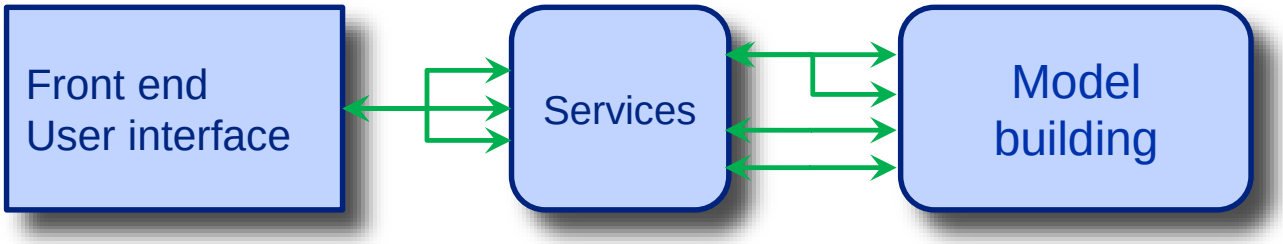
Merck Serono

MERCK
Living Innovation

Lightning Talk: PredictionKit

Paul Czodrowski / RDKit UGM

Communication frontend ← → backend



- Intranet
- Links to specificApps

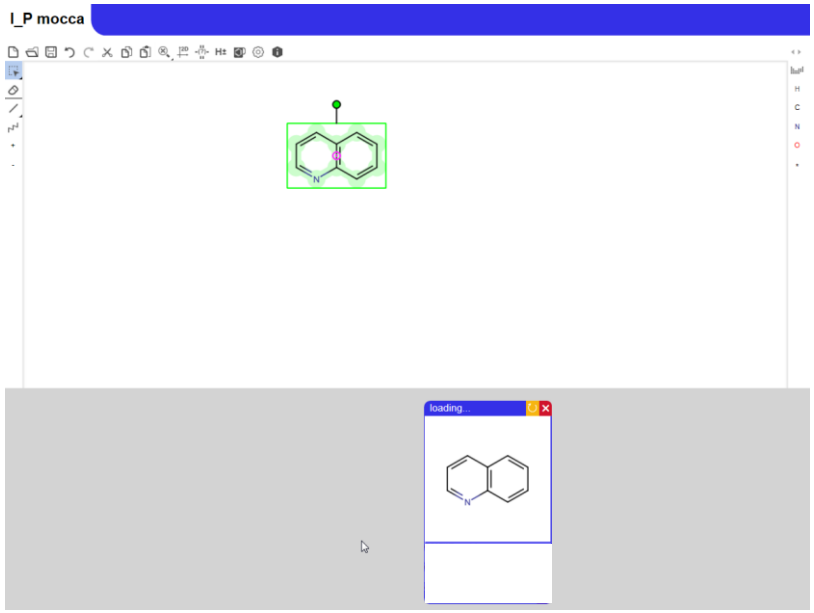
- RESTful
- SOAP
- NN

- RDKit & scikit-learn

CREAM a.k.a. PredictionKit

MOCCA

Merck **O**nline **C**omputational **C**hemistry **A**nalyzer



CREAM

Classification and **R**egression **A**t **M**erck

```
$ ./ctrain
usage: ctrain [-h] [--quiet] {listprops,addprops,train,predict} ...
ctrain: error: too few arguments
```

iPython notebook

famous_model model

Jump to input summary or training results.

