A Qualitative, Quantitative and User-based Methodology of Automated Machine Learning Systems Evaluation

Iordanis Xanthopoulos

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Professor *Ioannis Tsamardinos*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

University of Crete
Computer Science Department

A Qualitative, Quantitative and User-based Methodology of Automated MachineLearning Systems Evaluation

Thesis submitted by
Iordanis Xanthopoulos
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author: ______________________________
Iordanis Xanthopoulos

Committee approvals: ______________________________
Ioannis Tsamardinos
Professor, Thesis Supervisor

____
Vassilis Christophides
Professor, Committee Member

____
Joaquin Vanschoren
Assistant Professor, Committee Member

Departmental approval: ______________________________
Antonios Argyros
Professor, Director of Graduate Studies

Heraklion, September 2020

A Qualitative, Quantitative and User-based Methodology of Automated Machine Learning Systems Evaluation

Abstract

Automated Machine Learning (AutoML) is a rapidly rising sub-field of Machine Learning. AutoML aims to fully automate the machine learning process end-to-end, democratizing Machine Learning to non-experts and drastically increasing the productivity of expert analysts. So far, most comparisons of AutoML systems focus solely on quantitative criteria such as predictive performance and execution time. In this thesis, we present an multi-level methodology to adequately evaluate such complex systems. We start off by examining AutoML services for predictive modeling tasks from a user's perspective, going beyond predictive performance. We present a wide palette of criteria and dimensions on which to evaluate and compare these services as a user. The comparison indicates the strengths and weaknesses of each service, the needs that it covers, the segment of users that is most appropriate for, and the possibilities for improvements. For our quantitative evaluation methodology, we emphasize on the accuracy of the estimation of predictive performance, as well as a comparison of their hold-out performance. Additionally, we perform an analysis based on the data characteristics of our benchmark and evaluate how they affect the accuracy and quality of the systems' outcome. The results show most systems overestimate their output's performance, while there are no major differences between them when it comes to ranking them based on hold-out performance. In both cases, these results are correlated to the data metafeatures. Lastly, to evaluate the user experience, we create and conduct a custom user study, focusing on the user experience and usability of AutoML systems. In this study, the users are asked to perform a ML analysis using 3 state-of-the-art systems and grade their ease-of-use. Their responses provide useful feedback to the AutoML systems' development teams regarding UX bottlenecks and flawed design decisions.

Contents

1 Introduction **1**

2 Qualitative evaluation **5**

2.1 Introduction 5

2.2 AutoML Services Considered 5

2.3 Qualitative criteria 6

2.3.1 Estimates 6

2.3.2 Scope 7

2.3.3 Productivity 8

2.3.4 Interpretability 8

2.3.5 Customizability 11

2.3.6 Connectivity 12

2.4 Limitations and Discussion 13

2.5 Conclusion 15

3 User study **17**

3.1 Introduction 17

3.2 Included AutoML services 18

3.3 Participants and Location 18

3.4 Introductory presentation 20

3.5 User analysis 20

3.5.1 Custom form 21

3.5.2 Subtasks and corresponding custom form items 21

3.5.3 Comprehension questions 23

3.5.4 System Usability Scale (SUS) 23

3.6 Results 24

3.6.1 Custom form results 24

3.6.2 System Usability Scale form 30

3.6.3 Custom questions responses 36

3.7 Limitations and conclusions 37

4 Methodology of AutoML Quantitative Evaluation 39
4.1 Introduction . . . 39
4.2 Related work . . . 39
4.3 Participating AutoML systems . . . 40
4.4 Evaluation methodology . . . 43
4.4.1 Benchmark datasets selection . . . 43
4.4.2 Evaluation protocol . . . 44
4.4.3 Computing statistical significance . . . 45
4.4.4 Data metafeatures . . . 46
4.5 Experimental evaluation . . . 47
4.5.1 Predictive performance estimation bias . . . 47
4.5.1.1 Binary classification . . . 47
4.5.1.2 Regression . . . 48
4.5.2 Bias correlation to data metafeatures . . . 49
4.5.2.1 Binary classification . . . 49
4.5.2.2 Regression . . . 50
4.5.3 AutoML hold-out performance comparison . . . 51
4.5.4 Hold-out performance correlation to data metafeatures . . . 53
4.6 Limitations and future work . . . 54
4.7 Summary and Conclusion . . . 54

5 Acknowledgments 57

Bibliography 59

6 Appendix 67

List of Tables

2.1 Estimates and Scope criteria. 7

2.2 Productivity and Interpretability criteria. ✜: only for certain models 9

2.3 Customizability and Connectivity criteria. ✧: for RM server, not RM studio 11

4.1 Quantitative evaluation surveys of AutoML systems. **ML tasks**: C: Binary classification, MC: Multiclass classification, R: Regression, **OML**$^{*\mathbf{1}}$: OpenML benchmark, **OML**$^{*\mathbf{2}}$: OpenML100 + OpenMLCC18 + OpenML benchmark, $^{*\mathbf{3}}$: Included both AutoML and AutoDL systems. 40

4.2 Bias results (average and percentage of overestimations) of all AutoML systems across both ML tasks. We additionally present the results corresponding to harder problems ($\leq$ 0.9 hold-out performance). With a * we denote the statistically significant results, with a significance level of 0.05. We observe most AutoML systems significantly overestimate the true performance of their output on average. 47

4.3 Number of AutoML systems successful experiments in our evaluation. Most AutoML systems had an over 90% on both ML tasks. JADBio was the only system with a sub-80% completion rate in one of tasks (regression). 49

4.4 Examining the trends of overestimation with data metafeatures. We performed spearman correlation to detect linear trends. Pvalues are denoted based on different significance levels ($*1$: 0.05, $*2$: 0.01, $*3$: 0.001). JADBio has no trend. Most automl systems show a reduction of their bias when the sample size increases and for smaller samples to features ratio. Additionally, there is a trend regarding the number of missing values and the estimation bias in the regression results. 50

4.5 Examining the trends of performance ranking with dataset metalevel features. We performed spearman correlation to detect linear trends. Pvalues are denoted based on different significance levels (∗1: 0.05, ∗2: 0.01, ∗3: 0.001) Most automl systems show a reduction of their bias when the sample size increases and for smaller samples to features ratio. Additionally, there is a trend regarding the number of missing values and the estimation bias in the regression results. 53

6.1 Selected binary classification datasets for our benchmark. We mention the *Name, ID* and other available metafeatures from OpenML.org 68

6.2 Selected regression datasets for our benchmark. We mention the *Name, ID* and other available metafeatures from OpenML.org . . . 69

6.3 Binary classification experiments results. The train results correspond to the performance estimation after the training phase, while test results correspond to the true hold-out performance. 70

6.4 Regression experiments results. The train results correspond to the performance estimation after the training phase, while test results correspond to the true hold-out performance. 71

List of Figures

3.1 User study overview. The facilitator introduced the users to the study and basic ML terminology with the first 2 steps. Afterwards, each user was called to complete the given analysis and fill out the given forms for each of the participating AutoML systems. 17

3.2 ML knowledge of participants 19

3.3 Background knowledge of participants. The user study group was comprised of mostly ML amateurs and they came from various domains, such as Physics and Biology. 19

3.4 The AutoML pipeline as presented during the introductory presentation. Multiple details had been omitted, as the main goal of this overview was to give an understanding to the users as to how they would proceed with the user study analysis. 20

3.5 Custom form results. The maximum possible score 48. All systems share similar results, with DAI leading and JAD having the lowest score. 25

3.6 SUS form results. The results are explained in 3 different scales, all proposed by the SUS creators and existing users. DAI has a clear edge over the other 2 systems, but all 3 score below the true average (68) of this usability form. 31

4.1 The bias plots for all AutoML systems across all experiments, on both ML tasks. With red we mark the overestimated cases and with black the underestimated ones. The black horizontal line depicts the 0.9 hold-out performance limit.**Top**: The binary classification experiments results for each AutoML system. **Bottom**: The regression experiments results for each AutoML system. Autosklearn and TPOT show the biggest overestimation trends. 42

4.2 The datasets available in the OpenML repository and those we sampled for our benchmark, presented in the Samples/Features grid. **Top Left**: All available binary classification datasets in OpenML repository **Top Right**: All available regression datassets in OpenML repository **Bottom Left**: Our selection for the binary classification task, compared to the union of OpenMLCC18 and OpenML benchmarks, achieves a better coverage of the dimensions grid. both in terms of samples and features. **Bottom Right**: The related study[68] also including this ML task has oversampled datasets from a particular domain (QSAR datasets) and included multiclass classification datasets in its benchmark, therefore reducing its credibility. 44

4.3 Explanation of Spearman method results when used for finding correlations between our evaluation results and data characteristics. . 46

4.4 Classification model quality rankings comparison. **Top**: Scenario A - All participating datasets (Union) **Bottom**: Scenario B - Common completed cases across all systems (Intersection). The systems connected with a horizontal line have no significant differences in their average rankings, at a statistical significance level of 0.05. The CD is the minimum difference of average rankings between systems required to consider their performance significantly distinct. There is no clear winner among the participating systems in the classification ML task. 51

4.5 Regression model quality rankings comparison. **Top**: Scenario A - All participating datasets (Union) **Bottom**: Scenario B - Completed datasets across all systems (Intersection). The systems connected with a horizontal line have no significant differences in their average rankings, at a statistical significance level of 0.05. The CD is the minimum difference of average rankings between systems required to consider their performance significantly distinct. There is no clear winner among the participating systems in the regression ML task. 52

Chapter 1

Introduction

Automated Machine Learning (AutoML) is becoming a separate, independent subfield of Machine Learning, that is rapidly rising in attention, importance, and number of applications [31, 47]. AutoML goals are to completely automate the application of machine learning, statistical modeling, data mining, pattern recognition, and all advanced data analytics techniques. As an end result, AutoML could potentially democratize ML to non-experts (Citizen Data Scientists), boost the productivity of experts, shield against statistical methodological errors, and even surpass manual expert analysis performance (e.g., by using meta-level learning [14]). Finally, AutoML could improve replicability of analyses, sharing of results, and facilitate collaborative analyses.

To clarify the term AutoML, we consider the minimal requirements to be the ability to return (a) a predictive model that can be applied to new data, and (b) an estimate of predictive performance of that model, given a data source, e.g., a 2-dimensional matrix (*tabular data*). Thus, do-it-yourself tools that allow you to graphically construct the analysis pipeline (e.g. Microsoft's Azure ML [43]) are excluded. In addition, we distinguish between *libraries* and *services*. The former require coding and typically offer just the minimal requirements, namely return a model and a performance estimation. AutoML services, on the other hand, include a user interface and strive to democratize ML not only to coders, but to anybody with a computer; they typically offer a much wider range of functionalities.

Algorithmically, AutoML encompasses techniques regarding hyper-parameter optimization (HPO, [66, 3]), algorithm selection (CASH, [30]), automatic synthesis of analysis pipelines [48], performance estimation [74], and meta-level learning [75], to name a few. In addition, an AutoML system could not only automate the modeling process, but also the steps that come before and after. Pre-analysis steps include data integration, data preprocessing, data cleaning, and data engineering (feature construction). Post-analysis steps include interpretation, explanation, and visualization of the analysis process and the output model, model production, model monitoring, and model updating. The ideal AutoML system should only require the human to specify the data source(s), their semantics, and the goal of the analysis to create and maintain a model into production indefinitely.

Given the importance and potential of AutoML, several academic and commercial libraries, as well as services have appeared. The first AutoML system was the academic Gene Expression Model Selector (GEMS) [62]. Recent works formulate the AutoML problem [78, 79], introduce techniques and frameworks for creating new AutoML systems [9, 61], survey the existing ones [59, 79] and comparatively evaluate them [77, 72, 58]. This is a a technically challenging task requiring the availability of a plethora of datasets with different characteristics [18], extensive computational time, ability to set time-limits to all software and many others (see [23] for a discussion on the set up and results of the AutoML Challenge Series).

The problem we want to address is finding a way to properly evaluate and compare AutoML systems. What we need is to evaluate the quality of the user experience on these systems, assess the actual quality and correctness of their results and also grade them based on the functionalities provided to the users. The contribution of this work is 3-fold. Firstly, we present an evaluation methodology covering all the aforementioned dimensions. Moreover, we apply this methodology on multiple AutoML systems and show the results. Lastly, we come to conclusions about the current state of AutoML. We start off by providing a user-centric framework for comparing AutoML services. We define a set of qualitative criteria, spanning across six categories (Estimates, Scope, Productivity, Interpretability, Customizability, and Connectivity) that highlight user-experience beyond predictive performance when selecting or evaluating AutoML services. Using this framework we evaluated seven such services, namely Auger.AI [1], BigML [4], H2O's Driverless AI [24], Darwin [10], Just Add Data Bio [70], RapidMiner [53], and Watson [32]. The comparison is meant to indicate the strengths, weaknesses, scope, and usability of the services, indicating the needs it covers, the tasks it is most appropriate for, and the opportunities for improvement. Our second contribution is the creation of a custom user study to be used in order to evaluate the User Experience (UX) and usability of AutoML services. We propose a simple analysis flow, supplemented by a custom items form and comprehension questions. Additionally, we make use of the System Usability Scale (SUS) form, to identify issues regarding the ease-of-use of AutoML services. We conducted the user study on 19 users of various ML knowledge and coming from different domains, to highlight UX design flaws and general analysis pipeline errors on 3 AutoML services (BigML, H2O's Driverless AI and Just Add Data Bio). This is the first work introducing a complete framework for the qualitative, quantitative and user-based evaluation of AutoML systems. The last part of this work is establishing a methodology in order to evaluate AutoML systems based on the accuracy of their predictive performance estimation, also known as estimation bias, and conduct an out-of-sample performance comparison between our selected systems. Also, we are interested in finding trends of these results with the characteristics of the data we included in our benchmark. We define a set of rules a guidelines for selecting the participating systems and creating our own data benchmark, spanning across 2 ML task outcomes (binary classification and regression). This results in evaluating 5 AutoML systems (Autosklearn [15], GAMA [19], h2o.automl [24], JADBio [70] and TPOT [36]) on 200 datasets.

Chapter 2

Qualitative evaluation

2.1 Introduction

To properly evaluate the available functionalities of AutoML systems, we propose a collection of qualitative criteria. We have included 7 AutoML commercial systems and evaluated them based on their free trials and the documentation provided. To perform this evaluation we have created a list of criteria spanning across 6 different categories and give specific grading definitions. The results provide information regarding the scope and availability of the participating systems, which can aid users in selecting which of them to use. To this extent, it highlights areas where all systems might excel or have not invested as much.

2.2 AutoML Services Considered

In the present evaluation study we consider seven current AutoML service platforms that offer a free trial version, so we could base it on first-hand experience. All of these services, specialize on tabular data, helping us apply the qualitative criteria on all of them. **It was conducted from 01/12/2019 until 07/12/2019 and we used the live versions of the services at the time.** In alphabetical order, the services are:

- **Auger.AI**[1]: A new service, going live in 2019, Auger.AI boasts to have high accuracy and a well-implemented API to help users run experiments with ease.

- **BigML** [4]: One of the oldest ML services, BigML supports AutoML tasks and offers extended support, a custom programming language and a cloud infrastructure for the user.

- **Darwin** [10]: SparkCognition's new AutoML service, providing the users with convenient tools to speed-up their ML tasks.

- **Driverless AI (DAI) [24]**: One of the most well-known AutoML services, DAI supports various ML tasks and also has advanced interpretability mechanisms.

- **Just Add Data Bio (JADBio) [70]**: JADBio was launched in November 2019 focusing on the analysis of molecular biological data (small-sample, high-dimensional) with emphasis on feature selection.

- **RapidMiner Studio (RM) [53]**: The oldest AutoML service used in our evaluation, RM provides multiple tools to its users and supports user-created components. We are looking into the standard version, not including the available user-created add-ons.

- **IBM's Watson (Watson) [32]**: Watson contains multiple components, but here we focus on the *AutoAI experiment* toolkit[1], being closer to what we define as AutoML service for tabular data.

Due to registration fees, we were not able to include in our benchmark recent services such as Google AutoML Tables[2]. Regarding Data Robot[3], we were not able to obtain the free trial licence advertised on their website.

2.3 Qualitative criteria

To qualitatively evaluate the seven AutoML services, we present *32* user-centric qualitative criteria spanning across six different categories. The criteria are partitioned in the following categories. The *Estimates* category is concerned with metrics and estimates' properties about the predictive power of the final model. The *Scope* criteria describe the applicability scope of a service mainly in terms of data types and ML predictive tasks. The *Productivity* category is concerned with the ease of use, while *Interpretability* is concerned with the ability to interpret the results of the analysis. The last two categories are *Customizability* of the analysis and *Connectivity* of the service. The criteria are graded on a 4-level scale. F(ail) (✗), *C* for fulfilling the basic requirements of the criterion, *B* for providing additional functionalities and *A* for achieving a level that should satisfy most users in our opinion.

2.3.1 Estimates

Criteria for *Estimates* (Table 2.1), concern the wealth and depth of estimated quantities regarding the predictive model. *ROC curves* are a useful visualization for interpreting the performance of a classification model and are widely used by the ML community. We grade with *B* the services that output ROC curves

[1] https://www.ibm.com/cloud/watson-studio/autoai

[2] https://cloud.google.com/automl-tables/

[3] https://www.datarobot.com/

Table 2.1: Estimates and Scope criteria.

	Criteria	Auger.AI	BigML	DAI	Darwin	JADBio	RM	Watson
Estimates	ROC curves	B	B	A	✗	A	B	A
	STD/CI calculation	✗	B	B	✗	A	B	✗
	Label predictions	A	A	A	B	A	A	A
	Label probability estimations	✗	A	A	A	A	A	A
Scope	Outcome types	B	B	B	B	A	B	B
	Predictor types	A	A	A	A	B	A	A
	Clustered data handling	✗	✗	A	✗	A	✗	✗
	Missing values handling	A	A	A	A	A	A	A

(Auger.AI, BigML and RM) and with *A* the ones which also output performance metrics for different points on the curve (DAI, JADBio and Watson). In addition to the out-of-sample estimate of predictive performance, a service should be able to report the uncertainty of this estimation (criterion *STD/CI calculation* in Table 2.1 standing for standard deviation and confidence interval respectively). With *B*, we grade the services that only calculate the STD (BigML, DAI and RM) and with *A* the ones calculating the whole probability distribution of performance and its confidence intervals, a richer piece of information (JADBio). Regarding *Label Predictions* on new data, the services that support either individual samples predictions or batch predictions are graded with *B* (Darwin), and the ones supporting both with *A* (the rest of the services). For binary classification tasks, the services able to generate *Label probability estimations* get an *A* (all services except Auger.AI). Overall, JADBio has a full score on all the criteria, followed by DAI and RM.

2.3.2 Scope

Scope criteria (Table 2.1) cover the range of input data that can be analyzed. When it comes to *Outcome types*, services able to handle binary (classification), multi-class (classification), continuous (regression) and censored time-to-event outcomes (survival analysis) score *A* (JADBio), while the ones not handling survival analyses score *B* (the rest of the services). Regarding *Predictor types*, the services which support all the standard tabular data and also text or time-series data are graded with *A* (all services except for JADBio), while the ones only supporting the former with *B* (JADBio). The term *Clustered data* (not to be confused with clustering of data) in statistics refers to samples that are naturally grouped in clusters (or groups) of samples that may be correlated given the predictors. Examples include matched case-control data in medicine and repeated measurements taken on the same subject or client. With *A*, we grade the services able to handle clustered data (DAI and JADBio). It is important to mention the absence of clustered data and repeated measurements handling from most of the services. Essentially, most services assume independently and identically distributed (i.i.d.) data reducing their scope. Finally, we grade a service's ability to handle missing data with A (all services). In this category, DAI and JADBio lead with the highest score.

2.3.3 Productivity

The *Productivity* criteria (Table 2.2) concern the ease of use and boost of user productivity. We start off with *Data manipulation* functionalities available to prepare and manipulate the input data before analysis. Grade B goes to the services providing the user with custom data partitioning and preprocessing recommendations (DAI and Darwin) and grade A to the services that additionally provide data merging, filtering and sub-sampling (BigML, JADBio, RM, Watson). About *Pipeline automation*, the services where the best model is automatically selected according to pre-specified user preferences (e.g., maximize AUC) score A (DAI, Darwin, JADBio and Watson). The services producing a ranking of all tried models instead and require the user to select the one that satisfies their criteria the best score B (Auger.AI, BigML and RM). On one hand, ranking all the models arguably provides richer information to the user, on the other, it does reduce automation and could confuse the non-expert. So, our grading in this criterion is admittedly subjective. We next grade the ability to *Early stop or pause* an analysis. The services able to do both score A (RM) and in case they have implemented either one but not the other, they score B (the rest of the services). When it comes to *Collaboration features*, we grade a service with A if it has implemented mechanisms to create custom organizations and teams to allow sharing of resources, such as data and analyses (all services except DAI and Darwin). Lastly, about *Documentation and support*, the services providing e-mail support score C (JADBio). If they also deliver extensive documentation to the user, they score B (Auger.AI and Darwin) and when they additionally have direct technical support and user forums, their score is A (BigML, DAI, RM and Watson). In general, Productivity is a category emphasized by all services, making it relatively straightforward to any user to complete an ML analysis.

2.3.4 Interpretability

Interpretability criteria (Table 2.2) is arguably on the most important categories for selecting an AutoML service[45]. The criteria concern (a) Exploring and visualizing the data (*Data visualization*) before conducting the analysis. (b) Monitoring the execution of the analysis progress (*Progress report*). (c) Understanding and interpreting how the final model functions (*Final model interpretation*). A particular means to understanding of results is through *Feature selection*, which deserves its own criterion, along with the available mechanisms for the *Final feature set interpretation.* (d) Understanding and validating the process that took place during the analysis (*Analysis exploration*). Regarding *Data visualizations* prior to the analysis, a service which only provides histograms, scores C (JADBio). If it also implements correlation plots and data heatmaps, its score is B (BigML). The

Table 2.2: Productivity and Interpretability criteria. ✤: only for certain models

	Criteria	Auger.AI	BigML	DAI	Darwin	JADBio	RM	Watson
Productivity	Data manipulation	✗	A	B	B	A	A	A
	Pipeline automation	B	B	A	A	A	B	A
	Early stop or pause	B	✗	B	B	B	A	B
	Collaboration features	A	A	✗	✗	A	A	A
	Documentation and support	B	A	A	B	C	A	A
Interpretability	Data visualization	✗	B	A	✗	C	A	A
	Progress report	A	B	A	C	B	A	A
	Final model interpretation	B	B	A	C	A	B	C
	Feature selection	✗	C	C	✗	A	B	✗
	Final feature set interpretation	C	B	A	C	A	B	C
	Analysis exploration	A✤	B	B	✗	B	A	A

services with more options get *A* (DAI, RM and Watson). During the analysis (*Progress report*), if a service only reports the completion percentage, it gets the grade *C* (Darwin). When it shows additionally a performance estimation of the best model and keeps track of the analysis procedure, its grade is *B* (BigML and JADBio). The highest grade (*A*) goes to the services that also show variable importance rankings, generated models ranking and hardware usage (Auger.AI, DAI, RM and Watson).

