Introduction to Machine Learning for Brain Imaging

Steven Lemm^{a,c,*}, Benjamin Blankertz^{a,b,c}, Thorsten Dickhaus^a, Klaus-Robert Müller^{a,c}

^aBerlin Institute of Technology, Dept. of Computer Science, Machine Learning Group, Berlin, Germany ^bFraunhofer FIRST, Intelligent Data Analysis Group, Berlin, Germany ^cBernstein Focus: Neurotechnology, Berlin, Germany

Abstract

Machine learning and pattern recognition algorithms have in the past years developed to become a working horse in brain imaging and the computational neurosciences, as they are instrumental for mining vast amounts of neural data of ever increasing measurement precision and detecting minuscule signals from an overwhelming noise floor. They provide the means to decode and characterize task relevant brain states and to distinguish them from non-informative brain signals. While undoubtedly this machinery has helped to gain novel biological insights, it also holds the danger of potential unintentional abuse. Ideally machine learning techniques should be usable for any non-expert, however, unfortunately they are typically not. Overfitting and other pitfalls may occur and lead to spurious and nonsensical interpretation. The goal of this review is therefore to provide an accessible and clear introduction to the strengths and also the inherent dangers of machine learning usage in the neurosciences.

Keywords: pattern recognition, machine learning, model selection, cross validation

1. Introduction

The past years have seen an immense rise of interest in the decoding of brain states as measured invasively with multi-unit arrays and electrocorticography (ECoG) or non-invasively with functional Magnetic Resonance Imaging (fMRI), electroencephalography (EEG), or Near-infrared spectroscopy (NIRS). Instrumental to this development has been the use of modern machine learning and pattern recognition algorithms (e.g. Bishop (1995); Vapnik (1995); Duda et al. (2001); Hastie et al. (2001); Müller et al. (2001); Schölkopf and Smola (2002); Rasmussen and Williams (2005)). Clearly, in this development the field of realtime decoding for brain-computer interfacing (BCI) has been a technology motor (e.g. Dornhege et al. (2007); Kübler and Kotchoubey (2007); Kübler and Müller (2007); Wolpaw (2007); Birbaumer (2006); Pfurtscheller et al. (2005); Curran and Stokes (2003); Wolpaw et al. (2002); Kübler et al. (2001)). As a result a number of novel data driven approaches specific to neuroscience have emerged: (a) dimension reduction and projection methods (e.g. Hyvärinen et al. (2001); Ziehe et al. (2000); Parra and Sajda (2003);

Morup et al. (2008); von Bünau et al. (2009); Blanchard et al. (2006); Bießmann et al. (2009)), (b) classification methods (e.g. Müller et al. (2001); Müller et al. (2003); Parra et al. (2002, 2003); Lal et al. (2004); Blankertz et al. (2006a, 2008b); Müller et al. (2008); Parra et al. (2008); Tomioka et al. (2007); Tomioka and Müller (2010)), (c) spatio-temporal filtering algorithms (e.g. Fukunaga (1972); Koles (1991); Ramoser et al. (2000); Lemm et al. (2005); Blankertz et al. (2008b); Dornhege et al. (2006); Tomioka and Müller (2010); Parra et al. (2008); Dornhege et al. (2007); Blankertz et al. (2007)), (d) measures for determining synchrony, coherence or causal relations in data (e.g. Meinecke et al. (2005); Brunner et al. (2006); Nolte et al. (2004); Marzetti et al. (2008); Nolte et al. (2008, 2006); Gentili et al. (2009)) and (e) algorithms for assessing and counteracting non-stationarity in data (e.g. Shenoy et al. (2006); Sugiyama et al. (2007); von Bünau et al. (2009); Blankertz et al. (2008a); Krauledat et al. (2007)). Moreover a new generation of source localization techniques is now in use (e.g. Haufe et al. (2008, 2009)) and has been successfully applied in BCI research (e.g. Pun et al. (2006); Noirhomme et al. (2008); Grosse-Wentrup et al. (2007)).

Despite of this multitude of powerful novel data analytical methods, this review will place its focus mainly

^{*}corresponding author: steven.lemm@cs.tu-berlin.de

Preprint submitted to NeuroImage

on the essentials for decoding brain states, namely we will introduce the mathematical concept of classification and model selection, and explicate how to properly apply them to neurophysiological data. To this end, the paper first reviews the mathematical and algorithmic principles starting from a rather abstract level and regardless of the scope of application. The subsequent discussion of the practicalities and 'tricks' in the spirit of Orr and Müller (1998) will then link to the analysis of brain imaging data. Unlike existing reviews on this topic (cf. Wolpaw et al. (2002); Kübler and Müller (2007); Dornhege et al. (2007); Haynes and Rees (2006); Pereira et al. (2009)), we will elaborate on common pitfalls and how to avoid them when applying machine learning to brain imaging data.

A final note: while the paper introduces the main ideas of algorithms, we do not attempt a full treatment of the available literature. Rather, we present a somewhat biased view, mainly drawing from the author's work and providing – to the best of our knowledge – links and reference to related studies and further reading. We sincerely hope that it will nevertheless be useful to the reader.

The paper outline is the following: After a brief introduction, we discuss machine learning principles in Section 2 and then, Section 3 provides feature extraction and dimensionality reduction methods, both supervised and unsupervised. The main contributions of this review are a detailed account of how to validate and select models (Section 4), and the elaboration of the practical aspects of model evaluation and most common pitfalls specific to brain imaging data in Section 5.

2. Learning to classify

Neuroscientific experiments often aim at contrasting specific brain states. Typically, the experimenter chooses a neurophysiological paradigm that maximizes the contrast between the brain states (note that there may be more than two states), while controlling for task unrelated processes. After recording brain imaging data, the goal of analysis is to find significant differences in the spatial and the temporal characteristics of the data contrasting the different states as accurate as possible. While simple statistics such as considering, e.g., grand averages (averaging across trials and subjects) may help for model building, more sophisticated machine learning techniques have become increasingly popular due to their great modeling power. Note that the methods we will presented in this paper are likewise applicable to multivariate fMRI voxel time series, to single trial responses in fMRI or EEG, and brain imaging data from any other spatio-temporal, or spectral domain. Formally, the scenario of discrimination of brain



Figure 1: Left: Typical work flow for machine learning based classification of brain imaging data. First, brain imaging data are acquired according to the chosen neurophysiological paradigm. Then the data are preprocessed, e.g., by artifact rejection or bandpass filtering. The machine learning based approach comprises the reduction of the dimensionality by extraction of meaningful features and the final classification of the data in the feature space. Right: example of a linear classification of time series in the time frequency domain. Especially, the linear classifier partitions an appropriately chosen low dimensional feature space.

states can be cast into a so-called classification problem, where in a data-driven manner a classifier is computed that partitions a set of observations into subsets with distinct statistical characteristics (see Fig. 1). Note, however, that not only paradigms with known physiological connotation but also novel scenarios can be scrutinized, where (1) a new paradigm can be explored with respect to its neurophysiological signatures, (2) a hypothesis about underlying task relevant brain processes is generated automatically by the learning machine. Feeding this information back to the experimenter, may lead to a refinement of the initial paradigm, such that, in principle, a better understanding of the brain processes may follow. In this sense, machine learning may not only be the technology of choice for a generic modeling of a neuroscientific experiment, it can also be of great use in a semi-automatic exploration loop for testing new neurophysiological paradigms.

2.1. Some theoretical background

Let us start with the general notion of the learning problems that we consider in this paper. A classifier is a function which partitions a set of objects into several classes, for instance, recordings of brain activity during either auditory, visual, or cognitive processing into the three distinct modalities (classes) 1 . Thus, based on a set of observations, the machine learning task of classification is to find a rule, which assigns an observation x to one of several classes. Here, x denotes a vector of N-dimensional neuroimaging data. In the simplest case there are only two different classes. Hence, a classifier can be formalized as a decision function $f: \mathbb{R}^N \to \{-1, +1\}$, that assigns an observation x to one of the classes, denoted by -1 and 1, respectively. Typically, the set of possible decision functions f is constrained by the scientist to a (parameterized) class of functions \mathcal{F} , e.g., to the class of linear functions. Note, a linear decision function f corresponds to a separating hyperplane (e.g., see Fig. 1), that is parameterized by its normal vector \mathbf{w} and a bias term b. Here, the label y is predicted through

(2.1)
$$y = f(\mathbf{x}; \mathbf{w}, b) = \operatorname{sgn}(\mathbf{w}^{\mathsf{T}}\mathbf{x} + b).$$

Then, based on a set of observed input-output relation $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \in \mathbb{R}^N \times \{-1, +1\}$, learning can be formally described as the task of selecting the parameter value (\mathbf{w}, b) and hence selecting the decision function $f \in \mathcal{F}$ such that f will correctly classify unseen examples x. Here, the observed data (x_i, y_i) is assumed to be independently and identically distributed (i.i.d.) according to an unknown distribution P(x, y) that reflects the relationship between the objects x and the class labels y, e.g., between the recorded brain activity and the paradigm specific mental states. However, in order to find the *optimal* decision function one needs to specify a suitable loss function that evaluates the goodness of the partitioning. One of the most commonly used loss functions for classification is the so-called 0/1-loss (see Smola and Schölkopf (1998) for a discussion of other loss functions)

(2.2)
$$l(y, f(\mathbf{x})) = \begin{cases} 0 & y = f(\mathbf{x}) \\ 1 & else. \end{cases}$$

Given a particular loss function, the best decision function f one can obtain, is the one minimizing the expected risk (also often called *generalization error*)

(2.3)
$$R[f] = \int l(y, f(\boldsymbol{x})) \, \mathrm{d} \mathbf{P}(\boldsymbol{x}, y)$$

under the unknown distribution P(x, y). As the underlying probability distribution P(x, y) is unknown the expected risk cannot be minimized directly. Therefore, we

have to try to estimate the minimum of (2.3) based on the information available, such as the training sample and properties of the function class \mathcal{F} . A straightforward approach is to approximate the risk in (2.3) by the *empirical risk*, i.e., the averaged loss on the training sample

(2.4)
$$R_{emp}[f] = \frac{1}{n} \sum_{i=1}^{n} l(y_i, f(\boldsymbol{x}_i)),$$

and minimize the empirical risk with respect to f. It is possible to give conditions on the learning machine which ensure that asymptotically (as the number of observations $n \to \infty$) the minimum of the empirical risk will converge towards the one of the expected risk. Consequently, with an infinite amount of data the decision function f that minimizes the empirical risk will also minimize the expected risk. However, for small sample sizes this approximation is rather coarse and large deviations are possible. As a consequence of this, *overfitting* might occur, where the decision function f learns details of the sample rather than global properties of P(\mathbf{x} , y) (see Fig. 2 and 4).



Figure 2: Illustration of the overfitting dilemma: Given only a small sample (left) either, the solid or the dashed hypothesis might be true, the dashed one being more complex, but also having a smaller empirical error. Only with a large sample we are able to see which decision reflects the true distribution more closely. If the dashed hypothesis is correct the solid would underfit (middle); if the solid were correct the dashed hypothesis would overfit (right). From Müller et al. (2001).

