IEICE Transactions on Information and Systems,
vol.E96-D, no.7, pp.1513–1524, 2013.

Feature Selection via ℓ_1-Penalized Squared-Loss Mutual Information

Wittawat Jitkrittum, Hirotaka Hachiya, and Masashi Sugiyama
Tokyo Institute of Technology, Japan.
sugi@cs.titech.ac.jp
http://sugiyama-www.cs.titech.ac.jp/~sugi

Abstract

Feature selection is a technique to screen out less important features. Many existing supervised feature selection algorithms use redundancy and relevancy as the main criteria to select features. However, *feature interaction*, potentially a key characteristic in real-world problems, has not received much attention. As an attempt to take feature interaction into account, we propose ℓ_1-LSMI, an ℓ_1-regularization based algorithm that maximizes a squared-loss variant of mutual information between selected features and outputs. Numerical results show that ℓ_1-LSMI performs well in handling redundancy, detecting non-linear dependency, and considering feature interaction.

Keywords

feature selection, ℓ_1-regularization, squared-loss mutual information, density-ratio estimation, dimensionality reduction

1 Introduction

Recently, solving real-world complex problems with supervised-learning techniques has become more and more common. In supervised learning, using all variables as input to a learning algorithm works well when the number of variables is limited. However, when the number of variables is large (e.g., gene expression-based patient classification), using all variables in the learning process could lead to overfitting and a model-interpretability problem [38].

To overcome these problems, feature-selection techniques are useful. Feature selection aims at removing unnecessary variables and retaining only relevant variables for the target supervised-learning task. Many previous studies [25, 33] showed that feature selection is useful in finding relevant variables to gain more insight into the data. Moreover, the generalization ability of the learned model can be improved through the removal of noisy variables [23, 17].

Two conflicting criteria which are commonly used to select features are *relevancy* and *redundancy*. Features are relevant if they can explain outputs. Features are redundant if they are similar. It is trivial that more features are more likely to explain outputs well. However, more features are also more prone to be redundant [23, 38].

Feature interaction is another important criterion to consider. Feature interaction is a situation in which two or more weak features can explain the output well in the context of each other, even though each of them alone may not be explanatory. It is one of the key characteristics in real-world problems. To detect a group of interacting features, it is necessary to simultaneously consider all features. This is because, by definition, considering features individually will not reveal any relevancy to the output. Due to this difficulty, feature interaction has not received much attention from the community.

In this research, instead of focusing on only the relevancy and the redundancy as many previous studies did, we also take into consideration the interaction among features. We propose ℓ_1-LSMI, an ℓ_1-regularization based algorithm that maximizes a squared-loss variant of mutual information between selected features and outputs. We also experimentally compare the proposed method with several state-of-the-art feature selection algorithms on both artificial and real data. Numerical results show that ℓ_1-LSMI performs well in handling redundancy, detecting non-linear dependency, and considering feature interaction.

The structure of this paper is as follows. We formulate our feature-selection problem in Sect. 2. Then we describe optimization strategies commonly used in practice in Sect. 3, as well as several feature quality measures in Sect. 4. We argue that, among the listed strategies, ℓ_1-regularization based feature weighting is the best choice if we take into account the balance between computational load and the quality of features. As a feature quality measure, we show that squared-loss mutual information (SMI) [33] possesses various desirable properties. Based on this argument, in Sect. 5, we propose to combine ℓ_1-regularization and SMI, which we refer to as ℓ_1-LSMI. Experiments on artificial and real data are described in Sect. 6. Finally, we conclude the paper in Sect. 7.

2 Problem Formulation

A formal description of a supervised feature-selection problem is as follows. Assume we have an input data matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ and output data vector $\mathbf{Y} \in \mathbb{R}^n$, where m is the number of features and n is the sample size. $\mathbf{X}$ and $\mathbf{Y}$ are realizations of the random variable $X = (X_1, \ldots, X_m)$ and Y, respectively. Given the desired number of features k, supervised feature selection attempts to find a subset of features identified by the set of feature indices $\mathcal{I} \subset \{1, \ldots, m\}$, such that the underlying *feature quality measure* f is maximized. Formally, this can be formulated as an optimization problem as

$$\begin{aligned} &\underset{\mathcal{I}\subset\{1,\ldots,m\}}{\text{maximize}} && f(\mathbf{X}_{\mathcal{I}}, \mathbf{Y}) \\ &\text{subject to} && |\mathcal{I}| = k, \end{aligned} \tag{1}$$

where $|\cdot|$ denotes the set cardinality, and $\mathbf{X}_{\mathcal{I}}$ denotes the data matrix $\mathbf{X}$ retaining only rows indexed by $\mathcal{I}$.

In general, f can be any function which can quantify the desired characteristics of the selected features. A popular choice for f is the classification accuracy of a chosen classifier [14]. While the selected features $\widehat{\mathcal{I}}$ obtained from this approach can yield a good classification accuracy, they are only specifically fit to the predictor in use. As a result, an objective interpretation of $\widehat{\mathcal{I}}$ may be difficult [9]. In this work, we opt to focus on feature selection algorithms which are independent of a predictor for wide applicability.

In practice, searching for a good feature subset to maximize f in a reasonable amount of time can be challenging. In fact, finding the global optimal feature subset is known to be NP-hard [36, 22]. One way to guarantee that we can obtain the global optimal subset is to perform an exhaustive search over all possible subsets. However, since there are 2^m possible subsets in total, this approach is impractical for large m. Clearly, a good *optimization strategy* is needed to efficiently explore the subset space.

As shown above, optimization strategies and feature quality measures are two important research issues in feature selection. We describe standard optimization strategies in Sect. 3, and popular feature quality measures in Sect. 4.

3 Optimization Strategies

The optimization strategy defines how to search for a good feature subset. The complexity of these optimization strategies range, with respect to the number of features m, from linear (feature ranking) to exponential (exhaustive search). Optimization strategies in general attempt to find features which have high relevancy to the output. Higher complexity in some strategies follows from the fact that feature redundancy is also taken into consideration. We start the discussion with fast feature ranking technique which does not consider feature redundancy.

3.1 Feature Ranking

Feature ranking is one of the simplest feature optimization strategies. Given m features $\{X_1, \ldots, X_m\}$, the feature ranking approach solves the optimization problem of the form

$$\underset{\mathcal{I} \subset \{1,\ldots,m\}}{\text{maximize}} \quad \sum_{i \in \mathcal{I}} f(X_i, Y) \quad \text{subject to} \quad |\mathcal{I}| = k.$$

To solve this problem, we calculate $f(X_i, Y)$ for $i \in \{1, \ldots, m\}$, rank X_i in the descending order, and then select the top k features. The notable feature selection algorithms based on this ranking scheme are Pearson correlation ranking, SPEC [37], the Laplacian score [12], and the mutual information score [33].

Although simple and fast, feature ranking considers only the relevancy of features. Evaluating each feature individually does not take into account the redundancy among features. Specifically, if there are many relevant features which are similar in nature, all of them will be ranked top. This is not desirable since having many similar features is usually as good as having just one. In other words, k best features are not the best k features [23].

3.2 Sequential Search

To take feature redundancy into account, the popular sequential search [14, 28] can be used. It comes with two variants: forward and backward search. Forward search works iteratively by maintaining the currently selected features $\mathcal{X}_t$. At each step t, $\mathcal{X}_t$ is updated with

$$\mathcal{X}_t \leftarrow \mathcal{X}_{t-1} \cup \{X_t^*\},$$

where $X_t^* = \mathrm{argmax}_X\, f(\mathcal{X}_{t-1} \cup \{X\})$ and $\mathcal{X}_0 = \emptyset$. The backward search works similarly except that $\mathcal{X}_0$ contains the full feature set. At each step, a feature which reduces f the least is removed.

A potential drawback of the sequential search is its greedy search nature which is independent of k. That is, the search paths are nested for different values of k. Specifically, it is decremental for the backward search, and incremental for the forward search. The result is that, for the backward search, once a feature is removed, it will never be considered again. Likewise, for the forward search, once a feature is added, it will never be removed even if it is found to be redundant at latter iterations.

3.3 Feature Weighting

Feature weighting [34, 39, 19, 21] is an approach which can search for features with a continuous optimization. Formally, the feature weighting approach attempts to find a feature weight vector $\widehat{\boldsymbol{w}} \in \mathbb{R}^m$ which is the solution of the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{maximize}} \quad & f(\mathrm{diag}(\boldsymbol{w})\mathbf{X}, \mathbf{Y}) \\ \text{subject to} \quad & \|\boldsymbol{w}\|_1 \le r, \end{aligned} \tag{2}$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm, $\mathrm{diag}(\boldsymbol{w})$ is a diagonal matrix with $\boldsymbol{w}$ placed along its diagonal, and $r > 0$ is the tuning parameter for the radius of the ℓ_1-ball. It is known that if r is sufficiently small, then the solution tends to be on a vertex of the ℓ_1 simplex, which makes $\widehat{\boldsymbol{w}}$ sparse [34]. Features can then be selected according to the non-zero coefficients of the solution $\widehat{\boldsymbol{w}}$. In fact, observations reveal that the number of non-zero coefficients tends to increase as r increases. So, a simple bisection method may be used to search for the value of r which gives k features.

