AutoML

A state-of-the-art overview

Joseph Giovanelli
j.giovanelli@unibo.it

Alma Mater Studiorum • University of Bologna
BIG • Business Intelligence Group

Whoami

Ph.D. Candidate

in Computer Science and Engineering

Main research field: **AutoML**

UPC - Barcelona

Meta-learning, **Data Pre-processing**

LUH|AI - AutoML Hannover

Multi-objective, **Preference Learning**

Research & Development

projects on BI, Big Data, **Data Mining**

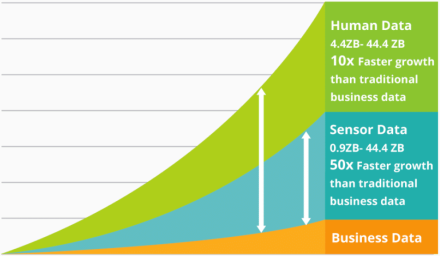
Table of contents

Introduction

The data growth

It has been reported that 2.5 quintillion bytes of data is being created everyday

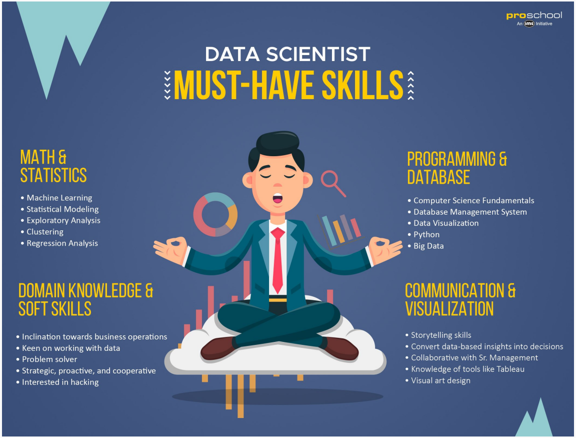
The 90% of stored data in the world, has been generated in the past two years only [1]



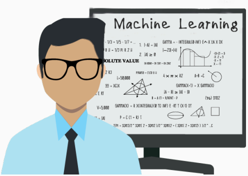
[1] Forbes: How Much Data Do We Create Every Day? May 21, 2018

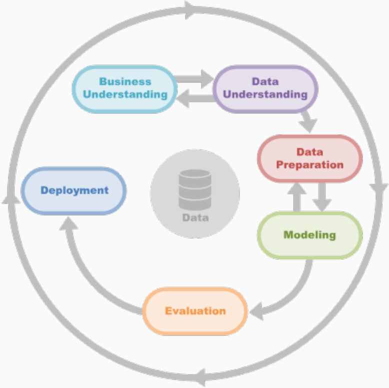
The sexiest job of the 21st century

The Data scientist has become one of the most sought figure



Data scientists use the Machine Learning toolbox to solve real-cases problems





The need

Data Scientists do not scale: [2]

- the **increasingly growing size of data** overcomes their availability
- the **numerous skills expected** (IT, mathematics, statistics, business, cooperation) make it difficult to increase their number

$\Downarrow$

More and more non-experts use data mining tools

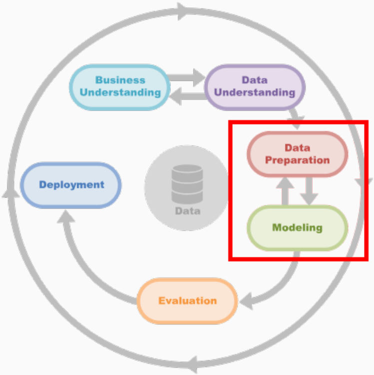
$\Downarrow$

Off the shelf solutions are required to assist them

[2] Harvard Business Review: Data Scientists Don't Scale, May 22, 2015

AutoML definition

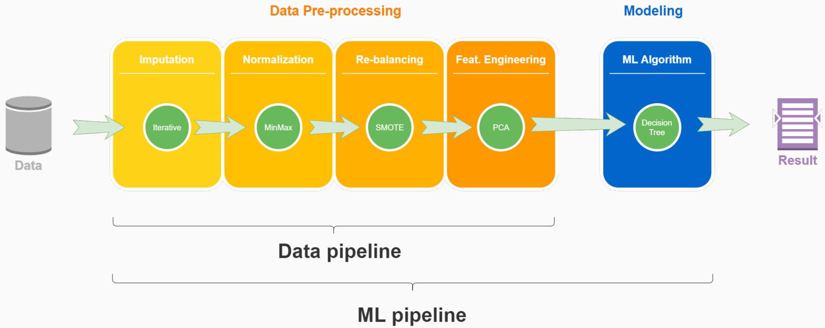
Automated Machine Learning is the process of automating the process of applying **Machine Learning**



Data scientists can spend **less tedious time on finding parameters/hyper-parameters**, and focus on the analysis

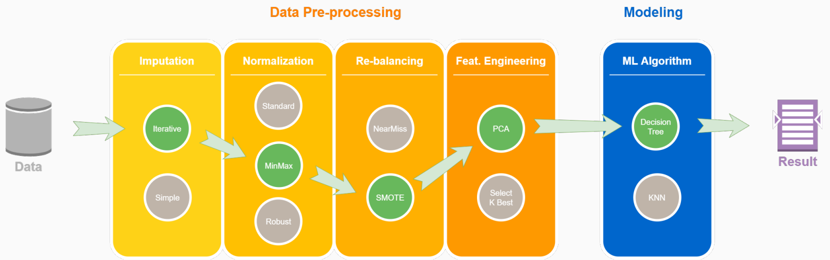
AutoML outcome

AutoML aims to find a ML pipeline



AutoML outcome

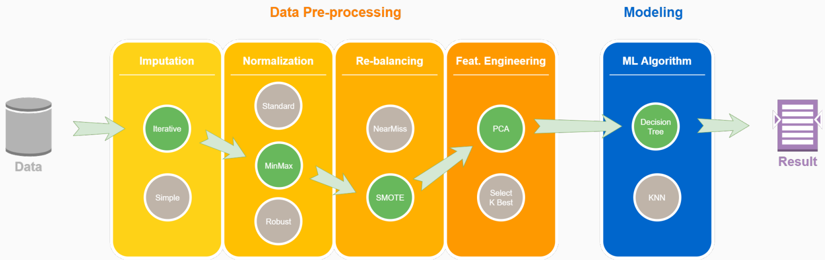
AutoML smartly explores huge search spaces.



