OpenRiskNet

RISK ASSESSMENT E-INFRASTRUCTURE

Case Study

Modelling for Prediction or Read Across [ModelRX]

SUMMARY

A training data set will be obtained from an OpenRiskNet data source. The model has then to be trained with OpenRiskNet modelling tools and the resulting model has to be packaged into a container, documented and ontologically annotated. The model will be validated using OECD guidelines. Finally, a prediction can be run.

References (Jennings et al. 2018).

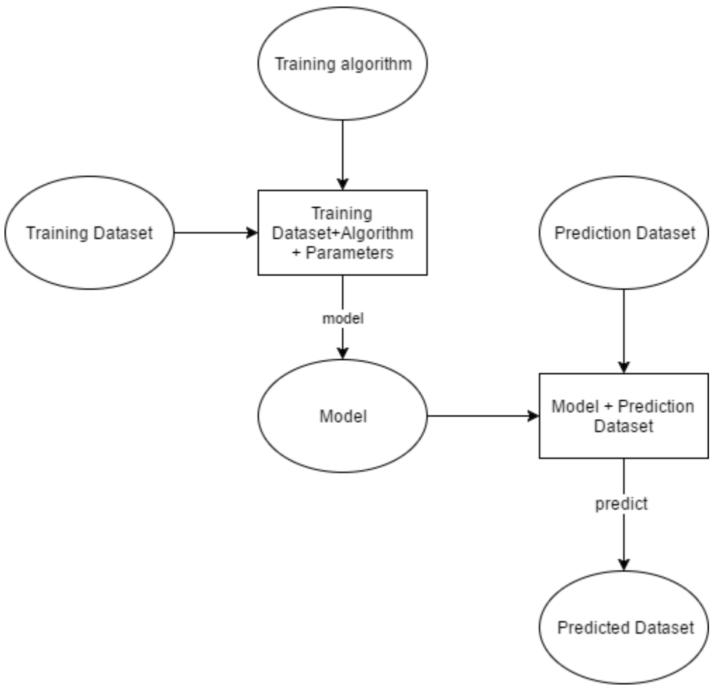


Figure 1. Building and using a prediction model workflow

DESCRIPTION

Implementation team

CS leader	Team
Harry Sarimveis (NTUA)	NTUA, JGU, UU

Case Study objective

The objectives of this case study are: support similarity identification in the DataCure case study (by providing tools for calculating theoretical descriptors of substances), fill gaps in incomplete datasets and use *in-silico* predictive modelling approaches (read-across, QSAR) to support final risk assessment.

Risk assessment framework

The ModelRX case study contributes in two tiers (Berggren et al. 2017):

- On the one hand, provides computational methods to support suitability assessment of existing data and identification of analogues (Tier 0);
- Secondly, it provides predictive modelling functionalities, which are essential in the field of final risk assessment (Tier 2).

DEVELOPMENT

Use Cases Associated

The ModelRX case study is associated with UC2 - Building and using a prediction model including the pseudocode[1].

Databases and tools

JaqPot Quattro (NTUA), CPSign (UU), JGU WEKA Rest service (JGU), Lazar (JGU/IST).

Technical implementation

From the developer's perspective, the Services used for this case study are deployed following the general steps that have been agreed for developing the OpenRiskNet infrastructure. Each application is deployed and delivered within containers. Docker is used as the basic engine for the containerisation of the applications. Above that a container orchestration tool Openshift, is used for the management of the containers and the services. Openshift provides many different options for deploying applications. Some recipes and examples have been documented in the OpenRiskNet GitHub page[2].

When an application is deployed, a service discovery mechanism is responsible for discovering the most suitable services for each application. Based upon the OpenAPI specification each API should be deployed with the swagger definition. This swagger file should then be integrated with the Json-LD annotations as dictated by the json-ld specification. The discovery service mechanism parses the resulting json-ld and resolves the annotations into RDF triplets. These triplets can then be queried with SPARQL. The result of the query lets the user to know which services are responsible for making models or predictions. The documentation can be found via the swagger definition of each application. This way the services are assigned into the OpenRiskNet deployment and can be used and digested from end user applications and other services.

