AutoML: Practical Considerations

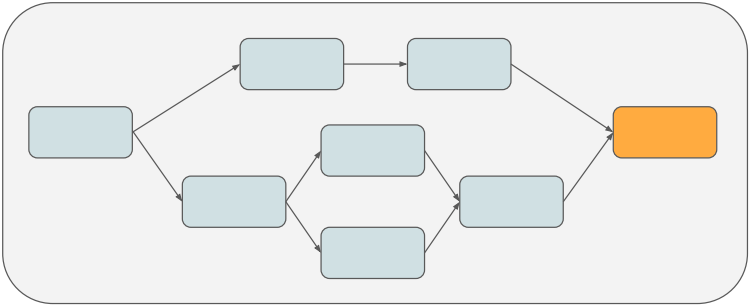
Machine Learning Pipelines

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Janek Thomas Joaquin Vanschoren

Pipelines and Workflows

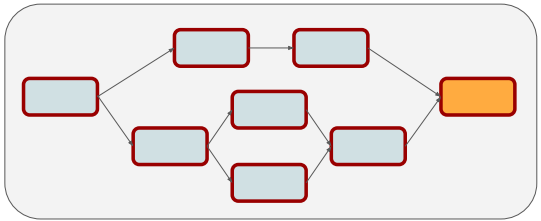
To build AutoML systems we need a language to describe:

- Machine learning algorithms
- Preprocessing operations
- Ensemble methodes like model averaging and stacking
- How data is passed from one stage to another

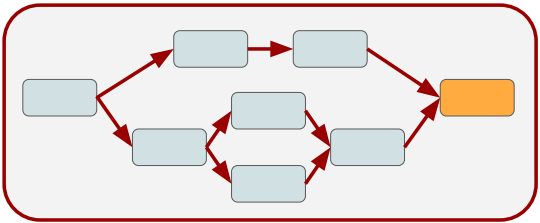


Pipelines and Workflows

Nodes: What is happening?



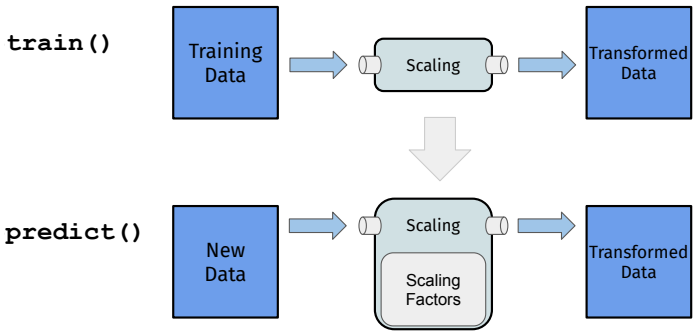
Edges: In what sequence is it happening?



$\longrightarrow$ We represent a ML workflow as a stateful directed Acyclic graph with nodes of operations and edged of data flow between them.

Training nodes

Each node can have *hyperparameters* $\mathbf{\Lambda}_{\text{node}}$, has to be *trained* and can *predict*.



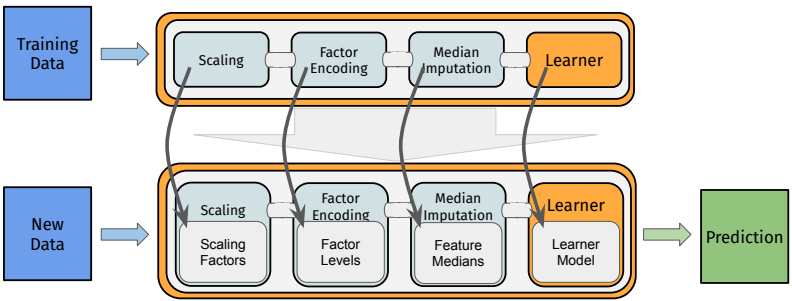
Transformed data can be

- a transformed version of the train/test data for preprocessing nodes, or
- predictions for learner nodes.

Linear Pipelines

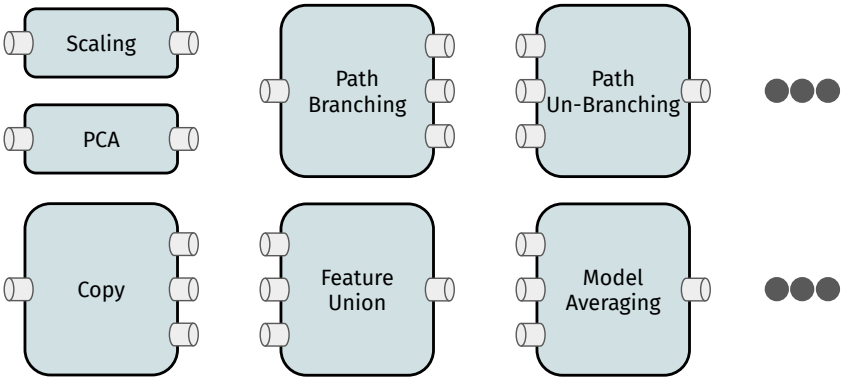
A linear pipeline behaves just as a machine learning algorithm.

