# Optimised Deep Learning - Jean Zay

## Hyperparameter Optimization

cnrs
Institut du Développement et des Ressources en Informatique Scientifique

# HPO = Hyperparameter Optimisation

Hyperparameters ◂

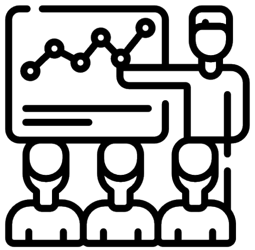
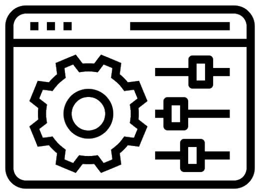
HPO ◂

Related Problems ◂

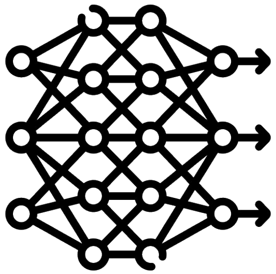
# Hyperparameters

In machine learning, a hyperparameter is **a parameter whose value is used to control the learning process**. By contrast, the values of other parameters (typically node weights) are derived via training.

## Hyperparameters

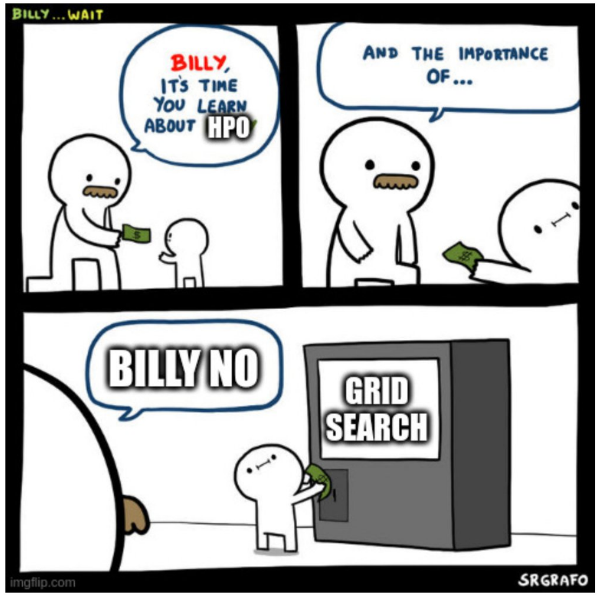




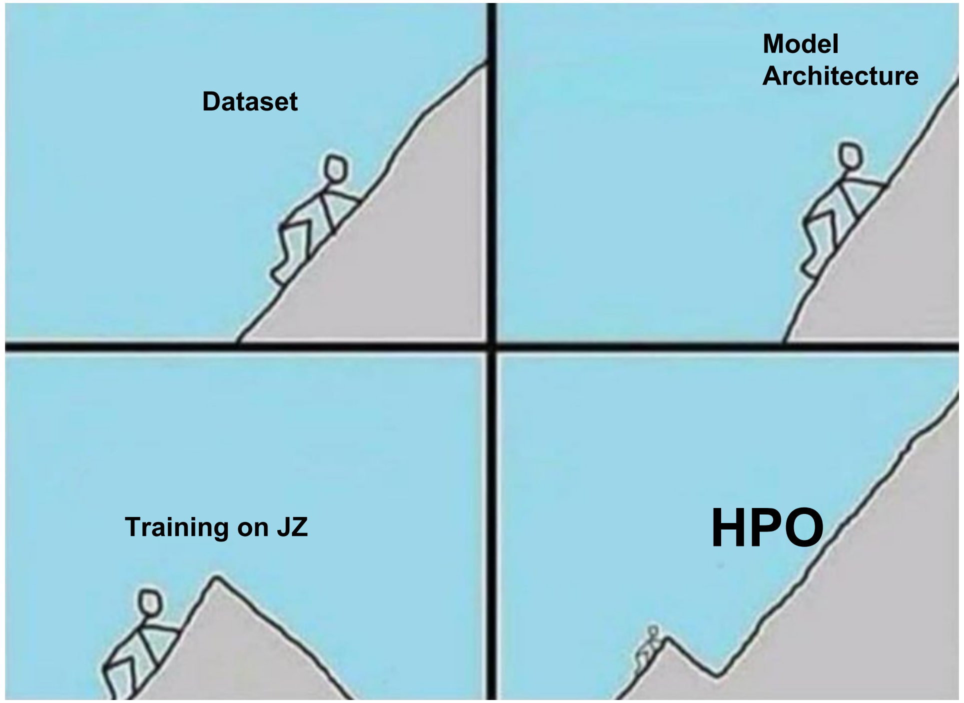
## Parameters



# HPO : Hyperparameter Optimisation

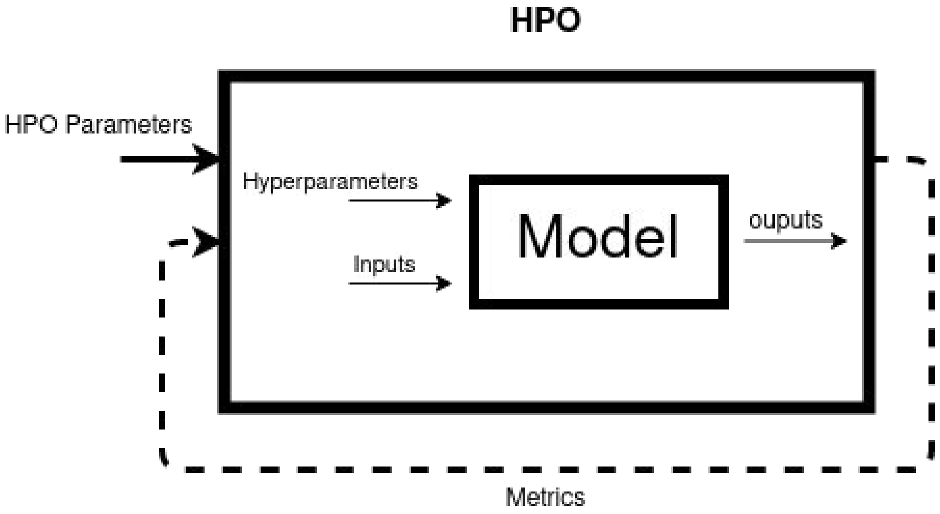
Machine learning algorithms are highly configurable by their hyperparameters.
**These parameters often substantially influence** the complexity, behavior, speed as well as other aspects of the learner, and their values must be selected with care in order to achieve optimal performance.
**Human trial-and-error to select these values is time-consuming, often somewhat biased, error-prone and computationally irreproducible.**

