Machine Learning Models for Lipophilicity and their Domain of Applicability

The original publication is available at pubs.acs.org
http://dx.doi.org/10.1021/mp0700413

Timon Schroeter[‡†], Anton Schwaighofer[†], Sebastian Mika[¶], Antonius Ter Laak[‖], Detlev Suelzle[‖], Ursula Ganzer[‖], Nikolaus Heinrich[‖], Klaus-Robert Müller[‡†]

[†] Fraunhofer FIRST, Kekuléstraße 7, 12489 Berlin, Germany
[‡] Technische Universität Berlin, Department of Computer Science, Franklinstraße 28/29, 10587 Berlin, Germany
[¶] idalab GmbH, Sophienstraße 24, 10178 Berlin, Germany
[‖] Research Laboratories of Bayer Schering Pharma AG, Müllerstraße 178, 13342 Berlin, Germany

July 21, 2007

Abstract

Unfavorable lipophilicity and water solubility cause many drug failures, therefore these properties have to be taken into account early on in lead discovery. Commercial tools for predicting lipophilicity usually have been trained on small and neutral molecules, and are thus often unable to accurately predict in-house data.

Using a modern Bayesian machine learning algorithm—a Gaussian Process model—this study constructs a $\log D_7$ model based on 14556 drug discovery compounds of Bayer Schering Pharma. Performance is compared with support vector machines, decision trees, ridge regression and four commercial tools. In a blind test on 7013 new measurements from the last months (including compounds from new projects) 81 % were predicted correctly within one log unit, compared to only 44 % achieved by commercial software. Additional evaluations using public data are presented.

We consider error bars for each method (model based error bars, ensemble based, and distance based approaches), and investigate how well they quantify the domain of applicability of each model.

1 Introduction

Lipophilicity of drugs is a major factor in both pharmacokinetics and pharmacodynamics. Since a large fraction of drug failures ($\sim 50\%$) [1] results from an unfavorable PC-ADME/T profile (absorption, distribution, metabolism, excretion, toxicity), the octanol water partition coefficients $\log P$ and $\log D$ are nowadays considered early on in lead discovery.

Due to the confidentiality of in-house data, makers of predictive tools are usually not able to incorporate such data from pharmaceutical companies. Commercial predictive tools are therefore typically constructed using publicly available measurements of relatively small and mostly neutral molecules. Often, their accuracy on the in-house compounds of pharmaceutical companies is relatively low [2].

In our work, we follow a different route to derive models for lipophilicity that are tailored to in-house data. We use a modern machine learning tool, a so-called Gaussian Process model [3] (short: GP), to obtain a nonlinear mapping from descriptors to lipophilicity. A specific advantage of the tool is its Bayesian framework for model selection, that provides theoretically well founded criteria to automatically choose the "right amount of nonlinearity" for modeling. We can avoid extensive grid search in cross-validation or expert intervention to choose optimal parameter settings. Thus, the process can be fully automated. For the first data analysis phase, structures do not need to be disclosed, since all modeling is descriptor based.

Apart from the high performance, the chosen modeling approach shows another virtue that makes it an excellent tool for applications in chemistry: GP models have their roots in Bayesian statistics, and thus can supply the user with an error bar for each individual prediction. This quantification of the prediction uncertainty allows to reduce the error rate, by discarding predictions with large error bars, or by re-confirming the prediction with a laboratory experiment. In our work, we also compare these error bars with error bar heuristics that can be applied to other commonly used modeling approaches.

Performance is compared with models constructed using three established machine learning algorithms: Support Vector Machines, Random Forests and linear Ridge Regression. We show that the different $\log \mathrm{P}$ and $\log \mathrm{D}_7$ models exhibit convincing prediction performance, both on benchmark data and in-house data of drug molecules. We compare our results with several commercial tools, and show a large improvement of performance, in particular on the in-house classes of compounds.

Using machine learning algorithms, one can construct models of biological and chemical properties of molecules from a limited set of measurements [4;5;6;7;8]. This so called training set is used to infer the underlying statistical properties and select a prediction model. Tuning of (hyper-)parameters is usually performed using cross-validation or re-sampling methods. To evaluate the performance of the model, one should use a set of data that was not used in model building in any form. In the best case, the model is evaluated in a *blind test*, where the modelers do not have access to the held out data. Instead, the final model is applied to the blind test data by an independent evaluating team. In normal benchmark evaluations, re-tuning models on held-out data is possible and typically results in too optimistic results. In contrast, the blind-test strategy is nearly unbiased, because "cheating", i.e., using results on the held-out data for re-tuning the model, becomes infeasible. Note however that the blind test set of data needs to be of somewhat reasonable size, and should represent the typical application scenario of the model that is to be evaluated.

GP models have been previously[1] applied in computational chemistry, but rather small data sets were used, and typically no blind test was conducted:

- Burden[9] learned the toxicity of compounds and their activity on muscarinic and benzodiazepine receptors using up to 277 compounds.
- Enot et al.[10] predicted log P using 44 compounds from a *1,2-dithiole-3-one* series.
- Tino et al.[11] built GP models for log P from a public data set of 6912 compounds. Here, a blind test was conducted, but the validation set (provided by Pfizer) contained only 266 compounds.

This study goes beyond the prior work: Our models were trained on large sets of public and in-house data (up to 14556 compounds). A blind test was performed by an independent evaluating team at Bayer Schering Pharma using a set of 7013 drug discovery molecules from recent projects, that have not been available to the modeling team Fraunhofer FIRST and idalab. To facilitate reproduction of our results by other researchers, the complete list of compounds in the public data set is included in the supporting information to our initial communication[4].

2 Estimating the Domain of Applicability of Models

A typical challenge for statistical models in the chemical space is to adequately determine the domain of applicability, i.e. the part of the chemical space where the model's predictions are reliable. To this end several "classical" approaches exists: *Range based methods* are based on checking whether descriptors of test set compounds exceed the range of the respective descriptor covered in training[12;13]. A warning message is raised when this occurs. Also, *geometric methods* that estimate the convex hull of the training data can be used to further detail such estimates[14]. Mind that both these methods are not able to detect "holes" in the training data, that is, regions that are only scarcely populated with data.[2]

If experimental data for some new compounds are available, error estimates based on the *library approach*[15] can be used. By considering the closest neighbors in the library of new compounds with known measurements, it is possible to get a rough estimate of the bias for the respective test compound.

Probability density distribution based methods could, theoretically, be used to estimate the model reliability[14]. Still, high dimensional density estimation is recognized as an extremely difficult task, in particular since the behavior of densities in high dimensions may be completely counterintuitive[16].

[1] Our own recent results of modeling aqueous solubility are presented in[5;6].

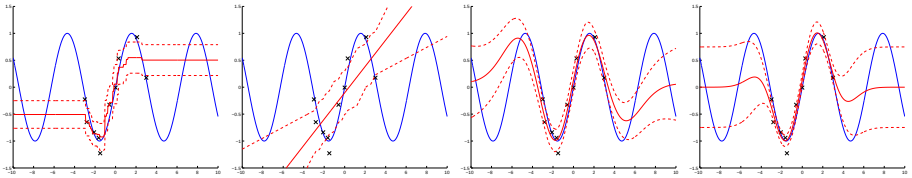
[2] Holes in the training data can, in principle, be detected using geometric methods in a suitable feature space. To the best of our knowledge, there exists no published study about this kind of approach.

Distance based methods and *extrapolation measures*[17;14;18;2;12] consider one of a number of distance measures (Mahalanobis, Euclidean etc.) to calculate the distance of a test compound to its closest neighbor(s) or the whole training set, respectively. Another way of using distance measures is to define a threshold and count the number of training compounds closer than the threshold. Hotellings test or the leverage rely on the assumption that the data follows a Gaussian distribution in descriptor space and compute the Mahalanobis distance to the whole training set. Tetko correctly states in[18] that descriptors have different relevance for predicting a specific property and concludes, that property specific distances (resp. similarities) should be used [3].

When estimating the domain of applicability with *ensemble methods*, a number of models is trained on different sets of data. Typically the sets are generated by (re)sampling from a larger set of available training data. Therefore the models will tend to agree in regions of the descriptor space where a lot of training compounds are available and will disagree in sparsely populated regions. Alternatively, the training sets for the individual models may be generated by adding noise to the descriptors, such that each model operates on 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 small descriptor changes is large. This methodology can be used with any type of models, but ensembles of ANNs[18;2;19;20;17] and ensembles of decision trees[13;17] ("random forests", Breiman[21]) are most commonly used.

