OpenML

OPEN, AUTOMATED MACHINE LEARNING

JOAQUIN VANSCHOREN, TU/E

www.openml.org

@open_ml

OpenML

You can be part of this presentation :)
Follow the code examples:

- **On Google Colab: goo.gl/VwbKb4**
- **On Github: https://git.io/fA3eL**

Machine Learning

Data is the new soil. We need algorithms that extract the right information to understand and interact with the world

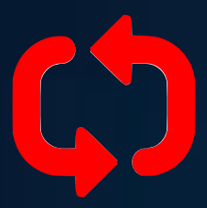
Each algorithm thrives on specific types of data. Predicting which ones will thrive is not easy.

We need clear experiments trying out algorithms on data, and analyze the results together

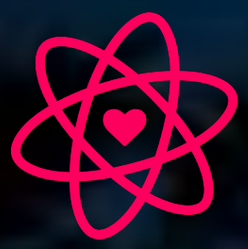
Unnecessary friction keeps us in the dark

Scattered, ill-described, datasets
Manual searching, reformatting, making assumptions

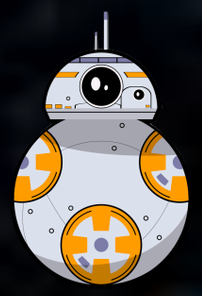
Myriad algorithms, versions, languages
Write code, set up experiments, store results,…



Reproducibility is **hard**
Manually tracking every detail is error-prone



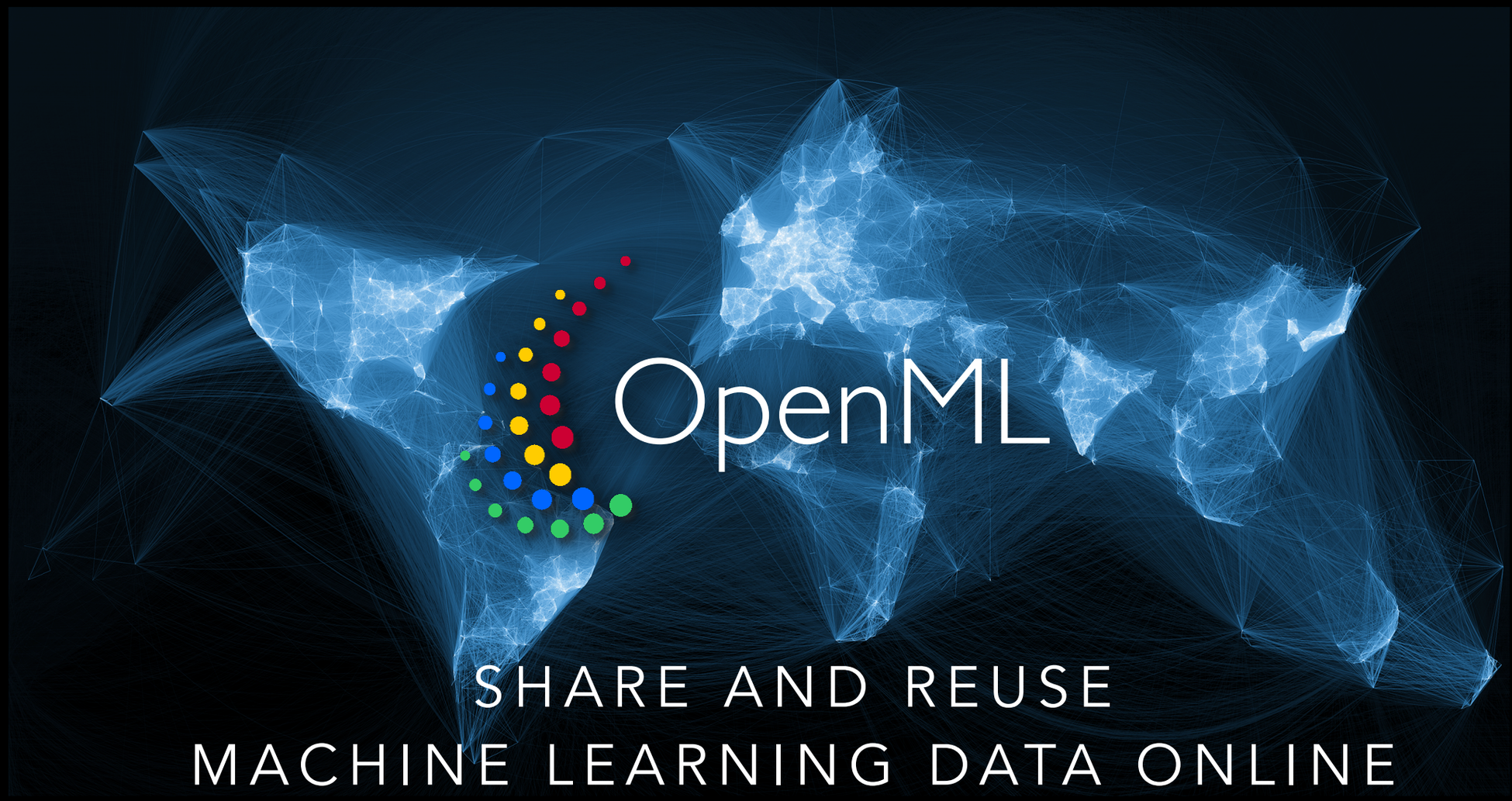
Hard to find and reuse prior results
No standards / hubs for sharing and organizing results



Hard to automate model building end-to-end
Requires automated data organization, clean APIs,…

What if…
we should easily share and reuse
machine learning datasets,
algorithms, and models?

Networked science: share and organize (all) data online



Easy to use: Integrated in many ML tools/environments

Easy to contribute: Automated sharing of data, code, results

Organized data: APIs to find & reuse data, models, experiments

Reward structure: Track your impact, build reputation

Self-learning: Learn from many experiments to help people

OpenML Search GUIDE SIGN IN

www.openml.org

OpenML beta

Machine learning, better, together

19903 data sets

Find or add **data** to analyse

46553 tasks

Download or create scientific **tasks**

4539 flows

Find or add data analysis **flows**

5913832 runs

Upload and explore all **results** online.

HACKATHON

Bring your own data, bring your own algorithms, or build cool new features.

Next location: 9-14 October in Leiden, the Netherlands

OpenML: Components

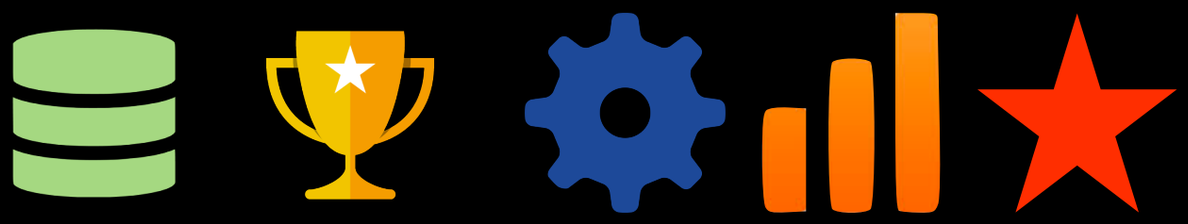
Datasets: Auto-annotated, organized, well-formatted
Easily find and load datasets + meta-data

Tasks: Auto-generated, machine-readable
Everyone's results are comparable, reusable

Flows: Uniform description of ML pipelines
Run locally (or wherever), auto-upload globally

Runs: All results from running flows on tasks
All details needed for tracking and reproducibility
Evaluations can be queried, compared, reused

OPENML API



REST (XML/JSON), PYTHON, R, JAVA,...

Interact with OpenML any way you want. All data is open.

- Search Data, Flows, Models, Evaluations
- Download data, meta-data and runs
- Upload new data, flows, runs
- Program against OpenML (e.g. easy benchmarking)
- Build your own bots that automate processes
  - Data cleaning, annotation, model building, evaluation

It starts with data

It starts with data

Data (tabular) easily uploaded or referenced (URL)

OpenML

```
from openml import datasets
my_data = datasets.OpenMLDataset(data_file='omg.arff',
                                 name='OMG_data',
                                 description='We want to...',
                                 licence='Public')
response = my_data.publish()
```

Name	Version

auto-versioned, analysed, organised online

OpenML Search

19587 results

Only showing active datasets (public or shared with you).

FILTERS SORT: **MOST RUNS** ID'S TABLE + ADD NEW

credit-g (1)
This dataset classifies people described by a set of attributes as good or bad c...
★ **439801 runs** ♥ **2 likes** **39 downloads** **41 reach** **11 impact**
1000 instances - 21 features - 2 classes - 0 missing values

blood-transfusion-service-c...
Data taken from the Blood Transfusion Service Center in Hsin-Chu City in Taiw...
★ **424540 runs** ♥ **1 likes** **21 downloads** **22 reach** **13 impact**
748 instances - 5 features - 2 classes - 0 missing values

Search (API)

Python, R, Java, C#

```
import openml as oml
openml_list = oml.datasets.list_datasets()
```

```
Found 2522 datasets
```

	did	name	NumberOfInstances	NumberOfFeatures	NumberOfClasses
2	2	anneal	898	39	5
3	3	kr-vs-kp	3196	37	2
4	4	labor	57	17	2
5	5	arrhythmia	452	280	13
6	6	letter	20000	17	26
7	7	audiology	226	70	24

Search (API)

Python, R, Java, C#

```
openml_list = oml.datasets.list_datasets(
    number_instances = '10000..20000',
    number_features = '10..20')
```

```
Found 11 datasets
```

	did	name	NumberOfInstances	NumberOfFeatures	NumberOfClasses
6	6	letter	20000	17	26
32	32	pendigits	10992	17	10
216	216	elevators	16599	19	61
846	846	elevators	16599	19	2
977	977	letter	20000	17	2
1019	1019	pendigits	10992	17	2
1120	1120	MagicTelescope	19020	12	2
1199	1199	BNG(echoMonths)	17496	10	17491
1222	1222	letter-challenge-unlabeled.arff	20000	17	3

Search (API)

Python, R, Java, C#

```
openml_list = oml.datasets.list_datasets(
                    data_name = 'eeg-eye-state')
```

	did	name	NumberOfInstances	NumberOfFeatures	NumberOfClasses
1471	1471	eeg-eye-state	14980.0	15.0	2.0

data id

Get (API)

Python, R, Java, C#

```
dataset = oml.datasets.get_dataset(1471)
dataset.description[:500]
```

```
**Author**: Oliver Roesler, it12148'@'lehre.dhbw-stuttgart.de
**Source**: [UCI](https://archive.ics.uci.edu/ml/datasets/EEG+E
rsity (DHBW), Stuttgart, Germany
**Please cite**:

All data is from one continuous EEG measurement with the Emotiv
 117 seconds. The eye state was detected via a camera during th
 after analysing the video fr
```

Get (API)