# HPO

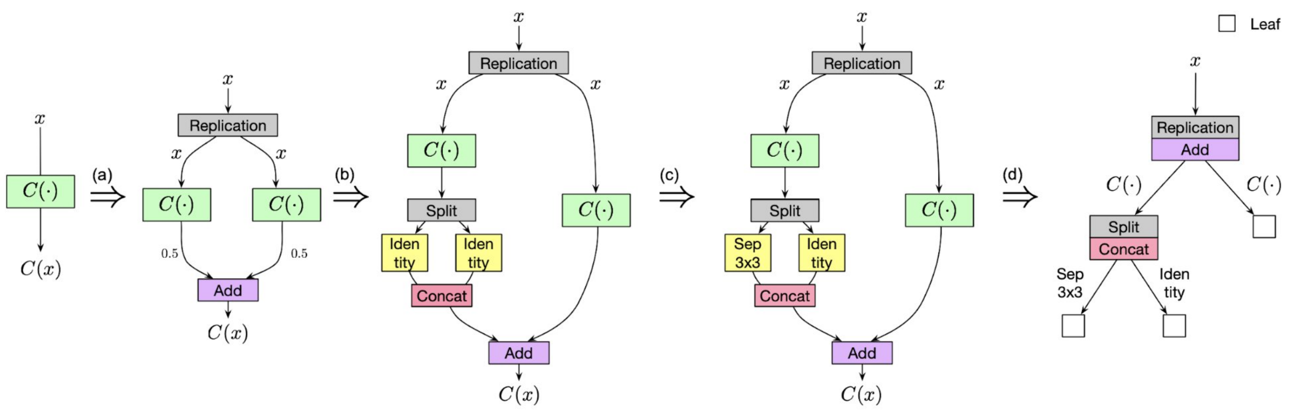




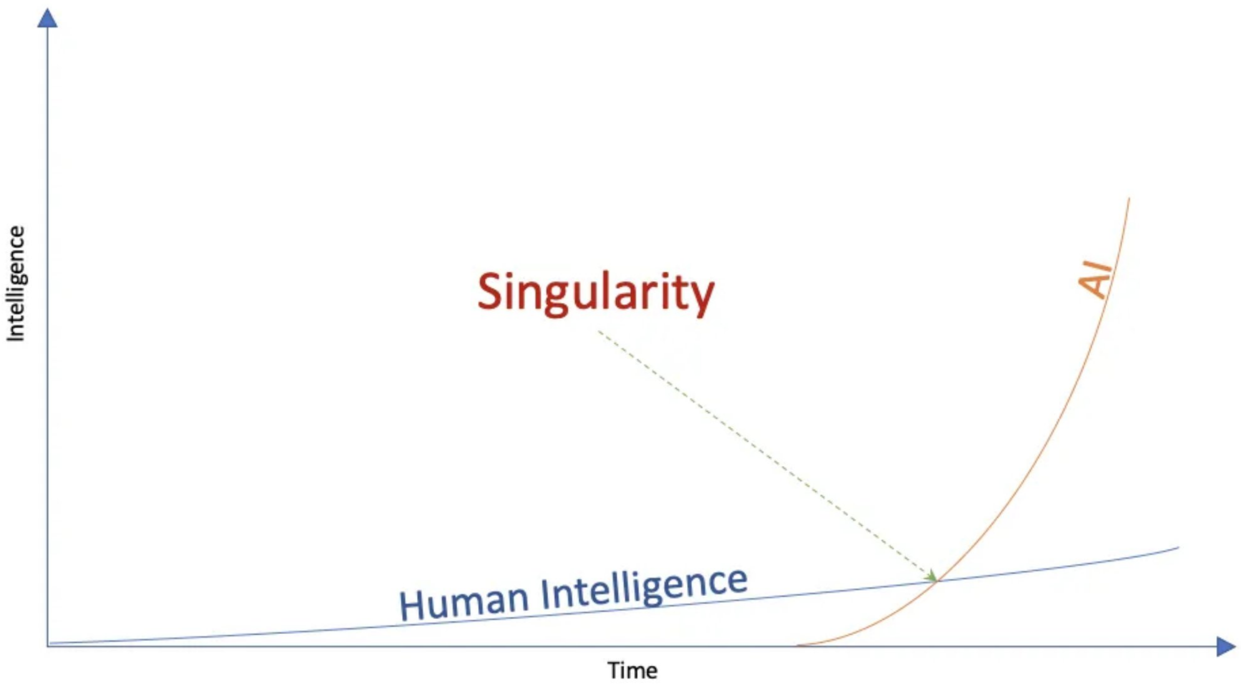

Hyperparameter Optimization == Bi-Level optimization problem

# Related Problems

- Neural Architecture Search **(NAS)**
- Algorithm Selection and traditional Meta-Learning
- Algorithm configuration **(AC)**
- Dynamic Algorithm Configuration **(DAC)**
- Learning to learn and to optimize




**A Comprehensive Survey of Neural Architecture Search: Challenges and Solution** (https://arxiv.org/pdf/2006.02903.pdf)





**Fastest wheel change on a moving car** - Guinness World Records

# Search Algorithms / Samplers

**Basic ◂**

Manual, Grid Search, Random Search

**Bayesian Optimisation ◂**

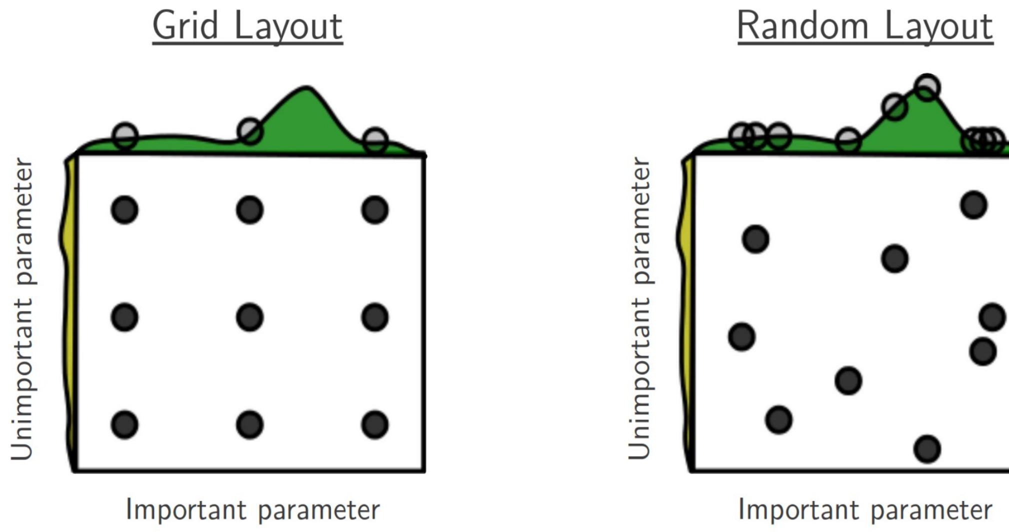
Tree-structured Parzen Estimator, Gaussian Process

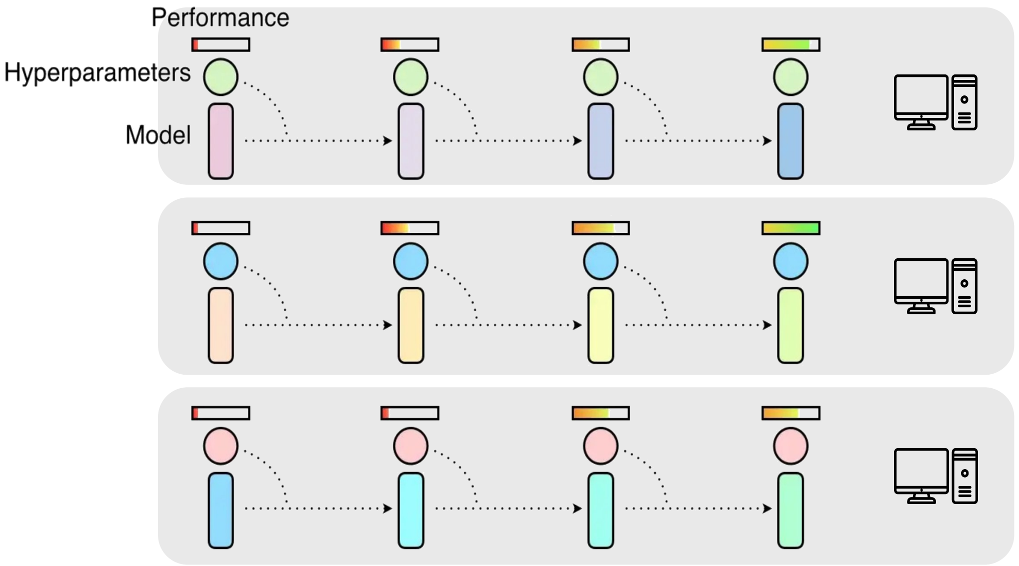
**Heuristic ◂**

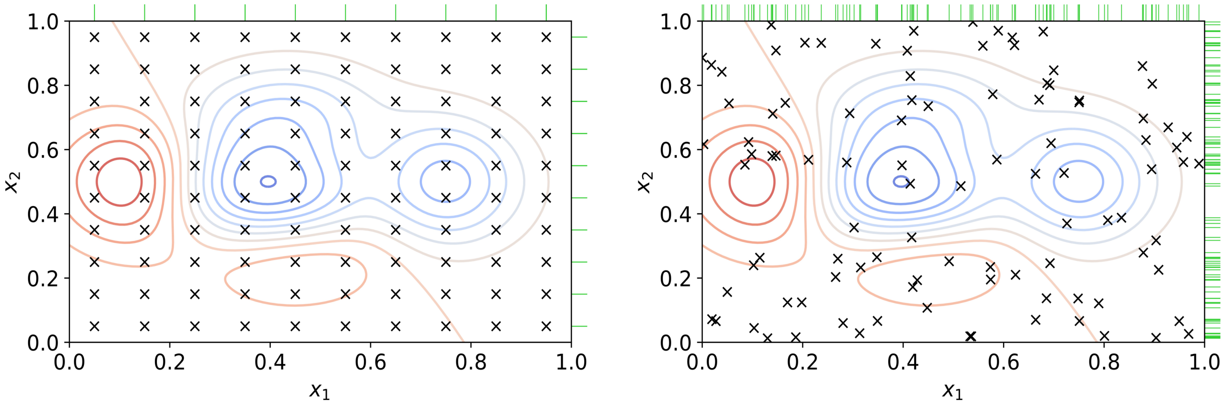
Genetic  Algorithm, Particle Swarm Optimization

