(IJRST) 2023, Vol. No. 13, Issue No. 3, Jul-Sep **e-ISSN: 2249-0604, p-ISSN: 2454-180X**

Enhancing Efficacy of Machine Learning Model Selection Process for Big Data Science Projects by Introducing an Adaptive Method Based on Dynamic Factors

Arnav Goenka

Vellore Institute of Technology, Vellore, Tamil Nadu, India

DOI:10.37648/ijrst.v13i03.014

[1]*Received: 18 August 2023; Accepted: 19 September 2023; Published: 24 September 2023*

ABSTRACT

Data science projects typically involve a machine learning (ML) process characterized by evolving data, code, and models. For instance, as datasets grow in size, they may become suitable for ML models that require larger datasets. However, the dynamic factors influencing model selection must be better understood and explicitly represented. This paper introduces ongoing work on an adaptive method for ML model selection in big data science projects. The proposed method includes (i) identifying the factors that influence model selection based on heuristics from the literature and (ii) modelling the variability of these factors using a feature diagram and constraints that trigger adaptive reconfiguration—changes in model selection due to shifts in these factors. The method's applicability is demonstrated through an illustrative use case. By providing a clearer understanding of the dynamic factors that influence model selection, this method shows how these factors can be explicitly represented and automated. This enhanced understanding can lead to a more explicit, efficient, adaptive, and explainable model selection process, ultimately laying the groundwork for developing novel dynamic software product lines to support this process.

INTRODUCTION

Data science projects are increasingly prevalent across diverse fields such as business, health, and commerce [1]. Despite this growth, especially in projects involving machine learning (ML), more process-oriented methodologies, tools, and frameworks still need to be developed to support these endeavours. These projects necessitate ML development processes incorporating data, code, models, and extensive team interactions. The complexity of these feedback loops and interactions makes developing software engineering solutions for ML applications more challenging than traditional software applications. While significant research has focused on proposing techniques and tools to facilitate ML model deployment, there is still a need for improved automation and process support for end-to-end ML application development.

Despite the increasing integration of ML models into production processes, the research on developing automated pipelines faces significant challenges. Efficiently incorporating ML models into production, particularly in monitoring and adapting to changes over time, remains a hurdle.

In the dynamic ML operational environment, constant model retraining is often required to maintain appropriate performance, quality, and efficiency. This necessitates continuous monitoring of ML models post-deployment to detect outliers and data drifts, assess model performance and data metrics, and support adaptations prompted by observed changes.

ML project lifecycles proposed in the literature, such as the Microsoft ML lifecycle [4], consider continuous monitoring after model deployment, allowing feedback loops to previous stages to manage needed changes and reconfiguration. For instance, since ML models are data-dependent and data is continuously changing, models must be retrained to ensure ongoing improvement and efficient application outcomes [2], [5]. According to [5], maintaining high-quality ML services requires additional procedures beyond deployment, such as data monitoring and model

[1] How to cite the article: Goenka A. (September 2023); Enhancing Efficacy of Machine Learning Model Selection Process for Big Data Science Projects by Introducing an Adaptive Method Based on Dynamic Factors; *International Journal of Research in Science and Technology*, Vol 13, Issue 3, 134-139, DOI: http://doi.org/10.37648/ijrst.v13i03.014

performance evaluation. Situations like concept drift, where there are discrepancies between training and test sets, may impact predictive performance and necessitate adaptive changes.

Adaptation in ML Model Selection

Model selection is a critical phase in the ML project lifecycle. It's essential to understand the factors affecting model selection and provide methods and tools to support changes and adaptation in this process. Choosing the appropriate algorithm is often challenging due to various influencing factors, including sample size, application goals, and data types.

Data is not static; it evolves over time, significantly impacting model selection, outcomes, and performance. In traditional ML modelling, the data sample and problem are defined, and algorithm selection may follow established heuristics. However, in dynamic scenarios, such as changing data characteristics or attributes, this process becomes more complicated and urgent. For instance, when evaluating a new drug's side effects, the initial data may be unlabelled and small in sample size. As additional labelled data is incorporated over time, the algorithm selection must be revisited to ensure model accuracy[3].

Thus, two key research questions arise: (RQ1) What factors affect model selection as defined by known heuristics? (RQ2) How can the variability of these factors be modelled, and how do they trigger the adaptive reconfiguration of the models?

Adaptive Method

This paper presents an adaptive method for ML model selection in data science projects. The proposed method involves (i) identifying the factors affecting model selection as proposed in the literature and (ii) modelling the variability of these factors using a feature diagram and constraints that trigger adaptive reconfiguration—changes in model selection due to shifts in these factors. An illustrative use case demonstrates the method's applicability.