Python, R, Java, C#

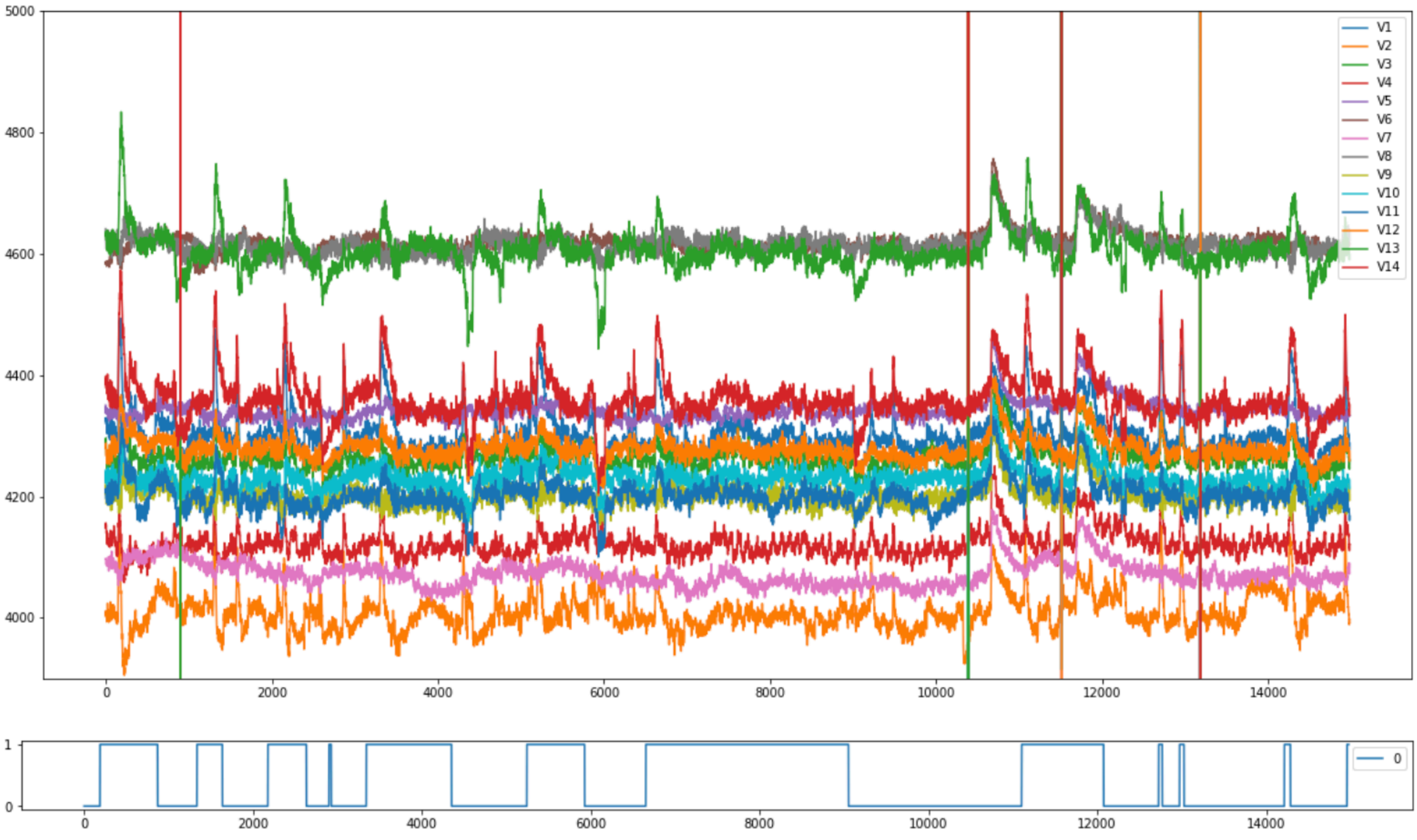
```
X, y, attribute_names = dataset.get_data(
    return_attribute_names=True)
eeg = pd.DataFrame(X, columns=attribute_names)
eeg['class'] = y
```

```
           V1           V2           V3           V4           V5  \
0  4329.229980  4009.229980  4289.229980  4148.209961  4350.259766
1  4324.620117  4004.620117  4293.850098  4148.720215  4342.049805
2  4327.689941  4006.669922  4295.379883  4156.410156  4336.919922
3  4328.720215  4011.790039  4296.410156  4155.899902  4343.589844
4  4326.149902  4011.790039  4292.310059  4151.279785  4347.689941
5  4321.029785  4004.620117  4284.100098  4153.330078  4345.640137
6  4319.490234  4001.030029  4280.509766  4151.790039  4343.589844
7  4325.640137  4006.669922  4278.459961  4143.080078  4344.100098
8  4326.149902  4010.770020  4276.410156  4139.490234  4345.129883
9  4326.149902  4011.280029  4276.919922  4142.049805  4344.100098

           V6           V7           V8           V9           V10  \
0  4586.149902  4096.919922  4641.029785  4222.049805  4238.459961
```

Get (API)

Python, R, Java, C#

```
eeg.plot()
pd.DataFrame(y).plot()
```



Get (API)

Python, R, Java, C#

```
dataset.qualities
```

130+ data properties (meta-features)

```
'NumberOfBinaryFeatures': 1.0,
'NumberOfClasses': 2.0,
'NumberOfFeatures': 15.0,
'NumberOfInstances': 14980.0,
'NumberOfInstancesWithMissingValues': 0.0,
'NumberOfMissingValues': 0.0,
'NumberOfNumericFeatures': 14.0,
'NumberOfSymbolicFeatures': 1.0,
'PercentageOfBinaryFeatures': 6.666666666666667,
'PercentageOfInstancesWithMissingValues': 0.0,
'PercentageOfMissingValues': 0.0,
'PercentageOfNumericFeatures': 93.33333333333333,
'PercentageOfSymbolicFeatures': 6.666666666666667,
'Quartile1AttributeEntropy': nan,
'Quartile1KurtosisOfNumericAtts': 2712.4725059647635,
'Quartile1MeansOfNumericAtts': 4193.079256508685,
'Quartile1MutualInformation': nan,
'Quartile1SkewnessOfNumericAtts': 22.472026210607 97,
'Quartile1StdDevOfNumericAtts': 37.98467231888507,
```

Via sklearn (0.20+)

Python, R, Java, C#

```
from sklearn.datasets import fetch_openml
mice_data = fetch_openml(name='miceprotein',
                                    version=4)
mice_data.details
```


{'id': '40966',
 'name': 'MiceProtein',
 'version': '4',
 'format': 'ARFF',
 'upload_date': '2017-11-08T16:00:15',
 'licence': 'Public',
 'url': 'https://openml2.win.tue.nl/data/v1/download/17928620/
MiceProtein.arff',
 'file_id': '17928620',
'default_target_attribute': 'class',
'row_id_attribute': 'MouseID',

Fit (API)

Python, R, Java, C#

```
from sklearn import neighbors
clf = neighbors.KNeighborsClassifier()
clf.fit(X, y)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=1, p=2,
           weights='uniform')
```

Complete code to build a model, automatically, anywhere

```
import openml as oml
from sklearn import neighbors, tree

dataset = oml.datasets.get_dataset(1471)
X, y = dataset.get_data()
clf = neighbors.KNeighborsClassifier()
clf.fit(X, y)
```

Tasks

auto-benchmarking and collaboration

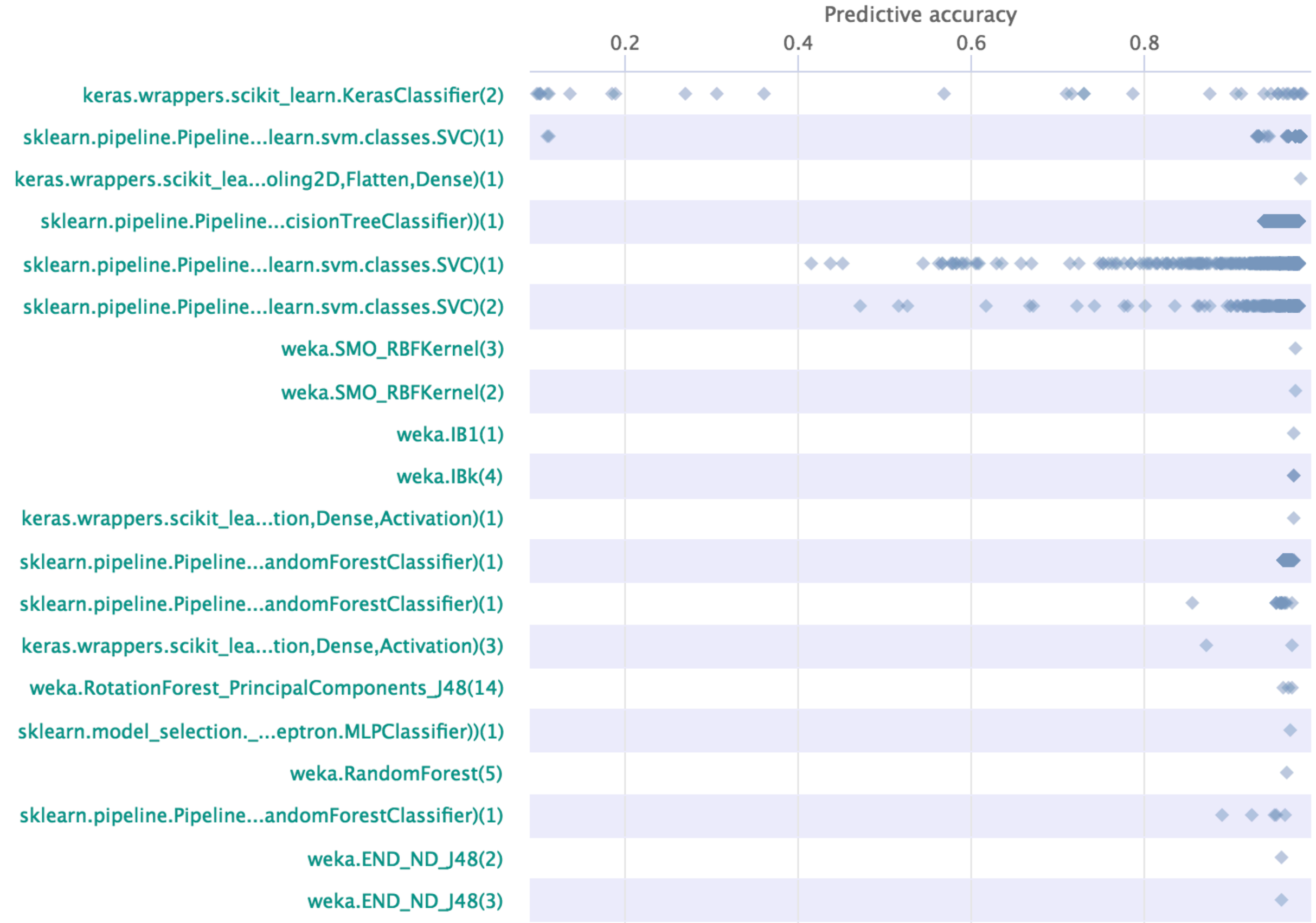
Tasks contain data, goals, procedures.