**Gradient-based Optimization ◂**

# Basic : Grid & Random Search

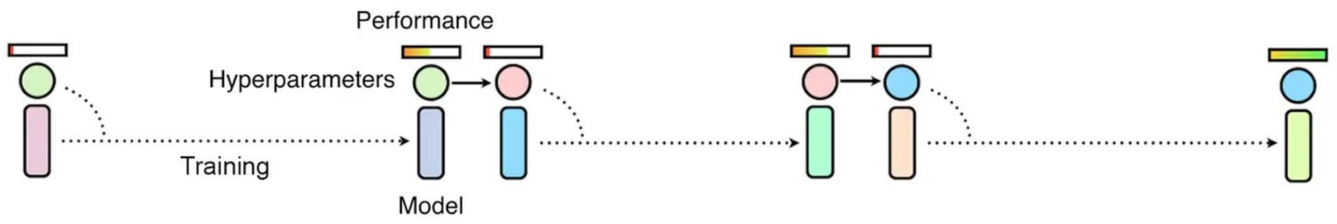


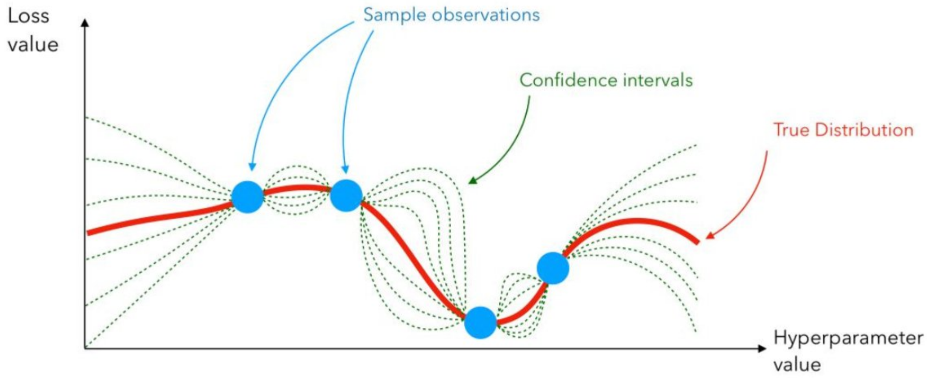



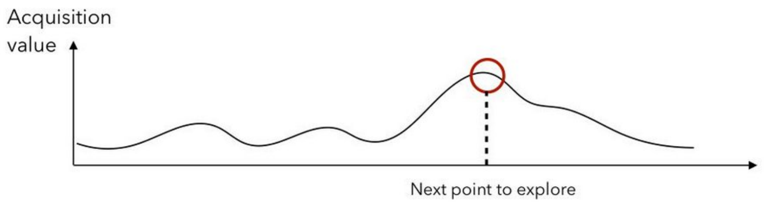
- Independent tests (which can be parallelized) which test a combination of hyperparameters.
- Very costly in resources and no guarantee of improved results.
- Random search is better for high dimensional space

# Bayesian Optimization : TPE & GP



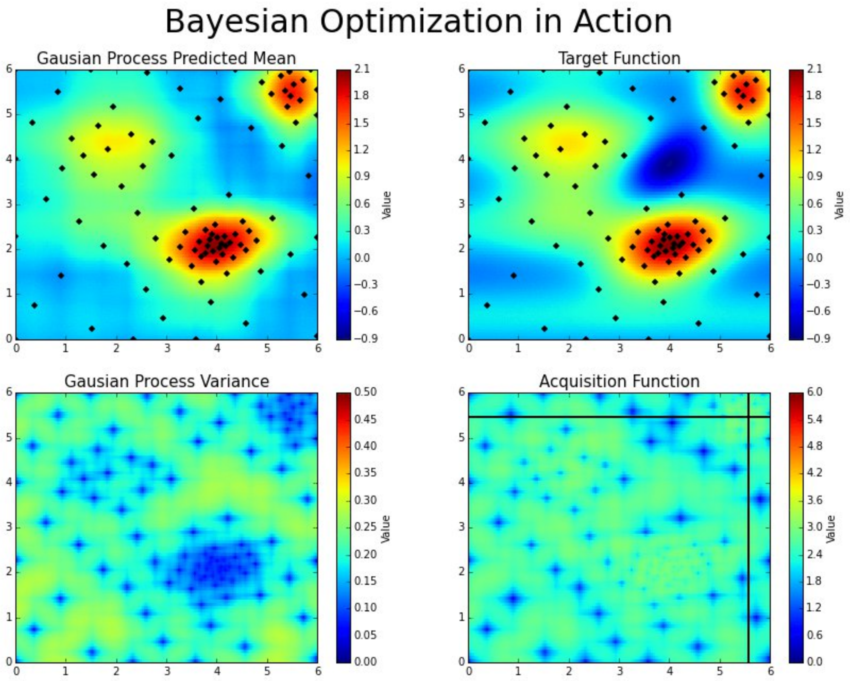




Expected metric score according to Hyper-parameters

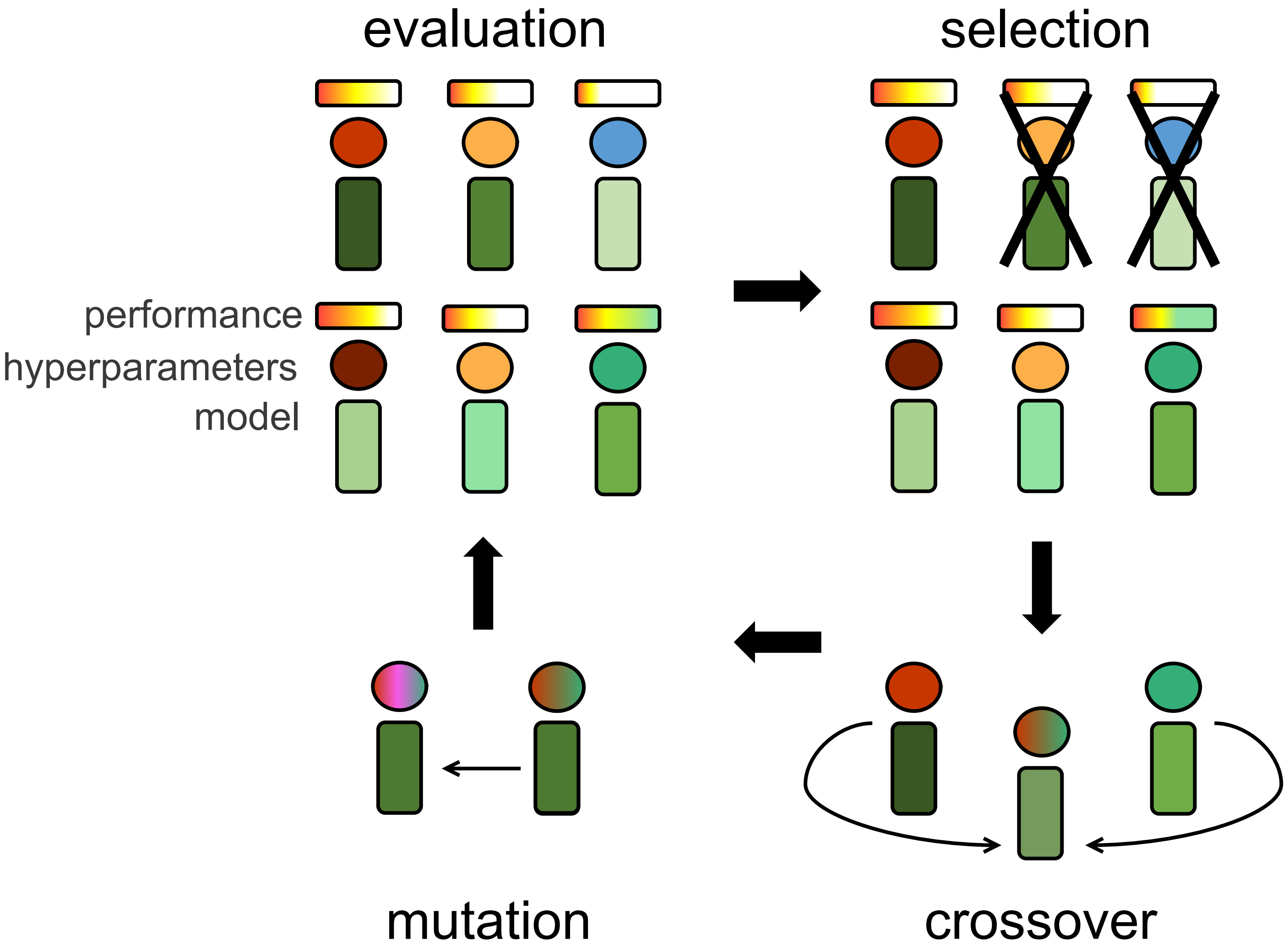



Maximize Acquisition function e.g. Expected Improvement

- Tree Parzen Estimator / Gaussian Process
- Sequential but allows to quickly find the global optimum.
- Proposes a new set of hyper parameters based on the scores obtained by the previous ones tested.