Setup and configuration

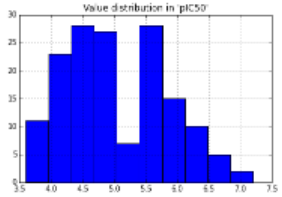
In [1]:
```
# Imports and default settings
%pylab
%matplotlib inline

import pandas as pd
import numpy as np
import numpy.random

pd.set_option("display.precision", 3)
```

```
Using matplotlib backend: Qt4Agg
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
model_name = 'famous_model'

# If specified, always save to that version number.
# If None, save to the next largest version number.
model_version = None

# If 'thresholds' is None, then 'pIC50' contains category names.
# If it's a list then the values are thresholds between different categories.
# Everything to the left of the first value is in the first category.
# Everything to the right of the last value is in the last category.
thresholds = [5.0]

# Names for each of the categories. The number of categories is one greater
# than the number of thresholds.
categories = ['0', '1']

# If a binary category, the one considered 'positive'. This is needed by
# some of the cross-validation measures.
pos_category = '1'



# The column names to use to make the prediction.
descriptors = ['MinAbsPartialCharge', 'NumRadicalElectrons',
  'HeavyAtomMolWt', 'MaxAbsEStateIndex', 'MaxAbsPartialCharge',
  'MaxEStateIndex', 'MinPartialCharge', 'ExactMolWt', 'MolWt',
  'NumValenceElectrons', 'MinEStateIndex', 'MinAbsEStateIndex',
  'MaxPartialCharge', 'BalabanJ', 'BertzCT', 'Chi0', 'Chi0n', 'Chi0v',
  'Chi1', 'Chi1n', 'Chi1v', 'Chi2n', 'Chi2v', 'Chi3n', 'Chi3v', 'Chi4n',
  'Chi4v', 'HallKierAlpha', 'Ipc', 'Kappa1', 'Kappa2', 'Kappa3',
  'LabuteASA', 'PEOE_VSA1', 'PEOE_VSA10', 'PEOE_VSA11', 'PEOE_VSA12',
  'PEOE_VSA13', 'PEOE_VSA14', 'PEOE_VSA2', 'PEOE_VSA3', 'PEOE_VSA4',
  'PEOE_VSA5', 'PEOE_VSA6', 'PEOE_VSA7', 'PEOE_VSA8', 'PEOE_VSA9',
  'SMR_VSA1', 'SMR_VSA10', 'SMR_VSA2', 'SMR_VSA3', 'SMR_VSA4',
  'SMR_VSA5', 'SMR_VSA6', 'SMR_VSA7', 'SMR_VSA8', 'SMR_VSA9',
  'SlogP_VSA1', 'SlogP_VSA10', 'SlogP_VSA11', 'SlogP_VSA12',
  'SlogP_VSA2', 'SlogP_VSA3', 'SlogP_VSA4', 'SlogP_VSA5', 'SlogP_VSA6',
  'SlogP_VSA7', 'SlogP_VSA8', 'SlogP_VSA9', 'TPSA', 'EState_VSA1',
  'EState_VSA10', 'EState_VSA11', 'EState_VSA2', 'EState_VSA3',
  'EState_VSA4', 'EState_VSA5', 'EState_VSA6', 'EState_VSA7',
  'EState_VSA8', 'EState_VSA9', 'VSA_EState1', 'VSA_EState10',
```

Input summary

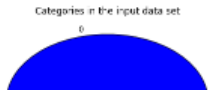
In [4]:
```
# Load the Pandas data table and extract the fields
table = pd.read_pickle(pandas_filename)
data_X = table[descriptors]
values = table[value_column]

# Show a histogram if it's continuous data.
if thresholds is not None:
    title("Value distribution in %r" % (value_column,))
    values.hist()
# Otherwise give a message for the IPython notebook
"Cannot show a histogram for categorical column %r" % value_column if thresholds is None else None
```



In [5]:
```
if thresholds is None:
    # Use the column values directly if it's categorical data
    data_y = values
else:
    # Segment the values. We only have the internal thresholds and cut() wants the min/max values
    # so create the full bins. Need to be careful that the min/max are indeed smaller/larger than
    # the bounding values.
    bins = [min(values.min(), thresholds[0])] + thresholds + [max(values.max(), thresholds[-1])]
    data_y = pd.cut(values, bins=bins, labels=categories,
                    include_lowest=True)

data_y.describe()
```

Out[5]:
```
count     136
unique      2
top         0
freq       89
Name: pIC50, dtype: object
```

In [6]:
```
# Show a pie chart of the different categories
value_counts = data_y.value_counts()
title("Categories in the input data set")
_ = pie([value_counts[category] for category in categories], labels=categories)
```



Documentation & interactive manipulation of the predictive model

Hosting of models

version of one model

```
$ ls -lrt v1/
total 17
-rw-r--r--    1 M163729  Administ     4305 Aug 19 16:30 ctrain.json
-rw-r--r--    1 M163729  Administ    27911 Aug 19 16:30 classifier.pkl
```

scikit-learn model

```
 "fr_unbrch_alkane",
 "fr_urea"
],
"model_name": "famous_model",
"model_type": "classifier",
"model_version": 1,


 "category": "classifier",
 "dtype": "object",
 "name": "famous_model_prediction"
},
{
 "category": "continuous",
 "dtype": "float",
 "name": "famous_model_probability"
},
{
 "category": "continuous",
 "dtype": "float",
 "name": "famous_model_probability_0"
},
{
 "category": "continuous",
 "dtype": "float",
 "name": "famous_model_probability_1"
}
]output_values": [
} {
```

Acknowledgment

Andrew Dalke

Jan Fiedler

Michael Krug

Wolf-Guido Bolick