Auto-build + evaluate models correctly

All evaluations are directly comparable

Evaluations per flow (multiple parameter settings)

every point is a run, click for details



Search

```
task_list = oml.tasks.list_tasks(size=5000)
```

First 5 of 5000 tasks:

tid	did	name	task_type	estimation_procedure	evaluation_measures
2	2	2	anneal	Supervised Classification	10-fold Crossvalidation	predictive_accuracy
3	3	3	kr-vs-kp	Supervised Classification	10-fold Crossvalidation	predictive_accuracy
4	4	4	labor	Supervised Classification	10-fold Crossvalidation	predictive_accuracy
5	5	5	arrhythmia	Supervised Classification	10-fold Crossvalidation	predictive_accuracy
6	6	6	letter	Supervised Classification	10-fold Crossvalidation	predictive_accuracy

task id

Search

```
mytasks = pd.DataFrame.from_dict(task_list)
mytasks.query('name=="eeg-eye-state"')
```

	tid	did	name	task_type	estimation_procedure	evaluation_measures
9983	9983	1471	eeg-eye-state	Supervised Classification	10-fold Crossvalidation	predictive_accuracy
14951	14951	1471	eeg-eye-state	Supervised Classification	10-fold Crossvalidation	NaN

Get

```
task = oml.tasks.get_task(14951)
```

```
{'class_labels': ['1', '2'],
 'cost_matrix': None,
 'dataset_id': 1471,
 'estimation_parameters': {'number_folds': '10',
                           'number_repeats': '1',
                           'percentage': '',
                           'stratified_sampling': 'true'},
 'estimation_procedure': {'data_splits_url': 'https://www.ope
```

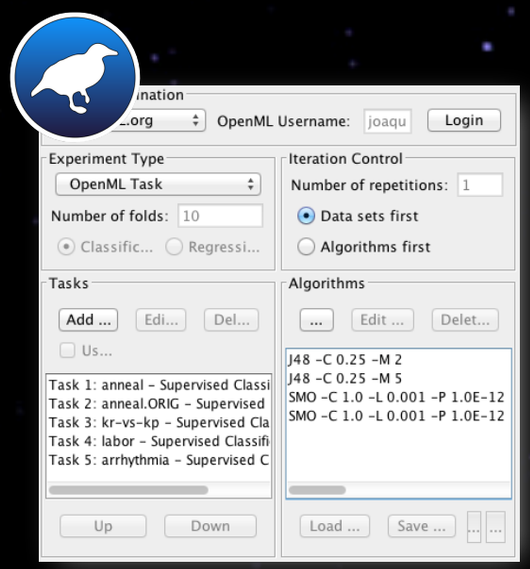
Flows

Run experiments ***locally***, share them ***globally***

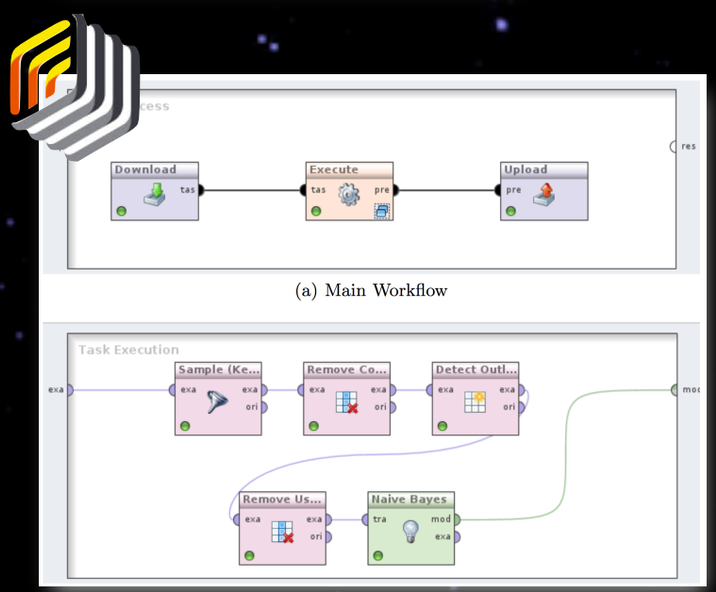
Auto-run algorithms/workflows on any task

Integrated in many machine learning tools (+ APIs)

Integrated in many machine learning tools (+ APIs)




```python
import openml as oml
from sklearn import tree

task = oml.tasks.get_task(14951)
clf = tree.ExtraTreeClassifier()
flow = oml.flows.sklearn_to_flow(clf)
run = oml.runs.run_flow_on_task(task, flow)
myrun = run.publish()
```



(a) Main Workflow

```r
library(OpenML)
library(mlr)
task = getOMLTask(10)
lrn = makeLearner("classif.rpart")
res = runTaskMlr(task, lrn)
run.id = uploadOMLRun(res)
```

Fit and share (*complete* code)

```
import openml as oml
from sklearn import tree

task = oml.tasks.get_task(14951)
clf = tree.ExtraTreeClassifier()
flow = oml.flows.sklearn_to_flow(clf)
run = oml.runs.run_flow_on_task(task, flow)
myrun = run.publish()
```

Uploaded to http://www.openml.org/r/9204488

Fit and share *pipelines*

```
from sklearn import pipeline, ensemble, preprocessing
from openml import tasks,runs, datasets
task = tasks.get_task(59)
pipe = pipeline.Pipeline(steps=[
            ('Imputer', preprocessing.Imputer()),
            ('OneHotEncoder', preprocessing.OneHotEncoder(),
            ('Classifier', ensemble.RandomForestClassifier())
            ])
flow = oml.flows.sklearn_to_flow(pipe)
run = oml.runs.run_flow_on_task(task, flow)
myrun = run.publish()
```

Uploaded to http://www.openml.org/r/7943199

Fit and share *deep learning models*

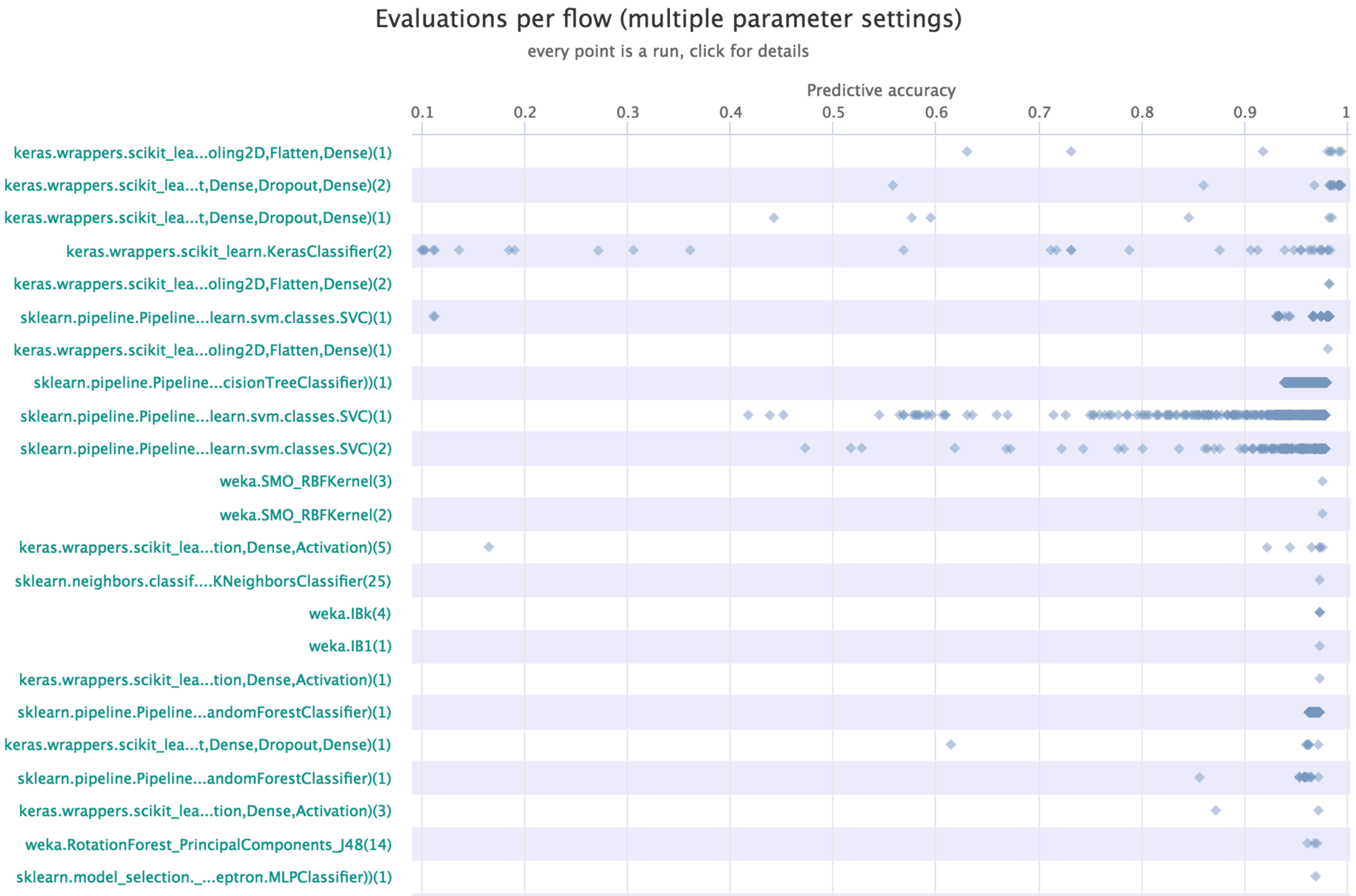
```
import keras
from keras.models import Sequential,
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPooling
from keras.layers.core import Activation

model = Sequential()
model.add(Reshape((28, 28, 1), input_shape=(784,)))
model.add(Conv2D(20, (5, 5), padding="same", input_shape=(28,28,1),
          activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Conv2D(50, (5, 5), padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Flatten())
model.add(Dense(500))
model.add(Activation('relu'))
model.add(Dense(10))
model.add(Activation('softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])
```

Fit and share *deep learning models*

```
task = tasks.get_task(3573) #MNIST
flow = oml.flows.keras_to_flow(model)
run = oml.runs.run_flow_on_task(task, flow)
myrun = run.publish()
```

Uploaded to https://www.openml.org/r/9204337

Compare to state-of-the-art



Runs

Share and reuse results

Experiments auto-uploaded, evaluated online

reproducible, linked to **data, flows, authors**

and **all other experiments**

Experiments auto-uploaded, evaluated online

Result files

Description

XML file describing the run, including user-defined evaluation measures.

