A comparison of random forest, support vector machine, deep learning and lazar algorithms for predicting mutagenicity

Christoph Helma[*1], Verena Schöning[2], Philipp Boss[2], and Jürgen Drewe[2]

[1]in silico toxicology gmbh, Rastatterstrasse 41, 4057 Basel, Switzerland
[2]Zeller AG, Seeblickstrasse 4, 8590 Romanshorn, Switzerland
[*] Correspondence: Christoph Helma <helma@in-silico.ch>

k-nearest neighbor (`lazar`), random forest, support vector machine and deep learning algorithms were applied to a new *Salmonella* mutagenicity dataset with 8281 unique chemical structures. Algorithm performance was evaluated using 5-fold crossvalidation. TODO - results - conclusion

Introduction

TODO: algo history

TODO: dataset history

TODO: open problems

Materials and Methods

Mutagenicity data

For all methods, the same training dataset was used. The training dataset was compiled from the following sources:

- Kazius/Bursi Dataset (4337 compounds, Kazius, McGuire, and Bursi (2005)): http://cheminformatics.org/datasets/bursi/cas_4337.zip
- Hansen Dataset (6513 compounds, Hansen et al. (2009)): http://doc.ml.tu-berlin.de/toxbenchmark/Mutagenicity_N6512.csv
- EFSA Dataset (695 compounds): https://data.europa.eu/euodp/data/storage/f/2017-0719T142131/GENOTOX%20data%20and%20dictionary.xls

Mutagenicity classifications from Kazius and Hansen datasets were used without further processing. To achieve consistency between these datasets, EFSA compounds were classified as mutagenic, if at least one positive result was found for TA98 or T100 Salmonella strains.

Dataset merges were based on unique SMILES (*Simplified Molecular Input Line Entry Specification*) strings of the compound structures. Duplicated experimental data with the same outcome was merged into a single value, because it is likely that it originated from the same experiment. Contradictory results were kept as multiple measurements in the database. The combined training dataset contains 8281 unique structures.

Source code for all data download, extraction and merge operations is publicly available from the git repository https://git.in-silico.ch/pyrrolizidine under a GPL3 License.

TODO: check/fix git repo

For the Random Forest (RF), Support Vector Machines (SVM), and Deep Learning (DL) models, molecular descriptors were calculated with the PaDEL-Descriptors program (http://www.yapcwsoft.com version 2.21, Yap (2011)).

TODO: sentence ??

From these descriptors were chosen, which were actually used for the generation of the DL model.

Algorithms

`lazar`

`lazar` (*lazy structure activity relationships*) is a modular framework for read-across model development and validation. It follows the following basic workflow: For a given chemical structure `lazar`:

- searches in a database for similar structures (neighbours) with experimental data,
- builds a local QSAR model with these neighbours and
- uses this model to predict the unknown activity of the query compound.

This procedure resembles an automated version of read across predictions in toxicology, in machine learning terms it would be classified as a k-nearest-neighbour algorithm.

Apart from this basic workflow, `lazar` is completely modular and allows the researcher to use any algorithm for similarity searches and local QSAR (*Quantitative structure–activity relationship*) modelling. Algorithms used within this study are described in the following sections.

Neighbour identification

Similarity calculations were based on MolPrint2D fingerprints (Bender et al. (2004)) from the OpenBabel cheminformatics library (O'Boyle et al. (2011)). The MolPrint2D fingerprint uses atom environments as molecular representation, which resembles basically the chemical concept of functional groups. For each atom in a molecule, it represents the chemical environment using the atom types of connected atoms.

MolPrint2D fingerprints are generated dynamically from chemical structures and do not rely on predefined lists of fragments (such as OpenBabel FP3, FP4 or MACCs fingerprints or lists of toxicophores/toxicophobes). This has the advantage that they may capture substructures of toxicological relevance that are not included in other fingerprints.

From MolPrint2D fingerprints a feature vector with all atom environments of a compound can be constructed that can be used to calculate chemical similarities.

