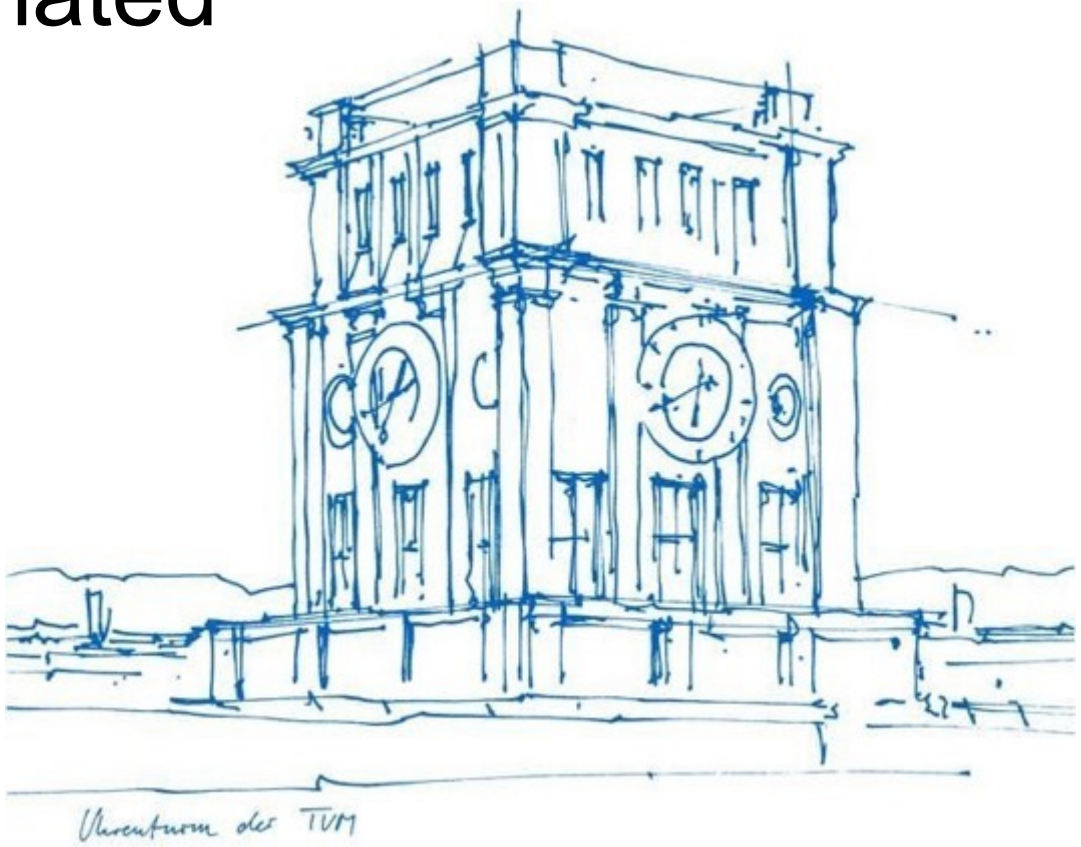
TUM Data Innovation Lab
Benchmarking Matrix for Automated Machine Learning

Björn Magnus Eschment, Aleksandr Zuev,
Natyra Bajraktari, Dominik Eichhorn

Technical University Munich

Munich, 22.07.2021



Team

Dominik Eichhorn

TUM Management & Technology

Natyra Bajraktari

TUM Data Engineering and Analytics

Aleksandr Zuev

TUM Data Engineering and Analytics

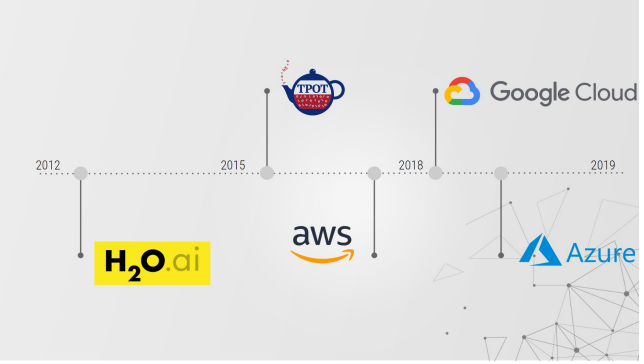
Magnus Eschment

TUM Management & Technology

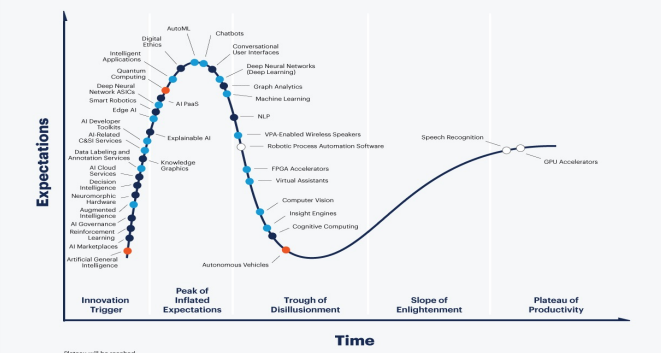
Introduction

What are future outlooks for Automated Machine Learning?

Market entry of big tech companies[1]



Gartner Hype Cycle[2]



With its high potential AutoML is expected to have a Compound annual growth rate (CAGR) of 43% from 2020 to 2030[3].

Our Definition of AutoML

Automated Machine Learning is the automation of the work of a Data Scientist which in the end creates a ML Pipeline.

Introduction

General information about the partner

- Global consulting and accounting firm
- Operating in different industries
- Over 280,000 employees

Motivation of the project

- Challenge the application of AutoML to real life problems
- Replaceability of Data Scientists
- Faster, simpler and cost efficient solution compared to conventional approach

Introduction

Project goals by PwC to fulfill the requirements

1		Research on existing tools
2		Selection of top four AutoML tools
3		Apply all four tools to an open-source data set from the domain of manufacturing
4		Develop a benchmarking matrix
		Building handcrafted ML pipeline & Comparison

Additional Achievement

Outline

1 Theory about AutoML
2 Data Exploration
3 Imbalanced Data Handling
4 Different AutoML Tools
5 Our Benchmarking Tool
6 Results
7 Conclusion

Theory

What does AutoML do?

