Tsiliki *et al. J Cheminform (2015) 7:46*
DOI 10.1186/s13321-015-0094-2

Journal of Cheminformatics

RESEARCH ARTICLE

Open Access

RRegrs: an R package for computer-aided model selection with multiple regression models

Georgia Tsiliki[1*†], Cristian R. Munteanu[2,3†], Jose A. Seoane[4], Carlos Fernandez-Lozano[2], Haralambos Sarimveis[1] and Egon L. Willighagen[3]

Abstract

Background: Predictive regression models can be created with many different modelling approaches. Choices need to be made for data set splitting, cross-validation methods, specific regression parameters and best model criteria, as they all affect the accuracy and efficiency of the produced predictive models, and therefore, raising model reproducibility and comparison issues. Cheminformatics and bioinformatics are extensively using predictive modelling and exhibit a need for standardization of these methodologies in order to assist model selection and speed up the process of predictive model development. A tool accessible to all users, irrespectively of their statistical knowledge, would be valuable if it tests several simple and complex regression models and validation schemes, produce unified reports, and offer the option to be integrated into more extensive studies. Additionally, such methodology should be implemented as a free programming package, in order to be continuously adapted and redistributed by others.

Results: We propose an integrated framework for creating multiple regression models, called RRegrs. The tool offers the option of ten simple and complex regression methods combined with repeated 10-fold and leave-one-out cross-validation. Methods include Multiple Linear regression, Generalized Linear Model with Stepwise Feature Selection, Partial Least Squares regression, Lasso regression, and Support Vector Machines Recursive Feature Elimination. The new framework is an automated fully validated procedure which produces standardized reports to quickly oversee the impact of choices in modelling algorithms and assess the model and cross-validation results. The methodology was implemented as an open source R package, available at https://www.github.com/enanomapper/RRegrs, by reusing and extending on the caret package.

Conclusion: The universality of the new methodology is demonstrated using five standard data sets from different scientific fields. Its efficiency in cheminformatics and QSAR modelling is shown with three use cases: proteomics data for surface-modified gold nanoparticles, nano-metal oxides descriptor data, and molecular descriptors for acute aquatic toxicity data. The results show that for all data sets RRegrs reports models with equal or better performance for both training and test sets than those reported in the original publications. Its good performance as well as its adaptability in terms of parameter optimization could make RRegrs a popular framework to assist the initial exploration of predictive models, and with that, the design of more comprehensive in silico screening applications.

Keywords: Multiple regression, QSAR, R package, Caret-based tool

*Correspondence: gtsiliki@central.ntua.gr
†Georgia Tsiliki and Cristian R. Munteanu contributed equally to this work
[1] School of Chemical Engineering, National Technical University of Athens, 9 Heroon Polytechneiou Street, Zografou Campus, 15780 Athens, Greece
Full list of author information is available at the end of the article

Background

Many open-source statistical, data-mining, and machine learning software projects give access to a wide range of data processing and modelling algorithms often providing graphical user interfaces. Among them are Weka (Waikato Environment for Knowledge Analysis) [1], RapidMiner [2], Keel (Knowledge Extraction based on Evolutionary Learning) [3], Orange [4], Scikit-learn [5], C1C2 [6] and KNIME (Konstanz Information Miner) [7], effectively, providing full predictive modelling frameworks.

Additionally, web-based platforms such as OpenTox [8] and Online Chemical Modelling Environment (OCHEM) [9] focus on the development of quantitative structure-activity relationship (QSAR) models, i.e. regression or classification models that are used for the in silica assessment of physicochemical properties and biological activities of chemical compounds such as toxicity, biological potency and side effects [10–12]. Such platforms typically consist of two major subsystems: the database of experimental measurements and the modelling framework. They allow users to create their own QSAR models, predict results for new chemicals, and share them. OpenTox in particular is a platform-independent collection of components which communicate through web services, so that the user can combine data, models and validation results from multiple sources in a dependable and time-effective way. Several other tools offer virtual evaluation of chemical properties and toxicity using implemented QSAR models; for instance, Vega [13], EPI Suite [14], Toxicity Estimation Software Tool (TEST) [15], QSAR4u [16], BuildQSAR [17], OECD QSAR Toolbox [18], AZOrange [19] or Bioclipse-R [20]. However, these tools are limited to supported data sets, QSAR models or specific regressions methods.

On the other hand, the R statistical language environment [21] offers many solutions for regression modelling and also some packages providing simultaneous access to multiple methods. For example, the glmulti package conducts automated model selection and model-averaging based on the Akaike Information Criterion (AIC) or the Bayesian Information Criterion (BIC) [22], the kernlab package applies Kernel-based machine learning methods for classification, regression, clustering, novelty detection, quantile regression and dimensionality reduction, and the e1071 package includes functions for latent class analysis, short time Fourier transform, fuzzy clustering, and support vector machines. Other R packages are specializing on particular set of algorithms, for instance the R package tree focuses on producing classification and regression trees.

Of particular interest is the R package for predictive modelling called caret (Classification and Regression Training) which gathers and simplifies numerous R algorithms for the development of a wide variety of predictive models by calling and integrating more than 25 other packages [23]. Unique features of caret include data splitting, pre-processing, characterizing performance and variable importance, and parallel processing tools.

Although this package is providing useful methods for the syntactical unification of regression and classification prediction modelling approaches, the various models have different inputs and the outcomes different formats, typically depending on their parameters. This makes it hard to run all the available methods for multiple data sets, compare all the outputs, and produce a standardized results summary.

Furthermore, the complexity of the workflows complicates the reproduction of the same regression results and it may affect decisions on issues, such as how to split the original data set, how often to split the data set, which cross-validation method is to be use, which data filtering to apply before regression (correlated features removal, "not available" (NA) values elimination, etc.), which data scaling is applied (normalization, standardization, etc.), which regression methods to test, which regression parameters and seeds to use, how to summarize and compare the results for several regression models, and which criteria to use in order to choose the best regression model.

To address these limitations, the current manuscript describes a standardized framework that automates the development of a reliable and well-validated QSAR model, or set of models. The so-called RRegrs tool (R Regressions) is based on the R caret package and is focusing on regression modelling. RRegrs workflow offers a fully validated procedure capturing any variability or inconsistency in the data. A single RRegrs function call is needed to run the entire workflow and obtain the produced validated QSAR model(s) in a reproducible format in contrast to the standard, inefficient and time-consuming QSAR modelling workflow, where the modeller tries many different algorithms and even needs to further search the parameter space of each algorithm. This is a considerable advantage for users with perhaps limited statistical knowledge or limited R experience. Also reproducibility is a comparative advantage since often the same procedure needs to be applied for different data sets. RRegrs implements an easy way to explore the models' search space of linear and non-linear models with special parameters specifications and cross validation schemes. Furthermore, model outputs are easily accessible and readable, organized by methods, centralized and averaged by multiple reproducible data set splits. Summary files are also produced helping the user to easily access all methodologies results, which can then be prioritized

based on various statistics. The current implementation of the RRegrs package contains ten different regression algorithms and supports parallel processing, if prompted. RRegrs function calls can be integrated into complex desktop and web tools for QSAR. RRegrs package is available as an open repository at https://www.github.com/enanomapper/RRegrs. The current release is available from ZENODO with the doi:10.5281/zenodo.21946.

Results and discussion

RRegrs is an R package for computer-aided model selection, designed and implemented as a collection of regression tools available from the caret package. RRegrs uses the R package testthat for testing [24]. It does not apply full unit testing, but several RRegrs parameter combinations are tested, which are run during the build process. RRegrs can be used to find the best regression model for any numerical data set using some or all of ten linear and non-linear regression models: Multiple Linear regression (LM), Generalized Linear Model with Stepwise Feature Selection (GLM) [25], Partial Least Squares Regression (PLS) [26], Lasso regression (LASSO) [27], Elastic Net regression (ENET) [28], Support vector machine using radial functions (SVRM) [29], Neural Networks regression (NN) [30], Random Forest (RF) [31], Random Forest Recursive Feature Elimination (RF-RFE) [32] and Support Vector Machines Recursive Feature Elimination (SVM-RFE) [33]. Using the above methods, we explore the model space and compare outputs to decide on the optimal model, given the data. We are setting the regression method parameters with grid functions which have been carefully chosen to optimize different models, and particularly this is done for PLS, SVRM, SVM-RFE, NN, RF, RF-RFE, ENET. Specifically for SVRM, SVM-RFE the user could also specify custom parameters.

The main scope of the presented tool is to be able to run a large number of regression methods from the caret package using only one function call, to use standardized cross-validation (CV) scheme for all the methods, to obtain standardized outputs, to generate result summary tables and comparison plots for the regression methods and to store for each method detailed statistics and fitting plots. RRegrs integrates results of individual models and decides on the best model given the data set and the user specified parameters, unlike caret. Therefore, RRegrs permits users, irrespective of their programming or statistics experience, to predict any type of numerical output using multiple regression methods. In addition, advanced users could integrate RRegrs in other applications or software packages. For example, in cheminformatics RRegrs can be used to generate predictive models using molecular descriptors calculated with the rcdk package, using functions offered by the Chemistry Development Kit [34].

