Auto-Sklearn: Automated Machine Learning in Python

Matthias Feurer /__mfeurer__
Katharina Eggensperger /KEggensperger

Department of Computer Science
University of Freiburg, Germany

AutoML.org

Find slides, notebooks and more here:
https://github.com/automl/auto-sklearn-talks

Our Vision

Machine Learning for everyone

in 4 lines of code

```
import autosklearn.classification
>>> cls = autosklearn.classification.AutoSklearnClassifier()
>>> cls.fit(X_train, y_train)
>>> predictions = cls.predict(X_test)
```

Goals for today & Outline

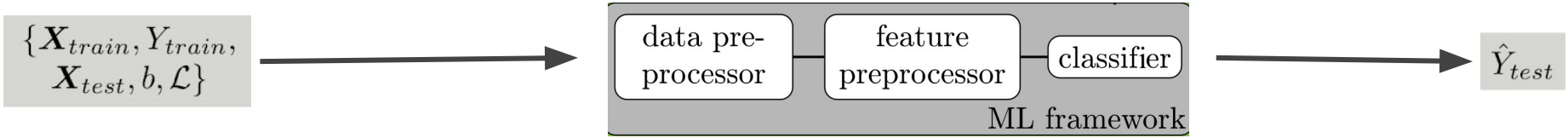
Goals

1. Understand how Auto-sklearn works
2. Apply Auto-Sklearn

Outline

1.	Introduction to Auto-sklearn	(20 mins)
2.	Moving to Gathertown & Setup	(10 mins)
3.	**Task 1**: BYOP	(15 mins)
4.	**Task 2**: ASKL	(20 mins)
5.	**Task 3**: EXTEND	(15 mins)
6.	+ Bonus Tasks	(? mins)

Design Space: Traditional ML with scikit-learn



Design Space: Traditional ML with scikit-learn

$\{\boldsymbol{X}_{train}, Y_{train}, \boldsymbol{X}_{test}, b, \mathcal{L}\}$

preprocessor	#λ
extreml. rand. trees prepr.	5
fast ICA	4
feature agglomeration	4
kernel PCA	5
rand. kitchen sinks	2
linear SVM prepr.	3
no preprocessing	-
nystroem sampler	5
PCA	2
polynomial	3
random trees embed.	4
select percentile	2
select rates	3
one-hot encoding	2
imputation	1
balancing	1
rescaling	1

classifier	#λ
AdaBoost (AB)	4
Bernoulli naïve Bayes	2
decision tree (DT)	4
extreml. rand. trees	5
Gaussian naïve Bayes	-
gradient boosting (GB)	6
kNN	3
LDA	4
linear SVM	4
kernel SVM	7
multinomial naïve Bayes	2
passive aggressive	3
QDA	2
random forest (RF)	5
Linear Class. (SGD)	10

