Fast Conditional Density Estimation for Quantitative Structure-Activity Relationships

Fabian Buchwald, Tobias Girschick and **Stefan Kramer**
Institut für Informatik/I12, TU München,
Boltzmannstr. 3, 85748 Garching b. München, Germany
{fabian.buchwald, tobias.girschick, stefan.kramer}@in.tum.de

Eibe Frank
Department of Computer Science, University of Waikato,
Private Bag 3105, Hamilton, New Zealand
{eibe@cs.waikato.ac.nz}

Abstract

Many methods for quantitative structure-activity relationships (QSARs) deliver point estimates only, without quantifying the uncertainty inherent in the prediction. One way to quantify the uncertainy of a QSAR prediction is to predict the conditional density of the activity given the structure instead of a point estimate. If a conditional density estimate is available, it is easy to derive prediction intervals of activities. In this paper, we experimentally evaluate and compare three methods for conditional density estimation for their suitability in QSAR modeling. In contrast to traditional methods for conditional density estimation, they are based on generic machine learning schemes, more specifically, class probability estimators. Our experiments show that a kernel estimator based on class probability estimates from a random forest classifier is highly competitive with Gaussian process regression, while taking only a fraction of the time for training. Therefore, generic machine-learning based methods for conditional density estimation may be a good and fast option for quantifying uncertainty in QSAR modeling.

Introduction

Quantitative structure-activity relationships (QSARs) are models relating chemical structure to biological activity. Technically speaking, QSAR modeling is often framed as a regression problem on chemical structures. While most regression methods only provide point estimates of an activity, it would be desirable in many applications to also quantify the uncertainty inherent in the prediction, ideally in the form of a conditional density estimate. Given a conditional density estimate, it is easy to derive prediction intervals and other relevant information for that prediction. While conditional density estimation has been studied extensively in economics and Bayesian statistics (for example, in Gaussian Processes for regression (Rasmussen and Williams 2005)), it has received only little attention in the machine learning literature. In this paper, we evaluate generic, machine-learning based conditional density estimation, as investigated recently by Frank and Bouckaert (2009), for its use in QSAR modeling. In contrast to traditional methods like GP regression used, e.g., by Schwaighofer et al. (2007) in a QSAR context, the proposed methods are fast and offer some flexibility in the use of underlying class probability estimators (in our case, random forests (Breiman 2001)).

To illustrate the concept of conditional density estimation in the context of QSAR, Figure 1 shows the 2D chemical structure of a compound, its actual target value and its density distributions for the target variable obtained with three conditional density estimators: a histogram estimator, a normal estimator, and a kernel estimator (see below). The figure shows that the histogram estimator exhibits quite sharp discontinuities. In contrast, the normal estimator can only represent unimodal activity distributions and is therefore quite limited in its expressiveness. Finally, the kernel estimator is both quite flexible and capable of smoothing.

The paper is organized as follows. In the next section, we introduce the three conditional density estimators we apply. Following that we give an overview of the datasets and descriptors used in the experiments. Then, we describe the experiments that we conducted and discuss and analyze the results. Finally, we present our conclusions.

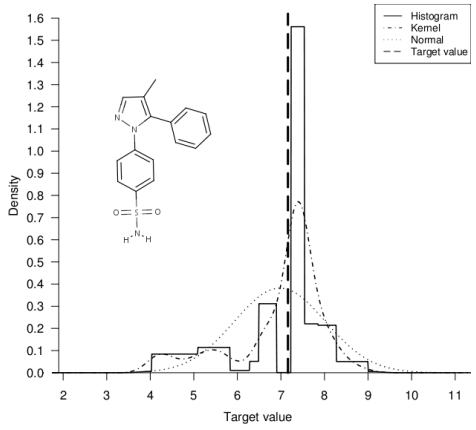

Figure 1: Exemplary conditional density estimates with random forests for molecule CID 10519259 (http://pubchem.ncbi.nlm.nih.gov/).

