J Comput Aided Mol Des (2007) 21:651–664
DOI 10.1007/s10822-007-9160-9

Estimating the domain of applicability for machine learning QSAR models: a study on aqueous solubility of drug discovery molecules

Timon Sebastian Schroeter · Anton Schwaighofer · Sebastian Mika · Antonius Ter Laak · Detlev Suelzle · Ursula Ganzer · Nikolaus Heinrich · Klaus-Robert Müller

Received: 12 March 2007 / Accepted: 11 June 2007 / Published online: 1 December 2007

Abstract We investigate the use of different Machine Learning methods to construct models for aqueous solubility. Models are based on about 4000 compounds, including an in-house set of 632 drug discovery molecules of Bayer Schering Pharma. For each method, we also consider an appropriate method to obtain error bars, in order to estimate the domain of applicability (DOA) for each model. Here, we investigate error bars from a Bayesian model (Gaussian Process (GP)), an ensemble based approach (Random Forest), and approaches based on the Mahalanobis distance to training data (for Support Vector Machine and Ridge Regression models). We evaluate all approaches in terms of their prediction accuracy (in cross-validation, and on an external validation set of 536 molecules) and in how far the individual error bars can faithfully represent the actual prediction error.

Keywords Solubility · Aqueous · Machine learning · Drug discovery · Domain of applicability · Error bar · Error estimation · Gaussian Process · Bayesian modeling · Random forest · Ensemble · Decision tree · Support vector machine · Ridge regression · Distance

This article has previously been published in issue 21/9, under DOI 10.1007/s10822-007-9125-z.

T. S. Schroeter (✉) · A. Schwaighofer · K.-R. Müller
Fraunhofer FIRST, Kekuléstraße 7, 12489 Berlin, Germany
e-mail: timon.schroeter@first.fraunhofer.de

T. S. Schroeter · K.-R. Müller
Department of Computer Science, Technical University of Berlin, Franklinstraße 28/29, 10587 Berlin, Germany

S. Mika
idalab GmbH, Sophienstraße 24, 10178 Berlin, Germany

A. Ter Laak · D. Suelzle · U. Ganzer · N. Heinrich
Research Laboratories of Bayer Schering Pharma AG, Müllerstraße 178, 13342 Berlin, Germany

Introduction

Aqueous solubility is of high importance to drug discovery and many other areas of chemical research. A lot of research has been devoted to developing in-silico models to predict aqueous solubility directly from a compound's structure (see [1–5] and references therein). As Goldman et al. [6] point out in their recent review, quite a few Machine Learning methods have matured to the point where they can be used by non-experts and have found widespread use in computational chemistry. However, the domain of applicability (DOA) of every conceivable QSAR model is limited, leading to unreliable predictions outside the DOA. When applying any kind of model, it is therefore essential to know the reliability of each individual prediction [7–10].

In Schwaighofer et al. [1] we reported about the first model for aqueous solubility based on the Gaussian Process (GP) methodology. The model was based on solubility measurements for about 4,000 electrolytes and non-electrolytes. Also, the well known dataset by Huuskonen was used for a performance comparison with results from literature.

In this paper we present the results of modeling aqueous solubility using four different Machine Learning methods. Our primary focus is on the different methods to estimate the domain of applicability (DOA) of each model. We consider ensemble based, distance based and Bayesian approaches to obtaining the confidence estimates. The evaluations are carried out on both public data and in-house data from Bayer Schering Pharma. Models were first evaluated in cross

validation, and subsequently with an external validation set (the data that was used to conduct the "blind test" in [1]).

Estimating the domain of applicability of models

For the practical use of all types of machine learning based QSAR models, the "domain of applicability" is a key question. How can, for example, a model estimate the solubility of a steroid correctly, if it has not been trained on steroids? Still, this important question has not received much attention for a long time. The methods proposed so far have recently been reviewed [7–9]. In this section we will briefly discuss the most relevant methods and then focus on the methods used in this study.

Range based methods check whether descriptors of test set compounds exceed the range of the respective descriptor covered in training [9, 11]. Rather than just raising a warning if one or more ranges are exceeded, some authors counted the number of descriptors that were out of range [10]. By considering each descriptor individually, range based methods span a hyper-cuboid in the space of descriptors. Depending on the actual distribution of the training data, this cuboid may contain a lot of empty space, i.e., it is very unlikely that the actual data uniformly populates the hyper-cuboid. Therefore, other *geometric methods* have been proposed. By considering e.g. the convex hull of the training data in the space of descriptors, the amount of empty space covered can be significantly reduced [7]. One should also bear in mind that range based and geometric methods suffer from a serious drawback: Empty spaces in in the descriptor space spanned by the training compounds can not be detected, thus predictions in these regions will not be flagged as unreliable.

Distance based methods (including the so called extrapolation measures) exist in many variants [7–10, 12]. Different distance measures (Euclidean distance, Mahalanobis distance etc.) can be used to calculate the distance from each test set compound to its closest neighbor(s) in the training set. Vice versa, one can define a threshold for the distance and count the number of training compounds closer than the threshold. Rather than considering distances to individual training data points, Hotelling's test or the leverage consider the distance to the whole training set, by assuming that the data follows a Gaussian distribution in descriptor space and computing the Mahalanobis distance. Tetko has argued in [8] that descriptors have different relevance for predicting a specific property. Instead of general distance measures, property specific distances (resp. similarities) should be used.

Probability density distribution based methods are another intuitive and appealing way of estimating the reliability of model predictions [7]. Predictions made in regions of a high density of training data can be expected to be more accurate than predictions in regions of low density. Clearly, such a methodology can identify "holes" in the data distribution. However, density estimation for high-dimensional data is recognized as an extremely difficult problem [13], and may thus severely limit the practical applicability of this approach.

Estimating the DOA using *ensemble methods* is based on the idea of comparing the predictions of a number of models trained on different sets of data. If these sets are generated by sampling from a larger set of available training data, the models will tend to agree in descriptor space regions where a lot of training compounds are available, but will disagree in sparsely populated regions. Alternatively, training sets for the individual models may be generated by adding noise to existing descriptors, such that each model gets a slightly modified version of the whole set of descriptors. In this case the models will agree in regions where the predictions are not very sensitive to small changes in the descriptors, and they will disagree in descriptor space regions where the sensitivity with respect to smal descriptor changes is large. ANNs [4, 8, 10, 12, 14] and decision trees [10, 11] (resp. their ensemble counterpart random forests) have been most commonly used to estimate the DOA from ensembles. The methodology can, in principle, be applied to any type of model.

Kühne et al. [15] used a *library* of new measurements (not used in model building) in the following way: For each compound to be predicted, they found a number of nearest neighbors in their library. Out of a number of available models they then chose the model that gave the best predictions for the neighbors of the unknown compound. Rather than choosing one out of many methods, this approach could also be used to estimate the specific model error for the test set compounds. This will work well, if there is close enough neighbors in the library. However, if a large library of compounds is available and predictions for compounds similar to the library are sought, using the new measurements to train a new model will result in more accurate predictions. This will usually be more desirable than using the library just for DOA estimation.

The idea behind *Bayesian methods* is to treat all parameters in the model as uncertain, and describe them via probability distributions. In such a framework, the model output is also a probability distribution (rather than a point prediction) which readily includes the required confidence estimates. The most simple and also most widely used method is the naive Bayes classifier [16, 17]. Gaussian Process models are an example of a more sophisticated Bayesian method, that will be discussed in Section "Gaussian Process models".

In this study, we employ methods based on distances and ensembles of models, along with a Bayesian learning

method. These three groups of models do not suffer from the "empty space" problem of range based and geometric methods. Furthermore, they can quantify their confidence (rather than just raising a warning for possibly unreliable predictions). As we will discuss later, the confidence estimates can be presented in a format that is intuitively understandable to chemists and other professionals.

