DELIVERABLE REPORT D4.6

Tools for generating QMRF and QPRF reports

GRANT AGREEMENT:	604134
ACRONYM:	eNanoMapper
NAME:	eNanoMapper - A Database and Ontology Framework for Nanomaterials Design and Safety Assessment

PROJECT COORDINATOR:	Douglas Connect GmbH
START DATE OF PROJECT; DURATION:	1 February 2014; 36 months
PARTNER(s) RESPONSIBLE FOR THIS DELIVERABLE:	IST
DATE:	07.11.2016
VERSION:	1.0

Call identifier	FP7-NMP-2013-SMALL-7
Document Type	Deliverable Report
WP/Task	WP4/T4.6
Document ID	eNanoMapper D4.6 - Tools for generating QMRF and QPRF reports
Status	Final

Partner Organisations	● Douglas Connect, GmbH (DC) ● National Technical University of Athens (NTUA) ● In Silico Toxicology (IST) ● Ideaconsult (IDEA) ● Karolinska Institutet (KI) ● European Bioinformatics Institute (EMBL-EBI) ● Maastricht University (UM) ● Misvik Biology (MB)

Authors	G. Drakakis, C. Chomenidis, G. Tsiliki, P. Doganis, E. Anagnostopoulou, H. Sarimveis, M. Rautenberg, D. Gebele, C. Helma, N. Jeliazkova, V. Jeliazkov Review by Barry Hardy (DC)
Purpose of the Document	*To report on Task 4.6 - Updating and extending OpenTox validation and reporting services, namely the integration of split-, cross- and external validation schemes and the generation of QPRF and QMRF reports*
Document History	Final version: 07.11.2016

TABLE OF CONTENTS

TABLE OF FIGURES

GLOSSARY

Abbreviation / acronym	Description
JQ	*JaqPot Quattro*
UI	*User Interface*
RA	*Read Across*
QSAR	*Quantitative Structure-Activity Relationship*
nQSAR	*nano-QSAR*
eNM	*eNanoMapper*
API	*Application Programming Interface*
CV	*Cross Validation*
QPRF	*QSAR Prediction Reporting Format*
QMRF	*QSAR Model Reporting Format*
DoA	Domain of applicability
CL	Confidence Level
LAZAR	Lazy Structure-Activity Relationships

1. EXECUTIVE SUMMARY

Scientific reports carry significant importance for the straightforward and effective transfer of knowledge, results and ideas. Good practice dictates that reports should be well-structured and concise. This deliverable describes the reporting services for models, predictions and validation tasks that have been integrated within the eNanoMapper (eNM) modelling infrastructure.

Validation services have been added to the Jaqpot Quattro (JQ) modelling platform and the nano-lazar read-across framework developed within WP4 to support eNM modelling activities. Moreover, we have proceeded with the development of reporting services for predictions and models, respectively QPRF and QMRF reports.

Therefore, in this deliverable, we first describe the three validation schemes created, namely training set split, cross- and external validation in detail and demonstrate their functionality both on API and UI levels. We then proceed with the description of the read across functionalities and finally, we present and describe the QPRF and QMRF reporting services.

2. INTRODUCTION

Deliverable 4.6 describes the functionalities and tools that have been developed during the eNanoMapper (eNM) project for validating the statistical and machine learning (ML) algorithms and reporting on the predictions and the models themselves.

Section 3 of this report describes the validation schemes and the respective reporting services made available by JQ and nano-lazar modelling frameworks for both regression and classification algorithms, which compute and provide statistics such as R-squared, R^2 Adjusted, RMSD, F-Value, Standard Error, Accuracy, Precision, Recall, F1-score, Jaccard index, as well as informative figures, wherever applicable.

Section 4 presents the read across methods and services which have been developed and integrated in the nano-lazar and JQ frameworks for filling data gaps and for complementing QSAR modelling tools.

Sections 5 and 6 describe the development of services for the automatic creation of QPRF and QMRF reports respectively, according to OECD guidelines. JQ and nano-lazar reporting services are presented, where QMRF or QPRF fields are automatically filled in, when the appropriate entries are present in the database (https://data.enanomapper.net), but are also fully editable for allowing users to fill in missing information or add descriptions. Section 6 also presents the QMRF Editor 3.0.0, which is a standalone Java application with user friendly installation, uses the eNM ontology and can be used independently to compile QMRF reports for any model.

Full demonstrations of the above processes are provided in this deliverable, with the help of the Swagger API documentation and/or the user interfaces that provide visualization of the web service functionalities.

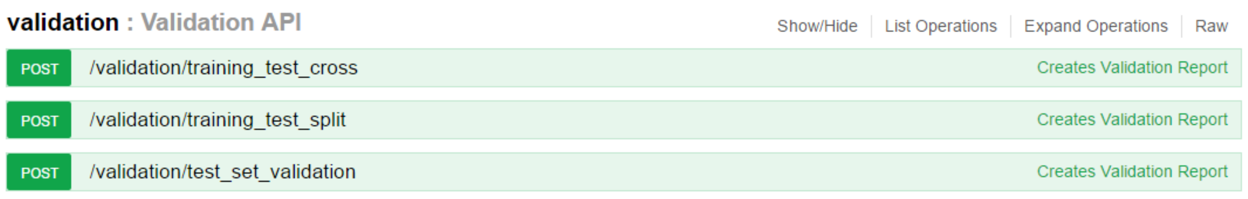
These additions may be considered as extensions to the OpenTox algorithm and modelling API.

3. Validation Schemes And Reporting Services

3.1 Validation Services In The JQ Modelling Framework

Validation schemes have been added to the JQ functionalities in order to provide users with the opportunity to evaluate the accuracy and robustness of their models. In this service, JQ provides information on the algorithm type, number of variables used and accuracy metrics depending on the type of algorithm. Two types of algorithms are complemented by validation services in JQ, namely classification and regression algorithms. For classification, the metrics provided are Accuracy, Precision, Recall, F1-score and Jaccard index. For regression, the service provides R-square, R^2 Adjusted (if applicable), RMSD, F-Value, Standard Error. In both cases the full list of predictions is provided. Finally, several figures are created: confusion matrix (for classification), real vs. predicted and QQ plot (for regression).

In the validation API users may submit datasets and models for validation using three POST methods. The available options are cross validation (stratified or random), training set split and external validation, as shown in Figure 1 found at http://test.jaqpot.org:8080/jaqpot/swagger/. All validation methods use the Task system in their execution. The result of the task will always be a Report document.



POST /validation/training_test_cross Creates Validation Report
POST /validation/training_test_split Creates Validation Report
POST /validation/test_set_validation Creates Validation Report

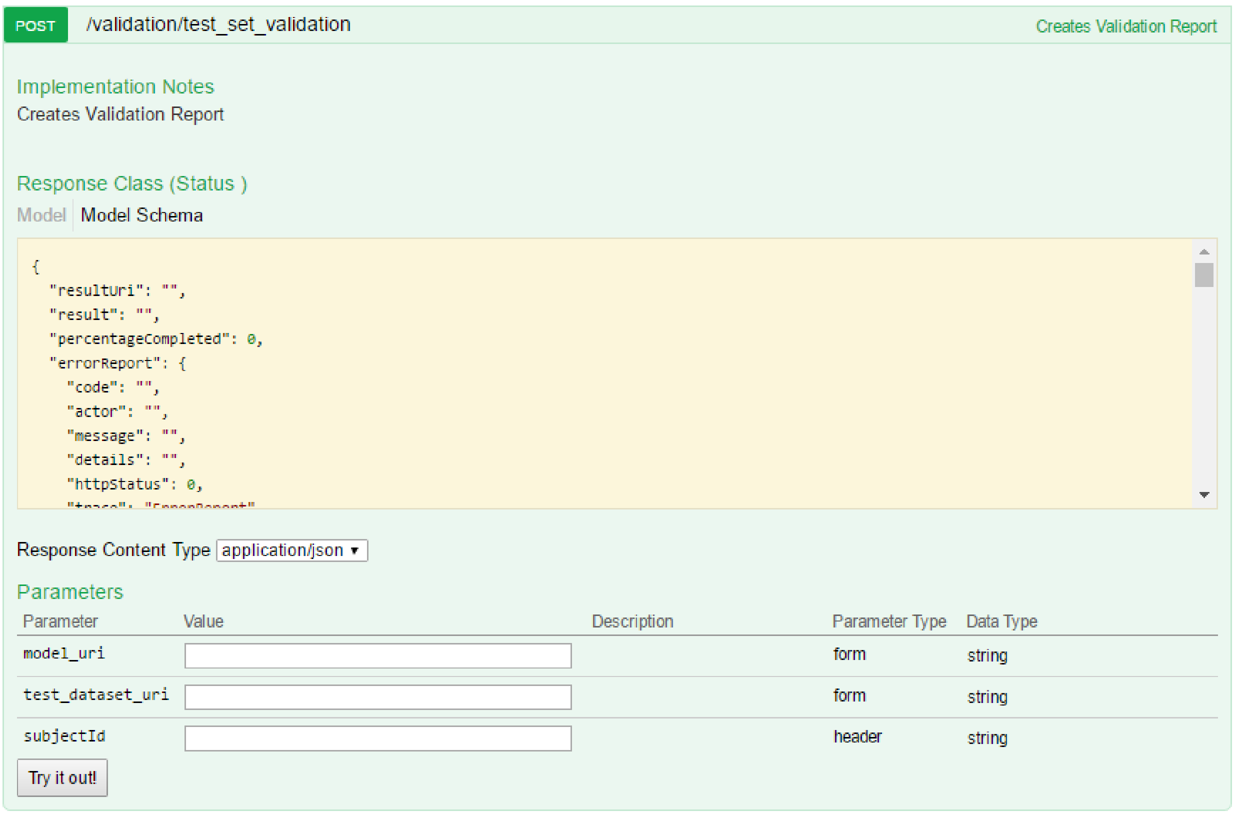
Figure 1: Validation options for JQ's API

3.1.1 External Validation

The Swagger documentation of the JQ external validation API is shown in Figure 2. A user must have already built or located an existing predictive model and have created a test set with the same variables as the training set. The following parameters should be supplied to the external validation POST method:

1) Model URI: This is the URI of the model to be used for making the prediction and therefore validation. It may be retrieved using the API by GET-ing all models visible by the user, or selected from those already in JQ, by GET-ing the list of featured models in text/csv format. A model can be further viewed by GET-ing the model by its ID.
2) Dataset URI: URI of the dataset on which to perform the external validation. Users must make sure the dataset used is compatible with the model, i.e. shares attributes with the training set used to build the model.

A task is then created which can be monitored via its ID using the GET task by ID API. Once completed, a report can be viewed in JSON format.


POST /validation/test_set_validation Creates Validation Report

Implementation Notes
Creates Validation Report

Response Class (Status)
Model Model Schema

```
{
  "resultUri": "",
  "result": "",
  "percentageCompleted": 0,
  "errorReport": {
    "code": "",
    "actor": "",
    "message": "",
    "details": "",
    "httpStatus": 0,
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
model_uri			form	string
test_dataset_uri			form	string
subjectId			header	string

Figure 2: JQ External Validation API

From the JQ user interface, this process is carried out in a rather straightforward manner, by first selecting a model already created (Figure 3) and then an external dataset (Figure 4) on which to perform the predictions and subsequently validate the model. Once the task is completed, the user may view the results in the form of a report, which contains the model predictions and various performance metrics, as well as informative plots such as QQ Plot.

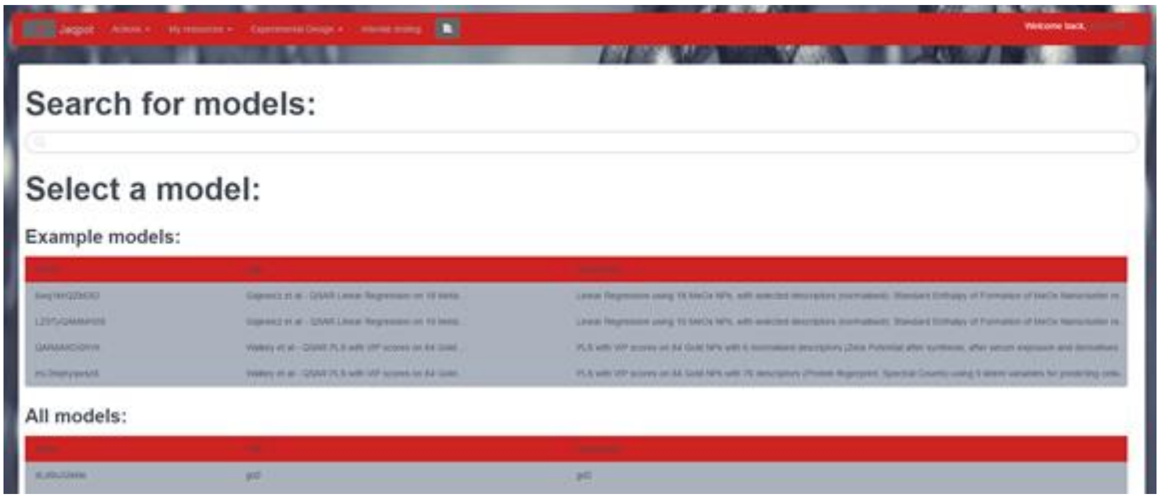


Figure 3: Select model for external validation

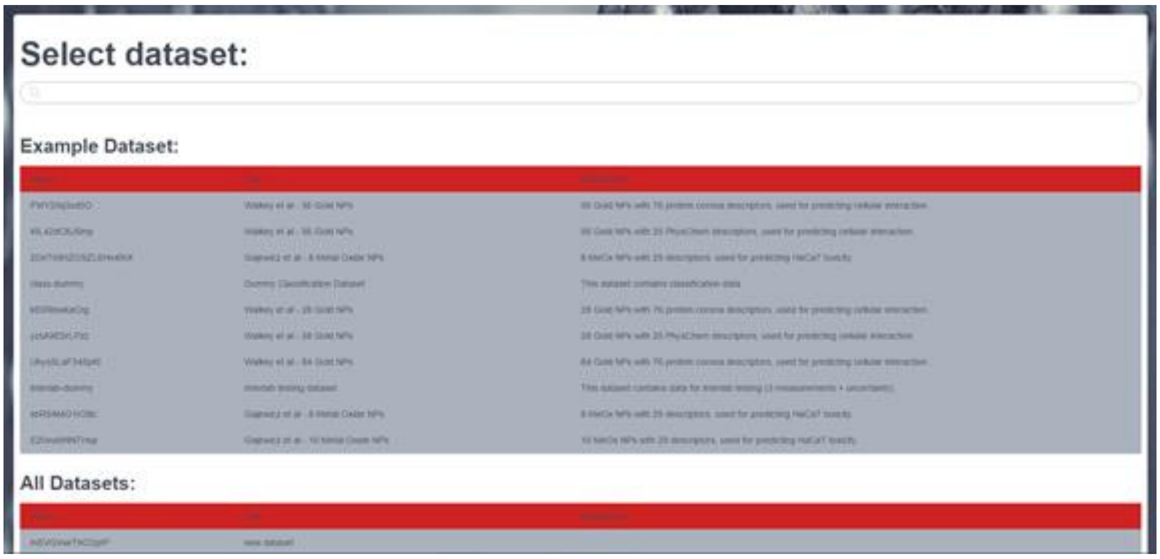
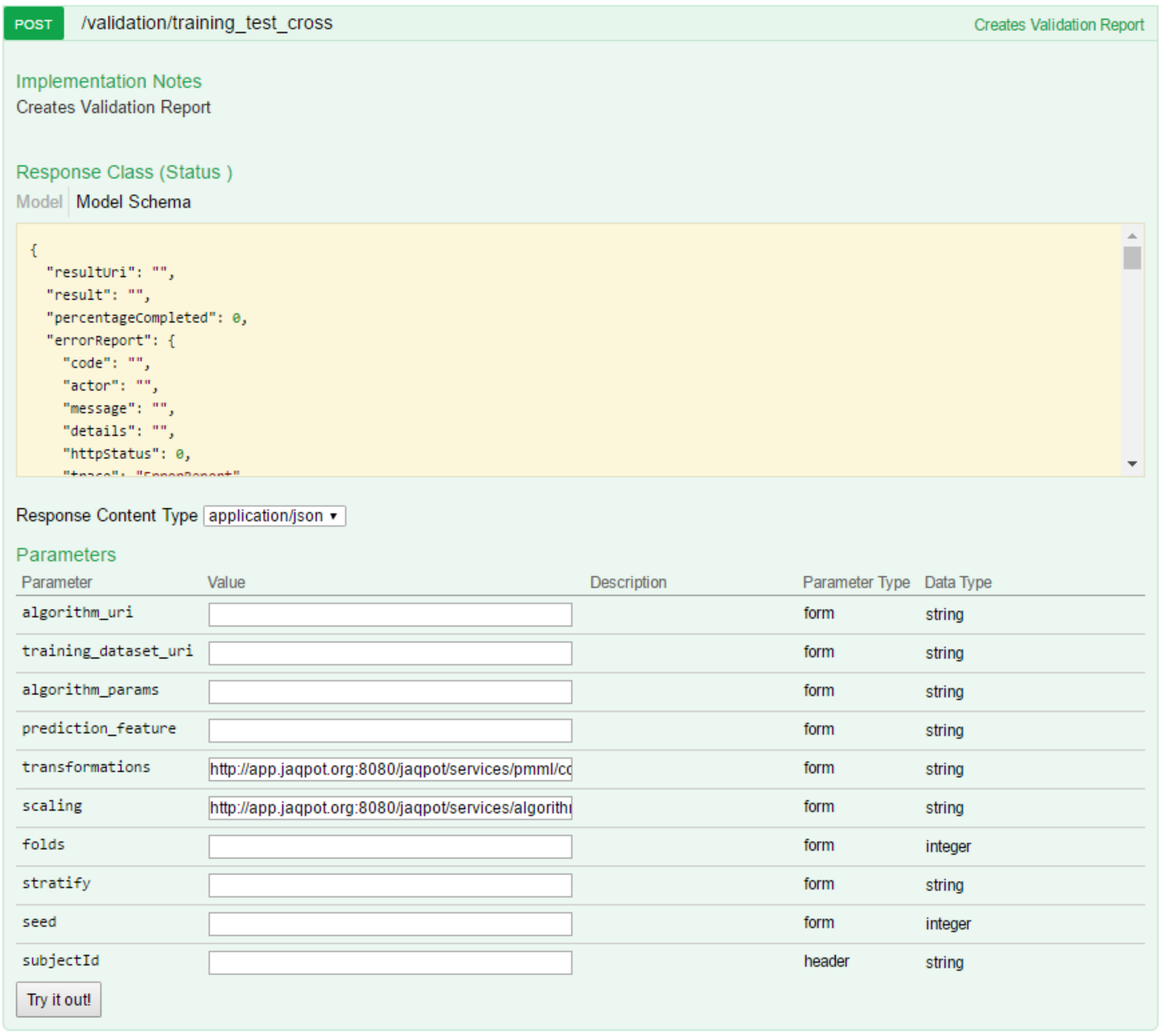


Figure 4: Select dataset for external validation

For the external validation example we used the following model http://test.jaqpot.org:8000/m_detail?name=gaj-10-lr, and the example dataset http://test.jaqpot.org:8000/data_detail?name=xbR5AMG1rOBc as the test set. The model has been built using the R linear regression algorithm http://test.jaqpot.org:8000/algorithm_detail?name=ocpu-lm on the training set suggested in Gajewicz *et al.* (10 of 18 published MeOX nanoparticles for predicting HaCaT toxicity, whilst the remaining instances were used as the test set). Two input variables were selected, namely Mullikens electronegativity Xc and Standard enthalpy of formation of metal oxide nanocluster DHcf, whilst the endpoint is described by logLC50. The produced report is provided in the following link: http://test.jaqpot.org:8000/report?name=q8HpVxGYDRFsQLp. The report can be downloaded as a PDF file.

3.1.2 Cross Validation

The Swagger documentation of the JQ cross validation API is shown in Figure 5. Users may choose to cross validate an existing algorithm on a particular data set before building the final model. In this case, options need to be provided for all data splits, including scaling or PMML transformations. Furthermore, the dataset, prediction feature and the modelling algorithm need to be provided, including potential algorithm tuning parameters if any (i.e. PLS with VIP scores requires the number of latent variables to be provided to the algorithm). Finally, the user may choose stratified or random cross validation, for which he/she will need to provide a random seed, as with commercial machine learning software applications. Please note that number of folds must be an integer and less than or equal to the number of instances. This option is only available through the API, as the number of folds is selected from a drop-down list in the UI.



POST /validation/training_test_cross Creates Validation Report

Implementation Notes
Creates Validation Report

Response Class (Status)
Model | Model Schema

