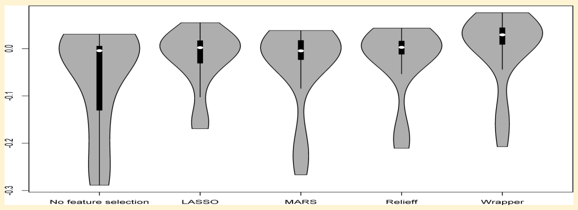
Choosing Feature Selection and Learning Algorithms in QSAR

Martin Eklund,[†,‡] Ulf Norinder,[¶] Scott Boyer,[‡] and Lars Carlsson*[,‡]

[†]Department of Pharmaceutical Biosciences Uppsala University, P.O. Box 591, SE-751 24 Uppsala, Sweden
[‡]AstraZeneca Research and Development, Pepparedsleden 1, SE-431 83 Mölndal, Sweden
[¶]H. Lundbeck A/S, Ottiliavej 9, DK-2500 Valby, Denmark



ABSTRACT: Feature selection is an important part of contemporary QSAR analysis. In a recently published paper, we investigated the performance of different feature selection methods in a large number of in silico experiments conducted using real QSAR datasets. However, an interesting question that we did not address is whether certain feature selection methods are better than others in combination with certain learning methods, in terms of producing models with high prediction accuracy. In this report we extend our work from the previous investigation by using four different feature selection methods (wrapper, ReliefF, MARS, and elastic nets), together with eight learners (MARS, elastic net, random forest, SVM, neural networks, multiple linear regression, PLS, *k*NN) in an empirical investigation to address this question. The results indicate that state-of-the-art learners (random forest, SVM, and neural networks) do not gain prediction accuracy from feature selection, and we found no evidence that a certain feature selection is particularly well-suited for use in combination with a certain learner.

■ INTRODUCTION

Feature selection is an important part of contemporary quantitative structure−activity relationship (QSAR) analysis, because of the often very large number of features in QSAR models, as well as the common requirement of model interpretability.[1] In a large number of in silico experiments conducted using real QSAR datasets, we recently investigated the performance of different feature selection methods (multivariate adaptive regression splines (MARS), lasso, random forest importance selection, univariate methods, backward elimination, forward selection, and a wrapper approach; see ref 2 for details). In summary, the conclusions of this work was:

(1) Using random forest as the learning method, we did not observe statistically significant improvements in prediction performance for any feature selection method, compared to when no feature selection was used.
(2) However, feature selection could reduce the number of variables in the QSAR datasets by up to 60% without a significant deterioration of the model's estimated prediction performance, indicating that feature selection enables better model transparency.
(3) There were clear differences in the performance of the studied feature selection methods across the experiments. MARS, lasso, and forward selection performed well, whereas univariate methods performed rather poorly.

These conclusions support the claim that random forest is well-suited for modeling datasets with a large number of features. However, an interesting question that we did not address in our previous paper is whether certain feature selection methods are better suited than others in combination with certain learning methods. Or put in another way: Are there interaction effects between choice of feature selection method and choice of learning method on the predictive performance of the resulting QSAR model? In the work presented herein, we address this question by empirical investigations on five real-world QSAR datasets.

Relatively little has been published on the combined impact of choices of feature selection method and learning method on the predictive performance of QSAR models. Goodarzi et al.[3] studied the use of three feature selection methods (a genetic algorithm, a successive projections algorithm, and fuzzy rough set ant colony optimization), in combination with three learners (multiple linear regression, artificial neural network, and support vector machines) on modeling the bioactivities of a series of 161 glycogen synthase kinase-3β inhibitors, and came to the conclusion that fuzzy rough set ant colony optimization in combination with support vector machine (SVM) worked best. Liu[4] studied a set of univariate feature selection methods (information gain, mutual information, the χ^2 statistic, odds ratio, and the GSS coefficient (essentially a simplified χ^2 statistic)), together with the Naïve Bayes classifier and SVM. Their results showed that Naïve Bayes benefited significantly from feature selection, while SVM performed better when all features were used, likely because of the fact that SVM handles a large number of features by penalization.

Outside the realm of QSAR, there is also a relatively limited set of papers studying the interaction between choice of feature selection method and learning method. Brank et al.[5] and Mladenić et al.[6] studied the effect of using three linear or univariate variable selection methods (odds ratio, information gain, and parameter size in a linear classifier) in combination

Received: October 4, 2013
Published: January 26, 2014

with three different learners (Naïve Bayes, the perceptron, and SVM) on the performance of document classification. Their conclusion was that more-complex (multivariate) feature scoring methods work better than simpler methods (univariate). Lu et al.[7] came to a similar conclusion when studying the Naïve Bayes classifiers, SVM, and stochastic gradient boosting trees in combination with feature selection by information gain, SVM feature ranking, or boosting feature selection for discriminating between navigational and non-navigational web searches. The previous study that comes closest in aim and scope to the work presented here is the work by Hall and Holmes,[8] who investigated six different feature selection methods (information gain, ReliefF, principal components, correlation-based feature selection, consistency, and wrapper) together with two learners (C4.5 and Naïve Bayes). The conclusion by Hall and Holmes was that, generally, feature selection is beneficial for improving the predictive performance of the learners and that some interaction effects between learners and feature selection methods tentatively could be observed.