AutoML fully automates the process of

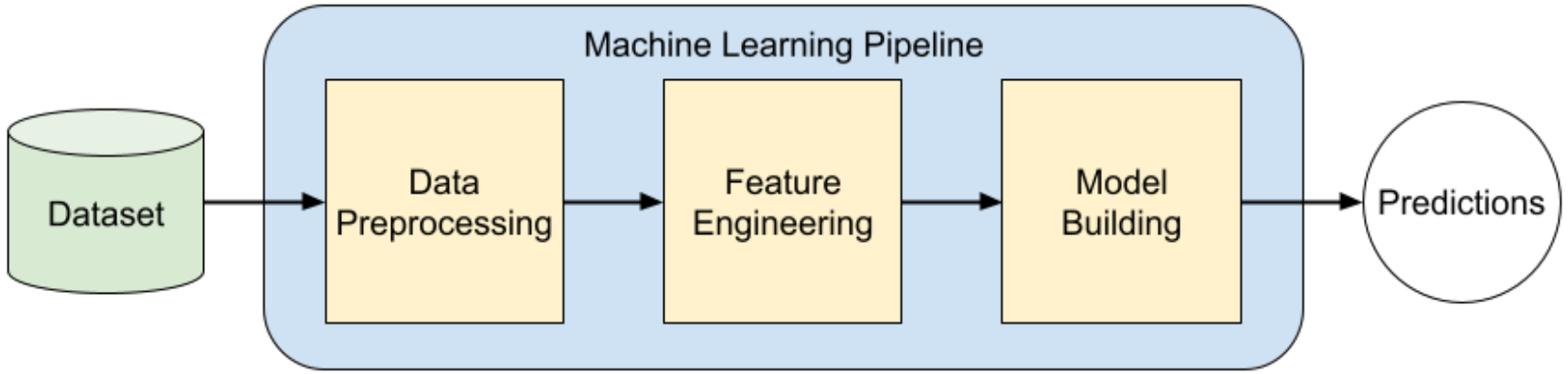
- Building multiple ML pipelines
- Selecting the ML pipeline which performs best w.r.t. to some loss function

Following questions

- What is an ML pipeline?
- How to find the optimal ML pipeline?

Theory

What is an ML pipeline?



	Data Preprocessing	Feature Engineering	Model Building
Set of algorithms to chose from	$\mathcal{A}_d$	$\mathcal{A}_f$	$\mathcal{A}_m$

- Unconfigured ML pipeline $P \in \mathcal{P} = \mathcal{A}_d \times \mathcal{A}_f \times \mathcal{A}_m$
- Configured ML pipeline (P, θ) consists of an unconfigured ML pipeline $P \in \mathcal{P}$ and its hyperparameters $\theta \in \Theta_P$

Theory

How to find the optimal ML pipeline?

Data scientist approach: Sequential

1. Configuration Optimization: Determine the optimal hyperparameters for a set of unconfigured ML pipelines w.r.t. some loss function.
2. Pipeline Selection: Out of these configured ML pipelines, determine the one which minimizes the loss function.

AutoML approach: Combined Pipeline Selection and Configuration

Determine simultaneously the ML pipeline and its corresponding hyperparameters, which minimize the k-fold cross validated loss function.

$$(P^*, \theta^*) \in \underset{\substack{P \in \mathcal{P}, \\ \theta \in \Theta_P}}{\operatorname{argmin}} \frac{1}{k} \sum_{j=1}^{k} \mathcal{L}\left((P, \theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{test}^{(j)}\right)$$

Theory

Which optimization techniques does AutoML use?

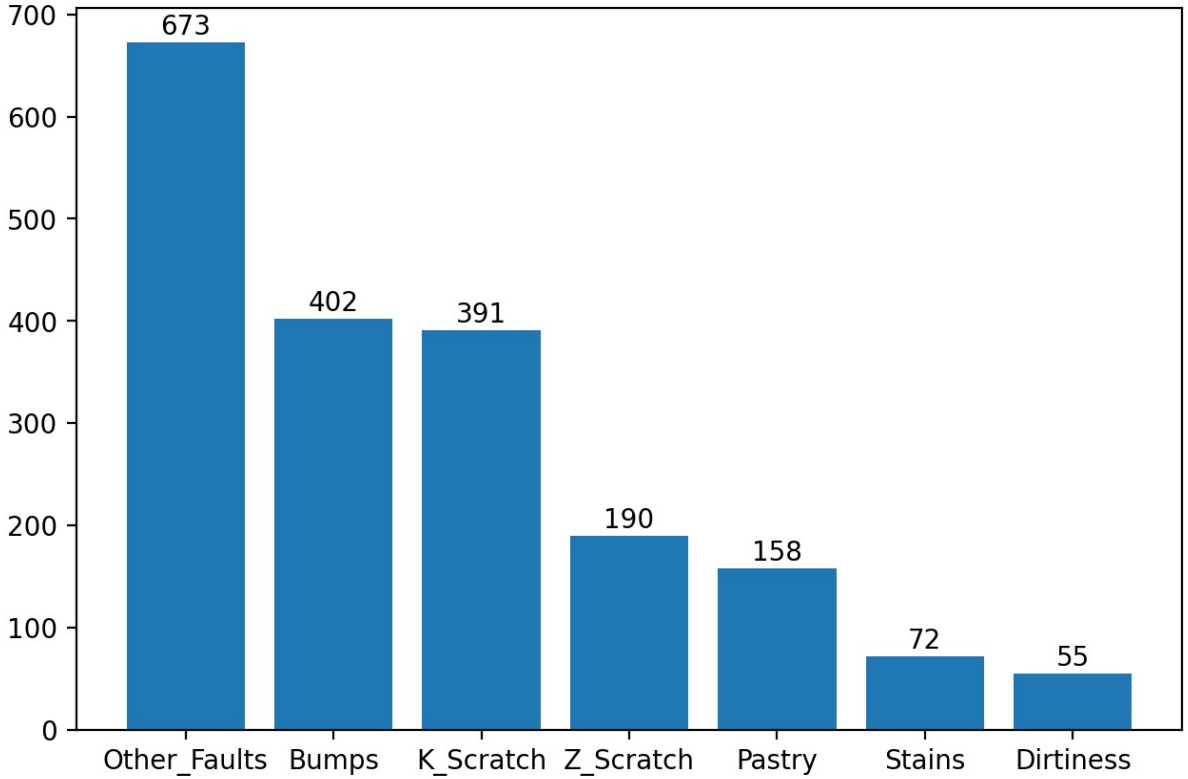
- (Objective function‘s) gradient based optimization techniques not applicable
- Use Black Box Optimization: grid search, random search, Bayesian optimization

