Towards Big Industrial Data Mining Through Explainable Automated Machine Learning

Moncef Garouani (mgarouani@gmail.com)
ULCO: Universite du Littoral Cote D'Opale
Adeel Ahmad
ULCO: Universite du Littoral Cote D'Opale
Mourad Bouneffa
ULCO: Universite du Littoral Cote D'Opale
Mohamed Hamlich
ULCO: Universite du Littoral Cote D'Opale
Gregory Bourguin
ULCO: Universite du Littoral Cote D'Opale
Arnaud Lewandowski
ULCO: Universite du Littoral Cote D'Opale

Research Article

Keywords: Machine learning, AutoML, Explainable AI, Data analysis, Decision-support systems, Industry 4.0

Posted Date: August 10th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-755783/v1

Version of Record: A version of this preprint was published at The International Journal of Advanced Manufacturing Technology on February 10th, 2022. See the published version at https://doi.org/10.1007/s00170-022-08761-9.

Noname manuscript No.
(will be inserted by the editor)

Towards big industrial data mining through explainable automated machine learning

Moncef Garouani [1, 2, 3] · Adeel Ahmad [1] · Mourad Bouneffa [1] · Mohamed Hamlich [2] · Gregory Bourguin [1] · Arnaud Lewandowski [1] ·

Received: date / Accepted: date

Abstract Industrial systems resources are capable of producing large amount of data. These data are often in heterogeneous formats and distributed, yet they provide means to mine the information which can allow the deployment of intelligent management tools for production activities. For this purpose, it is necessary to be able to implement knowledge extraction and prediction processes using Artificial Intelligence (AI) models but the selection and configuration of intended AI models tend to be increasingly complex for a non-expert user. In this paper, we present an approach and a software platform that may allow industrial actors, who are usually not familiar with AI, to select and configure algorithms optimally adapted to their needs. Hence, the approach is essentially based on automated machine learning. The resulting platform effectively enables a better choice among the combination of AI algorithms and hyper-parameter configurations. It also makes it possible to provide features of explainability of the resulting algorithms and models, thus increasing the acceptability of these models in practicing community of the users. The proposed approach has been applied in the field of predictive maintenance. Current tests are based on the analysis of more than 360 databases from the subjected field.

Keywords Machine learning · AutoML · Explainable AI · Data analysis · Decision-support systems · Industry 4.0

Moncef Garouani
E-mail: moncef.garouani@etu.univ-littoral.fr

[1] Univ. Littoral Cote d'Opale, UR 4491, LISIC, Laboratoire d'Informatique Signal et Image de la Cote d'Opale,F-62100 Calais, France

[2] CCPS Laboratory, ENSAM, University of Hassan II, Casablanca, Morocco

[3] Study and Research Center for Engineering and Management(CERIM), HESTIM, Casablanca, Morocco

1 Introduction

Data driven decision-making may be defined by the set of practices aiming to make decisions based on data analysis rather than on intuitive insights [1]. The business entities that have deployed data-driven decision-making activities have been observed as more profitable and productive compared to the traditional ones [2]. Nowadays, decision making tools are mainly based on results of the current AI research works [3]. The success of AI based tools is mainly due to the advances in machine learning approaches [4]. This is particularly stimulated by the availability of large datasets concerning various real-world features [3] and also through the increase of the computational gains which are generally attributed to the powerful GPU cards [5].

The manufacturing area is one of those generating huge amounts of data gathered by means of Cyber Physical System (CPS) devices. The availability of such data combined with the knowledge of manufacturing experts may be an opportunity to build AI based processes and models providing high value insights and assets for decision makers. Nevertheless, building such processes and models requires AI and data science skills and expertise that are not always available in the manufacturing area workbenches and laboratories.

The work shown in this paper aims to bridge the gap between AI expertise and manufacturing experts. We believe that automated machine learning (AutoML) [6]

is one of the most powerful approaches that can effectively deal with this problem; notably the ones proposed in [7–10]. Instead of searching appropriate Machine Learning (ML) algorithms and manually tuning the hyper-parameters in a virtually infinite space, the AutoML leads to automatically and iteratively configure the appropriate hyper-parameters of multiple machine learning algorithms in a virtually infinite space, in order to optimize these parameters for a predefined search space.

The use of AutoML approaches has gained significant attention in research community. A plethora of systems for AutoML, such as Auto-sklearn [7], TPOT [8],Auto-WEKA [9], ATM [10], and Google Cloud AutoML[1] have been developed in recent years.

The interest of building complex AI models that are able to achieve unprecedented performance levels has been gradually replaced by a growing concern for alternative design factors leading to an improved usability of the resulting tools. Indeed, in a manifold of application areas, complex AI models become of limited practical utility [11]. The major reason lies on the fact that AI models are often designed to focus the performance factors, thus leaving aside other important and even sometimes the crucial aspects such as confidence, transparency, fairness or accountability. The absence of explanation for predicted performing factors make the AI models usually black boxes, which only allows the prominent exhibition of input and output parameters but conceal the visibility of inherent associations among them. It is more preferably desired to avoid such lack of transparency in real-life applications such that in industrial manufacturing processes. Since, these applications may imply critical decision choices, it is favorable to have some justifications of individual predictions which are perceived trough an AI algorithm, more particularly, in an automated environment. Therefore acceptance of, and the trust in, an AutoML system is highly dependent on the transparency of the recommendations.

Because of the lack of transparency in AutoML systems as Decision Support Systems (DSS), users tend to question the validity of automatic results, such that : *did the AutoML run long enough? Did the AutoML miss some suitable models? Did the AutoML sufficiently explore the search space? Did the recommended configuration over or under fit?, etc.* Such queries may cause reluctance for users to apply the results of AutoML in more critical situations [12]. Meanwhile, when AutoML provides unsatisfactory results, users are unable to reason and thus cannot improve the obtained results. They may only increase the computational budget (e.g., the run-time) as much as possible, which can result as barriers of the AutoML effectiveness.

It is therefore a preliminary objective of the current work to make the outcome from such well-performing AutoML systems transparent, interpretable and self-explainable. This shall make AutoML support systems more reliable and operational through a set of different visual summary levels of the provided models and configurations. It may render the AutoML system more transparent and controllable, hence increasing its acceptance.

In the current work, we attempt a transparent and auto-explainable AutoML system for recommending the most adequate ML configuration for a given problem and explain the rationale traceability behind a recommendation. It may further allow to analyze the predictive results in an interpretable and reliable manner. In the proposed approach, the end users can explore the AutoML process at different levels, such as described in the following :

- The AutoML-oriented level (i.e. exploring the AutoML process from recommendation to refinement).
- The Data-oriented level (i.e. exploring data properties through different visualization levels).
- The Model-oriented level (i.e. exploring the models provided by the AutoML system (e.g. model performance, what-if-analysis, decision path, etc.)).

The system consists of two integrated modules : the *Automated Machine Learning module* and the *Automated Machine Learning explainer (AMLExplainer)*. However, the later, AMLExplainer module is not system or algorithm specific; it is inter-operable with a variety of AutoML frameworks. The main contributions of this work are summarized as follows:

1. Develop a premier transparent and self-explained AutoML system.
2. Provide an assisted traceability of the reasoning behind the AutoML recommendation generation process.
3. Develop a module that can explain the predictions of any recommendation through linked visual summary and/or textual information.
4. Provide a multi-level interactive visualization tool that facilitate the model operation and performance inspection to address the "trusting the model".
5. Provide a reliable guidance, when AutoML returns unsatisfying results in order to improve the expected performances by assessing the importance of an algorithm hyper-parameters.

The rest of the paper is organized as follows : the section 2 discusses the closely related works in respect

[1] https://cloud.google.com/automl

of ML-based data analytics solutions and the need for transparency to gain trust in AI models. The section 3 briefly describes the different types of explanations and also their respective information content with their use in practice. The section 4 provides an overview of the proposed framework and discusses how these components collaborate to achieve the pursued goals. Finally, the contents of this paper are concluded in section 5 along with outlines of future perspectives.

2 Related works

The available literature testifies a considerable progress in respect of automated machine learning methodologies and their application in multi-disciplinary areas. We have been attentively studying some of the most relevant works, particularly the recommendation-based decision support systems in manufacturing industry. We particularly considered the works dealing with transparency and explainability of automated machine learning and those related to big industrial data mining.

Through the advances on high-tech sensing and the widespread use of applications such as electronic manufacturing records, mobile sensors, and *Industrial Internet of Things (IIoT)* tools, manufacturing data are being accumulated at an exponentially growing rate annually. According to a recent research report [13], the global big data market size will grow from USD 138.9 billions in 2020 to USD 229.4 billion by 2025 (increase from Petabytes to Exabytes). Machine learning is a key technology to transform large manufacturing data sets, or "big industrial data", into actionable knowledge.

In this section, we observe the limitations of available research work on explainable AI /AutoML systems as DSS that primarily motivate the work described in this paper. The current literature, in this regard can be generally summarized in form of an overlapped overview of three major research areas, as described in the following subsections.

