ChemML Documentation

Mojtaba Haghighatlari

Aug 03, 2019

CHEMML WRAPPER DOCUMENTATION

ChemML is a machine learning and informatics program suite for the analysis, mining, and modeling of chemical and materials data.

- source repository on github: https://github.com/hachmannlab/chemml

CHAPTER

ONE

CODE DESIGN:

ChemML is developed in the Python 3 programming language and makes use of a host of data analysis and ML libraries(accessible through the Anaconda distribution), as well as domain-specific libraries. The development follows a strictly modular and object-oriented design to make the overall code as flexible and versatile as possible.

The format of library is similar to the well known libraries like Scikit-learn. ChemML will be soon available via graphical user interface provided by [ChemEco](https://github.com/hachmannlab/chemeco). ChemEco is a general-purpose framework for data mining without coding. It also interfaces with many of the libraries that supply methods for the representation, preprocessing, analysis, mining, and modeling of large-scale chemical data sets.

CHAPTER

TWO

LATEST VERSION:

- to find the latest version and release history, click here: https://pypi.org/project/chemml/#history

CHAPTER

THREE

INSTALLATION AND DEPENDENCIES:

You can download ChemML from Python Package Index (PyPI) via pip.

```
pip install chemml --user -U
```

Here is a list of external libraries that will be installed with chemml:

- numpy
- pandas
- tensorflow
- keras
- scikit-learn
- matplotlib
- seaborn
- lxml

Since conda installation is not available for ChemML yet, we recommend installing rdkit and openbabel in a conda virtual environment prior to installing ChemML. For doing so, you need to follow the conda installer:

```
conda create --name my_chemml_env python=3.6
source activate my_chemml_env
conda install -c openbabel openbabel
conda install -c rdkit rdkit
pip install chemml
```

CHAPTER

FOUR

PYTHON 2 FANS:

The library was initially compatible with Python 2 and 3, and we still develop compatible codes. However, since the Python community and some of the dependencies are phasing out support for Python 2, we also removed it from the classifier languages. However, you should still be able to clone the ChemML repository from github and install the older versions of some of the dependencies that support Python 2, prior to using ChemML locally.

CHAPTER

FIVE

CITATION:

Please cite the use of ChemML as:

```
Main citation:

@article{chemml2019,
author = {Haghighatlari, Mojtaba and Vishwakarma, Gaurav and Altarawy, Doaa and␣
↪Subramanian, Ramachandran and Kota, Bhargava Urala and Sonpal, Aditya and Setlur,␣
↪Srirangaraj and Hachmann, Johannes},
journal = {ChemRxiv},
pages = {8323271},
title = {ChemML: A Machine Learning and Informatics Program Package for the Analysis,␣
↪Mining, and Modeling of Chemical and Materials Data},
doi = {10.26434/chemrxiv.8323271.v1},
year = {2019}
}


Other references:

@article{chemml_review2019,
author = {Haghighatlari, Mojtaba and Hachmann, Johannes},
doi = {https://doi.org/10.1016/j.coche.2019.02.009},
issn = {2211-3398},
journal = {Current Opinion in Chemical Engineering},
month = {jan},
pages = {51--57},
title = {Advances of machine learning in molecular modeling and simulation},
volume = {23},
year = {2019}
}

@article{Hachmann2018,
author = {Hachmann, Johannes and Afzal, Mohammad Atif Faiz and Haghighatlari, Mojtaba␣
↪and Pal, Yudhajit},
doi = {10.1080/08927022.2018.1471692},
issn = {10290435},
journal = {Molecular Simulation},
number = {11},
pages = {921--929},
title = {Building and deploying a cyberinfrastructure for the data-driven design of␣
↪chemical systems and the exploration of chemical space},
volume = {44},
year = {2018}
}
```

5.1 ChemML Wrapper Tutorial

. . . ::: ChemML Wrapper is currently only available in the version 0.4.* (Python 2.7) :::. . .

ChemML Wrapper carry out a workflow of operations hosted by different open-source or commercial libraries/software. The workflow is similar to a directed graph and it can be designed in a text file. An input file (configuration file) is required to run the ChemML Wrapper.

In this section we walk you through all the required steps to run an input file.

5.1.1 Step #1: prepare an input file

ChemML Wrapper requires an input file to configure and run an arbitrary workflow. This file must be a text file in any arbitrary format. If you want to create or modify an input file manually, see *Input File Manual* for more information. You can also create, modify, and even visualize an input file using the graphical interface, *Input File GUI*.

The input file consists of methods that are connected to each other and make a data mining workflow. Here you can see a simple input script:

```
## (Enter,datasets)
    << host = chemml
    << function = load_xyz_polarizability
    >> coordinates 0
    >> polarizability 4
    >> polarizability 5

## (Represent,molecular descriptors)
    << host = chemml
    << function = CoulombMatrix
    >> 0 molecules
    >> df 1
    >> df 6

## (Store,file)
    << host = chemml
    << function = SaveFile
    << filename = CM_features
    >> 1 df

## (Store,figure)
    << host = chemml
    << function = SavePlot
    << kwargs = {'normed':True, 'bbox_inches':'tight'}
    << output_directory = plots
    << filename = performance
    >> 2 fig

## (Visualize,artist)
    << host = chemml
    << function = decorator
    << title = training performance
    << grid_color = g
    << xlabel = predicted polarizability (Bohr^3)
    << ylabel = calculated polarizability (Bohr^3)
    << grid = True
    << size = 12
```

(continues on next page)

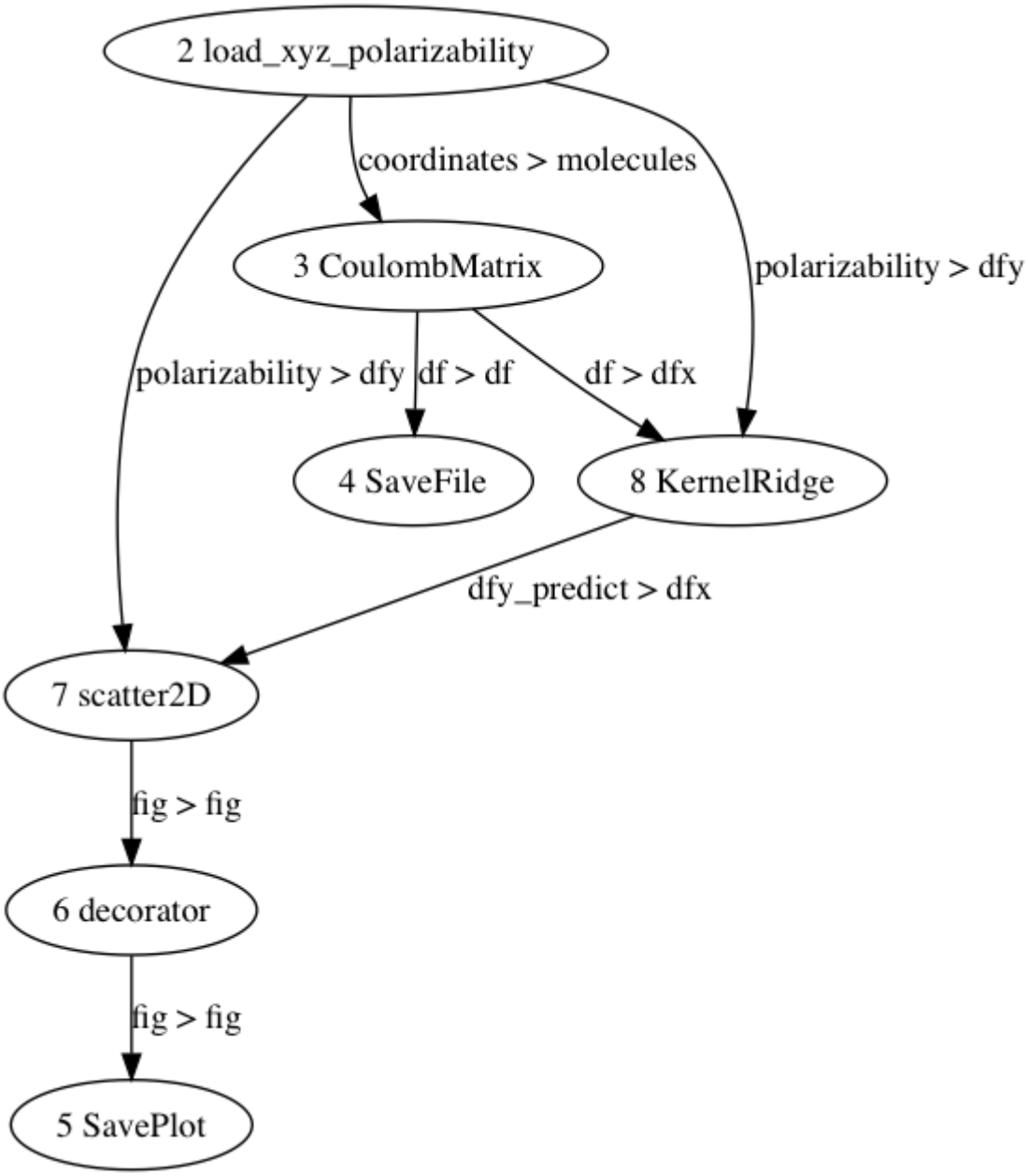
(continued from previous page)

```
    >> fig 2
    >> 3 fig

## (Visualize,plot)
    << host = chemml
    << function = scatter2D
    << y = 0
    << marker = o
    << x = 0
    >> fig 3
    >> 4 dfy
    >> 7 dfx

## (Model,regression)
    << host = sklearn
    << function = KernelRidge
    << func_method = fit
    << kernel = rbf
    >> 5 dfy
    >> 6 dfx
    >> dfy_predict 7
```

Let's create a text file with name 'tutorial.config' and copy the above input script into that file. If we copy this script into 'existing script' tab of the *Input File GUI*, a graphical visualization of the workflow will be presented as follow:



5.1.2 Step #2: run ChemML Wrapper

Now that we have prepared an input script/file, we can run it using any of the following ways.

method one: Running in the Terminal

Let's say you saved your script with the name 'tutorial.config' in your Desktop folder (name and format are both arbitrary). In the Terminal, navigate to the Desktop folder and enter the following command:

```
chemmlwrapper -i tutorial.config -o output_directory
```

method two: Running in any Python Interpreter

You can also run your input script with python codes as a python script or interactively, using:

```
# we assume you saved your script with the name 'tutorial.config' in your Desktop␣
↪folder
from cheml import wrapperRUN
wrapperRUN(INPUT_FILE = '/Desktop/tutorial.config', OUTPUT_DIRECTORY = 'output_
↪directory')

# or

script = """
            ## (Enter,datasets)
                << host = chemml
                << function = load_xyz_polarizability
                >> coordinates 0
                >> polarizability 4
                >> polarizability 5

            ## (Represent,molecular descriptors)
                << host = chemml
                << function = CoulombMatrix
                >> 0 molecules
                >> df 1
                >> df 6

            ## (Store,file)
                << host = chemml
                << function = SaveFile
                << filename = CM_features
                >> 1 df

            ## (Store,figure)
                << host = chemml
                << function = SavePlot
                << kwargs = {'normed':True, 'bbox_inches':'tight'}
                << output_directory = plots
                << filename = performance
                >> 2 fig

            ## (Visualize,artist)
                << host = chemml
                << function = decorator
                << title = training performance
                << grid_color = g
                << xlabel = predicted polarizability (Bohr^3)
                << ylabel = calculated polarizability (Bohr^3)
                << grid = True
                << size = 12
                >> fig 2
                >> 3 fig

            ## (Visualize,plot)
                << host = chemml
                << function = scatter2D
                << y = 0
                << marker = o
```

(continues on next page)

(continued from previous page)

```
                << x = 0
                >> fig 3
                >> 4 dfy
                >> 7 dfx

            ## (Model,regression)
                << host = sklearn
                << function = KernelRidge
                << func_method = fit
                << kernel = rbf
                >> 5 dfy
                >> 6 dfx
                >> dfy_predict 7

        """
wrapperRUN(INPUT_FILE = script, OUTPUT_DIRECTORY = 'output_directory')
```

Although you can run your script interactively, we recommend using the first method along with 'nohup' command to prevent Terminal disconnection from killing your job.