The chemical similarity between two compounds a and b is expressed as the proportion between atom environments common in both structures A B and the total number of atom environments A U B (Jaccard/Tanimoto index).

$$sim = \frac{|A \cap B|}{|A \cup B|}$$

Threshold selection is a trade-off between prediction accuracy (high threshold) and the number of predictable compounds (low threshold). As it is in many practical cases desirable to make predictions even in the absence of closely related neighbours, we follow a tiered approach:

- First a similarity threshold of 0.5 is used to collect neighbours, to create a local QSAR model and to make a prediction for the query compound.
- If any of these steps fails, the procedure is repeated with a similarity threshold of 0.2 and the prediction is flagged with a warning that it might be out of the applicability domain of the training data.
- Similarity thresholds of 0.5 and 0.2 are the default values chosen > by the software developers and remained unchanged during the > course of these experiments.

Compounds with the same structure as the query structure are automatically eliminated from neighbours to obtain unbiased predictions in the presence of duplicates.

Local QSAR models and predictions

Only similar compounds (neighbours) above the threshold are used for local QSAR models. In this investigation, we are using a weighted majority vote from the neighbour's experimental data for mutagenicity classifications. Probabilities for both classes (mutagenic/non-mutagenic) are calculated according to the following formula and the class with the higher probability is used as prediction outcome.

$$p_c = \frac{\sum \text{sim}_{n,c}}{\sum \text{sim}_n}$$

p_c Probability of class c (e.g. mutagenic or non-mutagenic)
$\sum \text{sim}_{n,c}$ Sum of similarities of neighbours with class c
$\sum \text{sim}_n$ Sum of all neighbours

Applicability domain

The applicability domain (AD) of `lazar` models is determined by the structural diversity of the training data. If no similar compounds are found in the training data no predictions will be generated. Warnings are issued if the similarity threshold had to be lowered from 0.5 to 0.2 in order to enable predictions. Predictions without warnings can be considered as close to the applicability domain and predictions with warnings as more distant from the applicability domain. Quantitative applicability domain information can be obtained from the similarities of individual neighbours.

Availability

- `lazar` experiments for this manuscript: https://git.in-silico.ch/pyrrolizidine (source code, GPL3)
- `lazar` framework: https://git.in-silico.ch/lazar (source code, GPL3)
- `lazar` GUI: https://git.in-silico.ch/lazar-gui (source code, GPL3)
- Public web interface: https://lazar.in-silico.ch

Random Forest, Support Vector Machines, and Deep Learning in R-project

In comparison to `lazar`, three other models (Random Forest (RF), Support Vector Machines (SVM), and Deep Learning (DL)) were evaluated.

For the generation of these models, molecular 1D and 2D descriptors of the training dataset were calculated using PaDEL-Descriptors (http://www.yapcwsoft.com version 2.21, Yap (2011)).

As the training dataset contained over 8280 instances, it was decided to delete instances with missing values during data pre-processing. Furthermore, substances with equivocal outcome were removed. The final training dataset contained 8080 instances with known mutagenic potential. The RF, SVM, and DL models were generated using the R software (R-project for Statistical Computing, https://www.r-project.org/*;* version 3.3.1), specific R packages used are identified for each step in the description below. During feature selection, descriptor with near zero variance were removed using '*NearZeroVar*'-function (package 'caret'). If the percentage of the most common value was more than

90% or when the frequency ratio of the most common value to the second most common value was greater than 95:5 (e.g. 95 instances of the most common value and only 5 or less instances of the second most common value), a descriptor was classified as having a near zero variance. After that, highly correlated descriptors were removed using the '*findCorrelation*'-function (package 'caret') with a cut-off of 0.9. This resulted in a training dataset with 516 descriptors. These descriptors were scaled to be in the range between 0 and 1 using the '*preProcess*'-function (package 'caret'). The scaling routine was saved in order to apply the same scaling on the testing dataset. As these three steps did not consider the outcome, it was decided that they do not need to be included in the cross-validation of the model. To further reduce the number of features, a LASSO (*least absolute shrinkage and selection operator*) regression was performed using the '*glmnet*'-function (package '*glmnet*'). The reduced dataset was used for the generation of the pre-trained models.