2.1 Challenges in selecting and configuring machine learning algorithms

The machine learning is widely used in many industrial applications across different levels, including processes, machines, shop floors, and supply chain levels. For instance, machine learning models can be used to control product quality [14], to monitor the condition of tools by tracking the evolution of their state [15], or to monitor the health of machines by predicting the time of occurrences of machine failures and also to estimate the criticality of these failures [16]. However, despite its countless benefits and advances, building a machine learning pipeline is still a challenging task, partly because of the difficulty in manually selecting an effective combination of an algorithm and hyper-parameters values for a given task or problem.

Owing to the development of open source ML packages and the active research in the ML field, there are dozens of machine learning algorithms, where each machine learning algorithm has two types of model parameters: (1) ordinary parameters that are automatically optimized or learned during the model training phase; (2) and hyper-parameters (categorical and continuous) that are typically set by the user manually before the training of the model (as shown in Table 1).

Table 1 Configuration space of some classification algorithms.

ML Algorithm	Number of Ordinary parameters	Number of Hyperparameters
Support Vector Machine	2	5
Decision Tree	1	3
Random Forest	2	4
Logistic Regression	4	6

To achieve the desired performance for a particular problem, users typically try a set of models and configurations based on their understanding of the algorithms and their observation of the data since there is no algorithm that performs well on all possible problems (i.e., No Free Lunch [17]). Then, based on the feedback about how the learning tools performed, the practitioner may adjust the configuration to verify if the performance can be improved. Such a trial-and-error process terminates once a desired performance is achieved or the computational budget runs out (as shown in Figure 1).

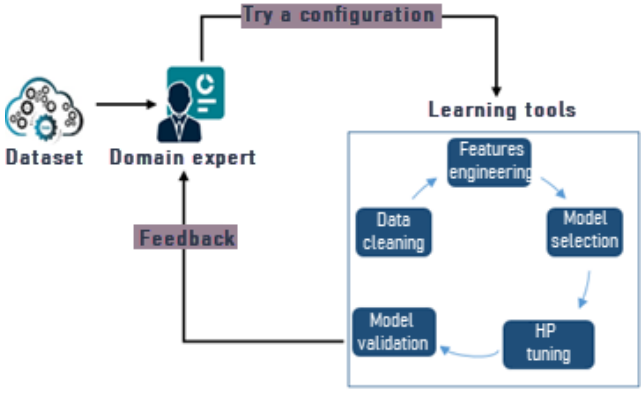

Fig. 1 The ML configuration tuning process.

2.2 Automated machine learning

The Automated Machine Learning or AutoML [18] field is among the rapidly emerging sub-fields of ML that attempts to address the theoretical and algorithmic challenges in order to fully automate the ML process. It addresses also the development and deployment of systems in this regard. AutoML has two main goals : (1) democratizing the application of ML to non-experts of data analysis by providing them with "off the shelf" solutions, and (2) enabling the knowledge practitioners to save time and effort.

Over the last few years, a plethora of AutoML systems have been developed providing partial or complete ML automation, such as Auto-sklearn [7], TPOT [8],Auto-WEKA [9], ATM [10], as well as commercial systems such as Google AutoML[1], RapidMiner[2], DarwinAI[3], and DataRobo[4]. These tools range from automatic data preprocessing [19, 20], automatic feature engineering [21, 22] to automatic model selection [18, 23] and automatic hyper-parameters tuning [24, 25]. Some approaches attempt to automatically and simultaneously choose a learning algorithm and optimize its hyper-parameters. These approaches are also known as Combined Algorithm Selection and Hyper-parameters optimization problem (CASH) [7–9, 25–28]. Table 2 shows a comparison among some of the most popular AutoML tools, in terms of training framework, supported ML tasks, automatic features engineering, user interface and process transparency.

The interest in developing complex AI models, despite their revolutionary characteristics and capabilities of achieving unprecedented levels of performance has been progressively perishing. The loss of sustainability is mainly concerned with alternative design factors particularly to make such models more usable in practice [3]. These systems emphasize on useful assistance but their usability is greatly limited to provide detailed analysis about the recommended configurations and the insightful working of the black-box models [12]. The reason lies on the fact that AI models are often designed with performance as their only design goal, thus ignoring other important matters such as privacy awareness, confidence, transparency, and accountability makes them usually untrustworthy black-boxes [12, 29].

[2] https://rapidminer.com
[3] https://darwinai.com/
[4] thttps://www.datarobot.com/

2.3 The need for Transparency to Trust in AI and in AutoML

Black-box AI systems have been used in various areas. Their implication in critical domains, like in power consumption forecasting or supply chain management to analyze the brands trends or consumer sentiments; usually have less focus to consider the quality features such as transparency and explainability rather considering more importantly the system's overall performance. However, even if these systems fail, e.g., the Quality Control system is mostly not able to detect the failure, the Equipment Failure Prevention system are less expected to identify the exact cause of failure and generally produces false or inaccurate predictions. The consequences are rather underwhelming. In industrial critical applications, the situations are different where the lack of transparency of ML techniques can be a disqualifying factor, if not limited. Specifically, a single wrong decision can be highly risked to put in danger the entire production line (e.g., failure of a critical unit) and can cause significant financial deprivations (e.g., product conformity). It is therefore, relying on an incomprehensible black-box data-driven system would not be the best option. The lack of transparency is among the most relevant reasons to question the adoption of AI models in manufacturing industry. The stakeholders are more cautious than doing so in the consumer entertainment, or e-commerce industries.

Explaining the reasoning behind one's decisions or actions is an important part of human interactions in the social dimension [30]. As the explanations help to build trust in human-to-human relationships, similarly, these should also be part of human-to-machine interactions [3]. In this work, we investigate the contributions and feasibility of a process designed to make such powerful DSS transparent, interpretable and self-explainable to foster trust, both in situations where the AI system has a supportive role (e.g., production planning) and in those where it provides directions and decision-making (e.g., Quality Control, predictive maintenance or autonomous driving). In the former cases, explanations provide extra information, which help the human in the loop to gain an overall view of the situation or the problem at hand in order to take decisions. It is similar to an expert who has to provide a detailed report explaining his/her findings, a supportive AI system should explain the decisions in detail instead of providing only a prediction or a decision.

Table 2 Summary of the main features of some state-of-the-art AutoML tools. (*): commercialized tools.

Tool	Training Framework	ML task	Automatic features engineering		User interface	Transparent / Self-explainable
			Categorical data processing	Missing values imputation		
Google AutoML *	TensorFlow, SparkML	Classification	NO	YES	YES	YES
DarwinAI *	TensorFlow	Classification	YES	YES	YES	YES
DataRobot *	Scikit-learn, TensorFlow, Spark	Classification	YES	YES	YES	YES
AutoSklearn[7]	Scikit-learn	Classification & Regression	NO	YES	NO	NO
TPOT [8]	Scikit-learn	Classification & Regression	YES	NO	NO	NO
AutoWeka[9]	Weka	Classification & Regression	NO	YES	YES	NO
Hyperopt-Sklearn [25]	Scikit-learn	Classification	YES	NO	NO	NO
AutoKeras [26]	Keras	Classification	NO	NO	NO	NO
SmartML[28]	RWeka	Classification	YES	NO	YES	NO

3 Explainable AI

Explainable AI (XAI) [11] refers to artificial intelligence technologies that can provide human-understandable explanations for their output or actions [31]. End users, by nature, may wonder about the reasoning behind how and why algorithms make or arrive to decisions [29]. As the complexity of the AI algorithms and systems grows, they are viewed as "black-boxes" [27, 32]. Increasing complexity can result in the lack of transparency that hampers understanding the reasoning of these systems, which negatively affects the users trustiness.

Model explainability can be divided into two categories : global explainability and local explainability. Global explainability means the users can understand the model directly from its overall structure. Local explainability just consider a specific input and it tries to find out why the model makes a certain decision.

The development of methods for explaining, visualizing and interpreting machine learning models has recently gained increasing attention under the Explainable AI (XAI) area [11, 12, 29, 31, 32]. In the recent years, the advancements in XAI are grown rapidly but there are still broader gaps to generalize XAI approaches. The current major XAI methodologies are only applicable to specific type of data and models. Such specificities mostly require the pre-configuration of input parameters that are not easily coded by non-experts. In contrast, our proposed system intends to support the analysis and inspection of all machine learning classification models without any data type dependency, neither even having to write any line of code. A variety of XAI methods characteristics in terms of data explanations level, data and model dependency, and pre-configuration requirements are highlighted in Table 3.

It can be useful to primarily establish a consensus of understanding on what the term *explainability* may refer in the context of artificial intelligence and, more specifically, in the area of machine learning. Different levels of explanations provide insights into different aspects of the model, ranging from information about the learned representations to the identification of distinct prediction strategies and the assessment of the overall model behavior. Depending on the recipient and his or her intent, it may be advantageous to focus on one particular level of explanation.

