USER MANUAL

QSAR-Mx
(Version 1.0.0)

LAQV@REQUIMTE
Department of Biochemistry and Chemistry
FACULDADE DE CIÊNCIAS DA
UNIVERSIDADE DO PORTO
Rua do Campo Alegre, s/n, 4169-007
Porto, Portugal
www.fc.up.pt

Overview of the tool

QSAR-Mx is specifically designed with the purpose of performing the quantitative structure activity relationship modelling (QSAR) with the mixtures. Nevertheless, this software may also be used for the development of linear multiple linear regression (MLR) models for datasets containing single molecules. The main objectives of the first version of this tool are to provide the following options:

(a) Calculations of mixture weighted descriptors for the mixtures from the descriptors of their respective components.

(b) Development of multiple data-distributions based on 'Points-out', 'Mixtures-out' and 'Compounds out validation strategy' and subsequent development of the linear models based on these divisions and based on the performance of the QSAR models.

(c) Linear model development using sequential forward selection (SFS) based MLR using different model development options.

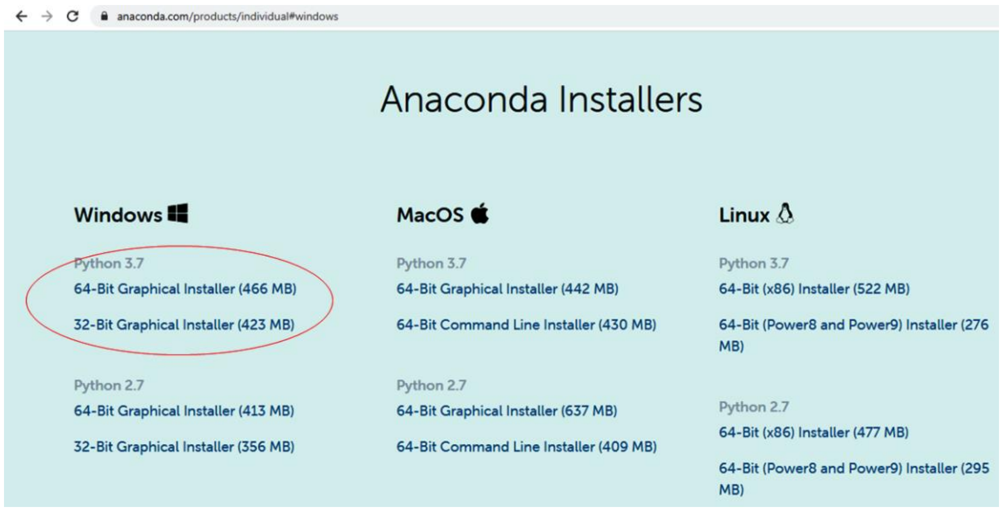
(d) Additionally, this tool helps in the determination of applicability domain of the models, in the assessment of inter-collinearity of the model' descriptors and in performing the Y-randomization method for testing robustness of the linear models.

(e) Additionally, the tool is also capable of performing multi-objective optimization (MOOP) to optimize desired and undesired properties in order to obtain global desirability scores.

QSAR-Mx is a Python 3-based[1] graphical user interface (GUI) that has multiple dependencies on python-based libraries. The installation of the dependencies in Windows platform is described below.

Installation of dependencies of *SFS-QSAR-tool*:

1. *SFS-QSAR-tool*is a python-based[1] tool that has multiple dependencies[2-5]. Therefore, the users should install anaconda (https://www.anaconda.com/products/individual#windows) with python-3.



anaconda.com/products/individual#windows

Anaconda Installers

Windows

Python 3.7
64-Bit Graphical Installer (466 MB)
32-Bit Graphical Installer (423 MB)

Python 2.7
64-Bit Graphical Installer (413 MB)
32-Bit Graphical Installer (356 MB)

MacOS

Python 3.7
64-Bit Graphical Installer (442 MB)
64-Bit Command Line Installer (430 MB)

Python 2.7
64-Bit Graphical Installer (637 MB)
64-Bit Command Line Installer (409 MB)

Linux

Python 3.7
64-Bit (x86) Installer (522 MB)
64-Bit (Power8 and Power9) Installer (276 MB)

Python 2.7
64-Bit (x86) Installer (477 MB)
64-Bit (Power8 and Power9) Installer (295 MB)

2. Additionally, the user needs to install some dependencies (see below). For this, open anaconda and type **'pip install -r requirements.txt'**.

3. If the *Mlxtend* or any other dependency fails to install, these may also be installed manually from Anaconda command prompt. As an example, the *Mlxtend* may be install by typing the command 'conda install -c conda-forge mlxtend' in Anaconda.

Functionalities of the *QSAR-Mx*

In the current instruction manual, we provided the instructions step-by-step with an example so that the user may easily understand the purpose of this tool, can easily utilize this tool and employ this tool for their own research.

Step 1- Preparation of input dataset: Along with this tool we provide a dataset named 'viscosity_dataset.csv'. The purpose is to develop QSAR model with viscosities of 73 deep eutectic solvents (DESs), which are binary mixtures of two components – hydrogen bond acceptor (HBA or salt) and hydrogen bond donor (HBD)[6]. All input data should be carefully prepared as described below.

- Column 1: containing the serial number
- Column 2 and 3: Names of the component 1 (HBA in this case) and component 2 (HBD in this case).

- Column 4 and 5: Molar fractions of two components (first HBA then HBD)
- Columns 6-8: The users must mention the dependent property or response variable, which is mentioned in the Column 8 as ‘logViscosity’. However, one may have additional properties that should be used without any variation (fixed properties). In the columns 6 and 7 we mentioned two ‘fixed independent properties’ that account for the presence or absence of chloride and bromide atoms in the HBA. These are the indicator parameters which we do not intend to modify similar to the response variable. The users may mention as many as such properties (such as temperature/pressure at which the response variable recorder). However, these should be placed after Column 5 and before the response variables. Keep in mind that these properties fixed independent properties will be treated as independent properties during model development along with mixture descriptors.
- Column 9 onwards: Here the user must provide descriptors of the components. First all descriptors of component 1 and then all descriptors of component 2. The number of descriptors for each component should be same; otherwise the tool will fail to operate. In the current dataset, the descriptors are calculated with Dragon software[7]. The user may calculate the starting descriptors using any commercial or non-commercial software.

