

ModelRX

Modelling for Prediction or Read Across

Led by Harry Sarimveis (NTUA)
NTUA, UU, JGU/IST

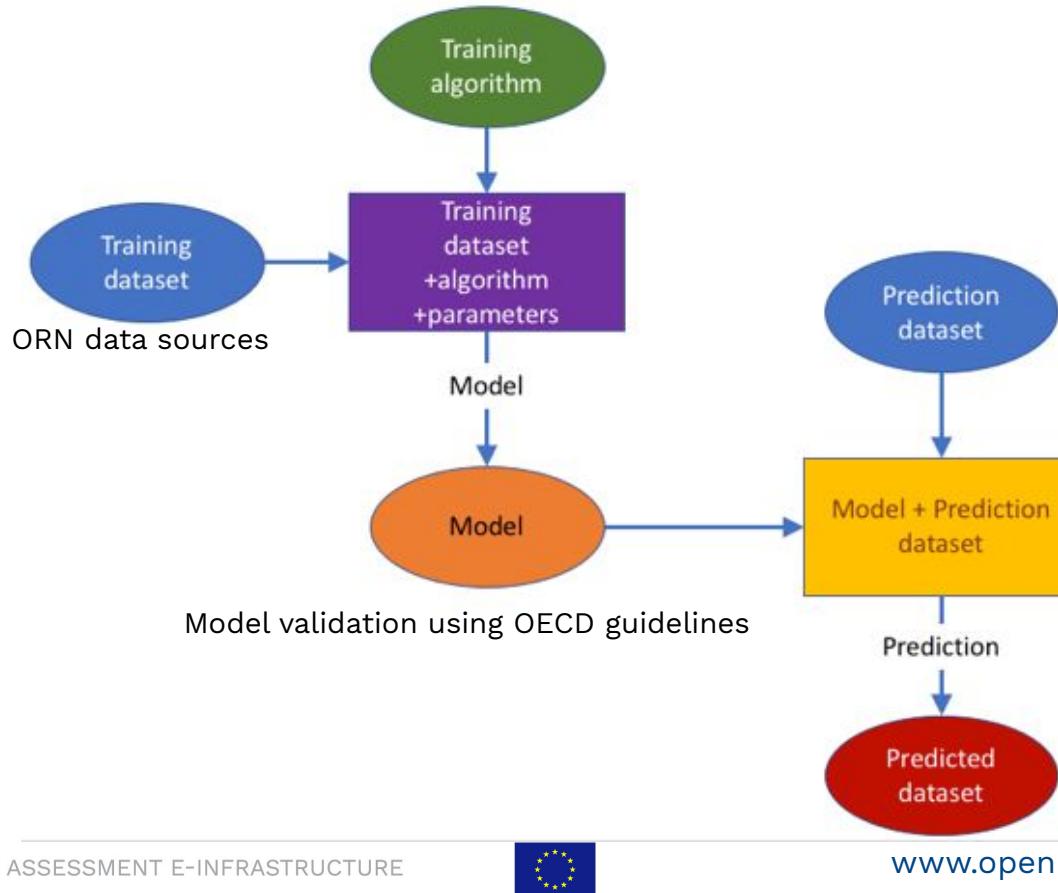
OpenRiskNet: Open e-Infrastructure to Support Data Sharing, Knowledge Integration and *in silico* Analysis and Modelling in Risk Assessment
Project Number 731075



Case Study objective

- support **similarity identification** in the DataCure case study (tools to calculate theoretical descriptors of substances)
- **fill gaps** in incomplete datasets
- use in-silico **predictive modelling approaches** (read-across, QSAR) to support final risk assessment

Workflow for prediction model



Jaqpot (NTUA)

The screenshot shows the Jaqpot web application interface. At the top, there's a navigation bar with the Jaqpot logo and a sun icon. Below it, a main header says "Train. Deploy. Share." with a subtitle "Jaqpot offers a new way to share your models". There are two main sections: one for "scikit learn" (with a blue and orange logo) and one for "R" (with a grey and blue R logo). Both sections have a "Deploy and share your models" button and a "Read the documentation" link. A "Soon to come" message is visible next to the R section.