Recent design recommendations put more focus on the importance of intuitive interfaces, along with a clean and concise presentation, among the explanation facilities, and easy user interactions [38]. In order to make AI systems as decision support systems accessible and effortless for both machine learning experts and neophytes, system builders and designers should present not only the final model prediction or recommendation coming out of the system, but also the pipeline steps and decisions made in each of those steps along with the prediction or decision generation process. In this argument, more clarity and transparency for users are expected from the AutoML system. To accommodate the user needs of transparency and trust, a few recent works have proposed prototypes design for increasing AI systems transparency [38–40]. However, most of these systems fall short in providing an overview of the AutoML process of how and why a rec-

Table 3 Properties of XAI state of the art tools. Level is the interpretability coverage: local or global. Dependency specifies necessary inputs.

XAI method	Level		Dependency		Require pre-configuration
	Local	Global	Data	Model	
LIME[12]	●	○	?	○	●
ANCHORS[33]	●	○	●	○	●
Node-Link Vis[34]	●	●	○	●	●
SHAP [35]	●	●	○	○	●
DeepTaylor[36]	●	○	●	●	●
Occlusion [37]	●	○	●	●	●
AMLExplainer	●	●	○	○	○

ommended pipeline configuration was generated, and their interfaces are complex and not suitable for manufacturing routine and needs. Furthermore, the incorporated explanation facilities are often insufficient and/or not tailored to the industrial domain.

Our proposed system aims to provide guidance to solve particular problems (Figure 2). Given a dataset, the tool automatically recommend the most adequate ML configurations and allow users to easily observe and analyze these models through an interactive multiple views module that explain the inner working of any machine learning classifier in an interpretable and faithful manner. The goal is manifold : (1) facilitate the models working and performance inspection through linked visual summaries and textual information (2) provide a visual summary of all evidence items and their relevance for the computation result, and (3) present a guided investigation of the reasoning behind the recommendation generation.

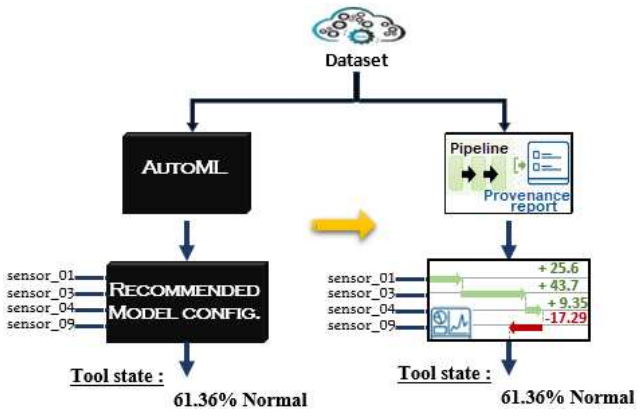

Fig. 2 "Black-box" model recommendation and prediction to "White box" model with explanations.

4 The conceptual framework

Given a predictive modeling problem for an industrial application, it is often difficult to build an accurate machine learning based predictive model that is easy to develop and to be interpreted by non-ML experts. The key idea for our transparent and explainable automated machine learning vision is to separate recommendations from explanations by using two modules simultaneously. The first module is used for making the recommendation of the most adequate ML configuration for a problem at hand and aims to maximize the requested predictive performance metric (e.g. Accuracy, Precision, Recall). The second module is used for providing the rationale behind the recommended configuration as well as automatically explaining the inner workings of the model.

The following section describes the design and implementation choices of the proposed tool, a complete, transparent and self-explainable AutoML system. As it is shown by the Figure 3. For the recommender module (AMLBID), when a new dataset is presented, AutoML is performed, and a list of candidate pipelines is provided based on the task at hand. The dataset characteristics, AutoML output and candidate pipelines list are supplied to the explanatory module to generate an interactive dash to help the end-user understand the provided results, diagnose the performance of the generated pipelines and explore the possibilities of performance refinement.

4.1 The recommender module

The Automated Machine Learning tool for Big Industrial Data (AMLBID) module is a Meta-learning [41] based system for automating the algorithm selection and tuning problem using a recommendation system that is bootstrapped with a meta-knowledge base. This knowledge-base, derived from a large set of experiments conducted on 360 real-world datasets from different

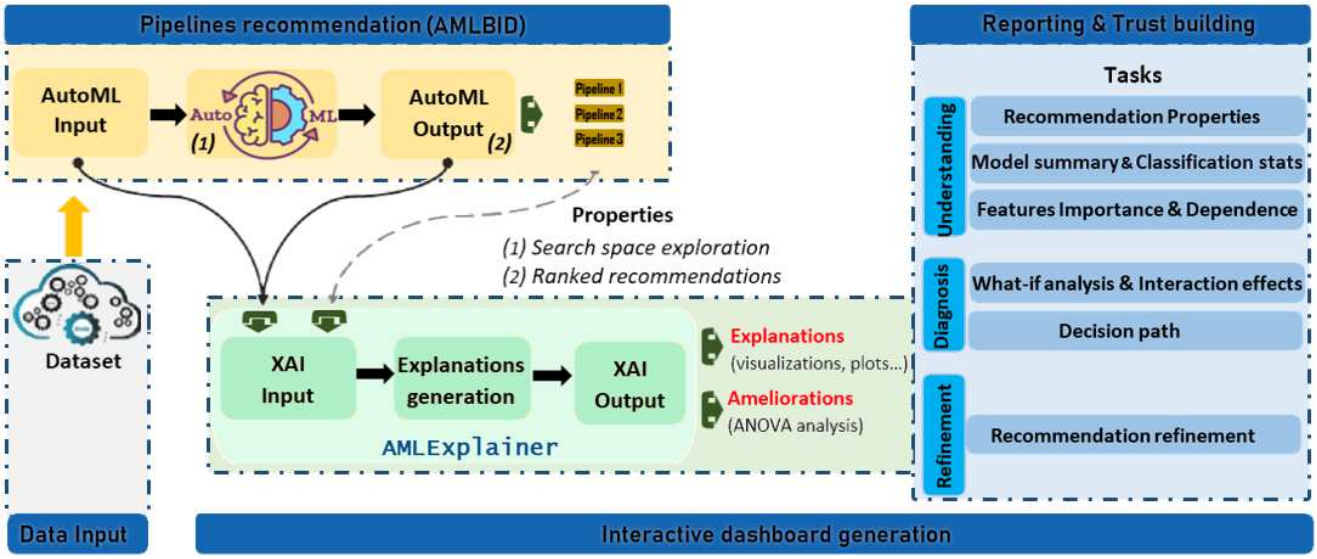

Fig. 3 The global architecture of the proposed white-box AutoML (AMLBID and AMLExplainer).

manufacturing levels generating more than 3 millions of different ML configurations (pipelines). Each pipeline consists of a choice of a machine learning model and its hyperparameters configuration. By exploring the interactions between datasets and pipelines topology, the system is able to identify effective pipelines without performing computationally expensive analysis.

Building a meta-learning based system to deal with the algorithms selection and configuration problem requires a meta-knowledge base for the learning process. This involves collecting datasets, choosing machine learning algorithms, extracting meta-features (datasets and pipelines characteristics), and determining the performance of the algorithm configurations according to different evaluation measures (e.g. Accuracy, precision, Recall). Though, when a new problem is presented to the system, the meta-features are extracted, and a recommendation mechanism that makes use of the meta-knowledge base provides the ranking of the pipeline(s) (algorithms and configurations) for the unseen problem according to the desired performance measure.

AMLBID consists of two main phases: the learning phase and the inferring one. During the learning phase, we evaluate different classification algorithms, analyze multiple datasets (to extract meta-features), and train a ranking meta-model. During the inference phase, the meta-model generated in the training phase is used to produce a ranked list of promising classification pipelines for a new dataset and a classification performance metric.

4.1.1 The learning phase

During the learning phase, we evaluate different classification algorithms with multiple hyper-parameters configurations on a large collection of various datasets. Then, we generate meta-features that are used to train a meta-model able to recommend promising classification pipelines for a given dataset and performance metric. The entire training phase is illustrated in Figure 4.

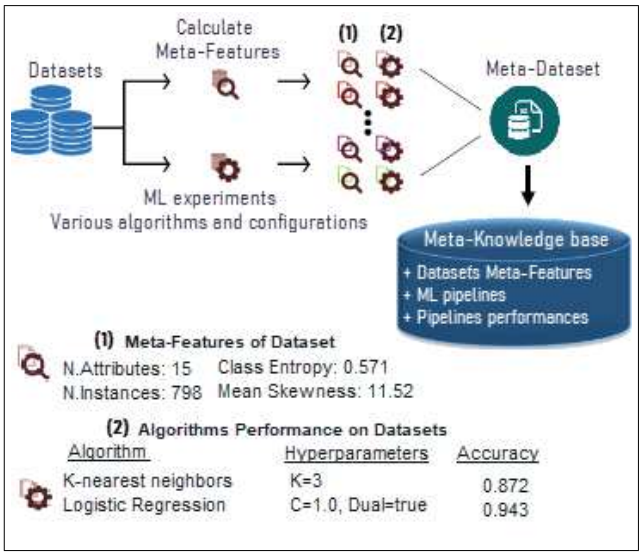

Fig. 4 Learning phase workflow.

The datasets: We conduct the experiments on 360 real-world manufacturing classification datasets which

are collected from the popular UCI[5] , OpenML[6] , Kaggle[7] and KEEL[8] repositories, among other real world scenarios. These datasets cover various tasks with respect to their size, the number of attributes, their composition and class imbalance. Datasets characteristics are illustrated in Table 4.