Under such circumstances, simply minimizing the empirical error (2.4) will not yield a small generalization error in general. One way to avoid the overfitting dilemma is to *restrict* the complexity of the function f Vapnik (1995). The intuition, which will be formalized in the following is that a "simple" (e.g., linear) function that explains most of the data is preferable over a complex one (Occam's razor, cf. MacKay (2003)). This is typically realized by adding a *regularization* term (e.g., Kimeldorf and Wahba (1971); Tikhonov and Arsenin (1977); Poggio and Girosi (1990); Cox and O'Sullivan (1990)) to the empirical risk, i.e.,

(2.5)
$$R_{reg}[f] = R_{emp} + \lambda ||Tf||^2$$

Here, an appropriately chosen regularization operator $||Tf||^2$ penalizes high complexity or other undesired

¹The classes could also correspond to complex brain states as in *mind reading* paradigms (see Haynes and Rees (2006)) or brain states such as attention, workload, emotions, etc.

properties of a function f; λ introduces an additional parameter to the model (often called hyperparameter) that needs to be estimated as well. The estimation of this parameter and hence taking model complexity into account, raises the problem of model selection (e.g., Akaike (1974); Poggio and Girosi (1990); Moody (1992); Murata et al. (1994)), i.e., how to find the optimal complexity of a function or accordingly the appropriate function class (in Section 4 we will discuss the practical aspects of model selection). As a brief remark, Linear Discriminant Analysis (LDA) employs regularization through shrinkage (Stein (1956); Friedman (1989)) while neural networks use early stopping (Amari et al. (1997)), weight decay regularization or asymptotic model selection criteria (e.g., network information criterion (Murata et al. (1994); Akaike (1974)), see also (Bishop (1995); Orr and Müller (1998))). On the other hand, support vector machines (SVMs) regularize according to what kernel is being used (Smola et al. (1998)) and limit their capacity according to Vapnik-Chervonenkis (VC) theory (Vapnik (1995)). We will now briefly outline a few algorithms.



Figure 3: Schematic illustration of model selection. The solid line represents the empirical error, the dashed line the expected error. With higher complexity, the ability of the model to overfit the sample data increases, visible from a low empirical and an increasing expected error. The task of model selection is to determine the model with the smallest generalization error.

2.2. Linear Classification

For computing the parameters of a linear decision function (cf. (2.1) and Fig. 1), namely the normal vector **w** and the bias *b*, we will in the following discuss different linear methods: Linear Discriminant Analysis (LDA) including procedures for regularizing LDA, as well as linear programming machines (LPM).

2.2.1. Linear Discriminant Analysis, Fisher's Discriminant and Regularization

In case of LDA the two classes are assumed to be normally distributed with different means but identical full rank covariance matrix. Suppose the true means μ_i (i=1,2) and the true covariance matrix Σ are known, then the normal vector **w** of the Bayes optimal separating hyperplane of the LDA classifier is given as

$$\mathbf{w} = \Sigma^{-1}(\mu_1 - \mu_2).$$

In order to compute \mathbf{w} for real data, the means and covariances need to be approximated empirically, see Section 2.2.2.

A more general framework is the well-known Fisher's Discriminant analysis (FDA), that maximizes the so-called Rayleigh coefficient

(2.6)
$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}}$$

where the within class scatter $S_w = \sum_{i=1}^2 S_i$ with $S_i = \sum_{\boldsymbol{x} \in C_i} (\boldsymbol{x} - \mu_i) (\boldsymbol{x} - \mu_i)^T$ and the class means are defined as $\mu_i = \frac{1}{n_i} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$ and n_i is the number of patterns \boldsymbol{x}_i in class C_i . The between class scatter $S_B = 1/2 \sum_{i=1}^2 (\mu - \mu_i)(\mu - \mu_i)^T$, where $\mu = 1/2 \sum_{i=1}^2 \mu_i$. A solution to (2.6) can be found by solving the generalized Eigenvalue problem (cf. Golub and van Loan (1996)). Considering only two classes, FDA and LDA can be shown to yield the same classifier solution. However, both methods can be extended for the application to multiple classes.

Although it is a common wisdom that linear methods such as LDA and FDA are less likely to overfit, we would like to stress that they also require careful regularization: *not only* for numerical reasons. Here, the regularization procedure will be less necessary to avoid the typical overfitting problems due to excessive complexity encountered for nonlinear methods (see Fig. 2 and Fig. 4-right). Rather regularization will help to limit the influence of outliers that can distort linear models (see Fig. 4-center). However, if possible, a removal of outliers prior to learning is to be preferred (e.g., Schölkopf et al. (2001); Harmeling et al. (2006)).

A mathematical programming formulation of regularized Fisher Discriminant analysis (RFDA) as a linear constrained, convex optimization problem was introduced in Mika et al. (2001) as

(2.7)
$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2} \|\mathbf{w}\|_{2}^{2} + \frac{C}{n} \|\xi\|_{2}^{2}$$

s.t.
$$y_{i} \cdot ((\mathbf{w}^{\mathsf{T}} \mathbf{x}_{i}) + b) = 1 - \xi_{i}, \ i = 1, \dots, n$$

$$\xi_{i} \geq 0,$$

where $||\mathbf{w}||_2$ denotes the 2 -norm $(||\mathbf{w}||_2^2 = \mathbf{w}^\top \mathbf{w})$ and C is a model parameter that controls for the amount of constraint violations introduced by the slack variables ξ_i . The constraints $y_i \cdot ((\mathbf{w}^\top \mathbf{x}_i) + b) = 1 - \xi_i$ ensure that the class means are projected to the corresponding class labels ±1. Minimizing the length of **w** maximizes the margin between the projected class means relative to the intra class variance. Note that (2.8) can be the starting point for further mathematical program formulations of classifiers such as the sparse FDA, which uses an 1-norm regularizer: $||\mathbf{w}||_1 = \sum |w_n|$ etc. (cf. Mika et al. (2003)).



Figure 4: The problem of finding a maximum margin "hyper-plane" on reliable data (left), data with outlier (middle) and with a mislabeled pattern (right). The solid line shows the resulting decision line, whereas the dashed line marks the margin area. In the middle and on the left the original decision line is plotted with dots. The hard margin implies noise sensitivity, because only one pattern can spoil the whole estimation of the decision line. Figure from Rätsch et al. (2001).

2.2.2. Linear Discriminant Analysis with Shrinkage

The optimality statement for LDA depends crucially on the never fullfilled assumption, that the *true* class distributions are known. Rather, means and covariance matrices of the distributions have to be estimated from the training data.

The standard estimator for a covariance matrix is the empirical covariance which is unbiased and has under usual conditions good properties. But for extreme cases of high-dimensional data with only a few data points that is typically encountered in neuroimaging data, the empirical estimation may become inprecise, because the number of unknown parameters that have to be estimated is quadratic in the number of dimensions. As substantiated in Blankertz et al. (2010), this results in a systematic error: Large eigenvalues of the original covariance matrix are estimated too large, and small eigenvalues are estimated too small. Shrinkage is a common remedy for the systematic bias (Stein (1956)) of the estimated covariance matrices (e.g., Friedman (1989)): the empirical covariance matrix $\hat{\Sigma}$ is replaced by

(2.8)
$$\tilde{\Sigma}(\gamma) := (1 - \gamma)\hat{\Sigma} + \gamma \nu \mathbf{I}$$

for a tuning parameter $\gamma \in [0, 1]$ and ν defined as average eigenvalue of $\hat{\Sigma}$ and I being the identity matrix.

Then the following holds: $\tilde{\Sigma}$ and $\hat{\Sigma}$ have the same Eigenvectors; extreme eigenvalues (large or small) are modified (shrunk or elongated) towards the average v; $\gamma = 0$ yields unregularized LDA, $\gamma = 1$ assumes spherical covariance matrices.

Using LDA with such a modified covariance matrix is termed regularized LDA or LDA with shrinkage. For a long time, complex or time-consuming methods have been used to select shrinkage parameter γ , e.g., by means of cross validation. Recently an analytic method to calculate the optimal shrinkage parameter for certain directions of shrinkage was found (Ledoit and Wolf (2004); see also Vidaurre et al. (2009) for the first application to brain imaging data) that is surprisingly simple. The optimal value only depends on the sample-to-sample variance of entries in the empirical covariance matrix (and values of $\hat{\Sigma}$ itself). When we denote by $(\mathbf{x}_k)_i$ and $(\hat{\mu})_i$ the *i*-th element of the vector \mathbf{x}_k and $\hat{\mu}$, respectively and denote by s_{ij} the element in the *i*-th row and *j*-th column of $\hat{\Sigma}$ and define

$$z_{ij}(k) = ((\mathbf{x}_k)_i - (\hat{\mu})_i) ((\mathbf{x}_k)_j - (\hat{\mu})_j),$$

then the optimal parameter γ for shrinkage towards identity (as defined by (2.8)) can be calculated as (Schäfer and Strimmer (2005))

$$\gamma^{\star} = \frac{n}{(n-1)^2} \frac{\sum_{i,j=1}^{d} \operatorname{var}_k(z_{ij}(k))}{\sum_{i \neq j} s_{ij}^2 + \sum_i (s_{ii} - \nu)^2}$$

2.2.3. Linear Programming Machines

Finally, we would like to introduce so-called Linear Programming Machines (LPMs, Bennett and Mangasarian (1992); Tibshirani (1994, 1996); Hastie et al. (2001); Müller et al. (2001); Rätsch et al. (2002)). Here, slack variables ξ corresponding to the estimation error incurred as well as parameters **w** are optimized to yield a sparse regularized solution

(2.9)
$$\min_{\mathbf{w},b,\xi} \frac{1}{2} ||w||_1 + \frac{C}{n} ||\xi||_1$$

s.t. $y_i \cdot ((\mathbf{w}^\top \mathbf{x}_i) + b) \ge 1 - \xi_i, \ i = 1, \dots, n$
 $\xi_i \ge 0.$

LPMs achieve sparse solutions (i.e. most values of w become zero) by means of explicitly optimizing the 1-norm in the objective function instead of the 2-norm, which is known to yield non-sparse solutions. Due to this property, LPM and sparse FDA are excellent tools for variable selection. In other words, while solving the classification problem, the user is not only supplied with a good classifier but also with the list of variables that are relevant for solving the classification task (Blankertz

et al. (2006a); Müller et al. (2001); Blankertz et al. (2002); Lal et al. (2004); Tomioka and Müller (2010)).

2.3. Beyond linear classifiers

Kernel based learning has taken the step from linear to nonlinear classification in a particularly interesting and efficient² manner: a linear algorithm is applied in some appropriate (kernel) feature space. While this idea first described in Schölkopf et al. (1998) is simple, it is yet very powerful as all beneficial properties (e.g., optimality) of linear classification are maintained³, but at the same time the overall classification is nonlinear in input space, since feature- and input space are nonlinearly related. A cartoon of this idea can be found in Fig. 5, where the classification in input space requires some complicated non-linear (multi-parameter) ellipsoid classifier. An appropriate feature space representation, in this case polynomials of second order, supply a convenient basis in which the problem can be most easily solved by a linear classifier.