modelling and analysis services

- scaling
- normalisation
- variable selection
- model validation
- algorithms for DoA
- QMRF reports



localhost

sus-single-web-C × lazar-workflow-get-mod- × jaqpot-descriptors.ipynb × jaqpot-model.ipynb × Python 3

Deploy the model to Jaqpot service

```
# URL to access the jaqpot service
jaqpot = Jaqpot("https://api-jaqpot.prod.openrisknet.org/jaqpot/services/")

# alternative link for Jaqpot services
#jaqpot = Jaqpot("https://api.jaqpot.org/jaqpot/services/")
```

Authentication

You can authenticate yourself either through username/password (have to enter everytime) or with API key obtained from the Jaqpot website (requires login and has validity for a limited time - does not have to be entered everytime)

```
PEA7CNVRjLs-CANi0zAuBzu-m0GKN8E4wYEL5VMDDUmKxv0RFh0itlxawouMzSwqhBPmb-5sKg_rL_p57dACh5Jc4
2019-10-22 18:14:23,710 - INFO - api key is set
```

From model to web service in 1 line of code

```
# deploy the model to jaqpot
url = jaqpot.deploy_linear_model(model, X_rfe, Y[['True']], title="OpenRiskNet/ModelRX", de
```

Lazar (JGU)

A modular framework for predictive toxicology.

in silico toxicity

<https://lazar.prod.openrisknet.org/api/api.json>

Explore

Lazar REST Service 1.4.0 OAS3

<https://lazar.prod.openrisknet.org/api/api.json>

REST API webservice for lazaz.

lazar (lazy structure–activity relationships) is a modular framework for predictive toxicology.

in silico toxicology gmbh - Website
Send email to in silico toxicology gmbh
GNU GENERAL PUBLIC LICENSE
See also documentation

Documentation

[See curl examples for a typical workflow.](#)

Lazar Graphical User Interface

[Lazar \(GUI\)](#)

Servers

<https://lazar.prod.openrisknet.org> - The API server

Similar to the read across procedure in toxicological risk assessment, lazaz creates local QSAR models for each compound to be predicted.

in silico toxicity OpenRiskNet

lazar toxicity predictions

Problems, bugs, ideas for improvements ? Please report at our [issue tracker](#), check out the [FAQ](#) page or send us an email. [version: 1.3.1]

Please cite DOI [10.3389/zenodo.10.3389](https://doi.org/10.3389/zenodo.10.3389) in scientific publications.

1. Draw a chemical structure

JSME Molecular Editor by Peter Ertl and Bruno Bienfait



WEKA services (JGU)

JGU weka-rest-ui <https://jguweka.prod.openrisknet.org/openapi/openapi.json> [Explore](#)

JGU WEKA REST Service 0.5.0-OAS3 OAS3

<https://jguweka.prod.openrisknet.org/openapi/openapi.json>

RESTful API Webservice to WEKA Machine Learning Algorithms. This webservice provides an [OpenRiskNet](#) compliant REST interface to machine learning algorithms from the WEKA Java Library. This application is developed by the [Institute of Computer Science](#) at the Johannes Gutenberg University Mainz.

OpenRiskNet is funded by the European Commission GA 731075. WEKA is developed by the [Machine Learning Group](#) at the University of Waikato.

See [Documentation](#), [Issue Tracker](#) and [Code](#) at Github.

Data Mining Group JGU Mainz - Website
Send email to Data Mining Group JGU Mainz
GNU General Public License 3
JGU WEKA REST Service Documentation on GitHub

algorithm

▼

GET /algorithm Get a list of algorithms.

POST /algorithm/BayesNet REST interface to the WEKA BayesNet classifier.

POST /algorithm/BayesNet/adaboost REST interface to the WEKA AdaBoost M1 with BayesNet classifier.