Table 4 Datasets's dimensions.

	Number of classes	Number of attributes	Number of instances
Min	2	5	1800
Max	18	1000	105908

The Meta-features : The meta-learning core paradigm is to relate the performance of learning algorithms and configurations to data characteristics (Meta-features). Meta-features are common characteristics of several problems, and their aim is to identify structural similarities and differences among problems. These characteristics can be divided into three categories [28] :

Simple : based on general measures, such as the number of instances, attributes and classes, dataset dimensionality. They are designed to some extent to measure the complexity of the underlying problem.

Statistical : based on statistical measures obtained from the dataset attributes, such as means, standard deviation, class entropy, and correlations, etc.

Landmark : that characterize the extent of datasets when basic machine learning algorithms (with default configuration) are performed on them. The used landmark characteristics in our system include performance of the linear discriminant analysis (LDA), Gaussian Naive Bayes (GNB), Decision Trees (DT), Gaussian Naive Bayes (GNB) and the K-Nearest Neighbor (KNN) landmarks.

The pipelines generation : To build the Meta-knowledge base, we used 08 classifiers from the popular Python-based machine learning library, Scikit-learn. These classifiers are *AdaBoost, Support Vector Classifier (SVC), Extra Trees, Gradient Boosting, Decision Tree, Logistic Regression, Random Forest, and Stochastic Gradient Descent (SGD)* classifiers. Detailed description of the algorithms and their tuned hyper-parameters are described on the Table A1-A7 in the Appendix.

For each run of a classifier $\boldsymbol{C}$ over a dataset $\boldsymbol{D}$, we generated 1000 different combinations of their hyper-parameters configurations. This process resulted in an average of 8000 pipelines per dataset. In particular, for each classifier, we have generated a list of all possible and reasonable combinations where we conducted, for each dataset, a random search among them [42].

During the training phase, we used a 5-fold stratified cross-validation strategy to construct our meta-datasets. As a result, our knowledge base consists of more than 3 millions evaluated classification pipelines. It is noted that due to the different number of algorithms hyper-parameters, not every algorithm had the same number of configurations / evaluations.

Measures : As part of our core idea, we aim to recommend high-performing ML pipelines for a given combination of datasets and evaluation measure. The point that most of state-of-the-art systems do not take into account, the proposed system supports various classification performance measures to evaluate the performance of the ML pipelines (ML algorithms and related hyper-parameters configuration). Table 5 shows supported measures details.

4.1.2 The recommending phase

The *recommending phase* is initiated when a new dataset to be analyzed occurs. At this point, the user selects a predictive analytic metric to be used for the analysis (e.g. Accuracy, Recall, F1 score), and then the system automatically recommends a set of machine learning algorithms and their related hyper-parameters configuration to be applied, such that the predictive performance is the first-rate. To do so, the system first, extracts the dataset characteristics (meta-features). Then, the extracted meta-features are fed to the meta-model to provide the candidate pipelines. Finally, the suggestion engine, according to the meta-knowledge base, ranks the pipelines in respect to the provided analytic metric. The recommending process is shown in figure 5.

Meta-model : After having generated a meta-dataset with all the necessary metadata, the goal is to build a predictive *meta-model* that can learn the complex relationship between a task meta-features and the utility of specific ML pipelines to recommend the most useful configuration Θ_{new} given the meta-features M of a new task t_{new}.

[5] https://archive.ics.uci.edu/
[6] https://www.openml.org/
[7] https://www.kaggle.com/
[8] https://sci2s.ugr.es/keel/

Table 5 Supported classification measures.

Measures	Description	Importance
Precision	Precision considered as a measure of exactness or quality.	Precision is used to retrieve fraction of instances that are relevant.
Recall	Recall is a measure of completeness or quantity.	Recall is used to retrieve fraction of relevant instances that are retrieved.
Accuracy	The accuracy is the proportion of the total number of predictions that were correct. Accuracy is related to the degree of bias in the measurements	Accuracy is used to represent the correct answer or percentage of accurate classification.
F1 score	F1 score or F-measure is defined as the harmonic mean of precision and recall. Commonly used as a single metric to evaluate the classifier performance.	A value closer to one implies that a better combined precision and recall is achieved by the classifier

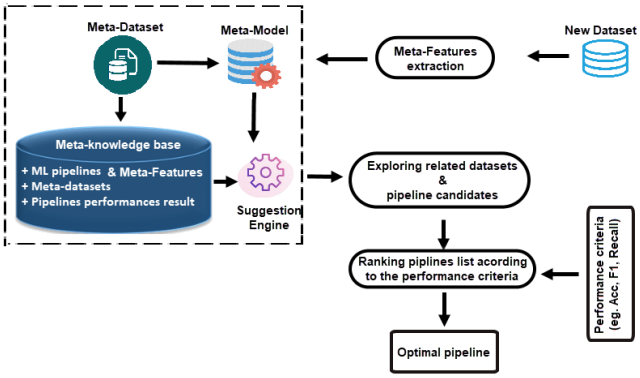

Fig. 5 Recommending phase workflow.

Formally, each task $t_j \in T$ is described by a vector $m(t_j) = (m_{j,1}, \ldots, m_{j,K})$ of K meta-features $m_{j,K} \in M$, the set of all known meta-features. This can be used to define the task similarity measure based on, for instance, the *Euclidian distance* between $m(t_i)$ and $m(t_j)$, so that we can transfer information from the most similar tasks to the new task t_{new}.

One of the aims of our work is to produce an enriched meta-model able to recommend the top-performing classification configuration(s) for a combination of an unseen dataset and classification evaluation measure. For this purpose, two state of the art learning algorithms were used to produce meta-models able to predict the most appropriate pipelines for the dataset at hand : Random Forest (RF) and k-Nearest Neighbor (k-NN). Thus, when the meta-learning system is applied to a new dataset, the meta-model returns a ranking of the most suitable classification algorithms and configurations, based on its meta-feature values.

Ranking using KNN classifier is a commonly used strategy to obtain the top-K rankings. When a new dataset is presented to the meta-learning system, the k-NN identify the closest neighbors of the candidate dataset in the meta-knowledge base, using the Euclidian distance metric, a weighted average of each individual neighbor's is used for forecasting the optimal pipeline configuration based on the relevant measure. While for the Random forest meta-model, we produce for each supported classification evaluation measure M_i a large labeled training set using the following process :

1. For each combination of $d \in D$ and $l_c \in L$ (D is the 360-learning datasets, l_c a learning algorithm configuration from the 3 millions evaluated configurations) we retrieve the set of all best predictive results $R(m, d, l_c)$ for each evaluation metric m (e.g., accuracy, F1-score, recall and precision).
2. For each $d \in D$ we designate the learner algorithm configuration l_c as *Class 1* (top performer algorithm configuration for the dataset) if its best predictive results for the dataset are greater than or equal to the highest performance achieved by all other algorithm configurations. Otherwise we label the l_c for the dataset as *Class 0* (low performer algorithm configuration).
3. For each combination of $d \in D$ and $l_c \in L$ we generate a joint set $M = \{M_d \cup M_{l_c}\}$, where :
   - M_d : the dataset's meta-features generated in the learning step.
   - M_{l_c} : a discrete feature describing the learning configuration l_c.
4. The joined meta-features vectors M are used to fit the RF meta-model for the top performing algorithms configurations, using the meta-features variables as predictors and the learner's labels as targets of the meta-model. For our Meta-Model, we have been mainly interested for optimizing the prediction recall of Class 1 (the classifier has the potential to be among the best performing classifiers). Therefore, we had to consider different levels of the decision tree model hyperparameters configuration where the configuration: $\{class\_weight : \{1 : 1, 0 : 0.7\}, criterion : gini, max\_features : None\}$ provided the best meta-model result.
   Figure 6 presents Random Forest and KNN meta-models performances on suggesting the best predictive pipeline configuration. The KNN based meta-

model clearly performs better than the random forest classifier based meta-learner according to the accuracy metric.

4.1.3 The evaluation of robustness

In this evaluation, we investigate the performance that can be achieved by using the proposed recommending module on various manufacturing related problems. We evaluate its ability to predict the ML algorithms with related hyper-parameters configurations that shall provide the best result of the analysis. We benchmark on a highly varied selection of datasets (30 datasets) covering binary and multi-class classification problems from different industry 4.0 levels with a sample size from 1000 to 90000 instance (this is a common sample size for general real-world datasets).
The proposed system uses the meta-model to predict all the pipelines in the meta-knowledge base with respect to the analyzed dataset and then returns its top-ranked pipelines according to the provided performance criteria. The top-ranked pipelines are then fitted on the datasets that was split into train and test sets using a 70% / 30% ratio. The results of this evaluation were used to compare the performances of AMLBID to those of the TPOT and Auto-sklearn state-of-the-art frameworks. It is important to stress again, that while the majority of state-of-the-art frameworks evaluate a set of pipelines by running them on the dataset at hand before the recommendation step, which demands considerable computational budget that is not always available and takes a huge amount of time in the majority of cases, making them unpractical solutions in real world problems as in the industrial ones, AMLBID immediately produces a ranked list of potential top pipelines configurations using its meta-model and meta-knowledge base at an imperceptible computational cost (in term of time and computational resources). The evaluation results are presented in Table 6.

