A maturity model for digital ML tools to be used in manufacturing environments

Jannik Rosemeyer[a]*, Christian Neunzig[b], Cem Akbal[a], Joachim Metternich[a], Bernd Kuhlenkötter[b]

[a] *Institut für Produktionsmanagement, Technologie und Werkzeugmaschinen, TU Darmstadt, Otto-Berndt-Straße 2, 64287 Darmstadt, Germany*
[b] *Lehrstuhl für Produktionssysteme, Ruhr-Universität Bochum, Industriestraße 38C, 44894 Bochum, Germany*

* Corresponding author. Tel.: +49-6151-822-9658; *E-mail address:* j.rosemeyer@ptw.tu-darmstadt.de

Abstract

Low or no code machine learning platforms, whereof tools such as KNIME, DataRobot or WEKA are among the best-known, have facilitated the implementation of machine learning applications in industrial environments in recent years by transferring programming tasks to an assistance system instead of demanding users to provide the respective skills. Despite the high number of innovations, to the best of the authors' knowledge, there is no comprehensive classification scheme to assess the autonomy of those tools. Hence, this paper demonstrates a maturity model that classifies the assistance level of existing digital machine learning tools with respect to the requirements of manufacturing environments. It is based on the levels of driving automation and concretized by the so-called CRISP-ML(Q) procedure model. The model allows researchers to rate newly developed tools against existing ones and aims to serve as a baseline for future research. To evaluate the added value to the research landscape, semi-structured interviews with four ML experts were conducted. Finally, five commercial tools were categorized in the model to show its applicability.

Keywords: machine learning; digital assistance systems; autonomy levels, levels of driving automation

1. Introduction

The rapid growth of machine learning (ML) applications in various industries has led to a surge in demand for user-friendly, accessible tools that can simplify the implementation process. Low or no-code ML platforms, such as KNIME [2], DataRobot [3], and WEKA [4], have emerged as viable solutions that enable users without extensive programming skills to develop and deploy ML applications in industrial settings. These platforms have gained popularity for their ability to bridge the gap between domain experts and data scientists, ensuring more efficient collaboration [5].
Despite the widespread adoption of those tools, there is currently no comprehensive classification scheme to assess the autonomy levels of such platforms. In this paper, a maturity model is proposed that systematically classifies the assistance levels of existing digital ML tools, with a focus on their application in manufacturing environments. Drawing inspiration from the levels of driving automation (LDA) [1] and leveraging the well-established Cross-Industry Standard Process model for the development of machine learning applications with a quality assurance methodology (CRISP-ML(Q)) [6] framework, the model aims to serve as a baseline for evaluating new and existing tools in terms of their autonomy. By providing a robust maturity model for assessing the autonomy of low or no-code ML platforms, this paper seeks to facilitate informed decision-making for researchers and practitioners, promote the development of more advanced and autonomous ML tools for industrial applications, and enable more efficient collaboration between domain experts and data scientists.

The remainder of this paper is structured as follows. Chapter 2 describes the existing research landscape with emphasis on digital assistance systems for autonomy in machine learning. In chapter 3, the methodology for the development is described and subsequently the maturity model itself is displayed. Thereupon, an evaluation with ML experts was conducted as well as existing commercial ML tools have been categorized through the newly developed model in chapter 4. Chapter 5 provides a summary and gives an outlook to future research.

2. Related Works

In this chapter, the literature landscape on autonomy in machine learning, with an emphasis on classifying autonomy levels of digital assistance systems for ML applications is discussed.
Lee et al.'s [7] influential work investigated autonomic machine learning, aiming to reduce expert intervention, and suggested a classification of autonomy levels based on diverse factors within the ML process. Standardization in machine learning has mostly focused on software development, defining the computing environment necessary for the international standardization of high-performance ML frameworks. Recent years have seen significant progress in autonomic ML platforms, self-directed learning, and

CC license: Published under CC-BY 4.0 International: https://creativecommons.org/licenses/by/4.0