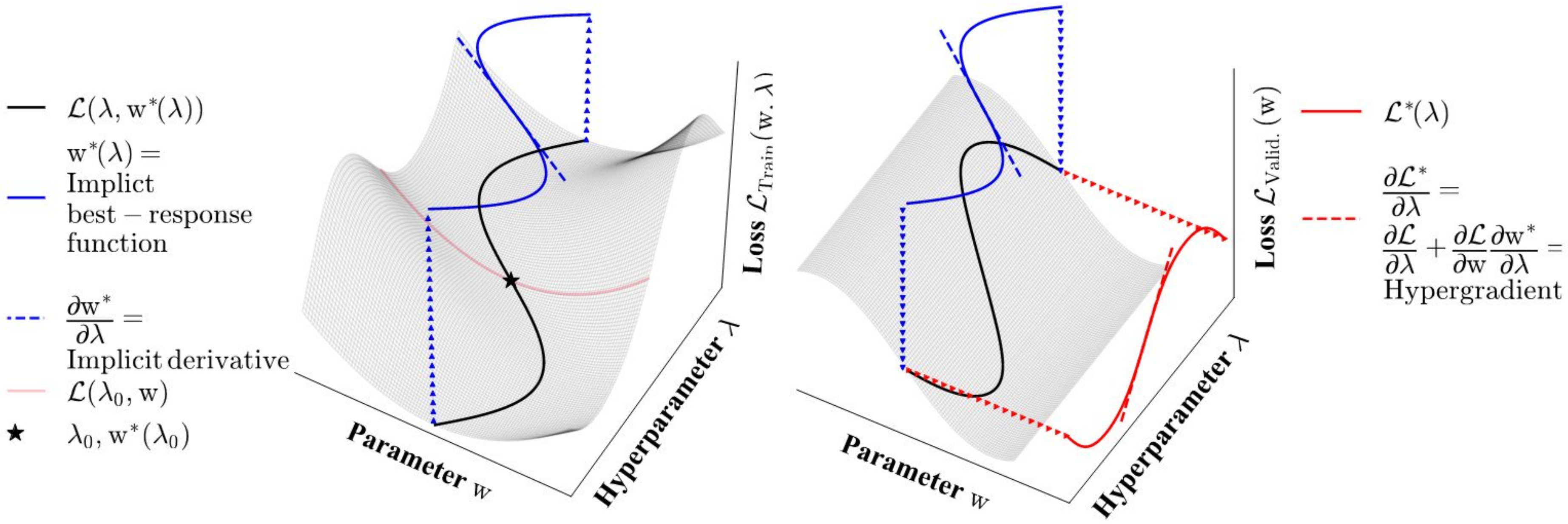
# Heuristic : Evolutionary Optimization




- Bio-inspired
- Can have fatal mutation

- **Genetic Algorithm (GA)**
- **Genetic Programming (GP)**
- **Evolution Strategy (ES)**
- **Particle Swarm Optimization (PSO)**
- **Estimation of Distribution Algorithms (EDA)**