- A data pipeline consists of a **sequence of transformations**
- Each transformation can be instantiated from a pool of **operators**
- Each operator has several **parameters**
- Each parameter has its own **search space**

AutoML outcome

AutoML smartly explores huge search spaces.

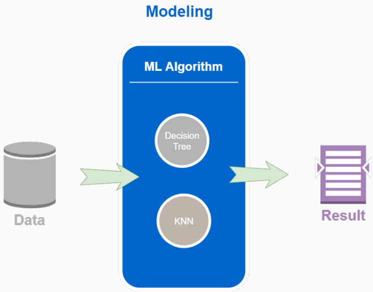


- The modeling phase involves the instantiation of a **algorithm** from a specific
- Each algorithm has several **hyper-parameters**
- Each hyper-parameter has its own **search space**

Building blocks

Auto-WEKA: the CASH problem

Auto-Weka introduces the **Combined Algorithm Selection and Hyper-parameter optimization problem (CASH)** [3]



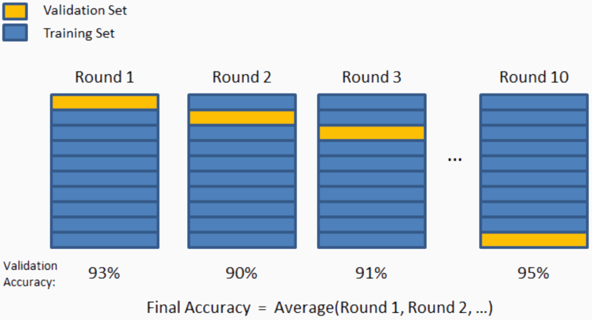
DecisioneTree.num_obj = [2, 3]
DecisioneTree.pruning = [True, False]
KNN.k = [3, 4]
KNN.distance_measure = [1 / distance, 1 - distance]

[3] Thornton, Chris, et al. "Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms." Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. 2013.

Auto-WEKA: the CASH problem

Given

- *a* **data-set** D *divided into* $D_{train}, D_{validation}$ according to k cross-validation
  - $D_{train} = \{D^1_{train}, \ldots, D^i_{train}, \ldots, D^k_{train}\}$
  - $D_{validation} = \{D^1_{validation}, \ldots, D^i_{validation}, \ldots, D^k_{validation}\}$
  - $D^i_{train} = D \setminus D^i_{validation}$



Auto-WEKA: the CASH problem

Given

- *a* ***data-set*** *D divided into $D_{train}, D_{validation}$* according to k cross-validation
  - $D_{train} = \{D^1_{train}, \ldots, D^i_{train}, \ldots, D^k_{train}\}$
  - $D_{validation} = \{D^1_{validation}, \ldots, D^i_{validation}, \ldots, D^k_{validation}\}$
  - $D^i_{train} = D \backslash D^i_{validation}$
- *a set of* **algorithms** $\mathcal{A} = \{A^1, \ldots, A^j, \ldots, A^n\}$ *with associated* ***hyper-parameter spaces*** $\{\Theta^1, \ldots, \Theta^j, \ldots, \Theta^n\}$

For instance:

$A^1 = DecisionTree$
$\Theta^1 = \{$
    num_obj = [2, 3],
    pruning = [True, False]
$\}$

$A^2 = KNN$
$\Theta^2 = \{$
    k = [3, 4],
    distance_measure = [1 / distance, 1 - distance]
$\}$

Auto-WEKA: the CASH problem

Given

- *a* ***data-set*** *D divided into $D_{train}, D_{validation}$* according to k cross-validation
  - $D_{train} = \{D^1_{train}, \ldots, D^i_{train}, \ldots, D^k_{train}\}$
  - $D_{validation} = \{D^1_{validation}, \ldots, D^i_{validation}, \ldots, D^k_{validation}\}$
  - $D^i_{train} = D \backslash D^i_{validation}$
- *a set of* ***algorithms*** $\mathcal{A} = \{A^1, \ldots, A^j, \ldots, A^n\}$ *with associated* ***hyper-parameter spaces*** $\{\Theta^1, \ldots, \Theta^j, \ldots, \Theta^n\}$
- *an* ***evaluation metric*** $\mathcal{M}(A^j_\theta, D^i_{train}, D^i_{validation})$

For instance:

- Accuracy
- Precision
- Recall

Auto-WEKA: the CASH problem

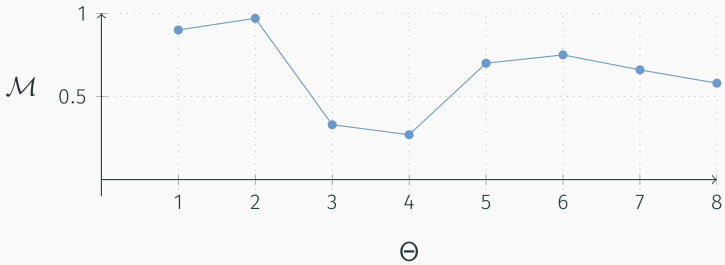
Given

- *a* ***data-set*** *D divided into $D_{train}, D_{validation}$* according to k cross-validation
  - $D_{train} = \{D^1_{train}, \ldots, D^i_{train}, \ldots, D^k_{train}\}$
  - $D_{validation} = \{D^1_{validation}, \ldots, D^i_{validation}, \ldots, D^k_{validation}\}$
  - $D^i_{train} = D \backslash D^i_{validation}$
- *a set of* ***algorithms*** *$\mathcal{A} = \{A^1, \ldots, A^n\}$ with associated* ***hyper-parameter spaces*** *$\Theta^1, \ldots, \Theta^n$*
- *an* ***evaluation metric*** *$\mathcal{M}(A^j, D^i_{train}, D^i_{validation})$*