autonomic feature extraction. Additionally, there is growing interest in automating the determination of hyperparameters for ML algorithms [8, 9].
Commercial platforms such as Google Cloud AutoML [10] have surfaced, allowing developers with limited ML knowledge to create and deploy ML models using cloud-based environments. These platforms often employ various methods for determining hyperparameters, including grid searches, random searches, Powell's method, the Nelder-Mead method, tree-based Parzen estimation (HyperOpt), sequential model-based algorithm setting (Auto-Weka), and Gaussian process-based models [11].
Prominent tools in the ML landscape, such as Auto-sklearn [8], Auto-Weka [12] and TPOT [13], utilize techniques like Bayesian optimization, sequential model-based algorithm configuration (SMAC), and genetic programming for algorithm selection and hyperparameter tuning [11]. These developments have contributed to the creation of more autonomous ML systems.
Considerable research has focused on distributed processing and parallel processing in machine learning to tackle the computational demands of these systems [14]. Platforms such as GraphLab [15], SystemML [16], SimSQL [17], and MLBase [18] have emerged, providing programming and runtime support for ML tasks. Furthermore, frameworks like Giraph [19], Apache Spark [20], and DryadLinq [21] have been developed to facilitate the execution of ML tasks, despite not being explicitly designed for this purpose.
The development and adoption of digital assistance systems for ML applications have garnered attention due to their potential to broaden access to advanced ML techniques. Low or no-code ML platforms, including KNIME [2], DataRobot [3], RapidMiner [22], and WEKA [4], have been extensively examined for their ability to enable non-experts to develop and deploy ML applications across various industries.
Incorporating human expertise and feedback into the ML process has led to significant advancements in human-in-the-loop ML and explainable AI (XAI), resulting in more transparent, interpretable, and user-friendly ML models. Automated machine learning (AutoML), an essential aspect of many low or no-code platforms, seeks to automate the process of selecting, configuring, and optimizing ML models, further advancing the development of autonomous digital assistance systems [7]. Indeed, low or no-code ML platforms have found success across diverse industries, such as manufacturing, healthcare, finance, and agriculture, showcasing the potential of digital assistance systems for ML to drive innovation and address real-world challenges.
Likewise, several maturity levels have been developed over the past years. Exemplarily, Karmaker et al [23] developed a rating system for the level of autonomy of ML implementation solutions with seven levels. It starts with the mere programming of known languages such as Python or Java (termed *no automation*) and then successively includes additional steps, whereby ending on a stage where all tasks from use case formulation to result summary as well as recommendation are incorporated. In addition, the enterprise Zelros AI describe a model ranging from a basic approach with a minimum of functionalities to more advanced systems that include domain knowledge [24]. In the early levels, tasks are reduced to the cross-validation, as well as the testing of several machine learning algorithms and their parameters, respectively. In return, the fourth and thereby last level displays current state of research and development.

In conclusion, the related works discussed here underscore the importance of classifying autonomy levels in digital assistance systems for ML applications. This section offers a comprehensive understanding of the current state of the art and the challenges in creating more autonomous ML systems. Yet, it was found that only few authors have already published similar levels of autonomy in the past and that those remain on a rather abstract level. Hence, this paper seeks to develop a maturity model that is in line with the existing ones but demonstrates advantages through its concretization.

3. Development of the maturity model

3.1. Foundations of the model

Within this subchapter, the underlying methodology for the development of the maturity model is outlined. Whereas this subchapter aims to provide the theoretical foundations of the LDA and CRISP-ML(Q), their intersection with respect to the maturity model will be outlined in the subsequent subchapter. The LDA is a recognized basis for maturity models, from which others are derived (e.g. Industry 4.0 [25]). It can be assumed through the existing transfer to the Industry 4.0 context that the model is also suitable for ML applications.
The LDA [1] is divided into six levels, ranging from no automation to full automation (a graphical overview can be taken from Figure 1). On **level 0**, a vehicle is controlled exclusively by the driver and no interactive system is used. This implies the absence of any assistance system. **Level 1** displays a driver assisted vehicle, which can take over certain functions. Such examples might be cruise or distance control to the vehicle ahead. **Level 2** of the LDA is defined as a partially automated vehicle behaviour. The goal is to be able to drive straight autonomously, having a lane assistance system and taking over complete behaviour in traffic jams. At **level 3**, a vehicle can conditionally control the drive. Respective examples include overtaking manoeuvres on a highway or intelligent speed control in relation to current traffic behaviour. This implies that a car can interact with its environment. However, the driver must be alert and intervene in case of urgent issues. The step up to **level 4**, which represents highly automated driving, requires no longer a driver. In certain situations, a vehicle system can solve problems automatically and is fully connected to the environment. The driver does not need to intervene or be attentive while driving. Exemplary functions at level 4 are autonomous parking or recognizing green and red phases of traffic light systems in cities. The **fifth** and last **level** of the LDA represents a vehicle that takes over all driving tasks and can recognize and cope with all road traffic problems without exception.

However, the LDA remains on an abstract level whereby not providing specific criteria for the classification of a given tool on a concrete level. Consequently, the LDA is specified by the CRISP-ML(Q) [6].

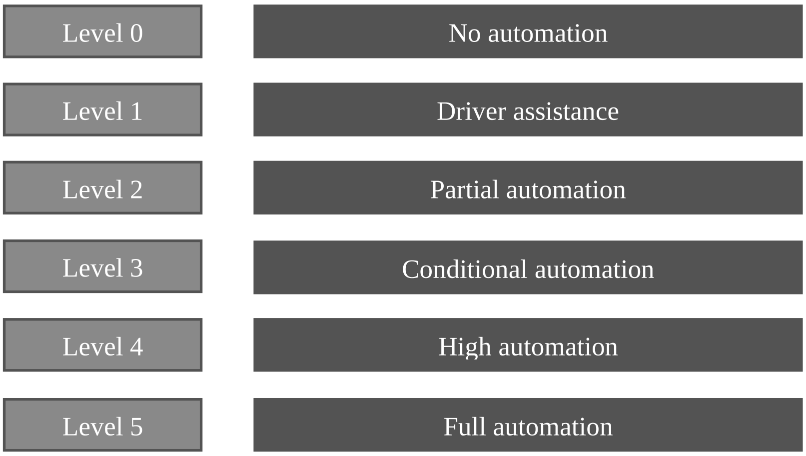

Figure 1: Levels of driving automation, Source: [1]