Once the analysis is complete, the AutoML service should be able to explain how the final model works. This adds transparency to the model and pinpoints possible flaws or bias in its decision making, making it more trustworthy. The interpretability of the results is a subdomain of ML with increasing popularity and every year multiple new mechanisms are introduced [45, 12]. We have selected a set of such mechanisms and grade the AutoML services based on how many of them they have implemented. The mechanisms are: a) the confusion matrix, which is created based on the predictions made during the training phase, to help the user understand what type of errors are produced by the final model; b) report of the performance of the final model using multiple performance metrics; c) residuals visualization, i.e. the difference between observed and predicted values of the data; d) PCA procedure [60] to highlight strong patterns of the data and visualize them on a 2-D space; e) visualization of the final model, when this is possible; f) techniques to explain the predictions in case of a complex final model (e.g. LIME-SUP [29], K-LIME, a variant of LIME [56], decision tree surrogate models [11], etc.). When the service has implemented at least 2 of the above mechanisms, its corresponding grade is *C* (Darwin and Watson), while for a service with more than 2 available mechanisms, its grade is *B* (Auger.AI, BigML, RM). The grade *A* is reserved for the services with more than 4 of the aforementioned mechanisms implemented (DAI and JADBio).

Feature selection is often the *primary* goal of an analysis. It leads to simpler models that require fewer measurements to provide a prediction, which may be important in several applications. Most importantly however, *feature selection is used as a tool for knowledge discovery* [40] to gain intuition and insight into the problem (hence, its inclusion in the interpretability category). A pharmacologist is not only interested in predicting cancer metastasis but also in the molecules involved in the prediction to identify drug targets; a business person is interested in the quantities that affect customer attrition to devise new promotions and advertisements. Such reasoning is theoretically supported by the fact that feature selection has been connected to the causal mechanisms that generate the data [69]. It is defined as the problem of identifying a *minimal-size* feature subset that *jointly* (multivariately) leads to an *optimal* prediction model (see [22] for a formal definition). Thus, feature selection removes not only irrelevant, but also redundant features. In some data distributions, there may be multiple solutions to the feature selection. For example, due to low sample size the truly best feature subset may be statistically indistinguishable from slightly sub-optimal feature subsets. Or, it could be the case there is informational redundancy that leads to feature subsets that are equally predictive. While all solutions are equivalent in terms of predictive performance, *returning all solutions is important when feature selection is used as a tool for knowledge discovery.*

The services which offer single feature selection functionality, score C (BigML and DAI). BigML treats feature selection as a preprocessing step, before the modeling process and the estimation of performance protocol. This approach is methodologically wrong and leads to overestimating performance (see [26], page 245). There are different notions of multiple feature selection. When a service returns several feature subsets as options, but does not provide any theoretical guarantees of statistical equivalence, its grade is B (RM). On the other hand, when a service returns several feature subsets that lead to models with statistically indistinguishable performance from the optimal, its grade is A (JADBio). Feature selection by itself is not enough. The services should also provide users with mechanisms for interpreting and understanding how each feature in the final set affects and contributes to the decision making of the final model. We base our grading on a set of *Final feature set interpretation* mechanisms and how many of them each AutoML service has implemented. The mechanisms are: a) random forest feature importance ranking of the participating features [7]; b) LOCO feature importance [37]; c) partial dependence plots (PDPs) [16]; d) SHAP plots [41]; e) ICE plots [20]; f) a report of the standardized individual and cumulative importance of the participating features; g) the actual standardized coefficient for each feature, in the case of a linear final model; h) information about the resulted feature sets, in the case of multiple feature selection. A service that has implemented at least 1 of these mechanisms, is graded with C (Auger.AI, Darwin and Watson). If more than 2 mechanisms are available, the service's grade is B (BigML, RM) and the grade A is reserved for the services with 4 or more mechanisms (DAI, JADBio).

Expert analysts would often like to verify the correctness and completeness of the analysis that took place. It is not only the results (model) that should not be treated as a black-box, but also how these results were obtained. A service

Table 2.3: Customizability and Connectivity criteria. ✧: for RM server, not RM studio

	Criteria	Auger.AI	BigML	DAI	Darwin	JADBio	RM	Watson
Customizability	Time budget	B	B	A	A	B	✗	✗
	Resources budget	✗	✗	A	✗	A	✗	B
	Analysis components customization	A	A	A	B	B	A	B
	Enforce Model Interpretability	✗	✗	A	✗	B	✗	✗
	Feature selection options	✗	A	A	✗	A	B	✗
	Visualizations customization	✗	A	B	✗	✗	A	A
Connectivity	Service deployment	✗	A	A	✗	✗	A✧	✗
	3rd party storage connection	A	A	A	✗	✗	A	A
	API access	A	A	A	A	A	A	A
	Downloadable results	A	A	A	✗	B	A	B
	Analysis components contribution	B	A	A	✗	✗	A	B
	Model deployment	A	A	A	A	✗	A	A
	Visualizations exportability	✗	B	B	✗	B	A	A

which displays an *Analysis exploration* graph, to help the users understand the methods used in each step scores *A* (Auger.AI, RM and Watson). If the service displays all pipelines that were tried, in the form of list instead of as a graph, its score is *B* (BigML, DAI and JADBio). When it comes to analysis interpretation, DAI and JADBio seem to be the best choice, providing the user with advanced mechanisms for understanding the final results. Some services, do not provide any information about which analysis pipelines they tried; the analysis process is essentially a black box to the user. We note that in our opinion, there is room for improvement regarding interpretability for most of the services.

2.3.5 Customizability

The *Customizability* category (Table 2.3) grades the ability of the services to customize analysis according to user choices and preferences. About *Time budget*, we grade with *B* the services giving the ability to impose a non-strict time limit on an analysis (Auger.AI, BigML and JADBio) and with *A* the ones which allow setting a strict time limit (DAI and Darwin). Our take on this subject is that every service should give the ability to pose a strict time budget, as an analysis can be part of a bigger project, running under specific time restrictions. Moving to the hardware *Resources budget*, if a service allows the user to select a preset hardware configuration, it scores *B* (Watson) and if it allows setting up the exact hardware specifications, *A* (DAI and JADBio). Next, we consider the *Customization of analysis components*, i.e. the ability to choose the methods and algorithms to try, along with their hyperparameters, in each step of the ML pipeline. If the user is able to fully customize the included components, the service gets *A* (Auger.AI, BigML, DAI and RM). If the service provides the user with a set of limited settings, it gets *B* (Darwin, JADBio and Watson).

A service that allows the user to *Enforce final model interpretability*, is graded with *B* (JADBio) and if it provides additional interpretability settings, with *A* (DAI). Another customization criterion is about the available *Feature selection options*. If the AutoML service allows the user to select the exact number of selected features, it is graded with *A* (BigML, DAI and JADBio) and if it allows the user to set certain parameters, such as the effort put in feature selection, with *B* (RM). Finally, we also consider the *Visualizations customization* options. When a service gives the user the ability to set user-specific thresholds on certain visualizations, its grade is *B* (DAI). If the user can fully customize the resulted visualizations (e.g. changing the axes, titles, legend, colors), the service's grade is *A* (BigML, RM and Watson). In general, when it comes to customizability, DAI has a clear edge over the competition, giving the users options to fine-tune and setup an analysis according to their needs. We distinguish two different schools of thought on this category. On one hand, services such as DAI, let the user fully customize the algorithms and hyperparameter values to search during an analysis. On the other hand, services like JADBio provide the user with a few preference choices that do not require expert knowledge of ML. The first approach empowers an expert analyst but it may be intimidating to the non-expert user. There is a fine line between providing enough choices to an expert to fully customize an analysis and achieve better results and providing too many choices that make the process complex and easy to break. For this reason, we would recommend to equip AutoML services with some kind of warning system that can actually detect when the selected setup might create problems and notify the user accordingly.

2.3.6 Connectivity

The *Connectivity* criteria (Table 2.3) grade the options offered to connect a service with external tools and resources. First, regarding the *Service's deployment* at an external infrastructure, the services supporting it score *A* (BigML, DAI and RM). The ones able to *Connect to 3rd party storage providers* also get an *A* (all except from Darwin and JADBio). Furthermore, all services have implemented their own *API* (grade *A*). We also look into the *Downloadable results* options. In the case where only part of the results are downloadable, the services are graded with *B* (JADBio and Watson) while the ones allowing the user to download all the results and also generate a summary report, with (A) (all services except JADBio and Watson). A user might be interested in *Adding custom components* to the AutoML service. If it is allowed to the user to add components through a service's API, the service is graded with *B* (Auger.AI and Watson). If the service has moreover implemented a complete system for user-defined components, by creating their own marketplace or extensions library, its grade is *A* (BigML, DAI and RM). Creating the best final model does not always suffice, as the user will probably want to deploy it in an external service and use it for new data predictions. Most of the participating services, have added various model deployment options (grade *A*) (all except JADBio). The currently implemented ideas are to use data transfer libraries, e.g. cURL (Auger.AI, Watson), create actionable models (BigML, Darwin, RM) or scoring pipelines (DAI). All of the above provide the same functionality; predicting labels on new unseen data. Finally, when writing reports or papers with the results, the visualizations need to be exported. The services which provide less than 3 export options score B (BigML, DAI and JADBio) and those with more, score A (RM and Watson). Taking a look at the participating services, most of them cover the majority of the proposed criteria. The export formats available for data visualizations are static in all systems, an area that could greatly be improved. Additionally, we find the lack of connections to public repositories, such as OpenML [73] important, as they can be useful to a user who is interested in conducting ML analyses for academic reasons.

2.4 Limitations and Discussion

Admittedly, this work has several limitations. We take the opportunity to discuss some in depth, pointing to important open issues and future work. First of all, we were not able to evaluate every known AutoML service.

Estimates: While all services provide estimated quantities from the data, the major question remains: **are the estimates returned correct and reliable**? Statistical estimations are particularly challenging with low samples; even more so with high dimensional data. Is performance overestimated, standard deviations underestimated, probabilities of individual predictions uncalibrated, feature importance's accurate, or multiple feature subsets returned not statistically equivalent? *Which AutoML services return reliable results one can trust, and which ones are actually misleading the user and potentially harmful*? In case of medical applications, overestimating performance or confidence in a prediction (uncalibrated predicted probabilities) is dangerous and could impact human health, while in business applications it may have significant monetary costs. Such questions require significant experimentation with all services to answer. Experimentation should be performed on datasets with a wide range of characteristics, e.g., sample size, number of features, percentage of missing values, mixture of types of predictors (continuous, discrete, ordinal, zero-inflated, etc.), outcomes, etc. to provide a full quantitative picture of the pros and cons of each service and its correctness properties. Unfortunately, most quantitative evaluations are currently performed on datasets with a limited range of such characteristics or are restricted by time limitations.

Scope: In this work, we are only concerned with predictive modeling (supervised learning) tasks and not other ML categories. Each different task would require a separate set of criteria that applies to it. *We do note, however, that BigML, DAI, RM, and Watson also support clustering, anomaly detection, and some NLP tasks* which are useful to numerous users. A major limitation of our scope grading is that it misses important criteria concerning the maximum volume of data a service can handle in reasonable time or memory resources, both in terms of number of features, samples, or their combination (total volume). Unfortunately, we are not able to test the limits of each service as we are confined to analyses that run on the free trial versions. However, regarding the scalability with respect to feature size, we note that almost all services have difficulty scaling to thousands of features. JADBio on the other hand, was created to scale up to the feature size of typical multi-omics datasets that can reach up to hundreds of thousands of features.

Productivity/Interpretability: Although, we presented a first qualitative assessment, a true measure of productivity increase requires an extensive user study with representative datasets spanning a wide-range of characteristics (in terms of the number of features and samples). In such a user-study, one should measure how much productivity has improved over manual scripting, eventually by trading off learning performance, and how much insight has been gained by the interpretation tools offered by each service. To assess how an AutoML system performs against human experts Kaggle[4] and other ML competitions could be exploited. As data and tasks are specific for a competition problem, solutions by human experts usually take the top positions as they apply domain-specific knowledge and sometimes create custom methods and mechanisms to help them win these competitions. Still, AutoML systems that have been tested on such tasks, achieve comparable performance. AutoML systems are becoming more and more sophisticated, by automating an increasing number of tasks in ML pipelines (e.g., feature engineering), while supporting meta-level learning techniques. This can lead to minimizing the gap between human experts and AutoML in competitive environments [61] and aid in producing high quality ML models for both commercial and academic purposes.

There are several other criteria categories that are missing from the present methodology, due to space limitations. These include *model monitoring and maintenance* that regards functionalities to maintain a model into production [42], such as monitor the health of the production model, raise alarms when there is a drift in the data distribution, automatically re-train and update the model, and others. As ML systems move from computer-science laboratories into the open world, their *accountability* [17] and *auditing* [13] becomes a high priority problem. In this respect, we need a deep understanding of the ML system behavior and its failures. Current evaluation methods such as single-score error metrics and confusion matrices provide aggregate views of system performance that hide important shortcomings. Understanding details about failures is important for finding ways for improvement, communicating the reliability of systems in different settings and for specifying appropriate human oversight and engagement [46].

Finally, we would like to mention that each category could be expanded with many more criteria. Only the criteria that were addressed by at least one of the services were included. Functionalities that were not addressed by any of the services examined are missing. One example is the ability to handle continuous signals and streaming data [52].

[4] https://kaggle.com

2.5 Conclusion

AutoML has made tremendous progress since its first embodiment in the GEMS system. Several AutoML services are already available, routinely analyzing business and scientific data for thousands of users. They do increase productivity and allow non-experts to perform sophisticated ML analyses. Our prediction is that within a few years, most of data analysis will involve the use of an AutoML service or library; scripting as a means to manual ML analysis will gradually become obsolete or pass to the next level, where it is customizing and invoking AutoML functionalities.

The proposed criteria intend to turn the spotlight back onto the human user. Users do not only consider learning performance when choosing a service. They also consider a plethora of other criteria such as the ones presented. One of the most important ones is interpretability of results. Users are rarely satisfied with just a predictive model; they also seek to understand the patterns in their data. Thus, results should not be a black-box, but explained, visualized, and interpreted. Users need to examine the analysis process and ensure its correctness or optimality: AutoML should automate, not obfuscate. The analysis process should be transparent, verifiable, and customizable by the user. Some of the AutoML services examined, clearly abide to these principles but some fail in this set of criteria. Arguably, it is perhaps interpretation of results and ease-of-use that will determine the success of an AutoML service, and not necessarily predictive performance.

Current AutoML systems mostly focus on tabular, iid-sampled data. Obviously however, most of the world's data is not in this format or sampled as iid. Ultimately, AutoML competes with the human expert not only in learning performance but in scope and the range of problems it can handle. There are ongoing efforts to develop AutoML solutions for regression or anomaly detection tasks in time-series, time-course data, and streaming data (e.g., Microsoft Azure [43], Yahoo EGADS [35], Facebook Prophet [64]), or to generate features from relational tables or CSV/JSON files [21]. Future AutoML systems should also automate more data preparation tasks including data cleaning (e.g. error correction and deduplication) [57] and support ML tasks such as reinforcement, transfer and federated learning, or causal modeling [51] to name a few. Still, interpreting the results of the analysis in each category is quite challenging and probably requires a different, specialized set of methods. Works such as this can guide both the users and development teams into creating comprehensive and useful AutoML systems, focused on real user needs.

Chapter 3

User study

3.1 Introduction

The aim of this user study was to evaluate the user experience (UX) and the usability of the candidate AutoML services by users of varying ML knowledge. Therefore, it was a strictly qualitative evaluation, not focusing on AutoML aspects such as predictive performance. Prior to the main part of the study, the users filled a preliminary form providing information about their background. Following, they were presented with a brief presentation regarding what is AutoML, tools and techniques used in ML, and information about the procedure of the user study. Moving to the main part of the user study, each participant was called to complete a pre-designed ML analysis, following specific instructions, for all 3 candidate services, in a 2-hour time frame. The participants were asked to fill out different forms and answer questions during the user study, evaluating their UX, while performing the analysis. An overview of the user study can be found at Fig. 3.1. In the next section, we introduce the AutoML services selected for this user study. Afterwards, we are providing details about the participants of the study. Next, we present the details of the analysis, going through all the subtasks the user was called to complete. Finally, we exhibit the results of our user study and comment on them.

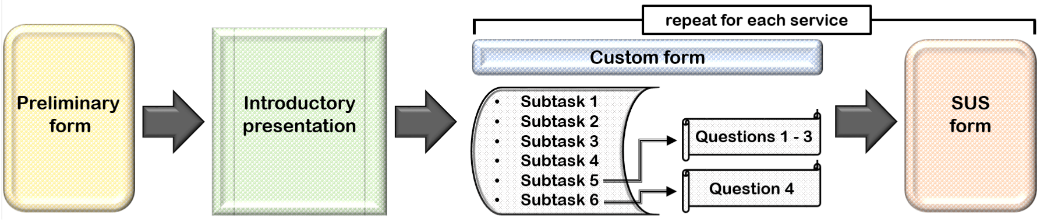


Figure 3.1: User study overview. The facilitator introduced the users to the study and basic ML terminology with the first 2 steps. Afterwards, each user was called to complete the given analysis and fill out the given forms for each of the participating AutoML systems.

3.2 Included AutoML services

We selected 3 AutoML services for this user study, BigML[4], H2O's Driverless AI (DAI)[24] and Just Add Data Bio[70]. We selected these services based on certain criteria. To begin with, these services are among the most complete, when it comes to implemented features, and they share a common view on how an AutoML pipeline should be structured. This makes it easier for a user, who has never used these services before, to operate them and complete all the tasks in the given time frame. Secondly, the services at hand are all available online. This allowed the participants to complete the user study using their own PCs, and eliminated potential installation or compatibility issues. Additionally, it helped us make sure that all candidates would have the same experience with the participating services. Lastly, we were able to get multiple accounts with full unrestricted access to the free-trial versions of these services, which did not have any limitations in terms of available functionalities, compared to the full versions. This provided us with the tools required, to design the ML analysis of the user study as we saw fit, and made it possible to run the user study with multiple users simultaneously.

3.3 Participants and Location

The user study was conducted with 2 groups of participants, in 2 different sites and dates. The first group was located at SAP SE, Walldorf, Germany, where 9 people participated, at the 22nd of August, 2019. The second group was located at the Computer Science department of the University of Crete, Greece, where we had 10 participants, at the 27th of September, 2019. In both sites, the user study was conducted in the same manner, having identical material. It is important to notice that, even though there was a gap of approximately a month between the 2 sessions, we did not need to do any adjustments to our user study design for the second session, guaranteeing the comparability of the results. Our aim when creating the groups was to pose no restrictions, regarding the participants' background or how much experience they had had using machine learning. By doing so, a total of 19 people took part in the user study. By gathering the results of our `Preliminary form`, we notice the participants originate from different scientific domains (Fig. 3.3). Additionally, they were not ML experts, but, on the contrary, most of them were complete amateurs or with little experience in applying machine learning (Fig. 3.2). This is important as the whole point of AutoML services is to make the setup and completion of a ML analysis intuitive and simple to the users, regardless of their prior ML knowledge. Lastly, to secure an unbiased result and no possible conflicts of interest, we made sure none of the participants had affiliations with any of the included AutoML services.

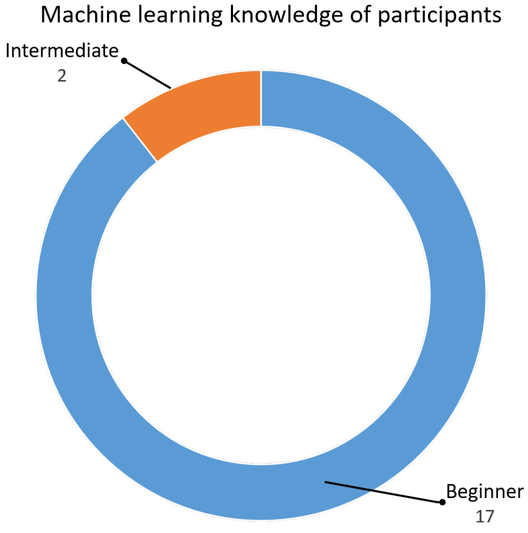


Figure 3.2: ML knowledge of participants

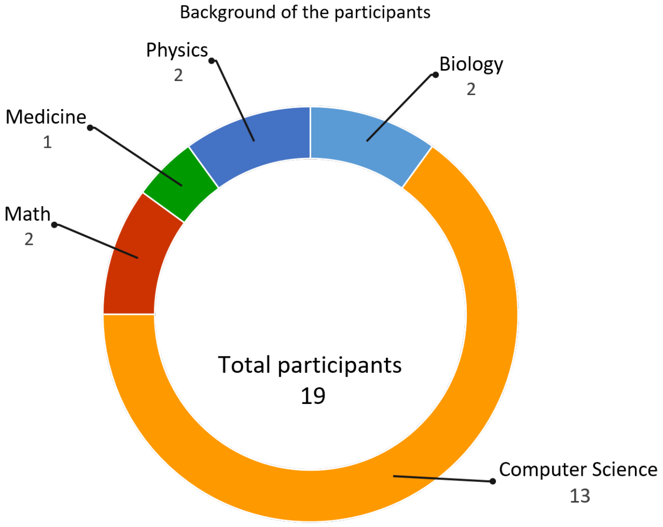


Figure 3.3: Background knowledge of participants. The user study group was comprised of mostly ML amateurs and they came from various domains, such as Physics and Biology.

3.4 Introductory presentation

Before the users started the analyses on the AutoML services, they were given a brief 15-minute introductory presentation. This was done, in order to help even the complete ML amateurs understand basic terminology and how the ML pipeline is represented in an AutoML service 3.4. Additionally, the analysis procedure was presented and explained, together with the 4 questions the users were asked to answer during the analysis procedure. Once the users' questions were answered, we displayed the data they were going to use for their analyses, discussing their format and characteristics to make the procedure even more intuitive.

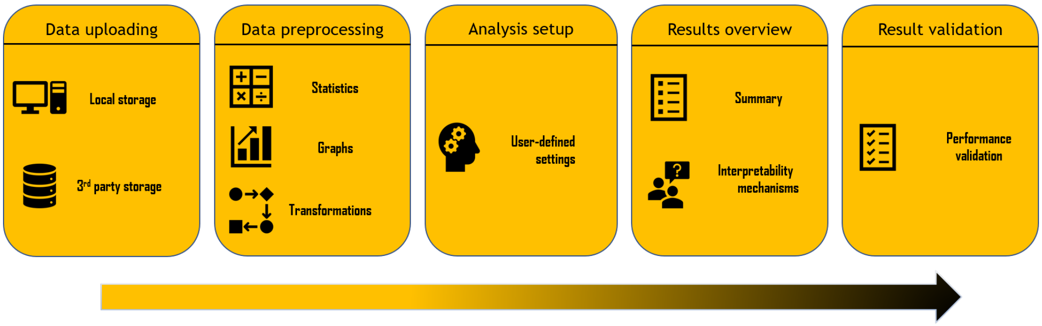


Figure 3.4: The AutoML pipeline as presented during the introductory presentation. Multiple details had been omitted, as the main goal of this overview was to give an understanding to the users as to how they would proceed with the user study analysis.

3.5 User analysis

Each participant was asked to complete a ML analysis on all 3 available AutoML services. The analyses were performed on real clinical data of arrhythmia patients, available at OpenML[73]. The type of the ML task was binary classification, and the user was asked to conduct an analysis and come up with a model able to predict whether or not a patient has arrhythmia. Afterwards, they had to use the resulted model to validate its performance on new unused test data. To be able to do so, we had previously partitioned the data in parts, namely train and test and provided them to the users. The users were instructed to use the train part for the data visualizations and training subtasks, while the test part was reserved for the validation subtask. This way we were able to check the validity of their answers to the 4 questions we posed to them in the latter stages of the user study. On a special note, we understand that using an AutoML service without prior, or limited, ML knowledge can be an overwhelming experience for the new user, especially when asked to do so in a limited time frame. To eliminate this problem, we provided the users with clear and easy-to-understand instructions, so that they were able to complete all the required subtasks and not get lost in the amount of information and functionalities available in the AutoML services. To make them feel even more comfortable running and completing the analyses, a facilitator was in the room, answering questions and helping them overcome possible dead ends. Finally, the order in which each user was called to use the 3 services was selected at random, to eliminate the possibility of having order bias[76].