Illustrative examples

Figure 1 is intended to give an intuition as to how the different methods of error estimation work. We apply each model used in this study along with its method of error estimation to the same one dimensional example. The function to be learned (the cosine function) is shown as a green line in each subplot. Models are trained using the nine points marked by black crosses. These "measurements" are generated by randomly choosing x-values and evaluating the cosine function at these points. As in real life, measurements are noisy: Normally distributed random numbers in the range between –0.3 and +0.3 are added to the y values.

The linear Ridge Regression model (subplot a) cannot fit the non-linear relationship in the example. Errors estimated using the distance of test points to training points are therefore misleading: Close to the training points, low errors are predicted, but the actual errors are quite large.

The random forest (subplot b) produces a reasonable fit to the training points, but does not extrapolate well. The same holds for the error predictions: Considering the noise applied when generating the "measurements", predicted errors are acceptable in the range of the training points. They do, however, not increase when the model is queried far from the training points and therefore underestimate the actual errors.

The Support Vector Machine (subplot c) captures the non-linearity in the input data and extrapolates reasonably. The distance based error estimation produces too small error bars in the region close the training points and somewhat underestimates the errors in the extrapolation region. In this example, an exponential function would have allowed for a more accurate distance based error estimation. However, we found that for real life data the linear and exponential functions worked equally well and decided for the simple linear function (see Section "Individual error estimation for interactive use"). Consequently, we use a linear function in the present example.

The GP (subplot d) nicely captures the non-linearity in the input data and extrapolates reasonably. Predicted errors

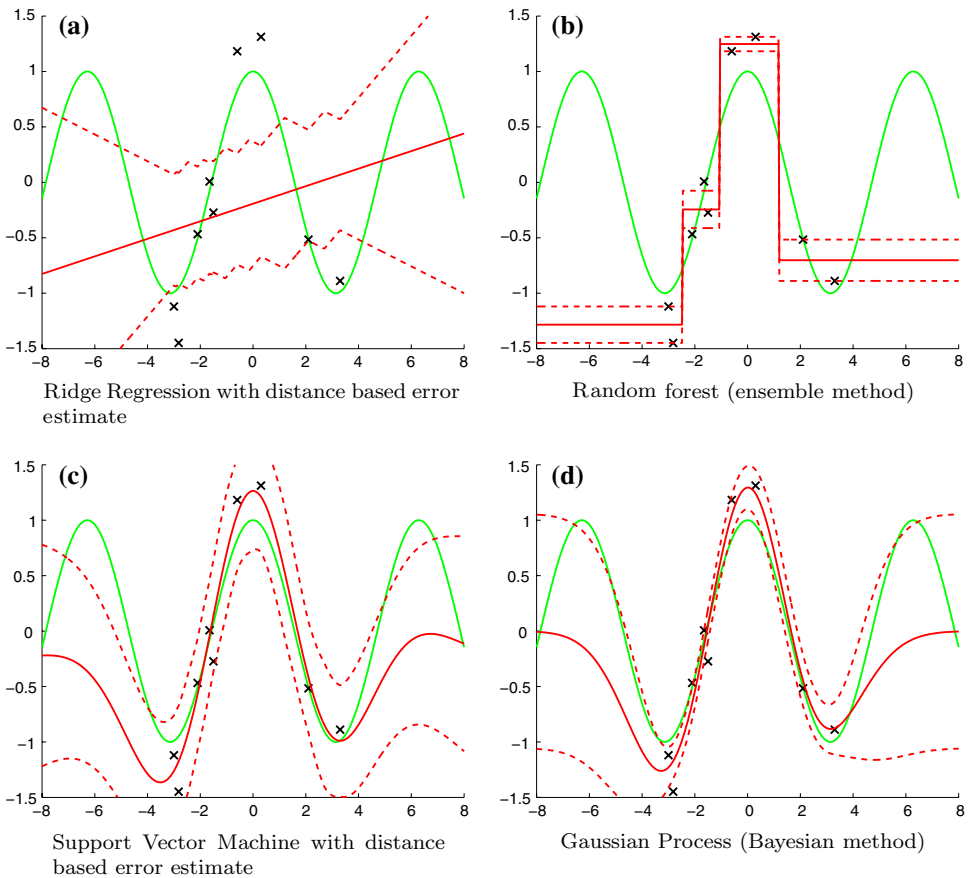


Fig. 1 Noisy measurements (black crosses) generated from a non-linear function (green line) are modeled using four different regression methods (**a**–**d**). Model predictions are drawn as solid red lines. Dashed red lines indicate errors estimated by the respective model (in case of the Gaussian Process and random forest) or by a distance based approach (in case of the Support Vector Machine and Ridge Regression model)

are adequately small in the region close to the training points and increase strong enough to avoid underestimation of errors in the extrapolation region.

Methods and data

Methodology overview

For each molecule, the 3D structure of one conformation in its neutral form is predicted using the program Corina [18]. From this 3D structure, 1,664 Dragon [19] descriptors are generated. Based on solubility measurements and molecular descriptors of several thousand compounds, different machine learning models are used to infer the relationship between the descriptors and the solubility for the given set of training data. When applying this model to a previously unseen compound, descriptors are calculated as described above and passed on to the trained model, which in turn produces an estimate of the solubility together with a confidence estimate (error bar).

Different kinds of solubility

Aqueous solubility is defined as the maximum amount of compound dissolved in water under equilibrium conditions. For electrolytes this property is strongly pH dependent, so we need more precise definitions: The solubility in a buffer solution at a specific pH is called *buffer solubility* or *apparent solubility*. The *intrinsic solubility* is the solubility of a compound in its neutral form. Intrinsic solubility is only reached at a pH where almost 100% of the compound is present in its neutral form. The *pure solubility* or *native solubility* can be observed when adding the compound to pure (unbuffered) water. The pH of the solution typically changes during this process. The *kinetic solubility* is the concentration when an induced precipitate first appears in a solution. In this paper, we focus only on experimental data of buffer solubility and native solubility.

Data preparation

Subsequently, we will only briefly describe these data sets, for detailed descriptions see ref [1].

- Data Set 1: Data from Physprop [20] and Beilstein [21], with 5,652 measurements for 3,307 individual compounds. All of these compounds are neutral.
- Data Set 2: "Flask", containing 688 high quality flask measurements of buffer solubility at pH 7 to 7.4 for 632 drug discovery molecules. 549 of these compounds are electrolytes.
- Data Set 3: "Flask external", with 536 measurements, following the same experimental protocol as the "Flask" set. This set has been recorded after the "Flask" set, and thus includes compounds from new projects. In our initial study, it was used to conduct a blind test of the GPsol model [1]. In the present paper, it serves as an external validation set (i.e., it does not influence model building in any form).
- Data Set 4: "Huuskonen". This is the well known benchmark dataset originally extracted from the Aquasol [22] and Physprop [20] databases by Huuskonen [23], and later revised by several authors [24–26]. We used the version at www.vcclab.org (January 2006), but had to restrict to a subset of 1290 of these compounds for which we could compute Dragon descriptors. The dataset contains measurements of pure solubility [27].

For all of the data sets listed here, we ensured that they were pairwise disjoint. If data sets contained multiple experimental values for compounds, the measurements were merged after outlier removal [1].

Training and validation setups

The data sets described in Section "Data preparation" were employed in different setups for model building. The setups are described subsequently, a graphical summary is given in Table 1. In all cases, it was ensured that no

Table 1 Summary of the different setups that are used for performance evaluation. See Section "Training and validation setups" for a description, and Section Data preparation for details on the individual data sets

Setup	Data
Flask	Training: Physprop/Beilstein (3307); Huuskonen neutral (704) — Validation: Flask (632)
Flask external validation	Training: Physprop/Beilstein (3307); Huuskonen neutral (704); Flask (632) — Validation: Flask external (536)
Huuskonen	Training: Huuskonen (1290) — Validation

molecules were used both for model training and evaluation.

