Analyzing Kernel Matrices for the Identification of Differentially Expressed Genes

Xiao-Lei Xia[1]*, Huanlai Xing[2], Xueqin Liu[3]

1 School of Mechanical and Electrical Engineering, Jiaxing University, Jiaxing, P.R. China, **2** School of Information Science and Technology, Southwest Jiaotong University, Chengdu, P.R. China, **3** School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast, United Kingdom

Abstract

One of the most important applications of microarray data is the class prediction of biological samples. For this purpose, statistical tests have often been applied to identify the differentially expressed genes (DEGs), followed by the employment of the state-of-the-art learning machines including the Support Vector Machines (SVM) in particular. The SVM is a typical sample-based classifier whose performance comes down to how discriminant samples are. However, DEGs identified by statistical tests are not guaranteed to result in a training dataset composed of discriminant samples. To tackle this problem, a novel gene ranking method namely the Kernel Matrix Gene Selection (KMGS) is proposed. The rationale of the method, which roots in the fundamental ideas of the SVM algorithm, is described. The notion of ”the separability of a sample” which is estimated by performing *t*-like statistics on each column of the kernel matrix, is first introduced. The separability of a classification problem is then measured, from which the significance of a specific gene is deduced. Also described is a method of Kernel Matrix Sequential Forward Selection (KMSFS) which shares the KMGS method's essential ideas but proceeds in a greedy manner. On three public microarray datasets, our proposed algorithms achieved noticeably competitive performance in terms of the B.632+ error rate.

Citation: Xia X-L, Xing H, Liu X (2013) Analyzing Kernel Matrices for the Identification of Differentially Expressed Genes. PLoS ONE 8(12): e81683. doi:10.1371/journal.pone.0081683

Editor: Ken Mills, Queen's University Belfast, United Kingdom

Received July 10, 2013; **Accepted** October 15, 2013; **Published** December 9, 2013

Funding: This work was supported by grants from Natural Science Foundation, Zhejiang Province, P.R. China (Project No. LQ13F030011) and National Science Foundation of P.R. China (Project No. 61133010). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: xxia01@qub.ac.uk

Introduction

Microarray data has been applied to the class prediction of different samples, from which the disease diagnosis and prognosis can benefit. A microarray dataset usually contains thousand of genes and a relatively much smaller number of samples (usually <100). For the purpose of predicting the type of biological samples, a majority of this genes are irrelevant and redundant. This fact has prompted the development of a variety of approaches which detect differentially expressed genes (DEGs) to accomplish an accurate classification of the samples.

The *t*-test has been one of the most widely-used parametric statistical methods for the identification of DEGs between populations of two classes. Variants of the *t*-test, which adopt different technologies to obtain a more stable estimate of the within-class variance for each gene, have been proposed [1–3]. The regularized *t*-test, for example, adjusted the gene-wise variance estimate by using a Bayesian probabilistic model [2]. For multiple testings, the *p*-value is calculated and adjusted to address the problem that the false positive rate is likely to accumulate over thousands of genes. Approaches in this categories range from those bounding the ”Family-Wise Error Rate” (FWER) which is the overall chance of one or more false positives [4–6] and strategies controlling the ”False Discovery Rate” (FDR) which is the expected percentage of false positives among the genes deemed as differentially expressed [1,7]. Because the null distribution is unknown, these methods often shuffle the class labels of the samples to estimate the *p*-value. The ANOVA *F*-test extends the *t*-test to multiple classes and a number of *F*-like statistics have been proposed which used different shrinkage estimators of the gene-wise variance [8,9].

Another family of statistical methods proposed to factor in the dependency information between genes. Representative examples include the gene pair selection method [10] and correlation-based methods the rationale behind which is that a good feature subset is highly correlated with the class and uncorrelated with each other [11,12]. Also included are the approaches derived from Markov blanket filtering [13–15]. Minimum redundancy maximum relevance [16] and uncorrelated shrunken centroid [17] are also well-established gene selection methods in this category.

When cast in the framework of pattern recognition, gene selection is a typical feature selection problem. Feature selection techniques in pattern recognition can be generalized into three types: filter, wrapper and embedded methods [18–20]. For filter methods, the feature selection is performed independently of a classification algorithm, which cover a majority of the aforementioned statistical tests. Wrapper methods, by contrast, use a classifier to evaluate a feature subset. The problem of choosing n out of d features involves altogether $\begin{pmatrix} n \\ d \end{pmatrix}$ feature subsets. An exhaustive evaluation of these subsets is computationally infeasible, particularly for microarray data of a large d. A number of heuristic search techniques are thus proposed, and among them are the

Sequential Forward Selection (SFS), the Sequential Backward Elimination (SBE), the Sequential Forward Floating Selection (SFFS) and the Sequential Backward Floating Elimination (SBFE). The SFS has been used to search for feature subsets which are evaluated by the leave-one-out cross validation accuracy of Least-Squares SVM [21,22]. Genetic Algorithms (GAs) are another family of search strategies that have attracted considerable research attention [23–25].

Embedded methods, on the other hand, use the intrinsic property of a specific classifier to evaluate feature subsets. For example, the SVM Recursive Feature Elimination (SVM-RFE) [26] regards that the normal vector of the linear SVM carries the significance information of the genes. Representative examples also include random forest induced approaches [27,28]. An extensive review of major feature selection techniques has been carried out [29]. No general consensus has yet been reached on which one is the best, despite the diversity and abundance of gene selection algorithms.

Empirically, wrappers and embedded methods have been observed to be more accurate than filters [30]. However, they require repetitive training of a specific classifier in order to guide the search in the space of feature subsets and are consequently very time consuming. Filters are, generally speaking, faster in the absence of interactions between feature subsets and a classifier. Thus filters, statistical tests in particular, have enjoyed considerable popularity in the field of gene selection for microarray data [4,9,31,32]. In fact, wrappers normally incorporate statistical tests as a preprocessing step to prune a majority of genes so that the number of feature subsets to be visited is reduced along the search pathway [21,22,26].

Meanwhile, although the choice of the classifier also presents a wide diversity, SVMs have been widely recognized for its generalization abilities [33] and remained as a predominant option [34–36].

In summary, a widely-accepted scheme for the analysis of microarray data has been ''identification of DEGs by statistical tests followed by sample classification using SVMs''. The justification is that the prediction accuracy of various classifiers including SVMs, depends on how discriminant the features are. However, SVMs belong to the family of sample-based classifiers whose generalization performance comes down to, more precisely, how discriminant the samples are. DEGs identified by statistical tests cannot guaranteed to establish a set of discriminant samples for SVMs. Consequently, it cannot be promised the highest degree of accuracy for sample classification. This problem necessitates the development of gene selection algorithms that are more consistent with the fundamental ideas of SVMs. It is naturally desired that, the proposed methods can bypass the computationally-expensive training procedure of SVMs, which is required by the SVM-RFE algorithm [26] and wrapper methods based on Least-Squares SVMs [21,22].

Materials and Methods

Support Vector Machines

Given a binary classification problem with the training data set of:

$$(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_\ell,y_\ell),\quad \mathbf{x}_i\in\mathbb{R}^d,\quad y_i\in\{-1,1\} \tag{1}$$

where d is the number of features and each $y_i(i=1,\ldots,\ell)$ is the class label for the training sample $\mathbf{x}_i$.

As depicted in Fig. 1, the SVM algorithm seeks the separating hyperplane H_0 which possesses optimal generalization abilities. The hyperplane H_0 takes the form of $\mathbf{w}^{\mathrm{T}}\mathbf{x}+b=0$ where $\mathbf{w}$ is the normal vector and the constant b the bias term. The classifier $<\mathbf{w},b>$ is constructed so that samples from the positive class lie above the hyperplane $H_+:\ \mathbf{w}^{\mathrm{T}}\mathbf{x}+b=+1$ while samples from the negative class lie beneath the hyperplane $H_-:\ \mathbf{w}^{\mathrm{T}}\mathbf{x}+b=-1$.

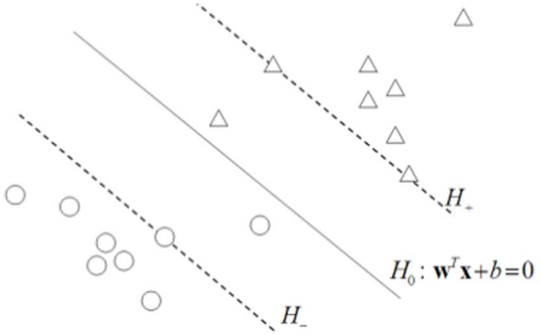


Figure 1. The linear SVM trained on samples from two classes. Samples locating on the hyperplanes of H_+ and H_- are referred to as ''boundary samples''.
doi:10.1371/journal.pone.0081683.g001

The condition of optimality requires that the vector $\mathbf{w}$ be a linear combination of the training samples:

$$\mathbf{w}=\sum_i^\ell \alpha_i y_i \mathbf{x}_i \tag{2}$$

Each constant $\alpha_i(i=1,\ldots,\ell)$ is the Lagrangian multiplier introduced for sample $\mathbf{x}_i$. The feasible value range for the α_i's is $[0,C]$ where C is the regularization parameter and tunes the tradeoff between generalization abilities and the empirical risk.

For nonlinear problems where the training data are not separable in the input space, a function, denoted as $\phi(\cdot)$, is applied, mapping the data to a feature space of higher dimensions where they become separable. Consequently, the normal vector of the resultant classifier becomes:

$$\mathbf{w}=\sum_i^\ell \alpha_i y_i \phi(\mathbf{x}_i) \tag{3}$$

Equation (2) which represents the solution in the linear case, can also be viewed as a special case of Equation (3) where $\phi(\mathbf{x}_i)=\mathbf{x}_i$.

On a test sample $\mathbf{z}$, the SVM classifier outputs a decision value of:

$$f(\mathbf{z})=\mathbf{w}^{\mathrm{T}}\phi(\mathbf{z})+b=\sum_i^\ell \alpha_i y_i \phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{z})+b \tag{4}$$

According to the sign of the $f(\mathbf{z})$, the sample $\mathbf{z}$ obtains a class label of either $+1$ or -1.

Equation (4) suggests that the SVM algorithm requires the knowledge of the dot product between $\phi(\mathbf{x}_i)$, rather than that of $\phi(\mathbf{x}_i)$ itself. Thus the SVM employs the ''kernel trick'' which allows the the dot product between $\phi(\mathbf{x}_i)$ to be computed without the explicit knowledge of the function $\phi(\cdot)$.

Mining the Information Hidden in the SVM Solution

As mentioned previously, each training sample is eventually assigned a Lagrangian multiplier $\alpha_i (i=1,\ldots,\ell)$, subject to $0 \leq \alpha_i \leq C$. The establishment of the SVM classifier is, in actual fact, a process of optimizing the values of these ℓ Lagrangian multipliers. In the SVM solution which is formulated by Equation (4), α_i's can be divided into three groups which respectively satisfy $\alpha_i = 0$, $0 < \alpha_i < C$ and $\alpha_i = C$.

Using Figure 1, we now focus on the linear SVM classifier and review the connection between the value of α_i and the geometric location of its associated training sample $\mathbf{x}_i$. It is worth attention that the connection arises, mathematically, from the optimality conditions of SVMs [37,38]. We then reveal the hidden information that can be mined out of this connection.

1. $\mathbf{x}_i$ with $\alpha_i = 0$

Depending on its class label y_i, $\mathbf{x}_i$ lies geometrically either in the space above $H_+ : \ \mathbf{w}^{\mathrm{T}}\mathbf{x}+b=+1$ for $y_i=+1$ or in the space beneath $H_- : \ \mathbf{w}^{\mathrm{T}}\mathbf{x}+b=-1$ for $y_i=-1$.

Consider a sample $\mathbf{x}_i$ whose $y_i=+1$. Since it locates in the subspace above H_+, we expect $\mathbf{x}_i$ bearing noticeable similarities to class "+" than to class "$-$". The similarity of $\mathbf{x}_i$ and class "+" can be measured by evaluating the the similarity between $\mathbf{x}_i$ and each representative sample from class "+". The training set of the SVM is, or has been supposed to be, composed of representative samples from each class.

We use the inner product to measure the similarity level between vectors. Denoting the number of the positive training samples as n_+, the inner products between $\mathbf{x}_i$ and each each positive training sample $\mathbf{x}_j$ form a population of n_+ measurements, denoted as $\{\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j, \quad j=1,\ldots,n_+\}$. The mean of these measurements, denoted as u_i^+, is regarded to be indicative of the similarity of $\mathbf{x}_i$ and class "+":

$$u_i^+ = \frac{1}{n_+}\sum^{n+} \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j \tag{5}$$

Likewise, denoting the number of the negative training samples as n_-, the similarity of $\mathbf{x}_i$ and class "$-$" can be measured as:

$$u_i^- = \frac{1}{n_-}\sum^{n-} \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j \tag{6}$$

where the set $\{\mathbf{x}_j, j=1,\ldots,n_-\}$ consists of all the negative training samples.

As a result, we can express, mathematically, the expectation that a positive sample bears more resemblance to class "+" than to class "$-$" as:

$$u_i^+ - u_i^- > 0 \tag{7}$$

And a negative training sample $\mathbf{x}_i$ whose $y_i=-1$ and $\alpha_i=0$ is expected to satisfy:

$$u_i^+ - u_i^- < 0 \tag{8}$$

Equation (7) and Equation (8) can be combined into:

$$y_i(u_i^+ - u_i^-) > 0 \tag{9}$$

2. $\mathbf{x}_i$ with $0 < \alpha_i < C$

$\mathbf{x}_i$ with $0 < \alpha_i < C$ lies exactly on either H_+ for $y_i=+1$ or H_- for $y_i=-1$. This group of training samples are normally referred to as "boundary samples".

