberg (2010). https://doi.org/10.1007/978-3-642-15314-3_14
129. DelPozo-Banos, M., Travieso, C.M., Weidemann, C.T., Alonso, J.B.: EEG biometric identification: A thorough exploration of the time-frequency domain. J. Neural Eng. **12**(5), 056019 (2015)
130. Jatupaiboon, N., Pan-ngum, S., Israsena, P.: Real-time EEG-based happiness detection system. Sci. World J. **2013**, 12 (2013). https://doi.org/10.1155/2013/618649
131. Handiru, V.S., Prasad, V.A.: Optimized bi-objective EEG channel selection and cross-subject generalization with brain–computer interfaces. IEEE Trans. Human Mach. Syst. **46**(6), 777–786 (2016). https://doi.org/10.1109/THMS.2016.2573827
132. Berrar, D.: Performance measures for binary classification. In: Ranganathan, S., Gribskov, M., Nakai, K., Schönbach, C. (eds.) Encyclopedia of Bioinformatics and Computational Biology, pp. 546–560. Academic Press, Oxford (2019)
133. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009). https://doi.org/10.1109/TKDE.2008.239
134. Tharwat, A.: Classification Assessment Methods, Applied Computing and Informatics (in press). https://doi.org/10.1016/j.aci.2018.08.003
135. Sun, Y., Wong, A.K.C., Kamel, M.S.: Classification of imbalanced data: A review. Int. J. Pattern Recogn. Artif. Intell. **23**(04), 687–719 (2009). https://doi.org/10.1142/S0218001409007326
136. Chawla, N.V.: Data mining for imbalanced datasets: An overview. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, pp. 875–886. Springer, Boston (2010)
137. Boughorbel, S., Jarray, F., El-Anbari, M.: Optimal classifier for imbalanced data using Matthews correlation coefficient metric. PLoS ONE **12**(6), 1–17 (2017). https://doi.org/10.1371/journal.pone.0177678
138. López, V., Fernández, A. García, S., Palade, V., Herrera, F.: An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Inf. Sci. **250**, 113–141 (2013). https://doi.org/10.1016/j.ins.2013.07.007
139. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. **45**(4), 427–437 (2009). https://doi.org/10.1016/j.ipm.2009.03.002
140. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006). https://doi.org/10.1016/j.patrec.2005.10.010
141. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: Proceedings of International Conference on Machine Learning, ICML '06, Vol. 23, pp. 233–240. ACM, New York (2006). https://doi.org/10.1145/1143844.1143874
142. Saito, T., Rehmsmeier, M.: The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS ONE **10**(3), e0118432 (2015). https://doi.org/10.1371/journal.pone.0118432
143. Vihinen, M.: How to evaluate performance of prediction methods? Measures and their interpretation in variation effect analysis. BMC Genom. **13**(4), S2 (2012). https://doi.org/10.1186/1471-2164-13-S4-S2
144. Vihinen, M.: Guidelines for reporting and using prediction tools for genetic variation analysis. Hum. Mutat. **34**(2), 275–282 (2013). https://doi.org/10.1002/humu.22253

145. Lehmann, C., Koenig, T., Jelic, V., Prichep, L., John, R.E., Wahlund, L.-O., Dodge, Y., Dierks, T.: Application and comparison of classification algorithms for recognition of Alzheimer's disease in electrical brain activity (EEG). J. Neurosci. Meth. **161**(2), 342–350 (2007). https://doi.org/10.1016/j.jneumeth.2006.10.023
146. Oh, S.L., Hagiwara, Y., Raghavendra, U., Yuvaraj, R., Arunkumar, N., Murugappan, M., Acharya, U.R.: A deep learning approach for Parkinson's disease diagnosis from EEG signals. Neural Comput. Appl. (2018). https://doi.org/10.1007/s00521-018-3689-5