Note: in all these methods if your arbitrary name of ouput_directory already exists, chemML Wrapper automatically creates a folder with sequential number.

5.1.3 Step #3: check the output directory

After running ChemML Wrapper you can collect all your saved files by navigating to the output directory. These saved files can be any type of results and fugures from your workflow, or default files, e.g. error file, log file, a copy of input file, and citation file.

5.2 Input File Manual

An instruction to the ChemML Wrapper input file is provided here. The input file configures parameters and settings for running a data mining workflow using ChemML Wrapper.

Although you can generate an input file manually in a text editor, a graphical user interface is also available to facilitate this process (see *Input File GUI*).

5.2.1 Input File Overview

The input file structure corresponds to a directed graph, where nodes are computation **blocks** and represent any available method in the package (*Table of Contents*), and edges represent the flow of data between nodes.

To determine edges of the workflow graph, fixed input and output tokens are defined for each node and you should connect them using unique random integers.

Here you can see a pseudo computation block in the input file:

```
## Task
    << host = name        << function = name
    << parameter = value    << ...
```

(continues on next page)

(continued from previous page)

```
<< ...                    << ...
>> token id
>> id token
```

All the methods in the ChemML Wrapper are divided into the following 8 tasks:

- Enter
- Represent
- Prepare
- Model
- Search
- Mix
- Visualize
- Store

Each task in turn is divided into a number of subtasks to allow easy retrieval of the methods/functions.

Note: you always need an Enter method in your workflow to initialize the computation graph with some data.

5.2.2 Specific Characters

Only five specific characters (#, <, =, >, @) are defined and can be used in the input files.

- **Pound sign (#)**
  - Pound sign (#) determines a computation block.
  - It must be used to separate different blocks from each other.
  - A double pound sign (##) is for an active block and a single pound sign (#) keeps the block out of the graph.
  - No other specific signs can be in the same line as pound sign. Having a task name or any other text after the pound sign is arbitrary.
- **Less-than sign (<)**
  - Less-than sign (<) are used to pass the parameters' value.
  - The parameter's name comes right after this sign.
  - A Double less-than sign (<<) keeps a parameter in the block and single less-than sign (<) ignore the assigned value to that parameter.
  - If you are going to ignore a parameter value make sure it's not a required parameter.

Note: two parameters are mandatory in all the blocks:

- **host**: to specify the main library/dependency that you want to get the method from
- **function**: to specify the ChemML Wrapper method

You can find a comprehensive list of the available methods in the *Table of Contents*

- **Equals sign (=)**
  - Equals sign (=) is for setting the value of parameters.
  - It comes right after the parameter name and is followed by its value.
  - The parameter value should be in its python format.
- **Greater-than sign (>)**
  - Greater-than sign (>) is all you need to connect blocks to each other (to send outputs or to receive inputs).
  - To keep track of inputs and outputs you have to assign a unique identification (ID) number to input/output tokens.
  - The ID number can be any integer. The ChemML Wrapper will extract, and assign the sent output of one block to the received input of another block through these unique IDs.
  - Note that the tokens are predefined for each block and can be found in the *Table of Contents*.
  - To distinguish the send and receive actions you just need to switch the position of token and ID as described below:
    - **to send an output token:**

      >> token ID

      e.g. >> molfile 7
    - **to receive an input token:**

      >> ID token

      e.g. >> 7 molfile
- **At sign (@)**
  - At sign (@) can be used to get a parameter value from the input/output values.
  - It comes right after equals sign (=) and should be followed by one of the input tokens (e.g. parameter = @df).

Note: please note that the first three characters (#, <, >) are reserved and you should avoid using them in the parameter values.

5.2.3 General Rules

A few general restrictions:

- You are not allowed to have two different specific charecters in one line of input file (except '=' and '@' signs).
- The input tokens and output tokens of each block may be similar but they might not have similar values.
- Only one input per an input token can be received.
- You are allowed to send output tokens to as many input tokens of different block as you want.
- Avoid any type of short loop. A short loop will be made when inputs of any block_i are received from one or a set of blocks that they require an output of block_i.
- If you make a short loop any place inside your workflow your run will be aborted immediately.

5.3 Input File GUI

. . . ::: ChemML Wrapper is currently only available in the version 0.4.* (Python 2.7) ::: . . .

The ChemML wrapper's graphical user interface (GUI) facilitate the generation of the input files. You can run GUI locally in the Jupyter notebook with two lines of python code:

```
from cheml.notebooks import wrapperGUI
ui = wrapperGUI()
```

Requirements:

- **Jupyter notebook**
  - installation: http://jupyter.org/install.html
- **ipywidgets and widgetsnbextension**
  - installation: https://github.com/jupyter-widgets/ipywidgets/blob/master/docs/source/user_install.md
  - ipywidgets and widgetsnbextension will be installed accompanied by chemml via pip.
- **graphviz**
  - installation: https://graphviz.readthedocs.io/en/stable/manual.html#installation
  - Using graphviz library, you will see a graphical visualization of your project's workflow simultaneously.
  - graphviz will be installed accompanied by chemml via pip.

We recommend downloading and installing Anaconda for python 2. This way Jupyter will be installed automatically. If you are using anaconda and you plan to use a virtual environment, please run the following commands to install ChemML and wrapperGUI (the first and third lines are unnecessary if you have already installed chemml):

```
conda create --name my_chemml_env python=2.7
source activate my_chemml_env
pip install chemml --user -U

jupyter nbextension install --py widgetsnbextension --user
jupyter nbextension enable --sys-prefix --py widgetsnbextension
conda install -c conda-forge nb_conda_kernels
```

The last command installs nb_conda_kernels, which provides a seprate Jupyter kernel for each conda environment. You need it to run a Jupyter notebook with a kernel pointing to 'my_chemml_env' environment.

To run a notebook, you just need to run the following command in the Terminal:

```
jupyter notebook
```

This will consequently open a notebook dashboard in your browser. Now if you click on the 'New' button in the top right corner and select the 'python: my_chemml_env', an empty notebook will be opened in a new tab. Please type the two above-mentioned lines of python code and press Ctrl+Enter to run the wrapperGUI.

A link to the web application of this GUI will be posted here soon.

5.4 Table of Contents

This is a complete list of all the methods that are available through ChemML Wrapper interface. You can click on each function for further information.

Table's columns describe:

- task and subtask: for an easier classification of methods
- host: the main library/dependency required for running a method
- function: the method name that determines a block/node of computation graph
- input and output tokens: available tokens in each block that collect specific information and send/receive it to/from other blocks

5.5 Wrapper Reference

5.5.1 Enter

ConvertFile

task

Enter

subtask

Convert

host

cheml

function

ConvertFile

input tokens (receivers)

`file_path` : the path to the file that needs to be converted
types: ("<type 'str'>", "<type 'dict'>")

output tokens (senders)

`converted_file_paths` : list of paths to the converted files
types: <type 'list'>

required packages

ChemML, 0.4.1
Babel, 2.3.4

config file view

```
##
    << host = cheml << function = ConvertFile
    << to_format = required_required
    << file_path = required_required
    << from_format = required_required
```

```
>> id file_path
>> id converted_file_paths
```

Note: The documentation page for function parameters: https://openbabel.org/wiki/Babel

PyScript

task

Enter

subtask

python script

host

cheml

function

PyScript

input tokens (receivers)

`iv4` : input variable, of any format
types: ()
`iv5` : input variable, of any format
types: ()
`iv6` : input variable, of any format
types: ()
`iv1` : input variable, of any format
types: ()
`iv2` : input variable, of any format
types: ()
`iv3` : input variable, of any format
types: ()

output tokens (senders)

`ov2` : output variable, of any format
types: ()
`ov3` : output variable, of any format
types: ()
`ov1` : output variable, of any format
types: ()
`ov6` : output variable, of any format
types: ()
`ov4` : output variable, of any format
types: ()
`ov5` : output variable, of any format
types: ()

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = PyScript
    << line08 = type python code
    << line09 = type python code
    << line01 = type python code
    << line02 = input tokens are available as ...
    << line03 = ... python variables
    << line04 = type python code
    << line05 = type python code
    << line20 = type python code
    << line07 = type python code
    << line06 = type python code
    << line17 = type python code
    << line16 = type python code
    << line15 = type python code
    << line14 = type python code
    << line13 = type python code
    << line12 = type python code
    << line11 = type python code
    << line10 = type python code
    << line19 = type python code
    << line18 = type python code
    >> id iv4
    >> id iv5
    >> id iv6
    >> id iv1
    >> id iv2
    >> id iv3
    >> id ov2
    >> id ov3
    >> id ov1
    >> id ov6
    >> id ov4
    >> id ov5
```

Note: The documentation page for function parameters:

XYZreader

task

Enter

subtask

xyz

host

cheml

function

XYZreader

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`molecules` : dictionary of molecules with ['mol', 'file'] keys
types: ("<type 'dict'>",)

required packages

ChemML, 0.4.1

config file view

