USER MANUAL

QSAR-Co-X
(Version 1.0.0)

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-Co-X is an extension version of *QSAR-Co* (QSAR with conditions) which is available in public domain (https://sites.google.com/view/qsar-co). *QSAR-Co* was developed with a purpose to provide a user-friendly platform to perform multitarget QSAR modeling using Box-Jenkins moving average method[1]. For details of this method and techniques, please read some recent publications from our group[2-4]. Other investigations reported with such modelling technique are [5-11].

Briefly, multitarget QSAR (mt-QSAR) allows incorporation of different experimental assay conditions which are expressed as experimental element or *cj*. The input descriptors (*Di*) are converted to deviation descriptors by Box-Jenkins moving average operator and these deviation descriptors are subsequently used for developing the mt-QSAR models. Previously launched *QSAR-Co* provides genetic algorithm based linear discriminant analysis (GA-LDA) for development of linear interpretable models. The non-linear models are developed with random forest (RF) method. *QSAR-Co-X,* on the other hand, provides additional features, which are as follows:

- Dataset division options
- Box-Jenkins moving average operators
- Feature selection methods
- Machine learning algorithms
- Hyperparameter tuning for machine learning methods
- *Yc*-randomization (Upgraded method of previously described *Y*-randomization[1])
- Correlation matrix analyses
- Condition-wise-prediction

Overall, *QSAR-Co-X* combined with previously launched *QSAR-Co* provide useful platforms for developing mt-QSAR models.

Important concepts

FS-LDA[2, 7]**:** the independent descriptors are included in the model stepwise depending on specific statistical parameter. Here, the features are selected and included in the model stepwise by the *p*-values in an *F*-statistic. Initially, criteria for forward selection (i.e. p-value to enter) and backward elimination (*p*-value to remove) are set. The descriptor with the lowest *p*-value is included first and subsequently other descriptors are included in the model based on the lowest p-value only if the criteria for forward selection is met. However, if the *p*-value of a descriptor included in the model is found to be greater than ‘p-value to remove’, it is eliminated from the model. In the current work, both p-to enter and p-to remove are fixed as 0.05. The final LDA models were developed and were subsequently validated using *LinearDiscriminantAnalysis* function of Scikit-learn[12, 13].

SFS-LDA[14]**:** It adds features into an empty set until the performance of the model is not improved either by addition of another feature or maximum number of features is reached. Similar to FS-LDA this technique is also a greedy search algorithm where the best subsets of descriptors are selected stepwise and the model performance is justified by the user specific statistical measure. In the current work, python based *SequentialFeatureSelector algorithm* mlxtend (http://rasbt.github.io/mlxtend/) library was used for the development of model.

***Yc*-randomization:** The *Yc*-randomization is a modification of *Y*-randomization method that was implemented in *QSAR-Co*. Generally, *Y*-based randomization test justifies that the linear model is not developed by chance. In conventional chemometric modelling, the response variable is randomly shuffled *n* times to generate *n* number of randomized models, the statistical parameters of which are then compared to that of the original model. In *Yc* randomization, the response variable is shuffled along with the experimental conditions and therefore for n run n number of randomized deviation descriptors are produced and these are then fitted to the original model. Ideally, the average Wilk’s lambda obtained from these randomized models should be considerably higher than that of original model.

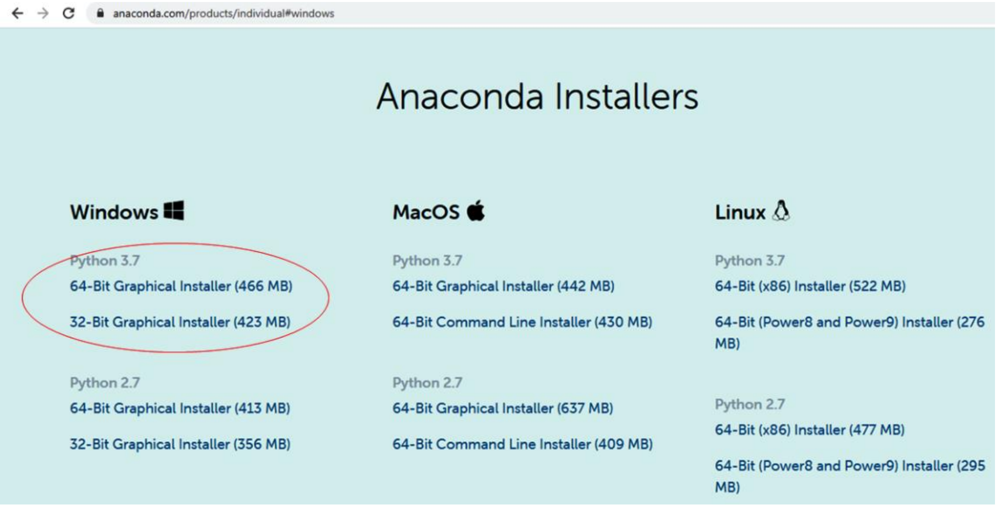
Statistical parameters: The goodness of fit predictability of the models are determined through a number of statistical parameters like, Wilk’s lambda (λ), p value, F-value, true positive (TP), true negative (TN), false positive (FP), false negative (FN), sensitivity, specificity, accuracy, F1 score (or F-measure), Matthews correlation coefficient (MCC) and area under the receiver operating characteristic curve (AUCROC). These parameters were discussed in detail in the *QSAR-Co* manual (https://sites.google.com/view/qsar-co/manual-and-license).

Applicability domain (AD): The AD of a linear model (i.e., FS-LDA or SFS-LDA) is determined through *Standardization Approach*[15] to identify the structural outliers. For non-linear models *Confidence Estimation Approach*[16, 17]is used with a threshold value of 0.5. For further details about these approach, check *QSAR-Co* manual (https://sites.google.com/view/qsar-co/manual-and-license).

Condition wise prediction: *QSAR-Co-X* introduced this automated and simple result analysis of mt-QSAR results generated by this tool. The mt-QSAR technique implemented in *QSAR-Co-X* (and also *QSAR-Co*) generates a unique model with dataset containing different experimental conditions. Therefore, after developing the model, we may need to assess how the model was predictive to a certain experimental condition. This module is used to observe how the model predicts different conditions. Moreover, with the applications of different model development strategies, we often end up with models, the predictability of which is very similar to each other. In such situation, the model which is more predictive to certain experimental condition may be preferred over the best model. Moreover, since this analysis allows us to identify the cases which are less predictive to certain experimental conditions and if necessary, such conditions may be removed and the models may be regenerated to obtain more predictive and/or more significant models. Finally, experimental conditions with negligible number of cases may be identified through this analysis and if the generated model is found less predictive towards such conditions, these may be removed to regenerate the model.

Installation of dependencies of *QSAR-Co-X*:

1. *QSAR-Co-X* is a python-based[18] tool that has multiple dependencies[19-22]. 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)

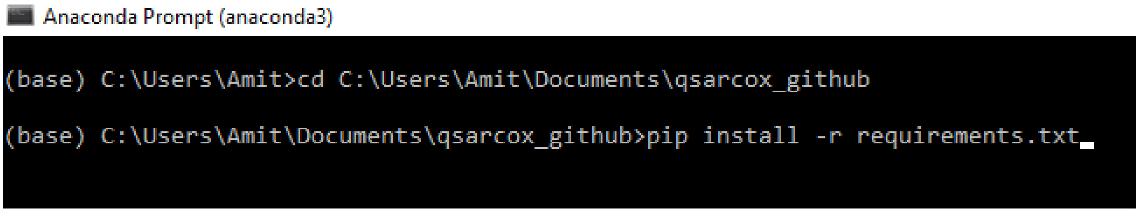
Python 2.7
64-Bit Graphical Installer (637 MB)
64-Bit Command Line Installer (409 MB)

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’**.

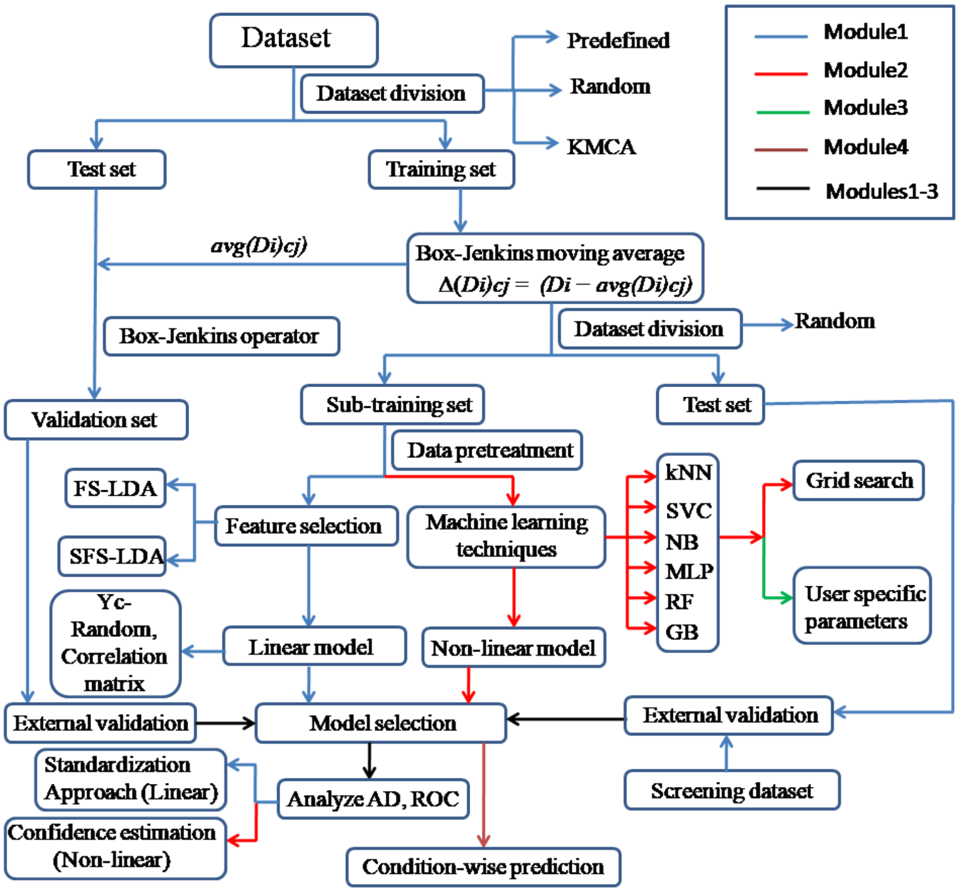


```
(base) C:\Users\Amit>cd C:\Users\Amit\Documents\qsarcox_github

(base) C:\Users\Amit\Documents\qsarcox_github>pip install -r requirements.txt_
```

Modules of the QSAR-Co-X

QSAR-Co-X is divided into four modules that are named as Module-1, Module-2, Module-3 and Module-4. These four modules should be run from Anaconda with command ‘**python Module1.py**’,‘**python Module2.py**’,‘**python Module3.py**’ and ‘**python Module4.py**’, respectively. The overall functionalities of this tool are shown below.



