Available online at www.sciencedirect.com

ScienceDirect

Magnetic Resonance Imaging 28 (2010) 583 – 593

MAGNETIC RESONANCE IMAGING

Classifier ensembles for fMRI data analysis: an experiment

Ludmila I. Kuncheva[a,*], Juan J. Rodríguez[b]

[a]School of Computer Science, Bangor University, LL57 1UT, UK

[b]Departamento de Ingeniería Civil, Universidad de Burgos, 09006 Spain

Received 28 July 2009; accepted 6 December 2009

Abstract

Functional magnetic resonance imaging (fMRI) is becoming a forefront brain–computer interface tool. To decipher brain patterns, fast, accurate and reliable classifier methods are needed. The support vector machine (SVM) classifier has been traditionally used. Here we argue that state-of-the-art methods from pattern recognition and machine learning, such as classifier ensembles, offer more accurate classification. This study compares 18 classification methods on a publicly available real data set due to Haxby et al. [*Science* 293 (2001) 2425–2430]. The data comes from a single-subject experiment, organized in 10 runs where eight classes of stimuli were presented in each run. The comparisons were carried out on voxel subsets of different sizes, selected through seven popular voxel selection methods. We found that, while SVM was robust, accurate and scalable, some classifier ensemble methods demonstrated significantly better performance. The best classifiers were found to be the random subspace ensemble of SVM classifiers, rotation forest and ensembles with random linear and random spherical oracle.

Keywords: Pattern recognition; fMRI; Classifier ensembles; Random subspace; Rotation forest; Random oracle

1. Introduction

Determining how mental states are mapped onto patterns of neural activity has been identified as a key challenge to cognitive neuroscience [1]. Functional magnetic resonance imaging (fMRI) measures blood oxygenation level-dependent (BOLD) signal, thereby providing quantitative data to infer such patterns. Pattern recognition has helped to identify brain patterns that correspond to simple categories such as faces, tools and houses [2]; unseen natural images [3]; even intention [4] and emotion [5]. While pattern recognition and machine learning have already lent a spectrum of tools to fMRI data analysis [6], there are state-of-the-art methods and approaches that have not been explored yet.

The two pattern recognition themes relevant to fMRI analysis are feature selection and classification. Feature selection translates into identification of a set of voxels in the brain that can recognize with the highest accuracy the categories in the fMRI experiment, e.g., positive vs. negative emotion or tools vs. houses or faces [7–8]. In this scenario, the voxels become the features, the stimuli become the class labels, and the brain responses to the stimuli become the instances (objects to classify). Once identified, these voxels are fed into a classifier model, which is trained to predict the category as accurately as possible. While feature selection and classification are intrinsically related, they are often performed separately. For example, relevant voxels can be selected through a univariate statistical method [9] and any classifier model can then be applied. The feature selection aspect of fMRI analysis is difficult for at least two reasons: (i) the feature-to-instance ratio is extremely large, in the order of 5000:1, while in a typical pattern-recognition problem it is expected to be much smaller than 1; (ii) there is a spatial relationship between the features which needs to be taken into account. Multivariate methods have been developed that are more time consuming than the univariate methods but offer higher accuracy and deeper insight into distributed patterns of brain functionality [1,2,6,10,11]. In fact, due to the extremely large dimensionality of the feature space, these methods are pseudo-multivariate. Although a classifier is trained on the large feature set, the weights (parameters of

* Corresponding author. Tel.: +44 1248383661; fax: +44 1248361429.
E-mail addresses: mas00a@bangor.ac.uk (L.I. Kuncheva), jjrodriguez@ubu.es (J.J. Rodríguez).

0730-725X/$ – see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.mri.2009.12.021

the trained classifier) are used individually to determine the importance of a feature. In true multivariate methods the merit of a feature subset should be measured by the classification accuracy of the whole subset, and not by the sum of the individual weights. The reason for this is that a voxel with a low weight may be a key component of a subset, and without that voxel the subset is not as relevant as it appears. For example, SVM will assign low weights to correlated voxels. So if there is a group of extremely important but correlated voxels, they may be left out of the selection in favor of individual uncorrelated voxels of lower relevance. Pereira et al. [6] give a warning to that effect too.

Why is classification important? The primary focus of the fMRI data analysis seems to be finding the activation patterns in the brain representing certain mental states [6]. Classification accuracy takes priority when the mental state needs to be labeled, for example, to provide feedback to the participant in the experiment or to the observing neuroscientist. Accurate recognition is of paramount importance when considering on-line physiological self-regulation of the local BOLD response [12]. This technique, known as *neuro-feedback*, tries to establish voluntary control of circumscribed brain areas. Abnormal activity in such areas may be suppressed through neurofeedback thereby serving as psychophysiological treatment [13,14].

A set of classifier methods used for fMRI classification, including the linear discriminant classifier (LDC), the support vector machine (SVM) classifier [15] and the Gaussian naïve Bayes (NB) [16], have been recently compared [17]. No clear winner has been declared for all the classification tasks; however, SVM appeared to have an edge over the other classifiers across different tasks. Sparse logistic regression has also been shown to be successful in voxel selection and classification, but comparisons with other classifiers have not been provided [18]. SVM [19] has three major advantages over other classifier models: (i) the weights of the trained classifier can be used to rank the features (voxels); this is also true for the LDC and the logistic classifier [20]; (ii) SVM is robust and accurate; and (iii) SVM can be trained and run on thousands of features in reasonable time, which is not true for most other classifier models.

In this article, we argue that pattern recognition and machine learning can offer more accurate classifiers than the currently used ones at the expense of a little increase of the computational complexity. The training of the classifiers may take longer, but the real-time operation will not be adversely affected. Classifier ensembles have proved to be consistently more accurate than individual classifiers across a variety of benchmark and real-life problems [21,22]. Here we use a real dataset from a single-subject fMRI experiment [2]. We apply seven standard voxel selection methods and compare 18 classifier models (seven single classifiers and 11 ensembles) to the currently favorite SVM classifier.

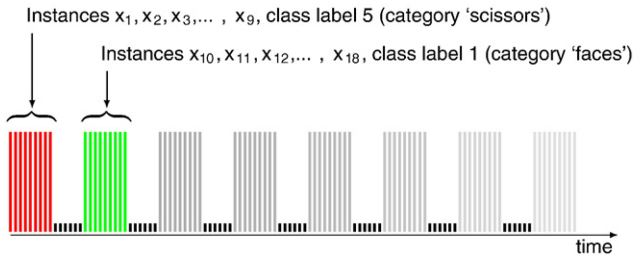

Instances $x_1, x_2, x_3, \ldots, x_9$, class label 5 (category 'scissors')
Instances $x_{10}, x_{11}, x_{12}, \ldots, x_{18}$, class label 1 (category 'faces')
time

Fig. 1. Construction of the data set from one run of the Haxby et al. [2] experiment.