```
##
    << host = cheml << function = XYZreader
    << path_root = None
    << skip_lines = [2, 0]
    << reader = auto
    << path_pattern = required_required
    << Z = {'Ru':  44.0, 'Re':  75.0, 'Rf':  104.0, 'Rg':  111.0,
    'Ra':  88.0, 'Rb':  37.0, 'Rn':  86.0, 'Rh':  45.0, 'Be':
    4.0, 'Ba':  56.0, 'Bh':  107.0, 'Bi':  83.0, 'Bk':  97.0,
    'Br':  35.0, 'H': 1.0, 'P': 15.0, 'Os':  76.0, 'Ge':  32.0,
    'Gd':  64.0, 'Ga':  31.0, 'Pr':  59.0, 'Pt':  78.0, 'Pu':
    94.0, 'C': 6.0, 'Pb':  82.0, 'Pa':  91.0, 'Pd':  46.0, 'Cd':
    48.0, 'Po':  84.0, 'Pm':  61.0, 'Hs':  108.0, 'Uup':  115.0,
    'Uus':  117.0, 'Uuo':  118.0, 'Ho':  67.0, 'Hf':  72.0, 'Hg':
    80.0, 'He':  2.0, 'Md':  101.0, 'Mg':  12.0, 'K': 19.0, 'Mn':
    25.0, 'O': 8.0, 'Mt':  109.0, 'S': 16.0, 'W': 74.0, 'Zn':
    30.0, 'Eu':  63.0, 'Zr':  40.0, 'Er':  68.0, 'Ni':  28.0,
    'No':  102.0, 'Na':  11.0, 'Nb':  41.0, 'Nd':  60.0, 'Ne':
    10.0, 'Np':  93.0, 'Fr':  87.0, 'Fe':  26.0, 'Fl':  114.0,
    'Fm':  100.0, 'B': 5.0, 'F': 9.0, 'Sr':  38.0, 'N': 7.0, 'Kr':
    36.0, 'Si':  14.0, 'Sn':  50.0, 'Sm':  62.0, 'V': 23.0, 'Sc':
    21.0, 'Sb':  51.0, 'Sg':  106.0, 'Se':  34.0, 'Co':  27.0,
    'Cn':  112.0, 'Cm':  96.0, 'Cl':  17.0, 'Ca':  20.0, 'Cf':
    98.0, 'Ce':  58.0, 'Xe':  54.0, 'Tm':  69.0, 'Cs':  55.0,
    'Cr':  24.0, 'Cu':  29.0, 'La':  57.0, 'Li':  3.0, 'Lv':
    116.0, 'Tl':  81.0, 'Lu':  71.0, 'Lr':  103.0, 'Th':  90.0,
    'Ti':  22.0, 'Te':  52.0, 'Tb':  65.0, 'Tc':  43.0, 'Ta':
    73.0, 'Yb':  70.0, 'Db':  105.0, 'Dy':  66.0, 'Ds':  110.0,
    'At':  85.0, 'I': 53.0, 'In':  49.0, 'U': 92.0, 'Y': 39.0,
    'Ac':  89.0, 'Ag':  47.0, 'Ir':  77.0, 'Am':  95.0, 'Al':
```

```
13.0, 'As':  33.0, 'Ar':  18.0, 'Au':  79.0, 'Es':  99.0,
'Uut':  113.0, 'Mo':  42.0}
>> id molecules
```

Note: The documentation page for function parameters:

load_cep_homo

task

Enter

subtask

datasets

host

cheml

function

load_cep_homo

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`smiles` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`homo` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
<< host = cheml << function = load_cep_homo
>> id smiles
>> id homo
```

Note: The documentation page for function parameters:

load_comp_energy

task

Enter

subtask

datasets

host

cheml

function

load_comp_energy

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`formation_energy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
     << host = cheml << function = load_comp_energy
     >> id formation_energy
     >> id entries
```

Note: The documentation page for function parameters:

---

load_crystal_structures

task

Enter

subtask

datasets

host

cheml

function

load_crystal_structures

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = load_crystal_structures
    >> id entries
```

Note: The documentation page for function parameters:

---

load_organic_density

task

Enter

subtask

datasets

host

cheml

function

load_organic_density

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`smiles` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`features` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`density` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = load_organic_density
    >> id smiles
    >> id features
```

```
>> id density
```

Note: The documentation page for function parameters:

load_xyz_polarizability

task

Enter

subtask

datasets

host

cheml

function

load_xyz_polarizability

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`coordinates` : dictionary of molecules represented by their xyz coordinates and atomic numbers
types: ("<type 'dict'>",)
`polarizability` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = load_xyz_polarizability
    >> id coordinates
    >> id polarizability
```

Note: The documentation page for function parameters:

read_excel

task

Enter

subtask

table

host

pandas

function

read_excel

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

pandas, 0.20.3

config file view

```
##
    << host = pandas << function = read_excel
    << engine = None
    << squeeze = False
    << index_col = None
    << date_parser = None
    << na_values = None
    << parse_dates = False
    << dtype = None
    << skiprows = None
    << sheet_name = 0
    << header = 0
    << skip_footer = 0
    << convert_float = True
    << names = None
    << io = required_required
    << usecols = None
    << true_values = None
    << false_values = None
    << thousands = None
    << converters = None
    >> id df
```

Note: The documentation page for function parameters: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_excel.html

---

read_table

task

Enter

subtask

table

host

pandas

function

read_table

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

pandas, 0.20.3

config file view

```
##
    << host = pandas << function = read_table
    << comment = None
    << escapechar = None
    << float_precision = None
    << na_filter = True
    << iterator = False
    << sep = required_required
    << mangle_dupe_cols = True
    << skip_blank_lines = True
    << keep_default_na = True
    << false_values = None
    << header = infer
    << prefix = None
    << memory_map = False
    << names = None
    << skipfooter = 0
    << verbose = False
    << compact_ints = None
    << lineterminator = None
    << compression = infer
    << dayfirst = False
    << low_memory = True
    << encoding = None
    << parse_dates = False
```

```
<< skip_footer = 0
<< dtype = None
<< quotechar = "
<< thousands = None
<< converters = None
<< warn_bad_lines = True
<< as_recarray = None
<< engine = None
<< dialect = None
<< chunksize = None
<< tupleize_cols = None
<< na_values = None
<< infer_datetime_format = False
<< keep_date_col = False
<< use_unsigned = None
<< nrows = None
<< true_values = None
<< delim_whitespace = False
<< usecols = None
<< squeeze = False
<< buffer_lines = None
<< index_col = None
<< skipinitialspace = False
<< decimal = .
<< skiprows = None
<< filepath_or_buffer = required_required
<< date_parser = None
<< delimiter = None
<< error_bad_lines = True
<< doublequote = True
<< quoting = 0
>> id df
```

Note: The documentation page for function parameters: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_table.html

5.5.2 Represent

APEAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

APEAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = APEAttributeGenerator
    << packing_threshold = None
    << n_nearest_to_eval = None
    << radius_property = None
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

APRDFAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

APRDFAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = APRDFAttributeGenerator
    << cut_off_distance = 10.0
    << num_points = 6
    << elemental_properties = required_required
    << smooth_parameter = 4.0
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

BagofBonds

task

Represent

subtask

molecular descriptors

host

cheml

function

BagofBonds

input tokens (receivers)

`molecules` : the molecule numpy array or data frame
types: ("<class 'pandas.core.frame.DataFrame'>", "<type 'numpy.ndarray'>", "<type 'dict'>")

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
```

```
<< host = cheml << function = BagofBonds
<< const = 1
>> id molecules
>> id df
```

Note: The documentation page for function parameters:

ChargeDependentAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

ChargeDependentAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    ChargeDependentAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

ChemicalOrderingAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

ChemicalOrderingAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    ChemicalOrderingAttributeGenerator
    << weighted = True
    << shells = [1, 2, 3]
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

CompositionEntry

task

Represent

subtask

inorganic input

host

cheml

function

CompositionEntry

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = CompositionEntry
    << filepath = required_required
    >> id entries
```

Note: The documentation page for function parameters:

CoordinationNumberAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

CoordinationNumberAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    CoordinationNumberAttributeGenerator
    >> id entries
```

```
>> id df
```

Note: The documentation page for function parameters:

CoulombMatrix

task

Represent

subtask

molecular descriptors

host

cheml

function

CoulombMatrix

input tokens (receivers)

`molecules` : the molecule numpy array or data frame
types: ("<class 'pandas.core.frame.DataFrame'>", "<type 'numpy.ndarray'>", "<type 'dict'>")

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = CoulombMatrix
    << const = 1
    << CMtype = SC
    << nPerm = 3
    << max_n_atoms = auto
    >> id molecules
    >> id df
```

Note: The documentation page for function parameters:

CoulombMatrixAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

CoulombMatrixAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = CoulombMatrixAttributeGenerator
    << n_eigenvalues = 30
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

CrystalStructureEntry

task

Represent

subtask

inorganic input

host

cheml

function

CrystalStructureEntry

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = CrystalStructureEntry
    << directory_path = required_required
    >> id entries
```

Note: The documentation page for function parameters:

DistanceMatrix

task

Represent

subtask

distance matrix

host

cheml

function

DistanceMatrix

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = DistanceMatrix
```

```
<< norm_type = fro
<< nCores = 1
>> id df
>> id df
```

Note: The documentation page for function parameters:

Dragon

task

Represent

subtask

molecular descriptors

host

cheml

function

Dragon

input tokens (receivers)

`molfile` : the molecule file path
types: ("<type 'str'>", "<type 'dict'>", "<type 'list'>")

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`script` : , (default:new)

choose one of: []

required packages

ChemML, 0.4.1
pandas, 0.20.3
Dragon, 7 or 6
lxml, 3.4.0

config file view

```
##
    << host = cheml << function = Dragon
    << script = new
    << SaveExcludeConst = False
    << MaxSR = '35'
    << SaveFilePath = Dragon_descriptors.txt
    << output_directory = ./
```

```
<< DisconnectedCalculationOption = '0'
<< SaveExcludeNearConst = False
<< Add2DHydrogens = False
<< SaveProject = False
<< Decimal_Separator = .
<< SaveOnlyData = False
<< script = new
<< RejectDisconnectedStrucuture = False
<< SaveExclusionOptionsToVariables = False
<< LogEdge = True
<< LogPathWalk = True
<< SaveLabelsOnSeparateFile = False
<< version = 6
<< DefaultMolFormat = '1'
<< MaxSRDetour = '30'
<< HelpBrowser = /usr/bin/xdg-open
<< SaveExcludeRejectedMolecules = False
<< knimemode = False
<< RejectUnusualValence = False
<< SaveProjectFile = Dragon_project.drp
<< SaveStdOut = False
<< SaveFormatSubBlock = %b-%s-%n-%m.txt
<< blocks = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,
15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
<< SaveExcludeCorrelated = False
<< molFile = required_required
<< consecutiveDelimiter = False
<< molInputFormat = SMILES
<< MaxAtomWalkPath = '2000'
<< SaveExcludeAllMisVal = False
<< SaveExcludeStdDev = False
<< Weights = ['Mass', 'VdWVolume', 'Electronegativity',
'Polarizability', 'Ionization', 'I-State']
<< external = False
<< RoundWeights = True
<< MaxSRforAllCircuit = '19'
<< fileName = None
<< RoundCoordinates = True
<< Missing_String = NaN
<< SaveExcludeMisVal = False
<< logFile = Dragon_log.txt
<< PreserveTemporaryProjects = True
<< SaveLayout = True
<< molInput = file
<< SaveFormatBlock = %b-%n.txt
<< MissingValue = NaN
<< SaveCorrThreshold = '0.95'
```

```
<< SaveType = singlefile
<< ShowWorksheet = False
<< delimiter = ,
<< RetainBiggestFragment = False
<< CheckUpdates = True
<< RoundDescriptorValues = True
<< SaveExcludeMisMolecules = False
<< SaveStdDevThreshold = '0.0001'
<< SaveFile = True
<< logMode = file
>> id molfile
>> id df
```

Note: The documentation page for function parameters:

EffectiveCoordinationNumberAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

EffectiveCoordinationNumberAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    EffectiveCoordinationNumberAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

ElementFractionAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

ElementFractionAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    ElementFractionAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

ElementPairPropertyAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

ElementPairPropertyAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    ElementPairPropertyAttributeGenerator
    << elemental_pair_properties = None
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

ElementalPropertyAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

ElementalPropertyAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

wrapper parameters

`elemental_properties` : , (default:None)

choose one of: []

