



The European Nanotechnology Community Informatics Platform: Bridging data and disciplinary gaps for industry and regulators

Grant Agreement No 731032

## Deliverable Report 5.4

<b>Deliverable</b>	D5.4 First predictive nanoQSAR models integrated into NanoCommons KnowledgeBase
<b>Work Package</b>	WP5: JRA3 - Analysis and Modelling tools
<b>Delivery date</b>	M18 - 23 August 2019
<b>Lead Beneficiary</b>	NTUA
<b>Nature of Deliverable</b>	Demonstrator
<b>Dissemination Level</b>	Public (PU)

**Submitted by** Harry Sarimveis, Philip Doganis, Pantelis Karatzas (NTUA),  
Dimitra-Danai Varsou, Antreas Afantitis (NovaM)

**Revised by** Martin Himly (PLUS) and Georgia Melagraki (NovaM)

**Approved by** Iseult Lynch (UoB)



---

## Table of contents

Abbreviations	4
Summary	5
Introduction	6
Description	7
Development of nanoQSAR model through the Jaqpot modelling platform	7
Jaqpot 4 (Jaqpot Quattro) GUI	8
Authorisation and User space	8
nanoQSAR Model training	10
nanoQSAR Model Validation	14
Prediction of end-point values using the nanoQSAR Model	22
Jaqpot 5 GUI	24
Model training & deployment to Jaqpot 5	25
Model page overview	28
The overview tab	28
The data tab	31
The predict/validate tab	31
Social networking and Sharing of resources in Jaqpot 5	34
Organisations	34
Getting access to the NanoCommons organisation	36
Discussions	37
Private and Shared space for models and datasets	37
Uploading a new dataset	40
Sharing a model or dataset	44
NanoCommons repository of nanoQSAR models	45
Integration of NanoCommons nanoQSAR modelling infrastructure and the knowledge base	48
Enalos Cloud Platform	50
Enalos QNAR Iron Oxide Toxicity Platform	50
Required input	50
Outputs-results	52
Enalos Cloud NanoInformatics Platform: A Safe-by-Design Tool for Functionalised NMs	55
Required input	55

---

File entry	58
Outputs-results	58
Conclusions	62
References	63
Annex	67
Appendix 1. Available algorithms in Jaqpot 4 for training nanoQSAR models	67
Appendix 2. An example of a full editable QPRF report generated by Jaqpot 4	70
Appendix 3. Guidelines on installing the <i>jaqpotpy</i> library	75
Appendix 4. Example QMRF report	83
Appendix 5. Example of a PMML representation of a nanoQSAR model	89
Appendix 6. Detailed information on examples of models, as implemented in Jaqpot 4/Jaqpot 5	91
Appendix 7. Python notebook for Jaqpot-Biomax communication with summarised output	101

---

## Abbreviations

AaaS: Algorithm as a Service  
API: Application Programming Interface  
CA: Carbonic Anhydrase  
CAS and EC numbers  
CSV: Comma Separated Values  
DoA: Domain of Applicability  
DOI: Digital Object Identifier  
ECHA: European Chemicals Agency  
ENMs: Engineered nanomaterials  
FP7: Seventh Framework (FP7 ) Programme of the European Union  
GUI: Graphical User Interface  
IATA: Integrated Approaches for Testing and Assessment  
InChi: International Chemical Identifier  
IUPAC: International Union of Pure and Applied Chemistry  
JDPI: Jaqpot Protocol for Data Interchange  
JRC: Joint Research Centre  
JSON: JavaScript Object Notation  
KB: Knowledge Base  
MaaS: Model as a Service  
MDL: MDL file format  
MLR: Multiple Linear Regression  
MOL: MOL file format  
MWCNTs: Multi-walled carbon nanotubes  
NM: Nanomaterial  
NP: Nanoparticle  
OECD: Organisation for Economic Co-operation and Development  
PDF: Portable Document Format  
PMML: Predictive Model Markup Language  
PVA: Poly Vinyl Alcohol  
QMRF: QSAR Model Reporting Format  
QNAR: Quantitative Nanostructure-Activity Relationship  
QPRF: QSAR Prediction Reporting Format  
QQ plot: quantile-quantile plot  
QSAR: Quantitative Structure Activity Relationships  
REST: representational state transfer  
SMILES: Simplified Molecular-Input Line-Entry System  
TA: Transnational Access  
UI: User Interface  
URI: Uniform Resource Identifier  
ZP: Zeta Potential

---

## Summary

This Deliverable report (D5.4) presents the services that have been developed and are available through the NanoCommons infrastructure for generating and validating nano-specific quantitative Structure-Activity Relationships (nanoQSAR) models and applying the models for predicting nanomaterial (NM) end-points for new materials that have not been tested experimentally. We offer two levels of modelling services: (i) Algorithm as a Service (AaaS) which provides the technical tools to model developers for creating nanoQSAR models and deploying them in the NanoCommons Knowledge Base, or (ii) Model as a Service (MaaS) which provides ready-to-use web implementations of nanoQSAR models, that can be used by the community to validate the models or calculate end-point predictions for other NMs as long as these are within the Domain of Applicability (DoA) of the model.

The nanoQSAR modelling tools have been developed in accordance with the Organisation for Economic Cooperation and Development (OECD) principles for validating QSAR models (OECD, 2007). In addition to the modelling services, we have developed and integrated a number of supporting tools: tools for determining the Domain of Applicability (DoA), tools for visualising the modelling and prediction results in summarised table or figure formats, as well as tools for producing and integrating QSAR Model Reporting Format (QMRF) and QSAR Prediction Reporting Format (QPRF) reports, which are a prerequisite to model acceptance by a regulatory entity such as the European Chemicals Agency (ECHA).

NanoCommons has integrated two nanoQSAR modelling platforms, namely the Jaqpot and Enalos platforms developed by partners NTUA and NM, respectively, into the NanoCommons Knowledge Base and Transnational Access (TA) portal. In this deliverable, the two platforms are fully demonstrated based on the presentation of complete nanoQSAR modelling workflows.

An important objective of NanoCommons Task 5.3 is to create and offer to the community a central repository of well-validated nanoQSAR models. In this deliverable we illustrate that the nanoQSAR model repository is already designed, implemented and populated with a number of nanoQSAR models. The models are available as ready- and easy-to-use web applications and are accompanied by meta-information which allow the users to search the available model catalogue and find the most suitable model for their specific needs.

In the deliverable report we also illustrate that nanoQSAR modelling through the NanoCommons infrastructure can guide and trigger TA activities during the course of the project.

---

## Introduction

Computational models that predict adverse biological effects of engineered NMs (ENMs) are becoming increasingly important to support risk assessment. This is primarily due to the cost-saving and reduction of attrition rates they help achieve, since market candidates under development can be assessed for toxicity early-on in the process. Therefore, candidates predicted to be toxic can be discarded before a significant amount of time and effort has been invested, and most significantly, before expensive experimental tests need to be carried out. *Quantitative Structure-Activity Relationship* (QSAR) models are mathematical models derived from the algorithmic analysis of available experimental activity training data, which are able to predict the unknown activities of other compounds, directly from structural characteristics (called “descriptors”). In the nano-domain it is common to refer to QSARs that are applied to NMs as nanoQSARs.

In this deliverable we describe and demonstrate the tools and services that have been developed during the first 18 months of the NanoCommons project for generating and validating nanoQSAR models, and the use of the models for computing end-point predictions. The first section of the deliverable describes the methodologies that are used for generating predictive models and the tools that are used by the nanosafety and the data modelling communities to validate the models, define their DoA and report model structures and model predictions. The two subsequent sections present in detail and demonstrate the two platforms that have been integrated into the NanoCommons infrastructure for nanoQSAR modelling, namely Jaqpot and Enalos. The fourth section demonstrates the integration of the modelling services with the NanoCommons infrastructure through an example where data contained in the NanoCommons Knowledge Base are used to create a nanoQSAR model and its implementation as a web service. The fifth section describes the repository of nanoQSAR models which is under development based on the Jaqpot modelling platform. The deliverable ends with the conclusion section, where we describe our vision for the future, regarding mainly the use of NanoCommons as a central repository of nanoQSAR models, where the community members will be allowed to freely deploy, share and discuss new nanoQSAR models and developments.

## Description

nanoQSAR models are mathematical models developed to predict properties or biological activities of NMs as functions of structural characteristics (called descriptors). Overall, nanoQSARs can be grouped into regression and classification models. Statistical and machine learning modelling techniques, such as *multiple linear regression* (MLR) (Gajewicz et al., 2015, Puzyn et al., 2011, Pan et al., 2016), *logistic regression* (Pandharipande et al., 2009), *Naïve Bayes* (Liu et al., 2013), *decision tree analysis* (Hansen et al., 2013), *random forest* (Oh et al., 2016), *k-nearest neighbour* (Cassotti et al., 2014 and Fourches et al., 2010), *partial least square regression* (Walkey et al., 2014, Brandmaier et al., 2012 and Wold et al., 2004), *Neural Networks* (Zarei et al., 2010), *support vector machines* (Fourches et al., 2010 and Liu et al., 2013), *ensemble learning* (Sellers et al., 2015 and Singh & Gupta, 2014) and *genetic algorithms* (Gajewicz et al., 2015) have been found useful for the establishment of the relationships between the molecular structures and biological activities of ENMs. Computational models are evaluated based on their predictive accuracy (*i.e.*  $R^2$  for regression models and balanced class accuracy for classification models) derived from several common validation methods.

For the development and the validation of QSAR models the scientific community has adopted the “OECD Principles for the Validation, for Regulatory Purposes, of (Q)SAR Models” (OECD, 2007). There are five principles that need to be taken into account: i) Defined endpoint, ii) An unambiguous algorithm, iii) A defined domain of applicability, iv) Appropriate measures of goodness-of-fit, robustness and predictivity for NM models, and v) Mechanistic interpretation, if possible.

In a recent publication (Puzyn et al., 2018), written by scientists involved in five recently completed European nanosafety modelling research projects, the application of the OECD validation principles in nanoQSAR modelling was critically reviewed. A main conclusion was that the OECD principles create an appropriate framework for validating nanoQSAR models, but there are issues that need particular attention. Among them, particular attention is drawn to the need for transparency and reproducibility of nanoQSAR models, documentation through the standard QMRF (QSAR Model Reporting Format) format and model representations using the Predictive Model Markup Language (PMML).

In the next sections of the deliverable report we demonstrate the Jaqpot and Enalos platforms that are part of the NanoCommons infrastructure and provide tools for creating, storing, validating and sharing nanoQSAR models. We illustrate how the OECD principles and the specific requirements for modelling NMs are taken into account in the NanoCommons modelling platforms.

## Development of nanoQSAR model through the Jaqpot modelling platform

Jaqpot is a computational platform for *in-silico* modelling of chemical compounds, which was originally developed by NTUA during the FP7 OpenTox project according to the OpenTox Application Programming Interfaces (APIs) and is continuously being extended, so that it is updated with new developments and advances. Jaqpot follows the microservice architecture where individual services

are interconnected and linked together and with external services through REST API calls. The Jaqpot APIs have been presented in Deliverable report D4.2, where it was mentioned that APIs constitute a means to develop user-friendly applications such as Graphical User Interfaces (GUIs), and Jupyter notebooks that consume Jaqpot services over the API, without the user having to deal with the technical burden of this implementation. In the current deliverable we mainly focus on the end-user requirements and needs and present the two GUIs which are currently available: *Jaqpot 4 (also mentioned as Jaqpot Quattro)* was developed during the eNanoMapper project and was the main outcome of the project on the analysis and modelling infrastructure. *Jaqpot 5* is being developed during the NanoCommons project. The two GUIs are currently complementary, but the goal is to eventually provide all modelling functionalities through the newest GUI. The two GUIs will be demonstrated in this deliverable by showing the complete workflow for developing, validating and sharing the linear nanoQSAR model which is described in Gharagheizi & Alamdari (2008), as an example of how the platform can be used to easily integrate models developed by any nanosafety community member into a single integrated platform, thus allowing users to select the optimal model for their needs, and facilitating benchmarking of models and integration of different approaches.

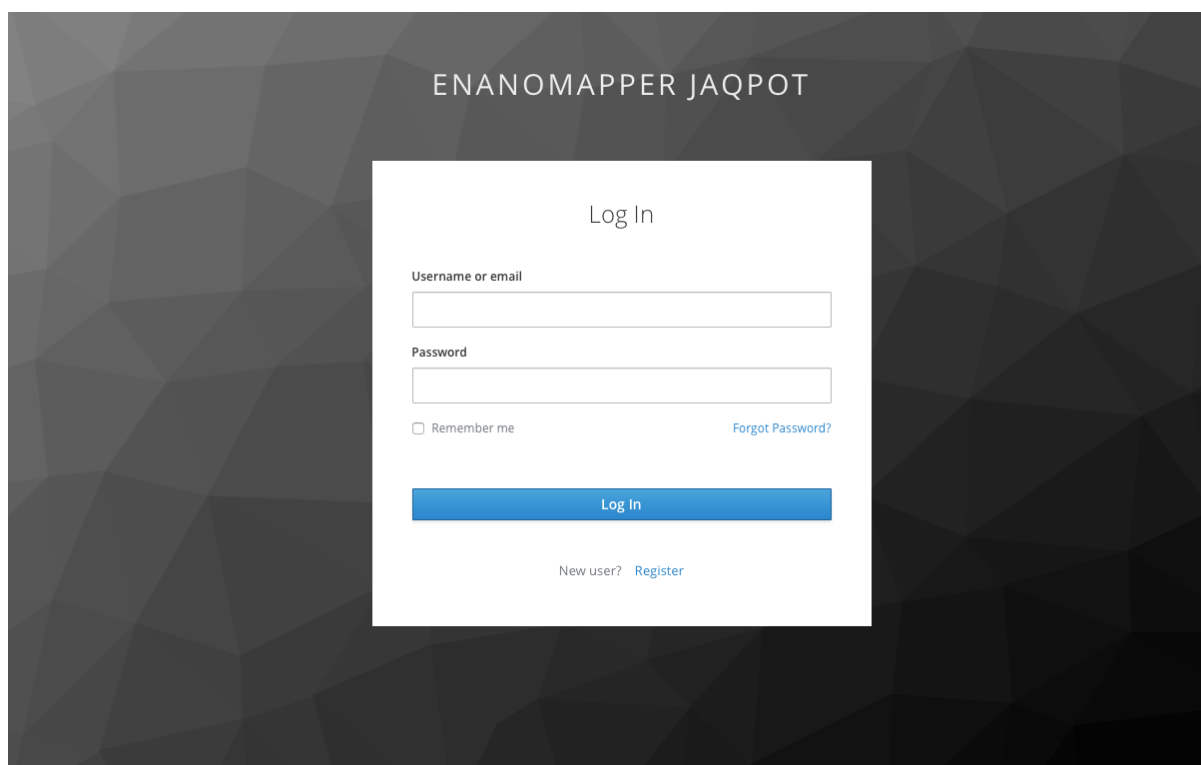
### Jaqpot 4 (Jaqpot Quattro) GUI

Jaqpot 4, the 4<sup>th</sup> edition of Jaqpot can be accessed at <http://www.jaqpot.org>. Communication to and from Jaqpot services is possible by exchanging JavaScript Object Notation (JSON) documents that contain no more information than what a modelling service needs. This flexibility allows easy integration with other services, as the only requirement is proper syntax of the JSON files.

#### Authorisation and User space

In order to access the platform, users need to sign in as a guest (username: *guest* password: *guest*), or create their own account by clicking the *Create account* button on the starting page, which takes them to the login page shown in Figure 1.





**Figure 1.** The login/registration screen (Jaqpot 4).

Creation of a new user is possible by clicking the *Register* button and adding their profile. Successful login leads to a screen where the username is displayed (Figure 2); please note that there is a time limit for which access is granted and users are advised to log off and on again after prolonged periods of time, in order to ensure smooth access to services. For the NanoCommons project we have created an account (username: *NanoCommons*, password: *NanoCommons*).

All Jaqpot 4 resources and components shown in this report have been created using the NanoCommons account. All links are accessible, by first entering Jaqpot 4 with the NanoCommons account and then pasting the link in the address line.

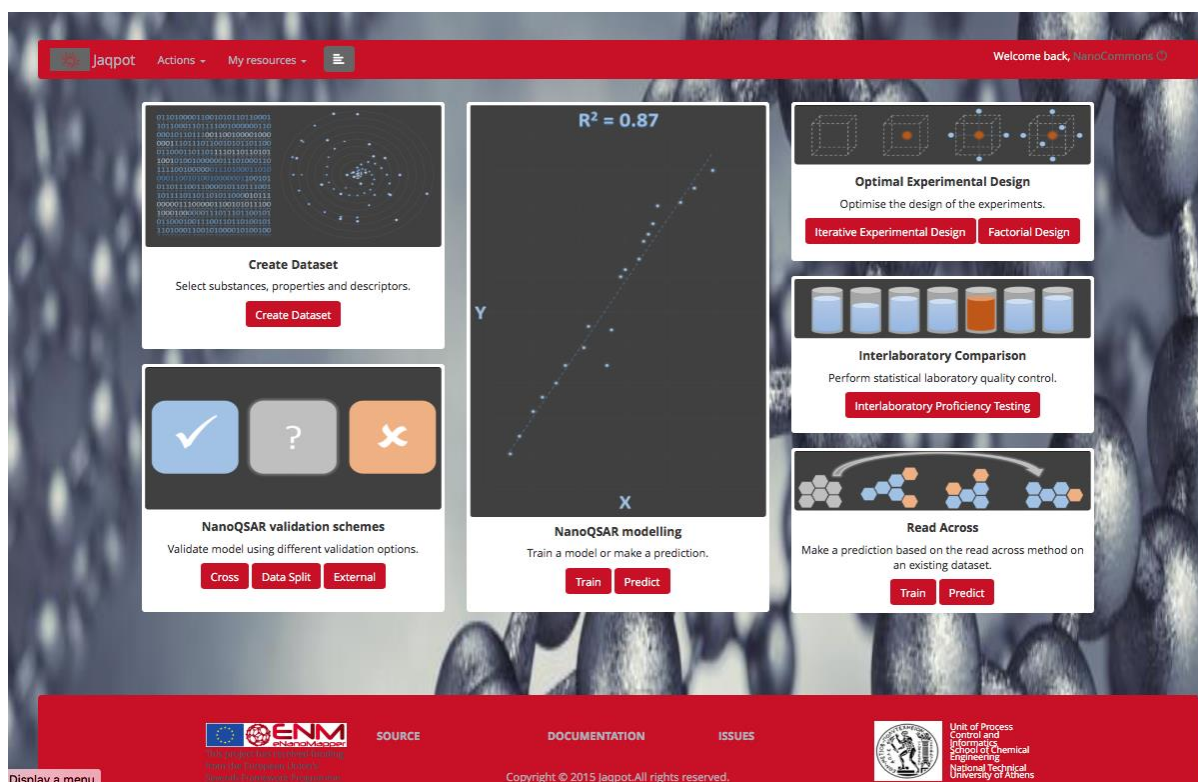


Figure 2. The Jaqpot 4 home screen after login.

The main screen of Jaqpot gives access to the available functionalities. We will focus on the components which are relevant to this deliverable, *i.e.* NanoQSAR modelling and NanoQSAR validation schemes. We assume that datasets are already prepared using the steps described in Appendix 1 of NanoCommons Deliverable report D5.2. For the purposes of the deliverable report we use a full dataset of 124 solvents that has been uploaded to Jaqpot and can be viewed in the URI [http://www.jaqpot.org/data\\_detail?name=nGF3G5SBo4wk5h](http://www.jaqpot.org/data_detail?name=nGF3G5SBo4wk5h). Additionally, the dataset has been split into a [training dataset](#) (of 93 solvents) and a [test dataset](#) (of 31 solvents). The training dataset will be used to construct the model and afterwards the model will be tested on the test dataset.

#### nanoQSAR Model training

The modelling procedure can commence in two ways, as shown in Figure 3:

1. by selecting “Actions” in the menu and then “Train a model”
2. by clicking the “Train” button in the “NanoQSAR modelling” section of the main Jaqpot screen.

In the next screen (Figure 4) we choose the [training dataset](#) consisting of 93 solvents as the dataset that will be used to train the model.

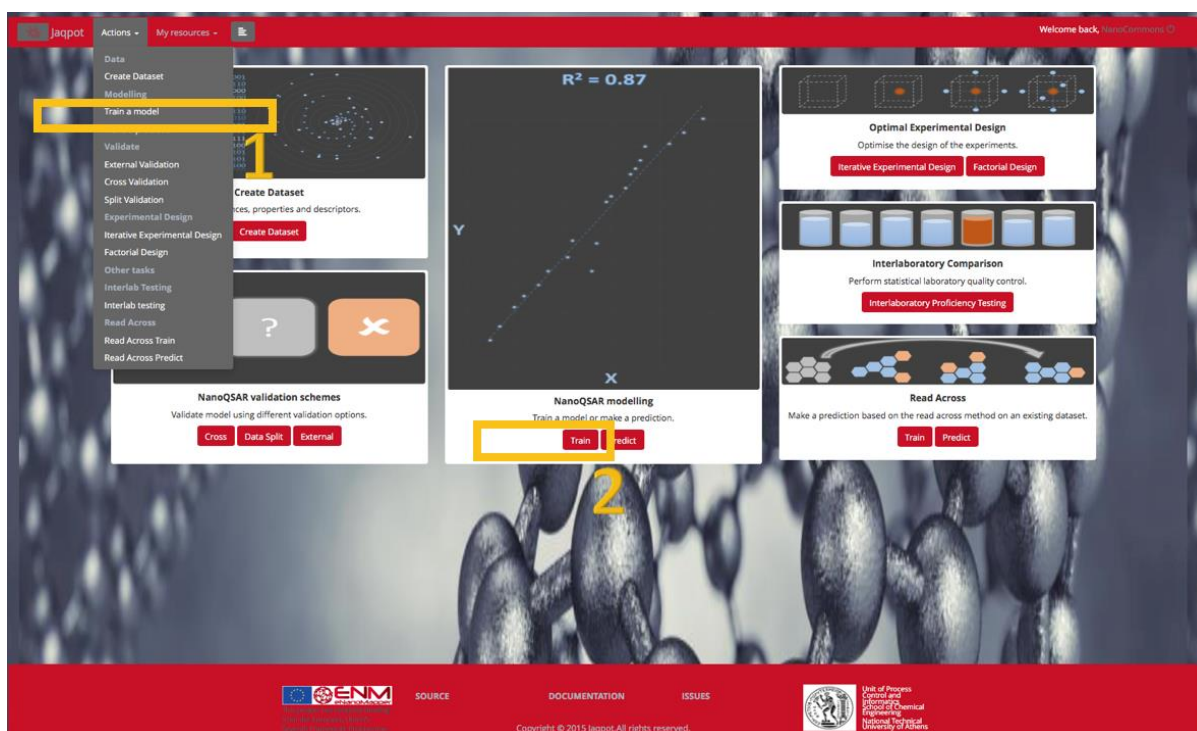


Figure 3. Highlighted menu items to initiate model training (Jaqpot 4).

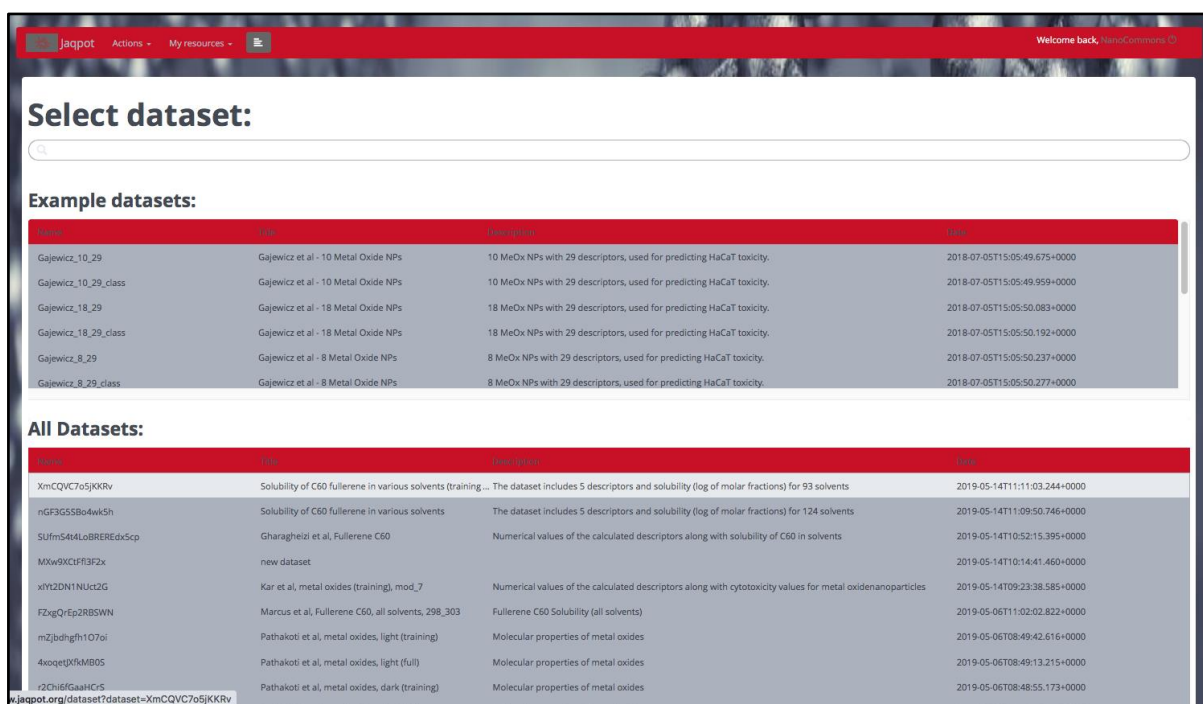


Figure 4. Dataset selection (Jaqpot 4).

In the next step, users choose from the algorithm library the Linear Regression from Python's Scikit-learn as the one that will be used for creating the model (Figure 5).

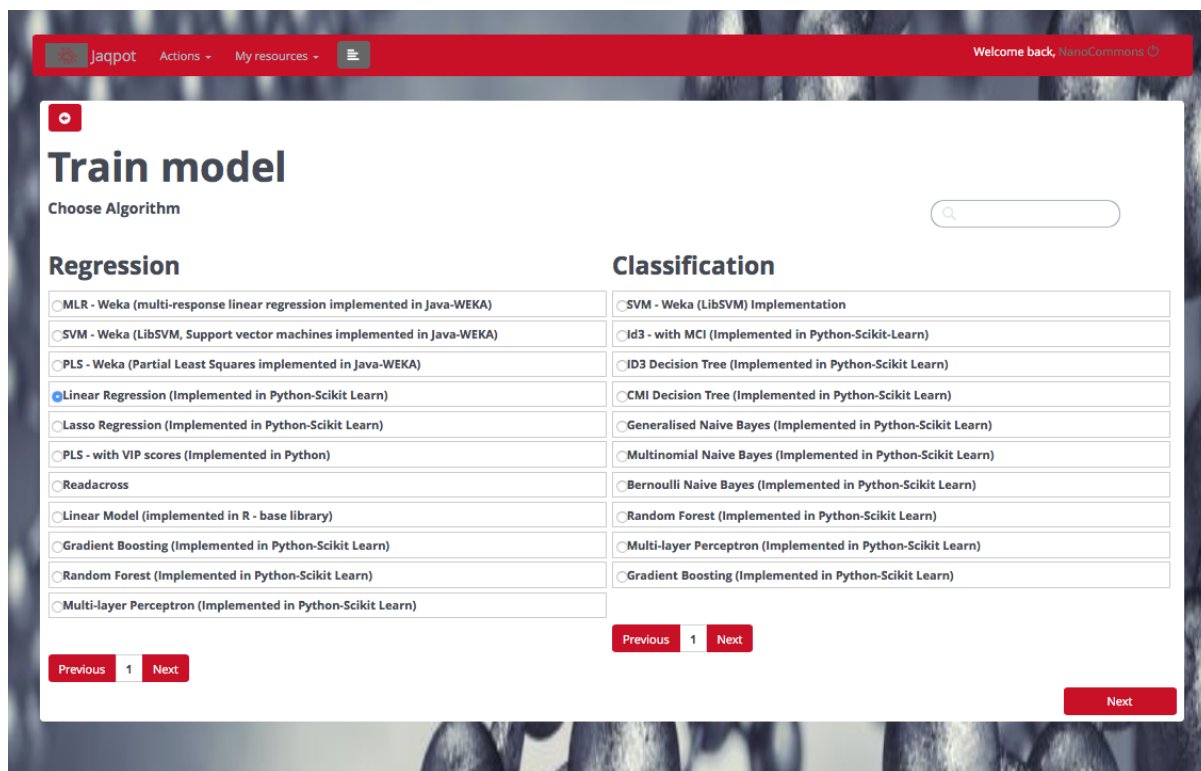


Figure 5. Algorithm selection (Jaqpot 4).

As can be observed, algorithms have been integrated from diverse sources and programming languages (WEKA (Hall et.al, 2009), Python (Pedregosa, 2011) and R (R Development Core Team, 2012)). This has been made possible through the implementation of the Jaqpot Protocol for Data Interchange (JPDI) (Chomenidis et al., 2015), first introduced in Jaqpot Quattro (Chomenidis et al., 2017) during the eNanoMapper project. Along with the basic information (Category / URI, Description), each algorithm has been defined ontologically using OpenTox Algorithm Ontological Classes as laid out in <http://old.opentox.org/dev/apis/api-1.1/Algorithm>. The full list of algorithms with ontological annotations are presented in Appendix 1.

Figure 6 illustrates the information that will accompany the model after its implementation regarding the algorithm used, describing the model, selecting the variables and defining transformations, selecting among scaling options and domain of applicability definitions:

- **Title of the algorithm:** Linear Regression (Implemented in Python-Scikit Learn). This is automatically filled in by Jaqpot according to the choice made in the previous screen (Figure 5)
- **Model name:** Linear nanoQSAR model predicting Solubility of C60
- **Model description:** The model is provided in the following publication: Farhad Gharagheizi & Reza Fareghi Alamdari (2008) A Molecular-Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents. Fullerenes, Nanotubes, and Carbon Nanostructures, 16:1, 40-57,

DOI: 10.1080/15363830701779315. Here we provide the context of the model, and a reference to research work developing the model, preferably with DOI.

- **Select variables:** the user selects the variables that will be used for the model. There are alternatives here, designed to cover varying user needs. We will choose the first alternative, *Select Input variable(s) and endpoint*, whereby the user quickly chooses input variables and the endpoint from drop-down lists. We choose to train the model with the variables *Seigp*, *H1m*, *More23e*, *ATS1m*, *piPC03* and *logS Exp* as the endpoint, while we omit the *Solvents* variable, since it has a non-numerical content (the names of the solvents).
- **Select scaling method:** In order to avoid having large deviations in the scale of the variables that will be used, it is often advisable to perform scaling on the variables as a preprocessing step. In Jaqpot, two choices are available: *Scaling between zero and one* or *Normalisation*. We choose *Scaling between zero and one*.
- **Select domain of applicability method:** This allows the user to make DoA (Roy & Kar, 2015, Roy et al., 2015, Netzeva et al., 2005) calculations on model predictions based on the Leverage method (Atkinson, 1985). The DoA values provide a measure of the model's ability to provide reliable predictions for each point predicted. We choose to make DoA calculations.

Figure 6. Algorithm parameters, model details and variable choice (Jaqpot 4).

Clicking on the “Train” button, starts the modelling task and the user is transferred to the following screen (Figure 7):

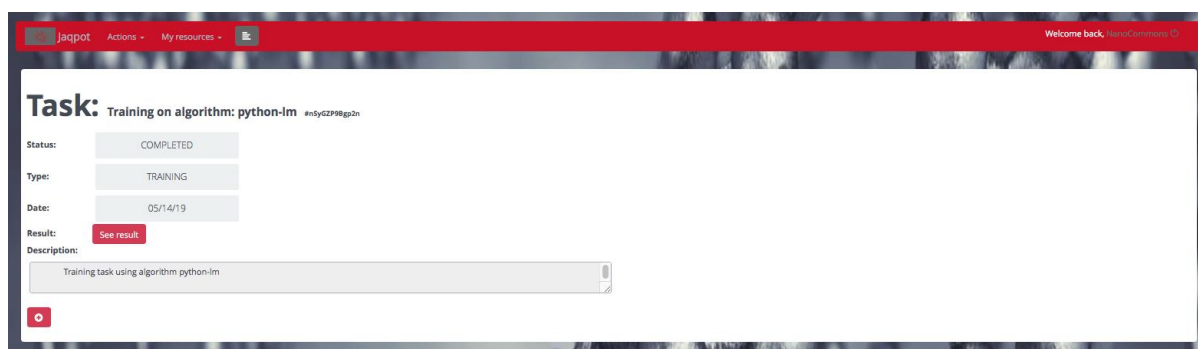


Figure 7. Intermediate task screen (Jaqpot 4).

Once the task has been completed, the nanoQSAR model has been generated, the “*See result*” button is activated and clicking it leads to the model web page (Figure 8). We have now created a web service for predictions of C60 fullerene solubility.

As with datasets, the model becomes available over a URI accessible by people that want to use it on the interface or by third parties that want to consume Jaqpot services over the API and offer them to through their systems: [http://www.jaqpot.org/m\\_detail?name=72GEEEmGhhavY00n7O209](http://www.jaqpot.org/m_detail?name=72GEEEmGhhavY00n7O209). The model page contains the information about the model as defined by its creator and also allows its use for predictions or validation.

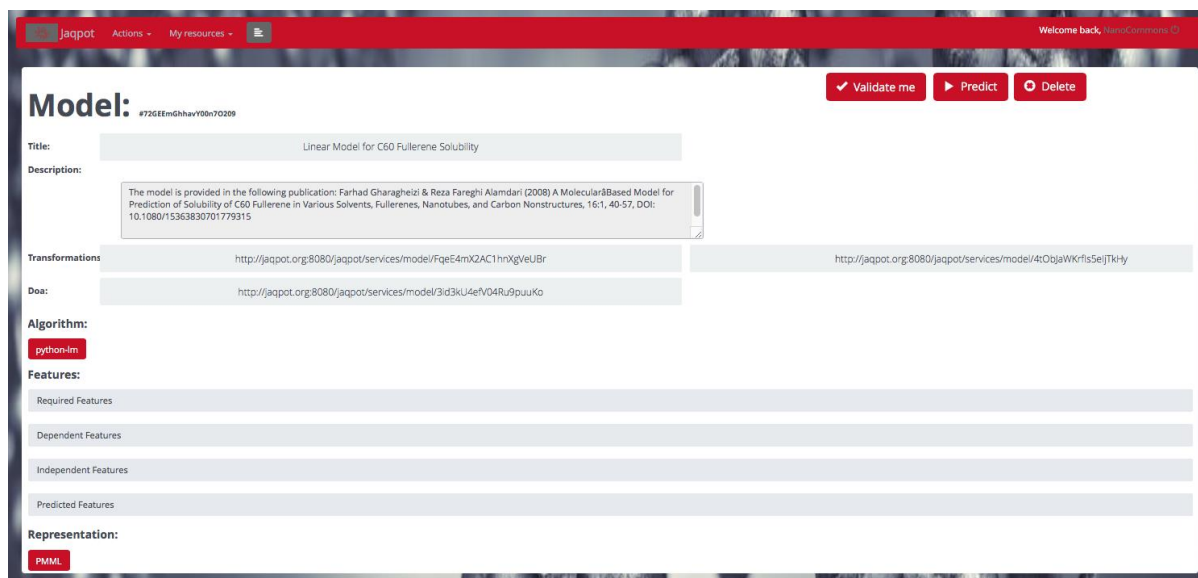


Figure 8. Model webpage (Jaqpot 4).