Model readable

A human-readable description of the model that was built.

Model serialized

A serialized description of the model that can be read by the tool that generated it.

Predictions

ARFF file with instance-level predictions generated by the model.

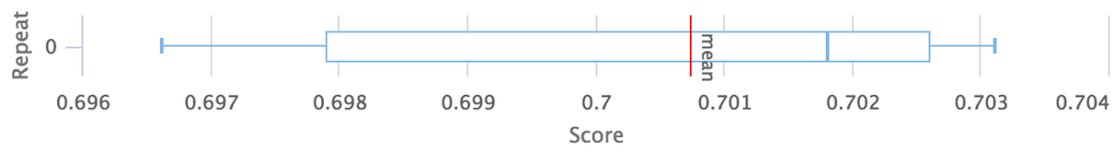
Area under ROC curve

0.7007 ± 0.0023

Per class

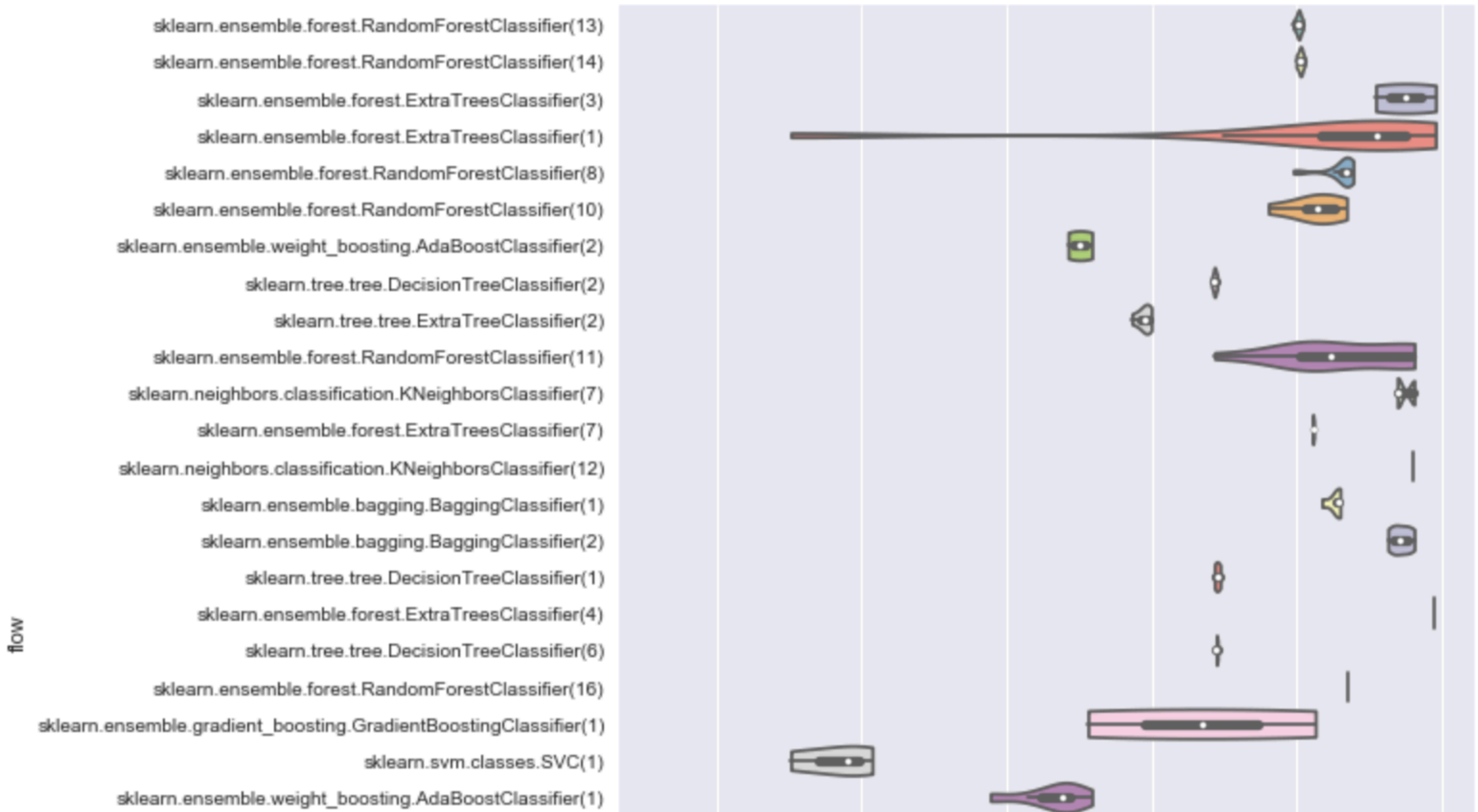
0	1
0.7007	0.7007

Cross-validation details (10-fold Crossvalidation)



★ Download, reuse runs

```
myruns = oml.runs.list_runs(task=[14951])
sns.violinplot(x="score", y="flow", data=pd.DataFrame(
```



The same algorithm can learn to walk in wildly different ways. YUVAL TASSA

Missing data hinder replication of artificial intelligence studies

By Matthew Hutson | Feb. 15, 2018 , 12:30 PM

website called OpenML. It hosts not only algorithms, but also data sets and more than 8 million experimental runs with all their attendant details. "The exact way that you run your experiments is full of undocumented assumptions and decisions," Vanschoren says. "A lot of this detail never makes it into papers."

Publishing, impact tracking

Heidi Seibold

PhD student in Computational Biostatistics at the University of Zurich. I am into R, open science and reproducible research.

University of Zurich Joined 2016-01-27

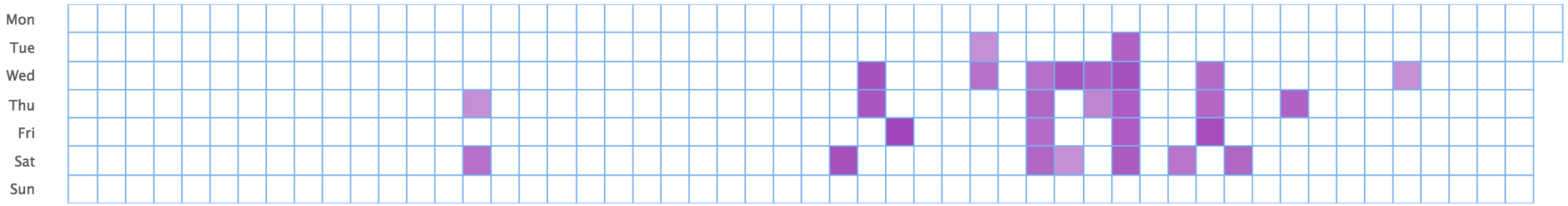
Activity	Reach	Impact	Uploads			
1649.5	12	195	1	35	0	1605

EDIT PROFILE

	Activity	Reach		Impact
Data Sets	1	4	0	0
Flows	35	7	0	195
Tasks	0	0	0	
Runs	1605	1	0	

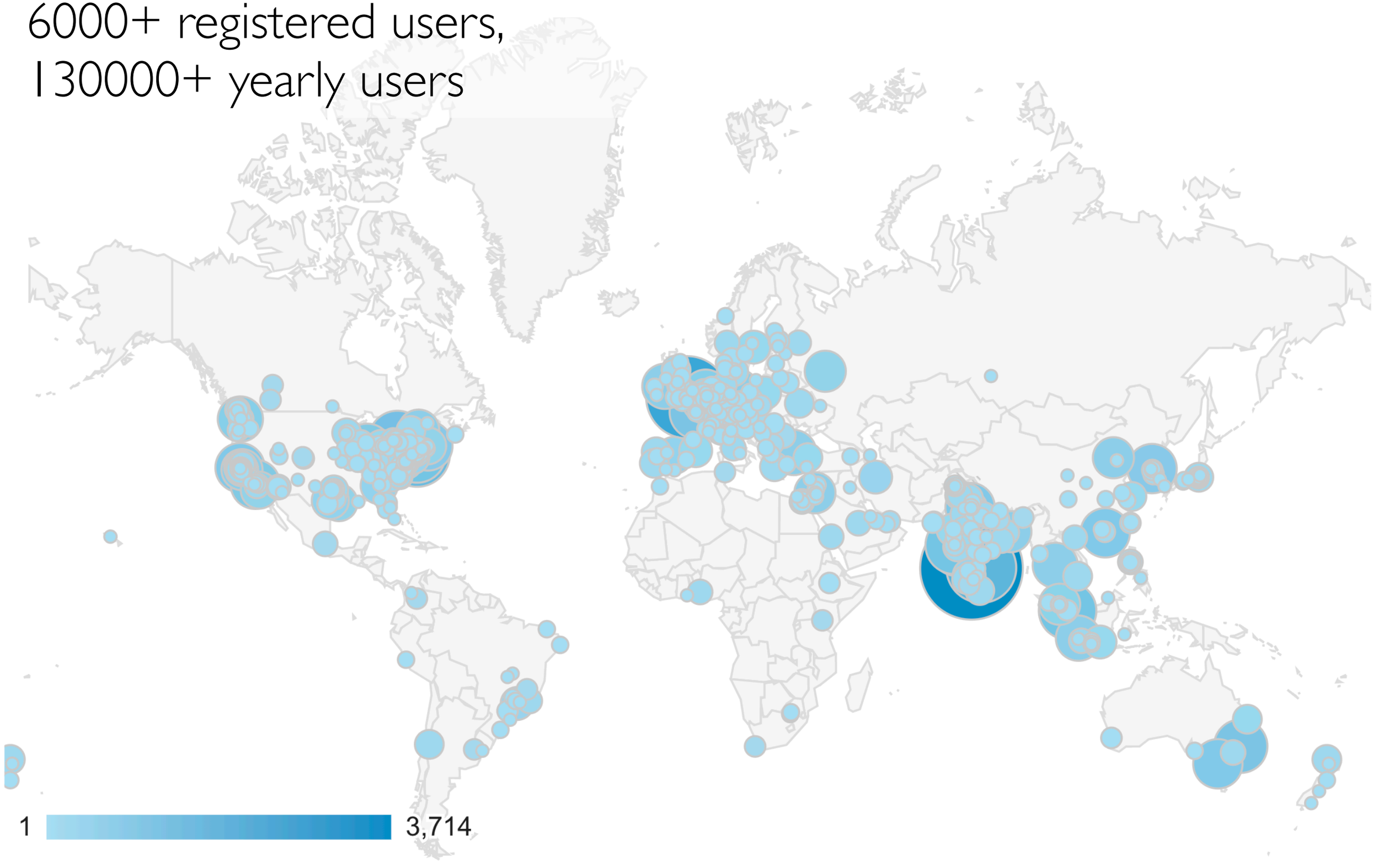
Activity: 1.65K 1.64K 0 17

Activity from Sunday 2016-05-29 to Monday 2017-05-29



OpenML Community

6000+ registered users,
130000+ yearly users



Automatic Machine Learning (AutoML)