We are searching for

$$A^*_{\theta^*} \epsilon \operatorname*{arg\,max}_{A^j \epsilon \mathcal{A}, \theta \epsilon \Theta^j} \frac{1}{k} \sum_{i=1}^{k} \mathcal{M}(A^j_\theta, D^i_{train}, D^i_{validation}) \qquad \text{(CASH)}$$

Auto-Weka: CASH reformulation



θ	Algorithm	num_obj	pruning	k	distance_measure
1	DecisionTree	2	True		
2	DecisionTree	2	False		
3	DecisionTree	3	True		
4	DecisionTree	3	False		
5	KNN			3	1/distance
6	KNN			3	1-distance
7	KNN			4	1/distance
8	KNN			4	1-distance

Auto-Weka: search space

Classifier	Categorical	Numeric
Bayes Net	2	0
Naive Bayes	2	0
Naive Bayes Multinomial	0	0
Gaussian Process	3	6
Linear Regression	2	1
Logistic Regression	0	1
Single-Layer Perceptron	5	2
Stochastic Gradient Descent	3	2
SVM	4	6
Simple Linear Regression	0	0
Simple Logistic Regression	2	1
Voted Perceptron	1	2
KNN	4	1
K-Star	2	1
Decision Table	4	0
RIPPER	3	1
M5 Rules	3	1
1-R	0	1
PART	2	2
0-R	0	0
Decision Stump	0	0
C4.5 Decision Tree	6	2
Logistic Model Tree	5	2
M5 Tree	3	1
Random Forest	2	3
Random Tree	4	4
REP Tree	2	3
Locally Weighted Learning*	3	0
AdaBoost M1*	2	2
Additive Regression*	1	2
Attribute Selected*	2	0
Bagging*	1	2
Classification via Regression*	0	0
LogitBoost*	4	4
MultiClass Classifier*	3	0
Random Committee*	0	1
Random Subspace*	0	2
Voting$^+$	1	0
Stacking$^+$	0	0

The table represents the considered classifiers in Auto-WEKA. Categorical and Numeric refer to the number of hyper-parameters of each kind for each classifier.

Explore all the configurations is **unfeasible** (786 hyper-parameters)

$\Rightarrow$ explore few of them but in a smart way

CASH resolution approches[4]

- Model free methods
  - Grid search
  - Random search
  - Heuristics
    - Ant colony optimization
    - Particle Swarm Optimization
    - Simulate Annealing
  - Genetic algorithms
  - Multi-resolution optimization
    - Successive Halving
    - Hyper-Band
- Bayesian optimization

[4] Elshawi, R., Maher, M., Sakr, S. (2019). Automated machine learning: State-of-the-art and open challenges.

Bayesian Optimization

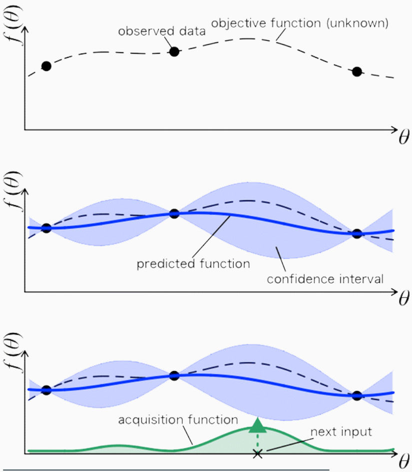
Explore all the configurations is **unfeasible**

$\Rightarrow$ explore few of them but in a smart way

We want to:

- divide the exploration in **iterations**
- keep track of **past evaluation scores**
- build/update a **probabilistic model**
- find promising configurations to explore

Bayesian Optimization[5]



- **objective function**: the function we want to maximize
- **observed data**: the tested hyper-parameters configurations

The probabilistic model consists of:

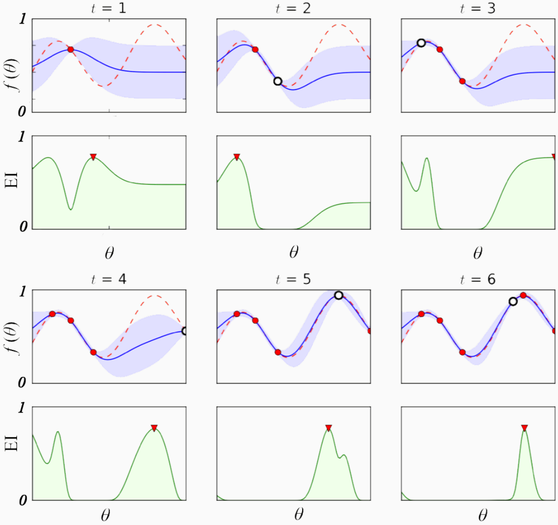
- **predicted function**, an estimation of the objective
- **confidence interval**, which indicates the possible variance

The acquisition functions suggests the next configuration to visit. It regulates:

- **explotation**
- **exploration**

[5]Brochu, Eric, Vlad M. Cora, and Nando De Freitas. ”A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning.” (2010).