For the RF model, the '*randomForest*'-function (package '*randomForest*') was used. A forest with 1000 trees with maximal terminal nodes of 200 was grown for the prediction.

The '*svm*'-function (package 'e1071') with a *radial basis function kernel* was used for the SVM model.

The DL model was generated using the '*h2o.deeplearning*'-function (package '*h2o*'). The DL contained four hidden layer with 70, 50, 50, and 10 neurons, respectively. Other hyperparameter were set as follows: l1=1.0E-7, l2=1.0E-11, epsilon = 1.0E-10, rho = 0.8, and quantile_alpha = 0.5. For all other hyperparameter, the default values were used. Weights and biases were in a first step determined with an unsupervised DL model. These values were then used for the actual, supervised DL model.

To validate these models, an internal cross-validation approach was chosen. The training dataset was randomly split in training data, which contained 95% of the data, and validation data, which contain 5% of the data. A feature selection with LASSO on the training data was performed, reducing the number of descriptors to approximately 100. This step was repeated five times. Based on each of the five different training data, the predictive models were trained and the performance tested with the validation data. This step was repeated 10 times. Furthermore, a y-randomisation using the RF model was performed. During y-randomisation, the outcome (y-variable) is randomly permuted. The theory is that after randomisation of the outcome, the model should not be able to correlate the outcome to the properties (descriptor values) of the substances. The performance of the model should therefore indicate a by change prediction with an accuracy of about 50%. If this is true, it can be concluded that correlation between actual outcome and properties of the substances is real and not by chance (Rücker, Rücker, and Meringer (2007)).