The proposed approach captures interactions among various abstractions that impact algorithm selection (e.g., dataset, prediction type, and outcomes such as performance) as these elements change over time. For example, an increased data sample size may lead to different algorithm selections. Changes in prediction categories, the discovery of new significant features, or variations in outcomes (e.g., accuracy) may necessitate model changes (e.g., addressing the drift problem) [6]. According to Hummer et al. [7], in the dynamic AI application lifecycle, new builds can be triggered by data or code changes, activating retraining processes or requiring model replacement.

Benefits of the Method

The proposed method enhances understanding of the dynamic factors influencing model selection, their explicit effects, and how these adaptive factors can be represented and automated. This understanding can lead to a more explicit, efficient, adaptive, and explainable model selection process, increasing productivity[8]. Additionally, the method advances the state of the art by introducing adaptive and explainable processes. Adaptive processes help manage variations in model selection, while explainable processes support accountability by clarifying the reasons behind method selection. This adaptive method can ultimately form the foundation for creating novel dynamic software product lines to support the model selection process.

APPROACH

ML Model Selection Mechanism

We have reviewed the literature and identified several approaches for ML model selection based on heuristics, including those provided by Scikit-Learn. Scikit-Learn is a user-friendly Python package that offers non-ML specialists' access to a wide range of state-of-the-art ML algorithms for solving supervised and unsupervised problems [9].

Our adaptive method for ML model selection is grounded in the Scikit-Learn heuristics, as illustrated in Figure 1. This figure highlights various factors that influence the model selection, including ML techniques, sample size, prediction type (e.g., category, quantity, structure), number of categories, type of data (e.g., text data), presence of labelled data, number of features, and performance metrics (e.g., accuracy, F1-Score).

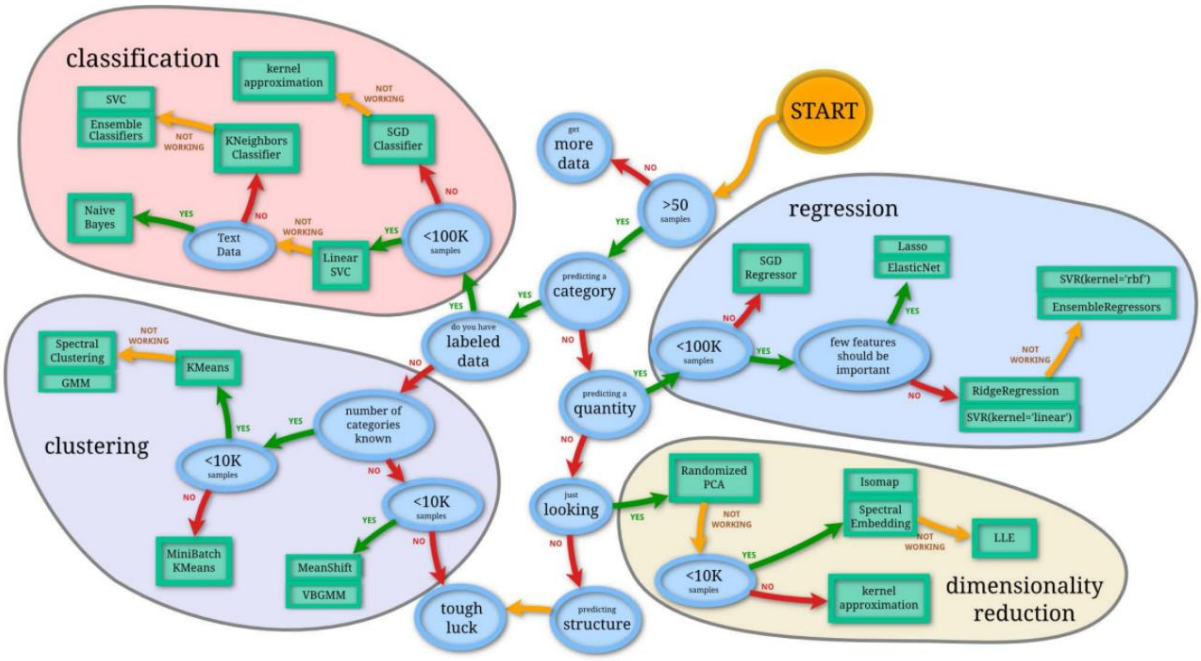

Fig. 1. Heuristics for selecting a machine learning algorithm from Scikit-Learn.

