Journal of Machine Learning Research 1 (2007) xxx-xxx Submitted xx/07; Published xx/07

# Feature Selection via Dependence Maximization

**Le Song** LESONG@IT.USYD.EDU.AU
*NICTA, Statistical Machine Learning Program, Canberra, ACT 0200, Australia*
*School of Information Technologies, University of Sydney, NSW 2006, Australia*

**Alex Smola** ALEX.SMOLA@NICTA.COM
*NICTA, Statistical Machine Learning Program, Canberra, ACT 0200, Australia*
*The Australian National University, Canberra, ACT 0200, Australia*

**Arthur Gretton** ARTHUR.GRETTON@TUEBINGEN.MPG.DE
*MPI for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany*

**Justin Bedo** JUSTIN.BEDO@NICTA.COM.AU
*NICTA, Statistical Machine Learning Program, Canberra, ACT 0200, Australia*
*The Australian National University, Canberra, ACT 0200, Australia*

**Karsten Borgwardt** BORGWARDT@DBS.IFI.LMU.DE
*LMU, Institute for Database Systems, Oettingenstr. 67, 80538 München, Germany*

**Editor:** F. Uzzy

## Abstract

We introduce a framework of feature selection based on dependence maximization between the selected features and the labels of an estimation problem, using the Hilbert-Schmidt Independence Criterion. The key idea is that good features should be highly dependent on the labels. Our approach leads to a greedy procedure for feature selection. We show that a number of existing feature selectors are special cases of this framework. Experiments on both artificial and real-world data show that our feature selector works well in practice.

## 1. Introduction

In data analysis we are typically given a set of observations $X = \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ which can be used for a number of tasks, such as novelty detection, low-dimensional representation, or a range of supervised learning problems. In the latter case we also have a set of labels $Y = \{y_1, \ldots, y_m\} \subseteq \mathcal{Y}$ at our disposition. Tasks include ranking, classification, regression, or sequence annotation. While not always true in practice, we assume in the following, that the data $X$ and $Y$ have been generated identically independently distributed (iid) from some underlying distribution $\Pr(x, y)$.

We often want to reduce the dimension of the data (the number of features) before the actual learning (Guyon and Elisseeff, 2003); a larger number of features can be associated with higher data collection cost, more difficulty in model interpretation, higher computational cost for the classifier, and *sometimes* decreased generalization ability. In other words, often there exist ulterior motives than finding a well performing estimator which make feature selection a necessity for reasons of speed or deployment cost. It is therefore important to select an informative feature subset.

The problem of supervised feature selection can be cast as a combinatorial optimization problem. We have a full set of features, denoted by $\mathcal{S}$ (each element in $\mathcal{S}$ corresponds to one dimension of the data). It is our aim to select a subset $\mathcal{T} \subseteq \mathcal{S}$ such that this subset retains the relevant information contained in $X$. Suppose the relevance of a feature subset (to the outcome) is quantified by $\mathcal{Q}(\mathcal{T})$, and is computed by restricting the data to the dimensions in $\mathcal{T}$. Feature selection can then be formulated as

$$\mathcal{T}_0 = \arg\max_{\mathcal{T} \subseteq \mathcal{S}} \ \mathcal{Q}(\mathcal{T}) \text{ subject to } |\,\mathcal{T}\,| \leq t, \tag{1}$$

where $|\cdot|$ computes the cardinality of a set and $t$ is an upper bound on the number of selected features. Two important aspects of problem (1) are the choice of the criterion $\mathcal{Q}(\mathcal{T})$ and the selection algorithm.

### 1.1 Criteria for Feature Selection

A number of quality functionals $\mathcal{Q}(\mathcal{T})$ are potential candidates to use for feature selection. For instance we could use a mutual-information related quantity or a Hilbert Space based estimator. In any case, the choice of $\mathcal{Q}(\mathcal{T})$ should respect the underlying tasks: supervised learning estimate functional dependence $f$ from training data and guarantee $f$ predicts well on test data. Therefore, good criteria should satisfy two conditions:

I: $\mathcal{Q}(\mathcal{T})$ is capable of detecting desired (linear or nonlinear) functional dependence between the data and the labels.

II: $\mathcal{Q}(\mathcal{T})$ is concentrated with respect to the underlying measure. This guarantees with high probability that detected functional dependence is preserved in test data.

While many criteria have been explored few take these two conditions explicitly into account. Examples include the leave-one-out error bound of SVM (Weston et al., 2000) and the mutual information (Zaffalon and Hutter, 2002). Although the latter has good theoretical justification, it requires density estimation, which is problematic for high dimensional and continuous variables. We sidestep these problems by employing a mutual-information *like* quantity — the Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005). HSIC uses kernels for measuring dependence and does not require density estimation. HSIC also has good uniform convergence guarantees. As we show in section 2, HSIC satisfies conditions I and II, required for $\mathcal{Q}(\mathcal{T})$.

### 1.2 Feature Selection Algorithms

Finding a global optimum for (1) is typically NP-hard (Weston et al., 2003), unless the criterion is easily decomposable or has properties which make optimization easier, e.g. submodularity Guestrin et al. (2005). Many algorithms transform (1) into a continuous problem by introducing weights on the dimensions (Weston et al., 2000; Bradley and Mangasarian, 1998; Weston et al., 2003; Neal, 1998). These methods perform well for linearly separable problems. For nonlinear problems, however, the optimisation usually becomes non-convex and a local optimum does not necessarily provide good features. Greedy approaches, forward selection and backward elimination, are often used to tackle problem (1) directly.

Forward selection tries to increase $\mathcal{Q}(\mathcal{T})$ as much as possible for each inclusion of features, and backward elimination tries to achieve this for each deletion of features (Guyon et al., 2002). Although forward selection is computationally more efficient, backward elimination provides better features in general since the features are assessed within the context of all others. See Section 7 for experimental details.

In principle, the Hilbert-Schmidt independence criterion can be employed for feature selection using either a weighting scheme, forward selection or a backward selection strategy, or even a mix of several strategies. While the main focus of this paper is on the backward elimination strategy, we also discuss the other selection strategies. As we shall see, several specific choices of a kernel function will lead to well known feature selection and feature rating methods.

Note that backward elimination using HSIC (BAHSIC) is a filter method for feature selection. It selects features independent of a particular classifier. Such decoupling not only facilitates subsequent feature interpretation but also speeds up the computation over wrapper and embedded methods.

We will see that BAHSIC is directly applicable to binary, multiclass, and regression problems. Most other feature selection methods are only formulated either for binary classification or regression. Multi-class extension of these methods is usually accomplished using a one-versus-the-rest strategy. Still fewer methods handle classification and regression cases at the same time. BAHSIC, on the other hand, accommodates all these cases *and* unsupervised feature selection in a principled way: by choosing different kernels, BAHSIC not only subsumes many existing methods as special cases, but also allows us to define new feature selectors. Such versatility of BAHSIC originates from the generality of HSIC. Therefore, we first introduce HSIC in the next section.

## 2. Measures of Dependence

We begin with the simple example of linear dependence detection, and then generalize to the detection of more general kinds of dependence. Consider spaces $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}^l$, on which we jointly sample observations $(x, y)$ from a distribution $\Pr(x, y)$. Denote by $\mathcal{C}_{xy}$ the covariance matrix

$$\mathcal{C}_{xy} = \mathbb{E}_{xy}\left[xy^\top\right] - \mathbb{E}_x\left[x\right]\mathbb{E}_y\left[y^\top\right], \tag{2}$$

which contains all second order dependence between the random variables. A statistic that efficiently summarizes the degree of *linear correlation* between $x$ and $y$ is the Frobenius norm of $\mathcal{C}_{xy}$. Given the singular values $\sigma_i$ of $\mathcal{C}_{xy}$ we can compute it via

$$\|\,\mathcal{C}_{xy}\,\|^2_{\text{Frob}} := \sum_i \sigma_i^2 = \operatorname{tr}\mathcal{C}_{xy}\,\mathcal{C}_{xy}^\top.$$

This quantity is zero if and only if there exists no *linear dependence* between $x$ and $y$. This statistic is limited in several respects, however, of which we mention two: first, dependence can exist in forms other than that detectable via covariance (and even when a second order relation exists, the full extent of the dependence between $x$ and $y$ may only be apparent when nonlinear effects are included). Second, the restriction to subsets of $\mathbb{R}^d$ excludes many interesting kinds of variables, such as strings and class labels. In the next section, therefore,

we generalize the notion of covariance to nonlinear relationships, and to a wider range of data types.

### 2.1 Hilbert-Schmidt Independence Criterion (HSIC)

In general $\mathcal{X}$ and $\mathcal{Y}$ will be two domains from which we draw samples $(x, y)$: these may be real valued, vector valued, class labels, strings (Lodhi et al., 2002), graphs (Gärtner et al., 2003), dynamical systems (Vishwanathan et al., 2007), parse trees (Collins and Duffy, 2001), images (Schölkopf, 1997), or similar. See (Schölkopf et al., 2004; Schölkopf and Smola, 2002) for further references.

We define a (possibly nonlinear) mapping $\phi : \mathcal{X} \to \mathcal{F}$ from each $x \in \mathcal{X}$ to a feature space $\mathcal{F}$ (and an analogous map $\psi : \mathcal{Y} \to \mathcal{G}$ wherever needed). In this case we may write the inner product between the features via the kernel functions

$$k(x, x') := \left\langle \phi(x), \phi(x') \right\rangle \text{ and } l(y, y') := \left\langle \psi(y), \psi(y') \right\rangle. \tag{3}$$

We usually associate with $k$ and $l$ Reproducing Kernel Hilbert Spaces (RKHS). With some abuse of notation[1] we also refer to them, too, $\mathcal{F}$ and $\mathcal{G}$. For instance, if $\mathcal{X} = \mathbb{R}^d$, then this could be as simple as a set of polynomials of order up to $b$ in the components of $x$, with kernel $k(\mathbf{x}, \mathbf{x}') = (\langle \mathbf{x}, \mathbf{x}' \rangle + a)^b$. Other kernels, like the Gaussian RBF kernel correspond to infinitely large feature spaces. We need never evaluate these feature representations explicitly, however.

We may now define a cross-covariance operator[2] between these feature maps, in accordance with Baker (1973); Fukumizu et al. (2004): this is a linear operator $\mathcal{C}_{xy} : \mathcal{G} \longmapsto \mathcal{F}$ such that

$$\mathcal{C}_{xy} := \mathbb{E}_{xy}\left[(\phi(x) - \mu_x) \otimes (\psi(y) - \mu_y)\right] \text{ where } \mu_x = \mathbb{E}_x[\phi(x)] \text{ and } \mu_y = \mathbb{E}_y[\psi(y)]. \tag{4}$$

Here $\otimes$ denotes the tensor product. We need to extend the notion of a Frobenius norm to operators. This leads us to the Hilbert-Schmidt norm, which is given by the trace of $\mathcal{C}_{xy}\mathcal{C}_{xy}^\top$. For operators with discrete spectrum this amounts to computing the $\ell_2$ norm of the singular values. We use the square of the Hilbert-Schmidt norm of the cross-covariance operator (HSIC), $\|\mathcal{C}_{xy}\|_{\mathrm{HS}}^2$ as our feature selection criterion $\mathcal{Q}(\mathcal{T})$. Gretton et al. (2005) show that HSIC can be expressed in terms of kernels as

$$\begin{aligned} \mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy}) &:= \|\mathcal{C}_{xy}\|_{\mathrm{HS}}^2 \\ &= \mathbb{E}_{xx'yy'}[k(x, x')l(y, y')] + \mathbb{E}_{xx'}[k(x, x')]\,\mathbb{E}_{yy'}[l(y, y')] - 2\,\mathbb{E}_{xy}[\mathbb{E}_{x'}[k(x, x')]\,\mathbb{E}_{y'}[l(y, y')]]. \end{aligned} \tag{5}$$

This allows us to compute a measure of dependence between $x$ and $y$ simply by taking expectations over a set of kernel functions $k$ and $l$ with respect to the joint and marginal distributions in $x$ and $y$ *without* the need to perform density estimation, as is the case with entropy based methods.

1. Strictly speaking there exist many feature maps which correspond to the Hilbert Spaces defined via $k$ and $l$.
2. Again we abuse the notation here by using the same subscript in the operator $\mathcal{C}_{xy}$ as in the covariance matrix of (2), even though we now refer to the covariance between feature maps.

### 2.2 Estimating the Hilbert-Schmidt Independence Criterion

We denote by $Z = (X, Y)$ the set of observations $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ drawn *i.i.d.*from $\Pr_{xy}$. Denote by $\mathbb{E}_Z$ the expectation with respect $Z$ as drawn from $\Pr_{xy}$. Moreover, denote by $\mathbf{K}, \mathbf{L} \in \mathbb{R}^{m \times m}$ kernel matrices obtained via $\mathbf{K}_{ij} = k(x_i, x_j)$ and $\mathbf{L}_{ij} = l(y_i, y_j)$. Finally let $\mathbf{H} = \mathbf{I} - m^{-1}\,\mathbf{1}\,\mathbf{1} \in \mathbb{R}^{m \times m}$ be a centering matrix which projects onto the space orthogonal to the vector $\mathbf{1}$.

Gretton et al. (2005) derive estimators of $\mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy})$ which have $O(m^{-1})$ *bias* and they show that this estimator is well concentrated by means of appropriate tail bounds. For completeness we briefly restate this estimator and its properties below.

**Theorem 1 (Biased estimator of HSIC (Gretton et al., 2005))** *The estimator*

$$\mathrm{HSIC}_0(\mathcal{F}, \mathcal{G}, Z) := (m-1)^{-2} \operatorname{tr} \mathbf{K}\,\mathbf{H}\,\mathbf{L}\,\mathbf{H} \tag{6}$$

*has bias $O(m^{-1})$, i.e.* $\mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy}) - \mathbb{E}_Z\left[\mathrm{HSIC}_0(\mathcal{F}, \mathcal{G}, Z)\right] = \mathcal{O}(m^{-1})$.

This bias arises from the self-interaction terms which are present in $\mathrm{HSIC}_0$, i.e. we still have $O(m)$ terms of the form $\mathbf{K}_{ij}\,\mathbf{L}_{il}$ and $\mathbf{K}_{ji}\,\mathbf{L}_{li}$ present in the sum, which leads to the $O(m^{-1})$ bias. To address this, we now devise an unbiased estimator which removes those additional terms while ensuring proper normalization. Our proposed estimator has the form

$$\mathrm{HSIC}_1(\mathcal{F}, \mathcal{G}, Z) := \frac{1}{m(m-3)}\left[\operatorname{tr}(\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}) + \frac{\mathbf{1}^\top \tilde{\mathbf{K}}\,\mathbf{1}\,\mathbf{1}^\top \tilde{\mathbf{L}}\,\mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2}\,\mathbf{1}^\top \tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1}\right], \tag{7}$$

where $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$ are related to $\mathbf{K}$ and $\mathbf{L}$ by $\tilde{\mathbf{K}}_{ij} = (1 - \delta_{ij})\,\mathbf{K}_{ij}$ and $\tilde{\mathbf{L}}_{ij} = (1 - \delta_{ij})\,\mathbf{L}_{ij}$ (i.e. the diagonal entries of $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$ are set to zero).

**Theorem 2 (Unbiased estimator of HSIC)** *The estimator* $\mathrm{HSIC}_1$ *is unbiased, that is, we have* $\mathbb{E}_Z\left[\mathrm{HSIC}_1(\mathcal{F}, \mathcal{G}, Z)\right] = \mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy})$.

**Proof** We prove the claim by constructing unbiased estimators for each term in (5). Note that we have three types of expectations, namely $\mathbb{E}_{xy}\,\mathbb{E}_{x'y'}$, a partially decoupled expectation $\mathbb{E}_{xy}\,\mathbb{E}_{x'}\,\mathbb{E}_{y'}$, and $\mathbb{E}_x\,\mathbb{E}_y\,\mathbb{E}_{x'}\,\mathbb{E}_{y'}$, which takes all four expectations independently.

If we want to replace the expectations by empirical averages, we need to take care to avoid using the same discrete indices more than once for independent random variables. In other words, when taking expectations over $n$ independent random variables, we need $n$-tuples of indices where each index occurs exactly once. We define the sets $\mathbf{i}_n^m$ to be the collections of indices satisfying this property. By simple combinatorics one can see that their cardinalities are given by the Pochhammer symbols $(m)_n = \frac{m!}{(m-n)!}$. Jointly drawn random variables, on the other hand, share the same index.