2. Materials and methods

2.1. Data

We used the eight-category data set of Haxby et al. [2] as provided within the MVPA MatLab Toolbox.[1] The data consists of 10 runs of presenting visual stimuli, in the form of images with different types of content, to a single subject. One sample of the whole brain is collected every TR seconds. 'TR' has been accepted in the fMRI literature to mean a 'discrete time point.' There are eight types of stimuli in the experiment (the c=8 classes in our data set): (i) faces, (ii) houses, (iii) cats, (iv) bottles, (v) scissors, (vi) shoes, (vii) chairs and (viii) 'nonsense pictures,' which were random textures. In each experiment, all eight categories were presented in random order. Each image was held for nine TRs, followed by five TRs of rest. The brain sample at every TR was taken as one instance in the data set, with class label corresponding to the stimuli of that TR. Thus each run produces 72 data points, 8 (categories)×9 (TRs). The whole data set (10 runs) contains 720 data points, 90 from each class. Fig. 1 shows how the data set from one run was constructed. The total number of features (voxels) was 43,193.

2.2. Classifier methods

Eighteen classifier methods have been examined. They are listed below and also illustrated in Figs. 2 and 3. Fig. 2 shows the classification regions of the seven *individual* classifier models on a toy data set with two banana-shaped classes. The error estimated on an unseen testing set is given in parentheses underneath the respective plot. The classes are not linearly separable; hence the error of the classifiers that build a linear boundary, including LDC and SVM, is high. The result for the most popular choice — the SVM classifier — is framed for emphasis. Fig. 3 shows the classification regions for the same toy problem obtained from the 11 *ensemble* methods considered here. The results with the ensemble methods are not consistently better than those with the individual classifiers, which reinforces the postulate that no classifier model is ideal for all problems.

[1] Princeton multivoxel pattern analysis manual, https://compmem.princeton.edu/mvpa_docs/.

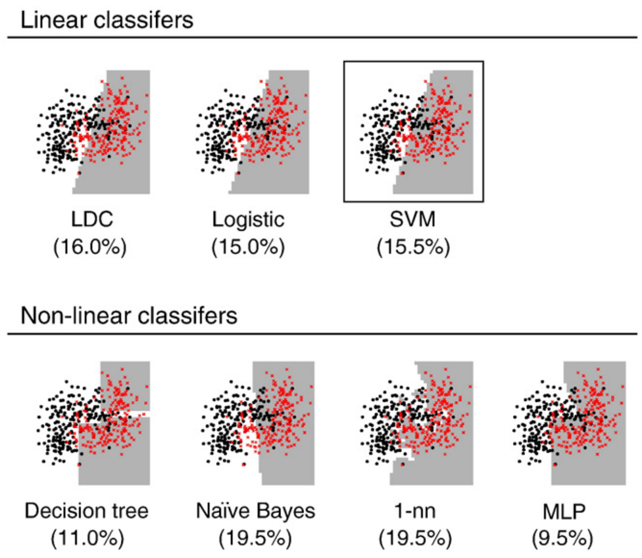


Fig. 2. Classification regions of the 7 individual classifier models on a toy data set with two banana-shaped classes. The testing classification error is given in parentheses.

In the list below, we only give the intuition of the classification methods; the reader is referred to the relevant literature for further details. In the past, classifier models were grouped into parametric and nonparametric [20]. Parametric classifiers were those where we assume the type of the underlying probability distributions and only estimate the parameters of these distributions, which amounts to training the classifiers. Nonparametric classifiers do not require any assumption of the underlying densities. The ‘parametric vs. nonparametric’ distinction causes some confusion because ‘parameters’ are estimated through training of both types of classifiers, and the term ‘parametric’ seems to apply to both groups. Recently, a new grouping has been proposed which roughly covers the former grouping — *generative*, for the parametric group, and *discriminative*, for the nonparametric group [23]. There is no clear indication as to which group should work better with the fMRI data, so we have tried both.

2.2.1. Individual classifiers

2.2.1.1. Linear discriminant classifier. LDC [20] belongs to the generative group of classifiers. The classes are assumed to have normal distributions and equal covariance matrices. Under this assumption, the optimal classifier reduces to calculating linear discriminant functions, one for each class (stimulus in an fMRI experiment). The class label of a given object **x** (brain state at a given time) is determined by the tag of the largest discriminant function.

2.2.1.2. Logistic classifier. The logistic classifier (Log) [20,23–25] is often chosen as the representative example of the group of discriminative classifiers because it approximates the posterior probabilities directly. It is also considered to be “semiparametric” because it does rely on an assumption about the posterior probability densities. The assumption is that for any pair of classes in the problem, the logarithm of the ratio of the two posterior probabilities can be modeled as a linear function of the input variables. Let $\mathbf{x}=[x_1, x_2,..x_n]$ be the n-dimensional input and $P(\omega_i|\mathbf{x})$ be the

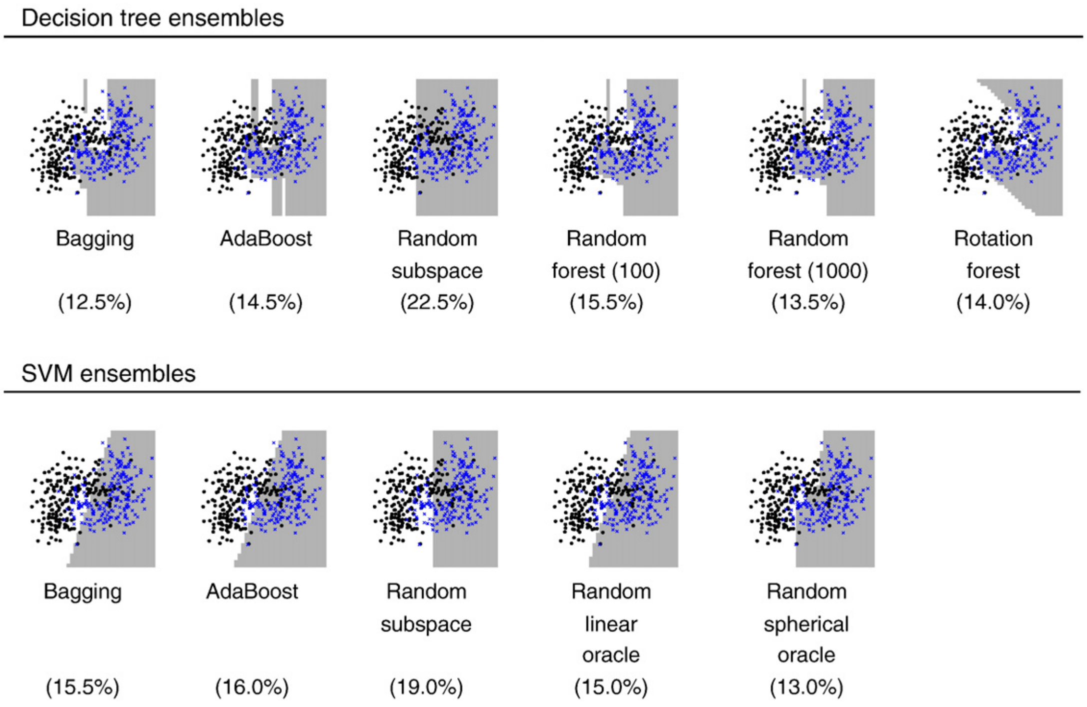


Fig. 3. Classification regions of the 11 ensemble classifier models on the toy data set. The testing classification error is given in parentheses.