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    ElementalPropertyAttributeGenerator
    << elemental_properties = None
    << use_default_properties = True
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

GCLPAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

GCLPAttributeGenerator

input tokens (receivers)

`energies` : to be passed to the parameter energies
types: (“<class ‘pandas.core.frame.DataFrame’>”,)
`phases` : to be passed to the parameter phases
types: (“<class ‘pandas.core.frame.DataFrame’>”,)
`entries` : list of entries from CompositionEntry class.
types: (“<type ‘list’>”,)

output tokens (senders)

`df` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = GCLPAttributeGenerator
    << count_phases = None
    << energies = []
    << phases = []
    >> id energies
    >> id phases
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

IonicCompoundProximityAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

IonicCompoundProximityAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    IonicCompoundProximityAttributeGenerator
```

```
<< max_formula_unit = 14
>> id entries
>> id df
```

Note: The documentation page for function parameters:

IonicityAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

IonicityAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
<< host = cheml << function = IonicityAttributeGenerator
>> id entries
>> id df
```

Note: The documentation page for function parameters:

LatticeSimilarityAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

LatticeSimilarityAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    LatticeSimilarityAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

LocalPropertyDifferenceAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

LocalPropertyDifferenceAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    LocalPropertyDifferenceAttributeGenerator
    << elemental_properties = required_required
    << shells = [1]
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

LocalPropertyVarianceAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

LocalPropertyVarianceAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: (“<type ‘list’>”,)

output tokens (senders)

`df` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    LocalPropertyVarianceAttributeGenerator
    << elemental_properties = required_required
```

```
<< shells = [1]
>> id entries
>> id df
```

Note: The documentation page for function parameters:

MeredigAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

MeredigAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = MeredigAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

PRDFAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

PRDFAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = PRDFAttributeGenerator
    << cut_off_distance = 10.0
    << n_points = 20
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

PackingEfficiencyAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

PackingEfficiencyAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    PackingEfficiencyAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

RDKitFingerprint

task

Represent

subtask

molecular descriptors

host

cheml

function

RDKitFingerprint

input tokens (receivers)

`molfile` : the molecule file path
types: ("<type 'str'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`removed_rows` : output variable, of any format
types: ()

required packages

ChemML, 0.4.1
pandas, 0.20.3
RDKit, 2016.03.1

config file view

```
##
    << host = cheml << function = RDKitFingerprint
```

```
<< nBits = 1024
<< molfile = required_required
<< removeHs = True
<< vector = bit
<< radius = 2
<< arguments = []
<< path = None
<< FPtype = Morgan
>> id molfile
>> id df
>> id removed_rows
```

Note: The documentation page for function parameters:

StoichiometricAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

StoichiometricAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`use_default_norms` : , (default:None)

choose one of: []

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
     << host = cheml << function = StoichiometricAttributeGenerator
```

```
<< use_default_norms = None
<< p_norms = None
>> id entries
>> id df
```

Note: The documentation page for function parameters:

StructuralHeterogeneityAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

StructuralHeterogeneityAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CrystalStructureEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function =
    StructuralHeterogeneityAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

ValenceShellAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

ValenceShellAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
     << host = cheml << function = ValenceShellAttributeGenerator
     >> id entries
     >> id df
```

Note: The documentation page for function parameters:

YangOmegaAttributeGenerator

task

Represent

subtask

inorganic descriptors

host

cheml

function

YangOmegaAttributeGenerator

input tokens (receivers)

`entries` : list of entries from CompositionEntry class.
types: ("<type 'list'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = YangOmegaAttributeGenerator
    >> id entries
    >> id df
```

Note: The documentation page for function parameters:

5.5.3 Prepare

ConstantColumns

task

Prepare

subtask

data cleaning

host

cheml

function

ConstantColumns

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of ChemML's Constant class
types: ("<class 'cheml.preprocessing.purge.ConstantColumns'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of ChemML's Constant class
types: ("<class 'cheml.preprocessing.purge.ConstantColumns'>",)
`removed_columns_` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('fit_transform', 'transform', None)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = ConstantColumns
    << func_method = None
    >> id df
    >> id api
    >> id df
    >> id api
    >> id removed_columns_
```

Note: The documentation page for function parameters:

MissingValues

task

Prepare

subtask

data cleaning

host

cheml

function

MissingValues

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of ChemML's MissingValues class
types: ("<class 'cheml.preprocessing.handle_missing.missing_values'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of ChemML's MissingValues class
types: ("<class 'cheml.preprocessing.handle_missing.missing_values'>",)

wrapper parameters

`func_method` : String, (default:None)

choose one of: ('fit_transform', 'transform', None)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = MissingValues
    << func_method = None
    << strategy = ignore_row
    << inf_as_null = True
    << string_as_null = True
    << missing_values = False
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters:

Outliers

task

Prepare

subtask

data cleaning

host

cheml

function

Outliers

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of ChemML's Constant class
types: ("<class 'cheml.preprocessing.purge.Outliers'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of ChemML's Constant class
types: ("<class 'cheml.preprocessing.purge.Outliers'>",)

`removed_columns_` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('fit_transform', 'transform', None)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = Outliers
    << func_method = None
    << m = 2.0
    << strategy = median
    >> id df
    >> id api
    >> id df
    >> id api
    >> id removed_columns_
```

Note: The documentation page for function parameters:

