`iSmartML`: An Interactive and User-Guided Framework for Automated Machine Learning

Shota Amashukeli, Radwa Elshawi, Sherif Sakr
Data Systems Group, University of Tartu, Estonia
{firstname.lastname}@ut.ee

ABSTRACT

Recently, several automated machine learning (AutoML) approaches have been developed to automate the process of building machine learning pipelines *without* any human intervention. In practice, completely *excluding* the human from the loop creates several limitations. For example, most of these approaches ignore the *user-preferences* on defining or controlling the search space which consequently can impact the acceptance of the returned models by the end-users. In addition, these approaches are *not interactive* in a way that forces the users to passively wait till the end of the allocated time budget to receive the results of the automated search process without any ability to monitor or understand the search process. Furthermore, these solutions require the user to define non-straightforward parameters (e.g., time budget). To address these limitations, we demonstrate `iSmartML`, an *interactive* and *user-guided* framework for automating the machine learning modeling process. Our framework is designed to support the end-users in *defining* and *refining* the search space of the AutoML process. In addition, it provides the ability of *explaining* and understanding the results. The framework is also designed to allow its users to monitor the progress of the search process and reports a *stream* of models with alerts whenever a better pipeline is found during any point of the allocated time budget for the search process. Moreover, the framework is equipped with a *recommendation engine* that helps the end-users to define the effective search space and the adequate time budget of the AutoML process. Furthermore, the framework is equipped with a *logging* mechanism so that repeated runs on the same dataset can be *more effective* by avoiding exploring the same candidate configurations on the search space. We show that our framework can significantly improve the *utility* and *usability* of the automated machine learning process.

1 INTRODUCTION

Nowadays, machine learning techniques and algorithms are employed in almost every application domain (e.g., financial applications, advertising, recommendation systems, user behavior analytics). In general, the process of building a machine learning model is a complex and highly iterative exploratory process which requires solid knowledge and understanding of different types of algorithms. With the booming demand for machine learning applications, it has been recognized that the number of knowledgeable data scientists can not scale with the growing data volumes and application needs in our digital world[1]. Therefore, recently, several automated machine learning (AutoML) approaches (e.g. `Auto-Weka` [7], `Auto-Sklearn` [5], `SmartML` [8]) have been developed to automate the process of building the machine learning pipelines *without* any human intervention [11]. In practice, completely *excluding* the human from the loop creates several limitations. One of these limitations is that these approaches are designed as *black-boxes* in a way that limits the user's ability for defining or controlling the search space (e.g. the set of the models to include and explore in the search process). For example, the automated search process of the `Auto-Sklearn` framework is designed over a search space of 15 classifiers. However, in practice, in several sensitive application domains (e.g. Healthcare), for a better trust, domain experts would prefer to limit the search space to include only *interpretable* machine learning models (e.g. Linear Regression, Decision Tree) and exclude the non-interpretable models (e.g., Support Vector Machine, Neural Network) [3]. In other scenarios, experienced data scientists can exploit their knowledge from building previous models on similar datasets to improve the efficiency of the search process by configuring the search space to include only a selected subset of the models which are expected to be highly performing. Thus, improving the utility and usability of AutoML approaches would require providing the end-users with the ability to easily configure and control the search process.

In practice, a common important parameter for AutoML frameworks is the allocated *time budget* for the automated search process. In particular, this is a user-defined parameter which specifies the time that the AutoML framework can spend to explore the search space. Clearly, the bigger the time budget, the more time that is available for the AutoML system to explore various options in the search space and the higher probability to get a better result. However, the bigger time budget used, the longer the *waiting time* for the end-user and the higher computing resource consumption, which could be translated into a higher monetary bill in the case of using cloud-based resources. On the other hand, a small-time budget means a shorter waiting time but a lower chance to get the best recommendation. A main limitation of existing AutoML frameworks is the *lack of interactivity* during the execution of the automated search process, i.e., the user has to passively wait to receive the result of the search process at the end of the allocated time budget. In addition, due to the hugeness of the search space, users tend to increase the time budget as much as they can in order to increase the chance of finding the best performing model. However, in practice, in most of the cases, over-increasing the allocated time budget is neither efficient nor effective. Thus, it is hard for the end-users to