Setup "Flask" We evaluate models in leave 50% out cross-validation on "Flask" data (buffer solubility at known pH 7.0 ... 7.4), while always including data set 1 (neutral compounds from Physprop and Beilstein) and the subset of 704 neutral compounds from data set 4 (Huuskonen) in the training set. This is repeated 10 times with different random splits of the "Flask" data.

Setup "Flask external validation" Models are trained on all data from the "Flask" setup and evaluated on the "Flask external" data set. The "Flask external" data is never used for training in any form.

Setup "Huuskonen" Models are evaluated in leave 33% out cross-validation on the "Huuskonen" data set. This is repeated 10 times with different random splits. Leave one third out cross-validation was chosen since it gives training sets of around 860 compounds. This matches well with most studies conducted by other researchers, where models were trained on around 800–880 compounds, see Section "Comparison with Other Approaches" in [1] and references therein.

Molecular descriptors

Starting from the full set of 1,664 Dragon descriptors (20 blocks of descriptors of different nature) [28] we used the weights in the covariance function of GP models (automatic relevance determination) to select the 200 most important descriptors [1]. It turned out that reducing the number of descriptors to 200 does not significantly impact the models performance, but makes the error predictions become too optimistic. We thus used those blocks of descriptors that contained the descriptors that were found to be most relevant (Dragon blocks 1, 2, 6, 9, 12, 15, 16, 17, 18, and 20). A prediction for log D at pH 7, obtained from a model we previously trained on 20,000 log D measurements [29, 30], was found to slightly increase the performance of the models and was therefore included as a descriptor.[1]

Machine Learning methods

Gaussian Process models

Gaussian Process models are a technique from the field of Bayesian statistics. O'Hagan [31] presented one of the seminal work on GPs, a recent book [32] presents an in-depth introduction.

The idea of GP models can be summarized in three graphs, see Fig. 2. We consider an infinitely large family of functions that could potentially model the dependence of solubility (function output, denoted by y) from the descriptor (function input, denoted by $\mathbf{x}$). We describe this space of functions by a *Gaussian Process prior*, thus each of the functions can be seen as a realization of a stochastic process. Twenty-five of the infinitely many functions in the prior are shown in Fig. 2 (left). In the next step we incorporate the experimental data. In Fig. 2 (middle), the available measurements are illustrated by three crosses. The x-position of the cross identifies the molecule by its descriptor value, the y-position is the experimental value. After observing the data, we only retain those functions from the prior that pass through a "tunnel" near the data. The tunnel accounts for the uncertainty in the measurements. In Fig. 2 (right), we summarize the remaining functions by averaging. For each descriptor value (that is, for each new molecule) we can compute the mean of these functions (red line) and the standard deviation. The shaded area encompasses ±2 standard deviations around the mean. Clearly, the size of the shaded area represents the uncertainty, and gives a good estimate for the DOA of the model. Note also that the uncertainty increases on points that are far from the measurements.

All the operations on the infinitely large function space are carried out by integral operations, leaving only few equations that are easy to implement. Note that, in Bayesian inference, we do not choose a single "optimal" function from the prior, but we actually weight them according to how well they match the experimental data.

The Math We assume that solubility can be described by an (unknown) function f that takes a vector $\mathbf{x}$ of d molecular descriptors as input, and outputs the solubility, $f(\mathbf{x})$. As our *a priori* information, we assume that f is a "random function", where functional values $f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)$ for any finite set of n points form a Gaussian distribution. This is closely related to the concept of Brownian motion, which also follows a GP with a specific covariance function. The stochastic process can be fully described by considering pairs of compounds $\mathbf{x}$ and $\mathbf{x}'$. The covariance for this pair is given by the covariance function,

$$\operatorname{cov}(f(\mathbf{x}), f(\mathbf{x}')) = k(\mathbf{x}, \mathbf{x}'), \tag{1}$$

similar to kernel functions in Support Vector Machines [33, 34]. All assumptions about the family of functions f are encoded in the covariance function k. Each of the possible functions f can be seen as one realization of the stochastic process.

[1] For some compounds, experimental values for both solubility and log D where available. For these compounds, we used log D predictions generated using a cross-validation procedure. This means that the predictions were always made using a log D model that has not been trained using the experimental log D value for the respective compound. This is necessary to avoid over-optimistic predictions.

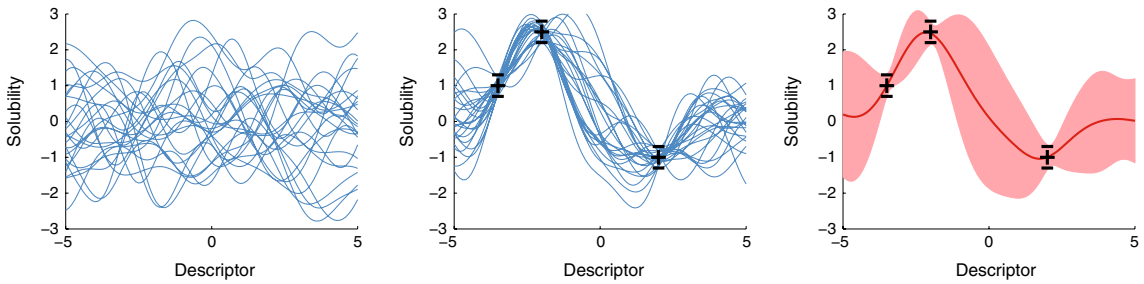


Fig. 2 Modelling with Gaussian Process priors. Please refer to Section "Gaussian Process model" for a detailed description of the graphs

The actual data for n compounds are n descriptor vectors, $\mathbf{x}_1 \ldots \mathbf{x}_n$, (each of length d), together with solubility measurements, $y_1,\ldots\ y_n$. We account for measurement uncertainty by assuming that the n measured values are related to actual solubility by

$$y_i = f(\mathbf{x}_i) + r, \tag{2}$$

where r is Gaussian measurement noise with mean 0 and standard deviation σ (see [1] for extensions with data-dependent measurement noise).

After a few steps of statistical inference (see [1, 32]) we obtain that the predicted solubility for a new compound $\mathbf{x}_*$ follows a Gaussian distribution with mean $\bar{f}(\mathbf{x}_*)$ and standard deviation std $f(\mathbf{x}_*)$, with

$$\bar{f}(\mathbf{x}_*) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_*, \mathbf{x}_i) \tag{3}$$

$$\mathrm{std} f(\mathbf{x}_*) = \sqrt{k(\mathbf{x}_*, \mathbf{x}_*) - \sum_{i=1}^{n}\sum_{j=1}^{n} k(\mathbf{x}_*, \mathbf{x}_i) k(\mathbf{x}_*, \mathbf{x}_j) L_{ij}} \tag{4}$$

Coefficients α_i are found by solving a system of linear equations, $(K + \sigma^2 I)\boldsymbol{\alpha} = y$, with $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. For the standard deviation, L_{ij} are the elements of the matrix $L = (K + \sigma^2\ I)^{-1}$.

We refer to [1, 32] for details on how the model parameters (the measurement noise σ) and parameters of the covariance function k can be inferred from the data.

Support vector machine

In this section we will give a short overview of the idea of SVR (see [35–37] for more details). As for GP models, the goal is to estimate an unknown function f. Given a vector $\mathbf{x}$ of descriptors for a compound, the quantity of interest y (in our case, solubility) can be predicted as $y = f(\mathbf{x})$.

