KoopaML: A Graphical Platform for Building Machine Learning Pipelines Adapted to Health Professionals

Francisco José García-Peñalvo[1], Andrea Vázquez-Ingelmo[1], Alicia García-Holgado[1], Jesús Sampedro-Gómez[2], Antonio Sánchez-Puente[2], Víctor Vicente-Palacios[3], P. Ignacio Dorado-Díaz[2], Pedro L. Sánchez[2] *

[1] GRIAL Research Group, University of Salamanca (Spain)
[2] Cardiology Department, Hospital Universitario de Salamanca, SACyL. IBSAL, Facultad de Medicina, University of Salamanca, and CIBERCV (ISCiii) (Spain)
[3] Philips Healthcare (Spain)

Received 23 February 2022 | Accepted 1 June 2022 | Early Access 23 January 2023

Abstract

Machine Learning (ML) has extended its use in several domains to support complex analyses of data. The medical field, in which significant quantities of data are continuously generated, is one of the domains that can benefit from the application of ML pipelines to solve specific problems such as diagnosis, classification, disease detection, segmentation, assessment of organ functions, etc. However, while health professionals are experts in their domain, they can lack programming and theoretical skills regarding ML applications. Therefore, it is necessary to train health professionals in using these paradigms to get the most out of the application of ML algorithms to their data. In this work, we present a platform to assist non-expert users in defining ML pipelines in the health domain. The system's design focuses on providing an educational experience to understand how ML algorithms work and how to interpret their outcomes and on fostering a flexible architecture to allow the evolution of the available components, algorithms, and heuristics.

Keywords

Artificial Intelligence, HCI, Health Platform, Information System, Medical Data Management, Medical Imaging Management.

DOI: 10.9781/ijimai.2023.01.006

I. Introduction

MACHINE Learning (ML) has become a powerful approach to tackle complex tasks that involve analyzing significant amounts of data. Data-intensive contexts, such as the health domain, benefit directly from applying ML algorithms to their data, supporting tasks such as identifying patterns, clustering, classification, predictions, etc., that could become time- and resource-consuming if approached through manual paradigms. The application of ML to health data has proven its usefulness in specific challenges like diagnoses, disease detection, segmentation, assessment of organ functions, etc. [1] -[3].

However, applying ML approaches is not straightforward. More specifically, using them in sensitive domains (such as health) could be hazardous if practitioners do not fully understand the results derived from the models.

ML does not only consist of applying a set of pre-defined functions. It needs a deep understanding of the input data, the transformations that need to be performed to fit a model, the selection of a proper model, and its quality metrics before using trained models in production. Otherwise, the outputs could lead to wrong conclusions, losses, discrimination, and even negligence [4] -[7].

Therefore, it is necessary to balance data domain knowledge and ML expertise. While ML experts have a wealth of knowledge about ML algorithms, they can lack understanding regarding the input data. The same applies to health professionals; they have a profound knowledge of the data domain, but they would not obtain quality models without programming or ML skills.

In this scenario, it is necessary to provide practitioners with tools that alleviate this knowledge gap, enabling health professionals to implement ML pipelines and learn how, when, and why to apply specific models or functions to their data. This way, the introduction of ML in medical tasks could yield complementary support to automate and enhance decision-making processes without consuming an excessive quantity of resources and time.

In this context we pose the following research question:

RQ1. Which features can ease the application of ML algorithms in the medical context?

Driven by this research question, we present a graphical platform (KoopaML) to offer intuitive and educational interfaces to build and run ML pipelines to tackle these challenges. The primary target audience of this platform is non-expert users interested in learning and applying ML models to their domain data. We followed a user-centered design approach to capture relevant requirements and necessities from potential user profiles involved in this context.

* Corresponding author.
E-mail addresses: fgarcia@usal.es (F. J. García-Peñalvo), andreavazquez@usal.es (A. Vázquez-Ingelmo), aliciagh@usal.es (A. García-Holgado), jmsampedro@saludcastillayleon.es (J. Sampedro-Gómez), asanchezpu@saludcastillayleon.es (A. Sánchez-Puente), victor.vicente.palacios@philips.com (V. Vicente-Palacios), pidorado@saludcastillayleon.es (P. Ignacio Dorado-Díaz), plsanchez@saludcastillayleon.es (P. L. Sánchez).

Please cite this article in press as: F. J. García-Peñalvo, A. Vázquez-Ingelmo, A. García-Holgado, J. Sampedro-Gómez, A. Sánchez-Puente, V. Vicente-Palacios, P. Ignacio Dorado-Díaz, P. L. Sánchez. KoopaML: A Graphical Platform for Building Machine Learning Pipelines Adapted to Health Professionals, International Journal of Interactive Multimedia and Artificial Intelligence, (2023), http://dx.doi.org/10.9781/ijimai.2023.01.006

In addition, we focused on providing a flexible architecture to allow expert users to extend the platform's functionality through new custom algorithms, components, or new heuristics to guide the definition of ML pipelines.

In this paper, we describe the design process, the platform's architecture, its underlying processes, and the feedback obtained from experts regarding the first development stages of the system.

The rest of the work is structured as follows. Section 2 provides an overview of similar tools for learning and building ML and data science pipelines. Section 3 describes the methodology followed for eliciting requirements and the technologies employed to implement the platform. Section 4 details the platform's architecture and modular decomposition, while section 5 describes the implemented functionalities. Finally, sections 7 and 8 discuss the results and conclude the work, respectively.

II. Related Work

Plenty of helper tools has been developed due to the increasing popularity of ML. Specifically, there are three main categories: programming frameworks and libraries, platforms for experts and non-experts, and platforms that support learning and understanding regarding how ML algorithms and pipelines work.