In the work presented here we extend and complement the work reviewed in the preceding paragraphs. First, we will use MARS and lasso (on top of a wrapper approach and ReliefF), which were not studied in any of the other papers. Second, we study a larger number of state-of-the art learners. Third, we will use a larger set of QSAR-specific datasets, which forms a more relevant testbed for the QSAR application domain. And lastly, except for Hall and Holmes,[8] none of the previous papers conducted a statistical analysis of the results in order to detect general trends.

■ METHODS

Our main interest in this paper is to employ feature selection to produce as accurate predictions as possible, as opposed to producing as transparent (small) models as possible. Given a dataset, we want to estimate the subset of features that produces the learner with the best predictive performance. More formally, let $\mathbf{D} = (\mathbf{y},\mathbf{X})$ be an $n \times (p + 1)$ data matrix and let $\gamma \in \{1, ..., 2^p\}$ index the subsets of $\mathbf{X}$. Let p_γ be the size of the γth subset. Our aim is to estimate a function $y = g_\gamma(\mathbf{x}_\gamma)$, according to

$$\hat{g}_\gamma = \underset{g_\gamma \in \mathcal{G}_\gamma, \gamma \in \{1,...,2^p\}}{\arg\min} \sum_{i=1}^{n} L(y_i - g(\mathbf{x}_{\gamma,i})) \tag{1}$$

where L is a loss function (mean squared error for regression and 0−1 loss for classification), and $\mathcal{G}_\gamma$ is a set of possible functions.

Table 1. Overview of the Datasets Used in the Benchmarking Experiments[a]

dataset	end point	n(class)	n(reg)	ref
COX2	cyclooxygenase-2	322	322	22
CPD	carcinogenic categorical activity in rats	1198	567	23
DHFR	dihydrofolate reductase	397	397	22
EPAFHM	fathead minnow acute toxicity	577	577	24
FDA	FDA maximum recommended daily dose	1216	1216	25

[a] n(class) and n(reg) indicate the number of compounds in each dataset with a classification (class) response and a regression (reg) response, respectively.

Feature Selection Methods. Feature selection methods can be grouped into three categories: *filters*, *wrappers*, and *embedded* methods.[9] Filters rank the available features according to some criterion (e.g., a *t*-test) and remove features that are deemed irrelevant for modeling the response. Wrapper methods combine feature selection with learning by searching the space of possible feature subsets and fitting a learner to each subset. The best subset is then directly estimated by the subset that produces the most accurate learner (according to some criterion, such as eq 1). In embedded feature selection, the search for an optimal subset of features is built into the learning method, usually by appropriately chosen constraints on the parameter sizes (see, e.g., eq 5 and Hastie et al.[10]).

In Eklund et al.,[2] we found MARS and lasso to be the feature selection methods that performed best among the methods included in the benchmarking experiments. Therefore, we use these feature selection methods here (or rather, we use a generalization of lasso: the elastic nets). Lasso/elastic nets belong to the class of embedded methods, whereas MARS can be regarded both as an embedded method and a wrapper. Therefore, we complement these methods with a clear-cut wrapper approach, as well as a filter method (ReliefF),[11] both of which have produced good results in previous investigations.[8] We also use all learners without feature selection as a baseline assessment. Below, we give a short description of each of the four feature selection methods included in the experiments.

MARS. MARS can be viewed as a generalization of stepwise linear regression

$$f(\mathbf{x}) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(\mathbf{x}) \tag{2}$$

where $\beta_0, ..., \beta_M$ are estimated by minimizing the residual sum of squares and $h(x)$ are basis functions of the type $\max(0,\mathbf{x} - c)$ and $\max(0,c - \mathbf{x})$ or products of such functions (c is a parameter). The number of basis functions is determined by a greedy *forward pass*, where basis functions are added until a present maximum number of terms have been added, and a *backward pass*, where the model is pruned by minimizing a generalized cross-validation criterion:

$$\mathrm{GCV}(\lambda) = \sum_{i=1}^{n} \frac{(y_i - f_\lambda(\mathbf{x}_i))^2}{(1 - M(\lambda)/n)^2} \tag{3}$$

Here, $M(\lambda)$ is the effective number of parameters in the model and λ is a penalty term controlling the size of the model. All variables are typically not included in the MARS model after the forward pass and the backward pass, and MARS can thus be used both as a learner and for feature selection (and then combined with another learning algorithm).