POST /algorithm/BayesNet/bagging REST interface to the WEKA Bagging with BayesNet classifier.

The WEKA modelling web-services by JGU provide an OpenRiskNet compliant REST interface to the machine learning algorithms from the WEKA Java Library.

CplogD, metpred and LTKB predictions (UU)

- Cplogd (UU): ([UI](#))
- Metpred (UU) ([UI](#))
- LTKB (UU): Classification for LTKB data ([No vs most](#), [No vs less](#), [No less vs most](#))
- Blood-brain-barrier-penetration (UU): (API)
[Venn-ABERS model](#) [Classification model](#)

The screenshot shows the cpLogD web application. At the top, there's a toolbar with various drawing and selection tools. Below it is a vertical element with letters C, N, O, S, F, Cl, Br, I. The main area has two tabs: "cpLogD - confidence predictor for logD" (selected) and "cpLogD - confidence predictor for logP". The "cpLogD - confidence predictor for logD" tab contains instructions: "Draw your molecule in the editor, the prediction underneath will update as you draw." It also includes a detailed description of the model's training data and a reference to a paper by Marie Lapins et al. at the bottom. A "Confidence:" slider is shown at the bottom left, set to 0.95. The "cpLogD" logo is at the top right.

This screenshot shows the MetPred API documentation using the Swagger interface. The top bar shows the URL <https://metpred.prod.openrisknet.org/v2/openapi.json>. The main content area is titled "MetPred 0.1.0" and includes a brief description of the service as a RESTful webserver. It lists the "Improving Site-of-Metabolism Predictions by Defining Reaction Centers using Reaction SMARTS Patterns" paper by Ola Sjöth, Lars Carlsson, Jonathan Alvarsson, Staffan Arvidsson Mc Shane, Catrin Hasselgren, Lucja Peterlin Masic, Wesley Schaal, and Scott Boyer. Below this are sections for "Submitted" and "PharmBio". The "Predict" section contains two API endpoints: "GET /predictionImage" for predicting a single compound from an image, and "GET /prediction" for predicting a single compound.

This screenshot shows the Blood Brain Barrier Penetration API documentation using the Swagger interface. The top bar shows the URL <http://blood-brain-barrier-penetration-cvap-cpsign.prod.openrisknet.org/openapi.json>. The main content area is titled "Blood Brain Barrier Penetration 0.1.0" and includes a base URL: <http://blood-brain-barrier-penetration-cvap-cpsign.prod.openrisknet.org/v1>. It describes the "A Venn-ABERS model built by CPSign for predicting molecules" and credits the developer, website, and copyright information to Aros Bio. The "Predict" section contains two API endpoints: "GET /predict" for making a prediction on a given molecule, and "GET /predictImage" for making a prediction image for the given molecule.

Jupyter & Squonk (IM)

A screenshot of a Jupyter Notebook interface. On the left, there's a sidebar with 'Files' (containing 'Jagpot API', 'RestAPI_2_ORNL.ipynb', and 'RestAPI_2.ipynb'), 'Commands', 'Cell Tools', and 'Tabs'. The main area shows two code cells. Cell 1 contains Python code for logging in to a REST API and validating tokens. Cell 2 shows the output of the validation. The status bar at the bottom indicates 'In [1]: responseValidation'.

```
import requests
# Getting token
headers = {
    'Content-Type': 'application/x-www-form-urlencoded',
    'Accept': 'application/json',
    'Authorization': ''
}

data = {
    'username': 'guest',
    'password': 'guest'
}

responseLogin = requests.post('http://jagpot.org:8888/jagpot/services/aa/login', headers=headers, data=data)
responseLogin
import json

todos = json.loads(responseLogin.text)
token=todos["authToken"]
print(token)

#Validating token
headers = {
    'Content-Type': 'application/json',
    'Accept': 'application/json',
    'Authorization': 'Bearer ' + token,
}
responseValidation = requests.post('http://jagpot.org:8888/jagpot/services/aa/validate/accessToken', headers=headers)
responseValidation
```