The class resemblance of a boundary sample to its supposed class is not as striking as those samples with $\alpha_i < 0$. Nevertheless, they are still the samples whose class labels can be correctly restored by the SVM solution and thus expected to satisfy Equation (9).

3. $\mathbf{x}_i$ with $\alpha_i = C$

$\mathbf{x}_i$ whose $y_i=+1$ and $\alpha_i=C$ can be located at one of the following three locations:

(a) exactly on the hyperplane of H_+;
(b) in the region between the hyperplanes of H_+ and H_- but closer to H_+;
(c) in the region between the hyperplanes of H_+ and H_- but closer to H_-.

A training sample from group (a), is a boundary sample but its class label can be correctly restored by the SVM solution. As with positive samples whose $0 < \alpha_i < C$, Equation (9) is expected to hold for samples from case (a).

For a training sample from group (b), the SVM classifier would not have been able to correctly restore its class label if it weren't for the introduction of the slack variables. Our interpretation is that, the class resemblance of this sample to its supposed class is so ambiguous that the SVM has difficulties in acknowledging its actual class membership.

For a training sample from group (c), the SVM classifier is simply unable to correctly restore its class label. It is very likely that the class resemblance of this sample to its supposed class in fact contradicts its given class label.

In mathematical terms, we reckon that a positive training sample of either group (b) or group (c) satisfies:

$$u_i^+ - u_i^- \leq 0 \tag{10}$$

The hidden information for $\mathbf{x}_i$ whose $y_i=+1$ and $\alpha_i=C$ can be inferred in a similar manner. And the formulation that describes a sample $\mathbf{x}_i$ with $\alpha_i=C$ can be generalized as:

$$y_i(u_i^+ - u_i^-) \leq 0 \tag{11}$$

In summary, the resultant value of α_i for the training samples $\mathbf{x}_i$ suggests how discriminant $\mathbf{x}_i$ is between two opposing classes. But the values of α_i's can only be obtained after the completion of the training procedure which is of a formidable time complexity of $O(\ell^3)$.

Luckily, our analysis above implies that that the function of $y_i(u_i^+ - u_i^-)$ is, promisingly, indicative of the discriminant level of $\mathbf{x}_i$. In other words, the vector of $\{\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j, j=1,\ldots,\ell\}$ is highly informative about the complexity of classifying $\mathbf{x}_i$ by the linear SVM classifier. It is easy to infer that, for nonlinear problems, this information can be obtained from the vector of $\{\phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j), j=1,\ldots,\ell\}$.

Estimating the Separability of a Problem

In the SVM algorithm, the vector of $\{\phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j), j=1,\ldots,\ell\}$ constitutes the i-th column of the input kernel matrix. The ℓ measurements in the i-th column can be separated into two populations, according to the class label of sample i, and respectively denoted as K_{+i} and K_{-i}. Performing the following test to the two populations yields a score s_i which measures the separability of $\mathbf{x}_i$:

$$s_i = y_i \cdot \frac{u_i^+ - u_i^-}{\sigma_i^+ + \sigma_i^-} \tag{12}$$

where $u_i^+(u_i^-)$ and $\sigma_i^+(\sigma_i^-)$ are the mean and the standard deviation of K_{+i} (K_{-i}).

We justify the introduction of standard deviations in the denominator by considering two positive training samples in the feature space. The first sample is assumed to have come from a region of denser population than the second one. We reckon that, compared with the second sample, the first sample is more typical a representative of class "+" and is believed to be more similar to class "+". The positive sample from a denser population is expected to exhibit a lower deviation of the elements $\{\phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j),(j=1,\ldots,n_+)\}$. Thus, the standard deviation is formulated into Equation (12), demonstrating our confidence in a higher separability of a sample from a denser population.

The values of s_i's can be split into three types, large positive ones, small positive ones and negative ones. A large positive value of s_i implies that, the training sample $\mathbf{x}_i$ is likely to be discriminant, statistically bearing more resemblance to the supposed class than to the other one. A small positive s_i suggests that, $\mathbf{x}_i$ might bear almost the same level of resemblance to both classes and thus, hard to classify. For a negative value on s_i, the class that $\mathbf{x}_i$ is computed as more similar to, is different from the actual one, which poses difficulties for the SVM classifier.

Meanwhile, the similarity between each sample and itself is supposed to be 1. However, It is not the case for all kernel functions to satisfy $K_{ii}=1$. Consequently a proper preprocessing procedure might be required prior to the application of Equation (12), depending on the kernel in use. For linear kernels, we divide each element of the i-th column of the kernel matrix by $\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_i$. For Gaussian RBF kernels [29] which take the form of

$$K_{ij} = \exp(-\lambda\|\mathbf{x}_i - \mathbf{x}_j\|^2) \tag{13}$$

it already holds that $K_{ii}=1$ and the preprocessing step is avoided. However, the value of the parameter λ is required to be optimized.

Since the separability of each sample has an impact on the the class separability of a problem, we propose to use the sum of each sample's separability score as an estimate of the separability of the problem.

A word about the formulation of Equation (12). In statistics, it is the norm of practice to add a small constant to the sum of variances, in order to guard against zero in the denominator. But for our algorithms, the designation of $K_{ii}=1$ prevents the occurrence of zero in the denominator of Equation (12). We explain how it is achieved for linear kernels and Gaussian RBF kernels:

(1) In the case of linear kernels, take a positive sample $\mathbf{x}_i$ for example. Since $K_{ii}=1$, in order to have $\sigma_i^+=0$ for Equation (12), it demands that $K_{ij}=\frac{\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j}{\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_i}=1$ for any j whose $\mathbf{x}_j$ is a positive sample. This requires that $\mathbf{x}_i^{\mathrm{T}}(\mathbf{x}_i-\mathbf{x}_j)=0$. This set of conditions can only satisfied either when $\mathbf{x}_i=0$ or $\mathbf{x}_i=\mathbf{x}_j$ which suggests that the training set only include one positive sample. We reckon that either case is unlikely for well-posed classification problems.

(2) In the case of Gaussian RBF kernels, in order to have $\sigma_i^+=0$ given a positive sample, it has to be met that $K_{ij}=\exp(-\lambda\|\mathbf{x}_i-\mathbf{x}_j\|^2)=1$ for any j whose $\mathbf{x}_j$ is a positive sample. This in fact implies that either $\mathbf{x}_i=\mathbf{x}_j$ which is hardly true with real-life microarray datasets, or the parameter λ has been assigned a value of zero, which can be easily avoided.

Kernel Matrix Induced Gene Selection Algorithms

Since each gene subset introduces a classification problem represented by the set of training samples, the gene subset thus corresponds to an estimate of the separability of the problem. Consequently, DEGs can be identified as those resulting in "easier problems" of high separability. This is the essential idea of our kernel matrix induced gene selection methods, which has been illustrated in Figure 2. This methodology is shared by the two gene selection algorithms we proposed below. The first algorithm, namely the Kernel Matrix Gene Selection (KMGS), ranks each gene individually, while the second one, namely the Kernel Matrix Sequential Forward Selection (KMSFS), identifies DEGs iteratively.

Kernel Matrix Gene Selection. Given a microarray dataset of ℓ samples with d genes, the n-th $(1\le n\le d)$ gene of the ℓ samples forms a vector. The vector, in fact, establishes a training set for the following classification problem:

$$(x_{1n},y_1),\ldots,(x_{\ell n},y_\ell),\quad x_{in}\in\mathbb{R},\quad y_i\in\{-1,1\},\quad i=1,\ldots,\ell \tag{14}$$

where x_{in} is the value of n-the gene for the i-th sample and the y_i is its given class label. Given the training set, the separability of each sample, denoted as $s_i(n)$, can be assessed using Equation (12). The class separability of the problem constructed from the n-th gene can thus be computed:

$$f(n) = \sum_{i=1}^{\ell} s_i(n) \tag{15}$$

while the reason behind using (15) is that the class separability of a problem is reflected by the sum of the separability of each sample.

Hence the function $f(n)$ maps a gene to the separability level, in the contexts of sample-based classifiers including the SVM.

The d genes are ranked according to their respective $f(n)$ value where $n\in[1,d]$. Genes achieving a large $f(n)$ obtain higher rankings.

Kernel Matrix Sequential Forward Selection. An alternative to the KMGS which proceeds in a greedy manner is also developed, which is namely the Kernel Matrix Sequential Forward Selection (KMSFS) algorithm. The algorithm starts with an empty set of selected DEGs. At each iteration, the algorithm identifies a single DEG which is then appended to the set. We now describe how the KMSFS algorithm proceeds between two consecutive iterations.

Given a microarray dataset of ℓ samples with d genes, at the n-th iteration, n genes has been collected into the set of DEGs. This in fact stands for a classification problem with the training set composed of ℓ samples, each of which is of n dimensions:

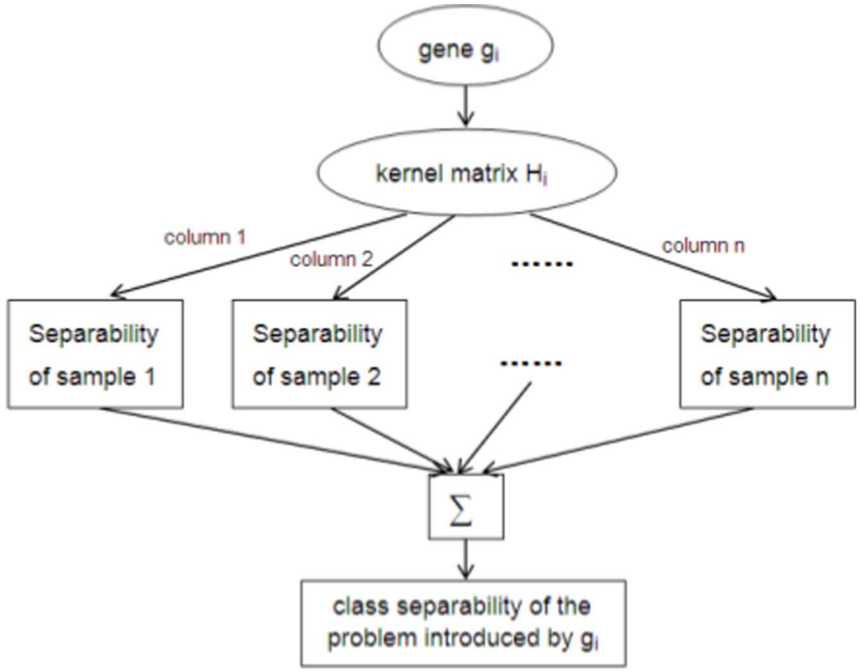


Figure 2. The essential idea of kernel matrix induced gene selection algorithms.
doi:10.1371/journal.pone.0081683.g002

$$(\mathbf{x}_{1n}, y_1), \ldots, (\mathbf{x}_{\ell n}, y_\ell), \quad \mathbf{x}_{in} \in \mathbb{R}^n, \quad y_i \in \{-1, 1\}, \quad i = 1, \ldots, \ell \qquad (16)$$

Each gene from the remaining $(d-n)$ genes is, in turn, appended to these n genes and forms a different classification problem with a training set of ℓ samples, each of which is of $(n+1)$ dimensions. This results in, altogether, $(d-n)$ data matrices of size $\ell \times (n+1)$ which are actually the training sets for $(d-n)$ classification problems. The complexity of each problem can be estimated and interpreted as the significance of the associated $(n+1)$-th gene. The $(n+1)$-th DEG is eventually identified to be the one which produces the problem featuring the highest separability.

The pseudo codes for the KMGS and KMSFS algorithms are given respectively in Table 1 and Table 2.

Merits of Proposed Algorithms

The proposed methods have noticeable merits:

1. Filter methods identify discriminant features, making them suitable for feature-based classifiers whose normal vector is the linear combination of features. However, Equation (3) demonstrates that the SVM classifier is the linear combination of training samples in the feature space. Thus the performance of the SVM comes down, more to discriminant levels of samples than those of features. Since discriminant features selected by filter methods are not guaranteed to generate a training set composed of discriminant samples, the resultant classifier cannot be ensured to be optimally accurate either. In contrast, our algorithms aim at unveiling the information regarding discriminant levels of samples using the kernel function. Our algorithms are developed upon the fundamental ideas of SVMs and thus more likely to produce a classifier of a higher degree of accuracy.
2. A majority of wrapper and embedded methods are based on the assumption that most microarray datasets pose linear problems. However, we reckon that, the problem presented by a set of DEGs can hardly be a linear one when the the set size is as small as only one or two.

But the generalization to nonlinear cases have been challenging for various wrappers and embedded methods. For example, the SVM-RFE [27] keeps unchanged the Lagrangian multipliers α_i's from the previous iteration and then selects the gene which makes the least change to the dual objective function. The strategy of fixing α_i's is likely to compromises the significance evaluation of each gene, as well as the generalization abilities of the resultant SVM classifier.

Advantageously, our algorithms can be directly applied to nonlinear cases by opting for Gaussian RBF kernels. The Gaussian RBF K_{ij}, according to Mercer's conditions, is an inner product of $\mathbf{x}_i$ and $\mathbf{x}_j$ in the feature space:

$$K_{ij} = \phi(\mathbf{x}_i)^{\mathrm{T}} \phi(\mathbf{x}_j) \qquad (17)$$

where Φ is the function mapping a sample from the input space to the feature space. Thus, for the nonlinear case, the Gaussian kernel matrix is still composed of similarity measurements between training samples.

3. The output of Equation (15) which measures the significance of genes is a real-valued number rather than an integer. This avoids the ties problem [40] which often occurs to count based wrapper methods including the one using the leave-one-out cross validation error as the selection criterion[21].