Moreover, this chapter also provides an introduction into the CRISP-ML(Q) procedure model following the descriptions of [6]. Its steps can simultaneously be seen in Figure 2. The CRISP-ML(Q) starts with **Business and Data Understanding**. Within this phase, business goals are defined with the help of business stakeholders and the translation of the use case into an ML goal is discussed. In addition, data specifications are captured and reviewed to finally assess the feasibility of the overall project. The following step 2 is defined as **Data Engineering**. It aims to ensure that a data set is available for the subsequent modeling phase. If not yet present, users are called to create a data set according to their needs. This phase is not a static application. Consequently, a repetition of this step can take place at a later point within the project. Data Engineering includes tasks such as feature selection, feature engineering and data augmentation, respectively. In addition, the data file format is specified, as some ML tools require specific variables and input types. The definition of standards helps to reduce the risk of errors in merging and detecting erroneous data. The choice of modeling techniques with respect to ML is related to the business objectives established in step 1. Step 3 of the CRISP-ML(Q) is termed **Model Building**. The requirements and constraints are used as inputs to perform the model selection. The goal of this phase is to develop several models that meet the requirements defined in the previous steps. Hence, model selection and model training are covered here. At the end of this phase, ensemble methods are applied to make a decision based on the

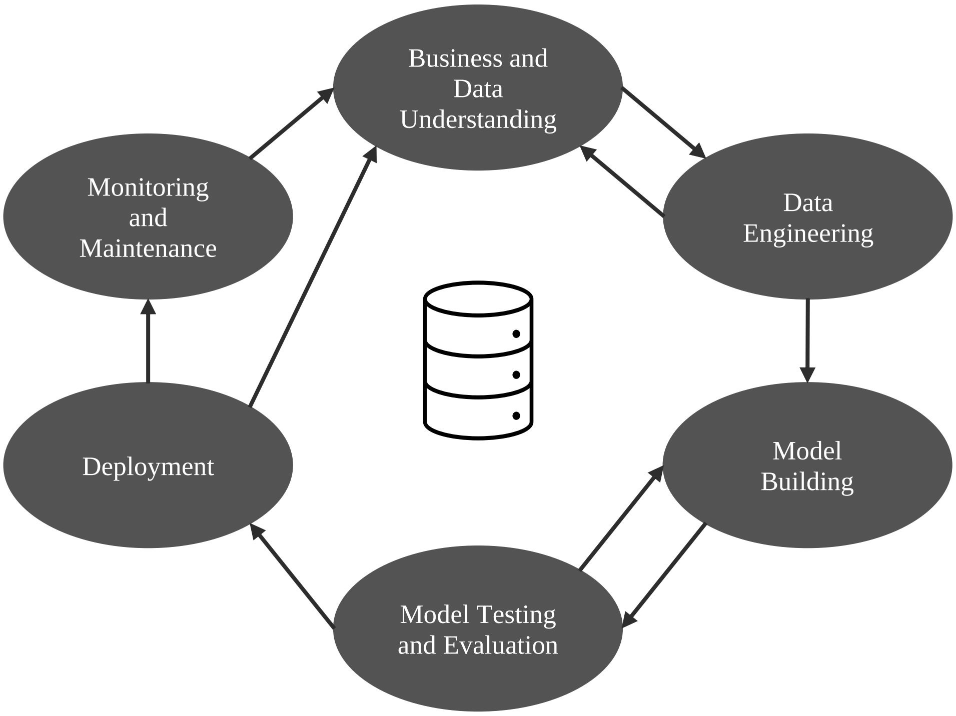

Figure 2: CRISP-ML (Q) Process, Source: [6]

aggregated decisions of the models. Techniques used are boosting, bagging, and mixture of experts. The subsequent phase **Model Testing and Evaluation** is defined as the fourth step of the CRISP-ML(Q) procedure. In this phase a final evaluation of the model is performed using a test and validation data set. Finally, the models used are decided with the help of ML experts. In case of not meeting the previously defined metrics, previous steps are repeated, or the project is cancelled. **Deployment** represents step 5 and marks the application under production environment. Among others, decisions are made for the hardware use. ML related options are for example the optimization of the target hardware in terms of a Central Processing Unit (CPU) and Graphics Process Unit (GPU) availability or the optimization of the target operating system. In addition, another challenge is distinguishing the production data with the training data. Previously made assumptions can therefore no longer apply. This leads to a degradation of the model. The sixth and last step of CRISP-ML(Q) is named **Monitoring and Maintenance**. Due to the long period for the application of the ML models, the entire life cycle has to be managed. In case of a missing model follow-up, this can lead to a minimization of performance and consequently to wrong predictions. The Monitoring process is used to monitor all input data. Based on the monitoring signals, models can be updated when input data change significantly or anomalies exceed a given threshold.