### nanoQSAR Model Validation

After the model has been created, we can validate it by clicking on “*Validate me*” button. We are prompted to “*Choose method*” for providing the data for the validation (Figure 9). This can be done either in the default “*Select dataset*” option, where a listed dataset is chosen, or in the “*Insert values*”

setting (Figure 10). In this case, an embedded spreadsheet appears where the user can type dataset values, or paste values from a spreadsheet in the respective columns. Here for demonstration purposes, we choose the second option and paste the test data, consisting of the descriptors and the end-point value of the 31 test solvents.

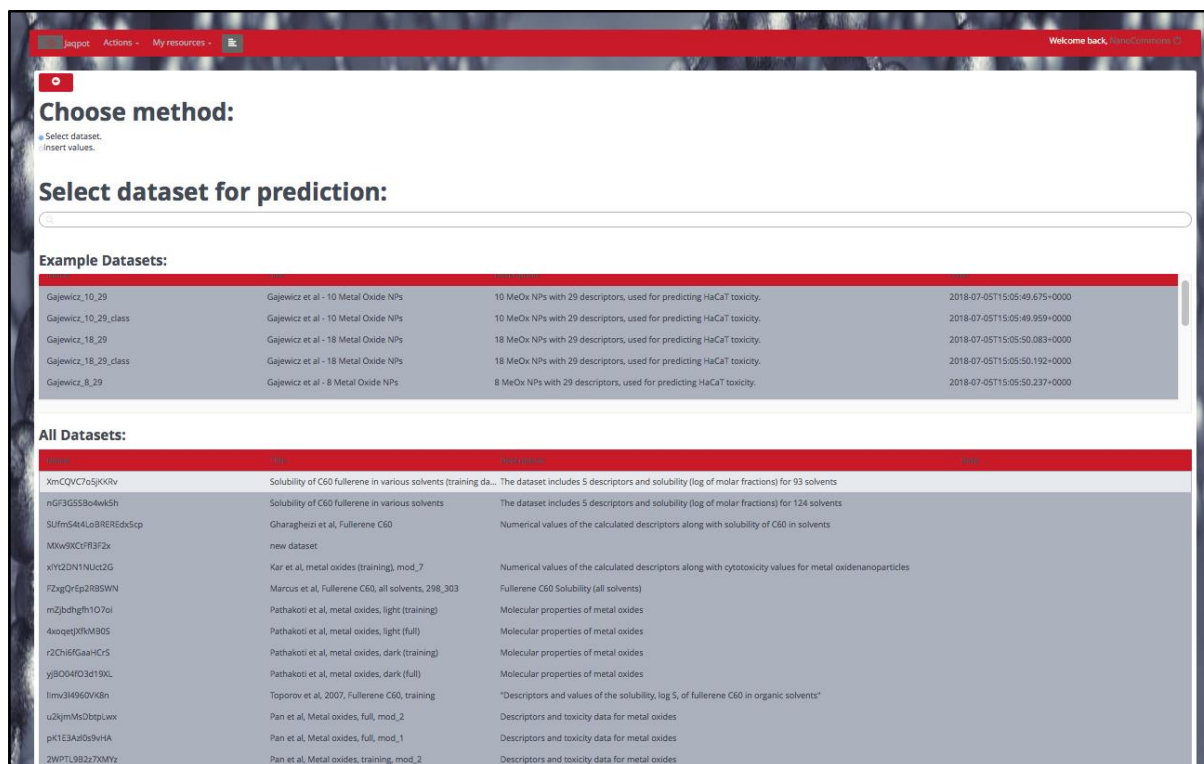


Figure 9. Dataset selection for Validation of the nanoQSAR model (Jaqpot 4).



Figure 10. Inserting values for Validation of the nanoQSAR model (Jaqpot 4).

After clicking the “Validate” button, the external validation screen appears (Figure 11), where the user is provided with an editable Validation report with relevant validation metrics, real versus predicted values in table form (Figure 12) and in a plot and a QQ plot (Figure 13).

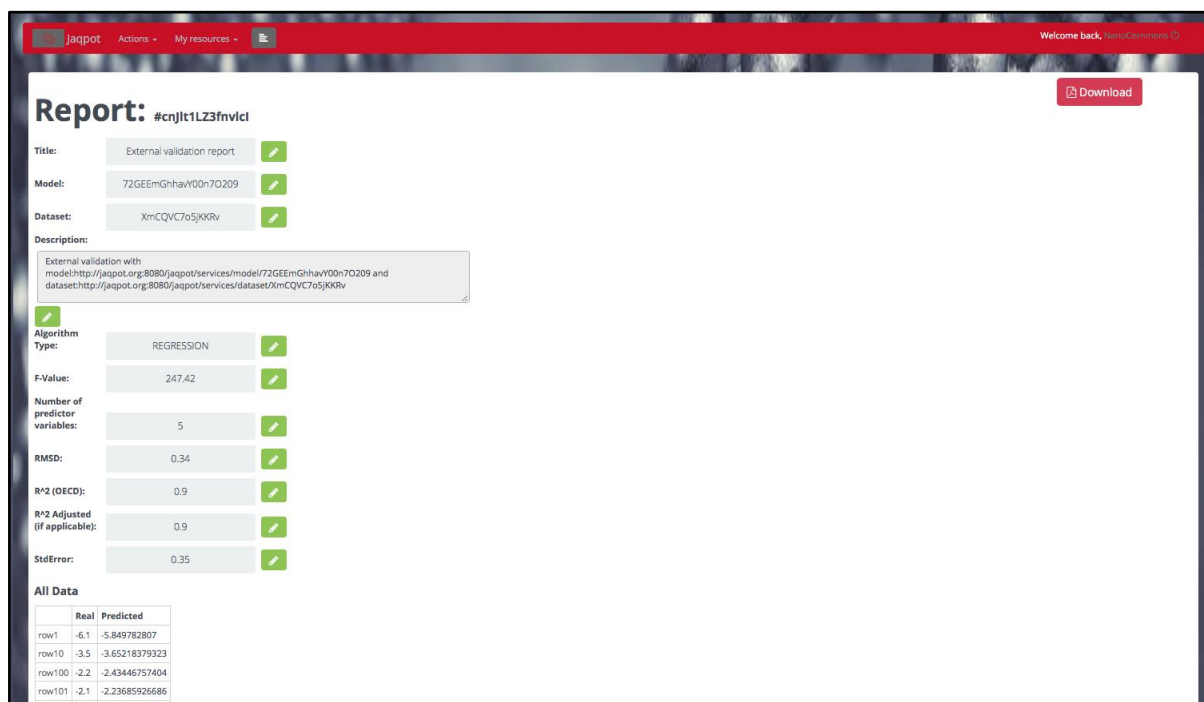
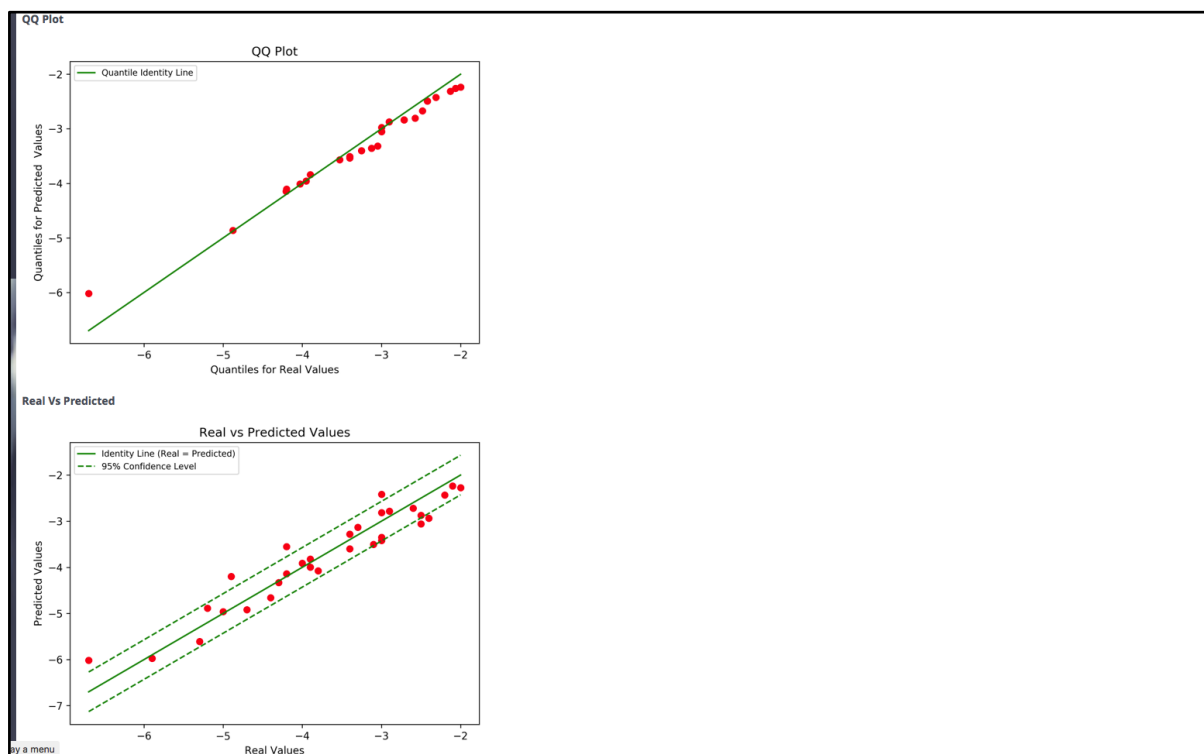


Figure 11. External Validation report {screen 1 of 3} (Jaqpot 4).



Figure 12. External Validation report {screen 2 of 3} (Jaqpot 4).





**Figure 13.** External Validation report {screen 3 of 3} (Jaqpot 4).

This model validation information can also be downloaded as a PDF-formatted file for easier communication of results to others (Figure 14).

Page 1 of 7

---

*This report has been automatically created by the JaqpotQuatro report service. Click here to navigate to our official webpage*

### External validation report

**Description:** [External validation with model:  
http://jaqpot.org:8080/jaqpot/services/model/72GEEemGhhavY00n7O209 and dataset:  
http://jaqpot.org:8080/jaqpot/services/dataset/XmCQVC7o5jKKRv]

**Model:** 72GEEemGhhavY00n7O209

**Dataset:** XmCQVC7o5jKKRv

**Algorithm Type:** REGRESSION

**F-Value:** 247.42

**Number of predictor variables:** 5

**RMSD:** 0.34

**R<sup>2</sup> (OECD):** 0.9

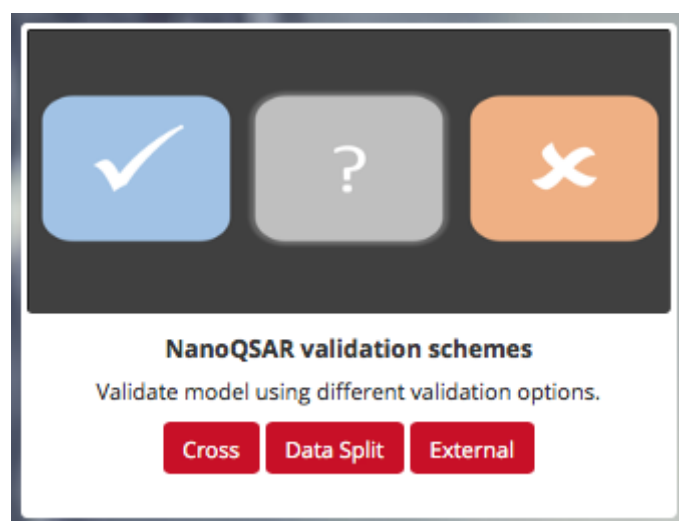
**R<sup>2</sup> Adjusted (if applicable):** 0.9

**StdError:** 0.35

**Figure 14.** External Validation report of the nanoQSAR model (Jaqpot 4).

Besides the external validation described so far, Jaqpot gives additional validation options in the section “*NanoQSAR validation schemes*” of the central Jaqpot page (Figure 15), namely:

- Cross: performs Cross Validation
- Data Split: splits the available data into training and test datasets according to a split ratio



**Figure 15.** Available Validation schemes (Jaqpot 4).

In the case of Cross Validation, the workflow is similar to that of creating a model. After selection of the dataset, the user is directed to the page shown in Figure 16 in order to define the algorithm data. Please note that in Cross Validation there are additional features: the definition of the number of folds that will be used in the cross-validation procedure and the option to stratify, which can be random, normal or not chosen at all.

**Figure 16.** Algorithm parameters, model details and variable choice during Cross Validation of nanoQSAR models (Jaqpot 4).

In contrast to the modelling workflow, after the algorithm screen, during the validation workflow users are transferred directly to the validation report page (Figure 17).

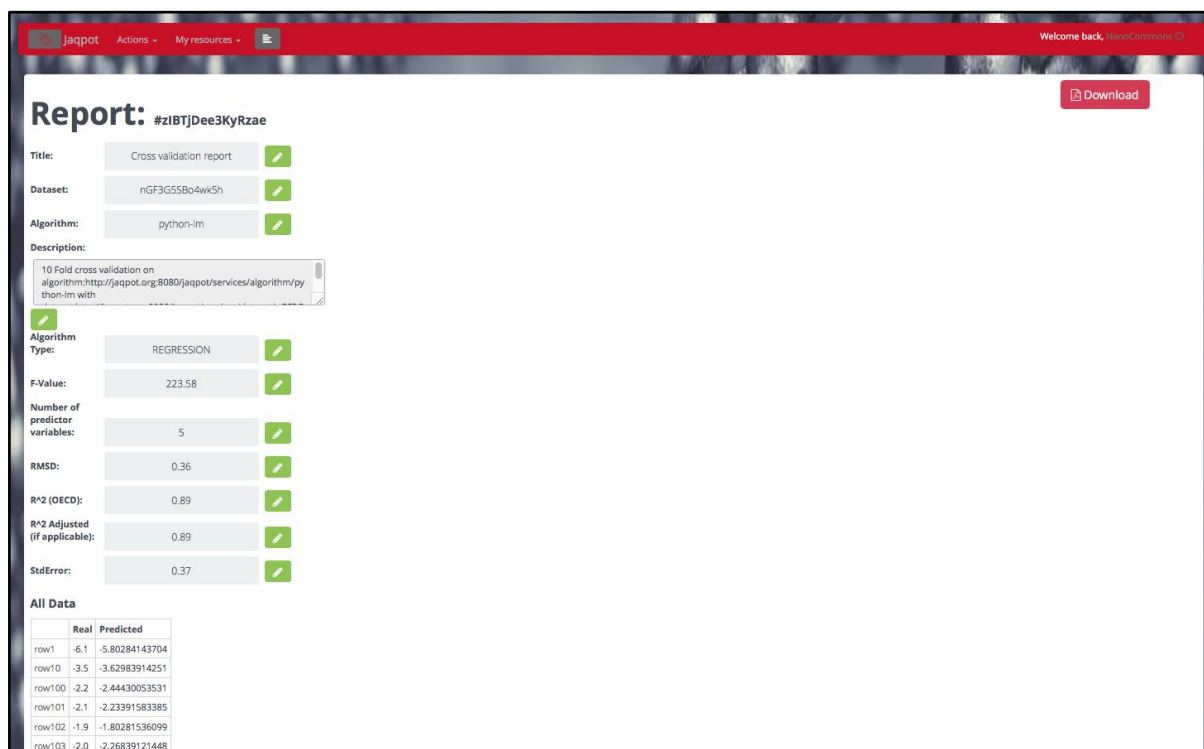


Figure 17. Cross Validation report for the nanoQSAR model (Jaqpote 4).

Similarly, if the Data Split option is chosen, the user is led to the screen in Figure 18 where users must define the split ratio, as well as stratification options.

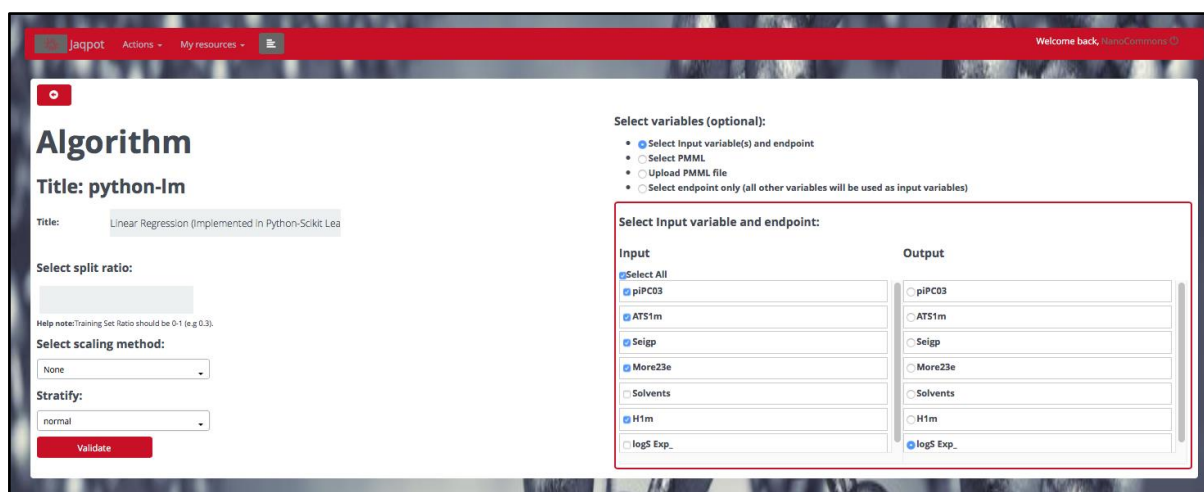


Figure 18. Algorithm parameters, model details and variable choice during Split Validation (Jaqpote 4).

After clicking on the “Validate” button, a validation report is automatically produced (Figure 19), which can be downloaded as a PDF file (Figure 20).

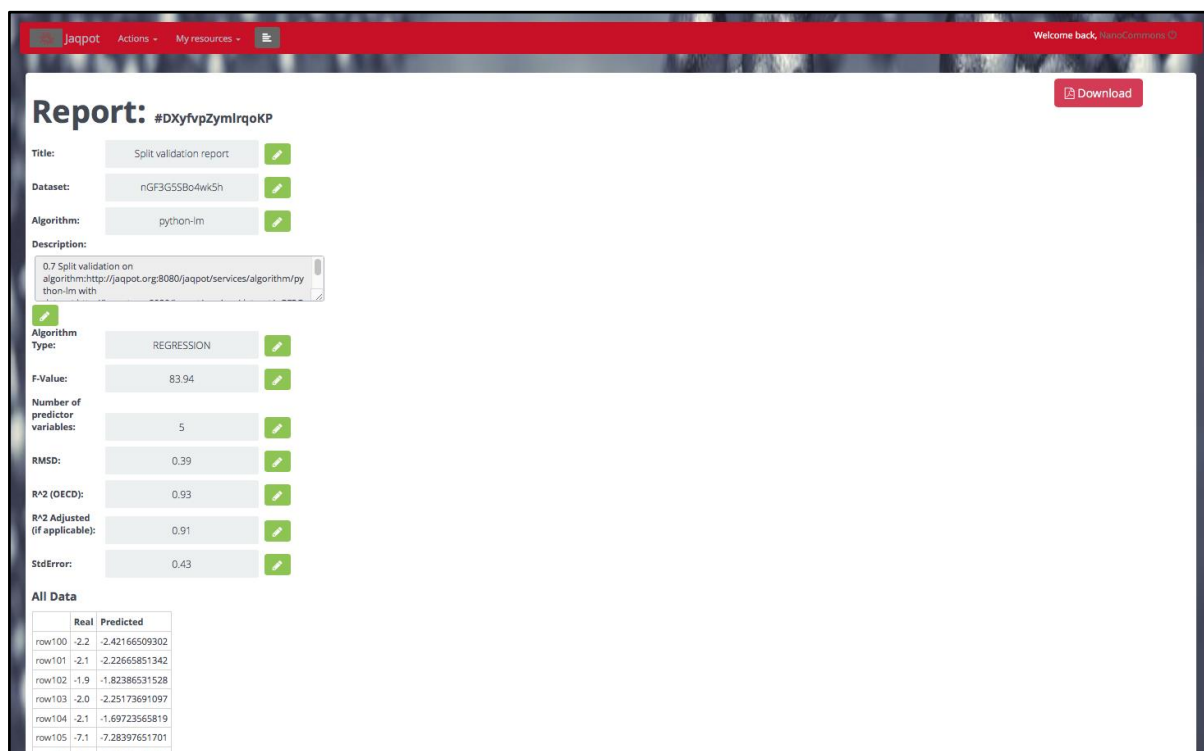


Figure 19. Algorithm parameters, model details & variable choice during Split Validation (Jaqpot 4).



*This report has been automatically created by the JaqpotQuatro report service. Click here to navigate to our official webpage*

### **Split validation report**

**Description:** [0.7 Split validation on algorithm:  
<http://jaqpot.org:8080/jaqpot/services/algorithm/python-lm> with dataset:  
<http://jaqpot.org:8080/jaqpot/services/dataset/nGF3G5SBo4wk5h>]

**Dataset:** nGF3G5SBo4wk5h

**Algorithm:** python-lm

**Algorithm Type:** REGRESSION

**F-Value:** 83.94

**Number of predictor variables:** 5

**RMSD:** 0.39

**R<sup>2</sup> (OECD):** 0.93

**R<sup>2</sup> Adjusted (if applicable):** 0.91

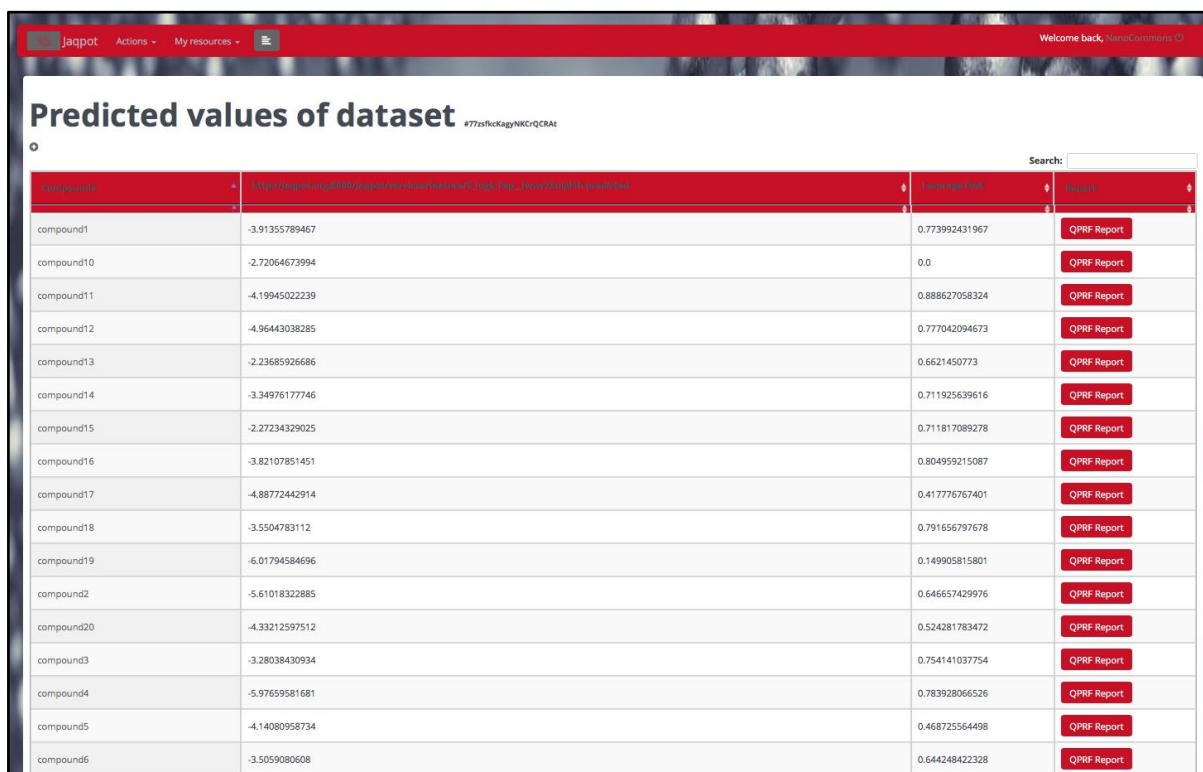
**StdError:** 0.43

**Procedure completed on:** Tue May 14 12:23:03 UTC 2019

**Figure 20.** Split Validation report for the nanoQSAR model (Jaqpot 4).

#### Prediction of end-point values using the nanoQSAR Model

After completing the Validation procedure, the nanoQSAR model web service can be used to make predictions of the end-point values. By clicking on the “*Predict*” button, users must choose a dataset, as in the Validation process. After providing the test data again and clicking “*Predict*”, the page in Figure 21 appears, where users are presented with the predicted values, calculated DoA values and a button that automatically creates the QSAR Prediction Reporting Format (QPRF) reports. DoA values close to 0 for predictions on a substance mean that this particular substance is out of the applicability domain, so we should not trust the prediction. In contrast, the further a DoA value is from 0, the more confidence the user can place in the reliability of the predictions, because the substance is within the DoA of the model.



**Predicted values of dataset** #772afckKagynKCrQCRAE

Search:

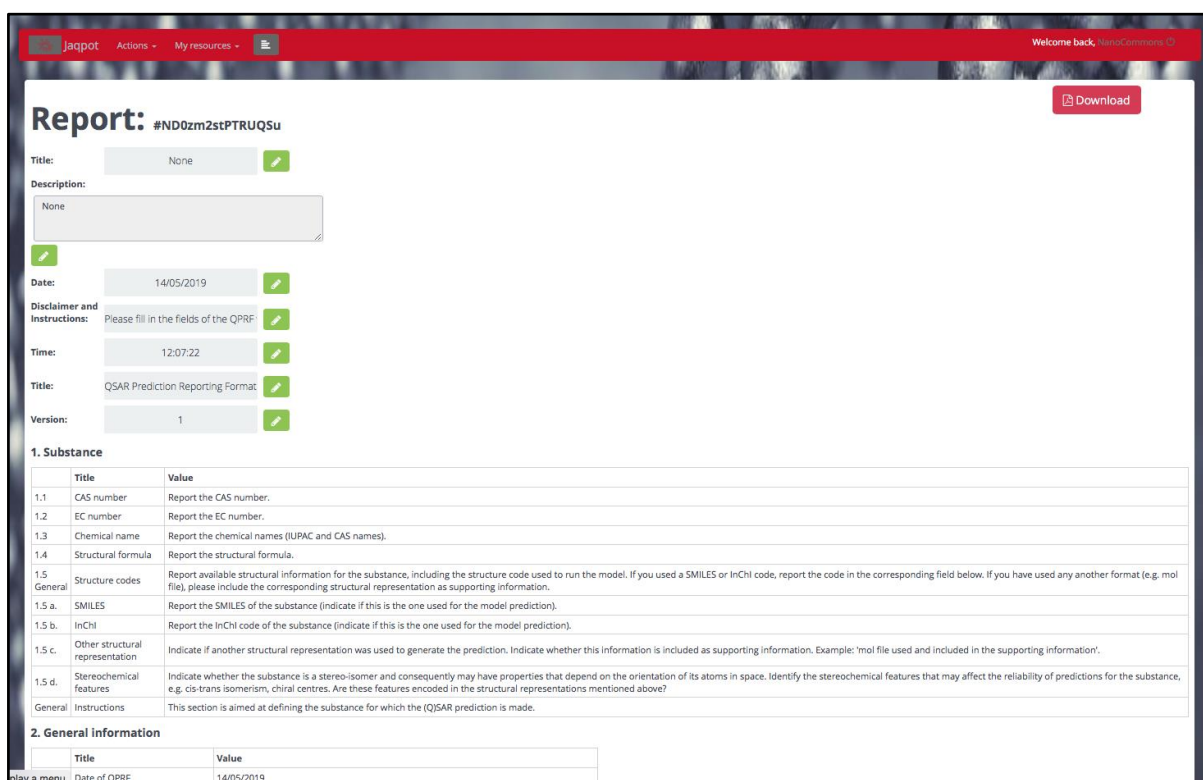
Compound	log <sub>10</sub> (K <sub>ow</sub> )	Toxicity DoA	Report
compound1	-3.91355789467	0.773992431967	<a href="#">QPRF Report</a>
compound10	-2.72064673994	0.0	<a href="#">QPRF Report</a>
compound11	-4.19945022239	0.888627058324	<a href="#">QPRF Report</a>
compound12	-4.96443038285	0.777042094673	<a href="#">QPRF Report</a>
compound13	-2.23685926686	0.6621450773	<a href="#">QPRF Report</a>
compound14	-3.34976177746	0.711925639616	<a href="#">QPRF Report</a>
compound15	-2.27234329025	0.711817089278	<a href="#">QPRF Report</a>
compound16	-3.82107851451	0.804959215087	<a href="#">QPRF Report</a>
compound17	-4.88772442914	0.417776767401	<a href="#">QPRF Report</a>
compound18	-3.5504783112	0.791656797678	<a href="#">QPRF Report</a>
compound19	-6.01794584696	0.149905815801	<a href="#">QPRF Report</a>
compound2	-5.61018322885	0.646657429976	<a href="#">QPRF Report</a>
compound20	-4.33212597512	0.524281783472	<a href="#">QPRF Report</a>
compound3	-3.28038430934	0.754141037754	<a href="#">QPRF Report</a>
compound4	-5.97659581681	0.783928066526	<a href="#">QPRF Report</a>
compound5	-4.14080958734	0.468725564498	<a href="#">QPRF Report</a>
compound6	-3.5059080608	0.644248422328	<a href="#">QPRF Report</a>

**Figure 21.** Prediction page with predicted values, DoA values and QPRF report for each prediction (each compound) (Jaqpot 4).

The QPRF report generated by Jaqpot (Figure 22) contains all the fields required by the OECD guidelines (OECD, 2007), namely:

- Substance
  - Contains information such as CAS and EC numbers, SMILES, InChi, *etc.* for all chemicals (solvents in this case, but equally for NMs) in the dataset
- 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.*

The function of QPRF report generation is particularly useful in the definition of Integrated Approaches for Testing and Assessment (IATA), where the OECD states that individual predictions, if applicable, should be reported using QPRFs (IATA, 2017). An example of a QPRF report page generated by Jaqpot is shown in Figure 22. All required fields are provided and are editable so they can be filled in with the additional details by the user. The report can also be downloaded as a PDF file. The produced editable QPRF report is given in Appendix 2.



**Report: #ND0zm2stPTRUQSu**

Title: None ✓

Description: None

Date: 14/05/2019 ✓

Disclaimer and Instructions: Please fill in the fields of the QPRF ✓

Time: 12:07:22 ✓

Title: QSAR Prediction Reporting Format ✓

Version: 1 ✓

**1. Substance**

Title	Value
1.1 CAS number	Report the CAS number.
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	Report available structural information for the substance, including the structure code used to run the model. If you used a SMILES or InChI code, report the code in the corresponding field below. If you have used any another format (e.g. mol file), please include the corresponding structural representation as supporting information.
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
Date of QPRF	14/05/2019

**Figure 22.** An example of a generated QPRF report (QSAR Prediction Reporting Format) for the fullerene C60 solubility nanoQSAR (Jaqpot 4). The QPRF report is available in Appendix 2.

## Jaqpot 5 GUI

The new Jaqpot GUI is being developed in the context of the NanoCommons project with the goals of scaling up with respect to number of users, size of datasets and algorithmic options available to the user. It also offers an improved user experience and more advanced options of sharing datasets and models with colleagues, specific organisations or the entire community.

Jaqpot has integrated the entire [Scikit-learn python library](#) which is the most comprehensive and perhaps the most popular open source library for machine learning, data mining and data analysis. There are plans to integrate algorithms from the [R language caret library](#) as well as techniques from Julia machine learning libraries, like [JuliaML](#).

The main tool developed by NTUA for integrating the Scikit-learn set of algorithms is the jaqpotpy library, which lets the user create a machine learning model in the python environment of their choice and deploy the model over the web. The guidelines for installing jaqpotpy can be found in the tool's [online documentation](#). They are also provided in this deliverable in Appendix 3.

The modeller can use the algorithm of their choice to fit the best possible model to the available training data and validate the model based on validation statistics. The user can also use optimal machine learning tools provided in Python, such as [TPOT](#), a Python Automated Machine Learning tool that optimizes machine learning pipelines using genetic programming.