probability that the true class of the given object $\mathbf{x}$ is ω_i. Then the assumption for which the logistic classifier is optimal is

$$\log \frac{P(\omega_i|\mathbf{x})}{P(\omega_j|\mathbf{x})} = \sum_{k=1}^{n} \beta_k x_k, \quad \beta_k \in \Re. \tag{1}$$

Training of the logistic classifier is carried out by choosing an arbitrary class as the 'baseline' (ω_j) and finding individual sets of β's for each of the other classes. The discriminant functions for a given $\mathbf{x}$ are evaluated as a linear combination of the inputs using the bespoke class weights [right hand side of Eq. (1)]; the value for the chosen baseline class is 0. The class label of $\mathbf{x}$ is again assigned as the tag of the largest discriminant function. The class regions are separated by piece-wise linear borders, as with LDC.

While the underlying assumptions for LDC and Log may not hold, both classifiers have been found to be fairly robust and accurate.

2.2.1.3. Support vector machines. SVMs [19] have a long history as the most popular classifier for fMRI data analysis, both for classification and feature selection [7,10,26–28]. The most useful version is the one with the linear kernel, whereby the discriminant function between two classes is a linear combination of the inputs. The importance of a feature is associated with the absolute value of the corresponding weight in the linear combination. SVM is originally derived for two classes. If there are c classes, a separate SVM can be trained for each pair of classes, and the $c(c-1)/2$ classifiers are used in conjunction to make up a multiclass SVM classifier. The classification boundary between the two classes is a hyperplane constructed in such a way that it is furthest away from the nearest points from the opposite classes. This maximizes the *margin* of the classifier, which translates into a better generalization performance. SVM is suitable for large dimensionality of the feature space.

2.2.1.4. Decision tree classifier. The small sample size and the large feature dimensionality make it difficult to construct a single accurate decision tree (DT) classifier [29]. We included DT in the experiment because classifier ensembles using DTs have been very successful. DT classifier lends itself to the ensemble paradigm because it is both fast and accurate across a wide range of problems. DT shows the second lowest error on the toy example in Fig. 2 but, being only two-dimensional and with relatively large training data, the example is not indicative to the problems of fMRI data.

Both SVM and DT belong to the discriminative group of classifiers because they do not attempt to model the class-conditional density but try to approximate the classification boundaries. While LDC, Log and SVM produce linear discriminant functions, and therefore linear boundaries between the classes, the standard DT composes the boundaries between the classes using hyperplanes orthogonal to the coordinate axes.

2.2.1.5. Naïve Bayes. NB [16,25] is optimal when the features are conditionally independent, i.e., when the probability density function for class ω_i, denoted $p(\mathbf{x}|\omega_i)$, can be decomposed as $p(\mathbf{x}|\omega_i) = \prod_{k=1}^{n} p(x_k|\omega_i)$. In this case, the densities can be estimated separately for each feature (voxel) which simplifies the training and makes NB feasible for very large feature sets. NB has been deemed "surprisingly accurate" [30] even when the independence assumption is clearly false. NB may produce linear boundaries between the classes. This will happen if the individual densities are assumed to be Gaussian with the same variance (called Gaussian NB with shared variance [16]). Only the means for the c classes need be estimated for each feature. Alternatively, variances for the classes can be estimated together with the means (Gaussian NB with distinct variance).

2.2.1.6. Nearest neighbor. Nearest neighbor (1−nn) is a discriminative classifier that does not need any training beyond specifying a reference labeled data set. The label assigned to $\mathbf{x}$ is the label of the nearest neighbor of $\mathbf{x}$ from the reference set.

2.2.1.7. Multilayer perceptron. Neural networks are versatile and powerful classifiers, but they rely on a number of parameter choices to specify the network architecture and to control the training process. The good result in the toy example is a likely outcome of a serendipitous choice of parameter values. The deficiency of training examples in fMRI experiments as well as the abundance of features makes tuning of the multilayer perceptron (MLP) [31] parameters very difficult.

2.2.2. Ensembles of classifiers

Classifiers that are not very accurate individually but tend to make mistakes on different objects may form a very accurate ensemble [21]. Different routes have been explored in a quest to explain why ensembles work better than single classifiers. Ensembles were found to increase the classification margin, hence reduce the chances of a classification mistake [32], exploit diversity to recover from individual classifier's mistakes [33–35] and reduce both the bias and the variance of a single classifier model [36–39]. Valentini and Dietterich [40] investigate the bias-variance decomposition of the error of the SVM classifier and how ensembles of SVMs may lead to higher accuracy. While the literature offers insights and recommendations for constructing classifier ensembles for 'standard' classification problems, the case of a very small set of examples and massive feature dimensionality (such as fMRI data) has not been fully investigated. Here we take an experimental approach and compare various classifier ensembles of SVMs and DTs to a single SVM classifier.

2.2.2.1. Bagging [37]. This ensemble approach uses a predefined number of classifiers, each one trained on a bootstrap sample of the training data. The label for an object **x** is decided by the majority vote between ensemble members. Other combination methods have also been developed [21]. The 'base' classifier model could be any, but the ensemble benefits most from diverse and accurate classifiers. Hence classifiers whose training is affected by small changes in the training data are more suitable than 'stable' classifiers. Decision trees and MLP ate typical choices for base classifiers, but bagging SVM classifiers have been shown to work well too [41].

2.2.2.2. AdaBoost [42]. This approach builds the ensemble sequentially, one classifier at a time, until a predefined number of classifiers is reached. Each subsequent classifier is trained on a sample explicitly focused on the instances misclassified by the previous classifiers. The ensemble decision is made by weighted voting. The weights are determined by the individual accuracies. Let p be the accuracy of classifier C in the AdaBoost ensemble. The weight for C is calculated as $\log(p/(1-p))$. AdaBoost has been found to be very useful but too sensitive to noise in the data [39].

2.2.2.3. Random subspace [43]. Each classifier in the ensemble is trained on a random subset of features. The subsets can be intersecting or disjoint. The outputs are aggregated by majority vote. Like Bagging and AdaBoost, the random subspace method is only a 'shell' and can be used with any base classifier. Classifiers that are stable with respect to small changes in the training data may become diverse if trained on different subsets of features, for example, 1−nn. The poor results with the toy example, with both types of base classifiers, are due to the fact that with n=2 features, there are only three different nonempty subsets of features, and the ensemble will consist of multiple copies of the same classifiers. A version of the random subspace method has been considered for large-scale feature selection and classification [44].

2.2.2.4. Random forest [45]. This is a version of bagging where the base classifier is a modified DT, termed "random tree". The difference is in the training of the classifier: while DT is a deterministic classifier, random tree is not. Hence substantially different trees can be constructed from identical training data. This introduces additional diversity in the ensemble without compromising the classification accuracy of the individual classifiers. In the toy example, we consider random forests of 100 and 1000 trees.