The first category encloses several well-known programming libraries: TensorFlow [8], Apache Mahout [9], and other Python frameworks like PyTorch (https://pytorch.org/), Scikit-learn (https://scikit-learn.org/), or Keras.io (https://keras.io/). These libraries provide an abstraction layer to implement ML models, but they require programming skills to employ them properly.

The second category focuses on visual environments that assist users through intuitive interfaces in creating and defining ML pipelines. Weka, for instance, provides a collection of algorithms for data mining tasks. One of its environments enables users to define data streams by connecting nodes representing data sources, preprocessing tasks, evaluation methodologies, visualizations, or algorithms, among other [10], [11].

On the other hand, Orange Data Mining Field [12] allows the definition of data mining workflows, with several methodologies, operations, and visualizations available through a user-friendly interface. The possibility of introducing customized dashboards [12] -[14] to present the outcomes of ML tasks an extremely valuable feature to ease the comprehension of the pipeline stages. Another tool with similar features is Rapid Miner [15], which follows a node-and-link philosophy to specify and define ML workflows. These applications provide robust and complex features through intuitive interfaces and interaction methods, adding abstraction to programming libraries.

The last category refers to platforms whose primary goal is to offer a didactic experience and learning resources to ease understanding ML algorithms and workflows. Tools within this category provide user-friendly and simple graphical interfaces avoiding technical details. Examples include Machine Learning for Kids (https://machinelearningforkids.co.uk/) or LearningML (https://web.learningml.org/).

Although several solutions are developed to assist non-expert users in the definition of ML pipelines, it is difficult to adapt them to specific contexts with particular necessities and requirements. For these reasons, we opted to develop a customized tool focused on the provenance of an educative experience for health professionals that want to start applying ML models to support their tasks.

III. Methodology

A. Requirements Elicitation

We identified the main features and specifications of the platform through a requirements elicitation process. Specifically, we interviewed potential users and domain experts, including physicians, computer scientists, and managers.

The output of this process was the description of the platform's basic features:

1. Definition of ML pipelines
2. Execution of ML pipelines
3. Interpretation of ML results
4. Visualization of ML results
5. Data validation
6. Heuristics management

The first two features are related to implementing ML pipelines by connecting different tasks, including data preprocessing and cleaning, ML algorithms, and evaluation functions. The platform allows users to choose different ML algorithms and configure their parameters. Users can personalize their pipelines by connecting nodes and analyzing each step's intermediate results.

Features 3 and 4 are related to the outcomes obtained during the execution of the pipelines. Each stage will output new results, and these results need to be understood to gain insights. For these reasons, the platform needs to provide methods to convey and assist the interpretation of the pipeline outputs through explanations, annotations, and data visualizations. This process is vital because a wrong interpretation of the results could lead to useless results and lose all its potential benefits.

On the other hand, the quality of the training process not only depends on the algorithm's configuration but also on the quality of input data. The platform needs to support validation processes and emphasize cleaning and preprocessing functions before training ML models. This feature focuses on providing information regarding the applicability of the available algorithms to the input data and potential issues (missing values, data imbalance, data samples, data types, etc.).

The last functionality refers to applying heuristics to assist non-expert users in the definition of ML pipelines. The management of heuristics and recommendation rules should be flexible to support the evolution of the suitability of algorithms depending on the context. Therefore, the platform will allow the modification and addition of new heuristics to provide more flexibility and build customized rule-based recommenders.

During the elicitation process, two user roles were identified. This categorization of users is essential to adapt the functionalities depending on the role, as well as their privileges:

- Non-expert users. The primary users of the platform. Non-expert users (mainly physicians) who know the data domain are interested in IA and ML but don't have enough skills to create ML pipelines programmatically.
- AI experts. Experts will have access to the ML pipelines workspace, but they will also have privileges to define and modify heuristics to configure the recommendations or preferred workflows of the platform.

B. Development

As introduced, one of the main goals of this work is to provide a flexible platform with the capability to evolve to include discoveries in ML. Therefore, it is crucial to rely on flexible technologies and paradigms that support the reusability of components.

ML pipelines share common features and can be represented through abstract elements to leverage their commonalities and foster the reusability of core assets. We followed the software product line (SPL) paradigm and domain engineering to capture ML pipelines and tasks' commonalities and variability points and arrange the software components accordingly [16] - [20].

With this approach, it is possible to reuse these "building blocks" and modify/add new ones without impacting the rest of them. On the other hand, building each pipeline task as an independent component with well-defined inputs and outputs also meets the requirement of inspecting intermediate results and even executing pipelines step by step.

We materialized the variability of pipelines through SciLuigi (https://github.com/pharmbio/sciluigi), a wrapper for Spotify's Luigi Python library (https://github.com/spotify/luigi), which supports the definition of dynamic workflows avoiding hard-coded dependencies [21], [22].

C. Validation

To validate the first version of KoopaML, we carried out an expert judgement [23] validation with experts from the medical and AI fields. Three experts were recruited to thoroughly test the platform and seek for issues regarding its contents and interaction mechanisms.

The three participants are AI developers in the medical domain, so they were able to test the platform from the two perspectives.

IV. Architecture

Providing flexible and extensible architecture is crucial in this field, as approaches constantly improve and evolve. This section outlines the platform's architecture and the mechanisms employed to support the evolution of its components.

A. Modules

The architecture of KoopaML is based on different modules connected by information flows. One of the primary purposes of this design is to provide flexible pipelines with reusable components.

In this regard, we followed a domain engineering approach through the previously described requirements elicitation process with potential users and literature reviews.

Following this approach, we propose four general functional blocks that will interact and collaborate among them to provide support for the implementation of flexible ML workflows:

- User management module
- Heuristics management module
- Pipelines management module
- Tasks management module

The user management module provides the services related to authentication, sessions, and roles. The heuristics management module allows IA experts to modify the heuristics through a graphic interface. The pipelines management module provides a workspace to create ML pipelines using visual elements. Finally, the tasks management module defines the operations related to each ML pipeline potential stage.