# Gradient-based optimization




**Optimizing Millions of Hyperparameters by Implicit Differentiation**
(https://arxiv.org/pdf/1911.02590.pdf)

- High dimensionality
- Bi-level optimisation

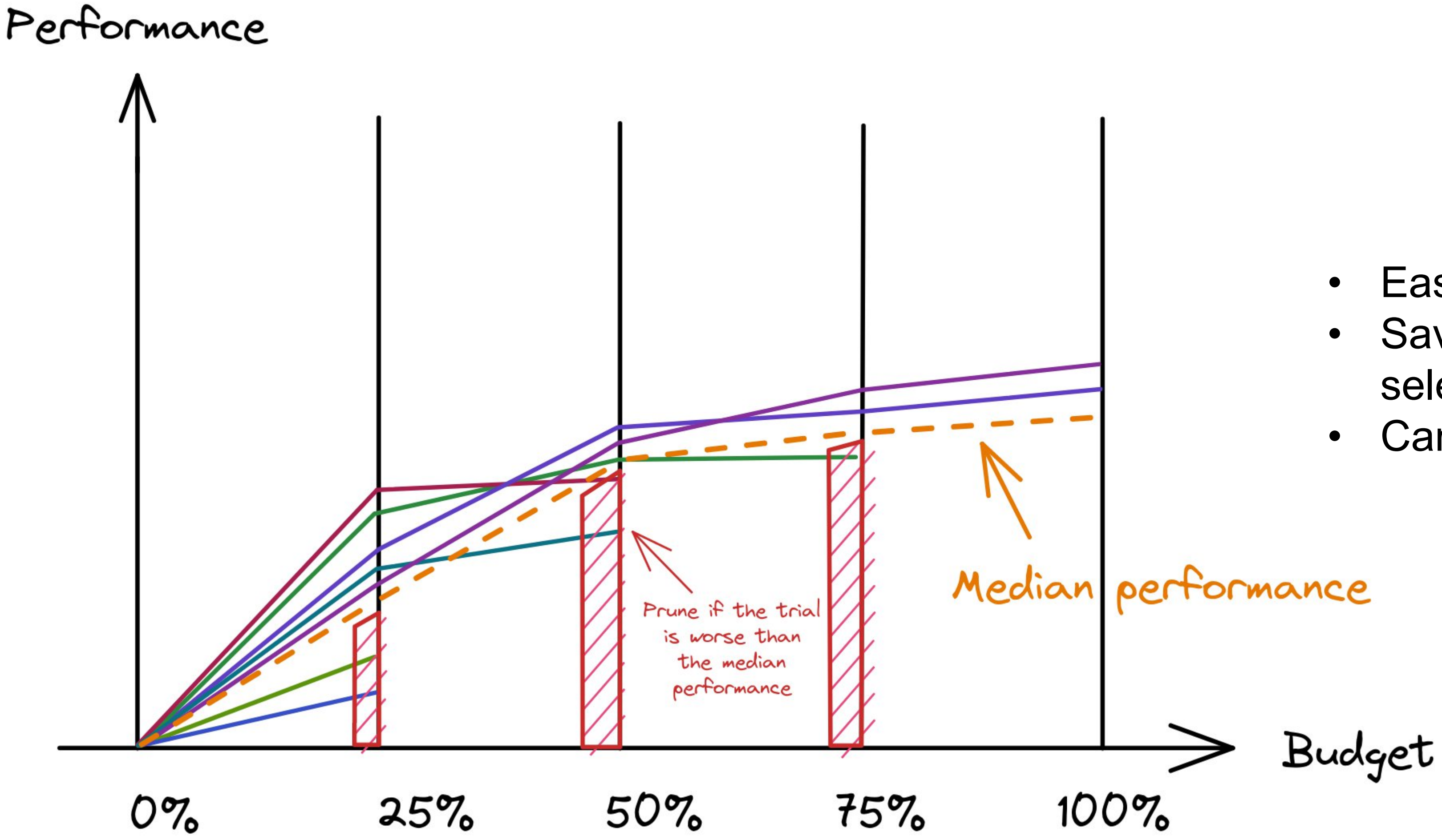
# Schedulers Algorithms / Pruners

Early Stopping ◂

SHA/ASHA ◂

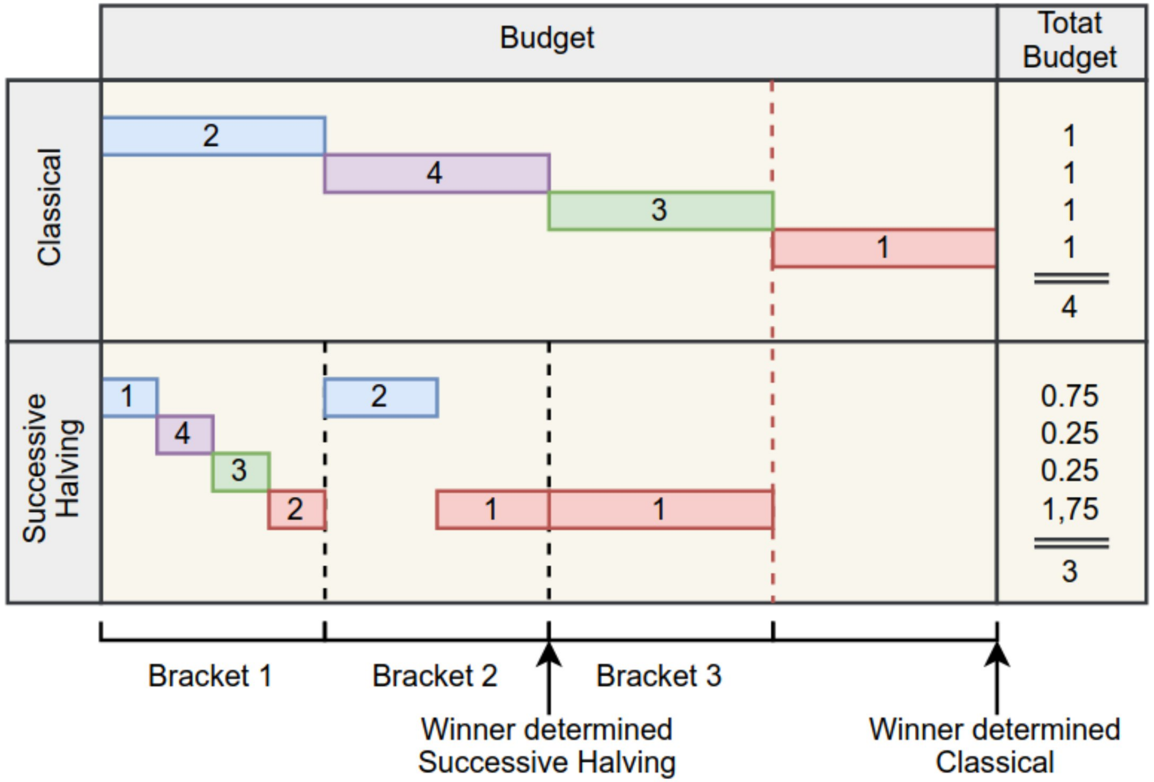
HyperBand ◂

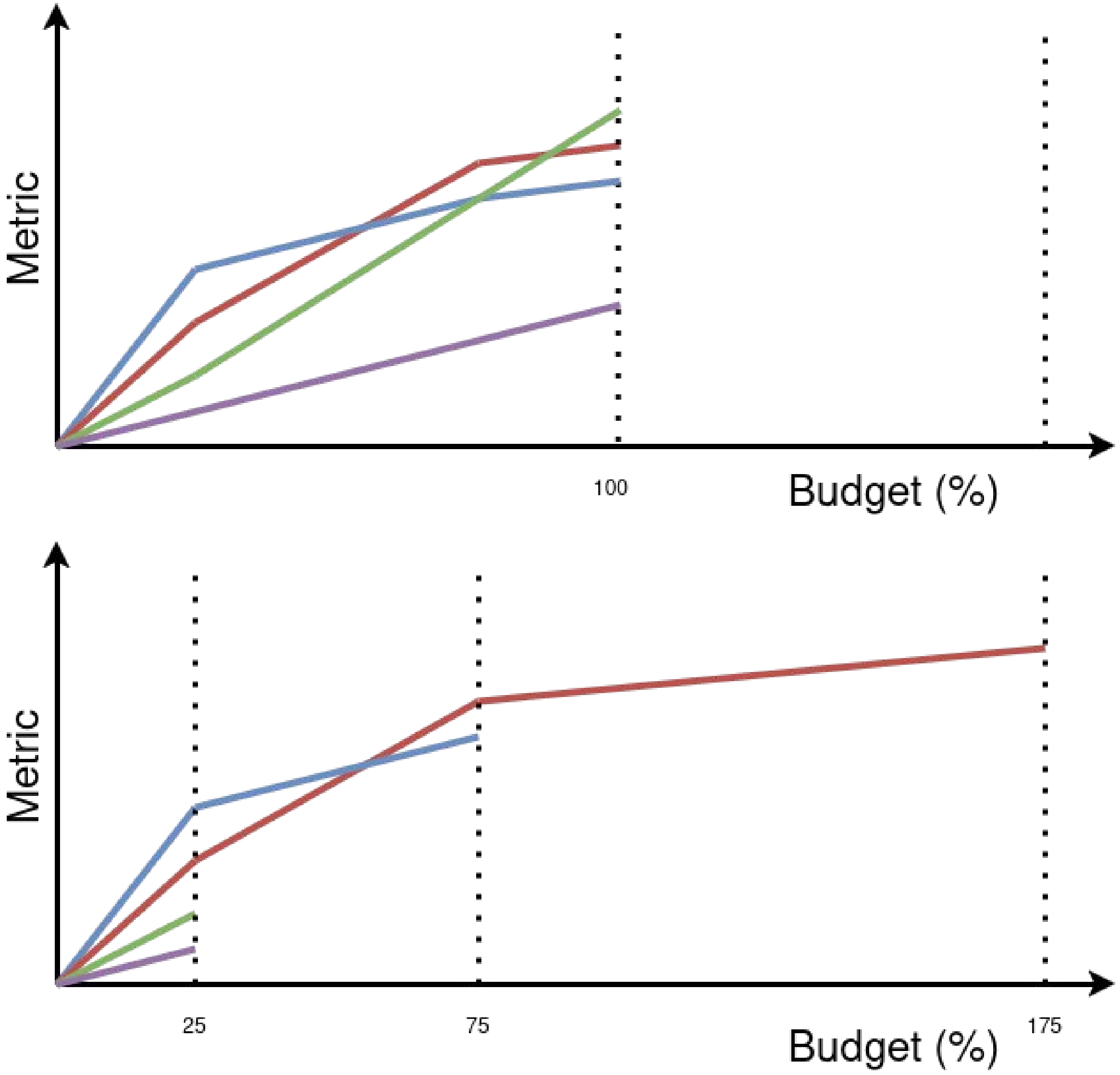
# Early Stopping




- Easy to implement
- Save ressources & make automatic selection
- Can be with acc%, time%, rank%, etc

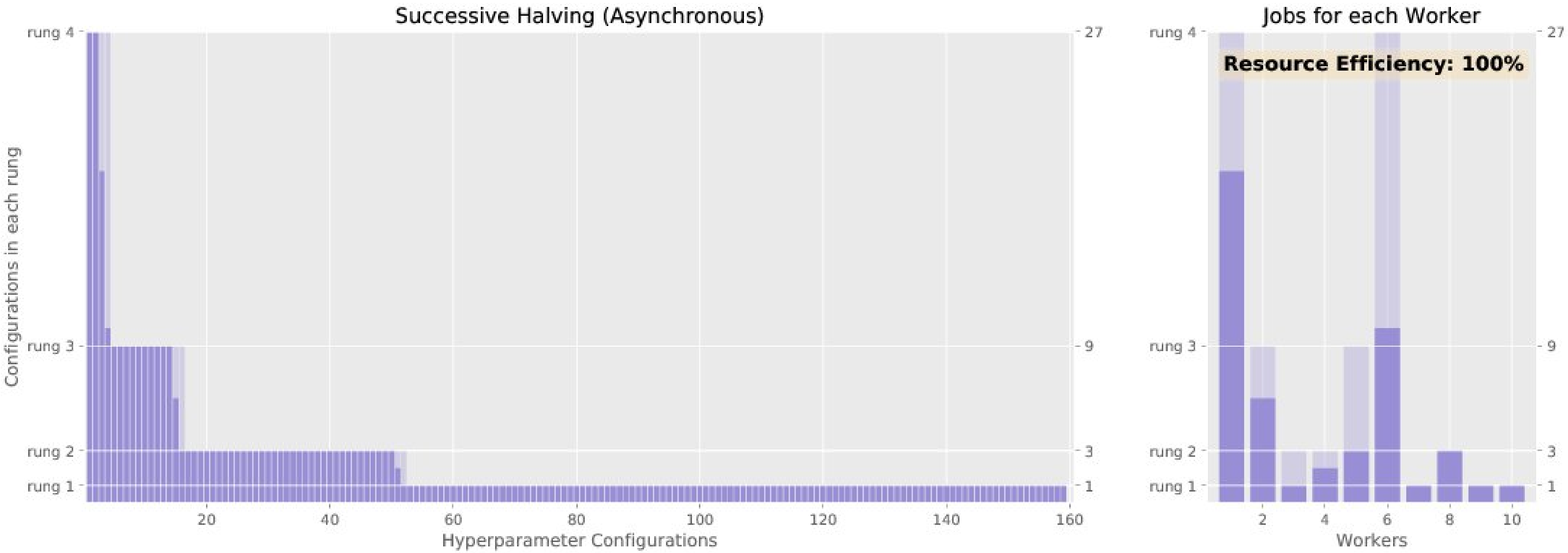
# SHA : Successive Halving Algorithm




- For sequentials trials
- Works well with small or medium model -> Trials must be fast !