2.2.2.5. Rotation forest [46]. The base classifier is again a DT. Each tree is built on a bootstrap sample from the data rotated in a random way. To classify a given **x**, the feature values undergo the rotation specific for each DT and the instance is subsequently classified. Let there be L classifiers in the ensemble. Each classifier produces estimates of the c posterior probabilities $P(\omega_i|\mathbf{x})$, i=1,…,c. Denote the posterior probability for class ω_i estimated by classifier j by $P_j(\omega_i|\mathbf{x})$, i=1,…,c. The outputs of the classifiers are combined using the average of the posterior probabilities for the classes. The class label assigned to an object **x** is the one corresponding to the maximum on i of $\sum_{j=1}^{L} P_j(\omega_i|\mathbf{x})$.

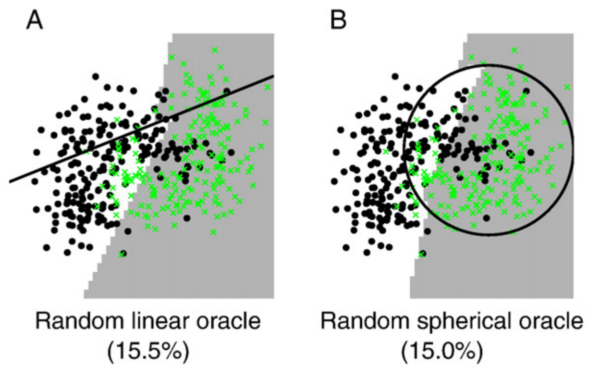


Fig. 4. Classification regions of a classifier with random linear oracle (Subplot A) and the spherical oracle (Subplot B) with SVM as the base classifier.

2.2.2.6. Random oracle [47,48]. Random oracle ensembles can be constructed with any base classifier. The idea is that the training data for each classifier is randomly split into two parts of the space and a separate classifier is trained for each part. The split can be done through a random hyperplane (linear oracle) or a hypersphere (spherical oracle), or in any other way. Thus each ensemble member is itself a mini-ensemble with two classifiers. Since the random oracle ensemble methods are less well known, we include an example using the two-class toy data. Fig. 4 shows examples of the classification regions produced by classifiers built using the linear and the spherical oracle. The base classifier for both is SVM. In each subspace, SVM builds a linear boundary between the classes. Due to the split, the classification boundary is no longer linear in the whole space, which illustrates the flexibility introduced by the random oracle. The individual accuracies are similar to that of SVM, which begs the question why we are using the oracle at all. The answer is that the oracle induces diversity. Different random splits will produce diverse classification regions with little or no reduction of individual accuracy. Diversity is a desirable property of the ensemble, which is demonstrated by the better ensemble results on the toy example in Fig. 3.

Classifier ensembles are not designed with a view to approximate probability density functions, so they can be considered in the discriminative group of classifiers.

2.3. Evaluation of performance and comparing classifiers

Ideally, the classifiers will be trained on part of the data and tested on unseen part of the data. Since the sample size is extremely small compared to the feature dimensionality, in most cases, having a separate unseen data set for testing is a luxury. "Peeking" refers to using the testing data in the training process [6], be it just for normalization of the data, or even worse, for preselection of active voxels, on which refined selection methods are subsequently tested.

In a K-fold cross-validation procedure (CV), one fold of the data is set aside, and all the training (voxel selection and classification) is done on the other $K-1$ folds pooled as one training set. This set can be split further into training and pseudo-testing parts, but the fold that was left out should not be seen for any testing or parameter tuning.[2] Golland and Fischl [49] criticize the use of the cross-validation tests for comparing classifiers. They point out that, while the mean classification accuracy across the CV folds is an unbiased estimate of the true accuracy, the variance may be optimistically biased for many classifiers. This leads to discovering differences that do not exist. The authors propose to use (nonparametric) permutation tests.

Similar argument is raised by Nadeau and Bengio [50] who propose a correction for the variance that reduces or eliminates the optimistic bias. The new estimate of the variance is more conservative and leads to a test with a specified size (chosen level of significance) and good power (low error rate in accepting that there is no difference if there is one). Consider a sample of K testing accuracies produced from a K-fold cross-validation. Let σ_μ be the sample standard deviation and $\sigma_{\hat{\mu}}$ be the standard error of the mean traditionally calculated as $\sigma_{\hat{\mu}} = \frac{\sigma_\mu}{\sqrt{K}}$. Nadeau and Bengio [50] propose instead

$$\sigma_{\hat{\mu}} = \sigma_\mu \sqrt{\frac{1}{K} + \frac{N_{\text{testing}}}{N_{\text{training}}}} \tag{2}$$

where N_{training} and N_{testing} are the sizes of the training and the testing sets, respectively

$$N_{\text{training}} = (K-1) \times \text{fold size}$$
$$N_{\text{testing}} = \text{fold size}.$$

This correction is implemented in the Weka system [51], which we use for all the experiments.[3]

2.4. Experimental protocol

2.4.1. Number of voxels

Univariate selection methods are widely used because of the easy way to attach statistical significance to the individual voxels. A test statistic is calculated to express the discriminative power of the voxel, e.g., t test or Wilcoxon rank-sum test (ANOVA and Kruskal–Wallis for multiple classes). A threshold is then applied to select the most relevant subset of voxels. The threshold can be the chosen significance level or a corrected value thereof. Correction methods include the Bonferroni correction for multiple comparisons, false discovery rate, random field theory [9] and more. It is difficult for the nonexpert to decide on the test statistic, significance level and correction method. Palatucci and Carlson [52] argue that depending on these choices, one may end up with a dramatically different number of selected voxels. The authors question the ability of such feature selection criteria to produce a highly discriminative, robust and interpretable voxel selection and propose an alternative criterion based on order statistics. In our experiments, we were faced with this problem too. We calculated the ANOVA p value for all 43,193 voxels. With no correction, at significance level .05, there were 37,422 significant voxels. This number was calculated as the average across 10-fold cross-validation run of the data, where one run was left aside, and ANOVA was calculated on the remaining nine runs. With Bonferroni correction for the total number of voxels, we got 14,368 relevant voxels. We applied further Bonferroni correction for the number of pairwise comparisons between the classes (28), which left us with 10,264 voxels. If, however, we calculated ANOVA for each class vs. all other classes together (t test for this case), did not correct for multiple comparisons, and took the intersection of voxels at significance level .05, the average dropped to 33 voxels. These results reveal the variability of the size of the selected set and led us to using a preset numbers of voxels, as also proposed in Ref. [6]. The sizes that we chose were M={5,20,50,100,200,1000}. The average size picked by the intersection method (33 voxels) was also included.

2.4.2. Voxel selection methods

We chose the seven voxel selection methods explained below.