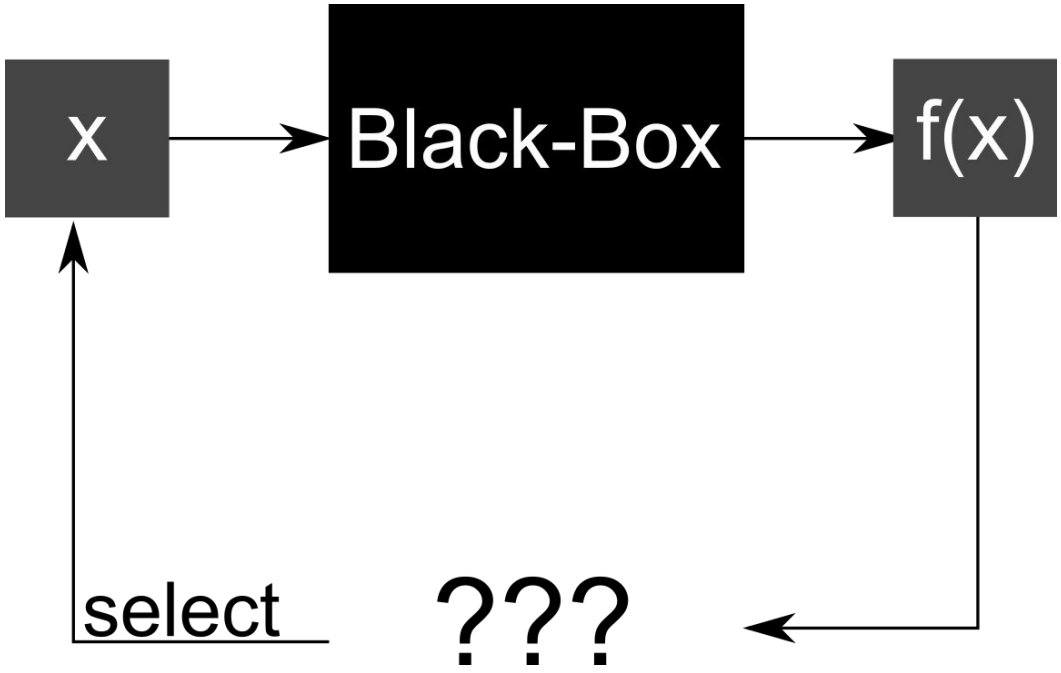
$\hat{Y}_{test}$

Black Box Optimization

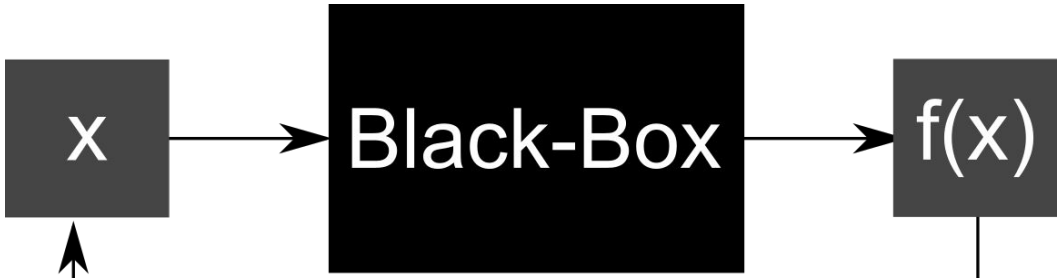
This is a configuration of a ML pipeline

This is your ML pipeline

This measures performance of your pipeline



Black Box Optimization: The Human Optimizer

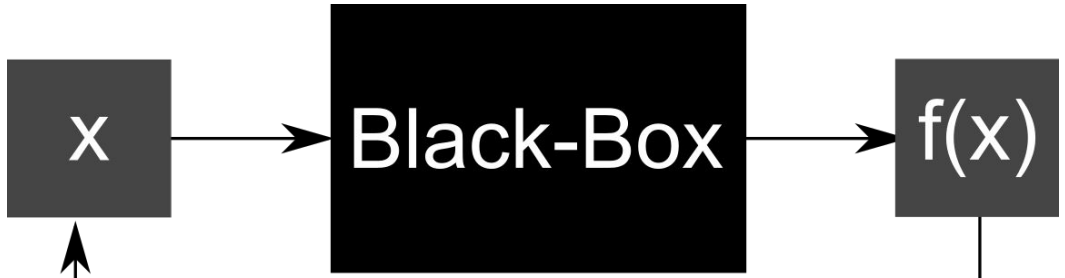


- You'll learn a lot about your system

select

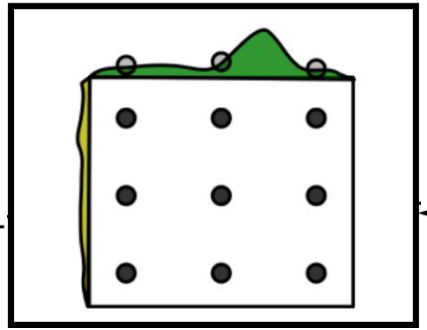
- Not very efficient
- Error-prone
- Requires expert knowledge