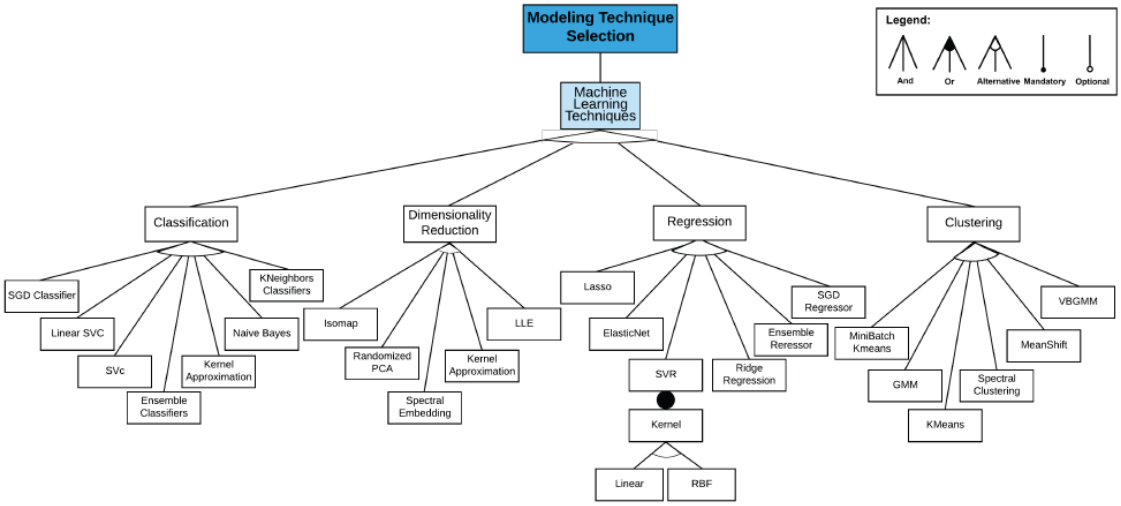

Fig. 2. Feature diagram of the proposed approach to represent Scikit Modelling Techniques.

Feature-Oriented Modelling

We modelled the variability of these factors using a feature diagram and constraints that trigger adaptive reconfiguration, meaning model selection changes due to variability factors. Based on the Scikit-Learn heuristics, we derived two feature diagrams.

The first diagram, shown in Figure 2, represents the feature diagram for Scikit-Learn modelling techniques, including classification, dimensionality reduction, regression, and clustering.

The second diagram, depicted in Figure 3, illustrates the feature diagram for Scikit-Learn application requirements, which encompass dataset requirements (e.g., sample size), functional requirements (e.g., prediction type), and non-functional requirements (e.g., performance metrics such as accuracy, precision, recall, and F1-Score). Feature model constraints act as reconfiguration triggers that activate depending on changes in the application requirements.

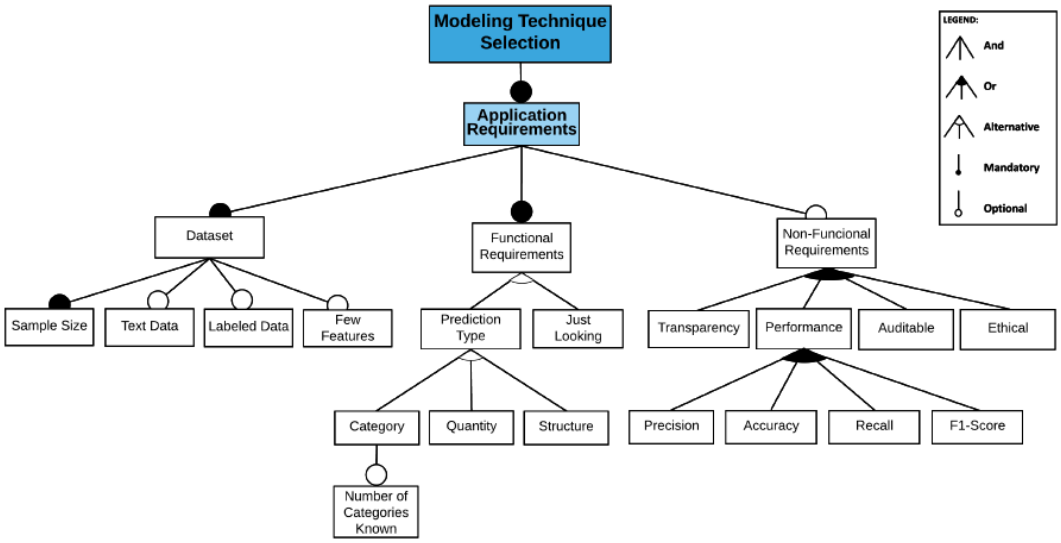

Fig. 3. Feature diagram of the proposed approach based on Scikit heuristics.