Table 6 Performances of selected AutoML frameworks on the benchmark datasets.

Dataset	AMLBID	TPOT	Auto-sklearn
[43]	**93.74**	91.20	92.83
[44]	**99.41**	99.07	99
[45]	**97.06**	95.17	93.56
APS Failure	99.10	**99.33**	98.4
Higgs	72.6	72.6	**72.9**
CustSat	**85.59**	82.76	82.90

As shown in Table 7, it is clear that the performances of AMLBID are comparable and even better than those of the baselines even though it does not run any pipeline on the dataset prior to the recommendation.

Table 7 Number of datasets for each baseline AutoML system for which the optimal algorithm configuration has been recommended.

System	Number of Datasets with Top Performance
Autosklearn	3 (10%)
TPOT	6 (20%)
AMLBID	21 (70%)

As state of the art systems support only the predictive accuracy as the performance measure of the recommended configuration, further robustness comparison on different performance measures such *Recall*, *F1 score* and *Precision* could not be done. Whereas in some concrete cases, Recall or Precision may be more important and informative than the predictive accuracy. Therefore, the proposed system is the first AutoML system to support different predictive performance measures (Precision, Recall, Accuracy and F1score).

Beyond their black box nature, one of the most shortcomings of AutoML solutions is their computational complexity, requiring a huge budget of time and resources. On the contrary, AMLBID has the advantage of the *O(1)* computational complexity, generating the recommendation in a negligible amount of time. Table 8 presents the running time of AMLBID, TPOT and Autosklearn on the benchmarked datasets.

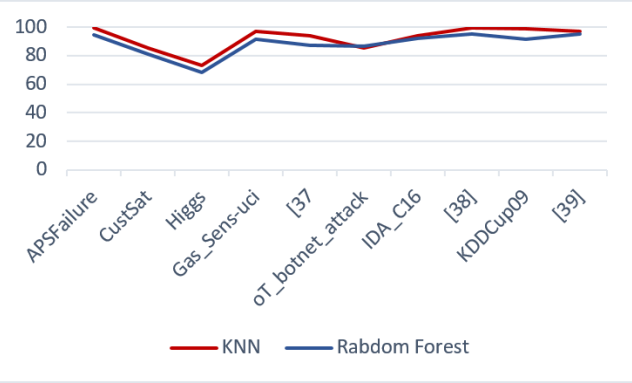

Fig. 6 Predictive performance of the KNN and RF meta-models.

Noting that the rather "long" time taken by some massive datasets for AMLBID relates to the calculations made for extracting the dataset's characteristics (Metafeatures).

Table 8 The running time of AMLBID and the baseline AutoML frameworks on the benchmark datasets. The running time format is HH:MM:SS.

Dataset	Ds size	AMLBID	TPOT	Autosklearn
[43]	95	**00:00:05**	00:08:14	01:23:47
[44]	61000	**00:05:29**	03:42:09	04:19:05
[45]	2000	**00:00:12**	00:13:57	01:49:21
APS Failure	60000	**00:05:39**	05:23:35	03:58:39
Higgs	110000	**00:06:16**	05:43:24	07:37:55
CustSat	76020	**00:04:06**	04:09:36	05:07:03

4.2 The explainer module

AMLExplainer is implemented as a client-server tool integrated with the recommender module. The server coordinates as an AutoML support system. As the client, the visual interface provides graphical interaction with AutoML results and maps the summary data for visualization through a set of different visual summary levels of the recommended models. AMLExplainer users are allowed to explore the models provided by the AutoML process at four main levels of detail (i.e. AutoML Overview, Recommendation-level View, What-if analysis-level View, and Refinement-level View). Meanwhile, AMLExplainer provide end users with a guidance, when AutoML returns unsatisfying results, to improve the predictive performances. Thence increases the transparency, controllability, and the acceptance of AutoML. The tool documentation and a detailed list of features with an illustrative example is available in the github repository [9].
The workflow of the proposed auto-explanatory AutoML system consists of two main components :

- The AutoML component, which shows the high-level of the AutoML process from recommendations to refinements.
- The recommended configuration component, that allows users to inspect the recommended model's inner working and decision's generation process (include the Recommendation-level and What-if analysis-level views).

The AutoML Overview : the AutoML overview level (Figure 7) summarizes high-level information of the AutoML process. Users will be able to compare and choose between the top K recommended configurations. They can focus their analysis on the top model configuration on the next level view, which highlights the corresponding algorithm in the detail views.
The recommendation-level View : the recommendation-level view enables users to inspect recommendations with respect to performance distribution. As shown in (Figure 8), a detailed explanation about the top performed recommendation is generated through multiple granularity levels, such as statistics about the configuration performances (Figure 8(A)), and a tree based explanation of the conducted predictions (Figure 8(B)).

By providing intelligible explanations about the process and reasoning behind an individual prediction, as illustrated in the Figure 8, it is clear that the decision-maker whether a manufacturing engineer or a machine learning practitioner is much better positioned to make decisions since He / She usually have prior knowledge about the data and the application domain, which can use to trust in and accept or reject a prediction if the reasoning behind it is well explained.
The What-if analysis-level View : (Figure 9) is designed to investigate the machine learning models. It enables understanding models by enabling end users to investigate attribution values for individual input features in relation to model predictions. Explaining the inner working of the model helps to gain an understanding of what the model does and does not do. This is important so that they can gain an intuition for when the model is likely missing information and may have to be overruled. Therefore explore scenarios, test, and evaluate / validate business assumptions, and gain intuition for modification.
The refinement-level View : (Figure 10) shows the correlation between performances and hyperparameters of a recommended algorithm. To accomplish that, we takes as input performances data gathered with different hyperparameter settings of the algorithm (from the recommender module's knowledge-base) , fits a random forest to capture the relationship between hyperparameters and performances, and then we apply functional Analysis of variance (ANOVA [46]) to assess how important each of the hyperparameters and each low-order interaction of hyperparameters is to performance. Guided by this in-depth analysis, end users have a guidance, when AutoML returns unsatisfying results, to improve to predictive performances.

4.3 Demonstration test case : Application to Manufacturing Quality Prediction

Our auto-explainer module works for any predictive modeling problem where machine learning is used. As our work's goal is to show the feasibility to achieve maximum possible performance for a specific predictive modeling problem and automatically explaining results for any machine learning predictive model, we evaluated our automatic explanation method on a Manufacturing Quality Prediction use case. The data contains

[9] https://github.com/LeMGarouani/AMLBID

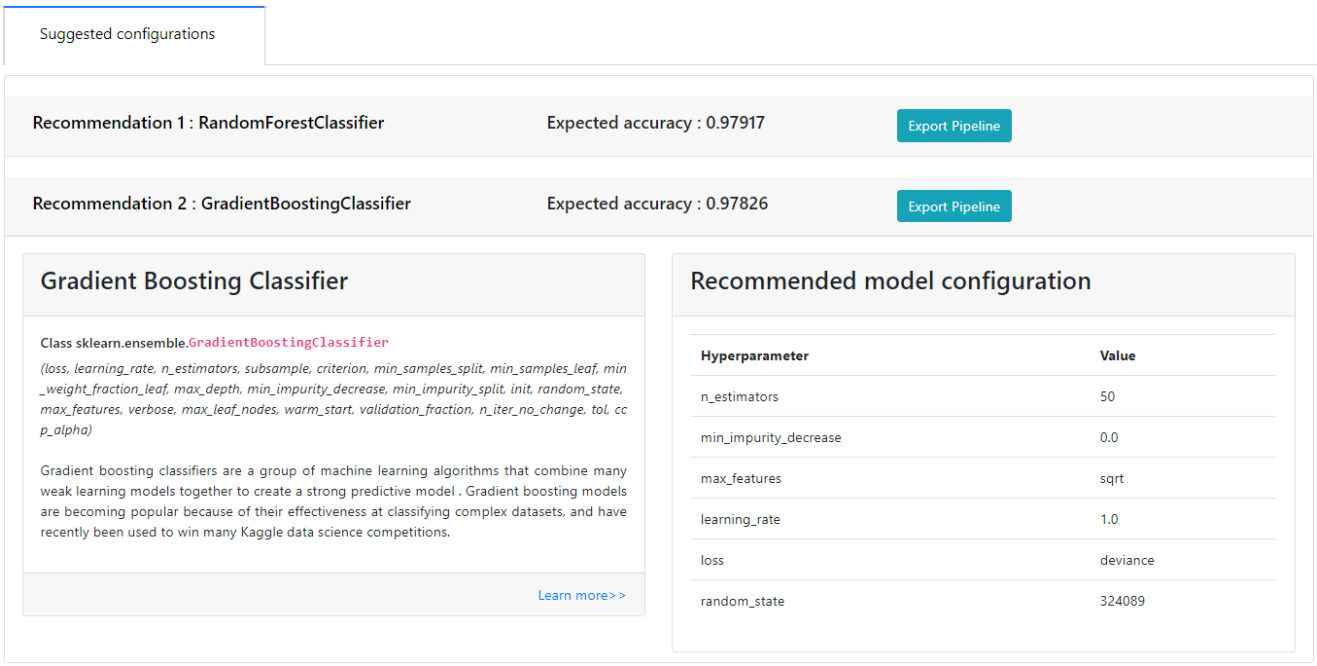

Fig. 7 AutoML overview.