3.5.1 Custom form

During the analysis procedure, the user was called to fill out a custom form about the ease-of-use of each of the AutoML services. It is consisted of 12 items, spread across the different subtasks. The users were called to select how much they agree with each item, once they had completed the corresponding subtask. Some items (8) have positive, while others (4) have negative wording. This technique was employed so that users did not go on auto-pilot and agree to all statements, also known as Acquiescent bias[28]. Another type of bias we wanted to avoid, by using a mixture of positive and negative items, is the Extreme response bias. Similar to Acquiescent bias, we wanted to deter users from selecting only extreme responses. This way, we aimed in getting the users pay attention to each individual item and provide meaningful feedback. We use a 5-point Likert scale[39] for the responses, which are: **1. Totally Disagree**, **2. Disagree**, **3. Neither Agree or Disagree**, **4. Agree**, and **5. Totally Agree**

3.5.2 Subtasks and corresponding custom form items

We tried to simulate a standard analysis procedure, where the user ought to a) upload the data to the AutoML service; b) use the available tools to visualize and understand the structure of the data; c) set up and start the analysis; d) overview the analysis procedure; e) examine the final result and f) apply the resulted final model to validation data and find the true predictive performance. In total, the user was instructed to complete 6 subtasks. After each subtask, they were called to fill the corresponding part of the custom form. The subtasks are:

- **Subtask 1: Data uploading**: Starting off the analysis, the users received the data and were asked to upload them to the AutoML service. Since this is a simple task, we had only 1 item in our custom form. The item was:
  - The data uploading process in this system is simple and intuitive.
- **Subtask 2: Data descriptive statistics and visualizations**: Once the data were uploaded, the users were called to use the mechanisms to get a better understanding of the data's characteristics and overall structure, and also look into how one can export this information. Through our instructions, the users were able to explore both the statistics, as well as the data visualizations provided by the AutoML services. The items of the custom form corresponding to this subtask were:

  - The statistics and visualizations provided helped me get a better understanding of the data.
  - Important statistics or visualizations are missing from the system.
  - The meaning of the data visualizations provided by the system is properly explained.

- **Subtask 3: Analysis creation**: Once the user understands the data characteristics, the next step is to set up and start the analysis. We provided the users with specific instructions to help them not get lost and have comparable results. Additionally, we asked from them to take some time and peruse through the available settings, in order to be able to respond to the custom form's items. These items are:

  - The process of analysis creation is easy and straightforward.
  - The available user - defined settings for the analysis setup are too technical.

- **Subtask 4: Analysis monitoring**: While the analysis was being executed, the users had the chance to look into the information provided by the AutoML services. This information opens up the black box of the AutoML analysis procedure and makes it, along with the final result, easier to trust by the user. This does not mean a service should give out all the raw information, but ideally present it in a way that will help the users understand each individual step of the analysis and not confuse them.As the analysis took at least 5-10 minutes to complete on each service, the users were able to explore the available information and convey their satisfaction via the items in the custom form. These are:

  - The information provided during the analysis monitoring is intuitive.
  - The amount of information provided during the analysis monitoring is insufficient.

- **Subtask 5: Result interpretation**: After the completion of the analysis, the users were asked to overview the final results summary and answer the 3 first intepretability questions. We did not emphasize on on interpretability mechanisms, since going through this information is a meticulous process requiring time and ML knowledge. A user study could be conducted just on this part of the AutoML services, with participants with advanced ML knowledge, in order to assess the quality of the provided interpretability mechanisms. The custom form item for this subtask is:

  - The summary presented is comprehensive.

- **Subtask 6: Result validation**: The last subtask of our analysis procedure is using the resulted model to validate its performance on external test data.

We focused on 2 parts of the validation mechanism, a) how easy it is to perform it on new data, and b) how intuitive the presentation of these results is.

- The mechanism for validating the final model is simple.
- The presentation of the results of the validation is confusing.

3.5.3 Comprehension questions

To understand if the users are able to find and extract the most basic information from the results of an analysis, we decided to ask 4 questions, after subtasks 5 and 6. It is important to note this was done to understand how easy-to-find the results are for a new user, and not to compare the predictive performance of the participating AutoML services. The questions were:

- Subtask 5:
  - **Question 1:** Report the performance (Area under the ROC curve/ ROC AUC/ AUC) after the completion of the training phase.
  - **Question 2:** Report the type of the selected final model.
  - **Question 3:** Write down the 2 most important (informative) features (variables) of the data, as reported by the AutoML service.
- Subtask 6:
  - **Question 4:** Report the performance (Area under the ROC curve/ ROC AUC/ AUC) of the final model on the test dataset.

The users were free to input their answers and were instructed to take some time to find the correct information.

3.5.4 System Usability Scale (SUS)

After each analysis, the users were called to fill in a System Usability Scale (SUS) [8] form. SUS is an industry standard when it comes to measuring the ease-of-use of a service and has been broadly used on multiple different domains[38]. It is consisted of 10 items, Odd-numbered items are all in a positive tone, while even-numbered items in a negative tone. The items are:

1. I think that I would like to use this system frequently.
2. I found this system unnecessarily complex.
3. I thought this system was easy to use.
4. I think that I would need assistance to be able to use this system.
5. I found the various functions in this system were well integrated.

6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found this system very cumbersome/awkward to use.
9. I felt very confident using this system.
10. I needed to learn a lot of things before I could get going with this system.

The positive aspects of SUS are:

- It is free to use.
- It is an intuitive scale and the user study's participants can understand its item with ease.
- It can be used on small sample sizes and retain the reliability of its results.
- It has been proven to work and can differentiate between usable and unusable systems.
- It has been broadly used and its results can be interpreted based on the final result, regardless of the type of the participating systems.

3.6 Results

In this section, we review the results of the user study in 3 parts. In the first part, we compare and comment on the the scores derived from our custom form and in the second part, we examine the answers given in the 4 questions we asked the users. In the final part, we discuss the SUS results.

3.6.1 Custom form results

To score the responses of our custom form and be able to use them in order to compare the 3 participating AutoML services, we came up with an idea similar to the SUS score is calculated. So, to the score a service based on the answers of a user, we follow these steps:

- A = Sum of the positive items' points - 8
- B = 20 - Sum of the negative items' points
- Custom Score = A + B

or, in a mathematical representation:

$$system\_score = \sum_{i=1}^{12} \begin{cases} response_{(i)} - 1, & \text{if } item_{(i)} = \text{positive item} \\ 5 - response_{(i)}, & \text{if } item_{(i)} = \text{negative item} \end{cases}$$

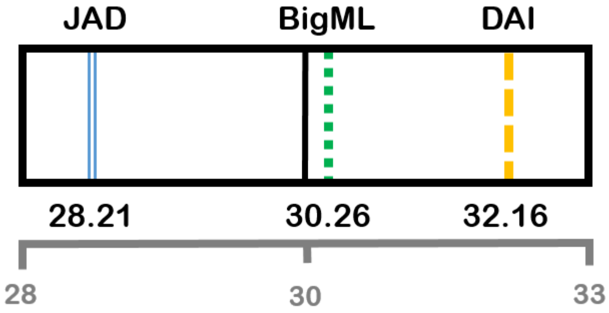
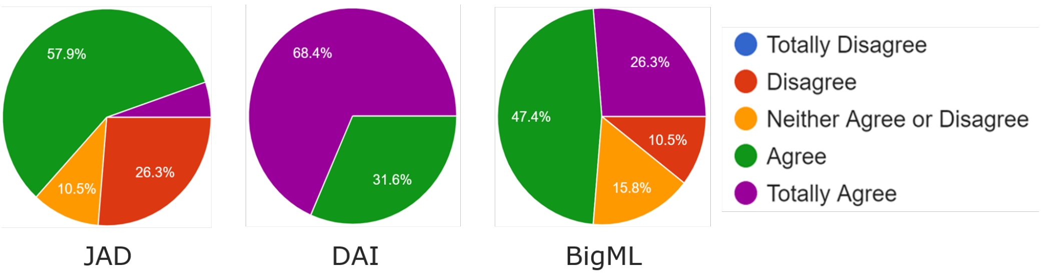


Figure 3.5: Custom form results. The maximum possible score 48. All systems share similar results, with DAI leading and JAD having the lowest score.

With this formula, the maximum score a service can achieve is **48**. In Fig. 3.5, we see the scores of the participating AutoML services. Even though, all 3 services are close in score, their overall performance is mediocre. Considering the perfect score is 48, DAI manages to get **(32.16)**, which equals to 67%, BigML **(30.26)** (63%) and JADBio gets **(28.65)** (60%). This is indicative of challenges users came across, during the completion of the analyses on all 3 services. Of course, we should not overlook the fact the users were ML amateurs, so the whole procedure is new to them and can be cumbersome. To find exactly which parts obfuscated the users, we are looking into the results of each of the subtasks, for all 3 services:

Subtask 1: Data uploading

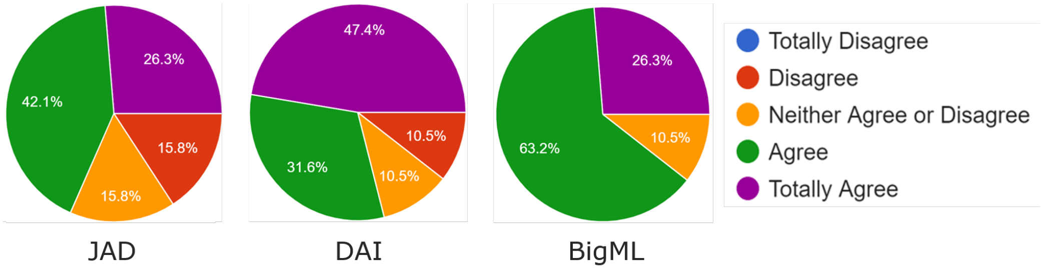
- The data uploading process in this system is simple and intuitive.



Even though data uploading is a straightforward task, we see differences on the user-experience among the 3 participating services. DAI is the only service which all users agreed this process is easy-to-complete, whereas for JADBio and BigML, some users found it confusing.

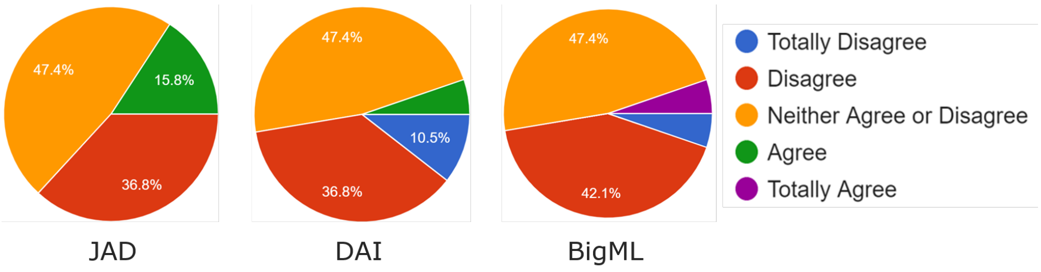
Subtask 2: Data descriptive statistics and visualizations

- The statistics and visualizations provided helped me get a better understanding of the data.



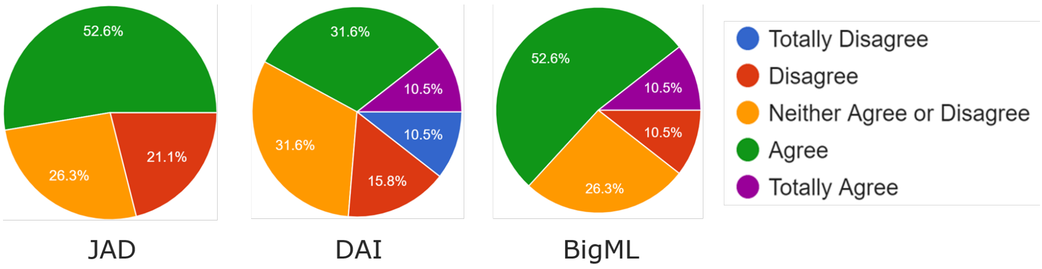
With this question, we are highlighting the importance of mechanisms able to explain the data and their characteristics to the users. Even though there are no major differences in the user-satisfaction, all users found the statistics and visualizations vital in understanding the data given to them for the analyses.

- Important statistics or visualizations are missing from the system.



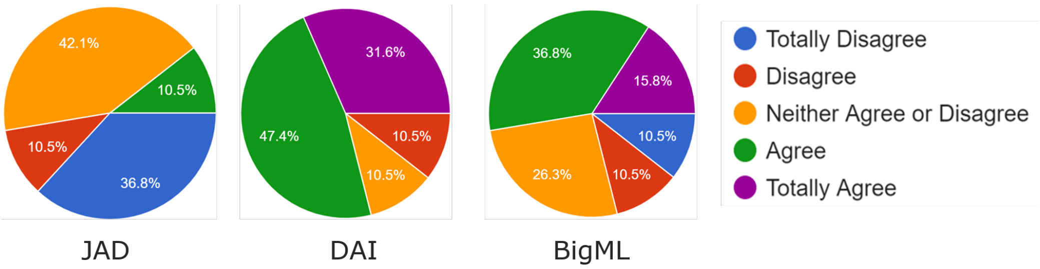
Due to the ML expertise level of the users, most of them were unsure as to whether or not there are data interpretation mechanisms that should be added to any of the services. However, the ones who answered with a non-neutral response, seem to be satisfied by what is provided.

- The meaning of the data visualizations provided by the system is properly explained.



Having multiple data interpretability mechanisms alone does not suffice. Proper explanation on how they work and what is their output, is as important as adding them to the AutoML service in the first place. From the responses, we notice the majority of users thought the provided explanations were adequate.

- The export options of the visualizations are satisfactory.

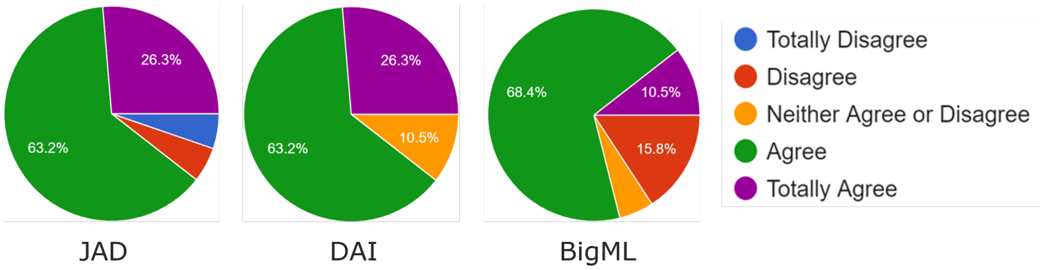


With this item we explored the available export options of the data visualizations provided by the services. Regarding DAI and BigML, the users seemed to be satisfied with what is available, while JADBio is the only service receiving negative responses.

After gathering up the results of this subtask, there is no AutoML service with a clear edge over the rest, when it comes to visualizing and getting information about the user's data.

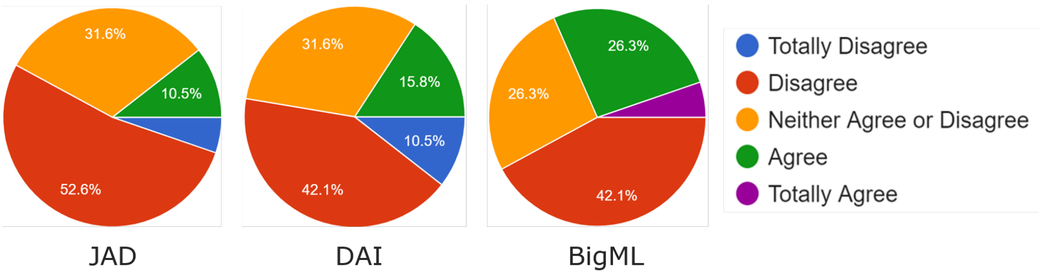
Subtask 3: Analysis creation

- The process of analysis creation is easy and straightforward.



In this subtask the users were called to create an analysis. Reviewing the results, we understand they found setting up an analysis simple in all participating services.

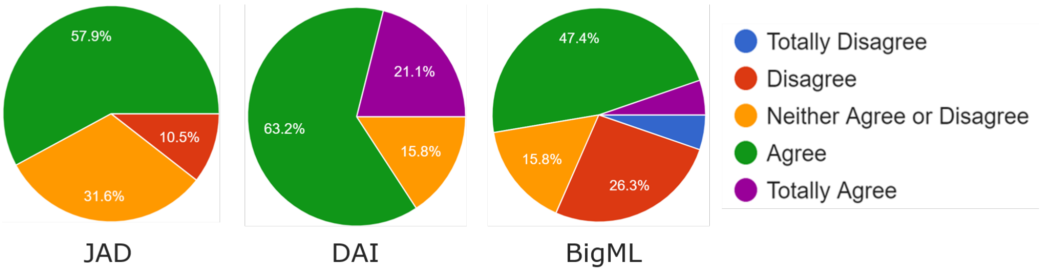
- The available user - defined settings for the analysis setup are too technical.



Even though the users were not proficient in using ML, the fact that AutoML services hide most of the technicalities and advanced options when creating an analysis, makes the analysis options easy-to-understand and setup for everyone. This is one of AutoML's main goals, removing the burden of choosing the correct settings for an analysis from the users and exposing only high-level options.

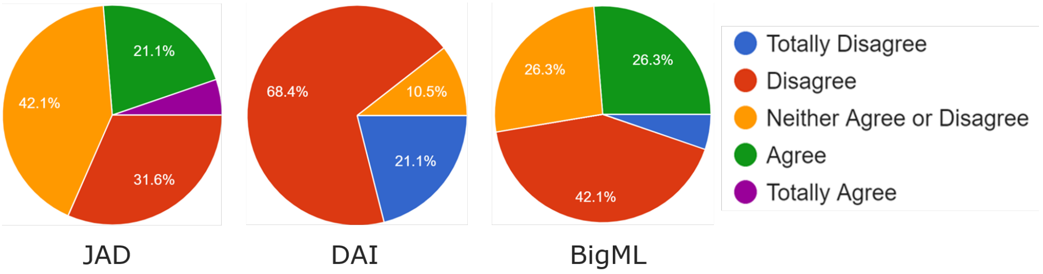
Subtask 4: Analysis monitoring

- The information provided during the analysis monitoring is intuitive.



About the information provided during the analysis execution, DAI had the highest score, with no negative responses, followed by JADBio and BigML, which received some negative feedback.

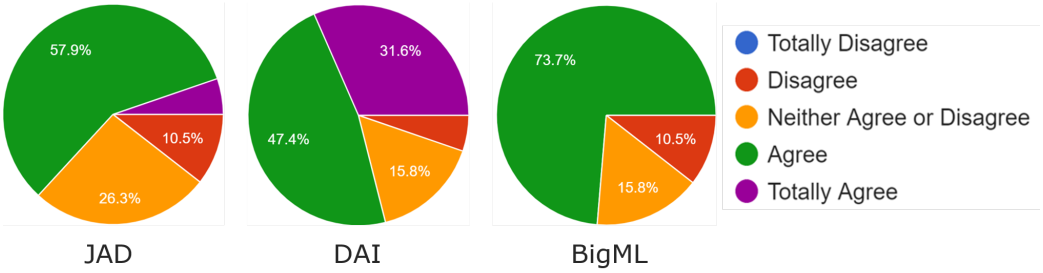
- The amount of information provided during the analysis monitoring is insufficient.



In this item, we see a notable difference in the responses. Regarding DAI, most of users where at least partially satisfied with the information provided during the analysis. For JADBio and BigML, less than half of the users thought that they were given enough insights on the analysis. In order to make an AutoML system trustworthy, it has to be transparent during the analysis by communicating as many analysis progression details as possible.

Subtask 5: Result interpretation:

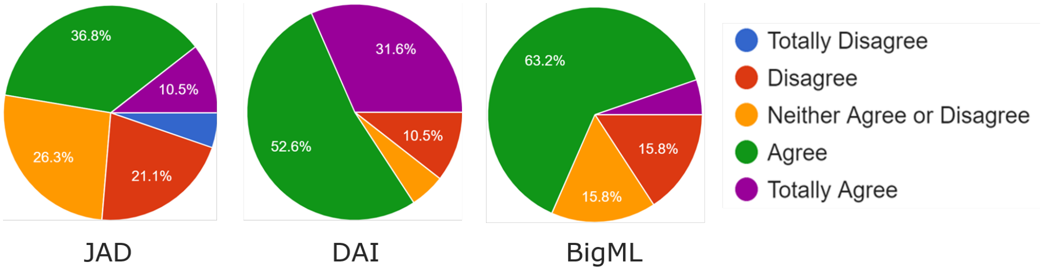
- The summary presented is comprehensive.



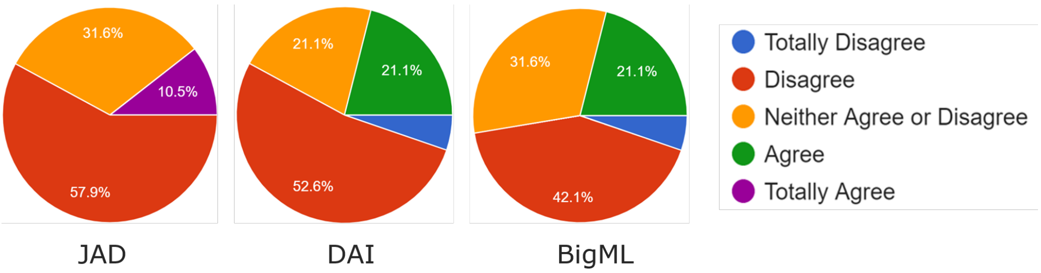
As we mentioned earlier, we did not want to focus on the interpretation of results, as it is consisted of multiple components and can be time-consuming. For these reasons, we focus on the final summary of the results. There are no major differences in the scores, with all of them being positive. This translates to the fact all services help the users get a clear grasp of the results and create understandable summaries.

Subtask 6: Result validation

- The mechanism for validating the final model is simple.



- *The presentation of the results of the validation is confusing.*



The last subtask was about validating the resulted model on new unseen data. About the way this mechanism is introduced in these AutoML, DAI and BigML receive a positive response from the majority of the users, while JADBio seems to have confused most of them. Still, when it comes to the way the validation results are presented, all 3 services receive a positive score, meaning that the users managed to understand them.

3.6.2 System Usability Scale form

Once the analysis was completed, the users were called to complete a SUS form. The SUS score can range from 0 to 100, but it should not be interpreted as a percentage, since the average score for a service is 68. The way SUS score is calculated is simple:

- SUS is composed of 10 items.
- The user is being provided with 5 responses, ranging from 1 (Strongly disagree) to 5 (Strongly Agree)
- For odd-numbered items, subtract one from the user response.
- For even-numbered items, subtract the user responses from 5.
- Add up the converted responses for each user and multiply that total by 2.5. This converts the range of possible values from 0 to 100 instead of from 0 to 40.