Elastic Nets. Elastic nets[12] predict the response of a vector of features according to the standard linear model

$$f(\mathbf{x}) = \beta_0 + \mathbf{x}^{\mathrm{T}}\beta \tag{4}$$

where β_0 is estimated by $\bar{y}$ and $\beta_1, ..., \beta_p$ by a penalized least-squares estimator

$$\hat{\beta}^{\mathrm{en}} = \underset{\beta}{\mathrm{argmin}}\Big[\sum_{i=1}^{n} (y_i - \beta_0 - \mathbf{x}_i^{\mathrm{T}}\beta)^2 + \lambda \sum_{j=1}^{p} (\alpha\beta_j^2 + (1 - \alpha)|\beta_j|)\Big] \tag{5}$$

Because of the nature of the penalty term, some coefficients will be shrunk exactly to zero for a sufficiently large λ (see, e.g., Tibshirani[13] for a geometric argument to why this happens). In a manner analogous to that used with MARS, we can therefore use the elastic nets both as a learning method and for feature selection. Note that the elastic nets contain both the ridge estimator and the lasso as a special cases ($\alpha = 1$ and $\alpha = 0$, respectively).

Wrapper. The wrapper approach to feature selection uses a search algorithm (e.g., a genetic algorithm[14] or a particle swarm optimizer[2]) to traverse the space of possible feature subsets and evaluates each subset by evaluating a given learner's performance using the variables in the subset.

ReliefF. The idea of the ReliefF method is to randomly draw observations from the dataset, compute the observations' nearest neighbors, and recalibrate a feature weighting vector to give more weight to features that discriminate the observation from neighbors of different classes. In the classification setting, it can be shown that ReliefF estimates

$$\begin{aligned} w_j &= P(\text{different value of } \mathbf{x}_j | \text{ nearest obs. from diff. class}) \\ &\quad - P(\text{different value of } \mathbf{x}_j | \text{ nearest obs. from same class}) \end{aligned} \tag{6}$$

to assign as the weight for each feature $\mathbf{x}_j$, $j = 1, ..., p$.[15]

For regression problems, eq 6 is modified so that the two terms estimate the probabilities that the predicted values of two observations are different (see Robnik-Šikonja and Kononenko[11] for an extensive discussion on this and for a good overview of the different ReliefF algorithms).

Learning Methods. In this section, we very briefly give an overview of the different learning methods used in the experiments. We also give key references and describe the settings and parameter choices that we used for each method.

Multiple Linear Regression. Multiple linear regression employs the linear model (eq 4) and estimates $\hat{\beta}_0, ..., \hat{\beta}_p$ by minimizing the ordinary least-squares estimates, i.e.,

$$\hat{\beta}^{\text{ols}} = \underset{\beta}{\text{argmin}} \sum_{i=1}^{n} (y_i - \beta_0 - \mathbf{x}_i^{\text{T}}\beta)^2 \tag{7}$$

Elastic Net. The elastic net estimator was described above. As can be seen from eqs 5 and 7, the elastic net is a penalized version of the ordinary least-squares estimator.

Partial Least-Squares. Partial least-squares regression[16] is an extension of the multiple linear regression model. The idea behind partial least-squares (commonly abbreviated PLS) is to produce a small number of linear combinations of the original features and use these linear combinations as input into the regression model. The rationale that justifies this idea is that a small number of latent (hidden) features are assumed to account for most of the variance in the dataset. What distinguishes PLS from principal component regression is that, in PLS, the features are weighted by the strength of their univariate effect on $\mathbf{y}$ in the construction of each latent feature. In other words, the relationship by the features and $\mathbf{y}$ is taken into consideration when estimating the hidden features, thus potentially producing latent features that are more relevant for prediction.

A closed form solution for PLS does not exist and numerical methods are used instead. We refer to Hastie et al.,[10] Wold,[16] and Gustafsson[17] for a more-detailed description of the PLS method and different views on the algorithm to produce the PLS estimates.

MARS. MARS was described above.

Random Forest. Random forest[18] grows a forest of decision trees, where each tree is constructed using a different bootstrap sample of the data (the observations not selected in a given bootstrap iteration are termed the out-of-bag samples and can be used to estimate prediction accuracy). Each node in each decision tree is split using the best among a subset of features randomly chosen at that node. Predictions from the forest are made by averaging over all predictions from the individual trees:

$$f_{\text{rf}}^{B}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} T(\mathbf{x}, \Theta_b) \tag{8}$$

where $T(x,\Theta_b)$ is the bth tree.

The idea in random forest is to reduce the variance of the individual predictions of a set of (approximately) unbiased, but noisy, decision trees by averaging. The random selection of features used for node splitting is done to improve the variance reduction by reducing the correlation between the trees, without excessively increasing the variance.