Fig. 1 shows the schematic overview of the platform's architecture with the C4 model notation [24].

B. Pipelines' Structure

While the previous functional blocks provide flexibility to evolve the system's features, they still need fine-grained flexibility regarding the implementation of ML pipelines.

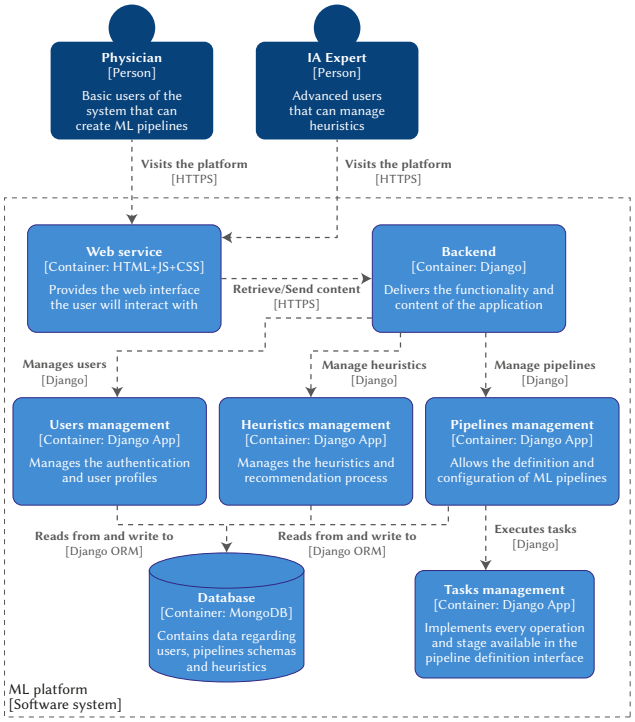

Fig. 1. Outline of the platform's architecture.

Following the software product line architecture paradigm [16] -[20], we divided ML pipelines into fine-grained tasks with well-defined inputs and outputs. Through this approach, the tasks management module acts as a repository of loosely coupled ML-related tasks, in which algorithms and operations can be added and modified without impacting the features of the remaining modules/tasks.

As explained in the methodology section, this encapsulation of ML tasks is achieved through the SciLuigi library. Fig. 2 outlines the structure of the pipelines following this approach.

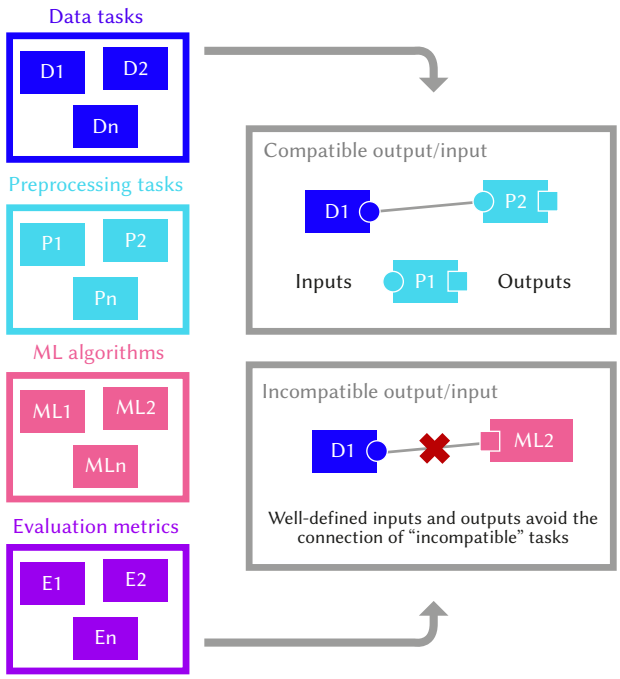

Fig. 2. Outline of programming approach. Each task contains its own logic and belongs to a specific category. Inputs and outputs compatibilities (in terms of information flows) are computed from each node's logic.

Tasks are categorized following their high-level functionality (tasks related to data upload, data preprocessing, ML algorithms, or evaluation metrics). Then, more specific tasks are implemented; for example, within the "ML algorithms" category, we can find particular algorithms such as Naïve Bayes, Random Forest, Linear Regression, etc.

Users can instantiate nodes from each category and connect them through their inputs and outputs. These inputs and outputs are also categorized to ensure that information flows are compatible among the instantiated nodes.

The connection restrictions between nodes are implemented in the interface to ensure that the SciLuigi pipeline is correctly instantiated. With this method, the construction of the final SciLuigi pipeline is straightforward.

The simplified code in Fig. 3 outlines the implementation of a SciLuigi workflow through the pipeline specification defined by the user in the graphical interface. The main challenge was related to the dynamic connection of inputs and outputs. SciLuigi requires knowing the specific inputs/outputs names beforehand to connect them through explicit attribute value assignment. Lines 14-17 (Fig. 3) show how this issue was solved using the *setattr* and *getattr* Python methods, allowing dynamic access to class attributes.

```
1  tasks = { }
2
3  for node in pipeline.config:
4      tasks[node.id] = self.new_task(create_instance[node.type],
5                                     pipeline_id=pipeline.id,
6                                     node_id=node.id,
7                                     node.params)
8
9      end_nodes = [ ]
10     for node in pipeline.config:
11         if len(node.inputs) > 0:
12             for input in node.inputs:
13             # Connect the node inputs to the corresponding node outputs
14             setattr(tasks[node.id],
15                     "in_{0}".format(input.input_name),
16                     getattr(tasks[input.connected_node.id],
17                     "out_{0}".format(input.connected_node.output_name)))
18
19         if len(node.outputs) == 0:
20             end_nodes.append(tasks[node_id])
21
22     return tuple(end_nodes)
```

Fig. 3. Algorithm to materialize a pipeline specification into a SciLuigi workflow (syntax simplified for readability).

