MEG Mind Reading: Strategies for Feature Selection

Heikki Huttunen, Tapio Manninen and Jussi Tohka
Department of Signal Processing, Tampere University of Technology

Abstract

The regularized logistic regression classifier has shown good performance in problems where feature selection is critical, including our recent winning submissions to the ICANN2011 MEG mind reading challenge [Huttunen et al. 2011; Huttunen et al. 2012], and to the DREAM 6 AML classification challenge [Manninen et al. 2011]. The benefit of the method is that it includes an embedded feature selection step, which automatically selects a good subset of input features, thus, simplifying the classifier and improving the generalization. However, explicit wrapper feature selection methods, such as the forward and backward feature selection, are also widely used in similar problems. In this paper, we compare the efficiency of the elastic net regularized logistic regression classifier with the support vector machine classifier in combination with various sequential feature selection methods.

Keywords: Classification, feature selection, MEG, Simulated Annealing, Logistic Regression, Elastic Net

1 Introduction

The task in the supervised classification is to make predictions about the class of an unknown object (represented by a feature vector $\mathbf{x}$) given a training set of P-dimensional feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ with known class memberships. An important special case of supervised classification problems arises when the number of features P is larger or nearly as large than the number of training samples N. These classification problems are increasingly important in genomics and neuroimaging [Saeys et al. 2007; Pereira et al. 2009]. These setups require efficient methods for feature selection, i.e., the selection of the subset of the most relevant features among the P possible features. To give a concrete example of the nature of the problem, we outline data used in this paper. The data is drawn from a competition for classification of brain magnetoencephalography (MEG) data that was organized together with the ICANN 2011 conference [Klami et al. 2011]. The task was to train a classifier for predicting the movie being shown to the test subject. There were five classes and the data consisted of 204 channels. Each measurement was one second in length and the sampling rate was 200 Hz. From each one-second measurement, the competitors had to derive discriminative features for classification. Since there were only a few hundred measurements, the number of features (40800) easily exceed the number of measurements (i.e., the size of the training set) and, thus the key problem is to select the most relevant features efficiently.

Our method based on regularized logistic regression was the most accurate among ten submissions to the competition [Huttunen et al. 2011]. While preparing for the submission, we tested various iterative feature selection methods including the forward stepwise selection and backward stepwise selection [Pudil et al. 1994] and Simulated Annealing Feature Selection (SAFS) [Debuse and Rayward-Smith 1997], but obtained the best results using Logistic Regression with elastic net penalty leading to a joint algorithm for classifier design and feature selection. However, due to lack of time, these experiments were not systematic, but rather intuitive tests aimed at finding a direction where to proceed with the challenge submission.

In our recent paper, we discovered that the ℓ_1-regularization based feature selection outperforms both forward selection and simulated annealing feature selection for interpreting brain signals when formulated as a regression problem [Kauppi et al. 2011]. In this paper we extend the study for classification problems, and compare regularized logistic regression with sequential forward selection (SFS), sequential backward selection (SBS), and SAFS that have been combined with support vector machine (SVM) classifiers.

Before moving on, we briefly explain the central role of careful analysis of the feature selection approaches to neuroimaging studies. In functional neuroimaging studies such as described above, the point of the true importance is not the prediction of the stimulus type based on the imaging as such (we know the true type of the stimulus), but the analysis of the classifier leading to a good prediction of the stimulus type[1]. The idea is that if the classifier makes good predictions about the stimulus type then it is also informative about the distinctions of the neural representations of the two or more behavioral tasks. Especially, when the classifier is linear and sparse (i.e., uses only a small subset of all available features), the localization of the system of brain regions responding differently to the tasks becomes possible. The major benefit of this classification approach termed multivoxel pattern analysis over the more traditional method based on massively univariate statistical hypothesis testing (see, e.g., [Smith et al. 2004]) is that it (in principle) allows the identification of the set of voxels whose activity is diagnostic for engagement of a particular task while the traditional approach allows the identification of the set of voxels that are activated by a task [Poldrack et al. 2009]. The accuracy of inferences made based on the classifiers is bounded by the accuracy of the classifiers used for making the inferences. Since the feature selection is the most central component of the classifier design in these applications, it is important to understand the relative performance of the different feature selection schemes.

The reasons for using feature selection are two-fold: On one hand, using only a subset of features tends to improve the classification performance, and on the other hand, recognizing the significant features may provide insight on the mechanisms of the underlying phenomenon. For example, in our case, the MEG channels related to selected features are of interest.

In the rest of this paper, we first describe the studied feature selection and classification methods in detail in Section 2. In Section 3, we describe the data we are using. Section 4 considers the question of overlearning the training data and how to avoid the pitfalls by elaborate validation of classification performance. In Section 5, we study the efficiency of alternative classification and feature selection methods in a systematic manner, and compare them with regularized logistic regression used in our winning submission. Finally, Section 6 discusses the results and draws some conclusions on the results.

2 Feature Selection Methods

The problem of selecting the best features among all available measurements is computationally very difficult. If there are altogether

[1]There are neuroimaging applications of the supervised classification where the main interest is the prediction in itself such as brain computer interfaces [Blankertz et al. 2010] and diagnosis of neurodegenerative diseases based on imaging data [Wolz et al. 2011]

P input features, the number of subsets is 2^P, which easily becomes prohibitively large for exhaustive search. For example, in the experiments of Section 5, the number of features is $P = 408$, and there are $2^{408} \approx 7 \times 10^{122}$ subsets. If the feasibility of one subset could be tested as fast as a single clock cycle of a 3 GHz processor, exhaustive search would take 7×10^{105} years, i.e., approximately 2×10^{88} times the age of the universe. Alternatively, exploiting the earth's entire computational power and assuming 58 % annual growth [Hilbert and López 2011], the exhaustive search would take approximately 517 years.