Split

task

Prepare

subtask

data manipulation

host

cheml

function

Split

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`df1` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`df2` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = Split
    << selection = 1
    >> id df
    >> id df1
    >> id df2
```

Note: The documentation page for function parameters:

Binarizer

task

Prepare

subtask

feature representation

host

sklearn

function

Binarizer

input tokens (receivers)

`df` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)
`api` : instance of scikit-learn’s Binarizer class
types: (“<class ‘sklearn.preprocessing.data.Binarizer’>”,)

output tokens (senders)

`df` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)
`api` : instance of scikit-learn’s Binarizer class
types: (“<class ‘sklearn.preprocessing.data.Binarizer’>”,)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe’s header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

fit_transform: always make a new api; transform: must receive an api; None: only make a new api

choose one of: ('fit_transform', 'transform', None)

required packages

scikit-learn, 0.19.0

pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = Binarizer
    << track_header = True
    << func_method = None
    << threshold = 0.0
    << copy = True
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Binarizer.html#sklearn.preprocessing.Binarizer

Imputer

task

Prepare

subtask

data cleaning

host

sklearn

function

Imputer

input tokens (receivers)

`df` : pandas dataframe

types: ("<class 'pandas.core.frame.DataFrame'>",)

`api` : instance of scikit-learn's Imputer class

types: ("<class 'sklearn.preprocessing.imputation.Imputer'>",)

output tokens (senders)

`df` : pandas dataframe

types: ("<class 'pandas.core.frame.DataFrame'>",)

`api` : instance of scikit-learn's Imputer class

types: ("<class 'sklearn.preprocessing.imputation.Imputer'>",)

wrapper parameters

`track_header` : Boolean, (default:True)

if True, the input dataframe's header will be transformed to the output dataframe

choose one of: (True, False)

`func_method` : string, (default:None)

fit_transform: always make a new api; transform: must receive an api; None: only make a new api

choose one of: ('fit_transform', 'transform', None)

required packages

scikit-learn, 0.19.0

pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = Imputer
    << track_header = True
    << func_method = None
    << verbose = 0
    << missing_values = NaN
    << strategy = mean
    << copy = True
    << axis = 0
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Imputer.html#sklearn.preprocessing.Imputer

KFold

task

Prepare

subtask

split

host

sklearn

function

KFold

input tokens (receivers)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's KFold class
types: ("<class 'sklearn.model_selection._split.KFold'>",)
`fold_gen` : Generator of indices to split data into training and test set
types: ("<type 'generator'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('split', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = KFold
    << func_method = None
    << random_state = None
    << shuffle = False
    << n_splits = 3
    >> id dfx
    >> id api
    >> id fold_gen
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

KernelPCA

task

Prepare

subtask

feature transformation

host

sklearn

function

KernelPCA

input tokens (receivers)

`df` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

`api` : instance of scikit-learn’s KernelPCA class

types: (“<class ‘sklearn.decomposition.kernel_pca.KernelPCA’>”,)

output tokens (senders)

`df` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

`api` : instance of scikit-learn’s KernelPCA class

types: (“<class ‘sklearn.decomposition.kernel_pca.KernelPCA’>”,)

wrapper parameters

`track_header` : Boolean, (default:False)

Always False, the header of input dataframe is not equivalent with the transformed dataframe

choose one of: False

`func_method` : string, (default:None)

fit_transform: always make a new api; transform: must receive an api; inverse_transform: must receive an api; None: only make a new api

choose one of: (‘fit_transform’, ‘transform’, ‘inverse_transform’, None)

required packages

scikit-learn, 0.19.0

pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = KernelPCA
     << track_header = False
     << func_method = None
     << fit_inverse_transform = False
     << kernel = linear
     << n_jobs = 1
     << eigen_solver = auto
     << degree = 3
     << max_iter = None
     << copy_X = True
     << kernel_params = None
     << random_state = None
     << n_components = None
     << remove_zero_eig = False
     << tol = 0
     << alpha = 1.0
     << coef0 = 1
     << gamma = None
     >> id df
     >> id api
     >> id df
     >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.KernelPCA.html#sklearn.decomposition.KernelPCA

LeaveOneOut

task

Prepare

subtask

split

host

sklearn

function

LeaveOneOut

input tokens (receivers)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's LeaveOneOut class
types: ("<class 'sklearn.model_selection._split.LeaveOneOut'>",)
`fold_gen` : Generator of indices to split data into training and test set
types: ("<type 'generator'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('split', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = LeaveOneOut
    << func_method = None
    >> id dfx
    >> id api
    >> id fold_gen
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.LeaveOneOut.html

MaxAbsScaler

task

Prepare

subtask

scaling

host

sklearn

function

MaxAbsScaler

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's MaxAbsScaler class
types: ("<class 'sklearn.preprocessing.data.MaxAbsScaler'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's MaxAbsScaler class
types: ("<class 'sklearn.preprocessing.data.MaxAbsScaler'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api; inverse_transform: must receive an api; None: only make a new api
choose one of: ('fit_transform', 'transform', 'inverse_transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = MaxAbsScaler
    << track_header = True
    << func_method = None
    << copy = True
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MaxAbsScaler.html#sklearn.preprocessing.MaxAbsScaler

MinMaxScaler

task

Prepare

subtask

scaling

host

sklearn

function

MinMaxScaler

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's MinMaxScaler class
types: ("<class 'sklearn.preprocessing.data.MinMaxScaler'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's MinMaxScaler class
types: ("<class 'sklearn.preprocessing.data.MinMaxScaler'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api; inverse_transform: must receive an api; None: only make a new api
choose one of: ('fit_transform', 'transform', 'inverse_transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = MinMaxScaler
    << track_header = True
    << func_method = None
    << copy = True
```

```
<< feature_range = (0, 1)
>> id df
>> id api
>> id df
>> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html#sklearn.preprocessing.MinMaxScaler

Normalizer

task

Prepare

subtask

scaling

host

sklearn

function

Normalizer

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's Normalizer class
types: ("<class 'sklearn.preprocessing.data.Normalizer'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's Normalizer class
types: ("<class 'sklearn.preprocessing.data.Normalizer'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api None: only make a new api
choose one of: ('fit_transform', 'transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = Normalizer
    << track_header = True
    << func_method = None
    << copy = True
    << norm = l2
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html#sklearn.preprocessing.Normalizer

OneHotEncoder

task

Prepare

subtask

feature representation

host

sklearn

function

OneHotEncoder

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's OneHotEncoder class
types: ("<class 'sklearn.preprocessing.data.OneHotEncoder'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's OneHotEncoder class
types: ("<class 'sklearn.preprocessing.data.OneHotEncoder'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

fit_transform: always make a new api; transform: must receive an api; None: only make a new api

choose one of: ('fit_transform', 'transform', None)

required packages

scikit-learn, 0.19.0

pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = OneHotEncoder
    << track_header = True
    << func_method = None
    << dtype = <type'numpy.float64'>
    << categorical_features = all
    << n_values = auto
    << sparse = True
    << handle_unknown = error
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html#sklearn.preprocessing.OneHotEncoder

PCA

task

Prepare

subtask

feature transformation

host

sklearn

function

PCA

input tokens (receivers)

`df` : pandas dataframe

types: ("<class 'pandas.core.frame.DataFrame'>",)

`api` : instance of scikit-learn's PCA class

types: ("<class 'sklearn.decomposition.pca.PCA'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's PCA class
types: ("<class 'sklearn.decomposition.pca.PCA'>",)

wrapper parameters

`track_header` : Boolean, (default:False)
Always False, the header of input dataframe is not equivalent with the transformed dataframe
choose one of: False
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api; inverse_transform: must receive an api; None: only make a new api
choose one of: ('fit_transform', 'transform', 'inverse_transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = PCA
    << track_header = False
    << func_method = None
    << svd_solver = auto
    << iterated_power = auto
    << random_state = None
    << whiten = False
    << tol = 0.0
    << copy = True
    << n_components = None
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html#sklearn.decomposition.PCA

PolynomialFeatures

task

Prepare

subtask

feature representation

host

sklearn

function

PolynomialFeatures

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's PolynomialFeatures class
types: ("<class 'sklearn.preprocessing.data.PolynomialFeatures'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's PolynomialFeatures class
types: ("<class 'sklearn.preprocessing.data.PolynomialFeatures'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api; None: only make a new api
choose one of: ('fit_transform', 'transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = PolynomialFeatures
    << track_header = True
    << func_method = None
    << include_bias = True
    << interaction_only = False
    << degree = 2
    >> id df
    >> id api
    >> id df
    >> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html#sklearn.preprocessing.PolynomialFeatures

RobustScaler

task

Prepare

subtask

scaling

host

sklearn

function

RobustScaler

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's RobustScaler class
types: ("<class 'sklearn.preprocessing.data.RobustScaler'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's RobustScaler class
types: ("<class 'sklearn.preprocessing.data.RobustScaler'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api; inverse_transform: must receive an api; None: only make a new api
choose one of: ('fit_transform', 'transform', 'inverse_transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = RobustScaler
    << track_header = True
    << func_method = None
    << copy = True
    << with_scaling = True
    << with_centering = True
    << quantile_range = (25.0, 75.0)
    >> id df
    >> id api
```

```
>> id df
>> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html#sklearn.preprocessing.RobustScaler

ShuffleSplit

task

Prepare

subtask

split

host

sklearn

function

ShuffleSplit

input tokens (receivers)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's ShuffleSplit class
types: ("<class 'sklearn.model_selection._split.ShuffleSplit'>",)
`fold_gen` : Generator of indices to split data into training and test set
types: ("<type 'generator'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('split', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = ShuffleSplit
    << func_method = None
    << n_splits = 10
    << train_size = None
    << random_state = None
    << test_size = default
    >> id dfx
```

```
>> id api
>> id fold_gen
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.ShuffleSplit.html#sklearn.model_selection.ShuffleSplit

StandardScaler

task

Prepare

subtask

scaling

host

sklearn

function

StandardScaler

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's StandardScaler class
types: ("<class 'sklearn.preprocessing.data.StandardScaler'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`api` : instance of scikit-learn's StandardScaler class
types: ("<class 'sklearn.preprocessing.data.StandardScaler'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)
fit_transform: always make a new api; transform: must receive an api; inverse_transform: must receive an api; None: only make a new api
choose one of: ('fit_transform', 'transform', 'inverse_transform', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = StandardScaler
```

```
<< track_header = True
<< func_method = None
<< copy = True
<< with_mean = True
<< with_std = True
>> id df
>> id api
>> id df
>> id api
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler

StratifiedShuffleSplit

task

Prepare

subtask

split

host

sklearn

function

StratifiedShuffleSplit

input tokens (receivers)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's StratifiedShuffleSplit class
types: ("<class 'sklearn.model_selection._split.StratifiedShuffleSplit'>",)
`fold_gen` : Generator of indices to split data into training and test set
types: ("<type 'generator'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('split', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
```

```
<< host = sklearn << function = StratifiedShuffleSplit
<< func_method = None
<< n_splits = 10
<< train_size = None
<< random_state = None
<< test_size = default
>> id dfx
>> id api
>> id fold_gen
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html#sklearn.model_selection.StratifiedShuffleSplit

train_test_split

task

Prepare

subtask

split

host

sklearn

function

train_test_split

input tokens (receivers)

`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`dfx_test` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfy_train` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfy_test` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx_train` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe

choose one of: (True, False)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = train_test_split
    << track_header = True
    << shuffle = True
    << train_size = None
    << random_state = None
    << test_size = 0.25
    << stratify = None
    >> id dfy
    >> id dfx
    >> id dfx_test
    >> id dfy_train
    >> id dfy_test
    >> id dfx_train
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

concat

task

Prepare

subtask

data manipulation

host

pandas

function

concat

input tokens (receivers)

`df1` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`df3` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`df2` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

pandas, 0.20.3

config file view

```
##
    << host = pandas << function = concat
    << join = outer
    << verify_integrity = False
    << keys = None
    << levels = None
    << ignore_index = False
    << names = None
    << join_axes = None
    << copy = True
    << axis = 0
    >> id df1
    >> id df3
    >> id df2
    >> id df
```

Note: The documentation page for function parameters: http://pandas.pydata.org/pandas-docs/stable/generated/pandas.concat.html

5.5.4 Model

MLP

task

Model

subtask

regression

host

cheml

function

MLP

input tokens (receivers)

`api` : instance of cheml.nn.keras.MLP class
types: ("<class 'cheml.nn.keras.mlp.MLP'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of cheml.nn.keras.MLP class
types: ("<class 'cheml.nn.keras.mlp.MLP'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

ChemML, 0.4.1
keras, 2.1.2
tensorflow, 1.4.1

config file view

```
##
    << host = cheml << function = MLP
    << func_method = None
    << nhidden = 1
    << loss = mean_squared_error
    << learning_rate = 0.01
    << layer_config_file = None
    << batch_size = 100
    << lr_decay = 0.0
    << regression = True
    << nclasses = None
    << activations = None
    << opt_config_file = None
    << nepochs = 100
    << nneurons = 100
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters:

MLP_sklearn

task

Model

subtask

regression

host

cheml

function

MLP_sklearn

input tokens (receivers)

`api` : instance of cheml.nn.keras.MLP_sklearn class
types: ("<class 'cheml.nn.keras.mlp.MLP_sklearn'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of cheml.nn.keras.MLP_sklearn class
types: ("<class 'cheml.nn.keras.mlp.MLP_sklearn'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