Support Vector Machine (SVM). A SVM[19] fits a model of the type

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + \beta_0 \tag{9}$$

where $K(\mathbf{x},\mathbf{x}_i)$ is a kernel function. The α_i are estimated by maximizing the Lagrangian

$$L = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_j' y_i y_j' K(\mathbf{x}_i, \mathbf{x}_j') \tag{10}$$

subject to $0 \leq \alpha_i \leq C$, where C is the cost parameter that determines the amount of regularization.

We used the following radial basis function (RBF) as a kernel function:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2) \tag{11}$$

with $\gamma = 4 \times 10^{-3}$. C was estimated in a cross-validated grid search, where C was allowed to take the values 1, $\sqrt{10}$, and 10.

Neural Network. Neural networks are systems of interconnected *neurons* that can compute values from inputs by feeding information through the network. The neurons are simple computing units, connected together in layers to form a network which mimics a biological neural network. There exists a large number of different neural network architectures. We used a single-hidden-layer neural network with a sigmoid activation function:

$$\mathbf{z}_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^{\text{T}}\mathbf{x})} \qquad m = 1, ..., M \tag{12}$$

$$\mathbf{w}_k = \beta_{0k} + \beta_k^T \mathbf{z} \qquad k = 0, 1 \tag{13}$$

$$f(\mathbf{w}_0) = \frac{\exp(\mathbf{w}_0)}{\exp(\mathbf{w}_0) + \exp(\mathbf{w}_1)} \tag{14}$$

(Equations 12–14 show the neural network for classification. Equation 14 is not needed for regression, and eq 13 is changed so that there is only a single k.) The parameters were estimated using penalized residual sum of squares (often called *weight decay* in the neural network literature).

***k*-Nearest Neighbors.** The *k*-nearest neighbors (*k*NN) estimate the response of a vector of features, according to

$$f(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{x}_i \in N_k(\mathbf{x})} y_i \tag{15}$$

where $N_k(\mathbf{x})$ is the neighborhood defined by the k closest observations in the training set. We here use the Euclidean distance to determine "closeness".

Optimization Method. Particle swarm optimization (PSO)[20] optimizes a problem by iteratively improving a candidate solution with regard to a given objective function. The optimization of a problem is performed by having a population of candidate solutions ("particles"), and moving these particles around in the search space, according to simple mathematical formulas of the particle's position and velocity. Each particle's movement is influenced by its locally best known position and is also guided toward the best known positions in the search space, which are updated as better positions are found by other particles.

Datasets. A set of five biopharmaceutical data sets was included in the study (see Table 1 for details and the Supporting Information of ref 2 for smiles codes and activity classes of the compounds in each dataset). All five datasets contain both a dichotomous response (active/inactive compound) and a continuous response (activity). The compounds were numerically described using the AZ oeselma descriptors. The AZ oeselma descriptors consist of an in-house AstraZeneca compilation of 93 molecular physicochemical descriptors, similar to the descriptors described by Labute,[21] containing 1D and 2D descriptors (e.g., counts of atoms and bonds, charges, surfaces and lipophilicity). The chemical structures in each dataset were modeled in their respective neutral state. Each continuous response and all features were scaled to unit variance. The classification datasets were well-balanced between the two classes.

Design of the Experiment. To estimate the effect of the choice of learner and feature selection method on the predictive performance on QSAR models, we designed a three-factor full factorial experiment. The factors represent the learners, the feature selection methods, and the datasets. We are not interested in the effect of the dataset factor per se; however, we needed to control for it when estimating the effect of the other factors (since it is difficult to achieve good models on some datasets and easier on others). The experiment (i.e., all factor combinations: the four feature selection methods plus no feature selection, the eight learners applied to the five regression and classification data sets, respectively) were run in five replicates, making the total number of experimental runs 1504.

Each feature selection method has a parameter (λ for MARS, λ and α for elastic net, γ for the wrapper, and the number of features to include in the model for ReliefF) that needs to be chosen. Similarly, each learner (except multiple linear regression and random forest, for which the default values were used throughout the entire experiment) has one or more parameters that we need to specify (C and γ for SVM, λ for MARS, λ and α for elastic net, M and the penalization parameter for neural networks, the number of latent variables for PLS, k for *k*NN; except for these parameters, the default settings in each respective R package were used). We used particle swarm optimization (maximum number of iterations set to 500) to choose these parameters, using the results from an inner loop 5-fold cross-validated estimate of predictive performance as objective function: the area under the receiver operating characteristics curve (AUC) for the binary response variables and mean squared error (MSE) for the continuous response variables (see Eklund et al.[2] for further details). An outer 5-fold cross-validation loop was employed for assessing the predictive performance of each combination of feature selection method and learner. Briefly, the protocol of this double-loop 5-fold cross-validation[14,26] procedure is to first split the dataset into five parts containing roughly the same number of observations (the outer loop). Four out of the five parts then enters the inner loop, where a 5-fold cross-validation procedure is used as an objective function for optimizing model selection (i.e., for choosing as good values as possible for the parameters C and γ for SVM, λ for MARS, λ and α for elastic net, M and the penalization parameter for neural networks, the number of latent variables for PLS, and k for *k*NN). A model is then fitted to the four parts in the inner loop using the chosen parameter values. This model's performance is subsequently assessed against the part of the dataset that was withheld in the outer loop, and thus has not been used for model selection, which gives us an unbiased estimate of the model's performance. This whole procedure is then repeated for each of the five parts in the outer loop cross-validation. The double loop cross-validation procedure was used to separate model selection from model assessment, in order to avoid a positive bias in the performance estimates.

