Benchmarking of 2D Fingerprints and Machine-Learning Methods

Sereina Riniker

2nd RDKit UGM, Genome Campus, Hinxton UK

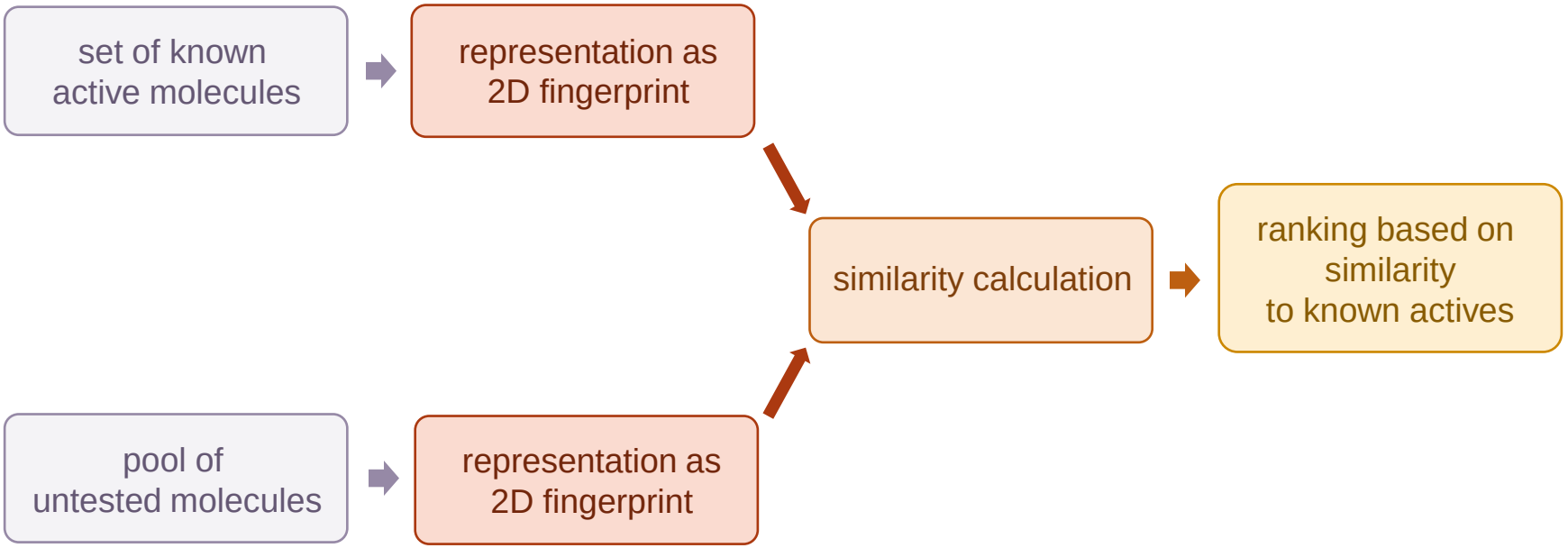
October 2, 2013

NOVARTIS

Ligand-Based Virtual Screening

Similarity search

Assumption: similar molecules have similar properties.

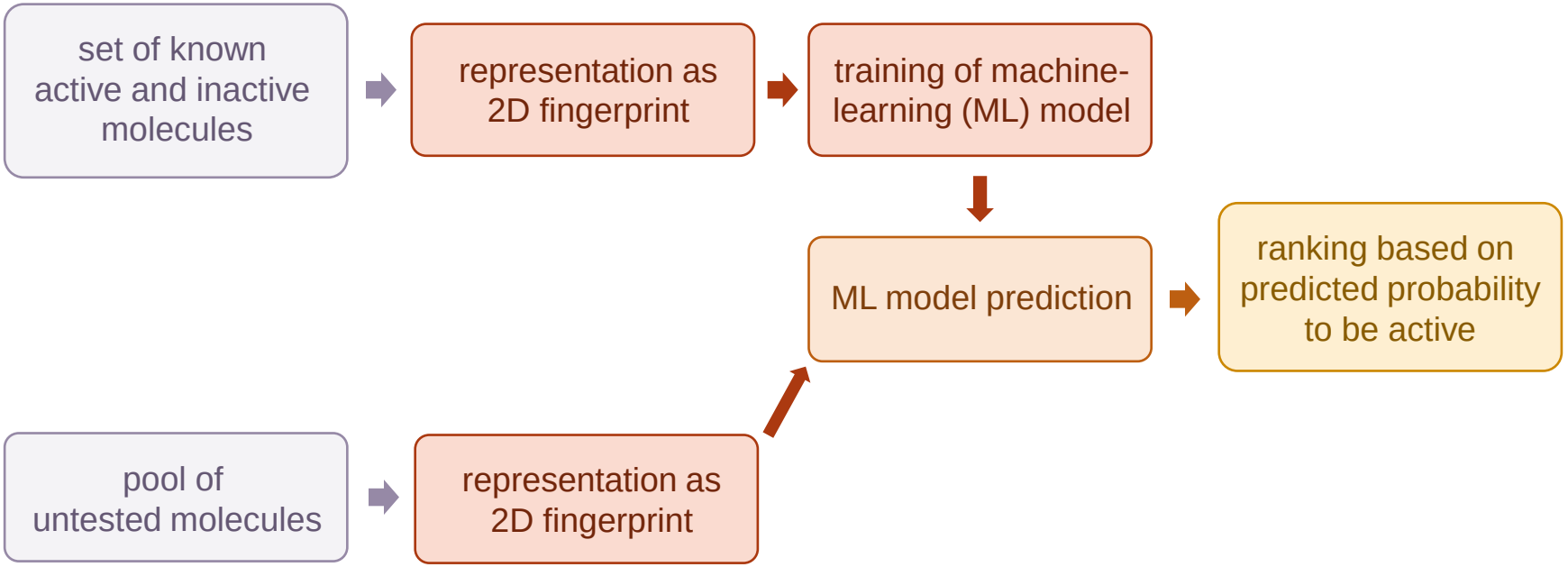


NOVARTIS

Ligand-Based Virtual Screening

Similarity search

Assumption: similar molecules have similar properties.