Before processing to its next steps, two concepts are important to understand the utilities of this program.

➢ ***Calculation of mixture weighted descriptors:***

Our tool provides two different methods of descriptor calculations and these are implemented as Method 1 and Method 2. Both Method1 and Method2 based descriptor calculations were suggested by Oprisue et al[8] and such descriptor calculation strategies were previously adapted in the following published works[9, 10].

Method 1: Sum of weighted (by molar fraction) descriptors. Final descriptors (MD_{pmix}) are calculated as:

$$MD_{pmix} = x_1D_1 + x_2D_2 \qquad (1)$$

Method 2: Sum and absolute difference of weighted (by molar fraction) descriptors. Final descriptors (*Dpmix*and *Dnmix*) are calculated as:

$$MD_{pmix} = x_1D_1 + x_2D_2$$
$$MD_{nmix} = |x_1D_1 - x_2D_2| \qquad (2)$$

Therefore Method 2 calculates larger number of descriptors as compared to Method 1. According to both strategies, descriptors of individual component (Descriptors D_1 and D_2 for component 1 and component 2, respectively) are weighted as per their molar fractions (x_1 and x_2 for component 1 and 2, respectively).

Alternatively, the users may use similar descriptors in the OCHEM webserver[8, 11].

➢ ***Dataset division strategies for the mixtures:***

QSAR-Mx provides three options for the data-distributions of the mixtures and these are 'Points-out', 'Mixtures-out' and 'Compounds-out' These types of datatset-division strategies were originally proposed by Muratov et al for the mixtures[8, 12, 13].. Below we provide short descriptions (also see the Figure 1). Details of these methods may be obtained from elsewhere.

- **Points-out**: Mixture data points are randomly distributed in a way so that each mixture is present in both the training and test sets.
- **Mixtures out**: Mixtures are distributed in a way so that some mixtures are present in the training set and the rest of the mixtures are placed in the test set. Therefore, each mixture is present either in the training set or in the test set, but never in both sets.
- **Compounds-out:** At least one compound of the dataset is never placed in the training set. **One must note that the compounds-out is the most rigorous method for the validation of the mixtures[9, 12].**

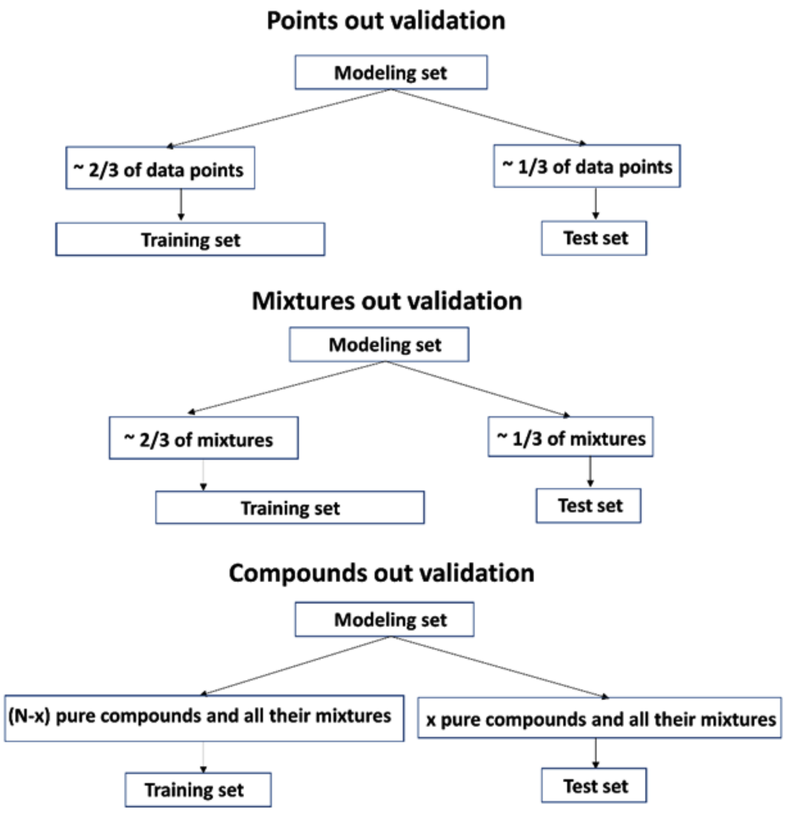
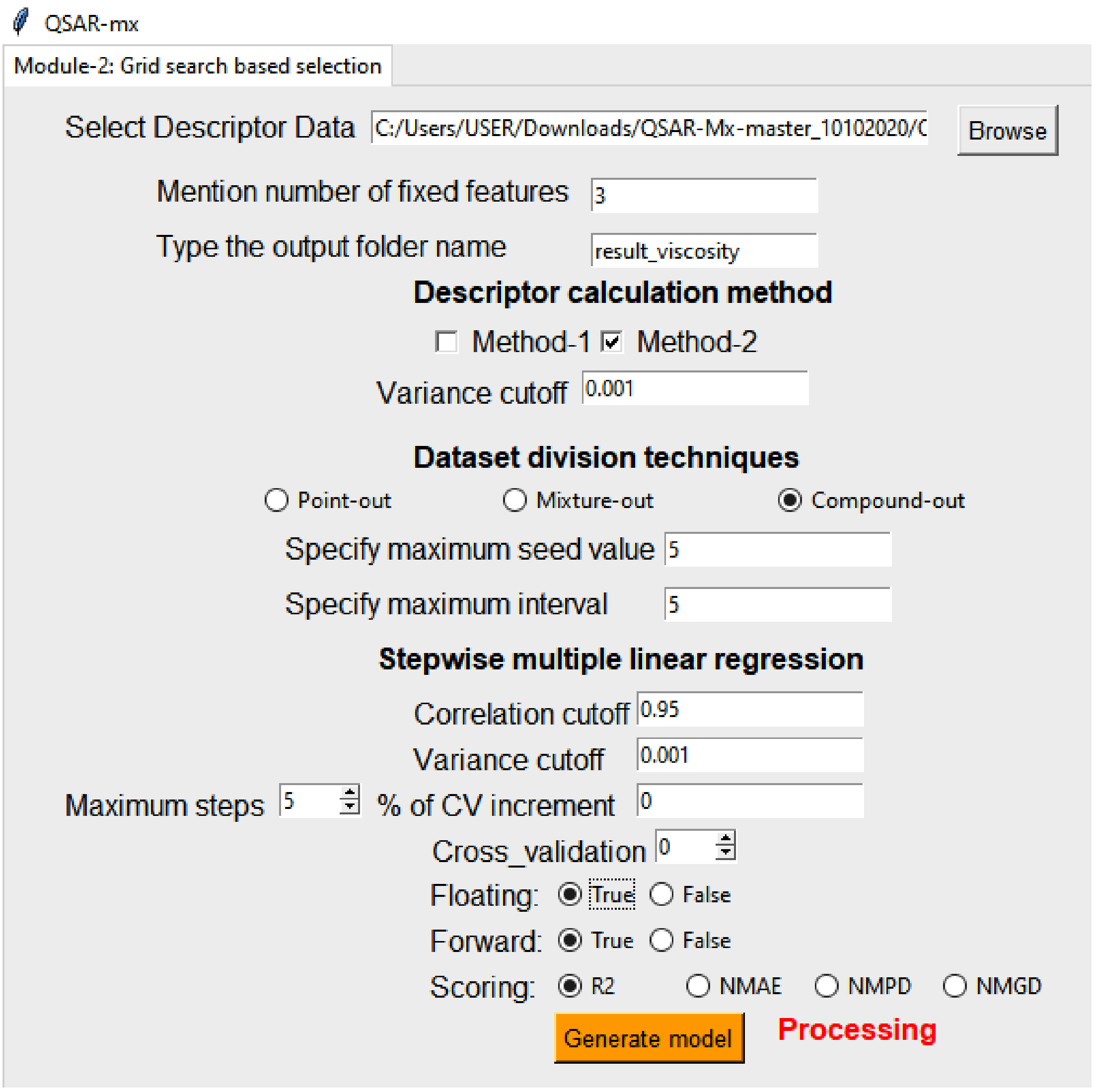