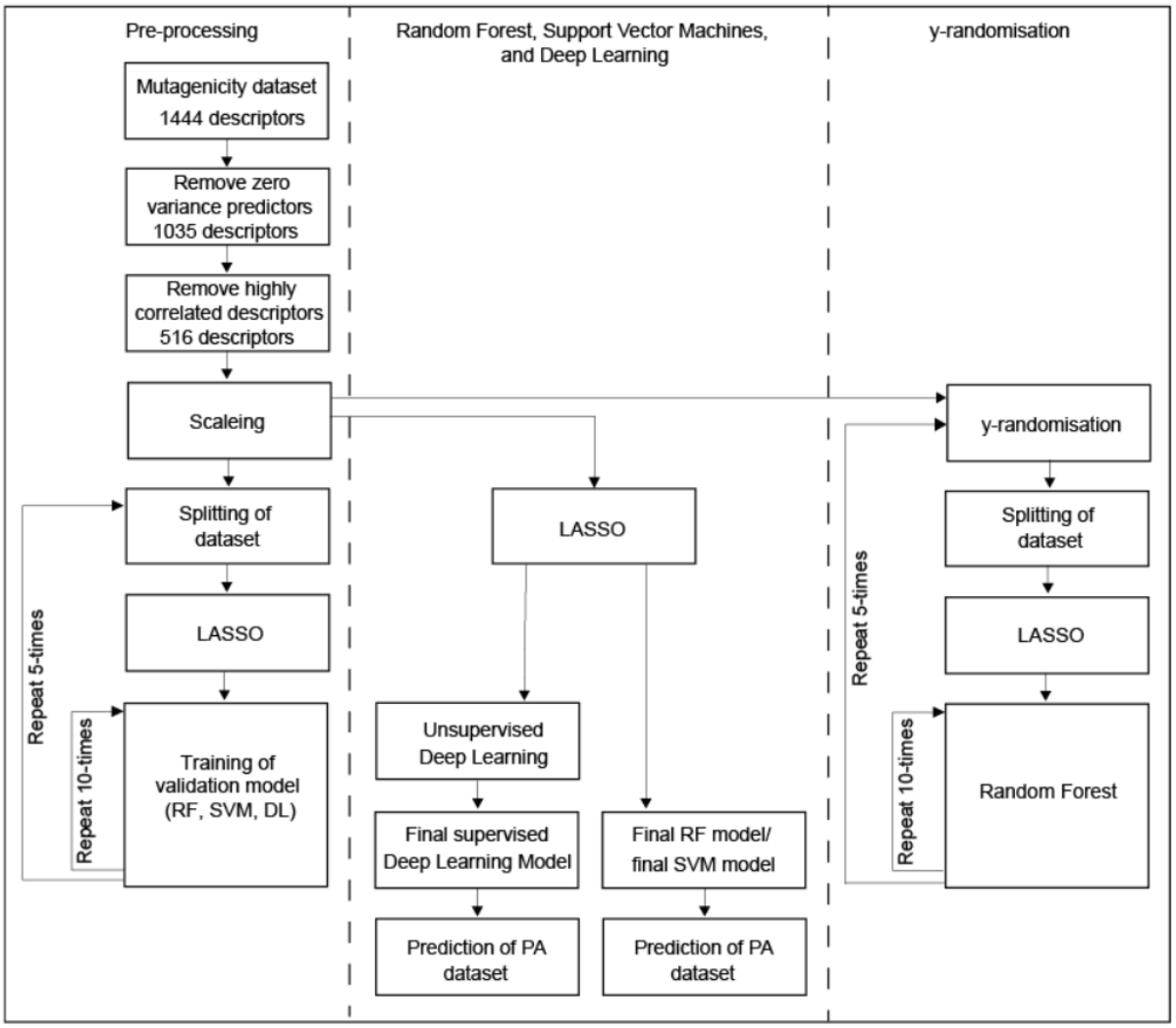

Figure 1: Flowchart of the generation and validation of the models generated in R-project

Applicability domain

The AD of the training dataset and the PA dataset was evaluated using the Jaccard distance. A Jaccard distance of '0' indicates that the substances are similar, whereas a value of '1' shows that the substances are different. The Jaccard distance was below 0.2 for all PAs relative to the training dataset. Therefore, PA dataset is within the AD of the training dataset and the models can be used to predict the genotoxic potential of the PA dataset.

y-randomisation

After y-randomisation of the outcome, the accuracy and CCR are around 50%, indicating a chance in the distribution of the results. This shows, that the outcome is actually related to the predictors and not by chance.

Deep Learning in TensorFlow

Alternatively, a DL model was established with Python-based TensorFlow program (https://www.tensorflow.org/) using the high-level API Keras (https://www.tensorflow.org/guide/keras) to build the models.

Data pre-processing was done by rank transformation using the '*QuantileTransformer*' procedure. A sequential model has been used. Four layers have been used: input layer, two hidden layers (with 12, 8 and 8 nodes, respectively) and one output layer. For the output layer, a sigmoidal activation function and for all other layers the ReLU ('*Rectified Linear Unit*') activation function was used. Additionally, a L^2-penalty of 0.001 was used for the input layer. For training of the model, the ADAM algorithm was used to minimise the cross-entropy loss using the default parameters of Keras. Training was performed for 100 epochs with a batch size of 64. The model was implemented with Python 3.6 and Keras. For training of the model, a 6-fold cross-validation was used. Accuracy was estimated by ROC-AUC and confusion matrix.

Validation

Results

`lazar`

Random Forest

The validation showed that the RF model has an accuracy of 64%, a sensitivity of 66% and a specificity of 63%. The confusion matrix of the model, calculated for 8080 instances, is provided in Table 1.

Table 1: Confusion matrix of the RF model

		Predicted genotoxicity		
Measured genotoxicity		***PP***	***PN***	***Total***
	TP	2274	1163	3437
	TN	1736	2907	4643
	Total	4010	4070	8080

PP: Predicted positive; PN: Predicted negative, TP: True positive, TN: True negative

Support Vector Machines

The validation showed that the SVM model has an accuracy of 62%, a sensitivity of 65% and a specificity of 60%. The confusion matrix of SVM model, calculated for 8080 instances, is provided in Table 2.

Table 2: Confusion matrix of the SVM model

	Predicted genotoxicity		
Measured genotoxicity	***PP***	***PN***	***Total***
TP	2057	1107	3164
TN	1953	2963	4916
Total	4010	4070	8080

PP: Predicted positive; PN: Predicted negative, TP: True positive, TN: True negative

Deep Learning (R-project)

The validation showed that the DL model generated in R has an accuracy of 59%, a sensitivity of 89% and a specificity of 30%. The confusion matrix of the model, normalised to 8080 instances, is provided in Table 3.