Conditional Density Estimation via Class Probability Estimation

Before we present the three conditional density estimators applied in our experiments we have to make some assumptions that are necessary to work with them. First, we assume that the continuous range of values of the target variable is transformed into bins by applying equal-frequency discretization. The binning can be seen as a transformation from a regression to a classification problem. To solve this problem we assume that we have a method that provides class probability estimates $p(c|X)$ for each class value c and an instance described by attribute values X. In this way we can obtain a weight w for each instance in the training data, conditioned on X. These weights are then used in a standard univariate density estimator (e.g. a normal estimator) to obtain a conditional density estimate $f(y|X)$.

A weight $w(y_i|X)$ for a particular target value y_i given an instance X is computed by "spreading" the predicted class probability for bin c_{yi} that contains y_i across all $n_{c_{yi}}$ target values in that bin. Formally it is given by:

$$w(y_i|X) = n\frac{p(c_{yi}|X)}{n_{c_{yi}}}$$

where n is the total number of target values in the training data and n_c is the number of target values in bin c. Intuitively the weight $w(y_i|X)$ can be seen as an estimate of how likely it is that a future observed target value associated with attribute values X will be close to y_i, based on the class probability estimation model that has been inferred from the discretized training data. In the following we will present three conditional density estimators that are based on these weights.

Histogram Density Estimator The histogram-based estimator is probably the most obvious univariate density estimator in this context since it is based on the bins provided by the discretization. In each bin the density is assumed to be constant and based on the bin's width r_c and the class probability or weighted target value assigned to it. The histogram estimator can be written as follows:

$$f_{bins}(y|X) = \frac{1}{n}\sum_{i=1}^{n} w(y_i|X)\frac{I(c_{yi}=c_y)}{r_{c_y}}$$

where $I(a)$ is set to 1 if the proposition a is true and 0 otherwise. Note that with this estimator potentially valuable information about the relative position of y with respect to the individual y_i in a bin is discarded and that no smoothing is performed, meaning that there can be sharp discontinuities at the interval borders.

Normal Density Estimator A simple estimator is the standard univariate normal estimator:

$$f_{normal}(y|X) = \mathcal{N}(\mu_X, \sigma_X^2)$$

with mean μ_X and variance σ_X that are both computed based on weighted target values. The mean is defined as

$$\mu_X = \frac{1}{n}\sum_{i=0}^{n} w(y_i|X)y_i$$

and the variance as

$$\sigma_X^2 = \frac{1}{n}\sum_{i=0}^{n} w(y_i|X)(y_i - \mu_X)^2$$

The normal estimator is suitable when the data is approximately normally distributed. If this is not the case, it is not able to model the data correctly due to its inflexibility. A popular non-parametric alternative that is also able to model smooth, but additionally able to model multi-modal distributions is the kernel density estimator.

Kernel Density Estimator In kernel density estimation an instance from the training dataset is represented with a Gaussian kernel (distribution). The mean of this Gaussian distribution is the target value of the training instance; its variance, called the kernel bandwidth σ_{kernel}, determines how closely the estimator fits the (weighted) data points. A small kernel bandwidth can lead to undersmoothing, and a large σ_{kernel} to oversmoothing. In our experiments we used the following data-dependent value based on the global (weighted) standard deviation and the number of data points: $\sigma_{kernel} = \frac{\sigma_X}{n^{1/4}}$. Formally, the kernel density estimator is given by:

$$f_{kernel}(y|X) = \frac{1}{n}\sum_{i=0}^{n} w(y_i|X)\mathcal{N}(y_i, \sigma_{Kernel}^2)$$

In the remaining part of the paper we evaluate these three conditional density estimators using QSAR datasets.

Datasets and Descriptors

In this section we describe the datasets that we used in our study. Furthermore, we present an overview of the structural and physicochemical descriptors used for encoding the molecules.

Datasets

For our experiments we chose eight well-known publicly available QSAR datasets from the cheminformatics web repository[1].

