Advances in Artificial Intelligence and Machine Learning; Research 4 (4) 3071-3096 Received 15-09-2024; Accepted 20-12-2024; Published 27-12-2024

SML-AutoML: A Smart Meta-Learning Automated Machine Learning Framework

Ibrahim Gomaa i.gomaa@fci-cu.edu.eg
Faculty of Computers and Artificial Intelligence, Cairo University, Cairo, Egypt

Hoda M. O. Mokhtar h.mokhtar@fci-cu.edu.eg
Faculty of Computers and Artificial Intelligence, Cairo University, Cairo, Egypt
Faculty of Computing and Information Sciences, Egypt University of Informatics, Cairo, Egypt

Neamat El-Tazi n.eltazi@fci-cu.edu.eg
Faculty of Computers and Artificial Intelligence, Cairo University, Cairo, Egypt

Ali Zidane a.zidane@fci-cu.edu.eg
Faculty of Computers and Artificial Intelligence, Cairo University, Cairo, Egypt

Corresponding Author: Ibrahim Gomaa

Abstract

In recent years, Machine Learning (ML) and Automated Machine Learning (Auto-ML) have attracted significant attention. The ML pipeline involves repetitive tasks such as data preprocessing, feature engineering, model selection, and hyperparameter tuning. Developing a machine learning model demands considerable time for development, stress testing, and numerous experiments. Additionally, constructing a model with a limited search space of pipeline steps and various algorithms can take hours. As a result, Auto-ML has become widely adopted to reduce the time and effort required for these tasks. However, most current Auto-ML frameworks primarily concentrate on algorithm selection and hyperparameter optimization, known as CASH, while overlooking other critical ML pipeline steps like data preprocessing and feature engineering. This limited focus often results in suboptimal pipelines for specific datasets. Moreover, a significant number of frameworks overlook the integration of meta-learning, resulting in the promotion of high-performing pipelines customized for individual tasks rather than a universally optimal solution. Consequently, this deficiency necessitates the quest for a new pipeline tailored to each unique task, further underscoring the importance of a more comprehensive approach in Auto-ML frameworks. Additionally, while some Auto-ML frameworks address the entire pipeline, they often overlook the challenges posed by imbalanced datasets. To address these issues, we propose a novel and efficient meta-learning Auto-ML framework that effectively manages imbalanced datasets. The proposed framework outperforms state-of-the-art results in terms of accuracy, precision, recall,

Citation: Ibrahim Gomaa, et al. SML-AutoML: A Smart Meta-Learning Automated Machine Learning Framework. Advances in Artificial Intelligence and Machine Learning. 2024;4(4):176.

and time, demonstrating, on average, more than 5% improvement compared to the existing auto-ML frameworks. This paper also illustrates how our proposed framework outperforms current state-of-the-art solutions.

Keywords: Automated Machine Learning (Auto-ML), supervised learning, CASH, hyper-parameter optimization (HPO), Meta-learning

1. INTRODUCTION

The success of artificial intelligence and ML has been evident across various sectors and domains, garnering significant attention from both business and research communities. The effectiveness of ML algorithms largely depends on the availability of extensive datasets, making data a crucial and influential component of the ML process. The proliferation of the internet, social media, diverse applications, devices, and data sources has led to an unprecedented growth in data volume. Generally, the quality of data directly impacts the performance of ML outcomes. While ML has found broad application in fields like speech recognition [1], predictive analytics [2–6], image classification [7–9], text classification [10], and recommendation systems [11–14], a multitude of ML and deep learning techniques are being employed across various domains, particularly in the current era of Big Data [15, 16].The rise of Big Data calls for a robust data science presence to handle the daily flood of data. However, finding skilled data scientists for manual analysis is challenging. This has sparked interest in automating the ML process to streamline pipeline construction. Data scientists often conduct iterative experiments to build quality ML models, making the process complex and time-consuming. This process involves a series of steps, as illustrated in FIGURE 1 [17].

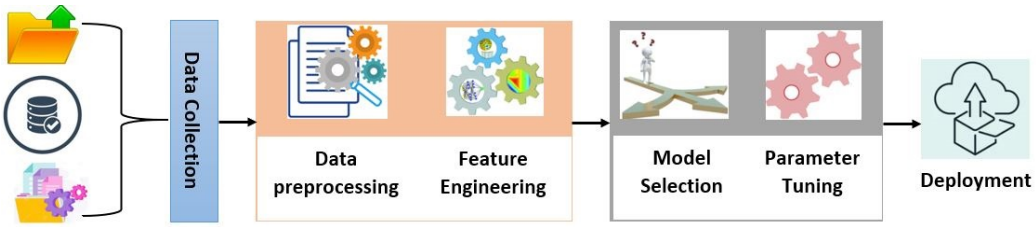

Figure 1: The standard ML pipeline

Data scientists encounter numerous challenges in their work. These challenges encompass a diverse array of potential ML algorithms that must be navigated, including XGBoost, linear regression, SVM, decision tree, K-nearest neighbor, neural network, random forest, and Bayesian classifier. The nature of the problem at hand, whether supervised or unsupervised, dictates the selection of algorithms that are applicable and suitable for the specific task. Additionally, data scientists are tasked with fine-tuning a set of hyperparameters for the chosen algorithm. Evaluating the model's performance involves a variety of metrics like accuracy, recall, precision, and the F1 score, adding another layer of complexity in selecting the most appropriate metric. Decisions made by data scientists at each stage significantly impact the final performance results and the quality of the ML model being developed [18–20]. Experimenting with multiple ML classifiers on the same dataset often yields divergent outcomes and performance metrics, typically necessitating human intervention in the decision-making process. While human-in-the-loop ML offers a practical

approach to optimizing the manual ML process, it remains resource-intensive, time-consuming, and reliant on manual labor. Consequently, there has been an increase in interest in automating the entire process of constructing an ML pipeline. Presently, Auto-ML has gained significant traction as a means of automating ML procedures, emerging as a crucial research area due to the rapid proliferation of ML across various industries and domains. Auto-ML, a term denoting techniques that automate the implementation and development of ML models, aims to deliver high-performance solutions within specified time constraints [21].

Initially, Auto-ML was designed to automate a particular component of the ML workflow, namely the Combined Algorithm Selection and Hyperparameter optimization (CASH) [22]. CASH refers to a process that involves both selecting the most suitable ML algorithm for a given task and optimizing the hyperparameters associated with that algorithm. CASH aims to enhance efficiency, performance, and the overall quality of ML models by finding the best algorithm and hyperparameter settings for a specific dataset and problem. Over time, Auto-ML expanded to include broader components of the ML workflow, such as feature engineering, feature preprocessing, model selection, and model interpretability. As Auto-ML progressed, and to circumvent the significant time investment required to construct a ML model for a specific dataset by exhaustively testing all pipelines, meta-learning has been implemented to leverage insights from past experiments, eliminating the necessity to initiate each pipeline from the beginning. Meta-learning refers to the process of developing AI systems capable of adapting to new tasks and enhancing their performance without requiring extensive retraining. These algorithms typically involve training a model across a spectrum of diverse tasks to acquire transferable knowledge that can be applied to new tasks. This contrasts with traditional ML, where a model is typically trained for a singular task and exclusively utilized for that specific task.

While current Auto-ML frameworks have made progress in reducing the manual effort involved in constructing ML models, they struggle with effectively managing imbalanced datasets, which are prevalent across various real-world applications and industries. These imbalanced datasets are found in contexts such as fraud detection in banking transactions, fraud detection in auditing processes, ransomware detection in backup systems, cancer detection in medical diagnosis, prediction of machine failures, identification of cyber threats, and predicting customer churn. Given the widespread reliance on imbalanced datasets, it is crucial to identify an optimal approach for handling such data. Several essential tasks must be undertaken when preparing imbalanced datasets before constructing an ML model, including data resampling, transformation, and preparation. These tasks play a critical role in enabling the model to learn efficiently from the data without being biased by the majority class. For instance, consider a scenario where a training dataset for banking transactions, D, contains 10,000 records (transactions) with a target variable, C, having two distinct values (non-fraud, fraud). In this dataset, 9,900 records are assigned the value “non-fraud”, while 100 records are assigned the value “fraud”. After training an ML classifier to predict the target variable C, a test dataset with 1,000 records is used for evaluation, comprising 950 records with the value “non-fraud” and 50 records with the value “fraud”. When the trained model is employed to predict the classes for the test dataset, it assigns all instances the value “non-fraud” and fails to predict any instances with the value “fraud”. This issue arises due to the model being biased by the majority class value “non-fraud”, which accounts for 99% of the target class. The primary function of the model is to detect fraudulent transactions, a task in which it failed to identify any instances.

Existing Auto-ML frameworks still face limitations such as not including all ML pipeline steps and focusing solely on the CASH component in specific frameworks like OAML [23], AMLBID [24], and Auto-Weka [25]; lacking integration with meta-learning techniques in certain frameworks like Auto-Weka, TPOT [26], H2O [27], ML-Plan [28], STREAMLINE [29], OAML, AutoKeras [30], LightAutoML [31], and Naive AutoML [32]; ineffective management of imbalanced datasets across all frameworks; and a requirement for prior technical proficiency. To address these challenges, the proposed framework has been developed. This framework is designed with specific objectives in mind. Firstly, it aims to integrate meta-learning, utilizing past ML pipeline performance to expedite learning for new tasks. Secondly, it addresses the issue of imbalanced datasets to ensure accurate and unbiased predictions. Thirdly, it is intended to be user-friendly for individuals with varying technical backgrounds. Lastly, the framework provides intelligent pipeline recommendations, reducing the need for trial-and-error experimentation. Overall, our framework aims to provide an Auto-ML solution that is efficient and effective for a diverse range of tasks and various types of datasets, spanning balanced, imbalanced, binary class, and multi-class datasets.