RRegrs function wraps up all the above mentioned procedures within just one call. The following steps are included (see Fig. 1): load parameters and data set, remove the NA values, remove near zero variance features, scale the data set, remove correlated features, split data set into training and test sets, run the selected regression methods using the selected cross-validation method(s), summarize the statistics for all methods and splittings, average for each method and cross-validation type for all splittings, apply the best model of each method and split on the test sets, apply Y-randomization on the best model, and assess the Applicability Domain. For each model, a CV scheme is introduced with two options: 10-fold repeated CV and Leave-One-Out (LOO) CV. The more time-consuming regression methods (RF, SVM-RFE, RF-RFE) are using only repeated cross-validation (other validation methods could be very slow for these complex functions), whereas for computationally demanding methods RRegrs offers parallel processing for a defined number of CPU cores. Detailed output files for all regression methods are produced, plots for individual model statistics, as well as summary statistics and comparison plots between methods, resulting in a significant number of CSV, PDF, PNG outputs files. Additionally, several summary files are created. A CSV output file is created with all the basic statistics (17 values) for each method, data splitting and CV type. Based on the above, averaged statistics are calculated for each regression method and across all data splits, which are the values needed to decide on the final best model performance. The best model is further validated with Y-randomization runs (100 by default).

For each regression method, caret package utilities are employed. For example, RRegrs uses the trainControl and train functions to set the training conditions (10 repetitions; RMSE used as metrics to choose the model) and train the model, respectively. For each method, RRegrs is generating the same list of 17 statistics values: regression name, split number, cross-validation type, number of model features, names of model features, training adjusted R-squared (adj.R^2), training root mean squared error ($RMSE_{CV/LOO}$), training $R^2_{CV/LOO}$, training standardized RMSE, test adjusted R^2, test RMSE ($RMSE_{test}$), test R^2 (R^2_{test}), test correlation, and corresponding values for both sets. If the user requests detailed output (the default flag is set to True), several files are generated such as a CSV file with statistics about each regression model listing the following information: regression method, splitting number, cross-validation type, training set summary, test set summary, fitting summary, list of predictors, training/test predictors, a full list of statistics as defined above, feature importance, residuals of the fitted model, assessment of applicability domain/leverage

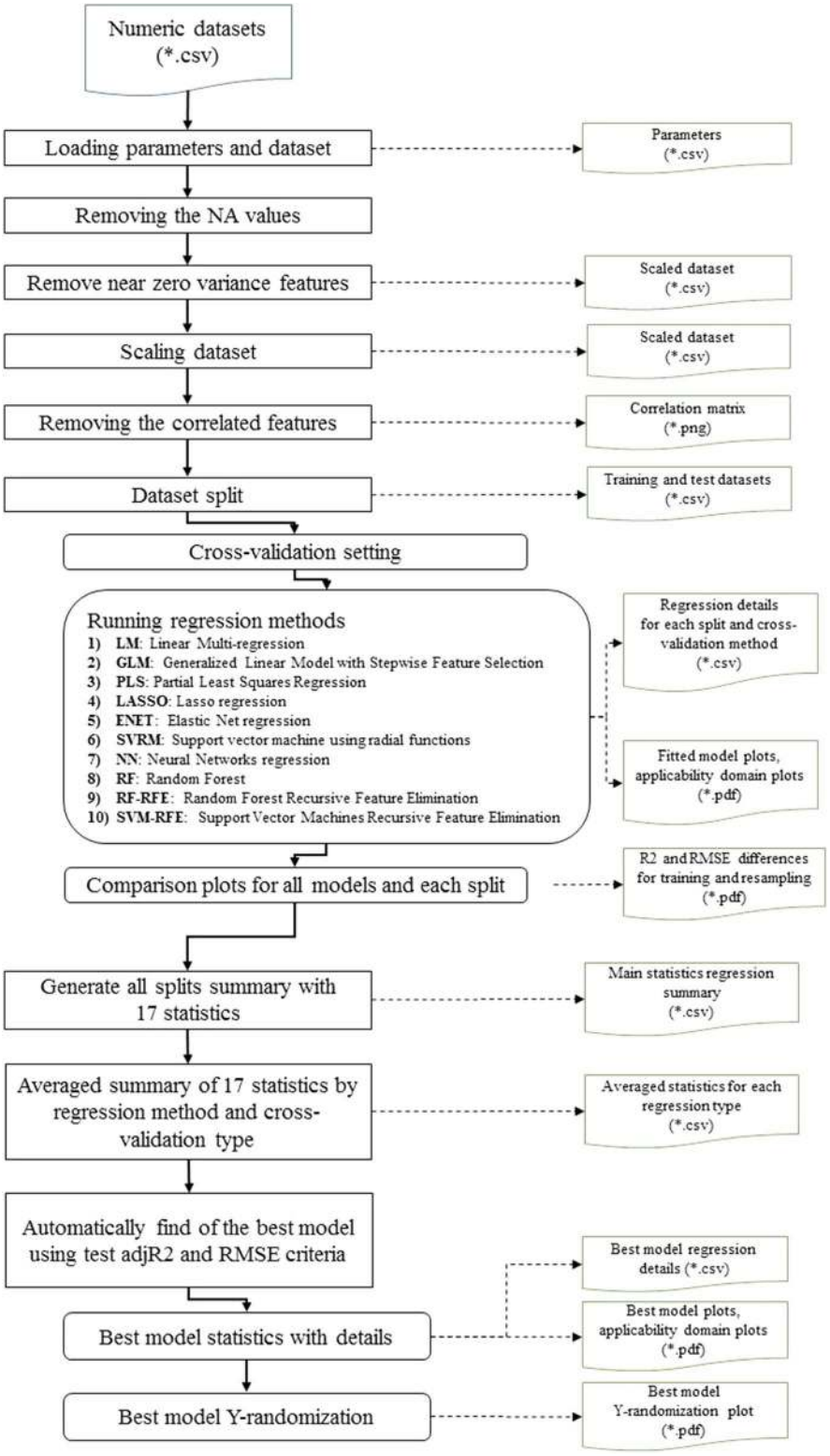

Fig. 1 RRegrs methodology flowchart. Outline of the steps performed by the RRegrs function

analysis such as mean of hat values, hat values with warnings, leverage threshold, list of points with leverage greater than threshold, Cook's distances, Cook's distance cutoff, points influence. Particularly, for each data splitting and CV method, the following plots are produced: observed versus predicted values for training/test sets, feature importance, fitted versus residuals for the fitted model, leverage statistics for fitted model, Cook's distance for fitted model, and six standard fitting plots including Cook's distance cutoff.

Moreover, RRegrs offers an exhaustive validation framework by introducing multiple random data splittings. For each algorithm and data split, the model is produced based on training and validation sets. We are reporting both CV and external validation statistics, however, the test set is used to select the final best model, i.e. the best performing amongst the optimal models produced by different algorithms (both linear and non-linear). Decision is made on the averaged statistics across data splits to remove any bias towards the structure of the test set. The best regression model is selected based on the following criterion: from the best averaged R^2_{test} (+/−0.05), the model with minimum $RMSE_{test}$ is the final one. Alternatively, the test adjusted R^2 can be used to select the best model. For the best model, an additional CSV file is generated providing detailed statistics as well as PDF plots for important statistics.

As mentioned above, parallel processing is employed during training steps by enabling caret's parallel design, and it is activated by either using caret's TrainControl "allowParallel" option, or in the case of RFE methods also within the model selection (iterating with a parallel foreach through the cross validation model selection for each feature size) using RFEControl "allowParallel" option. Libraries doMC (Linux/Mac) and doSNOW (Windows) provide foreach parallel adaptor.

The uses of RRegrs reported here are aimed at finding QSAR models for cheminformatics and nanotoxicology for the eNanoMapper European FP7 project. RRegrs was first tested with five standard data sets from UC Irvine Machine Learning Repository [35], followed by a demonstration the efficiency of RRegrs in cheminformatics and bioinformatics areas, using three publicly available data sets, as presented in the following sections.

RRegrs models for standard data sets

To benchmark RRegrs we first used five standard regression data sets from the UC Irvine machine learning repository [35]: the housing [36], computer hardware, wine quality [37], automobile [38], and Parkinsons telemonitoring [39] data sets. Based on these data sets we demonstrate the RRegrs methodology capabilities in different scientific fields. The number of cases and features of these data sets are described in the Methods section.

The housing data set is the most used standard data set for complex regression methods: combination of regression estimators as genetic algorithm-based selective neural network ensemble [40], distributed multivariate regression using wavelet-based collective data mining [41], application of the Bayesian evidence framework to support vector regression (SVR) [42], Principal Components approach that combines regression estimates [43], regression on feature projections (RFP) method [44], subset-based least squares subspace regression in reproducing Kernel Hilbert space (RKHS) [45], Smola and Scholkopf's sequential minimal optimization (SMO) algorithm for SVM regression [46], etc.