Automatically find optimal machine learning algorithms, pipelines, neural architectures

Given dataset, search the space of all possible algorithms

(Not so) Automatic Machine Learning

AutoML systems still need
to overcome many sources of friction

Should not start from scratch every time
Collect and learn from *experience* (meta-data)

Meta-Learning

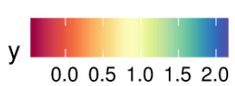
Learn how to learn

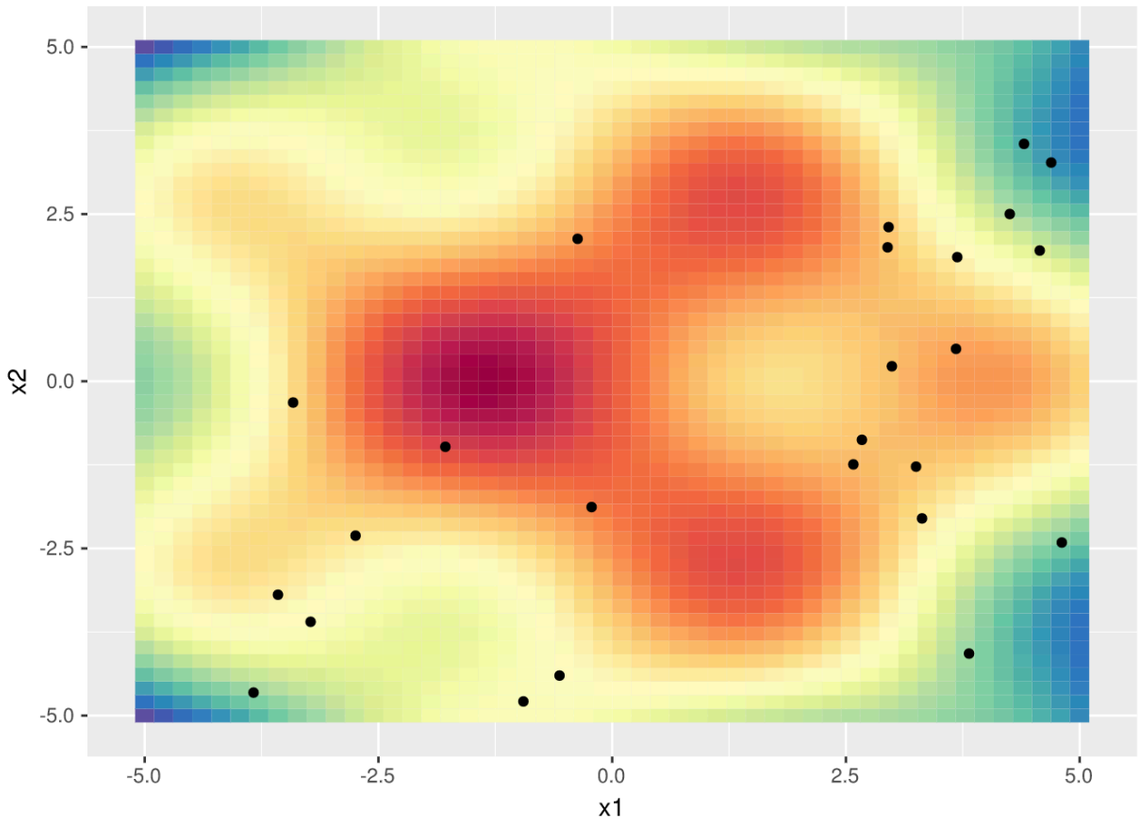
Transfer knowledge from old problems to new problems

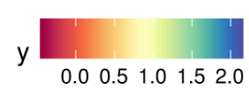
Collect data from previous tries learn to solve new tasks better

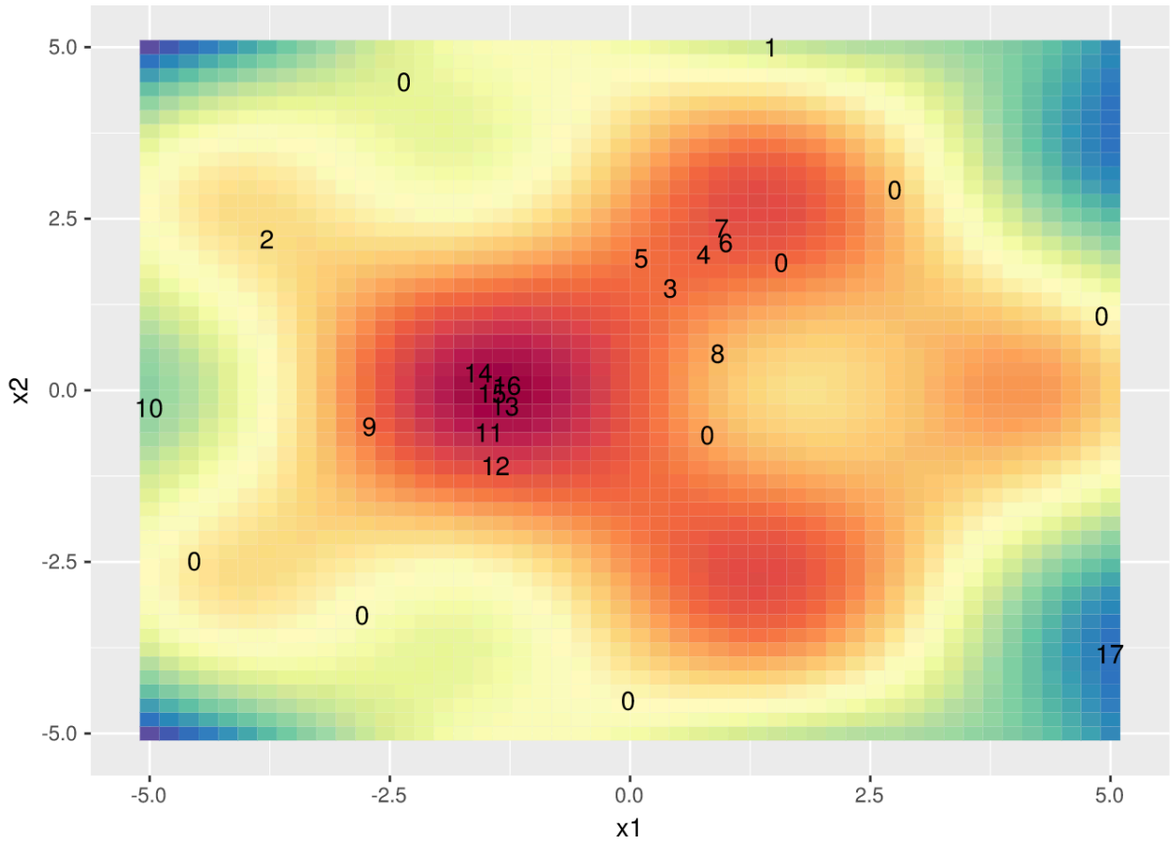
AUTOML: META-LEARNING

- **Find similar datasets**
  - 20,000+ versioned datasets, with 130+ meta-features
  - Instead of starting from scratch, start from configurations that worked well on *similar* datasets





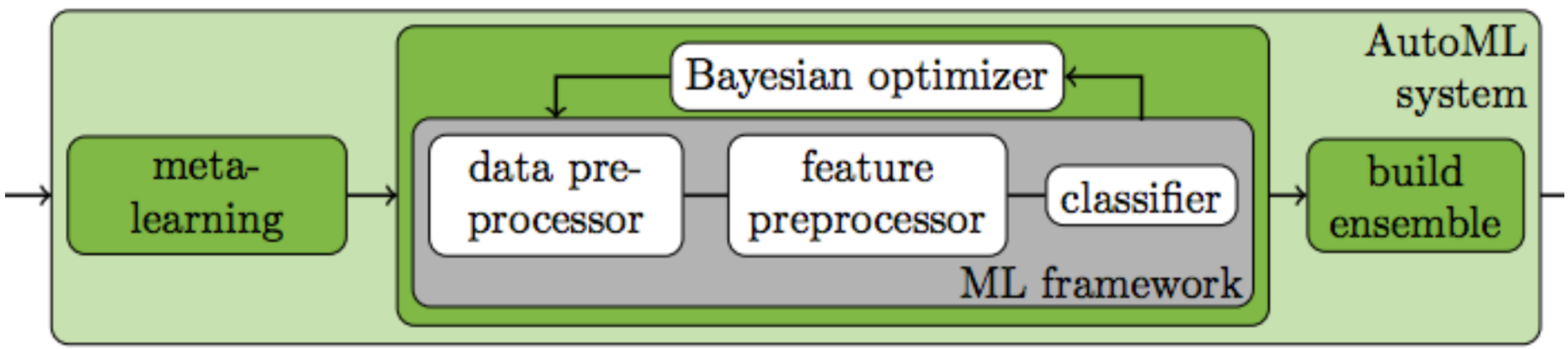




AUTOML: META-LEARNING

- **Find similar datasets**
  - 20,000+ versioned datasets, with 130+ meta-features
  - Instead of starting from scratch, start from configurations that worked well on *similar* datasets

- **Auto-sklearn** (AutoML challenge winner, NIPS 2016)
  - Lookup similar datasets, start with best pipelines