Unlike the sequential search, the feature weighting approach incorporates k into the problem through r from the beginning. So, the solutions for different values of k are not necessarily nested. This characteristic is particularly useful when there are multiple optimal feature subsets of different sizes which are disjoint.

4 Feature Quality Measures

In this section, we describe a number of feature quality measures commonly used in practice. A feature quality measure is a criterion which indicates how good the selected features are, and is the counterpart of the optimization strategy. Here, we focus on predictor-independent criteria.

4.1 Pearson Correlation

Pearson correlation (PC) is a well-known univariate statistical quantity which can be used to measure a linear dependency between two random variables X and Y. It is defined as

$$\rho(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma(X)\sigma(Y)}, \tag{3}$$

where $\mathrm{cov}(X,Y)$ denotes the covariance between X and Y, and $\sigma(X)$ and $\sigma(Y)$ are population standard deviation of X and Y, respectively.

Although the independence of X and Y implies $\rho = 0$, the converse is not necessarily true since the correlation is capable of detecting only a linear dependency. An example would be a quadratic dependence $Y = X^2$, which gives $\rho = 0$ due to the cancellation of the negatively and the positively correlated components.

For a feature selection purpose, $|\rho|$ can be used to rank features. There are many feature selection algorithms based on Pearson correlation [24, 11, 23].

4.2 Hilbert-Schmidt Independence Criterion

The Hilbert-Schmidt independence criterion (HSIC) [8] is a multivariate dependence measure which can detect a non-linear dependency, and does not require a density estimation.

The formal definition of HSIC is given as follows. Let $\mathcal{D}_X$ and $\mathcal{D}_Y$ be the domains of X and Y. Define a mapping $\phi(\boldsymbol{x}) \in \mathcal{F}$ from all $\boldsymbol{x} \in \mathcal{D}_X$ to the feature space $\mathcal{F}$ in such a way that the inner product of points in $\mathcal{F}$ is given by a kernel function $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$. This can be achieved if $\mathcal{F}$ is a reproducing kernel Hilbert space on $\mathcal{D}_X$ [2]. Similarly, define another reproducing kernel Hilbert space. $\mathcal{G}$ for $\mathcal{D}_Y$ with feature map ψ and kernel $l(\boldsymbol{y}, \boldsymbol{y}') = \langle \psi(\boldsymbol{y}), \psi(\boldsymbol{y}') \rangle$. Then, the cross-covariance operator [7] associated with the joint probability p_{xy} is a linear operator C_{XY} defined as

$$C_{XY} := \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}[(\phi(\boldsymbol{x}) - \mu_{\boldsymbol{x}}) \otimes (\psi(\boldsymbol{y}) - \mu_{\boldsymbol{y}})],$$

where $\otimes$ is the tensor product. HSIC is defined as the squared Hilbert-Schmidt norm of the cross-covariance operator

$$\mathrm{HSIC}(p_{xy}, \mathcal{F}, \mathcal{G}) := \|C_{XY}\|^2_{\mathrm{HS}},$$

which could be expressed in terms of kernels [8] as

$$\begin{aligned}\mathrm{HSIC}(p_{xy}, \mathcal{F}, \mathcal{G}) =& \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}',\boldsymbol{y},\boldsymbol{y}'}[k(\boldsymbol{x},\boldsymbol{x}')l(\boldsymbol{y},\boldsymbol{y}')] \\ &+ \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'}[k(\boldsymbol{x},\boldsymbol{x}')]\mathbb{E}_{\boldsymbol{y},\boldsymbol{y}'}[l(\boldsymbol{y},\boldsymbol{y}')] \\ &- 2\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}[\mathbb{E}_{\boldsymbol{x}'}[k(\boldsymbol{x},\boldsymbol{x}')]\mathbb{E}_{\boldsymbol{y}'}[l(\boldsymbol{y},\boldsymbol{y}')]].\end{aligned}$$

$\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}',\boldsymbol{y},\boldsymbol{y}'}[k(\boldsymbol{x},\boldsymbol{x}')l(\boldsymbol{y},\boldsymbol{y}')]$ is the expectation over independent pairs $(\boldsymbol{x}, \boldsymbol{y})$ and $(\boldsymbol{x}', \boldsymbol{y}')$ drawn from p_{xy}. Given an i.i.d. paired sample $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, an empirical estimator of HSIC is given by

$$\mathrm{HSIC}(\mathcal{S}, \mathcal{F}, \mathcal{G}) = \frac{1}{(n-1)^2}\mathrm{tr}(KHLH), \tag{4}$$

where $K, L, H \in \mathbb{R}^{n \times n}$, $(K)_{i,j} := k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $(L)_{i,j} := l(\boldsymbol{y}_i, \boldsymbol{y}_j)$, and $H := I_n - \mathbf{1}\mathbf{1}^T/n$ (centering matrix). It was also shown that, if k and l are universal kernels (e.g., Gaussian kernels) [29], then $\mathrm{HSIC}(p_{xy}, \mathcal{F}, \mathcal{G}) = 0$ if and only if X and Y are independent. So, HSIC can also be used as a dependence measure.

In spite of the strong theoretical properties of HSIC, there is no known objective criterion for model selection of the kernel functions k and l. A popular heuristic choice is to use a Gaussian kernel with its width set to the median of the pairwise distance of the data points [27].

4.3 Mutual Information

In information theory, mutual information [4] is an important quantity which can be used to detect a general non-linear dependency between two random variables. It has been widely used as the criterion for feature selection [23, 31, 24] as well as feature extraction [35]. Mutual information is defined as

$$I(X, Y) := \iint \log\left(\frac{p_{xy}(\boldsymbol{x}, \boldsymbol{y})}{p_x(\boldsymbol{x})p_y(\boldsymbol{y})}\right) p_{xy}(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{y}, \tag{5}$$

which is the Kullback-Leibler divergence [16] from $p_{xy}(\boldsymbol{x}, \boldsymbol{y})$ to $p_x(\boldsymbol{x})p_y(\boldsymbol{y})$. Mutual information is a measure of dependence in the sense that it is always non-negative, symmetric ($I(X, Y) = I(Y, X)$), and vanishes if and only if X and Y are independent, i.e., $p_{xy}(\boldsymbol{x}, \boldsymbol{y}) = p_x(\boldsymbol{x})p_y(\boldsymbol{y})$.

Even though mutual information is a powerful multivariate measure, accurate estimation of the densities p_{xy}, p_x and p_y is difficult in high-dimensional case. A recent approach which avoids taking the ratio of estimated densities by directly modeling the density ratio $\frac{p_{xy}(\boldsymbol{x}, \boldsymbol{y})}{p_x(\boldsymbol{x})p_y(\boldsymbol{y})}$ is Maximum Likelihood Mutual Information (MLMI) [31]. Although MLMI was demonstrated to be accurate, its estimation is computationally rather expensive due to the existence of the logarithm function.

4.4 Squared-loss Mutual Information

Another mutual information variant which has received much attention recently is Squared-loss Mutual Information (SMI) [33, 30, 10, 32] defined as

$$I_s(X, Y) := \frac{1}{2} \iint \left(\frac{p_{xy}(\boldsymbol{x}, \boldsymbol{y})}{p_x(\boldsymbol{x})p_y(\boldsymbol{y})} - 1\right)^2 p_x(\boldsymbol{x})p_y(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{y}. \tag{6}$$

SMI is based on the f-divergence [1, 5] with a squared loss (also known as the Pearson divergence, [20]), as opposed to the ordinary mutual information which is based on the f-divergence with a log loss (Kullback-Leibler divergence, [16]). Note that $I_s(X, Y) = I_s(Y, X)$, $I_s(X, Y) \geq 0$, and $I_s(X, Y) = 0$ if and only if $p_{xy}(\boldsymbol{x}, \boldsymbol{y}) = p_x(\boldsymbol{x})p_y(\boldsymbol{y})$, just like the ordinary mutual information. Therefore, SMI can also be used as a measure of dependence between X and Y.