[1] https://hbr.org/2015/05/data-scientists-dont-scale

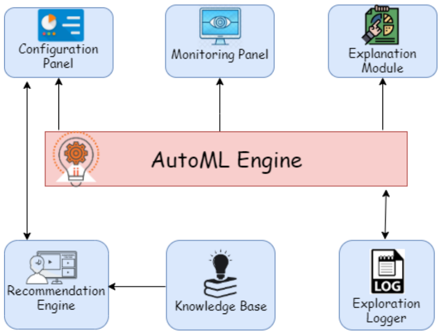

Figure 1: `iSmartML` : Framework Architecture

accurately predict or estimate what would be the adequate time budget to allocate for a given dataset.

In practice, with unsatisfactory results, users of AutoML framework may need to execute multiple runs of the automated search process over the same dataset (e.g., using different time budgets or using different search space configurations). Currently, AutoML frameworks are designed to deal with each run on the same dataset independently without any exploitation for the results of the explored search space configurations on previous runs. Such blind treatment for the different runs on the same dataset reduces the efficiency and effectiveness of the repetitiveness of the automated search process.

To address the above challenges, in this demonstration, we present `iSmartML`, an interactive and user-guided framework for improving the utility and usability of the AutoML process with the following main features:

- The framework provides the end-user with a user-friendly configuration control panel that allows non-technical users and domain experts (e.g., physicians) to easily define, configure and control the search space for the AutoML search process according to their own preferences.
- The framework is equipped with a recommendation engine, that uses a meta-learning mechanism, to help the end-users on defining the effective search space for the input dataset, potentially useful pre-processors and accurately estimating the time budget.
- The framework provides the end-user with a monitoring panel that allows tracking the progress of the search process during the whole allocated time budget and reports a stream of model configurations by sending alerts whenever a better pipeline is found during any point of time through the search process.
- The framework is equipped with a logging mechanism which enables storing the results of the explored configurations over a given dataset on one run so that repeated runs on the same dataset can be more effective by avoiding re-exploring the same candidate configurations on the search space.
- The framework is equipped with an explanation module which allows the end-user to understand and diagnose the design of the returned machine learning models using various explanation techniques. In particular, the explanation module allows the end-user to choose the model with the best satisfactory explanation for a higher trust or to use the information of the explanation process to refine and optimize a new iteration of the automated search process.

2 ISMARTML: FRAMEWORK ARCHITECTURE

Figure 1 illustrates the architecture of the `iSmartML` framework which consists of the following main components: configuration control panel, recommendation engine, monitoring panel, logger and explainer. In the following subsections, we describe each of these components.

2.1 Configuration Control Panel

In general, most of the current AutoML frameworks can not be considered to be user friendly. In particular, they still need sophisticated technical skills to be deployed and used. Such a challenge limits the usability and wide acceptance among layman users and domain experts (e.g., physicians, accountants) who commonly have limited technical skills. `iSmartML` tackles this challenge by providing the end-users with an interactive and light-weight web interface which enables them to easily define the input parameters of the search process (e.g., dataset, time budget), configure and control the search space, and visually analyze and understand the returned models.