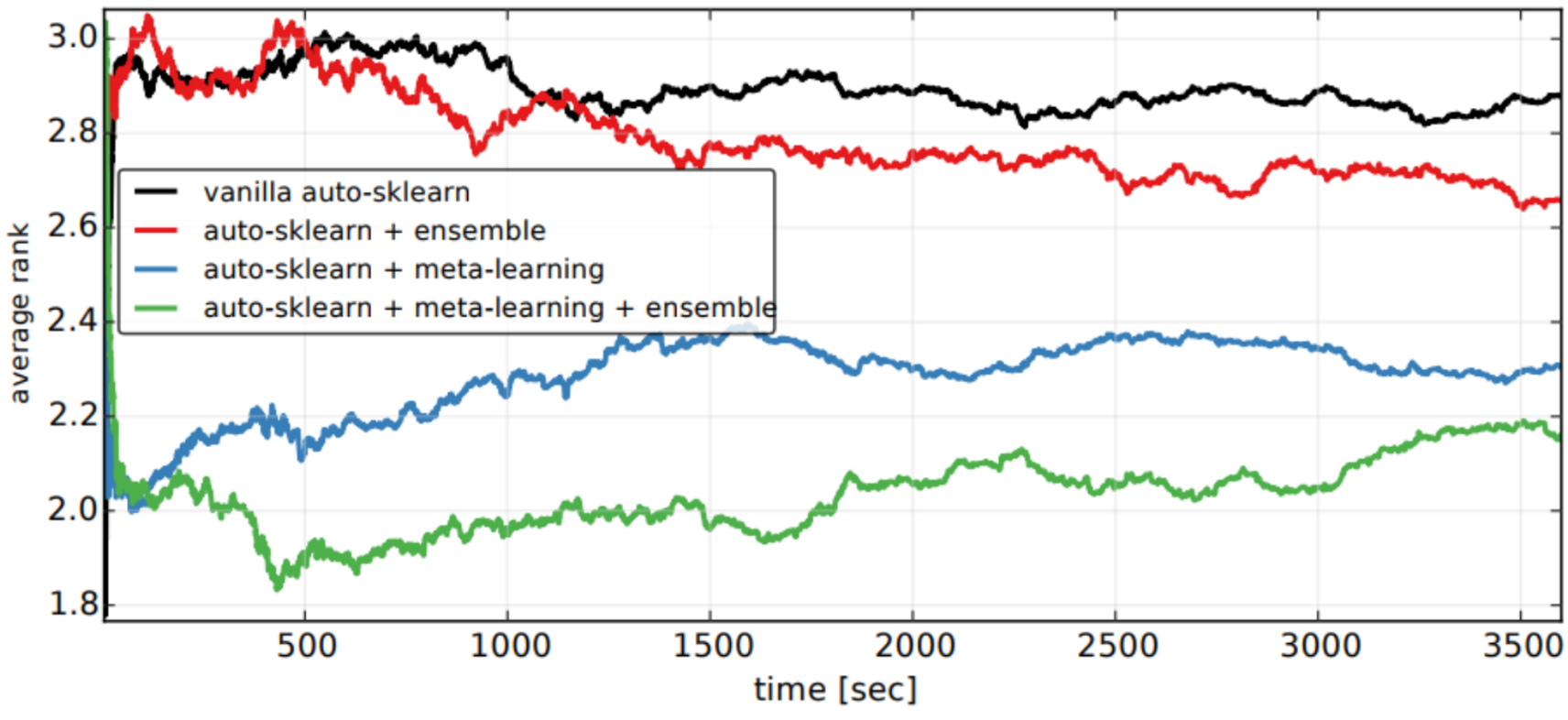


Matthias Feurer et al. (2016) NIPS

AUTOML: META-LEARNING

- **Find similar datasets**
  - 20,000+ versioned datasets, with 130+ meta-features
  - Instead of starting from scratch, start from configurations that worked well on *similar* datasets

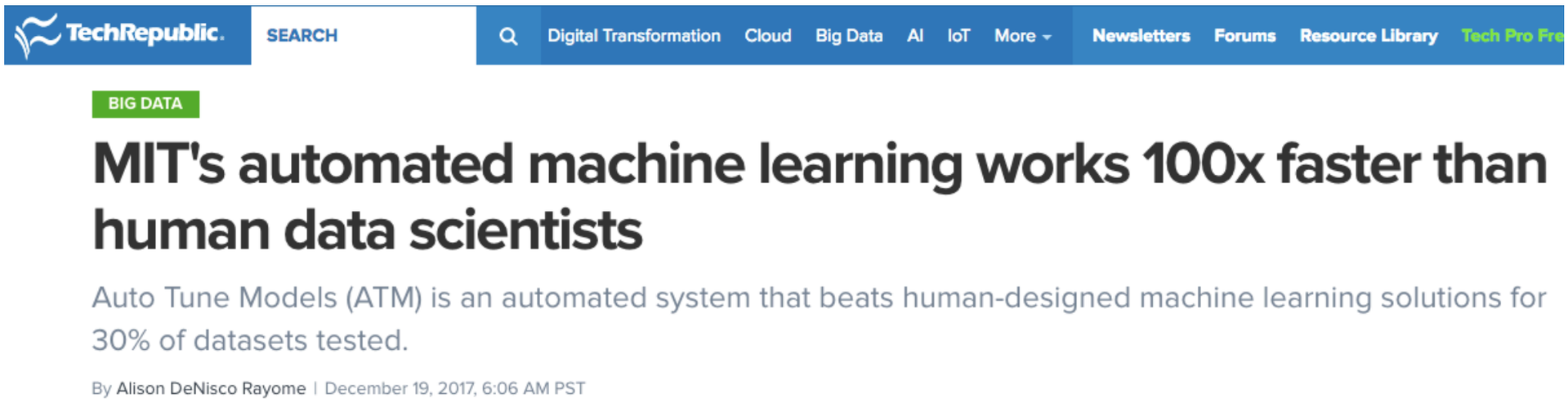
- **Auto-sklearn** (AutoML challenge winner, NIPS 2016)
  - Lookup similar datasets, start with best pipelines



AUTOML: META-LEARNING

- **Reuse (millions of) prior model evaluations:**
  - Benchmark new algorithms against state-of-the-art
  - Meta-models: E.g. predict performance or training time

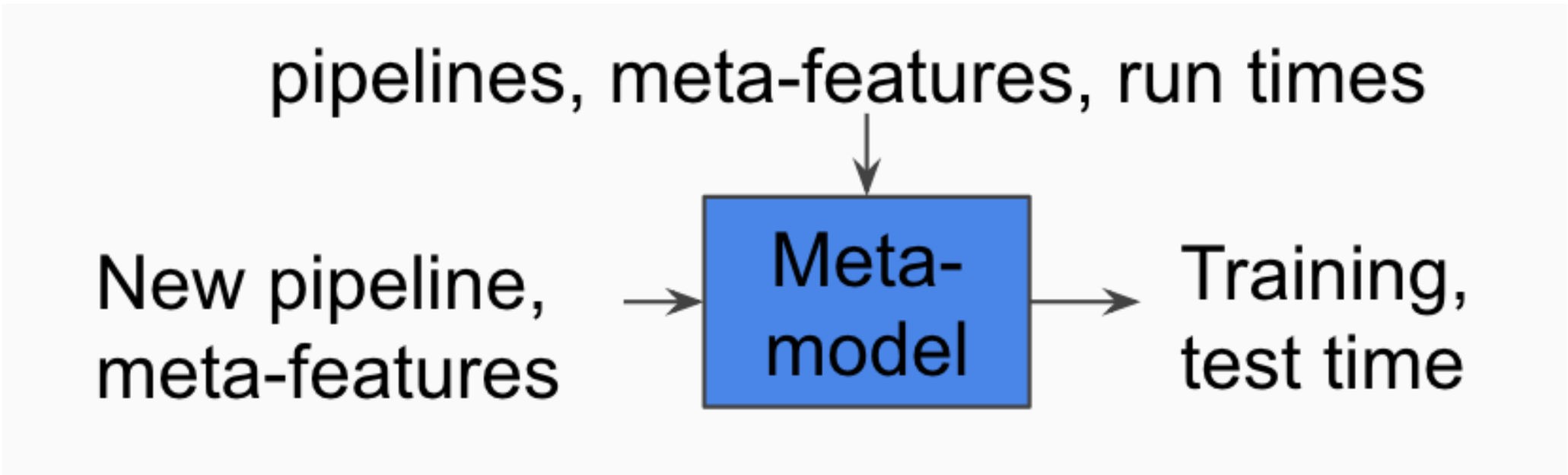
- **MIT AutoML system (ICBD 2017)**
  - Uses and compares against OpenML results



AUTOML: META-LEARNING

- **Reuse (millions of) prior model evaluations:**
  - Benchmark new algorithms against state-of-the-art
  - Meta-models: E.g. predict performance or training time

- **Runtime prediction**



AUTOML: META-LEARNING

- **Reuse (millions of) prior model evaluations:**
  - Benchmark new algorithms against state-of-the-art
  - Meta-models: E.g. predict performance or training time

- **Faster TPOT (in progress)**
  - Build meta-models (Random Forest works well)
  - Focus on fast configurations first

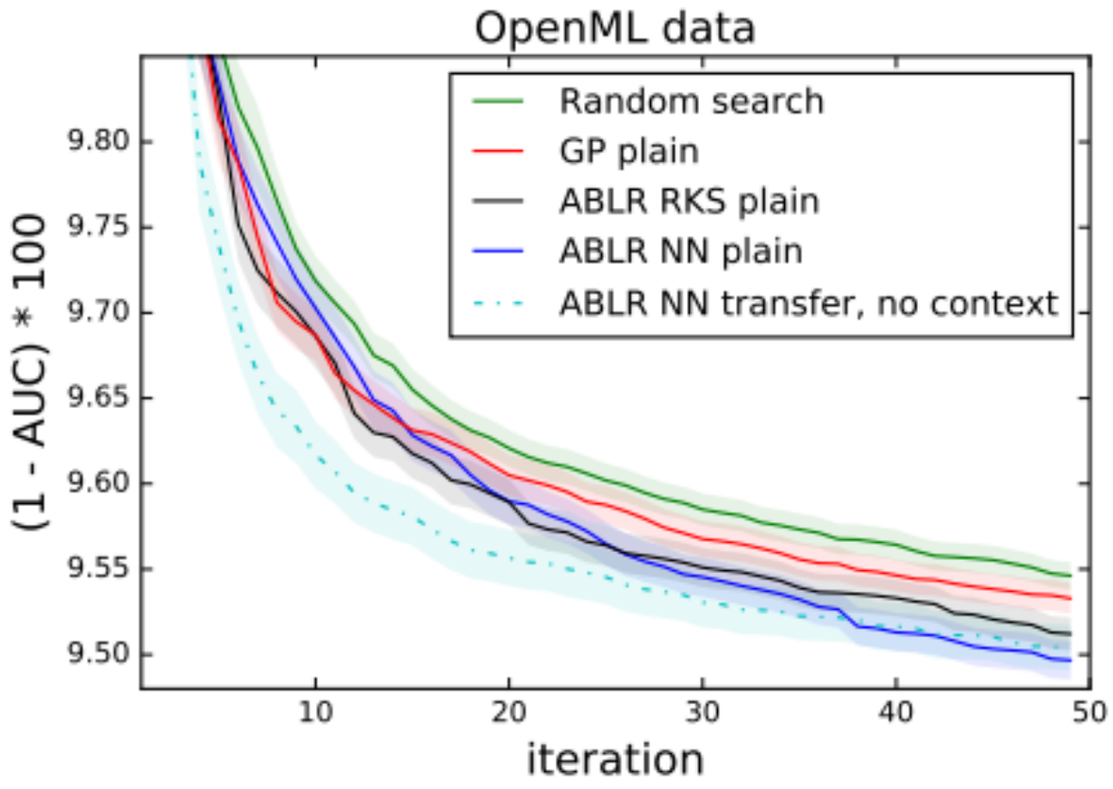
Regressor	RF	SVC	Tree	NB	Boosting	LR	kNN
Median	0.39	0.88	0.74	0.83	0.86	0.57	0.72
Mean	0.4	0.88	0.75	0.86	0.86	0.57	0.73
Extrapolate	0.44	0.80	0.45	1.33	0.33	0.84	1.09
RR	0.12	0.15	0.43	0.54	0.1	0.21	0.3
RF	**0.07**	**0.12**	**0.28**	**0.47**	**0.07**	**0.11**	**0.16**
SVR	0.39	0.87	0.72	0.86	0.84	0.56	0.71

Table 3: Mean Absolute Deviation of $\log_{10}$ runtime predictions. Best results are bold.