# ASHA


Successive Halving (Asynchronous)
Configurations in each rung
rung 4
rung 3
rung 2
rung 1
20
40
60
80
100
120
140
160
Hyperparameter Configurations
27
9
3
1
Jobs for each Worker
Resource Efficiency: 100%
2
4
6
8
10
Workers

# Hyperband

**Algorithm 1:** HYPERBAND algorithm for hyperparameter optimization.

**input** : $R$, $\eta$ (default $\eta = 3$)
**initialization** : $s_{\max} = \lfloor \log_\eta(R) \rfloor$, $B = (s_{\max} + 1)R$
**for** $s \in \{s_{\max}, s_{\max} - 1, \dots, 0\}$ **do**
  $n = \lceil \frac{B}{R} \frac{\eta^s}{(s+1)} \rceil, \quad r = R\eta^{-s}$
  `// begin SuccessiveHalving with (n,r) inner loop`
  $T =$ `get_hyperparameter_configuration`$(n)$
  **for** $i \in \{0, \dots, s\}$ **do**
    $n_i = \lfloor n\eta^{-i} \rfloor$
    $r_i = r\eta^i$
    $L = \{$`run_then_return_val_loss`$(t, r_i) : t \in T\}$
    $T =$ `top_k`$(T, L, \lfloor n_i/\eta \rfloor)$
  **end**
**end**
**return** *Configuration with the smallest intermediate loss seen so far.*

| | $s=4$ | | $s=3$ | | $s=2$ | | $s=1$ | | $s=0$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $i$ | $n_i$ | $r_i$ | $n_i$ | $r_i$ | $n_i$ | $r_i$ | $n_i$ | $r_i$ | $n_i$ | $r_i$ |
| 0 | 81 | 1 | 27 | 3 | 9 | 9 | 6 | 27 | 5 | 81 |
| 1 | 27 | 3 | 9 | 9 | 3 | 27 | 2 | 81 | | |
| 2 | 9 | 9 | 3 | 27 | 1 | 81 | | | | |
| 3 | 3 | 27 | 1 | 81 | | | | | | |
| 4 | 1 | 81 | | | | | | | | |

Table 1: Values of $n_i$ and $r_i$ for the brackets of HYPERBAND when $R = 81$ and $\eta = 3$.

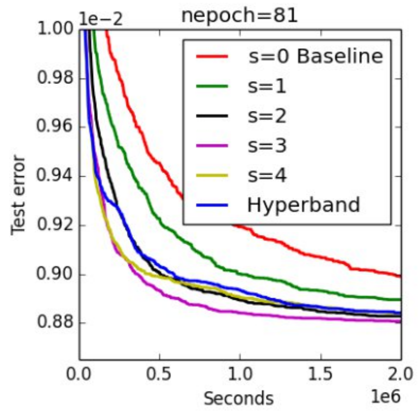
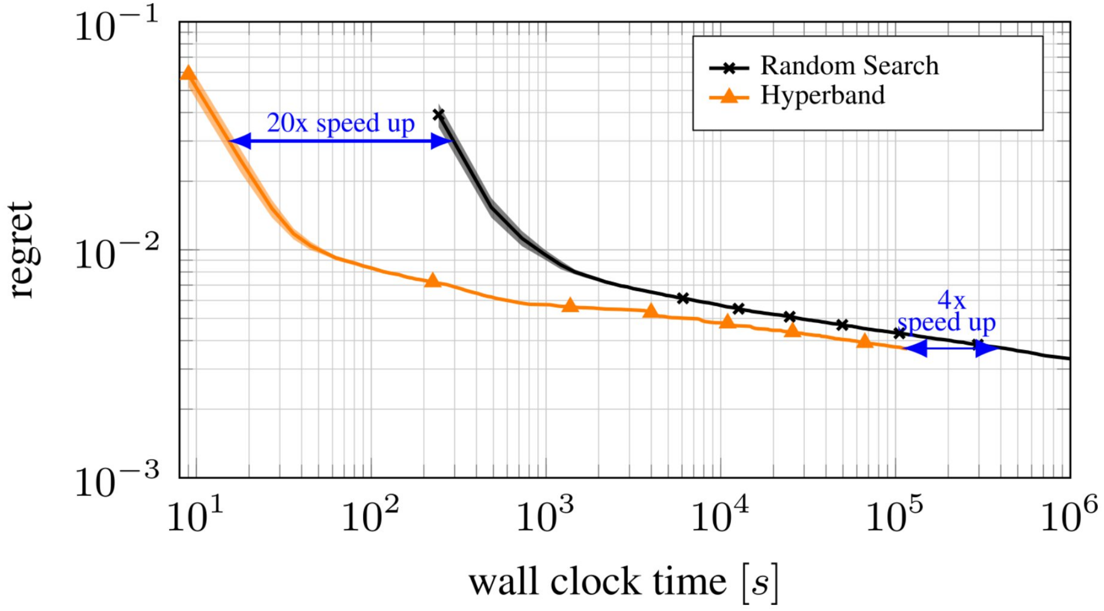



Figure 2: Performance of individual brackets $s$ and HYPERBAND.




- Repeatedly calls SuccessiveHalving but mitigate it's drawbacks
- Limited convergence