Module-1

Module1 helps in the data-preparation and development of linear interpretable models. It has three tabs, namely (a) Data preparation, (b) Model development and (c)Screening/validation.

(A) Data preparation:

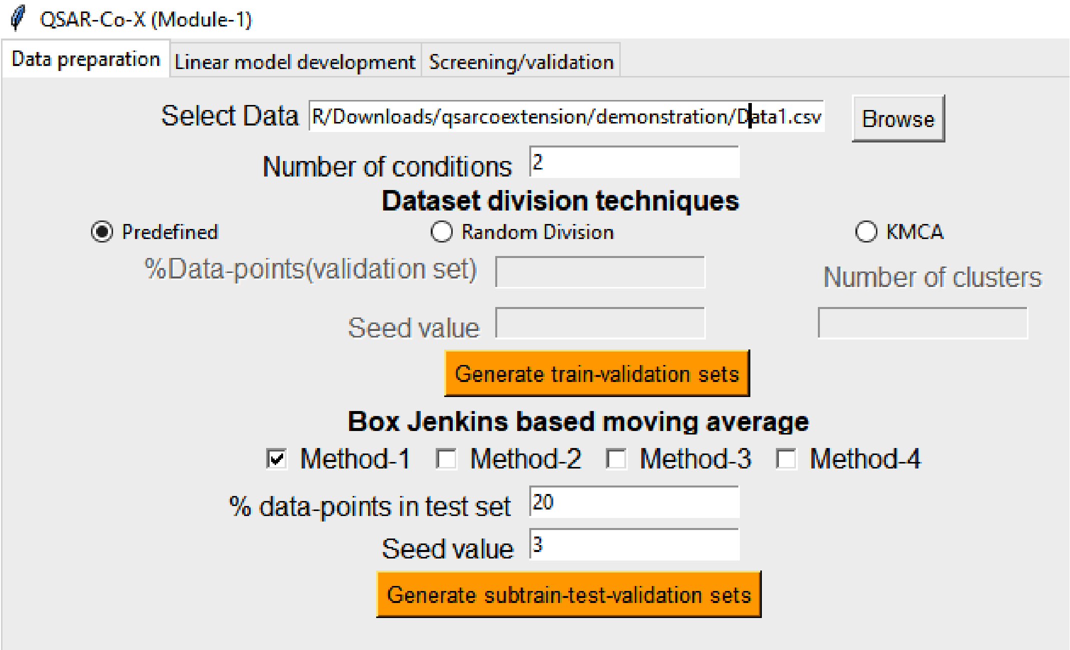
Similar to QSAR-Co, the current tool requires a dataset in .csv format where the sample number, response variable, experimental elements (c_j) and the starting descriptors (*Di*) should be clearly mentioned one after another in a well-organized manner.

(i) Under the tab '**Select data**', Browse and open the input dataset. The current directory will be used as default folder for dataset/file selection and saving.

(ii) Mention the '**Number of conditions**'. For example, in the enclosed **'Data1.csv'** there are two experimental elements namely 'C1_me' and 'C2_bs'. Therefore, under this option 2 should be selected. **Note: The users are advised to mentain such format for mention the conditions in the input .csv files. For example, in order to mention conditions more than two, the named should be presented as 'C1_me', 'C2_bt', 'C3_at', and so on.**

(iii) The next option is dataset division and under the option **'Dataset division technique'** select '**Predefined**' or '**Random Division**' or '**KMCA**'.

- **Predefined**: For predefined training and validation sets the last column should contain a column (with any name) describing the training set compounds as 'Train' and test set compounds as 'Test'. The enclosed **'Data1.csv'** is an example.



QSAR-Co-X (Module-1)

Data preparation | Linear model development | Screening/validation

Select Data: R/Downloads/qsarcoextension/demonstration/Data1.csv — Browse

Number of conditions: 2

Dataset division techniques

● Predefined ○ Random Division ○ KMCA

%Data-points(validation set):

Seed value:

Number of clusters:

Generate train-validation sets

Box Jenkins based moving average

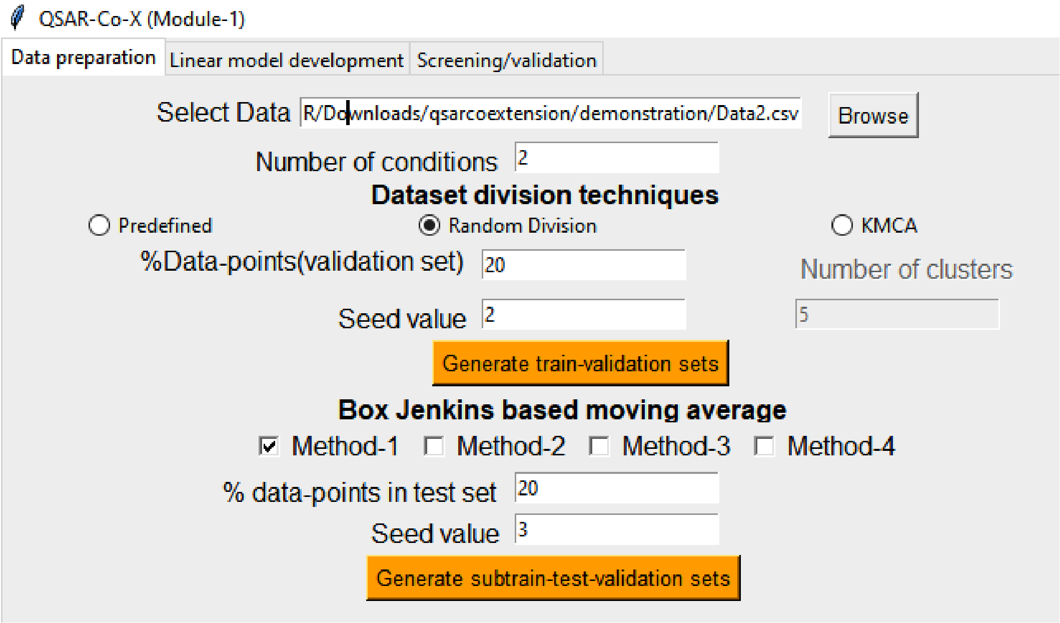
☑ Method-1 □ Method-2 □ Method-3 □ Method-4

% data-points in test set: 20

Seed value: 3

Generate subtrain-test-validation sets

- **Random division**: The random division will randomly divide the data depending on the '%Data-points (validation set)' and 'Seed value'. For example, click on **'Random'**, put '20' in the **'%Data-points (validation set)'** option in order to place 20% of the data in the validation set and remaining 80% data as training set. **'Seed value'** is nothing but a number (e.g., 1,2,3, etc) change of which will alter the data-distribution. The user may get *n* number of data-distributions by changing its value. Consider the enclosed file **'Data2.csv'**.



QSAR-Co-X (Module-1)

Data preparation | Linear model development | Screening/validation

Select Data: R/Downloads/qsarcoextension/demonstration/Data2.csv — Browse

Number of conditions: 2

Dataset division techniques

○ Predefined ● Random Division ○ KMCA

%Data-points(validation set): 20

Seed value: 2

Number of clusters: 5

Generate train-validation sets

Box Jenkins based moving average

☑ Method-1 □ Method-2 □ Method-3 □ Method-4

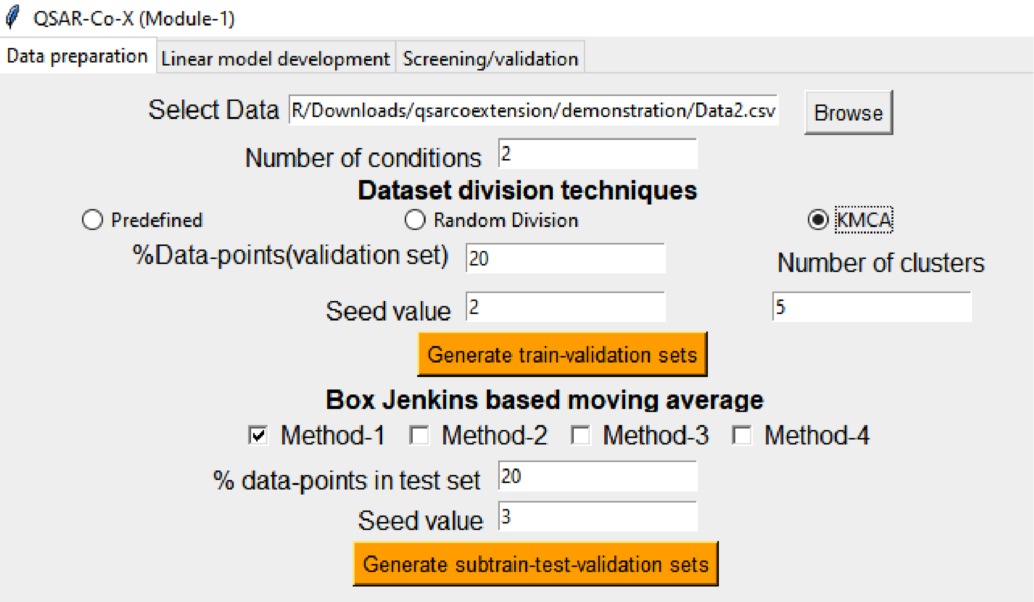
% data-points in test set: 20

Seed value: 3

Generate subtrain-test-validation sets

- **KMCA**: The ‘k-means cluster analysis’ or **‘KMCA’** is a rational data-division method that initially creates ‘n’ number of clusters depending on the value put by the user in the **‘Number of clusters’** option. For example, if a value 5 is set in this option, the data will be divided into 5 clusters on the basis of response variable and starting descriptors. The user needs to mention **‘%Data-points (validation set)’** and **‘Seed value’** (similar to random division) since from each cluster the test set compounds will be collected randomly. Consider the attached file **‘Data2.csv’**.