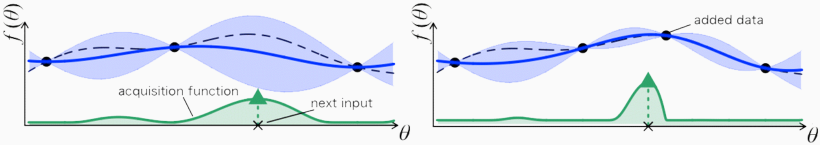
Bayesian Optimization: working example



Bayesian Optimization: SMBO

Sequential Model-Based Optimization (SMBO) is a formalization of Bayesian Optimization:

1. **Evaluate** some **random hyper-parameters configurations**
2. **Build** a probabilistic model
3. **Exploit** the **model** and the acquisition function to **find the next hyper-parameters configuration** to evaluate
4. **Evaluate the hyper-parameters configuration**
5. **Update** the **probabilistic model** incorporating the new results
6. **Repeat** steps 3–5 **until the budget exceeded**



Bayesian Optimization: probabilistic models

The **implementations of SMBO** differ in how they construct the **probabilistic model**

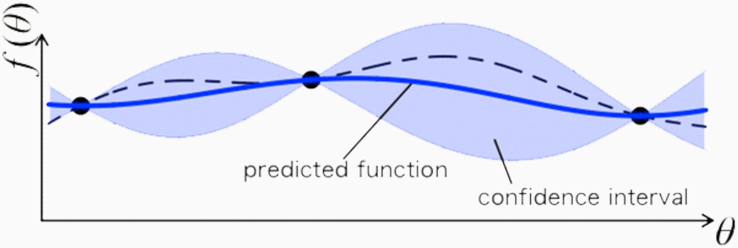
- using Gaussian Process (GP)
- using Tree Parzen Estimators (TPE)
- using Random Forest (SMAC)[6]

[6] F. Hutter, H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. Proc. of LION-5, pages 507–523, 2011.

Bayesian Optimization: SMAC

Random Forest is not usually treated as probabilistic models.
SMAC obtains:

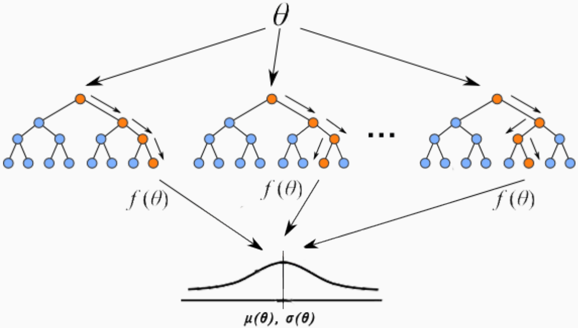
- the predicted function, as the **mean** over the predictions of its individual trees for θ
- the confidence interval, as the **variance** over the predictions of its individual trees for θ



Bayesian Optimization: SMAC

Random Forest is not usually treated as probabilistic models.
SMAC obtains:

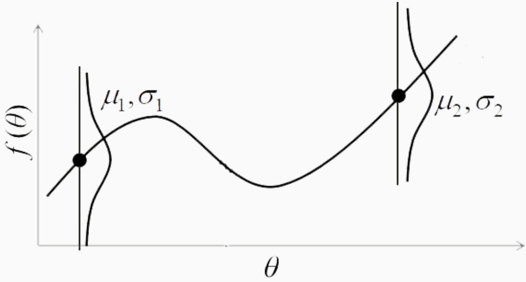
- the predicted function, as the **mean** over the predictions of its individual trees for θ
- the confidence interval, as the **variance** over the predictions of its individual trees for θ



Bayesian Optimization: SMAC

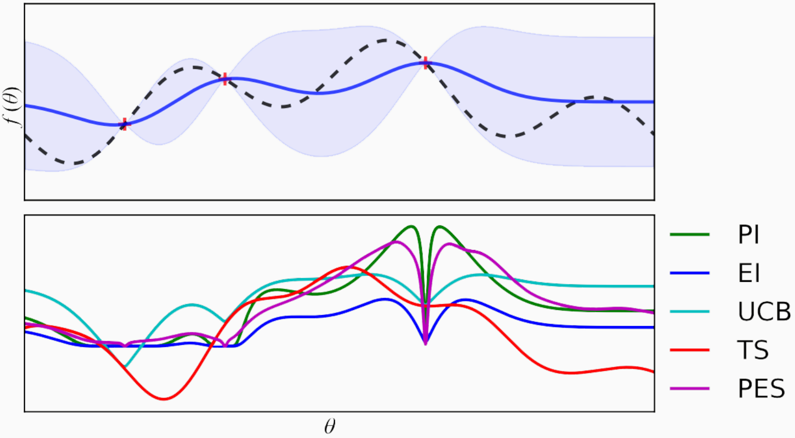
Random Forest is not usually treated as probabilistic models.
SMAC obtains:

- the predicted function, as the **mean** over the predictions of its individual trees for θ
- the confidence interval, as the **variance** over the predictions of its individual trees for θ



Bayesian Optimization: acquisition functions

The acquisition function is the criteria by which the next set of hyper-parameters are chosen from the surrogate function



Bayesian Optimization: Sum up

Pros:

- converge with a **low budget**
- provide **fine-grained information**

Cons:

- **slow to start** for large hyper-parameter spaces
  $\Rightarrow$ a.k.a cold-start problem
- there is no optimization to reduce the **evaluation costs**

State of the art

AutoML tools

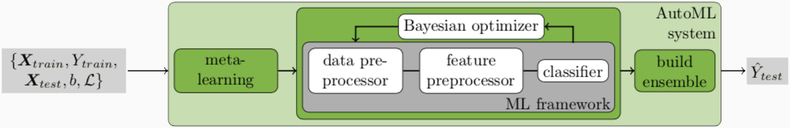
There are three main kinds of framework:

- Cloud-Based
  - Google AutoML
  - Amazon AutoML
  - Azure AutoML
  - Data Iku
  - Data Robot
- Distributed
  - MLBase
  - TrasmogrifAI
  - MLBox
  - ATM
  - Rafiki
- Centralised
  - Auto-Weka
  - Auto-MEKA
  - Auto-Sklearn
  - HyperOpt
  - HyperOpt-Sklearn
  - TPOT
  - SmartML
  - H2O

Auto-Sklearn [7]

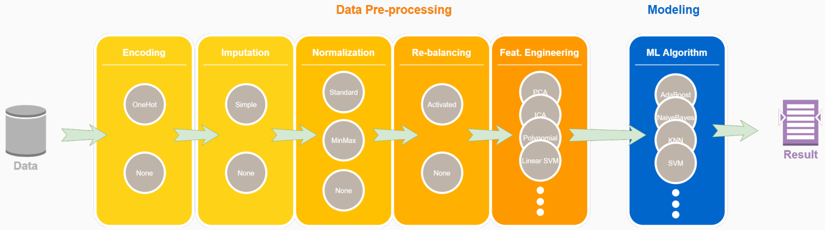
Architecture:

- Meta-learning
- Optimization
  - Scikit-learn as ML framework
  - SMAC as Bayesian optimizer
- Enseambling



[7] Feurer, Matthias, et al. "Auto-sklearn: efficient and robust automated machine learning." Automated Machine Learning. Springer, Cham, 2019. 113-134.