However, by virtue of the kernel-trick the input space does not need to be explicitly mapped to a high dimensional feature space by means of a non-linear function $\Phi : \mathbf{x} \mapsto \Phi(\mathbf{x})$. Instead, kernel based methods take advantage from the fact that most linear methods only require the computation of dot products. Hence, the trick in kernel based learning is to substitute an appropriate kernel function in the input space for the dot products in the feature space, i.e.,

(2.10)
$$k : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R},$$

such that $k(\boldsymbol{x}, \boldsymbol{x}') = \phi(\boldsymbol{x})^\top \phi(\boldsymbol{x}').$

More precisely, kernel-based methods restrict to the particular class of kernel functions k that correspond to dot products in feature spaces and hence only implicitly maps the input space to the corresponding feature space. Commonly used kernels are for instance:

(2.11) GAUSSIAN
$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right),$$

 $\sigma > 0$

(2.12) POLYNOMIAL
$$k(x, x') = (x^{\top}x' + c)^d$$

(2.13) SIGMOID
$$k(\mathbf{x}, \mathbf{x}') = \tanh(\kappa(\mathbf{x}^{\top}\mathbf{x}') + \theta),$$

 $\kappa > 0, \ \theta < 0$

Examples of kernel-based learning machines are among others, e.g., Support Vector Machines (SVMs) (Vapnik (1995); Müller et al. (2001)), Kernel Fisher Discriminant (Mika et al. (2003)) or Kernel Principal Component Analysis (Schölkopf et al. (1998)).



Figure 5: Two dimensional classification example. Using the second order monomials x_1^2 , $\sqrt{2}x_1x_2$ and x_2^2 as features a separation in feature space can be found using a *linear* hyperplane (right). In input space this construction corresponds to a *non-linear* ellipsoidal decision boundary (left). From Müller et al. (2001).

2.4. Support Vector Machines

In order to illustrate the application of the kerneltrick, let us consider the example of the SVM (Vapnik (1995); Müller et al. (2001)). Here, the primal optimization problem of a linear SVM is given similar to (2.7) and (2.9) as

(2.14)
$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2} ||w||_2^2 + C \sum_{i=1}^n \xi_i$$

s.t. $y_i \cdot ((\mathbf{w}^\top \mathbf{x}_i) + b) \ge 1 - \xi_i, \ i = 1, \dots, n$
 $\xi_i \ge 0.$

However, in order to apply the kernel trick we rather use the dual of (2.14), i.e.,

(2.15)
$$\max_{\alpha} \sum_{i=1}^{n} \alpha_{i} - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_{i} \alpha_{j} y_{i} y_{j} \cdot \left(\boldsymbol{x}_{i}^{\top} \boldsymbol{x}_{j} \right)$$

s.t. $C \ge \alpha_{i} \ge 0, \quad i = 1, \dots, n,$
$$\sum_{i=1}^{n} \alpha_{i} y_{i} = 0,$$

To construct a nonlinear SVM in the input space, one (implicitly) maps the inputs \mathbf{x} to the feature space by a non-linear feature map $\Phi(\mathbf{x})$ and computes an optimal hyperplane (with threshold) in feature space. To this end, one substitutes $\Phi(\mathbf{x}_i)$ for each training example \mathbf{x}_i in (2.15). As the \mathbf{x}_i only occur in dot products, one can apply the kernel trick and substitute a kernel k for the dot products, i.e., $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^{\top} \Phi(\mathbf{x}_j)$ (cf. Boser et al. (1992); Guyon et al. (1993)). Hence, the non-linear formulation of the SVM is identical to

²By virtue of the so-called 'kernel trick' Vapnik (1995).

³As we do linear classification in this feature space.

(2.15), with the exception that the kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ substitutes for the dot $(\mathbf{x}_i^{\top} \mathbf{x}_j)$. Advantageously, many of the α_i will be vanishing; the samples associated with these non-zero coefficients are referred to as support vectors (SV). As the weight vector \mathbf{w} in the primal formulation reads $\mathbf{w} = \sum_i y_i \alpha_i \Phi(\mathbf{x}_i)$, we obtain the nonlinear decision function as

$$f(\mathbf{x}) = \operatorname{sgn}\left(\mathbf{w}^{\top} \Phi(\mathbf{x}) + b\right)$$

= $\operatorname{sgn}\left(\sum_{i: \alpha_i \neq 0} y_i \alpha_i \cdot \left(\Phi(\mathbf{x}_i)^{\top} \Phi(\mathbf{x})\right) + b\right)$
(2.16) = $\operatorname{sgn}\left(\sum_{i: \alpha_i \neq 0} y_i \alpha_i \cdot k(\mathbf{x}_i, \mathbf{x}) + b\right)$

Here, the sum in (2.16) is a sparse expansion as it only runs over the set of SVs. Note, while the α_i are determined from quadratic optimization problem ⁴ (2.15), the threshold *b* can be computed by exploiting the fact that for all support vectors \mathbf{x}_i with $C \ge \alpha_i \ge 0$, the slack variable ξ_i is zero, and hence

(2.17)
$$\sum_{j=1}^{n} y_j \alpha_j \cdot k(\boldsymbol{x}_i, \boldsymbol{x}_j) + b = y_i.$$

If one uses an optimizer that works with the double dual (see, e.g., Vanderbei and Shanno (1997); Boyd and Vandenberghe (2004)), one can also recover the value of the primal variable b directly from the corresponding double dual variable.

Note that recently Braun et al. (2008) has observed that the excellent generalization that is typically observed when using SVMs in high dimensional applications with few samples is due to its very economic representation in Kernel Hilbert space. Given the appropriate kernel, only a very low dimensional subspace is task relevant.

2.5. k-nearest neighbors

A further well-known non-linear algorithm for classification is the so-called *k*-nearest neighbor method. Here, every unseen point x is compared through a distance function $dist(x, x_i)$ to all points x_i (i = 1, ..., n) of the training set. The *k* minimal distances are computed and the majority over the corresponding labels y_i is taken as a resulting label for x (note that this simple rule also holds for multiclass problems). This strategy provides a very simple local approximation of the conditional density function. The *k*-nearest neighbor method is known to work well, if a reasonable distance (typically the Euclidean one) is available and if the number of data points in the training set is not huge. The hyperparameter k can be selected, e.g., by cross-validation; if the data is noisy a large k should be chosen.

2.6. Application to brain imaging data

The predominant methods both for fMRI and EEG/MEG analysis are so far mostly linear, however, nonlinear methods can easily be included in the analysis by including these model classes into the model selection loop, as we will discuss in a later section in detail. For a brief discussion of linear versus nonlinear methods see, e.g., Müller et al. (2003).

In EEG studies mainly LDA, shrinkage/regularized LDA, sparse Fisher, and linear programs are in use (e.g., Dornhege et al. (2007)) are in use. Here, it was observed that through a proper preprocessing the class conditional distributions become Gaussian with a very similar covariance structure (cf. Blankertz et al. (2002, 2006a); Dornhege et al. (2007)). Under such circumstances, theory suggests that LDA would be the optimal classifier. However, changes in the underlying distribution, the use of multimodal features (e.g., Dornhege et al. (2004)), or the presence of outliers may require to proceed to nonlinear methods (see the discussion in Müller et al. (2003)).

The driving insight in fMRI analysis was to go from univariate analysis tools that correlate single voxels with behavior to a full multivariate correlation analysis (e.g., Hansen et al. (1999); Haxby and et al (2001); Strother et al. (2002); Cox and Savoy (2003); Kamitani and Tong (2005); Haynes and Rees (2006); Kriegeskorte et al. (2006); Mitchell et al. (2008)). A main argument for using especially SVMs and LPMs was their well-known benign properties in the case where the number of input dimensions in x is high while the number of samples is low (e.g., Vapnik (1995); Müller et al. (2001); Braun et al. (2008)). This particular very unbalanced situation is an important issue in fMRI, since the number of voxels is of the order ten-thousands while the number of samples rarely exceeds a few hundreds (e.g., Hansen et al. (1999); Strother et al. (2002); Cox and Savoy (2003)). Physiological priors that allow, e.g., to define regions of interest, have led to further specialized analysis tools like the search-light method. Here, the signal to noise ratio is improved by discarding some potentially noisy and task unrelated areas while enhancing the interesting bit of information spatially (e.g., Kriegeskorte et al. (2006)). In some cases improvements through the use of nonlinear kernel functions have been reported, e.g., LaConte et al. (2005) studied polynomial kernels.

⁴A selection of Open Source software for SVMs can be found on www.kernel-machines.org.

Other priors have been derived from the paradigm. For example, for understanding the neural basis of word representation, Mitchell et al. (2008) used word cooccurances in language inferred from a large scale document corpus, to derive a codebook of fMRI patterns corresponding to brain activity for representing words. These patterns could be superimposed for out of sample forecasting, thus predicting the most probable way a certain novel word is represented in the fMRI pattern.

3. Dimensionality Reduction

So far, we did not much concern about the domain of the input data and solely assumed the classifiers to be applicable to the data (x, y). In practice and particularly in the domain of computational neuroscience, the input data x often exhibit an adverse ratio between its dimensionality and the number of samples. For example, a typical a common task for EEG-based Brain-Computer Interfaces (BCI) requires the classification of one second intervals of brain activity, recorded at sampling frequencies up to 1 kHz, from possibly 128 electrodes. Hence, the dimensionality of the input data x amounts approximately to 10⁵, while the number of training samples is typically rather small, up to a few hundred samples. Moreover, the data is contaminated by various sources of interfering noise, while the discriminative information that is the task relevant part of the data is often concentrated in a low dimensional subspace. Consequently, in order to make the classification task feasible the dimensionality of the data needs to be significantly reduced, and informative features have to be extracted (see also Fig. 1). Thus, feature extraction likewise involves spatial, spectral, and temporal preprocessing of the input.

Nevertheless, the extraction of relevant features is a highly paradigm specific task and also differs for the various recording techniques due to their temporal and spatial resolutions. Moreover, the extraction of task relevant features should be facilitated by incorporating prior neurophysiological knowledge, e.g., about the cognitive processes underlying the specific paradigm. In return, purely data driven feature extraction methods can potentially provide new findings about the involved cognitive processing and might therefore contribute to the generation of neurophysiological hypotheses (Blankertz et al. (2006a)). Thus, feature extraction has to be considered not just as a data analytical but rather as a heavily interdisciplinary endeavor.

The common approaches for dimensionality reduction can be subdivided into two main categories. On the one hand side there are variants of factor



Figure 6: Essential difference between PCA and ICA. The left panel shows a mixture of two super-Gaussian signals in the observation coordinates, along with the estimated PCA axes (green) and ICA axes (red). Projecting the observed data to these axes reveals, that PCA did not properly identify the original independent variables (central), while ICA has well identified the original independent data variables (right).

models, such as the well-known Principle Component Analysis (PCA), Independent Component Analysis (ICA) (cf. Comon (1994); Bell and Sejnowski (1995); Ziehe and Müller (1998); Hyvärinen et al. (2001)), non-negative matrix factorization, archetypal analysis, sparse PCA (Lee and Seung (2000); Zou et al. (2004); Schölkopf et al. (1998)) or Non-Gaussian Component Analysis (Blanchard et al. (2006)); that perform a factorization of the input data x in a purely *unsupervised* manner, i.e., without using the class information. The application of these methods serves several purposes (a) dimensionality reduction by projecting onto a few (meaningful) factors, (b) removal of interfering noise from the data to increase the signal-to-noise ratio of the signals of interest, (c) removal of nonstationary effects in data or (d) grouping of effects. Notwithstanding their general applicability, unsupervised factor analysis often requires manual identification of the factors of interest.