Bayesian optimization

- Surrogate model: relationship between hyperparameter values and loss function
- Acquisition function: based on the surrogate model, determines the „information gain“ of evaluating the loss function at a given hyperparameter value
- Procedure:
  1. Evaluate the loss function at a hyperparameter value
  2. Fit/update the surrogate model
  3. Update the acquisition function
  4. Using the acquisition function, determine the next hyperparameter value for loss function evaluation and start over

Data Exploration

General information on the data set



Dataset selection

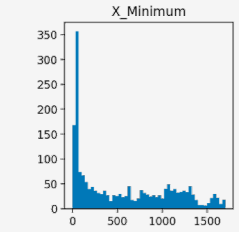
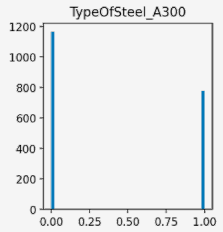
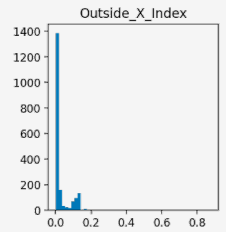
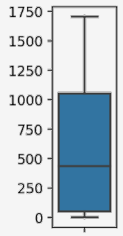
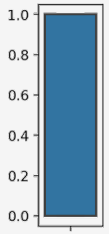
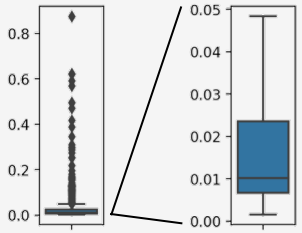
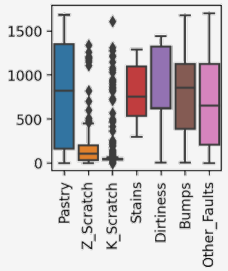
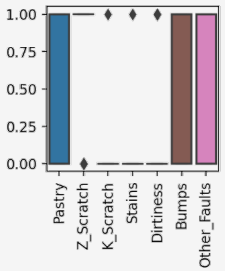
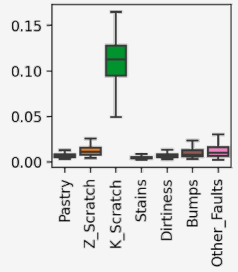
- Supervised learning
- Not a time series data
- Open source
- Manufacturing context

Steel Plates Faults dataset

- University of California Irvine repository
- 27 features
- 1941 observations
- Multi-class classification
- Imbalanced class distribution

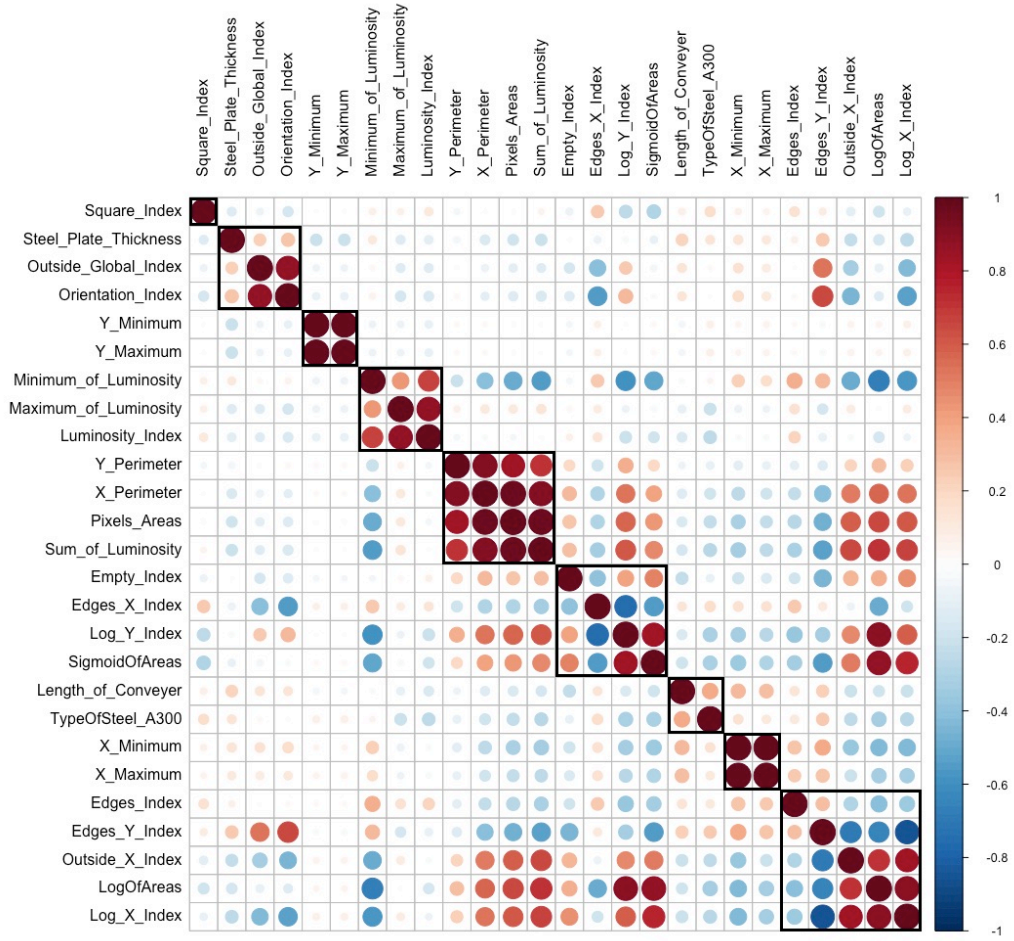
Data Exploration

Distribution of specific variables

X_Minimum	TypeOfSteel_A300	Outside_X_Index	
			Mixed features (numerical and categorical)
			Different range of values Outliers existing
			Different degree of informativeness