3.2. Autonomy levels for digital ML tools

Within this section, the development of the maturity model is presented in more detail. As previously described, it combines the LDA with the CRISP-ML(Q). Consequently, their intersection and its transmission to the context of ML is pointed out. For the sake of using the maturity model as a ranking, several criteria for each stage are defined, which will likewise be explained. It should be noted that the model is built incrementally in that sense that a system rated on a specific level automatically fulfils *all* criteria on the previous levels.
Before elaborating on the model, it is essential to define the two terms autonomy and automation that are relevant for understanding the paper. A system can be described as autonomous when it solves complex tasks, makes decisions and reacts to unforeseen events on its own authority and without detailed programming [26]. In contrast, a system can be said to be automated when thoughtful, causal relationships are programmed into it – regardless of their scope and effort [27].

The following six steps displayed in Figure 3 provides an overview over the maturity model based on the LDA. A more detailed description is given below.

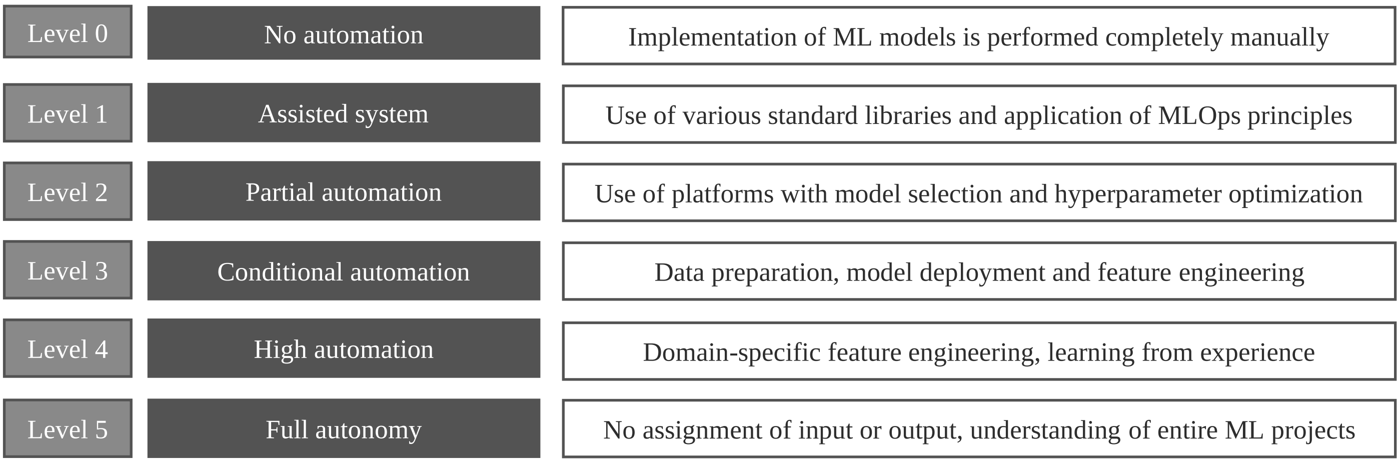

Figure 3: Six steps of the maturity model

Subsequently, each of them is explained in depth and worked through. A summary of the maturity model with respect to the single phases of CRISP-ML(Q) is given in Table 1. A step displayed in brackets has been partly included. The single levels are explained in more detail in the following.

Table 1: Summary of the maturity model with respect to respective criteria

Level	Title	Included CRISP-ML(Q) steps	Criteria for specification
0	No automation	-	-
1	Assisted system	• (Model Building) • (Model Testing and Evaluation)	1.1 Use of standardized libraries 1.2 Application of Machine Learning Operations (MLOps)

Level	Title	Included CRISP-ML(Q) steps	Criteria for specification
2	Partial automation	• (Data Engineering) • Model Building • Model Testing and Evaluation	2.1 Possibility of integrating or uploading specific data types 2.2 Automatic recognition of different data types 2.3 Automatic selection of the appropriate model 2.4 Automated splitting into training, testing and validation data sets 2.5 Automated hyperparameter tuning and ensembling
3	Conditional automation	• Data Engineering • Model Building • Model Testing and Evaluation • Deployment • (Model Monitoring and Maintenance)	3.1 Automated Feature Engineering 3.2 Automated Feature Selection and Feature Extraction 3.3 Techniques of data augmentation 3.4 Creation of a graphical user interface 3.5 Automated Deployment
4	High automation	• Data Engineering • Model Building • Model Testing and Evaluation • Deployment • Monitoring and Maintenance	4.1 Automated domain-specific feature engineering 4.2 Advanced hyperparameter optimization 4.3 Automated addition of data
5	Full autonomy	• (Business and Data Understanding) • Data Engineering • Model Building • Model Testing and Evaluation • Deployment • Monitoring and Maintenance	5.1 Automated domain knowledge 5.2 Fully communicative interaction between user and software 5.3 Automated opportunity to extend ML systems by using ML

Just as in the LDA, **level 0** of the maturity model defines a situation of **no automation**. The implementation and development of algorithms related to ML are programmed from scratch by software engineers and computer scientists. It represents the development of models on programming languages such as C++ or Java. Historically, implementing ML, without the use of standard libraries, required a high level of expertise in this area. Indeed, ML model development is an experimental and iterative process. Accordingly, no functions and steps from CRISP-ML(Q) are automated. All areas are programmed and processed manually. However, level 0 is outdated and with respect to level 1 nowadays inapplicable.