Auto-Sklearn: Optimization



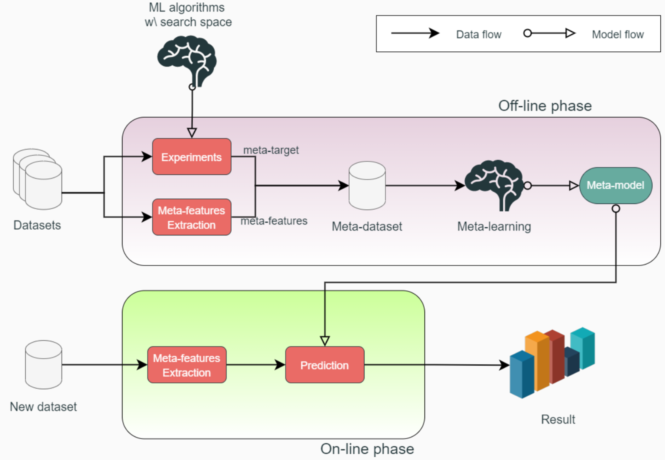
name	#λ	cat (cond)	cont (cond)
AdaBoost (AB)	4	1 (-)	3 (-)
Bernoulli naïve Bayes	2	1 (-)	1 (-)
decision tree (DT)	4	1 (-)	3 (-)
extreml. rand. trees	5	2 (-)	3 (-)
Gaussian naïve Bayes	-	-	-
gradient boosting (GB)	6	-	6 (-)
kNN	3	2 (-)	1 (-)
LDA	4	1 (-)	3 (1)
linear SVM	4	2 (-)	2 (-)
kernel SVM	7	2 (-)	5 (2)
multinomial naïve Bayes	2	1 (-)	1 (-)
passive aggressive	3	1 (-)	2 (-)
QDA	2	-	2 (-)
random forest (RF)	5	2 (-)	3 (-)
Linear Class. (SGD)	10	4 (-)	6 (3)

(a) classification algorithms

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

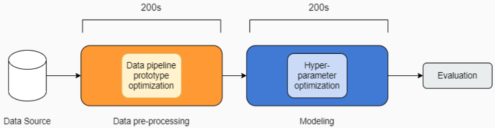
(b) preprocessing methods

Auto-Sklearn: Meta-learning



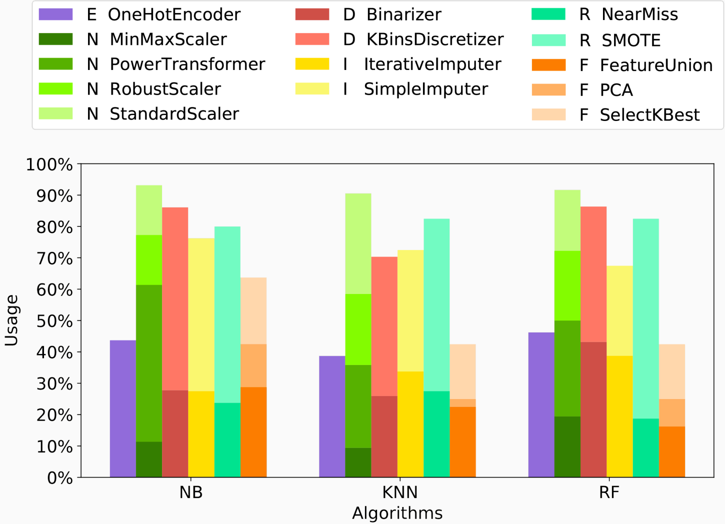
Getting insight with meta-learning

ID	Pipeline prototype	ID	Pipeline prototype
1	$I \to E \to N \to D \to F \to R$	13	$I \to E \to F \to N \to D \to R$
2	$I \to E \to N \to D \to R \to F$	14	$I \to E \to F \to N \to R \to D$
3	$I \to E \to N \to F \to D \to R$	15	$I \to E \to F \to D \to N \to R$
4	$I \to E \to N \to F \to R \to D$	16	$I \to E \to F \to D \to R \to N$
5	$I \to E \to N \to R \to D \to F$	17	$I \to E \to F \to R \to N \to D$
6	$I \to E \to N \to R \to F \to D$	18	$I \to E \to F \to R \to D \to N$
7	$I \to E \to D \to N \to F \to R$	19	$I \to E \to R \to N \to D \to F$
8	$I \to E \to D \to N \to R \to F$	20	$I \to E \to R \to N \to F \to D$
9	$I \to E \to D \to F \to N \to R$	21	$I \to E \to R \to D \to N \to F$
10	$I \to E \to D \to F \to R \to N$	23	$I \to E \to R \to D \to F \to N$
11	$I \to E \to D \to R \to N \to F$	23	$I \to E \to R \to F \to N \to D$
12	$I \to E \to D \to R \to F \to N$	24	$I \to E \to R \to F \to D \to N$