### Model training & deployment to Jaqpot 5

For the particular case study that we use in this report for demonstration purposes, we can reproduce the model described in the literature and publish the model in Jaqpot in only a few lines of code which are shown next:

- A) Model training: After importing the jaqpotpy library and various components from the pandas library, the user uses commands to:
- read the csv file containing the data
  - define the independent features and the end-point
  - split the data into training and test sets
  - define a pipeline consisting of a scaling preprocessing step and the multiple linear regression algorithm
  - perform a 5-fold validation test
  - train the model

The python code with the output produced when it is run follows:

```
import pandas as pd

from jaqpotpy import Jaqpot

from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score, GridSearchCV, RandomizedSearchCV

df=pd.read_csv('70_model_reduced.csv') # Reads the data

print(list(df)) # Prints the headers of all columns
```

```
['Solvents', 'piPC03', 'ATS1m', 'Seigp', 'More23e', 'H1m', 'logS Exp.']
```

```
Xall=df[['piPC03', 'ATS1m', 'Seigp', 'More23e', 'H1m']] # Defines the columns that will be used as
independent features

Yall=df['logS Exp.'] # Defines the end-point

X_train, X_test, Y_train, Y_test = train_test_split(Xall, Yall, train_size=0.75, test_size=0.25,
random_state=1)
# Splits the data into training and test sets
```

```
stepslinear = [('scaler', MinMaxScaler()), ('MLR', LinearRegression())]
pipelinelinear = Pipeline(stepslinear) # define the pipeline object.

cross_val_score(estimator=pipelinelinear, X=X_train, y=Y_train, cv=5, n_jobs=-1) #Performs a 5-fold
cross validation
```

```
array([0.91906039, 0.88995619, 0.90445436, 0.86506266, 0.62316459])
```

```
pipelinelinear.fit(X_train, Y_train)
print('Training score: ', pipelinelinear.score(X_train, Y_train))
print('Testing score: ', pipelinelinear.score(X_test, Y_test))
print('Total score: ', pipelinelinear.score(Xall, Yall)) #Trains the model and prints R^2
statistics
```

Training score: 0.8994088488271355

Testing score: 0.9043040438111096

Total score: 0.9034772311898356

Model integration into Jaqpot: The user enters a username and password to enter their Jaqpot account and with only one command creates a web implementation of the model:

```
jaqpot = Jaqpot("https://api.jaqpot.org/jaqpot/services/")

jaqpot.request_key_safe()
```

Username: hсарimv

Password: .....

```
jaqpot.deploy_pipeline(pipelinelinear,Xall,Yall,"Linear Model for Predicting Solubility of C60 Fullerenes
in Various Solvents","Linear Model","linearmodel")
```

Model with id: Kz6NZU5Aqk5WajFx8OAO created. Please visit <https://app.jaqpot.org/>

- B) Creation of a PMML representation of the model: Optionally the user can apply the functionalities offered by the scikit learn library that automatically creates a PMML representation of the produced predictive model. A PMML model offers two significant benefits: a) a complete representation of the model that is both independent of programming language, thus allowing flow of information across platforms and architectures, and b) transparency on model parameters, so that experienced modellers can review the model,

which is crucial for them to accept the model. The PMML representation can be added to the documentation of the model as will be shown later in this report.

```
from sklearn2pmml.pipeline import PMMLPipeline

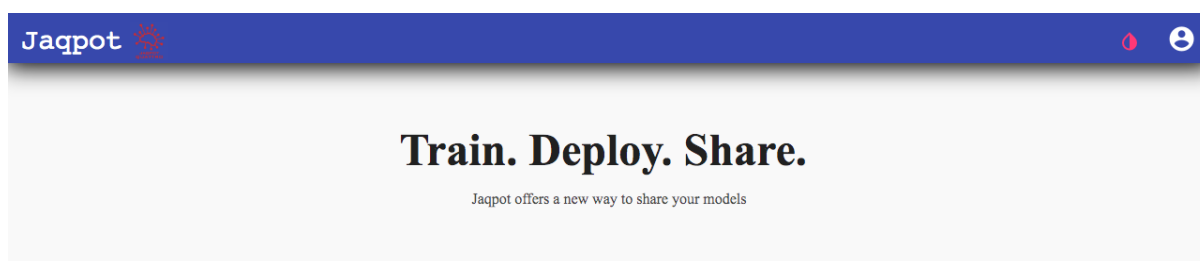
from sklearn2pmml.pipeline import PMMLPipeline
pipelinepmmllinear = PMMLPipeline([
    ("scaler", MinMaxScaler()), ("MLR", LinearRegression())
])
pipelinepmmllinear.fit(X_train, Y_train)

from sklearn2pmml import sklearn2pmml

sklearn2pmml(pipelinepmmllinear, "SolubilityC60linear.pmml", with_repr = True)
```

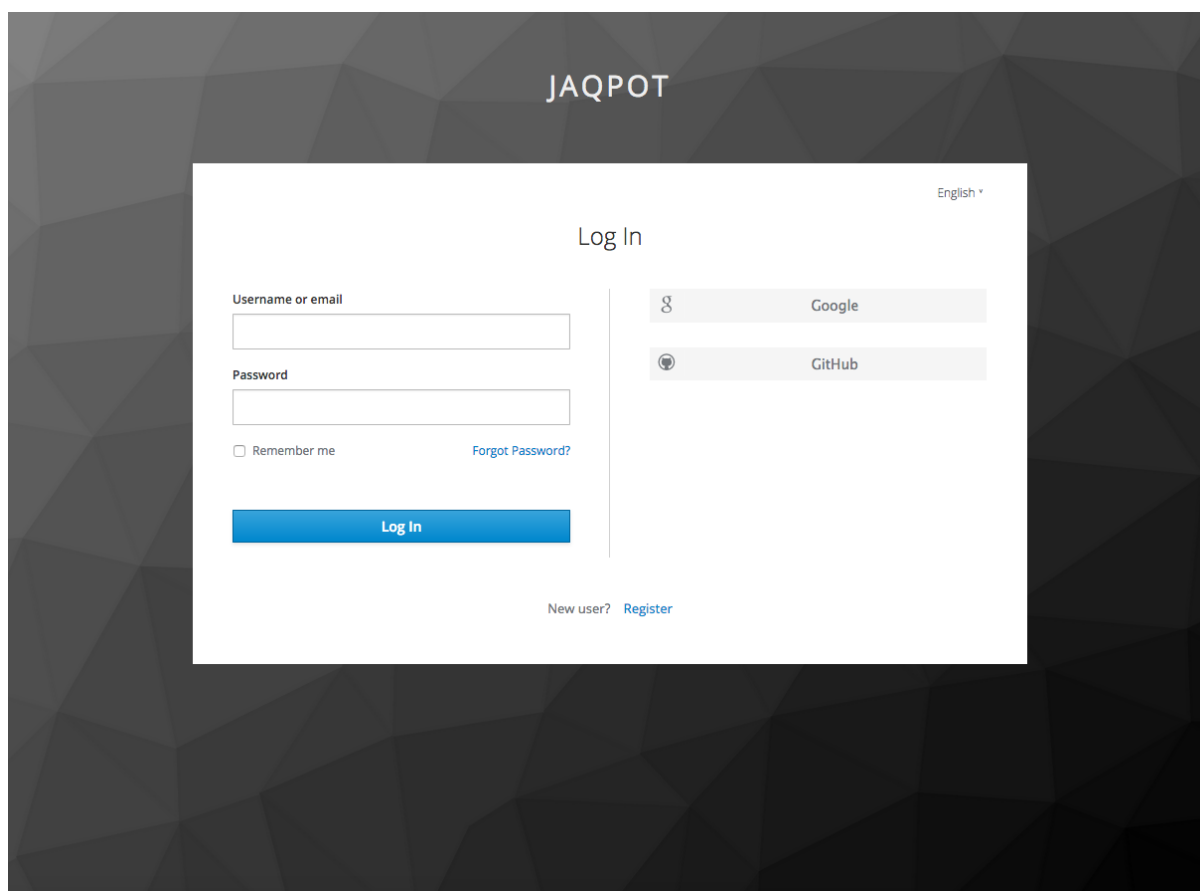
If the above steps are followed correctly, a ready-to-use web implementation of the model has been created with a unique URI identifier. The Jaqpot application offers tabs where the user can provide information and details about the model.

Please note that models are password-protected, so in order to view the model, the user needs to log in first. The starting page viewed by the user is shown in Figure 23:



**Figure 23.** Initial page of Jaqpot 5.

By clicking the icon on the top right of this screen, users can log in, as shown in the following screen (Figure 24):



**Figure 24.** Login page (Jaqpot 5).

Users are encouraged to create their own username using a username/password of their choice, or by using their existing accounts in Google or Github. To have access to the information shared in this report, a user should additionally be a member of the Jaqpot 5 NanoCommons organisations. To be invited, the user should send an e-mail message to the administrator of the NanoCommons Jaqpot 5 organisation ([hsarimv@central.ntua.gr](mailto:hsarimv@central.ntua.gr)) (more details about Jaqpot organisations and their use in NanoCommons are presented later in this report). For convenience, any information shared here can also be accessed after logging in with the guest account (username: *guest*, password: *guest*) and pasting the corresponding link, which however does not allow editing or running the models.

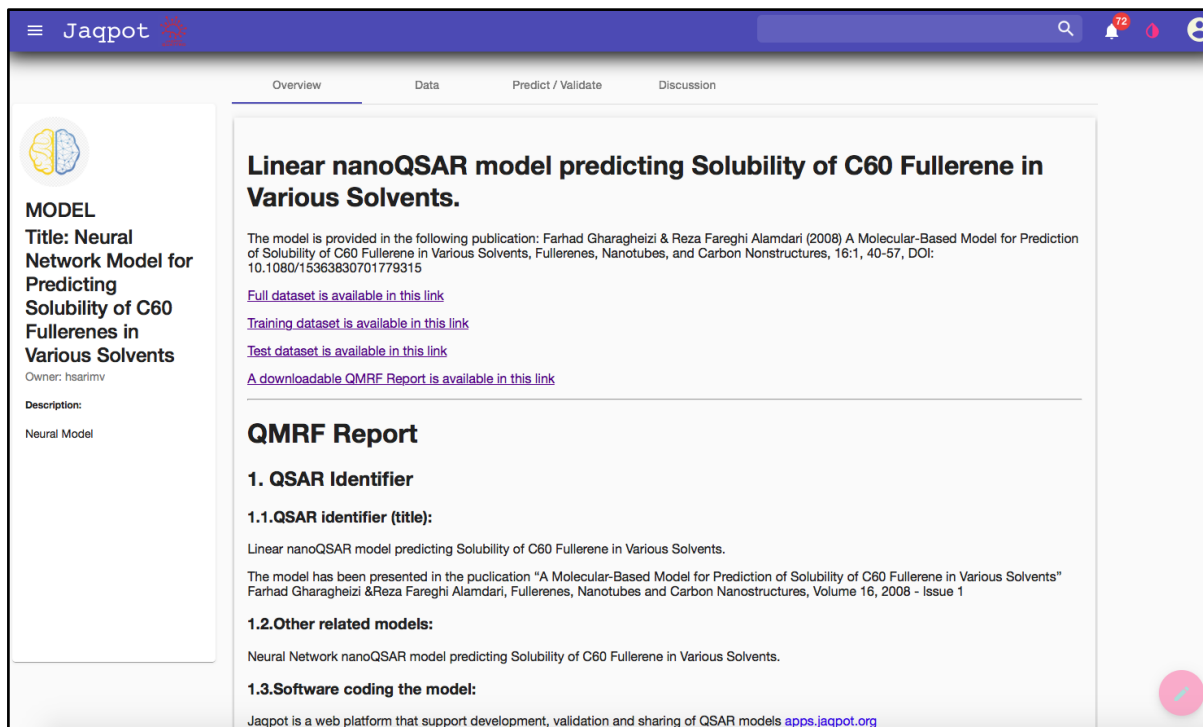
In the example described here we have created a model predicting solubility of C60 fullerenes that can be found in the following URI: <https://app.jaqpot.org/model/uxRBCMsV9IkSQT1Kw7km> (Figure 25).

There we are provided with a detailed description of the model:

Model page overview

The overview tab

The overview tab opens a markdown free-text editor where the user can include any information about the model. The editing mode can be activated by clicking the red icon on the bottom right of the page (Figure 25) and is deactivated by clicking on the floppy disk icon.



The screenshot displays the Jaqpot web interface. The top navigation bar includes a search icon, a notification bell with '72', and a user profile icon. Below the navigation bar, there are tabs for 'Overview', 'Data', 'Predict / Validate', and 'Discussion'. The main content area is titled 'Linear nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents.' and includes a description of the model, links to the full dataset, training dataset, and test dataset, and a link to a downloadable QMRF report. A sidebar on the left contains a 'MODEL' section with the title 'Neural Network Model for Predicting Solubility of C60 Fullerenes in Various Solvents' and the owner 'hsarimv'. The QMRF report section is titled 'QMRF Report' and contains three sub-sections: '1. QSAR Identifier', '1.1. QSAR identifier (title):', and '1.2. Other related models:'. The footer of the page states 'Jaqpot is a web platform that support development, validation and sharing of QSAR models [apps.jaqpot.org](https://apps.jaqpot.org)'.

Figure 25. Model web page - Overview tab (Jaqpot 5).

In the model webpage links to the full dataset, the training dataset and the test data sets are provided, as well as a link to a full QMRF report generated using the QMRF editor provided by JRC: <https://sourceforge.net/projects/qmrf/>. The full text of the QMRF report is provided in Appendix 4. A screenshot of the first part of the QMRF report is shown below (Figure 26).

	<b>QMRF identifier (JRC Inventory): To be entered by JRC</b>
	<b>QMRF Title:</b> Linear nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents. The model has been presented in the publication “A Molecular?Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents” Farhad Gharagheizi & Reza Fareghi Alamdari, Fullerenes, Nanotubes and Carbon Nanostructures, Volume 16, 2008 - Issue 1
<b>Printing Date: 22-Apr-2019</b>	

<b>1.QSAR identifier</b>
--------------------------

**1.1.QSAR identifier (title):**  
Linear nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents. The model has been presented in the publication “A Molecular?Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents” Farhad Gharagheizi & Reza Fareghi Alamdari, Fullerenes, Nanotubes and Carbon Nanostructures, Volume 16, 2008 - Issue 1

**1.2.Other related models:**  
Neural Network nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents.

**1.3.Software coding the model:**  
Jaqpote  
Jaqpote is a web platform that supports development, validation and sharing of QSAR models  
Haralambos Sarimveis  
apps.jaqpote.org

<b>2.General information</b>
------------------------------

**2.1.Date of QMRF:**  
21 April 2019

**Figure 26.** QMRF (QSAR Model Reporting Format) report (Jaqpote 5). The full QMRF report is available in Appendix 4.

In the overview tab, we have also included the QMRF report in an editable form, in case the creator of the model wants to add, delete or modify some information about the model. To assist the users in adding editable versions of QMRF reports, we have created a QMRF markdown template, which can be [downloaded here](#). The user only needs to provide the necessary information under each section and the QMRF report is generated in an easy-to-read format.

In the overview tab, we have also included the PMML representation of the model, which provides the full mathematical description of the model (variables used, scaling factors and model coefficients)

and allows easy transferring of the model to other platforms. The PMML representation of the specific model generated using the commands shown above (Jaqpote 5) is given in Appendix 5.

### The data tab

Here the model creator can provide specific information about the independent features and the end-point of the model (Figure 27): descriptions, units and ontological classes, which will allow the model to understand data sets that are ontologically annotated automatically. (Figure 27 provides a screenshot of the data tab under the C60 solubility model.

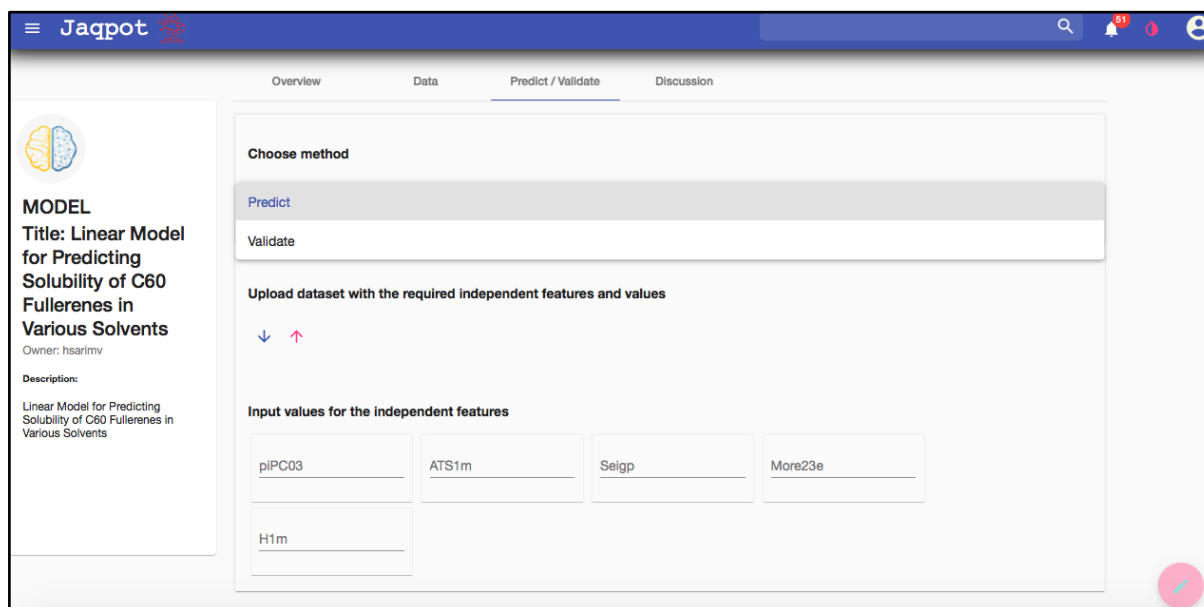
The screenshot shows the Jaqpote web interface. The top navigation bar includes the Jaqpote logo, a search icon, a notification bell with '72', and a user profile icon. Below the navigation bar are four tabs: Overview, Data (selected), Predict / Validate, and Discussion. The main content area is divided into two sections. On the left is a sidebar with a model icon and the following text: **MODEL**, Title: Neural Network Model for Predicting Solubility of C60 Fullerenes in Various Solvents, Owner: hsrarimv, Description: Neural Model. The main content area is titled 'Dependent feature / Predicted feature' and contains a box for 'logS Exp.' with a description: 'Feature created to link to independent feature of model Neural Network Model for Predicting Solubility of C60 Fullerenes in Various Solvents'. Below this is the 'Independent features' section, which contains three boxes: 'ATS1m', 'piPC03', and 'Seigp', each with a similar description. A pink circular icon is visible in the bottom right corner of the main content area.

**Figure 27.** Model webpage - Data tab (Jaqpote 5).

### The predict/validate tab

This tab contains the main functionalities of the model (Figure 28). The user who has access to the model can either generate predictions for NMs with unknown end-point values or test the model with a data set containing end-point values.

The data can be either entered by hand (for relatively small datasets) or uploaded using csv templates which are automatically generated for each model and can be downloaded in order to populate them with data (by clicking the blue down-pointing arrow). The templates contain all input variable names, so the user can add the respective values in each column and upload the data by clicking the red upwards-pointing arrow (Figure 28).



**MODEL**  
**Title: Linear Model for Predicting Solubility of C60 Fullerenes in Various Solvents**  
 Owner: hsarimv  
 Description:  
 Linear Model for Predicting Solubility of C60 Fullerenes in Various Solvents

**Choose method**

Predict  
 Validate

Upload dataset with the required independent features and values

↓ ↑

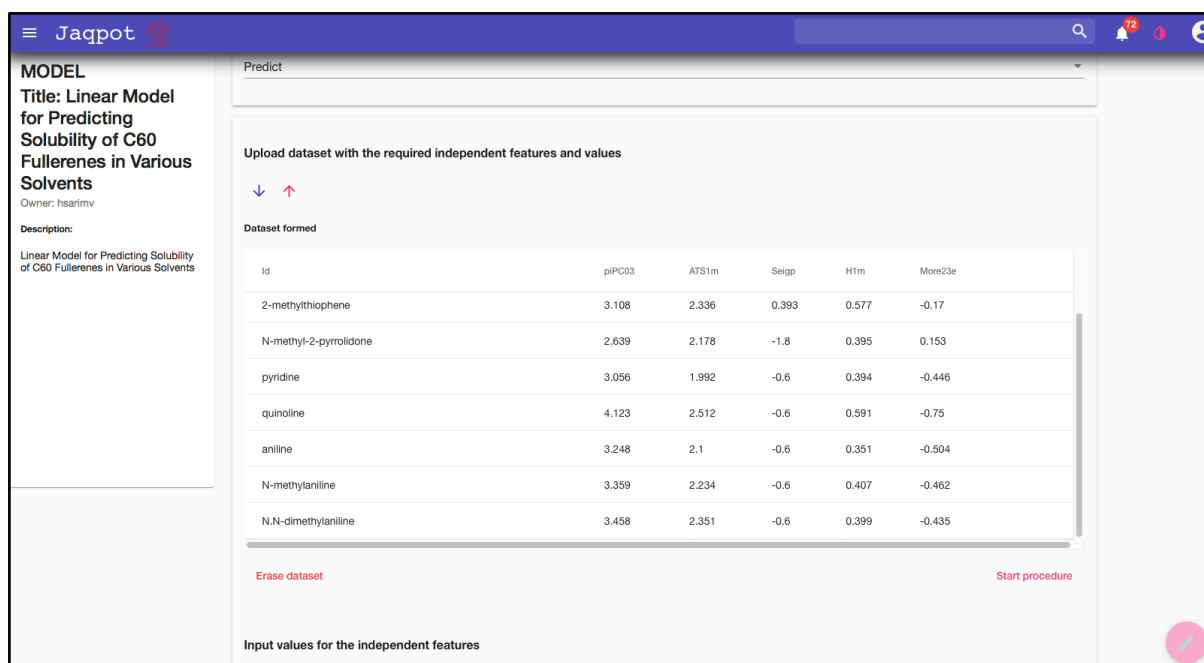
**Input values for the independent features**

piPC03    ATS1m    Seigp    More23e

H1m

**Figure 28.** Model webpage - Predict/Validate tab and input data template download (green downward facing arrow) and completed dataset upload (red upward facing arrow) (Jaqpot 5).

After uploading the data from the template with an “id” column, the user instructs Jaqpot to accept the id’s of each compound which generates a preview of the dataset (Figure 29). Clicking the “Start procedure” button shows us Task progress (Figure 30).



**MODEL**  
**Title: Linear Model for Predicting Solubility of C60 Fullerenes in Various Solvents**  
 Owner: hsarimv  
 Description:  
 Linear Model for Predicting Solubility of C60 Fullerenes in Various Solvents

Predict

Upload dataset with the required independent features and values

↓ ↑

**Dataset formed**

Id	piPC03	ATS1m	Seigp	H1m	More23e
2-methylthiophene	3.108	2.336	0.393	0.577	-0.17
N-methyl-2-pyrrolidone	2.639	2.178	-1.8	0.395	0.153
pyridine	3.056	1.992	-0.6	0.394	-0.446
quinoline	4.123	2.512	-0.6	0.591	-0.75
aniline	3.248	2.1	-0.6	0.351	-0.504
N-methylaniline	3.359	2.234	-0.6	0.407	-0.462
N,N-dimethylaniline	3.458	2.351	-0.6	0.399	-0.435

Erase dataset    Start procedure

Input values for the independent features

**Figure 29.** Model webpage - Predict/Validate tab (Jaqpot 5).



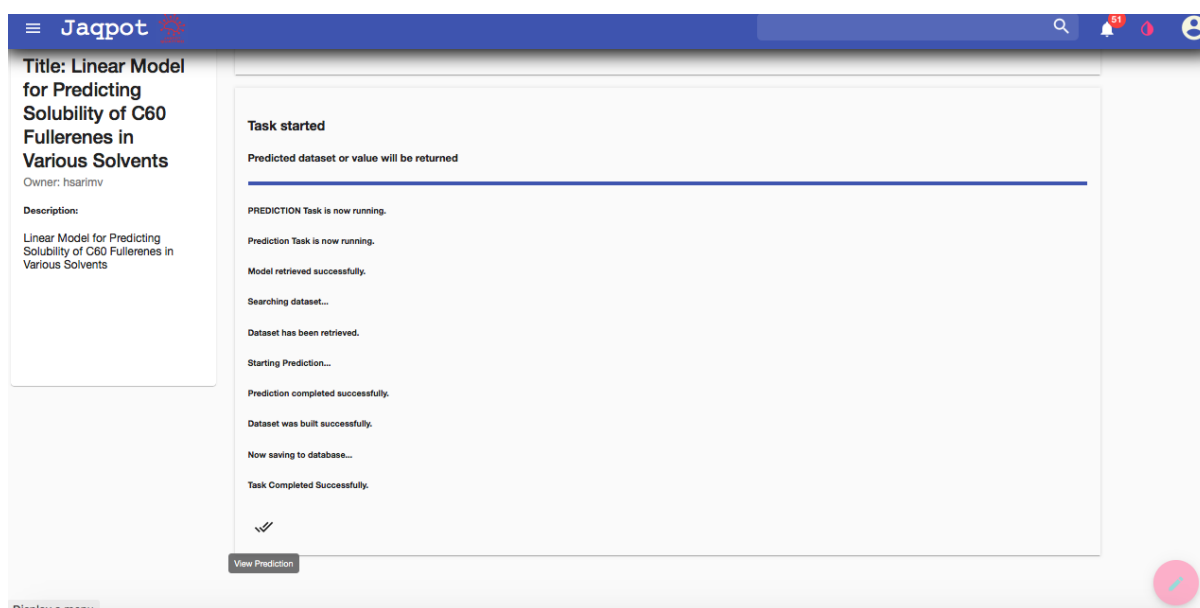
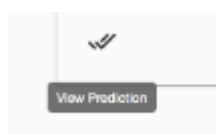
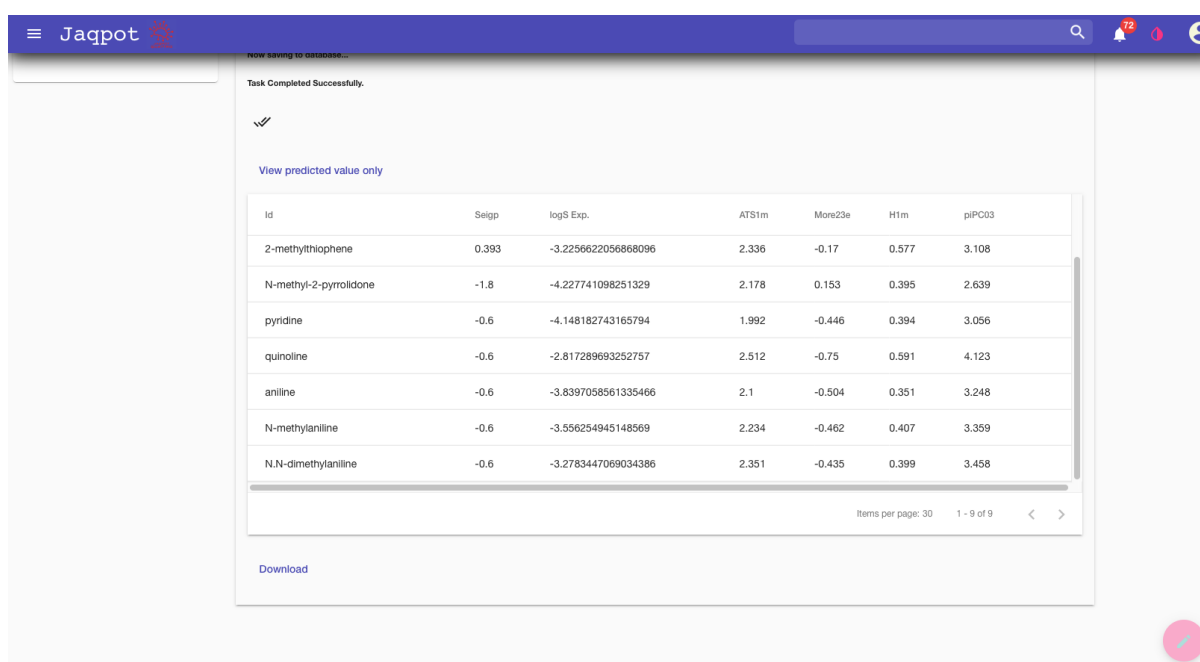


Figure 30. Model webpage - Predict/Validate tab (Jaqpot 5).



Clicking the View Predictions icon (Figure 30), allows the user to view the predictions (Figure 31). In order to validate the model, the user selects the Validate option in Figure 28, which produces a report with validation statistics (Figure 31), accompanied with a QQ plot (Figure 32) and *Real vs Predicted* values plot (Figure 33), giving insight into the effectiveness of the model.



Id	Seipp	logS Exp.	ATStm	More23e	H1m	piPC03
2-methylthiophene	0.393	-3.2256622056868096	2.336	-0.17	0.577	3.108
N-methyl-2-pyrrolidone	-1.8	-4.227741098251329	2.178	0.153	0.395	2.639
pyridine	-0.6	-4.148182743165794	1.992	-0.446	0.394	3.056
quinoline	-0.6	-2.817289693252757	2.512	-0.75	0.591	4.123
aniline	-0.6	-3.8397058561335466	2.1	-0.504	0.351	3.248
N-methylaniline	-0.6	-3.56254945148569	2.234	-0.462	0.407	3.359
N,N-dimethylaniline	-0.6	-3.2783447069034386	2.351	-0.435	0.399	3.458

Figure 31. Model webpage - Predicted values after modelling (Jaqpot 5).

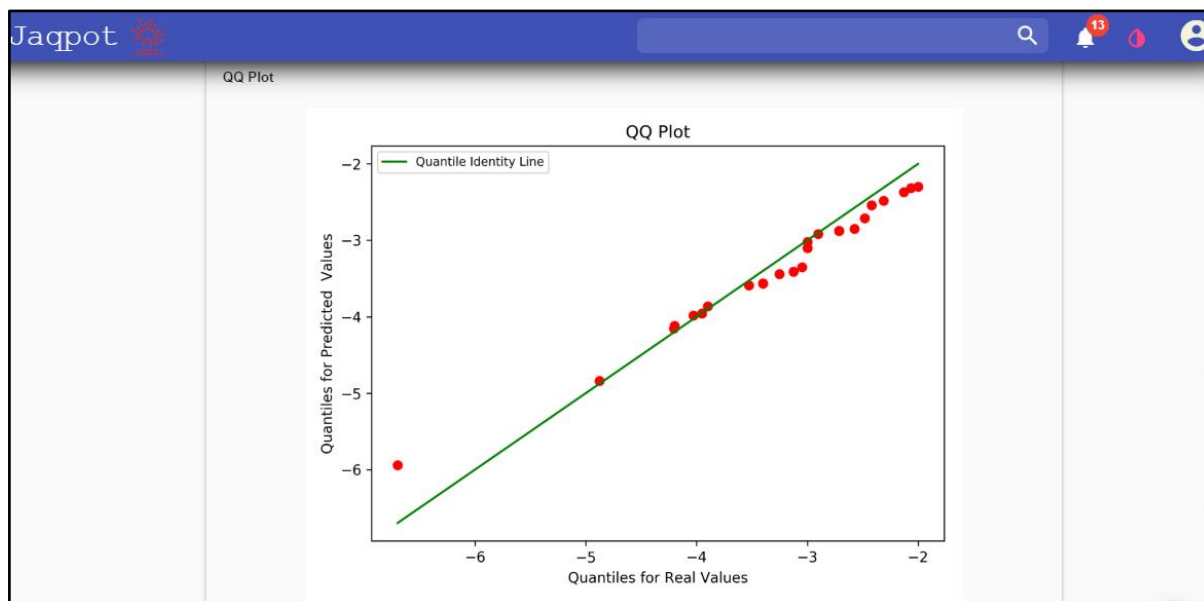


Figure 32. Validation results - QQ plot (Jaqpot 5).

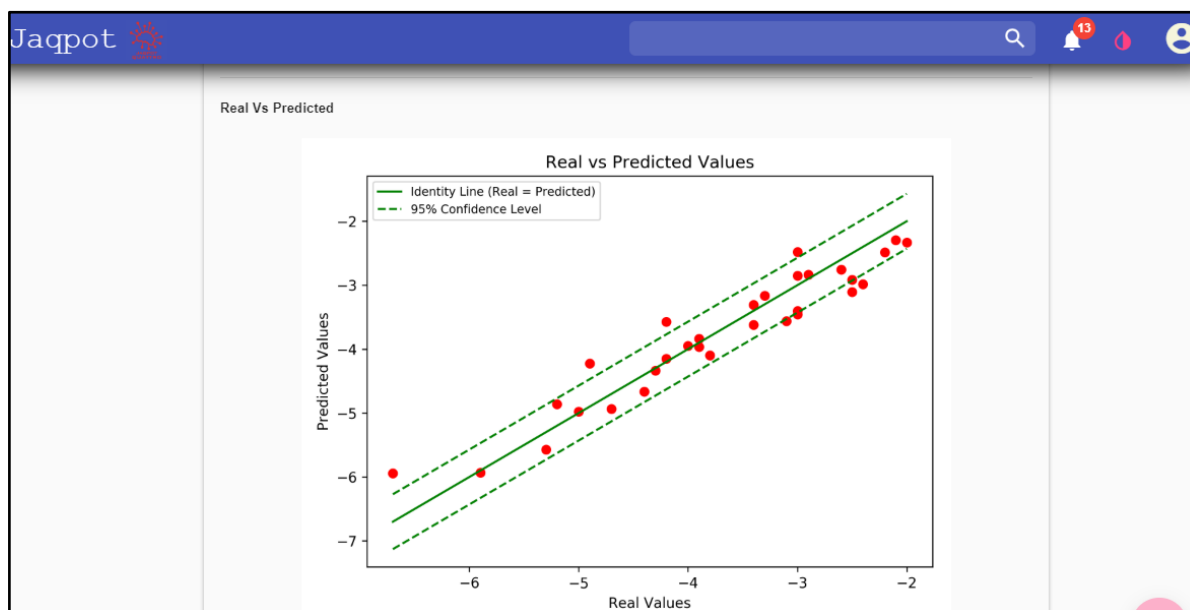


Figure 33. Validation results - *Real vs Predicted* plot (Jaqpot 5).

### Social networking and Sharing of resources in Jaqpot 5

Collaboration is a central part of advancing science. In NanoCommons, collaboration in nanoinformatics is elaborated by the social networking and sharing features in Jaqpot.

### Organisations

A feature introduced in Jaqpot 5 is Organisations, which are common spaces for resources, *i.e.* datasets and models. They allow the user to instantly share a resource with a group and receive

feedback. A user can check the organisations they are participating in their profile page (Figure 34), accessible by clicking the icon on the top right of the screen.

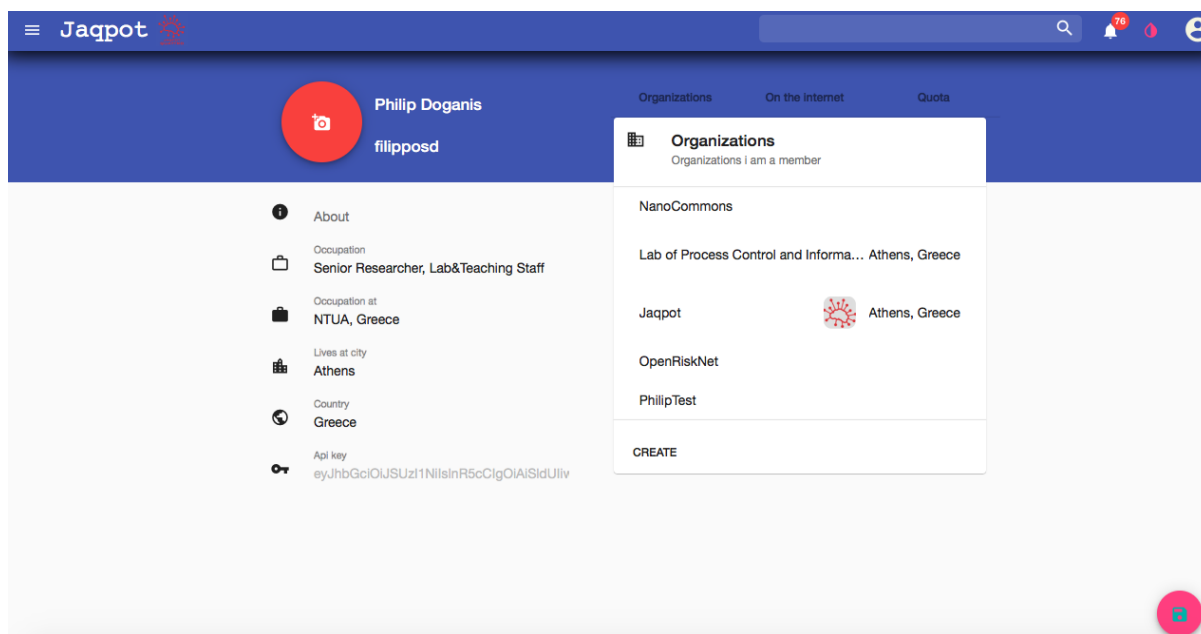


Figure 34. Example Profile page (Jaqpot 5).

Clicking on one of the organisations, opens a small preview window with brief information about the organisation - the NanoCommons project in this example (Figure 35).

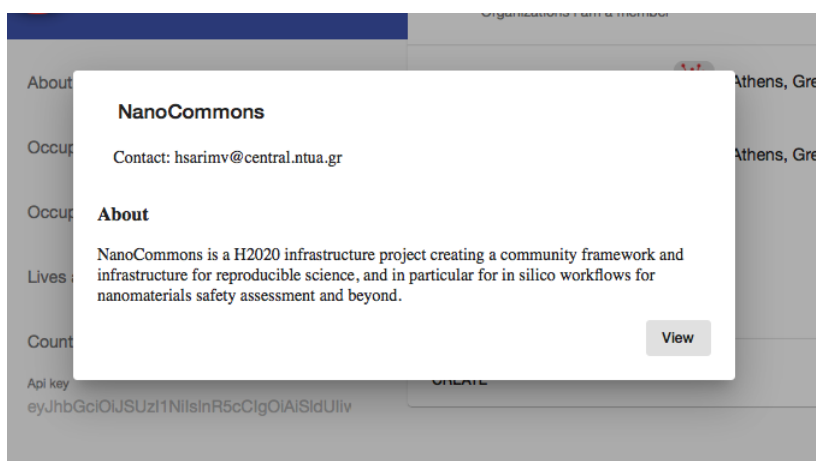


Figure 35. Organisation information preview (Jaqpot 5).

The *View* button leads to the NanoCommons organisations homepage, where one can access an overview, its members and the contact person (Figure 36).