The idea behind *Bayesian methods* is to treat all quantities involved in modeling as random variables. By means of Bayesian inference, the *a priori* assumptions about parameters are combined with the experimental data, to obtain the *a posteriori* knowledge. Hence, such models naturally output a probability distribution, instead of the "point prediction" in conventional learning methods. Regions of high predictive variance not only indicate compounds outside the domain of applicability, but also regions of contradictory or scarce measurements. The most simple and also most widely used method is the naive Bayes classifier[22;23]. Gaussian Process regression and classification are more sophisticated Bayesian methods, see Sec. 3.5.4.

In the present study, we use the Bayesian Gaussian Process models, ensembles and distance based methods. All of these can handle empty regions in descriptor-space and quantify their confidence, rather than just marking some predictions as possibly unreliable. Confidence estimates will be presented in a form that is intuitively understandable to chemists and other scientists.



(a) Random forest (ensemble) (b) Ridge Regression (distance based) (c) S.V. Machine (distance based) (d) Gaussian Process (Bayesian)

Figure 1: The four different regression models employed in this study are trained on a small number of noisy measurements (black crosses) of the sine function (blue line). Predictions from each model are drawn as solid red lines, while dashed red lines indicate errors estimated by the respective model (in case of the Gaussian Process and random forest) or a distance based approach (in case of the Support Vector Machine and Ridge Regression model).

2.1 One Dimensional Examples

Figure 1 shows a simple one-dimensional example of the four different methods of error estimation we use in this study. The sine function (shown as a blue line in each subplot) is to be learned. The available training data are ten points marked by black crosses. These are generated by randomly choosing x-values and evaluating the sine function at these points. We simulate measurement noise by adding Gaussian distributed random numbers with standard deviation 0.2 to the y-values.

The random forest, Figure 1 (a), does provide a reasonable fit to the training points (yet the prediction is not smooth, due to the space dividing property of the decision trees). Predicted errors are acceptable in the vicinity of the training points, but overconfident when predictions far from the training points are sought. It should be noted that the behavior of error bars in regions outside of the training data depends solely on the ensemble members on the boundary of the training data. If the ensemble members, by chance, agree in their prediction, an error bar of zero would be the result.

The linear model, Figure 1 (b), clearly cannot fit the points from the non-linear function. Therefore, the distance based error estimations are misleading: Low errors are predicted in regions close to the training points, but the actual error is quite large due to the poorly fitting model. This shows that the process of error estimation should not be decoupled from the actual model fitting: The error estimate should also indicate regions of poor fit.

The Support Vector Machine, Figure 1 (c), adapts to the non-linearity in the input data and extrapolates well. The error estimation (the same distance based procedure as described for the real data, Sec. 4.4) produces slightly conservative

[3]There is an interesting parrallel to Gaussian Process models: When allowing GP models to assign weights to each descriptor that enters the model as input, they implicitly construct a property specific distance measure and use it both for making predictions and for estimating prediction errors.

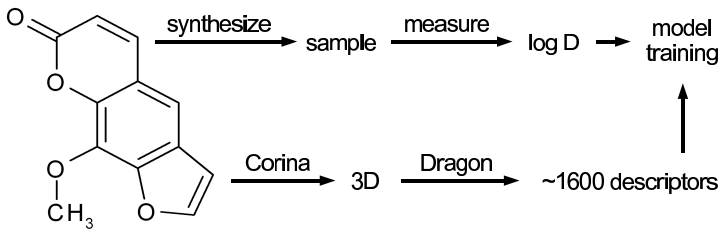


Figure 2: The process of model building.

(large) error bars in the region close the training points, and too small errors when extrapolating.

The Gaussian Process, Figure 1 (d) also captures the non-linearity in the input data and is able to extrapolate. Predicted errors are small in the region close to the training points and increase strong enough in the extrapolation region.

3 Methods and Data

3.1 Methodology Overview

The training procedure is outlined in Figure 2. We use Corina[24] to generate a 3D structure for each molecule. Molecular descriptors are calculated using the Dragon[25] software. Finally, a number of machine learning algorithms is used to "train" models, i.e., to infer the relationship between the descriptors and the experimental values for $\log \mathrm{P}$ and $\log \mathrm{D}_7$.

To make predictions for new compounds, structures are again converted to 3D and descriptors are calculated. From the descriptors of each molecule, the model generates a prediction of $\log \mathrm{P}$ and/or $\log \mathrm{D}_7$, and in case of the Gaussian Process and random forest also a confidence estimate (error bar).

3.2 Data Preparation

3.2.1 Multiple Measurements

If multiple measurements exist for the same compound, we merge them as described in the following to obtain a consensus value for model building. For each compound we generate the histogram of experimental values. Characteristic properties of histograms are the spread of values (y-spread) and the spread of the bin heights (z-spread). If all measured values are similar (small y-spread) the median value is taken as consensus value. If a group of similar measurements and smaller number of far apart measurements exists, both y-spread and z-spread are large. In this case we treat the far apart measurements as outliers, i.e., we remove them and then use the median of the agreeing measurements as consensus value. If an equal number of measurements supports on of two (or

more) far apart values (high y-spread and zero z-spread) we discard the compound. Initial experiments suggested that 0.5 (on the measurements log-scale) is a suitable value for the threshold between small and large y-spreads.

3.2.2 Dataset 1: in-house

Dataset 1 consists of 14556 drug discovery compounds of Bayer Schering Pharma. log D was measured following the experimental procedure described in Sec. A. For the majority of compounds, log D was measured at $pH = 7.0$. For about 600 compounds log D was measured at $pH = 7.4$. Although for particular compounds with pKa-values close to $pH = 7$ one can expect deviations in log D of up to 0.4 (extreme case), first experiments showed that building separate models is not necessary. No negative impact on the model accuracy was observed when the measurements performed at $pH = 7.4$ are included in the larger set.

3.2.3 Dataset 2: in-house validation

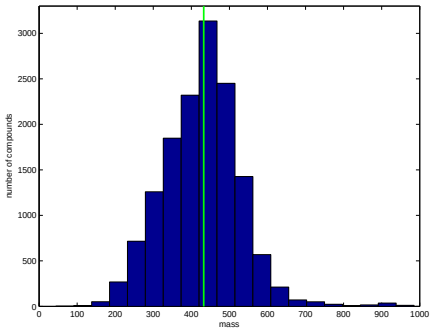
Dataset 2 is a set of 7013 new measurements of drug discovery molecules of Bayer Schering Pharma that were collected in the months after dataset 1 had been measured, and thus also includes compounds from new projects. log D was measured following the same experimental procedure as was used for dataset 1, see Sec. A.

3.2.4 Dataset 3: public

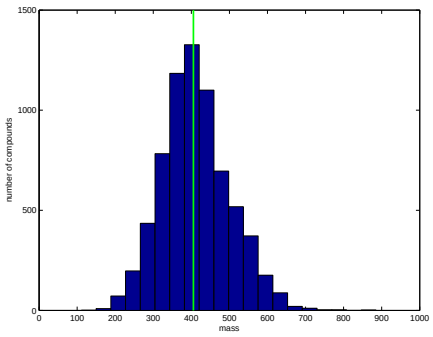
This set contains measurements of log P for 7926 unique compounds extracted from the Physprop[26] and Beilstein[27] databases. log D measurements performed at various pH values are often reported as log P in the literature, despite the fact that log P applies, by definition, only to a molecule in its neutral form (i.e. the pH of the solution has to be adjusted so that the molecule is neutral). To avoid these wrongly reported log P values, the set was restricted to compounds predicted to be completely neutral at pH 2 to 11 by ACDLabs v9, since for these compounds, log D values in the given pH ranges coincide with the correct log P values.