Datasets and Data Preprocessing

Prostate dataset. The dataset contains, in total, 136 samples of two types which respectively have 77 and 59 cases. Each sample includes expression values of 12600 genes.

Table 1. The Algorithm of Kernel Matrix Gene Selection.

INPUT:	
	- The data set $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_\ell,y_\ell)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ $(i=1,\ldots,\ell)$
	- m is a user-defined integer which indicates the number of selected genes.
FOR $n=1,\ldots,d$	
	- A training data set is formed by the n-th feature of the original training set:
	$(x_{1n},y_1),\ldots,(x_{\ell n},y_\ell), \quad y_i \in \{-1,1\}, \quad i=1,\ldots,\ell$
	where $x_{in} \in \mathbb{R}$ is the n-th feature of the i-th sample $\mathbf{x}_i$.
	- Construct the kernel matrix $K^{(n)}$. In the case of the linear kernel function, apply the aforementioned preprocessing procedure to ensure that: $K_{ii}^{(n)}=1$ for each column.
	- At the j-th column entry, the measurement $K_{ij}^{(n)}$ is assigned into either the population $K_{+j}^{(n)}$ or $K_{-j}^{(n)}$ depending on the class label y_i. Perform the following statistical test on the two populations:
	$s_i(n)=y_i \cdot \frac{u_i^+ - u_i^-}{\sigma_i^+ + \sigma_i^-}$
	where u_i^+ (u_i^-) and σ_i^+ (σ_i^-) are the means and standard deviations of $K_{+i}^{(n)}$ $(K_{-i}^{(n)})$.
	- The class separability w.r.t feature n can be assessed by $f(n)$:
	$f(n)=\sum_{i=1}^{\ell} s_i(n)$
OUTPUT:	
	- Choose the genes which correspond to the top m values of $f(n)$ $(n=1,\ldots,d)$

doi:10.1371/journal.pone.0081683.t001

Table 2. The Algorithm of Kernel Matrix Sequential Forward Selection.

INPUT:	
	- The data set $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_\ell,y_\ell)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ $(i=1,\ldots,\ell)$
	- m is a user-defined integer which indicates the number of selected genes.
	- γ is the index set of candidate features. Initially $\gamma_i=i$ where $i \in \{1,\ldots,d\}$.
	- D is a $\ell \times \ell$ matrix which is $\mathbf{D}_0=\overleftarrow{0}$ at the start.
-FOR $k=1,\ldots,m$	
	- FOR $n=k,\ldots,d$
	- A matrix $\mathbf{D}^{(n)}$ is established whose element $\mathbf{D}_{ij}^{(n)}$ is:
	$\mathbf{D}_{ij}^{(n)}=(x_{in}-x_{jn})^2$
	where $x_{in} \in \mathbb{R}$ is the γ_n-th feature of the original i-th training sample $\mathbf{x}_i$.
	- Add the γ_n-th feature to the set of previously $(k-1)$ selected features whose indices are $\{\gamma_1,\ldots,\gamma_{k-1}\}$.
	A training data set is formed by the k features of the original training set:
	$(\mathbf{x}_1^{(n)},y_1),\ldots,(\mathbf{x}_\ell^{(n)},y_\ell), \quad \mathbf{x}^{(n)} \in R^k, \quad y_i \in \{-1,1\}$
	- Construct the kernel matrix $K^{(n)}$. In the case of the linear kernel function, apply the aforementioned preprocessing procedure to ensure that $K_{ii}^{(n)}=1$ for each column.
	- Calculate the score $f(n)$ w.r.t $K_{ij}^{(n)}$ using (12) and (15).
	- The k-th iteration of the outer loop finds a feature γ_n:
	$\gamma_n = \underset{n=k,\ldots,d}{\operatorname{argmax}} f(n)$
	- Swap γ_n and γ_k so that γ_k always points to the newly selected feature:
	$\gamma_k \leftrightarrow \gamma_n$
	- Update $\mathbf{D}_{k-1}$ to $\mathbf{D}_k$:
	$\mathbf{D}_k=\mathbf{D}_{k-1}+\mathbf{D}^{(k)}$
OUTPUT:	
	- The index of m selected genes are given by $\{\gamma_1,\ldots,\gamma_m\}$ in chronological order.

doi:10.1371/journal.pone.0081683.t002

Colon dataset. The dataset contains the expression values of 2000 genes from 62 tissues, of which 22 are normal and 40 are cancerous.

Leukaemia dataset. The dataset was collected from 72 patients. 47 of them were diagnosed with acute acute lymphoblastic leukemia (ALL) and 25 with acute myeloid leukemia (AML). Expression values of 7129 genes were measured.

Both the prostate dataset and the colon dataset were normalized using the following procedure. A microarray dataset with ℓ samples and d genes was arranged as a matrix of ℓ rows and d columns. Each row of the matrix was standardized so that the mean and the standard deviation for the row vector are respectively zero and unity. Next, each column of the resultant matrix was standardized to have zero mean and unity standard deviation. No further processing was conducted. All the simulations and comparisons have been performed on the standardized data.

For the leukemia dataset, we applied the pre-processing procedure proposed by Dudoit et al. [41] which consisted of (i) thresholding (floor of 100 and ceiling of 16000), (ii) filtering (exclusion of genes with max/min$=5$ and max-min$=500$ across the samples), (iii) base 10 logarithmic transformation, leaving us with 3571 genes. Next, we applied Fisher's ratio and selected the 1000 top DEGs. For each individual gene, Fisher's ratio assigns it a score using the function $f=(\mu_+-\mu_-)^2/(\sigma_+^2+\sigma_-^2)$ where $\mu_+(\mu_+)$ and $\sigma_+(\sigma_-)$ are respectively the mean and the standard deviation across samples from the positive(negative) class. The preprocessing strategy which was also employed by [21] makes possible a fairer comparison between our experiment results and those reported in [21]. All the simulations and comparisons regarding the leukemia dataset have been performed on the preprocessed and pre-selected data.

Error Rate Estimation Techniques

Various gene selection algorithms are evaluated and compared by the error rate of SVMs. The simplest technique for error estimation is the holdout method which splits the dataset into a training set and a test set. The gene selection algorithm is performed on the training set and sample classification on the test set. However, the holdout method has been highly discouraged for microarray datasets which usually contain a small number of samples. In contrast to researchers who applied gene selection to the entire training set and employed k-fold cross validation to assess the selected DEGs, Ambroise and McLachlan [42] emphasized to exclude samples used for validation from the gene selection procedure and labelled techniques that follow their recommendation as "external" ones. They suggested that external 10-fold cross validation and external B.632+ bootstrap could produce unbiased estimate [42,43]. Due to the problem of high variance with cross validation techniques when applied to microarray datasets [44], we used the external B.632+ estimator for the comparison of gene selection algorithms.

The B.632+ estimator involves resampling, with replacement, of the original dataset. From a dataset of ℓ samples denoted as $\mathbf{X}=\{\mathbf{x}_1,\ldots,\mathbf{x}_\ell\}$, a single sample is randomly drawn and then put back at each time. This process is repeated ℓ times, leading to a new set which is denoted as $\mathbf{X}^*=(\mathbf{x}_1^*,\ldots,\mathbf{x}_\ell^*)$. The resampled set $\mathbf{X}^*$ includes, with probability, duplicates of a sample from the original set $\mathbf{X}$. The number of duplicates for a sample included in $\mathbf{X}^*$ ranges from 0 to ℓ. The set of $\mathbf{X}^*$ is used for both gene selection and training a SVM. The SVM classifier is then tested on the set of $(\mathbf{X}-\mathbf{X}^*)$. A good error estimator requires the generation of B resampled sets which are denoted as $\mathbf{X}^{*1},\mathbf{X}^{*2},\ldots,\mathbf{X}^{*B}$ where it was recommended that $B\geq 200$. We set $B=200$ for all our experiments. Meanwhile, for each sample $\mathbf{x}_i\in\mathbf{X}(i=1,\ldots,\ell)$, its overall number of occurrences in the B resampled sets is ensured to be B, which further reduces the variance.

The flow chart for evaluating a gene selection algorithm using the B.632+ technique has been given by Figure 3.

Gene Selection Algorithms

For the methods of KMSFS and KMGS, both the linear kernel and the Gaussian RBF kernel were tested. This resulted in altogether four algorithms which are referred to as Gaussian KMSFS, Gaussian KMGS, linear KMSFS and linear KMGS. They were compared with two wrapper methods: the leave-one-out calculation sequential forward selection (LOOSFS) which improved the least-squares bound measure [21] by easing the ties problem, and the gradient-based leave-one-out gene selection (GLGS) method [22] which was claimed to outperform the SVM-RFE algorithm [26]. Comparisons were also made to a number of filter methods, including the aforementioned Fisher's ratio [45], Cho's [46] and two other methods of Yang's [47]. We described the ideas of these gene selection algorithms below.

Leave-One-Out Calculation Sequential Forward Selection (LOOSFS). The Leave-One-Out Cross-Validation(LOOCV) error has been generally used for measuring the generalization abilities of SVMs and Least-Squares SVMs (LS-SVMs). The LOOSFS method thus identifies as DEGs those genes which result in a LS-SVM classifier with the minimal Leave-One-Out Cross-Validation(LOOCV) error rate. The beauty of the algorithm consists in the efficient and exact computation of the LOOCV error. To address the "ties problem" in which multiple gene subsets achieve the same LOOCV error rate, a further selection

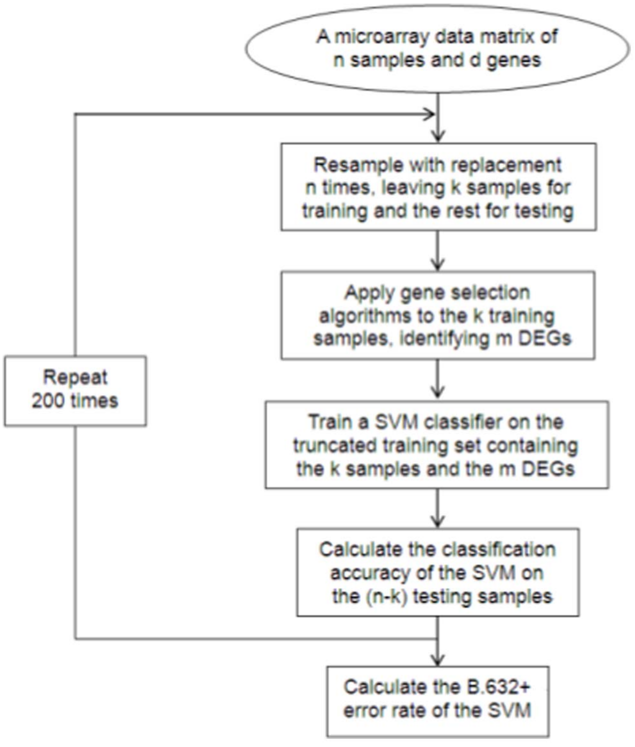

A microarray data matrix of n samples and d genes
Resample with replacement n times, leaving k samples for training and the rest for testing
Apply gene selection algorithms to the k training samples, identifying m DEGs
Train a SVM classifier on the truncated training set containing the k samples and the m DEGs
Calculate the classification accuracy of the SVM on the (n-k) testing samples
Repeat 200 times
Calculate the B.632+ error rate of the SVM

Figure 3. The flow chart for evaluating a gene selection algorithm using the B.632+ technique.
doi:10.1371/journal.pone.0081683.g003

criterion is imposed which favors the gene subset with minimal empirical risk.

Gradient-Based Leave-One-Out Gene Selection (GLGS). The starting point of the GLGS method is also the employment of the exact formulation of the LOOCV error for LS-SVMs. The method then utilizes the gradient descent algorithm to seek a diagonal matrix which eventually minimizes the LOOCV error. Genes are ranked according to the absolute values of the diagonal elements of the diagonal matrix.

With Cho's method and Yang's methods, genes are individually ranked. We use the following notations for their descriptions. Each micorarray dataset with ℓ samples and d genes is treated as a matrix, denoted as $\{a_{ij}\}$, where a_{ij} indicates the expression value of gene j for sample i. Given a n-class problem, the average expression value of each class, in terms of gene j, can be computed and denoted as the set of $(\bar{a}_{1j},\bar{a}_{1j},\ldots,\bar{a}_{nj})$. Denote the standard deviation for the set as $\hat{a}_j$. A matrix, denoted as $\{x_{ij}\}$ is also introduced, where $x_{ij}=\sqrt{(a_{ij}-\bar{a}_{ij})^2}$.

Cho's Method. The score, denoted as $s(j)$, that gene j obtains eventually is:

$$s(j)=\frac{1}{\bar{a}_j}\cdot\mathrm{mean}(j)\cdot\mathrm{std}(j) \tag{18}$$

where

$$\mathrm{mean}(j)=\sum_{i=1}^{\ell}\frac{w_i}{W}x_{ij} \tag{19}$$

$$\mathrm{std}(j)=\sqrt{\frac{\sum\limits_{i=1}^{\ell}(x_{ij}-\mathrm{mean}(j))^2}{(\ell-1/\ell\sum\limits_{i=1}^{\ell}w_i)}} \tag{20}$$

w_i is the reciprocal of the number of samples that share the same class label as sample i and $W=\sum_{i=1}^{\ell}w_i$. A small value of $s(j)$ indicates that samples of the j-th gene are clustered the centroid of each class.