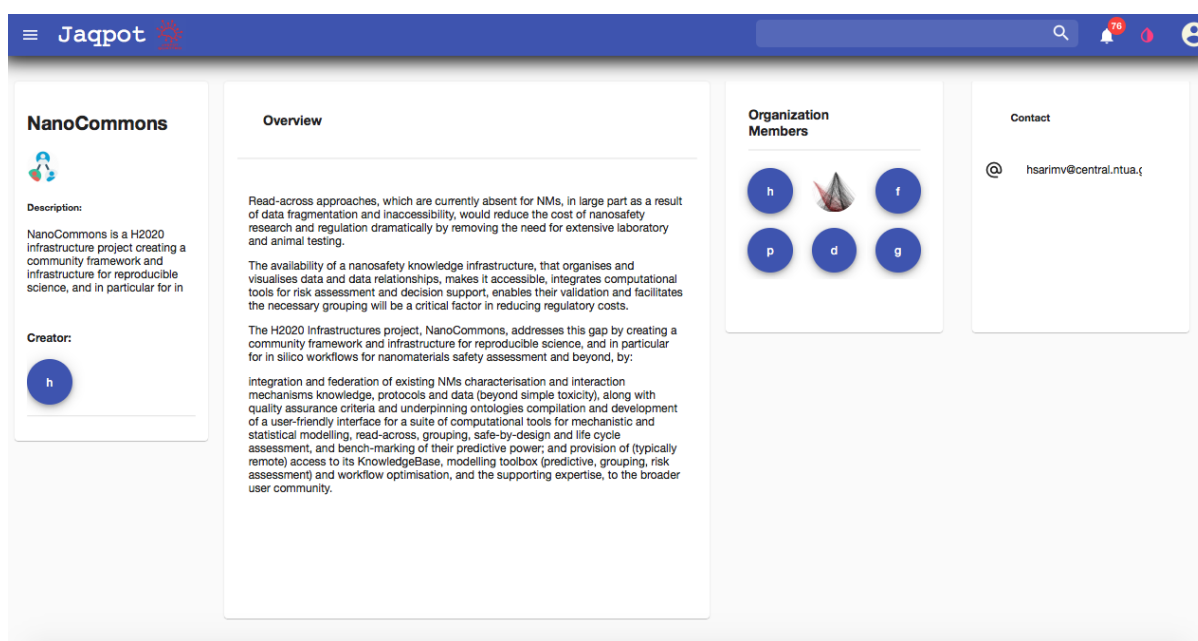


Figure 36. NanoCommons organisation page (Jaqpot 5).

A new organisation can be created by clicking the *CREATE* button in the profile page (Figure 34). A window appears, where the user is required to enter a name for the new organisation, which can be accessed at its URL <https://app.jaqpot.org/organization/TestPhilip> (Figure 37).

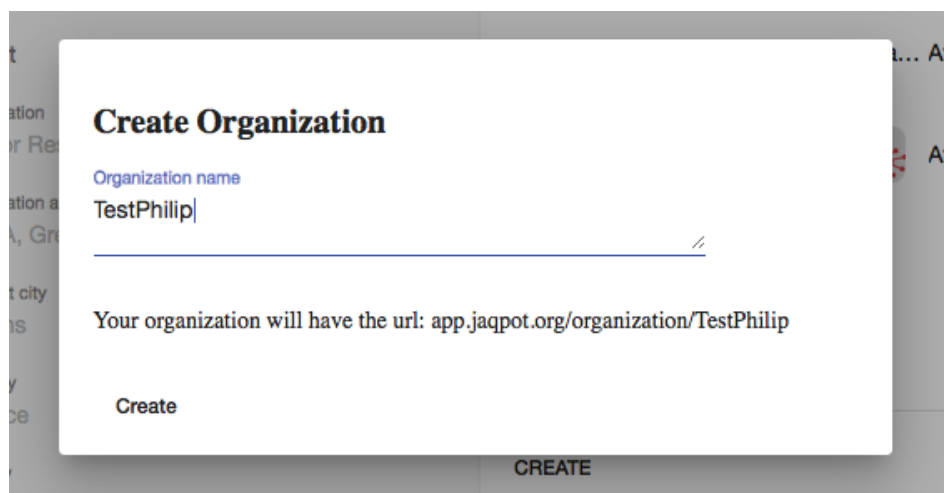


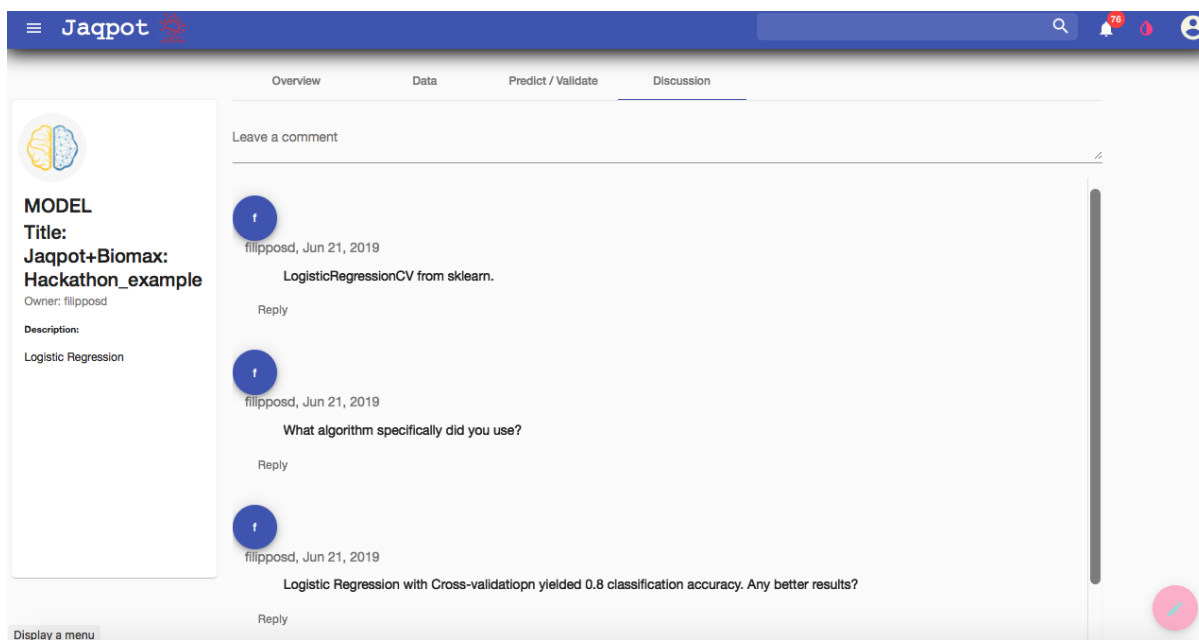
Figure 37. Example of a user-created organisation page, called TestPhilip (Jaqpot 5).

### Getting access to the NanoCommons organisation

As mentioned before, access to an organisation can be given by invitation from its administrator, whose contact details are provided in the *Contact* field in the organisation homepage. As shown in Figure 36, in the case of the NanoCommons organisation, one should email this address: [hsarimv@central.ntua.gr](mailto:hsarimv@central.ntua.gr), in order to contact Prof. Haralambos Sarimveis.

## Discussions

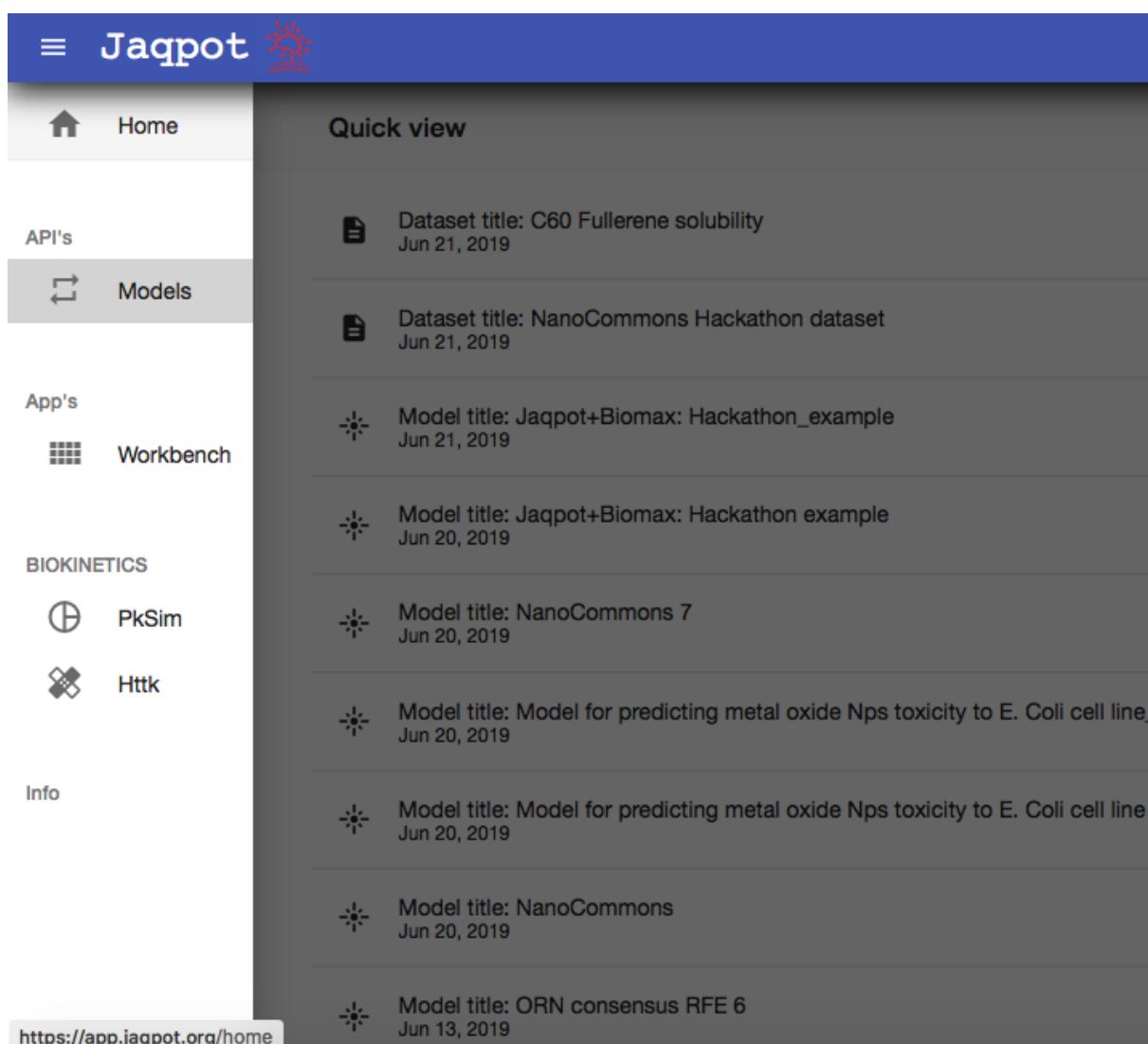
A vital element that is provided in models and datasets in Jaqpot 5 is accessed through the *Discussions* tab (Figure 38). Through the exchange of ideas, researchers (but also model/dataset providers in general) can receive feedback on their work and improve it. New users can build on previous collaborative work and provide their own insights.



**Figure 38.** Discussions tab of a model (Jaqpot 5).

## Private and Shared space for models and datasets

By clicking the icon with the three stripes on the top left corner, the user can access a slide-out menu that links to the user home (Figure 39) which provides a quick view of the user's private space, sorted by date.



https://app.jaqpot.org/home

**Figure 39.** Side menu showing an overview of the user's private space, sorted by date (Jaqpot 5).

By selecting the *Models* item from the menu, first the user sees private models (as indicated by *Models*> *Mine*). Clicking on the arrow next to *Mine*, a pop up menu giving a choice between *Mine* and *Shared* is revealed (Figure 40).

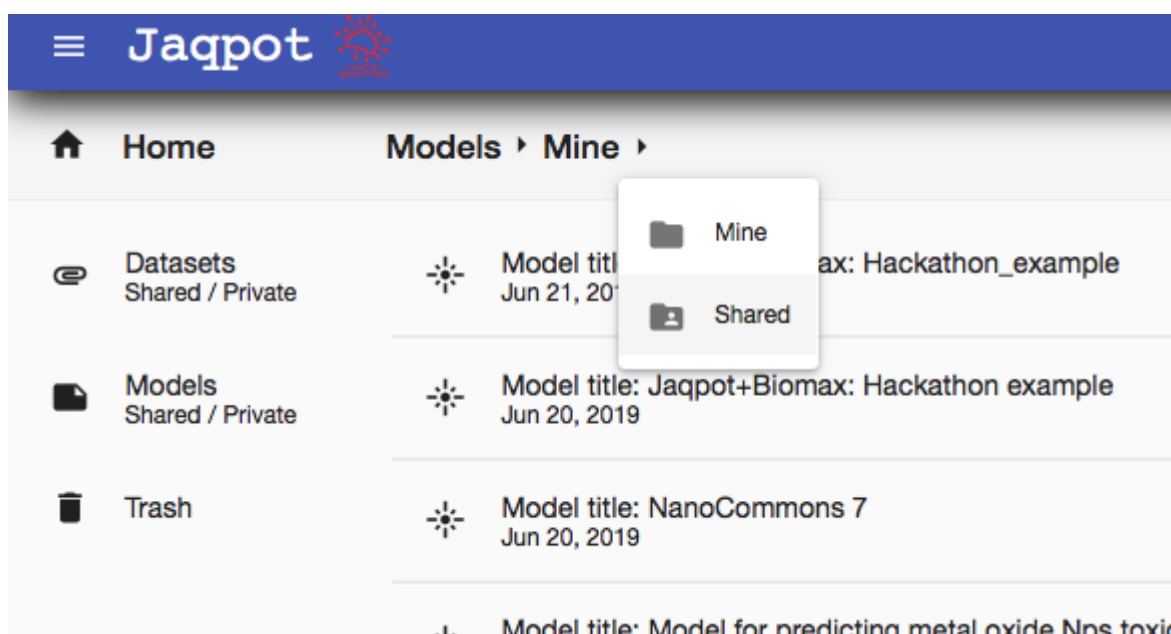


Figure 40. Revealing the Shared models space (Jaqpot 5).

A subsequent pop-up menu allows a choice among the organisations to which the user participates (Figure 41). Selecting NanoCommons, the Shared models page of the NanoCommons organisation is revealed (Figure 42), where clicking the “eye” icon in the highlighted C60 Fullerenes model leads to the model page, as shown above in Figure 25.

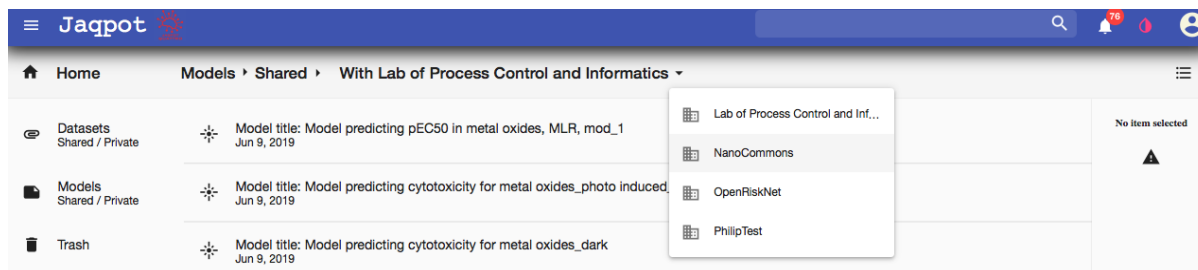
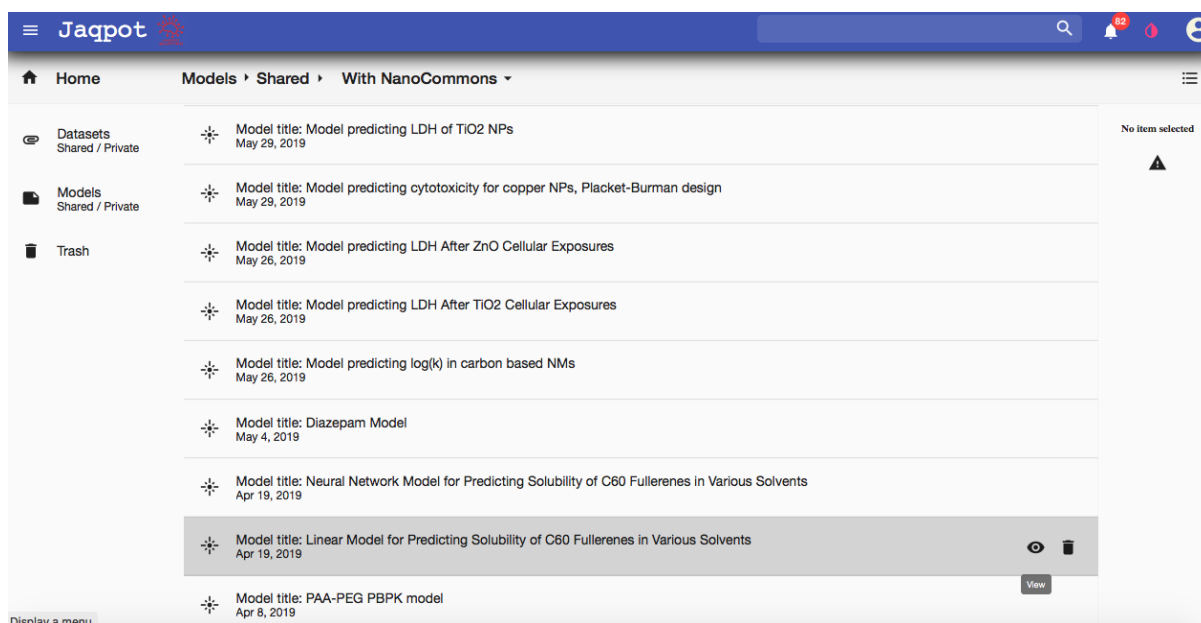


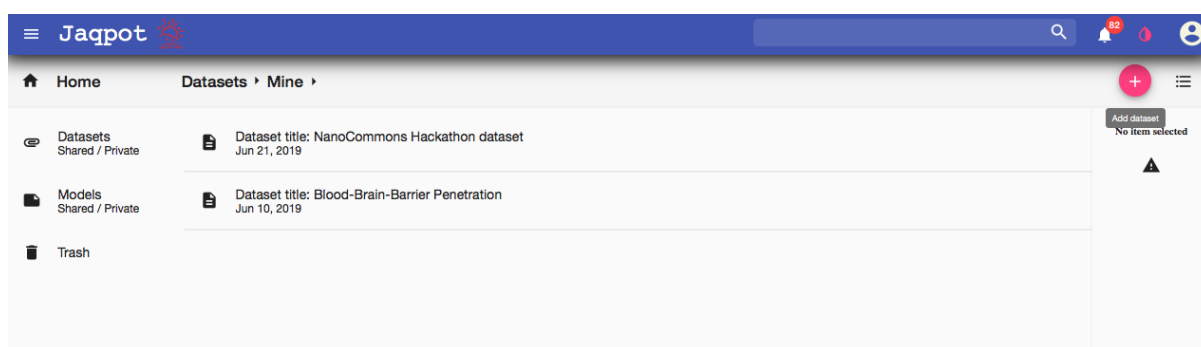
Figure 41. Revealing the Shared models space (Jaqpot 5).



**Figure 42.** The Shared models space of the NanoCommons organisation. The C<sub>60</sub> Fullerenes solubility model is highlighted (Jaqpot 5).

### Uploading a new dataset

The dataset upload feature of Jaqpot 5 is demonstrated by uploading the C<sub>60</sub> Fullerenes dataset from Gharagheizi & Alamdari (2008). A new dataset can be uploaded when we have accessed the *Datasets* item and the “Add dataset” icon appears (Figure 43) and after selecting the CSV file (Figure 44), which must be structured with a header row for the names of the features and a column for compound names (as in Figure 45).



**Figure 43.** Private dataset space, showing the icon for adding a dataset (Jaqpot 5).



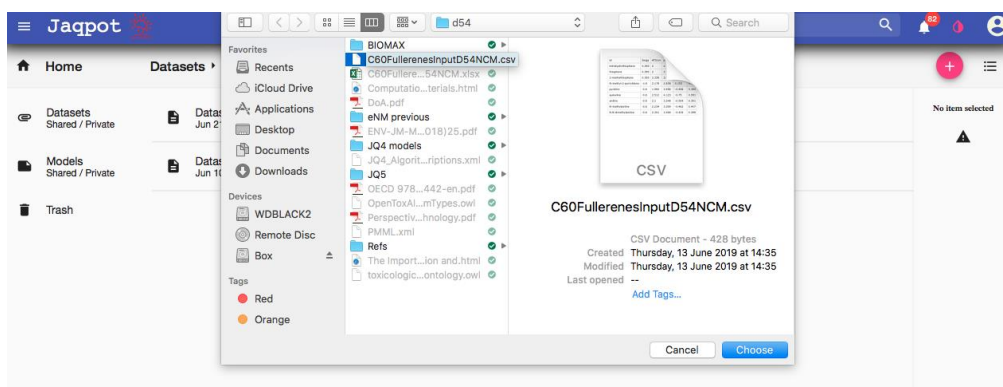
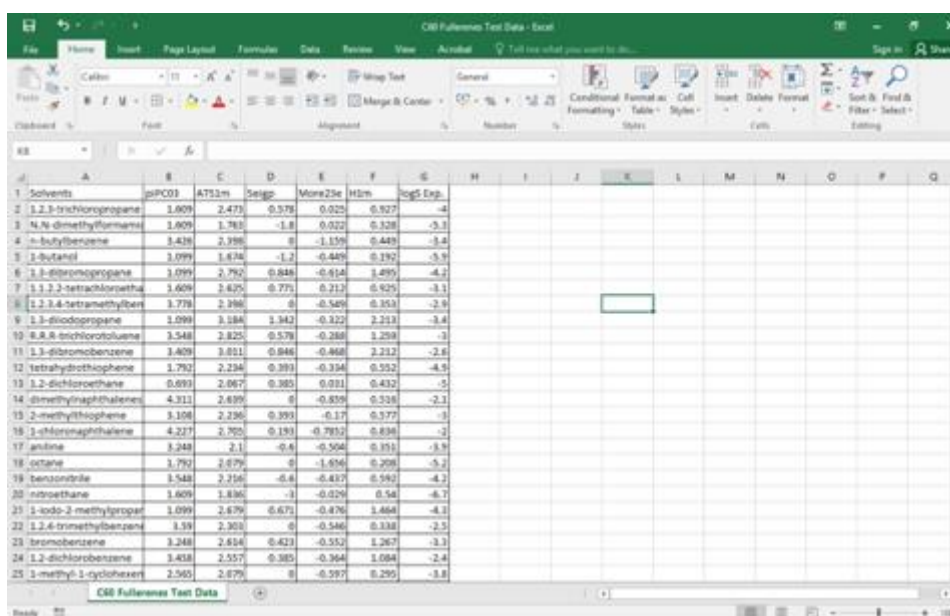


Figure 44. Adding a dataset - Selection of CSV file (Jaqpot 5).



The screenshot shows an Excel spreadsheet titled 'C60 Fullerenes Test Data - Excel'. The spreadsheet contains a table with 25 rows and 7 columns. The columns are labeled: Solvents, spFC03, AT13m, Selgc, Minr23e, H3m, and logp Exp. The data represents various solvents and their corresponding parameters.

Solvents	spFC03	AT13m	Selgc	Minr23e	H3m	logp Exp.
1,2,3-trichloropropane	1.809	2.473	-0.578	0.025	0.927	-4
N,N-dimethylformamide	1.809	1.782	-1.8	0.025	0.328	-9.3
n-hexylbenzene	1.438	2.398	0	-1.159	0.449	-8.4
1-butanol	1.099	1.824	-1.2	-0.449	0.192	-9.3
1,1-dibromopropane	1.099	1.792	0.848	-0.614	1.499	-4.2
1,1,2,2-tetrachloroethane	1.609	2.405	0.771	0.212	0.925	-3.1
1,2,3,4-tetramethylbenzene	1.778	2.346	0	-0.549	0.353	-2.9
1,3-dioxopropane	1.099	1.884	1.342	-0.322	2.213	-8.4
8,8,8-trichlorotoluene	1.548	2.825	0.578	-0.286	1.258	-1
1,3-dibromobenzene	1.409	1.811	0.846	-0.468	2.212	-2.6
tetrahydrothiophene	1.792	2.234	0.391	-0.334	0.552	-4.9
1,2-dichloroethane	-0.893	2.067	0.385	0.031	0.432	-5
dimethylnaphthalene	-4.311	2.499	0	-0.859	0.538	-2.1
2-methylthiophene	1.108	2.236	0.399	-0.17	0.577	-1
1-chloronaphthalene	-4.227	2.705	0.191	-0.7932	0.834	-2
aniline	1.248	2.1	-0.4	-0.504	0.351	-3.9
octane	1.792	2.679	0	-1.654	0.208	-5.2
benzotrifluoride	1.548	2.216	-0.4	-0.437	0.542	-4.2
nproethane	1.609	1.836	-1	-0.029	0.54	-6.7
1-iodo-2-methylpropane	1.099	2.679	-0.671	-0.476	1.464	-4.1
1,2,4-trimethylbenzene	1.59	2.301	0	-0.546	0.338	-2.5
benzobenzene	1.248	2.434	0.423	-0.552	1.267	-1.3
1,2-dichlorobenzene	1.438	2.557	0.385	-0.364	1.084	-2.4
1-methyl-1-cyclohexene	2.565	2.679	0	-0.597	0.295	-3.8

Figure 45. Adding a dataset - General structure of CSV file (Jaqpot 5).

After that, the user is prompted to choose the id of the dataset (Figure 46), which can be *None*, or one of the feature names specified in the first row of the CSV. In a subsequent window (Figure 47), a preview of the dataset entered is shown (please bear in mind that poorly formatted CSVs can lead to problematic uploads), defining the main parameters of the dataset and providing information on the variables involved (a brief Description, Units and Ontological Classes, where available).

The user is informed of a successful upload with a pop-up window (Figure 48), displaying the id of the dataset and its title.



Figure 46. Adding a dataset - Choice of dataset id from the first-row entries (Jaqpot 5).

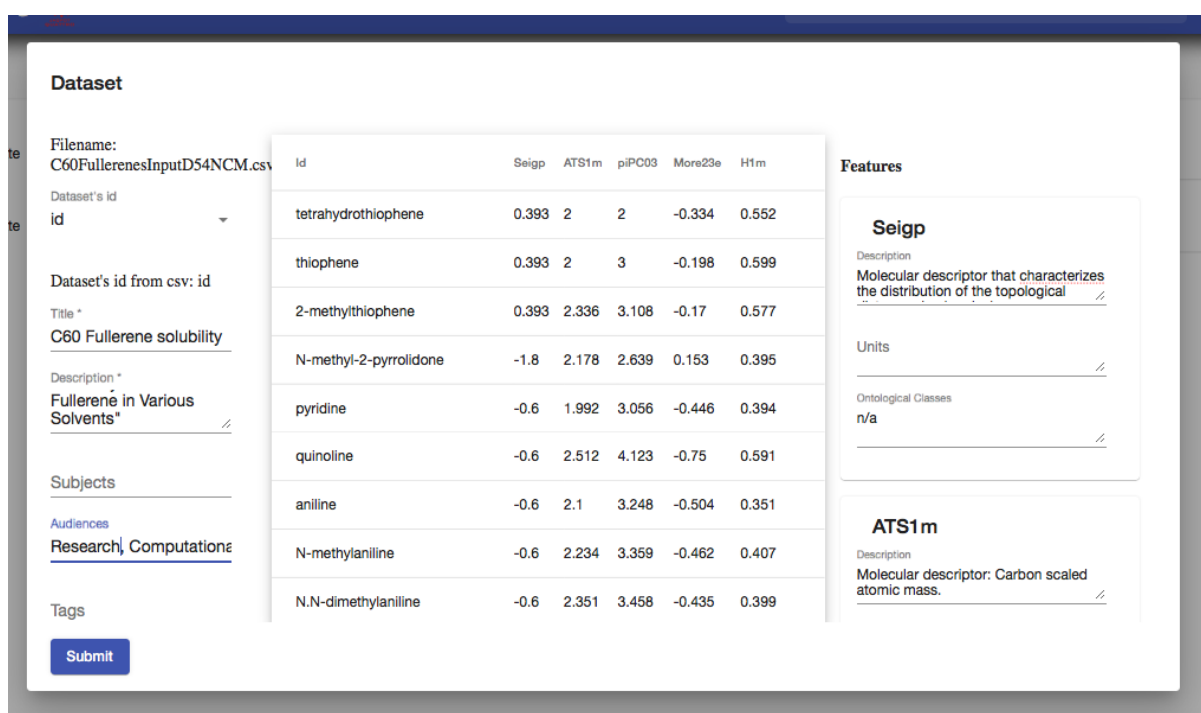


Figure 47. Adding a dataset - Filling in dataset information (Jaqpot 5).

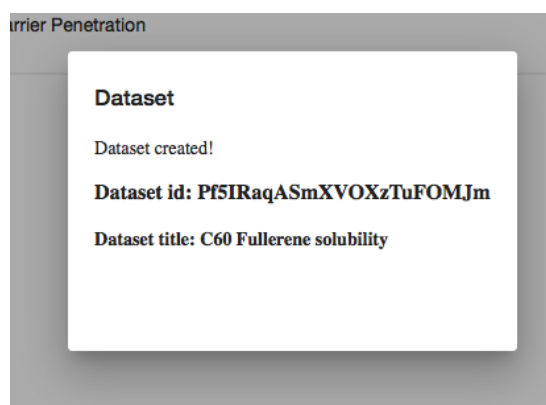
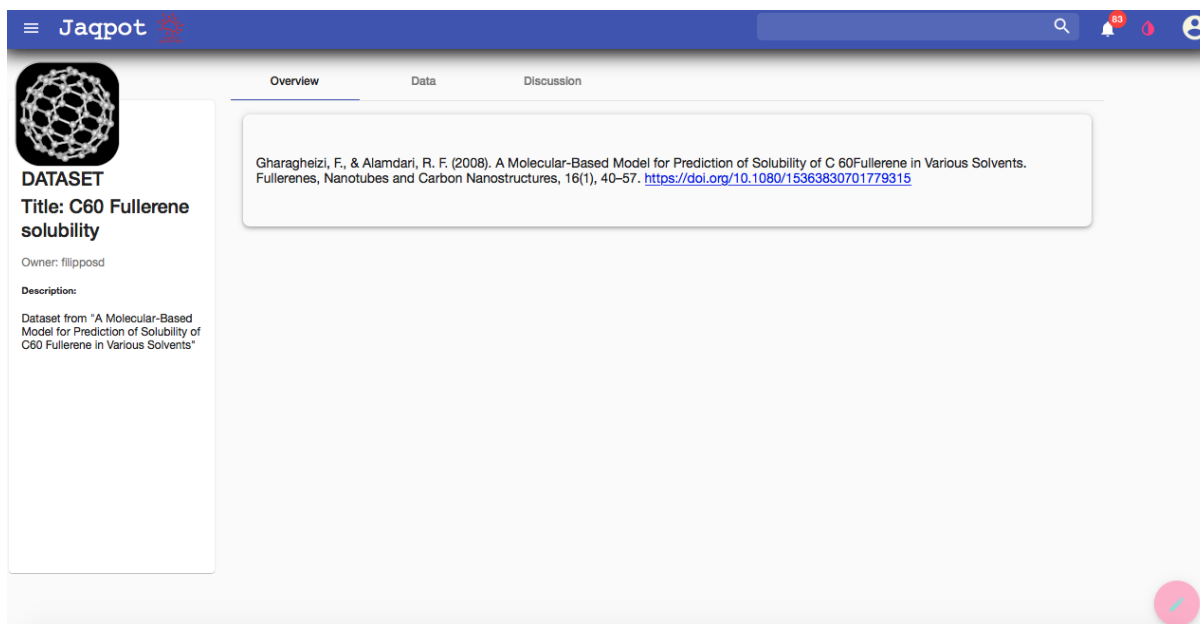


Figure 48. Adding a dataset - Successful upload window (Jaqpot 5).

The dataset uploaded is available at <https://app.jaqpot.org/dataset/Pf5IRaqASmXVOxzTuFOMJm> and its homepage is shown in Figure 49, where we have added the publication information. The *Data* tab shows the data as it has been imported (Figure 50) and the *Discussion* tab provides a space for discussions on the dataset, similarly to the *Discussion* tab in models.

Please note the fullerene picture in the dataset page. This has been added after the upload of the model, by clicking the generic grey icon in that position and uploading a relevant image (Figure 51).

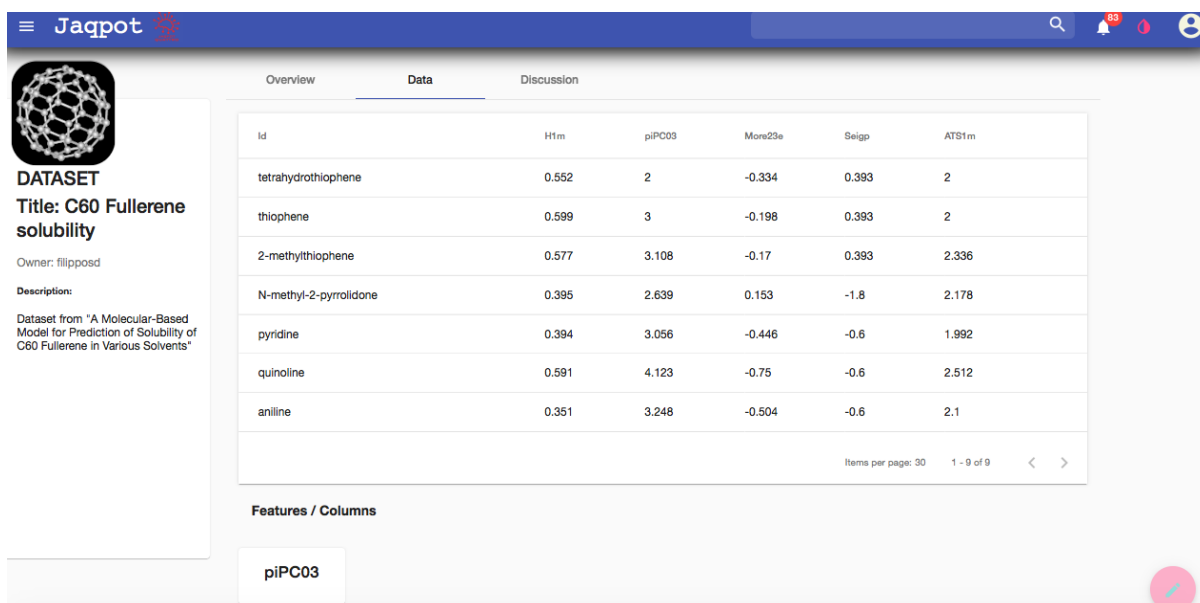


**Jaqpot**

**DATASET**  
**Title: C60 Fullerene solubility**  
 Owner: filipposd  
 Description:  
 Dataset from "A Molecular-Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents"

Gharagheizi, F., & Alamdari, R. F. (2008). A Molecular-Based Model for Prediction of Solubility of C 60Fullerene in Various Solvents. Fullerenes, Nanotubes and Carbon Nanostructures, 16(1), 40-57. <https://doi.org/10.1080/15363830701779315>

Figure 49. C60 Fullerenes solubility dataset - Homepage (Jaqpot 5).