The second class of methods, namely *supervised* methods, make explicit use of the class labels in order to find a transformation of the input data to a reduced set of features with high task relevance. For example, the Common Spatial Pattern (CSP) (cf. Fukunaga (1972); Koles (1991); Ramoser et al. (2000)) algorithm and derivatives thereof (Blankertz et al. (2008b); Lemm et al. (2005); Dornhege et al. (2006); Tomioka and Müller (2010)) is widely used to extract discriminative oscillatory features.

In the following, we will briefly describe two frequently used dimensionality reduction methods. First we will briefly introduce the unsupervised Independent Component Analysis, and subsequently discuss the supervised CSP algorithm.

3.1. Independent component analysis

Under the often valid assumption that the electric fields of different bioelectric current sources superim-

pose linearly, the measured neurophysiological data can be modeled as a linear combination of component vectors. For instance, in case of independent component analysis (ICA) it is assumed that the observed signals x(t) are a linear mixture of $M \le N$ mutually independent sources s(t), i.e., data are modeled as a linear combination of component vectors:

$$\mathbf{x}(t) = A\mathbf{s}(t)$$

where $A \in \mathbb{R}^{N \times M}$ denotes the linear mixing matrix. In this case (3.18) is invertible and ICA decomposes the observed data $\mathbf{x}(t)$ into independent components $\mathbf{y}(t)$ by estimating the inverse decomposition matrix $W \cong A^{-1}$, such that $\mathbf{y}(t) = W\mathbf{x}(t)$.

There exists a vast number of ICA algorithms that can solve the task of estimating the mixing matrix A. They only differ in the particular choice of a so-called index funtion and the respective numerical procedures to optimize this function. In general, the index function employs a statistical property that takes on its extreme values if the projected sources are independent. Most research conducted in the field of ICA uses higher-order statistics for the estimation of the independent components (Comon (1994); Hyvärinen et al. (2001)). For instance, the Jade algorithm (Cardoso and Souloumiac (1993)) is based on the joint diagonalization of matrices obtained from "parallel slices" of the 4th-order cumulant tensor. Although this algorithm performs very efficiently on low dimensional data, it becomes computational infeasible for high dimensional problems, as the effort for storing and processing the 4th-order cumulants is $O(m^4)$ in the number of sources. As a remedy for this problem Hyvärinen and Oja (1997) developed a general fix-point iteration algorithm termed FastICA, that optimizes a contrast function measuring the distance of the source probability distributions from a Gaussian distribution.

Note that ICA can recover the original sources s(t) only up to scaling and permutation. Fig. 6 sketches the essential difference between ICA and the well known PCA method.

3.2. Common spatial pattern

Unlike unsupervised methods such as PCA and ICA, the common spatial pattern (CSP) algorithm (Fukunaga (1972)) makes explicit use of the label information in order to calculate discriminative spatial filters that emphasize differences in signal power of oscillatory processes (Koles and Soong (1998)). To illustrate the basic idea of CSP: suppose we observe two class distributions in a high-dimensional space, the CSP algorithm finds



Figure 7: Essential steps of CSP: The blue and green ellipsoids refer to the two class conditional covariance matrices along with the principal axes, while the mutual covariance matrix is depicted in gray. Left: original data. Central: data distribution after *whitening*. Right: after a final rotation the variance along the horizontal direction is maximal for the *green* class, while it is minimal for the *blue* class and vice versa along the vertical direction.

directions (spatial filters) that maximize the signal variance for one class, while simultaneously minimizing the signal variance for the opposite class.

To be more precise, let Σ_1 and Σ_2 denote the two class conditional signal covariance matrices. The spatial CSP filters *w* are obtained as the generalized Eigenvectors of the following system

(3.19)
$$\Sigma_1 w = \lambda \Sigma_2 w.$$

A solution to (3.19) is typically derived in two steps: first the data are whitened with respect to the mutual covariance matric $\Sigma_1 + \Sigma_2$; secondly a terminal rotation aligns the principal axes with the coordinate axes (see Fig. 7). However, the interpretation of filter matrix *W* is two-fold, the rows of *W* are the spatial filters, whereas the columns of W^{-1} can be seen as the *common spatial patterns*, i.e., the time-invariant coupling coefficients of each source with the different sensors. For a detailed discussion on the relation between spatial patterns and spatial filters see (Blankertz et al. (2010)).

Originally the CSP algorithm was conceived for discriminating between two classes, but has also been extended to multi-class problems (Dornhege et al. (2004); Grosse-Wentrup and Buss (2008)). Further extension of CSP were proposed in Lemm et al. (2005); Dornhege et al. (2006); Tomioka and Müller (2010); Farquhar (2009); Li and Guan (2006) with the goal of simultaneously optimizing discriminative spatial and spectral filters. For a comprehensive overview of optimizing spatial filters we refer to Blankertz et al. (2008b).

4. Cross-validation and Model selection

Given the data sample, the task of model selection is to choose a statistical model from a set of potential models (the function class), which may have produced the data with maximum likelihood, i.e., to choose the model which resembles the underlying functional relationship best. The set of potential models is in general not restricted, although in practice limitations are imposed by the preselection of a model class by the scientists. A typical setting may, for example, comprise models from distinct classes, such as linear and non-linear models; but may also solely consist of non-linear models that differ in the employed SV-kernel function which needs to be selected. On the other hand, model selection is also frequently used to determine the optimal value of model hyperparameters⁵. However, in all these cases the general task remains the same: the expected out-ofsample performances of the different models need to be evaluated on the common basis of the given data sample.

As discussed previously, complex models can potentially better adapt to details of the data. On the other hand an excessively complex model will tend to overfit, i.e., will rather fit to the noise than to the underlying functional relationship. Hence, its out-of-sample performance will deteriorate (see Fig. 2 for an illustration of overfitting). However, overfitting not only occurs when a model has too many degrees of freedom, in relation to the amount of data available. Also simple models tend to overfit, if the influence of outliers is not treated appropriately (see Fig. 4). However, in any case the training error does not provide an unbiased estimate of the model performance and hence can not be used to select the best model.

At this point we would like to stress, that an unbiased estimation of the model performance is one of the most fundamental issues in statistical data analysis, as it provides the answer to: How will the model generalize on new, previously unseen data and hence how accurately it will perform in *practice*. In terms of machine learning, this quantity is called the *generalization error* or *expected risk* of a model (cf. (2.3)), which applies likewise to a regression or classification model. So model selection in essence reduces to *reliably estimating* the ability of a model to generalize well to new unseen data and to pick the model with the smallest expected error (see Fig. 3).

4.1. Estimation of the Generalization Error

In order to present a universally applicable conceptual framework of estimating the generalization error, let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ represent our original sample set of *n* labeled instances. Moreover, let $f(\cdot|\mathcal{D})$ be the model that has been learned on the sample set and correspondingly $f(\mathbf{x}|\mathcal{D})$ denotes the model prediction at the instance x. Suppose further that the error of the model at an instance (\mathbf{x}, y) from the sample space is measured by a given loss function $err = l(y, f(\mathbf{x}|\mathcal{D}))$, e.g., by the mean squared error, or the 0/1-loss as it is commonly used for classification. Based on this notation, we will introduce the prevalent concepts for assessing the generalization error of a statistical model given a sample set. All of the following concepts have in common that they are based on a holdout strategy.

A holdout strategy generally splits the sample set in two independent, complementary subsets. One subset, commonly referred to as *training set*, is solely used for fitting the model, i.e., to estimate the model parameters, such as, the normal vector of the separating hyperplane of an SVM. In contrast, the second subset is exclusively used for the purpose of validating the estimated model on an independent data set and is therefore termed *validation set*. Formally, let $\mathcal{D}_v \subset \mathcal{D}$ denote the holdout validation set of size n_v , and define $\mathcal{D}_t = \mathcal{D} \setminus \mathcal{D}_v$ as the complementary sample set for training. The estimated validation error is defined as

(4.20)
$$err_{v} = \frac{1}{n_{v}} \sum_{i \in \mathcal{D}_{v}} l(y_{i}, f(\boldsymbol{x}_{i} | \mathcal{D}_{t})).$$

Note, the learned model as well as the accuracy of the estimated generalization error will depend on the particularly chosen partition of the original sample into training and validation set and especially on the size of these sets. For example, the more instances we leave for validation, the less samples remain for training and hence the model becomes less accurate. Consequently, a bias is introduced to the estimated model error. On the contrary, using fewer instances for validating the model will increase the variance of the estimated model error.

4.1.1. Cross-validation

To trade off between bias and variance several approaches have been proposed. On the one hand side, there is a multitude of *cross-validation* (CV) schemes, where the process of splitting the sample in two is repeated several times using different partitions of the sample data. Subsequently, the resulting validation errors are averaged across the multiple rounds of CV. The miscellaneous CV schemes differ by the way they split the sample data in two. The most widely used method is *K-fold CV*. Here, the sample data is randomly divided into *K* disjoint subsets $\mathcal{D}_1, \ldots, \mathcal{D}_K$ of approximately equal size. The model is then trained *K* times, using all of the data subsamples except for one, which is left out as validation set. In particular, in the *k*-th run \mathcal{D}_k

⁵This might simultaneously be the regularization strength *C* of a SVM and kernel width σ of say a Gaussian kernel.

is selected as validation set, while the union of the remaining K-1 subsamples, i.e., $\mathcal{D} \setminus \mathcal{D}_k$ serves as training set. The *K*-fold CV-error is then the averaged error of the *K* estimated models, where each model is evaluated separately on its corresponding validation set

(4.21)
$$err_{CV} = \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in \mathcal{D}_k} l(y_i, f(\mathbf{x}_i | \mathcal{D} \setminus \mathcal{D}_k)).$$

Note, that the cross-validation error is still a random number that depends on the particular partition of the data into the K folds. Therefore, it would be highly desirable to perform a *complete K-fold CV* to estimate its mean and variance by averaging across all possible partitions of the data into K folds. In practice this is computationally intractable even for small samples. Nevertheless, repeating the K-fold cross-validation several times can additionally reduce the variance of the estimator at lower computational costs.