Black Box Optimization: Grid Search

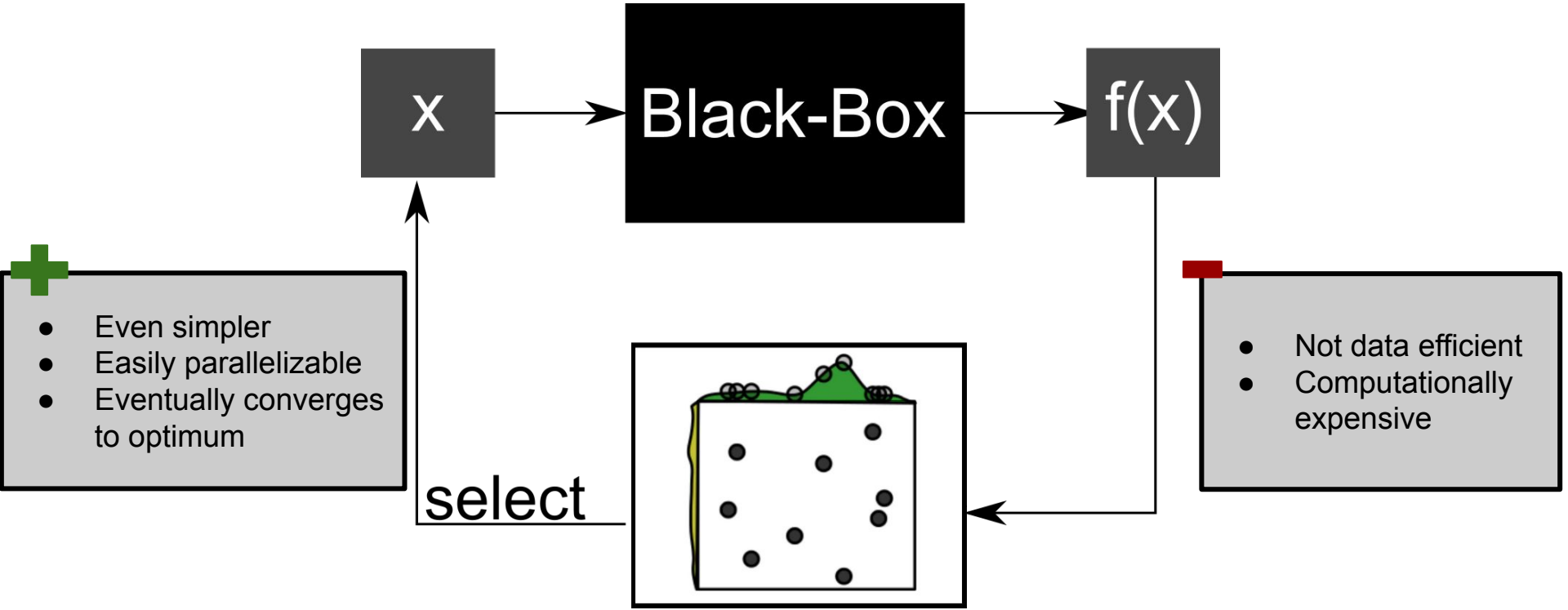


- Very simple approach
- Easily parallelizable
- Can be used to study the problem

- Does not scale to high dimensions
- Grid needs to be defined



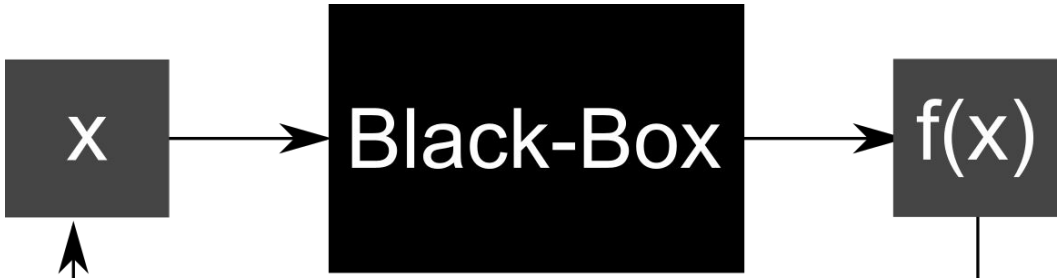
Black Box Optimization: Random Search



- Even simpler
- Easily parallelizable
- Eventually converges to optimum

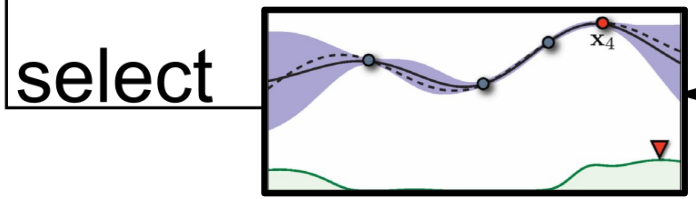
- Not data efficient
- Computationally expensive