Yang's Methods. The between-class variation with respect to gene j, denoted as scatter(j), is formulated as:

$$\mathrm{scatter}(j)=\sqrt{\frac{1}{n}\sum_{k=1}^{n}(\bar{a}_{kj}-\hat{a}_j)^2}+\frac{1}{2}\min_{p\neq q}|\bar{a}_{pj}-\bar{a}_{qj}| \tag{21}$$

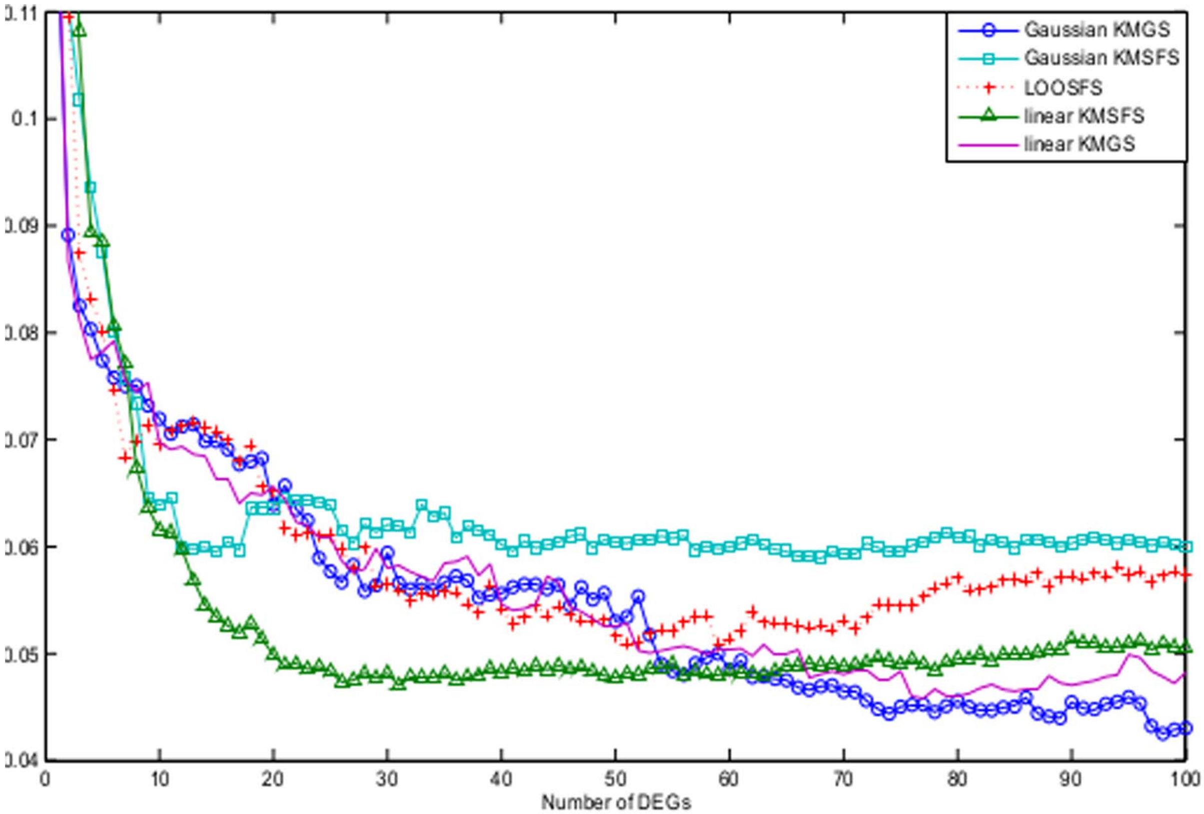


Figure 4. The B.632+ error shown as a function of the number of DEGs for the prostate dataset. The curves depict the performance of the following wrapper methods with their respective optimal parameter settings: Gaussian KMGS with $\lambda=1$ and $C=0.1$; Gaussian KMSFS with $\lambda=0.1$ and $C=0.1$; LOOSFS with $\lambda=0.0001$ and $C=0.1$; linear KMSFS with $C=0.01$; linear KMGS with $C=0.1$. The performance of linear KMSFS was the best when the number of DEGS was between 10 and 60, while Gaussian KMGS outperformed the rest when the number of DEGs increases further to 100. The lowest B.632+ rate was achieved by Gaussian KMGS.
doi:10.1371/journal.pone.0081683.g004

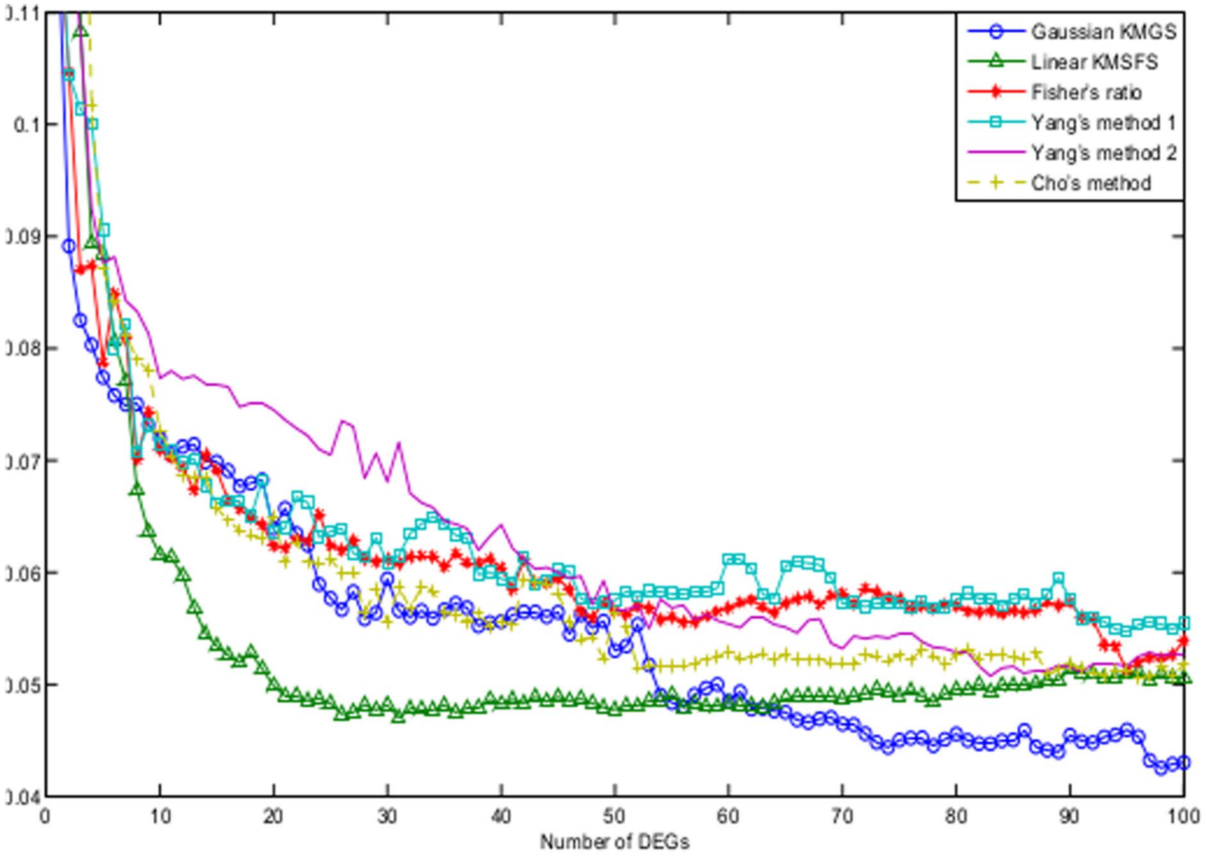


Figure 5. The B.632+ error shown as a function of the number of DEGs for the prostate dataset. The curves are obtained from the following algorithms with their respective optimal parameter settings: Fisher's ratio with $C=0.1$; Yang's methods both of which with $C=0.1$; Cho's method with $C=0.1$; linear KMSFS with $C=0.01$; Gaussian KMGS with $\lambda=1$ and $C=0.1$. Linear KMSFS and the Gaussian KMGS performed better than the 4 filter methods.
doi:10.1371/journal.pone.0081683.g005

In order to estimate within-class variations in terms of gene j, a function $d(j)$ is first introduced:

$$d(j)=\sqrt{\frac{1}{n}\sum_{k=1}^{n}f(\{x_{kj}\})} \quad (22)$$

where $f(\{x_{kj}\})$ is a function of $\{x_{kj}\}$ which are composed of the elements from the j-th column of the matrix $\{x_{ij}\}$ and associated with the k-th class. Denote $\bar{x}_{kj}$ as the mean of of $\{x_{kj}\}$. $f(\{x_{kj}\})$ can be either the squared $\bar{x}_{kj}$ or the mean of $\{x_{kj}^2\}$, which results in two forms of $d(j)$ which are referred to as $d_1(j)$ and $d_2(j)$ respectively.

Two metrics for measuring within-class variations, denoted as compact$_1(j)$ and compact$_2(j)$ which are derived respectively upon $d_1(j)$ and $d_2(j)$, are proposed:

$$\mathrm{compact}_p(j)=d_p(j)+\sqrt{d_p(j)^2-\mu(j)^2}, \quad p=1,2 \quad (23)$$

where $\mu(j)$ is the mean of the set of $\{\bar{x}_{kj}\}$ where $k=1,\ldots,n$.

Eventually, two score functions which decide the ranking of gene j, are given:

$s_p(j)=\mathrm{compact}_p(j)/\mathrm{scatter}(j)$, p = 1,2

We refer to these two score functions respectively as Yang's method 1 and Yang's method 2.

Cho's method and Yang's two methods identify as DEGs those genes whose associated $s(j)$ are smaller. For all the gene selection algorithms, it has been emphasized to exclude the test subset each time from the gene selection procedure in order to obtain an unbiased evaluation [42,43]. The gene selection algorithms terminated when a specific number of DEGs have been identified and we set this number to be 100.

Parameter Tuning

We employed grid search and Friedman rank sum tests combined with Holm correction to tune the parameters for different algorithms.

Grid Search. Among the total 10 gene selection algorithm, both Gaussian KMSFS and Gaussian KMGS require the parameter λ in Equation (13) to be optimized. For the LOOSFS algorithm, the regularization parameter, denoted here as λ for consistency, has to be tuned. λ was varied sequentially from 10^{-5} to 10^{5} in multiples of 10, which made up a total of 11 different values. With respect to sample classification, the linear SVM was used throughout. Its regularization parameter, denoted as C, ranged from 10^{-3} to 10^{3} in multiples of 10, which gives 7 different values. Thus, 77 value pairs for (λ,C) were tested for Gaussian KMSFS, Gaussian KMGS and LOOSFS algorithms, while 7

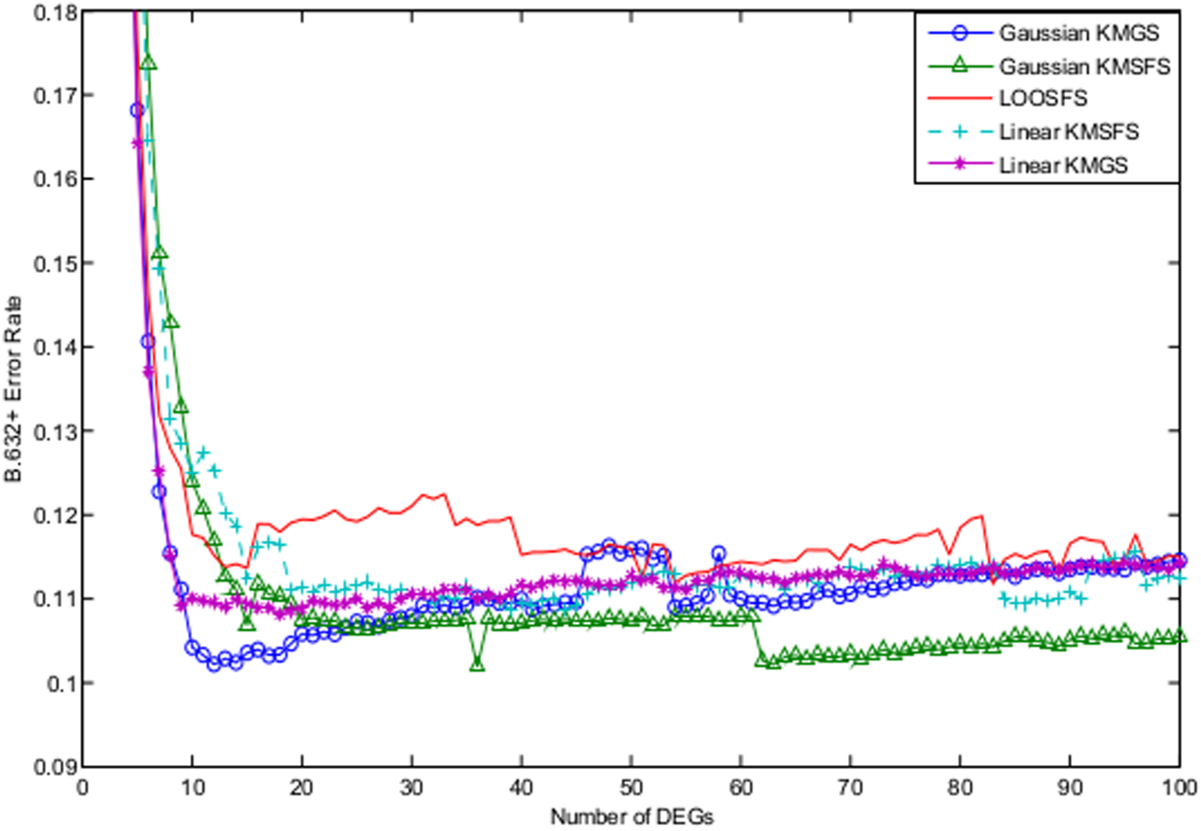


Figure 6. The B.632+ error shown as a function of the number of DEGs for the colon dataset. The curves depict the performance of the following wrapper methods with their respective optimal parameter settings: Gaussian KMGS with $\lambda = 1$ and $C = 0.01$; Gaussian KMSFS with $\lambda = 0.1$ and $C = 0.01$; LOOSFS with $\lambda = 0.001$ and $C = 0.1$; linear KMSFS with $C = 0.01$; linear KMGS with $C = 0.01$. The performance of Gaussian KMSFS was shown to be the best while the performance of the LOOSFS was the worst.
doi:10.1371/journal.pone.0081683.g006

different values of C were evaluated for the rest of the 10 gene selection algorithms.