Tables 1 and 2 present two statistic values for these standard data sets: R^2_{test} and $RMSE_{test}$ values, averaged by 10 different data set splits, using 10-fold repeated CV and 10 Y-randomization. The results show that advanced methods such as RF-RFE and RF give the highest R^2 values. In the case of the Housing data set, PLS provides a very low R^2_{test} of 0.266 compared with the RF-RFE/RF that shows 0.875/0.874 (R^2_{test} for LM is 0.707). Because of its slightly lower $RMSE_{test}$ value compared to RF-RFE

Table 1 Test averaged R^2 values for five standard data sets

RRegrs method	Housing	Computer hardware	Red wine quality	Automobile	Parkinson telemonitoring
LM	0.707	0.822	0.355	0.824	0.154
GLM	0.709	0.825	0.353	0.824	0.153
PLS	0.266	0.740	0.066	0.757	0.091
LASSO	0.704	0.828	0.354	0.831	0.154
ENET	0.705	0.826	0.354	0.830	0.154
SVRM	0.845	0.765	0.396	0.853	0.637
NN	0.688	0.824	0.352	0.829	0.142
RF	0.874	0.907	0.500	0.915	0.972
RF-RFE	0.875	0.903	0.501	0.914	0.900
SVM-RFE	0.717	0.742	0.383	0.714	0.479

Table 2 Test averaged RMSE values for five standard data sets

RRegrs method	Housing	Computer hardware	Red wine quality	Automobile	Parkinson telemonitoring
LM	0.007	0.001	0.002	0.076	0.034
GLM	0.007	0.001	0.002	0.076	0.034
PLS	0.011	0.001	0.003	0.094	0.035
LASSO	0.007	0.001	0.002	0.074	0.034
ENET	0.007	0.001	0.002	0.075	0.034
SVRM	0.005	0.001	0.002	0.067	0.022
NN	0.007	0.001	0.002	0.075	0.034
RF	0.005	0.001	0.002	0.052	0.006
RF-RFE	0.005	0.001	0.002	0.052	0.013
SVM-RFE	0.008	0.002	0.002	0.113	0.027

(less than 0.001), RRegrs suggests RF as the best model. Figure 2 shows the differences for R^2_{test} and $RMSE_{CV}$ on the training set (data split 1) and Fig. 3 presents the comparison for resampling on the training set (data split 1). In order to observe the quality of the regression models, Fig. 4 presents the observed versus predicted values in the test set for the best models for five data sets (10-fold repeated CV). The applicability domain section of RRegrs plotted the leverage for the Housing best fitted model (RF) as in Fig. 5.

The best model for the Computer Hardware data set was obtained with the RF method (R^2_{test} of 0.907) and the worst one using PLS (R^2_{test} of 0.740). The LM method has R^2_{test} of 0.822. Models for the Red Wine data set do not produce good values for R^2_{test} (>0.501) due to the non-continuous values of the output variable. When RRegrs is applied to the Automobile data set, R^2_{test} values vary from about 0.915 for RF/RF-RFE to 0.714 for SVM-RFE (R^2_{test} for LM is 0.824).

If the RRegrs call uses all available CPU cores for the complex methods, one dataset split, one Y-randomizations, and all the regression methods, the following execution times (in seconds) are obtained for the Boston standard dataset (see Table 3). The computer was an

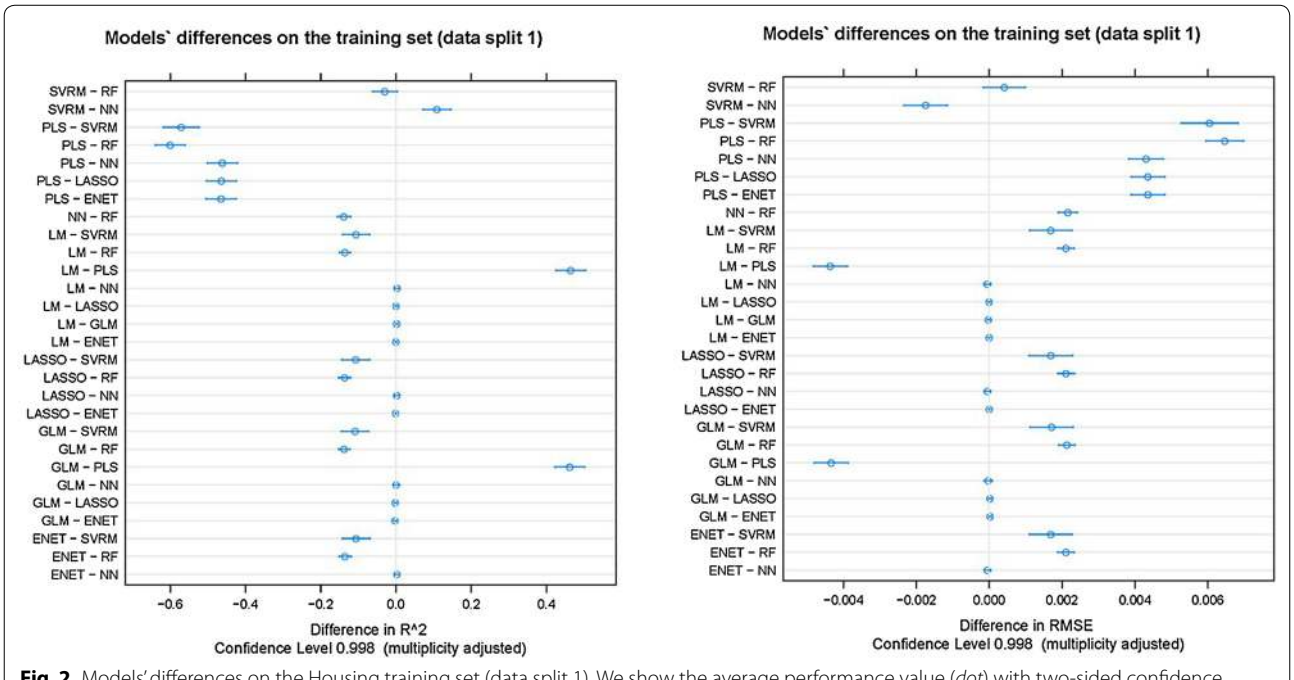

Fig. 2 Models' differences on the Housing training set (data split 1). We show the average performance value (*dot*) with two-sided confidence limits as computed by Student *t* test with Bonferroni multiplicity correction. Results are shown for RMSE and R^2 statistics and all pairwise model comparisons

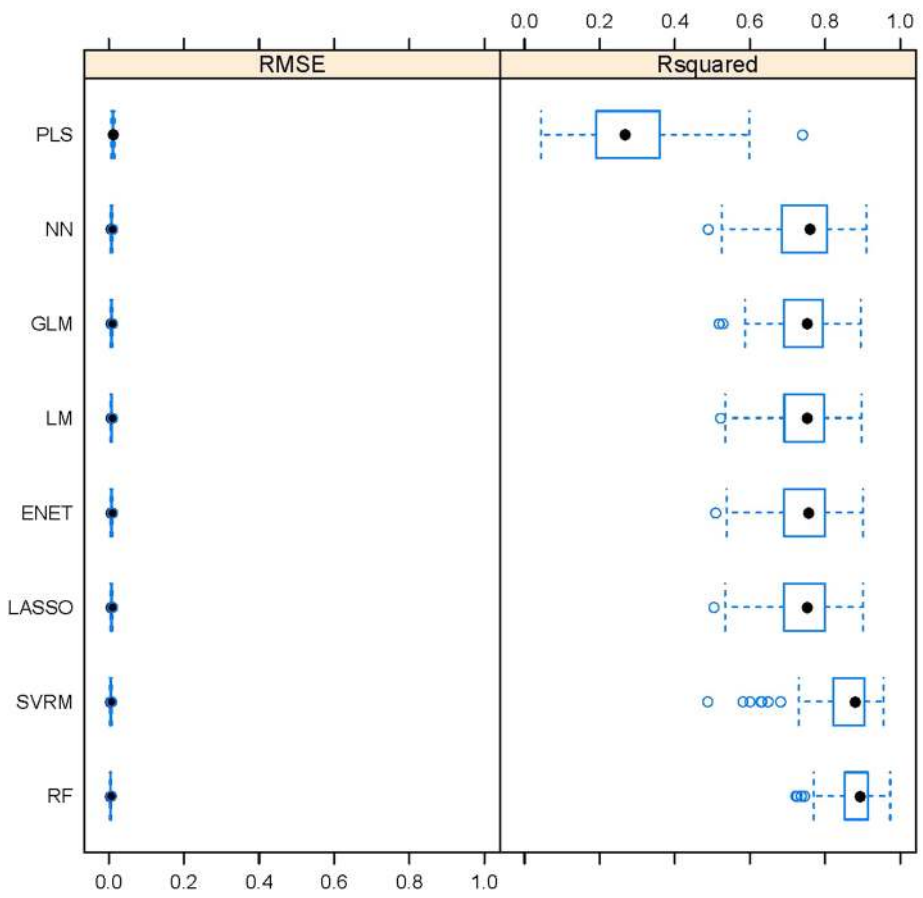

Fig. 3 Models' comparison of resampling results for Housing training set (data split 1). Univariate visualization of the resampling distributions of RMSE and R^2 statistics, for the various RRegrs models

Windows 8.1 64bit with i7-4790 CPU (3.60 GHz, 4 cores, 8 logical cores), 16G RAM. The total execution time was 5.43 min (325.64 s).

Use case 1: RRegrs application on protein corona data

Recent studies have shown that the presence of serum proteins in vitro cell culture systems form a protein adsorption layer (a.k.a. the 'protein corona') on the surface of nanoparticles (NPs). This corona is reported to affect the nanoparticle-cell interactions as well as change the cell response [47, 48], and defines the NP's 'biological identity' [49]. It thus encodes information about the interface formed between the NP core and the cell surface within a physiological environment.

This section presents results for proteomics data recently published that characterizes the serum protein corona 'fingerprint' formed around a library of 105 distinct surface-modified gold NPs [49]. The authors used LC–MS/MS to identify 129 serum proteins which were considered suitable for relative quantification. The relative abundances for each of these proteins on the corona of a nanoparticle formulation defines the serum protein 'fingerprint' for that formulation. The authors presented a multivariate regression model that uses the protein corona fingerprint to predict cell association for the gold NPs and found a model predicted cell association with a R^2_{LOO} of 0.81. Specifically, they applied a PLS regression model along with an internal iterative filtering procedure using the Variable Importance to the Projection (VIP) score and jackknife resampling.