Black Box Optimization: Bayesian Optimization

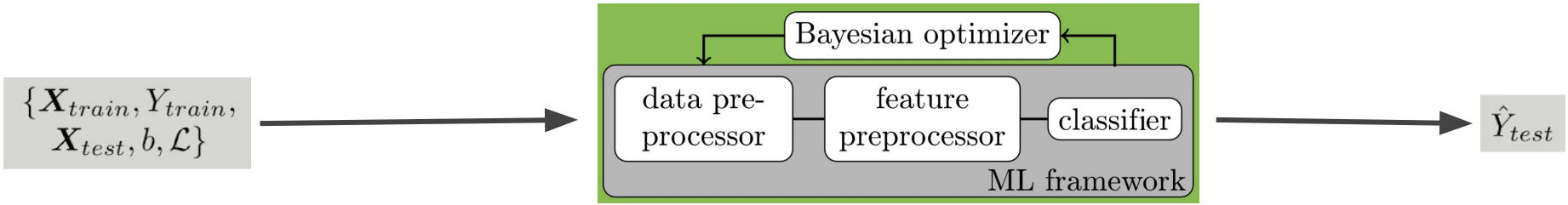


- Trade-off Exploration & Exploitation
- Data efficient
- State-of-the-art

- (Does not scale well with parallel resources)

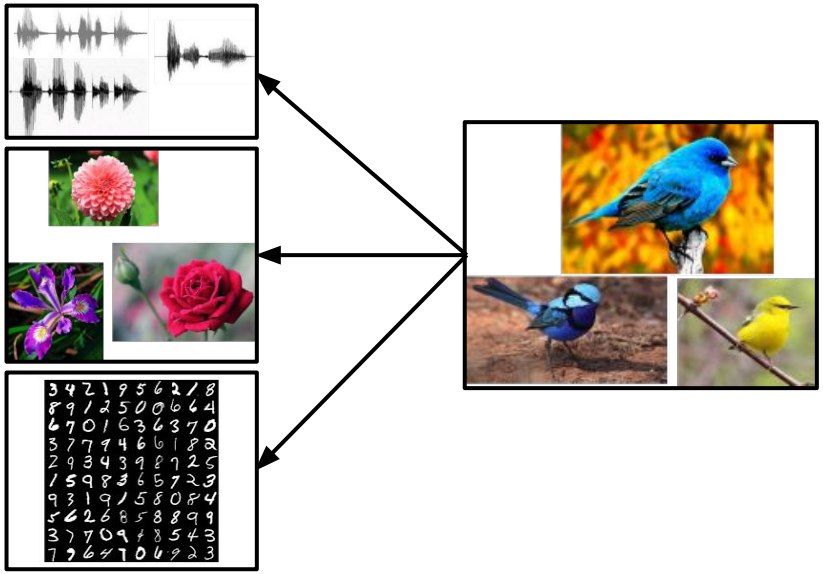


Design Space: Traditional ML with scikit-learn



How to reuse previous experience?

→ Warmstart Bayesian Optimization



Offline / Before:

1) Collect >200 datasets
2) Find the best pipeline on each dataset

Online / For a new dataset:

1) Compute 38 meta-features, select 25 most similar previous datasets
2) Initialize optimization with best pipelines on those datasets

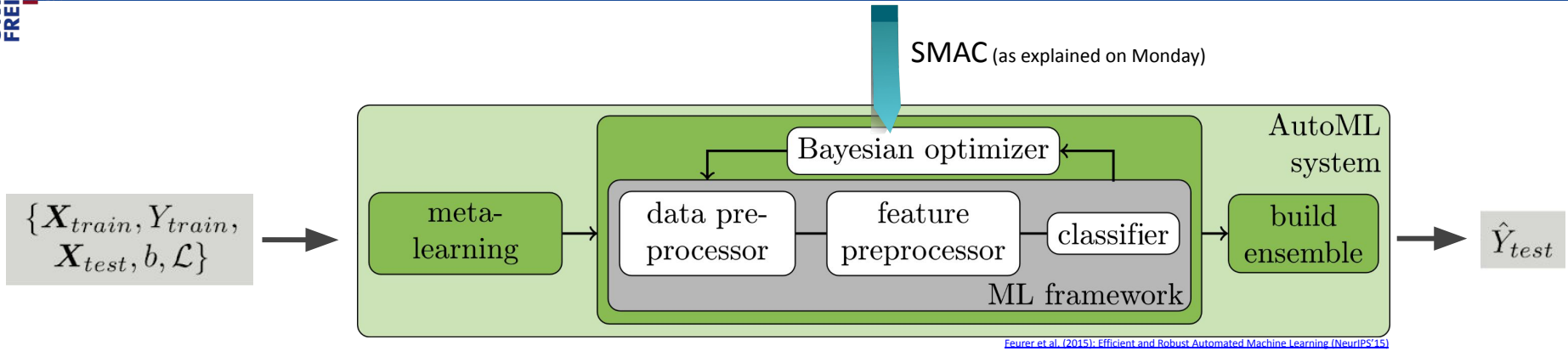
More II: Ensembling

How to get the best out of all evaluated models?