In this paper, we introduce SML-AutoML, a novel and efficient meta-learning-based Auto-ML framework that concentrates on the entire ML pipeline rather than solely automating the CASH process. The CASH component typically accounts for 20% of the time spent by data scientists in constructing an ML pipeline for a specific problem [33]. Furthermore, our proposed framework leverages meta-learning to easily adapt to new tasks with minimal steps. Without meta-learning, traditional Auto-ML frameworks can identify effective architectures for individual tasks but struggle to do so for new tasks. Additionally, our framework excels in handling imbalanced datasets and enhancing performance by incorporating critical feature engineering and preprocessing steps such as feature selection, data resampling, collinearity assessment, and advanced transformation algorithms. Moreover, our framework speeds up recommending the optimal pipeline for a given dataset. The proposed framework is applicable across a range of real-world applications and sectors, including fraud detection in banking transactions, credit risk assessment in the financial technology industry, fraud detection in auditing processes, ransomware identification in backup systems, diabetes and cancer detection in medical diagnosis, machine failure prediction, cyber threat identification, and customer churn prediction.

The rest of this paper is organized as follows: in Section 2, we present a brief review of related work. In Sections 3, 4, and 5, we propose our solution for recommending a complete ML pipeline for any given dataset. The experimental results are presented in Section 6. Finally, Section 7 concludes and proposes directions for possible future work.

2. RELATED WORK

ML is a very important science that can be adapted to different domains and used to solve complicated problems. In addition, Auto-ML frameworks have gained significant attention in recent years due to their ability to simplify the ML process and enable non-experts to build effective models. In this literature review, we explore three categories of Auto-ML frameworks: meta-learning-based frameworks, non-meta-learning-based frameworks, and cloud-based frameworks. We discuss the key features of each framework within these categories.

2.1 Meta-Learning-Based Auto-ML Frameworks

In recent years, several Auto-ML frameworks have been developed using centralized ML packages. Among these frameworks, some fall into the category of meta-learning-based frameworks:

Auto-Sklearn [34], is the most popular framework among the existing Auto-ML frameworks because it was developed on top of the most popular Python ML library, Scikit-Learn [20]. Auto-Sklearn includes some improvements over Auto-WEKA, such as considering meta-learning [35]. The meta-learning approach depends on the recognition of the characteristics of the datasets. It recommends the appropriate ML pipeline that fits those characteristics. Auto-Sklearn uses Bayesian optimization [36], to determine and store the best ML pipeline with the highest performance for each dataset. According to the study that had been made by [37], Auto-Sklearn outperformed Auto-WEKA system in most of the cases. Although Auto-Sklearn 2.0 [38], includes some improvements over Auto-Sklearn for a faster and more efficient Auto-ML solution, it still lacks the ability to handle imbalanced datasets effectively. Auto-Sklearn 2.0 depended on constructing a portfolio, which can be built offline and consists of ML pipelines to perform well on as many datasets as possible. AlphaD3M [39], is an Auto-ML framework that uses reinforcement learning techniques to optimize the ML pipeline. In AlphaD3M, the process of model discovery or model recommendation is achieved by performing iterative experiments with different ML pipelines. These ML pipelines are generated by inserting, deleting, or replacing different pipeline parts. The optimal pipeline is eventually identified through these trained pipelines, as all actions and decisions are included. AMLBID [40], is an open-source auto-ML framework. AMLBID is a meta-learning-based approach that has been proposed to automate ML models built over industrial data. ATM [41], is an example of a distributed and scalable Auto-ML framework that helps ML users upload their datasets, choose among ML algorithms, and define a search space for hyperparameters. Consequently, ATM recommends the optimal ML pipeline for those datasets by utilizing Bayesian optimization system and meta-learning techniques. Additionally, as R is one of the most popular programming and statistical languages in data science, SmartML has been proposed as the first Auto-ML framework for automating classification problems depending on R packages [42]. The automation process in SmartML is divided into two phases. In the first phase, a knowledge base of meta-features and algorithms performance across different training datasets is constructed. In the second phase, for a given dataset, the meta-features are extracted and compared with the meta-features that are stored in the framework's knowledge base; then the nearest neighbor technique is used to identify the similar datasets in the knowledge base. The retrieved datasets are used to identify the best-performing algorithms on them, hence recommending the dominant algorithm. SmartML depends on SMAC Bayesian Optimization [16], for hyperparameter tuning step.

2.2 Non-Meta-Learning-Based Auto-ML Frameworks

On the other hand, there are also frameworks that do not belong to the meta-learning-based category, such as Auto-WEKA [25, 43], TPOT [26], H2O [27], ML-Plan [28], STREAMLINE [29], OAML (GAMA) [23], AutoKeras [30], LightAutoML [31], and Naive AutoML [32].

Auto-WEKA [25], is an Auto-ML framework that has been implemented on top of the popular data mining tool named WEKA [44]. Auto-WEKA uses the SMAC [45], optimization algorithm and depends on some algorithms of feature selection that have been implemented in WEKA to

solve the CASH problem. In Auto-WEKA 2.0 [43], the new release of Auto-WEKA, updates have been developed to be integrated within the WEKA ecosystem rather than being a standalone piece of software. Besides, including regression problems instead of only supporting classification problems. TPOT [26], was initially developed to deal with biomedical data science, but later it was adapted to handle any ML problem. TPOT is an open-source Auto-ML framework. It automates the ML process by handling feature preprocessing, model selection, and hyperparameter optimization tasks. Similar to Auto-Sklearn, TPOT gained popularity for being built on top of Python's popular ML library, Scikit-Learn. TPOT uses a genetic programming algorithm described in [46], to construct genetic programming trees to combine the algorithms used in ML pipelines. Moreover, TPOT utilizes different algorithms for each task; for example, for feature preprocessing, it utilizes algorithms such as Principal Component Analysis (PCA) and scalers; for feature selection, it uses algorithms such as Recursive Feature Elimination (RFE) [47], and variance thresholds, and for classification problems, it employs algorithms such as K-Nearest Neighbors (KNN), Decision Tree and Random Forest. In general, TPOT is one of the most popular Auto-ML frameworks today. H2O [27], is a distributed Auto-ML framework that automates the process of training many different ML models. The H2O training step is executed on the server and can be accessed by APIs of different programming languages such as R, Python, Java, and Scala. H2O automates feature engineering, data preprocessing, model selection, and hyperparameter optimization. H2O adapts fast random search and stacked ensembles to optimize the recommended pipeline. LightAutoML [31], is an open-source auto-ML framework that was proposed to serve the financial sector. It can automate ML pipeline steps such as feature engineering, data preprocessing, model selection, and hyperparameter optimization. ML-Plan [28], is another Auto-ML framework that depends on hierarchical task networks (HTNs) [48]. ML-Plan automates the ML process by automating algorithm selection and algorithm configuration tasks. STREAMLINE [29], is an Auto-ML framework that focuses on conducting a transparent end-to-end ML pipeline. It was designed to easily conduct ML modeling and analysis for binary classification problems. It focused exclusively on the binary classification supervised learning problem with tabular data. In [49], authors proposed an ML software development pipeline that integrates Auto-ML with MLOps. Through the Auto-ML module, the software automates the process of building and tuning ML models for supervised learning classification problems. On the other hand, in the MLOps module, the software automates the process of deploying and monitoring the machine models. OAML [23], is an Auto-ML method that automates the online learning process. OAML focused on finding the optimal configuration of ML pipelines in the context of batch learning. OAML was developed to solve the online CASH problem in classification problems. OAML relied on the drift concept to monitor the models and update the pipelines when a drift was detected. AutoKeras [30], is an Auto-ML library that automates the process of building and training deep neural networks to solve standard ML problems. AutoKeras provides an end-to-end deep learning solution to users, where the search space can be customized by the user. AutoKeras is built on top of Keras and TensorFlow. In [50], authors proposed an Auto-ML solution to automate the process of building ML models for big data. The solution focused on hyperparameter optimization and the training process. The solution utilized Fabolas [51], and learning curve extrapolation [52], for hyperparameter optimization. Moreover, in the training process, the solution adapted methods that fit the large-scale datasets, such as Bag of Little Bootstraps [53], k-means clustering for SVMs, subsample size selection for gradient descent, and subsampling for logistic regression. Naïve AutoML [32], Introduced as a benchmark to serve as a reference point for comparing Auto-ML frameworks. This approach constructs the ML pipeline sequentially, one step at a time, without considering the potential benefits of jointly optimizing algorithm selection and hyperparameters across the entire pipeline. Its primary objective is to quickly provide a satisfactory

model rather than finding the optimal model. To achieve this, Naïve AutoML incorporates an aggressive early-stopping heuristic. In [49], authors proposed an ML software development pipeline that integrates Auto-ML with MLOps. Through the Auto-ML module, the software automates the process of building and tuning ML models for supervised learning classification problems. On the other hand, in the MLOps module, the software automates the process of deploying and monitoring the machine models.

2.3 Cloud-Based Auto-ML Frameworks

Nevertheless, several cloud-based platforms have considered automating the ML process, considering the high computational power that characterizes cloud environments and hence can facilitate trying different experiments with different ML algorithms and with a wide range of hyperparameters. For example, Google Auto-ML [54], is a cloud-based service provided by Google Cloud to automate the ML process to help ML experts or traditional users with less technical backgrounds. Google Auto-ML provides a wide range of ML algorithms for different tasks, such as traditional ML, natural language processing (NLP), and computer vision. For traditional ML tasks, the Auto-ML Tables service can be used for tabular and structured data by automating ML pipeline tasks such as feature engineering, model selection, and hyperparameter tuning. Moreover, for natural Language processing tasks, Auto-ML Natural Language and Auto-ML Translation services can be used to deal with text in tasks like text analysis, language detection, sentiment analysis, and text similarity. Furthermore, for computer vision tasks, Auto-ML Vision and video intelligence services can be used to extract insights from visual data such as object detection, image classification, and object localization.