```
{
  "resultUri": "",
  "result": "",
  "percentageCompleted": 0,
  "errorReport": {
    "code": "",
    "actor": "",
    "message": "",
    "details": "",
    "httpStatus": 0,
    "trace": "ErrorReport"
```

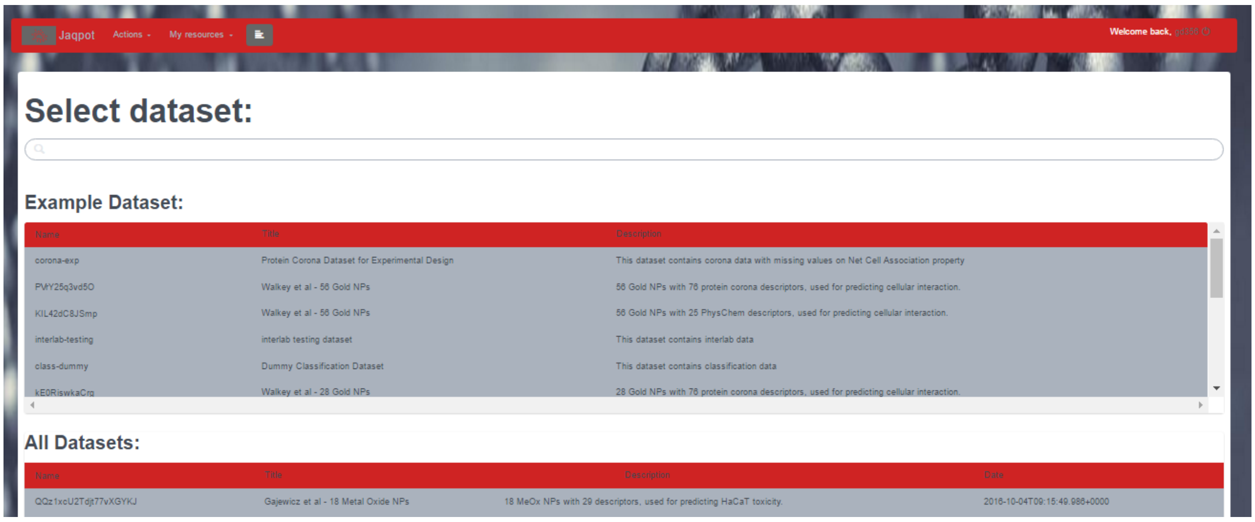
Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
algorithm_uri			form	string
training_dataset_uri			form	string
algorithm_params			form	string
prediction_feature			form	string
transformations	http://app.jaqpot.org:8080/jaqpot/services/pmml/cc		form	string
scaling	http://app.jaqpot.org:8080/jaqpot/services/algorithi		form	string
folds			form	integer
stratify			form	string
seed			form	integer
subjectId			header	string

Figure 5: JQ Cross validation API

Performing a cross validation with the UI, requires the selection of a dataset, algorithm and parameters. A Dataset may be selected either from the example datasets or the list of datasets visible to the user, as shown in Figure 6. In this example, we selected the same dataset discussed above, specifically from Gajewicz et al, comprising 10 MeOX nanoparticles described by 29 properties (found at http://test.jaqpot.org:8000/data_detail?name=E20vuMNNTHsp).



Select dataset:

Example Dataset:

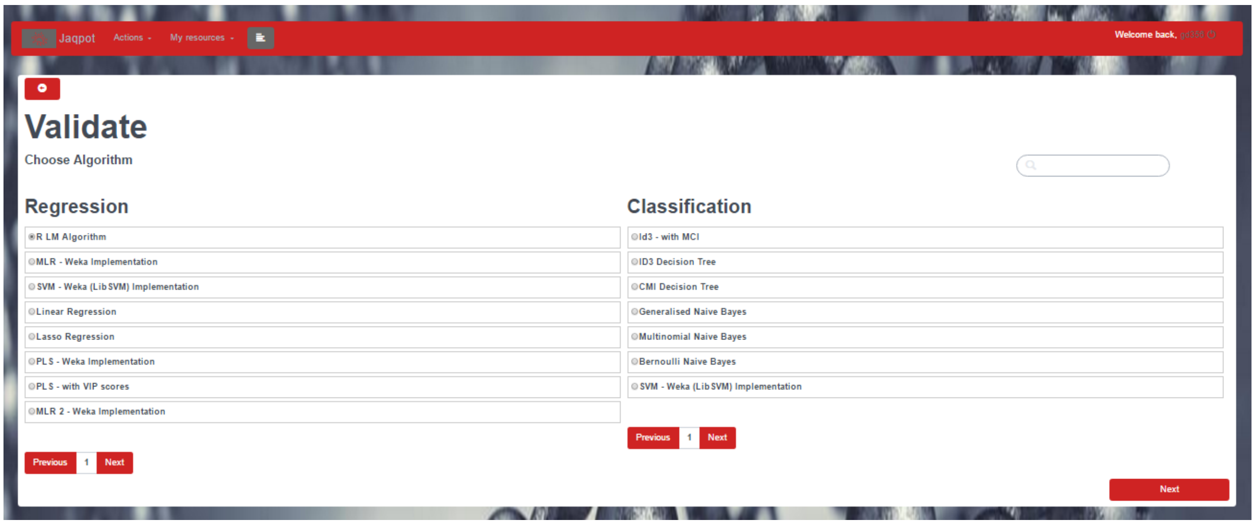
Name	Title	Description
corona-exp	Protein Corona Dataset for Experimental Design	This dataset contains corona data with missing values on Net Cell Association property
PVrY25q3vd5O	Walkey et al - 56 Gold NPs	56 Gold NPs with 76 protein corona descriptors, used for predicting cellular interaction.
KIL42dC8JSmp	Walkey et al - 56 Gold NPs	56 Gold NPs with 25 PhysChem descriptors, used for predicting cellular interaction.
interlab-testing	interlab testing dataset	This dataset contains interlab data
class-dummy	Dummy Classification Dataset	This dataset contains classification data
kE0RiswkaCrg	Walkey et al - 28 Gold NPs	28 Gold NPs with 76 protein corona descriptors, used for predicting cellular interaction.

All Datasets:

Name	Title	Description	Date
QQz1xcU2Tdjt77vXGYKJ	Gajewicz et al - 18 Metal Oxide NPs	18 MeOx NPs with 29 descriptors, used for predicting HaCaT toxicity.	2016-10-04T09:15:49.986+0000

Figure 6: Select dataset for cross validation screen

On the next screen, an appropriate algorithm should be chosen from the regression or classification list. For this example, we have chosen the first option available, namely the R language Linear Regression (http://test.jaqpot.org:8000/algorithm_detail?name=ocpu-lm). This is shown In Figure 7.



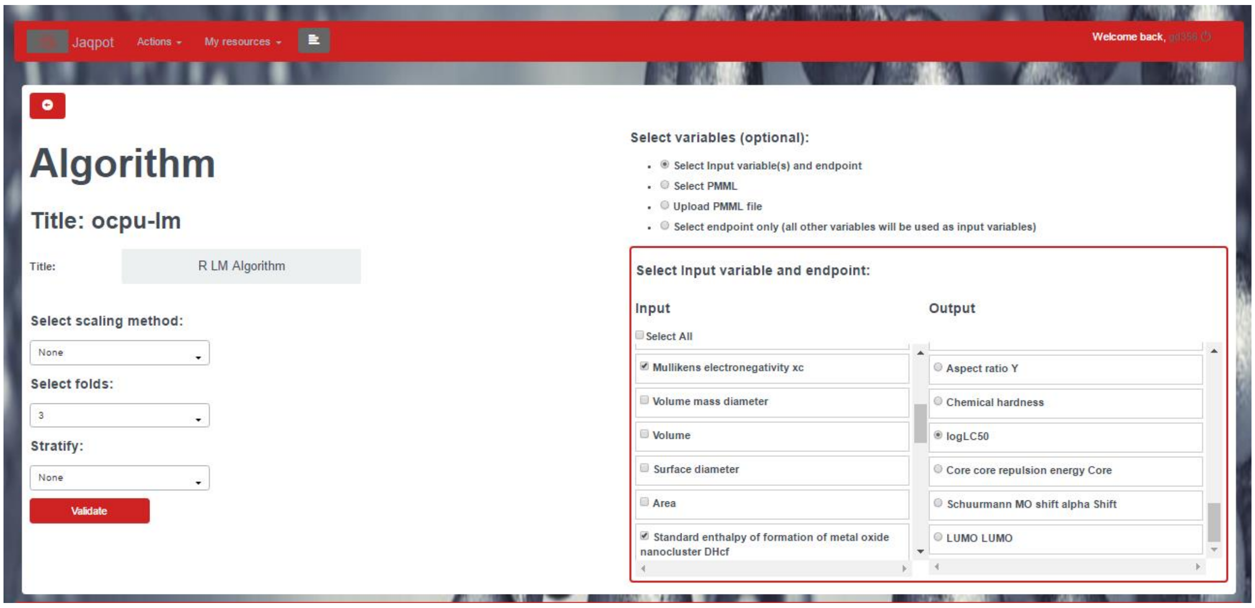
Validate

Choose Algorithm

Regression	Classification
● R LM Algorithm	○ Id3 - with MCI
○ MLR - Weka Implementation	○ ID3 Decision Tree
○ SVM - Weka (Lib SVM) Implementation	○ CMI Decision Tree
○ Linear Regression	○ Generalised Naive Bayes
○ Lasso Regression	○ Multinomial Naive Bayes
○ PLS - Weka Implementation	○ Bernoulli Naive Bayes
○ PLS - with VIP scores	○ SVM - Weka (Lib SVM) Implementation
○ MLR 2 - Weka Implementation	

Figure 7: Select algorithm for cross validation screenshot

Finally, the validation parameters need to be set. This screen is almost identical to the "Train a model" screen (see Deliverable 4.3). Apart from the endpoint selection and data scaling normalisation, the user now has two additional options to specify, the number of folds and whether the training dataset in each fold should be stratified (normal), random (user provides a seed for random function) or split based on the observed order of data instances (None). A screenshot of this section is shown in Figure 8. As discussed briefly above, users of the JQ UI are offered 3, 5 and 10-fold cross validation options in order to avoid mistakes (i.e. disproportional number of folds) and also avoid the over-consumption of resources.



Algorithm

Title: ocpu-lm

Title: R LM Algorithm

Select scaling method: None

Select folds: 3

Stratify: None

Select variables (optional):

- [✓ Select Input variable(s) and endpoint]
- Select PMML
- Upload PMML file
- Select endpoint only (all other variables will be used as input variables)

Select Input variable and endpoint:

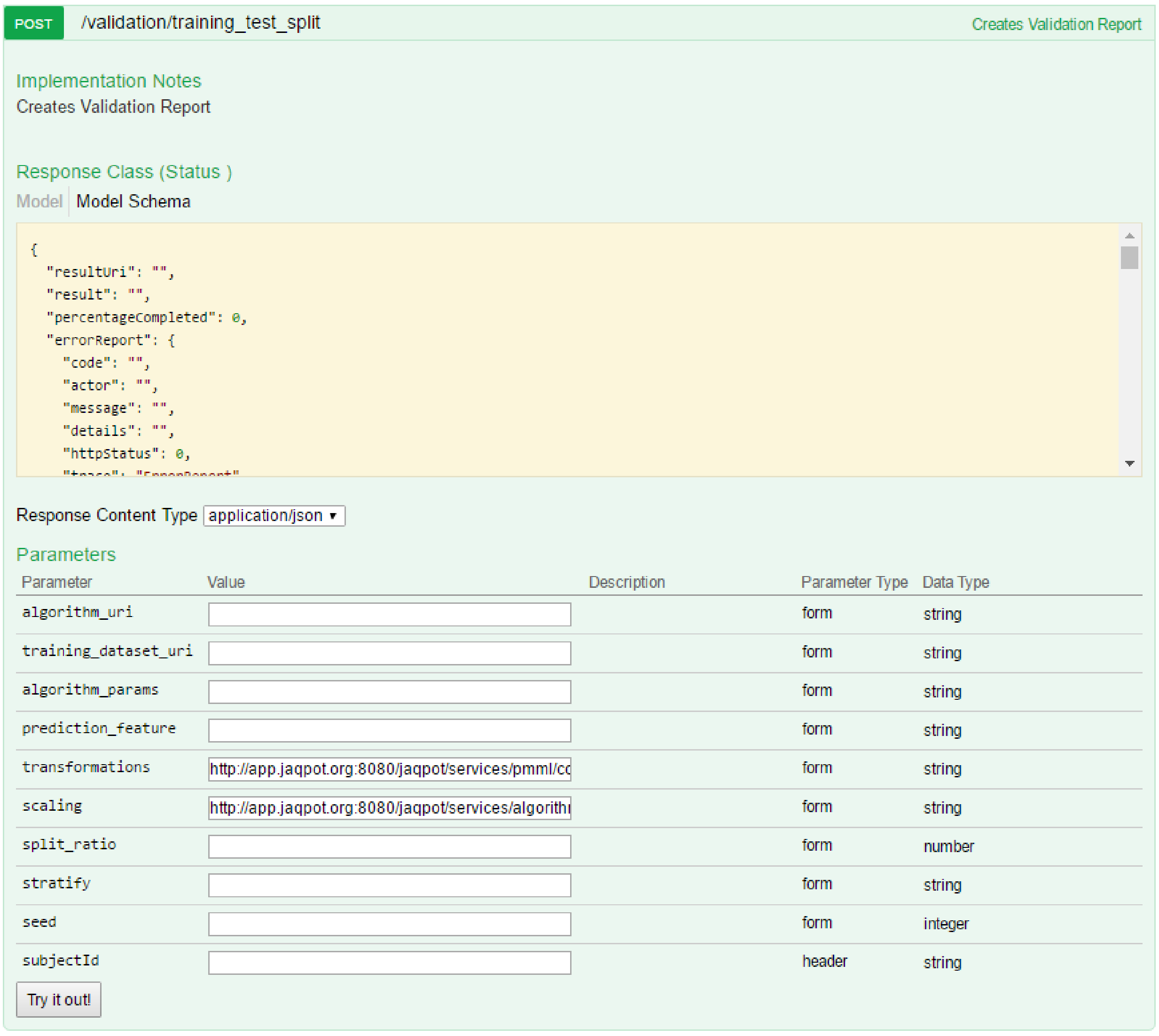
Input	Output
Select All	
✓ Mullikens electronegativity xc	Aspect ratio Y
Volume mass diameter	Chemical hardness
Volume	● logLC50
Surface diameter	Core core repulsion energy Core
Area	Schuurmann MO shift alpha Shift
✓ Standard enthalpy of formation of metal oxide nanocluster DHcf	LUMO LUMO

Figure 8: Screenshot of additional algorithm parameters (folds, stratify and seed) for cross validation

For this example, we followed again the instructions set by Gajewicz et al. and have selected the same two input variables, Mullikens electronegativity Xc and Standard enthalpy of formation of metal oxide nanocluster DHcf, and logLC50 as the endpoint. Scaling and stratification were both set to "None". By clicking on "validate", we have run a 3-fold (default) cross validation on the specified dataset using linear regression for the current example. Once completed, users may view the resulting report by clicking on "See result". This is located at: http://test.jaqpot.org:8000/report?name=eJnCCdzAwNCvh2x

3.1.3 Training Set Split Validation

The last option available in the validation API is to validate an existing algorithm by splitting a dataset into training and test sets on a specified ratio. The Swagger documentation of the JQ training set split validation API is shown in Figure 9. As in the cross validation case, transformations and scaling need to be provided if required, as well as the algorithm and prediction feature. More specifically, users must provide the following options: algorithm URI, training dataset URI, algorithm parameters (if applicable), prediction feature (usually biological endpoint), PMML transformations (optional), scaling (normalize or scale between 0 and 1; optional), ratio (training to test set split; between 0 and 1) and stratify (optional; ‘normal’, ‘none’ or ‘random’). Option ‘None’ splits the data sequentially, ‘normal’ adds stratification, for ‘random’ users must provide a seed).



POST /validation/training_test_split Creates Validation Report

Implementation Notes
Creates Validation Report

Response Class (Status)
Model | Model Schema

```
{
  "resultUri": "",
  "result": "",
  "percentageCompleted": 0,
  "errorReport": {
    "code": "",
    "actor": "",
    "message": "",
    "details": "",
    "httpStatus": 0,
```

Response Content Type application/json

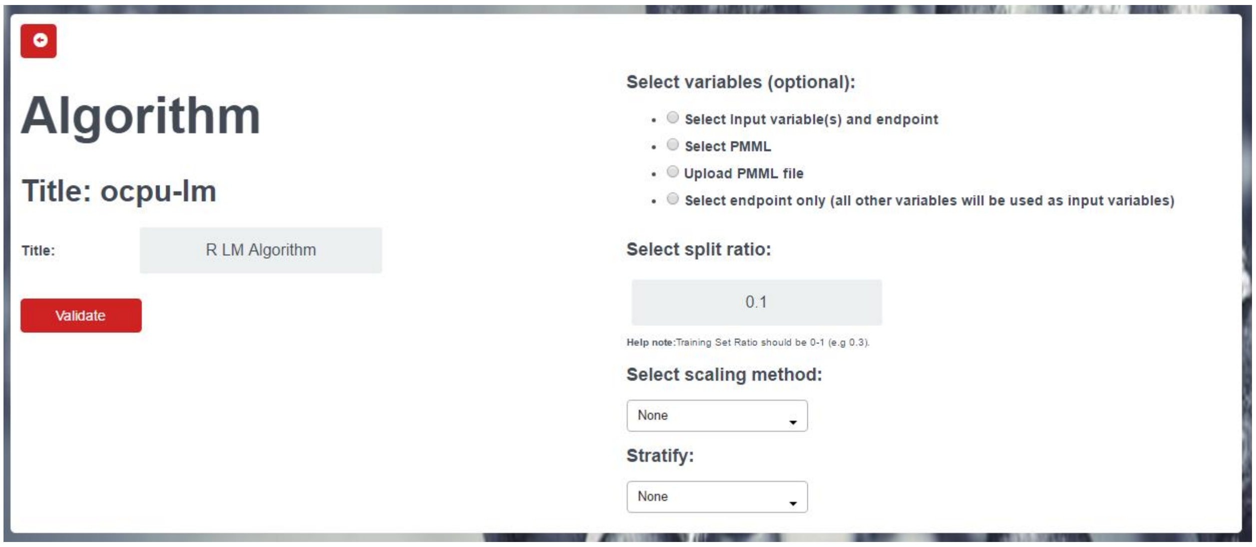
Parameters

Parameter	Value	Description	Parameter Type	Data Type
algorithm_uri			form	string
training_dataset_uri			form	string
algorithm_params			form	string
prediction_feature			form	string
transformations	http://app.jaqpot.org:8080/jaqpot/services/pmml/c		form	string
scaling	http://app.jaqpot.org:8080/jaqpot/services/algorith		form	string
split_ratio			form	number
stratify			form	string
seed			form	integer
subjectId			header	string

Try it out!

Figure 9: JQ Training set split validation API

Users of JQ through the user interface must select a dataset and algorithm as described above in detail. The only additional option for the algorithm is the “Select Split Ratio” field, which should be a value larger than zero and smaller than one, which describes the training-to-test-set-ratio to the algorithm. Therefore 0.7 would suggest that 70% of instances will be used for training and 30% will be held out for testing/validation. This is shown in Figure 10. The “Stratify” drop down menu is still available for the different instance selection options. The resulting report can be viewed once the task is completed.



Algorithm

Title: ocpu-lm

Title: R LM Algorithm

Select variables (optional):

- Select Input variable(s) and endpoint
- Select PMML
- Upload PMML file
- Select endpoint only (all other variables will be used as input variables)

Select split ratio: 0.1

Help note:Training Set Ratio should be 0-1 (e.g 0.3).

Select scaling method: None

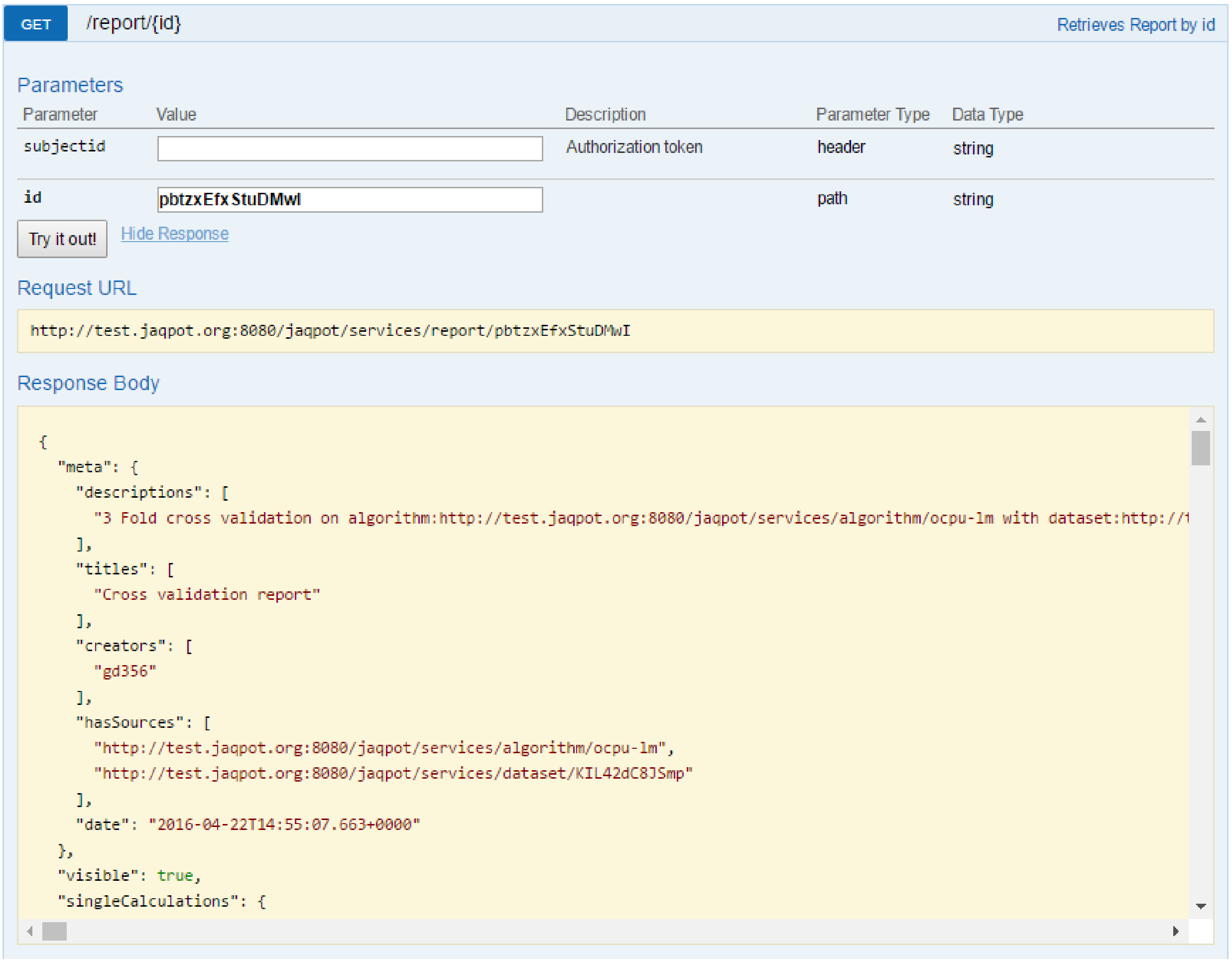
Stratify: None

Figure 10: Algorithm options for training set split validation

Using the same dataset (Gajewicz *et al.,* 10 MeOX NPs, found at http://test.jaqpot.org:8000/d_validate?dataset=E20vuMNNTHsp) and algorithm (JQ's R linear regression implementation) as the example above, we demonstrate the training set split validation. Mullikens electronegativity Xc and Standard enthalpy of formation of metal oxide nanocluster DHcf were selected again as input variables and logLC50 as the endpoint, according to Gajewicz *et al*. The split ratio was set to 0.7 (see Figure 10), whilst scaling and stratification were both set to "None". The resulting split validation report can be found at: http://test.jaqpot.org:8000/report?name=BS4vJNc72Qomsea.

3.1.4 Validation Reports

In order to view a validation report using the API, users must paste the report ID generated from one of the validation services in the "ID" field, as demonstrated in Figure 11. This process is the same for any type of validation scheme. The API then returns the report in JSON format in the response body, which contains single calculations, array calculations and figures. Single calculations are metrics such as R^2, standard error etc., array calculations contain the real and predicted values for comparison, and figures contain plots such as a QQ plot.



Parameters

Parameter	Value	Description	Parameter Type	Data Type
subjectid		Authorization token	header	string
id	pbtzxEfxStuDMwI		path	string

Request URL