Contrary to the GP approach, which is based on a probabilistic approach to quantify the agreement of the model with the data, SVR is based on the notion of structural risk minimization. The basic idea is as follows: Similar to GP models, we consider a specific class of functions (representing a type of model). Within this class of functions, we aim at finding the function (i.e., finding the model parameters) that minimizes some notion of error. Errors are measured by the so-called loss function. Obviously, there is a trade-off to be made: Using a class of functions that represents an overly complex model, one can achieve an empirical error of zero for any training set and every reasonable loss function (over-fitting, such a function will produce merely random predictions for new compounds). On the other hand, choosing a class of functions that contains only very simple models might not account for the complexity of the problem and will produce predictions that are not accurate either.

Choosing the right function class can be achieved by regularization: Our objective function is the empirical error on the training data, to which we now add a penalty term for the complexity of the solution, and then minimize the sum. For SVR, we use the theory of structural risk minimization as the specific way of regularizing the learning problem. Under the assumption that test data follows the same distribution as the training data, one can prove that this way of choosing the function class will lead to an optimal model. Using the theory of structural risk minimization for linear models is equivalent to choosing the weight vector of the linear regression model such that its vector norm is small whilst minimizing the empirical error.

The Math The goal of (linear) SVR is to find a predictor $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$, such that the empirical error as well as the norm of the weight vector $\mathbf{w}$ are small. The loss is measured by the so-called ε-insensitive loss function which only penalizes deviations from the true target that are larger than some small ε. This problem can conveniently be formulated as the following convex quadratic optimization problem:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i, \\ \text{subject to} \quad & |f(\mathbf{x}_i) - y_i| \le \epsilon + \xi_i, \\ & \xi_i \ge 0, \quad i = 1, \ldots, n. \end{aligned}$$

The constraints state that the difference between the prediction $f(x_i)$ and the target value y_i for each training point should be smaller than ε. However, to allow for outliers in the data, the "slack variable" ξ is introduced and penalized in the objective function such that deviations by more than ε linearly increase the objective function. The constants ε and C need to be chosen a priori (in our implementation, we chose them by cross-validation, but more sophisticated approaches can be used here as well [38]).

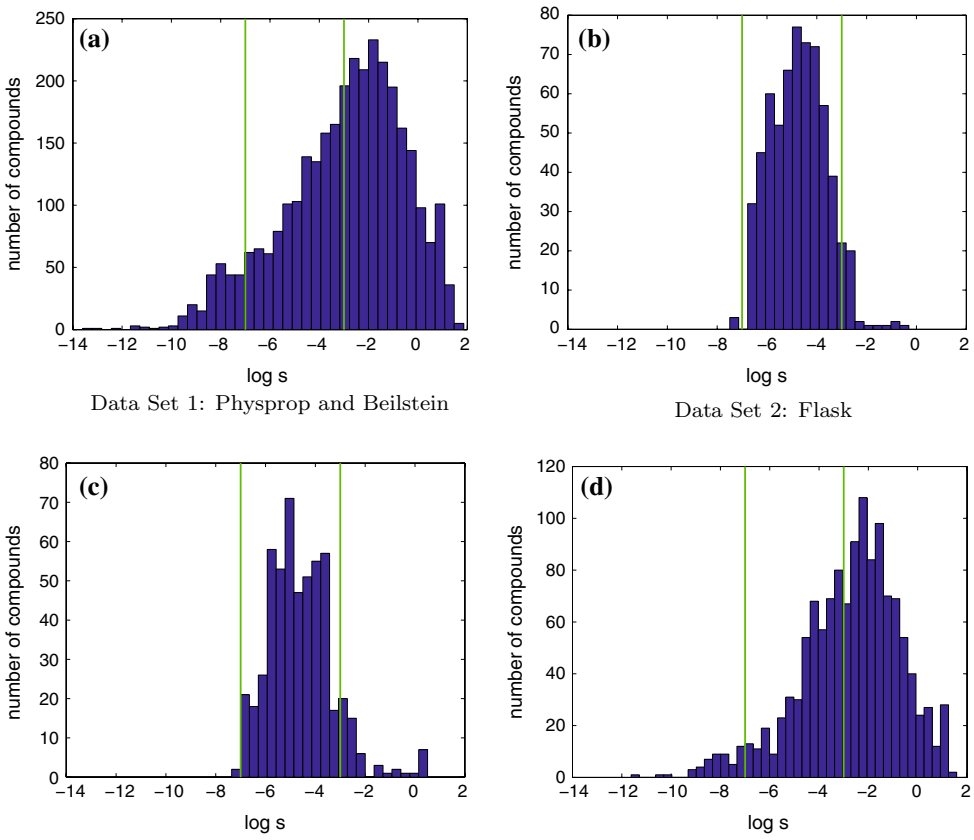
To allow for non-linear models, the so-called kernel trick [34, 37] can be used. It is possible to generate almost arbitrary nonlinear functions f of the form $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b$, by reformulating the above problem such that all occurrences of the data $\mathbf{x}$ are only in the form of scalar products $(\mathbf{x}_i^\top \mathbf{x}_j)$. The scalar products can in turn be computed by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$, that implicitly maps the descriptors into a hight-dimensional space and computes the scalar product there. This way, we can turn the linear formulation into non-linear SVR. Interestingly, the space of valid kernel functions (functions that map to a high-dimensional space and compute the scalar product) is exactly the same as the space of valid covariance functions for a GP model (functions that give a valid covariance matrix for any set of points $\mathbf{x}_1, \ldots, \mathbf{x}_M$).

Random forest and linear Ridge Regression

We chose to briefly discuss GP models and Support Vector Machines b.c. GPs are still relatively new in the context of chemoinformatics and there are interesting parallels with Support Vector Machines. For the long established methods we note how our implementation differes from the original and refer the reader to the literature for in depth descriptions.

We use the random forest method developed by Breiman [39], with the following two modifications: Firstly, no boostrapping or bagging of the sample is done to construct the trees. Secondly, single trees are pruned using a CART-style error-size tradeoff. The output variance is estimated by a simple average between the mean estimated variance from training points found at each tree leaf reached by the test point, and the variance of these leaves prediction output.

Ridge regression [40] is a technique to regularize a standard linear regression model, that is particularly important when descriptors are correlated. In a typical linear model, a wildly large positive coefficient can be canceled by a similarly large negative coefficient in its correlated counterpart. Ridge regression imposes a penalty on the sum of squares of the coefficients. Effectively, the



Data Set 1: Physprop and Beilstein

Data Set 2: Flask

Data Set 3: Flask external validation

Data Set 4: Huuskonen

Fig. 3 Histograms of S_W for all datasets used in this study. See Section. "Data preparation" for details. The vertical green lines mark the "fit-for-purpose" range [3, 5] to assess the performance of models in the S_W range relevant to drug discovery

coefficients are shrunk towards zero and towards each other in the optimization process. The amount of shrinkage is controlled by the complexity parameter λ: The greater the λ, the greater the amount of shrinkage. The idea of penalizing by the sum of squares of the parameters is closely related to regularization in SVM.

Results & discussion

Fitness for purpose

In drug discovery projects, aqueous solubility is typically between 0.1 μg/L and 250 μg/L. For a compound with a molecular weight of 500 g/mol this corresponds roughly to the S_W range from –7 to –3.5. Delaney [5] observed that a lot of models in the literature are trained on public datasets spanning more than ten orders of magnitude. Compounds with low S_W are usually harder to predict than soluble ones, nevertheless statistics are typically presented for the whole range of S_W. Delaney suggests that studies should be assessed using an element of "fit-for-purpose" (FFP). Johnson et al. [3] picked up the suggestion and evaluated a number of studies, taking into account the performance of the models in the S_W range from –7 to –3.