This solution provides more flexibility and eases the addition and modification of new tasks, as the whole pipeline can be instantiated without hard-coding specific dependencies or class types.

V. KoopaML

A. Prototype

A prototype was developed and evaluated to complement the requirement elicitation process through a focus group. This methodology enabled us to capture more requirements and validate the platform's conceptual design before its implementation.

Fig. 4 shows a screenshot of the interface for defining ML pipelines through node-link structures.

The focus group involved different user profiles, including physicians and AI experts related to the health domain. The outcomes of this study can be consulted in [25]. The feedback was positive and helpful for starting the implementation of the tool.

B. Functional System

As explained throughout this work, a crucial characteristic of the interface is that it should be simple to avoid overwhelming users with several complex concepts at once and robust to enable the definition

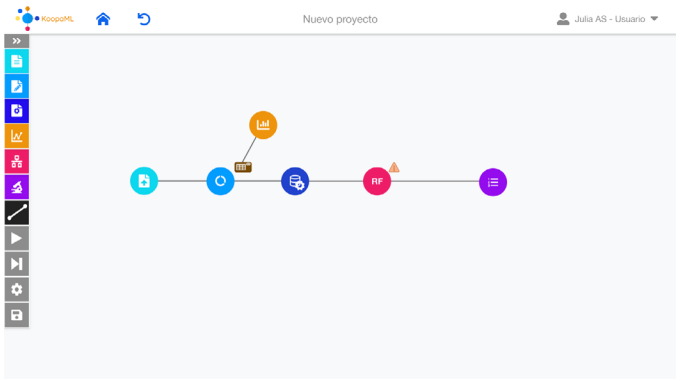

Fig. 4. Prototype of the workspace to define ML pipelines.

of ML pipelines with enough detail. This section provides an overview of the interface proposal and the different features of the first version of KoopaML.

1. ML Pipelines

When creating a new project or pipeline, the system displays an empty workspace with a toolbar containing the tasks included in the ML workflow. As introduced in previous sections, tasks are divided into high-level categories to ease users' search of specific nodes (Fig. 5).

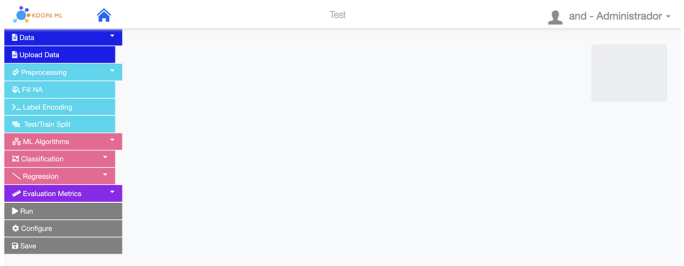

Fig. 5. New project workspace and available nodes.

Users can click on specific tasks or drag and drop them into the workspace to start configuring the pipeline. Fig. 6 shows the "Upload CSV" node. This node is particularly complex because several circumstances need to be considered when uploading data:

1. CSV files can be separated by different characters, such as commas or semicolons. For this reason, the node allows the configuration of varying separator values through a text input.
2. Some nodes could take as an input a single column (or a subset of them). That is why each dataset's columns need to be considered single outputs and be accessible to create connections among nodes.
3. The whole dataset is also considered a single output ("Data" socket in Fig. 6) to avoid multiple column connections and ease the data flows. This output includes the whole dataset (the set of all columns contained in the uploaded file).
4. Related to the previous point, some columns might be discarded from the dataset (i.e., columns that hold several missing values or aren't relevant to the problem). A checkbox beside each column allows users to select the columns that will be part of the dataset.
5. Finally, data related to the health domain can hold a significant quantity of variables. However, showing all variables as outputs in the "Upload CSV" node at once could impact the user experience. For this reason, a threshold has been configured to show only the first five columns of a dataset, allowing the user to add the remaining columns through a multiple selection input. This way, users can focus only on variables that need explicit connections through their ML pipeline.

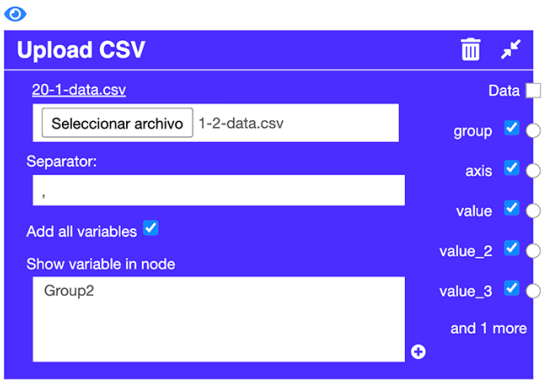

Fig. 6. A node for uploading CSV files.

Fig. 7 shows a simple ML workflow in which:

1. Categorical data is encoded through a Label Encoder.
2. The output from the label encoding process is then split into training and test sets. This node needs to know the variable to predict to perform the division of data. In this specific case, the variable "group" will be the one to be expected through this pipeline.
3. Training datasets are connected to a Random Forest classifier.
4. Finally, the trained model and the test datasets are connected to an evaluation node to measure the model's accuracy.

Users can execute the pipeline whenever they want by clicking the "Run" button, triggering the backend to build the pipeline by connecting tasks using the algorithm presented in Fig. 3. Once the pipeline has been executed, the workspace displays the results (or any error) individually in each node (bottom image of Fig. 7). Storing intermediate results leverages one of the main benefits of using SciLuigi, which is the possibility of re-running failed tasks individually without triggering the whole pipeline again.