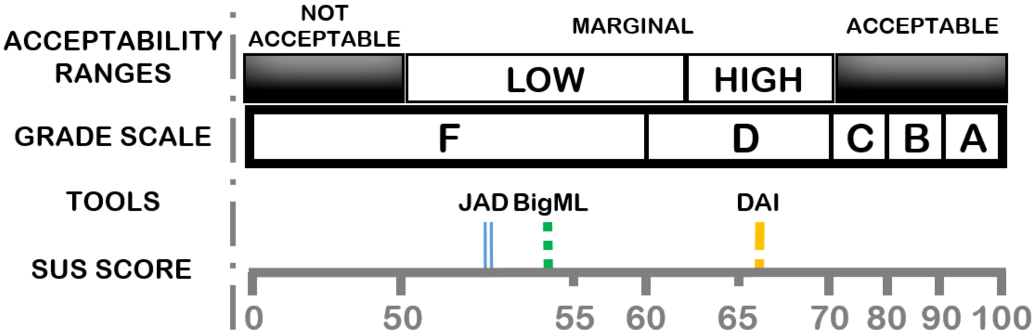
or, by using an equation:

$$system\_score = 2.5 \times \sum_{i=1}^{10} \begin{cases} response_{(i)} - 1, & \text{if } i \text{ is odd} \\ 5 - response_{(i)}, & \text{if } i \text{ is even} \end{cases}$$

In Fig 3.6, you can see the results, accompanied by different scales used to interpret them. We make 2 observations on them. Firstly, all 3 services scored low, compared to the max score (100), and they were below the average (68). Secondly, DAI seems to have the edge over JADBio and BigML in this comparison, scoring

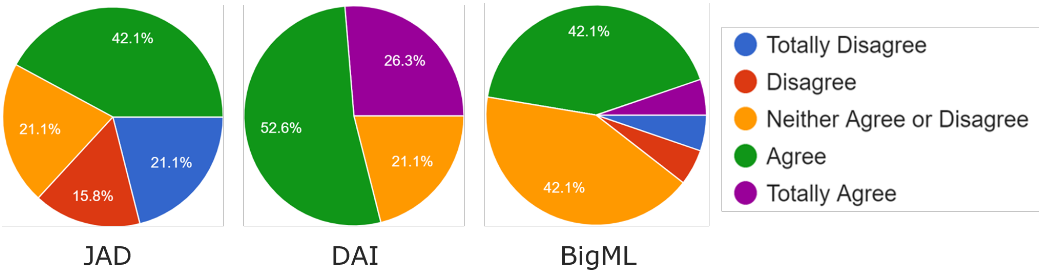
66.6 points, whereas the former two scored 53.3 and 53.4 respectively. This makes DAI the only system able to get a D grade (JADBio and BigML score F), and also being on the high-marginal acceptability range. What this result translates to, is that all systems get a low usability grade and there is ample room for improvement in this aspect.

Figure 3.6: SUS form results. The results are explained in 3 different scales, all proposed by the SUS creators and existing users. DAI has a clear edge over the other 2 systems, but all 3 score below the true average (68) of this usability form.



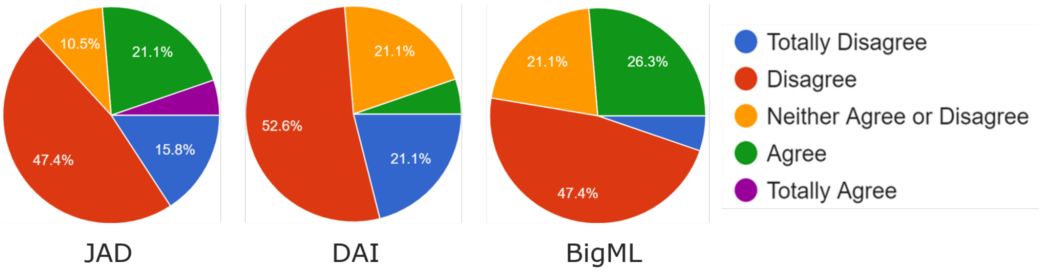
Following, we are looking at each of the SUS' items individually, to gain more insights on why the systems got these scores.

1 I think that I would like to use this system frequently.



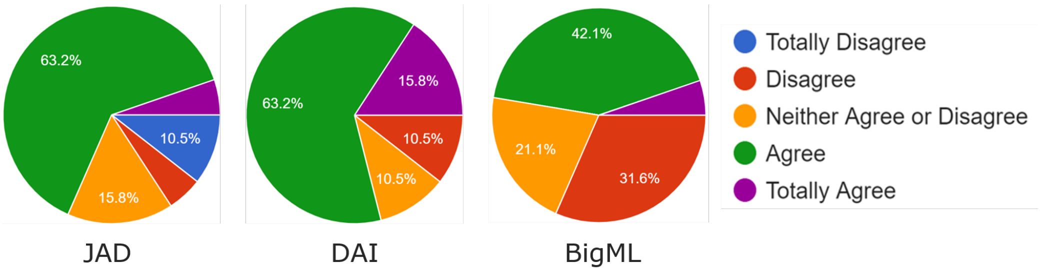
Starting off with our first item, it is about whether or not the users were intrigued enough by this system, in order to re-use it in the future. As we can see, DAI is the only system where the vast majority of the participants said they would come back, with BigML and JADBio following, having a score.

2 I found this system unnecessarily complex.



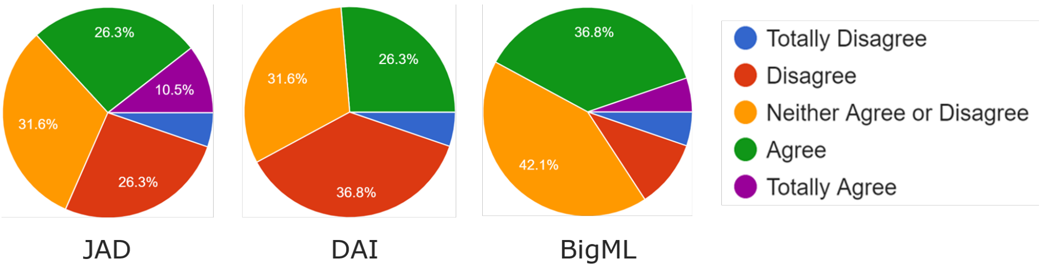
When questioned about the complexity of the systems, all had good scores, with DAI once again having the highest. This is an important finding, as AutoML systems are trying to remove as much of the technicalities and complexity when constructing an analysis pipeline. By looking at the results, they are moving to the right direction and this score should only increase in the future.

3 I thought this system was easy to use.



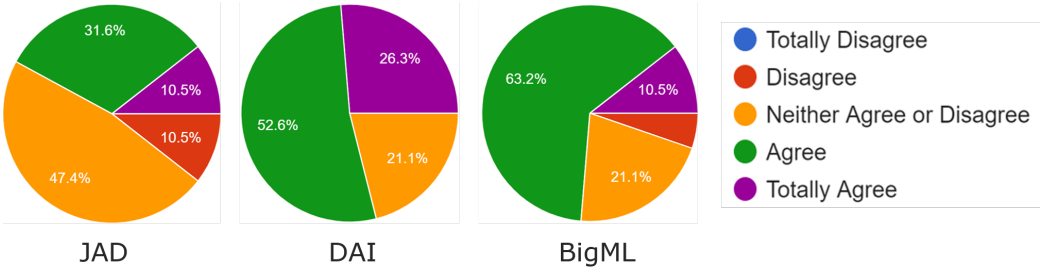
Complexity and ease-of-use are 2 different things. A simple system can be hard to use, due to bad design decisions or by missing basic functionalities. On the other hand, a complex system, such as an AutoML service, wants to make the whole process as clear as possible to the users and guide them through its entirety. The results of this item showed that even though the users had no previous experience on these systems, while some of them had no ML experience at all, the majority did not find trouble using any of them.

4 I think that I would need assistance to be able to use this system.



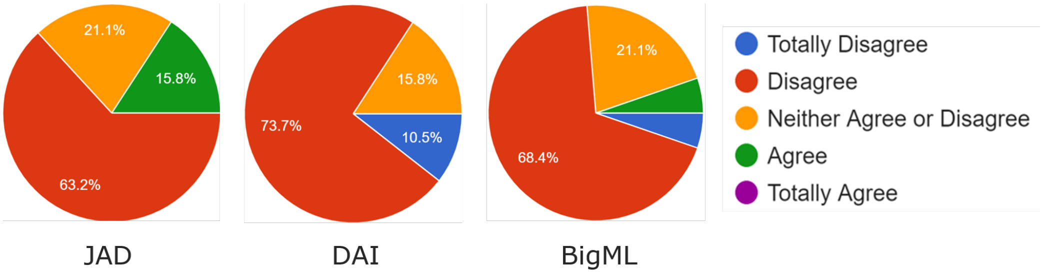
The main idea of AutoML, is that a user should be able to correctly perform an analysis without the guidance of a ML or AutoML system expert. Viewing the results, the majority of the users felt confident they would be able to use all 3 systems unassisted, with no system being significantly better or worse than the others.

5 I found the various functions in this system were well integrated.



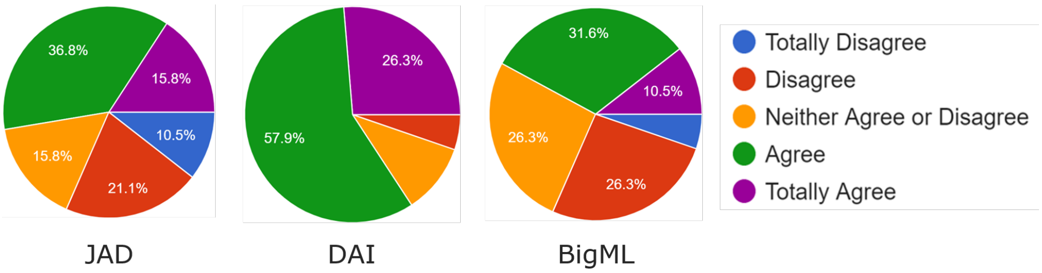
Apart from having multiple mechanisms, an AutoML system should additionally make sure they are correctly integrated. An error can greatly decrease the user experience and make the system appear less trustworthy and robust. When asked, the users seemed to be indecisive regarding JADBio, and were much happier with how the various functionalities were implemented in DAI and BigML.

6 I thought there was too much inconsistency in this system.



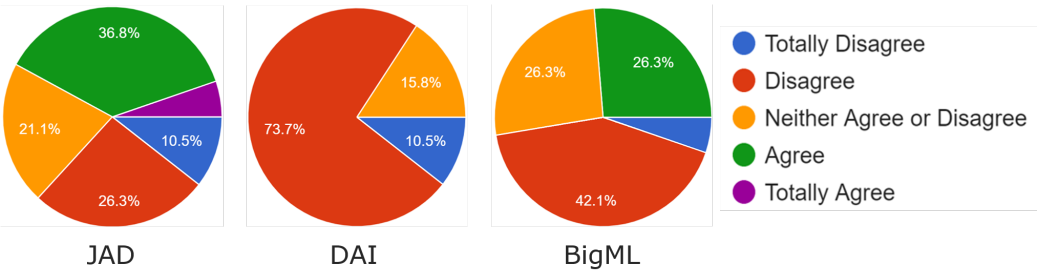
This item focuses on whether or not the systems deliver their content in an accordant way. Making constant changes can impact the UX, as they may lead to inconsistencies between different parts of the system. This, as a result, makes navigating through the system harder, as not having a unified approach on how information is communicated to the users, can lead to confusion. The users should be able to easily find what they are looking for at any time, thus increasing their productivity and decreasing their frustration. All participating systems have good scores in this item, with DAI having the edge, with no negative replies.

7 I would imagine that most people would learn to use this system very quickly.



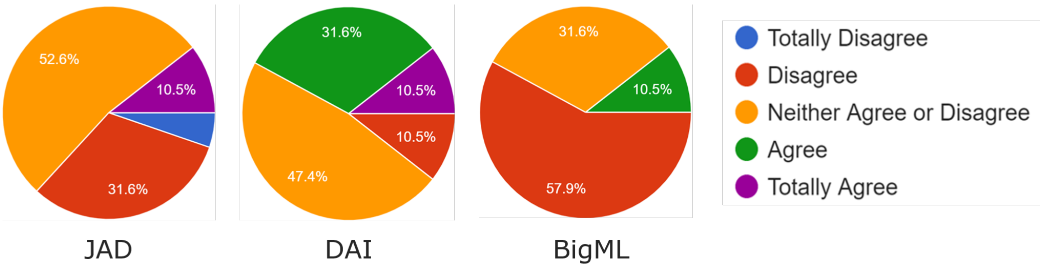
How easy it is to get started using a system plays an important role when choosing among multiple options. A steep learning curve can prove to be time-consuming and a make-or-break factor for both inexperienced and expert users. The results depict a major difference between the score of DAI and those of JADBio and BigML, with the participants thinking DAI was the easiest one to get accustomed to.

8 I found this system very cumbersome/awkward to use.



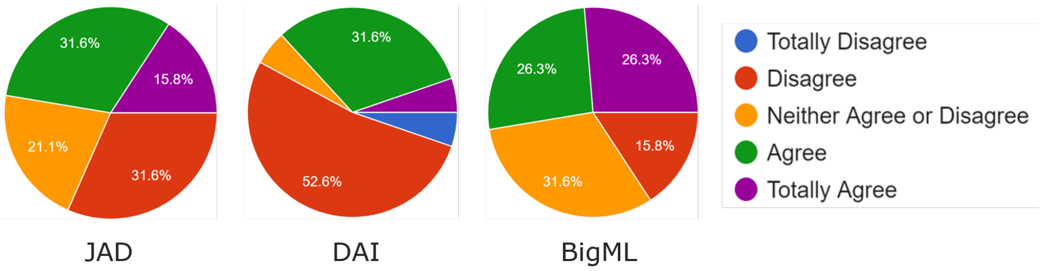
This item focuses on how easy it was for the users to navigate through the systems and complete the required tasks. Once again, DAI received the highest score, with no negative responses, while users seemed to find JADBio and BigML more awkward to use.

9 I felt very confident using this system.



When creating an AutoML system, one of the top priorities should be to make users self-assured about validity of their actions and the correctness of the final result. This can be done by streamlining all available functionalities. In this item, we notice all systems had a low score, with BigML gathering the lowest score. This outcome was expected, as the participants were called to use systems previously unknown to them, to complete a ML analysis, which is also something they were not experts of.

10 I needed to learn a lot of things before I could get going with this system.



The last item of the form is about the prior knowledge required to use these AutoML systems. DAI scores above average, while JADBio and BigML have a negative score, with the latter having only 2 of the users thinking they could use it without learning other things in advance. This is an interesting find, as all users were called to complete the exact same analysis across all participating systems. This goes to show the importance of creating an easy-to-use system, which the users can take advantage of by spending the minimum time possible in learning how to operate it.

To summarize the SUS form results, DAI had the highest score, while JADBio and BigML had similar results across most of them.Particularly, DAI seemed to be the most consistent and easy-to-use system, adding to the fact the users would most likely choose to use it again for their own projects. Still, it is important to notice not JADBio or BigML failed in the overall score, being on the marginal-low range of acceptability scale and that all of the systems scored less than the SUS average. If we take into account the limitations of using the SUS form in our user study, i.e. having participants with an amateur ML knowledge using advanced AutoML systems, we see there is room for improvement in all areas. However, one specific area all systems could vastly improve, is making the users feel confident using them, as this can increase the quality of user-experience and their popularity.

3.6.3 Custom questions responses

In this section we comment on the results of the custom questions, which the user was called to answer during subtasks 6 and 7. The aim of these questions was to see if the users are able to pinpoint some of the most vital results of a ML pipelines in the participating systems. What we report as results, is the percentage of correct answers for every question, across each individual system. More specifically:

- **Question 1:** Report the performance (Area under the ROC curve/ ROC AUC/ AUC) after the completion of the training phase.
  BigML (15/19 - 79.0%), DAI (19/19 - 100%), JADBio (18/19 - 94.7%)

- **Question 2:** Report the type of the selected final model.
  BigML (18/19 - 94.7%), DAI (9/19 - 47.4%), JADBio (10/19 - 52.6%)

- **Question 3:** Write down the 2 most important (informative) features (variables) of the data, as reported by the AutoML service.
  BigML (16/19 - 84.2%), DAI (18/19 - 94.7%), JADBio (16/19 - 84.2%)

- **Question 4:** Report the performance (Area under the ROC curve/ ROC AUC/ AUC) of the final model on the test dataset.
  BigML (12/19 - 63.2%), DAI (19/19 - 100%), JADBio (3/19 - 15.6%)

Starting off with Question 1, the information regarding the training performance estimation of the resulted model appears to be easy-to-find on all systems. However, when it comes to reporting the type of the final model, the users could not find this information with ease at DAI and JADBio. Continuing, we asked the users to report the 2 most informative variables (or features) of the data. In this question, most of the users had no issues locating where this information was located on all 3 systems. Lastly, once the users completed the validation subtask, we asked from them to report the validation performance, as found on the systems. In this question, we had 3 very different results. Starting off with the highest performance of correct answers, in DAI, all users replied correctly, whereas, in BigML, 12 out of 19 users were able to find this information. However, this was not the case for JADBio, where only 3 out of the 19 users reported the correct value. Questions like these can directly aid the development teams in identifying bad design decisions and help them understand how the users view the information provided by the system.

3.7 Limitations and conclusions

We discuss the limitations and conclusions of our user study. To begin with, the results of this user study have been mostly invalidated by now. AutoML is becoming more and more popular, and so the participating AutoML systems also evolve to be able to compete. Some have expanded their functionalities, while others have radically changed their whole UI interface, alternating the delivered user-experience. Of course, this does not mean that user studies have no point, just that their results should be considered immediately after being conducted. Another limitation, is the amount of participants and their characteristics. We gathered 19 users, with the vast majority of them being complete ML amateur. This means that we have a limited number of opinions and we miss out on the ML experts' opinions, which could differ to a great extent. Continuing with the analysis each user had to complete, we need to mention we created generic subtasks, trying to cover most of the key points of an analysis, without putting emphasis on a particular one. This caused as a result not reviewing other key aspects of AutoML systems, such as results interpretability mechanisms, final model export options, and available deployment options, to name a few. Lastly, about the time frame of the user study, even though 2 hours were enough time for most of the users, some struggled completing all 3 analyses, so there may be room for adjusting it as well. In general, AutoML systems have a steeper-than-expected learning curve, mainly due to the complexity of the problem they try to solve. AutoML teams should try make the analysis process more intuitive and improve the systems (tutorials, available documentation, working examples etc.) created to help the the users understand how to easily and correctly use their systems.

Chapter 4

Methodology of AutoML Quantitative Evaluation

4.1 Introduction

In the final chapter, we describe our methodology for quantitatively evaluating AutoML systems regarding the correctness and quality of their results, and present our findings. The first aspect of AutoML systems we want to evaluate is the accuracy of their predictive performance estimation, also known as estimation bias. Secondly, we conduct an out-of-sample performance comparison to search for systems which might have the edge over the rest when it comes to predictive performance. Lastly, we are interested in finding trends of our results with the characteristics of the data we included in our benchmark. Our results show most AutoML systems significantly fail to accurately predict the true performance of their outcome. Regarding the hold-out performance comparison, there was no single winning system for either of the ML tasks we included in our evaluation (binary classification and regression). Finally, depending on the system, we found all of the aforementioned results to be statistically significantly correlated to different data characteristics.

4.2 Related work

There has been an increasing interest in finding ways to adequately evaluate AutoML systems. Starting with the scope of these studies, most are focused on the classification task (both binary and multiclass). This creates a gap as regression has only been highlighted by one recent study [68] and there are other tasks not being considered at all (e.g. time-to-event, survival analysis). Moving over to data used, most studies use tabular data from the OpenML repository (OpenML benchmarks[18][5] or custom data lists), while a few focus on image data [6][25]. AutoML systems are complex systems providing multiple functionalities to the users. Apart from comparing them solely based on their hold-out performance, related work tries to cover other dimensions of them as well. To start off, they test their ability to consistently deliver similar results when repeating an analysis [68].Furthermore, there have been works emphasizing on the structure and complexity of the systems' outputs[79]. This provides the users with vital information, as many could be more inclined to use a system which outputs simpler results, depending on the problem and application. Lastly, many related studies are interested in tuning the time limit hyperparameter of AutoML systems. This way they investigate which is the optimal time budget in order to achieve the best possible result, based on different problem cases. We decided to focus on creating a methodology to evaluate AutoML systems based on; a) the accuracy of their predictive performance estimation, also known as estimation bias and b) an out-of-sample performance comparison. Also, we were interested in finding trends of these results with the characteristics of the data we included in our benchmark and examine their statistical significance.

Table 4.1: Quantitative evaluation surveys of AutoML systems. **ML tasks**: C: Binary classification, MC: Multiclass classification, R: Regression, **OML**$^{*\mathbf{1}}$: OpenML benchmark, **OML**$^{*\mathbf{2}}$: OpenML100 + OpenMLCC18 + OpenML benchmark, $^{*\mathbf{3}}$: Included both AutoML and AutoDL systems.

	Study	[18]	[79]	[68]	[6]	[25]	Our evaluation
Scope	ML tasks	C, MC	C, MC	C, MC, R	C	MC	C, R
	Data types	Tabular	Tabular	Tabular	Images	Images	Tabular
	Data benchmark used	OML^{*1}	OML^{*2}	Custom	Custom	Custom	Custom
	Participating datasets	39	73	300	6	2	200
	Number of AutoML systems	4	5	6^{*3}	2	3^{*3}	5
Dimensions	Hold-out performance	Y	Y	Y	Y	Y	Y
	Estimation of predictive performance bias	N	N	N	N	N	Y
	Statistical trends of results with data metafeatures	N	N	N	N	N	Y
	Similarity of results across repeated analyses	N	N	Y	N	N	N
	Noisy data handling	N	N	N	N	Y	N
	Output pipeline size & complexity	N	Y	N	N	N	N
	Time-performance tradeoff	Y	N	Y	N	Y	N

4.3 Participating AutoML systems

AutoML is an emerging domain with multiple potential applications, so naturally the number of available open source and commercial AutoML systems is increasing rapidly. For our study, we selected the participating systems based on the following criteria:

- The scope of the system must cover the scope of our study. We focus on binary classification and regression ML tasks. Regarding the data type, we use tabular data with both continuous and discrete features, as well as missing values for our experiments.

- A system should provide us with free unrestricted access of its functionalities. This is obvious when it comes to open source systems, but is often not the case for the commercial ones. Most of them do not allow their inclusion in a public comparative performance evaluation and restrict it through their licensing. Since we were interested in adding them in this evaluation, we asked from multiple commercial AutoML systems[1] for persmission to include them to our study. Unfortunately, only JADBio responded.

- We emphasize on AutoML and do not include Automated deep learning (AutoDL) systems. They belong in the Deep Learning subdomain, hence a different evaluation including only AutoDL systems ought to be more appropriate. This translates to popular AutoDL systems, such as Autokeras [34] and Ludwig [44], not being included in our evaluation.

- The system should be currently maintained or developed. This restriction allows us to focus at comparing the popular up-to-date systems and not highlight known issues with the outdated ones.

- The system ought to report or store the training performance estimation once the training phase is completed.

Applying all the aforementioned guidelines and restrictions, led us to a total of 5 AutoML systems (4 open source and 1 commercial):

- **Autosklearn** (ver. 0.6.0)[15], one of the most popular AutoML systems and provides the users with an automated sklearn [49] estimator replacement. Autosklearn uses SMAC [30] to optimize its models, metalearning to warmstart the hyperparameter optimization procedure and its default output is a 50-model ensemble. For the performance estimation, it uses the Hold-out protocol.

- **GAMA** (ver. 20.2.0) [19], a new AutoML system utilizing genetic programming for its optimization procedure. It uses the Cross Validation protocol[55] to estimate the performance of its output.

- **h2o.automl** (ver. 3.28.0.2) [24], another well-known AutoML system built using the H2O open source ML platform. It uses a mixture of grid search and random search [2] for model optimization and boasts 2 different ensembling techniques, to ensure high predictive performance. To calculate the performance, it also uses Cross Validation.

- **Just Add Data Bio (JADBio)** (live version)[70], a commercial system focusing on feature selection and analyzing low-sample, high-dimensional data. It uses Repeated Cross Validation with bias correction[71], to correctly report the performance of its final model.

- **TPOT** (ver. 0.11.1)[36], another widely-used system using genetic programming for machine learning optimization. The main difference with GAMA

[1] Auger.AI, BigML, Darwin, Datarobot, DriverlessAI, JADBio

lies in the evolutionary algorithm it employs in the optimization stage. It also employs Cross Validation for the performance estimation.