SMI can be estimated by directly modeling the ratio $g^*(\boldsymbol{x},\boldsymbol{y}) = \frac{p_{xy}(\boldsymbol{x},\boldsymbol{y})}{p_x(\boldsymbol{x})p_y(\boldsymbol{y})}$ itself without going through the estimation of the densities. The goal is to find a density ratio estimate $\widehat{g}(\boldsymbol{x},\boldsymbol{y})$ which is as close to the true density ratio $g^*(\boldsymbol{x},\boldsymbol{y})$ as possible. Here, the estimation can be formulated as a least-squares problem. That is, to find $\widehat{g}(\boldsymbol{x},\boldsymbol{y})$ such that its expected squared difference from $g^*(\boldsymbol{x},\boldsymbol{y})$ is minimized:

$$\min_{g} \frac{1}{2} \iint \left(g(\boldsymbol{x},\boldsymbol{y}) - g^*(\boldsymbol{x},\boldsymbol{y})\right)^2 p_x(\boldsymbol{x})p_y(\boldsymbol{y})\, \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{y}. \tag{7}$$

Since finding g over all measurable functions is not tractable [30], the model g is restricted to be in a linear subspace $\mathcal{G}$ defined as

$$\mathcal{G} := \{\boldsymbol{\alpha}^T\boldsymbol{\varphi}(\boldsymbol{x},\boldsymbol{y}) \,|\, \boldsymbol{\alpha} = (\alpha_1,\ldots,\alpha_b)^T \in \mathbb{R}^b\},$$

where $\boldsymbol{\alpha}$ is the model parameter to be learned, and $\boldsymbol{\varphi}(\boldsymbol{x},\boldsymbol{y}) = (\varphi_1(\boldsymbol{x},\boldsymbol{y}),\ldots,\varphi_b(\boldsymbol{x},\boldsymbol{y}))^T$ is a basis function vector such that $\forall l, \varphi_l(\boldsymbol{x},\boldsymbol{y}) \geq 0$. The basis also admits kernel functions which depend on samples.

With $\mathcal{G}$, finding $\widehat{g}$ amounts to finding the optimal $\boldsymbol{\alpha}$. By using an empirical approximation, Eq. (7) can be written as

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^b} \frac{1}{2}\boldsymbol{\alpha}^T\widehat{\boldsymbol{H}}\boldsymbol{\alpha} - \widehat{\boldsymbol{h}}^T\boldsymbol{\alpha} + \frac{\lambda}{2}\boldsymbol{\alpha}^T\boldsymbol{\alpha}, \tag{8}$$

where the term $\frac{\lambda}{2}\boldsymbol{\alpha}^T\boldsymbol{\alpha}$ with a regularization parameter $\lambda > 0$ is included for a regularization purpose, and

$$\widehat{\boldsymbol{H}} := \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\boldsymbol{\varphi}(\boldsymbol{x}_i,\boldsymbol{y}_j)\boldsymbol{\varphi}(\boldsymbol{x}_i,\boldsymbol{y}_j)^T,$$
$$\widehat{\boldsymbol{h}} := \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\varphi}(\boldsymbol{x}_i,\boldsymbol{y}_i).$$

By differentiating Eq. (8) with respect to $\boldsymbol{\alpha}$ and equating it to zero, the solution $\widehat{\boldsymbol{\alpha}}$ can be computed analytically as

$$\widehat{\boldsymbol{\alpha}} = \left(\widehat{\boldsymbol{H}} + \lambda\boldsymbol{I}\right)^{-1}\widehat{\boldsymbol{h}},$$

where $\boldsymbol{I}$ denotes the identity matrix. Finally, using $\widehat{\boldsymbol{\alpha}}$, SMI in Eq. (6) can be estimated as

$$\widehat{I_s} = \frac{1}{2}\widehat{\boldsymbol{h}}^T\widehat{\boldsymbol{\alpha}} - \frac{1}{2}. \tag{9}$$

The estimator in Eq. (9) is called Least-Squares Mutual Information (LSMI).

LSMI possesses many good properties [30]. For example, it has an optimal convergence rate in n under non-parametric setup. Also, LSMI is equipped with a model selection criterion for determining $\boldsymbol{\varphi}$ and λ. Model selection by K-fold cross validation is described as follows. First, randomly split samples $\{(\boldsymbol{x}_i,\boldsymbol{y}_i)\}_{i=1}^n$ into (roughly) equal K disjoint subsets $\{\mathcal{S}_k\}_{k=1}^K$. An estimator $\widehat{\boldsymbol{\alpha}}_{\mathcal{S}_{-k}}$ is then obtained using $\mathcal{S}_{-k} := \{\mathcal{S}_j\}_{j \neq k}$. Finally, the approximation error for the held-out samples $\mathcal{S}_k$ is computed. The procedure is repeated K times, and $(\boldsymbol{\varphi}, \lambda)$ which minimizes the mean $\widehat{J}^{(K-CV)}$ is chosen:

$$\widehat{J}^{(K-CV)} := \frac{1}{K} \sum_{k=1}^{K} \left(\frac{1}{2} \widehat{\boldsymbol{\alpha}}_{\mathcal{S}_{-k}}^T \widehat{\boldsymbol{H}}_{\mathcal{S}_k} \widehat{\boldsymbol{\alpha}}_{\mathcal{S}_{-k}} - \widehat{\boldsymbol{h}}_{\mathcal{S}_k}^T \widehat{\boldsymbol{\alpha}}_{\mathcal{S}_{-k}} \right).$$

5 Proposed Method

In this section, we describe our proposed method.

5.1 Motivations

As mentioned previously, there are a number of factors which cause the difficulty of feature selection, i.e., non-linear dependency, feature interaction, and feature redundancy. Although existing combinations of optimization strategies and measures can handle these problems, the trade-off of the computational complexity and the obtained abilities to deal with such issues is not well balanced.

A summary of properties of common optimization strategies is shown in Table 1. Ranking is very fast since it completely disregards feature redundancy and feature interaction, and focuses on only feature relevancy. Forward search improves this by maintaining a set of selected features, and greedily adding each feature to the set. This allows the forward search to deal with feature redundancy by not adding a redundant feature to the set. Nevertheless, feature interaction cannot be detected since features are not considered in the presence of each other. This is why backward search comes to play by starting from the full feature set and iteratively removing a feature instead. Although this scheme has a potential to detect interacting features, the complexity goes from $O(m)$ to $O(m^2)$ which could be problematic when the number of features, m, is large. Considering all strategies, an ℓ_1-based approach seems to be the optimal choice here. It offers a continuous optimization which is usually easier than a discrete optimization. Also, since all features are considered simultaneously by optimizing their weights, it can take into account feature redundancy and feature interaction.

A summary of properties of feature quality measures is shown in Table 2. PC is very efficient to compute. However, only linear dependency can be identified. HSIC can reveal a non-linear dependency. Nonetheless, it is unclear how to objectively choose the right kernel function. MI is another measure that is capable of detecting a nonlinear dependency but the existence of log causes computational inefficiency. It can be seen that SMI has balanced properties here. Not only is it able to capture a non-linear dependency, using a squared loss instead of a log loss also permits its estimator to have an analytic form, which can be efficiently computed.

Table 3 shows the combinations of optimization strategies and feature quality measures. Many of them have already been proposed in the past. Exhaustive search is marked

Table 1: Summary of properties of optimization strategies. “disc.” and “cont.” denote “discrete” and “continuous”, respectively.

	Ranking	Forward	Backward	Exhaustive	ℓ_1
Optimization	disc.	disc.	disc.	disc.	cont.
Complexity	m	m	m^2	2^m	m
Redundancy	×	△	◯	◎	◯
Interaction	×	×	◯	◎	◯

×: Not considered, △: Weak, ◯: Good, ◎: Excellent

Table 2: Summary of properties of feature quality measures.

	PC	HSIC	MI	SMI
Non-linear Dependency	×	◯	◯	◯
Model Selection	not needed	×	◯	◯
Computational Efficiency	◎	◯	×	△

×: Not available/Poor, △: Weak, ◯: Good, ◎: Excellent

Table 3: Summary of combinations of optimization strategies and feature quality measures.

	Ranking	Forward	Backward	Exhaustive	ℓ_1
PC	◯[11]	×	×	×	×
HSIC	–	◯[28]	◯[28]	×	△[22]
MI	◯[31]	◯	◯	×	–
SMI	◯[33]	◯[33, 10]	◯[33]	×	–

◯: Method exists, △: Variation exists,
–: Method does not exist, × Method is unreasonable, impractical

impractical since it is computationally intractable. PC is a univariate measure which considers one feature at a time. Combining it with a feature-set optimization strategy (i.e., forward, backward search, ℓ_1 approach) would degenerate back to a ranking approach. Hence, the combinations are marked unreasonable.

It can be seen that the feature weighting with ℓ_1-regularization is the best among the optimization strategies. Also, SMI has the best balance among the listed feature quality measures. We therefore propose to combine ℓ_1-regularized feature weighting with SMI, which we call ℓ_1-LSMI.

5.2 Formulation of ℓ_1-LSMI

ℓ_1-LSMI attempts to find an m-dimensional sparse weight vector by solving the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{w}\in\mathbb{R}^m}{\text{maximize}} \quad & \widehat{I_s}(\mathrm{diag}(\boldsymbol{w})\mathbf{X}, \mathbf{Y}) \\ \text{subject to} \quad & \mathbf{1}^T\boldsymbol{w} \le r \\ & \boldsymbol{w} \ge \mathbf{0}, \end{aligned} \tag{10}$$

where $\widehat{I_s}$ is the LSMI defined in Eq. (9), $r > 0$ is the radius of the ℓ_1-ball, $\mathbf{1}$ is the m-dimensional vector consisting of only 1's, and "$\ge$" in $\boldsymbol{w} \ge \mathbf{0}$ is applied element-wise. Features are selected according to the non-zero coefficients of the learned $\widehat{\boldsymbol{w}}$. Here, since the sign of $\widehat{w}_j$ does not affect the feature selection process, we only consider the positive orthant in $\mathbb{R}^m$. Thus, the constraint $\boldsymbol{w} \ge \mathbf{0}$ is imposed, and $\|\boldsymbol{w}\|_1$ reduces to $\mathbf{1}^T\boldsymbol{w}$.