ChemML, 0.4.1
scikit-learn, 0.19.0
keras, 2.1.2
tensorflow, 1.4.1

config file view

```
##
     << host = cheml << function = MLP_sklearn
     << func_method = None
     << nhidden = 1
     << loss = mean_squared_error
     << learning_rate = 0.01
     << layer_config_file = None
     << batch_size = 100
     << lr_decay = 0.0
     << regression = True
     << nclasses = None
     << activations = None
     << opt_config_file = None
```

```
<< nepochs = 100
<< nneurons = 100
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters:

ARDRegression

task

Model

subtask

regression

host

sklearn

function

ARDRegression

input tokens (receivers)

`api` : instance of scikit-learn's ARDRegression class
types: ("<class 'sklearn.linear_model.bayes.ARDRegression'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's ARDRegression class
types: ("<class 'sklearn.linear_model.bayes.ARDRegression'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = ARDRegression
     << track_header = True
     << func_method = None
     << normalize = False
     << n_iter = 300
     << verbose = False
     << lambda_2 = 1e-06
     << fit_intercept = True
     << threshold_lambda = 10000.0
     << compute_score = False
     << alpha_2 = 1e-06
     << tol = 0.001
     << alpha_1 = 1e-06
     << copy_X = True
     << lambda_1 = 1e-06
     >> id api
     >> id dfy
     >> id dfx
     >> id api
     >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ARDRegression.html

BayesianRidge

task

Model

subtask

regression

host

sklearn

function

BayesianRidge

input tokens (receivers)

`api` : instance of scikit-learn's BayesianRidge class
types: ("<class 'sklearn.linear_model.bayes.BayesianRidge'>",)
`dfy` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

`dfx` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`api` : instance of scikit-learn’s BayesianRidge class

types: (“<class ‘sklearn.linear_model.bayes.BayesianRidge’>”,)

`dfy_predict` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

wrapper parameters

`track_header` : Boolean, (default:True)

if True, the input dataframe’s header will be transformed to the output dataframe

choose one of: (True, False)

`func_method` : string, (default:None)

choose one of: (‘fit’, ‘predict’, None)

required packages

scikit-learn, 0.19.0

pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = BayesianRidge
    << track_header = True
    << func_method = None
    << normalize = False
    << n_iter = 300
    << verbose = False
    << lambda_2 = 1e-06
    << fit_intercept = True
    << compute_score = False
    << alpha_2 = 1e-06
    << tol = 0.001
    << alpha_1 = 1e-06
    << copy_X = True
    << lambda_1 = 1e-06
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.BayesianRidge.html

ElasticNet

task

Model

subtask

regression

host

sklearn

function

ElasticNet

input tokens (receivers)

`api` : instance of scikit-learn's ElasticNet class
types: ("<class 'sklearn.linear_model.coordinate_descent.ElasticNet'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's ElasticNet class
types: ("<class 'sklearn.linear_model.coordinate_descent.ElasticNet'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = ElasticNet
     << track_header = True
     << func_method = None
     << normalize = False
     << warm_start = False
     << selection = cyclic
     << fit_intercept = True
     << l1_ratio = 0.5
```

```
<< max_iter = 1000
<< precompute = False
<< random_state = None
<< tol = 0.0001
<< positive = False
<< copy_X = True
<< alpha = 1.0
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html

KernelRidge

task

Model

subtask

regression

host

sklearn

function

KernelRidge

input tokens (receivers)

`api` : instance of scikit-learn's KernelRidge class
types: ("<class 'sklearn.kernel_ridge.KernelRidge'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's KernelRidge class
types: ("<class 'sklearn.kernel_ridge.KernelRidge'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe

choose one of: (True, False)

`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = KernelRidge
    << track_header = True
    << func_method = None
    << kernel = linear
    << degree = 3
    << kernel_params = None
    << alpha = 1
    << coef0 = 1
    << gamma = None
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.kernel_ridge.KernelRidge.html

Lars

task

Model

subtask

regression

host

sklearn

function

Lars

input tokens (receivers)

`api` : instance of scikit-learn's Lars class
types: ("<class 'sklearn.linear_model.least_angle.Lars'>",)

`dfy` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

`dfx` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`api` : instance of scikit-learn’s Lars class

types: (“<class ‘sklearn.linear_model.least_angle.Lars’>”,)

`dfy_predict` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

wrapper parameters

`track_header` : Boolean, (default:True)

if True, the input dataframe’s header will be transformed to the output dataframe

choose one of: (True, False)

`func_method` : string, (default:None)

choose one of: (‘fit’, ‘predict’, None)

required packages

scikit-learn, 0.19.0

pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = Lars
    << track_header = True
    << func_method = None
    << n_nonzero_coefs = 500
    << normalize = True
    << fit_path = True
    << fit_intercept = True
    << positive = False
    << eps = 2.22044604925e-16
    << precompute = auto
    << copy_X = True
    << verbose = False
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lars.html

Lasso

task

Model

subtask

regression

host

sklearn

function

Lasso

input tokens (receivers)

`api` : instance of scikit-learn's Lasso class
types: ("<class 'sklearn.linear_model.coordinate_descent.Lasso'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's Lasso class
types: ("<class 'sklearn.linear_model.coordinate_descent.Lasso'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = Lasso
    << track_header = True
    << func_method = None
    << normalize = False
    << warm_start = False
    << selection = cyclic
    << fit_intercept = True
    << positive = False
```

```
<< max_iter = 1000
<< precompute = False
<< random_state = None
<< tol = 0.0001
<< copy_X = True
<< alpha = 1.0
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html

LassoLars

task

Model

subtask

regression

host

sklearn

function

LassoLars

input tokens (receivers)

`api` : instance of scikit-learn's LassoLars class
types: ("<class 'sklearn.linear_model.least_angle.LassoLars'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's LassoLars class
types: ("<class 'sklearn.linear_model.least_angle.LassoLars'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)

`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = LassoLars
    << track_header = True
    << func_method = None
    << normalize = True
    << fit_path = True
    << fit_intercept = True
    << positive = False
    << max_iter = 500
    << eps = 2.22044604925e-16
    << precompute = auto
    << copy_X = True
    << alpha = 1.0
    << verbose = False
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LassoLars.html

---

LinearRegression

task

Model

subtask

regression

host

sklearn

function

LinearRegression

input tokens (receivers)

`api` : instance of scikit-learn's LinearRegression class
types: ("<class 'sklearn.linear_model.base.LinearRegression'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's LinearRegression class
types: ("<class 'sklearn.linear_model.base.LinearRegression'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = LinearRegression
    << track_header = True
    << func_method = None
    << normalize = False
    << n_jobs = 1
    << fit_intercept = True
    << copy_X = True
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

LinearSVR

task

Model

subtask

regression

host

sklearn

function

LinearSVR

input tokens (receivers)

`api` : instance of scikit-learn's LinearSVR class
types: ("<class 'sklearn.svm.classes.LinearSVR'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's LinearSVR class
types: ("<class 'sklearn.svm.classes.LinearSVR'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = LinearSVR
     << track_header = True
     << func_method = None
     << loss = epsilon_insensitive
     << intercept_scaling = 1.0
     << fit_intercept = True
     << epsilon = 0.0
     << max_iter = 1000
     << C = 1.0
     << random_state = None
     << dual = True
```

```
<< tol = 0.0001
<< verbose = 0
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html

LogisticRegression

task

Model

subtask

regression

host

sklearn

function

LogisticRegression

input tokens (receivers)

`api` : instance of scikit-learn's LogisticRegression class
types: ("<class 'sklearn.linear_model.logistic.LogisticRegression'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's LogisticRegression class
types: ("<class 'sklearn.linear_model.logistic.LogisticRegression'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = LogisticRegression
     << track_header = True
     << func_method = None
     << warm_start = False
     << n_jobs = 1
     << intercept_scaling = 1
     << fit_intercept = True
     << max_iter = 100
     << class_weight = None
     << C = 1.0
     << penalty = l2
     << multi_class = ovr
     << random_state = None
     << dual = False
     << tol = 0.0001
     << solver = liblinear
     << verbose = 0
     >> id api
     >> id dfy
     >> id dfx
     >> id api
     >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

MLPRegressor

task

Model

subtask

regression

host

sklearn

function

MLPRegressor

input tokens (receivers)

`api` : instance of scikit-learn's MLPRegressor class

types: (“<class ‘sklearn.neural_network.multilayer_perceptron.MLPRegressor’>”,)

`dfy` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

`dfx` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`api` : instance of scikit-learn’s MLPRegressor class
types: (“<class ‘sklearn.neural_network.multilayer_perceptron.MLPRegressor’>”,)

`dfy_predict` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe’s header will be transformed to the output dataframe
choose one of: (True, False)

`func_method` : string, (default:None)

choose one of: (‘fit’, ‘predict’, None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = MLPRegressor
     << track_header = True
     << func_method = None
     << shuffle = True
     << verbose = False
     << random_state = None
     << tol = 0.0001
     << validation_fraction = 0.1
     << learning_rate = constant
     << momentum = 0.9
     << warm_start = False
     << epsilon = 1e-08
     << activation = relu
     << max_iter = 200
     << batch_size = auto
     << alpha = 0.0001
     << early_stopping = False
     << beta_1 = 0.9
     << beta_2 = 0.999
     << nesterovs_momentum = True
     << hidden_layer_sizes = (100,)
     << solver = adam
```

```
<< power_t = 0.5
<< learning_rate_init = 0.001
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html#sklearn.neural_network.MLPRegressor

MultiTaskElasticNet

task

Model

subtask

regression

host

sklearn

function

MultiTaskElasticNet

input tokens (receivers)

`api` : instance of scikit-learn's MultiTaskElasticNet class
types: ("<class 'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's MultiTaskElasticNet class
types: ("<class 'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = MultiTaskElasticNet
     << track_header = True
     << func_method = None
     << normalize = False
     << warm_start = False
     << selection = cyclic
     << fit_intercept = True
     << l1_ratio = 0.5
     << max_iter = 1000
     << random_state = None
     << tol = 0.0001
     << copy_X = True
     << alpha = 1.0
     >> id api
     >> id dfy
     >> id dfx
     >> id api
     >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.MultiTaskElasticNet.html

MultiTaskLasso

task

Model

subtask

regression

host

sklearn

function

MultiTaskLasso

input tokens (receivers)

`api` : instance of scikit-learn's MultiTaskLasso class
types: ("<class 'sklearn.linear_model.coordinate_descent.MultiTaskLasso'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`api` : instance of scikit-learn’s MultiTaskLasso class
types: (“<class ‘sklearn.linear_model.coordinate_descent.MultiTaskLasso’>”,)
`dfy_predict` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe’s header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: (‘fit’, ‘predict’, None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = MultiTaskLasso
    << track_header = True
    << func_method = None
    << normalize = False
    << warm_start = False
    << selection = cyclic
    << fit_intercept = True
    << max_iter = 1000
    << random_state = None
    << tol = 0.0001
    << copy_X = True
    << alpha = 1.0
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.MultiTaskLasso.html

NuSVR

task

Model

subtask

regression

host

sklearn

function

NuSVR

input tokens (receivers)

`api` : instance of scikit-learn's NuSVR class
types: ("<class 'sklearn.svm.classes.NuSVR'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's NuSVR class
types: ("<class 'sklearn.svm.classes.NuSVR'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = NuSVR
     << track_header = True
     << func_method = None
     << kernel = rbf
     << verbose = False
     << degree = 3
     << coef0 = 0.0
     << max_iter = -1
     << C = 1.0
     << tol = 0.001
     << cache_size = 200
```

```
<< shrinking = True
<< nu = 0.5
<< gamma = auto
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVR.html

Ridge

task

Model

subtask

regression

host

sklearn

function

Ridge

input tokens (receivers)

`api` : instance of scikit-learn's Ridge class
types: ("<class 'sklearn.linear_model.ridge.Ridge'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's Ridge class
types: ("<class 'sklearn.linear_model.ridge.Ridge'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
     << host = sklearn << function = Ridge
     << track_header = True
     << func_method = None
     << normalize = False
     << fit_intercept = True
     << max_iter = None
     << random_state = None
     << tol = 0.001
     << copy_X = True
     << alpha = 1.0
     << solver = auto
     >> id api
     >> id dfy
     >> id dfx
     >> id api
     >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html

SGDRegressor

task

Model

subtask

regression

host

sklearn

function

SGDRegressor

input tokens (receivers)

`api` : instance of scikit-learn's SGDRegressor class
types: ("<class 'sklearn.linear_model.stochastic_gradient.SGDRegressor'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`api` : instance of scikit-learn’s SGDRegressor class
types: (“<class ‘sklearn.linear_model.stochastic_gradient.SGDRegressor’>”,)
`dfy_predict` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe’s header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: (‘fit’, ‘predict’, None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = SGDRegressor
    << track_header = True
    << func_method = None
    << warm_start = False
    << loss = squared_loss
    << eta0 = 0.01
    << verbose = 0
    << fit_intercept = True
    << l1_ratio = 0.15
    << average = False
    << n_iter = 5
    << penalty = l2
    << power_t = 0.25
    << alpha = 0.0001
    << random_state = None
    << epsilon = 0.1
    << shuffle = True
    << learning_rate = invscaling
    >> id api
    >> id dfy
    >> id dfx
    >> id api
    >> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/

generated/sklearn.linear_model.SGDRegressor.html

SVR

task

Model

subtask

regression

host

sklearn

function

SVR

input tokens (receivers)

`api` : instance of scikit-learn's SVR class
types: ("<class 'sklearn.svm.classes.SVR'>",)
`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`api` : instance of scikit-learn's SVR class
types: ("<class 'sklearn.svm.classes.SVR'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`func_method` : string, (default:None)

choose one of: ('fit', 'predict', None)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = SVR
    << track_header = True
    << func_method = None
    << kernel = rbf
    << verbose = False
```

```
<< degree = 3
<< coef0 = 0.0
<< epsilon = 0.1
<< max_iter = -1
<< C = 1.0
<< tol = 0.001
<< cache_size = 200
<< shrinking = True
<< gamma = auto
>> id api
>> id dfy
>> id dfx
>> id api
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

5.5.5 Search

GA_DEAP

task

Search

subtask

genetic algorithm

host

cheml

function

GA_DEAP

input tokens (receivers)

`evaluate` : a function that receives a list of individuals and returns the score
types: ("<type 'function'>",)

output tokens (senders)

`best_individual` : pandas dataframe of the best individual
types: ("<class 'pandas.core.frame.DataFrame'>",)
`best_ind_df` : pandas dataframe of best individuals after each iteration
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`func_method` : string, (default:algorithm_1)
a method of the GA_DEAP class that should be applied

choose one of: ('algorithm_1', 'algorithm_2', 'algorithm_3', 'algorithm_4')

required packages

ChemML, 0.4.1
pandas, 0.20.3
deap, 1.2.2

config file view