Figure 1. Different strategies used for the validation of the QSTR models of mixtures.

Step 2- Model generation: The user should open the Anaconda command prompt and type 'python QSAR-Mx-M2.py' to open the GUI named 'Module-2: Grid-search based selection'. We should start with this tool as it provides options for generating multiple data-distributions and models.

First, ‘Browse’ and select descriptor data to upload the input data file (visc_dataset.csv) and mention the options provided in Figure 2 and press ‘Generate model’ option to run this tool.


QSAR-mx

Module-2: Grid search based selection

Select Descriptor Data: C:/Users/USER/Downloads/QSAR-Mx-master_10102020/C

Browse

Mention number of fixed features: 3

Type the output folder name: result_viscosity

Descriptor calculation method

Method-1 [✓ Method-2]

Variance cutoff: 0.001

Dataset division techniques

Point-out Mixture-out [✓ Compound-out]

Specify maximum seed value: 5

Specify maximum interval: 5

Stepwise multiple linear regression

Correlation cutoff: 0.95

Variance cutoff: 0.001

Maximum steps: 5

% of CV increment: 0

Cross_validation: 0

Floating: [✓ True] False

Forward: [✓ True] False

Scoring: [✓ R2] NMAE NMPD NMGD

Generate model

Processing

Figure 2. Snapshot of the Module 2 (Grid search based selection) of *QSAR-Mx*

This tool provided multiple options for developing models. In this example, we generated descriptors with Method-2 (described above) and we attempted to create maximum 16 (=4x4) ‘compounds-out’ validation based data-distributions by mentioning the ‘Seed value’ and ‘Interval’ as 5. These two options are the main keys to generate multiple data-divisions.

➢ **Seed value and Interval**

This tool requires two parameters namely the ‘seed value’ and ‘interval’. Initially, the unique component-1 names of the input dataset are sorted out based on their number of instances and then on the basis of seed value (which is basically a starting point) and interval, a few component-1 names are collected. All mixtures containing these names

are placed in the test set. This method is then repeated with component 2 of the dataset. With increasing number of seed value, greater number of data-distributions is generated.
It is noteworthy that, this tool does not accept 0 (starting number in Python) as seed value or interval as the component with maximum number of instances (in modelling dataset) is always placed in the training set.

Furthermore, the models are not generated by this tool with the data-distributions in which the number of test set samples is greater than that of the training set samples. Similarly, this tool also rejects the data-distributions in which the test set sample size is less than 15% of total number of data-points. Therefore, total number of generated models may be less. In this example, the user will get 13 models (instead of 16) because the other models are rejected. The other options are:

- Mention the number of fixed feature: 3 (two indicator parameters + response variable)
- Type the result output folder name: The folder in which the generated files will be submitted
- Descriptor calculation method: Method-1 or Method-2 (do not select both options)
- Variance cut-off: Constant descriptor values must be removed after generation of the mixture descriptors as SFS-MLR fails to run with these. Mention at least 0.001 to remove descriptors with less variance
- Dataset division techniques: Points-out/Mixtures-out/Compounds-out
- **Stepwise multiple linear regression-** A correlation cut-off (current value 0.95) should be mentioned to avoid highly intercorrelated descriptors in the model. The user may also set a variance cut-off (current value 0.001) to be considered during feature selection. Maximum step (current value 5) should be fixed to determine maximum number of descriptors in the model. The importance of '%MAE(LOO) reduction' (current value 0) is described below. Remaining options are specific to the *SequentialFeatureSelector* program of *Mlxtend* (http://rasbt.github.io/mlxtend/). In the current example, we performed model selection with 'R2' as the scoring function and no cross-validation is performed by setting the 'Cross validation' as 0 (in case of 5-fold and 10-fold CV the user should mention 5 and 10 respectively).