Sutherland

For the **BZR** (benzodiazepine receptor ligands), **COX2** (cyclooxygenase-2 inhibition) and **DHFR** (dihydrofolate reductase inhibitors) datasets (Sutherland, O'Brien, and Weaver 2003) the target variable is the logarithm of the half maximal inhibitory concentration ($\text{pIC}_{50} = -log\,(\text{IC}_{50})$). This quantitative measure is commonly used in pharmacology and indicates how much of a given substance is needed to block a biological activity by half. We had to remove a number of instances, which were considered to be inactive in the original publication and marked with default values.

ER This dataset (Sutherland, O'Brien, and Weaver 2004) lists measurements of relative binding affinities (RBA) of compounds to the estrogen receptor with respect to β-estradiol. We removed all inactive compounds from the dataset. Since it is based on substance concentrations, we again used its logarithm as a more meaningful target value.

CPDB The two CPDB datasets for mouse and rat data are generated from data obtained by the carcinogenic potency project.[2] The compounds' carcinogenicity is rated

[1] http://www.cheminformatics.org

[2] http://potency.berkeley.edu/chemicalsummary.html

Short-hand	n	Endpoint
BZR	333	pIC_{50}
$COX2$	414	pIC_{50}
$DHFR$	673	pIC_{50}
ER	410	log_{10}(RBA)
$CPDB\_mouse$	444	molar TD_{50}
$CPDB\_rat$	580	molar TD_{50}
$EPAFHM$	580	$log_{10}(LC_{50}\_mg)$
$FDAMDD$	1216	log_{10}(Dose_MRDD_mg)

Table 1: Overview of the datasets used for assessing our methods. n denotes the number of compounds in the respective dataset.

according to the TD_{50} value, a numerical measure of the carcinogenic potency. Based on a suggestion by the authors, we transformed the listed TD_{50} values to molar values: $TD_{50}^{m} = log_{10}\left(\frac{MW}{TD_{50}}\right)$, where MW is the molecular weight. The two datasets contained several instances where the actual structure of the compound was missing. If the molecule could be identified uniquely via the given CAS number, we downloaded the structure from the NCBI PubChem database[3]. If this was not possible, the molecule was removed from the set.

EPAFHM The EPAFHM (Russom et al. 1997) dataset was generated for the purpose of developing an expert system to predict acute toxicity from chemical structure. For our conditional density estimates we took the logarithm of the LC_{50} values.

FDAMDD The FDAMDD dataset (Matthews et al. 2004) was generated in order to predict the Maximum Recommended Daily Dose (MRDD) for pharmaceutical substances. Again, as for the EPAFHM dataset, we took the logarithm of the target values.

A brief overview of the datasets is given in Table 1. All datasets are available on the web[4].

Descriptors

For this paper we used three different sets of descriptors: closed free trees (cFTs), chemical descriptors (JOE) and a combination of both (cFTs + JOE). Firstly, we generated substructural descriptors with the Free Tree Miner software (FTM)(Rückert and Kramer 2004). More precisely, the generated substructures are frequently occurring unrooted trees. The frequency threshold was set individually for each dataset so that approximately 1000 free trees were found. The resulting set was further reduced to the set of closed trees, i.e., trees occurring in the same set of structures as one of the supertrees were removed. The occurrence of the substructures was encoded in binary fingerprints. Secondly, we used chemical descriptors computed with the cheminformatics library JOELib2[5] to represent the molecules. We obtained a chemical description in the form of a pharmacophoric (binary) fingerprint containing more than 50 chemical descriptors. For each instance, the existence of single atoms, functional subgroups or stereochemical descriptors was tested. Additionally, we computed combinations of the JOELib2 descriptors with the free tree descriptors to assess if this leads to a gain in performance.

The JOELib2 fingerprints contain some bits for substructure occurrences and are enriched with physicochemical properties of the molecule so we expect them to perform better than the cFTs which encode solely substructural descriptors. However, we expect the descriptor combinations to further enhance performance, as the JOELib2 substructural descriptors are not sufficient to capture all structural information of a molecule.

Experiments

We empirically compare the performance of the conditional density estimators described above using the methodology introduced in Frank and Bouckaert (2009). The experiments were conducted with the WEKA workbench (Hall et al. 2009). All results are obtained using 10-times 10-fold cross-validation. To test for significant differences, the corrected resampled paired t-test (Nadeau and Bengio 2003) is used at a 95% significance level.