3.2.5 Differences between in-house and public data

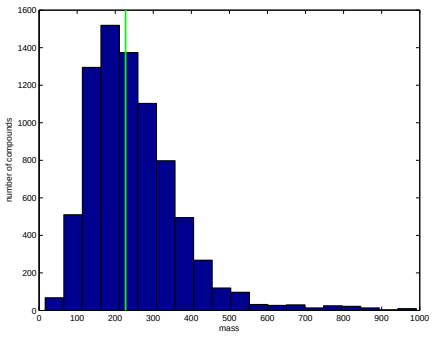
Histograms of the molecular weight for each dataset are given in Figure 3. The median of the molecular weight is 227 g/mol for the public dataset, 432 g/mol for the in-house set and 405 g/mol for the in-house validation set (marked by vertical green lines in the plots). As we can see from the histogram, more than 90% of the compounds in the public set have a molecular mass lower than 400 g/mol, that is well below the median of the molecular mass for the two in-house sets of data. In this study, we separately evaluate models on the public and in-house sets of data. In principle, data from internal and external sources can be combined. However, care has to be taken when evaluating models on mixed sets, since such models typically perform well on compounds with low molecular



(a) Data Set 1: in-house



(b) Data Set 2: in-house validation



(c) Data Set 3: public

Figure 3: Histograms of molecular weight. Vertical green lines mark the median of the molecular weight of the respective data set.

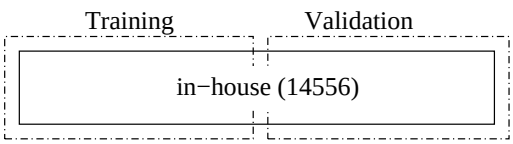
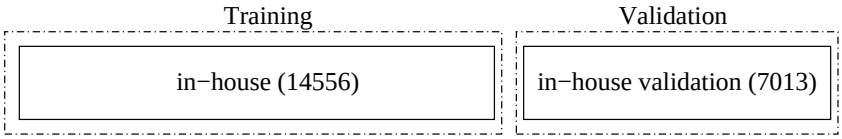
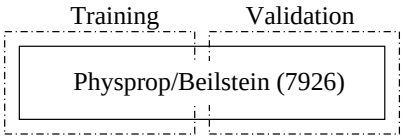
Setup	Prediction	Data
In-house	log D	
In-house validation	log D	
Public	log P	

Table 1: Summary of the different setups that are used for performance evaluation. See Sec. 3.3 for a description and Sec. 3.2 for details on the individual data sets

weight (see Sec. 4.2) but are less accurate for the larger compounds relevant to drug discovery (see Sec. 4.3).

3.3 Training and Validation Setups

3.3.1 Cross-Validation

On the *in-house* and *public* set of data, models are evaluated in leave 50% out cross-validation, i.e. the data is randomly split into two halves. A model is trained on the first half and evaluated on the other half. This is repeated with the two halves of the validation set exchanged, so that predictions for all compounds in the set are generated. The overall procedure is then repeated 10 times with a different random split. Each prediction is an out-of-sample prediction, made by a model that has not seen the particular compound in its training data.

3.3.2 Blind test

Gaussian Process models built by the modelers at Fraunhofer FIRST and idalab on the *in-house* set of data were evaluated by researchers at Bayer Schering Pharma on the *in-house validation* set of data. At this point in time, the modelers had no knowledge of the nature or log D values of the validation set. Later, the validation data was revealed to the modelers and used as an external validation set to assess the performance of other types of models.

3.4 Molecular Descriptors

We use the Dragon descriptors by Todeschini et al.[28]. They are organized in 20 blocks and include, among others, constitutional descriptors, topological descriptors, walk and path counts, eigenvalue-based indices, functional group counts and atom-centered fragments. A full list including references can be found online[29].

As one of their most pronounced features, Gaussian Process models allow to assign weights to each descriptor that enters the model as input. The similarity for two compounds as computed by the GP model takes into account that the ith descriptor contributes to the similarity with weight w_i (see 3.5.4). These weights are chosen automatically during model fitting and can then be inspected in order to get an impression of the relevance of individual descriptors.

We found that using a small (< 50) set of descriptors results in only slightly decreased accuracy when comparing to models built on the full set of 1,664 descriptors. The error predictions, however, turn out to be too optimistic in this case. Including whole blocks containing important descriptors leads to both accurate predictions and accurate error estimations (see Sec. 4.1). In this study, we used the full Dragon blocks 1, 2, 6, 9, 12, 15, 16, 17, 18, and 20. A discussion of the importance of individual descriptors can be found in Sec. 4.1.

3.5 Machine Learning Methods

3.5.1 Introductory remarks

Since the application of Gaussian Process regression is still relatively new in the field of chemoinformatics we chose to explain and illustrate the modeling idea. Support vector machines are seen as established, but still deserve some discussion due to interesting parallels and differences with the Bayesian GP approach.

Linear ridge regression, decision trees and ensembles of trees (random forests) are considered established methods - here we mainly note how the employed implementation differs from the original algorithm, for which the reader is referred to the literature.

3.5.2 Linear Ridge Regression

Ridge regression combines a linear model with a regularization term that effectively shrinks coefficients of the model towards zero. This is particularly important for our application since a standard linear model runs into problems when descriptors are correlated. We choose the complexity parameter λ that controls the amount of shrinkage by grid search in nested cross-validation.

3.5.3 Random Forest

A modified version the random forests method of Breiman[21] is employed. Trees are constructed without bagging or bootstrapping and pruning of individual

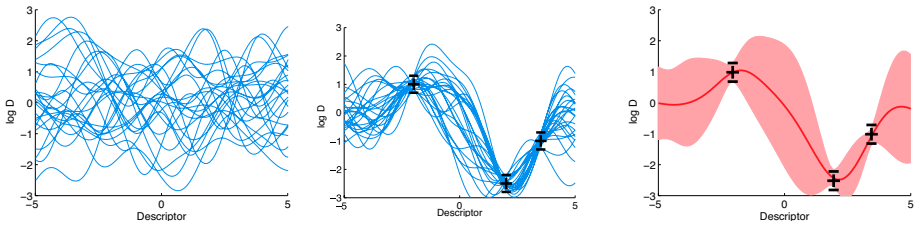


Figure 4: Bayesian modeling with Gaussian Processes

trees is done using a CART-style error-size trade-off.

The predictive variance is calculated by a averaging the variance of the predictions from the different trees in the forest and the average estimated variance from training points found at each tree leaf.

3.5.4 Gaussian Process Regression

Gaussian Process (GP) models have their origin[30] in the field of Bayesian statistics. A description of the methodology, including mathematical derivations, can be found in Schwaighofer et al.[6]. For in depth coverage we refer the reader to a recent book by Rasmussen[3].

Figure 4 illustrates the principles behind GP models: Before having measured log D values, any relationship between the descriptor (in this 2 dimensional example, only one descriptor is used and plotted on the x-axis) and log D (y-axis) is equally likely. This is represented by an infinitely large family of functions that map from descriptor space to log D space. The family is described by a *Gaussian Process prior*, 25 examples are shown in Figure 4 (left).

When training the model with log D values for a number of molecules (symbolized by black crosses in Figure 4 (middle)), we discard (or put lower weight on) all functions that do not pass near by these known data points.

To predict log D values for new molecules, we just average over the functions remaining in the pool (the red line in Figure 4 (right)) and read off the value corresponding to the new molecules' descriptors. To predict error bars, we calculate the standard deviation of the functions remaining in the pool at the position given by each new molecule's descriptors. The 2σ environment for all descriptor values on the x-axis is marked by the red region in Figure 4 (right). Close to known points, the uncertainty is small, but not zero: Measurements are assumed to be noisy. The uncertainty increases far from known points and in regions where measurements disagree.

Effectively, all the steps described above are not implemented by sampling, but via integral operations[6]. The Bayesian concept of a weighed average of functions with a certain mean (log D prediction) and standard deviation (error bar) is, however, preserved.

In order to derive the GP model prediction, let f be a function that depends on a vector $\mathbf{x}$ of d molecular descriptors and outputs log D, i.e. $f(\mathbf{x}) \approx \log \mathrm{D}(\mathbf{x})$.

We assume that each possible function f is a realization of a Gaussian stochastic process, and thus can be fully described by considering pairs of compounds $\mathbf{x}$ and $\mathbf{x}'$. By the properties of the Gaussian process, functional values $f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)$ for any finite set of n points form a Gaussian distribution. The covariance for each pair is then given by the covariance function,

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

which has a role similar to the kernel function in Support Vector Machines[31;8] and other kernel based learning methods. Any previous knowledge of the phenomenon to be predicted is expressed in the covariance function k.