Friedman Rank Sum Test with Holm Correction. The Friedman rank sum test is a non-parametric alternative to ANOVA with repeated measures. The test statistic for the Friedman test is a Chi-square with $n-1$ degrees of freedom, where n is the number of repeated measures.

We take the algorithm of Gaussian KMSFS as an example to explain how to apply Friedman test for the discovery of optimal values on (λ,C). As mentioned previously, 100 DEGs were selected, from each a new classification problem arose. We thus obtained altogether 100 B.632+ error rates for each setting on (λ,C). As we tried 77 settings for the parameter pair, 77 groups of classification accuracies were obtained.

Friedman rank sum test was used to detect statistical differences among these 77 groups. The test was based on 100 sets of ranks, with each set corresponding to an individual classification problem. The performances of different parameter settings analyzed are ranked separately for each problem. If we rejected the null-hypothesis stating that all the 77 settings led to equal performance in mean ranking, we employed the Holm post-hoc analysis to identify which setting was significantly better than the rest.

All the gene selection methods were coded in Matlab. The linear SVM was implemented using LIBSVM [48] and the Friedman test with Holm correction was coded in R. The specifications of the computer running the experiments were: Intel core i5-2320 quad-core processor 3.0 GHz, Memory 4 GBytes and the operating system of Windows 7.

Results

Results on the Prostate Dataset

Optimal Parameter Settings. Using Friedman tests with Holm correction, optimal settings on (λ,C) for Gaussian KMGS, Gaussian KMSFS and LOOSFS were found to be (1,0.1), (0.1,0.1) and (0.0001,0.1) respectively.

GLGS and linear KMSFS shared the optimal setting of $C = 0.01$. For linear KMGS and all the filter methods which are respectively Fisher's ratio, Cho's method and the two methods of Yang's, the optimal parameter settings were uniformly $C = 0.1$.

Comparisons against Wrappers. With a minimal error rate of 0.1240 and a mean error rate of 0.1932, the performance of GLGS was much worse than that of the other 9 methods. Thus its simulation results were not graphically presented.

Figure 4 illustrates the the B.632+ error rates as a function of the number of DEGs, for the algorithms of Gaussian KMGS, Gaussian KMSFS, LOOSFS, linear KMSFS and linear KMGS. It can been seen that, when the number of DEGs fell between 10 and around 60, linear KMSFS which is represented by the green solid line dotted with upper triangles, remained the best. As the number of DEGs further increased, Gaussian KMGS outperformed the rest and achieved the lowest B.632+ error rate.

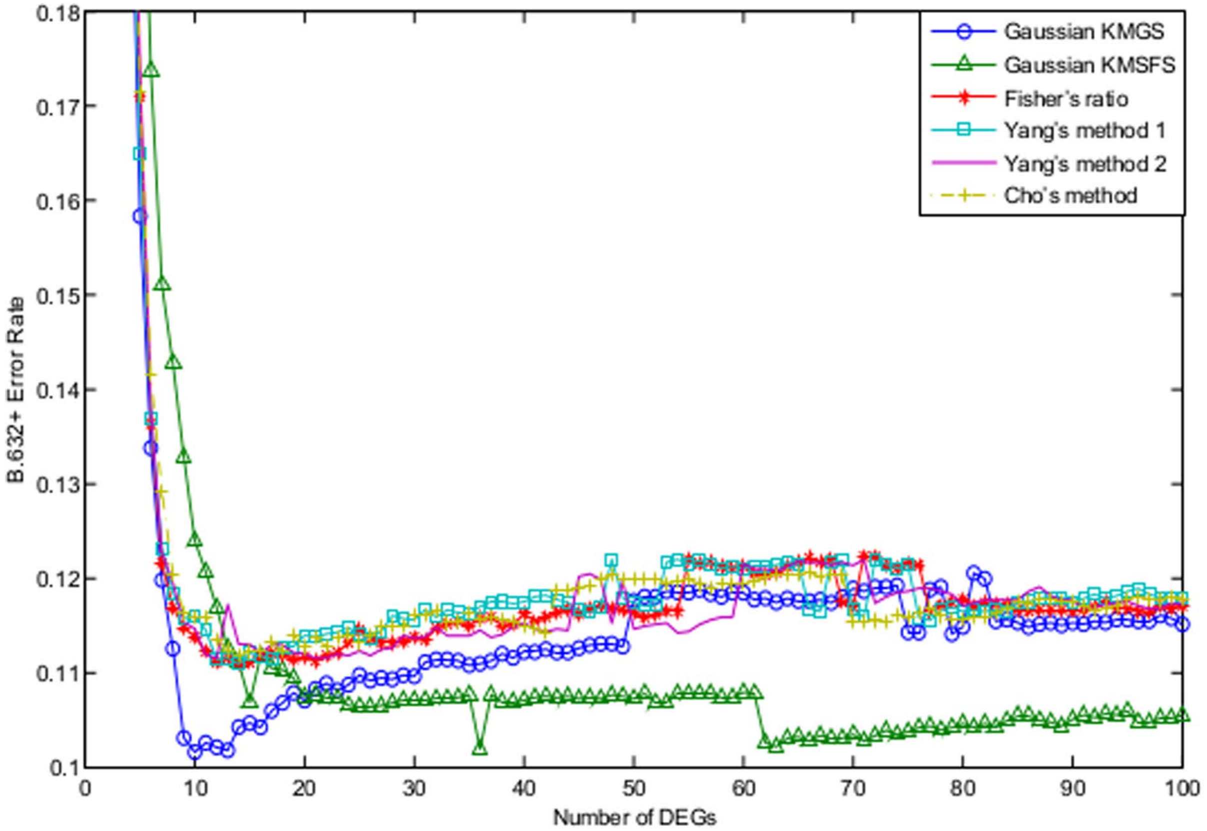


Figure 7. The B.632+ error shown as a function of the number of DEGs for the colon dataset. The curves are obtained from the filter methods among which are Fisher's ratio, Cho's method and Yang's methods with the parameter C uniformly set at 0.01. Gaussian KMSFS outperformed all the filter methods noticeably.
doi:10.1371/journal.pone.0081683.g007

As shown by Figure 4, the classical LOOSFS was outperformed by our algorithms including the Gaussian KMGS, linear KMSFS and linear KMGS. When the number of DEGs ranged between 10 and 20, Gaussian KMSFS also performed better than LOOSFS.

Comparisons against Filters. Figure 5 compares linear KMSFS, Gaussian KMGS against the filter methods of Fisher's ratio, Cho's method as well as the two methods of Yang's.

The error rate of linear KMSFS remained noticeably lower than the filter methods when the number of DEGs fell between 10 and 60. When the number of DEGs grew larger, Gaussian KMGS showed better than performance than the four filter methods.

The performance of Gaussian KMGS and linear KMSFS remained competitive to those of the filter methods, respectively between the value ranges of [0,60] and [60,100] for the number of DEGs.

These comparisons lead to the conclusion that linear KMSFS and Gaussian KMGS are the two best methods for the prostate dataset.

Results on the Colon Dataset

Optimal Parameter Settings. For Gaussian KMGS, Gaussian KMSFS and LOOSFS, the optimal settings on (λ,C) were respectively (1,0.01), (0.1,0.01) and (0.001,0.01).

For the other 3 wrapper methods which are respectively linear KMGS, linear KMSFS, GLGS and the 4 filter methods which are respectively Fisher's ratio, Cho's method and the two methods of Yang's, their optimal parameter settings were found to be $C=0.01$.

Comparisons against Wrappers. With a minimal error rate of 0.1577 and a mean error rate of 0.2100, GLGS performed much worse than the other nine methods. Thus again its simulation results were not graphically presented.

Figure 6 illustrates the the B.632+ error rates of Gaussian KMGS, Gaussian KMSFS, linear KMSFS, linear KMGS and LOOSFS. It can been seen that Gaussian KMSFS demonstrated the best performance while the LOOSFS the worst performance. Gaussian KMGS, linear KMSFS and linear KMGS also performed slightly better than LOOSFS, particularly when the number of DEGs ranged between 15 and 45.

It is interesting to note that Gaussian KMGS, with only 10 DEGs, reached the lowest B.632+ error rate which was approximately 0.10. Also the lowest B.632+ error rate of LOOSFS, which was 0.11 was lower than that reported in [21] which was around 0.15 on the colon data. We reckon it could be due to the employment of different data preprocessing strategies.

Comparisons against Filters. Figure 7 compares linear KMSFS and Gaussian KMGS against the filter methods of Fisher's ratio, Cho's method as well as the two methods of Yang's.

Gaussian KMSFS remained better than the 4 filter methods whose performances were comparable between each other. Meanwhile, the error rates of Gaussian KMGS were also lower than those of the filter methods, particularly for a smaller number of selected DEGs.

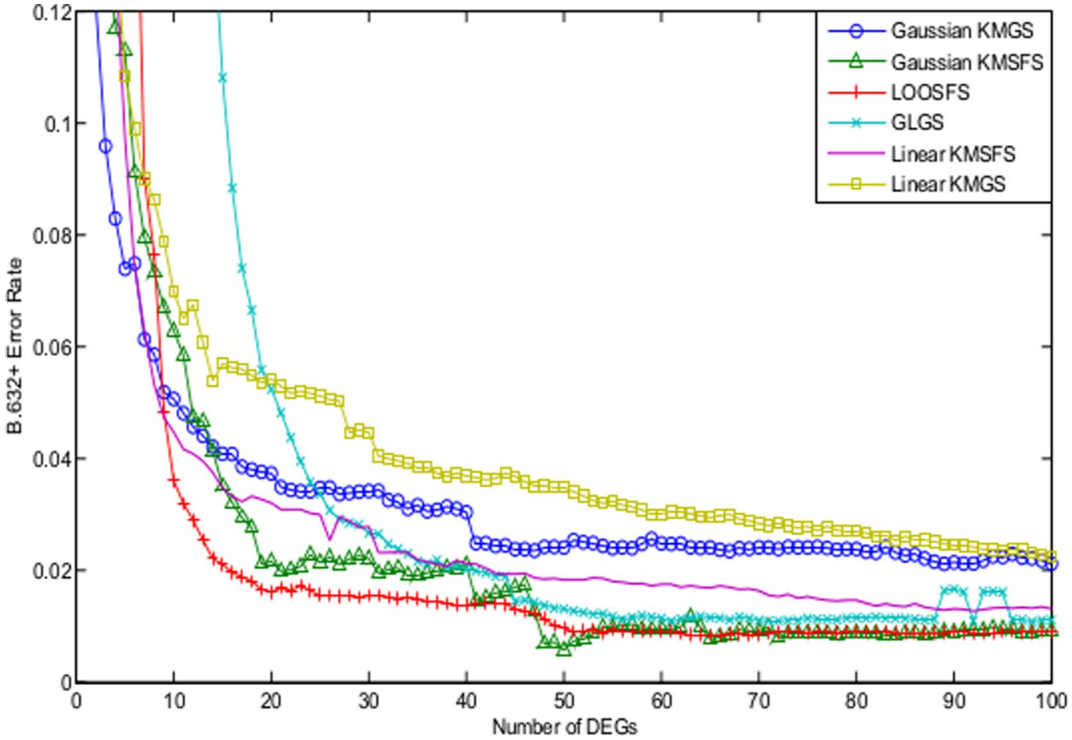


Figure 8. The B.632+ error shown as a function of the number of DEGs for the leukemia dataset. The curves depict the performance of the following wrapper methods with their respective optimal parameter settings: Gaussian KMGS with $\lambda = 10^4$ and $C = 0.1$; Gaussian KMSFS with $\lambda = 10^5$ and $C = 0.1$; LOOSFS with $\lambda = 0.1$ and $C = 0.01$; GLGS with $C = 0.01$; linear KMSFS and linear KMGS with $C = 0.1$. The performance of Gaussian KMSFS remained competitive to that of LOOSFS. Meanwhile, the lowest B.632+ error rate was achieved by Gaussian KMSFS with around 50 selected DEGs.
doi:10.1371/journal.pone.0081683.g008