- Can be evaluated with resampling and ensures that preprocessing does not cause data leakage.
- Hyperparameters $\mathbf{\Lambda} = \mathbf{\Lambda}_{\text{preproc}} \times \mathbf{\Lambda}_{\mathcal{I}}$ can be optimized jointly.



Nodes with Multiple Inputs or Outputs

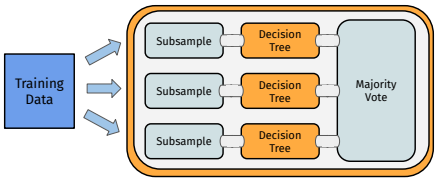
Nodes are not restricted to single inputs and single outputs.



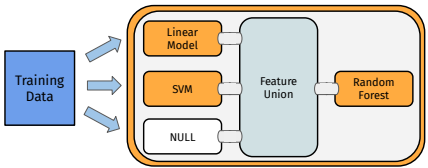
Ensemble Algorithms

This view allows easy representation of different ensemble algorithms:

Bagging:

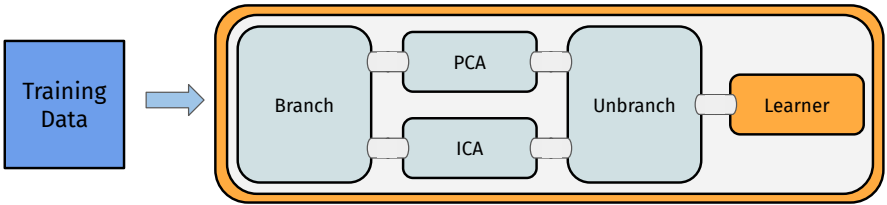


Stacking:

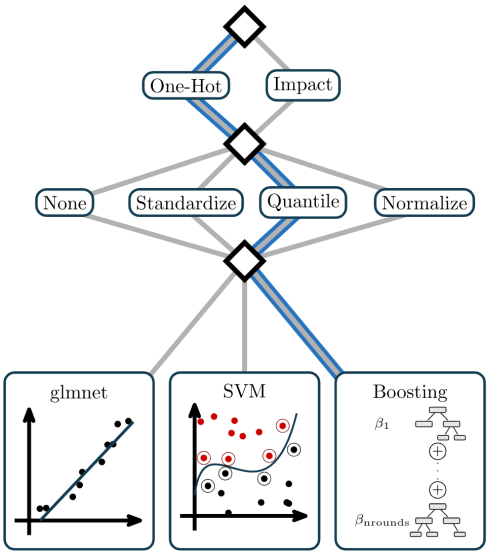


Branching

To represent the search space of an AutoML system there needs to be a branching node with a selection hyperparameter $\lambda_{\text{branch}} = (choice_1, ..., choice_k)$.



Example of a simple AutoML system



Search Space $\tilde{\Lambda}$

Name	Type	Bounds/Values	Trafo
◇ `encoding`	C	One-Hot, Impact	
◇ `scaling`	C	None, Standardize, Normalize, Quantile Transformation	
◇ `learner`	C	glmnet, SVM, Boosting	
`if learner = glmnet`			
`s`	R	$[-12, 12]$	2^x
`alpha`	R	$[0, 1]$	–
`if learner = SVM`			
`cost`	R	$[-12, 12]$	2^x
`gamma`	R	$[-12, 12]$	2^x
`if learner = Boosting`			
`eta`	R	$[-4, 0]$	10^x
`nrounds`	I	$\{1, \ldots, 5000\}$	–
`max_depth`	I	$\{1, \ldots, 20\}$	–

Pipeline Systems for Machine Learning Frameworks

Different frameworks to define and control such pipelines exist for most common programming languages:

- `scikit-learn` with `Pipeline`, `FeatureUnion` and `ColumnTransformer` classes for python.
- `mlr3` with `mlr3pipelines` extension for R.
- `ML.Net` for C#.
- `tfx` for tensorflow.
- `AutoMLPipeline` for Julia.
- ...

Each framework has slightly different features and limitations.