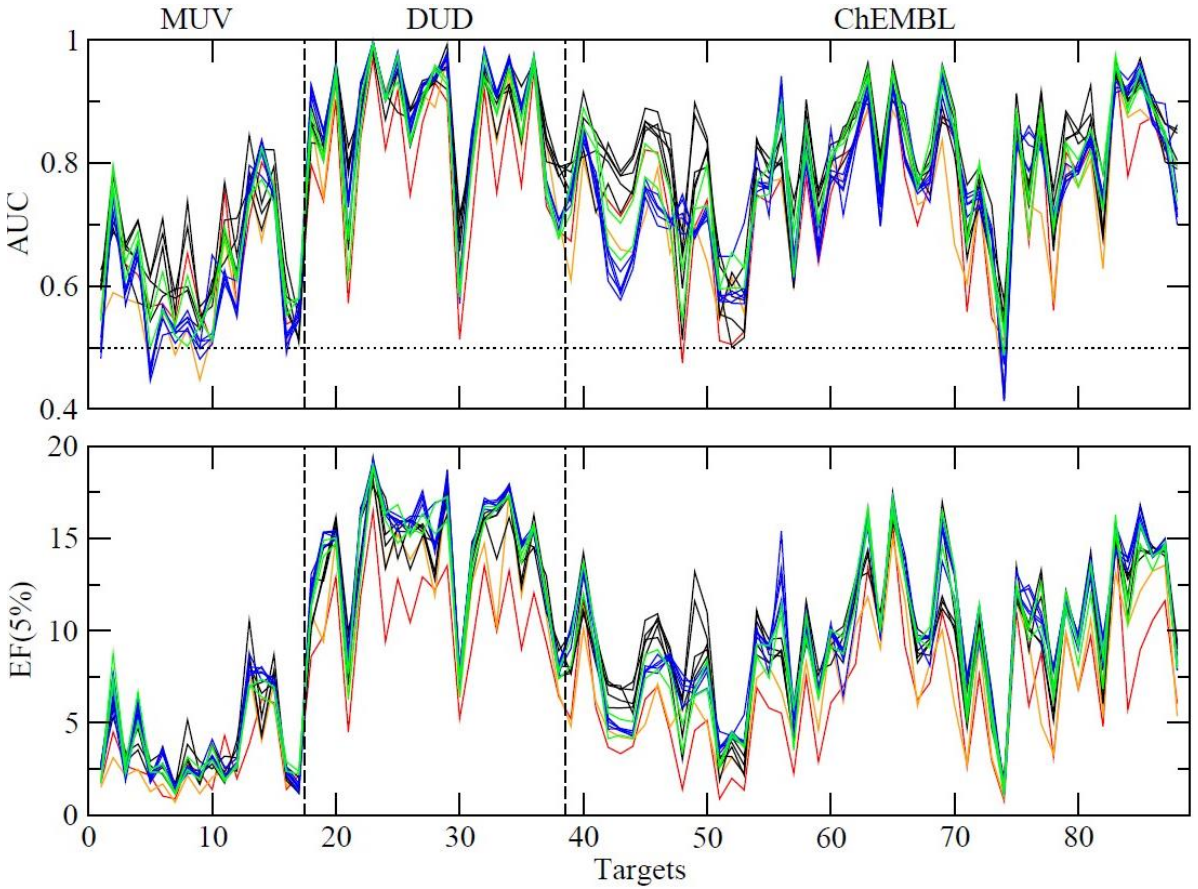


NOVARTIS

Benchmarking of Standard 2D Fingerprints

Data sets I (original)

- **Baseline fps:**
  ECFC0, MACCS
- **Path-based fps:**
  atom pairs (AP),
  torsions (TT),
  Avalon,
  RDK5
- **Circular (bit vector) fps:**
  ECFP4, ECFP6, FCFP4
- **Circular (counts) fps:**
  ECFC4, FCFC4
- **Normal size:**
  1024 bits
- **Long** (16384 bits) for:
  Avalon, ECFP4, ECFP6



MAX fusion for basic fingerprint
Average of 50 repetitions
Query molecules: 10

→ *Inter*-target differences larger than *intra*-target

S. Riniker, G. Landrum, *J. Cheminf.* (2013), **5**, 26.

Benchmarking of Standard 2D Fingerprints

Data sets I (original)

- Statistical analysis: pairwise post-hoc Friedman tests

AUC

	TT	AP	ECFC4	RDK5	Avalon	lAvalon	FCFP4	lECFP4	FCFC4	lECFP6	ECFP4	ECFP6	MACCS	ECFC0	Rank
TT		X	X	X	X	-	-	-	-	-	-	-	-	-	1
AP			X	X	X	X	o	o	-	-	-	-	-	-	1
ECFC4				X	X	X	X	X	o	o	o	-	-	-	1
RDK5					X	X	X	X	X	o	o	-	-	-	1
Avalon						X	X	X	X	o	o	-	-	-	1
lAvalon							X	X	X	X	X	X	-	-	1
FCFP4								X	X	X	X	X	-	-	1
lECFP4									X	X	X	X	-	-	1
FCFC4										X	X	X	o	-	1
lECFP6											X	X	o	o	1
ECFP4												X	o	o	1
ECFP6													X	o	1
MACCS														X	1
ECFC0															1

EF(5%)

	lECFP4	TT	lECFP6	ECFP4	ECFC4	ECFP6	FCFP4	RDK5	AP	Avalon	lAvalon	FCFC4	MACCS	ECFC0	Rank
lECFP4		X	X	X	X	o	o	o	-	-	-	-	-	-	1
TT			X	X	X	X	X	X	o	o	-	-	-	-	1
lECFP6				X	X	X	X	X	o	o	o	-	-	-	1
ECFP4					X	X	X	X	X	X	X	-	-	-	1
ECFC4						X	X	X	X	X	X	-	-	-	1
ECFP6							X	X	X	X	X	o	-	-	1
FCFP4								X	X	X	X	-	-	-	1
RDK5									X	X	X	o	-	-	1
AP										X	X	o	-	-	1
Avalon											X	X	-	-	1
lAvalon												X	-	-	1
FCFC4													-	-	1
MACCS														X	13
ECFC0															13