(iv) After selecting the option from **‘Dataset division techniques’** press **‘Generate train-validation sets’.**



Select Data R/Downloads/qsarcoextension/demonstration/Data2.csv

Number of conditions 2

Dataset division techniques

○ Predefined ○ Random Division ● KMCA

%Data-points(validation set) 20

Seed value 2

Number of clusters 5

Box Jenkins based moving average

☑ Method-1 □ Method-2 □ Method-3 □ Method-4

% data-points in test set 20

Seed value 3

(v) In the next step, the Box-Jenkins operator will be applied on the training set to produce the deviation descriptors ($\Delta(D_i)cj$) from the input/starting descriptors (Di). The current tool has four different Box-Jenkins operators as listed in Table 1. Subsequently, the deviation descriptors will be calculated for the validation set. The training set will be randomly divided into a sub-training set and test set depending on the random division options discussed above.

Table 1. Box-Jenkins operators currently available in *QSAR-Co-X*

Method	Operator	Remark
Method-1	$\Delta(D_i)cj = D_i - avg(D_i)c_j$	$avg(D_i)c_j = \sum_{i=1}^{n(c_j)} D_i$
Method-2	$\Delta(D_i)cj=(D_i- avg(D_i)c_j)/(D_{imax} -D_{imin})$	D_{imax}=Maximum value of Di D_{imin}= Minimum value of Di
Method-3	$\Delta(D_i)cj= (D_i - avg(D_i)c_j)/[(D_{imax} -D_{imin})*p(c_j)_c]$	$p(c_j)_c$= n(c_j)/N
Method-4	$\Delta(D_i)cj= (D_i - avg(D_i)c_j)/[p(c_j)_u]$	$p(c_j)_u$ is user-specific $p(c_j)$*

Method-5	$\Delta(D_i)cj= (D_i - avg(D_i)c_j)/[(D_{imax} -D_{imin})*p(c_j)_u]$	Not implemented in QSAR-Co-X, follow the instructions below[#]

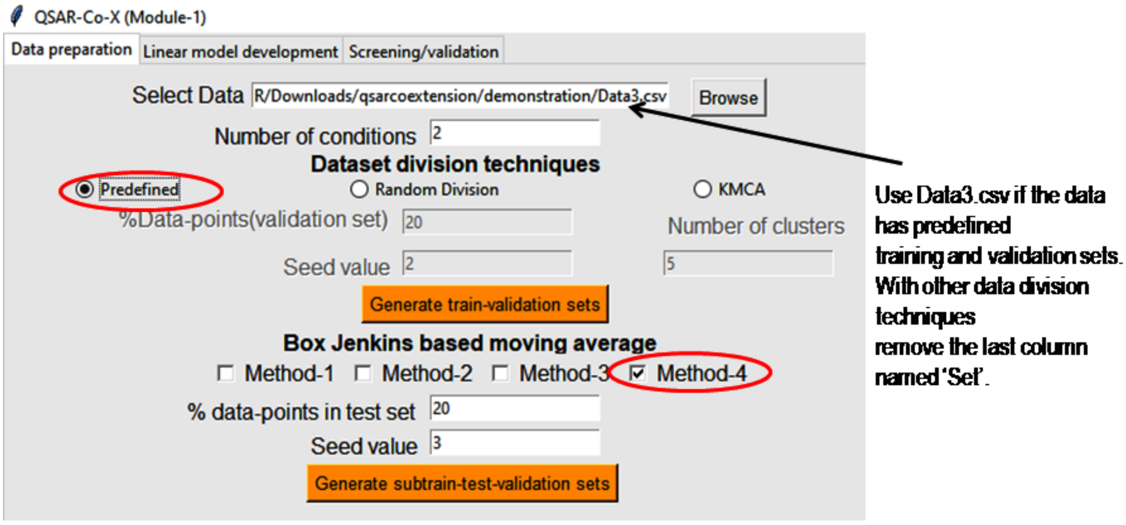
*Note that for Method-4, the user needs to specify the p(cj)u values at the end of the of the input file with specific column names as (i.e., p_cj). The attached file **'Data3.csv'** is an example (see graphics below).

#Also note that a Method-5 may be adopted by simply changing the sentence 'from boxjenk import boxjenk as bj' to 'from boxjenk2 import boxjenk as bj' in the file *'boxjenk4.py'*.

(vi) After selecting a 'Method' under 'Box Jenkins based moving average' select '**%Data-points(test set)**' (e.g., 20) and '**Seed value**' (e.g., 3). Press **'Generate subtrain-test-validation sets'.**

Files to be generated:

- The training set file with starting descriptors: *_tr.csv
- The test set file with starting descriptors: *_ts.csv
- The sub-training set file with deviation descriptors: *_strbj.csv
- The sub-training set file with deviation descriptors: *_tsbj.csv
- The sub-training set file with deviation descriptors: *_vdbj.csv


QSAR-Co-X (Module-1)
Data preparation
Linear model development
Screening/validation
Browse
Dataset division techniques
Generate train-validation sets
Box Jenkins based moving average
Generate subtrain-test-validation sets

Select Data R/Downloads/qsarcoextension/demonstration/Data3.csv

Number of conditions 2

● Predefined ○ Random Division ○ KMCA

%Data-points(validation set) 20

Seed value 2

Number of clusters 5

□ Method-1 □ Method-2 □ Method-3 ☑ Method-4

% data-points in test set 20

Seed value 3

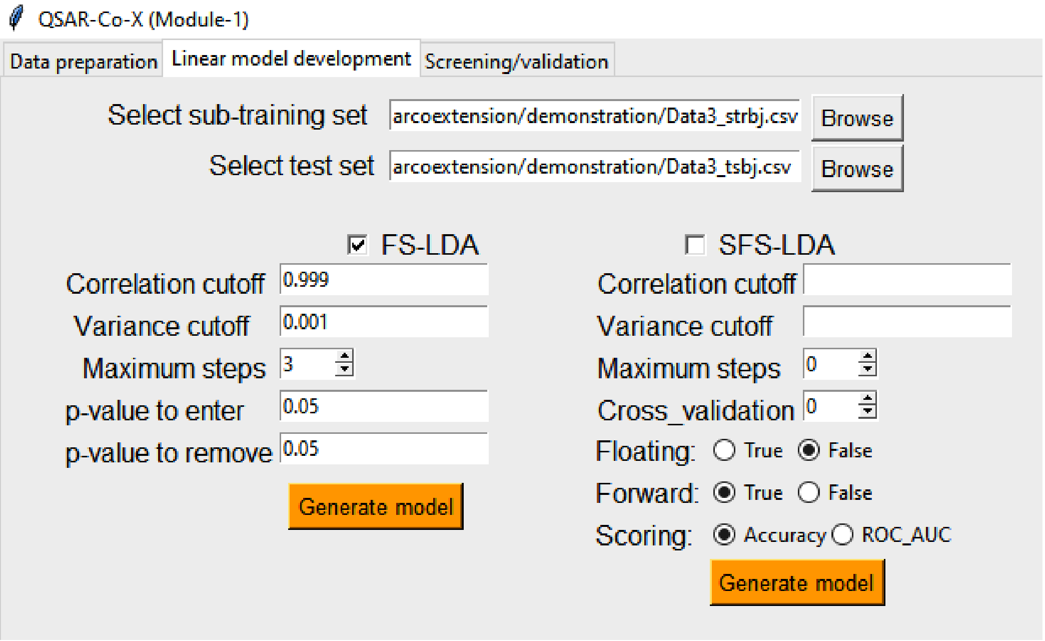
Use Data3.csv if the data has predefined training and validation sets. With other data division techniques remove the last column named 'Set'.

(B) Linear model development:

The user can now develop the model either with FS-LDA and/or SFS-LDA.

(i) Go to Tab2 'Model development', browse sub-training set (***_strbj.csv**) and test set (***_tsbj.csv**) files from the options **'Select sub-training set'** and **'Select test set'** options, respectively.

(ii) For **'FS-LDA'** click on it (**Note:** *Do not click both on FS-LDA and SFS-LDA simultaneously*) and mention the options for model development (e.g., Correlation cut-off:0.999, variance cut-off: 0.001, maximum steps:3, p-value to enter:0.05, p-value to remove:0.05). Press **'Generate model'** to complete.



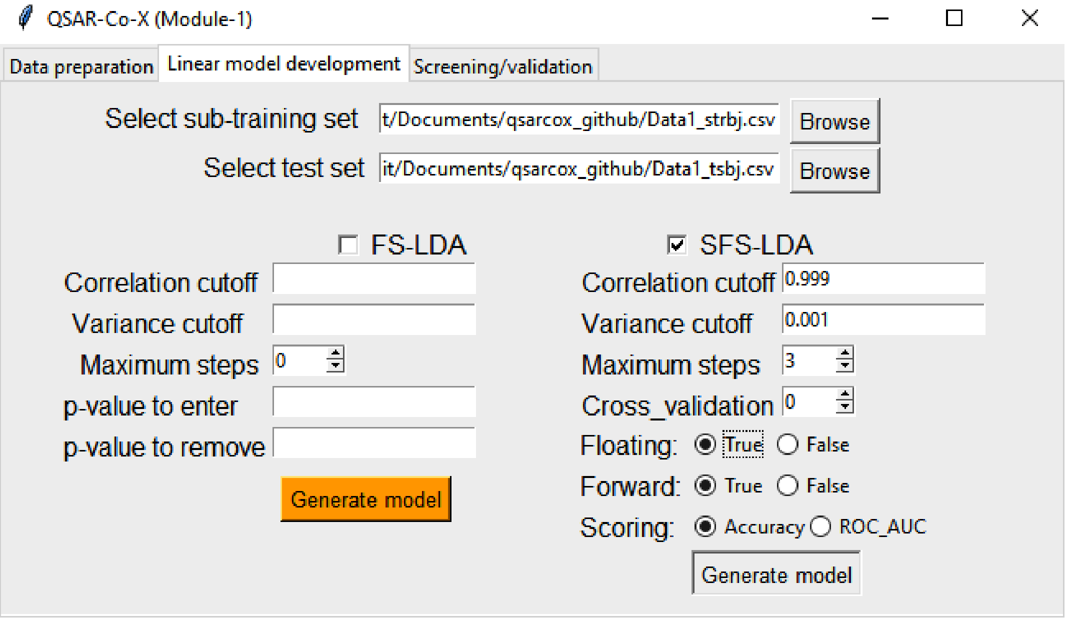
Select sub-training set: arcoextension/demonstration/Data3_strbj.csv

Select test set: arcoextension/demonstration/Data3_tsbj.csv

☑ FS-LDA
- Correlation cutoff: 0.999
- Variance cutoff: 0.001
- Maximum steps: 3
- p-value to enter: 0.05
- p-value to remove: 0.05

□ SFS-LDA
- Correlation cutoff:
- Variance cutoff:
- Maximum steps: 0
- Cross_validation: 0
- Floating: ○ True ● False
- Forward: ● True ○ False
- Scoring: ● Accuracy ○ ROC_AUC

(iii) Similarly mention the options for SFS-LDA and press **‘General model’**.