For n compounds the actual data consist of n $\log \mathrm{D}$ measurements, $y_1, \ldots y_n$ and n descriptor vectors, $\mathbf{x}_1 \ldots \mathbf{x}_n$, (each of length d).

Assuming that measurements are noisy, we relate the n measured values to the true $\log \mathrm{D}$ by

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

where ϵ is Gaussian noise with standard deviation σ[4]

Applying a number of transformations and steps of statistical inference[6] we find that the predicted $\log \mathrm{D}$ for a new compound $\mathbf{x}_*$ follows a Gaussian distribution with mean $\bar{f}(\mathbf{x}_*)$ and standard deviation $\operatorname{std} f(\mathbf{x}_*)$, with

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

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

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

Details on inferring the parameters of the covariance function k and the measurement noise σ can be found in Schwaighofer et al.[6] .

Recent developments of approximation and sampling techniques[32] allow to train Gaussian process models on thousands of data points. However, memory demand and computing time still increase with the third power of the number of data points (compounds). For the larger datasets treated in this study, we therefore precede the actual GP training with a k-means clustering, such that each cluster contains up to 5000 compounds and train one GP per cluster. When applying the model, predictions from the individual GP models are generated and the prediction with the highest confidence (smallest error bar) is chosen.

[4] σ can be a scalar, meaning that all measurements are equally noisy. σ can also be a vector, allowing, in principle, to use a different noise level for each individual compound. In practice we found it useful to assume equal measurement noise for groups of compounds that e.g. have been measured in the same laboratory. In this way, model performance can be improved and we can learn the noise level resulting from different (or uniform) experimental procedures directly from the data[6].

3.5.5 Support Vector Regression

Support Vector Machines for regression and classification are based on the principle of structural risk minimization. Out of a certain class of functions we want to find the function that minimizes some notion of error, measured by the so-called loss function. Using a very large class of functions (i.e., a very complex model) one can perfectly fit to the training data, but the resulting function will not generalize to new, unseen data (over-fitting). On the contrary, using a small class of functions (simple, e.g. linear models) one may not be able to fit the data reasonably, again resulting in inaccurate predictions.

Choosing a function class with functions of the right complexity can be achieved by regularization: We combine the empirical loss on the training data with a penalty term for the complexity and then minimize the sum (objective function). Under certain assumptions (for example, that the training and test data are sampled from the same distribution), it can be proven that this way of choosing the function class leads to an optimal model[33;34;35].

In the following we will first describe the idea behind linear SVR and then generalize to the non-linear case.

Given a vector $\mathbf{x}$ of descriptors for a compound, the quantity of interest y (in our case $\log \mathrm{D}$) will be predicted as $y = f(\mathbf{x})$. Linear SVM finds 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 minimal. We employ an ϵ-insensitive loss function which does not penalize deviations from the measured value that are smaller than ϵ. Model training is done by solving the convex quadratic optimization problem:

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

The threshold ϵ from the loss function manifests in the constraints. "Slack variables" ξ are introduced and penalized in the objective function such that deviation by more than ϵ increases the objective function only linearly. This reduces the influence of outliers in the data. The constants ϵ and C are chosen by cross-validation[5].

Employing the so-called kernel trick[8;35] one can generalize to non-linear models. Functions f of the form $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b$ can be generated by rewriting the liner SVM equations such that the descriptors $\mathbf{x}$ only appear inside scalar products $(\mathbf{x}_i^\top \mathbf{x}_j)$. These scalar products can then be replaced by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$, that implicitly maps the descriptors into a high-dimensional feature-space and computes the scalar product there. An interesting connection with Gaussian Processes exists: Valid kernel functions for support vector algorithms are also valid covariance functions for a GP model

[5] In principle, it is also possible to use more sophisticated approaches[36] that compute SVR solutions for multiple parameter values in a efficient manner.

and vice versa. In this study, we do regression using Support Vector Machines with the RBF kernel.

4 Results & Discussion

4.1 Choice of Descriptors

Gaussian Process models can assign weights to each descriptor that enters the model as input (see Sec. 3.4 for details). The importance of individual descriptors was evaluated using subsets of the training data. The 30 interpretable descriptors with highest weight are clearly connected with $\log \mathrm{P}$ and $\log \mathrm{D}_7$. They include the sum of geometrical distances between pairs of oxygen atoms, counts of the following functional groups,

- donor atoms for H-bonds (N and O)
- H attached to hetero atom
- hydroxyl groups
- hydroxyl groups in phenol, enol, carboxyl
- ether groups
- oxygen atoms
- benzene-like rings
- carbon atoms
- quaternary nitrogen
- tertiary amines
- secondary amines

and a number of continuous quantities:

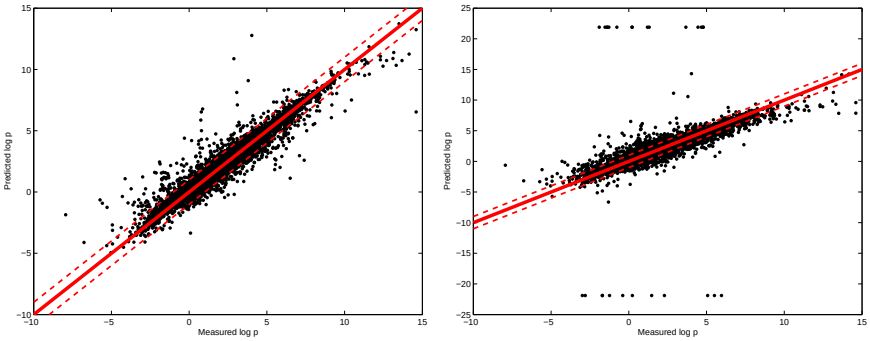
- topological polar surface area using N, O polar contributions
- topological polar surface area using N, O, S, P polar contributions
- mean atomic van der Waals volume (scaled on Carbon atom)
- harmonic oscillator model of aromaticity index total
- molar refractivity
- hydrophilic factor[?]
- molecular weight and 11 other measures of size, e.g. sum of conventional bond orders, sum of atomic van der Waals volumes and size indices

Public data Physprop/Beilstein	MAE	RMSE	$\% \pm 1$
Gaussian Process	0.38	0.66	92.6
Linear Ridge Regression	0.59	0.89	84.4
Support Vector Machine	0.40	0.71	91.8
Random Forest	0.52	0.82	87.6
ACDLabs v9	0.43	0.90	89.2
Wskowwin v1.41	0.25	0.90	91.6
AdmetPredictor v1.2.3	0.65	1.32	86.9
QikProp v2.2	0.76	1.23	79.6
baseline: predict mean $\log \mathrm{P}$	1.68	2.24	40.7

Table 2: Accuracy achieved on the public data sets Physprop/Beilstein using different machine learning methods compared with the performance of commercial tools. MAE, RMSE and $\% \pm 1$ denote the mean absolute error, the root mean squared error, and the percentage of compounds predicted with less than one log unit error.

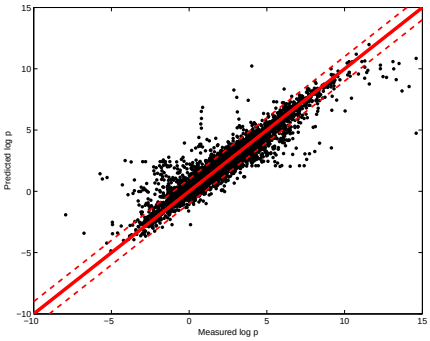
We found that using a small set of descriptors results in only slightly decreased accuracy when comparing to models built on the full set of 1,664 descriptors. The error predictions, however, turn out to be too optimistic. In other words: The $\log \mathrm{D}_7$ is predicted accurately for most compounds, but the model can not correctly detect whether the test compound has, for example, additional functional groups. These functional groups might not have occurred in the training data, and were thus not included by the feature selection step. In the test case, the information about these additional functional groups is important since it helps to detect, that these compounds are different from those the model has been trained on, i.e. the error bar should increase. Including whole blocks containing important descriptors leads to both accurate predictions and accurate error estimations. For, e.g. , a GP model these *surplus* descriptors will get only a small weight during training – but the weight will not be zero. In consequence the model has more information than it needs for predicting $\log \mathrm{D}_7$ and will respond to new properties (functional groups etc.) of molecules by estimating a larger prediction error.