Feature selection methods can be divided in two general categories, which are the *filter model* and the *wrapper model*. The wrapper model is the more intuitive of these: wrapper algorithms iteratively test the classifier performance with some or all of the 2^P subsets. The obvious problem is in the computational complexity especially with large data sets. Moreover, the best features for some classification algorithm might not be optimal for another one.

The objective of the filter model is to find general characteristics of the data to evaluate feature subsets without involving any classification algorithm. The statistical fitness of a feature can be evaluated with many different measures, of which popular ones include correlation-based measures and the Fisher ratio, i.e., the ratio of between-class-variance and the within-class-variance. The main drawback of the filter model is that the produced ranking of features does not take into account their joint predictive power, but instead treats them individually. In this respect, the wrapper model is typically better in finding features complementary to each other.

Because of these reasons, our study focuses on different wrapper-like algorithms. There exists vast literature on the topic, and the approaches can be roughly divided into four categories [Liu and Yu 2005].

- **Exhaustive search** is the simplest algorithm, with the obvious drawback of computational cost. However, the search space can be limited with different heuristics, such as the *branch and bound* [Narendra and Fukunaga 1977].
- **Sequential search** iteratively updates the current subset selection by adding or deleting features from the active set. The method can be thought of as an instance of greedy hill-climbing, where the optimality is not guaranteed. Typical heuristics include *forward selection* (start with empty set and add features one by one), *backward selection* (start with all the features and eliminate one at a time) and *forward-backward feature selection*, which alternates between addition and selection steps [Hastie et al. 2009].
- **Randomized search** algorithms introduces randomness into the selection through well-known randomized optimization algorithms such as *simulated annealing* [Lin et al. 2008] and *genetic algorithms* [Siedlecki and Sklansky 1989]. The randomness extends the sequential search by adding possibility of escaping from local optima. However, the difficulty in randomized algorithms is indeed their randomness: ten test runs result in ten different results, which makes drawing any conclusions difficult.
- **Embedded feature selection** is an alternative to the wrapper approach that embeds the feature selection into the performance criterion. The feature selection can be added to the performance criterion by introducing a penalty for the number of nonzero coefficients in the model. The most popular of embedded feature selection methods is the *logistic regression* model, where feature selection can be embedded by sparsity promoting prior in the Bayesian framework [Krishnapuram et al. 2005; Friedman et al. 2010].

In the subsequent sections, we will concentrate on embedded feature selection and sequential search.

2.1 Embedded Feature Selection

A recent approach to feature selection embeds the feature selection into classifier design. One of the most successful methods uses the *logistic regression* model, also known as the *logit model*. In addition to designing a classifier, the cost function includes a sparsity enforcing regularization term and, thus, works as an embedded feature selector that automatically selects the set of relevant features and channels from the pool of candidates.

More specifically, the symmetric multinomial logistic regression models the conditional probability of class $k = 1, 2, \ldots, K$ given the P-dimensional feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_P)^T$ as

$$p_k(\mathbf{x}) = \frac{\exp(\beta_{k0} + \boldsymbol{\beta}_k^T \mathbf{x})}{\sum_{j=1}^{K} \exp(\beta_{j0} + \boldsymbol{\beta}_j^T \mathbf{x})}, \tag{1}$$

where β_{k0} and $\boldsymbol{\beta}_k = (\beta_{k1}, \beta_{k2}, \ldots, \beta_{kP})^T$ are the coefficients of the model [Hastie et al. 2009]. For this model to be valid we have to assume mixture or $\mathbf{x}$-conditional sampling [Anderson and Blair 1982] or – in a more relaxed form – that the class frequencies are (approximately) the same in the training and test data. Despite of the apparent nonlinearity of Equation (1), the resulting classifier is linear and the class k^* of a test sample $\mathbf{x}$ is selected as $k^* = \arg\max_k \left\{\beta_{k0} + \boldsymbol{\beta}_k^T \mathbf{x}\right\}$.

The training of the logistic regression model consists of maximizing the following likelihood function:

$$\sum_{i=1}^{N} \log p_{y_i}(\mathbf{x}_i), \tag{2}$$

where $y_i \in \{1, 2, \ldots, K\}$ denotes the true class of the i^{th} training sample $\mathbf{x}_i$ ($i = 1, 2, \ldots, N$).

The feature selection is introduced by adding a sparsity promoting penalty term into Equation (2). A widely used penalty term is the ℓ_1 cost, which results in the problem of maximizing the equation:

$$\sum_{i=1}^{N} \log p_{y_i}(\mathbf{x}_i) - \lambda \sum_{k=1}^{K} ||\boldsymbol{\beta}_k||_1, \tag{3}$$

where $||\boldsymbol{\beta}_k||_1$ denotes the ℓ_1 norm of the coefficient vector $\boldsymbol{\beta}_k$. The extent of regularization is controlled by the regularization parameter $\lambda \geq 0$.

The traditional regularization approach uses ℓ_2 penalty instead of ℓ_1 penalty. However, the ℓ_2 does not result in sparse solution, and its use has been limited to improved generalization due to shrinkage. Nevertheless, the ℓ_2 penalty possesses many favourable properties that ℓ_1 does not, and in many cases results in better generalization than ℓ_1. Thus, a combination of the two penalty term can result in a sparse classifier with good generalization. Such a combination is known as the *elastic net* penalty and the penalized log-likelihood function to be maximized is defined as [Zou and Hastie 2005]

$$\sum_{i=1}^{N} \log p_{y_i}(\mathbf{x}_i) - \lambda \sum_{k=1}^{K} \left(\alpha ||\boldsymbol{\beta}_k||_1 + (1-\alpha) ||\boldsymbol{\beta}_k||_2^2\right), \tag{4}$$

The regularization term is a combination of the ℓ_1 and ℓ_2 norms of the coefficient vectors $\boldsymbol{\beta}_k$, and the weights for both types of norms are determined by the mixing parameter $\alpha \in [0, 1]$.