Specific to this case study, the technical implementation is based on rigorously defined modelling APIs based on the OpenTox standards[3]. Modelling APIs need a high level of integration into the OpenRiskNet ecosystem. Integration with the DataCure CS is vital. On the semantic interoperability layer, training datasets should be compatible with an algorithm and prediction datasets should be compatible with a prediction model. Additionally, the generated models and datasets need to be accompanied with semantic metadata on their life cycle, thus enforcing semantic enrichment of the dynamically created entities. Algorithms, models and predicted datasets are built as services, discoverable by OpenRiskNet discovery service. This is a step that should occur whenever an entity (algorithm, model, predicted dataset) is created.

[1] https://github.com/OpenRiskNet/home/wiki/UseCase-2-Pseudocode
[2] https://github.com/OpenRiskNet/home/tree/master/openshift
[3] http://www.opentox.net/opentox-api

From the user's point of view the steps that need to be taken in order to produce a model and use it for predictions are the following:

1. Selecting a training data set

The user will be able to choose from a variety of different OpenRiskNet compliant data sets already accessible through the discovery service. A **Dataset** should include, at a minimum:

- a dataset URI
- substances: substance URIs (each substance URI will be associated with a term from the ontology)
- features:
    - feature URIs (each feature URI will be associated with a term from the ontology)
    - values in numerical format
    - category (experimental/computed)
    - if computed, the URI of the model used to generate the values
    - Units

2. Selecting a (suitable) modelling algorithm

The user will be able to choose from a list of suitable algorithms. **Algorithms** should include at a minimum:

- algorithm URI
- title
- description
- algorithm type (regression/classification)
- default values for its parameters (where applicable)

3. Specifying parameters and Generating Predictive model

Once an algorithm has been selected, the user should define the endpoint, select the tuning parameters, (only if different values from the default ones are desired) and run the algorithm. The generated **Model** should contain, at a minimum:

- model URI
- title
- description
- the URI of the dataset that was used to create it
- the URIs of the input features
- the URI of the predicted feature
- values of tuning parameters

Possible extensions:

- Include services/APIs for validation of the generated model
- Provide mechanisms to pick out the best algorithm for a specific dataset (e.g. RRegrs)
- Include algorithms to calculate domain of applicability

4. Selecting a prediction data set

After the creation of a model, the user will be able to select a prediction dataset, which should meet all the requirements specified in (Chomenidis et al. 2017). This dataset will be tested for compatibility against the required features of the model in terms of feature URIs, i.e. the dataset should contain all the subset of features used to produce the model. Additional features are allowed, however they will be ignored.

5. Running predictive model on the prediction data set

The predictive model is applied on the prediction dataset to generate the predicted dataset, which should be compatible with the requirements specified in (Chomenidis et al. 2017). The predicted dataset augments the prediction dataset with all necessary information about the predicted feature:

- prediction feature URIs (each feature URI will be associated with a term from the ontology)
- values in numerical format
- category (computed)
- the URI of the model used to generate the values
- units

Examples of implementation are annexed below.

OUTCOMES

This case study suggests a workflow that produces semantically annotated predictive models that can be shared, tested, validated and eventually applied for predicting adverse effect of substances in a safe by design and/or risk assessment regulatory framework. OpenRiskNet provides the necessary functionalities that allows researchers and practitioners to easily produce and expose their models as ready-to use web applications. The OpenRiskNet e-infrastructure can serve as a central model repository in the area of predictive toxicology. For example, when a research group publishes a predictive model in a scientific journal, they can additionally provide the implementation of the model as a web service using the OpenRiskNet implementation. The produced models contain all the necessary metadata and ontological information to make them easily searchable by the users and define systematically and rigorously their domain of applicability. Most importantly, the produced resources are not just static representation of the models, but actual web applications where the users can supply the necessary information for query substances and receive the predictions for their adverse effects.

REFERENCES

Berggren, Elisabet, Andrew White, Gladys Ouedraogo, Alicia Paini, Andrea-Nicole Richarz, Frederic Y. Bois, Thomas Exner, et al. 2017. “Ab Initio Chemical Safety Assessment: A Workflow Based on Exposure Considerations and Non-Animal Methods.” *Computational Toxicology (Amsterdam, Netherlands)* 4 (November): 31–44.

Chomenidis, Charalampos, Georgios Drakakis, Georgia Tsiliki, Evangelia Anagnostopoulou, Angelos Valsamis, Philip Doganis, Pantelis Sopasakis, and Haralambos Sarimveis. 2017. “Jaqpot Quattro: A Novel Computational Web Platform for Modeling and Analysis in Nanoinformatics.” *Journal of Chemical Information and Modeling* 57 (9): 2161–72.