This experimental design can be analyzed using a normal linear model with the model assessments estimates as response variables (i.e., the AUC and MSE estimates from the outer-loop cross-validation for classification and regression, respectively). Evidence for interaction effects between the choice of feature selection and learning methods can be tested by including the interaction term between the learner and feature selection method in the linear model.

Software. The experiments were implemented in the R statistical programming language.[27] The following R packages were used: *glmnet*[28] for elastic nets, *pls*[29] for PLS, *earth*[30] for MARS, *CORElearn*[31] for ReliefF, *randomForest*[32] for random forest, *e1071*[33] for SVM, *nnet*[34] for neural networks, *FNN*[35] for *k*NN, *pso*[36] for PSO, *pROC*[37] for computing AUC, and *vioplot*[38] for producing violin plots.

■ RESULTS

Parameter estimates and *p*-values from a normal linear model fit to the results from the benchmarking experiment are shown in Tables 2 and 3 for regression and classification, respectively. The normal assumption was checked with quantile−quantile plots and the homoscedasticity assumption by plotting the response against the residuals (no clear deviations were observed). The significance level was set to 5%. All reported *p*-values are two-sided.

Figures 1 and 2 show violin plots[39] of results obtained with different learning and feature selection methods, respectively, across datasets (a violin plot is a combination of a box plot and a rotated kernel density plot on each side of the box plot). The violins show the distribution of MSE and AUC obtained across the five replicates and all datasets (we have subtracted the dataset specific intercept terms from the MSE and AUC to make results obtained on different datasets comparable).

Table 2. Results on Regression End Points; Random Forest Learner and No Variable Selection are the Reference Factor Levels

	factor level	effect estimate	*p*-value
Intercept	Random forest with no feature selection	0.59	<0.001
Feature selection effect	MARS	0.04	0.15
	Elastic net	0.01	0.75
	ReliefF	0.02	0.48
	Wrapper	−0.02	0.61
Learner effect	MLR	0.40	<0.001
	Elastic net	0.01	0.63
	PLS	0.08	0.005
	MARS	0.16	0.004
	SVM	<|0.005|	0.99
	Neural networks	<|0.005|	0.98
	kNN	0.39	<0.001
Interaction effect	kNN: Elastic net	−0.37	<0.001
	MLR: Elastic net	−0.37	<0.001
	Neural networks: Elastic net	−0.01	0.88
	PLS: Elastic net	−0.05	0.22
	SVM: Elastic net	−0.03	0.47
	kNN: MARS	−0.22	<0.001
	MLR: MARS	−0.32	<0.001
	Neural networks: MARS	<|0.005|	0.90
	PLS: MARS	−0.03	0.41
	SVM: MARS	0.02	0.66
	kNN: ReliefF	−0.27	<0.001
	MLR: ReliefF	−0.36	<0.001
	Neural networks: ReliefF	−0.01	0.77
	PLS: ReliefF	−0.05	0.17
	SVM: ReliefF	0.16	<0.001
	kNN: Wrapper	−0.32	<0.001
	MLR: Wrapper	−0.41	<0.001
	Neural networks: Wrapper	−0.01	0.74
	PLS: Wrapper	−0.07	0.07
	SVM: Wrapper	−0.04	0.35

Table 3. Results on Classification End Points; Random Forest Learner and No Variable Selection are the Reference Factor Levels