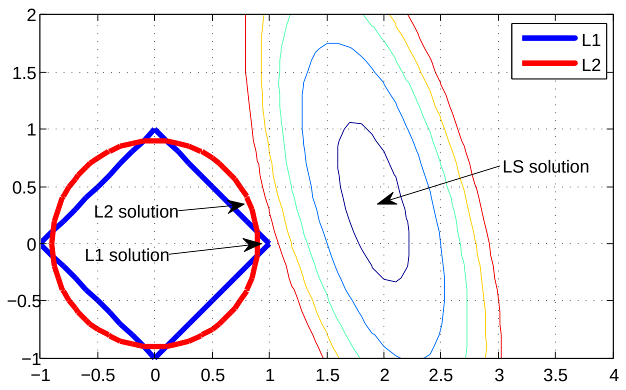

Figure 1: *The isosurfaces of the ℓ_1 and the ℓ_2 penalty. The minimum of the cost function inside the square defining the ℓ_1 penalty is along the horizontal axis thus resulting in a sparse solution.*

The role of parameter α is to determine the type of regularization. When α is zero, the ℓ_1 norm vanishes and results in a special case of *Tikhonov regularization*. The ℓ_2 penalty can be expected to work well in cases, where there are several noisy and mutually correlating features. This is because penalizing the ℓ_2 norm brings the coefficients of the correlating features closer to each other resulting in noise reduction in form of averaging. On the other hand, when α is equal to one, the ℓ_2 norm disappears, which produces the ℓ_1 regularized logistic regression of Equation (3). The ℓ_1 regularization is known for its ability to produce sparse solutions where only a few of the coefficients are non-zero, and this property carries over to the elastic net (except for the case $\alpha = 0$). Thus, both the ℓ_1 regularization and the elastic net can be efficiently used as an implicit feature selectors.

The role of the parameter λ is to control the strength of the regularization effect: the larger the value of λ, the heavier the regularization. For small values of λ, the solution is close to the maximum likelihood solution, while large values of λ allow only restricted solutions and push the coefficients towards zero. In practice, the values of both regularization parameters α and λ are determined by cross-validation, i.e., all combinations over a fixed grid are tested and the CV errors are compared.

The sparsity produced by the ℓ_1 norm is often explained graphically by a figure similar to Figure 1. The figure shows the isosurfaces of the ℓ_1 (square) and the ℓ_2 (round) penalty in terms of parameters β_1 and β_2 in a two-dimensional case when λ has been fixed. The ℓ_1 regularized solution is the point where the quadratic cost represented by the ellipses reaches its minimum inside the square. Due to the cornered shape of the constraint, this tends to appear at one of the corners of the constraint region, and the likelihood increases as the square is scaled down by increasing the value of λ. The same effect is not apparent with the round constraint area of the ℓ_2 penalty.

There are two approaches for estimating the parameters for model (1): Either through maximization of the likelihood function separately for each λ [Yamashita et al. 2008], or simultaneously for the whole regularization path [Friedman et al. 2010]. In the experiments below, we use the latter method due to higher speed.

Algorithm 1 Forward selection.

```
Initialize the parameter subset as $S = \emptyset$.
Initialize the classification error as $\epsilon(S) = \infty$.
**while** not terminated **do**
    **for** For each variable $x_p \notin S, p = 1, 2, \ldots, P$ **do**
        $S' = S \cup \{x_p\}$
        // $M$-fold CV loop:
        **for** $j = 1 \to M$ **do**
            Use the feature set $S'$ and train with all training data
            except the $j^{\text{th}}$ fold.
            Estimate the error $\epsilon_j$ by classifying the $j^{\text{th}}$ fold.
        **end for**
        The classifier error estimate $\epsilon_p$ is the mean of $\epsilon_j$.
    **end for**
    Find $\hat{p} = \arg\min_p \epsilon_p$.
    **if** $\epsilon_{\hat{p}} < \epsilon(S)$ **then**
        Let $S \leftarrow S \cup \{x_{\hat{p}}\}$
    **end if**
    If an improved subset was not found on this iteration, exit the
    while loop.
**end while**
```

2.2 Sequential Feature Selection Methods

2.2.1 Forward Selection and Backward Selection

Among the sequential feature selection methods, the simplest ones are forward selection and backward selection. Their difference is in the starting point of the iteration: forward selection starts with an empty feature set and iteratively adds new features, while backward selection starts with all features and deletes the most harmful features one by one. These selection algorithms are described in more detail in Algorithm 1 and Algorithm 2. The algorithms can also be combined to alternate between addition and deletion steps, and different rules of thumb can be constructed for balancing the probability of addition and deletion (see e.g., [Zhang 2011] for a recent adaptive acceptance rule). In the experiments of Section 5, we experimented also with a few forward-backward algorithms, and it turned out that the feature selection path was always one-directional, i.e., only either forward or backward steps were taken, depending on the starting point (empty feature set or full feature set).