5.3 Advantages of ℓ_1-LSMI

Using SMI allows a detection of nonlinear dependency between X and Y. Furthermore, by combining it with the ℓ_1-regularization feature weighting scheme, feature interaction is also taken into account since all features are considered simultaneously. In general, the use of ℓ_1-regularization does not necessarily give an ability to deal with redundant features. That is, the weights of all redundant features may be all high. This drawback of ℓ_1-regularization is covered by the use of SMI. Since adding a redundant feature to the selected subset does not increase the SMI value (i.e., no new information), ℓ_1-LSMI implicitly deals with the feature redundancy issue by avoiding the inclusion of redundant features. This is achieved by simply maximizing SMI between the weighted features and the output. The use of density-ratio estimation in approximating SMI also helps avoid the density estimation problem, which is difficult when m is large.

5.4 Solving ℓ_1-LSMI

Here, we explain how we solve the ℓ_1-LSMI optimization problem.

5.4.1 Algorithm Overview

Algorithm 1 is executed to find a k-feature subset by a binary-search-liked scheme. Based on the observation that the number of obtained features tends to increase as r increases, the idea is to systematically vary r so that k features can be obtained. The ℓ_1-LSMI optimization problem is solved by iteratively performing gradient ascent and projection (constraint satisfaction) where $\boldsymbol{w}$ is initially set to a random feasible vector due to the non-convexity of the problem. Starting from a low r, if k features can be obtained from the current r, then return them. Otherwise, r is doubled (starting from `2:` in Algorithm 1) until more than k features are obtained. The value of r firstly found to give more than k features is denoted by r_h, and is assumed to be the upper bound of the value of r

Algorithm 1 Pseudo code of ℓ_1-LSMI to search for a k-feature subset.

```
Require: $k$ (desired number of features)
1: $r \leftarrow 0.1$ //$r$ is initially low
2: repeat //try to find an upper bound $r_\mathrm{h}$
3:     $r \leftarrow 2r$
4:     $\boldsymbol{w}_0 \leftarrow$ randomly initialize a feasible $\boldsymbol{w}$
5:     $\mathcal{X}_r \leftarrow$ Solve Eq. (10) with $(\boldsymbol{w}_0, r)$ //$\mathcal{X}_r$: set of features obtained using $r$
6:     if $|\mathcal{X}_r| = k$ then
7:         return $\mathcal{X}_r$
8:     end if
9: until $|\mathcal{X}_r| > k$ or time limit exceeded
10: $r_\mathrm{h} \leftarrow r$
11: $r_\mathrm{l} \leftarrow r_\mathrm{h}/2$
12: while time limit not exceeded do //find $r \in (r_\mathrm{l}, r_\mathrm{h})$ which gives $k$ features
    with a binary search
13:     $r_\mathrm{m} \leftarrow (r_\mathrm{h} + r_\mathrm{l})/2$
14:     $\boldsymbol{w}_0 \leftarrow$ randomly initialize a feasible $\boldsymbol{w}$
15:     $\mathcal{X}_{r_\mathrm{m}} \leftarrow$ Solve Eq. (10) with $(\boldsymbol{w}_0, r_\mathrm{m})$
16:     if $|\mathcal{X}_{r_\mathrm{m}}| = k$ then
17:         return $\mathcal{X}_{r_\mathrm{m}}$
18:     else if $|\mathcal{X}_{r_\mathrm{m}}| < k$ then
19:         $r_\mathrm{l} \leftarrow r_\mathrm{m}$
20:     else if $|\mathcal{X}_{r_\mathrm{m}}| > k$ then
21:         $r_\mathrm{h} \leftarrow r_\mathrm{m}$
22:     end if
23: end while
24: $\mathbb{S} \leftarrow$ list of all $\mathcal{X}$ found so far, sorted in the ascending order by $||\mathcal{X}| - k|, |\mathcal{X}| -$
    $k, -\widehat{I}_s(\mathbf{X}_\mathcal{X}, \mathbf{Y})$
25: return the first $\mathcal{X}$ in $\mathbb{S}$
```

which can give k features. The lower bound r_l is then set to $r_\mathrm{h}/2$ which gives strictly less than k features. The rest of the procedure (starting from `12:` in Algorithm 1) is to find $r \in (r_\mathrm{l}, r_\mathrm{h})$ using a binary search scheme, so that k features can be obtained. In each step of the search, Eq. (10) is solved using the middle point r_m between r_h and r_l. If k features cannot be found, r_h or r_l is updated accordingly. This halving procedure is repeated until k features are found, or the time limit is reached.

In case that a k-feature subset cannot be found, obtained feature subsets $\mathcal{X}$ are sorted in ascending order of three keys given by $||\mathcal{X}| - k|, |\mathcal{X}| - k, -\widehat{I}_s(\mathbf{X}_\mathcal{X}, \mathbf{Y})$. This means that the feature subsets whose size is closest to k are to be put towards the head of the list. With two sets whose size is equally closest to k, then prefer the smaller one (due to $|\mathcal{X}| - k$). If there are still many such subsets, bring the ones with highest $\widehat{I}_s(\mathbf{X}_\mathcal{X}, \mathbf{Y})$ to the head of the list, where $\mathbf{X}_\mathcal{X}$ denotes the data matrix $\mathbf{X}$ with only rows indexed by $\mathcal{X}$.

In the end, the feature subset $\mathcal{X}$ at the head of the list is selected.

5.4.2 Basis Function Design

Estimation of SMI requires b basis functions. Here, we choose the basis functions to be in the form of a product kernel defined as

$$\varphi_l(\mathrm{diag}(\boldsymbol{w})\boldsymbol{x}, \boldsymbol{y}) = \phi_l^x(\mathrm{diag}(\boldsymbol{w})\boldsymbol{x})\phi_l^y(\boldsymbol{y}) \text{ for } l = 1, \ldots, b. \tag{11}$$

$\phi_l^x(\cdot)$ is defined to be the Gaussian kernel,

$$\phi_l^x(\mathrm{diag}(\boldsymbol{w})\boldsymbol{x}) = \exp\left(-\frac{\|\,\mathrm{diag}(\boldsymbol{w})(\boldsymbol{x} - \boldsymbol{x}_{c(l)})\|^2}{2\sigma^2}\right).$$

$c(l) \in \{1, \ldots, n\}$ is a randomly chosen sample index without overlap. The definition of $\phi_l^y(\boldsymbol{y})$ depends on the task. For a regression task, $\phi_l^y(y)$ is also defined to be a Gaussian kernel,

$$\phi_l^y(y) = \exp\left(-\frac{(y - y_{c(l)})^2}{2\sigma^2}\right).$$

For a C-class classification task in which $Y \in \{1, \ldots, C\}$, the delta kernel is used on $\mathbf{Y}$, i.e., $\phi_l^y(y)$ takes 1 if $y = y_{c(l)}$, and 0 otherwise. Using these definitions, model selection for $(\boldsymbol{\varphi}, \lambda)$ is reduced to selecting (σ, λ).

5.4.3 Optimization

Given an initial point $\boldsymbol{w}_0$ and the radius r, the ℓ_1-LSMI optimization problem is simply solved by gradient ascent. To guarantee the feasibility, the updated $\boldsymbol{w}$ is projected onto the positive orthant of the constrained ℓ_1-ball in each iteration. The projection can be carried out by first projecting $\boldsymbol{w}$ onto the positive orthant with $\max(\boldsymbol{w}, \mathbf{0})$, where the max function is applied element-wise. This is then followed by a projection onto the ℓ_1-ball which can be carried out in $O(m)$ time [6].

In practice, there are many more sophisticated methods for solving Eq. (10), e.g., projected Newton-type methods [18, 26]. These methods generally converge super-linearly, and are faster (in terms of the convergence rate) than ordinary gradient ascent algorithms which converge linearly. However, the notion of convergence does not take into account the number of function evaluations. In general, methods with a good convergence rate rely on a large number of function evaluations per iteration, i.e., performing line search to find a good step size. In our case, function evaluation is expensive since model selection for (σ, λ) has to be performed. It turns out that using a more sophisticated solver may take more time to actually solve the problem even though the convergence rate is better. So, we decided to simply use a gradient ascent algorithm to solve the problem. Additionally, to further improve the computational efficiency, model selection is performed every five iterations, instead of every iteration. This is based on the fact that, in each iteration, $\boldsymbol{w}$ is not significantly altered. Hence, it makes sense to assume that the selected (σ, λ) from the previous iteration are approximately correct.