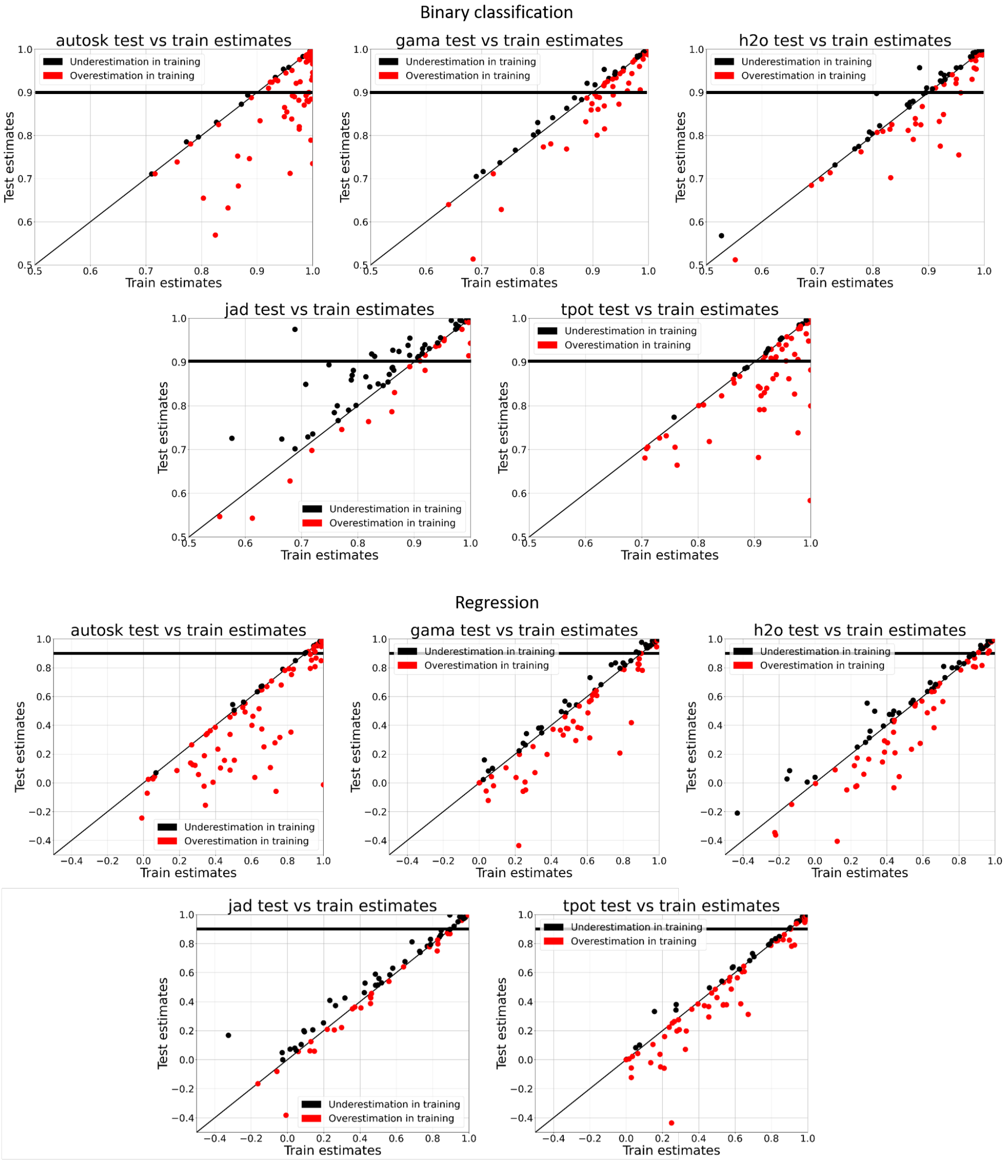


Figure 4.1: The bias plots for all AutoML systems across all experiments, on both ML tasks. With red we mark the overestimated cases and with black the underestimated ones. The black horizontal line depicts the 0.9 hold-out performance limit.**Top**: The binary classification experiments results for each AutoML system. **Bottom**: The regression experiments results for each AutoML system. Autosklearn and TPOT show the biggest overestimation trends.

4.4 Evaluation methodology

In this section we describe our methodology for evaluating the selected AutoML systems. We start off by presenting our data benchmark creation process. Afterwards, we present the evaluation protocol we follow in our experiments. Lastly, we describe the tests used to study the statistical significance of our results.

4.4.1 Benchmark datasets selection

In order to adequately evaluate the participating AutoML systems, our objective was to create a representative benchmark covering a wide range of data characteristics, for both binary classification and regression ML tasks. To do so, we started off by defining the dimensions range we considered when selecting datasets.

Regarding the *Number of Samples*, we included datasets with at least 100 but no more than $150k$ samples. The lower bound is set, as a previous study[70], conducted on more than 600 datasets with less than 100 samples, evaluated JADBio and Autosklearn on this kind of datasets, so there was no need to include them here. Regarding the upper bound, it is set because a) bigger sample sizes translate to easier problems[54] and thus, providing little to no useful information regarding their performance and generalization; b) extremely large sample-size problems are regularly used for evaluating the scaling aspect of systems.

Continuing, we also limit the *Number of features* between 10 and $100k$. This way we avoid datasets with too few features, while also allowing the inclusion of high-dimensional problems, which are usually harder-to-solve and have not been adequately represented in other AutoML benchmarks [5].

Our benchmark is consisted of datasets from the OpenML repository [73], as it provides curated data, together with multiple data characteristics in the form of metafeatures. This aids us in selecting datasets that abide to certain requirements and cover a wide range of available problems. However, the pool of available datasets is not uniformly distributed across the Samples/Features grid (Fig 4.2, thus, a random selection of datasets will result in some areas of the grid not being represented in the benchmark. We aim for an approximately uniform distribution of datasets in this Features/Samples grid. To do so, we have created and used a simple methodology. We started off by sampling 1 dataset from each grid cell and repeat this across all cells until we have our desired number of datasets. Moreover, to ensure we also included data with missing values or big imbalance in their class, we manually added extra datasets. The result of this procedure led to selecting 200 datasets, 100 for binary classification and 100 for the regression task (Figure 4.2). All selected datasets can be found at Table 6.1 and Table 6.2.

By comparing our benchmark with others which have been used in related works, we validate that our methodology is successfully managing to create representative data lists using the OpenML repository datasets. Starting with the binary classification list, we compare it against the OpenML benchmark.

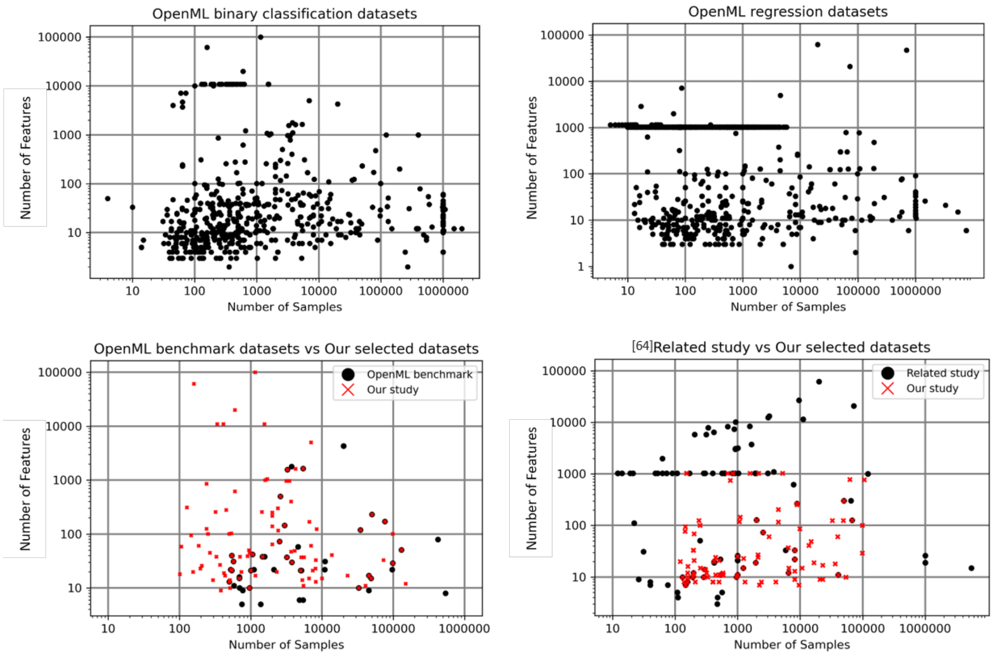


Figure 4.2: The datasets available in the OpenML repository and those we sampled for our benchmark, presented in the Samples/Features grid. **Top Left**: All available binary classification datasets in OpenML repository **Top Right**: All available regression datassets in OpenML repository **Bottom Left**: Our selection for the binary classification task, compared to the union of OpenMLCC18 and OpenML benchmarks, achieves a better coverage of the dimensions grid. both in terms of samples and features. **Bottom Right**: The related study[68] also including this ML task has oversampled datasets from a particular domain (QSAR datasets) and included multiclass classification datasets in its benchmark, therefore reducing its credibility.

4.4.2 Evaluation protocol

We describe our evaluation protocol by starting with our experimental setup. We set only the mandatory settings for each system, i.e. time budget for each analysis and the performance metric to optimize for. Regarding the time budget, we decided to set it to 1-hour, as the related work[73][18][68] indicates it is ample time for analyzing data of small to medium dimensions. For our hardware setup, we run the experiments of the open source systems (Autosklearn, GAMA, h2o.automl, TPOT), in GRNet's HPC `phi nodes` infrastructure. For each analysis, we utilize 8 CPU cores and 50GB of memory. Since JADBio is a commercial system running in its own infrastructure, what we can do is also limit the available cores for each analysis to 8.

When it comes to data preparation for our analyses, the only step we perform is a

50-50 split, stratified for classification datasets. We use the first 50% for training and getting the predictive performance estimation of the systems' output and the other 50% as a hold-out set used for obtaining the true test performance.

We use 2 different performance metrics for optimizing the AutoML systems and reporting their train and hold-out performances. For binary classification, we use the *Area under the ROC Curve (AUC)*. On how it measures the predictive performance, AUC considers all pairs of one positive and one negative samples. It equals to the probability that a positive sample will get a higher score by the model than the negative one. AUC is a widely used performance metric for binary classification analyses, as it measures the quality of scores of the model to rank samples correctly and is independent of the class distribution. Its baseline performance is 0.5 and its range is between 0 and 1. For regression we use the Coefficient of Determination (R^2). R^2 measures the reduction in uncertainty (variance) of the predictions by using the model compared to using a trivial model, most commonly the mean value of predictions. Moreover, it is independent of the scale of measurements, meaning no normalization is needed. R^2's baseline performance is 0, i.e. always predicts the mean value of the outcome, and its range is between $-\infty$ and 1.

We keep track of 2 measurements to see how well the participating AutoML systems can estimate their output's true performance. Firstly, we find the percentage of cases where the train estimation is larger than the test performance, i.e. the number of overestimations. Secondly, we calculate the average bias (train - test performances difference) across all experiments.

Lastly, we compare the hold-out performance of the systems in 2 scenarios. In the first scenario (Scenario A) we include all participating datasets and penalize the systems when failing to complete an analysis, by scoring them with the lowest performance possible. In Scenario B, we only include the datasets all systems successfully analyzed.

4.4.3 Computing statistical significance

We want to measure the statistical significance of our results for both the accuracy and the quality of the systems' output. For the average bias results, we perform 2 one-sample one-sided student's t-tests[63]. With the first test we examine whether the AutoML systems statistically significantly overestimate the predictive performance of their models:

$$H_o = \text{average bias is 0}$$

$$H_1 = \text{average bias is greater than 0}$$

While, with the second test, we check for the exact opposite, i.e. if the systems statistically significantly underestimate their predictive performance:

$$H_o = \text{average bias is 0}$$

$$H_1 = \text{average bias is less than 0}$$

In both tests, we safely reject the null hypothesis (H_0), if the pvalue is less than 0.05.
For our holdout performance comparison, we use the *autorank* tool[27] to perform 2 tests to check for the statistical significance of the evaluation results. The first is a Friedman's test[50] on the results of all AutoML systems. Its null hypothesis (H_0) states there are no statistically significant differences in the average rankings of the systems, while the alternative hypothesis (H_1), states the opposite. We can safely reject the H_0 and accept the H_1, if the pvalue is less than 0.05. If we accept the H_1, then the tool employs the Nemenyi post hoc test[67], to identify the systems having a statistically significantly different average ranking when compared to the others. The Nemenyi test also calculates the critical distance, or CD. CD translates to the minimum difference in the average rankings of 2 systems required in order to consider their results statistically significantly different.
Lastly, we use the Spearman method to find the linear relations between our results and the characteristics of benchmark data. We use 3 levels of significance for the results (0.05, 0.01 and 0.001). To better explain how to interpret the spearman results, a positive correlation translates to bigger overestimation or lower ranking in performance, while a negative correlation means more accurate performance estimation or higher ranking in performance (Fig. 4.3).

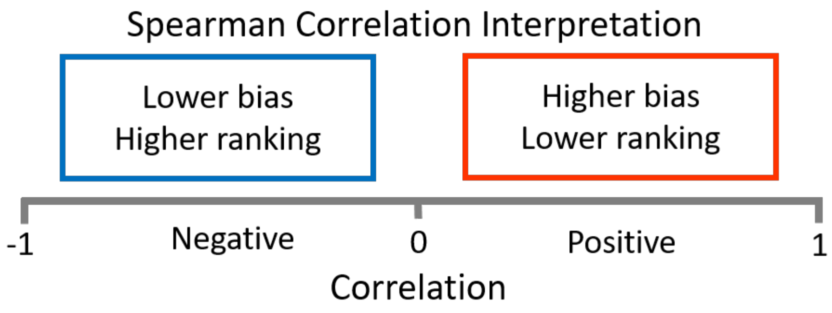


Figure 4.3: Explanation of Spearman method results when used for finding correlations between our evaluation results and data characteristics.

4.4.4 Data metafeatures

In order to find if the characteristics of our data have an impact on the bias and the hold-out performance of the systems, we keep track of metafeatures describing our data characteristics, which are readily available at OpenML.org. Our first metafeatures are the *Samples size* and the *Samples to features ratio*. These metafeatures can be critical, as small sample sizes can break the estimation and lower the performance of ML models, while a bigger sample size usually translates to better results. Additionally, we keep track of the *Features size*. In general, high-dimensional data need special handling and can cause computational and estimation problems. We also keep track of the number of different types of features, namely *Number of Continuous features*, *Number of Discrete features* and *Percentage of Discrete features.* We keep track of these metafeatures, as some data types require different statistical methods in order to properly handle them during training. Our last 2 metafeatures are about the *percentages of missing values* and the *majority class*, the latter for classification task. *Missing values* require special non-trivial imputations when handling them, while *Class imbalancing* can have an impact on the performance of the AutoML's output.

4.5 Experimental evaluation

In this section we present the average bias and hold-out comparison results, as well as the results of their correlation to the data characteristics of our benchmark.

Table 4.2: Bias results (average and percentage of overestimations) of all AutoML systems across both ML tasks. We additionally present the results corresponding to harder problems ($\leq$ 0.9 hold-out performance). With a * we denote the statistically significant results, with a significance level of 0.05. We observe most AutoML systems significantly overestimate the true performance of their output on average.

	AutoML system	Average bias	Overestimated cases (%)	Hard cases average bias	Overestimated hard cases (%)
Classification	Autosklearn	0.05*	81.25%	0.11	84.21%
	GAMA	0.01*	63.15%	0.03	65.62%
	h2o.automl	0.01*	45.74%	0.03	59.46%
	JADBio	−0.01*	31.52%	-0.02	28.57%
	TPOT	0.04*	71.27%	0.08	89.19%
Regression	Autosklearn	0.12*	82.61%	0.16	85.07%
	GAMA	0.07*	55.10%	0.10	64.86%
	h2o.automl	0.04*	53.00%	0.05	57.14%
	JADBio	0.00*	39.74%	0.01	43.55%
	TPOT	0.08*	68.00%	0.11	73.68%

4.5.1 Predictive performance estimation bias

Our first point of interest in this quantitative evaluation is the accuracy of AutoML systems. We start off by looking at the results of the classification analyses and continue with regression. Moreover, we examine whether these results are correlated to the data characteristics of our benchmark.

4.5.1.1 Binary classification

Looking at the results of our experiments at Figure 4.1, we notice all but one AutoML systems statistically significantly overestimate the performance of their models on average. Particularly, Autosklearn overestimates in more than 85% of the cases, with an average bias of 0.05. TPOT is another system showing major discrepancy between its performance estimation in the training phase and the true hold-out performance. Even though the percentage of overestimations is lower, it still stands above 75%, with an average bias of 0.04. Moving to GAMA and h2o.automl, they did better in terms of estimation accuracy, both having an average bias of 0.01 and the latter overestimating the true performance in half of the cases. Last but not least, JADBio is the only system giving statistically significantly conservative estimations. This is depicted in both the number of cases it overestimates (32.61%) and in its average bias, which is negative (-0.01). The experiments results show that for many cases the AutoML systems achieve both very good train and test performance. Even though this is expected as they all have implemented sophisticated ML pipelines, it may conceal the true extent of the performance overestimation problem. To address this issue, we filter out the experiments in which the AutoML systems achieve over 0.9 AUC in hold-out performance. The results confirm what we stated above, as now most systems see an increase in the percentage of overestimated results and in their average bias. This shows that for harder-to-solve problems, the majority of AutoML systems faces issues with the performance estimation during training. In particular, Autosklearn now overestimates in 84.22% of the experiments and the average bias is 0.11 (increased from 0.05). TPOT also produces worse results, as it overestimates 89.19% of the problems with an average bias of 0.08 (increased from 0.04). GAMA sees a minor decrease in the number of overestimations (65.62% from 67.47%) and h2o.automl a small increase (from 50% to 59.46%). However, we observe an increased average bias for both (0.03 from 0.01). The system not following the same patterns, is JADBio. It is the only one benefiting from our subsampling as it now outputs even less overestimations of the predictive performance (28.57% from 32.61%) and has an even lower average bias (-0.02 from -0.01), showing it is even more conservative in harder problems.

4.5.1.2 Regression

Continuing with the regression results, again most systems show a statistically significantly positive estimation bias (overestimation) and we observe similar behaviors. Autosklearn is the system with the most overestimations (82.61% of the total cases) and with the biggest average bias (0.12). Regarding the rest of the systems, h2o.automl, GAMA and TPOT have comparable performance, with TPOT leading in the number of overestimations (68%, average bias 0.08), while h2o.automl (53%, 0.04) and GAMA (55.1%, 0.07) are closer to 50%. Lastly, JADBio is the only system reporting conservative estimations for the majority of the cases (61.26%) and accurately estimates the true performance on average (0.00 bias). For our next step, we once again drop the experiments where the systems performed well and managed to achieve over 0.9 R^2. In this subset of datasets, all systems have worse performance both in terms of the number of overestimated cases and average bias. Autosklearn now overestimates the performance in 85.07%

of the experiments (from 82.61%), TPOT in 73.68% (from 68.00%) and GAMA in 64.86% (from 55.10%). We see a smaller increase in the number of overestimations for h2o.automl (57.14% from 53.00%) and JADBio (43.55% from 39.74%), with the latter being the only system giving conservative results in the majority of them. The bigger change is observed in the average bias of the AutoML systems. Autosklearn shows the biggest bias increase (0.16 from 0.12), and together with TPOT (0.11 from 0.08) and GAMA (0.10 from 0.07) reach a bias of at least 0.10. h2o.automl and JADBio are the only systems that manage to have only a marginal increase in their bias to 0.05 and 0.01 respectively.
Compared to the binary classification task, systems overestimate the true performance by a larger margin on average and in more cases. Particularly for autosklearn, it fails to estimate the predictive performance of its output, for both high and low-performing experiments. JADBio is the system with the most reliable estimations, showing close to optimal performance in both tasks.

Table 4.3: Number of AutoML systems successful experiments in our evaluation. Most AutoML systems had an over 90% on both ML tasks. JADBio was the only system with a sub-80% completion rate in one of tasks (regression).

AutoML system	Classification	Regression
Autosklearn	96	92
GAMA	95	98
h2o.automl	94	100
JAD	92	78
TPOT	94	100

4.5.2 Bias correlation to data metafeatures

We are interested in searching for trends between the accuracy of the systems' estimations and the data characteristics of our benchmark. To do so, we utilize the data metafeatures we collected during our benchmark creation process and user the Spearman method to look for relations between them and the average bias of the systems. We set the significance level to 0.05. The results show most estimations of AutoML systems are influenced by specific data characteristics. All the results can be found at Table 4.4.

4.5.2.1 Binary classification

Going into more details, we start off with the binary classification task. We discuss the statistically significant results for each system and provide possible explanations for their behavior.

Our first observation is that JADBio's performance estimations are independent of the data characteristics. Moving on, increased Sample size and Samples to

Table 4.4: Examining the trends of overestimation with data metafeatures. We performed spearman correlation to detect linear trends. Pvalues are denoted based on different significance levels ($*1$: 0.05, $*2$: 0.01, $*3$: 0.001). JADBio has no trend. Most automl systems show a reduction of their bias when the sample size increases and for smaller samples to features ratio. Additionally, there is a trend regarding the number of missing values and the estimation bias in the regression results.

Task	Binary classifcation					Regression				
Metafeatures	Autosklearn	GAMA	h2o.automl	JADBio	TPOT	Autosklearn	GAMA	h2o.automl	JADBio	TPOT
Samples size	-0.56^{*3}	-0.50^{*3}	-0.10	0.05	-0.44^{*3}	-0.63^{*3}	-0.48^{*3}	-0.33^{*3}	0.04	-0.49^{*3}
Features size	0.02	0.08	-0.33^{*2}	-0.08	-0.07	0.02	-0.02	-0.08	-0.10	-0.14
Continuous features (#)	0.07	0.08	-0.36^{*3}	-0.13	-0.20	-0.14	-0.10	-0.15	-0.22	-0.28
Discrete features (#)	-0.06	-0.19	0.24^{*1}	0.01	0.20	0.40^{*3}	0.28^{*2}	0.17	0.07	0.35^{*3}
Discrete features (%)	-0.06	-0.15	0.27^{*2}	0.07	0.21^{*1}	0.37^{*3}	0.27^{*3}	0.21^{*1}	0.08	0.37^{*3}
Missing values (%)	0.21^{*1}	-0.05	0.17	0.09	0.18	0.06	0.14	-0.01	0.04	0.21^{*1}
Samples to Features ratio	-0.67^{*3}	-0.58^{*3}	-0.18	0.16	-0.56^{*3}	-0.55^{*3}	-0.41^{*3}	-0.25^{*1}	0.15	-0.34^{*3}
Majority class (%)	-0.38^{*3}	-0.36^{*3}	0.14	0.09	-0.24^{*1}					

Features ratio, lead to smaller overestimation effects for Autosklearn, GAMA and TPOT. This is to be expected, as a bigger sample size translates to more available information during the training phase, which is used to create high-performing models and accurately estimate their performance. Focusing more on h2o.automl, its estimations become more accurate as the number of Features increases. The same is observed regarding the number of Continuous features. However, when the number of Discrete features increase, the bias increases. This shows that the system might come across issues when dealing with a specific type of data (Discrete). Lastly, an interesting finding is that systems have a lower bias as the class imbalance gets bigger.