Select sub-training set: t/Documents/qsarcox_github/Data1_strbj.csv

Select test set: it/Documents/qsarcox_github/Data1_tsbj.csv

□ FS-LDA
- Correlation cutoff:
- Variance cutoff:
- Maximum steps: 0
- p-value to enter:
- p-value to remove:

☑ SFS-LDA
- Correlation cutoff: 0.999
- Variance cutoff: 0.001
- Maximum steps: 3
- Cross_validation: 0
- Floating: ● True ○ False
- Forward: ● True ○ False
- Scoring: ● Accuracy ○ ROC_AUC

Files to be generated:

(a) A text file (as *_strbj_fslda.txt / *_strbj_sfslda.txt) with model descriptors, their coefficients, statistical results of the sub-training and the test sets.

(b) A .csv file (as *_strbj_pred.csv) containing the predicted values of sub-training and the test set samples (i.e., as ‘Pred’), prior probabilities (i.e. ‘%Prob(-1)’ and ‘%Prob(+1)’ and outlier information (estimated by Standardization approach[15]).

(c) Another .csv file (as *_strbj_corr.csv), which depicts the intercorrelation among the descriptors of the generated model.

(C) Validation/Screening + *Yc*-randomization

(i) For validation of the model go to Tab3: **‘Screening/validation’**. Open the training set result file (i.e., ***_strbj_pred.csv***) in the option **‘Open training set result file’** and mention the number of descriptors in the model (e.g., 3) from the option **‘Number of descriptors’**. Then select the validation/screening set (e.g., (***_vdbj.csv***) from the option **‘Select validation/screening set’**. Clicking on **‘Solution’** will save the validation set results.

(ii) **Note:** The difference between the ‘validation set’ and ‘screening set’ is that the lack of dependent parameter column in the input file. For example, simply removal of the ‘*BEq(cr)*’ column from **‘Data1_vdbj/Data2_vdbj /Data3_vdbj.csv’** will make a screening set.

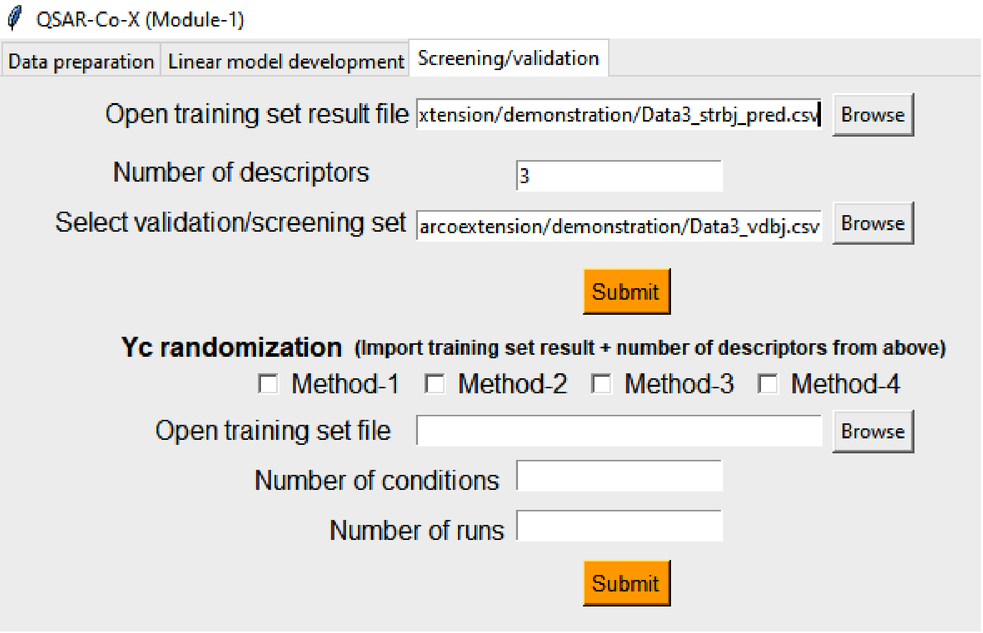
Files to be generated:

(a) A text file (as *_vdbj_pred.txt) with statistical results of the validation set.

(b) A .csv file (as *_vdbj_pred.csv) file containing the predicted values of validation set samples (i.e., as ‘Pred’), prior probabilities (i.e. ‘%Prob(-1)’ and ‘%Prob(+1)’ and outlier information (estimated by Standardization approach[15]).

(c) ROC plots of sub-training, test and validation sets as *_vdbj_ROC.png

(d) For screening set, only a .csv file (as *_scpred.csv) file will be generated.



Open training set result file: xtension/demonstration/Data3_strbj_pred.csv

Number of descriptors: 3

Select validation/screening set: arcoextension/demonstration/Data3_vdbj.csv

☐ Method-1 ☐ Method-2 ☐ Method-3 ☐ Method-4

Open training set file:

Number of conditions:

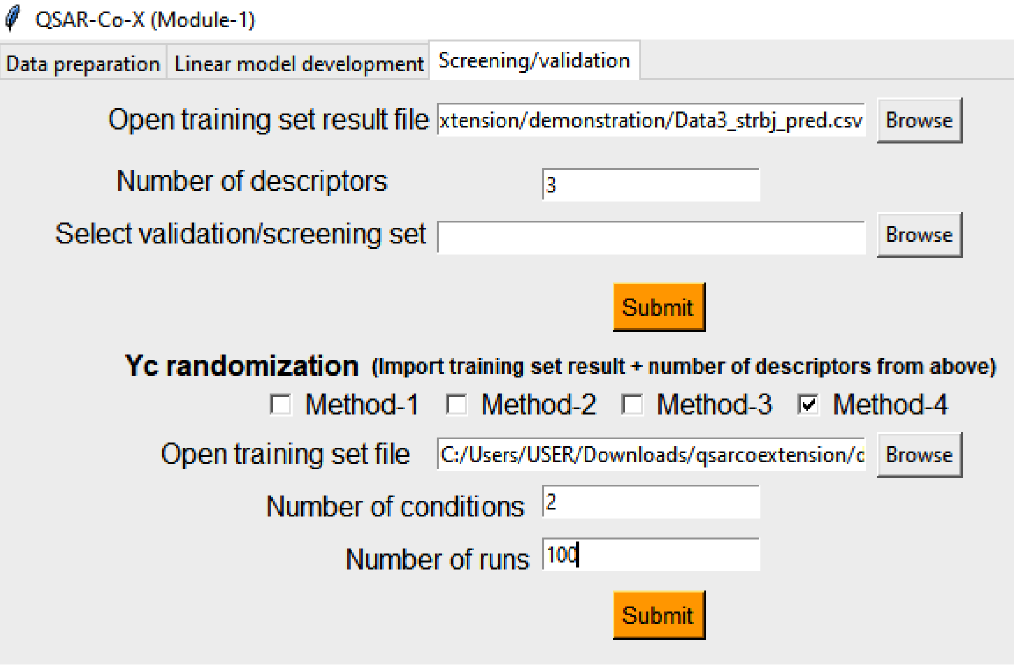
Number of runs:

(i) For *Yc*-randomization, first import training set result file (i.e., ***_strbj_pred.csv***), training set file (i.e., ***_tr.csv***), mention number of experimental conditions (e.g., 2) and number of randomized runs (e.g., 100).

(ii) #Note: Yc randomization is time consuming if the training set file (i.e., ***_*tr.csv**) is uploaded with all starting descriptor values. Therefore, for faster calculation, prepare this training set file with only those descriptors which are found in the linear model (of-course in modified forms).

Files to be generated:

(a) The result will be saved as '*_ycresult.txt'. The average Wilk's λ and average accuracy value of the randomized models will be found in here.