Gajewicz, Agnieszka, Nicole Schaeublin, Bakhtiyor Rasulev, Saber Hussain, Danuta Leszczynska, Tomasz Puzyn, and Jerzy Leszczynski. 2015. “Towards Understanding Mechanisms Governing Cytotoxicity of Metal Oxides Nanoparticles: Hints from Nano-QSAR Studies.” *Nanotoxicology* 9 (3): 313–25.

Jennings, Paul, Thomas Exner, Lucian Farcal, Noffisat Oki, Harry Sarimveis, Philip Doganis, Danyel Jennen, et al. 2018. “Final Definition of Case Studies (Deliverable 1.3),” November. https://doi.org/10.5281/zenodo.1479127.

ANNEX

Jaqpot (NTUA)

This example presents the use of the JaqPot services for creating a predictive model and using it for predictions. For better visualisation and use of the API, the swagger ui of the jaqpot API is recommended (https://api-jaqpot.prod.openrisknet.org/jaqpot/swagger/).

For this example we semantically annotated and uploaded through the Jaqpot API a dataset that contains physicochemical and image descriptors for 8 Metal oxide nanoparticles and the corresponding log2 transformation of LC50 HaCaT toxicity as the end-point. The data are publically available through publication Gajewicz et al., 2015 (Gajewicz et al. 2015). The dataset can be found with the id "o0TO3TMsMrFKFS". Its URI is (http://jaqpot-api-jaqpot.dev.openrisknet.org/jaqpot/services/dataset/o0TO3TMsMrFKFS).

The modelling process was performed using URI (http://jaqpot-api-jaqpot.dev.openrisknet.org/jaqpot/services/algorithm/{id}, where we supplied the id of the support vector machine algorithm (weka-svm3), the dataset URI, and the prediction feature of the dataset we need to predict (the dependend feature is under the URI `https://apps.ideaconsult.net/enmtest/property/TOX/UNKNOWN_TOXICITY_SECTION/logLC50/63A6D4C95CE54B1630770EB000899C3EECE8DFDF/2ced6b25-4648-3217-87eb-70595145c4b1`.

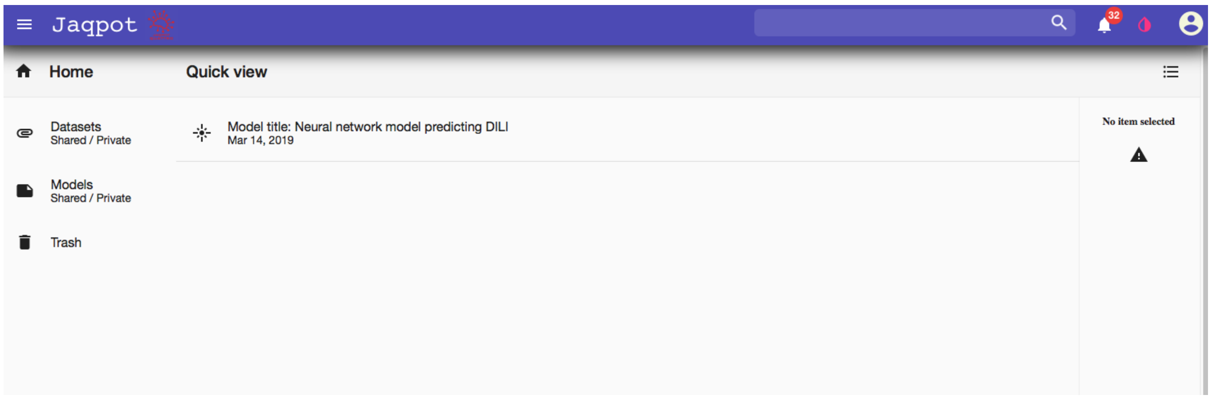
We also need to provide a title and a description of the model. We left the parameters field empty, which means that the default parameters of the algorithm will be used (optionally the tuning parameters of the algorithm can be modified). The API returns to the end user a Task ID, because the modelling procedure is performed in an asynchronous way.