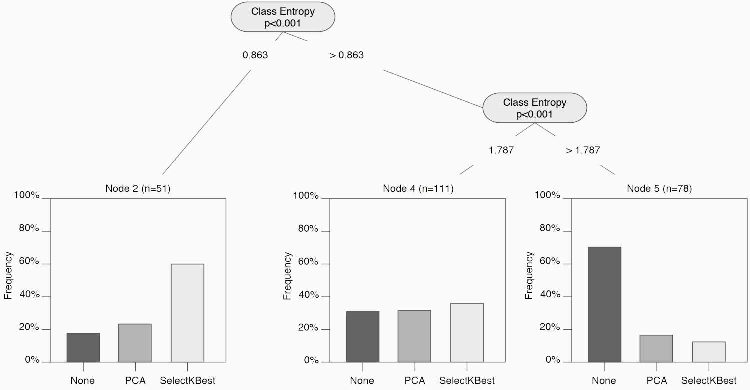
Getting insight with meta-learning

Percentage of use of transformations' operators:



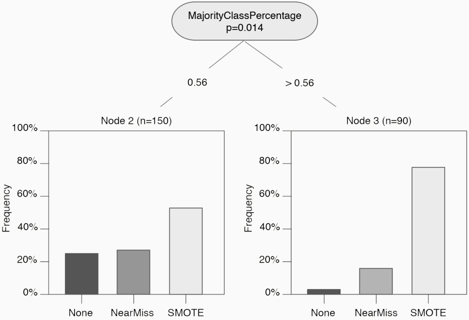
Getting insight with meta-learning

Conditional Inference Tree built for Features Engineering:



Getting insight with meta-learning

Conditional Inference Tree built for Rebalancing:



Meta-learning as a warm-starting procedure

Pros:

- converge with a **low budget**
- provide **fine-grained information**

Cons:

- **slow to start** for large hyper-parameter spaces
- there is no optimization to reduce the **evaluation costs**

Meta-learning as a warm-starting procedure

Pros:

- converge with a **low budget**
- provide **fine-grained information**

Cons:

- **slow to start** for large hyper-parameter spaces
  $\Rightarrow$ a.k.a cold-start problem
- there is no optimization to reduce the **evaluation costs**

Meta-learning as a warm-starting procedure

Pros:

- converge with a **low budget**
- provide **fine-grained information**

Cons:

- **slow to start** for large hyper-parameter spaces
  $\Rightarrow$ a.k.a cold-start problem
- there is no optimization to reduce the **evaluation costs**
  $\Rightarrow$ multi-fidelity optimization

Multi-fidelity optimization

Pros:

- **Evaluate** configurations **incrementally** (e.g., folds by folds)
- **Discard non-performing** configurations

Cons:

- **Model-free** approaches

Main methods:

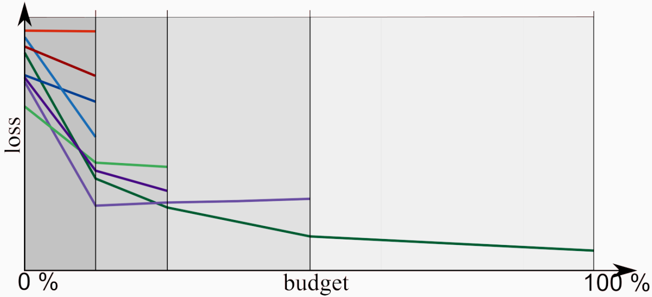
1. **Successive halving**
2. **Hyper Band**

Successive halving

Given:

- N different configurations
- a precise budget β

The evaluation starts for **all the N configurations** concurrently

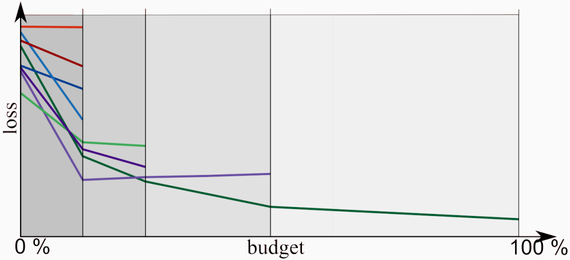


At each cut just **the best halve of the configurations are kept**

Hyper Band

Successive Halving issues:

- How we decide N **number of configurations**?
- How we decide the **number of cuts**?

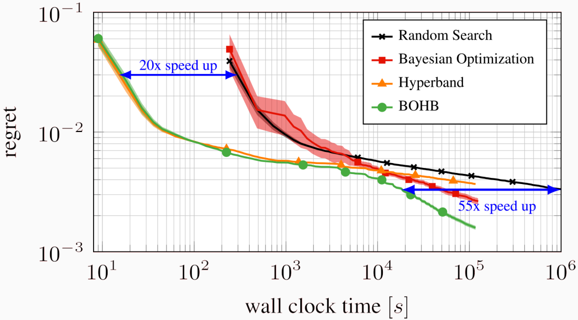


Hyper Band performs frequently **Successive Halving** varying:

- the **number of tested configurations**
- the **budget**

Bayesian Optimization Hyper Band (BOHB)

Bayesian Optimization:	• Model-based	• really slow
Hyper Band:	• Model-free	• really fast



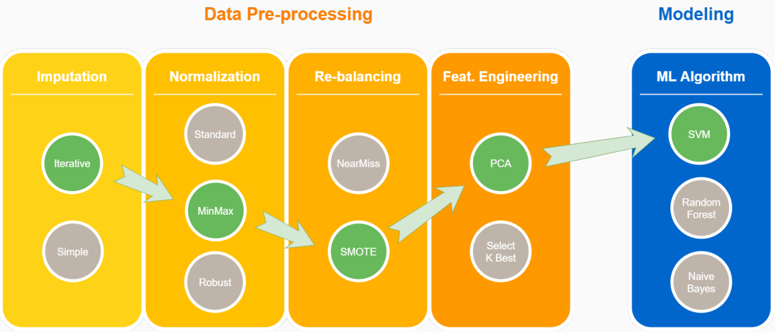
BOHB makes the most out of Bayesian Optimization and Hyper Band:

- **Bayesian Optimization** to not go blindly
- **Hyper Band** to evaluate *N* iterations concurrently