Let $v_1,\ldots,v_n$ be the voxels in the brain image and $\omega_1,\ldots,\omega_c$ be the classes (corresponding to stimuli). Denote by $m_i(v)$ the mean of the values of voxel v for class i, and by $\sigma_i(v)$ the standard deviation for the class, where i=1,2,…,c. Denote by

$$A_i(v) = \frac{m_i(v)}{\sqrt{\frac{\sigma_i^2(v)}{N_i}}}$$

the "activation" of voxel v with respect to class ω_i, where N_i is the number of objects from that class. To construct reduced voxel sets we used two measures of activation [8].

(a). Activation (sum). Choose the M top-ranked voxels according to $\sum_{i=1}^{c} A_i(v)$.

(b). Activation (per class). The voxels are ranked individually for each class using $A_i(v)$. The ranked lists are merged and the top M voxels are selected as the final voxel set.

(c). ANOVA. The voxels are sorted in ascending order of the p value and the top M voxels are selected.

[2] The terms 'testing set' and 'validation set' have been used in the literature to denote both the unseen testing data and the surrogate testing data cut from the training set. To avoid confusion, we suggest to call the fold set aside *the testing* data and the part cut from the training data, *the pseudo-testing* data.

[3] Weka is a collection of machine learning algorithms for data mining tasks. It is open source software issued under the GNU General Public License.

(d). SVM. An SVM classifier is trained using all voxels. The weights of the trained classifier (one for each voxel) are sorted in descending order of their absolute values. The voxels corresponding to the top M weights are selected. For multiple classes, the voxels are ranked separately for each class, using a one-vs.-all method. The rank lists are merged and the top M voxels are selected.

(e). RFE. The recursive feature evaluation method (RFE) [53] starts by training an SVM classifier on all the voxels. A prespecified number of voxels T, with the smallest weights, are discarded. Another SVM classifier is trained on the remaining voxels, and the elimination step is carried out again. The training-elimination steps are run until the remaining set of voxels cannot be reduced further. This means that if T voxels are taken away, the remaining set will contain fewer voxels than the desired number M. For this last set, we can either eliminate one feature at a time or simply take away the excess number of voxels in one step. In our experiment, T varies from one step to the next; at each iteration, we eliminate 5% of the *remaining* voxels.

(f). Activation+RFE (SVM). As recommended by De Martino et al. [8], we apply RFE to the 2000 voxels selected through the activation (per class) method explained above.

(g). RFE+SFS. In this method, we first apply RFE to preselect 2000 voxels and then reduce the set using the sequential forward selection (SFS) procedure [54]. SFS operates by constructing the feature set progressively, starting from an empty set and ending with a set with M features. The first feature that enters the set is the individually best feature. The second feature is the one that makes the best pair with the already selected feature. In order to do that, all pairs containing the already selected feature are examined. Thus if we start with 2000 features, after selecting the best feature, 1999 pairs need to be examined. A third feature is added in the same way, and so on until M features are selected. In the smaller scale problems, the merit of a subset of features is typically evaluated by the classification accuracy of a chosen classifier using only that subset of features. For large dimensionality, as in this study, simpler evaluation criteria are employed [55]. SFS has quadratic complexity on the number of features, which makes it infeasible on the whole brain data.

We acknowledge the existence of many more voxel selection methods and approaches. Note that the purpose of this study was not to examine voxel selection methods but to compare classification methods on an already selected voxel subset.

2.4.3. Classifiers

The 18 classifier models detailed above were used in cross-validation experiments. We ran a 10-fold CV, whereby one run of the data was set aside in each CV fold, and the nine remaining runs were pooled as the training data. No pseudo-training was carried out. All classifiers were trained using their default parameter values in Weka. Note that Weka offers two SVM implementations called SMO and LIBLINEAR [56]. We found that better results were obtained with LIBLINEAR, so we used that version in all the comparisons. Rotation forest has recently been added to Weka.[4] Random oracle classifier ensembles, however, are not distributed with the standard Weka version. Java implementations of the oracle ensemble methods, compatible with Weka, are available by request from the authors.

The classification accuracy was calculated as the average across the cross-validation folds. All classifiers were compared to the SVM using the corrected variance [50].

3. Results

We carried out 18 (classifiers)×7 (selection methods)×7 (voxel set sizes)=882 CV experiments. The accuracy of each classifier is stored in a 7×7 table. The rows of the table are the voxel selection methods, and the columns are the voxel set sizes. As an illustration, Table 1 shows the tables with values of the classification accuracy, averaged across the cross-validation folds for the single SVM classifier and the random spherical oracle ensemble.

Instead of including an overwhelming number of tables, we decided to visualize the data in a nontraditional way. Tables 2–4 present the results. The matrix on the left gives a color-coded result from the comparison between the given classifier and SVM. As in Table 1, the columns are the seven voxel set sizes, and the rows are the voxel selection methods (a)–(g). The comparison is done using the t test with the corrected variance. A red square indicates that the classifier for the table is significantly better than SVM (significance level .05) for the respective pair of set size and selection method. Green shading indicates that the classifier has been more accurate than SVM in the experiment, but the difference was not found significant. Black indicates that the SVM classifier was significantly better than the classifier of the table.[5] Therefore, if the predominant color of a table is red or green, the classifier is consistently better than SVM.

Clearly, the accuracy of the classifiers depends on the voxel selection method. Counterintuitive as it may be, in order to get a summarized view over the classifier performance, we averaged the classification accuracies across the seven selection methods (took the average of each column) and called this the ‘overall accuracy’. Thus each classifier is represented by seven values of overall accuracy corresponding to the voxel set sizes. To the right of each subtable we plot the SVM overall accuracy (black dots)

[4] Weka can be downloaded from http://www.cs.waikato.ac.nz/ml/weka.

[5] All tables with the *values* of the classification accuracy, averaged across the cross-validation folds, are available from the authors.

Table 1
Classification accuracy for the SVM and the random spherical oracle

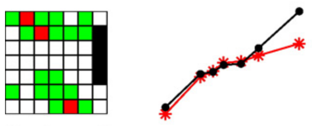
SVM	5	20	33	50	100	200	1000
(a)	9.03	8.19	11.25	14.31	17.78	16.67	29.03
(b)	32.50	37.50	35.83	41.81	42.78	47.36	57.92
(c)	21.53	30.97	30.00	31.81	34.72	43.06	65.69
(d)	38.47	58.47	60.28	63.19	62.78	64.44	69.86
(e)	35.56	53.47	51.94	52.36	54.17	59.03	63.33
(f)	36.39	46.94	47.22	48.61	51.11	55.83	64.17
(g)	46.25	52.78	55.97	54.72	45.83	54.86	67.08
Random Spherical Oracle							
	5	20	33	50	100	200	1000
(a)	10.00	9.17	10.14	12.50	16.81	16.81	27.36
(b)	33.47	37.64	36.25	45.97	46.39	52.64	58.89
(c)	22.78	29.86	28.47	31.53	36.39	43.75	66.67
(d)	39.44	59.17	62.22	64.17	69.17	71.25	72.22
(e)	36.67	54.31	53.61	55.83	59.31	63.75	64.58
(f)	37.64	47.22	48.89	52.08	55.97	59.58	67.36
(g)	47.36	55.42	55.97	56.39	53.47	60.56	68.75

The column headings are voxel set sizes; (a)–(g) are voxel selection methods. Each entry is the testing accuracy averaged across the cross-validation folds. The shaded values indicate that the random spherical oracle is significantly better than SVM.