→ Majority of fingerprints show no statistically significant difference

S. Riniker, G. Landrum, *J. Cheminf.* (2013), **5**, 26.

NOVARTIS

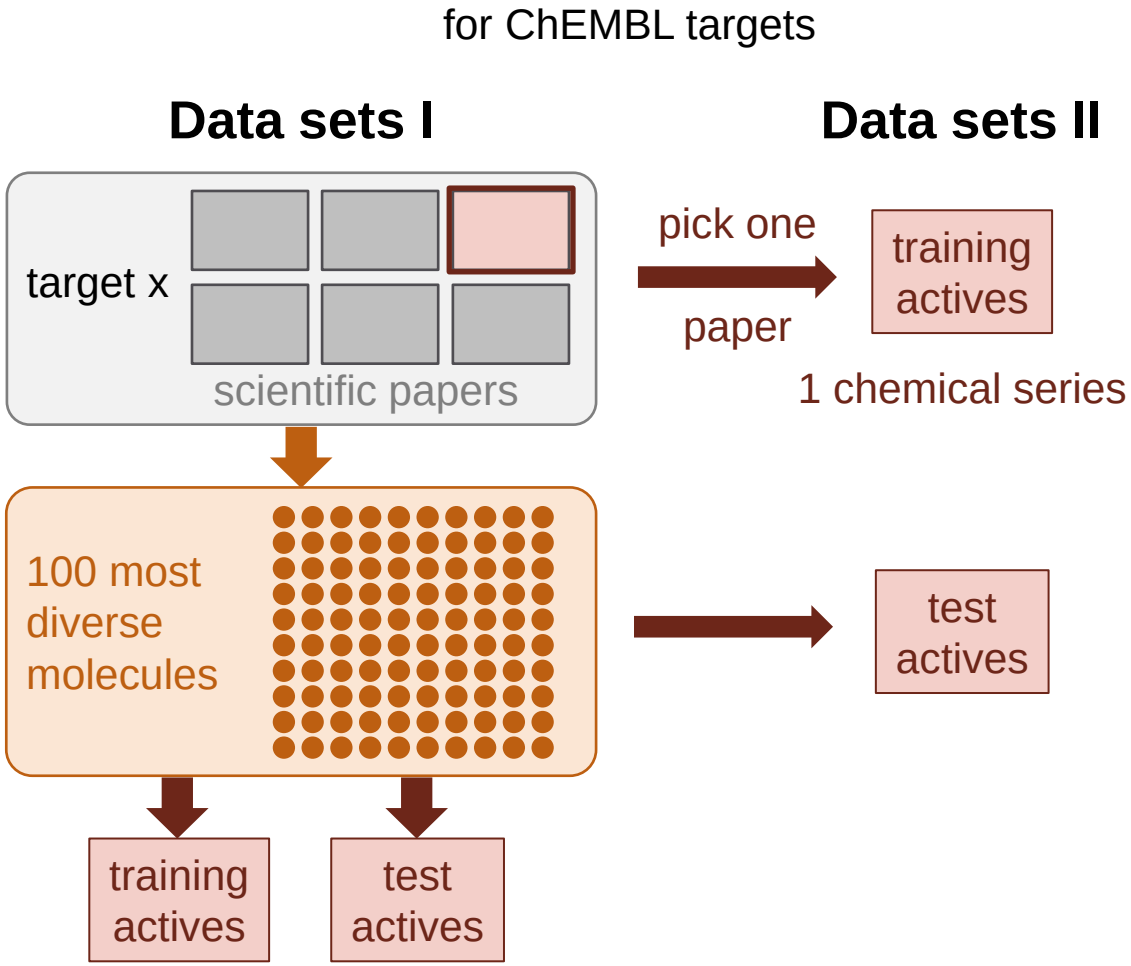
Data Sets

Two common use cases of virtual screening (VS)

- Use case 1:
  - A small set of diverse actives from e.g. a HTS is available

    ➔ **data sets I**
- Use case 2:
  - A small set of related actives, i.e. compounds sharing one or two common scaffolds, from a publication or patent is available

    ➔ **data sets II**



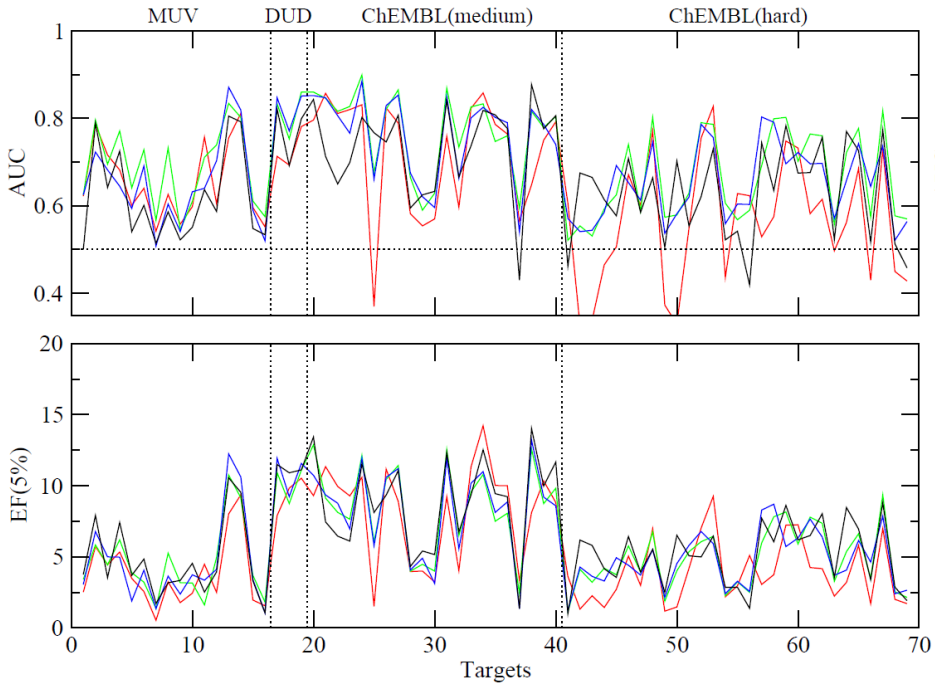
S. Riniker, N. Fechner, G. Landrum, *J. Chem. Inf. Model*. (2013), *submitted*.

NOVARTIS

Benchmarking of Standard 2D Fingerprints

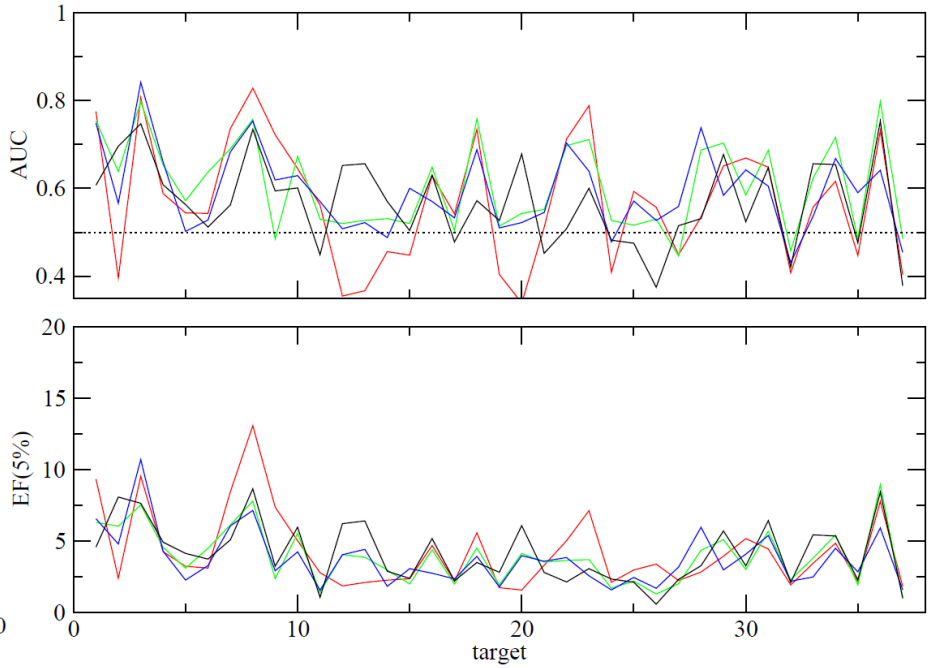
Performance of 4 fingerprints

Data sets I (difficulty filtered)