```
http://test.jaqpot.org:8080/jaqpot/services/report/pbtzxEfxStuDMwI
```

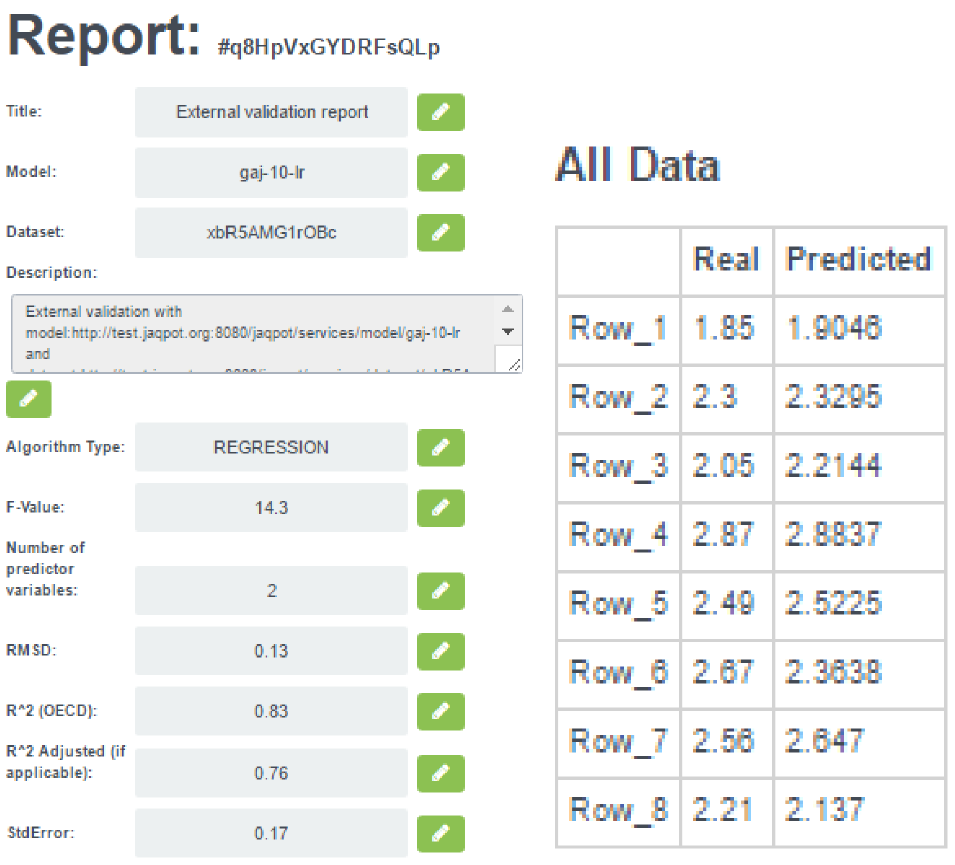
Response Body

```
{
  "meta": {
    "descriptions": [
      "3 Fold cross validation on algorithm:http://test.jaqpot.org:8080/jaqpot/services/algorithm/ocpu-lm with dataset:http://t
    ],
    "titles": [
      "Cross validation report"
    ],
    "creators": [
      "gd356"
    ],
    "hasSources": [
      "http://test.jaqpot.org:8080/jaqpot/services/algorithm/ocpu-lm",
      "http://test.jaqpot.org:8080/jaqpot/services/dataset/KIL42dC8JSmp"
    ],
    "date": "2016-04-22T14:55:07.663+0000"
  },
  "visible": true,
  "singleCalculations": {
```

Figure 11: JQ GET Report by ID API

On the UI level, users may view the report directly after a validation task is completed, as well as under “My Resources”. As mentioned above, these reports contain single calculations and array calculations (demonstrated in Figure 12), as well as images (shown in Figure 13) and can be downloaded as PDF files. Figure 13 present metrics and plots corresponding to the case study presented in section 3.1.1.

(http://test.jaqpot.org:8000/report?name=q8HpVxGYDRFsQLp).



Report: #q8HpVxGYDRFsQLp

Title:	External validation report
Model:	gaj-10-lr
Dataset:	xbR5AMG1rOBc
Description:	External validation with model:http://test.jaqpot.org:8080/jaqpot/services/model/gaj-10-lr and
Algorithm Type:	REGRESSION
F-Value:	14.3
Number of predictor variables:	2
RMSD:	0.13
R^2 (OECD):	0.83
R^2 Adjusted (if applicable):	0.76
StdError:	0.17

All Data

	Real	Predicted
Row_1	1.85	1.9046
Row_2	2.3	2.3295
Row_3	2.05	2.2144
Row_4	2.87	2.8837
Row_5	2.49	2.5225
Row_6	2.67	2.3638
Row_7	2.56	2.647
Row_8	2.21	2.137

Figure 12: Validation Performance metrics and predictions

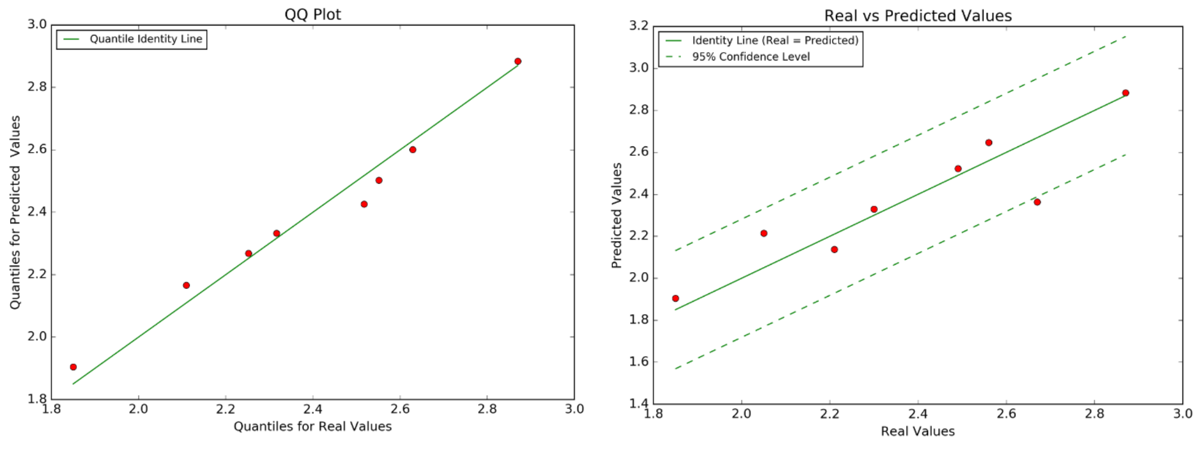


Figure 13: Plots contained within validation report

3.2 Validation Services In The Nano-Lazar Modelling Framework

Within the OpenTox project we have developed and implemented a REST-based validation service that is applicable to all OpenTox compliant modelling services. This service was updated to reflect the latest API changes and is available at https://services.in-silico.ch/validation.

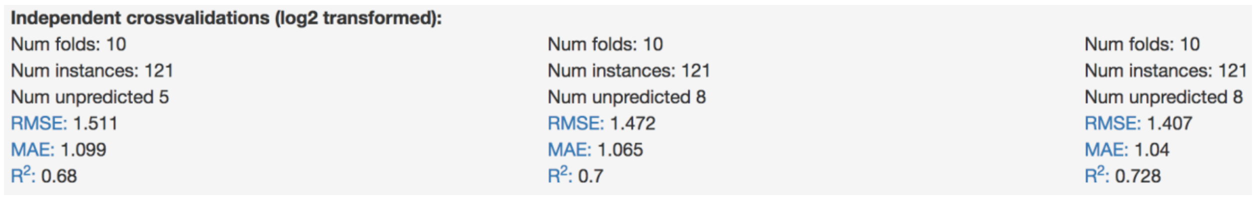
Practical experience with large scale validation experiments have shown however that the REST approach is time consuming, inefficient, error prone and hard to debug, especially during the development phase. For this reason, validation schemes were tightly integrated into the lazar system, which led to a significant speedup of validation runtimes and a much-improved tracing of prediction errors. Based on user feedback applicability domain definitions for classifications were revised. The slightly unintuitive confidence index was replaced by positive and negative prediction probabilities.

At present lazar supports all major validation schemes for classification and regression including

- external validation
- training-test set splits
- cross validation
- leave-one-out validation

These validations generate the usual validation metrics (e.g. accuracy, sensitivity, specificity, true positive rate, true negative rate, r^2, RMSE, MAE) and graphical depictions (e.g. correlation plots). Apart from these standard validation features lazar provides tools specifically for model developers, e.g. the inspection of individual predictions, sorting of predictions according to prediction errors or the distance to the applicability domain.

The validation API is documented at http://www.rubydoc.info/gems/lazar. Like the rest of lazar validations are created on the command line, but lazar provides convenient methods for model building, that generate prediction models together with a default set of validations (three independent cross validations and leave-one-out validation) with a single method call. Models and validation results can be integrated into GUIs (e.g. nano-lazar, https://nano-lazar.in-silico.ch) without any modifications (Figure 14).

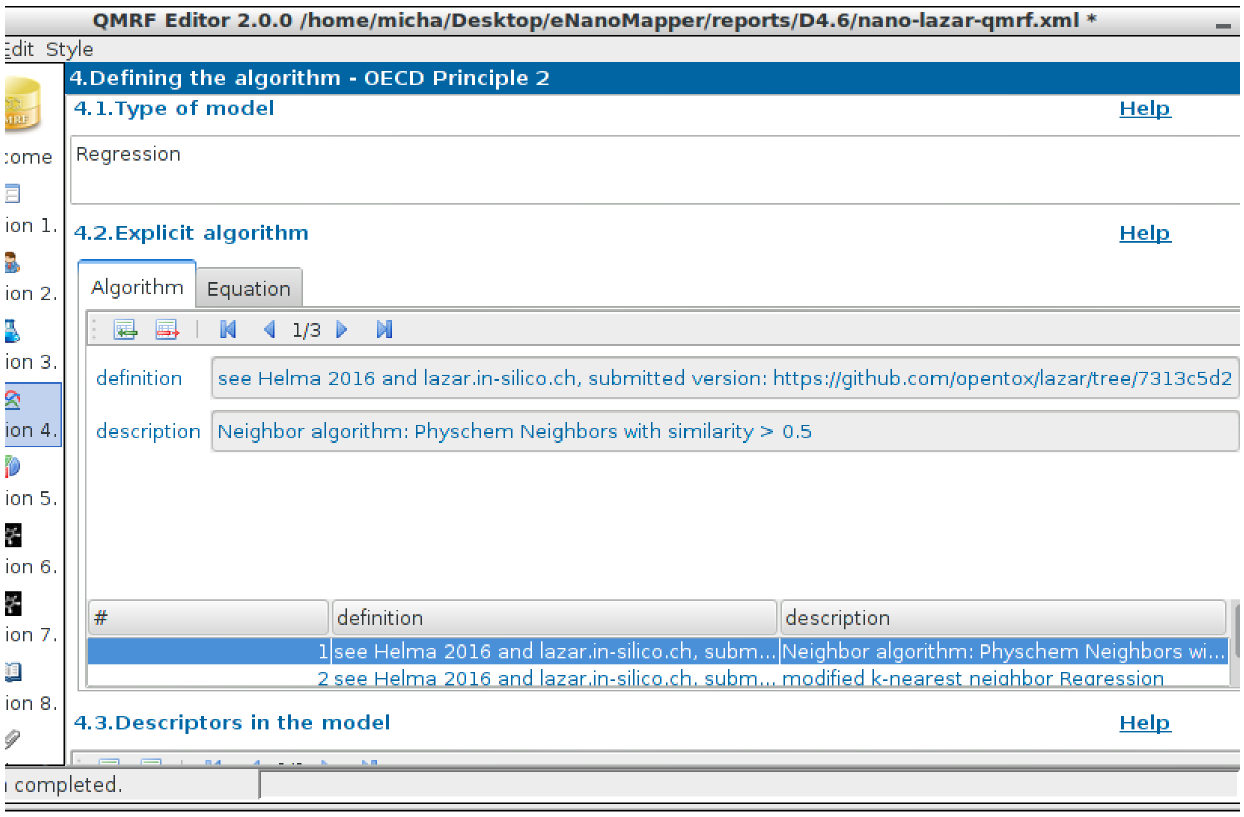


Independent crossvalidations (log2 transformed):

Num folds: 10	Num folds: 10	Num folds: 10
Num instances: 121	Num instances: 121	Num instances: 121
Num unpredicted 5	Num unpredicted 8	Num unpredicted 8
RMSE: 1.511	RMSE: 1.472	RMSE: 1.407
MAE: 1.099	MAE: 1.065	MAE: 1.04
R^2: 0.68	R^2: 0.7	R^2: 0.728

Figure 14: Three independent cross validation results in the GUI

The lazar QMRF report service (section 7) creates QMRF reports directly from model validations. In current development versions (https://nano-lazar-dev.in-silico.ch) QMRF and QPRF reports will be directly available from the nano-lazar GUI (Figure 15).



QMRF Editor 2.0.0 /home/micha/Desktop/eNanoMapper/reports/D4.6/nano-lazar-qmrf.xml *

4.Defining the algorithm - OECD Principle 2

4.1.Type of model

Regression

4.2.Explicit algorithm

definition: see Helma 2016 and lazar.in-silico.ch, submitted version: https://github.com/opentox/lazar/tree/7313c5d2

description: Neighbor algorithm: Physchem Neighbors with similarity > 0.5

#	definition	description
1	see Helma 2016 and lazar.in-silico.ch, subm...	Neighbor algorithm: Physchem Neighbors wi...
2	see Helma 2016 and lazar.in-silico.ch, subm...	modified k-nearest neighbor Regression

4.3.Descriptors in the model

Figure 15: Downloaded QMRF report from nano-lazar GUI

4. Read Across Training And Predicting Functionalities

In this section, we will describe the read across functionality of JQ and nano-lazar, which allows users to make predictions for a query nanoparticle based on its similarity to nanoparticles used in a training set in order to build a model.

4.1 JQ Read Across SERVICES

As with all other JQ services described in previous deliverables, read across options can be accessed both directly on the right-hand side of the main JQ UI page, or under "Actions". The two options currently available are "Train" and "Test". These work similarly to the modelling services described in Deliverable 4.3. For training a model, users must provide a dataset, a distance metric (currently available: euclidean, manhattan or ensemble), a distance threshold between zero and one, and whether they would like to compute a confidence level when making a prediction. For testing, users must simply select a dataset whose attributes are compatible with those of the dataset used in training for building the read across model. In the following sections, JQ read across training and testing functionalities will be explained on both API and UI levels.

4.1.1 JQ Read Across API

The read across functionalities on the API level can be found at http://test.jaqpot.org:8080/jaqpot/swagger/.

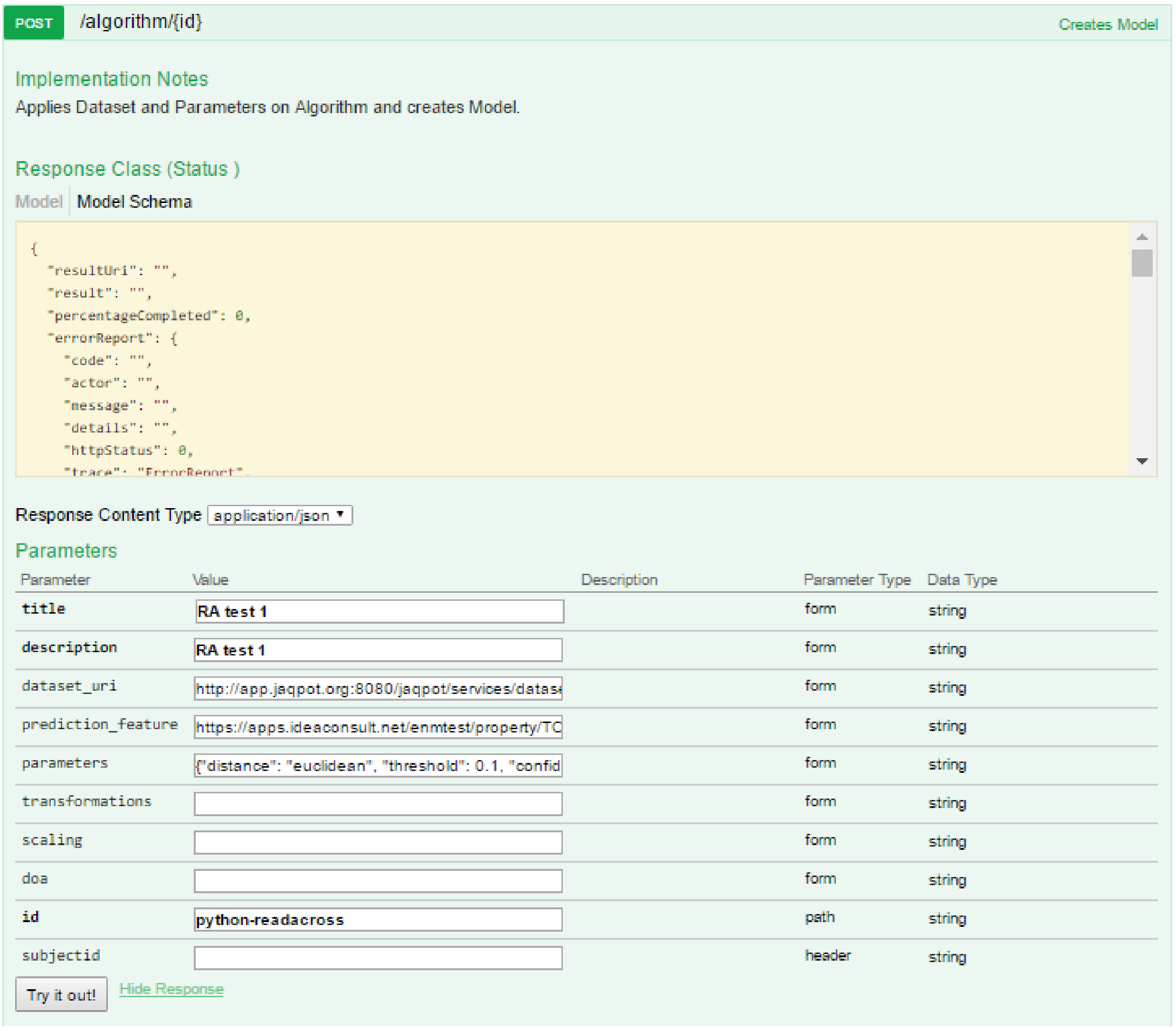
For training a read across model, users may use the POST algorithm by ID API by inserting the algorithm ID python-read-across. This (and indeed the full list of algorithms) can also be retrieved using the GET algorithm API. Figure 16 shows a screenshot of the API, which requires several fields to be filled in by the user. These comprise the model title and description, a dataset URI and the endpoint variable (prediction feature) URI, as well as whether transformations or scaling/normalisation are appropriate. The algorithm ID should be set to "python-read-across" and three parameters must be supplied in dictionary format, namely "distance", "threshold" and "confidence". Specifically:

- Distance: "euclidean", "manhattan" or "ensemble"

- Threshold: any number between 0 and 1
- Confidence 0 (no) or 1 (yes) depending on whether confidence levels should be calculated upon testing

Therefore, the parameters dictionary (see previous deliverables) is along the lines of:

{"confidence": 1, "distance": "ensemble", "threshold": 0.1}



POST /algorithm/{id} Creates Model

Implementation Notes

Applies Dataset and Parameters on Algorithm and creates Model.

Response Class (Status)

Model | Model Schema