Step-3 Result interpretation: The results of the models will automatically be submitted in the 'results_viscosity' folder one by with respect to the data-distributions. After

completion of the calculation, the user should first check the 'Results_table.csv' file in which some major statistical parameters of the generated models are tabulated. This result helps in identifying the best model on the basis of internal and external predictability. More information about all these models may also be observed in the .txt file named 'Cumulative_results'. A 'Warning.txt' appears or not. This text actually appears when the 'compounds-out' based data-distribution is formed with only one component. In such cases, for better validation the user may discard these data-divisions.
Other generated files include data-distribution information files (as *_division.csv), model descriptors and predicted values (as *_trpred,csv/*_tspred.csv) and full statistical details of the individual models (as *_results.txt). The detailed discussions on the statistical parameters may be obtained elsewhere[14, 15]. Only one parameter namely Q^2_{LCO} is discussed in here.

- **Q^2_{LCO} (leave chemical out cross-validation R^2)**

This parameter is used for the assessment of internal predictability based on compounds-out validation menthod. Here, all mixtures formed by a new chemical (with observed property *Yi*) that belongs to component-1 (C1) of the training dataset is removed one by one and after each removal, their predicted values ($\hat{Y}_{\mathrm{L(C1)O}}$) are obtained with the same model generated on the remaining training set samples. Subsequently, each chemical (i.e, all its mixtures) of component-2 (C2) is then treated in the same manner to obtain $\hat{Y}_{\mathrm{L(C2)O}}$. The final parameter Q^2_{LCO} is then calculated as:

$$Q^2_{LCO} = \frac{\left(1-\frac{\Sigma(Y_i-\hat{Y}_{\mathrm{L(C1)O}})^2}{\Sigma(Y_i-Y_m)^2}\right)+\left(1-\frac{\Sigma(Y_i-\hat{Y}_{\mathrm{L(C2)O}})^2}{\Sigma(Y_i-Y_m)^2}\right)}{2}$$

In Eq. (5), *Ym* is the average observed property of the training setsamples. It may be inferred thatalthough Q^2_{LCO} uses the idea of well-known leave-many-out cross-validation approach but this is particularly useful for the internal validation of the models developed with the mixtures. Similarly, the validation parameters such as MAE_{LCO}, rm^2_{LCO}, Δrm^2_{LCO} are also I in the result files.
The large difference between the values of Q^2_{LOO} and Q^2_{LCO} (or MAE_{LOO} and MAE_{LCO}) indicates that the model fitting with mixtures of at least one component is unsatisfactory. It is better to avoid such models.
Finally, LCO based parameters are mainly relevant when we perform 'compounds-out' validation. For 'Points-out' and 'Mixtures-out', these parameters are also generated but we may ignore it during assessment of the result.

From the 'Results_table.csv', one may note that the statistical predictability of the models varied considerably with different test sets. The model developed with data distribution made on seed value of 4 and interval 1 may be considered the best one among these. This model is developed with 52 training and 21 test set samples. To further assess the quality of the modes we examined its corresponding result file '41_results.txt'. First the model showed satisfactory goodness-of-fit as suggested from OLS regression results (This part of the result file is the output we obtained from statsmodel.OLS). The model is developed with 5 descriptors due to the fact maximum 5 descriptors were allowed during model development (i.e, by setting maximum steps:5). Then, one may note that ‘Maximum intercorrelation between descriptors’ is 0.766, indicating that the model does not suffer from high inter-collinearity. Obviously, the user may use less correlation cut-off such as 0.75 if maximum intercorrelation less than this value is desired. Furthermore a number of statistical parameters are provided in this result file to judge internal and external predictabilities of the models.

For further assessment of the descriptor values and to predicted activity of the training and the test sets, one should check ‘41_trpred.csv’ and ‘41_tspred.csv’, respectively. Of course, these results may also be used for further calculations of new statistical parameters.

Further suggestions:

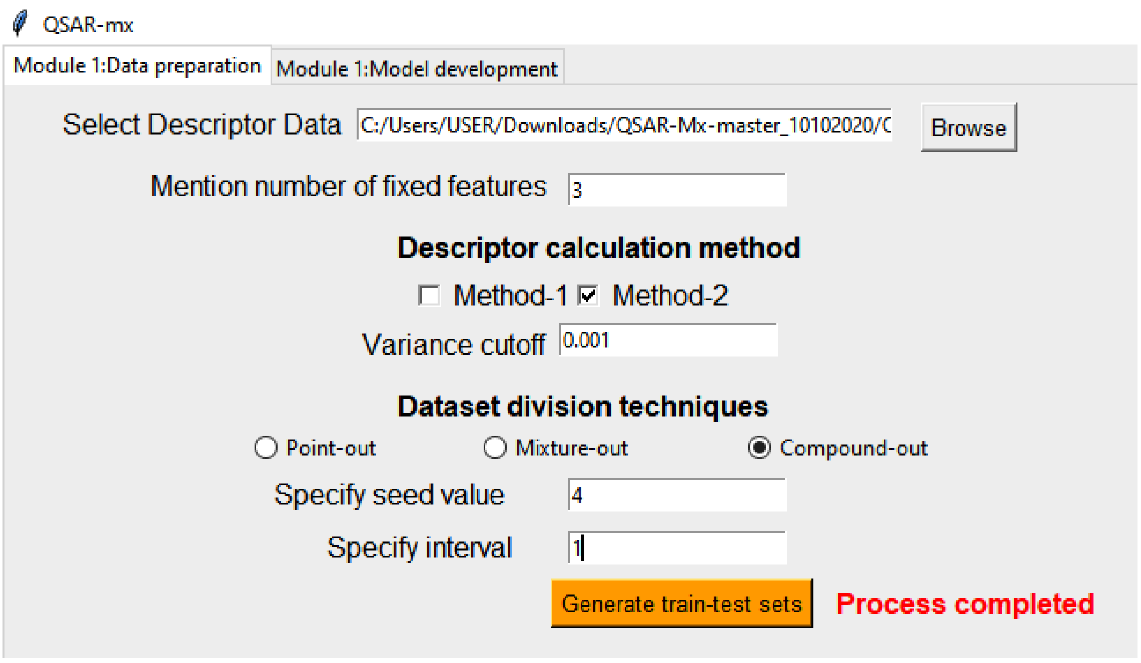
- The above example only shows modelling with one data-division method, i.e, ‘compounds-out’. The users may increase the values of ‘seed value’ and ‘interval’ to obtain more models.
- The user may generate models with ‘mixtures-out’ and ‘points-out’ methods. Note that among these ‘points-out’ is considered as the weakest method as it generates the overfitted models for mixtures. However, one may first compare the models generated with other two methods.
- ‘Mixtures-out’ is preferred when the developed model is expected to predict new mixtures formed with the chemicals present in the modelling dataset. Researchers may prefer reporting both ‘mixtures-out’ and ‘compounds-out’ based models[8].
- For some models, Method-1 based descriptor calculation may produce the most significant model. One can use both these techniques one by one to check which one generates the best model.
- It is strongly suggested that during model development other options of the SFS-QSAR modelling, such as scoring with ‘NMAE’, ‘NMPD’ etc, 5/10-fold CV based selection are utilized one by one. The chance of obtaining more predictive models are increased when a greater number of scoring functions and CV based selections are tried.