2.2.2 Simulated Annealing Feature Selection

In order to avoid local minima, *Simulated Annealing* (SA) has also been used for feature selection [Lin et al. 2008]. SA is a randomized search heuristic with roots in condensed matter physics, where slowed-down cooling is used to reduce the defects of the material by allowing the molecule configuration to reach its global minimum state. The method has been successfully used in various optimization problems with multiple local extrema. More specifically, SA starts with the empty feature subset, and at each iteration attempts to add or remove a random feature from the set. The change in cross-validated prediction error then used to determine whether the new subset is accepted. All improved results are accepted, while worse solutions are accepted at random with probability

$$\exp\left(\frac{\epsilon(S) - \epsilon(S')}{T}\right), \tag{5}$$

where T is the simulated temperature and $\epsilon(S)$ and $\epsilon(S')$ are the error estimates for the old (better) and the new (worse) solution, respectively. The temperature T is initialized to a high value where almost all configurations are accepted, and it is decreased at each

Algorithm 2 Backward selection.

```
Initialize the parameter subset as $S = \{\text{all available features}\}$.
Initialize the classification error as $\epsilon(S) = \infty$.
while not terminated do
    for For each variable $x_p \in S, p = 1, 2, \ldots, P$ do
        $S' = S \setminus \{x_p\}$
        // $M$-fold CV loop:
        for $j = 1 \to M$ do
            Use the feature set $S'$ and train with all training data except the $j^{\text{th}}$ fold.
            Estimate the error $\epsilon_j$ by classifying the $j^{\text{th}}$ fold.
        end for
        The classifier error estimate $\epsilon_p$ is the mean of $\epsilon_j$.
    end for
    Find $\hat{p} = \arg\min_p \epsilon_p$.
    if $\epsilon_{\hat{p}} < \epsilon(S)$ then
        Let $S \leftarrow S \cup \{x_{\hat{p}}\}$
    end if
    If an improved subset was not found on this iteration, exit the while loop.
end while
```

iteration according to the rule $T \leftarrow \alpha T$ with $\alpha < 1$. The method is described in detail in Algorithm 3.

2.2.3 Wrapped Classifier: The SVM

Sequential feature selection methods always have to be coupled with a classifier, whose performance is iteratively tested with candidate feature sets. In the results below, we choose to use a support vector machine (SVM) for this task. The SVM is is a maximum margin classifier, which (in binary case) maximizes the minimum distance of the training samples from the decision boundary.

The success of the SVM is based on the observation that decision functions can be represented through inner products with the training samples as follows [Schölkopf and Smola 2001]:

$$c(\mathbf{x}) = \operatorname{sgn}\left(\sum_{n=1}^{N} \alpha_n y_n \langle \mathbf{x}, \mathbf{x}_n \rangle + \beta_0\right), \qquad (6)$$

where $c(\mathbf{x}) = \{-1, +1\}$ is the predicted class label for sample $\mathbf{x}$, and $\mathbf{x}_n$ and y_n $(n = 1, 2, \ldots, N)$ are the training samples and classes, respectively. Moreover, α_n, $n = 1, 2, \ldots, N$ and β_0 are the model parameters inferred from the training data. The representation (6) enables the use of the *kernel trick* [Schölkopf and Smola 2001], which implicitly maps the data into a higher dimensional space through a mapping $\phi(\mathbf{x})$. Instead of explicitly mapping the data into higher dimension, it is enough to calculate the inner products $\langle \phi(\mathbf{x}), \phi(\mathbf{x}_n) \rangle$. The kernel trick substitutes this inner product with a kernel function $\kappa(\mathbf{x}, \mathbf{x}_n) = \langle \phi(\mathbf{x}), \phi(\mathbf{x}_n) \rangle$, and it can be shown that all positive definite kernels $\kappa(\cdot, \cdot)$ correspond to a mapping $\phi(\cdot)$. Thus, the kernelized version of (6) is defined as

$$c(\mathbf{x}) = \operatorname{sgn}\left(\sum_{n=1}^{N} \alpha_n y_n \kappa(\mathbf{x}, \mathbf{x}_n) + \beta_0\right), \qquad (7)$$

Despite the apparent linearity of the SVM classifier, the implicit mapping in fact allows nonlinear decision boundaries. This makes the SVM very interesting for our comparison: While preparing for the ICANN MEG challenge submission, we were concerned with how significant is the degradation in performance caused by the linearity of the decision boundaries of the logistic regression classifier.

Algorithm 3 Simulated annealing selection.

```
Initialize the parameter subset as $S = \emptyset$.
Initialize the classification error as $\epsilon(S) = \infty$.
Initialize the temperature $T$.
while not terminated do
    Randomly select $p \in \{1, 2, \ldots, P\}$.
    if $x_p \in S$ then
        $S' = S \setminus \{x_p\}$
    else
        $S' = S \cup \{x_p\}$
    end if
    // $M$-fold CV loop:
    for $j = 1 \to M$ do
        Use the feature set $S'$ and train with all training data except the $j^{\text{th}}$ fold.
        Estimate the error $\epsilon_j$ by classifying the $j^{\text{th}}$ fold.
    end for
    The classifier error estimate $\epsilon(S')$ is the mean of $\epsilon_j$.
    Let $S \leftarrow S'$ with probability $\min\{1, \exp(\epsilon(S) - \epsilon(S')/T)\}$
    $T \leftarrow \alpha T$.
    If an improved subset has not been found for $N$ iterations, exit the while loop.
end while
```

In the results section we will study its efficiency with two kernels: the linear kernel and the popular radial basis function (RBF) kernel.

A multiclass classification problem can be reduced to multiple binary problems. Multiclass SVM is implemented, e.g., in the `LibSVM` package [Chang and Lin 2011], which we also use in our experiments.