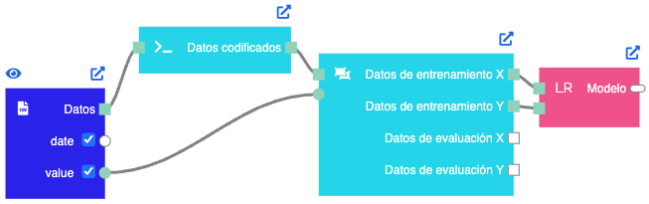

Fig. 7. Execution and results of an ML pipeline. Intermediate results of each node can be consulted by clicking on the top-right icon of each node.

Through this approach, intermediate results can also be inspected individually. On the one hand, Fig. 8 displays the intermediate results from the test/train splitting node. This node yields four results: test and training datasets separated by the column to predict. The fig. 8 shows two of these intermediate results (the test datasets).

Evaluation metrics are also treated as intermediate results. In this case, the measurement of the accuracy of the trained model yielded 33% of correct predictions (Fig. 9).

2. Data Validation

Data validation and exploratory analysis are crucial steps when building successful ML pipelines. If data is not properly inspected and preprocessed, trained models could yield useless results. KoopaML provides a summary screen to assist users in the exploration process. This section is divided into three main blocks.

The first block provides a table view of the whole dataset. This view allows users to see all columns and rows of the uploaded data files and navigate through them in detail (Fig. 10).

Fig. 8. Results were derived from splitting the uploaded data into test and training datasets.

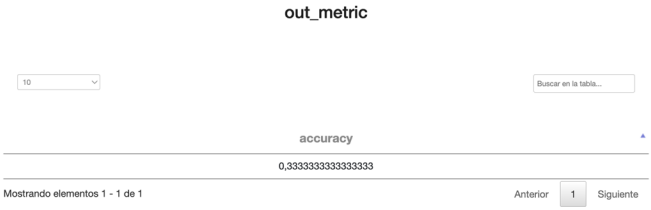

Fig. 9. Accuracy of the trained model. Note that low accuracy is related to the small dataset that illustrates the system's functionalities.

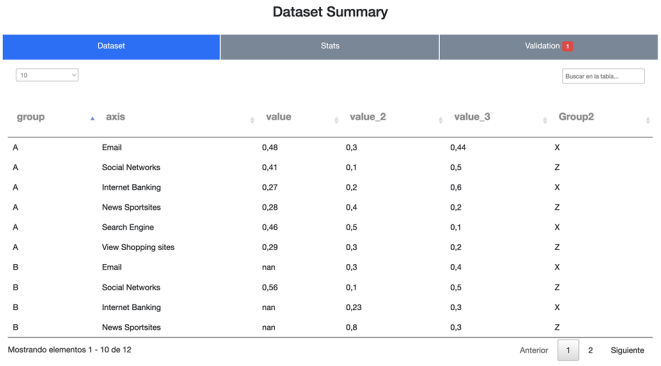

Fig. 10. Results were derived from splitting the uploaded data into test and training datasets.

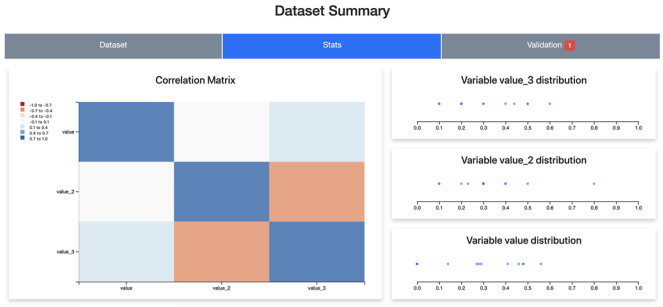

Fig. 11. Information dashboard of the input dataset characteristics.

The second summary block is a data dashboard in which practical data details, such as the distribution of values, data types, number of missing values, or a correlation matrix, are presented visually to ease the analysis of the dataset characteristics (Fig. 11). The dashboard is automatically generated and tailored according to the user needs [26] - [28].

Finally, the last block is focused on alerting users regarding potential issues of the dataset (Fig. 12), such as columns with significant quantities missing values, mixed data types, unbalanced categories, etc. Users are encouraged to consider or solve these issues through this feature before using the dataset in a pipeline.

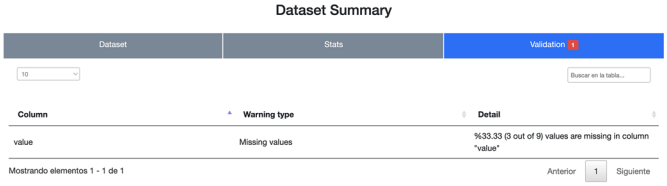

Fig. 12. Validation screen.

3. Heuristics Management

As explained before, one of the goals of the platform users is to learn from the experience of developing pipelines and build skills related to the application of ML. However, this learning experience needs to be guided by expert knowledge.

We have tackled this challenge through the definition and management of custom heuristics. KoopaML allows expert users to design heuristics in graphical decision trees to yield recommendations and guide the implementation of pipelines.

Heuristics are represented through the DSL provided by the flowchart.js (https://github.com/adrai/flowchart.js) library. This library allows textual and graphical representation of flow charts, providing a fine-grain manipulation of heuristics and rule-based recommendations (Fig. 13).

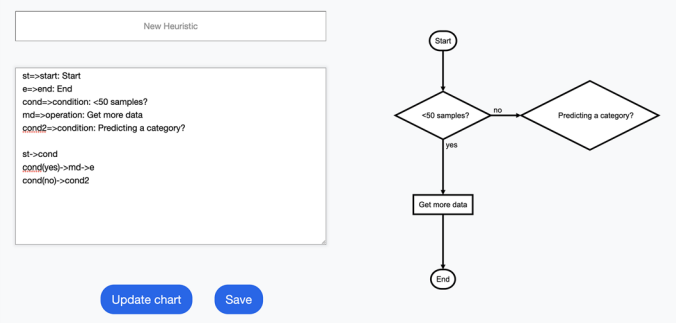

Fig. 13. Example of the definition of a heuristic.