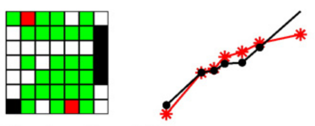
Table 3
Comparison of *ensemble* classifiers using DTs to the single SVM classifier

Ensemble classifiers - decision trees

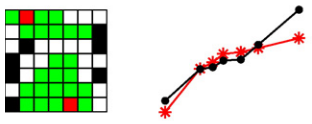
Bagging



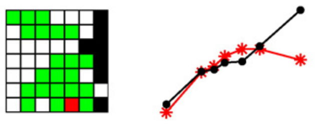
Boosting



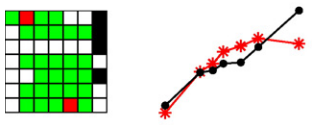
Random subspace



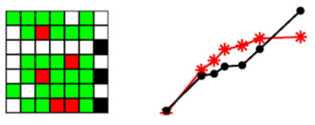
Random Forest 100



Random Forest 1000



Rotation Forest



as well the overall accuracy of the classifier of the table (red stars), as functions of the logarithm of the voxel set sizes. Classifiers are better than SVM when the red curve is consistently above the black one.

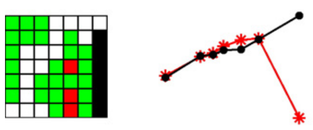
In addition, we plot in Fig. 5 the voxel set responsible for the best testing classification accuracy, 73.19%, obtained by the random forest ensembles with 1000 trees, using 200 voxels selected through one run of the plain SVM [voxel selection method (e)]. We took the intersection of the 10 sets of 200 voxels obtained in the cross-validation experiment and obtained 93 voxels common for all the selections. To display the result, we scanned the brain slices on all three axes and chose the slice with the largest number of selected voxels to show in the figure. The blue hair-cross in the plots indicates the cut-off position of the displayed slices.

Finally, in order to draw up a numerical ranking of the 18 classification methods, we calculated the average rank of each method. Consider the 49 comparisons in each table as independent tests. Each classifier obtains a rank for each cell, i.e., for each combination of voxel selection method (row) and number of voxels (column). The accuracies are arranged in descending order. The most accurate classifier receives Rank 1, the second best receives Rank 2, and so on. If there is a tie, the ranks are shared, so that the total sum of the ranks is the same (171 for 18 methods). Then the average rank for each classifier is calculated across the 49 tests. The best

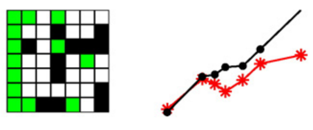
Table 2
Comparison of *individual* classifiers to SVM

Individual classifiers

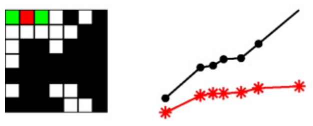
LDC



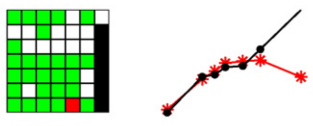
Logistic



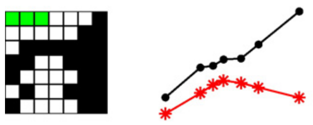
Decision tree



NB



1-nn



MPL

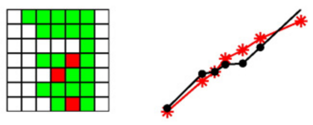
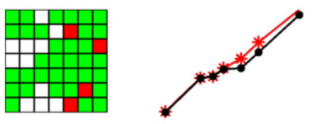


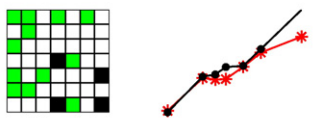
Table 4
Comparison of *ensemble* classifiers using SVM to the single SVM classifier

Ensemble classifiers - SVM

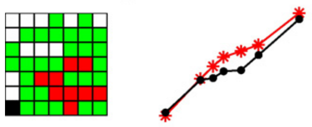
Bagging



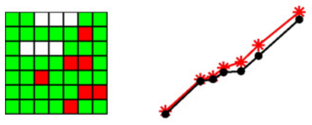
Boosting



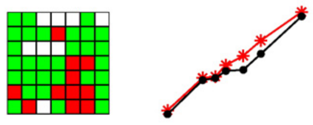
Random sub space



Random Linear Oracle



Random Spherical Oracle



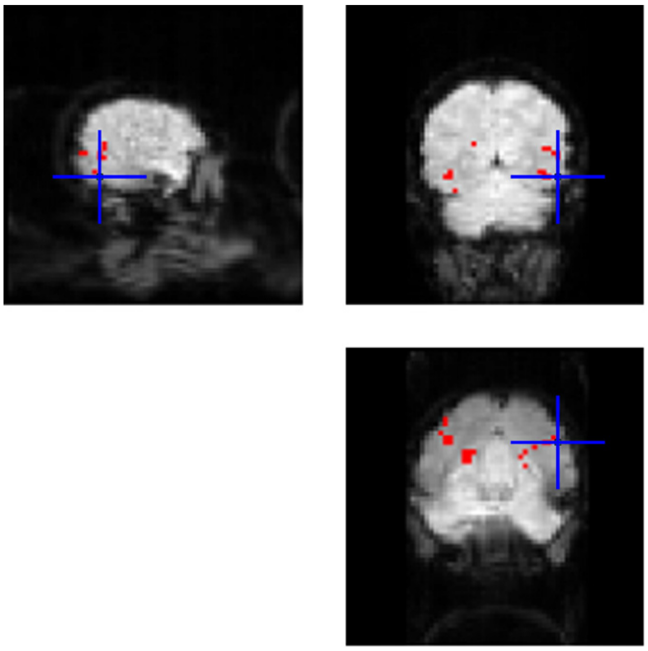


Fig. 5. The voxel subset with the best testing classification accuracy [random forest with 1000 trees on 200 voxels selected through SVM method (d)].

classifier will be the one with the lowest average rank. Table 5 shows the result from this calculation.

4. Discussion

Looking for a "best" classifier is an ill-posed problem because there is no one classifier that is best for all types of data. The results reveal that ensemble classifiers are not universally better than SVM. As expected, SVM is among the best single classifier models studied here. It dominates demonstrably the 1−nn classifier and the DT classifier, which are often praised for their accuracy and robustness. This is seen in Table 2, where the subtables for 1−nn and DT contain copious black cells, indicating that SVM is significantly better than the classifier of the table. SVM is also better than the logistic classifier: while there are 12 out of 49 comparisons where SVM wins with a significant result, there is none where Log wins. SVM is roughly on the par with NB and the LDC. However, both NB and LDC fail badly on the largest voxel set (1000 voxels). SVM scales quite well with the number of features which makes it the current favorite for fMRI data. The only single classifier that showed a consistently better result than SVM in this experiment was the MLP. MLP could be very accurate or could be lost if its parameters are not well tuned. MLP appeared to be the slowest method to train among all classifiers including the classifier ensembles. We used the default parameter setting in Weka: one hidden layer where the number of neurons is the average of the number of features and the number of classes; learning rate 0.3; momentum 0.2, 500 training epochs.