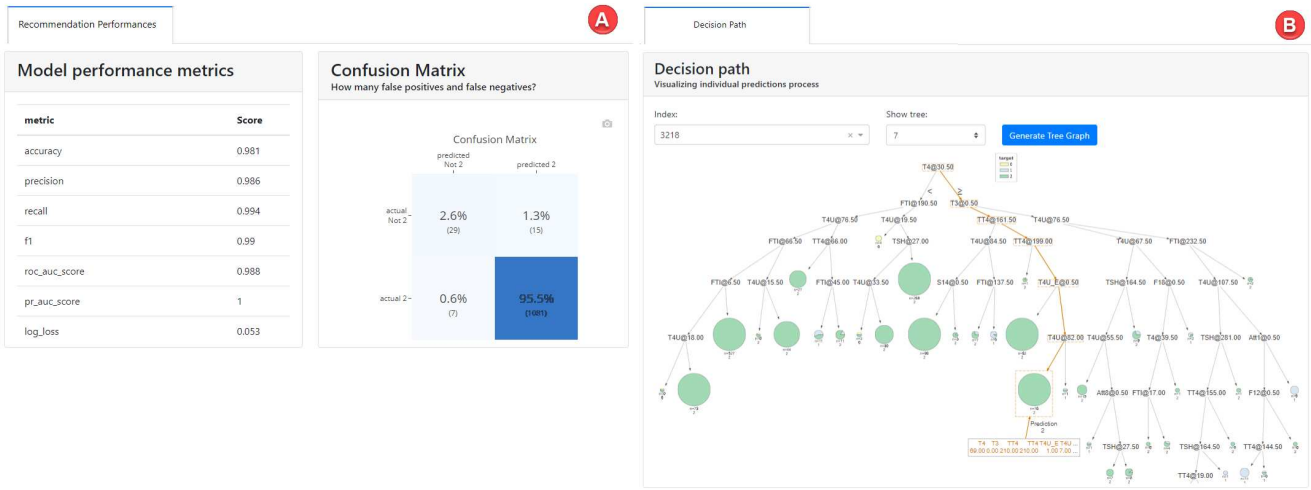

Fig. 8 Recommendation-level view.

187.156 historical 1-year records of a production unit. Among these records, 74.39% were diagnosed as compliant products.

4.3.1 Expert interview

To evaluate the proposed white-box AutoML system as a decision support system, we conduct interviews with two closely collaborating experts (E1 and E2) with particular expertise in machine learning. We collected their feedback about the AutoML module as a black-box decision support system assisting the experts to choose and configure ML models for their problems initially and afterwards with the entire system (recommender module and the explanatory one). Based on their feedback, we summarize two main appreciation of the proposed tool:

– AutoML can help stakeholders (neophyte as well as experts) to better apply machine learning algorithms. AutoML enables quick experimentations with a large number of models and configurations, whose results could provide useful knowledge to ML researchers and domain practitioners. On the test set, the recommended machine learning predictive model configuration achieved an accuracy of 97.81% while their configuration based on their understanding of the algorithms and their observation

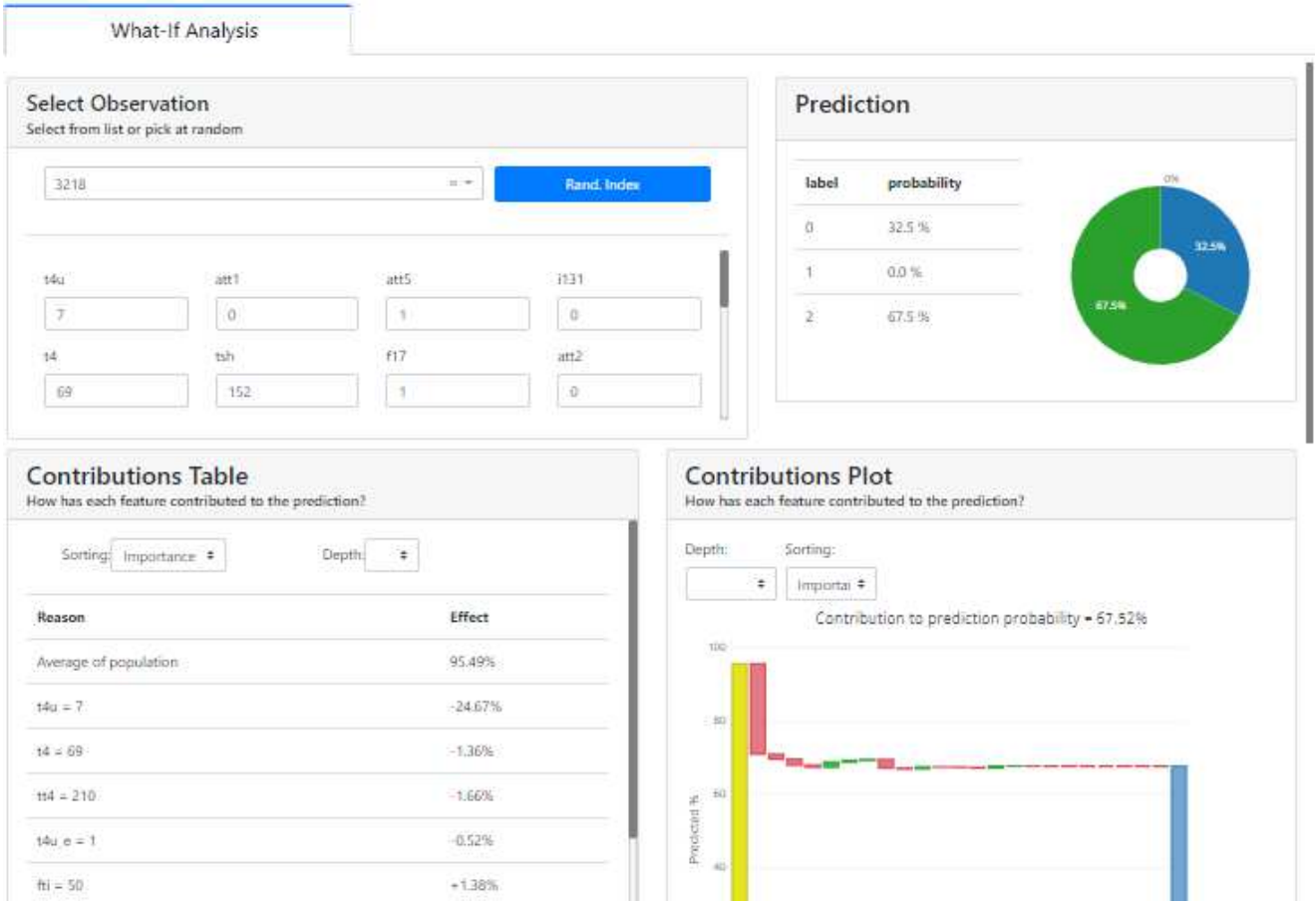

Fig. 9 What-if analysis-level view.

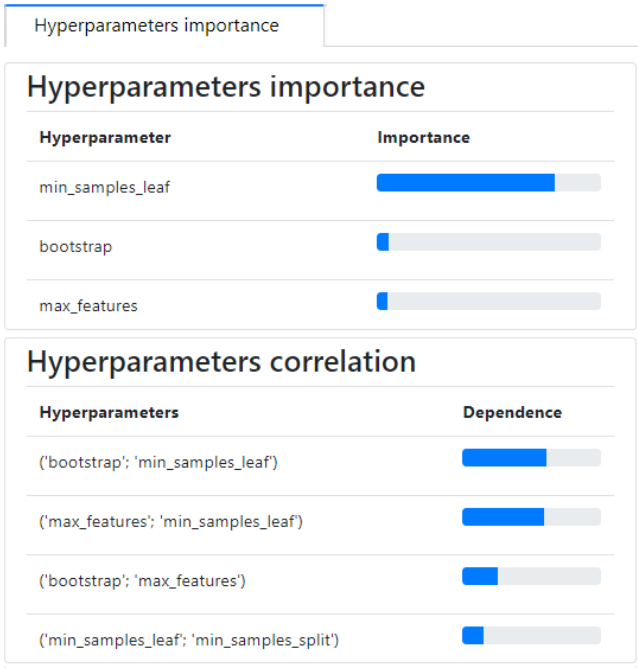

Fig. 10 Refinement-level view.

of the data achieved a predictive accuracy of 91.42%. These findings can inform users of the importance of hyperparameter tuning for ML algorithms. E2 commented that being able to match prior knowledge about machine learning to the visualizations produced by AMLExplainer creates confidence in the underlying AutoML process and increases the likelihood of adopting AutoML.