4.5.2.2 Regression

Continuing, we look at what influences the average bias of the AutoML systems in the regression task experiments. Once again, JADBio's performance estimations do not trend with any of the data metafeatures, showing the robustness of its performance estimation protocol. Looking at the impact of the dimensions of the included datasets, we find the same trends, as Autosklearn, GAMA, h2o.automl and TPOT have a lower bias as the number of samples and samples to features ratio increase. In this task, it is more apparent that the type of the data affects the accuracy of the systems' performance estimation. In particular, Autosklearn, GAMA h2o.automl and TPOT become more inaccurate as the Number of Discrete Features increases. Moreover, TPOT is the only system which sees its average bias statistically significantly increasing, the bigger the number of missing values in the data.

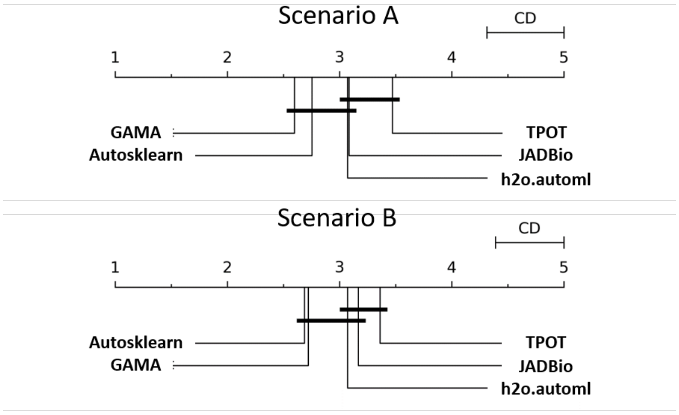


Figure 4.4: Classification model quality rankings comparison. **Top**: Scenario A - All participating datasets (Union) **Bottom**: Scenario B - Common completed cases across all systems (Intersection). The systems connected with a horizontal line have no significant differences in their average rankings, at a statistical significance level of 0.05. The CD is the minimum difference of average rankings between systems required to consider their performance significantly distinct. There is no clear winner among the participating systems in the classification ML task.

4.5.3 AutoML hold-out performance comparison

We evaluate the hold-out performance of the participating AutoML systems and look for trends of their average ranking with the data characteristics of our benchmark. To do so, we split our comparison into 2 scenarios. On the first scenario (*scenario A*), we include the entirety of our benchmark and mark with the lowest ranking the systems that fail to give an output. For*scenario B*, we evaluate the differences of average rankings only on the datasets all systems successfully analyzed. To adequately explore the results and look for statistically significant differences in the rankings of the systems, we use the autorank Python package. Because our results do not follow a normal distribution, autorank uses the non-parametric Friedman's test[50] as omnibus test to decide if there are significant differences between the median values of the results. Additionally, it uses the post-hoc Nemenyi test [67] to infer which differences are statistically significant. This holds if the difference of the mean ranks between systems is greater than the critical distance (CD) of the Nemenyi test. For both tasks, the CD value is 0.61.

We examine the results of each task, starting with the binary classification. For both scenarios (Figure4.4) we notice there is no clear winner, as Autosklearn, GAMA, h2o.automl and JADBio have no significant differences in their average ranks and, therefore, in their performance. Additionally, we observe Autosklearn and GAMA manage to be statistically significantly better than TPOT, which is the lowest performing system.

Moving to the regression task, the differences between the results across the 2

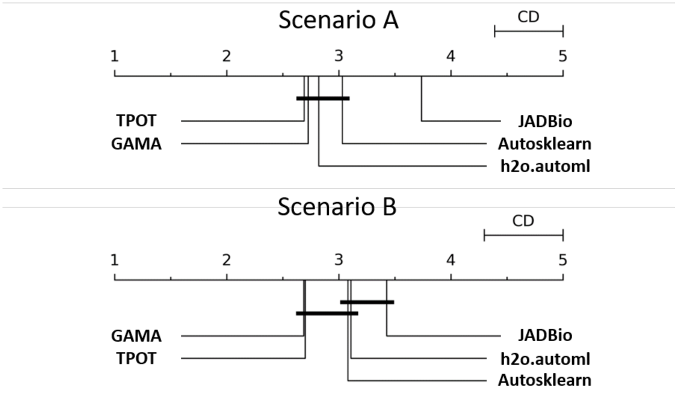


Figure 4.5: Regression model quality rankings comparison. **Top**: Scenario A - All participating datasets (Union) **Bottom**: Scenario B - Completed datasets across all systems (Intersection). The systems connected with a horizontal line have no significant differences in their average rankings, at a statistical significance level of 0.05. The CD is the minimum difference of average rankings between systems required to consider their performance significantly distinct. There is no clear winner among the participating systems in the regression ML task.

scenarios are noticeable. In *Scenario A*, all systems but JADBio achieve indistinguishable performance and have significantly better performance than the latter. This difference is expected, as JADBio has the lowest completion rate on the regression task (78%, Table 4.3) and by adding these cases to the overall evaluation, results in a drop of its average ranking from 3.43 to 3.66. In *Scenario B*, GAMA and TPOT score significant wins over JADBio and have comparable results with Autosklearn and TPOT.
The results of these 2 scenarios cannot provide us with a significant winner for either of these tasks. Still, we can use them to extract useful information. In the binary classification task, 4 out of 5 systems have comparable performance, with Autosklearn and GAMA able to achieve a significant win over TPOT. Regarding the regression task, GAMA and TPOT are statistically significantly better than JADBio in both scenarios, and the low completion rate of JADBio leads to it being statistically significantly worse than the rest of the systems in *Scenario A*. When it comes to the most stable system in terms of ranking across both ML tasks, GAMA consistently secures first or second position across both tasks. On the other hand, TPOT struggles with the binary classification and achieves top-2 positions in the regression. Moving to JADBio, it fails often in regression, but performs well in binary classification. Lastly, looking at Autosklearn and h2o.automl rankings, even though they might differ between the tasks, this deviation is not statistically significant. It is important to note, the best performing systems have an average rank of around 2.5. This means that no system has a clear edge over the rest when it comes to hold-out performance, and therefore no clear victor can be decided.

Table 4.5: Examining the trends of performance ranking with dataset metalevel features. We performed spearman correlation to detect linear trends. Pvalues are denoted based on different significance levels (∗1: 0.05, ∗2: 0.01, ∗3: 0.001) Most automl systems show a reduction of their bias when the sample size increases and for smaller samples to features ratio. Additionally, there is a trend regarding the number of missing values and the estimation bias in the regression results.

Task	Binary classifcation					Regression				
Metafeatures	Autosklearn	GAMA	h2o.automl	JADBio	TPOT	Autosklearn	GAMA	h2o.automl	JADBio	TPOT
Samples size	-0.32^{*2}	0.22^{*1}	-0.11	0.14	-0.06	0.02	-0.02	-0.39^{*3}	0.46^{*3}	-0.08
Features size	-0.11	0.06	-0.17	0.07	0.12	-0.14	0.05	-0.15	0.25^{*1}	-0.01
Continuous features (#)	-0.039	0.02	-0.09	0.10	0.05	-0.07	0.02	-0.19	0.26^{*1}	-0.06
Discrete features (#)	-0.02	-0.01	0.03	-0.13	0.11	-0.22^{*1}	0.21^{*1}	0.19	-0.18	0.21^{*1}
Discrete features (%)	-0.01	-0.04	0.08	-0.12	0.05	-0.22^{*1}	0.20^{*1}	0.21^{*1}	-0.20^{*1}	0.21^{*1}
Missing values (%)	0.16	-0.12	-0.08	-0.16	0.14	-0.22^{*1}	0.25^{*1}	0.12	-0.21^{*1}	0.30^{*2}
Samples to Features ratio	-0.27^{*2}	0.23^{*1}	-0.18	0.25^{*1}	-0.13	0.12	-0.10	-0.25^{*1}	0.28^{*2}	-0.08
Majority class (%)	0.10	-0.05	0.07	-0.26^{*2}	0.13					

This allows the AutoML users to use additional criteria (e.g. complexity of the final model, available feature preprocessing methods, level of automation etc.) when selecting which system to use for their analysis.

4.5.4 Hold-out performance correlation to data metafeatures

We are interested in examining whether our tracked data metafeatures have a correlation with the average rankings of the participating AutoML systems. This could be an indication of the type of data each system excels on, based on its characteristics. Starting with the binary classification task, the *Samples to Features ratio* is the most informative metafeature. GAMA and JADBio has lower performance as the ratio's value increased, whereas Autosklearn achieved significantly better results. Lastly, *Majority class percentage* is another metafeature correlated with the final ranking of JADBio systems, as it performs significantly better on highly imbalanced datasets.
In the regression results, we have similar findings regarding the correlation of the systems' bias to the data characteristics. Starting with Autosklearn, it achieves a higher ranking as the Number of Discrete features and Missing values increase. Continuing, h2o.automl has a higher ranking on bigger Sample size datasets and together with GAMA and TPOT, they achieve a lower ranking, as the Number of Discrete features increases. Furthermore, GAMA and TPOTb●●s ranking lowers as the Missing values percentage increases. Lastlym focusing of JADBio, its ranking is higher when the Number of Missing values and Discrete features increase. However, it becomes lower, as the Volume of data increases, a result directly related to the low completion rate of the system in the bigger regression analyses.

4.6 Limitations and future work

It is natural this study comes with a set of limitations but the methodology we suggest can easily be extended to cover other capabilities of AutoML systems as well. To begin with, the scope of our study is limited. We considered only 2 supervised tasks (binary classification and regression), completely leaving out other widely used tasks, such as multiclass classification or time-to-event, and the unsupervised tasks. Additionally, we decided to focus only continuous, discrete, independent and identically distributed tabular data, not including other types, such as time-series, or different formats, like images. We also need to extend the list of data in our benchmark by including more datasets, to cover more types of problems and tasks. The same goes for the selected systems; every year AutoML is increasing in popularity, with more ideas and systems being implemented for both open source and commercial use, so we should include as many as possible in the future. Another limitation is about the validity of our results. As systems continuously evolve, the results of this study will eventually become obsolete. To be able to keep them up-to-date, we must automate the evaluation procedure, similar to what AutoML benchmark[18] has done, and slowly add other evaluation dimensions, such as the complexity of a system's output.

4.7 Summary and Conclusion

This is the first work evaluating the estimation bias of multiple AutoML systems and looking for possible trends of the bias and hold - out performance with specific data characteristics. What we propose is, a methodology for evaluating AutoML systems. We have created rules and guidelines on how to select datasets in order to cover multiple different cases and created our own expandable benchmark. Moreover, we presented a detailed protocol to conduct our evaluation, covering all selected performance metrics, as well as the statistical tests required. Lastly, our study is the first to include a commercial system.

By performing this evaluation, we have come across multiple interesting findings. Regarding the average bias, most AutoML systems overestimate the performance of their output regardless of the ML task. JADBio is the only system statistically significantly giving conservative results on both tasks. Moreover, we observe that for most systems (Autosklearn, GAMA, h2o.automl, TPOT), the bias is correlated to the characteristics of the data. JADBio, however, shows no trend of its estimations to any metafeature. An important question that needs to be asked, is why it this happening. During an analysis, AutoML systems try hundreds to thousands of pipelines in order to find the best performing one. This leads to the multiple Induction problem[33], also known as winners curse[65] in bidding. When trying numerous pipelines, the cross-validated accuracy of the winning pipeline is overestimated. Most AutoML systems do not take into account this bias, and therefore, show overestimated results. This is an alarming finding, since the majority of the participating systems can potentially misinform their users about the true predictive performance of their output. This can escalate out of proportions, as these systems tend to be used mostly by non-expert ML users, meaning they have little if any ML knowledge and cannot easily spot these estimation errors. When AutoML systems are being used in domains, such as biomedicine or business, overestimating the true performance can be dangerous and disruptive. To address this issue, JADBio uses the boostrap bias correction[71] (BBC) method before outputting its estimation. However, this method comes with its own set of limitations, as BBC works only with static HPO search strategies, which possibly affects the quality of the outputted model in terms of predictive performance. Moving to the hold out performance comparison, there is no statistically significant winner. Moreover, We found that no system dominates both ML tasks. However GAMA and h2o.automl seem to be the most consistent in terms of average ranking. Lastly, the predictive performance of all systems is correlated to some data characteristics.

The take home message for AutoML users is that they should not trust the performance estimations of AutoML systems, excluding JADBio. Also, this work can help them decide which automl system fits best on their needs, depending on the characteristics of the data they want to analyze.

Chapter 5

Acknowledgments

I would like to thank my professor, Ioannis Tsamardinos, for his guidance and mentoring over the last 4 years. Also, Vassilis Christophides for always providing useful feedback and sharing his knowledge when working together. I would to express my sincere appreciation to Joaquin Vanschoren for being in my Master's presentation committee. Of course, I am grateful for the support of my family, girlfriend and friends all these years. Moreover, I feel lucky I met and worked with so many bright individuals in the MensXMachina research group. Last but not least, I would like to thank Dr. Alejandro Salinger and Dr. Eric Simon from SAP for their ideas and insights when we first designed the contents of this project, during my internship at SAP.
The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 617393
This work was supported by computational time granted from the National Infrastructures for Research and Technology S.A. (GRNET S.A.) in the National HPC facility - ARIS - under project pa191203

Bibliography

[1] Auger.AI. *Auger.AI*, 2019. `https://auger.ai/`.

[2] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

[3] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc., 2011.

[4] BigML. *BigML*, 2012. `https://bigml.com/`.

[5] Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael Mantovani, Jan van Rijn, and Joaquin Vanschoren. Openml benchmarking suites and the openml100. 08 2017.

[6] A. A. Borkowski, C. P. Wilson, S. A. Borkowski, L. B. Thomas, L. A. Deland, S. J. Grewe, and S. M. Mastorides. Comparing Artificial Intelligence Platforms for Histopathologic Cancer Diagnosis. *Fed Pract*, 36(10):456–463, Oct 2019.

[7] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[8] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[9] Yi-Wei Chen, Qingquan Song, and Xia Hu. Techniques for automated machine learning. *CoRR*, abs/1907.08908, 2019.

[10] Spark Cognition. *Darwin*, 2019. `https://www.sparkcognition.com/product/darwin/`.

[11] Mark W. Craven and Jude W. Shavlik. Extracting tree-structured representations of trained networks. In *Proceedings of the 8th International Conference on Neural Information Processing Systems*, NIPS'95, pages 24–30, Cambridge, MA, USA, 1995. MIT Press.

[12] Mengnan Du, Ninghao Liu, and Xia Hu. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77, 2019.

[13] Amitai Etzioni and Oren Etzioni. Designing ai systems that obey our laws and values. *Commun. ACM*, 59(9):29–31, August 2016.

[14] Matthias Feurer and Frank Hutter. Towards further automation in automl. In *ICML 2018 AutoML Workshop*, July 2018.

[15] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2962–2970. Curran Associates, Inc., 2015.

[16] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[17] Krishna Gade, Sahin Cem Geyik, Krishnaram Kenthapadi, Varun Mithal, and Ankur Taly. Explainable ai in industry. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, pages 3203–3204, New York, NY, USA, 2019. Association for Computing Machinery.

[18] Pieter Gijsbers, Erin LeDell, Janek Thomas, Sébastien Poirier, Bernd Bischl, and Joaquin Vanschoren. An open source automl benchmark. *CoRR*, abs/1907.00909, 2019.

[19] Pieter Gijsbers and Joaquin Vanschoren. GAMA: Genetic automated machine learning assistant. *Journal of Open Source Software*, 4(33):1132, jan 2019.

[20] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

[21] Google. *AutoML Tables*, 2019. `https://cloud.google.com/automl-tables/`.

[22] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[23] Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michèle Sebag, Alexander Statnikov, Wei-Wei Tu, and Evelyne Viegas. *Analysis of the AutoML Challenge Series 2015–2018*, pages 177–219. Springer International Publishing, Cham, 2019.

[24] H2O.ai. *H2O AutoML*, June 2017. H2O version 3.30.0.1.

[25] Tuomas Halvari, Jukka K Nurminen, and Tommi Mikkonen. Testing the robustness of automl systems. *arXiv preprint arXiv:2005.02649*, 2020.

[26] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction.* Springer Science & Business Media, 2009.

[27] Steffen Herbold. Autorank: A python package for automated ranking of classifiers. *Journal of Open Source Software*, 5(48):2173, 2020.

[28] Bregje Holleman, Naomi Kamoen, André Krouwel, Jasper van de Pol, and Claes de Vreese. Positive vs. negative: The impact of question polarity in voting advice applications. *PLOS ONE*, 11(10):1–17, 10 2016.

[29] Linwei Hu, Jie Chen, Vijayan Nair, and Agus Sudjianto. Locally interpretable models and effects based on supervised partitioning (lime-sup). 06 2018.

[30] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In Carlos A. Coello Coello, editor, *Learning and Intelligent Optimization*, pages 507–523, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[31] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors. *Automated Machine Learning: Methods, Systems, Challenges.* Springer, 2018. In press, available at http://automl.org/book.

[32] IBM. *IBM Watson Studio*, 2015. `https://www.ibm.com/watson`.

[33] David D. Jensen and Paul R. Cohen. Multiple comparisons in induction algorithms. *Machine Learning*, 38(3):309–338, Mar 2000.

[34] Haifeng Jin, Qingquan Song, and Xia Hu. Auto-keras: An efficient neural architecture search system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1946–1956. ACM, 2019.

[35] Nikolay Laptev, Saeed Amizadeh, and Ian Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1939–1947, New York, NY, USA, 2015. Association for Computing Machinery.

[36] Trang T Le, Weixuan Fu, and Jason H Moore. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, 36(1):250–256, 2020.

[37] Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018.

[38] James R. Lewis. The system usability scale: Past, present, and future. *International Journal of Human-Computer Interaction*, 34(7):577–590, 2018.

[39] Rensis Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[40] Huan Liu and Hiroshi Motoda. *Feature Selection for Knowledge Discovery and Data Mining.* Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[41] Scott Lundberg, Gabriel Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. 02 2018.

[42] Jorge G Madrid, Hugo Jair Escalante, Eduardo F Morales, Wei-Wei Tu, Yang Yu, Lisheng Sun-Hosoya, Isabelle Guyon, and Michèle Sebag. Towards AutoML in the presence of Drift: first results. In *Workshop AutoML 2018 @ ICML/IJCAI-ECAI*, Stockholm, Sweden, July 2018. Pavel Brazdil, Christophe Giraud-Carrier, and Isabelle Guyon.

[43] Microsoft. *Azure Machine Learning Studio*, 2015. `https://studio.azureml.net/`.

[44] Piero Molino, Yaroslav Dudin, and Sai Sumanth Miryala. Ludwig: a type-based declarative deep learning toolbox, 2019.

[45] Christoph Molnar. *Interpretable Machine Learning.* 2019. `https://christophm.github.io/interpretable-ml-book/`.

[46] Besmira Nushi, Ece Kamar, and Eric Horvitz. Towards accountable AI: hybrid human-machine analyses for characterizing system failure. In *Proceedings of the Sixth AAAI Conference on Human Computation and Crowdsourcing, HCOMP 2018, Zürich, Switzerland, July 5-8, 2018*, pages 126–135, 2018.

[47] Meghana Padmanabhan, Pengyu Yuan, Govind Chada, and Hien Van Nguyen. Physician-friendly machine learning: A case study with cardiovascular disease risk prediction. In *Journal of clinical medicine*, 2019.

[48] Magnus Palmblad, Anna-Lena Lamprecht, Jon Ison, and Veit Schwämmle. Automated workflow composition in mass spectrometry-based proteomics. *Bioinformatics*, 35(4):656–664, 2018.

[49] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron

Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

[50] D. Pereira, Anabela Afonso, and Fátima Medeiros. Overview of friedman's test and post-hoc analysis. *Communications in Statistics - Simulation and Computation*, 44:2636–2653, 11 2015.

[51] J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms.* MIT Press, Cambridge, MA, USA, 2017.

[52] Fábio Pinto, Marco O. P. Sampaio, and Pedro Bizarro. Automatic model monitoring for data streams. *CoRR*, abs/1908.04240, 2019.

[53] RapidMiner. *RapidMiner*, 2006. `https://rapidminer.com/`.

[54] Sarunas J Raudys, Anil K Jain, et al. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on pattern analysis and machine intelligence*, 13(3):252–264, 1991.

[55] Payam Refaeilzadeh, Lei Tang, and Huan Liu. *Cross-Validation*, pages 532–538. Springer US, Boston, MA, 2009.

[56] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM.

[57] Vraj Shah and Arun Kumar. The ml data prep zoo: Towards semi-automatic data preparation for ml. In *Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning*, DEEM'19, New York, NY, USA, 2019. Association for Computing Machinery.

[58] Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Ferdinand Kossmann, Philipp Eichmann, Yeounoh Chung, Carsten Binnig, Eli Upfal, and Tim Kraska. Democratizing data science through interactive curation of ml pipelines. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, pages 1171–1188, New York, NY, USA, 2019. ACM.

[59] Radwa El Shawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. *CoRR*, abs/1906.02287, 2019.

[60] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[61] Micah J. Smith, Carles Sala, James Max Kanter, and Kalyan Veeramachaneni. The machine learning bazaar: Harnessing the ML ecosystem for effective system development. *CoRR*, abs/1905.08942, 2019.

[62] Alexander Statnikov, Ioannis Tsamardinos, Yerbolat Dosbayev, and Constantin F Aliferis. Gems: a system for automated cancer diagnosis and biomarker discovery from microarray gene expression data. *International journal of medical informatics*, 74(7-8):491–503, 2005.

[63] Student. The probable error of a mean. *Biometrika*, 6:33–57.

[64] Sean Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72, 09 2017.

[65] Richard H. Thaler. Anomalies: The winner's curse. *Journal of Economic Perspectives*, 2(1):191–202, March 1988.

[66] Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Auto-weka: Automated selection and hyper-parameter optimization of classification algorithms. *CoRR, abs/1208.3719*, 2012.

[67] Ene-Margit Tiit. Nonparametric statistical methods. myles hollander and douglas a. wolfe, wiley, chichester, 1999. *Statistics in Medicine*, 19(10):1386–1388, 2000.

[68] A. Truong, A. Walters, J. Goodsitt, K. Hines, C. B. Bruss, and R. Farivar. Towards automated machine learning: Evaluation and comparison of automl approaches and tools. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1471–1479, 2019.

[69] Ioannis Tsamardinos and Constantin F Aliferis. Towards principled feature selection: relevancy, filters and wrappers. In *AISTATS*, 2003.

[70] Ioannis Tsamardinos, Paulos Charonyktakis, Kleanthi Lakiotaki, Giorgos Borboudakis, Jean Zenklusen, Hartmut Juhl, Ekaterini Chatzaki, and Vincenzo Lagani. *Just Add Data: Automated Predictive Modeling and BioSignature Discovery*, 05 2020.

[71] Ioannis Tsamardinos, Elissavet Greasidou, and Giorgos Borboudakis. Bootstrapping the out-of-sample predictions for efficient and accurate cross-validation. *Machine learning*, 107(12):1895–1922, 2018.

[72] Lukas Tuggener, Mohammadreza Amirian, Katharina Rombach, Stefan Lörwald, Anastasia Varlet, Christian Westermann, and Thilo Stadelmann. Automated machine learning in practice: State of the art and recent results. *CoRR*, abs/1907.08392, 2019.

[73] Joaquin Vanschoren, Jan van Rijn, Bernd Bischl, and LuG•s Torgo. Openml: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15:49–60, 12 2013.

[74] Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, 7(1):91, 2006.

[75] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, Jun 2002.

[76] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[77] Ziqiao Weng. From conventional machine learning to AutoML. *Journal of Physics: Conference Series*, 1207:012015, apr 2019.

[78] Quanming Yao, Mengshuo Wang, Hugo Jair Escalante, Isabelle Guyon, Yi-Qi Hu, Yu-Feng Li, Wei-Wei Tu, Qiang Yang, and Yang Yu. Taking human out of learning applications: A survey on automated machine learning. *CoRR*, abs/1810.13306, 2018.