Let’s now move to the Module-1, which should be opened with the command ‘python *QSAR*-*Mx*-M1.py’ in Anaconda command prompt. The functionalities of this module are mostly similar to the Module 2 described above. Therefore, detailed description is unnecessary. Module 1 may be used for the following reasons.

- If the user wishes to generate model with only one randomly selected data-distribution.
- For further processing of the best model selected with Module 2. Module 2 takes long time to complete and the best model can thus be quickly regenerated with Module 1.
- More importantly, this module may perform screening of new datasets (with or without response variable) that is not possible with the Module-2.
- Additionally, this module performs *Y*-randomization test to further determine the robustness of the model.
- Generation of scatter plots containing observed and predicted properties.
- To obtain correlation matrix and information about the outliers determined by standardization approach based applicability domain (AD) analysis, as suggested by Roy et al[16].

In contrary to Module-2, where the results are submitted in a new folder of the current working directory (CWD), the results generated by this module are deposited in the CWD. Names of most of these results files will start from the input data files.

Moving on, as we already know that the best viscosity model is generated with seed value 4 and interval 1, we will now regenerate the same model using Module-1 starting with the input file 'visc_dataset.csv'. First, we need to create the data-distribution based on these values from the 'Data preparation' (Tab1) of this module by mentioning the options provided below.


QSAR-mx

Module 1:Data preparation | Module 1:Model development

Select Descriptor Data: C:/Users/USER/Downloads/QSAR-Mx-master_10102020/C

Mention number of fixed features: 3

Descriptor calculation method

☐ Method-1 ☑ Method-2

Variance cutoff: 0.001

Dataset division techniques

○ Point-out ○ Mixture-out ● Compound-out

Specify seed value: 4

Specify interval: 1

Process completed

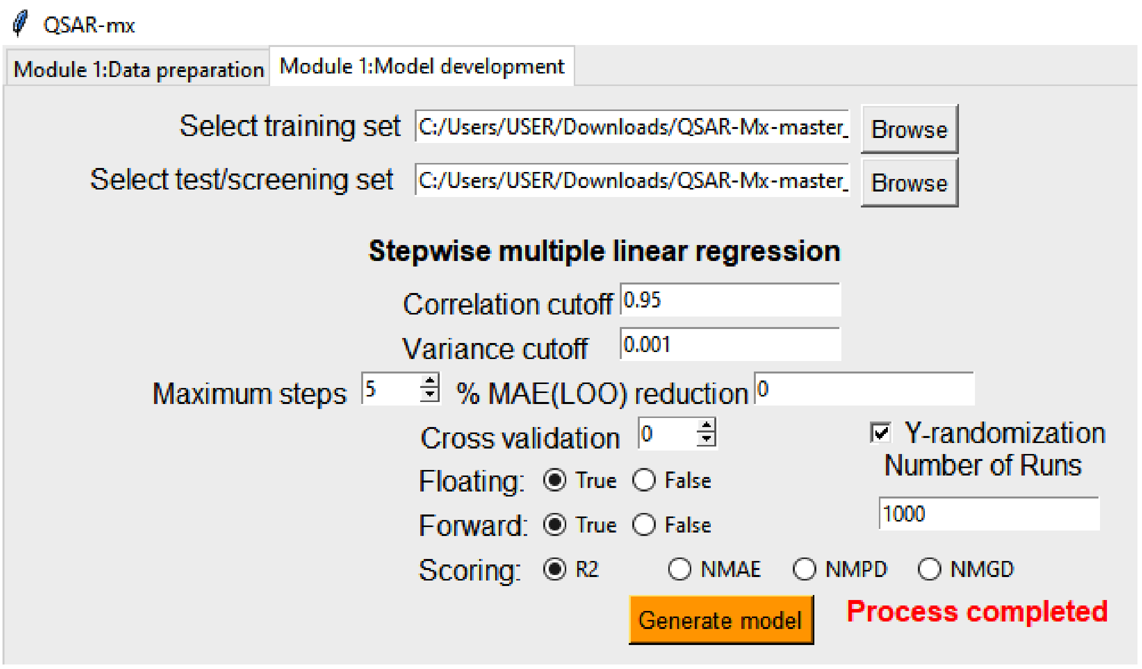
Figure 3. Input data preparation with Module-1 of *QSAR-Mx*

Once completed, the training set and test set data with mixed descriptors (generated with Method-2) will be found as 'visc_datasetco_s_4_i_1_tr.csv' and 'visc_datasetco_s_4_i_1_ts.csv'. Additional files will include the data-division information file as 'visc_dataset_co_division.csv'.