QSAR-Co-X (Module-1)

Data preparation | Linear model development | Screening/validation

Open training set result file: xtension/demonstration/Data3_strbj_pred.csv

Number of descriptors: 3

Select validation/screening set:

Yc randomization **(Import training set result + number of descriptors from above)**

☐ Method-1 ☐ Method-2 ☐ Method-3 ☑ Method-4

Open training set file: C:/Users/USER/Downloads/qsarcoextension/c

Number of conditions: 2

Number of runs: 100

Module-2

This module helps in developing the non-linear models using six machine learning algorithms with hyperparameter tuning. The machine learning techniques are as follows:

(a) k-nearest neighborhood (kNN)[23]

(b) Bernoulli Naïve Bayes (NB) classifier[24]

(c) Support vector classifier (SVC)[25]

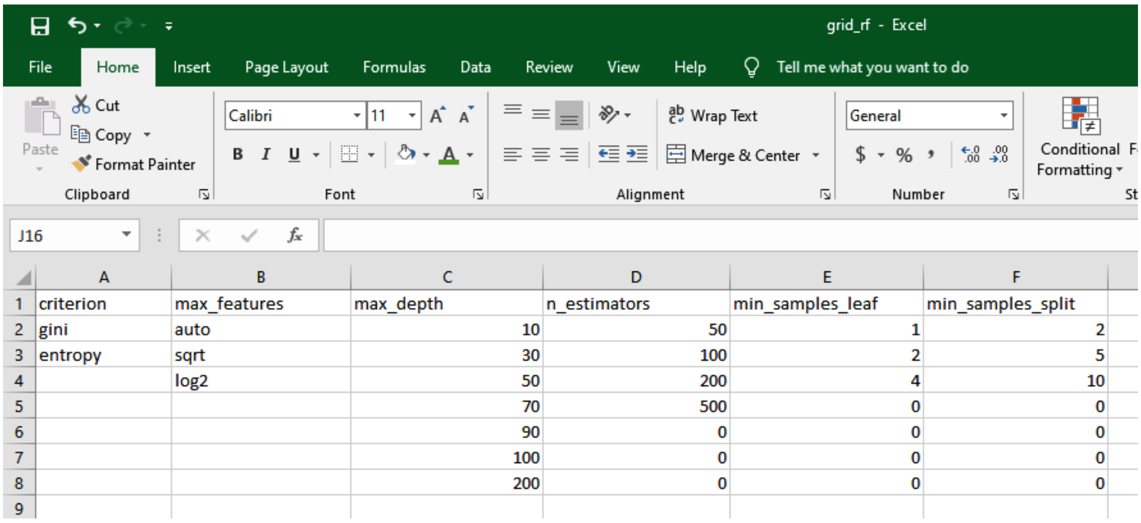
(d) Random forest (RF)[26] classifier

(e) Gradient boosting (GB)[27]

(f) Multilayer perception (MLP)[28].

In this Module-2, the user needs to upload a parameter file (.csv format) corresponding to specific machine learning technique using the option 'Select parameter file'. In this file, the name of the parameters and their values should be mentioned in a systematic order. The values of these parameters will be optimized in Module-2 by cross-validation

(CV) based grid search techniques. Six such parameter files (named grid_knn.csv, grid_nb.csv, grid_svc.csv, grid_mlp.csv, grid_rf.csv and grid_gb.csv) are provided based on the values provided below in Table 1. Below a snapshot of grid_rf.csv is provided.



	A	B	C	D	E	F
1	criterion	max_features	max_depth	n_estimators	min_samples_leaf	min_samples_split
2	gini	auto	10	50	1	2
3	entropy	sqrt	30	100	2	5
4		log2	50	200	4	10
5			70	500	0	0
6			90	0	0	0
7			100	0	0	0
8			200	0	0	0
9						

Table 1. Hyper-parameters tuning options available in *QSAR-Co-X*

Technique	Parameters tuning*	References
RF	**Bootstrap:** True/ False (this option is automatically selected)	For explanations of parameters and their importance, please visit: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
	Criterion: Gini (for Gini impurity), Entropy (for information gain) **Maximum depth:** 10, 30, 50, 70, 90, 100, 200 **Maximum features:** Auto, Sqrt, Log2 **Minimum samples leaf:** 1, 2, 4 **Minimum samples split:** 2, 5, 10 **Number of estimators:** 50, 100, 200,500	
***k*NN**	**Number of neighbours:** 1-50	For explanations of parameters and their importance, please visit: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
	Weight options: Uniform (for uniform weights), Distance (i.e., weight points by the inverse of their distance)	

	Algorithms (used to compute the nearest neighbours): Auto, Ball tree (i.e., BallTree), KD_tree (i.e., KDTree), Brute (i.e., brute-force serch)	
Bernoulli NB	**Alpha** (i.e., smoothing parameter): 1, 0.5, 0.1 **Fit_prior**: True, False (this option is automatically selected)	For explanations of parameters and their importance, please visit: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB.html
SVC	**C**(i.e., regularization parameter): 0.1, 1, 10, 100, 1000 **Gamma**: 1, 0.1, 0.01, 0.001 **Kernel**: RBF, Linear, Poly, Sigmoid	For explanations of parameters and their importance, please visit: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
MLP	**Hidden layer sizes:** To be specified by the user **Activation:** Identity, Logistic, Tanh, Relu **Solver:** SGD, Adam **Alpha:** 0.0001, 0.001, 0.01, 1 **Learning rate:** Constant, Adaptive, Invscaling	For explanations of parameters and their importance, please visit: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html
GB	**Loss:** deviance, exponential **Learning rate**: 0.01, 0.05, 0.1, 0.2 **Min samples split**: 0.1,0.2,0.3,0.4,0.5 **Minimum samples leaf**: 0.1,0.2,0.3,0.4,0.5 **Maximum depth**: 3,5,8 **Maximum features**: Log2, Sqrt **Criterion:** Friedman MSE, MAE **Subsample:** 0.5, 0.6, 0.8 **Number of estimators:** 50,100,200,300	For explanations of parameters and their importance, please visit: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

The user may change the options for hyperparameter tuning by just altering the values of the parameters provided in these parameter files. Alternatively, new parameters may also be introduced in these files with their values. The weblinks provided in Table 1 should be consulted to understand the importance of the parameters for each machine learning method.

Steps of model development:

(i) Browse and upload the sub-training set file (***_strbj.csv** generated from Module 1) from **'Select sub-training set'** option.

(ii) Browse and upload the test set file **(*_tsbj.csv** generated from Module 1) from **'Select test set'** option.

(iii) Upload the .csv file with parameter names and their values from the option 'Select parameter file'. As an example, import 'grid_rf.csv' for RF.

(iv) Click the machine learning method listed under **'Method:'** after selecting the parameter file. As an example, after importing 'grid_rf.csv', click on 'Random forest'.

(v) The option 'Hidden layers (for MLP only)', will be activated if the option 'Multilayer perception' is chosen. By default, the hidden layer size (100,) will be selected. The user may select other hidden layers by altering the values placed after this option.

(vi) The option 'Random state (for MLP/RF/GB only)' is activated if the user selects 'Multilayer perception', 'Random forest' and 'Gradient boosting'. For these three machine learning techniques, 'random_state' is used as an input parameter. Note that if the option 'None' is selected slightly different models may be generated since no random state is specified. By providing a 'number' for 'random_state', the same model will be generated with repeated runs.

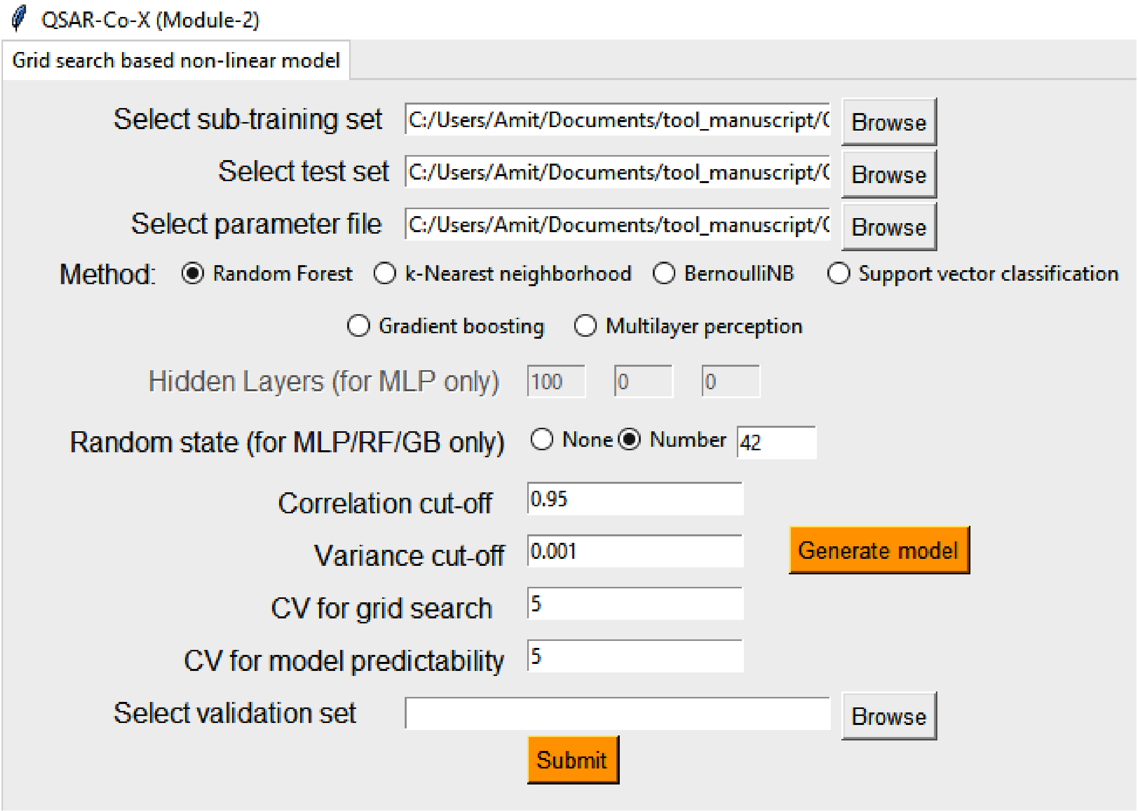
(vii) Mention **'Correlation cut-off'** (for example, 0.95).

(viii) Mention **'Variance cut-off'** (for example 0.001).

(ix) Mention cross-validation (CV) for grid search in **'CV for grid search'** option. For example, for 5-fold CV, put 5, for 10-fold CV put 10.

(x) Mention cross-validation (CV) for determining model predictability of sub-training set in **'CV for model predictability'** option. For example, for 5-fold CV, put 5, for 10-fold CV put 10.

(xi) Click on **'Generate model'** option for model development.