In this study, we used the full Dragon blocks 1, 2, 6, 9, 12, 15, 16, 17, 18 and 20, thereby including constitutional descriptors, topological descriptors, 2D autocorrelations, topological charge indices geometrical descriptors, WHIM descriptors, GETAWAY descriptors, functional group counts, atom-centered fragments and molecular properties. With this set of 904 descriptors, the models accuracy is only slightly smaller the accuracy of models built on all 1,664 descriptors, but the computational cost and memory requirements are significantly reduced, and predicted error bars display close to ideal statistical properties (see Sec. 4.4 and Sec. 4.5).

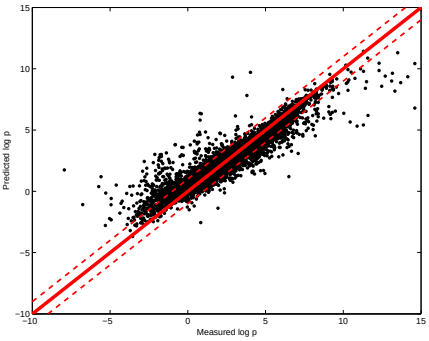


(a) GP

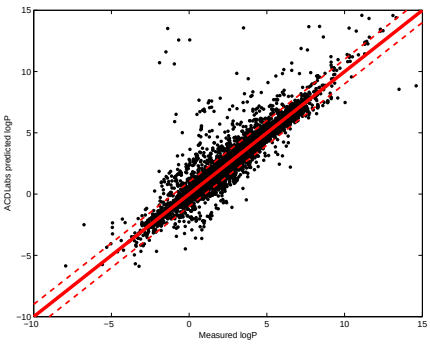
(b) Ridge regression



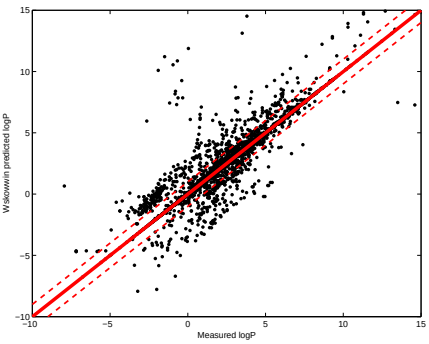
(c) SVM



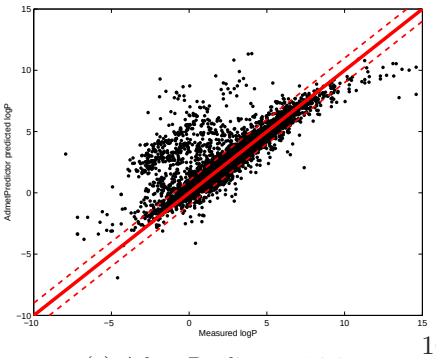
(d) Random forests



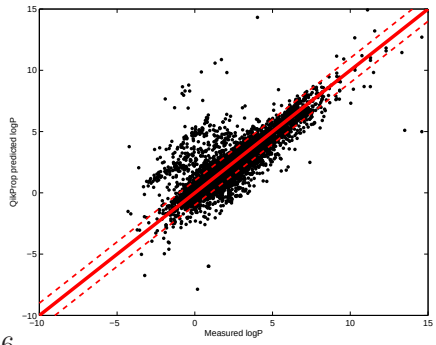
(e) ACDLabs v9



(f) Wskowwin v1.41



(g) AdmetPredictor v1.2.3



(h) QikProp v2.2

Figure 5: Scatter plots for GP, SVM, Ridge regression and random forests (one arbitrarily chosen cross-validation run each) and all four commercial tools on the public data set (Physprop/Beilstein)

4.2 Overall Accuracy: Public Data

The accuracy achieved on the public data set using different machine learning methods is compared with the performance of ACDLabs v9, Wskowwin v1.41, AdmetPredictor v1.2.3 and QikProp v2.2 in Table 2. The row labeled "baseline" lists the performance achieved when constantly predicting the average $\log P$ of the dataset. Scatter plots for all methods (one arbitrarily chosen cross-validation run each) and all four commercial tools are given in Figure 5.

The Support Vector Machine and random forest models exhibit similarly high performance (91.6% resp. 87.6% correct within one log unit) as the three best performing commercial tools ACDLabs v9, Wskowwin v1.41 and AdmetPredictor v1.2.3 (86.9% to 91.6 % correct ± 1). The Gaussian Process model performs slightly better (92.6 % $\pm$ 1) than the best performing commercial tool (91.6 % $\pm$ 1). The linear Ridge Regression model predicted a number of $\log P$ values as high as 10^{16}. For all plots and statistical evaluations, predictions from the linear Ridge Regression model were post processed, setting 1.5 times the highest/lowest $\log P$ values in the training data as upper/lower limits. Thus, error measures like mean absolute error can be used in a more meaningful way. 84.4 % of all predictions were correct within one log unit. In general, we found that the non-linear methods are more accurate and, in particular, produce fewer "far off" predictions, as can be seen in Figure 5 a, c and d.

Examining Figure 5 e through h, we find that all four commercial tools produce a number of outliers. ACDLabs v9 and Wskowwin v1.41 generate less than 10 very "far off" predictions, but their $\log P$ is overestimated by more than 10 orders of magnitude. For ≈ 50 compounds the predicted values are too high by two or three log units. Still, the overall performance of both ACDLabs v9 and Wskowwin v1.41 is good, which is also reflected in the low MAE and RMSE, see Table 2. Neither QikProp v2.2 nor AdmetPredictor v1.2.3 produce very "far off" predictions ($>$ ten orders of magnitude). For several hundreds of compounds, $\log P$ is predicted too high by two or three orders of magnitude, reducing the overall performance (measured by MAE, RMSE and the percentage of compounds correct within one log unit, see Table 2).

All four commercial tools have been trained using a number of compounds that are also included in the Beilstein and Physprop databases. In these cases the correct value is reproduced, rather than predicted. This effect can be seen most clearly in the results for Wskowwin, where many of the model predictions for the public data are right on the optimal prediction line. Thus, the presented evaluation is, most likely, biased in favor of the commercial tools.

Our own results were obtained in 2-fold cross-validation (train on half of the data, evaluate on the other half), repeated 10 times with different random splits of the data. Therefore, test and training data tend to have a similar distribution across different compound classes. This is not the case in the typical application scenario of such models: In new projects, new compound classes will be investigated, resulting in less accurate predictions. To get a realistic estimate of the performance on unseen data, a "blind test" evaluation on data including different compound classes is important. For models built on the Bayer Schering

In-house cross-validation	MAE	RMSE	$\% \pm 1$
Gaussian Process	0.41	0.66	90.7
Linear Ridge Regression	0.53	0.96	88.3
Support Vector Machine	0.44	0.70	89.8
Random Forest	0.55	0.80	84.4
ACDLabs v9	1.41	1.90	46.6
baseline: predict mean $\log D_7$	1.13	1.47	53.4

In-house blind test	MAE	RMSE	$\% \pm 1$
Gaussian Process	0.60	0.82	81.2
Linear Ridge Regression	0.60	0.83	82.2
Support Vector Machine	0.58	0.81	81.6
Random Forest	0.74	1.00	74.8
ACDLabs v9	1.40	1.79	44.2
baseline: predict mean $\log D_7$	1.17	1.51	51.7

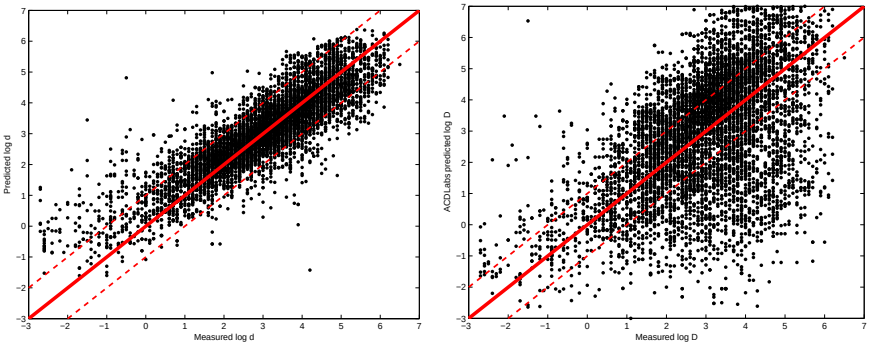
Table 3: Accuracy achieved using Gaussian Process models, Support Vector Machines, linear Ridge Regression and Random Forests for the respective datasets, compared with the performance of ACDLabs v9. MAE, RMSE and $\% \pm 1$ denote the mean absolute error, the root mean squared error, and the percentage of compounds predicted with an error less than one

Pharma in-house data, we present such an evaluation in the subsequent section.