6 Experiments

In this section, we report experimental results.

6.1 Methods to be Compared

We compare the performance of the following feature selection algorithms:

- PC (Pearson correlation ranking).
- F-HSIC (forward search with HSIC).
- F-LSMI (forward search with LSMI) [10].
- B-HSIC (backward search with HSIC) [28].
- B-LSMI (backward search with LSMI).
- ℓ_1-HSIC (similar to ℓ_1-LSMI, but the objective function is replaced with $\mathrm{HSIC}(\mathrm{diag}(\boldsymbol{w})\mathbf{X}, \mathbf{Y}))$.
- ℓ_1-LSMI[1] (proposed method).
- mRMR (Minimum Redundancy Maximum Relevance) [23]. mRMR is one of the state-of-the-art algorithms which selects features by solving

$$\begin{aligned}\underset{\mathcal{I}\subset\{1,\ldots,m\}}{\text{maximize}} \quad & \overbrace{\frac{1}{k}\sum_{i\in\mathcal{I}} I(X_i, Y)}^{\text{relevancy measure}} - \overbrace{\frac{1}{k^2}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}} I(X_i, X_j)}^{\text{redundancy measure}}\\ \text{subject to} \quad & |\mathcal{I}| = k.\end{aligned}$$

  That is, it uses mutual information to select relevant features which are not too redundant. mRMR solves the optimization problem by greedily adding one feature at a time until k features can be obtained. This scheme is similar to a forward search algorithm.

- QPFS (Quadratic Programming Feature Selection) [24]. QPFS formulates the feature selection task as a quadratic programming problem of the form:

$$\begin{aligned}\underset{\boldsymbol{w}\in\mathbb{R}^m}{\text{minimize}} \quad & \frac{1}{2}(1-\alpha)\boldsymbol{w}^T\boldsymbol{Q}\boldsymbol{w} - \alpha\boldsymbol{f}^T\boldsymbol{w}\\ \text{subject to} \quad & \mathbf{1}^T\boldsymbol{w} = 1\\ & \boldsymbol{w} \geq \mathbf{0},\end{aligned}$$

  where $0 \leq \alpha \leq 1$ controls the trade-off between high relevancy (high α) and low redundancy of the selected features. $\boldsymbol{Q} = [q_{ij}] = |\rho(X_i, X_j)|$ is the absolute value of

[1]Implementation of ℓ_1-LSMI is freely available at `http://wittawat.com/software/l1lsmi/`

the Pearson correlation between X_i and X_j as in Eq. (3), and $\boldsymbol{f} = [f_i] = |\rho(X_i, Y)|$. In the case that Y is categorical, the correlation for categorical variable as in [11] is used. In this experiment, we use the recommended value of $\alpha = \bar{q}/(\bar{q} + \bar{f})$ where $\bar{q} = \frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m} q_{ij}$ and $\bar{f} = \frac{1}{m}\sum_{i=1}^{m} f_i$ [24]. Notice that if $\alpha = 1$, QPFS reduces to PC.

- Lasso [34]. Lasso is a well-known method of least squares which imposes an ℓ_1-norm constraint on the weight vector. Specifically, it solves the problem of the form:

$$\underset{\boldsymbol{w}\in\mathbb{R}^m}{\text{minimize}} \quad \|\mathbf{Y} - \boldsymbol{w}^T\mathbf{X}\|^2 + \lambda\|\boldsymbol{w}\|_1,$$

  where $\lambda \geq 0$ is the sparseness regularization parameter. In this experiment, λ is varied so that k features can be obtained.

- Relief [13, 15]. Relief is another state-of-the-art heuristic algorithm which scores each feature based on how it can discriminate different classes (distance-based).

6.2 Toy Data Experiment

An experiment is conducted on the following three toy datasets:

1. `and-or`
   - Binary classification (4 true / 6 distracting features).
   - $Y = (X_1 \wedge X_2) \vee (X_3 \wedge X_4)$.
   - $X_1, \ldots, X_7 \sim$ Bernoulli(0.5), where Bernoulli(p) denotes the Bernoulli distribution taking value 1 with probability p.
   - $X_8, \ldots, X_{10} = Y$ with 0.2 chance of bit flip.
   - Characteristics: Feature redundancy and weak interaction.
2. `quad`
   - Regression (2 true / 8 distracting features).
   - $Y = \frac{X_1^2 + X_2}{0.5 + (X_2 + 1.5)^2} + 0.1\epsilon$.
   - $X_1, \ldots, X_8, \epsilon \sim \mathcal{N}(0, 1)$, where $\mathcal{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2.
   - $X_9 \sim 0.5X_1 + \mathcal{U}(-1, 1)$, where $\mathcal{U}(a, b)$ is the uniform distribution on $[a, b]$.
   - $X_{10} \sim 0.5X_2 + \mathcal{U}(-1, 1)$.
   - Characteristic: Non-linear dependency.
3. `xor`
   - Binary classification (2 true / 8 distracting features).

Table 4: Averaged F-measures on the `and-or`, `quad`, and `xor` datasets.

Dataset	PC	F-HSIC	F-LSMI	B-HSIC	B-LSMI
`and-or`	0.25 (.00)	0.25 (.00)	0.57 (.22)	0.25 (.00)	0.85 (.22)
`quad`	0.57 (.20)	0.95 (.15)	**1.00 (.00)**	0.95 (.15)	**1.00 (.00)**
`xor`	0.25 (.31)	0.52 (.50)	0.53 (.50)	**1.00 (.00)**	**1.00 (.00)**

Dataset	ℓ_1-HSIC	ℓ_1-LSMI	mRMR	QPFS	Lasso	Relief
`and-or`	0.25 (.00)	**1.00 (.00)**	0.25 (.00)	0.41 (.17)	0.21 (.09)	0.55 (.15)
`quad`	0.64 (.23)	**1.00 (.00)**	**1.00 (.00)**	0.64 (.23)	0.66 (.25)	**1.00 (.00)**
`xor`	**1.00 (.00)**	**1.00 (.00)**	0.28 (.31)	0.25 (.32)	0.26 (.32)	**1.00 (.00)**

- $Y = \text{xor}(X_1, X_2)$, where $\text{xor}(X_1, X_2)$ denotes the XOR function for X_1 and X_2.
- $X_1, \ldots, X_5 \sim \text{Bernoulli}(0.5)$.
- $X_6, \ldots, X_{10} \sim \text{Bernoulli}(0.75)$.
- Characteristic: Feature interaction.

The number of features to select, k, is set to the number of true features in the respective dataset. For LSMI-based methods, Gaussian kernels are used as the basis functions and b is set to 100. Five-fold cross validation is carried out on a grid of (σ, λ) candidates for model selection. For σ, the candidates are also adaptively scaled with the median of pairwise sample distance σ_{med}, which depends on the currently selected features.

$$\sigma_{\text{med}} = \text{median}(\{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2\}_{i<j}).$$

Gaussian kernels are also used in HSIC-based methods. However, since model selection is not available for HSIC, in F-HSIC and B-HSIC, the Gaussian width is heuristically set to σ_{med} [27]. For ℓ_1-HSIC, the Gaussian width is adaptively set to the median of pairwise distance of $\text{diag}(\boldsymbol{w})\mathbf{X}$ every five iterations. Due to the non-convexity of the objective functions, ℓ_1-LSMI and ℓ_1-HSIC are restarted 20 times with randomly chosen initial points.

The experiment is repeated 50 times with $n = 400$ points sampled in each trial. For each method and each dataset, an average of the F-measure over all trials is reported. The F-measure is defined as $f = 2pr/(p + r)$, where

- $p =$ (number of correctly selected features) / (number of selected features).
- $r =$ (number of correctly selected features) / (number of correct features).

An F-measure is bounded between 0 and 1, and 1 is achieved if and only if all the true features are selected and none of the distracting features is selected. The results are shown in Table 4.

PC ranks the relevance of each feature individually without taking into account the redundancy among features. This results in a failure on the `and-or` dataset since

$X_8, \dots, X_{10}$, which are redundant, would simply be ranked top due to their similarity to Y.

The forward search variants do not work on problems with feature interaction. To detect interacting features, it is necessary that all features be considered simultaneously. For this reason, F-HSIC and F-LSMI fail in the *xor* problem.

The performance of HSIC-based methods seems to be unstable in many cases. A possible cause of the instability is from the use of an incorrect parameter: The heuristic of using σ_{med} for the Gaussian width does not always work. As an example, given a fixed data matrix $\mathbf{X}$, the more features selected, the larger σ_{med} may become. This is because the Euclidean distance is a non-decreasing function of the dimension. So, inclusion of many irrelevant features obviously unnecessarily makes σ_{med} larger. B-HSIC is subject to this weakness since it starts the search with all features.