→ **Build an ensemble**

Image credit: Photo by Denisse Leon

Auto-Sklearn 1.0



Feurer et al. (2015): Efficient and Robust Automated Machine Learning (NeurIPS'15)

→ Similar to Auto-Pytorch (as explained on Tuesday)

However, some things to be improved

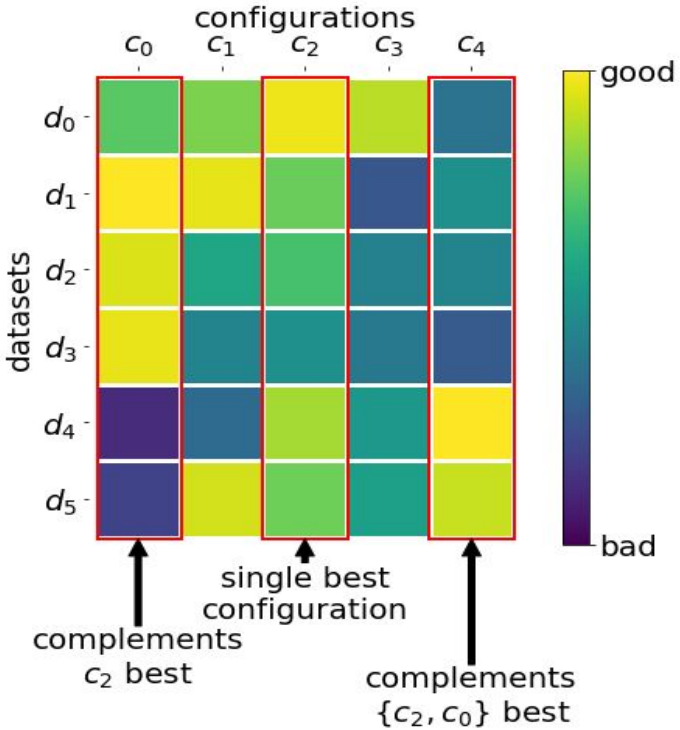
- meta-features can be expensive to compute
- large datasets can be an issue

continues

Even More I: Portfolios

Goal: Meta-Learning without meta-features

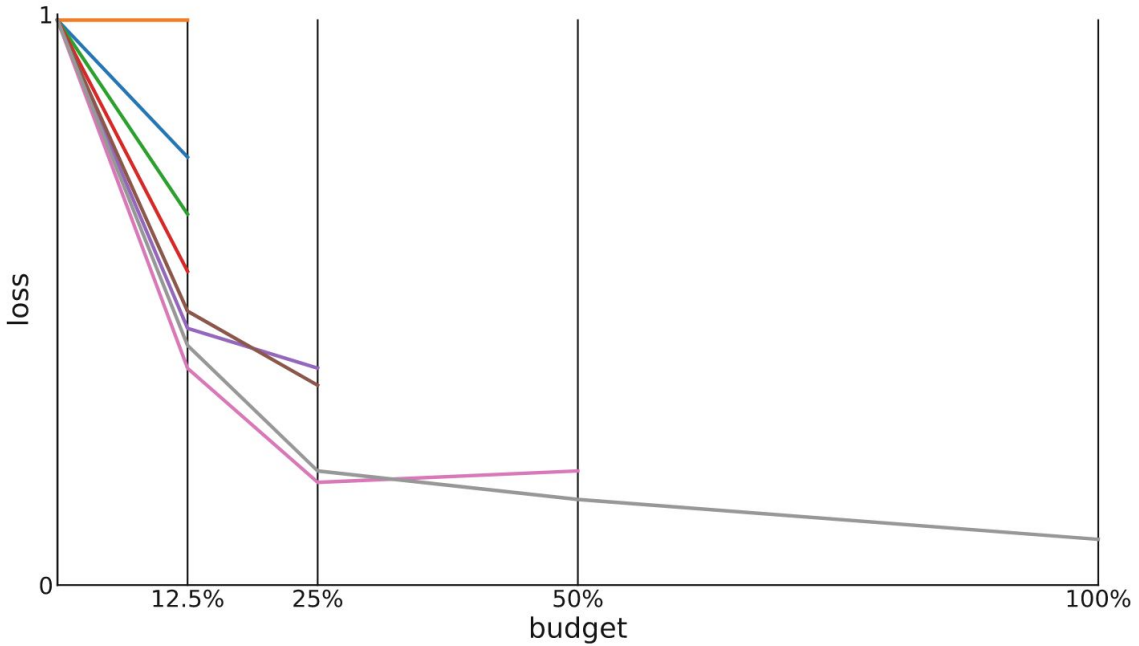
Idea: Construct a Portfolio (a list of diverse pipelines)