Here we present results on the initial set of 129 × 84 proteins to gold NPs data (21 neutral NPs were excluded from analysis as in Walkey et al. [49]), and also on a set of 76 × 84 proteins to gold NPs data. These 76 proteins are selected in [49] with VIP ≥ 0.6 threshold. RRegrs was applied with 10 random splits of the data (75 % train and 25 % test) along with 10 Y-randomization runs for the best model. Table 4 shows the best model selected by

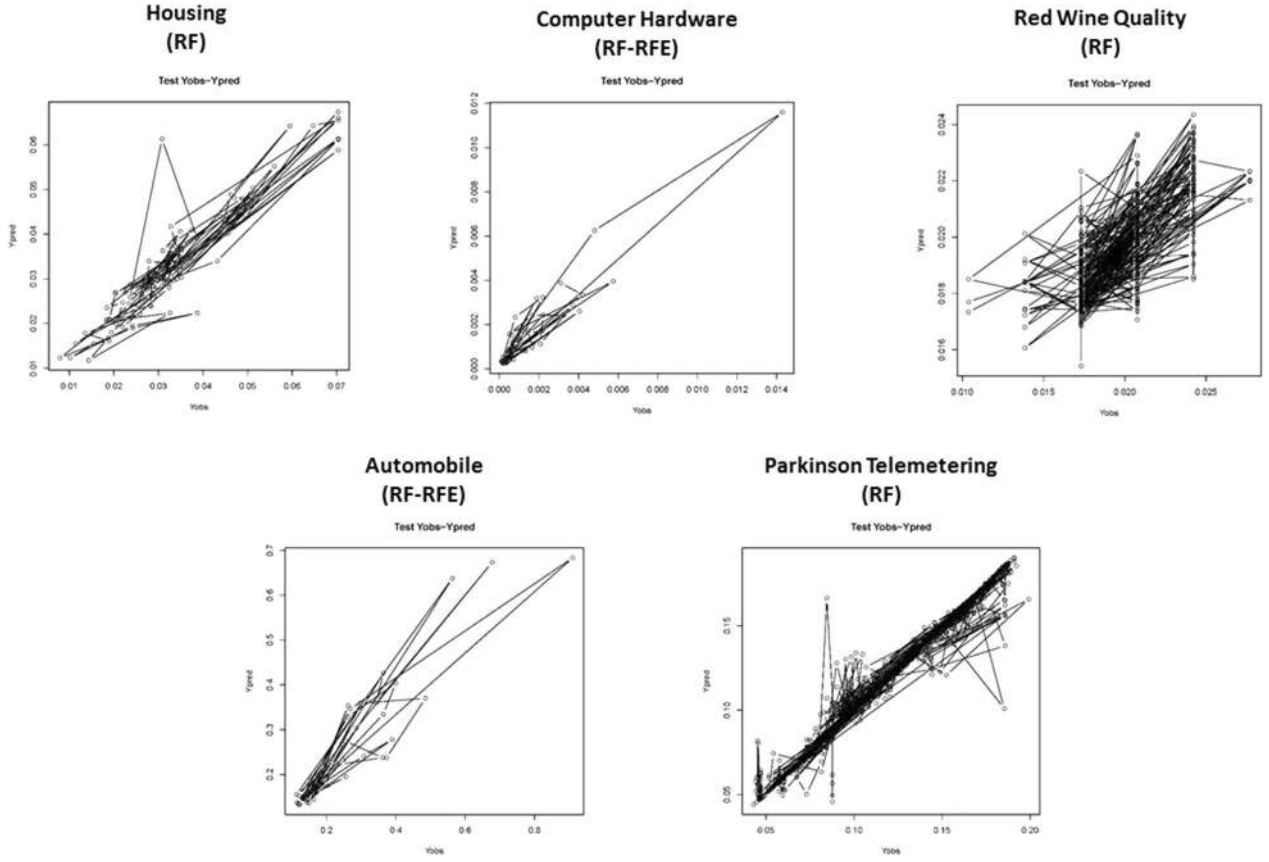

Fig. 4 Test Yobs—Ypred for the five standard data sets best models (10-fold cross-validation). Plots for the observed versus the predicted values and the best model found for each of the five data sets

RRegrs, its number of features, the adj.R^2 and the R^2 and RMSE values for the train and test sets, averaged over 10 random splits of the data. Table 5 shows the best model found in all data splits, i.e. we compare all methodologies and data splits to find the best R^2_{test}. Data are normalized and filtered using the RRegrs near zero variance and correlation filters, for that reason the 129 proteins are filtered to be 99 and the 76 proteins data set are reduced to 60 features.

For the data set with 129 proteins, the best model is an SVRM model with $R^2_{test} = 0.631$. CV results on the training set can be seen in Fig. 6, where we can observe that the LM and GLM models are not suitable for the protein corona data, whereas the remaining methodologies perform similarly. It can be observed in Table 5 that the highest value reported was $R^2_{test} = 0.844$ for individual split nine of the data set.

When we study the set with 76 proteins, we find that the best model is an SVRM model with averaged $R^2_{test} = 0.728$, whereas the best individual split value is $R^2_{test} = 0.89$. CV results on the training set are shown in Fig. 7. The corresponding RRegrs results for the PLS model are $R^2_{test} = 0.7$ (averaged over 10 data splits), whereas the highest values are reported for individual split five $R^2_{test} = 0.885$ (for repeated CV) and $R^2_{test} = 0.873$ (for LOO). Although the last number cannot be directly compared to $R^2_{LOO} = 0.81$ reported by the authors, it gives an indication of how our PLS implementation performs for the specific data set.

Use case 2: RRegrs application on metal oxides data set

The authors of [50] combined experimental and theoretical measurements to develop a nano-QSAR model that describes the toxicity of eighteen nano-metal oxides (MeOx) to the human keratinocyte (HaCaT) cell line, which is a common in vitro model for keratinocyte response during toxic dermal exposure. The study was aimed at exposing and explaining the differences in modes of toxic action of metal oxide nanoparticles between the eukaryotic system and the prokaryotic system (*E. coli*).

They calculated 32 parameters that quantitatively describe the variability of the nanoparticles' structure, called nano-descriptors, which included

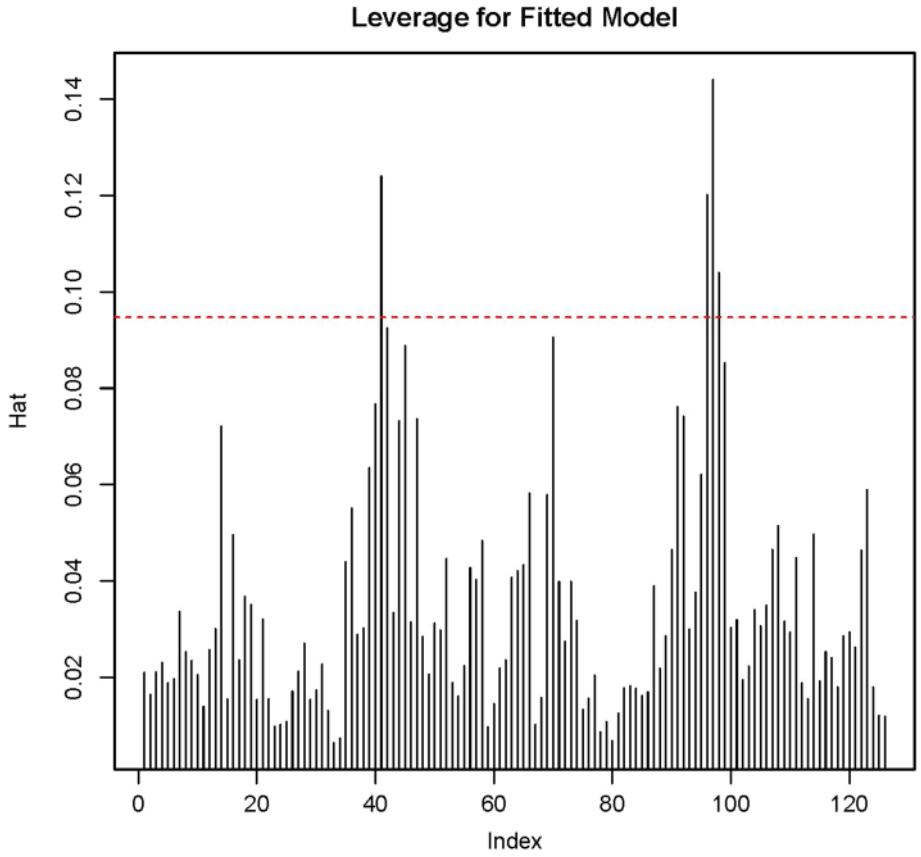

Fig. 5 Leverage for Housing best fitted model. Histogram showing the Hat values for the RF fitted model. The *red dashed line* indicates the leverage threshold value ($\frac{3m}{n}$, where m are the number of model parameters and n the number of observations)

Table 3 RRegrs execution time (in seconds) for one split of Boston House dataset

Method	Repeated CV	LOOCV
LM	11.97	1.78
GLM	2.14	5.48
PLS	0.99	1.40
Lasso	1.32	–
ENET	12.70	45.30
SVM radial	4.62	13.77
NN	12.53	49.97
RF	88.89	–
RF-RFE	3.89	–
SVM-RFE	46.36	–

quantum-mechanical descriptors derived from quantum-chemical calculations, and image descriptors derived from transmission electron microscopy (TEM) images. Some of the descriptors included are: particle size and size distribution, agglomeration state, particle shape, crystal structure, chemical composition, surface area, surface chemistry, surface charge, electronic properties (reactivity, conductivity, interaction energies, etc.), and porosity. Additionally, the LC_{50} was calculated from experimental data for all MeOx NPs. This is the concentration that caused a 50 % reduction of the cells after 24 h exposure, whereas the $-\log(LC_{50})$ values were used in modelling, as the dependent variable t be predicted.