Level 1 of the maturity model is referred to as an **Assisted system** for manual programming and model testing. This means, that the actual ML application is programmed by the user, who therefore requires a high level of expertise in software development and ML. Level 1 is subdivided into two criteria. One of the main features at this level is the use of standardized libraries (criterion 1.1) on high-level programming languages, such as Python. These libraries are indispensable today in the creation of ML algorithms. In the area of data processing and ML implementation, scikit-learn [28], NumPy [29] and Pandas [30] are used. On the other hand, PyTorch and TensorFlow [31] are used in case of neural networks. In order to finally be able to visualize the results, Matplotlib [32] or seaborn [33], among others, are used to visualize the results. Criterion 1.2 includes the application of Machine Learning Operations (MLOps) [34]. This criterion shows an automation of certain manual development in the context of high-level programming languages and automation of certain functionalities. By integrating MLOps, development times are reduced as well as techniques such as testing or the deployment of large software systems are integrated, whereby increasing verifiability and reliability. With respect to CRISP-ML(Q), first automation steps take place in Model Building, to be able to implement several algorithms more efficiently.

Level 2 is called **Partial automation**. From level 2 onwards, platforms including AutoML solutions for the end-user are considered. The first criterion (2.1) is defined by the possibility of integrating or uploading specific data types. This means that clearly structured data can be uploaded, such as in a tabular. This automates the insertion of data and eliminates the need for the integration by a data scientist writing code. Criterion 2.2 builds on the previous function and defines the automatic recognition of different data types. A distinction can be made between numeric data, categorical data or time series. This function facilitates the selection of target variables for the subsequent determination of the ML algorithm. Since input is mostly numerical, data can be passed unchanged in certain situations. Categorical data, on the other hand, are divided into a finite set of classes [35]. The automatic selection of the appropriate model is presented as criterion 2.3. The selection of the model can be a challenge for ML-developments considering the No-Free-Lunch-Theorem [36]. Criterion 2.4 aims at automated splitting into training, testing and validation data sets. The automated splitting depends on the size of the data. If a certain threshold of data is exceeded, a standard division into training, test and validation data sets can be performed. In contrast, below the certain threshold different methods are used to achieve the split with a lower amount of data. These methods include, for example k-fold cross-validation and Monte Carlo cross-validation. Criterion 2.5 then encompasses automated hyperparameter tuning and ensembling. The selection of hyperparameters with respect to ML takes place before the training step and is an iterative process.

Level 3 is referred to as **Conditional automation**. Its first specifying criterion 3.1 - Automated Feature Engineering - refers to the step Data Engineering displayed in the CRISP-ML(Q) process model. The automation of data preparation is dealt with in greater detail here. Criterion 3.2 is defined as Automated Feature Selection and Feature Extraction. Here, it is of relevance to consider only the necessary features from the data and to eliminate the unusable features for the development of the subsequent ML model. Criterion 3.3 includes techniques of data augmentation, which is used to create multiple data using already existing data. For example, mathematical transformations of the existing data can be used to ensure an expansion of the data sets achieved. The creation of a graphical user interface (GUI) is presented by criterion 3.4. The GUI serves as an interaction between humans and software and can easily be operated by any end user. It displays visualizations of end results with the help of dashboards, model explanations and metrices. Additionally, by using drag-and-drop techniques or functions on the user interface, various steps for ML development can be generated without explicitly programming ML algorithms. Additional supplements through interfaces to other programming languages or a direct implementation on the platform used are also possible. The last criterion of Level 3 is Automated Deployment (Criterion 3.5). This criterion contains the automated or simplified possibility to apply the already trained model under production environment.

Level 4 of the maturity model is defined as a level with **High automation**. Automated domain-specific feature engineering (criterion 4.1) is presented as the first criterion of this stage. In practice, data is often collected and stored using various sensors. In the era of "Big Data", data is defined with high data volume, frequency and challenges in terms of processing speed [37]. Data sets generated by the different sensors contain different amounts of information. The analysis and further processing are major challenges here. In addition to the demand for new software architectures and platforms, an explicit analysis of the data is also necessary to be able to pass it on to the model. Criterion 4.1 contains the automated function to analyze the data generated from a wide variety of sensors. This includes an examination of the data information and the decision, i.e., to merge data. A simple integration and aggregation of different data sources into one file can be used for the modelling process of ML. Thereby a deep basic understanding of data is required, since understanding the data often requires expert knowledge. The next criterion of this level is the advanced hyperparameter optimization (criterion 4.2). This includes a sophisticated function for automatic determination of the hyperparameters. The fulfilment of this criterion requires the ability of a system to remember past ML applications and to apply this experience to new problems, i.e., by making use of so-called meta-learning [38]. A system at this level should continuously learn from new data and retain knowledge from previously used data. The automated addition of data specifies criterion 4.3. The goal is to be able to automatically add new data based on the understanding of the data.