VI. Expert Validation

The results of the expert validation were favorable. Overall, the platform was rated as useful to overcome the difficulties of creating ML pipelines in a medical context.

Regarding the issues encountered, apart from minor bugs that were fixed, the following can be highlighted:

- **Error reports.** The experts pointed out the possibility of having a variety of errors related to the execution of the pipeline. In the current version of KoopaML, these errors were displayed through tooltips associated to each node. However, experts indicated that it might be useful to have an unified report listing every error or warning raised during the execution of the pipeline.
- **Model metrics.** KoopaML allows the computation of different metrics to validate the trained models. For this matter, the user needs to select and connect every metric they want to calculate. This could be time-consuming if several metrics are to be analyzed. In this sense, the experts advised the possibility of unifying every metric on a node, and let the user select the metrics directly from there instead of carrying out the selection one by one.
- **Data visualizations.** The data summary presented in the previous section was highly valued by the experts. Following this idea, they recommended implementing a dashboard with visualizations related to model metrics as well.
- **Cross validation.** The experts pointed out that, in practice, they use cross validation [29], and thus, that the platform should support this approach.

Other comments were related to the addition of a wide set of algorithms and metrics, as well the possibility of configuring the hyperparameters of the algorithms through the interface.

VII. Discussion

This work presents the first version of KoopaML: a platform for automating and learning the definition of ML pipelines. We followed a user-centered approach for the design and development process, considering the primary goal of the system: to ease the application of ML for non-specialized users.

This version has been subject to iterative development with continuous feedback from experts. For instance, the "Upload CSV" node design shown in Fig. 6 resulted from different evaluations in which domain experts exposed issues encountered or potential improvements when uploading their domain data.

Although there are commercial tools that tackle the automation of these processes, the specific requirements that arise from the medical context asked for a customized platform that aligns with the necessities of end-users (in this case, physicians with lack of data science skills but that are interested in applying ML).

On the other hand, another related benefit of the customized tool is implementing communication mechanisms among other already developed devices for the cardiology department at the University Hospital of Salamanca [30]. Connecting different platforms would foster the creation of a technological ecosystem [31] with powerful and transparent data management and data science features adapted to the health sector requirements.

The platform's architecture is designed to allow flexibility and evolution due to the changing nature of AI and ML methods. The abstraction of pipelines into tasks with well-defined inputs and outputs has facilitated the user interface design and the final implementation of the workflows through libraries such as SciLuigi, matching the same node-link structures.

In addition to the workspace for instantiating pipelines, the platform also provides an interface to support the exploratory analysis of data. This interface was included after the evaluation of the platform by expert users, who asked for more feedback related to the input data.

Finally, one novel feature of KoopaML is the heuristics management module. This module enables the definition of heuristics through a DSL and its graphical representation. Heuristics can be stored to rely on different knowledge bases depending on the data domain, for example. The dynamic heuristic definition fosters the flexibility of the recommendations and guided support provided within the workspace during the implementation of ML pipelines. Moreover, their structured format allows the inclusion of external heuristics from other knowledge bases stores [32], [33].

Regarding the expert validation, the results were highly valuable and useful to set the foundations of new improvements and features, as well as to identify minor bugs. Having experts from both AI and medical fields enabled the identification of issues and shortcomings of the current version of the platform. For these reasons, we will continue performing this kind of evaluations, as they provide insights related to theoretical concepts that will be difficult to reach with lay users.

Following the research question posed in the introduction and the results of the expert validation, the platform has been developed taking into account the necessities of the medical domain. The implementation of an interface with simple and visual mechanisms (such as drag and drop or visually connecting two nodes to instantiate a pipeline) set the foundations for a platform that can be used by non-expert users.

On the other hand, the development of the heuristics management module will also allow the definition of recommendations that could be adapted to any kind of user. These features will provide additional assistance while creating and interpreting ML pipelines.

VIII. Conclusions

This work describes the design process, architecture, and features of KoopaML: a graphical platform for building machine learning pipelines adapted to health professionals.

The platform has been designed to support the evolution and addition of new tasks related to ML pipelines through abstraction mechanisms. The abstraction of tasks has allowed simplifying the user interface and the automatic implementation of the graphically instantiated pipelines.

KoopaML assists users in the definition of ML pipelines, execution of ML pipelines, interpretation and visualization of ML results, data validation, and heuristics management.

Future research lines will involve further expert validations of the platform, as well as in-depth user tests to measure the usability, ease of use, and effectiveness of the tool.

Acknowledgment

This research work has been supported by the Spanish *Ministry of Education and Vocational Training* under an FPU fellowship (FPU17/03276). This research was partially funded by the Spanish Government Ministry of Economy and Competitiveness through the DEFINES project grant number (TIN2016-80172-R) and the Ministry of Science and Innovation through the AVisSA project grant number (PID2020-118345RB-I00). This work was also supported by national (PI14/00695, PIE14/00066, PI17/00145, DTS19/00098, PI19/00658, PI19/00656 Institute of Health Carlos III, Spanish Ministry of Economy and Competitiveness and co-funded by ERDF/ESF, "Investing in your future") and community (GRS 2033/A/19, GRS 2030/A/19, GRS 2031/A/19, GRS 2032/A/19, SACYL, Junta Castilla y León) competitive grants.

References

[1] G. Litjens *et al.*, "A survey on deep learning in medical image analysis," (in eng), *Med Image Anal,* vol. 42, pp. 60-88, Dec 2017, doi: 10.1016/j.media.2017.07.005.