B-LSMI performs well in detecting non-linear dependency (`quad`) and feature interaction (`xor`). However, due to its greedy nature, the redundant features in the `and-or` problem are sometimes chosen. That is, in the first few iterations, all redundant features are kept, and one of the true features is eliminated instead.

mRMR and QPFS have similar optimization strategies. That is, both of them measure the relevancy of each feature, and have a pairwise feature redundancy constraint. Regardless of the feature measure in use, considering features in a univariate way cannot reveal interacting features (by definition of feature interaction). Therefore, it is not surprising that both of them fail on the `xor` and `and-or` datasets. Nevertheless, mRMR works well on the `quad` dataset since mutual information can reveal a non-linear dependency. On the other hand, QPFS and Lasso do not perform well on the `quad` dataset since both of them use a linear measure.

Relief is one of the few feature ranking algorithms which can consider feature interaction (the *xor* dataset) because of its distance-based nature. However, it suffers the same drawback as other ranking algorithms in that no redundancy is considered. Hence, it fails on the *and-or* dataset with the same reason as PC.

The proposed ℓ_1-LSMI performs well on all datasets. This clearly shows that ℓ_1-LSMI can consider redundancy, detect non-linear dependency, and consider feature interaction. ℓ_1-based feature optimization enables a simultaneous consideration of features, which is the key in tackling the feature interaction problem. By using ℓ_1-regularization in combination with SMI which can detect a non-linear dependency, ℓ_1-LSMI can correctly choose the two true features in the `quad` problem. For the `and-or` problem, the pitfall is to choose $X_8, \dots, X_{10}$ because of their high correlation to Y. However, due to the usage of ℓ_1-regularization, ℓ_1-LSMI attempts to find the four-feature subset which maximizes LSMI in a non-greedy manner. Since $X_8, \dots, X_{10}$ contain bit-flip noise, inclusion of any of them will not deliver the maximum LSMI. In this case, the only four features which give the maximum LSMI are $X_1, \dots, X_4$, and thus preferred over any of $X_8, \dots, X_{10}$.

As an illustration of LSMI, Table 5 shows all possible 35 four-feature subsets of $\{X_1, \dots, X_4\} \cup \{X_8, \dots, X_{10}\}$ in the `and-or` problem and their corresponding LSMI values. It is evident that the correct subset $\{X_1, \dots, X_4\}$ has the highest LSMI. Inclusion of any of $X_8, \dots, X_{10}$ (and thus remove some from $\{X_1, \dots, X_4\}$) would cause a significant

Table 5: All possible 35 four-feature subsets of $\{X_1, \ldots, X_4\} \cup \{X_8, \ldots, X_{10}\}$ in the `and-or` dataset, and their corresponding values of LSMI to the output $Y = (X_1 \wedge X_2) \vee (X_3 \wedge X_4)$.

Feature indices				LSMI
1	2	3	4	**0.496**
1	2	3	8	0.365
1	2	3	9	0.381
1	2	3	10	0.357
1	2	4	8	0.376
1	2	4	9	0.384
1	2	4	10	0.372
1	2	8	9	0.346
1	2	8	10	0.330
1	2	9	10	0.336
1	3	4	8	0.382
1	3	4	9	0.376
1	3	4	10	0.392
1	3	8	9	0.325
1	3	8	10	0.330
1	3	9	10	0.333
1	4	8	9	0.342
1	4	9	10	0.341
2	3	4	8	0.367
2	3	4	9	0.382
2	3	4	10	0.390
2	3	8	9	0.341
2	3	8	10	0.312
2	3	9	10	0.322
2	4	8	9	0.340
2	4	8	10	0.328
2	4	9	10	0.328
3	4	8	9	0.356
3	4	8	10	0.349
3	4	9	10	0.353
1	8	9	10	0.330
2	8	9	10	0.334
3	8	9	10	0.303
4	8	9	10	0.335

drop of the LSMI value. In the extreme case, with all $X_8, \ldots, X_{10}$ in the selected set (shown at the bottom of the table), the LSMI score becomes considerably low. This is because each of $X_8, \ldots, X_{10}$ contains roughly the same information to explain Y. Thus, there is no gain in adding more features which share very similar information.

6.3 Real-Data Experiment

To demonstrate the practical use of the proposed ℓ_1-LSMI, we conduct experiments on real datasets without any specific domains. All the real datasets used in the experiments are summarized in Table 6. The "Task" column denotes the type of the problem (R for regression, and Cx for x-class classification problem). The datasets cover a wide range of domains including image, speech, and bioinformatics.

The experiment is repeated 20 times with $n = 400$ points sampled in each trial. In each trial, k is varied in the low range with a step size proportional to the entire dimensionality m. For classification, each selected k-feature subset is scored with the test error of a support vector classifier (SVC) with Gaussian kernels. For regression, the root mean squared error of support vector regression (SVR) with Gaussian kernels is used. The hyper-parameters of SVC and SVR are chosen with cross validation. We use the implementations of SVC and SVR given in the LIBSVM library [3][2]. The results are

[2]LIBSVM: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Table 6: Summary of the real datasets used in the experiments.

Dataset	m	n	Task	Class balance (%)
abalone	8	4177	R	-
bcancer	9	277	C2	70.8/29.2
cpuact	21	3000	R	-
ctslices	379	53500	R	-
flaresolar	9	1066	C2	44.7/55.3
german	20	1000	C2	70.0/30.0
glass	9	214	C6	32.7/35.5/7.9/6.1/4.2/13.6
housing	13	506	R	-
image	18	1155	C2	42.9/57.1
ionosphere	33	351	C2	64.1/35.9
isolet	617	6238	C26	about 3.85% per class
msd	90	10000	R	-
musk1	166	476	C2	56.5/43.5
musk2	166	6598	C2	84.6/15.4
satimage	36	6435	C6	23.8/10.9/21.1/9.7/11.0/23.4
segment	18	2310	C7	14.3% per class
senseval2	50	534	C3	33.3% per class
sonar	60	208	C2	46.6/53.4
spectf	44	267	C2	20.6/79.4
speech	50	400	C2	50.0/50.0
vehicle	18	846	C4	25.1/25.7/25.8/23.5
vowel	13	990	C11	9.1% per class
wine	13	178	C3	33.1/39.9/27.0

All datasets were taken from UCI Machine Learning Repository: `http://archive.ics.uci.edu/ml/`, except that `cpuact` is from `http://mldata.org/repository/data/viewslug/uci-20070111-cpu_act/`, `SENSEVAL-2` is from the Second International Workshop on Evaluating Word Sense Disambiguation Systems: `http://www.sle.sharp.co.uk/senseval2`, and `speech` is our In-house developed voice dataset.

shown in Fig. 1.

Overall, results suggest that using LSMI can give better features than HSIC (judged by the error of SVC/SVR). This shows the importance of the availability of a model selection criterion. ℓ_1-LSMI and mRMR are competitive, especially on multi-class classification problems with many classes (e.g., segment and satimage). This is in contrast to PC and Relief which do not handle multi-class problems well. As in the case of the toy data experiment, PC does not perform well in most cases since it does not take redundancy among features into account. An exception would be the `senseval2` problem in which PC performs the best among others. This is because 50 features in the `senseval2` dataset are derived from the first 50 principal components obtained by principal component analysis. Since principal components are orthogonal by definition, no redundancy has to be considered for this problem. In some cases, considering feature redundancy may hurt the performance. This can be seen on `image`, `cpuact`, `senseval2`, and `musk2` datasets when PC outperforms QPFS, suggesting that features may not be correlated. Thus, ignoring redundancy and considering just relevancy gives a better performance. ℓ_1-HSIC performs well in many cases, but the performance may become unstable when k is high due to the mentioned fact that σ_{med} also gets larger.

To objectively compare the performance, another experiment with the same setting is carried out on 22 datasets. The number of trials is set to 50. For each method and dataset, k is set to either 4, 10, or 20 depending on how large m is. The selected k-feature subsets are evaluated by SVC or SVR, as in the previous experiment. The results are given in Table 7, where for each dataset, the method with the best performance is shown in bold face. Other methods which have insignificant performance difference (based on the one-sided paired t-test with 5% significance level) to the best one are also marked in the same way. Note that Lasso works on only binary and regression problems. Thus, the results for multi-class problems are not available. For F-HSIC and F-LSMI, we omit the results on the `ctslices` and `isolet` datasets due to the large computation time involved.

From the table, it can be seen quantitatively that overall ℓ_1-LSMI performs the best by judging from the number of times it ranks top. Interestingly, although worse on small datasets, the performance of mRMR approaches that of ℓ_1-LSMI on high-dimensional datasets (i.e., the `musk1`, `musk2`, `ctslices`, and `isolet` datasets). One reasonable explanation for this phenomenon is that, a large number of features provide more freedom in choosing an alternative subset. Even though there are interacting features, there may be many other alternative non-interacting subsets which give an almost equivalent explanatory power. For this reason, the fact that mRMR cannot detect interacting features may be less significant.