Moreover, Azure Auto-ML [55], is another cloud-based service provided by Microsoft to automate both classification and regression tasks. Azure Auto-ML depends on Bayesian optimization and collaborative filtering in searching for the optimal ML pipeline for a given dataset. Azure Auto-ML is based on the search space of Scikit-Learn. Moreover, Amazon SageMaker [56], is a cloud-based service provided by Amazon to automate the ML process. Among the different Auto-ML cloud platforms, Amazon SageMaker has had a wide spread in the last few years. Amazon SageMaker provides ML users with a wide range of ML and deep learning frameworks. Moreover, deploying ML models can be performed on auto-scaling clusters in multiple zones, thereby guaranteeing high availability and high performance during online predictions. Furthermore, Amazon provides a large set of pre-trained models for different tasks such as recommendation systems, image classification, text analysis, and voice recognition.

In addition, Auto-AI [57], is a cloud-based service provided by IBM to automate ML classification and regression tasks. Auto-AI automates the ML process through a set of steps. First, identify the best model (model selection) for the given dataset. Second, apply the feature selection step to keep only the features that support the problem and eliminate the remaining features. Finally, examine a wide range of hyperparameters. Auto-AI accordingly recommends the best-performing ML pipeline based on metrics such as accuracy and precision.

2.4 Domain-Specific AutoML Applications

Generally, Auto-ML has been broadly adapted in various domains, such as healthcare [58–64], blockchain [65], finance [31, 66, 67], transportation [68, 69], and manufacturing [40, 70]. This wide spectrum of applications increases the need for more work and research to enhance the capabilities of existing Auto-ML approaches.

While all of these Auto-ML frameworks provide partial or complete ML pipeline automation, each one works differently and targets different algorithms or dataset structures. Although considering meta-learning approaches and ensembling algorithms in some frameworks, some challenges still exist.

Challenge 1: These frameworks are inefficient when applied to imbalanced datasets.

Challenge 2: Lack of advanced data preprocessing and feature engineering steps

Challenge 3: the high computational power that is needed to perform auto-ML experiments using these frameworks.

In the next section, we will show our proposed Auto-ML framework, which tackles these three challenges.

3. Proposed Auto-ML Framework: SML-AutoML

In this section, we present a new Auto-ML framework called SML-AutoML, designed to address the challenges and constraints encountered in current frameworks. SML-AutoML stands as an Auto-ML framework constructed upon the foundation of the Python library Scikit-Learn. This innovative framework incorporates a repository of data collected from past ML experiments, leveraging this information to construct models illustrating how various ML algorithms are likely to fare across different datasets. When presented with a novel dataset, SML-AutoML utilizes these models to direct experimentation and suggest the most suitable ML pipeline. Each recommendation provides the SML-AutoML framework with additional insights via a feedback mechanism, enabling it to refine its models and enhance their performance metrics (accuracy, recall, and precision).

FIGURE 2 provides an overview of the architecture of the SML-AutoML framework. This framework consists of two main components. The first component is the controller, which interacts with the training datasets. The controller gathers information regarding the various datasets, aiding in the understanding of their unique characteristics. To retrieve this information from the training datasets, the controller utilizes standard APIs like OpenML-Python [71]. The second component is the ML pipeline manager. This manager receives the characteristics of new datasets from the controller and archives them in its repository alongside the features of past datasets. Within the data repository are also stored the outcomes of diverse ML experiments conducted on varied training datasets with distinct characteristics. These experiments showcase the performance of the ML pipeline across different datasets. The ML pipeline manager is complemented by background processes that continuously evaluate new datasets and enhance SML-AutoML's internal ML models.

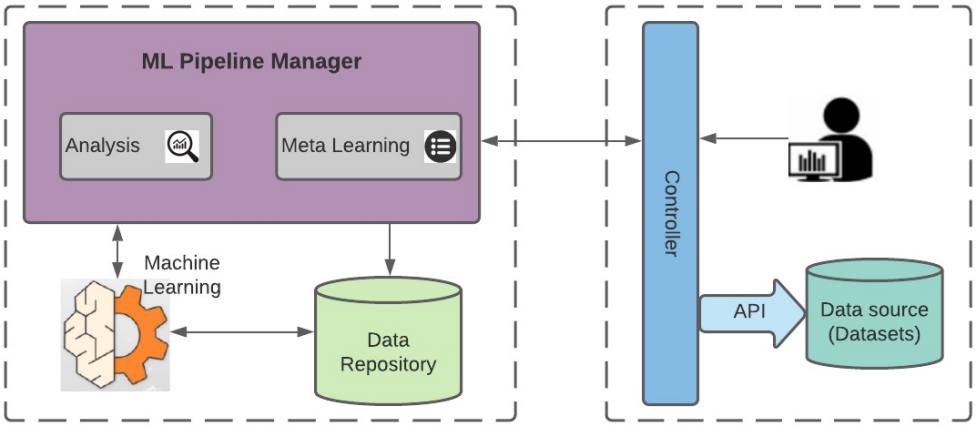

Figure 2: SML-AutoML architecture

These models empower the ML pipeline manager to identify the most analogous datasets and, subsequently, the pertinent ML pipeline for the new datasets.

The development of the SML-AutoML framework progresses through two main phases. The initial phase is the offline training stage, during which the framework undergoes training using different ML algorithms and datasets possessing distinct features. The subsequent phase is the online prediction stage, where an ML pipeline is suggested for a given new dataset utilizing the models established during the offline stage.

Subsequent sections delve into how SML-AutoML acquires meta-features, which are characteristics of datasets. A series of ML experiments is introduced, featuring diverse pipelines to monitor the performance of each pipeline across each dataset. Additionally, the utilization of this knowledge base to recommend the fitting ML pipeline for a specific dataset is elucidated.

3.1 SML-AutoML Offline Stage

In this learning step, SML-AutoML discovers the characteristics of the training datasets and performs experimental ML over those training datasets. FIGURE 3 illustrates the offline SML-AutoML stage. The offline SML-AutoML pipeline stage consists of a set of steps. First, training datasets from the OpenML repository [72], and Kaggle are input to the SML-AutoML framework. For each training dataset, the framework performs two tasks:

- Identifying datasets' characteristics
- Evaluating ML pipelines on the training datasets

In the first task, datasets' characteristics are extracted by calculating some meta-features that describe the datasets and help in identifying their main features. The calculated meta-features are then stored as metadata to be utilized in matching new datasets through the online stage.

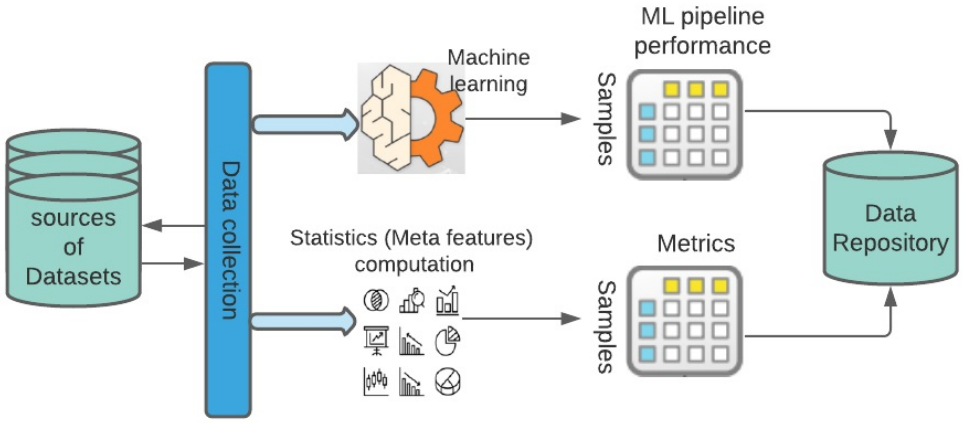

Figure 3: SML-AutoML Offline Stage

In the second task, an ML training process is performed, and multiple pipelines are performed on the training datasets. The training process consists of two main parts: data preprocessing and CASH. In the data preprocessing part, different preprocessing pipelines, which represent the combination of all preprocessing and feature engineering algorithms, are applied to the given training dataset. Pipelines that fit the characteristics of the given dataset are applied, and the remaining pipelines are excluded. For example, if the given dataset has no missing values, all pipelines that include imputation algorithms are excluded. After that, the preprocessed dataset that was obtained from each pipeline is transferred to the CASH part. In the CASH part,a grid search approach is used to try different classifiers with different hyperparameters and select the best classifier and hyperparameters that achieve the highest performance [73]. TABLE 1 shows the different algorithms that have been used in the complete pipeline. The performance of the dominant pipeline is then stored in an evaluation matrix as metadata to be utilized in learning new datasets through the online stage. We outline the process of offline training in Algorithm 1.

The proposed framework considers all tasks of the complete ML pipeline (i.e., data pre-processing, feature engineering, model selection, and hyperparameter optimization). The data pre-processing task considers the most popular techniques for processing datasets. For example, it considers multiple imputation algorithms to fill missing values such as mean, median, mode, and K-nearest neighbors [24]. For encoding data and transforming categorical features to numerical features, algorithms such as One Hot Encoder, Ordinal Encoder, and Log Transformer are considered. Also, for scaling and normalizing the data, scaler algorithms such as Min-Max scaler are considered [74]. In addition, for data balancing, data resampling techniques such as oversampling [75], undersampling [76], and combined sampling [77], are used. Moreover, PCA [78], and polynomial features are employed in the feature engineering part. Finally, for feature selection, RFE-Random Forest, remove collinearity, and variance threshold techniques are examined. Currently, the proposed SML-AutoML framework has been trained using 14 different classifiers (as shown in TABLE 2) to complete the whole Auto-ML pipeline selection lifecycle.