➢ ***Removal of duplicate data***

Both Module 1 and Module 2 are capable of searching duplicate data-points in the input dataset. These programs search for duplicates on the basis of values provided in the second column to the column specifying the response variable (i.e, ninth column of the current dataset). Therefore, if the same sample is present with two different response variables, the program will not identify these as duplicates. If the values are found same, a warning will appear during the data-division as *'Duplicates are found in the data, will be removed'*. If duplicates are found and removed, the user may get additional files in the CWD as '*duplicates.csv' and '*duplicatesremoved.csv' to check what duplicates are present in the data and to obtain the dataset without any duplicate sample.

Next, for the model generation move to 'Model development' tab and upload the training and test set files, one by one and mention the model generation options (See Figure 4) and press 'Generate model'.


QSAR-mx

Module 1:Data preparation | Module 1:Model development

Select training set: C:/Users/USER/Downloads/QSAR-Mx-master_

Select test/screening set: C:/Users/USER/Downloads/QSAR-Mx-master_

Stepwise multiple linear regression

Correlation cutoff: 0.95

Variance cutoff: 0.001

Maximum steps: 5

% MAE(LOO) reduction: 0

Cross validation: 0

Floating: [✓ True] False

Forward: [✓ True] False

Scoring: [✓ R2] NMAE NMPD NMGD

[✓ Y-randomization]

Number of Runs: 1000

Process completed

Figure 4. Model development with Module-1 of *QSAR-Mx*

Note that we additionally perform the Y-randomization test. In case the user does not want to perform it, keep the '*Y*-randomization' option unclicked. After completion of the run, the statistical results of the generated model could be checked from the 'visc_datasetco_s_4_i_1_tr_results.txt'. These results must be same as the model generated with Module 2. Note that the at the end of this file, *Y*-randomization results are provided as 'Crp2 after 1000 run:'. A high value of it implies that the generated model is unique and it is not developed by chance. The scatter plot generated with the Matplotlib program may be examined to understand how model is fitted. Next, we need to check the files *_trpr.csv and *_tspr.csv that contained the selected descriptors, predicted activity and outlier information. Note that no outlier was found in the training set whereas the test set contains only one outlier. From the file '*_corr.csv', the correlation matrix may be checked.

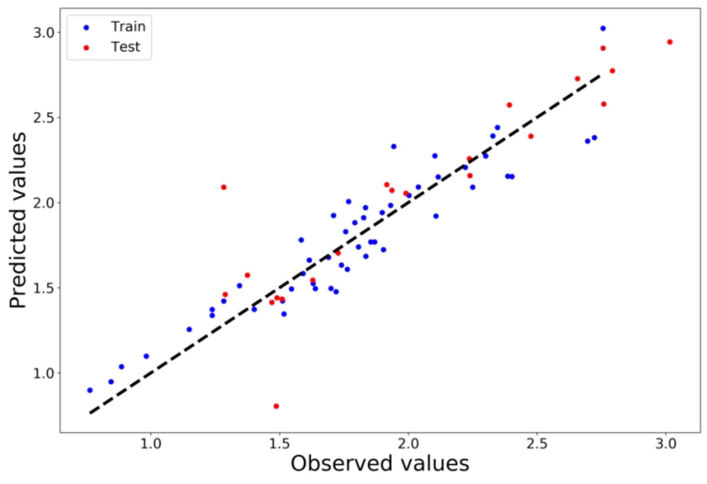


Figure 5. Scatter plot generated with *QSAR-Mx* after model development

In the current model 52 training datapoints are characterized with 5 descriptors. Now, after the model development, we may check if the model has sufficient number of variables or not. Since, SFS-MLR includes the descriptors one by one and increasing the internal validation parameters. One such important parameter is MAE_{LOO} (leave-one-out based mean absolute error). '%MAE(LOO) reduction' option of both Module 1 and 2 are designed to allow the rule of addition of descriptor during feature selection of the model based on the internal predictability. For this repeat the model generation described above by setting the '%MAE(LOO) reduction' option as 10 and increase the 'Maximum steps to 8'. After finishing the run the same model containing 5 descriptors will be regenerated. Here, the program will allow inclusion of additional descriptor only when the MAE_{LOO} value of the new model (i.e., after addition of the descriptor) is reduced to >10% as compared to the MAE_{LOO} of the existing model. When the reduction is less than 10%, the addition is terminated. The user may determine what value is to be set in the '%MAE(LOO) reduction' option. The condition we defined (10% reduction) justified that the model has sufficient number descriptors. This method is useful to determine the number of descriptor when we have very limited number of data-samples or considerably large number data-samples. To further understand its utility, set the '%MAE(LOO) reduction' option as 5 increasing the 'Maximum steps' to 10 and regenerate the model. This time the model will be developed with a total 8 descriptors (since we relaxed the condition of descriptor addition). However, note that the external predictability is also reduced as compared to the previous model.

Multi-objective optimization

Finally, we include a short demonstration of the utility of the third module of the *QSAR-Mx*, which is used for the multi-objective optimization (MOOP) method. This method has been described in detail previously[17-23]. The *MOOP.py* provided with the *QSAR-Mx* is a simple tool that performs MOOP with *Derringer's desirability function*.

Let's assume a situation where we have two properties, one desirable (i.e, high value of which is required, i.e, higher therapeutic activity from a drug) and the other is undesirable (i.e, high value of which is required, i.e, higher toxicity from a drug). With the observed properties we develop two different predictive QSAR models, one for the desired property and another for the undesired property. Now, observed and properties properties of these two models may be optimized by *Derringer's desirability function* in order to obtain the observed and predicted desirability scores of the desired (*Ddes*) and undesired (*Dudes*) properties. Finally we may compute the obsrved and predicted global desirability scores (D) for the compounds. The global desirability scores are highly usedful to identify the compunds with optimized properties (i.e., increased desired property and reduced undesired property).