```
{
  "resultUri": "",
  "result": "",
  "percentageCompleted": 0,
  "errorReport": {
    "code": "",
    "actor": "",
    "message": "",
    "details": "",
    "httpStatus": 0,
    "trace": "ErrorReport"
```

Response Content Type application/json

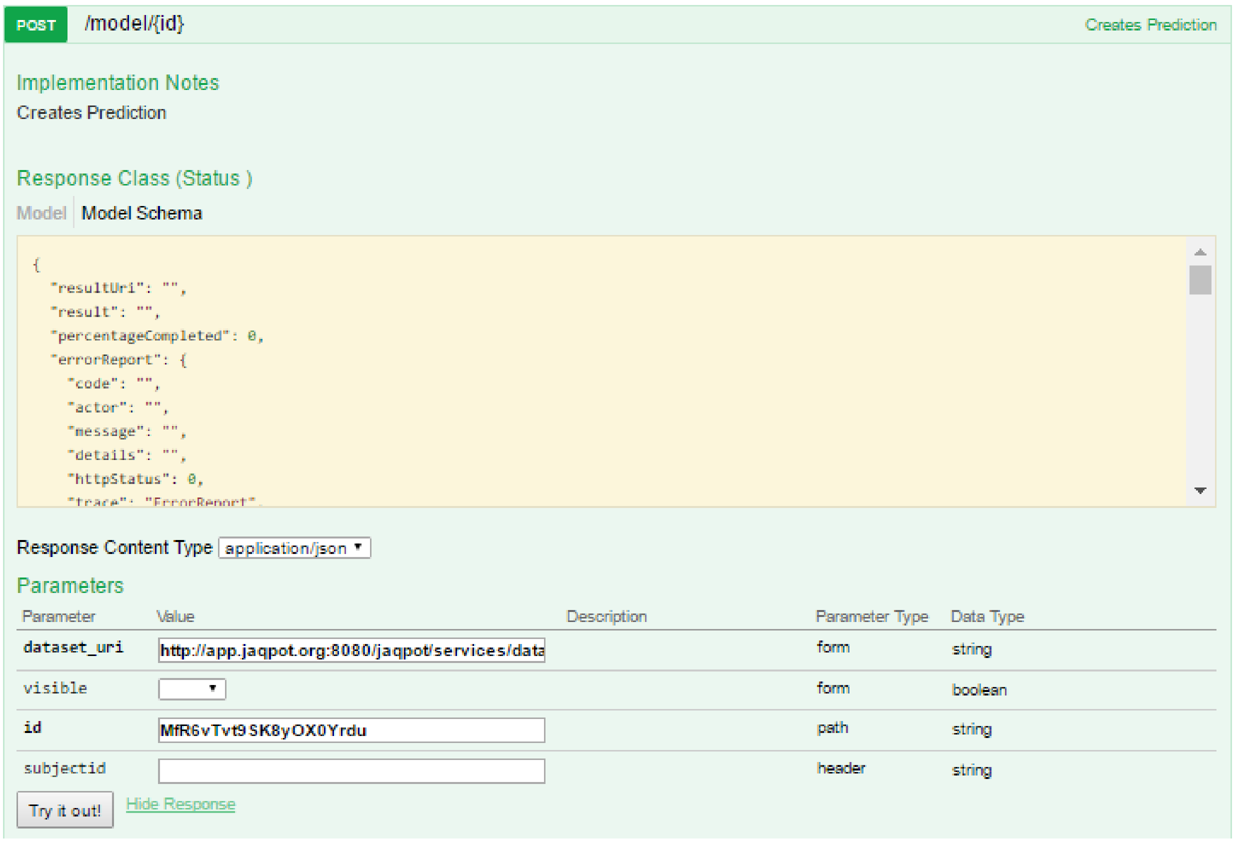
Parameters

Parameter	Value	Description	Parameter Type	Data Type
title	RA test 1		form	string
description	RA test 1		form	string
dataset_uri	http://app.jaqpot.org:8080/jaqpot/services/datase		form	string
prediction_feature	https://apps.ideaconsult.net/enmtest/property/TC		form	string
parameters	{"distance": "euclidean", "threshold": 0.1, "confid		form	string
transformations			form	string
scaling			form	string
doa			form	string
id	python-readacross		path	string
subjectid			header	string

Try it out! Hide Response

Figure 16: JQ Read Across training a model using JQ POST algorithm by ID API

Making a prediction using the read across method is identical to all QSAR testing methods available in JQ. Users must provide a compatible dataset to the training set and the ID of the model built to the POST model by ID API found under model on the JQ swagger page. This is shown in Figure 17. The resulting predictions are demonstrated via the UI later in this deliverable, as it was deemed to be preferable for the reader due to friendly visualisation.


POST /model/{id} Creates Prediction

Implementation Notes

Creates Prediction

Response Class (Status)

Model | Model Schema

```
{
  "resultUri": "",
  "result": "",
  "percentageCompleted": 0,
  "errorReport": {
    "code": "",
    "actor": "",
    "message": "",
    "details": "",
    "httpStatus": 0,
    "trace": "ErrorReport",
```

Response Content Type application/json

Parameters

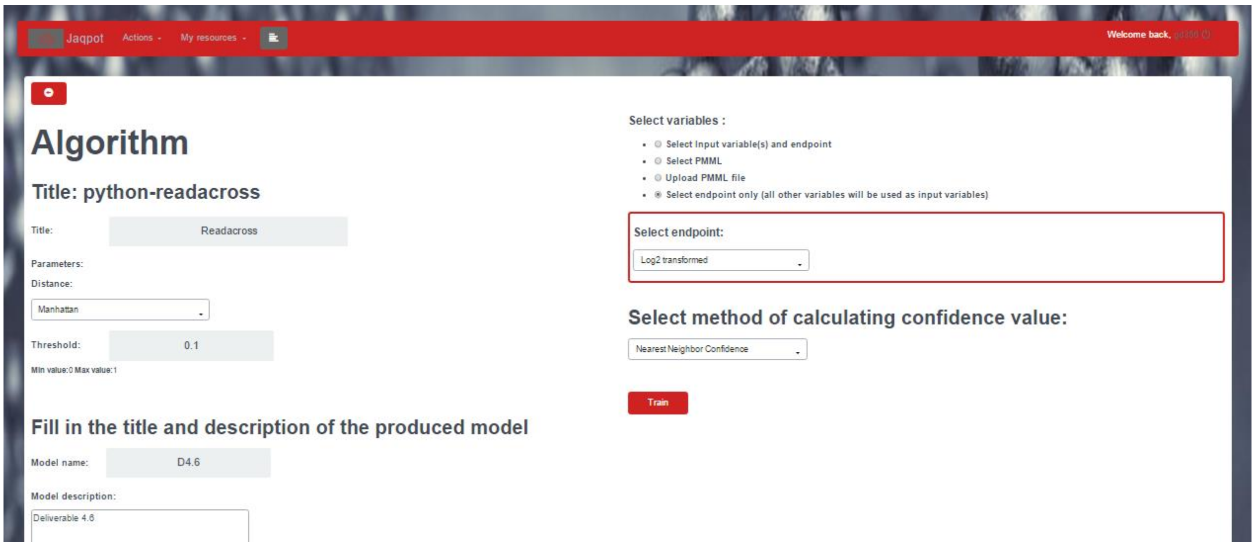
Parameter	Value	Description	Parameter Type	Data Type
dataset_uri	http://app.jaqpot.org:8080/jaqpot/services/data		form	string
visible			form	boolean
id	MfR6vTvt9SK8yOX0Yrdu		path	string
subjectid			header	string

Figure 17: JQ's POST model by ID for making read across predictions

4.1.2 JQ Read Across Functionalities From The UI

In this section, we will demonstrate the read across functionalities of JQ on the UI level. These can be found on the main screen (bottom right; "Read Across" train/test) or under the "actions" menu. As with regular modelling services, read across allows users to create a model in order to be later used to make a prediction based on similarity during the testing process.

Since the algorithm had already been selected (user chooses read across prediction), this first requires the choice of a dataset on which to build the model. In the next screen users, may parameterise the algorithm, by selecting a distance metric between 'euclidean', 'manhattan' and 'ensemble' and a threshold between 0 and 1 (the closest to 0 the strictest) which will be used to assess nanoparticle similarity. As with the other modelling services, users must provide a model name and description, as well as select variables and endpoint. Finally, users may select a method to calculate the prediction confidence level. Currently, the only option available is "Nearest Neighbour Confidence" (or "None"). This is demonstrated in Figure 18.



Algorithm

Title: python-readacross

Title: Readacross

Parameters:

Distance: Manhattan

Threshold: 0.1

Min value:0 Max value:1

Fill in the title and description of the produced model

Model name: D4.6

Model description: Deliverable 4.6

Select variables :

- Select Input variable(s) and endpoint
- Select PMML
- Upload PMML file
- [✓ Select endpoint only (all other variables will be used as input variables)]

Select endpoint: Log2 transformed

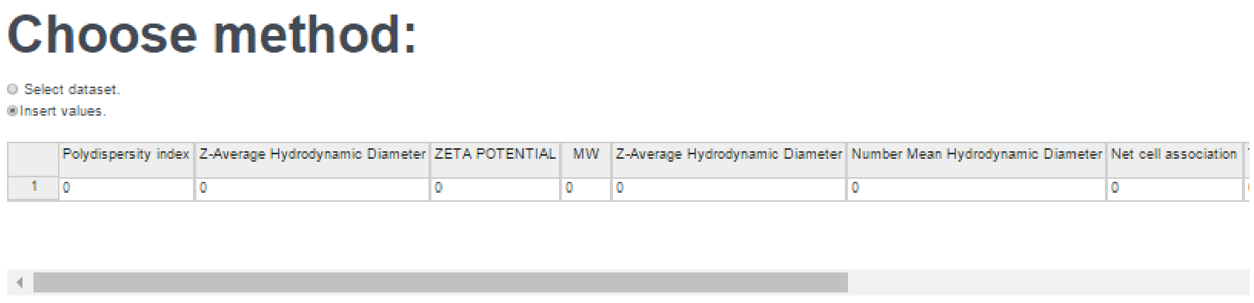
Select method of calculating confidence value: Nearest Neighbor Confidence

Figure 18: Options available to the user for training a model using the read across algorithm

By clicking on "Train", a task is created. Users may monitor this task. When it is completed, an option "See result" appears under "Task Completed". By clicking on "See result", the user may view the model details, such as the description, the algorithm used to create it, the dependent/independent/etc. features, and three options: predict, validate and delete. The model can be retrieved at any time under the "My resources" tab.

For this example, the dataset used for building the model comprises 56 gold NPs described by 25 physicochemical descriptors and the endpoint to be predicted is the log2 transformed Net Cell Association. This dataset is taken from Walkey *et al.* and can be found under JQ's example datasets (http://test.jaqpot.org:8000/data_detail?name=ghh9Usgiv2ZuOjjsaZHA). Manhattan distance was chosen with a threshold of 0.1 and the confidence value was calculated using "Nearest Neighbour Confidence". The resulting model used for this demonstration is found at: http://test.jaqpot.org:8000/m_detail?name=shWpOGagqSJ5l6PfvkpJ.

Read across prediction services are available both on the main screen in the "Read Across" box, as well as under "Actions". As with other modelling services, users must first select an already trained model. Users must then either select a dataset on which read across predictions will be performed or select "Insert Values", for which JQ will provide a table to be filled in containing the full list of variables used by the read across model. Users may add more rows or directly paste a table from a spreadsheet (i.e. from Microsoft Excel). This is shown in Figure 19. Should users choose to select a dataset, they must ensure that it is compatible with the model, meaning that the dataset shares the same attributes. Such a dataset is the test dataset provided by Walkey et al., comprising 28 gold NPs described by 25 physicochemical properties (http://test.jaqpot.org:8000/data_detail?name=yzsAXE5rLPzz). This dataset is also found in JQ's example datasets.

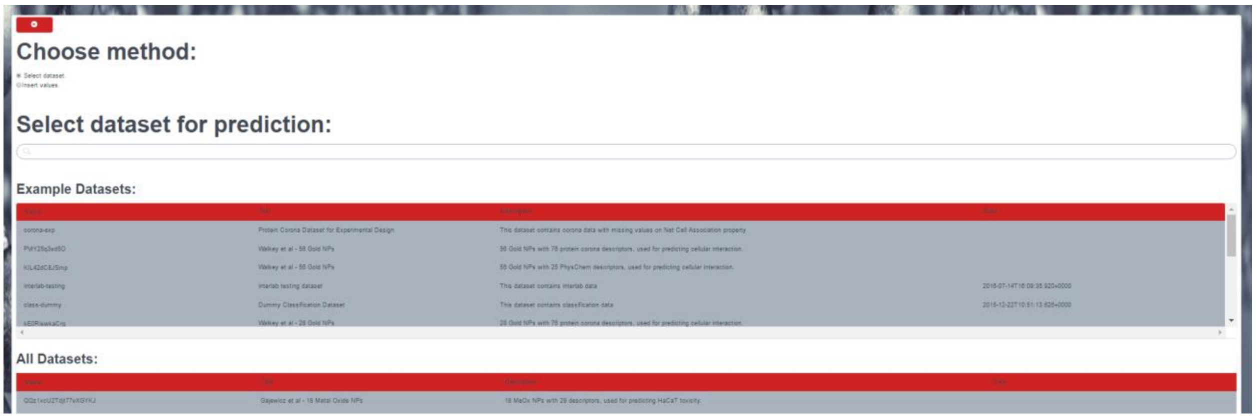


Choose method:

○ Select dataset.
◉ Insert values.

	Polydispersity index	Z-Average Hydrodynamic Diameter	ZETA POTENTIAL	MW	Z-Average Hydrodynamic Diameter	Number Mean Hydrodynamic Diameter	Net cell association
1	0	0	0	0	0	0	0

Figure 19: Insert values example of JQ read across prediction functionality



Choose method:

◉ Select dataset
○ Insert values

Select dataset for prediction:

Example Datasets:

Name	Title	Description	Date
corona-exp	Protein Corona Dataset for Experimental Design	This dataset contains corona data with missing values on Net Cell Association property	
PWY26q3vd5O	Walkey et al - 56 Gold NPs	56 Gold NPs with 78 protein corona descriptors, used for predicting cellular interaction.	
KIL42dC8JSmp	Walkey et al - 56 Gold NPs	56 Gold NPs with 25 PhysChem descriptors, used for predicting cellular interaction.	
interlab-testing	interlab testing dataset	This dataset contains interlab data	2016-07-14T16:09:35.920+0000
class-dummy	Dummy Classification Dataset	This dataset contains classification data	2016-12-22T10:51:13.626+0000
kE0RlswkaCrq	Walkey et al - 28 Gold NPs	28 Gold NPs with 78 protein corona descriptors, used for predicting cellular interaction.	

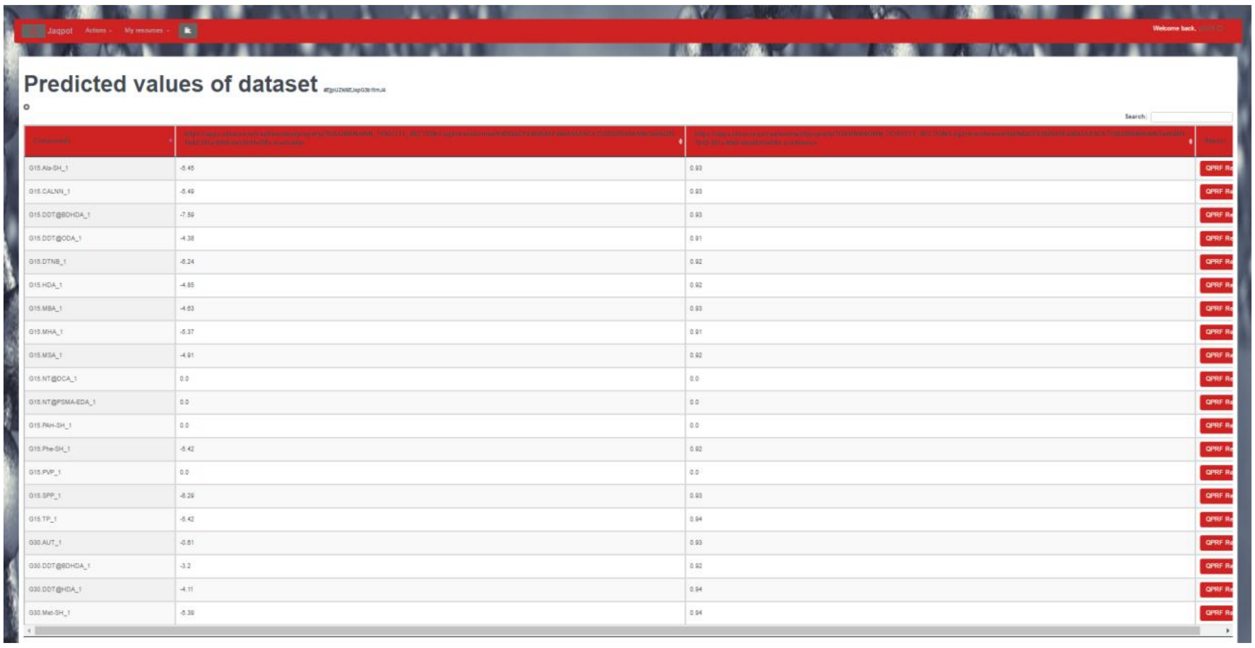
All Datasets:

Name	Title	Description	Date
QOz1xcU2Tdjt77vXGYKJ	Gajewicz et al - 18 Metal Oxide NPs	18 MeOx NPs with 29 descriptors, used for predicting HaCaT toxicity.	

Figure 20: Select dataset example of JQ read across prediction functionality

As described above, once a prediction is completed users may click on "See result". The screen will display the nanoparticles in the dataset used for prediction, along with the prediction feature (endpoint) and, if selected, the confidence value of the prediction. This is demonstrated in Figure 20 for which the dataset can be found at

http://test.jaqpot.org/predicted_dataset?name=EjpUZkl6EJxpG3b1tmJ4&model=shWpOGagqSJ5l6PfvkpJ. The prediction dataset created can be retrieved by the owner at any time under "My resources".



Predicted values of dataset EjpUZkl6EJxpG3b1tmJ4

[illegible]	[illegible]	[illegible]	[illegible]
G15.Ala-SH_1	-5.45	0.93	QPRF Re
G15.CALNN_1	-5.49	0.93	QPRF Re
G15.DDT@BDHDA_1	-7.59	0.93	QPRF Re
G15.DDT@ODA_1	-4.38	0.91	QPRF Re
G15.DTNB_1	-6.24	0.92	QPRF Re
G15.HDA_1	-4.85	0.92	QPRF Re
G15.MBA_1	-4.63	0.93	QPRF Re
G15.MHA_1	-5.37	0.91	QPRF Re
G15.MSA_1	-4.91	0.92	QPRF Re
G15.NT@DCA_1	0.0	0.0	QPRF Re
G15.NT@PSMA-EDA_1	0.0	0.0	QPRF Re
G15.PAH-SH_1	0.0	0.0	QPRF Re
G15.Phe-SH_1	-5.42	0.92	QPRF Re
G15.PVP_1	0.0	0.0	QPRF Re
G15.SPP_1	-8.29	0.93	QPRF Re
G15.TP_1	-5.42	0.94	QPRF Re
G30.AUT_1	-0.61	0.93	QPRF Re
G30.DDT@BDHDA_1	-3.2	0.92	QPRF Re
G30.DDT@HDA_1	-4.11	0.94	QPRF Re
G30.Met-SH_1	-5.39	0.94	QPRF Re

Figure 21: Predictions derived by JQ read across algorithm, along with the prediction confidence level between 0 and 1 (the closer to 1 the more confidence in the prediction).

4.2 Nano-Lazar Read Across SERVICES

4.2.1 NANO-LAZAR RUBY API

Nanoparticle model building, validation and predictions are now fully integrated in the lazar read across framework. Its primary use is to create and validate read across models in Ruby scripts or in an interactive Ruby shell. Installation and usage of the lazar framework is quite straightforward:

Installation:

```
    gem install lazar
```

Mirror eNM database into a local directory:

```
    require 'lazar'
    OpenTox::Import::Enanomapper.mirror enm_data_dir
```

Create and validate prediction model with default settings:

```
    model = OpenTox::Model::NanoPrediction.from_json_dump
enm_data_dir
```

This command creates a nano-lazar model and validates it with three independent 10-fold cross validations.

Inspect cross validations:

```
    model.crossvalidations
```

Predict a random nanoparticle from the training dataset:

```
query_particle = model.training_dataset.substances.sample

model.predict query_particle
```

Although lazar has reasonable and tested defaults for different QSAR problems, algorithms and parameters for similarity calculations and local models can be selected via a simple declarative interface. In order to create a model with custom algorithms and parameters we would execute for example:

```
algorithms = {
   :descriptors => {
     :method => "properties",
     :categories => ["P-CHEM","Proteomics"]
   },
  :prediction => {:method => 'Algorithm::Caret.pls' },
}

model.model = OpenTox::Model::Lazar.create(

  training_dataset: model.training_dataset,

  prediction_feature: model.prediction_feature,

  algorithms: algorithms

)
```

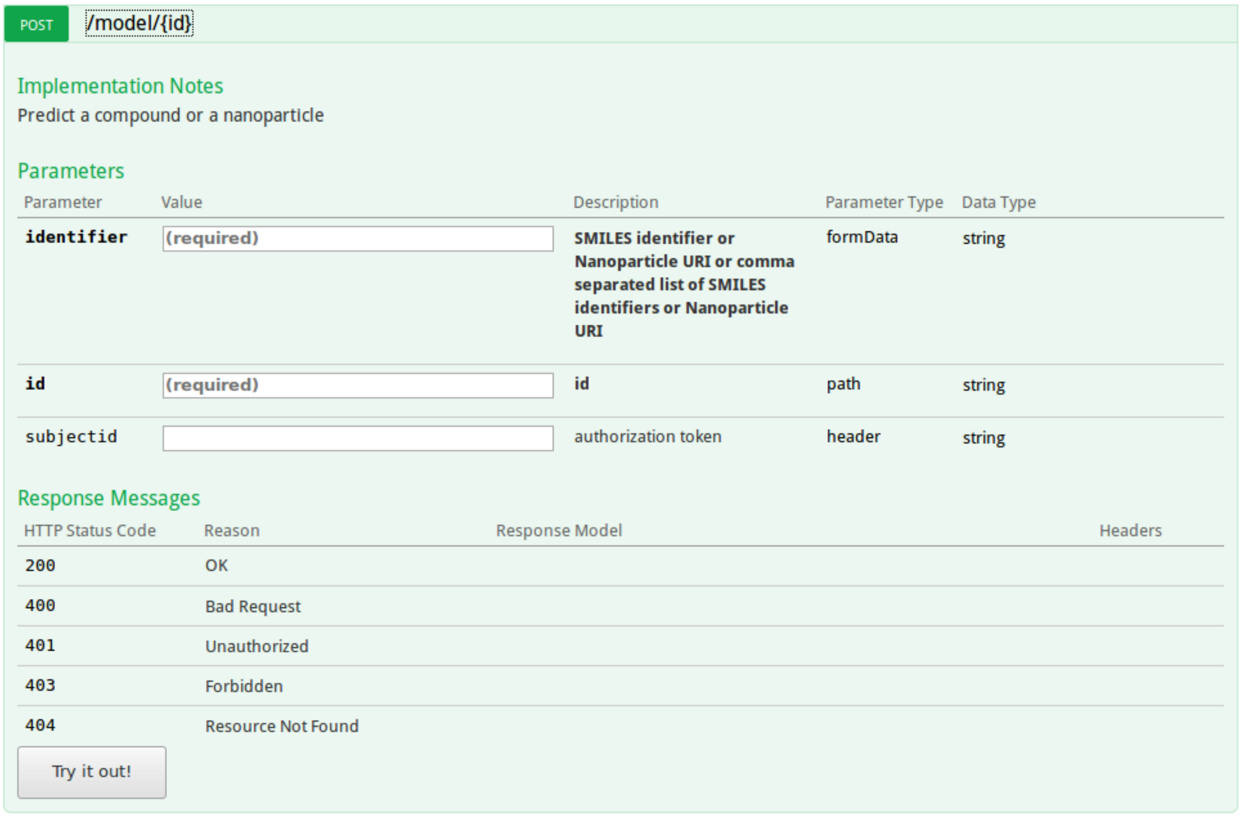
These commands create local partial least squares (PLS) models with physchem and proteomics descriptors.

The complete lazar Ruby API is documented at http://www.rubydoc.info/gems/lazar.

Although lazar is primarily designed for command line model development, cross validation results and nanoparticle predictions and other functions are also accessible from a web GUI (https://nano-lazar.in-silico.ch) and a REST interface (URI). Both are described in more detail in the following sections.

4.2.2 NANO-LAZAR REST API

The nano-lazar Rest API interface provides a Restful web service. It is the web service implementation of the nano-lazar Ruby API for obtaining information on data in the nano-lazar framework and for performing predictions and descriptor calculations. The web service is documented with Swagger 2.0 specification. Additionally a Swagger-UI web interface at https://enm.in-silico.ch/swagger/ provides a possibility to read and operate the API.


POST /model/{id}

Implementation Notes

Predict a compound or a nanoparticle

Parameters

Parameter	Value	Description	Parameter Type	Data Type
identifier	(required)	SMILES identifier or Nanoparticle URI or comma separated list of SMILES identifiers or Nanoparticle URI	formData	string
id	(required)	id	path	string
subjectid		authorization token	header	string

Response Messages

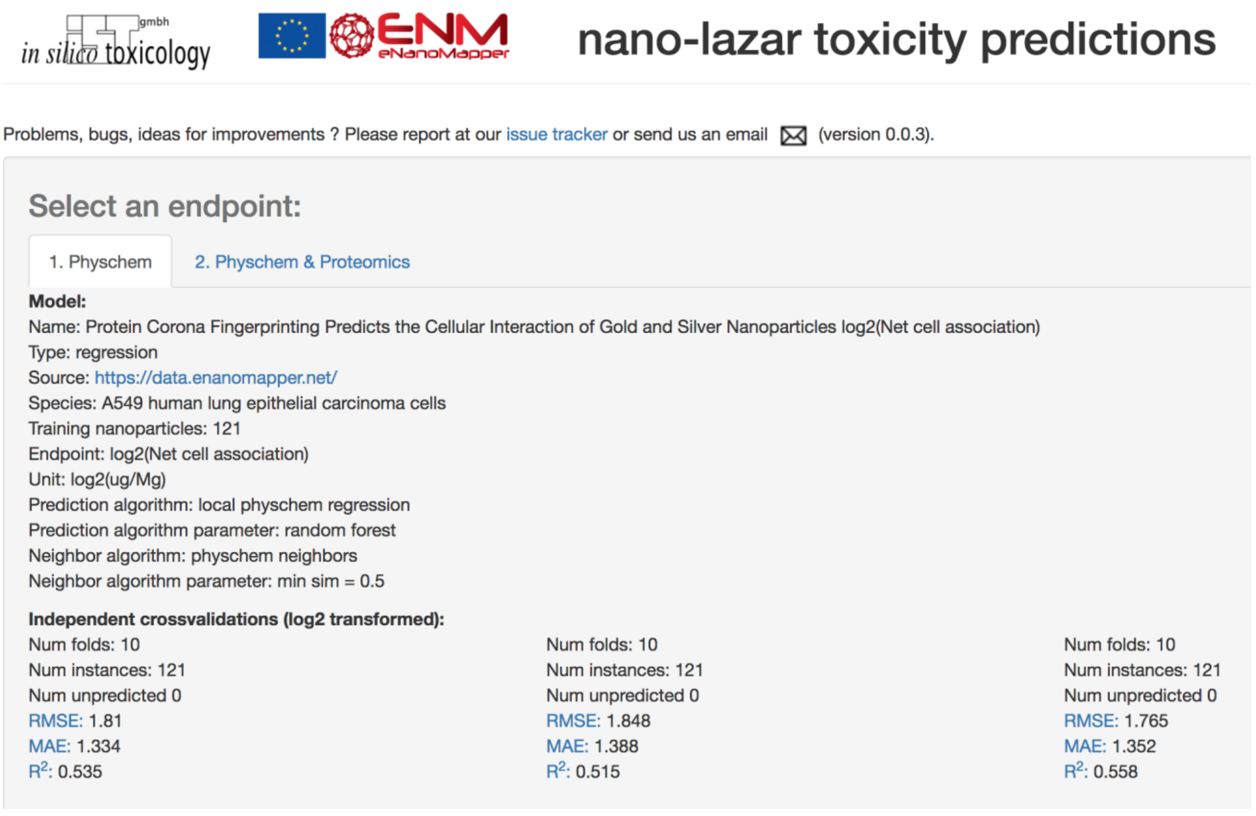
HTTP Status Code	Reason	Response Model	Headers
200	OK		
400	Bad Request		
401	Unauthorized		
403	Forbidden		
404	Resource Not Found		

Figure 22: Predict a compound or nanoparticle

4.2.3 NANO-LAZAR GUI

The nano-lazar GUI (https://nano-lazar.in-silico.ch/) is tightly integrated with the eNM infrastructure. It uses data from the eNM data warehouse for model building and eNM ontologies for data annotation and GUI elements (e.g. description of domain specific terms).

Figure 22 shows a screenshot of the nano-lazar query page with a summary of cross validation results and entry forms for nanoparticle properties.


in silico toxicology gmbh

nano-lazar toxicity predictions

Problems, bugs, ideas for improvements ? Please report at our issue tracker or send us an email (version 0.0.3).

Select an endpoint:

1. Physchem | 2. Physchem & Proteomics

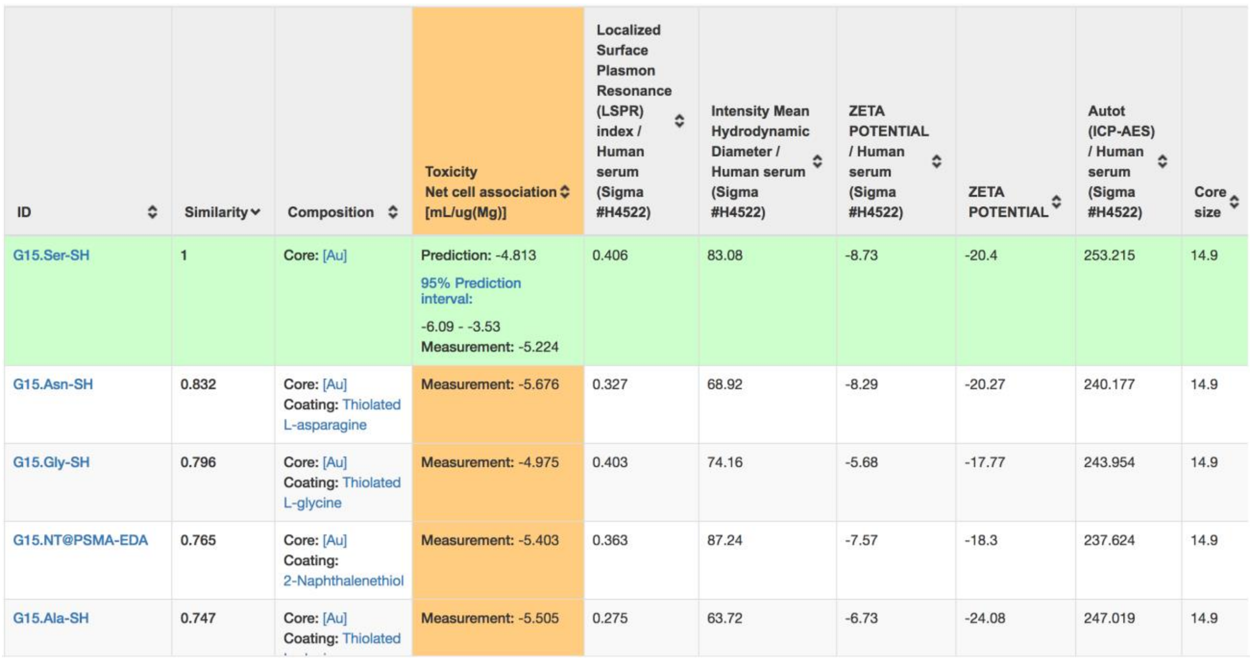
Model:
Name: Protein Corona Fingerprinting Predicts the Cellular Interaction of Gold and Silver Nanoparticles log2(Net cell association)
Type: regression
Source: https://data.enanomapper.net/
Species: A549 human lung epithelial carcinoma cells
Training nanoparticles: 121
Endpoint: log2(Net cell association)
Unit: log2(ug/Mg)
Prediction algorithm: local physchem regression
Prediction algorithm parameter: random forest
Neighbor algorithm: physchem neighbors
Neighbor algorithm parameter: min sim = 0.5

Independent crossvalidations (log2 transformed):

Num folds: 10	Num folds: 10	Num folds: 10
Num instances: 121	Num instances: 121	Num instances: 121
Num unpredicted 0	Num unpredicted 0	Num unpredicted 0
RMSE: 1.81	RMSE: 1.848	RMSE: 1.765
MAE: 1.334	MAE: 1.388	MAE: 1.352
R^2: 0.535	R^2: 0.515	R^2: 0.558

Figure 23: Summary of crossvalidation results

Nano-lazar predictions are depicted on the following screenshot (Figure 23). nano-lazar presents predictions in tabular form with the query nanoparticle on top followed by similar nanoparticles for the local QSAR model. Each nanoparticle is characterised by its similarity to the query compound, endpoint measurements and the properties relevant for the particular endpoint. If the query nanoparticle is part of the training data (as in this example) all information from this substance is removed from the training set before making the prediction. In this case the measured value is displayed for comparison purposes.



ID	Similarity	Composition	Toxicity Net cell association [mL/ug(Mg)]	Localized Surface Plasmon Resonance (LSPR) index / Human serum (Sigma #H4522)	Intensity Mean Hydrodynamic Diameter / Human serum (Sigma #H4522)	ZETA POTENTIAL / Human serum (Sigma #H4522)	ZETA POTENTIAL	Autot (ICP-AES) / Human serum (Sigma #H4522)	Core size
G15.Ser-SH	1	Core: [Au]	Prediction: -4.813 95% Prediction interval: -6.09 - -3.53 Measurement: -5.224	0.406	83.08	-8.73	-20.4	253.215	14.9
G15.Asn-SH	0.832	Core: [Au] Coating: Thiolated L-asparagine	Measurement: -5.676	0.327	68.92	-8.29	-20.27	240.177	14.9
G15.Gly-SH	0.796	Core: [Au] Coating: Thiolated L-glycine	Measurement: -4.975	0.403	74.16	-5.68	-17.77	243.954	14.9
G15.NT@PSMA-EDA	0.765	Core: [Au] Coating: 2-Naphthalenethiol	Measurement: -5.403	0.363	87.24	-7.57	-18.3	237.624	14.9
G15.Ala-SH	0.747	Core: [Au] Coating: Thiolated	Measurement: -5.505	0.275	63.72	-6.73	-24.08	247.019	14.9

Figure 24: nano-lazar prediction.

5. QPRF Reporting Services

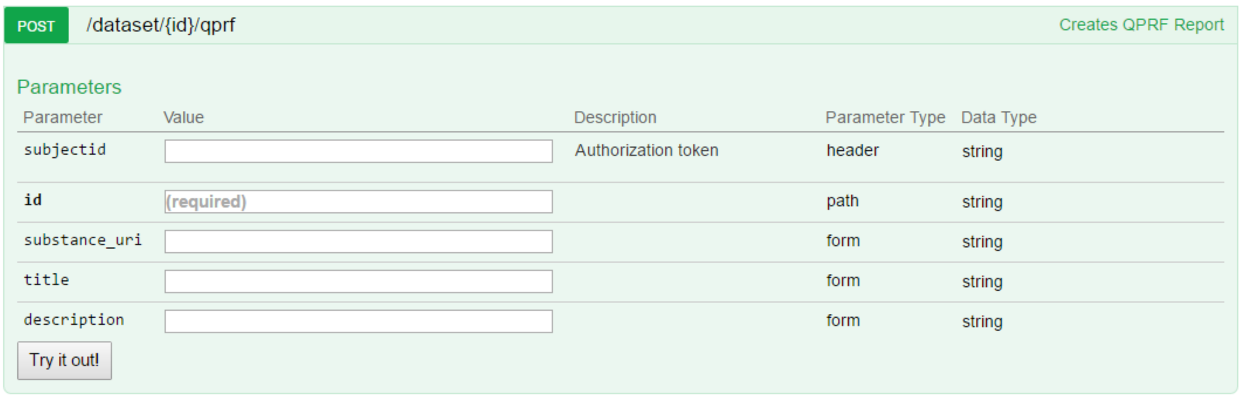
5.1 QPRF Reports In The JQ Modelling Framework

QPRF reports may be generated for each ENM after a prediction has been performed. These contain all fields set by the OECD guidelines [1], namely:

- Substance: Contains information such as CAS and EC numbers, SMILES, InChi, etc.
- General: information such as date and creator name and email.
- Prediction: Biological endpoint, variables, model used, DoA, etc
- Adequacy: Optional field, containing regulatory purpose, conclusion etc.

All fields made available from the dataset, as well as information about the model and the prediction are added to the report automatically. Should information be missing, what is expected to be filled in is described by the field. All fields are editable, in order to allow the user to add extra information not contained in the dataset or model, as well as the optional fields such as conclusion1s. As with all reports, QPRF reports are downloadable as PDF documents.

The Swagger documentation of the JQ QPRF API is shown in Figure 25. Users must POST the dataset ID and substance URI (nanoparticle in question) in order to receive the QPRF report. Once the task generated has been completed, users may view the report using the GET Report by ID API.

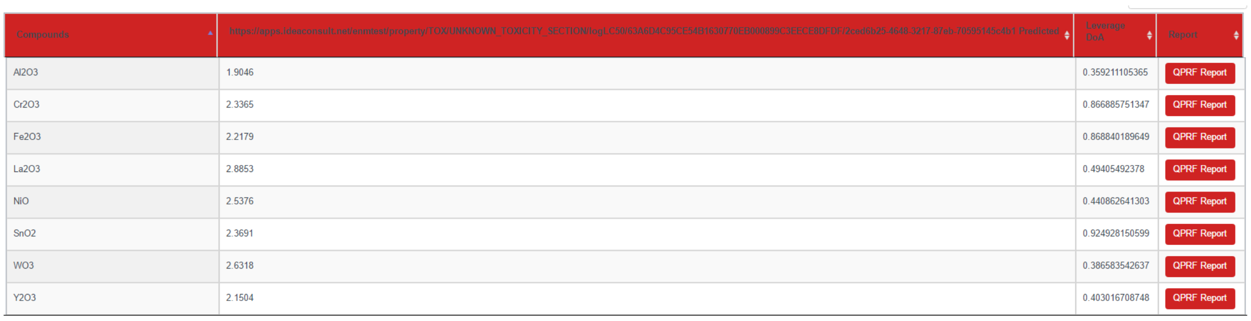

POST /dataset/{id}/qprf Creates QPRF Report

Parameters

Parameter	Value	Description	Parameter Type	Data Type
subjectid		Authorization token	header	string
id	(required)		path	string
substance_uri			form	string
title			form	string
description			form	string

Figure 25: Produce QPRF report using JQ's POST Dataset ID QPRF API

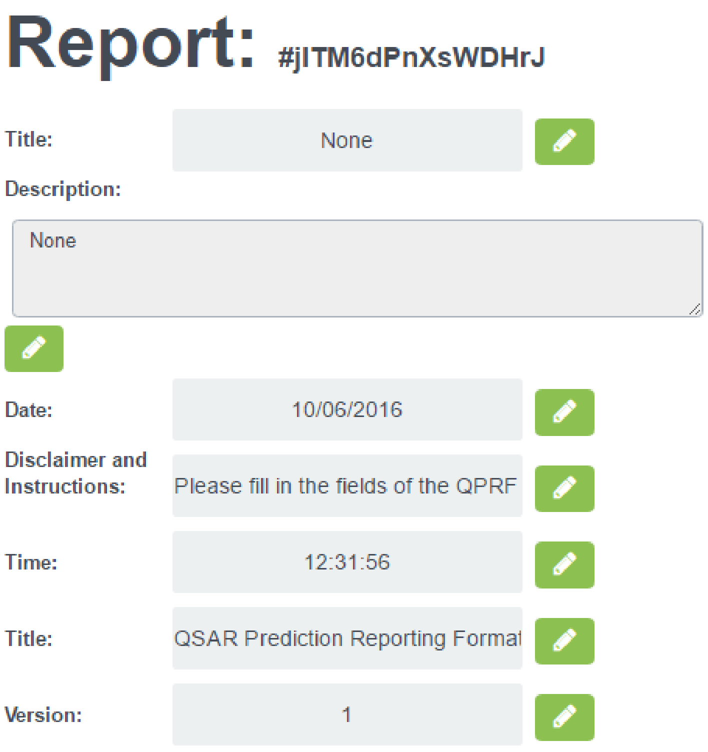
In order to receive a QPRF report, a user must have made a prediction (by selecting a model and dataset; as demonstrated in previous deliverables). Upon clicking on “See result” the prediction dataset has an extra column on the far right, namely “QPRF Report”, as seen in Figure 25. By clicking on this, they are redirected to the QPRF report page. The report (as with any other report) may be downloaded as a PDF. For this demonstration, the report can be viewed using the JQ UI at http://test.jaqpot.org:8000/report?name=ALZ8LDBrwXrCtsY.



Compounds	https://apps.ideaconsult.net/enmtest/property/TOX/UNKNOWN_TOXICITY_SECTION/logLC50/63A6D4C95CE54B1630770EB000899C3EECE8DFDF/2ced6b25-4648-3217-87eb-70595145c4b1 Predicted	Leverage DoA	Report
Al2O3	1.9046	0.359211105365	QPRF Report
Cr2O3	2.3365	0.866885751347	QPRF Report
Fe2O3	2.2179	0.868840189649	QPRF Report
La2O3	2.8853	0.49405492378	QPRF Report
NiO	2.5376	0.440862641303	QPRF Report
SnO2	2.3691	0.924928150599	QPRF Report
WO3	2.6318	0.386583542637	QPRF Report
Y2O3	2.1504	0.403016708748	QPRF Report

Figure 26: Prediction screen containing QPRF report for each nanoparticle (Gajewicz et al. dataset, 8 MeOX NPs for predicting HaCaT toxicity).

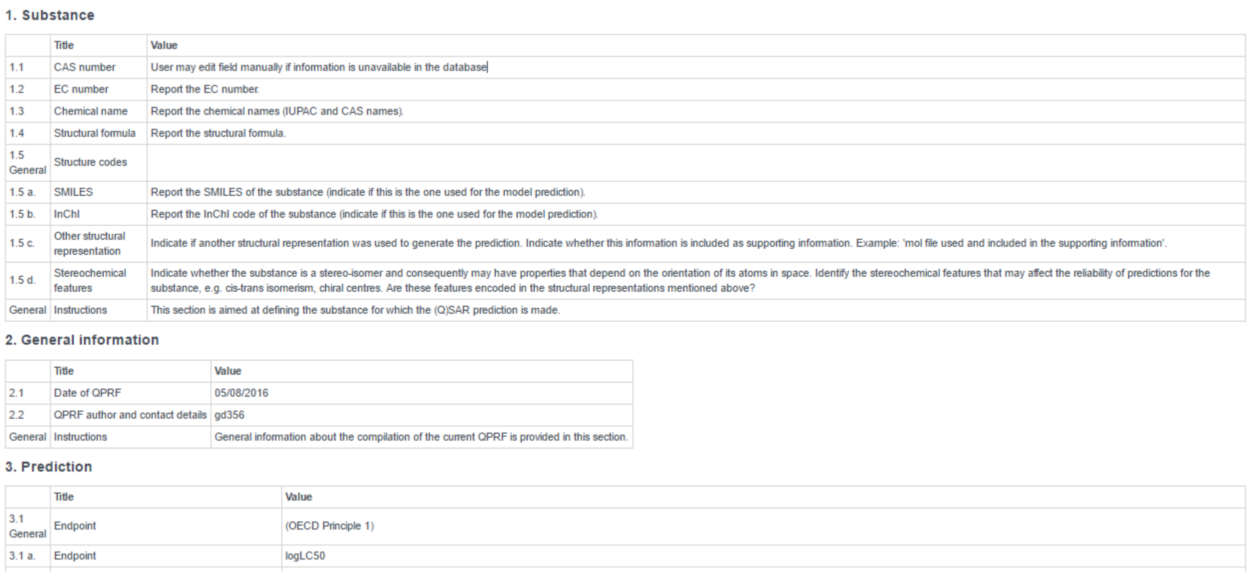
We reiterate that all entries in the report are editable. This functionality has been added to allow users to fill in gaps according to their knowledge, should the entries not exist in the database, but also complete fields such as experiment purpose or conclusion which are not automatically generated. This can be seen in Figure 27-Figure 28.



Report: #jlTM6dPnXsWDHrJ

Title:	None
Description:	None
Date:	10/06/2016
Disclaimer and Instructions:	Please fill in the fields of the QPRF
Time:	12:31:56
Title:	QSAR Prediction Reporting Format
Version:	1

Figure 27: Top section of report shown in JQ UI. It can be seen all fields are editable.



1. Substance

	Title	Value
1.1	CAS number	User may edit field manually if information is unavailable in the database
1.2	EC number	Report the EC number.
1.3	Chemical name	Report the chemical names (IUPAC and CAS names).
1.4	Structural formula	Report the structural formula.
1.5 General	Structure codes	
1.5 a.	SMILES	Report the SMILES of the substance (indicate if this is the one used for the model prediction).
1.5 b.	InChI	Report the InChI code of the substance (indicate if this is the one used for the model prediction).
1.5 c.	Other structural representation	Indicate if another structural representation was used to generate the prediction. Indicate whether this information is included as supporting information. Example: 'mol file used and included in the supporting information'.
1.5 d.	Stereochemical features	Indicate whether the substance is a stereo-isomer and consequently may have properties that depend on the orientation of its atoms in space. Identify the stereochemical features that may affect the reliability of predictions for the substance, e.g. cis-trans isomerism, chiral centres. Are these features encoded in the structural representations mentioned above?
General	Instructions	This section is aimed at defining the substance for which the (Q)SAR prediction is made.

2. General information

	Title	Value
2.1	Date of QPRF	05/08/2016
2.2	QPRF author and contact details	gd356
General	Instructions	General information about the compilation of the current QPRF is provided in this section.

3. Prediction

	Title	Value
3.1 General	Endpoint	(OECD Principle 1)
3.1 a.	Endpoint	logLC50

Figure 28: Screenshot of fields available in JQ QPRF report service viewed via the UI

5.2 QPRF Reports In Nano-Lazar

With the Ruby QSAR reporting library QSAR-report (see 6.1 Ruby QSAR reporting library) it is possible to produce QPRF reports in ruby applications. The library generates QPRF reports in Version 1.1[1] in JSON format. In the upcoming months, until M36 of the project, we will implement the QPRF reporting feature into nano-lazar. A full documentation of the QPRF library usage is documented at RubyDocs.org[2].

[1] EC Joint Research Center: QSAR Prediction Reporting Format (QPRF) (version 1.1, May 2008) https://eurl-ecvam.jrc.ec.europa.eu/laboratories-research/predictive_toxicology/qsar_tools/qrf/QPRF_version_1.1.pdf

[2] RubyDocs: QSAR-report, Class: OpenTox::QPRFReport

http://www.rubydoc.info/gems/qsar-report/OpenTox/QPRFReport

6. QMRF Reporting Services

6.1 Ruby QSAR report library

We have developed a ruby library to produce QMRF and QPRF reports. As for many programming languages, the Ruby language has a huge variety of libraries and it supports the inclusion of third-party libraries. These libraries are mostly used in ruby code by so called gems. Gems are ruby library packages hosted in the public ruby community's gem hosting service at https://rubygems.org[3].

The QSAR-report gem was developed to extend the lazar and nano-lazar toxicity prediction programs with QMRF and QPRF reporting features. The library gem is independent from lazar or nano-lazar and can also be used in any other ruby code. The library code is in general public license version 3 (GPLv3[4]) . The open source code is freely available and versioned at the Github[5] code hosting platform.

The library has two main classes OpenTox::QMRFReport and OpenTox::QPRFReport for the report types QMRF and QPRF. Instances of these classes can be easily filled with contents. There is a full documentation of the library with examples at RubyDoc.info[6] .

For full information on QSAR reporting see: JRC QSAR Model Database and QSAR Model Reporting Formats https://eurl-ecvam.jrc.ec.europa.eu/databases/jrc-qsar-model-database-and-qsar-model-reporting-formats

6.2 QMRF report in nano-lazar

The QMRF reporting feature is implemented in eNanoMapper's nano-lazar REST API[7] and the Nano-Lazar toxicity prediction web-application[8]. REST API operations can be operated in a Swagger based API

[3] RubyGems the Ruby community's gem hosting service - QSAR report gem: https://rubygems.org/gems/qsar-report/

[4] GNU General Public License 3 https://www.gnu.org/licenses/gpl-3.0.en.html

[5] Github code hosting platform https://github.com/opentox/qsar-report

[6] RubyDoc.info YARD documentation server - QSAR report gem: http://www.rubydoc.info/gems/qsar-report

[7] Lazar & Nano-Lazar REST Service API https://enm.in-silico.ch/swagger

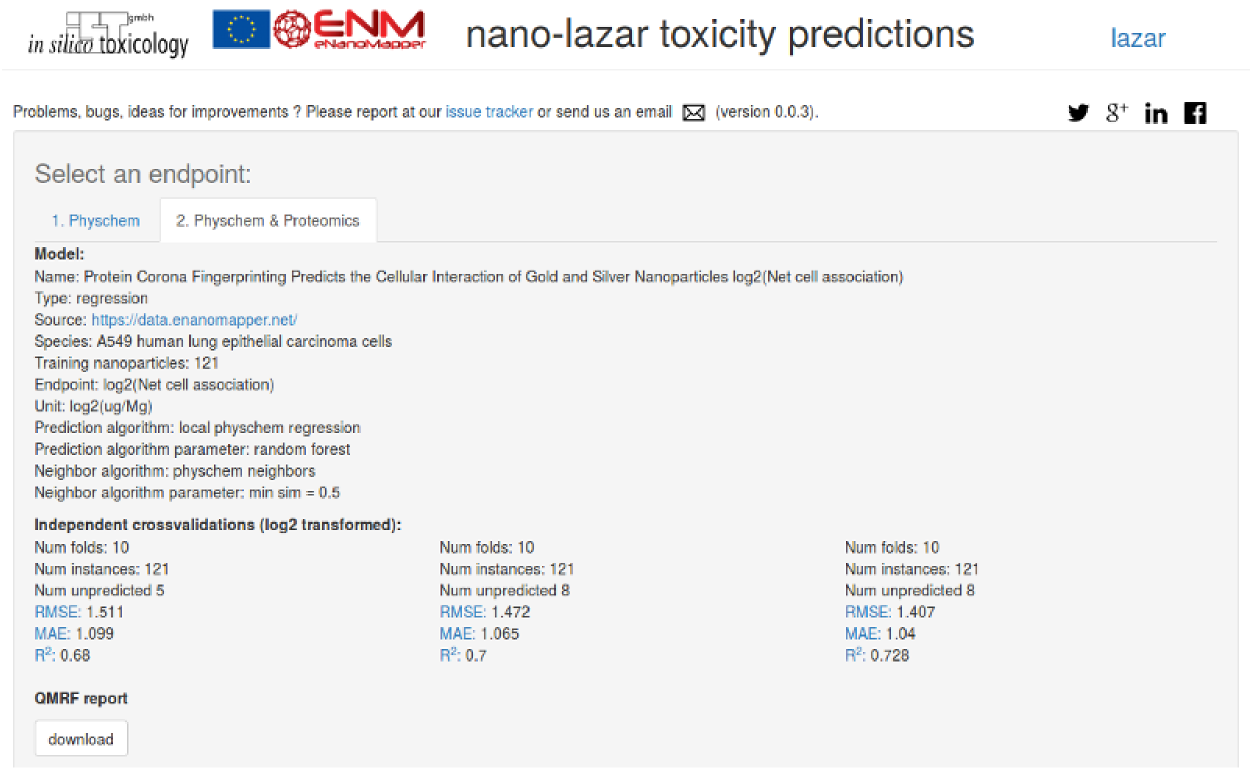
[8] Nano-lazar toxicity prediction https://nano-lazar.in-silico.ch/predict (current version) https://nano-lazar-dev.in-silico.ch/predict (development version)

documentation with a Swagger UI interface. QMRF reports can be generated for each lazar model by performing a GET request to the specific model.