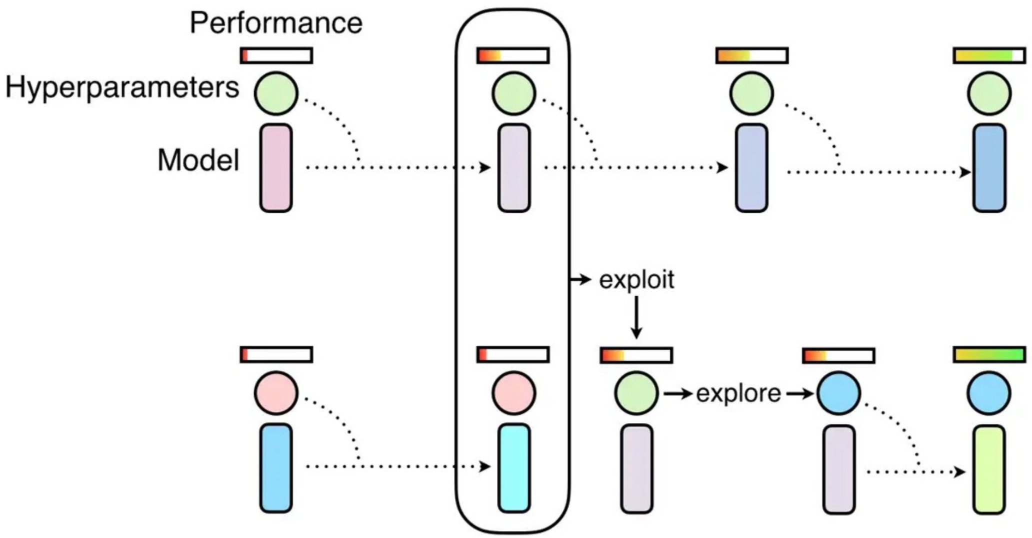
# Advanced Algorithms

***Hybrid time !***

PBT ◂

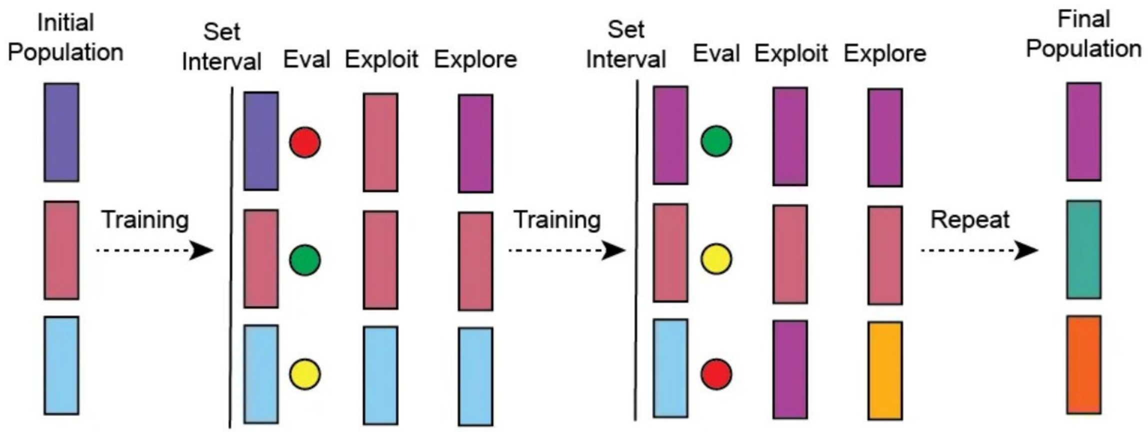
BOHB, DEHB ◂

# PBT : Population Based Training



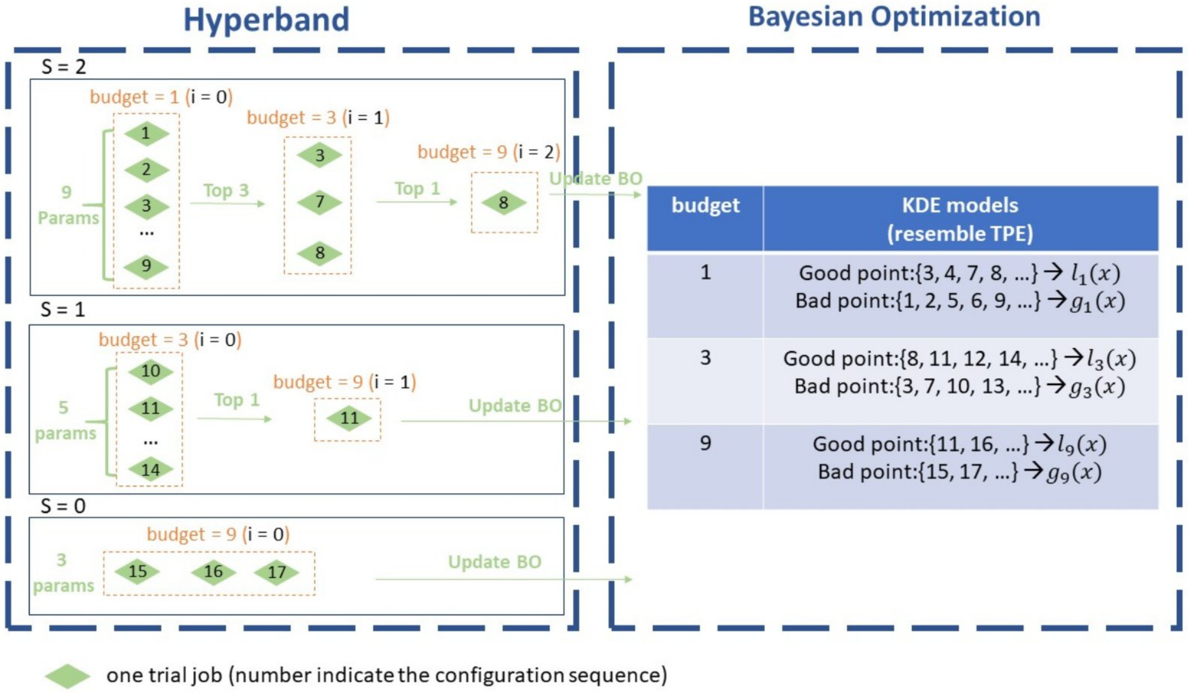

- Research and optimization of hyper parameters during training
- For large models with long and poorly parallelizable tests on a few machines.

- **Exploit** = Copy of the weights of the best model
- **Explore** = Bayesian Optimization



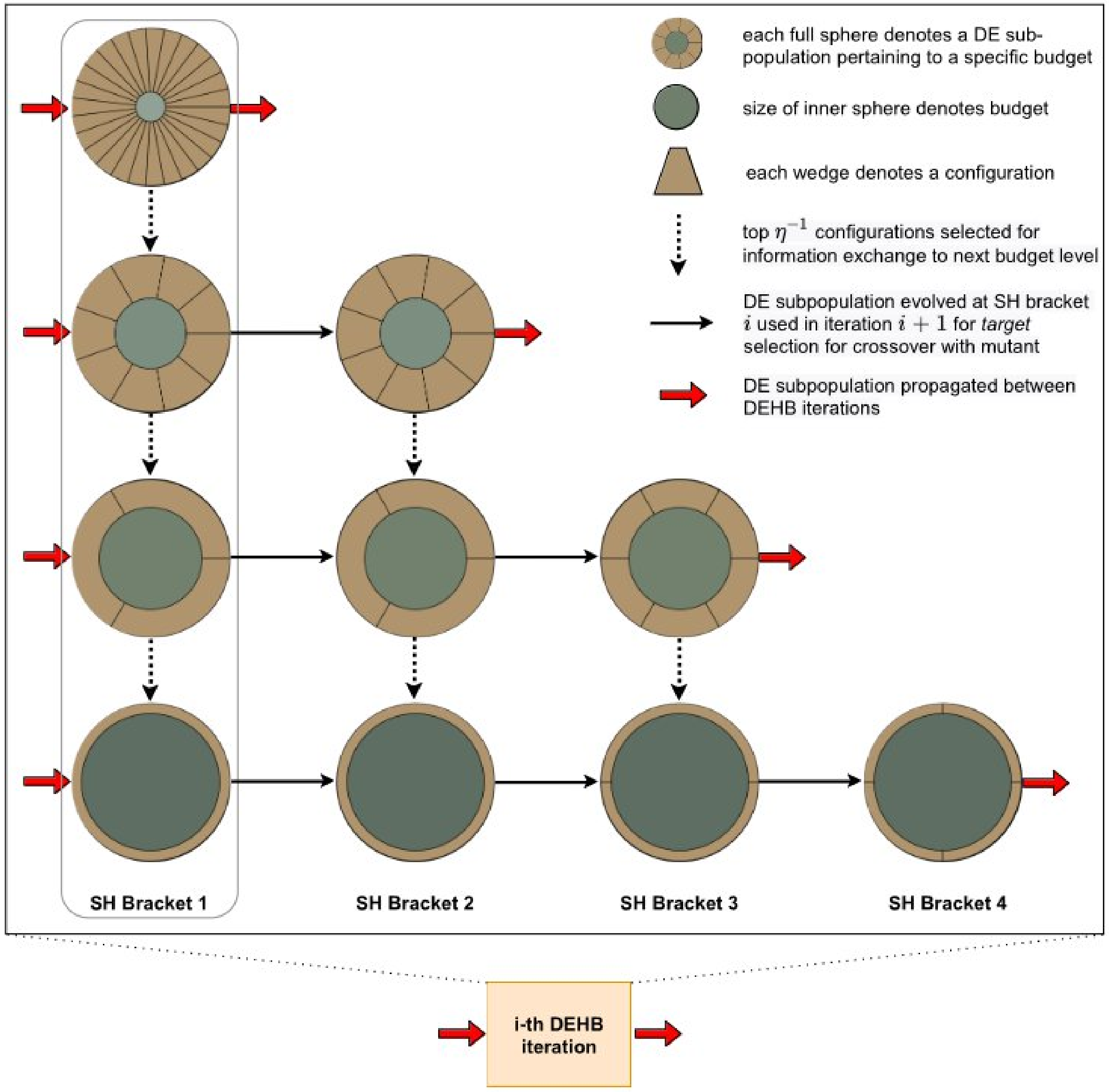
# BOHB, DEHB

## BOHB : Bayesian Optimization Hyperband




| budget | KDE models (resemble TPE) |
|---|---|
| 1 | Good point:{3, 4, 7, 8, ...} → $l_1(x)$<br>Bad point:{1, 2, 5, 6, 9, ...} → $g_1(x)$ |
| 3 | Good point:{8, 11, 12, 14, ...} → $l_3(x)$<br>Bad point:{3, 7, 10, 13, ...} → $g_3(x)$ |
| 9 | Good point:{11, 16, ...} → $l_9(x)$<br>Bad point:{15, 17, ...} → $g_9(x)$ |