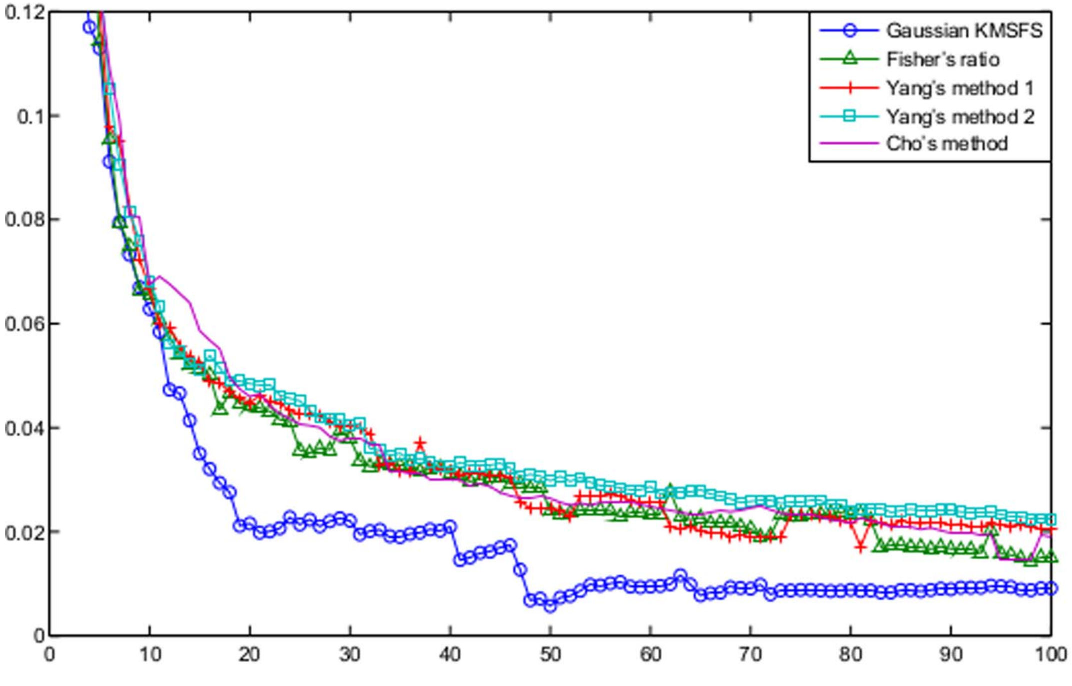
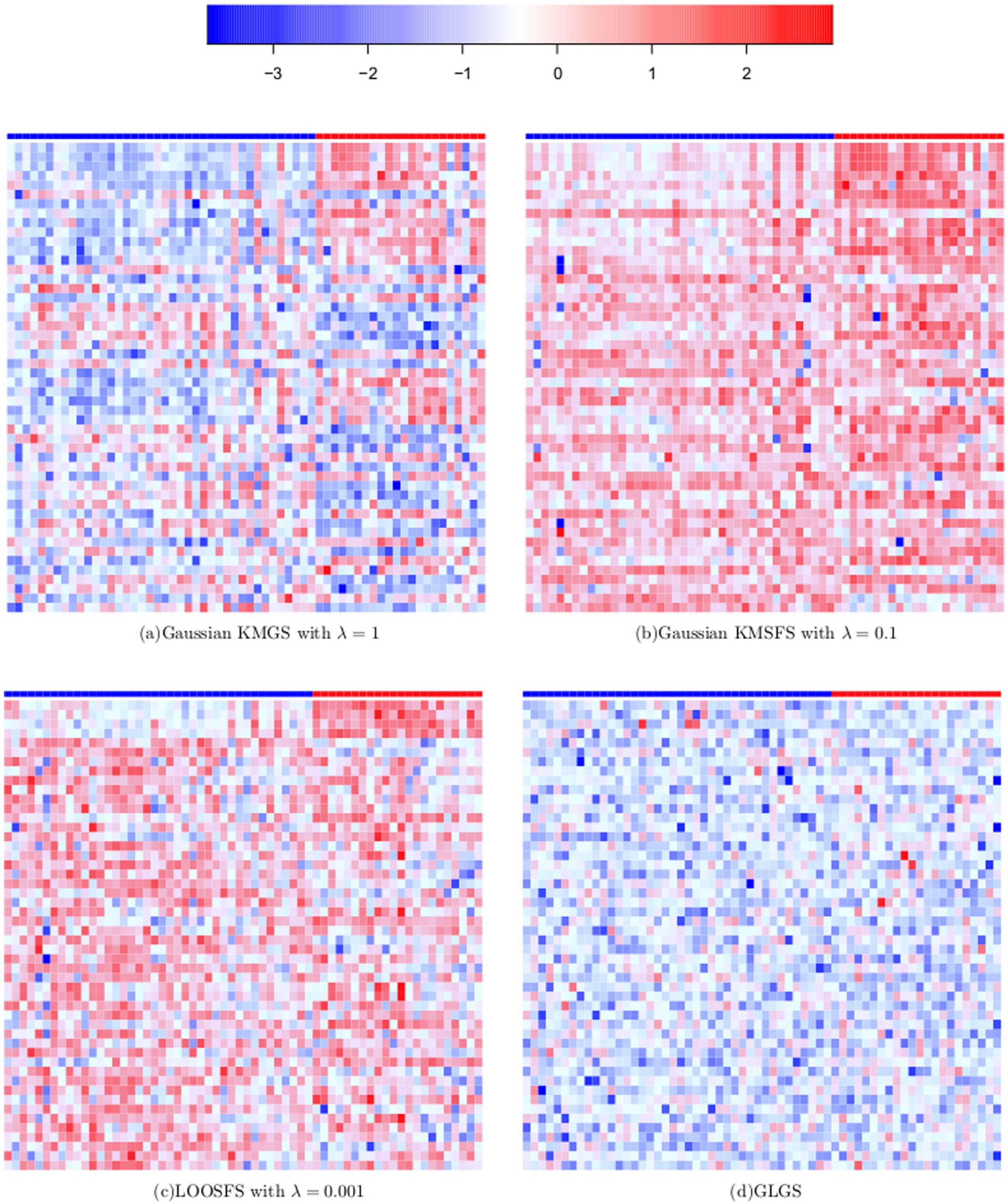


Figure 9. The B.632+ error shown as a function of the number of DEGs for the leukemia dataset. The curves are obtained from the filter methods among which are Fisher's ratio, Cho's method and Yang's methods with C set to be uniformly. Gaussian KMSFS performed better than the 4 filter methods.
doi:10.1371/journal.pone.0081683.g009



(a)Gaussian KMGS with $\lambda = 1$

(b)Gaussian KMSFS with $\lambda = 0.1$

(c)LOOSFS with $\lambda = 0.001$

(d)GLGS

Figure 10. Heatmaps of top 50 DEGs selected most frequently by Gaussian KMGS, Gaussian KMSFS, LOOSFS, GLGS respectively with their optimal parameter settings on λ.
doi:10.1371/journal.pone.0081683.g010

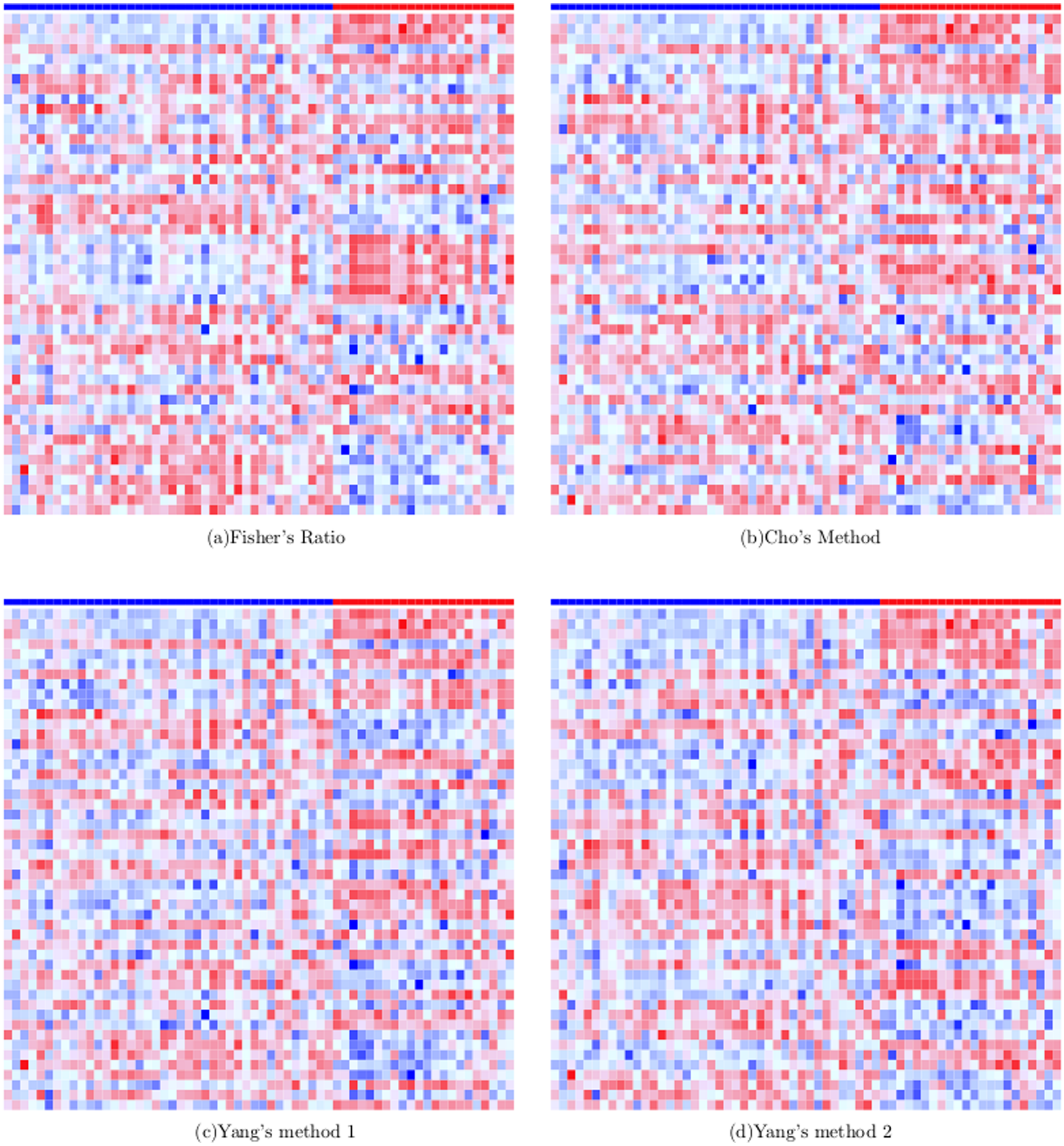


Figure 11. Heatmaps of top 50 DEGs selected most frequently by Fisher's ratio, Cho's methods and Yang's two methods.
doi:10.1371/journal.pone.0081683.g011

In conclusion, Gaussian KMSFS and Gaussian KMGS have proved to be the best methods for the colon dataset.

Results on the Leukemia Dataset

Optimal Parameter Settings. For Gaussian KMGS, Gaussian KMSFS and LOOSFS, the optimal setting on the parameter pair of (λ,C) were respectively $(10^4,0.1)$, $(10^5,0.1)$ and $(0.1,0.01)$.

For GLGS, the optimal setting was found to be $C=0.01$. For linear KMGS, linear KMSFS and all the filter methods which are respectively Fisher's ratio, Cho's method and the two methods of Yang's, their optimal parameter settings were uniformly $C=0.1$.

Comparisons against Filters. Figure 8 illustrates the the B.632+ error rates of Gaussian KMGS, Gaussian KMSFS, LOOSFS, GLGS, linear KMSFS and linear KMGS. The performance of LOOSFS depicted by Figure 8 was in fact

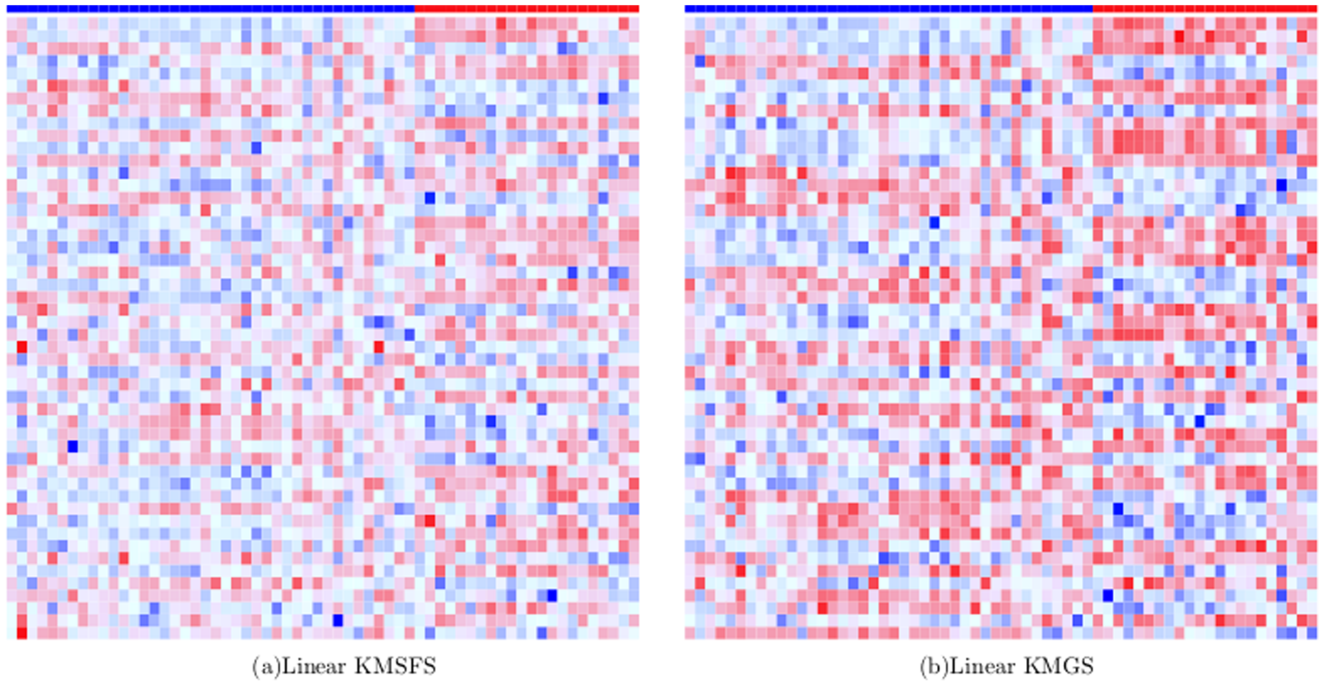


Figure 12. Heatmaps of top 50 DEGs selected most frequently by linear KMSFS and linear KMGS.
doi:10.1371/journal.pone.0081683.g012

consistent with that reported in [21]. The performance of Gaussian KMSFS remained competitive to that of LOOSFS. Meanwhile, the lowest B.632+ error rate was achieved by the Gaussian KMSFS with around 50 selected DEGs.