Level 5 represents the goal of **full autonomy** of all six steps of the CRISP-ML(Q) process model. The highest level of automation is achieved and, at the same time, requires the least or no manual effort. Thus, the use of ML implementation solutions would be possible without detailed knowledge of the functionalities of ML. Criterion 5.1 describes the goal of automated domain knowledge. This means that the first step of CRISP-ML(Q) can be solved autonomously in the last stage of the evaluation system. Thus, an automatic recognition of business problems with respect to the company goals is achieved. The quantification of the potential business value in relation to ML projects displays a challenging process and is still solved with the long-term experience of domain experts. In addition to the business stakeholders, domain experts and data scientists can also result, as well as critically analyze the achievement of business goals. With criterion 5.2, a fully communicative interaction between user and software is achieved. On the one hand, it means that no explicit specifications are necessary for the creation of an ML solution. On the other hand, however, a presentation of the prediction or the solution is necessary. Another advantage is the use of an ML application by non-technical users. In addition, ML experts can also use criterion 5.2 to formulate highly complex problems and issues using conversational interaction. The last criterion 5.3 describes the automated opportunity to extend ML systems by using ML. This approach requires further research in unsupervised learning or transfer learning. Criterion 5.3 is accompanied by numerous issues, as an ML problem requires a high number of parameters and data to achieve optimal performance. This notion for Level 5 points to a future scenario with the ideology towards fully automated ML application. The achievement of the last level requires an interdisciplinary research approach involving multiple disciplines of computer science and engineering sciences. Among them are the development of human-computer interactions, automated data generation with its processing, and advanced data science.

4. Evaluation of the model

For assessing the applicability and its contribution to the research landscape, two steps were taken. First, an evaluation in terms of applicability was conducted. For this, semi-structured expert interviews with n = 4 experts working in ML research and data science in industry were held. They were asked the following questions:

1. How do you assess the logic and reasonableness of the criteria for the application of AutoML?
2. In view of the CRISP-ML(Q) procedure model, have the functions been correctly assigned according to their degree of difficulty of automation?
3. How do you evaluate the logic for deriving the evaluation system, in that sense that it was inspired through a classification scheme and specified with the help of CRISP-ML(Q)?

4. How do you assess the applicability of the evaluation system to be used for the evaluation of existing platforms and thus to reflect the current state of the art as well as future fields of research?

All experts describe the maturity model to be comprehensible and logical, thus appropriate as a guideline for assessing the autonomy of ML tools. The CRISP-ML(Q) was the right choice for deriving the model. For motivation and analogy, a maturity model on autonomy levels was entitled to be useful, but also clear differences need to be considered. As such, whereas the goals and ideas for driving autonomy have been clearly defined, the definition of full autonomy for ML implementation solutions is more complex. One expert highlighted the opportunity for additional examination of the single levels, e.g., by counter-questions. In detail, it could be checked how much additional programming tasks are necessary at a specific level to achieve the respective level. Additionally, one expert outlined that a further division of the categories and their in-depth concretization would be a possible future research direction. Despite the increasing development of corresponding platforms, the need for domain expertise to use given platforms was marked by two experts. In conclusion, the expert interviews revealed that the maturity model can be seen as useful for evaluating ML tools. Consequently, users might make a selection of a given platform regarding the application.

Second, the proposed maturity model was applied to the six commercial tools KNIME [2], Google AutoML [10], Azure ML [39], WEKA [4], Data Robot [3] and Orange [40]. This application demonstrates the model's usefulness in categorizing digital ML platforms and guiding future development in the field. Table 2 displays a summary of the single ratings. It must be noted that the authors of this publication did not test the actual assistance systems but solely concentrated on the publicly available descriptions. Besides, as evident, only available information that is explicitly described can be considered for the classification. In return, non-existing information needed to be identified by reading between the lines. As evident in Table 1, level 3 – Conditional autonomy – displays the current state of development. Indeed, most tools fall into this category.