QSAR-Co-X (Module-2)

Grid search based non-linear model

Select sub-training set: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Select test set: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Select parameter file: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Method: ● Random Forest ○ k-Nearest neighborhood ○ BernoulliNB ○ Support vector classification ○ Gradient boosting ○ Multilayer perception

Hidden Layers (for MLP only): 100 | 0 | 0

Random state (for MLP/RF/GB only): ○ None ● Number 42

Correlation cut-off: 0.95

Variance cut-off: 0.001

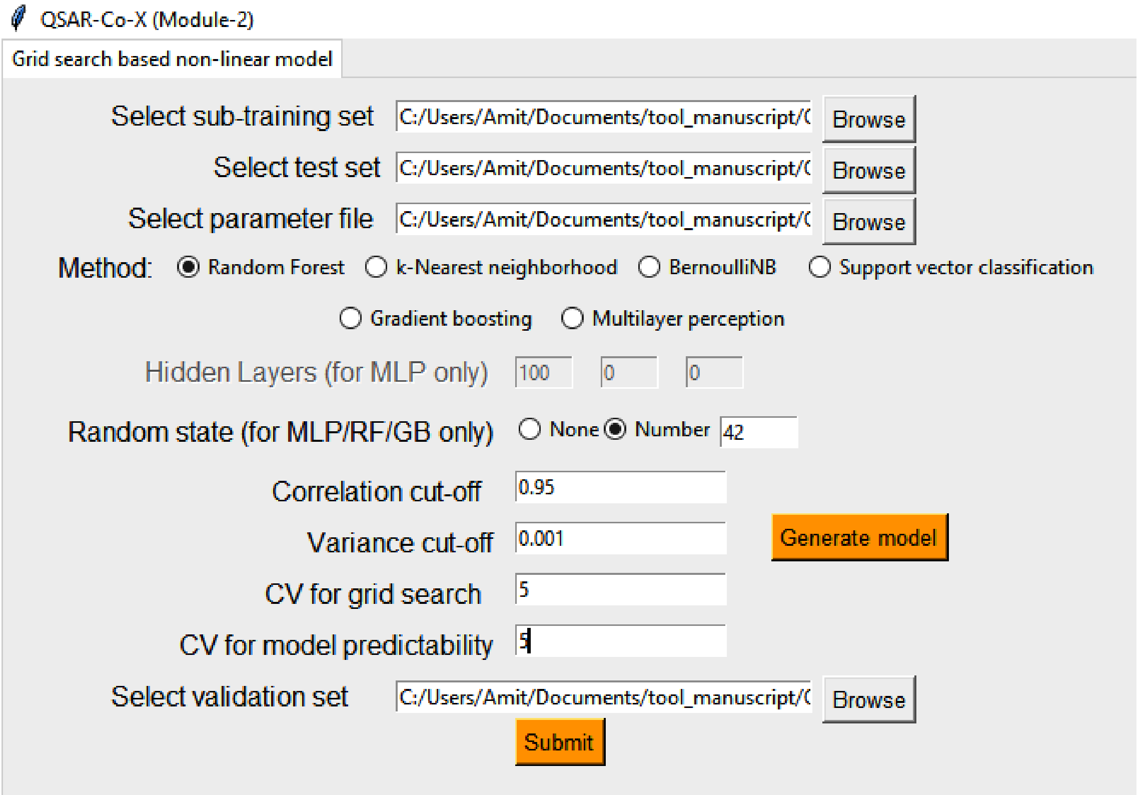
CV for grid search: 5

CV for model predictability: 5

Select validation set: [Browse]

(xii) After model generation, browse and open the validation set file (***_vdbj.csv** generated from Module 1) from **'Select validation set'** option.

(xiii) Click on **'Submit'** option.



QSAR-Co-X (Module-2)

Grid search based non-linear model

Select sub-training set: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Select test set: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Select parameter file: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Method: ● Random Forest ○ k-Nearest neighborhood ○ BernoulliNB ○ Support vector classification ○ Gradient boosting ○ Multilayer perception

Hidden Layers (for MLP only): 100 | 0 | 0

Random state (for MLP/RF/GB only): ○ None ● Number 42

Correlation cut-off: 0.95

Variance cut-off: 0.001

CV for grid search: 5

CV for model predictability: 5

Select validation set: C:/Users/Amit/Documents/tool_manuscript/([Browse]

Files to be generated:

(a) A text file (as *g_(Method name)_tr.txt) with training set result containing the best (i.e., optimized) estimator, n-fold cross validation statistics of sub-training set and the test set results.

(b) A .csv file (as *g_(Method name)_tspred.csv) containing the predicted values of test set samples (i.e., as 'Pred'), prior probabilities (i.e. '%Prob(-1)' and '%Prob(+1)' and outlier information (estimated by confidence estimation approach[16, 17]).

(c) A text file (as *g_(Method name)_vd.txt) with statistical results of the validation set.

(d) A .csv file (as *g_(Method name)_vdpred.csv) containing the predicted values of validation set samples (i.e., as 'Pred'), prior probabilities (i.e. '%Prob(-1)' and '%Prob(+1)' and outlier information (estimated by confidence estimation approach[16, 17]).

(e) ROC plots of test set and validation set as *_(Method name)_ROC.png.

(f) For screening set, only a .csv file (as *_scpred.csv) file will be generated.

Module-3

This module helps in developing the non-linear models using six machine learning algorithms (mentioned above in Module 2) with user-specific parameters. The machine learning techniques are as follows:

Steps of model development:

(i) Browse and open the sub-training set file (***_strbj.csv** generated from Module 1) from **'Select sub-training set'** option.

(ii) Browse and open the test set file (***_tsbj.csv** generated from Module 1) from **'Select test set'** option.

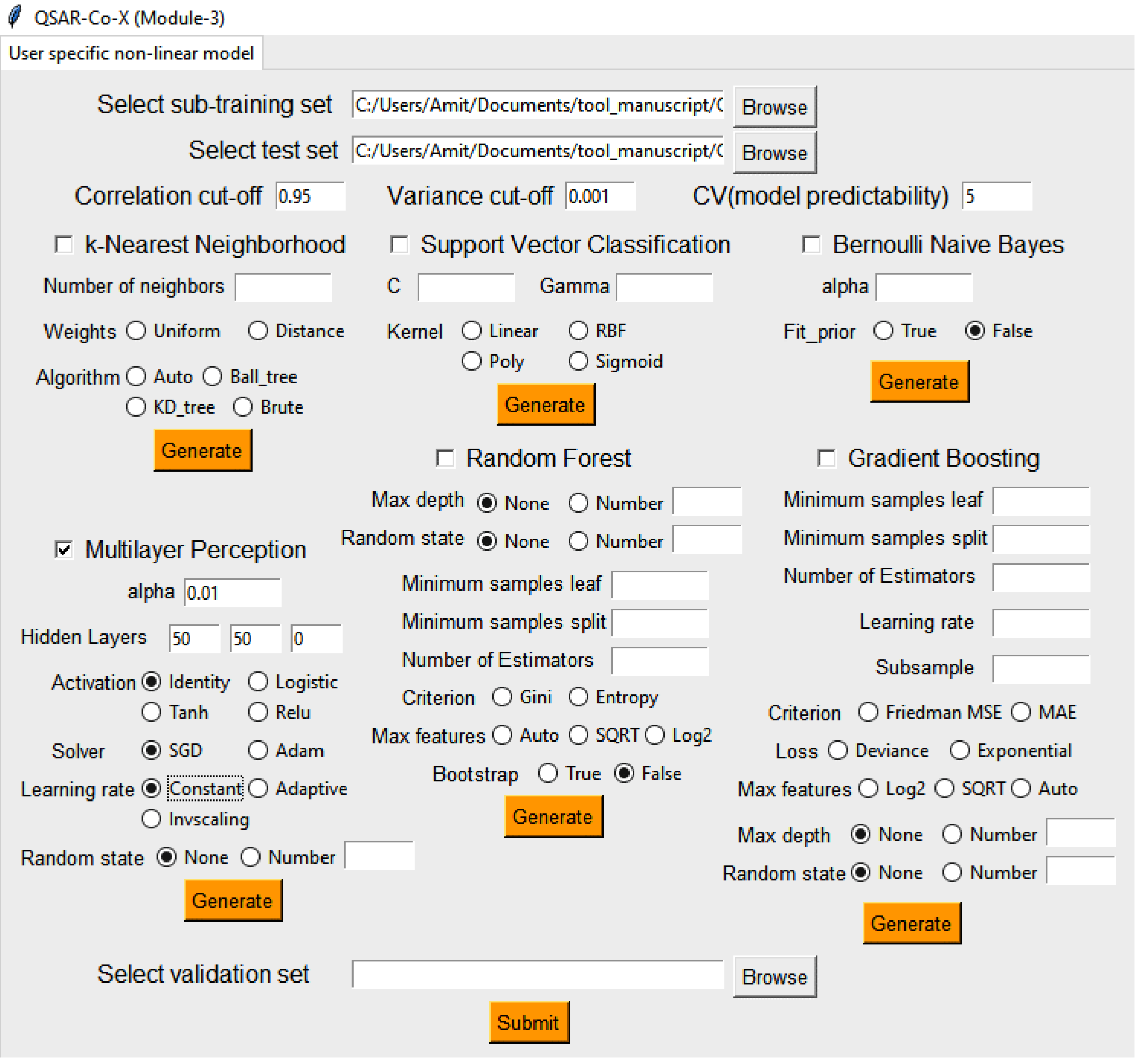
(iii) Mention **'Correlation cut-off'** (for example, 0.95).

(iv) Mention **'Variance cut-off'** (for example 0.001).

(v) Mention cross-validation (CV) for determining model predictability of sub-training set in **'CV (model predictability)'** option. For example, for 5-fold CV, put 5, for 10-fold CV put 10.

(vi) Click on the specific machine learning method (do not click more than one option at once) and choose the parameters. For the parameters, user may check Table 1.

(vii) Click on **'Generate'** option for model development.