MAX group fusion for standard fingerprint
Average of 50 repetitions
Query molecules: 10

Data sets II



MAX group fusion for standard fingerprint
Average of papers

hashed AP, hashed TT,
ECFP4, RDK5

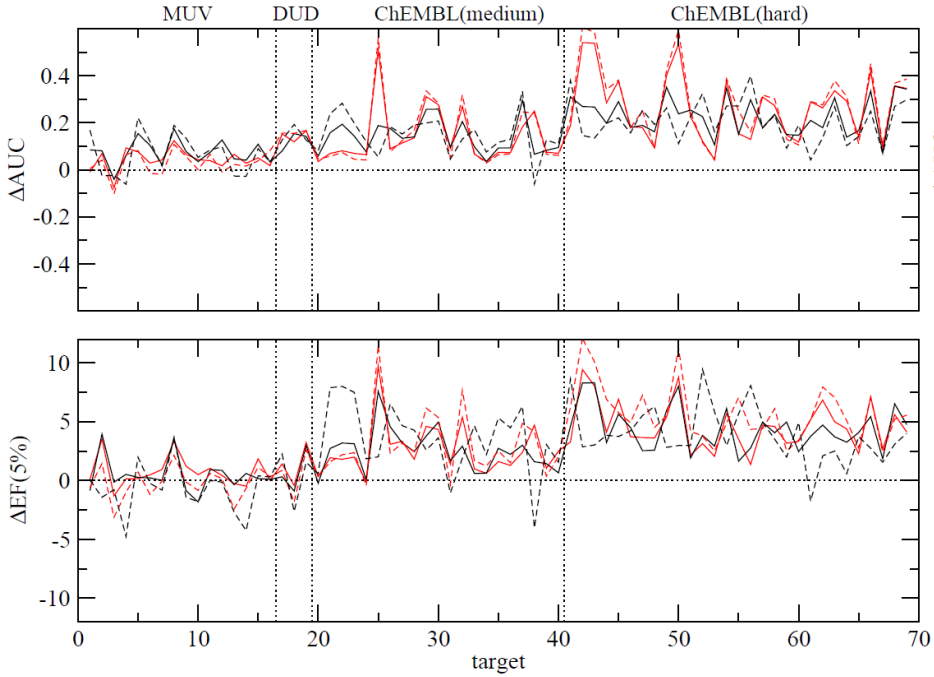
S. Riniker, N. Fechner, G. Landrum, *J. Chem. Inf. Model*. (2013), *submitted*.

NOVARTIS

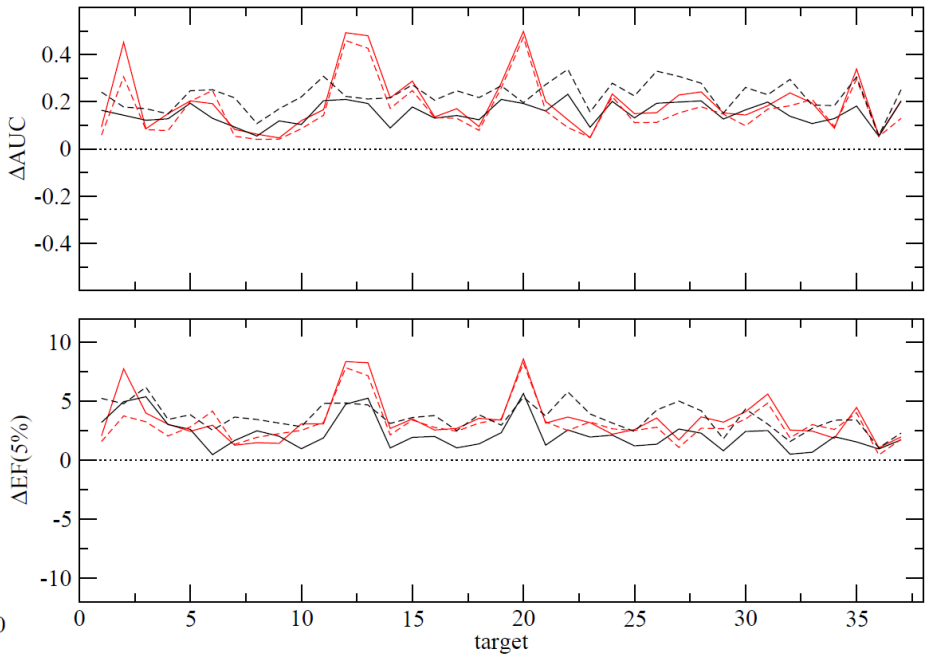
Benchmarking of Machine-Learning Methods

Performance of random forest and naïve Bayes

Data sets I (difficulty filtered)