Data Exploration

Further analysis of variables

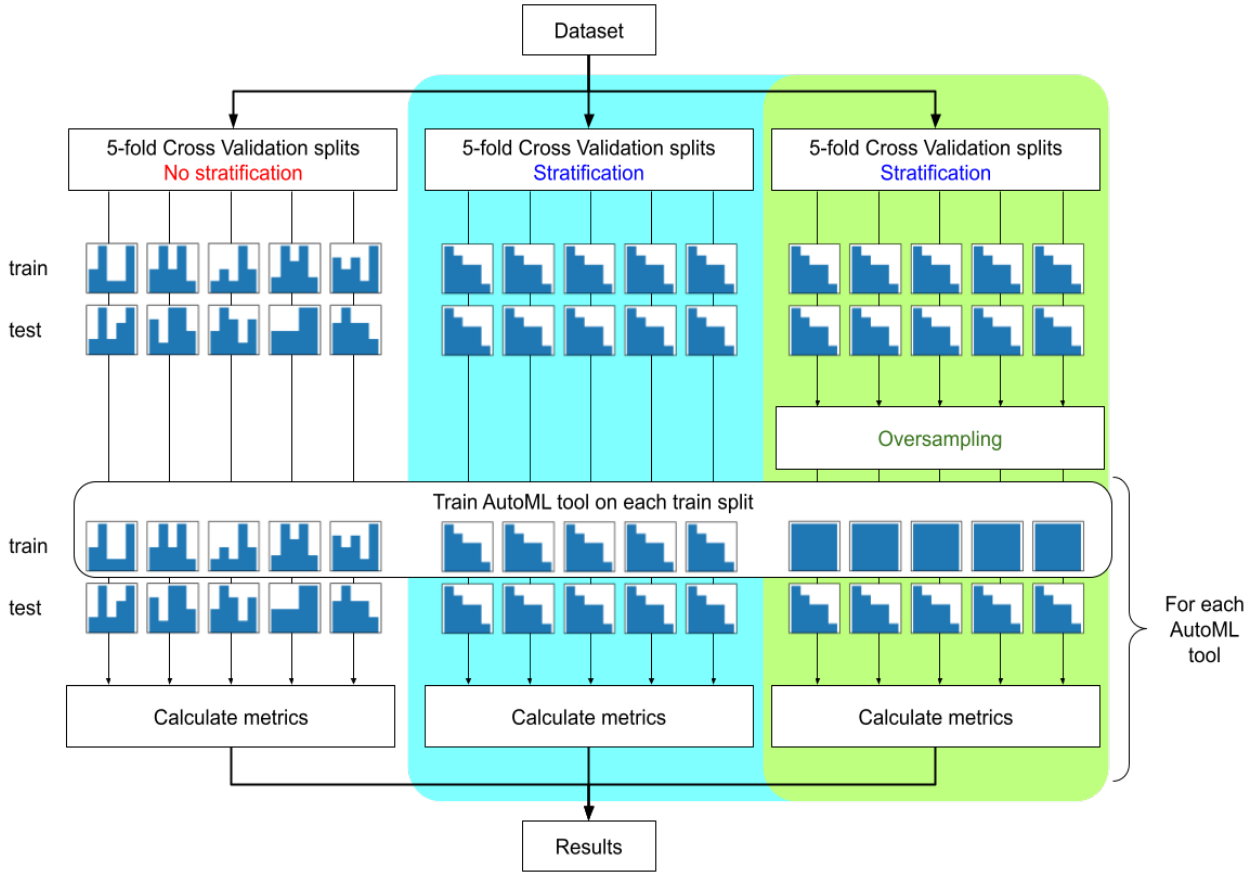


Top-9 hierarchical clusters of features

Absolute correlation coefficient is used as distance

Imbalanced Data Handling

Which strategies can be applied for imbalanced data?



* the histograms are just an example for illustration, not the real data histograms

Stratified 5-fold Cross Validation

Random Oversampling

Three settings:

1. No stratification – no oversampling (snon)
2. Stratification – no oversampling (syon)
3. Stratification – oversampling (syoy)

Performance Metrics

What other challenges do the imbalanced datasets bring?

⚠ Imbalanced Dataset Issue

More prediction errors on the minor (less frequent) classes

🔽 Standard classification metrics e.g **Accuracy** cannot be interpreted properly

High Accuracy achieved only by predicting the major class

Performance Metrics Cont.

Which metric is reported in our experiments?

Metric	Formula	Reasons
Balanced Accuracy	$\frac{\sum_{i=1}^{N} Recall_{class_i}}{N}$	• Same importance to each class (no domain knowledge) • Understand how well the model find samples on average
*Recall	$\frac{TP}{TP+FN}$	• how well the positive class is predicted, out of all actual positive class samples.

AutoML Tools

Which AutoML tools were taken into account?

Initial list of 23 researched tools

Final list of four tools

ALTERYX	Auto-Sklearn	Azure	H2O AutoML
• Commercial toolkit: automatic ML pipelines creation for classification / regression, best model selection	• Open source toolkit: automatic feature engineering, hyperparameter tuning, best model selection	• Commercial toolkit: automatic ML pipelines creation for classification/ regression, time series, best model selection	• Open source toolkit: automatic data preprocessing, training/tunning several models, best model selection

Benchmarking Aspects

How to evaluate the tools?

Tool	Data preprocessing					Feature engineering	
	Missing value imputation	Error/outlier detection	Feature scaling	Imbalanced data handling	Data encoding	Feature generation	Dimensionality reduction
Alteryx	✓	✗	✗	-	✓	✗	✗
Auto-sklearn	✓	✗	✓	✓	✓	✓	✓
Azure	✓	✗	✓	✓	✓	✓	✓
H_2O	✓	✗	✗	✓	✓	✗	✓

Tool	Task			# ML models	Optimization	Ensemble	Model export
	Classification	Regression	Time-series				
Alteryx	✓	✓	✗	4	-	✗	✗
Auto-sklearn	✓	✓	✗	29	Bayesian	✓	✓
Azure	✓	✓	✓	14	Bayesian	✓	✓
H_2O	✓	✓	(✓)	10	Grid search	✓	✓

Tables 1, 2: Comparison of different views among AutoML tools

Benchmarking tool

Input of data into the benchmarking tool

Licence	GUI, i.e. code-...	Feature engineering (automatically) implemented (e.g. ...	Time-series modelling pos...	Variety of implemented m...
Commercial	No	No	No	High
Open Source	Optional	Optional	Optional	Low
	Yes	Yes	Yes	Medium