Figure 3 contains histograms of S_W for all four datasets used in this study. Johnson's FFP range is indicated by vertical green lines. Indeed, < 40% of the compounds in the two public sets are in the FFP range (37% of the "Huuskonen" set and 38% of the "Physprop/Beilstein" set, respectively). On the other hand more than 90% of the compounds in the two in-house data sets from Bayer Schering Pharma (compounds from drug discovery projects) are in the FFP range (94% of the "Flask" dataset and 91% of the "Flask external validation" dataset, respectively).

We present separate evaluations for the "Huuskonen", "Flask" and "Flask external validation" sets of data (see Section "Overall accuracy"). The datasets of Bayer Schering Pharma are by all means "fit-for-purpose" for drug discovery. Therefore, further separation of data with respect to the FFP range mentioned above is not necessary.

Overall accuracy

The accuracy achieved using GP models, Support Vector Machines, linear Ridge Regression and Random Forests is listed in Table 2. The line labeled "mean" lists the performance achieved when constantly predicting the average S_W of the respective dataset. Scatterplots for the two best performing models on each of the three datasets can be found in Fig. 4. Summarizing these results, we note:

- From Table 2, it is clear that the predictions of all four models are significantly better than just predicting the average solubility. This effect is of course most pronounced for the "Huuskonen" dataset, because of its large spread of experimental data, see also Fig. 3.
- Gaussian Process models (GPsol) and Support Vector Machines always take places one and two. This confirms our assumption that the relationship between descriptors and solubility is indeed non-linear.
- GPsol and SVM predictions appear to be "vertically compressed".
- Inspecting Fig. 4 we find that a lot of points representing predictions for the "Flask" and "Huuskonen" sets are very close the diagonal (i.e., very accurate). The spread is much larger in case of the "Flask external validation" setup. Also, in Table 2, we can see clearly that the performance decreases significantly when comparing "Flask external" with the cross-validation results on "Flask".

Table 2 Accuracy achieved using Gaussian Process models, Support Vector Machines, linear Ridge Regression and Random Forests for the respective datasets. "Mean" denotes the results achieved when always predicting the arithmetic mean S_W of the respective dataset. MAE, RMSE and % ±1 denote the mean absolute error, the root mean squared error, and the percentage of compounds predicted correctly within one log unit.

	MAE	RMSE	%±1
Flask cross.			
GPsol	0.573	0.747	84.2
RR	0.667	0.862	77.8
SVM	0.617	0.803	81.3
Forest	0.650	0.840	78.7
Mean	0.908	1.117	60.2
Flask ext.			
GPsol	0.656	0.846	78.5
RR	0.657	0.847	78.4
SVM	0.657	0.848	78.5
Forest	0.663	0.855	78.0
Mean	1.008	1.305	59.0
Huuskonen			
GPsol	0.412	0.579	91.2
RR	0.586	0.996	86.1
SVM	0.431	0.600	90.2
Forest	0.485	0.660	88.3
Mean	1.593	2.04	38.8

The decrease in performance observed when looking at external validation data could be taken as a hint that the more complex models did over-fit their training data. We did, however, not observe typical symptoms of over-fitting, e.g. a too large number of support vectors in a SVR model. Figure 5 shows histograms of Mahalanobis distances from each compound to the closest compound in the respective training set. Distances were calculated based on the same

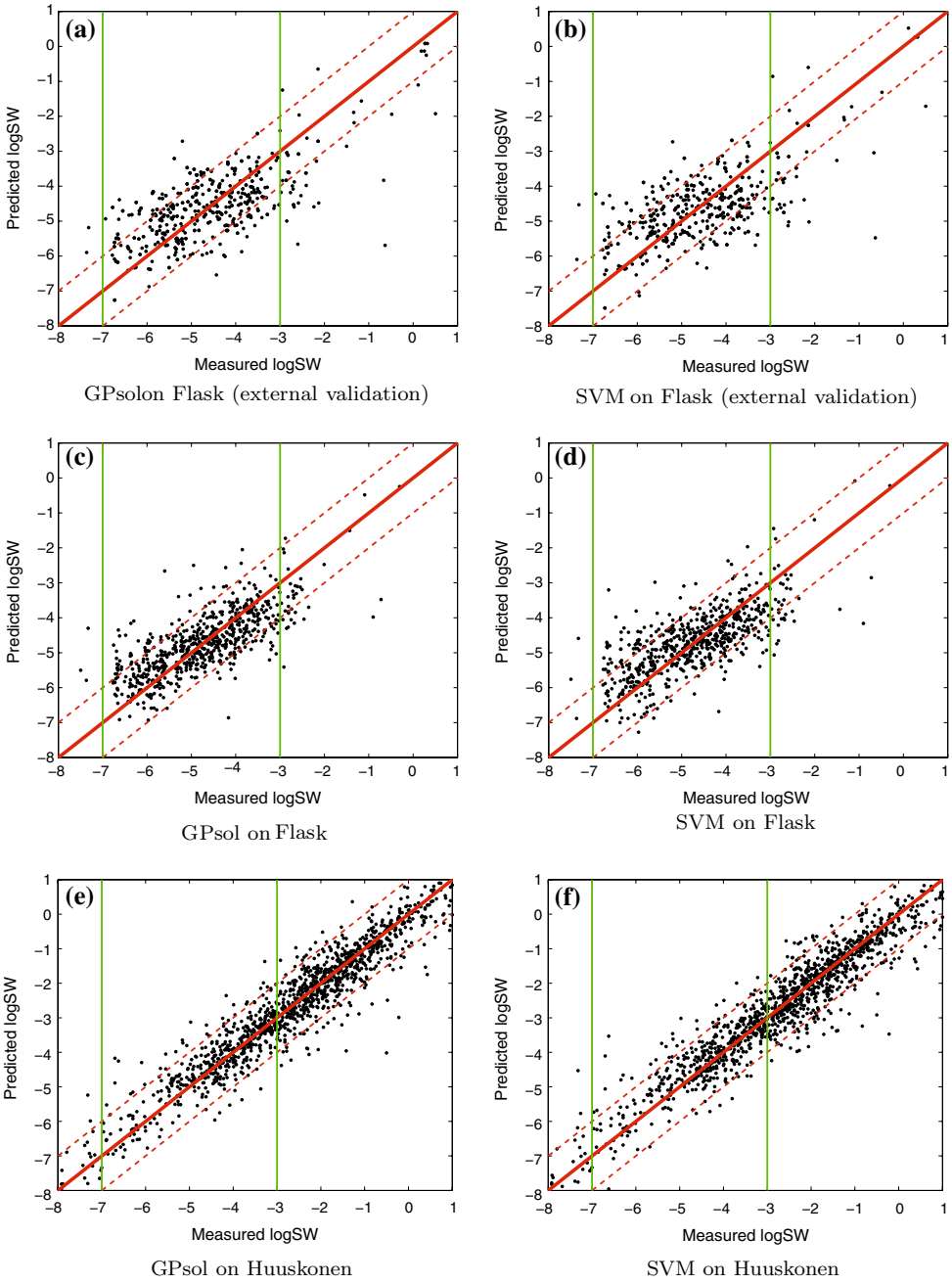


Fig. 4 Scatterplots for the two best performing models on each of the three datasets. The vertical green lines mark the "fit-for-purpose" range [3, 5] to assess the performance of models in the S_W range relevant to drug discovery

set of descriptors that was used to build the models. In the investigated training/validation-split of the "Flask" validation setup, 97% of the compounds have at least one neighbor at a distance smaller than 1500 units. In the "Flask external validation" setup, only 48% of the compound have neighbors closer than 1500 unites. This clearly confirms that "Flask external" is a set of compounds that is, to a large extent, structurally dissimilar to the training data in "Flask". Thus, we can assume that the decrease in performance is caused by a large number of compounds being dissimilar to the training set compounds. These compounds are thus outside of the models' respective domains of applicability. If this assumption holds, it should be possible to achieve higher performance by rejecting compounds that are outside the DOA. We address this question in Section "Improving accuracy by focussing on domain of applicability".