Data sets II



solid: RF
dashed: NB

hashed AP, ECFP4

→ Use of ML method increases performance

best method/fingerprint is target-dependent

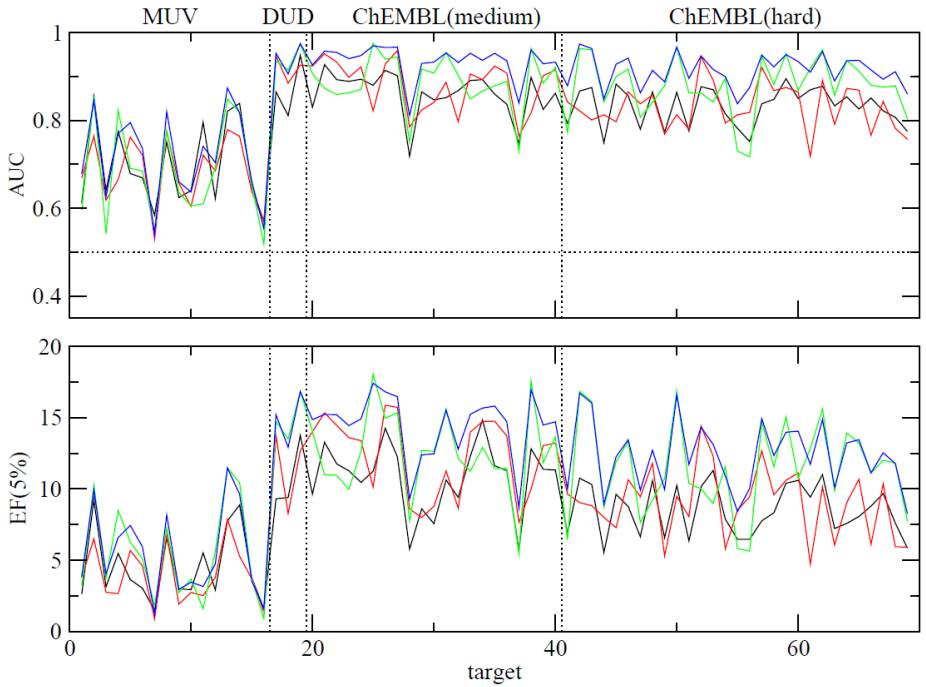
S. Riniker, N. Fechner, G. Landrum, *J. Chem. Inf. Model*. (2013), *submitted*.

NOVARTIS

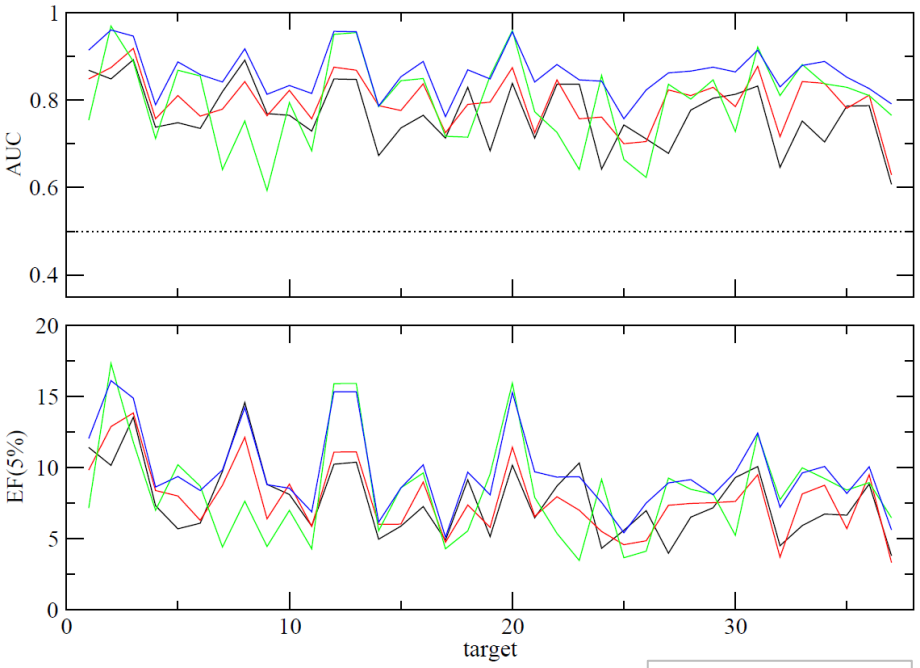
Heterogeneous Classifier Fusion

Take advantage of differences among ML methods / fingerprints

Data sets I (difficulty filtered)



Data sets II



rank-based MAX fusion of the three models (RF, NB, LR)

→ Classifier fusion similar or better than individiual models

RF(AP),
NB(ECFP4),
LR(ECFP4),
Fusion

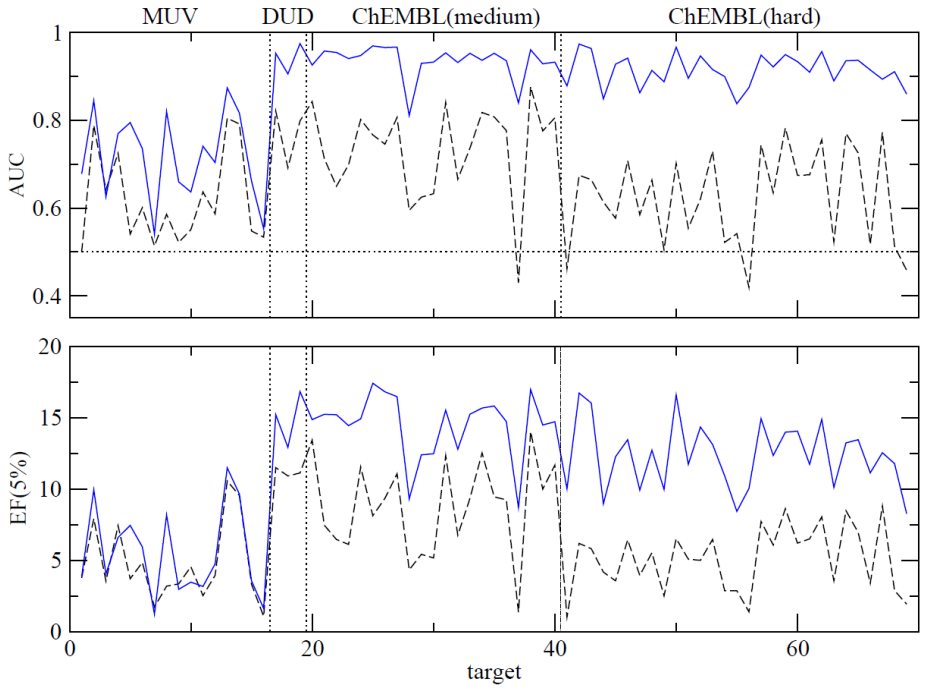
S. Riniker, N. Fechner, G. Landrum, *J. Chem. Inf. Model*. (2013), *submitted*.