## DEHB : Differential Evolution Hyperband



# Summary

- **Tree-search Algorithms** — exhaustive search space
  - Trial / Error
  - Manual Search
  - Grid Search (**GS**)
  - Random Search (**RS**)
- **Bayesian Optimization (BO)** — variants
  - Bayesian Opt. Gaussian Process (**GP**)
  - Bayesian Opt. Tree Structured Parzen Estimators (**TPE**)
  - Sequential Model-based Algorithm (**SMAC**)
- **Hyperparameter Tuning based**
  - HyperBand (**HB**)
  - Fabolas
  - Population-based Training (**PBT**)
  - Bayesian Opt. Hyperband (**BOHB**)
- **Early Stopping-based Optimization**
  - Median Stopping
  - Curve Fitting
  - Successive Halving (**SHA**) & Hyperband (**HB**)
  - Asynchronous Successive Halving (**ASHA**) & Bayesian Opt. Hyperband (**BOHB**)
- **Metaheuristic Algorithms** — Evolutionary Optimization
  - Genetic Algorithm (**GA**)
  - Particle Swarm Optimization (**PSO**)
- **Multi-fidelity Optimization** — employs low-fidelity info
  - Successive Halving (**SHA**)
  - HyperBand (**HB**)
  - Bayesian Opt. Hyperband (**BOHB**)
- **Other**
  - Radial based Function
  - Spectral Approach
  - Nelder-Mead
  - Surrogate model
  - Sequential Model-based Optimization (**SMBO**)

# Have the right tools

HPO frameworks ◂

Visualisation & Experiments Tracking ◂

# HPO Frameworks & tools

Hydra

- Based on config file
- Easy to use
- Not only used for ML/DL

OPTUNA

- Work with an objective function
- Efficient Optimization Algorithms

RAY tune

- Scalable HPO framework
- State of the art algorithms (PBT)
- Integrates with a wide range of additional HPO tools

# Experiments tracking

mlflow ou Weights & Biases +

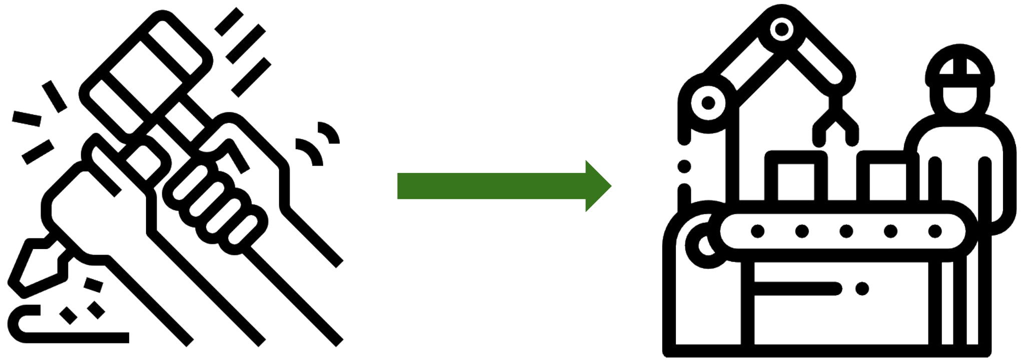
Source Control git

Data Version Control DVC

Advantages :

- allows you to save and order the results
- allows easy comparison and visualization of results
- provides all the information needed to reproduce the results

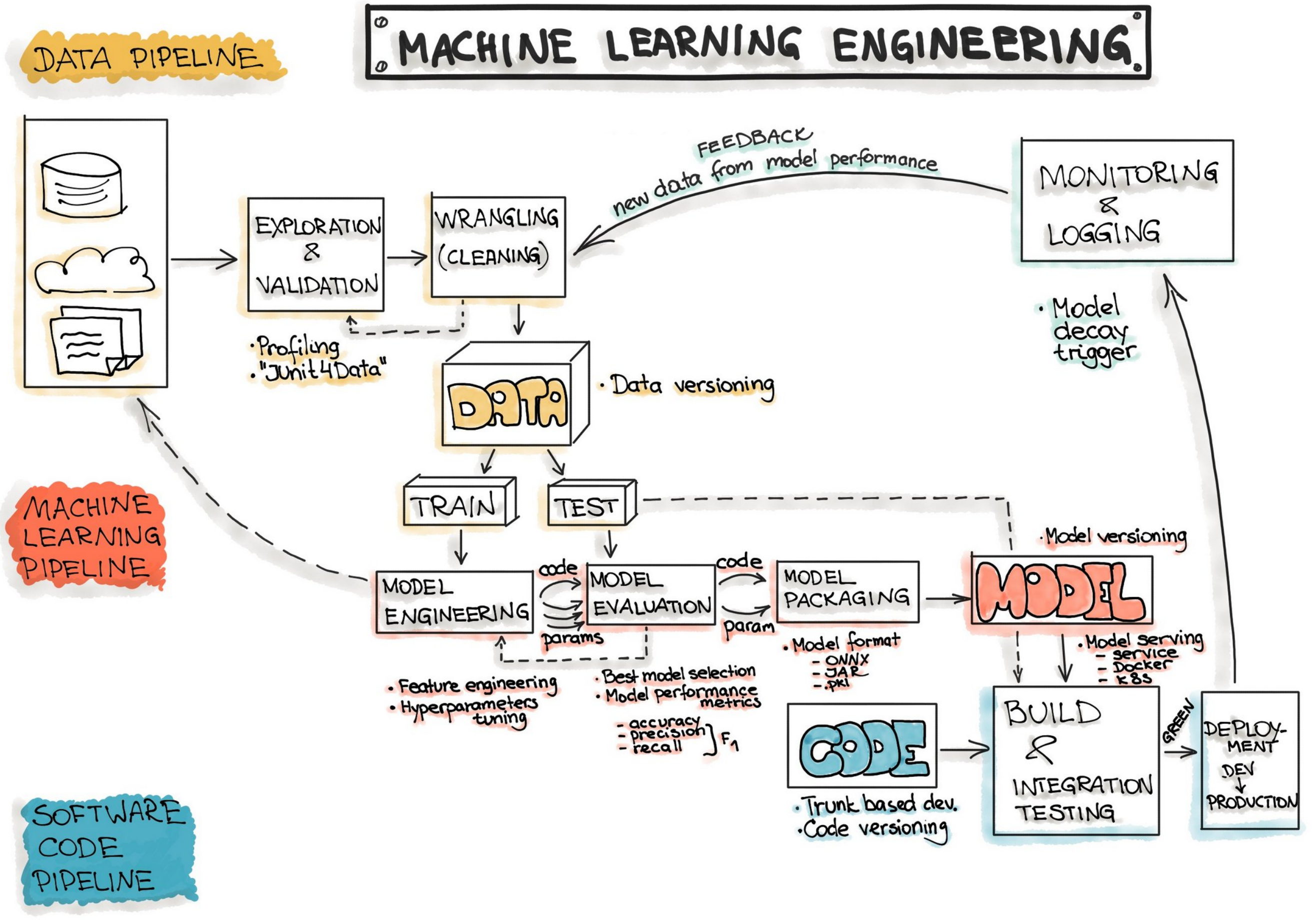
# HPO and MLOps



- As soon as our HPO requires a lot of resources (time, money or both) it is necessary to scale up and industrialize the experience process.
- Taking inspiration from MLOps processes and tools is a good start

Kubeflow

mlflow™




https://ml-ops.org/content/end-to-end-ml-workflow

# Sources


- Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges (https://wires.onlinelibrary.wiley.com/doi/pdfdirect/10.1002/widm.1484)
- https://www.automl.org/
- Gradient-based Hyperparameter Optimization Over Long Horizons (https://openreview.net/pdf?id=6x8tcREIL2W)
- Seld-Tuning networks : Bilevel Optimization of Hyperparameters using structured best-response functions (https://openreview.net/pdf?id=r1eEG20qKQ)
- https://maelfabien.github.io/machinelearning/Explorium4/#
- https://towardsdatascience.com/a-novices-guide-to-hyperparameter-optimization-at-scale-bfb4e5047150#e813
- Population Based Training : https://www.deepmind.com/blog/population-based-training-of-neural-networks
- Hyper-Parameter Optimization: A Review of Algorithms and Applications : https://arxiv.org/pdf/2003.05689