For the joint expectation over pairs we have

$$\mathbb{E}_{xy}\,\mathbb{E}_{x'y'}\left[k(x, x')l(y, y')\right] = (m)_2^{-1}\,\mathbb{E}_Z\Big[\sum_{(i,j)\in \mathbf{i}_2^m} \mathbf{K}_{ij}\,\mathbf{L}_{ij}\Big] = (m)_2^{-1}\,\mathbb{E}_Z\left[\operatorname{tr}\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\right]. \tag{8}$$

Recall that we set $\tilde{\mathbf{K}}_{ii} = \tilde{\mathbf{L}}_{ii} = 0$. In the case of the expectation over three independent terms $\mathbb{E}_{xy}\,\mathbb{E}_{x'}\,\mathbb{E}_{y'}\left[k(x,x')l(y,y')\right]$ we obtain

$$(m)_3^{-1}\,\mathbb{E}_Z\Big[\sum_{(i,j,q)\in \mathbf{i}_3^m} \mathbf{K}_{ij}\,\mathbf{L}_{iq}\Big] = (m)_3^{-1}\,\mathbb{E}_Z\left[\mathbf{1}^\top\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1} - \operatorname{tr}\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\right]. \tag{9}$$

For four independent random variables $\mathbb{E}_x\,\mathbb{E}_y\,\mathbb{E}_{x'}\,\mathbb{E}_{y'}\left[k(x,x')l(y,y')\right]$,

$$(m)_4^{-1}\,\mathbb{E}_Z\Big[\sum_{(i,j,q,r)\in \mathbf{i}_4^m} \mathbf{K}_{ij}\,\mathbf{L}_{qr}\Big] = (m)_4^{-1}\,\mathbb{E}_Z\left[\mathbf{1}^\top\tilde{\mathbf{K}}\,\mathbf{1}\,\mathbf{1}^\top\tilde{\mathbf{L}}\,\mathbf{1} - 4\,\mathbf{1}^\top\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1} + 2\operatorname{tr}\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\right]. \tag{10}$$

To obtain an expression for HSIC we only need to take linear combinations using (5). Collecting terms related to $\operatorname{tr}\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}$, $\mathbf{1}^\top\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1}$, and $\mathbf{1}^\top\tilde{\mathbf{K}}\,\mathbf{1}\,\mathbf{1}^\top\tilde{\mathbf{L}}\,\mathbf{1}$ yields

$$\operatorname{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) = \frac{1}{m(m-3)}\,\mathbb{E}_Z\left[\operatorname{tr}\tilde{\mathbf{K}}\,\tilde{\mathbf{L}} + \frac{\mathbf{1}^\top\tilde{\mathbf{K}}\,\mathbf{1}\,\mathbf{1}^\top\tilde{\mathbf{L}}\,\mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2}\,\mathbf{1}^\top\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1}\right]. \tag{11}$$

This is the expected value of $\operatorname{HSIC}_1[\mathcal{F},\mathcal{G},Z]$. ■

Note that neither $\operatorname{HSIC}_0$ nor $\operatorname{HSIC}_1$ require any explicit regularization parameters, unlike earlier work on kernel dependence estimation. This is encapsulated in the choice of the kernels.

For suitable kernels $\operatorname{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) = 0$ if and only if $x$ and $y$ are independent. Hence the empirical estimate $\operatorname{HSIC}_1$ can be used to design nonparametric tests of independence. A key feature is that $\operatorname{HSIC}_1$ itself is *unbiased* and its computation is simple. Compare this to quantities based on the mutual information, which requires sophisticated bias correction strategies (e.g. Nemenman et al., 2002).

Previous work used HSIC to *measure* independence between two sets of random variables (Feuerverger, 1993; Gretton et al., 2005). Here we use it to *select* a subset $\mathcal{T}$ from the first full set of random variables $\mathcal{S}$. We next describe properties of HSIC which support its use as a feature selection criterion.

### 2.3 HSIC detects arbitrary dependence (Property I)

We make use of Theorem 4 of Gretton et al. (2005). It states that whenever $\mathcal{F},\mathcal{G}$ are RKHSs with universal kernels $k,l$ on respective compact domains $\mathcal{X}$ and $\mathcal{Y}$ in the sense of Steinwart (2002), then $\operatorname{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) = 0$ if and only if $x$ and $y$ are independent.

In terms of feature selection, a universal kernel such as the Gaussian RBF kernel or the Laplace kernel permits HSIC to detect any dependence between $\mathcal{X}$ and $\mathcal{Y}$. HSIC is zero only if features and labels are independent. Clearly we want to reach the opposite result, namely strong dependence between features and labels. Hence we try to select features that maximize HSIC. Likewise, whenever we want to select a subset of features from $X$ we will try to retain maximal dependence between $X$ and its reduced version.

Note that non-universal kernels can also be used for HSIC, although they may not guarantee that all dependence is detected. Different kernels incorporate distinctive prior knowledge into the dependence estimation, and they focus HSIC on dependence of a certain type. For instance, a linear kernel requires HSIC to seek only second order dependence,

whereas a polynomial kernel of degree $b$ restricts HSIC to test for dependences of degree (up to) $b$. Clearly HSIC is capable of finding and exploiting dependence of a much more general nature by kernels on graphs, strings, or other discrete domains.

### 2.4 HSIC is concentrated (Property II)

$\mathrm{HSIC}_1$, the estimator in (7), can be alternatively formulated using U-statistics (Hoeffding, 1948). This reformulation allows us to derive a uniform convergence bound for $\mathrm{HSIC}_1$. In term of feature selection, this means that the quality of features evaluated using $\mathrm{HSIC}_1$ closely reflects its population counterpart HSIC. Furthermore, the same features should consistently be selected to achieve high dependence if data are repeatedly drawn from the same distribution. We formalize these results below.

**Theorem 3 (U-statisic of HSIC)** $\mathrm{HSIC}_1$ *can be rewritten in terms of a U-statistic*

$$\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z) = (m)_4^{-1} \sum_{(i,j,q,r)\in \mathbf{i}_4^m} h(i,j,q,r), \tag{12}$$

*where the kernel $h$ of the U-statistic is defined by*

$$h(i,j,q,r) = \frac{1}{24} \sum_{(s,t,u,v)}^{(i,j,q,r)} \mathbf{K}_{st}[\mathbf{L}_{st} + \mathbf{L}_{uv} - 2\,\mathbf{L}_{su}] \tag{13}$$

$$= \frac{1}{6} \sum_{(s\prec t),(u\prec v)}^{(i,j,q,r)} \mathbf{K}_{st}[\mathbf{L}_{st} + \mathbf{L}_{uv}] - \frac{1}{12} \sum_{(s,t,u)}^{(i,j,q,r)} \mathbf{K}_{st}\,\mathbf{L}_{su} \tag{14}$$

Here the first sum represents all $4! = 24$ quadruples $(s,t,u,v)$ which can be selected without replacement from $(i,j,q,r)$. Likewise the sum over $(s,t,u)$ is the sum over all triples chosen without replacement. Finally, the sum over $(s \prec t),(u \prec v)$ has the additional condition that the order imposed by $(i,j,q,r)$ is preserved. That is $(i,q)$ and $(j,r)$ are valid pairs, whereas $(q,i)$ or $(r,q)$ are not.

**Proof** Combining the three unbiased estimators in (8-10) we obtain a single U-statistic

$$\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z) = (m)_4^{-1} \sum_{(i,j,q,r)\in \mathbf{i}_4^m} \left(\mathbf{K}_{ij}\,\mathbf{L}_{ij} + \mathbf{K}_{ij}\,\mathbf{L}_{qr} - 2\,\mathbf{K}_{ij}\,\mathbf{L}_{iq}\right). \tag{15}$$

In this form, however, the kernel $h(i,j,q,r) = \mathbf{K}_{ij}\,\mathbf{L}_{ij} + \mathbf{K}_{ij}\,\mathbf{L}_{qr} - 2\,\mathbf{K}_{ij}\,\mathbf{L}_{iq}$ is not symmetric in its arguments. For instance $h(i,j,q,r) \neq h(q,j,r,i)$. The same holds for other permutations of the indices. Thus, we replace the kernel with a symmetrized version, which yields

$$h(i,j,q,r) := \frac{1}{4!} \sum_{(s,t,u,v)}^{(i,j,q,r)} \left(\mathbf{K}_{st}\,\mathbf{L}_{st} + \mathbf{K}_{st}\,\mathbf{L}_{uv} - 2\,\mathbf{K}_{st}\,\mathbf{L}_{su}\right) \tag{16}$$

where the sum in (16) represents all ordered quadruples $(s,t,u,v)$ selected without replacement from $(i,j,q,r)$.

This kernel can be simplified, since $\mathbf{K}_{st} = \mathbf{K}_{ts}$ and $\mathbf{L}_{st} = \mathbf{L}_{ts}$. The first one only contains terms $\mathbf{L}_{st}\,\mathbf{K}_{st}$, hence the indices $(u, v)$ are irrelevant. Exploiting symmetry we may impose $(s \prec t)$ without loss of generality. The same holds for the second term. The third term remains unchanged, which completes the proof. ■

We now show that $\mathrm{HSIC}_1(\mathcal{F}, \mathcal{G}, Z)$ is concentrated and that it converges to $\mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy})$ with rate $1/\sqrt{m}$. The latter is a slight improvement over the convergence of the biased estimator $\mathrm{HSIC}_0(\mathcal{F}, \mathcal{G}, Z)$, proposed by Gretton et al. (2005).

**Theorem 4 (HSIC is Concentrated)** *Assume $k, l$ are bounded almost everywhere by 1, and are non-negative. Then for $m > 1$ and all $\delta > 0$, with probability at least $1 - \delta$ for all $\Pr_{xy}$*

$$\left|\mathrm{HSIC}_1(\mathcal{F}, \mathcal{G}, Z) - \mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy})\right| \leq 8\sqrt{\log(2/\delta)/m}$$

**Proof** [Sketch] By virtue of (12) we see immediately that $\mathrm{HSIC}_1$ is a U-statistic of order 4, where each term is contained in $[-2, 2]$. Applying Hoeffding's bound for U-statistics as in (Gretton et al., 2005) proves the result. ■

If $k$ and $l$ were just bounded by 1 in terms of absolute value the bound of Theorem 4 would be worse by a factor of 2.

### 2.5 Asymptotic Normality

Theorem 4 gives *worst case* bounds on the deviation between HSIC and $\mathrm{HSIC}_1$. However, in many cases more accurate bounds for the *typical* case are needed. In particular, we would like to know the limiting distribution of $\mathrm{HSIC}_1$ for large sample sizes. We now show that, in fact, $\mathrm{HSIC}_1$ is asymptotically normal and we derive its variance. These results are useful since they allow us to formulate statistics for a significance test. In term of feature selection, this provides us with an assessment of the functional dependence between selected features and labels.

**Theorem 5 (Asymptotic Normality)** *If $\mathbb{E}[h^2] < \infty$, and data and labels are not independent, then as $m \to \infty$, $\mathrm{HSIC}_1$ converges in distribution to a Gaussian random variable with mean $\mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy})$ and estimated variance*

$$\sigma^2_{\mathrm{HSIC}_1} = \frac{16}{m}\left(R - \mathrm{HSIC}_1^2\right) \quad \textit{where} \quad R = \frac{1}{m}\sum_{i=1}^{m}\Big((m-1)_3^{-1}\sum_{(j,q,r)\in \mathbf{i}_3^m \setminus\{i\}} h(i,j,q,r)\Big)^2, \tag{17}$$

*where $\mathbf{i}_n^m \setminus \{i\}$ denotes the set of all $n$-tuples drawn without replacement from $\{1, \ldots, m\}\setminus\{i\}$.*

**Proof** [Sketch] This follows directly from Serfling (1980, Theorem B, p. 193), which shows asymptotic normality of U-statistics. ■

Unfortunately (17) is expensive to compute by means of an explicit summation: even computing the kernel $h$ of the U-statistic itself is a nontrivial task. For practical purposes we need an expression which can exploit fast matrix operations. As we shall see, $\sigma^2_{\mathrm{HSIC}_1}$ can be computed in $O(m^2)$, given the matrices $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$. To do so, we first form a vector $\mathbf{h}$ with

its $i$th entry corresponding to $\sum_{(j,q,r)\in \mathbf{i}_3^m \setminus \{i\}} h(i,j,q,r)$. Collecting terms in (13) related to matrices $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$, $\mathbf{h}$ can be written as

$$\begin{aligned}\mathbf{h} =&(m-2)^2(\tilde{\mathbf{K}}\circ\tilde{\mathbf{L}})\,\mathbf{1}+(m-2)\Big((\operatorname{tr}\tilde{\mathbf{K}}\,\tilde{\mathbf{L}})\,\mathbf{1}-\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1}-\tilde{\mathbf{L}}\,\tilde{\mathbf{K}}\,\mathbf{1}\Big)-m(\tilde{\mathbf{K}}\,\mathbf{1})\circ(\tilde{\mathbf{L}}\,\mathbf{1})\\ &+(\mathbf{1}^\top\tilde{\mathbf{L}}\,\mathbf{1})\,\tilde{\mathbf{K}}\,\mathbf{1}+(\mathbf{1}^\top\tilde{\mathbf{K}}\,\mathbf{1})\,\tilde{\mathbf{L}}\,\mathbf{1}-(\mathbf{1}^\top\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1})\,\mathbf{1}\end{aligned} \tag{18}$$

where $\circ$ denotes elementwise matrix multiplication. Then $R$ in (17) can be computed as $R=(4m)^{-1}(m-1)_3^{-2}\,\mathbf{h}^\top\mathbf{h}$. Combining this with the the unbiased estimator in (7) leads to the matrix computation of $\sigma^2_{\mathrm{HSIC}_1}$.

### 2.6 Computation

**Exact Computation of $\mathrm{HSIC}_0$ and $\mathrm{HSIC}_1$** Note that both $\mathrm{HSIC}_0$ and $\mathrm{HSIC}_1$ are simple to compute, since only the kernel matrices $\mathbf{K}$ and $\mathbf{L}$ are needed, and no density estimation is involved. Assume that computing an entry in $\mathbf{K}$ and $\mathbf{L}$ takes constant time, then computing the full matrix takes $\mathcal{O}(m^2)$ time. In term of the sample size $m$, we have the following analysis of the time complexity of $\mathrm{HSIC}_0$ and $\mathrm{HSIC}_1$ (by considering summation and multiplication as atomic operations):

**$\mathrm{HSIC}_0$** Centering $\mathbf{L}$ takes $\mathcal{O}(m^2)$ time. Since $\operatorname{tr}(\mathbf{K}\,\mathbf{H}\,\mathbf{L}\,\mathbf{H})$ is equivalent to $\mathbf{1}^\top(\mathbf{K}\circ\mathbf{H}\,\mathbf{L}\,\mathbf{H})\,\mathbf{1}$, it also takes $\mathcal{O}(m^2)$ time. Overall, computing $\mathrm{HSIC}_0$ takes $\mathcal{O}(m^2)$ time.

**$\mathrm{HSIC}_1$** Each of the three terms in $\mathrm{HSIC}_1$, namely $\operatorname{tr}(\tilde{\mathbf{K}}\,\tilde{\mathbf{L}})$, $\mathbf{1}^\top\tilde{\mathbf{K}}\,\mathbf{1}\,\mathbf{1}^\top\tilde{\mathbf{L}}\,\mathbf{1}$ and $\mathbf{1}^\top\tilde{\mathbf{K}}\,\tilde{\mathbf{L}}\,\mathbf{1}$, takes $\mathcal{O}(m^2)$ time. Overall, computing $\mathrm{HSIC}_1$ also takes $\mathcal{O}(m^2)$ time.

**Approximate Computation of $\mathrm{HSIC}_0$ and $\mathrm{HSIC}_1$** Further speedup is also possible via a low rank approximation of the kernel matrices. Particularly, using incomplete Cholesky decomposition, Gretton et al. (2005) derive an efficient approximation of $\mathrm{HSIC}_0$. Formally, it can be summarized as the following lemma:

**Lemma 6 (Efficient Approximation to $\mathrm{HSIC}_0$)** *Let $\mathbf{K}\approx\mathbf{A}\,\mathbf{A}^\top$ and $\mathbf{L}\approx\mathbf{B}\,\mathbf{B}^\top$, where $\mathbf{A}\in\mathbb{R}^{m\times d_f}$ and $\mathbf{B}\in\mathbb{R}^{m\times d_g}$. Then $\mathrm{HSIC}_0$ can be approximated in $\mathcal{O}(m(d_f^2+d_g^2))$ time.*

Note that in this case the dominant computation comes from the incomplete Cholesky decomposition, which can be carried out in $\mathcal{O}(md_f^2)$ and $\mathcal{O}(md_g^2)$ time respectively (Fine and Scheinberg, 2000).