147. Ieracitano, C., Mammone, N., Bramanti, A., Hussain, A., Morabito, F.C.: A convolutional neural network approach for classification of dementia stages based on 2D-spectral representation of EEG recordings. Neurocomputing **323**, 96–107 (2019). https://doi.org/10.1016/j.neucom.2018.09.071
148. Nourmohammadi, A., Jafari, M., Zander, T.O.: A survey on unmanned aerial vehicle remote control using brain–computer interface. IEEE Trans. Hum. Mach. Syst. **48**(4), 337–348 (2018). https://doi.org/10.1109/THMS.2018.2830647
149. Nguyen-Ky, T., Wen, P., Li, Y.: Consciousness and depth of anesthesia assessment based on Bayesian analysis of EEG signals. IEEE Trans. Biomed. Eng. **60**(6), 1488–1498 (2013). https://doi.org/10.1109/TBME.2012.2236649
150. Shoorangiz, R., Weddell, S.J., Jones, R.D.: Prediction of microsleeps from EEG: Preliminary results. In: Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 38, pp. 4650–4653. IEEE (2016). https://doi.org/10.1109/EMBC.2016.7591764
151. van Gerven, M. Farquhar, J., Schaefer, R., Vlek, R., Geuze, J., Nijholt, A., Ramsey, N., Haselager, P., Vuurpijl, L., Gielen, S., Desain, P.: The brain–computer interface cycle. J. Neural Eng. **6**(4), 041001 (2009)
152. Hwang, H.-J., Kim, S., Choi, S., Im, C.-H.: EEG-based brain-computer interfaces: A thorough literature survey. Int. J. Hum. Comput. Interact. **29**(12), 814–826 (2013). https://doi.org/10.1080/10447318.2013.780869
153. Vaughan, T.M.: Guest editorial brain-computer interface technology: A review of the second international meeting. IEEE Trans. Neural Syst. Rehabilitat. Eng. **11**(2), 94–109 (2003). https://doi.org/10.1109/TNSRE.2003.814799
154. Zhang, S., Wang, S., Zheng, D., Zhu, K., Dai, M.: A novel pattern with high-level commands for encoding motor imagery-based brain computer interface. Pattern Recogn. Lett. **125**, 28–34 (2019). https://doi.org/10.1016/j.patrec.2019.03.017
155. Li, M.-A., Wang, Y.-F., Jia, S.-M., Sun, Y.-J., Yang, J.-F.: Decoding of motor imagery EEG based on brain source estimation. Neurocomputing **339**, 182–193 (2019). https://doi.org/10.1016/j.neucom.2019.02.006
156. Xu, Y., Ding, C., Shu, X., Gui, K., Bezsudnova, Y., Sheng, X., Zhang, D.: Shared control of a robotic arm using non-invasive brain–computer interface and computer vision guidance. Robot. Autonom. Syst. **115**, 121–129 (2019). https://doi.org/10.1016/j.robot.2019.02.014
157. Higashi, H., Tanaka, T.: Simultaneous design of FIR filter banks and spatial patterns for EEG signal classification. IEEE Trans. Biomed. Eng. **60**(4), 1100–1110 (2013). https://doi.org/10.1109/TBME.2012.2215960
158. Kevric, J., Subasi, A.: Comparison of signal decomposition methods in classification of EEG signals for motor-imagery BCI system. Biomed. Signal Process. Control **31**, 398–406 (2017). https://doi.org/10.1016/j.bspc.2016.09.007
159. Hortal, E., Planelles, D., Costa, A., Iáñez, E., Úbeda, A., Azorín, J.M., Fernández, E.: SVM-based brain–machine interface for controlling a robot arm through four mental tasks. Neurocomputing **151**, 116–121 (2015). https://doi.org/10.1016/j.neucom.2014.09.078
160. Tabar, Y.R., Halici, U.: A novel deep learning approach for classification of EEG motor imagery signals. J. Neural Eng. **14**(1), 016003 (2016). https://doi.org/10.1088/1741-2560/14/1/016003