– Both experts appreciated the human-machine interaction introduced in AMLExplainer. They believed such interaction could improve an AutoML process and enhance user experience and make such powerful black boxes trusty. E1 commented, "Users with more domain knowledge, such as myself, are usually critical of automated methods and like to be in control. I do not like getting a score back and hearing *trust me*".

Our well-documented real-world evaluation case illustrates how to overcome the intransparency problem of AutoML systems as decision support systems, namely, the absence of human interaction and analysis of the inner working and reasoning of such tools. This could extend the use of and trust in the intelligent AutoML

systems to areas where they were so far neglected due to their insistence on comprehensible models. Separating the selection and configuration of machine learning algorithms from model explanation is another benefit of expert and intelligent systems.

5 Discussion and conclusion

There has been significant progress in democratizing the application of ML to non-experts of data analysis by providing them with "off the shelf" solutions. However, these powerful support systems fail to provide detailed instructions about the recommended configurations and the inner working of these models, thence making them less trustworthy highly performant black-boxes. In this work, we present a novel transparent and self-explained AutoML system along with an interactive visualization module that supports machine learning experts and neophytes in analyzing the automatic results of an AutoML DSS.

To our knowledge, the proposed system is the first application of the general explanation methods of AutoML systems as decision support systems. We explore several levels of explanations, ranged from individual decisions to the entire model's recommendations and predictions. The explanations of the prediction models and what-if analysis proved to be an effective support for manufacturing related problems. A set of evaluations demonstrate the utility and usability of AMLBID in a real-world manufacturing problem. We show how powerful black-box ML systems could be made transparent and help domain experts to iteratively evaluate and update their beliefs. Based on the promising findings presented in this paper, further validation of the proposed framework in other real-world applications with a larger and more diverse group of users shall improve the visualization and presentation of explanations. We plan to provide the proposed system as an open source python package, which we are currently in process of publishing.

Acknowledgements The authors would like to thank all the participants involved in the system evaluation for their constructive discussions and valuable suggestions.

Funding This work is supported, in part, by School of engineering's and business' sciences and technics (HESTIM), Casablanca-Morocco, CNRST Morocco, and the Université du Littoral Côte D'Opale, Calais France.

Availability of data and materials All data generated or analyzed during this study are included in this paper.

Code availability Software code is included in the study github repository : `https://github.com/LeMGarouani/AMLBID`.

Declarations

Ethics approval All authors confirm that this article does not have any academic ethics issues and strictly follows the journal submission rules.

Consent to participate All authors agree to participate in the research work of this paper and publish it in the International Journal of Advanced Manufacturing Technology.

Consent for publication All authors agree to publish this article in the International Journal of Advanced Manufacturing Technology.

Conflict of interest The authors declare that they have no conflict of interest.

References

[1] F. Provost and T. Fawcett. "Data Science and Its Relationship to Big Data and Data-Driven Decision Making". In: *Big Data* 1.1 (2013), pp. 51–59. DOI: 10.1089/big.2013.1508.

[2] E. Brynjolfsson, L. M. Hitt, and H. H. Kim. *Strength in Numbers: How Does Data-Driven Decisionmaking Affect Firm Performance?* SSRN Scholarly Paper ID 1819486. Rochester, NY: Social Science Research Network, 2011. DOI: 10.2139/ssrn.1819486.

[3] W. Samek and K.-R. Müller. "Towards Explainable Artificial Intelligence". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Ed. by W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 5–22. DOI: 10.1007/978-3-030-28954-6.

[4] Y. LeCun, Y. Bengio, and G. Hinton. "Deep Learning". In: *Nature* 521.7553 (2015), pp. 436–444. DOI: 10.1038/nature14539.

[5] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym. "NVIDIA Tesla: A Unified Graphics and Computing Architecture". In: *IEEE Micro* 28.2 (2008), pp. 39–55. DOI: 10.1109/MM.2008.31.

[6] M. V. Nural, H. Peng, and J. A. Miller. "Using Meta-Learning for Model Type Selection in Predictive Big Data Analytics". In: *2017 IEEE International Conference on Big Data (Big Data)*. 2017, pp. 2027–2036. DOI: 10.1109/BigData.2017.8258149.

[7] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter. "Efficient and Robust Automated Machine Learning". In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 2755–2763.

[8] R. S. Olson and J. H. Moore. "TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning". In: *Automated Machine Learning: Methods, Systems, Challenges*. Ed. by F. Hutter, L. Kotthoff, and J. Vanschoren. The Springer Series on Challenges in Machine Learning. Cham: Springer International Publishing, 2019, pp. 151–160. DOI: 10.1007/978-3-030-05318-5_8.

[9] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown. "Auto-WEKA: Automatic Model Selection and Hyperparameter Optimization in WEKA". In: *Automated Machine Learning: Methods, Systems, Challenges*. Ed. by F. Hutter, L. Kotthoff, and J. Vanschoren. The Springer Series on Challenges in Machine Learning. Cham: Springer International Publishing, 2019, pp. 81–95. DOI: 10.1007/978-3-030-05318-5_4.

[10] T. Swearingen, W. Drevo, B. Cyphers, A. Cuesta-Infante, A. Ross, and K. Veeramachaneni. "ATM: A Distributed, Collaborative, Scalable System for Automated Machine Learning". In: *2017 IEEE International Conference on Big Data (Big Data)*. 2017, pp. 151–162. DOI: 10.1109/BigData.2017.8257923.

[11] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu. "Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges". In: *Natural Language Processing and Chinese Computing*. Ed. by J. Tang, M.-Y. Kan, D. Zhao, S. Li, and H. Zan. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 563–574. DOI: 10.1007/978-3-030-32236-6_51.

[12] M. T. Ribeiro, S. Singh, and C. 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. New York, NY, USA: Association for Computing Machinery, 2016, pp. 1135–1144. DOI: 10.1145/2939672.2939778.

[13] R. a. M. ltd. *Big Data Market by Component, Deployment Mode, Organization Size, Business Function (Operations, Finance, and Marketing and Sales), Industry Vertical (BFSI, Manufacturing, and Healthcare and Life Sciences), and Region - Global Forecast to 2025*.

[14] M. Cuartas, E. Ruiz, D. Ferreño, J. Setién, V. Arroyo, and F. Gutiérrez-Solana. "Machine Learning Algorithms for the Prediction of Non-Metallic Inclusions in Steel Wires for Tire Reinforcement". en. In: *Journal of Intelligent Manufacturing* (2020). DOI: 10.1007/s10845-020-01623-9.

[15] R. Medina, J. C. Macancela, P. Lucero, D. Cabrera, R.-V. Sánchez, and M. Cerrada. "Gear and Bearing Fault Classification under Different Load and Speed by Using Poincaré Plot Features and SVM". en. In: *Journal of Intelligent Manufacturing* (2020). DOI: 10.1007/s10845-020-01712-9.

[16] A. Jalali, C. Heistracher, A. Schindler, B. Haslhofer, T. Nemeth, R. Glawar, W. Sihn, and P. De Boer. "Predicting Time-to-Failure of Plasma Etching Equipment Using Machine Learning". In: *2019 IEEE International Conference on Prognostics and Health Management (ICPHM)*. 2019, pp. 1–8. DOI: 10.1109/ICPHM.2019.8819404.

[17] D. Wolpert and W. Macready. "No free lunch theorems for optimization". In: *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 67–82. DOI: 10.1109/4235.585893.

[18] M. Reif, F. Shafait, M. Goldstein, T. Breuel, and A. Dengel. "Automatic Classifier Selection for Non-Experts". In: *Pattern Analysis and Applications* 17.1 (2014), pp. 83–96. DOI: 10.1007/s10044-012-0280-z.

[19] B. Bilalli, A. Abelló, T. Aluja-Banet, and R. Wrembel. "Automated Data Pre-Processing via Meta-Learning". In: *Model and Data Engineering*. Ed. by L. Bellatreche, Ó. Pastor, J. M. Almendros Jiménez, and Y. Aït-Ameur. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 194–208. DOI: 10.1007/978-3-319-45547-1_16.

[20] B. Bilalli, A. Abelló, T. Aluja-Banet, R. F. Munir, and R. Wrembel. "PRESISTANT: Data Pre-Processing Assistant". In: *Information Systems in the Big Data Era*. Ed. by J. Mendling and H. Mouratidis. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, 2018, pp. 57–65. DOI: 10.1007/978-3-319-92901-9_6.

[21] U. Khurana, H. Samulowitz, and D. Turaga. "Feature Engineering for Predictive Modeling Using Reinforcement Learning". In: *arXiv e-prints* 1709 (2017), arXiv:1709.07150.

[22] F. Nargesian, H. Samulowitz, U. Khurana, E. B. Khalil, and D. Turaga. "Learning Feature Engi-

neering for Classification". In: (2017), pp. 2529–2535.

[23] R. Vainshtein, A. Greenstein-Messica, G. Katz, B. Shapira, and L. Rokach. "A Hybrid Approach for Automatic Model Recommendation". In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. CIKM '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1623–1626. DOI: 10.1145/3269206.3269299.

[24] M. Feurer, J. T. Springenberg, and F. Hutter. "Initializing Bayesian Hyperparameter Optimization via Meta-Learning". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI'15. Austin, Texas: AAAI Press, 2015, pp. 1128–1135.