The three terms in $\mathrm{HSIC}_1$ can be computed analogously. Denote by $\mathbf{D}_\mathbf{K}=\operatorname{diag}(\mathbf{A}\,\mathbf{A}^\top)$ and $\mathbf{D}_\mathbf{L}=\operatorname{diag}(\mathbf{B}\,\mathbf{B}^\top)$ the diagonal matrices of the approximating terms. The latter can be computed in $\mathcal{O}(md_f)$ and $\mathcal{O}(md_g)$ time respectively. We have

$$\mathbf{1}^\top\tilde{\mathbf{K}}\,\mathbf{1}=\mathbf{1}^\top(\mathbf{A}\,\mathbf{A}^\top-\mathbf{D}_\mathbf{K})\,\mathbf{1}=\|\,\mathbf{1}^\top\mathbf{A}\,\|^2+\mathbf{1}^\top\mathbf{D}_\mathbf{K}\,\mathbf{1}\,.$$

Computation requires $\mathcal{O}(md_f)$ time. The same holds when computing $\mathbf{1}^\top\tilde{\mathbf{L}}\,\mathbf{1}$. To obtain the second term we exploit that

$$\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\,\mathbf{1}=\mathbf{1}^\top(\mathbf{A}\,\mathbf{A}^\top-\mathbf{D}_\mathbf{K})(\mathbf{B}\,\mathbf{B}^\top-\mathbf{D}_\mathbf{K})\,\mathbf{1}=((\mathbf{A}(\mathbf{A}^\top\mathbf{1}))-\mathbf{D}_\mathbf{K}\,\mathbf{1})^\top((\mathbf{B}(\mathbf{B}^\top\mathbf{1}))-\mathbf{D}_\mathbf{L}\,\mathbf{1}).$$

This can be computed in $\mathcal{O}(m(d_f+d_g))$. Finally, to compute the third term we use

$$\begin{aligned}\operatorname{tr}\tilde{\mathbf{K}}\tilde{\mathbf{L}}&=\operatorname{tr}(\mathbf{A}\,\mathbf{A}^\top-\mathbf{D}_\mathbf{K})(\mathbf{B}\,\mathbf{B}^\top-\mathbf{D}_\mathbf{L})\\ &=\|\,\mathbf{A}^\top\mathbf{B}\,\|^2_{\mathrm{Frob}}-\operatorname{tr}\mathbf{B}^\top\mathbf{D}_\mathbf{K}\,\mathbf{B}-\operatorname{tr}\mathbf{A}^\top\mathbf{D}_\mathbf{L}\,\mathbf{A}+\operatorname{tr}\mathbf{D}_\mathbf{K}\,\mathbf{D}_\mathbf{L}\,.\end{aligned}$$

This can be computed in $\mathcal{O}(md_fd_g)$ time. It is the most costly of all operations, since it takes all interactions between the reduced factorizations of $\mathbf{K}$ and $\mathbf{L}$ into account. Hence we may compute $\mathrm{HSIC}_1$ efficiently (note again that dominant computation comes from the incomplete Cholesky decomposition):

**Lemma 7 (Efficient Approximation of $\mathrm{HSIC}_1$)** *Let $\mathbf{K} \approx \mathbf{A}\,\mathbf{A}^\top$ and $\mathbf{L} \approx \mathbf{B}\,\mathbf{B}^\top$, where $\mathbf{A} \in \mathbb{R}^{m\times d_f}$ and $\mathbf{B} \in \mathbb{R}^{m\times d_g}$. Then $\mathrm{HSIC}_1$ can be approximated in $\mathcal{O}(m(d_f^2 + d_g^2))$ time.*

**Variance of $\mathrm{HSIC}_1$** To compute the variance of $\mathrm{HSIC}_1$ we also need to deal with $(\tilde{\mathbf{K}} \circ \tilde{\mathbf{L}})\,\mathbf{1}$. For the latter, no immediate linear algebra expansion is available. However, we may use of the following decomposition. Assume that $\mathbf{a}$ and $\mathbf{b}$ are vectors in $\mathbb{R}^m$. In this case

$$((\mathbf{a}\,\mathbf{a}^\top) \circ (\mathbf{b}\,\mathbf{b}^\top))\,\mathbf{1} = (\mathbf{a} \circ \mathbf{b})(\mathbf{a} \circ \mathbf{b})^\top\,\mathbf{1}$$

which can be computed in $\mathcal{O}(m)$ time. Hence we may compute

$$((\mathbf{A}\,\mathbf{A}^\top) \circ (\mathbf{B}\,\mathbf{B}^\top))\,\mathbf{1} = \sum_{i=1}^{d_f}\sum_{j=1}^{d_g}((\mathbf{A}_i \circ \mathbf{B}_j)(\mathbf{A}_i \circ \mathbf{B}_j)^\top)\,\mathbf{1}$$

which can be carried out in $\mathcal{O}(md_fd_g)$ time. To take care of the diagonal corrections note that $(\mathbf{A}\,\mathbf{A}^\top - \mathbf{D}_\mathbf{K}) \circ \mathbf{D}_\mathbf{L} = \mathbf{0}$. The same holds for $\mathbf{B}$ and $\mathbf{D}_\mathbf{K}$. The remaining term $\mathbf{D}_\mathbf{K}\,\mathbf{D}_\mathbf{L}\,\mathbf{1}$ is obviously also computable in $\mathcal{O}(m)$ time.

## 3. Notation

In the following sections, we will deal mainly with vectorial data. Whenever we have vectorial data, we use $\mathbf{X}$ as a shorthand to denote the matrix of all vectorial observations $\mathbf{x}_i \in \mathbb{R}^d$ (the $i$th row of $\mathbf{X}$ corresponds to $\mathbf{x}_i^\top$). Likewise, whenever the labels can be bundled into a matrix $\mathbf{Y}$ or a vector $\mathbf{y}$ (for binary classification), we will use the latter for a more concise notation. Also, we will refer to the $j$th column of $\mathbf{X}$ and $\mathbf{Y}$ as $\mathbf{x}_{*j}$ and $\mathbf{y}_{*j}$ respectively as needed.

Furthermore, we denote the mean and standard deviation of the $j$th feature (dimension) by $\bar{\mathrm{x}}_j = \frac{1}{m}\sum_i^m x_{ij}$ and $s_j = (\frac{1}{m}\sum_i^m (x_{ij} - \bar{\mathrm{x}}_j)^2)^{1/2}$ respectively ($x_{ij}$ is the value of the $j$th feature of data $\mathbf{x}_i$). For binary classification problems we denote by $m_+$ and $m_-$ the numbers of positive and negative observations. Moreover, $\bar{\mathrm{x}}_{j+}$ and $\bar{\mathrm{x}}_{j-}$ correspond respectively to the means of the positive and negative classes at the $j$th feature (the corresponding standard deviations are $s_{j+}$ and $s_{j-}$). More generally, let $m_y$ be the number of samples with class label equal to $y$ (this notation is also applicable to multiclass problems). Finally, let $\mathbf{1}_n$ be a vector of all ones with length $n$ and $\mathbf{0}_n$ be a vector of all zeros.

For non-vectorial or scalar data, we will use lower case letters to denote them. Very often the labels are scalars, we use $y$ to denote them. The mean and standard deviation of the labels are $\bar{\mathrm{y}}$ and $s_y$ respectively.

## 4. Feature Selection via HSIC

Having defined our feature selection *criterion*, we now describe *algorithms* that conduct feature selection on the basis of this dependence measure. Denote by $\mathcal{S}$ the full set of

features, $\mathcal{T}$ a subset of features ($\mathcal{T} \subseteq \mathcal{S}$). We want to find $\mathcal{T}$ such that the dependence between features in $\mathcal{T}$ and the labels is maximized. Moreover, we may choose between different feature selection strategies, that is, whether we would like to build up a catalog of features in an incremental fashion (forward selection) or whether we would like to remove irrelevant features from a catalog (backward selection). For certain kernels, such as a linear kernel, both selection methods are equivalent: the objective function decomposes into individual coordinates, and thus feature selection can be done without recursion in one go. Although forward selection is computationally more efficient, backward elimination in general yields better features (especially for nonlinear features), since the quality of the features is assessed within the context of all other features (Guyon and Elisseeff, 2003).

### 4.1 Backward Elimination Using HSIC (BAHSIC)

BAHSIC works by generating a list $\mathcal{S}^\dagger$ which contains the features in increasing degree of relevance. At each step $\mathcal{S}^\dagger$ is appended by a feature from $\mathcal{S}$ which is not contained in $\mathcal{S}^\dagger$ yet by selecting the features which are least dependent on the reference set (i.e. $Y$ or the full set $X$).

Once we perform this operation, the feature selection problem in (1) can be solved by simply taking the last $t$ elements from $\mathcal{S}^\dagger$. Our algorithm produces $\mathcal{S}^\dagger$ recursively, eliminating the least relevant features from $\mathcal{S}$ and adding them to the end of $\mathcal{S}^\dagger$ at each iteration. For convenience, we also denote HSIC as $\mathrm{HSIC}(\sigma, \mathcal{S})$, where $\mathcal{S}$ are the features used in computing the data kernel matrix $\mathbf{K}$, and $\sigma$ is the parameter for the data kernel (for instance, this might be the size of a Gaussian kernel $k(\mathbf{x}, \mathbf{x}') = \exp(-\sigma \|\mathbf{x} - \mathbf{x}'\|^2)$).

Step 3 of the algorithm denotes a policy for adapting the kernel parameters. Depending on the availability of prior knowledge and the type of preprocessing, we explored three types of policies

1. If we have prior knowledge about the nature of the nonlinearity in the data, we can use a fixed kernel parameter throughout the iterations. For instance, we can use a polynomial kernel of fixed degree, e.g. $(\langle \mathbf{x}, \mathbf{x}' \rangle + 1)^2$, to select the features for the XOR dataset showed in Figure 1(a).

2. If we have no prior knowledge, we can optimize HSIC over a set of kernel parameters. In this case, the policy corresponds to $\arg\max_{\sigma \in \Theta} \mathrm{HSIC}(\sigma, \mathcal{S})$, where $\Theta$ is a set of parameters that ensure the kernel is bounded. For instance, $\sigma$ can be the scale parameter of a Gaussian kernel, $k(\mathbf{x}, \mathbf{x}') = \exp(-\sigma \|\mathbf{x} - \mathbf{x}'\|^2)$. Optimizing over the scaling parameter allows us to adapt to the scale of the nonlinearity present in the (feature-reduced) data.

3. Adapting kernel parameters via optimization is computational intensive. Alternatively we can use a policy that produces approximate parameters in each iteration. For instance, if we normalize each feature separately to zero mean and unit variance, we know that the expected value of the distance between data points, $\mathbb{E}\left[(\mathbf{x} - \mathbf{x}')^2\right]$, is $2d$ ($d$ is the dimension of the data). When using a Gaussian kernel, we can then use a policy that assigns $\sigma$ to $1/(2d)$ as the dimension of the data is reduced.

Step 4 of the algorithm is concerned with the selection of a set $\mathcal{I}$ of features to eliminate. While one could choose a single element of $\mathcal{S}$, this would be inefficient when there are a

**Algorithm 1** BAHSIC

**Input**: The full set of features $\mathcal{S}$
**Output**: An ordered set of features $\mathcal{S}^\dagger$

1: $\mathcal{S}^\dagger \leftarrow \varnothing$
2: **repeat**
3: $\quad\sigma \leftarrow \Xi$
4: $\quad\mathcal{I} \leftarrow \arg\max_{\mathcal{I}} \sum_{j\in\mathcal{I}} \mathrm{HSIC}(\sigma, \mathcal{S}\setminus\{j\}), \quad \mathcal{I} \subset \mathcal{S}$
5: $\quad\mathcal{S} \leftarrow \mathcal{S}\setminus\mathcal{I}$
6: $\quad\mathcal{S}^\dagger \leftarrow \mathcal{S}^\dagger \cup \mathcal{I}$
7: **until** $\mathcal{S} = \varnothing$

large number of irrelevant features. On the other hand, removing too many features at once risks the loss of relevant features. In our experiments, we found a good compromise between speed and feature quality was to remove 10% of the current features at each iteration.

In BAHSIC, the kernel matrix $\mathbf{L}$ for the labels is fixed through the whole process. It can be precomputed and stored for speedup if needed. Therefore, the major computation comes from repeated calculation of the kernel matrix $\mathbf{K}$ for the dimension-reduced data. Assume that computing an entry in $\mathbf{K}$ requires constant time irrespective of the dimension of the data, then the $i$th iteration of BAHSIC takes $\mathcal{O}(\beta^{i-1}dm^2)$ time ($d$ is the total number of features, $\beta^{i-1}d$ features remain after $i-1$ iterations, we have $m^2$ elements in the kernel matrix in total). If we want to reduce the number of features to $t$ we need at most $\tau = \log_\beta(t/d)$ iterations. This brings the total time complexity to $\mathcal{O}\left(\frac{1-\beta^\tau}{1-\beta}dm^2\right) = \mathcal{O}\left(\frac{d-t}{1-\beta}m^2\right)$ operations. When using incomplete Cholesky factorization we may reduce computational complexity somewhat further to $\mathcal{O}\left(\frac{d-t}{1-\beta}m(d_f^2+d_g^2)\right)$ time. This saving is significant as long as $d_f d_g < m$, which may happen, for instance whenever $\mathbf{Y}$ is a binary label matrix. In this case $d_g = 1$, hence incomplete factorizations may yield significant computational gains.

### 4.2 Forward Selection Using HSIC (FOHSIC)

FOHSIC uses the converse approach to backward selection: it builds a list of features in *decreasing* degree of relevance. This is achieved by adding one feature at a time to the set of features $\mathcal{S}^\dagger$ obtained so far using HSIC as a criterion for the quality of the so-added features. For faster selection of features, we can choose a group of features (for instance, a fixed proportion $\gamma$) at step 4 and add them in one shot at step 6. The adaptation of kernel parameters in step 3 follows the same policies as those for BAHSIC. The feature selection problem in (1) can be solved by simply taking the *first* $t$ elements from $\mathcal{S}^\dagger$.

**Time Complexity** Under the same assumption as BAHSIC, the $i$th iteration of FOHSIC takes $\mathcal{O}((1-\gamma)^{i-1}dm^2)$ time. The total number of iterations $\tau$ to obtain $t$ features is $t = [1-(1-\gamma)^\tau]d$, that is $\tau = \frac{\log(d-t)-\log d}{\log(1-\gamma)}$ iterations. Performing $\tau$ steps will therefore take $\sum_{i=0}^{\tau-1} d(1-\gamma)^i = d(1-(1-\gamma)^\tau)/\gamma = t/\gamma$ operations. This means that FOHSIC takes $O(tm^2/\gamma)$ time to extract $t$ features.

**Algorithm 2** FOHSIC

**Input**: The full set of features $\mathcal{S}$, desired number of features $t$
**Output**: An ordered set of features $\mathcal{S}^\dagger$

$\mathcal{S}^\dagger \leftarrow \varnothing$
**repeat**
  $\sigma \leftarrow \Xi$
  $\mathcal{I} \leftarrow \arg\max_{\mathcal{I}} \sum_{j\in\mathcal{I}} \mathrm{HSIC}(\sigma, \mathcal{S}^\dagger \cup \{j\}), \quad \mathcal{I} \subset \mathcal{S}$
  $\mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{I}$
  $\mathcal{S}^\dagger \leftarrow \mathcal{S}^\dagger \cup \mathcal{I}$
**until** $|\mathcal{S}^\dagger| = t$

## 5. Variants of BAHSIC

So far we discussed a set of algorithms to select features *once* we decided to choose a certain family of kernels $k, l$ to measure dependence between two sets of observations. We now proceed to discussing a number of design choices for $k$ and $l$. This will happen in two parts: in the current section we discuss generic choices of kernels on data and labels. Various combinations of such kernels will then lead to new algorithms that aim to discover different types of dependence between features and labels (or between a full and a restricted dataset whenever we are interested in unsupervised feature selection). After that (in section 6) we will study specific choices of kernels which correspond to existing feature selection methods.

### 5.1 Kernels on Data

There exists a great number of kernels on data. Obviously, different kernels will correspond to a range of different assumptions on the type of dependence between the random variables $x$ and $y$. Hence different kernels induce distinctive similarity measure on the data.

**Linear kernel** The simplest choice for $k$ is to take a linear kernel $k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$. This means that we are just using the underlying Euclidean space to define the similarity measure. Whenever the dimensionality $d$ of $\mathbf{x}$ is very high, this may allow for more complexity in the function class than what we could measure and assess otherwise. An additional advantage of this setting is that the kernel decomposes into the sum of products between individual coordinates. This means that any expression of the type $\mathrm{tr}\,\mathbf{K}\,\mathbf{M}$ can be maximized with respect to the subset of available features via

$$\sum_{j=1}^{d} \mathbf{x}_{*j}^\top \mathbf{M}\, \mathbf{x}_{*j} \tag{19}$$

This means that the optimality criterion decomposes into a sum over the scores of individual coordinates. Hence maximization with respect to a subset of size $t$ is trivial, since it just involves finding the $t$ largest contributors. Using (11) we can see that for $\mathrm{HSIC}_1$ the matrix $\mathbf{M}$ is given by

$$\mathbf{M} = \frac{1}{m(m-3)} \left[ \tilde{\mathbf{L}} + \left( \mathbf{1}\,\mathbf{1}^\top - \mathbf{I} \right) \frac{\mathbf{1}^\top \tilde{\mathbf{L}}\, \mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2} \left( \tilde{\mathbf{L}}\, \mathbf{1}\, \mathbf{1}^\top - \mathrm{diag}\left( \tilde{\mathbf{L}}\, \mathbf{1} \right) \right) \right]. \tag{20}$$

These terms are essentially rank-1 and diagonal updates on $\tilde{\mathbf{L}}$, which means that they can be computed very efficiently. Note also that in this case FOHSIC and BAHSIC generate the *optimal* feature selection with respect to the criterion applied.

**Polynomial kernel** Clearly in some cases the use of linear features can be quite limiting. It is possible, though, to use higher order correlations between data for the purpose of feature selection. This is achieved by using a polynomial kernel

$$k(\mathbf{x}, \mathbf{x}') = \left(\langle \mathbf{x}, \mathbf{x}' \rangle + a\right)^b \text{ for some } a \geq 0 \text{ and } b \in \mathbb{N}. \tag{21}$$

This kernel incorporates all polynomial interactions up to degree $b$ (provided that $a > 0$). For instance, if we wanted to take only mean and variance into account, we would only need to consider $b = 2$ and $a = 1$. Placing a higher emphasis on means is achieved by increasing the constant offset $a$.