By its construction, predictions from the GP model get closer to the mean log S_W when new compounds are more dissimilar to those in the training set. At the same time, the size of the predicted error bars increases. In Fig. 6 we present a scatterplot for GPsol on the "Flask external" validation set. Black points represent the confident predictions, whereas grey points represent the less confident

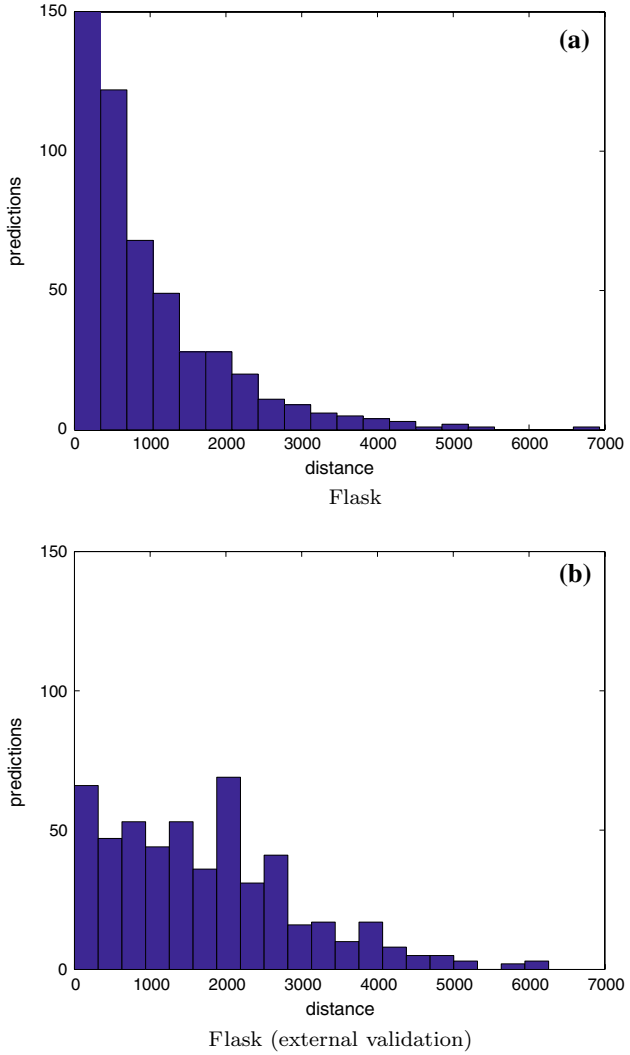


Fig. 5 Histograms of Mahalanobis distances from each compound to the closest compound in the respective training set. Distances for the cross validated "Flask" setup were calculated for the training/validation-split of one arbitrarily chosen cross validation run

predictions with predicted error bars larger than 0.6 log units. Vertical compression can be observed in the cloud of grey points, but is not present in the cloud of black points. Thus, we conclude that the vertical compression observed in Fig. 4 is indeed caused by compounds that are dissimilar to the training set.

Improving accuracy by focussing on domain of applicability

In the "Flask" setup, most compounds have close near neighbors in the respective training set. In contrast, in the "Flask external validation" setup, a lot of compounds do not have close next neighbors and may be outside of the DOA of the respective model (see Section "Overall accuracy" for details). The scatterplot of high and low

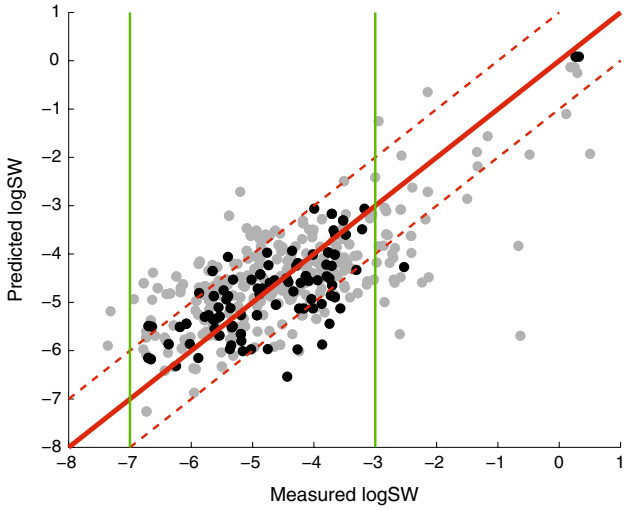


Fig. 6 Scatterplot for GPsol on the "Flask external" validation set. Black points represent confident predictions, grey points represent less confident predictions with predicted error bars larger than 0.6

confidence predictions in Fig. 6 suggests that performance statistics can be improved by taking the predicted error bars into account. In the following section, we will investigate whether the prediction accuracy can be increased by rejecting compounds outside of the DOA and quantify how much accuracy can be gained.

The different model classes offer different ways of estimating the DOA:

- Gaussian Process models are inherently probabilistic, and offer a direct estimate of predictive variance.
- For random forests, the DOA can be estimated by the spread of predictions in the committee of decision trees.
- Ridge Regression and Support Vector Machine do not provide error bars at all. We estimate it by considering the Mahalanobis distance to the closest point in the training set.

We proceed by assigning bins for the model based error bar (GP and random forest) respectively distance (for ridge regression and SVM). Within each bin (representing the predicted error), we compute the mean absolute error of all predictions that fall into this bin (the actual error made). The resulting plots are shown in Fig. 7. For all four models, there is a clear trend towards small prediction errors for the bins with small model based error bars (or distances), and, vice-versa, large errors when the error bars (distances) are large.

Comparing the mean absolute errors from the "Flask external validation" setup presented in Table 3 and 2 we find that the prediction accuracy indeed increases when focussing on predictions with small model based error bars (or distances). This effect is most pronounced for the two best performing models, Gaussian Processes and SVM: In