The authors applied a Genetic Algorithm (GA) to independently select the most efficient combination of the molecular descriptors which were then analyzed using multiple linear regression. They found that two descriptors were sufficient to predict NPs toxicity with high statistical significance, namely ΔH_c^f descriptor, which is the enthalpy of formation of metal oxide nanocluster representing a fragment of the surface and, χ^c descriptor, which is the Mulliken's electronegativity of the cluster. The reported values are: $R^2 = 0.93$ (RMSE $= 0.12$), $R^2_{LOO} = 0.86$ ($RMSE_{LOO} = 0.16$), $R^2_{test} = 0.83$ ($RMSE_{test} = 0.13$). Note that $R^{2,}$ here and in other cases, refers to the coefficient of determination for fitting the model to the training data.

Table 4 RRegrs averaged statistics reported for the three use cases, under the 10-fold repeated CVscheme

Use case	Best model	Features no.	adj.R^2	R^2_{CV}	R^2_{test}	$RMSE_{CV}$	$RMSE_{test}$
UC1: protein corona							
129 proteins	SVRM	99	1.02	0.687	0.631	0.558	0.612
76 proteins	SVRM	60	0.582	0.777	0.728	0.477	0.538
UC2: metal oxides	ENET	8.8	1	0.933	0.746	0.639	0.639
UC3: toxicity data	SVRM	8	0.7	0.556	0.537	0.68	0.67

Averaged values are reported across the ten different data splits

Table 5 RRegrs best model statistics reported for the three use cases. Both LOO and CV values are considered

Use case	Best model	Data split	Features no.	Validation type	adj.R^2	$R^2_{CV/LOO}$	R^2_{test}	$RMSE_{CV/LOO}$	$RMSE_{test}$
UC1: protein corona									
129 proteins	SVRM	5	99	LOO, CV	1.03	0.644/0.61	0.844	0.618/0.643	0.357
76 proteins	SVRM	5	60	LOO, CV	0.407	0.767/0.741	0.89	0.525/0.527	0.296
UC2: metal oxides	ENET	8	8	LOO	0.808	0.7	0.998	0.588	0.246
UC3: toxicity data	SVRM	2	8	LOO	0.685	0.506	0.657	0.705	0.609

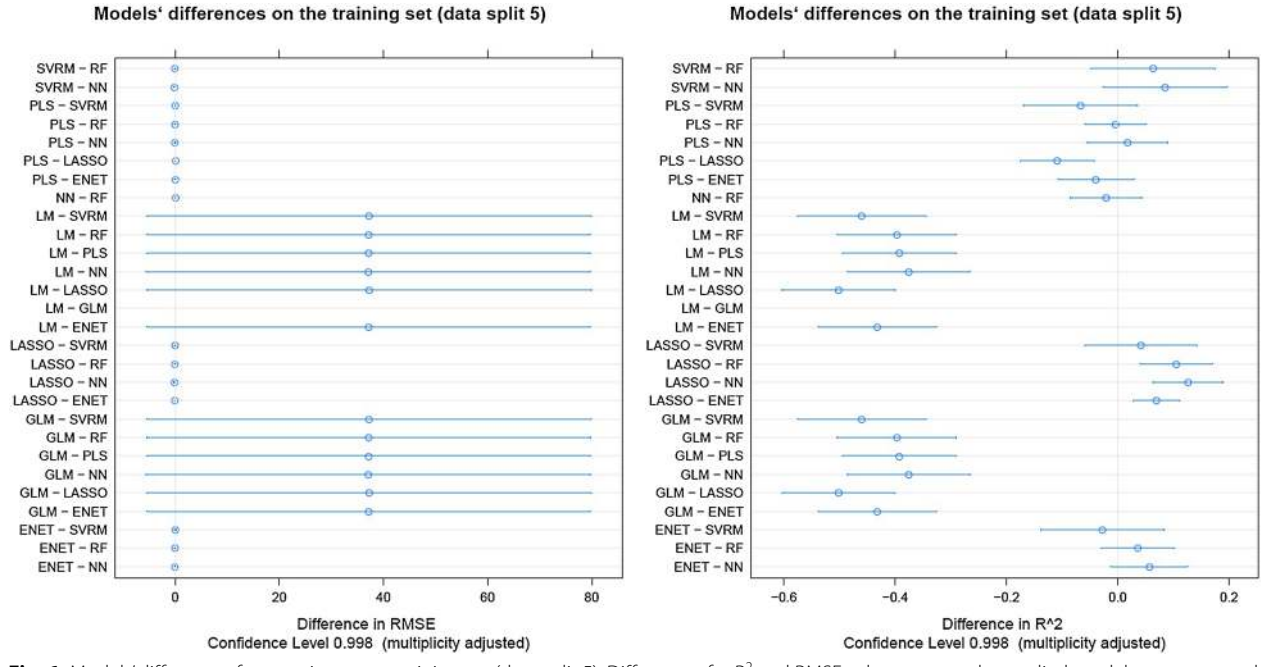

Fig. 6 Models' differences for protein corona training set (data split 5). Differences for R^2 and RMSE values among the applied models are presented. The average performance with two-sided confidence limits are plotted as derived by the Student t-test with Bonferroni multiplicity correction. We can observe that LM and GLM models are not fitting the data very well (large RMSE values)

Ten random splits of the data were performed (75 % train and 25 % test) along with ten Y-randomization runs for the best model. Tables 4 and 5 show RRegrs results for the initial set of 32 parameters to the eighteen MeOx's, averaged or non-averaged values across the ten data splits, respectively. Because of the restricted number of samples and descriptors RRegrs was applied without filtering options, whereas data were normalized, as in [50]. The best model was selected between those that perform feature selection, i.e. GLM, LASSO,

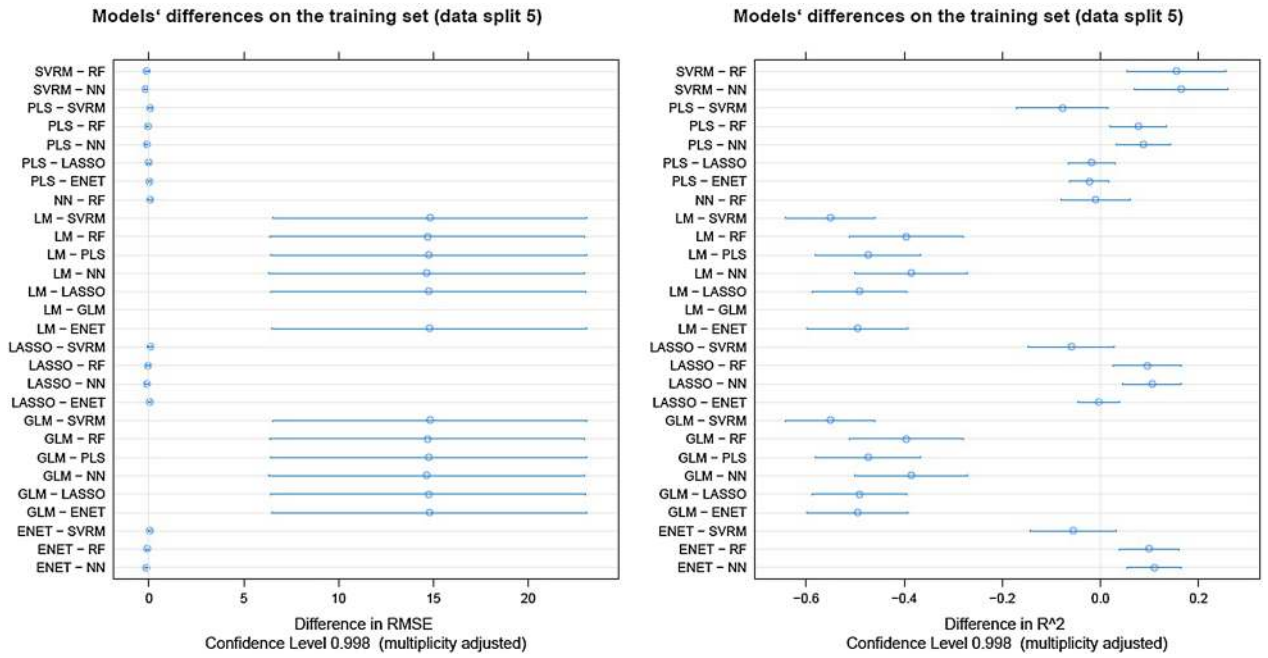

Fig. 7 Models' differences for protein corona optimal training set (data split 5). Differences for R^2 and RMSE values among the applied models are presented for the trimmed corona data set (76 proteins). The average performance with two-sided confidence limits are plotted as derived by the Student t-test with Bonferroni multiplicity correction

SVM-RFE, RF-RFE, ENET. As can be seen from the tables, the best performance model and the best averaged model is ENET in this case, keeping on average 8.8 variables from the data including the two important variables (ΔH^c, χ^c) selected in the original publication. The ENET averaged statistics for 10 splits of the data are $R^2_{test} = 0.746$, $R^2_{CV} = 0.933$, which are very similar to the values reported by the authors. The best individual split value is equal to $R^2_{test} = 0.998$ for ENET model with eight variables including the final two suggested by the authors (LOO at the eighth split of the data). The boxplots in Fig. 8 show the resampling values of $RMSE_{CV}$ and R^2_{CV} values in the training set for split eight, where only methodologies with the same resampling scheme are included in the graph. Figure 8 also includes the fit of the selected model ENET for the same data split.