AUTOML: META-LEARNING

- **Reuse results on many hyperparameter settings**
  - Surrogate models: predict best hyperparameter settings
  - Study hyperparameter effects/importance

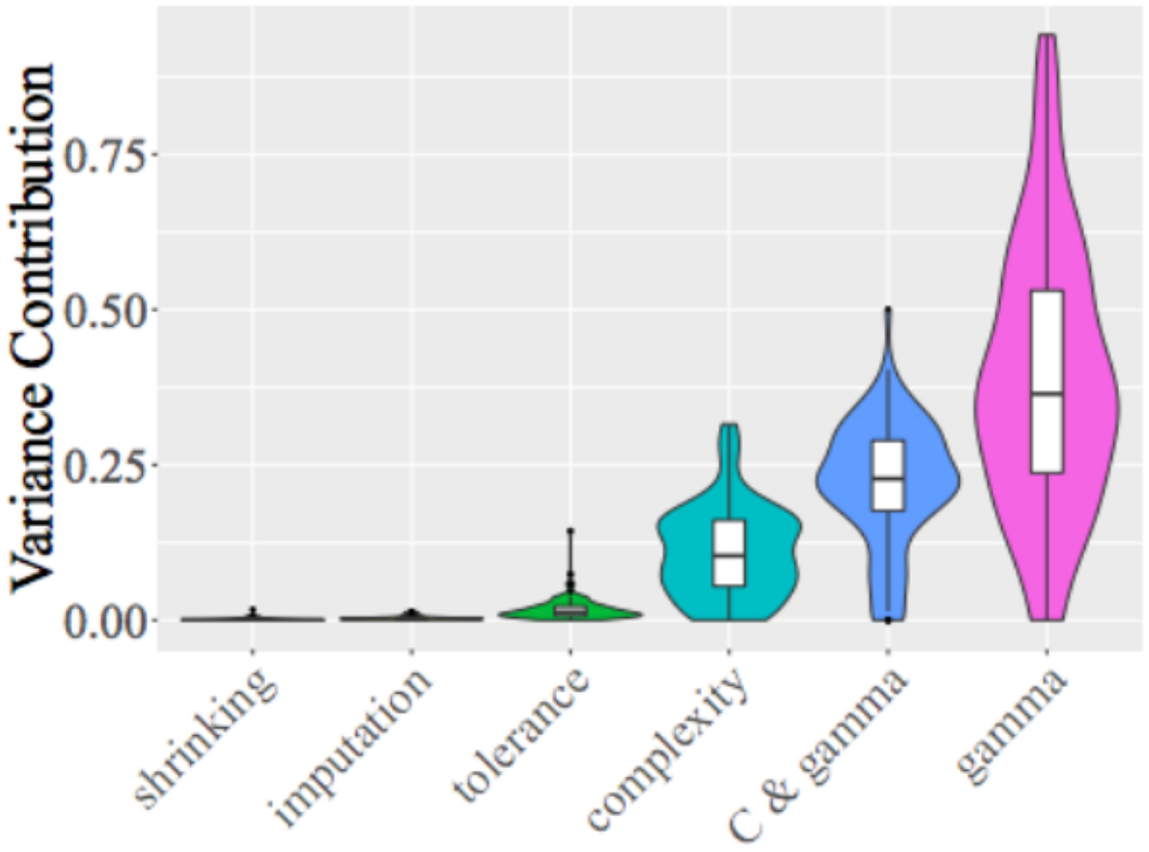
- **Amazon's multi-task learning AutoML (Meta-learning@NIPS 2017)**
  - Trains surrogate models per task
  - On new tasks: learns how to combine them with neural net



AUTOML: META-LEARNING

- **Reuse results on many hyperparameter settings**
  - Surrogate models: predict best hyperparameter settings
  - Study hyperparameter effects/importance

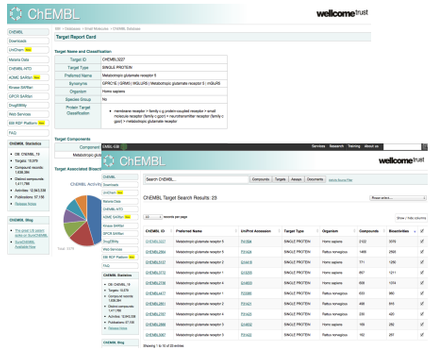
- **Hyperparameter space design**
  - Use OpenML data to learn which hyperparameters to tune



Jan van Rijn et al. (2017) AutoML@ICML

AUTOML: META-LEARNING

- **Never-ending Automatic Machine Learning**:
  - AutoML methods built on top of OpenML get increasingly better as more meta-data is added
- **Faster drug discovery (QSAR)**
  - Meta-learning to build better models that recommend drug candidates for rare diseases



ChEMBL DB: 1.4M compounds,
10k proteins,12.8M activities

Molecule representations →

MW	LogP	TPSA	b1	b2	b3	b4	b5	b6	b7	b8	b9	
377.435	3.883	77.85	1	1	0	0	0	0	0	0	0	
341.361	3.411	74.73	1	1	0	1	0	0	0	0	0	...
197.188	-2.089	103.78	1	1	0	1	0	0	0	1	0	
346.813	4.705	50.70	1	0	0	1	0	0	0	0	0	
							⋮					

16,000 regression datasets
x52 pipelines (on OpenML)

all data on new protein →  → **optimal models to predict activity**

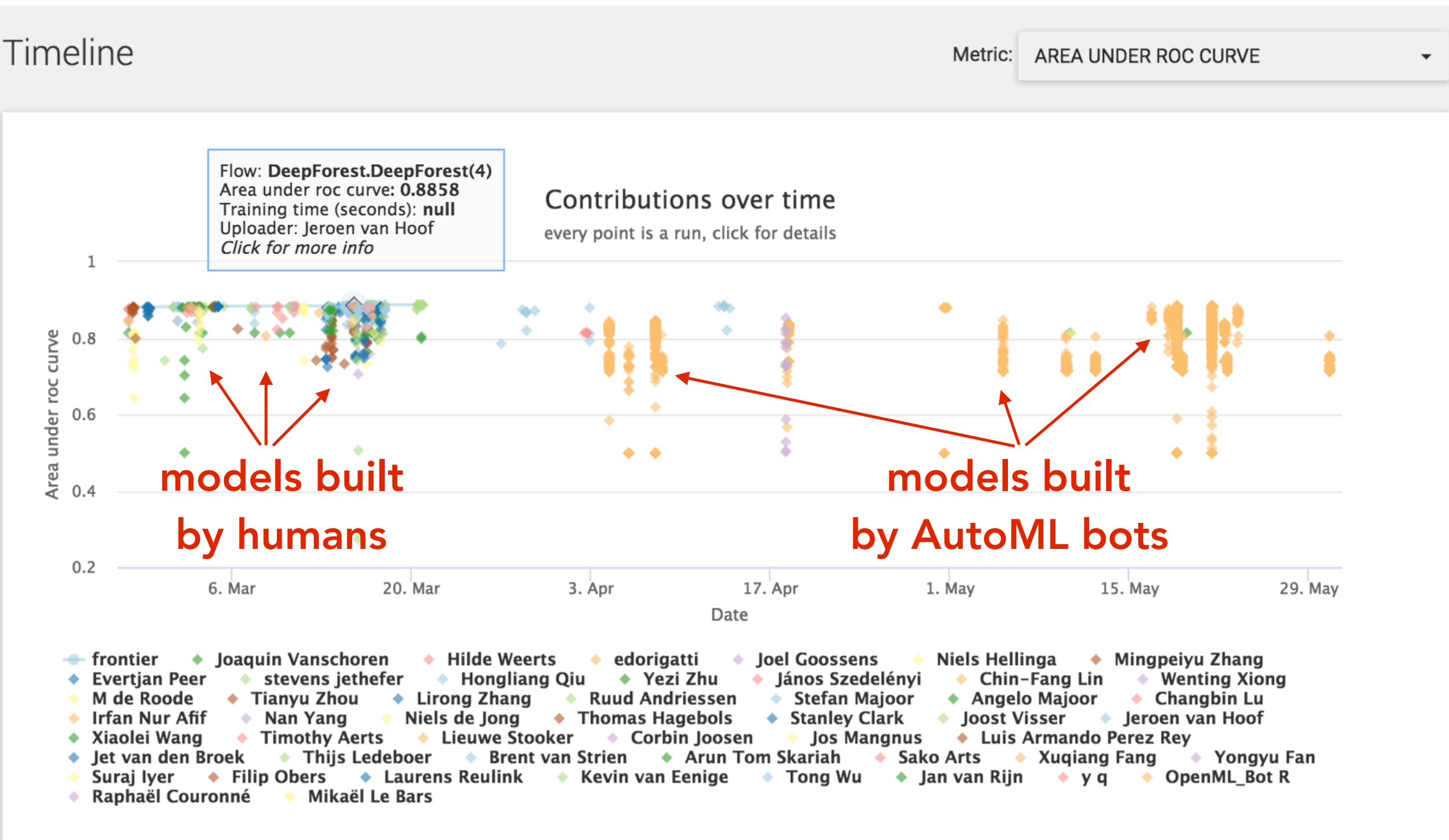
meta-model

(Olier et al., Machine Learning 107(1), 2018)

Learning to learn

AutoML bots learn from models shared by humans (on OpenML)

Humans learn from models built by humans *and* bots (*AI assist*)



Thanks to the entire OpenML team

Jan
van Rijn

Guiseppe
Casalicchio

Mattias
Feurer

Bernd
Bischl

Andreas
Mueller

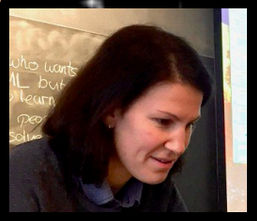
Heidi
Seibold

Bilge
Celik

Pieter
Gijsbers

Andrey
Ustyuzhanin

William
Raynaut



Erin
LeDell

Tobias
Glasmachers

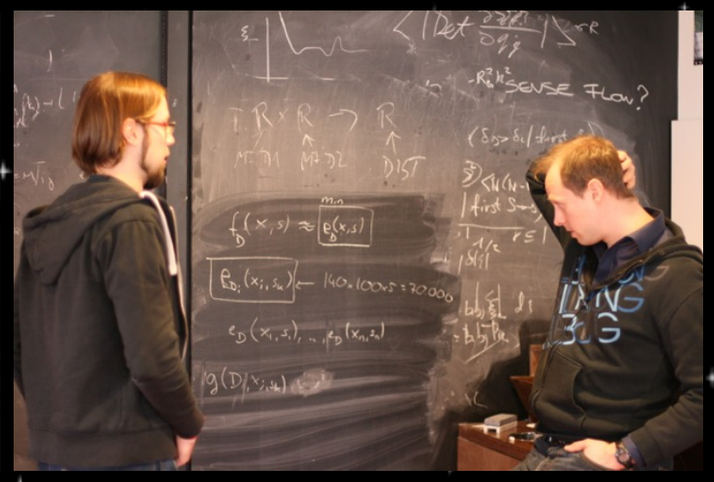
Jakub
Smid

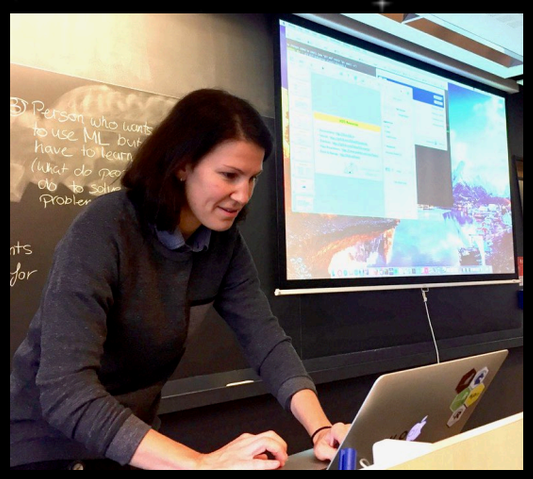
And many more!