161. Li, J., Struzik, Z., Zhang, L., Cichocki, A.: Feature learning from incomplete EEG with denoising autoencoder. Neurocomputing **165**, 23–31 (2015). https://doi.org/10.1016/j.neucom.2014.08.092

162. Raza, H., Rathee, D., Zhou, S.-M., Cecotti, H., Prasad, G.: Covariate shift estimation based adaptive ensemble learning for handling non-stationarity in motor imagery related EEG-based brain-computer interface. Neurocomputing **343**, 154–166 (2019). https://doi.org/10.1016/j.neucom.2018.04.087
163. Schirrmeister, R.T., Springenberg, J.T., Fiederer, L.D.J., Glasstetter, M., Eggensperger, K., Tangermann, M., Hutter, F., Burgard, W., Ball, T.: Deep learning with convolutional neural networks for EEG decoding and visualization. Hum. Brain Mapp. **38**(11), 5391–5420 (2017). https://doi.org/10.1002/hbm.23730
164. Farwell, L.A., Donchin, E.: Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials. Electroencephalography Clin. Neurophysiol. **70**(6), 510–523 (1988). https://doi.org/10.1016/0013-4694(88)90149-6
165. Liu, M., Wu, W., Gu, Z., Yu, Z., Qi, F., Li, Y.: Deep learning based on batch normalization for P300 signal detection. Neurocomputing **275**, 288–297 (2018). https://doi.org/10.1016/j.neucom.2017.08.039
166. Tomioka, R., M uller, K.-R.: A regularized discriminative framework for EEG analysis with application to brain–computer interface. NeuroImage **49**(1), 415–432 (2010). https://doi.org/10.1016/j.neuroimage.2009.07.045
167. Krusienski, D.J., Sellers, E.W., McFarland, D.J., Vaughan, T.M., Wolpaw, J.R.: Toward enhanced P300 speller performance. J Neurosci. Meth. **167**(1), 15–21 (2008). https://doi.org/10.1016/j.jneumeth.2007.07.017
168. Chen, Y., Ke, Y., Meng, G., Jiang, J., Qi, H., Jiao, X., Xu, M., Zhou, P., He, F., Ming, D.: Enhancing performance of P300-speller under mental workload by incorporating dual-task data during classifier training. Comput. Meth. Progr. Biomed. **152**, 35–43 (2017). https://doi.org/10.1016/j.cmpb.2017.09.002
169. Salvaris, M., Sepulveda, F.: Visual modifications on the P300 speller BCI paradigm. J. Neural Eng. **6**(4), 046011 (2009). https://doi.org/10.1088/1741-2560/6/4/046011
170. Lawhern, V.J., Solon, A.J., Waytowich, N.R., Gordon, S.M., Hung, C.P., Lance, B.J.: EEGNet: A compact convolutional neural network for EEG-based brain–computer interfaces. J. Neural Eng. **15**(5), 056013 (2018). https://doi.org/10.1088/1741-2552/aace8c
171. Li, Q., Shi, K., Ma, S., Gao, N.: Improving classification accuracy of SVM ensemble using random training set for BCI P300-speller. In: IEEE International Conference on Mechatronics and Automation, pp. 2611–2616. IEEE (2016). https://doi.org/10.1109/ICMA.2016.7558978
172. Lee, Y.-R., Kim, H.-N.: A data partitioning method for increasing ensemble diversity of an eSVM-based P300 speller. Biomed. Signal Process. Control **39**, 53–63 (2018). https://doi.org/10.1016/j.bspc.2017.07.025
173. Akram, F., Han, S.M., Kim, T.-S.: An efficient word typing P300-BCI system using a modified T9 interface and random forest classifier. Comput. Biol. Med. **56**, 30–36 (2015). https://doi.org/10.1016/j.compbiomed.2014.10.021
174. Zhu, D., Bieger, J., Garcia Molina, G. Aarts, R.M.: A survey of stimulation methods used in SSVEP-based BCIs. Comput. Intell. Neurosci. **2010**, 12 (2010). https://doi.org/10.1155/2010/702357