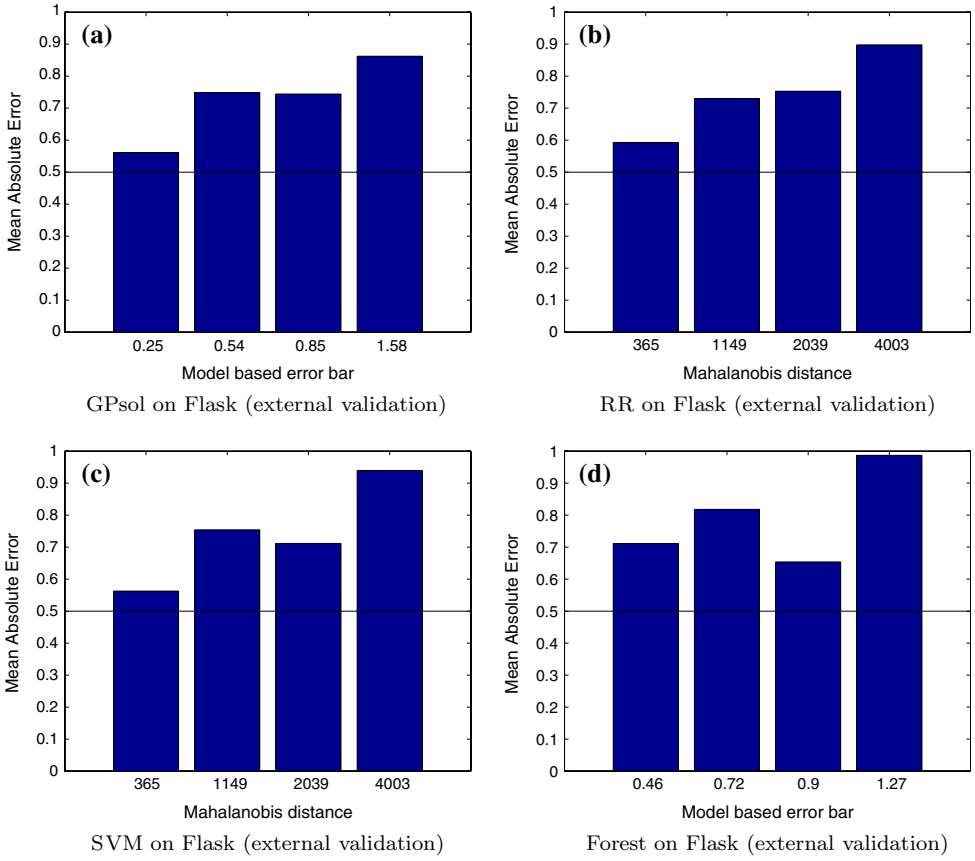


Fig. 7 Mean absolute error achieved when binning by the model based error bar (in he case of GPsol and the random forest) and the Mahalanobis distance to the closest point in the training set (linear Ridge Regression and Support Vector Machines do not provide error bars). Each column represents one fourth (133 compounds) of the "Flask external" validation set. Corresponding numbers can be found in Table 3

both cases, the mean absolute error decreases from 0.66 to 0.56.

In the "Flask" cross-validation setup, only the best performing model, the GP, had a mean absolute error as low as 0.57, with the SVM being second with a MAE of 0.61. By focussing on the most reliable predictions, i.e., the leftmost bar/column in Fig. 7 and Table 3, respectively, both models achieve as good results in the "Flask external validation" setup as was previously estimated using the "Flask" cross validation setup.

We conclude, that the decrease in performance observed when moving from the "Flask" cross-validation setup to the "Flask external validation" setup results from the fact that the "Flask external validation" dataset contains a lot of compounds that are dissimilar to the compounds in the "Flask" dataset used for training. Using model based error bars (or distances) to identify the predictions that are clearly inside the DOA increases the accuracy to the level previously estimated using cross-validation.

Table 3 Mean absolute error achieved when binning by the model based error bar (in the case of GP and random forest models) resp. the Mahalanobis distance to the closest point in the training set (for linear Ridge Regression and Support Vector Machines)

Bin number	1	2	3	4
error bar GPsol (average within bin)	0.25	0.54	0.85	1.58
MAE GPsol	0.56	0.75	0.74	0.86
error bar forest (average within bin)	0.46	0.72	0.90	1.27
MAE forest	0.71	0.82	0.65	0.99
distance (average within bin)	365	1149	2039	4003
MAE RR	0.59	0.73	0.75	0.90
MAE SVM	0.56	0.75	0.71	0.94

Bins were chosen such that each bin contains one fourth (133 compounds) of the "Flask external" validation set. A graphical representation of this information can be found in Fig. 7

Individual error estimation for interactive use

To the best of our knowledge, the usefulness of individual confidence estimates for interactive use in ADME/Tox and physical property prediction has so far not been discussed or investigated in scientific studies outside of our own group [1, 29, 30]. The typical approach is to present error estimates for predictions binned by distance, i.e., averaging over a large number of predictions. In practical applications, the user thus always has to bear in mind the actual relationships between predicted and expected error, and know that, for example, a distance around 1100 represents an average error of 0.7 (see Table 3).

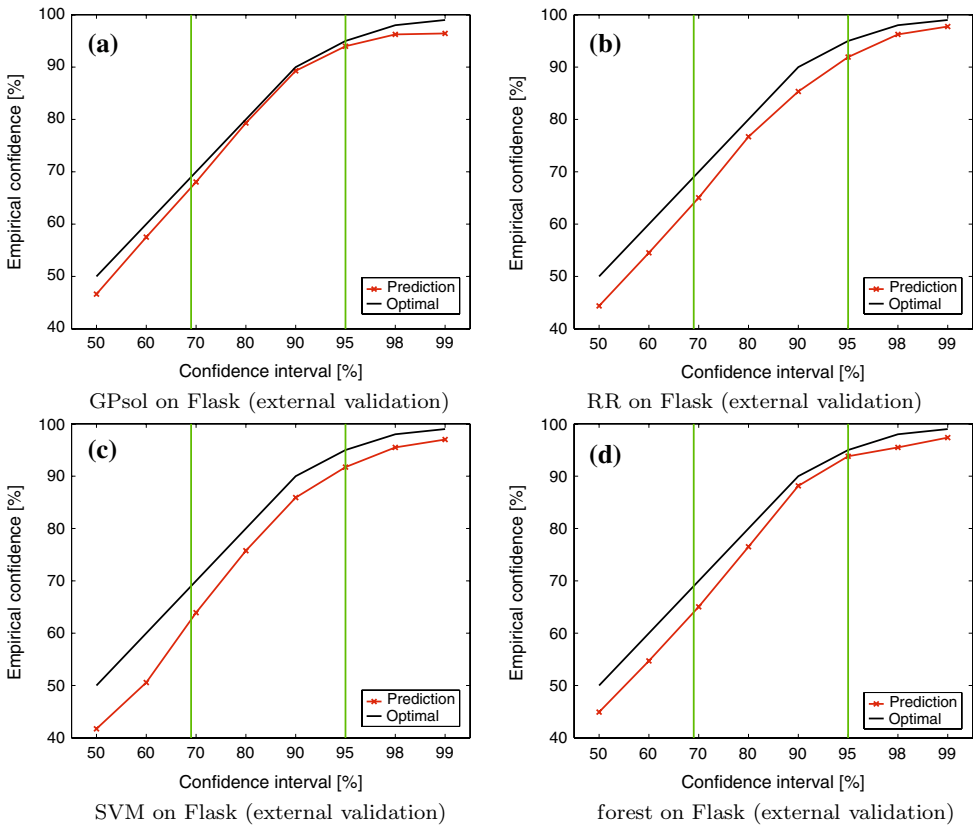


Fig. 8 Predicted error bars can be evaluated by counting how many predictions are actually within a σ, 2σ etc. environment (red line) and comparing with the optimal percentage (black line). The vertical green lines indicate the σ and 2 σ environments, the corresponding numbers can be found in Table 4

A more intuitive approach is to relate the predicted error to a probability distribution. The most commonly used definition of uncertainty (for example, to model measurement errors, prediction errors, etc.) in chemistry, physics and other fields is based on the assumption that errors follow a Gaussian distribution. Assuming that the model prediction is a Gaussian (that is, the prediction $\bar{f}$ describes the mean, and the error bar describes the standard deviation σ), it follows that the true value is in the interval $\bar{f} \pm \sigma$ with 68% confidence, and in the interval $\bar{f} \pm 2\sigma$ with 95% confidence, etc.

The quality of the predicted error bars can therefore be evaluated by comparing with the true experimental values, and simply counting how many of the experimental values are actually within the σ, 2 σ, etc. intervals[2].

Support vector machine and linear Ridge Regression do not readily provide error bars. For these two models, error bars were estimated by fitting both linear and exponential [8] functions to the distances and prediction errors observed in cross-validation on the training data. Both linear and exponential functions worked equally well, we thus chose the more simple one and used linear regression models to convert distances to error bars.

Figure 8 shows that the predicted error bars exhibit reasonable behavior for all four models investigated, with the GPsol error predictions being closest to the ideal distribution. The vertical green lines highlight the σ and 2 σ environments. Numerical values for these two points are given in Table 4. The numbers confirm that all models produce error predictions with the correct statistical properties, with the GPsol error predictions being closest to the optimal behavior.