**Jaqpot**

**DATASET**  
**Title: C60 Fullerene solubility**  
 Owner: filipposd  
 Description:  
 Dataset from "A Molecular-Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents"

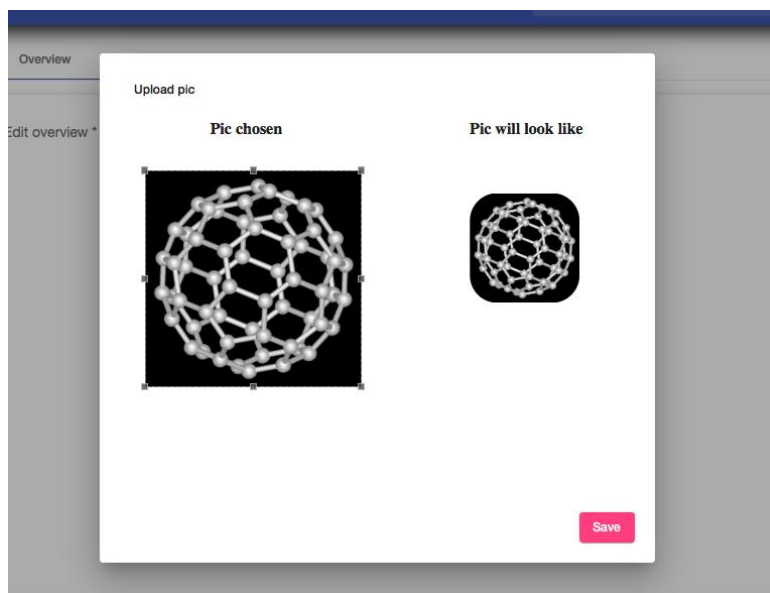
Id	H1m	piPC03	More23e	Seigp	ATStm
tetrahydrothiophene	0.552	2	-0.334	0.393	2
thiophene	0.599	3	-0.198	0.393	2
2-methylthiophene	0.577	3.108	-0.17	0.393	2.336
N-methyl-2-pyrrolidone	0.395	2.639	0.153	-1.8	2.178
pyridine	0.394	3.056	-0.446	-0.6	1.992
quinoline	0.591	4.123	-0.75	-0.6	2.512
aniline	0.351	3.248	-0.504	-0.6	2.1

Items per page: 30 1 - 9 of 9

**Features / Columns**

piPC03

Figure 50. C60 Fullerenes solubility dataset (Jaqpot 5).



**Figure 51.** Adding a picture to a dataset - Upload and preview prompt (Jaqpot 5).

#### Sharing a model or dataset

In order to specify the organisations with which to share the dataset, the user has to enter the editing mode by clicking the bottom right icon and selecting the respective organisations for each functionality:

- *Read*: this first level of access only allows viewing the dataset (or model)
- *Execute*: this second level of access allows viewing and executing a model on the dataset (or equivalently for models, allows model execution)
- *Write*: this third level of access allows users to additionally modify resources. Please note that a model or dataset can only be deleted by its creator.

A sensible policy would be for a user to share *Write* level access within an organisation represented by her/his lab or project and provide *Execute* rights to a wider community.

In our example, we provide *Write* access to collaborators of the NanoCommons organisation (Figure 52). After clicking the blue icon with the arrow, a “Successfully shared!” message appears above the icon.

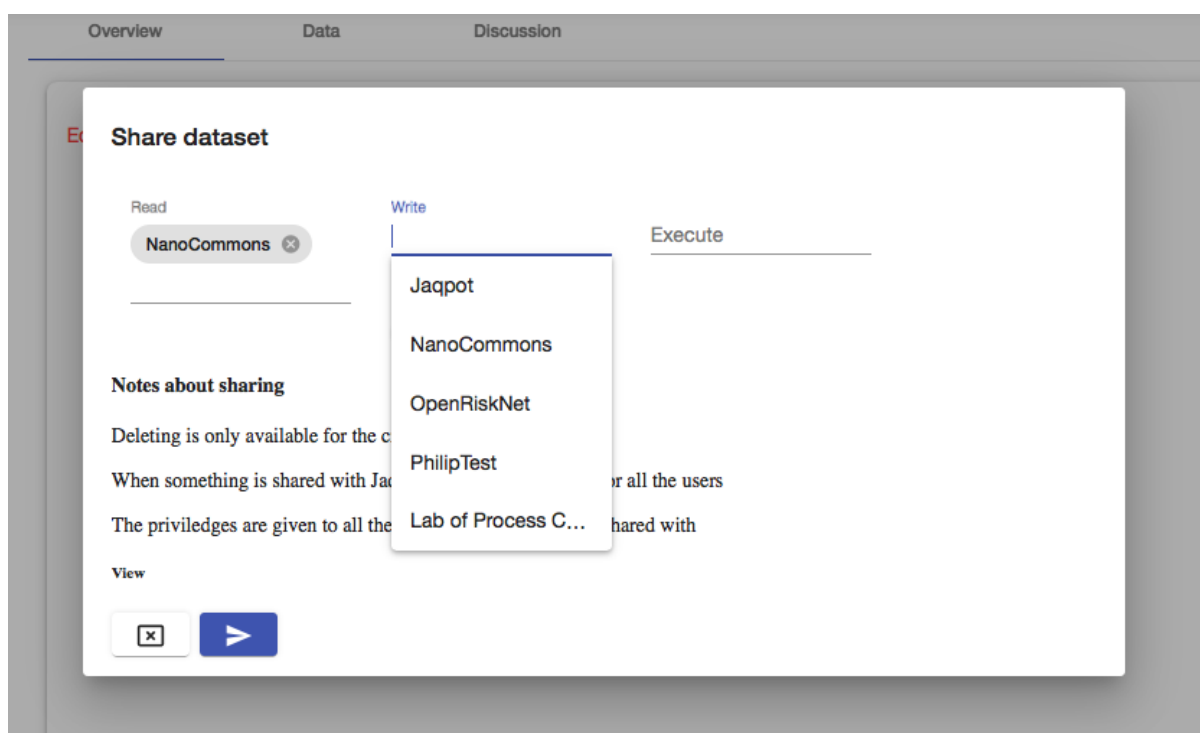


Figure 52. Specifying access levels to organisations (Jaqpot 5).

## NanoCommons repository of nanoQSAR models

In the previous sections, it was illustrated that NanoCommons offers all the necessary tools for developing, storing, validating and sharing nanoQSAR models. Therefore, the NanoCommons infrastructure can provide a central repository of well validated nanoQSAR models that can be freely shared among the community as easy and ready-to-use web applications. To this end, we initiated the development of this central repository of nanoQSAR models, by integrating a number of nanoQSAR models that have been presented in the literature in both Jaqpot 4 and Jaqpot 5. The models were selected after an investigation through the scientific literature. A variety of recommended reviews of nanoQSAR models have been published during the last few years (Burello (2017), Chen et al. (2017), Oksel et al. (2015a, 2015b, 2017), Tantra et al. (2015), Winkler (2016), Worth et al. (2017)). An extensive review of the current status of nanoQSAR models was conducted recently by the EU Joint Research Centre (JRC) in the context of the NanoComput project and resulted in a comprehensive Excel workbook containing a full list of nanoQSAR models (Worth et al. (2017), Lamon et al. (2019)).

Models included in the JRC catalogue cover a total of 44 different MNs, including metals, metal oxides and carbon, polymeric and lipid-based particles. The JRC catalogue was our main source of information and provided a pool of nanoQSAR models from which a few were selected for populating the first NanoCommons repository of nanoQSAR models. The most important criterion for the

selection of nanoQSAR models, was the availability of the data that allowed the development of the models. The library will be updated regularly, so that it will always be up-to-date. Table 1 shows the nanoQSAR models that have been included so far in the NanoCommons infrastructure and are available through both Jaqpot 4 and Jaqpot 5. More details about these models can be found in Appendix 6.

Login instructions to Jaqpot 4 and Jaqpot 5 are repeated next for convenience:

To access and use a Jaqpot 4 model, a user should click on the homepage: [www.jaqpot.org](http://www.jaqpot.org), login with the NanoCommons account (username: *NanoCommons*, password: *NanoCommons*) and paste the corresponding URL shown in Table 1 in the address line of the browser.

To access and use a Jaqpot 5 model, a user should click on the homepage: <https://app.jaqpot.org/>, login with his own account and send an invitation request to the NanoCommons organisation administrator ([hsarimv@central.ntua.gr](mailto:hsarimv@central.ntua.gr)). After being accepted as a member, the user can access the model by selecting it through the catalogue of models under the NanoCommons organisation or by just pasting the corresponding url shown in Table 1 in the address line of the browser. Alternatively, the user can access the model by entering with the guest account (username: *guest*, password: *guest*) and pasting the corresponding link, which however does not allow running the models.

**Table 1. Examples of models in Jaqpot 4 & Jaqpot 5 with basic information**

	Model name	Endpoint	Model URIs
1	Methodology for developing structure-activity evaluation to identify combinations of physical features of NMsl that influence potential cell damage by MLR/Linear Discriminant Analysis (TiO <sub>2</sub> case) <b>Sayes &amp; Ivanov (2010)</b>	In vitro - Cytotoxicity - membrane damage measured as lactate dehydrogenase (LDH) release [units/L]	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=2KoKHclgMJloSeWuZ03a">http://jaqpot.org/m_detail?name=2KoKHclgMJloSeWuZ03a</a> Jaqpot 5 <a href="https://app.jaqpot.org/model/izMnmc5LMbgC6o7Fkj8">https://app.jaqpot.org/model/izMnmc5LMbgC6o7Fkj8</a>
2	Methodology for developing structure-activity evaluation to identify combinations of physical features of nanomaterial that influence potential cell damage by MLR/LDA (ZnO case) <b>Sayes &amp; Ivanov (2010)</b>	In vitro - Cytotoxicity - membrane damage measured as lactate dehydrogenase (LDH) release [units/L]	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=cm49KGhUjkMw6wyntKQF">http://jaqpot.org/m_detail?name=cm49KGhUjkMw6wyntKQF</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/fvAe4KnlOOiNgGf7Ve1p">https://app.jaqpot.org/model/fvAe4KnlOOiNgGf7Ve1p</a>
3	Regression model to understand the aggregated zero valent copper nanoparticles against E. Coli by MLR (Plackett-Burman design) <b>Rispoli et al. (2010)</b>	In vitro - Cytotoxicity - measured as percentage of dead E. Coli population	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=48JYATz0KFTkZjGd8AfS">http://jaqpot.org/m_detail?name=48JYATz0KFTkZjGd8AfS</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/8su6n4fclJpzZD2NDZGN">https://app.jaqpot.org/model/8su6n4fclJpzZD2NDZGN</a>
4	Prediction of the Biological surface adsorption index on different NMs by MLR <b>Xia et al. (2011)</b>	log(k) k: adsorption coefficient	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=DjRQk8AqG42nckg5KoxZ">http://jaqpot.org/m_detail?name=DjRQk8AqG42nckg5KoxZ</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/gSvjUZ17EEAV5OWL7Uls">https://app.jaqpot.org/model/gSvjUZ17EEAV5OWL7Uls</a>

5	Predictive model of TiO <sub>2</sub> NPs damage on cell membrane by SMILES-based optimal descriptor and Monte Carlo technique (CORAL software) <b>Toropova &amp; Toropov (2013)</b>	In vitro - Cytotoxicity - membrane damage measured as lactate dehydrogenase (LDH) release [units/L]	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=4oxlwXBZMJ4suYFTSI4d">http://jaqpot.org/m_detail?name=4oxlwXBZMJ4suYFTSI4d</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/nTJgb4Ss3zHIYZEcbg78">https://app.jaqpot.org/model/nTJgb4Ss3zHIYZEcbg78</a>
6	Cytotoxicity of metal oxide to bacteria E. Coli models by Periodic table-based descriptors and stepwise-MLR <b>Kar et al. (2014)</b>	In vitro - Cytotoxicity - measured as pEC50	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=EFFfILYKMgLUq3qNYBw">http://jaqpot.org/m_detail?name=EFFfILYKMgLUq3qNYBw</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/QgRRwyU8r7e0NubEuDdX">https://app.jaqpot.org/model/QgRRwyU8r7e0NubEuDdX</a>
7	Photo-induced toxicity of metal oxide NMs to E. coli by MLR (dark condition case) <b>Pathakoti et al. (2014)</b>	In vitro - Cytotoxicity - measured as - log(LC50)	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=KIWUeelVM8x7x1iC7cXi">http://jaqpot.org/m_detail?name=KIWUeelVM8x7x1iC7cXi</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/hygpzrH71XS1Wr8IGS69">https://app.jaqpot.org/model/hygpzrH71XS1Wr8IGS69</a>
8	Photo-induced toxicity of metal oxide NMs to E. Coli by MLR (Photo-induced (light) case) <b>Pathakoti et al. (2014)</b>	In vitro - Cytotoxicity - measured as - log(LC50)	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=o6Jr81BfQtUddgmwqae">http://jaqpot.org/m_detail?name=o6Jr81BfQtUddgmwqae</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/5gCY316DzDh1Fdw4aigo">https://app.jaqpot.org/model/5gCY316DzDh1Fdw4aigo</a>
9	Predicting metal oxide NMs toxicity to E. Coli cell line by MLR <b>Mu et al. (2016)</b>	In vitro - Cytotoxicity - measured as log(1/EC50)	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=qu6HILHSypXWX8zvMQ3">http://jaqpot.org/m_detail?name=qu6HILHSypXWX8zvMQ3</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/OAiBYue5PLJ7F580f2J">https://app.jaqpot.org/model/OAiBYue5PLJ7F580f2J</a>
10	Predicting C60 solubility in organic solvents by SMILES-based optimal descriptor and Monte Carlo technique <b>Gharagheizi &amp; Alamdari (2008)</b>	Solubility in organic solvents	Jaqpot 4: <a href="http://jaqpot.org/m_detail?name=sCoqY3D3xCpSuyS6RdoQ">http://jaqpot.org/m_detail?name=sCoqY3D3xCpSuyS6RdoQ</a> Jaqpot 5: <a href="https://app.jaqpot.org/model/VRp8f6A4DuJc8fsavvpB">https://app.jaqpot.org/model/VRp8f6A4DuJc8fsavvpB</a>

**Abbreviations:** EC50: Half maximal effective concentration; MLR: Multiple Linear Regression; LC50: median lethal dose; LDA: Linear Discriminant Analysis; LDH: lactate dehydrogenase; pEC50: negative logarithm of the half maximal effective concentration.

## Integration of NanoCommons nanoQSAR modelling infrastructure and the knowledge base

The integration of the NanoCommons Knowledge Base (KB) and the nanoQSAR modelling infrastructure was demonstrated through a Jupyter notebook (provided in Appendix 7). The example dataset used was presented in the NanoCommons hackathon in Athens in October 2018.

The procedure followed was:

### 1. Communication with the NanoCommons KB

Making the call to the NanoCommons KB, receiving the full dataset and pulling the values needed from the schema.

### 2. Preprocessing

Involves going from the semantically annotated dataset, to having the specific input variables and the prediction endpoint properly formatted for modelling, in our case classification.

### 3. Modelling

Logistic Regression with cross-validation was used, more specifically the *LogisticRegressionCV* algorithm from scikit-learn. The classification accuracy achieved was 0.8 and the confusion matrices before and after normalisation were plotted (Figures 53 and 54 respectively).

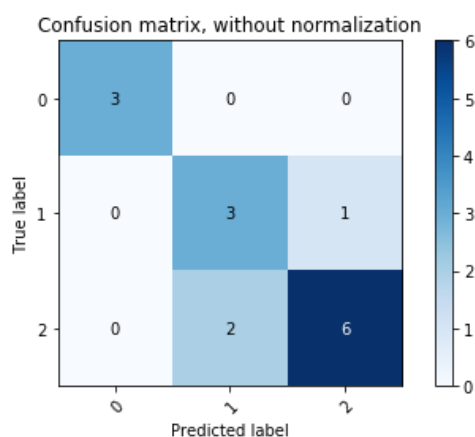


Figure 53. Confusion matrix without normalisation.



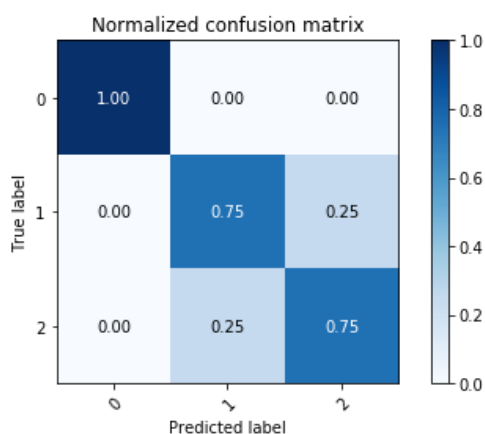


Figure 54. Normalised Confusion matrix.

#### 4. Publishing a local model into a web service with Jaqpot

A local model can be transformed into a web service that produces predictions using one simple Jaqpot command using the jaqpotpy package. The model that was produced in this example is accessible at <https://app.jaqpot.org/model/5mph5QzpkCeu3mfC2Gcm> (Figure 55) and has been shared within the NanoCommons organisation.

#### 5. Using the published model to get predictions

Once it becomes a web service, it can either be accessed through the user interface or be consumed as a web service from the python notebook and return predictions. We demonstrate such a call at the end of the notebook (Appendix 7).

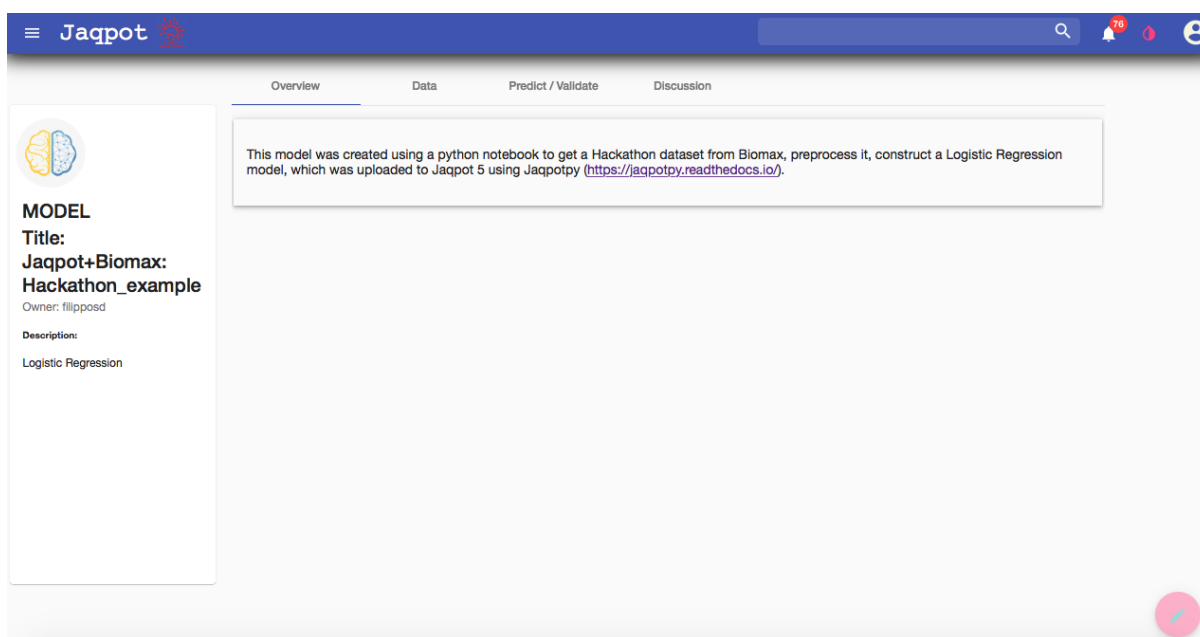


Figure 55. Webpage for model created in the integration example (Jaqpot 5).

---

## Enalos Cloud Platform

The Enalos Cloud Platform, developed by NovaMechanics Ltd, is an online, freely available cheminformatics and nanoinformatics cloud platform that hosts predictive models released as web services. These web services aim to address the need to reduce the amount of time and cost spent on experimental testing during the drug discovery and the risk assessment procedures for small molecules and NMs. Several predictive models, based on open source and in-house algorithms and software, are already available within the Enalos Cloud platform, including models for NMs toxicity, biological activity and properties evaluation.

Through the Enalos Cloud Platform a variety of state of the art modelling tools for the hazard prediction and risk assessment are available through a user-friendly environment, especially designed for non-informatics experts. The platform can be easily accessed, there is no need for authorization and the user can directly use the provided services. Under NanoCommons two web services were developed during the first 18 months of the project for generating and validating nanoQSAR models; the “Enalos QNAR Iron Oxide Toxicity Platform” and the “Enalos Cloud NanoInformatics Platform: A Safe-by-Design Tool for Functionalised Nanomaterials”, which are presented in the following paragraphs. The advantage of these web services is the possibility to produce within seconds reliable predictions over the toxicity of NMs and can host multiple users simultaneously.

### Enalos QNAR Iron Oxide Toxicity Platform

Online toxicity predictions for Iron Oxide NMs are made available through Enalos QNAR Iron Oxide Toxicity Platform. Enalos QNAR Iron Oxide Toxicity Platform hosts a fully validated predictive model (Melagraki et al. (2015)) which generates toxicity predictions based on a set of indicated properties.

The web service provides the functionality to virtually screen a set of NMs of interest based on the validated model, and thus yielding a preliminary *in silico* testing. The Enalos Cloud Platform for NMs aspires to act as a useful aid within a virtual screening framework, for the design of novel NMs or the prioritization of novel potent NMs based on their predicted toxic effect.

The tool can be easily accessed through the link:

[http://enaloscloud.novamechanics.com/EnalosWebApps/QNAR\\_IronOxide\\_Toxicity/](http://enaloscloud.novamechanics.com/EnalosWebApps/QNAR_IronOxide_Toxicity/)

#### Required input

To initiate a prediction, the user must provide values for the four following parameters; the NP size [nm], the zeta potential (ZP) [mV] and the relaxivities R1 and R2 [ $\text{mM}^{-1}\text{s}^{-1}$ ]. In addition, the user must indicate the coating among the three alternatives of cross-linked dextran, poly(vinyl alcohol) (PVA) and other. The user has two alternative ways to provide the above information - either by completing the provided form (Figure 56A) or by importing a .csv file (Figure 56B).

## Enalos QNAR Iron Oxide Toxicity Platform

[User Guide](#)

**A**

MNP Numbr	Size (nM)	ZP (mV)	R1 (mM-1S-1)	R2 (mM-1S-1)	Coating
1	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
2	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
3	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
4	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
5	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
6	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
7	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
8	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
9	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
10	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
11	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
12	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
13	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
14	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
15	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
16	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
17	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
18	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
19	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>
20	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Other <input type="text"/>

**D**  **E**

**B**  CSV title

**C**

**D**  **E**

**Figure 56.** Enalos QNAR Iron Oxide Toxicity Platform. At the top page the input form can be seen and at the bottom by clicking on Upload csv file button, the user can import a .csv file with all the required properties.

### Manual entry

For only one entry: Each row corresponds to one NP. The user provides values for the four parameters (Size, ZP, R1, R2) by entering a value in the corresponding field of the first row and in addition, selects from the drop-down menu a coating among the alternatives (cross-linked dextran/PVA/other).

For multiple entries: Different rows correspond to multiple different NPs. The user provides information as above and now fills in data for multiple rows each corresponding to one NP. When directly submitting data through the online form, multiple entries are available for up to 20 NPs at a time.

### File entry

For one NP or for a set of NPs, all necessary information can be submitted via a .csv file by clicking the **Upload csv file** button. The file must have a specific format (Figure 57) as described below. A template file is also available to download (Figure 56C).

Rows: In the first row the following column names should be included in the specific order: "row ID", "Size", "ZP", "R1", "R2" and "Coating\_new".

Columns: Data should be included in each of the six columns for as many rows as the number of NPs as

shown in Figure 57. Columns (2) – (5) contain numerical fields whereas Column (6) contains text. Column (1): An id is given by the user for their own internal use and identification of the NP when results are generated, Column (2): Value for the NP size is given in nm, Column (3): Value for the ZP is given in mV, Columns (4) and (5): Values for R1 and R2 are given in  $\text{mM}^{-1}\text{S}^{-1}$  respectively, Column (6): A text is provided by the user among three options: “cross-linked dextran”, “PVA” and “other”.

	A	B	C	D	E	F	G
1	row ID	Size	ZP	R1	R2	Coating_new	
2	1	36	-19.9	19	45	cross-linked dextran	
3	2	30	-9.22	26	74	cross-linked dextran	
4	3	29	1.95	22	51	cross-linked dextran	
5	4	38	-10.1	21	62	cross-linked dextran	
6	5	28	2.34	19	39	cross-linked dextran	
7	6	38	-9.34	21	62	cross-linked dextran	
8	7	40	-3.77	15	30	PVA	
9	8	20	-4.3	0.5	0.5	Other	
10	9	20	-6.54	0.5	0.5	PVA	
11	10	20	-7.7	0.5	0.5	PVA	
12	11	20	-6.75	0.5	0.5	PVA	
13	12	28	-9.23	32	60	Other	
14	13	25	-37	29	49	Other	

**Figure 57.** Required format of the .csv file with a sample of input data.

By clicking the **Execute computation** button found under the online form or under the imported .csv file (Figure 56D), depending on the method chosen to provide the input data, predictions for each NP are generated.

#### Outputs-results

As described above, when properties are uploaded for a set of NMs, a prediction is generated by submitting the input values, the predictive model is then applied to the data provided and the output is generated within seconds and presented as a Table (Figure 58).

The “Prediction” column contains the toxicity prediction. The toxicity prediction is given by assigning a class to each of the submitted NPs. The class “active” and the class “inactive” are the two options that indicate a possible toxic or non-toxic effect respectively, for a given submitted NM, based on the predictive model.

The “Domain” column contains an indication of the reliability of predictions based on the model’s DoA limits. Two options are available: The “reliable” option which indicates a prediction within the DoA of the model and the “unreliable” option which is a warning for a prediction out of the DoA of the predictive model.

Download files

Row ID	Prediction	Domain
"Row1"	"inactive"	"reliable"
"Row2"	"inactive"	"reliable"
"Row3"	"inactive"	"reliable"
"Row4"	"inactive"	"reliable"
"Row5"	"inactive"	"reliable"
"Row6"	"inactive"	"reliable"
"Row7"	"active"	"reliable"
"Row8"	"active"	"reliable"
"Row9"	"active"	"reliable"
"Row10"	"active"	"reliable"
"Row11"	"active"	"reliable"
"Row12"	"inactive"	"reliable"
"Row13"	"inactive"	"reliable"

**Figure 58.** Generated output page. The first column of the results table contains the prediction for each input NM and the second column contains the reliability of each prediction based on the model's DoA. This table can also be downloaded in .csv and .html format by clicking in the corresponding button.

By clicking the **Download files** button, the above table is downloaded on both .csv and .html format.

If more NPs need to be submitted, the user can return to the initial page and by clicking on the **Reset** button next to **Execute computation** button (Figure 56E) the data are cleared, and new input data can be submitted.

A step by step tutorial is available at:

[http://enaloscloud.novamechanics.com/EnalosWebApps/QNAR\\_IronOxide\\_Toxicity/instructions.zul](http://enaloscloud.novamechanics.com/EnalosWebApps/QNAR_IronOxide_Toxicity/instructions.zul)



NovaMechanics

[www.novamechanics.com](http://www.novamechanics.com)

---

# Enalos QNAR Iron Oxide Toxicity Platform

---

*A brief tutorial*

NovaMechanics Ltd

[info@novamechanics.com](mailto:info@novamechanics.com)

Version, February 2019

## Overview

Online toxicity predictions for Iron Oxide nanoparticles (NPs) are made available through Enalos QNAR Iron Oxide Toxicity Platform. Enalos QNAR Iron Oxide Toxicity Platform hosts a fully validated predictive model developed by NovaMechanics Ltd which generates toxicity predictions based on a set of indicated properties.

The platform can be easily accessed, there is no need for authorization and the user can directly use the provided service. The web service provides the functionality to virtually screen a set of NPs of interest based on the validated model, and thus yielding a preliminary *in silico* testing. Enalos Cloud Platform for NPs aspires to act as a useful aid within a virtual screening framework, for the design of novel NPs or the prioritization of novel potent NPs based on their predicted toxic effect.

The tool can be easily accessed through the link:

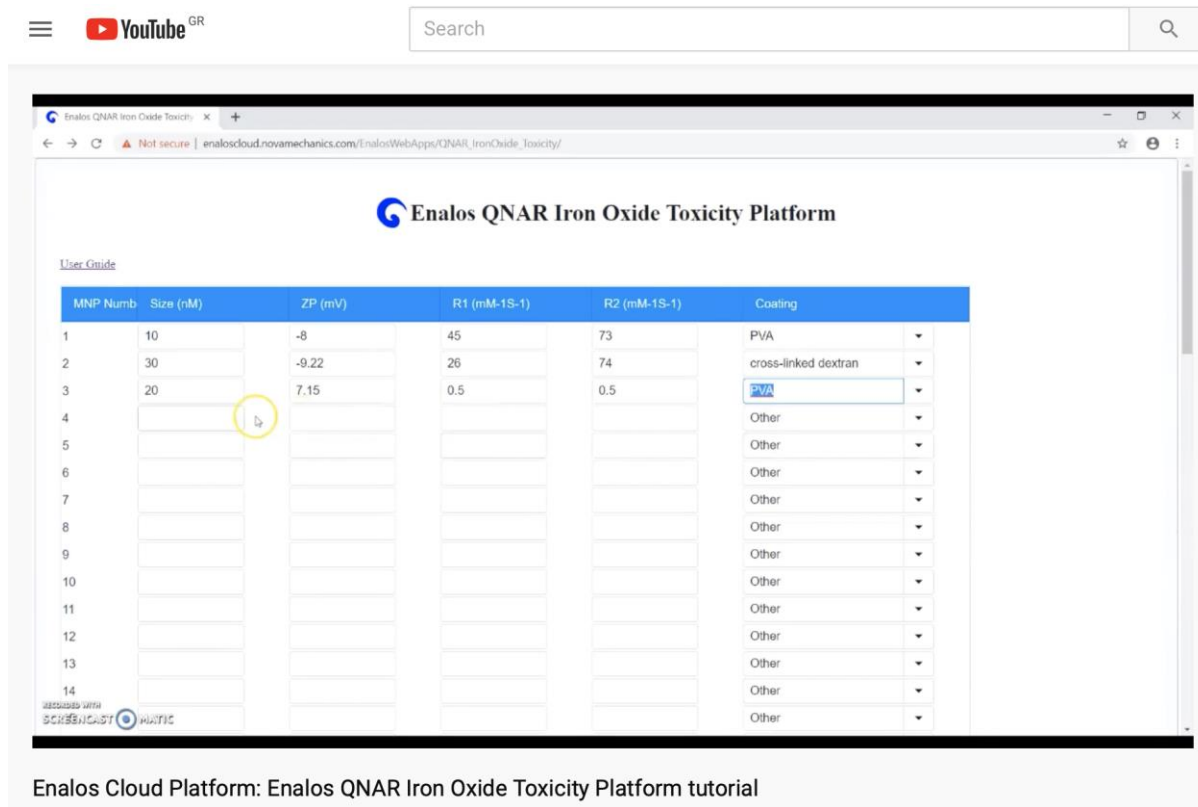
[http://enaloscloud.novamechanics.com/EnalosWebApps/QNAR\\_IronOxide\\_Toxicity/](http://enaloscloud.novamechanics.com/EnalosWebApps/QNAR_IronOxide_Toxicity/)

Full documentation on the model development can be found in: G. Melagraki and A. Afantitis, *A Risk Assessment Tool for the Virtual Screening of Metal Oxide Nanoparticles through Enalos InSilicoNano Platform*, Current Topics in Medicinal Chemistry, vol. 15, no. 18, pp. 1827-36, 2015.

**Figure 59.** Detailed tutorial for Enalos QNAR Iron Oxide Toxicity Platform

A demonstration video is available at:

<https://www.youtube.com/watch?v=HtUg1EXLr28&list=PL333udxtaNNZeooAwcsqFcXwMrEWU5mMY&index=2>



The screenshot shows a YouTube video player displaying a web application titled "Enalos QNAR Iron Oxide Toxicity Platform". The application interface includes a search bar at the top and a table with the following columns: MNP Num, Size (nm), ZP (mV), R1 (mM-1S-1), R2 (mM-1S-1), and Coating. The table contains 14 rows of data. The Coating column has a dropdown menu with options: PVA, cross-linked dextran, PVA (highlighted), Other, Other, Other, Other, Other, Other, Other, Other, Other, Other, and Other. A yellow circle highlights the "PVA" option in the dropdown menu.

MNP Num	Size (nm)	ZP (mV)	R1 (mM-1S-1)	R2 (mM-1S-1)	Coating
1	10	-8	45	73	PVA
2	30	-9.22	26	74	cross-linked dextran
3	20	7.15	0.5	0.5	PVA
4					Other
5					Other
6					Other
7					Other
8					Other
9					Other
10					Other
11					Other
12					Other
13					Other
14					Other

Enalos Cloud Platform: Enalos QNAR Iron Oxide Toxicity Platform tutorial

**Figure 60.** YouTube video showing the functionalities of the Enalos QNAR Iron Oxide Toxicity Platform.

### Enalos Cloud NanoInformatics Platform: A Safe-by-Design Tool for Functionalised NMs

Online toxicity and protein binding of carbonic anhydrase (CA) predictions for decorated multi-walled carbon nanotubes (MWCNTs) are made available through Enalos Cloud Platform, which hosts a fully validated predictive model based only on the structure of the decorating molecules (Varsou, D.-D. et al. (2019), Varsou et al. (2018)). During a safe-by-design process, different data sets with decorators of interest can be imported, and their effects on the biological and toxicity behavior of the resulting decorated MWCNTs can be studied.

The web-service can be easily accessed through the link:

<http://enaloscloud.novamechanics.com/EnalosWebApps/CNT/>

Required input

To initiate a prediction, the user must provide one or several structures of compounds being considered as potential decorating molecules for MWCNTs (Figure 59) and get, within seconds, the prediction of the

CA binding and their toxicity profile, along with a warning on the reliability of the predictions based on the models' DoA limits. For this purpose, the platform provides three different options for inserting the required data to the model (Figure 60).

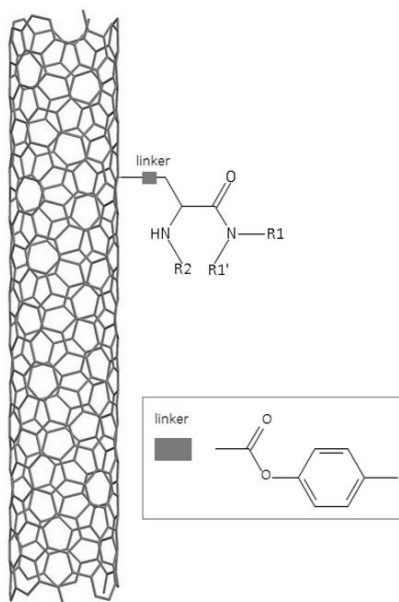


Figure 61. Core MWCNT and decorating molecule structure and position.