175. Wang, M., Daly, I., Allison, B.Z., Jin, J., Zhang, Y., Chen, L., Wang, X.: A new hybrid BCI paradigm based on P300 and SSVEP. J. Neurosci. Meth. **244**, 16–25 (2015). https://doi.org/10.1016/j.jneumeth.2014.06.003
176. Fomina, T., Lohmann, G., Erb, M., Ethofer, T., Schölkopf, B., Grosse-Wentrup, M.: Self-regulation of brain rhythms in the precuneus: A novel BCI paradigm for patients with ALS. J. Neural Eng. **13**(6), 066021 (2016). https://doi.org/10.1088/1741-2560/13/6/066021
177. Zhang, N., Zhou, Z., Liu, Y., Yin, E., Jiang, J., Hu, D.: A novel single-character visual BCI paradigm with multiple active cognitive tasks. IEEE Trans. Biomed. Eng. (in press). https://doi.org/10.1109/TBME.2019.2900555
178. Kaongoen, N., Jo, S.: A novel hybrid auditory BCI paradigm combining ASSR and P300. J. Neurosci. Meth. **279**, 44–51 (2017). https://doi.org/10.1016/j.jneumeth.2017.01.011
179. Jones, R.D., Poudel, G.R., Innes, C.R.H., Davidson, P.R., Peiris, M.T.R., Malla, A.M., Signal, T.L., Carroll, G.J., Watts, R., Bones, P.J.: Lapses of responsiveness: Characteristics,

detection, and underlying mechanisms. In: Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 32, pp. 1788–1791. IEEE (2010). https://doi.org/10.1109/IEMBS.2010.5626385
180. Peiris, M.T.R. Jones, R.D., Davidson, P.R., Carroll, G.J., Bones, P.J.: Frequent lapses of responsiveness during an extended visuomotor tracking task in non-sleep-deprived subjects. J. Sleep Res. **15**(3), 291–300 (2006). https://doi.org/10.1111/j.1365-2869.2006.00545.x
181. Innes, C.R.H., Poudel, G.R., Jones, R.D.: Efficient and regular patterns of nighttime sleep are related to increased vulnerability to microsleeps following a single night of sleep restriction. Chronobiol. Int. **30**(9), 1187–1196 (2013). https://doi.org/10.3109/07420528.2013.810222
182. Poudel, G.R., Innes, C.R.H., Bones, P.J., Watts, R., Jones, R.D.: Losing the struggle to stay awake: Divergent thalamic and cortical activity during microsleeps. Hum. Brain Mapp. **35**(1), 257–269 (2014). https://doi.org/10.1002/hbm.22178
183. Golz, M., Sommer, D., Chen, M., Trutschel, U., Mandic, D.: Feature fusion for the detection of microsleep events. J. VLSI Signal Process. Syst. Signal Image Video Technol. **49**(2), 329–342 (2007). https://doi.org/10.1007/s11265-007-0083-4
184. Ayyagari, S.: Reservoir computing approaches to EEG-based detection of microsleeps Ph.D. Thesis (University of Canterbury, Christchurch, New Zealand (2017)
185. Golz, M., Sommer, D., Krajewski, J.: Prediction of immediately occurring microsleep events from brain electric signals. Curr. Directions Biomed. Eng. **2**(1), 149–153 (2016). https://doi.org/10.1515/cdbme-2016-0035
186. Baseer, A., Weddell, S.J., Jones, R.D.: Prediction of microsleeps using pairwise joint entropy and mutual information between EEG channels. In: Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 39, pp. 4495–4498. IEEE (2017). https://doi.org/10.1109/EMBC.2017.8037855
187. Buriro, A.B., Shoorangiz, R., Weddell, S.J., Jones, R.D.: Ensemble learning based on overlapping clusters of subjects to predict microsleep states from EEG. In: Proceedings of International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 40, pp. 3036–3039. IEEE (2018). https://doi.org/10.1109/EMBC.2018.8512962
188. Shoorangiz, R.: Prediction of microsleeps from EEG using Bayesian approaches Ph.D. Thesis. University of Canterbury, Christchurch, New Zealand (2018)