**Radial Basis Function kernel** Note that polynomial kernels only map data into a *finite* dimensional space: while potentially huge, the dimensionality of polynomials of bounded degree is finite, hence criteria arising from such kernels will not provide us with guarantees for a good dependence measure. On the other hand, many radial basis function kernels, such as the Gaussian RBF kernel map $\mathbf{x}$ into an *infinite* dimensional space. One may show that these kernels are in fact universal in the sense of Steinwart (2002). That is, we use kernels of the form

$$k(\mathbf{x}, \mathbf{x}') = \kappa(\| \mathbf{x} - \mathbf{x}' \|) \text{ where } \kappa(\xi) = \exp(-\xi) \text{ or } \kappa(\xi) = \exp(-\xi^2) \tag{22}$$

to obtain Laplace and Gaussian kernels respectively. Since the spectrum of the corresponding matrices is rapidly decaying it is easy to compute incomplete Cholesky factorizations of the kernel matrix efficiently.

**String and Graph kernel** One of the key advantages of our approach is that it is not limited to vectorial data. For instance, we can perform feature selection on documents or graphs. For many such situations we have

$$k(x, x') = \sum_{a \sqsubseteq x} w_a \#_a(x) \#_a(x'), \tag{23}$$

where $a \sqsubseteq x$ is a substring of $x$ (Vishwanathan and Smola, 2003; Leslie et al., 2002). Similar decompositions can be made for graphs, where kernels on random walks and paths can be defined. As before, we could use BAHSIC to remove or FOHSIC to generate a list of features such that only relevant ones remain. That said, given that such kernels are additive in their features, we can use the same argument as made above for linear kernels to determine meaningful features in one go.

### 5.2 Kernels on Labels

The kernels on the data described our inherent assumptions on which properties of $x$ (e.g. linear, polynomial, or nonparametric) are relevant for estimation. We now describe the complementary part, namely a set of possible choices for kernels on labels. Note that these kernels can be just as general as those defined on the data. This means that we

may apply our algorithms to classification, regression, Poisson models, ranking, etc. in the same fashion. This is a significant difference to previous approaches which only apply to specialized settings such as binary classification. For completeness we begin with the latter.

**Binary Classification** The simplest kernel we may choose is

$$l(y, y') = yy' \text{ where } y, y' \in \{\pm 1\}. \tag{24}$$

In this case the label kernel matrix $\mathbf{L} = \mathbf{y}\,\mathbf{y}^\top$ has rank 1 and it is simply the outer product of the vector of labels. Note that we could transform $l$ by adding a positive constant $c$, such as to obtain $l(y, y') = yy' + c$ which yields $l(y, y') = 2\delta_{y,y'}$ for $c = 1$. This transformation, however, is immaterial: once $\mathbf{K}$ has been centered it is orthogonal to constant matrices.

A second transformation, however, leads to nontrivial changes: we may change the relative weights of positive and negative classes. This is achieved by transforming $y \to c_y y$. For instance, we may pick $c_+ = m_+^{-1}$ and $c_- = m_-^{-1}$. That is, we choose

$$\mathbf{y} = \left(m_+^{-1}\,\mathbf{1}_{m_+}^\top, m_-^{-1}\,\mathbf{1}_{m_-}^\top\right)^\top \text{ which leads to } l(y, y') = m_y^{-1} m_{y'}^{-1} yy' \tag{25}$$

That is, we give different weight to positive and negative class according to their sample size. As we shall see in the next section, this corresponds to making the feature selection independent of the class size and it will lead to criteria derived from Maximum Mean Discrepancy estimators (Gretton et al., 2007).

**Multiclass Classification** Here we have a somewhat larger choice of options to contend with. Clearly the simplest kernel would be

$$l(y, y') = c_y \delta_{y,y'} \text{ where } c_y > 0. \tag{26}$$

For $c_y = m_y^{-1}$ we obtain a per-class normalization. Clearly, for $n$ classes, the kernel matrix $\mathbf{L}$ can be represented by the outer product of a rank-$n$ matrix, where each row is given by $c_{y_i}\,\mathbf{e}_{y_i}^\top$, where $\mathbf{e}_y$ denotes the $y$-th unit vector in $\mathbb{R}^n$. Alternatively, we may adjust the inner product between classes to obtain

$$\begin{aligned} l(y, y') &= \left\langle \psi(y), \psi(y') \right\rangle \\ \text{where } \psi(y) &= \mathbf{e}_y \frac{m}{m_y(m - m_y)} - \mathbf{z} \text{ and } \mathbf{z} = ((m - m_1)^{-1}, \ldots, (m - m_n)^{-1})^\top. \end{aligned} \tag{27}$$

This corresponds to assigning a "one versus the rest" feature to each class and taking the inner product between them. As before in the binary case, note that we may drop $\mathbf{z}$ from the expansion, since constant offsets do not change the relative values of HSIC for feature selection. In this case we recover (26) with $c_y = m^2 m_y^{-2}(m - m_y)^{-2}$.

**Regression** This is one of the situations where the advantages of using HSIC are clearly apparent: we are able to adjust our method to such situations simply by choosing appropriate kernels. Clearly, we could just use a linear kernel $l(y, y') = yy'$ which would select simple correlations between data and labels.

Another choice is to use an RBF kernel on the labels, such as

$$l(y, y') = \exp\left(-\bar{\sigma}\left\|y - y'\right\|^2\right). \tag{28}$$

This will ensure that we capture arbitrary nonlinear dependence between $\mathbf{x}$ and $y$. The price is that in this case $\mathbf{L}$ will have full rank, hence computation of BAHSIC and FOHSIC are correspondingly more expensive.

## 6. Connections to Other Approaches

We now show that several feature selection criteria are special cases of BAHSIC by choosing appropriate preprocessing of data and kernels. We will directly relate these criteria to the biased estimator $\mathrm{HSIC}_0$ in (6). Given the fact that $\mathrm{HSIC}_0$ converges to $\mathrm{HSIC}_1$ with rate $\mathcal{O}(m^{-1})$ it follows that the criteria are well related. Additionally we can infer from this that by using $\mathrm{HSIC}_1$ these other criteria could also be improved by correcting their bias. In summary BAHSIC is capable of finding and exploiting dependence of a much more general nature (for instance, dependence between data and labels with graph and string values).

### 6.1 Pearson Correlation

Pearson's correlation is commonly used in microarray analysis (van't Veer et al., 2002; Ein-Dor et al., 2006). It is defined as

$$R_j := \frac{1}{m}\sum_{i=1}^{m}\left(\frac{x_{ij}-\bar{\mathrm{x}}_j}{s_{x_j}}\right)\left(\frac{y_i-\bar{\mathrm{y}}}{s_y}\right) \text{ where} \tag{29}$$

$$\bar{\mathrm{x}}_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij} \text{ and } \bar{\mathrm{y}} = \frac{1}{m}\sum_{i=1}^{m} y_i \text{ and } s_{x_j}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_{ij}-\bar{\mathrm{x}}_j)^2 \text{ and } s_y^2 = \frac{1}{m}\sum_{i=1}^{m}(y_i-\bar{\mathrm{y}})^2.$$

This means that all features are individually centered by $\bar{\mathrm{x}}_j$ and scaled by their coordinate-wise variance $s_{x_j}$ as a preprocessing step. Performing those operations before applying a linear kernel yields the equivalent $\mathrm{HSIC}_0$ formulation:

$$\operatorname{tr}\mathbf{K}\mathbf{H}\mathbf{L}\mathbf{H} = \operatorname{tr}\left(\mathbf{X}\mathbf{X}^\top\mathbf{H}\mathbf{y}\mathbf{y}^\top\mathbf{H}\right) = \left\|\mathbf{H}\mathbf{X}^\top\mathbf{H}\mathbf{y}\right\|^2 \tag{30}$$

$$= \sum_{j=1}^{d}\left(\sum_{i=1}^{m}\left(\frac{x_{ij}-\bar{\mathrm{x}}_j}{s_{x_j}}\right)\left(\frac{y_i-\bar{\mathrm{y}}}{s_y}\right)\right)^2 = \sum_{j=1}^{d} R_j^2. \tag{31}$$

Hence $\mathrm{HSIC}_1$ computes the sum of the squares of the Pearson Correlation (pc) coefficients. Since the terms are additive, feature selection is straightforward by picking the list of best performing features.

### 6.2 Mean Difference and its Variants

The difference between the means of the positive and negative classes at the $j$th feature, $(\bar{\mathrm{x}}_{j+}-\bar{\mathrm{x}}_{j-})$, is useful for scoring individual features. With different normalization of the data and the labels, many variants can be derived. In our experiments we compare a number of these variants. For example, the centroid (lin) (Bedo et al., 2006), $t$-statistic (t), signal-to-noise ratio (snr), moderated $t$-score (m-t) and B-statistics (lods) (Smyth, 2004) all belong to this family. In the following we make those connections more explicit.

**Centroid** Bedo et al. (2006) use $v_j := \lambda\,\bar{\mathrm{x}}_{j+} - (1-\lambda)\,\bar{\mathrm{x}}_{j-}$ for $\lambda \in (0,1)$ as the score for feature $j$.[3] Features are subsequently selected according to the absolute value $|v_j|$. In experiments the authors typically choose $\lambda = \frac{1}{2}$.

For $\lambda = \frac{1}{2}$ we can achieve the same goal by choosing $\mathbf{L}_{ii'} = \frac{y_i y_{i'}}{m_{y_i} m_{y_{i'}}}$ ($y_i, y_{i'} \in \{\pm 1\}$), in which case $\mathbf{H}\,\mathbf{L}\,\mathbf{H} = \mathbf{L}$, since the label kernel matrix is already centered. Hence we have

$$\operatorname{tr} \mathbf{K}\,\mathbf{H}\,\mathbf{L}\,\mathbf{H} = \sum_{i,i'=1}^{m} \frac{y_i y_{i'}}{m_{y_i} m_{y_{i'}}} \mathbf{x}_i^\top \mathbf{x}_{i'} = \sum_{j=1}^{d} \left( \sum_{i,i'=1}^{m} \frac{y_i y_{i'} x_{ij} x_{i'j}}{m_{y_i} m_{y_{i'}}} \right) = \sum_{j=1}^{d} (\bar{\mathrm{x}}_{j+} - \bar{\mathrm{x}}_{j-})^2. \tag{32}$$

This proves that the centroid feature selector can be viewed as a special case of BAHSIC in the case of $\lambda = \frac{1}{2}$. From our analysis we see that other values of $\lambda$ amount to effectively rescaling the patterns $\mathbf{x}_i$ *differently* for different classes, which may lead to undesirable features being selected.

**$t$-Statistic** The normalization for the $j$th feature is computed as

$$\bar{\mathrm{s}}_j = \left[ \frac{s_{j+}^2}{m_+} + \frac{s_{j-}^2}{m_-} \right]^{\frac{1}{2}} \tag{33}$$

In this case we define the $t$-statistic for the $j$th feature via $t_j = (\bar{\mathrm{x}}_{j+} - \bar{\mathrm{x}}_{j-}) / \bar{\mathrm{s}}_j$.

Compared to the Pearson correlation, the key difference is that now we normalize each feature not by the overall sample standard deviation but rather by a value which takes each of the two classes separately into account.

**Signal to noise ratio** is yet another criterion to use in feature selection. The key idea is to normalize each feature by $\bar{\mathrm{s}}_j = s_{j+} + s_{j-}$ instead. Subsequently the $(\bar{\mathrm{x}}_{j+} - \bar{\mathrm{x}}_{j-}) / \bar{\mathrm{s}}_j$ are used to score features.

**Moderated $t$-score** is similar to $t$-statistic and is used for microarray analysis (Smyth, 2004). Its normalization for the $j$th feature is derived via a Bayes approach as

$$\tilde{\mathrm{s}}_j = \frac{m\,\bar{\mathrm{s}}_j^2 + m_0\,\bar{\mathrm{s}}_0^2}{m + m_0} \tag{34}$$

where $\bar{\mathrm{s}}_j$ is from (33), and $\bar{\mathrm{s}}_0$ and $m_0$ are hyperparameters for the prior distribution on $\bar{\mathrm{s}}_j$ (all $\bar{\mathrm{s}}_j$ are assumed to be iid). $\bar{\mathrm{s}}_0$ and $m_0$ are estimated using information from all feature dimensions. This effectively borrows information from the ensemble of features to aid with the scoring of an individual feature. More specifically, $\bar{\mathrm{s}}_0$ and $m_0$ can be computed as (Smyth, 2004)

$$m_0 = 2\Gamma'^{-1}\left( \frac{1}{d} \sum_{j=1}^{d} (z_j - \bar{\mathrm{z}})^2 - \Gamma'\left(\frac{m}{2}\right) \right) \tag{35}$$

$$\bar{\mathrm{s}}_0^2 = \exp\left( \bar{\mathrm{z}} - \Gamma\left(\frac{m}{2}\right) + \Gamma\left(\frac{m_0}{2}\right) - \ln\left(\frac{m_0}{m}\right) \right) \tag{36}$$

3. The parameterization in (Bedo et al., 2006) is different but it can be shown to be equivalent.

where $\Gamma(\cdot)$ is the gamma function, $'$ denotes derivative, $z_j = \ln(\bar{\mathrm{s}}_j^2)$ and $\bar{\mathrm{z}} = \frac{1}{d}\sum_{j=1}^{d} z_j$.

***B*-statistic** is the logarithm of the posterior odds (lods) that a feature is differentially expressed. Lönnstedt and Speed (2002); Smyth (2004) show that, for large number of features, B-statistic is given by

$$B_j = a + b\,\tilde{\mathrm{t}}_j^2 \tag{37}$$

where both $a$ and $b$ are constant ($b > 0$), and $\tilde{\mathrm{t}}_j$ is the moderated-$t$ statistic for the $j$th feature. Here we see that $B_j$ is monotonic increasing in $\tilde{\mathrm{t}}_j$, and thus results in the same gene ranking as the moderated-$t$ statistic.

The reason why these connections work is that the signal-to-noise ratio, moderated $t$-statistic, and $B$-statistic are three variants of the $t$-test. They differ only in their respective denominators, and are thus special cases of $\mathrm{HSIC}_0$ if we normalize the data accordingly.

### 6.3 Maximum Mean Discrepancy

For binary classification, an alternative criterion for selecting features is to check whether the distributions $\Pr(x|y=1)$ and $\Pr(x|y=-1)$ differ and subsequently pick those coordinates of the data which primarily contribute to the difference between the two distributions.

More specifically, we could use Maximum Mean Discrepancy (MMD) (Borgwardt et al., 2006), which is a generalization of mean difference for Reproducing Kernel Hilbert Spaces, given by

$$\mathrm{MMD} = \left\|\mathbb{E}_x\left[\phi(x)|y=1\right] - \mathbb{E}_x\left[\phi(x)|y=-1\right]\right\|_{\mathcal{H}}^2. \tag{38}$$

A biased estimator of the above quantity can be obtained simply by replacing expectations by averages over a finite sample. We relate a biased estimator of MMD to $\mathrm{HSIC}_0$ again by setting $m_+^{-1}$ as the labels for positive samples and $-m_-^{-1}$ for negative samples. If we apply a linear kernel on labels, $\mathbf{L}$ is automatically centered, i.e. $\mathbf{L}\,\mathbf{1} = \mathbf{0}$ and $\mathbf{H}\,\mathbf{L}\,\mathbf{H} = \mathbf{L}$. This yields

$$\begin{aligned}\operatorname{tr}\mathbf{K}\,\mathbf{H}\,\mathbf{L}\,\mathbf{H} &= \operatorname{tr}\mathbf{K}\,\mathbf{L} \\ &= \frac{1}{m_+^2}\sum_{i,j}^{m_+} k(x_i,x_j) + \frac{1}{m_-^2}\sum_{i,j}^{m_-} k(x_i,x_j) - \frac{2}{m_+ m_-}\sum_{i}^{m_+}\sum_{j}^{m_-} k(x_i,x_j) \\ &= \left\|\frac{1}{m_+}\sum_{i}^{m_+}\phi(x_i) - \frac{1}{m_-}\sum_{j}^{m_-}\phi(x_j)\right\|_{\mathcal{H}}^2.\end{aligned} \tag{39}$$

The quantity in the last line is an estimator of MMD with bias $\mathcal{O}(m^{-1})$ (Borgwardt et al., 2006). This implies that $\mathrm{HSIC}_0$ and the biased estimator of MMD are identical up to a constant factor. Since the bias of $\mathrm{HSIC}_0$ is also $\mathcal{O}(m^{-1})$, this effectively show that scaled MMD and $\mathrm{HSIC}_1$ converges to each other with rate $\mathcal{O}(m^{-1})$.

### 6.4 Kernel Target Alignment

Alternatively, one could use Kernel Target Alignment (KTA) (Cristianini et al., 2003) to test directly whether there exists any correlation between data and labels. KTA has been used for feature selection in this context. Formally it is defined as $\operatorname{tr}(\mathbf{K}\,\mathbf{L})/\|\mathbf{K}\|\|\mathbf{L}\|$, that is, the cosine between the kernel matrix and the label matrix.

The nonlinear dependence on $\mathbf{K}$ makes it somewhat hard to optimize for. Indeed, for computational convenience the normalization is often omitted in practise (Neumann et al., 2005), which leaves us with $\operatorname{tr}\mathbf{K}\,\mathbf{L}$, the corresponding estimator of MMD. Note the key difference, though, that normalization of $\mathbf{L}$ according to label size does not occur. Nor does KTA take centering into account. Whenever the sample sizes for both classes are approximately matched, such lack of normalization is negligible and we see that both criteria effectively check a similar criterion.