Table 3: Confusion matrix of the DL model (R-project)

	Predicted genotoxicity		
Measured genotoxicity	***PP***	***PN***	***Total***
TP	3575	435	4010
TN	2853	1217	4070
Total	6428	1652	8080

PP: Predicted positive; PN: Predicted negative, TP: True positive, TN: True negative

DL model (TensorFlow)

The validation showed that the DL model generated in TensorFlow has an accuracy of 68%, a sensitivity of 70% and a specificity of 46%. The confusion matrix of the model, normalised to 8080 instances, is provided in Table 4.

Table 4: Confusion matrix of the DL model (TensorFlow)

	Predicted genotoxicity		
Measured genotoxicity	***PP***	***PN***	***Total***

	Predicted genotoxicity		
TP	2851	1227	4078
TN	1825	2177	4002
Total	4676	3404	8080

PP: Predicted positive; PN: Predicted negative, TP: True positive, TN: True negative

The ROC curves from the 6-fold validation are shown in Figure 7.

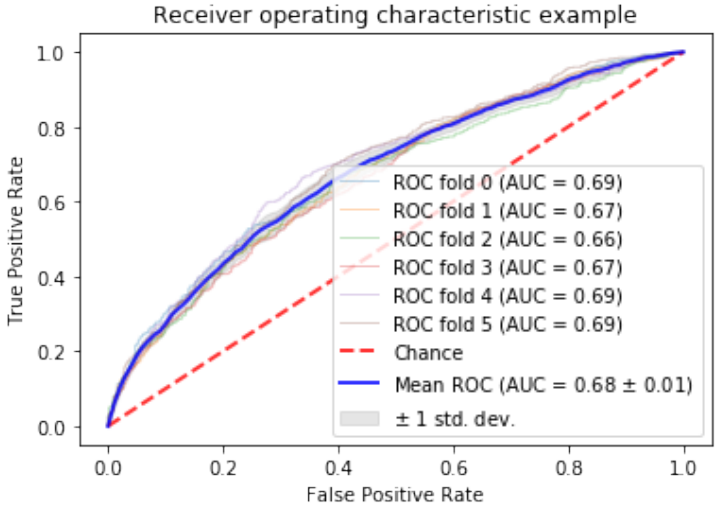

Figure 7: Six-fold cross-validation of TensorFlow DL model show an average area under the ROC-curve (ROC-AUC; measure of accuracy) of 68%.

In summary, the validation results of the four methods are presented in the following table.

Table 5 Results of the cross-validation of the four models and after y-randomisation

	Accuracy	CCR	Sensitivity	Specificity
RF model	64.1%	64.4%	66.2%	62.6%
SVM model	62.1%	62.6%	65.0%	60.3%
DL model (R-project)	59.3%	59.5%	89.2%	29.9%
DL model (TensorFlow)	68%	62.2%	69.9%	45.6%
y-randomisation	50.5%	50.4%	50.3%	50.6%

CCR (correct classification rate)

Discussion

General model performance

Based on the results of the cross-validation for all models, `lazar`, RF, SVM, DL (R-project) and DL (TensorFlow) it can be state that the prediction results are not optimal due to different reasons. The accuracy as measured during cross-validation of the four models (RF, SVM, DL (R-project and TensorFlow)) was partly low with CCR values between 59.3 and 68%, with the R-generated DL model and the TensorFlow-generated DL model showing the worst and the best performance, respectively. The validation of the R-generated DL model revealed a high sensitivity (89.2%) but an unacceptably low specificity of 29.9% indicating a high number of false positive estimates. The TensorFlow-generated DL model, however, showed an acceptable but not optimal accuracy of 68%, a sensitivity of 69.9% and a specificity of 45.6%. The low specificity indicates that both DL models tends to predict too many instances as positive (genotoxic), and therefore have a high false positive rate. This allows at least with the TensorFlow generated DL model to make group statements, but the confidence for estimations of single PAs appears to be insufficiently low.