[25] B. Komer, J. Bergstra, and C. Eliasmith. "Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn". In: *Proceedings of the 13th Python in Science Conference* (2014), pp. 32–37. DOI: 10.25080/Majora-14bd3278-006.

[26] H. Jin, Q. Song, and X. Hu. "Auto-Keras: An Efficient Neural Architecture Search System". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1946–1956. DOI: 10.1145/3292500.3330648.

[27] M. Garouani., A. Ahmad., M. Bouneffa., A. Lewandowski., G. Bourguin., and M. Hamlich. "Towards the Automation of Industrial Data Science: A Meta-learning based Approach". In: *Proceedings of the 23rd International Conference on Enterprise Information Systems - Volume 1: ICEIS,* INSTICC. SciTePress, 2021, pp. 709–716. DOI: 10.5220/0010457107090716.

[28] M. Maher and S. Sakr. *SmartML: A Meta Learning-Based Framework for Automated Selection and Hyperparameter Tuning for Machine Learning Algorithms*. 2019. DOI: 10.5441/002/edbt.2019.54.

[29] D. Shin and Y. J. Park. "Role of Fairness, Accountability, and Transparency in Algorithmic Affordance". en. In: *Computers in Human Behavior* 98 (2019), pp. 277–284. DOI: 10.1016/j.chb.2019.04.019.

[30] R. L. Heath and J. Bryant. *Human Communication Theory and Research: Concepts, Contexts, and Challenges*. English. 2nd edition. Mahwah, N.J: Routledge, 2000.

[31] D. Gunning, M. Stefik, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang. "XAI—Explainable Artificial Intelligence". In: *Science Robotics* 4.37 (2019). DOI: 10.1126/scirobotics.aay7120.

[32] D. Castelvecchi. "Can We Open the Black Box of AI?" In: *Nature News* 538.7623 (2016), p. 20. DOI: 10.1038/538020a.

[33] M. T. Ribeiro, S. Singh, and C. Guestrin. "Anchors: High-Precision Model-Agnostic Explanations". en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2018).

[34] A. W. Harley. "An Interactive Node-Link Visualization of Convolutional Neural Networks". In: *Advances in Visual Computing*. Ed. by G. Bebis et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 867–877. DOI: 10.1007/978-3-319-27857-5_77.

[35] S. M. Lundberg et al. "From Local Explanations to Global Understanding with Explainable AI for Trees". In: *Nature Machine Intelligence* 2.1 (2020), pp. 56–67. DOI: 10.1038/s42256-019-0138-9.

[36] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. "Explaining Nonlinear Classification Decisions with Deep Taylor Decomposition". en. In: *Pattern Recognition* 65 (2017), pp. 211–222. DOI: 10.1016/j.patcog.2016.11.008.

[37] M. D. Zeiler and R. Fergus. "Visualizing and Understanding Convolutional Networks". In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 818–833. DOI: 10.1007/978-3-319-10590-1_53.

[38] J. Müller, M. Stoehr, A. Oeser, J. Gaebel, M. Streit, A. Dietz, and S. Oeltze-Jafra. "A Visual Approach to Explainable Computerized Clinical Decision Support". en. In: *Computers & Graphics* 91 (2020), pp. 1–11. DOI: 10.1016/j.cag.2020.06.004.

[39] T. Spinner, U. Schlegel, H. Schäfer, and M. El-Assady. "explAIner: A Visual Analytics Framework for Interactive and Explainable Machine Learning". In: *IEEE Transactions on Visualization and Computer Graphics* 26.1 (2020), pp. 1064–1074. DOI: 10.1109/TVCG.2019.2934629.

[40] Q. Wang, Y. Ming, Z. Jin, Q. Shen, D. Liu, M. J. Smith, K. Veeramachaneni, and H. Qu. "ATMSeer: Increasing Transparency and Controllability in Automated Machine Learning". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. New York,

NY, USA: Association for Computing Machinery, 2019, pp. 1–12. DOI: 10.1145/3290605.3300911.

[41] C. Lemke, M. Budka, and B. Gabrys. "Metalearning: A Survey of Trends and Technologies". In: *Artificial Intelligence Review* 44.1 (2015), pp. 117–130. DOI: 10.1007/s10462-013-9406-y.

[42] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. "Algorithms for Hyper-Parameter Optimization". In: *Proceedings of the 24th International Conference on Neural Information Processing Systems*. NIPS'11. Red Hook, NY, USA: Curran Associates Inc., 2011, pp. 2546–2554.

[43] R. K. Mazumder, A. M. Salman, and Y. Li. "Failure Risk Analysis of Pipelines Using Data-Driven Machine Learning Algorithms". en. In: *Structural Safety* 89 (2021), p. 102047. DOI: 10.1016/j.strusafe.2020.102047.

[44] C. F. Costa and M. A. Nascimento. "IDA 2016 Industrial Challenge: Using Machine Learning for Predicting Failures". In: *Advances in Intelligent Data Analysis XV*. Ed. by H. Boström, A. Knobbe, C. Soares, and P. Papapetrou. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 381–386. DOI: 10.1007/978-3-319-46349-0_33.

[45] S. Saravanamurugan, S. Thiyagu, N. R. Sakthivel, and B. Nair. "Chatter Prediction in Boring Process Using Machine Learning Technique". en. In: *Int. J. Manuf. Res.* (2017). DOI: 10.1504/IJMR.2017.10007082.

[46] J. N. Rouder, C. R. Engelhardt, S. McCabe, and R. D. Morey. "Model Comparison in ANOVA". In: *Psychonomic Bulletin & Review* 23.6 (2016), pp. 1779–1786. DOI: 10.3758/s13423-016-1026-5.

Appendices

Table A1 SVM hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
complexity (or: 'C')	[$1e^{-10}$, 500] (log-scale)	Soft-margin constant, controlling the trade-off between model simplicity and model fit.
Kernel	{'poly', 'rbf'}	The function of kernel is to take data as input and transform it into the required form (linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid).
coef0	[0., 10]	Additional coefficient used by the kernel (sigmoid kernel only).
gamma	[$1e^{-3}$, 1.01] (log-scale)	Length-scale of the kernel function, determining its locality.
Degree	[2, 3]	Degree for the 'poly' kernel.

Table A2 Random Forest & Extra Trees Hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
bootstrap	{true, false}	Whether to train on bootstrap samples or on the full train set.
Max_features	[0.1, 0.9]	Fraction of random features sampled per node.
Min_samples_leaf	[1, 20]	The minimal number of data points required in order to create a leaf.
Min_samples_split	[2, 20]	The minimal number of data points required to split an internal node.
imputation	mean, median, mode	Strategy for imputing missing numeric variables.
split criterion	{entropy, gini}	Function to determine the quality of a possible split.

Table A3 Adaboost Hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
algorithm	{SAMME, SAMME.R}	Determines which boosting algorithm to use.
N_estimators	[50, 501]	Number of estimators to build.
learning rate	[0.01, 2.0] (log-scale)	Learning rate shrinks the contribution of each classifier.
Max_depth	[1, 11]	The maximal depth of the decision trees.

Table A4 Decision Trees Hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
max features	[0.1, 0.9]	Number of features to consider when computing the best node split.
min_samples_leaf	[1, 21]	The minimum number of samples required to be at a leaf node.
Min_samples_split	[2, 21]	The minimum number of samples required to split an internal node.
criterion	{'entropy', 'gini' }	Function used to measure the quality of a split.

Table A5 Logistic Regression Hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
C	[$1e^{-10}$, 10.] (log-scale)	Regularization strength.
penalty	{'l2', 'l1' }	Whether to use Lasso or Ridge regularization.
Fit_intercept	True, False	Whether or not the intercept of the linear classifier should be computed.

Table A6 SGD Classifier Hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
loss	{'hinge','perceptron','log', 'squared_hinge','modified_huber'}	Loss function to be optimized.
penalty	{'l2', 'l1', 'elasticnet' }	Whether to use Lasso, Ridge, or ElasticNet regularization.
learning rate	{'constant', 'optimal', 'invscaling' }	Shrinks the contribution of each successive training update.
fit intercept	{True, False}	Whether or not the intercept of the linear classifier should be computed.
l1 ratio	[0., 1.]	Ratio of Lasso vs. Ridge regularization to use. Only used when the 'penalty' is ElasticNet.
eta0	[0., 5.]	Initial learning rate.
Power_t	[0., 5.]	Exponent for inverse scaling of the learning rate.

Table A7 Gradient Boosting Hyperparameters tuned in the experiments.

Hyperparameter	Values	Description
Learning_rate	[0.01, 1]	Shrinks the contribution of each successive decision tree in the ensemble.
criterion	{'friedman_mse', 'mse' }	The function to measure the quality of a split.
N_estimators	[50, 501]	Number of decision trees in the ensemble.
max depth	[1, 11]	Maximum depth of the decision trees. Controls the complexity of the decision trees
Min_samples_split	[2, 21]	The minimum number of samples required to split an internal node.
Min_samples_leaf	[1, 21]	The minimum number of samples required to be at a leaf node.