Results are discussed using random forests as the underlying class probability estimators. We used the WEKA implementation of random forests and chose an ensemble size of 100 trees. To eliminate bias in the probability estimates, each tree was grown from 66% of the training data, sampled without replacement, and the remaining data was used to estimate the class probabilities at the leaf nodes. We obtained qualitatively similar results with linear SVMs. Since we have to deal with a discretized regression problem that is actually an ordinal classification problem, we applied WEKA's ordinal class classifier (Frank and Hall 2001) to exploit the ordering information in the discretized target variable (we use 10 bins) by creating multiple two-class problems, with the smoothing method from Schapire et al. (2002). For each of these binary classification problems class probability estimates were produced using random forests.

Mean entropy gain (MEG) The average log-likelihood represents one method to measure the performance of a conditional density estimator. This measure is made more informative by subtracting the log-likelihood obtained for an unconditional kernel density estimator, also called prior kernel, that is generated by applying a kernel density estimator to the unweighted target values. Formally the mean entropy gain is given by:

$$MEG = \frac{\sum_{i=0}^{n_{test}} log_2(f(y_i|X)) - log_2(f_{prior}(y_i))}{n_{test}},$$

where y_i is the actual target value of the test instance and n_{test} is the number of test instances. The desired outcome, a positive value, means that the conditional estimator manages to improve on the prior estimator. Note that in the context of the histogram estimator we perform some smoothing to avoid density values that are zero, see Frank and Hall (2001) for details.

[3]http://pubchem.ncbi.nlm.nih.gov/

[4]http://wwwkramer.in.tum.de/research/machine_learning/fcde

[5]http://www-ra.informatik.uni-tuebingen.de/software/joelib

Coverage and relative width In practical applications, conditional density estimation is often used to obtain prediction intervals. Thus, it makes sense to evaluate the quality of the prediction intervals obtained from these density estimates. To this end, we use two quality measures, the empirical coverage of target values in the test data as well as the average width of the predicted intervals. The goal is to obtain narrow intervals with empirical coverage close to the chosen confidence level. For the results shown here, a 95% confidence level was chosen. Empirical coverage is expressed as the average percentage of target values in the test data that were in the range of the predicted intervals J_i:

$$EmpCov = \frac{\sum_{i=0}^{n_{test}} \delta_{iJ_i}}{n_{test}} \cdot 100, \text{with } \delta_{iJ_i} = \begin{cases} 1 & \text{if } y_i \in J_i \\ 0 & \text{else} \end{cases}$$

Interval width is expressed relative to the full range of target values in the training dataset. For instance, a relative width of 100 means that the total interval width is equal to the full range of target values in the training data, yielding no useful information. We define relative width as follows:

$$RelWid = \frac{\sum_{i=0}^{n_{test}} \sum_{j=0}^{m} (b_{i_{j+1}} - b_{i_j})}{n_{test} \cdot (max(Y_{train}) - min(Y_{train}))} \cdot 100,$$

where $b_{i_{j+1}}$ and b_{i_j} are the interval borders (the histogram and kernel estimators can produce multiple intervals) and Y_{train} is the set of training target values.

Results

The following section evaluates the experimental results and is structured as follows. In the first part the three conditional density estimators are compared with respect to the mean entropy gain and the best descriptor set is determined. In the second part we will analyze the coverage and the size of the predicted intervals. In the third part the performance of GP regression is compared to that of the conditional density estimators described above.

Descriptor Analysis and Results for Mean Entropy Gain

The main goal of this first evaluation section is to compare the different conditional density estimators on different feature sets and also to find out which descriptor set is working best. In the result table (Table 2a), the statistical significance is measured with respect to the kernel estimator.