### Enalos Nanoinformatics Cloud Platform: A Safe-by-Design Tool for Functionalised Nanomaterials

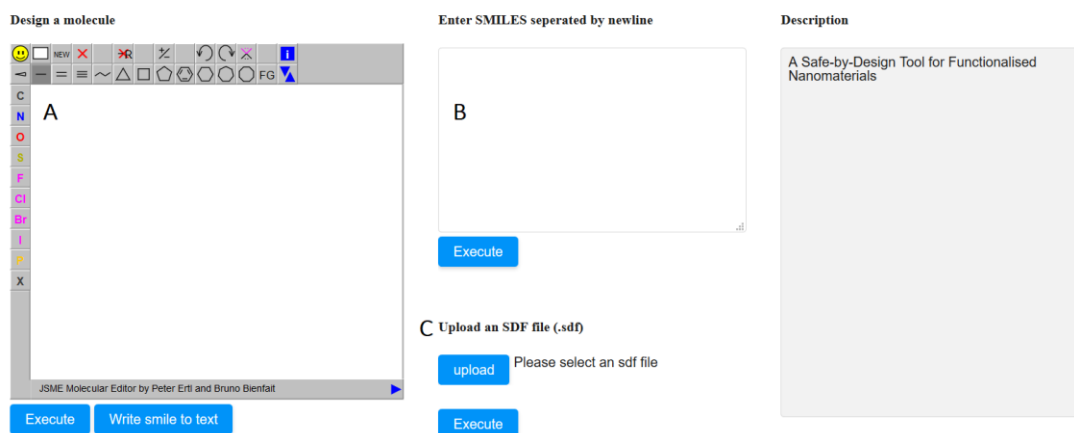
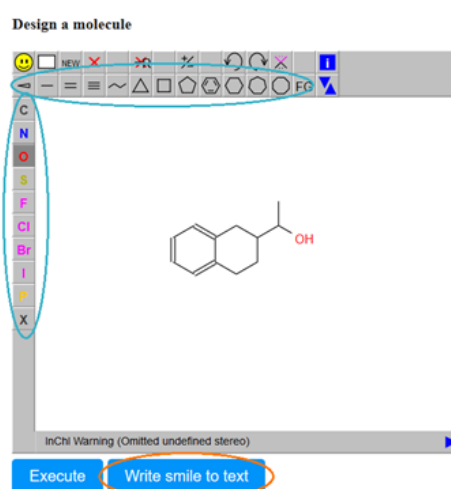


Figure 62. The Enalos Nanoinformatics Cloud platform user-friendly interface. [A] At the left-handed side the molecular drawing tool is found. [B] At the top right-handed the SMILES input form can be seen followed by [C] the option of importing a .sdf file.

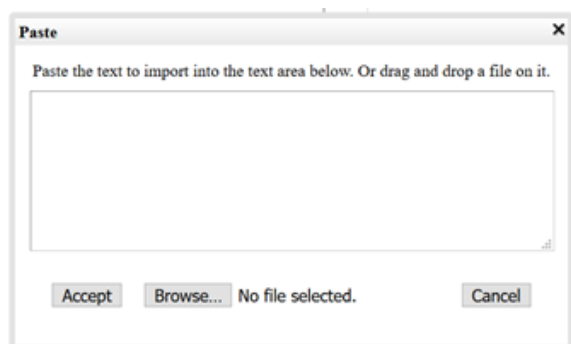


### Manual entry

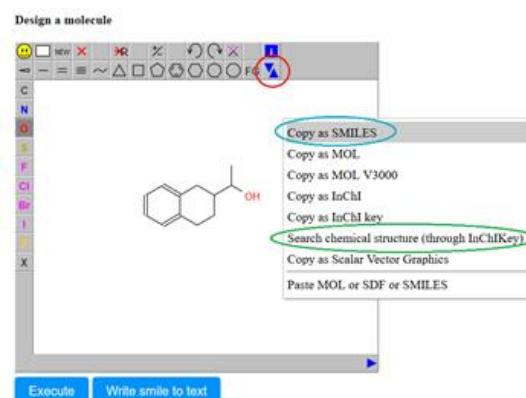
For only one entry: The user can draw the molecular structure of interest using the drawing tool (Figure 60A). When using the drawing tool, only one structure can be submitted at a time. The user can easily select from the different panels' specific atoms, bonds or substructures and then construct the decorating molecule (Figure 61, see blue circles). The functionality also enables the user to open, save and convert files with a variety of chemical formats such as, SMILES, IUPAC Chemical Identifier, MDL MOL file (Figure 62) using the drop-down menu (Figure 63, see red circle) of the online sketcher. It is also possible to search via InChIKey identification, information about the molecular structure of interest (Figure 63, see green circle).



**Figure 63.** Molecular drawing tool. The user can design the structure of a compound using the different substructures at the top and left ribbons.



**Figure 64.** By clicking on “Paste any chemical text repr.” a new window opens, and the user can enter either manually either by uploading structures in several chemical formats.



**Figure 65.** With the right mouse click or by clicking on the drop-down menu, the user can copy the structure in different formats (useful for the generation of several structures).

For multiple entries: The user can enter the SMILES notation of one or several structures separated by newlines (Figure 62B).

Even if the SMILES notation is not initially known, the chemical sketcher included gives the users the opportunity to draw the molecular structure and then copy the structure as SMILES by right mouse click or by using the drop-down menu (Figure 61, see orange circle or Figure 63, blue circle). This facilitates the generation of several structures, by allowing a multitude of modifications to be performed using the sketcher and then coping all structures as SMILES and pasting in the appropriate field (Figure 66), so that a prediction for the whole set of produced structures is obtained. Thus, the modifications can be visualized, and multiple predictions can be realized at once.



**Figure 66.** Inserting structures in SMILES identification format. Each SMILE must be separated by newline.

## File entry

The user can select and import a .sdf file with several structures, by clicking the **Upload** button (Figure 60C).

This type of files contains molecular structure records, used as a standard exchange format for chemicals information. The decorating molecule's structure in .sdf format can be extracted from the PubChem database or other repositories.

When structures are uploaded in either way, a prediction is generated (usually within seconds) by clicking the **Execute** button of the corresponding field.

## Outputs-results

As described above, when properties are uploaded for a set of decorating molecules, a prediction is generated by submitting the input structures, the predictive model is then applied to the data provided and the output is generated.

[Download files](#)

**Safe-by-Design:  
Functionalised Nanomaterials** *Knime report powered by Birt*

"Toxicity"	"Domain (Toxicity)"	"Activity"	"Domain (Activity)"
toxic	reliable	non-binder	reliable
non-toxic	reliable	binder	reliable
toxic	reliable	non-binder	reliable
toxic	unreliable	binder	reliable

Date: Apr 2, 2019 5:43 PM      Author: NovaMechanics Ltd      1 of 1  
www.knime.com

**Figure 67.** Results table. The first and the third columns contain the prediction for each input sample and the second and fourth columns contain the reliability of the corresponding prediction according to the DoA.

The results include the predicted CA binding class (“binder”/“non-binder”) to the MWCNTs and the toxicity class (“toxic”/“non-toxic”) of the resultant decorated-MWCNTs for each structure entered, and an indication of whether this prediction could be considered reliable based on the DoA of the model (Figure 65). Two options are available: The “reliable” option which indicates a prediction within the DoA limits of the model and the “unreliable” option which is a warning for a prediction out of the DoA of the predictive model.

By clicking the **Download files** button, the above table is downloaded on both .csv and .html format. In the produced .csv files, the interested users can observe the neighbours of the training set used for the prediction of each one of the input samples (Figure 66).

	A	B	C	D	E	F	G	H	I
1	row ID	Activity	Neighbor 0	Distance of Neighbor 0	Neighbor 1	Distance of Neighbor 1	Neighbor 2	Distance of Neighbor 2	Domain (Activity)
2	Row0	non-binder	c1cccc(c1)C(Oc1cc	0.182235425	c1cccc(c1)C(Oc1cc	0.347628682	c1cccc(c1)C(Oc	0.360303746	reliable
3	Row1	binder	c1cccc(c1)C(Oc1cc	0.125957089	c1cccc(c1)C(Oc1cc	0.130089531	c1cccc(c1)C(Oc	0.144106101	reliable
4	Row2	non-binder	c1cccc(c1)C(Oc1cc	0.182235425	c1cccc(c1)C(Oc1cc	0.347628682	c1cccc(c1)C(Oc	0.360303746	reliable
5	Row3	binder	c1cccc(c1)C(Oc1cc	0.112993142	c1cccc(c1)C(Oc1cc	0.123445261	c1cccc(c1)C(Oc	0.154328932	reliable

**Figure 68.** Example of the output file containing the neighbors in the training set used for the activity prediction of each input sample.

A step by step tutorial is available at:

<http://enaloscloud.novamechanics.com/EnalosWebApps/CNT/instructions.zul>



The European Nanotechnology Community Informatics Platform: Bridging data and disciplinary gaps for industry and regulators

## Enalos Nanoinformatics Cloud Platform: A Safe-by-Design Tool for Functionalized Nanomaterials

*A brief tutorial*

NovaMechanics Ltd

info@novamechanics.com

Version, April 2019

### Overview

Online toxicity and protein binding of carbonic anhydrase (CA) predictions for decorated multi-walled carbon nanotubes (MWCNTs) are made available through Enalos Cloud Platform, which hosts a fully validated predictive model developed by NovaMechanics Ltd based only on the structure of the decorating molecules.

The user-friendly web service will facilitate the computer-aided design of novel MWCNTs by the interested users (computational experts or not); the Enalos Cloud platform can be easily accessed and can be directly explored by anyone interested in MWCNTs design to optimise functionality and safety (i.e. safe-by-design), without any need for prior programming skills. During a safe-by-design process, different data sets with decorators of interest can be imported, and their effects on the biological and toxicity behavior of the resulting decorated MWCNTs can be studied.

The web-service can be easily accessed through the link:

<http://enaloscloud.novamechanics.com/EnalosWebApps/CNT/>



This project has received funding from European Union Horizon 2020 Programme (H2020) under grant agreement n° 731032.

**Figure 69.** Detailed tutorial for the Safe-by-Design Tool for Functionalised Nanomaterials powered by Enalos Nanoinformatics Cloud Platform

A demonstration video (Figure 70) is available at: <https://youtu.be/BLuG-LwuQ1E>.

Enalos Nanoinformatics Cloud Platform: A Safe-by-Design Tool for Functionalised Nanomaterials

Design a molecule

Enter SMILES separated by newline

Description

Execute

Execute

Execute

Upload an SDF file (.sdf)

upload Please select an sdf file

Execute

Enalos Nanoinformatics Cloud Platform: A Safe-by-Design Tool for Functionalized Nanomaterials

Figure 70. YouTube video showing the functionalities of the Safe-by-Design Tool for Functionalised NMs.

---

## Conclusions

The work described regarding the integration of the first predictive nanoQSAR models into the NanoCommons KnowledgeBase was structured around the two platforms offering nanoQSAR modelling within NanoCommons. The Jaqpot platform offers its functionality over an API and GUIs built on top of the API to facilitate users. At present, versions 4 and 5 coexist in complementarity to each other, but all modelling features will be integrated into the newer version (version 5). It should also be noted that Jaqpot follows the guidelines set by the OECD regarding QSAR modelling. Starting from an example in fullerene solubility modelling, Jaqpot 4 functions of training a model, making it available as a web service, validating the model and getting predictions on new values were demonstrated. In addition, a novel feature in Jaqpot 5 is the addition of social networking and sharing of resources through organisations. By making models or datasets available to a narrower or wider circle and providing a social networking space, Jaqpot simplifies getting feedback from the community on aspects of the model (*i.e.* prediction accuracy, applicability domain) or dataset (*i.e.* annotation of features, models that have been developed) in question. Starting with a first batch of example models that have been added to both Jaqpot 4 and Jaqpot 5 for users to test and review, the necessary elements of Jaqpot to serve as a central repository for nanoQSAR models that is both open and accessible with low technical proficiency have been laid out. Additionally, an example of integration of the NanoCommons KB and nanoQSAR modelling infrastructure was presented, built around a python notebook and leading to the creation of a Jaqpot model web service that can be used for predictions through the notebook. Moreover, two novel nanoQSAR models for Iron Oxide NPs and MWCNTs were also developed using the Enalos Cloud platform. Firstly, a fully validated workflow for the prediction of both the binding of carbonic anhydrase (CA) to organic molecule functionalised MWCNTs and the toxicity of the functionalised MWCNTs has been developed and was disseminated as a user-friendly web service through the Enalos Cloud platform. This study was based on the open-source KNIME platform, combining KNIME and Enalos+ nodes, which facilitate the manipulation of big data, the modelling, the validation and the virtual screening processes. In addition, a fully validated and predictive QNAR model that can be used for the risk assessment of different metal oxide NPs was also developed. NovaM in-house-made Enalos+ KNIME nodes were integrated in our proposed workflow to perform crucial procedures such as the domain of applicability determination. The predictive model was hosted and published directly on the web through Enalos Cloud Platform allowing the researchers to do virtual screening and/or design novel NMs. Both models aspire to act as a useful aid within a virtual screening framework for the design of novel NMs or the prioritization of novel potent NMs based on their predicted toxic effect. Having built the nanoQSAR tools around community needs, and by providing an integrated capability for collaborative work via the platform, NanoCommons is both a fitting choice for TA users and an e-Infrastructure that will be enriched by the input from external users. Indeed, a specific call for model owners to integrate their nanoQSAR models into NanoCommons in early 2020.

## References

1. Atkinson, A. C. (1985). Plots, transformations and regression. Clarendon Press, Oxford. *Plots, transformations and regression*. Clarendon Press, Oxford.
2. Brandmaier, S., Sahlin, U., Tetko, I. V., & Öberg, T. (2012). PLS-optimal: a stepwise D-optimal design based on latent variables. *Journal of Chemical Information and Modeling*, 52(4), 975–983. <https://doi.org/10.1021/ci3000198>
3. Burello, E. (2017). Review of (Q) SAR models for regulatory assessment of nanomaterials risks. *NanoImpact*, 8, 48-58. <https://doi.org/10.1016/j.impact.2017.07.002>
4. Cassotti, M., Ballabio, D., Consonni, V., Mauri, A., Tetko, I. V., & Todeschini, R. (2014). Prediction of acute aquatic toxicity toward *Daphnia magna* by using the GA-kNN method. *Alternatives to Laboratory Animals: ATLA*, 42(1), 31–41. <https://doi.org/10.1177/026119291404200106>
5. Chen, G., Peijnenburg, W., Xiao, Y., & Vijver, M. (2017). Current knowledge on the use of computational toxicology in hazard assessment of metallic engineered nanomaterials. *International journal of molecular sciences*, 18(7), 1504. <https://doi.org/10.3390/ijms18071504>
6. Chomenidis, C., Drakakis, G., Tsiliki, G., Anagnostopoulou, E., Valsamis, A., Doganis, P., Sopsakis, P. & Sarimveis, H. (2017). Jaqpot Quattro: a novel computational web platform for modeling and analysis in nanoinformatics. *Journal of chemical information and modeling*, 57(9), 2161-2172. <https://doi.org/10.1021/acs.jcim.7b00223>
7. Chomenidis, C., Philip Doganis, George Drakakis, Georgia Tsiliki, Haralambos Sarimveis, Barry Hardy. (2015, July 31). Deliverable Report D4.3 nQSAR Modelling infrastructure. Zenodo. <http://doi.org/10.5281/zenodo.375610>
8. Farhad Gharagheizi & Reza Fareghi Alamdari (2008) A Molecular-Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents, Fullerenes, Nanotubes, and Carbon Nanostructures, 16:1, 40-57. <https://doi.org/10.1080/15363830701779315>
9. Fourches, D., Pu, D., Tassa, C., Weissleder, R., Shaw, S. Y., Mumper, R. J., & Tropsha, A. (2010). Quantitative nanostructure-activity relationship modeling. *ACS Nano*, 4(10), 5703–5712. <https://doi.org/10.1021/nn1013484>
10. Gajewicz, A., Schaeublin, N., Rasulev, B., Hussain, S., Leszczynska, D., Puzyn, T., & Leszczynski, J. (2015). Towards understanding mechanisms governing cytotoxicity of metal oxides nanoparticles: Hints from nano-QSAR studies. *Nanotoxicology*, 9(3), 313–325. <https://doi.org/10.3109/17435390.2014.930195>
11. Gharagheizi, F., & Alamdari, R. F. (2008). A Molecular-Based Model for Prediction of Solubility of C<sub>60</sub> Fullerene in Various Solvents. *Fullerenes, Nanotubes and Carbon Nanostructures*, 16(1), 40–57. <https://doi.org/10.1080/15363830701779315>
12. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1), 10-18. <https://doi.org/10.1145/1656274.1656278>
13. Hansen, S. F., Jensen, K. A., & Baun, A. (2013). NanoRiskCat: a conceptual tool for categorization and communication of exposure potentials and hazards of nanomaterials in consumer products. *Journal of Nanoparticle Research*, 16(1), 2195. <https://doi.org/10.1007/s11051-013-2195-z>
14. Kar, S., Gajewicz, A., Puzyn, T., Roy, K., & Leszczynski, J. (2014). Periodic table-based descriptors to encode cytotoxicity profile of metal oxide nanoparticles: A mechanistic QSTR approach. *Ecotoxicology and Environmental Safety*, 107, 162–169. <https://doi.org/10.1016/j.ecoenv.2014.05.026>
15. Lamon, L., Asturiol, D., Vilchez, A., Ruperez-Illescas, R., Cabellos, J., Richarz, A., & Worth, A. (2019). Computational models for the assessment of manufactured nanomaterials: Development of model reporting standards and mapping of the model landscape. *Computational Toxicology*, 9, 143–151. <https://doi.org/10.1016/j.comtox.2018.12.002>

16. Liu, R., Jiang, W., Walkey, C. D., Chan, W. C. W., & Cohen, Y. (2015). Prediction of nanoparticles-cell association based on corona proteins and physicochemical properties. *Nanoscale*, 7(21), 9664–9675. <https://doi.org/10.1039/C5NR01537E>
17. Liu, R., Rallo, R., Weissleder, R., Tassa, C., Shaw, S., & Cohen, Y. (2013). Nano-SAR development for bioactivity of nanoparticles with considerations of decision boundaries. *Small (Weinheim an Der Bergstrasse, Germany)*, 9(9–10), 1842–1852. <https://doi.org/10.1002/sml.201201903>
18. Melagraki, G., & Afantitis, A. (2015). A risk assessment tool for the virtual screening of metal oxide nanoparticles through enalos insiliconano platform. *Current topics in medicinal chemistry*, 15(18), 1827-1836.
19. Mu, Y., Wu, F., Zhao, Q., Ji, R., Qie, Y., Zhou, Y., ... & Xing, B. (2016). Predicting toxic potencies of metal oxide nanoparticles by means of nano-QSARs. *Nanotoxicology*, 10(9), 1207-1214. <http://doi.org/10.1080/17435390.2016.1202352>
20. Netzeva, T. I., Worth, A., Aldenberg, T., Benigni, R., Cronin, M. T. D., Gramatica, P., ... Yang, C. (2005). Current status of methods for defining the applicability domain of (quantitative) structure-activity relationships. The report and recommendations of ECVAM Workshop 52. *Alternatives to Laboratory Animals: ATLA*, 33(2), 155–173. <https://doi.org/10.1177/026119290503300209>
21. 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: 31/05/19)
22. Oh, E., Liu, R., Nel, A., Gemill, K. B., Bilal, M., Cohen, Y., & Medintz, I. L. (2016). Meta-analysis of cellular toxicity for cadmium-containing quantum dots. *Nature Nanotechnology*, 11(5), 479–486. <https://doi.org/10.1038/nnano.2015.338>
23. Oksel, C., Ma, C. Y., & Wang, X. Z. (2015b). Current situation on the availability of nanostructure–biological activity data. *SAR and QSAR in Environmental Research*, 26(2), 79-94. <https://doi.org/10.1080/1062936X.2014.993702>
24. Oksel, C., Ma, C. Y., Liu, J. J., Wilkins, T., & Wang, X. Z. (2017). Literature review of (Q) SAR modelling of nanomaterial toxicity. In *Modelling the Toxicity of Nanoparticles* (pp. 103-142). Springer, Cham. [https://doi.org/10.1007/978-3-319-47754-1\\_5](https://doi.org/10.1007/978-3-319-47754-1_5)
25. Oksel, C., Ma, C.Y., Liu, J.J., Wilkins, T., Wang, X.Z. (2015a). (Q)SAR modelling of nanomaterial toxicity: a critical review. *Particuology* 21, 1-19. <https://doi.org/10.1016/j.partic.2014.12.001>
26. Pan, Y., Li, T., Cheng, J., Telesca, D., Zink, J. I., & Jiang, J. (2016). Nano-QSAR modeling for predicting the cytotoxicity of metal oxide nanoparticles using novel descriptors. *RSC Advances*, 6(31), 25766–25775. <https://doi.org/10.1039/C6RA01298A>
27. Pandharipande, P. V., Mora, J. T., Uppot, R. N., Goehler, A., Braschi, M., Halpern, E. F., ... Harisinghani, M. G. (2009). Lymphotropic nanoparticle-enhanced MRI for independent prediction of lymph node malignancy: a logistic regression model. *AJR. American Journal of Roentgenology*, 193(3), W230-237. <https://doi.org/10.2214/AJR.08.2175>
28. Pathakoti, K., Huang, M.-J., Watts, J. D., He, X., & Hwang, H.-M. (2014). Using experimental data of Escherichia coli to develop a QSAR model for predicting the photo-induced cytotoxicity of metal oxide nanoparticles. *Journal of Photochemistry and Photobiology B: Biology*, 130, 234–240. <https://doi.org/10.1016/j.jphotobiol.2013.11.023>
29. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12(Oct), 2825-2830.
30. Puzyn, T., Jeliaskova, N., Sarimveis, H., Robinson, R. L. M., Lobaskin, V., Rallo, R., ... & Cronin, M. T. (2018). Perspectives from the NanoSafety Modelling Cluster on the validation criteria for (Q) SAR models used in nanotechnology. *Food and Chemical Toxicology*, 112, 478-494. <https://doi.org/10.1016/j.fct.2017.09.037>



31. Puzyn, T., Rasulev, B., Gajewicz, A., Hu, X., Dasari, T. P., Michalkova, A., ... Leszczynski, J. (2011). Using nano-QSAR to predict the cytotoxicity of metal oxide nanoparticles. *Nature Nanotechnology*, 6(3), 175–178. <https://doi.org/10.1038/nnano.2011.10>
32. R Development Core Team. (2012) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org/>. R Found. Stat. Comput. Vienna, Austria.
33. Report on Considerations from Case Studies on Integrated Approaches for Testing and Assessment (IATA) - First Review Cycle (2015) Case Studies on Grouping Methods as a Part of IATA. Third Review Cycle (2017) Series on Testing and Assessment No. 289. [http://www.oecd.org/officialdocuments/publicdisplaydocumentpdf/?cote=ENV/JM/MONO\(2018\)25&docLanguage=En](http://www.oecd.org/officialdocuments/publicdisplaydocumentpdf/?cote=ENV/JM/MONO(2018)25&docLanguage=En) (accessed online 31/05/2019)
34. Rispoli, F., Angelov, A., Badia, D., Kumar, A., Seal, S., & Shah, V. (2010). Understanding the toxicity of aggregated zero valent copper nanoparticles against Escherichia coli. *Journal of Hazardous Materials*, 180(1–3), 212–216. <https://doi.org/10.1016/j.jhazmat.2010.04.016>
35. Roy, K., Kar, S., & Das, R. N. (2015). Understanding the basics of QSAR for applications in pharmaceutical sciences and risk assessment. Academic press. <https://doi.org/10.1016/C2014-0-00286-9>
36. Roy K, Kar S (2015) Importance of applicability domain of QSAR models. In: Roy K (ed) Quantitative structure-activity relationships in drug design, predictive toxicology, and risk assessment. IGI Global, Hershey PA, USA, pp 180–211. <https://doi.org/10.4018/978-1-4666-8136-1.ch005>
37. Sayes, C., & Ivanov, I. (2010). Comparative Study of Predictive Computational Models for Nanoparticle-Induced Cytotoxicity: Comparative Study of Predictive Computational Models. *Risk Analysis*, 30(11), 1723–1734. <https://doi.org/10.1111/j.1539-6924.2010.01438.x>
38. Sellers, K., Deleebeeck, N., Messiaen, M., Jackson, M., Bleeker, E. A. J., Sijm, D., & A. van Broekhuizen, F. (2015). *Grouping Nanomaterials - A strategy towards grouping and read-across*.
39. Singh, K. P., & Gupta, S. (2014). Nano-QSAR modeling for predicting biological activity of diverse nanomaterials. *RSC Advances*, 4(26), 13215–13230. <https://doi.org/10.1039/C4RA01274G>
40. Tantra, R., Oksel, C., Puzyn, T., Wang, J., Robinson, K. N., Wang, X. Z., ... & Wilkins, T. (2015). Nano (Q) SAR: Challenges, pitfalls and perspectives. *Nanotoxicology*, 9(5), 636–642. <https://doi.org/10.3109/17435390.2014.952698>
41. Toropov, A. A., Rasulev, B. F., Leszczynska, D., & Leszczynski, J. (2008). Multiplicative SMILES-based optimal descriptors: QSPR modeling of fullerene C60 solubility in organic solvents. *Chemical Physics Letters*, 457(4–6), 332–336. <https://doi.org/10.1016/j.cplett.2008.04.013>
42. Toropova, A. P., & Toropov, A. A. (2013). Optimal descriptor as a translator of eclectic information into the prediction of membrane damage by means of various TiO2 nanoparticles. *Chemosphere*, 93(10), 2650–2655. <https://doi.org/10.1016/j.chemosphere.2013.09.089>
43. Varsou, D. D., Afantitis, A., Tsoumanis, A., Melagraki, G., Sarimveis, H., Valsami-Jones, E., & Lynch, I. (2019). A safe-by-design tool for functionalised nanomaterials through the Enalos Nanoinformatics Cloud platform. *Nanoscale Advances*, 1(2), 706–718. doi: 10.1039/c8na00142a
44. Varsou, D.-D., Nikolakopoulos, S., Tsoumanis, A., Melagraki, G., & Afantitis, A. (2018). Enalos KNIME Nodes: New Cheminformatics Tools for Drug Discovery. *Methods in Molecular Biology Rational Drug Design*, 113–138.
45. Walkey, C. D., Olsen, J. B., Song, F., Liu, R., Guo, H., Olsen, D. W. H., ... Chan, W. C. W. (2014). Protein Corona Fingerprinting Predicts the Cellular Interaction of Gold and Silver Nanoparticles. *ACS Nano*, 8(3), 2439–2455. <https://doi.org/10.1021/nn406018g>
46. Winkler, D. A. (2016). Recent advances, and unresolved issues, in the application of computational modelling to the prediction of the biological effects of nanomaterials. *Toxicology and applied pharmacology*, 299, 96–100. <https://doi.org/10.1016/j.taap.2015.12.016>
47. Wold, S., Josefson, M., Gottfries, J., & Linusson, A. (2004). The utility of multivariate design in PLS modeling. *Journal of Chemometrics*, 18(3–4), 156–165. <https://doi.org/10.1002/cem.861>

- 
48. Worth, A., Aschberger, K., Asturiol, D., Bessems, J., Gerloff, K., Graepel, R., ... & Richarz, A. N. (2017). Evaluation of the availability and applicability of computational approaches in the safety assessment of nanomaterials: Final report of the Nanocomput project. Luxembourg: Publications Office of the European Union. <http://dx.doi.org/10.2760/248139>
  49. Xia, X. R., Monteiro-Riviere, N. A., Mathur, S., Song, X., Xiao, L., Oldenberg, S. J., ... Riviere, J. E. (2011). Mapping the Surface Adsorption Forces of Nanomaterials in Biological Systems. *ACS Nano*, 5(11), 9074–9081. <https://doi.org/10.1021/nn203303c>
  50. Zarei, M., Khataee, A. R., Ordikhani-Seyedlar, R., & Fathinia, M. (2010). Photoelectro-Fenton combined with photocatalytic process for degradation of an azo dye using supported TiO<sub>2</sub> nanoparticles and carbon nanotube cathode: Neural network modeling. *Electrochimica Acta*, 55(24), 7259–7265. <https://doi.org/10.1016/j.electacta.2010.07.050>

## Annex

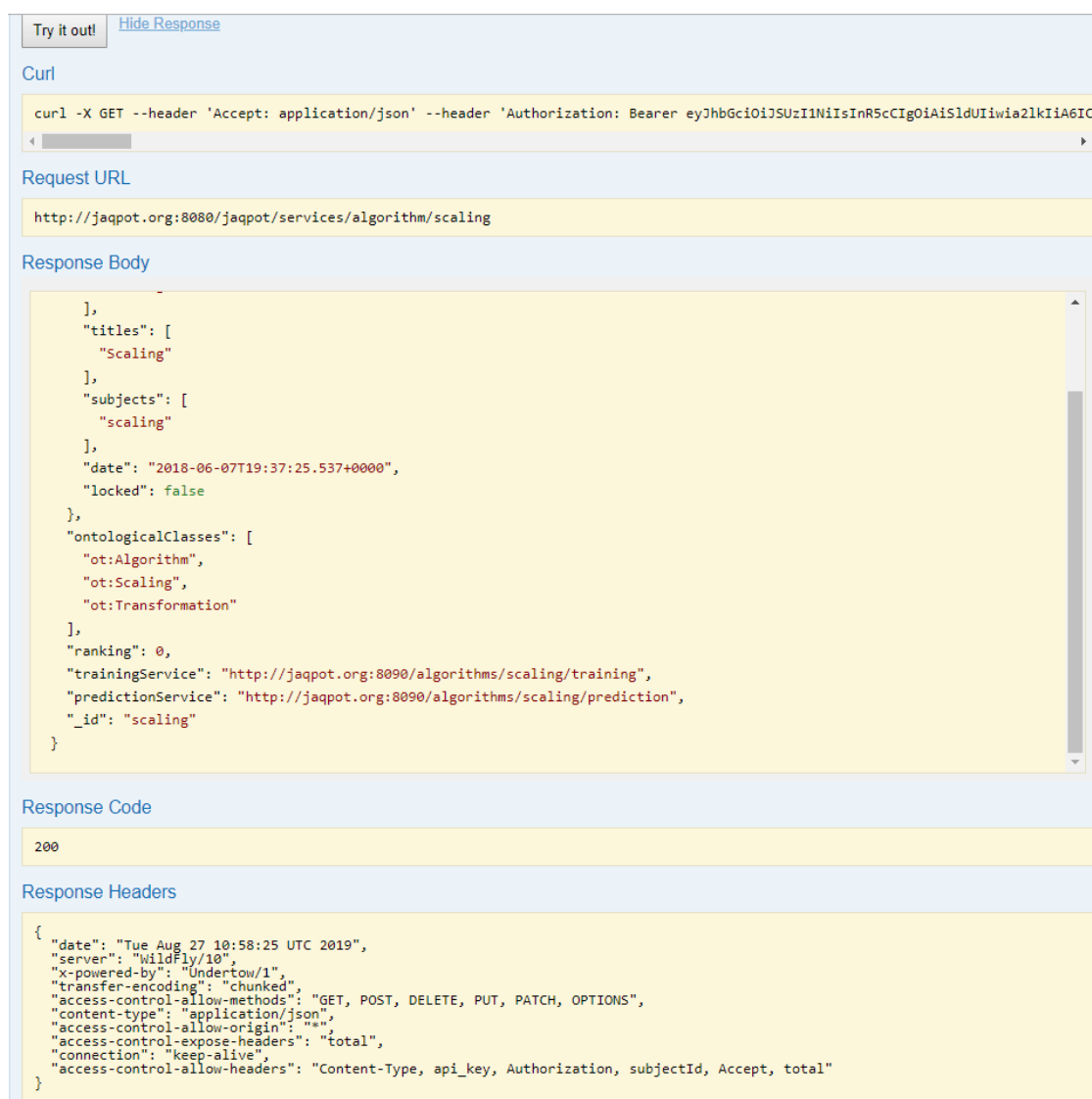
### Appendix 1. Available algorithms in Jaqpot 4 for training nanoQSAR models

	Category / URI	Description	OpenTox Ontological Classes
	<b>Preparation-additional</b>		
1	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/scaling">http://jaqpot.org:8080/jaqpot/services/algorithm/scaling</a>	Scaling	"ot:Algorithm", "ot:Scaling", "ot:Transformation"
2	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/standardization">http://jaqpot.org:8080/jaqpot/services/algorithm/standardization</a>	Standardisation	"ot:Algorithm", "ot:Scaling", "ot:Transformation"
3	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/pmmml">http://jaqpot.org:8080/jaqpot/services/algorithm/pmmml</a>	PMML Transformation algorithm	"ot:Algorithm", "ot:Transformation"
4	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/leverage">http://jaqpot.org:8080/jaqpot/services/algorithm/leverage</a>	Domain of Applicability (DoA) calculation	"ot:Algorithm", "ot:ApplicabilityDomain", "ot:Leverage"
	<b>WEKA (Java)</b>		
1	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/weka-mlr">http://jaqpot.org:8080/jaqpot/services/algorithm/weka-mlr</a>	MLR - Weka (multi-response linear regression implemented in Java-WEKA)	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
2	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/weka-svm">http://jaqpot.org:8080/jaqpot/services/algorithm/weka-svm</a>	LibSVM implementation for training a support vector classifier.	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
3	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/weka-pls">http://jaqpot.org:8080/jaqpot/services/algorithm/weka-pls</a>	PLS - Weka (Partial Least Squares implemented in Java-WEKA)	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
4	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/weka-svm-class">http://jaqpot.org:8080/jaqpot/services/algorithm/weka-svm-class</a>	LibSVM implementation for training a support vector classifier.	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
	<b>Python</b>		
1	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-id3-mci">http://jaqpot.org:8080/jaqpot/services/algorithm/python-id3-mci</a>	Id3 - with MCI (Implemented in Python-Scikit-Learn)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
2	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-linear-regression">http://jaqpot.org:8080/jaqpot/services/algorithm/python-linear-regression</a>	Linear Regression (Implemented in	"ot:Algorithm",

	<a href="#">/algorithm/python-lm</a>	Python-Scikit Learn)	"ot:Regression", "ot:SupervisedLearning"
3	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-lasso">http://jaqpot.org:8080/jaqpot/services/algorithm/python-lasso</a>	Lasso Regression (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
4	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-id3">http://jaqpot.org:8080/jaqpot/services/algorithm/python-id3</a>	Decision Tree with ID3 method by Quinlan.(Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
5	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-cmi">http://jaqpot.org:8080/jaqpot/services/algorithm/python-cmi</a>	Decision Tree with splitting criterion: Conditional Mutual Information. (Implemented in Python-Scikit Learn)"	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
6	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-gnb">http://jaqpot.org:8080/jaqpot/services/algorithm/python-gnb</a>	Generalised Naive Bayes by Scikit Learn python library. (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
7	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-mnb">http://jaqpot.org:8080/jaqpot/services/algorithm/python-mnb</a>	Multinomial Naive Bayes by Scikit Learn python library. (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
8	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-bnb">http://jaqpot.org:8080/jaqpot/services/algorithm/python-bnb</a>	Bernoulli Naive Bayes by Scikit Learn python library. (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
9	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-pls-vip">http://jaqpot.org:8080/jaqpot/services/algorithm/python-pls-vip</a>	PLS with VIP scores python algorithm. (Implemented in Python)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
10	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-rfc">http://jaqpot.org:8080/jaqpot/services/algorithm/python-rfc</a>	Random Forest Classifier	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
11	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-gbr">http://jaqpot.org:8080/jaqpot/services/algorithm/python-gbr</a>	Gradient Boosting Regressor	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
12	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-mlpc">http://jaqpot.org:8080/jaqpot/services/algorithm/python-mlpc</a>	Multi-layer Perceptron classifier. (Implemented in Python-Scikit Learn)"	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
13	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-rfr">http://jaqpot.org:8080/jaqpot/services/algorithm/python-rfr</a>	Random Forest Regressor. (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
14	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-gbc">http://jaqpot.org:8080/jaqpot/services/algorithm/python-gbc</a>	Gradient Boosting Classifier. (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Classification", "ot:SupervisedLearning"
15	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/python-mlpr">http://jaqpot.org:8080/jaqpot/services/algorithm/python-mlpr</a>	Multi-layer Perceptron Regressor. (Implemented in Python-Scikit Learn)	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
	<b>R</b>		