Hence in some cases in binary classification, selecting features that maximizes HSIC also maximizes MMD and KTA. Note that in general (multiclass, regression, or generic binary classification) this connection does not hold. Moreover, the use of HSIC offers uniform convergence bounds on the tails of the distribution of the estimators.

### 6.5 Shrunken Centroid

The shrunken centroid (pam) method (Tibshirani et al., 2002, 2003) performs feature ranking using the differences from the class centroids to the centroid of all the data, ie.

$$(\bar{\mathrm{x}}_{j+} - \bar{\mathrm{x}}_j)^2 + (\bar{\mathrm{x}}_{j-} - \bar{\mathrm{x}}_j)^2\,, \tag{40}$$

as a criterion to determine the relevance of a given feature. It also scores each feature separately.

To show that this criterion is related to HSIC we need to devise an appropriate map for the labels $y$. Consider the feature map $\psi(y)$ with $\psi(1) = (m_+^{-1}, 0)^\top$ and $\psi(-1) = (0, m_-^{-1})^\top$. Clearly, when applying $\mathbf{H}$ to $\mathbf{Y}$ we obtain the following centered effective feature maps

$$\bar{\psi}(1) = (m_+^{-1} - m^{-1}, -m^{-1}) \text{ and } \bar{\psi}(-1) = (-m^{-1}, m_-^{-1} - m^{-1}). \tag{41}$$

Consequently we may express $\operatorname{tr}\mathbf{K}\,\mathbf{H}\,\mathbf{L}\,\mathbf{H}$ via

$$\operatorname{tr}\mathbf{K}\,\mathbf{H}\,\mathbf{L}\,\mathbf{H} = \left\|\frac{1}{m_+}\sum_{i=1}^{m_+}\mathbf{x}_i - \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\right\|^2 + \left\|\frac{1}{m_-}\sum_{i=1}^{m_-}\mathbf{x}_i - \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\right\|^2 \tag{42}$$

$$= \sum_{j=1}^{d}\left(\left(\frac{1}{m_+}\sum_{i=1}^{m_+}x_{ij} - \frac{1}{m}\sum_{i=1}^{m}x_{ij}\right)^2 + \left(\frac{1}{m_-}\sum_{i=1}^{m_-}x_{ij} - \frac{1}{m}\sum_{i=1}^{m}x_{ij}\right)^2\right) \tag{43}$$

$$= \sum_{j=1}^{d}\left((\bar{\mathrm{x}}_{j+} - \bar{\mathrm{x}}_j)^2 + (\bar{\mathrm{x}}_{j-} - \bar{\mathrm{x}}_j)^2\right) \tag{44}$$

This is the information used by the shrunken centroid method, hence we see that it can be seen to be a special case of HSIC when using a linear kernel on the data and a specific feature map on the labels. Note that we could assign different weights to the two classes,

which would lead to a weighted linear combination of distances from the centroid. Finally, it is straightforward how this definition can be extended to multiclass settings, simply by considering the map $\psi : y \to m_y^{-1} \mathbf{e}_y$.

### 6.6 Ridge Regression

BAHSIC can also be used to select features for regression problems, except that in this case the labels are continuous variables. We could, in principle, use an RBF kernel or similar on the labels to address the feature selection issue. What we show now is that even for a simple linear kernel, interesting results can be obtained. More to the point, we show that feature selection using ridge regression can also be seen to arise as a special case of HSIC feature selection. We assume here that $\mathbf{y}$ is centered.

In ridge regression (Hastie et al., 2001), we estimate the outputs $\mathbf{y}$ using the design matrix $\mathbf{V}$ and a parameter vector $\mathbf{w}$ by minimizing the following regularized risk functional

$$J = \|\mathbf{y} - \mathbf{V}\,\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2 . \tag{45}$$

Here the second term is known as the regularizer. If we choose $\mathbf{V} = \mathbf{X}$ we obtain the family of *linear* models. In the general (nonlinear) case $\mathbf{V}$ may be an arbitrary matrix, where each row consists of a set of basis functions, e.g. a feature map $\phi(\mathbf{x})$. One might conclude that small values of $J$ correspond to good sets of features, since there a $\mathbf{w}$ with small norm would still lead to a small approximation error. It turns out that $J$ is minimized for $\mathbf{w} = (\mathbf{V}^\top \mathbf{V} + \lambda \mathbf{I})^{-1} \mathbf{y}$. Hence the minimum is given by

$$J^* = \mathbf{y}^\top \mathbf{y} - \mathbf{y}^\top \mathbf{V}(\mathbf{V}^\top \mathbf{V} + \lambda \mathbf{I})^{-1} \mathbf{V}^\top \mathbf{y} \tag{46}$$

$$= \text{constant} - \operatorname{tr}\left[\mathbf{V}(\mathbf{V}^\top \mathbf{V} + \lambda \mathbf{I})^{-1} \mathbf{V}^\top\right] \mathbf{y}\,\mathbf{y}^\top . \tag{47}$$

Whenever we are only given $\mathbf{K} = \mathbf{V}^\top \mathbf{V}$ we have the following equality

$$J^* = \text{constant} - \operatorname{tr}\left[\mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1}\right] \mathbf{y}\,\mathbf{y}^\top . \tag{48}$$

This means that the matrices

$$\bar{\mathbf{K}} := \mathbf{V}(\mathbf{V}^\top \mathbf{V} + \lambda \mathbf{I})^{-1} \mathbf{V}^\top \text{ and } \bar{\mathbf{K}} := \mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1} \tag{49}$$

are equivalent kernel matrices to be used in BAHSIC. Note that obviously instead of using $\mathbf{y}\,\mathbf{y}^\top$ as a kernel on the labels $\mathbf{L}$ we could use a nonlinear kernel *in conjunction* with the matrices arrived at from feature selection by ridge regression. It also generalizes the setting of Hastie et al. (2001) to situations other than regression.

### 6.7 Quadratic Mutual Information

Torkkola (2003) introduces the quadratic mutual information for feature selection. That is, he uses the $L_2$ distance between the joint and the marginal distributions on $x$ and $y$ as a criterion for how dependent the two distributions are:

$$I(x, y) = \int\int (\Pr(x, y) - \Pr(x)\Pr(y))^2 \mathrm{d}x\mathrm{d}y \tag{50}$$

In general, (50) is not efficiently computable. That said, when using a Parzen windows estimate of the joint and the marginals, it is possible to evaluate $I(x, y)$ explicitly. Since we only have a finite number of observations, one uses the estimates

$$\hat{p}(x) = \frac{1}{m}\sum_{i=1}^{m} \kappa_x(x_i - x) \tag{51a}$$

$$\hat{p}(y) = \frac{1}{m}\sum_{i=1}^{m} \kappa_y(y_i - y) \tag{51b}$$

$$\hat{p}(x, y) = \frac{1}{m}\sum_{i=1}^{m} \kappa_x(x_i - x)\kappa_y(y_i - y). \tag{51c}$$

Here $\kappa_x$ and $\kappa_y$ are appropriate kernels of the Parzen windows density estimator. Denote by

$$\kappa_{ij} = \int \kappa_x(x_i - x)\kappa_x(x_j - x)dx \text{ and } \nu_{ij} = \int \kappa_y(y_i - y)\kappa_y(y_j - y)dy \tag{52}$$

inner products between Parzen windows kernels. In this case we have

$$\|\hat{p}(x, y) - \hat{p}(x) \cdot \hat{p}(y)\|^2 = m^{-2}\left[\operatorname{tr} \kappa\nu - 2\,\mathbf{1}^\top \kappa\nu\,\mathbf{1} + \mathbf{1}^\top \kappa\,\mathbf{1}\,\mathbf{1}^\top \nu\,\mathbf{1}\right] = m^{-2}\kappa\,\mathbf{H}\,\nu\,\mathbf{H}\,. \tag{53}$$

In other words, we obtain the same criterion as what can be derived from a biased estimator of HSIC. The key difference, though, is that this analogy only works whenever $\kappa$ and $\nu$ can be seen to be arising from an inner product between Parzen windows kernel estimates. This is not universally true: for instance, for graphs, trees, or strings no simple density estimates can be found. This is a serious limitation. Moreover, since we are using a plug-in estimate of the densities, we inherit an innate slow-down of convergence due to the convergence of the density estimators. This issue is discussed in detail in (Anderson et al., 1994).

### 6.8 Recursive Feature Elimination with Support Vectors

Another popular feature selection algorithm is to use Support Vector Machines and to determine the relevance of features by the size of the induced margin as a solution of the dual optimization problem (Guyon et al., 2002). While the connection to BAHSIC is somewhat more tenuous in this context, it is still possible to recast this algorithm in our framework. Before we do so, we describe the basic idea of the method, using $\nu$-SVM instead of plain $C$-SVMs: for $\nu$-SVM without a constant offset $b$ we have the following dual optimization problem (Schölkopf et al., 1999).

$$\underset{\alpha}{\text{minimize}}\, \frac{1}{2}\alpha^\top(\mathbf{K} \circ \mathbf{L})\alpha \text{ subject to } \alpha^\top \mathbf{1} = \nu m \text{ and } \alpha_i \in [0, 1]. \tag{54}$$

This problem is first solved with respect to $\alpha$ for the full set of features. Features are then selected from (54) by removing coordinates such that the objective function decreases least (if at all). For computational convenience, $\alpha$ is not recomputed for a number of feature removals, since repeated solving a quadratic program tends to be computationally expensive.

We now show that this procedure can be viewed as a special case of BAHSIC, where now the class of kernels, parameterized by $\sigma$ is the one of *conformal* kernels. Given a base kernel $k(\mathbf{x}, \mathbf{x}')$ Amari and Wu (1999) propose the following kernel:

$$\bar{k}(\mathbf{x}, \mathbf{x}') = \alpha(\mathbf{x})\alpha(\mathbf{x}')k(\mathbf{x}, \mathbf{x}') \text{ where } \alpha(\mathbf{x}) \geq 0. \tag{55}$$

It is easy to see that

$$\alpha^\top (\mathbf{K} \circ \mathbf{L})\alpha = \mathbf{y}^\top [\operatorname{diag} \alpha] \mathbf{K} [\operatorname{diag} \alpha] \mathbf{y} = \mathbf{y}^\top \bar{\mathbf{K}} \mathbf{y}, \tag{56}$$

where $\bar{\mathbf{K}}$ is the kernel matrix arising from the conformal kernel $\bar{k}(\mathbf{x}, \mathbf{x}')$. Hence for fixed $\alpha$ the objective function is given by a quantity which can be interpreted as a biased version of HSIC. Re-optimization with respect to $\alpha$ is consistent with the kernel adjustment step in Algorithm 1. The only difference being that here the kernel parameters are given by $\alpha$ rather than a kernel width $\sigma$. That said, it is also clear from the optimization problem that this style of feature selection may not be as desirable, since the choice of kernel parameters emphasizes only points close to the decision boundary.

## 7. Experiments

We analyze BAHSIC and related algorithms in an extensive set of experiments. The current section contains results on synthetic and real benchmark data, that is, data from Statlib, the UCI repository, and data from the NIPS feature selection challenge. Sections 8 and 9 then discusses applications to biological data, namely brain signal analysis and feature selection for microarrays.

Since the number of possible choices for feature selection within the BAHSIC family is huge, it is clearly impossible to investigate and compare all of them to all possible other feature selectors. In the present section we pick the following three feature selectors as representative examples. A wider range of kernels and choices is investigated in Section 8 and 9 in the context of biomedical applications.

In this section, we presents three concrete examples of BAHSIC which are used for our later experiments. We apply a Gaussian kernel $k(\mathbf{x}, \mathbf{x}') = \exp(-\sigma \| \mathbf{x} - \mathbf{x}' \|^2)$ on data, while varying the kernels on labels. These BAHSIC variants are dedicated respectively to the following setttings:

**Binary classification (BIN)** Use the feature map in (25) and apply a linear kernel.

**Multiclass classification (MUL)** Use the feature map in (26) and apply a linear kernel.

**Regression problem (REG)** Use the kernel in (28), i.e. a Gaussian RBF kernel on $\mathbf{Y}$.

For the above variants a further speedup of BAHSIC is possible by updating entries in the data kernel matrix incrementally. We use the fact that distance computation of a RBF kernel decomposes into individual coordinates, i.e. we use that $\| \mathbf{x}_i - \mathbf{x}_{i'} \|^2 = \sum_{j=1}^{d} \|x_{ij} - x_{i'j}\|^2$. Hence $\| \mathbf{x}_i - \mathbf{x}_{i'} \|^2$ needs to be computed only once, and subsequent updates are effected by subtracting $\|x_{ij} - x_{i'j}\|^2$.

We will use BIN, MUL and REG as the particular instances of BAHSIC in our experiments. We will refer to them commonly as BAHSIC since the exact meaning will be clear

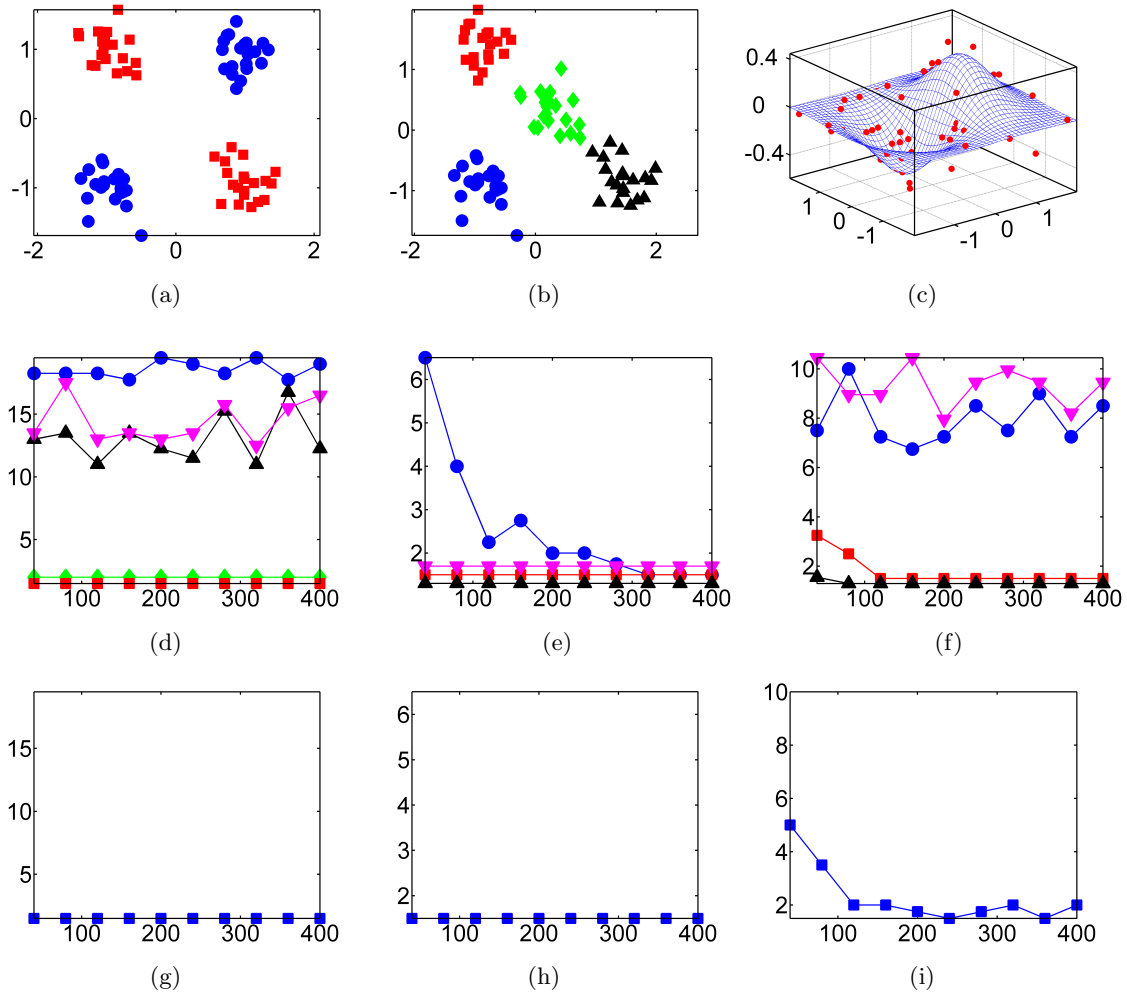


Figure 1: Artificial datasets and the performance of different methods when varying the number of observations. **The first row** contains plots for the first 2 dimension of the (a) binary (b) multiclass and (c) regression data. Different classes are encoded with different colours. **The second row** plots the median rank (y-axis) of the two relevant features as a function of sample size (x-axis) for the corresponding datasets in the first row. **The third row** plots median rank (y-axis) of the two relevant features produced in the first iteration of BAHSIC as a function of the sample size. (Blue circle: Pearson's correlation; Green triangle: RELIEF; Magenta downward triangle: mutual information; Black triangle: FOHSIC; Red square: BAHSIC. Note that RELIEF only works for binary classification.)

depending on the datasets encountered. Furthermore, we also instantiate FOHSIC using the same kernels as BIN, MUL and REG, and we adopt the same convention when we refer to it in our experiments.