However, Gaussian KMGS, linear KMGS and linear KMSFS failed to perform as well as LOOSFS. We reckon it might be attributable to the preprocessing procedure which resulted in the removal of over 86% of the original 7029 genes, although this viewpoint has to be confirmed with more experiments.

Comparisons against Filters. Figure 9 further compares the performance of Gaussian KMSFS against the filter methods of Fisher's ratio, Cho's method as well as the two methods of Yang's. It was demonstrated that, the error rates of Gaussian KMSFS remained noticeably low than those of the 4 filter methods throughout.

We regarded the LOOSFS and the Gaussian KMSFS as the two best gene selection algorithms for the leukemia dataset.

Discussion

Heatmaps of Differentially Expressed Genes

Due to the employment of B.632+ error estimation technique, each gene selection algorithm was applied to the 200 sets of bootstrap samples as well as the original training set. For each of these 201 sample sets, we selected a sequence of 100 DEGs. This resulted in altogether 201 sets each of which contained 100 DEGs.

We calculated the frequency with which each of the 2000 genes was selected into the 201 sets of DEGs and drew the heatmaps of 50 DEGs that were selected most frequently. For the algorithms of Gaussian KMGS, Gaussian KMSFS and LOOSFS, the outcome of gene selection procedures is influenced by the value setting on the parameter λ and we used the optimal values reported in the previous section.

Heatmaps for the ten gene selection methods, were shown by Figure 10, Figure 11 and Figure 12. In each heatmap, each column corresponds to a sample and each row is the normalized expression values of a selected DEG across the 62 samples. A grid of each heatmap is colored according to the color key at the top of Figure 10 which maps a normalized expression value to a specific color between blue and red. The class of a sample at each column is indicated by the color bar at the top of each heatmap where blue indicates the cancerous case and red the normal case. Along the downward direction, the 50 DEGs are displayed in descending order of their frequency of occurrence in the 201 sets of selected genes.

It can be seen from Figure 11 that, the filter methods tend to favor "discriminant features" whose color forms an obvious contrast between the cancerous population and the normal population at each row. In Figure 10(b) which represents Gaussian KMSFS method, the color of genes at each row is in a pattern of "occasional dotting of red versus a majority of blue". The color contrast at each row of Figure 10(b) is less noticeable than Figure 11. Nevertheless, Gaussian KMSFS demonstrated the best prediction accuracies among all the methods, as reported in the previous section.

The second best gene selection algorithm for the colon data is Gaussian KMGS whose selected DEGs have been presented by Figure 10(a). Interestingly, between the two opposing classes, Figure 10(a) exhibited a sharper color contrast than the one exhibited by Figure 10(b).

The above facts suggest that, although filter methods selected genes whose values, in general, differ significantly between opposing classes, our kernel induced algorithms seemed not to hold it as the selection criterion. Instead, our methods endeavored to select genes that could establish a set of "discriminant samples" for SVMs. This possibly accounts for their superiority in terms of B.632+ error rates on the colon dataset.

Table 3. Scores obtained from Friedman rank sum tests with Holm correction for choices of λ with C fixed at a specific value at each row.

prostate data												
	$C\lambda$	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10^{1}	10^{2}	10^{3}	10^{4}	10^{5}
	10^{-3}	4.095	**4.035**	**4.12**	**4.265**	**5.47**	**4.585**	**5.42**	**6.555**	**8.45**	**9.285**	**9.72**
	10^{-2}	6.165	6.085	5.87	5.11	3.16	**1.78**	**3.15**	**5.67**	**8.2**	**9.92**	**10.89**
	10^{-1}	(*) 4.555	4.565	4.43	3.89	2.5	**1.85**	**6.45**	**8.13**	**8.63**	**10**	**11**
Gaussian	1	4.66	4.67	4.765	4.16	2.8	**1.92**	**5.455**	**7.85**	**8.72**	**10**	**11**
KMGS	10^{1}	4.95	4.995	4.835	4.38	2.62	**2.28**	**4.98**	**7.56**	**8.4**	**10**	**11**
	10^{2}	4.875	4.96	4.905	4.32	**2.4**	**2.45**	**5.02**	**7.51**	**8.56**	**10**	**11**
	10^{3}	4.82	4.895	4.855	4.25	**2.33**	**2.57**	**5.15**	**7.68**	**8.45**	**10**	**11**
	10^{-3}	3.82	3.73	**3.535**	**3.965**	**6.12**	**8.545**	**7.07**	**5.835**	**8.44**	**8.25**	**6.69**
	10^{-2}	2.74	3.015	3.34	4.065	**2.01**	**6.09**	**7.8**	**9.01**	**7.35**	**9.86**	**10.72**
	10^{-1}	(*) 3.86	4.325	3.96	2.405	**1.14**	**5.31**	**7**	**8.47**	**8.53**	**10**	**11**
Gaussian	1	4.08	4.32	4.055	2.385	**1.24**	**4.92**	**7**	**8**	**9**	**10**	**11**
KMSFS	10^{1}	3.965	4.2	3.79	2.165	**1.19**	**5.88**	**6.81**	**8**	**9**	**10**	**11**
	10^{2}	4.005	4.2	3.84	2.155	**1.21**	**6.01**	**6.58**	**8**	**9**	**10**	**11**
	10^{3}	4.005	4.195	3.815	2.085	**1.2**	**6.01**	**6.69**	**8**	**9**	**10**	**11**
colon data												
	$C\lambda$	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10^{1}	10^{2}	10^{3}	10^{4}	10^{5}
	10^{-3}	5.01	5.03	5.145	5.345	5.265	7	7.795	8.305	6.905	5.275	**4.925**
	10^{-2}	(*) 5.545	5.56	5.465	5.54	5.3	2.715	**2.62**	**4.97**	**8.21**	**9.695**	**10.38**
	10^{-1}	4.815	4.875	4.865	4.94	5.56	3.76	**2.69**	**4.81**	**9.345**	**9.96**	**10.38**
Gaussian	1	5.32	5.41	5.31	5.455	4.815	3.89	3.31	**2.94**	**9**	**10.225**	**10.325**
KMGS	10^{1}	5.32	5.39	5.215	5.44	4.955	3.91	**3.07**	**3.15**	**8.9**	**10.325**	**10.325**
	10^{2}	5.295	5.395	5.17	5.385	4.955	4	**3.07**	**3.18**	**8.91**	**10.345**	**10.295**
	10^{3}	5.31	5.395	5.2	5.39	4.925	4.01	**3.03**	**3.19**	**8.91**	**10.335**	**10.305**
	10^{-3}	7.47	7.81	7.73	7.525	8.855	**4.435**	**4.435**	**4.435**	**4.435**	**4.435**	**4.435**
	10^{-2}	(*) 3.8	3.985	4.185	2.89	**1.39**	**6.02**	**7.02**	**8.42**	**9.595**	**9.335**	**9.36**
	10^{-1}	3.23	3.645	4.23	2.77	**1.56**	**5.645**	**7.21**	**8.2**	**9.52**	**10.52**	**9.47**
Gaussian	1	2.735	3.23	3.83	**2.395**	**3.06**	**5.76**	**7.3**	**8.26**	**9.9**	**10.465**	**9.065**
KMSFS	10^{1}	2.655	3.155	3.755	**2.445**	**3.17**	**5.83**	**7.33**	**8.26**	**9.91**	**10.495**	**8.995**
	10^{2}	2.61	3.13	3.705	**2.535**	**3.24**	**5.79**	**7.33**	**8.26**	**9.92**	**10.505**	**8.975**
	10^{3}	2.6	3.13	3.705	**2.535**	**3.22**	**5.82**	**7.33**	**8.26**	**9.92**	**10.475**	**9.005**

doi:10.1371/journal.pone.0081683.t003

Parameter Sensitivity Analysis

For linear KMGS and linear KMSFS, their B.632+ error rates are affected by value settings on C. For Gaussian KMGS and Gaussian KMSFS, their B.632+ error rates are influenced by both parameters of λ and C.

Using the prostate dataset and the colon dataset, we employed the Friedman rank sum test with Holm correction to study the sensitivity of sample classification to value settings on C and λ respectively.

Sensitivity of Sample Classification to λ. We kept C fixed at a specific value and ran Friedman rank sum tests with Holm correction for various choices of λ. The results were given by Table 3 each row of which reports the score for different values on λ with C fixed at a specific value. The best choice is the one which obtained the lowest score and has been highlighted in bold for each row.

Prostate Dataset. We first analyzed the sensitivity of sample selection for Gaussian KMGS. At the row with $C=10^{-3}$, the setting of 10^{-4} on λ is significantly better than the choices of 10^{-1},10^{1},10^{2}, 10^{3},10^{4},10^{5} at confidence levels of both 0.95 and 0.99. It shows that sample classification is insensitive to λ only between 10^{-5} and 10^{-2} for $C=10^{-3}$. For the row with $C=10^{-2}$, $\lambda=1$ is significantly better than the rest at confidence levels of both 0.95 and 0.99.

Since the optimal value setting for C for Gaussian KMGS was found to be 0.1, we have labelled the associated row with an asterisk. It can be seen that, fixing C at 0.1, $\lambda=1$ is significantly better than all the rest, except for $\lambda=0.1$, at confidence levels of both 0.95 and 0.99. This suggests sample classification is insensitive to choices of λ between 0.1 and 1 for $C=0.1$. For $C=1$, $\lambda=1$ is significantly better than all the rest at confidence levels of 0.95 and 0.99. Thus sample classification is insensitive to choices of λ between 0.1 and 1 for $C=0.1$.

Table 4. Scores obtained from Friedman rank sum tests with Holm correction for choices of C with λ fixed at a specific value at each row.

prostate data														
	Gaussian KMGS							Gaussian KMSFS						
λC	10^{-3}	10^{-2}	10^{-1}	1	10^1	10^2	10^3	10^{-3}	10^{-2}	10^{-1}	1	10^1	10^2	10^3
10^{-5}	6.99	4.42	**1.26**	**2.99**	**3.96**	**4.12**	**4.26**	**5.57**	**1.58**	**2.28**	**4.32**	**4.58**	**4.805**	**4.865**
10^{-4}	6.99	4.42	**1.26**	**2.99**	**3.97**	**4.115**	**4.255**	**5.195**	**1.57**	**2.485**	**4.365**	**4.625**	**4.85**	**4.91**
10^{-3}	6.99	4.43	**1.26**	**2.98**	**3.94**	**4.13**	**4.27**	**5.27**	**1.57**	**2.29**	**4.425**	**4.705**	**4.84**	**4.9**
10^{-2}	6.99	4.49	**1.26**	**2.95**	**3.96**	**4.1**	**4.25**	**6.99**	**1.52**	**2.355**	**3.905**	**4.35**	**4.415**	**4.465**
10^{-1}	6.99	4.67	**1.22**	**3**	**3.8**	**4.105**	**4.215**	**(*)6.99**	**1.59**	**1.96**	**3.755**	**4.405**	**4.605**	**4.695**
1	(*)6.99	4.24	**1.12**	**3.115**	**4.075**	**4.15**	**4.31**	**6.98**	**6.02**	**1.44**	**1.89**	**3.98**	**3.8**	**3.89**
10^1	6.99	3.59	**1.45**	**3.325**	**4.105**	**4.265**	**4.275**	**6.93**	**6.07**	**2.52**	**1.88**	**3.34**	**3.525**	**3.735**
10^2	6.99	2.95	**1.4**	**3.61**	**4.27**	**4.315**	**4.465**	**6.93**	**6.07**	**2.49**	**2.18**	**2.94**	**3.635**	**3.755**
10^3	6.99	2.08	**1.65**	**4.03**	**4.31**	**4.465**	**4.475**	**6.89**	**5.12**	**1.92**	**2.52**	**3.44**	**3.965**	**4.145**
10^4	6.93	2.17	**1.63**	**3.495**	**4.295**	**4.625**	**4.855**	**6.63**	**6.11**	**1.8**	**2.58**	**3.32**	**3.72**	**3.84**
10^5	6.83	2.77	**1.68**	**3.2**	**4.11**	**4.615**	**4.795**	**6.19**	**5.95**	**1.6**	**2.375**	**3.67**	**3.985**	**4.23**
	Linear KMGS							Linear KMSFS						
NA	6.99	4.64	**1.16**	**3.195**	**3.985**	**3.97**	**4.06**	**4.92**	**1.36**	**3.15**	**4.45**	**4.63**	**4.725**	**4.765**
colon data														
	Gaussian KMGS							Gaussian KMSFS						
λC	10^{-3}	10^{-2}	10^{-1}	1	10^1	10^2	10^3	10^{-3}	10^{-2}	10^{-1}	10^0	1	10^2	10^3
10^{-5}	6.195	**1.295**	**2.07**	**4.395**	**4.585**	**4.695**	**4.765**	**6.96**	**1.4**	**2.36**	**4.185**	**4.385**	**4.32**	**4.39**
10^{-4}	6.195	**1.295**	**2.07**	**4.395**	**4.585**	**4.695**	**4.765**	**6.96**	**1.4**	**2.32**	**4.195**	**4.395**	**4.33**	**4.4**
10^{-3}	6.185	**1.295**	**2.08**	**4.395**	**4.595**	**4.69**	**4.76**	**6.96**	**1.4**	**2.42**	**4.17**	**4.37**	**4.305**	**4.375**
10^{-2}	6.235	**1.295**	**2.07**	**4.415**	**4.555**	**4.68**	**4.75**	**6.96**	**1.39**	**2.25**	**4.09**	**4.41**	**4.395**	**4.505**
10^{-1}	6.255	**1.295**	**2.05**	**4.28**	**4.59**	**4.735**	**4.795**	**6.96**	**1.39**	**2.79**	**4.125**	**4.205**	**4.245**	**4.285**
1	(*)6.96	**1.28**	**1.99**	**3.995**	**4.535**	**4.6**	**4.64**	**6.92**	**1.5**	**1.89**	**3.9**	**4.44**	**4.63**	**4.72**
10^1	6.915	**1.285**	**2.03**	**3.995**	**4.525**	**4.565**	**4.685**	**(*) 5.97**	**2.48**	**1.92**	**3.59**	**4.49**	**4.72**	**4.83**
10^2	6.855	**1.415**	**2.85**	**3.84**	**4.2**	**4.37**	**4.47**	**5.035**	**2.605**	**1.95**	**4.035**	**4.605**	**4.845**	**4.925**
10^3	5.735	**1.615**	**2.055**	**4.105**	**4.65**	**4.845**	**4.995**	**4.61**	**2.9**	**1.91**	**4.125**	**4.665**	**4.87**	**4.92**
10^4	5.555	2.225	**1.93**	**4.025**	**4.665**	**4.73**	**4.87**	**4.62**	**2.86**	**1.81**	**4.215**	**4.645**	**4.87**	**4.98**
10^5	5.52	2.41	**1.99**	**4.055**	**4.545**	**4.685**	**4.795**	**5.325**	**2.685**	**2.1**	**3.95**	**4.56**	**4.64**	**4.74**
	Linear KMGS							Linear KMSFS						
NA	6.33	**1.3**	**2.1**	**3.975**	**4.615**	**4.785**	**4.895**	**6.95**	**1.31**	**2.9**	**4.015**	**4.235**	**4.24**	**4.35**

doi:10.1371/journal.pone.0081683.t004

For the other rows for which C was fixed at 10, 10^2 and 10^3 respectively, sample classification remained insensitive to choices of λ between 0.1 and 1.