```
curl -X GET --header "Accept:application/xml" https://enm.in-
silico.ch/report/57dbdc7b4375ab563b601b3a
```

curl example

The QMRF report produces QMRF XML in version 1.3. The API generates a first version of the report. This 'prototype' can be downloaded in XML format. This XML file can be finalized for publication in the QMRF Editor 2.0 (https://sourceforge.net/projects/qmrf/). Example code[9] and full documentation can be found at RubyDoc.info .


in silico toxicology gmbh

nano-lazar toxicity predictions

lazar

Problems, bugs, ideas for improvements ? Please report at our issue tracker or send us an email (version 0.0.3).

Select an endpoint:

1. Physchem | 2. Physchem & Proteomics

Model:
Name: Protein Corona Fingerprinting Predicts the Cellular Interaction of Gold and Silver Nanoparticles log2(Net cell association)
Type: regression
Source: https://data.enanomapper.net/
Species: A549 human lung epithelial carcinoma cells
Training nanoparticles: 121
Endpoint: log2(Net cell association)
Unit: log2(ug/Mg)
Prediction algorithm: local physchem regression
Prediction algorithm parameter: random forest
Neighbor algorithm: physchem neighbors
Neighbor algorithm parameter: min sim = 0.5

Independent crossvalidations (log2 transformed):

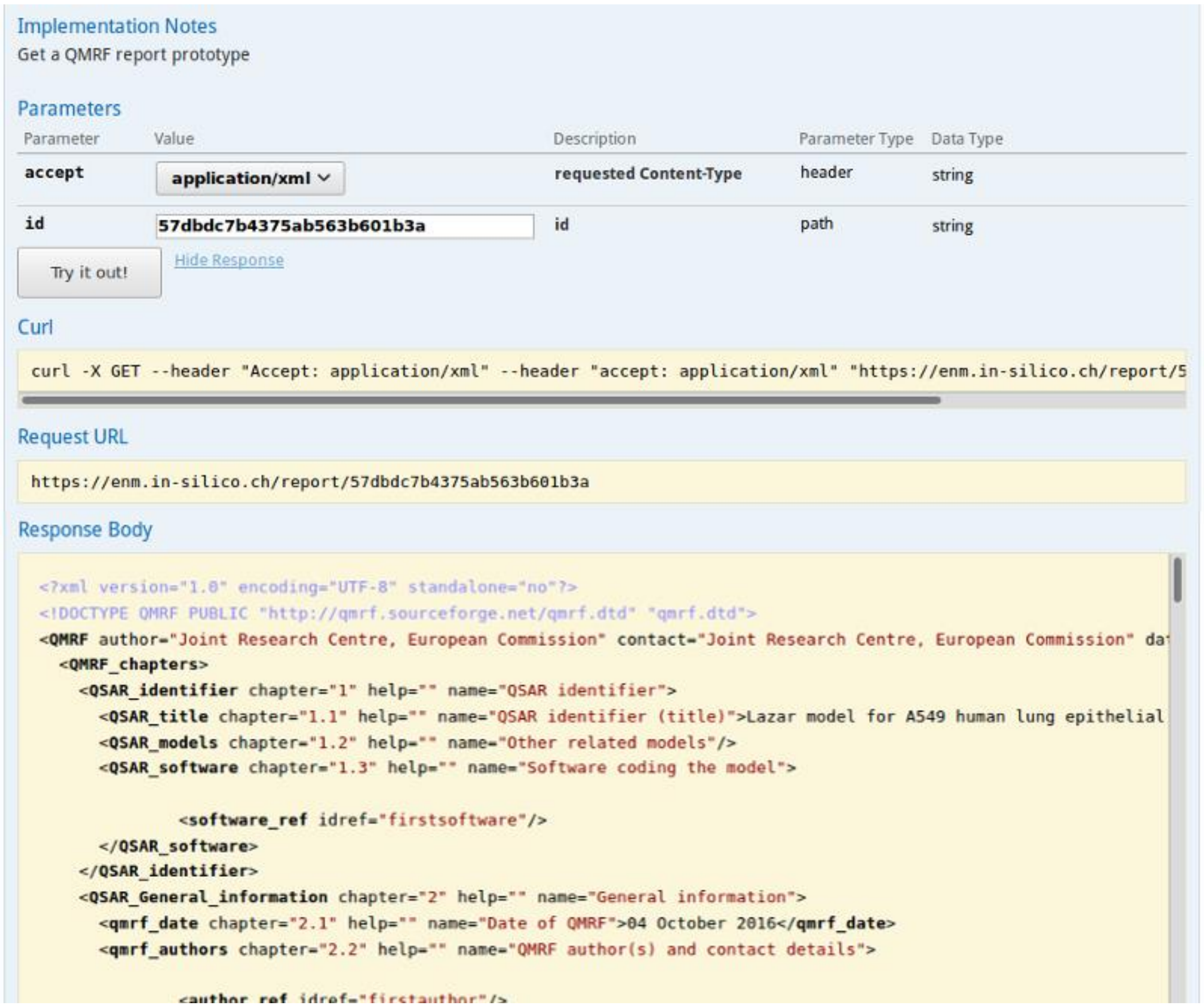
Num folds: 10	Num folds: 10	Num folds: 10
Num instances: 121	Num instances: 121	Num instances: 121
Num unpredicted 5	Num unpredicted 8	Num unpredicted 8
RMSE: 1.511	RMSE: 1.472	RMSE: 1.407
MAE: 1.099	MAE: 1.065	MAE: 1.04
R^2: 0.68	R^2: 0.7	R^2: 0.728

QMRF report

download

Figure 29: QMRF report integration in the nano-lazar toxicity prediction web application

[9] Nano-lazar example code: http://www.rubydoc.info/gems/qsar-report/file/example/example.rb



Implementation Notes

Get a QMRF report prototype

Parameters

Parameter	Value	Description	Parameter Type	Data Type
accept	application/xml	requested Content-Type	header	string
id	57dbdc7b4375ab563b601b3a	id	path	string

Curl