7 Conclusion

Feature selection is an important dimensionality reduction technique which can help to improve prediction performance and speed, and to facilitate the interpretation of a learned predictive model. There are a number of factors which cause the difficulty of feature selection. These include non-linear dependency, feature redundancy, and feature interaction.

The proposed ℓ_1-LSMI is an ℓ_1-based algorithm that maximizes SMI between the selected feature and the output. The main idea is to learn a sparse feature weight vector whose coefficients can be used to determine the importance of features. Only features corresponding to the non-zero coefficients in the weight vector need to be kept. The use of ℓ_1-regularization allows simultaneous consideration of features, which is essential in detecting a group of interacting features.

By combining it with SMI which is able to detect a non-linear dependency and implicitly handle feature redundancy, a powerful feature selection algorithm is obtained.

Extensive experiments were conducted to confirm the usefulness of ℓ_1-LSMI. We there-

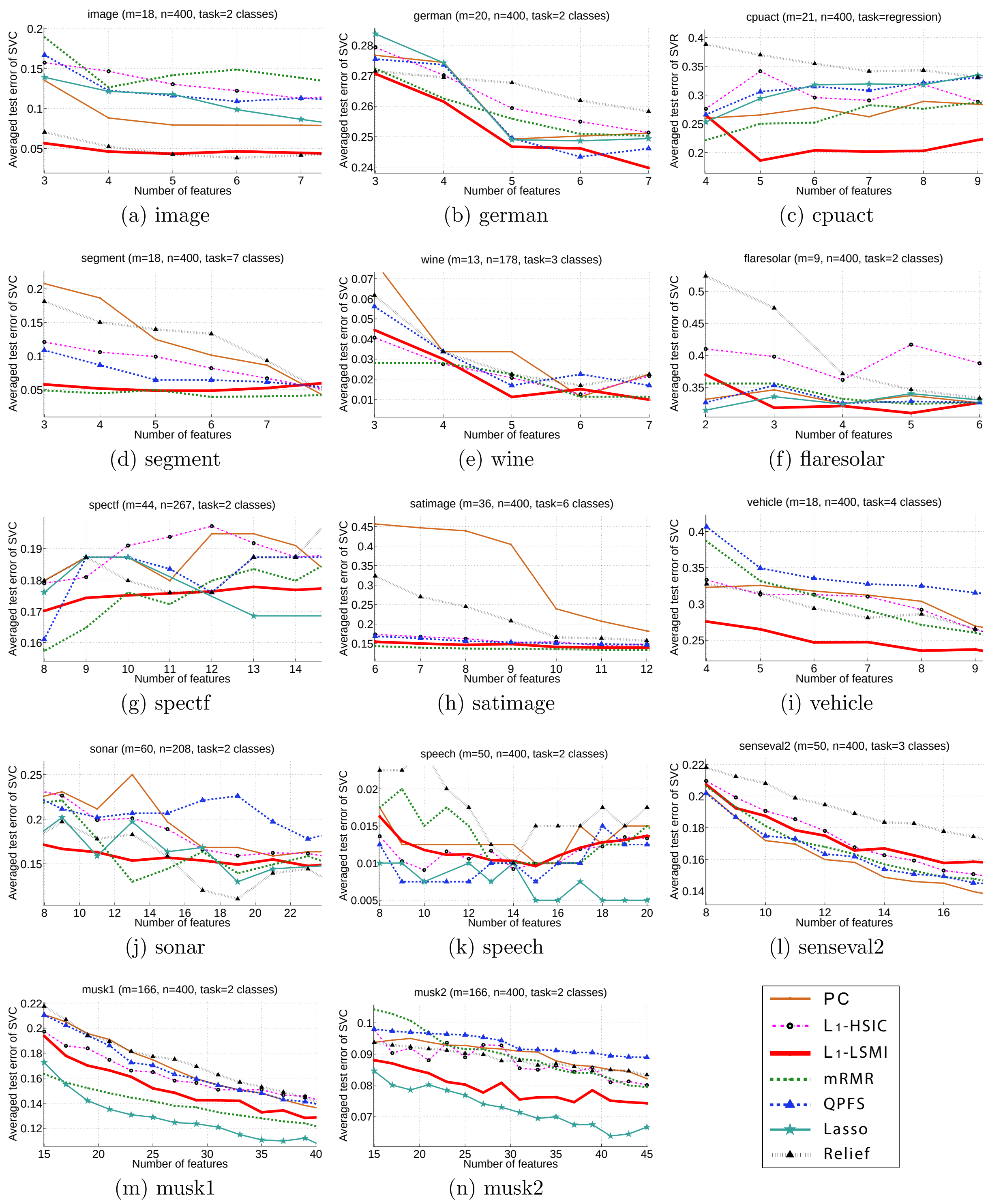

Figure 1: Comparison of SVC/SVR errors of features selected by PC, ℓ_1-HSIC, ℓ_1-LSMI, mRMR, QPFS, Lasso and Relief.

Table 7: SVC/SVR errors of the features selected by PC, F-HSIC, F-LSMI, ℓ_1-HSIC, ℓ_1-LSMI, mRMR, QPFS, Lasso, and Relief on real datasets.

Dataset	m	n	k	PC	F-HSIC	F-LSMI	ℓ_1-HSIC	ℓ_1-LSMI	mRMR	QPFS	Lasso	Relief
abalone (R)	8	400	4	0.73 (.04)	0.74 (.04)	0.70 (.05)	0.73 (.04)	0.70 (.05)	0.73 (.05)	0.75 (.04)	0.70 (.04)	**0.69 (.04)**
bcancer (C2)	9	277	4	0.24 (.00)	0.24 (.00)	0.23 (.01)	**0.23 (.00)**	**0.23 (.01)**	0.25 (.00)	0.23 (.00)	0.24 (.00)	0.26 (.00)
glass (C6)	9	214	4	0.29 (.00)	**0.28 (.00)**	0.30 (.01)	0.30 (.01)	0.30 (.01)	0.30 (.00)	0.29 (.00)	–	0.31 (.00)
housing (R)	13	400	4	4.03 (.19)	4.14 (.20)	4.20 (.21)	3.95 (.20)	**3.91 (.19)**	3.97 (.20)	4.11 (.23)	4.14 (.27)	4.10 (.21)
vowel (C11)	13	400	4	**0.20 (.02)**	0.23 (.03)	0.24 (.03)	0.20 (.02)	0.21 (.02)	**0.20 (.02)**	0.20 (.02)	–	0.21 (.02)
wine (C3)	13	178	4	0.03 (.00)	0.03 (.00)	**0.03 (.01)**	0.03 (.01)	**0.03 (.01)**	0.03 (.00)	0.03 (.00)	–	0.03 (.00)
image (C2)	18	400	4	0.10 (.01)	0.19 (.03)	0.17 (.03)	0.13 (.03)	0.06 (.02)	0.14 (.02)	0.11 (.02)	0.11 (.02)	**0.05 (.01)**
segment (C7)	18	400	4	0.19 (.03)	0.24 (.03)	0.17 (.02)	0.11 (.03)	**0.05 (.01)**	**0.05 (.01)**	0.08 (.03)	–	0.13 (.02)
vehicle (C4)	18	400	4	0.32 (.02)	0.33 (.03)	**0.28 (.02)**	0.34 (.03)	**0.27 (.02)**	0.39 (.05)	0.39 (.05)	–	0.32 (.04)
german (C2)	20	400	4	**0.25 (.02)**	0.29 (.01)	0.29 (.02)	0.25 (.02)	**0.25 (.02)**	0.25 (.02)	0.25 (.02)	0.25 (.02)	0.26 (.02)
cpuact (R)	21	400	4	0.25 (.03)	0.33 (.12)	0.28 (.07)	0.54 (.31)	**0.25 (.16)**	**0.23 (.06)**	0.27 (.04)	0.26 (.04)	0.37 (.09)
ionosphere (C2)	33	351	4	0.07 (.00)	**0.07 (.00)**	0.08 (.01)	0.07 (.00)	0.07 (.00)	0.09 (.00)	0.07 (.00)	0.07 (.00)	0.07 (.00)
satimage (C6)	36	400	10	0.22 (.02)	0.14 (.01)	**0.13 (.02)**	0.14 (.02)	**0.13 (.02)**	0.14 (.01)	0.14 (.02)	–	0.16 (.02)
spectf (C2)	44	267	10	0.19 (.00)	0.17 (.00)	**0.17 (.01)**	0.19 (.01)	**0.17 (.01)**	0.18 (.00)	0.18 (.00)	0.18 (.00)	0.18 (.00)
senseval2 (C3)	50	400	10	**0.18 (.01)**	0.18 (.01)	0.18 (.02)	0.19 (.02)	**0.18 (.01)**	0.18 (.01)	**0.18 (.01)**	–	0.21 (.01)
speech (C2)	50	400	10	0.01 (.00)	0.01 (.00)	0.01 (.00)	0.01 (.00)	0.01 (.00)	0.02 (.00)	0.01 (.00)	**0.01 (.00)**	0.03 (.00)
sonar (C2)	60	400	10	0.23 (.00)	0.22 (.00)	**0.14 (.02)**	0.21 (.02)	0.16 (.02)	0.18 (.00)	0.19 (.00)	0.16 (.00)	0.19 (.00)
msd (R)	90	400	10	0.95 (.06)	0.94 (.06)	**0.92 (.06)**	0.94 (.06)	0.93 (.06)	0.97 (.06)	0.94 (.06)	**0.92 (.06)**	0.96 (.06)
musk1 (C2)	166	400	20	0.19 (.02)	0.17 (.02)	0.14 (.02)	0.16 (.02)	0.16 (.02)	0.15 (.02)	0.18 (.02)	**0.13 (.01)**	0.19 (.03)
musk2 (C2)	166	400	20	0.09 (.01)	0.08 (.01)	**0.07 (.01)**	0.09 (.01)	0.08 (.01)	0.09 (.01)	0.09 (.02)	0.07 (.01)	0.09 (.01)
ctslices (R)	379	400	20	0.79 (.07)	–	–	0.64 (.05)	0.60 (.07)	0.45 (.04)	0.46 (.02)	**0.41 (.03)**	0.56 (.05)
isolet (C26)	617	400	20	0.54 (.03)	–	–	0.36 (.04)	**0.27 (.03)**	0.30 (.03)	0.30 (.03)	–	0.49 (.03)
			Top Count	3	2	7	1	11	3	1	4	2