4.3 Overall Accuracy: In-house Data

The results for predicting $\log D_7$ on Bayer Schering Pharma in-house data are listed in Table 3. The corresponding scatter-plots are given in Figure 6. When evaluated in 2-fold cross-validation on the in-house data (see Table 3, top), the Gaussian Process model, the Support Vector Machine and the linear Ridge Regression yielded good results (88.3 to 90.7 % correct within one log unit), with the Gaussian Process model performing best ($90.7\% \pm 1$). This model was then validated in blind evaluation at Bayer Schering Pharma on a set of 7013 new measurements from the last months. Later, the data was made available to the modeling team at Fraunhofer and idalab and other methods were evaluated, treating the former blind test data as an external validation set. These results are given in Table 3 (bottom). Amongst the commercial tools that were available to us, only ACDLabs is able to calculate $\log D_7$, and can thus be used as a benchmark.

With ACDLabs v9, only 44.2% of the compounds are predicted correctly within one log unit. Mind that ACD has been trained on shake-flask measurements, while the in-house measurements used in this study were performed with the HPLC methodology described in Sec. A. With our tailored models, we achieved 81.2% to 82.2% correct predictions. These are very good results, con-



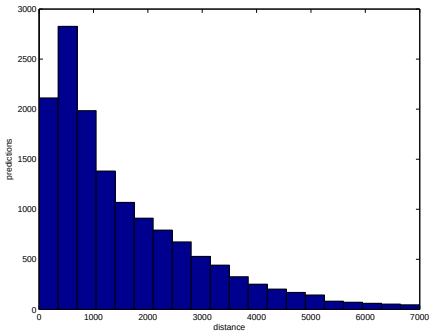
(a) GP (in-house blind test)

(b) ACD (in-house blind test)

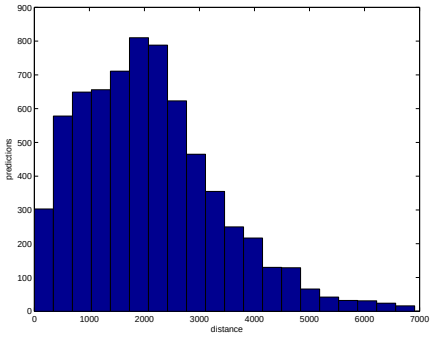
(c) GP (in-house cross-validation)

(d) ACD (in-house cross-validation)

Figure 6: Scatter-plots for Gaussian Process and ACDLabs v9 on in-house validation data in blind test (subplots a, b) and on in-house data in cross-validation (subplots c,d)



(a) In-house cross-validation



(b) In-house blind test

Figure 7: Histograms of Mahalanobis distances from each compound to the closest compound in the respective training set. Distances for the cross validated in-house setup a) were calculated for the training/validation-split of one arbitrarily chosen cross-validation run.

environment	pred $\pm \sigma$	pred $\pm 2\sigma$
optimal %	68.7	95.0
GP	67.5	90.4
RR	62.6	88.0
SVM	63.7	87.9
forest	62.5	90.2

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

sidering that the structures were at no point in time available to the modeling team at FIRST/idalab. Furthermore, the blind test data stems from new drug discovery projects, and thus represents different structural classes than those present in the training data.

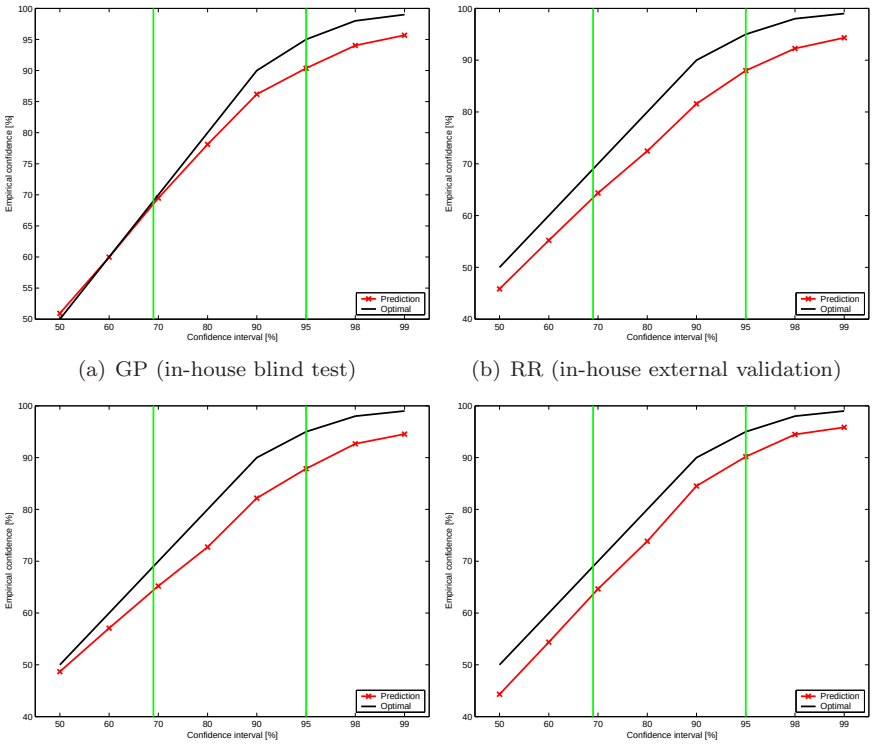
The fact that performance decreases when comparing the results achieved in cross-validation with the blind test could be taken as a hint that the non-linear models did overfit to their training data. However, typical symptoms of overfitting, like a too large number of support vectors in SVM models, were not present. A large fraction of all compounds in the validation set is, however, very dissimilar to the training data. Histograms of Mahalanobis distances from each compound in the validation to the closest training compound are presented in Figure 7. We used the same set of descriptors for both model building and distance calculation.

In a typical cross-validation run on the in-house data set, 50 % of the compounds have a nearest neighbor closer than 1100 units, see Figure 7, top. In the blind test set, less than 25 % of the compounds have neighbors closer that 1100 units, see Figure 7 bottom.

This supports our hypothesis that the difference in performance between the cross-validation results and the blind test is caused by a large number of compounds being dissimilar to the training set compounds. Therefore it should be possible to achieve higher performance by focusing on compounds that are clearly inside the domain of applicability of the respective model. We investigate this question in Sec. 4.5.

4.4 Individual Error Estimation for Interactive Use

Researchers establishing error estimations based on the distance of compounds to the training data typically present plots or tables where prediction errors are binned by distance, i.e., averaging over a large number of predictions, because the correlation between distances and errors is typically not too strong when considering individual compounds. When binning by the distance, one can clearly see how the error increases as the distance increases[17;14]. One can fit



(a) GP (in-house blind test)

(b) RR (in-house external validation)

(c) SVM (in-house external validation)

(d) forest (in-house external validation)

Figure 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 2σ environments, the corresponding numbers can be found in Table 4.

a function to this relationship and use it to generate an error prediction for each prediction the model makes. But how does the user know, what an error prediction of e.g. 0.6 log units really means? In how many cases does the user expect the error to be larger than the predicted error? How much larger can errors turn out?