	factor level	effect estimate	*p*-value
Intercept	Random forest with no feature selection	0.84	<0.001
Feature selection effect	MARS	−0.02	0.44
	Elastic net	−0.02	0.49
	ReliefF	0.01	0.71
	Wrapper	0.02	0.30
Learner effect	MLR	−0.13	<0.001
	Elastic net	0.02	0.26
	PLS	−0.01	0.69
	MARS	0.02	0.36
	SVM	<|0.005|	0.91
	Neural networks	0.01	0.63
	kNN	−0.26	<0.001
Interaction effect	kNN: Elastic net	0.12	<0.001
	MLR: Elastic net	0.14	<0.001
	Neural networks: Elastic net	<|0.005|	0.87
	PLS: Elastic net	0.04	0.25
	SVM: Elastic net	0.01	0.78
	kNN: MARS	0.06	0.02
	MLR: MARS	0.15	<0.001
	Neural networks: MARS	−0.04	0.16
	PLS: MARS	0.01	0.82
	SVM: MARS	0.04	0.21
	kNN: ReliefF	0.07	0.02
	MLR: ReliefF	0.13	<0.001
	Neural networks: ReliefF	−0.01	0.80
	PLS: ReliefF	<|0.005|	0.94
	SVM: ReliefF	−0.01	0.77
	kNN: Wrapper	0.08	0.003
	MLR: Wrapper	0.16	<0.001
	Neural networks: Wrapper	−0.01	0.83
	PLS: Wrapper	<|0.005|	0.86
	SVM: Wrapper	0.02	0.50

■ DISCUSSION

Previous articles have indicated that there may be feature selection methods that are particularly well-suited for use in combination with certain learners (see, e.g., Hall and Holmes[8]). In this paper, we empirically investigated this hypothesis in the QSAR setting using a large number of in silico experiments. The results do not give any clear support for particularly favorable learner−feature selection method combinations, among the learners and feature selection methods that we studied. (Although random forest may have a slightly worse performance with the wrapper approach than with other feature selection methods, see below. However, this effect is not significant.) This conclusion was consistent when studying each dataset individually, i.e., the nonsignificant results stem from consistent performance across datasets rather than from large variance.

The only learners that conclusively gain in prediction performance from feature selection were MLR and *k*NN, however they do not gain significantly more by using a certain feature selection method in particular. It is not surprising that these two learners were helped by feature selection, considering that they were not designed for problems with a large number of features *p* in relation to the number of observations *n*. Random forest, SVM, and neural networks were the learners that consistently performed the best.

The wrapper method seems to be the feature selection method that performs the best, although the absolute differences compared with the other methods were small and statistically not significant. However, the wrapper approach is computationally expensive and requires a search algorithms to traverse all possible combinations of features. Elastic nets in particular is extremely fast and performs almost as well as the wrapper approach.

In our previous paper,[2] we did not observe such a good performance of the wrapper approach as we did in this paper. We believe that there are two reasons for this. First, in this work we increased the number of iterations in the particle swarm optimization (from 100 to 500), meaning that the PSO had more iterations to find a good feature subset (out of the 2^p possible ones). Second, in Eklund et al.,[2] we only used random

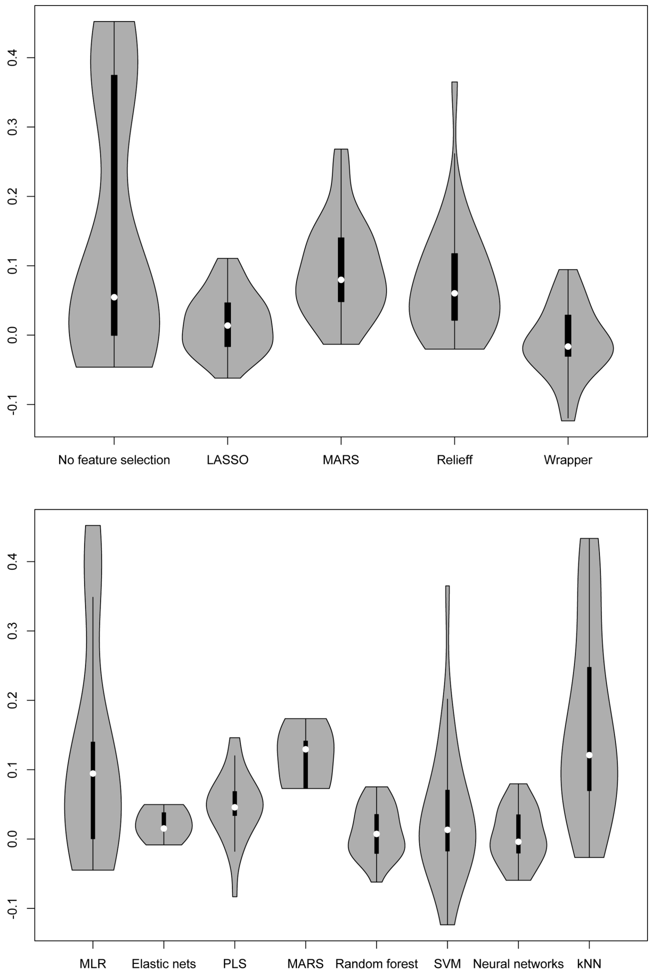


Figure 1. (Top panel) Distributions of MSE obtained using (top panel) different feature selection methods across all datasets, replications, and learners and (bottom panel) different learners across all datasets, replications, and feature selection methods. We have subtracted the dataset-specific intercept term from the MSE to make results obtained on different datasets comparable, which is why the distributions are centered around zero (0). Note that a low value indicates a good model.