Human-centered AutoML

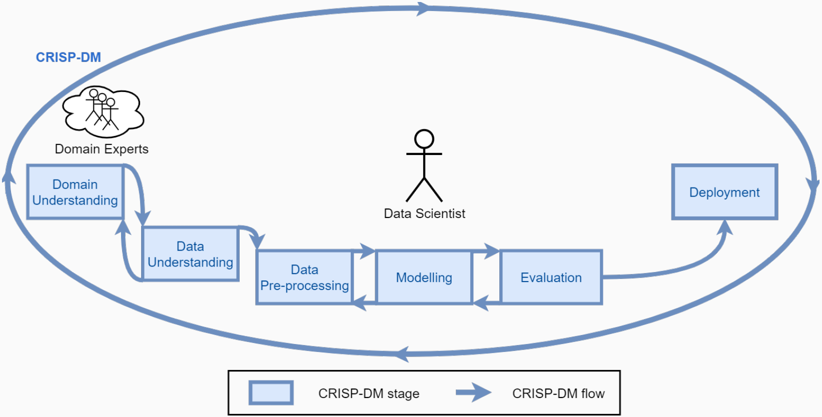
AutoML

AutoML aims at find the best ML pipeline



- At each step, a **technique** must be selected
- For each technique, a set of **hyper-parameters** must be set
- Each hyper-parameter has its own **search space**

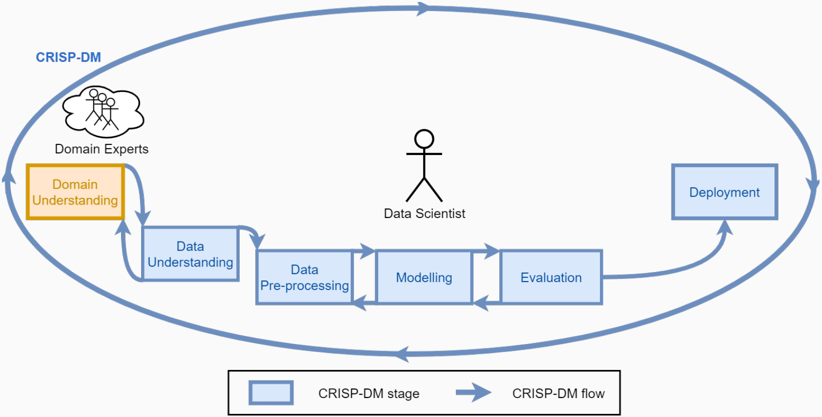
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

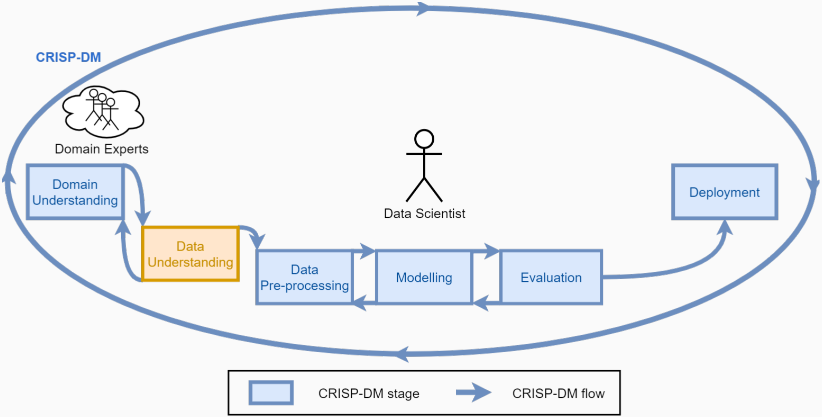
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

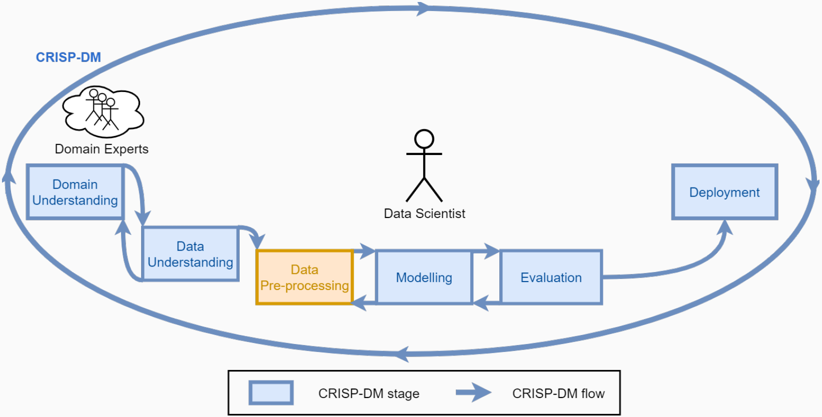
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

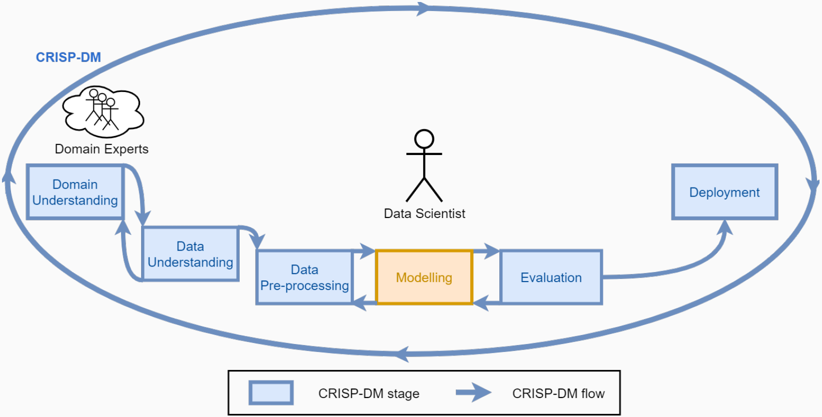
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