An alternative cross-validation scheme is the socalled *leave-one-out cross-validation* (LOO-CV). As the name already indicates, LOO-CV uses all but a single data point of the original sample for training the model. The estimated model is then validated on the single observation left out. This procedure is repeated, until each data point once served as validation set. In particular, in the *i*-th run the validation set corresponds to $\mathcal{D}_i = (\mathbf{x}_i, y_i)$, while the model is trained on the complement $\mathcal{D} \setminus \mathcal{D}_i$. The LOO-CV estimator is defined as the averaged error

(4.22)
$$err_{LOO} = \frac{1}{n} \sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i | \mathcal{D} \setminus \mathcal{D}_i)).$$

Note that LOO-CV actually performs a complete *n*-fold CV. However, since the model has to be trained *n* times, LOO-CV is computational demanding. On the other hand, it leads to an almost unbiased estimate of the generalization error, but at the expense of an increased variance of the estimator. In general, cross-validation schemes provide a nearly unbiased estimate of the generalization error, at the cost of significant variability, particularly for discontinuous loss functions Efron and Tibshirani (1997). In order to achieve a good compromise between bias and variance the use of 10-fold or 5-fold CV are often recommended.

4.1.2. Bootstrapping

In case of discontinuous error functions *bootstrap methods* Efron and Tibshirani (1993) may smooth over possible discontinuity. A bootstrap sample b is created by sampling n instances with replacement from

the original sample \mathcal{D} . The bootstrap sample *b* is then used for training, while the set left out that is $\mathcal{D} \setminus b$ serves as validation set. Using a sampling procedure with replacement, the expected size of the validation set is $n(1-\frac{1}{n})^n \approx 0.368n$. Correspondingly, the training set, which is of size *n*, has $\approx 0.632n$ unique observations which leads to an overestimation of the prediction error. The .632 *bootstrap* estimator Efron and Tibshirani (1993) corrects for this, by adding the underestimated resubstitution error,

(4.23)
$$err_{boot} = \frac{1}{B} \sum_{b} 0.632 \cdot \sum_{i \notin b} l(y_i, f(\mathbf{x}_i | b)) + 0.368 \cdot \sum_{i=1}^{n} l(y_i, f(\mathbf{x}_i | D)).$$

However, the standard bootstrap estimate is an upwardly biased estimator of the model accuracy. In particular, it can become to overly optimistic for excessively complex models that are capable to highly overfit the data.

In the context of a feature subset selection experiment for regression, a comprehensive comparison of different schemes for estimating the generalization error has been conducted in Breiman and Spector (1992). Among others schemes, they compared leave-one-out cross-validation, K-fold cross-validation for various K, stratified version of cross-validation and bias corrected bootstrap on artificially generated data. For the task of model selection they concluded ten-fold crossvalidation to outperform the other methods.

5. Practical Hints for Model Evaluation

5.1. Models with Hyperparameters

Many preprocessing and classification methods have one or more hyperparameters that need to be adjusted to the data by means of model selection. Examples are the kernel width of an Gaussian kernel, the regularization parameter λ of the regularized LDA, but also the number of neighbors in a k-nearest neighbor approach, or the number of principal components to be selected. According to the previously introduced general concept of model selection, those hyperparameters have to be selected by means of an unbiased estimate of the generalization performance, i.e., have to be evaluated on a validation set that is *independent* of data used for training. Since the selection of the hyperparameter is an integral part of the model, the model selection scheme and therefore the estimation of the validation error becomes part of the model itself. Consequently, the cross-validation error that has been used for adjusting the hyperparameter of a model becomes a biased estimate of the overall model performance, as it has been minimized by the model selection. Consequently, to estimate the generalization error of the entire model (including the hyperparameter selection) another independent data set is required.

To emphasize the severeness of this issue, let us consider the following illustrative example. Given a fixed data set the classification performance of linear discriminant analysis (without hyperparameters) needs to be compared with the performance of an SVM with a Gaussian kernel. Cross-validation is performed for LDA and also for an SVM using a Gaussian kernel for various combinations of the hyperparameters, i.e., the kernel width and the regularization parameter. One can expect the SVM to yield better results than LDA for some combinations, while it will perform worse for others. Nevertheless, it cannot be concluded that the SVM with the 'optimal selection' of its hyperparameters outperforms the LDA solution. Obviously, as it has been minimized by the model selection procedure, the CV-error of the selected SVM is to optimistic about the model performance. Note that the CV-error is a random number that exhibits a certain degree of variability. Accordingly, repeated sampling from the distribution of the CV error will favor models with a larger set of parameter combinations (here, the SVM model). In particular, the smaller CV error for some parameter combinations of the SVM could just be a lucky coincidence induced by repeatedly evaluating the SVM model. The severeness of biasing the results depends on several factors, e.g., how much complexity can be controlled by the parameters, dimensionality of the features, and the methods of selecting the parameters.

Nested Cross-Validation: In order to obtain an unbiased estimate of the generalization performance of the complete model (including selection of the hyperparameter), an additional data set is required which is independent from both, the training and the validation data. To this end, a nested cross-validation scheme is most appropriate. Algorithmically, it can be described as two nested loops of cross-validation. In the inner loop, the hyperparameter as part of the model has to be selected according to inner CV error, while in the outer loop, the selected models are evaluated with respect to their generalization ability on an independent validation set. The outer loop CV-error is similar to (4.21)

(5.24)
$$err_{CV} = \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in \mathcal{D}_k} l\left(y_i, f_{CV}\left(x_i | \mathcal{D}^{\vee k}\right)\right).$$

Here, \mathcal{D}_k denotes the k^{th} validation set of the outer CVloop, while we use the short hand notation $\mathcal{D}^{\setminus k}$ for the corresponding outer loop training set $\mathcal{D} \setminus \mathcal{D}_k$. However, the main difference in the above equation compared to the ordinary *K*-fold CV is that the model $f_{\text{CV}}(\cdot | \mathcal{D}^{\setminus k})$ refers to the model that has been selected via the inner *K*-fold CV over the data set $\mathcal{D}^{\setminus k}$, i.e., (5.25)

$$f_{\text{CV}}\left(\cdot | \mathcal{D}^{\forall k}\right) := \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{l=1}^{K} \sum_{i \in \mathcal{D}_{l}^{\forall k}} l\left(y_{i}, f\left(x_{i} | \mathcal{D}^{\forall k} \setminus \mathcal{D}_{l}^{\forall k}\right)\right),$$

with \mathcal{F} denoting the set of models corresponding to different value of the hyperparameter and $\mathcal{D}_l^{\setminus k}$ denoting the l^{th} validation set of the inner CV loop. In order to distinguish more easily between the different layers of crossvalidation, one often refers to the holdout set \mathcal{D}_k of the outer loop as *test sets*. Note, in each round of the outer CV loop the model $f_{\text{CV}}(\cdot|\mathcal{D}^{\setminus k})$ and hence the hyperparameter that is selected can be different. Consequently, nested CV will provide a probability distribution how often a hyperparameter had been selected by the model selection scheme rather than a particular value. On the other hand nested CV gives an unbiased estimate of the generalization performance of the complete model (including selection of the hyperparameters).

5.2. Cross-Validation for Dependent Data

Another practical issue when estimating the generalization error of a model is the validity of the assumption about the independence of the training and the validation data. Obviously, if all instances are distributed independently and identically then arbitrary splits into training and validation set will yield stochastically independent samples and the prerequisite is trivially fulfilled. However, often the experimental design induces dependencies between samples. In such a case, special care is required when splitting the sample in order to ensure the aforementioned independence assumption.

For example, a frequently used paradigm in human brain research divides the course of an experiment into blocks of different experimental conditions. We say that an experiment has a block design, if each block comprises several single-trials all belonging to the same condition, see Fig. 8-a.

In such a setting, special care has to be taken in validating the classification of single-trials according to the experimental conditions. Typically, samples within such a block are likely to be stochastically dependent, while stochastic independence can be generally assumed for samples belonging to different blocks. To



Figure 8: Schema of a block design experiment. (a) Alternation between the two conditions that are to be discriminated is on a longer time scale, compared to shorter segments for which classification is evaluated. This setting requires special treatment for validation. (b) Applying cross validation on the epochs violates the assumption that samples in the training set are independent from the ones in the test set. Due to slowly changing nonstationarities in the brain signals, a trial in the test set is very likely to be classified correctly, if trials from the same block are in the training set.

illustrate this, let us consider a neuroimaging recording. Here, typically many slowly varying processes of background activity exist, and hence neighboring trials within a single block are dependent with respect to the information they share about the state of these slow cortical processes. Consequently, applying a simple holdout strategies as they are conventionally employed by generic LOO or K-fold CV, to block design data will most likely violate the assumption of independence between the training and validation data, see Fig. 8-b.

In order to demonstrate the possible effect of applying a standard CV scheme to block-wise data, we will conduct an illustrative experiment to demonstrate the failure of a generic CV, that severely underestimates the generalization performance. Moreover, we show that the block CV scheme introduced below yields an accurate estimate of the generalization error. However, before we present this example, we will formally introduce a cross-validation scheme that is tailored to the particualar needs of block-wise data.

5.2.1. Block Cross-validation

As indicated, data within a single block are likely to be stochastically dependent, while stochastic independence can be assumed for data across blocks. Consequently, the most simple form of a cross-validation scheme for block-wise data corresponds to a *leave-oneblock-out cross-validation*. Analog to LOO-CV a single block is left out for validation and the model is trained on the remaining data. For most of the experimental paradigms such a CV scheme will be appropriate.

However, in order to introduce a more general block-wise CV method, we will assume that for some pre-defined constant $h \in \mathbb{N}$ the covariance matrices

Cov((x_i, y_i), (x_{i+j}, y_{i+j})) (measured in a suitable norm) are of negligible magnitude for all |j| > h. That is, samples that are further apart than h samples are assumed to be uncorrelated. A so-called h-block cross-validation scheme is a modification of the LOO-CV scheme. Similar to LOO-CV it uses each instance (x_i, y_i) once as validation set, but unlike LOO-CV it leaves an h-block (of size 2h + 1) of the neighboring h samples from each side of the i^{th} sample out of the training set. Following the notation in Racine (2000), we denote the h-block by ($\mathbf{x}_{(i:h)}, y_{(i:h)}$), thus the training set of the i^{th} iteration corresponds to $\mathcal{D}_{(-i:h)} := \mathcal{D} \setminus (\mathbf{x}_{(i:h)}, y_{(i:h)})$. The generalization error of the model is consequently obtained by the cross validation function

(5.26)
$$err_{h} = \frac{1}{n} \sum_{i=1}^{n} l(y_{i}, f(\mathbf{x}_{i} | \mathcal{D}_{(-i:h)}))$$

Although the latter policy resolves the problem of dependencies, it still has one major drawback. As it was shown in Shao (1993) for the particular choice of the mean squared error as loss function, due to the small size of the validation sets the *h*-block cross-validation is inconsistent for the important model class of linear functions. This means, even asymptotically (i.e., for $n \to \infty$) *h*-block cross-validation does not reliably select the *true* model f^* from a class of linear models.