3 Material

In the results section we study the efficiency of two feature selection and classification approaches using the data of the MEG Mind Reading challenge of ICANN 2011 conference[2]. The data set consists of MEG signals recorded from a test subject while watching five different video stimuli without audio:

1. **Artificial:** Animated shapes or text
2. **Nature:** Nature documentary clips
3. **Football:** Soccer match clips
4. **Bean:** Part from the comedy series "Mr. Bean"
5. **Chaplin:** Part from a Chaplin movie

The provided measurements consist of 204 gradiometer channels, and the length of each individual epoch is one second and the sampling rate is 200 Hz. Moreover, the five band-pass filtered versions of the signal are also included in the measurement data, with bands centered on the frequencies of 2 Hz, 5 Hz, 10 Hz, 20 Hz, and 35 Hz.[3]

The MEG measurements were recorded on two separate days such that the same set of video stimuli was shown to a test person on both days. Stimuli labeled as either Artificial, Nature, or Football (short clips) were presented as randomly ordered sequences of length 6 –

[2]The data can be downloaded from `http://www.cis.hut.fi/icann2011/meg/measurements.html`

[3]Note, that the challenge report [Klami et al. 2011] erroneously states the frequency features to be *the envelopes* of the frequency bands. However, the data consists of the plain frequency bands; see the erratum at `http://www.cis.hut.fi/icann2011/meg/megicann_erratum.pdf`.

26 s with a 5 s rest period between the clips, while Bean and Chaplin (movies) were presented in two consecutive clips of approximately 10 minutes. In the competition data, the measurements are cut into one-second epochs that are further divided into training and testing such that the training data with known class labels contains 677 epochs of first day data and 50 epochs of second day data while the secret test data contains 653 epochs of second day data only. Notice that the ground truth class labels for the test recordings have been released after the end of the competition.

The data is provided in a randomized order, and the complete signal cannot be reconstructed based on the individual signal epochs. During the competition, the competitors were given the information that the secret test data comes from the second day measurements only and that – similar to the training data – it is approximately class-balanced.

The division between training and test data was elaborate. In particular, 33 % of the test samples consist of recording during stimuli not seen in the training phase in order to test the ability of the classifiers to generalize to new stimuli. A more detailed description of the data can be found in the challenge report by Klami et al. [Klami et al. 2011].

For each one-second epoch of the data in 204 gradiometer channeles, we apply a feature extraction step. For the competition we experimented with numerous features fed to the classifier, and attempted to design discriminative features using various techniques [Huttunen et al. 2011]. Our understanding is that those ended up being too specific for the the first day data and eventually a simplistic solution turned out to be the best, resulting in the following features:

- The detrended mean for each channel, i.e., the parameter $\hat{b}$ of the linear model $y = ax + b$ fitted to the time series.
- The standard deviation of the residual of the fit for each channel, i.e., stddev$(\hat{y} - y)$.

Both features are calculated from the raw data; we were unable to gain any improvement from the filtered channels. Since there are 204 channel, this makes a total of 408 features from which to select in the feature selection step.

4 Performance Assessment

An important aspect for the classifier design is the error assessment. However, the designer would like to spend only a small portion of the annotated training data for testing. The typical approach is to use some kind of cross-validation technique, where a subset of training data is used for assessing the performance of the data. However, there are several pitfalls that the designer should avoid.

There are two kinds of errors that challenge the performance of a classifier: *training error* and *generalization error*. The former refers to the classification error caused by insufficient separation between the classes: Overlapping classes can not be separated no matter how good the classifier is. The training error can be efficiently estimated from the training data, and its significance can thus be easily assessed. The latter kind of error is typically more tricky: Generalization error is the error ultimately caused by training on a finite sample and testing on a finite sample. Because of this, there is always random variation present in both samples, and the classifier will learn to exploit patterns present in the training set but not in the test set. In most cases, the generalization error is the main cause of poor performance, and we believe that our assessment of generalization error was the key to our success in the two recent challenges [Huttunen et al. 2011; Manninen et al. 2011].

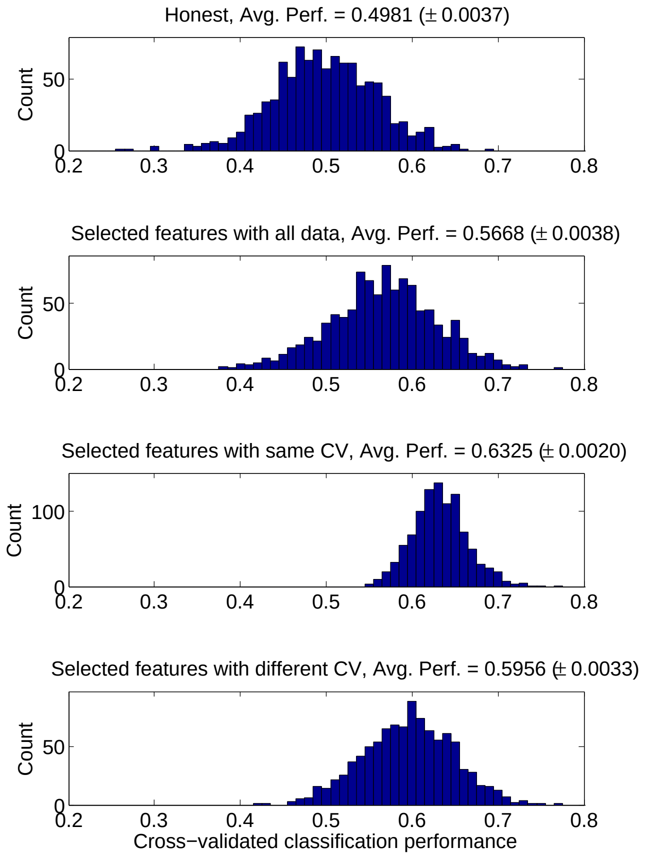

Figure 2: *Results of the wrong and right way to do cross-validation: (a) accuracy over 1000 tests without feature selection; (b) feature selection with all data; (c) feature selection using 10-fold CV twice; (d) feature selection using 10-fold CV twice with different seeds.*

In a data-rich situation the preferred solution is to split the data into three parts: *training set*, *validation set* and *test set* [Hastie et al. 2009, p. 222]. The training set is used for fitting the model, the validation set for estimating the prediction error when developing the model (e.g., when testing which features to include), and the test set for assessing the generalization error. An important point is to keep the last set "in vault" until the very last moment, and should only be introduced into the process as late as possible to avoid overlearning to it. Every time the test set is used for making decisions about the classifier, the estimated generalization error and, thus, the true performance, become more and more optimistic. The following section describes an experiment, where this principle is violated.