```
curl -X GET --header "Accept: application/xml" --header "accept: application/xml" "https://enm.in-silico.ch/report/5
```

Request URL

```
https://enm.in-silico.ch/report/57dbdc7b4375ab563b601b3a
```

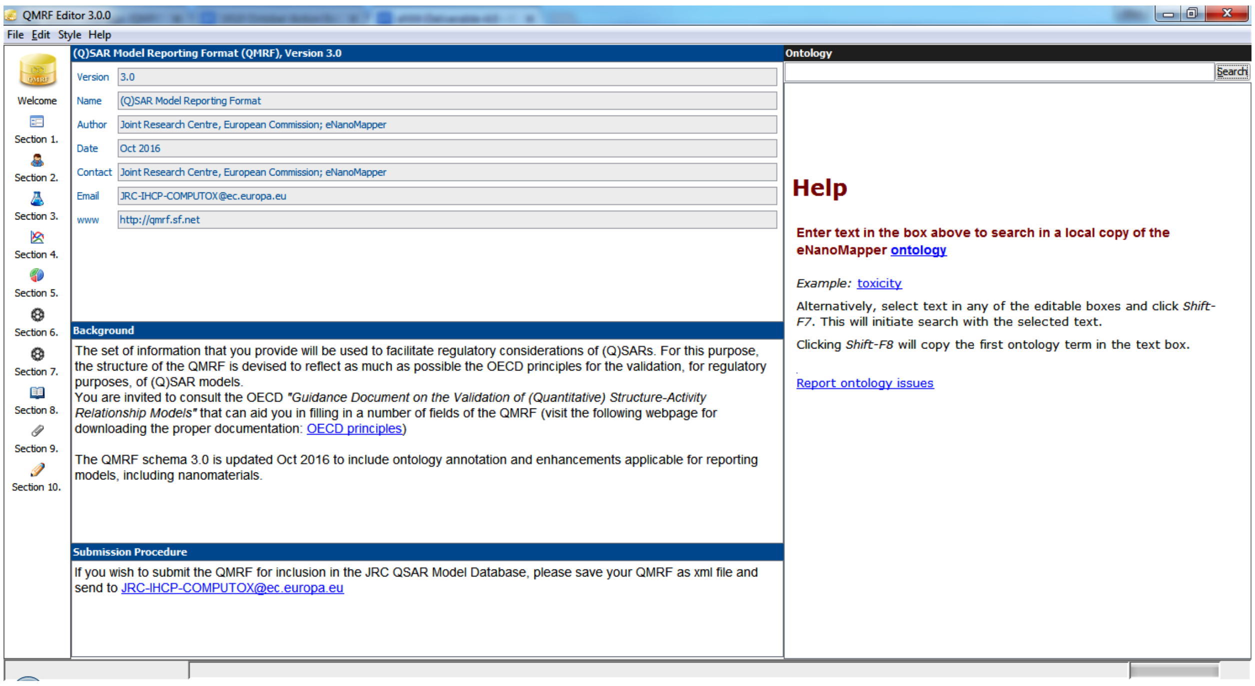
Response Body

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE QMRF PUBLIC "http://qmrf.sourceforge.net/qmrf.dtd" "qmrf.dtd">
<QMRF author="Joint Research Centre, European Commission" contact="Joint Research Centre, European Commission" da
  <QMRF_chapters>
    <QSAR_identifier chapter="1" help="" name="QSAR identifier">
      <QSAR_title chapter="1.1" help="" name="QSAR identifier (title)">Lazar model for A549 human lung epithelial
      <QSAR_models chapter="1.2" help="" name="Other related models"/>
      <QSAR_software chapter="1.3" help="" name="Software coding the model">

              <software_ref idref="firstsoftware"/>
      </QSAR_software>
    </QSAR_identifier>
    <QSAR_General_information chapter="2" help="" name="General information">
      <qmrf_date chapter="2.1" help="" name="Date of QMRF">04 October 2016</qmrf_date>
      <qmrf_authors chapter="2.2" help="" name="QMRF author(s) and contact details">

              <author_ref idref="firstauthor"/>
```

Figure 30: Getting QMRF report from nano-lazar API

6.3 QMRF EDITOR 3.0.0

The QMRF XML schema and the QMRF Editor was originally developed in 2007 on behalf of JRC by Ideaconsult Ltd. The QMRF Editor 2.0.0[10] was available for download since 2013. Within the framework of eNanoMapper, the qmrf.dtd schema was updated to support ontology annotation and include indication for nanomaterials in training and test set. The list of endpoints is modified to include separate field for protocols (assays) related to an endpoint. The list of endpoints is updated to contain the recently released (April 2016) OECD Harmonized templates and list of assays from the publicly released NANoREG templates[11]. A PMML support is also introduced, allowing to load PMML fields, specifying descriptor and software into the QMRFEditor.



(Q)SAR Model Reporting Format (QMRF), Version 3.0

Version	3.0
Name	(Q)SAR Model Reporting Format
Author	Joint Research Centre, European Commission; eNanoMapper
Date	Oct 2016
Contact	Joint Research Centre, European Commission; eNanoMapper
Email	JRC-IHCP-COMPUTOX@ec.europa.eu
www	http://qmrf.sf.net

Background

The set of information that you provide will be used to facilitate regulatory considerations of (Q)SARs. For this purpose, the structure of the QMRF is devised to reflect as much as possible the OECD principles for the validation, for regulatory purposes, of (Q)SAR models.
You are invited to consult the OECD *"Guidance Document on the Validation of (Quantitative) Structure-Activity Relationship Models"* that can aid you in filling in a number of fields of the QMRF (visit the following webpage for downloading the proper documentation: OECD principles)

The QMRF schema 3.0 is updated Oct 2016 to include ontology annotation and enhancements applicable for reporting models, including nanomaterials.

Submission Procedure

If you wish to submit the QMRF for inclusion in the JRC QSAR Model Database, please save your QMRF as xml file and send to JRC-IHCP-COMPUTOX@ec.europa.eu

Help

Enter text in the box above to search in a local copy of the eNanoMapper ontology

Example: toxicity

Alternatively, select text in any of the editable boxes and click *Shift-F7*. This will initiate search with the selected text.

Clicking *Shift-F8* will copy the first ontology term in the text box.

Report ontology issues

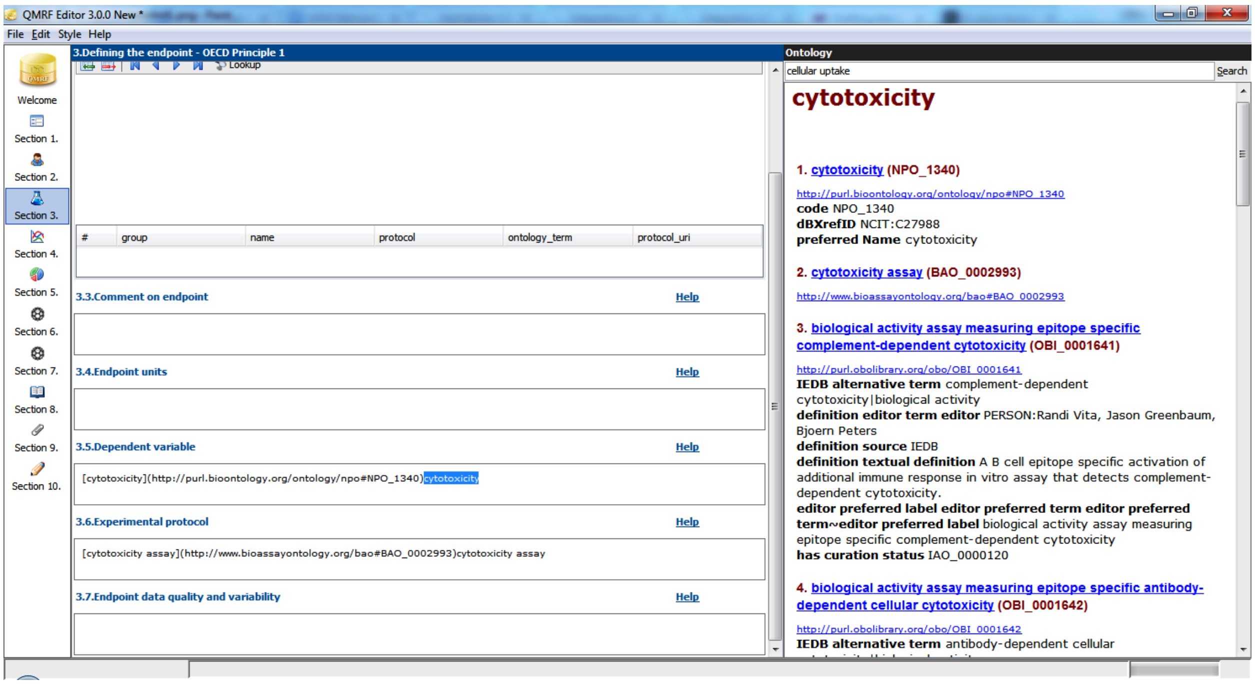
Figure 31: QMRF Editor 3.0.0

[10] https://sourceforge.net/projects/qmrf/files/QMRF%20Editor/2.0.0/

[11] http://www.nanoreg.eu/media-and-downloads/templates/269-templates-for-experimental-data-logging

The QMRF Editor 3.0.0 includes a new panel, allowing free text search of a local copy of the eNanoMapper ontology. The query is entered into a text box at the top and the results are displayed as clickable entries. Clicking on any of the subsequent hits will move it to the first place (by launching a search for itself). While the cursor is in any editable text box in QMRF chapters, selecting a text (e.g. "cytotoxicity") and clicking *Shift+F7* launches a search with the selected text and shows the results in the right panel. Clicking *Shift+F8* will paste the name and the ontology URI of the first hit, e.g.

[cytotoxicity](http://purl.bioontology.org/ontology/npo#NPO_1340)



3.Defining the endpoint - OECD Principle 1

#	group	name	protocol	ontology_term	protocol_uri

3.3.Comment on endpoint

3.4.Endpoint units

3.5.Dependent variable

[cytotoxicity](http://purl.bioontology.org/ontology/npo#NPO_1340)cytotoxicity

3.6.Experimental protocol

[cytotoxicity assay](http://www.bioassayontology.org/bao#BAO_0002993)cytotoxicity assay

3.7.Endpoint data quality and variability

Ontology

cellular uptake

cytotoxicity

1. cytotoxicity (NPO_1340)

http://purl.bioontology.org/ontology/npo#NPO_1340
code NPO_1340
dBXrefID NCIT:C27988
preferred Name cytotoxicity

2. cytotoxicity assay (BAO_0002993)

http://www.bioassayontology.org/bao#BAO_0002993

3. biological activity assay measuring epitope specific complement-dependent cytotoxicity (OBI_0001641)

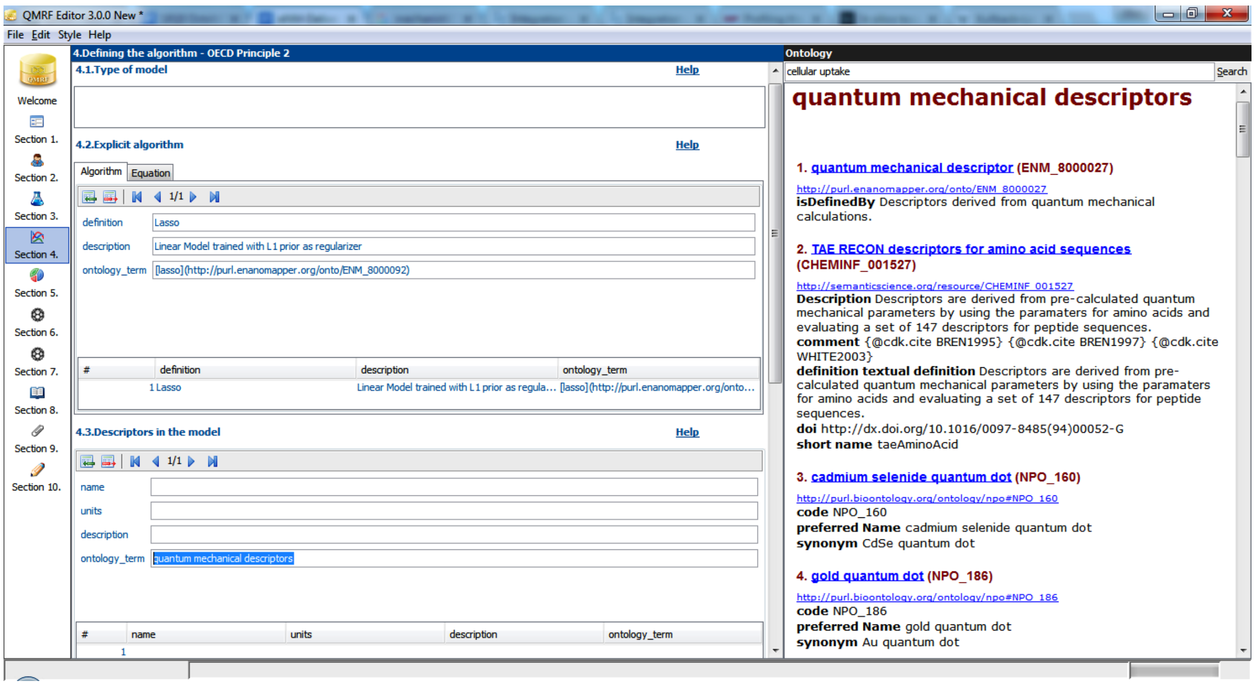
http://purl.obolibrary.org/obo/OBI_0001641
IEDB alternative term complement-dependent cytotoxicity|biological activity
definition editor term editor PERSON:Randi Vita, Jason Greenbaum, Bjoern Peters
definition source IEDB
definition textual definition A B cell epitope specific activation of additional immune response in vitro assay that detects complement-dependent cytotoxicity.
editor preferred label editor preferred term editor preferred term~editor preferred label biological activity assay measuring epitope specific complement-dependent cytotoxicity
has curation status IAO_0000120

4. biological activity assay measuring epitope specific antibody-dependent cellular cytotoxicity (OBI_0001642)

http://purl.obolibrary.org/obo/OBI_0001642
IEDB alternative term antibody-dependent cellular

Figure 32: QMRF Editor 3.0.0 Defining the endpoint (1)

A field for ontology terms is added to the section *1.3. Software coding the model*, *3.2. Endpoint, 4.2. Explicit Algorithm*, *4.3 Descriptors in the model*, *4.6. Software name and version for descriptor generation*, *5.3. Software name and version for applicability domain assessment*.



4.Defining the algorithm - OECD Principle 2

4.1.Type of model

4.2.Explicit algorithm

definition: Lasso
description: Linear Model trained with L1 prior as regularizer
ontology_term: [lasso](http://purl.enanomapper.org/onto/ENM_8000092)

#	definition	description	ontology_term
1	Lasso	Linear Model trained with L1 prior as regula…	[lasso](http://purl.enanomapper.org/onto…

4.3.Descriptors in the model

name:
units:
description:
ontology_term: quantum mechanical descriptors

#	name	units	description	ontology_term
1				

cellular uptake

quantum mechanical descriptors

1. quantum mechanical descriptor (ENM_8000027)
http://purl.enanomapper.org/onto/ENM_8000027
isDefinedBy Descriptors derived from quantum mechanical calculations.

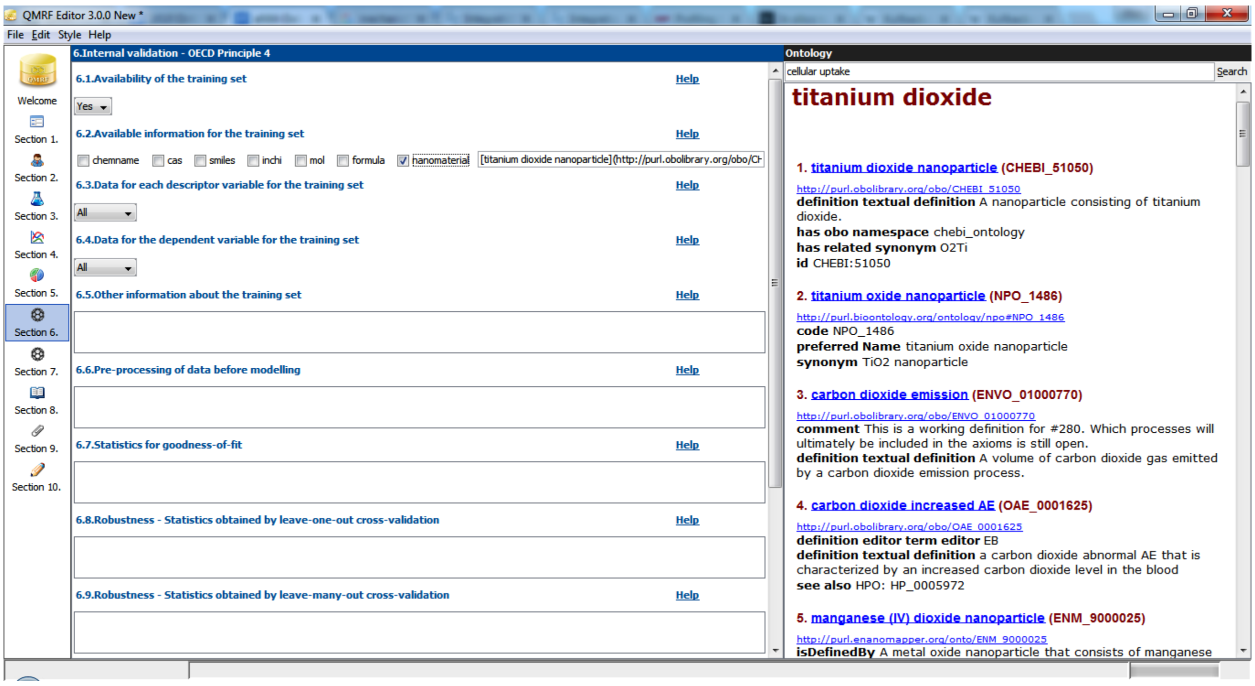
2. TAE RECON descriptors for amino acid sequences (CHEMINF_001527)
http://semanticscience.org/resource/CHEMINF_001527
Description Descriptors are derived from pre-calculated quantum mechanical parameters by using the paramaters for amino acids and evaluating a set of 147 descriptors for peptide sequences.
comment {@cdk.cite BREN1995} {@cdk.cite BREN1997} {@cdk.cite WHITE2003}
definition textual definition Descriptors are derived from pre-calculated quantum mechanical parameters by using the paramaters for amino acids and evaluating a set of 147 descriptors for peptide sequences.
doi http://dx.doi.org/10.1016/0097-8485(94)00052-G
short name taeAminoAcid

3. cadmium selenide quantum dot (NPO_160)
http://purl.bioontology.org/ontology/npo#NPO_160
code NPO_160
preferred Name cadmium selenide quantum dot
synonym CdSe quantum dot

4. gold quantum dot (NPO_186)
http://purl.bioontology.org/ontology/npo#NPO_186
code NPO_186
preferred Name gold quantum dot
synonym Au quantum dot

Figure 33: QMRF Editor 3.0.0 Defining the algorithm

Nanomaterial indication for training and test datasets is added.



QMRF Editor 3.0.0 New *

File Edit Style Help

Welcome · Section 1. · Section 2. · Section 3. · Section 4. · Section 5. · Section 6. · Section 7. · Section 8. · Section 9. · Section 10.