We can conclude that B.632+ error rates are insensitive to choices of λ between 0.1 and 1 in terms of Gaussian KMGS.

At the first row for Gaussian KMSFS, we can see that 10^{-3} is significantly better than 10^{-1} and other larger settings on λ. At the next row, $\lambda = 0.1$ is significantly better than the rest, exclusive of $\lambda = 10^{-5}$, at confidence levels of 0.95. $\lambda = 10^{-4}$ was also excluded at the confidence level of 0.99. The row of $C = 0.1$ has been labelled with an asterisk, indicating that 0.1 is the optimal choice for C for Gaussian KMSFS. And we can see that $\lambda = 0.1$ is significantly better than the other settings at confidence levels of both 0.95 and 0.99. This is also the case with the row corresponding to $C = 1$.

For the remaining three rows whose C was fixed at respectively 10, 10^2 and 10^3, uniformly, $\lambda = 10^{-1}$ is significantly better than the rest at the confidence level of 0.95.

Thus, for Gaussian KMSFS, B.632+ error rates are sensitive to choice of λ, when C was fixed at 0.1 and 1.

Colon Dataset. Similar analysis can be performed for the colon dataset, whose scores for various λ produced by the Friedman tests have been reported at the bottom half of Table 3.

Regarding Gaussian KMGS, we can see that B.632+ error rates are insensitive to choices of λ between 10 and 100 when C goes from 1 to 10^3 in multiples of 10. For $C = 10^{-2}$ which is its optimal setting, B.632+ rates are insensitive to choices of λ between 0.1 and 1.

In terms of Gaussian KMSFS, we can see that sample classification is sensitive to λ at the rows of $C = 10^{-2}$ and $C = 10^{-1}$. For $C \in \{1, 10^2, 100^3\}$, it shows that B.632+ error rates are insensitive to λ at $\lambda = 10^{-5}$ and $\lambda = 10^{-2}$. Larger values for λ caused severe performance degradation, as suggested by the scores at the bottom right in Table 3.

Sensitivity of Sample Classification to *C*. We then kept λ fixed at a specific value and ran Friedman rank sum tests with Holm correction for varied C's. The results were given by Table 4 each row of which reports scores for various choices of C at a specific value setting on λ. The best choice at each row is the one with the lowest score and has been highlighted in bold.

Prostate Dataset. For Gaussian KMGS, we can see that B.632+ error rates are sensitive to choices of C. For different λ's, $C = 0.1$ remained the setting that linear SVMs achieved the best performance.

For Gaussian KMSFS, when λ grows from 10^{-5} to 10^{-1}, B.632+ error rates remaine sensitive to λ and the best performance was always achieved at $\lambda = 10^{-2}$. As C continues to grow larger, B.632+ error rates appear to be insensitive between 0.1 and 1, which is particularly true with the row corresponding to $\lambda = 1$.

C is the sole parameter for linear KMGS and linear KMSFS. We can also see from Table 4 that, for both algorithms, B.632+ error rates are sensitive to C.

Colon Dataset. Table 4 indicates that, for both Gaussian KMGS and Gaussian KMSFS, the classification performance is insensitive to choices of C between 0.01 and 0.1, for values of λ greater than 1. Nevertheless, for smaller values on λ with both algorithms, B.632+ error rates are sensitive to C.

In term of both linear KMGS and linear KMSFS, B.632+ error rates remain sensitive to C, as with the results on the prostate dataset.

Conclusions

Statistical tests select genes whose expression values differ significantly between the two opposing classes, i.e., the discriminant genes. Samples-based learning machines including SVMs favor genes which results in a set of discriminant samples. The discriminant genes can not be guaranteed to result in a set of discriminant samples. We have shown that the genes leading to discriminant training samples can be detected by applying statistical tests to the kernel matrix.

In addition to the competitive performance demonstrated on the three public microarray datasets, the proposed kernel matrix induced gene selection algorithms offer extra advantages:

1. **Generality**. Our methods are considered applicable to any kernel classifiers, not just SVMs.
2. **Flexibility**. For the implementation of our methods, users can opt for any mercer kernel which can be linear, Gaussian RBF, sigmoid, or polynomials. However, depending on the specific kernel, properly-designed preprocessing steps may be required. For examples, Gaussian RBF kernels require the tuning of the width parameter. For linear kernels, strategies are required to ensure the diagonal elements of the kernel matrix, each of which suggests the similarity between a sample and itself, to be uniformly one.

It is also worth attention that microarray datasets have usually been assumed to present linear problems. However, it is unlikely to be true in the case that the number of DEGs is as few as one. Interestingly, linear problems can be solved by SVMs with nonlinear kernels, while nonlinear problems are hardly solvable with SVMs using linear kernels. A potential solution to the nonlinear problem posed by a small number of DEGs could be the application of nonlinear SVM classifiers. But our method suggested a successful alternative which is the use of the nonlinear Gaussian RBF kernel for the identification of DEGs. We reckon that this effort of instilling ”nonlinearity” into the identification of DEGs has contributed to the better empirical performance of our methods.

Supporting Information

Additional File S1 The three microarray datasets (PROSTATE, COLON, and LEU) in MATLAB format which were used in this work.
(ZIP)

Acknowledgments

The authors would like to thank the reviewer and the Editor for their helpful comments.

Author Contributions

Conceived and designed the experiments: XLX. Performed the experiments: HX. Analyzed the data: XL. Contributed reagents/materials/analysis tools: XLX. Wrote the paper: XLX.

References

1. Tusher V, Tibshirani R, Chu G (2001) Significance analysis of microarrays applied to the ionizing radiation response. Proceedings of the National Academy of Sciences 98: 5116–5121.
2. Baldi P, Long A (2001) A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inferences of gene changes. Bioinformatics 17: 509–519.
3. Lönnstedt I, Speed T (2002) Replicated microarray data. Statistica sinica 12: 31–46.
4. Dudoit S, Yang Y, Callow M, Speed T (2002) Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. Statistica Sinica 12: 111–140.
5. Holm S (1979) A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics: 65–70.
6. Westfall P, Young S (1993) Resampling-Based Multiple Testing: Examples and Methods for p-Value Adjustment. Wiley-Interscience.
7. Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society Series B (Methodological): 289–300.
8. Cui X, Hwang JG, Qiu J, Blades NJ, Churchill GA (2005) Improved statistical tests for differential gene expression by shrinking variance components estimates. Biostatistics 6: 59–75.
9. Cui X, Churchill G (2003) Statistical tests for differential expression in cDNA microarray experiments. Genome Biology 4: 210.
10. Bo T, Jonassen I (2002) New feature subset selection procedures for classification of expression profiles. Genome Biology 3: 0017.
11. Yeoh E, Ross M, Shurtleff S, Williams W, Patel D, et al. (2002) Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. Cancer Cell 1: 133–143.
12. Wang Y, Tetko I, Hall M, Frank E, Facius A, et al. (2005) Gene selection from microarray data for cancer classificationa machine learning approach. Computational Biology and Chemistry 29: 37–46.
13. Gevaert O, Smet F, Timmerman D, Moreau Y, Moor B (2006) Predicting the prognosis of breast cancer by integrating clinical and microarray data with Bayesian networks. Bioinformatics 22.
14. Xing E, Jordan M, Karp R (2001) Feature selection for high-dimensional genomic microarray data. In: Proceedings of the Eighteenth International Conference on Machine Learning. Morgan Kaufmann, pp.601–608.

15. Mamitsuka H (2006) Selecting features in microarray classification using ROC curves. Pattern Recognition 39: 2393–2404.
16. Ding C, Peng H (2005) Minimum redundancy feature selection from microarray gene expression data. Journal of Bioinformatics and Computational Biology 3: 185–206.
17. Yeung K, Bumgarner R (2003) Multiclass classification of microarray data with repeated measurements: application to cancer. Genome Biology 4: R83.
18. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. The Journal of Machine Learning Research 3: 1157–1182.
19. Devijver P, Kittler J (1982) Pattern Recognition: A Statistical Approach. Prentice Hall.
20. Kohavi R, John G (1997) Wrappers for feature subset selection. Artificial Intelligence 97: 273–324.
21. Zhou X, Mao K (2005) LS bound based gene selection for DNA microarray data. Bioinformatics 21: 1559–1564.
22. Tang E, Suganthan P, Yao X (2006) Gene selection algorithms for microarray data based on least squares support vector machine. BMC Bioinformatics 7: 95.
23. Jirapech-Umpai T, Aitken S (2005) Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes. BMC bioinformatics 6: 148.
24. Li L, Weinberg C, Darden T, Pedersen L (2001) Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. Bioinformatics 17: 1131–1142.
25. Ooi C, Tan P (2003) Genetic algorithms applied to multi-class prediction for the analysis of gene expression data. Bioinformatics 19: 37–44.
26. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Machine learning 46: 389–422.
27. Díaz-Uriarte R, de Andrés A (2006) Gene selection and classification of microarray data using random forest. BMC bioinformatics 7: 3.
28. Jiang H, Deng Y, Chen H, Tao L, Sha Q, et al. (2004) Joint analysis of two microarray geneexpression data sets to select lung adenocarcinoma marker genes. BMC bioinformatics 5: 81.
29. Saeys Y, Inza I, Larranaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 23: 2507.
30. Inza I, Larrañaga P, Blanco R, Cerrolaza AJ (2004) Filter versus wrapper gene selection approaches in dna microarray domains. Artificial intelligence in medicine 31: 91–103.
31. Pan W (2002) A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. Bioinformatics 18: 546–554.
32. Slonim D (2002) From patterns to pathways: gene expression data analysis comes of age. Nature Genetics: 502–508.
33. Statnikov A,Wang L, Aliferis C (2008) A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. BMC Bioinformatics 9: 319.
34. Brown M, Grundy W, Lin D, Cristianini N, Sugnet C, et al. (2000) Knowledge-based analysis of microarray gene expression data by using support vector machines. Proceedings of the National Academy of Sciences 97: 262–267.
35. Shipp M, Ross K, Tamayo P,Weng A, Kutok J, et al. (2002) Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. Nature Medicine 8: 68–74.
36. Furey T, Cristianini N, Duffy N, Bednarski D, Schummer M, et al. (2000) Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics 16: 906–914.
37. Cristianini N, Shawe-Taylor J (2000) An introduction to support Vector Machines: and other kernel-based learning methods. Cambridge Univ Pr.
38. Burges C (1998) A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2: 121–167.
39. Huang DS (1999) Radial basis probabilistic neural networks: Model and application. International Journal of Pattern Recognition and Artificial Intelligence 13: 1083–1101.
40. Zhou X, Mao K (2006) The ties problem resulting from counting-based error estimators and its impact on gene selection algorithms. Bioinformatics 22: 2507–2515.
41. Dudoit S, Fridlyand J, Speed T (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97: 77–87.
42. Ambroise C, McLachlan G (2002) Selection bias in gene extraction on the basis of microarray gene-expression data. Proceedings of the National Academy of Sciences 99: 6562.
43. Simon R, Radmacher M, Dobbin K, McShane L (2003) Pitfalls in the use of DNA microarray data for diagnostic and prognostic classification. Journal of the National Cancer Institute 95: 14–18.
44. Braga-Neto U, Dougherty E (2004) Is cross-validation valid for small-sample microarray classification? Bioinformatics 20: 374–380.
45. Pavlidis P, Weston J, Cai J, Grundy W (2001) Gene functional classification from heterogeneous data. In: Proceedings of the 5th Annual International Conference on Computational Biology. ACM New York, USA , pp.249–255.
46. Cho JH, Lee D, Park JH, Lee IB (2003) New gene selection method for classification of cancer subtypes considering within-class variation. FEBS letters 551: 3–7.
47. Yang K, Cai Z, Li J, Lin G (2006) A stable gene selection in microarray data analysis. BMC Bioinformatics 7: 228.
48. Chang CC, Lin CJ (2011) Libsvm: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2: 27.