	General					AutoML Functionality								
Tool	Licence	GUI	Support	Specific OS Requirements	Operating in	Accepted Data Sources: spreadsheet (excel,	Data preprocessing (automatically) implemented	Feature engineering (automatically) implemented	Type of Learning	Learning Tasks	Time-series modelling possible	Variety of implemented models	Saving best model possible	Exporting best model to python possible
alteryx	Commercial	Yes	Yes, within 2 days	only on Windows	Locally installed Alteryx App	Both, spreadsheet and data lake	Yes	No	Supervised	Classification, Regression	No	Low	Yes	No
auto-sklearn	Open Source	No	No	only on Linux	Python	Both, spreadsheet and data lake via SQLAlchemy	Yes	Yes	Supervised	Classification, Regression	No	High	Yes	Yes
azure	Commercial	Yes	Yes, within 2 days	None	Azure Cloud	Both, spreadsheet and data lake	Yes	Yes	Supervised	Classification, Regression, Time Series	Yes	Medium	Yes	Yes
h2o	Open Source	Optional	No	for GUI: Java sdk has to be installed	Python, R	Both, spreadsheet and data lake via SQLAlchemy	Yes	Optional	Supervised	Classification, Regression, Time Series (Optional)	Optional	Medium	Yes	Yes

Benchmarking tool

Output from the benchmarking tool

Licence	GUI, i.e. code-...	Feature engineering (automatically) implemented (e.g. ...	Time-series modelling pos...	Variety of implemented m...
Open Source	No	Optional	Optional	Medium
Commercial	Optional	No	No	High
	Yes	Yes	Yes	Low

	General						AutoML Functionality								
Tool	Licence	GUI	Support	Specific OS Requirements	Operating in	Accepted Data Sources: spreadsheet (excel,	Data preprocessing (automatically) implemented	Feature engineering (automatically) implemented	Type of Learning	Learning Tasks	Time-series modelling possible	Variety of implemented models	Saving best model possible	Exporting best model to python possible
h2o	Open Source	Optional	No	for GUI: Java sdk has to be installed	Python, R	Both, spreadsheet and data lake via SQLAlchemy	Yes	Optional	Supervised	Classification, Regression, Time Series (Optional)	Optional	Medium	Yes	Yes

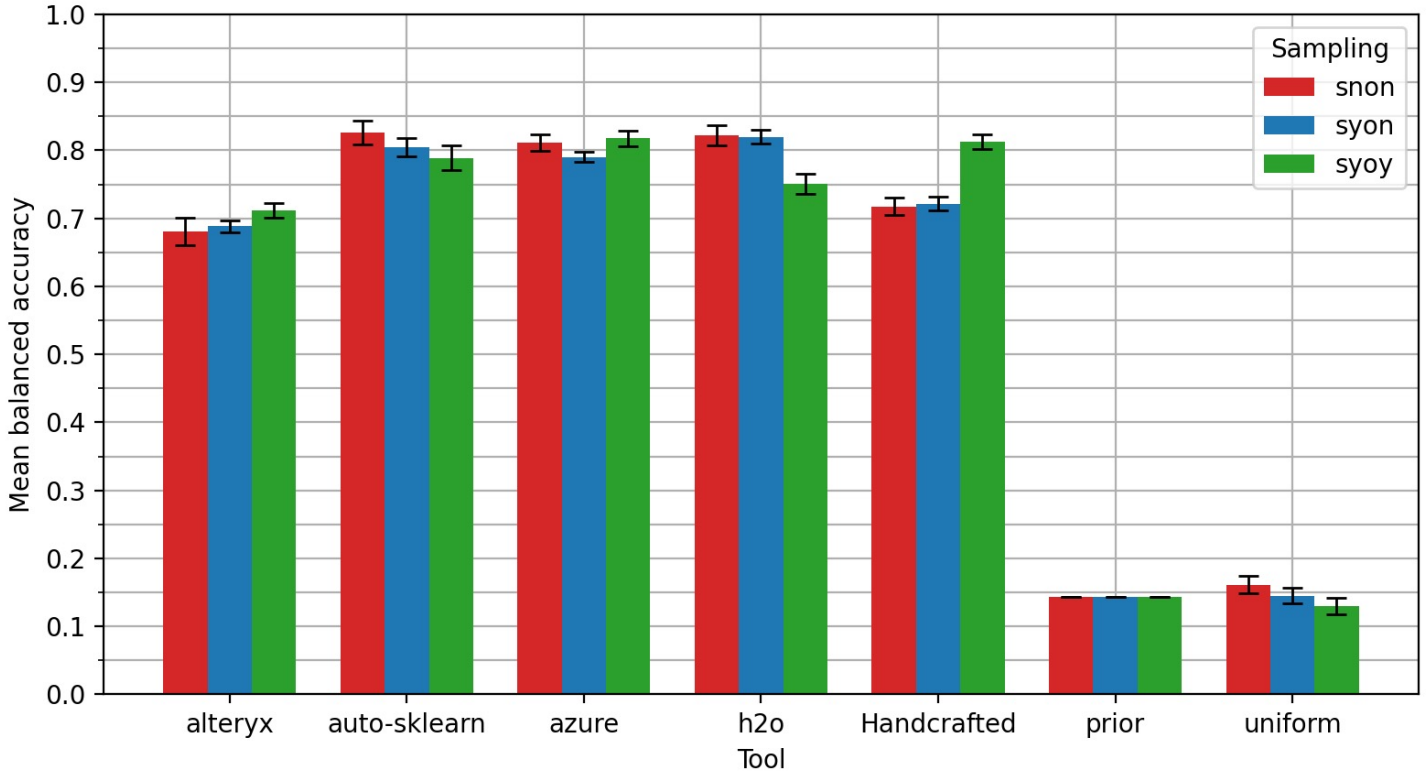
Handcrafted ML Pipeline

Building our handcrafted ML pipeline

Insights from Exploratory Data Analysis	What we did
Variables have different ranges	Standardization of numerical variables
Numerical and categorical variables	Factor Analysis of Mixed Data
Groups of highly correlated variables	
	SVM with Gaussian RBF kernel

Results

Mean balanced accuracy across 5 folds



Handcrafted ML pipeline

Comparing results of **syon** vs **syoy**: **Handcrafted without** vs **with imbalanced data handling**