ILLUSTRATIVE EXAMPLE

We selected an application example from the data science literature to provide more details about the proposed adaptive method and illustrate its use.

Dataset: Heart Failure Prediction

This application aims to predict the survival of patients with heart failure based on clinical information [10]. The dataset comprises data from 299 patients with heart failure, containing 12 clinical features such as age, sex, diabetes status, and serum creatinine levels. The target variable, 'death event,' is binary, indicating whether the patient died or survived before the end of an average follow-up period of 130 days [10].

1. ML Model Selection: After defining the dataset and application purpose, we applied our adaptive method to select the most suitable ML algorithm. Based on the Scikit-Learn heuristics (see section III-A), our approach first identifies the problem category—regression, classification, clustering, or dimensionality reduction. After determining the category, the system selects the most appropriate algorithm from the available options. For this application, LinearSVC was selected. We set the F1-Score as the performance criterion, which maximizes both sensitivity and specificity, a crucial aspect in medical contexts, according to Leenings et al. [10].

Figure 4 presents the feature model of the application example, showing the initial configuration representing the application requirements described by Leenings et al. [10] and the ML model selected by our mechanism. The dotted figures in Figures 5 and 6 indicate other possible feature configurations that could be instantiated based on changes in the application dataset or requirements.

2. Adaptations in ML Model Selection: We also explored other possibilities for implementing the same application based on potential changes at design or runtime, demonstrating how these changes could drive adaptations in ML model selection.

Figure 5 illustrates an adaptation triggered by an increased dataset volume to over 100K entries. According to Scikit-Learn heuristics, the SGD classifier is recommended for such large datasets. Thus, a new feature diagram would be instantiated to represent the application after this adaptation, replacing the Linear SVC technique with the SGD classifier.

CONCLUSIONS AND FUTURE WORK

This paper presents an adaptive method for ML model selection in data science projects. The proposed method identifies factors affecting model selection based on heuristics and models the variability of these factors using a feature diagram and constraints that trigger adaptive reconfiguration. This method advances the state of the art by enabling a more adaptive and explainable model selection process. The model selection feature diagram can be extended to incorporate factors defined by other heuristic approaches (e.g., from IBM and Microsoft). Ultimately, this

method can form the foundation for creating novel dynamic software product lines to support the model selection process, potentially implemented via a service-oriented approach where service calls adapt to handle changes.