4.1 Wrong Ways to Use Cross Validation in Feature Selection

Hastie *et al.* describe *the wrong and right way to do cross-validation* in their book [Hastie et al. 2009], and the wrong way appears frequently in various domains [Ambroise and McLachlan 2002]. The wrong way frequently appears in scientific literature, and uses *all data* for making decisions about the model. In particular, Hastie *et al.* concentrate on feature selection: The incorrect strategy consists of three steps [Hastie et al. 2009, Sec. 7.10.2]:

1. Find a subset of good predictors that exhibit strong correlation with the class labels

2. Using only this subset, design a multivariate classifier.
3. Use cross-validation to tune the model and estimate the prediction error.

The problem of the above approach is that it uses all data in the first step and, thus, produces a too optimistic error estimate and a worse performance for truly independent test data. However, it turns out that the misuse can be a lot more subtle and difficult to recognize as we will see.

Following the example of [Hastie et al. 2009], we did the following experiment.

1. Generate $N = 100$ samples of random data of dimension $P = 30$.
2. Generate binary class labels for the samples also at random.
3. **Honest way:** Design a classifier and calculate the classification error using 10-fold cross validation. In this example we used a 3-nearest-neighbor (3NN) classifier.
4. **Cheating (version 1):** Violate the CV principles as in [Hastie et al. 2009]:
   (a) Find the single feature among $P = 30$ that gives least classifier error with 3NN.
   (b) Estimate the error as in step 3, but with only the best feature.
5. **Cheating (version 2):** Use the 10-fold CV *twice*:
   (a) Estimate classification error for each feature using the 10-fold CV.
   (b) Estimate the error using 10-fold CV as in step 3, but with only the best feature found in step 5a.
6. **Cheating (version 3):** Proceed as in step 5, but change the random seed between the two CV's, thus using different division of samples.

The results of this experiments are summarized in Figure 2. It can be seen that the honest way without any feature selection estimates the accuracy in a realistic manner; close to 0.5. However, all three ways of cheating give an optimistic estimate of the performance clearly above 0.5. One might think that using cross-validation in the feature selection step would be less harmful than using all data because the best performing feature in each CV fold is selected by testing on an independent set of data. However, it turns out that the double-CV approaches are clearly the most dangerous ways of incorrectly assessing the classification performance. There are at least two reasons for this to happen: 1) The illusion about the independence of the CV folds is broken right after the first feature selection CV, when the CV results of the best performing features are combined by averaging. 2) Both the first and the second CV use 10 folds, which allows the first feature selection CV to exploit the information about the exact sample size that is going to be used in the second CV that estimates the final classification performance.

4.2 Error Assessment in Our ICANN MEG Submission

In our submission to the ICANN MEG challenge, we used a tailored version of cross-validation, which emphasizes the performance of the second day test data. More specifically, the training and error estimation procedures consist of two nested cross-validation loops as illustrated in Algorithm 4. The outer loop is used for estimating the performance for the unlabeled test data, while the inner loop is used for selection of classifier parameters α and λ (see section 2.1). The high computational complexity of simultaneous error estimation and parameter selection can be clearly seen from the pseudo code. In order to speed up the development, our method uses parallel validation spread over numerous processors as also described in section 4. A Matlab implementation of our method can be downloaded at `http://www.cs.tut.fi/~hehu/mindreading.html`.

Algorithm 4 Error estimation and parameter selection using nested cross validation.

```
// Outer CV loop:
for $n = 1 \to N$ do
    Divide the training data into training and validation sets as described in section 4
    // Search over all parameter combinations:
    for $\alpha = \alpha_{\text{min}} \to \alpha_{\text{max}}$ do
        for $\lambda = \lambda_{\text{min}} \to \lambda_{\text{max}}$ do
            // Inner $M$-fold CV loop:
            for $j = 1 \to M$ do
                Train with all training data except the $j^{\text{th}}$ fold.
                Estimate the error $e_j$ by classifying the $j^{\text{th}}$ fold.
            end for
            The error estimate $e_{\alpha,\lambda}$ is the mean of $e_j$.
        end for
    end for
    Classify the test data using the classifier with smallest $e_{\alpha,\lambda}$.
    Denote the test error by $e_n$.
end for
The final error estimate is the mean of all $e_n$.
```

A natural cross-validation (CV) error estimation technique would be the leave-one-out error estimator for the second day data. More specifically, we would train with all the first day data and 49 samples of the second day data and test with the remaining second day sample. This way there would be 50 test cases whose mean would be the leave-one-out error estimate. However, we were concerned about the small number of test cases, and decided to consider alternative divisions of the second day data to training and testing.

Instead, we randomly divided the 50 test day samples into two parts of 25 samples. The first set of 25 samples was used for training, and the other for performance assessment. Since the division can be done in $\binom{50}{25} > 10^{14}$ ways, we have more than enough test cases for estimating the error distribution. This approach gives slightly too pessimistic error estimates because only half of the second day data is used for training (as opposed to 98 % with leave-one-out), but has smaller variance due to larger number of test cases. Moreover, the pessimistic bias is not a problem, because we are primarily interested in comparing feature sets during method development rather than actually assessing the prediction error.

The imbalance in the number of samples between the first and second day data is quite significant, because with the above division the training set contains more than 25 times more first day data than second day data. Since we wanted to emphasize the role of the second day data, we increased its relative weight in training error. After experimentation, the second day weight was set to three. In training a logistic regression classifier, the weighting can be implemented in a straightforward manner by multiplying each log-odd in Equation 2 by the corresponding weight value.