AutoML

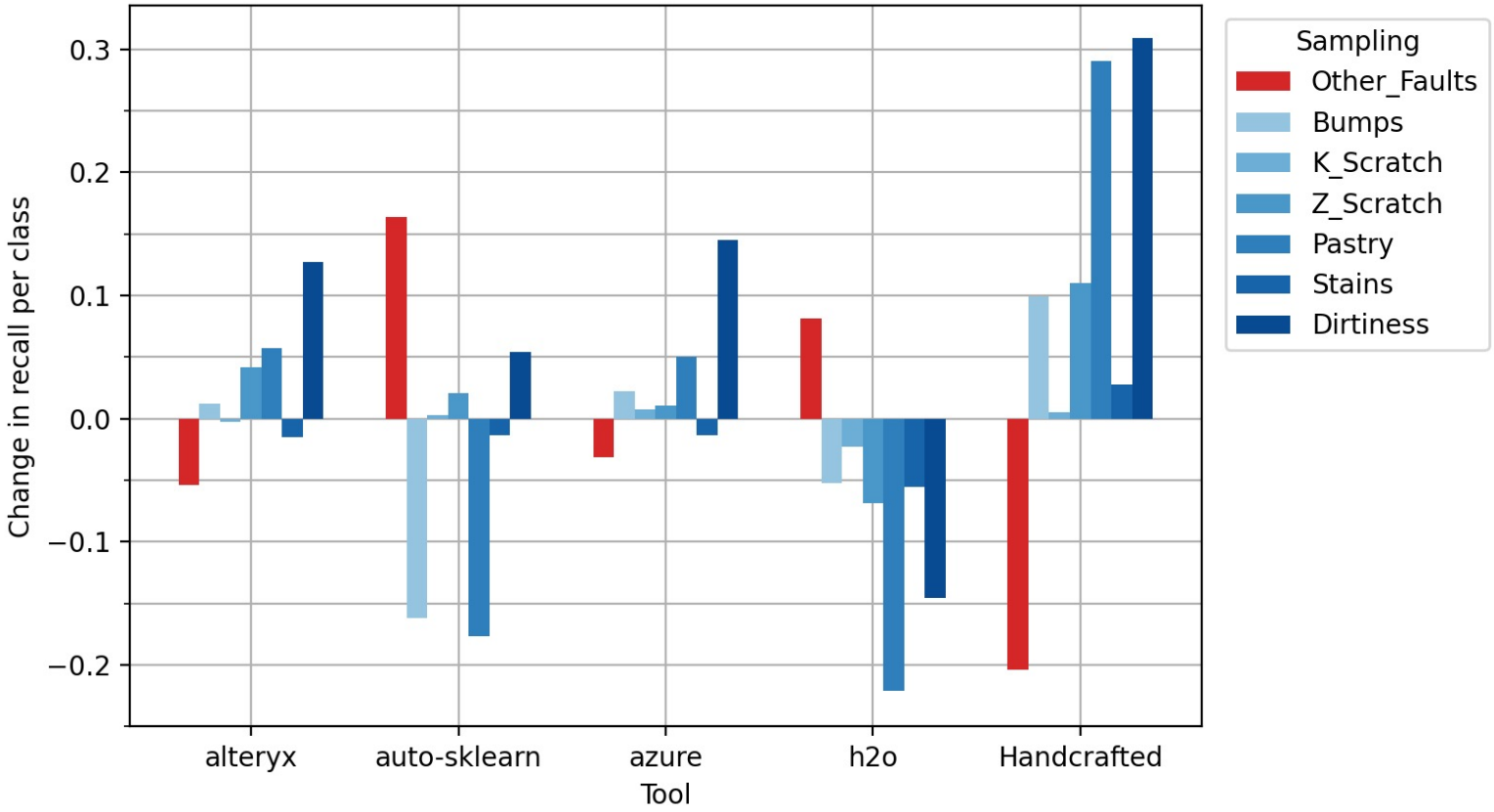
Comparing results of **syon** vs **syoy: Internal AutoML imbalanced data handling** vs **Manual imbalanced data handling**

Results

Change from syon to syoy in recall per class

Impact of Random Oversampling before vs after

Recall of the *majority* class *decreases*,
The recall of the *minor* class *increases*
and vice-versa



Conclusion

Limitations

Reduced generalizability because **only classification** task was done and **only four tools** were used

Learnings

Similar performance except Alteryx

Similar performance with human Data Scientist solution only with data handling beforehand

Limited potential for AutoML especially for unsupervised learning and for time series data

No full potential of AutoML due to missing domain knowledge

Recommendations

Combination of AutoML and domain knowledge from Data Scientist

Overall & functionality: Auto-sklearn

User friendliness: Azure

Thank you for listening and the opportunity to work on this project!

References

References

PricewaterhouseCoopers - Künstliche Intelligenz – die zehn wichtigsten Technologietrends 2018. URL: https://pwc.to/3B9Zu9y (visited on 07/15/2021).

Gartner - Top Trends on the Gartner Hype Cycle for Artificial Intelligence, 2019. URL: https://www.gartner.com/smarterwithgartner/top-trends-on-the-gartner-hype-cycle-for-artificial-intelligence-2019/ (visited on 07/15/2021).

PricewaterhouseCoopers - About us. URL: https://www.pwc.de/en/about-us.html (visited on 07/13/2021).

Marc-André Zöller and Marco F Huber. "Benchmark and survey of automated machine learning frameworks". In: *Journal of Artificial Intelligence Research* 70 (2021), pp. 409–472.

Radwa Elshawi, Mohamed Maher, and Sherif Sakr. "Automated machine learning: State-of-the-art and open challenges". In: *arXiv preprint arXiv:1906.02287* (2019).

Giannis Poulakis. "Unsupervised AutoML: a study on automated machine learning in the context of clustering". MA thesis. Πανεπιστήμιο Πειραιώς, 2020.

Ahmed Alaa and Mihaela Schaar. "Autoprognosis: Automated clinical prognostic modeling via bayesian optimization with structured kernel learning". In: *International conference on machine learning*. PMLR. 2018, pp. 139–148.

Yi-Wei Chen, Qingquan Song, and Xia Hu. "Techniques for automated machine learning". In: *ACM SIGKDD Explorations Newsletter* 22.2 (2021), pp. 35–50.

Max Kuhn and Kjell Johnson. *Applied predictive modeling*. Springer, 2016. Chap. 16, pp. 419–429.

Jason Brownlee. *Tour of Evaluation Metrics for Imbalanced Classification*. Apr. 2021. URL: https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/ (visited on 07/13/2021).

Margherita Grandini, Enrico Bagli, and Giorgio Visani. *Metrics for Multi-Class Classification: an Overview*. Aug. 2020. URL: https://arxiv.org/abs/2008.05756 (visited on 07/13/2021).

Hossin M and Sulaiman M.n. "A Review on Evaluation Metrics for Data Classification Evaluations". In: *International Journal of Data Mining Knowledge Management Process* 5.2 (2015), 01â€“5. DOI: 10.5121/ijdkp.2015.5201.