As worked out in Racine (2000), a way out of this pitfall is a technique called *hv-cross validation*. Heuristically spoken, *hv*-cross validation enlarges the validation sets sufficiently. To this end, for a "sufficiently large" *v* each validation set is expanded by *v* additional observations from either side, yielding $\mathcal{D}_i = (\mathbf{x}_{i:v}, y_{i:v})$. Hence, each validation set \mathcal{D}_i is of size $n_v = 2v + 1$. Moreover, in order to take care of the dependencies, *h* observations on either side of $(\mathbf{x}_{(i:v)}, y_{(i:v)})$ are additionally removed to form the training data. Hence, the training data of the *i*th iteration is $\mathcal{D}_{(-i:(h+v))} := \mathcal{D} \setminus (\mathbf{x}_{(i:(h+v))}, y_{(i:(h+v))})$. Now, the cross-validation function (5.27)

$$err_{hv} = \frac{1}{n_v(n-2v)} \sum_{i=v}^{n-1} \sum_{j \in \mathcal{D}_i} ||y_j - f(\mathbf{x}_j)\mathcal{D}_{(-i:(h+v))}||^2$$

n-1

is an appropriate measure of the generalization error. For asymptotic consistency it is necessary that $\lim_{n\to\infty} n_v/n = 1$, thus choosing v such that $v = (n - n^{\delta} - 2h - 1)/2$ for a constant $0 < \delta < 1$ is sufficient to achieve consistency (Shao (1993)).

5.3. Caveats in Applying Cross-Validation

Preprocessing the data prior to the application of cross-validation also requires particular care to avoid bi-

asing the generalization error of the model. Here, in order to adhere to the independence assumption of training and validation set, any parameters of the preprocessing needs to be estimated solely on the training set and not on the test set. This holds likewise for the estimation of principle and independent component analysis, but also for simpler preprocessing strategies, such as normalization of the data. If cross-validation is used, the estimation of the parameters of preprocessing has to be performed within the cross-validation loop on each training set separately and afterwards transfer to the corresponding test and validation data, respectively. In order to achieve a stringently sound validation, it is for example inappropriate to perform ICA on the entire data, select desirable components and use features extracted from those components as input to a classifier, whose performance needs to be evaluated by cross-validation. Although the bias induced by unsupervised preprocessing techniques will usually be not as severe, it can result in improper model selection and overoptimistic results. In contrast, strong overfitting may occur, if a preprocessing method which uses class label information is performed on the whole data set.

5.4. Model Evaluation Allowing for Feature Selection

Feature selection is widely used in order to decrease the dimensionality of the feature vectors and thus to facilitate classification. Typically, feature selection methods are supervised, i.e., they exploit the label information of the data. A simple strategy is, e.g., to select those features that have a large separability index like Fisher Score (Müller et al. (2004)). A more sophisticated strategy is to use classifiers for feature selection, as, e.g., in Müller et al. (2004); Lal et al. (2004); Blankertz et al. (2006a); Tomioka and Müller (2010). For the same reasons as laid out in Section 5.3 it is vital for a sound validation that such feature selection methods are not performed on the whole data. Again, cross-validation has to be used to evaluate the entire model including feature selection scheme, rather then just cross validating the final step of classification. More specifically, feature selection has to be performed within the cross-validation loop on each training set.

5.5. Model Evaluation Allowing for Outlier Rejection

It requires a prudent approach to fairly evaluate the performance of models which employ outlier rejection schemes. While the rejection of outliers from the training data set is unproblematic, their exclusion from the test set is rather not. Here, two issues have to be considered. (1) The rejection criterion must not use the labels of the test data, and all parameters (such as thresholds) have to be estimated on the training data. (2) For evaluation, a metric has to be used, that takes into account the rejection of trials. Obvioulsy, the rejection of test samples in a classification task means a reduced amount of information compared to a method that obtains the same classification accuarcy on all test samples. See Ferrez and Millán (2005) for an application of a performance measure based on Shannon's information theory in the context of rejected samples based on the detection of an error-related brain response.

An outlier rejection method may use label information (e.g., when class-conditional Malahanobis distances are used), but only training data may be used to determine parameters of the method, as in the estimation of the covariance matrix for the Malahanobis distance.

5.6. Loss Functions Allowing for Unbalanced Classes

The classification performance is always evaluated by some loss function, see Section 4.1. Typical examples are 0-1-loss (i.e., average number of misclassified samples) and area under the receiver operator characteristic (ROC) curve (Fawcett (2006)). When using misclassification rate, it must be assured that the classes have approximately the same number of samples. Otherwise, a performance measure that respects the sizes of the classes (or the priors one has about their frequency in the application case) has to be employed. In oddball paradigms, e.g., the task is to discriminate brain responses to an attended rare stimulus from responses to a frequent stimulus, a typical ratio of frequent-torare stimuli is 85:15. In such a setting, an uninformative classifier which always predicts the majority class would obtain an accuracy of 85%. Accordingly, a different loss functions needs to be employed. Denoting the number of samples in class i by n_i , the normalized error can be calculated as weighted average, where errors committed on samples of class *i* are weighted by N/n_i with $N = \sum_k n_k$.

5.7. Regarding Nonstationarities

It is worth to note that any cross-validation scheme implicitly relies on the assumption that the samples are identically distributed. In the context of neurophysiological data this transfers to an assumption of stationarity. Unfortunately, nonstationarities are ubiquitous in neuroimaging data (e.g. Shenoy et al. (2006)). Accordingly, the characteristics of brain signals and, in particular, the feature distributions, often change slowly with time. Therefore, a model fitted to data from the beginning of an experiment may not generalize well on data towards the end of the experiment. This detrimental effect is obscured, when estimating the generalization performance by cross validation, since training samples are drawn from the full time course of the experiment. Accordingly, the classifier is, so to say, prepared for the nonstationarities that occur within the data. Therefore, it is advisable to compare the results of cross-validation with a so called chronological validation, in which the (chronologically) first half of the data is used for training and the second half for testing. If the data includes nonstationarities and the classifier fails to be invariant against them, the chronological validation would predict a substantially worse performance than cross-validation. This situation indicates that the method will suffer from nonstationarity during online operation. In general, one can avoid non-stationarity by (a) constructing invariant features (e.g. Blankertz et al. (2008a)), (b) tracking nonstationarity (e.g. Schlögl et al. (2009); Vidaurre and Blankertz (2010); Vidaurre et al. (2010)), (c) modeling nonstationarity and adapting CV schemes (Sugiyama et al. (2007)), or by (d) projecting to stationary subspaces (von Bünau et al. (2009)).

6. Conclusion

Decoding brain states is a difficult data analytic endeavor, e.g., due to the unfavorable signal to noise ratio, the vast dimensionality of the data, the high trial-to-trial variability etc. In the past, machine learning and pattern recognition have provided significant contributions to alleviate these issues and thus have had their share in many of the recent exciting developments in the neurosciences. In this work, we have introduced some of the most common algorithmic concepts, first from a theoretical viewpoint and then from a practical neuroscience data analyst's perspective. Our main original contribution in this review is a clear account for the typical pitfalls, see Table 1.

Due to space constraints, the level of mathematical sophistication and the number of algorithms described are limited. However, a detailed account was given for the problems of hyper-parameter choice and model selection, where a proper cross-validation procedure is essential for obtaining realistic results that maintain their validity out-of-sample. Moreover, it should be highly emphasized that the proper inclusion of physiological apriori knowledge is helpful as it can provide the learning machines with representations that are more useful for prediction than if operating on raw data itself. We consider such a close interdisciplinary interaction between paradigm and computational model as essential.

Potential pitfall	See
Preprocessing the data based on global statistics of the entire data (e.g., normalization using the global mean and variance)	Section 5.3
Global rejection of artifacts or outliers prior to the analysis (re- sulting in a simplified test set)	Section 5.5
Global extraction or selection of features (illegitimate use of infor- mation about the test data)	Section 5.4
Simultaneously selecting model parameters and estimating the model performance by cross val- idation on the same data (yield- ing a too optimistic estimate of the generalization error)	Section 5.1
Insufficient model evaluation for paradigms with block design	Section 5.2
Neglecting unbalanced class fre- quencies	Section 5.6
Disregarding effects of non- stationarity	Section 5.7

Table 1: Hall of Pitfalls. The table presents a (incomplete) list of the most prominent sources of error that one need to take into consideration, when applying machine learning methods to brain imaging data.

We would like to end by adding the remark that an unforeseen progress in algorithmic development has been caused by the availability of high quality data with a clear task description. This has allowed a number of interested non-(computational) neuroscientists to study experimental data even without access to expensive measurement technology Sajda et al. (2003); Blankertz et al. (2004, 2006b). The authors express their hope that this development will extend also beyond EEG data.

Acknowledgement

We would like to thank our former co-authors for allowing us to use prior published material in this review. The studies were partly supported by the *Bundesministerium für Bildung und Forschung* (BMBF), Fkz 01IB001A/B, 01GQ0850, by the German Science Foundation (DFG, contract MU 987/3-2), by the European Union under the PASCAL2 Network of Excellence, ICT-216886. This publication only reflects the authors' views. Funding agencies are not liable for any use that may be made of the information contained herein.

Author contributions

SL and KRM produced the draft for the part on the machine learning methods and SL work it out in detail and particularly elaborated on the mathematical foundations of model selection and cross-validation. BB contributed the part on pitfalls in model evaluation. TD contributed the part on cross-validation for dependent data. All authors read and approved the manuscript.

- Akaike, H., 1974. A new look at the statistical model identification. IEEE Trans. Automat. Control 19 (6), 716–723.
- Amari, S., Murata, N., Müller, K.-R., Finke, M., Yang, H., 1997. Asymptotic statistical theory of overtraining and cross-validation. IEEE Trans Neural Netw 8 (5), 985–996.
- Bell, A. J., Sejnowski, T. J., 1995. An information-maximization approach to blind separation and blind deconvolution. Neural Computation 7 (6), 1129–1159.
- Bennett, K., Mangasarian, O., 1992. Robust linear programming discrimination of two linearly inseparable sets. Optimization Methods and Software 1, 23–34.
- Bießmann, F., Meinecke, F. C., Gretton, A., Rauch, A., Rainer, G., Logothetis, N., Müller, K.-R., 2009. Temporal kernel canonical correlation analysis and its application in multimodal neuronal data analysis. Machine Learning 79 (1-2), 5—27.
- Birbaumer, N., Mar 2006. Brain-computer-interface research: coming of age. Clin Neurophysiol 117, 479–483.
- Bishop, C., 1995. Neural Networks for Pattern Recognition. Oxford University Press.
- Blanchard, G., Sugiyama, M., Kawanabe, M., Spokoiny, V., Müller, K.-R., 2006. In search of non-Gaussian components of a highdimensional distribution. Journal of Machine Learning Research 7, 247–282.