### 7.1 Artificial Data

We constructed 3 artificial datasets, as illustrated in Figure 1, to illustrate the difference between BAHSIC variants with linear and nonlinear kernels. Each dataset has 22 dimensions — only the first two dimensions are related to the prediction task and the rest are just Gaussian noise. These datasets are (*i*) **Binary XOR data**: samples belonging to the same class have multimodal distributions; (*ii*) **Multiclass data**: there are 4 classes but 3 of them are collinear; (*iii*) **Nonlinear regression data**: labels are related to the first two dimension of the data by $y = x_1 \exp(-x_1^2 - x_2^2) + \epsilon$, where $\epsilon$ denotes additive Gaussian noise. We compare BAHSIC to FOHSIC, Pearson's correlation, mutual information (Zaffalon and Hutter, 2002), and RELIEF (RELIEF works only for binary problems). We aim to show that when nonlinear dependencies exist in the data, BAHSIC with nonlinear kernels is very competent in finding them.

We instantiate the artificial datasets over a range of sample sizes (from 40 to 400), and plot the median rank, produced by various methods, for the first two dimensions of the data. All numbers in Figure 1 are averaged over 10 runs. In all cases, BAHSIC shows good performance. More specifically, we observe:

**Binary XOR** Both BAHSIC and RELIEF correctly select the first two dimensions of the data even for small sample sizes; while FOHSIC, Pearson's correlation, and mutual information fail. This is because the latter three evaluate the goodness of each feature independently. Hence they are unable to capture nonlinear interaction between features.

**Multiclass Data** BAHSIC, FOHSIC and mutual information select the correct features irrespective of the size of the sample. Pearson's correlation only works for large sample size. The collinearity of 3 classes provides linear correlation between the data and the labels, but due to the interference of the fourth class such correlation is picked up by Pearson's correlation only for a large sample size.

**Nonlinear Regression Data** The performance of Pearson's correlation and mutual information is slightly better than random. BAHSIC and FOHSIC quickly converge to the correct answer as the sample size increases.

In fact, we observe that as the sample size increases, BAHSIC is able to rank the relevant features (the first two dimensions) almost correctly in the first iteration. In the third row of Figure 1, we show the median rank of the relevant features produced in the first iteration as a function of the sample size. It is clear from the pictures that BAHSIC effectively selects features in a single iteration when the sample size is large enough. For the regression case, we also see that BAHSIC with several iterations, indicated by the red square in Figure 1(f), slightly improves the correct ranking over BAHSIC with a single iteration, given by the blue square in Figure 1(i).

While this does not prove BAHSIC with nonlinear kernels is always better than that with a linear kernel, it illustrates the competence of BAHSIC in detecting nonlinear features. This is obviously useful in a real-world situations. The second advantage of BAHSIC is that it is readily applicable to both classification and regression problems, by simply choosing a different kernel on the labels.

### 7.2 Public Benchmark Data

**Algorithms** In this experiment, we show that the performance of BAHSIC can be comparable to other state-of-the-art feature selectors, namely SVM Recursive Feature Elimination (RFE) (Guyon et al., 2002), RELIEF (Kira and Rendell, 1992), $L_0$-norm SVM ($L_0$) (Weston et al., 2003), and R2W2 (Weston et al., 2000). We used the implementation of these algorithms as given in the Spider machine learning toolbox, since those were the only publicly available implementations.[4] Furthermore, we also include filter methods, namely FOHSIC, Pearson's correlation (PC), and mutual information (MI), in our comparisons.

**Datasets** We used various real world datasets taken from the UCI repository,[5] the Statlib repository,[6] the LibSVM website,[7] and the NIPS feature selection challenge[8] for comparison. Due to scalability issues in Spider, we produced a balanced random sample of size less than 2000 for datasets with more than 2000 samples.

**Experimental Protocol** We report the performance of an SVM using a Gaussian kernel on a feature subset of size 5 and 10-fold cross-validation. These 5 features were selected per fold using different methods. Since we are comparing the selected features, we used the same SVM for all methods: a Gaussian kernel with $\sigma$ set as the median distance between points in the sample (Schölkopf and Smola, 2002) and regularization parameter $C = 100$. On classification datasets, we measured the performance using the error rate, and on regression datasets we used the percentage of variance *not*-explained (also known as $1 - r^2$). The results for binary datasets are summarized in the first part of Table 1. Those for multiclass and regression datasets are reported respectively in the second and the third parts of Table 1.

To provide a concise summary of the performance of various methods on binary datasets, we measured how the methods compare with the best performing one in each dataset in Table 1. We recorded the best absolute performance of *all* feature selectors as the baseline, and computed the distance of each algorithm to the best possible result. In this context it makes sense to penalize catastrophic failures more than small deviations. In other words, we would like to have a method which is at least almost always very close to the best performing one. Taking the $\ell_2$ distance achieves this effect, by penalizing larger differences more heavily. It is also our goal to choose an algorithm that performs homogeneously well across all datasets. The $\ell_2$ distance scores are listed for the binary datasets in Table 1. In general, the smaller the $\ell_2$ distance, the better the method. In this respect, BAHSIC and FOHSIC have the best performance. We did not produce the $\ell_2$ distance for multiclass and regression datasets, since the limited number of such datasets did not allow us to draw statistically significant conclusions.

Besides using 5 features, we also plot the performance of the learners as a function of the number of selected features for 9 datasets (covertype, ionosphere, sonar, satimage, segment, vehicle, housing, bodyfat and abalone) in Figure 2. Generally speaking, the smaller the plotted number the better the performance of the corresponding learner. For multiclass

4. http://www.kyb.tuebingen.mpg.de/bs/people/spider
5. http://www.ics.uci.edu/ mlearn/MLSummary.html
6. http://lib.stat.cmu.edu/datasets/
7. http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/
8. http://clopinet.com/isabelle/Projects/NIPS2003/

Table 1: Classification error (%) or percentage of variance *not*-explained (%). The best result, and those results not significantly worse than it, are highlighted in bold (one-sided Welch t-test with 95% confidence level). 100.0±0.0*: program is not finished in a week or crashed. -: not applicable.

| Data | BAHSIC | FOHSIC | PC | MI | RFE | RELIEF | $L_0$ | R2W2 |
|---|---|---|---|---|---|---|---|---|
| covertype | **26.3±1.5** | 37.9±1.7 | 40.3±1.3 | **26.7±1.1** | 33.0±1.9 | 42.7±0.7 | 43.4±0.7 | 44.2±1.7 |
| ionosphere | **12.3±1.7** | **12.8±1.6** | **12.3±1.5** | **13.1±1.7** | 20.2±2.2 | **11.7±2.0** | 35.9±0.4 | 13.7±2.7 |
| sonar | 27.9±3.1 | 25.0±2.3 | 25.5±2.4 | 26.9±1.9 | **21.6±3.4** | 24.0±2.4 | 36.5±3.3 | 32.3±1.8 |
| heart | **14.8±2.4** | **14.4±2.4** | 16.7±2.4 | **15.2±2.5** | 21.9±3.0 | 21.9±3.4 | 30.7±2.8 | 19.3±2.6 |
| breastcancer | 3.8±0.4 | 3.8±0.4 | 4.0±0.4 | 3.5±0.5 | **3.4±0.6** | **3.1±0.3** | 32.7±2.3 | **3.4±0.4** |
| australian | **14.3±1.3** | **14.3±1.3** | **14.5±1.3** | **14.5±1.3** | **14.8±1.2** | **14.5±1.3** | 35.9±1.0 | **14.5±1.3** |
| splice | 22.6±1.1 | 22.6±1.1 | 22.8±0.9 | 21.9±1.0 | **20.7±1.0** | 22.3±1.0 | 45.2±1.2 | 24.0±1.0 |
| svmguide3 | **20.8±0.6** | **20.9±0.6** | 21.2±0.6 | **20.4±0.7** | 21.0±0.7 | 21.6±0.4 | 23.3±0.3 | 23.9±0.2 |
| adult | 24.8±0.2 | 24.4±0.6 | **18.3±1.1** | 21.6±1.1 | 21.3±0.9 | 24.4±0.2 | 24.7±0.1 | 100.0±0.0* |
| cleveland | **19.0±2.1** | **20.5±1.9** | 21.9±1.7 | **19.5±2.2** | 20.9±2.1 | 22.4±2.5 | 25.2±0.6 | 21.5±1.3 |
| derm | **0.3±0.3** | **0.3±0.3** | **0.3±0.3** | **0.3±0.3** | **0.3±0.3** | **0.3±0.3** | 24.3±2.6 | **0.3±0.3** |
| hepatitis | **13.8±3.5** | **15.0±2.5** | **15.0±4.1** | **15.0±4.1** | **15.0±2.5** | 17.5±2.0 | 16.3±1.9 | 17.5±2.0 |
| musk | 29.9±2.5 | 29.6±1.8 | **26.9±2.0** | 31.9±2.0 | 34.7±2.5 | **27.7±1.6** | 42.6±2.2 | 36.4±2.4 |
| optdigits | **0.5±0.2** | **0.5±0.2** | **0.5±0.2** | 3.4±0.6 | 3.0±1.6 | 0.9±0.3 | 12.5±1.7 | 0.8±0.3 |
| specft | **20.0±2.8** | **20.0±2.8** | **18.8±3.4** | **18.8±3.4** | 37.5±6.7 | 26.3±3.5 | 36.3±4.4 | 31.3±3.4 |
| wdbc | **5.3±0.6** | **5.3±0.6** | **5.3±0.7** | 6.7±0.5 | 7.7±1.8 | 7.2±1.0 | 16.7±2.7 | 6.8±1.2 |
| wine | **1.7±1.1** | **1.7±1.1** | **1.7±1.1** | **1.7±1.1** | 3.4±1.4 | 4.2±1.9 | 25.1±7.2 | **1.7±1.1** |
| german | 29.2±1.9 | 29.2±1.8 | 26.2±1.5 | 26.2±1.7 | 27.2±2.4 | 33.2±1.1 | 32.0±0.0 | **24.8±1.4** |
| gisette | **12.4±1.0** | **13.0±0.9** | 16.0±0.7 | 50.0±0.0 | 42.8±1.3 | 16.7±0.6 | 42.7±0.7 | 100.0±0.0* |
| arcene | **22.0±5.1** | **19.0±3.1** | 31.0±3.5 | 45.0±2.7 | 34.0±4.5 | 30.0±3.9 | 46.0±6.2 | 32.0±5.5 |
| madelon | **37.9±0.8** | **38.0±0.7** | 38.4±0.6 | 51.6±1.0 | 41.5±0.8 | 38.6±0.7 | 51.3±1.1 | 100.0±0.0* |
| $\ell_2$ | **11.2** | 14.8 | 19.7 | 48.6 | 42.2 | 25.9 | 85.0 | 138.3 |
| satimage | **15.8±1.0** | 17.9±0.8 | 52.6±1.7 | 22.7±0.9 | 18.7±1.3 | - | 22.1±1.8 | - |
| segment | 28.6±1.3 | 33.9±0.9 | **22.9±0.5** | 27.1±1.3 | 24.5±0.8 | - | 68.7±7.1 | - |
| vehicle | **36.4±1.5** | 48.7±2.2 | 42.8±1.4 | 45.8±2.5 | **35.7±1.3** | - | 40.7±1.4 | - |
| svmguide2 | **22.8±2.7** | **22.2±2.8** | 26.4±2.5 | 27.4±1.6 | 35.6±1.3 | - | 34.5±1.7 | - |
| vowel | **44.7±2.0** | **44.7±2.0** | 48.1±2.0 | **45.4±2.2** | 51.9±2.0 | - | 85.6±1.0 | - |
| usps | **43.4±1.3** | **43.4±1.3** | 73.7±2.2 | 67.8±1.8 | 55.8±2.6 | - | 67.0±2.2 | - |
| housing | **18.5±2.6** | **18.9±3.6** | 25.3±2.5 | **18.9±2.7** | - | - | - | - |
| bodyfat | **3.5±2.5** | **3.5±2.5** | **3.4±2.5** | **3.4±2.5** | - | - | - | - |
| abalone | **55.1±2.7** | **55.9±2.9** | **54.2±3.3** | 56.5±2.6 | - | - | - | - |

and regression datasets, it is clear that the curves for BAHSIC very often lie along the lower bound of all methods. For binary classification, however, SVM-RFE as a member of our framework performs the best in general. The advantage of BAHSIC becomes apparent when a small percentage of features is selected. For instance, BAHSIC is the best when only 5 features are selected from data set 1 and 2. Note that in these cases, the performance produced by BAHSIC is very close to that using all features. In a sense, BAHSIC is able to shortlist the most informative features.

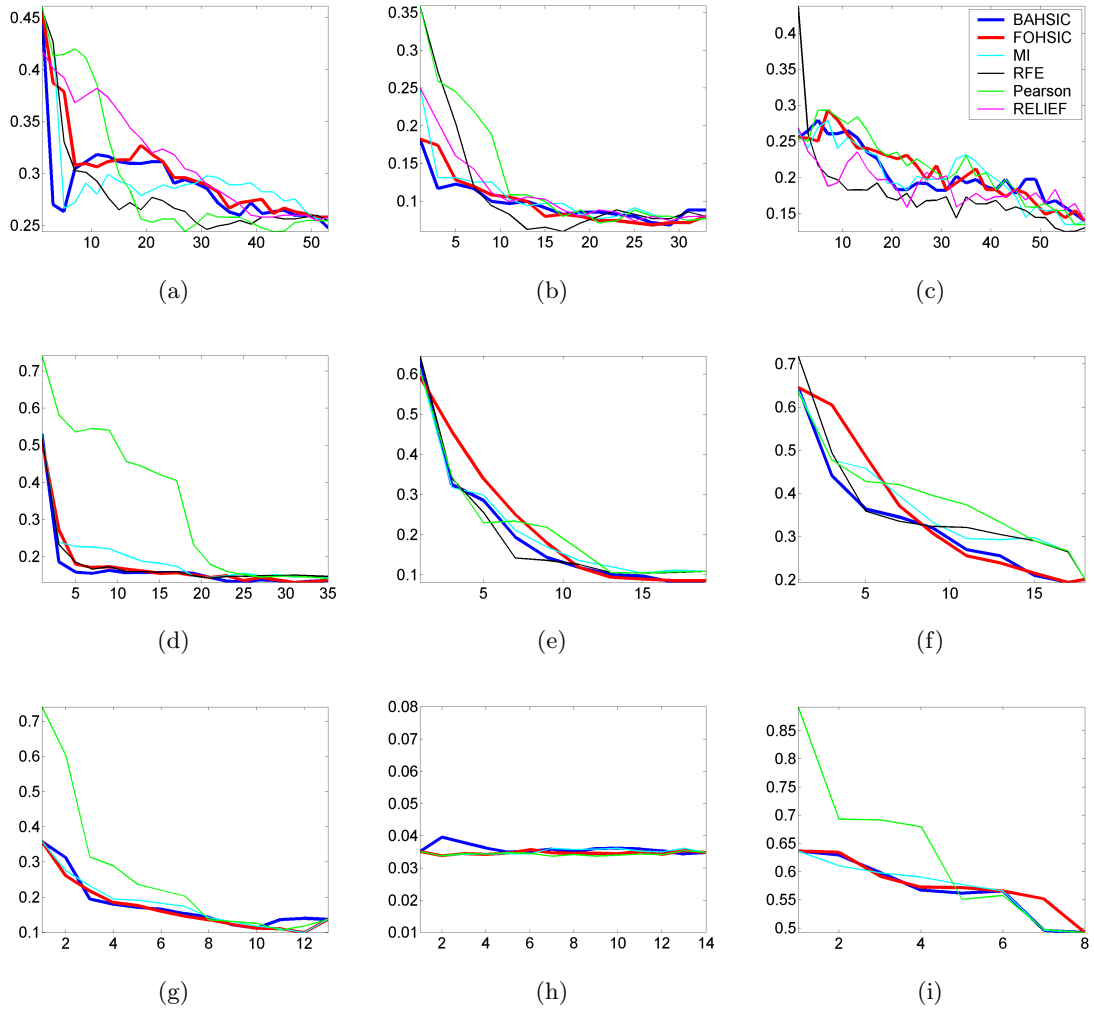


Figure 2: The performance of a classifier or a regressor (vertical axes) as a function of the number of selected features (horizontal axes). Note that the maximum of the horizontal axes are equal to the total number of features in each data set. (a-c) Balanced error rate by a SVM classifier on the binary data sets Covertype (1), Ionosphere (2) and Sonar (3) respectively; (d-f) balanced error rate by a one-versus-the-rest SVM classfier on multiclass data sets Satimage (22), Segment (23) and Vehicle (24) respectively; (g-i) percentage of variance *not*-explained by a SVR regressor on regression data set Housing (25), Body fat (26) and Abalone (27) respectively.

## 8. Analysis of Brain Computer Interface Data

In this experiment, we show that BAHSIC selects features that are meaningful in practice. Here we use it to select a frequency band for a brain-computer interface (BCI) data set from the Berlin BCI group (Dornhege et al., 2004). The data contains EEG signals (118 channels, sampled at 100 Hz) from five healthy subjects ('aa', 'al', 'av', 'aw' and 'ay') recorded during two types of motor imaginations. The task is to classify the imagination for individual trials.

Our experiment proceeds in 3 steps: (*i*) A Fast Fourier transformation (FFT) is performed on each channel and the power spectrum is computed. (*ii*) The power spectra from all channels are averaged to obtain a single spectrum for each trial. (*iii*) BAHSIC is used to select the top 5 discriminative frequency components based on the power spectrum. The 5 selected frequencies and their 4 nearest neighbours are used to reconstruct the temporal signals (with all other Fourier coefficients eliminated). The result is then passed to a normal CSP method (Dornhege et al., 2004) for feature extraction and then classified using a linear SVM.