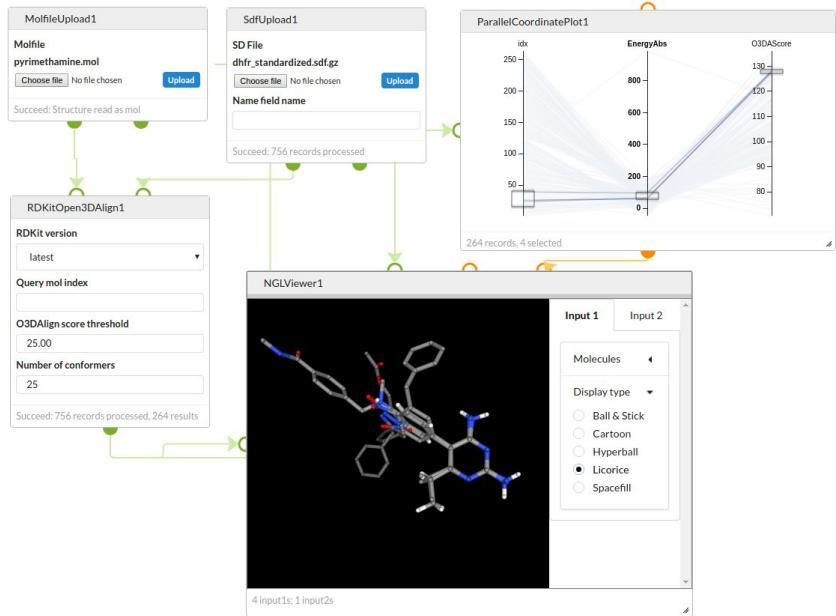
In [1]: responseValidation

Out [1]: Response [200]

In [2]: responseValidation

Jupyter notebooks infrastructure

Jupyter Notebooks ([UI](#))
Squonk Computational Notebook ([UI](#))



Squonk Computational notebook



Consensus modelling (EwC)

File Edit View Run Kernel Tabs Settings Help

+

/ ... / ModelRX / Blood-brain barrier - Lazar /

Name	Last Modified
compounds.csv	18 hours ago
lazar_workflow_workshop.postman_col...	18 hours ago
lazar-api-model.ipynb	18 hours ago
lazar-predict.ipynb	18 hours ago
lazar-workflow-get-model-training-da...	18 hours ago
lazar-workflow-get-model-validation-...	18 hours ago
lazar-workflow.ipynb	18 hours ago
Lazar.md	18 hours ago
predictions_Lazar.csv	18 hours ago

consensus-single-web-C X lazar-workflow-get-mod X jaqpot-descriptors.ipynb X jaqpot-model.ipynb X

Python 3

```
ps = axs[1].bar(1, delta["JLW"], width, bottom=ps1["JLW"], color=cOLORnonpenetrating)

axs[1].set_xticks([0, 1])
axs[1].set_xticklabels(['Yager', 'Dempster'])
axs[1].set_xlim(-0.5, len(models) - 0.5)
axs[1].set_xlabel('')
axs[1].set_ylabel('')
axs[1].set_yticklabels('')
axs[1].set_ylim(0, 1)
axs[1].set_title('Consensus predictions')

fig.legend((p1[0], p2[0], p3[0]), ('penetrating', 'uncertainty', 'non-penetrating'), bbox_t
fig.show()
```

The figure consists of two side-by-side bar charts. The left chart, titled 'Individual model predictions', shows four bars for CPSign, Lazar, Yager, and Dempster. The right chart, titled 'Consensus predictions', shows four bars for the same models. A legend indicates three categories: 'penetrating' (red), 'uncertainty' (green), and 'non-penetrating' (blue). In the individual predictions, CPSign and Lazar show high red bars (~0.95). Yager and Dempster show lower red bars (~0.85). In the consensus predictions, the bars for all models are significantly shorter, with CPSign and Lazar reaching near 1.0, while Yager and Dempster reach ~0.95.