The results show that the kernel estimator improves in a statistically significant way over the histogram estimator in all cases. In fact, the latter is outperformed by the prior estimator in nearly all cases. Although the normal estimator is not flexible (because it is only able to model an unimodal Gaussian distribution), it also performs well compared to the histogram estimator. In this context it is illustrative to consider the distribution of measured values (Figure 2). Most datasets show roughly a normal distribution (sometimes skewed or with a tendency towards a multimodal distribution (ER)). In comparison to the kernel estimator, there are statistically significant differences in about 50% of the cases. The normal estimator performs equally well or worse

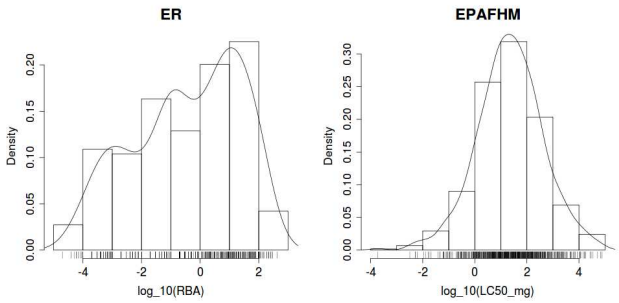

Figure 2: Distribution of the target values for two representative datasets.

than the kernel estimator, indicating that additional flexibility can be important.

In the next step we compare the three different descriptor sets. The performance is somewhat inconsistent. For the histogram estimator the JOELib2 descriptors are in most cases significantly better than the descriptor combination sets, indicating that this estimator is not able to deal with large amounts of descriptors (overfitting). However, for the kernel, but also for the normal estimator, a dominance of the closed free trees combined with JOELib2 descriptors is visible. The kernel estimator seems to be the better choice compared to the normal estimator for the combined descriptor groups. For the kernel and normal estimator, it can be stated that the combination of descriptor groups works, for most datasets, better than single descriptor groups.

Increasing the number of discretization bins from 10 to 20 leads to a significant degradation for the histogram estimator, but for the normal and kernel estimator the effect is relatively small in most cases and significant differences are never observable (results not shown here).

Results for Prediction Intervals

To evaluate the prediction intervals, we use the two quality measures from above, the empirical coverage of target values in the test data as well as the relative width of the predicted intervals. The aim is to obtain narrow intervals with empirical coverage close to the chosen confidence level. For the results shown here a 95% confidence level was chosen.

All results (Table 3) are calculated with random forests with cFTs+JOE descriptors and are evaluated by coverage and relative width. The remaining results are omitted due to lack of space. Furthermore, we present only the results for 10 discretization intervals. Results for 20 intervals yield a comparable relative performance for the kernel and the normal estimator.

The results show that the kernel as well as the normal estimator always produces a significantly higher *coverage* than the histogram estimator. Both show in all cases a coverage that is often above or near the confidence interval of 95%. This is the desired outcome, which is important for practical applications where the end user expects reliable prediction intervals. The strong performance of the kernel and normal estimator is contrasted by the failure of the histogram esti-