Table 5
Average ranks of the 18 classification methods

Method	Rank
Random subspace (SVM)	4.97
Rotation forest	5.57
Random spherical oracle (SVM)	6.12
Random linear oracle (SVM)	6.41
Random forest (1000 trees)	7.51
Bagging (SVM)	7.58
Multilayer perceptron (MLP)	8.85
Boosting (DT)	8.90
Linear discriminant classifier (LDC)	9.34
SVM	9.59
Random subspace (DT)	9.76
Naïve Bayes (NB)	9.92
Random forest (100 trees)	10.06
Boosting (SVM)	10.94
Bagging (DT)	10.96
Logistic classifier	11.88
Decision tree (DT)	16.33
Nearest neighbor (1−nn)	16.33

Ensemble classifiers are expected to fare better than single classifiers. Ensembles with DTs live to this expectation but invariably come worse than SVM on the largest voxel subset, regardless of the voxel selection method (Table 3). The overall accuracy curves reinforce this observation by the drop of the very last point of the ensemble curve in each graph. This is indicative of the possibility to overtrain sophisticated classifiers, such as ensembles, while single classifiers could be immune to overtraining for similar feature set sizes. The best among the DT ensembles were the rotation forest [46] and random forest with 1000 DTs. However, only the rotation forest ensemble from this group scores more significant wins (5) than suffers significant losses (3) out of the 49 comparisons with SVM. All the losses are for the largest voxel subset, which gives food for thoughts about modification of the rotation forest method that will scale better with the feature set size.

The best ensemble group is the one where SVM is used as the base classifier in the ensemble (Table 4). Apart from boosting, all ensembles are better than the single SVM (green cells correspond to better but nonsignificant accuracy, while the red cells indicate significant difference in favor of the classifier of the subtable). The graphs on the right may suggest that the ensemble is only marginally better than SVM, but significant differences were found for the individual voxel selection methods, even by the conservative correction of the variance. Boosting, declared in the past to be "the best off-the-shelf classifier" [57], fails, suggesting that the data contains substantial noise. Boosting has been known for its low tolerance to noise [39]. The best ensemble in this group was the spherical random oracle [48] with 11 wins, 0 losses and 38 draws, of which only eight comparisons were in favor of SVM, while in the remaining 30 comparisons the random oracle was better.

The overall ranking places the random subspace ensemble with SVM classifiers on the top of the list, followed by rotation forest and the two types of random oracles. SVM is in the middle of the table, leaving behind most individual classification methods and most ensembles with DTs. This shows that, among the single classifiers, SVM is justly favored for the fMRI data analysis. SVM had a definite advantage to single classifiers and ensembles alike for large voxel set sizes, which raises the question and the perspective for developing more scalable classifiers and ensembles. On the other hand, recent ensemble methods were found to be better than SVM in our experiments.

Computational complexity of ensemble classifiers is obviously higher than that of a single SVM classifier but not by much. Note that training complexity is not an issue here because we are looking for a classifier that can be run on-line in an fMRI experiment. Running time of an ensemble of classifiers is determined by the time of calculating the outputs of multiple SVMs plus a little overhead for the combination of the individual decisions. Ensembles lend themselves naturally to parallel computing, so the running complexity is not likely to be an obstacle.

We found that different voxel selection methods led to dramatically different classification accuracies. This can be seen in Table 1, where the classification accuracy for 200 selected voxels varies from 17% to 71% for the same classifier, depending on the voxel selection method. We did not set off to compare voxel selection methods but rather to apply some popular choices and compare classification methods on these.

Modern machine learning and pattern recognition can offer sophisticated classifiers, but there is no reason to stop at this. There are intricate and accurate feature selection methods, for example, floating search [58], which are yet to be extended to cope with large feature sets and rival the currently favorite RFE-based methods.

References

[1] Norman KA, Polyn AM, Detre GJ, Haxby JV. Beyond mind-reading: multi-voxel pattern analysis of fMRI data. Trends Cogn Sci 2006;10:424–30.

[2] Haxby JV, Gobbini M, Furey ML, Ishal A, Schouten JL, Pietrini P. Distributed and overlapping representation of faces and objects in ventral temporal cortex. Science 2001;293:2425–30.

[3] Kay KN, Naselaris T, Prenger RJ, Gallant JL. Identifying natural images from human brain activity. Nat Lett 2008;452:352–5.

[4] Haynes J-D, Sakai K, Rees G, Gilbert S, Frith C, Passingham RE. Reading hidden intentions in the human brain. Curr Biol 2007;17:323–8.

[5] Hardoon DR, Mourao-Miranda J, Brammer M, Shawe-Taylor J. Unsupervised analysis of fMRI data using kernel canonical correlation. NeuroImage 2007;37(4):1250–9.

[6] Pereira F, Mitchell T, Botvinick M. Machine learning classifiers and fMRI: a tutorial overview. NeuroImage 2009;45(1, Supplement 1):S199–209.

[7] Liang LC, Cherkassky V, Rottenberg DA. Spatial SVM for feature selection and fMRI activation detection. Proceedings of the IEEE International Joint Conference on Neural Networks, Vancouver, Canada; 2006. p. 1463–9.

[8] De Martino F, Valente G, Staeren N, Ashburner J, Goebel R, Formisano E. Combining multivariate voxel selection and support vector machines for mapping and classification of fMRI spatial patterns. NeuroImage 2008;43(1):44–58.

[9] Friston K, Ashburner J, Kiebel S, Nichols T, Penny W, editors. Statistical Parametric Mapping: The Analysis of Functional Brain Images. London: Academic Press; 2007. p. 465–9.

[10] Clithero JA, Carter RM, Huettel SA. Local pattern classification differentiates processes of economic valuation. NeuroImage 2009;45 (4):1329–38.

[11] Kamitani Y, Tong F. Decoding the visual and subjective contents of the human brain. Nat Neurosci 2005;8:679–85.

[12] Mitchell T, Hutchinson R, Just M, Niculescu R, Pereira F, Wang X. Classifying instantaneous cognitive states from fMRI data. American Medical Informatics Association Symposium; 2003. p. 465–9.

[13] Weiskopf N, Scharnowski F, Veit R, Goebel R, Birbaumer N, Mathiak K. Self-regulation of local brain activity using real-time functional magnetic resonance imaging (fMRI). J Physiol 2004;98 (4-6):357–73.

[14] Weiskopf N, Sitaramb R, Josephsa O, Veitb R, Scharnowskid F, Goebele R, et al. Real-time functional magnetic resonance imaging: methods and applications. Magn Reson Imaging 2007;25:989–1003.