At the URI http://jaqpot-api-jaqpot.dev.openrisknet.org/jaqpot/services/task/{id} we can monitor the status of the task. When the task is completed, the model id is in the response schema of the API, and can be found on the URI (http://jaqpot-api-jaqpot.dev.openrisknet.org/jaqpot/services/model/{id}). In our particular run, the following model id was produced "`crk7TnF4y127foOrcBQJ`"

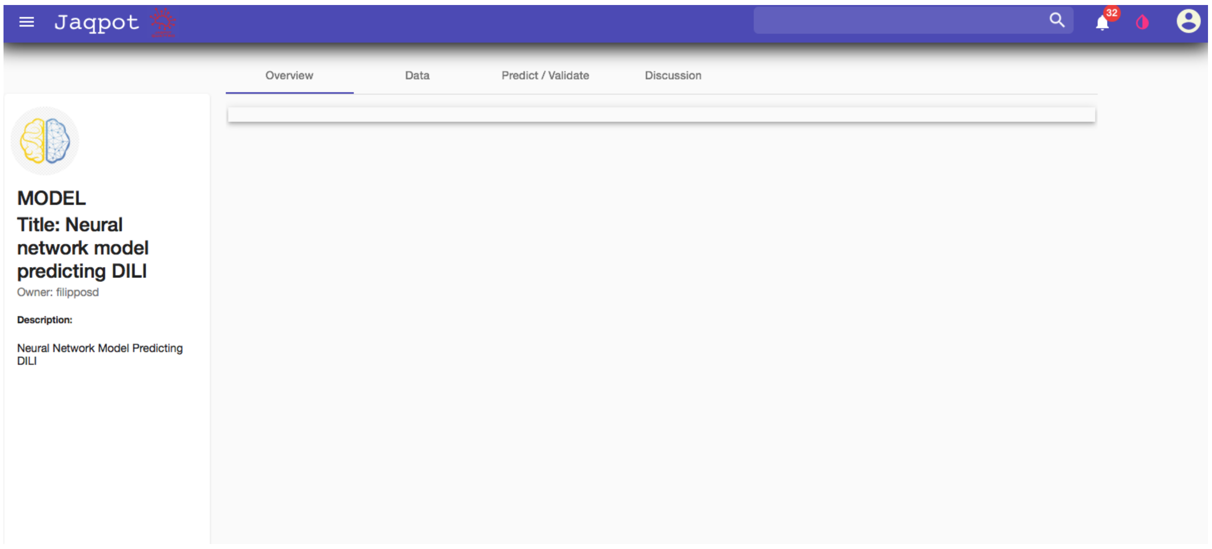
This model then can be used for making predictions for a new dataset using the URI http://jaqpot-api-jaqpot.dev.openrisknet.org/jaqpot/services/model/{id}, where we supply the id of the model and the URI of the dataset to be predicted. This operation returns a new dataset containing the model predictions. To test the prediction service we provided the training dataset and received the dataset found on the URI: "`http://jaqpot-api-jaqpot.dev.openrisknet.org/jaqpot/services/dataset/nWkoRVebPFChbaTbzVK3`"