Automatic filtering using BAHSIC is then compared to other filtering approaches: normal CSP method with manual filtering (8-40 Hz), the CSSP method (Lemm et al., 2005) and the CSSSP method (Dornhege et al., 2006). All results presented in Table 2 are obtained using $50 \times 2$-fold cross-validation. Our method is very competitive and obtains the first and second place for 4 of the 5 subjects. While the CSSP and the CSSSP methods are *specialized* embedded methods (w.r.t. the CSP method) for frequency selection on BCI data, our method is entirely generic. BAHSIC decouples feature selection from CSP, while proving competitive.

In Figure 3, we use HSIC to visualize the responsiveness of different frequency bands to motor imagination. The horizontal and the vertical axes in each subfigure represent the lower and upper bounds for a frequency band, respectively. HSIC is computed for each of these bands. Dornhege et al. (2006) report that the $\mu$ rhythm (approx. 12 Hz) of EEG is most responsive to motor imagination, and that the $\beta$ rhythm (approx. 22 Hz) is also responsive. We expect that HSIC will create a strong peak at the $\mu$ rhythm and a weaker peak at the $\beta$ rhythm, and the absence of other responsive frequency components will create block patterns. Both predictions are confirmed in Figure 3. Furthermore, the large area of the red region for subject 'al' indicates good responsiveness of his $\mu$ rhythm. This also corresponds well with the lowest classification error obtained for him in Table 2.

Table 2: Classification errors (%) on BCI data after selecting a frequency range.

| Method | aa | al | av | aw | ay |
|---|---|---|---|---|---|
| **CSP(8-40Hz)** | 17.5±2.5 | 3.1±1.2 | 32.1±2.5 | 7.3±2.7 | **6.0±1.6** |
| **CSSP** | 14.9±2.9 | 2.4±1.3 | 33.0±2.7 | **5.4±1.9** | 6.2±1.5 |
| **CSSSP** | **12.2±2.1** | 2.2±0.9 | 31.8±2.8 | 6.3±1.8 | 12.7±2.0 |
| **BAHSIC** | 13.7±4.3 | **1.9±1.3** | **30.5±3.3** | 6.1±3.8 | 9.0±6.0 |

## 9. Analysis of Microarray Data

The fact that BAHSIC may be instantiated in numerous ways may create problems for application, that is, it is not immediately clear which criteria we might want to choose. Here we provide guidelines for choosing a specific member of the BAHSIC family by using gene selection as an illustration.

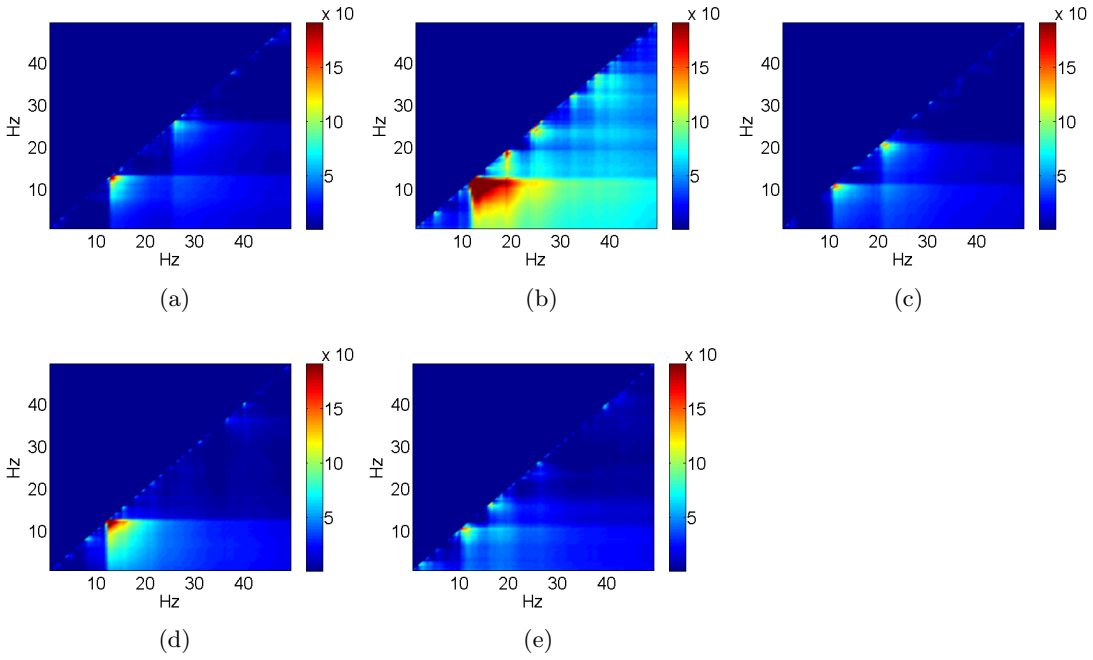


Figure 3: HSIC, encoded by the colour value for different frequency bands. The x-axis corresponds to the upper cutoff and the y-axis denotes the lower cutoff (clearly no signal can be found where the lower bound exceeds the upper bound). Red corresponds to strong dependence, whereas blue indicates that no dependence was found. The figures are for subject (a) 'aa', (b) 'al', (c) 'av', (d) 'aw' and (e) 'ay'.

### 9.1 Datasets

While some past work focused on analysis of a *specific* single microarray dataset we decided to perform a large scale comparison of a raft of techniques on many datasets. We believe that this leads to a more accurate description of the performance of feature selectors. We ran our experiments on 28 datasets, of which 15 are two-class datasets and 13 are multiclass datasets. These datasets are assigned a reference number for convenience. Two-class datasets have a reference number less than or equal to 15, and multiclass datasets have reference numbers of 16 and above. Only one dataset, yeast, has feature dimension less than 1000 (79 features). All other datasets have dimensions ranging from approximately 2000 to 25000. The number of samples varies between approximately 50 and 300 samples. A summary of the datasets and their sources is as follows:

- The six datasets studied in (Ein-Dor et al., 2006). Three deal with breast cancer (van't Veer et al., 2002; van de Vijver et al., 2002; Wang et al., 2005) (numbered 1, 2 and 3), two with lung cancer (Bhattacharjee et al., 2001; Beer et al., 2002) (4, 5), and one with hepatocellular carcinoma (Iizuka et al., 2003) (6). The B cell lymphoma dataset (Rosenwald et al., 2002) is not used because none of the tested methods produce classification errors lower than 40%.

- The six datasets studied in (Warnat et al., 2005). Two deal with prostate cancer (Dhanasekaran et al., 2001; Welsh et al., 2001) (7, 8), two with breast cancer (Gruvberger et al., 2001; West et al., 2001) (9, 10), and two with leukaemia (Bullinger et al., 2004; Valk et al., 2004) (16, 17).

- Five commonly used bioinformatics benchmark datasets on colon cancer (Alon et al., 1999) (11), ovarian cancer (Berchuck et al., 2005) (12), leukaemia (Golub et al., 1999)(13), lymphoma (Alizadeh et al., 2000)(18), and yeast (Brown et al., 2000)(19).

- Nine datasets from the NCBI GEO database. The GDS IDs and reference numbers for this paper are GDS1962 (20), GDS330 (21), GDS531 (14), GDS589 (22), GDS968 (23), GDS1021 (24), GDS1027 (25), GDS1244 (26), GDS1319 (27), GDS1454 (28), and GDS1490 (15), respectively.

### 9.2 Classification Error and Robustness of Genes

We used stratified 10-fold cross-validation and SVMs to evaluate the predictive performance of the top 10 features selected by various members of BAHSIC. For two-class datasets, a nonlinear SVM with an Gaussian RBF kernel, $k(x, x') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$, was used. The regularization constant $C$ and the kernel width $\sigma$ were tuned on a grid of $\{0.1, 1, 10, 10^2, 10^3\} \times \{1, 10, 10^2, 10^3\}$. Classification performance is measured as the fraction of misclassified samples. For multiclass datasets, all procedures are the same except that we used the SVM in a one-versus-the-rest fashion. A new BAHSIC member are also included in the comparison, with kernels $(\|\mathbf{x}-\mathbf{x}'\|+\epsilon)^{-1}$ (dis; $\epsilon$ is a small positive number to avoid singularity) on the data.

The classification results for binary and multiclass datasets are reported in Table 3 and Table 4, respectively. In addition to error rate we also report the overlap between the top 10 gene lists created in each fold. The multiclass results are presented separately since some older members of the BAHSIC family, and some competitors, are not naturally extensible to multiclass datasets. From the experiments we make the following observations:

When comparing the overall performance of various gene selection algorithms, it is of primary interest to choose a method which works well *everywhere*, rather than one which sometimes works well and sometimes performs catastrophically. It turns out that the linear kernel (lin) outperforms all other methods in this regard, both for binary and multiclass problems.

To show this, we measure how various methods compare with the best performing one in each dataset in Tables 3 and 4. The deviation between algorithms is taken as the square of the difference in performance. This measure is chosen because gene expression data is relative expensive to obtain, and we want an algorithm to select the best genes from them. If an algorithm selects genes that are far inferior to the best possible among all algorithms (catastrophic case), we downgrade the algorithm more heavily. Squaring the performance difference achieves exactly this effect, by penalising larger differences more heavily. In other words, we want to choose an algorithm that performs homogeneously well in all datasets. To provide a concise summary, we add these deviations over the datasets and take the square root as the measure of goodness. These scores (called $\ell_2$ distance) are listed in Tables 3 and

4. In general, the smaller the $\ell_2$ distance, the better the method. It can been seen that the linear kernel has the smallest $\ell_2$ distance on both the binary and multiclass datasets.

### 9.3 Subtype Discrimination using Nonlinear Kernels

We now investigate why it is that nonlinear kernels (RBF and dis) provide better genes for classification in three datasets from Table 4 (datasets 18 (Alizadeh et al., 2000), 27 (GDS1319), and 28 (GDS1454)). These datasets all represent multiclass problems, where at least two of the classes are subtypes with respect to the same supertype.[9] Ideally, the selected genes should contain information discriminating the classes. To visualise this information, we plot in Figure 4 the expression value of the top-ranked gene against that of a second gene ranked in the top 10. This second gene is chosen so that it has minimal correlation with the first gene. We use colours and shapes to distinguish data from different classes (datasets 18 and 28 each contain 3 classes, therefore we use 3 different colour and shape combinations for them; dataset 27 has 4 classes, so we use 4 such combinations).

We found that genes selected using nonlinear kernels provide better separation between the two classes that correspond to the same supertype (red dots and green diamonds), while the genes selected with the linear kernel do not separate these subtypes well. In the case of dataset 27, the increased discrimination between red and green comes at the cost of a greater number of errors in another class (black triangle), however these mistakes are less severe than the errors made between the two subtypes by the linear kernel. This eventually leads to better classification performance for the nonlinear kernels (see Table 4).

The principal characteristic of the datasets is that the blue square class is clearly separated from the rest, while the difference between the two subtypes (red dots and green diamonds) is less clear. The first gene provides information that distinguishes the blue square class, however it provides almost no information about the separation between the two subtypes. The linear kernel does not search for information complementary to the first gene, whereas nonlinear kernels are able to incorporate complementary information. In fact, the second gene that distinguishes the two subtypes (red dots and green diamonds) does not separate all classes. From this gene alone, the blue square class is heavily mixed with other classes. However, combining the two genes together results in better separation between all classes.

### 9.4 Rules of Thumb and Implication to Gene Activity

To conclude these experiments, considering the fact that the linear kernel performed best in our feature selection evaluation, yet also taking into account the existence of nonlinear interaction between genes (as demonstrated in section 9.3), we propose the following two rules of thumb for gene selection:

1. Always apply a linear kernel for general purpose gene selection.

9. For dataset 18, the 3 subtypes are diffuse large B-cell lymphoma and leukemia, follicular lymphoma, and chronic lymphocytic leukemia; For dataset 27, the 4 subtypes are various C blastomere mutant embryos: wild type, pie-1, pie-1+pal-1, and mex-3+skn-1; For dataset 28, the 3 subtypes are normal cell, IgV unmutated B-cell, and IgV mutated B-cell.

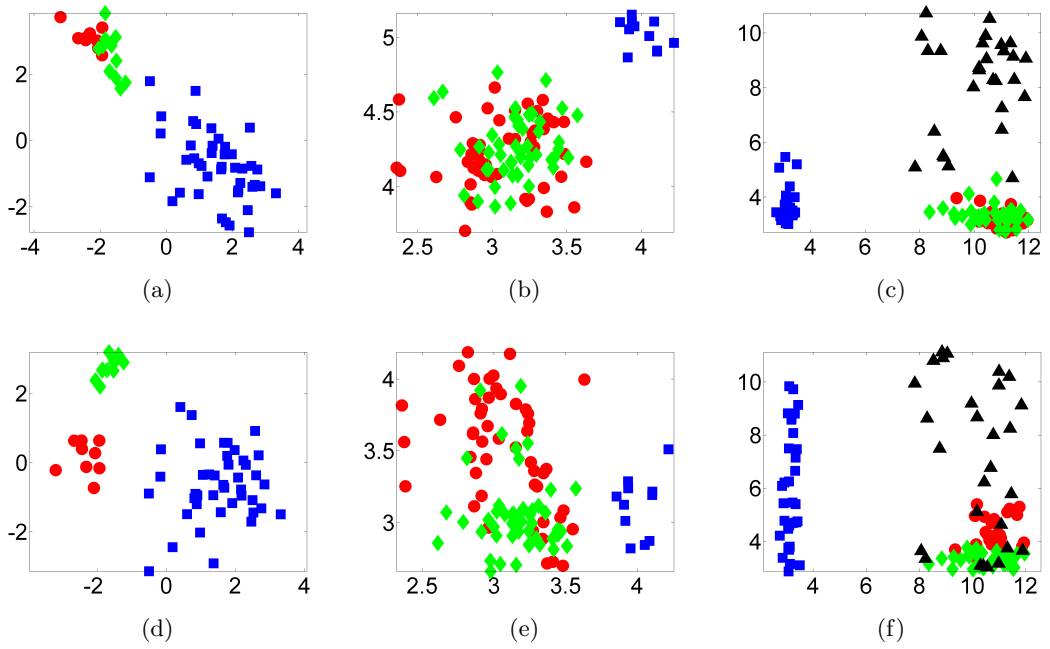


Figure 4: Nonlinear kernels (MUL and dis) select genes that discriminate subtypes (red dots and green diamonds) where the linear kernel fails. The two genes in the first row are representative of those selected by the linear kernel, while those in the second row are produced with a nonlinear kernel for the corresponding datasets. Different colors and shapes represent data from different classes. (a,d) dataset 18; (b,e) dataset 28; and (e,f) dataset 27.

2. Apply a Gaussian kernel if nonlinear effects are present, such as multimodality or complementary effects of different genes.

This result should come as no surprise, due to the high dimensionality of microarray datasets, but we corroborate our claims by means of an extensive experimental evaluation. These experiments also imply a desirable property of gene activity as a whole: it correlates well with the observed outcomes. Multimodal and highly nonlinear situations exist, where a nonlinear feature selector is needed (as can be seen in the outcomes on datasets 18, 27 and 28), yet they occur relatively rarely in practice.

## 10. Conclusion

This paper provides a *unifying* framework for a raft of feature selection methods. This allows us to give tail bounds and asymptotic expansions for feature selectors. Moreover, we are able to design new feature selectors which work well in practice by means of the Hilbert-Schmidt Independence Criterion (HSIC).