Use case 3: RRegrs application on aquatic toxicity data set

The authors of [51] developed a QSAR model based on 546 organic molecules, to predict acute aquatic toxicity towards Daphnia magna, which is the organism preferred for short-term aquatic toxicity testing according to REACH [52]. Ad hoc-designed workflows were used for data curation and filtering, as well as for the extraction of LC_{50} data, which in this case is defined to be the concentration that causes death in 50 % of test Daphnia magna over a test duration of 48 h. For modelling purposes the $-\log(LC_{50})$ values were considered as the dependent variable to be predicted. Other experimental data on aquatic toxicity were retrieved from three databases and available scientific publications, as well as one-dimensional (1-D) and2-D molecular descriptors implemented with DRAGON software [51], resulting in a total of 2, 187 molecular descriptors.

A modified k-Nearest Neighbour (kNN) strategy coupled with GA algorithms was used to select the relevant molecular descriptors. The final data set comprised of 546 organic molecules and a set of 201 descriptors. The GA-kNN strategy was implemented with a threshold on the average Mahalanobis distance from the k nearest neighbours, so that only molecules satisfying the threshold criterion were predicted. Particularly, predictions for molecules with an average distance greater than 1.26 from their three neighbours, were considered to be outside of the applicability domain. The training molecules exceeding the threshold did not contribute to the model's statistics, but were not removed from the data set. The final model showed good performance when the average distance threshold was applied, namely $R^2_{CV} = R^2_{test} = 0.78$ (5-fold CV), $R^2 = 0.72$. The model selected eight molecular descriptors that encoded information about lipophilicity, the formation of H-bonds, polar surface area, polarisability, nucleophilicity and electrophilicity. When no distance threshold is applied,

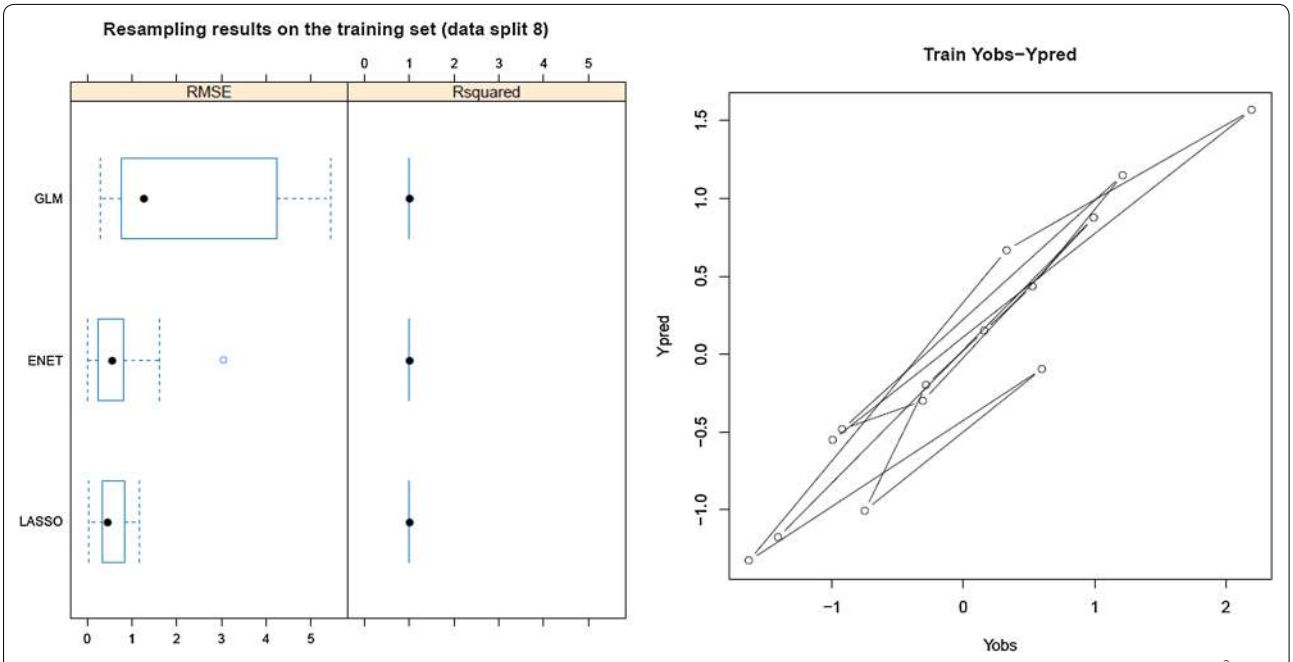

Fig. 8 Results on MeOx data set (data split 8). For the three selected methodologies we show the resampling distributions for the RMSE and R^2 statistics. On the right hand-side, we show the observed versus the predicted values for the ENET model in the training set

the corresponding values are: $R^2 = 0.60$, $R^2_{CV} = 0.61$, $R^2_{test} = 0.43$.

Tables 4 and 5 show the results for the final set of eight descriptors for 546 organic molecules: the averaged values across data splits and the best model statistics for all data splits are presented. RRegrs is applied using normalization and filtering options. Ten random splits of the data were performed (75 % train and 25 % test) along with ten Y-randomization runs for the best model. As can be seen from the tables the best performance model and the best averaged model is SVRM in this case, keeping all 8 descriptors in the data. The SVRM averaged statistics for ten splits of the data are $R^2_{CV} = 0.556$ (10-fold CV), $R^2_{test} = 0.537$, which are close to the ones reported by the authors when the distance threshold is not applied. The adjusted $R^2 = 0.7$, exceeding the 0.6 value reported without the distance threshold application, and still approaching the 0.78 value when the distance threshold was applied. The best individual split value is reported to be $R^2_{test} = 0.657$ for SVRM model with eight variables (LOO at the second split of the data). Figure 9 shows the differences between the various models in terms of R^2_{CV} and $RMSE_{CV}$ values in the training set of the second data split, i.e. the train data where the highest R^2_{test} was observed. We can observe that the PLS model differs from all others, having the worst performance, whilst LM, GLM, LASSO, and ENET models have very similar performances.

Conclusions

This paper introduces RRegrs as a new computer-aided model selection framework using a single R function call. The aim of RRegrs is to automatically obtain the best regression model given the data set, and the set of all ten regression models available, after an extensive search of the model space. A fully validated procedure is suggested where data are split in training and test sets, ten times by default, capturing any variability or inconsistency in the data. The best model is then found across different data splits and cross-validation schemes, based on the averaged data splits statistics. RRegrs produces easily accessible summary files that provide an overview of model details and allows methodology comparisons using the same statistics, enabling QSAR model selection. These direct comparisons are built on top of the caret package, and in that respect provide a useful flexibility for all users. However, use of this package does not require advanced knowledge of R, while, on the other hand, experienced R users can easily modify and extend the package with additional algorithms of choice. The single function call makes it easy to integrate into larger QSAR and in silico molecular screening studies. The new tool was tested with five standard data sets from several domains and three use cases originating from cheminformatics and nanotoxicity, showing good performance in all cases. RRegrs is open source and available from https://www.github.com/enanomapper/RRegrs (doi:10.5281/zenodo.21946).

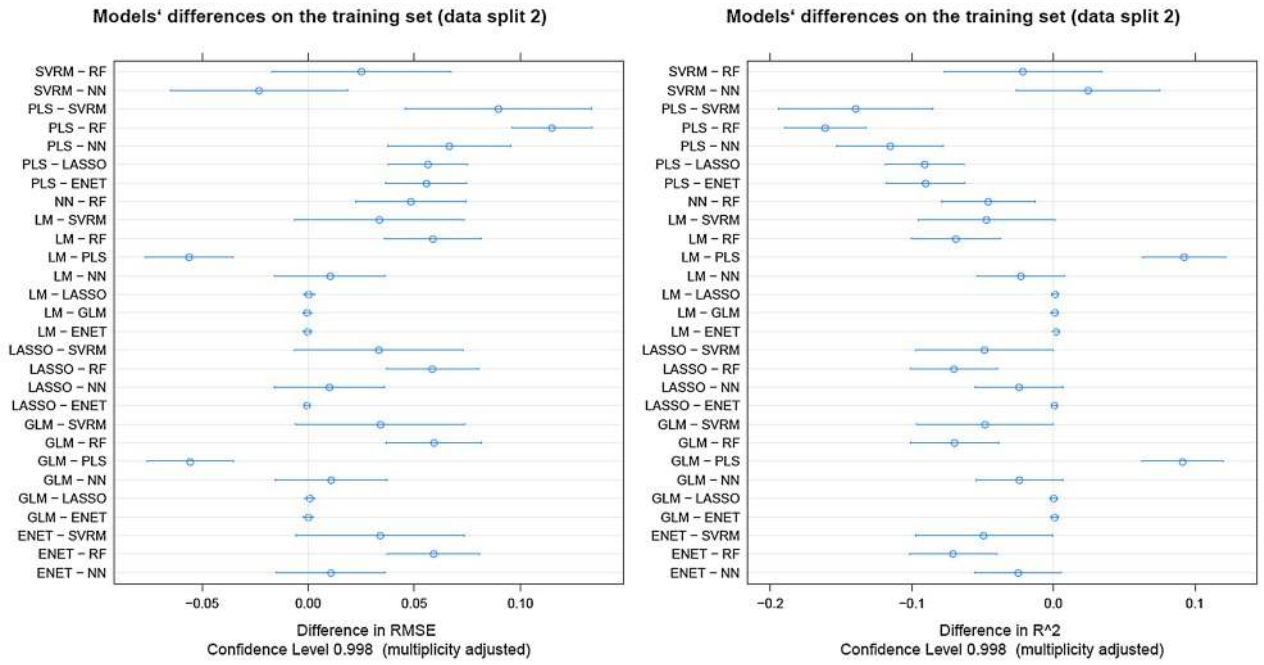

Fig. 9 Models' differences for toxicity training set (data split 2). Differences for R^2 and RMSE values among the applied models are presented. The average performance with two-sided confidence limits are plotted as derived by the Student t-test with Bonferroni multiplicity correction. We can observe small differences for LM, LASSO, ENET models, whereas the performance of the best model, SVRM, appears to be close to that of RF and NN models