```
##
    << host = cheml << function = GA_DEAP
    << func_method = algorithm_1
    << mut_float_dev = 1
    << init_pop_frac = 0.35
    << crossover_type = Blend
    << chromosome_type = (1,)
    << n_generations = 20
    << Evaluate = @evaluate
    << chromosome_length = 1
    << crossover_pop_frac = 0.35
    << crossover_prob = 0.4
    << Weights = (-1.0,)
    << mut_float_mean = 0
    << bit_limits = ((0, 10),)
    << mut_int_lower = (1,)
    << mut_int_upper = (10,)
    << pop_size = 50
    << mutation_prob = 0.4
    >> id evaluate
    >> id best_individual
    >> id best_ind_df
```

Note: The documentation page for function parameters:

GridSearchCV

task

Search

subtask

grid

host

sklearn

function

GridSearchCV

input tokens (receivers)

`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`estimator` : instance of a machine learning class
types: ("<type 'str'>", "<class 'sklearn.linear_model.base.LinearRegression'>", "<class 'sklearn.linear_model.ridge.Ridge'>", "<class 'sklearn.kernel_ridge.KernelRidge'>", "<class 'sklearn.linear_model.coordinate_descent.Lasso'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskLasso'>", "<class 'sklearn.linear_model.coordinate_descent.ElasticNet'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>", "<class 'sklearn.linear_model.least_angle.Lars'>", "<class 'sklearn.linear_model.least_angle.LassoLars'>", "<class 'sklearn.linear_model.bayes.BayesianRidge'>", "<class 'sklearn.linear_model.bayes.ARDRegression'>", "<class 'sklearn.linear_model.logistic.LogisticRegression'>", "<class 'sklearn.linear_model.stochastic_gradient.SGDRegressor'>", "<class 'sklearn.svm.classes.SVR'>", "<class 'sklearn.svm.classes.NuSVR'>", "<class 'sklearn.svm.classes.LinearSVR'>", "<class 'sklearn.neural_network.multilayer_perceptron.MLPRegressor'>", "<class 'cheml.nn.keras.mlp.MLP_sklearn'>")

`scorer` : instance of scikit-learn's make_scorer class
types: ("<class 'sklearn.metrics.scorer._PredictScorer'>",)

`cv` : instance of scikit-learn's cross validation generator or instance object
types: ("<type 'generator'>", "<class 'sklearn.model_selection._split.KFold'>", "<class 'sklearn.model_selection._split.ShuffleSplit'>", "<class 'sklearn.model_selection._split.StratifiedShuffleSplit'>", "<class 'sklearn.model_selection._split.LeaveOneOut'>")

output tokens (senders)

`cv_results_` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`api` : instance of scikit-learn's GridSearchCV class
types: ("<class 'sklearn.grid_search.GridSearchCV'>",)

`best_estimator_` : instance of a machine learning class
types: ("<class 'sklearn.linear_model.base.LinearRegression'>", "<class 'sklearn.linear_model.ridge.Ridge'>", "<class 'sklearn.kernel_ridge.KernelRidge'>", "<class 'sklearn.linear_model.coordinate_descent.Lasso'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskLasso'>", "<class 'sklearn.linear_model.coordinate_descent.ElasticNet'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>", "<class 'sklearn.linear_model.least_angle.Lars'>", "<class 'sklearn.linear_model.least_angle.LassoLars'>", "<class 'sklearn.linear_model.bayes.BayesianRidge'>", "<class 'sklearn.linear_model.bayes.ARDRegression'>", "<class 'sklearn.linear_model.logistic.LogisticRegression'>", "<class 'sklearn.linear_model.stochastic_gradient.SGDRegressor'>", "<class 'sklearn.svm.classes.SVR'>", "<class 'sklearn.svm.classes.NuSVR'>", "<class 'sklearn.svm.classes.LinearSVR'>", "<class 'sklearn.neural_network.multilayer_perceptron.MLPRegressor'>", "<class

'cheml.nn.keras.mlp.MLP_sklearn'>")

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = GridSearchCV
    << track_header = True
    << scoring = None
    << n_jobs = 1
    << verbose = 0
    << fit_params = None
    << refit = True
    << return_train_score = True
    << iid = True
    << estimator = @estimator
    << error_score = raise
    << pre_dispatch = 2 * n_jobs
    << param_grid = {}
    << cv = None
    >> id dfy
    >> id dfx
    >> id estimator
    >> id scorer
    >> id cv
    >> id cv_results_
    >> id api
    >> id best_estimator_
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

cross_val_predict

task

Search

subtask

validate

host

sklearn

function

cross_val_predict

input tokens (receivers)

`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`estimator` : instance of a machine learning class
types: ("<class 'sklearn.linear_model.base.LinearRegression'>", "<class 'sklearn.linear_model.ridge.Ridge'>", "<class 'sklearn.kernel_ridge.KernelRidge'>", "<class 'sklearn.linear_model.coordinate_descent.Lasso'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskLasso'>", "<class 'sklearn.linear_model.coordinate_descent.ElasticNet'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>", "<class 'sklearn.linear_model.least_angle.Lars'>", "<class 'sklearn.linear_model.least_angle.LassoLars'>", "<class 'sklearn.linear_model.bayes.BayesianRidge'>", "<class 'sklearn.linear_model.bayes.ARDRegression'>", "<class 'sklearn.linear_model.logistic.LogisticRegression'>", "<class 'sklearn.linear_model.stochastic_gradient.SGDRegressor'>", "<class 'sklearn.svm.classes.SVR'>", "<class 'sklearn.svm.classes.NuSVR'>", "<class 'sklearn.svm.classes.LinearSVR'>", "<class 'sklearn.neural_network.multilayer_perceptron.MLPRegressor'>", "<class 'cheml.nn.keras.mlp.MLP_sklearn'>")

`scorer` : instance of scikit-learn's make_scorer class
types: ("<class 'sklearn.metrics.scorer._PredictScorer'>",)

`cv` : cross-validation generator or instance object
types: ("<type 'generator'>", "<class 'sklearn.model_selection._split.KFold'>", "<class 'sklearn.model_selection._split.ShuffleSplit'>", "<class 'sklearn.model_selection._split.StratifiedShuffleSplit'>", "<class 'sklearn.model_selection._split.LeaveOneOut'>")

output tokens (senders)

`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
```

```
<< host = sklearn << function = cross_val_predict
<< track_header = True
<< n_jobs = 1
<< verbose = 0
<< fit_params = None
<< method = predict
<< pre_dispatch = 2 * n_jobs
<< estimator = @estimator
<< groups = None
<< y = None
<< X = @dfx
<< cv = None
>> id dfy
>> id dfx
>> id estimator
>> id scorer
>> id cv
>> id dfy_predict
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_predict.html#sklearn.model_selection.cross_val_predict

cross_val_score

task

Search

subtask

validate

host

sklearn

function

cross_val_score

input tokens (receivers)

`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`estimator` : instance of a machine learning class
types: ("<class 'sklearn.linear_model.base.LinearRegression'>", "<class 'sklearn.linear_model.ridge.Ridge'>", "<class 'sklearn.kernel_ridge.KernelRidge'>", "<class 'sklearn.linear_model.coordinate_descent.Lasso'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskLasso'>", "<class

'sklearn.linear_model.coordinate_descent.ElasticNet'>", "<class
'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>", "<class
'sklearn.linear_model.least_angle.Lars'>", "<class
'sklearn.linear_model.least_angle.LassoLars'>", "<class
'sklearn.linear_model.bayes.BayesianRidge'>", "<class
'sklearn.linear_model.bayes.ARDRegression'>", "<class
'sklearn.linear_model.logistic.LogisticRegression'>", "<class
'sklearn.linear_model.stochastic_gradient.SGDRegressor'>", "<class
'sklearn.svm.classes.SVR'>", "<class 'sklearn.svm.classes.NuSVR'>", "<class
'sklearn.svm.classes.LinearSVR'>", "<class
'sklearn.neural_network.multilayer_perceptron.MLPRegressor'>", "<class
'cheml.nn.keras.mlp.MLP_sklearn'>")

`scorer` : instance of scikit-learn's make_scorer class

types: ("<class 'sklearn.metrics.scorer._PredictScorer'>",)

`cv` : cross-validation generator or instance object

types: ("<type 'generator'>", "<class 'sklearn.model_selection._split.KFold'>", "<class
'sklearn.model_selection._split.ShuffleSplit'>", "<class
'sklearn.model_selection._split.StratifiedShuffleSplit'>", "<class
'sklearn.model_selection._split.LeaveOneOut'>")

output tokens (senders)

`scores` : pandas dataframe

types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)

if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = cross_val_score
    << track_header = True
    << scoring = None
    << n_jobs = 1
    << verbose = 0
    << fit_params = None
    << pre_dispatch = 2 * n_jobs
    << estimator = @estimator
    << groups = None
    << y = None
    << X = @dfx
    << cv = None
    >> id dfy
    >> id dfx
    >> id estimator
```

```
>> id scorer
>> id cv
>> id scores
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html#sklearn.model_selection.cross_val_score

evaluate_regression

task

Search

subtask

evaluate

host

sklearn

function

evaluate_regression

input tokens (receivers)

`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`dfy_predict` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`evaluation_results_` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)
`evaluator` : dictionary of metrics and their score function
types: ("<type 'dict'>",)

wrapper parameters

`mae_multioutput` : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html#sklearn.metrics.mean_absolute_error, (default:uniform_average)

choose one of: ('raw_values', 'uniform_average')
`r2_score` : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score, (default:False)

choose one of: (True, False)
`mean_absolute_error` : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html#sklearn.metrics.mean_absolute_error, (default:False)

choose one of: (True, False)

multioutput : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score, (default:uniform_average)

choose one of: ('raw_values', 'uniform_average', 'variance_weighted')

mse_sample_weight : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error, (default:None)

choose one of: []

rmse_multioutput : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error, (default:uniform_average)

choose one of: ('raw_values', 'uniform_average')

median_absolute_error : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.median_absolute_error.html#sklearn.metrics.median_absolute_error, (default:False)

choose one of: (True, False)

mae_sample_weight : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html#sklearn.metrics.mean_absolute_error, (default:None)

choose one of: []

rmse_sample_weight : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error, (default:None)

choose one of: []

track_header : Boolean, (default:True)

if True, the input dataframe's header will be transformed to the output dataframe

choose one of: (True, False)

mean_squared_error : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error, (default:False)

choose one of: (True, False)

root_mean_squared_error : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error, (default:False)

choose one of: (True, False)

explained_variance_score : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.explained_variance_score.html#sklearn.metrics.explained_variance_score, (default:False)

choose one of: (True, False)

r2_sample_weight : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score, (default:None)

choose one of: []

ev_sample_weight : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.explained_variance_score.html#sklearn.metrics.explained_variance_score, (default:None)

choose one of: []

`ev_multioutput` : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.explained_variance_score.html#sklearn.metrics.explained_variance_score, (default:uniform_average)

choose one of: ('raw_values', 'uniform_average', 'variance_weighted')

`mse_multioutput` : http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error, (default:uniform_average)

choose one of: ('raw_values', 'uniform_average')

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = evaluate_regression
    << mae_multioutput = uniform_average
    << r2_score = False
    << mean_absolute_error = False
    << multioutput = uniform_average
    << mse_sample_weight = None
    << rmse_multioutput = uniform_average
    << median_absolute_error = False
    << mae_sample_weight = None
    << rmse_sample_weight = None
    << track_header = True
    << mean_squared_error = False
    << root_mean_squared_error = False
    << explained_variance_score = False
    << r2_sample_weight = None
    << ev_sample_weight = None
    << ev_multioutput = uniform_average
    << mse_multioutput = uniform_average
    >> id dfy
    >> id dfy_predict
    >> id evaluation_results_
    >> id evaluator
```