- **Derringer's desirability function based MOOP**

The mathematical fromula for Derringer's desirability function are provided elsewhere. Briefly, the the function applied to *desired (Ddes)* and undesired (Dudes) property are defined as Eq. 1 and 2:

$$d_i = \begin{cases} 0 & \text{if } Y_i^{pred} \le L_i \\ \left[\dfrac{Y_i^{pred} - L_i}{T_i - L_i}\right]^s & \text{if } L_i < Y_i^{pred} < T_i \\ 1 & \text{if } Y_i^{pred} \ge T_i = U_i \end{cases} \qquad (1)$$

$$d_i = \begin{cases} 1 & \text{if } Y_i^{pred} \le T_i = L_i \\ \left[\dfrac{Y_i^{pred} - U_i}{T_i - U_i}\right]^s & \text{if } U_i > Y_i^{pred} > T_i \\ 0 & \text{if } Y_i^{pred} \ge U_i \end{cases} \qquad (2)$$

Where, YiPred is the predicted activity, L_i and U_i are the selected minimum and maximum values values of the property to scale. The value of the exponent *s* is determined by setting a predicted property's value to a desirability value equal to 0.5. Therefore, s of Eq. 1 and 2 may be calculated as:

$$s = \frac{\log(0.5)}{\log\left(\dfrac{x - L_i}{U_i - L_i}\right)}$$

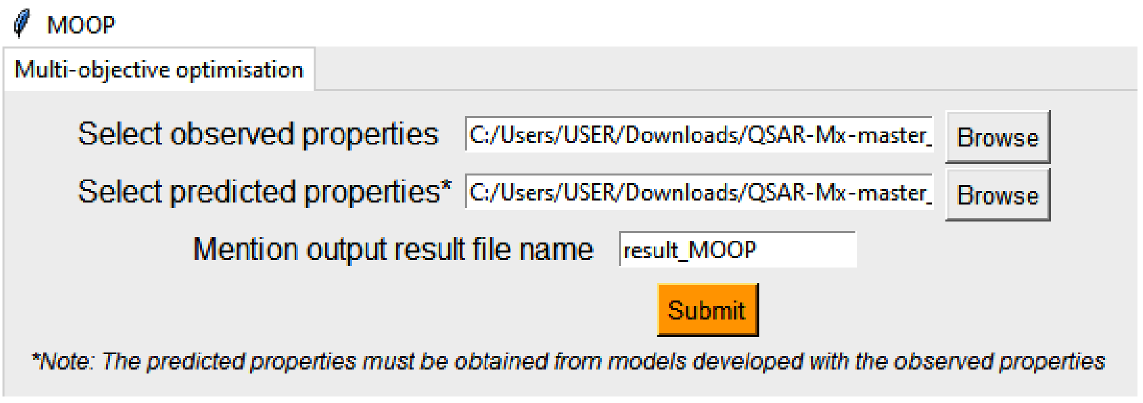
In this MOOP.py program, x is the median of the observed property. The global desirability scores are calculated as:

$$D = (D_{des} \times D_{udes})^{1/2}$$

One should note that for the same samples used for modelling of the desired property, should be used for the modelling of the undesired property.

In the previous example, we developed QSAR models with viscosity of 73 DESs. High viscosity of DESs is often associated high industrial processing cost, and therefore, high value of it is undesired. At the same time, some DESs are capable of absorbing gas such as CO_2, SO_2, etc. In such cases, high CO_2 absorption is considered as a desired property. Therefore, another QSAR model was developed with the same samples used in viscosity QSAR model, but this time as response variable we used their reported CO_2 absorption capacity. After developing models for CO_2 absorption and viscosity, we will utilize experimental and predicted values of the two models for optimization.

First open the GUI by the command 'python MOOP.py' in Anaconda command prompt. Upload the attached 'observed_properties.csv' and 'predicted_propeties'. Also mention the output .csv file name in which the results will be found.


MOOP

Multi-objective optimisation

Select observed properties: C:/Users/USER/Downloads/QSAR-Mx-master_

Select predicted properties*: C:/Users/USER/Downloads/QSAR-Mx-master_

Mention output result file name: result_MOOP

*Note: The predicted properties must be obtained from models developed with the observed properties

Figure 6. *QSAR-Mx* GUI for multi-objective optimisation technique

On should note that samples we used in these two files are different. However, in both these files, there are two columns. In the first column the desired property must be mentioned and in the second, the undesired property should exist. Open the output file to check the results. The desirability scores of desired and undesired properties will be found in the columns named 'Des_d' and 'Des_u'. In the last column (named 'Global_des'), the global desirability scores for each data-point is computed. By sorting the last column, we will find the samples with maximum global desirability scores. In this case, XVS025 should appear with the maximum score of 0.77 indicating that this solvent is the suitable among all other mixtures (present in this file) when high gas absorption is required along with low viscosity.

Project Leader

***Prof.* M. Natália D. S. Cordeiro**
Associate Professor
ncordeir@fc.up.pt
LAQV@REQUIMTE
Department of Biochemistry and Chemistry
FACULDADE DE CIÊNCIAS DA
UNIVERSIDADE DO PORTO
Rua do Campo Alegre, s/n, 4169-007 Porto, Portugal
www.fc.up.pt

Software Developers

Core Programmer:

Dr. Amit Kumar Haldar

(amit.halder@fc.up.pt)

Co-Developer:

***Prof.* M. Natália D. S. Cordeiro**

(ncordeir@fc.up.pt)

Funded By

This work was supported by UID/QUI/50006/2020 with funding from FCT/MCTES through national funds.

Ackowledgement

The developers of this tool acknowledge the contributions of Dr. Yunierkis Perez (University of Las Américas, Quito, Ecuador) and Dr. Pravin Ambure (ProtoQSAR SL, Valenicia, Spain) for the evolution of this tool and for their valuable suggestions to improve its functionalities.

External Libraries used

The current tool utilizes some well-known Python based libraries such as NumPy [5], SciPy[4], Pandas[3], Matplotlib[2], Tkinter (https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/index.html), and Scikit-learn [24, 25], MLxtend(http://rasbt.github.io/mlxtend/).

Citation

Halder, A. K., Ambure, P., Perez, Y, Cordeiro, M. N. D. S. Turning deep-eutectic solvents into value-added products for CO2 capture: A desirability multi-objective optimisation study (Communicated).