Dataset	RF Kernel	RF Hist.	RF Normal	GPR
BZR cFTs	0.27± 0.22	-0.71± 0.49 •	0.27± 0.18	0.15± 0.18
BZR JOE	0.41± 0.14	**−0.31±0.40** •	0.28± 0.13 •	0.18± 0.16 •
BZR comb.	**0.45±0.17**	-0.39± 0.43 •	**0.34±0.14** •	**0.23±0.20** •
COX2 cFTs	0.36± 0.22	-0.19± 0.41 •	0.17± 0.27 •	0.24± 0.18
COX2 JOE	0.56± 0.12	**0.05±0.34** •	0.35± 0.10 •	**0.40±0.14** •
COX2 comb.	**0.58±0.15**	-0.03± 0.37 •	**0.37±0.14** •	0.38± 0.15 •
DHFR cFTs	0.53± 0.21	-0.70± 0.47 •	**0.76±0.30** ○	0.74± 0.27 ○
DHFR JOE	0.56± 0.30	**−0.16±0.34** •	0.57± 0.25	0.67± 0.30
DHFR comb.	**0.65±0.21**	-0.17± 0.32 •	0.72± 0.27	**0.79±0.27**
ER cFTs	0.30± 0.26	-0.65± 0.57 •	0.41± 0.19	0.39± 0.14
ER JOE	**0.65±0.17**	**0.09±0.36** •	0.51± 0.10 •	0.52± 0.21
ER comb.	0.61± 0.20	-0.07± 0.41 •	**0.56±0.13**	**0.70±0.19**
CPDB_mouse cFTs	0.13± 0.22	-1.11± 0.55 •	0.24± 0.15	0.14± 0.21
CPDB_mouse JOE	0.19± 0.10	**−0.54±0.37** •	0.21± 0.19	**0.20±0.25**
CPDB_mouse comb.	**0.25±0.16**	-0.58± 0.43 •	**0.26±0.16**	**0.20±0.20**
CPDB_rat cFTs	0.28± 0.15	-0.73± 0.41 •	0.33± 0.11	0.28± 0.16
CPDB_rat JOE	0.24± 0.09	-0.37± 0.28 •	0.22± 0.07	0.25± 0.15
CPDB_rat comb.	**0.35±0.11**	**−0.33±0.33** •	**0.34±0.08**	**0.30±0.14**
EPAFHM cFTs	0.47± 0.15	-0.57± 0.42 •	0.49± 0.12	0.54± 0.18
EPAFHM JOE	**0.69±0.15**	**0.11±0.36** •	**0.60±0.13** •	0.64± 0.25
EPAFHM comb.	0.68± 0.14	0.05± 0.34 •	0.59± 0.12 •	**0.75±0.26**
FDAMDD cFTs	0.49± 0.11	-0.32± 0.29 •	**0.37±0.08** •	0.27± 0.09 •
FDAMDD JOE	0.43± 0.08	**−0.02±0.19** •	0.27± 0.07 •	0.27± 0.09 •
FDAMDD comb.	**0.52±0.09**	-0.07± 0.24 •	0.35± 0.07 •	**0.28±0.12** •

○, • statistically significant improvement or degradation

(a)

Dataset	RF	GPR
BZR cFTs	7.49±0.20	301.32± 37.20 •
BZR JOE	2.80±0.04	200.10± 18.70 •
BZR comb.	6.45±0.10	372.87± 50.55 •
COX2 cFTs	9.54±0.26	530.62± 65.39 •
COX2 JOE	3.51±0.05	402.93± 33.26 •
COX2 comb.	8.42±0.13	628.83± 84.60 •
DHFR cFTs	16.76±0.32	1939.98± 181.66 •
DHFR JOE	5.91±0.08	1434.38± 113.72 •
DHFR comb.	12.85±0.16	2179.09± 224.18 •
ER cFTs	19.85±0.48	1890.43± 187.80 •
ER JOE	3.10±0.43	324.63± 53.35 •
ER comb.	14.86±0.26	2140.66± 226.27 •
CPDB_mouse cFTs	18.54±0.30	1158.29± 31.15 •
CPDB_mouse JOE	4.24±0.06	437.31± 20.23 •
CPDB_mouse comb.	14.38±0.17	1295.11± 34.37 •
CPDB_rat cFTs	25.50±0.41	2321.26± 57.94 •
CPDB_rat JOE	5.45±0.08	884.70± 43.78 •
CPDB_rat comb.	19.29±0.21	2532.12± 88.24 •
EPAFHM cFTs	25.94±0.40	2486.02± 97.88 •
EPAFHM JOE	4.67±0.07	932.37± 50.16 •
EPAFHM comb.	19.02±0.20	2765.35± 113.96 •
FDAMDD cFTs	66.96±0.93	16928.35±1486.98 •
FDAMDD JOE	11.50±0.18	8160.61± 666.56 •
FDAMDD comb.	56.15±0.77	15753.52± 707.52 •

○, • statistically significant improvement or degradation

(b)

Table 2: Results with standard deviations. (a) Mean improvement in log-likelihood vs prior estimate, using random forests and 10 bins. The best feature set for a conditional density estimator and a dataset is marked in bold. (b) Training times in seconds.

mator which is unable to get close to the confidence interval of 95%. Regarding the *relative width* of the prediction intervals, the histogram estimator shows narrower (better) interval widths than the kernel and/or normal estimator in three of the eight cases. This is due to its low coverage and explained by the fact that the kernel and the normal estimator perform smoothing.