Methods

Regression methods in RRegrs

The RRegrs tool is using ten different linear and non-linear regression models briefly described in this section, to explore the model space. The most basic model in this package is the LM model [33]. Variable selection could improve the result of prediction in regression models. For that reason we have included a generalized linear model, denoted by GLM, which selects variables that minimize the AIC score. LASSO and ENET are also penalizing the number of variables via an embedded minimization process [27, 28].

Apart from the standard regression methodologies included in RRegrs, other methods that focus on specific characteristics of the data are included. PLS uses linear projections of input and output sets, which is a useful strategy when many of the inputs are correlated. PLS coefficient optimization algorithm improves previous regression coefficient algorithms because the search path is directed to high variance and high correlations paths [26]. The SVMR algorithm attempts to find the hyperplane that separates the positive and negative samples, practically allowing a non-linear solution to a regression problem by transforming the data to a hyperdimensional feature space using the kernel functions [53, 54]. SVMR in RRegrs uses the radial function or Gaussian function. RRegrs package also allows the use of a support vector regression model where the w^2 (the square of SVM hyperplane weight vector) measures the importance of each feature [29].

RRegrs includes two additional learning algorithms, namely NN and ensemble RF. NN is usually defined as a network of a large number of connected neurons (simple processors), which produce good results with imprecise and complicated data [30]. RF is a bagging method constructing decision trees based on the random subspace method [31].

Finally, we have included two of the best performing methodologies with extra feature selection characteristics. Particularly, because the SVM and RF methods can be time-consuming, we have considered their implementation with random feature elimination (RFE), a feature selection method also introduced in caret where less important features are sequentially removed from the model until optimal performance is reached [32]. The two methods are here labeled as RF-RFE and SVM-RFE. Further details for the functions' main parameters are available in the online tutorial of the RRegrs package.

Model optimization

Two CV schemes are employed within RRegrs, namely 10-fold repeated CV and LOO CV. In the case of repeated CV, we run ten repeats of 10-fold CV for all models except SVM-RFE (3-folds, one repeat) and RF-RFE (5-folds, one repeat), which are particularly time-consuming methods. The procedure followed by caret and also introduced in RRegrs tool, randomly splits the data in K distinct blocks of roughly equal size (K = 10, 3, 5 depending on the method). Each block of data is

left out sequentially, and a model is fit to the remaining of the data; this model is used to predict the held out block. The process is repeated where for each repetition a random proportion of the data are used to train the model (default value is 0.75) while the remainder is used for testing the models. Average performance across the number of repeats are reported: R^2_{test}, $RMSE_{test,}$, R^2_{CV}, R^2_{LOO}, RM $SE_{C\,V}$, RM SE_{LOO}. The best model is selected based on the averaged R^2_{test} value; if multiple models only differ by $\leq$0.005 from the best R^2_{test}value, the model with the lowest $RMSE_{test}$ statistic is selected.

In order to further validate RRegrs test results, Y-randomization is applied to the best model found. For the last data split and the best model found, RRegrs performs Y-randomization for the 10-fold repeated CV scheme, and compares R^2_{test} values to the best model corresponding value.

RRegrs tested data sets

RRegrs has been used to find the best regression models for eight data sets: three from nano- and cheminformatics (use cases), and five standard data sets from different fields. The standard data sets have been downloaded from UC Irvine machine learning repository [35]: housing [36], computer hardware, wine quality [37], automobile [38] and Parkinsons telemonitoring [39] data sets. The non-numeric columns have been eliminated, whereas the first column is the dependent variable (output of the model). The number of initial features/cases are the following: 13/506 for Housing data set, 6/209 for Computer Hardware data set, 11/1, 599 for Red Wine Quality data set, 14/195 for Automobile data set, and 19/5, 875 for Parkinson telemonitoring data set. The use case data sets were derived from their original publications; the initial number of features and cases are the following: the protein corona data set [49] has 129 features and 84 cases, the metal oxide data set [50] has 32 features and 18 cases and the toxicity data set [51] contains 8 features, and 546 cases.

RRegrs call in R

The main function of RRegrs (also called RRegrs) permits the call of the entire RRegrs methodology in a single line. All details about functions' main parameters are given in the R package documentation. All these parameters have default values. The default values imply a default location for the output files, execution of all modelling steps (removal of NA, and near zero variance features, and of correlated features), normalization of the data set, ten splits, ten Y-randomization steps, and running of all ten regression methods. The user can alter any step or parameter of the RRegrs methodology.

The following examples show simple calls of the RRegrs() function using a specific dataset file entitled "MyDataSet.csv" that it should be provided by the user:

```
> library(RRegrs)
>
> # Run RRegrs with all default parameters
> #  default data set file ("ds.House.csv") and
> #  working directory ("DataResults")
> #  run all regression methods
> #  10 splittings, 100 times Y-randomization,
> #  no parallel support for CPU cores
> RRegrsResults = RRegrs()
>
> # Run RRegrs for a specific data set file with default parameters
> #  including the default directory ("DataResults")
> RRegrsResults = RRegrs(DataFileName="MyDataSet.csv")
>
> # Run RRegrs for a specific data set file ("MyDataSet.csv") and
> # working folder ("MyResultsFolder"); both should exist
> # the rest of RRegrs parameters have default values
> RRegrsResults = RRegrs(DataFileName="MyDataSet.csv",
>                        PathDataSet="MyResultsFolder")
```

The output variable RRegrsResults is a complex object which contains the object of the fitted models and the main statistics for each regression model. Details about each function are presented into the tutorial of the RRegrs package.

The following example could be used to test the RRegrs package using a dataset file from RRegrs GitHub URL:

```
> library(RRegrs)
>
> # Create default working directory "DataResults"
> dir.create("DataResults")
>
> # Get Housing dataset "ds.House.csv" from RRegrs GitHub
> # in the default RRegrs directory "DataResults"
> # Windows
> setInternet2(TRUE)
>  download.file("https://raw.githubusercontent.com/enanomapper/RRegrs/
master/TEST/ds/ds.House.csv","DataResults/ds.House.csv",method="auto",
quiet=FALSE)
>
> # Linux/Mac OS
>  download.file("https://raw.githubusercontent.com/enanomapper/RRegrs/
master/TEST/ds/ds.House.csv", "DataResults/ds.House.csv",metho d="curl",
quiet=FALSE)
> # RRegrs call with default parameters
> RRegrsResults = RRegrs()
```

Availability and requirements

- Project name: RRegrs
- Project home page: RRegrs
- Operating system(s): Platform-independent
- Programming language: R programming language
- Other requirements: R 3.1.0 or higher
- License: NewBSD or MIT
- Any restrictions to use by non-academics: none other than those defined by the license

Abbreviations

RRegrs: R regressions (package); QSAR: quantitative structure-activity relationship; R^2: R-squared; RMSE: root-mean-square error; AIC: akaike information criterion; LM: linear multi-regression; GLM: generalized linear model with stepwise feature selection; PLS: partial least squares regression; LASSO: Lasso regression; ENET: elastic net regression; SVRM: support vector machine using radial functions; NN: neural networks regression; RF: random forest; RF-RFE: random forest recursive feature elimination; SVM-RFE: support vector machines recursive feature elimination; CSV: comma-separated values file format; PDF: portable document format of files; PNG: portable network graphics file format; NP: nanoparticle.

Authors' contributions

GT and CRM equally contributed in the conceptual framework, implementation of the framework and writing the paper, supplementary information, and R manual files. JAS and CFL contributed in implementing ENET, RF, SVMRFE, RFRFE methods, writing the corresponding supporting information, and R manual files. HS and ELW contributed in the conceptual framework and writing of the paper. ELW built and maintains the RRegrs R package. All authors read and approved the final manuscript.

Author details

[1] School of Chemical Engineering, National Technical University of Athens, 9 Heroon Polytechneiou Street, Zografou Campus, 15780 Athens, Greece. [2] Computer Science Faculty, University of A Coruna, Campus Elviña, s/n, 15071 A Coruña, Spain. [3] Department of Bioinformatics-BiGCaT, NUTRIM, Maastricht University, P.O. Box 616, UNS50 Box 19, 6200 MD Maastricht, The Netherlands. [4] Stanford Cancer Institute, Stanford University, C.J.Huang Building, 780 Welch Road, Palo Alto, CA 94304, USA.