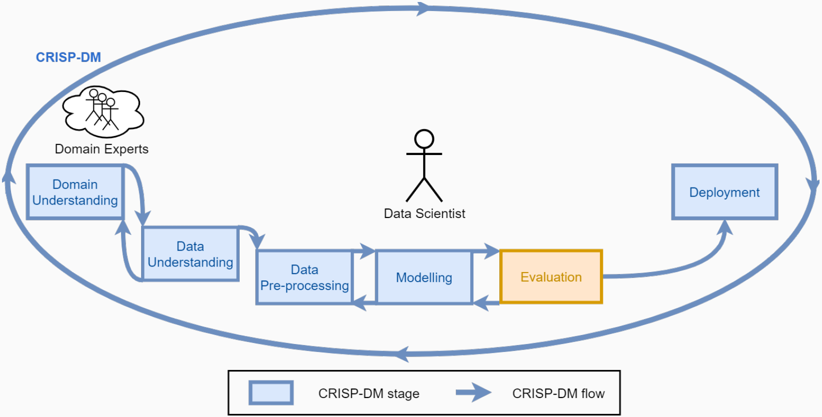
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

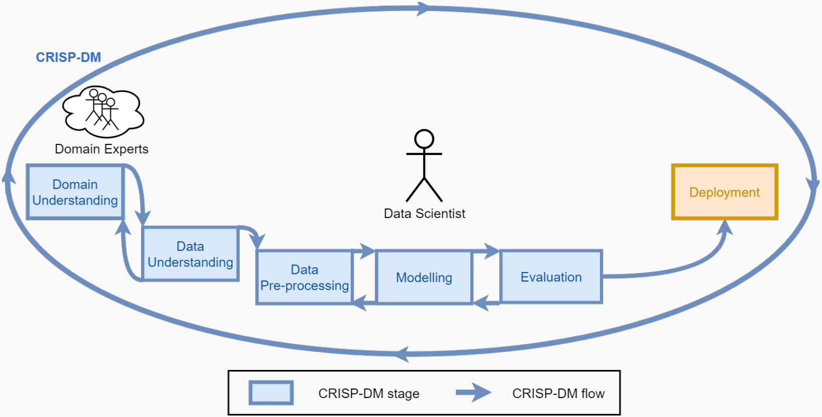
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

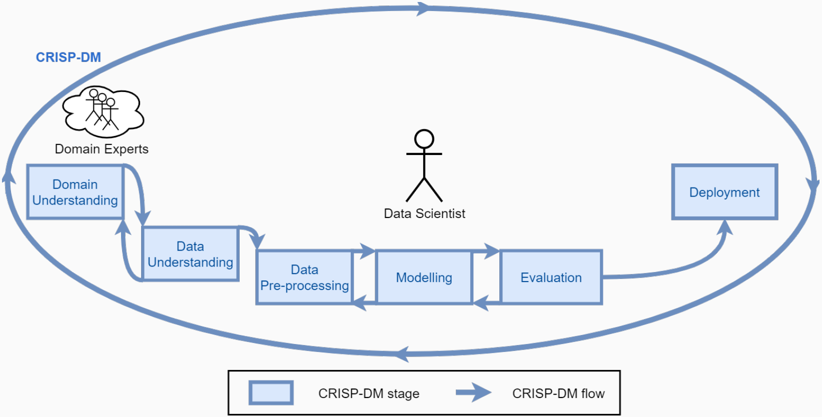
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

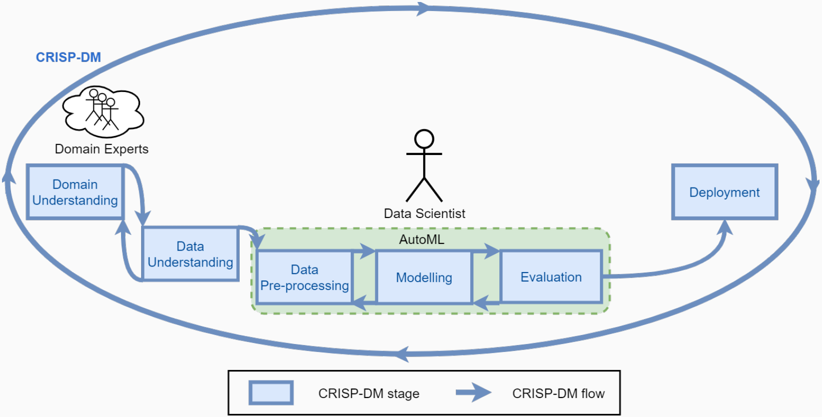
CRISP-DM: Cross Industry Standard Process for Data Mining



CRISP-DM enables the exploration of **ML Constraints**:

- domain-related;
- transformation-related;
- cross-cutting (e.g., ethical, legal).
- data-related;
- algorithm-related;

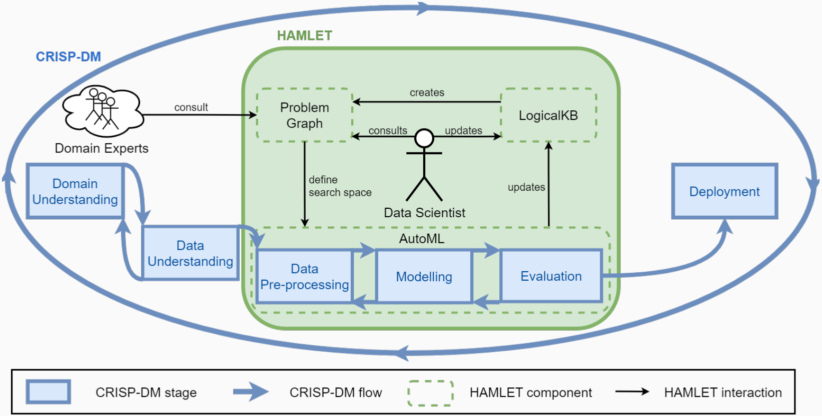
AutoML



AutoML aims at automating the ML pipeline instantiation:

- it is difficult to consider all the constraints together;
- it is not transparent;
- it doesn't allow a proper knowledge augmentation.

HAMLET: Human-centric AutoML via Logic and Argumentation



HAMLET leverages :

- **Logic** to give a structure to the knowledge;
- **Argumentation** to deal with inconsistencies, and revise the results.

Questions?