References

1. Van Rossum, G., & Drake, F. L. , Python 3 Reference Manual. Scotts Valley, CA: CreateSpace. **2009**.
2. Hunter, J. D., Matplotlib: A 2D graphics environment. *Comput Sci Eng* **2007,** *9* (3), 90-95.
3. W, M., Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010).
4. Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, I.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P.; Contributors, S., SciPy 1.0: fundamental algorithms for scientific computing in Python (vol 17, pg 261, 2020). *Nat Methods* **2020,** *17* (3), 352-352.
5. van der Walt, S.; Colbert, S. C.; Varoquaux, G., The NumPy Array: A Structure for Efficient Numerical Computation. *Comput Sci Eng* **2011,** *13* (2), 22-30.
6. Garcia, G.; Aparicio, S.; Ullah, R.; Atilhan, M., Deep Eutectic Solvents: Physicochemical Properties and Gas Separation Applications. *Energ Fuel* **2015,** *29* (4), 2616-2644.
7. Mauri, A.; Consonni, V.; Pavan, M.; Todeschini, R., Dragon software: An easy approach to molecular descriptor calculations. *Match-Commun Math Co* **2006,** *56* (2), 237-248.
8. Oprisiu, I.; Novotarskyi, S.; Tetko, I. V., Modeling of non-additive mixture properties using the Online CHEmical database and Modeling environment (OCHEM). *J Cheminformatics* **2013,** *5*.
9. Halder, A. K., Cordeiro, M. N. D. S., Development of Predictive Linear and Non-linear QSTR Models for Aliivibrio Fischeri Toxicity of Deep Eutectic Solvents. *International Journal of Quantitative Structure-Property Relationships (IJQSPR)* **2019,** *4* (4), 50-69.
10. Halder, A. K.; Natalia, M.; Cordeiro, D. S., Probing the Environmental Toxicity of Deep Eutectic Solvents and Their Components: An In Silico Modeling Approach. *Acs Sustain Chem Eng* **2019,** *7* (12), 10649-10660.
11. Abdelaziz, A.; Tetko, I. V.; Sushko, I.; Novotarskii, S.; Koerner, R., OCHEM: Online public platform for human and environmental toxicity modeling. *Abstr Pap Am Chem S* **2014,** *248.*
12. Muratov, E. N.; Varlamova, E. V.; Artemenko, A. G.; Polishchuk, P. G.; Kuz'min, V. E., Existing and Developing Approaches for QSAR Analysis of Mixtures. *Mol Inform* **2012,** *31* (3-4), 202-221.
13. Muratov, E. N.; Varlamova, E. V.; Artemenko, A. G.; Polishchuk, P. G.; Nikolaeva-Glomb, L.; Galabov, A. S.; Kuz'min, V. E., QSAR analysis of poliovirus inhibition by dual combinations of antivirals. *Struct Chem* **2013,** *24* (5), 1665-1679.
14. Gramatica, P., On the development and validation of QSAR models. *Methods Mol Biol* **2013,** *930*, 499-526.
15. Roy, K.; Kar, S.; Das, R. N., A Primer on QSAR/QSPR Modeling : Fundamental Concepts. In *SpringerBriefs in Molecular Science,* [Online] 1st ed.; Springer International Publishing : Imprint: Springer,: Cham, 2015; pp. 1 online resource (X, 121 pages 47 illustrations).

16. Roy, K.; Kar, S.; Ambure, P., On a simple approach for determining applicability domain of QSAR models. *Chemometr Intell Lab* **2015,** *145*, 22-29.
17. Cruz-Monteagudo, M.; Borges, F.; Cordeiro, M. N.; Cagide Fajin, J. L.; Morell, C.; Ruiz, R. M.; Canizares-Carmenate, Y.; Dominguez, E. R., Desirability-based methods of multiobjective optimization and ranking for global QSAR studies. Filtering safe and potent drug candidates from combinatorial libraries. *J Comb Chem* **2008,** *10* (6), 897-913.
18. Cruz-Monteagudo, M.; Borges, F.; Cordeiro, M. N. D. S., Desirability-based multiobjective optimization for global QSAR studies: Application to the design of novel NSAIDs with improved analgesic, antiinflammatory, and ulcerogenic profiles. *J Comput Chem* **2008,** *29* (14), 2445-2459.
19. Cruz-Monteagudo, M.; Borges, F.; Cordeiro, M. N. D. S., Jointly Handling Potency and Toxicity of Antimicrobial Peptidomimetics by Simple Rules from Desirability Theory and Chemoinformatics. *J Chem Inf Model* **2011,** *51* (12), 3060-3077.
20. Cruz-Monteagudo, M.; Cordeiro, M. N. D. S.; Teijeira, M.; Gonzalez, M. P.; Borges, F., Multidimensional Drug Design: Simultaneous Analysis of Binding and Relative Efficacy Profiles of N6-substituted-4'-thioadenosines A(3) Adenosine Receptor Agonists. *Chem Biol Drug Des* **2010,** *75* (6), 607-618.
21. Cruz-Monteagudo, M.; Cordeiro, M. N. D. S.; Tejera, E.; Dominguez, E. R.; Borges, F., Desirability-Based Multi-Objective QSAR in Drug Discovery. *Mini-Rev Med Chem* **2012,** *12* (10), 920-935.
22. Cruz-Monteagudo, M.; PhamThe, H.; Cordeiro, M. N. D. S.; Borges, F., Prioritizing Hits with Appropriate Trade-Offs Between HIV-1 Reverse Transcriptase Inhibitory Efficacy and MT4 Blood Cells Toxicity Through Desirability-Based Multiobjective Optimization and Ranking. *Mol Inform* **2010,** *29* (4), 303-321.
23. Perez-Castillo, Y.; Sanchez-Rodriguez, A.; Tejera, E.; Cruz-Monteagudo, M.; Borges, F.; Cordeiro, M. N. D. S.; Huong, L. T. T.; Hai, P. T., A desirability-based multi objective approach for the virtual screening discovery of broad-spectrum anti-gastric cancer agents. *Plos One* **2018,** *13* (2).
24. Hao, J. G.; Ho, T. K., Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language. *J Educ Behav Stat* **2019,** *44* (3), 348-361.
25. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E., Scikit-learn: Machine Learning in Python. *J Mach Learn Res* **2011,** *12*, 2825-2830.