Several factors have likely contributed to the low to moderate performance of the used methods as shown during the cross-validation:

1. The outcome in the training dataset was based on the results of AMES tests for genotoxicity ICH 2011(), an *in vitro* test in different strains of the bacteria *Salmonella typhimurium*. In this test, mutagenicity is evaluated with and without prior metabolic activation of the test substance. Metabolic activation could result in the formation of genotoxic metabolites from non-genotoxic parent compounds. However, no distinction was made in the training dataset between substances that needed metabolic activation before being mutagenic and those that were mutagenic without metabolic activation. `lazar` is able to handle this 'inaccuracy' in the training dataset well due to the way the algorithm works: `lazar` predicts the genotoxic potential based on the neighbours of substances with comparable structural features, considering mutagenic and not mutagenic neighbours. Based on the structural similarity, a probability for mutagenicity and no mutagenicity is calculated independently from each other (meaning that the sum of probabilities does not necessarily adds up to 100%). The class with the higher outcome is then the overall outcome for the substance.

   In contrast, the other models need to be trained first to recognise the structural features that are responsible for genotoxicity. Therefore, the mixture of substances being mutagenic with and without metabolic activation in the training dataset may have adversely affected the ability to separate the dataset in two distinct classes and thus explains the relatively low performance of these models.

2. Machine learning algorithms try to find an optimized solution in a high-dimensional

(one dimension per each predictor) space. Sometimes these methods do not find the global optimum of estimates but only local (not optimal) solutions. Strategies to find the global solutions are systematic variation (grid search) of the hyperparameters of the methods, which may be very time consuming in particular in large datasets.

Conclusions

In this study, an attempt was made to predict the genotoxic potential of PAs using five different machine learning techniques (`lazar`, RF, SVM, DL (R-project and TensorFlow). The results of all models fitted only partly to the findings in literature, with best results obtained with the TensorFlow DL model. Therefore, modelling allows statements on the relative risks of genotoxicity of the different PA groups. Individual predictions for selective PAs appear, however, not reliable on the current basis of the used training dataset.

This study emphasises the importance of critical assessment of predictions by QSAR models. This includes not only extensive literature research to assess the plausibility of the predictions, but also a good knowledge of the metabolism of the test substances and understanding for possible mechanisms of toxicity.

In further studies, additional machine learning techniques or a modified (extended) training dataset should be used for an additional attempt to predict the genotoxic potential of PAs.

References

Bender, Andreas, Hamse Y. Mussa, Robert C. Glen, and Stephan Reiling. 2004. "Molecular Similarity Searching Using Atom Environments, Information-Based Feature Selection, and a Naïve Bayesian Classifier." *Journal of Chemical Information and Computer Sciences* 44 (1): 170–78. https://doi.org/10.1021/ci034207y.

Hansen, Katja, Sebastian Mika, Timon Schroeter, Andreas Sutter, Antonius ter Laak, Thomas Steger-Hartmann, Nikolaus Heinrich, and Klaus-Robert Müller. 2009. "Benchmark Data Set for in Silico Prediction of Ames Mutagenicity." *Journal of Chemical Information and Modeling* 49 (9): 2077–81. https://doi.org/10.1021/ci900161g.

Kazius, J., R. McGuire, and R. Bursi. 2005. "Derivation and Validation of Toxicophores for Mutagenicity Prediction." *J Med Chem*, no. 48: 312–20.

O'Boyle, Noel, Michael Banck, Craig James, Chris Morley, Tim Vandermeersch, and Geoffrey Hutchison. 2011. "Open Babel: An open chemical toolbox." *J. Cheminf.* 3 (1): 33. https://doi.org/doi:10.1186/1758-2946-3-33.

Rücker, C, G Rücker, and M. Meringer. 2007. "Y-Randomization and Its Variants in Qspr/Qsar." *J. Chem. Inf. Model.*, no. 47: 2345–57.

Yap, CW. 2011. "PaDEL-Descriptor: An Open Source Software to Calculate Molecular Descriptors and Fingerprints." *Journal of Computational Chemistry*, no. 32: 1466–74.