Acknowledgements

The eNanoMapper project is funded by the European Union's Seventh Framework Programme for research, technological development and demonstration (FP7-NMP-2013-SMALL-7) under grant agreement no 604134. The authors acknowledge the support by the Galician Network of Drugs R+D REGID (Xunta de Galicia R2014/025) and by "Collaborative Project on Medical Informatics (CIMED)" P I 13/00280 funded by the Carlos III Health Institute from the Spanish National plan for Scientific and Technical Research and Innovation 2013–2016 and the European Regional Development Funds (FEDER).

Compliance with ethical guidelines

Competing interests

The authors declare that they have no competing interests.

Received: 7 April 2015 Accepted: 24 August 2015
Published online: 15 September 2015

References

1. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The WEKA data mining software: an update. SIGKDD Explor. Newsl 11(1):10–18
2. Hofmann M, Klinkenberg R (2013) RapidMiner: Data mining use cases and business analytics applications. Chapman and Hall, CRC Press, Boca Raton
3. Alcal´a-Fdez J, S´anchez L, Garc´ıa S, del Jesu´s MJ, Ventura S, Garrell J, Otero J, Romero C, Bacardit J, Rivas VM et al (2009) KEEL: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput 13(3):307–318
4. Demšar J, Zupan B, Leban G, Curk T (2004) Orange: From experimental machine learning to interactive data mining. Springer, Berlin Heidelberg
5. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
6. Eklund M, Spjuth O, Wikberg JE (2008) The c1c2: a framework for simultaneous model selection and assessment. BMC Bioinform 9(1):360
7. Berthold MR, Cebron N, Dill F, Gabriel TR, Kotter T, Meinl T, Ohl P, Sieb C, Thiel K, Wiswedel B (2008) KNIME: the Konstanz information miner. Springer, Berlin, Heidelberg

8. Hardy B, Douglas N, Helma C, Rautenberg M, Jeliazkova N, Jeliazkov V, Nikolova I, Benigni R, Tcheremenskaia O, Kramer S et al (2010) Collaborative development of predictive toxicology applications. J Cheminform 2(1):1–29
9. Sushko I, Novotarskyi S, Körner R, Pandey AK, Rupp M, Teetz W, Brandmaier S, Abdelaziz A, Prokopenko VV, Tanchuk VY et al (2011) Online chemical modeling environment (OCHEM): web platform for data storage, model development and publishing of chemical information. J Comp Aided Mol Design 25(6):533–554
10. Cases M, Briggs K, Steger-Hartmann T, Pognan F, Marc P, Kleinöder T, Schwab CH, Pastor M, Wichard J, Sanz F (2014) The eTOX data-sharing project to advance in silico drug-induced toxicity prediction. Int J Mol Sci 15(11):21136–21154
11. Ekins S (2014) Progress in computational toxicology. J Pharmacol Toxicol Methods 69(2):115–140
12. Cherkasov A, Muratov EN, Fourches D, Varnek A, Baskin II, Cronin M, Dearden J, Gramatica P, Martin YC, Todeschini R, Consonni V, Kuzmin VE, Cramer R, Benigni R, Yang C, Rathman J, Terfloth L, Gasteiger J, Richard A, Tropsha A (2014) QSAR modeling: where have you been? where are you going to? J Med Chem 57(12):4977–5010
13. Fjodorova N, Vracko M, Novic M, Roncaglioni A, Benfenati E (2010) New public QSAR model for carcinogenicity. Chem Cent J 4(Suppl 1):3
14. US Environmental Protection Agency (2012) EPI Suite software. http://www.epa.gov/oppt/exposure/pubs/episuitedl.htm
15. US Environmental Protection Agency (2012) Toxicity estimation software tool (TEST). http://www.epa.gov/nrmrl/std/qsar/qsar.html#TEST
16. National Academy of Sciences of Ukraine (2012) QSAR4u. http://www.qsar4u.com/
17. de Oliveira DB, Gaudio AC (2000) Buildqsar: A new computer program for qsar analysis. Quant Struct Act Relat 19(6):599–601
18. OECD (2011) OECD QSAR Toolbox. http://www.oecd.org/chemicalsafety/risk-assessment/theoecdqsartoolbox.htm
19. Stålring JC, Carlsson L, Almeida P, Boyer S (2011) AZOrange-High performance open source machine learning for QSAR modeling in a graphical programming environment. J Cheminform 3:28
20. Spjuth O, Georgiev V, Carlsson L, Alvarsson J, Berg A, Willighagen E, Wikberg JES, Eklund M (2013) Bioclipse-R: Integrating management and visualization of life science data with statistical analysis. Bioinformatics 29(2):286–9
21. Team RC et al (2011) R: A language and environment for statistical computing. The R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. http://www.R-project.org/
22. Venables WN, Ripley BD (2002) Modern Applied Statistics with S, 4th edn. Springer, New York
23. Kuhn M (2008) Building predictive models in R using the caret package. J Stat Softw 28(5):1–26
24. Wickham H (2011) testthat: get started with testing. R J 3:5–10
25. Hocking RR (1976) The Analysis And Selection Of Variables In Linear Regression. Biometrics 32(1):1–49
26. Wold S, Ruhe A, Wold H, Dunn W III (1984) The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses. SIAM J Sci Stat Comput 5(3):735–743
27. Tibshirani R (1994) Regression selection and shrinkage via the lasso. J R Stat Soc Ser B Stat Methodol 58:267–288
28. Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J R Stat Soc Ser B Stat Methodol 67:301–320
29. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Mach Learn 46(1–3):389–422
30. Bishop CM (1995) Neural networks for pattern recognition. Oxford University Press Inc, New York
31. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
32. Saeys Y, Inza I, Larrñaaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 23(19):2507–2517
33. Dobson AJ, Barnett AG (2008) An introduction to generalized linear models. Chapman and Hall, CRC Press, Boca Raton
34. Guha R (2007) Chemical informatics functionality in R. J Stat Softw 18(5):1–16
35. Bache K, Lichman M (2013) UCI machine learning repository. http://www.archive.ics.uci.edu/ml
36. Harrison D, Rubinfeld DL (1978) Hedonic housing prices and the demand for clean air. J Environ Econ Manage 5(1):81–102
37. Cortez P, Cerdeira A, Almeida F, Matos T, Reis J (2009) Modeling wine preferences by data mining from physicochemical properties. Decis Support Syst 47(4):547–553
38. Kibler D, Aha DW, Albert MK (1989) Instance-based prediction of real-valued attributes. Comput Intell 5(2):51–57
39. Tsanas A, Little MA, McSharry PE, Ramig LO (2010) Accurate telemonitoring of parkinsons disease progression by noninvasive speech tests. Biomed Eng IEEE Trans 57(4):884–893
40. Zhou Z-H, Wu J-X, Tang W, Chen Z (2001) Combining regression estimators: GA-based selective neural network ensemble. Int J Comput Intell Appl 1:341
41. Hershberger DE, Kargupta H (2001) Distributed multivariate regression using wavelet-based collective data mining. J Parallel Distrib Comput 61:372
42. Law MHC, Kwok JT (2001) Applying the bayesian evidence framework to υ-support vector regression. In: ECML, pp 312
43. Merz CJ, Pazzani MJ (1999) A principal components approach to combining regression estimates. Mach Learn 36:9
44. Guvenir HA, Uysal I (2000) Regression on feature projections. Knowl Based Syst 13(4):207–214
45. Hoegaerts L, Suykens JAK, Vandewalle J, De Moor B (2005) Subset based least squares subspace regression in RKHS. Neurocomputing 63:293–323
46. Shevade SK, Keerthi SS, Bhattacharyya C, Murthy KRK (2000) Improvements to the smo algorithm for svm regression. Neural Netw IEEE Trans 11(5):1188–1193
47. Ge C, Du J, Zhao L, Wang L, Liu Y, Li D, Yang Y, Zhou R, Zhao Y, Chai Z et al (2011) Binding of blood proteins to carbon nanotubes reduces cytotoxicity. Proc Natl Acad Sci 108(41):16968–16973
48. Lesniak A, Fenaroli F, Monopoli MP, Aberg C, Dawson KA, Salvati A (2012) Effects of the presence or absence of a protein corona on silica nanoparticle uptake and impact on cells. ACS Nano 6(7):5845–5857
49. Walkey CD, Olsen JB, Song F, Liu R, Guo H, Olsen DWH, Cohen Y, Emili A, Chan WCW (2014) Protein corona fingerprinting predicts the cellular interaction of gold and silver nanoparticles. ACS Nano 8(3):2439–2455
50. Gajewicz A, Schaeublin N, Rasulev B, Hussain S, Leszczynska D, Puzyn T, Leszczynski J (2015) Towards understanding mechanisms governing cytotoxicity of metal oxides nanoparticles: Hints from nano-QSAR studies. Nanotoxicology 9(3):313–325
51. Cassotti M, Ballabio D, Consonni V, Mauri A, Tetko I, Todeschini R (2014) Prediction of acute aquatic toxicity toward daphnia magna by using the ga-knn method. Altern Lab Anim ATLA 42(1):31–41
52. Lahl U, Gundert-Remy U (2008) The use of (Q)SAR methods in the context of REACH. Toxicol Mech Methods 18(2–3):149–158
53. Brereton RG, Lloyd GR (2010) Support vector machines for classification and regression. Analyst 135(2):230–267
54. Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. Stat Comput 14(3):199–222