QSAR-Co-X (Module-3)

User specific non-linear model

Select sub-training set C:/Users/Amit/Documents/tool_manuscript/(Browse

Select test set C:/Users/Amit/Documents/tool_manuscript/(Browse

Correlation cut-off 0.95 Variance cut-off 0.001 CV(model predictability) 5

□ k-Nearest Neighborhood

Number of neighbors

Weights ○ Uniform ○ Distance

Algorithm ○ Auto ○ Ball_tree ○ KD_tree ○ Brute

Generate

☑ Multilayer Perception

alpha 0.01

Hidden Layers 50 50 0

Activation ● Identity ○ Logistic ○ Tanh ○ Relu

Solver ● SGD ○ Adam

Learning rate ● Constant ○ Adaptive ○ Invscaling

Random state ● None ○ Number

Generate

□ Support Vector Classification

C Gamma

Kernel ○ Linear ○ RBF ○ Poly ○ Sigmoid

Generate

□ Random Forest

Max depth ● None ○ Number

Random state ● None ○ Number

Minimum samples leaf

Minimum samples split

Number of Estimators

Criterion ○ Gini ○ Entropy

Max features ○ Auto ○ SQRT ○ Log2

Bootstrap ○ True ● False

Generate

□ Bernoulli Naive Bayes

alpha

Fit_prior ○ True ● False

Generate

□ Gradient Boosting

Minimum samples leaf

Minimum samples split

Number of Estimators

Learning rate

Subsample

Criterion ○ Friedman MSE ○ MAE

Loss ○ Deviance ○ Exponential

Max features ○ Log2 ○ SQRT ○ Auto

Max depth ● None ○ Number

Random state ● None ○ Number

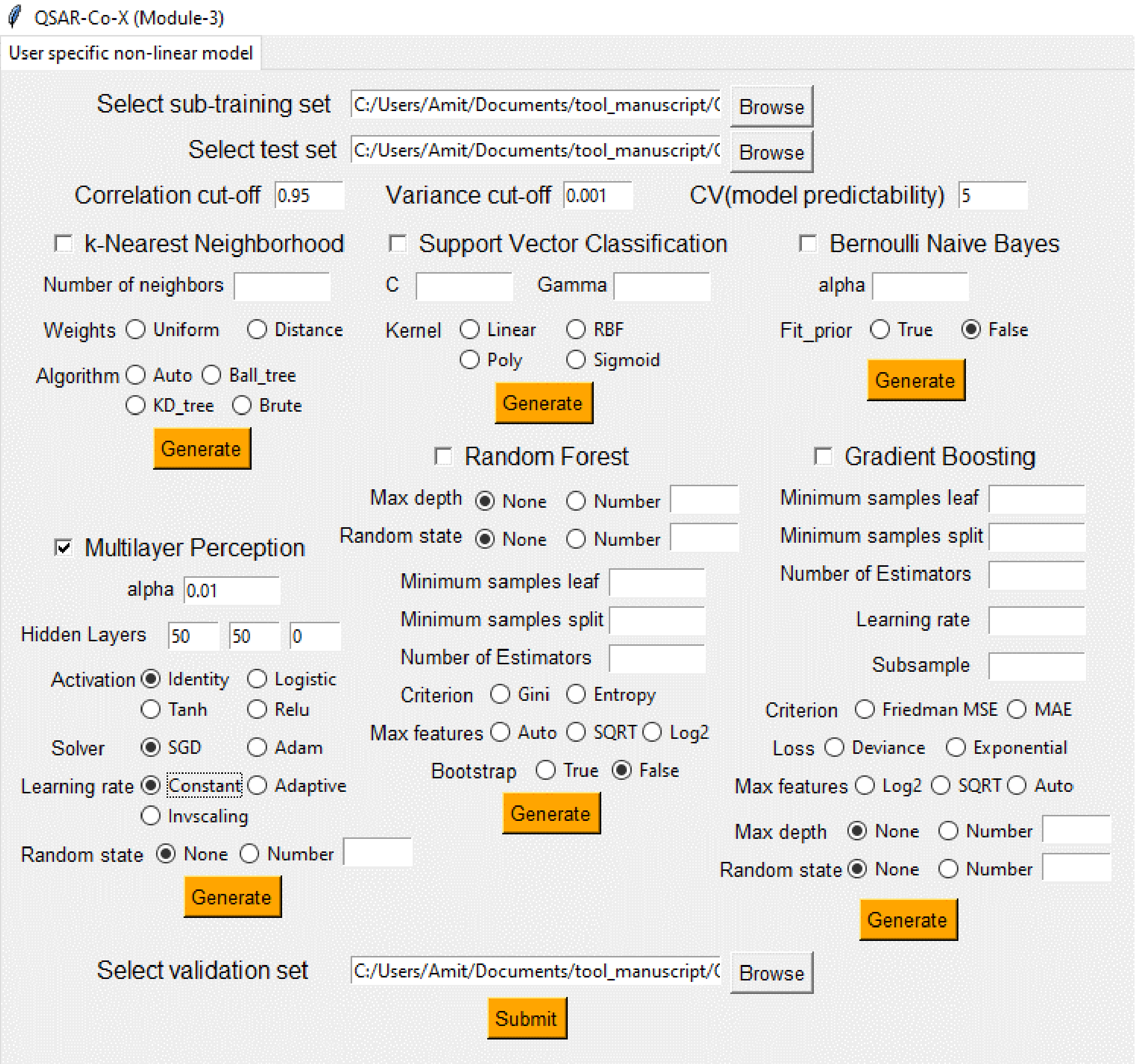
Generate

Select validation set Browse

Submit

(viii) After model generation, browse and open the validation set file (***_vdbj.csv** generated from Module 1) from **'Select validation set'** option.

(ix) Click on **'Submit'** option.



QSAR-Co-X (Module-3)

User specific non-linear model

Select sub-training set C:/Users/Amit/Documents/tool_manuscript/(

Select test set C:/Users/Amit/Documents/tool_manuscript/(

Correlation cut-off 0.95 Variance cut-off 0.001 CV(model predictability) 5

☐ k-Nearest Neighborhood
Number of neighbors
Weights ○ Uniform ○ Distance
Algorithm ○ Auto ○ Ball_tree ○ KD_tree ○ Brute

☐ Support Vector Classification
C Gamma
Kernel ○ Linear ○ RBF ○ Poly ○ Sigmoid

☐ Bernoulli Naive Bayes
alpha
Fit_prior ○ True ● False

☐ Random Forest
Max depth ● None ○ Number
Random state ● None ○ Number
Minimum samples leaf
Minimum samples split
Number of Estimators
Criterion ○ Gini ○ Entropy
Max features ○ Auto ○ SQRT ○ Log2
Bootstrap ○ True ● False

☐ Gradient Boosting
Minimum samples leaf
Minimum samples split
Number of Estimators
Learning rate
Subsample
Criterion ○ Friedman MSE ○ MAE
Loss ○ Deviance ○ Exponential
Max features ○ Log2 ○ SQRT ○ Auto
Max depth ● None ○ Number
Random state ● None ○ Number

☑ Multilayer Perception
alpha 0.01
Hidden Layers 50 50 0
Activation ● Identity ○ Logistic ○ Tanh ○ Relu
Solver ● SGD ○ Adam
Learning rate ● Constant ○ Adaptive ○ Invscaling
Random state ● None ○ Number

Select validation set C:/Users/Amit/Documents/tool_manuscript/(

Files generated:

(a) A text file (as *(Method name)_tr.txt) with training set result containing the estimator, n-fold cross validation statistics of sub-training set and the test set results.

(b) A .csv file (as *(Method name)_tspred.csv) containing the predicted values of test set samples (i.e., as ‘Pred’), prior probabilities (i.e. ‘%Prob(-1)’ and ‘%Prob(+1)’ and outlier information (estimated by confidence estimation approach[16, 17]).

(c) A text file (as *(Method name)_vd.txt) with statistical results of the validation set.

(d) A .csv file (as *g_(Method name)_vdpred.csv) containing the predicted values of validation set samples (i.e., as ‘Pred’), prior probabilities (i.e. ‘%Prob(-1)’ and ‘%Prob(+1)’ and outlier information (estimated by confidence estimation approach[16, 17]).

Module-4

Note that this module produces the **'Condition-wise-prediction'** for the test set and the validation set.

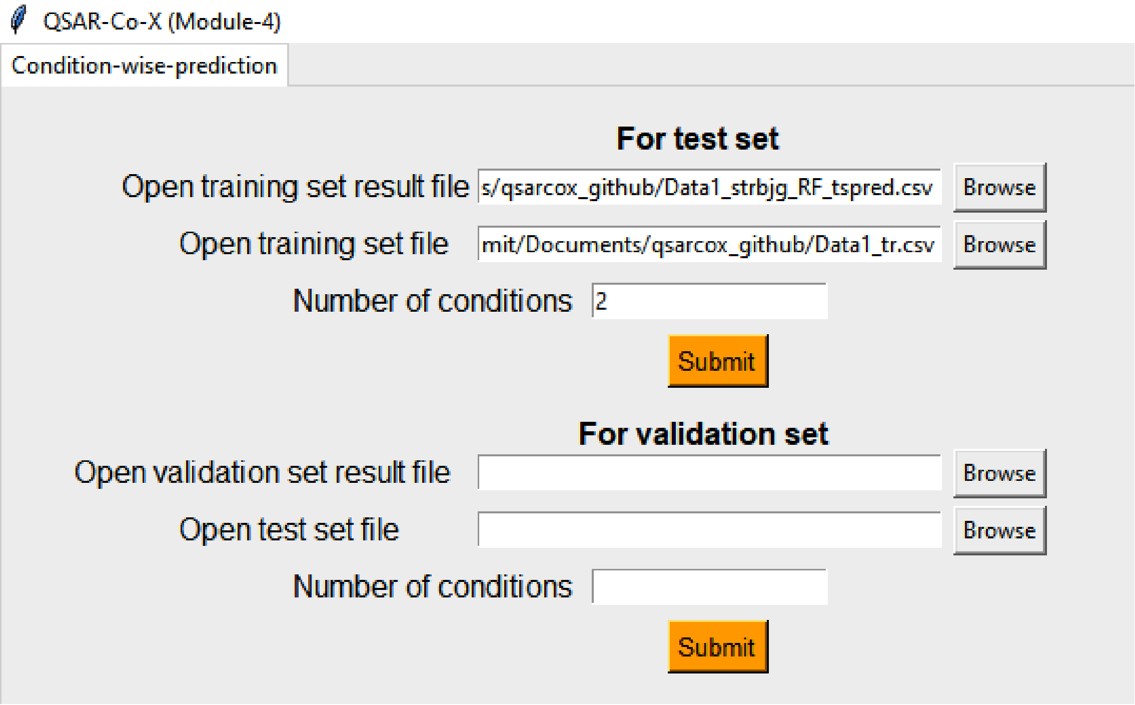
Follow the steps mentioned below under the heading **'For test set'**:

(i) **'Open training set result file'** which is saved as ***_strbj_pred.csv*** (generated by Module-1) or as ***g_(Method name)_tspred.csv*** (if generated by Module-2) or ***(Method name)_tspred.csv*** (if generated by Module-3).