[15] Cox DD, Savoy RL. Functional magnetic resonance imaging (fmri) : detecting and classifying distributed patterns of fmri activity in human visual cortex. NeuroImage 2003;19(2):261–70.

[16] Mitchell T, Hutchinson R, Niculescu R, Pereira F, Wang X, Just M, et al. Learning to decode cognitive states from brain images. Mach Learn 2004;57(1-2):145–75.

[17] Ku S-P, Gretton A, Macke J, Logothetis NK. Comparison of pattern recognition methods in classifying high-resolution BOLD signals obtained at high magnetic field in monkeys. Magn Reson Imaging 2008;26(7):1007–14.

[18] Yamashita O, Sato M-A, Yoshioka T, Tong F, Kamitani Y. Sparse estimation automatically selects voxels relevant for the decoding of fMRI activity patterns. NeuroImage 2008;42:1414–29.

[19] Burges CJC. A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov 1998;2(2):121–67.

[20] Duda RO, Hart PE, Stork DG. Pattern Classification. 2nd ed. New York: John Wiley & Sons; 2001.

[21] Kuncheva LI. Combining Pattern Classifiers. Methods and Algorithms. N.Y.: John Wiley and Sons; 2004

[22] Brown G. Ensemble learning. In: Sammut C, Webb G, editors. Encyclopedia of Machine Learning. Springer; 2009. http://www.springer.com/computer/artificial/book/978-0-387-30768-8.

[23] Ng A, Jordan M. On discriminative vs. generative classifiers: a comparison of logistic regression and naive Bayes. Advances in Neural Information Processing Systems (NIPS); 2001. p. 841–8.

[24] Hastie T, Tibshirani R, Friedman J. The Elements of Statistical Learning. New York: Springer; 2001.

[25] Mitchell T. Machine learning, McGraw Hill, 1997, draft chapter (2005): generative and discriminative classifiers: naive Bayes and logistic regression. http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf.

[26] LaConte S, Strother S, Cherkassky V, Anderson J, Hu X. Support vector machines for temporal classification of block design fmri data. NeuroImage 2005;26(2):317–29.

[27] Mourao-Miranda J, Bokde AL, Born C, Hampel H, Stetter M. Classifying brain states and determining the discriminating activation patterns: support vector machine on functional mri data. NeuroImage 2005;28(4):980–95.

[28] Mourao-Miranda J, Reynaud E, McGlone F, Calvert G, Brammer M. The impact of temporal compression and space selection on SVM analysis of single-subject and multi- subject fMRI data. NeuroImage 2006;33(4):1055–65.

[29] Quinlan JR. C4.5: Programs for Machine Learning. San Mateo, Calif: Morgan Kaufmann; 1993.

[30] Hand DJ, Yu K. Idiot's Bayes — not so stupid after all? Int Stat Rev 2001;69:385–98.

[31] Bishop C. Neural Networks for Pattern Recognition. Oxford: Clarendon Press; 1995.

[32] Allwein EL, Schapire RE, Singer Y. Reducing multiclass to binary: a unifying approach for margin classifiers. J Mach Learn Res 2000;1: 113–41.

[33] Tumer K, Ghosh J. Error correlation and error reduction in ensemble classifiers. Connect Sci 1996;8(3/4):385–404.

[34] Kuncheva LI, Whitaker CJ. Measures of diversity in classifier ensembles. Mach Learn 2003;51:181–207.

[35] Kuncheva L, Whitaker C, Shipp C, Duin R. Limits on the majority vote accuracy in classifier fusion. Pattern Anal Appl 2003;6:22–31.

[36] Friedman N, Geiger D, Goldszmid M. Bayesian network classifiers. Mach Learn 1997;29(2):131–63.

[37] Breiman L. Bagging predictors. Mach Learn 1996;26(2):123–40.

[38] Domingos P, Pazzani M. On the optimality of the simple Bayesian classifier under zero-one loss. Mach Learn 1997;29:103–30.

[39] Bauer E, Kohavi R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Mach Learn 1999;36: 105–42.

[40] Valentini G, Dietterich TG. Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. J Mach Learn Res 2004:725–75.

[41] Valentini G. Random aggregated and bagged ensembles of SVMs: an empirical bias-variance analysis. In: Roli F, Kittler J, Windeatt T, editors. Fifth International Workshop on Multiple Classifier Systems, Vol. 3077. Lecture Notes in Computer Science; 2004. p. 263–72.

[42] Freund Y, Schapire RE. Experiments with a new boosting algorithm. Thirteenth International Conference on Machine Learning. San Francisco: Morgan Kaufmann; 1996. p. 148–56.

[43] Ho TK. The random space method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell 1998;20(8):832–44.

[44] Lai C, Reinders MJ, Wessels L. Random subspace method for multivariate feature selection. Pattern Recogn Lett 2006;27(10): 1067–76.

[45] Breiman L. Random forests. Mach Learn 2001;45:5–32.

[46] Rodríguez JJ, Kuncheva LI, Alonso CJ. Rotation forest: a new classifier ensemble method. IEEE Trans Pattern Anal Mach Intell 2006;28(10):1619–30.

[47] Kuncheva L, Rodríguez JJ. Classifier ensembles with a random linear oracle. IEEE Trans Knowl Data Eng 2007;19(4):500–8.

[48] Rodriguez JJ, Kuncheva LI. Naïve Bayes ensembles with a random oracle. Proc 7th International Workshop on Multiple Classifier Systems, MCS'07, Vol. LNCS 4472, Prague, Czech Republic; 2007. p. 450–8.

[49] Golland P, Fischl B. Permutation tests for classification: towards statistical significance in image-based studies. Proceedings of the International Conference on Information Processing in Medical Imaging (IPMI); 2003. p. 330–41.

[50] Nadeau C, Bengio Y. Inference for the generalization error. Mach Learn 2003;62:239–81.

[51] Witten IH, Frank E. Data Mining: Practical Machine Learning Tools and Techniques. 2nd ed. San Francisco (CA): Morgan Kaufmann; 2005.

[52] Palatucci M, Carlson A. On the chance accuracies of large collections of classifiers. Proceedings of the 25th International Conference on Machine Learning; 2008. p. 744–51.

[53] Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Mach Learn 2002;46: 389–422.

[54] Stearns S. On selecting features for pattern classifiers. 3-d International Conference on Pattern Recognition, Coronado, CA; 1976. p. 71–5.

[55] Hall MA. Correlation-based feature subset selection for machine learning, Ph.D. thesis, University of Waikato, Hamilton, New Zealand (1998).

[56] Fan R-E, Chang K-W, Hsieh C-J, Wang X-R, Lin C-J. LIBLINEAR — a library for large linear classification. http://www.csie.ntu.edu.tw/~cjlin/liblinear/2008.

[57] Breiman L. Arcing classifiers. Ann Stat 1998;26(3):801–49.

[58] Pudil P, Novovičová J, Kittler J. Floating search methods in feature selection. Pattern Recogn Lett 1994;15:1119–25.