In general, the search space of current AutoML frameworks is commonly defined over a *fixed* and *pre-defined* set of classifiers and pre-processors. For every input dataset, the automated search process is executed over the same search space without any consideration for the user-preferences. However, in practice, considering the user-preferences can have several advantages. One of these advantages is increasing the user acceptance of the returned models. For example, physicians might not be satisfied if the returned model for predicting a medical outcome from the automated search process is neural network-based, even if achieves a very high accuracy, due to the lack of interpretability. However, they could be more satisfied if the automated process has been restricted to include only interpretable models (e.g. Linear Regression, Decision Tree) even if they would achieve a slightly lower accuracy [13]. Another advantage of exploiting the user-preferences and the knowledge of some experienced user is making the search process more effective. In general, the various optimization techniques (e.g., Bayesian Optimization, Bandit Algorithms) of the automated search process attempt to tackle the challenge of the trade-off for utilizing the time budget on either *exploring* larger number of classifiers with smaller number of different hyperparameter configurations for each classifier or *exploiting* smaller number of classifier with larger number of different hyperparameter configurations [16]. Therefore, utilizing the user guidance on the search process can improve its effectiveness by focusing only on few classifiers which have the higher potential to provide the best performance on the input dataset and explore a larger number of its hyperparameter configurations. To achieve these goals, the configuration control panel of `iSmartML` allows the end-user to visually configure the various elements of the search space (e.g., pre-processors, classifiers) according to their preferences and passes this information to the underlying automated search engine to restrict and guide its search process accordingly.

2.2 Recommendation Engine

In general, the process of building a high-quality machine learning model is a highly iterative and explorative process that involves exploring the performance of various machine learning algorithms with their different configurations. Therefore, in practice, nowadays, the main challenge for data scientists is not to develop new algorithms. However, the main challenge is how to utilize their knowledge and experience from developing previous models on similar datasets to effectively build new models for new datasets and problems. Meta-Learning [14] is described as the process of learning from previous experience gained during applying various learning algorithms on different types of data, and hence reducing the needed time to learn new tasks. In the context of AutoML, a main advantage of meta-learning techniques is that they allow hand-engineered algorithms to be replaced with novel automated methods which are designed in a data-driven way. Thus, it is able to simulate the role of the machine learning expert for non-technical users and domain experts.

The recommendation engine of `iSmartML` is based on a meta-learning mechanism which is based on a knowledge base that is populated with the results of running 15 different classifiers from the popular Python-based `Scikit-learn`[2] machine learning library with different hyper-parameter configurations (up to 500), over 200 datasets with different characteristics on a set of 25 meta-features (e.g. *number of instances*, *number of features*, *number of classes*, *skewness* and *kurtosis of numerical features*) [1]. The content of the knowledge base, which is continuously growing with the results of processing any new datasets, is used for building *Meta-Models* for providing the end-users with various recommendations such as follows [4]:

- A meta-model for predicting the best performing classifiers on a given dataset based on its meta-features.
- A meta-model for predicting the adequate time budget for the automated search process based on the meta-features of the dataset, the expected training time, the expected testing time and the *tunability* for the selected classifiers. In principle, tunability is a crucial piece of information for effectively managing the time budget during the AutoML optimization process. In general, the tunability of a classifier is measured by the magnitude of its performance variance when tuning its hyperparameters [4, 9].

In addition, the recommendation engine is equipped with a *rule-based* component for recommending adequate preprocessors for the input dataset [10]. For example, one of the main challenges for the classification process is dealing with the imbalanced datasets [12]. If the meta-features of the input dataset show such a problem, the recommendation engine automatically suggests for the end-user to apply a sampling technique (e.g. SMOTE [2]) as a required preprocessing step for improving the accuracy and quality of the classification process.

2.3 Monitoring Panel

In practice, with the lack of interactivity and transparency by the AutoML frameworks during the allocated time budget, the end-user would not able to understand several aspects of the automated search process including whether the search space has been sufficiently explored and whether the allocated time budget is sufficient or should be increased for more effective search. This lack of information represents a main challenge when the returned models are not satisfactory as the end-user would not be able to reason whether these results can be improved and how. In this situation, many users would increase the time budget as much as they can hoping for better results. However, in practice, in many cases, this extra time budget can be used for exploring more of the unpromising branches in the search space or exploring branches that have very little gain, if any. For example, the accuracy of the returned models from running the `AutoSklearn` framework over the `Abalone` dataset[3] with time budgets of 4 hours and 8 hours are almost the same (25%).