Both the log S_W predictions and the predicted error bars from the GPsol model were initially validated in a "blind test" [1], where the modelling team at Fraunhofer and idalab did not have access to the experimental values for the "Flask external" validation set. This set contains data from recent drug discovery projects, which differ significantly from the compounds included in the training set (see Section "Overall accuracy"). Therefore we conclude that the correct behavior of the predicted error bars is not limited to the training set used. The approaches described allow identifying compounds outside of the DOA and even quantify the reliability of a prediction in a straightforward way. Error bars are generally known by chemists and other professionals and can therefore be used directly to give an intuition of the reliability of measurements.

[2] It has been suggested to use numeric criteria, such as log probability of the predictive distribution, for this purpose. Our experience suggests that these criteria can be misleading, they thus have not been used. In particular, log probability tends to favor over-optimistic models.

Table 4 Predicted error bars can be evaluated by counting how many predictions are actually within a σ, 2 σ etc. environment and comparing with the optimal percentage. A graphical representation of these results including fractions of σ can be found in Fig. 8

Environment Optimal %:	Pred ± σ 68.7	Pred ± 2σ 95.0
GPsol	65.0	94.0
RR	62.8	91.9
SVM	62.0	91.7
Forest	62.2	93.8

Summary

Using four different Machine Learning methods, we constructed models for aqueous solubility based on about 4000 compounds, including an in-house set of 632 drug discovery molecules of Bayer Schering Pharma. Error bars were estimated using a Bayesian modelling approach (GP model), an ensemble based approach (Random forest) and distance based approaches (Mahalanobis distance to training data for SVM and Ridge Regression models).

We conclude that all four methods investigated predict error bars that can be used to reject predictions for compounds outside of the DOA of the respective model. In this setting, prediction accuracy on the external validation set (536 measurements of drug discovery molecules investigated during the last year, including compounds from new projects) is as high as was estimated from the training data using cross-validation. Furthermore, we find that individual error bars from all four methods are suitable for interactive application of the model. Considering the statistical properties of the actual errors made on the external validation set, predicted error bars form the GP model are closest to the ideal behavior.

The goal of future research is to continuously improve modeling for computational chemistry using Machine Learning methods.

Acknowledgements The authors gratefully acknowledge partial support from the PASCAL Network of Excellence (EU #506778) and DFG grant MU 987/4-1. We thank Vincent Schütz and Carsten Jahn for maintaining the PCADMET database, and Gilles Blanchard for implementing the random forest method as part of our machine learning toolbox.

References

1. Schwaighofer A, Schroeter T, Mika S, Laub J, ter Laak A, Sülzle D, Ganzer U, Heinrich N, Müller K-R (2007) J Chem Inf Model 47:407 URL http://dx.doi.org/10.1021/ci600205
2. Balakin KV, Savchuk NP, Tetko IV (2006) Curr Med Chem 13:223
3. Johnson SR, Zheng W (2006) The AAPS J 8:E27 URL http://www.aapsj.org/articles/aapsj0801/aapsj080104/aapsj080104.pdf
4. Göller AH, Matthias H, Jörg K, Timothy C (2006) J Chem Inf Model 46:648
5. Delaney JS (2005) Drug Discovery Today 10:289
6. Goldman BB, Walters WP (2006) Machine learning in computational chemistry, vol 2, chapter 8, Elsevier, pp 127
7. Netzeva TI, Worth AP, Aldenberg T, Benigni R, Cronin MTD, Gramatica P, Jaworska JS, Kahn S, Klopman G, Marchant CA, Myatt G, Nikolova-Jeliazkova N, Patlewicz GY, Perkins R, Roberts DW, Schultz TW, Stanton DT, van de Sandt JJM, Tong W, Veith G, Yang C (2005) Altern Lab Anim 33:1
8. Tetko IV, Bruneau P, Mewes H-W, Rohrer DC, Poda GI (2006) Drug Discovery Today 11:700
9. Tropsha A (2006) Variable selection qsar modeling, model validation, and virtual screening. In: Spellmeyer DC (ed) Annual reports in computational chemistry, vol 2, chapter 7, Elsevier, pp 113
10. Bruneau P, McElroy NR (2004) J Chem Inf Model 44:1912
11. Tong W, Xie Q, Hong U, Shi L, Fang H, Perkins R (2004) Environ Health Perspect 112:1249
12. Bruneau P, McElroy NR (2006) J Chem Inf Model 46:1379
13. Silverman BW (1986) Density estimation for statistics and data analysis. Number 26 in Monographs on Statistics and Applied Probability. Chapman & Hall
14. Manallack DT, Tehan BG, Gancia E, Hudson BD, Ford MG, Livingstone DJ, Whitley DC, Pitt WR (2003) J Chem Inf Model 43:674
15. Kühne R, Ebert R-U, Schüürmann G (2006) J Chem Inf Model 46:636
16. Bender A, Mussa HY, Glen RC (2005) J Biomol Screen 10:658 http://jbx.sagepub.com/cgi/content/abstract/10/7/658
17. Sun H (2006) Chem Med Chem 1:315
18. Sadowski J, Schwab C, Gasteiger J Corina v3.1. Erlangen, Germany
19. Todeschini R, Consonni V, Mauri A, Pavan M DRAGON v1.2. Milano, Italy
20. Physical/Chemical Property Database (PHYSPROP). Syracuse, NY, USA
21. Beilstein CrossFire Database. San Ramon, CA, USA
22. Yalkowsky SH, Dannelfelser RM The arizona database of aqueous solubility. Tuscon, AZ, USA
23. Huuskonen J (2000) J Chem Inf Comput Sci 40:773
24. Ran Y, Jain N, Yalkowsky SH (2001) J Chem Inf Comput Sci 41:1208
25. Tetko IV, Tanchuk VY, Kasheva TN, Villa AEP (2001) J Chem Inf Comput Sci 41:1488
26. Yan A, Gasteiger J (2003) QSAR Comb Sci 22:821
27. Livingstone DJ, Martyn F, Huuskonenc JJ, Salt DW (2001) J Comput-Aided Mol Des 15:741
28. Todeschini R, Consonni V, Mauri A, Pavan M, Dragon for windows and linux 2006. URL http://www.talete.mi.it/help/dragon_help/ (accessed 14 May 2006)
29. Schroeter T, Schwaighofer A, Mika S, Ter Laak A, Suelzle D, Ganzer U, Heinrich N, Muller K-R (2007) Chem Med Chem http://dx.doi.org/10.1002/cmdc.200700041
30. Schroeter T, Schwaighofer A, Mika S, Ter Laak A, Suelzle D, Ganzer U, Heinrich N, Müller K-R (2007) URL http://dx.doi.org/10.1021/mp0700413
31. O'Hagan A (1978) J R Stat Soc Ser B: Methodological 40:1
32. Rasmussen CE, Williams CKI (2005) Gaussian Processes for machine learning. MIT Press
33. Schölkopf B, Smola AJ (2002) Learning with kernels. MIT Press
34. Müller K-R, Mika S, Rätsch G, Tsuda K, Schölkopf B (2001) IEEE Trans Neural Netw 12:181
35. Vapnik VN (1998) Statistical learning theory. Wiley, New York
36. Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines. Cambridge University Press, Cambridge, UK

37. Schölkopf B, Smola AJ (2002) Learning with kernels. MIT Press, Cambridge MA
38. Wang G, Yeung D-Y, Lochovsky FH (2006) Two-dimensional solution path for support vector regression. In: De Raedt L, Wrobel S (eds) Proceedings of ICML06, ACM Press, pp 993 URL http://www.icml2006.org/icml_documents/camera-ready/125_Two_Dimensional_Solu.pdf
39. Breiman L (2001) Mach Learn 45:5 URL http://dx.doi.org/10.1023/A:1010933404324
40. Hastie T, Tibshirani R, Friedman J (2001) The elements of statistical learning: data mining, inference and prediction. Springer series in statistics. Springer, New York, NY