Join us! (and change the world)

Active open source community
We need more bright people

- ML/DB experts
- Developers
- UX

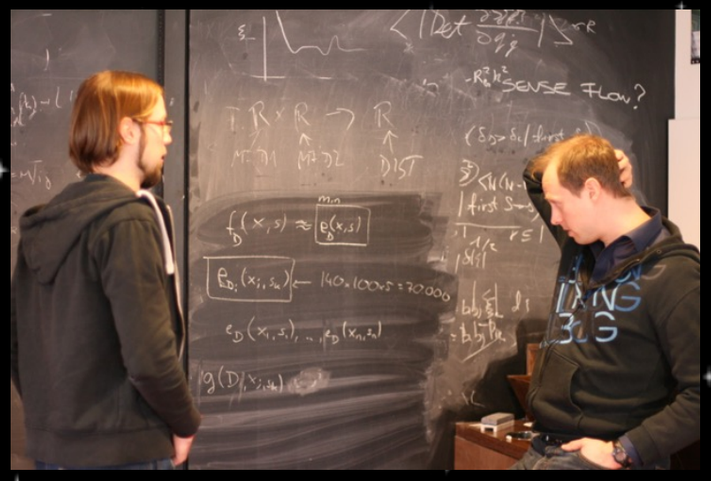




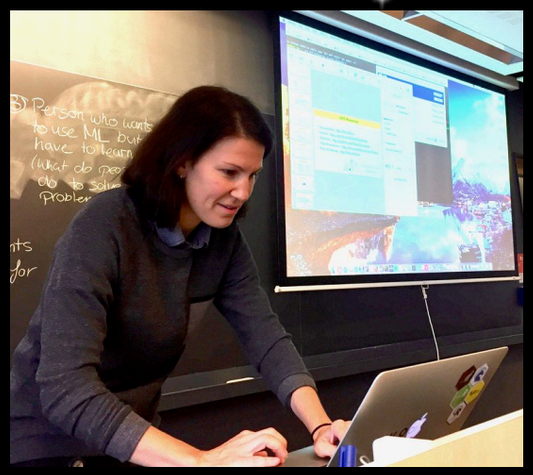
Support is welcome!

Workshop sponsorship (hackathons 2x/year)
Donations: OpenML foundation
Compute time
Project ideas







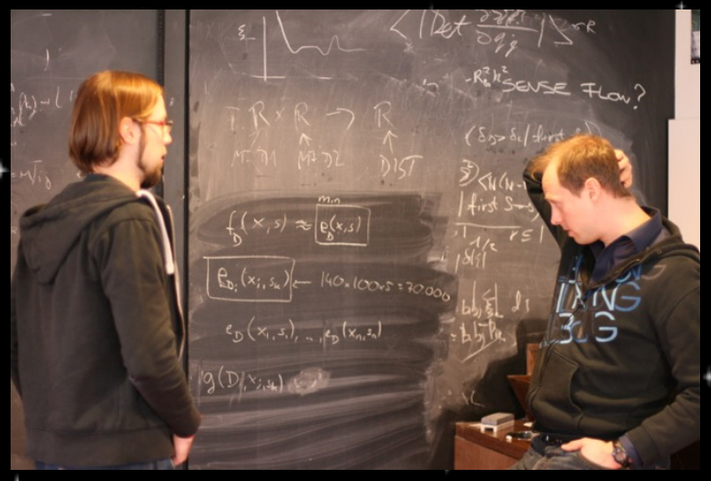


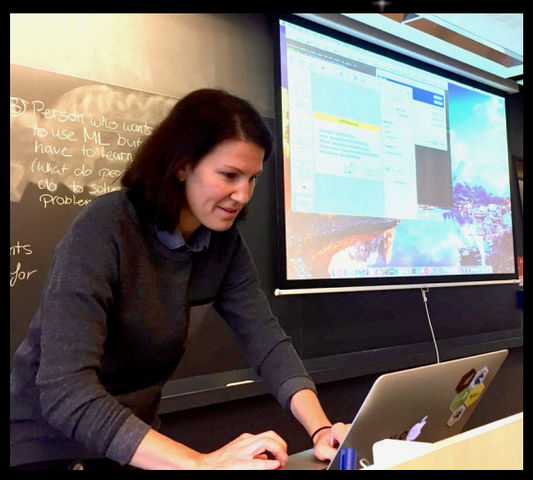

@open_ml

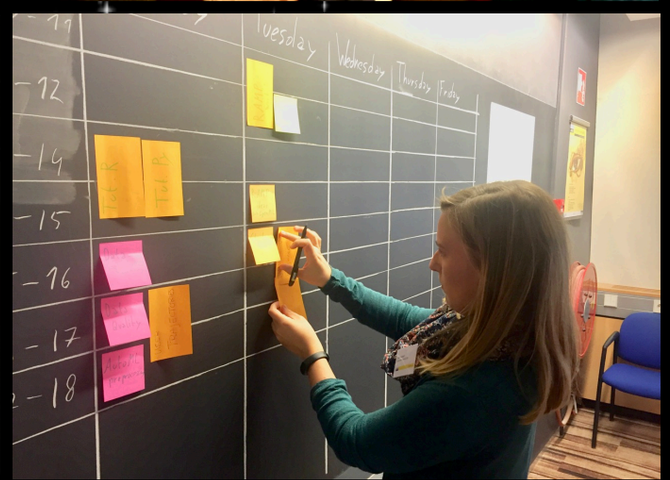
OpenML

Thank you!

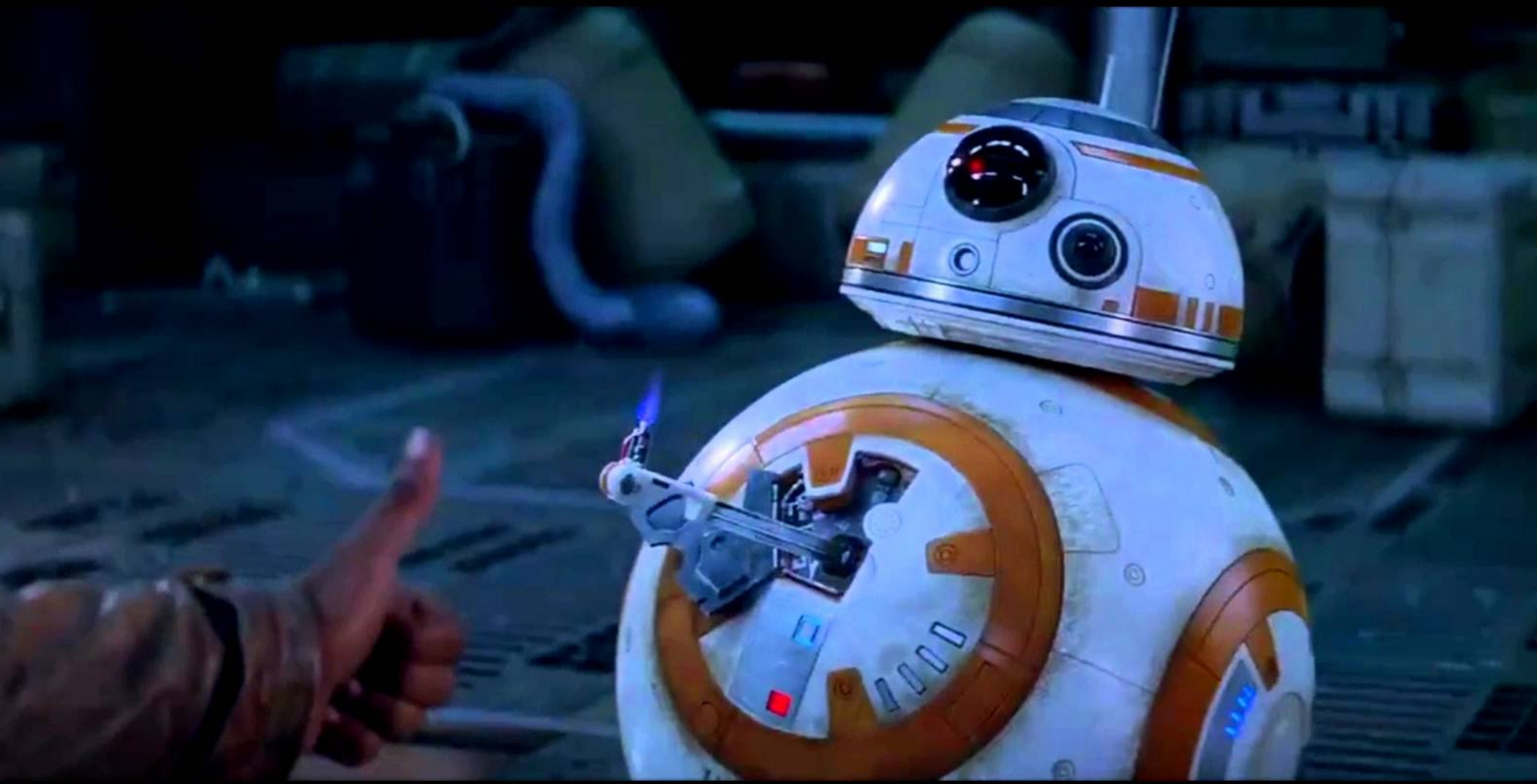
谢谢







Questions?



EINDHOVEN UNIVERSITY

Looking for:

- PhD Students
- Scientific programmer