1	<a href="http://jaqpot.org:8080/jaqpot/services/algorithm/ocpu-lm">http://jaqpot.org:8080/jaqpot/services/algorithm/ocpu-lm</a>	Linear Model by R under OpenCPU (implemented in R - base library)	"ot:Algorithm", "ot:Regression", "ot:SupervisedLearning"
---	---	--	--

The algorithm information is available through the Jaqpot 4 API documentation at: <http://jaqpot.org:8080/jaqpot/swagger/#!/algorithm/getAlgorithm>. The user just needs to type the name of the algorithm in the *id* field and press the "Try it out!" Button. For example for the scaling algorithm, the response is shown in the following figure:



Try it out! [Hide Response](#)

Curl

```
curl -X GET --header 'Accept: application/json' --header 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCI6IWR5bG93a2kiOiA6TC
```

Request URL

```
http://jaqpot.org:8080/jaqpot/services/algorithm/scaling
```

Response Body

```
{
  "titles": [
    "Scaling"
  ],
  "subjects": [
    "scaling"
  ],
  "date": "2018-06-07T19:37:25.537+0000",
  "locked": false
},
{
  "ontologicalClasses": [
    "ot:Algorithm",
    "ot:Scaling",
    "ot:Transformation"
  ],
  "ranking": 0,
  "trainingService": "http://jaqpot.org:8090/algorithms/scaling/training",
  "predictionService": "http://jaqpot.org:8090/algorithms/scaling/prediction",
  "_id": "scaling"
}
```

Response Code

```
200
```

Response Headers

```
{
  "date": "Tue Aug 27 10:58:25 UTC 2019",
  "server": "WildFly/10",
  "x-powered-by": "Undertow/1",
  "transfer-encoding": "chunked",
  "access-control-allow-methods": "GET, POST, DELETE, PUT, PATCH, OPTIONS",
  "content-type": "application/json",
  "access-control-allow-origin": "*",
  "access-control-expose-headers": "total",
  "connection": "keep-alive",
  "access-control-allow-headers": "Content-Type, api_key, Authorization, subjectId, Accept, total"
}
```

Figure 71. Example of the Jaqpot 4 API response on the GET/algorithm/{id} method

## Appendix 2. An example of a full editable QPRF report generated by Jaqpot 4

**Report: #ND0zm2stPTRUQSu**

Title:

Description:

Date:

Disclaimer and Instructions:

Time:

Title:

Version:

**1. Substance**

	Title	Value
1.1	CAS number	Report the CAS number.
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	Report available structural information for the substance, including the structure code used to run the model. If you used a SMILES or InChI code, report the code in the corresponding field below. If you have used any another format (e.g. mol file), please include the corresponding structural representation as supporting information.
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	14/05/2019
2.2	QPRF author and contact details	2816d93a-cffb-4dab-8680-1471f1428a46
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	logS Exp_
3.1 b.	Dependent variable	logS Exp_
3.2 General	Algorithm	(OECD Principle 2)
3.2 a.	Model or submodel name	Linear Regression (Implemented in Python-Scikit Learn)
3.2 b.	Model version	Identify, where relevant, the version number and/or date of the model and submodel.
3.2 c.	Reference to QMRF	Provide relevant information about the QMRF that stores information about the model used to make the prediction. Possible useful pieces of information are: availability, source, reference number (if any) of the QMRF. Examples: 'The corresponding QMRF named -BIOWIN for Biodegradation- has been downloaded from the JRC QSAR Model Database'; 'The corresponding QMRF named -TOPKAT Skin Irritation Acyclics (Acids, Amines, Esters) MOD v SEV Model- has been newly compiled'.
3.2 d.	Predicted value (model result)	-3.91355789467
3.2 e.	Predicted value (comments)	If the result is qualitative (e.g. yes/no) or semi-quantitative (e.g. low/medium/high), explain the cut-off values that were used as the basis for classification. In reporting the predicted value, pay attention to the transformations (e.g. if the prediction is made in log units, apply anti-logarithm function).
3.2 f.	Input for prediction	Specify what kind of input was used to generate the prediction (SMILES, mol file, graphical interface etc). Please provide the structure code used to generate the

		prediction (unless already provided in section 1.5).
3.2 g.	Descriptor values	<a href="http://jaqpote.org:8080/jaqpote/services/feature/3_Seigp_zXaWqIFNEaNK">http://jaqpote.org:8080/jaqpote/services/feature/3_Seigp_zXaWqIFNEaNK</a> = 0.578, <a href="http://jaqpote.org:8080/jaqpote/services/feature/5_H1m_ynAkFCadXM1p">http://jaqpote.org:8080/jaqpote/services/feature/5_H1m_ynAkFCadXM1p</a> = 0.927, <a href="http://jaqpote.org:8080/jaqpote/services/feature/1_piPC03_MNGtMDajKNog">http://jaqpote.org:8080/jaqpote/services/feature/1_piPC03_MNGtMDajKNog</a> = 1.609, <a href="http://jaqpote.org:8080/jaqpote/services/feature/2_ATS1m_Cga8dtiGBr6R">http://jaqpote.org:8080/jaqpote/services/feature/2_ATS1m_Cga8dtiGBr6R</a> = 2.473, <a href="http://jaqpote.org:8080/jaqpote/services/feature/4_More23e_XOyvLRga2vTW">http://jaqpote.org:8080/jaqpote/services/feature/4_More23e_XOyvLRga2vTW</a> = 0.025,
3.3 General	Applicability domain	(OECD principle 3)
3.3 a.	Domains	Value: 0.773992431967 for method: leverage. Also, please see PCA figure included in this document.
3.3 b.	Structural analogues	<a href="#">/substance/49</a> , <a href="#">/substance/57</a> , <a href="#">/substance/32</a>
3.3 c.	Considerations on structural analogues	Discuss how predicted and experimental data for analogues support the prediction of the chemical under consideration.
3.4	The uncertainty of the prediction (OECD principle 4)	If possible, comment on the uncertainty of the prediction for this chemical, taking into account relevant information (e.g. variability of the experimental results).
3.5	The chemical and biological mechanisms according to the model underpinning the predicted result (OECD principle 5)	Discuss the mechanistic interpretation of the model prediction for this specific chemical. For an expert system based on structural alerts (e.g. Derek for Windows, OncologicTM) the rationale for the structural alert fired should be provided.
General	Instructions	The information provided in this section will help to facilitate considerations on the scientific validity of the model (as defined in the OECD Principles for the validation of (Q)SAR models) and the reliability of the prediction. Detailed information on the model are stored in the corresponding QMRF which is devised to reflect as much as possible the OECD principles. Remember that the QMRF and the QPRF are complementary, and a QPRF should always be associated with a defined QMRF.

#### 4. Adequacy (Optional)

	Title	Value
4.1	Regulatory purpose	Explain the regulatory purpose for which the prediction described in Section 3 is being used.
4.2	Approach for regulatory interpretation of	Describe how the predicted result is going to be interpreted in light of the specific regulatory purpose (e.g. by applying an algorithm or regulatory criteria). This may involve the need to convert the units of the dependent variable (e.g. from log molar



	the model result	units to mg/l). It may also involve the application of another algorithm, an assessment factor, or regulatory criteria, and the use or consideration of additional information in a weight-of-evidence assessment.
4.3	Outcome	Report the interpretation of the model result in relation to the defined regulatory purpose.
4.4	Conclusion	Provide an assessment of whether the final result is considered adequate for a regulatory conclusion, or whether additional information is required (and, if so, what this additional information should be).
General	Instructions	The information provided in this section might be useful, depending on the reporting needs and formats of the regulatory framework of interest. This information aims to facilitate considerations about the adequacy of the (Q)SAR prediction (result) estimate. A (Q)SAR prediction may or may not be considered adequate ('fit-for-purpose'), depending on whether the prediction is sufficiently reliable and relevant in relation to the particular regulatory purpose. The adequacy of the prediction also depends on the availability of other information, and is determined in a weight-of-evidence assessment.

**PCA of Query instance vs. Training Dataset**



---

## Appendix 3. Guidelines on installing the *jaqpotpy* library

### Installation

In order to use *jaqpotpy*, users need to install it first. Installation can be executed conveniently as a `pip` package.

```
pip install jaqpotpy
```

### Usage and initialization

#### Import Jaqpot

After installation, *Jaqpot* needs to be imported with the following command:

```
from jaqpotpy import Jaqpot
```

#### Initialize *Jaqpotpy* on the services where *jaqpot* lives.

```
jaqpot = Jaqpot("https://api.jaqpot.org/jaqpot/services/")
```

#### User authentication by *Jaqpot*

In order to access *Jaqpot* services, first the authentication of the user is required. First it is necessary to define the web location of the *Jaqpot* instance being used.

```
jaqpot = Jaqpot("https://api.jaqpot.org/jaqpot/services/")
```

The following command will send your username and password, execute your login and set the API key that is needed:

```
jaqpot.request_key('username', 'password')
```

Same as above, this command hides the password if *Jaqpot* is used through a jupyter notebook etc. and initiates a prompt for your username and password, only visible to the user:

```
jaqpot.request_key_safe()
```

Alternatively, for users that have logged in through Google or GitHub it is possible to login with the use of an API key. At the account page, the user can find an API key that can be used in order to have access to the services and send it to Jaqpot by substituting the `api_key` field. Please note that these keys have a short life and should be updated on each login.

```
jaqpot.set_api_key("api_key")
```

## Deploy your models!

Users can use the commands below, customised per model type, in order to make models trained with scikit-learn algorithms available as web services through Jaqpot. Please note:

- all models should be trained with variables that are pandas dataframes
- when calling a `jaqpot.deploy` function users should use exactly the same variables as used to train the model
- The Y variable (prediction endpoint) should have a name (Can be checked for both X and Y with `list(dfName)` if x, y are dataframes or with `dfName.name` if Y is a pandas series).

### 1. `deploy_linear_model()`

```
jaqpot.deploy_linear_model()
```

Lets users deploy linear models that are created with scikit-learn. Aa list of the produced models that can be deployed with this function is given here:

- `linear_model.ARDRRegression()`
- `linear_model.BayesianRidge()`
- `linear_model.ElasticNet()`
- `linear_model.ElasticNetCV()`
- `linear_model.HuberRegressor()`
- `linear_model.Lars()`
- `linear_model.LarsCV()`
- `linear_model.Lasso()`
- `linear_model.LassoCV()`
- `linear_model.LassoLars()`
- `linear_model.LassoLarsCV()`
- `linear_model.LassoLarsIC()`
- `linear_model.LinearRegression()`
- `linear_model.LogisticRegression()`
- `linear_model.LogisticRegressionCV()`
- `linear_model.MultiTaskLasso()`

- `linear_model.MultiTaskElasticNet()`
- `linear_model.MultiTaskLassoCV()`
- `linear_model.MultiTaskElasticNetCV()`
- `linear_model.OrthogonalMatchingPursuit()`
- `linear_model.OrthogonalMatchingPursuitCV()`
- `linear_model.PassiveAggressiveClassifier()`
- `linear_model.PassiveAggressiveRegressor()`
- `linear_model.Perceptron()`
- `linear_model.RANSACRegressor()`
- `linear_model.Ridge()`
- `linear_model.RidgeClassifier()`
- `linear_model.RidgeClassifierCV()`
- `linear_model.RidgeCV()`
- `linear_model.SGDClassifier()`
- `linear_model.SGDRegressor()`
- `linear_model.TheilSenRegressor()`
- `linear_model.enet_path()`
- `linear_model.lars_path()`
- `linear_model.lasso_path()`
- `linear_model.logistic_regression_path()`
- `linear_model.orthogonal_mp()`
- `linear_model.orthogonal_mp_gram()`
- `linear_model.ridge_regression()`

`deploy_linear_model()` parameters are:

- **model** :{is a sklearn trained model} A trained model that occurs from the `sklearn.linear_model` family of algorithms
- **X** : {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y** : {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

#### *Example usage*

```
from jaqpotpy import Jaqpot
import pandas as pd
from sklearn import linear_model

df = pd.read_csv('/path/train.csv')
X = df[['Pclass', 'SibSp', 'Parch', 'Fare']]
y = df['Survived']
```

```
clf = LogisticRegression(random_state=0, solver='lbfgs', multi_class='multinomial').fit(X, y)

jaqpot.deploy_linear_model(clf, X, y, title="Sklearn 2", description="Logistic regression model from python for the
titanic dataset",
    algorithm="logistic regression")
```

On the above example a linear model (in our case a logistic regression) is created and deployed on Jaqpot. The dataset is read as a pandas dataframe (a requirement for Jaqpot 5) and the X and y data frames are created, on which the algorithm is trained and the model is created.

## 2. deploy\_cluster()

Allows deployment of cluster models that are created from scikit-learn algorithms:

- cluster.AffinityPropagation()
- cluster.AgglomerativeClustering()
- cluster.Birch()
- cluster.DBSCAN()
- cluster.FeatureAgglomeration()
- cluster.KMeans()
- cluster.MinibatchKMeans()
- cluster.MeanShift()
- cluster.SpectralClustering()

jaqpot.deploy\_deploy\_cluster() parameters are:

- **model**: {is a sklearn trained model} a trained model that occurs from the sklearn.linear\_model family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

## 3. deploy\_ensemble()

Allows deployment of models that are created from scikit-learn algorithms:

- ensemble.AdaBoostClassifier()
- ensemble.AdaBoostRegressor()
- ensemble.BaggingClassifier()
- ensemble.BaggingRegressor()
- ensemble.ExtraTreesClassifier()
- ensemble.ExtraTreesRegressor()
- ensemble.GradientBoostingClassifier()
- ensemble.GradientBoostingRegressor()

- ensemble.IsolationForest()
- ensemble.RandomForestClassifier()
- ensemble.RandomForestRegressor()
- ensemble.RandomTreesEmbedding()
- ensemble.VotingClassifier()

jaqpot.deploy\_ensemble() parameters are:

- **model**: {is a sklearn trained model} is a trained model that occurs from the sklearn.linear\_model family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y**: {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

#### 4. deploy\_naive\_bayess()

Allows deployment of naive\_bayess models that are created from scikit-learn algorithms:

- naive\_bayess.BernoulliNB()
- naive\_bayess.GaussianNB()
- naive\_bayess.MultinomialNB()
- naive\_bayess.ComplementNB()

jaqpot.deploy\_naive\_bayess() parameters are:

- **model**: {is a sklearn trained model} is a trained model that occurs from the sklearn.linear\_model family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y**: {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

#### 5. deploy\_nearest\_neighbors()

Allows deployment of nearest\_neighbors models that are created from scikit-learn algorithms:

- neighbors.KNeighborsClassifier()
- neighbors.KNeighborsRegressor()
- neighbors.LocalOutlierFactor()
- neighbors.RadiusNeighborsClassifier()

- `neighbors.RadiusNeighborsRegressor()`
- `neighbors.NearestCentroid()`
- `neighbors.NearestNeighbors()`
- `neighbors.kneighbors_graph()`
- `neighbors.radius_neighbors_graph()`

`jaqpot.deploy_nearest_neighbors()` parameters are:

- **model**: {is a sklearn trained model} is a trained model that occurs from the `sklearn.linear_model` family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y**: {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

If `y` is empty, Jaqpot generates an empty dataframe with the title of the predicted feature.

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

## 6. `deploy_neural_network()`

Allows deployment of `neural_network` models that are created from scikit-learn algorithms:

- `neural_network.BernoulliRBM()`
- `neural_network.MLPClassifier()`
- `neural_network.MLPRegressor()`

`jaqpot.deploy_neural_network()` parameters are:

- **model**: {is a sklearn trained model} is a trained model that occurs from the `sklearn.linear_model` family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y**: {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

## 7. `deploy_svm()`

Allows deployment of `svm` models that are created from scikit-learn algorithms:

- `svm.LinearSVC()`
- `svm.LinearSVR()`
- `svm.NuSVC()`



- `svm.NuSVR()`
- `svm.OneClassSVM()`
- `svm.SVC()`
- `svm.SVR()`
- `svm.l1_min_c()`

`jaqpot.deploy_svm()` parameters are:

- **model**: {is a sklearn trained model} is a trained model that occurs from the `sklearn.linear_model` family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y**: {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

if y is empty generate an empty dataframe with the title of the predicted feature.

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

## 8. `deploy_tree()`

Allows deployment of tree models that are created from scikit-learn algorithms:

- `tree.DecisionTreeClassifier()`
- `tree.DecisionTreeRegressor()`
- `tree.ExtraTreeClassifier()`
- `tree.ExtraTreeRegressor()`

`jaqpot.deploy_tree()` parameters are:

- **model**: {is a sklearn trained model} is a trained model that occurs from the `sklearn.linear_model` family of algorithms
- **X**: {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y**: {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

## 9. `deploy_pipeline()`

Allows deployment of pipelined models that are created from scikit-learn algorithms.

`jaqpot.deploy_pipeline()` parameters are:

- **pipeline**: sklearn pipeline model is a trained model that occurs from the `sklearn.linear_model` family of algorithms

- **X** : {is a pandas dataframe} The dataframe that is used to train the model (X variables).
- **y** : {is a pandas dataframe} The dataframe that is used to train the model (y variables).
- **title**: {is a String} The title of the model
- **description**: {is a String} The description of the model
- **algorithm**: {is a String} The algorithm that the model implements string

The id of the model / pipeline is returned. The model can be found on the Jaqpot homepage of the user for editing / sharing / execution (create predictions).

#### Example usage

```
from jaqpotpy import Jaqpot
import pandas as pd
from sklearn import linear_model


df = pd.read_csv('/path/train.csv')
X = df[['Pclass', 'SibSp', 'Parch', 'Fare']]
y = df['Survived']

clf = LogisticRegression(random_state=0, solver='lbfgs', multi_class='multinomial').fit(X, y)

jaqpot.deploy_linear_model(clf, X, y, title="Sklearn 2", description="Logistic regression model from python for the
titanic dataset",
algorithm="logistic regression")
```

On the above example a linear model (in our case a Logistic Regression model) is created and deployed on Jaqpot. The dataset is read as a pandas dataframe (having the variables used for training as pandas dataframes is a requirement for all algorithms in Jaqpot 5) and the X and y data frames are created, on which the algorithm is trained and the model is created.

## Appendix 4. Example QMRF report

	<b>QMRF identifier (JRC Inventory):</b> To be entered by JRC
	<b>QMRF Title:</b> Linear nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents. The model has been presented in the publication “A Molecular Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents” Farhad Gharagheizi &Reza Fareghi Alamdari, Fullerenes, Nanotubes and Carbon Nanostructures,Volume 16, 2008 - Issue 1
	<b>Printing Date:</b> 22-Apr-2019

### 1. QSAR identifier

#### 1.1. QSAR identifier (title):

Linear nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents. The model has been presented in the publication “A Molecular-Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents” Farhad Gharagheizi &Reza Fareghi Alamdari, Fullerenes, Nanotubes and Carbon Nanostructures,Volume 16, 2008 - Issue 1.

#### 1.2. Other related models:

Neural Network nanoQSAR model predicting Solubility of C60 Fullerene in Various Solvents.

#### 1.3. Software coding the model:

Jaqpot

Jaqpot is a web platform that support development, validation and sharing of QSAR models Haralambos Sarimveis

[apps.jaqpot.org](http://apps.jaqpot.org)

### 2. General Information

#### 2.1. Date of QMRF:

21 April 2019

#### 2.2. QMRF author(s) and contact details:

Haralambos Sarimveis National Technical University of Athens [hsarimv@central.ntua.gr](mailto:hsarimv@central.ntua.gr)  
[https://www.chemeng.ntua.gr/labs/control\\_lab/sarimveis.html](https://www.chemeng.ntua.gr/labs/control_lab/sarimveis.html)

**2.3. Date of QMRF update(s):**

22 July 2019

**2.4. QMRF update(s):**

**2.5. Model developer(s) and contact details:**

[1] Farhad Gharagheizi Department of Chemical Engineering, Faculty of Engineering, University of Tehran, Tehran, Iran fghara@ut.ac.ir

[2] Reza Fareghi Alamdari Department of Chemistry, Faculty of Materials and Chemical Engineering, Malek-Ashtar University of Technology, Lavizan, Tehran, Iran

**2.6. Date of model development and/or publication:**

“A Molecular-Based Model for Prediction of Solubility of C60 Fullerene in Various Solvents” Farhad Gharagheizi & Reza Fareghi Alamdari, Fullerenes, Nanotubes and Carbon Nanostructures, Volume 16, 2008 - Issue 1.

**2.7. Reference(s) to main scientific papers and/or software package:**

**2.8. Availability of information about the model:**

**2.9. Availability of another QMRF for exactly the same model:**

**3. Defining the endpoint - OECD Principle 1**

**3.1. Species:**

**3.2. Endpoint:**

P-CHEM 4.9. Solubility in organic solvents

**3.3. Comment on endpoint:**

The solubility values are not given in weight units (e.g., mg/mL) but in terms of logarithmic values of molar fractions  $\log(S)$  because the  $\log(S)$  values correspond to the Gibbs free energy changes in the solvation process

**3.4. Endpoint units:**

Logarithmic values of molar fractions  $\log(S)$

**3.5. Dependent variable:**

Logarithmic values of molar fractions  $\log(S)$

**3.6. Experimental protocol:**

### 3.7.Endpoint data quality and variability:

#### 4. Defining the algorithm - OECD Principle 2

##### 4.1.Type of model:

nanoQSAR linear model

##### 4.2.Explicit algorithm:

Multiple linear regression (MLR)

##### 4.3.Descriptors in the model:

[1 ]piPC03 Molecular multiple path count of order 03

[2] ATS1m Broto-Mreanu autocorrelation of a topological structure-lag 1/weighted by atomic masses [3]  
SEigp Eigenvalue sum from polarizability weighted distance matrix

[4] More23e 3D-MORSE-signal 23/weighted by atomic sanderson electronegativities

[5]H1m H autocorrelation of lag 1/weighted by atomic masses

##### 4.4.Descriptor selection:

The GA-MLR algorithm proposed by Leardi et al.

##### 4.5.Algorithm and descriptor generation:

The full set contains 1664 molecular descriptors. After calculating molecular descriptors, the pool of molecular descriptors was reduced by removing descriptors that could not be calculated for every structure in the dataset, and those descriptors with an essentially constant value for all the structures. In this step, the pool of 1664 molecular descriptors was reduced to a new pool of 1259 molecular descriptors. The GA-MLR algorithm was applied on this set of descriptors.

##### 4.6.Software name and version for descriptor generation:

Dragon

Dragon 7.0 calculates 5,270 molecular descriptors, organized in different logical blocks as in the previous versions. Blocks are further divided into sub-blocks to make management, selection, and analysis of descriptors easier.

[chm@kode-solutions.net](mailto:chm@kode-solutions.net)

[https://chm.kode-solutions.net/products\\_dragon.php](https://chm.kode-solutions.net/products_dragon.php)

##### 4.7.Chemicals/Descriptors ratio:

128/5

### 5. Defining the applicability domain - OECD Principle 3

**5.1. Description of the applicability domain of the model:**

**5.2. Method used to assess the applicability domain:**

**5.3. Software name and version for applicability domain assessment:**

**5.4. Limits of applicability:**

### 6. Internal validation - OECD Principle 4

**6.1. Availability of the training set:**

Yes

**6.2. Available information for the training set:**

CAS RN: No

Chemical Name: Yes Smiles: No

Formula: No INChI: No

MOL file: No NanoMaterial: No

**6.3. Data for each descriptor variable for the training set:**

All

**6.4. Data for the dependent variable for the training set:**

All

**6.5. Other information about the training set:**

4 solvents of the original dataset were removed because they are considered as outliers in the original paper

**6.6. Pre-processing of data before modelling:**

The data are scaled in the range [0,1].

**6.7. Statistics for goodness-of-fit:**

R2 score: 0.899

**6.8. Robustness - Statistics obtained by leave-one-out cross-validation:**

**6.9. Robustness - Statistics obtained by leave-many-outcross-validation:**

R2 scores for 5-fold cross validation: 0.91906039, 0.88995619, 0.90445436, 0.86506266,  
0.62316459

**6.10. Robustness - Statistics obtained by Y-scrambling:**

**6.11. Robustness - Statistics obtained by bootstrap:**

**6.12. Robustness - Statistics obtained by other methods:**

External validation: R2\_external: 0.904

**7. External validation - OECD Principle 4**

**7.1. Availability of the external validation set:**

Yes

**7.2. Available information for the external validation set:**

CAS RN: No

Chemical Name: Yes

Smiles: No

Formula: No

INChI: No

MOL file: No

NanoMaterial: No

**7.3. Data for each descriptor variable for the external validation set: All**

**7.4. Data for the dependent variable for the external validation set: All**

**7.5. Other information about the external validation set:**

The external validation set consists of 25% of the original data selected randomly

**7.6. Experimental design of test set:**

By randomly setting aside 25% of chemicals in the training data

**7.7.Predictivity - Statistics obtained by external validation:**

External validation: R2\_external: 0.904

**7.8.Predictivity - Assessment of the external validation set:**

**7.9.Comments on the external validation of the model:**

**8.Providing a mechanistic interpretation - OECD Principle 5**

**8.1.Mechanistic basis of the model:**

**8.2.A priori or a posteriori mechanistic interpretation:**

**8.3.Other information about the mechanistic interpretation:**

**9.Miscellaneous information**

**9.1.Comments:**

**9.2.Bibliography:**

**9.3.Supporting information:**

**Training set(s)**

70_model_reduced.csv	<a href="https://app.jaqpot.org/dataset/cjCXljkX0kBbjelkHMZuNg">https://app.jaqpot.org/dataset/cjCXljkX0kBbjelkHMZuNg</a>
----------------------	---

**Test set(s) Supporting information**

**10. Summary (JRC QSAR Model Database)**

**10.1.QMRF number:**

To be entered by JRC

**10.2.Publication date:**

To be entered by JRC

**10.3.Keywords:**

To be entered by JRC

**10.4.Comments:**

To be entered by JRC



## Appendix 5. Example of a PMML representation of a nanoQSAR model

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<PMML xmlns="http://www.dmg.org/PMML-4_3" xmlns:data="http://jpmml.org/jpmml-model/InlineTable" version="4.3">
  <Header>
    <Application name="JPMML-SkLearn" version="1.5.13"/>
    <Timestamp>2019-05-02T23:16:38Z</Timestamp>
  </Header>
  <MiningBuildTask>
    <Extension>PMMLPipeline(steps=[('scaler', MinMaxScaler(copy=True, feature_range=(0, 1))),
    ('MLR', LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
    normalize=False))])</Extension>
  </MiningBuildTask>
  <DataDictionary>
    <DataField name="logS Exp." optype="continuous" dataType="double"/>
    <DataField name="piPC03" optype="continuous" dataType="double"/>
    <DataField name="ATS1m" optype="continuous" dataType="double"/>
    <DataField name="Seigp" optype="continuous" dataType="double"/>
    <DataField name="More23e" optype="continuous" dataType="double"/>
    <DataField name="H1m" optype="continuous" dataType="double"/>
  </DataDictionary>
  <TransformationDictionary>
    <DerivedField name="mix_max_scaler(piPC03)" optype="continuous" dataType="double">
      <Apply function="*">
        <FieldRef field="piPC03"/>
        <Constant dataType="double">0.2161227577263886</Constant>
      </Apply>
    </DerivedField>
    <DerivedField name="mix_max_scaler(ATS1m)" optype="continuous" dataType="double">
      <Apply function="+">
        <Apply function="*">
          <FieldRef field="ATS1m"/>
          <Constant dataType="double">0.5181347150259068</Constant>
        </Apply>
        <Constant dataType="double">-0.6238341968911918</Constant>
      </Apply>
    </DerivedField>
    <DerivedField name="mix_max_scaler(Seigp)" optype="continuous" dataType="double">
      <Apply function="+">
        <Apply function="*">
          <FieldRef field="Seigp"/>
          <Constant dataType="double">0.1404691670178396</Constant>
        </Apply>
        <Constant dataType="double">0.8217446270543616</Constant>
      </Apply>
    </DerivedField>
    <DerivedField name="mix_max_scaler(More23e)" optype="continuous" dataType="double">
      <Apply function="+">
        <Apply function="*">
          <FieldRef field="More23e"/>

```