NOVARTIS

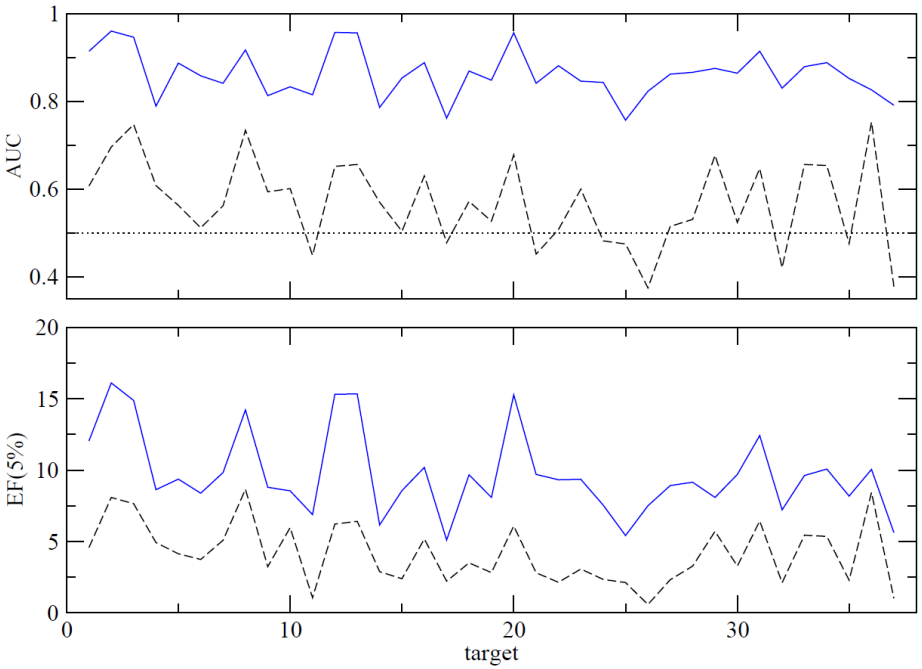
Heterogeneous Classifier Fusion

Take advantage of differences among ML methods / fingerprints

Data sets I (difficulty filtered)



Data sets II



rank-based MAX fusion of the three models

→ Large increase compared to standard fingerprint

ECFP4,
Fusion

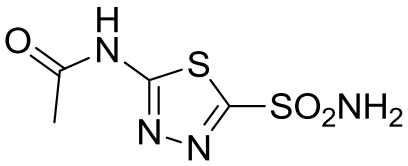
S. Riniker, N. Fechner, G. Landrum, *J. Chem. Inf. Model*. (2013), *submitted*.

NOVARTIS

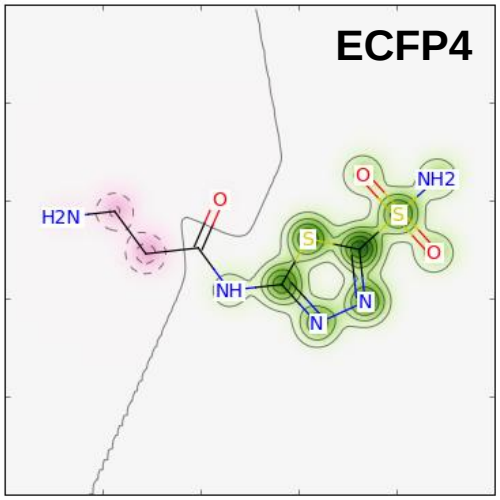
Why Do Machine-Learning Methods Help?

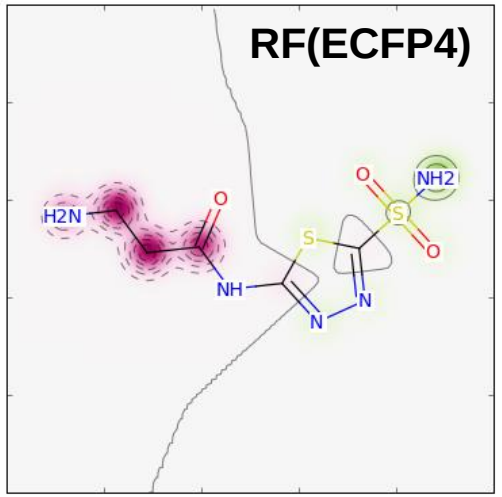
Similarity maps of Carbonic anhydrase II ligands

training active

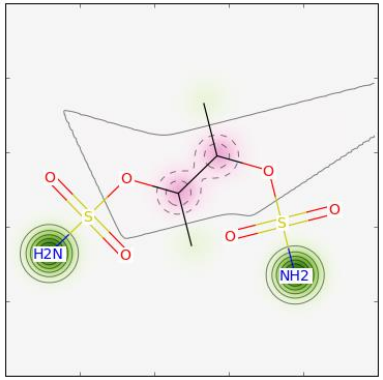


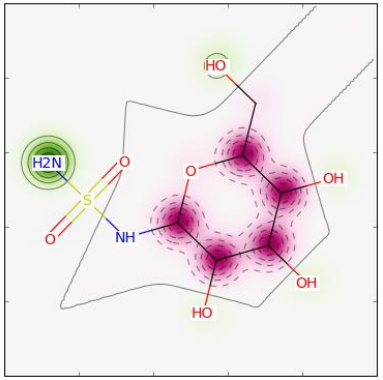
s_{Dice}(ECFP4) = 0.77
probability(RF) = 0.61

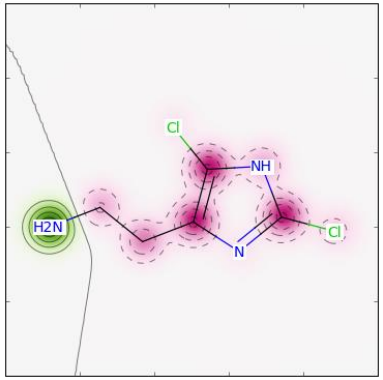




other actives found by the random forest (RF) trained with ECFP4







➔ RF picks up important motif for binding (zinc ion in active site)

green: removing bits *de*creases similarity; **pink**: removing bits *in*creases similarity