Shalabh. *Lecture Notes 4 : Stratified Sampling*. URL: http://home.iitk.ac.in/~shalab/sampling/chapter4-sampling-stratified-sampling.pdf.

D. Opitz and R. Maclin. "Popular Ensemble Methods: An Empirical Study". In: *Journal of Artificial Intelligence Research* 11 (1999), 169â€“198. DOI: 10.1613/jair.614.

Alteryx Intelligence Suite. URL: https://www.alteryx.com/de/products/alteryx-platform/intelligence-suite (visited on 07/13/2021).

Alteryx Product Pricing. URL: https://www.alteryx.com/de/products/platform-details/pricing (visited on 07/13/2021).

Alteryx AutoML. URL: https://help.alteryx.com/20212/designer/automl (visited on 07/13/2021).

Matthias Feurer et al. "Efficient and Robust Automated Machine Learning". In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes et al. Curran Associates, Inc., 2015, pp. 2962–2970. URL: http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf.

Matthias Feurer et al. "Auto-Sklearn 2.0". In: *arXiv:2007.04074 [cs.LG]* (2020).

auto-sklearn — AutoSklearn 0.12.6 documentation. URL: https://automl.github.io/auto-sklearn/master/index.html (visited on 07/07/2021).

Microsoft Documentation - Azure Machine Learning. URL: https://docs.microsoft.com/de-de/azure/machine-learning/concept-automated-ml (visited on 07/13/2021).

Azure Machine Learning Pricing. URL: (https://azure.microsoft.com/de-de/pricing/details/machine-learning/) (visited on 07/13/2021).

References

References

H_2O Overview. URL: https://h2o-release.s3.amazonaws.com/h2o/rel-xu/5/docs-website/h2o-docs/index.html (visited on 07/13/2021).

Steel Plates Faults Data Set. July 2021. URL: http://archive.ics.uci.edu/ml/datasets/steel+plates+faults (visited on 07/07/2021).

Marie Chavent et al. "Multivariate analysis of mixed data: The R Package PCAmixdata". In: *arXiv preprint arXiv:1411.4911* (2014).

David Meyer and FH Technikum Wien. "Support vector machines". In: *The Interface to libsvm in package e1071* 28 (2015).

Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization." In: *Journal of machine learning research* 13.2 (2012).

Links to sources in the presentation

[1] https://datasolut.com/automl/
[2] https://www.gartner.de/de/artikel/5-trends-bestimmen-den-gartner-hype-cycle-for-emerging-technologies-2020
[3] https://www.globenewswire.com/news-release/2020/02/11/1982792/0/en/Automated-Machine-Learning-Market-is-Forecasted-to-Post-14-511-9-Million-by-2030-P-S-Intelligence.html

Appendix

Theory

What is an ML pipeline?

- Sets of data preprocessing (d), feature engineering (f) and model building algorithms (m):
  - $\mathcal{A}_d = \left\{A_d^{(1)}, \dots, A_d^{(n_d)}\right\}$
  - $\mathcal{A}_f = \left\{A_f^{(1)}, \dots, A_f^{(n_f)}\right\}$
  - $\mathcal{A}_m = \left\{A_m^{(1)}, \dots, A_m^{(n_m)}\right\}$
- Each algorithm $A_v^{(r)}$, the r-th algorithm in $\mathcal{A}_v$, $v \in \{d, f, m\}$, can be configured by (algorithm) hyperparameters λ_v^r from the domain $\Lambda_{A_v^{(r)}}$
- Unconfigured ML pipeline $P \in \mathcal{P} = \mathcal{A}_d \times \mathcal{A}_f \times \mathcal{A}_m$
- For an unconfigured ML pipeline $P = \left(A_d^{(r)}, A_f^{(s)}, A_m^{(t)}\right)$, the corresponding hyperparameter domain Θ_P is given by $\Theta_P = \Lambda_{A_d^{(r)}} \times \Lambda_{A_f^{(s)}} \times \Lambda_{A_m^{(t)}}$
- Configured ML pipeline (P, θ) consists of an unconfigured ML pipeline $P \in \mathcal{P}$ and its hyperparameters $\theta \in \Theta_P$

Theory

k-fold cross-validation

- Feature space X, space of the target variable Y
- Dataset $\mathcal{D} = \{(\vec{x}_i, y_i) | i = 1, \dots, n, \vec{x}_i \in X, y_i \in Y\}$
- Loss function $\mathcal{L}$
- Given a dataset $\mathcal{D}$ and a Loss function $\mathcal{L}$, the performance of a configured ML pipeline (P, θ) can be evaluated using k-fold cross-validation as follows:
  - Partition $\mathcal{D}$ into k equal-sized folds, $\mathcal{D}_{valid}^{(1)}, \dots, \mathcal{D}_{valid}^{(k)}$
  - Then set $\mathcal{D}_{train}^{(j)} = \mathcal{D} \backslash \mathcal{D}_{valid}^{(j)}, j = 1, \dots, k$
  - For each $1, \dots, k$ compute $\mathcal{L}\left((P, \theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right)$, the value of the loss function achieved by the configured ML pipeline (P, θ), when trained on $\mathcal{D}_{train}^{(j)}$ and evaluated on $\mathcal{D}_{valid}^{(j)}$
  - Finally, take the arithmetic mean of $\mathcal{L}\left((P, \theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right)$ over all k folds
  - Thus, the k-fold cross-validated empirical loss of a configured ML pipeline (P, θ) is given by:

$$\frac{1}{k} \sum_{j=1}^{k} \mathcal{L}\left((P, \theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right)$$

Theory

Pipeline Selection

Given a set of configured ML pipelines $\mathcal{P}_\theta = \{(P,\theta)^{(1)}, \ldots, (P,\theta)^{(p)}\}$ and a dataset $\mathcal{D}$, determine the optimal configured ML pipeline among $\mathcal{P}_\theta$ in terms of minimal k-fold cross-validated empirical loss. This can be written as:

$$(P,\theta) \in \underset{(P,\theta)\in\mathcal{P}_\theta}{\operatorname{argmin}} \frac{1}{k}\sum_{j=1}^{k} \mathcal{L}\left((P,\theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right).$$