Table 1: The list of algorithms used by SML-AutoML

Data preprocessing	Feature engineering	Feature selection	Classifiers
Ordinal Encoder			GaussianNB
One Hot Encoder			SVM
Log Transformer			KNeighbors
Imputer – Mean	PCA	RFE-RandomForest	Decision Tree
Imputer – Median	Polynomial Features	Remove Collinearity	RandomForest
Imputer – Simple (Mode)		Variance Threshold	ExtraTreesClassifier
Imputer – KNN			GradientBoostingClassifier
Resampling – Over Sampling			Logistic Regression
Resampling – Under Sampling			Bagging
Resampling – combine sampling			LDA
Min Max - Scaler			XGBoost
			SGD
			Ridge
			LGBM

Algorithm 1 Offline training

Input: Datasets D, preprocessing pipelines Pre_{pip}, CASH search space $CASH_{sp}$
Output: datasets characteristics D_{Meta}, evaluation matrix E_{matrix}

for $d_i \subset D$ **do**
  $d_{iMeta} \leftarrow$ extract characteristics of d_i
  Append d_{iMeta} *to* D_{Meta}
  for $p_i \subset Pre_{pip}$ **do**
    if p_i $fits$ d_{iMeta} **then** ▷ //check whether all tasks in p_i can be performed on d_i
      $d_{iTemp} \leftarrow d_i$
      $Model_{best} \leftarrow$ grid search (d_{iTemp} , $CASH_{sp}$)
      Append $Model_{best}$ *to* E_{matrix}
    else
      continue
    end if
  end for
end for

To the best of our knowledge, another limitation of most of the existing Auto-ML frameworks is that they do not handle imbalanced datasets carefully, even though imbalanced datasets are widely spread. In fact, imbalanced datasets exist in various use cases such as retail, churn prediction, fraud detection, and cancer prediction. To tackle this challenge, SML-AutoML added some tasks in the search space such as log transformers, data resampling, and collinearity removal.

After identifying the characteristics of the training dataset and performing ML pipelines on these training datasets, the result of these two steps is stored in a data repository to be utilized for the online prediction and for recommending the appropriate ML pipeline for any new dataset not in the training set.

In addition, the data stored in the data repository acts as a knowledge base for analyzing the performance of each ML pipeline on the different datasets. This in turn helps in extracting the relationship

Table 2: Characteristics of the 30 datasets used for the evaluation

Dataset	Cardinality	Degree	Num of Classes	Minor Class %	Dataset	Cardinality	Degree	Num of Classes	Minor Class %
Credit-Card-Fraud	284807	31	2	0.17	Heart-Attack-Analysis-Prediction	303	13	2	83.6
Telco-Customer-Churn	7043	21	2	26	Hiring-Decisions-in-Recruitment	1500	10	2	45
Anomaly-Detection	134229	8	2	4.8	manufacturing-defect	3240	16	2	19
Covid-19	5644	106	2	9.9	Marketing-Campaign	2240	28	2	17.5
Malware-Detection	5210	70	2	47.7	Minsk2020-ALS	64	134	2	94
House-Sales-Prediction	21613	4	5	0.13	online-course-engagement	9000	8	2	65.7
Oil-Spill	937	50	2	4.37	pet-adoption	2007	12	2	49
Wine-Quality	1599	12	6	0.62	Simple-Loan-Classification	61	7	2	35.5
Occupancy-Detection	8143	7	2	21.2	Thyroid-Disease	383	16	2	39
Abalone	4168	9	21	0.14	Titanic-Dataset	891	11	2	62
Bank-Marketing	41188	20	2	12.7	weather-classification	13200	10	4	100
diabetes-prediction	100000	8	2	8.5	air-quality-health-impact	5811	13	5	1.16
Drink-and-Drive	12282	5	2	45.6	Student-performance	2392	14	5	9
E-Commerce-Shipping	10999	11	2	67.5	Advanced-IoT-Agriculture	30000	14	6	100
Employee-Attrition	59598	23	2	90.6	diagnosed-cbc	1281	14	9	3.27

between the ML pipeline performance and the features of the datasets so that future pipeline recommendations better match the dataset features.

3.2 SML-AutoML Online Stage

After constructing the knowledge base in the training stage, the knowledge base is utilized in prediction and pipeline recommendation for new datasets. In general, SML-AutoML follows a set of steps to recommend an ML pipeline for any given dataset. These steps are shown in FIGURE 4.

The process of recommending an ML pipeline for a given dataset is executed through three steps:

- Dataset characterization, where the SML-AutoML framework identifies the characteristics of the given dataset by computing a set of meta-features.
- Similar datasets identification, where the computed meta-features are compared with the meta-features of all training datasets stored in the data repository, and identify the three nearest datasets.

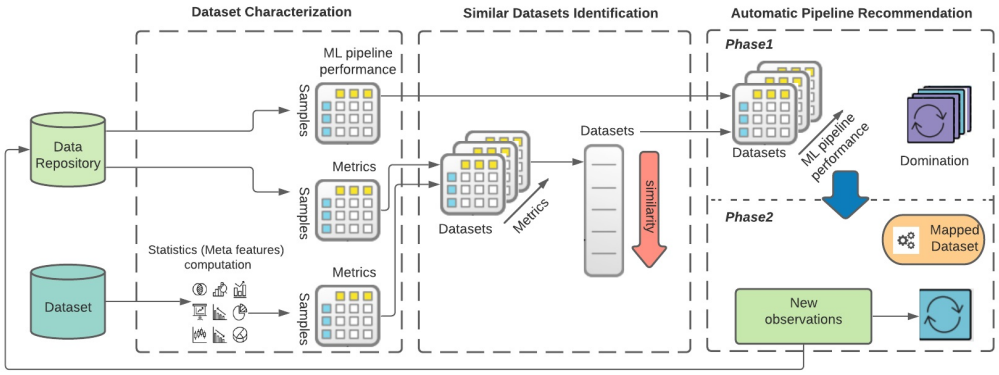

Figure 4: SML-AutoML Online Stage

- Automatic pipeline recommendation, where the extraction of the stored ML pipelines performance that has been stored in the data repository of those three nearest datasets happens, and the recommendation of the dominant ML pipelines occurs. If no similar datasets exist for the given dataset, the characteristics of the given dataset are stored in a data repository to be trained using a customized search space. We outline the process of online pipeline recommendation in algorithm 2.

Algorithm 2 Online prediction

Input: Dataset to be trained d, evaluation matrix E_{matrix}, Training datasets' characteristics D_{Meta}, preprocessing pipelines Pre_{pip}, CASH search space $CASH_{sp}$, Similarity threshold TH
Output: Recommended pipeline $Pip_{recomend}$, E_{matrix}, D_{Meta}

$d_{Meta} \leftarrow$ extract characteristics of d
$d_{iSimilar} \leftarrow$ get the nearest dataset to d (d_{Meta}, D_{Meta})
if $similarity(d, d_{iSimilar}) >= TH$ **then**
    $Pip_{recomend} \leftarrow$ get dominant pipeline ($d_{iSimilar}$, E_{matrix})
else
    Append d_{Meta} *to* D_{Meta}
    for $p_i \subset Pre_{pip}$ **do**
        if p_i $fits$ d_{Meta} **then** ▷ //check whether all tasks in p_i can be applied on d
            $d_{Temp} \leftarrow$ d
            $Model_{best} \leftarrow$ grid search (d_{Temp} , $CASH_{sp}$)
            Append $Model_{best}$ *to* E_{matrix}
        else
            continue
        end if
    end for
end if

4. EXPERIMENTAL EVALUATION

In this section, we present an evaluation of SML-AutoML's ability to automatically recommend the best ML pipeline for a given dataset. We implement all of our code in Python version 3.9.

4.1 Training Data Collection

As discussed in Section 3.1, SML-AutoML requires a corpus of previous training sessions where the framework explores different training datasets that have different characteristics and are trained using different ML algorithms. We evaluate our framework on 300 classification datasets with different characteristics, such as number of instances, number of features, type of features, number of classes, and class imbalance. All datasets are obtained from various sources, including the OpenML repository [72], and Kaggle.

4.2 ML Pipeline Recommendation Evaluation

In this section, we demonstrate how learning from the training sessions improves SML-AutoML's ability to find a good ML pipeline for a given dataset. To accomplish this, we compare SML-AutoML with the most popular Auto-ML pipeline generation platforms. We performed four different experiments where we tracked multiple evaluation measures such as accuracy, precision, recall, and time. First, we compare SML-AutoML with two open-source Auto-ML pipeline generation platforms: TPOT and H2O. Second, we compare SML-AutoML with one open-source Auto-ML pipeline generation platform: Auto-Sklearn. In the third experiment, we compare SML-AutoML with two cloud-based Auto-ML platforms: Microsoft Azure Auto-ML and IBM Auto-AI. And finally, we compare SML-AutoML with one open-source Auto-ML pipeline generation platform: LightAutoML [31].

4.2.1 Datasets

We used 30 public datasets[1] with different characteristics to compare our proposed framework to the existing frameworks. TABLE 2 presents details about the datasets used in the evaluation.