Example using Jaqpot 5

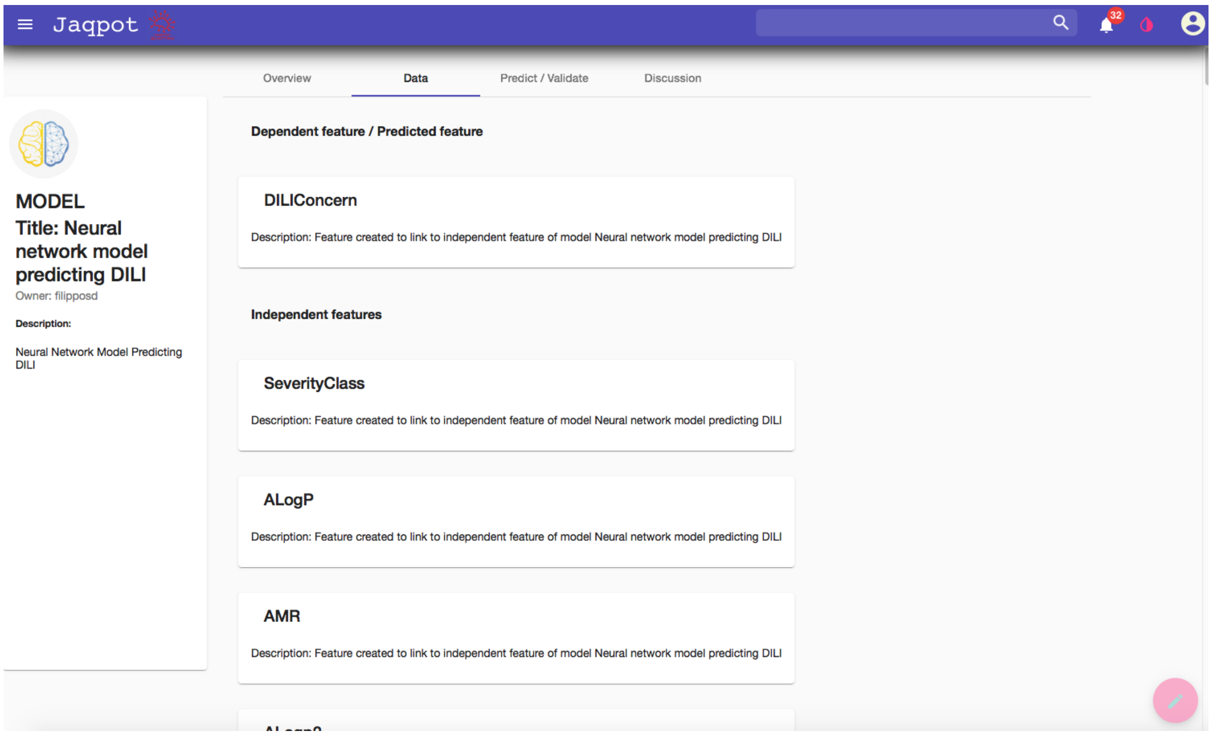
A model will be trained in a python notebook and the model will be uploaded to Jaqpot.



Model title: Neural network model predicting DILI
Mar 14, 2019



MODEL
Title: Neural network model predicting DILI
Owner: filipposd
Description:
Neural Network Model Predicting DILI



MODEL
Title: Neural network model predicting DILI
Owner: filipposd
Description:
Neural Network Model Predicting DILI

Dependent feature / Predicted feature

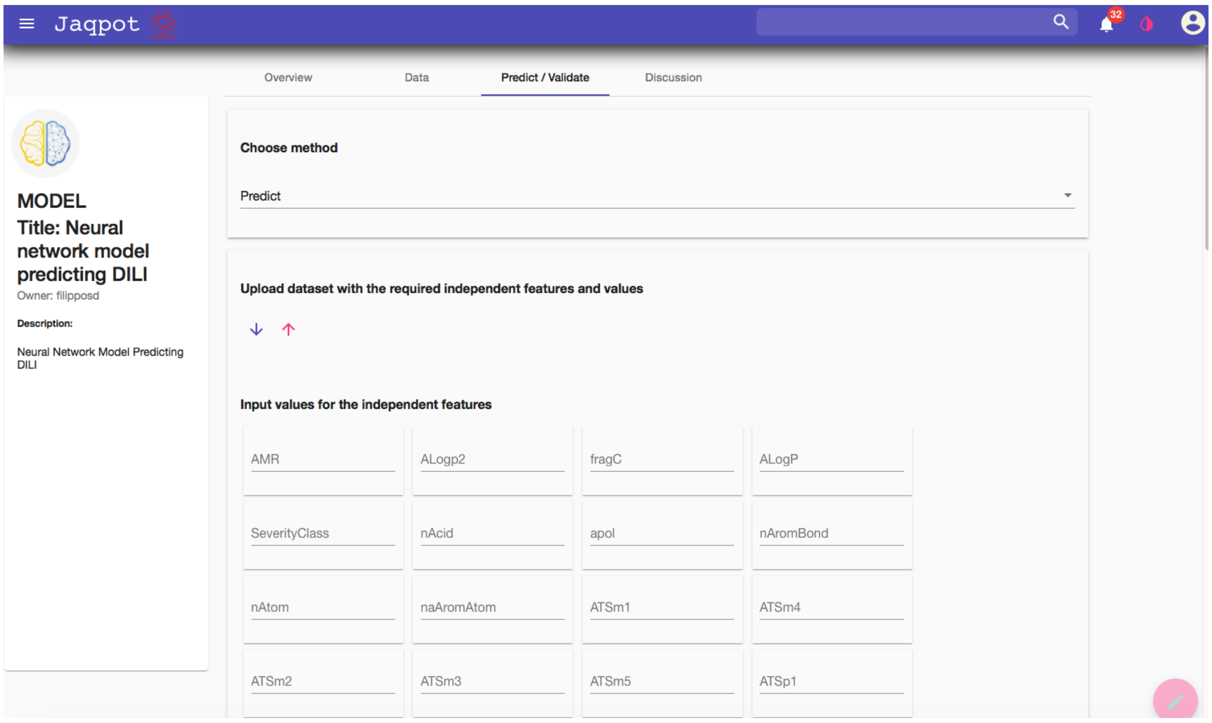
DILIConcern
Description: Feature created to link to independent feature of model Neural network model predicting DILI