Theory

Configuration Optimization

Given one unconfigured ML pipeline P with corresponding (pipeline) hyperparameter domain Θ_P, determine the optimal (pipeline) hyperparameter value in terms of minimal k-fold cross-validated empirical loss. This can be written as:

$$\theta^* \in \underset{\theta \in \Theta_P}{\operatorname{argmin}} \frac{1}{k} \sum_{j=1}^{k} \mathcal{L}\left((P, \theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right).$$

Theory

Pipeline Selection and Configuration Problem

Given the set of all possible unconfigured ML pipelines $\mathcal{P} = \mathcal{A}_d \times \mathcal{A}_f \times \mathcal{A}_m$ and a dataset $\mathcal{D}$, determine simultaneously the pipeline and its corresponding (pipeline) hyperparameters, which minimizes the k-fold cross-validated empirical loss. This can be written as:

$$(P^*, \theta^*) \in \underset{\substack{P \in \mathcal{P}, \\ \theta \in \Theta_P}}{\operatorname{argmin}} \frac{1}{k} \sum_{j=1}^{k} \mathcal{L}\left((P, \theta), \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right).$$

Theory

Pipeline Selection and Configuration Problem (cont'd)

Treat choice which algorithm in each step of the ML pipeline to use as additional categorical meta-hyperparameters $\lambda_d, \lambda_f, \lambda_m$. Let $\lambda_d = r$ denote that the r-th data preprocessing algorithm of $\mathcal{A}_d$ is chosen for the data preprocessing step inside a pipeline (analogously feature engineering (f) and model building (m)). Then the complete hyperparameter space for the pipeline selection and configuration problem Λ, can be written as:

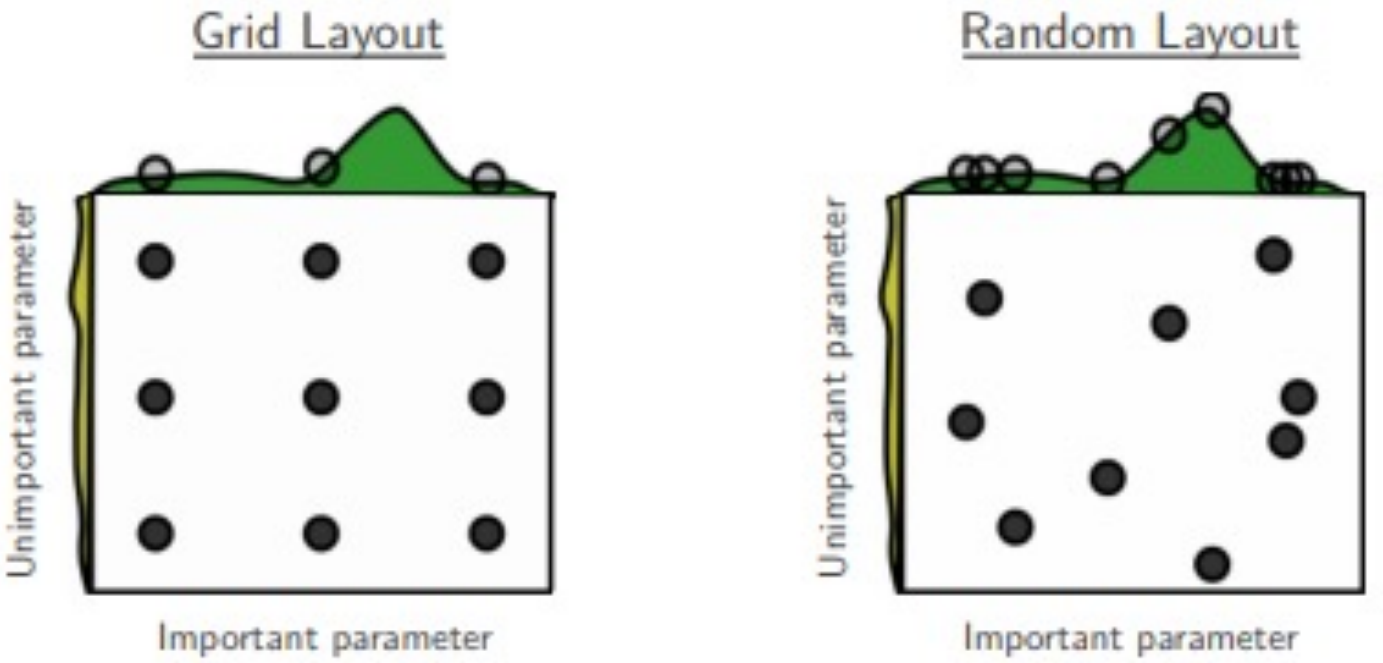
$$\Lambda = \{1, \dots, n_d\} \times \{1, \dots, n_f\} \times \{1, \dots, n_m\} \times \left(\times_{r=1}^{n_d} \Lambda_{A_d^{(r)}} \right) \times \left(\times_{s=1}^{n_f} \Lambda_{A_f^{(s)}} \right) \times \left(\times_{t=1}^{n_m} \Lambda_{A_m^{(t)}} \right).$$

Based on this the pipeline selection and configuration problem can be rewritten as:

$$\lambda^* \in \underset{\lambda \in \Lambda}{\operatorname{argmin}} \ \frac{1}{k} \sum_{j=1}^{k} \mathcal{L}\left(\lambda, \mathcal{D}_{train}^{(j)}, \mathcal{D}_{valid}^{(j)}\right).$$

Theory

Grid search vs. random search



Performance Metrics

Metric	Formula	Description
Precision	$\frac{TP}{TP+FP}$	Represents the fraction of the samples predicted as positive that actually belong to the positive class. It tells us how much we can *trust* the model when it predicts a sample as Positive.
Recall	$\frac{TP}{TP+FN}$	Represents how well the positive class is predicted, out of all actual positive class samples. It tells us how well the model can *find* all the positive samples.
F1-Score	$\frac{2*(Precision*Recall)}{Precision+Recall}$	Looks for a balanced measure between precision and recall and represents the harmonic mean between them.

Results

Set up of the experiment

- 3 settings:
  - snon, syon yield **imbalanced train set**
  - syoy yields **balanced train set**
- Handcrafted ML pipeline: Comparing syon and syoy means to handcrafted without and with imbalanced data handling
- AutoML:
  - If train set imbalanced, AutoML applies internal imbalanced data handling
  - Under syoy (our external imbalanced data handling), the AutoML tools do not apply their internal imbalanced data handling
  - Comparing results of syon and syoy means comparing results with the internal imbalanced data handling of AutoML to our external imbalanced data handling