4.2.2 Solution environment and hardware

Our proposed framework was implemented in Python 3.9 on a Windows 10 Core i7 CPU machine with 16GB of RAM to compare it with Naive AutoML, TPOT, H2O, LightAutoML, OAML, and Streamline. TABLE 3 shows the time in minutes it takes for Naive AutoML, TPOT, H2O, LightAutoML, OAML, Streamline, and SML-AutoML to recommend an ML pipeline for each of the evaluation datasets.

[1] https://github.com/IbrahemGomaa/ML-Datasets/blob/main/Datasets.txt

Table 3: Performance Evaluation (time) SML-AutoML vs Naive AutoML, TPOT, H2O, LightAutoML, OAML, and Streamline

Datasets	Auto-ML Frameworks						
	Naive AutoML	TPOT	H2O	Light AutoML	OAML	Stream line	SML-AutoML
Credit-Card-Fraud	4.1	700	1.8	13.7	2.85	2916	0.67
Telco-Customer-Churn	10.5	29.1	1.4	2.25	2.7	628	0.62
Anomaly-Detection	12.7	295.2	1.7	19.9	2.78	676	4.3
Covid-19	5.6	35.7	1.5	10.5	2.7	396	1.1
Malware-Detection	6.3	31.9	1.72	7.7	2.7	421	0.48
House-Sales-Prediction	35.7	155	1.33	11.3	2.7		1.1
Oil-Spill	4.6	5.12	1.06	0.97	2.7	242	0.16
Wine-Quality	0.22	20.7	1.63	8	2.7		0.15
Occupancy-Detection	27.3	12.4	1.7	3.5	2.7	299	0.5
Abalone	11.5	104.2	1.73	13.4	2.7		1.83
Bank-Marketing	8.2	167.5	2.3	12.8	2.8	472	1.9
diabetes-prediction	9.8	204.6	1.37	6.8	2.7	566	2.5
Drink-and-Drive	0.1	14.9	1.5	8.6	2.7	458	0.9
E-Commerce-Shipping	0.5	16.3	1.6	12.8	2.7	495	1.1
Employee-Attrition	19.4	275	1.8	16.3	2.8	586	3.2
Heart-Attack-Analysis-Prediction	0.15	4.8	1.3	2.1	2.7	132	0.4
Hiring-Decisions-in-Recruitment	6.5	14.3	1.4	8.4	2.7	169	1.2
manufacturing-defect	3.13	17.2	1.5	11.4	2.7	177	1.3
Marketing-Campaign	20.7	16.8	1.7	10.9	2.7	212	2.3
Minsk2020-ALS	3.35	5.4	1.4	7.5	2.7	98	0.4
online-course-engagement	15.1	13.9	1.7	14.2	2.7	507	2.4
pet-adoption	4.2	8.6	1.4	7.6	2.7	133	0.8
Simple-Loan-Classification	2.1	4.2	1.3	5.8	2.7	82.1	0.3
Thyroid-Disease	3.8	12.2	1.6	11.3	2.7	162	1.2
Titanic-Dataset	10.7	56	1.8	15.2	2.7	418	2.2
weather-classification	14.6	112.3	1.7	9.4	2.7		2.6
air-quality-health-impact	18	101.7	1.5	8.2	2.7		1.3
Student-performance	8.7	49.5	1.7	6.4	2.7		1.1
Advanced-IoT-Agriculture	2.9	142	1.8	11.3	2.7		1.8
diagnosed-cbc	7.9	35.8	1.5	6.3	2.7		0.5

Moreover, the proposed framework was implemented in Python 3.9 using a virtual machine with Ubuntu 20.04, 12 GB of RAM, and 4 processors to compare it with Auto-Sklearn. TABLE 4 shows the time in minutes it takes for Auto-Sklearn and SML-AutoML to recommend an ML pipeline for each of the evaluation datasets.

Table 4: Performance Evaluation (time) SML-AutoML vs Auto-Sklearn

	Auto-ML Frameworks	
Datasets	Auto-Sklearn	SML-AutoML
Credit-Card-Fraud	60.1	0.66
Telco-Customer-Churn	59.9	0.55
Anomaly-Detection	60	4.1
Covid-19	59.8	1
Malware-Detection	60.1	0.45
House-Sales-Prediction	59.9	1.1
Oil-Spill	59.8	0.15
Wine-Quality	60	0.15
Occupancy-Detection	59.9	0.38
Abalone	59.8	1.8
Bank-Marketing	59.8	1.9
diabetes-prediction	59.9	2.4
Drink-and-Drive	60.1	0.8
E-Commerce-Shipping	60	1.1
Employee-Attrition	60.1	3.1
Heart-Attack-Analysis-Prediction	60.1	0.4
Hiring-Decisions-in-Recruitment	60	1.2
manufacturing-defect	59.8	1.2
Marketing-Campaign	60	2.3
Minsk2020-ALS	59.8	0.4
online-course-engagement	60	2.3
pet-adoption	60.1	0.8
Simple-Loan-Classification	60.1	0.4
Thyroid-Disease	60	1.1
Titanic-Dataset	59.8	2.3
weather-classification	59.8	2.5
air-quality-health-impact	59.8	1.4
Student-performance	60	1.3
Advanced-IoT-Agriculture	60.1	2.1
diagnosed-cbc	60	0.4

On the other hand, we compared our framework with Azure Auto-ML and IBM Auto-AI. We used a compute cluster with 4 vCPUs (cores), 14 GB of RAM, and 28 GB of storage to perform Auto-ML experiments for the 30 datasets on Microsoft Azure. On the other hand, we used an environment with 8 vCPUs and 32 GB of RAM to perform Auto-ML experiments for the 30 datasets on IBM Watson ML. TABLE 5 shows the time in minutes it takes for Azure Auto-ML, IBM Auto-AI, and SML-AutoML to recommend an ML pipeline for each of the evaluation datasets.

Finally, TABLE 6 and TABLE 7, summarize the performance of the proposed framework against the existing frameworks for the binary and multi-class classification datasets.

Table 5: Performance Evaluation (time) SML-AutoML vs Azure Auto-ML and IBM Auto-AI

	Auto-ML Frameworks		
Datasets	Azure Auto-ML	IBM Auto-AI	SML-AutoML
Credit-Card-Fraud	84	127	0.67
Telco-Customer-Churn	52.6	5	0.62
Anomaly-Detection	112.3	5.3	4.3
Covid-19	50.8	6.1	1.1
Malware-Detection	123.4	7.2	0.48
House-Sales-Prediction	58.6	9.4	1.1
Oil-Spill	63.7	6.3	0.16
Wine-Quality	62.6	11.7	0.15
Occupancy-Detection	59.46	6.6	0.5
Abalone	120.8	8.9	1.83
Bank-Marketing	78	12.9	1.9
diabetes-prediction	86.5	15.7	2.5
Drink-and-Drive	50.3	12.3	0.9
E-Commerce-Shipping	51.9	11.6	1.1
Employee-Attrition	102.6	38.2	3.2
Heart-Attack-Analysis-Prediction	48.6	6.4	0.4
Hiring-Decisions-in-Recruitment	53	11.8	1.2
manufacturing-defect	58.2	17.3	1.3
Marketing-Campaign	72.6	24	2.3
Minsk2020-ALS	48.3	9.7	0.4
online-course-engagement	66.3	22.3	2.4
pet-adoption	51.3	14.3	0.8
Simple-Loan-Classification	47.6	4.5	0.3
Thyroid-Disease	56.2	7.8	1.2
Titanic-Dataset	66.4	15.4	2.2
weather-classification	71.4	19.4	2.6
air-quality-health-impact	59	14.5	1.3
Student-performance	54.8	12.3	1.1
Advanced-IoT-Agriculture	61.8	11.2	1.8
diagnosed-cbc	48.3	6.3	0.5

4.3 Evaluation Results and Analysis

In this section, we will analyze the experimental results that have been performed to compare our proposed framework, SML-AutoML, with the existing frameworks. In the results of the experiments, presented in TABLE 6, and TABLE 7, it is shown that the performance of the proposed framework, SML-AutoML, and Naive AutoML, TPOT, H2O, OAML, Streamline, and Auto-Sklearn frameworks is almost similar in terms of accuracy, precision, and recall measures of binary class datasets, with a slight superiority of SML-AutoML in those measures. However, this superiority appears greatly in the same measures (accuracy, precision, and recall) for multi-class datasets. Additionally, SML-AutoML dominates TPOT, H2O, LightAutoML, OAML, Streamline, and Auto-