To tackle these challenges, `iSmartML` is equipped with a monitoring panel which is designed to be continuously and dynamically updated with the progress of the automated search process reporting several information including a summary of the searched models with their explored configurations and performance. In addition, the monitoring panel provides continuous alerts with the best performing model discovered over time. The monitoring panel also allows the end-user to stop the search process at any point of time if the end-user has been satisfied with early results (before the end of the time budget) or if the user decided to start a new run after re-configuring the search space.

2.4 Exploration Logger

In general, the optimization techniques (e.g., Bayesian Optimization, Bandit Algorithms, Genetic Algorithms) of the automated search process of the various AutoML frameworks are non-deterministic techniques [11]. In particular, they have random components where the repeated runs of the same optimization process on the same dataset would not explore identical subset of the search space. In practice, repeated runs would explore different parts of the search space, however, with considerable potential overlap. In general, the users tend to repeat the search process when they got unsatisfactory results from the previous run. Therefore, in the repeated runs, there is a crucial need to well exploit the new allocated time budget by effectively avoiding the previously explored parts of the search space and utilizing the results of the previous runs. `iSmartML` is equipped with a logger component that allows the end-user to save the results of the explored parts during one run so that future runs on the same dataset can avoid re-training and re-testing previously explored model configuration and thus increasing the effectiveness of the reruns by exploring larger unseen parts of the search process.

2.5 Explanation Module

Recently, the interpretability of machine learning models has been receiving huge attention as the General Data Protection Regulation (GDPR) requested industries to explain any automated decision in a meaningful way [15]. In principle, interpretability techniques focus on providing insights into the black-box model to be explained and describe how a specific automated decision is taken or illustrate the most

[2]https://scikit-learn.org/

[3]https://www.openml.org/d/183

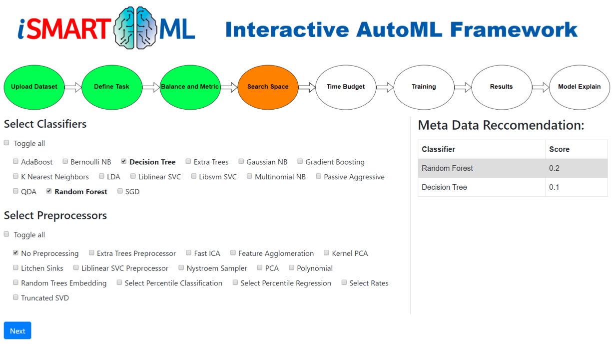

Figure 2: Screenshot: Configuring an experiment for a dataset

critical features in the input data that leads to the decision. Thus, it plays a crucial role on contributing positively toward establishing trust and confidence on machine learning models especially for non-technical users and domain experts in various application domains.

The explanation module of `iSmartML` provides the end-users with the ability to interpret and diagnose the design of the returned models from the automated search process using various visual explanation techniques (e.g., Feature Importance, Partial Dependence Plot, Individual Conditional Expectation, Feature Interaction, Global Surrogate Models) [6]. With this facility, end-users are able to diagnose the explanation of the best performing models and choose the one with the most trusted and convincing explanation for them to use. The explanation facility can also allow the end-user to understand and discover any pitfalls on the design of the return model and use this information for re-running the automated search process with refined configurations and better guidance. Clearly, with the explanation facility, the user-acceptance of the selected model can be significantly increased.