The remaining problem in estimating the error distribution is the computational load. One run of training the classifier with CV of the parameters takes typically 10 – 30 minutes. If, for example, we want to test with 100 test set splits, we would be finished after a day or two. For method development and for testing different features this is certainly too slow. However, the error estimation can be easily parallelized; simply by testing each division of the test data on a different processor. For example, in our case we had access to a grid computing environment with approximately 1000

Table 1: *A comparison of the performance of different feature selection and classification methods on the ICANN2011 challenge test data. The results for the SAFS selection are averages over 20 test runs due to the stochastic nature of simulated annealing.*

Feat. sel.	Classifier	# feat.	Perf.	p-value
none	*SVM (Lin.)*	408	66.16 %	0.316
none	*SVM (RBF)*	408	67.99 %	0.765
SFS	*SVM (Lin.)*	15	55.44 %	7.01e-7
SFS	*SVM (RBF)*	18	56.66 %	6.14e-6
SBS	*SVM (Lin.)*	334	68.30 %	0.858
SBS	*SVM (RBF)*	334	67.84 %	0.721
SAFS	*SVM (Lin.)*	193.9	65.79 %	0.253
SAFS	*SVM (RBF)*	191.8	65.48 %	0.207
El. Net.	*Log. Regr.*	219	**68.76 %**	-

processors, and we were able to obtain an accurate error estimate in a matter of minutes instead of hours or days.

5 Experimental Results

In this section, we consider the multiclass classification problem of the ICANN2011 MEG mind reading challenge. The training and testing data and the feature extraction step are described in Section 3. We will compare the elastic net penalized logistic regression model (LR-ELNET) that we used in winning the challenge with different combinations of a feature selection algorithm and the SVM. For feature selection, we will consider Sequential Forward Selection (SFS), Sequential Backward Selection (SBS), and Simulated Annealing Selection (SAFS). For classification, we will consider SVMs with linear and radial basis function (RBF) kernels. We used the `libsvm` implementation of the SVM classifier [Chang and Lin 2011], which uses a one-against-one strategy for multinomial classification. The SVM kernel width was chosen as $\gamma = \frac{1}{P}$ and the penalty parameter for the error term was chosen as $C = 1$. We do acknowledge that a cross-validated selection of these parameters would probably improve the performance, but we believe the results are close to optimal.

The test results are shown in Table 1. The first column shows the method used in feature selection and the second column shows the method used in classification. The third column shows how many features of the total 408 ended up in the selected model. The fourth column shows the percentage of correct classifications for the test data. As there are 5 different classes roughly balanced in the test set, the level of random guess is 20 %.

The method used in our original submission seems to be the best performer also in this comparison. However, there are a few alternatives that are within a small margin. The rightmost column shows the p-value of observing the respective performances under the null hypothesis that the true performance is equal to that of regularized logistic regression given the training and test sets[Dietterich 1998]. Thus, only the cases with SFS selection exhibit a poorer performance with statistical significance. However, with the competition test data the winning method remains the same: logistic regression with elastic net regularization.

In particular, both variants of the SVM seem to be successful with backward selection. The forward selection is not as successful, because it gets trapped in a local minimum and includes far too few features in the design. During the experiments, it turned out that alternating between forward and backward steps does not help with local minima: the search path is in practice always one-directional, and the optimizer takes only either addition or elimination steps.

It seems, that increasing the number of features tends to improve the performance until a saturation point somewhere near approximately 300 features. This conclusion is also supported by the fact that among the 20 test runs of the SAFS, there is a high correlation between the number of features and the prediction accuracy. An interesting coincidence is that when using SBS, both SVM variants end up with the same amount of features. The feature sets are not the same, however.

The results also reveal, that a linear classifier has enough discriminative power for the particular application: The two SVM variants are equal in performance. This is probably due to the high dimensionality of the problem, which helps in finding well separating hyperplanes even without the kernel trick.

It is slightly surprising that the simulated annealing does not perform very well. This may be because the optimization starts with an empty feature set, similarly to the forward selection. Thus, the distance to the optimum region with approx. 300 features is very long, and the process cools down too early. One solution to improve the performance could be to start with the full set of features instead of the empty set. However, this would increase the computation time of the already slow annealing even further, because the SVM design time depends on the number of features.

6 Conclusions

In this paper, we have compared the accuracy of the elastic net-regularized logistic regression classifier with the support vector machine classifier combined with various wrapper-based feature selection methods. The data for the comparison was drawn from a recent MEG mind reading competition, where regularized logistic regression outperformed other competitors. The dataset in question has nearly equal number of training samples and features when the number of features is first reduced using an ad-hoc technique described in Section 3. While the logistic regression algorithm was the most accurate in our experiments, the wrapper-SVMs combined with SBS and SAFS and SVM without any feature selection gave nearly as good results as the regularized logistic regression. In other words, the differences between the accuracies of these algorithms were non-significant. Instead, sequential forward feature selection selected always too few features and was significantly less accurate than the regularized logistic regression. We experimented with two different SVM kernels, but found virtually no difference in accuracies of classifiers produced with them. An important aspect in using wrapper methods for feature selection is the performance evaluation, usually using a cross-validation technique. As we have outlined in detail in Section 4, it deceptively easy to misuse the cross-validation and eventually produce too optimistic estimates of the actual error rates.

7 Acknowledgements

Supported by the Academy of Finland grant no. 130275.

References

AMBROISE, C., AND MCLACHLAN, G. J. 2002. Selection bias in gene extraction on the basis of microarray gene-expression data. *Proceedings of the National Academy of Sciences 99*, 10, 6562–6566.

ANDERSON, J., AND BLAIR, V. 1982. Penalized maximum likelihood estimation in logistic regression and discrimination. *Biometrika 69*, 123–136.