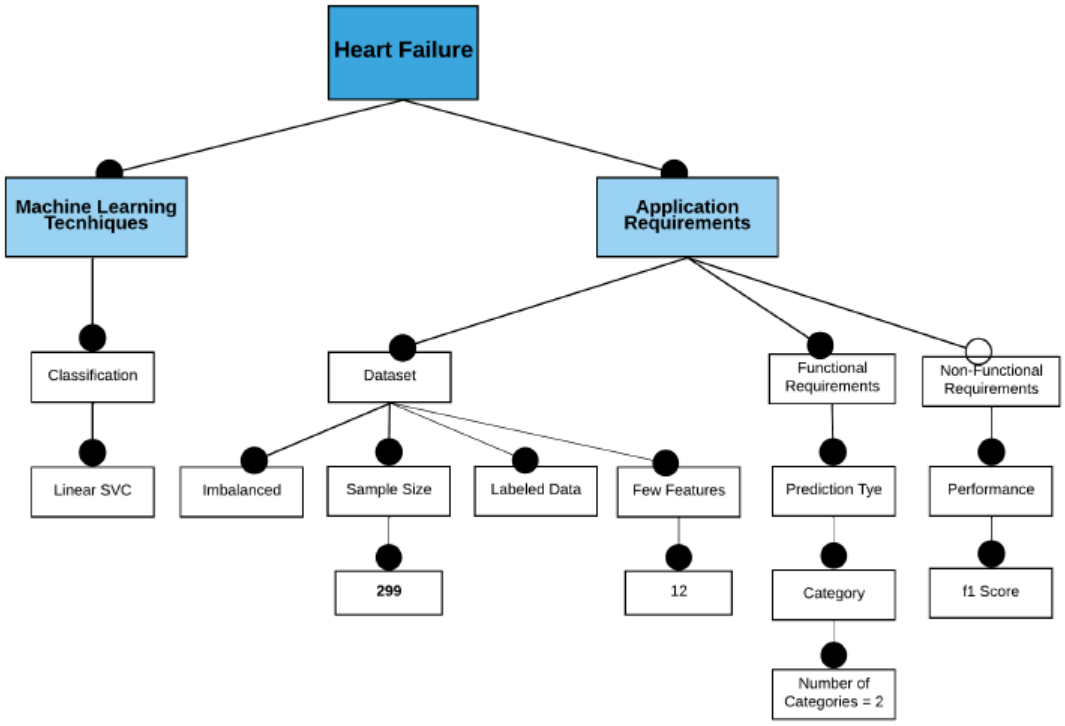

Fig. 4. Feature model of the application example illustrated according to the configuration that was implemented.

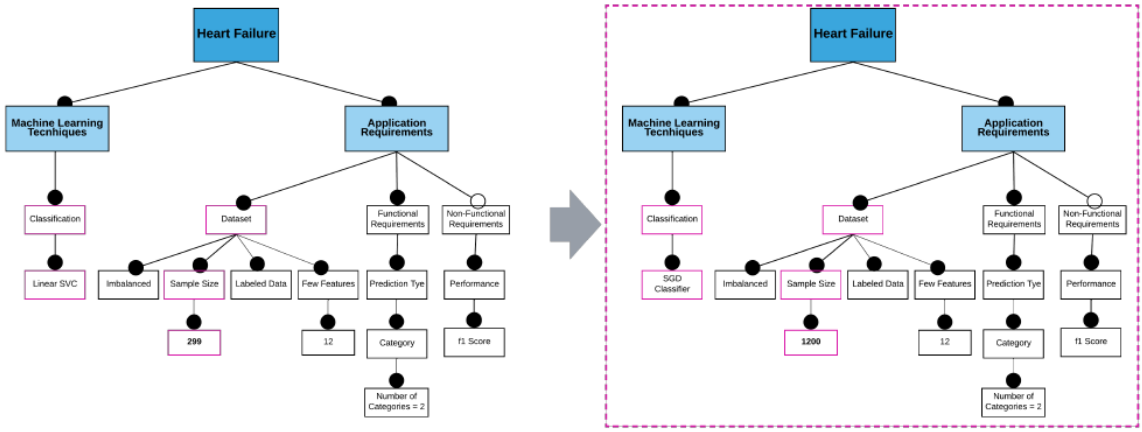

Fig. 5. Feature model representing an adaptation triggered by changes in the application dataset.

REFERENCES

[1] J. S. Saltz and I. Krasteva, “Current approaches for executing big data science projects—a systematic literature review,” PeerJ Computer Science, vol. 8, p. e862, 2022.

[2] G. Symeonidis, E. Nerantzis, A. Kazakis, and G. A. Papakostas, “Mlopsdefinitions, tools and challenges,” in 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2022, pp. 0453–0460.

[3] S. K. Karmaker, M. M. Hassan, M. J. Smith, L. Xu, C. Zhai, and K. Veeramachaneni, “Automl to date and beyond: Challenges and opportunities,” ACM Computing Surveys (CSUR), vol. 54, no. 8, pp. 1–36, 2021.

[4] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP). IEEE, 2019, pp. 291–300.

[5] J. Klaise, A. Van Looveren, C. Cox, G. Vacanti, and A. Coca, "Monitoring and explainability of models in production," arXiv preprint arXiv:2007.06299, 2020.

[6] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," IEEE Transactions on Knowledge and Data Engineering, vol. 31, no. 12, pp. 2346–2363, 2018.

[7] W. Hummer, V. Muthusamy, T. Rausch, P. Dube, K. El Maghraoui, A. Murthi, and P. Oum, "Modelops: Cloud-based lifecycle management for reliable and trusted ai," in 2019 IEEE International Conference on Cloud Engineering (IC2E). IEEE, 2019, pp. 113–120.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., "Scikit-learn: Machine learning in python," the Journal of machine Learning research, vol. 12, pp. 2825–2830, 2011.

[9] D. Chicco and G. Jurman, "Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone," BMC medical informatics and decision making, vol. 20, no. 1, pp. 1–16, 2020.

[10] R. Leenings, N. R. Winter, L. Plagwitz, V. Holstein, J. Ernsting, K. Sarink, L. Fisch, J. Steenweg, L. Kleine-Vennekate, J. Gebker et al., "Photonai—a python api for rapid machine learning model development," Plos one, vol. 16, no. 7, p. e0254062, 2021.