[2] S. González Izard, R. Sánchez Torres, Ó. Alonso Plaza, J. A. Juanes Méndez, and F. J. García-Peñalvo, "Nextmed: Automatic Imaging Segmentation, 3D Reconstruction, and 3D Model Visualization Platform Using Augmented and Virtual Reality," (in eng), *Sensors (Basel),* vol. 20, no. 10, p. 2962, 2020, doi: 10.3390/s20102962.

[3] S. G. Izard, J. A. Juanes, F. J. García Peñalvo, J. M. G. Estella, M. J. S. Ledesma, and P. Ruisoto, "Virtual Reality as an Educational and Training Tool for Medicine," *Journal of Medical Systems,* vol. 42, no. 3, p. 50, 2018/02/01 2018, doi: 10.1007/s10916-018-0900-2.

[4] J. C. Weyerer and P. F. Langer, "Garbage in, garbage out: The vicious cycle of ai-based discrimination in the public sector," in *Proceedings of the 20th Annual International Conference on Digital Government Research*, Dubai, United Arab Emirates, 2019, pp. 509-511, doi: https://doi.org/10.1145/3325112.3328220.

[5] X. Ferrer, T. van Nuenen, J. M. Such, M. Coté, and N. Criado, "Bias and Discrimination in AI: a cross-disciplinary perspective," *IEEE Technology and Society Magazine,* vol. 40, no. 2, pp. 72-80, 2021, doi: 10.1109/MTS.2021.3056293.

[6] S. Hoffman, "The Emerging Hazard of AI-Related Health Care Discrimination," *Hastings Center Report,* vol. 51, no. 1, pp. 8-9, 2021, doi: 10.1002/hast.1203.

[7] S. Wachter, B. Mittelstadt, and C. Russell, "Why fairness cannot be automated: Bridging the gap between EU non-discrimination law and AI," *Computer Law & Security Review,* vol. 41, p. 105567, 2021, doi: 10.2139/ssrn.3547922.

[8] M. Abadi *et al.*, "TensorFlow: A System for Large-Scale Machine Learning," in *12th USENIX Symposium on Operating Systems Design and Implementation OSDI 16*. Savannah, GA: USENIX Association, 2016, pp. 265-283.

[9] R. Anil *et al.*, "Apache Mahout: Machine Learning on Distributed Dataflow Systems," *Journal of Machine Learning Research,* vol. 21, no. 127, pp. 1-6, 2020. [Online]. Available: https://jmlr.org/papers/v21/18-800.html.

[10] E. Frank, M. Hall, G. Holmes, R. Kirkby, B. Pfahringer, and I. H. Witten, "Weka-A Machine Learning Workbench for Data Mining," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach Eds. Boston, MA: Springer, 2009.

[11] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *SIGKDD Explor. Newsl.,* vol. 11, no. 1, pp. 10–18, 2009, doi: 10.1145/1656274.1656278.

[12] A. Vázquez-Ingelmo, F. J. García-Peñalvo, and R. Therón, "Information Dashboards and Tailoring Capabilities - A Systematic Literature Review," *IEEE Access,* vol. 7, pp. 109673-109688, 2019, doi: 10.1109/ACCESS.2019.2933472.

[13] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher, "What Do We Talk About When We Talk About Dashboards?," *IEEE Transactions on Visualization Computer Graphics,* vol. 25, no. 1, pp. 682 - 692, 2018, doi: 10.1109/TVCG.2018.2864903.

[14] S. Few, *Information dashboard design*. Sebastopol, CA, USA: O'Reilly Media, 2006.

[15] S. Land and S. Fischer, "Rapid miner 5," *Rapid-I GmbH,* 2012.

[16] J. Bosch, "From software product lines to software ecosystems," in *SPLC*, 2009, vol. 9, pp. 111-119.

[17] L. Chen, M. Ali Babar, and N. Ali, "Variability management in software product lines: a systematic review," 2009.

[18] P. Clements and L. Northrop, *Software product lines*. Addison-Wesley Boston, 2002.

[19] C. Kästner, S. Apel, and M. Kuhlemann, "Granularity in software product lines," in *2008 ACM/IEEE 30th International Conference on Software Engineering*, 2008: IEEE, pp. 311-320.

[20] J. Van Gurp, J. Bosch, and M. Svahnberg, "On the notion of variability in software product lines," in *Proceedings Working IEEE/IFIP Conference on Software Architecture*, 2001: IEEE, pp. 45-54.

[21] S. Lampa, J. Alvarsson, and O. Spjuth, "Towards agile large-scale predictive modelling in drug discovery with flow-based programming design principles," *Journal of Cheminformatics,* vol. 8, no. 1, p. 67, 2016/11/24 2016, doi: 10.1186/s13321-016-0179-6.

[22] S. Lampa, M. Dahlö, J. Alvarsson, and O. Spjuth, "SciPipe: A workflow library for agile development of complex and dynamic bioinformatics pipelines," *GigaScience,* vol. 8, no. 5, 2019, doi: 10.1093/gigascience/giz044.

[23] R. Skjong and B. H. Wentworth, "Expert judgment and risk perception," in *Proceedings of the Eleventh (2001) International Offshore and Polar Engineering Conference (Stavanger, Norway, June 17-22, 2001)*: International Society of Offshore and Polar Engineers, 2001.

[24] S. Brown. "The C4 Model for Software Architecture." https://c4model.com/ (accessed 16-05-2022).

[25] A. García-Holgado *et al.*, "User-Centered Design Approach for a Machine Learning Platform for Medical Purpose," Cham, 2021: Springer International Publishing, in Human-Computer Interaction, pp. 237-249.