[79] Marc-André Zöller and Marco F. Huber. Survey on automated machine learning. *CoRR*, abs/1904.12054, 2019.

Chapter 6

Appendix

Table 6.1: Selected binary classification datasets for our benchmark. We mention the *Name, ID* and other available metafeatures from OpenML.org

	Filename	ID	Number of Samples	Number of Features	Continuous Features	Categorical Features	Missing values percentage	Majority class percentage	Minority class size	Samples to Features ratio
0	kr-vs-kp	3	3196	37	0	36	0.000000	52.221527	1527	86.378378
1	mushroom	24	8124	23	0	22	1.327254	51.797144	3916	353.217391
2	colic	25	368	27	7	19	19.394122	63.043478	136	13.629630
3	heart-h	51	294	14	6	7	18.999028	63.945578	106	21.000000
4	hepatitis	55	155	20	6	13	5.387097	79.354839	32	7.750000
5	vote	56	435	17	0	16	5.300879	61.379310	168	25.588235
6	molecular-biology_promoters	164	106	59	0	58	0.000000	50.000000	53	1.796610
7	oil_spill	311	937	50	49	0	0.000000	95.624333	41	18.740000
8	yeast_ml8	316	2417	117	103	13	0.000000	98.593297	34	20.658120
9	vehicle_sensIT	357	98528	101	100	0	0.000000	50.000000	49264	975.524753
10	ailerons	734	13750	41	40	0	0.000000	57.614545	5828	335.365854
11	cpu_small	735	8192	13	12	0	0.000000	69.763184	2477	630.153846
12	fri_c4_500_100	742	500	101	100	0	0.000000	56.600000	217	4.950495
13	meta	757	528	22	19	2	4.338843	89.772727	54	24.000000
14	pbc	810	418	19	10	8	15.600604	55.023923	188	22.000000
15	house_16H	821	22784	17	16	0	0.000000	70.400281	6744	1340.235294
16	bank32nh	833	8192	33	32	0	0.000000	68.957520	2543	248.242424
17	stock	841	950	10	9	0	0.000000	51.368421	462	95.000000
18	wind	847	6574	15	14	0	0.000000	53.255248	3073	438.266667
19	tecator	851	240	125	124	0	0.000000	57.500000	102	1.920000
20	boston	853	506	14	12	1	0.000000	58.695652	209	36.142857
21	colleges_aaup	897	1161	17	13	3	1.297056	70.025840	348	68.294118
22	fried	901	40768	11	10	0	0.000000	50.105475	20341	3706.181818
23	spectrometer	954	531	103	100	2	0.000000	89.642185	55	5.155340
24	autos	975	205	26	15	10	1.106942	67.317073	67	7.884615
25	JapaneseVowels	976	9961	15	14	0	0.000000	83.796808	1614	664.066667
26	mfeat-factors	978	2000	217	216	0	0.000000	90.000000	200	9.216590
27	anneal	989	898	39	6	32	63.317343	76.169265	214	23.025641
28	kdd_ipums_la_97-small	993	7019	61	33	27	10.233114	63.043169	2594	115.065574
29	hypothyroid	1000	3772	30	7	22	5.358784	92.285260	291	125.733333
30	arrhythmia	1017	452	280	206	73	0.322377	54.203540	207	1.614286
31	page-blocks	1021	5473	11	10	0	0.000000	89.767952	560	497.545455
32	gina_agnostic	1038	3468	971	970	0	0.000000	50.836217	1705	3.571576
33	hiva_agnostic	1039	4229	1618	1617	0	0.000000	96.476708	149	2.613721
34	kc1-top5	1045	145	95	94	0	0.000000	94.482759	8	1.526316
35	pc4	1049	1458	38	37	0	0.000000	87.791495	178	38.368421
36	pc2	1069	5589	37	36	0	0.000000	99.588477	23	151.054054
37	KDDCup09_churn	1112	50000	231	192	38	69.473177	92.656000	3672	216.450217
38	CastMetal1	1447	327	38	37	0	0.000000	87.155963	42	8.605263
39	bank-marketing	1461	45211	17	7	9	0.000000	88.301520	5289	2659.470588
40	lsvt	1484	126	311	310	0	0.000000	66.666667	42	0.405145
41	madelon	1485	2600	501	500	0	0.000000	50.000000	1300	5.189621
42	nomao	1486	34465	119	89	29	0.000000	71.437690	9844	289.621849
43	ozone-level-8hr	1487	2534	73	72	0	0.000000	93.685872	160	34.712329
44	parkinsons	1488	195	23	22	0	0.000000	75.384615	48	8.478261
45	qsar-biodeg	1494	1055	42	41	0	0.000000	66.255924	356	25.119048
46	ringnorm	1496	7400	21	20	0	0.000000	50.486486	3664	352.380952
47	autoUniv-au1-1000	1547	1000	21	20	0	0.000000	74.100000	259	47.619048
48	adult	1590	48842	15	6	8	0.882437	76.071823	11687	3256.133333
49	Dexter	4136	600	20001	20000	0	0.000000	50.000000	300	0.029999
50	cylinder-bands	6332	540	40	18	21	4.625000	57.777778	228	13.500000
51	dresses-sales	23381	500	13	1	11	12.846154	58.000000	210	38.461538
52	higgs	23512	98050	29	28	0	0.000317	52.857726	46223	3381.034483
53	SpeedDating	40536	8378	123	59	63	1.782834	83.528288	1380	68.113821
54	enron	40590	1702	1054	0	1053	0.000000	98.472385	26	1.614801
55	image	40592	2000	140	135	4	0.000000	79.550000	409	14.285714
56	reuters	40594	2000	250	243	6	0.000000	58.450000	831	8.000000
57	scene	40595	2407	300	294	5	0.000000	82.260075	427	8.023333
58	GAMETES_Epistasis_2-Way_1000atts_0.4H_EDM-1_ED...	40645	1600	1001	0	1000	0.000000	50.000000	800	1.598402
59	clean2	40666	6598	169	168	0	0.000000	84.586238	1017	39.041420
60	threeOf9	40690	512	10	0	9	0.000000	53.515625	238	51.200000
61	churn	40701	5000	21	16	4	0.000000	85.860000	707	238.095238
62	Satellite	40900	5100	37	36	0	0.000000	98.529412	75	137.837838
63	Speech	40910	3686	401	400	0	0.000000	98.345090	61	9.192020
64	Internet-Advertisements	40978	3279	1559	3	1555	0.000000	86.001830	459	2.103271
65	Australian	40981	690	15	6	8	0.000000	55.507246	307	46.000000
66	jungle_chess_2pcs_endgame_rat_rat	41005	3660	47	20	26	0.000000	56.147541	1605	77.872340
67	gisette	41026	7000	5001	5000	0	0.000000	50.000000	3500	1.399720
68	APSFailure	41138	76000	171	170	0	8.300208	98.190789	1375	444.444444
69	christine	41142	5418	1637	1599	37	0.000000	50.000000	2709	3.309713
70	jasmine	41143	2984	145	8	136	0.000000	50.000000	1492	20.579310
71	gina	41158	3153	971	970	0	0.000000	50.840469	1550	3.247168
72	kick	41162	72983	33	14	18	6.197832	87.701245	8976	2211.606061
73	USPS	41964	1424	257	256	0	0.000000	50.280899	708	5.540856
74	isolet	41966	600	618	617	0	0.000000	50.000000	300	0.970874
75	cnae-9	41967	240	857	856	0	0.000000	50.000000	120	0.280047
76	semeion	41973	319	257	256	0	0.000000	50.470219	158	1.241245
77	compas-two-years	42193	5278	14	7	6	0.000000	52.955665	2483	377.000000
78	fri_c4_250_100	834	250	101	100	0	0.000000	56.000000	110	2.475248
79	mfeat-karhunen	1020	2000	65	64	0	0.000000	90.000000	200	30.769231
80	Click_prediction_small	1217	149639	12	11	0	0.000000	95.529240	6690	12469.916670
81	Amazon_employee_access	4135	32769	10	0	9	0.000000	94.210992	1897	3276.900000
82	anthracyclineTaxaneChemotherapy	1085	159	61360	61359	0	0.000000	59.748428	64	0.002591
83	AP_Breast_Prostate	1122	413	10937	10936	0	0.000000	83.292978	69	0.037762
84	OVA_Omentum	1139	1545	10937	10936	0	0.000000	95.016181	77	0.141264
85	AP_Omentum_Kidney	1147	337	10937	10936	0	0.000000	77.151335	77	0.030813
86	ada	41156	4147	49	48	0	0.000000	75.186882	1029	84.632653
87	eucalyptus	990	736	20	14	5	3.043478	70.923913	214	36.800000
88	SPECTF	1600	267	45	44	0	0.000000	79.400749	55	5.933333
89	tokyo1	40705	959	45	42	2	0.000000	63.920751	346	21.311111
90	triazines	788	186	61	60	0	0.000000	58.602151	77	3.049180
91	segment	958	2310	20	19	0	0.000000	85.714286	330	115.500000
92	analcatdata_reviewer	1008	379	9	0	8	40.105541	56.992084	163	42.111111
93	climate-model-simulation-crashes	1467	540	21	20	0	0.000000	91.481481	46	25.714286
94	MiniBooNE	41150	130064	51	50	0	0.000000	71.937661	36499	2550.274510
95	zoo	965	101	18	1	16	0.000000	59.405941	41	5.611111
96	Dorothea	4137	1150	100001	100000	0	0.000000	90.260870	112	0.011500
97	tic-tac-toe	50	958	10	0	10	0.000000	65.344468	332	95.800000
98	wdbc	1510	569	31	30	1	0.000000	62.741652	212	18.354839
99	PieChart1	1451	705	38	37	1	0.000000	91.347518	61	18.552632

Table 6.2: Selected regression datasets for our benchmark. We mention the *Name*, *ID* and other available metafeatures from OpenML.org

	Filename	ID	Number of Samples	Number of Features	Continuous Features	Categorical Features	Missing values percentage	Samples to Features ratio
0	2dplanes	215	40768	11	10	0	0.000000	3706.181818
1	a3a	1424	32561	124	123	0	0.000000	262.588710
2	a9a	1430	48842	124	123	0	0.000000	393.887097
3	Ailerons	296	13750	41	40	0	0.000000	335.365854
4	analcatdata_gsssexsurvey	506	159	10	4	5	0.377358	15.900000
5	analcatdata_ncaa	521	120	20	3	17	0.000000	6.000000
6	analcatdata_supreme	504	4052	8	7	0	0.000000	506.500000
7	auto_price	195	159	16	14	1	0.000000	9.937500
8	autoMpg	196	398	8	4	3	0.188442	49.750000
9	bank32nh	558	8192	33	32	0	0.000000	248.242424
10	bank8FM	572	8192	9	8	0	0.000000	910.222222
11	benzo32	434	195	33	32	0	0.000000	5.909091
12	bodyfat	560	252	15	14	0	0.000000	16.800000
13	boston_corrected	543	506	21	17	3	0.000000	24.095238
14	breastTumor	224	286	10	1	8	0.314685	28.600000
15	cholesterol	204	303	14	6	7	0.141443	21.642857
16	chscase_census5	670	400	8	7	0	0.000000	50.000000
17	cleveland	194	303	14	6	7	0.141443	21.642857
18	coil2000	298	9822	86	85	0	0.000000	114.209302
19	connect-4	1591	67557	127	126	0	0.000000	531.944882
20	CPMP-2015-regression	41700	2108	27	23	2	0.000000	78.074074
21	CPMP-2015-runtime-regression	41928	2108	24	22	1	0.000000	87.833333
22	cps_85_wages	534	534	11	3	7	0.000000	48.545455
23	cpu	561	209	8	6	1	0.000000	26.125000
24	cpu_act	197	8192	22	21	0	0.000000	372.363636
25	cpu_small	227	8192	13	12	0	0.000000	630.153846
26	crimecommunitynums	41968	1994	127	126	0	15.480299	15.700787
27	dataset_sales	42183	10738	15	14	0	0.000000	715.866667
28	dataset-autoHorse_fixed	42224	201	69	68	1	0.000000	2.913043
29	debutanizer	23516	2394	8	7	0	0.000000	299.250000
30	delta_elevators	198	9517	7	6	0	0.000000	1359.571429
31	Diabetes(scikit-learn)	41514	442	11	10	0	0.000000	40.181818
32	diamonds	42225	53940	10	6	3	0.000000	5394.000000
33	echoMonths	222	130	10	6	3	7.461538	13.000000
34	elevators	216	16599	19	18	0	0.000000	873.631579
35	fishcatch	232	158	8	5	2	6.882911	19.750000
36	fri_c3_1000_10	608	1000	11	10	0	0.000000	90.909091
37	fri_c4_250_100	580	250	101	100	0	0.000000	2.475248
38	fried	564	40768	11	10	0	0.000000	3706.181818
39	GeographicalOriginalofMusic	4544	1059	118	117	0	0.000000	8.974576
40	german.numer	1436	1000	25	24	0	0.000000	40.000000
41	HappinessRank_2015	40916	158	12	9	2	0.000000	13.166667
42	higgs	4532	98050	29	28	0	0.000317	3381.034483
43	house_16H	574	22784	17	16	0	0.000000	1340.235294
44	house_8L	218	22784	9	8	0	0.000000	2531.555556
45	houses	537	20640	9	8	0	0.000000	2293.333333
46	hungarian	231	294	14	6	8	18.999028	21.000000
47	ICU	1097	200	21	20	0	0.000000	9.523810
48	ilpd-numeric	41943	583	11	10	0	0.000000	53.000000
49	kc1-numeric	1070	145	95	94	0	0.000000	1.526316
50	kdd_coil_3	570	316	12	8	3	1.476793	26.333333
51	kin8nm	189	8192	9	8	0	0.000000	910.222222
52	LoanDefaultPrediction	-	105471	771	764	6	0.966519	136.797665
53	lowbwt	203	189	10	2	7	0.000000	18.900000
54	lungcancer_shedden	1245	442	24	20	3	0.000000	18.416667
55	mauna-loa-atmospheric-co2	41187	2225	7	5	1	0.000000	317.857143
56	MIP-2016-PAR10-regression	41938	1090	145	143	1	0.000000	7.517241
57	mnist_rotation	41065	62000	785	784	0	0.000000	78.980892
58	Moneyball	41021	1232	15	8	6	19.480519	82.133333
59	mtp	405	4450	203	202	0	0.000000	21.921182
60	mv	344	40768	11	7	3	0.000000	3706.181818
61	NewFuelCar	41506	36203	18	17	0	1.376651	2011.277778
62	nki70.arff	1228	144	77	72	4	0.000000	1.870130
63	no2	547	500	8	7	0	0.000000	62.500000
64	OnlineNewsPopularity	4545	39644	61	59	1	0.000000	649.901639
65	ozone_level	301	2536	73	0	72	0.000000	34.739726
66	parkinson-speech-uci	42176	756	754	753	0	0.000000	1.002653
67	pbc	200	418	20	13	6	12.356459	20.900000
68	pbcseq	516	1945	19	12	6	3.065891	102.368421
69	pharynx	213	195	12	1	11	0.085470	16.250000
70	places	509	329	10	8	1	0.000000	32.900000
71	pol	201	15000	49	48	0	0.000000	306.122449
72	puma32H	308	8192	33	32	0	0.000000	248.242424
73	pwLinear	229	200	11	10	0	0.000000	18.181818
74	QSAR-TID-10541	3169	151	1026	1024	2	0.000000	0.147173
75	QSAR-TID-10849	3079	1580	1026	1024	2	0.000000	1.539961
76	QSAR-TID-13004	3789	692	1026	1024	1	0.000000	0.674464
77	QSAR-TID-17061	3267	152	1026	1024	2	0.000000	0.148148
78	QSAR-TID-194	3991	5188	1026	1024	1	0.000000	5.056530
79	QSAR-TID-234	3081	2145	1026	1024	2	0.000000	2.090643
80	QSAR-TID-30008	3183	837	1026	1024	2	0.000000	0.815789
81	rmftsa_ladata	666	508	11	10	0	0.000000	46.181818
82	SAT11-HAND-runtime-regression	41980	4440	117	115	1	5.226380	37.948718
83	satellite_image	294	6435	37	36	0	0.000000	173.918919
84	SensIT-Vehicle-Combined	1593	98528	101	100	0	0.000000	975.524753
85	sensory	546	576	12	0	11	0.000000	48.000000
86	sleuth_case2002	665	147	7	2	5	0.000000	21.000000
87	splice	46	3175	61	60	0	0.000000	52.049180
88	stock	223	950	10	9	0	0.000000	95.000000
89	svmguide3	1589	1243	23	22	0	0.000000	54.043478
90	SWD	1028	1000	11	10	0	0.000000	90.909091
91	tecator	505	240	125	124	0	0.000000	1.920000
92	Titanic	41265	1307	8	7	0	0.000000	163.375000
93	topo_2_1	422	8885	267	266	0	0.000000	33.277154
94	veteran	497	137	8	3	4	0.000000	17.125000
95	w1a	1581	49749	301	300	0	0.000000	165.279070
96	wind	503	6574	15	14	0	0.000000	438.266667
97	wine_quality	287	6497	12	11	0	0.000000	541.416667
98	wisconsin	191	194	33	32	1	0.000000	5.878788
99	yprop_4_1	416	8885	252	251	0	0.000000	35.257937

Table 6.3: Binary classification experiments results. The train results correspond to the performance estimation after the training phase, while test results correspond to the true hold-out performance.