```
<Constant dataType="double">0.3033060357901122</Constant>
</Apply>
<Constant dataType="double">0.9241734910524718</Constant>
</Apply>
</DerivedField>
<DerivedField name="mix_max_scaler(H1m)" optype="continuous" dataType="double">
  <Apply function="*">
    <FieldRef field="H1m"/>
    <Constant dataType="double">0.47619047619047616</Constant>
  </Apply>
</DerivedField>
</TransformationDictionary>
<RegressionModel functionName="regression">
  <MiningSchema>
    <MiningField name="logS Exp." usageType="target"/>
    <MiningField name="piPC03"/>
    <MiningField name="ATS1m"/>
    <MiningField name="Seigp"/>
    <MiningField name="More23e"/>
    <MiningField name="H1m"/>
  </MiningSchema>
  <RegressionTable intercept="-9.568037894195744">
    <NumericPredictor name="mix_max_scaler(piPC03)" coefficient="2.735971368129383"/>
    <NumericPredictor name="mix_max_scaler(ATS1m)" coefficient="3.393836107356581"/>
    <NumericPredictor name="mix_max_scaler(Seigp)" coefficient="2.1398594506439514"/>
    <NumericPredictor name="mix_max_scaler(More23e)" coefficient="1.1075112876187143"/>
    <NumericPredictor name="mix_max_scaler(H1m)" coefficient="-1.1972497501470432"/>
  </RegressionTable>
</RegressionModel>
</PMML>
```

## Appendix 6. Detailed information on examples of models, as implemented in Jaqpot 4 and Jaqpot 5

<b>Methodology for developing structure-activity evaluation to identify combinations of physical features of nanomaterial that influence potential cell damage by MLR/LDA (TiO<sub>2</sub> case)</b>	
<b>Source</b>	Sayes, C., & Ivanov, I. (2010). Comparative Study of Predictive Computational Models for Nanoparticle-Induced Cytotoxicity. Risk Analysis, 30(11), 1723–1734. (TiO <sub>2</sub> case) <a href="http://doi.org/10.1111/j.1539-6924.2010.01438.x">http://doi.org/10.1111/j.1539-6924.2010.01438.x</a>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - membrane damage measured as lactate dehydrogenase (LDH) release [units/L]
<b>Input Variables</b>	Size in water Concentration [mg/L] Zeta Potential [mV]
<b>Dataset</b>	<a href="http://jaqpot.org/data_detail?name=OVRXa54JPK1iSk">http://jaqpot.org/data_detail?name=OVRXa54JPK1iSk</a> Metal Oxide: TiO <sub>2</sub> 24 measures, combination of: Engineered Size (30, 45, 125) 2 x Concentration (25, 50, 100, 200)
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpot 4</b>	<a href="http://jaqpot.org/m_detail?name=2KoKHclgMJloSeWuZ03a">http://jaqpot.org/m_detail?name=2KoKHclgMJloSeWuZ03a</a>
<b>Jaqpot 5</b>	<a href="https://app.jaqpot.org/model/izMncmc5LMbgC6o7Fkj8">https://app.jaqpot.org/model/izMncmc5LMbgC6o7Fkj8</a>

<b>Methodology for developing structure-activity evaluation to identify combinations of physical features of nanomaterial that influence potential cell damage by MLR/LDA (ZnO case)</b>	
<b>Source</b>	Sayes, C., & Ivanov, I. (2010). Comparative Study of Predictive Computational Models for Nanoparticle-Induced Cytotoxicity. Risk Analysis, 30(11), 1723–1734. (ZnO case) <a href="http://doi.org/10.1111/j.1539-6924.2010.01438.x">http://doi.org/10.1111/j.1539-6924.2010.01438.x</a>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - membrane damage measured as lactate dehydrogenase (LDH) release [units/L]
<b>Input Variables</b>	Size in water Concentration [mg/L] Size in CCM
<b>Dataset</b>	<a href="http://jaqpote.org/data_detail?name=OVRXa54JPK1iSk">http://jaqpote.org/data_detail?name=OVRXa54JPK1iSk</a> Metal Oxide: ZnO 18 measures, combination of: Engineered Size (50, 60, 70, 1000, 1200, 1500) Concentration (25, 50, 100)
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpote 4</b>	<a href="http://jaqpote.org/m_detail?name=cm49KGhUjkMw6wyntKQF">http://jaqpote.org/m_detail?name=cm49KGhUjkMw6wyntKQF</a>
<b>Jaqpote 5</b>	<a href="https://app.jaqpote.org/model/fvAe4KnIOOiNgGf7Ve1p">https://app.jaqpote.org/model/fvAe4KnIOOiNgGf7Ve1p</a>

<b>Regression model to understand the aggregated ZVCN against E.Coli by MLR (Placket-Burman design)</b>	
<b>Source</b>	Rispoli, F., Angelov, A., Badia, D., Kumar, A., Seal, S., & Shah, V. (2010). Understanding the toxicity of aggregated zero valent copper nanoparticles against Escherichia coli. Journal of Hazardous Materials, 180(1-3), 212–216. <a href="http://doi.org/10.1016/j.jhazmat.2010.04.016">http://doi.org/10.1016/j.jhazmat.2010.04.016</a>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - measured as percentage of dead E. Coli population
<b>Input Variables</b>	pH Temperature Aeration rate Concentration of nanoparticles Concentration of bacteria
<b>Dataset</b>	<a href="http://jaqpot.org/data_detail?name=7Sra2sRbK1IJRt">http://jaqpot.org/data_detail?name=7Sra2sRbK1IJRt</a> 16 Metal ZVCN: zero valent copper Cu nanoparticle
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpot 4</b>	<a href="http://jaqpot.org/m_detail?name=48JYATz0KFTkZjGd8AfS">http://jaqpot.org/m_detail?name=48JYATz0KFTkZjGd8AfS</a>
<b>Jaqpot 5</b>	<a href="https://app.jaqpot.org/model/8su6n4cfcJpzZD2NDZGN">https://app.jaqpot.org/model/8su6n4cfcJpzZD2NDZGN</a>

<b>Prediction of the Biological surface adsorption index (BSAI) on different NPs by MLR</b>	
<b>Source</b>	"Xia, X. R., Monteiro-Riviere, N. A., Mathur, S., Song, X., Xiao, L., Oldenberg, S. J., ... Riviere, J. E. (2011). Mapping the surface adsorption forces of nanomaterials in biological systems. ACS Nano, 5(11), 9074-9081 <a href="http://doi.org/10.1021/nn203303c">http://doi.org/10.1021/nn203303c</a>
<b>Predicted Endpoint</b>	log(k)  k: adsorption coefficient
<b>Input Variables</b>	V: Lipophilicity interaction β: Hydrogenbond basicity α: Hydrogenbond acidity π: Dipolarity/polarizability R: lone-pair electrons
<b>Dataset</b>	<a href="http://jaqpot.org/data_detail?name=IPirs0YsISompB">http://jaqpot.org/data_detail?name=IPirs0YsISompB</a> 28 Carbon-based MWCNT40nm-COOH
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpot 4</b>	<a href="http://jaqpot.org/m_detail?name=DjRQk8AqG42nckg5KoxZ">http://jaqpot.org/m_detail?name=DjRQk8AqG42nckg5KoxZ</a>
<b>Jaqpot 5</b>	<a href="https://app.jaqpot.org/model/gSvjUZ17EEAV5OWL7Uls">https://app.jaqpot.org/model/gSvjUZ17EEAV5OWL7Uls</a>

<b>Predictive model of TiO<sub>2</sub> NPs damage on membrane cell by SMILES-based optimal descriptor and Monte Carlo technique (CORAL software)</b>	
<b>Source</b>	<p>Toropova, A. P., &amp; Toropov, A. A. (2013). Optimal descriptor as a translator of eclectic information into the prediction of membrane damage by means of various TiO<sub>2</sub> nanoparticles. <i>Chemosphere</i>, 93(10), 2650–2655.</p> <p>Toropova, A. P., Toropov, A. A., Benfenati, E., Puzyn, T., Leszczynska, D., &amp; Leszczynski, J. (2014). Optimal descriptor as a translator of eclectic information into the prediction of membrane damage: The case of a group of ZnO and TiO<sub>2</sub> nanoparticles. <i>Ecotoxicology and Environmental Safety</i>, 108, 203–209.  <a href="http://doi.org/10.1016/j.chemosphere.2013.09.089">http://doi.org/10.1016/j.chemosphere.2013.09.089</a></p>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - membrane damage measured as lactate dehydrogenase (LDH) release [units/L]
<b>Input Variables</b>	<p>Engineered Size                      Size in water                      Size in PBS                      Concentration                      Zeta potential</p>
<b>Dataset</b>	<p><a href="http://www.jaqpot.org/data_detail?name=yW3pfohTS3l33m&amp;page=2">http://www.jaqpot.org/data_detail?name=yW3pfohTS3l33m&amp;page=2</a>                      TiO<sub>2</sub> Metal Oxide                      Engineered Size: 30, 45, 125                      Size in water: 101-967                      Size in PBS: 961-3871/</p>
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpot 4</b>	<a href="http://jaqpot.org/m_detail?name=4oxlwXBZMJ4suYFTSl4d">http://jaqpot.org/m_detail?name=4oxlwXBZMJ4suYFTSl4d</a>
<b>Jaqpot 5</b>	<a href="https://app.jaqpot.org/model/nTJgb4Ss3zHIYZEcbg78">https://app.jaqpot.org/model/nTJgb4Ss3zHIYZEcbg78</a>

<b>Cytotoxicity of metal oxide to bacteria E.Coli models by Periodic table-based descriptors and stepwise-MLR</b>	
<b>Source</b>	Kar, S., Gajewicz, A., Puzyn, T., Roy, K., & Leszczynski, J. (2014). Periodic table-based descriptors to encode cytotoxicity profile of metal oxide nanoparticles: A mechanistic QSTR approach. <i>Ecotoxicology and Environmental Safety</i> , 107, 162–169. <a href="http://doi.org/10.1016/j.ecoenv.2014.05.026">http://doi.org/10.1016/j.ecoenv.2014.05.026</a>
<b>Source</b>	In vitro - Cytotoxicity - measured as pEC50
<b>Predicted Endpoint</b>	Xox: charge of metal cation corresponding to a given oxide
<b>Input Variables</b>	<a href="http://jaqpote.org/data_detail?name=7E6XB8VzAGEZNI">http://jaqpote.org/data_detail?name=7E6XB8VzAGEZNI</a> 17 Metal Oxides ZnO, CuO, Al <sub>2</sub> O <sub>3</sub> , Fe <sub>2</sub> O <sub>3</sub> , SnO <sub>2</sub> , TiO <sub>2</sub> , V <sub>2</sub> O <sub>3</sub> , Y <sub>2</sub> O <sub>3</sub> , Bi <sub>2</sub> O <sub>3</sub> , In <sub>2</sub> O <sub>3</sub> , Sb <sub>2</sub> O <sub>3</sub> , SiO <sub>2</sub> , ZrO <sub>2</sub> , CoO, NiO, Cr <sub>2</sub> O <sub>3</sub> , La <sub>2</sub> O <sub>3</sub>
<b>Dataset</b>	<b>Multiple Linear Regression</b>
<b>Algorithm</b>	<a href="http://jaqpote.org/m_detail?name=EFfilLYKMgLUq3qNYBw">http://jaqpote.org/m_detail?name=EFfilLYKMgLUq3qNYBw</a>
<b>Jaqpote 4</b>	<a href="https://app.jaqpote.org/model/QgRRwyU8r7e0NubEuDdX">https://app.jaqpote.org/model/QgRRwyU8r7e0NubEuDdX</a>



<b>Photo-induced toxicity of metal oxide NPs to E. Coli by MLR (dark condition case)</b>	
<b>Source</b>	Pathakoti, K., Huang, M.-J., Watts, J. D., He, X., & Hwang, H.-M. (2014). Using experimental data of Escherichia coli to develop a QSAR model for predicting the photo-induced cytotoxicity of metal oxide nanoparticles. Journal of Photochemistry and Photobiology B: Biology, 130, 234–240. (dark condition case) <a href="http://doi.org/10.1016/j.jphotobiol.2013.11.023">http://doi.org/10.1016/j.jphotobiol.2013.11.023</a>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - measured as -log(LC50)
<b>Input Variables</b>	MELECT: the absolute electronegativity of the metal atom LZELEHHO: the absolute electronegativity of the metal oxide
<b>Dataset</b>	<a href="http://jaqpot.org/data_detail?name=yjBO04fO3d19XL">http://jaqpot.org/data_detail?name=yjBO04fO3d19XL</a> 17 Metal Oxides ZnO, CuO, Al <sub>2</sub> O <sub>3</sub> , Fe <sub>2</sub> O <sub>3</sub> , SnO <sub>2</sub> , TiO <sub>2</sub> , V <sub>2</sub> O <sub>3</sub> , Y <sub>2</sub> O <sub>3</sub> , Bi <sub>2</sub> O <sub>3</sub> , In <sub>2</sub> O <sub>3</sub> , Sb <sub>2</sub> O <sub>3</sub> , SiO <sub>2</sub> , ZrO <sub>2</sub> , CoO, NiO, Cr <sub>2</sub> O <sub>3</sub> , La <sub>2</sub> O <sub>3</sub>
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpot 4</b>	<a href="http://jaqpot.org/m_detail?name=KIWUeelVM8x7x1iC7cXi">http://jaqpot.org/m_detail?name=KIWUeelVM8x7x1iC7cXi</a>
<b>Jaqpot 5</b>	<a href="https://app.jaqpot.org/model/hygpzrH71XS1Wr8IGS69">https://app.jaqpot.org/model/hygpzrH71XS1Wr8IGS69</a>

<b>Photo-induced toxicity of metal oxide NPs to E. Coli by MLR(Photo-induced (light) case)</b>	
<b>Source</b>	Pathakoti, K., Huang, M.-J., Watts, J. D., He, X., & Hwang, H.-M. (2014). Using experimental data of Escherichia coli to develop a QSAR model for predicting the photo-induced cytotoxicity of metal oxide nanoparticles. Journal of Photochemistry and Photobiology B: Biology, 130, 234–240. (Photo-induced (light) case) <a href="http://doi.org/10.1016/j.jphotobiol.2013.11.023">http://doi.org/10.1016/j.jphotobiol.2013.11.023</a>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - measured as -log(LC50)
<b>Input Variables</b>	Cp is the literature molar heat capacity of the metal oxide at 298.15 K. ALZLUMO is the average of the alpha and beta LUMO energies of the metal oxide.
<b>Dataset</b>	<a href="http://jaqpote.org/data_detail?name=4xoqetJXfkMB0S">http://jaqpote.org/data_detail?name=4xoqetJXfkMB0S</a> 17 Metal Oxides ZnO, CuO, Al <sub>2</sub> O <sub>3</sub> , Fe <sub>2</sub> O <sub>3</sub> , SnO <sub>2</sub> , TiO <sub>2</sub> , V <sub>2</sub> O <sub>3</sub> , Y <sub>2</sub> O <sub>3</sub> , Bi <sub>2</sub> O <sub>3</sub> , In <sub>2</sub> O <sub>3</sub> , Sb <sub>2</sub> O <sub>3</sub> , SiO <sub>2</sub> , ZrO <sub>2</sub> , CoO, NiO, Cr <sub>2</sub> O <sub>3</sub> , La <sub>2</sub> O <sub>3</sub>
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpote 4</b>	<a href="http://jaqpote.org/m_detail?name=o6Jr81BfQtUddgmwqae">http://jaqpote.org/m_detail?name=o6Jr81BfQtUddgmwqae</a>
<b>Jaqpote 5</b>	<a href="https://app.jaqpote.org/model/5gCY316DzDh1Fdw4aigo">https://app.jaqpote.org/model/5gCY316DzDh1Fdw4aigo</a>

<b>Predicting metal oxide Nps toxicity to E. Coli cell line by MLR</b>	
<b>Source</b>	Mu, Y., Wu, F., Zhao, Q., Ji, R., Qie, Y., Zhou, Y., ... Xing, B. (2016). Predicting toxic potencies of metal oxide nanoparticles by means of nano-QSARs. Nanotoxicology. State Key Laboratory of Environmental Criteria and Risk Assessment, Chinese Research Academy of Environmental Sciences, Beijing, China. <a href="http://doi.org/10.1080/17435390.2016.1202352">http://doi.org/10.1080/17435390.2016.1202352</a>
<b>Predicted Endpoint</b>	In vitro - Cytotoxicity - measured as log(1/EC50)
<b>Input Variables</b>	Z/r : Polarization force parameter $\Delta H_{Me+}$ : represents the enthalpy of formation of a gaseous cation having the same oxidation state as that in the metal oxide structure.
<b>Dataset</b>	<a href="http://jaqpot.org/data_detail?name=4fk3ZAJrRhskX4">http://jaqpot.org/data_detail?name=4fk3ZAJrRhskX4</a> 17 Metal Oxides ZnO, CuO, Al <sub>2</sub> O <sub>3</sub> , Fe <sub>2</sub> O <sub>3</sub> , SnO <sub>2</sub> , TiO <sub>2</sub> , V <sub>2</sub> O <sub>3</sub> , Y <sub>2</sub> O <sub>3</sub> , Bi <sub>2</sub> O <sub>3</sub> , In <sub>2</sub> O <sub>3</sub> , Sb <sub>2</sub> O <sub>3</sub> , SiO <sub>2</sub> , ZrO <sub>2</sub> , CoO, NiO, Cr <sub>2</sub> O <sub>3</sub> , La <sub>2</sub> O <sub>3</sub>
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpot 4</b>	<a href="http://jaqpot.org/m_detail?name=qul6HILHSypXWX8zvMQ3">http://jaqpot.org/m_detail?name=qul6HILHSypXWX8zvMQ3</a>
<b>Jaqpot 5</b>	<a href="https://app.jaqpot.org/model/OAiBYuee5PLJ7F580f2J">https://app.jaqpot.org/model/OAiBYuee5PLJ7F580f2J</a>

<b>Predicting C60 solubility in organic solvents by SMILES-based optimal descriptor and Monte Carlo technique</b>	
<b>Source</b>	Gharagheizi, F., & Alamdari, R. F. (2008). A molecular-based model for prediction of solubility of C60 fullerene in various solvents. Fullerenes Nanotubes and Carbon Nanostructures, 16(1), 40–57. <a href="http://doi.org/10.1080/15363830701779315">http://doi.org/10.1080/15363830701779315</a>
<b>Predicted Endpoint</b>	Solubility in organic solvents- measured as logS Exp
<b>Input Variables</b>	Molecular descriptors are defined for solvents according to chemical structure using the Dragon Software. piPC03: Molecular multiple path count of order 03 (walk and path counts) ATS1m 2D: Broto-Mreau autocorrelation of a topological structure-lag 1/weighted by atomic masses (2D autocorrelations) Seigp: Eigenvalue sum from polarizability weighted distance matrix (Eigenvalue 0 based indices) More23e: 3D-MORSE-signal 23/weighted by atomic sanderson electronegativities (More23e 3D-MORSE descriptors ) H1m: H autocorrelation of lag 1/weighted by atomic masses (GETAWAY descriptors)"
<b>Dataset</b>	Training: <a href="http://www.jaqpot.org/data_detail?name=XmCQVC7o5jKKRv">http://www.jaqpot.org/data_detail?name=XmCQVC7o5jKKRv</a> Test <a href="http://www.jaqpot.org/data_detail?name=3UbgEJPIIdT2Ovs">http://www.jaqpot.org/data_detail?name=3UbgEJPIIdT2Ovs</a>  Carbon-based NM, Fullerene C60
<b>Algorithm</b>	<b>Multiple Linear Regression</b>
<b>Jaqpote 4</b>	<a href="http://jaqpote.org/m_detail?name=sCoqY3D3xCpSuyS6RdoQ">http://jaqpote.org/m_detail?name=sCoqY3D3xCpSuyS6RdoQ</a>
<b>Jaqpote 5</b>	<a href="https://app.jaqpot.org/model/VRp8f6A4DuJc8fsavvpB">https://app.jaqpot.org/model/VRp8f6A4DuJc8fsavvpB</a>

## Appendix 7. Python notebook for Jaqpot-Biomax communication with summarised output

```
import sys
pip install requests
pip install pandas
!{sys.executable} -m pip install
git+https://github.com/KinkyDesign/jaqpotpy
pip install sklearn
pip install matplotlib
import getpass
import requests
import sys
import http.client as http_client
import logging
import json
import pandas as pd
import numpy as np
from jaqpotpy import Jaqpot

#Communicating with BIOMAX for the PhysicoChemical Data
user = "Hackathon"

pw = getpass.getpass("Login password for user '{}': ".format(user))

url = "https://ssl.biomax.de/nanocommons/bioxm/rest/api"

proxies = {
    #'https': 'server:8080'
}

print("Opening session...")
response = requests.get(url +
"/createUserSessionSimple?name={}&password={}".format(user, pw),
proxies=proxies)

session_id = ""
if response.status_code == 200:
    session_id = response.text
    print("Opened session " + session_id)
else:
    print("Failed to open session: " + str(response.text))
    sys.exit(1)

request_data= """
{
  "SearchResultWithReportsRequest":
  {
    "firstIndex": 0,
    "maxCount": 100,
    "local": False,
```

```
        "viewName": "ivan",
        "query": {
            "queryName": "ivan"
        }
    }
}
"""

try:
    headers = {'Content-type': 'application/json'}

    http_client.HTTPConnection.debuglevel = 1

    logging.basicConfig()
    logging.getLogger().setLevel(logging.DEBUG)
    requests_log = logging.getLogger("requests.packages.urllib3")
    requests_log.setLevel(logging.DEBUG)
    requests_log.propagate = True

    response = requests.post(url +
"/getSearchResultWithReports?userSessionId=" + session_id,
data=request_data,
                            headers=headers, proxies=proxies)
    print("Status: " + str(response.status_code))

    if (response.status_code == 200):
#         print(response.text)
        json_data2 = json.loads(response.text)
        json_dataset2 =
json_data2['SearchResultWithReports']['objectReports']

        else:
            print("ERROR:")
            print(response.text)

finally:
    response = requests.get(url + "/destroyUserSession?userSessionId=" +
session_id, proxies=proxies)
    if (response.status_code == 200):
        print("\nClosed session.")
    else:
        print("\nError: Failed to close session " + session_id)
data_dict = {}
data_dict['Id'] = []
for dato in json_data2['SearchResultWithReports']['objectReports']:
    for atr in dato['attributes']:
        data_dict[atr['name']] = []

for dato in json_data2['SearchResultWithReports']['objectReports']:
    data_dict['Id'].append(dato["objectURL"])
```

```
for key, value in data_dict.items():
    for atr in dato['attributes']:
        if atr['name'] == key:
            try:
                nr = atr['nestedReports']
                data_dict[key].append(nr['attributes']['values'])
            except KeyError:
                data_dict[key].append(None)
            continue

df = pd.DataFrame(data_dict)
```

### GETTING TOXICITY DATA

```
user = "Hackathon"

pw = getpass.getpass("Login password for user '{}': ".format(user))

url = "https://ssl.biomax.de/nanocommons/bioxm/rest/api"

proxies = {
    #'https': 'server:8080'
}

print("Opening session...")
response = requests.get(url +
    "/createUserSessionSimple?name={}&password={}".format(user, pw),
    proxies=proxies)

session_id = ""
if response.status_code == 200:
    session_id = response.text
    print("Opened session " + session_id)
else:
    print("Failed to open session: " + str(response.text))
    sys.exit(1)

request_data= """
{
  "SearchResultWithReportsRequest":
  {
    "firstIndex": 0,
    "maxCount": 100,
    "local": True,
    "viewName": "Experiment - Toxicity by aliquot",
    "query": {
      "queryName": "Experiment - Toxicity - KIT IH A549"
    }
  }
}
"""

try:
```

```
headers = {'Content-type': 'application/json'}

http_client.HTTPConnection.debuglevel = 1

logging.basicConfig()
logging.getLogger().setLevel(logging.DEBUG)
requests_log = logging.getLogger("requests.packages.urllib3")
requests_log.setLevel(logging.DEBUG)
requests_log.propagate = True

response = requests.post(url +
"/getSearchResultWithReports?userSessionId=" + session_id,
data=request_data,
                        headers=headers, proxies=proxies)
print("Status: " + str(response.status_code))

if (response.status_code == 200):
#     print(response.text)
    json_data2 = json.loads(response.text)
    json_dataset2 =
json_data2['SearchResultWithReports']['objectReports']

else:
    print("ERROR:")
    print(response.text)

finally:
    response = requests.get(url + "/destroyUserSession?userSessionId=" +
session_id, proxies=proxies)
    if (response.status_code == 200):
        print("\nClosed session.")
    else:
        print("\nError: Failed to close session " + session_id)
```

```
data_dictTox = {}
data_dictTox['Id'] = []
for dato in json_data2['SearchResultWithReports']['objectReports']:
    for atr in dato['attributes']:
        data_dictTox[atr['name']] = []
for dato in json_data2['SearchResultWithReports']['objectReports']:
    data_dictTox['Id'].append(dato["objectURL"])
    for key, value in data_dictTox.items():
        for atr in dato['attributes']:
            if atr['name'] == key:
                try:
                    nr = atr['nestedReports']
                    data_dictTox[key].append(nr['attributes'])
                except KeyError:
                    data_dictTox[key].append(None)
                continue
dfTox = pd.DataFrame(data_dictTox)
```



Preprocessing Y values - Formatting toxicity data as dataframe

```
dfToxNames=dfTox['_Particle']
CompoundNames={}
coumpountDict = {}
for compound in range(1,len(dfToxNames)):
    coumpountDict[compound] = dfToxNames[compound][0]['values']

CompoundNames['particle'] = coumpountDict

print(CompoundNames)
```

```
{'particle': {1: 'NP00375', 2: 'NP00262', 3: 'NP00266', 4: 'NP00269', 5:
'NP00255', 6: 'NP00256', 7: 'NP00257', 8: 'NP00258', 9: 'NP00259', 10: 'NP00260',
11: 'NP00254', 12: 'NP00193', 13: 'NP00192', 14: 'NP00282', 15: 'NP00283', 16:
'NP00214', 17: 'NP00214'}}
```

```
dfTox2=dfTox['_Toxicity.A549_Dose_39.1_IH']
ViableCellCount={}
i=0
for compound in range(len(dfTox2)):
    df3=dfTox2[compound]
    df5=pd.DataFrame(df3, columns =['name','values'])
    ViableCellCount[compound]=df5.iloc[0]['values']
ViableCellCount
```

```
{0: nan,
1: nan,
2: 1.422835941,
3: -0.401260535,
4: 2.434066141,
5: -0.040606937,
6: 0.084394911,
7: 0.057642569,
8: 0.284920008,
9: -0.450941764,
10: 0.525510532,
11: -0.213755604,
12: 0.247470485,
13: 0.818199323,
14: -7.30424163,
15: -9.238114492,
```

16: -16.06765605,  
17: -16.06765605}

```
dictF = {}  
dictF['ViableCellCount']=ViableCellCount  
ViableCellCountdf = pd.DataFrame.from_dict(dictF)  
dfNames = pd.DataFrame.from_dict(CompoundNames)  
hackathonToxdata2 = pd.concat([dfNames, ViableCellCountdf], axis=1)  
print(hackathonToxdata2)
```

	particle	ViableCellCount
0	NaN	NaN
1	NP00375	NaN
2	NP00262	1.422836
3	NP00266	-0.401261
4	NP00269	2.434066
5	NP00255	-0.040607
6	NP00256	0.084395
7	NP00257	0.057643
8	NP00258	0.284920
9	NP00259	-0.450942
10	NP00260	0.525511
11	NP00254	-0.213756
12	NP00193	0.247470
13	NP00192	0.818199
14	NP00282	-7.304242
15	NP00283	-9.238114
16	NP00214	-16.067656
17	NP00214	-16.067656

```
#Removing the 2 top rows as they do not contain values  
hackathonToxdata2=hackathonToxdata2.drop(hackathonToxdata2.index[[0,1]])  
hackathonToxdata2.index=range(len(hackathonToxdata2))  
Yall=hackathonToxdata2['ViableCellCount']  
Yall=Yall.drop(Yall.index[[13]])  
Yall.index=range(len(Yall))  
Yall
```

0	1.422836
1	-0.401261
2	2.434066

```
3    -0.040607
4     0.084395
5     0.057643
6     0.284920
7    -0.450942
8     0.525511
9    -0.213756
10    0.247470
11    0.818199
12   -7.304242
13  -16.067656
14  -16.067656
```

Name: ViableCellCount, dtype: float64

```
#Transforming the Viable Cells Count values into binned values (to be used
in classification)
YallBins=pd.cut(Yall, [ -16.086,-1.266,0,2.435])
print(YallBins.dtypes)
#YallBins
YallBins.dtypes
```

category

```
CategoricalDtype(categories=[(-16.086, -1.266], (-1.266, 0.0], (0.0, 2.435]],
                    ordered=True)
```

YallBins

```
0    (0.0, 2.435]
1    (-1.266, 0.0]
2    (0.0, 2.435]
3    (-1.266, 0.0]
4    (0.0, 2.435]
5    (0.0, 2.435]
6    (0.0, 2.435]
7    (-1.266, 0.0]
8    (0.0, 2.435]
9    (-1.266, 0.0]
10   (0.0, 2.435]
11   (0.0, 2.435]
12   (-16.086, -1.266]
13   (-16.086, -1.266]
14   (-16.086, -1.266]
```

Name: ViableCellCount, dtype: category

```
Categories (3, interval[float64]): [(-16.086, -1.266] < (-1.266, 0.0] < (0.0, 2.435]]
```

```
#Encoding the ViableCellCount bins as integers
from sklearn                import metrics
from sklearn.linear_model   import LogisticRegression
```

```

from sklearn import preprocessing
from sklearn import utils
training_scores_Y=YallBins
lab_enc = preprocessing.LabelEncoder()
training_scores_encoded = lab_enc.fit_transform(training_scores_Y)
print(training_scores_encoded)
print(training_scores_Y.dtypes)
print(utils.multiclass.type_of_target(training_scores_Y))
print(utils.multiclass.type_of_target(training_scores_encoded))

```

```
[2 1 2 1 2 2 2 1 2 1 2 2 0 0 0]
```

category

unknown

multiclass

```

#Preprocessing X values
Xall_BIO=df
Xall_BIO=Xall_BIO.drop(['_DLS SD (nm)', '_PDI SD', '_Zeta SD (mV)', '_Electrophoretic mobility SD (µmcm/Vs)', '_TEM (nm)', '_TEM SD (nm)', '_TEM Shortest (nm)', '_TEM Shortest SD (nm)', '_TEM Longest (nm)', '_TEM Longest SD (nm)', '_STEM (nm)', '_STEM SD (nm)', '_BET (m^2/g)', '_BET SD(m^2/g)', '_Energy Band Gap (eV)'],axis=1)
Xall_BIO=Xall_BIO.drop(Xall_BIO.index[[13]])
Xall_BIO=Xall_BIO.drop(['Id', 'Particle'],axis=1)
Xall_BIO.index=range(len(Xall_BIO))
Xall_BIO

```

	..Type	..Coating	..Type of Coating	..Surface Modification	..Shape	..Nominal Size (nm)	..DLS (nm)	..PDI	..Zeta (mV)	..Electrophoretic mobility (µmcm/Vs)	..Geometric Surface Area (nm <sup>2</sup> )
0	Si	aminoacid	neutral	None	Spherical	20	28.90	0.110	-21.80	0.0879	1206.874234
1	Si	aminoalkyl	positive	None	Spherical	20	23.48	0.172	25.90	2.0300	1206.874234
2	Si	aminoacid	negative	None	Spherical	20	25.24	0.174	-31.70	-2.4840	1206.874234
3	Ti	uncoated	uncoated	None	Spherical	10	1325.00	0.277	20.50	1.6050	180.271745
4	Ti	PVP	neutral	None	Square/Spherical/Rods	10	3185.00	0.240	17.30	1.3610	224.541375
5	Ti	F127	neutral	None	Square	10	4391.00	0.172	5.30	0.4158	325.380000
6	Ti	AA040	neutral	None	Square	10	3109.00	0.177	17.10	1.3440	1159.260000
7	Ti	uncoated	uncoated	hydrophobic rutile	Nanorods	20	1609.00	0.895	23.00	1.8040	2828.925645
8	Ti	uncoated	uncoated	hydrophilic rutile	Nanorods	20	403.00	0.810	25.30	1.9810	4404.764228
9	Ti	uncoated	uncoated	None	Spherical/ Square	70	172.40	0.183	41.00	3.2170	3532.494233
10	Ce	uncoated	uncoated	None	Spherical	20	269.50	0.291	32.90	2.3810	84.948665
11	Ce	uncoated	uncoated	None	Cubic	33	257.70	0.510	56.40	4.0850	1754.460000
12	Zn	uncoated	uncoated	None	Various shapes	150	411.80	0.239	-20.30	-1.5940	35155.320080
13	Ag	uncoated	uncoated	None	Spherical	15	27.36	0.471	-5.52	-0.3994	1398.217746
14	Ag	uncoated	uncoated	None	Spherical	15	27.36	0.471	-5.52	-0.3994	1398.217746

```

#Encode the categories into numerical values
from sklearn import preprocessing
LE = preprocessing.LabelEncoder()
LE.fit(Xall_BIO['..Type'])
Xall_BIO['..Type']= LE.transform(Xall_BIO['..Type'])

```

```
LE.fit(Xall_BIO['..Coating'])
Xall_BIO['..Coating']= LE.transform(Xall_BIO['..Coating'])

LE.fit(Xall_BIO['..Type of Coating'])
Xall_BIO['..Type of Coating']= LE.transform(Xall_BIO['..Type of Coating'])

LE.fit(Xall_BIO['..Shape'])
Xall_BIO['..Shape']= LE.transform(Xall_BIO['..Shape'])

Xall_BIO['..Surface Modification']=Xall_BIO['..Surface
Modification'].astype(str)
LE.fit(Xall_BIO['..Surface Modification'])
Xall_BIO['..Surface Modification']= LE.transform(Xall_BIO['..Surface
Modification'])

#One last look at the X and Y data
training_scores_encodedPD
```

ViableCellBins	
0	2
1	1
2	2
3	1
4	2
5	2
6	2
7	1
8	2
9	1
10	2
11	2
12	0
13	0
14	0

Xall\_BIO

..Type	..Coating	..Type of Coating	..Surface Modification	..Shape	..Nominal Size (nm)	..DLS (nm)	..PDI	..Zeta (mV)	..Electrophoretic mobility (µmcm/Vs)	..Geometric Surface Area (nm <sup>2</sup> )	
0	2	3	1	0	2	20	28.90	0.110	-21.80	0.0879	1206.874234
1	2	4	2	0	2	20	23.48	0.172	25.90	2.0300	1206.874234
2	2	3	0	0	2	20	25.24	0.174	-31.70	-2.4840	1206.874234
3	3	5	3	0	2	10	1325.00	0.277	20.50	1.6050	180.271745
4	3	2	1	0	5	10	3185.00	0.240	17.30	1.3610	224.541375
5	3	1	1	0	4	10	4391.00	0.172	5.30	0.4158	325.380000
6	3	0	1	0	4	10	3109.00	0.177	17.10	1.3440	1159.260000
7	3	5	3	2	1	20	1609.00	0.895	23.00	1.8040	2828.925645
8	3	5	3	1	1	20	403.00	0.810	25.30	1.9810	4404.764228
9	3	5	3	0	3	70	172.40	0.183	41.00	3.2170	3532.494233
10	1	5	3	0	2	20	269.50	0.291	32.90	2.3810	84.948665
11	1	5	3	0	0	33	257.70	0.510	56.40	4.0850	1754.460000
12	4	5	3	0	6	150	411.80	0.239	-20.30	-1.5940	35155.320080
13	0	5	3	0	2	15	27.36	0.471	-5.52	-0.3994	1398.217746
14	0	5	3	0	2	15	27.36	0.471	-5.52	-0.3994	1398.217746

```
clf = LogisticRegressionCV(cv=8,
random_state=0,multi_class='auto').fit(Xall_BIO,training_scores_encodedPD
clf.predict(Xall_BIO)
clf.score(Xall_BIO,training_scores_encodedPD)
```

0.8

```
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
import matplotlib.pyplot as plt
from sklearn import svm, datasets
from sklearn.metrics import confusion_matrix
from sklearn.utils.multiclass import unique_labels
```

```
def plot_confusion_matrix(y_true, y_pred, classes,
                           normalize=False,
                           title=None,
                           cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if not title:
        if normalize:
            title = 'Normalized confusion matrix'
        else:
            title = 'Confusion matrix, without normalization'

    # Compute confusion matrix
    cm = confusion_matrix(y_true, y_pred)
    # Only use the labels that appear in the data
    classes = Ynames
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
```

```
else:
    print('Confusion matrix, without normalization')

print(cm)

fig, ax = plt.subplots()
im = ax.imshow(cm, interpolation='nearest', cmap=cmap)
ax.figure.colorbar(im, ax=ax)
# We want to show all ticks...
ax.set(xticks=np.arange(cm.shape[1]),
       yticks=np.arange(cm.shape[0]),
       # ... and label them with the respective list entries
       xticklabels=classes, yticklabels=classes,
       title=title,
       ylabel='True label',
       xlabel='Predicted label')

# Rotate the tick labels and set their alignment.
plt.setp(ax.get_xticklabels(), rotation=45, ha="right",
         rotation_mode="anchor")

# Loop over data dimensions and create text annotations.
fmt = '.2f' if normalize else 'd'
thresh = cm.max() / 2.
for i in range(cm.shape[0]):
    for j in range(cm.shape[1]):
        ax.text(j, i, format(cm[i, j], fmt),
                ha="center", va="center",
                color="white" if cm[i, j] > thresh else "black")
fig.tight_layout()
return ax
```

```
y_pred_test=clf.predict(Xall_BIO)
y_test=training_scores_encodedPD
Ynames=['0', '1', '2']

np.set_printoptions(precision=2)

# Plot non-normalized confusion matrix
plot_confusion_matrix(y_test, y_pred_test, classes=Ynames,
                      title='Confusion matrix, without normalization')

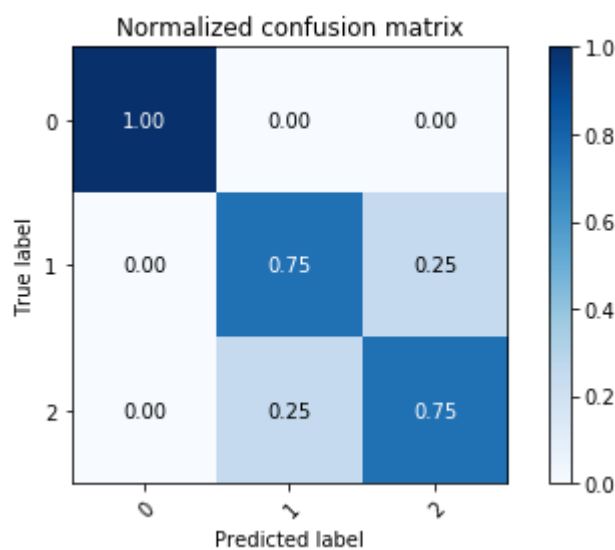
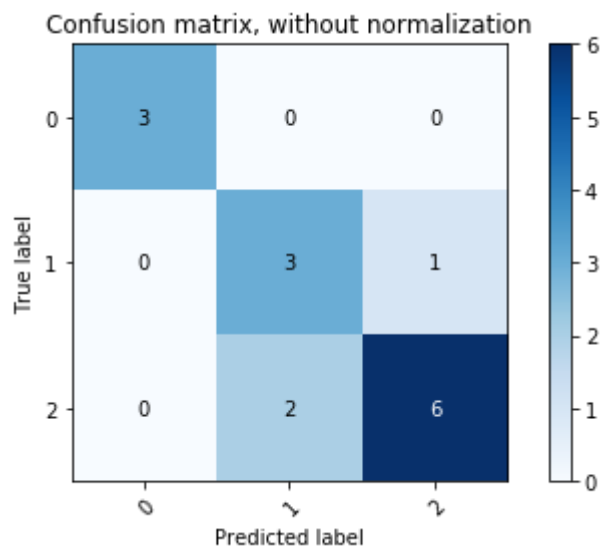
# Plot normalized confusion matrix
plot_confusion_matrix(y_test, y_pred_test, classes=Ynames, normalize=True,
                      title='Normalized confusion matrix')

plt.show()
#print("Accuracy -TRAIN:", metrics.accuracy_score(y_train, y_pred))
print("Accuracy :", metrics.accuracy_score(y_test, y_pred_test))
```

Confusion matrix, without normalization

```
[[3 0 0]
 [0 3 1]]
```

```
[0 2 6]]  
Normalized confusion matrix  
[[1.  0.  0. ]  
 [0.  0.75 0.25]  
 [0.  0.25 0.75]]
```



Accuracy : 0.8

```
from jaqpotpy import Jaqpot  
jaqpot = Jaqpot("https://api.jaqpot.org/jaqpot/services/")  
jaqpot.request_key_safe()  
url=jaqpot.deploy_pipeline(clf,Xall_BIO,training_scores_encodedPD, "Jaqpot+Biomax: Hackathon example", "Logistic Regression", "linearmodel")  
url
```



2019-06-21 11:41:33,087 - INFO - Model with id: 5mph5QzpkCeu3mfC2Gcm created. Please visit the application to proceed

'5mph5QzpkCeu3mfC2Gcm'

```
dfJQ_Biomax, predicts_Biomax = jaqpot.predict(Xall_BIO, modelId=url)
dfJQ_Biomax
```

	..Nominal Size (nm)	..Type of Coating	..Geometric Surface Area (nm <sup>2</sup> )	..Surface Modification	..Zeta (mV)	..Shape	ViableCellBins	..PDI	..DLS (nm)	..Electrophoretic mobility (µmcm/Vs)	..Coating	..Type
0	20	1	1206.874234	0	-21.80	2	2	0.110	28.90	0.0879	3	2
1	20	2	1206.874234	0	25.90	2	2	0.172	23.48	2.0300	4	2
10	20	3	84.948665	0	32.90	2	2	0.291	269.50	2.3810	5	1
11	33	3	1754.460000	0	56.40	0	1	0.510	257.70	4.0850	5	1
12	150	3	35155.320080	0	-20.30	6	0	0.239	411.80	-1.5940	5	4
13	15	3	1398.217746	0	-5.52	2	0	0.471	27.36	-0.3994	5	0
14	15	3	1398.217746	0	-5.52	2	0	0.471	27.36	-0.3994	5	0
2	20	0	1206.874234	0	-31.70	2	2	0.174	25.24	-2.4840	3	2
3	10	3	180.271745	0	20.50	2	2	0.277	1325.00	1.6050	5	3
4	10	1	224.541375	0	17.30	5	2	0.240	3185.00	1.3610	2	3
5	10	1	325.380000	0	5.30	4	2	0.172	4391.00	0.4158	1	3
6	10	1	1159.260000	0	17.10	4	2	0.177	3109.00	1.3440	0	3
7	20	3	2828.925645	2	23.00	1	2	0.895	1609.00	1.8040	5	3
8	20	3	4404.764228	1	25.30	1	2	0.810	403.00	1.9810	5	3
9	70	3	3532.494233	0	41.00	3	1	0.183	172.40	3.2170	5	3