6.Internal validation - OECD Principle 4

6.1.Availability of the training set — Help

Yes

6.2.Available information for the training set — Help

☐ chemname ☐ cas ☐ smiles ☐ inchi ☐ mol ☐ formula ☑ nanomaterial [titanium dioxide nanoparticle](http://purl.obolibrary.org/obo/CH

6.3.Data for each descriptor variable for the training set — Help

All

6.4.Data for the dependent variable for the training set — Help

All

6.5.Other information about the training set — Help

6.6.Pre-processing of data before modelling — Help

6.7.Statistics for goodness-of-fit — Help

6.8.Robustness - Statistics obtained by leave-one-out cross-validation — Help

6.9.Robustness - Statistics obtained by leave-many-out cross-validation — Help

Ontology

cellular uptake — Search

titanium dioxide

1. titanium dioxide nanoparticle (CHEBI_51050)
http://purl.obolibrary.org/obo/CHEBI_51050
definition textual definition A nanoparticle consisting of titanium dioxide.
has obo namespace chebi_ontology
has related synonym O2Ti
id CHEBI:51050

2. titanium oxide nanoparticle (NPO_1486)
http://purl.bioontology.org/ontology/npo#NPO_1486
code NPO_1486
preferred Name titanium oxide nanoparticle
synonym TiO2 nanoparticle

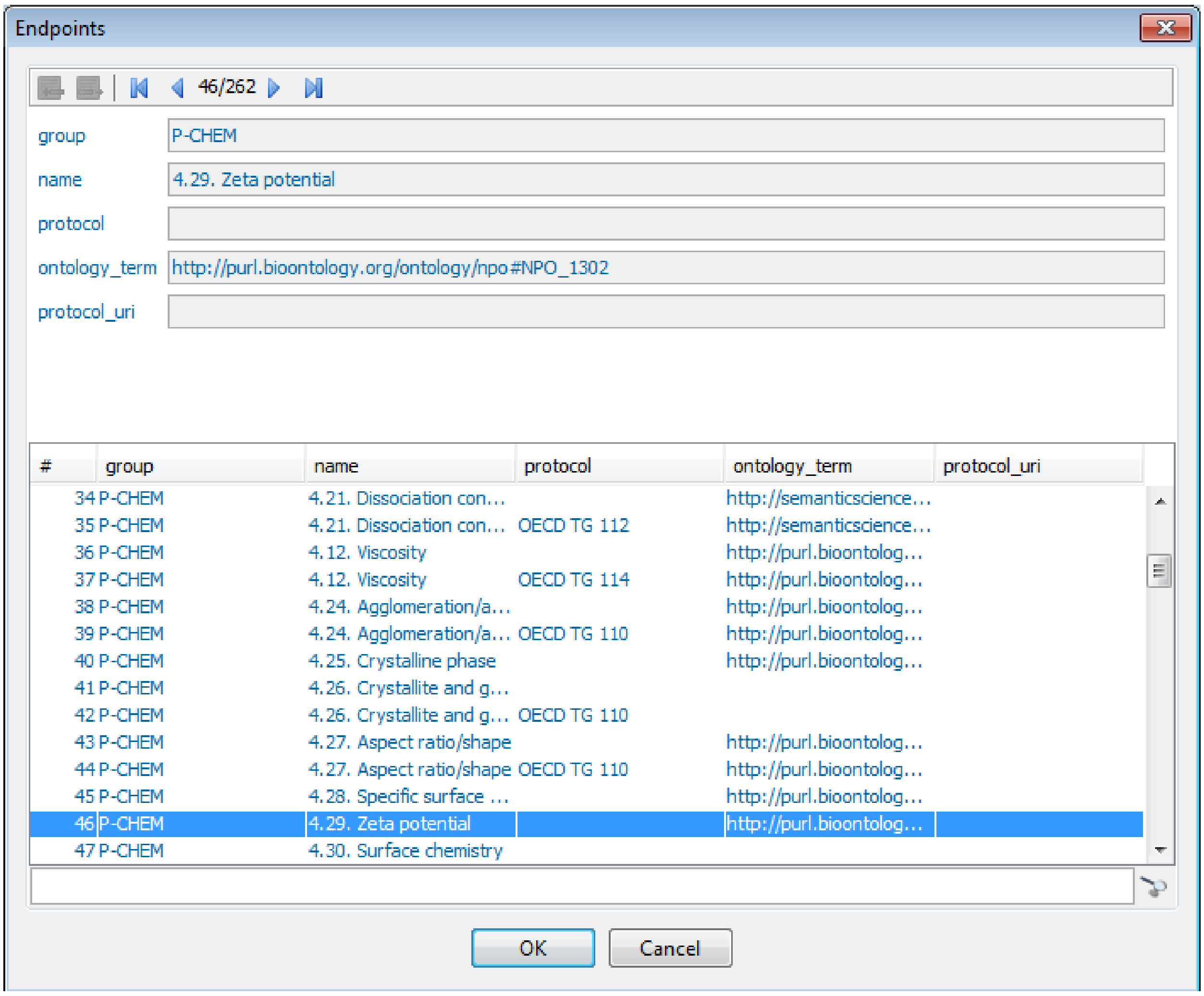
3. carbon dioxide emission (ENVO_01000770)
http://purl.obolibrary.org/obo/ENVO_01000770
comment This is a working definition for #280. Which processes will ultimately be included in the axioms is still open.
definition textual definition A volume of carbon dioxide gas emitted by a carbon dioxide emission process.

4. carbon dioxide increased AE (OAE_0001625)
http://purl.obolibrary.org/obo/OAE_0001625
definition editor term editor EB
definition textual definition a carbon dioxide abnormal AE that is characterized by an increased carbon dioxide level in the blood
see also HPO: HP_0005972

5. manganese (IV) dioxide nanoparticle (ENM_9000025)
http://purl.enanomapper.org/onto/ENM_9000025
isDefinedBy A metal oxide nanoparticle that consists of manganese

Figure 34: QMRF Editor 3.0.0 Internal validation

The endpoint field is extended to explicitly list protocols and ontology terms. The text fields are editable and can be modified.



Endpoints

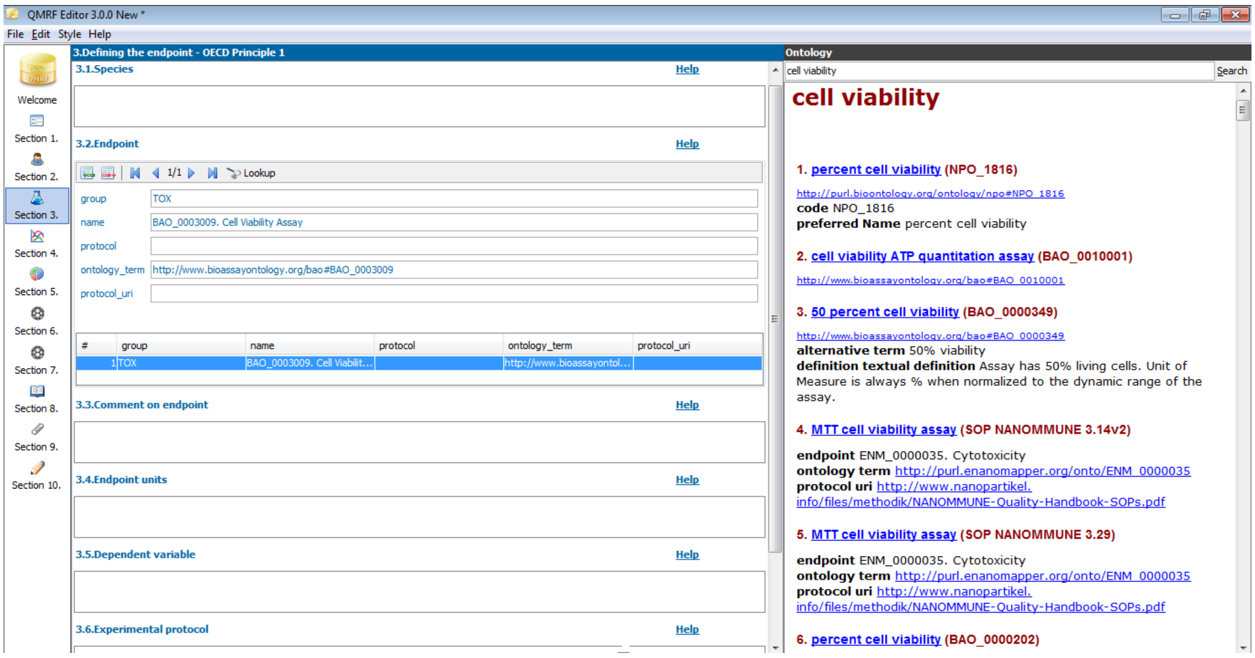
46/262

group	P-CHEM
name	4.29. Zeta potential
protocol	
ontology_term	http://purl.bioontology.org/ontology/npo#NPO_1302
protocol_uri	

#	group	name	protocol	ontology_term	protocol_uri
34	P-CHEM	4.21. Dissociation con…		http://semanticscience…	
35	P-CHEM	4.21. Dissociation con…	OECD TG 112	http://semanticscience…	
36	P-CHEM	4.12. Viscosity		http://purl.bioontolog…	
37	P-CHEM	4.12. Viscosity	OECD TG 114	http://purl.bioontolog…	
38	P-CHEM	4.24. Agglomeration/a…		http://purl.bioontolog…	
39	P-CHEM	4.24. Agglomeration/a…	OECD TG 110	http://purl.bioontolog…	
40	P-CHEM	4.25. Crystalline phase		http://purl.bioontolog…	
41	P-CHEM	4.26. Crystallite and g…			
42	P-CHEM	4.26. Crystallite and g…	OECD TG 110		
43	P-CHEM	4.27. Aspect ratio/shape		http://purl.bioontolog…	
44	P-CHEM	4.27. Aspect ratio/shape	OECD TG 110	http://purl.bioontolog…	
45	P-CHEM	4.28. Specific surface …		http://purl.bioontolog…	
46	P-CHEM	4.29. Zeta potential		http://purl.bioontolog…	
47	P-CHEM	4.30. Surface chemistry			

Figure 35: QMRF Editor 3.0.0 Endpoints

The new search tool facilitates finding and selecting proper protocol and ontology terms.



3.Defining the endpoint - OECD Principle 1

3.1.Species

3.2.Endpoint

group: TOX
name: BAO_0003009. Cell Viability Assay
protocol:
ontology_term: http://www.bioassayontology.org/bao#BAO_0003009
protocol_uri:

#	group	name	protocol	ontology_term	protocol_uri
1	TOX	BAO_0003009. Cell Viabilit...		http://www.bioassayontol...	

3.3.Comment on endpoint

3.4.Endpoint units

3.5.Dependent variable

3.6.Experimental protocol

Ontology

cell viability

cell viability

1. percent cell viability (NPO_1816)
http://purl.bioontology.org/ontology/npo#NPO_1816
code NPO_1816
preferred Name percent cell viability

2. cell viability ATP quantitation assay (BAO_0010001)
http://www.bioassayontology.org/bao#BAO_0010001

3. 50 percent cell viability (BAO_0000349)
http://www.bioassayontology.org/bao#BAO_0000349
alternative term 50% viability
definition textual definition Assay has 50% living cells. Unit of Measure is always % when normalized to the dynamic range of the assay.

4. MTT cell viability assay (SOP NANOMMUNE 3.14v2)
endpoint ENM_0000035. Cytotoxicity
ontology term http://purl.enanomapper.org/onto/ENM_0000035
protocol uri http://www.nanopartikel.info/files/methodik/NANOMMUNE-Quality-Handbook-SOPs.pdf

5. MTT cell viability assay (SOP NANOMMUNE 3.29)
endpoint ENM_0000035. Cytotoxicity
ontology term http://purl.enanomapper.org/onto/ENM_0000035
protocol uri http://www.nanopartikel.info/files/methodik/NANOMMUNE-Quality-Handbook-SOPs.pdf

6. percent cell viability (BAO_0000202)

Figure 36: QMRF Editor 3.0.0 Defining the endpoint (2)

The QMRF Editor 3.0.0 and qmrf.dtd (v.3.0) are released and available for download.

https://sourceforge.net/projects/qmrf/files/QMRF%20Editor/3.0.0/

7. CONCLUSION

In this report, we have outlined the new functionalities added to JQ, nano-lazar, and the QMRF Editor 3.0.0, namely validation schemes, read across training/prediction services and QPRF/QMRF reports. Therefore, users may now use similarity for making predictions, validate their models and receive automated reports on both their predictions and models. Output reports can be viewed and edited online, and later saved as PDF files to the user's personal computer. This information, in addition to the services reported in previous deliverables of WP4 (creating datasets, training models, making predictions, mechanistic modelling, experimental design and interlaboratory testing), can provide an extensive computational analysis framework for ENM safety assessment.

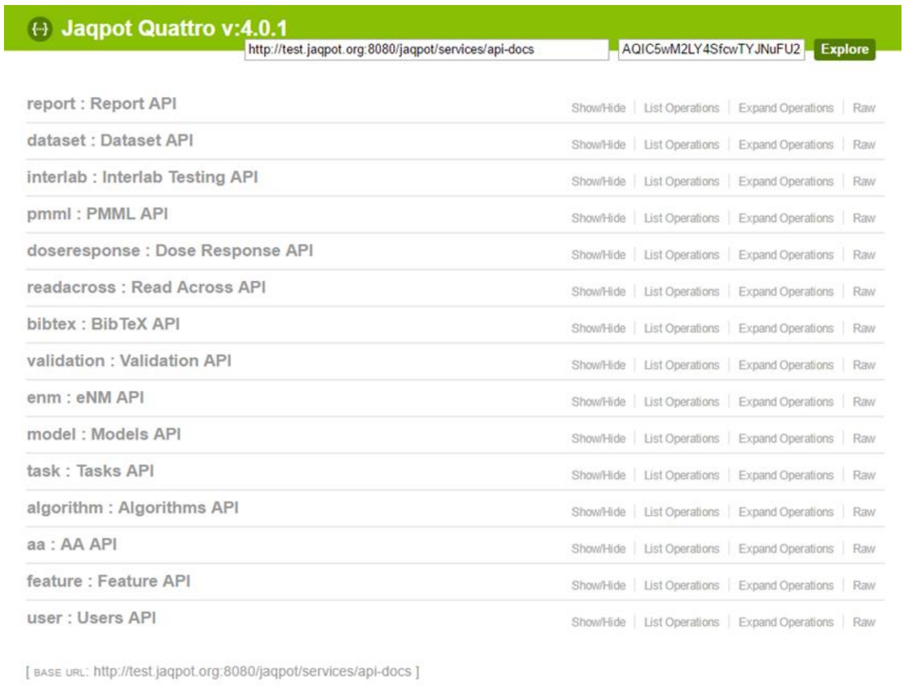
BIBLIOGRAPHY

1. *OECD 2007, OECD Guidance Document on the Validation of (Quantitative) Structure-Activity Relationship [(Q)SAR] Models. Retrieved from http://www.oecd.org/env/guidance-document-on-the-validation-of-quantitative-structure-activity-relationship-q-sar-models-9789264085442-en.htm (Accessed: 31/08/16)*
2. *QSAR Tools - QPRF 1.1, European Union Reference Laboratory for alternatives to animal testing (EURL ECVAM), Retrieved from https://eurl-ecvam.jrc.ec.europa.eu/laboratories-research/predictive_toxicology/qsar_tools/qrf/QPRF_version_1.1.doc/view (Accessed: 31/08/16)*
3. *Walkey, C. D., et al. (2014). Protein corona fingerprinting predicts the cellular interaction of gold and silver nanoparticles, ACS Nano 8(3): 2439-2455.*
4. *Gajewicz et al. (2015). Towards Understanding Mechanisms Governing Cytotoxicity of Metal Oxides Nanoparticles: Hints from Nano-QSAR Studies, Nanotoxicology 9 (3): 313-325*

APPENDIX

The Jaqpot Quattro (JQ) API documentation can be found at http://test.jaqpot.org:8080/jaqpot/swagger/. JQ is an extension of the Jaqpot web application, which was originally developed during the OpenTox project and features improved efficiency and additional functionality. JQ is an open-source project, written in Java and licensed with the GNU GPL v3 license. It provides asynchronous execution of tasks submitted by users, authentication, authorisation and accounting mechanisms powered by OpenAM. JQ is part of the eNM framework and communicates with other web services in the framework via the common REST API described. The source code is publicly available from https://github.com/KinkyDesign/JaqpotQuattro. A screenshot of the API can be seen in Figure 37. All functionalities made available for this deliverable can also be accessed using the user interface (UI). A screenshot of the UI can be seen in Figure 38, found at http://test.jaqpot.org:8000/.

Within the last months nano-lazar functionality was completely integrated into the lazar framework which was created during the OpenTox project. lazar is a GNU GPL v3 licensed open-source project written in Ruby. It is a very flexible library for the generation, validation and inspection of read-across models, that can use measured and calculated descriptors from OpenBabel, CDK and JOELib libraries, custom similarity measures for neighbor identification and any R algorithm for local QSAR models. lazar communicates with other eNanoMapper services via the common REST API. Source code is publicly available from https://github.com/opentox/lazar and the whole system can be installed with a single command as a Ruby gem from https://rubygems.org/gems/lazar. Documentation for the lazar API can be found at http://www.rubydoc.info/gems/lazar. Although lazar is primarily a command-line oriented library it serves as the basis of graphical user interfaces, e.g. the nano-lazar web-application at https://nano-lazar.in-silico.ch (Figure 24).


Jaqpot Quattro v:4.0.1

http://test.jaqpot.org:8080/jaqpot/services/api-docs | AQIC5wM2LY4SfcwTYJNuFU2 | Explore

report : Report API Show/Hide | List Operations | Expand Operations | Raw
dataset : Dataset API Show/Hide | List Operations | Expand Operations | Raw
interlab : Interlab Testing API Show/Hide | List Operations | Expand Operations | Raw
pmml : PMML API Show/Hide | List Operations | Expand Operations | Raw
doseresponse : Dose Response API Show/Hide | List Operations | Expand Operations | Raw
readacross : Read Across API Show/Hide | List Operations | Expand Operations | Raw
bibtex : BibTeX API Show/Hide | List Operations | Expand Operations | Raw
validation : Validation API Show/Hide | List Operations | Expand Operations | Raw
enm : eNM API Show/Hide | List Operations | Expand Operations | Raw
model : Models API Show/Hide | List Operations | Expand Operations | Raw
task : Tasks API Show/Hide | List Operations | Expand Operations | Raw
algorithm : Algorithms API Show/Hide | List Operations | Expand Operations | Raw
aa : AA API Show/Hide | List Operations | Expand Operations | Raw
feature : Feature API Show/Hide | List Operations | Expand Operations | Raw
user : Users API Show/Hide | List Operations | Expand Operations | Raw

[BASE URL: http://test.jaqpot.org:8080/jaqpot/services/api-docs]

Figure 37: Screenshot of Jaqpot API found at http://test.jaqpot.org:8080/jaqpot/swagger/

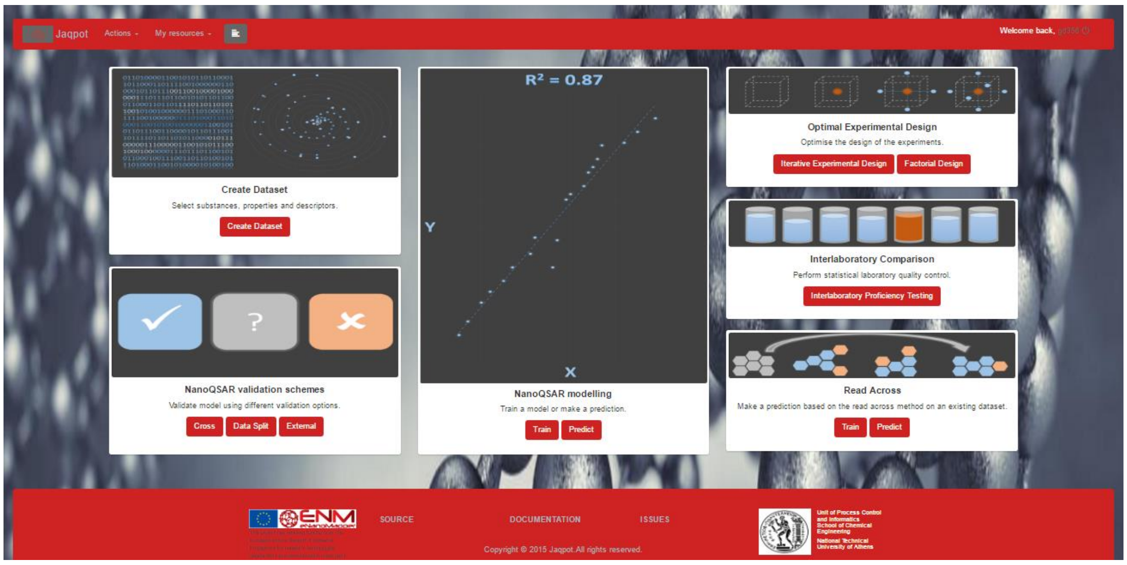

Jaqpot
Actions
My resources
Welcome back,
Create Dataset
Select substances, properties and descriptors.
Create Dataset
$R^2 = 0.87$
Y
X
NanoQSAR modelling
Train a model or make a prediction.
Train
Predict
Optimal Experimental Design
Optimise the design of the experiments.
Iterative Experimental Design
Factorial Design
Interlaboratory Comparison
Perform statistical laboratory quality control.
Interlaboratory Proficiency Testing
NanoQSAR validation schemes
Validate model using different validation options.
Cross
Data Split
External
Read Across
Make a prediction based on the read across method on an existing dataset.
Train
Predict
SOURCE
DOCUMENTATION
ISSUES
Copyright © 2015 Jaqpot.All rights reserved.
Unit of Process Control and Informatics School of Chemical Engineering National Technical University of Athens

Figure 38: *Screenshot of Jaqpot UI found at http://test.jaqpot.org:8000/*