Independent features

SeverityClass
Description: Feature created to link to independent feature of model Neural network model predicting DILI

ALogP
Description: Feature created to link to independent feature of model Neural network model predicting DILI

AMR
Description: Feature created to link to independent feature of model Neural network model predicting DILI



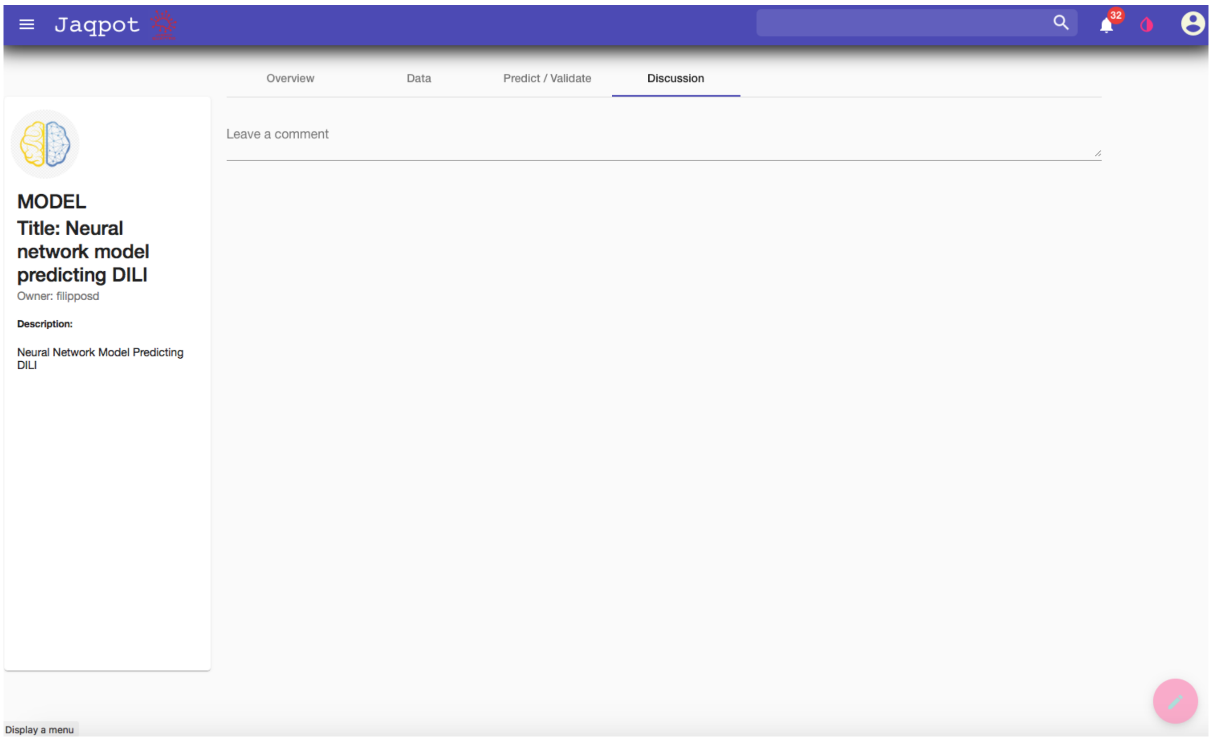
MODEL
Title: Neural network model predicting DILI
Owner: filipposd
Description:
Neural Network Model Predicting DILI

Choose method
Predict

Upload dataset with the required independent features and values

Input values for the independent features

AMR	ALogp2	fragC	ALogP
SeverityClass	nAcid	apol	nAromBond
nAtom	naAromAtom	ATSm1	ATSm4
ATSm2	ATSm3	ATSm5	ATSp1



Leave a comment

MODEL
Title: Neural network model predicting DILI
Owner: filipposd

Description:

Neural Network Model Predicting DILI