- Blankertz, B., Curio, G., Müller, K.-R., 2002. Classifying single trial EEG: Towards brain computer interfacing. In: Diettrich, T. G., Becker, S., Ghahramani, Z. (Eds.), Advances in Neural Inf. Proc. Systems (NIPS 01). Vol. 14. pp. 157–164.
- Blankertz, B., Dornhege, G., Krauledat, M., Müller, K.-R., Curio, G., 2007. The non-invasive Berlin Brain-Computer Interface: Fast acquisition of effective performance in untrained subjects. Neuroimage 37 (2), 539–550.
- Blankertz, B., Dornhege, G., Lemm, S., Krauledat, M., Curio, G., Müller, K.-R., 2006a. The Berlin Brain-Computer Interface: Machine learning based detection of user specific brain states. J Universal Computer Sci 12 (6), 581–607.
- Blankertz, B., Kawanabe, M., Tomioka, R., Hohlefeld, F., Nikulin, V., Müller, K.-R., 2008a. Invariant common spatial patterns: Alleviating nonstationarities in brain-computer interfacing. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (Eds.), Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, MA, pp. 113–120.
- Blankertz, B., Lemm, S., Treder, M. S., Haufe, S., Müller, K.-R., 2010. Single-trial analysis and classification of ERP components – a tutorial. NeuroimageIn press.
- Blankertz, B., Müller, K.-R., Curio, G., Vaughan, T. M., Schalk, G., Wolpaw, J. R., Schlögl, A., Neuper, C., Pfurtscheller, G., Hinterberger, T., Schröder, M., Birbaumer, N., 2004. The BCI competition 2003: Progress and perspectives in detection and discrimination of EEG single trials. IEEE Trans Biomed Eng 51 (6), 1044– 1051.
- Blankertz, B., Müller, K.-R., Krusienski, D., Schalk, G., Wolpaw, J. R., Schlögl, A., Pfurtscheller, G., del R. Millán, J., Schröder, M., Birbaumer, N., 2006b. The BCI competition III: Validating alternative approachs to actual BCI problems. IEEE Trans Neural Syst Rehabil Eng 14 (2), 153–159.
- Blankertz, B., Tomioka, R., Lemm, S., Kawanabe, M., Müller, K.-R., Jan. 2008b. Optimizing spatial filters for robust EEG single-trial analysis. IEEE Signal Process Mag 25 (1), 41–56.
- Boser, B., Guyon, I., Vapnik, V., 1992. A training algorithm for optimal margin classifiers. In: Haussler, D. (Ed.), Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory. pp. 144–152.
- Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press, Cambridge, UK.
- Braun, M. L., Buhmann, J., Müller, K.-R., Aug 2008. On relevant dimensions in kernel feature spaces. Journal of Machine Learning Research 9, 1875–1908.
- Breiman, L., Spector, P., 1992. Submodel selection and evaluation in regression: The *x*-random case. International Statistical Review 60, 291–319.
- Brunner, C., Scherer, R., Graimann, B., Supp, G., Pfurtscheller, G., Dec 2006. Online control of a brain-computer interface using phase synchronization. IEEE Trans Biomed Eng 53, 2501–2506.
- Cardoso, J.-F., Souloumiac, A., 1993. Blind beamforming for non gaussian signals. IEE Proceedings-F 140, 362–370.
- Comon, P., 1994. Independent component analysis, a new concept? Signal Process. 36 (3), 287–314.
- Cox, D., O'Sullivan, F., 1990. Asymptotic analysis of penalized likelihood and related estimates. The Annals of Statistics 18 (4), 1676– 1695.
- Cox, D., Savoy, R., 2003. Functional magnetic resonance imaging (fmri) 'brain reading': Detecting and classifying distributed patterns of fmri activity in human visual cortex. NeuroImage 19, 261¢??270.
- Curran, E. A., Stokes, M. J., 2003. Learning to control brain activity: A review of the production and control of EEG components for driving brain-computer interface (BCI) systems. Brain Cogn 51, 326–336.

- Dornhege, G., Blankertz, B., Curio, G., Müller, K.-R., Jun. 2004. Boosting bit rates in non-invasive EEG single-trial classifications by feature combination and multi-class paradigms. IEEE Trans Biomed Eng 51 (6), 993–1002.
- Dornhege, G., Blankertz, B., Krauledat, M., Losch, F., Curio, G., Müller, K.-R., 2006. Combined optimization of spatial and temporal filters for improving brain-computer interfacing. IEEE Trans Biomed Eng 53 (11), 2274–2281.
- Dornhege, G., del R. Millán, J., Hinterberger, T., McFarland, D., Müller, K.-R. (Eds.), 2007. Toward Brain-Computer Interfacing. MIT Press, Cambridge, MA.
- Duda, R., P.E.Hart, D.G.Stork, 2001. Pattern classification, 2nd Edition. John Wiley & Sons.
- Efron, B., Tibshirani, R., 1997. Improvements on cross-validation: The .632+ bootstrap method. Journal of the American Statistical Association 92 (438), 548–560.
- Efron, B., Tibshirani, R. J., 1993. An introduction to the Bootstrap. Vol. 57 of Monographs on Statistics and Applied Probability. Chapman and Hall.
- Farquhar, J., Nov 2009. A linear feature space for simultaneous learning of spatio-spectral filters in BCI. Neural Netw 22, 1278–1285.
- Fawcett, T., 2006. An introduction to ROC analysis. Pattern recognition letters 27 (88), 861–874.
- Ferrez, P., Millán, J., 2005. You are wrong! automatic detection of interaction errors from brain waves. In: 19th International Joint Conference on Artificial Intelligence. pp. 1413–1418.
- Friedman, J. H., 1989. Regularized discriminant analysis. J Amer Statist Assoc 84 (405), 165–175.
- Fukunaga, K., 1972. Introduction to Statistical Pattern Recognition. Academic Press, New York.
- Gentili, R. J., Bradberry, T. J., Hatfield, B. D., Contreras-Vidal, J. L., 2009. Brain biomarkers of motor adaptation using phase synchronization. Conf Proc IEEE Eng Med Biol Soc 1, 5930–5933.
- Golub, G., van Loan, C., 1996. Matrix Computations, 3rd Edition. John Hopkins University Press, Baltimore, London.
- Grosse-Wentrup, M., Buss, M., Aug 2008. Multiclass common spatial patterns and information theoretic feature extraction. IEEE Trans Biomed Eng 55, 1991–2000.
- Grosse-Wentrup, M., Gramann, K., Buss, M., 2007. Adaptive spatial filters with predefined region of interest for EEG based braincomputer-interfaces. In: Schölkopf, B., Platt, J., Hoffman, T. (Eds.), Advances in Neural Information Processing Systems 19. MIT Press, pp. 537–544.
- Guyon, I., Boser, B., Vapnik, V., 1993. Automatic capacity tuning of very large VC-dimension classifiers. In: Hanson, S., Cowan, J., Giles, C. (Eds.), Advances in Neural Information Processing Systems. Vol. 5. Morgan Kaufmann, San Mateo, CA, pp. 147–155.
- Hansen, L., Larsen, J., Nielsen, F., Strother, S., Rostrup, E., Savoy, R., Lange, N., Sidtis, J., Svarer, C., Paulson, O., 1999. Generalizable patterns in neuroimaging: how many principal components? Neuroimage 9 (5), 534–544.
- Harmeling, S., Dornhege, G., Tax, D., Meinecke, F. C., Müller, K.-R., 2006. From outliers to prototypes: ordering data. Neurocomputing 69 (13–15), 1608–1618.
- Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning. Springer.
- Haufe, S., Nikulin, V. V., Ziehe, A., Müller, K.-R., Nolte, G., Aug 2008. Combining sparsity and rotational invariance in EEG/MEG source reconstruction. Neuroimage 42 (2), 726–738.
- Haufe, S., Nikulin, V. V., Ziehe, A., Müller, K.-R., Nolte, G., 2009. Estimating vector fields using sparse basis field expansions. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (Eds.), Advances in Neural Information Processing Systems 21. MIT Press, Cambridge, MA, pp. 617–624.
- Haxby, J., et al, 2001. Distributed and overlapping representations

of faces and objects in ventral temporal cortex. Science 293, 2425¢??-2430.

- Haynes, J., Rees, G., 2006. Decoding mental states from brain activity in humans. Nat Neurosci 7, 523–534.
- Hyvärinen, A., Karhunen, J., Oja, E., 2001. Independent Component Analysis. Wiley.
- Hyvärinen, A., Oja, E., 1997. A fast fixed-point algorithm for independent component analysis. Neural Computation 9 (7), 1483–92.
- Kamitani, Y., Tong, F., 2005. Decoding the visual and subjective contents of the human brain. Nat Neurosci 8 (5), 679¢??685.
- Kimeldorf, G., Wahba, G., 1971. Some results on Tchebycheffian spline functions. J. Math. Anal. Applic. 33, 82–95.
- Koles, Z. J., 1991. The quantitative extraction and topographic mapping of the abnormal components in the clinical EEG. Electroencephalogr Clin Neurophysiol 79 (6), 440–447.
- Koles, Z. J., Soong, A. C. K., 1998. EEG source localization: implementing the spatio-temporal decomposition approach. Electroencephalogr Clin Neurophysiol 107, 343–352.
- Krauledat, M., Shenoy, P., Blankertz, B., Rao, R. P. N., Müller, K.-R., 2007. Adaptation in CSP-based BCI systems. In: Dornhege, G., del R. Millán, J., Hinterberger, T., McFarland, D., Müller, K.-R. (Eds.), Toward Brain-Computer Interfacing. MIT Press, Cambridge, MA, pp. 305–309.
- Kriegeskorte, N., Goebel, R., Bandettini, P., Mar 2006. Informationbased functional brain mapping. Proc Natl Acad Sci U S A 103, 3863–3868.
- Kübler, A., Kotchoubey, B., Dec 2007. Brain-computer interfaces in the continuum of consciousness. Curr Opin Neurol 20, 643–649.
- Kübler, A., Kotchoubey, B., Kaiser, J., Wolpaw, J., Birbaumer, N., 2001. Brain-computer communication: Unlocking the locked in. Psychol Bull 127 (3), 358–375.
- Kübler, A., Müller, K.-R., 2007. An introducton to brain computer interfacing. In: Dornhege, G., del R. Millán, J., Hinterberger, T., McFarland, D., Müller, K.-R. (Eds.), Toward Brain-Computer Interfacing. MIT press, Cambridge, MA, pp. 1–25.
- LaConte, S., Strother, S., Cherkassky, V., Anderson, J., Hu, X., 2005. Support vector machines for temporal classification of block design fmri data. NeuroImage 26, 317¢??–329.
- Lal, T., Schröder, M., Hinterberger, T., Weston, J., Bogdan, M., Birbaumer, N., Schölkopf, B., 2004. Support vector channel selection in BCI. IEEE Trans Biomed Eng 51 (6), 1003–1010.
- Ledoit, O., Wolf, M., 2004. A well-conditioned estimator for largedimensional covariance matrices. J Multivar Anal 88 (2), 365–411.
- Lee, D. D., Seung, H. S., 2000. Algorithms for non-negative matrix factorization. In: NIPS. pp. 556–562.
- Lemm, S., Blankertz, B., Curio, G., Müller, K.-R., 2005. Spatiospectral filters for improving classification of single trial EEG. IEEE Trans Biomed Eng 52 (9), 1541–1548.
- Li, Y., Guan, C., 2006. An extended EM algorithm for joint feature extraction and classification in brain-computer interfaces. Neural Comput 18 (11), 2730–2761.
- MacKay, D. J. C., 2003. Information Theory, Inference, and Learning Algorithms. Cambridge University Press, available from http://www.inference.phy.cam.ac.uk/mackay/itila/.
- Marzetti, L., Gratta, C. D., Nolte, G., 2008. Understanding brain connectivity from EEG data by identifying systems composed of interacting sources. NeuroImage 42 (1), 87 – 98.
- Meinecke, F. C., Ziehe, A., Kurths, J., Müller, K.-R., 2005. Measuring Phase Synchronization of Superimposed Signals. Physical Review Letters 94 (8), 084102.
- Mika, S., Rätsch, G., Müller, K.-R., 2001. A mathematical programming approach to the Kernel Fisher algorithm. In: Leen, T., Dietterich, T., Tresp, V. (Eds.), Advances in Neural Information Processing Systems. Vol. 13. MIT Press, pp. 591–597.
- Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Smola, A., Müller,

K.-R., May 2003. Constructing descriptive and discriminative nonlinear features: Rayleigh coefficients in kernel feature spaces. IEEE Transaction on Pattern Analysis and Machine Intelligence 25 (5), 623–628.