Table 2: Evaluation of existing commercial tools

Criterion	**Short description**	**KNIME**	**Google AutoML**	**Azure ML**	**WEKA**	**Data Robot**	**Orange**
1.1	Use of standardized libraries	✓	✓	✓	✓	✓	✓
1.2	Application of MLOps	✓	✓	✓	✓	✓	✓
2.1	Possibility of integrating or uploading specific data types	✓	✓	✓	✓	✓	✓
2.2	Automatic recognition of different data types	✓	✓	✓	✓	✓	✓
2.3	Automatic selection of an appropriate model	✓	✓	✓	✓	✓	✗
2.4	Automated splitting into training, testing and validation data sets	✓	✓	✓	✓	✓	✓
2.5	Automated hyperparameter tuning and ensembling	✓	✓	✓	✓	✓	✗
3.1	Automated Feature Engineering	✓	✓	✓	✓	✓	✗
3.2	Automated Feature Selection and Feature Extraction	✓	✓	✓	✓	✓	✗
3.3	Techniques of data augmentation	✓	✓	✓	✓	✓	✗
3.4	Inclusion of a GUI	✓	✓	✓	✓	✓	✓
3.5	Automated Deployment	✓	✓	✓	✗	✓	✗
4.1	Automated domain-specific feature engineering	✗	✗	✗	✗	✗	✗
4.2	Advanced hyperparameter optimization	✗	✗	✗	✗	✗	✗
4.3	Automated addition of data	✗	✗	✗	✗	✗	✗
5.1	Automated domain knowledge	✗	✗	✗	✗	✗	✗
5.2	Fully communicative interaction between user and software	✗	✗	✗	✗	✗	✗
5.3	Automated opportunity to extend ML	✗	✗	✗	✗	✗	✗
Assigned Level		**3**	**3**	**3**	**2**	**3**	**1**

5. Conclusion and outlook

The present publication describes a maturity model aiming to assess the autonomy level of existing digital ML tools that follow the goal to simplify the implementation process of ML applications by domain experts in manufacturing environments. It is based on the levels of driving automation and concretized by the CRISP-ML (Q) process. Inspired by the LDA, the model is divided into six steps ranging from no automation to full automation. Whereas level 0 is outdated and, level 5 remains a vision for the future. The conducted evaluation demonstrates that level 3 can be denoted as state of the art where most progress is performed on. The result of the paper aims at researchers investigating this field and who want to rate newly developed systems against the state of the art. Reliability in the context of the procedure is ensured on the one hand by the systematic derivation of the system based on a common classification scheme and a procedure model and on the other hand by the orientation towards existing and in literature published models.

The maturity model leaves space for future extensions. As such, it can be elaborated by the human intervention. The intervention should consequently decrease with increasing level. A systematic integration into the described model could not be carried out. Furthermore, it might be necessary to carry out a further division or concretization of the single levels as well as conduct more interviews to underline the findings described above. Although the model was developed for the manufacturing industry, it might be transferable to other domains. Yet, an evaluation of a potential transfer was not tested as it was out of the scope of this paper. As described above, the experts surveyed deal with this topic in depth and can therefore provide profound answers. Nevertheless, the sample of four experts was small. More in-depth insights could therefore be gained by expanding the evaluation. Lastly, in next works a further evaluation through the quantitative testing of the platforms can be executed as an extension of the evaluation based on the tool descriptions. Hence, a proper testing of the platforms is necessary to underline the rating described in chapter 4.

References

[1] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, On-Road Automated Driving (ORAD) committee, 400 Commonwealth Drive, Warrendale, PA, United States, 2021.

[2] M. R. Berthold *et al.*, "KNIME - the Konstanz information miner," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 26–31, 2009, doi: 10.1145/1656274.1656280.

[3] DataRobot, *DataRobot AI Platform.* [Online]. Available: https://www.datarobot.com/platform/ (accessed: Apr. 12 2023).

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

[5] H. Hagras, "Toward Human-Understandable, Explainable AI," *Computer*, vol. 51, no. 9, pp. 28–36, 2018, doi: 10.1109/MC.2018.3620965.

[6] S. Studer *et al.*, "Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology," *MAKE*, vol. 3, no. 2, pp. 392–413, 2021, doi: 10.3390/make3020020.

[7] K. M. Lee, J. Yoo, S.-W. Kim, J.-H. Lee, and J. Hong, "Autonomic machine learning platform," *International Journal of Information Management*, vol. 49, pp. 491–501, 2019, doi: 10.1016/j.ijinfomgt.2019.07.003.

[8] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Auto-sklearn: Efficient and Robust Automated Machine Learning," in *Springer eBooks Computer Science, Automated Machine Learning: Methods, Systems, Challenges*, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., Cham: Springer, 2019, pp. 113–134.

[9] J. Waring, C. Lindvall, and R. Umeton, "Automated machine learning: Review of the state-of-the-art and opportunities for healthcare," *Artificial Intelligence in Medicine*, vol. 104, p. 101822, 2020, doi: 10.1016/j.artmed.2020.101822.

[10] Google Cloud, *Google Cloud AutoML.* [Online]. Available: https://cloud.google.com/automl (accessed: Apr. 12 2023).

[11] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Automated Machine Learning: Methods, Systems, Challenges*. Cham: Springer, 2019.

[12] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-WEKA: Automatic Model Selection and Hyperparameter Optimization in WEKA," in *Springer eBooks Computer Science, Automated Machine Learning: Methods, Systems, Challenges*, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., Cham: Springer, 2019, pp. 81–95.

[13] R. S. Olson and J. H. Moore, "TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning," in *Springer eBooks Computer Science, Automated Machine Learning: Methods, Systems, Challenges*, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., Cham: Springer, 2019, pp. 151–160.