Table 6: AutoML frameworks' performance comparison for binary classification

		Open-source solutions							Cloud-based solutions		Proposed
Datasets	Measure	Naive AutoML	TPOT	H2O	Light AutoML	OAML	Stream line	Auto-sklearn	Azure Auto-ML	IBM Auto-AI	SML-AutoML
Credit-Card-Fraud	Accuracy	99.9%	99.9%	99.9%	99.8%	99.8%	99.9%	99.9%	99.9%	99.9%	99.9%
	Precision	99.9%	99.9%	83.7%		99.8%	94.9%	99.9%	99.9%	93.6%	99.9%
	Recall	99.9%	99.9%	79.5%		99.8%	90.4%	99.9%	99.9%	79.3%	99.9%
Telco-Customer-Churn	Accuracy	79.8%	79.6%	78.2%	80.6%	78.8%	80.5%	79.5%	80.9%	80.1%	85.3%
	Precision	78.7%	83.3%	60.9%		77.5%	67.2%	78.3%	79.9%	73.6%	85.3%
	Recall	79.8%	98.9%	80.4%		78.8%	81.5%	79.5%	80.9%	53.2%	98.7%
Anomaly-Detection	Accuracy	96.2%	97.3%	96.3%	97%	96.8%	96.9%	97.1%	97%	95.3%	98%
	Precision	95.7%	96.4%	62.4%		96.4%	78.5%	96.9%	96.7%	79%	98%
	Recall	96.2%	96.5%	64.6%		96.8%	88.4%	97.1%	97%	21.4%	98%
Covid-19	Accuracy	90.9%	91.2%	83.8%	10%	90.3%	90%	90.8%	90.7%	90.3%	93.8%
	Precision	88.2%	100%	24.1%		91.2%	50.8%	89.4%	89.3%	72.2%	100%
	Recall	90.9%	97.7%	78.6%		90.3%	98.7%	90.8%	90.7%	27%	97.4%
Malware-Detection	Accuracy	99.3%	99.2%	99.4%	98.9%	98.6%	99.1%	99.3%	99.1%	96.5%	99.3%
	Precision	99.3%	99.2%	99.3%		98.6%	99%	99.3%	99.1%	95%	99.3%
	Recall	99.3%	99.8%	99.5%		98.6%	99.5%	99.3%	99.1%	95.6%	99.3%
Oil-Spill	Accuracy	96%	97.4%	96%	96.2%	96.5%	96.3%	96.1%	97.3%	97.6%	98.7%
	Precision	95.7%	33.3%	53%		96%	68.3%	95.1%	97.2%	81.5%	98.7%
	Recall	96%	77.7%	83.3%		96.5%	95%	96.1%	97.3%	64.5%	98.7%
Occupancy-Detection	Accuracy	99.4%	99.6%	99.4%	99.2%	99.7%	99.3%	99.6%	99.4%	99.3%	99.6%
	Precision	99.4%	99.3%	98.7%		99.7%	98%	99.6%	99.4%	98.2%	99.6%
	Recall	99.4%	100%	99.5%		99.7%	99.8%	99.6%	99.4%	98.8%	99.6%
Bank-Marketing	Accuracy	91.6%	91%	90.2%	90.6%	91.4%	91%	90.9%	92.1%	86.4%	93.5%
	Precision	91.1%	91.8%	88.4%		92.5%	91%	92%	92.8%	84.1%	93.6%
	Recall	91.6%	91%	90.2%		91.4%	91%	90.9%	92.1%	86.4%	93.5%
diabetes-prediction	Accuracy	97%	97.4%	81.5%	96.8%	96.4%	96.3%	97.8%	97.2%	97%	98.1%
	Precision	97%	98.7%	69.2%		96.4%	96.7%	98.1%	97.5%	79%	98.1%
	Recall	97%	97.4%	81.5%		96.4%	96.3%	97.8%	97.2%	71.6%	98.1%
Drink-and-Drive	Accuracy	67.9%	67.8%	61.8%	65.8%	68%	62%	68%	68.1%	63%	82.5%
	Precision	64.4%	78.5%	45%		67%	46.6%	64.2%	65.2%	27.5%	83%
	Recall	67.9%	84.8%	61.8%		68%	62%	68%	68.1%	39.5%	91%
E-Commerce-Shipping	Accuracy	68.5%	70.1%	64.6%	67.8%	69.4%	67.9%	68.3%	68.7%	59.7%	87.7%
	Precision	80%	83.8%	58.6%		80.5%	79.2%	80.1%	80.6%	45%	87.7%
	Recall	68.5%	70.1%	51.3%		69.4%	67.9%	68.3%	68.7%	34.7%	94.5%
Employee-Attrition	Accuracy	75.6%	75.2%	75.7%	79.2%	76.3%	77.2%	74.8%	77.8%	79%	89.4%
	Precision	75.6%	81%	60.5%		76.3%	69.8%	74.8%	77.8%	72.3%	89%
	Recall	75.6%	95.2%	73.4%		76.3%	77.2%	74.8%	77.8%	52.2%	92.4%

Table 6: Continued..

		Open-source solutions							Cloud-based solutions		Proposed
Datasets	Measure	Naive AutoML	TPOT	H2O	Light AutoML	OAML	Stream line	Auto-sklearn	Azure Auto-ML	IBM Auto-AI	SML-AutoML
Heart-Attack-Analysis-Prediction	Accuracy	78%	78%	79.5%	77.6%	78.6%	77.3%	79.2%	79%	77.8%	83.2%
	Precision	78%	81%	61%		78.6%	73%	79.2%	79%	68.8%	83.2%
	Recall	78%	94.4%	56.3%		78.6%	68.5%	79.2%	79%	62.4%	92.7%
Hiring-Decisions-in-Recruitment	Accuracy	92%	94.2%	91.8%	91.2%	93.2%	91.8%	94.3%	94.3%	92.1%	94.6%
	Precision	92%	93.2%	91.8%		93%	84.5%	94%	94.3%	90.5%	94.6%
	Recall	92%	94.2%	91.8%		93.2%	86.4%	94.3%	94.3%	92.1%	94.6%
manufacturing-defect	Accuracy	96%	97.3%	95.6%	96%	96.5%	95.8%	95.7%	96.2%	95.7%	97.4%
	Precision	96%	97.3%	93.8%		96.5%	91%	95.7%	96.2%	95.7%	98.6%
	Recall	96%	97.8%	94%		96.5%	90.4%	95.7%	96.2%	95.7%	97.4%
Marketing-Campaign	Accuracy	85.4%	86.2%	85.5%	83.2%	84%	84%	85.2%	85.6%	84.1%	89%
	Precision	82.9%	83.5%	79.3%		81.4%	75.5%	83.1%	83.4%	72%	85.4%
	Recall	85.4%	91%	71.8%		84%	79.7%	85.2%	85.6%	68.8%	89%
Minsk2020-ALS	Accuracy	75%	74.5%	72.7%	23.4%	75%	74.3%	75.1%	75.6%	73.8%	80.2%
	Precision	76%	74.5%	66%		75.5%	66.2%	75.8%	77.2%	72%	75.8%
	Recall	75%	74.5%	64.5%		75%	62.3%	75.1%	75.6%	73.8%	80.2%
online-course-engagement	Accuracy	96.7%	97.4%	96%	97.1%	96.8%	96.3%	97.4%	97%	96%	97.3%
	Precision	96.7%	96.8%	72%		96.5%	88%	96.8%	96.7%	92.2%	97.3%
	Recall	96.7%	96.8%	73%		96.8%	87.5%	96.8%	97%	92.6%	97.3%
pet-adoption	Accuracy	94.5%	95.2%	95%	95.7%	95.3%	94.8%	95%	94.8%	93.7%	96.3%
	Precision	94.5%	95.4%	62.4%		94%	89%	94.6%	94.8%	94%	96.3%
	Recall	94.5%	95.4%	61.3%		94%	87.4%	95%	94.8%	93.7%	96.3%
Simple-Loan-Classification	Accuracy	94.7%	95.4%	93.8%	95.5%	94.2%	93.8%	94.7%	95.2%	93.7%	95.3%
	Precision	95.3%	95.4%	71%		94.2%	89.5%	94.9%	95.4%	90.5%	97.2%
	Recall	94.7%	95.4%	66.7%		94%	86.4%	94.7%	95.2%	89.4%	95.3%
Thyroid-Disease	Accuracy	95.6%	97.2%	94.8%	94.3%	95.5%	95.2%	96%	97.5%	95.6%	97.4%
	Precision	95.8%	97.5%	87.4%		95.5%	92.7%	96.6%	98.4%	91.5%	98.6%
	Recall	95.6%	97.2%	88.6%		95.5%	89.5%	96%	97.5%	95.6%	97.4%
Titanic-Dataset	Accuracy	78.7%	85.4%	78.2%	78.4%	77.6%	78.4%	79.2%	86.4%	77.9%	88.3%
	Precision	78.7%	85.4%	67.3%		77.6%	74.3%	79.5%	86.4%	64.2%	88.3%
	Recall	78.7%	85.4%	53.8%		77.6%	72.6%	79.2%	86.4%	58.4%	88.3%

Sklearn frameworks over time, as shown in TABLE 3, and Table 4. SML-AutoML can recommend the appropriate pipeline faster for most datasets. Moreover, the TPOT framework dominates the H2O framework in accuracy, precision, and recall for all datasets. On the other hand, H2O and OAML take less time in all experiments than TPOT because of their dependence on a sample of the data instead of the total population or the whole dataset.

Table 7: AutoML frameworks' performance comparison for multi-class classification