Even More II: Successive Halving

Goal: Scale to large datasets.

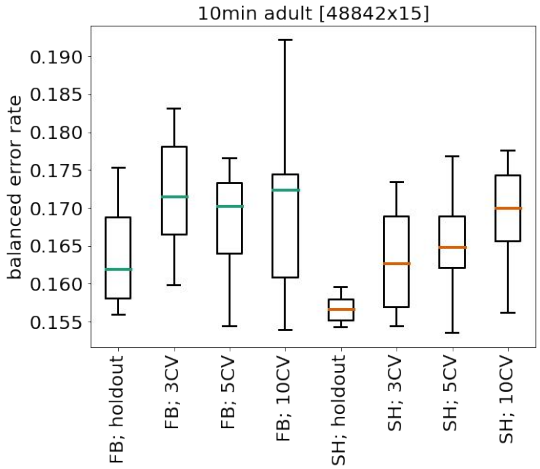
Idea: Allocate more resources to promising pipelines

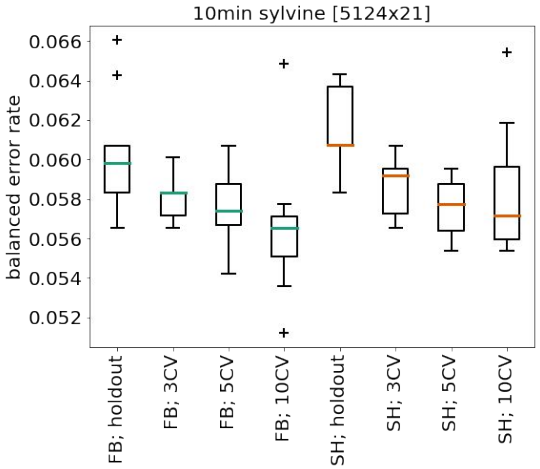


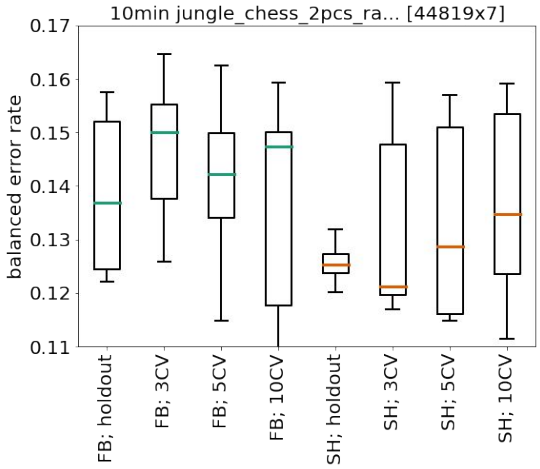
But what about small datasets?

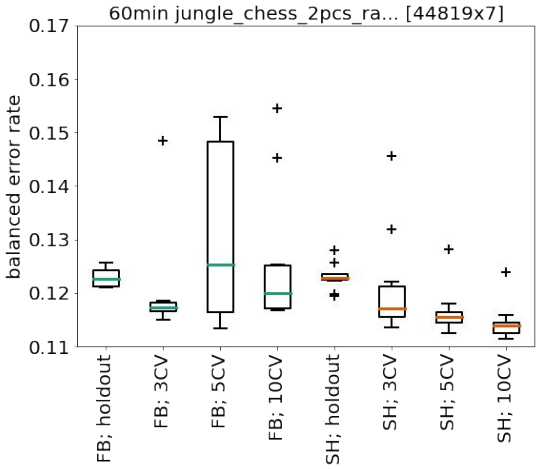
Matthias Feurer and Frank Hutter: *Hyperparameter optimization*
Automated Machine Learning, The Springer Series on Challenges in Machine Learning

Impact of the Optimization Strategy









But wait … did we make it worse?