Generally, the end user requires reliable prediction intervals, so that more emphasis should be given to coverage than to relative width. But particular attention must be paid to those cases where one method dominates another one based on both criteria. For the kernel and normal estimator vs the histogram estimator, this is the case for five of eight datasets. The opposite is never the case. This confirms the strong performance of the kernel and normal estimators in comparison to the histogram-based method.

Considering the relative performance of the normal and kernel estimators for interval estimation, we can see that they are largely competitive. The former generally exhibits elevated coverage levels and larger intervals. There are two cases where the former dominates the latter with respect two both criteria (the CPDB datasets). Vice versa, there is one dataset (COX2) where the kernel estimators performs better according to both criteria.

Comparison to Gaussian Process Regression

In the following, we present our experimental comparison with Gaussian Process regression on the eight datasets. Gaussian Process regression was applied with inputs normalized to $[0, 1]$. The target was also normalized, to make choosing an appropriate noise level σ easier. Predictions were transformed back into the original range of the target variable. The noise level σ for GPR, and γ for the RBF kernel, were optimized using internal cross-validation, based on optimizing for mean entropy gain, with values 10^i with $i \in \{-5, 4, ..., 4, 5\}$ for γ, and values 10^i with $i \in \{-2, -1.8, -1.6, ..., -0.4, -0.2, 0\}$ for σ. To save runtime, 5-fold cross-validation was used to perform an initial complete exploration of all pairs of parameter values. Subsequently, greedy exploration based on 10-fold cross-validation was performed, starting with the best pair found in the initial complete run.

We performed experiments with linear kernels and RBF kernels. As the results using linear kernels were worse than all other results presented here, we focus on the results for RBF kernels. In Tables 2a and 3, the performance of Gaussian Process regression using RBF kernels is given in the last column for each quantity (mean entropy gain, coverage and relative width). As can be seen, the performance of GP regression is generally on par or only slightly worse than the one of the kernel estimator. However, looking at the running times in Table 2b, it is clear that GP regression requires substantially more time for training (sometimes up to a factor of 100 or more), because parameter optimization is necessary to reach competitive levels of performance.

	Coverage				Relative width			
Dataset	RF Kernel	RF Hist.	RF Normal	GPR	RF Kernel	RF Hist.	RF Normal	GPR
BZR	95.79± 3.69	85.42±6.48 •	**97.93±2.32** ○	93.70±3.82	59.92±3.05	**58.85±2.58**	61.70± 3.30 •	66.16± 4.39 •
COX2	**95.68±2.76**	86.76±4.80 •	95.26± 3.14	94.13±3.37	69.82±1.93	**61.00±1.94** ○	70.69± 1.98 •	67.38± 0.94 ○
DHFR	96.30± 2.12	90.97±3.40 •	**97.86±1.72** ○	95.75±2.72	35.85±3.20	47.50± 1.67 •	35.97± 3.07	**30.35±4.09** ○
ER	95.90± 2.94	86.78±5.74 •	**97.29±2.77**	92.59±3.95 •	66.27±2.13	59.62± 2.12 ○	68.59± 2.24 •	**47.45±1.41** ○
CPDB_mouse	95.11± 3.56	88.37±5.11 •	**95.27±3.35**	94.14±4.36	49.89±2.51	64.34± 2.55 •	**48.80±2.31** ○	49.99± 4.29
CPDB_rat	95.83± 2.69	90.41±3.93 •	**96.60±2.36**	95.17±3.21	52.32±2.14	62.34± 2.05 •	51.05± 2.06 ○	**50.77±1.63**
EPAFHM	96.81± 2.20	91.97±3.54 •	**97.03±2.12**	94.57±3.11	51.72±2.59	58.90± 1.55 •	48.16± 2.37 ○	**35.72±4.69** ○
FDAMDD	96.03± 1.71	89.92±2.94 •	**96.76±1.42** ○	92.26±2.35 •	46.27±0.61	56.19± 1.50 •	47.38± 0.64 •	**41.29±1.19** ○

○/● statistically significant improvement/degradation; best density estimator per dataset in bold print

Table 3: Quality of prediction intervals with standard deviations, using random forests and 10 bins and Gaussian Process regression, with cFTs combined with JOE descriptors. Best density estimator per dataset and criterion in bold print.