		Open-source solutions						Cloud-based solutions		Proposed
Datasets	Measure	Naive AutoML	TPOT	H2O	Light AutoML	OAML	Auto-sklearn	Azure Auto-ML	IBM Auto-AI	SML-AutoML
House-Sales-Prediction	Accuracy	64.8%	64.7%	65.2%	0.01%	64.9%	64.8%	66.5%	33%	72.6%
	Precision	59.3%	71.3%	68.8%		42%	55.9%	65.8%	43.2%	72.6%
	Recall	64.8%	64.7%	65.2%		64.9%	64.8%	66.5%	33%	72.9%
Wine-Quality	Accuracy	58.7%	68.5%	69.8%	0.0%	71%	64.2%	68.9%	59.5%	88.2%
	Precision	55.7%	62.7%	66.2%		67.6%	60.8%	66%	55.9%	88.1%
	Recall	58.7%	68.5%	69.8%		71%	64.2%	68.9%	59.5%	88.2%
Abalone	Accuracy	25.9%	26.5%	28.3%	0.0%	26.6%	24.5%	27.8%	28.1%	80%
	Precision	24.6%	29.6%	29.2%		23%	22.1%	25.7%	26.2%	80%
	Recall	25.9%	27.7%	28.3%		26.6%	24.5%	27.8%	28.1%	79.1%
weather-classification	Accuracy	91.3%	92.4%	92.8%	0.0%	92.6%	90.7%	90.5%	90.2%	97.4%
	Precision	91.4%	92.7%	92.8%		92.8%	91.5%	90.6%	87.6%	98.2%
	Recall	91.3%	92.4%	92.8%		92.6%	90.7%	90.5%	85.2%	97.4%
air-quality-health-impact	Accuracy	95%	92.3%	94.5%	0.0%	95.2%	95.4%	94.9%	90.2%	98.6%
	Precision	94.6%	92%	89.2%		94.6%	94.5%	94.3%	88.5%	99.1%
	Recall	95%	92.3%	91.3%		95.2%	95.4%	94.9%	89.2%	98.6%
Student-performance	Accuracy	92.3%	93.3%	92.2%	0.8%	93.4%	92.2%	92%	91.5%	97.3%
	Precision	92.1%	93.2%	91.4%		93.2%	91.6%	91.3%	79.5%	95.6%
	Recall	92.3%	93.3%	92.2%		93.4%	92.2%	92%	86.5%	97.3%
Advanced-IoT-Agriculture	Accuracy	100%	99.9%	99.9%	0.3%	99.9%	99.9%	99.9%	94.8%	99.9%
	Precision	100%	99.9%	99.4%		99.9%	99.9%	99.9%	86.7%	99.9%
	Recall	100%	99.9%	99.5%		99.9%	99.9%	99.9%	94.8%	99.9%
diagnosed-cbc	Accuracy	98.1%	98%	97.6%	0.0%	98.5%	97.8%	98.7%	88.5%	99.8%
	Precision	98.1%	98%	89.2%		98.5%	97.8%	98.7%	87.7%	99.8%
	Recall	98.1%	98%	87%		98.5%	97.8%	98.7%	85.7%	99.8%

Moreover, SML-AutoML and Azure Auto-ML outperform IBM Auto-AI in all measures of most of the datasets. It is also shown that the performance is almost similar between the SML-AutoML framework and Azure Auto-ML, with a slight superiority for SML-AutoML. However, this superiority appears greatly in performance and time in the case of multi-class datasets, as shown in TABLE 6, and Table 7.

Our proposed framework demonstrates a notable superiority in terms of cost-efficiency, as it requires less time to recommend an appropriate pipeline for a given dataset. This advantage stems from its implementation of a meta-learning approach and its utilization of diverse datasets during the training phase.

On the other hand, SML-AutoML outperforms existing frameworks in terms of accuracy, precision, and recall, particularly for multi-class datasets. The exceptional performance of SML-AutoML can be attributed to the comprehensive range of tasks integrated into its pipeline. These tasks encompass various preprocessing steps, such as data resampling, ensemble algorithm utilization, and collinearity detection, as well as an extensive array of ML algorithms and their corresponding hyperparameters. By adopting this comprehensive approach, SML-AutoML effectively optimizes the end-to-end process for a given dataset, resulting in enhanced overall performance compared to other frameworks.

However, it is important to acknowledge that some existing Auto-ML frameworks achieve comparable results for imbalanced binary datasets by incorporating ensemble algorithms. Nevertheless, it should be emphasized that relying exclusively on ensemble algorithms is insufficient to consistently produce well-performing models across all instances of imbalanced datasets.

5. CONCLUSION AND FUTURE WORK

In this paper, we propose a new Auto-ML framework named SML-AutoML for the automatic recommendation of ML pipelines. The significant advantage of our proposed framework is its ability to recommend the appropriate pipeline for a given dataset in a short time. Moreover, its ability to deal with imbalanced binary and multiclass datasets. Those advantages are due to integrating multiple components, such as utilizing meta-learning, including data resampling techniques, using additional transformation techniques that fit the nature of imbalanced datasets, and making use of both ensemble and basic algorithms. Thus, adapting those components enables our framework to outperform state-of-the-art results in terms of accuracy, precision, recall, and time. Experiments and analysis show that our proposed framework achieves, on average, more than 5% outperformance compared to the existing auto-ML frameworks. For future work, we intend to extend our proposed framework to include unsupervised learning and time series forecasting problems.

References

[1] Graves A, Mohamed A, Hinton G. Speech Recognition With Deep Recurrent Neural Networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing. 2013:6645-6649.

[2] Abdulkareem NM, Abdulazeez AM, Zeebaree DQ, Hasan DA. COVID19 World Vaccination Progress Using Machine Learning Classification Algorithms. Qubahan Acad J. 2021;1:100-105.

[3] Sujitha R, Seenivasagam V. Retracted Article: Classification of Lung Cancer Stages With Machine Learning Over Big Data Healthcare Framework. J Ambient Intell Human Comput. 2021;12:5639-5649.

[4] Jain H, Yadav G, Manoov R. Churn Prediction and Retention in Banking, Telecom and It Sectors Using Machine Learning Techniques. In: Patnaik S, Yang XS, Sethi IK, editors. Advances in machine learning and computational intelligence. Springer Singapore. 2021:137-156.

[5] Iatrellis O, Savvas IK, Fitsilis P, Gerogiannis VC. A Two-Phase Machine Learning Approach for Predicting Student Outcomes. Educ Inf Technol. 2021;26:69-88.

[6] Li C, Xu P. Application on Traffic Flow Prediction of Machine Learning in Intelligent Transportation. Neural Comput Appl. 2021;33:613-624.

[7] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In the Proceedings of the IEEE conference on computer vision and pattern recognition. 2016:770-778.

[8] Sheykhmousa M, Mahdianpari M, Ghanbari H, Mohammadimanesh F, Ghamisi P, et. al. Support Vector Machine Versus Random Forest for Remote Sensing Image Classification: A Meta-Analysis and Systematic Review. IEEE J Sel Top Appl Earth Obs Remote Sens. 2020;13:6308-6325.

[9] Chandra MA, Bedi SS. Survey on SVM and their application in image classification. Int J Inf Technol. 2021;13:1-11.

[10] Minaee S, Kalchbrenner N, Cambria E, Nikzad N, Chenaghlu M, et al. DEep Learning–Based Text Classification: A Comprehensive Review. ACM Comput. Surv. CSUR. 2021;54:1-40.

[11] Fararni KA, Nafis F, Aghoutane B, Yahyaouy A, Riffi J, et. al. Hybrid Recommender System for Tourism Based on Big Data and AI: A Conceptual Framework. Big Data Min Anal. 2021;4:47-55.

[12] Covington P, Adams J, Sargin E. Deep Neural Networks for YouTube Recommendations. In: Proceedings of the 10th ACM conference on recommender systems New York USA. New York USA: ACM. 2016:191-198.

[13] Choudhury SS, Mohanty SN, Jagadev AK. Multimodal Trust Based Recommender System With Machine Learning Approaches for Movie Recommendation. Int J Inf Technol. 2021;13:475-482.

[14] Putri DCG, Leu JS, Seda P. Design of an Unsupervised Machine Learning-Based Movie Recommender System. Symmetry. 2020;12:185.

[15] Zomaya AY, Sakr S. Handbook of Big Data Technologies. Springer International Publishing. 2017.

[16] Sakr S, Zomaya AY, editors. Encyclopedia of Big Data Technologies. Springer International Publishing. 2019.

[17] Elshawi R, Maher M, Sakr S. Automated Machine Learning: State-Of-The-Art and Open Challenges. 2019. ArXiv preprint: https://arxiv.org/pdf/1906.02287

[18] Vafeiadis T, Diamantaras KI, Sarigiannidis G, Chatzisavvas KCh. A Comparison of Machine Learning Techniques for Customer Churn Prediction. Simul Modell Pract Theor. 2015;55:1-9.

[19] Probst P, Boulesteix AL. To Tune or Not to Tune the Number of Trees in Random Forest. J Mach Learn Res. 2017;18:1-8.

[20] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, et. al. Scikit-Learn: Machine Learning in Python. J Mach Learn Res. 2011;12:2825-2830.

[21] Hutter F, Kotthoff L, Vanschoren J. Automated Machine Learning: Methods Systems Challenges. Springer Nature. 2019.

[22] Yang L, Shami A. On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice. Neurocomputing. 2020;415:295-316.

[23] Celik B, Singh P, Vanschoren J. Online Automl: An Adaptive Automl Framework for Online Learning. Mach Learn. 2023;112:1897-1921.

[24] Zhang C, Zhu X, Zhang J, Qin Y, Zhang S. GBKII: An Imputation Method for Missing Values. In Pacific-Asia Conference on Knowledge Discovery and Data Mining. Berlin, Heidelberg. Springer. 2007:1080-1087.

[25] Thornton C, Hutter F, Hoos HH, Leyton-Brown K. Auto-Weka: Combined Selection and Hyperparameter Optimization of Classification Algorithms. In: proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York USA: ACM. 2013:847-855.

[26] Olson RS, Urbanowicz RJ, Andrews PC, Lavender NA, Kidd LC, et. al. Automating Biomedical Data Science Through Tree-Based Pipeline Optimization Applications of Evolutionary Computation. Springer. 2016:123-137.

[27] LeDell E, Poirier S. H2O Automl: Scalable Automatic Machine Learning. In: Proceedings of the AutoML workshop at ICML. San Diego, CA, USA. ICML. 2020;2020:16.

[28] Mohr F, Wever M, Hüllermeier E. ML-Plan: Automated Machine Learning via Hierarchical Planning. Mach Learn. 2018;107:1495-1515.

[29] Urbanowicz R, Zhang R, Cui Y, Suri P. Streamline: A Simple, Transparent, End-To-End Automated Machine Learning Pipeline Facilitating Data Analysis and Algorithm Comparison. Genetic Programming Theory and Practice XIX. Singapore: Springer Nature Singapore. 2023:201-231.

[30] Jin H, Chollet F, Song Q, Hu X. Autokeras: An Automl Library for Deep Learning. J Mach Learn Res. 2023;24:1-6.