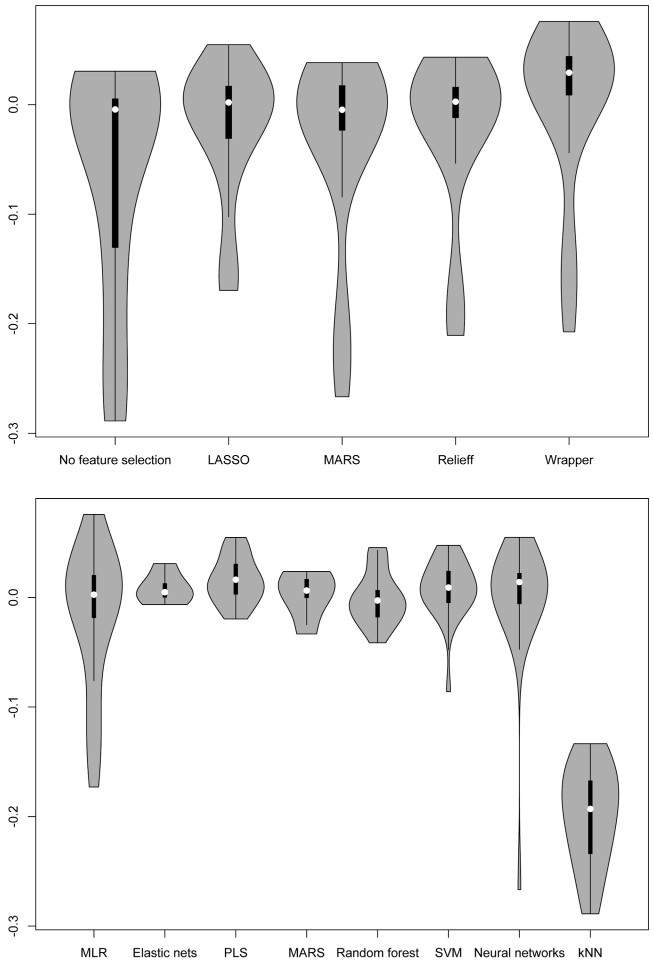


Figure 2. Distributions of AUC obtained using (top panel) different feature selection methods across all datasets, replications, and learners and (bottom panel) difference learners across all datasets, replications, and feature selection methods. We have subtracted the dataset-specific intercept term from the AUC to make results obtained on different datasets comparable, which is why the distributions are centered around zero (0). Note that a high value indicates a good model.

forest as a learner. Random forest seems to be the learner where the wrapper approach performed the worst (see Tables 2 and 3). Although this effect is small and nonsignificant it was likely more pronounced when the number of PSO iterations was lower.

To conclude, we do not find any strong support for interaction effects between specific learners and feature selection methods. Unsurprisingly, learners that do not handle a large number of features, compared to the number of observations (unpenalized methods such as MLR and *k*NN), are the learning methods that benefit the most from feature selection.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: lars.a.carlsson@astrazeneca.com.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

The computations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at Uppsala University (UPPMAX).

■ REFERENCES

(1) Arakawa, M.; Hasegawa, K.; Funatsu, K. The Recent Trend in QSAR Modeling—Variable Selection and 3D-QSAR Methods. *Curr. Comput.-Aided Drug Des.* **2007**, *3*, 254−262.
(2) Eklund, M.; Norinder, U.; Boyer, S.; Carlsson, L. Benchmarking Variable Selection in QSAR. *Mol. Inform.* **2012**, *31*, 173−179.
(3) Goodarzi, M.; Freitas, M. P.; Jensen, R. Feature selection and linear/nonlinear regression methods for the accurate prediction of glycogen synthase kinase-3beta inhibitory activities. *J. Chem. Inf. Model.* **2009**, *49*, 824−832.
(4) Liu, Y. A comparative study on feature selection methods for drug discovery. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1823−1828.
(5) Brank, J.; Grobelnik, M.; Milic-Frayling, N.; Mladenic, D. Interaction of Feature Selection Methods and Linear Classification Models. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML-02)*, Sydney, Australia, July 8−12, 2002.
(6) Mladenić, D.; Brank, J.; Grobelnik, M.; Milic-Frayling, N. Feature selection using linear classifier weights: Interaction with classification

models. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*; Association for Computing Machinery (ACM): New York, 2004; pp 234–241 (http://doi.acm.org/10.1145/1008992.1009034).

(7) Lu, Y.; Peng, F.; Li, X.; Ahmed, N. Coupling Feature Selection and Machine Learning Methods for Navigational Query Identification. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM '06)*, Arlington, VA, 2006; pp 682-689.

(8) Hall, M. A.; Holmes, G. Benchmarking Attribute Selection Techniques for Discrete Class Data Mining. *IEEE Trans. Knowl. Data Eng.* **2003**, *15*, 1437–1447.