(ii) **'Open training set file'** which is saved as ***_tr.csv*** (generated with Module 1)

(iii) Select the **'Number of conditions'** (for example 2 if there are 2 experimental elements

(iv) Press **'Submit'**



QSAR-Co-X (Module-4)

Condition-wise-prediction

For test set

Open training set result file: s/qsarcox_github/Data1_strbjg_RF_tspred.csv

Open training set file: mit/Documents/qsarcox_github/Data1_tr.csv

Number of conditions: 2

For validation set

Open validation set result file:

Open test set file:

Number of conditions:

Files to be generated:

(a) A .csv file will be generated as *strbj_pred_cond.csv, where against each condition of the test set, the number of instances and accuracy will be depicted.

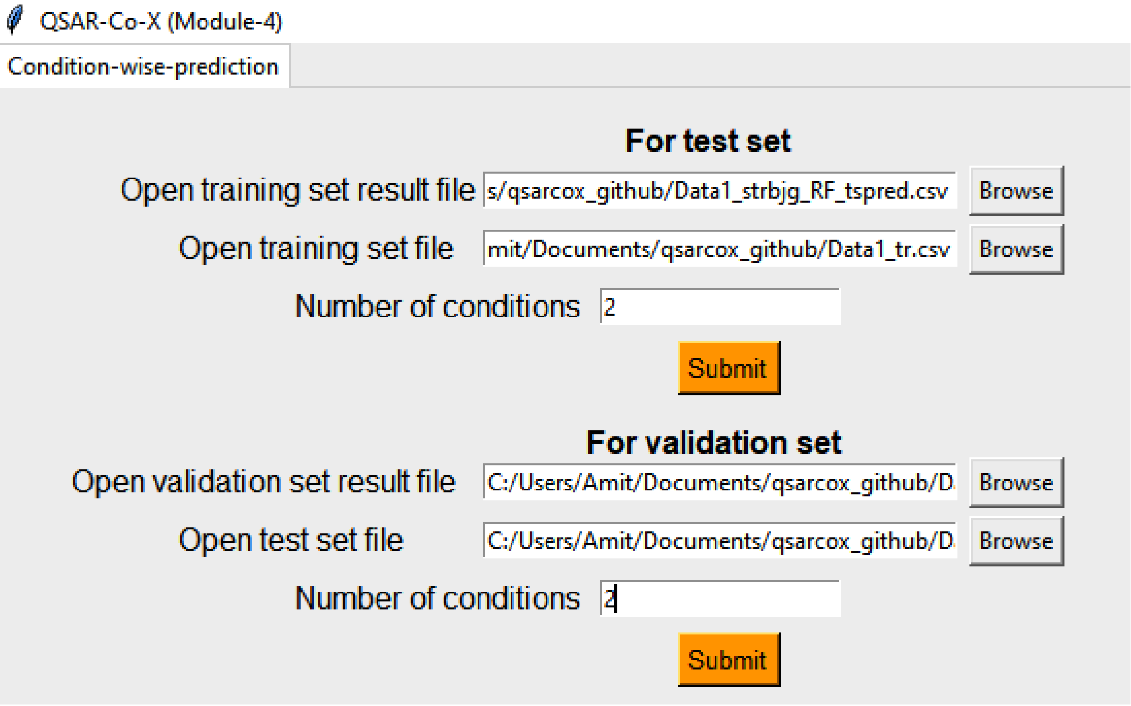
Follow the steps mentioned below under the heading **'For validation set'**:

(i) **'Open validation set result file'** which is saved as ***_vdbj_pred.csv*** (generated by Module-1) or as ***g_(Method name)_vdpred.csv*** (if generated by Module-2) or ***(Method name)_vdpred.csv*** (if generated by Module-3).

(ii) **'Open test set file'** which is saved as ***_ts.csv*** (generated with Module 1)

(iii) Select the **'Number of conditions'** (for example 2 if there are 2 experimental elements

(iv) Press **'Submit'.**



QSAR-Co-X (Module-4)

Condition-wise-prediction

For test set

Open training set result file: s/qsarcox_github/Data1_strbjg_RF_tspred.csv

Open training set file: mit/Documents/qsarcox_github/Data1_tr.csv

Number of conditions: 2

For validation set

Open validation set result file: C:/Users/Amit/Documents/qsarcox_github/D

Open test set file: C:/Users/Amit/Documents/qsarcox_github/D

Number of conditions: 2

Files to be generated:

(a) A .csv file will be generated as *vdbj_pred_cond.csv, where against each condition of the test set, the number of instances and accuracy will be depicted.

References

1. Ambure P, Halder AK, Diaz HG, Cordeiro MNDS (2019) QSAR-Co: An Open Source Software for Developing Robust Multitasking or Multitarget Classification-Based QSAR Models. Journal of Chemical Information and Modeling 59:2538-2544.
2. Halder AK, Natalia M, Cordeiro DS (2019) Probing the Environmental Toxicity of Deep Eutectic Solvents and Their Components: An In Silico Modeling Approach. Acs Sustainable Chemistry & Engineering 7:10649-10660.
3. Halder AK, Giri AK, Cordeiro M (2019) Multi-Target Chemometric Modelling, Fragment Analysis and Virtual Screening with ERK Inhibitors as Potential Anticancer Agents. Molecules 24.
4. Halder AK, Cordeiro MNDS (2019) Development of Multi-Target Chemometric Models for the Inhibition of Class I PI3K Enzyme Isoforms: A Case Study Using QSAR-Co Tool. International Journal of Molecular Sciences 20.
5. Speck-Planche A, Scotti MT (2019) BET bromodomain inhibitors: fragment-based in silico design using multi-target QSAR models. Molecular Diversity 23:555-572.
6. Speck-Planche A, Cordeiro MNDS (2017) Fragment-based in silico modeling of multi-target inhibitors against breast cancer-related proteins (vol 21, pg 511, 2017). Molecular Diversity 21.
7. Speck-Planche A, Cordeiro MNDS (2017) De novo computational design of compounds virtually displaying potent antibacterial activity and desirable in vitro ADMET profiles. Medicinal Chemistry Research 26:2345-2356.
8. Speck-Planche A. 2020. Multi-scale QSAR Approach for Simultaneous Modeling of Ecotoxic Effects of Pesticides, p 639-660. *In* Roy K (ed), Ecotoxicological QSARs doi:10.1007/978-1-0716-0150-1_26. Springer US, New York, NY.
9. Speck-Planche A (2019) Multicellular Target QSAR Model for Simultaneous Prediction and Design of Anti-Pancreatic Cancer Agents. Acs Omega 4:3122-3132.
10. Speck-Planche A (2018) Combining Ensemble Learning with a Fragment-Based Topological Approach To Generate New Molecular Diversity in Drug Discovery: In Silico Design of Hsp90 Inhibitors. Acs Omega 3:14704-14716.
11. Kleandrova VV, Ruso JM, Speck-Planche A, Cordeiro MNDS (2016) Enabling the Discovery and Virtual Screening of Potent and Safe Antimicrobial Peptides. Simultaneous Prediction of Antibacterial Activity and Cytotoxicity. Acs Combinatorial Science 18:490-498.
12. Hao JG, Ho TK (2019) Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language. Journal of Educational and Behavioral Statistics 44:348-361.
13. 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 (2011) Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12:2825-2830.
14. Menzies T, Kocagüneli E, Minku L, Peters F, Turhan B. 2015. Chapter 22 - Complexity: Using Assemblies of Multiple Models, p 291-304. *In* Menzies T, Kocagüneli E, Minku L, Peters F, Turhan B (ed), Sharing Data and Models in Software Engineering doi:https://doi.org/10.1016/B978-0-12-417295-1.00022-9. Morgan Kaufmann, Boston.
15. Roy K, Kar S, Ambure P (2015) On a simple approach for determining applicability domain of QSAR models. Chemometrics and Intelligent Laboratory Systems 145:22-29.
16. Ambure P, Bhat J, Puzyn T, Roy K (2019) Identifying natural compounds as multi-target-directed ligands against Alzheimer's disease: an in silico approach. Journal of Biomolecular Structure & Dynamics 37:1282-1306.
17. Mathea M, Klingspohn W, Baumann K (2016) Chemoinformatic Classification Methods and their Applicability Domain. Molecular Informatics 35:160-180.

18. Van Rossum G, & Drake, F. L. (2009) Python 3 Reference Manual. Scotts Valley, CA: CreateSpace.
19. Hunter JD (2007) Matplotlib: A 2D graphics environment. Computing in Science & Engineering 9:90-95.
20. W M Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010).
21. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Polat I, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P, Contributors S (2020) SciPy 1.0: fundamental algorithms for scientific computing in Python (vol 17, pg 261, 2020). Nature Methods 17:352-352.
22. van der Walt S, Colbert SC, Varoquaux G (2011) The NumPy Array: A Structure for Efficient Numerical Computation. Computing in Science & Engineering 13:22-30.
23. Cover TM, Hart PE (1967) Nearest Neighbor Pattern Classification. Ieee Transactions on Information Theory 13:21-+.
24. McCallum A, Nigam K (2001) A Comparison of Event Models for Naive Bayes Text Classification. Work Learn Text Categ 752.
25. Boser BE, Guyon, I. M., & Vapnik, V. N. A training algorithm for optimal margin classifiers. . In Proceedings of the fifth annual workshop on Computational learning theory ACM 144–152.
26. Breiman L (2001) Random forests. Machine Learning 45:5-32.
27. Friedman JH (2001) Greedy function approximation: A gradient boosting machine. Annals of Statistics 29:1189-1232.
28. Huang GB, Babri HA (1998) Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions. IEEE Trans Neural Netw 9:224-9.

Project Leader

Prof. **M. Natália D. S. Cordeiro**
Associate Professor
ncordeir@fc.up.pt
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.

External Libraries used

The current tool utilizes some well-known Python based libraries such as NumPy [22], SciPy[21], Pandas[20], Matplotlib[19], Tkinter (https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/index.html), and Scikit-learn [12, 13], MLxtend (http://rasbt.github.io/mlxtend/).