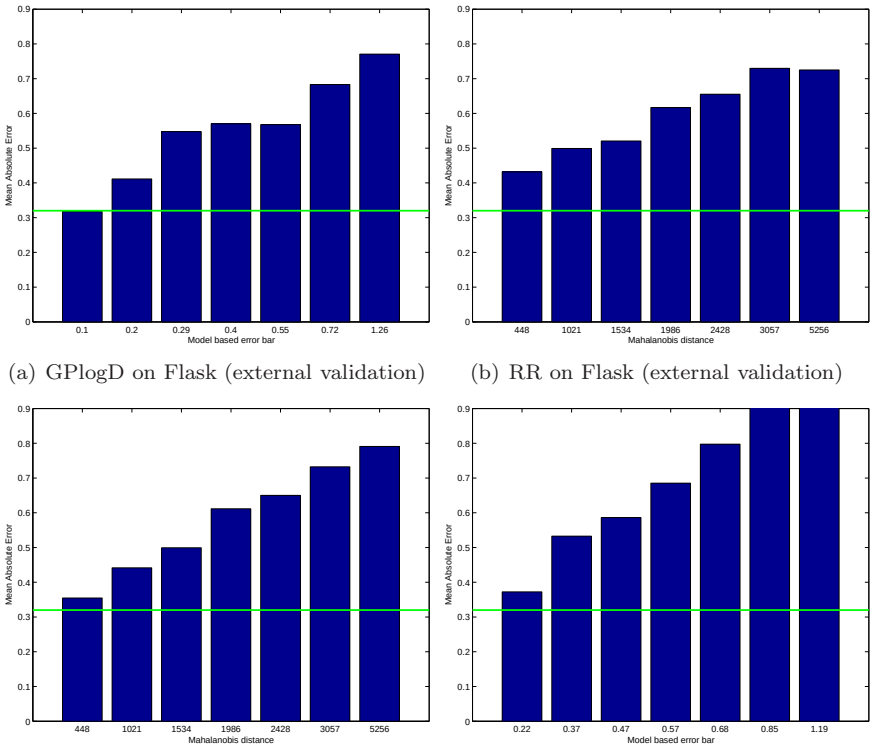
The most commonly used description of uncertainty (such as measurement errors, prediction errors, etc.) in chemistry, physics and other fields is the error bar. Its definition is based on the assumption that errors follow a Gaussian distribution. When using a probabilistic model that predicts a Gaussian (i.e., a mean $\bar{f}$ and a standard deviation σ), it follows that the true value has to be in the interval $\bar{f} \pm \sigma$ with 68% confidence, and in the interval $\bar{f} \pm 2\sigma$ with 95% confidence, etc. To evaluate the quality of the predicted error bars, one can therefore compare with the true experimental values, and count how many of them are actually within the σ, 2σ etc. intervals.[6]

Gaussian Process model can directly predict error bars. In the implementation of random forests used in this study, the predictive variance is calculated by averaging the variance of the predictions from the different trees in the forest and the average estimated variance from training points found at each tree leaf.

For the and linear Ridge Regression models and the Support Vector Machines, error bars were estimated by fitting exponential and linear functions to the errors observed when evaluating the models in cross-validation and the mahalanobis distances to the closest neighbours in the training set of the respective split. Since both linear and exponential functions worked equally well, we chose the simple linear functions to estimate to error bars from the distances.

Plots of the empirical confidence versus the confidence interval are presented in Figure 8 (red line). The optimal curve is marked by the black line. The σ and 2σ environments are marked by green lines, with the corresponding percentages of predictions within each environment being listed in Table 4. Predicted error bars of all four models exhibit the correct statistical properties, with the GPlogD error predictions being closest to the ideal distribution. The results presented for the GP model stem from a "blind test" of the final model delivered to Bayer Schering Pharma[4;5;6;7;8]. The remaining algorithms have been evaluated a posteriori, after the experimental values for the validation set had been revealed.

In conclusion, using Bayesian models, ensemble models or distance based approaches one can not only identify compounds outside of the models domain of applicability, but also quantify the reliability of a prediction in a way that is intuitively understandable for the user.



(a) GPlogD on Flask (external validation)

(b) RR on Flask (external validation)

(c) SVM on Flask (external validation)

(d) Forest on Flask (external validation)

Figure 9: Mean absolute error achieved when binning by the model based error bar (in the case of the GP and the random forest) resp. the Mahalanobis distance to the closest point in the training set (linear Ridge Regression and Support Vector Machines do not provide error bars). Each bin contains one seventh (1000 compounds) of the in-house validation set. Corresponding numbers can be found in Table 5.

error bar (average in bin)	0.10	0.20	0.29	0.40	0.55	0.72	1.26
MAE GP	0.32	0.41	0.55	0.57	0.57	0.68	0.77
error bar (average in bin)	0.22	0.37	0.47	0.57	0.68	0.85	1.19
MAE (forest)	0.37	0.53	0.59	0.69	0.80	0.95	1.24
distance (average in bin)	448	1021	1534	1986	2428	3057	5256
MAE (RR)	0.43	0.50	0.52	0.62	0.66	0.73	0.73
MAE (SVM)	0.35	0.44	0.50	0.61	0.65	0.73	0.79

Table 5: Mean absolute error achieved when binning by the model based error bar (for GP and random forest) resp. the Mahalanobis distance to the closest point in the training set (linear Ridge Regression and SVM, since these methods do not provide model-based error bars). Bins were chosen such that each contains one seventh (around 1000 compounds) of the in-house validation set. A graphical representation of this information can be found in Figure 9.

4.5 Increasing Accuracy by focusing on the Domain of Applicability

In Sec. 4.3 we presented statistics obtained by applying our models to all compounds in the respective test sets, without considering the models' domain of applicability. In Sec. 4.4 we have evaluated methods for quantifying the confidence in predictions, and found that this can be achieved in a reliable way. Therefore it should be possible to increase model performance by focusing on more confident predictions or, in other words, on compounds clearly inside the domain of applicability.

In Figure 9 we present a histogram-like bar plot obtained in the following way: We assign compounds to bins based on the confidence in the prediction, i.e., the model based error bar (GP and random forest) or distance to training points (for ridge regression and SVM), such that each of the seven bins contains 1000 compounds (one seventh of the in-house validation data). Within each bin (representing a different degree of confidence in the predictions), we compute the mean absolute error. For each algorithm tested, the mean absolute error decreases, as compounds in bins with higher confidence are considered. In case of the GP model, the mean absolute error decreases from 0.55 to 0.42, when focusing on the 3000 compound with the lowest predicted error bars. When focusing on the 1000 compounds with lowest predicted error bars, the mean absolute error can even be reduced to only 0.32 log units.

In conclusion, focusing on confident predictions, i.e., compounds within the domain of applicability, allows us to achieve more accurate predictions than we found when validating models on the whole in-house validation set (Table 3). The previously observed decrease in performance relative to the cross-validation

[6] We found that numeric criteria, i.e., the log probability of the predictive distribution, can be misleading.

on the training data can therefore be avoided.

5 Summary

We presented results of modeling lipophilicity using the Gaussian Process methodology on public and in-house data. The statistical evaluations show that the prediction quality of our GP models compares favorably with four commercial tools and three other machine learning algorithms that were applied to the same sets of data. The positive results achieved with the model on in-house drug discovery compounds are re-confirmed by a blind evaluation on a large set of measurements from new drug discovery projects at Bayer Schering Pharma.

It should be noted that GP models are not only capable of making accurate predictions, but can also provide fully automatic adaptable tools: Using a Bayesian model selection criteria allows for re-training without user intervention whenever new data becomes available. As a further advantage for every day use in drug discovery applications, GP models quantify their domain of applicability in a statistically well founded manner. The confidence of each prediction is quantified by error bars, an intuitively understood quantity. This allows both, increasing the average accuracy of predictions by focusing on predictions that are inside the domain of applicability of the model, and judging the reliability of individual predictions in interactive use.

Acknowledgments

The authors gratefully acknowledge DFG grant MU 987/4-1 and partial support from the PASCAL Network of Excellence (EU #506778). 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.

A Appendix: Measuring log D_7 using HPLC

High Performance Liquid Chromatography (HPLC) is performed on analytical columns packed with a commercially available solid phase containing long hydrocarbon chains chemically bound onto silica. Chemicals injected onto such a column move along it by partitioning between the mobile solvent phase and the hydrocarbon stationary phase. The chemicals are retained in proportion to their hydrocarbon-water partition coefficient, with water-soluble chemicals eluted first and oil-soluble chemicals last. This enables the relationship between the retention time on a reverse-phase column and the n-octanol/water partition coefficient to be established. The partition coefficient is deduced from the capacity factor $k = \frac{t_r - t_0}{t_0}$, where t_r is the retention time of the test substance and t_0 is the dead time, i.e., the average time a solvent molecule needs to pass the column. In order to correlate the measured capacity factor k of a compound with its $\log D_7$, a calibration graph is established. The partition coefficients

of the test compounds are obtained by interpolation of the calculated capacity factors on the calibration graph using a proprietary software tool "POW Determination".