fore conclude that ℓ_1-LSMI is a promising method for practical use.

Acknowledgments

We thank Dr. Makoto Yamada for his valuable comments. WJ acknowledges the Okazaki Kaheita International Scholarship Foundation, HH acknowledges the FIRST Program, and MS acknowledges the MEXT KAKENHI 23120004.

References

[1] S.M. Ali and S.D. Silvey, "A general class of coefficients of divergence of one distribution from another," Journal of the Royal Statistical Society, Series B, vol.28, no.1, pp.131–142, 1966.

[2] N. Aronszajn, "Theory of reproducing kernels," Transactions of the American Mathematical Society, vol.68, pp.337–404, 1950.

[3] C.C. Chang and C.J. Lin, "LIBSVM: A library for support vector machines," tech. rep., Department of Computer Science, National Taiwan University, 2001. http://www.csie.ntu.edu.tw/˜cjlin/libsvm/.

[4] T.M. Cover and J.A. Thomas, Elements of Information Theory, 2nd ed., John Wiley & Sons, Inc., Hoboken, NJ, USA, 2006.

[5] I. Csiszár, "Information-type measures of difference of probability distributions and indirect observation," Studia Scientiarum Mathematicarum Hungarica, vol.2, pp.229–318, 1967.

[6] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the ℓ_1-ball for learning in high dimensions," Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008), ed. A. McCallum and S. Roweis, pp.272–279, Omnipress, 2008.

[7] K. Fukumizu, F.R. Bach, and M.I. Jordan, "Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces," Journal of Machine Learning Research, vol.5, no.Jan, pp.73–99, 2004.

[8] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with Hilbert-Schmidt norms," Algorithmic Learning Theory, ed. S. Jain, H.U. Simon, and E. Tomita, Lecture Notes in Artificial Intelligence, Berlin, Germany, pp.63–77, Springer-Verlag, 2005.

[9] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," Journal of Machine Learning Research, vol.3, no.Mar, pp.1157–1182, 2003.

[10] H. Hachiya and M. Sugiyama, "Feature selection for reinforcement learning: Evaluating implicit state-reward dependency via conditional mutual information," Machine Learning and Knowledge Discovery in Databases, Part I, ed. J.L. Balcázar, A.G. F. Bonchi, and M. Sebag, Lecture Notes in Computer Science, vol.6321, Berlin, pp.474–489, Springer, 2010.

[11] M.A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, CA, USA, pp.359–366, 2000.

[12] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in Advances in Neural Information Processing Systems 18, ed. Y. Weiss, B. Schölkopf, and J. Platt, pp.507–514, MIT Press, Cambridge, MA, 2006.

[13] K. Kira and L.A. Rendell, "A practical approach to feature selection," Proceedings of the Ninth International Workshop on Machine Learning, San Francisco, CA, USA, pp.249–256, 1992.

[14] R. Kohavi and G.H. John, "Wrappers for feature subset selection," Artificial Intelligence, vol.97, no.1, pp.273–324, 1997.

[15] I. Kononenko, "Estimating attributes: Analysis and extensions of RELIEF," European Conference on Machine Learning, ed. F. Bergadano and L.D. Raedt, New York, NY, USA, pp.171–182, Springer, 1994.

[16] S. Kullback and R.A. Leibler, "On information and sufficiency," The Annals of Mathematical Statistics, vol.22, pp.79–86, 1951.

[17] P. Langley, "Selection of relevant features in machine learning," In Proceedings of the AAAI Fall Symposium on Relevance, Menlo Park, CA, USA, pp.140–144, AAAI Press, 1994.

[18] S.I. Lee, H. Lee, P. Abbeel, and A.Y. Ng, "Efficient L1 regularized logistic regression," Proceedings of the 21st National Conference on Artificial Intelligence (AAAI), pp.401–408, 2006.

[19] F. Li, Y. Yang, and E. Xing, "From lasso regression to feature vector machine," in Advances in Neural Information Processing Systems 18, ed. Y. Weiss, B. Schölkopf, and J. Platt, pp.779–786, MIT Press, Cambridge, MA, 2006.

[20] F. Liese and I. Vajda, "On divergences and informations in statistics and information theory," IEEE Transactions on Information Theory, vol.52, no.10, pp.4394–4412, 2006.

[21] J. Liu, J. Chen, and J. Ye, "Large-scale sparse logistic regression," Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, pp.547–556, 2009.

[22] M. Masaeli, G. Fung, and J.G. Dy, "From transformation-based dimensionality reduction to feature selection," Proceedings of 27th International Conference on Machine Learning, pp.751–758, 2010.

[23] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.27, no.8, pp.1226–1238, 2005.

[24] I. Rodriguez-Lujan, R. Huerta, C. Elkan, and C.S. Cruz, "Quadratic programming feature selection," Journal of Machine Learning Research, vol.11, pp.1491–1516, Aug. 2010.

[25] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," Bioinformatics, vol.23, no.19, pp.2507–2517, 2007.

[26] M.W. Schmidt, G. Fung, and R. Rosales, "Fast optimization methods for L1 regularization: A comparative study and two new approaches," European Conference on Machine Learning, pp.286–297, 2007.

[27] B. Schölkopf and A.J. Smola, Learning with Kernels, MIT Press, Cambridge, MA, USA, 2002.

[28] L. Song, A. Smola, A. Gretton, K.M. Borgwardt, and J. Bedo, "Supervised feature selection via dependence estimation," Proceedings of the 24th Annual International Conference on Machine Learning, pp.823–830, 2007.

[29] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," Journal of Machine Learning Research, vol.2, pp.67–93, 2001.

[30] T. Suzuki and M. Sugiyama, "Sufficient dimension reduction via squared-loss mutual information estimation," Neural Computation, 2012. to appear.

[31] T. Suzuki, M. Sugiyama, J. Sese, and T. Kanamori, "Approximating mutual information by maximum likelihood density ratio estimation," Proceedings of ECML-PKDD2008 Workshop on New Challenges for Feature Selection in Data Mining and Knowledge Discovery 2008 (FSDM2008), ed. Y. Saeys, H. Liu, I. Inza, L. Wehenkel, and Y.V. de Peer, JMLR Workshop and Conference Proceedings, vol.4, Antwerp, Belgium, pp.5–20, Sep. 15 2008.

[32] T. Suzuki and M. Sugiyama, "Least-squares independent component analysis," Neural Computation, vol.23, no.1, pp.284–301, 2011.

[33] T. Suzuki, M. Sugiyama, T. Kanamori, and J. Sese, "Mutual information estimation reveals global associations between stimuli and biological processes," BMC Bioinformatics, vol.10, no.S-1, 2009.

[34] R. Tibshirani, "Regression shrinkage and selection via the lasso," Journal of the Royal Statistical Society (Series B), vol.58, pp.267–288, 1996.

[35] K. Torkkola, "Feature extraction by non-parametric mutual information maximization," Journal of Machine Learning Research, vol.3, pp.1415–1438, March 2003.

[36] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, "Use of the zero norm with linear models and kernel methods," Journal of Machine Learning Research, vol.3, pp.1439–1461, Mar. 2003.

[37] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," Proceedings of the 24th International Conference on Machine Learning, New York, NY, USA, pp.1151–1157, ACM, 2007.

[38] Z. Zhao, L. Wang, and H. Liu, "Efficient spectral feature selection with minimum redundancy," Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, pp.673–678, 2010.

[39] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, "1-norm support vector machines," Advances in Neural Information Processing Systems 16, ed. S. Thrun, L. Saul, and B. Schölkopf, Cambridge, MA, USA, MIT Press, 2004.