Conclusion

In this paper we considered methods for conditional density estimation based on generic machine learning methods, more precisely, class probability estimators, for their use in QSAR modeling. Our experimental results show that kernel density estimators based on weighted target values are the best choice for a wide range of target distributions, in particular with respect to the log-likelihood. If the target values for the dataset are (conditionally) normally or roughly normally distributed, the normal estimator is also a good choice. The latter appears to work particularly well for the prediction of intervals. The use of the histogram estimator should rather be avoided, as its lack of smoothing generally leads to sharp discontinuities between the bins.

We also compared with Gaussian Process (GP) regression, one of the standard methods for conditional density estimation. GP regression with a linear kernel failed to produce competitive results, whereas GP regression with an RBF kernel produced results on par with the best obtained using class probability estimation with random forests. Due to the required parameter optimization, however, the running times exceeded those of the method presented here by a large factor. Therefore, conditional density estimation based on class probability estimation, based on using kernel and normal estimators in conjunction with random forests can be seen as good options for QSAR problems. Finally, it is interesting to note that the approach considered here can be applied to take into account costs at prediction time as the full conditional density could be taken into account for this purpose.

Acknowledgements

This work was partially supported by the EU FP7 project (HEALTH-F5-2008-200787) OpenTox (http://www.opentox.org).

References

Breiman, L. 2001. Random forests. *Machine Learning* 45(1):5–32.

Frank, E., and Bouckaert, R. 2009. Conditional density estimation with class probability estimators. In *Proc. of ACML '09*. Springer.

Frank, E., and Hall, M. 2001. A simple approach to ordinal classification. In *Proc. of ECML '01*, 145–156. Springer.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I., H. 2009. The WEKA data mining software: An update. *SIGKDD Explorations* 11(1).

Matthews, E.; Kruhlak, N.; Benz, R.; and Contrera, J. 2004. Assessment of the health effects of chemicals in humans: I. QSAR estimation of the maximum recommended therapeutic dose (MRTD) and no effect level (NOEL) of organic chemicals based on clinical trial data. *Current Drug Discovery Technologies* 1(1):61–76.

Nadeau, C., and Bengio, Y. 2003. Inference for the generalization error. *Machine Learning* 52(3):239–281.

Rasmussen, C. E., and Williams, C. K. I. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. MIT Press.

Rückert, U., and Kramer, S. 2004. Frequent Free Tree Discovery in Graph Data. In *Proc. of SAC '04*, 564–570. ACM Press.

Russom, C.; Bradbury, S.; Broderius, D.; Hammermeister, S.; and Drummond, R. 1997. Predicting modes of action from chemical structure: Acute toxicity in the fathead minnow (pimephales promelas). *Environmental Toxicology and Chemistry* 5(16):948–967.

Schapire, R. E.; Stone, P.; McAllester, D. A.; Littman, M. L.; and Csirik, J. A. 2002. Modeling auction price uncertainty using boosting-based conditional density estimation. In *Proc. of ICML '01*, 546–553. Morgan Kaufmann.

Schwaighofer, A.; Schroeter, T.; Mika, S.; Laub, J.; ter Laak, A.; Sülze, D.; Ganzer, U.; Heinrich, N.; and Müller, K.-R. 2007. Accurate solubility prediction with error bars for electrolytes: A machine learning approach. *J. Chem. Inf. Model.* 47:407–424.

Sutherland, J. J.; O'Brien, L. A.; and Weaver, D. F. 2003. Spline-fitting with a genetic algorithm: A method for developing classification structure-activity relationships. *J. Chem. Inf. Model.* 43(6):1906–1915.

Sutherland, J. J.; O'Brien, L. A.; and Weaver, D. F. 2004. A comparison of methods for modeling quantitative structure-activity relationships. *J. Med. Chem.* 47(22):5541–5554.