[14] C. Liu, F. Tang, Y. Hu, K. Li, Z. Tang, and K. Li, "Distributed Task Migration Optimization in MEC by Extending Multi-Agent Deep Reinforcement Learning Approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1603–1614, 2021, doi: 10.1109/TPDS.2020.3046737.

[15] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein, "Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud," *Proc. VLDB Endow.*, vol. 5, no. 8, pp. 716–727, 2012, doi: 10.14778/2212351.2212354.

[16] M. Boehm *et al.*, "SystemML: Declarative machine learning on spark," *Proc. VLDB Endow.*, vol. 9, no. 13, pp. 1425–1436, 2016, doi: 10.14778/3007263.3007279.

[17] C. Jermain, *Large Scale Machine Learning with the SimSQL System.* [Online]. Available: https://cmj4.web.rice.edu/SimSQLNew.pdf (accessed: Jun. 1 2023).

[18] T. Kraska, A. Talkwalkar, J. Duchi, R. Griffith, M. J. Franklin, and M. Jordan, "MLbase: A Distributed Machine-learning System," in *6th Biennial Conference on Innovative Data Systems Research*, Asilomar, California, USA, 2013.
[19] G. Malewicz *et al.*, "Pregel: A system for large-scale graph processing," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, Indianapolis Indiana USA, 2010, pp. 135–146.
[20] E. Shaikh, I. Mohiuddin, Y. Alufaisan, and I. Nahvi, "Apache Spark: A Big Data Processing Engine," in *Communications for sustainable development: IEEE MENACOMM'19 : 2nd IEEE Middle East & North Africa Communications Conference : 2019, 19-21 November, the Diplomat Radisson Blu Hotel, Kingdom of Bahrain*, Manama, Bahrain, 2019, pp. 1–6.
[21] Y. Yu *et al.*, "DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language," in *Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation*, San Diego, USA, 2008.
[22] M. Hofmann, *RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Hoboken: Taylor and Francis, 2013.
[23] S. 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, pp. 1–36, 2020.
[24] Zelros AI, *The Four Maturity Levels of Automated Machine Learning: Towards Domain Specific AutoML.* [Online]. Available: https://zelros.medium.com/the-four-maturity-levels-of-automated-machine-learning-towards-domain-specific-automl-d6e6cfdbbcf1 (accessed: May 9 2023).
[25] G. Schuh, R. Anderl, J. Gausemeier, M. ten Hompel, and W. Wahlster, *Industrie 4.0 Maturity Index: Die digitale Transformation von Unternehmen gestalten*. München: utzverlag, 2017. [Online]. Available: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=6898389
[26] EFI - Expertenkommission Forschung und Innovation, *Gutachten zu Forschung, Innovation und technologischer Leistungsfähigkeit Deutschlands 2018*. Berlin, 2018. Accessed: Jul. 19 2023. [Online]. Available: https://www.e-fi.de/fileadmin/Assets/Gutachten/2018/EFI_Gutachten_2018.pdf
[27] M. P. Groover, *Fundamentals of modern manufacturing: Materials, processes, and systems*, 4th ed. Hoboken, NJ: J. Wiley & Sons, 2011.
[28] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf
[29] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020, doi: 10.1038/s41586-020-2649-2.
[30] W. McKinney, *pandas: a Foundational Python Library for Data Analysis and Statistics.* [Online]. Available: https://www.dlr.de/sc/en/Portaldata/15/Resources/dokumente/pyhpc2011/submissions/pyhpc2011_submission_9.pdf
[31] M. Abadi *et al.*, "TensorFlow: A System for Large-Scale Machine Learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation*, Savannah, GA, USA, 2016, pp. 265–283.
[32] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
[33] M. L. Waskom, "Seaborn: statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, 2021, doi: 10.21105/joss.03021.
[34] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine Learning Operations (MLOps): Overview, Definition, and Architecture," May. 2022. [Online]. Available: https://arxiv.org/pdf/2205.02302
[35] V. Lakshmanan, S. Robinson, and M. Munn, *Machine learning design patterns: Solutions to common challenges in data preparation, model building, and MLOps*. Sebastopol, CA: O'Reilly, 2020. [Online]. Available: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=6372578
[36] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997, doi: 10.1109/4235.585893.
[37] B. Vogel-Heuser, T. Bauernhansl, and M. ten Hompel, *Handbuch Industrie 4.0: Produktion, Automatisierung und Logistik*. Wiesbaden: Springer Fachmedien Wiesbaden, 2015.
[38] X. He, K. Zhao, and X. Chu, "AutoML: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, p. 106622, 2021, doi: 10.1016/j.knosys.2020.106622.
[39] Microsoft Inc., *Automated machine learning: Automatically build machine learning models with speed and scale.* [Online]. Available: https://azure.microsoft.com/en-us/products/machine-learning/automatedml/ (accessed: Dec. 18 2023).
[40] Orange Data Mining, *Orange Data Mining Library*. [Online]. Available: https://orange3.readthedocs.io/projects/orange-data-mining-library/en/latest/ (accessed: Dec. 18 2023).