Note: The documentation page for function parameters: http://scikit-learn.org/dev/modules/model_evaluation.html#regression-metrics

learning_curve

task

Search

subtask

grid

host

sklearn

function

learning_curve

input tokens (receivers)

`dfy` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`dfx` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`estimator` : instance of a machine learning class
types: ("<class 'sklearn.linear_model.base.LinearRegression'>", "<class 'sklearn.linear_model.ridge.Ridge'>", "<class 'sklearn.kernel_ridge.KernelRidge'>", "<class 'sklearn.linear_model.coordinate_descent.Lasso'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskLasso'>", "<class 'sklearn.linear_model.coordinate_descent.ElasticNet'>", "<class 'sklearn.linear_model.coordinate_descent.MultiTaskElasticNet'>", "<class 'sklearn.linear_model.least_angle.Lars'>", "<class 'sklearn.linear_model.least_angle.LassoLars'>", "<class 'sklearn.linear_model.bayes.BayesianRidge'>", "<class 'sklearn.linear_model.bayes.ARDRegression'>", "<class 'sklearn.linear_model.logistic.LogisticRegression'>", "<class 'sklearn.linear_model.stochastic_gradient.SGDRegressor'>", "<class 'sklearn.svm.classes.SVR'>", "<class 'sklearn.svm.classes.NuSVR'>", "<class 'sklearn.svm.classes.LinearSVR'>", "<class 'sklearn.neural_network.multilayer_perceptron.MLPRegressor'>", "<class 'cheml.nn.keras.mlp.MLP_sklearn'>")

`scorer` : instance of scikit-learn's make_scorer class
types: ("<class 'sklearn.metrics.scorer._PredictScorer'>",)

`cv` : instance of scikit-learn's cross validation generator or instance object
types: ("<type 'generator'>", "<class 'sklearn.model_selection._split.KFold'>", "<class 'sklearn.model_selection._split.ShuffleSplit'>", "<class 'sklearn.model_selection._split.StratifiedShuffleSplit'>", "<class 'sklearn.model_selection._split.LeaveOneOut'>")

output tokens (senders)

`train_sizes_abs` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`extended_result_` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`test_scores` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

`train_scores` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

wrapper parameters

`track_header` : Boolean, (default:True)

if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = learning_curve
    << track_header = True
    << scoring = None
    << n_jobs = 1
    << shuffle = False
    << groups = None
    << random_state = None
    << pre_dispatch = all
    << estimator = @estimator
    << exploit_incremental_learning = False
    << train_sizes = [0.1, 0.33, 0.55, 0.78, 1.0]
    << y = None
    << X = @dfx
    << cv = None
    << verbose = 0
    >> id dfy
    >> id dfx
    >> id estimator
    >> id scorer
    >> id cv
    >> id train_sizes_abs
    >> id extended_result_
    >> id test_scores
    >> id train_scores
```

Note: The documentation page for function parameters: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.learning_curve.html#sklearn.model_selection.learning_curve

scorer_regression

task

Search

subtask

evaluate

host

sklearn

function

scorer_regression

input tokens (receivers)

this block doesn't receive anything

output tokens (senders)

`scorer` : Callable object that returns a scalar score
types: ("<class 'sklearn.metrics.scorer._PredictScorer'>",)

wrapper parameters

`track_header` : Boolean, (default:True)
if True, the input dataframe's header will be transformed to the output dataframe
choose one of: (True, False)
`metric` : string: 'mae', 'mse', 'r2', (default:mae)
http://scikit-learn.org/dev/modules/model_evaluation.html#regression-metrics
choose one of: ('mae', 'mse', 'r2')

required packages

scikit-learn, 0.19.0
pandas, 0.20.3

config file view

```
##
    << host = sklearn << function = scorer_regression
    << track_header = True
    << metric = mae
    << greater_is_better = True
    << needs_threshold = False
    << needs_proba = False
    << kwargs = {}
    >> id scorer
```

Note: The documentation page for function parameters: http://scikit-learn.org/0.15/modules/generated/sklearn.metrics.make_scorer.html#sklearn.metrics.make_scorer

---

corr

task

Search

subtask

evaluate

host

pandas

function

corr

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

required packages

pandas, 0.20.3

config file view

```
##
    << host = pandas << function = corr
    << min_periods = 1
    << method = pearson
    >> id df
    >> id df
```

Note: The documentation page for function parameters: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.corr.html

5.5.6 Mix

5.5.7 Visualize

decorator

task

Visualize

subtask

artist

host

cheml

function

decorator

input tokens (receivers)

`fig` : a matplotlib object
types: ("<class 'matplotlib.figure.Figure'>", "<class 'matplotlib.axes._subplots.AxesSubplot'>")

output tokens (senders)

`fig` : a matplotlib object
types: ("<class 'matplotlib.figure.Figure'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3
matplotlib, 1.5.1

config file view

```
##
    << host = cheml << function = decorator
    << weight = normal
    << family = normal
    << xlim = (None, None)
    ‘‘<< title = ‘‘
    << grid_color = k
    << variant = normal
    << style = normal
    << grid_linestyle = --
    ‘‘<< xlabel = ‘‘
    << grid_linewidth = 0.5
    ‘‘<< ylabel = ‘‘
    << grid = True
    << ylim = (None, None)
    << size = 18
    >> id fig
    >> id fig
```

Note: The documentation page for function parameters:

hist

task

Visualize

subtask

plot

host

cheml

function

hist

input tokens (receivers)

`dfx` : a pandas dataframe

types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`fig` : a matplotlib object
types: (“<class ‘matplotlib.figure.Figure’>”,)

required packages

ChemML, 0.4.1
pandas, 0.20.3
matplotlib, 1.5.1

config file view

```
##
     << host = cheml << function = hist
     << color = None
     << kwargs = {}
     << x = required_required
     << bins = None
     >> id dfx
     >> id fig
```

Note: The documentation page for function parameters:

scatter2D

task

Visualize

subtask

plot

host

cheml

function

scatter2D

input tokens (receivers)

`dfy` : a pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)
`dfx` : a pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`fig` : a matplotlib.Figure object
types: (“<class ‘matplotlib.figure.Figure’>”,)

required packages

ChemML, 0.4.1
pandas, 0.20.3
matplotlib, 1.5.1

config file view

```
##
    << host = cheml << function = scatter2D
    << color = b
    << marker = .
    << y = required_required
    << x = required_required
    << linewidth = 2
    ‘‘<< linestyle = ‘‘
    >> id dfy
    >> id dfx
    >> id fig
```

Note: The documentation page for function parameters:

---

plot

task

Visualize

subtask

plot

host

pandas

function

plot

input tokens (receivers)

`df` : pandas dataframe
types: (“<class ‘pandas.core.frame.DataFrame’>”,)

output tokens (senders)

`fig` : matplotlib figure or axes object
types: (“<class ‘matplotlib.axes._subplots.AxesSubplot’>”, “<class ‘matplotlib.figure.Figure’>”)

required packages

pandas, 0.20.3
matplotlib, 1.5.1

config file view

```
##
```

```
<< host = pandas << function = plot
<< xlim = None
<< xerr = None
<< yerr = None
<< logx = False
<< logy = False
<< table = False
<< ax = None
<< rot = None
<< ylim = None
<< style = None
<< sharey = False
<< sharex = None
<< title = None
<< use_index = True
<< xticks = None
<< fontsize = None
<< sort_columns = False
<< loglog = False
<< colormap = None
<< grid = None
<< layout = None
<< legend = True
<< secondary_y = False
<< kind = line
<< subplots = False
<< figsize = None
<< yticks = None
<< y = None
<< x = None
>> id df
>> id fig
```

Note: The documentation page for function parameters: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.plot.html

5.5.8 Store

SaveFile

task

Store

subtask

file

host

cheml

function

SaveFile

input tokens (receivers)

`df` : pandas dataframe
types: ("<class 'pandas.core.frame.DataFrame'>",)

output tokens (senders)

`filepath` : pandas dataframe
types: ("<type 'str'>",)

required packages

ChemML, 0.4.1
pandas, 0.20.3

config file view

```
##
    << host = cheml << function = SaveFile
    << index = False
    << record_time = False
    << format = csv
    << output_directory = None
    << header = True
    << filename = required_required
    >> id df
    >> id filepath
```

Note: The documentation page for function parameters:

SavePlot

task

Store

subtask

figure

host

cheml

function

SavePlot

input tokens (receivers)

`fig` : a matplotlib object

types: (“<class ‘matplotlib.figure.Figure’>”, “<class ‘matplotlib.axes._subplots.AxesSubplot’>”)

input tokens (receivers)

this block doesn’t send anything

required packages

ChemML, 0.4.1
pandas, 0.20.3
matplotlib, 1.5.1

config file view

```
##
    << host = cheml << function = SavePlot
    << format = png
    << output_directory = None
    << kwargs = {}
    << filename = required_required
    >> id fig
```

Note: The documentation page for function parameters: https://matplotlib.org/users/index.html

5.6 Chem module

5.7 Magpie_Python module

5.8 Initialization module

5.9 Datasets module

5.10 Preprocessing module

5.11 Models module

5.12 Optimization module

5.13 Visualization module

CHAPTER

SIX

LICENSE:

CHAPTER

SEVEN

ABOUT US:

Maintainers

- Johannes Hachmann, hachmann@buffalo.edu
- Mojtaba Haghighatlari

University at Buffalo - The State University of New York (UB)

Contributors

- Doaa Altarawy (MolSSI): scientific advice and software mentor
- Gaurav Vishwakarma (UB): automated model optimization
- Ramachandran Subramanian (UB): Magpie descriptor library port
- Bhargava Urala Kota (UB): library database
- Aditya Sonpal (UB): debugging
- Srirangaraj Setlur (UB): scientific advice
- Venugopal Govindaraju (UB): scientific advice
- Krishna Rajan (UB): scientific advice
- We encourage any contributions and feedback. Feel free to fork and make pull-request to the "development" branch.

Acknowledgements

- ChemML is based upon work supported by the U.S. National Science Foundation under grant #OAC-1751161 and in part by #OAC-1640867.
- ChemML was also supported by start-up funds provided by UB's School of Engineering and Applied Science and UB's Department of Chemical and Biological Engineering, the New York State Center of Excellence in Materials Informatics through seed grant #1140384-8-75163, and the U.S. Department of Energy under grant #DE-SC0017193.
- Mojtaba Haghighatlari received 2018 Phase-I and 2019 Phase-II Software Fellowships by the Molecular Sciences Software Institute (MolSSI) for his work on ChemML.