(9) Saeys, Y.; Inza, I.; Larrañaga, P. A review of feature selection techniques in bioinformatics. *Bioinformatics* **2007**, *23*, 2507–17.

(10) Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning*, 2nd Edition; Springer–Verlag: New York, 2009.

(11) Robnik-Šikonja, M.; Kononenko, I. Theoretical and Empirical Analysis of ReliefF and RReliefF. *Mach. Learn.* **2003**, *53*, 23–69.

(12) Zou, H.; Hastie, T. Regularization and variable selection via the Elastic Net. *J. R. Stat. Soc., Ser. B* **2005**, *67*, 301–320.

(13) Tibshirani, R. Regression Shrinkage and Selection Via the Lasso. *J. R. Stat. Soc., Ser. B* **1994**, *58*, 267–288.

(14) Eklund, M.; Spjuth, O.; Wikberg, J. E. S. The C^1C^1: A framework for simultaneous model selection and assessment. *BMC Bioinform.* **2008**, *9*, 360.

(15) Kononenko, I. Estimating Attributes: Analysis and Extensions of RELIEF. 1994.

(16) Wold, H. Soft modelling by latent variables: the nonlinear iterative partial least squares (NIPALS) approach. In *Perspectives in Probability and Statistics: Papers in Honor of M. S. Bartlett*; Gani, J., Ed.; Academic Press: New York, 1975

(17) Gustafsson, M. G. A probabilistic derivation of the partial least-squares algorithm. *J. Chem. Inf. Comput. Sci.* **2001**, *41*, 288–94.

(18) Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32.

(19) Vapnik, V. N. *Statistical Learning Theory*, 1st ed.; Wiley: New York, 1998.

(20) Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In *Proceedings of the 1995 IEEE International Symposium of Neural Networks (ISNN)*, Perth, WA, Nov. 27–Dec. 1, 1995; IEEE: Piscataway, NJ, 1995; Vol. *4*, pp 1942–1948.

(21) Labute, P. A widely applicable set of descriptors. *J. Mol. Graph. Model.* **2000**, *18*, 464–77.

(22) Bruce, C. L.; Melville, J. L.; Pickett, S. D.; Hirst, J. D. Contemporary QSAR classifiers compared. *J. Chem. Inf. Model.* **2007**, *47*, 219–27.

(23) http://www.epa.gov/ncct/dsstox/sdf_cpdbas.html.

(24) http://www.epa.gov/ncct/dsstox/sdf_epafhm.html.

(25) http://www.epa.gov/ncct/dsstox/sdf_fdamdd.html.

(26) Stone, M. Cross-Validatory Choice and Assessment of Statistical Predictions. *J. R. Stat. Soc., Ser. B* **1974**, *36*, 111–147.

(27) R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing: Vienna, Austria, 2012 (ISBN 3 900051 07 0).

(28) Friedman, J.; Hastie, T.; Tibshirani, R. Regularization Paths for Generalized Linear Models via Coordinate Descent. *J. Stat. Software* **2010**, *33*, 1–22.

(29) Mevik, B.-H.; Wehrens, R.; Liland, K. H. *pls: Partial Least Squares and Principal Component regression*; R package version 2.3-0; 2011.

(30) Hastie, T.; Tibshirani, R. *earth: Multivariate Adaptive Regression Spline Models*; R package version 3.2-2; 2012 (from mda:mars).

(31) Robnik-Sikonja, M.; Savicky, P. *CORElearn: CORElearn—Classification, regression, feature evaluation and ordinal evaluation*; R package version 0.9-39; 2012.

(32) Liaw, A.; Wiener, M. Classification and Regression by randomForest. *R News* **2002**, *2*, 18–22.

(33) Dimitriadou, E.; Hornik, K.; Leisch, F.; Meyer, D.; Weingessel, A. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*; R package version 1.6; 2011.

(34) Venables, W. N.; Ripley, B. D. *Modern Applied Statistics with S*, 4th ed.; Springer: New York, 2002 (ISBN 0 387954 57 0).

(35) Li, S. *FNN: Fast Nearest Neighbor Search Algorithms and Applications*; R package version 0.6-2; 2010.

(36) Bendtsen, C. *pso: Particle Swarm Optimization*; R package version 1.0.1. 2011.

(37) Robin, X.; Turck, N.; Hainard, A.; Tiberti, N.; Lisacek, F.; Sanchez, J.-C.; M?ller, M. pROC: An open-source package for R and S + to analyze and compare ROC curves. *BMC Bioinform.* **2011**, *12*, 77.

(38) Adler, D. *vioplot: Violin plot*; R package version 0.2; 2005.

(39) Hintze, J. L.; Nelson, R. D. Violin Plots: A Box Plot-Density Trace Synergism. *Am. Stat.* **1998**, *52*, 181–84.