A.1 Apparatus and Materials

Experiments are carried out following the *OECD Guideline for Testing of Chemicals No. 117*. A set of 9 reference compounds with known $\log D_7$values selected from this guideline is used.

- HPLC: Waters Alliance HT 2790 with DAD- and MS-detection
- Column: Spherisorb ODS 3 μm, 4.6×60 mm
- Mobile phase: Methanol/0.01 m Ammoniumacetate buffer (pH 7) 75:25
- Dead time compound: Formamide, Stock solution in MeOH
- Test compounds: 10 mmolar DMSO stock
- Reference compounds (Stock solutions in MeOH): Acetanilide, 4-Methylbenzyl alcohol, Methyl benzoate, Ethyl benzoate, Naphthalene, 1,2,4-Trichlorobenzene, 2,6-Diphenylpyridine, Triphenylamine, DDT.

References

[1] T.J. Hou and X.J. Xu. Adme evaluation in drug discovery. 3. modeling blood-brain barrier partitioning using simple molecular descriptors. *J. Chem. Inf. Comput. Sci.*, 43(6):2137–2152, 2003.

[2] Pierre Bruneau and Nathan R. McElroy. Generalized fragment-substructure based property prediction method. *J. Chem. Inf. Model.*, 46: 1379–1387, 2006.

[3] Carl Edward Rasmussen and Christopher K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2005.

[4] Timon Schroeter, Anton Schwaighofer, Sebastian Mika, Antonius Ter Laak, Detlev Suelzle, Ursula Ganzer, Nikolaus Heinrich, and Klaus-Robert Müller. Predicting lipophilicity of drug discovery molecules using gaussian process models. *ChemMedChem*, 2007. submitted.

[5] Timon Schroeter, Anton Schwaighofer, Sebastian Mika, Antonius Ter Laak, Detlev Suelzle, Ursula Ganzer, Nikolaus Heinrich, and Klaus-Robert Müller. Estimating the domain of applicability for machine learning qsar rmodels: A study on aqueous solubility of drug discovery molecules. *J. Comput.-Aided Mol. Des.*, 2007. submitted.

[6] Anton Schwaighofer, Timon Schroeter, Sebastian Mika, Julian Laub, Antonius ter Laak, Detlev Sülzle, Ursula Ganzer, Nikolaus Heinrich, and Klaus-Robert Müller. Accurate solubility prediction with error bars for electrolytes: A machine learning approach. *Journal of Chemical Information and Modelling*, 47(2):407–424, 2007. URL `http://dx.doi.org/10.1021/ci600205g`.

[7] Klaus-Robert Müller, Gunnar Rätsch, Sören Sonnenburg, Sebastian Mika, Michael Grimm, and Nikolaus Heinrich. Classifying 'drug-likeness' with kernel-based learning methods. *J. Chem. Inf. Model*, 45:249–253, 2005.

[8] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[9] Frank R. Burden. Quantitative structure-activity relationship studies using Gaussian processes. *J. Chem. Inf. Comput. Sci.*, 41(3):830–835, 2000.

[10] D.P. Enot, R. Gautier, and J.Y. Le Marouille. Gaussian process: an efficient technique to solve quantitative structure-property relationship problems. *SAR QSAR Environ. Res.*, 12(5):461–469, 2001.

[11] Peter Tino, Ian Nabney, Bruce S. Williams, Jens Lösel, and Yi Sun. Nonlinear prediction of quantitative structure-activity relationships. *J. Chem. Inf. Comput. Sci.*, 44(5):1647–1653, 2004.

[12] Alexander Tropsha. Variable selection qsar modeling, model validation, and virtual screening. In David C. Spellmeyer, editor, *Annual Reports in Computational Chemistry*, volume 2, chapter 7, pages 113–126. Elsevier, 2006.

[13] Weida Tong, Qian Xie, uixiao Hong, Leming Shi, Hong Fang, and Roger Perkins. Assessment of prediction confidence and domain extrapolation of two structure-activity relationship models for predicting estrogen receptor binding activity. *Environmental Health Perspectives*, 112(12):1249–1254, 2004.

[14] Tatiana I. Netzeva, Andrew P. Worth, Tom Aldenberg, Romualdo Benigni, Mark T.D. Cronin, Paola Gramatica, Joanna S. Jaworska, Scott Kahn, Gilles Klopman, Carol A. Marchant, Glenn Myatt, Nina Nikolova-Jeliazkova, Grace Y. Patlewicz, Roger Perkins, David W. Roberts, Terry W. Schultz, David T. Stanton, Johannes J.M. van de Sandt, Weida Tong, Gilman Veith, and Chihae Yang. Current status of methods for defining the applicability domain of (quantitative) structure-activity relationships. *Alternatives to Laboratory Animals*, 33(2):1–19, 2005.

[15] Ralph Kühne, Ralf-Uwe Ebert, and Gerrit Schüürmann. Model selection based on structural similarity-method description and application to water solubility prediction. *J. Chem. Inf. Model.*, 46:636–641, 2006.

[16] Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Number 26 in Monographs on Statistics and Applied Probability. Chapman & Hall, 1986.

[17] Pierre Bruneau and Nathan R. McElroy. Generalized fragment-substructure based property prediction method. *J. Chem. Inf. Model.*, 44: 1912–1928, 2004.

[18] Igor V. Tetko, Pierre Bruneau, Hans-Werner Mewes, Douglas C. Rohrer, and Gennadiy I. Poda. Can we estimate the accuracy of ADME-tox predictions? *Drug Discovery Today*, 11(15/16):700–707, August 2006.

[19] Andreas H. Göller, Matthias Hennemann, Jörg Keldenich, and Timothy Clark. In silico prediction of buffer solubility based on quantum-mechanical and hqsar- and topology-based descriptors. *J. Chem. Inf. Model.*, 46(2): 648–658, 2006.

[20] David T. Manallack, Benjamin G. Tehan, Emanuela Gancia, Brian D. Hudson, Martyn G. Ford, David J. Livingstone, David C. Whitley, , and Will R. Pitt. A consensus neural network-based technique for discriminating soluble and poorly soluble compounds. *J. Chem. Inf. Model.*, 43:674–679, 2003.

[21] Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001. URL `http://dx.doi.org/10.1023/A:1010933404324`.

[22] Andreas Bender, Hamse Y. Mussa, and Robert C. Glen. Screening for dihydrofolate reductase inhibitors using molprint 2d, a fast fragment-based method employing the nave bayesian classifier: Limitations of the descriptor and the importance of balanced chemistry in training and test sets. *Journal of Biomolecular Screening*, 10(7):658–666, 2005. http://jbx.sagepub.com/cgi/content/abstract/10/7/658.

[23] Hongmao Sun. An accurate and interpretable bayesian classification model for prediction of herg liability. *ChemMedChem*, 1(3):315–322, 2006.

[24] J. Sadowski, C. Schwab, and J. Gasteiger. *Corina v3.1.* Erlangen, Germany.

[25] R. Todeschini, V. Consonni, A. Mauri, and M. Pavan. *DRAGON v1.2.* Milano, Italy, .

[26] *Physical/Chemical Property Database (PHYSPROP).* Syracuse, NY, USA.

[27] *Beilstein CrossFire Database.* San Ramon, CA, USA.

[28] Roberto Todeschini and Viviana Consonni. *Handbook of Molecular Descriptors.* John Wiley & Sons, Ltd., 2000.

[29] R. Todeschini, V. Consonni, A. Mauri, and M. Pavan. Dragon for windows and linux 2006. `http://www.talete.mi.it/help/dragon_help/` (accessed 14 May 2006), .

[30] Anthony O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society, Series B: Methodological*, 40(1):1–42, 1978.

[31] Bernhard Schölkopf and Alex J. Smola. *Learning with Kernels*. MIT Press, 2002.

[32] Joaquin Quionero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, December 2005. URL `http://www.jmlr.org/papers/volume6/quinonero-candela05a/quinonero-candela05a.pdf`.

[33] V.N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[34] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

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

[36] Gang Wang, Dit-Yan Yeung, and Frederick H. Lochovsky. Two-dimensional solution path for support vector regression. In Luc De Raedt and Stefan Wrobel, editors, *Proceedings of ICML06*, pages 993–1000. ACM Press, 2006. URL `http://www.icml2006.org/icml_documents/camera-ready/125_Two_Dimensional_Solu.pdf`.