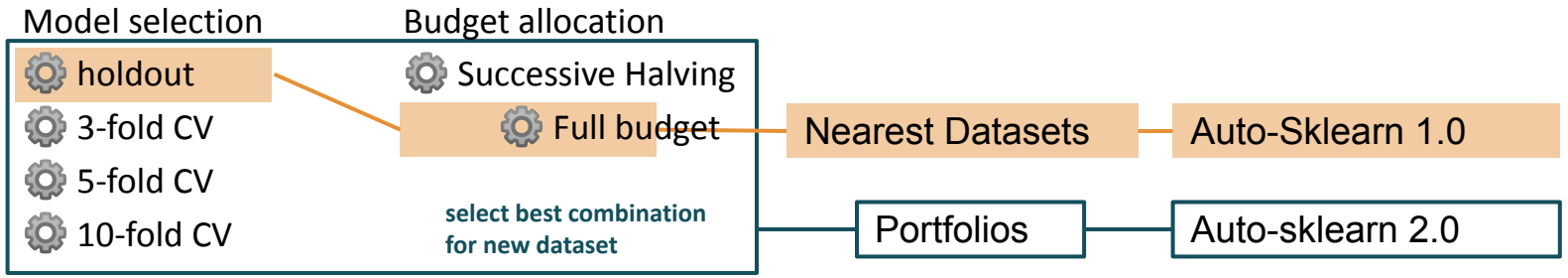
Can we automatically select an optimization policy?

Image Credit - CC BY-NC-ND 2.0; by Beagle Mama

Yes, with a learned selector!

For more details see “Feurer et al. (2021): Auto-Sklearn 2.0: Hands-free AutoML via Meta-Learning“

PoSH-Auto-sklearn



	10MIN		60MIN	
	∅	std	∅	std

Feurer et al. (2021): Auto-Sklearn 2.0: Hands-free AutoML via Meta-Learning

Practical features

- Simplicity → follows scikit-learn API
- Parallelism → Uses Dask
- Extensibility → Simply add new algorithms
- Robustness → Limits on time and memory usage

- and many more:
  - configurable (access to underlying SMAC)
  - compatible with Pandas/numpy arrays
  - …

Auto-sklearn vs SMAC vs NASLib

	SMAC	NASLib	Auto-sklearn
Goal	• Algorithm configuration • Hyperparameter tuning • Blackbox optimization	Neural Architecture Search research	• *ML in 4 lines of code* • Automated Machine Learning • Drop-in replacement for scikit-learn
Comments	Formed the basis for the winning entry to the NeurIPS 2020 Black-Box optimization competition!	Uses Blackbox and Greybox optimization under the hood	Uses SMAC under the hood
Alternatives	Hyperopt, Optuna, Ray, AX, DEHB, mlrMBO, etc...	None (so far)	Auto-PyTorch, AutoXGBoost, H2O, Auto-Gluon, etc...

Summary

```
import autosklearn.classification
>>> cls = autosklearn.classification.AutoSklearnClassifier()
>>> cls.fit(X_train, y_train)
>>> predictions = cls.predict(X_test)
```

- based on **scikit-learn; simple & familiar API**
- integrates **latest research** (**>1K citations**)
- **>20K** downloads per month

- **BSD-3-Clause** License
- works best under **Linux**
- requires **Python>=3.7**

/automl/auto-sklearn

Matthias Feurer
PhD Student @ ML Lab Freiburg

Katharina Eggensperger
PhD Student @ ML Lab Freiburg

Edward Bergman
Research Engineer @ ML Lab Freiburg

Prof. Dr. Marius Lindauer
Head of the ML Lab Hannover

Prof. Dr. Frank Hutter
Head of the ML Lab Freiburg

Now: **Hands-On Session**

- Go to the hands-on session room in Gathertown
- Open the Colab notebook (see here)
- Work through
  - 1: BYOP (10min; warmup)
  - 2: ASKL (20min)
  - 3: EXTEND (20min)
  - + Bonus tasks