	filename	autosklearn train	autosklearn test	gama train	gama test	h2o train	h2o test	jad train	jad test	tpot train	tpot test
0	ada	0.920560	0.909597	0.920115	0.915375	0.912641	0.910055	0.907748	0.912602	0.919111	0.913669
1	adult	0.926999	0.923554	0.919773	0.915515	0.921191	0.926685	0.919565	0.915365	0.923715	0.926818
2	ailerons	0.956998	0.957725	0.955223	0.955605	0.957784	0.958089	0.953299	0.952743	0.957765	0.957696
3	Amazon_employee_access	0.826967	0.830619	0.827652	0.841161	0.833975	0.825348	0.821252	0.843222	0.841868	0.822666
4	analcatdata_reviewer	0.847737	0.632057	0.735311	0.628557	NaN	NaN	0.679569	0.627992	0.705676	0.680556
5	anneal	1.000000	0.971389	0.999877	0.993961	0.998606	0.999344	0.999508	0.996475	0.993247	0.988728
6	anthracyclineTaxaneChemotherapy	0.920455	0.465495	0.684508	0.513672	0.527344	0.567819	0.612660	0.542969	0.605000	0.490027
7	AP_Breast_Prostate	1.000000	0.994186	1.000000	0.986877	0.999003	0.999829	0.996992	0.914286	NaN	NaN
8	AP_Omentum_Kidney	1.000000	0.998817	0.997976	0.998817	0.993491	0.996660	0.993714	0.999211	NaN	NaN
9	APSFailure	0.995887	0.991899	0.988389	0.988035	0.988209	0.989656	NaN	NaN	0.991317	0.988708
10	arrhythmia	0.905308	0.834174	0.907096	0.889896	0.861917	0.871261	0.861831	0.887610	0.907127	0.844029
11	Australian	0.947401	0.953696	NaN	NaN	0.957894	0.899016	0.892410	0.954545	0.971625	0.917399
12	autos	0.988142	0.892583	0.920509	0.953112	0.929668	0.930171	0.873020	0.924126	0.977473	0.905797
13	autoUniv-au1-1000	0.803374	0.654719	0.720920	0.711299	0.689189	0.684511	0.664951	0.723888	0.759096	0.705197
14	bank32nh	0.889551	0.887698	0.889355	0.886767	0.890976	0.896874	0.892210	0.889355	0.887004	0.887514
15	bank-marketing	0.932840	0.934783	0.928896	0.932325	0.928961	0.928989	0.914326	0.919703	0.929972	0.929089
16	boston	0.998251	0.928764	0.970250	0.943629	0.949356	0.920625	0.943926	0.937645	0.965796	0.954053
17	CastMetal1	0.866261	0.682984	0.811033	0.773560	0.831835	0.702213	0.576338	0.725608	0.907291	0.681757
18	christine	0.830743	0.825288	0.801588	0.808424	0.817784	0.809574	0.783516	0.789920	0.809825	0.801961
19	churn	0.937177	0.926825	0.919480	0.915786	0.927838	0.918520	0.892017	0.915372	0.936814	0.906182
20	clean2	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.999991	1.000000	1.000000	1.000000
21	Click_prediction_small	0.710333	0.710878	0.702645	0.716382	0.707427	0.699033	0.688422	0.701581	0.708565	0.702065
22	climate-model-simulation-crashes	0.984756	0.870445	0.907587	0.800739	0.876078	0.839289	0.863791	0.881183	0.917255	0.791234
23	cnae-9	1.000000	0.999444	0.999722	0.998889	1.000000	0.999444	0.999150	0.998333	1.000000	0.999722
24	colic	0.975973	0.820487	0.947008	0.872591	0.895030	0.899468	0.919371	0.880958	0.933110	0.861879
25	colleges_aaup	NaN	NaN	0.998570	0.999675	NaN	NaN	0.999234	0.998687	1.000000	0.998981
26	compas-two-years	0.756108	0.738693	0.732841	0.737042	0.731612	0.731694	0.719881	0.735525	0.743487	0.731292
27	cpu_small	0.976494	0.976042	0.975872	0.975639	0.976663	0.975873	0.973743	0.974955	0.978109	0.975945
28	cylinder-bands	0.949393	0.864935	0.879808	0.883125	0.876687	0.827036	0.853583	0.854307	0.911998	0.822818
29	Dexter	0.991429	0.973111	0.983956	0.960089	NaN	NaN	0.763479	0.799978	NaN	NaN
30	Dorothea	NaN	NaN	0.508602	0.453330	NaN	NaN	NaN	NaN	NaN	NaN
31	dresses-sales	0.825000	0.569195	0.640460	0.639901	0.607225	0.489130	0.554661	0.546667	0.689984	0.471002
32	enron	0.997292	0.884432	0.898522	0.873784	0.790343	0.790986	0.718332	0.697815	0.762809	0.664219
33	eucalyptus	0.949097	0.843556	0.887313	0.831847	0.830522	0.814445	0.865170	0.830236	0.865406	0.871451
34	fri_c4_250_100	1.000000	0.936364	0.936623	0.893766	0.806494	0.897403	0.792016	0.880779	0.915584	0.904935
35	fri_c4_500_100	NaN	NaN	0.941929	0.933653	0.928013	0.939684	0.906339	0.910016	0.946055	0.951396
36	fried	0.988708	0.986223	0.985349	0.984608	0.985616	0.987100	0.979123	0.979034	0.983754	0.985732
37	GAMETES_Epistasis_2-Way_1000atts_0.4H_EDM-1_ED...	0.772899	0.785184	NaN	NaN	0.552125	0.511912	0.487115	0.510978	0.573937	0.445653
38	gina	0.988812	0.982460	0.984856	0.984250	0.975431	0.982528	NaN	NaN	0.979468	0.984593
39	gina_agnostic	0.990532	0.987591	0.984970	0.985687	0.978103	0.981534	0.978781	0.981399	0.979653	0.980843
40	gisette	0.997106	0.996934	0.995554	0.996639	NaN	NaN	NaN	NaN	0.995360	0.994027
41	heart-h	0.962366	0.838218	0.903854	0.917503	0.919310	0.832597	0.862143	0.926937	0.956295	0.902549
42	hepatitis	1.000000	0.734879	0.911885	0.888105	0.886089	0.824795	0.824582	0.918347	0.961271	0.861680
43	higgs	0.794382	0.796516	0.793147	0.801296	0.807450	0.807176	0.796803	0.800802	0.801205	0.800336
44	hiva_agnostic	0.865047	0.752268	NaN	NaN	0.767232	0.769018	0.818993	0.763784	0.757467	0.773814
45	house_16H	0.962847	0.950832	0.954728	0.949463	0.949994	0.956793	0.953168	0.948860	0.948525	0.954738
46	hypothyroid	1.000000	0.994241	0.999846	0.992881	0.999480	0.998958	0.998524	0.998614	0.999762	0.991998
47	image	NaN	NaN	0.940287	0.947076	0.950049	0.930583	0.919142	0.939490	0.956338	0.938895
48	Internet-Advertisements	0.988398	0.970128	NaN	NaN	0.981224	0.979874	0.975337	0.976536	0.981529	0.976514
49	isolet	1.000000	1.000000	0.999867	1.000000	1.000000	0.999778	0.999383	1.000000	1.000000	0.990933
50	JapaneseVowels	0.999298	0.999526	0.999364	0.999449	0.999237	0.999460	0.998785	0.999753	0.999297	0.999509
51	jasmine	0.883118	0.893590	0.867014	0.887307	0.866582	0.876712	0.860826	0.886396	0.874014	0.867262
52	jungle_chess_2pcs_endgame_rat_rat	1.000000	1.000000	NaN	NaN	1.000000	1.000000	0.999997	1.000000	1.000000	1.000000
53	kc1-top5	1.000000	0.978261	0.985294	0.905797	0.978261	0.930147	0.688316	0.974638	NaN	NaN
54	kdd_ipums_la_97-small	0.995244	0.992318	0.994139	0.992619	0.992474	0.993209	0.991548	0.992265	0.985470	0.987091
55	KDDCup09_churn	0.716341	0.711292	0.690227	0.705054	0.722887	0.713700	0.711228	0.728711	0.710738	0.705851
56	kick	0.780399	0.780350	0.760670	0.765962	0.774304	0.774662	0.765043	0.766170	0.731611	0.726160
57	kr-vs-kp	1.000000	0.998405	0.999934	0.998293	0.998588	0.996880	0.998870	0.997052	0.999193	0.999498
58	lsvt	0.989796	0.880952	0.920635	0.868481	0.926871	0.848639	0.814287	0.866213	0.977500	0.738095
59	madelon	0.933123	0.929290	0.889664	0.921063	0.872037	0.879347	0.758208	0.784338	0.883136	0.885051
60	meta	0.953586	0.854509	0.903657	0.895921	0.863260	0.811767	0.707497	0.848961	0.921941	0.840131
61	mfeat-factors	1.000000	0.999778	0.999856	0.999467	0.999500	0.998256	0.999568	0.999311	0.999722	0.999044
62	mfeat-karhunen	1.000000	0.999222	0.999900	0.999700	0.994833	0.997589	0.997760	0.997900	0.999278	0.998400
63	MiniBooNE	0.982569	0.983007	0.980526	0.981413	0.982305	0.982120	NaN	NaN	0.982148	0.981907
64	molecular-biology_promoters	1.000000	0.965812	0.988604	0.984330	0.954416	0.754986	0.844970	0.846154	1.000000	0.881766
65	mushroom	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
66	nomao	0.994905	0.994970	0.993300	0.994071	0.993038	0.993709	NaN	NaN	0.993449	0.994110
67	oil_spill	0.993243	0.899660	0.964648	0.921237	0.883875	0.957031	0.915105	0.930378	0.938517	0.911719
68	OVA_Omentum	0.990146	0.897611	0.962247	0.903427	0.896580	0.910261	NaN	NaN	NaN	NaN
69	ozone-level-8hr	0.963594	0.892397	0.926495	0.925237	0.906887	0.908119	0.910587	0.902454	0.929624	0.908772
70	page-blocks	0.994726	0.993136	0.990826	0.992673	0.993594	0.987262	0.989549	0.992029	0.994214	0.988538
71	parkinsons	1.000000	0.945946	0.972032	0.970158	0.981982	0.956621	0.889151	0.938063	1.000000	0.799658
72	pbc	0.886248	0.746346	0.824329	0.780574	0.778446	0.762257	0.771437	0.745698	0.820010	0.718224
73	pc2	0.975789	0.814768	0.896988	0.859235	0.921083	0.775324	0.748725	0.893610	0.971410	0.826691
74	pc4	0.977554	0.924508	0.948718	0.943188	0.936903	0.941389	0.938502	0.935630	0.949254	0.929205
75	qsar-biodeg	0.970376	0.921011	0.935627	0.930754	0.922287	0.944126	0.926867	0.930698	0.924844	0.930298
76	reuters	0.987557	0.983128	0.984326	0.985767	0.979234	0.983914	0.989990	0.991828	0.980229	0.978067
77	ringnorm	0.997961	0.997277	0.998037	0.996568	NaN	NaN	0.997835	0.996923	0.997453	0.997848
78	Satellite	0.999297	0.995569	0.996446	0.993798	0.984967	0.973328	0.965978	0.995118	0.996262	0.947941
79	scene	0.998952	0.992740	0.993418	0.989469	0.980239	0.991606	0.946842	0.955454	0.993927	0.991407
80	segment	1.000000	0.999994	0.999985	1.000000	0.999976	0.999063	0.998936	0.999994	1.000000	0.999945
81	semeion	1.000000	0.999375	1.000000	0.993280	0.999687	1.000000	0.997079	0.999219	1.000000	0.999209
82	SPECTF	0.996825	0.789084	0.921034	0.815364	0.798181	0.804333	0.860407	0.786388	0.909322	0.791055
83	spectrometer	1.000000	0.992047	0.997666	0.993247	0.987020	0.984827	0.983632	0.994598	0.996930	0.997510
84	Speech	0.975125	0.881631	0.800864	0.829986	0.793474	0.808035	0.836018	0.849979	0.864170	0.851719
85	SpeedDating	0.871740	0.872513	0.852970	0.863050	0.865095	0.865992	0.855813	0.871605	0.862697	0.860733
86	stock	0.999188	0.985487	0.994110	0.988601	0.994482	0.997445	0.993886	0.992619	0.996984	0.995405
87	tecator	1.000000	0.980108	0.990622	0.976982	0.984086	0.992896	0.984914	0.974709	0.994346	0.997727
88	threeOf9	1.000000	1.000000	1.000000	1.000000	0.998896	0.999080	0.995029	0.999693	1.000000	1.000000
89	tokyo1	0.997420	0.981605	0.980052	0.984071	0.985653	0.953965	0.974072	0.986246	0.992086	0.964515
90	triazines	0.959402	0.712201	0.909307	0.860526	0.811962	0.822650	0.789903	0.869617	0.910227	0.840456
91	USPS	1.000000	0.999361	0.999274	0.998382	0.997230	0.999148	0.996716	0.997578	0.998070	0.999341
92	vehicle_sensIT	0.923826	0.923650	0.921897	0.921579	0.923661	0.924989	NaN	NaN	0.920645	0.920991
93	vote	0.997565	0.994225	0.994003	0.996446	0.996979	0.986574	0.981338	0.995647	0.999129	0.583333
94	wind	0.951847	0.944758	0.939545	0.941931	0.942646	0.940746	0.941552	0.943769	0.943613	0.940032
95	yeast_ml8	0.967345	0.889681	0.936040	0.913443	0.888670	0.867141	0.829834	0.912900	0.939108	0.871240
96	zoo	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.942857	1.000000	1.000000
97	tic-tac-toe	1.000000	0.999557	1.000000	0.999827	0.997710	1.000000	0.996738	0.990396	1.000000	1.000000
98	PieChart1	0.992500	0.878581	0.852743	0.768734	0.872571	0.790890	0.788366	0.858946	0.917795	0.830021
99	wdbc	1.000000	0.991673	0.999788	0.987720	0.991251	0.999523	0.996772	0.990671	0.992063	0.995707

Table 6.4: Regression experiments results. The train results correspond to the performance estimation after the training phase, while test results correspond to the true hold-out performance.

	filename	autosklearn train	autosklearn test	gama train	gama test	h2o train	h2o test	jad train	jad test	tpot train	tpot test
0	2dplanes	0.950085	0.947823	0.948234	9.478484e-01	0.948368	0.947715	NaN	NaN	0.948682	9.478484e-01
1	a3a	NaN	NaN	0.000029	-9.370000e-07	0.437922	0.423751	NaN	NaN	0.000000	-9.370000e-07
2	a9a	NaN	NaN	0.000178	-2.003530e-04	0.430259	0.433101	NaN	NaN	0.000000	-2.003530e-04
3	Ailerons	0.844426	0.849280	0.842660	8.487743e-01	0.846203	0.845496	0.837597	0.845300	0.843548	8.487743e-01
4	analcatdata_gsssexsurvey	0.736523	-0.058873	0.219816	-4.359572e-01	0.123470	-0.405896	-0.007635	-0.382715	0.250038	-4.359572e-01
5	analcatdata_ncaa	-0.009513	-0.244697	0.845351	4.183875e-01	-0.431792	-0.210301	-0.380070	-1.882442	0.654247	-7.561709e-01
6	analcatdata_supreme	0.987197	0.980342	0.980512	9.809991e-01	0.981112	0.981510	0.977265	0.980840	0.980454	9.809991e-01
7	auto_price	0.987197	0.848882	0.883043	7.894307e-01	0.880902	0.808512	0.825325	0.749301	0.927502	7.894307e-01
8	autoMpg	0.933684	0.852854	0.888371	8.234891e-01	0.887098	0.866480	0.895716	0.867934	0.899263	8.234891e-01
9	bank32nh	0.564623	0.551588	0.540222	5.421816e-01	0.594315	0.567778	0.559441	0.539324	0.567352	5.421816e-01
10	bank8FM	0.960201	0.961919	0.960560	9.623406e-01	0.962423	0.964331	0.959945	0.959532	0.963601	9.623406e-01
11	benzo32	0.504836	0.157110	0.309577	7.109068e-02	0.298736	0.164558	0.094044	0.192288	0.326411	7.109068e-02
12	bodyfat	0.988213	0.947249	0.961712	9.860697e-01	0.961252	0.987987	0.949436	0.985496	0.970967	9.860697e-01
13	boston_corrected	0.832821	0.789849	0.804931	7.872240e-01	0.715240	0.809159	0.686447	0.811721	0.797423	7.872240e-01
14	breastTumor	0.344039	-0.155506	0.079374	-2.061685e-02	-0.223517	-0.345427	0.014988	0.072279	0.135368	-2.061685e-02
15	cholesterol	0.386261	0.004173	0.049906	-1.228303e-01	-0.218510	-0.362353	-0.057712	-0.081221	0.028303	-1.228303e-01
16	chscase_census5	0.020566	-0.072108	0.038440	-5.693800e-02	0.002734	-0.004619	-0.025640	-0.000455	0.028240	-5.693800e-02
17	cleveland	0.657419	0.374300	0.614305	3.122782e-01	0.386512	0.213183	0.456919	0.434471	0.672299	3.122782e-01
18	coil2000	0.061328	0.043434	0.067966	4.254096e-02	0.000254	0.038544	0.060034	0.055768	0.063751	4.254096e-02
19	connect-4	NaN	NaN	0.000034	-7.560000e-07	0.665156	0.668400	NaN	NaN	0.000000	-7.560000e-07
20	CPMP-2015-regression	0.944621	0.930249	0.931641	9.367511e-01	0.922156	0.930583	0.918229	0.919693	0.936972	9.367511e-01
21	CPMP-2015-runtime-regression	0.371083	0.362246	0.347000	3.475402e-01	0.306083	0.359280	0.198950	0.253678	0.361740	3.475402e-01
22	cps_85_wages	0.450054	0.156129	0.376815	1.975608e-01	0.444580	0.208708	0.258460	0.205082	0.331212	1.975608e-01
23	cpu	0.999708	0.978418	0.985743	9.453963e-01	0.961037	0.901753	0.894700	0.997423	0.983939	9.453963e-01
24	cpu_act	0.983665	0.982894	0.984506	9.833933e-01	0.984960	0.984210	0.980745	0.979559	0.983783	9.833933e-01
25	cpu_small	0.978140	0.976902	0.978650	9.780977e-01	0.978236	0.977886	0.974567	0.974231	0.977909	9.780977e-01
26	crimecommunitynums	0.555785	0.561352	0.555847	5.331432e-01	0.558059	0.548594	0.485575	0.512828	0.536783	5.331432e-01
27	dataset_sales	0.773870	0.789667	0.809132	8.197059e-01	0.801697	0.835368	0.728460	0.746349	0.803840	8.197059e-01
28	dataset-autoHorse_fixed	0.958975	0.867300	0.906527	7.816151e-01	0.879936	0.897490	NaN	NaN	0.911290	7.816151e-01
29	debutanizer	0.822399	0.752430	0.799694	8.333914e-01	0.762101	0.799955	0.731500	0.739038	0.824381	8.333914e-01
30	delta_elevators	0.651399	0.644568	0.642843	6.434138e-01	0.625387	0.635007	0.639879	0.639597	0.647760	6.434138e-01
31	Diabetes(scikit-learn)	NaN	NaN	0.481122	4.849097e-01	0.413696	0.475645	0.463311	0.456137	0.490987	4.849097e-01
32	diamonds	0.980203	0.980162	0.980821	9.808838e-01	0.981763	0.981651	0.971201	0.971706	0.981334	9.808838e-01
33	echoMonths	0.730556	0.278147	0.522363	4.279333e-01	0.331913	0.498021	0.459503	0.425888	0.500324	4.279333e-01
34	elevators	0.895705	0.899131	0.872884	9.099019e-01	0.891978	0.804711	0.881738	0.875438	0.904168	9.099019e-01
35	fishcatch	0.980813	0.984135	0.902652	9.767629e-01	0.968938	0.916793	0.960591	0.981452	0.981740	9.767629e-01
36	fri_c3_1000_10	0.966603	0.951329	0.926965	9.563453e-01	0.929952	0.935258	0.827685	0.824738	0.951268	9.563453e-01
37	fri_c4_250_100	0.953711	0.911005	0.733871	8.207397e-01	0.682567	0.672136	0.484751	0.589262	0.835330	8.207397e-01
38	fried	0.956615	0.957324	0.950330	9.542782e-01	0.955653	0.955775	NaN	NaN	0.953762	9.542782e-01
39	GeographicalOriginalofMusic	0.785564	0.780751	0.792036	7.914615e-01	0.809836	0.783981	0.770029	0.781654	0.784336	7.914615e-01
40	german.numer	0.338121	0.188194	0.223243	1.974322e-01	0.234352	0.172616	0.219483	0.208786	0.279431	1.974322e-01
41	HappinessRank_2015	1.000000	1.000000	1.000000	9.999999e-01	0.999595	0.999575	0.999997	0.999998	1.000000	9.999999e-01
42	higgs	0.267635	0.263955	0.256264	2.632008e-01	0.280751	0.281519	NaN	NaN	0.265594	2.632008e-01
43	house_16H	NaN	NaN	0.629198	6.072566e-01	0.662429	0.635991	NaN	NaN	0.650418	6.072566e-01
44	house_8L	0.659398	0.672787	0.676854	6.827985e-01	0.692773	0.691468	NaN	NaN	0.680634	6.827985e-01
45	houses	0.808763	0.793055	0.809253	8.180517e-01	0.850312	0.840862	0.782917	0.777692	0.826368	8.180517e-01
46	hungarian	0.607929	0.462574	0.492862	3.758636e-01	0.430021	0.475086	NaN	NaN	0.534355	3.758636e-01
47	ICU	1.000000	-0.013502	0.593095	-1.849905e+00	0.438543	-0.033855	-0.162530	-0.165680	0.827381	-1.849905e+00
48	ilpd-numeric	0.286539	0.121781	0.204533	3.718360e-02	0.226220	-0.026495	0.129527	0.124208	0.186366	3.718360e-02
49	kc1-numeric	0.819453	0.352679	0.781667	2.069835e-01	0.112011	0.090396	-0.324801	0.167734	0.292608	2.069835e-01
50	kdd_coil_3	0.305624	0.057717	0.257731	-4.953387e-02	-0.464817	-0.618412	0.048544	0.063575	0.189489	-4.953387e-02
51	kin8nm	0.916116	0.912071	0.757327	8.338447e-01	0.857413	0.876850	0.886760	0.893795	0.823118	8.338447e-01
52	LoanDefaultPrediction	0.052746	0.027256	0.002700	2.232458e-03	-0.043168	0.005355	NaN	NaN	0.001114	2.232458e-03
53	lowbwt	0.658200	0.646666	0.622044	5.745034e-01	0.624137	0.486804	0.563992	0.584902	0.584667	6.333592e-01
54	lungcancer_shedden	0.413244	0.234145	0.244446	2.750745e-01	0.219061	0.119400	0.139952	0.205553	0.288418	2.750745e-01
55	mauna-loa-atmospheric-co2	0.999267	0.999148	0.999273	9.992855e-01	0.999163	0.999259	0.997125	0.997137	0.999323	9.992855e-01
56	MIP-2016-PAR10-regression	0.354588	0.343488	0.330283	3.801132e-01	0.386078	0.292126	0.090497	0.199494	0.275671	3.801132e-01
57	mnist_rotation	0.347968	0.334708	0.298883	2.523194e-01	0.440543	0.434807	NaN	NaN	0.254766	2.523194e-01
58	Moneyball	0.945988	0.952824	0.938744	9.520130e-01	0.932932	0.933429	0.939828	0.950385	0.938950	9.520130e-01
59	mtp	0.551569	0.523771	0.500002	5.408654e-01	0.545502	0.574002	0.518560	0.529185	0.527194	5.408654e-01
60	mv	0.999977	0.999974	0.999979	9.999678e-01	0.999966	0.999970	NaN	NaN	0.999968	9.999678e-01
61	NewFuelCar	0.998474	0.998360	0.998469	9.986077e-01	0.998558	0.998824	0.998606	0.998820	0.998409	9.986077e-01
62	nki70.arff	0.705168	0.106110	0.536599	2.942097e-01	0.403330	0.278388	0.298491	0.221449	0.455394	2.942097e-01
63	no2	0.642183	0.515281	0.611418	5.639128e-01	0.555740	0.533970	0.498183	0.514822	0.612489	5.639128e-01
64	OnlineNewsPopularity	0.026906	0.024591	0.022556	2.266651e-02	0.232649	-0.020801	NaN	NaN	0.034781	2.266651e-02
65	ozone_level	0.273696	0.125584	0.028376	1.592058e-01	0.271440	0.059570	NaN	NaN	0.211678	1.592058e-01
66	parkinson-speech-uci	0.564673	0.492351	0.484438	3.790009e-01	0.660209	0.516128	0.371383	0.362527	0.535722	3.790009e-01
67	pbc	0.765393	0.679868	0.603921	4.863010e-01	0.533665	0.555724	0.504209	0.559513	0.578851	4.863010e-01
68	pbcseq	0.969652	0.964035	0.924936	9.608404e-01	0.920616	0.955036	0.847168	0.890355	0.988141	9.608404e-01
69	pharynx	0.757788	0.315957	0.547900	3.851159e-01	0.584963	0.274331	0.458018	0.386998	0.631170	3.851159e-01
70	places	0.336711	-0.023505	0.243678	-5.903724e-02	-0.129658	-0.149993	0.124239	0.060509	0.209429	-5.903724e-02
71	pol	0.982332	0.983921	0.985297	9.866268e-01	0.991802	0.991998	0.976825	0.978256	0.986278	9.866268e-01
72	puma32H	0.948734	0.946741	0.935956	9.385457e-01	0.933694	0.936264	0.884204	0.868853	0.938898	9.385457e-01
73	pwLinear	NaN	NaN	0.889402	8.617689e-01	0.827714	0.828719	0.788436	0.829365	0.871173	8.617689e-01
74	QSAR-TID-10541	0.669418	0.250264	0.466881	3.319923e-01	0.536353	0.233499	0.265147	0.372691	0.155973	3.319923e-01
75	QSAR-TID-10849	0.505181	0.482853	0.476051	4.591149e-01	0.466621	0.486321	NaN	NaN	0.470936	4.591149e-01
76	QSAR-TID-13004	0.602427	0.399552	0.565493	3.784108e-01	0.661614	0.382989	NaN	NaN	0.552632	3.784108e-01
77	QSAR-TID-17061	0.953444	0.807320	0.882326	8.266515e-01	0.928129	0.815622	0.829189	0.797543	0.864782	8.266515e-01
78	QSAR-TID-194	0.711738	0.707700	NaN	NaN	0.722343	0.731620	NaN	NaN	0.704520	7.097799e-01
79	QSAR-TID-234	0.685873	0.668730	0.653335	6.061380e-01	0.670283	0.683821	NaN	NaN	0.638942	6.061380e-01
80	QSAR-TID-30008	0.399110	0.386144	0.262423	3.420940e-01	0.380416	0.394667	0.233915	0.408344	0.274423	3.420940e-01
81	rmftsa_ladata	0.503739	0.505485	0.650247	6.397771e-01	0.710629	0.564846	0.580401	0.630360	0.589662	6.397771e-01
82	SAT11-HAND-runtime-regression	0.653813	0.669612	0.615672	7.312475e-01	0.640220	0.696739	0.647682	0.676002	0.696072	7.312475e-01
83	satellite_image	0.902689	0.905195	0.899926	8.995034e-01	0.909112	0.906400	0.895869	0.897909	0.906002	8.995034e-01
84	SensIT-Vehicle-Combined	0.632394	0.632969	0.637937	6.238363e-01	0.649091	0.651342	NaN	NaN	0.623628	6.238363e-01
85	sensory	0.261557	0.137996	0.221455	2.229183e-01	0.235012	0.248597	0.148766	0.058455	0.237281	2.229183e-01
86	sleuth_case2002	0.185500	0.085488	0.345402	3.839839e-01	0.375166	0.144786	0.316733	0.424992	0.395589	3.839839e-01
87	splice	0.927011	0.892213	0.886026	9.035729e-01	0.882179	0.890844	0.844578	0.858991	0.899572	9.035729e-01
88	stock	0.988509	0.983949	0.987596	9.865390e-01	0.980064	0.985285	0.981477	0.983955	0.989445	9.865390e-01
89	svmguide3	0.481150	0.336668	0.410252	3.722528e-01	0.301758	0.312988	0.358379	0.350387	0.430679	3.722528e-01
90	SWD	NaN	NaN	0.451091	3.658839e-01	0.436488	0.351577	0.405124	0.356556	0.454674	3.658839e-01
91	tecator	0.998665	0.998052	0.996552	9.982181e-01	0.996541	0.995753	0.996051	0.995840	0.997569	9.982181e-01
92	Titanic	0.495757	0.544274	0.476388	5.671089e-01	0.289854	0.553820	0.427494	0.527648	0.571786	5.671089e-01
93	topo_2_1	0.069028	0.069840	0.050968	8.257449e-02	-0.155664	0.027032	NaN	NaN	0.052783	8.257449e-02
94	veteran	0.618567	0.037673	0.254505	5.491190e-03	0.468031	0.043177	-0.028580	0.048614	0.016252	5.491190e-03
95	w1a	NaN	NaN	NaN	NaN	0.649156	0.632939	NaN	NaN	0.000000	-1.030000e-05
96	wind	0.927011	0.795072	0.782373	7.964196e-01	0.789053	0.800111	0.788856	0.793607	0.790414	7.964196e-01
97	wine_quality	0.481150	0.456860	0.456438	4.949813e-01	0.437803	0.498808	0.423714	0.462014	0.458906	4.949813e-01
98	wisconsin	0.427787	0.103395	0.148574	1.049189e-01	0.176749	-0.049034	0.040914	0.079296	0.147728	1.049189e-01
99	yprop_4_1	0.481150	0.088314	0.073699	1.010145e-01	-0.141030	0.084769	0.076026	0.104879	0.073626	1.010145e-01