BLANKERTZ, B., TANGERMANN, M., VIDAURRE, C., FAZLI, S., SANNELLI, C., HAUFE, S., MAEDER, C., RAMSEY, L., STURM, I., CURIO, G., AND MLLER, K.-R. 2010. The Berlin Brain-Computer Interface: Non-Medical Uses of BCI Technology. *Front Neurosci 4*, 198.

CHANG, C.-C., AND LIN, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2*, 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

DEBUSE, J. C., AND RAYWARD-SMITH, V. J. 1997. Feature subset selection within a simulated annealing data mining algorithm. *Journal of Intelligent Information Systems 9*, 57–81.

DIETTERICH, T. G. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput. 10*, 7 (oct), 1895–1923.

FRIEDMAN, J. H., HASTIE, T., AND TIBSHIRANI, R. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software 33*, 1, 1–22.

HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. 2009. *The elements of statistical learning: Data mining, inference, and prediction*, Second ed. Springer Series in Statistics. Springer.

HILBERT, M., AND LÓPEZ, P. 2011. The World's Technological Capacity to Store, Communicate, and Compute Information. *Science 332*, 6025 (Apr), 60–65.

HUTTUNEN, H., KAUPPI, J.-P., AND TOHKA, J. 2011. Regularized logistic regression for mind reading with parallel validation. In *ICANN2011 MEG challenge*.

HUTTUNEN, H., MANNINEN, T., KAUPPI, J.-P., AND TOHKA, J. 2012. Mind reading with regularized multinomial logistic regression. *Machine Vision and Applications* (Jan.). Submitted.

KAUPPI, J.-P., HUTTUNEN, H., KORKALA, H., JÄÄSKELÄINEN, I. P., SAMS, M., AND TOHKA, J. 2011. Face prediction from fmri data during movie stimulus: Strategies for feature selection. In *ICANN (2)*, Springer, T. Honkela, W. Duch, M. A. Girolami, and S. Kaski, Eds., vol. 6792 of *Lecture Notes in Computer Science*, 189–196.

KLAMI, A., RAMKUMAR, P., VIRTANEN, S., PARKKONEN, L., HARI, R., AND KASKI, S., 2011. ICANN/PASCAL2 Challenge: MEG Mind-Reading — Overview and Results.

KRISHNAPURAM, B., CARIN, L., FIGUEIREDO, M., AND HARTEMINK, A. 2005. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 27*, 6 (june), 957 –968.

LIN, S.-W., LEE, Z.-J., CHEN, S.-C., AND TSENG, T.-Y. 2008. Parameter determination of support vector machine and feature selection using simulated annealing approach. *Appl. Soft Comput. 8*, 1505–1512.

LIU, H., AND YU, L. 2005. Toward integrating feature selection algorithms for classification and clustering. *Knowledge and Data Engineering, IEEE Transactions on 17*, 4 (april), 491 – 502.

MANNINEN, T., HUTTUNEN, H., RUUSUVUORI, P., AND NYKTER, M. 2011. Logistic regression for AML prediction. In *Dialogue for Reverse Engineering Assessments and Methods, DREAM6*.

NARENDRA, P., AND FUKUNAGA, K. 1977. A branch and bound algorithm for feature subset selection. *Computers, IEEE Transactions on C-26*, 9 (sept.), 917 –922.

PEREIRA, F., MITCHELL, T., AND BOTVINICK, M. 2009. Machine learning classifiers and fmri: a tutorial overview. *NeuroImage 45*, Suppl 1, S199 – S209.

POLDRACK, R., HALCHENKO, Y., AND HANSON, S. 2009. Decoding the large-scale structure of brain function by classifying mental states across individuals. *Psychological Science 20*, 1364 – 1372.

PUDIL, P., NOVOVIČOVÁ, J., AND KITTLER, J. 1994. Floating search methods in feature selection. *Pattern Recogn. Lett. 15*, 11 (nov), 1119–1125.

SAEYS, Y., INZA, I., AND LARRAAGA, P. 2007. A review of feature selection techniques in bioinformatics. *Bioinformatics 23*, 2507 – 17.

SCHÖLKOPF, B., AND SMOLA, A. J. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, 1st ed. The MIT Press.

SIEDLECKI, W., AND SKLANSKY, J. 1989. A note on genetic algorithms for large-scale feature selection. *Pattern Recogn. Lett. 10*, 5 (nov), 335–347.

SMITH, S., JENKINSON, M., WOOLRICH, M., BECKMANN, C., BEHRENS, T., JOHANSEN-BERG, H., BANNISTER, P., LUCA, M. D., DROBNJAK, I., FLITNEY, D., NIAZY, R., SAUNDERS, J., VICKERS, J., ZHANG, Y., STEFANO, N. D., BRADY, J., AND MATTHEWS, P. 2004. Advances in functional and structural MR image analysis and implementation as FSL. *Neuroimage 23*, S1, 208 – 219.

WOLZ, R., JULKUNEN, V., KOIKKALAINEN, J., NISKANEN, E., ZHANG, D. P., RUECKERT, D., SOININEN, H., LOTJONEN, J., AND THE ALZHEIMER'S DISEASE NEUROIMAGING INITIATIVE. 2011. Multi-method analysis of mri images in early diagnostics of alzheimer's disease. *PLoS ONE 6*, 10 (10), e25446.

YAMASHITA, O., SATO, M., YOSHIOKA, T., TONG, F., AND KAMITANI, Y. 2008. Sparse estimation automatically selects voxels relevant for the decoding of fmri activity patterns. *NeuroImage 42*, 4, 1414 – 1429.

ZHANG, T. 2011. Adaptive forward-backward greedy algorithm for learning sparse representations. *Information Theory, IEEE Transactions on 57*, 7 (July), 4689 –4708.

ZOU, H., AND HASTIE, T. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67*, 2, 301–320.