The idea behind the resulting algorithm, BAHSIC, is to choose the feature subset that maximises the dependence between the data and labels. The absence of bias and good convergence properties of the empirical HSIC estimate provide a strong theoretical jutification for using HSIC in this context. Although BAHSIC is a filter method, it still demonstrates

Table 3: Two-class datasets: classification error (%) and number of common genes (overlap) for 10-fold cross-validation using the top 10 selected features. Each *row* shows the results for a dataset, and each *column* is a method. Each entry in the table contains two numbers separated by "|": the first number is the classification error and the second number is the number of overlaps. For classification error, the best result, and those results not significantly worse than it, are highlighted in bold (one-sided Welch t-test with 95% confidence level; a table containing the standard errors is provided in the *supplementary material*). For the overlap, largest overlaps for each dataset are highlighted (no significance test is performed). The second last *row* summarises the number of times a method was the best. The last *row* contains the $\ell_2$ distance of the error vectors between a method and the best performing method on each dataset. We use the following abbreviations: pc - Pearson's correlation, snr - signal-to-noise ratio, pam - shrunken centroid, t - t-statistics, m-t - moderated t-statistics, lods - B-statistics, lin - centroid, dis - $(\| \mathbf{x} - \mathbf{x}' \| + \epsilon)^{-1}$, rfe - svm recursive feature elimination)

| **Dataset** | **pc** | **snr** | **pam** | **t** | **m-t** | **lods** | **lin** | **RBF** | **dis** | **rfe** |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 12.7\|3 | 11.4\|3 | 11.4\|**4** | 12.9\|3 | 12.9\|**4** | 12.9\|**4** | 15.5\|3 | 19.1\|1 | 13.9\|2 | 14.3\|0 |
| 2 | 33.2\|1 | 33.9\|2 | 33.9\|1 | **29.5**\|1 | 29.5\|1 | **27.8**\|1 | 32.9\|2 | 31.5\|**3** | 32.8\|2 | 34.2\|0 |
| 3 | 37.4\|0 | 37.4\|0 | 37.4\|0 | **34.6**\|**6** | **34.6**\|**6** | **34.6**\|**6** | 37.4\|1 | 37.4\|0 | 37.4\|0 | 37.4\|0 |
| 4 | 41.6\|0 | **38.8**\|0 | 41.6\|0 | 40.7\|**1** | **40.7**\|0 | **37.8**\|0 | 41.6\|0 | 41.6\|0 | **39.7**\|0 | 41.6\|0 |
| 5 | **27.8**\|0 | **26.7**\|0 | **27.8**\|0 | **26.7**\|**2** | **26.7**\|**2** | **26.7**\|**2** | **27.8**\|0 | **27.8**\|0 | **27.6**\|0 | **27.8**\|0 |
| 6 | 30.0\|2 | **25.0**\|0 | 31.7\|0 | **25.0**\|**5** | **25.0**\|**5** | **25.0**\|**5** | 30.0\|0 | 31.7\|0 | 30.0\|1 | 30.0\|0 |
| 7 | **2.0**\|**6** | **2.0**\|5 | **2.0**\|5 | 28.7\|4 | 26.3\|4 | 26.3\|4 | **2.0**\|3 | **2.0**\|4 | 30.0\|0 | **2.0**\|0 |
| 8 | 3.3\|3 | **0.0**\|4 | **0.0**\|4 | **0.0**\|4 | 3.3\|**6** | 3.3\|**6** | 3.3\|2 | 3.3\|1 | 6.7\|2 | **0.0**\|0 |
| 9 | **10.0**\|**6** | **10.0**\|**6** | **8.7**\|4 | 34.0\|5 | 37.7\|**6** | 37.7\|**6** | 12.0\|3 | **10.0**\|5 | 12.0\|1 | **10.0**\|0 |
| 10 | **16.0**\|2 | **18.0**\|2 | **14.0**\|2 | **14.0**\|8 | 22.0\|**9** | 22.0\|**9** | **16.0**\|2 | **16.0**\|0 | **18.0**\|0 | 32.5\|0 |
| 11 | 12.9\|5 | 12.9\|5 | 12.9\|5 | 19.5\|0 | 22.1\|0 | 33.6\|0 | **11.2**\|4 | **9.5**\|**6** | 16.0\|4 | 19.0\|0 |
| 12 | 30.3\|2 | 36.0\|2 | 31.3\|2 | 26.7\|**3** | 35.7\|0 | 35.7\|0 | **18.7**\|1 | 35.0\|0 | 33.0\|1 | 29.7\|0 |
| 13 | 8.4\|5 | 11.1\|0 | 7.0\|5 | 22.1\|3 | 27.9\|**6** | 15.4\|1 | 7.0\|2 | 9.6\|0 | 11.1\|0 | **4.3**\|1 |
| 14 | 20.8\|1 | 20.8\|1 | 20.2\|0 | 20.8\|**3** | 20.8\|**3** | 20.8\|**3** | 20.8\|0 | 20.2\|0 | **19.7**\|0 | 20.8\|0 |
| 15 | **0.0**\|7 | 0.7\|1 | **0.0**\|5 | 4.0\|1 | 0.7\|**8** | 0.7\|**8** | **0.0**\|3 | **0.0**\|2 | 2.0\|2 | **0.0**\|1 |
| **best** | 5\|2 | 7\|1 | 6\|1 | 6\|6 | 4\|10 | 5\|9 | 6\|0 | 6\|2 | 4\|0 | 6\|0 |
| $\ell_2$ | 16.9 | 20.9 | 17.3 | 43.5 | 50.5 | 50.3 | **13.2** | 22.9 | 35.4 | 26.3 |

Table 4: Multiclass datasets: in this case *columns* are the datasets, and *rows* are the methods. The remaining conventions follow Table 3.

| **Data** | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | **best** | $\ell_2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **lin** | 36.7\|1 | **0.0**\|3 | 5.0\|**3** | 10.5\|6 | 35.0\|3 | **37.5**\|**6** | **18.6**\|1 | **40.3**\|**3** | **28.1**\|**3** | **26.6**\|**6** | **5.6**\|**6** | 27.9\|7 | 45.1\|1 | 7\|6 | **32.4** |
| **RBF** | **33.3**\|**3** | 5.1\|4 | **1.7**\|**3** | **7.2**\|**9** | 33.3\|0 | 40.0\|1 | 22.1\|0 | 72.5\|0 | 39.5\|0 | **24.7**\|4 | **5.6**\|**6** | 22.1\|**10** | **21.5**\|3 | 6\|5 | 37.9 |
| **dis** | **29.7**\|2 | 28.8\|**5** | 6.7\|0 | **8.2**\|**9** | **29.4**\|**7** | **38.3**\|4 | 43.4\|**4** | 66.1\|0 | 40.8\|0 | 38.9\|4 | 7.6\|1 | **8.2**\|8 | 31.6\|3 | 5\|4 | 51.0 |

good performance compared with more specialised methods in both artificial and real world data. It is also very competitive in terms of runtime performance.[10]

A variant of BAHSIC can also be used to perform feature selection for unlabeled data. In this case, we want to select a subset $\mathcal{T}$ of variables such that it is strongly correlated with the full dataset. In other words, we want to find a compressed representation of the data itself in the hope that it is useful for a subsequent learning tasks. BAHSIC readily accommodates this by simply using the full data set $X$ as the labels. Clearly, we want to maximize dependence between the selected variables and $X$ without adding many variables which are simply very much correlated to each other. This ingredient is not yet explicitly formulated in the BAHSIC framework. We will investigate this in the future.

## Appendix: Feature Weighting Using HSIC

Besides the backward elimination algorithm, feature selection using HSIC can also proceed by converting problem (1) into a continuous optimization problem. By adding a penalty on the number of nonzero terms, such as a relaxed $\ell_0$ "norm" of a weight vector over the features we are able to solve the problem with continuous optimization methods. Unfortunately, this approach does not perform as well as the the backward elimination procedure proposed in the main text. For completeness and since related methods are somewhat popular in the literature, the approach is described below.

We introduce a weighting $\mathbf{w} \in \mathbb{R}^n$ on the dimensions of the data: $x \longmapsto \mathbf{w} \circ x$, where $\circ$ denotes element-wise product. Thus feature selection using HSIC becomes an optimization problem with respect to $\mathbf{w}$ (for convenience we write HSIC as a function of $\mathbf{w}$, HSIC($\mathbf{w}$)). To obtain a sparse solution of the selected features, the zero "norm" $\| \mathbf{w} \|_0$ is also incorporated into our objective function (clearly $\|.\|_0$ is not a proper norm). $\| \mathbf{w} \|_0$ computes the number of non-zero entries in $\mathbf{w}$ and the sparsity is achieved by imposing heavier penalty on solutions with large number of non-zero entries. In summary, feature selection using HSIC can be formulated as:

$$\mathbf{w} = \arg\max_{\mathbf{w}} \ \mathrm{HSIC}(\mathbf{w}) - \lambda \|\mathbf{w}\|_0 \ \text{ where } \ \mathbf{w} \in [0, \infty)^n \tag{57}$$

The zero "norm" is not a continuous function. However, it can be approximated well by a concave function (Fung et al., 2002) ($\alpha = 5$ works well in practice):

$$\|\mathbf{w}\|_0 \approx \mathbf{1}^\top (\mathbf{1} - \exp -\alpha \, \mathbf{w}) \tag{58}$$

While the optimization problem in (57) is non-convex, we may use relatively more efficient optimization procedures for the concave approximation of the $\ell_0$ norm. For instance, we may use the convex-concave procedure (CCCP) of Yuille and Rangarajan (2003). For a Gaussian kernel HSIC can be decomposed into the sum of a convex and a concave function:

$$\mathrm{HSIC}(\mathbf{w}) - \lambda \| \mathbf{w} \|_0 \approx \mathrm{tr}(\mathbf{K}(\mathbf{I} - m^{-1} \mathbf{1}\mathbf{1}^\top) \mathbf{L} (\mathbf{I} - m^{-1} \mathbf{1}\mathbf{1}^\top)) - \lambda \mathbf{1}^\top (\mathbf{1} - e^{-\alpha \mathbf{w}}) \tag{59}$$

Depending on the choice of $\mathbf{L}$ we need to assign all terms involving exp with positive coefficients into the convex and all terms involving negative coefficients to the concave function.

10. Code is available as part of the Elefant package at http://elefant.developer.nicta.com.au.

## References

A. Alizadeh, M. Eisen, R. Davis, et al. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci.*, 96:6745–6750, 1999.

S. Amari and S. Wu. An information-geometrical method for improving performance of support vector machine classifiers. In D. Willshaw and A. Murray, editors, *Proceedings of ICANN'99*, volume 1, pages 85–90. IEE Press, 1999.

N. Anderson, P. Hall, and D. Titterington. Two-sample test statistics for measuring discrepancies between two multivariate probability density functions using kernel-based density estimates. *Journal of Multivariate Analysis*, 50:41–54, 1994.

C. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

J. Bedo, C. Sanderson, and A. Kowalczyk. An efficient alternative to svm based recursive feature elimination with applications in natural language processing and bioinformatics. In *Artificial Intelligence*, 2006. to appear.

D. G. Beer, S. L. Kardia, S. L. Huang, et al. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nat. Med.*, 8:816–824, 2002.

A. Berchuck, E. Iversen, and J. Lancaster et al. Patterns of gene expression that characterize long-term survival in advanced stage serous ovarian cancers. *Clin. Cancer Res.*, 11:3686–3696, 2005.

A. Bhattacharjee, W. G. Richards, W. G. Staunton, et al. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proc. Natl. Acad. Sci.*, 98:13790–13795, 2001.

K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics (ISMB)*, 22(14):e49–e57, 2006.

P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Proc. Intl. Conf. Machine Learning*, pages 82–90, San Francisco, California, 1998. Morgan Kaufmann Publishers. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-03.ps.Z.

M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. Furey, M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc. Natl. Acad. Sci.*, 97:262–267, 2000.

L. Bullinger, K. Dohner, E. Bair, S. Frohling, R. F. Schlenk, R. Tibshirani, H. Dohner, and J. R. Pollack. Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia. *New England Journal of Medicine*, 350(16):1605–1616, Apr 2004.

M. Collins and N. Duffy. Convolution kernels for natural language. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 625–632, Cambridge, MA, 2001. MIT Press.

N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On optimizing kernel alignment. Technical report, UC Davis Department of Statistics, 2003.

S. M. Dhanasekaran, T. R. Barrette, D. Ghosh, R. Shah, S. Varambally, K. Kurachi, K. J. Pienta, M. A. Rubin, and A. M. Chinnaiyan. Delineation of prognostic biomarkers in prostate cancer. *Nature*, 412(6849):822–826, Aug 2001.

G. Dornhege, B. Blankertz, G. Curio, and K. Müller. Boosting bit rates in non-invasive EEG single-trial classifications by feature combination and multi-class paradigms. *IEEE Trans. Biomed. Eng.*, 51:993–1002, 2004.

G. Dornhege, B. Blankertz, M. Krauledat, F. Losch, G. Curio, and K. Müller. Optimizing spatio-temporal filters for improving BCI. In *Advances in Neural Information Processing Systems 18*, 2006.

L. Ein-Dor, O. Zuk, and E. Domany. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proc. Natl. Acad. Sci. USA*, 103(15): 5923–5928, Apr 2006.

Andrey Feuerverger. A consistent test for bivariate dependence. *International Statistical Review*, 61(3):419–433, 1993.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representation. Technical report, IBM Watson Research Center, New York, 2000.

K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

G. Fung, O. L. Mangasarian, and A. J. Smola. Minimal kernel classifiers. *Journal of Machine Learning Research*, 3:303–321, 2002.

T. Gärtner, P.A. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In B. Schölkopf and M. K. Warmuth, editors, *Proc. Annual Conf. Computational Learning Theory*, pages 129–143. Springer, 2003.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, Oct 1999.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two sample problem. Technical Report 157, MPI for Biological Cybernetics, 2007.

A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Proc. Intl. Conf. on Algorithmic Learning Theory*, pages 63–78, 2005.

S. Gruvberger, M. Ringner, Y. Chen, S. Panavally, L. H. Saal, A. Borg, M. Ferno, C. Peterson, and P. S. Meltzer. Estrogen receptor status in breast cancer is associated with remarkably distinct gene expression patterns. *Cancer Res*, 61(16):5979–5984, Aug 2001.

C. Guestrin, A. Krause, and A. Singh. Near-optimal sensor placements in gaussian processes. In *International Conference on Machine Learning ICML'05*, 2005.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

Wassily Hoeffding. A class of statistics with asymptotically normal distribution. *The Annals of Mathematical Statistics*, 19(3):293–325, 1948.

N. Iizuka, M. Oka, H. Yamada-Okabe, et al. Oligonucleotide microarray for prediction of early intrahepatic recurrence of hepatocellular carcinoma after curative resection. *Lancet*, 361:923–929, 2003.

K. Kira and L. Rendell. A practical approach to feature selection. In *Proc. 9th Intl. Workshop on Machine Learning*, pages 249–256, 1992.

S. Lemm, B. Blankertz, G. Curio, and K.-R. Mülller. Spatio-spectral filters for improving the classification of single trial EEG. *IEEE Trans. Biomed. Eng.*, 52:1541–1548, 2005.

C. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for SVM protein classification. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, volume 15, Cambridge, MA, 2002. MIT Press.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, February 2002.

Ingrid Lönnstedt and Terry Speed. Replicated microarray data. *Statistica Sinica*, 12:31–46, 2002.

Radford M. Neal. Assessing relevance determination methods using delve. In *Neural Networks and Machine Learning*, pages 97–129. Springer, 1998.

I Nemenman, F Shafee, and W Bialek. Entropy and inference, revisited. In *Neural Information Processing Systems*, volume 14, Cambridge, MA, 2002. MIT Press.

J. Neumann, C. Schnörr, and G. Steidl. Combined SVM-based feature selection and classification. *Machine Learning*, 61:129–150, 2005.

A. Rosenwald, G. Wright, G. Chan, et al. The use of molecular profiling to predict survival after chemotherapy for diffuse large-b-cell lymphoma. *N. Engl. J. Med.*, 346:1937–1947, 2002.

B. Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, Munich, 1997. Download: http://www.kernel-machines.org.

B. Schölkopf, P. L. Bartlett, A. J. Smola, and R. C. Williamson. Shrinking the tube: a new support vector regression algorithm. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 330–336, Cambridge, MA, 1999. MIT Press.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA, 2004.

R. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, New York, 1980.

G.K. Smyth. Linear models and empirical bayes methods for assessing differential expressionin microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3, 2004.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *J. Mach. Learn. Res.*, 2:67–93, 2002.

R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. In *National Academy of Sciences*, volume 99, pages 6567–6572, 2002.

R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Class prediction by nearest shrunken centroids, with applicaitons to dna microarrays. *Stat Sci*, 18:104–117, 2003.

Kari Torkkola. Feature extraction by non-parametric mutual information maximization. *J. Mach. Learn. Res.*, 3:1415–1438, 2003.

P. J. Valk, R. G. Verhaak, M. A. Beijen, C. A. Erpelinck, S. Barjesteh van Waalwijk van Doorn-Khosrovani, J. M. Boer, H. B. Beverloo, M. J. Moorhouse, P. J. van der Spek, B. Lowenberg, and R. Delwel. Prognostically useful gene-expression profiles in acute myeloid leukemia. *New England Journal of Medicine*, 350(16):1617–1628, Apr 2004.

M. J. van de Vijver, Y. D. He, L. J. van 't Veer, et al. A gene-expression signature as a predictor of survival in breast cancer. *N. Engl. J. Med.*, 247:1999–2009, 2002.

L. J. van't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. M. Hart, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, 2002.

S. V. N. Vishwanathan and A. J. Smola. Fast kernels for string and tree matching. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 569–576. MIT Press, Cambridge, MA, 2003.

S. V. N. Vishwanathan, A. J. Smola, and R. Vidal. Binet-Cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes. *International Journal of Computer Vision*, 73(1):95–119, 2007.

Y. Wang, J. G. Klijn, Y. Zhang, et al. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365:671–679, 2005.

P. Warnat, R. Eils, and B. Brors. Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes. *BMC Bioinformatics*, 6:265, Nov 2005.

J. B. Welsh, L. M. Sapinoso, A. I. Su, S. G. Kern, J. Wang-Rodriguez, C. A. Moskaluk, J. r. Frierson HF, and G. M. Hampton. Analysis of gene expression identifies candidate markers and pharmacological targets in prostate cancer. *Cancer Res*, 61(16):5974–5978, Aug 2001.

M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H Zuzan, J.A. Olson Jr, J.R.Marks, and J.R.Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *PNAS*, 98(20), 2001.

J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems 13*, pages 668–674, 2000.

A.L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15: 915–936, 2003.

M. Zaffalon and M. Hutter. Robust feature selection using distributions of mutual information. In A. Darwiche and N. Friedman, editors, *Proceedings of the 18th International Conference on Uncertainty in Artificial Intelligence (UAI-2002)*, pages 577–584, San Francisco, CA., 2002. Morgan Kaufmann.

**Acknowledgments** We thank Vishy Vishwanathan and Bernhard Schölkopf for helpful discussions. NICTA is funded through the Australian Government's *Baking Australia's Ability* initiative, in part through the Australian Research Council.