[26] A. Vázquez Ingelmo, A. García-Holgado, F. J. García-Peñalvo, and R. Therón Sánchez, "A Meta-modeling Approach to Take into Account Data Domain Characteristics and Relationships in Information Visualizations," in *9th World Conference on Information Systems and Technologies*, Azores, Portugal, Á. Rocha, H. Adeli, G. Dzemyda, F. Moreira, and A. M. R. Correia, Eds., 2021, vol. 2: Springer Nature, in Trends and Innovations in Information Systems and Technologies, WorldCIST 2021, pp. 570-580, doi: 10.1007/978-3-030-72651-5_54. [Online]. Available: http://hdl.handle.net/10366/145626

[27] A. Vázquez-Ingelmo, A. García-Holgado, F. J. García-Peñalvo, and R. Therón, "Proof-of-concept of an information visualization classification approach based on their fine-grained features," *Expert Systems*, vol. n/a, no. n/a, p. e12872, In Press, doi: https://doi.org/10.1111/exsy.12872.

[28] A. Vázquez-Ingelmo, F. J. García-Peñalvo, and R. Therón, "Taking advantage of the software product line paradigm to generate customized user interfaces for decision-making processes: a case study on university employability," *PeerJ Computer Science*, vol. 5, p. e203, 2019/07/01 2019, doi: 10.7717/peerj-cs.203.

[29] C. Schaffer, "Selecting a classification method by cross-validation," *Machine Learning*, vol. 13, no. 1, pp. 135-143, 1993.

[30] F. García-Peñalvo *et al.*, "Application of Artificial Intelligence Algorithms Within the Medical Context for Non-Specialized Users: the CARTIER-IA Platform," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 6, 2021.

[31] A. García-Holgado and F. J. García-Peñalvo, "Validation of the learning ecosystem metamodel using transformation rules," *Future Generation Computer Systems*, vol. 91, pp. 300-310, 2019/02/01/ 2019, doi: https://doi.org/10.1016/j.future.2018.09.011.

[32] A. Martínez-Rojas, A. Jiménez-Ramírez, and J. Enríquez, "Towards a Unified Model Representation of Machine Learning Knowledge," presented at the Proceedings of the 15th International Conference on Web Information Systems and Technologies, Vienna, Austria, 2019. [Online]. Available: https://doi.org/10.5220/0008559204700476.

[33] C. Kumar, M. Käppel, N. Schützenmeier, P. Eisenhuth, and S. Jablonski, "A Comparative Study for the Selection of Machine Learning Algorithms based on Descriptive Parameters," in *Proceedings of the 8th International Conference on Data Science, Technology and Applications. Volume 1. DATA*, Prague, Czech Republic, 2019, pp. 408-415, doi: 10.5220/0008117404080415.

Francisco José García-Peñalvo

He received the degrees in computing from the University of Salamanca and the University of Valladolid, and a Ph.D. from the University of Salamanca (USAL). He is Full Professor of the Computer Science Department at the University of Salamanca. In addition, he is a Distinguished Professor of the School of Humanities and Education of the Tecnológico de Monterrey, Mexico. Since 2006 he is the head of the GRIAL Research Group GRIAL. He is head of the Consolidated Research Unit of the Junta de Castilla y León (UIC 81). He was Vice-dean of Innovation and New Technologies of the Faculty of Sciences of the USAL between 2004 and 2007 and Vice-Chancellor of Technological Innovation of this University between 2007 and 2009. He is currently the Coordinator of the PhD Programme in Education in the Knowledge Society at USAL. He is a member of IEEE (Education Society and Computer Society) and ACM.

Andrea Vázquez-Ingelmo

Andrea Vázquez-Ingelmo received the bachelor's degree in computer science from the University of Salamanca, Salamanca, in 2016 and the master's degree in computer science from the same university in 2018. She is a member of the Research Group of Interaction and eLearning (GRIAL), where she is pursuing her PhD degree in computer science. Her area of research is related to human-computer interaction, software engineering, data visualization and machine learning applications.

Alicia García-Holgado

She received the degree in Computer Sciences (2011), a M.Sc. in Intelligent Systems (2013) and a Ph.D. (2018) from the University of Salamanca, Spain. She is member of the GRIAL Research Group of the University of Salamanca since 2009. Her main lines of research are related to the development of technological ecosystems for knowledge and learning processes management in heterogeneous contexts, and the gender gap in the technological field. She has participated in many national and international R&D projects. She is a member of IEEE (Women in Engineering, Education Society and Computer Society), ACM (and ACM-W) and AMIT (Spanish Association for Women in Science and Technology).

Jesús Sampedro-Gómez

Jesús Sampedro-Gómez is industrial engineer by the Universidad Politécnica of Madrid. He is also a PhD student by the University of Salamanca and works as data scientist in the Cardiology Department of the University Hospital of Salamanca.

Antonio Sánchez-Puente

Antonio Sánchez Puente, PhD, is a junior researcher from CIBER working at the cardiology department of the University Hospital of Salamanca as a data scientist. He was awarded his doctorate in physics by the University of Valencia for his study of gravity theories before turning his career around the application of artificial intelligence in medicine.

Víctor Vicente-Palacios

Víctor Vicente-Palacios holds a PhD from the University of Salamanca in Statistics and works as a Data Scientist at Philips Healthcare in the area of AI applied to Medicine. He is also an alumnus of the Data Science for Social Good program (University of Chicago) and organizer of PyData Salamanca.

P. Ignacio Dorado-Díaz

P. Ignacio Dorado-Díaz holds a PhD from the University of Salamanca in Statistics. He is currently working as a research coordinator in the Cardiology Department of the Hospital de Salamanca in addition to being a professor at the Universidad Pontificia de Salamanca.

Pedro Luis Sánchez

Pedro Luis Sánchez holds a doctorate in medicine from the University of Salamanca. He is currently the head of the Cardiology Department at the University Hospital of Salamanca, in addition to being a professor at the University of Salamanca.