[31] Vakhrushev A, Ryzhkov A, Savchenko M, Simakov D, Damdinov R, et. al. Lightautoml: Automl Solution for a Large Financial Services Ecosystem. 2021. ArXiv preprint: https://arxiv.org/pdf/2109.01528

[32] Mohr F, Wever M. Naive Automated Machine Learning. Mach Learn. 2023;112:1131-1170.

[33] Zöller MA, Huber MF. Benchmark and Survey of Automated Machine Learning Frameworks. J Artif Intell Res. 2021;70:409-472.

[34] Feurer M, Klein A, Eggensperger K, Springenberg J, Blum M, et. al. Efficient and Robust Automated Machine Learning. Adv Neural Inf Process Syst. 2015;28.

[35] Vanschoren J. Meta-Learning. In: Hutter F, Kotthoff L, Vanschoren J. (eds). Automated Machine Learning. The Springer Series on Challenges in Machine Learning. Springer. 2019:35-61.

[36] Feurer M, Springenberg J, Hutter F. Initializing Bayesian Hyperparameter Optimization via Meta-Learning. In Proceedings of the AAAI Conference on Artificial Intelligence. 2015;29.

[37] Guyon I, Bennett K, Cawley G, Escalante HJ, Escalera S, et. al. Design of the Chalearn Automl Challenge. 2015 International Joint Conference on Neural Networks (IJCNN). Ireland. 2015:1-8.

[38] Feurer M, Eggensperger K, Falkner S, Lindauer M, Hutter F. Auto-Sklearn 2.0: Hands-Free Automl via Meta-Learning. J Mach Learn Res. 2022;23:1-61.

[39] Drori I, Krishnamurthy Y, Rampin R, Lourenco RD, Ono JP, et al. AlphaD3M: Machine Learning Pipeline Synthesis. 2021.Arxiv preprint: https://arxiv.org/pdf/2111.02508

[40] Garouani M, Ahmad A, Bouneffa M, Hamlich M, Bourguin G, et. al. Using Meta-Learning for Automated Algorithms Selection and Configuration: An Experimental Framework for Industrial Big Data. J Big Data. 2022;9:57.

[41] Swearingen T, Drevo W, Cyphers B, Cuesta-Infante A, Ross A, et. al. ATM: A Distributed, Collaborative, Scalable System for Automated Machine Learning. Proceedings of the IEEE international conference. Big Data. 2017:151-162.

[42] Maher MM, Sakr S. Smartml: A Meta Learning-Based Framework for Automated Selection and Hyperparameter Tuning for Machine Learning Algorithms. In: EDBT: 22nd International Conference on Extending Database Technology. 2019.

[43] Kotthoff L, Thornton C, Hoos HH, Hutter F, Leyton-Brown K. Auto-Weka: Automatic Model Selection and Hyperparameter Optimization in Weka. Cham: Springer International Publishing. 2019:81-95.

[44] Holmes G, Donkin A, Witten IH. Weka: A Machine Learning Workbench. In: Proceedings of the ANZIIS'94 – Australian new Zealnd intelligent information systems Conference. New York: IEEE. 1994:357-361.

[45] Hutter F, Hoos HH, Leyton-Brown K. Sequential Model-Based Optimization for General Algorithm Configuration. In: Coello CA, editor. Learning and intelligent optimization. Springer Berlin Heidelberg. 2011:507-523.

[46] Banzhaf W, Nordin P, Keller RE, Francone FD. Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications. Morgan Kaufmann Publishers Incorporated. 1998.

[47] Guyon I, Weston J, Barnhill S, Vapnik V. Gene Selection for Cancer Classification Using Support Vector Machines. Mach Learn. 2002;46:389-422.

[48] Ghallab M, Nau D, Traverso P. Automated Planning: Theory and Practice. Elsevier. 2004.

[49] Filippou K, Aifantis G, Papakostas GA, Tsekouras GE. Structure Learning and Hyperparameter Optimization Using an Automated Machine Learning (Automl) Pipeline. Information. 2023;14:232.

[50] Karras A, Karras C, Schizas N, Avlonitis M, Sioutas S. Automl With Bayesian Optimizations for Big Data Management. Information. 2023;14:223.

[51] Klein A, Falkner S, Bartels S, Hennig P, Hutter F. Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. In: Proceedings of the 20th international conference on artificial intelligence and statistics. Ft. Lauderdale FL USA. 2017;54:528-536.

[52] Domhan T, Springenberg JT, Hutter F. Speeding Up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves. In: Proceedings of the 24th international conference on artificial intelligence Buenos Aires Argentina. 2015:3460-3468.

[53] Kleiner A, Talwalkar A, Sarkar P, Jordan MI. A Scalable Bootstrap for Massive Data. J R Stat Soc B. 2014;76:795-816.

[54] https://cloud.google.com/vertex-ai

[55] https://learn.microsoft.com/enus/azure/machine-learning/tutorial-first-experiment-automated-ml

[56] https://docs.aws.amazon.com/sagemaker/latest/dg/gs.html

[57] IBM.: https://www.ibm.com/cloud/watsonstudio/autoai

[58] Mustafa A, Rahimi Azghadi M. Automated Machine Learning for Healthcare and Clinical Notes Analysis. Computers. 2021;10:24.

[59] Lagerev D, Korsakov A, Zakharova A. Exploratory Analysis of Biomedical Data in Order to Construct Intelligent Analytical Models for Assessing the Risk of Cancer. 2021;31:917-929.

[60] Schwen LO, Schacherer D, Geißler C, Homeyer A. Evaluating Generic Automl Tools for Computational Pathology. Inform Med Unlocked. 2022;29:100853.

[61] Singh D, Pant PK, Pant H, Dobhal DC. Robust Automated Machine Learning (Automl) System for Early Stage Hepatic Disease Detection. In Intelligent data communication technologies and internet of things: Proceedings of ICICI.Springer Singapore. 2021:65-76.

[62] Abbas A, O'Byrne C, Fu DJ, Moraes G, Balaskas K, et. al. Evaluating an Automated Machine Learning Model That Predicts Visual Acuity Outcomes in Patients With Neovascular Age-Related Macular Degeneration. Graefes Arch Clin Exp Ophthalmol. 2022;260:2461-2473.

[63] Zhang Y, Wan DH, Chen M, Li YL, Ying H et. al. Automated Machine Learningbased Model for the Prediction of Delirium in Patients After Surgery for Degenerative Spinal Disease. CNS Neurosci Ther. 2023;29:282-295.

[64] Imrie F, Cebere B, McKinney EF, van der Schaar M. Autoprognosis 2.0: Democratizing Diagnostic and Prognostic Modeling in Healthcare With Automated Machine Learning. PLOS Digit Health. 2023;2:e0000276.

[65] Yang F, Qiao Y, Qi Y, Bo J, Wang X. Bacs: Blockchain and Automl-Based Technology for Efficient Credit Scoring Classification. Ann Oper Res. 2022:1-24.

[66] Garg V, Chaudhary S, Mishra A. Analysing Auto ML Model for Credit Card Fraud Detection. Int J Innov Res Comput Sci Technol. 2021:2347-5552.

[67] Agrapetidou A, Charonyktakis P, Gogas P, Papadimitriou T, Tsamardinos I. An Automl Application to Forecasting Bank Failures. Appl Econ Lett. 2021;28:5-9.

[68] Angarita-Zapata JS, Maestre-Gongora G, Calderín JF. A Case Study of Automl for Supervised Crash Severity Prediction. In Joint Proceedings of the 19th World Congress of the International Fuzzy Systems Association (IFSA), 12th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT), and 11th International Summer School on Aggregation Operators (AGOP). Atlantis Press. 2021:187-194.

[69] Angarita-Zapata JS, Masegosa AD, Triguero I. General-Purpose Automated Machine Learning for Transportation: A Case Study of Auto-Sklearn for Traffic Forecasting. In: Lesot MJ, Vieira S, Reformat MZ, Carvalho JP, Wilbik A, Bouchon-Meunier B, Yager RR, editors. Information processing and management of uncertainty in Knowledge Based Systems. Springer International Publishing. 2020:728-744.

[70] Garouani M, Ahmad A, Bouneffa M, Hamlich M. Amlbid: An Auto-Explained Automated Machine Learning Tool for Big Industrial Data. SoftwareX. 2022;17:100919.

[71] Feurer M, Van Rijn JN, Kadra A, Gijsbers P, Mallik N, et al. OpenML-Python: An Extensible Python API for Openml. Arxiv preprint: https://arxiv.org/pdf/1911.02490

[72] Vanschoren J, Van Rijn JN, Bischl B, Torgo L. Openml: Networked Science in Machine Learning. ACM SIGKDD Explor Newsl. 2014;15:49-60.

[73] Belete DM, Huchaiah MD. Grid Search in Hyperparameter Optimization of Machine Learning Models for Prediction of HIV/Aids Test Results. Int J Comput Appl. 2022;44:875-886.

[74] Zheng A, Casari A. Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists. O'Reilly Media. 2018.

[75] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer Wp. Smote: Synthetic Minority Over-Sampling Technique. J Artif Intell Res. 2002;16:321-357.

[76] Zhang JP, Mani I. kNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. Proceeding of International Conference on Machine Learning Workshop on Learning from Imbalanced Data Sets. 2003;126:1-7.

[77] Junsomboon N, Phienthrakul T. Combining Over-Sampling and Under-Sampling Techniques for Imbalance Dataset. In: Proceedings of the 9th international conference on machine learning and computing. USA. ACM. 2017:243-247.

[78] Song F, Guo Z, Mei D. Feature Selection Using Principal Component Analysis. In: Engineering Design and Manufacturing Informatization International Conference on System Science. 2010;1:27-30.