3 DEMO SCENARIO

`iSmartML` is available as a Web application[4] implemented on top of the popular AutoML framework, `Auto-Sklearn`[5], a winner of two ChaLearn AutoML challenges[6]. However, we would like to note that our framework remains agnostic towards the underlying AutoML engine. In this demonstration[7], we will present to the audience the various components of the `iSmartML` framework (Figure 1)[8]. In particular, we will show how our approach can improve the utility, usability and transparency of the AutoML process by supporting non-expert machine learning users to effectively configure and guide the search process in away that can achieve optimal or near-optimal accuracy for their datasets with little effort. In our demonstration, we will be using various open datasets from the popular repository, `OpenML`[9], in various application domains (e.g., healthcare, finance).

[4] https://bigdata.cs.ut.ee/ismartml
[5] https://automl.github.io/auto-sklearn/
[6] http://automl.chalearn.org/
[7] A demonstration screencast is available on https://youtu.be/aug5UXd1dNI
[8] The source code of the `iSmartML` framework is available on https://github.com/DataSystemsGroupUT/ismartml
[9] https://www.openml.org/

We start by introducing to the audience the challenges we tackle, the main goal and the functionalities of our framework. Then, we take the audience through the interactive automated model building process for sample datasets. We start by showing different features which is provided for the end-user. For example, the user can upload a dataset file, use the control panel to choose the classifiers and pre-processors to be included in the automated research process (Figure 2), specify the time budget, consider and evaluate the suggestions for the different parameters from the recommendation engine, monitor the automated search process and receive continuous live updates, receive and diagnose the returned models using the various explanation techniques until the user is able to choose the model with the most satisfactory performance and trust.

ACKNOWLEDGMENT

This work is funded by the European Regional Development Funds via the Mobilitas Plus programme (grant MOBTT75)

REFERENCES

[1] Besim Bilalli, Alberto Abelló, and Tomas Aluja-Banet. 2017. On the predictive power of meta-features in OpenML. *International Journal of Applied Mathematics and Computer Science* 27, 4 (2017).
[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.
[3] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).
[4] Salijona Dyrmishi, Radwa Elshawi, and Sherif Sakr. 2019. A Decision Support Framework for AutoML Systems: A Meta-Learning Approach. In *Proceedings of The 1st IEEE ICDM Workshop on Autonomous Machine Learning (AML)*.
[5] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2019. Auto-sklearn: Efficient and Robust Automated Machine Learning. In *Automated Machine Learning*. Springer.
[6] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2019. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)* 51, 5 (2019).
[7] Lars Kotthoff et al. 2017. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *The Journal of Machine Learning Research* 18, 1 (2017).
[8] Mohamed Maher and Sherif Sakr. 2019. SmartML: A Meta Learning-Based Framework for Automated Selection and Hyperparameter Tuning for Machine Learning Algorithms. In *EDBT*.
[9] Philipp Probst, Anne-Laure Boulesteix, and Bernd Bischl. 2019. Tunability: Importance of Hyperparameters of Machine Learning Algorithms. *Journal of Machine Learning Research* 20, 53 (2019).
[10] Sergio Ramírez-Gallego et al. 2017. A survey on data preprocessing for data stream mining: Current status and future directions. *Neurocomputing* 239 (2017).
[11] Radwa El Shawi, Mohamed Maher, and Sherif Sakr. 2019. Automated Machine Learning: State-of-The-Art and Open Challenges. *CoRR* abs/1906.02287 (2019). arXiv:1906.02287 http://arxiv.org/abs/1906.02287
[12] Yanmin Sun, Andrew KC Wong, and Mohamed S Kamel. 2009. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence* 23, 04 (2009), 687–719.
[13] Gilmer Valdes et al. 2016. MediBoost: a patient stratification tool for interpretable decision making in the era of precision medicine. *Scientific reports* 6 (2016).
[14] Joaquin Vanschoren. 2018. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548* (2018).
[15] Paul Voigt and Axel Von dem Bussche. 2017. The EU General Data Protection Regulation (GDPR). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* (2017).
[16] Marc-André Zöller and Marco F Huber. 2019. Survey on Automated Machine Learning. *arXiv:1904.12054* (2019).