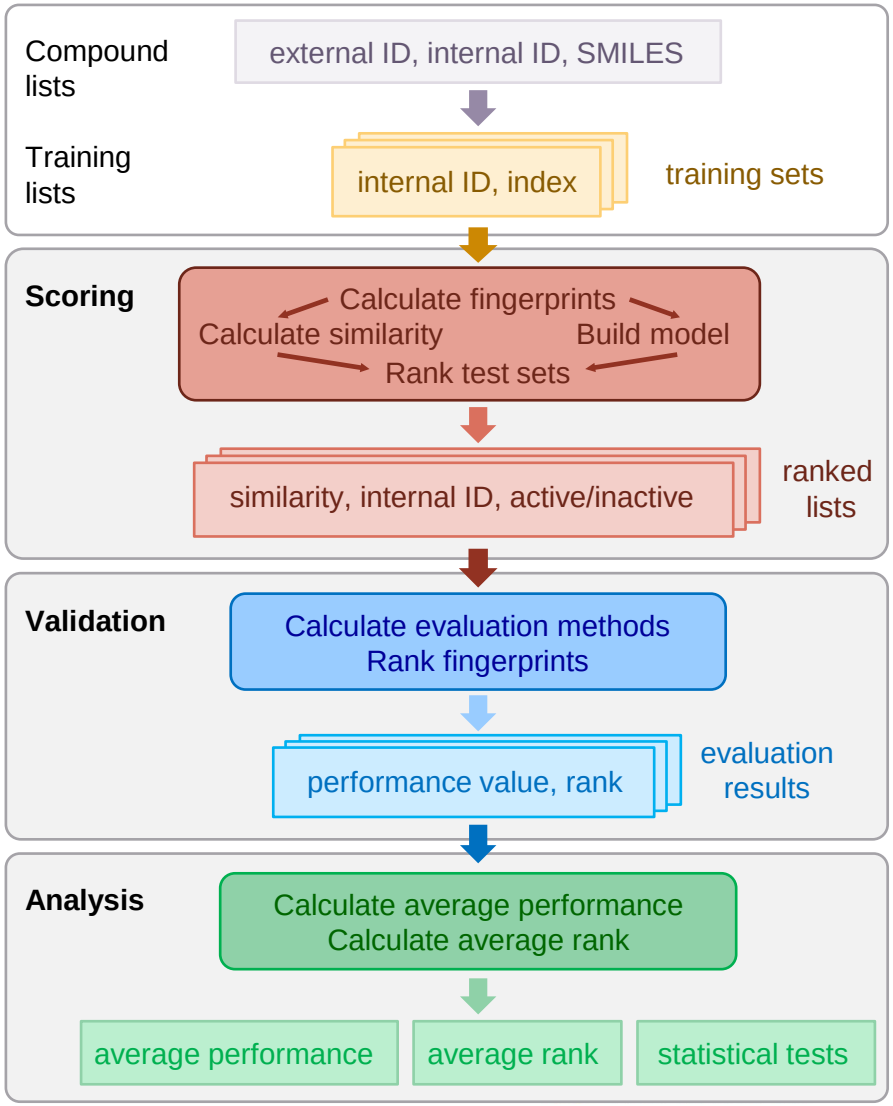
S. Riniker, G. Landrum, *J. Cheminf.* (2013), **5**, 43.

NOVARTIS

Benchmarking Platform

For ligand-based virtual screening

- Virtual screening in 3 steps: scoring, validation, analysis
- For standard fingerprints and machine-learning methods
- Access to source code and data sets:
  - supplementary material of *J. Cheminf*. (2013), **5**, 26
  - from github: rdkit/benchmarking_platform



S. Riniker, G. Landrum, *J. Cheminf*. (2013), **5**, 26.
S. Riniker, N. Fechner, G. Landrum, *J. Chem. Inf. Model.* (2013), *subm.*

Benchmarking Platform

For ligand-based virtual screening

- Files:
  - configuration_file.py
  - compounds/ → ChEMBL/, MUV/, DUD/
  - query_lists/ → ChEMBL/, MUV/, DUD/
  - scoring/ → calculate_scored_lists.py, fingerprint_lib.py
  - validation/ → calculate_validation_methods.py
  - analysis/ → run_analysis.py, run_method_summary.py, run_fp_summary.py, run_stat_analysis.py, Friedman_Test.Target_x_FP_Data.R
- Data sets from three public data sources:
  - MUV:[1] 17 targets
  - DUD:[2] 21 targets
  - ChEMBL:[3] 50 medium (+ 30 hard targets)

[1] S. G. Rohrer, K. Baumann, *J. Chem. Inf. Model*. (2009), **49**, 169.
[2] A. Zahn *et al*., *J. Cheminf*. (2009), **1**, 14.
[3] K. Heikamp, J. Bajorath, *J. Chem. Inf. Model*. (2011), **51**, 1831.

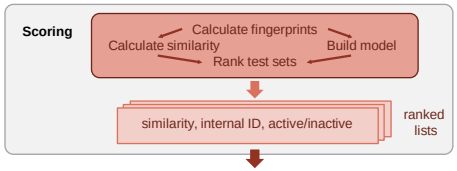
Benchmarking Platform

For ligand-based virtual screening

- Three subsets:
  - **data sets I (original):** (currently available)
    - 17 MUV targets
    - 21 DUD targets
    - 50 medium ChEMBL targets
  - **data sets I (difficulty filtered):** AUC(ECFC0) < 0.8
    - 16 MUV targets
    - 3 DUD targets
    - 21 medium ChEMBL targets
    - 29 hard ChEMBL targets
  - **data sets II:**
    - 37 ChEMBL targets
- Coming soon:
  - additional ChEMBL targets and ML-adapted scoring scripts

1. Step: Scoring – 2D Fingerprints

Fingerprint library



- In calculate_scored_lists.py:

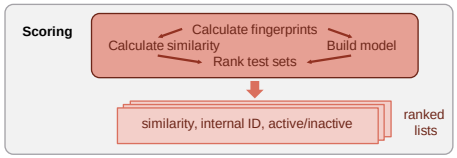
```
import fingerprint_lib

def getFPDict(fpnames, smiles):
  fp_dict = {}
  for fp in fpnames:
    fp_dict[fp] = fingerprint_lib.CalculateFP(fp, smiles)
  return fp_dict
```

- In fingerprint_lib.py:
  - Fingerprint library using other sources can be used as well
    - needs only to provide a function `CalculateFP` (input: name, SMILES; output: fp)

```
from rdkit import Chem, MACCSkeys
from rdkit.Chem import rdMolDescriptors as rdM

fpdict = {}
fpdict['ecfc0'] = lambda m: rdM.GetMorganFingerprint(m, 0)
fpdict['ecfp4'] = lambda m: rdM.GetMorganFingerprintAsBitVect(m, 2, nBits=nbits)
fpdict['maccs'] = lambda m: MACCSkeys.GenMACCSKeys(m)
fpdict['rdk5'] = lambda m: Chem.RDKFingerprint(m, maxPath=5, fpSize=nbits)
fpdict['hashap'] = lambda m: rdM.GetHashedAtomPairFingerprintAsBitVect(m,nBits=nbits)

def CalculateFP(fp_name, smiles):
  m = Chem.MolFromSmiles(smiles)
  return fpdict[fp_name](m)
```

1. Step: Scoring – 2D Fingerprints

Similarity metric



- In calculate_scored_lists.py:

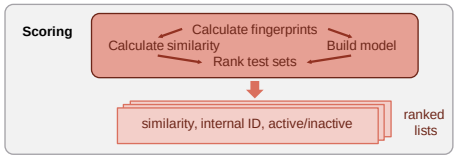
```
simil_dict = {}
simil_dict['Dice'] = lambda x,y: DataStructs.BulkDiceSimilarity(x,y)
simil_dict['Tanimoto'] = lambda x,y: DataStructs.BulkTanimotoSimilarity(x,y)
simil_dict['Manhattan'] = lambda x,y: DataStructs.BulkAllBitSimilarity(x,y)
simil_dict['Cosine'] = lambda x,y: DataStructs.BulkCosineSimilarity(x,y)

def getBulkSimilarity(fp, fp_list, simil_metric):
  simils = simil_dict[simil_metric](fp, fp_list)
  simils.sort(reverse=True)
  return simils
```

  - more RDKit similarity metrics:
    - Russel
    - Kulczynski
    - McConnaughey
    - RogotGoldberg
  - other similarity/distance metrics can be added

NOVARTIS

1. Step: Scoring – ML Methods

Changes to calculate_scored_lists.py



- Using scikit-learn
- Only a single fingerprint is calculated

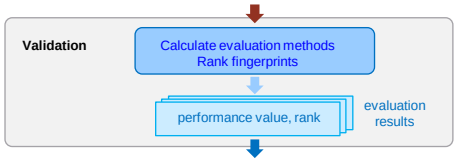
```
import fingerprint_lib

def getFP(fpname, smiles):
  return fingerprint_lib.CalculateFP(fpname, smiles)
```

- Fingerprints are converted to numpy arrays (for speed)

```
def getNumpy(inlist):
  outlist = []
  for fp in inlist:
    arr = numpy.zeros((1,))
    DataStructs.ConvertToNumpyArray(fp[1], arr) # fp is second element of i
    outlist.append(arr)
  return outlist
```

- ML model is retrained with new training molecules each repetition

2. Step: Validation



- Required input: pickled file per target
  - entry for each fingerprint → [fpname, scored_lists]
  - scored_lists = list with ranked list for each repetition (i.e. 50 elements)
  - each repetition = ranked list of length $N_{\text{test molecules}}$
  - element of ranked list = [similarity, {other similarities...}, internal ID, 0/1]

```
import gzip, cPickle

scores = {}
myfile = gzip.open(scored_list, 'r')
while 1:
  try: tmp = cPickle.load(myfile)
  except (EOFError): break
  else: scores[tmp[0]] = tmp[1] # [fpname, list of scored lists]
```

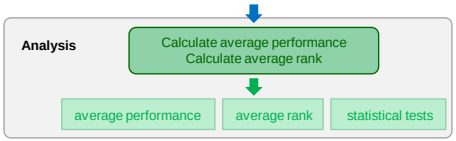
- Evaluation methods: `rdkit.ML.Scoring`

```
from rdkit.ML.Scoring import Scoring

results['AUC'] = Scoring.CalcAUC(scores[fp][q], -1)
results['EF5'] = Scoring.CalcEnrichment(scores[fp][q], -1, [0.05])[0]
results['RIE20'] = Scoring.CalcRIE(scores[fp][q], -1, 20.0)
results['BEDROC20'] = Scoring.CalcBEDROC(scores[fp][q], -1, 20.0)
```


NOVARTIS

3. Step: Analysis

Workflow



validation

- dict (methods) of dicts (fingerprints) of lists (repetitions) → [score, rank]

run_analysis.py

- file with (fps x method) per **target**

→ in a directory `data set`

- file with <rank> over targets (fps x method)
- csv with (fps and rank x (target x rep)) per method → in directory `stat_analysis`

run_method_summary.py

- file with (fps x target) per **method**

→ in directory `method_summary`

run_fp_summary.py

- file with (method x target) per **fps**

→ in directory `fp_summary`

run_stat_analysis.py

- csv with <rank> (fps x fps) per **method**

→ in directory `stat_analysis`

- csv with p-values (fps x fps) per **method**

→ in directory `stat_analysis`

- file with correlation (fps x fps) per **method**

→ in directory `stat_analysis`

Friedman_Test.Target_x_FP_Data.R

NOVARTIS

Similarity Maps

Code example for standard fingerprints

```
from rdkit import Chem, DataStructs
from rdkit.Chem.Draw import SimilarityMaps as rdSM

mol = Chem.MolFromSmiles('c1ccccc1')
refm = Chem.MolFromSmiles('c1ccncc1')

# atom pairs
fig, maxweight = rdSM.GetSimilarityMapsForFingerprint(refm, mol,
   lambda x,y: rdSM.GetAPFingerprint(x, y, fpType='normal'),
   metric=DataStructs.DiceSimilarity)
```

- Atom pairs: `SimilarityMaps.GetAPFingerprint`
  - fpType = '**normal**', 'hashed', 'bv'
- Torsions: `SimilarityMaps.GetTTFingerprint`
  - fpType = 'normal', 'hashed', 'bv'
- Circular fingerprints: `SimilarityMaps.GetMorganFingerprint`
  - fpType = '**bv**', 'count'
  - useFeatures = **False**, True
- Similarity metric: any function taking two fps and returning a float
  - Default: Dice similarity

S. Riniker, G. Landrum, *J. Cheminf*. (2013), **5**, 43.

Similarity Maps

Code example for random forest

```
from rdkit import Chem, DataStructs
from rdkit.Chem.Draw import SimilarityMaps as rdSM
from sklearn.ensemble import RandomForestClassifier
import cPickle

mol = Chem.MolFromSmiles('c1ccccc1')
rf = cPickle.load(open('rf.pkl', 'r')) # a previously trained RF model

# metric
def getProba(fp, predictionFunction):
  return predictionFunction(fp)[0][1]

# with Morgan fingerprint
fig, maxweight = rdSM.GetSimilarityMapsForModel(mol,
   lambda x,y: rdSM.GetMorganFingerprint(x, y, radius=2, fpType='bv'),
   lambda x: getProba(x, rf.predict_proba))
```

- Same fingerprint functions
- «Similarity» metric: any function taking a fp and returning a float
  - no default
  - sklearn models return an array [n_samples, n_classes]
  - for naïve Bayes: use `nb.predict_log_proba`

S. Riniker, G. Landrum, *J. Cheminf*. (2013), **5**, 43.

Summary

- Benchmarking platform in 3 steps: scoring, validation, analysis
- *Inter*-target differences larger than *intra*-target differences between standard fingerprints
  - TT (from 1987) among the top fingerprints in all evaluation methods
- Evaluation methods:
  - high correlations between the «early-recognition» methods
- Use of machine-learning methods increases performance
- Similarity maps:
  - visualization of atomic contributions to similarity or predicted probability of machine-learning models

Acknowledgements

- Greg Landrum
- Nikolas Fechner
- Hanspeter Gubler
- Michael Berthold (Uni Konstanz)

NOVARTIS