- Mitchell, T. M., Shinkareva, S. V., Carlson, A., Chang, K., Malave, V. L., Mason, R. A., Just, M. A., 2008. Predicting human brain activity associated with the meanings of nouns. Science 320, 1191.
- Moody, J., 1992. The *effective* number of parameters: An analysis of generalization and regularization in non-linear learning systems. In: J. Moody, S. H., Lippman, R. (Eds.), Advances in Neural information processings systems. Vol. 4. Morgan Kaufman, San Mateo, CA, pp. 847–854.
- Morup, M., Hansen, L., Arnfred, S., Lim, L.-K., Madsen, K., 2008. Shift invariant multilinear decomposition of neuroimaging data. Neuroimage 42 (4), 1439–1450.
- Müller, K.-R., Anderson, C. W., Birch, G. E., 2003. Linear and nonlinear methods for brain-computer interfaces. IEEE Trans Neural Syst Rehabil Eng 11 (2), 165–169.
- Müller, K.-R., Krauledat, M., Dornhege, G., Curio, G., Blankertz, B., 2004. Machine learning techniques for brain-computer interfaces. Biomed Tech 49 (1), 11–22.
- Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., Schölkopf, B., May 2001. An introduction to kernel-based learning algorithms. IEEE Neural Networks 12 (2), 181–201.
- Müller, K.-R., Tangermann, M., Dornhege, G., Krauledat, M., Curio, G., Blankertz, B., 2008. Machine learning for real-time single-trial EEG-analysis: From brain-computer interfacing to mental state monitoring. J Neurosci Methods 167 (1), 82–90.
- Murata, N., Amari, S., Yoshizawa, S., 1994. Network information criterion — determining the number of hidden units for an artificial neural network model. IEEE Transactions on Neural Networks 5, 865–872.
- Noirhomme, Q., Kitney, R. I., Macq, B., May 2008. Single-trial EEG source reconstruction for brain-computer interface. IEEE Trans Biomed Eng 55, 1592–1601.
- Nolte, G., Bai, O., Wheaton, L., Mari, Z., Vorbach, S., Hallett, M., 2004. Identifying true brain interaction from eeg data using the imaginary part of coherency. Clinical Neurophysiology 115, 2292– 2307.
- Nolte, G., Meinecke, F. C., Ziehe, A., Müller, K.-R., 2006. Identifying interactions in mixed and noisy complex systems. Physical Review E 73, 051913.
- Nolte, G., Ziehe, A., Nikulin, V., Schlögl, A., Krämer, N., Brismar, T., Müller, K.-R., June 2008. Robustly Estimating the Flow Direction of Information in Complex Physical Systems. Physical Review Letters 100, 234101.
- Orr, G., Müller, K.-R. (Eds.), 1998. Neural Networks: Tricks of the Trade. No. 1524 in LNCS. Springer Heidelberg.
- Parra, L., Alvino, C., Tang, A., Pearlmutter, B., Yeung, N., Osman, A., Sajda, P., Jun 2003. Single-Trial Detection in EEG and MEG: Keeping it Linear. Neurocomputing 52-54, 177–183.
- Parra, L., Alvino, C., Tang, A. C., Pearlmutter, B. A., Yeung, N., Osman, A., Sajda, P., 2002. Linear spatial integration for single trial detection in encephalography. Neuroimage 7 (1), 223–230.
- Parra, L., Christoforou, C., Gerson, A., Dyrholm, M., Luo, A., Wagner, M., Philiastides, M., Sajda, P., 2008. Spatiotemporal linear decoding of brain state. IEEE Signal Process Mag 25 (1), 107–115.
- Parra, L., Sajda, P., 2003. Blind source separation via generalized eigenvalue decomposition. J Mach Learn Res 4 (7-8), 1261–1269.
- Pereira, F., Mitchell, T., Botvinick, M., 2009. Machine learning classifiers and fmri: A tutorial overview. Neuroimage 45 (1, Supplement 1), S199 – S209, mathematics in Brain Imaging.
- Pfurtscheller, G., Neuper, C., Birbaumer, N., 2005. Human Brain-Computer Interface. In: Riehle, A., Vaadia, E. (Eds.), Motor Cortex in Voluntary Movements. CRC Press, New York, Ch. 14, pp.

367-401.

- Poggio, T., Girosi, F., 1990. Regularization algorithms for learning that are equivalent to multilayer networks. Science 247, 978–982.
- Pun, T., Alecu, T. I., Chanel, G., Kronegg, J., Voloshynovskiy, S., Jun 2006. Brain-computer interaction research at the Computer Vision and Multimedia Laboratory, University of Geneva. IEEE Trans Neural Syst Rehabil Eng 14, 210–213.
- Racine, J., 2000. Consistent cross-validatory model-selection for dependent data: hv-block cross-validation. J. Econom. 99 (1), 39–61.
- Ramoser, H., Müller-Gerking, J., Pfurtscheller, G., 2000. Optimal spatial filtering of single trial EEG during imagined hand movement. IEEE Trans Rehabil Eng 8 (4), 441–446.
- Rasmussen, C., Williams, C., 2005. Gaussian Processes for Machine Learning. MIT Press.
- Rätsch, G., Mika, S., Schölkopf, B., Müller, K.-R., Sep. 2002. Constructing boosting algorithms from SVMs: An application to oneclass classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (9), 1184–1199.
- Rätsch, G., Onoda, T., Müller, K.-R., Mar. 2001. Soft margins for AdaBoost. Machine Learning 42 (3), 287–320.
- Sajda, P., Gerson, A., Müller, K.-R., Blankertz, B., Parra, L., 2003. A data analysis competition to evaluate machine learning algorithms for use in brain-computer interfaces. IEEE Trans Neural Syst Rehabil Eng 11 (2), 184–185.
- Schäfer, J., Strimmer, K., 2005. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. Statistical Applications in Genetics and Molecular Biology 4 (1).
- Schlögl, A., Vidaurre, C., Müller, K.-R., 2009. Adaptive methods in BCI research – an introductory tutorial. In: Allison, B., Graimann, B., Pfurtscheller, G. (Eds.), The Frontiers Collection. Springer, in press.
- Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R., 2001. Estimating the support of a high-dimensional distribution. Neural Computation 13 (7), 1443–1471.
- Schölkopf, B., Smola, A., 2002. Learning with Kernels. MIT Press, Cambridge, MA.
- Schölkopf, B., Smola, A., Müller, K.-R., 1998. Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 10 (5), 1299–1319.
- Shao, J., 1993. Linear model selection by cross-validation. J. Am. Stat. Assoc. 88 (422), 486–494.
- Shenoy, P., Krauledat, M., Blankertz, B., Rao, R. P. N., Müller, K.-R., 2006. Towards adaptive classification for BCI. J Neural Eng 3 (1), R13–R23.
- Smola, A., Schölkopf, B., 1998. On a kernel–based method for pattern recognition, regression, approximation and operator inversion. Algorithmica 22, 211–231.
- Smola, A., Schölkopf, B., Müller, K.-R., 1998. The connection between regularization operators and support vector kernels. Neural Networks 11, 637–649.
- Stein, C., 1956. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. Proc. 3rd Berkeley Sympos. Math. Statist. Probability 1, 197-206 (1956).
- Strother, S., Anderson, J., Hansen, L., annd R. Kustra, U. K., Siditis, J., Frutiger, S., Muley, S., LaConte, S., Rottenberg, D., 2002. The quantitative evaluation of functional neuroimaging experiments: The npairs data analysis framework. NeuroImage 15, 747¢??771.
- Sugiyama, M., Krauledat, M., Müller, K.-R., 2007. Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research 8, 1027–1061.
- Tibshirani, R., June 1994. Regression selection and shrinkage via the LASSO. Tech. rep., Department of Statistics, University of Toronto, ftp://utstat.toronto.edu/pub/tibs/lasso.ps.
- Tibshirani, R., 1996. Regression shrinkage and selection via the lasso.

J Royal Statist Soc B 58 (1), 267-288.

- Tikhonov, A., Arsenin, V., 1977. Solutions of Ill-posed Problems. W.H. Winston, Washington, D.C.
- Tomioka, R., Aihara, K., Müller, K.-R., 2007. Logistic regression for single trial EEG classification. In: Schölkopf, B., Platt, J., Hoffman, T. (Eds.), Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, MA, pp. 1377–1384.
- Tomioka, R., Müller, K. R., 2010. A regularized discriminative framework for EEG based communication. Neuroimage 49, 415–432.
- Vanderbei, R., Shanno, D., 1997. An interior point algorithm for nonconvex nonlinear programming. Tech. Rep. SOR-97-21, Statistics and Operations Research, Princeton University.
- Vapnik, V., 1995. The Nature of Statistical Learning Theory. Springer, New York.
- Vidaurre, C., Blankertz, B., 2010. Towards a cure for BCI illiteracy. Brain Topogr 23, 194–198, open Access.
- Vidaurre, C., Krämer, N., Blankertz, B., Schlögl, A., 2009. Time domain parameters as a feature for eeg-based brain computer interfaces. Neural Networks 22, 1313–1319.
- Vidaurre, C., Sannelli, C., Müller, K.-R., Blankertz, B., 2010. Machine-learning based co-adaptive calibration. Neural ComputIn press.
- von Bünau, P., Meinecke, F. C., Király, F., Müller, K.-R., 2009. Finding stationary subspaces in multivariate time series. Physical Review Letters 103, 214101.
- Wolpaw, J., Mar 2007. Brain-computer interfaces as new brain output pathways. J Physiol 579, 613–619.
- Wolpaw, J. R., Birbaumer, N., McFarland, D. J., Pfurtscheller, G., Vaughan, T. M., 2002. Brain-computer interfaces for communication and control. Clin Neurophysiol 113 (6), 767–791.
- Ziehe, A., Müller, K.-R., 1998. TDSEP an efficient algorithm for blind separation using time structure. In: Niklasson, L., Bodén, M., Ziemke, T. (Eds.), Proceedings of the 8th International Conference on Artificial Neural Networks, ICANN'98. Perspectives in Neural Computing. Springer Verlag, Berlin, pp. 675 – 680.
- Ziehe, A., Müller, K.-R., Nolte, G., Mackert, B.-M., Curio, G., January 2000. Artifact reduction in magnetoneurography based on time-delayed second-order correlations. IEEE Trans Biomed Eng 47 (1), 75–87.
- Zou, H., Hastie, T., Tibshirani, R., 2004. Sparse principal component analysis. Journal of Computational and Graphical Statistics 15, 2006.